VAISALA

USER'S MANUAL

Digital IF Receiver / Doppler Signal Processor RVP8

M211321EN-C

PUBLISHED BY

Vaisala Oyj
P.O. Box 26
FI-00421 Helsinki
Finland

Phone (int.): +358 9 8949 1
Fax: +358 9 8949 2227

Visit our Internet pages at www.vaisala.com

Table of Contents

List of Figures

List of Tables

CHAPTER 1

GENERAL INFORMATION

1.1 Hardware Limited Warranty

Vaisala, Inc. warrants its IRIS hardware (RVP8 and RCP8) to function according to the hardware User's Manual documentation for a period of one year following delivery. In the event of a failure during the warranty period, the customer should notify Vaisala to obtain a Return Authorization. Upon receiving the Return Authorization from Vaisala, the customer ships the failed unit by pre-paid freight. Vaisala, at its option, will repair or replace the defective unit within 30 days and return the unit to the customer.

Damage caused by fire, flood, lightning, or other catastrophe, and damage caused by misuse or abuse are not covered by this warranty.

In no event shall Vaisala, Inc. be liable for any direct, indirect, special, incidental, or consequential damages arising out of the use or inability to use the hardware or documentation provided by Vaisala, Inc. Vaisala, Inc. makes no warranty, either express or implied, with respect to any of the hardware or documentation, as to the quality, performance, merchantability, or fitness for a particular purpose.

1.2 Preface

This manual provides technical information on the RVP8 digital receiver and Doppler signal processor.

1.2.1 About This Manual

This manual is used primarily by engineers for installation and troubleshooting, or by users interested in understanding the signal processing features, algorithms, and control and data formats.

Chapter 2, Introduction and Specifications, on page 17, describes the major features of the RVP8 signal processor and gives its technical specifications.

Chapter 3, Hardware Installation, on page 65, discusses the electrical issues involved with installing the RVP8 processor and IFD receiver module. This includes power supply connections, radar analog and digital signal interfaces and computer interface connections. Software installation is covered in a separate Appendix.

Chapter 4, TTY Nonvolatile Setups, on page 113, continues the installation discussion by describing how to use the local TTY to configure the actual operation of the RVP8. This includes a detailed description of the (approximately one hundred) setup parameters that affect the operation of the RVP8.

Chapter 5, Plot-Assisted Setups, on page 159, completes the installation discussion by using the oscilloscope plotting modes to configure and align the radar receiver, and measure its performance.

Chapter 6, Processing Algorithms, on page 199, gives mathematical descriptions of the processing algorithms implemented in the RVP8 signal processor. This information can be useful to those writing their own interface to the RVP8, or for those who want to learn more about the internal workings of the signal processor.

Chapter 7, Host Computer Commands, on page 275, contains a description of the digital commands that the host computer must use to set up and control the RVP8 processor. The introductory section discusses processor I/O in general, and gives an overview of how to set up the RVP8 for recording data. Each command is then detailed in subsequent sections.

The appendixes give information on the RVP8 standard chassis, installation testing, dual-polarization options, time series recording, and developer's details.

1.2.2 Version Information

Manual Code	Description
M211321EN-C	This manual. Third version. November 2013
M211321EN-B	Previous manual. Second version. March 2013
M211321EN-A	Previous manual. First version.

1.2.3 Related Manuals

Manual Code	Manual Name
M211315EN	Software Installation Manual
M211316EN	IRIS and RDA Utilities Manual
M211317EN	IRIS Radar Manual
M211318EN	IRIS Programmer's Manual
M211319EN	IRIS Product and Display Manual
M211320EN	RCP8 User's Manual
M211322EN	RVP900 User's Manual
M211452EN	IRIS and RDA Dual Polarization User's Manual

You can download the latest versions of the manuals from Vaisala product website, http://www.vaisala.com. They can be read online using by Adobe® Reader®, which is installed with IRIS.

Vaisala encourages you to send your comments and/or corrections to:

Vaisala Inc.
7A Lyberty Way
Westford, MA 01886
email: helpdesk@vaisala.com

1.2.4 Documentation Conventions

Throughout the manual, important safety considerations are highlighted as follows:

WARNING	Warning alerts you to a serious hazard. If you do not read and follow instructions very carefully at this point, there is a risk of injury or even death.

CAUTION	Caution warns you of a potential hazard. If you do not read and follow instructions carefully at this point, the product could be damaged or important data could be lost.

NOTE	Note highlights important information on using the product.

CHAPTER 2
INTRODUCTION AND SPECIFICATIONS

The RVP8 Lineage

Vaisala has a 20–year history of supplying innovative, high-quality signal processing products to the weather radar community. The history of Vaisala products reads like a history of weather radar signal processing:

Year	Model	Units Sold	Major Technical Milestones
1981	FFT	10	First commercial FFT-based Doppler signal processor for weather radar applications. Featured Simultaneous Doppler and intensity processing.
1985	RVP5	161	First single-board low-cost Doppler signal processor. First commercial application of dual PRF velocity unfolding algorithm.
1986	PP02	12	First high-performance commercial pulse pair processor with 18.75-m bin spacing and 1024 bins.
1992	RVP6	150	First commercial floating-point DSP-chip based processor. First commercial processor to implement selectable pulse pair, FFT or random phase 2nd trip echo filtering.
1996	RVP7	>200	First commercial processor to implement fully digital IF processing for weather radar.
2003	RVP8		First digital receiver/signal processor to be implemented using an open hardware and software architecture on standard PC hardware under the Linux operating system. Public API's are provided so that customers may implement their own custom processing algorithms.

Much of the proven, tested, documented software from the highly-successful RVP7 (written in C) is ported directly to the new RVP8 architecture. This allows Vaisala to reduce time-to-market and produce a high-quality, reliable system from day one. However, the new RVP8 is not simply a re-hosting of the RVP7. The RVP8 provides new capabilities for weather radar systems that, until now, were not available outside of the research community.

Advanced Digital Transmitter Option

For example, the RVP8 takes the next logical step after a digital receiver- a digitally synthesized IF transmit waveform output that is mixed with the STALO to provide the RF waveform to the transmitter amplifier (for example, Klystron or TWT). The optional RVP8/Tx card opens the door for advanced processing algorithms such as pulse compression, frequency agility and phase agility that were not possible before, or done in more costly ways.

Open Hardware and Software Design

Compared to previous processors that were built around proprietary DSP chips, perhaps the most innovative aspect of the RVP8 is that it is implemented on standard PC hardware and software that can be purchased from a wide variety of sources. The Intel Pentium/PCI approach promises continued improvement in processor speed, bus bandwidth and the availability of low–cost compatible hardware and peripherals. The performance of an entry level RVP8 (currently dual 2.4 GHz Pentium processors) is 6 times faster than the fastest RVP7 ever produced (with two RVP7/AUX boards).

Aside from the open hardware approach, the RVP8 has an open software approach as well. The RVP8 runs in the context of the Linux operating system. The code is structured and public API's are provided so that research customers can modify/replace existing Vaisala algorithms, or write their own software from scratch using the RVP8 software structure as a foundation on which to build.

The advantage of the open hardware and software PCI approach is reduced cost and the ability for customers to maintain, upgrade and expand the processor in the future by purchasing standard, low cost PC components from local sources.

SoftPlane High–Speed I/O Interconnect

There are potentially many different I/O signals emanating from the backpanel of the RVP8. Most of these conform to well-known electrical and protocol standards (VGA, SCSI, 10–BaseT, RS-232 Serial, PS/2 Keyboard, etc.), and can be driven by standard commercial boards that are available from multiple vendors. However, there are other interface signals such as triggers and clocks that require careful timing. These precise signals cannot tolerate the PCI bus latency. For signals that have medium–speed requirements (–1 microsec latency) for which the PCI bus is inappropriate; and others that require a high–speed (– 1 ns latency) connection that can only be achieved with a dedicated wire, the RVP8 Softplane™ provides the solution.

Physically, the Softplane™ is a 16-wire digital "daisy-chain" bus that plugs into the tops of the RVP8/Rx, RVP8/Tx, and I/O boards. The wires connect to the FPGA chips on each card, and the function of each wire is assigned at run–time based on the connectivity needs of the overall system. The Softplane™ allocates a dedicated wire to carry each high-speed signal; but groups of medium-speed signals are multiplexed onto single wires in order to conserve resources. Even though there are only 16 wires available, the Softplane is able to carry several high-speed signals and hundreds of medium–speed signals, as long as the total bandwidth does not exceed about 600MBits/sec.

The Softplane™ I/O is configured at run–time based on a file description rather than custom wiring such as wirewrap. Neither the PCI ackplane nor the physical Softplane™ are customized in any way. Since there is no custom wiring, a failed board can be replaced with a generic off–the–shelf spare, and that spare will automatically resume whatever functions had been assigned to the original board. Similarly, if the chassis itself were to fail, then simply plugging the boards into another generic chassis would restore complete operation. Cards and chassis can be swapped between systems without needing to worry about custom wiring.

Standard LAN Interconnection for Data Transfer or Parallel Processing

For communication with the outside world, the RVP8 supports as standard a 10/100/1000 Base T Ethernet. For most applications, the 100 BaseT Ethernet is used to transfer moment results (Z, T, V, W) to the applications host computer (for example, a product generator). However, the gigabit Ethernet is sufficiently fast to allow UDP broadcast of the I and Q values for the purpose of archiving and/or parallel processing. In other words, a completely separate signal processor can ingest and process the I and Q values generated by the RVP8.

2.1 System Configuration Concepts

The hardware building blocks of an RVP8 system are actually quite few in number:

- **RVP8/IFD™ IF Digitizer Unit**- This is a separate sealed unit usually mounted in the receiver cabinet. The primary input to the IFD is the received IF signal. In addition, the IFD has channels to sample the transmit pulse and to take in an external clock to phase lock the A/D conversion with the transmit pulse (not used for magnetron systems).
- **RVP8/Rx™ Card**- A PCI card mounted in the chassis. It connects to the IFD by a CAT-5E cable which can be up to 25m long. In addition, there are two BNC trigger outputs and four RS-422 programmable I/O signals.
- **I/O-62™ Card and Connector Panel**- These handle all of the various I/O associated with a radar signal processor, such as triggers, antenna angles, polarization switch controls, pulse width control, etc. The Connector Panel is mounted on either the front or rear of the equipment rack and a cable (supplied) connects the panel to the I/O-62.
- **Optional RVP8/Tx™ card**- This supplies two IF output signals with programmable frequency, phase and amplitude modulation. In the simplest case it might merely supply the COHO which is mixed with the STALO to generate the transmit RF for Klystron or TWT systems. More interesting applications include pulse compression and frequency agility scanning. This card is not necessary for magnetron systems.
- **PC Chassis and Processor with various peripherals**- a robust 4U rack mount unit with a dual-Xeon mother board, diagnostic front panel display, disk (mechanical or flash), CDRW, keyboard, mouse and optional monitor for local diagnostic work. Redundant power supplies are used, and there are redundant fans as well.

This modular hardware approach allows the various components to be mixed and matched to support applications ranging from a simple magnetron system to an advanced dual polarization system with pulse compression. Typically Vaisala supplies turn-key systems, although some OEM customers who produce many systems purchase individual components and integrate them by themselves. This allows OEM customers to put their own custom "stamp" on the processor and even their own custom software if they so choose.

For the turnkey systems provided by Vaisala the basic chassis is a 6U rack mount unit as described above. A 2U chassis can be provided for applications for which space is limited. A very low cost approach is to use

a desk side PC, but this is not recommended for applications that require long periods of unattended operation.

To illustrate various RVP8 configurations, some typical examples are shown below. For clarity, all the examples show the single–board computer approach. A mother board approach is equivalent.

Example 1: Basic Magnetron System

The building blocks required to construct the basic system are:

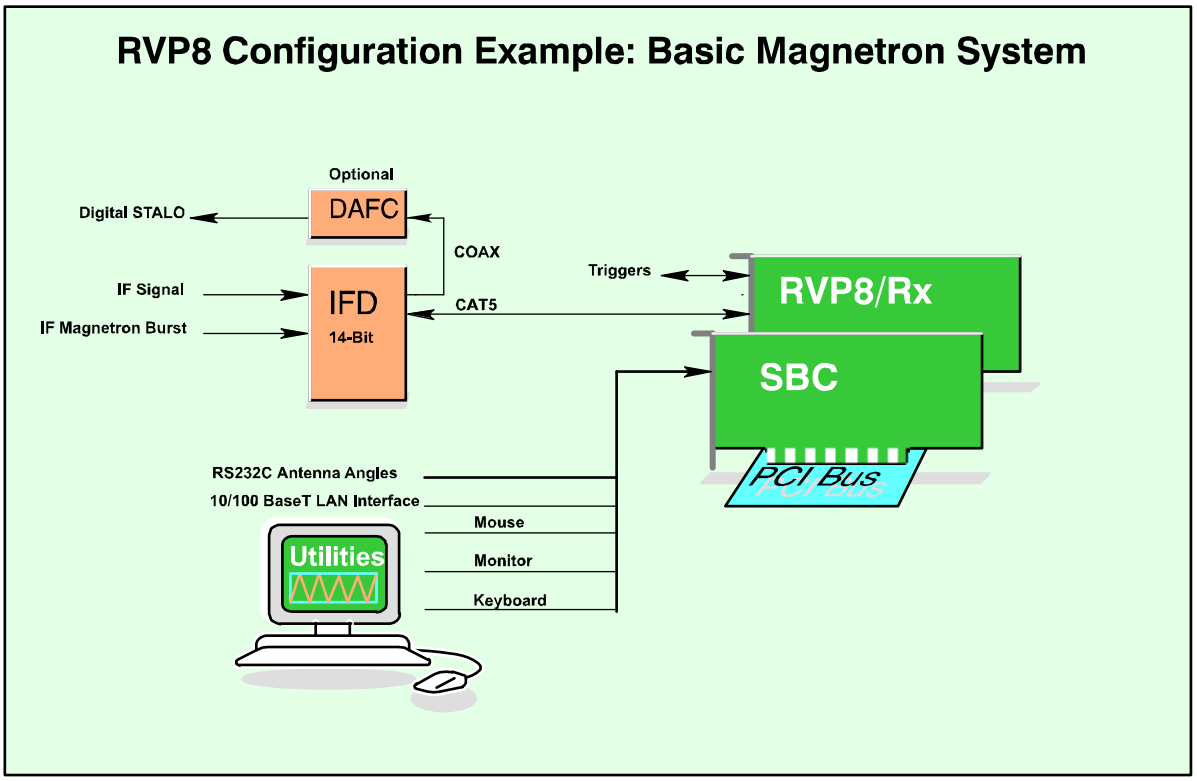


Figure 1 RVP8 Configuration Example: Basic Magnetron System

- **IFD**- IF Digitizer installed in the radar receiver cabinet. This can be located up to 100 meters from the RVP8 main chassis (fiber optic connection). The DAFC (Digital AFC) is an option to interface to a digitally controlled STALO. Like the RVP7, the RVP8 provides full AFC with burst pulse auto-tracking.
- **RVP8/Rx**- The digital receiver collects digitized samples from the IFD and does the processing to obtain I/Q. It also provides two trigger connections configurable for input or output.
- **SBC Card**- Single Board Computer with dual SMP processors (PC) running Linux.

The figure above shows a basic magnetron system constructed with an IFD, and two PCI cards. A standard RS-232 serial input (included with the SBC) is used for obtaining the antenna angles and the output/input trigger is provided directly from the Rx card. This system has 5 times the processing power of the fastest version of the previous generation processor (RVP7/Main board plus 2 RVP7/AUX boards) so that it is capable of performing DFT processing in 2048 rangebins with advanced algorithms such as random phase 2nd trip echo filtering and recovery.

Example 2: Klystron System with Digital Tx

In this case, the IFD can receive a master clock from the radar system (for example, the COHO). This ensures that the entire system is phase locked. As compared to the previous example there are two additional cards shown in this example:

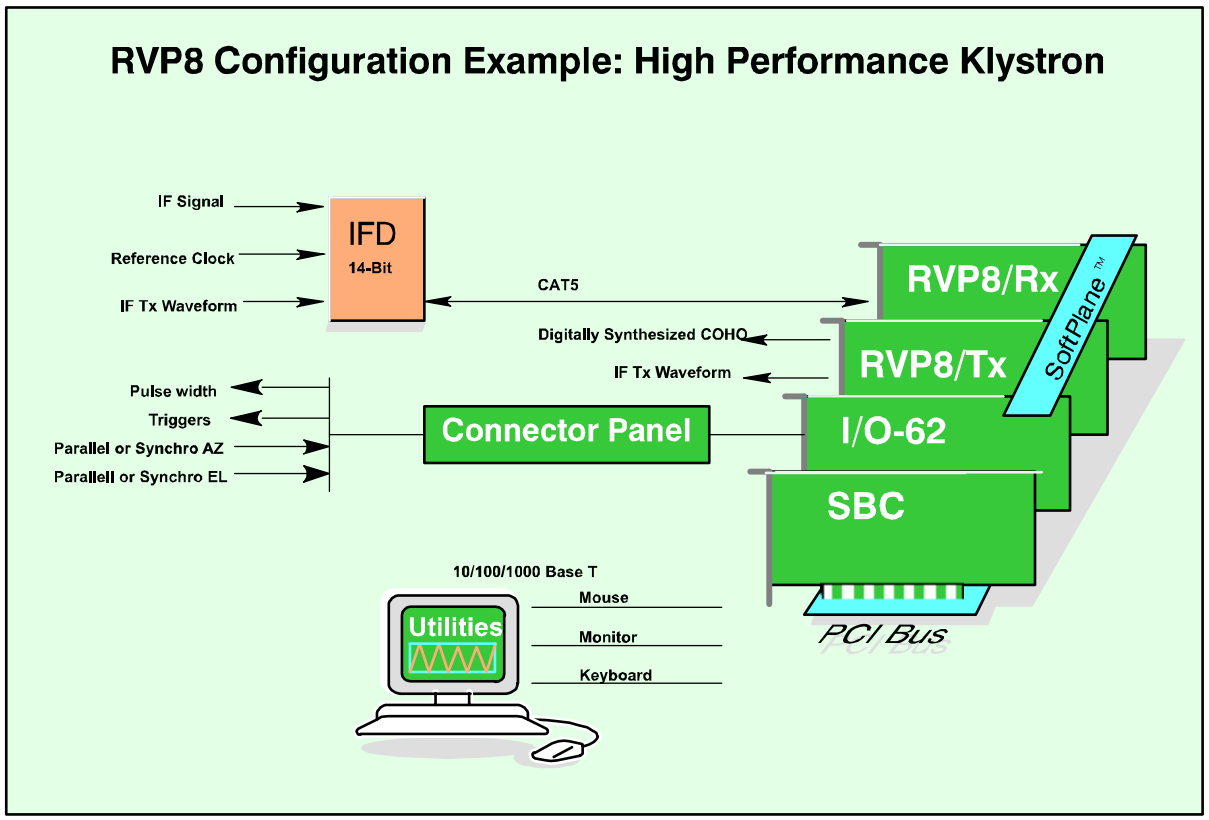


Figure 2 RVP8 Configuration Example: High Performance Klystron

- **RVP8/Tx-** The digital transmitter card provides the digital Tx waveform. A second output can be used to provide a COHO in the event that the RVP8 is used to provide the system master clock. In any case, the IF transit waveform and the A/D sampling are phase locked.
- **Vaisala I/O-62** card for additional triggers, parallel, synchro or encoder AZ and EL angle inputs, pulse width control, spot blanking control output, etc. These signals are brought in via the connector panel.

The figure shows the Vaisala SoftPlane™ which carries time-critical I/O such as clock and trigger information which is not appropriate for the PCI bus. These signals are limited to the cards provided by Vaisala, that is, the SoftPlane™ is not connected to any of the standard commercial cards.

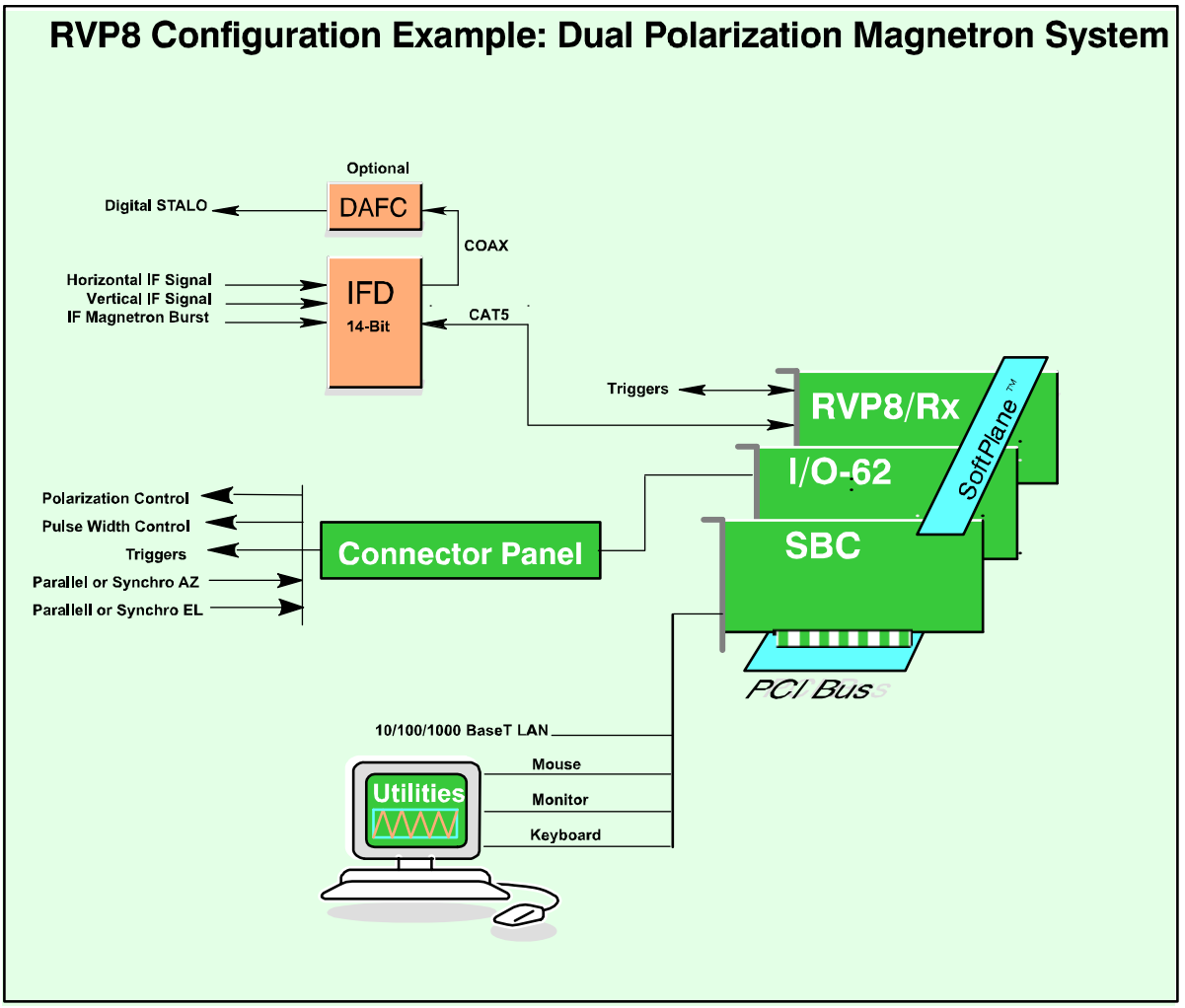


Figure 3 RVP8 Configuration Example: Dual Polarization Magnetron System

Example 3: Dual Polarization Magnetron System

In this system 2 IFD's and two RVP8/Rx cards are used for the horizontal and vertical channels of a dual-channel receiver. The legacy RVP7 technique of using a single IFD and two IF frequencies for the horizontal and vertical channels (for example, 24 and 30 MHz) is also supported by the RVP8. In the case of either dual or single IFD's, there is a synch clock provided by either the STALO reference frequency (for example, 10 MHz) or by the RVP8 itself.

The RVP8 supports calculation of the complete covariance matrix for dual pol, including ZDR, PHIDP (KDP), RHOHV, LDR, etc. Which of these variables is available depends on whether the system is a single–channel switching system (alternate H and V), a STAR system (simultaneous transmit and receive) or a dual channel switching system (co and cross receivers). Note that for the special case of a single channel switching system, only one IFD is required.

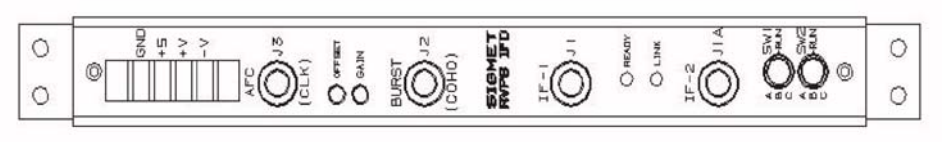
COTS Accessories

Aside from the basic PCI cards required for the radar application, there are additional cards that can be installed to meet different customer requirements, for example,

- 10/100–BaseT Ethernet card for additional network I/O (for example, a backup network).
- RS-232/RS-422 serial cards for serial angles, remote TTY control, etc.
- Sound card to synthesize audio waveforms for wind profiler applications.
- GPS card for time synch.
- IEEE 488 GPIB card for control of test equipment.

The bottom line is that the PCI open hardware approach provides unparalleled hardware flexibility. In addition, the availability of compatible low-cost replacement or upgrade parts is assured for years into the future.

2.1.1 IFD IF Digitizer



The IFD 14-bit IF digitizer is a totally sealed unit for optimum low-noise performance. The use of digital components within the IFD is minimized and the unit is carefully grounded and shielded to make the cleanest possible digital capture of the input IF signal. Because of this, the IFD achieves the theoretical minimum noise level for the A/D convertors.

There are 4 inputs to the IFD:

- IF video signal.
- A secondary IF video signal, used for dual polarization or very wide dynamic range applications.
- IF Burst Pulse for magnetron or IF COHO for Klystron.
- Optional reference clock for system synchronization. For a Klystron system, the COHO can be input. Magnetron systems do not require this signal. This clock can even come from the RVP8/Tx card itself.

All of these inputs are on SMA connectors. The IF signal input is made immediately after the STALO mixing/sideband filtering step of the receiver where a traditional log receiver would normally be installed. The required signal level for both the IF signal and burst is +6.5 dBm for the strongest expected input signal. A fixed attenuator or IF amplifier may be used to adjust the signal level to be in this range.

Digitizing is performed for both the IF signal and burst/COHO channels at approximately 72 MHz to 14-bits. This provides 92 to 105 dB of dynamic range (depending on pulse width) without using complex AGC, dual A/D ranging or down mixing to a lower IF frequency.

All communication to the main RVP8 chassis goes over a special CAT5E type cable. The major volume of data is the raw time series samples sent down to the RVP8 Rx card. Coming back up is trigger timing and AFC information to the IFD.

The RVP8 provides comprehensive AFC support for tuning the STALO of a magnetron system. Alternatively, the magnetron itself can be tuned by a motorized tuning circuit controlled by the RVP8. Both analog (±10V) and digital tuning (with optional DAFC to 24 bits) are supported.

2.1.2 Digital Receiver PCI Card (RVP8/Rx)

The RVP8/Rx card receives the digitized IF samples from the IFD via the fiber optic link. The advantage of this design is that the receiver electronics (LNA, RF mixer, IF preamp, and IFD) can be located as far as 100–meters away from the RVP8 main chassis. This makes it possible to choose optimum locations for both the IFD and the RVP8, for example, the IFD could be mounted on the antenna itself, and the processor box in a nearby equipment room.

The RVP8/Rx is 100% compatible with the 14-bit RVP7/IFD, but it also includes hooks for future IFD's operating at higher sampling clock rates. Two additional BNC connectors are included on the board's faceplate. These can be used for trigger input, programmable trigger output, or a simple LOG analog ascope waveform.

A remarkable amount of computing power is resident on the receiver board, in the form of an FIR filter array that can execute 6.9 billion multiply/accumulate cycles per second. These chips serve as the first stage of processing of the raw IF data samples. Their job is to perform the down–conversion, bandpass, and deconvolution steps that are required to produce (I,Q) time series. The time series data are then transferred over the PCI bus to the SBC for final processing.

The FIR filter array can buffer as much as 80 microsec of 36MHz IF samples, and then compute a pair of 2880–point dot products on those data every 0.83 microsec. This could be used to produce over-sampled (I,Q) time series having a range resolution of 125–meters and a bandwidth as narrow as 30Khz. The same computation could also yield independent 125–meter time series data from an 80 microsec compressed pulse whose transmit bandwidth was approximately 1MHz.

Finer range resolutions are also possible, down to a minimum of 25–meters. A special feature of the RVP8/Rx is that the bin spacing of the (I,Q) data can be set to any desired value between 25 and 2000 meters. Range bins are placed accurately to within +2.2 meters of any selected grid, which does not have to be an integer multiple of the sampling clock. However, when an integer multiple (N x 8.333–meters) is selected, the error in bin placement effectively drops to zero.

Dual polarization radars that are capable of simultaneous reception for both horizontal and vertical channels can be interfaced to the RVP8 using

a separate RVP8/Rx and IFD for each channel. Note that the multiplexed dual IF approach used for the RVP7 with a single IFD can also be used.

One of the primary advantages of the digital receiver approach is that wide linear dynamic range can be achieved without the need for complex AGC circuits that require both phase and amplitude calibration.

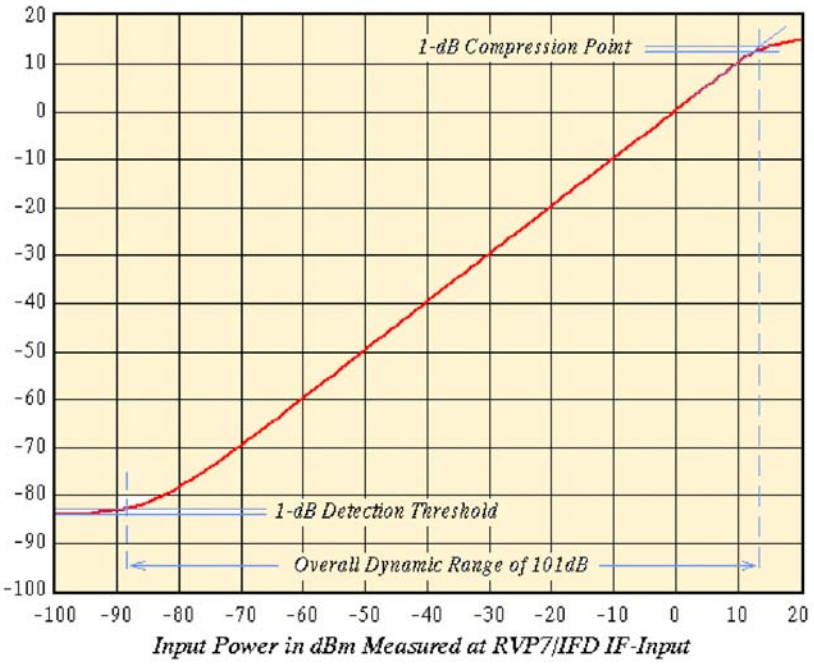


Figure 4 Calibration Plot for RVP8/IFD

Figure 4 on page 27 shows a calibration plot for a 14-bit IFD with the digital filter matched to a 2 microsecond pulse. The performance in this case is >100 dB dynamic range.

The RVP8 performs several real time signal corrections to the I/Q samples from the Rx, including:

Amplitude Correction - A running average of the transmit pulse power in the magnetron burst channel is computed in real-time by the RVP8/Rx. The individual received I/Q samples are corrected for pulse–to–pulse deviations from this average. This can substantially improve the "phase stability" of a magnetron system to improve the clutter cancelation performance to near Klystron levels.

Phase Correction - The phase of the transmit waveform is measured for each pulse (either the burst pulse for magnetron systems or the Tx Waveform for coherent systems). The I/Q values are adjusted for the actual measured phase. The coherency achievable is better than 0.1 degrees by this technique.

Large Signal Linearization - When an IF signal saturates, there is still considerable information in the signal since only the peaks are clipped. The proprietary large signal linearization algorithm used in the RVP8 provides an extra 3 to 4 dB of dynamic range by accounting for the effects of saturation.

The RVP8/Rx card provides the same comprehensive configuration and test utilities as theRVP7, with the difference that no external host computer is required to run the utilities. These utilities can be run either locally or remotely, over the network. Some examples are shown below:

Digital IF Band Pass Design Tool

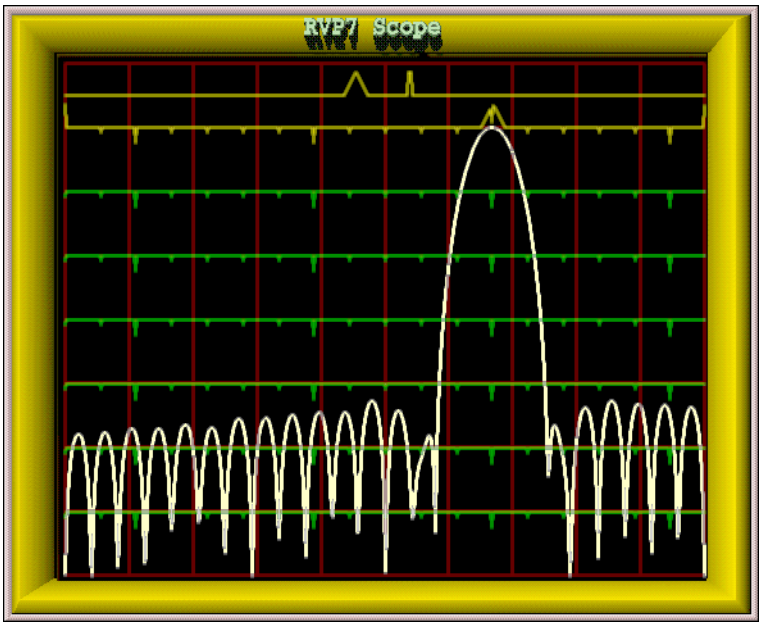


Figure 5 Digital IF Band Pass Design Tool

The built–in filter design tool makes it easy for anyone to design the optimal IF filter to match each pulse width and application. Simply specify the impulse response and pass band and the filter appears. The user interface makes it easy to widen/ narrow the filter with simple keyboard commands. There is even a command to automatically search for an optimal filter.

This display can also show the actual spectrum of the transmit burst pulse for quality control and comparison with the filter.

Burst Pulse Alignment Tool

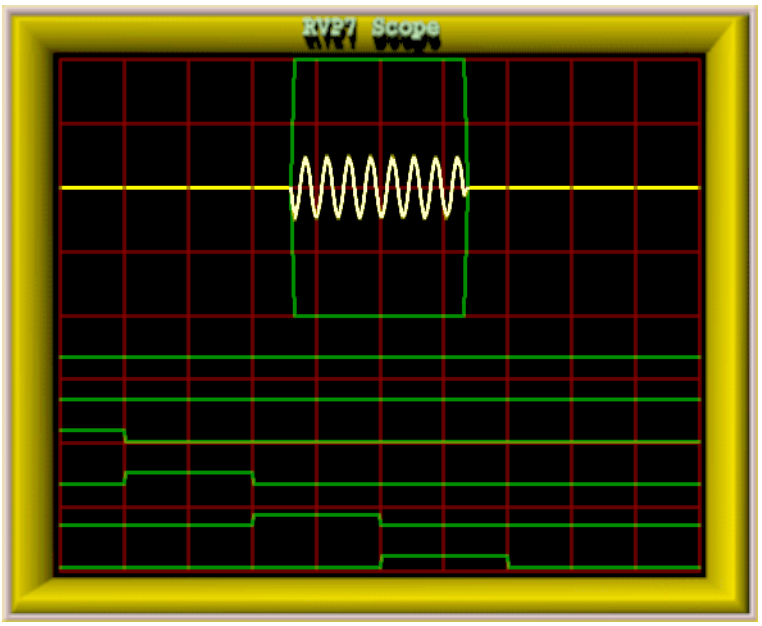


Figure 6 Burst Pulse Alignment Tool

The quality assessment of the transmit burst pulse and its precise alignment at range zero are easy to do, either manually using this tool and/or automatically using the burst pulse auto-track feature. This performs a 2D search in both time and frequency space if a validburst pulse is not detected. The automatic tracking makes the AFC robust to start–up temperature changes and pulse width changes that can effect the magnetron frequency.

AFC alignment/check is now much easier since it can be done manually from a central maintenance site or fully automatically.

Received Signal Spectrum Analysis Tool

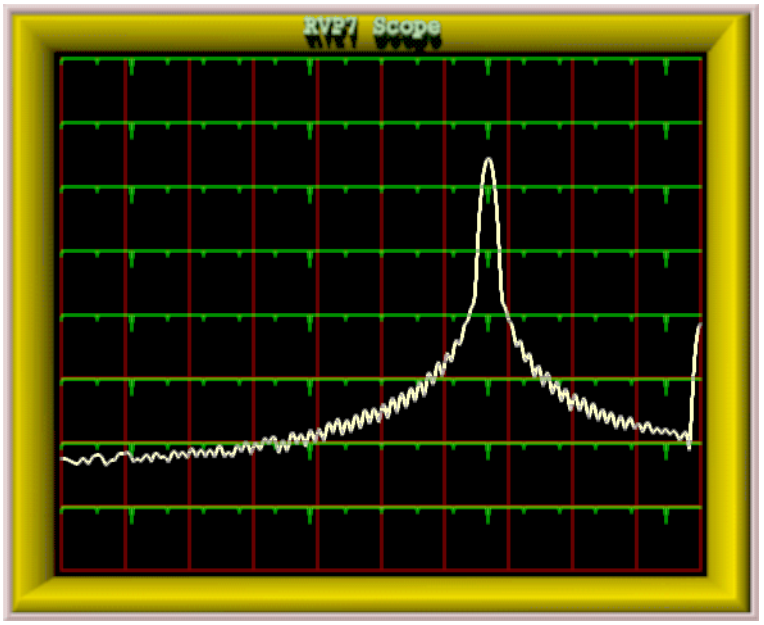
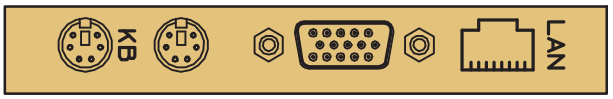


Figure 7 Receiver Signal Spectrum Analysis Tool

The RVP8 provides plots of the IF signal versus range as well as spectrum analysis of the signal as shown in this example.

In the past, these types of displays and tools required that a highly-skilled engineer transport some very expensive test equipment to the radar site. Now, detailed analysis and configuration can all be done from a central maintenance facility via the network. For a multi-radar network this results in substantial savings in equipment, time and labor.

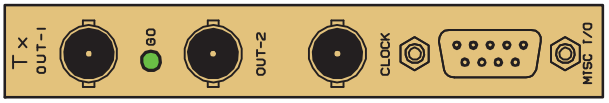
2.1.3 Mother Board or Single-Board Computer (SBC)



The dual-CPU Pentium mother board or single-board computer (SBC) acts as the host to the Linux operating system and provides all of the compute resources for processing the I/Q values that are generated by the RVP8/Rx card. Standard keyboard, mouse and monitor connections are on the Rx backpanel, along with a 10/100/1000 BaseT Ethernet port. The system does not require that a keyboard, mouse or monitor be connected which is typically the case at an unattended site. An SBC example is shown on the left. Motherboards and SBC's are available from many vendors, at various speeds Typically the SBC is equipped with 128 MB RAM. The RVP8 chassis has a front bay for either a >20 GB hard disk or a Flash Disk. The Flash Disk approach is well suited to applications where high–reliability is important. CDRW is also provided for software maintenance. Note that the latest versions of the RVP8 software and documentation can always be down-loaded from Vaisala web site for free.

The SBC also plays host for Vaisala RVP8 Utilities which provide est, configuration, control and monitoring software as well as built–in on-line documentation.

2.1.4 Digital Transmitter PCI Card (RVP8/Tx)



Many of the exciting new meteorological applications for the RVP8 are made possible by its ability to function as a digital radar transmitter. The RVP8/Tx PCI card synthesizes an output waveform that is centered at at the radar's intermediate frequency. This signal is filtered using analog components, then up–converted to RF, and finally amplified for transmission. The actual transmitter can be a solid state or vacuum tube device. The RVP8 can even correct for waveform distortion by adaptively "pre–distorting" the transmit waveform, based on the measured transmit burst sample.

The Tx card has a BNC output for the IF Tx waveform. In addition, there is a second output for an auxiliary signal or clock, or for a clock input. At the bottom of the card is a 9–pin connector for arbitrary I/O (for example, TTL, RS422, additional clock).

The RVP8 digital transmitter finds a place within the overall radar system that exactly complements the digital receiver. The receiver samples an IF waveform that has been down–converted from RF, and the transmitter synthesizes an IF waveform for up–conversion to RF. The beauty of this approach is that the RVP8 now has complete control over both halves of the radar, making possible a whole new realm of matched Tx/Rx processing algorithms. Some examples are given below:

- **Phase Modulation**- Some radar processing algorithms rely on modulating the phase of the transmitter from pulse to pulse. This is traditionally done using an external IF phase modulator that is operated by digital control lines. While this usually works well, it requires additional hardware and cabling within the radar cabinet, and the phase/amplitude characteristics may not be precise or repeatable. In contrast, the RVP8/Tx can perform precise phase modulation to any desired angle, without requiring the use of external phase shifting hardware.
- **Pulse Compression**- There is increasing demand for siting radars in urban areas that also happen to have strict regulations on transmit emissions. Often the peak transmit power is limited in these areas; so the job for the weather radar is to somehow illuminate its targets using longer pulses at lower power. The problem, of course, is that a simple long pulse lacks the ability (bandwidth) to discern targets in range. The remedy is to increase the Tx bandwidth by modulating the overall pulse envelope, so that a reasonable range resolution is restored. The exceptional fidelity of the RVP8/Tx waveform can accomplish this without introducing any of the spurious modulation components that often occur when external phase modulation hardware is used.
- **Frequency Agility**- This has been well studied within the research community, but has remained out of the reach of practical weather radars. The RVP8/Tx changes all of this, because frequency agility is as simple as changing the center frequency of the synthesized IF waveform. Many new Range/Doppler unfolding algorithms become possible when multiple transmit frequencies can coexist. Frequency agilitycan also be combined with pulse compression to remedy the blind spot at close ranges while the long pulse is being transmitted.
- **COHO synthesis**- The RVP8/Tx output waveform can be programmed to be a simple CW sine wave. It can be synthesized at any desired frequency and amplitude, and its phase is locked to the other system clocks. If you need a dedicated oscillator at some random frequency in the IF band, this is a simple way to get it.

2.1.5 I/O-62 PCI Card and I/O Panel

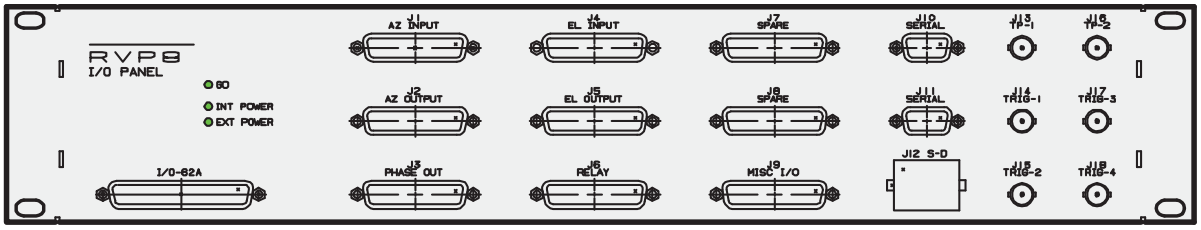
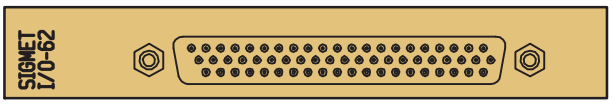


Figure 8 I/O-62 PCI Card and I/O Panel

The Vaisala I/O-62 is a short format PCI card that provides extensive I/O capabilities for the RVP8. A typical installation would have one I/O-62 and an RVP8 Connector Panel shown above. The Softplane™ is used to interconnect the I/O 62 with other Vaisala PCI cards. Note that the identical card is used in the Vaisala RCP8 radar/antenna control processor which in general does not use the Softplane™ connection. The I/O-62 has a single 62-position, high-density "D" connector. This is attached to the RVP8 Connector Panel (typically mounted on the front or back of the rack which holds the RVP8). A standard 1:1 cable connects the remote panel to the I/O-62 card in the RCP8 chassis. The standard connector panel provided by Vaisala meets the needs of most radar sites.



The best part is that the I/O-62 is configurable in software, that is, there is no need to open the chassis to configure jumpers or switches. This means that when a spare board is added, there is no need to perform hardware configuration or custom wiring.

The physical I/O lines are summarized in the system specifications section.

ESD Protection Features

Since the I/O lines are connected to the radar system, there is a potential for lightning or other ESD type damage. This is addressed aggressively by the I/O-62 in two ways:

- Every wire is protected by a **Tranzorb™** diode which transitions from an open to a full clamp between ±27 to ±35 VDC. Additionally, the Connector Panel uses **Tranzorb™** diodes on every I/O line for double protection.
- High-voltage tolerant front-end receivers/drivers are used. All components connected to the external pins can tolerate up to ±40V. For example, the TTL and wide range inputs use protectors that normally look like 100 Ohm resistors, but open at high voltage.

Run Time FPGA Configuration

The Vaisala I/O-62 card is built around a 100K–Gate FPGA which, in addition to driving the I/O signals on the 62-position connector, also coordinates the PCI and Softplane™ traffic. These chips are SRAM–based, meaning that they are configured at run time. This allows the FPGA code to be automatically upgraded during each RVP8 code release without needing to physically reprogram any parts.

The board's basic I/O services use up only 40% of the complete FPGA. The leftover space makes it possible to add smart processing right on the I/O-62 board to handle custom needs. For example the 16–bit floating–point (I,Q) data in the previous example could be reformatted into a 32–bit fixed–point stream. Other examples include generating custom serial formats, data debouncing, and signal transition detection. In general, I/O functions that would either be tedious or inappropriate for the host computer SBC can likely be moved onto the I/O-62 card itself.

2.2 Comparison of Analog vs Digital Radar Receivers

2.2.1 What is a Digital IF Receiver?

A digital IF receiver accepts the analog IF signal (typically 30 MHz), processes it and outputs a stream of wide dynamic range digital "I" and "Q" values. These quantities are then processed to obtain the moment data (for example, Z, V, W or polarization variables). Additionally, the digital receiver can accept the transmit pulse "burst sample" for the purpose of measuring the frequency, phase and power of the transmit pulse. The functions that can be performed by the digital receiver are:

- IF band pass filtering
- "I" and "Q" calculation over wide dynamic range
- Phase measurement and correction of transmitted pulse for magnetron systems – from burst sample
- Amplitude measurement and correction of transmitted pulse – from burst sample
- Frequency measurement for AFC output – from burst sample

The digital approach replaces virtually all of the traditional IF receiver components with flexible software-controlled modules that can be easily adapted to function for a wide variety of radars and operational requirements.

The digital receiver approach made a very rapid entry into the weather radar market. Up until the about 1997 weather radars were not supplied with digital receivers. Today in 2003 nearly all new weather radars and weather radar upgrades use the digital receiver approach. Much of this rapid change is attributed to the previous generation RVP7 which is the most widely sold weather radar signal processor of all time.

The number one advantage of a digital receiver is that it achieves a wide linear dynamic range (for example, >95dB depending on pulse width) without having to use AGC circuits which are complex to build, calibrate and maintain. However, there are other advantages as well:

- Lower initial cost by eliminating virtually all IF receiver components.
- Lower life cycle cost do to reduced maintenance.
- Selectable IF frequency.
- Software controlled AFC with automatic alignment.
- Programmable band pass filter
- Dual or multiple IF multiplexing
- Improved remote monitoring down to the IF level.

The following sections compare the digital receiver approach to the analog receiver approach. This illustrates the advantages of the digital approach and what functions are performed by a digital receiver.

2.2.2 Magnetron Receiver Example

A typical analog receiver for a magnetron system is shown in the top portion of Figure 9 on page 38. The received RF signal from the LNA is first mixed with the STALO (RF–IF) and the resulting IF signal is applied to one of several bandpass filters that match the width of the transmitted pulse. The filter selection is usually done with relays. The narrow band waveform is then split. Half is applied to a LOG amplifier having a dynamic range of 80–100dB, from which a calibrated measurement of signal power can be obtained. The LOG amplifier is required because it is almost impossible to build a linear amplifier with the required dynamic range. However, phase distortion within the LOG amplifier renders it unsuitable for making Doppler measurements; hence, a separate linear channel is still required.

The linear amplifier is fed from the other half of the bandpass filter split. It may be preceded by a gain control circuit (IAGC) which adjusts the instantaneous signal strength to fall within the limited dynamic range of the linear amplifier. The amplitude and phase characteristics of the IAGCattenuator must be calibrated so that the "I" and "Q" samples may be corrected during processing.

The IF output from the linear amplifier is applied to a pair of mixers that produce "I" and "Q". The mixer pair must have very symmetric phase and gain characteristics, and each must be supplied with an accurate 0-degree and 90-degree version of the Coherent Local Oscillator (COHO). The later is usually obtained by sampling a portion of the transmitted pulse, and then phase locking an oscillator (COHO) that continues to "ring" afterward.

Phase locked COHO's of this sort can be very troublesome – they often fail to lock properly, drift with age, and fail to maintain coherence over the full unambiguous range.

The transmit burst that locks the COHO is also used by the Automatic Frequency Control (AFC) loop. The AFC relies on an FM discriminator and low pass filter to produce a correction voltage that maintains a constant difference between the magnetron frequency and the reference STALO frequency. The AFC circuit is often troublesome to set and maintain. Also, since it operates continuously, small phase errors are continually being introduced within each coherent processing interval.

In contrast, the RVP8 digital receiver is shown in to lower portion of Figure 9 on page 38. The only oldparts that still remain are the microwave STALO oscillator, and the mixer that produces the transmit burst. The burst pulse and the analog IF waveform are cabled directly into the IFD on SMA coax cables. Likewise, the AFC control voltage is also a simple direct connection either with analog tuning (+-10V from IFD) or digital control via the optional DAFC interface. These cables constitute the complete interface to the radar's internal signals; no other connections are required within the receiver cabinet.

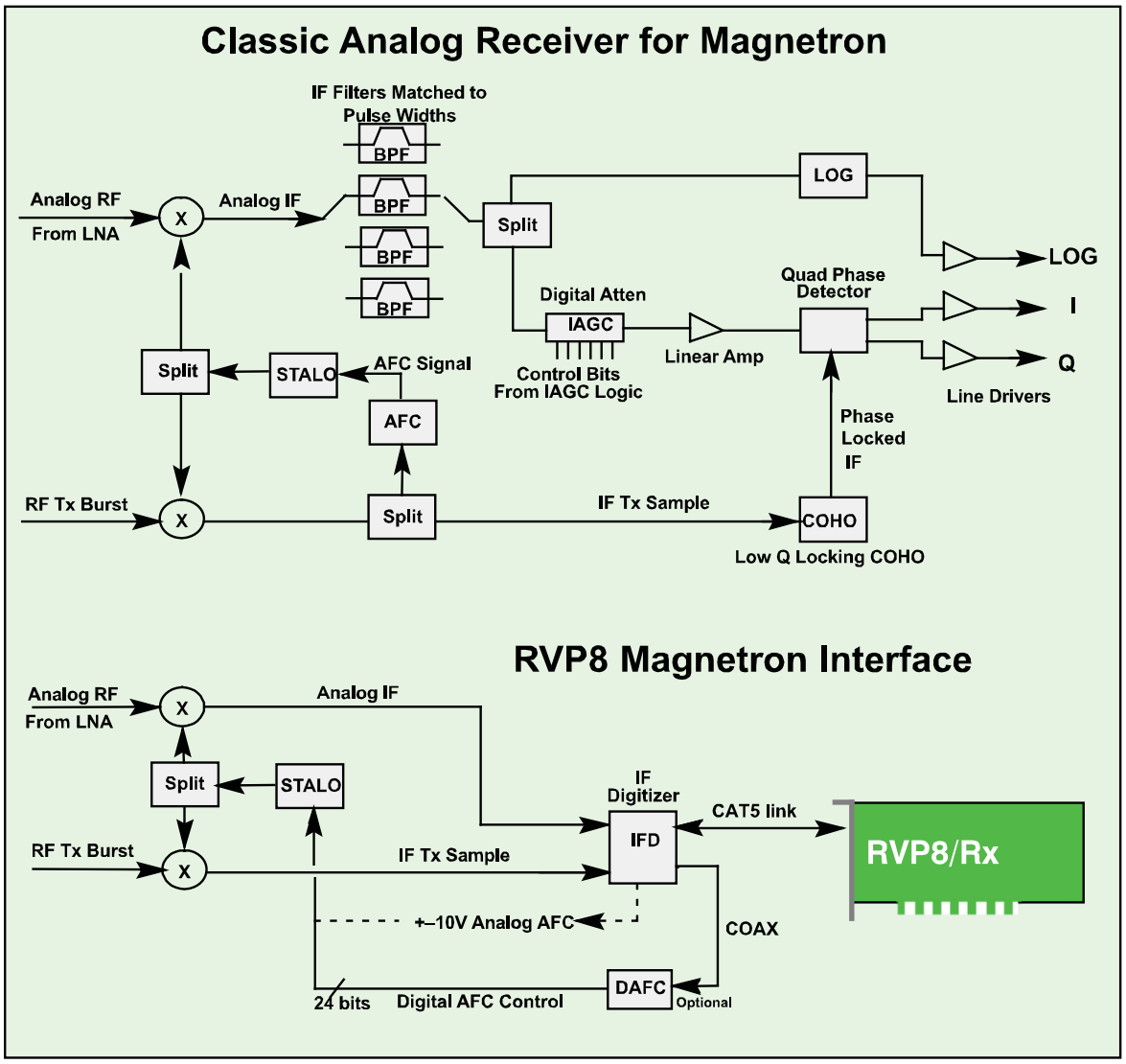


Figure 9 Analog vs Digital Receiver for Magnetron Systems

2.2.3 Klystron or TWT Receiver and Transmit RF Example

A typical analog receiver for a klystron system is shown in the top portion of Figure 10 on page 39. The arrangement of components is similar to the magnetron case, except that the COHO operates at a fixed phase and frequency, a phase shifter is included for 2nd trip echo filtering and there is no AFC feedback required. The phase stability of a Klystron system is better than a magnetron, but the system is still constrained by limited linear dynamic range, IAGC inaccuracy, quad phase detector asymmetries, phase shifter inaccuracies, etc.

The RVP8/Tx card now plays the role of a programmable COHO. The digitally synthesized transmit waveform can be phase, frequency and amplitude modulated (no separate phase shifter is required) and even produce multiple simultaneous transmit frequencies. These capabilities are used to support advanced algorithms, for example, range/velocity ambiguity resolution or pulse compression for low power TWT systems.

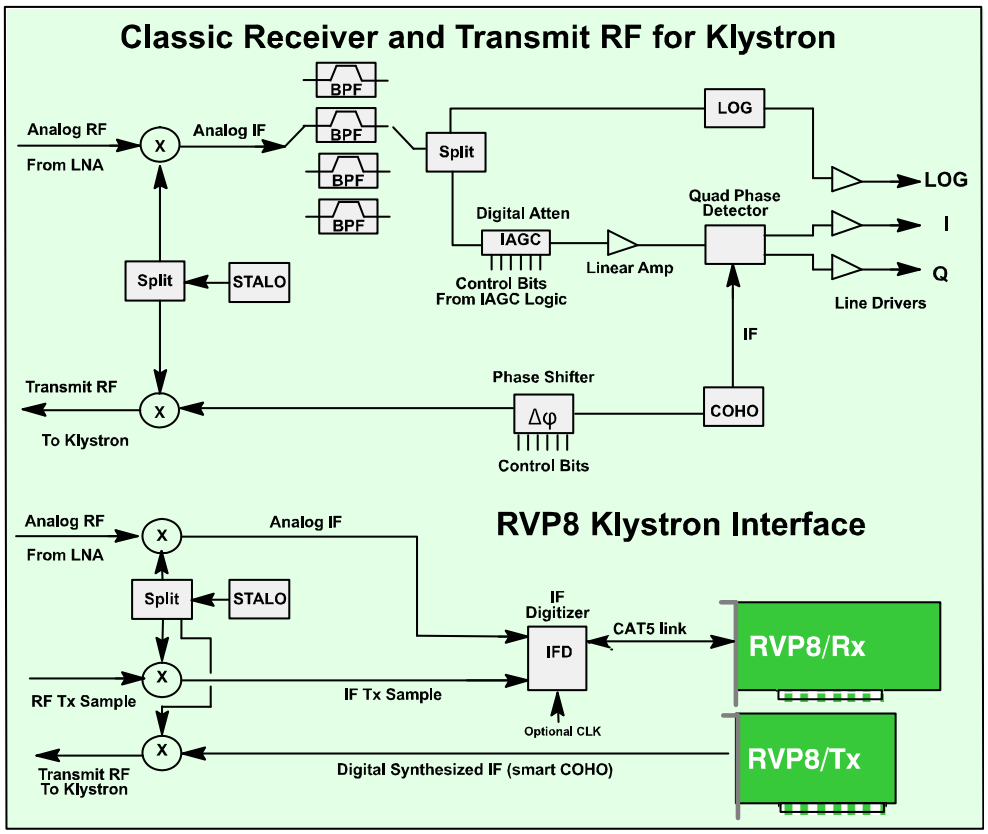


Figure 10 Analog vs Digital Receiver for Klystron Systems

2.3 RVP8 IF Signal Processing

2.3.1 IFD Data Capture and Timing

The RVP8 design concept is to perform very little signal processing within the IFD digitizer module itself. This is to minimize the presence of digital components that might interfere with the clean capture of the IF signals.

The digitized IF and burst pulse samples are multiplexed onto the fiber channel link which provides the digital data to the RVP8/Main board at approximately 540-MBits/sec. The 14-bit samples are encoded for transmission over a fiber channel link. This optical link allows the IFD to

be as far as 100 meters away from the RVP8/Main board and provides an added degree of noise immunity and isolation.

The uplink input from the RVP8/Main board provides the timing for multiplexing the burst pulse sample with the IF signal. In addition, it is used to set the AFC DAC or digital output level, and to perform self tests.

The sample clock oscillator in the IFD is selected to be very stable. The sample clock serves a similar function to the COHO on a traditional Klystron system, that is, it is the master time keeper. Because of this the IFD sample clock is used to phase lock the entire RVP8, that is, the Rx, Tx, IO-62 boards and the SoftPlane are all phase locked to the IFD sample clock. Designers have two choices for factory configuration of the IFD sample clock:

- A fixed crystal frequency selected to achieve a desired range resolution. The standard range resolution corresponds to 25 m increments
- A very narrow band VCXO (50 ppm) selected to lock to an input reference signal from the radar, and provide a desired range resolution. Vaisala stocks VCXO's for 25 m range resolution increments for reference inputs of 10, 20, 30 and 60 MHz. Custom frequency VCXO's are available on request. Examples of external reference signal sources are an external COHO, external STALO reference or perhaps even a GPS clock).

2.3.2 Burst Pulse Analysis for Amplitude/ Frequency/Phase

The burst pulse analysis provides the amplitude, frequency and phase of the transmitted pulse. The phase measurement is analogous to the COHO locking that is performed by a traditional magnetron radar. The difference is that the phase is known in the digital technique, so that range dealiasing using the phase modulation techniques is possible. Amplitude measurement (not performed by traditional radars) can provide enhanced performance by allowing the "I" and "Q" values to be corrected for variations in the both the average and the pulseto- pulse transmitted power. In addition, a warning is issued if the burst pulse amplitude falls below a threshold value.

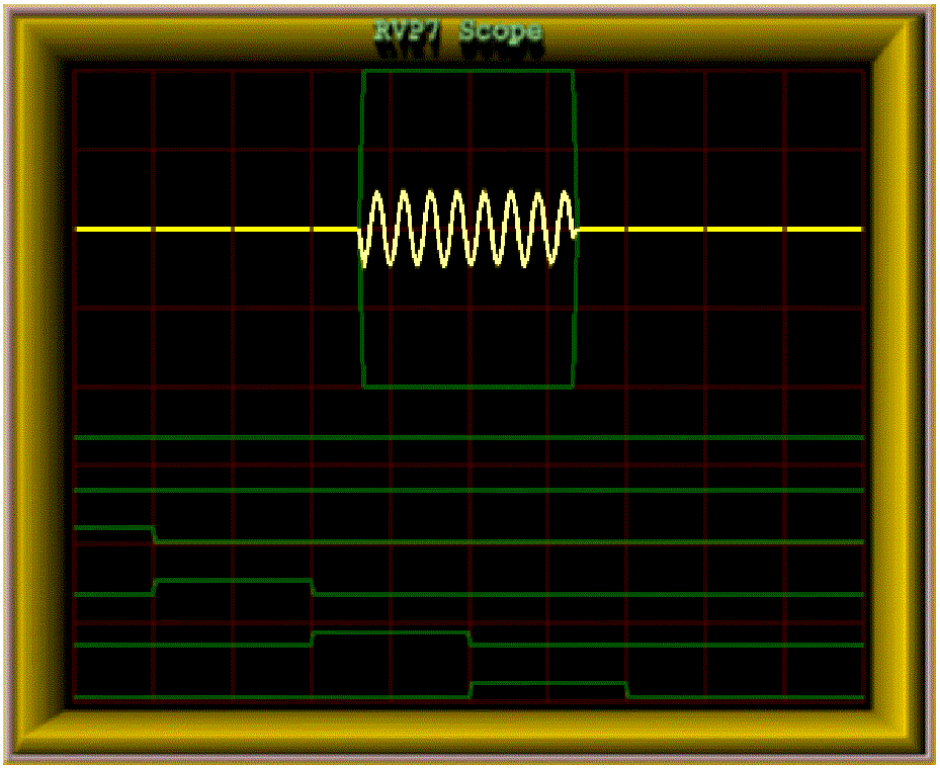


Figure 11 Burst Pulse Stream

The burst pulse data stream is first analyzed by an adaptive algorithm to locate the burst pulse power envelope (for example, 0.8 µsec). The algorithm first does a coarse search for the burst pulse in the time/ frequency domain (by scanning the AFC) and then does a fine search in both time and frequency, to assure that the burst is centered at "range 0" and is at the required IF value. The power-weighted phase of the burst pulse and the total burst pulse power is then computed. The power weighted average phase is used to make the digital phase correction. Phase jitter for magnetron systems with good quality modulator and STALO is better than 0.5 degrees RMS, as measured on actual nearby clutter targets. For Klystron systems, the phase locking is better than 0.1 degree RMS.

The burst pulse frequency is also analyzed to calculate the frequency error from the nominal IF frequency. For magnetron systems, the error is filtered with a selectable time constant which is typically set to several minutes to compensate for slow drift of the magnetron. The digital frequency error is sent via the uplink to the IFD in the receiver cabinet where a DAC converts it into an analog output to the magnetron STALO. Optionally, a DAFC unit can be Teed off the uplink cable to interface to Klystron systems do not require the AFC.

2.3.3 Rx Board and CPU IF to I/Q Processing

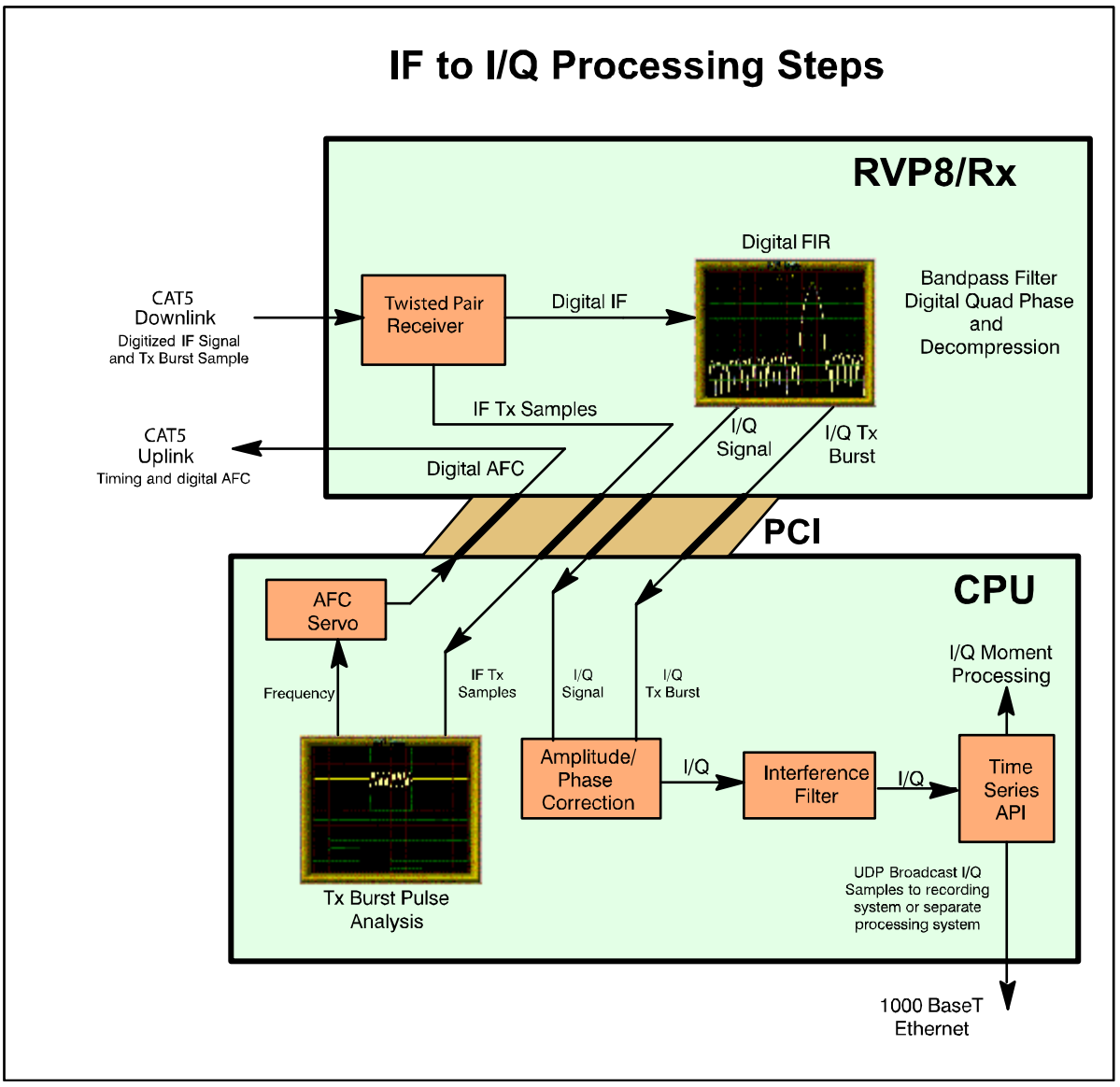
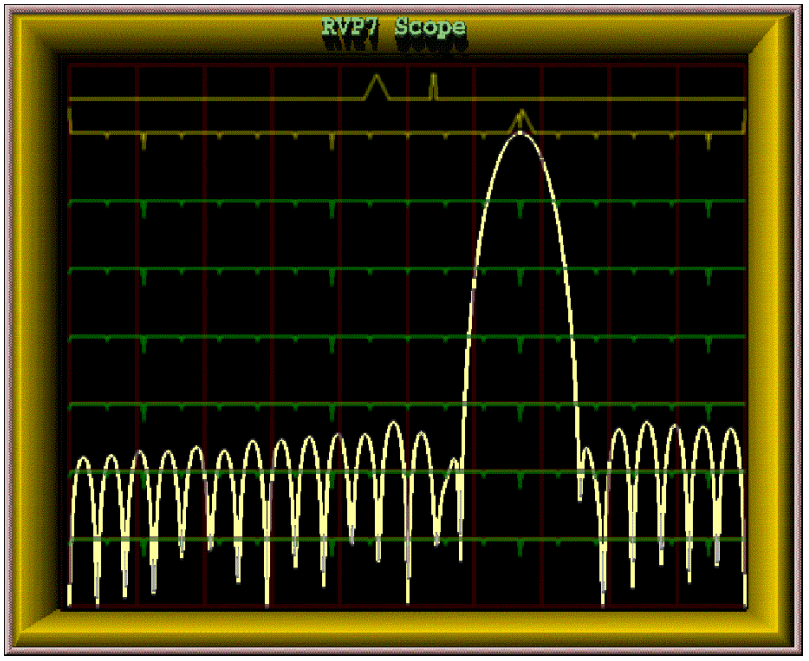


Figure 12 IF to I/Q Processing Steps

The RVP8/Rx board performs the initial processing of the IF digital data stream and outputs "I" and "Q" data values to the host computer via the PCI bus. In addition, the frequency, phase and amplitude of the burst pulse are measured. The functions performed by the processor are:

- Reception of the digital serial fiber optic data stream.
- Band pass filtering of the IF signal using configurable digital FIR filter matched to thepulsewidth.
- Range gating and optional coherent averaging (essentially performed during the band pass filtering step).
- Computation of "I" and "Q" quadrature values (also performed during the band pass filtering step).
- Transmit burst sample frequency, phase and amplitude calculation
- I and Q phase and amplitude correction based on transmit burst sample.
- Interference rejection algorithm.
- AFC frequency error calculation with output to IFD for digital or analog control of STALO (for magnetron systems).

The advantage of the digital approach is that the software algorithms for these functions can be easily changed. Configuration information (for example, processor major mode, PRF, pulsewidth, gate spacing, etc.) is supplied from the host computer.



The digital matched filter that computes "I" and "Q" is designed in an interactive manner using a TTY and oscilloscope for graphical display. The filter's passband width and impulse response length are chosen by the user, and the RVP8 constructs the filter coefficients using built-in design software. The frequency response of the filter can be displayed and compared to the frequency content of the actual transmitted pulse.

Microwave energy can come from a variety of transmitters such as ground-based, ship-based or airborne radars as well as communications links. These can cause substantial interference to a weather radar system. Interference rejection is provided as standard in the RVP8. Three different interference rejection algorithms are supported.

The RVP8/Rx board places the wide dynamic range "I" and "Q" samples directly on the PCI bus where they are sent to the processor section of the PC (for example, dual Pentium processors on a single-board computer or motherboard). The I/Q values are then processed on the Pentium processors to extract the moment information (Z, V, W and optional polarization parameters).

The I and Q values can also be placed on a gigabit Ethernet line (1000 BaseT) which is provided directly on the processor board. This means that there is no second PCI bus "hit" required to send the data to a recording system or a completely separate processing system.

2.4 RVP8 Weather Signal Processing

The processing of weather signals by the RVP8 is based on the algorithms used in the previous generation RVP7 and RVP6. However, the performance of the RVP8 allows a different approach to some of the processing algorithms, especially the frequency domain spectrum processing. All of the algorithms start with the wide dynamic range I and Q samples that are obtained from the Rx card over the PCI bus.

The resulting intensity, radial velocity, spectrum width and polarization measurements are then sent to a separate host computer to serve as input for applications such as:

- Quantitative Rainfall Measurement
- Vertical Wind Profiling
- ZDR Hail Detection
- Tornado Detection and Microburst Detection
- Gust Front Detection
- Particle Identification
- Target Detection and Tracking
- General Weather Monitoring

To obtain the basic moments, the RVP8 offers the option of several major processing modes:

- Pulse Pair Mode Time Domain Processing
- DFT/FFT Mode Frequency Domain Processing
- Random Phase Mode for 2nd trip echo filtering
- Polarization Mode Processing

Note that the RVP8 is the first commercial processor to perform discrete Fourier transforms (DFT) as well as fast Fourier transforms (FFT). FFT is more computationally efficient than DFT, but the sample size is limited to be a power of two (16, 32, 64, #hellip#) This is too restrictive on the scan strategy for a modern Doppler radar since this means, for example, that a one degree azimuth radial must be constructed from say exactly 64 input I/Q values. The RVP8 has the processing power such that when the sample size is not a power of 2, a DFT is performed instead of an FFT

These modes share some common features that are described first, followed by descriptions of the unique features of each mode.

2.4.1 General Processing features

Figure 13 on page 45 shows a block diagram of the processing steps. These are discussed below.

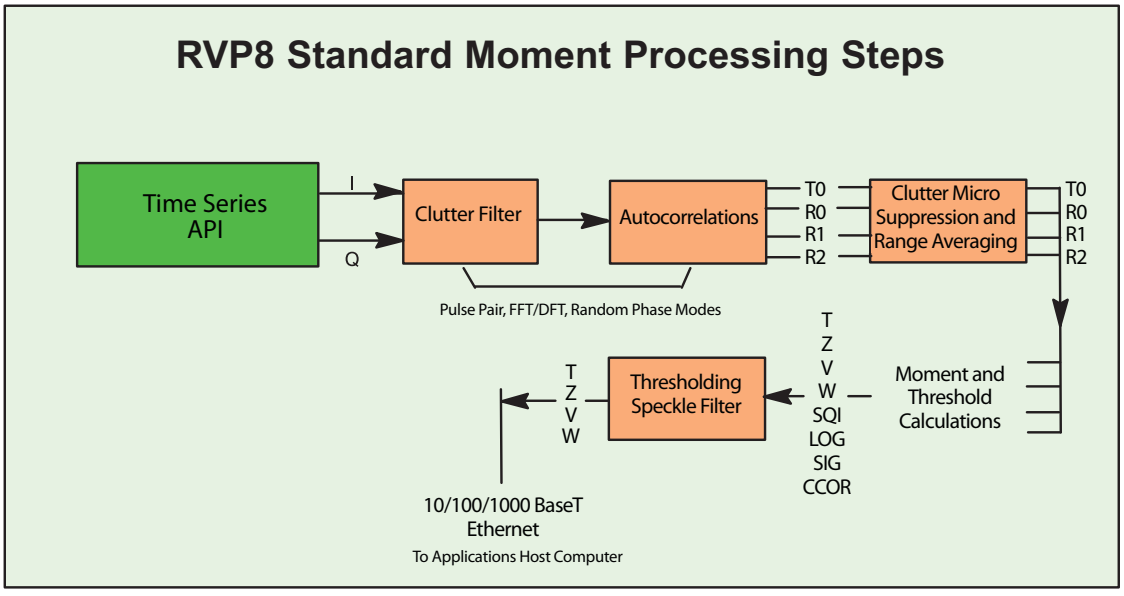


Figure 13 I/Q Processing for Weather Moment Extraction

Autocorrelations

The autocorrelations R0, R1 and R2 are produced by all processing modes. However, the way that they are produced is different, see the discussion below.

Time (azimuth) Averaging

The autocorrelations are based on input "I" and "Q" values over a selectable number of pulses between 8, 9, 10, #hellip#,256. Any integer number of pulses in this interval may be used including DFT/FFT and random phase modes.

Selectable angle synchronization using the input AZ and EL tag lines assures that all possible pulses are used during averaging for each, say, 1 degree interval. This minimizes the number of "wasted" pulses for maximum sensitivity. Azimuth angle synchronization also assures the accurate vertical alignment of radial data from different elevation angles in a volume scan (see below)..

TAG Angle Samples of Azimuth and Elevation

During data acquisition and processing it is usually necessary to associate each output ray with an antenna position. To make this task simpler the RVP8 samples 32 digital input "TAG" lines, once at the beginning and once at the end of each data acquisition period. These samples are output in a four-word header of each processed ray. When connected to antenna azimuth and elevation, the TAG samples provide starting and ending angles for the ray, from which the midpoint could easily be deduced. Since the bits are merely passed on to the user, any angle coding scheme may be used. The processor also supports an angle synchronization mode, in which data rays are automatically aligned with a user-defined table of positions. For that application, angles may be input either in binary or BCD.

Range Averaging and Clutter Microsuppression

To improve the accuracy of the reflectivity measurements, the RVP8 can perform range averaging. When this is done, autocorrelations from consecutive range bins are averaged, and the result is treated as if it were a single bin. This type of averaging is useful to lower the number of range bins that the host computer must process.

Range averaging of the autocorrelations may be performed over 2, 3, 4, #hellip#, 16 bins. Prior to range averaging, any bins that exceed the selectable clutter-to-signal threshold are discarded. This prevents isolated strong clutter targets from corrupting the range average, which improves the sub-clutter visibility.

Moment Extraction

The autocorrelations serve as the basis for the Doppler moment calculations,

- Mean velocity – from Arg [R1]
- Spectrum width – from |R1| and |R2| assuming Gaussian spectrum
- dBZ – from R0 with correction for ground clutter, system noise and gaseous attenuation. Uses calibration information supplied by host computer.
- dBT – identical to dBZ except without ground clutter.

These are the standard parameters that are output to the host computer on the high-speed Ethernet interface.

Thresholding

The RVP8 calculates several parameters that are used to threshold (discard) bins with weak or corrupted signals. The thresholding parameters are:

- Signal quality index (SQI=|R1|/R0)
- LOG (or incoherent) signal-to-noise ratio (LOG)
- SIG (coherent) signal-to-noise ratio
- CCOR clutter correction

These parameters are computed for each range bin and can be applied in AND/OR logical expressions independently for dBZ, V and W.

Speckle Filter

The speckle filter can be selected to remove isolated single bins of either velocity/width or intensity. This feature eliminates single pixel speckles which allows the thresholds to be reduced for greater sensitivity with fewer false alarms (speckles). Both a 1D (single azimuth ray) and 2D (3 azimuth rays by 3 range bins) are supported.

Velocity Unfolding

A special feature of the RVP8 processor is its ability to "unfold" mean velocity measurements based on a dual PRF algorithm. In this technique two different radar PRF's are used for alternate N-pulse processing intervals. The internal trigger generator automatically produces the correct dual-PRF trigger, but an external trigger can also be applied. In the later case, the ENDRAY_ output line provides the indication of when to switch rates. The RVP8 measures the PRF to determine which rate (high or low) was present on a given processing interval, and then unfolds based on

either a 2:3, 3:4 or 4:5 frequency ratio. Table 1 on page 48 gives typical unambiguous velocity intervals for a variety of radar wavelengths and PRF's.

Table 1 Examples of Dual PRF Velocity Unfolding

			Unambiguous Velocity (m/s) for Various Radar Wavelengths			
PRF1	**PRF2**	**Unambiguous Range (km)**	**3 cm**	**5 cm**	**10 cm**	
500	*	300	3.75	6.25	12.50	No Unfolding
1000	*	150	7.50	12.50	25.00	
2000	*	75	15.00	25.00	50.00	
500	333	300	7.50	12.50	25.00	Two Times Unfolding
1000	667	150	15.00	25.00	50.00	
2000	1333	75	30.00	50.00	100.00	
500	375	300	11.25	18.75	37.50	Three Times Unfolding
1000	750	150	22.50	37.50	75.00	
2000	1500	75	45.00	75.00	150.00	
500	400	300	15.00	25.00	50.00	Four Times Unfolding
1000	800	150	30.00	15.00	100.00	
2000	1600	75	60.00	100.00	200.00	

2.4.2 RVP8 Pulse Pair Time Domain Processing

Pulse pair processing is done by direct calculation of the autocorrelation. Prior to pulse pair processing, the input "I" and "Q" values are filtered for clutter using a a time domain notch filter. Filters of various selectable widths are available for either 40 or 50 dB stop band attenuation. The filtered I/Q values are processed to obtain the autocorrelation lags R0, R1 and R2. The unfiltered power is also calculated (T0). The autocorrelations are then sent to the range averaging and moment extraction steps.

2.4.3 RVP8 DFT/FFT Processing

The DFT/FFT mode allows clutter cancelation to be performed in the frequency domain. DFT is used in general, with FFT's used if the requested sample size is a power of 2.

Three standard windows are supported to provide the best match of window width to the spectrum dynamic range:

- Rectangular
- Hamming
- Blackman
- Exact Blackman
- Von Han

After the FFT step, clutter cancelation is done using a selectable fixed width filter that interpolates across the noise or any overlapped weather or an adaptive filter which automatically determines the optimalwidth. This technique preserves overlapped weather as compared to time domain notch filters which will always attenuate overlapped weather to some extent, depending on the spectrum width. After clutter cancelation, R0, R1 and R2 are computed by inverse transform and these are used for moment estimation.

2.4.4 Random Phase Processing for 2nd Trip Echo

Second trip echoes can be a serious problem for applications that require operation at a high PRF. Second trip echoes can appear separately or can be overlaid on first trip echoes (second trip obscuration). The random phase technique separates the first and second trip echoes so that:

- In nearly all cases, the 2nd trip echo can be removed from the first trip even in the case of overlapped 1st and 2nd trip echoes. The benefit is a clean first trip display.
- The 2nd trip echoes can be recovered and placed at their proper range at 1st trip/2nd trip signal ratios of up to 40 dB difference for overlapped echoes. Because of the wide dynamic range of weather echoes, this power limit will sometimes be exceeded.

The technique requires that the phase of each pulse be random. Digital phase correction is then applied in the processor for the first and second trips. The critical step is the adaptive filter which removes the echo of the other trip to increase the SNR. Magnetrons have a naturally random phase. For Klystron radars, a digitally controlled precision IF phase shifter is required. The RVP8 provides an 8-bit RS422 output for the phase shifter.

For more information on the technique refer to Joe, et. al., 1995.

2.4.5 Polarization Mode Processing

Polarization processing uses a time domain autocorrelation approach to calculate the various parameters of the polarization co-variance matrix, that is, ZDR, LDR, PHIDP, RHOHV, PHIDP (KDP), etc. In addition, the standard moments T, V, Z, W are also calculated. Which parameters are available and which algorithms are used to calculate them depends on the type of polarization radar, for example, single channel switching, simultaneous transmit and receive (STAR), dual channel switching. Vaisala is licensed by US National Severe Storms Laboratory (NSSL) to use the STAR hardware and processing techniques and algorithms.

The polarization variables are computed from data filtered for clutter. The settings are the same as for standard moments, except that a zero value is inserted within the clutter notch. GMAP filters are not available in dual-pol mode. The filter is applied identically for each polarimetric channel, required in adaptive approaches, too. Low signal-to-noise ratios have an effect on polarimetric data, which can be corrected for. The variables ZDR and LDR are proportional to echo powers and require special calibration of the residual channel offsets, while the polarimetric phase measurements RHOHV, RHOH, RHOV, PHIDP, PHIH, PHIV and KDP have the advantage of being immune to radar calibration.

2.4.6 Output Data

The RVP8 output data for standard moment calculations consist of mean radial velocity (V), Spectrum Width (W), Corrected Reflectivity(Z or dBZ) and Uncorrected Reflectivity (T or dBT). Other data outputs include I/Q time series, DFT/FFT power spectrum points and polarization parameters. The output can be made in either 8 or 16-bit format. 8-bit format is preferred over 16-bit format for most applications since the accuracy is more than adequate for an operational radar system, and the data communications are reduced by 50%. 16-bit formats are sometimes used by research customers for data archive purposes. Note that time series and DFT are always 16-bit formats. All data formats are documented in Chapter 7, Host Computer Commands, on page 275.

A standard output is the I/Q time series on gigabit network (1000 BaseT). These are sent via UDP broadcast to an I/Q archiving system or even a completely independent parallel processing system.

2.5 RVP8 Control and Maintenance Features

2.5.1 Radar Control Functions

The RVP8 also performs several important radar control functions:

- Trigger generation- up to 6 programmable triggers.
- Pulsewidth control (four states controlled by four bits).
- Angle/data synchronization- to collect data at precise azimuth intervals (for example, every 0.5, 1, 1.5 degrees) based on the AZ/EL angle inputs
- Phase shifter- to control the phase on legacy Klystron systems. New or upgrade Klystron or TWT systems can use the RVP8/Tx card to provide very accurate phase shifting.
- ZDR switch control- for horizontal/vertical or other polarization switching scheme.
- AFC output (digital or analog) based on the burst pulse analysis for magnetron systems.

Pulsewidth and trigger control are both built into the RVP8. Four TTL output lines can be programmed to drive external relays that control the transmitter pulsewidth. The internal trigger generator drives six separate lines, each of which can be programmed to produce a desired waveform. The trigger generator is unique in that the waveforms are stored in RAM and can be modified interactively by user software. Thus, precisely delayed and jitter-free strobes and gates can easily be produced. For each pulsewidth there is a corresponding maximum trigger rate that can be generated. Note, however, that the RVP8 can also operate from an external user-supplied trigger. In either case, the processor measures the trigger period between pulses so that user software can monitor it as needed.

The RVP8 also supports trigger blanking during which one or more (selectable) of the transmit triggers can be inhibited. Trigger blanking is used to avoid interference with other electronic equipment and to protect nearby personnel from radiation hazard. There are two techniques for this:

- 2D AZ/EL sector blanking areas can be defined in the RVP8 itself.
- An external trigger blanking signal (switch closure to ground, TTL or RS422) can be supplied, for example from a proximity switch that triggers when the antenna goes below a safe elevation angle or connected to the radome access hatch.

2.5.2 Power-Up Setup Configuration

The RVP8 stores on disk an extensive set of configuration information. The purpose of these data is to define the exact configuration of the RVP8 upon startup. The setup information can be accessed and modified using either a local keyboard and monitor, or over the network. For multiple radar networks, the configuration management can be centrally administered by copying tested "master" configuration files to the various network radars. It is not necessary to go to the radar to change ROM's as was the case for previous generation processors.

2.5.3 Built-In Diagnostics

On power-up, the RVP8 performs a sequence of internal self-tests. The test sequence requires about four seconds to perform, and tests approximately 95% of the internal digital circuitry. Errors are isolated to specific sections of the board as much as possible. If any check fails, the user can be certain that some component is not functioning correctly. However, there is a very small chance that even a defective board may pass all the tests; the failure may be in one of the few areas that can not be checked.

The RVP8 displays the test results on the LED front panel (for a standard Vaisala chassis). In this way, there is immediate visual confirmation of the diagnostic tests, even if the host computer has not yet been connected. The local keyboard and monitor or a networked workstation can be used to see the test results in the TTY menus or even invoke a power–up reset and test.

2.6 Support Utilities and Available Application Software

The RVP8 system includes a complete set of tools for the calibration, alignment and configuration of the RVP8. These includes the following utilities:

- **ascope**- a comprehensive utility for manual signal processor control and data display of moments, times series and Doppler spectra. ascope includes a realistic signal simulator capable of producing both first and second trip targets. Recording/playback of time series and moments is included as well.
- **dspx**- an ASCII text-based program to access and control the signal processor, including providing access to the local setup menus.
- **speed**- a performance measuring utility.

- **DspExport**- exports the RVP8 to another workstation over the network. This allows utilities on a remote network to run locally, as opposed to exporting the utility display window over the network
- **setup**- interactive GUI for creating/editing the RVP8 configuration files.
- **zauto**- calibration utility for use with a test signal generator.

These tools can be run locally on the RVP8 itself or over the network from a central maintenance facility. The DspExport utility improves the performance of the utilities for network applications by letting them be run on the workstation that is remote from the RVP8. Note that standard X–Window export is of course supported but requires more bandwidth.

In addition, complete radar application software can be purchased from Vaisala:

- **IRIS/Radar** on a separate PC, interfaces to the RVP8 by 100 BaseT Ethernet. IRIS/Radar controls both the RVP8 and the Vaisala RCP8 radar/antenna control processor. The package provides complete local and remote control/monitoring, data processing and communication for a radar system.
- **IRIS/Analysis** (and options) runs on a separate PC, often at a central site. One IRIS/Analysis can support up to 20 radar systems. This functions as a radar product generator (RPG) to provide outputs such as CAPPI, rain accumulations, echo tops, automatic warning and tracking, etc. Optional software packages are provided for special applications: wind shear and microburst detection, hydrometeorology with raingage calibration and subcatchments, composite, dual Doppler and 3D Display.
- **IRIS/Web** provides IRIS displays to network users on standard PC's (Windows or Linux) running Netscape or Internet Explorer.
- **IRIS/Display** can display products sent to it and, with password authorization, can serve as a remote control and monitoring site for networked radar systems. Features such as looping, cross–section, track, local warning, annotation, etc. are all provided by IRIS/Display. Note that both IRIS/Analysis and IRIS/Radar have all of the capabilities of IRIS/Display in addition to their own functions. This means that any IRIS system can display products.

2.7 System Network Architecture

The RVP8 provides considerable flexibility for network operation. This allows remote control and monitoring of the system from virtually anywhere on the network, subject to the user's particular security restrictions.

Unlike the previous generation RVP7, which used a SCSI interface, the RVP8 uses a network interface exclusively. The "dsp lib" runs locally on the RVP8 and a utility, called DspExport, exports the library over the network using a TCP/IP socket. Typically this is exported to a local host radar control workstation (RCW) on the network. Perhaps this workstation is running the Vaisala IRIS software. At least 10BaseT connection is recommended for this connection.

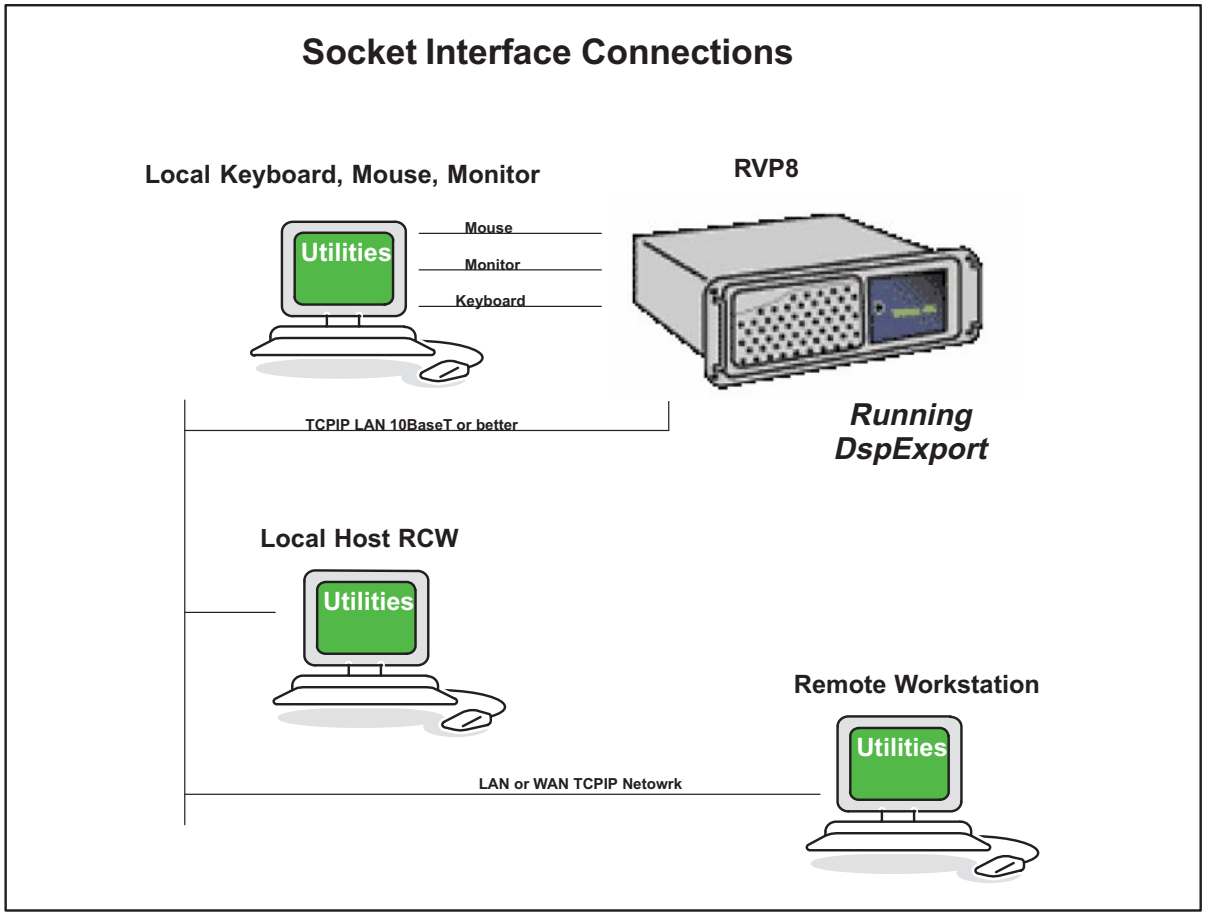


Figure 14 Network Architecture for Socket Interface with DspExport

A remote workstation on the network can also use the DspExport technique to communicate for configuration, monitoring and diagnostic testing.

2.8 Open Architecture and Published API

Vaisala recognizes that certain users may require the ability to write their own signal processing algorithms which will run on the RVP8. To accommodate this, the RVP8 software is organized to allow separately compiled plug-in modules to be statically linked into the running code. The application program interface (API) allows user code to be inserted at the following stages of processing:

- Tx/Rx waveform synthesis and matched filter generation— The API allows the transmit waveforms to be defined from pulse to pulse, along with the corresponding FIR coefficients that will extract (I,Q) from that Tx waveform. This allows users to experiment with arbitrary waveforms for pulse compression and frequency agility.
- Time series and spectra processing from (I,Q)- The API allows you to modify the default time series and spectra data, for example, to perform averaging or windowing in a different way.
- Parameter generation from (I,Q)- This is probably where the greatest activity will occur for user–supplied code. The API allows you to redefine how the standard parameters (dBZ, Velocity, Width, PHIDP, etc.) are computed from the incoming (I,Q) time series. You may also create brand new parameter types that are not included in the basic RVP8 data set.

Note that the standard Vaisala algorithms are not made public in this model. Rather, the interface hooks and development tools are provided so that users can add their own software extensions to the RVP8 framework. Many of the library routines that are fundamental to the RVP8 are also documented and can be called by user code; but the source to these routines is not generally released. Development tools which are not under public license must be purchased separately by the customer.

While most customers will use the signal processing software supplied by Vaisala, the new open software architecture approach employed by the RVP8 will be very useful to those research customers who want to try innovative new approaches to signal processing, or to those OEM manufacturers who are interested in having their own "custom" stamp on the product.

2.9 RVP8 Technical Specifications

2.9.1 IFD Digitizer Module, Rev E or later

Input Signals

- IF Received Signal: 50Ω, + 6.5 dBm full–scale, +20dBm absolute max
- IF Burst or COHO: 50Ω, +6.5 dBm full–scale, +20dBm absolute max
- Optional Reference Clock: 2–60 MHz –10 to 0 dBm

IF Ranges:

- 12—34 MHz, 38—70 MHz

Linear Dynamic Range

- 85 to >100dB depending on pulsewidth/bandwidth filter

A/D Conversion

- Resolution 14 bit with jitter <2.5 picosec
- Sampling rate 67 to 79 MHz (selectable, standard is 71.9364 MHz)

AFC Output

- Analog -10 to +10V
- Optional Digital AFC (DAFC) with up to 24 programmable output bits.
- Automatic 2-D (time/frequency) burst pulse search and fine tracking algorithms.

IFD Link

- Uses shielded CAT 5E cable, non standard signals, requires RVP8/Rx card, rev C or later.

Cable length to RVP8/Rx

- 2—25 meters, with automatic calibration of round trip time and range correction.

2.9.2 RVP8/Rx PCI Card, Rev C or later

Pulse Repetition Frequency

- 50 Hz to 20 KHz +0.1%, continuously selectable.

IF Band Pass Filter

- Programmable Digital FIR with software selectable bandwidth. Built-in filter design software with graphical user interface.

Impulse Response

- Up to 3024 FIR filter taps, corresponding to 75 µsec impulse response length for 72 MHz IF samples at 125 meter range resolution. These very long filters are intended for use with pulse compression.

Range Resolution

- Minimum bin spacing of 25 meters selectable in N*8.33 meter steps. Bins can be positioned in a configurable range mask with resolution of N* the fundamental bin spacing, or arbitrarily to an accuracy of ±2.2 meters.

Maximum Range

- Up to 1024 km

Number of Range Bins

- Full unambiguous range at minimum resolution or 3096 range bins (whichever is less). The RVP8 processor may only be fast enough to process an average of 50 meter bins.

Electrical Interfaces

- CAT 5E cable from the IFD, rev E or later.
- BNC #1 for trigger output (12V, 75Ω), or pretrigger input.

  BNC #2 for trigger output (12V, 75Ω).
- 9-pin "D" connector supporting four RS-422 differential signals for miscellaneous input and output with SoftPlane™ support. Each line pair can operate as a transmitter or as a receiver depending on what's needed. Possible uses are: alternate reference clock input, gating input for CW modes, additional trigger outputs, external phase shift requests, etc.

Data Output via PCI Bus

- 16-bit floating I and Q values
- 14-bit raw IF samples

2.9.3 RVP8/Tx PCI Card

Analog Waveform Applications

- Digitally synthesized IF transmit waveform for pulse compression, frequency agility, and phase modulation applications.
- Master clock or COHO signal to the radar; can be phase locked or free running, arbitrary frequency.

Analog Output Waveform Characteristics

- Two independent, digitally synthesized, analog output waveforms (BNC). These two outputs are electrically identical and logically independent IF waveform synthesizers that can produce phase modulated CW signals, finite duration pulses, compressed pulses, etc.
- Can drive up to +12dBm into 50Ω.
- 14-bit interpolating TxDAC provides 71dB Signal-to-Noise Ratio.
- IF center frequency selectable from 8 to 32.4 MHz, and from 48.6 to 75MHz.
- Signal bandwidth as large as 15MHz for wideband/multiband Tx applications. Band width is adjustable in software.
- Continuous or pulse modulated output with band width limiting on pulse modulation output.
- Precise phase shifting with transient band width limiting.
- Total harmonic distortion less than -74dB.
- Waveform pre-emphasis compensates for both static and dynamic Tx nonlinearities.

Other I/O signals

- Clock In/Out 50Ω SMA connector. This can receive a CW reference frequency to which the RVP8/Tx can lock to a P/Q frequency multiple (much like the RVP8/IFD can lock to an external reference). This connector can also supply the TxData Clock, optionally divided by some N between 1 and 16, in order to supply external circuitry with +10dBm clock reference at 50Ω.
- 9-pin "D" connector supporting four RS-422 differential signals for miscellaneous input and output with SoftPlane™ support. Each line pair can operate as a transmitter or as a receiver depending on what's needed. Possible uses are: alternate reference clock input, gating input for CW modes, additional trigger outputs, external phase shift requests, etc.

2.9.4 Vaisala I/O-62 PCI Card

- Short format PCI card with 62-position "D" connector. Multiple cards may be installed.
- Includes D/A, A/D, discrete inputs and outputs (TTL, wide range, RS422, etc.) See summary table below.
- I/O pin assignment mapping by **softplane.conf** file.
- Standard or custom remote backpanels available.
- ESD protection using Tranzorb™ silicon avalanche diode surge suppression and high-voltage tolerant components.

Table 2 Vaisala I/O-62 Summary of Electrical Interfaces

Quantity	Description
40	Lines configurable in groups of 8 to be either inputs or outputs. The electrical specifications are software defined within each group as follows: - Single-ended TTL input or output with software–configured pull-up or pull-down resistors for inputs. - Wide range inputs (±27VDC, threshold +2.5VDC), often used for "lamp voltage" status inputs. - RS-422/485 @ 10 MBit/sec (requires two lines each). RS-422 receivers can be configured in software to have 100Ω termination between each pair.
8	A/D convertors configurable as 0, 4, or 8 convertors, ±2V, 12 bits @ 10 MHz, These lines are shared with some of the 40 I/O lines listed above.
2	D/A convertors, ±10V 1 MHz update rate, output can drive a 75Ω load.
2	SPDT relays on the board. These are often used for switching high power relays. Contacts are diode protected.
2	RS-232C full duplex lines (Tx and Rx)
4	12V 75Ω trigger drivers .
2	Power/Ground pairs of 12V power (filtered, fused) for external equipment or remote backpanel use (up to 24 W total). Polyfuse technology acts like a circuit breaker with auto reset in the event of an overload.
8	Ground wires for signal grounds from the remote back panel.

2.9.5 I/O-62 Standard Connector Panel

- Mounts on front or rear of standard 19" EIA rack
- Connects to I/O-62 via 1:1 62–pin 1.8–m cable (provided).
- Provides standard inputs and outputs required by most weather radars such as triggers, polarization control, pulse width control and antenna angles.
- Az and El synchro and reference inputs (nominal 100V 60 Hz)
- 3 internal relays and 4 12V relay control signals for switching external devices.
- Programmable scope test points with source waveforms selectable in software.
- Diagnostic power supply and self test LED's for troubleshooting.

Table 3 RVP8 Connector Panel Summary

J-ID	Label	Type	Description
J1	AZ INPUT	DBF25	Up to 16–bits of parallel TTL binary or BCD angle
J2	AZ OUTPUT	DBF25	Up to 16–bits of parallel TTL binary or BCD angle
J3	PHASE OUT	DBF25	Up to 8–bits of parallel TTL or RS422. Angles are configurable.
J4	EL INPUT	DBF25	Up to 16–bits of parallel TTL binary or BCD angle
J5	EL OUTPUT	DBF25	Up to 16–bits of parallel TTL binary or BCD angle
J6	RELAY	DBF25	3 internal relays, contact rating 0.5 A continuous. The switching load is 0.25 A and 100V, with the additional constraint that the total power not exceed 4VA. 4, 12V relay control signals, up to 200mA. (Note that external relays should be equipped with proper diode protection to shunt the back EMF).
J7	SPARE	DBF25	20 additional TTL I/O lines each configurable to be input or output.
J8	SPARE	DBF25	10 differential analog inputs, up to ±20V max multiplexed into A/D convertor sampling each at >1000 Hz.
J9	MISC I/O	DBF25	7 additional RS422 lines and 2 each dedicated (non–multiplexed) A/D inputs (±580V with pot adjust) and D/A outputs (±10V).
J10	SERIAL	DBF9	RS232C
J11	SERIAL	DBF9	RS232C
J12	S–D	Modular	3 × 4 matrix connector for AZ and EL synchro and reference inputs
J13	TP–1	BNC	Programmable scope test point. 75 Ohms
J14	TP–2	BNC	Programmable scope test point. 75 Ohms
J15	TRIG–1	BNC	12V trigger into 75 Ohms
J16	TRIG–2	BNC	12V trigger into 75 Ohms
J17	TRIG–3	BNC	12V trigger into 75 Ohms
J18	TRIG–4	BNC	12V trigger into 75 Ohms

2.9.6 RVP8 Processing Algorithms

Input from Rx Board

- 16-bit I/Q samples
- Optional dual-channel I/Q samples (for example, for polarization systems or dual frequency systems)

IQ Signal Correction Options

- Amplitude jitter correction based on running average of transmit power from burst pulse.
- Interference correction for single pulse interference
- Saturation correction (3 to 5 dB)

Primary Processing Modes

- Poly-Pulse Pair (PPP)
- DFT
- Random or Phase Coded 2nd trip echo filtering/recovery
- Optional Polarization with full co-variance matrix (ZDR, PHIDP, LDR, RHOHV, etc.)
- Optional Pulse Compression

Processing Options

- FIR Clutter filters (40 and 50 dB) in pulse pair mode.
- Adaptive width clutter filters in DFT and phase coded 2nd trip mode.
- Velocity De-Aliasing: Dual PRF Velocity unfolding at 3:2, 4:3 and 5:4 PRF ratios or Dual PRT Velocity processing for selectable inter-pulse intervals.
- Range De-aliasing: Phase coding method (random phase for magnetron)

  Frequency coding method (not available for magnetron)
- Scan angle synchronization for data acquisition.
- Pulse integration up to 1024
- Corrections for gaseous attenuation and $1/R^2$.
- Up to 4 pulse widths

Data Outputs

- dBZ Calibrated equivalent radar reflectivity, 8 or 16 bits
- V Mean radial velocity, 8 or 16 bits
- W Spectrum width, 8 or 16 bits
- I/Q Time series, 16 bits each per sample
- DFT Doppler Spectrum output option in DFT mode, 16 bits per component
- Optional: ZDR, PHIDP, RHOHV, LDR, RHO, 8 or 16 bits

Data Quality Thresholds

- Signal–to–noise ratio (SNR) Used to reject bins having weak signals.

  Typically applied to dBZ.
- Signal quality index (SQI) Used to reject bins having incoherent signals.

  Typically applied to mean velocity and width.
- Clutter-to-signal ratio (CSR) Used to reject range bins having very strong clutter.

  Typically applied to mean velocity, width and dBZ.
- Speckle Filter Filter removes single-bin targets such as aircraft or noise

  Fills isolated missing pixels as well.

2.9.7 RVP8 Input/Output Summary

Ethernet Input/Output from Host Computer

- Data output of calibrated dBZ, V and W during normal operation. Full I/Q timeseries recording with a separate **tsarchive** utility, or through a customer's application using a public API. Signal processor configuration and verification read–back is performed via the Ethernet interface.

RS-232C Serial Data I/O

- For real time display/monitoring or data remoting.

AZ/EL Angle Input Options

- Serial AZ/EL angle tag input using standard Vaisala RCP format.
- 16-bit each parallel TTL binary angles via the I/O-62 card.
- Synchro angle inputs via the I/O-62 card.
- Vaisala network antenna packet protocol.

Trigger Output

- Up to 10 total triggers available on various connector pins. Triggers are programmable with respect to trigger start, trigger width and sense (normal or inverted).

Optional Polarization Control

- RS-422 differential control for polarization switch.

2.9.8 Physical and Environmental Characteristics

Packaging

- Motherboard Configuration 4U rack mount with 6 PCI slots
- Custom PC configurations available or packaged by customer.
- Dimensions of standard 4U chassis
  43.2 wide x 43.2 long x 17.8 cm high
  17 wide x 17 long x 7.00 inch high
- Dimensions IF Digitizer
  2.5 wide x 10.9 long x 23.6 cm high
  1 wide x 4.3 long x 9.3 inch high
- Redundant Power Supplies. Three hot–swap modules with audio failure alarm.

Input Power

- IFD
  100–240 VAC 47–63 Hz auto–ranging
- Main Chassis

  60/50 Hz 115/230 VAC Manual Switches

Power Consumption

- RVP8/Main Processor
  180 Watts with Rx and Motherboard
- RVP8/IFD IF Digitizer

  12 Watts

Environmental

- Temperature
  0C (32F) to 50C (122F)
- Humidity

  0 to 95% non–condensing

Reliability

- MTBF>50,000 hours (based on actual RVP7 field data).

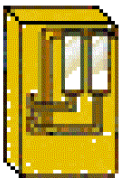
CHAPTER 3

HARDWARE INSTALLATION

3.1 Overview and Input Power Requirements

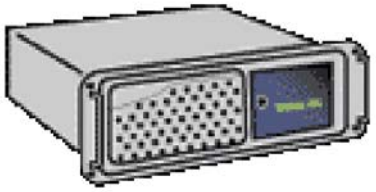
This chapter describes how to install the RVP8 hardware. Topics include mechanical installation and siting, electrical specifications of the interface signals, system-level considerations and the standard connector panel that is provided.

There are three major modules supplied with the RVP8. These are:



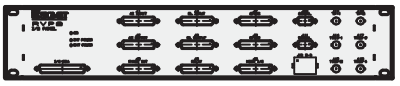
IFD (IF Digitizer) Typically mounted in the radar receiver cabinet.

Input Power 47–63 Hz 100–240 VAC Auto-ranging



Main Chassis Usually mounted in 19" EIA rack.

Input Power 60/50 Hz 115/230 VAC Manual Switches.



I/O-62 Connector Panel Usually mounted in 19" EIA rack within 2 m of Main Chassis

Much of the RVP8 I/O is configured via software. This makes the unit very flexible. Also, since there is virtually no custom wiring, it is very easy to insert spare modules and circuit cards. The software configuration of the I/O is described in Appendix A, Serial Status Formats, on page 357.

This section, in conjunction with Appendix B, Optional Dual Polarization-ZDR, PHIDP, KDP, LDR, ..., on page 363, describes the physical installation of the hardware.

WARNING	The Main Chassis redundant power supplies are NOT auto-ranging like the IFD. These are factory configured for the expected voltage, but should be VERIFIED by the customer before power is applied to the system.

3.2 IFD IF Digitizer Module Installation

WARNING	The IFD mains power is to be permanently "hard wired" in a NEMA electrical enclosure that is accessible only to a trained technician. The ground (earth) connection should be attached directly to the IFD case mounting screw then brought to the power supply ground connection.

WARNING	Disconnect the the mains power before opening the IFD for service. The IFD is best serviced by disconnecting the mains power, removing it from its mount and placing it on a bench.

3.2.1 IFD Introduction

The IFD IF digitizer is housed in an electrically sealed solid metal enclosure to achieve good immunity to external electrical noise. The internal circuitry has been designed to minimize the number of digital components, and it is carefully grounded and shielded to make the cleanest possible samples of the input IF signal. The unit is cooled by direct conduction of heat through the metal chassis; there are no openings required for airflow.

The IFD replaces all of the IF receiver components that are found in a traditional analog receiver system, that is,

- Band Pass Filters
- LOG Receiver
- AFC Circuit
- AGC or IAGC circuit
- Quad Phase Detector
- COHO (on magnetron systems)
- Line drivers for base band video

Indeed, one of the most time consuming parts of an upgrade is often the removal of old components. Many customers choose to simply bypass them and leave them in place. In some cases there will be other receiver modifications required to match the IFD signal input specifications. For example, IF attenuators or an IF amplifier are sometimes required.

NOTE	If you are doing an upgrade of an older system, you might want to consider purchase of a new STALO which can make significant improvements in Doppler performance.

You should carefully document and red-line your system schematics to reflect any changes to the receiver.

3.2.2 IFD Revision History

There have been several hardware revisions of the IFD module since its introduction initially with the RVP7. Table 4 on page 68 summarizes the differences among all of the versions that have been manufactured so far. The remainder of this chapter covers only the 14-bit units, although the previous generation 12-bit units are compatible with the RVP8 as well.

Table 4 Differences Among Versions of the IFD

	Rev.B	*Rev.C*	*Rev.D*	*Rev.E & Higher*
A/D Chip	Analog Devices AD9042, 12–Bits		AD6644, 14–Bits	AD6645, 14–Bits
Nominal IF Sample Rate	36MHz			72MHz
IF Inputs	Single IF Input Channel			Dual IF Inputs
A/D Noise Density	-76dBm/MHz		-82dBm/MHz	-85dBm/MHz
Dynamic Range	86dB at 0.5MHz		93dB at 0.5MHz	96dB at 0.5MHz
Link to Rx	Coax uplink, Fiber downlink			Integrated CAT-5E
Upgradability	FPGA chips must be manually reprogrammed			ReFlashable via Rx-Link
Input Signal Level	A/D saturation at +4.5dBm		A/D saturation at +6.0dBm	
Ext-Clock	No	Yes (shared with AFC connector)		
Noise Generator	None. The A/D dither power must be supplied from wideband thermal noise in the RF/IF chain.		Built-in noise source supplies A/D dither power in the 200–900KHz range.	
Power Supplies	+5.17V, +12V, -12V		+5.23V Primarily. +12/15V required only for analog AFC output	+5.33V Primarily. +12/15V required for AFC, Hi-DAFC or stable VCXO (recommended). -12/15V required only for analog AFC output.
Jumpers (Table 10 on page 72)	None	AFC/Clock I/O	AFC/Clock I/O, Dither and Config Selections	AFC/Clock I/O, JTAG, DAFC/Clock and Config Selections
Uplink Protocols	AFC-16	AFC-16 & PLL-16	Supports full set of protocols defined in 3.5.1 Using the Legacy IFD Coax Uplink on page 106	
First Production	March 1997	April 1998	December 2000	August 2004

3.2.3 IFD Power, Size and Mounting Considerations

The IFD is a compact sealed module with dimensions 23.6 x 10.9 x 3.0 cm. (9.3 x 4.3 x 1.2 in). The unit is designed to be mounted on edge such that the 23.6 x 3.0 cm. surface is flush on the back of the receiver cabinet with 10.9 cm. protrusion into the cabinet. The unit is typically placed where a traditional LOG receiver would be installed. The IFD is cooled by direct conduction through its metal enclosure. It should be positioned so that air can freely convect around it, or bolted to a larger surface that will conduct the heat away.

The power supply module is separate and can be mounted nearby in the radar cabinet, or it can be attached directly to the IFD using a special

mounting bracket. The power supply and bracket will add 3.3 cm. (1.3 in) of overall width to the receiver module.

The power supply is a low noise, low ripple, switching unit; the input voltage range is 100–240 VAC 47–63 Hz, autoranging. The IFD has an internal 3-stage power supply input filter to minimize interference from the power cable. Nonetheless, it is still good practice to insure that the four supply wires (+5V, -12V, +12V, and Ground) be kept short and twisted together. A ferrite choke around the supply wires near the terminal strip is also recommended.

NOTE The inductive filtering components inside the IFD introduce a voltage drop in the +5V supply. To produce the correct internal voltage, the supply voltage measured at the external terminal block should be 5.33V for Rev.E and later, 5.23V for Rev.D, and 5.17V for Rev.C and earlier boards.

NOTE The voltage drop across the inductive filtering components causes the ground terminal of the power supply to float several tenths of a volt above chassis ground. For this reason, the IFD power supply should never be tapped to supply power to other nearby equipment.

Mounting space should also be reserved for the external analog anti-alias filters. These filters can be mounted in the radar cabinet itself, or they can be attached directly to the IFD on the opposite side of the power supply. The filters and mounting bracket will add 2.0 cm. (0.8 in) of overall width.

The 72MHz CAT-5E IFD (Rev.F) represents a factor–of–two improvement in A/D sampling rate and communications bandwidth between the IFD and RVP8/Rx card. This provides important advantages in the performance of your radar system, but it does also place greater demands on the connecting link. The CAT-5E cable carries real–time 1.080MBaud downlink serial data on three of its four twisted pairs, and uses the fourth pair for uplink communication. The data rate on each downlink pair actually exceeds the data rate for GigaBit ethernet, hence very high quality cable must be used, and maximum cable length is limited to 25-meters. There is also a minimum cable length of 2-meters.

We recommend using a shielded CAT–5E cable (certified to >= 350MHz) having shielded RJ45 plugs on each end. The Rx board provides a DC return path for the cable shield, while the IFD provides an AC GND only (isolated to 2KV). This design prevents ground loop currents from flowing between units, even when they're plugged into different AC/Mains.

3.2.4 IFD I/O Summary

The connectors on the IFD are labelled and described below for each hardware revision.

Table 5 IFD Connectors (All Revisions)

IFD I/O Summary			
Connector Label	**Style**	**Description**	**Reference**
J1 IF-IN	SMA	IF signal from LNA/mixer; via an anti-aliasing filter centered at IF (supplied by Vaisala). 50W, + 6.5 dBm max	2.2.6 2.2.7 2.2.8 2.2.10
J2 BURST (COHO)	SMA	IF Tx sample from waveguide tap and mixer; via an anti-aliasing filter centered at IF (supplied by Vaisala). 50W, +6.5 dBm max	2.2.6
J3 AFC (CLK) (CLK)	SMA	AFC output (+-10V) or reference clock input for coherent systems (2-60 MHz -10 to 0 dBm). The function of the connector is controlled by jumper selection within the IFD.	2.2.11 AFC 2.2.12 CLK

Since they share the same connector, analog AFC output and reference clock input can not be used simultaneously. However, this is a very rare requirement since an analog AFC output is used for magnetron systems and a reference clock input is typically used for fully coherent TWT and Klystron systems.

Table 6 IFD Connectors (Rev.A–Rev.D)

IFD I/O Summary			
Connector Label	**Style**	**Description**	**Reference**
J4 UPLINK	SMA/BNC	Connects to the RVP8/Rx PCI card by 75 Ohm shielded cable. The connector is SMA with an SMA/BNC adapter provided.	2.2.13
J5 FIBER-OUT	ST	62.5/125 micron multimode optical cable terminated in type ST connectors. IFD cam be located up to 100m from the RVP8/Rx PCI card.	2.2.13

Table 7 IFD Connectors (Rev.E & Greater)

IFD I/O Summary			
Connector Label	**Style**	**Description**	**Reference**
J4 DAFC(CLK)	SMA	Synthesized legacy coax uplink stream for backward compatibility, or Expansion Clock input or output.	2.2.13
J5 Rx-Link	RJ-45	Connects to the RVP8/Rx PCI card via CAT-5E cable up to 25-meters in length.	2.2.13

3.2.5 IFD Adjustments and Test/Status Indicators

The IFD is packaged in a tight metal enclosure for maximum noise immunity. The only adjustments on the module are the internal gain and offset pots that adjust the AFC analog output. Two switches on the unit provide standalone test features to verify the proper functioning of the IFD and to assist with setting the voltage span of the AFC DAC.

Table 8 IFD Toggle Switch Settings

SW1	SW2	Function
A	A	AFC Test Low Voltage
A	B	AFC Test Midpoint Voltage
A	C	AFC Test High Voltage
B	A	Swap Burst and IF Input Signals
B	B	Normal Operation (*also labeled as "run"*)
B	C	Reserved (*downlink test pattern*)
C	A	Reserved
C	B	Reserved (*downlink test pattern*)
C	C	Reserved (*downlink test pattern*)

Two LEDs provide status information for the IFD itself, as well as status of the communication link(s) to the RVP8/Rx PCI card. These LEDs have the same interpretation across all revisions of the IFD. For Rev.B through Rev.D the words "uplink" and "downlink" refer to the physical coax uplink and fiber downlink cables. For Rev.E and higher, those words refer to conceptually similar uses of the four twisted pairs within the integrated CAT-5E link.

Table 9 IFD LED Indicator Interpretations

Red (Uplink)	Green (Ready)	Meaning
Blink	Blink	Reset sequence (*powerup, or from uplink*)
Blink	Off	Uplink is dead (*no uplink protocol from RVP8/Rx*)
On	Off	Uplink is alive, but downlink is dead
On	On	Normal Operation (*IFD and Main are both okay*)

For IFDs at Rev.E and higher, the two LEDs on the RJ-45 connector also convey status about the CAT-5E link itself. Green indicates that valid clock and framing waveforms are present on the uplink. Yellow indicates that the RVP8/Rx card is receiving valid data from the IFD, including the IFD's report of the uplink being okay. These LEDs show valid status at all times (not just when the RVP8 software is running) and thus, both

indicators should be illuminated whenever the CAT-5E cable is connected. Moreover, the Green/Yellow interpretation is consistent at both the IFD and RVP8/Rx ends: green indicates the reception of proper low-level electrical andframing protocols, and yellow indicates that the green LED is ON at the other end, that is, that the other end is receiving our transmissions correctly and is able to communicate that information back to us.

The internal jumper settings are summarized in the following table. Refer also to Sections 3.2.11 IFD Analog AFC Output Voltage (Optional) on page 81 and 3.2.12 IFD Reference Clock Input (Optional) on page 82 for more information on setting up the AFC or External Clock options.

Table 10 IFD Internal Jumper Settings

	Rev.B	Rev.C	Rev.D	Rev.E and Higher
JP1	N/A	AB: AFC Voltage Output BC: External Clock Input, 50Ω Termination Open: External Clock Input, No Termination		
JP2	N/A		Reserved (Open)	AB: Normal JTAG control BC: Factory reserved
JP3	N/A		AB: Unused BC: External Clock Open: AFC Voltage Output	*Power Supply to Oscillator Use wirewrap, not jumper*AB: Regulated from +12V BC: Direct from +5V
JP4	N/A		AB: Dither Applied to Burst BC: No Dither on Burst	AB: Oscillator is a VCXO BC: Oscillator is fixed XO
JP5	N/A			Reserved for factory tests Must be left open
JP6	N/A			*J4 Protocol Selector*AB: Legacy DAFC output BC: Auxiliary CLK In/Out
JP7	N/A			*J4 DAFC Output Level*AB: +5V Signaling BC: +12V Signaling

3.2.6 IFD Input A/D Saturation Levels

There are two analog signals that must be supplied to the IFD:

- IF receiver signal
- IF Tx Sample (Burst Pulse) for magnetron, or COHO reference for klystron.

Both of these inputs are on SMA connectors. The IF signal should be driven by the front-end mixer/LNA/IF-Amp. components, similar to the way that a LOG receiver would normally be installed. The magnetron burst pulse or klystron COHO reference is also derived in the same manner as a traditional analog receiver.

NOTE	Even for fully coherent Klystron and TWT systems, Vaisala recommends the use of an actual IF Tx sample. If this is not possible, then the COHO itself may be used instead. If there is phase modulation, then the phase-shifted COHO should be input.

The A/D input saturation level for both the IF-Input and Burst-Input is +6 dBm (4.5 dBm for Rev.C or earlier). In almost all installations an external anti-alias filter is installed on both of these inputs. These filters (if supplied by Vaisala) are mounted externally on one side of the IFD, and have an insertion loss of approximately 1–2dB. Thus, the input saturation level will be +8dBm measured at the filter inputs.

For the burst pulse or COHO reference it is important not to exceed the A/D saturation level. This reference signal should be strong enough so that most of the bits in the A/D converter are used effectively, but it should also allow a few deciBels below the saturation level for safety. The recommended power level is in the range -12 to +1 dBm, measured as described in D.14 Burst Pulse Alignment on page 439. This is important for making a precise phase measurement on each pulse.

In contrast, for the IF receiver input it is permissible (in fact desirable) to occasionally exceed the A/D input saturation level at the strongest targets. The RVP8 employs a statistical linearization algorithm to derive correct power levels from targets that are as much as 6dB above saturation. The actual IF signal level should be established by weak-signal and noise considerations (see below), rather than by working backwards from the saturation level.

3.2.7 IF Bandwidth and Dynamic Range

The RVP8 performs best with a wide bandwidth IF input signal. This is because a wideband signal can be made free of phase distortions within the (relatively narrow) matched passband of the received signal. The RVP8 uses an external analog anti-aliasing filter at each of its IF and Burst inputs. The purpose of these filters is to block frequencies that would otherwise alias into the matched filter passband. The anti-alias filters have a nominal passband width of 14 MHz centered at 30MHz, that is, from 23MHz to 37MHz. This is the recommended operating bandwidth for the IF signal, although the RVP8 will still work successfully with lesser IF bandwidth.

At the 36MHz sampling rate the quantization noise introduced by LSB uncertainties is spread over an 18MHz bandwidth. For an ideal 14-bit A/D converter that saturates at +6dBm the effective quantization noise level would be:

$$+6dBm - 20log_{10}(2^{14}) - 10log_{10}\left(\frac{18MHz}{1MHz}\right) = -90dBm \ (at\ 1MHz\ BW)$$

If samples from this ideal converter were processed with a digital filter having a bandwidth of 1MHz, then an input signal at -90dBm would have a signal-to-noise ratio of 0dB. A narrower FIR passband (corresponding to a longer transmitted pulse) would decrease the quantization noise even further, so that 0dB SNR would be achieved at even lower input power.

In practice, the 14-bit A/D converter used inside the IFD does not behave quite this well. The Analog Devices AD6644 chip has been measured to have a wideband SNR of 76dB, that is, 8dB less than the 84dB range expected for an ideal converter. The above calculation for noise density thus becomes:

$$+6dBm - 76dB - 10log_{10}\left(\frac{18MHz}{1MHz}\right) = -82dBm(at\ 1MHz\ BW)$$

Indeed, the RVP8's receiver power monitor described in 5.5 Pr — Plot Receiver Waveforms on page 183 will show a filtered power level of approximately -82dBm when the FIR bandwidth is 1MHz and the IFD inputs are terminated in 50–Ohms.

The inverse correspondence between filter bandwidth and the 0dB SNR signal level leads to an interesting and useful property of wideband digital receivers: they can operate over a dynamic range that is much greater than the inherent SNR of their A/D converter would imply. If this particular A/D chip were performing direct conversion at "base band" it would have a dynamic range of only 76dB. However, by utilizing the extra bandwidth of the converter, the RVP8 is able to extend the dynamic range to approximately 100dB.

To understand this, begin with the 88dB interval between the converter's +6dBm saturation level and the -82dBm 0dB SNR level at 1MHz bandwidth. Add to this:

- 6dB for the statistical linearization that is performed on signals that exceed the saturation level. The RVP8 can recover signal power accurately even when the A/D converter is driven beyond saturation. Velocity data will also be valid, but spectral width may be overestimated.
- 4dB for usable dynamic range below the 0dB SNR level. In practice, a coherent signal at -4dB SNR can easily be measured when 25 or more pulses are used.

Thus, the overall dynamic range at 1MHz bandwidth (approx. 1 µsec transmit pulse) is 88+6+4 = 98dB. For a 0.5 µsec pulse the dynamic range

would be reduced to 95dB; but it would increase to 101dB for a 2.0 µsec pulse. An actual calibration curve demonstrating this performance is shown in Figure 15 on page 75, for which the RVP8's digital bandwidth was set to 0.53MHz and external signal generator steps of 1dB were used over the full operating range.

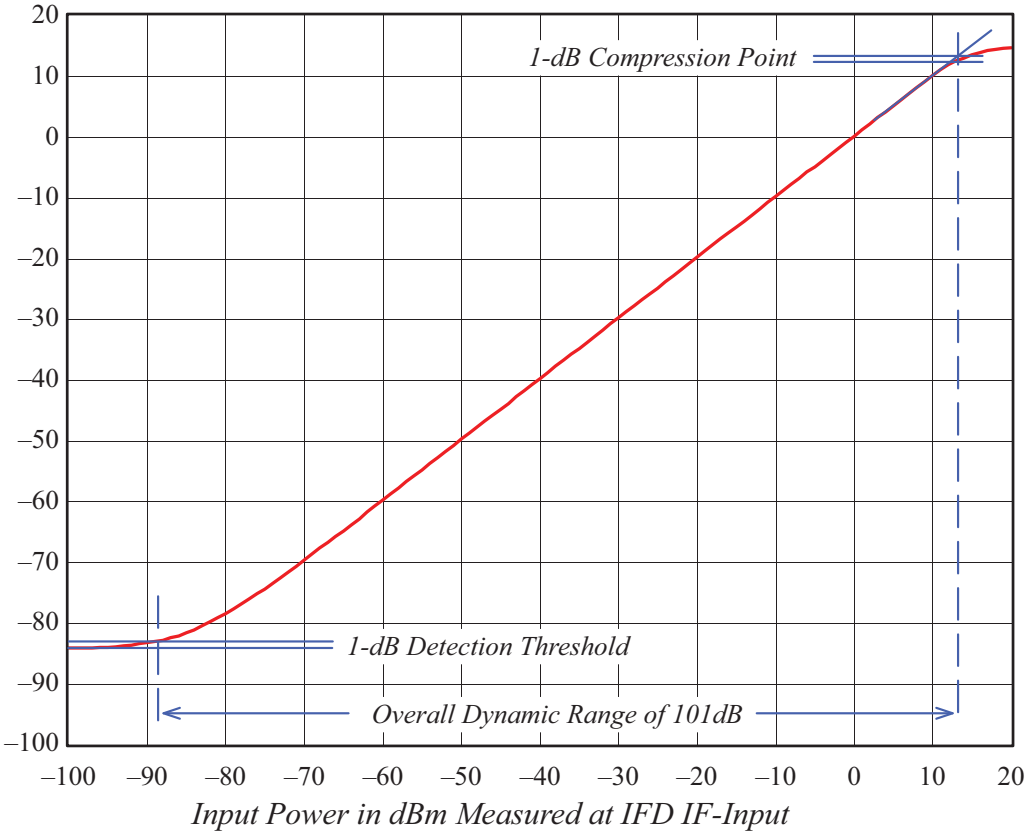


Figure 15 Calibration Plot for a Stand-alone 14-Bit IFD

3.2.8 IF Gain and System Performance

The previous discussion was concerned with measuring the dynamic range of a stand-alone IFD. We will now examine how the unit performs in the context of a complete radar receiver. We assume that an LNA/Mixer has already been selected that offers an appropriate balance between price and noise figure. Having chosen these front-end components, the only parameter that remains to be determined is the total RF/IF gain between the antenna waveguide and the IFD.

Assume that the thermal noise (kT) of the system is -114dBm/MHz, and that the noise figure of the LNA/Mixer is 2dB. We wish to bring this -112dBm/MHz noise level up into the working range of the IFD so that the received echoes can be optimally processed. However, in trying to select

the required gain, we realize that we must make a tradeoff between preserving the receiver sensitivity that has been established by the LNA, and preserving the overall dynamic range of the IFD. This is the exact same tradeoff that is made in traditional multi-stage analog receiver systems that include a wide dynamic range LOG receiver.

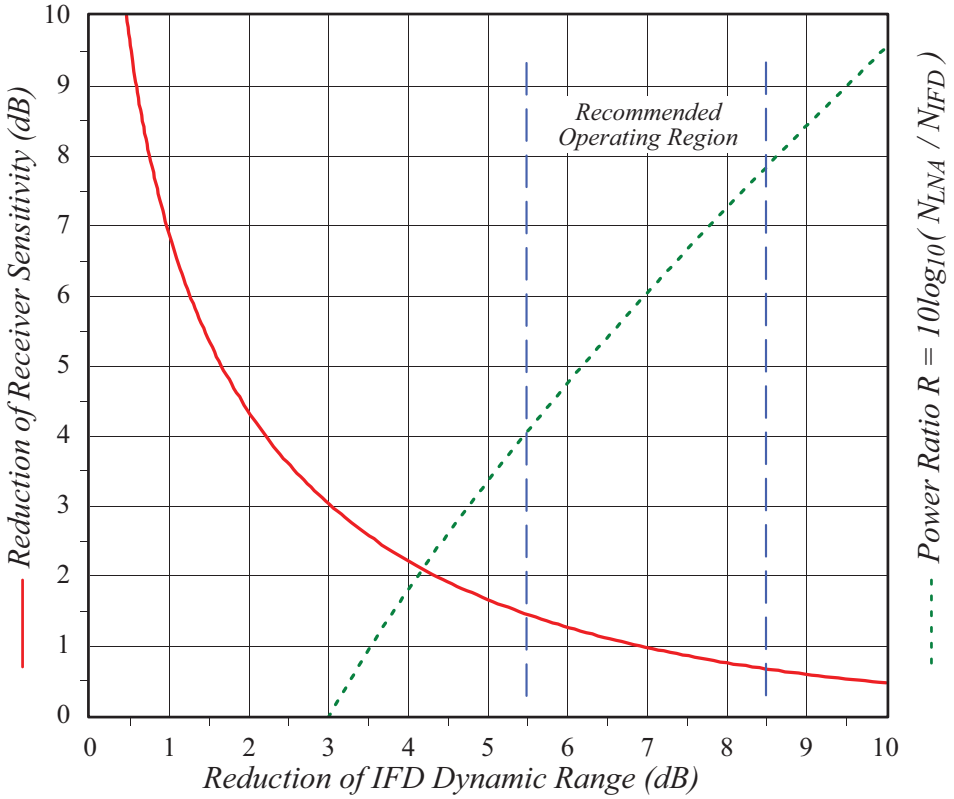


Figure 16 Tradeoff Between Dynamic Range and Sensitivity

The solid red curve in Figure 16 on page 76 shows that these two variables interact in a symmetric manner, so that any operating point (*x*,*y*) is always matched by a dual operating point at (*y*,*x*). To understand the construction of this plot, let N_{IFD} represent the stand-alone (terminated input) noise power of the IFD over some bandwidth. Similarly, let N_{LNA}represent the LNA/Mixer thermal noise power over that same bandwidth, and after amplification by all RF and IF stages. Note that N_{IFD} is primarily due to the quantization noise that is introduced by the A/D converter, whereas N_{LNA} has its origins in the fundamental thermal noise of the receiving system. The reduction of receiver sensitivity is the amount by which the LNA thermal noise is increased over the original level established by the front-end components:

$$\Delta Sensitivity = 10\log_{10}(N_{LNA} + N_{IFD}) - 10\log_{10}(N_{LNA}) = 10\log_{10}\left(1 + \frac{N_{IFD}}{N_{LNA}}\right)$$

Likewise, the reduction of RVP8 dynamic range is the amount by which the IFD quantization noise is increased over its stand-alone value:

$$\Delta DynamicRange = 10\log_{10}(N_{LNA} + N_{IFD}) - 10\log_{10}(N_{IFD}) = 10\log_{10}\left(1 + \frac{N_{LNA}}{N_{IFD}}\right)$$

Note that both of these quantities depend only on the ratio of the two powers; hence, the two equations define a parametric relationship in the dimensionless variable $R = (N_{LNA} / N_{IFD})$. Figure 16 on page 76 was created by sweeping the value of R from 1/9 to 9. The solid red curve shows the locus of (*ΔDynamicRange*, *ΔSensitivity*) points, and the dashed green curve shows R itself (expressed in dB) as a function of *ΔDynamicRange* . For example, when the LNA noise power is equal to the IFD noise power, R is 1.0 (0dB) and there will be a 3dB reduction in both sensitivity and dynamic range.

The recommended operating region is the portion of the curve that limits the loss of sensitivity to between 1.4dB and 0.65dB. The attendant loss of dynamic range will fall between 5.5dB and 8.5dB respectively. Each axis of the plot has an important physical interpretation within the radar system.

- The horizontal axis is equivalent to the increase in the RVP8's report of filtered power when the IF-Input coax cable is connected versus disconnected. This is an easy quantity to measure, and thus provides a simple way to check the overall gain of the LNA/Mixer/IF components.
- The vertical axis is equivalent to a worsening of the LNA/Mixer noise figure. This can also be interpreted as the amount of transmit power that is, in some sense, "wasted" when observing very weak echoes. If you have installed an expensive LNA with a very low noise figure, then you will want to pick an operating point that makes the most of preserving that investment.

Figure 16 on page 76 can be used to calculate the net gain that is required by the front-end components, and to predict the final system performance:

1. Choose an operating point that balances your need for sensitivity versus dynamic range. For this example, we will allow a 1dB loss of sensitivity from the theoretical limit of the LNA/Mixer, and will assume a bandwidth of 0.5MHz.
2. For a 1dB loss of sensitivity, the *ΔDynamicRange* is first determined from the solid red curve as 7dB. The required noise ratio R is then read vertically on the dashed green curve as 6.1dB.
3. Thus, the RF/IF gain must bring the front-end thermal noise at -112dBm/MHz up to a level that is 6.1dB higher than the IFD noise density of -82dBm/MHz. The gain is therefore (-82dBm/MHz + 6dB) - (-112dBm/MHz) = 36dB. Note that this gain does not depend on bandwidth, and therefore will be correct for all pulsewidth/bandwidth combinations.
4. The dynamic range for the complete system at 0.5MHz bandwidth may now be calculated as 101dB - 7dB = 94dB.
5. After assembling all of the RF and IF components we can check whether we achieved the correct gain by verifying a 7dB rise (independent of bandwidth) in RVP8 filtered power when the IF-Input cable is connected versus disconnected.

Keep in mind when designing your RF and IF components that the final amplifier driving the IFD must be capable of driving up to, perhaps, +12dBm, so that signals above saturation can still be correctly measured.

3.2.9 IF Gain Based on System Noise Figure

The previous section described how to compute the front-end RF/IF gain based on the desired tradeoff of dynamic range versus sensitivity. Since arriving at the proper gain is so important, we present an alternate but equivalent approach based on system noise figure.

Every amplifier can be partially characterized by its gain "*G*" and noise figure "*F*". Gain is measured quite simply by injecting a test signal at the mid-power range of the amplifier and measuring the ratio of Output/Input power. Noise figure is a little trickier, and is measured by terminating the input of the amplifier, measuring the output power within some prescribed bandwidth, and then dividing by the thermal noise power expected over that same bandwidth from an ideal amplifier having the same gain. For example, suppose that an amplifier with a gain of 20dB delivers -90dBm of output power within a 1MHz bandwidth when its input is terminated. We would expect the Boltzman thermal input noise at -114dBm/MHz to

produce -94dBm from an ideal 20dB amplifier under the same conditions. The noise figure of the real amplifier is therefore +4dB, that is, -90 minus -94.

Although the above definitions are typically applied to linear analog amplifiers, these same terms can be applied to hybrid analog/digital systems such as the RVP8.

- To calculate the gain of the RVP8/IFD we apply a calibrated mid-power signal generator directly to its IF-Input and use the **Pr** plot (5.5 Pr — Plot Receiver Waveforms on page 183) to print the measured power. For a wide range of analog input power levels the RVP8 will report the exact same measured digital power; hence the overall analog/digital gain is 1.0 (0 dB).
- To calculate the noise figure of the RVP8/IFD, we set the receiver bandwidth to 1MHz (5.4.2 Available Subcommands Within Ps on page 171), terminate the IF-Input in 50-Ohms, and again use the **Pr** plot, this time to examine the in-band thermal noise power. For the Rev.D IFD this measured noise level will be around -82dBm. Since an ideal unity gain amplifier would have produced a noise power of -114dBm in an equivalent bandwidth, the noise figure of the RVP8/IFD is 32dB.

When two amplifiers are cascaded so that the output of the first drives the input of the second, the overall gain is the product of the two linear gains G^1_{lin} and G^2_{lin}, and the overall noise figure is computed from the two noise factors F^1_{lin} and F^2_{lin} as:

$$NoiseFigure = 10\log_{10}\left[F^1_{lin} + \left(\frac{F^2_{lin} - 1}{G^1_{lin}}\right)\right]$$

where the two noise factors are simply the linear representations of the noise figures that were expressed in deciBels:

$$NoiseFigure = 10\log_{10}[NoiseFactor]$$

Suppose that our first amplifier is an LNA/Preamp with a 2dB noise figure (noise factor 1.58), and we want to know what gain it must have such that, when cascaded into the RVP8/IFD, the overall noise figure will be 3dB. The 32dB noise figure of the IFD is equivalent to a noise factor of 1585, hence we have:

$$3dB = 10 \log_{10}\left[1.58 + \left(\frac{1585 - 1}{G^{1}_{lin}}\right)\right]$$

from which we solve $G^{1}_{\text{lin}} = 3771$, that is, 35.8dB. This agrees with the 36dB of gain that was computed in the example of the previous section for the same RF/IF components and desired overall performance.

3.2.10 Choice of Intermediate Frequency

The RVP8 does not assume any particular relationship between the A/D sample clock and the receiver's intermediate frequency. You may operate at any IF that is at least 2MHz away from any multiple of half the 35.9751MHz sampling rate (nominally 18, 36, 54, 72 MHz). The valid frequency bands are thus:

6-16MHz, 20-34 MHz, 38-52 MHz, 56-70 MHz

There are many reasons for staying clear of the Nyquist frequency multiples. Most of these considerations would apply to all types of digital processors, and are not specific to the RVP8.

As an example of what can go wrong at the Nyquist frequencies, suppose that an intermediate frequency of 35MHz was used. This is only 1MHz away from the (approximately) 36MHz sampling rate. The external anti-alias filter must now be designed much more carefully since a spurious input signal at 37MHz would be aliased into the valid 35MHz band. If the valid signal bandwidth were 2MHz, then the anti-alias filter would have the difficult task of passing 34–36MHz free of distortion while rejecting everything above 36MHz. The filter's transition zone would have to be very sharp, and this is difficult to achieve.

Another problem that would arise with a 35MHz IF on a magnetron system would be the RVP8's computation of AFC. If the processor can not distinguish 37MHz from 35MHz, then it can not tell the difference between the STALO being correctly on frequency, versus being 2MHz too high. The symmetric AFC tracking range would be reduced to the very small interval 34–36MHz.

For similar reasons (that is, transition band width), the digital FIR filter itself also becomes difficult to design when its passband is near a Nyquist multiple. But there is an additional constraint that the digital filter should have a very large attenuation at DC. This is so that fixed offsets in the A/D converter do not propagate into the synthesized "I" and "Q" data. Since

36MHz is aliased into DC, we are left with the contradictory requirements of a zero very close to the edge of the filter's passband.

3.2.11 IFD Analog AFC Output Voltage (Optional)

An analog AFC voltage is produced by a 16-bit DAC whose output limits are -10V to +10V. Gain and Offset potentiometers on the IFD module set the actual operating span within these limits. Use the switch settings described below to force the low, center, and high voltages to be output, and then adjust the two potentiometers so that the desired voltage span is achieved. The Offset adjustment is independent of the Gain adjustment. Hence, a good strategy is to first set the switches for the midpoint voltage, and adjust the Offset potentiometer so that the center IF frequency is produced by the STALO mixer. Then, adjust the Gain potentiometer for the desired tuning range around that center point. The midpoint voltage will not change as you vary the overall span.

AFC voltage output is always enabled on Rev.B (and earlier) IFD boards. On Rev.C (and later) boards, the AFC function shares the same connector with the optional reference clock input (See 3.2.12 IFD Reference Clock Input (Optional) on page 82). AFC can be enabled on a Rev.C board as follows:

- Remove U14
- Install U11, U12, U13
- Set JP1 to its AB position, which is also labeled "AFC".
- Install fixed frequency stable 35.975MHz oscillator at U5.

The instructions are similar for a Rev.D board except that you do not need to remove U14, and you must check that no jumper has been placed on JP3/BC (See Table 10 on page 72).

Additional information about using AFC can be found in Sections 3.4 Digital AFC Module (DAFC) on page 99, 4.2.7 Mb — Burst Pulse and AFC on page 142, and 6.1.3 Automatic Frequency Control (AFC) on page 210.

3.2.12 IFD Reference Clock Input (Optional)

When the RVP8 is used in a klystron system, or in any type of synchronous radar, the radar COHO is supplied to the IFD so that the processor can digitally lock to it. The COHO phase is measured at the beginning of each transmitted pulse, and is used to lock the subsequent (I,Q) data for that pulse. The COHO phase is measured relative to the IFD's own internal stable sampling clock, which is nominally 35.975MHz. The internal sampling clock itself is not affected by the application of the COHO. Rather, A/D samples of the COHO are obtained at the fixed sampling rate, and the (I,Q) data are digitally locked downstream in the RVP8 IF-to-I/Q processing chain (see Figure 12 on page 42). The procedure is identical to the manner in which phase is recovered in a magnetron system, except that the COHO signal is used in place of a sample of the transmit burst.

There are two special concerns that may come up when the RVP8 is used in the above manner within a synchronous radar system. Both concerns are the result of the IFD's sampling clock being asynchronous with the radar system clock.

- *RVP8 Generates the Radar Trigger*

  The trigger signals supplied by the RVP8 are synchronous with the IFD data sampling clock. This is accomplished by a clock recovery PLL on the RVP8/Rx that provides on-board timing which is identical to the sampling clock in the IFD. However, since the IFD sampling clock is asynchronous with the radar clock(s), the RVP8 trigger outputs are likewise asynchronous. The result is that each transmitted pulse envelope will be triggered independently of the COHO phase. The transmitted pulse is still synchronous — but the precise alignment of the amplitude modulated envelope will vary.

- In almost all cases, the exact placement of the transmitter's amplitude envelope does not affect the overall system stability, nor the ability of the RVP8 to reject ground clutter and to process multi-mode return signals. For this reason, a synchronous radar system that is triggered using the RVP8 triggers will still perform optimally using the standard digital COHO locking techniques. In spite of this, however, some system designers may still prefer that the amplitude envelope itself be locked to the COHO.

- *RVP8 Receives the Existing Radar Trigger*

  When an external trigger is supplied to the RVP8, the processor synchronizes its internal range bin selection circuitry to that external trigger. The placement of the range bins themselves, however, is always synchronous with the IFD's 35.975MHz acquisition clock.

The result is that 27.8ns of jitter is introduced in the placement of the RVP8's range bins relative to the transmitted pulse itself.

- The effect of this synchronization jitter is that targets appear to be fluctuating in range by approximately 4.2 meters. Although this is small relative to the range bin spacing itself, and thus does not affect the range accuracy of the data, the effect on overall system stability is more severe. Using both numerical modeling and actual field measurements, we have found that sub-clutter visibility of a µsec pulse may be limited to approximately 43dB as a result of this 27.8ns range jitter. This falls quite short of the usual expectations of a synchronous radar system in which clutter rejection of 55–60dB should be attainable.

The solution to either of the above concerns is to provide some means for the IFD's internal sampling clock to be phase locked to the radar system. If the RVP8 provides the radar triggers, then those triggers would become synchronous with the radar COHO; and if the RVP8 receives an external trigger, then its range bin clock would be synchronous with that external trigger, and thus, there will be no synchronization jitter in the range bins.

Beginning with Rev. C, the IFD offers the option of locking itssampling clock to an external system clock reference. This results in an RVP8 that is fully synchronous with the existing radar timing. Rather than being derived from a fixed-frequency oscillator, the phase locked IFD sampling clock is driven by a custom Voltage-Controlled-Crystal-Oscillator (VCXO). This oscillator can have a center frequency in the 33.5 to 39.5MHz range, which is any rational multiple P/Q of twice the input reference frequency, where P and Q are integers between 1 and 128 (See also, 4.2.7 Mb — Burst Pulse and AFC on page 142). The tuning range of the VCXO is purposely kept very narrow (to improve the clock stability), and is restricted to approximately +/-50ppm. Thus, the input reference clock frequency (range 2–60MHz) must be precisely specified so as to stay within these limits. The reference clock input power level level should be between -10 and 0dBm.

Use the following configuration to allow a Rev.C IFD to lock its sampling clock to an external reference:

- Install U14
- Remove U11, U12, U13
- Set JP1 to its BC position to terminate the reference input in 50Ω, or leave the jumper open to achieve a high-impedance input (approx. 5KΩ).
- Install custom Voltage-Controlled-Crystal-Oscillator (VCXO) at U5. Contact Vaisala for assistance in specifying this device.

For a Rev.D board the instructions are similar except that you do not need to remove any components, and should place a jumper on JP3/BC (See Table 10 on page 72).

NOTE	As described in the previous section, for Rev.C boards U14 *Must Be Removed* whenever the VCXO phase lock mode is not being used, that is, when a free-running crystal is installed.

3.2.13 Communications Between the IFD and RVP8/Rx

For all revisions of the IFD hardware, the RVP8 software measures the round trip cable delays each time it boots up and then uses that information to correct for range and timing offsets due to cable length.

In the Rev.A through Rev.D IFD modules there are two cable links between the IFD and the RVP8/Rx PCI card. These cables can be any length up to 100 meters apiece.

- Copper coax uplink from the RVP8/Rx board. This cable provides timing information for the burst pulse window, and 16-bit data for setting the AFC output level. The uplink input from the RVP8/Rx is an SMA connector from a 75Ω shielded cable (for example, RG59 cable). This cable is electrically isolated from the receiver's ground (40KΩ isolation) so that any noise or ground loops picked up by the cable will not be coupled into the receiver circuitry. Details of the uplink protocol can be found in 3.5.1 Using the Legacy IFD Coax Uplink on page 106.
- Optical fiber downlink to the RVP8/Rx board. The receiver and burst pulse data samples are encoded into a 540MHz 8B/10B serial stream. The downlink operates at the infrared wavelength of 850nm, using a 62.5/125 micron multimode optical cable terminated in type ST connectors.

In the Rev.E and higher IFD modules there is a single CAT-5E cable (quad twisted pair) with RJ-45 connectors that links the IFD to the RVP8/Rx PCI card. This cable carries GigaBit rate data and should be of high quality construction. The cable length can be up to 75 feet, but a minimum length of six feet is also recommended. Electrical isolation at the IFD side is >2KV for the data pairs and outer cable shield. In most radar applications it is highly recommended that shielded twisted pair cable be used rather than the more common unshielded variety. The DC shield ground is established only at the RVP8/Rx side to avoid ground loops between the IFD and PCI chassis.

Three of the four CAT-5E twisted pairs are used as dedicated downlink channels, and the fourth pair carries a dedicated uplink channel. Thus, each driver/receiver operates in a single direction only, that is, the data direction is fixed on each wire pair. The RJ-45 connector on both the IFD and Rx cards is a Gigjack T12 (JK0-0016 from Pulse Engineering, www.pulseeng.com).

- The three downlink channels are identical and use RJ-45 line pairs MX1, MX2 and MX3. Each line is driven from the PECL outputs of a Cypress CY7B923 "Hotlink" transmitter via 33Ω series resistors. Each transmitter chip produces an independent 360MBaud data stream using 8B/10B encoding, yielding an aggregate line rate of 1080MBaud (payload rate of 864MBaud). The spectral characteristics of each downlink twisted pair can be predicted entirely from this baud rate and encoding technique. Maximum differential delay (propagation skew) among the three downlink pairs must be less than 25ns.
- The single uplink channel uses RJ-45 line pair MX4, and is driven by a PECL transmitter having 33Ω series output resistors. The uplink data rate is 72MBaud, and is presented in a manner having nearly the same spectral characteristics as Manchester encoding.

The exact content of the four CAT-5E data streams is beyond the scope of this manual, but the above descriptions should be sufficient to understand the bandwidth and electrical properties of the signals on each twisted pair. This information is applicable for cable selection, or toward the design of a repeater to carry the CAT-5E signals over different media.

3.2.14 Summary of Crystal and Filter Configurations

The RVP8/Rx, RVP8/Tx and IFD can operate in many different clocking and sampling configurations, depending on the requirements of the radar in which they are installed. The following summary describes how to setup the crystals and filters in your equipment.

Step 1. Choose the Intermediate Frequency

Custom analog bandpass filters are installed in the RVP8/Tx and IFD to match to the radar's Intermediate Frequency. The RVP8/Tx filters are soldered directly onto the PCI card and are not really meant to be changed by the user. The IFD filters, however, attach via coax cables and are located on a serviceable mounting plate attached to the IFD.

Simply verify that the center frequency of all of your bandpass filters match the IF of your radar. Our standard filter frequencies are: 16MHz, 30MHz, 57.5MHz, and 60MHz.

Step 2. Choose Clock Locking Options

The RVP8 acquisition and trigger clocks can operate as free-running oscillators, or they can be phase locked to an external reference signal.

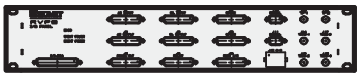
- For simple magnetron radars there is generally no system reference clock, and the free-running clock mode is therefore appropriate. However, if a synthesized STALO is being used as the RF source, then you may want to lock your RVP8 to the STALO's own reference clock (generally 10MHz). For dual-pol magnetron systems you must lock the RVP8 in this manner to measure differential phase.
- For klystron and other synchronous radars there will always be some kind of reference clock in the system. In the simplest case of a single-pol radar in which the RVP8 fires the transmitter directly, you may be able to get away with using free-running clocks. However, in virtually all cases it is best to lock the RVP8 to (one of) the radar's existing timing reference(s).

The following table lists the different clock modes for the RVP8/Rx, RVP8/Tx, and IFD. The frequency of the required crystal oscillator is given, along with the type that is required. The VF155 oscillators are general purpose units, VFAC170 is low-jitter free-running, and VF940 is low-jitter Voltage Controlled Crystal Oscillator (VCXO).

Table 11 Clock Locking Component Options

Lock Mode	RVP8/Rx	RVP8/Tx	IFD
Free-Running	U16: 26.983 (VF155)	U9 : 80.944 (VF940)	35.975 (VFAC170)
	The IFD is free-running, and the Rx and Tx cards lock to it. Use this mode for simple magnetron radars in which there is no system reference clock.		
N-MHz External Reference	U16: 26.983 (VF155)	U9 : 81.000 (VF940)	36.000 (VF940)
	A reference clock at some integer N-MHz is applied to both the IFD and Tx inputs, both of which lock directly to it. This is the recommended hookup for klystron radars or magnetron radars that use a synthesized STALO derived from, for example, 10MHz.		
57.5491 MHz WSR88D COHO	U16: 26.983 (VF155)	U9 : 80.928 (VF940)	35.968 (VF940)
	For NEXRAD systems, the COHO is fed directly into both the IFD and Tx inputs. The IFD sampling rate is 5/8 of the COHO frequency, and the Tx rate is 45/32.		
31.0703 MHz ASR-9 COHO	U16: 27.186 (VF155)	N/A	36.248 (VF940)
	These special frequencies are used for the ASR-9 surveillance radars. The IFD locks to 7/6 times the COHO frequency of the radar.		

3.3 RVP8 Chassis



3.3.1 RVP8 Chassis Overview

The RVP8 main chassis can assume a variety of forms depending on the customer requirements. Appendix C, RVP8/RCP8 Packaging, on page 393 describes a standard Vaisala system. A typical unit supplied by Vaisala contains at least the following:

- A dual CPU on either motherboard or SBC in a passive PCI backplane
- RVP8/Rx Card
- I/O-62 Card and Connector Panel

The system is also shipped with an integrated hard disk drive (HDD), 1.44 MB floppy (FDD) and CDRW unit. Note some installations may use a flash disk drive instead of an HDD. There is an LED display panel on the front of the chassis that is used to report system status.

3.3.2 Power Requirements, Size and Physical Mounting

WARNING	The Main Chassis redundant power supplies are NOT auto-ranging like the IFD. These are factory configured for the expected voltage, but should be VERIFIED by the customer before power is applied to the system.

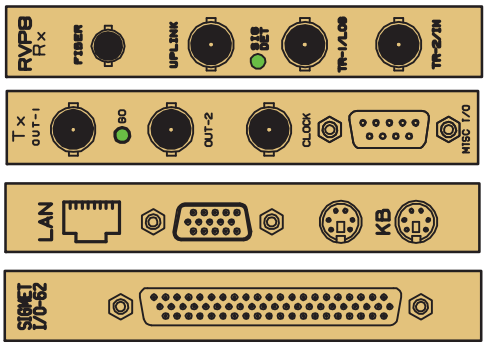
There a three redundant power supplies.

The standard Vaisala chassis is a 19" EIA 4U rackmount unit, 17" (43 cm) deep. The chassis is usually mounted in a nearby equipment rack on rack slides (provided as standard). The connector panel is usually mounted on

either the front or the rear of the same rack. The standard cable provided to connect the I/O-62 card in the main chassis to the connector panel is 6 feet long (1.8 m) .

The power requirements are 100–240 VAC 47–63 Hz. The IFD is autoranging, that is, there are no switches or jumpers that must be set. However, the main chassis is not. To check that the power supply is properly configured for your line voltage, follow the procedure in C.1.3 Main Chassis Back Panel Power Section on page 401.

3.3.3 Main Chassis Direct Connections



The direct connections to the RVP8 chassis are made either to the back of the unit to PCI cards (for example, left) or to the remote connector panel. The direct connections are summarized in the table below.

Table 12 Direct Connections to RVP8 Main Chassis

IFD I/O Summary			
Connector Label	**Style**	**Description**	
Rx Card Connections			
Uplink	BNC	COAX uplink to IFD (75-Ohm shielded cable)	For IFD Rev.A-D
Fiber	ST	Fiberoptic downlink from IFD (orange cable)	
IFD-Link	RJ-45	CAT-5E (UTP) link between IFD and RVP8/Rx	For IFD Rev.E
Misc-I/O	DB9F	Four RS-422 I/O lines for future expansion	
TR-1 / LOG	BNC	Trigger outputs (5 or 12V, 75-Ohm) or pre-trigger input (1.8V thresh-old 75 Ohm) Jumpers on the card select the function.	
TR-2 / PreTrigger In	BNC		
SBC or Motherboard Connections			
Network	RJ-45	10/100/1000 BaseT TCP/IP	
Keyboard	PS/2	Standard PC Keyboard	
Mouse	PS/2	Standard PC Mouse	
Monitor	VGA	Standard PC Video Monitor	
I/O-62 Connections			
<no label>	DB-62F	Vaisala-supplied cable to IO62/CP remote panel	
Optional Tx Card			
IF Out 1	BNC	Two independently synthesized IF output waveforms, up to +12dBm into 50 Ohm 8 75MHz.	
IF Out 2	BNC		
CLK	BNC	Optional input or output reference clock (50-Ohm)	
Misc	DB-9F	Four optional RS-422 clocks or control lines	

Depending on the installation, the jumpers on the I-O 62, Rx and Tx Cards may require configuration. These are described in Appendix C, RVP8/RCP8 Packaging, on page 393.

3.3.4 External Pre-Trigger Input

Users may supply the RVP8 with their own CMOS-Level pre-trigger for installations in which adequate trigger control already exists. The trigger input is provided directly on the Rx Card (bottom BNC connector on the card panel labeled "TR-2 / In). See Appendix C, RVP8/RCP8 Packaging, on page 393 for instructions on how to set the Rx card jumpers to enable a pre–trigger input.

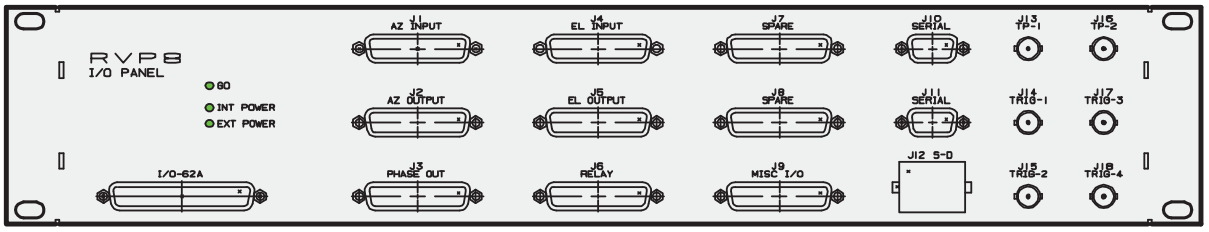
The trigger input uses CMOS levels (1.5V max low, 2.5V min high) for improved noise immunity. The trigger input may also be driven as high as +100V or as low as -100V without damage. This makes it easier to connect to existing high-voltage trigger distribution systems. The rising or falling

edge of this external "TRIGIN"signal is interpreted by the RVP8 as the pretrigger point; the actual pulsewidth of the signal does not matter. The delay to range zero is configured via the TTY Setups. The other trigger outputs are then synchronized to the input trigger. The synchronization jitter between the user pretrigger and the other trigger outputs is less than 0.014 microseconds.

Trigger jitter can be improved in the case of coherent systems, by phase locking the IFD to the same reference clock used to generate the external triggers (typically the COHO). This provides approximately 10 dB of additional phase stability.

The RVP8's response to a missing external trigger is that the processor will insert fake (software) "triggers" at a rate of 250Hz whenever the trigger input is missing for more than 0.100 seconds. These fake triggers will keep the RVP8's internal code and external outputs running in spite of the missing input (the data values will all be zero, and the "no trigger" bit will be set in GPARM immediate status word #1). Normal operation automatically resumes as soon as the external trigger is restored.

3.3.5 Connector Panel I/O Connections



Most of the connections between the radar and the RVP8 are made using the RVP8 Connector Panel which connects to the I/O-62 by 1.8m (6 foot) cable. The panel is usually mounted on the front or the back of the same 19" EIA rack that contains the RVP8 chassis. The I/O-62 cable may be plugged into either the front or the back of the connector panel to optimize the cable run.

The Connector Panel uses a DC–DC converter to convert 12V unregulated input from the PCI card into regulated +5V, +3.3V, and +/-12V to run the main electronics on the panel. The LEDs on the panel are described below:

- **EXT LED** indicates that the 12V input power is present
- **INT LED** indicates that +3.3V is present
- **GO LED** indicates that the panel is properly communicating with the PCI card. It will blink slowly when communication is absent and very rapidly during the BRIEF times that the backpanel firmware are being updated with an rdaflash command. It will be solid when the panel is being used by the RCP8 software.

The table in 2.9.5 I/O-62 Standard Connector Panel on page 60 provides a summary of the I/O for each connector. Detailed pin–out assignments are given in Appendix C. Descriptions of the various signals are provided below.

J1 & J4- AZ/EL Input: TTL parallel angles

Thirty two TTL-Level input lines. These are sampled by the RVP8, and the bits accompany each processed output ray (See PROC command, 7.7 Initiate Processing (PROC) on page 295). The inputs can also be read directly via the GPARM command (See 7.9 Get Processor Parameters (GPARM) on page 309). The RVP8 supports an antenna synchronizing mode and inserts the AZ and EL start and stop angles into the ray header of each radial (nominally 1 degree). Whenever antenna angle data are required, the processor reads the azimuth lines up to ten times in a row (spaced by 0.5 µsec) until two successive values compare equal. This is done so that unsynchronized input data will be latched in a valid state. If after ten retries the lines were never observed in a consistent state, then the last observed state is used. Sampling for elevation is identical.

The format can be BCD or binary angle. Detailed pin assignments are given in Appendix B.

J2 & J5- AZ/EL Output: TTL parallel angles

These provide output of the AZ and EL angles in TTL BCD or binary angle format. Detailed pin assignments are given in Appendix B. This feature could output the parallel angles to a separate antenna controller for example.

J3- PHASE OUT: 8-bit RS422 phase shifter control output

Can be used as differential RS422 or as single–ended TTL. This is used to control a phase shifter for coherent systems that use phase modulation, but do not have a Tx card. This is typically used for legacy systems.

J6- RELAY: Control for external equipment

Often, external equipment in the radar will require relay control (for example, power on, radiate on, environmental systems, reset lines, slow

polarization switch). This connector has connections for 3 internal relays that are on the connector panel itself. The maximum current through the relay contacts is 0.5 A continuous. The switching load is 0.25 A and 100V, with the additional constraint that the total power not exceed 4VA.

If larger current and voltage loads are required, then the connector panel relays can be used to switch external relays provided by the customer. Another alternative is to use the additional 4, 12V relay signals (up to 200mA) that are also supported on this connector.

WARNING	External relays must be equipped with proper diode protection against back-EMF or damage to the I/O-62 and or the connector panel might result.

J7 SPARE: Configurable 20 lines of TTL I/O

This connector supports 20 lines of TTL each of which can be configured as either input or output via the softplane.conf file.

J8 SPARE: Analog Inputs

10 differential analog inputs, up to ±20V max multiplexed into a single A/D convertor sampling each at >1000 Hz. This can be used for monitoring environmental systems at the radar site.

J9- MISC: RS422 I/O, D/A and A/D

7 additional RS422 lines, each configurable to be either input or output, and 2 each dedicated (non–multiplexed) A/D inputs (±580V with pot adjust) and D/A outputs (±10V). The RS422 lines are convenient for high-speed polarization switch control.

J10-11: RS232C I/O

These two connectors can be used for serial angle input. The most common format is the RCV01 format, although custom formats from antenna/pedestal manufacturers such as Orbit, Andrew and Scientific Atlanta are also supported.

J12: S-D- AZ and EL synchro input

For systems that have synchros, the RVP8 can accept a direct synchro input from both AZ and EL. The nominal voltage and frequency are 100V @ 60 Hz. S/D conversion is performed in the I/O-62.

J13-14: TP1 & TP2: Programmable test point scope outputs

Am exciting feature of the RVP8 is the programmable test points. These are usually used to connect to an oscilloscope. The user can then specify what is output to the test points in the form of an analog voltage for display on the scope. Some examples are:

- "LOG" receiver power output (an old–time radar A–Scope)
- Burst pulse
- Analog input monitor

The advantage of using the test points is that technicians can leave them permanently connected to a rackmount oscilloscope and then select what is displayed. This saves time and reduces cabling errors when test switching cables.

J15-18: TRIG1-4- Output triggers

The waveforms appearing on the four trigger outputs are programmed by the user to meet the radar's exact timing needs. These correspond to the trigger generators TGEN1, TGEN2, TGEN3 and TGEN4. More triggers can be configured on the "SPARE" connectors if they are required. All lines may be setup and used independently and can contain, for example, pre-trigger pulses, calibration gates, range strobes, scope triggers, etc. The triggers are driven at +12V into 75Ω and can be independently-timed at rates between 50Hz and 20000Hz with better than 0.02% accuracy. For dual-PRF velocity unfolding applications, the RVP8 trigger generator must be used as opposed to an externally supplied pre-trigger (see next section).

The timing of the triggers is phase-locked to the sample clock in the IFD, which can be phase locked to the COHO of a coherent system. For coherent systems that do not sample the actual transmit pulse (for phase correction), this is recommended.

The trigger waveforms are configurable in software using the "mt" commands. This sets the trigger timing, trigger sense (active high or active low pulse) and the minimum and maximum PRF for each pulse width. See 4.2.4 Mt — General Trigger Setups on page 130.

NOTE	It is sometimes useful to dedicate one of the TRIG outputs to trigger an oscilloscope. 3.3.4 External Pre-Trigger Input on page 89 for a description of how to configure an input pre-trigger from an external source such as an existing radar trigger system.

Selectable input pre-trigger

Users may supply the RVP8 with their own CMOS-Level pre-trigger for installations in which adequate trigger control already exists. The trigger input is provided directly on the Rx Card (bottom BNC connector on the card panel). The trigger input uses CMOS levels (1.5V max low, 2.5V min high) for improved noise immunity. The trigger input may also be driven as high as +100V or as low as -100V without damage. This makes it easier to connect to existing high-voltage trigger distribution systems. The rising or falling edge of this external "TRIGIN"signal is interpreted by the RVP8 as the pretrigger point; the actual pulsewidth of the signal does not matter. The delay to range zero is configured via the TTY Setups. The other trigger outputs are then synchronized to the input trigger. The synchronization jitter between the user pretrigger and the other trigger outputs is less than 0.014 microseconds.

Trigger jitter can be improved in the case of coherent systems, by phase locking the IFD to the same reference clock used to generate the external triggers (typically the COHO). This provides approximately 10 dB of additional phase stability.

The RVP8's response to a missing external trigger is that the processor will insert fake (software) "triggers" at a rate of 250Hz whenever the trigger input is missing for more than 0.100 seconds. These fake triggers will keep the RVP8's internal code and external outputs running in spite of the missing input (the data values will all be zero, and the "no trigger" bit will be set in GPARM immediate status word #1). Normal operation automatically resumes as soon as the external trigger is restored.

3.3.6 Power-Up Details

WARNING	The Main Chassis redundant power supplies are NOT auto-ranging like the IFD. These are factory configured for the expected voltage, but should be VERIFIED by the customer before power is applied to the system.

When the RVP8 is powered–up or reset, the host Linux PC goes through an automated boot process that ultimately starts the RVP8 application. The RVP8 then runs extensive internal diagnostics. In most cases, there is no display connected to the RVP8 to monitor the boot sequence. For troubleshooting it is useful to connect a display to view any error messages.

The RDA front panel display shows the status of the Linux boot sequence and summary status of the diagnostic self-tests. During the Linux boot stage, the front panel shows the following which indicates how much time has elapsed since the start of the boot process.

```
SIGMET Inc.          Open RDA
Boot < 00:24 >
```

After the Linux boot process is complete, the RVP8 runs its internal self-tests during which the front panel display appears as follows.

```
RVP8 V4.3            Starting
Running
Diagnostics
```

After successful completion of the self-tests, the display will show the following,

```
RVP8 V4.3            Starting
Diagnostics Pass
```

and then switch to the standard operational display which shows the current azimuth and elevation, the major mode, the number of range bins, the PRF and XXX:

```
179.23    AZ/EL                14.96
FFT       1646      1800 Hz    x1
```

In the event that the diagnostics do not pass, then the display will indicate the number of tests that "Fail". Note that the "dspx –nochat" utility "gparm" command or the "v" command in the TTY setups can be used to learn the details of the failures.

3.3.7 Socket Interface

The RVP8 as shipped is configured to listen on a network port. It is ready to interface to a host computer via the network using a program called **DspExport**. It is also ready to run some commands on the RVP8 itself. The RVP8 comes with some built–in Vaisala supplied utilities such as **setup, dspx** and **ascope**. These utilities are described in the *IRIS Utilities Manual*. Because the RVP8 can only have one program controlling it at a time, use of a local program like **dspx** will block network access, and vice versa.

How DspExport Works

DspExport is a daemon program which is normally configured to run all the time. When it receives a socket connection request it will establish a connection to the RVP8. At this point, multiple connections are allowed. It will only handle the "INFO", "SETU" and "OPEN" commands. Once the "OPEN" command is sent, an exclusive connection for I/O to the RVP8 is established. If a second OPEN request comes in while the first is still active, it will fail, and return the message "Device allocated to another user". To see if it is running on your RVP8, try typing

```
$ ps –aef | grep DspExport
```

During development, it can always be started up manually by typing "DspExport" at a shell prompt. It can be started with the "–v" option for more detailed logging. It defaults to using port 30740. If you wish to use another port, start it with an option such as "–port:12345". The command line option "–help" lists these options.

Source Examples

The source code for **DspExport** and for the dsp library is supplied on the RVP8 release cdrom. This can be optionally installed as part of the upgrade procedure as discussed in *Software Installation manual*. You will find **DspExport** in ${IRIS_ROOT}utils/dsp, and you will find the dsp library in ${IRIS_ROOT}libs/dsp. In the library, you will find example code which talks to **DspExport** in file OpenSocket.c, dsp_read.c and dsp_write.c. Search for the string "SOCKET", and you can see how the code differs between SCSI interface and socket interface.

Socket protocol

The socket interface basically supports all the "Host Computer Commands" in Chapter 7, Host Computer Commands, on page 275. There are a few layers of formatting on top of that. All messages going both ways consist at the lowest level of an 8-character decimal ASCII number, followed by a block of data. The decimal number indicates how many bytes follow. Generally, all data transfers are initiated by the host computer by sending a block of data which consists of a command word followed by the "|" character, followed by optional data.

It will respond to all commands with either an "Ack|" indicating acknowledgment that the command was OK, or "Nak|" indicating that there was an error. For Nak, the reply will always include a string indicating what the error was. For Ack there is optional data following.

On initial socket connection request **DspExport** will provide a response of either Nak indicating the connection failed, and why, or Ack followed by

some connection information. This Ack string is in the form of name/value pairs, and will look something like:

```
Ack|CanCompress=1,Model=RVP8,Version=7.32
```

Your program can choose to evaluate or ignore any of these keywords. "CanCompress=1" indicates that the **DspExport** computer supports compression. The host computer can then choose to use compression if it wants to. When you first connect, you are in the "info only" mode. That means that the server will only respond to INFO and OPEN commands. **DspExport** supports only the 6 commands discussed individually below:

Read command (READ)

Example: "READ|100|" means read 100 bytes from the RVP8. Since the RVP8 interface is a 16-bit word interface, these read sizes should always be even. It will always reply with a "Ack|" followed by 100 bytes of binary data, or with a "Nak|", in other words there can be no partial reads.

Write command (WRIT)

Example: "WRIT|<data>" Where <data> is some binary data. This data is written to the RVP8. Again, the data size should be even.

Read Status command (STAT)

Example: "STAT|" This reads the status bits back from the RVP8. This is a 1 bit value, set to 1 if the RVP8 has data available in its output buffer. It will return either "Ack|0", or "Ack|1", or a "Nak". This is the equivalent of the dspr_status() call in the dsp library.

Set Information command (INFO)

Example:
"INFO|ByteOrder=LittleEndian,WillCompress=1,Version=7.32". This command can be used to inform RVP8's **DspExport** about the host computer. Current options available are:

ByteOrder to inform **DspExport** of the byte order of the host computer. This is needed because all the data read or written to/from the RVP8 is in 16-bit words. If the host computer has a different byte order from the RVP8, **DspExport** will byte swap the data.

WillCompress to inform **DspExport** to use compression or not. Compression is only used if both sides agree to use it. The host computer should only set this to 1 if it received a "CanCompress" of 1 on initial connection. The only thing compressed is the data from normal READ commands. If it is compressed, it will reply with the acknowledge compressed string of "AkC". The compression program is the zlib

compress and uncompress. The uncompress function requires that the caller know the expected uncompressed size. This is true for RVP8 reads, because the reader always specifies the read size.

Version, send the IRIS version.

Read data available command (RDAV)

Example: "RDAV|100|2|" This means read up to 100 bytes of data from the RVP8 in individual DMA transfers of 2 bytes each. Before each read, the status is checked to see if there is more data available. If not, the read stops, and the number of bytes read is returned. This is merely a performance enhancing command since the same feature is available by using the READ command and the STAT command.

Open the connection for I/O (OPEN)

Example: "OPEN" This means switch from open for "info only" mode to open for I/O. If the signal processor is in use by another device, you will get an error in response to this command. Multiple clients are allowed to connect for info only, but only one can do I/O. Note that if you run **DspExport** with the –803 command line option, you will get the legacy behavior which means that every connection will automatically send the OPEN command. There is no reverse command to switch back to open for info only. There is also no such library call in the driver.

Read Z cal information (RCAL)

Example: "ZCAL" This means read the dsp_refl_cal structure from the RVP8 machine and send it back in an ASCII name=value pair format. This is the structure configured by **zauto** and by **zcal**. That configuration is served out to all clients who wish to use the RVP8.

Reset Kernel FIFOs (RKFF)

Example: "RKFF|2|" This means reset the kernel FIFOs on the RVP8. The argument specifies which direction FIFOs to reset.

Read Setup information (SETU)

Example: "SETU" This means read the dsp_manual_setup structure from the RVP8 machine and send it back in an ASCII name=value pair format. This is the structure configured in the RVP section of setup. That configuration is served out to all clients who wish to use the RVP8.

Write Z cal Information (WCAL)

Example: "WCAL|...". This command writes the dsp_refl_cal structure to the RVP8 to be saved there.

Notes on migrating from the SCSI interface

Here are suggestions for customers who are converting an existing program which used a SCSI interface to the RVP7 to the socket interface to the RVP8. First take a look at our source code which handles either SCSI or socket. In OpenSocket.c you can see the code which replaces the SCSI device open call. The SCSI inquiry command is replaced by reading the string returned after the socket is opened. The SCSI read command is replaced by the "READ|.." command. The SCSI mode sense command is replaced by the "STAT|" command. The SCSI write command is replaced by the "WRIT|..." command. You should get your code working first without using the RDAV command or using compression.

There is a significant difference between the RVP7 and RVP8 in regards to the FIFO reset command. This is the RVP8 command 0x008C (see 7.11 Reset (RESET) on page 326). The RVP8 is unable to read incoming commands if the output FIFO is entirely full. Therefore, if you put the RVP8 into continuous output mode, then issue the FIFO reset command to return to interactive mode, it may hang. We have added special dsp library support to solve this. To see how we have handled this problem, look in the source file DspResetFifo.c.

3.4 Digital AFC Module (DAFC)

The DAFC is a small self-contained circuit board which can passively "eavesdrop" on the RVP8's serial uplink transmissions. Its purpose is to generate a set of digital AFC control lines that could be applied, for example, to a custom STALO frequency synthesizer. A full size (3"x3.75") assembly diagram of the board is shown in Figure 17 on page 100. It can be installed in the radar system either as a bare board, or packaged into a small metal enclosure.

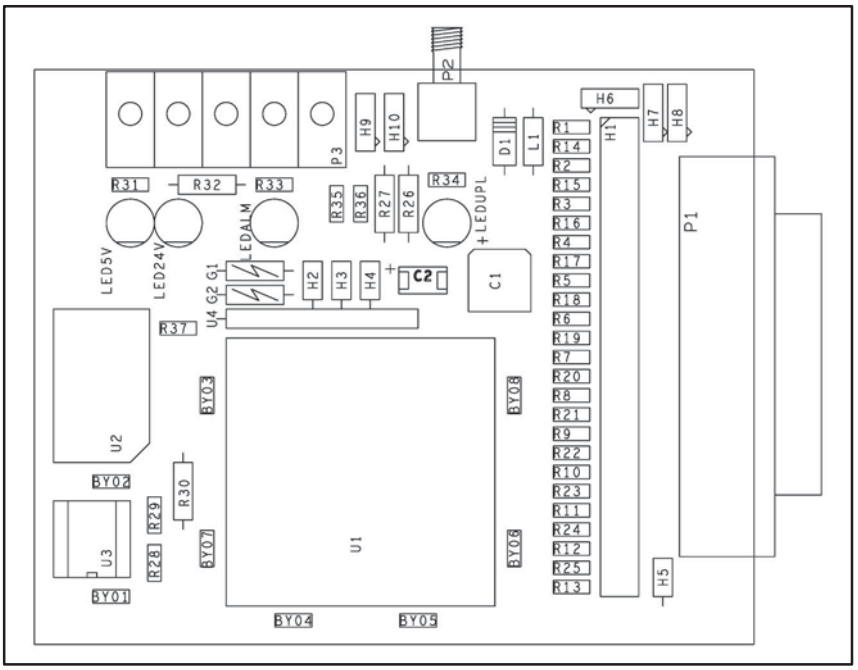


Figure 17 Assembly Diagram of the DAFC

Vaisala recommends that the DAFC board be used in new system designs whenever AFC is required, as it offers these advantages over other methods of frequency control:

1. The use of a digital frequency synthesizer is superior to using analog AFC because the stability of a synthesized STALO can be made much greater than that of a tunable cavity oscillator. Also, noise on the AFC control voltage directly contributes to phase noise in the received weather targets in analog AFC systems, so cabling of the control signal can become tricky.
2. The RVP8 Connector Panel can also be made to output 8–bit AFC (TTL or RS422). However, this is not in general recommended because of the potential for noise on the cable which is typically run >2m into a radar cabinet. Using the DAFC module is preferable because the board can be physically located very close to the STALO. The length of the control cable and its susceptibility to noise and ground loops are therefore reduced. Also, the DAFC board can supply up to 24 output control lines, rather than just eight.

The digital output lines are made available as TTL levels on a 25-pin female "D" connector (P1). There are 130Ω resistors (R1–R25) in series with each output line to help protect the board against momentary application of non-TTL voltages on its external pins. However, these resistors do impose a restriction on the input line configuration of the receiving device. To assure a valid TTL low level of 0.6V max. requires

that the STALO inputs be pulled up to +5 with nothing less than (approx.) 1.2KΩ. Put another way, the low level input current of the receiving device should not exceed 4.5mA. Most STALOs that we have seen use 5-20KΩ pull-up resistors, so this should not be a problem.

All twenty five pins of the "D" connector are wired identically on the DAFC board, that is, each pin connects to one end of a 2-pin jumper (2x25 header H1), the other end of which connects to a Programmable Logic Device (PLD) chip. The PLD lines can be configured either as inputs or outputs, and this single chip handles all of the decoding and driving needs for the entire board. For each "D" connector pin that is to be used as an AFC output or Fault Status input, you should install the corresponding jumper to connect that pin through to the PLD, or use a wirewrap wire if the pin must go to a different PLD line. The "D" connector pin numbers are printed next to each of the jumper locations. Because of the ordering of the pins in the connector housing, jumpers 1 through 13 are interleaved with jumpers 14 through 25.

The uplink protocol that the board should be expecting is selected by jumpers H3 and H4, as summarized in Table 13 on page 101. The first three table entries describe three fixed mappings of the traditional AFC-16 uplink format onto various pins of the 25-pin "D" connector. One of these choices must be used whenever the DAFC is interfaced to an RVP8 system whose uplink uses the older style 16-bit AFC uplink format. In this case you will have to make most or all of the pin assignments using wirewrap wire to connect each bit to its corresponding pin. This will be somewhat tedious, but hopefully one of the three formats will be a reasonable starting point for doing the wiring. By far the most preferable solution, however, is to use the Pinmap uplink protocol (available since Rev.19) which allows for complete software mapping of all 25 external pins.

Table 13 DAFC Protocol Jumper Selections

H4	H3	Function
On	On	AFC-16 format, Bits<0:15> on Pins<1:16>, Fault input on Pin 25
On	Off	AFC-16 format, Bits<0:15> on Pins<25:10>, Fault input on Pin 3
Off	On	AFC-16 format, Bits<0:15> on Pins<18, 19, 6, 7, 21, 22, 23, 11, 10, 9, 20, 8, 12, 25, 13, 24>, Fault input on Pin 4
Off	Off	Pinmap format, software assignment of all pins

Ground, +5V, and +24V power supply pins on the "D" connector should be connected with wirewrap wire to the nearby power and ground posts H6, H7, and H8. The PLD jumpers for these power supply pins must not be installed. Two 3K/6K resistive terminators are also available at H5 for

pulling pins up to approximately +3.3V when that is appropriate. Unused "D" connector pins should remain both unwired and not jumpered.

WARNING It is important that the jumpers only be installed for pins that carry TTL inputs or outputs destined for the on-board PLD. The jumpers must be removed for all power supply pins, and for unused and reserved pins of the external device.

The DAFC board runs off of a single +5V power supply which can be applied either from the STALO through the "D" connector, or externally through the terminal block. There are also provisions for supplying +24V (approx.) between the terminal block and the "D" connector, which is handy for cabling power to a STALO that requires the second voltage. Two green LEDs indicate the presence of +5V and +24V. Terminal block Pin #1 is +5V, Pin #2 is +24V, and Pin#3 is Ground. Pin #1 is the one nearest the corner of the board.

There is an option for having a "Fault Status" input on the "D" connector of the DAFC. Since the board is completely passive in its connection to the uplink, the fault status bit will not affect the uplink in any way. Rather, the bit is simply received by the board (with optional polarity reversal) and driven onto the terminal block (P3) from whence it can be wired to some other device, for example, a BITE input line of an RCP02. A yellow LED is included to indicate the presence of any external fault conditions.

The "AB" position of the 3-pin "Alarm" jumper (H9) connects the Fault Status signal to Pin #4 of the terminal block, whereas the "BC" position grounds that terminal block pin. A second ground can be made available at Pin #5 of the terminal block by installing a jumper in the "BC" position of the "Spare" 3-pin jumper (H10). This second ground could be used as a ground return when the Fault Status line is driven off of the terminal block. The "AB" position of the "Spare" jumper is reserved for some future input or output line on the terminal block.

Both the shield and the center conductor of the uplink SMA input connector (P2) are electrically isolated (> 100KΩ) from the rest of the DAFC board. Moreover, the SMA connector pins themselves are high-impedance and unterminated. What this means is that the board can be teed into the uplink cable anywhere in the cable run from the RVP8/Rx board to the IFD. Since the cable is driven by the RVP8/Rx, it must be at one end of the cable; and since termination is provided by the IFD, it must be at the other end. The DAFC can be anywhere in the middle. Be sure, however, that the TEE is located right at the DAFC itself so that an unterminated cable stub is not created. A red LED is included to indicate that a valid uplink data stream is being received.

A crystal oscillator is used to supply the operating clock for the on-board logic, and there are two choices of frequency to use. If jumper H2 is "Off" then the crystal frequency should be equal to the IFD's sampling clock f_{aq}, and if H2 is "On" the frequency should be $(0.75 \times f_{aq})$.

Additional information about using AFC can be found in Sections 3.2.11 IFD Analog AFC Output Voltage (Optional) on page 81, 4.2.7 Mb — Burst Pulse and AFC on page 142, and 6.1.3 Automatic Frequency Control (AFC) on page 210.

3.4.1 Example Hookup to a CTI MVSR-xxx STALO

Here is a complete example of what would need to be done in hardware and software to interface the DAFC to a Communication Techniques Inc. digital STALO. The electrical interface for the STALO is via a 26-pin ribbon cable which carries both Control and Status, as well as DC power. This cable can be crimped onto a mass-terminated 25-pin "D" connector (with one wire removed) and plugged directly into the DAFC. The resulting pinout is shown in Table 14 on page 103.

The STALO frequency is controlled by a 14-bit binary integer whose LSB has a weight of 100 KiloHertz. In addition, the "Inhb" pin must be low for the STALO to function. Power is supplied on the +5V and +24V pins, and two grounds are provided. An "alarm" output is also available.

Table 14 Pinout for the CTI MVSR-xxx STALO

Ribbon Pin	"D" Pin	Function	Ribbon Pin	"D" Pin	Function
1	1	Ground	2	14	--
3	2	+5V	4	15	--
5	3	+24V	6	16	--
7	4	Alarm	8	17	--
9	5	--	10	18	Bit0
11	6	Bit2	12	19	Bit1
13	7	Bit3	14	20	Bit10
15	8	Bit11	16	21	Bit4
17	9	Bit9	18	22	Bit5
19	10	Bit8	20	23	Bit6
21	11	Bit7	22	24	Ground
23	12	Bit12	24	25	Bit13
25	13		26		--

First configure the IFD pins themselves. Pins 1 and 24 are power supply grounds, and are connected with wirewrap wire to the nearby ground posts. Pins 2 and 3 supply +5V and +24V to the STALO, and should be wire wrapped to the internal power posts. The STALO power, as well as the DAFC power, is then supplied externally via the terminal block on the DAFC itself.

Sixteen jumpers should be installed to connect the Control and Status lines, that is, pins 4, 6–13, 18–23, and 25. We will use pinmap uplink protocol, so H3 and H4 are removed; and a x1 on-board crystal, so H2 is also removed.

The STALO has an output frequency range from 5200–6020MHz in 100KHz steps. In this example we will assume that we need an AFC frequency span of 5580–5600MHz. This can be done with the following setups from the **Mb** section:

```
AFC span– [-100%,+100%] maps into [ 3800 , 4000 ]

AFC format– 0:Bin, 1:BCD, 2:8B4D: 0, ActLow: NO

AFC uplink protocol– 0:Off, 1:Normal, 2:PinMap : 2
```

```
PinMap Table (Type 31 for GND, 30 for +5)
Pin01:GND      Pin02:GND      Pin03:GND      Pin04:GND      Pin05:GND
Pin06:02       Pin07:03       Pin08:11       Pin09:09       Pin10:08
Pin11:07       Pin12:12       Pin13:GND      Pin14:GND      Pin15:GND
Pin16:GND      Pin17:GND      Pin18:00       Pin19:01       Pin20:10
Pin21:04       Pin22:05       Pin23:06       Pin24:GND      Pin25:13
FAULT status pin (0:None): 4, ActLow: NO
```

We map the AFC interval into the numeric span 3800–4000, and choose the "Bin" (simple binary) encoding format. The actual frequency limits therefore match the desired values:

$$5200MHz + (3800 \times 100KHz) = 5580MHz$$

$$5200MHz + (4000 \times 100KHz) = 5600MHz$$

The "Inhb" line is held low, and fault status is input on Pin 4. Note that all pins that are not directly controlled by the software uplink (for example, power pins, and unused pins) are merely set to "GND" in the setup table.

3.4.2 Example Hookup to a MITEQ MFS-xxx STALO

The electrical interface for this STALO uses a 25-pin "D" connector with the following pin assignments

- GROUND on pins 1 and 2.
- Four BCD digits of 1KHz, 10KHz, 100KHz, and 1MHz frequency steps, using Pins <25:22>, <21:18>, <17:14>, <13:10>.
- Seven binary bits of representing 10MHz steps, Bits<0:6> on Pins<9:3>.

First configure the IFD pins themselves. Pins 1 and 2 are ground, and are connected with wirewrap wire to the nearby ground posts. Pins 3 through 25 all are signal pins, so we plug in a jumper for each of these 23 pins. We will use pinmap uplink protocol, so H3 and H4 are removed; and a x1 on-board crystal, so H2 is also removed.

In this example we will assume that we wish to control the STALO in 20KHz steps from 1.350GHz to 1.365GHz. This can be done with the following setups from the **Mb** section:

```
AFC span– [-100%,+100%] maps into [ 1350000 , 1365000 ]

AFC format– 0:Bin, 1:BCD, 2:8B4D: 2, ActLow: NO

AFC uplink protocol– 0:Off, 1:Normal, 2:PinMap : 2

PinMap Table (Type 31 for GND, 30 for +5)
Pin01:GND      Pin02:GND      Pin03:22       Pin04:21       Pin05:20
Pin06:19       Pin07:18       Pin08:17       Pin09:16       Pin10:15
Pin11:14       Pin12:13       Pin13:12       Pin14:11       Pin15:10
Pin16:09       Pin17:08       Pin18:07       Pin19:06       Pin20:05
Pin21:GND      Pin22:GND      Pin23:GND      Pin24:GND      Pin25:GND
FAULT status pin (0:None): 0, ActLow: NO
```

We map the AFC interval into a numeric span from 1350000 to 1365000, and choose the "8B4D" mixed-radix encoding format. The STALO itself has 1KHz frequency steps, but the AFC servo will be easier to tune if we intentionally degrade this to 20KHz. This is done simply by grounding all four of the 1KHz BCD input lines, plus the LSB of the 10KHz BCD digit. A more creative use for one of these unused pins would be to remove the pin 25 jumper, wirewrap pin 25 to ground (so the STALO sill reads it a

logic low), and assign pin 25 as a fault status input. That pin could then be connected to an external fault line, if the STALO has one.

3.5 RVP8 Custom Interfaces

This section describes some additional points of interface to the RVP8. These hookups are less conventional than the "standard" interfaces described earlier in this chapter, but they sometimes can supply exactly what is needed in exactly the right place. For the most part, these custom interfaces are merely taps into existing internal signals that would normally not be seen by the user.

3.5.1 Using the Legacy IFD Coax Uplink

In the previous RVP7 processor the Coax Uplink was the IFD's single line of communication from the main processor board. All of the information that was needed by the IFD would arrive through this uplink; and as such, it contained information that might also be useful for other parts of the radar system. In particular, it is a convenient source of digital AFC.

The RVP8 uses a single CAT-5E Uplink/Downlink cable between the IFD and RVP8/Rx PCI card. The legacy coax uplink protocol is no longer used directly; but to help with backward compatibility, the waveform is now synthesized as an *output* from the IFD. Any hardware that used to be attached to the RVP7 coax uplink can still be driven from this new IFD port.

The uplink is a single digital transmission line that carries a hybrid serial protocol. The two logic states, "zero" and "one" are represented by 0-Volt and +12-Volt (open circuit) electrical levels. The output impedance of the uplink driver is approximately 55Ω . When the cable is terminated in 75Ω the overall positive voltage swing will be approximately 8.6-Volts.

The electrical characteristics of the uplink have been optimized for balanced "groundless" reception. The recommended eavesdropping circuit is shown in Figure 18 on page 107, and consists of a high speed comparator (Maxim MAX913, or equivalent) and input conditioning resistors. Both the shield and the center conductor of the coax uplink feed the comparator through 33KΩ isolation resistors; no direct ground attachment is made to the shield itself. The 500Ω resistors provide the local ground reference, and the 47KΩ resistor supplies a bias to shift the unipolar uplink signal into a bipolar range for the comparator.

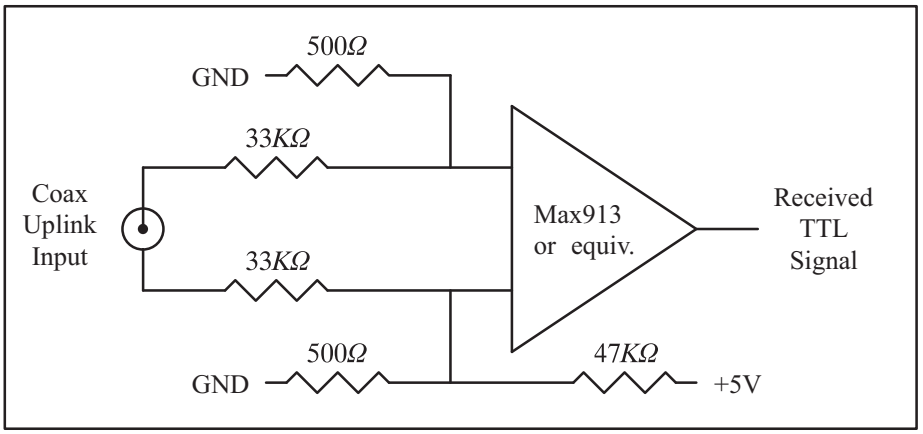


Figure 18 Recommended Receiving Circuit for the Coax Uplink

The uplink signal, shown in Figure 19 on page 107, is periodic at the radar pulse repetition frequency, and conveys two distinct types of information to the IFD. The signal is normally low most of the time (to minimize driver and termination power), but begins a transition sequence at the beginning of each transmitted pulse.

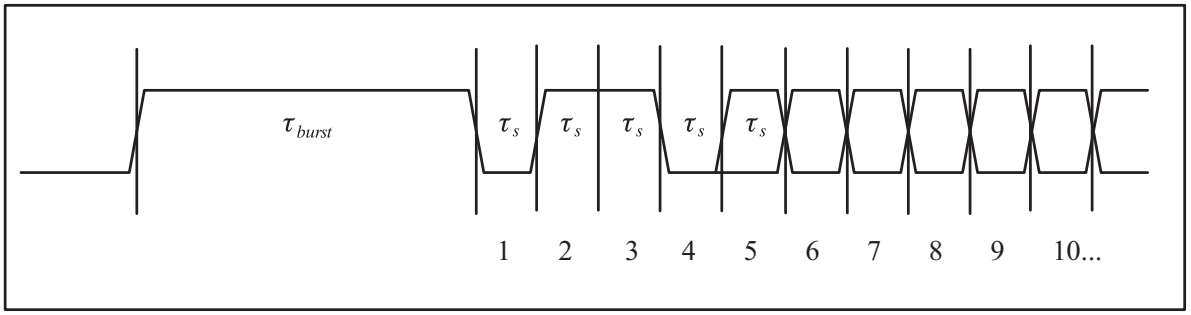


Figure 19 Timing Diagram of the IFD Coax Uplink

The first part of each pulse sequence is a variable length "burst window" which is centered on the transmitted pulse itself, and which has a duration T_{burst} approximately 800ns greater than the length of the current FIR matched filter. The burst window defines the interval of time during which the IFD transmits digitized burst pulse samples, rather than digitized IF samples, on its downlink. The exact placement and width of the burst window will depend on the trigger timing and digital filter specifications that the user has chosen, usually via the **Pb** and **Ps** plotting setup commands.

Following the burst window is a fixed-length sequence of 25 serial data bits which convey information from the RVP8/Rx board. The first four data bits form a characteristic (0,1,1,0) marker pattern. The first zero in this

pattern effectively marks the end of the variable length burst window, and the other three bits should be checked for added confidence that a valid bit sequence is being received. Table 15 on page 108 defines the interpretation of the serial data bits.

Table 15 Bit Assignments for the IFD Coax Uplink

Bit(s)	Meaning
14	Marker Sequence (0,1,1,0). This fixed 4-bit sequence identifies the start of a valid data sequence following the variable-length burst window.
520	16-bit multi-purpose data word, MSB is transmitted first (See below)
21	Reset Request. This bit will be set in just one transmitted sequence whenever an RVP8 reset occurs.
22	If set, then interpret the 16-bit data word as 4-bits of command and 12-bits of data, rather than as a single 16-bit quantity (See below)
2324	Diagnostic select bits. These are used by the RVP8 power-up diagnostic routines; they will both be zero during normal operation.
25	Green LED Request; 0=Off, 1=On. The state of this bit normally follows the "Downlink Detect" LED on the RVP8/Rx board.

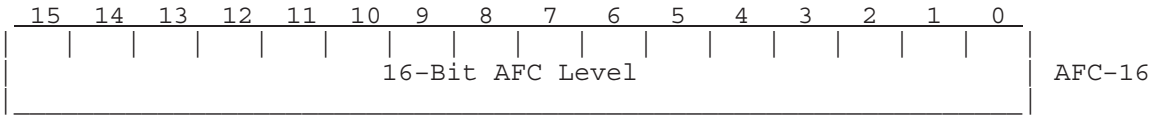
The period τ_s of the serial data is $(128/f_{aq})$, where f_{aq} is the acquisition clock frequency given in the Mc section of the RVP8 setup menu. For the default clock frequency of 71.9502MHz, the period of the serial data will be 1.779 sec. The logic that is receiving the serial data should first locate the center of the first data bit at $(0.5 \times \tau_s)$ past the falling edge at the end of the burst window. Subsequent data bits are then sampled at uniform τ_s intervals.

The actual data sampling rate can be in error by as much as one part in 75 while still maintaining accurate reception. This is because the data sequence is only 25-bits long, and hence, the last data bit would still be sampled within ±1/3 bit time of its center. Having this flexibility makes it easier to design the receiving logic. For example, if a 5MHz or 10MHz clock were available, then sampling at 1.8µsec intervals (1:85 error) would be fine. Likewise, one could sample at 1.75µsec based on a 4MHz or 8MHz clock (1:61 error), but only if the first sample were moved slightly ahead of center so that the sampling errors were equalized over the 25-bit span.

Interpreting the Serial 16-bit Data Word

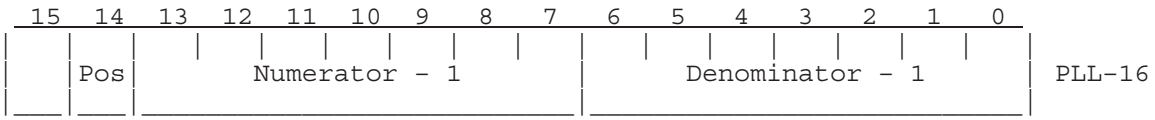
The serial 16-bit data word has several different interpretations according to how the RVP8 has been configured, and whether Bit #22 of the uplink stream is set or clear. The evolution of these different formats has been in response to new features being added to the IFD (3.2 IFD IF Digitizer Module Installation on page 66), and the production of the DAFC Digital AFC Module (3.4 Digital AFC Module (DAFC) on page 99).

The original use of the uplink data word was simply to convey a 16-bit AFC level, generally for use with a magnetron system. Bit #22 is clear in this case, and the word is interpreted as a linear signed binary value. The use of this format is discouraged for new hardware designs, but it will remain available to guarantee compatibility with older equipment.



Level	0111111111111111	(most positive AFC voltage)
	0000000000000000	(center AFC voltage)
	1000000000000000	(most negative AFC voltage)

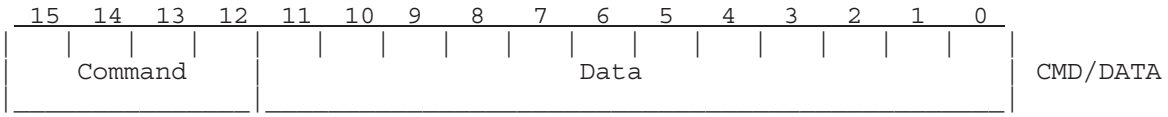
When the IFD is jumpered for phase locking to an external reference clock, then Bit #22 will be clear and the data word conveys the PLL clock ratio, and the Positive/Negative deviation sign of the Voltage Controlled Crystal Oscillator (VCXO). This format is commonly used with klystron systems, especially when the RVP8 is locking to an external trigger.



Note that the AFC-16 and PLL-16 formats can never be interleaved for use at the same time, since there would be no way to distinguish them at the receiving end.

Finally, an expanded format has been defined to handle all future requirements of the serial uplink. Bit #22 is set in this case, and the data word is interpreted as a 4-bit command and 12-bit data value. A total of *16x12=192* auxiliary data bits thus become available via sequential transmission of one or more of these words. The CMD/DATA words can

also be used along with *one* of the AFC-16 or PLL-16 formats, since Bit #22 marks them differently.



Commands #1, #2, and #3 control the 25 output pin levels of the DAFC board. These transmissions may be interspersed with the PLL-16 format in systems that require both clock locking and AFC, for example, a dual-receiver magnetron system using a digitally synthesized COHO. Note that the entire 25-bits of pin information are transferred synchronously to the output pins only when CMD=3 is received. This assures that momentary invalid patterns will not be produced upon arrival of CMD=1 or CMD=2 when the output bits are changing.

CMD=1	Data<0>	DAFC output pin 25
	Data<6>	Fault Input is active high
	Data<11:7>	Which pin to use for Fault Input (0:None)
CMD=2	Data<11:0>	DAFC output pins 24 through 13
CMD=3	Data<11:0>	DAFC output pins 12 through 1

These three digital AFC pinmap commands are recommended as a replacement for the original AFC-16 format in all new hardware designs. If you only need 12-bits of linear AFC, then map the AFC range into the -2048 to +2047 numeric span, and select binary coding format (See 4.2.7 Mb — Burst Pulse and AFC on page 142); the 12-bit data with CMD=3 will then hold the required values. To get a full 16-bit value, use a -32768 to +32767 span and extract the full word from both CMD=2 and CMD=3. Of course, other combinations of bit formats and number of bits (up to 25) are also possible.

Command #4 is used to control some of the internal features of the IFD. Bits <4:0> configure the on-board noise generator so that it adds a selectable amount of dither power to the A/D converters. This noise is bandlimited using a 10-pole lowpass filter so that most of the energy is within the 150KHz to 900KHz band, with negligible residual power above 1.4MHz. Each of the five bits switch in additional noise power when they are set, with the upper bits making successively greater contributions. Bits

<6:5> permit the IF-Input and Burst-Input signals to be reassigned on the downlink.

CMD=4	Data<4:0>	Built-in noise generator level	
		IF-Input and Burst-Input selection	
	Data<6:5>	00 : Normal	01 : Swap IF/Burst
		10 : Burst Always	11 : IF Always
	Data<7>	0 : Normal	1: Swap Pri/Sec IF
	Data<8>	Downlink IF data stream format	
		0 : Normal 72MHz single channel	
		1 : Half-band 36MHz dual channel	
	Data<9>	0 : Low Half-band	1: High Half-band

CHAPTER 4
TTY NONVOLATILE SETUPS

The RVP8 provides an interactive setup menu that can be accessed either from a serial TTY, or from the host computer interface. Most of the RVP8's operating parameters can be viewed and modified with this menu, and the settings can be saved in non–volatile RAM so that they take effect immediately on start–up. This permits custom trigger patterns, pulsewidth control, matched FIR filter specs, PRF, etc., to be configured by the user in the field.

The TTY menu also gives access to a collection of graphical setup and monitoring procedures that use an ordinary oscilloscope as a synthesized visual display. The burst pulse and receiver waveforms can be examined in detail (both in the time and frequency domain) and the digital FIR filter can be designed interactively to match the characteristics of the transmitted pulse.

4.1 Overview of Setup Procedures

This section describes basic operations within the setup menus such as making TTY connections, entering and exiting the menus, and saving and restoring the configurations.

The setup TTY menu can be accessed by executing the following command:

$dspx

You will then be prompted by the following:

```
$dspx

Digital Signal Processor 'Chat'

Checking for code upgrades...Okay

(Type ^C to exit Chat Mode)
```

The interactive setup menu is invoked by pressing the **Escape key** on the TTY. If that key can not be found on the keyboard, you can sometimes use Control "[" to generate the ESC code. The RVP8 then responds with the following banner and command prompt.

```
SIGMET - Part of Vaisala Group

RVP8 Digital IF Signal Processor V12.9 IRIS-8.12.9

RVP8>
```

The banner identifies the RVP8 product, and displays the RVP8 software version (for example, V3.9) and IRIS software version (for example, IRIS 8.03.6). This information is important whenever RVP8 support is required, and it is also repeated in the printout of the **V** command (See below).

The **Q** command is used to exit from the menus and to reload the RVP8 with the (possibly changed) set of current values. It is important to quit from the menus before attempting to resume normal RVP8 operation. Portions of the RVP8 command interpreter remain running while the menus are active (so that the TTYOP command works properly), but the processor as a whole will not function until the menus are exited.

From the command prompt, typing "**help**" or "?" gives the following list of available commands.

```
Command List:
```

```
F: Use Factory Defaults

S: Save Current Settings

R: Restore Saved Settings

M: Modify/View Current Settings Mb – Burst Pulse and AFC

Mc – Overall Configuration

Mf – Clutter Filters

Mp – Processing Options

Mt<n> – Trigger/Timing <for PW n>

Mz – Transmissions and Modulations

M+ – Debug Options

P: Plot with Oscilloscope Pb – Burst Pulse Timing

Ps – Burst Spectra and AFC

Pr – Receiver Waveforms

P+ – Visual Test Pattern

V: View Card and System Status

?: Print all Menu Commands (this list)

?? – Print all Current Setup Settings

*: Sample Receiver Noise Levels

@: Display/Change the Current Major Mode

~: Swap Burst/IF Inputs on IFD

Q: Quit
```

4.1.1 Factory, Saved, and Current Settings

The *current settings* are the collection of setup values with which the RVP8 is presently operating; the *saved settings* are the collection of values stored in non–volatile RAM. The saved settings are restored (made current) each time the RVP8 starts. The **S** command saves the current settings into the non–volatile RAM, and the **R** command restores those non–volatile values so that they become the current settings. The **F** command initializes the current settings with factory default values. Thus, **F** followed by **S** saves factory defaults in non–volatile RAM, so that the RVP8 powers up in its original configuration as shipped.

The RVP8 retains all of its saved settings when new software releases are installed; the new version of code will automatically use all of the previous saved values. However, if the RVP8 detects that the new release requires a setup parameter that did not exist in the previous release, then a factory default value will automatically be filled in for that parameter. A warning is printed whenever this occurs (See also, 4.1.2 V — View Card and System Status on page 116).

There is also support for intermediate minor releases of RVP8 code. Each software release has a major version number (the one that it always had), plus a minor version number for intermediate "unofficial" releases. The minor number starts from zero at the time of each "official" release, and then increments until the next "official" release. The RVP8 includes the minor release number (if it is not zero) in the printout of the **V** command. Likewise, the minor release number of the code that was last saved in the nonvolatile RAM is also shown. This is an improvement over having to check the date of the code to determine which minor release was running.

Note that the RVP8 does not actually begin using the current settings until after the **Q** command is entered, so that the processor exits the TTY setup mode and returns to normal operation.

4.1.2 V — View Card and System Status

The **V** command displays internal diagnostics. This information is for inspection only, and can not be changed from the TTY. The view listing begins with the banner:

```
Configuration and Internal Status
```

and then prints the following lines:

```
RVP8 Digital IF Signal Processor V3.10(Pol) IRIS-8.04
```

This line shows the revision level of the RVP8 software, the IRIS version.

```
Settings were last saved using V3.10
```

This line tells which version of RVP8 code was the last to write into the non–volatile RAM. It is printed only if that last version was different from the version that is currently running. The information is included so that a "smart upgrade" can often be done, that is, values that did not exist in the prior release can be filled in with a guess that is better than merely taking the factory default.

```
RVP8 started at: 13:07:33 3 NOV 2003

Current time is: 13:14:03 3 NOV 2003
```

These lines provide information about when the RVP8 was started, the current system time, and implicitly, the uptime.

```
CPU–Type: Pentium(R) III
```

This line displays processor information.

```
IPP–Library: ippsa6 v2.0 gold SP1 2.0.6.39
```

This line displays information about the Intel libraries used for RVP8 processing.

```
Physical hardware inventory:
Found PCI Card RVP8/Rx  Rev.A    Serial:1628   Code:14
(/dev/rda/rvp8rx1)
Found PCI Card RVP8/Tx  Rev.B    Serial:1887   Code:9
(/dev/rda/rvp8tx0)
Found PCI Card I/O62  Rev.B      Serial:1841   Code:19
(/dev/rda/io620)
\> IO62CP Backpanel  Rev.B       Serial:1822   Code:3
```

The physical hardware inventory provides information about the system hardware being used by the RVP8. This list ONLY displays hardware that is being used, not all hardware in the system (that is, an RCP8 could be present in the same chasis, but the RCP8 hardware would not be included in this list).

```
Diagnostics: PASS
```

If errors were detected by the startup diagnostics then an error bitmask will be shown on the first line. The word "PASS" indicates that no errors were detected.

```
Processes and Threads:

RVP8Proc-0 – PID:28503 Priority:10 Policy:RealTimeRR
```

The "Process and Threads" list displays RVP8 processes and their related priority. All RVP8 processes/threads should be running under RealTimeRR policy to guarantee adequate attention from the processors.

```
Shared library build dates:
```

This section provides RVP8 developers with information about code resources.

```
Front panel display:
```

0.00		AZ/EL	0.00
FFT	100B	1000Hz	x1

The front panel display mirrors the display on the front of the RVP8 chasis. This is helpful if you are at a remote location using DspExport.

```
Tx/Clk:Okay TrigRAM is 99.0% free, TrigCount:378921
```

The Tx/Clk field displays information about the RVP8/Tx clock (if applicable). TrigRAM provides resource information for those who are implementing custom waveforms.

```
IFD:Okay Link: Delay = 0.541 usec, Jitter = 0.014 usec
```

This first section of this line summarizes the receiver status and Burst input signal parameters. The status may show:

Okay	RVP8/IFD and connecting cables are all working properly
DnErr	Problem in DownLink connection from RVP8/IFD > RVP8
UpErr	Problem in UpLink connection from RVP8 > RVP8/IFD
NoPLL	RVP8/IFD PLL is not locked to external user-supplied clock reference
DiagSW	RVP8/IFD test switches are not in their normal operating position

The section second describes the IFD link status. During startup the RVP8 measures the round trip delay along 1) the uplink to the receiver module, 2) the pipeline delays within the receiver module, 3) the downlink, and 4) pipeline delays in the data decoding hardware. The time shown is accurate to within 14ns, and is used internally to insure that the absolute calibration of trigger and burst pulse timing remains unaffected by the distance

between the main card and the receiver module. You may freely splice any lengths of cable without affecting the calibrations; the delay time will change, but the trigger and burst calibrations will remain constant.

The standard deviation of the measured delay is also shown. If the link to the IFD is working properly this variation should be less than half the period of its acquisition clock. Larger errors may indicate a problem in the cabling. A diagnostic error bit is set if the error is greater than two acquisition clock periods.

```
AFC:0.00% (Disabled), Burst Pwr:48.6 dBm, Freq:30.000 MHz
```

AFC indicates the level and status of the AFC voltage at the RVP8/IFD module. The number is the present output level in D–Units ranging from -100 to +100. The shorter "%" symbol is used since percentage units correspond in a natural way to the D–Units.

Burst Pwr indicates the mean power within the full window of burst samples. DC offsets in the A/D converter do not affect the computation of the power, that is, the value shown truly represents the waveform's (Signal+Noise) energy. Freq indicates the mean frequency of the burst, derived from a 4th order correlation model.

For more information refer to Chapter 5, Plot-Assisted Setups, on page 159.

4.2 View/Modify Dialogs

The **M** menu may be used to view, and optionally to modify, all of the current settings. The current value of each parameter is printed on the screen, and the TTY pauses for input at the end of the line. Pressing Return advances to the next parameter, leaving the present one unchanged. You may also type **U** to move back up in the list, and **Q** to exit from the list at any time.

Typing a numeric or YES/NO response (as appropriate to the parameter) changes the parameter's value, and displays the line again with the new value. All numbers are entered in base ten, and may include a decimal point and minus sign. In some cases, several parameters are displayed on one line, in which case, as many parameters are changed as there are new values entered. In all cases, the numbers are checked to be within reasonable bounds, and an error message (listing those bounds) is printed if the limits are exceeded. Note that changes to the settings (generally) do not take effect until after the **Q** command is typed, at which point the RVP8 exits the local TTY menu and resumes its normal processing operations.

The **M** menu provides access to a large number of configuration settings. As a result, all of the **M** menu commands begin with the letter "M" and are followed by a lower case letter which represents a subcategory, that is, **Mb** (Burst Pulse and AFC), **Mc** (Overall Configuration), **Mf** (Clutter Filters), **Mp** (Processing Options), **Mt** (Triggers and Timing), **Mz** (Transmissions and Modulations), **M+** (Debug Options). The **??** command by itself prints the entire set of questions so that you can make a hard copy.

The **M** menu always works from the current parameter values, not from the saved values in non–volatile RAM. If the host computer has modified some of the current values, then you will see these changes as you skip through the setup list. However, typing **S** at that point would save all of the current settings and would, perhaps, make many changes to the original non–volatile settings. In general, to make an incremental change to the saved settings, first type **R** to restore all of the saved values, then use the **M** menu to make the changes starting from that point, and **S** to save the new values.

A listing of the parameters that can be viewed and modified with the **M** menu is detailed in the following subsections. In each case, the line of text is shown exactly as it appears on the TTY with the factory default settings. A definition of each parameter is given and, if applicable, the lower and upper numeric bounds are shown.

4.2.1 Mc — Top Level Configuration

This set of commands configure general properties of the RVP8/IFD and RVP8 cards.

```
Acquisition clock: 35.9751 MHz
```

This is the frequency of the acqusition sampling clock in the IFD module. This will generally be in the 72MHz range for the RVP8 CAT–5E IFD, and in the 36MHz range for the legacy RVP7 IFD. If you are locking the IFD to an external reference, the center frequency of the installed VCXO is entered (see 3.2.12 IFD Reference Clock Input (Optional) on page 82).

Limits: 33.33 to 80 MHz

```
Live Angle Input – 0:None, 1:Sim, 2:TAGS, 3:S/D : 2
```

This setting is used to configure the input of live angles. In most situations, the angles will be coming in from the RCP via TAGS. The S/D option provides direct conversion of 3–wire synchro waveforms for the AZ and EL position angles. You may directly hookup AZ/EL synchros to the 12–pin input connector on teh IO-62 standard backpanel when you choose S/D.

```
Primary RVP8/Rx PCI Card (-1:None) : 0
```

This setting configures which RVP8/Rx card will be used by the RVP8.

```
Primary RVP8/Tx PCI Card (-1:None) : -1
```

This setting configures which RVP8/Tx card will be used by the RVP8.

```
Primary I/0-62 PCI Card (-1:None : 0

Run I/0-62 external line powerup tests:NO
```

This setting configures which RVP8/Rx card will be used by the RVP8. The I/O-62 external line powerup tests are used for debugging the backpanel and should be turned off during normal operation. When enabled, the backpanel receives a spread of signals which could cause problems in an operational environment (that is, firing of the transmitter).

```
Provide IRIS RPC network status server: NO
```

The default value is NO in order to reduce network security concerns. When enabled, it opens up a network port.

```
PWINFO command enabled: No
```

The "Pulsewidth Information" user interface command can be disabled, thus further protecting the radar against inappropriate combinations of pulsewidth and PRF. This is a more safe setting in general, and is even more important when DPRT triggers are being generated. It can also be useful when running user code that is not yet fully debugged.

```
TRIGWF command enabled: NO
```

The "Trigger Waveform" user interface command can be disabled if you want to prevent the host computer from overwriting the RVP8's stored trigger specifications. This is the default setting, based on the assumption that the built–in plotting commands would be used to configure the triggers. Answering "YES" will allow new waveforms to be loaded from the host computer.

```
Fundamental RVP8 Operating Mode: 0
```

The RVP8/IFD and RVP8/Rx cards can operate in one of several fundamental modes for the acquisition of (I,Q) timeseries data. See 6.1.2 RVP8/Rx Receiver Modes on page 205 for details.

NOTE	The receiver mode is chosen in the "Mc" menu, but changes do not take effect until they are saved and the RVP8 is restarted.

4.2.2 Mp — Processing Options

```
Window– 0:User, 1:Rect, 2:Hamming, 3:Blackman : 0
```

Whenever power spectra are computed by the RVP8, the time series data are multiplied by a (real) window prior to computation of the Fourier Transform. You may use whichever window has been selected via SOPRM word #10, or force a particular window to be used.

```
R2 Processing– 0:Never, 1:User, 2:Always : 1
```

Controls R0/R1 versus R0/R1/R2 processing. Selecting "0" unconditionally disables the R2 algorithms, regardless of what the host computer requests in the SOPRM command. Likewise, selecting "2" unconditionally enables R2 processing. These choices allow the RVP8 to run one way or the other without having to rewrite the user code. This is useful for compatibility with existing applications.

```
Clutter Microsuppression– 0:Never, 1:User, 2:Always : 1
```

Controls whether individual "cluttery" bins are rejected prior to being averaged in range. Same interpretation of cases as for "R2 Processing" above.

```
2D Final Speckle/Unfold – 0:Never, 1:User, 2:Always : 1
```

The Doppler parameter modes (PPP, DFT, etc) include an optional 3x3 interpolation and speckle removal filter that is applied to the final output rays. This 2–dimensional filter examines three adjacent range bins from three successive rays in order to assign a value to the center point. Thus, for each output point, its eight neighboring bins in range and time are available to the filter. Only the *dBZ, dBT, Vel,* and *Width* data are candidates for this filtering step; all other parameters are processed using the normal 1–dimensional (three bins in range) speckle remover. See 6.4.3 Speckle Filters on page 248 for more details.

```
Unfold Velocity (Vh–Vl) – 0:Never, 1:User, 2:Always : 0
```

This question allows you to choose whether the RVP8 will unfold velocities using a simple (*Vhigh – Vlow*) algorithm, rather than the standard algorithm described in 6.7 Dual PRF Velocity Unfolding on page 260. Bit–11 of SOPPRM word #10 is the host computer's interface to this function when the "1:User" case is selected (See 7.3 Setup Operating Parameters (SOPRM) on page 279).

NOTE This setup question is included for research customers only. The standard unfolding algorithm should still be used in all operational systems because of its lower variance. For this reason, the factory default value of this parameter is "0:Never".

```
Process w/ custom trigs – 0:Never, 1:User, 2:Always : 0
```

This question allows you to choose whether the RVP8 will attempt to run its standard processing algorithms even when a custom trigger pattern has been selected via the SETPWF command. Generally it does not make sense to do this, so the default setting is "0:Never". Bit–12 of OPPRM word #10 is the host computer's interface to this function when the "1:User" case is selected (See 7.3 Setup Operating Parameters (SOPRM) on page 279).

```
Use High-SNR 16-bit packed timeseries format: Yes
```

This parameter provides an additional 6dB of SNR. It can be disabled to provide compatibility with legacy systems.

```
Minimum freerunning ray holdoff: 100% of dwell
```

This parameter controls the rate at which the RVP8 processes free–running rays. This prevents rays from being produced at the full CPU limit or I/O limit of the processor (whichever was slower); which could result in highly overlapping data being output at an unusably fast rate. Note that this behavior will only occur when running without angle syncing, such as during IRIS Manual and RHI scans.

To make these free–running modes more useful, you may establish a minimum holdoff between successive rays, expressed as a percentage of the number of pulses contributing to each ray. Choosing 100% (the default) will produce rays whose input data do not overlap at all, that is, whose rate will be exactly the PRF divided by the sample size. Choosing 0% will give the unregulated behavior in which no minimum overlap is enforced and rays may be produced very quickly.

Limits: 0 to 100%

```
Linearized saturation headroom: 4.0 dB
```

The RVP8 uses a statistical saturation algorithm that estimates the real signal power correctly even when the IF receiver is overdriven (that is, for input power levels above +4dBm). The algorithm works quite well in extending the headroom above the top end of the A/D converter, although the accuracy decreases as the overdrive becomes more severe. This parameter allows you to place an upper bound on the maximum

extrapolation that will ever be applied. Choosing 0dB will disable the algorithm entirely.

Limits: 0 to 5dB

```
Apply amplitude correction based on Burst/COHO: YES
```

```
Time constant of mean amplitude estimator: 70 pulses
```

The RVP8 can perform pulse–to–pulse amplitude correction of the digital (I,Q) data stream based on the amplitude of the Burst/COHO input. Please see 6.1.7 Correction for Tx Power Fluctuations on page 216 for a complete discussion of this feature.

Limits: 10 to 500 pulses

```
IFD built–in noise dither source: -57.0dBm
```

This question will only appear if the processor is attached to a Rev.D RVP8/IFD that includes an out–of–band noise generator to supply dither power for the A/D converters. The available power levels are { Off, -57dBm, -37dBm, -32dBm, -27dBm, -22dBm, -19dBm }. The closest available level to your typed–in value will be used. You can observe the band–limited noise easily in the **Pr** plot to confirm its amplitude and spectral properties.

For standard operation, we recommend running at -57dBm. The problem higher levels of dither level is that, for certain choices of (I,Q) FIR filter, the stopband of the filter may not give enough attenuation to preserve the RVP8/IFD's inherent noise level. For example, the factory default 1MHz bandwidth Hamming filter has a stopband attenuation near DC of approximately 43dB. You can see this graphically at the right edge of the **Ps** menu. The in–band contribution of dither power is therefore approximately (-37dBm) – 43dB = -80dBm, which exceeds the A/D converter's 1MHz bandwidth noise of -81.5dBm.

```
IFD Wide Dynamic Range Parameters
```

```
Channel separation: 20.00 dB, 0.0 deg
```

```
Maximum deviation : 0.50 dB, 5.0 deg
```

```
Overlap/Interpolate interval: 30.00 dB
```

The *Channel Separation* and *Overlap/Interpolate Interval* should be determined from the **Pr** printout described below. Sweep a SigGen across the shared power region of the two channels to determine a representative channel separation, along with the size of the overlap region at the top of the HiGain channel within which that separation remains steady and

constant, that is, unaffected by eventually approaching the noise floor of the LoGain channel.

The RVP8 continually measures and updates the complex channel separation during normal operation. Ratios of echoes that fall within the overlap/interpolate interval are averaged over several minutes, thereby tracking gain and phase variations that occur with temperature changes and component aging. If the channel separation ever exceeds the specified maximum deviation, the GI4S_IFDCHANERR bit (11) will be set in GPARM Immediate Status Word #4.

```
TAG bits to invert          AZ:0000       EL:0000
TAG scale factors           AZ:1.0000     EL:1.0000
TAG offsets (degrees)       AZ:0.00       EL:0.00
```

The incoming TAG input bits may be selectively inverted via each of the 16–bit words. The values are displayed in Hex. Setting a bit will cause the corresponding AZ (bits 0–15) or EL (bits 16–31) lines to be inverted. Note that the SOPRM command also specifies TAG bits to invert. Both specifications are XOR'ed together to yield the net inversion for each TAG line.

The overall operations are performed in the order listed. Incoming bits are first inverted according to the two 16–bit XOR masks. This yields an unsigned 16–bit integer value which is then multiplied by the signed scale factor. The result is interpreted as a 16–bit binary angle (in the low sixteen bits), to which the offset angle is finally added.

As an example, suppose that the elevation angle input to the RVP8 was in an awkward form such as unsigned integer tenths of degrees, that is, 0x0000 for zero degrees, 0x000a for one degree, 0x0e06 for minus one degree, etc. If we apply a scale factor of 65536/3600 = 18.2044 to these units, we will get 16–bit binary angles in the standard format. If we further suppose that the input angle rotated "backwards", we could take care of this too using a multiplier of -18.2044.

```
Interference Filter  0:None, Alg.1, Alg.2, Alg.3: 1

Threshold parameter C1: 10.00 dB

Threshold parameter C2: 12.00 dB
```

The RVP8 can optionally apply an interference filter to remove impulsive–type noise from the demodulated (I,Q) data stream. See 6.1.5 Interference Filter on page 212 for a complete descriptionof this family of algorithms.

```
Provide WSR88D legacy BATCH major mode: YES
```

```
Maximum range to unfold: 600.0 km

Low–PRF bins range averaged on each side: 2

Overlay power – Refl:5.0dB Vel:8.0dB Width:12.0db

LowSamps = ( 0.00000 x HiSamps ) + 6.00 :

LowPRF = ( 0.00000 x HiPRF ) + 250.00 :
```

This is actually a fully general implementation of a Lo/Hi Surveillance/ Doppler PRF unfolding scheme that provides all of the legacy features as special cases. The parameters are defined as follows:

- The maximum range to unfold is given in km. This allows you to set an upper bound on how many Doppler trips will be unfolded according to the echoes seen in the surveillance data.
- The surveillance data set uses very few pulses and therefore is somewhat noisy. You may choose the number of bins that will be range averaged from both sides of these bins to provide a lower variance power estimate. A value of zero means "No averaging", a value of one would average three points total, etc.
- The unfolding algorithm flags obscured range bins according to three different power thresholds for reflectivity, velocity, and width, and outputs these bits in the DB_FLAGS data parameter. Each of these thresholds is specified in deciBels.
- The fundamental RVP8 operating parameters (PRF, Sample Size, etc) all apply to the high PRF portion of the BATCH trigger waveform. The low PRF rate and sample size are derived from these high values using a slope and offset. In the example shown above, the slopes are both zero, so that the surveillance data will be fixed at 6–pulses and 250–Hz. Making the slopes nonzero would cause the low–PRF parameters to vary automatically if desired.

These setup parameters are accessible through the DSP driver using the new entry points *dspw_batchSetup()* and *dspw_batchSetup()*. These use the custom opcode that is defined separately by each major mode, so you may find *customUserOpcode_batch()* to be a useful model for how to build such things.

```
Polarimetric Power Params – NoiseCorrected:YES

Polarimetric Correlations – NoiseCorrected:YES

PhiDP – Negate: NO , Offset:0.0 deg

KDP – Length: 5.00 km
```

The first question decides whether noise corrected echo powers are used for computing ZDR. The second question specifies whether RhoHV, RhoH and RhoV normalized to noise corrected echo powers. These settings are relevant when echo powers are low.

The third and forth questions define the sign and offset corrections for φ_{DP} and the length scale for *KDP*.

```
T/Z/V/W computed from: H–Xmt:YES V–Xmt:YES

T/Z/V/W computed from: Co–Rcv:YES Cx–Rcv:NO
```

These questions control how the standard parameters (Total Reflectivity, Corrected Reflectivity, Velocity, and Width) are computed in a multiple polarization system. Answering *YES* to *H–Xmt* and/or *V–Xmt* means that data from those transmit polarizations should be used whenever there is more than one choice available. Thus, these selections only apply to the Alternating and Simultaneous transmit modes. Likewise, answering *YES* to *Co–Rcv* and/or *Cx–Rcv* means to use the received data from the co–channel or cross–channel. The receiver question will only appear when dual simultaneous receivers have been configured.

A typical installation might use *H–Xmt:YES, V–Xmt:YES, Co–Rcv:YES, Cx–Rcv:NO*. This will compute (T/Z/V/W) from the co–polarized receiver using both H&V transmissions. Including both transmissions will decrease the variance of (T/Z/V/W); although some researchers prefer excluding *V–Xmt* because that is more standard in the literature. Also, if your polarizations are such that the main power is returned on the cross channel, then you will probably want *Co–Rcv:NO* and *Cx–Rcv:YES*.

```
DualRx – Sum H+V Time Series: NO
```

In dual–receiver systems, you may choose whether the (H+V) time series data consist of the sum of the "H" and "V" samples or the concatenation of half the "H" samples followed by half the "V" samples. The later is more useful when custom software is being used to analyze the data from the two separate receive channels.

4.2.3 Mf — Clutter Filters

```
Residual clutter LOG noise margin: 0.15 dB/dB
```

Whenever a clutter correction is applied to the reflectivity data, the LOG noise threshold needs to be increased slightly in order to continue to provide reliable qualification of the corrected values. The reason for this is that the uncertainty in the corrected reflectivity becomes greater after the clutter is subtracted away.

For example, if we observe 20dB of total power above receiver noise, and then apply a clutter correction of 19dB, we are left with an apparent weather signal power of +1dB above noise. However, the uncertainty of this +1dB residual signal is much greater than that of a pure weather target at the same +1dB signal level.

The "Residual Clutter LOG Noise Margin" allows you to increase the LOG noise threshold in response to increasing clutter power. In the previous example, and with the default setting of 0.15dB/dB, the LOG threshold would be increased by 19x0.15 = 2.85dB. This helps eliminate noisy speckles from the corrected reflectivity data.

```
Spectral Clutter Filters
Filter #1     Type:0(Fixed)       Width:1          EdgePts:2
Filter #2     Type:0(Fixed)       Width:2          EdgePts:2
Filter #3     Type:0(Fixed)       Width:3          EdgePts:3
Filter #4     Type:1(Variable)    Width:3          EdgePts:2        Hunt:3
Filter #5     Type:2(Variable)    (Gaussian Model)
                                  Spectrum width: 0.200 m/sec
Filter #6     Type:2(Variable)    (Gaussian Model)
                                  Spectrum width: 0.300 m/sec
Filter #7     Type:2(Variable)    (Gaussian Model)
                                  Spectrum width: 0.500 m/sec
```

These questions define the heuristic clutter filters that operate on power spectra during the DFT–type major modes. Filter #0 is reserved as "all pass", and cannot be re–defined here. For filters #1 through #7, enter a digit to choose the filter type, followed by however many parameters that type requires. The three filter types are all described in detail in 6.2.5 Clutter Filtering Approaches on page 225.

Fixed Width Filters (Type 0)

These are defined by two parameters. The "Width" sets the number of spectral points that are removed around the zero velocity term. A width of one will remove just the DC term; a width of two will remove the DC term plus one point on either side; three will remove DC plus two points on either side, etc. Spectral points are removed by replacing them with a linear interpolating line. The endpoints of this line are determined by taking the minimum of "EdgeMinPts" past the removed interval on each side.

Variable Width, Single Slope (Type 1)

The RVP8 supports variable–width frequency–domain clutter filters. These filters perform the same spectral interpolation as the fixed–width filters, except that their notch width automatically adapts to the clutter. The filters are characterized by the same *Width* and *EdgePts* parameters in the **Mf** menu, except that the *Width* is now interpreted as a minimum width. An additional parameter *Hunt* allows you to choose how far to extend the notch beyond Width in order to capture all of the clutter power. Setting *Hunt=0* effectively converts a variable–width filter back into a fixed–width filter.

The algorithm for extending the notch width is based on the slope of adjacent spectral points. Beginning (Width-1) points away from zero, the filter is extended in each direction as long as the power continues to decrease in that direction, up to adding a maximum of Hunt additional points. If you have been running with a fixed Width=3 filter, you might try experimenting with a variable *Width=2* and *Hunt=1* filter. Perhaps the original fixed width was actually failing at times, but you were reluctant to increase it just to cover those rare cases. In that case, try selecting a variable *Width=2* and *Hunt=2* filter as an alternative. In general, make your variable filters "wider" by increasing *Hunt* rather than increasing *Width*. This will preserve more flexibility in how they can adapt to whatever clutter is present.

Gaussian Model Adaptive Processing (GMAP) (Type 2)

This type of processing is the most advanced form of clutter filtering and moment estimation (see 6.2.5 Clutter Filtering Approaches on page 225). For GMAP processing, the only thing that needs be specified is the spectrum width of clutter. Note that the algorithm is not too sensitive to the exact value of this. Several widths should be configured to cover the antenna rotation rates that are commonly used. It is useful to turn off clutter filtering (select the all pass filter #0) and then look at actual measurements of the clutter width while the antenna is rotating, for example, using the ascope utility or application software such as the Vaisala IRIS system.

```
Whitening Parameters for Tx:Random

Secondary SQI Threshold Slope:0.50 Offset:-0.05
```

The two values in this question define a secondary SQI threshold that is used to qualify the LOG data during Random Phase processing. The secondary SQI level is computed by multiplying the primary user–supplied SQI threshold by the SLOPE, and adding the OFFSET. See also 6.8.3 Tuning for Optimal Performance on page 267.

Limits: SLOPE: 0.0 to 2.0, OFFSET -2.0 to 1.0

4.2.4 Mt — General Trigger Setups

These questions are accessed by typing "Mt" with no additional arguments. They configure general properties of the RVP8 trigger generator

```
Pulse Repetition Frequency: 500.00 Hz
```

This is the Pulse Repetition Frequency of the internal trigger generator. Limits: 50 to 20000Hz.

```
Transmit pulse width: 0
```

Limits: 0 to 3

```
Use external pretrigger: NO
```

```
PreTrigger active on rising edge: YES
```

```
PreTrigger is synchronous with IFD AQ clock: No
```

```
PreTrigger fires the transmitter directly: NO
```

When an external pretrigger is applied to the TRIGIN input of the RVP8, either the rising or falling edge of that signal initiates operation. This decision also affects which signal edge becomes the reference point for the pretrigger delay times given in the "**Mt**<n>" section.

Answer the second sub–question according to whether the radar transmitter is directly fired by the the external pretrigger, rather than by one of the RVP8's trigger outputs. In other words, answer "YES" if the transmitter would continue running fine even if the RVP8 TRIGIN signal were removed. This information is used by the **L** and **R** subcommands of the **Pb** plotting command, that is, when slewing left and right to find the burst pulse, the pretrigger delay will be affected rather than the start times of the six output triggers.

```
Number of user–defined output triggers: 6
```

This setting defines the number of user–defined output triggers.

Limit: 12 (including polarization output controls)

```
Number of polarization output controls: 2
```

This setting defines the number of polarization output controls.

```
2–way (Tx+Rx) total waveguide length: 0 meters
```

Use this question to compensate for the offset in range that is due to the length of waveguide connecting the transmitter, antenna, and receiver. You should specify the total 2–way length of waveguide, that is, the span from transmitter to antenna, plus the span from antenna to receiver. The RVP8 range selection will compensate for the additional waveguide length to within plus–or–minus half a bin, and works properly at all range resolutions.

```
POLAR1 is high for vertical polarization : NO
```

```
POLAR2 is high for vertical polarization : NO
```

These questions define the logical sense of the two polarization control signals POLAR1 and POLAR2. In a dual–polarization radar POLAR1 should be used to select one of two possible states (nominally horizontal and vertical, but any other polarization pair may also be used). The control signal will either remain at a fixed level, or will alternate from pulse to pulse with a selectable transition point (See 4.2.5 Mt<n>— Triggers for Pulsewidth #n on page 133). POLAR2 is identical to POLAR1, but may be configured with a different polarity and switch point. This second signal could be used if the radar's polarization switch required more than one control line transition when changing states.

```
Quantize trigger PRT to ((1 x AQ) + 0) clocks
```

It is possible to control the exact quantization of the PRT of the internal trigger generator. Normally the trigger PRT is chosen as the closest multiple of AQ (the acquisition clock period) that approximates the requested period. This question allows the possible PRT's to be constrained to ((N x AQ) + M) clock cycles. This feature can be useful for synchronous receiver systems in which the trigger period must be some exact multiple of the COHO period.

```
Blank output triggers within AZ and EL sectors: NO

Sector #1  InUse:NOAZ: 0.0,0.0        EL: 0.0,0.0
Sector #2  InUse:NOAZ: 0.0,0.0        EL: 0.0,0.0
Sector #3  InUse:NOAZ: 0.0,0.0        EL: 0.0,0.0
Sector #4  InUse:NOAZ: 0.0,0.0        EL: 0.0,0.0
Sector #5  InUse:NOAZ: 0.0,0.0        EL: 0.0,0.0
Sector #6  InUse:NOAZ: 0.0,0.0        EL: 0.0,0.0
Sector #7  InUse:NOAZ: 0.0,0.0        EL: 0.0,0.0
Sector #8  InUse:NOAZ: 0.0,0.0        EL: 0.0,0.0
```

These settings can be modified to reduce erroneous transmissions into physical obstructions.

```
Blank output triggers via softplane sTrigBlank : NO

Blank triggers 1:YES 2:YES 3:YES 4:YES 5:YES 6:YES
```

These questions control trigger blanking based on the TAG0 input line. You first select whether the trigger blanking feature is enabled; and then optionally choose the polarity of TAG0 that will result in blanking, and which subset of the six user definable triggers are to be blanked.

```
Blank output triggers during noise measurement : NO
```

The RVP8 can inhibit the subset of blankable trigger lines whenever a noise measurement is taken. This will be forced whenever trigger blanking (based on TAG0) is enabled, but it can also be selected in general via this question. Since noise triggers must be blanked whenever trigger blanking is enabled, this question only appears if trigger blanking is disabled.

This question permits the state of the triggers during noise measurements to be consistent and known, regardless of whether the antenna happens to be within a blanked sector; and you have the additional flexibility of choosing blanked noise triggers all the time.

```
Rx-Fixed Triggers: #1:N #2:N #3:N #4:N #5:N #6:N P0:N P1:N
Z:N
```

You have explicit control over which RVP8 trigger outputs are timed relative to the transmitter pre–fire sequence, versus those which are relative to the actual received target ranges. Triggers in the first category will be moved left/right by the "L/R" keys in the **Pb** plot, and will also be slewed in response to Burst Pulse Tracking. Triggers in the second category remain fixed relative to "receiver range zero", and are not affected by the "L/R" keys or by tracking.

This question specifies which triggers are Tx–relative and which are Rx–relative. Answer with a sequence of "Y" or "N" responses for each of the six trigger lines, for the two polarization control lines, and for the timing of the phase control lines. You should answer *No* for any trigger that is involved with the pre–fire timing of the transmitter. If you enable the Burst Pulse Tracker (6.1.4 Burst Pulse Tracking on page 211) you will probably want to assign a *Yes* to some of your triggers so that they remain fixed relative to the burst itself.

It is very helpful to have these two categories of trigger start times. Triggers that fire the transmitter, either directly or indirectly, should all be moved as a group when hunting for the burst pulse and moving it to the center of the FIR window. However, triggers that function as range strobes should be fixed relative to range zero, that is, the center of that window,

and the center of the burst. This distinction becomes important when the transmitter's pre–fire delay drifts with time and temperature.

```
Merge triggers to create composite waveforms: YES

Merge Trigger #1 into :     #1:     #2:     #3:     #4:     #5:     #6:
Merge Trigger #2 into :     #1:N    #2:     #3:     #4:     #5:     #6:
Merge Trigger #3 into :     #1:Y    #2:     #3:     #4:     #5:     #6:
Merge Trigger #4 into :     #1:     #2:Y    #3:     #4:     #5:     #6:
Merge Trigger #5 into :     #1:     #2:Y    #3:     #4:     #5:     #6:
Merge Trigger #6 into :     #1:     #2:     #3:     #4:     #5:     #6:
```

These questions allow you to merge the six user triggers together; resulting in trigger patterns that can be much more complex. In this example, Trigger #3 will be merged into Trigger #1; Trigger #3 will be unaltered, and Trigger #1 will be the "OR" of itself with Trigger #3. Likewise, Triggers #4 and #5 will be merged into Trigger #2 so that the later will contain three distinct pulses within each PRT. Answer each question with a sequence of up to six "Y" or "N" responses in order to set the merged destinations for each trigger line.

Note that the six triggers are still defined in the usual way in the **Mt<n>** menu, that is, start time, width, etc. The only change is that you may now combine these individual pulse definitions into a more complex composite output waveform.

4.2.5 Mt<n>— Triggers for Pulsewidth #n

These questions are accessed by typing "Mt", with an additional argument giving the pulsewidth number. They configure specific trigger, transmit waveform, and FIR filter properties for the indicated pulsewidth only.

```
Trigger          #1  Start:        0.00 usec
                 #1  Width:        1.00 usec High:YES
Trigger          #2  Start:        0.00 usec + (
                                   0.500000 * PRT )
                 #2  Width:        10.00 usec High:YES
Trigger          #3  Start:        3.00 usec
                 #3  Width:        1.00 usec High:YES
Trigger          #4  Start:        2.00 usec
                 #4  Width:        1.00 usec High:YES
```

```
Trigger            #5  Start:         1.00 usec
                   #5  Width:         1.00 usec High:YES
Trigger            #6  Start:         5.00 usec +
                                      (0.001000 * PRT )
                   #6  Width:         2.00 usec High:NO
```

These parameters list the starting times (in microseconds relative to range zero), the widths (in microseconds), and the active sense of each of the six triggers generated by the internal trigger generator. Setting a width to zero inhibits the trigger on that line.

The Start Time can include an additional term consisting of the pulse period times a fractional multiplier between -1.0 and +1.0. This allows you to produce trigger patterns that would not otherwise be possible, for example, a trigger that occurs half way between every pair of transmitted pulses, and remains correctly positioned regardless of changes in the PRF Enter this multiplier as "0" if you do not wish to use this term, and it will be omitted entirely from the printout.

In the above example, Trigger #2 is a 10.0 µsec active–high pulse whose leading edge occurs precisely halfway between the zero–range of every pair of pulses. Likewise, Trigger #6 is a 2.0 µsec active–low pulse whose falling edge is nominally 5.0 µsec prior to range zero, but which is advanced by 1.0 µsec for every millisecond of trigger period. All other triggers behave normally, and have fixed starting times that do not vary with trigger period.

Some subtleties of these variable start times are:

- The PRT multipliers can only be used in conjunction with the RVP8's internal trigger generator. The PRT–relative start times are completely disabled whenever an external trigger source is chosen from the **Mt** menu.
- When PRT–relative triggers are plotted by the **Pb** command, the active portion of the trigger will be drawn cross–hatched and at a location computed according to the current PRF. The cross–hatching serves as a reminder that the actual location of that trigger may vary from it's presently plotted position.
- The PRT multiplier for a given pulse is applied to the interval of time between that pulse and the next one. This distinction is important whenever the RVP8 is generating multiple–PRT triggers, for example, during DPRT mode, or during Dual–PRF processing. Multipliers from 0.0 to +1.0 are generally safe to use because they shift the trigger into the same pulse period that originally defined it. For example, a start time of (0.0 µsec + (0.98 * PRT)) would position a trigger 98% of the way up to the next range zero. But, if -0.98 were

used, and if the period of the previous pulse was shorter than the current one, then that shorter period would become incorrect (longer) as a result of having to fit in the very early trigger.

A small but important detail is built into the algorithm for producing the six user trigger waveforms. It applies whenever a) the trigger period is internally determined, that is, the external pretrigger input is not being used, and b) the overall span of the six trigger definitions combined does not fit into that period. What happens in this case is that any waveforms that do not fit will be zeroed (not output) so that the desired period is preserved. This means that you can define triggers with large positive start times, and they will pop into existence only when the PRF is low enough to accommodate them.

For example, if Trigger #2 is defined as a 200.0µsec pulse starting at +400.0µsec, then that trigger would be suppressed if the PRF were 2000Hz, but it would be present at a PRF of 1000Hz. Whenever a trigger does not completely fit within the overall period it is suppressed entirely. Thus, even though the +400.0µsec start time is still valid at 2000Hz, the entire 200.0µsec pulse would not fit, and so the pulse is eliminated altogether.

Start limits: -5000 to 5000 µsec. Width limits: 0 to 5000 µsec.

```
Maximum number of Pulses/Sec: 2000.0

Maximum instantaneous 'PRF' : 2000.0 (/Sec)
```

These are the PRF protection limits for this pulsewidth.

The wording of the "Maximum number of Pulses/Sec" question serves as a reminder that the number shown is not only an upper bound on the PRF, but also a duty cycle limit when DPRT mode is enabled.

The "Maximum instantaneous 'PRF'" question allows you to configure the maximum instantaneous rate at which triggers are allowed to occur, that is, the reciprocal of the minimum time between any two adjacent triggers. This parameter is included so that you can limit the maximum DPRT trigger rate individually for each pulsewidth. Note that the maximum instantaneous PRF can not be set lower than the maximum number of pulses per second.

PRF limits: 50 to 20000Hz.

```
External pretrigger delay to range zero: 3.00 usec
```

Range Zero is time at which the signal from a target at zero range would appear at the radar receiver outputs. This parameter adjusts the delay from the active edge of the external trigger to range zero. It is important that this

delay be correct when the RVP8 is operating with an external trigger, since the zero range point is a fixed time offset from that trigger. When the transmitter is driven from the internal trigger signals, those signals themselves are adjusted (see Burst Pulse alignment procedures) to accomplish the alignment of range zero.

Limits: 0.1 to 1000 μsec.

```
Range mask spacing: 125.00 meters
```

The range resolution of the RVP8 is determined by the decimation factor of the digital matched FIR filter that computes "I" and "Q". This decimation factor is the ratio of the filter's input and output data rates, that is, the output rate is some integer divisor of the IFD Acquisition Clock (See **Mc** Section). For the legacy RVP7 IFD operating at its standard frequency of 35.9751MHz, the available range resolutions (in meters) are: 25.0, 28.3, 36.7, 50.0, 58.3, 66.7, 75.0, 83.3, 91.7, 100.0, 108.3, 116.7, 125.0, and 133.3. The RVP8 IFD operating in the 72MHz range provides twice the resolution between steps, and four times the number of steps.

The ranges that are selected by the bit mask in the LRMSK command are spaced according to the range resolution that is chosen here. Also, the upper limit on the impulse response length of the matched FIR filter (see below) is constrained by the range resolution. If you choose a range resolution that can not be computed at the present filter length, then a message of the form: "Warning: Impulse response shortened from 72 to 42 taps" will appear.

Limits: 25 to 1000 meters.

```
FIR-Filter impulse response length: 1.33 usec
```

The RVP8 computes "I" and "Q" using a digital FIR (Finite Impulse Response) matched filter. The length of that filter (in microseconds) is chosen here.

The filter length should be based on several considerations:

- It should be at least as long as the transmitted pulsewidth. If it were shorter, then some of the returned energy would be thrown away when "I" and "Q" are computed at each bin. The SNR would be reduced as a result.
- It should be at least as long as the range bin spacing. The goal here is to choose the longest filter that retains statistical independence among successive bins. If the filter length is less than the bin spacing, then no IF samples would be shared among successive bins, and those bins would certainly not be correlated.

- It should be "slightly longer" than either of the above bounds would imply, so that the filter can do a better job of rejecting out–of–band noise and spurious signals. The SNR of weak signals will be improved by doing this.

In practice, a small degree of bin–to–bin correlation is acceptable in exchange for the filter improvements that become possible with a longer impulse response. The FIR coefficients taper off to zero on each end; hence, the power contributed by overlapping edge samples is minimal. Vaisala recommends beginning with an impulse response length of 1.2–1.5 times the pulsewidth or bin spacing, whichever is greater.

The maximum possible filter length is bounded according to the range resolution that has been chosen; a finer bin spacing leaves less time for computing a long filter. For the RVP8 Rev.A processor, the filter length must be less than 2.92 µsec at 125–meter resolution; for Rev.B and higher this limit increases to 6.67 µsec.

NOTE Cascade filter software is being contemplated that will extend the maximum impulse response length to at least 50 µsec. This is of interest when very long (uncoded CW) transmitted pulses are used.

```
Burst Freq Estimator– Length: 1.33 usec, Start: 0.00 usec
```

This estimator is mostly used with the Pb (plotting commands) and can be referenced in 5.3.2 Available Subcommands Within Pb on page 165.

```
FIR–Filter prototype passband width: 0.503 MHz
```

This is the passband width of the ideal lowpass filter that is used to design the matched FIR bandpass filter. The actual bandwidth of the final FIR filter will depend on 1) the filter's impulse response length, and 2) the design window used in the process. The actual 3dB bandwidth will be:

- Larger than the ideal bandwidth if that bandwidth is narrow and the FIR length is too short to realize that degree of frequency discrimination. In these cases it may be reasonable to increase the filter length.
- Smaller than the ideal bandwidth if the FIR length easily resolves the frequency band. This is because of the interaction within the filter's transition band of the ideal filter and the particular design window being used. For example, for a Hamming window and sufficiently long filter length, the ideal bandwidth is an approximation of the 6dB (not 3dB) attenuation point. Hence, the 3dB width is narrower than the ideal prototype width.

This parameter should be tuned using the TTY output and interactive visual plot from the **Ps** command. The actual 3dB bandwidth is shown there, so that it can be compared with the ideal prototype bandwidth.

Limits: 0.05 to 10.0 MHz.

```
Output control 4-bit pattern: 0001
```

These are the hardware control bits for this pulsewidth. The bits are the 4–bit binary pattern that is output on PWBW0:3

Bit Limits: 0 to 15 (input must be typed in decimal)

```
Current noise level: -75.00 dBm

Powerup noise level: -75.00 dBm
```

–or–

```
Current noise levels – PriRx: -75.00 dBm, SecRx: -75.00 dBm

Powerup noise levels – PriRx: -75.00 dBm, SecRx: -75.00 dBm
```

These questions allow you to set the current value and the power–up value of the receiver noise level for either a single or dual receiver system. The noise level(s) are shown in dBm, and you may alter either one from the TTY. The power–up level(s) are assigned by default when the RVP8 first starts up, and whenever the RESET opcode is issued with Bit #8 set. Likewise, the current noise level is revised whenever the SNOISE opcode is issued. These setup questions are intended for applications in which the RVP8 must operate with a reasonable default value, up until the time that an SNOISE command is actually received. They may also be used to compare the receiver noise levels during normal operation, which serves as a check that each FIR filter is behaving as expected when presented with thermal noise.

```
Transmitter phase switch point: -1.00 usec
```

This is the transition time of the RVP8's phase control output lines during random phase processing modes. The switch point should be selected so that there is adequate settling time prior to the burst/COHO phase measurement on each pulse. This question only appears if the PHOUT[0:7] lines are actually configured for phase control (See 4.2.1 Mc — Top Level Configuration on page 120).

Limits: -500 to 500 µsec.

```
Polarization switch point for POLAR1: -1.00 usec
```

```
Polarization switch point for POLAR2: 1.00 usec
```

The RVP8's POLAR1 and POLAR2 digital output lines control the polarization switch in a dual–polarization radar. During data processing modes in which the polarization alternates from pulse to pulse, the transition points of these control signals are set by these two questions. The values are in microseconds relative to range zero; the same units used to define the start times of the six user triggers. The logical sense of POLAR1 and POLAR2 is set by questions described in 4.2.4 Mt — General Trigger Setups on page 130.

Limits: -500 to 500 µsec.

4.2.6 Special Options for Tx Synthesis

Several of the dialogs described in the previous section will be modified when the RVP8 is equipped with an RVP8/Tx Digital Transmitter Card that has been configured for Tx waveform synthesis in the **Mz** menu. In this case, each of the RVP8's four "pulsewidths" can select an entirely different type of transmit waveform and associated matched receiver.

For example, PW–0 and PW–1 could transmit conventional 0.5µsec and 2.0µsec CW pulses that are received using the bandpass filters described in 4.2.5 Mt<n>— Triggers for Pulsewidth #n on page 133. But within this same system, PW–2 and PW–3 could be further configured as, perhaps, 20µsec and 40µsec compressed non–linear FM waveforms. This makes it very easy for application software such as **ascope** to transparently switch between radically different Tx waveforms simply by requesting a different "pulsewidth" for each one.

The following questions will appear in the **Mt**<n> menu (immediately after the *Range Mask Spacing* question) when digital Tx waveforms are being synthesized.

```
Tx Waveform - 0:CWPulse, 1:LinFM, 2:NLFM : 2
```

The RVP8 supports three standard Tx waveforms: a conventional fixed–frequency CW pulse, a linear FM chirp, and non–linear FM. The CWPulse can be used as a pulsed Doppler waveform in all the same ways that a Klystron or Magnetron system having a traditional pulse forming network would be used. The linear and non–linear FM waveforms, however, are compressed pulses that are intended to be transmitted by a wide–bandwidth Klystron/TWT/SolidState amplifier.

NOTE The RVP8 internal APIs permit code developers to create arbitrary waveforms for transmission. The three types mentioned above are the out–of–the–box selections that are standard on all RVP8 processors.

```
Bandwidth of transmit pulse: 3.25 MHz

Pulselength of transmit pulse: 15.00 usec
```

These questions select the bandwidth and pulse length of the Tx waveform. The bandwidth value represents the true spectrum width of the complete waveform, that is, including all the effects of whatever frequency modulation and amplitude modulation the waveform happens to use. Thus, a spectrum analyzer (or the RVP8's **Ps** plot) would show an overall spectrum width equal to this desired value.

Likewise, the pulse length value represents the entire time duration of the waveform, including whatever amplitude modulations may be included at the tails.

```
Zero offset of transmit pulse: 0.00 usec
```

The Tx waveform is normally synthesized with its center lined up with range zero. If the radar's high–power amplifier had zero delay, this would serve to define the middle of the transmit pulse as range zero, which is the usual RVP8 convention. This offset question is provided so that the RVP8/Tx output waveform can be shifted in time to compensate for whatever delays are present in the radar's IF/RF electronics.

NOTE This transmit pulse timing offset is typically checked via the **Pb** plot by making sure that the Tx burst is centered within the FIR data window.

```
TxWave MIN tuning params: 0.0000, 0.0000, 0.0000

TxWave MAX tuning params: 1.0000, 1.0000, 1.0000

TxWave tuning parameters: 0.9500, 1.0000, 0.0490
```

The RVP8 uses three real–valued tuning parameters to make the synthesis of complex waveforms more flexible. Each waveform class can be altered and fine tuned with up to three degrees of freedom, making it possible for a single class (for example, the non–linear FM class) to generate a huge variety of actual waveforms. These adjustable constants also form the basis of the automatic waveform optimization procedure described for the **Pa** command in 5.6.2 Available Subcommands Within Pa on page 191.

Each of the three parameters has a minimum value, a maximum value, and a current value, all of which can be changed from this menu. The Min/Max limits are used within the **Pa** command to maintain sensible bounds as the parameters are adjusted. In general, the Min/Max values will be entered from the **Mt<n>** menu, but the actual values will be tuned using either manual or automatic procedures found in the **Pa** command.

The CWPulse class of waveforms do not use any of the tuning parameters because the Tx waveform is completely determined by the desired bandwidth and pulsewidth, that is, there are no remaining degrees of freedom to adjust. Thus, these three questions do not appear in the CWPulse case.

The linear FM class is also entirely specified by just the bandwidth and pulsewidth values, and does not reference any of the tuning parameters. However, the non–linear FM class is the most flexible of all, and references all three tuning parameters as follows:

- Parameters #1 and #2 are the (X,Y) location of the non–linear "breakpoint" for the FM curve. Referring to the white plot line in Figure 28 on page 190, the Time/Frequency behavior of the pulse can be drawn in a coordinate system whose abscissa ranges from -1 to +1 over the complete time duration of the pulse, and whose ordinate ranges from -1 to +1 over the complete frequency span of the pulse.
- The class of non–linear FM curves always pass through the points (-1,-,1), (0,0), and (1,1), that is, they begin at the lowest frequency at the start of the pulse, end at the highest frequency when the pulse completes, and pass through the origin (to maintain symmetry across both halves of the pulse). Between the points (0,0) and (1,1) the curves also pass through the tunable (X,Y) "breakpoint" defined by the first two parameters. In other words, the positive–,time portion of the FM curve consists of two linear segments; one from (0,0) to (X,Y), and the other from (X,Y) to (1,1). By tuning the breakpoint we create a diverse class of FM modulations, but all of them adhere to the physical bandwidth and pulsewidth limits imposed by the earlier setup questions. Note that to maintain symmetry, the breakpoint is also mirrored on the negative–,time side as line segments from (-1,-1) to (–X,–Y), and from (–X,–Y) to (0,0).
- Parameter #3 specifies the X location of the start of the amplitude taper of the non–linear FM waveform. For example, setting X to 0.95 will result in a pulse having full amplitude over the middle 95% of its duration, but then having raised cosine amplitude weighting applied to the leading and trailing 5% of its edges.

Some examples may be helpful:

P1 = 0.0, P2 = 0.0, P3 = 1.0

P1 and P2 place the FM breakpoint at the origin. But the FM curve passes through that point anyway, so the response reverts to linear FM. P3 indicates that amplitude modulation should not be applied until the very end of the pulse, and thus will not occur at all. The resulting waveform is therefore linear FM having abrupt On/Off transitions.

P1 = 0.9, P2 = 0.7, P3 = 1.0

During the middle 90% of the waveform's duration the frequency chirp uses 70% of its available bandwidth. Then, within the 10% pulse tails, the remaining 30% of the bandwidth suddenly gets covered. No amplitude modulation is applied. Pulses of this type have been studied theoretically, but do not perform very well for a given total bandwidth that includes the leading/trailing "ears".

P1 = 0.9, P2 = 1.0, P3 = 0.8

The entire frequency band is chirped within the middle 90% of the pulse duration, so that the frequency remains constant in the 10% pulse tails. An amplitude modulation is also applied over 20% of the pulse tails, that is, encompassing both the ends of the chirp and the entire constant frequency intervals. Pulses of this type have superior sidelobe behavior and fit very neatly within their prescribed bandwidth limits. We recommend using non–linear FM waveforms that combine chirp limits and amplitude modulation in this manner.

4.2.7 Mb — Burst Pulse and AFC

These questions are accessed by typing "Mb". They set the parameters that influence the phase and frequency analysis of the burst pulse, and the operation of the AFC feedback loop.

```
Receiver Intermediate Frequency: 30.0000 MHz
```

This is the center frequency of the IF receiver and burst pulse waveform.

- With the legacy RVP7 IFD, the RVP8 can operate at an intermediate frequency within any of the alias bands 22–32MHz, 40–50MHz, and 58–68MHz.
- With the CAT–5E IFD, the RVP8 can operate at an intermediate frequency within any of the alias bands 8–32MHz and 40–68MHz

These bands are delineated by 4MHz safety zones on either side of integer multiples of half the IFD's sampling frequency. The value entered here implicitly defines the band being used.

Limits: 6 to 72 MHz.

```
Primary Receiver Intermediate Frequency: 30.0000 MHz
```

```
Secondary Receiver Intermediate Frequency: 24.0000 MHz
```

These alternate questions will replace the previous question whenever the RVP8's dual–receiver mode is selected. You should enter the two intermediate frequencies for your primary and secondary (nominally horizontal and vertical polarized) receivers. Note that you can easily swap receiver channels merely by exchanging the two frequency values.

```
IF increases for an approaching target: YES
```

The intermediate frequency is derived at the receiver's front end by a microwave mixer and sideband filter. The filter passes either the lower sideband or the upper sideband, and rejects the other. Depending on which sideband is chosen, an increase in microwave frequency may either increase (STALO below transmitter) or decrease (STALO above transmitter) the receiver's intermediate frequency. This question influences the sign of the Doppler velocities that are computed by the RVP8.

```
PhaseLock to the burst pulse: YES
```

This question controls whether the RVP8 locks the phase of its synthesized "I" and "Q" data to the measured phase of the burst pulse. For an operational magnetron system this should always be "YES", since the transmitter's random phase must be known in order to recover Doppler data. The "NO" option is appropriate for non phase modulated Klystron systems in which the RVP8/IFD sampling clock is locked to the COHO. It is also useful for bench testing in general. In these "NO" cases the phase of "I" and "Q" is determined relative to the stable internal sampling clock in the RVP8/IFD module.

```
Minimum power for valid burst pulse: -15.0 dBm
```

This is the minimum mean power that must be present in the burst pulse for it to be considered valid, that is, suitable for input into the algorithms for frequency estimation and AFC. The reporting of burst pulse power is described in 5.3 Pb — Plot Burst Pulse Timing on page 163; the value entered here should be, perhaps, 8 dB less. This insures that burst pulses will still be properly detected even if the transmitter power fades slightly.

The mean power level of the burst is computed within the narrowed set of samples that are used for AFC frequency estimation. The narrow

subwindow will contain only the active portion of the burst, and thus a mean power measurement is meaningful. The full FIR window would include the leading and trailing pulse edges and would not produce a meaningful average power. Since radar peak power tends to be independent of pulse width, this single threshold value can be applied for all pulsewidths.

Limits: -60 to +10 dBm.

```
Design/Analysis Window– 0:Rect, 1:Hamming, 2:Blackman : 1
```

You may choose the window that is used in 1) the design of the FIR matched filter, and 2) the presentation of the power spectra for the various scope plots. Choices are rectangular, Hamming, and Blackman; the Hamming window being the best overall choice. The Blackman window is useful if you are trying to see plotted spectral components that are more than 40dB below the strongest signal present. It is especially useful in the "Pr" plot when a long span of data are available. FIR filters designed with the Blackman window will have greater stopband attenuation than those designed with the Hamming window, but the wider main lobe may be undesirable. The rectangular window is included mostly as a teaching tool, and should never be used in an operational setting.

```
Settling time (to 1%) of burst frequency estimator: 5.0 sec
```

The burst frequency estimator uses a 4th order correlation model to estimate the center frequency of the transmitted pulses. Each burst pulse will typically occupy approximately one microsecond; yet the frequency estimate feeding the AFC loop needs to be accurate to, perhaps, 10KHz. Obviously this accuracy can not be achieved using just one pulse. However, several hundred of the (unbiased) individual estimates can be averaged to produce an accurate mean. This averaging is done with an exponential filter whose time constant is chosen here.

Limits: 0.1 to 120 seconds.

```
Lock IFD sampling clock to external reference: NO
```

This question determines the usage of the shared SMA connector that is labeled "AFC/(CLK)" on the RVP8/IFD. It is generally not necessary to phase lock the IFD sampling clock to the radar system clock, since very good stability is obtained from the burst phase measurements during normal operation. However, two cases that benefit from clock locking are 1) using the RVP8 in a klystron system where an external trigger is provided, and 2) dual–receiver systems in which computation of φ_{DP} is important.

The following two questions will appear only if you have requested that the IFD sampling clock be locked to an external clock reference. See 3.2.12 IFD Reference Clock Input (Optional) on page 82 for a description of the hardware setups that must accompany this selection.

```
PLL ratio of (1/1) ==> Input reference at 17.9876 MHz
```

The VCXO phase–locked–loop (PLL) in the RVP8/IFD can work with any input reference clock whose frequency is a rational multiple (P/Q) of half the desired sampling frequency, that is, the center frequency of the VCXO that is entered in the "mc" command (4.2.1 Mc — Top Level Configuration on page 120). This question allows this ratio to be established. In general, the best PLL performance will be attained when the ratio is reduced to lowest terms, for example, use a ratio of 6/5 rather than 12/10. For example, assume:

VCXO center frequency f*samp* = 36 MHz

Reference clock frequency f*ref* = 10 MHz

$$f_{ref} = \frac{P}{Q}\frac{f_{samp}}{2}\ ;\ \frac{P}{Q} = \frac{2f_{ref}}{f_{samp}} = \frac{2 \cdot 10\ MHz}{36\ MHz} = \frac{5}{9}$$

In this case enter the numbers "5 9". The proper input reference frequency (for example, 10.0000 MHz) should be displayed after you enter the values. The **v** command can be used to verify that the PLL is "Okay".

Limits: 1 to 128 for both numerator and denominator.

```
VCXO has positive frequency deviation: YES
```

Most VCXOs have positive frequency deviation, that is, their output frequency increases with increasing input control voltage. This question will generally be answered "yes", but is included to accommodate the other case as well. The PLL will not lock, and will be completely unstable, if the wrong choice is made.

```
Enable AFC and MFC functions: YES
```

AFC is required in a magnetron system to maintain the fixed intermediate frequency difference between the transmitter and the STALO. AFC is not required in a klystron system since the transmitted pulse is inherently at the correct frequency.

The following rather long list of questions will appear only if AFC and MFC functions have been enabled.

```
AFC Servo- 0:DC Coupled, 1:Motor/Integrator : 0
```

The AFC servo loop can be configured to operate with an external Motor/Integrator frequency controller, rather than the usual direct–coupled FM control. This type of servo loop is required for tuned magnetron systems in which the tuning actuator is moved back and forth by a motor, but remains fixed in place when motor drive is removed. These systems require that the AFC output voltage (motor drive) be zero when the loop is locked; and that the voltage be proportional to frequency error while tracking. Please see 4.2.8 AFC Motor/Integrator Option on page 151 for more details.

```
Wait time before applying AFC: 10.0 sec
```

After a magnetron transmitter is first turned on, it may be several seconds or even minutes until its output frequency becomes stable. It would not make sense for the AFC loop to be running during this time since there is nothing gained by chasing the startup transient. This question allows you to set a holdoff delay from the time that valid burst pulses are detected to the time that the AFC loop actually begins running.

Limits: 0 to 300 seconds.

```
AFC hysteresis -- Inner: 5.0 KHz, Outer: 15.0 KHz
```

These are the frequency error tolerances for the AFC loop. The loop will apply active feedback whenever the outer frequency limit is exceeded, but will hold a fixed level once the inner limit has been achieved. The hysteresis zone minimizes the amount of thrashing done by the feedback loop. The AFC control voltage will remain constant most of the time; making small and brief adjustments only occasionally as the need arises.

```
AFC outer tolerance during data processing: 50.0 KHz
```

In general, the AFC feedback loop is active only when the RVP8 is not processing data rays. This is because the Doppler phase measurements are seriously degraded whenever the AFC control voltage makes a change. To avoid this, the AFC loop is only allowed to run in between intervals of sustained data processing. This is fine as long as the host computer allows a few seconds of idle time every few minutes; but if the RVP8 were constantly busy, the AFC loop would never have a chance to run. This question allows you to place an upper bound on the frequency error that is tolerated during sustained data processing. AFC is guaranteed to be applied whenever this limit is exceeded.

Limits: 15 to 4000 KHz.

```
AFC feedback slope: 0.0100 D-Units/sec / KHz

AFC minimum slew rate: 0.0000 DUnits/sec

AFC maximum slew rate: 0.5000 D-Units/sec
```

These questions control the actual feedback computations of the AFC loop.

The overall span of the AFC output voltage is set by Gain and Offset potentiometers on the RVP8/IFD module (See 3.2.11 IFD Analog AFC Output Voltage (Optional) on page 81). The control level that is applied to the AFC's 16–bit Digital–to–Analog converter is specified here in "D–Units", that is, arbitrary units ranging from -100 to +100 corresponding to the complete span of the D/A converter. Since the D–Unit corresponds in a natural way to a percentage scale, the shorter "%" symbol is sometimes used.

AFC feedback will be applied in proportion to the frequency error that the algorithm is attempting to correct. The feedback slope determines the sensitivity and time constant of the loop by establishing the AFC's rate of change in (D–Units / sec) per thousand Hertz of frequency error. For example, a slope of 0.01 and a frequency error of 30KHz would result in a control voltage slew of 0.3 D–Units per second. At that rate it would take approximately 67 seconds for the output voltage to slew one tenth of its total span (20 D–Units / (0.3 D–Units / sec) = 67 sec). AFC is intended to track very slow drifts in the radar system, so response times of this magnitude are reasonable.

Keep in mind that the feedback slew is based on a frequency error which itself is derived from a time averaging process (see Burst Frequency Estimator Settling Time described above). The AFC loop will become unstable if a large feedback slope is used together with a long settling time constant, due to the phase lag introduced by the averaging process. Keep the loop stable by choosing a small enough slope that the loop easily comes to a stop within the inner hysteresis zone.

See 4.2.8 AFC Motor/Integrator Option on page 151 for more information about these slope and slew rate parameters.

```
AFC span– [-100%,+100%] maps into [ -32768 , 32767 ]

AFC format– 0:Bin, 1:BCD, 2:8B4D: 0, ActLow: NO

AFC uplink protocol– 0:Off, 1:Normal, 2:PinMap : 1
```

The RVP8's implementation of AFC has been generalized so that there is no difference between configuring an analog loop and a digital loop. The AFC feedback loop parameters are setup the same way in each case; the only difference being the model for how the AFC information is made available to the outside world. Many types of interfaces and protocols thus become possible according to how these three questions are answered. AFC output follows these three steps:

- The internal feedback loop uses a conceptual [-100%,+100%] range of values. However, this range may be mapped into an arbitrary numeric span for eventual output. For example, choosing the span from -32768 to +32767 would result in 16–bit AFC, and 0 to 999 might be appropriate for 3–digit BCD; but any other span could also be selected from the full 32–bit integer range.
- Next, an encoding format is chosen for the specified numeric span. The result of the encoding step is another 32–bit pattern which represents the above numeric value. Vaisala will make an effort to include in the list of supported formats all custom encodings that our customers encounter from their vendors.
- Available formats include straight binary, BCD, and mixed–radix formats that might be required by a specialized piece of equipment. The "8B4D" format encodes the low four decimal digits as four BCD digits, and the remaining upper bits in binary. For example, 659999 base–10 would encode into 0x00419999 Hex.
- Finally, an output protocol is selected for the bit pattern that was produced by encoding the numeric value. The bits may be written to the eight RVP8 backpanel RS232 outputs, or sent on the uplink as a value to be received by the RVP8/IFD and converted to an analog voltage. Yet another option is for the bits to be sent on the uplink and received by the DAFC, which supports arbitrary remapping of its output pins.

To summarize: the internal AFC feedback level is first mapped into an arbitrary numeric span, then encoded using a choice of formats, and finally mapped into an arbitrary set of pins for digital output. We are hopeful that this degree of flexibility will allow easy hookup to virtually any STALO synthesizer that one might encounter.

```
PinMap Table (Type 31 for GND, 30 for +5)
Pin01:00       Pin02:01       Pin03:02       Pin04:03       Pin05:04
Pin06:05       Pin07:06       Pin08:07       Pin09:08       Pin10:09
Pin11:10       Pin12:11       Pin13:12       Pin14:13       Pin15:14
Pin16:15       Pin17:16       Pin18:17       Pin19:18       Pin20:19
Pin21:20       Pin22:21       Pin23:22       Pin24:23       Pin25:24
FAULT status pin (0:None): 0, ActLow: NO
```

These questions only appear when the "PinMap" uplink protocol has been selected. The table assigns a bit from the encoded numeric word to each of the 25 pins of the RVP8/DAFC module. For example, the default table shown above simply assigns the low 25 bits of the encoded bit pattern to pins 1–25 in that order. You may also pull a pin high or low by assigning it to +5 or GND. Note that such assignments produce a logic–high or logic–low signal level, not an actual power or ground connection. The latter must be done with actual physical wires.

One of the RVP8/DAFC pins can optionally be selected as a Fault Status indicator. You may choose which pin to use for this purpose, as well as the polarity of the incoming signal level. Note that the standard RVP8/DAFC module only supports the selection of pins 1, 3, 4, 13, 14, and 25 as inputs. This setup question allows you to choose any pin, however, because it does not know what kind of hardware may be listening on the uplink and what its constraints might be.

```
Burst frequency increases with increasing AFC voltage: NO
```

If the frequency of the transmit burst increases when the AFC control voltage increases, then answer this question "Yes"; otherwise answer "No". When this question is answered correctly, a numerical increase in the AFC drive (D–Units) will result in an increase in the estimated burst frequency. If the AFC loop is completely unstable, try reversing this parameter.

```
Mirror AFC voltage on– 0:None, 1:I, 2:Q : 0
```

AFC/MFC can be mirrored on a backpanel output of the main chassis using this question. When either "I" or "Q" is selected, the AFC/MFC voltage will be present on the corresponding BNC output, and the other output will be used for scope plotting. This configuration would be useful, for example, in a dual–receiver magnetron system that needs a phase locked acquisition clock in the RVP8/IFD, but also needs an AFC tuning voltage to control the transmit frequency. When "None" is selected, scope plotting will revert to its normal "Q" output.

The voltage range of the "I" and "Q" outputs is approximately ±1 Volt, and is not adjustable. When AFC/MFC is mirrored on these lines, you will probably need to add an external Op–Amp circuit to adjust the voltage span and offset to match your RF components. We also recommend that you add significant low–pass filtering (cutoff at 3Hz) to remove any power line noise or crosstalk that may be originating within the RVP8 chassis.

```
Enable Burst Pulse Tracking: YES
```

This question enables the Burst Pulse Tracking algorithm that is described in 6.1.4 Burst Pulse Tracking on page 211. Remarkably, for such an intricate new feature, there are no additional parameters to configure. The characteristic settling times for the burst are already defined elsewhere in this menu, and the tracking algorithm uses dynamic thresholds to control the feedback.

```
Enable Time/Freq hunt for missing burst: No

Number of frequency intervals to search: 5
```

```
Settling time for each frequency hop: 0.25 sec

Automatically hunt immediately after being reset: YES

Repeat auto hunt every: 60.00 sec
```

These questions configure the process of hunting for a missing burst pulse. The trigger timing interval that is checked during Hunt Mode is always the maximum ±20µsec; hence no further setup questions are needed to define the hunting process in time. The hunt in frequency is a different matter. The overall frequency range will always be the full -100% to +100% AFC span; but the number of subintervals to check must be specified, along with the STALO settling time after making each AFC change. With the default values shown, AFC levels of -66%, -33%, 0%, +33%, and +66% will be tried, with a one–quarter second wait time before checking for a valid burst at each AFC setting.

You should choose the number of AFC intervals so that the hunt procedure can deduce an initial AFC level that is within a few megaHertz of the correct value. The normal AFC loop will then take over from there to keep the radar in tune. For example, if your radar drifts considerably in frequency so that the AFC range had to be as large as 35MHz, then choosing fifteen subintervals might be a good choice. The hunt procedure would then be able to get within 2.3MHz of the correct AFC level. The settling time can usually be fairly short, unless you have a STALO that wobbles for a while after making a frequency change. Note that hunting in frequency is not allowed for Motor/Integrator AFC loops, and the two AFC questions will be suppressed in that case.

The RVP8 can optionally begin hunting for a missing burst pulse immediately after being reset, but before any activity has been detected from the host computer. This might be useful in systems that both drift a lot and generally have their transmitter *On*. However, this option is really included just as a work around; the correct way for a burst pulse hunt to occur is via an explicit request from the host computer which "knows" when the pulse really should be present. Blindly hunting in the absence of that knowledge can not be done because there are many reasons why the burst pulse may legitimately be missing, for example, during a radar calibration.

The automatic hunt for the burst pulse will always run at least once whenever the feature is enabled. The automatic hunting ceases, however, as soon as any activity is detected from the host computer. Only use this feature on radars with a serious drift problem in their burst pulse timing.

```
Simulate burst pulse samples: NO
```

The RVP8 can simulate a one microsecond envelope of burst samples. This is useful only as a testing and teaching aid, and should never be used in an operational system.

A two–tone simulation will be produced when the RVP8 is setup in dual–receiver mode. The pulse will be the sum of two transmit pulses at the primary and secondary intermediate frequencies. To make the simulation more realistic, the two signal strengths are unequal; the primary pulse is 3dB stronger than the secondary pulse.

```
Frequency span of simulated burst: 27.00 MHz to 32.00 MHz
```

The simulated burst responds to AFC just as a real radar would. The frequency span from minimum AFC to maximum AFC is given here.

4.2.8 AFC Motor/Integrator Option

The question *"AFC Servo– 0:DC Coupled, 1:Motor/Integrator"* selects whether the AFC loop runs in the normal manner (direct control over frequency), or with an external Motor/Integrator type of actuator. The question *"AFC minimum slew request:..."* provides additional control when interfacing to mechanical actuators whose starting and sustaining friction needs to be overcome.

The DC–Coupled AFC loop questions (changes shown in bold) are:

AFC Servo– 0:DC Coupled, 1:Motor/Integrator : 0

```
Wait time before applying AFC: 10.0 sec

AFC hysteresis– Inner: 5.0 KHz, Outer: 15.0 KHz

AFC outer tolerance during data processing: 50.0 KHz
```

AFC feedback slope: 0.0100 D–Units/sec / KHz

AFC minimum slew rate: 0.0000 D–Units/sec

AFC maximum slew rate: 0.5000 D–Units/sec

and the Motor/Integrator loop questions are:

AFC Servo– 0:DC Coupled, 1:Motor/Integrator : 1

```
Wait time before applying AFC: 10.0 sec

AFC hysteresis– Inner: 5.0 KHz, Outer: 15.0 KHz
```

```
AFC outer tolerance during data processing: 50.0 KHz
```

AFC feedback slope: 1.0000 D–Units / KHz

AFC minimum slew request: 15.0000 D–Units

AFC maximum slew request: 90.0000 D–Units

Notice that the physical units for the feedback slope and slew rate limits are different in the two cases. In the DC–Coupled case the AFC output voltage controls the frequency directly, so the units for the feedback and slew parameters use *D–Units/Second*. In the Motor/Integrator case, the AFC output determines the rate of change of frequency; hence *D–Units* are used directly.

The above example illustrates typical values that might be used with a Motor/Integrator servo loop. The feedback slope of 1.0 *D–Units/KHz* means that a frequency error of 100KHz would produce the full–scale (100 D–*Units*) AFC output. But this is modified by the minimum and maximum slew requests as follows:

- A zero *D–Unit* output will always be produced whenever AFC is locked.
- When AFC is tracking, the output drive will always be at least ±15 D–*Units*. This minimum non–zero drive should be set to the sustaining drive level of the motor actuator, that is, the minimum drive that actually keeps the motor turning.
- When AFC is tracking, the output drive will never exceed ±90 D–*Units*. This parameter can be used to limit the maximum motor speed, even when the frequency error is very large.

The AFC Motor/Integrator feedback loop works properly even if the motor has become stuck in a "cold start", that is, after the radar has been turned off for a period of time. The mechanical starting friction can sometimes be larger than normal, and additional motor drive is required to break out of the stuck condition. But once the motor begins to turn at all, then the normal AFC parameters (minimum slew, maximum slew, feedback slope) all resume working properly. The algorithm operates as follows:

- Whenever AFC correction is being applied, the RVP8 calculates how long it would take to reach the desired IF frequency at the present rate of change. For example, if we are 1MHz away from the desired IF frequency, and the measured rate of change of the IF burst frequency is 20KHz/sec, then it will be 50 seconds until the loop reaches equilibrium.
- Whenever the AFC loop is in Track–Mode, but the time to equilibrium is greater than two minutes, then the "Minimum Slew"

parameter will be slowly increased. The idea is to gradually increase the starting motor drive whenever it appears that the IF frequency is not actually converging toward the correct value, that is, the motor is stuck.

- As soon as the frequency is observed to begin changing, such that the desired IF would be reached in less than two minutes, then the "Minimum Slew" parameter is immediately put back to its correct setup value. The loop then continues to run properly using its normal setup values.

Manual Frequency Control (MFC) operates unchanged in both of the AFC servo modes. Whenever MFC is enabled in the **Ps** command, it always has the effect of directly controlling the output voltage of the AFC D/A converter. The MFC mode can be useful when testing the motor response under different drive levels, and when determining the correct value for the minimum slew request.

4.2.9 M+ — Debug Options

A collection of debugging options has been added to the RVP8 to help users with the development and debugging of their applications code. For the most part, these options should remain disabled during normal radar operation. These questions are included so that the RVP8 can be placed into unusual, and perhaps occasionally useful, operating states.

```
Noise level for simulated data: -50.0 dB
```

This is the noise level that is assumed when simulated "I" and "Q" data are injected into the RVP8 via the LSIMUL command. The noise level is measured relative to the power of a full–scale complex (I,Q) sinusoid, and matches the levels shown on the slide pots of the ASCOPE digital signal simulator.

Limits: –100dB to 0dB

```
Simulate output rays: NO
```

Answering "YES" to this question causes the RVP8 to output bands of simulated data. The bands can occupy a selectable range interval, and span a selectable interval of data values.

```
Start bin:0, Width:10 bins, Bands:16
```

This question is only asked if we are simulating output rays. The Start Bin chooses the bin number (origin zero) where the simulated bands will begin. The width of each band (in bins), and the total number of bands are also

selected. The upper limit for all parameters is the maximum bin count for the RVP8.

Limits: Start: 0–Max, Width: 1–Max, Bands: 1–Max

```
Start data value:0, Increment:16
```

This question is only asked if we are simulating output rays. The data value that will be assigned to the first simulated band, and the data increment from one band to the next, are selected. The permissible values are from 0 to 65535, that is, the full unsigned 16–bit integer range. This full range is useful when simulating 16–bit output data; for the more typical 8–bit output formats, only the low byte of the start and increment are significant.

Limits: 0 to 65535

```
Real Time TTY Monitor: NO
```

The Real Time TTY Monitor is a stream of characters that are continuously sent to a serial output of the RVP8, and which monitors selectable internal variables. When this live monitor is enabled, status lines will be printed continuously. You may choose the update rate, and which parameters are to be printed, using the questions that optionally follow.

```
Pathname of TTY/FIFO: '/dev/ttyS0'

Serial data rate:

Update rate: 2.00 lines/sec

Show burst frequency: NO

Show burst pulse power: NO

Show AFC information: NO

Show pulse width: NO

Show PRF: NO

Show LOG noise: NO

Show Polarization: NO

Show IFD and link info: NO

Show burst timing slew: NO
```

Most of the data fields are printed in self–explanatory scientific units. The PRF is in Hertz, but is printed without an "Hz" suffix. For historical reasons, the LOG Noise is printed in old fashioned 8–bit A/D units, taken as the upper eight bits of the 12–bit long LOG format described in 7.7 Initiate Processing (PROC) on page 295. An 8–bit *NSE* value may be converted to an absolute dBm level using:

$$dBm = p_{Max} + 16s\ (NSE - 224)$$

Where *s* is the LOG Slope (nominally equal to 0.03), and P_{Max} is +4.5dBm for the 12–bit IFD and +6.0dBm for the 14–bit IFD.

4.2.10 Mz — Transmissions and Modulations

These questions are used to configure the 8–Bit phase modulation codes that may be used to control the phase of a coherent transmitter. The RVP8/Tx will output a pseudo–random sequence of phase codes that are chosen from a specified set of available codes, that is, all 8–bit patterns that are valid for the phase modulation hardware. The random sequence is output only when the RVP8 is in one of its random phase processing modes (time series or parameter). At all other times, a fixed "idle" phase code pattern is output. See also 4.2.1 Mc — Top Level Configuration on page 120 and 4.2.5 Mt<n>— Triggers for Pulsewidth #n on page 133 where related phase control questions are found.

```
8-Bit code to output when idle: 0x00
```

This is the bit pattern to be output whenever the RVP8 is not in a random phase processing mode. Note that this "idle" code does not have to be one of the "active"codes that are enabled below.

```
Selection of Valid 8–Bit States

Selection of Valid 8-Bit States
00-0F: Y - - - - - - - - - - - - - - - -
10-1F: - - - - - - - - - - - - - - - - -
20-2F: - - - - - - - - - - - - - - - - -
30-3F: - - - - - - - - - - - - - - - - -
40-4F: - - - - - - - - - - - - - - - - -
50-5F: - - - - - - - - - - - - - - - - -
60-6F: - - - - - - - - - - - - - - - - -
70-7F: - - - - - - - - - - - - - - - - -
80-8F: - - - - - - - - - - - - - - - - -
90-9F: - - - - - - - - - - - - - - - - -
A0-AF: - - - - - - - - - - - - - - - - -
B0-BF: - - - - - - - - - - - - - - - - -
C0-CF: - - - - - - - - - - - - - - - - -
D0-DF: - - - - - - - - - - - - - - - - -
E0-EF: - - - - - - - - - - - - - - - - -
F0-FF: - - - - - - - - - - - - - - - - -
```

This set of questions defines the subset of active 8–bit codes that are valid states for the transmit phase modulator. Answer each line with a sequence of Y's or N's to indicate whether the corresponding 8–bit code is enabled. Only the codes that appear with a "Y" will be used by the RVP8; the "–" indicates an unused code. The "–' character was used instead of "N" so that the visual contrast of the printed table would be improved.

As an example, if your klystron transmitter has an octant phase modulator that is controlled by three digital lines, you might enable phase codes zero through seven, and then cable the modulator to the low three bits of the 8–bit code. The upper five bits would not need to be used in this case.

4.3 Advanced Options

4.3.1 * — Sample current noise levels

The "*" samples current noise levels from the receiver and then subtracts that noise from subsequent measurements. More information is provided in the Sample Noise Level (SNOISE) 7.6 Sample Noise Level (SNOISE) on page 292.

4.3.2 @ — Display/Change current Major Mode

This command provides developers with a simple way of switching modes enabling on–the–fly testing of code. The top level RVP8 operating modes are described in the documentation of the SOPRM command word #9. This question allows you to use the mode that has been selected by that command, or to force the use of a particular mode.

4.3.3 ~ — Burst–In / IF–In Swap Command (Rev.D IFD)

The "~" command swaps the Burst and IF inputs at the IFD. Requests to toggle the state are made from the top level as follows:

```
RVP8> ~

IFD Burst/IF Inputs are: SWAPPED

RVP8> ~

IFD Burst/IF Inputs are: NORMAL
```

The selection remains in effect for the duration of the setup session, but then returns to NORMAL upon exiting the TTY monitor. The "~" command is very handy because it allows the **Pb, Pr,** and **Ps** plotting commands to easily run with one input or the other. Here are two examples of how this might be useful.

- When checking the range alignment on a Klystron system, the **Pb** plot can not be used in the usual way to center the Tx burst because a continuous–wave COHO (rather than a burst pulse) is typically used as the phase reference in these systems. However, if you swap the Burst and IF inputs, you can then use the **Pb** command to view and center the received leakage of the Tx pulse, and thus locate range zero.
- When setting up the AFC loop, you can use your RF signal generator to simulate the transmitter's frequency, and then run the loop with swapped RVP8/IFD inputs. The AFC servo will then hunt and follow the siggen frequency supplied via the receiver. You can then make step changes in that frequency to verify that the loop responds properly.

Note that the same input swapping function is also available via the RVP8/IFD toggle switches. However, those switches may be located far away from the operator's terminal; hence, the command interface is still a valuable addition. The "~" command can only be used with a Rev.D or later IFD; the command is unimplemented, and will not show up in the "Help" list, when earlier receivers are connected.

CHAPTER 5
PLOT-ASSISTED SETUPS

The IFD receiver module replaces virtually all of the IF components in a traditional analog receiver. The alignment procedures for those analog components are usually very tedious, and require continued maintenance even after they are first performed. Subtle drifts in component specifications often go unnoticed until they become so severe that the radars data are compromised.

The RVP8 makes a big improvement over this by providing an interactive graphical alignment procedure for burst pulse detection, Tx/Rx phase locking, and calibration of the AFC feedback loop. You may view the actual samples of the burst pulse and receiver waveform, examine their frequency content, design an appropriate matched filter, and observe live operation of the AFC. It is a simple matter to check the spectral purity of the transmitter on a regular basis, and to discover the presence of any unwanted noise or harmonics. Moreover, the RVP8 is able to track and modify the initial settings so that proper operation is maintained even with changes in temperature and aging of the microwave components.

The Plot–Assisted Setups are accessed using the various **P** commands within the normal TTY setup interface. These commands are described later in this chapter. The RVP8 supports opcodes that allow the host computer to monitor the data being plotted. The **dspx** utility can display these plots directly on the workstation screen, and thus, can carry out the graphical checkup and alignment procedures remotely via a network.

5.1 P+ — Plot Test Pattern

The RVP8 can produce a simple test pattern to verify that the display software is working properly. From the TTY monitor enter the **P+** command. This will print the message **Plotting Test Pattern...** on the TTY and then produce the plot shown in Figure 20 on page 160

This display is actually an overlay of six different strokes: 1) bottom line, 2) middle line, 3) top line, 4) line sloping up, 5) line sloping down, and 6) the sine wave pattern. The later changes phase with each plot so that, with a little imagination, it appears to be radiating from the left side of the display.

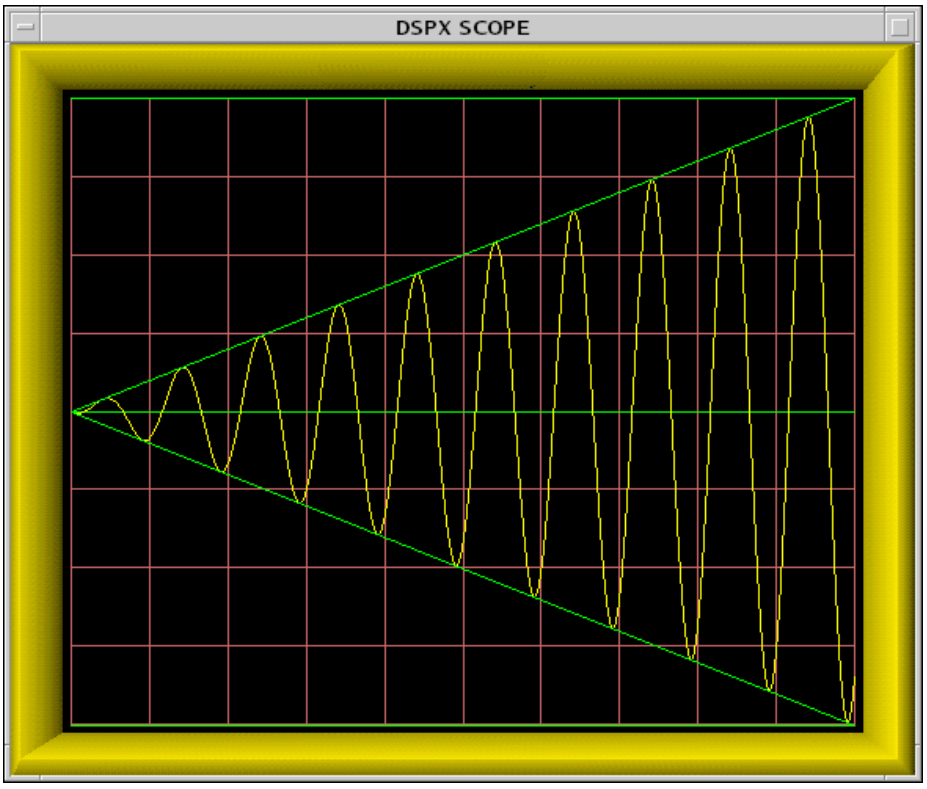


Figure 20 The Test Pattern Display

When you are satisfied that the plot is being drawn correctly, type **Q** or hit ESC to return to the TTY monitor.

5.2 General Conventions Within the Plot Commands

The **Pb**, **Ps**, and **Pr** commands all have a similar structure to their TTY user interface. Each command begins by printing a list of subcommands that are valid in that context. These subcommands are single keystrokes that are executed immediately by the RVP8 as they are typed. The ENTER key is not required. The available subcommands are different for each plot command; but, as much as possible, each key has a similar meaning across all commands.

The working and measured parameters for each plot command are printed on the TTY as two lines of information following the subcommand list. The first line contains settings that only change when a subcommand is issued; but the second line is live and reflects the current status of the burst input, the IF input, or the AFC output. The first line is printed just once, but the second line is continually overprinted on top of itself. This makes it appear as a live status line whose values always remain up to date. The **Pb**, **Ps**, and **Pr** commands will report "No Trigger" on the TTY status line whenever the external trigger is expected but missing.

The TTY screen will scroll upward each time a new subcommand is executed, so that a history of information lines and command activity can be seen on the screen. You may also use the Carriage–Return key to scroll the display up at any time. If the initial list of subcommands disappears off the top, you may type **?** to force a reprint. To exit the plot command entirely and return to the TTY main menu type **Q** or **ESC**. These basic "help" and "exit" keystrokes apply everywhere within the RVP8 setup menus. To save space and minimize clutter on the TTY screen, they are not shown in the itemized list of subcommands.

Most commands have a lowercase and an uppercase version. If a lowercase command does something, then its uppercase version does the same thing but even more so (or in reverse). For example, if the **w** subcommand widens something by a little bit, then "W" would widen it a lot. This simple convention reduces the number of different subcommand keys that are needed, and makes the interface easier to memorize.

The graphical display and TTY status lines are continually updated with fresh data several times per second. Occasionally it is useful to freeze a plot so that it can be studied in more detail, or compared with earlier versions. To accomplish this, every plotting command supports a "Single Step" mode that is accessed by typing the "." (period) key. This key causes the display and TTY status lines to freeze in their present state, and the message **"Paused..."** will be printed. Subsequently, typing another . will single step to the next data update, but the plot and printout will still remain

frozen. Typing **Q** or **ESC** will exit the plot command entirely (as they normally do). All other keys return the plot command to its normal live updating, but the key is otherwise discarded (that is, subcommand keys are not executed while exiting from single step mode).

All of the plot commands support subcommands whose only purpose is to alter the appearance of the display, for example, zoom, stretch, etc. These subcommands make no changes to the actual working RVP8 calibrations. However, the display settings are stored in nonvolatile RAM just like all of the other setup parameters. This means that all previous display settings will be restored whenever you restart each plot command. This is very convenient when alternating among the various plots.

The **Pb**, **Ps**, and **Pr** commands are intended to be used together for the combined purpose of configuring the RVP8s digital front end. You may, of course, run any of the commands at any time; but the following procedure may be used as a guideline for first time setups. The full procedure must be repeated for each individual pulsewidth that the radar supports.

1. Use **Mb** to set the systems intermediate frequency (See 4.2.7 Mb — Burst Pulse and AFC on page 142).
2. Use **Mt** to choose the PRF and pulsewidth (See 4.2.4 Mt — General Trigger Setups on page 130). Also, choose the range resolution now, as it may constrain the design of the matched filter later.
3. Use **Mt0, Mt1,** etc., to set the relative timing of all RVP8 triggers that are used by the the radar. Do not worry about the absolute values of the trigger start times. Just setup their polarity and width, and their start times relative to each other (See 4.2.5 Mt<n>— Triggers for Pulsewidth #n on page 133). Make an initial guess of FIR filter length as 1.5 times the pulsewidth.
4. Use **Pb** to slew the start times of all triggers so that the burst pulse is properly sampled (See 5.3 Pb — Plot Burst Pulse Timing on page 163). Refine the impulse response length if necessary so that all samples easily fit within the display window.
5. Use **Ps** to design the matched FIR filter (See 5.4 Ps — Plot Burst Spectra and AFC on page 168). Further refine the impulse response length and passband width to achieve a filter that matches the spectral width of the burst, and that has strong attenuation at DC. If the FIR length is changed, return to **Pb** to verify that the burst is still being sampled properly.
6. Continue using **Ps** and **Mb** to tune up the AFC feedback loop. The settings that work for one pulsewidth should also work for all others.
7. Use **Pr** to verify that targets are being detected with good sensitivity (See 5.5 Pr — Plot Receiver Waveforms on page 183).

Sometimes it is useful to run the **Pb** and **Ps** commands with samples from the IF–Input of the IFD, rather than from the Burst–Input. Likewise, it is sometimes useful to view the **Pr** plots on samples of Burst data. The top–level "~" command (See 4.3.3 ~ — Burst–In / IF–In Swap Command (Rev.D IFD) on page 157) allows you to do thiseasily.

5.3 Pb — Plot Burst Pulse Timing

For magnetron radars the RVP8 relies on samples of the transmit pulse to lock the phase of its synthesized "I" and "Q" data, and to run the AFC feedback loop. The **Pb** command is used to adjust the trigger timing and A/D sampling window so that the burst pulse is correctly measured.

5.3.1 Interpreting the Burst Timing Plot

The display plot will ultimately resemble Figure 21 on page 163, which shows a successful capture of the transmitter's burst pulse. The horizontal axis of the display represents time, and the overall time span from the left edge to the right edge is listed as "PlotSpan" on the TTY.

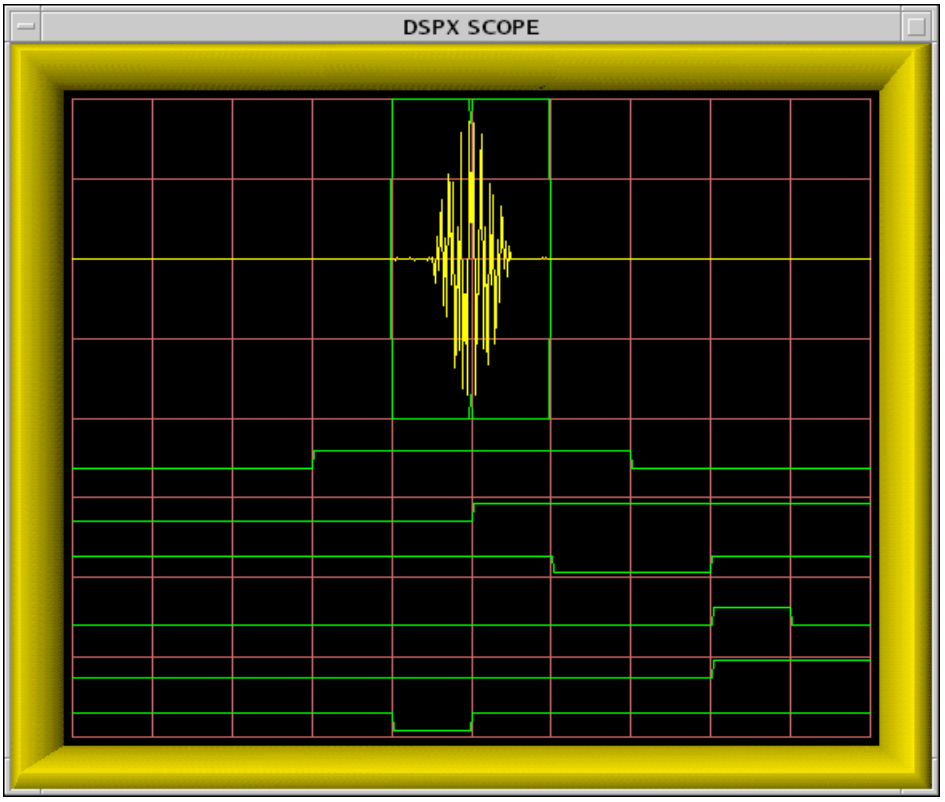


Figure 21 Successful Capture of the Transmit Burst

The upper portion of the plot shows the sampling window wherein the burst pulse is measured. The duration of this window is determined by the impulse response length of the matched FIR filter. This is because the same FIR coefficients that compute "I" and "Q" are also used to compute the reference phase vectors for the burst pulses. The A/D samples of the RVP8/IFD's burst input are plotted (somewhat brighter) within the sample window.

The RVP8 computes the power–weighted center–of–mass (COM) of the burst pulse envelope. This allows the processor to determine the location of the "middle" of the transmitted pulse within the burst analysis window. The **Pb** plot displays small tick marks on the top and bottom of the burst sample window to indicate the location of the COM. These markers are only displayed when valid burst power is detected. A second "error bar" is drawn surrounding the tick mark to indicate the uncertainty of the mark itself. This error interval is used by the burst pulse tracking algorithm to decide when a timing change can be made with confidence.

It is possible to independently choose a subinterval of burst pulse samples that are used by the AFC frequency estimator. Thus, the AFC feedback loop is not constrained to use the same set of samples that are chosen for the FIR filter window. The FIR window typically is longer than the actual transmitted pulse, and thus, the samples contributing to the frequency estimate will include the leading and trailing edges of the pulse. These edges tend to have severe chirps and sidebands, compared to the more pure center portion of the pulse. The AFC frequency estimate (which is power weighted) could be mislead by these edges and might not tune to the optimum center frequency if they were included.

The lower portion of the plot shows the six triggers that are output by the RVP8. Trigger #0 is at the top, and Trigger #5 is on the bottom. They are drawn in their correct polarity and timing relative to each other, and relative to the burst sample window. Note that the sample window is always drawn in the center of the overall time span. Thus, depending on the PlotSpan and location of the six trigger's edges, triggers that do not vary within the plotted time span will appear simply as flat lines.

The RVP8 defines "Range Zero" to occur at the center of the burst sample window. This also defines the zero reference point for the starting times of the six programmable triggers. For example, a trigger whose starting time is zero will be plotted with its leading edge in the exact horizontal center of the display. Knowing this convention makes the absolute value of the trigger start times more meaningful.

5.3.2 Available Subcommands Within Pb

The list of subcommands is printed on the TTY:

```
Available Subcommands within 'Pb':

I/i         Impulse response length Up/Dn
A/a & S/s   Aperture & Start of AFC window
L/l & R/r   Shift triggers & RVP8/Tx waveform left/right,
T/t         Plot time span Up/Dn
Z/z         Amplitude zoom
B/b         BP Tracking On/Off (temporary)
+           Hunt for missing burst
-           Single Step
```

These subcommands change the matched filter's impulse response length, shift the radar triggers, and alter the format of the display.

I/i The **I** command increments or decrements the length of the matched filter's impulse response. Each keystroke raises or lowers the FIR length by one tap.

A/a & S/s These commands raise/lower the aperture/start of the subwindow of burst pulse samples for AFC. If you never use these commands, then the full FIR window will be used; however, shortening the AFC interval will result in two sample windows being drawn on the plot. The smaller AFC window should be positioned into the center portion of the transmitted pulse, where the burst amplitude and frequency are fairly stable.

L/l & R/r These two commands shift the entire group of six RVP8 triggers left or right (earlier or later in time). The lowercase commands shift in 0.025 µsec steps, and the uppercase commands shift in 1.000 µsec steps (approximately). The reason for shifting all six triggers at once is that the relative timing among the triggers remains preserved. However, the absolute timing (relative to range zero) will vary, and this will cause the burst pulse A/D samples to move within the sample window.

T/t The **T** command increments or decrements the overall time span of the plot. The available spans are 2, 5, 10, 20, 50, 100, 200, 500, 1000, 2000 and 5000 microseconds. The value is reported on the TTY as "PlotSpan".

Z/z The **Z** command zooms the amplitude of the burst pulse samples so that they can be seen more easily. The value is reported on the TTY as "Zoom".

B/b These keys temporarily disable or re-enable the Burst Pulse Tracker. The tracker must be disabled in order for the L/R keys to be used to shift the nominal trigger timing. The "b" key disables tracking and sets the trigger slew to zero; the "B" key re-enables tracking starting from that zero value.

+ The + subcommand initiates a hunt for the burst pulse. Progress messages are printed as successive AFC values are tried, and the trigger slew and AFC level are set according to where the pulse was found. If no burst pulse can be found, then the previous trigger slew and AFC are not changed.

5.3.3 TTY Information Lines Within Pb

The TTY information lines will resemble:

```
Zoom:x2, PlotSpan:5 usec, FIR:1.36 usec (49 Taps)

Freq:27.817 MHz, Pwr:-53.9 dBm, DC:0.14%, Trig#1:-5.00,
BPT:0.00
```

Zoom Indicates the magnification (in amplitude) of the plotted samples. A zoom level of "x1" means that a full scale A/D waveform exactly fills the height of the sample window. Generally, the signal strength of the burst pulse will not be quite this high. Thus, use larger zoom levels to see the weaker samples more clearly. You may zoom in powers of two up to x128.

PlotSpan Indicates the overall time span in microseconds of the complete scope display, from left edge to right edge.

FIR Indicates the length of the impulse response of the matched FIR filter, and hence, the duration of the burst pulse sample window. The length is reported both as a number of taps, and as a time duration in microseconds.

Freq Indicates the mean frequency of the burst, derived from a 4^{th} order correlation model. The control parameters for this model are set using the **Mb** command (4.2.7 Mb — Burst Pulse and AFC on page 142).

Pwr Indicates the mean power within the full window of burst samples. DC offsets in the A/D converter do not affect the computation of the power, that is, the value shown truly represents the waveform's (Signal+Noise) energy.

DC Indicates the nominal DC offset of the burst pulse A/D converter. This is of interest only as a check on the integrity of the front end analog components. The value should be in the range ±2.0%.

Trig#1 Indicates the starting time of the first (of six) RVP8 trigger outputs. This number will vary as the **L** and **R** subcommands cause the triggers to slew left and right. Note that if the radar transmitter is directly fired by an external pretrigger, then the pretrigger delay (in the form "**PreDly:6.87**") will be printed instead.

BPT This shows the present value of timing slew (measured in microseconds) being applied to track the burst. The slew will be zero initially when the RVP8 is first powered up, meaning that the normal trigger start times are all being used.

5.3.4 Recommended Adjustment Procedures

The burst pulse timing must be calibrated separately for each individual pulsewidth. Repeat the following procedure for each pulsewidth that you plan to use. Each iteration is independent.

It is first necessary to setup the proper relative timing for all RVP8 triggers that are being used. The six trigger output lines are completely interchangeable, and each may be assigned to any function within the radar system. For example, Trigger #0 might be the transmitter pretrigger, Triggers #2 and #3 might synchronize external displays, and Triggers #1, #4, and #5 might be unused.

Choose an initial impulse response length of 1.5 times the transmit pulsewidth. This length will be refined later when the matched filter is designed (See 5.4 Ps — Plot Burst Spectra and AFC on page 168). Adjust the plot time span to view a small region around the sample window, and set the initial amplitude zoom to x16. This assures that the plotted waveform will still be noticeable even if the burst signal strength is very weak.

Verify that the transmitter is radiating, and observe the burst pulse samples on the display. Use the **L** and **R** commands to shift the timing of all six triggers relative to range zero. This has the effect of moving the burst sampling window relative to the transmitted pulse. Depending on whether the triggers are set properly, you may at first see nothing more than a flat line of misplaced A/D samples. However, after a few moments of hunting, the burst pulse should appear on the display screen. Fine tune the triggers

so that the burst envelope is centered in the window, and adjust the amplitude zoom for a comfortable size display.

The clean center portion of the burst pulse should then be isolated to a narrower subwindow of the overall FIR interval. Use the **A** and **S** commands to change the aperture and start of the narrowed region from which the AFC frequency estimators data will be derived.

Check that the burst pulse signal strength is reasonably matched to the input span of the RVP8/IFD's A/D converter. The maximum analog signal level is +4dBm. Exceeding this level produces distorted samples that would seriously degrade the algorithms for phase locking and AFC. However, if the signal is too weak, then the upper bits of the A/D converter are wasted and noise is unnecessarily introduced. We recommend a peak signal level between –6dBm and +1dBm, that is, a signal that might be viewed at x2 or x4 zoom. Take note of the burst energy level reported on the TTY; it will help decide the minimum energy threshold for a valid burst pulse, which is needed in 4.2.7 Mb — Burst Pulse and AFC on page 142.

5.4 Ps — Plot Burst Spectra and AFC

Once the transmit burst pulse has been captured the next step is to analyze its frequency content and to design a bandpass filter that is matched to the pulse. In a traditional analog receiver the matched filters use discrete components that can not be adjusted, and the transmit spectrum can not be viewed unless a spectrum analyzer is on hand. The RVP8 eliminates all of these restrictions via its **Ps** command, which plots the burst spectrum, designs the bandpass filter, plots its frequency response, and also helps with alignment of the AFC.

5.4.1 Interpreting the Burst Spectra Plots

An example of a plot from the **Ps** command is shown in Figure 22 on page 169. The display screen is divided into two independent areas. The major portion (the lower seven eighths) is devoted to power spectrum plots of the burst pulse and/or the matched filter response. The top portion (single line) serves as a visual indicator of the present AFC level.

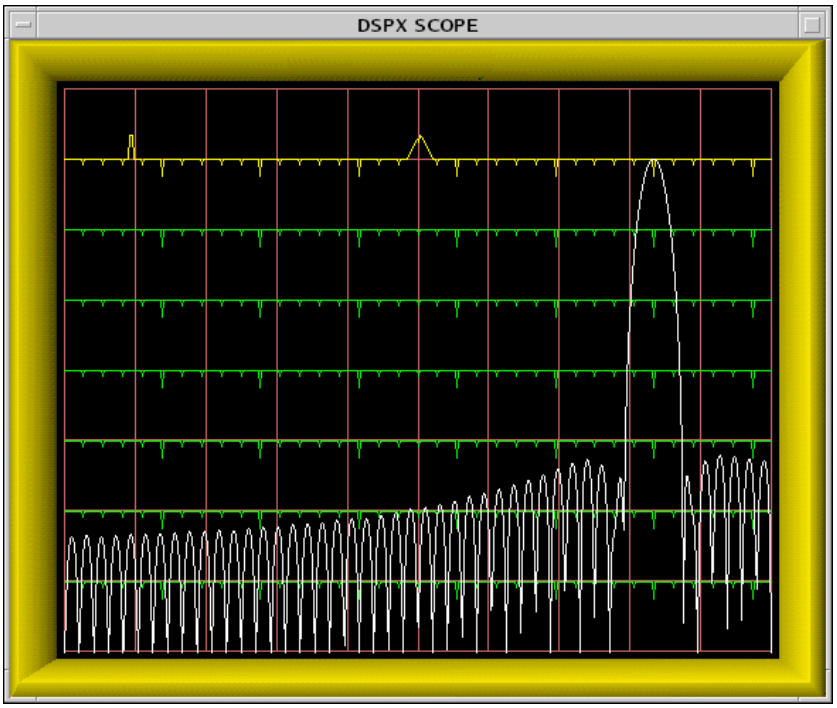


Figure 22 Example of a Filter With Excellent DC Rejection

The horizontal axis of the spectrum plot represents frequency. The overall span from the left edge to the right edge is 36MHz for the RVP8 CAT–5E IFD, and 18MHz when the legacy RVP7 FibreOptic IFD is used. The remainder of this chapter will refer only to the newer RVP8 unit.

The exact endpoints of the plot depend on which alias band the radar's intermediate frequency falls in. For example, a 30MHz IF would imply a horizontal axis range of DC to 36MHz, whereas a 60MHz IF would make the range 36MHz to 72MHz. The frequency span is printed on the TTY when the command is first entered. Since the left edge of the spectral plot always represents an integer multiple of 36MHz, either the left side or the right side will always be a multiple of 36MHz. This is important to remember when designing the matched filter, since fixed DC offsets in the A/D converters will appear aliased at these 72MHz multiples.

The vertical axis of the spectrum plot is logarithmic and is marked with faint horizontal lines in 10–dB increments. An overall dynamic range of 70

dB can be viewed at once. The horizontal lines also contain major and minor tick marks to help calibrate the frequency axis. Major marks are small downward triangles that represent integer multiples of 5MHz; minor marks are in between and represent 1–MHz steps. The power spectrum example in Figure 22 on page 169 is from asystem with an intermediate frequency of 30MHz. Thus, the left edge of the plot begins at DC, and the graph is centered on the sixth major tick, that is, 30MHz.

Two types of spectra can be plotted on the screen: 1) the frequency response of the FIR filter, and 2) the frequency content of the burst pulse itself. The burst spectrum is computed by first applying a Hamming window to the raw samples. You may choose to view either plot individually, or both at the same time.

Figure 22 on page 169 is an example of a single filter response plot, whereas Figure 23 on page 180 shows a combined display of both spectra. The combined display makes it easy to compare the filter being designed with the live waveform that it is intended to selectively pass. Note that the filter's frequency response is always drawn with its passband peak touching the top of the plot. The vertical height of the burst spectrum, however, will vary with signal strength but can be adjusted using the **Z** subcommand.

The horizontal line at the top of the plotting area is also marked with an upward pointing major and minor tick. These indicate the present value of the burst pulse frequency estimator. The major tick is a triangle whose position along the horizontal axis corresponds directly to the estimated frequency. It should always be positioned directly over the main lobe of spectral power. The minor tick gives finer scale resolution by indicating the fractional part of each 1–MHz multiple.

It is helpful to read the minor tick relative to the ten horizontal division lines that are present on most scopes. Motion of the minor tick is apparent even with very small changes in burst pulse frequency; a change of just 5 KHz can easily be seen. This means that you can observe the frequency drift of the magnetron in great detail, and also watch the AFC's behavior in real time.

The horizontal line at the very top of the display (above the spectra plot) serves to indicate the present value of the AFC control voltage. The line contains an upward pointing major and minor tick, similar to the ones used to represent the burst frequency estimate on the line below. However, the horizontal axis now represents voltage rather than frequency, and the overall span is the complete range of the AFC's digital–to–analog converter. The major tick will move from the left edge to the right edge as the AFC varies from its minimum to maximum value. The minor tick will traverse the screen at ten times this rate.

5.4.2 Available Subcommands Within Ps

The list of subcommands is printed on the TTY:

```
Frequency span of the plot is 36.0 MHz to 72.0 MHz.

Available Subcommands within 'Ps':

I/i                Impulse response length Up/Dn
N/n & W/w          Filter bandwidth Narrower/Wider
U/u & D/d          MFC Up/Down (On/Off '=' , Test '|')
A/a & S/s          Aperture & Start of AFC window
#                  Print filter coefficients
$                  Search for an optimal filter
V/v                Number of spectra averaged
Z/z                Amplitude zoom
<space>            Alternate Plots
%                  Toggle between dual receivers
-                  Single Step
```

These subcommands change the design of the matched FIR filter, assist with calibration of the AFC loop, and alter the format of the display.

I/i The **I** command increments or decrements the length of the matched filter's impulse response. Each keystroke raises or lowers the FIR length by one tap. Often the matched filter's characteristics can be very much improved merely by changing the FIR length by one or two taps. Be sure to experiment with this as you design your filter.

N/n & W/w The **N** and **W** commands change the passband width of the matched filter, making it narrower or wider. The lower case commands make changes in 1KHz steps, and the upper case commands use 100KHz steps. The value is reported on the TTY as "BW". Often a small change in passband width will shift the exact locations of the filter's zeros, and possibly improve the DC rejection.

U/u & D/d The **U** and **D** commands implement the Manual Frequency Control (MFC) override, and allow the RVP8/IFDs AFC output voltage to be manually set to any fixed level. The lower case commands make changes in 0.05 D-Unit steps, and the upper case commands use 1.0 D-Unit steps. The value is reported on the TTY as "AFC".

= MFC mode is toggled on and off using the "=" key. A warning will be printed if the **Ps** command is exited while MFC is enabled, and you will be given a second chance to reenable AFC.

| The AFC test submode is entered by typing the "|" key. The following list of keybindings will be shown, and will remain in effect until the test mode is exited by typing "Q".

```
AFC Test Mode Subcommands
            W         Use WalkingOnes pattern
            P         Toggle Pin/Bit numbering
            09,AO     Toggle AFC Bits 024 (Pins 125)
                      2 2 2 2 2 1 1 1 1 1 1 1 1 1 1
                      4 3 2 1 0 9 8 7 6 5 4 3 2 1 0 9 8 7
                      6 5 4 3 2 1 0

                      O N M L K J I H G F E D C B A 9 8 7
                      6 5 4 3 2 1 0
```

The **Ps** command continues to run normally during the AFC test mode. The customary AFC information will be replaced with a hexadecimal readout of the present 25-bit value. Your live display may look something like:

```
Navg:3, FIR:1.33 usec (48 Taps), BW:1.000 MHz,
DCGain:ZERO
Freq:26.610 MHz, Pwr:64.6 dBm, AFCTest:0000207F (Bits)
```

Initially, a walking-ones bit pattern will be output in lieu of the normal formatted AFC value. This test pattern shifts a single "1" downward through the AFC word, making a transition approximately every 4ms. It is intended to help ring out and test the wiring for digital AFC installations. The walking-ones test is handy as an oscilloscope diagnostic, and you may return to it at any time by typing "W".

Typing any of the characters "0" through "9" or "A" through "O" will enter a new mode in which a static 25-bit digital AFC pattern is controlled directly. Each key toggles its corresponding bit, as summarized in the keybindings printout. Any 25-bit pattern can be made by toggling the appropriate bits (initially all zero) to one. Within any particular pattern, it is also easy to toggle a particular bit On/Off in order to verify its function.

The **P** command lets you decide whether the 25-bit word represents a numeric AFC span that is mapped into pins via the pin-map table in the **Mb** menu; or whether it represents those pins directly. The printed hex test value will be followed either by "(Bits)" or "(Pins)" accordingly. When in "Pins" mode, the "0" key toggles Pin-1, the "1" key toggles Pin-2, etc. When in "Bits" mode, the "0" key toggles whatever pin or pins have been designated to be driven from Bit-0 of the numeric AFC. The "Pins" mode is useful when you are doing the initial electrical tests of the wiring of each pin. After the pin wiring has been verified and the **Mb** mapping table has been created, then the "Bits" mode allows you to test the complete digital AFC interface.

The # command results in a printout of the coefficients of the current FIR filter. The values are scaled by the coefficient with the largest absolute value, so that they all fall within the 1 to +1 range. This detailed information may be used to model the behavior of the filter for point targets that fall in between discrete range bins, for example, as will happen when performing a radar sphere calibration. See 6.1.1 FIR (Matched) Filter on page 203 for the exact definition of these coefficients.

$** The **$ command performs an automatic search for optimal (DC gain of zero) filters in the vicinity of the current one. As an example, suppose that we wanted an optimal filter that was approximately 2.2 µsec long and 650 KHz wide. We would first use the **I/i** and **W/wN/n** subcommands to manually move to that starting point. Typing "**$**" would then print a dialog line in which the search span length and width are chosen. You may keep the indicated values or type in new ones, just as for all RVP8 setup questions. The search begins when the spans are accepted.

The search procedure may require a few seconds to a few minutes, depending on the length and width spans that are being scanned. During this time, a progress message is printed showing the length and width currently under examination. You may type **Q** to abort the search and retain the original filter settings. When the search completes normally, it will print "Done" and replace the old filter settings with the best ones that could be found.

	In dual-receiver mode, the **$** command will search for a filter that minimizes the maximum width and DC offset at both receivers intermediate frequencies. The final filter will be the one that has the best simultaneous performance at both IFs.
V/v	The **V** command increments or decrements the number of burst pulse spectra that are averaged together to create the plot. The count ranges from one (no averaging) to 25, and is reported on the TTY as "Navg".
Z/z	The **Z** command zooms (that is, shifts on a logarithmic scale) in 1.0-dB steps the amplitude of the burst pulse spectra. This is useful when the overall 70dB plot span is not sufficient to hold the full range. Zoom can also be used to line up the burst spectrum with the filter response so that the two can be compared. The zoom level is not printed on the TTY because there is nothing useful that could be done with it.
<space>	The space bar alternates among three choices for the type of spectra that are plotted: 1) FIR frequency response, 2) Burst pulse spectrum, and 3) Both.
%	In dual-receiver mode, the **%** command toggles between each receiver. The printed status line is prefixed with "Rx:Pri" or "Rx:Sec" according to which receiver is selected. Specifically, typing "%" will toggle the plot of the FIR filters frequency response, and the printout of its DCGain. However, the plotted spectrum and printed power levels are always based on the sum of all input signals, and thus do not change with "%".

5.4.3 TTY Information Lines Within Ps

The TTY information lines will resemble:

```
Navg:3, FIR:1.33usec (48 Taps), BW:1.000, MHz, DCGain:ZERO

Freq:30.027 MHz, Pwr:64.2 dBm, Loss:1.2dB, AFC:23.05%
(Manual)
```

Navg	Indicates the number of burst spectra that are averaged together prior to plotting. Larger amounts of averaging increase the ability to see subtle spectral components, but the display will update more slowly.
FIR	Indicates the length of the impulse response of the matched FIR filter. See 5.3.3 TTY Information Lines Within Pb on page 166

BW Indicates the actual 3dB bandwidth of the matched filter. This is the complete width of the passband from the lower frequency edge to upper frequency edge. Note that the filters center frequency is fixed at the radars intermediate frequency, as chosen in the **Mb** setup command.

DC-Gain Indicates the filters response to DC (zero frequency) input. The value is a negative number in decibels, or the word "ZERO" if the filter has a true zero at DC. The filters DC gain should be kept at a minimum so that fixed offsets in the A/D converters will not propagate into the synthesized "I" and "Q" values.

Freq Indicates the mean frequency of the burst. See 5.3.3 TTY Information Lines Within Pb on page 166 on page 166

Pwr Indicates the average power in the full burst sample window. See on page 166

Loss The filter loss is a positive number in deciBels, and is only displayed if the overall burst power exceeds the minimum valid burst threshold set in the Mb command (clearly, it would not be possible to compute the filter loss when the burst waveform is missing). The filter loss is discussed further in 5.4.4 Computation of Filter Loss on page 176.

AFC Indicates the level and status of the AFC voltage at the RVP8/IFD module. The number is the present output level in D-Units ranging from 100 to +100. The shorter "%" symbol is used since percentage units correspond in a natural way to the D-Units.
An additional number in square brackets will be printed to the right of the AFC level to show the encoded bit pattern which corresponds to that level. This will only appear when the RVP8 deduces that a special digital format is being used, i.e., when the backpanel phase-out lines have been configured for AFC, or when any of the following are not true: a) the low and high numeric AFC span is 32768 to +32767, b) the uplink is enabled, c) the uplink format is binary, and d) pinmap protocol is OFF. Binary format is printed in base-10, BCD format is printed in Hex, and 8B4D format is printed with the low 16-bits (four BCD digits) in Hex and the upper bits in base-10.
The AFC mode s shown to the right of the numerical value(s), and can take on the following states.

(Disabled) Indicates that neither AFC nor MFC are enabled. The output voltage remains fixed at 0% (center of its range).

(Manual) Manual Frequency Control (MFC) is overriding AFC. The "U" and "D" commands can be used to slew the voltage up and down.

Whenever any of the following four states appears, it implies that AFC is enabled and that MFC is disabled.

(NoBurst) The energy in the burst is below the minimum energy threshold for a valid pulse (See Page 325). The AFC loop remains idle.

(Wait) The burst pulse has become valid just recently, but the AFC loop is idle until the transmitter stabilizes (See Page 326)

(Track) The burst pulse is valid, and the AFC loop is tracking in order to bring the burst frequency within the inner hysteresis limits.

(Locked) The burst pulse is valid and the AFC loop is locked. The burst frequency is now within the outer hysteresis limits and has previously been within the inner limits while tracking. This is the stable operational mode in which data acquisition should take place.

5.4.4 Computation of Filter Loss

The **Ps** printout displays the power loss (calibration error) that results when the given filter is applied to the given transmit burst waveform. This allows you to correct for the difference between what a broad–band power meter measures as the overall transmit power, and what the RVP8 narrow–band receiver will detect within its passband. The filter loss is a subtle quantity that depends on the combined characteristics of both the transmit waveform and the receiver matched filter.

The filter loss is zero if the burst waveform consists of a pure sinusoid at the designated intermediate frequency. It is also very near zero as long as most of the burst energy is confined within the passband of the RVP8s filter. The filter loss will increase as the bandwidth of the burst waveform increases and begins to spill out of that passband. Typical losses for a well–matched filter are in the 0.5–1.8dB range, depending on the FIR length and other design criteria.

As an example, consider how the RVP8 filters would respond to a simple rectangular pulse of energy lasting T_o seconds. For this discussion we can ignore the sinusoidal IF carrier that must also be present within the pulse, and just focus on the rectangular envelope. This is valid because the signal bandwidth, and hence the filter loss, is determined entirely by the shape of the modulation envelope. For a pulse of length T_o to have unit–energy it must have an amplitude of $1/\sqrt{T_o}$. By centering this pulse at time zero the power spectrum is easily computed using a real-valued integral:

$$S(f) = \left(\int_{-T_0/2}^{T_0/2} \frac{1}{\sqrt{T_0}} \cos(2(\pi f t)t)\,dt \right)^2 = \frac{\sin^2(\pi f T_0)}{\pi^2 f^2 T_0}$$

where *f* is the frequency in Hertz. This is the familiar "synch" function, whose main frequency lobe extends from $1/T_o$ to $1/T_o$ Hertz, and whose total power integrated over all frequencies is 1.0.

We can now examine what the filter loss (dB_{loss}) would be if this pulse were applied to a bandpass filter. The filter loss is simply the ratio of the power that is passed by the filter, divided by the total input power (1.0 in this case). Assume for the moment that the filter is an ideal bandpass filter centered at zero Hertz (corresponding to how *S(f)* was defined) and having a bandwidth B_w , then:

$$dB_{loss} = -10\log_{10}\left(\int_{-B_w/2}^{B_w/2} S(f)\,df \right)$$

This integral can be computed for a few "interesting" filter bandwidths, yielding filter losses of 0.44dB, 1.11dB, and 3.31dB when B_w is $2/T_o$, $1/T_o$, and $1/2T_o$ respectively. These three example bandwidths correspond to filters that pass the entire main frequency lobe, half of that lobe, and one quarter of it.

You can experimentally verify these results using the RVP8 as follows:

- Using the **Mt0** command, setup a T_o = 0.5 µsec trigger pulse from the RVP8 in the vicinity of range zero, and use that trigger to gate a signal generator whose output is applied to the RVP8/IFD Burst Input. Also setup 125–meter range resolution, and a rather long 6.0 µsec impulse response length. The long length will make the transition edges of the matched filter as steep as possible, so that it becomes a reasonably good approximation to the ideal bandpass filter used in the above analysis.
- Use the **Pb** command to verify that the burst pulse is present, and position the triggers left and right until the pulse is centered exactly at zero.
- Use the **Ps** command to examine the frequency spectrum of the pulse. You should see a main energy lobe that is 4MHz wide and centered at the radars IF. There should also be weaker lobes spaced 2MHz apart on both sides of the main lobe. If the lobe spacing does not look quite

right, it may be because the signal generator has slightly shortened or lengthened the trigger gate.

- Continue using **Ps** to examine filters that are 4MHz, 2MHz, and 1MHz wide at their 3dB points. You should see filter losses reported that are very close to the theoretical values for the ideal bandpass filter.

In the above analysis we have assumed that S(*f*) is the idealized power spectrum of a continuous time signal. Of course, the RVP8 filter loss algorithm can only work from an estimate of S(*f*) that is obtained from a finite number of samples. The filter loss calculation thus becomes more complicated than the above example in which we integrated an idealized filter response over an idealized power spectrum.

Let $\hat{B}(f)$ denote the estimated power spectrum of the continuous–time Tx burst waveform, for which we have only a finite number of discrete samples $\{b_n\}$. For purposes of this discussion we can assume that the frequency variable *f* is continuous. Furthermore, let $\hat{C}(f)$ denote a power spectrum estimate that is derived in an identical manner using the same number of samples, but of a pure sine wave at the radar's IF. The RVP8 determines $\hat{B}(f)$ according to its sampled measurement of the transmitted waveform; however it can calculate $\hat{C}(f)$ internally based on an idealized sinusoid. The reported filter loss is then:

$$dB_{loss} = -10\log_{10}\left(\frac{\int |H(f)|^2 \hat{B}(f)df}{\int \hat{B}(f)df} \div \frac{\int |H(f)|^2 \hat{C}(f)df}{\int \hat{C}(f)df}\right)$$

Where $|H(f)|^2$ is the spectral response of the RVP8 IF filter, and the integrals are performed over the Nyquist frequency band that is implied by the RVP8/IFD sampling rate. Note that the two integrals involving $\hat{C}(f)$ will have constant value and need only be computed once. They serve to normalize the $\hat{B}(f)$ integrals in such a way that the filter loss evaluates to 0dB whenever the transmit burst is a pure tone at IF.

This normalization is necessary for the filter loss values to be meaningful. Regardless of the bandwidth and center frequency of $H(f)$, the filter loss should be reported as 0dB whenever the Tx waveform appears to have zero spectral width, i.e., is indistinguishable from a pure IF sinusoid. Of course, the real Tx waveform has only finite duration, and thus should never look like a pure tone as long as the RVP8 is able to "see" the entire Tx envelope. For this reason, it is important that the filter's impulse response length be set long enough (using the **Pb** plot) to insure that all of the details of the Tx

waveform are being captured. If the entire Tx envelope does not fit within the FIR filter, then the filter loss will be underestimated because the Tx spectrum will appear to be narrower than it really is.

The RVP8's calculation of digital filter loss is very similar to how the loss of an analog filter would be measured on a test bench. Suppose we are given an analog bandpass filter and are asked to determine its spectral loss when a given waveform is presented. We could use a power meter to measure the waveform power before and after the filter is inserted, and compute the ratio of these two numbers. This corresponds to the first integral ratio in the above equation. However, this is not by itself an accurate measure of filter loss because it does not take into account the bandwidth–independent insertion loss. Put another way, a flat 3dB pad would seem to produce a 3dB filter loss in the above measurement, but that is certainly not the result that we desire. The remedy is to make a second pair of power measurements of the filter's response to a CW tone at the passband center. This serves to calibrate the gain of the filter, and allows us to compute a filter loss that captures the effects of spectral shape independent of overall gain. This normalization step corresponds to the second integral ratio in the above equation.

If your radar calibration was performed using CW waveforms, then the reported filter loss should either be added to the receiver calibration losses, or subtracted from the effective transmit power; the net result being that dBZ_0 will increase slightly.

In dual–receiver systems the filter loss is computed for the primary and secondary channels using only the portion of bandwidth that is allocated to that channel. For example, if the two IFs are 24MHz and 30MHz, then the filter losses for each channel would use the frequency intervals 21–27MHz and 27–33MHz respectively. This is necessary to avoid picking up energy from the other receiver and interpreting it as out–of–band input power. A consequence, however, is that the real out–of–band power is underestimated, that is, the filter loss itself is underestimated. We recommend temporarily switching dual–receiver systems back to single–receiver mode when the filter loss is being measured. This is easily done by changing the **Mc** setup question back to "single", and disconnecting the secondary burst input to the RVP8/IFD.

5.4.5 Recommended Adjustment Procedures

The **Ps** command should be used only after the burst pulse has been successfully captured by way of the **Pb** command. Use the <space> key to display the burst spectrum plot by itself, and use the Z key to shift the entire graph into view. You are now looking at the actual frequency content of the transmitted pulse. The plot should show a clean main power lobe centered at the receivers intermediate frequency. Check the spectrum for spurious harmonics, excessive width, and other out–of–band noise. Make any adjustments in the transmitter that might give a sharper main lobe or reduced spurious noise.

Once we know the power spectrum of the transmitted pulse we can begin designing the matched FIR filter. Use the <space> key to display both the filter response and the burst spectrum on the same plot. Use the "Z" key to shift the bursts main lobe up to the top horizontal line of the graph. This makes it level with the filters peak lobe, which is always drawn tangent to the same top line.

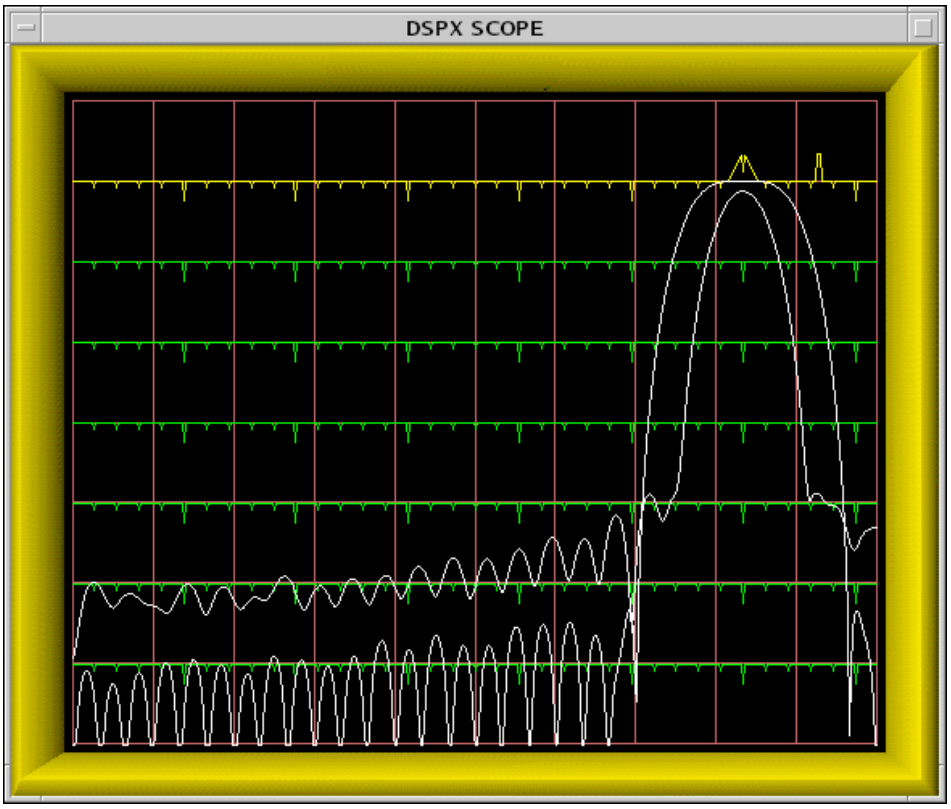


Figure 23 Example of a Poorly Matched Filter

Begin with the FIR length that was chosen previously in the **Pb** command, and use the "N" and "W" keys to set an initial bandwidth equal to the reciprocal of the pulsewidth. The main lobes of the two plots should more–or–less overlap. Experiment with changing the FIR length and bandwidth to achieve a filter with the following properties.

- The filter width should be no greater than the burst spectral width. A wider passband will reduce the SNR of the received signal because out–of–band noise would be allowed to pass.
- The DC gain should be as small as possible, preferably less than –64dB (See discussion below).
- If there are conspicuous interference spikes at particular frequencies, try to adjust the location of the filters zeros so that the interference is maximally attenuated.

The filter should not pass any frequencies that do not actually contain useful information from the original transmitted pulse. If anything, choose a filter whose width is slightly narrower than the bursts spectral width. Figure 23 on page 180 shows an example of a filter that is poorly matched to thepulse. Although the filter has fairly good DC rejection, it passes frequencies that are outside of the transmitter's broadcast range. These frequencies contribute nothing but noise to the synthesized "I" and "Q" data stream.

There are two procedures for optimizing the performance of the FIR filter:

- **Manual Method** — The process of arriving at a nearly optimal filter will require a few minutes of hunting with the "I", "W", and "N" keys. Every time you press any of these keys the RVP8 designs a new FIR filter from scratch, and displays the results. Fortunately, the DSP chips are fast enough that this can be done quickly and interactively. Even though the user must still control two degrees of freedom (length and bandwidth), the RVP8's internal design calculations are actually performing several hundred iterative steps each time, which preferentially select for the best filter. Because the FIR coefficients are quantized in the filter chips themselves, the process of finding an optimal filter becomes quite nonlinear.
- **Automatic Method** — Simply type the $ command and let the RVP8 do all of the work (See description in 5.4.2 Available Subcommands Within Ps on page 171).

The offset error of the RVP8/IFD's A/D converter is at most 10mV, that is, -27dBm into its 50Ω input. If we wish to achieve 85–dB of dynamic range below the converters +4dBm saturation level, then we expect usable "I" and "Q" values to be obtainable from a (sub–LSB) input signal at -81dBm. This is 54dB below the interference that would result from the worst–case A/D offset. But a weak input signal at -81dBm would still be damaged by

even an equal level of DC interference. Therefore, adding another 10dB safety margin, we get -64dB as the recommended maximum DC gain of the matched filter. This DC gain should be reduced even further if it is known that coherent leakage is present in the receive signal at a level greater than the -27dBm worse–case A/D offset.

Figure 24 on page 182 shows a 60MHz filter with particularly poor (-42dB) DC rejection. The frequency range of the plot is 36–72MHz; hence, DC appears aliased at the right edge and we can see a peak in the filters stopband at DC. Contrast this with the filter shown in Figure 22 on page 169 that has atrue zero at DC. In general, a poor filter can be converted into a "nearby" good filter by making only incremental changes to the impulse response length and/or desired bandwidth.

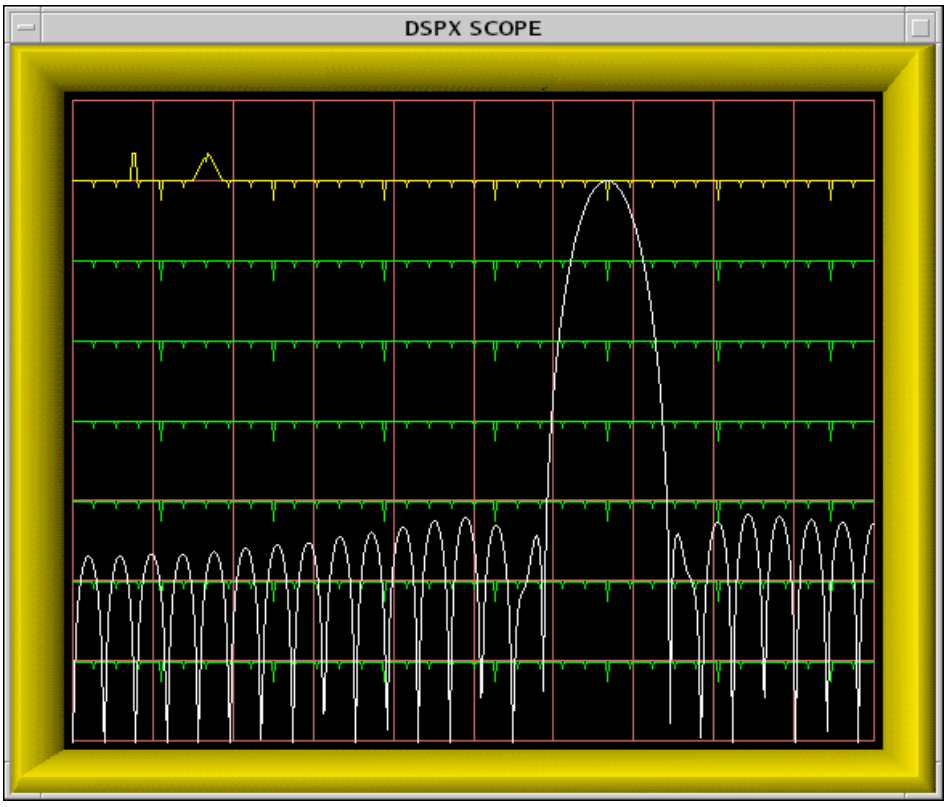


Figure 24 Example of a Filter With Poor DC Rejection

5.5 Pr — Plot Receiver Waveforms

The **Pb** and **Ps** commands described in the previous sections have been used to analyze the signal that is applied to the "Burst–In" connector of the RVP8/IFD receiver module. The task that remains is to checkout the actual received signal that is connected to "IF–In". The goal is to verify that the received signal is clean and appropriately scaled, and that nearby targets can be seen clearly. The **Pr** command is used to make these measurements.

5.5.1 Interpreting the Receiver Waveform Plots

An example of a plot from the **Pr** command is shown in Figure 25 on page 183. The horizontal axis represents time (range) starting from a selectable offset and spanning a selectable interval. The data are acquired from a single transmitted pulse, are are plotted both as raw IF samples and as the LOG of the detected power using the FIR filter for the current pulsewidth.

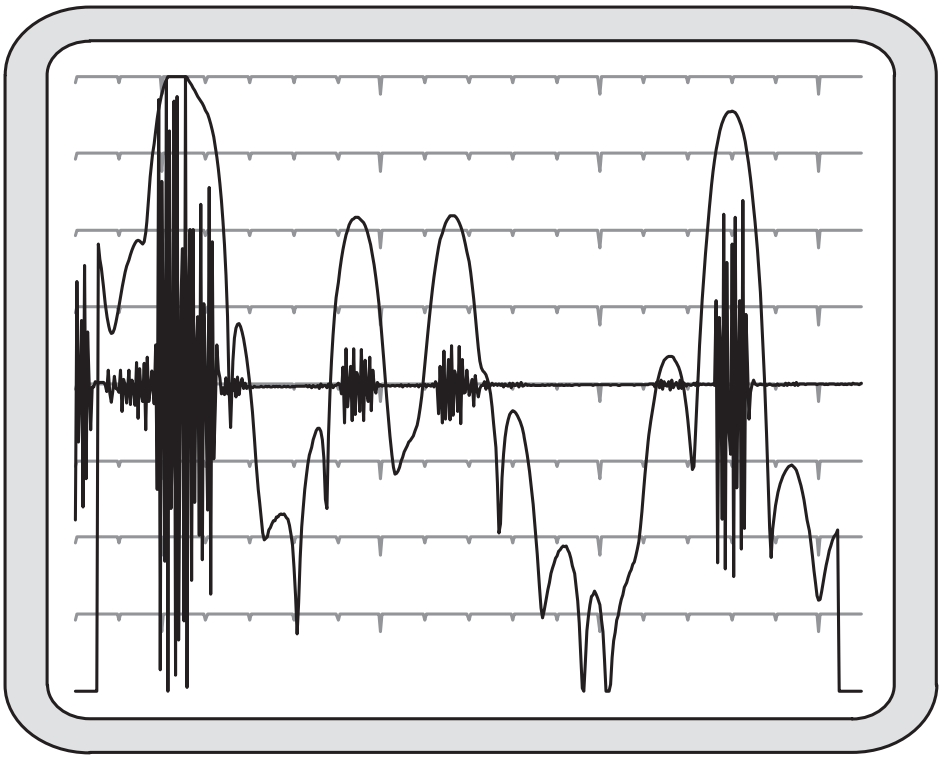


Figure 25 Example of Combined IF Sample and LOG Plot

The IF samples are plotted on a linear scale as signed quantities, with zero appearing at the center line of the scope. Any DC offset that may be present in the A/D converter is not removed, and will be seen as a shift in the

baseline at higher zoom levels. For example, the converter's worst case DC offset of 10mv would appear as a 91–count offset in the 12–bit range spanning –2048 to +2047. At the x32 or higher zoom scales, this offset would peg the sample plot off scale. Typically the DC offset will be much less than this worst case value; but the RVP8 preserves the DC term in the **Pr** sample plot so that its presence is not forgotten.

The "AC" amplitude of the IF samples will increase wherever targets are present. On top of these samples is drawn the detected power on a logarithmic scale. Each horizontal line represents a 10dB change in power. The graph is scaled so that the LOG power reaches the top display line when the samples occupy the full amplitude span. Using Figure 25 on page 183 as an example, the two equal–power targets just to the left of center are approximately 18dB down from the top. The amplitude of the samples is thus $10(-^{18/20}) = 0.13$, that is, 13% of full scale. This correspondence between the LOG scale and the amplitude scale applies regardless of the plot's zoom level. As the IF samples are zoomed up and down by factors of two, the LOG plot will shift up and down in 6dB steps.

The LOG plot is obtained by convoluting the FIR filter coefficients with the raw IF data samples, and then plotting $\log(I^2 + Q^2)$ at each possible offset along the sampling interval. This convolution produces only (1 + N - I) output points, where N is the number of sample points and "I" is the length of the FIR filter. For this reason the LOG plot begins approximately I/2 samples from left side and ends approximately I/2 samples from the right.

The LOG points are computed at each possible offset within the raw IF samples. At the nominal 72MHz sampling rate the spacing between LOG samples will be a mere 4.17 meters. Thus, the LOG plot gives a very detailed view of received power versus range. Of course, successive LOG points will be highly correlated because successive input data intervals differ by only one sample point. This is why the LOG plots appear smooth compared to the instantaneous variation of the raw IF samples.

As the starting offset of the **Pr** plot is decreased to range zero you will begin to see part of the burst pulse (the second half of it) appear at the left edge of the plot. This is because the burst data samples are multiplexed onto the same fiber cable that carries the IF data samples. Zero range is defined to occur at the center of the burst window; hence, the later half of the burst pulse will be visible when the plot begins at range zero.

A second type of **Pr** display is shown in Figure 26 on page 185. This plot shows a frequency spectrum of the received data samples in a format that is nearly identical to the **Ps** display. The horizontal axis represents the same band of frequencies (half the sampling rate), and the vertical axis represents power in 10dB steps. The entire vertical axis is used so that an

overall span of 80dB is visible. This particular plot was made with the time span set to 50 µsec, and with a 1–meter antenna attached to the IF input so that a broad range of signals (radio stations, electrical noise, etc.) would be detected.

The purpose of the **Pr** power spectrum is to check for spurious interference in the IF signal from the radar receiver. The spectrum should be viewed with the transmitter turned off, and with the starting range moved out so that the burst samples are not mixed in with the receiver data. The power spectrum is computed using the complete interval of raw IF samples which, depending on the chosen time span, may contain many hundreds of points. The frequency resolution of the **Pr** spectrum can therefore be quite fine; making it possible to discern any interfering frequencies with some detail.

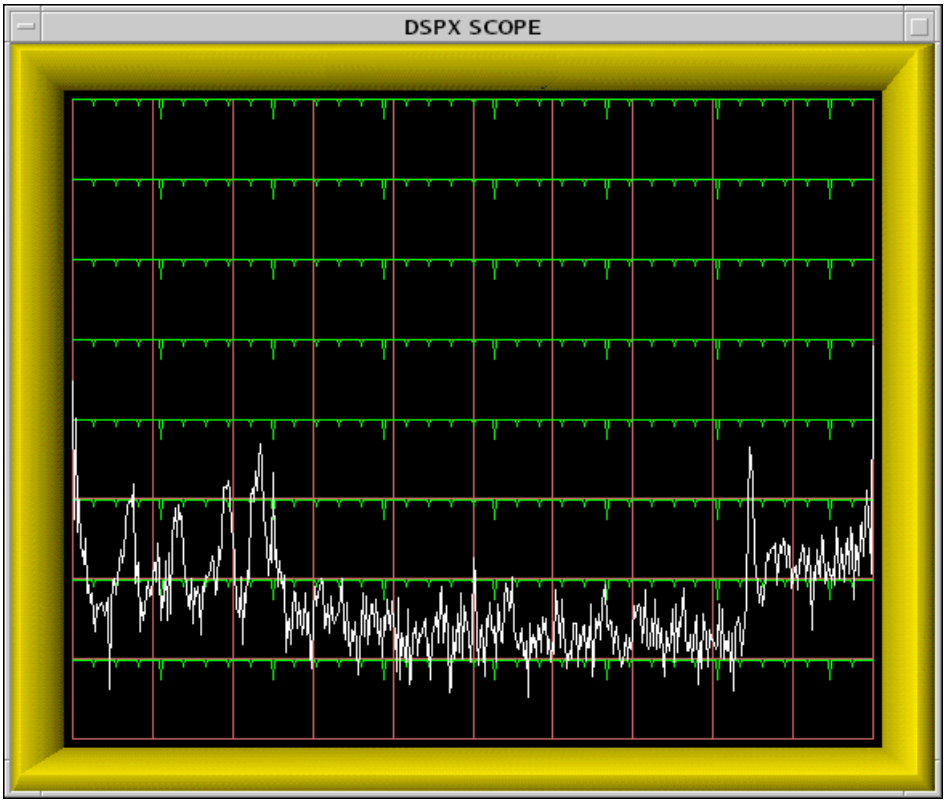


Figure 26 Example of a Noisy High Resolution Pr Spectrum

The **Pr** spectrum plot will properly show a 0–Hz peak from any DC offset in the A/D converter, and is thus consistent with how the DC offset is presented in the **Pr** sample plot. Both of these plots preserve the DC component of the IF samples so that it can be monitored as part of the routine maintenance of the receiver system. This is one of the few places in the RVP8 menus and processing algorithms where the DC term deliberately remains intact.

5.5.2 Available Subcommands Within Pr

The list of subcommands is printed on the TTY:

```
Available Subcommands within Pr:

L/l & R/r Start range Left/Right
T/t       Plot time span Up/Dn
V/v       Number of spectra averaged
Z/z       Amplitude zoom
<space>   Alternate Plots
%         Toggle between dual receivers
-         Single Step
```

These subcommands change the start time and span of the IF sampling window, and alter the format of the display.

L/l & R/r The **L** and **R** commands shift left and right the starting point of the window of IF samples. The lower case commands shift in 0.25 µsec steps, and the upper case commands use 10 µsec steps. The starting point is displayed both in microseconds and kilometers on the TTY, and is not allowed to be set earlier than range zero.

T/t The **T** command increments or decrements the time duration of the window of IF samples. The window is not allowed to become shorter than the impulse response length of the FIR filter, since that would preclude calculating even a single LOG power point. The value is reported in microseconds on the TTY, and the largest permitted span is 50 µsec.

V/v The **V** command increments or decrements the number of power spectra that are averaged together to create the plot. The count ranges from one (no averaging) to 25, and is reported on the TTY as "Navg".

Z/z The **Z** command zooms the amplitude of the IF samples by factors of two from one to 128. The LOG plots are shifted in corresponding 6dB increments as the amplitude is zoomed up and down. The zoom level is reported on the TTY so that absolute power levels and A/D usage can be assessed.

<space> The space bar alternates among three choices for the type of data that are plotted: 1) Received Samples, 2) Received Samples and LOG Power, and 3) Received Power Spectrum.

% In dual-receiver mode, the **%** command toggles between each receiver. The printed status line is prefixed with "Rx:Pri" or "Rx:Sec" according to which receiver is selected. Specifically, typing **%** will toggle the LOG plot of the received power, and the printout of the "Total", "Filtered", and "Midpoint" powers. However, the plots of power spectra and raw IF data samples are always based on the sum of all input signals, and thus do not change with "%".

5.5.3 TTY Information Lines Within Pr

The TTY information lines will resemble:

```
Zoom:x1, Navg:4, Start:0.00 usec (0.00 km), Span:5 usec

Total:63.3 dBm, Filtered:77.6 dBm, MidSamp:77.4 dBm
```

Zoom Indicates the magnification (in amplitude) of the plotted samples. A zoom level of "x1" means that a full scale A/D waveform exactly fills the vertical height of the plot. Generally, the IF signal strength will not be quite this high. Thus, use larger zoom levels to see the weaker samples more clearly. You may zoom in powers of two up to x128.

Navg Indicates the number of spectra and/or LOG powers that are averaged together prior to plotting. Larger amounts of averaging increase the ability to observe subtleties of the signals, but the display will update more slowly.

Start Indicates the starting time of the IF sample window relative to range zero. The time is shown both in microseconds and in kilometers.

Span Indicates the time span of the IF sample window in microseconds.

Total Indicates the total RMS power that is being detected by the IF-Input A/D converters. This total is computed using all of the raw IF samples in the chosen interval, and is the sum of power at all frequencies other than 0 Hz (and its aliases).

Filtered Indicates the RMS power that falls only within the passband of the FIR filter for the current pulsewidth. This is computed using all of the raw IF samples in the chosen interval.

MidSamp Also indicates the RMS power within the passband of the FIR filter, but using only the raw IF samples in the exact center of the chosen interval.

The computation of "Total Power" is performed using the same subset of central IF samples that are used to compute "Filtered Power". This smaller subset of IF samples comes about because filtering the data requires a convolution with the current FIR filter, and this computation does not produce results all the way to the edges of the input data. This is the same reason that the LOG plots do not extend across the full screen.

Because of this definition, it is valid to intercompare the "Total Power" and "Filtered Power". The two numbers will match exactly as long as all of the incoming power falls within the passband of the FIR filter. The difference between the two powers can be used as a measure of the "filter loss" for a given pulse shape, that is, the portion of signal that is lost outside of the filter's passband.

NOTE	The "Total", "Filtered", and "MidSamp" values represent true RMS power (that is, variance), and not merely a sum–of–squares. Thus, any DC offset present in the A/D converter will not affect these power levels.

5.6 Pa — Plot Tx Waveform Ambiguity

With the introduction of the RVP8/Tx Digital Transmitter PCI Card it is now possible for the RVP8 to make radar observations using compressed pulse waveforms. This opens up many new opportunities within the weather radar community for using low–power solid–state transmitters that employ very long pulse lengths (20–80µ sec). Transmitters of this kind are less expensive, both to build and to maintain, compared with traditional Magnetron or Klystron systems.

However, the signal processing and waveform design that are required to make good use of these long transmit pulses is also much more complex. To help with this, the RVP8 provides the **Pa** (Plot Ambiguity) command in which compressed transmit waveforms can be designed, studied, and optimized. Within the **Pa** plots you can experiment with different waveform designs, try out various bandwidth and pulsewidth options, and examine and optimize the range/time sidelobes of your waveform.

5.6.1 Interpreting the Ambiguity Plots

Figure 27 on page 189 shows one form of **Pa** plot in which the magnitude of the Tx/Rx range sidelobes are drawn on a log scale having 10dB vertical ticks. The horizontal span of the plot is equal to the length of the pulse, and consequently, only half of the complete ambiguity diagram is shown. This was done to make the plots more viewable; and no information lost since the zero–Doppler response (white plot) can safely be assumed to be symmetric. In this example the pulsewidth is 30 sec, bandwidth is 3MHz, PSL is -61.2dB and ISL is -50.8dB, Doppler shift +/- 50 KHz.

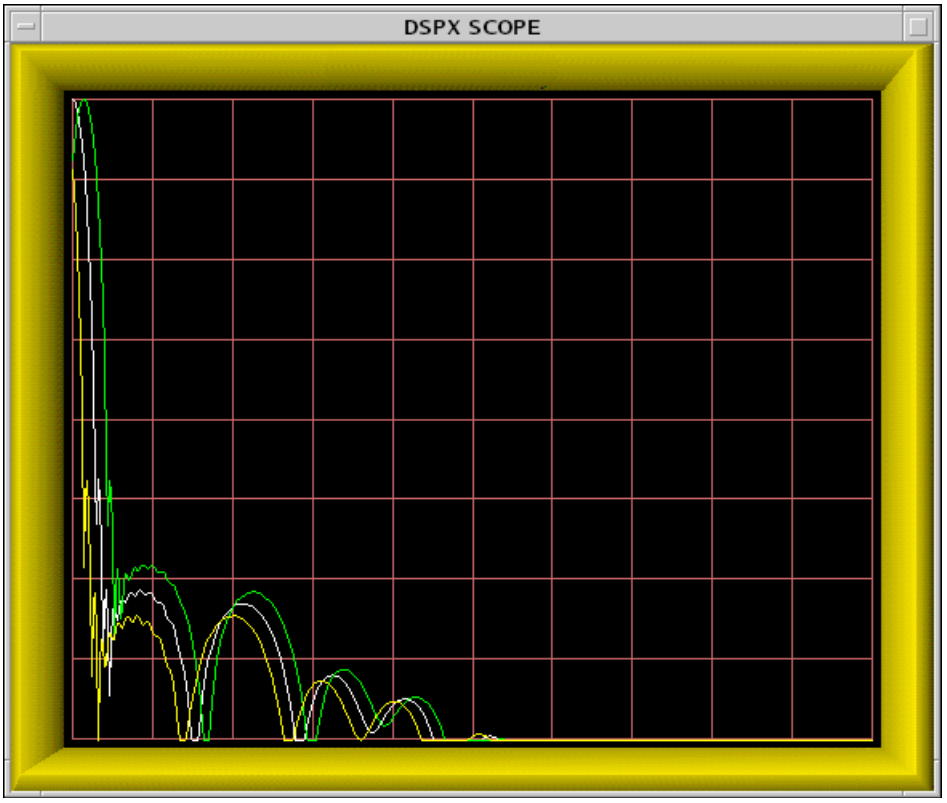


Figure 27 Ambiguity Diagram of a Compressed Tx Pulse

Also shown in yellow and green are the Tx/Rx responses when the overall waveform is modified by a 50KHz target Doppler shift. Real weather targets would never have such a large Doppler component, but the **Pa** menu allows you to study its effect anyway.

An alternate form of **Pa** plot of the same Tx waveform is shown in Figure 28 on page 190. The horizontal axis again represents time, but now spans the entire duration of the pulse. Three different plots are drawn, hence the vertical axis is interpreted differently in each case:

- The instantaneous frequency across the full length of the pulse is shown in white. The vertical scale is normalized to hold the overall frequency span, which is also shown numerically in the **Pa** TTY output.
- The waveform baseband phase is shown in green, and is normalized so that the vertical axis holds the full span of values. Note that the phase, which generally spans a few thousand degrees, is "unwound" in this plot so that you can see its behavior nicely.
- The amplitude of the Tx waveform envelope is shown in yellow. It is drawn using a linear vertical scale which occupies only the middle half of the plot. This is to avoid creating too much "plotting clutter" in the corners.

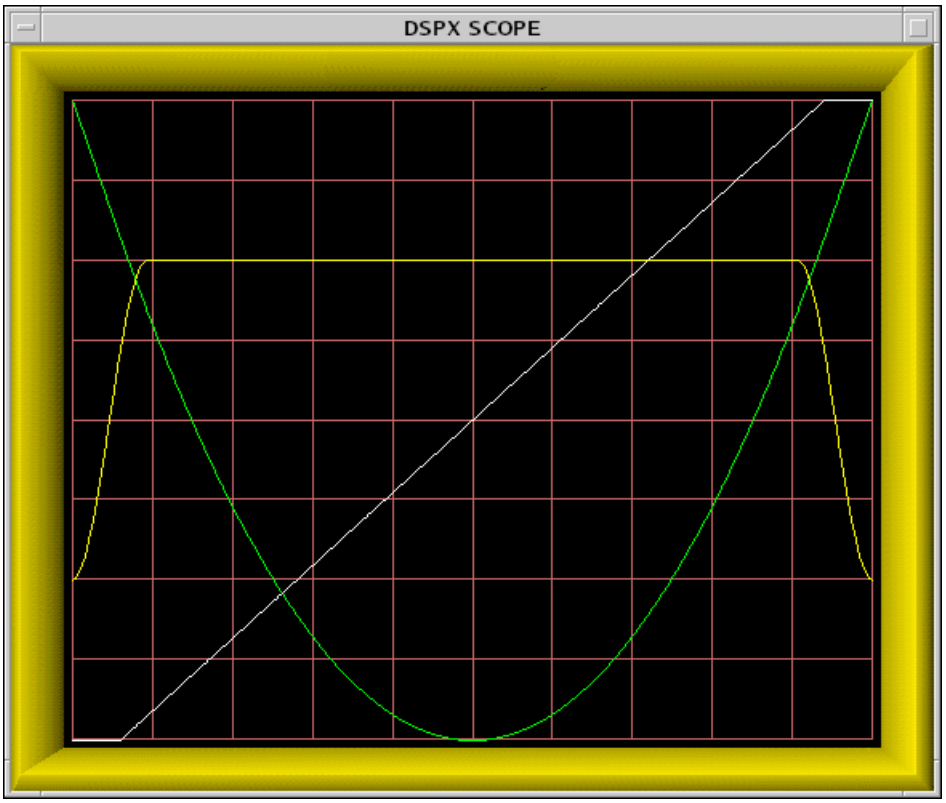


Figure 28 Frequency, Phase and Amplitude of a Compressed Tx Pulse

From Figure 28 on page 190 we can see that the waveform consists of a linear FM chirp that occupies about 87% of the central pulse duration. The frequency remains nearly constant in the leading and trailing edges, hence the overall label "Non–Linear FM". The central chirp is contained within a somewhat larger amplitude modulation envelope that applies full scale power within the middle 82% of the pulse, and also provides bandlimited shaping of the leading and trailing edges.

This waveform was designed using the "$" automatic search–and–optimize command in the **Pa** menu. For a given pulse length and bandwidth of the Tx waveform, this command allows you to try thousands of combinations of FM shape and amplitude shape, searching for the combination that minimizes the sum of PSL and ISL (in dB). This gives the best overall waveform for weather radar observations in which both the PSL and ISL are important.

5.6.2 Available Subcommands Within Pa

The list of subcommands is printed on the TTY:

```
Available Subcommands within 'Pa':

S/s & L/l Pulse Length Shorter/Longer
N/n & W/w Bandwidth Narrower/Wider
1/2/3     Select Tuning Parameter to Change
D/d & U/u Selected Tuning parameter Down/Up
V/v       Doppler Frequency Shift Up/Down
Z/z       Amplitude Zoom
$         Search for Optimal Waveform
```

These subcommands change the bandwidth, pulsewidth, and shaping parameters of the transmit waveform, and alter the format of the display.

S/s & L/l The "**S**horter" and "**L**onger" commands decrease or increase the time duration (that is, pulsewidth) of the transmitted pulse. The lower case commands shift in 0.05 µsec steps, while the upper case commands use 1 µsec steps.

N/n & W/w The "**N**arrower" and "**W**ider" commands decrease or increase the bandwidth of the transmitted pulse. The lower case commands shift in 10KHz steps, while the upper case commands use 200KHz steps.

1 & 2 & 3 Typing one of these numbers chooses which of the three waveform tuning parameters will be altered by the "D" and "U" keys.

D/d & U/u	The "**D**own" and "**U**p" commands decrease or increase the waveform tuning parameter that has been chosen by the most recent "1", "2" or "3" key. The lower case commands shift in 0.001 (dimensionless) steps, while the upper case commands use 0.05 steps. Present values of all three parameters will be printed each time a Down/Up key is pressed, for example: **Tuning parameters: 0.9000 1.0000 0.1770** Please see 4.2.6 Special Options for Tx Synthesis on page 139 for a description of how each of the tuning parameters are used.
V/v	The "**V**elocity" command allows you to investigate how the overall compressed pulse Tx/Rx system will respond to the the effects of target Doppler shift. Frequency shifts as large as 100KHz can be introduced in order to see their effect on the Range/Time sidelobes. The **Pa** plot will show the effects of +V and V shifts as green and yellow plots in addition to the standard white (zero Doppler) plot.
Z/z	The dynamic range of the **Pa** sidelobe plot is 80dB. Usually this will give plenty of room to examine the properties of the waveform. But for very wide dynamic range pulses, you can shift the plot up/down in 10dB steps using these "**Z**oom" keys.
$	Designs an optimal compressed waveform. For a given pulsewidth and bandwidth of the Tx waveform, this command allows you to try many thousands of combinations of FM shape and amplitude shape, searching for the one that minimizes the sum of PSL and ISL (in dB). This gives the best overall waveform for weather radar observations in which both the PSL and ISL are important.

The following dialog appears in response to the "$" command:

```
TuningParam #1 (0.9000)  1 Grid Point from
0.9000 to 0.9000
TuningParam #2 (1.0000)  1 Grid Point from
1.0000 to 1.0000
TuningParam #3 (0.1774)  1 Grid Point from
0.1774 to 0.1774
```

As the line is printed for each parameter, you may either accept the default single grid point (no search), or enter a desired number of grid search points followed by a span of values within which to search. For example, typing:

200 .9 .95

will request that the parameter be searched using 200 evenly spaced grid points lying between 0.9000 and 0.9500 inclusive. After all three parameter spans have been entered, the RVP8 will begin searching for the optimum waveform. Progress messages are printed on the TTY, and the plot will update every time a better waveform is discovered. In this way it is easy to tell whether the search is converging.

The process normally runs to completion on its own; but if the search is taking too long, or youve changed your mind about which intervals to search, typing "Q" will exit right away. In either case, the prompt:

Keep this waveform ? [Y]

will appear. Typing "Y" (or Enter) will keep the optimized tuning parameters that were just discovered, overwriting whatever starting values were originally there. Typing "N" will discard the search results and return to the original settings, as if "$" had never been typed.

5.6.3 TTY Information Lines Within Pa

The TTY information lines will resemble:

```
BW:3.40MHz PW:29.99usec PSL:61.2dB ISL:51.3dB

TxLoss:0.5dB RxLoss:2.4dB
```

BW Bandwidth of the Tx waveform in MegaHertz.

PW Pulsewidth (pulse length) of the Tx waveform in microseconds.

PSL Peak Sidelobe Level of the ambiguity diagram, expressed in deciBels relative to the main lobe level. This is the peak height of the strongest range/time sidelobe, and measures the ability of the compressed pulse to distinguish a given target from a small number of individual point targets that also lie within the pulse volume. The waveforms ability to "see" between clutter targets is largely determined by the PSL level.

ISL Integrated Sidelobe Level of the ambiguity diagram, expressed in dB relative to the main lobe. This is the total power in all range/time sidelobes divided by the total power in the main lobe. ISL measures the ability of the compressed pulse to distinguish a given target from other distributed targets (such as rain) that also lie within the pulse volume.

TxLoss The TxLoss is calculated as the total power in the transmit waveform divided by the power that would be contained in an equal length ideal rectangular pulse. It is a measure of how much power does not get transmitted due to the amplitude shaping of the synthesized waveform. TxLoss should be included in the computation of the radar constant, since the latter is based on a nominal pulsewidth equal to the overall length of the entire Tx waveform (including the amplitude tapering).

RxLoss The RxLoss is a measure of how much information is "thrown away" by the receiving filter in order to achieve the desired level of sidelobe suppression. These two quantities often trade off against each other in receiver systems, so that optimum range/time sidelobes can only be achieved at the expense of a few deciBels of loss of sensitivity. The receiver filter loss is calculated as:

$$dB_{loss} = -10\log_{10}\left(\frac{\left|\int T(t)R(t)dt\right|^2}{\int|T(t)|^2dt\int|R(t)|^2dt}\right)$$

Where T(t) is the complex-valued transmit waveform, and R(t) is the complex-valued filter being used to receive it. When R(t) is designed to be the complex conjugate of T(t), we have the ideal matched filter case whose receive loss is 0dB. However, this matched filter has rather poor sidelobe behavior that makes it unsuitable for use directly in the receiver. Instead, a windowed version (Hamming, Blackman, etc.) of the ideal matched filter is used to achieve the desired sidelobe levels. Of course, that windowing operation also has the effect of discarding some valid information in the leading and trailing portions of the pulse. Hence, there is a loss in receive sensitivity whenever a window is applied.

NOTE Given a compressed transmit waveform, the RVP8 designs the appropriate "mismatch" Rx filter automatically, using an optimized Blackman window in all cases. Code developers can also access the internal APIs directly to design any desired transmit waveform along with the associated FIR filter to receive it.

5.6.4 Bench Testing of Compressed Waveforms

Working with compressed pulse waveforms can be tricky, so it is reassuring to run some simple bench tests to verify that things are working properly. Once the Tx waveform has been designed it can be injected into the IFD for testing with the **Pr** command. This not only verifies that the analog waveform is generated properly, but also that the matched filtering on the RVP8/Rx card is able to deconvolve the compressed information.

To setup the test, simply connect the Channel #1 or Channel #2 output of the RVP8/Tx card to the IF–Input of the IFD. Use whichever RVP8/Tx channel has been configured for waveform synthesis in the **Mz** menu, and set the *Zero Offset of the Transmitter Pulse* in the **Mt<n>** menu to, perhaps, 50µ sec. The latter step will shift the waveform out in range so that the **Pr** plot is able to see it.

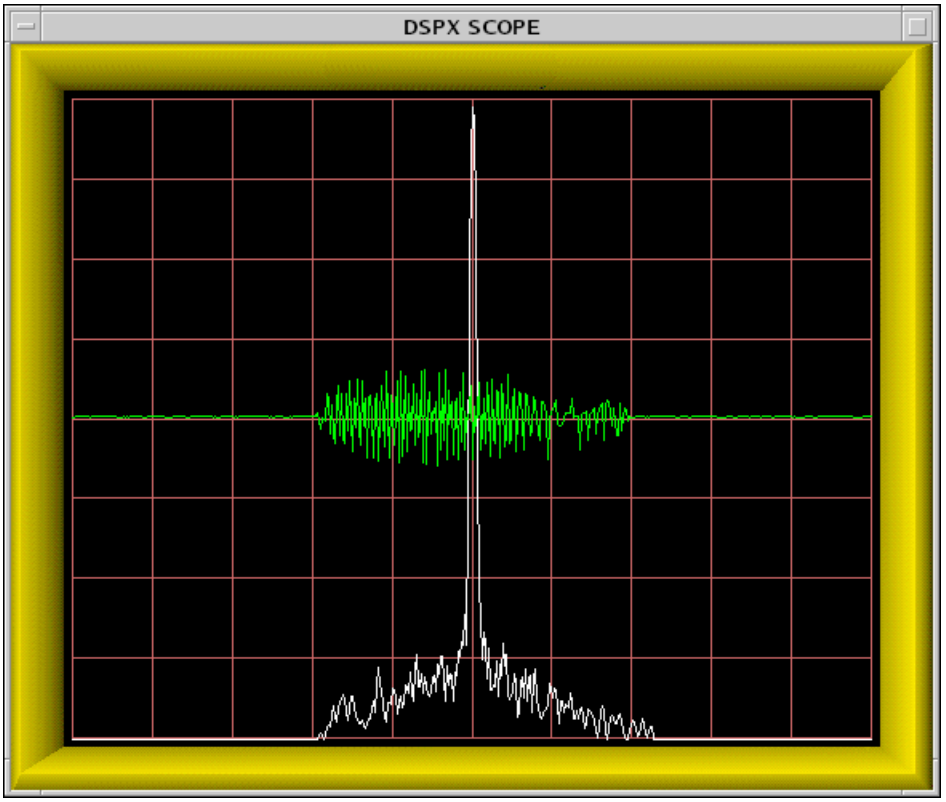


Figure 29 IFD Sampling of Optimized Compressed Tx Waveform

Figure 29 on page 196 shows an actual **Pr** plot of a 40µ sec, 5MHz optimized waveform generated by the RVP8/Tx card and fed into the IFD. In this example, the ideal Tx waveform has a Peak Sidelobe Level (PSL) of -76.7dB and an Integrated Sidelobe Level (ISL) of -62.3dB. The measured testbench performance is several dB short of this, probably because of the uncompensated analog bandpass filters on the RVP8/Tx and IFD. These filters have several tenths of a dB of amplitude ripple as well as minor deviations from linear phase within the 5MHz signal bandwidth. The effect is that the sampled analog waveform is not quite identical to the ideal waveform.

The **Ps** plotting command can also be used to examine the ideal transmit spectrum and actual received spectrum of compressed pulses. An example is shown in Figure 30 on page 197 below for a 60MHz, 40µ sec linear FM pulse having a bandwidth of 2MHz. The energy in the pulse is both sharply

contained within and uniformly distributed over the 2MHz frequency interval centered on the IF carrier. This demonstrates the ability of synthesized transmit waveforms to remain cleanly within their allocated bounds.

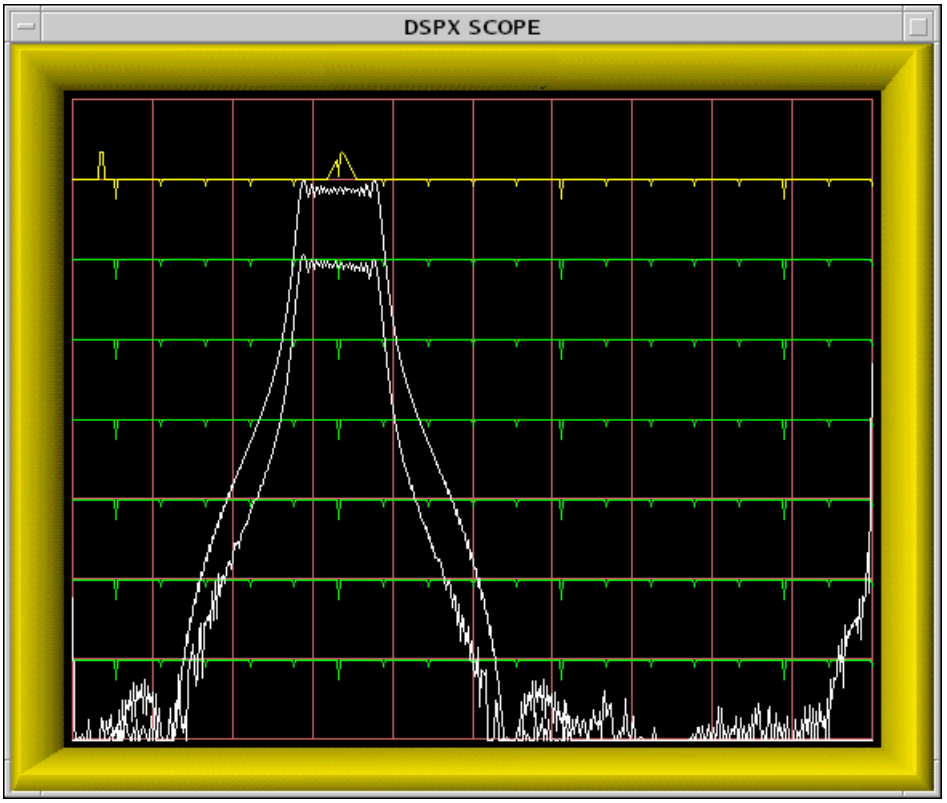


Figure 30 Ideal and Actual Linear–FM Spectrum Displayed in Ps Plot

CHAPTER 6
PROCESSING ALGORITHMS

NOTE Optional dual polarization processing algorithms are described in Appendix B, Optional Dual Polarization- ZDR, PHIDP, KDP, LDR, ..., on page 363.

This chapter describes the processing algorithms implemented within the RVP8 signal processor. The discussion is confined to the mathematical description of these algorithms. Figure 31 on page 202 shows the overall process by which the RVP8 converts the IF signal into corrected reflectivity, velocity, and width. Table 16 on page 201 summarizes the quantities that are measured and computed by the RVP8. The type of the quantity (that is, real or complex) is also given. Subscripts are sometimes used to denote successive samples in time from a given range bin. For example, sn denotes the "I" and "Q" time series or "video" sample from the n'th pulse from a given range bin. In cases where it is obvious, the subscripts denoting the pulse (time) are dropped. The descriptions of all the data processing algorithms are phrased in terms of the operations performed on data from a single range bin- identical processing then being applied to all of the selected ranges. Thus, there is no need to include a range subscript in this data notation.

It is frequently convenient to combine two simultaneous samples of "I" and "Q" into a single complex number (called a phaser) of the form:

$$s = I + jQ$$

where "j" is the square root of -1. Most of the algorithms presented in this chapter are defined in terms of the operations performed on the "s"'s, rather than the "I"'s and "Q"'s. The use of the complex terms leads to a more concise mathematical expression of the signal processing techniques being used. In actual operation, the complex arithmetic is simply broken down into its real-valued component parts in order to be computed by the RVP8 hardware. For example, the complex product:

$s = W \times Y$

is computed as

$Real\{s\} = Real\{W\}\ Real\{Y\} - Imag\{W\}\ Imag\{Y\}$

$Imag\{s\} = Real\{W\}\ Imag\{Y\} + Imag\{W\}\ Real\{Y\}$

where "Real{}" and "Imag{}" represent the real and imaginary parts of their complex-valued argument. Note that all of the expanded computations are themselves real-valued.

In addition to the usual operations of addition, subtraction, division, and multiplication of complex numbers, we employ three additional unary operators: "||", "Arg" and "*". Given a number "s" in the complex plane, the magnitude (or modulus) of s is equal to the length of the vector joining the origin with "s", that is by Pythagoras:

$$|s| = \sqrt{Real\{s\}^2 + Imag\{s\}^2}$$

The signed (CCW positive) angle made between the positive real axis and the above vector is:

$$\measuredangle = Arg\{s\} = \arctan\left[\frac{Imag\{s\}}{Real\{s\}}\right]$$

where this angle lies between - π and + π and the signs of Real{s} and Imag {s} determine the proper quadrant. Note that this angle is real, and is uniquely defined as long as |s| is non-zero. When |s| is equal to zero, Arg{s} is undefined. Finally, the "complex conjugate" of "s" is that value obtained by negating the imaginary part of the number, i.e.,

$s^* = Real\{s\} - j\ Imag\{s\}$.

Note that Arg{s*} = –Arg{s}. The reader is referred to any introductory text on complex numbers for clarification of these points.

Table 16 Algebraic Quantities Within the RVP8 Processor

p	Instantaneous IF-receiver data sample	*Real*
b	Instantaneous Burst-pulse data sample	*Real*
I, Q	Instantaneous quadrature receiver components	*Real*
s	Instantaneous time series phaser value	*Complex*
s'	Time series after clutter filter	*Complex*
T_0	Zeroth lag autocorrelation of *A* values	*Real*
R_0	Zeroth lag autocorrelation of *A'* values	*Real*
R_1	First lag autocorrelation of *A'* values	*Complex*
R_2	Second lag autocorrelation of *A'* values	*Complex*
SQI	Signal Quality Index	*Real*
V	Mean velocity	*Real*
W	Spectrum Width	*Real*
CCOR	Clutter correction	*Real*
LOG	(Signal+Noise)/Noise ratio for thresholding	*Real*
SIG	Signal power of weather	*Real*
C	Clutter power	*Real*
N	Noise power	*Real*
Z	Corrected Reflectivity factor	*Real*
T	UnCorrected Reflectivity factor	*Real*

The following sections cover the various parts of the diagram shown in Figure 31 on page 202, that is,

- IF Signal Processing
- I/Q processing and clutter filtering
- Range averaging and clutter microsuppression
- Moment calculations (reflectivity, velocity, spectrum width)
- Thresholding for data quality and Speckle Filtering
- Reflectivity Calibration
- Special algorithms for ambiguity resolution (dual PRF, dual PRT, Random Phase)
- Calibration and Testing

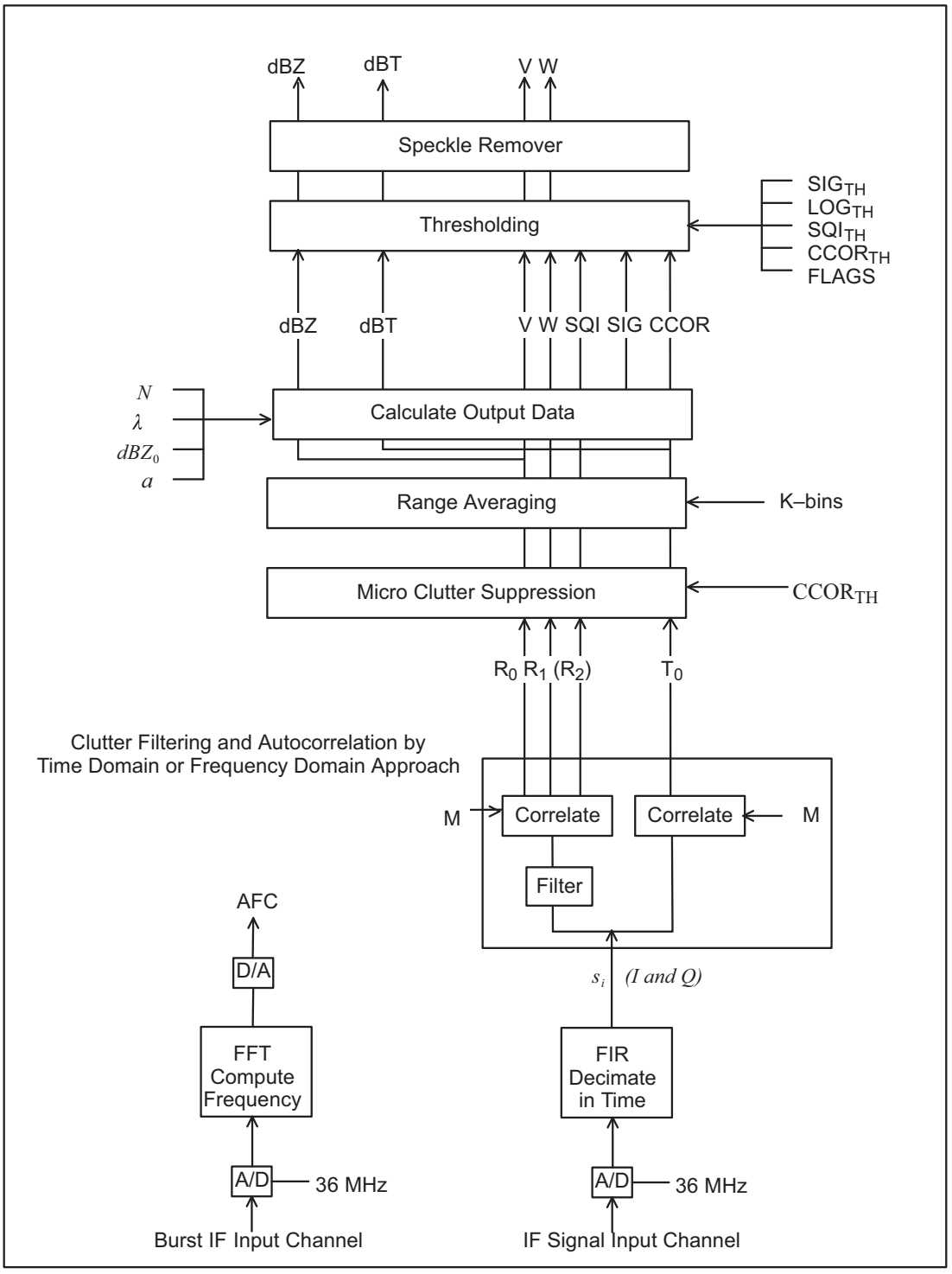


Figure 31 Flow Diagram of RVP8 Processing

6.1 IF Signal Processing

The starting point for all computations within the RVP8 are the instantaneous IF-receiver samples p_n and, the instantaneous burst-pulse or COHO reference samples b_n . These data are available at a very high sampling rate (typically 36MHz), which makes possible the digital implementation of functions that are traditionally performed by discrete components in an analog receiver. The RVP8's all-digital approach replaces a great deal of analog hardware, avoids problems of aging and maintenance, and makes it easy to tune-up the receiver and alter its parameters.

This section describes these IF signal processing steps. Refer to Figure 12 on page 42 for a block diagram of the IF processing that is performed.

6.1.1 FIR (Matched) Filter

The RVP8 implements a digital version of the "matched" filter that is found in the traditional analog radar receiver. The equivalent Finite-Impulse-Response (FIR) filter is designed using an interactive graphical procedure described in 5.4 Ps — Plot Burst Spectra and AFC on page 168. The filter length (number of taps), center frequency, and bandwidth are all adjustable. The design procedure computes two sets of filter coefficients f^i_n and f^q_n such that the instantaneous quadrature samples at a given bin are:

$$I = \sum_{N=0}^{N-1} f^i_n \times p_n \,, \qquad Q = \sum_{N=0}^{N-1} f^q_n \times pn$$

where N is the length of the filter. The input samples pn are centered on the range bin to which the (I, Q) pair is assigned. Note that some of the p_n are likely to overlap among adjacent bins, that is, the filter length may be chosen to be greater than the bin spacing. Such an overlap introduces a slight correlation between successive bins, but the longer length allows a better filter to be designed.

The sums above for I and Q are computed on the RVP8/Rx board using dedicated FIR chips (for revisions A and B) that can perform up to 576 million sums of products per second. The Rev C RVP8/Rx uses a more flexible FPGA. The p_n are represented as 16-bit signed integers, and the f^i_n and f^q_n are represented as 10-bit (Rev.A/B) or 16-bit (Rev.C) signed integers. A numerical optimization procedure is used to quantize the ideal

filter coefficients into their hardware values. The overall spectral purity of the FIR filter will typically be greater than 66dBc (Rev.A/B) and 84dBc (Rev.C).

The reference phase for each transmitted pulse is computed using the same two FIR sums, except with b_n substituted for the pn . For a magnetron system the N b_n samples are centered on the transmitted burst; for a Klystron system they may be obtained from the burst pulse (recommended) or from the CW COHO. If the Klystron is phase modulated by an external phase shifter (as opposed to the RVP8/Tx digital transmitter board), then the samples should be from the modulated COHO.

The f_n^i coefficients are computed as:

$$f^{i}_{n} = l_n \times \sin\left[\frac{\pi}{4} + 2\pi\frac{f_{IF}}{f_{SAMP}}\left(n - \frac{N-1}{2}\right)\right], n = 0 \ldots N-1$$

where f_{IF} is the radar intermediate frequency, f_{SAMP} is the RVP8/IFD crystal sampling frequency, and l_n are the coefficients of an N-point symmetric low-pass FIR filter that is matched to the bandwidth of the transmitted pulse. The multiplication of the l_n terms by the *sin()* terms effectively converts to the low-pass filter to a band-pass filter centered at the radar IF. The formula for the f^{q}_{n} coefficients is identical except that *sin()* is replaced with *cos()*.

The phase of the sinusoid terms, and the symmetry of the l_n terms, has been carefully chosen to have a valuable overall symmetry property when *n* is replaced with (*N-1*)-*n*, that is, the sequence is reversed:

$$f^{i}_{(N-1)-n} = l_{(N-1)-n} \times \sin\left[\frac{\pi}{4} + 2\pi\frac{f_{IF}}{f_{SAMP}}\left(((N-1)-n) - \frac{N-1}{2}\right)\right]$$

$$f^{i}_{(N-1)-n} = l_n \times \cos\left[\frac{\pi}{4} + 2\pi\frac{f_{IF}}{f_{SAMP}}\left(n - \frac{N-1}{2}\right)\right]$$

$$f^{i}_{(N-1)-n} = f^{q}_{n}$$

Thus, the coefficients needed to compute I are merely the reversal of the coefficients needed to compute Q; if you know f_n^i, then you also know f_n^q.

6.1.2 RVP8/Rx Receiver Modes

The RVP8 supports six fundamental IFD and RVP8/Rx configurations which allow you to choose the best style of IF processing for your particular site. The following table summarizes the options where, **BW** is the net IF sampling rate (full 72MHz, or halfband filtered 36MHz), **DynR** is the dynamic range (normal single channel, or extra wide dual channel), **Pol** is the number of polarizations, **Freq** is the number of distinct intermediate frequencies, and **IFD** is the number of IFD's, along with their corresponding RVP8/Rx cards.

#	BW	DynR	Filt	Pol	Freq	IFD	Description
0	Full	Norm	Norm	1	1	1	Standard single channel
1	Full	Norm	Norm	2	2	1	Dual Pol on two frequencies
2	Full	Norm	Norm	2	1	2	Dual Pol on separate IFDs
3	Half	Norm	Norm	2	1	1	Dual Pol on single IFD
4	Half	Wide	Norm	1	1	1	Extra wide dynamic range
5	Half	Norm	Long	1	1	1	Extra long/fast FIR filters

The first three modes were already supported in the previous RVP7 processor. The last three modes are unique to the RVP8 and bring some exciting additional capabilities to the signal processor. The six receiver modes are summarized below. See the *Discussion of Halfband Filtering*6.1.2.1 Discussion of Halfband Filtering Modes 3-5 on page 207 as it applies to Modes 3-5, and the *Discussion of Wide Dynamic Range*6.1.2.2 Discussion of Wide Dynamic Range Mode-4 on page 208 for additional details on using Mode-4.

- **Mode-0: Standard Single Channel** This is the most common "vanilla" mode that is used by single-polarization CW-pulsed radars whose front-end LNA has a dynamic range less than ≤92dB. The (I,Q) data are computed from IF samples at their full acquisition rate (32MHz for Rev.D IFDs, and 72MHz for Rev.F), and the resulting dynamic range from 14-bit IFD samples is well matched to the RF components.
- **Mode-1: Dual-Pol On Two Frequencies** This was the original dual-Pol configuration used by the RVP7 several years ago. A single IFD A/D converter receives the "H" and "V" channels using two distinct intermediate frequencies. Two different STALOs are required in this configuration, making the RF/IF components a bit more expensive, but only one IFD is required.
- **Mode-2: Dual-Pol On Separate IFDs** This mode was introduced into the RVP8 in 2003, and provides dual polarization data using two IFDs connected to two RVP8/Rx cards in the same PCI chassis. A single intermediate frequency is used, hence only one STALO is required.
- **Mode-3: Dual-Pol On Single IFD** This is the recommended dual polarization mode for all new RVP8 installations. The "H" and "V" channels are fed into the Primary and Secondary IFD inputs using a single intermediate frequency. System cost and complexity are both optimized in this design since only a single IFD, RVP8/Rx card, and STALO are required to process both polarization channels.
- **Mode-4: Extra Wide Dynamic Range** Radars having very high performance front-end LNAs can preserve the full benefit of that investment by running two separate IF signals into the Primary (HiGain) and Secondary (LoGain) IFD inputs. A nominal channel separation of 25–30dB might be used to achieve an overall dynamic range of up to 110dB.
- **Mode-5: Extra Long/Fast FIR Filters** This mode is intended for pulse compression systems that require unusually long filters (up to 80μ sec), or finer range resolution in order to employ higher compressed bandwidths without the risk of missing echoes between bins. For example, a 30μ sec pulse could be processed at an incoming range resolution of 50 meters and then range averaged down to 150meter output spacing.

6.1.2.1 Discussion of Halfband Filtering Modes 3-5

Traditionally, the IFD used by the RVP7 and RVP8 has sent raw 14-bit A/D samples from its Burst and IF inputs directly to either the RVP7/Main or RVP8/Rx cards for FIR filtering and conversion into complex (I,Q) values. The IFD would function simply as a waveform sampling device (hence the acronym **IF D**igitizer), and all of the front-end signal processing took place downstream of it.

This model has changed with the introduction of the Rev.F IFD which has the ability to carry out several billion multiply-accumulate cycles per second. This means that IF samples from multiple signals can be preprocessed entirely within the IFD and then encoded without loss onto the fixed bandwidth of its digital downlink. The new receiver modes 3 through 5 rely on this hardware capability and use a method known as "Halfband Filtering" to effectively double the downlink data rate.

3.2.7 IF Bandwidth and Dynamic Range on page 73 of the *RVP8 User's Manual* contains a detailed account of how A/D quantization noise affects the dynamic range of the IFD. Briefly, for the Rev.F A/D converter which runs at 72MHz, the contribution of A/D quantization noise within any given 1MHz interval is 72 times smaller than the total noise of the converter itself. This is an important property of all wideband sampling systems: the noise floor after processing, and hence the dynamic range, are improved by increasing the fundamental A/D sampling rate.

Normally the IFD sends 72MHz A/D samples from a single input channel directly down to the RVP8/Rx PCI card. The samples are sent at full speed in order to realize maximum reduc-tion of the final (I,Q) noise floor. But suppose we wanted to send two A/D waveforms down the same data link by interleaving the samples together. Each channel would have to be down-sampled to 36MHz in order to fit within this format, but that would cause its (I,Q) noise floor to increase by 3dB.

To avoid this, we do not create the 36MHz streams merely by discarding every other A/D sample, but rather, by passing the original 72MHz data through a halfband digital filter and then discarding every other point of this filtered A/D stream. The difference is important. Since the halfband filter has removed all of the A/D quantization noise from half of the original Nyquist interval, there will be no increase in noise density within the passband of the (I,Q) filter when the halfband stream is down sampled to 36MHz. Thus, the A/D noise that would normally have folded into the (I,Q) data at 36MHz is first removed by the halfband filter so that we're left with a 36MHz stream having *the same dynamic range* of the original 72MHz samples.

The IFD halfband filter is a 49-Tap equiripple FIR filter having 40dB of stopband rejection and 0.175dB of passband ripple. The passband extends

either from 0–16.5MHz when configured as a lowpass filter, or 19.5–36MHz when configured for highpass. The RVP8 automatically selects the correct type of filter depending on the intermediate frequency specified in the **Mb** menu. The halfband filter has linear phase and is therefore non-dispersive. This means that it is totally suitable for handling compressed pulses and other wideband Tx/Rx waveforms.

6.1.2.2 Discussion of Wide Dynamic Range Mode-4

When a two channel IFD is used as an extended dynamic range receiver there are some important decisions to make with respect to setting up the RF/IF levels that drive the IFD.

The first of these is the amount of signal level separation between the high gain and the low gain IFD inputs. There is an absolute minimum and absolute maximum channel separation that still allows the IFD to capture the full dynamic range of the receiver. If a signal level separation is made that is outside of these absolute limits valuable receiver dynamic range will be lost.

- *The absolute minimum separation* of the channels is equal to the total dynamic range of the receiver minus the dynamic range of a single channel of the IFD. Generally, the total dynamic range of the receiver is set by the LNA. For example, if we are considering a 1µ sec pulse (1MHz bandwidth), the dynamic range of the LNA may be about 105dB, and the dynamic range of a single channel of the IFD is about 84dB (from -78dBm to +6dBm). In this case, the minimum separation would be 21dB. At minimum separation, the overlap of the low gain channel and the high gain channel will be maximized, and that overlap is equal to the dynamic range of a signal channel of the IFD minus the separation. In this case, the overlap is (84dB - 21dB) = 63dB.
- *The absolute maximum separation* of the channels is simply the dynamic range of a single channel of the IFD. In the above example this would be 84dB. At maximum separation, the overlap of the low gain channel and the high gain channel is zero -- we begin using one as soon as the other has begun to saturate.

We see that there can be a large difference between the absolute minimum and maximum signal level separations; thus additional criteria must be considered to choose an optimum value that is between these diverse limits.

Choosing a proper separation value is a tradeoff of several factors. If the separation value is too low, the IFDs may end up operating very close to their noise floors. And if the separation is too high, then the overlap between the two channels is reduced which makes it difficult for the IFD to make a smooth transition as it combines the data from both channels.

Too high a separation may also result in receiver components that are not practical to build.

As a rule of thumb, channel separations in the 22–30dB range provide a good balance of the above criteria. In the case of a 1µ sec pulse this results in an overlap interval of approximately 55-63dB, which is sufficient for good IFD transitions and also leads to receiver components that are practical to build.

Once a separation value has been chosen, one must consider how to build the receiver to achieve this. The basic receiver will take the form of an LNA and a mixer followed by a splitter resulting in a low gain channel and a high gain channel. We know the gain difference in the two channels (the separation value), but we must find the actual gain to use in each channel.

If we consider the total system dynamic range as generally set by the LNA (105dB in the above example), we can estimate the minimum detectable signal input to the LNA as well as the maximum usable linear level at the IFD. If the LNA has a noise figure of 1dB and we are using a 1µ sec pulse, the minimum detectable signal at the LNA input is -113dBm, and thus the maximum signal is 105dB above this, or -8dBm. If we add to these number the gain of the LNA and the conversion loss of the mixer (and any other losses experienced through the power splitter for the low gain and high gain channels), we can use this information to determine the signal values of the components in these two channels.

For example, if the LNA has a gain of 17dB, the mixer has a conversion loss of 7dB, there is 1dB miscellaneous losses and 3dB loss in the power splitter, then the signal level at the output of the power splitter is (-113 + 18 - 7 - 1 - 3) = -106dBm for the minimum signal, and and -1dBm for the maximum signal. In the low gain channel, we need to bring the -1dBm up to the maximum input value of the IFD (+6dBm). To do this we need about 8dB of amplification (7dB plus one more deciBel to account for the anti–alias filter loss of the IFD). If we assume 25dB of channel separation, on the high gain channel we require about +33dB of amplification. Finally, this tells us that on the low gain channel, the minimum and maximum signals presented to the IFD are (-106 + 8) = -98dBm and (-1 + 8) = 7dBm. For the high gain channel, the signal levels are (-106 + 33) = -73dBm and (-1 + 33) = +32dBm. Note that as +32dBm is above the maximum input level tolerated by the IFD, the amplifier on the high gain channel must limit its output to less than +16dBm. Thus an amplifier with an output saturation value of between +10dBm and +15dBm should be used.

6.1.3 Automatic Frequency Control (AFC)

AFC is used on magnetron systems to tune the STALO to compensate for magnetron frequency drift. It is not required for Klystron systems. The STALO is typically tuned 30 or 60 MHz away from the magnetron frequency. The maximum tuning range of the AFC feedback is approximately 7MHz on each side of the center frequency. This is limited by the analog filters that are installed just before the signal and burst IF inputs on the IFD. It is important that the system's IF frequency is at least 4MHz away from any multiple of half the digital sampling frequency, that is, 18, 36, 54, or 72MHz.

The RVP8 analyzes the burst pulse samples from each pulse, and produces a running estimate of the power-weighted center frequency of the transmitted waveform. This frequency estimate is the basis of the RVP8's AFC feedback loop, whose purpose is to maintain a fixed intermediate frequency from the radar receiver.

The instantaneous frequency estimate is computed using four autocorrelation lags from each set of N b_n samples. This estimate is valid over the entire Nyquist interval (for example, 18MHz to 36MHz), but becomes noisy within 10% of each end. Since the span of the burst pulse samples is only approximately one microsecond, several hundred estimates must be averaged together to get an estimate that is accurate to several kiloHertz. Thus, the AFC feedback loop will typically have a time constant of several seconds or more.

Most of the burst pulse analysis routines, including the AFC feedback loop, are inhibited from running immediately after making a pulsewidth change. The center-of-mass calculations are held off according to the value of *Settling time (to 1%) of burst frequency estimator*, and the AFC loop is held off by the *Wait time before applying AFC* (**Mb** 4.2.7 Mb — Burst Pulse and AFC on page 142). This prevents the introduction of transients into the burst analysis algorithms each time the pulsewidth changes.

Additional information about using AFC can be found in Sections 3.2.11 IFD Analog AFC Output Voltage (Optional) on page 81, 3.4 Digital AFC Module (DAFC) on page 99, and 4.2.7 Mb — Burst Pulse and AFC on page 142.

6.1.4 Burst Pulse Tracking

The RVP8 has the ability to track the power-weighted center-of-mass of the burst pulse, and to automatically shift the trigger timing so that the pulse remains in the center of the burst analysis window of the **Pb** plot. This means that external sources of drift in the timing of the transmitted pulse (temperature, aging, etc.) will be tracked and nulled out during normal operation; so that fixed targets will remain fixed in range, and clean Tx phase measurements will always be available on every pulse.

The Burst Pulse Tracker feedback loop makes changes to the trigger timing in response to the measured position of the burst. Timing changes will generally be made only when the RVP8 is not actively acquiring data, in the same way that AFC feedback is held off for similar "quiet" times. However, if the center-of-mass has drifted more than 1/3 the width of the burst analysis window, then the timing adjustment will be made right away. Also, there will be an approximately 5ms interruption in the normal trigger sequence whenever any timing changes are made.

The Burst Pulse Tracker and AFC feedback loop are each fine-tuning servos that keep the burst pulse "centered" in time and frequency. These servos have been expanded to include a combined "Hunt Mode" that will track down a missing burst pulse when we are uncertain of both its time and frequency. This coarse-tuning mode is especially valuable for initializing the two fine-tuning servos in radar systems that drift significantly with time and temperature.

When the radar transmitter is *On* but the burst pulse is missing, it may be because either of the following have happened:

- It is misplaced in time, that is, the Tx pulse is outside of the window displayed in the **Pb** plotting command. In this case, the trigger timing needs to be changed in order to bring the center of the pulse back to the center of the window.
- It is mistuned in frequency, that is, the AFC feedback is incorrect and has caused the burst frequency to fall outside of the passband of the RVP8 anti-alias filters. In this case the AFC (or DAFC) needs to be changed so that proper tuning is restored.

The Hunt Mode performs a 2-dimensional search in time and frequency to locate the burst; searching across a ±20µsec time window, and across the entire AFC span. If a valid Tx pulse (that is, meeting the minimum power requirement) can be found anywhere within those intervals then the Burst Pulse Tracker and AFC loops will be initialized with the time and frequency values that were discovered. The fine servos then commence running with a good burst signal starting from those initial points.

Depending on how the hunting process has been configured in the **Mb** menu, the whole procedure may take several seconds to complete. The RVP8's host computer interface remains completely functional during this time, but any acquired data would certainly be questionable. GPARM status bits in word #55 indicate when the hunt procedure is running, and whether it has completed successfully. The BPHUNT (7.26 Hunt for Burst Pulse (BPHUNT) on page 345) opcode allows the host computer to initiate Hunt Mode when it knows or can sense that a burst pulse should be present

6.1.5 Interference Filter

The interference filter is an optional processing step that can be applied to the raw (*I,Q*) samples that emerge from the FIR filter chips. The intention of the filter is to remove strong but sporadic interfering signals that are occasionally received from nearby man-made sources. The technique relies on the statistics of such interference being noticeably different from that of weather.

For each range bin at which (*I,Q*) data are available, the interference filter algorithm uses the received power (in deciBels) from the three most recent pulses:

$$P_{n-2}\ ,\ P_{n-1}\ ,\ \text{and}\ P_n$$

where:

$$P_n\ =\ 10\log_{10}(I_n^2 + Q_n^2)$$

If the three pulse powers have the property that:

$$\left|P_{n-1} - P_{n-2}\right| < C_1 \ \text{and}\ \left|P_n - P_{n-1}\right| > C_2$$

(Alg.1)

then (I_n, Q_n) is replaced by (I_{n-1}, Q_{n-1}) . Here C_1 and C_2 are constants that can be tuned by the user to match the type of interference that is anticipated, and the error rates that can be tolerated. For certain environments it may be the case that good results can be obtained with $C_1 = C_2$; but the RVP8 does not force that restriction.

This 3-pulse algorithm is only intended to remove interference that arrives on isolated pulses, and for which there are at least two clear pulses in between. Interference that tends to arrive in bursts will not be rejected.

Two variations on the fundamental algorithm are also defined. The CFGINTF command (7.23 Configure Interference Filter (CFGINTF) on page 343) allows you to choose which of these algorithms to use, and to tune the two threshold constants. You may also do this directly from the **Mp** setup menu (4.2.2 Mp — Processing Options on page 122).

$$|P_{n-1} - P_{n-2}| < C_1 \text{ and } P_n - P_{n-1} > C_2$$

(Alg. 2)

$$|P_{n-1} - P_{n-2}| < C_1 \text{ and } P_n - LinAvg(P_{n-1}, P_{n-2}) > C_2$$

(Alg. 3)

Where *LinAvg()* denotes the deciBel value of the linear average of the two deciBel powers. The Alg.2 and Alg.3 algorithms also include the receiver noise level(s) as part of their decision criteria. Whenever power levels are intercompared in the algorithms, any power that is less than the noise level is first set equal to that noise level. This makes the filters much more robust and properly tunable, so that interference is more successfully rejected on top of blank receiver noise.

Optimum values for C_1 and C_2 will vary from site to site, but some guidance can be obtained using numerical simulations. The results shown below were obtained when the algorithms were applied to realistic weather time series having a spectrum width = 0.1 (Nyquist), SNR = +10dB, and an intermittent additive interference signal that was 16dB stronger than the weather. The interference arrived in isolated single pulses with a probability of 2%.

Performance of the three algorithms is summarized in the first three columns of Table 17 on page 214, for which C_1 and C_2 have the common value shown. The fourth column also uses Algorithm #3, but with the value of C_1 raised by 2dB. The "Missed" rate is defined as the percentage of interference points that manage to get through the filtering process without being removed. The "False" (false alarm) rate is the percentage of non-interference points that are incorrectly modified when they should have been left alone.

Table 17 Algorithm Results for +16dB Interference

C1,C2	Alg.1 Missed/False		Alg.2 Missed/False		Alg.3 Missed/False		Alg.3, C1+=2dB Missed/False	
6.0dB	17.8%	10.91%	17.8%	4.06%	17.8%	3.48%	10.3%	4.15%
8.0dB	10.5%	6.57%	10.5%	2.42%	10.4%	1.71%	6.1%	1.92%
9.0dB	8.5%	5.09%	8.5%	1.81%	8.3%	1.16%	5.4%	1.28%
10.0dB	7.3%	4.01%	7.3%	1.42%	7.5%	0.79%	5.4%	0.85%
11.0dB	8.9%	3.14%	8.9%	1.06%	8.3%	0.51%	6.5%	0.54%
12.0dB	11.6%	2.53%	11.6%	0.85%	11.3%	0.33%	9.9%	0.35%
13.0dB	17.0%	2.07%	17.0%	0.67%	16.3%	0.22%	15.3%	0.23%
14.0dB	23.5%	1.70%	23.5%	0.54%	22.4%	0.14%	21.6%	0.15%
16.0dB	39.2%	1.21%	39.2%	0.35%	39.6%	0.06%	38.9%	0.06%
20.0dB	67.3%	0.65%	67.3%	0.14%	72.5%	0.01%	72.4%	0.01%

It is important to minimize both types of errors. If too much interference is missed, then the filter is not doing an adequate job of cleaning up the received signal. If the false alarm rate is too high, then background damage is done at all times and the overall signal quality (especially sub-clutter visibility) may be compromised. We suggest that you try to keep the false alarm rate fairly low, perhaps below 1%; and then let the missed percentage fall where it may.

To summarize the numerical results in Table 17 on page 214:

- The "Missed" rates of Alg.1 and Alg.2 are identical, but the "False" rate of Alg.1 is much higher. Alg.1 clearly does not perform as well for additive interference, but it is included in the suite for historical reasons.
- The "Missed" error rate for Alg.3 is nearly identical to that of Alg.2, but Alg.3 has a significantly lower false alarm rate. This is because of the somewhat improved statistics that result when the linear mean of P_{n-2} and P_{n-1} is used in the second comparison, rather than just P_{n-1} by itself. We recommend that Alg.3 generally be chosen in preference to the other two.
- Alg.3 can be further tuned by allowing the two constants to differ. For example, by raising C_1 slightly above C_2 (fourth column), we can trade off a decrease in the "Missed" rate for an increase in the "False" rate. Lowering C_1 would have the opposite effect.

Keep in mind that optimum tuning will depend on the type of interference you are trying to remove. In the previous example, where the interfering signal is only 16dB stronger than the weather, there was a close tradeoff between the "Missed" and "False" error rates. However, Table 18 on page

215 shows the results that would be obtained if the interference dominates by 26db.

Table 18 Algorithm Results for +26dB Interference

C1,C2	Alg.1 Missed/False		Alg.2 Missed/False		Alg.3 Missed/False		Alg.3, C2+=5dB Missed/False	
6.0dB	17.8%	10.75%	17.8%	3.95%	17.8%	3.44%	17.8%	0.34%
8.0dB	9.9%	6.48%	9.9%	2.31%	9.9%	1.68%	9.9%	0.15%
9.0dB	7.4%	4.99%	7.4%	1.75%	7.4%	1.14%	7.4%	0.10%
10.0dB	5.9%	3.91%	5.9%	1.36%	5.9%	0.76%	5.9%	0.06%
11.0dB	4.8%	3.06%	4.8%	1.06%	4.8%	0.50%	4.8%	0.04%
12.0dB	3.2%	2.37%	3.2%	0.83%	3.2%	0.33%	3.2%	0.03%
13.0dB	2.6%	1.83%	2.6%	0.62%	2.6%	0.20%	2.8%	0.01%
14.0dB	1.9%	1.45%	1.9%	0.50%	1.9%	0.12%	2.6%	0.01%
16.0dB	1.3%	0.90%	1.3%	0.30%	1.3%	0.05%	5.8%	0.00%
20.0dB	3.1%	0.39%	3.1%	0.12%	2.0%	0.01%	31.5%	0.00%

Notice that we can now re-tune the constants and operate with $C_1 = 13dB$ and $C_2 = 18dB$ (fourth column); which yields a low 2.8% "Missed" rate, and an extremely low 0.01% false alarm rate. Since the false alarm rate is (approximately) independent of the interference power, these filter settings would leave all "clean" weather virtually untouched, that is, we would have a very safe filter that is intended only to remove fairly strong interference. Such a filter could be left running at all times without too much worry about side effects.

6.1.6 Large-Signal Linearization

The RVP8 is able to recover the signal power of targets that saturate the IF-Input A/D converter by as much as 4–6 deciBels. This is possible because an overdriven IF waveform still spends some of its time in the valid range of the converter, and thus, it is still possible to deduce information about the signal.

Figure 32 on page 216 shows actual signal generator test measurements with normal A/D saturation (lower line), and with the extrapolation algorithms turned on (upper line). The high-end linear range begins to roll off at approximately +10dBm versus +5dBm, and thus has been extended by 5dB.

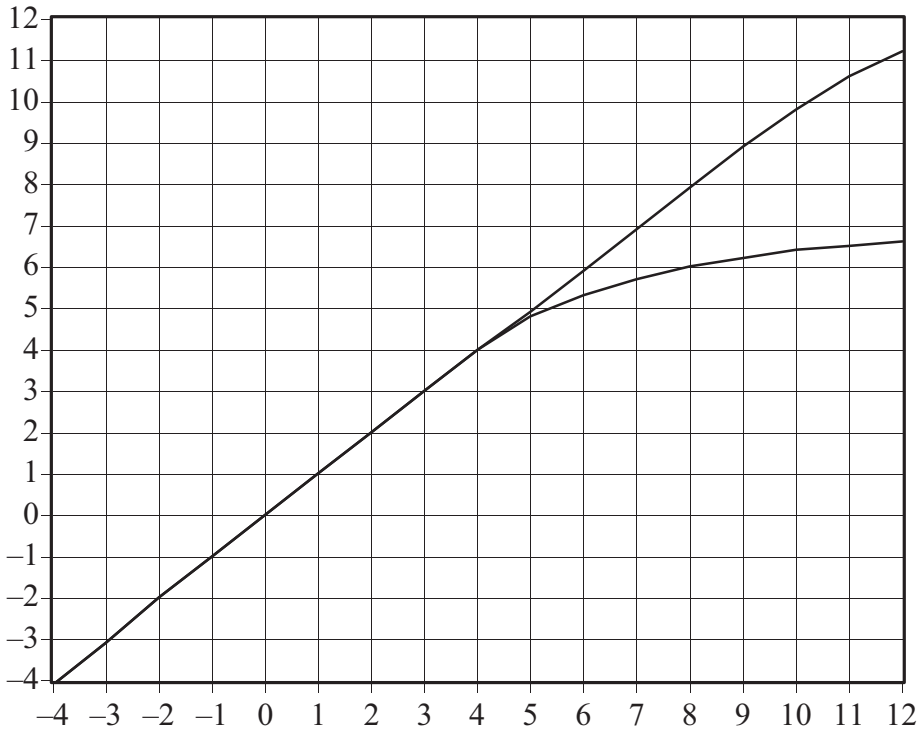


Figure 32 Linearization of Saturated Signals Above +4.5dBm (Rev B/C IFD)

The roll off starts at +4.5 dBm for the Rev. B&C IFD, and at +6 dBm for the Rev. D.

6.1.7 Correction for Tx Power Fluctuations

The RVP8 can perform pulse-to-pulse amplitude correction of the digital (I,Q) data stream based on the amplitude of the Burst/COHO input. The technique computes a (real valued) correction factor at each pulse by dividing the mean amplitude of the burst by the instantaneous amplitude of the burst. The (I,Q) data for that pulse are then multiplied by this scale factor to obtain corrected time series. The amplitude correction is applied after the Linearized Saturation Headroom correction.

The mean burst amplitude is computed by an exponential average whose (*1/e*) time constant is selected as a number of pulses (See 4.2.2 Mp — Processing Options on page 122). A short time constant will settle faster, but will not be as thorough in removing amplitude variations (since the mean itself will be varying). Longer time constants do a better job, but will require a second or two before valid data is available when the transmitter

is first turned on. The default value of 70 will give excellent results in almost all cases.

Whenever the RVP8 enters a new internal processing mode (time series, FFT, PPP, etc.), the burst power estimator is reinitialized from the level of the first pulse encountered, and an additional pipeline delay is introduced to allow the estimator to completely settle. Thus, valid corrected data are produced even when the RVP8 is alternating rapidly between different data acquisition tasks, for example, in a multi-function ASCOPE display. The additional pipeline delay will not affect the high-speed performance when the RVP8 runs continuously in any single mode.

For amplitude correction to be applied, the instantaneous Burst/COHO signal level must exceed the minimum valid burst power specified in the "**Mb**" setup section. If that level is not met, for example, if the transmitter is turned off, then no correction is performed. Thus, the amplitude correction feature conveniently "gets out of the way" when receiver-only tests are being performed.

The maximum correction that will ever be applied is ±5dB. If the burst power in a given pulse is more than 5dB above the mean, or less than 5dB below it, then the correction is clamped at those limits. The power variation of a typical transmitter will easily be contained within this interval (it is typically less than 0.3dB).

Instantaneous amplitude correction is a unique feature of the RVP8 digital receiver. Bench tests with a signal generator reveal that an amplitude modulated waveform having 2.0dB of pulse-to-pulse variation is reduced to less than 0.02dB RMS of (I,Q) variation after applying the amplitude correction.

6.2 Time Series (I and Q) Signal Processing

6.2.1 Time Series Processing Overview

This section describes the processing of the radar time series data (also called linear "video" or "I" and "Q") to obtain the meteorologically significant "moment" parameters: reflectivity, total power, velocity, width, signal quality index, clutter power correction, and optional polarization variables.

Recall that the time series synthesized by the FIR filter consist of an array of complex numbers:

$$s_m = [I_m + jQ_m] \text{ for } m = 1, 2, 3, \ldots, M$$

where "j" is - $1^{1/2}$. The time series, are the starting point for all calculations performed within the RVP8. There are several excellent references on the details of I and Q processing. The reader is referred to Doviak and Zrnic's text on the subject. The top part of Figure 33 on page 219 shows I and Q values for a simulated time series using the ascope utility.

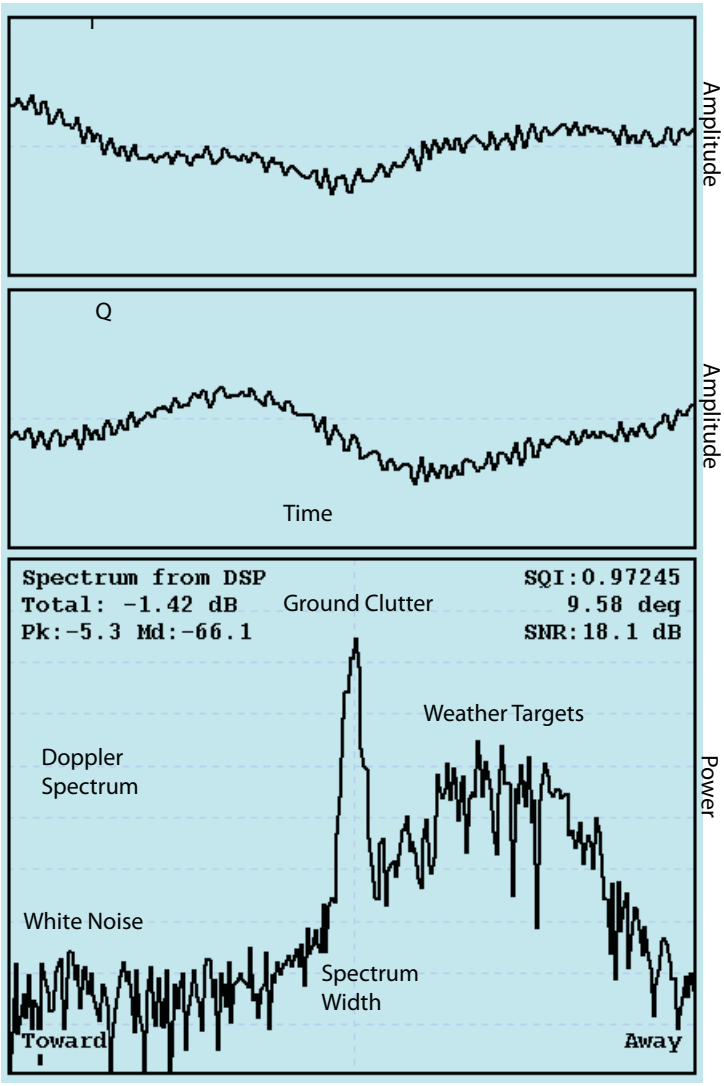
There are two broad categories of time series signal processing:

- Time Domain Processing using the I and Q samples directly to calculate "autocorrelations" and then using the autocorrelations to compute the moments. This is used by many systems since the algorithms are very efficient requiring minimal storage and computational power. However, time domain algorithms are generally not adaptive or very flexible.
- Frequency Domain Processing using the I and Q samples to calculate a Doppler power spectrum and then applying algorithms, such as clutter filtering or 2nd trip echo filtering/extraction, in the frequency domain. The Doppler spectrum is then inverted to obtain the autocorrelation functions and these are used to calculate the moments. The frequency domain is well suited to more complex adaptive algorithms, that is, where the processing algorithm is optimized for the data.

The RVP8 supports the concept of "major modes" or processing modes to process the time series. Currently the following major modes are supported by Vaisala:

- DFT/FFT Mode is a frequency domain approach which is used for most operational processing applications. There are a variety of clutter filtering options, including the GMAP algorithms (Gaussian Model Adaptive Processing).
- Pulse Pair Processing or PPP Mode is a time domain approach that is used primarily for dual polarization applications.
- Random Phase Mode or RPHASE is a frequency domain approach similar to the DFT/FFT, except that filtering and extraction of both the first and second trip echoes is supported.
- Batch Mode during which a small batch of low PRF pulses is transmitted (for example, for 0.1 degree of scanning) followed by a large batch of higher PRF pulses (for example, for 0.9 degrees of scanning) to determine which ranges are likely contaminated by second trip echo. This was developed to support a US WSR88D legacy requirement. It is not supported in Vaisala's IRIS software.

The time and frequency domain approaches are described in the sections below.



Time series of I and Q and the corresponding Doppler power spectrum obtainded from the ascope utility using the built-in simulator. The Doppler spectrum displays the radial velocity on the X-axis over the unambiguous range or "Nyquist" interval and the power in dB relative to saturation on the y-axis.

Note that for illustration, this example is based on 256 time series points (one point per pulse) which yields 256 spectrum components. This is more than is usually processed in actual operation.

The spectrum shows the three major components of the Doppler spectrum:

* White noise.
* Ground clutter at zero radial velocity
* A spectrum of the weather targets having a Gaussian shape characterized by the weather power, mean velocity, and width (standard deviation), that is, the spectrum moments.

Figure 33 Example of Time Series and Doppler Power Spectrum

6.2.2 Frequency Domain Processing- Doppler Power Spectrum

The Doppler power spectrum, or simply the "Doppler spectrum", is the easiest way to visualize the meteorological information content of the time series. The bottom part of Figure 33 on page 219 shows an example of a Doppler power spectrum for the time series shown in the upper part of the figure. The figure above shows the various components of the Doppler spectrum, that is, typically there is white noise, weather signal and ground clutter. Other types of targets such as sea clutter, birds, insects, aircraft, surface traffic, second trip echo, etc. may also be present.

The "Doppler power spectrum" is obtained by taking the magnitude squared of the input time series, that is, for a continuous time series,

$$S(\omega) = \left| f\{s(t)\} \right|^2$$

Here S denotes the power spectrum as a function of frequency ω, and f denotes the Fourier transform of the continuous complex time series s(t). The Doppler power spectrum is real-valued since it is the magnitude squared of the complex Fourier transform of s(t).

In practice a pulsed radar operates with discrete rather than continuous time series, that is, there is an I and Q value for each range bin for each pulse. In this case we use the discrete Fourier transform or DFT to calculate the discrete power spectrum. Note that in the special case when we have 2n input time series samples (for example, 16, 32, 64, 128, ...), we use the fast Fourier transform algorithm (FFT), so called because it is significantly faster than the full DFT.

The DFT has the form:

$$S_k = \left| DFT_k\{w_m s_m\} \right|^2 = \left| \sum_{m=0}^{M} w_m s_m e^{-j(2(\pi/M))mk} \right|^2$$

Typically a weighting function or "window" w_m is applied to the input time series s_m to mitigate the effect of the DFT assumption of periodic time series. The RVP8 supports different windows such as the Hamming, Blackman, Von Han, Exact Blackman and of course the rectangular window for which all spectral components are weighted equally. The typical form of a spectrum window is shown in the figure below which illustrates how the edge points of the time series are de–emphasized and the center points are over emphasized. The dashed line would correspond

to the rectangular window. Note that the "gain" of the window is set to preserve the total power.

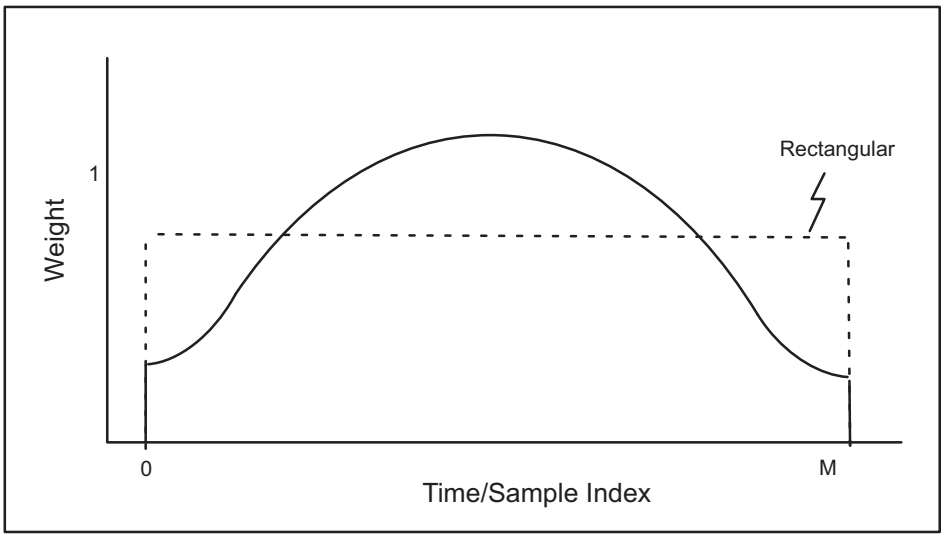


Figure 34 Typical Form of Time Series Window

Even though the window gain can be adjusted to conserve the total power, there is an effective reduction in the number of samples which increases the variance (or uncertainty) of the moment estimates. For example the variance of the total power is greater when computed from a spectrum with Blackman weighting as compared to using a rectangular window. This is because there are effectively fewer samples because of the de-emphasis of the end points. This is a negative side to using a window.

The DFT of the window itself is known as its impulse response which shows all of the frequencies that are generated by the window itself. A generic example is shown in Figure 35 on page 222 below which illustrates that these "side lobe" frequencies can have substantial power. This is not a problem for weather signals alone, but if there is strong clutter mixed in, then the side lobe power from the clutter can obscure the weaker weather signals. The rectangular window has the worst sidelobes, but the narrowest window width. However, the rectangular window provides the lowest variance estimates of the moment parameters (in the absence of clutter. More "aggressive" windows have lower side lobe power at the expense of a broader impulse response and an increased variance of the moment estimates.

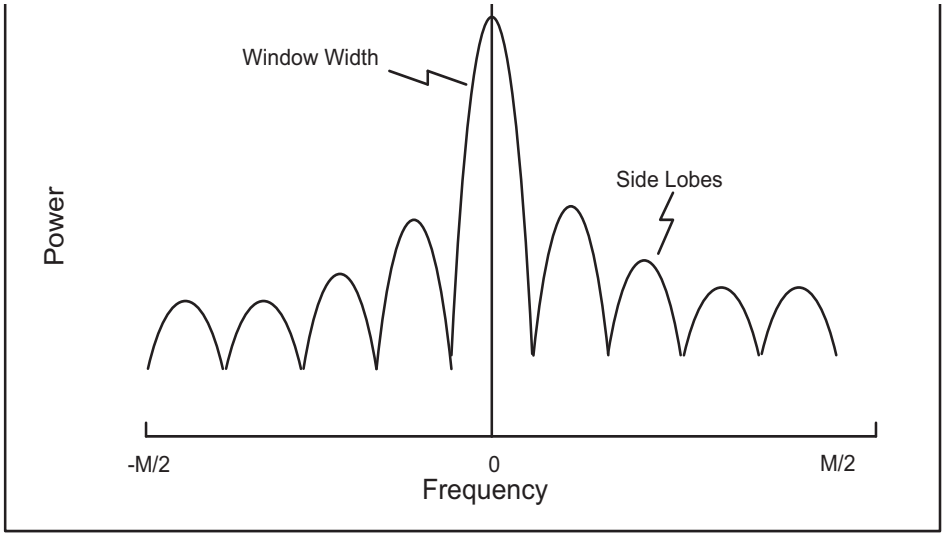


Figure 35 Impulse Response of Typical Window

So in summary of the DFT approach and spectrum windows:

- When the clutter is strong, an aggressive spectrum window is required to contain the clutter power so that the side lobes of the window do not mask the weather targets. The side lobe levels of some common windows are:

Rectangular	12 dB
Hamming	40 dB
Blackman	55 dB

- More aggressive windows typically have a wider impulse response. This effectively increases the spectrum width. Rectangular is narrow, Hamming intermediate and Blackman the widest.
- Windows effectively reduce the number of samples resulting in higher variance moment estimates. Rectangular is the best case, Hamming is intermediate and Blackman provides the highest variance moment estimates.

These facts suggest the best approach is to use the least aggressive window possible in order to contain the clutter power that is actually present, that is, an adaptive approach is the best.

6.2.3 Autocorrelations

The final spectrum moment calculation (for total power or SNR, mean velocity and spectrum width) in all processing modes is based on autocorrelation moment estimation techniques. Typically the first three lags are calculated, denoted as R_0, R_1 and R_2. However, there are two ways to calculate these, that is, time domain or frequency domain calculation. In the PPP mode for dual polarization, the autocorrelations are computed directly in the time domain while in the DFT mode, they are computed by taking the inverse DFT the Doppler power spectrum in the frequency domain. Note that only the first three terms need be calculated in the inverse DFT case. The time domain and frequency domain techniques are nearly identical except that the method of taking the inverse DFT of the power spectrum relies on the assumption that the time series is periodic. Another difference is that for time domain calculation only a rectangular weighting is used.

The time domain calculation of the autocorrelations and the corresponding physical models are:

Parameter and Definition	**Physical model**
$T_0 = \frac{1}{M}\sum_{n=1}^{M} s_n {s_n}^*$	$g^r g^t (S+C)+N$
$R_0 = \frac{1}{M}\sum_{n=1}^{M} s'_n {s'_n}^*$	$g^r g^t S+N$
$R_1 = \frac{1}{M-1}\sum_{n=1}^{M-1} s'_n {s'_{n+1}}^*$	$g^r g^t S\, e^{j\pi V' - \pi^2 W^2/2}$
$R_2 = \frac{1}{M-2}\sum_{n=1}^{M-2} s'_n {s'_{n+2}}^*$	$g^r g^t S\, e^{j2\pi V' - 2\pi^2 W^2}$

where M is the number of pulses in the time average. Here, s' denotes the clutter-filtered time series, s denotes the original unfiltered time series and the * denotes a complex conjugate. g^r and g^t represent the transmitter and receiver gains, that is, their product represents the total system gain. Since the RVP8 is a linear receiver, there is a single gain number that relates the measured autocorrelation magnitude to the absolute received power. However, since many of the algorithms do not require absolute calibration of the power, the gain terms will be ignored in the discussion of these. T_o for the unfiltered time series is proportional to the sum of the

meteorological signal S, the clutter power C and the noise power N. R_0 is equal to the sum of the meteorological signal S and noise power N which is measured directly on the RVP8 by periodic noise sampling. T_o and R_0 are used for calculating the dBZ values- the equivalent radar reflectivity factor which is a calibrated measurement. The physical models for R_0, R_1 and R_2 correspond to a Gaussian weather signal and white noise as shown in Figure 33 on page 219. W is thespectrum width and V' the mean velocity, both for the normalized Nyquist interval on [-1 to 1].

The autocorrelation lags above and the corresponding physical models have five unknowns: N, S, C, V', W. Because the R1 and R2 lags are complex, this yields, effectively, five equations in five unknowns using the constraint provided by the argument of R1. This closed system of equations can be solved for the unknowns which is the basis for calculating the moments from the autocorrelations.

6.2.4 Angle Synchronization

The exact value of *M* that is used for each time average will generally be the "Sample Size" that is selected by the SOPRM command (See 7.3 Setup Operating Parameters (SOPRM) on page 279). However, when the RVP8 is in PPP mode and antenna angle synchronization is enabled, the actual number of pulses used may be limited by the number that fit within each ray's angular limits at the current antenna scan rate. The value of *M* will never be greater than the SOPRM Sample Size, but it may sometimes be less. For example, at 1KHz PRF, 20º/sec scan rate, 1º ray synchronization, and a Sample Size of 80, there will be 50 pulses used for each ray (not 80). Note, however, that the number of pulses used in the "batched" (non-PPP) modes will always be exactly equal to the Sample Size, since those modes are allowed to use overlapping pulses.

6.2.5 Clutter Filtering Approaches

Each major mode implements clutter filtering as follows:

- DFT Mode uses frequency domain clutter filters, including GMAP.
- PPP Mode is used only for dual polarization. For efficiency, PPP computations are performed using DFT techniques that are algebraically equivalent to the traditional time-domain algorithms. As such, the fixed width and variable width spectral clutter filters can be used in PPP mode.
- Random Phase Mode uses frequency domain clutter filters.
- Batch Mode uses a simple DC removal for the small batch clutter filter. The high PRF large batch is then processed using the DFT mode.

In previous versions of Vaisala processors, an IIR (infinite implodes response) filter was offered. The IIR filter, while requiring minimal storage and computation, has three major drawbacks:

- The infinite impulse response requires a settling time when a transient occurs such as a PRF change, or a spike clutter target. During the settling time, the transient response degrades the performance of the filter.
- The filter is fixed width in the Nyquist interval. This means that it may be sufficiently wide to remove moderate or weak clutter, but may not be wide enough to remove all of the clutter when the clutter power is very strong and consequently wider in the Nyquist interval. This causes operators to select wider filters than necessary so that strongest clutter is adequately removed.
- The filter does significant damage to overlapped (zero velocity) weather signals, that is, these will be significantly attenuated by the filter.

With the advent of the high speed processors such as the RVP8, there is sufficient storage and computational power to implement frequency domain filters that, in some cases, are adaptive. Because of the superiority of these filters, the legacy time domain IIR approach is no longer used in the RVP8. The only mode that uses time domain filtering is the Batch mode for the low PRF pulses (subtraction of the average I and Q to remove the DC component).

The various frequency domain filters available in the RVP8 are configured using the **mf** setup command (4.2.3 Mf — Clutter Filters on page 127). These are:

- Type 0: Fixed width filters with interpolation
- Type 1:Variable width single slope adaptive processing
- Type 2: Gaussian model adaptive processing (GMAP)

These filters are described in in the sections detail below.

6.2.5.1 Fixed Width Clutter Filters

This filter, illustrated in Figure 36 on page 226, removes a specified number of spectrum components (5 in the example) and then interpolates across the gap using the minimum of a specified number of "edge points" (2 in the example) to anchor the interpolation at each end of the gap. This is a fairly simple legacy approach that uses interpolation to repair the damage caused by the removal of components.

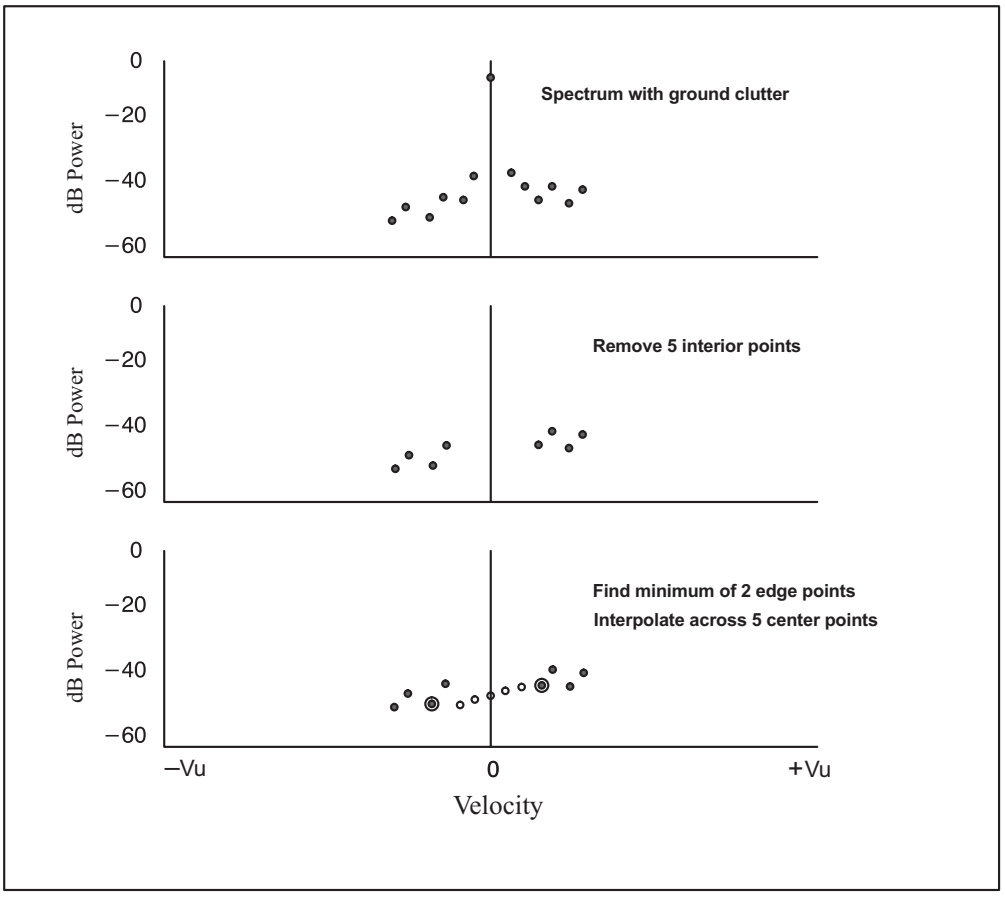


Figure 36 Example of Fixed Width

This procedure attempts to preserve the noise level and/or overlapped weather targets. The result is that more accurate estimates of dBZ are obtained. In extreme cases when the weather spectrum is very narrow, there can still be some attenuation of weather of a broad filter is selected.

6.2.5.2 Variable Width Clutter Filter

This is similar in many ways to the fixed width filter except that the algorithm attempts to extend the boundary of the clutter by determining which is the first component outside the clutter region to increase in power. The filter is illustrated in the figure below.

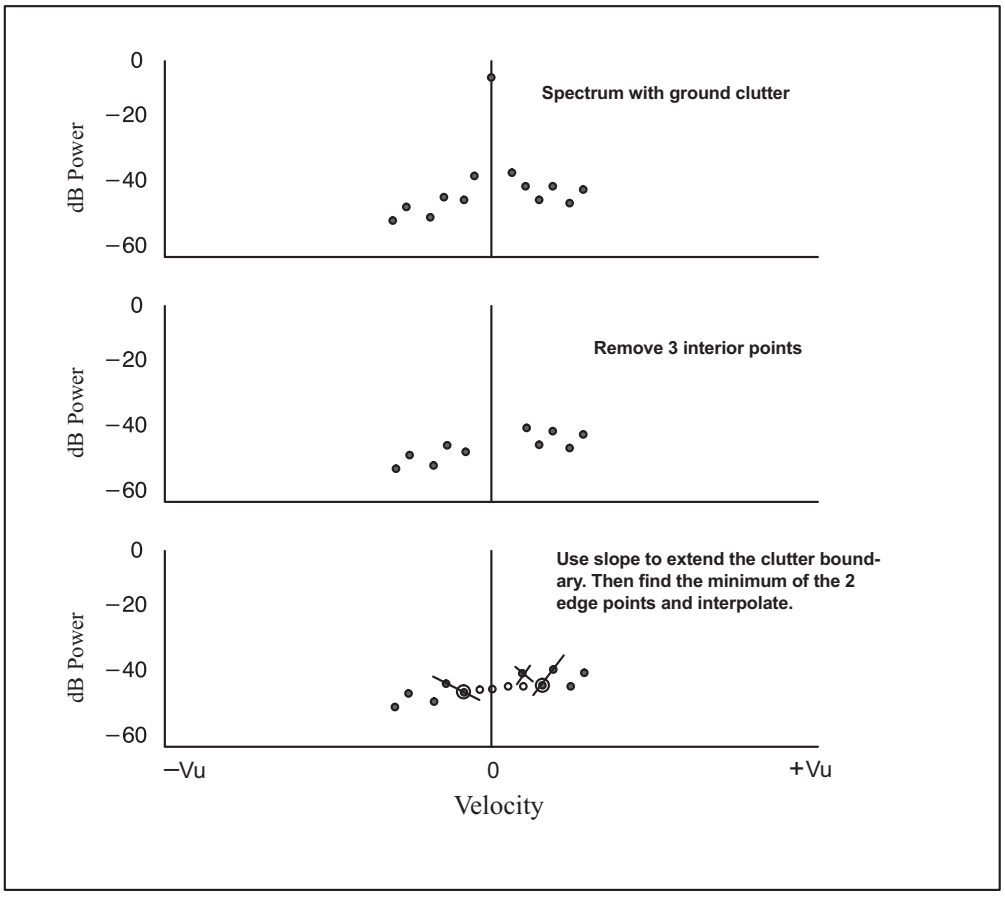


Figure 37 Variable Width Clutter Filter

In the example above, the minimum number of points to reject is set to 3. The filter starts at zero velocity and checks the slope to determine the point at which the power starts to increase. In the example, this results in the filter being extended by one point on the right. Note that there is a selectable maximum number of points that the filter will "hunt". The use of the edge points for interpolation is identical to the fixed width case.

This filter allows users to specify a narrower nominal filter than the fixed width case and then when the clutter is strong, this width is extended by the algorithm (the "hunt"). The interpolation attempts to preserve any overlapped clutter and weather.

6.2.5.3 Gaussian Model Adaptive Processing (GMAP)

GMAP is a new adaptive technique developed at Vaisala that is possible on a high-speed processor such as the RVP8. GMAP has the following advantages as compared to fixed width frequency domain filters or time domain filtering such as the IIR approach:

- The width adapts in the frequency domain to adjust for the effects of PRF, number of samples and the absolute amplitude of the clutter power. This means that minimal operator intervention is required to set the filter.
- If there is no clutter present, then GMAP does little or no filtering.
- GMAP repairs the damage to overlapped (near zero velocity) weather targets.
- The DFT window is determined automatically to be the least aggressive possible to remove the clutter. This reduces the variance of the moment estimates.

The GMAP algorithm is described below.

GMAP Model Assumptions

GMAP makes several assumptions about the model for clutter, weather and noise, that is,

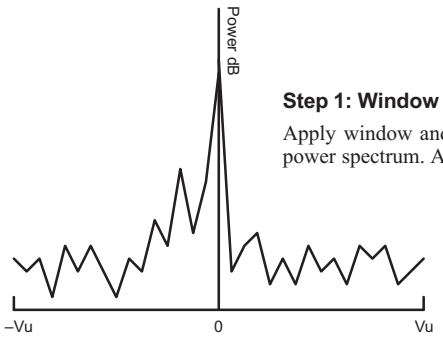
- The spectrum width of the weather signal is greater than that of the clutter. This is a fundamental assumption required of all Doppler clutter filters.
- The Doppler spectrum consists of ground clutter, a single weather target and noise. Bi-modal weather targets, aircraft or birds mixed with weather would violate this assumption.
- The width of the clutter is approximately known. This is determined primarily by the scan speed and to a lesser extent by the climatology of the local clutter targets. The assumed width is used to determine how many interior clutter points are removed.
- The shape of the clutter is approximately Gaussian. This shape is used to calculate how many interior clutter points are removed.
- The shape of the weather is approximately Gaussian. This shape is used to reconstruct filtered points in overlapped weather.

GMAP Algorithm Steps

The steps used to implement the GMAP approach are shown schematically in Figure 38 on page 230 and summarized below.

- **Step 1: Window and DFT**

  First a Hamming window weighting function is applied to the IQ values and a discrete Fourier transform (DFT) is then performed. This provides better spectrum resolution than a fast Fourier Transform (FFT) which requires that the number of IQ values be a power of 2. Note that if the requested number of samples is exactly a power of 2, then an FFT is used.

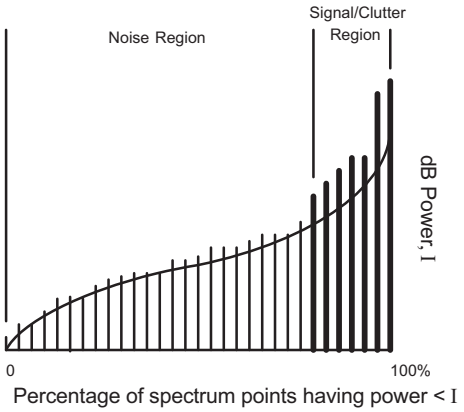


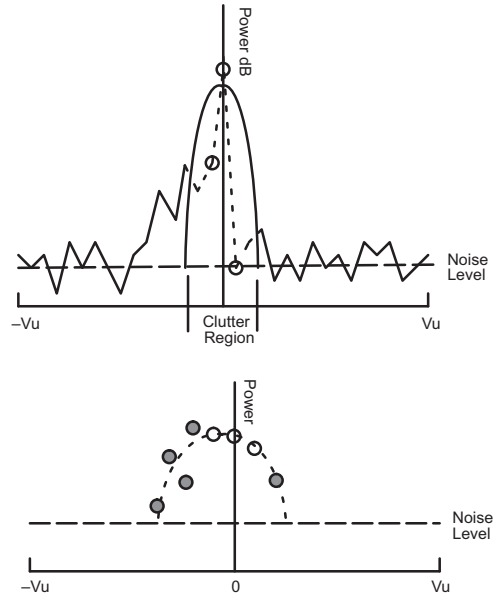
Step 1: Window and DFT

Apply window and DFT the input time series to obtain the Doppler power spectrum. A Hamming window is used for the first trial.

Step 2 (Optional): Dynamic noise power

If the noise level is not known, or if GMAP is recalculated using the Blackman window for CSR>40 dB, then this step is performed. Re-organize the spectrum components in ascending order of intensity. The theoretical relationship for noise is the curved line. The sum of the power in the range 5% to 40% is calculated. This is used to determine the noise level by comparing with the sum value corresponding to the theoretical curve. Next, the power is summed beyond the 40% point for both the actual and theoretical rank spectra. The point where the actual power sum exceeds the theoretical value by 2 dB determines the boundary between the noise region and the signal/clutter region.





Step 3: Remove clutter points

Use the total power of the three central spectrum points (indicated by the three open circles) to fit a Gaussian having the selected nominal spectrum width in m/s (a function of the number of spectrum samples, PRF and wavelength). The points within the intersection of the Gaussian clutter and the noise level (the "Clutter Region") are discarded (indicated by the dashed lines).

Step 4: Replace clutter points

Dynamic Noise Case: Using the components which have been determined to be neither clutter nor noise (indicated by the filled circles), fit a Gaussian and fill-in the clutter points that were removed in the previous step (indicated by the open circles). Then re-fit the Gaussian with the replacement values inserted. Repeat the iteration until the computed power does not change by more than 0.2dB AND the velocity does not change by more than 0.5% of the Nyquist velocity.

Fixed Noise Case: Similar except the spectrum points that are larger than the noise level are used.

Step 5: Recompute GMAP with optimal window

Determine if the optimal window was used based on the clutter-to-signal ratio (CSR)

IF CSR > 40 dB repeat GMAP using a Blackman window and dynamic noise calculation.

IF CSR > 20 dB repeat GMAP using a Blackman window. Then if CSR>25dB use Blackman results.

IF CSR < 2.5 dB repeat GMAP using a rectangular window. Then if CSR < 1 dB use rectangular results.

ELSE accept the Hamming window result.

Figure 38 GMAP Algorithm Steps

- As mentioned in 6.2.2 Frequency Domain Processing- Doppler Power Spectrum on page 220, when there is no or very little clutter, use of a rectangular weighting function leads to the lowest-variance estimates of intensity, mean velocity and spectrum width. When there is a very large amount of clutter, then the aggressive Blackman window is required to reduce the "spill-over" of power from the clutter target into the sidelobes of the impulse response function. The Hamming window is used as the first guess. After the first pass GMAP analysis is complete, a decision is made to either accept the Hamming results, or recalculate for either rectangular or Blackman depending on the clutter-to-signal ratio (CSR) computed from the Hamming analysis. The recalculated results are then checked to determine whether to use these or the original Hamming result (see Figure 38 on page 230 for details).
- **Step 2: Determine the noise power**

  In general, the spectrum noise power is known from periodic noise power measurements. Since the receiver is linear and requires no STC or AGC, the noise power is well–behaved at all ranges. The only time that the spectrum noise power will differ from the measured noise power is for very strong clutter targets. In this case, the clutter contributes power to all frequencies, essentially increasing the spectrum noise level. This occurs for two reasons: 1) In the presence of very strong clutter, even a small amount of phase noise causes the spectrum noise level to increase, and 2) There is significant power that occurs in the window side-lobes. For a Hamming window, the window side lobes are down by 40 dB from the peak at zero velocity. Thus 50 dB clutter targets will have spectrum noise that is dominated by the window sidelobes in the Hamming case. The more aggressive Blackman window has approximately 55 dB window sidelobes at the expense of having a wider impulse response and larger negative effect on the variance of the estimates.
- When the noise power is not known, it is optionally computed using a dynamic approach similar to that of Hildebrand and Sekhon (1974). The Doppler spectrum components are first sorted in order of their power. As shown in Figure 38 on page 230, the sorting places the weakest component on the left and the strongest component on the right. The vertical axis is the power of the component. The horizontal axis is the percentage of components that have power less than the y-axis power value. Plotted on a dB scale, Poisson distributed noise has a distinct shape, as shown by the curved line in Figure 38 on page 230. This shape shows a strong singularity at the left associated with taking the log of numbers near zero, and a strong maximum at the right where there is always a finite probability that a few components will have extremely large values.

- There are generally two regions: a noise region on the left (weaker power) and a signal/clutter region on the right (stronger power). The noise level and the transition between these two regions is determined by first summing the power in the range 5% to 40%. This sum is used to determine the noise level by comparing with the sum value corresponding to the theoretical curve. Next, the power is summed beyond the 40% point for both the actual and theoretical rank spectra. The point where the actual power sum exceeds the theoretical value by 2 dB determines the boundary between the noise region and the signal/clutter region.
- Finally there are two outputs from this step: a spectrum noise level and a list of components that are either signal or clutter
- **Step 3: Remove the clutter points**

  The inputs for this step are the Doppler power spectrum, the assumed clutter width in m/s and the noise level, either known from noise measurement or optionally calculated from the previous step. First the power in the three central spectrum components is summed (DC ±1 component) and compared to the power that would be in the three central components of a normalized Gaussian spectrum having the specified clutter width and discretized in the identical manner. This serves as a basis for normalizing the power in the Gaussian to the observed power. The Gaussian is extended down to the noise level and all spectral components that fall within the Gaussian curve are removed. The power in the components that are removed is the "clutter power".
- A subtle point is the use of the three central points to do the power normalization of the actual vs the idealized spectrum of clutter. This is more robust than using a single point since for some realizations of clutter targets viewed with a scanning antenna, the DC component is not necessarily the maximum. Averaging over the three central components is a more robust way to characterize the clutter power.
- The very substantial algorithmic work that has been done thus far is to eliminate the proper number of central points. The operator only has to specify a nominal clutter width in m/s. This means that the operator does not need to consider the PRF, wavelength or number of spectrum points- GMAP accounts for these automatically.
- A key point is that in the event that the sum of the three central components is less than the corresponding noise power, then it is assumed that there is no clutter and all of the moments are then calculated using a rectangular window. If the power in the three central components is only slightly larger than the noise level, then the computed width for clutter removal will be so narrow that only the central (DC) point shall be removed. This is very important since, if there is no clutter then we want to do nothing or at worst only remove the central component.

- Because of this behaviour, there is no need to do a clutter bypass map, that is, turn-off the clutter filter at specific ranges, azimuths and elevation for which the map declares that there is no clutter. Because of the day-to-day variations in the clutter and the presence of AP, the clutter map will often be incorrect. Since GMAP determines the no-filter case automatically and then processes accordingly, a clutter map is not required.
- **Step 4: Replace clutter points**

  The assumption of a Gaussian weather spectrum now comes into play to replace the points that have been removed by the clutter filter. There are two cases depending on how the noise level is determined under Step 2, that is, the dynamic noise case and the fixed noise level case.
- **Dynamic noise level case:** From Step 2, we know which spectrum components are noise. From Step 3 we know which spectrum components are clutter. Presumably, everything that is left is weather signal. An inverse DFT using only these components is performed to obtain the autocorrelation at lags 0, 1. This is very computationally efficient since there are typically few remaining points and only the first two lags need be calculated. The pulse pair mean velocity and spectrum width are calculated using the Gaussian model (for example, see Doviak and Zrnic, 1993). Note that since the noise has already been removed, there is no need to do a noise correction. The Gaussian model is then applied using the calculated moments to determine a substitution value for each of the spectrum components that were removed in Step 3.
- In the case of overlapped weather as shown in the Figure 38 on page 230 example, the replacement power is typically too small. For this reason, the algorithm recomputes R0 and R1 using both the observed and the replacement points and computes new replacement points. This procedure is done iteratively until the power difference between two successive iterations is less than 0.2 dB and the velocity difference is less than 0.5% of the Nyquist interval.
- In summary of this step, the Gaussian weather model is used to repair the filter bias, that is, the damage that is caused by removing the clutter points. An IIR filtering approach makes no attempt to repair filter bias, rather the filter simply "digs a hole" into overlapped weather.

- **Step 5: Check for appropriate window and recalculate the moments if necessary.**

  The clutter power is known from the spectrum components that were removed in Step 3. Since the weather spectrum moments and the noise are also known from Step 4, the CSR can be calculated. The value of the CSR, is used to decide whether the Hamming window is the most appropriate. The scenarios are described in Figure 38 on page 230. The end result is that very weak clutter is processed using a rectangular window, moderate clutter a Hamming window, while severe clutter requires a Blackman window. Note that if no clutter were removed in Step 3, then the spectrum is processed with a rectangular window.
- The benefit of adaptive windowing is that the least aggressive window is used for the calculation of the spectrum moments, resulting in the minimum variance of the moment estimates.

GMAP Configuration

The **mf** command in the dspx TTY setups is used to configure GMAP filters. In the section for the spectrum filters select filter "Type 2" and specify the width of the ground clutter in m/s. This width is determined largely by your antenna rotation rate so you will want to con-figure several widths to deal with the different rotation rates in your operational scenario. An example might be filters indexed 1-5 corresponding to widths from 0.1 to 0.5.

A good practice is to make a scan on a clear day while using ascope or other utility and ob-serve what the actual width of the clutter is for your various scan rates. You will need to turnoff the clutter filtering to do this (pick "filter 0" for the all pass filter).

Example of Implementation

GMAP has undergone extensive evaluation for use in the US WSR88D ORDA network up-grade (Ice et al, 2004). They conclude that GMAP meets the ORDA requirements. Their study was based on a built-in simulator that is provided as part of the RVP8 and the ascope utility. The simulator allows users to construct Doppler spectra, process them and evaluate the results (Sirmans and Bumgarner, 1975). This is an essential tool for evaluating the system performance.

Figure 39 on page 236 shows an example of the simulations for the very difficult case when the weather has zero velocity, that is, it is perfectly overlapped with clutter. The upper left graph shows the weather signal with -40 dB power without any clutter and without any GMAP filtering. The graph at the upper right shows the same spectrum with 0 dB of clutter power added for a clutter width of 0.012 (0.3 m/s at S band, 1000 Hz PRF).

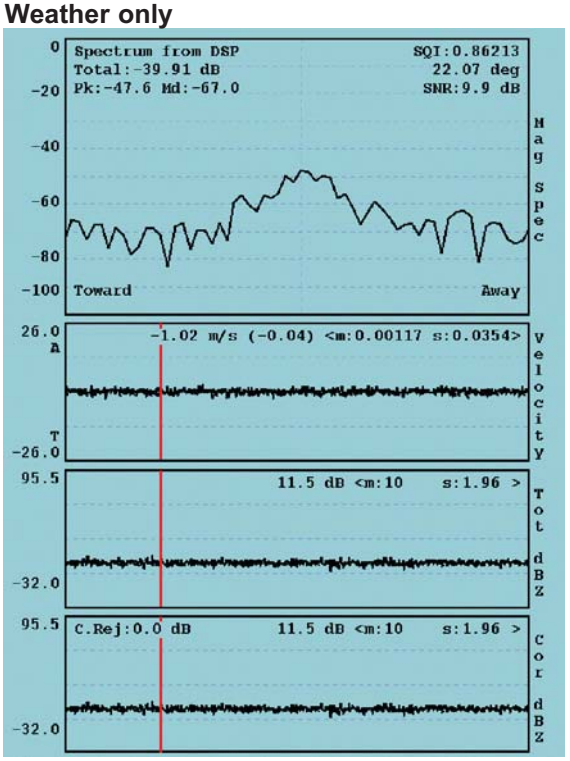
This is a CSR of 40 dB. The panel at the lower left shows the weather signal after GMAP filtering.

In each of the moment plots, there are several values that are displayed. The left-most num-ber shows the value at the range cursor which is positioned as indicated by the vertical line. To the right, the "m" value is the mean and the "s" value the standard deviation as averaged over all range bins (1000 in this example). For velocity these are in normalized units expressed as a fraction of the Nyquist interval. For reflectivity the values are in dB.

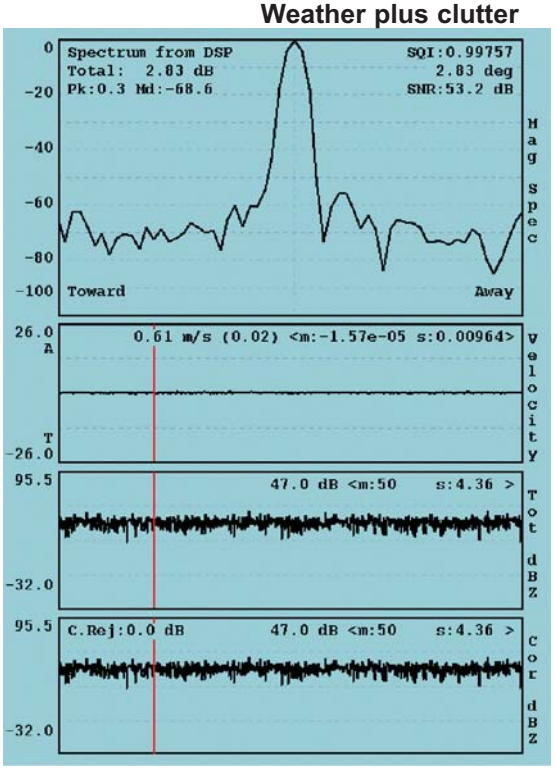
Some key points are:

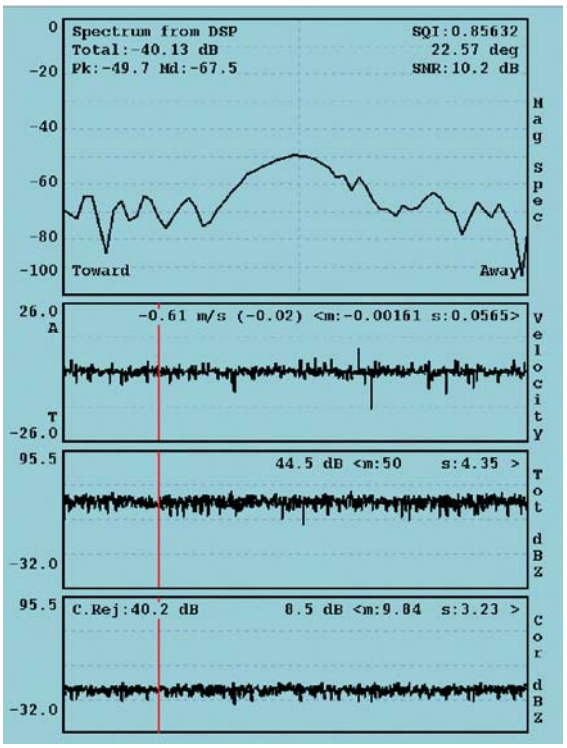
- The mean velocity is correctly recovered as expected (the "m" value in the plot), but the standard deviation is higher (0.06 vs 0.04 in normalized units).
- The "Cor dBZ" shows 40.2 dB of "C.Rej". This is the difference between the "Tot dBZ" and the "Cor dBZ" values. The expected value is 40 dB in this case. This indicates that GMAP has recovered the weather signal in spite of the aggressive clutter filtering that is required.
- The standard deviation of the "Tot dBZ" is greater in the weather plus clutter (4.35 normalized units) as compared to the weather-only case. This is caused by the fluctuations in the clutter power in the Gaussian clutter model.
- The standard deviation of the Cor dBZ after GMAP filtering, while not as low as for the weather-only case are lower than the weather plus clutter case. In other words, the GMAP processing removes some of the high variance in the dBZ estimates that is caused by clutter, but is not quite as good as doing nothing.

Weather only



Weather plus clutter





Simulation Characteristics

	Clutter	Weather	Units
Power	0	–40	dB
Vel	0	0	Any
Width	0.012	0.1	Normalized

PRF	1000 Hz	**Window**	Blackman
Mode	FFT	**Samples**	64

"Mag Spec": Doppler Spectrum in dB Units spanning the Nyquist interval.

"Velocity": Mean velocity of the spectrum in over Nyquist interval. Mean "m" and standard deviation values "s" are for the normalized interval ±1.

"Tot dBZ": Power in dB of weather and clutter. Mean "m" and standard deviation values "s" are in dB.

"Cor dBZ": Power in dB after GMAP filtering. Mean "m" and standard deviation values "s" are in dB.

"<m: ...s: ...>" mean and standard deviation over all ranges, in this case 1000 range bins.

Figure 39 GMAP Example

6.3 Autocorrelation R(n) Processing

6.3.1 Point Clutter Remover

The first step in autocorrelation processing is the optional removal of point clutter. "Point Clutter" is a non-meterological target of very narrow range. These are either small strong clutter targets, or airplanes.

There are 2 adjustable parameters for the point clutter algorithm:

TCM Point clutter threshold factor

Offset Point clutter range offset in bins

The first pass called "Clutter detection" is to screen all range bins to see if their power exceeds the normal level before and after in range. In other words, flag all bins if

$$[(T_0(r) > TCM * T_0(r - Offset)) \text{ and } (T_0(r) > TCM * T_0(r + Offset))]$$

These flag bits are output in the optional flag word. The second pass called "Clutter sensoring" involves linearly interpolating all the autocorrelation values in range over the interval of clutter bins including the offsets at either sides if the offset is larger than 1:

$$T_0(r) = \left(\frac{r_{end} - r}{r_{end} - r_{start}}\right) * T_0(r_{start}) + \left(\frac{r - r_{start}}{r_{end} - r_{start}}\right) * T_0(r_{end}) \text{ , where } r = r_{start}, r_{end}$$

Note that the same thing is done for R_0, R_1 and R_2.

6.3.2 Range averaging and Clutter Microsuppression

The next step (optional) is to perform range averaging. Range averaging can be performed over 2, 3, ..., 16 bins. This is accomplished by simply averaging the T_0, R_0, R_1 and R_2 values. This reduces the number of bins in the final output to save processing both in the RVP8 and in the host computer.

At the user's option, the range averaged data can be restricted to include only those bins which have an estimated clutter-to-signal ratio that falls within the CCOR threshold interval. By excluding isolated point clutter

targets from the range average the sub-clutter visibility of the averaged data is increased. Specifically, the Doppler test that is applied to each bin in order that it contribute to the overall sum is:

$$10\log R_0 - 10\log T_0 > CCOR_{thresh}$$

The total power T_0 is conserved after the exclusion of the isolated point clutter targets.

6.3.3 Reflectivity

The corrected reflectivity Z is output using a log scale based on the following equation:

$$dBZ = 10\log\left[\frac{T_0 - N}{N}\right] + dBZ_0 + 20\log r + ar + CCOR$$

This equation is simply a dB version of the familiar radar equation for distributed targets. The relationship between the measured autocorrelation function, the received signal and the noise can be expressed as:

$$T_0 = g^t g^r S + N$$

where g^t and g^r represent the transmitter and receiver gains, S is the average back scattered power from the targets and N is the measured average noise power. Neglecting attenuation and the contribution of ground clutter (for the moment), the radar equation can be written as.

$$Z = CSr^2 = \left[\frac{Cr_0^2 N}{g^r g^t}\right]\left[\frac{r^2}{r_0^2}\right]\left[\frac{T_0 - N}{N}\right]$$

where C is the radar constant and ro is a reference range which we will later set to 1 km. This is identical to the first three terms of the dB version of the equation with the definition that:

$$Z_0 = \frac{Cr_0^2 N}{g^r g^t} = Cr_0{}^2 I_0 \; where \; I_0 = \frac{N}{g^r g^t}$$

Z_o is called the calibration reflectivity factor. It is the equivalent radar reflectivity factor at the reference range when the return signal power is

equal to the noise power (SNR=0 dB). It is sometimes called the minimum detectable dBZ at 1 km, though it is more correct to call it the 0dB SNR detection level (See Figure 42 on page 252). The parameter I_o is the measured noise power at IF with appropriate calibration for the system gain. Calibration of the RVP8 involves defining the radar constant C and measuring the value of I_o. This is discussed in detail in 6.5 Reflectivity Calibration on page 252.

Essentially, the measurement of I_o is based on the measurement of the system noise at the time of calibration. However, if the receiver gain were to change after calibration, the use of periodic noise sampling properly corrects for this. For example, if the receiver gain were to change by a factor k, then we would measure a noise value of kN and an autocorrelation value of kT_o, i.e.,

$$Z = CSr^2 = \left[\frac{Cr_0^2 N}{g_r g_t}\right]\left[\frac{r^2}{r_0^2}\right]\left[\frac{kT_0 - kN}{kN}\right]$$

Thus the k's cancel to give us the same result for Z. This makes the approach robust to system gain fluctuations. Another way of saying this is that as long as the system sensitivity (noise figure) does not change, then the system does not require re-calibration.

The individual terms in the dB form of the equation are summarized below.

1st Term : $10\log\left[\frac{T_0 - N}{N}\right]$ **Signal to Noise Ratio**

The effect of this term is to subtract the measured noise and then divide by that noise. The result is a Signal-to-Noise ratio.

2rd Term:dBZ_o**: Calibration Reflectivity (see discussion above)**

dBZ_o is the minimum detectable dBZ at a reference range r_o =1 km,

3th Term: 20 log r : **Range Normalization**

This term is the range normalization expressed in dB form.

$$\left[\frac{r}{ro}\right]^2$$

4th Term:ar**: Gaseous Attenuation Correction**

This term accounts for gaseous attenuation. The constant a is set in the RVP8 EEROM since it is a function of wavelength. For a C-band system the default value is 0.016 dB per km (for two-way path attenuation).

5th Term: CCOR: Clutter Correction

This term corrects for the measured ground clutter. It's derivation is discussed in 6.3.7 Clutter Correction (CCOR threshold) on page 243.

6.3.4 Velocity

For a Doppler power spectrum that is symmetric about its mean velocity, the velocity is obtained directly from the argument of the autocorrelation at the first lag, that is,

$$V = \frac{\lambda}{4\pi\tau_s}\theta_1 \text{ where } \theta_1 = \arg\{R_1\} .$$

λ is the radar wavelength, τ_s is the sampling time (1/PRF). θ_1 is constrained to be on the interval [-π, π]. When $\theta_1 = \pm\pi$, then $V = \pm V_u$ where the unambiguous velocity is ,

$$V_u = \frac{\lambda}{4\tau_s}.$$

If the absolute value of the true velocity of the scatterers is greater than V_u, then the velocity calculated by the RVP8 is folded into the interval [- V_u, V_u] , which is called the Nyquist interval. Folding is usually easily recognized on a color display by a discontinuous jump in velocities. For example, if the true velocity is $V_u + \Delta V$, then the velocity calculated by the RVP8 is - $V_u + \Delta V$, which is $2V_u$ away from the true mean velocity.

For 8-bit outputs, rather than calculating the absolute velocity in scientific units, the RVP8 calculates the mean velocity for the normalized Nyquist interval [-1,1], that is, the output values are,

$$V' = \frac{\theta_1}{\pi}.$$

For example, an output value of -0.5 corresponds to a mean velocity of - V_u/2. The normalized velocity V' is more efficient use of the limited number of bits.

6.3.5 Spectrum Width Algorithms

The spectrum width is a measure of the combined effects of shear and turbulence. To a lesser extent, the antenna rotation rate can also effect the spectrum width. At high elevation angles, the fall speed dispersion of the scatterers also effects spectrum width.

There are two choices for the spectrum width algorithm used in the RVP8, depending on the speed and accuracy that are required for the application:

R_0, R_1 "fast" algorithm valid when SNR >> 10 dB

R_0, R_1, R_2 "accurate" algorithm for SNR >> 0 to 5 dB

The approach used is selected in the SOPRM command. The two approaches are described below:

R0, R1 Width Algorithm

Given samples of the Doppler autocorrelation function, numerous estimates of spectral variance can be computed (Passarelli & Siggia, 1983). The particular estimator used by the RVP8 employs the magnitudes of R_0 and R_1 and assumes that the Doppler spectrum is Gaussian (usually an acceptable assumption) and that the signal-to-noise ratio is large. Specifically we have (similar to Srivastava, et al 1979):

$$Variance = 2\ln\left[\frac{R_0}{|R_1|}\right] = -2\ln[SQI]$$

where "ln" represents the natural logarithm. This can be compared to the expression in the preceding section for SQI to illustrate that this expression for the variance is only valid when:

$$\frac{SNR}{SNR+1} \approx 1$$

which occurs when the SNR is large.

This variance estimator is normalized to the Nyquist interval in units of [-π, π]. Thus, for example, a variance of $\pi^2/25$ would be obtained from a Gaussian spectrum having a standard deviation equal to one fifth of the total width of the plotted spectral distribution. For scientific purposes, the spectrum width (standard deviation) is more physically meaningful than the variance, since it scales linearly with the severity of wind shear and turbulence. For these reasons, the width W is output by the RVP8:

$$W = \frac{\sqrt{Variance}}{\pi}$$

Again, for efficient packing in 8-bits, width is normalized to the Nyquist interval [-1, 1]. For the example given above, the output width W would be (1/5). To obtain the width in meters per second, one multiplies the output width by V_u .

R0, R1, R2 Width Algorithm

The width algorithm in this case is similar except that the addition of R_2 extends the validity of the width estimates to weak signals. In this case the variance is:

$$Variance = \frac{2}{3}\ln\left[\frac{|R_1|}{|R_2|}\right]$$

The output width W is then defined as in the previous section.

6.3.6 Signal Quality Index (SQI threshold)

An important feature of the RVP8 is its ability to eliminate signals which are either too weak to be useful, or which have widths too large to justify further analysis. This is done via the signal quality index (SQI) which is defined as:

$$SQI = \frac{|R_1|}{R_0}$$

The SQI is the normalized magnitude of the autocorrelation at lag 1 and varies between 0 for an uncorrelated signal (white noise) to 1 for a noise-free zero-width signal (pure tone). Mean velocity estimates are degraded when the spectrum, width is large or when the signal-to-noise ratio is weak. The SQI is a good measure of the uncertainty in the velocity estimates and is a convenient screening parameter to compute. In terms of the Gaussian model, the SQI is :

$$SQI = \frac{SNR}{SNR + 1}\, e^{\frac{-\pi^2 W^2}{2}}$$

where the SNR is the signal-to-noise ratio. For very large SNR's the SQI is a function of the spectrum width only. For a zero-width pure tone (W=0),

the SQI is a function of the SNR only (for example, for W=0, an SNR of 1 corresponds to SQI=0.5). The SQI threshold is typically set to a value of 0.4 to 0.5.

6.3.7 Clutter Correction (CCOR threshold)

In addition to calculating the R_0, R_1 and optional R_2 autocorrelation terms, which are based on filtered time series data, the RVP8 also computes T_0 which is the total unfiltered power. By comparing the total filtered and unfiltered powers at each range bin, a clutter power, and hence a clutter correction, for that bin can be derived. The clutter correction is defined as,

$$CCOR = 10\log\frac{S}{C+S} = 10\log\frac{1}{CSR+1}$$

where S is the weather signal power, C is the clutter power and CSR is the clutter-to-signal ratio. The algorithm for calculating CCOR depends on whether the optional R_2 autocorrelation lag is computed as described below.

R0, R1, R2 Clutter Correction

In this case CCOR is estimated from,

$$CCOR_{est} = 10\log\left[\frac{R_0}{T_0}\right]$$

$$= 10\log\left[\frac{S+N}{C+S+N}\right] = 10\log\left[\frac{1+\frac{1}{SNR}}{CSR+1+\frac{1}{SNR}}\right]$$

Here, the expression is strictly valid only when the signal-to-noise ratio (SNR=S/N) is large. Thus when the 2-lag approach is used, the clutter corrections are not as accurate for weak weather signals. However, the error is typically less than 3 dB.

R0, R1, R2 Clutter Correction

In this case there is enough information to compute the clutter signal and noise power inde-pendently. The algorithm for CCOR is:

$$CCOR_{est} = 10\log\frac{S}{C+S} = 10\log\frac{1}{CSR+1}$$

The clutter power is computed from:

$$C = T_0 - R_0 = [C + S + N] - [S + N]$$

The signal power S is then computed from:

$$S = \left|R_1\right| \exp \frac{\pi^2 W^2}{2}$$

W is the width that has been previously calculated. This approach yields more accurate results for the clutter correction in the case of a low SNR.

6.3.8 Weather Signal Power (SIG threshold)

A parameter called SIG is also calculated to provide an estimate of the weather signal-to-noise ratio in dB for thresholding. The SIG calculation is different depending on the whether the optional R2 autocorrelation is computed.

R_0, R_1Calculation

In this case the SIG is computed as follows:

$$SIG = 10\log\left[\frac{T_0 - N}{N}\right] + CCOR$$

This term represents the SNR after the removal of clutter. The *CCOR* value is the one de-scribed for R_0, R_1 in the previous section.

R_0, R_1, R_2Calculation

In this case the SIG is computed based on the SNR which is:

$$SIG = 10\log\left[\frac{2\pi S}{R_0 - 2\pi S}\right]$$

where the signal power S is determined as described in the preceding section.

6.3.9 (Signal+Noise)/Noise Ratio (LOG threshold)

A threshold parameter called LOG is also calculated to provide a signal strength estimate that is useful for qualifying reflectivity. For historical reasons, the LOG threshold is not the true SNR (whose dB representation can be both positive and negative) but rather, the ratio of Signal plus Noise to Noise, which always has a positive representation in dB. Specifically:

$$LOG = 10\log\left[\frac{T_0}{N}\right] \ (when\ applied\ to\ the\ dBT\ parameter)$$

$$LOG = 10\log\left[\frac{R_0}{N}\right] \ (when\ applied\ to\ the\ other\ parameter)$$

6.4 Thresholding

An important feature of the RVP8 is its ability to accept or reject incoming data based on derived properties of the signals themselves. Typically, "rejected" data are not displayed by the user's software, thus making for very clean weather presentations.

6.4.1 Threshold Qualifiers

For data quality control, each RVP8 output parameter can be qualified, that is, either accepted or rejected for output, based on four threshold criteria:

ID	Criterion Name	Pass Criterion
LOG	(Signal+Noise)-to-Noise Ratio	LOG > threshold
SQI	Signal Quality Index	SQI > threshold
CCOR	Clutter Correction	CCOR > threshold
SIG	Weather Signal Power	SIG > threshold

The calculation of the measured levels (for example, SQI) for each of these qualifications has been described in previous sections of this chapter. All four qualification criteria can be switched on and off independently, and the threshold levels (for example, SQI_{thresh}) can each be set independently. Further, each qualifier test can be AND'd and OR'd with any other. This

allows very complex thresholds criteria to be constructed as required. The four threshold qualifiers are summarized below.

LOG — This is a measure of signal strength that is usually used for the thresholding of reflectivity data. The default LOG threshold value is 0.5 dB.

SQI — The SQI threshold is typically used for velocity and width thresholding since it is a measure of the coherency. It is a number between 0 and 1 (dimensionless) where 0 is perfect white noise and 1 is a pure tone (perfect Doppler signal). The default SQI threshold value is 0.5.

CCOR — The clutter correction threshold is typically used to reject measurements when the clutter in a range bin is very strong (i.e., when the calculated CCOR is a large negative number in dB). The appropriate value depends on the coherency of the radar system. The default threshold is set to 25 dB. Threshold values less than this (more negative) reject fewer clutter bins. Threshold values closer to zero reject more clutter bins.

SIG — This is typically used only for thresholding the spectrum width to assure that the signal power is strong enough for an accurate width measurement. The default threshold value is 10 dB. If R_2 processing is used, this can usually be reduced to 5 dB for width thresholding.

The following are the default threshold combinations for each of the parameters that can be selected for output from the RVP8:

Parameter	Description	Threshold
dBZ	Reflectivity with clutter correction	LOG and CCOR
dBT	Reflectivity without clutter correction	LOG
V	Mean velocity	SQI and CCOR
W	Spectrum width	SQI and CCOR and SIG
Dual Pol	Differential reflectivity	LOG

6.4.2 Adjusting Threshold Qualifiers

The effect of the various threshold qualifiers for each output parameter are discussed in this section. In optimizing thresholds for your application, it is recommended that you change only one parameter (level or criterion) at a time so that you can verify the effect. Some hints for optimizing the levels for the default criteria are provided below:

LOG — To optimize the LOG level, display dBT or dBZ and select the lowest value of the threshold that eliminates the display noise. If the LOG level is set too high you lose sensitivity. Note that if you average more pulses or ranges, then the threshold level can usually be reduced.

SQI — To optimize the SQI level, display velocity and select the lowest value of the threshold that eliminates the display noise. If the SQI level is set too high you lose sensitivity. In general, you should see a greater area covered by velocity than reflectivity since the velocity is more sensitive. If you do not, you should reduce your SQI threshold. Note that if you average more pulses or ranges, then the threshold level can usually be reduced.

CCOR — This is used to eliminate clutter targets that are very strong. It should not be set to eliminate all clutter targets on a clear day since this means that you are losing sensitivity. To optimize the CCOR threshold it is best to know your system coherency in terms of dB of clutter cancelation. Start at a value of 10 dB greater (closer to 0) than this. Now display a PPI of dBZ at an antenna elevation of ~1 degree. The display should be relatively clean of any clutter targets since most will be rejected. Now reduce the CCOR (more negative) to increase the number of clutter targets on the display until the number of clutter targets does not increase. The optimum value of the CCOR is approximately 5 dB more (closer to zero) than this point. For example, if the number of clutter targets is a maximum at 35 dB, then set the CCOR to ~30 dB. Note that your clutter filter selection will effect the result.

SIG — This should be done last. To optimize the SIG level, display the width W and select the lowest value of the threshold that eliminates the display noise. If the SIG level is set too high you lose sensitivity. Note that if you average more pulses or ranges, then the threshold level can usually be reduced.

When thresholding dBZ and dBT reflectivity data with SQI, the comparison value for accepting those data is the secondary SQI threshold that is defined via a slope and offset from the primary user value (see **Mf** command). This secondary threshold is more permissive (lower valued),

and is traditionally used to qualify LOG data only in the Random Phase processing mode. But the secondary SQI threshold is applied uniformly in all processing modes whenever reflectivity data are specified as being thresholded by SQI.

This gives you more freedom in applying an SQI threshold to your LOG data, because the cutoff value for reflectivity can be chosen independently from the cutoff value for the other Doppler parameters. The full SQI test would not normally be applied to LOG data because of the so-called "black hole" problem, that is, loss of LOG data within regions of high shear, even though the reflectivity itself was strong. You may experiment with applying a secondary SQI threshold to help cleanup the LOG data, but without introducing any significant black holes.

6.4.3 Speckle Filters

A speckle filter is a final pass over each output ray, wherein isolated bins are removed. There are two speckle removers in the RVP8.

- 1D single-ray speckle filter. This can be used for any output parameter.
- 2D 3x3 speckle filter. If enabled, this is applied only to T, Z, V and W.

The 1D speckle filter is the default technique. The 2D 3x3 filter is enabled by selection in the mp TTY setups:

```
2D Final Speckle/Unfold "User" or "Always"
```

Both of these speckle filters remove isolated data points that are likely to be noise, interference, aircraft, birds or other point targets. Meteorological targets typically occupy multiple range bins so are not effected by the speckle filter. There are two primary benefits derived from using a speckle filter:

- Displays look "cleaner" to observers.
- Thresholds can be set slightly more sensitive without increasing the number of noise pixels.

The 2D 3x3 filter actually performs data filling of "missing speckles" as well as eliminating isolated speckle bins. The two algorithms are discussed below.

1D Speckle Filter

A ray is the basic azimuth unit of the RVP8 (for example, 1 degree) over which the samples are averaged to obtain the output base data (T, Z, V, W). For this filter, a speckle is defined as any single, valid bin (not thresholded), having thresholded bins on either side of it in range. Any such isolated bin in a ray is set to "threshold". The algorithm is shown schematically below.

Note that there are two independent 1D speckle removers– one for the reflectivity data (dBT, dBZ and ZDR) and one for the Doppler data (V and W). Each one should be switched on or off, depending on the specific nature of the targets being observed. For example, when making a clutter map of the area, one would certainly want to switch both speckle filters off.

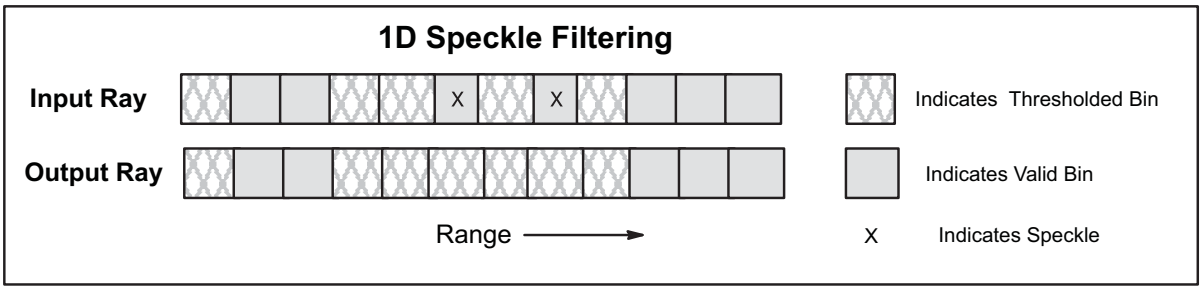


Figure 40 1D Speckle Filtering

2D 3x3 Speckle Filter

The 2D filter examines three adjacent range bins from three successive rays in order to assign a value to the center point. Thus, for each output point, its eight neighboring bins in range and time are available to the filter. Only the dBZ, dBT, Vel, and Width data are candidates for this filtering step; all other parameters are processed using the default 1D speckle filter.

The rules for the filter are as follows:

	Center Point Action	
	Assign Threshold	**Else**
Valid Center Point	If there are no or only one other valid point in the 3x3.	Do Nothing. Pass the center point value as-is.
Thresholded Center Point	If there are 5 or fewer valid neighbors in the 3x3.	If there are 6 or more valid neighbors in the 3x3, average to fill the center point.

Thus the 2D 3×3 filter performs 2 functions:

- Filling by interpolation.
- Thresholding of isolated noise bins.

Some examples are shown graphically in the figure below.

For dBZ, dBT, and Width, the interpolated value for filling is computed as the arithmetic mean of all available neighbors. The procedure for Velocity is similar except that the 8-bit angles are first converted to Cartesian vectors, then averaged and converted back to polar.

The filter has some interesting properties when combined with other algorithms.

Dual PRF Unfolding

Dual–PRF velocity unfolding is computed within the 3x3 filter whenever both are enabled. There are two steps to the process:

- Step 1: The most recent and the previous ray are used. For every valid point in the most recent ray, the algorithm performs a search among the three nearest neighbors in the previous ray to find a valid velocity. The search pattern is shown at the bottom of the previous figure. This larger selection of alternate–PRF bins makes it more likely that the algorithm will find the pairs of Low/High PRF data that are required for unfolding.

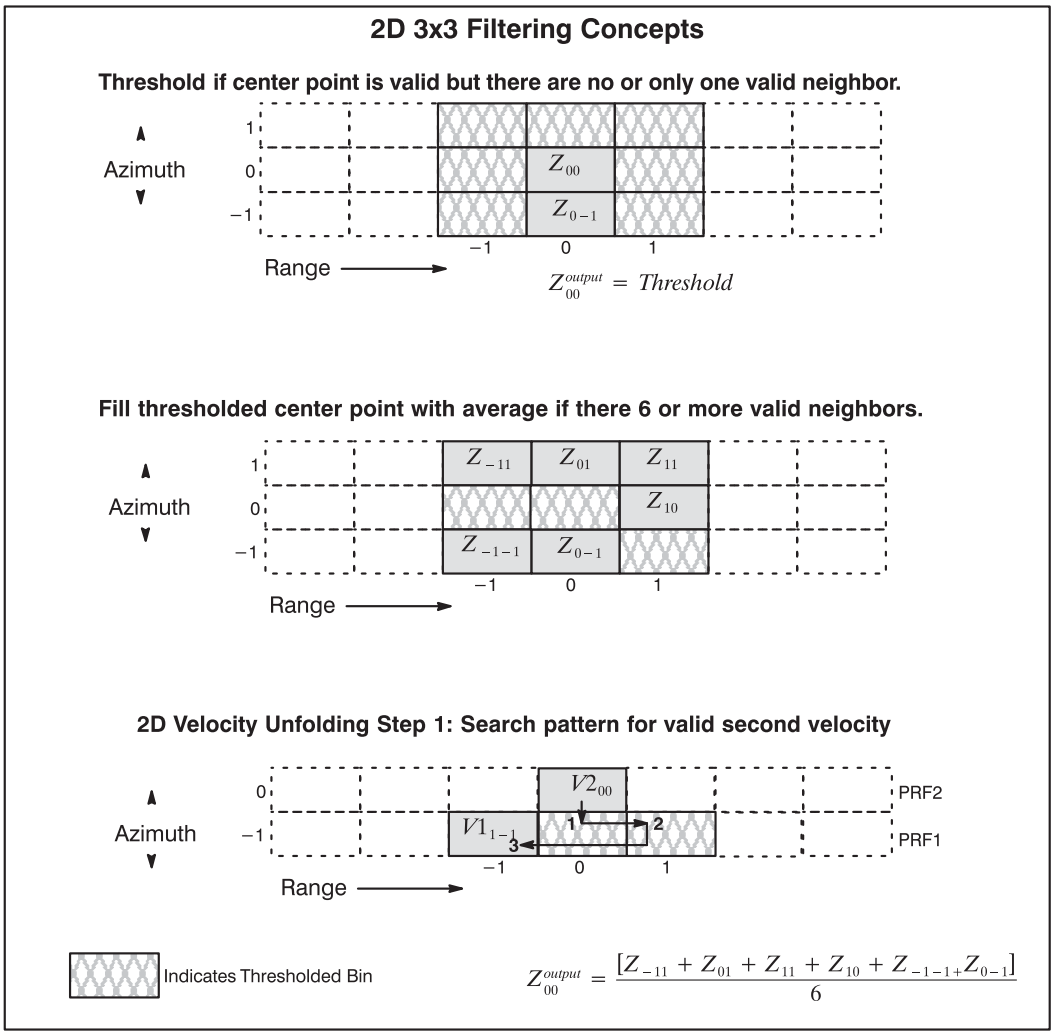


Figure 41 2D 3x3 Filtering Concepts

- Step 2: The unfolded velocities are then subjected to the standard 3x3 filtering.

Dual PRF, Random Phase Processing

In random phase processing, the "seam" at the start of the second trip is always problematic since the transmitter main bang and nearby clutter will virtually always wipe out the first few 2nd trip range bins. At a constant PRF the 2nd trip seam is always at the same range, but in dual PRF random phase mode, the seam is different each ray. Thus thresholded bins at the seam of the high PRF can be surrounded on either side by valid bins taken at the low PRF. The 3x3 filter has the effect of interpolating the reflectivity and width data over the bins at the 2nd trip seam. Velocity data will also be filled–in using the nearest neighbor. Thus the 2D filter mitigates much of the damage that is caused at the 2nd trip seam to make a nearly seamless display.

6.5 Reflectivity Calibration

The calculation of reflectivity described in 6.3.3 Reflectivity on page 238 required the calibration reflectivity dBZ_o . This section describes it's derivation. You can use the **zauto** utility to perform the calibration. (See the *IRIS/RDA Utilities Manual*.)

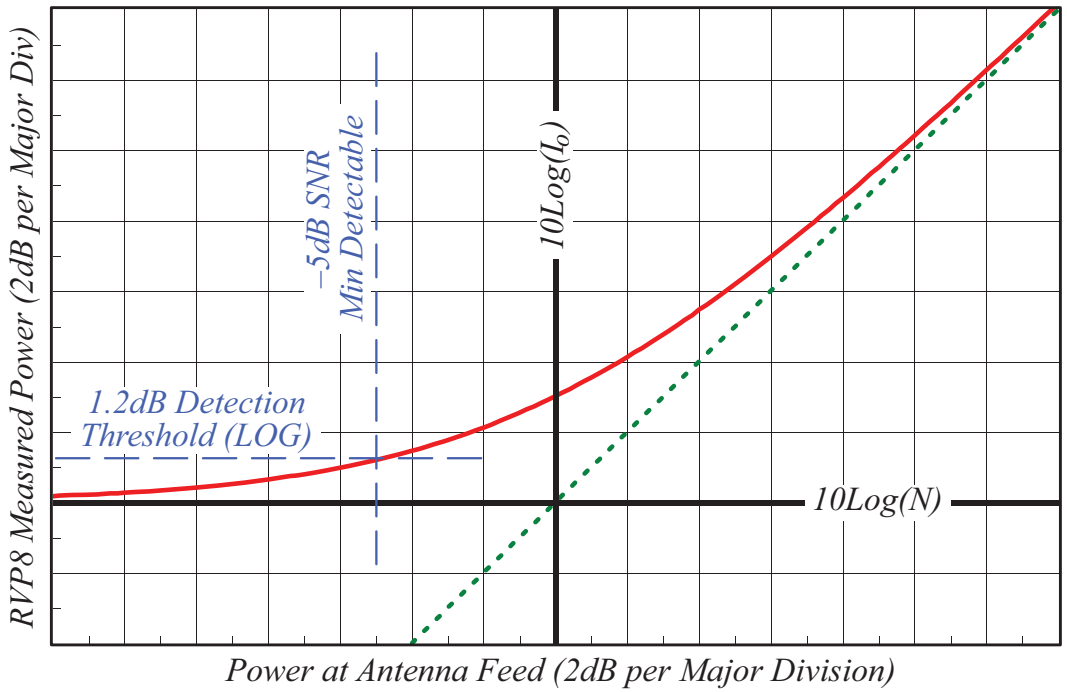


Figure 42 Model Intensity Curve

6.5.1 Plot Method for Calibration of I_o

This approach generates the curve shown above (red) which determines the value of I_o . The general procedure is to connect a calibrated signal generator to the radar receiver and inject known power levels to generate a calibration plot of measured power vs the inserted power at the antenna feed, similar to that in Figure 42 on page 252. The calibration reflectivity dBZ_o is computed from the radar constant and the value of I_o , which is the intercept of the straight line fit (green) with the Noise level.

Why does this geometric construction yield the value of I_o ? Let G_{dB} represent the overall gain of the RF and IF components leading up to the RVP8. The green line can be interpreted as the response of an ideal noise-free amplifier having gain GdB , while the red curve is the response of the real-world amplifier(s) whose equivalent front-end noise is I_o :

$$(Red)\ 10\log_{10}(P_{OUT}) = G_{dB} + 10\log_{10}(P_{IN} + I_0)$$

$$(Green)\ 10\log_{10}(P_{OUT}) = G_{dB} + 10\log_{10}(P_{IN})$$

The measured receiver noise is the horizontal asymptote of the red curve, that is, the value of the red curve when the input power P_{IN} is zero:

$$10\log_{10}(N) = G_{dB} + 10\log_{10}(I_0).$$

Intersecting this measured noise level with the green straight line gives:

$$G_{dB} + 10\log_{10}(I_0) = G_{dB} + 10\log_{10}(P_{IN})$$

From which we see that the input power at the point of intersection is, indeed, I_o .

Note that I_o is the received signal level that will produce 0dB SNR, that is, signal power equal to noise power. This should not be confused with the minimum detectable power P_{MDS} which typically will be several dB lower, depending on processor settings. In the above example, a 1.2dB LOG detection threshold is shown (horizontal blue line) for the received signal. If the RVP8 is applying sufficient range and time averaging so that thermal noise alone produces very few false alarms above 1.2dB, then P_{MDS} will be a full 5dB lower than I_o . We would expect a detection rate of roughly 50% for echoes arriving at this "minimum detectable" level.

Typically a CW test signal is used to generate the test curve shown in Figure 42 on page 252. Follow the instructions provided by the radar manufacturer for injecting a test signal. During calibration, the radar should be fully operational, so that all sources of noise are present. Ideally the transmitter should be turned on during calibration.

NOTE	Verify with the radar manufacturer that no damage will occur to the signal generator if the transmitter is running during the calibration.

To perform the calibration, insert signals at steps of 5 or 10 dB over the entire range of the system. Draw the plot shown in Figure 42 on page 252. You can utilize fine resolution steps at the ends of the scale to observer the details of the roll off. Be sure to raise the antenna up a few degrees to avoid ground thermal noise. Also tune the frequency of the signal generator using the setup command **pr**, and displaying the received signal spectrum. Be sure to check the tuning at the end of the calibration to make sure the signal generator and IFD have not drifted apart.

Each time that a new signal level is injected, the measured power values are obtained by first invoking the SNOISE command and then reading-back the results using the GPARM command. The Log of Measured Noise Level (Word 6) from GPARM should be used. This procedure averages many samples together. For IRIS users, this is all handled by the **zauto** utility.

Finally turn it all the way down and make one more sample to measure the noise level N. I_0 is obtained from the intercept of the horizontal line at N and the straight line fit to the linear portion of the curve. This value must be corrected for losses as discussed in the section below.

6.5.2 Single-Point Direct Method for Calibration of Io

This calibration method requires no support software. The approach uses the TTY setups commands. Again the signal generator output must be calibrated in absolute dBm. Use a power meter to check the calibration.

- Turn the radiate off and connect the signal generator to the test signal injection point.
- Raise the antenna to at least 20 degrees, and set the azimuth to point away from any known RF sources including the sun.
- Select the pulse width using the mt command.
- Select the pr command and use the commands to set the following:

  ```
  Plotting Received Power Spectrum...
  ```

  ```
  Rx:Pri, Zoom:x1-x8, Navg:25, Start:100.01 usec (14.99 km), Span:50 usec
  ```
- Set the signal generator to the approximate radar RF frequency with a power level corresponding to a strong signal (30 dB above the noise), and use a CW signal (not a pulse). This signal should be visible as a peak in the spectrum display. Adjust the siggen RF signal frequency so that produces the precise IF frequency (for example, IF frequency of 30 MHz).
- Turn the signal generator off and record the "Filtered" power level. Note that because of the large averaging it will require several seconds for the average to stabilize.
- Turn the signal generator on, verify that the peak is still at the IF frequency and adjust the power level to obtain precisely 3 dB more "Filtered" power than was observed with the noise only. Again, allow several seconds for the averaging to stabilize after you make each amplitude adjustment.

This is the value of I_o , that is, the test signal signal power equals the noise power. The next step is to correct the value of I_o for losses as discussed in the section below.

6.5.3 Treatment of Losses in the Calibration

In the calibration of the dBm level of the test signal, be sure to account for any losses that may occur between the antenna feed and the injection point, and in the cable and coupler that is used to connect the signal generator to the injection point. Figure 43 on page 256 illustrates the nomenclature of the various losses that are involved in the calibration. The relationship between the injected test signal and the value of the received power relative to the feed is:

$$dBm_{Feed} = dBm_{Injected} + dBL_{Feed:Coupler}$$

$$dBm_{Feed} = dBm_{Siggen} - dBL_{Coupler} - dBL_{Cable} + dBL_{Feed:Coupler}$$

For example, assume the following:

Loss between the feed and the coupler	$dBL_{\text{Feed:Coupler}}$	3 dB
Loss caused by the coupler	dBL_{Coupler}	30 dB
Loss in the cable from siggen to coupler	dBL_{Cable}	2 dB

Then if the test signal generator output is -50 dBm, the injected power is

$$dBm_{Injected} = -50 - [30 + 2] = -82 \text{ dBm.}$$

The equivalent power at the feed is then 3 dB more than this

$$dBm_{Feed} = -82 + 3 = -79 \text{ dBm.}$$

During the calibration, there are several ways to handle the losses using these equations. Two examples are:

- Each signal generator value can be corrected for losses so that the calibration plot shows IFD measured power vs received power at the feed. This is recommended for manual calibration.
- The signal generator values can be plotted directly and the intercept power I_o can be corrected for losses so that it is properly referenced to power at the feed. This is the approach used by the IRIS **zauto** utility.

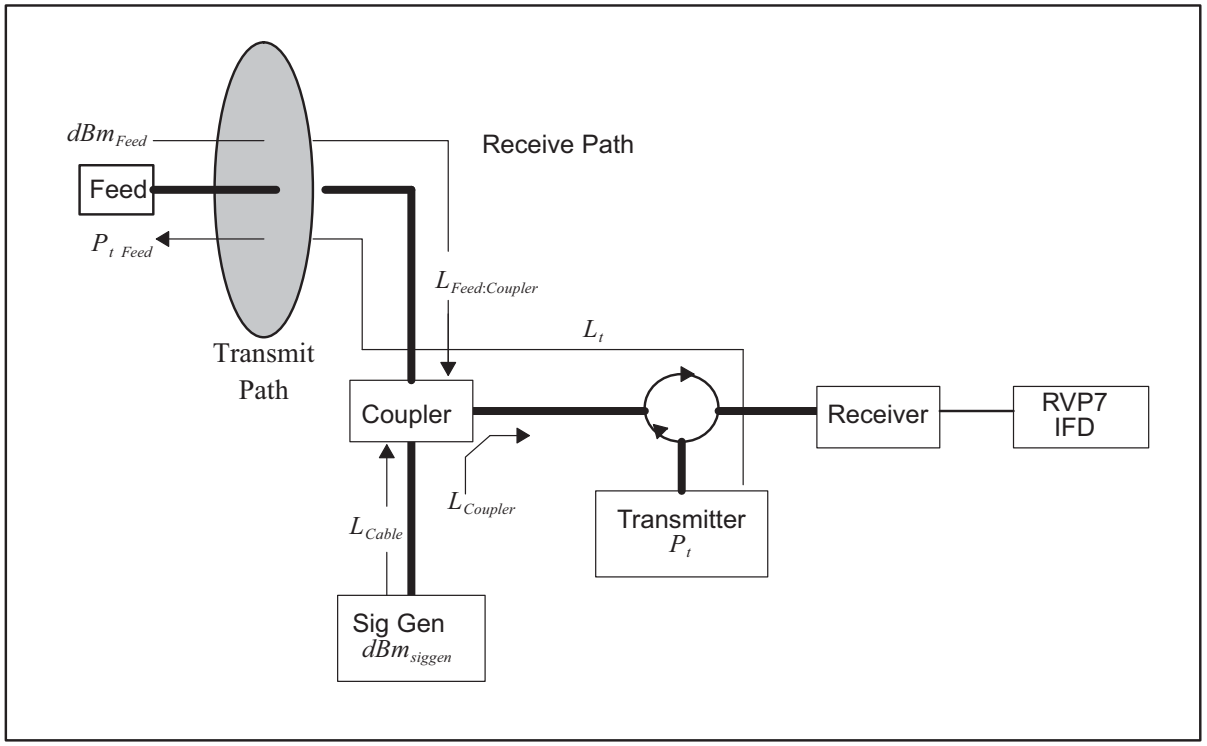


Figure 43 Illustration of Losses that Affect LOG Calibration

6.5.4 Determination of dBZ$_o$

The calibration reflectivity is determined from the radar equation as follows:

$$dBZ_0 = 10\log[Cr_0^2 I_0]$$

where I_o is in mW (corrected for receive losses), the reference range r_o is 1 km, and the radar constant C is:

$$C = \frac{2.69 \times 10^{16} \lambda^2}{P_t \tau \theta \phi G^2} L_t$$

where,

λ	Radar wavelength in cm.
P_t	Transmitted peak power in kW.
L_t	Transmit loss (for example, 3 dB corresponds to $L_t = 2$)
T	Pulse width in microseconds.
θ	Horizontal half-power full beamwidth.
φ	Vertical half-power full beamwidth.
G	Antenna gain (dimensionless) on beam axis.

The radar constant is determined from the characteristics of your radar (check with the manufacturer if you are unsure of the values). Note that transmit losses are accounted for in the radar constant, while receiver loss is usually included in the calculation of I_o .

Finally, if the value of I_o calculated above was not based on loss-corrected dBm values, correct I_o as follows:

$$dBI_{0\ corrected} = dBI_0 - dBL_{Coupler} - dBL_{Cable} + dBL_{Feed:Coupler}$$

Example Calculation of dBZo:

This sample calculation is provided so that programmers can check their arithmetic. The radar parameters:

λ	Radar wavelength in cm.	5 cm
P_t	Transmitted power in kW.	500 kW
L_t	Transmit Loss	2 (3 dB)
T	Pulse width in microseconds	1 microsecond
θ	Horizontal half-power beamwidth in degrees	1 degree
φ	Vertical half-power beamwidth in degrees	1 degree
G	Antenna gain (dimensionless) on beam axis	19,953 (43.0 dB)

The radar constant for this example is,

$$C = \frac{2.69 \times 10^{16} \lambda^2}{P_t \tau \theta \phi G^2} L_t = \frac{(2.69 \times 10^{16})(5)^2}{(500)(1)(1)(19,953)^2} \quad (2.0)$$

$$= 6.76 \times 10^6 [mm^6 m^{-3} km^{-2} mW^{-1}]$$

Assume that I_o with loss correction is calculated to be -105 dBm (3.16×10^{-11} mW), then dBZ_o is,

$$dBZ_0 = 10\log[Cr_0^2 I_0] = 10\log[(6.76 \times 10^6)(1)^2(3.16 \times 10^{-11})]$$

$$= -36.7 dB(mm^6 m^{-3})$$

This value would be down-loaded to the signal processor using the SOPRM command.

6.6 Dual PRT Processing Mode

NOTE	These modes were originally implemented in the RVP7 processor but have not yet been ported into the RVP8.

The RVP8 supports two major modes for Dual PRT processing, that is, algorithms using triggers that consist of alternate short and long periods. Most of the Doppler parameters are available in each of these modes. You may also request time series data in both cases; the samples will be organized so that the first pulse of a short PRT pair always comes first.

6.6.1 DPRT-1 Mode

The DPRT-1 trigger consists of a very short PRT from which Doppler data are obtained, followed by a much longer PRT whose purpose is to limit the average duty cycle of the transmitter. No information is extracted from the long PRT pair, but Dual-PRF techniques can still be used by varying the short period from ray to ray. The "-1" suffix in the name for this mode is a reminder that Doppler parameters are computed from the short PRT only. The DPRT-1 mode is intended for millimeter wavelength radars that must

run at a very high effective PRF (up to 20KHz) to get an acceptable unambiguous velocity, but which also have a much lower duty cycle constraint on the average number of pulses transmitted each second.

In DPRT-1 mode the requested PRF from the host computer will generally be quite large (up to 20KHz); and the reciprocal of this "effective instantaneous PRF" will determine the trigger's short PRT interval. In this way, all subsequent physical calculations will be scaled correctly, for example, unambiguous velocity, maximum first trip range, etc., are all supposed to be based on the short PRT interval. The host computer must therefore be configured so that it can ask for these very high trigger rates.

The duration of the long PRT interval is not specified directly by the host computer. Rather, the RVP8's "Maximum number of Pulses/Second" setup parameter is used to compute how much delay to insert in order to insure that the transmitter's duty cycle is not exceeded. This special treatment applies only in DPRT mode; all other modes that have uniform triggers continue to interpret the RVP8's trigger bound as a simple "Maximum PRF".

Since DPRT-1 mode uses only the short pairs of pulses, it is not possible to run the "R2" moment estimation algorithms. The RVP8 will return the GPARM "Invalid Processor Configuration" bit if "R2" is requested in DPRT mode. The error bit will also be returned if the number of pulses requested (sample size) is not even. All other error conditions are the same as FFT mode.

WARNING Since the RVP8's "Maximum number of Pulses/Sec" is used to enforce the duty cycle limit, it is essential that it not be overwritten by the host computer's upper PRF limit, which typically will be much higher. To insure this, you must make sure that the PWINFO command is disabled in the RVP8 "Mc" setup menu. You will have no duty cycle protection if you do not do this.

NOTE You may still choose to run Dual-PRF velocity unfolding within the DPRT-1 mode. What will happen is that the short PRT will vary in the selected 3:2, 4:3, or 5:4 ratio, but the overall duty cycle will remain constant. The combination of Dual–PRF and DPRT-1 is tremendously effective in extending the radar's unambiguous velocity interval.

6.6.2 DPRT-2 Mode

The trigger consists of alternating short and long period pulses, where the ratio of the periods is determined by the velocity unfolding ratio that has been selected. Doppler data are extracted from both the short and long pulse pairs (hence the "-2" suffix), and unfolded velocities are made available on each ray based on the combined PRT data from that ray alone. DPRT-2 mode is intended for rapidly scanning radars where the ray-to-ray spatial continuity assumptions of the traditional Dual-PRF algorithms do not apply.

The DPRT-2 velocity unfolding algorithm uses a modified version of the standard Dual-PRF algorithm. Both start by computing a simple velocity difference as a first approximation of the unfolded result. The standard algorithm uses that difference to unfold the velocity from the most recent ray, which yields a lower variance estimate than the difference itself. The DPRT-2 algorithm is similar, except that the folded velocity from both PRTs are unfolded independently and then averaged together.

In addition to the above, the RVP8 also computes the DC average of the (I,Q) data within each bin. This is used as a simple estimate of clutter power, so that corrected reflectivities are available in DPRT-2 mode whenever a non-zero clutter filter is selected. DPRT-1 mode is the same in this respect. However, the DPRT-2 widths use an improved algorithm based on the two different PRTs, and which avoids the SNR sensitivity of the DPRT-1 width estimator.

6.7 Dual PRF Velocity Unfolding

For a radar of wavelength λ operating at a fixed sampling period $\tau_s = 1/PRF$, the unambiguous velocity and range intervals are given by:

$$V_u = \frac{\lambda}{4\tau_s} \text{ and } R_u = c\frac{\tau_s}{2}$$

where "c" is the speed of light. Often these intervals do not fully cover the span of velocity and range that one would like to measure. The problem is generally worse for short wavelength radars, since that unambiguous velocity span is directly proportional to λ for a given τ_s. If the unambiguous range interval is made sufficiently large by increasing τ_s, then the resulting velocity span may be unacceptably small.

The RVP8 provides a built-in mechanism for extending the unambiguous velocity span by a factor of two, three, or four beyond that given above. The technique, called Dual PRF velocity unfolding, uses two pulse periods

rather than one, and relies on the extra information thus obtained to correct (i.e. unfold) the mean velocity measurement from each individual period. The Dual PRF trigger pattern consists of alternating (N+k)-pulse intervals where the period in each interval is either τ_l(for the low-PRF) or τ_h (for the high-PRF). Here "N" is the sample size, and "k" represents a delay that permits the clutter filter to equilibrate to the new PRF after each change. The clutter filter impulse response lengths vary according to which filter is selected.

The two trigger periods τ_l and τ_h must be chosen in either a 3:2, 4:3, or 5:4 ratio. These ratios give factors of two, three, and four times velocity expansion over the τ_h period alone. The unfolding algorithm makes use of the following results. Suppose that the radar observes a target with mean velocity V at each of the two trigger periods. The measured phase angles for the R_1 autocorrelations at the two PRFs are:

$$\theta_l = \frac{4\pi V \tau_l}{\lambda} \text{ and } \theta_h = \frac{4\pi V \tau_h}{\lambda}$$

where angles outside the basic [- π, π] interval are returned to that interval by appropriate additions of ± 2π . These angles correspond to the ordinary single-PRF Doppler velocity measurements, and the ± 2π uncertainties reflects the fact that each measurement is folded into its own unambiguous interval:

$$V_{ul} = \frac{\lambda}{4\tau_l} \text{ and } V_{uh} = \frac{\lambda}{4\tau_h}$$

If we define φ to be the difference between the two measured phases then:

$$\phi = \theta_l - \theta_h = \frac{4\pi}{\lambda}[\tau_l - \tau_h]$$

which can be interpreted as a phase angle within the unfolded interval:

$$V_{u\ unfold} = \frac{\lambda}{4(\tau_l - \tau_h)}$$

Now if τ_l and figure τ_h are in a 3:2 ratio, then:

$$\tau_l - \tau_h = \frac{\tau_l}{3} = \frac{\tau_h}{2}$$

and thus

$$V_{u\ unfold} = 3V_{ul} = 2V_{uh}$$

The angle Ø represents a velocity phase angle in [- π, π] , but with respect to an enlarged unambiguous interval. Thus, by simply differencing the folded angles from the high and low PRFs, we obtain an angle that is unfolded to a larger velocity span. Similar reasoning shows that the 4:3 ratio gives a factor of three improvement over V_{uh} , and 5:4 gives a factor of four.

In practice, the unfolded angle Ø is not in itself a suitable velocity estimator. The reason is that the variance of Ø is equal to the sum of the variances of each of its components, that is, twice that of the individual measurements alone. If the target is at all noisy, then this increase in variance can be severe. Rather than use Ø directly, the RVP8 uses it only as a rough estimate in determining how to unfold the individual velocity measured from each PRF.

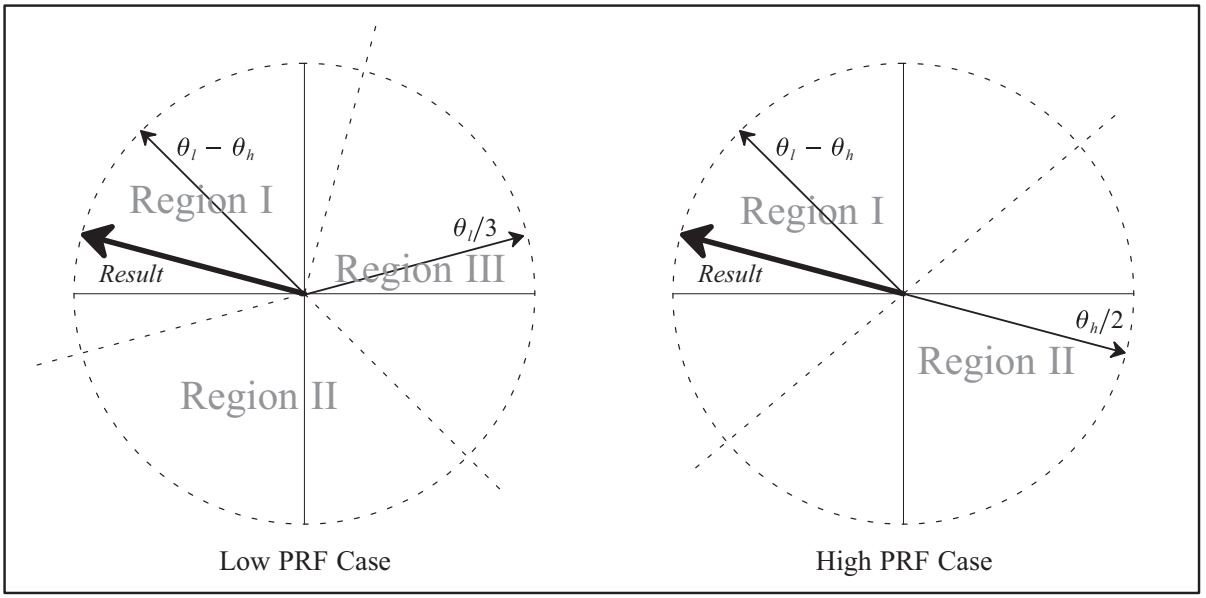


Figure 44 Dual PRF Concepts

This technique is illustrated in Figure 44 on page 262. The figure shows how the low-PRF and high-PRF angles are unfolded based on the difference angle. The diagrams show phase planes representing the large unfolded velocity interval, and the locations of various vectors on those planes. Referring first to the right figure, the difference angle is plotted, and the plane is divided into two equal size regions, one of which is centered on the difference vector. The high-PRF angle is then divided by two and plotted. The resultant unfolded velocity angle must either be this

vector, or this vector plus . Since adding places the vector into acceptance Region 1 where it is nearest the difference angle, we conclude that this is the correct unfolding. Likewise, on the left diagram we unfold the low-PRF angle by dividing the plane into thirds centered on the difference angle. The result angle is either

$$\frac{\theta_l}{3}, \frac{\theta_l}{3} + \frac{2\pi}{3} \text{ or } \frac{\theta_l}{3} + \frac{4\pi}{3}$$

depending on which one falls into the acceptance Region 1. Note that the resultant angle is the same in each case.

The RVP8 makes efficient use of the incoming data by unfolding velocities from both the low and the high-PRF data, making use each time of information in the previous ray. When low-PRF data are taken the derived velocities are unfolded by combining information from the previous high-PRF interval. Likewise, when high-PRF data are acquired the velocities are unfolded based on the previous low-PRF interval. Thus, when operating in the Dual PRF mode, the RVP8 outputs one data ray for each (N+k)-pulse interval. However, the velocity data in the Dual PRF rays are unfolded, so that the [-1,+1] interval now represents either two or three times the prior velocity range. Put another way, the data are still interpreted as described in the section on mean velocity estimation, except that V_u is now larger.

The width data are also modified somewhat during Dual PRF unfolding. Although valid widths are obtained independently on all rays, those measured at low-PRF are larger than those at high-PRF. This is simply because the dimensionless width units are with respect to a larger velocity interval in the latter case. To compensate for this, low-PRF widths are multiplied by either 2/3 or 3/4 before being output. This puts them in the same scale as the high-PRF values, and thus, the widths do not vary on alternate pulses. A useful consequence of this is that width data can be sent directly to a color display generator without having to plot every other ray in a different scale.

There are a few words of caution that should be kept in mind when using the RVP8 in the Dual PRF processing modes. The unfolding algorithms make the assumption that targets are more-or-less continuous from ray to ray. Otherwise, it would not make sense to use data from a previous ray to unfold velocities in the current ray. Users must therefore assure that their antenna scan rate and beamwidth are such that each target is illuminated, at least partially, over each full 2(N+k)-pulse interval. In practice, a certain amount of decorrelation from ray to ray is acceptable, since the previous rays are used only to decide into which unfolded interval the current ray should be placed. Small errors in the previous ray data, therefore, cause no error in the output. However, large previous-ray errors would lead to incorrect unfolding.

A more subtle side effect of Dual PRF processing arises from clutter filtering because clutter notches now appear at several locations in the unfolded velocity span, rather than just at zero velocity. These additional rejection points come about because the original velocity intervals are mapped some integer number of times to create the unfolded interval. Since each original interval has a clutter notch at DC, it follows that the final expanded velocity interval will have several such notches. For example, in the 3:2 case, in addition to removing DC the clutter filter removes velocities at - $2V_u/3$, + $2V_u/3$, and V_u .

Unfortunately, these clutter filter "images" are a fundamental consequence of the Dual PRF processing technique and are not easily removed. They can cause trouble not only for the velocity unfolding itself, but because the computed clutter corrections to be wrong at the image points. However, there is a useful work-around in the RVP8 to minimize their impact — turning the clutter filter off at far ranges where little clutter is expected and using a narrow clutter filter minimizes the effects of the clutter filter on weather targets.

The 4:3 and 5:4 PRF unfolding ratios are more susceptible to unfolding errors in cases where the spectrum width is large and/or the SNR is low. The user should experiment with these ratios to determine which provides the best results for their particular application. Although the RVP8 trigger generator can produce any trigger frequency, only the 3:2, 4:3, and 5:4 ratios can be used with the built-in unfolding algorithms. The RVP8 still permits other PRT ratios to be explored, but the unfolding technique must then be manually programmed on the user's host computer.

Oscilloscope observations of Dual PRF triggers can sometimes be confusing. Figure 45 on page 265 shows seven possible scope traces (and their associated probabilities) for the RVP8 trigger during Dual PRF operation. The PRF ratio is 4:3, and the sample size is 50 pulses at the high PRF, and 37 pulses at the low PRF. The signal labelled "SCOPE" is the composite of these traces, and is what would actually be seen on an oscilloscope. Notice that there are a number of low probability pulses. The exact details of the sample sizes and the trigger hold off time can make the low probability pulses appear to come and go randomly. This is normal, and is no cause for alarm.

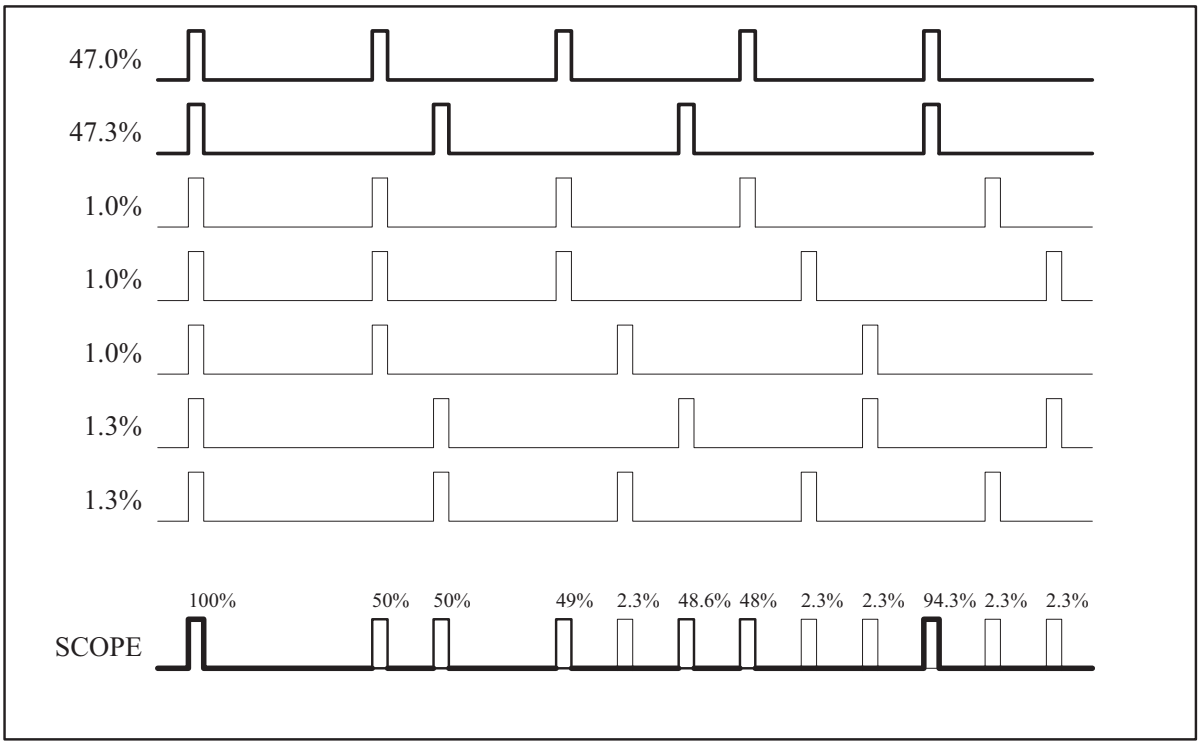


Figure 45 Example of Dual PRF Trigger Waveforms

6.8 Random Phase 2nd Trip Processing

6.8.1 Overview

Second trip echoes can be a serious problem for applications when the radar is operated at high PRF (for example, >500 Hz). Second trip echoes are caused by the range aliasing of targets. They appear as false echoes on the display, usually elongated in the radial direction. On Klystron systems they will have valid Doppler velocities. On magnetron systems, the Doppler velocities are not valid, but the noise from the 2nd trip echoes can obscure valid first trip velocity information.

The RVP8 has optional random phase processing for the filtering and recovery of second trip echoes. Details of the technique are proprietary to Vaisala, Inc. However, the general principle is described here, along with a discussion of the various configuration options to optimize the algorithm performance.

The information that is used to separate the first and second trip echoes is the phase. For a magnetron radar, the phase of each pulse is different. This means that when 1st. and 2nd trip echoes are received simultaneously, the

phase of the first trip return is different from the phase of the second trip return. For a magnetron radar, the RVP8 measures the phase of the transmitted pulse and the phase locking is done digitally as opposed to the traditionally locking COHO. For a Klystron radar, the phase is controlled by the RVP8 via a digital phase shifter that is precisely calibrated. Typically the Klystron COHO is phase shifted so that each transmit pulse has a different phase. The sequencing is controlled by the RVP8.

6.8.2 Algorithm

Figure 46 on page 270 shows a schematic of the data processing for random phase. The figure shows the Doppler spectra for the 1st. and 2nd trip in the various processing stages. The vertical scale is in dB and the horizontal scale is velocity. In this example, the second trip echo is shown as being stronger than the first trip echo (usually the reverse is true).

Ideal 1st and 2nd Trip Echoes

The ideal 1st and 2nd trip echoes represent the echoes as they would appear individually. The ideal 1st trip echo is the echo that would be measured if there were no 2nd trip echo interference. The ideal 2nd trip echo represents what would be measured if there were no 1st trip echo interference. If there is no interference from the other trip, a standard Klystron system can measure the ideal spectra, but there is no way to know whether the echoes are in the 1st or 2nd trip.

Raw 1st and 2nd Trip Echoes

This figure shows how the echoes from the first trip and second trip interfere with each other. For the case of a standard magnetron system, the first trip echo is coherent, while the second trip echo is incoherent (white noise) since the phase of the second trip echo is random. This is because the receiver is phase locked only to the first trip.

Another way to implement a magnetron system is to let the COHO free-run (rather than phase locking to the transmit pulse), measure the phase of each transmit pulse and digitally correcting for the transmit phase. Using this digital phase locking technique, the RVP8 can phase lock or "cohere" to either the first or the second trip.

Using this technique alone, it is possible to distinguish between 1st and 2nd trip echoes for the case when the echoes are not overlapped. In other words, the echoes will appear as the idealized 1st and 2nd trip echoes. This range de-aliasing effectively doubles the range of the radar. The problem is that when echoes are overlapped, the noise contamination from the stronger echo will make it impossible to measure the weaker echo. This is illustrated in the figure. Thus if the first trip echo has a good signal-to-noise

ratio of 10 dB, then the 2nd trip echo will have a signal-to noise-ratio no better than -10 dB. This is the fundamental problem with using phase alone to separate the 1st and 2nd trip echoes.

Filtered 1st and 2nd Trip Echoes

Since the strong echo generates noise that obscures the weaker echo, the approach used in the RVP8 is to filter the echo from the other trip — the whitening filter. This is shown in the figure. The adaptive whitening filter removes both the clutter and the weather. All of the phase information for the other trip is then contained in the white noise portion of the spectrum. Note that the phase information under the coherent echo that is removed will be dominated by the coherent echo, that is, the other trip phase information will be contaminated. For this reason, the filtering should effect as small a region of the spectrum as possible.

6.8.3 Tuning for Optimal Performance

The Random Phase algorithms are controlled by the same collection of setup and operational parameters that apply to all of the other processing modes, for example, choice of sample size, clutter filter, angle sync, calibration, etc. However, a few parameters are special to Random Phase mode, and these are described below.

Secondary SQI Threshold

In standard Doppler processing, an SQI threshold is normally not applied to Reflectivity data because it would cause those data to be rejected in regions of high spectral width. In Random Phase mode we need to relax this convention because reflected power can only be assigned to a particular trip when it is coherent within that trip. Incoherent echoes, regardless of their strength, can not be placed into either trip.

Thus, an SQI threshold is required to qualify Reflectivity data in Random Phase mode. The RVP8 defines a secondary SQI threshold SQI_2 which is computed from the standard threshold value simply as:

$$SQI_2 = Offset + (Slope \times SQI)$$

Where *Slope* and *Offset* are the Random Phase SQI threshold parameters defined in the **Mf** setup section. The factory default values are (*Slope* = 0.50) and (*Offset* = 0.05) , that is, the secondary threshold is a little less than half of the standard value. The Random Phase algorithms check whether the SQI of each recovered trip is less than the secondary SQI threshold, and if so, the LOG portion of the data are rejected. This SQI test is necessary

for a clean LOG picture, but we need to use a more permissive (lower) threshold value than would usually be applied to the Doppler data alone.

The *Slope* and *Offset* values should be adjusted so that the density of speckles in Random Phase LOG data is approximately the same as the density of speckles in FFT velocity data for a given primary SQI value. You may then adjust the primary SQI threshold to achieve the appropriate tradeoff of speckles vs. sensitivity for your system in all modes of operation. Even with proper adjustment, it is normal for Random Phase *dBZ* and *dBT* data to show "holes" in regions of weather that have high turbulence or shear. These dropouts will usually match up with similar gaps in the velocity and width data, both of which are traditionally thresholded by SQI.

Maximum Power Ratio Between Trips

The adaptive filtering that is performed on the data for each trip greatly extends the visibility of a weak echo that is overlapped with a much stronger one. In practice, the filtering process is often able to remove 25-35dB of dominant power in order to reveal a much weaker echo in the other trip. The performance depends on many factors, primarily the spectral width of the dominant echo, and the overall stability of the radar system.

The difficulties of removing a dominant "other trip" echo from a weather signal are analogous to the challenge of removing a dominant clutter target from that same signal. In both cases we are trying to extract a weak weather signature using a filtering procedure that relies on the spectral confinement of the stronger signal. The RVP8 already has a parameter that can be adjusted to control sub-clutter visibility, that is, the Clutter-to-Signal Ratio (CSR). Just as the CSR applies to the clutter filters, it can likewise be used to place similar limits on the depth of visibility of the adaptive filters.

As an example, suppose that the RVP8 is operating in Random Phase mode at a PRF of 1500Hz, and is observing widespread weather having uniform intensity in both the first 100Km trip and the second 100Km trip. If the CSR were set too conservatively at only 15dB, then the algorithm would generally be blind to second-trip weather in the range interval from 100km to 117.8km.

The explanation for this can be found in the $1/r^2$ geometric correction for weather echo intensity. At ranges less than 17.8km, the first trip weather would generally dominate the second trip weather by more than 15dB. Thus, the initial 17.8km ring of second trip data would be rejected by the CSR criteria. However, if the CSR were increased to 30dB, then the size of this missing ring would be reduced to only 3.2km.

If the CSR is set too low you will notice an abrupt ring of missing data in the beginning of the second trip. If set too high, there will be speckles and

other spurious effects within this same interval. The optimum setting should strike a balance between these two effects.

R1 vs. R2 Algorithms

The Random Phase algorithms for adaptive filtering and separation of trips relies on having the best possible information about the weather's SNR and spectral width. Thus, the "R2" Doppler algorithms are always used, regardless of the setting of the R1/R2 flag in the user's operational parameters.

Random Phase and Dual PRF

The random phase processing works seamlessly with the dual PRF processing to provide advanced range and velocity ambiguity resolution. Both the first and 2nd trip echoes can be recovered and displayed to a maximum range of 2X the unambiguous range corresponding to the high PRF.

For optimum performance, the 2D 3x3 speckle filter should be used to smooth the 2nd trip seams that occur for each ray. In fact, this smoothing of the 2nd trip seam makes the dual PRF random phase mode work even better than the single PRF random phase.

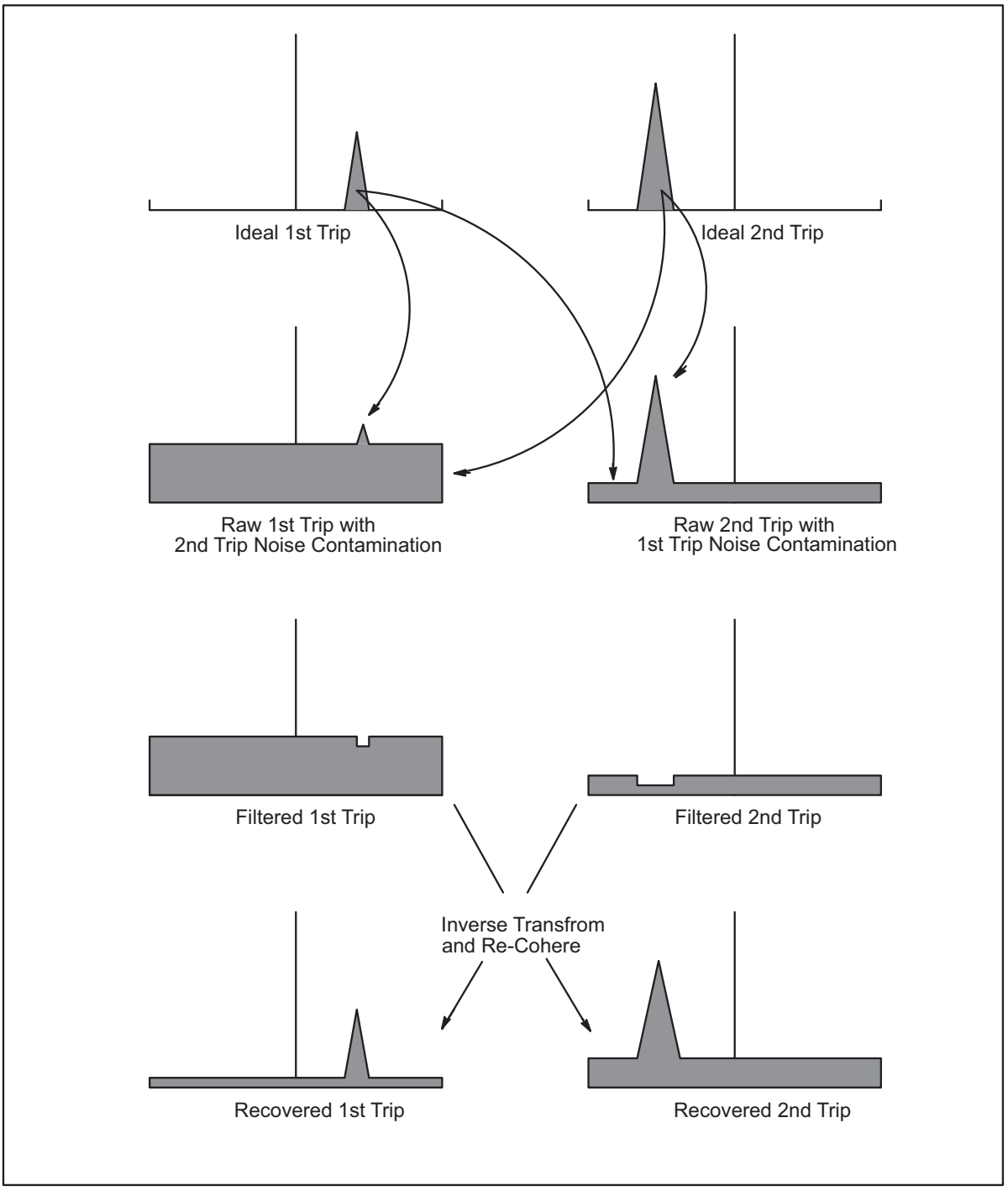


Figure 46 Random Phase Processing Algorithm

6.9 Signal Generator Testing of the Algorithms

This section describes a variety of IF signal generator tests that can be used to verify correctness of the RVP8 processing algorithms. These tests are routinely performed at Vaisala whenever new algorithms and/or major modes are added to the processor. We have include a few of the test descriptions here so that they can be used by customers who need to debug their systems, or who want to better understand how they work. Additional tests for receiver sensitivity and dynamic range can be found in Appendix D, Installation and Test Procedure, on page 427.

6.9.1 Linear Ramp of Velocity with Range

Suppose that a continuous-wave IF waveform has an instantaneous frequency $f(t)$ in Hertz (cycles/sec). Consider a range bin located at time τ_{bin} within a set of pulses that are separated by $\tau_s = 1/PRF$. The phase measured at that bin on the n^{th} pulse will be the integral of the frequency within that pulse starting from range zero (since the RVP8 is phase locked to range zero):

$$\Phi_n = \int_{n\tau_s}^{n\tau_s + \tau_{bin}} f(t)dt$$

If we assume that the input frequency is a linear Frequency Modulation (FM) at the rate of M cycles/sec/sec on top of a base frequency T_o, then:

$$\Phi_{n+1} - \Phi_n = \int_{(n+1)\tau_s}^{(n+1)\tau_s + \tau_{bin}} (T_0 + Mt)dt - \int_{n\tau_s}^{n\tau_s + \tau_{bin}} (T_0 + Mt)dt = (M\tau_s)\tau_{bin}$$

which, remarkably, is independent of both T_o and n. Thus, a linear FM input signal produces a fixed (I,Q) phase difference from pulse-to-pulse at any given range. The magnitude of the phase difference is proportional to the range, and the slope is $(M\tau_s)$ cycles for each second of delay in range. For example, if the test signal generator is sweeping 100KHz every two seconds, then the velocity observed at a range of 300km at 250Hz PRF will be:

$$\Phi_{n+1} - \Phi_n = \left(\frac{100KHz}{2sec}\right) \times \left(\frac{1}{250}sec\right) \times (300km) \times \left(\frac{6.6\mu sec}{1km}\right) = 0.40cycles$$

We would thus observe a velocity of (0.8 × V_u) at 300km, where V_u is the unambiguous Doppler velocity in meters/sec. Note that these phase difference calculations have made no assumptions about the RVP8 processing mode, and thus are valid in all major modes (PPP, FFT, DPRT, RPH), as well as in all Dual-PRF unfolding modes.

Interestingly, this simple FM signal generator will also produce valid second trip velocities that can be seen during Random Phase processing. This follows from the above analysis because we've never assumed that τ_{bin} was smaller than τ_s, that is, it is fine for the range bin to be located in any higher-order trip.

6.9.2 Verifying PHIDP and KDP

The PHIDP and KDP processing algorithms can be tested using CW signal sources at IF. In the alternating-transmitter single-receiver case, a single FM signal generator is modulated with an RVP8 polarization select line so that slightly different frequencies are generated for the H and V pulses. A maximum FM depth of several kilohertz is all that is required. In the dual-receiver case, two (unmodulated) signal generators are used for each of the H and V intermediate frequencies, and one or the other is detuned slightly from its correct center frequency. In either case the frequency difference that produces a KDP value of 1.0 degree/km will be:

$$\left(1.0\frac{degree}{km}\right) \times \left(\frac{1}{360}\frac{cycles}{degree}\right) \times \left(299792\frac{km}{second}\right) = 833\frac{cycles}{second}$$

6.9.3 Verifying RHOH, RHOV, and RHOHV

These three terms measure the normalized cross-channel covariance in a polarization radar. They all are computed in essentially the same way having the form:

$$RHOAB = \frac{< s_A^n \, s_B^{n*} >}{\sqrt{< s_A^2 > \, < s_B^2 >}}$$

Where the $S^n{}_A$ and $S^n{}_B$ are complex (I,Q) vectors from two receiver channels A and B, and "<>" denotes expected value. This suggests that some form of amplitude modulation (AM) of the input signal might be helpful.

Suppose that the $S^n{}_A$ and $S^n{}_B$ samples are coming from two signal generators installed on a dual-receiver system, and that only the B-Channel is AM modulated so that:

$$\left|s_A^n\right| = \{S_A, S_A, S_A, S_A, S_a \ldots\}, \left|s_B^n\right| = \{S_B, 0, S_B, 0, S_B \ldots\}$$

Then the above estimator reduces to:

$$RHOAB = \frac{\left(\frac{1}{2}\right) S_A S_B}{\sqrt{S^2{}_A \times \left(\frac{1}{2}\right) S^2{}_B}} = 0.707$$

A simple way to create these data is to set the A-Channel siggen for 95% AM depth, and use a sinusoidal modulation source of, perhaps, 400Hz. The reason for not choosing 100% depth is that we would loose the Burst phase reference when the amplitude became smallest. The 26dB reduction in S_B is a close enough approximation to zero in the above formula.

If we now observe the two receive channels with the RVP8 at a PRF of 800Hz, we will see the various RHOAB terms varying with range; reaching a high value of 1.00, and a low value of 0.707. The plots will be nearly stationary on the **ascope** screen because the PRF is almost precisely twice the modulation rate (though they are free-running relative to each other).

Adjusting the amplitude of either signal generator will not affect the *p* terms, but it will have an interesting effect on SQI. If (T,Z,V,W) are being computed from both channels combined, then the SQI is:

$$SQI = \frac{S_A^2}{S^2{}_A + \left(\frac{1}{2}\right) S^2{}_B}$$

If we solve this equation for SQI=0.5 we find that the individual S_A terms must have twice the power of the individual S_B terms. This can be checked by adjusting either signal generator until the minimum plotted SQI is 0.5,

and then verifying that the average *H* and *V* powers are identical; or, equivalently, that *ZDR*, *LDRH* and *LDRV* are zero.

The linear FM ramp described in 6.9.1 Linear Ramp of Velocity with Range on page 271 can also be used as a test of RHOAB in adual-receiver system. With one siggen modulated and the other fixed, one receive channel will appear to be rotating relative to the other. If the FM modulation is such that *1/N* of a full revolution occurs per pulse at a given range, then if the sample size is *N* pulses we will observe *RHOAB* = 0 at that range. In fact, the plot of *RHOAB* will show a characteristic *sin (x)/x* behavior as a function of range.

CHAPTER 7
HOST COMPUTER COMMANDS

This chapter describes the digital commands that the host computer must use to set up and control the RVP8 processor for recording data. Each command is described in detailed in a separate section of this chapter. Note that a command mnemonic, or shorthand reference name, is given in each section heading. These names are frequently used to refer to particular commands.

The write-up for each command includes a description of what the command does and a pictorial layout of the bits in the 16-bit command word. Commands consist of an initial command word containing an opcode in the low five bits. If additional arguments are required, they are listed as "Input 1", "Input 2", etc. Finally, if the command produces output, those words are listed as "Output 1", "Output 2", etc. Often each word is broken down into several independent fields, each consisting of one or more bits. In such cases, the pictorial layouts show the placement of the bit fields within the word, and each field is described individually. All data transferred to or from the RVP8 are in the form of 16-bit words.

Before attempting to program the RVP8, it is a good idea to at least skim through the descriptions of every command. The instruction set has been designed to be as concise and orthogonal as possible. User programs should always execute the IOTEST command on power-up to ensure that the interface connections are all intact. The diagnostic result registers from GPARM should also be checked initially to verify that the RVP8 passed all internal checks. Since all internal RVP8 tables and parameters are set to reasonable values on power-up, it is conceivable that PROC commands could be issued immediately to acquire and process radar data. More realistically, however, the default information is first modified to meet the users needs.

To set up for data acquisition and processing the following sequence of commands might be executed. Trigger and pulse width are first established using the SETPWF commands. Range bin placement and processor

options are then chosen using LRMSK, and SOPRM, and receiver noise samples are taken with SNOISE. The noise levels are not automatically sampled on power-up, so SNOISE must be issued at least once by the user. LFILT is executed if clutter filters are needed. If data rays are to be synchronized with antenna motion, then LSYNC is used to specify a table of antenna angles. After all setups are complete, PROC commands are issued to actually collect, process, and output the data. Errors detected during the execution of commands are noted by the RVP8 and can be monitored using GPARM.

The RVP8 contains a 4096-word sfirst-in-first-out (FIFO) buffer through which all output data flow. This buffer is included to simplify the requirements of the user's interface hardware. The FIFO holds each sequential word generated by the RVP8 until such time as the user is ready to accept it. Thus, when reading from the processor, it is permissible to fall behind by as many as 4096 words before any slowdown in performance occurs. The RVP8 writes to the FIFO at full speed as long as it is not full, and the internal processing is not affected by the exact speed at which user I/O actually occurs. This continues as long as the average I/O rate on, perhaps 10ms intervals, matches the average rate at which data are being produced.

The sequence of events described above is altered when the FIFO becomes completely full. Then, when the processor generates the next output word, it waits in an idle loop until the user makes room in the FIFO by reading out one or more words. Until this space becomes available, the RVP8 simply waits and does not proceed any further with its internal processing. This, of course, leads to a slowdown in performance, but it is not a disastrous one. The user always obtains correct data no matter how long it takes to read it. One could take advantage of this fact to synchronize the acquisition of data by the RVP8 with the post-processing and display of that data by the user. In this case, RVP8 would be instructed to output data at the maximum rate, the user would read these words at the user's maximum rate, and the overall system would automatically run at the slower of those two speeds.

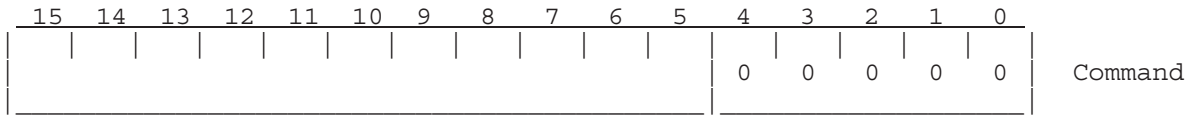
When the output FIFO is full and the RVP8 has the next word ready for output, there is another way that the idle wait loop can be exited, that is, if the processor detects that the user is performing a write I/O cycle. Since the user should have been reading data by now, the presence of a write cycle is taken to mean that some more important condition has arisen. As such, the wait loop is terminated and the RVP8 accepts the write data soon afterward. If the new data are commands, they are executed right away, but any output they try to produce may be lost in a similar manner. The net effect is that the processor continues to execute all commands correctly, but that their output is discarded.

The discarded output data are not in fact lost. Rather, the data are eventually replaced with an equal number of zeros. Each time the RVP8 discards an output word, it also increments an internal 24-bit count. When FIFO space becomes available in the future, the processor replaces all of the missing data with zero-valued placeholders.

Writing when the FIFO is full can be particularly useful if the new command is a RESET which calls for clearing of the output FIFO. When the RESET is processed, all past and present output data are discarded, leaving the RVP8 output section completely empty. This is useful whenever the processor has pending output data which the user wants to truly throw away.

7.1 No-Operation (NOP)

This single-word instruction is simply ignored by the the Signal Processor. The NOP is useful when a number of words are to be flushed through the RVP8 with no side effects.



7.2 Load Range Mask (LRMSK)

This command informs the signal processor of the ranges at which data are to be collected. An arbitrary set of range bins are selected via an 8192-bit mask. The Nth bit in the mask determines whether data are acquired and processed at a range equal to RES x (N-1). The Range resolution is specified by a TTY setup question (see 4.2.5 Mt<n>— Triggers for Pulsewidth #n on page 133), in the range 25 through 1000 meters. Any collection of ranges may be chosen from integer multiples of that distance. The example below is given for the default resolution of 125 meters. The range mask is passed to the RVP8 packed into 512 16-bit words. The least significant bit of each packed word represents the nearest range, and the most significant bit represents the furthest range in each group of 16.

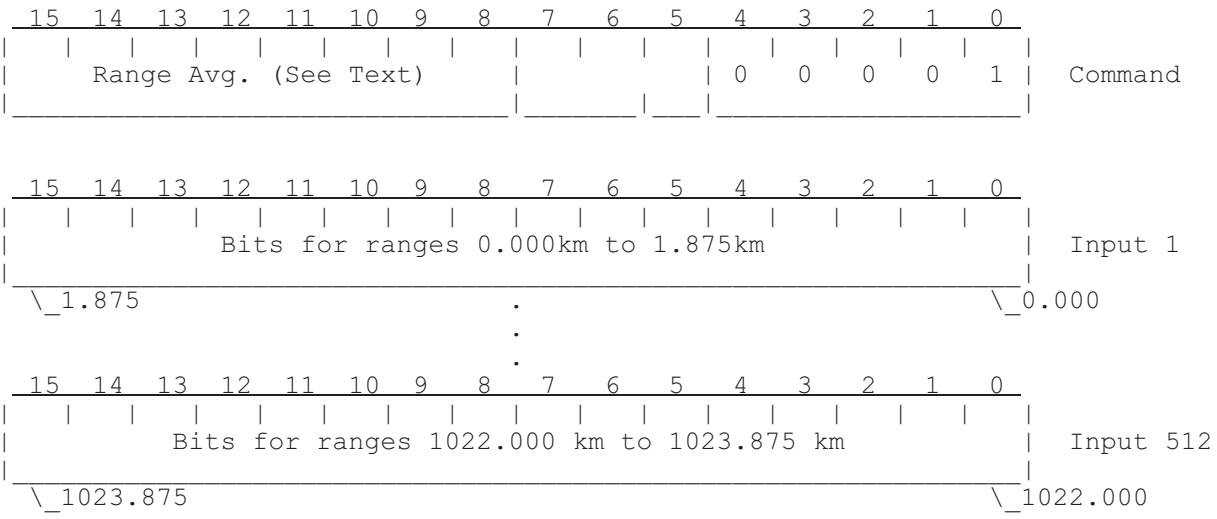
According to the range bins that are selected in the mask, the signal processor computes and stores internally a range normalization table which is later used to convert receiver intensity levels into reflectivity levels in dBZ. Note that the LRMSK command implicitly specifies the number of bins to be processed and output. The maximum bin count is

3072, though depending on the computational intensity of the configuration, the RVP8 may be able to compute fewer bins. If the number of bins selected in the bit mask exceeds this maximum, the trailing bins are truncated. If the new mask does not specify any active bins, then a single bin at range zero is forced on. The default power-up mask selects 256 bins equally spaced by 1.0km starting from zero range.

Range averaging is also determined by LRMSK. The upper byte of the command controls how many consecutive bins are grouped together. A value of zero means no averaging; one means that pairs of samples are averaged; 255 means that 256 terms are summed, etc. The individual samples that go into each average are still taken according to the bits that are set in the mask, except that they are now grouped together so that only one net bin results from the several data samples. Note that the limitation of 3072 sampled ranges applies to the bin count prior to averaging.

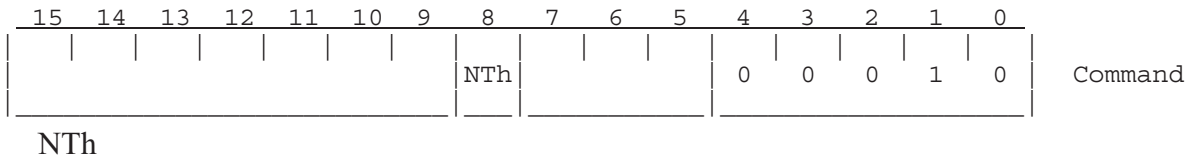
For example, suppose 100 bits are selected in the range mask and no averaging is elected. Then parameters are computed at those 100 ranges, and 100 bins of data are output. If the averaging were set to one, rather than zero, samples would still be taken at the same ranges, but pairs of bins would be averaged together and only 50 ranges would result. Note that the parameters are averaged by summing the autocorrelations for each bin. The range normalization value associated with the averaged bin is computed according to the midpoint of the first and last sample.

Incompletely averaged bins are discarded by the LRMSK command. In the above example, if the averaging were set to two so that triples of samples were summed, then only 33 bins would be output. This is because the 100-bit mask left a dangling 100th sample. In the extreme case where there are not enough mask bits to result in even one complete bin, the RVP8 forces the averaging to zero and turns on a single bin at zero range.



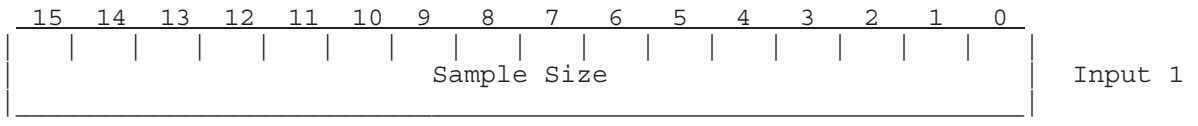
7.3 Setup Operating Parameters (SOPRM)

This command is used to configure the Signal Processor. The command should be issued whenever any of the parameters in the list change. The default parameter list consists of twenty 16-bit input words. These can be followed by optional XARG parameters as needed.

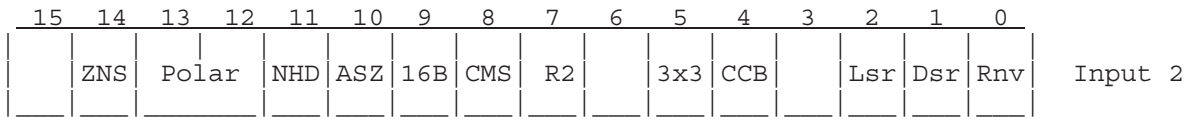


NTh

NTh If 1, then no threshold values are set. This means ignore input words 4, 5, 6, 7, 11, 12, 13, 14, and 18. This is usually used in conjunction with the THRESH command (see 7.29 Set Individual Thresholds (THRESH) on page 348), when setting individual thresholds.



The sample size is continually adjustable from 1 to 256 pulses. However, during the alternating polarization mode, the sample size must be even. If an odd value is entered it is rounded up by one in that case.



Each of the single-bit fields selects whether the given processing or threshold option is enabled (1) or disabled (0).

ZNS — If Rnv is zero (no range normalization), then setting ZNS will cause the dBZ and dBT outputs to be power relative to noise (P/N) rather than SNR ((PN)/N). This format is useful when collecting data that are near or below noise because there is no discontinuity at the noise level. ZNS has no effect when Rnv is set.

Polar — Configures transmit polarization and Zdr processing:
00 Fixed polarization, Horizontal
01 Fixed polarization, Vertical
10 Alternating polarization pulse-to-pulse
11 Dual simultaneous transmission

NHD — Disables inclusion of header words in the processed data that are output by the PROC command (See also, CFGHDR command).

ASZ — The "Any Spectrum Size" bit requests that DFT processing algorithms, clutter filters, spectral output, etc. all operate on spectra whose size exactly matches the number of available pulses (rather than rounding the spectrum size down to the next lower power-of-two).

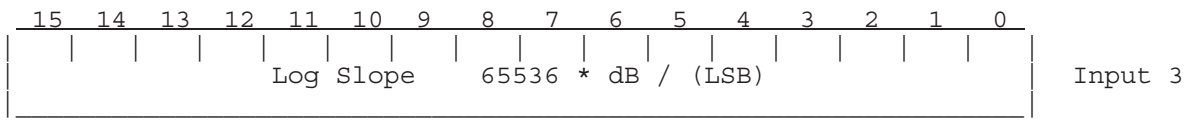
16B — Configures for 16-bit (rather than 8-bit) data output from the PROC command. This bit affects the single-parameter versions of Reflectivity, Velocity, Width, and Zdr data. However, the PROC command's archive format always holds 8-bit data, regardless of the setting of 16B. This gives the option of extracting both 8-bit and 16-bit data simultaneously from each ray.

CMS — Enables Clutter Microsuppression, in which individual range bins are rejected (based on excessive clutter) prior to being averaged together in range.

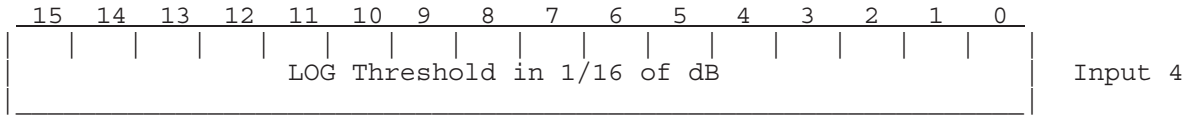
R2 — Use three lag (R0/R1/R2) algorithms for width, signal power, and clutter correction.

3×3 — Switches on the 3×3 output filter (See 6.4.3 Speckle Filters on page 248). The RVP8 automatically handles all of the pipelining overhead associated with running the 3×3 filter, that is, valid output data are always obtained in response to every PROC command.

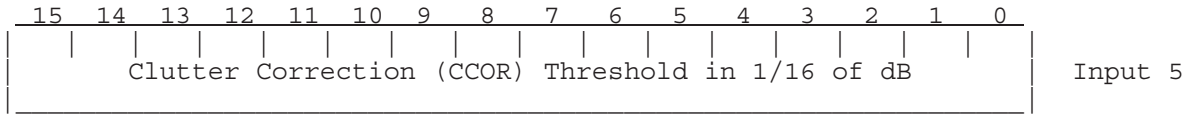
CCB	Circular Autocorrelation Bias correction. Setting this bit causes non-windowed spectra to produce autocorrelation terms that exactly match those that would be computed by traditional PPP sums, that is, with the spurious end-around term removed.
Lsr	Reflectivity speckle remover. When set, range speckles in the corrected and uncorrected reflectivity data are removed.
Dsr	Doppler speckle remover. When set, range speckles in the velocity and width data are removed.
Rnv	Range normalization of reflectivity data. This bit also enables intervening gas attenuation correction.



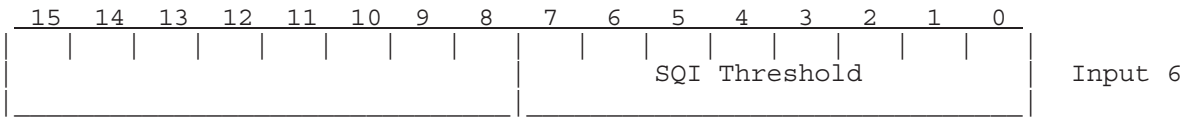
This number defines the multiplicative constant that converts the signal power in dB to the units of the 12-bit "Log of power in sample" time series outputs. One fourth of this slope is used to generate the "Log of Measured Noise Level" output from GPARM (word 6). The recommended value to use here is 0.03 (1966). This gives a dynamic range of 122 dB in 12 bits.



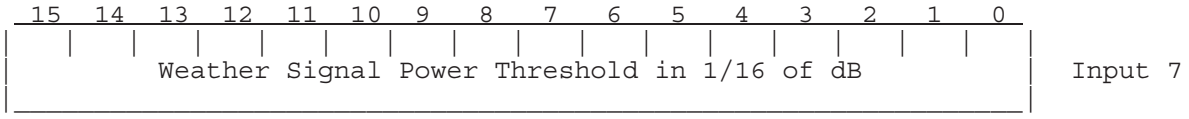
Reflectivity values below this level can result in thresholding of data, if the threshold control flags (see below) include LOG Noise bits. The threshold value is always non-negative, and the comparison test is described in section 6.4 Thresholding on page 245.



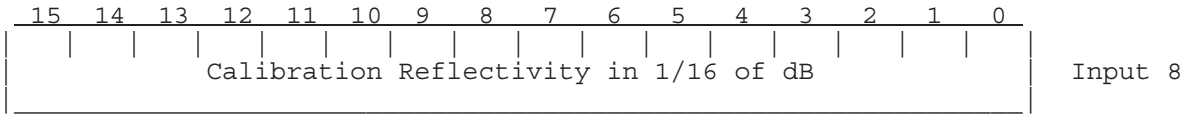
The clutter correction threshold is a bound on the computed log receiver adjustment for clutter. These corrections (in dB) are always negative. Any clutter correction which is more negative than the above value can result in thresholding of data.



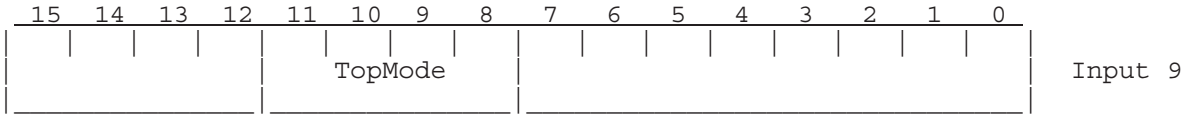
The Signal Quality Index (SQI) threshold is an unsigned binary fraction in the range 0 to 255/256. When the SQI for a range bin falls below the stated value it may result in thresholding of data.



Weather Signal Power (SIG) is an estimate of the SNR of the weather component of the received signal. When the SIG (see 6.3.8 Weather Signal Power (SIG threshold) on page 244) falls below this comparison value it may result inthresholding of data.



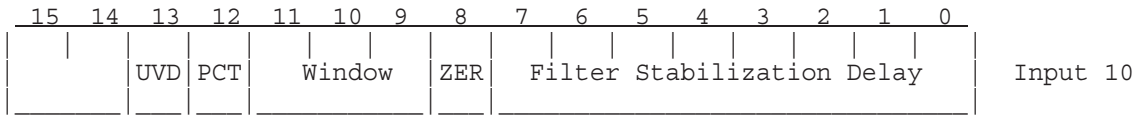
The calibration reflectivity is referenced to 1.0 kilometers.



The TopMode bits select the overall data acquisition and processing mode for the RVP8. Although the processing algorithms that are used in each top level mode are quite different, the RVP8 command set works in a uniform way in all modes.

0000	Pulse Pair Processing Mode. Doppler clutter filters are 4th-order IIR high pass; data are processed one pulse at a time as each pulse arrives (see 6.2.3 Autocorrelations on page 223).
0001	FFT Processing Mode. Doppler clutter filters use nonlinear frequency-domain approach; data are processed in batches of pulses (see 6.2.2 Frequency Domain Processing-Doppler Power Spectrum on page 220).

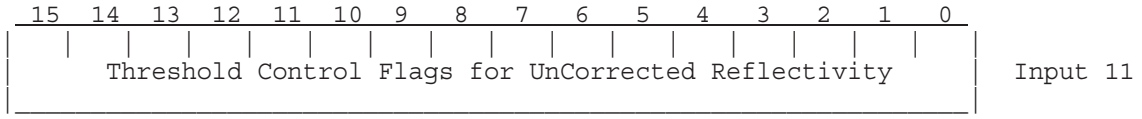
0010	Random Phase Processing Mode. Data from first and second trips are dealiased in range based on knowledge of the radar transmitter phase (see 6.8 Random Phase 2nd Trip Processing on page 265).
0100	DPRT-1 Processing Mode. The trigger generator produces alternate short and long pulses, and Doppler autocorrelations are computed using only the short pairs (see 6.6 Dual PRT Processing Mode on page 258).
0101	DPRT-2 Processing Mode. The trigger generator produces alternate short and long pulses, and Doppler autocorrelations are computed using both pairs (see 6.6 Dual PRT Processing Mode on page 258).
11XX	Four codes reserved for custom user modes.



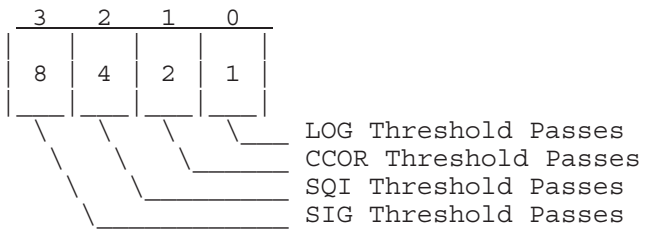
The RVP8 clutter filters are controlled by this word.

Delay	This delay is introduced prior to processing the next ray of data whenever Dual- PRF velocity unfolding is enabled or the RVP8 has been reconfigured by user commands. The delay permits the clutter filter transients to settle down following PRF and gain switches. The value is specified as the number of pulses, and hence, the number of filter iterations, to wait.
ZER	If set, then the clutter filter's internal state variables are zeroed prior to waiting the delay time. For some signal conditions, this may give better results than allowing the filter to naturally flow into the new data.
Window	Selects the type of window that is applied to time series data prior to computing power spectra via a DFT. Choices are: 0:Rectangular, 1:Hamming, 2:Blackman, 3:Exact Blackman, 4:VonHann.
Window	Selects the type of window that is applied to time series data prior to computing power spectra via a DFT. Choices are: 0:Rectangular, 1:Hamming, 2:Blackman, 3:Exact Blackman, 4:VonHann.
PCT	If set, the RVP8 will attempt to run its standard processing algorithms even when a custom trigger pattern has been selected via the SETPWF command.

UVD Unfold velocities using a simple *(Vhigh Vlow)* algorithm, rather than the standard algorithm described in 6.7 Dual PRF Velocity Unfolding on page 260.



These flags select which threshold comparisons result in unCorrected reflectivity being accepted or rejected at each bin. There are four test comparisons that are made at each range, as described above for input words 4, 5, 6, and 7. Each test either passes and produces a code of 1, 2, 4, and 8 respectively, or fails and produces a code of zero. The sum of the codes for each of the four tests is a number between 0 and 15, which can also be interpreted as the following four-bit binary number:



The individual bits of the Threshold Control Flag word each specify whether data are to be accepted (1) or rejected (0) in each of the sixteen possible combinations of threshold outcomes. Thus, the pattern of bits in the flag word actually represents a truth table for a given logical function of the four threshold outcomes.

The following examples show actual values of the Flag word for the stated combinations of acceptance criteria:

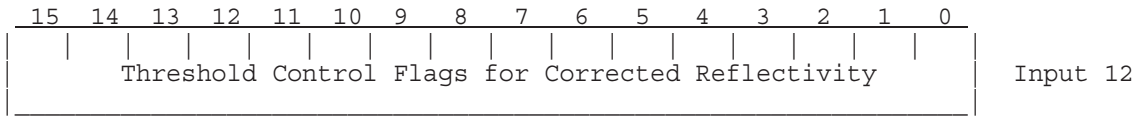
Value	Criteria
FFFF	All Pass (Thresholds disabled)
0000	All Fail (No data are passed)
AAAA	LOG
8888	LOG and CSR
A0A0	LOG and SQI
8080	LOG and CSR and SQI
F0F0	SQI
FAFA	SQI or LOG
C0C0	SQI and CSR
F000	SQI and SIG
C000	SQI and SIG and CSR
FFF0	SQI or SIG
CCC0	(SQI or SIG) and CSR

A simple way to generate these values is to imagine four 16-bit quantities having the following names and values: LOG=AAAA, CSR=CCCC, SQI=F0F0, SIG=FF00. The flag value needed to represent a given logical combination of threshold outcomes is obtained as the result when that same logical combination is applied to these special numbers.

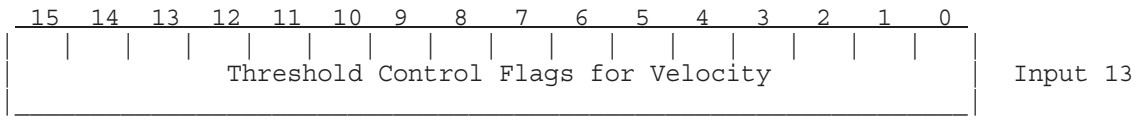
For example:

```
(SQI or SIG) and CSR    =    (F0F0 or FF00 ) and CCCC
                        =    (FFF0) and CCCC
                        =    CCC0
```

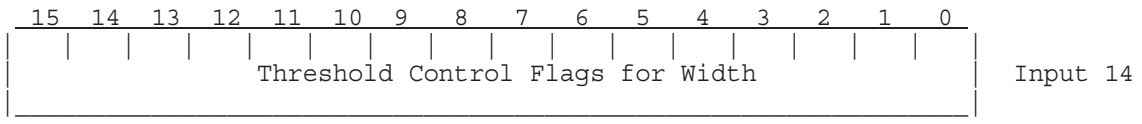
which corresponds with one of the examples given above.



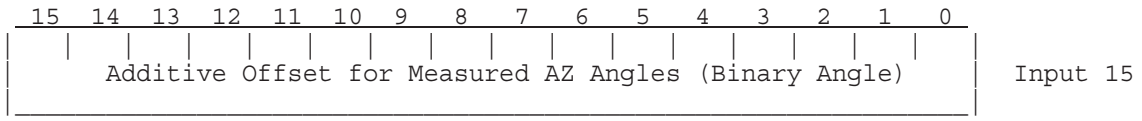
See Description for Input #11.

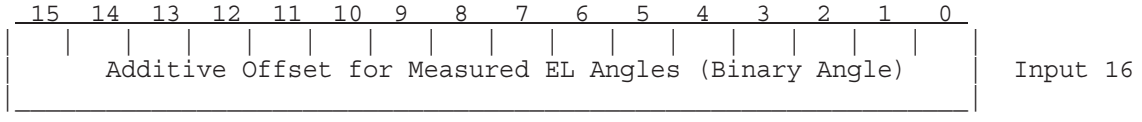


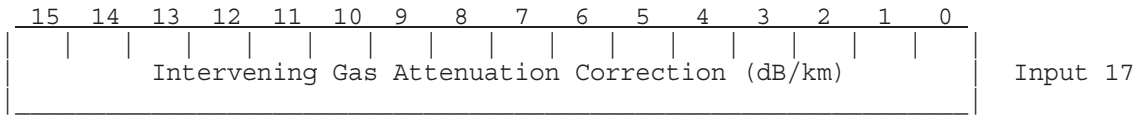
See Description for Input #11.



See Description for Input #11.







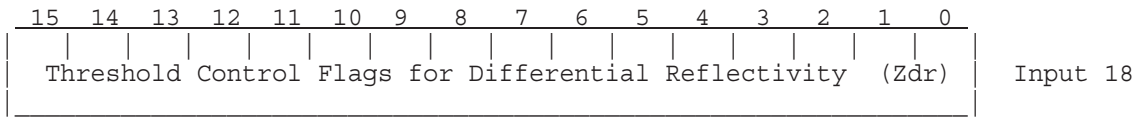
Gas attenuation correction attempts to compensate for overall (two-way) beam losses due to absorption by atmospheric gasses. The correction is linear with range, and is added to the data along with range normalization. Therefore, clearing the RNV bit in Word #2 above disables the correction. Of course, gas attenuation compensation can still be turned off even when RNV is on, simply by setting a slope of 0.0 dB/km.

An attenuation of G db/km is encoded into the unsigned 16-bit word N as follows:

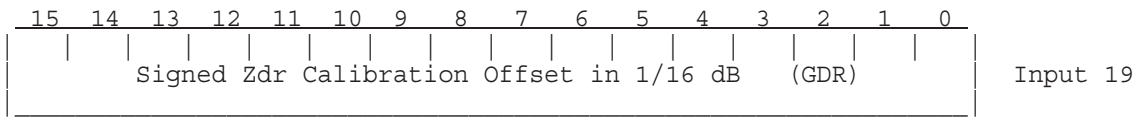
$0 \leq N \leq 10000$ $G = N / 100000$

else $G = 0.1 + (N - 10000)/10000$

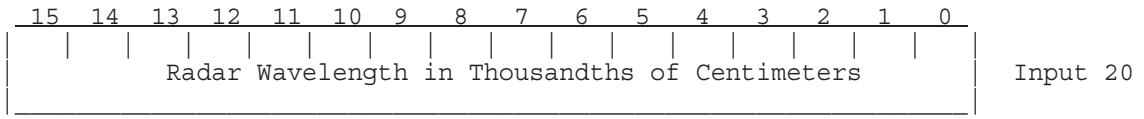
This format is backward compatible with the previous linear format for all values between 0.0 and 0.1dB/km; but it extends the upper range of values from 0.65535 up to 5.6535. These larger attenuation corrections are needed for very short wavelength radars.



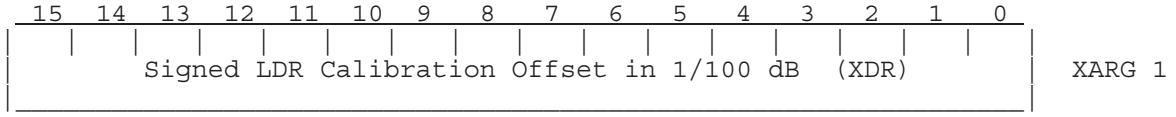
See Description for Input #11.



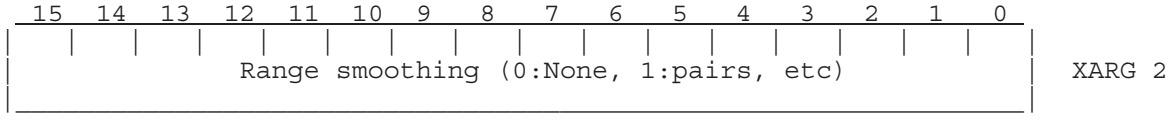
When differential reflectivity is computed there is a possibility that radar asymmetries will introduce a bias in the Zdr values, that is, that Zdr will be non-zero even when observing purely spherical targets. This calibration offset permits nulling out this effect. The GDR offset accounts for the overall Tx/Rx gain imbalance between the two channels of the radar.



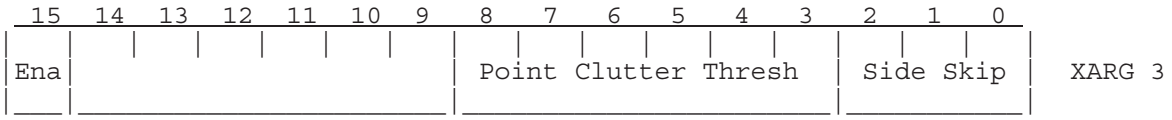
The radar wavelength is used in the calculation of 16-bit velocity and width data, to convert from Nyquist units to absolute physical units.



The XDR offset is used in the Linear Depolarization Ratio equations, and is the differential receiver gain between the two channels. Note that unlike the GDR offset (used for ZDR), the gain difference does not depend on differential transmit power.

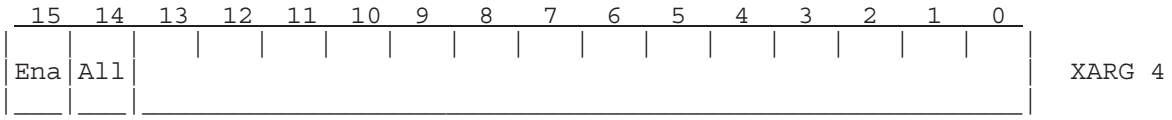


Range smoothing can be performed on raw moment data prior to the computation of scientific parameters. The number of bins to sum together is given here. This should generally be an odd integer so that no range bias is introduced by the smoothing operation.



Point clutter detection is configured with this word. A bin will be flagged as containing clutter if it's power exceeds that of its two neighboring bins by more than the detection threshold (in deciBels). Up to seven bins may optionally be skipped on each side of the central bin prior to making these two comparisons.

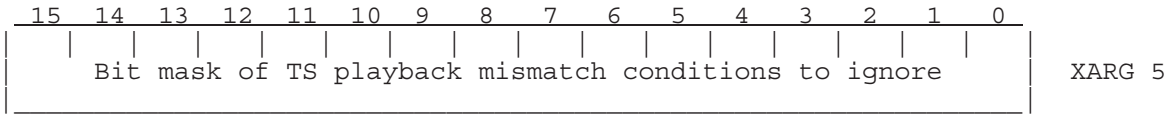
Ena — This bit is set to enable point clutter detection. Flag bits will then be reported in the "Flg" output data type of the PROC command.



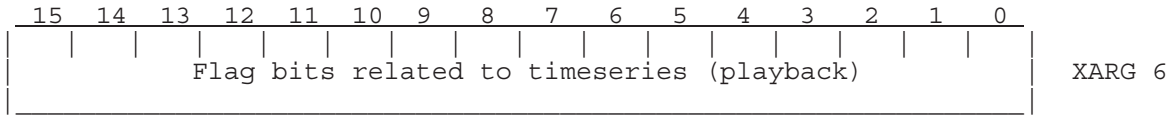
Point clutter censoring is configured with this word.

Ena — This is set to enable point clutter censoring. Raw moment data containing point clutter will be interpolated from valid signal levels on either side.

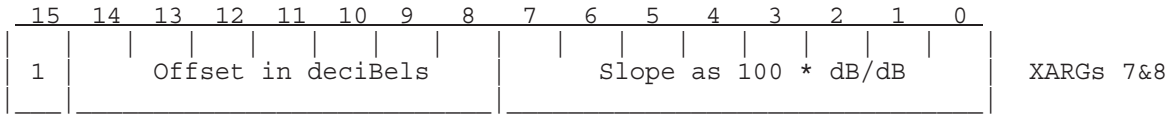
All — Optionally expand the reported detection flags to show the entire replaced interval, not just the original detected bins. This gives a more honest view of the data bins that have been altered.



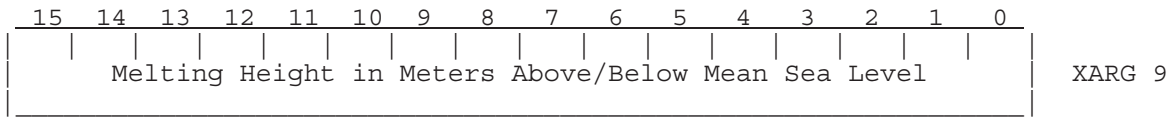
This word is a combination of MMTS_xxx bits specifying what types of mismatches are okay (do not cause an all-zero ray to be produced) during PROC command processing of timeseries data that are played back from an external source into the RVP8.



Combination of OPTS_xxx bits which modify details of timeseries behavior.



These two words allow you to set the breakpoints and slopes that modify the LOG threshold according to the Clutter-to-Noise ratio of the target. This makes the LOG threshold behave properly even as the noise floor becomes elevated due to very strong clutter targets. A value of zero will restore the RVP8 defaults from the **Mf** menu.



Note that during time series playback, the height recorded with the time series is used instead of this value.

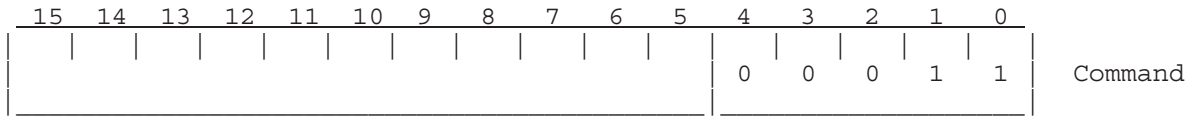
The default (power-up) values for the above parameters are listed below. Both the scientific units and the integer-input required by the command to set up that value are given. Most of these defaults will likely be reasonable for a wide variety of radars.

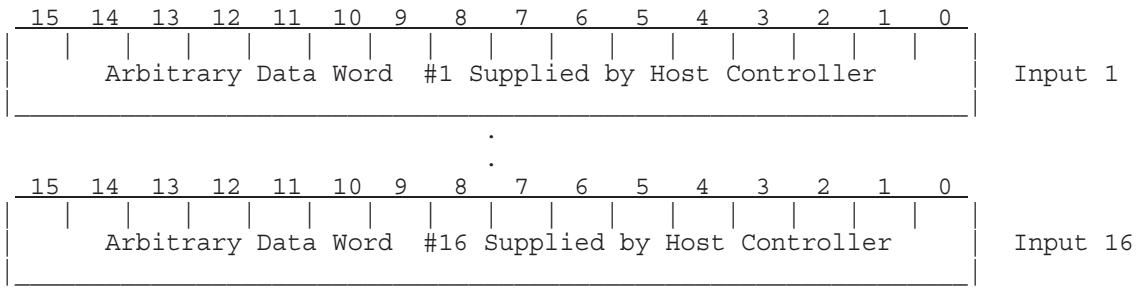
Table 19 Default Values For Operating Parameters

Parameter	Scientific Units	Input
Sample Size	25 pulses	25
Flag Word		0007 Hex
Log Slope	0.03 dB/LSB	1966
LOG Threshold	0.5 dB	8
CCOR Threshold	25.0 dB	400
Signal Quality Index Threshold	0.5 (dimensionless)	128
SIG Threshold	10.0 dB	160
Calibration Reflectivity	22.0 dBZ	352
Gas Attenuation	0.016 dB/km	1600
Zdr Offset (GDR)	0.0 dB	0
LDR Offset (XDR)	0.0 dB	0
AGC Integration Period	8 pulses	8
Radar Wavelength	5.3 cm.	5300
Dual PRF Filter Stabilization	10 pulses	10
UnCor Refl. Thresh. Control Flag	LOG	AAAA Hex
Cor Refl. Thresh. Control Flag	LOG & CSR	8888 Hex
Velocity Thresh. Control Flag	SQI & CSR	C0C0 Hex
Width Thresh. Control Flag	SQI & CSR & SIG	C000 Hex
Zdr Refl. Thresh. Control Flag	LOG	AAAA Hex
AZ/EL Angle Offsets	0 degrees	0000 Hex
Altitude of radar	0 meters MSL	0

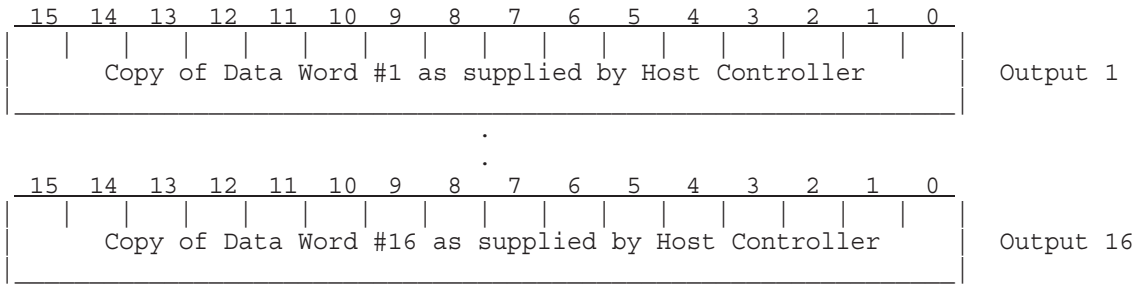
7.4 Interface Input/Output Test (IOTEST)

This command is used to test both the input and output data busses of the signal processor interface. When issued, the command causes sixteen words to be read from the host controller, after which those same sixteen words are written back out. Typically, the controller supplies a "barber pole" input sequence consisting, for example, of successive powers of two. If all of the output words are correct, one may conclude that there are no malfunctioning bits in the interface hardware.



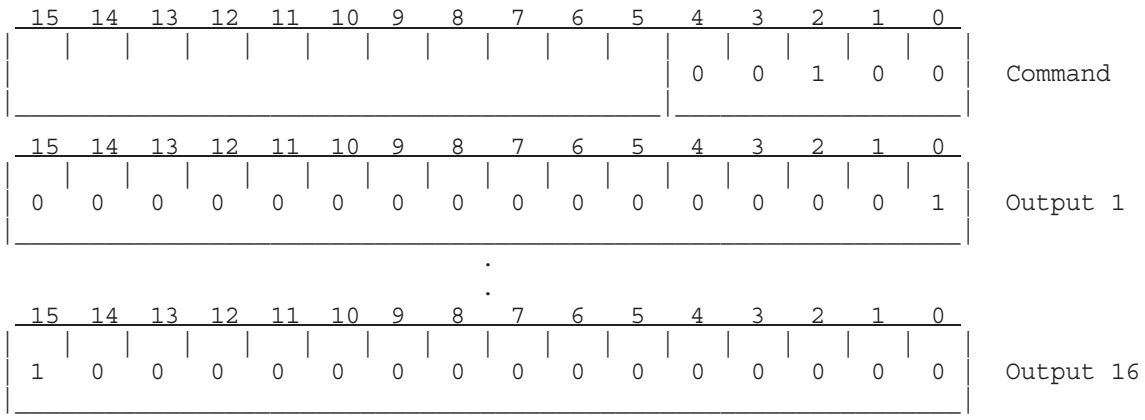


NOTE The IOTEST command can also process and echo up to 128 additional XARGS data words (see 7.20 Pass Auxiliary Arguments to Opcodes (XARGS) on page 339).



7.5 Interface Output Test (OTEST)

This command is used to test the integrity of the data being output by the signal processor. The command causes sixteen words to be output consisting of successive powers of two starting from one. By verifying whether each output word is correct, malfunctioning bits in the interface data bus can easily be isolated. This test is less stringent than the input/output test IOTEST, since the input data paths to the processor are not being checked. Typically, the OTEST is performed only when the IOTEST fails, and then to determine whether the fault was on input or output.



7.6 Sample Noise Level (SNOISE)

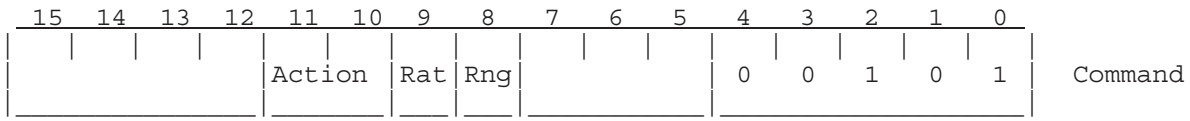
This command is used to estimate the current noise level from the receiver, so that the noise can be subtracted from subsequent measurements. Data are sampled for 256 pulses at 256 bins, beginning at a selectable range and spaced by the range resolution at that pulse width. The internal trigger generator is temporarily set to a special noise rate (usually much lower than the operating rate) during the process. It is ultimately the user's responsibility to insure that no returned power is present within the approximately 32km sampling interval. In some cases it may be necessary to raise the antenna during the noise measurement to avoid thermal noise pickup from the ground, or from weather targets.

SNOISE has the option of setting up a new sampling range and trigger generator rate each time it is called. Two bits in the command word determine which (if any) of the new values overrides the current values stored in the RVP8. The power-up sampling range is 250km (input value of 250), and the power-up trigger rate is 200Hz (input value of 30000). These initial values persist until such time as they are altered here. Note that both input words must always be supplied after the command, even if the command calls for ignoring one or both of them. The range is supplied directly in kilometers up to a maximum of 992km. The trigger rate resulting from a given input is 6MHz divided by the input value, that is, the input value is the trigger period in 0.1667 microsecond increments. Keep in mind that the given rate is bounded against the minimum PRT allowed for the current radar pulse width.

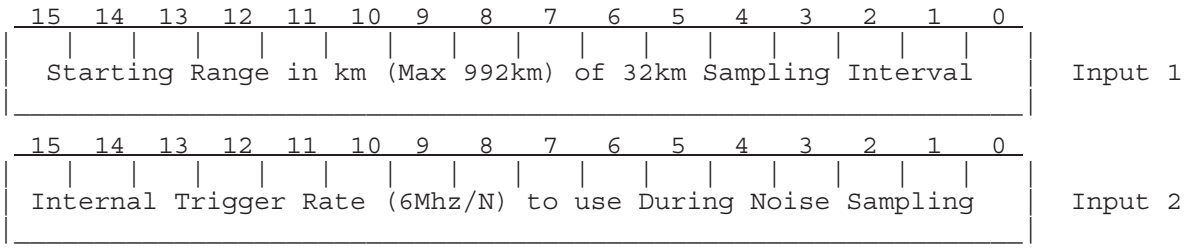
The SNOISE command bounds the requested starting range of the noise sampling interval. This is to insure that the noise samples will fit within the specified PRT, and within the range mask hardware RAM. The RVP8 sets an error bit when an improper range is requested.

The SNOISE command should be re-issued now and then to compensate for drift in the RF and A/D systems. However, because DC offsets do not propagate into the "I" and "Q" values, reissuing the command is much less critical than with the RVP6. The noise levels must be measured for the RVP8 to properly process data. This can be done by issuing the SNOISE at least once after power-up, or by setting the correct values for the powerup noise levels with the "mt" setup command, see 4.2.5 Mt<n>— Triggers for Pulsewidth #n on page 133. The RVP8 does not automatically take a noise sample as part of its initialization procedure.

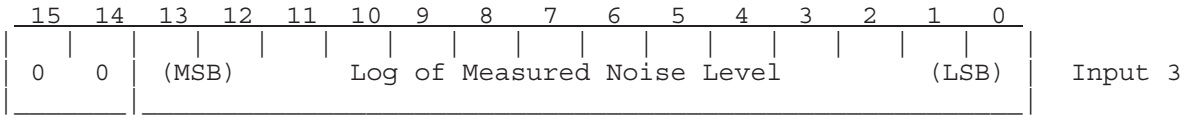
The measured offsets are stored internally for all subsequent uses inside the RVP8. The offset values may be inspected via the GPARM command, as may the current range and rate values themselves. Of course, whenever the range or rate are changed the user must ensure that the new trigger rate allows at least 32km following the new noise range. If this requirement is not met, or if other failures are detected during the noise measurement, appropriate bits are set in the GPARM latched status word. This word should generally be checked after SNOISE to make sure that everything worked properly.



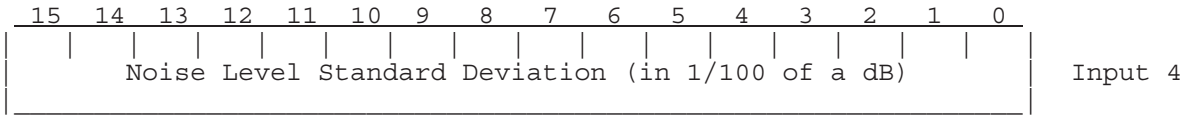
Rng	If 1, then the range in input word 1 is taken as the starting noise range for this and all subsequent SNOISE calls.
Rat	If 1, then the trigger rate in input word 2 is taken as the noise rate for this and all subsequent SNOISE calls.
Action	Specifies what action is carried out by the command.
0	Compute a new noise sample based on the present IFD input signals.
1	Do not compute a noise sample, but rather, read new noise values from the host computer and use them for subsequent processing. Four additional input words supply the noise information, and GPARM words 6, 9, and 44-50 will be changed to reflect the new noise settings.
2	Do not compute a noise sample, but rather, restore the powerup noise defaults.



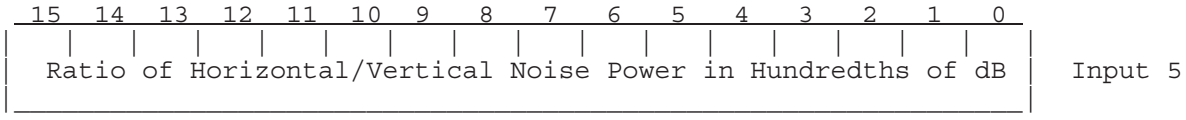
The following input words are optional, only if Action=1.



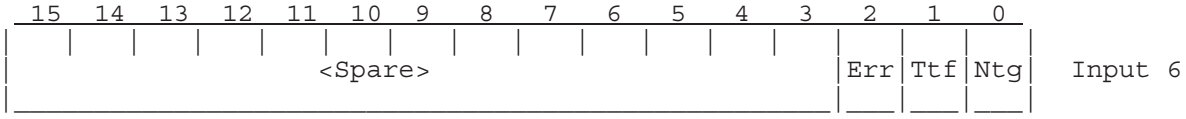
The is the same number as GPARM output 6. See the discussion in Sections 7.7 Initiate Processing (PROC) on page 295 and 7.9 Get Processor Parameters (GPARM) on page 309.



This is the same number as the GPARM output 49.



This is the same number as the GPARM output 50.



These fault bits will then be output in the latched status GPARM word 9 .

Ntg	No Trigger during noise measurement.
Ttf	Trigger too fast during noise measurement, that is, some of the noise sample bins were positioned past the trigger range.
Err	Error detected during the SNOISE command.

7.7 Initiate Processing (PROC)

The PROC command controls the actual processing and output of radar data. The operating modes and types of data available from the RVP8 are described in detail in Chapter 1. That section also describes the proper use and application of the RVP8 to different radar environments.

PROC is a single-word command that specifies the type of processing to be performed, and the type of output to be generated. The two mode bits in the command word select either

- Synchronous mode — The processor acquires, processes, and outputs one ray in response to each PROC command. Processing is begun only after each command is actually received.
- Free running mode — A single PROC command is issued and rays are continually output as fast as they can be produced and consumed. This continues until any other command is written, for example, a NOP could be used to terminate the free running mode with no other consequences.
- Time Series mode — Always produced in a synchronous manner, this mode require a new PROC command to initiate each new set of samples. Data are output as 8-bit time series, 16-bit time series, or 16-bit power spectra.

Optional Dual-PRF velocity unfolding is chosen by command bits eight and nine. For Doppler data either a 2:3, 3:4, or 4:5 PRF unfolding ratio may be selected. The RVP8 carries out all of the unfolding steps internally, so that mean velocity is now output with respect to the larger unambiguous interval. There is no additional velocity processing needed by the user, except of course, to change the velocity scale on any displays being generated. Furthermore, spectral widths are scaled consistently with respect to the higher PRF, and require no user modification before being plotted.

When unfolding is selected, the internal trigger generator automatically switches rates on alternate rays. The switch over occurs immediately after the last pulse of the current ray has been acquired; thus overlapping the internal post-processing and output time, with transmitter stabilization and data acquisition at the new rate.

Output data are selected by the upper six bits of the PROC command. Packed archive output is selected by setting the ARC bit. Individual byte or word display output is selected by setting any or all of the Z, T, V, W, Zdr, and KDP command word bits. When more than one of these bits is set, the output array consists of all of the bins for the leftmost selected parameter, followed by all of the bins for the next selected parameter, etc. Bits selected in XARG #1 behave the same way, except that the output

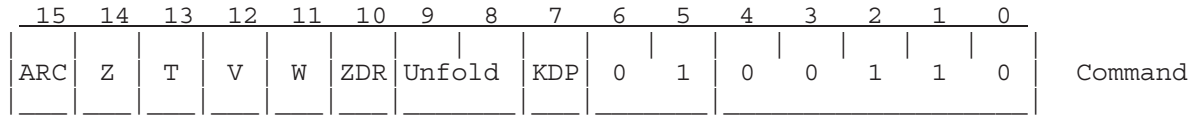
order is right-to-left. Both archive and display formats can be selected simultaneously, in which case the archive format is output first, followed by whichever individual display format values were also selected. The archive format is not recommended for use with new drivers because it can only handle four of the many possible output parameter types.

When time series mode is selected there are three output data formats available. For backwards compatibility, there is an 8-bit integer format in which the eight most significant bits from the I, Q, and LOG signals are represented in a byte. This format is not recommended because it will generally miss weak signals. We recommend the floating-point format that uses 16-bits per A/D sample. There is also a 16-bit power spectrum output that is accurate to 0.01dB. (See also GPARM output word #10).

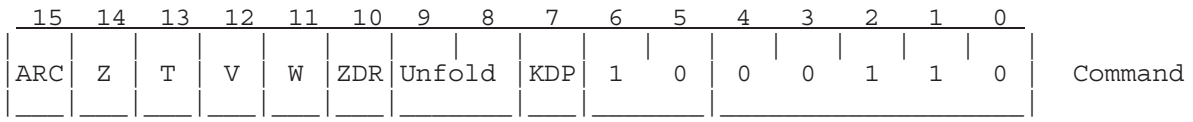
In addition to the above output data, the first words of each ray optionally contain additional information about the ray itself. These header words are configured by the CFGHDR opcode, and are included only if the NHD (No-Headers) bit in SOPRM Input #2 is clear. For example, if TAG angle headers are requested, if the ARC, Z and V bits are all set, and if there are 100 bins selected in the current range mask, then each RVP8 output ray consists of the following:

```
1]              TAG15  TAG0       \                From Start of
                                                   Acquisition
2]              TAG31  TAG16      /                 Interval
3]              TAG15  TAG0       \                From End of
                                                   Acquisition
4]              TAG31  TAG16      /                 Interval
*               200 words of packed archive data,
*               100 words of Corrected Reflectivity data in low byte
                only.
*               100 words of Velocity data in low byte only,
```

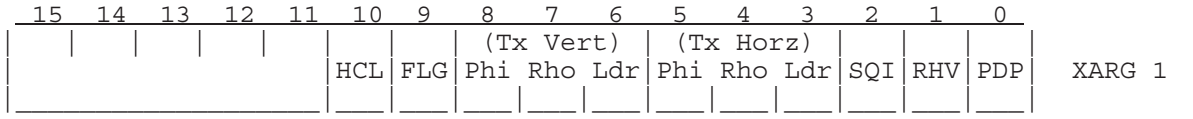
The Command word format for Synchronous Doppler Mode is:



The Command word format for Free Running Doppler Mode is:



Either of these may be augmented by an optional XARG word (See 7.20 Pass Auxiliary Arguments to Opcodes (XARGS) on page 339)

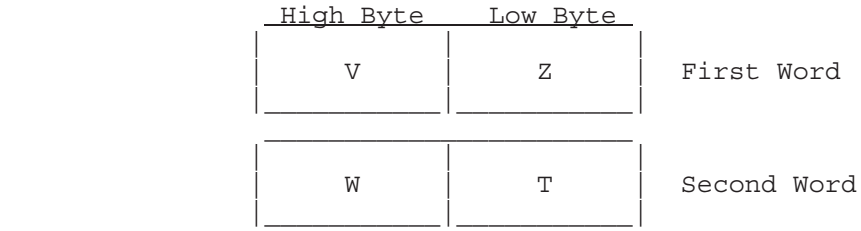


Unfold Selects DualPRF unfolding scheme:

00 : No Unfolding 01 : Ratio of 2:3

10 : Ratio of 3:4: 11: Ratio of 4:5

ARC Selects archive output format in which four data bytes (see 8-Bit descriptions below) are packed into two output words per bin as follows:



The remaining data parameters are available in both 8-Bit and 16-bit formats, according to SOPRM Command input word #2 (See 7.3 Setup Operating Parameters (SOPRM) on page 279). The same SOPRM word configures the RVP8 for Single or Dual polarization. The later is required for KDP, PDP, and RHV to be computed properly.

V — Selects radial velocity data.

8-Bit Velocity Format Mean velocity, expressed as a fraction of the unambiguous velocity interval, is computed from the unsigned byte N as:
0 : Indicates velocity data is not available at this range
1 : Maximum velocity towards the radar
128 : Zero velocity
255 : Maximum velocity away from the radar
When velocity unfolding is selected, the output is still interpreted as above, except that the unambiguous interval is increased by factors of 2, 3, and 4 for for 2:3, 3:4, and 4:5 unfolding.

16-Bit Velocity Format Mean velocity in meters/second is computed from the unsigned word N as:
The overall range is from 327.67m/sec to +327.66m/sec in one centimeter/second steps as follows:
0 : Indicates velocity data is not available at this range
1 : 327.67 m/sec (towards the radar)
32768 : 0.00 m/sec
65534 : +327.66 m/sec (away from the radar)
65535 : Reserved Code

W — Selects spectral width data.

8-Bit Width Format Spectral width is computed from the unsigned byte N as:
The overall range is a fraction between 1/256 to 255/256 of the unambiguous interval. The code of zero indicates that width data was not available at this range.

16-Bit Width Format Spectral width in meters/second is computed from the unsigned word N as:
The overall range is from 0.01m/sec to 655.34m/sec in one centimeter/second steps as follows:
0 : Indicates width data is not available at this range
1 : 0.01 m/sec
65535 : Reserved Code

Z — Selects clutter corrected reflectivity data.

8-Bit deciBel Format The level in decibels is computed from the unsigned byte N as:
dBZ = (N64)/2.
The overall range is therefore from 31.5 dBZ to +95.5 dBZ in half-dB steps as follows:
0 : Indicates no reflectivity data available at this range
1 : 31.5 dBZ
64 : 0.0 dBZ
128 : 32.0 dBZ
255 : +95.5 dBZ

16-Bit deciBel Format The level in decibels is computed from the unsigned word N as:
dBZ = (N32768) / 100
The overall range is from 327.67dB to +327.66dB in 1/100dB steps as follows:
0 : Indicates no reflectivity data available at this range
1 : 327.67 dBZ
32768 : 0.00 dBZ
65534 : +327.66 dBZ
65535 : Reserved Code

T — Selects total reflectivity. Same 8-bit and 16-bit coding formats as for clutter corrected reflectivity above.

ZDR — Selects differential reflectivity data.

8-Bit ZDR Format The level in decibels is computed from the unsigned byte N as:
dB = (N128) / 16
The overall range is from 7.935dB to +7.935dB in one-sixteenth dB steps as follows:
0 : Indicates no reflectivity data available at this range
1 : 7.9375 dB
128 : 0.0000 dB
255 : +7.9375 dB

16-Bit ZDR Format Same as 16-bit deciBel format.

KDP — Selects dual polarization specific differential phase data.

8-Bit KDP Format Values are coded into an unsigned byte using a logarithmic scale. The KDP angles are multiplied by the wavelength in cm. (to reduce dynamic range) and then converted to a log scale separately for both signs. The minimum value is 0.25 deg*cm/km, and the maximum value is 150.0 deg*cm/km. A code of zero represents no data, and a code of 128 represents 0 deg*cm/km. The conversion equation for positive values (codes from 129 to 255) is:
The conversion equation for negative values (codes from 1 to 127) is:

16-Bit KDP Format Same as 16-bit deciBel format, except that the units are hundredths of degrees per kilometer. No weighting by wavelength is introduced.

PDP — Selects dual polarization differential phase φ_{DP} data.

8-Bit φ_{DP}Format The phase angle in degrees is computed on a 180-degree interval from the unsigned byte N as:
0 : Indicates no φ_{DP} data available at this range
1 : 0.00 deg
254 : 179.29 deg
255 : Reserved Code

16-Bit φ_{DP}Format The phase angle in degrees is computed on a 360-degree interval from the unsigned word N as:
0 : Indicates no φ_{DP} data available at this range
1 : 0.00 deg
65534 : 359.995 deg
65535 : Reserved Code

RHV — Selects dual polarization correlation coefficient ρ_{HV} data.

8-Bit ρ_{HV}Format The correlation coefficient is computed on the interval 0.0 to 1.0 using a square root weighting of the unsigned byte N as:
0 : Indicates no ρ_{HV} data available at this range
1 : 0.0000 (dimensionless)
2 : 0.0629
253 : 0.9980
254 : 1.0000
255 : Reserved Code

16-Bit ρ_{HV}Format The correlation coefficient is computed on the interval 0.0 to 1.0 linearly from the unsigned word N as:
0 : Indicates no ρ_{HV} data available at this range
1 : 0.0 (dimensionless)
65534 : 1.0
65535 : Reserved Code

SQI — Selects Signal Quality Index data. This dimensionless parameter uses the same 8-bit and 16-bit data formats as RHV (ρ_{HV}).

LDR Selects Linear Depolarization Ratio, measured either on the horizontal receive channel while transmitting vertically, or on the vertical receive channel while transmitting horizontally.
8-Bit LDR Format The level in decibels is computed from the unsigned byte N as:
dB = 45.0 + (N1) / 5
This spans an asymmetric interval around zero decibels, and allows for cross channel isolation as large as 45dB. The overall range is from 45.0dB to +5.6dB in 0.2dB steps as follows:
0 : Indicates no LDR data available at this range
1 : 45.0 dB
226 : 0.0 dB
254 : +5.6 dB
255 : Reserved Code
16-Bit LDR Format Same as 16-bit deciBel format.

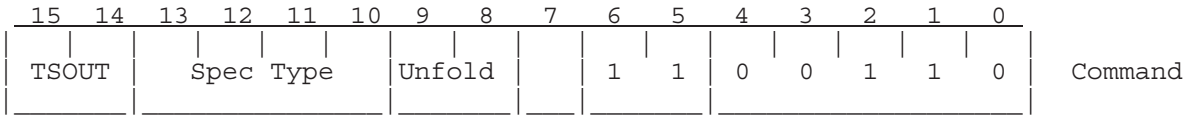
RHO Selects Signal Quality Index data. This dimensionless parameter uses the same 8-bit and 16-bit data formats as RHV (ρ_{HV}).

PHI Selects the cross channel differential phase. This parameter uses the same 8-bit and 16-bit angular data formats as PDP (φ_{DP}).

FLG Selects flag word output, bits defined as follows:
0 Reflectivity obscured at this bin
1 Velocity obscured at this bin
2 Width obscured at this bin
3 Point clutter detected at this bin

HCL Hydro-Class (Hydrometeor Classification) parameter as follows:
0 No measurement available
1 Non-meteorological target
2 Rain
3 Wet snow
4 Snow
5 Graupel
6 Hail

The Command word format for Time Series Mode is:



TSOUT Selects type of data to be output.

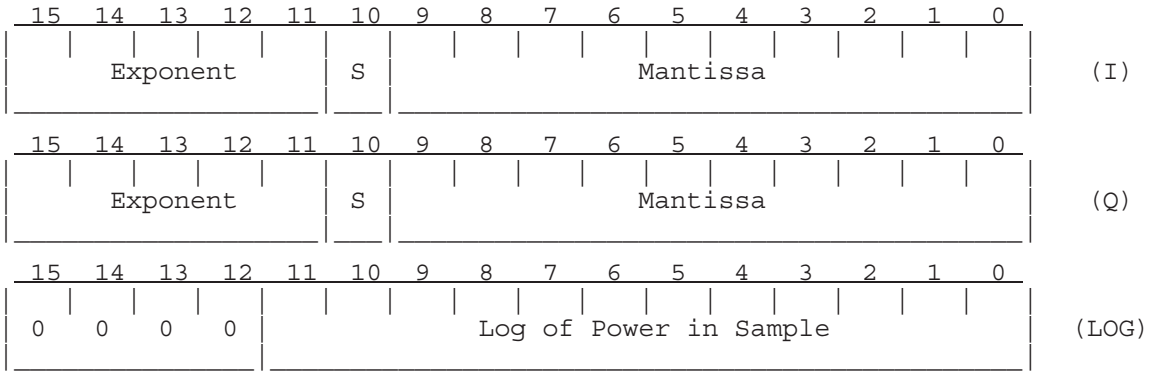
TSOUT Selects type of data to be output.
00 : 8-bit Time Series 01 : Power Spectrum
10 : 16-bit Time Series 11 : Unused

When the TSOUT bits select "Power Spectrum" then, depending on the current major mode, a further choice may be needed to select one of several spectral view points. For the Random Phase major mode the possible values of "Spec Type" are:

0: Raw First Trip
1: Whitened First Trip
2: Cleaned First Trip
3: Final First Trip
4: Raw Second Trip
5: Whitened Second Trip
6: Cleaned Second Trip
7: Final Second Trip

When time series output is selected the output data consist either of (3xBxN) or (2xBxN) words, depending on the output format, where B is the number of bins in the current range mask, and N is the number of pulses per ray. Data samples for each bin of pulse #1 are output first, followed by those for each bin of pulse #2, etc. up to pulse #N. In other words, the data are output in the same time-order that they were acquired.

In the floating point format, three words are used for each bin:



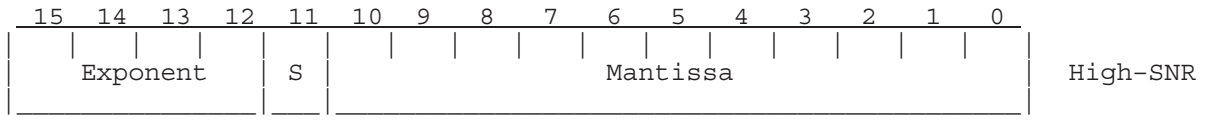
To convert these "legacy format" floating I and Q samples to voltages: First create a 12-bit signed integer in which bits zero through nine are

copied from the Mantissa field, and bits ten and eleven are either 01 or 10 depending on whether S is 0 or 1. Then, multiply this number by 2**(exponent-40), where the exponent field is interpreted as an unsigned 5-bit integer. Finally multiply by the maximum voltage. The resulting value has 12-bits of precision and a dynamic range of approximately 190dB. The large dynamic range is necessary to cover the full range of data. In summary:

$$Voltage = V_{MAX} \times (Sign, Mantissa) \times 2^{[Exponent - 40]}$$

Note that the resulting voltage span is actually $\pm 4 \times V_{MAX}$. The extra factor of four is built into the format so that transient excursions above the full scale input voltage can still be encoded properly. These may arise for time series data that have been processed by an IIR clutter filter.

An improved "High-SNR" packed floating format is also available that offers nearly the same dynamic range but provides a 6dB improvement in SNR, that is, a commensurate improvement in sub-clutter visibility of -78dB versus -72dB.



The High-SNR packed format is similar to the legacy packed format except that it uses one extra mantissa bit and one fewer exponent bit. The dynamic range lost in the exponent is recovered through a formatting trick known as "soft underflow", that is, the mantissa is allowed to become unnormalized when the exponent is zero.

To decode this format when the exponent is non-zero, first create a 13-bit signed integer in which bits zero through ten are copied from the Mantissa field, and bits eleven and twelve are either 01 or 10 depending on whether S is 0 or 1. Then, multiply this by 2**(exponent-25), where the exponent field is interpreted as an unsigned 4-bit integer.

To decode the High-SNR format when the exponent is zero simply interpret the mantissa as a 12-bit signed integer and multiply by 2**-24.

A complete analysis of the noise properties of the floating point codes would be fairly tricky. For the High-SNR format, the 12-bit mantissa with hidden normalization bit will vary from 2048 to 4095. The SNR will therefore vary from 66dB to 72dB and we can assign a mean value of 69dB. Another 9dB of useful range is contained within the code as follows:

- In a floating point encoding format, the notion of fixed additive quantization noise is not really correct. For a signal having a given power, the additive noise within each instantaneous sample will scale down according to the magnitude of that sample. The ensemble of noise terms thus contributes an RMS power that is smaller than the Peak–to–Noise ratio would imply. In the case of a sinusoidal input, this gives a 3dB boost in effective SNR.
- The format, of course, also represents negative amplitudes with the same relative precision as positive values. In a fixed–point format this would add 6dB (one more bit) to the overall dynamic range and large–signal SNR. In the floating format we really only gain 3dB (half a bit) because the RMS noises add independently on the positive and negative excursions.
- The packed format is used to encode timeseries (I,Q) pairs, and it's the SNR properties of these pairs that we're really concerned about. To a first approximation, having a pair of values roughly doubles the information content and adds another 3dB to the SNR.

The last of the three timeseries output words, the "Log of Power in Sample", is provided mainly for backwards compatibility. It can be calculated from the I and Q numbers. To convert to dBm it requires a slope and offset as follows:

$$dBm = P_{MAX} + Slope \times [Value - 3584]$$

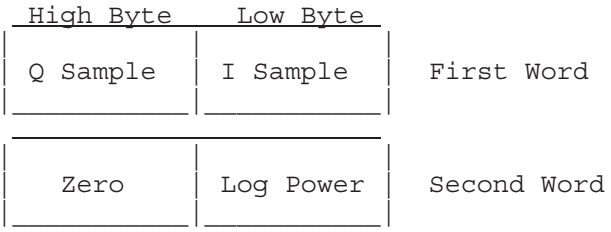
Where:

P_{MAX} = +4.5dBm for 12-bit IFD, +6.0dBm for 14-bit IFD

V_{MAX} = 0.5309 Volts for 12-bit IFD, 0.6310 Volts for 14-bit IFD

Slope = "Log Power Slope" word 3 of SOPRM command. 0.03 recommended.

For backwards compatibility the RVP8 produces a 8-bit fixed point time series format. Because of the limited dynamic range available, this will only show strong signals, and is not recommended for use. The I, Q, and Log power triplets are packed into two 16-bit output words as follows:



The "Log Power" value is the upper 8 bits of the long format. The other numbers are produced by the equation:

$$Voltage = V_{MAX} \times \left[\frac{Sample}{128} \right]$$

When Power Spectrum output is selected, the spectrum size is chosen as the largest power of two (N2) that is less than or equal to the current sample size (N). When the sample size is not a power of two, a smaller spectrum is computed that by averaging the spectra from the first N2 and the last N2 points. The data format is one word/bin/pulse, in the same order as for time series output. Each word gives the spectral power in hundredths of dB, with zero representing the level that would result from the strongest possible input signal (PMAX). Thus, the spectral output terms are almost always negative.

The time series that are output by the RVP8 are the filtered versions of the raw data, when available. If a non-zero time-domain clutter filter is selected at a bin, then the I and Q data for that bin show the effects of the filter. Whenever you need to observe the raw samples, make sure that no clutter filters are being applied.

In pulse pair time series mode with dual receivers, selecting (H+V) will produce data in one of two formats according to the "Sum H+V Time Series" question in the **Mp** setup section:

- Answering "Yes" will result in summed time series from both channels, but spectra from the DSP will be the averaged spectra from each channel individually. This allows the IRIS ascope utility to display either the spectrum-of-sum or sum-of-spectra according to whether the "Spectra from DSP" button is pressed in the Processing/ Gen-Setup window.
- Answering "No" will still produce the usual (BxN) time series output samples, except that the first half of these samples will be the first half of the "H" data in their normal order. This will be followed by a zero sample if (BxN) is odd; followed by the first half of the "V" data, also in their normal order.
- In other words, only the first halves of the individual "H" and "V" sample arrays are output by the RVP8. As an example, if you select

25 bins and 100 pulses, then the output data will consist of 1250 "H" samples (from all bins in the first 50 pulses), followed by 1250 "V" samples from the exact same set of bins and pulses. This is the more useful option when custom algorithms are being run on the data from the two separate receivers.

When the number of output words is large there is a possibility that the internal buffering within the RVP8 may overflow and data may be lost. Due to internal memory limitations, the product (BxN) must be less than 12000. A bit in the latched status word (See GPARM) indicates when time series overflows occur. In such cases, the correct number of words are still output, but they are all zero after the point at which overflow was detected.

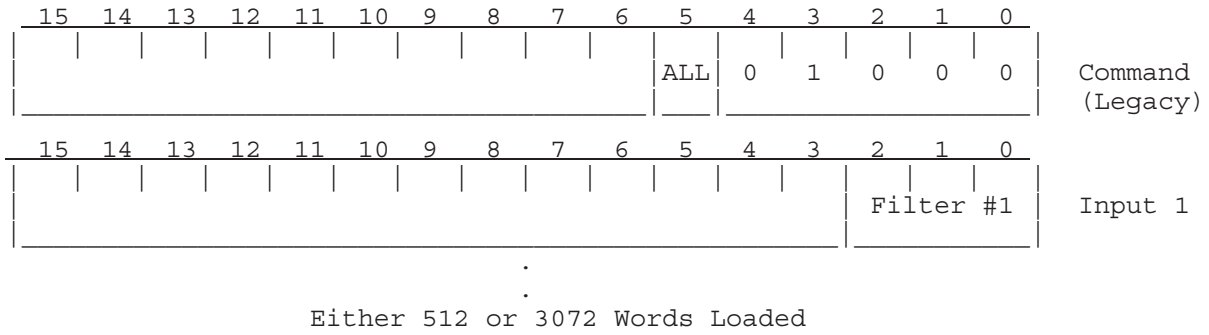
7.8 Load Clutter Filter Flags (LFILT)

A special feature of the RVP8 processor is that any of the available clutter filters may be chosen independently at each selected range. This range-dependent clutter removal is useful when the clutter characteristics vary with increasing range. Typically, clutter interference is most severe in the immediate vicinity of the radar. Thus, a highly rejective filter might be chosen for near ranges, and a less rejective or perhaps no filter could be used at far ranges.

Legacy Version

In the legacy version of LFILT, the input words following the command specify the choice of filter to be applied at each of the selected range bins. A fixed size filter table is always loaded, regardless of whether the range mask (See LRMSK) is using the full number of bins. In such cases, the later filter codes are simply ignored for the current range mask. However, if a longer range mask is loaded in the future, then those later codes would apply to the correspondingly numbered bins. Put another way, each filter code is associated with a particular bin number, not with a particular range. The correspondence between bin numbers and actual ranges is made only through the range mask.

Only the low three bits are used in each word to specify the filter number. If the ALL bit is set in the Command, then 3072 words are loaded, corresponding to the maximum number of range bins that are allowed. Otherwise only 512 words are loaded, and the 512th filter choice is replicated for all bins further in range.



Enhanced Version

The RVP8 supports an enhanced version of the LFILT command that provides "Clutter Maps", that is, much greater flexibility by allowing filter choices to depend on antenna angle as well as range. This lets you specify a 2D or even 3D table of clutter filter selections that are dynamically selected during live data processing.

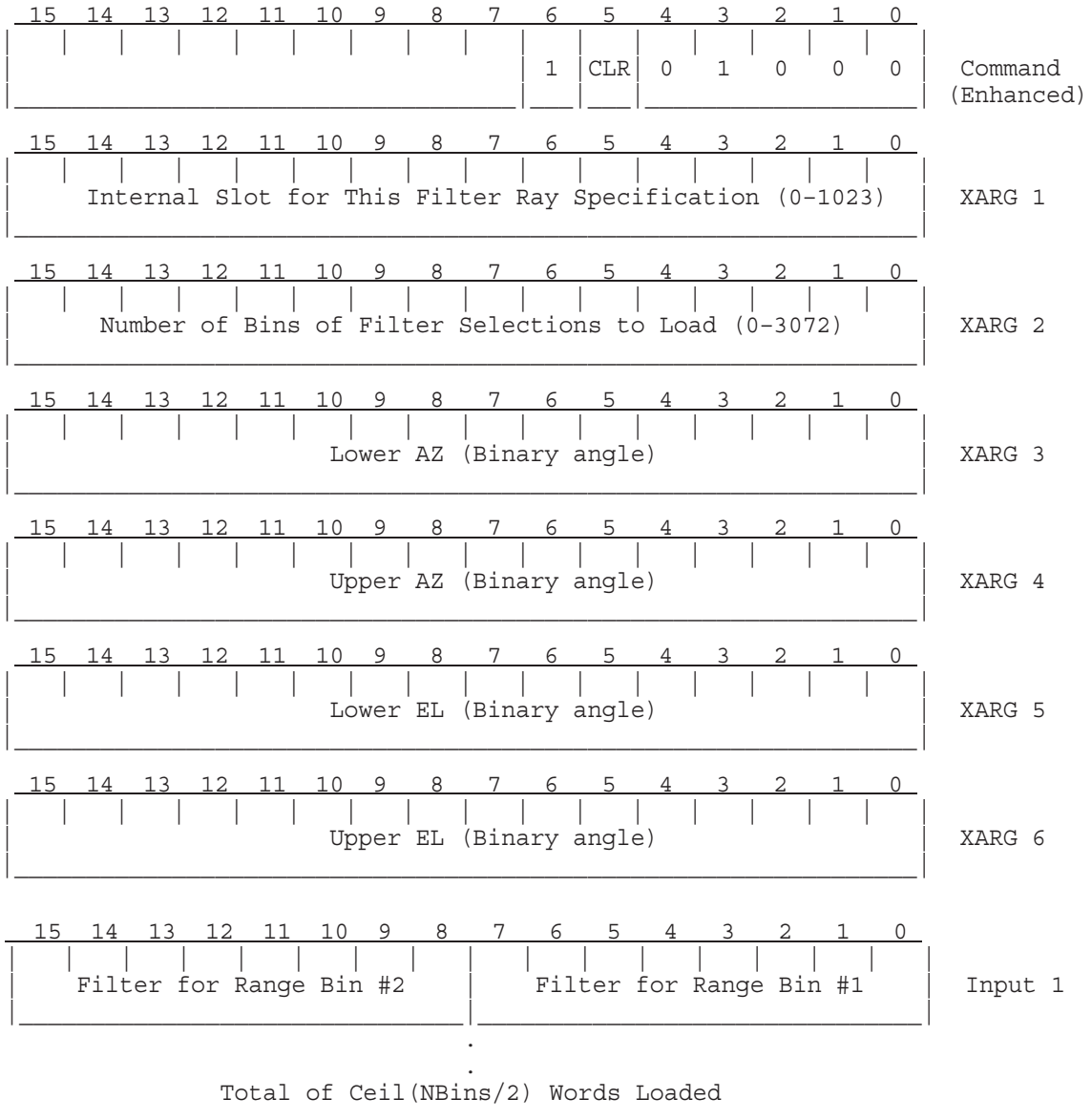
The RVP8 maintains an internal array of up to 1024 different filter-versus-range tables, each of which is keyed to a particular solid angle AZ/EL sector. Each enhanced LFILT command fills in one of these slots with a filter selection table similar to that of the legacy command, except that the number of range bins is specified explicitly and eight bits are used for filter selection rather than three. Then, for each live ray being processed, the RVP8 applies clutter filters according to the filter slot whose solid sector includes the midpoint AZ/EL of the ray. If the antenna angle of the ray does not fall within any of the defined filter sectors, then the all-pass filter (#0) will be used at all ranges.

The low and high angle limits in each filter slot are inclusive; hence, the pair (0x000, 0xFFFF) would span the full 360-degree circle with no gaps. Also, the filter array can be sparse (not all slots filled in), and have overlapping sectors (in which case the highest numbered slot that spans a ray's AZ/EL midpoint will be used). Choosing the highest numbered encompassing slot is a subtle but important detail that allows complex regions to be defined as a layered hierarchy of overlapping sectors. For example Slot-0 might define a default 360-degree filter-versus-range table, while Slot-1 defines special filtering within 0-90 degrees that is further modified by Slot-2 filters in a 40-50 degree span. A 45-degree ray would then be filtered according to Slot-2, a 60-degree ray would use Slot-1, and a 100-degree ray would use Slot-0.

If the CLR bit is set in the opcode, then no additional arguments follow and the entire internal filter array (all slots) will be invalidated. The result is that no clutter filter will be applied to any of the processed data, regardless of Range, AZ, or EL. Moreover, loading a given slot with a table consisting

of zero bins of filter data will invalidate just that one slot. This allows some data to be removed from the table without having to resort to a complete CLR operation.

The legacy LFILT command is equivalent to calling the enhanced command first with the CLR bit set, followed by a second call that writes the legacy filter choices into slot #0 using AZ/EL limits that cover all of space. Thus, the legacy behavior is obtained as a special case of the enhanced mechanism.



7.9 Get Processor Parameters (GPARM)

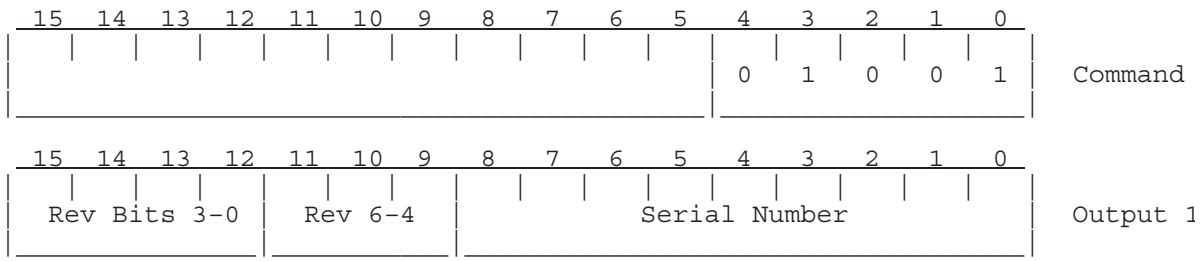
This command is used to access status information from the RVP8 processor. Sixty-four words are always transferred, some later words are reserved for future compatibility and are read as zeros. For convenience, a shorthand table of the output words is given in Table 20 on page 309.

Table 20 RVP8 Status Output Words

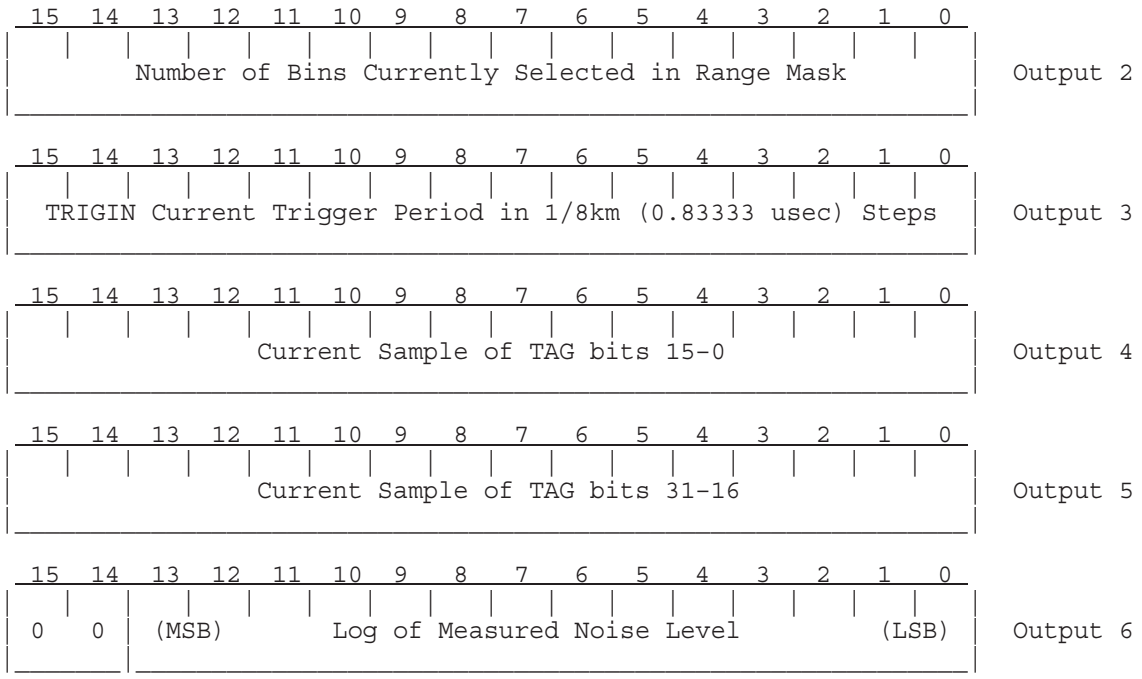
Word	Description
1	Revision / Serial number
2	Number of Range Bins
3	Current trigger period
4	Current TAG00 TAG15
5	Current TAG16 TAG31
6	Log of Measured Noise Level
7	"I" Channel DC Offset
8	"Q" Channel DC Offset
9	Latched Processor Status
10	Immediate Status Word #1
11	Diagnostic Register A
12	Diagnostic Register B
13	Number of Pulses / Ray
14	Trigger count (Low 16-bits)
15	Trigger Count (High 8-bits)
16	No. of Properly Acquired Bins
17	No. of Properly Processed Bins
18	Immediate Status Word #2
19	Noise Range in Km
20	Noise Trigger Period
21	Pulse Width 0 min. Trig. Period
22	Pulse Width 1 min. Trig. Period
23	Pulse Width 2 min. Trig. Period
24	Pulse Width 3 min. Trig. Period
25	Pulse Width Bit Patterns
26	Current /Pulse Width
27	Current Trigger Gen. Period
28	Desired Trigger Gen. Period
29	PRT at Start of Last Ray
30	PRT at End of Last Ray
31	Processing/Threshold Flags
32	Log Slope
33	LOG Threshold
34	CCOR Threshold
35	SQI threshold
36	SIG Threshold for Width
37	Calibration Reflectivity

Table 20 RVP8 Status Output Words (Continued)

Word	Description
38	Reserved
39	Reserved
40	Range Averaging Choice
41	Reserved
42	Reserved
43	Header configuration of PROC data
44	I-Squared Noise (Low 16-bits)
45	I-Squared Noise (High 16-bits)
46	Q-Squared Noise (Low 16-bits)
47	Q-Squared Noise (High 16-bits)
48	Log of Measured Noise Level
49	LOG Noise Standard Deviation
50	Horizontal/Vertical Noise Ratio
51	AFC/MFC Control Value
52	Interference Filter Select
53	Interference Filter C1 Constant
54	Interference Filter C2 Constant
55	Immediate Status Word #3
56	Burst Tracking Slew
57	Polarization Algorithm Choices
58	Range Mask Spacing
59	Immediate Status Word #4
60	Reserved
61	Reserved
62	Reserved
63	Reserved
64	Reserved

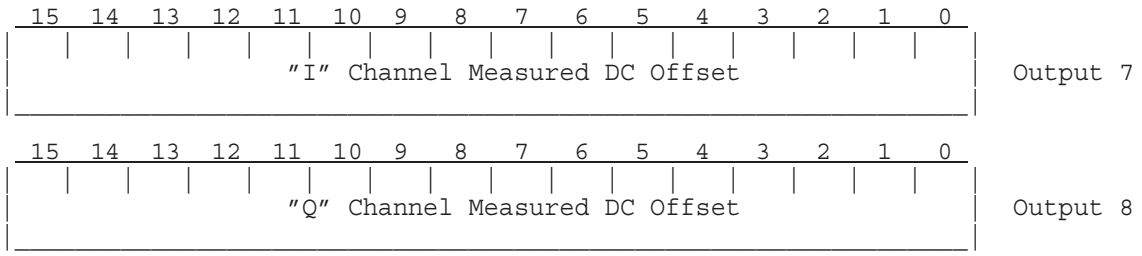


The revision and serial numbers of the particular RVP8 board are accessible here. This information is useful when computer software is being designed to handle a variety of signal processor revisions. The revision number is seven bits total; four of which are still in the high four bits of the word for compatibility with an older format.

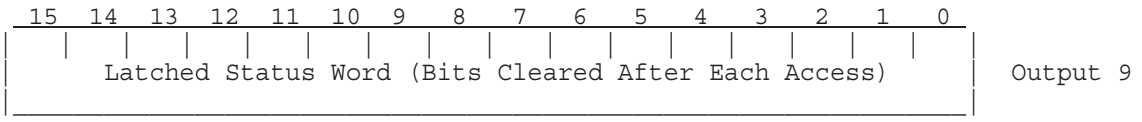


This value is scaled 4 times higher than the time series LOG format (see the discussion in 7.7 Initiate Processing (PROC) on page 295). To convert to dBm, use the equation:

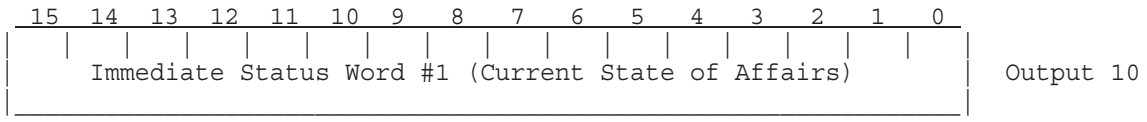
$$dBm = P_{MAX} + Slope \times [(Value/4) - 3584]$$



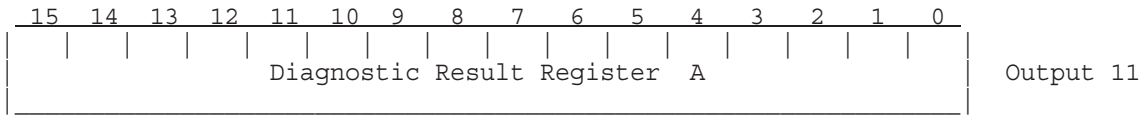
These two words convey the measured "I" and "Q" DC offsets from the last noise sample. The output format is either signed 16-bit values in which ±32767 represent ±1.0 (legacy format), or packed timeseries values using the High-SNR encoding format. Bit-9 of GPARM Word-59 tells which format to use.



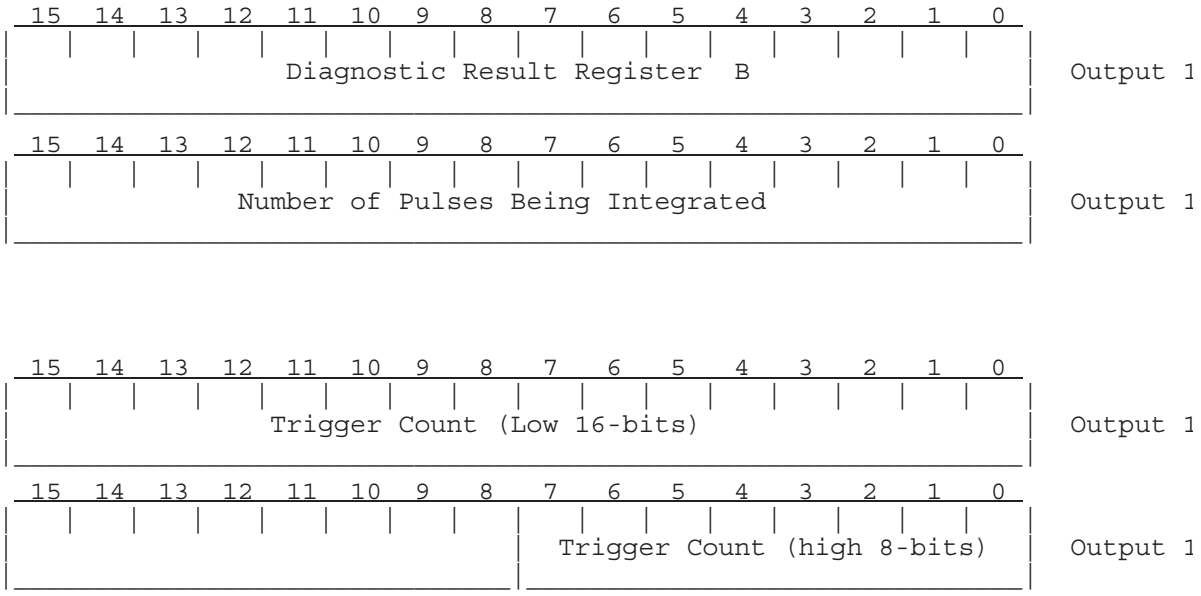
Bit 0	No Trigger during noise measurement.
Bit 1	Trigger too fast during noise measurement, that is, some of the noise sample bins were positioned past the trigger range.
Bit 2	No trigger during PROC command.
Bit 3	PRT varied by more than 10 microseconds within portions of a processing interval that should have been at a fixed rate.
Bit 4	Error in polarization control and/or polarization status readback
Bit 5	FIFO overflow during last PROC command.
Bit 6	Command received while waiting for output FIFO space. The command was processed but some output data has been lost (zeroed).
Bit 7	CError detected during last SNOISE command.
Bit 9	Error in last Load Range Mask (LRMSK) Command. This generally means that too many range bins were selected.
Bit 10	Error in LSIMUL command protocol.
Bit 11	Measured phase sequence is incorrect.
Bit 15	Invalid processor configuration. This bit is set if the last PROC command called for an illegal combination of parameters. The possible causes are: - Spectrum size greater than 128 or less than 4 - More than 342 bins/slave in FFT modes - (bins/slave) x (4 + sample size) exceeds 26200 in FFT modes - (bins/slave) x (sample size) exceeds 3000 for Time Series or Spectra output - Odd number of bins selected during fast polarization switching - Bad combination of polarization parameters



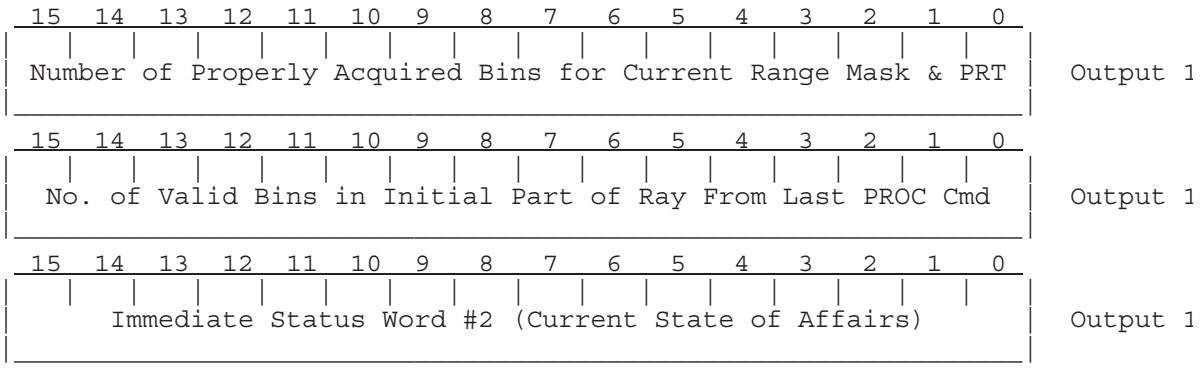
Bit 0	No trigger, or, more than 50ms since last trigger.
Bit 1	Error in loading trigger angle table (See LSYNC Command)
Bit 2	PWINFO command is disabled.
Bit 3	Angle sync input is BCD (Else binary angle)
Bit 4	Angle sync is on elevation axis (Else azimuth axis).
Bit 5	Angle sync is enabled.
Bit 6	Angle sync allows short output rays.
Bit 7	Angle sync is dynamic (else rays begin on sync angles).
Bit 8	DSP has full IAGC hardware and firmware configuration.
Bit 9	DSP supports 16-bit floating time series.
Bit 11, 10	Current unfolding mode.
Bit 13, 12	Number of RVP8/PROC compute processes minus one.
Bit 14	DSP supports Power Spectrum output.



Bit 0	RVP8/Rx card #1 failure
Bit 1	RVP8/Rx card #2 failure
Bit 2	RVP8/Tx card #1 failure
Bit 3	RVP8/Tx card #2 failure
Bit 4	IO62 card #1 failure
Bit 5	IO62 card #2 failure
Bit 6	Error loading config/setup files
Bit 7	Error attaching to antenna library
Bit 8	Problem when forking compute processes
Bit 9	Error(s) in softplane configuration
Bit 10	Signals raised during startup
Bit 11	RVP8 running without root privileges
Bit 12	Problem creating daemon process
Bit 13	Inconsistent setup values detected



The trigger count is a running tally of the number of triggers received by the RVP8 on the TRIGIN line. It is a full 24-bit counter.



Bit 0 Processor supports FFT algorithms

Bit 1 Processor supports Random Phase algorithms

Bit 2 Reserved (zero)

Bit 3 Processor supports DPRT-1 (Dual-PRT) algorithms

On dual IFD systems: Bits 4,5,7, and 11 are set if either IFD fails:

Bit 4 Problem in UpLink cable from RVP8/Rx > IFD

Bit 5 Problem in DownLink cable from IFD > RVP8/Rx

Bit 7 IFD PLL is not locked to external user-supplied clock reference

Bits 8-10 Status of burst pulse and AFC feedback
1: AFC Disabled
2: Manual Frequency Control
3: No burst pulse detected
4: AFC is waiting for warm-up
5: AFC is locked
6: AFC is tracking

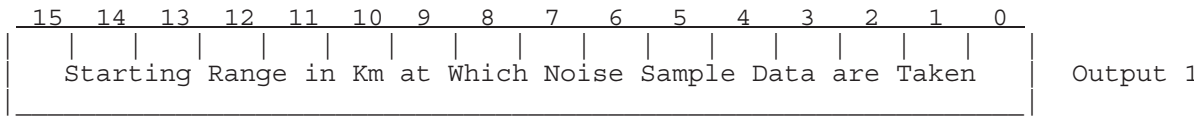
Bit 11 IFD test switches are not in their normal operating position

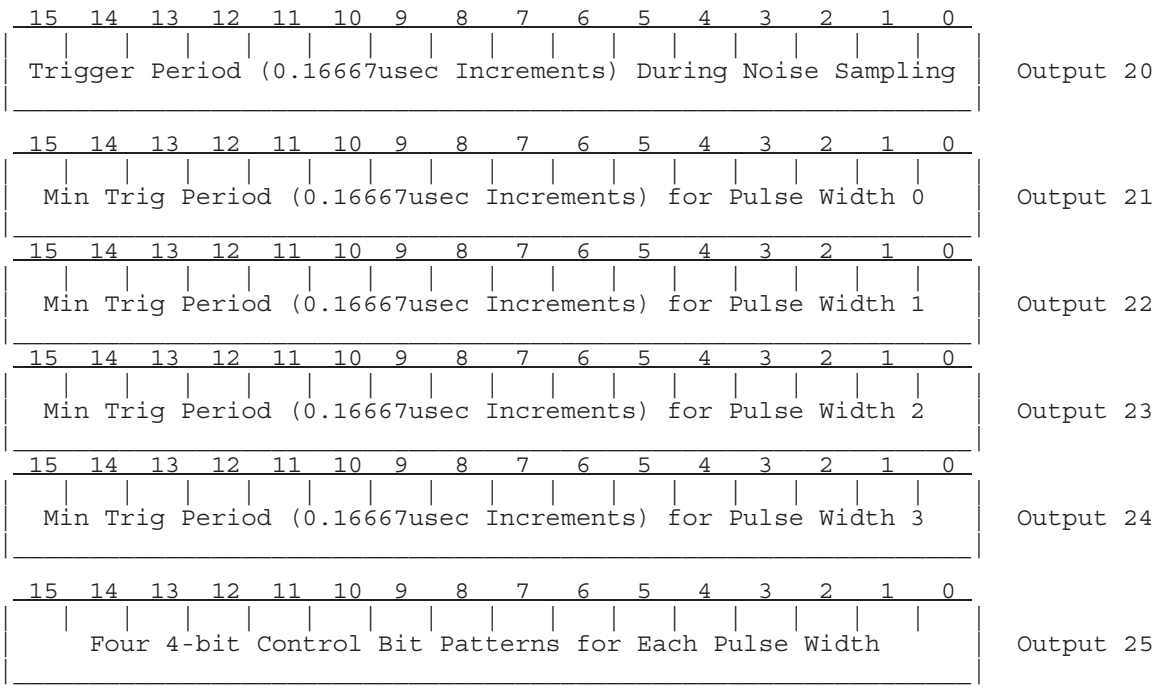
Bit 12 Set according to whether the RVP8 is performing trigger blanking. This allows the host computer to decide whether to interpret the End-TAG-0 bit in the output ray header as a blanking flag, or as a normal TAG line.

Bit 13 Missing signal at IFD #1 Burst Input

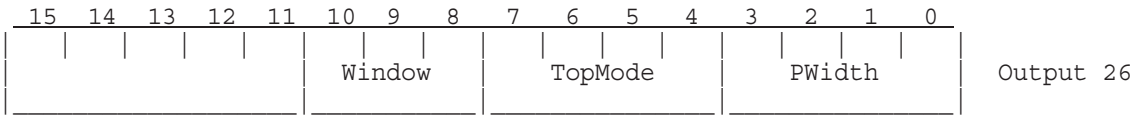
Bit 14 Reserved (zero)

Bit 15 Set when valid burst power is detected but the center-of-mass lies outside of the aperture sub-window that defines the portion of the pulse used for AFC analysis. This error bit effectively flags when the burst pulse has drifted out of its optimal placement within the sampling window.

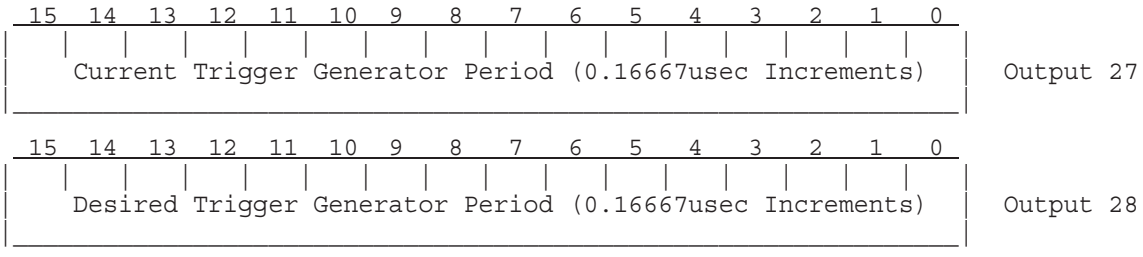




See PWINFO command, input word #1, for definition of these bits.

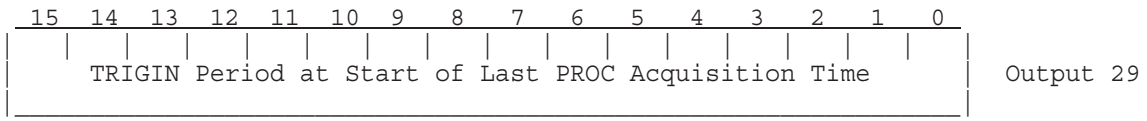


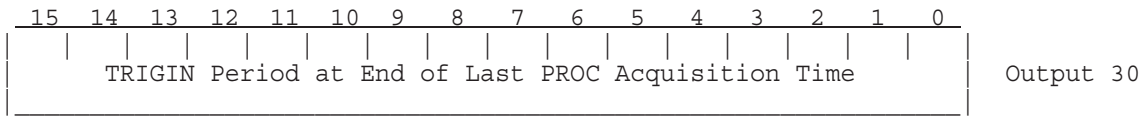
PWidth	Currently selected radar pulse width
TopMode	Major Mode (See SOPRM Input #9)
Window	Spectral Window Choice (See SOPRM Input #10)



The desired trigger generator rate is that which was selected in the most recently issued SETPWF command (or power-up rate if SETPWF was never issued). The current rate may be different from the desired rate due to bounding against limits for the current pulse width, or being in an odd

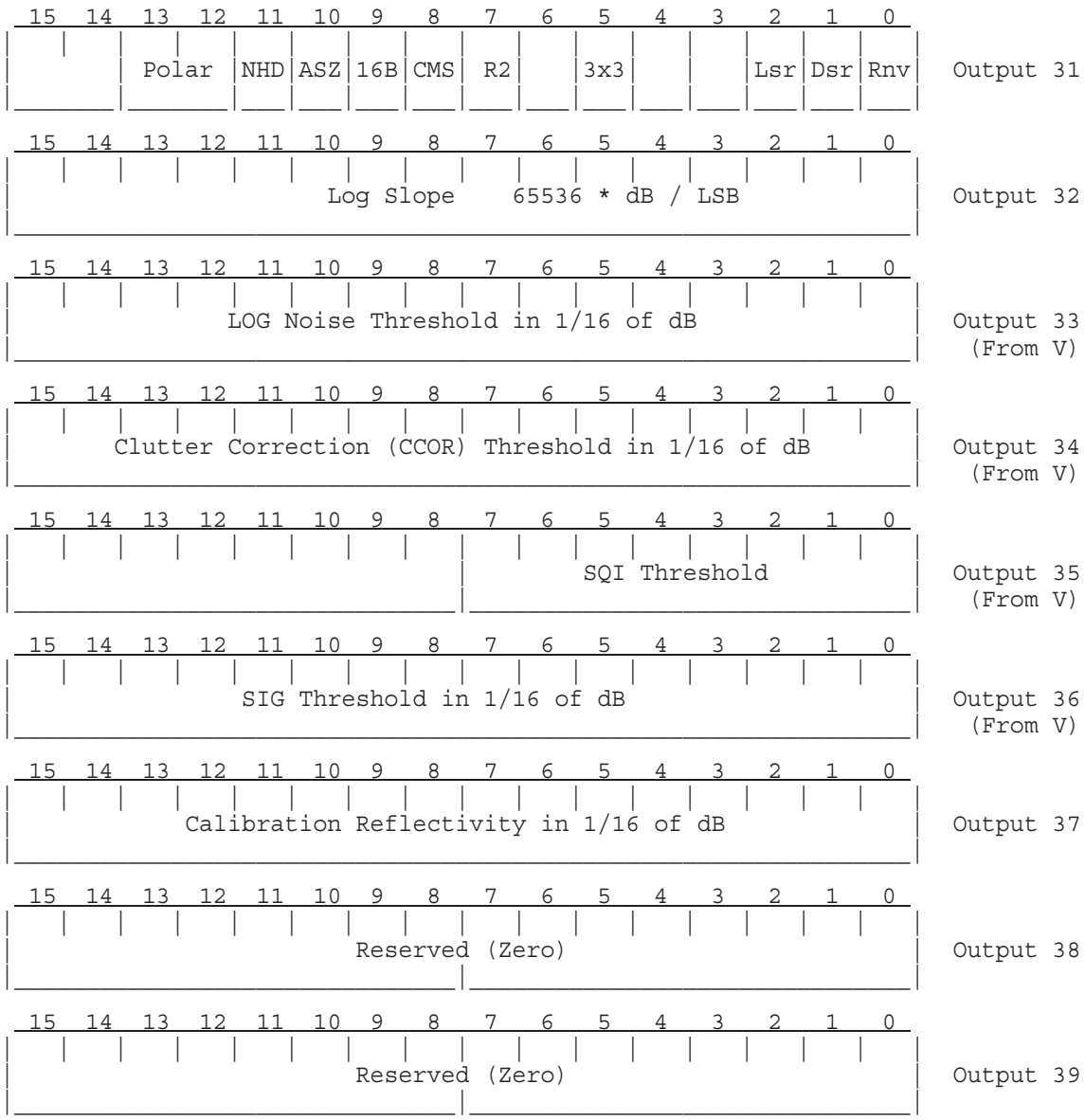
ray cycle during dual-PRT processing. The measured PRT's are forced to 0xFFFF (the maximum unsigned value) whenever the external trigger is expected but missing.

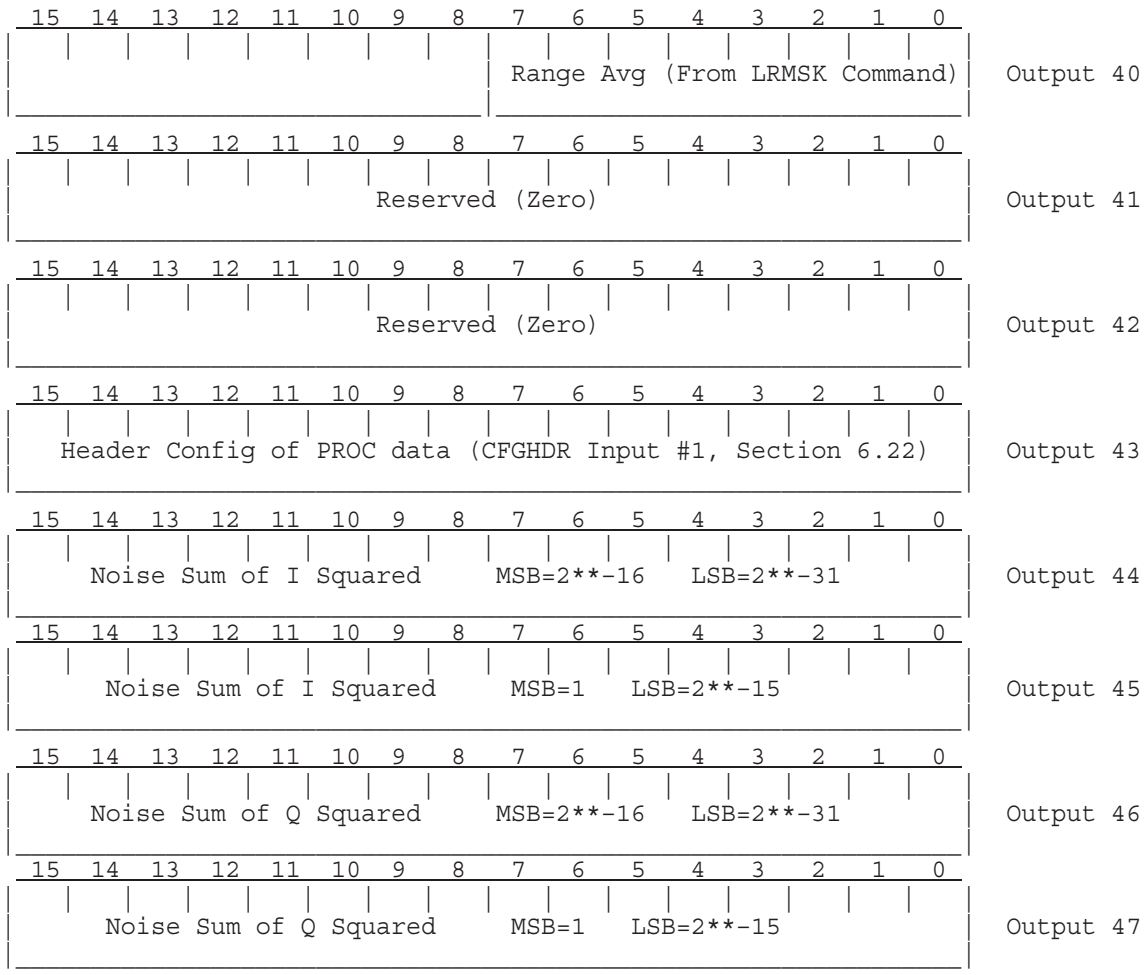




The PRTs from the start and end of the last ray are the actual measured values whenever possible, that is, when non-simulated data are being processed, and we either have an external trigger, or an internal trigger that is not in any of the Dual-PRT modes. The units are the same as for the measured current trigger period in Output #3.

Outputs 31 through 37 are the current processing and threshold parameters set by SOPRM. See 7.3 Setup Operating Parameters (SOPRM) on page 279 for additional notes on each of these parameters. Since the threshold levels for each data parameter can be different (See THRESH command, 7.29 Set Individual Thresholds (THRESH) on page 348), words 33-36 are taken from the velocity parameter.


15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
Polar
NHD
ASZ
16B
CMS
R2
3x3
Lsr
Dsr
Rnv
Output 31
Log Slope 65536 * dB / LSB
Output 32
LOG Noise Threshold in 1/16 of dB
Output 33 (From V)
Clutter Correction (CCOR) Threshold in 1/16 of dB
Output 34 (From V)
SQI Threshold
Output 35 (From V)
SIG Threshold in 1/16 of dB
Output 36 (From V)
Calibration Reflectivity in 1/16 of dB
Output 37
Reserved (Zero)
Output 38
Reserved (Zero)
Output 39



To compute the noise power in dBm from Words 44-47, first calculate:

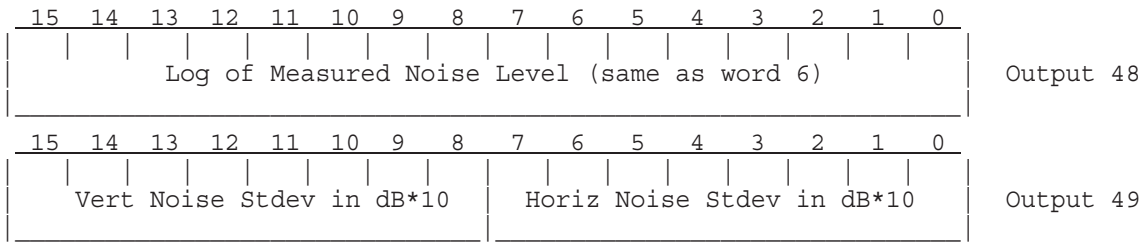
$$N_I = (Word\ 45) \times 2^{-15} + (Word\ 44) \times 2^{-31}$$

$$N_Q = (Word\ 47) \times 2^{-15} + (Word\ 46) \times 2^{-31}$$

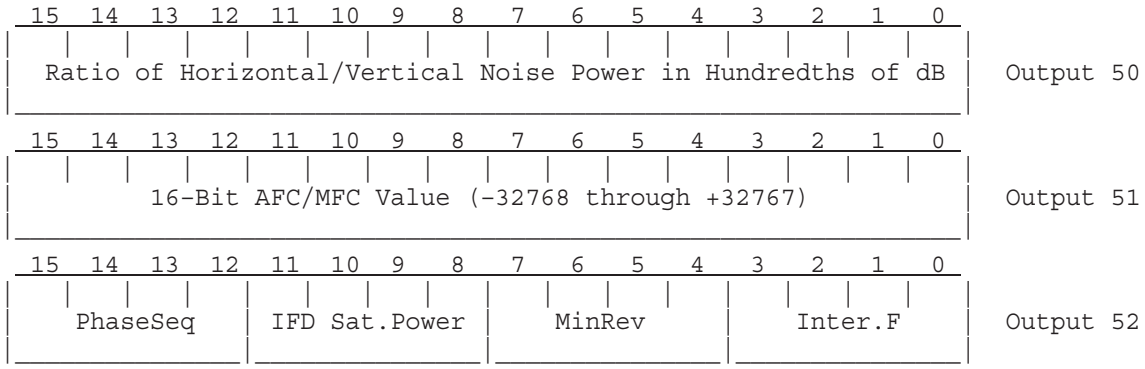
From which we obtain:

$$dBm = P_{MAX} + 10\log_{10}(N_I + N_Q) - 3dB$$

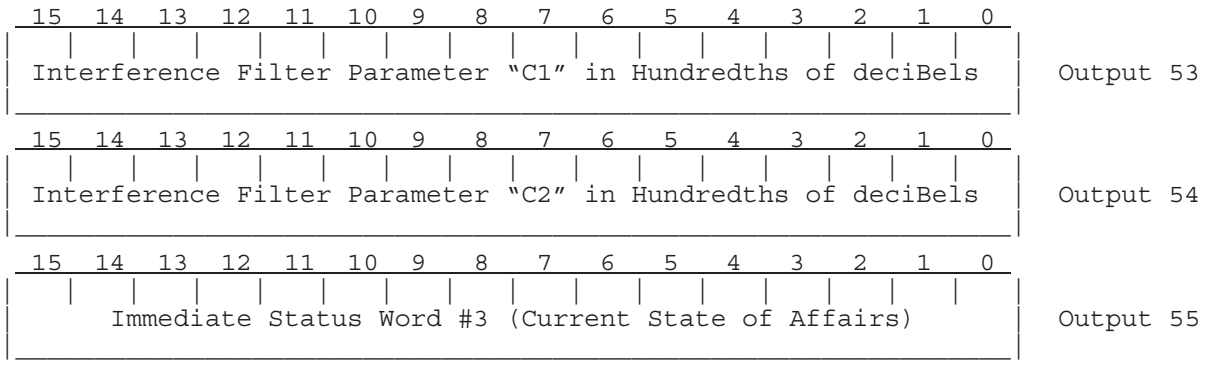
Note that the four integer values become rather small and severely quantized when the noise power drops to low values. Historically, these four words were used to balance the individual gain of the "I" and "Q" channels in the RVP6 in the presence of a strong test signal. Since "I" and "Q" are inherently balanced in the RVP8, these output words are no longer of much value.



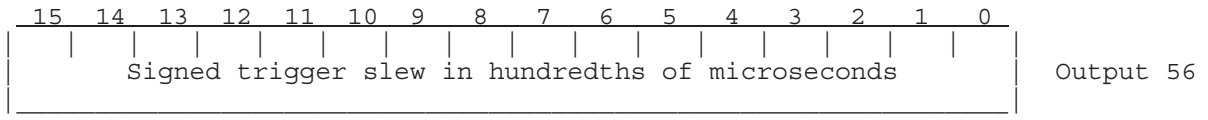
The noise standard deviations for each receive channel are normalized to the mean power. The values reported here will therefore hover around 0dB for ordinary exponentially distributed noise in which the standard deviation scales directly with the mean.



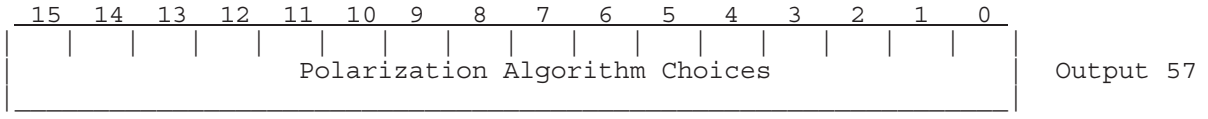
Inter.F	Specifies which interference filter is running. Zero means "none"; see 6.1.5 Interference Filter on page 212 for a description of the interference filter algorithms.
MinRev	Minor revision level of the RVP8 code that is currently running.
IFD Sat.Power (*PMAX*)	Input power required to saturate the IF-Input A/D converter for the RVP8/IFD receiver that is currently attached. 0: +4.5dBm 1: +6.0dBm
PhaseSeq	Tx Phase modulation sequence (See 7.27 Configure Phase Modulation (CFGPHZ) on page 346)



Bit 0	Burst pulse timing adjustments can be made
Bit 1	Burst pulse frequency adjustments can be made
Bit 2	Burst pulse hunting is enabled
Bit 3	Burst pulse hunt is running right now
Bit 4	Last burst pulse hunt was unsuccessful
Bit 5	Processor supports DPRT-2 (Dual-PRT) algorithms
Bit 6	Could not generate the requested phase sequence
Bit 7	Problem with digital transmitter clock
Bits 8-11	User-defined Major Modes 1-4 are supported



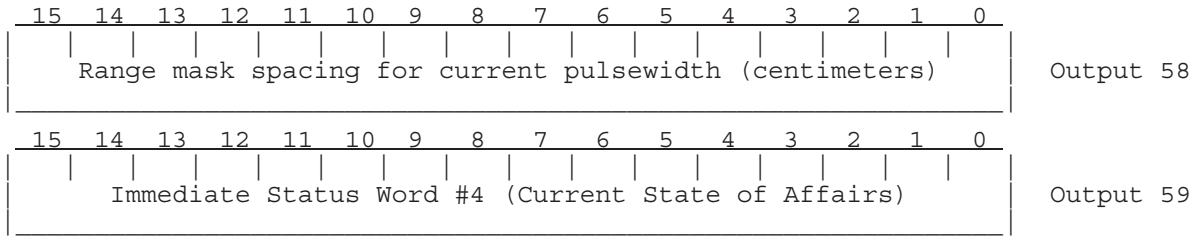
This is the same format that is used by the SETSLEW command to set the current trigger slew (See 7.25 Set Trigger Timing Slew (SETSLEW) on page 345)



Bit 0	Use H transmissions for (T,Z,V,W)
Bit 1	Use V transmissions for (T,Z,V,W)
Bit 2	Use Co-Pol reception for (T,Z,V,W)
Bit 3	Use Cross-Pol reception for (T,Z,V,W)
Bit 4	Correct all polar Parameters for noise
Bit 5	Use filtered data for all polar Parameters

Bit 6 Sign convention for PHIdp

Bit 7 Z and ZDR are corrected for attenuation using PhiDP



Bit 0 Internal power spectra size matches sample size (else power-of-2)

Bit 1 PROC command output spectra match sample size (else power-of-2)

Bit 2 Trigger pattern has been altered to fit within the desired PRT

Bit 3 PRT has been altered to preserve the desired trigger pattern

Bit 4 Using HighSNR packed (I,Q) format

Bit 5 Trigger sequence truncated due to insufficient pattern memory

Bit 6 TimeSeries data source is external to the RVP8

Bit 7 WSR88D "Batch" mode is supported

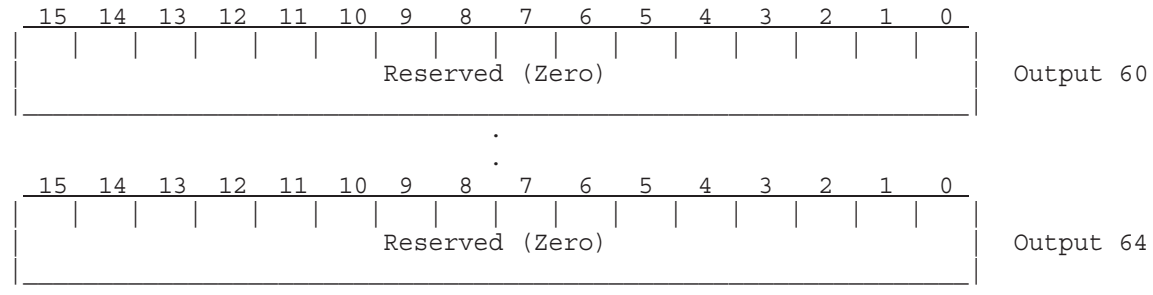
Bit 8 Major mode refuses to use external trigger

Bit 9 GPARM outputs #7 and #8 use Hi-SNR format, else linear

Bit 10 Receiver protection fault

Bit 11 IFD dual-channel inconsistency (for example, power and/or phase out of bounds for ratio of HiGain-to-LoGain channels)

Bit 12 GPS 1-pulse-per-second input clock error

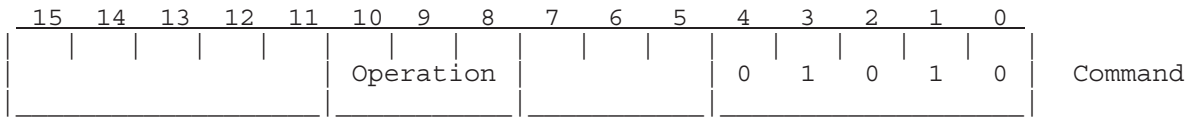


7.10 Load Simulated Time Series Data (LSIMUL)

This command is provided as a diagnostic for proper functioning of the RVP8 algorithms. It permits arbitrary simulated data samples to be input to the processing routines, rather than sampled data from the A/D converters as is ordinarily the case. Since the properties of the simulated data are known exactly, it is possible to verify that the calculations within the RVP8 are proceeding correctly.

The LSIMUL command (with operation=1) should be issued prior to the PROC command which is being tested. This enables the simulated data mode. The next PROC command will then wait for *N* (*N* = sample size) LSIMUL commands (with operation=2) prior to outputting each ray. The arrival of any other command during that time will cause the simulated data mode to be exited, and error bit #10 will be set in the GPARM latched status word. The error bit is also set if an LSIMUL command with operation=2 is received while simulated data mode is disabled.

You may specify a single simulated data sample for every range bin, or a pattern or simulated samples to be replicated over the range of bins. Most RVP8 algorithms are independent of range, and can be tested with identical data at every bin. Notable exceptions, however, are the "pop" clutter filter, and range bin averaging procedures. In its full generality, the LSIMUL command permits independent I and Q samples to be simulated at every bin of every pulse. If this results in more host computer I/O than is practical, then specify fewer simulated bins and allow the RVP8 to replicate them internally.

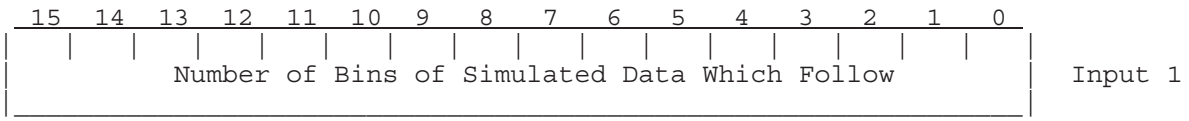


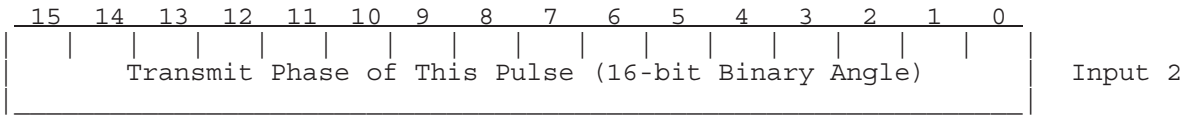
The available operations are:

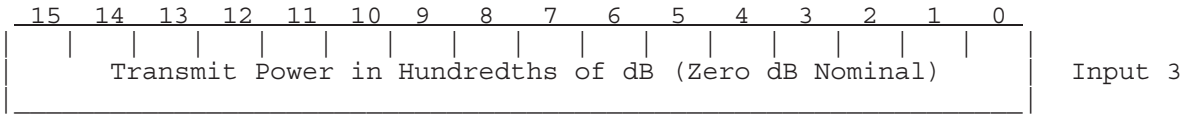
0 Disable the use of simulated data. RVP8 returns to acquisition and processing of live data from the A/D converters.

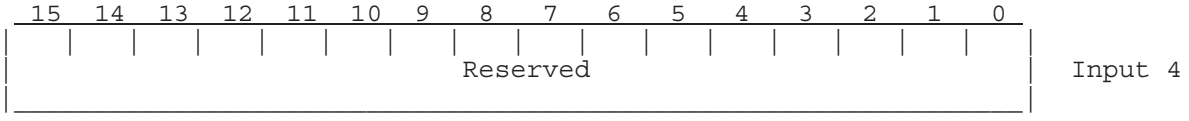
1 — Enable processing of simulated data. Subsequent PROC commands will use the data supplied in the next *N* (*N* = sample size) LSIMUL commands with Operation= 2. The receiver noise and offset levels which are internally maintained by the RVP8 are set to their special simulated values (from the **M**+ setup menu) by this command. This is because the measured offsets are not relevant to the simulated data, and must not be used in the subsequent computations. Thus, it is important to issue the SNOISE command before resuming the acquisition and processing of live radar data.

2 or 3 — Load one pulse of data samples beginning with the following 4-word header, and continuing with an array of items each representing a single instantaneous sample of (I,Q) data. You may specify one or more bins to be loaded, and the RVP8 will replicate these data as necessary in order to fill out the entire count of acquired bins. If the number of bins is zero, then a zero-valued sample is applied for all channels.

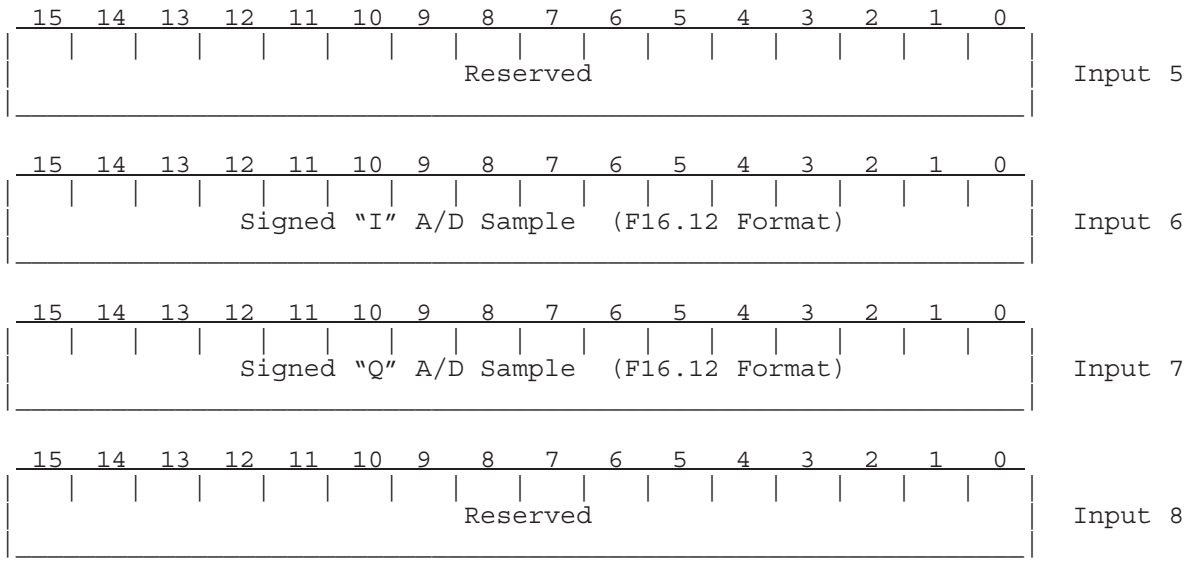




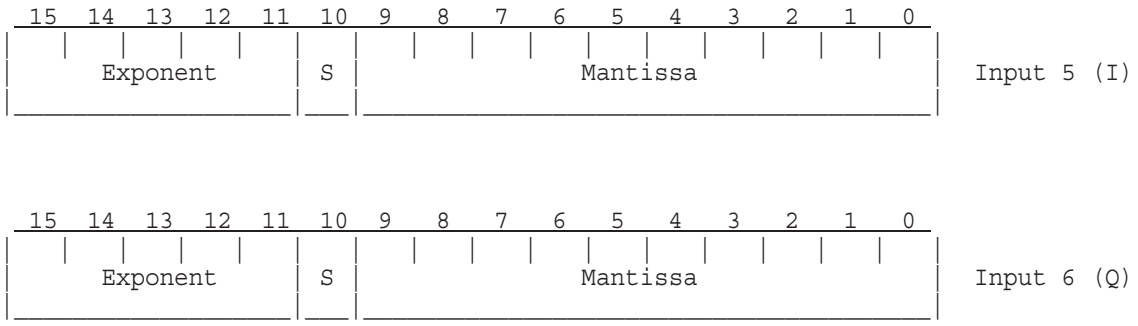




In the legacy Format #2 (RVP5-RVP8) each bin within the pulse is represented by four 16-bit fixed point words. Thus, the total number of words loaded is *(4+4B)*, where *B* is the bin count specified in Word #1. This takes account of the four "header" words, plus four words for every bin being defined.

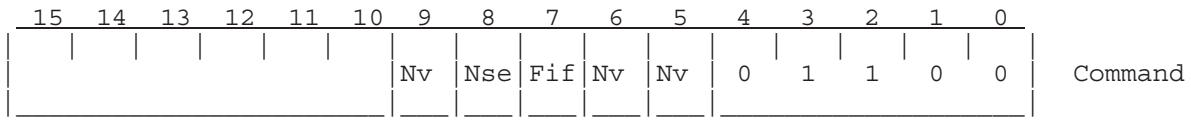


In Format #3 (new in the RVP8) each bin within the pulse is represented by two 16-bit floating point words having the exact same format as the packed (I,Q) time-series data that are output by the PROC command (See 7.7 Initiate Processing (PROC) on page 295). Compared to the legacy 4-word format, this 2-word format uses half the I/O bandwidth, has su-perior dynamic range, and allows data to be fed back into the signal processor in their native packed format. The total number of words loaded (including the initial header section) is (*4+2B*), where *B* is the bin count specified in Word #1.



7.11 Reset (RESET)

The RESET command permits resetting either the entire RVP8 processor, or selected portions thereof. Flags within the command word determine the action to be taken.



Nv	Reloads configuration from the saved nonvolatile settings. For compatibility with RPV6 and RVP7, any of 3 bits will trigger this response.
Nse	Reset the receiver noise levels to the power-up default value for all pulsewidths as defined in the **Mt** setup questions (See 4.2.5 Mt<n>— Triggers for Pulsewidth #n on page 133).
Fif	Remove any data currently in the output FIFO's. This permits flushing output data that was left from a previous command, so that new output can be read from scratch. See notes in the *Introduction* to this chapter concerning actions taken by the RVP8 when the output FIFO becomes full.

7.12 Define Trigger Generator Waveforms (TRIGWF)

NOTE This opcode is obsolete, and is included only for backward compatibility with the RVP6. The opcode is disabled by default (See 4.2.1 Mc — Top Level Configuration on page 120), because the interactive trigger setup procedure described in 5.3 Pb — Plot Burst Pulse Timing on page 163 is the preferred method of defining all RVP8 triggers and timing. TRIGWF should not be used in any new code applications that drive the RVP8.

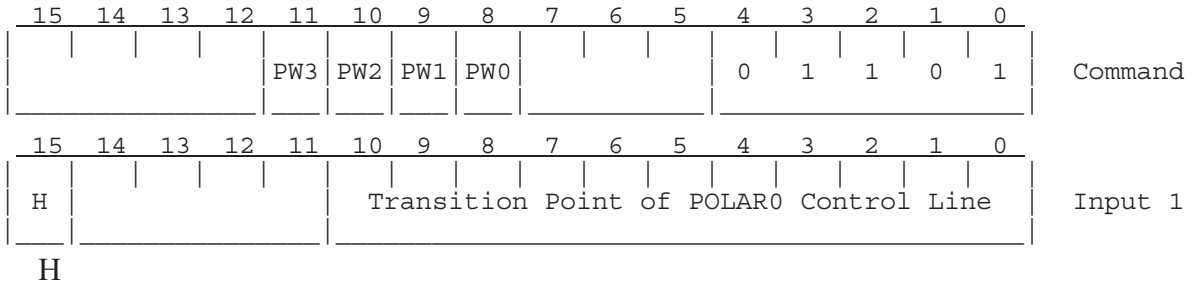
The RVP8 has a built-in trigger generator that can synthesize six independent digital output waveforms, each having arbitrary shape and being active anywhere in a window centered around zero-range. The six trigger outputs can be defined by a 2048-word by 6-bit table which is loaded from the user computer. The patterns are automatically read from the table and output to the six trigger lines during each radar pulse. The six outputs can be used for transmitter triggers, scope triggers, range strobes,

PLL gates, etc. The writable waveform table makes the RVP8 unique, in that the detailed timing of trigger and related control signals can be easily adjusted in software, without having to resort to reprogramming PROMs. This makes it possible for user software to edit the trigger timing in a convenient interactive manner.

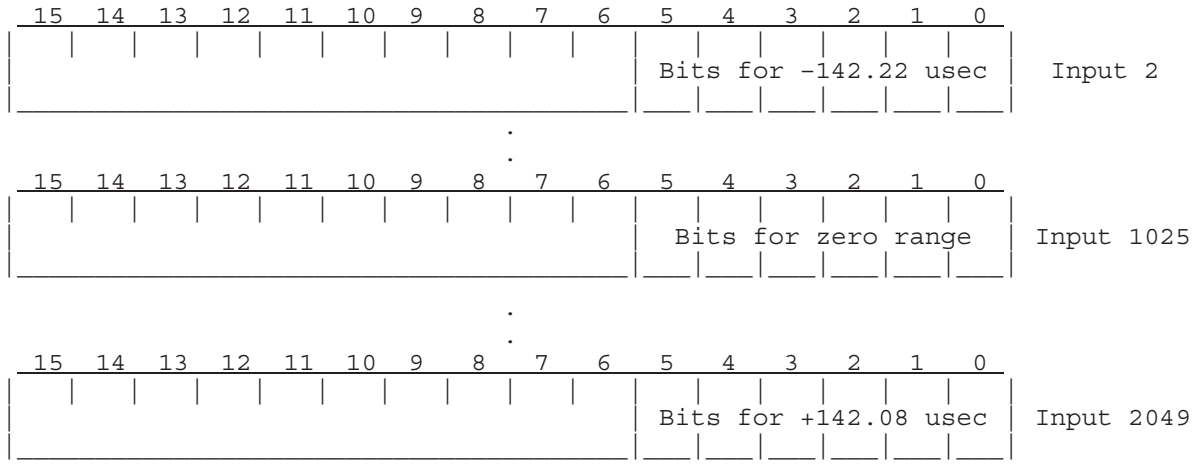
Trigger waveforms are loaded using the TRIGWF command. Four bits in the command word (PW0 through PW3) select which pulsewidths will receive the new waveforms. On power-up, all four pulsewidths are initialized to user-selected waveforms.

The first word following the TRIGWF command specifies the transition point of the POLAR0 polarization control signal. This control signal is either held low or high for the cases of fixed horizontal or vertical polarization, or it alternates from pulse to pulse for fast-switching polarization measurements such as Zdr. The transition point is specified as a value between 0 and 2047, where 1024 represents range zero. These units are the same as the time units for the waveforms which follow, that is, a 2048-word array holding 6-bit trigger patterns. Bit 0 in each of these words affects the TGEN0 digital output line, bit 1 affects TGEN1, etc. The bits are output at a 7.195MHz rate, and the beginning of the 1024th array word (1025th word following the command) corresponds exactly to the instant at which data at range zero are sampled by the RVP8. Note that the output rate can also be interpreted as a new bit coming every 1/48 km. In some cases this is a more useful view.

As an example, suppose we wish to make the TGEN0 output be a 0.42 microsecond pretrigger pulse, with a rising edge exactly five microseconds prior to range zero. This would be done by setting bit 0 in input words 988, 989, and 990 following the TRIGWF command, and leaving all other bit 0's clear. Further, if TGEN1 was to be a 0.14 microsecond marker strobe at 20km, we would simply set bit 1 of input word 1984.



H This bit defines the sense of the control line when horizontal polarization is selected.



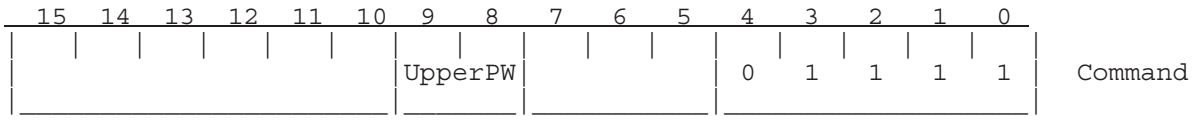
7.13 Define Pulse Width Control and PRT Limits (PWINFO)

The RVP8 is equipped to control the radar transmitter's pulse width and corresponding receiver bandwidth. There are sixteen pulse/bandwidth codes, numbered 0 through 15. The association between codes and pulse widths is completely determined by the needs and capabilities of the particular radar on hand. In some cases, the zero code might represent 0.25 microsecond pulse width, and in other cases it may represent 2.0 microseconds, and some radars may use all sixteen codes whereas others provide fewer options from which to choose. The PWINFO command defines what happens for each pulsewidth code, but does not actually select which code is being used. The later function is performed by SETPWF. For historical reasons, the PWINFO command loads four codes at a time according to the *UpperPW* bits: 00 loads codes 0–3, 01 loads codes 4–7, etc.

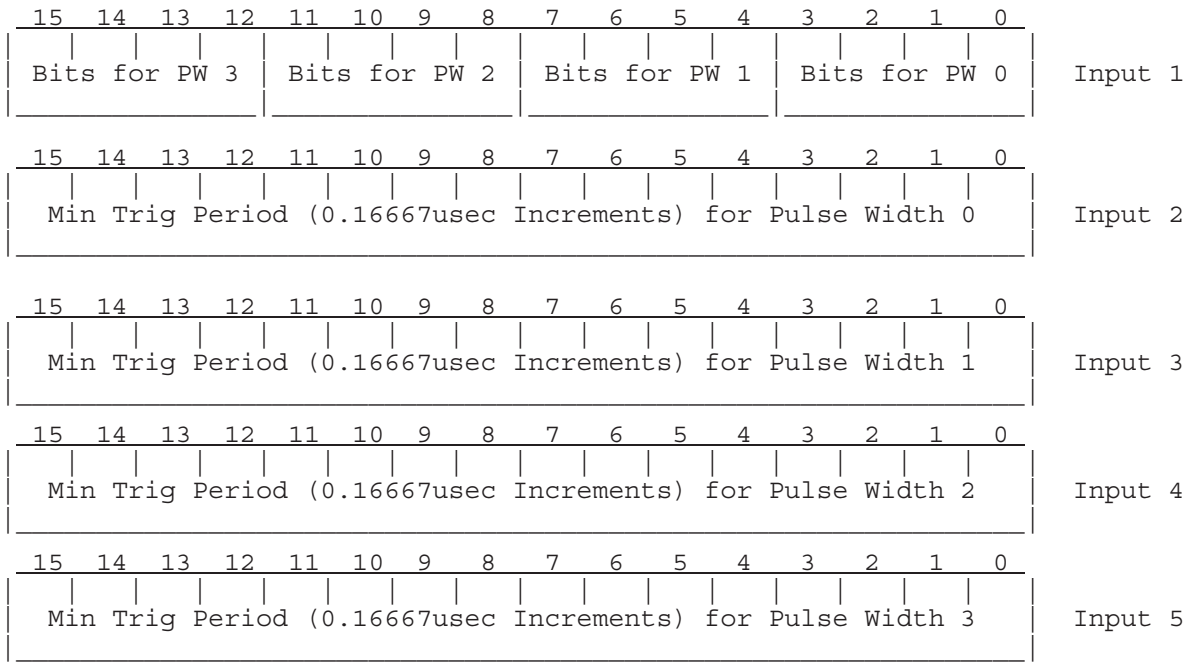
The RVP8 drives four TTL output lines (PWBW0 – 3) which are intended to control the radar pulse/bandwidth hardware. Typically this control is via relays or solid-state switches in the transmitter and receiver. The user decides what state the four lines assume for each pulse width code. This is done using word #1 following the command, which contains four codes packed into one 16-bit word. The power-up default is to drive output line N low for a code of N, keeping all other lines high (Input of 7BDE Hex). The flexibility in defining the output bits usually makes the radar hardware connections very simple. For example, if pulsewidth selection relied on choosing one of four relays, then each PWBWn line could serve directly as a relay driver using the default pattern.

For each pulse width there is a corresponding minimum trigger PRT permitted. This bound is intended to limit the transmitter duty cycle to a safe value under all conditions. PWINFO sets up these minimum PRT's using words 2 through 5 following the command. The maximum frequency of the internal trigger generator is then constrained at each pulse width to the indicated rate. This protection applies at all times, that is, during noise sampling, during ray processing, and during the standby time between rays. The default PRT bounds are 2000, 1000, 750, and 500 Hertz (Inputs of 3000, 6000, 8000, and 12000). If your radar does not use all of the pulse width codes, it is still a good idea to set the unused PRT limits to reasonable values. This way protection is still provided in the event that SETPWF accidently selects one of the unused states. If the internal trigger generator is not being used, then the PRT limits no longer affect the actual trigger rate and transmitter protection becomes the responsibility of the the user hardware. Finally, note that the entire pulse/bandwidth mechanism can be effectively turned off by setting the bit patterns and PRT limits all to the same value.

The PWINFO command can be disabled (for transmitter safety), so that PRT limits cannot accidently be changed by the host computer. When this is done the RVP8 still reads the five input words, but no changes are made to the pulse width and PRT information. Thus, the command I/O behaves the same way, whether enabled or disabled.

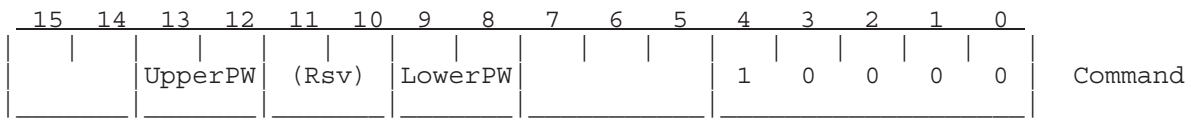


UpperPW Upper two bits of the four 4-bit pulsewidths being defined.



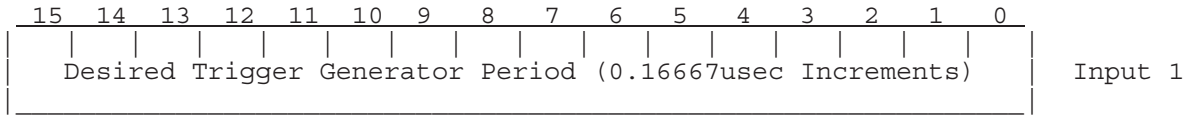
7.14 Set Pulse Width and PRF (SETPWF)

This command selects the pulsewidth and trigger rate. A 4-bit pulse width code is passed in bits (13,12,9,8) of the command word, and selects one of sixteen pulse widths as described under PWINFO. The new radar PRT is passed in word #1. For all processing modes that use a fixed trigger rate, this value defines the trigger period that is output at all times except during noise measurements. For Dual-PRF applications, this word defines the short period (high PRF) rate. The long period is internally computed as either 3/2, 4/3, or 5/4 the short period, and the trigger generator alternates between the short and long rates on each successive ray.

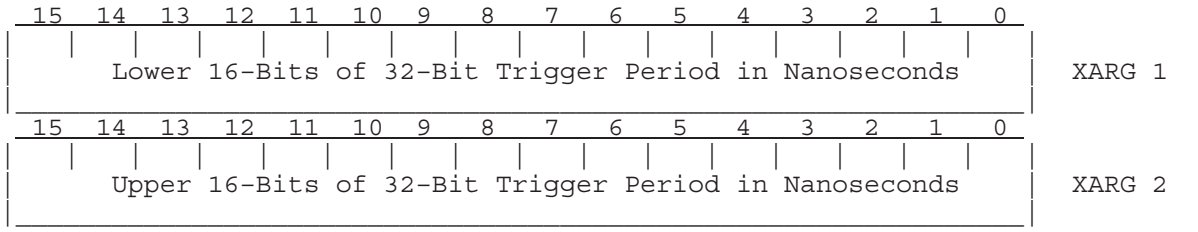


UpperPW Upper two bits of overall 4-bit pulsewidth selection

LowerPW Lower two bits of overall 4-bit pulsewidth selection



When Input #1 is zero, then the arguments take on an alternate form that allows an array of N (up to 64) trigger periods to be specified, and also gives much finer time resolution in the choice of each period. The XARGS command is first used to load an array of *N* 32-bit words that define the trigger period(s) in nanoseconds. The RVP8 will then generate triggers whose shapes (relative starts and widths) are identical for each pulse, but whose periods follow the selected sequence. Trigger patterns such as these are intended to support research customers who use the real–time (I,Q) data stream directly.



7.15 Load Antenna Synchronization Table (LSYNC)

The RVP8 can operate in a mode wherein radar data are acquired in synchronization with the antenna motion along either the azimuth or elevation axis. This special feature frees the user computer from having to separately monitor the antenna angles and request each data ray individually. To use this mode, it is assumed that TAG0-15 are wired to receive azimuth angles, and that TAG15-31 are wired to receive elevation. Angle input may be in the form of either 16-bit binary angles, or four-digit BCD. This synchronization mode is the only one which ascribes any meaning to the TAG inputs; ordinarily they are merely passed on to the user computer as ancillary information.

Antenna synchronization is accomplished by way of a table of trigger angles. This table, which contains between three and 4096 angles, is used to define the angle boundaries for each processed ray. The trigger angles need not be uniformly spaced, nor must they span the full 360-degrees of rotation. This gives considerable flexibility in the choice of angles. For example, if local obstructions cause shadows in the radar image, then those

regions can be skipped merely by omitting table entries in their vicinity. Likewise, as the antenna rotates data can be acquired within one or more sectors by simply specifying the appropriate sets of contiguous bearings at whatever angular resolution is desired. Note that on power-up the angle table is initialized to 360 values corresponding to half-integer-valued degrees from 0.5–359.5.

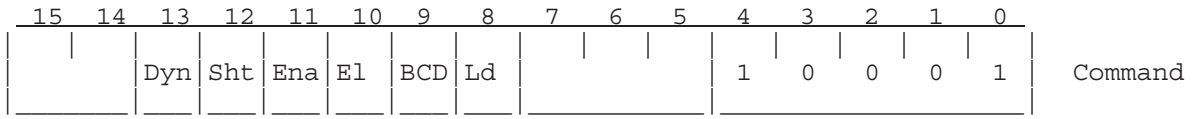
The synchronization algorithm works automatically with either clockwise or counterclockwise antenna rotation, and can tolerate any sequence of changes in direction, for example, if the antenna itself is scanning a sector, or if it is turning erratically. Moreover, the trigger angles do not have to be hit exactly in order to start each new ray – the antenna need only move across them. This minimizes the possibility of losing data due to missing codes in the angle encoders. The RVP8 will automatically produce an output ray after one second of waiting, even if no trigger angles have been crossed. This is to avoid timeouts with the host computer when the antenna is not moving at all.

To use the synchronization mode, the trigger angle table is first loaded using the LSYNC command. The user chooses the number of table entries and then writes the required number of words to the RVP8. The angles must be supplied in a clockwise strictly increasing order, and they must neither reach nor pass zero degrees by the table's end. The first value, however, may be zero. Binary angle representation is used wherein Bit 15 represents 180 degrees, Bit 14 represents 90 degrees, etc. The Ld bit must be set in the command word to indicate that a new table size and set of angles are being loaded. A flag bit is to be set (See GPARM) if errors are detected when loading the table of angles.

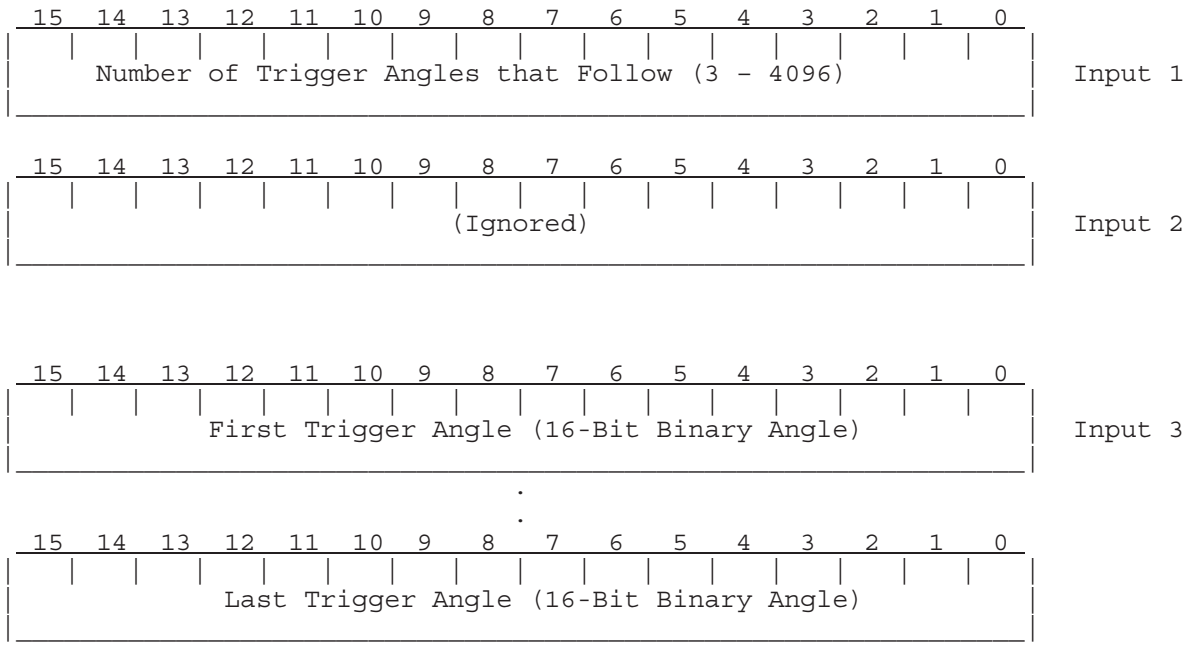
To actually enable synchronized operation the Ena command bit must eventually be set, and EL and BCD should be either set or cleared according to the user's needs. These bits may be used independent of reloading the actual table values. Thus, antenna synchronization may be turned on and off without having to reload the table each time. However, if there were errors when the table was last loaded, the processor ignores the Ena bit and synchronization is forced off. Once enabled, PROC commands are then issued in the usual manner to acquire and process the radar data. Either the single-cycle or free-run PROC mode may be used. Data collection proceeds as usual, except that the rays are now automatically aligned with the trigger angles.

The angle sync algorithm is dynamic and works as follows. Each ray begins immediately upon the user's request, or upon completion of the previous ray when in continuous processing mode. At the start of the ray, the RVP8 finds the pair of sync angles that enclose the previous trigger angle. The current ray then runs until the antenna passes outside of either limit, at which point processing for that ray is terminated. Once this happens, a new trigger angle is assigned based on which limit was crossed.

The maximum number of pulses that will be present in each ray during angle syncing is given by the Sample Size field of the SOPRM command. This is the actual number of pulses that will be used when Dyn=1. When Dyn=0 the actual number of pulses may be less if a trigger angle is crossed before the full pulse count can be accumulated

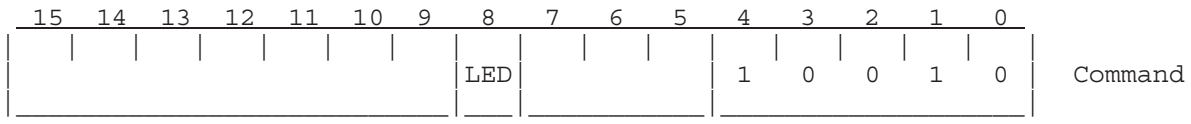


Dyn	If set, then synchronize the endpoints of each ray but let their angular width vary by whatever is required to collect the SOPRM number of pulses. If clear, then the angular width of each ray is fixed (between successive table entries) by adjusting the pulse count and reinterpreting the SOPRM sample size as a maximum value.
Sht	Synchronized rays ordinarily will be the full width of successive table entries (in the static case) or the full requested pulse count (dynamic case). This bit modifies the behavior in both modes to allow short rays to be produced, wherein the pulse count is less than expected due to encountering some feature in the CPI (usually a trigger transition) that would normally have resulted in the entire ray being thrown out. When this bit is set it becomes the responsibility of the user's code to check the actual pulse count that went into each ray and manually discard those that are too short to contain useful data.
Ena	Enables antenna synchronization.
El	Synchronization is based on TAG1531 (Elevation) inputs, else TAG015 (Azimuth) is used.
BCD	Specifies that TAG angle input is in the form of 4-digit Binary Coded Decimal; otherwise, a 16-bit binary angle is assumed.
Ld	Indicates that a new table size and array of values follow the command. If Ld = 0, then LSYNC is a one-word command only. Otherwise, the following words are used to load the new table:



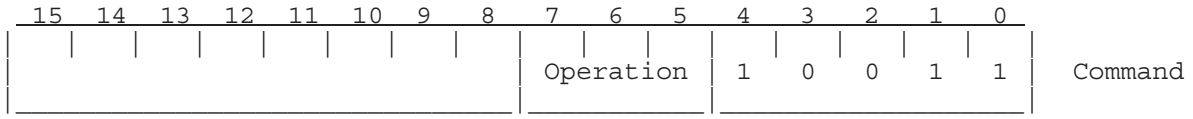
7.16 Set/Clear User LED (SLED)

This command simply turns the red user LED on and off under program control. The LED is on during the initial running of internal diagnostics, and then remains off unless changed by this command. Note that the red LED can be configured to serve as an internal activity indicator (see TTY setups), in which case this command has no effect.



7.17 TTY Operation (TTYOP)

This command controls the TTY "chat mode" interface to the host computer. The command can simulate the typing of characters on the RVP8's setup TTY. Characters entered in this manner are indistinguishable from those typed on the actual TTY; hence, whatever one can do via the TTY, one can also do via this command. The RVP8 sends all TTY output to whichever stream (TTY, or host computer) provided the most recent input character. This command is also used to monitor the graphical data from the special scope plotting modes.



The operation codes are as follows:

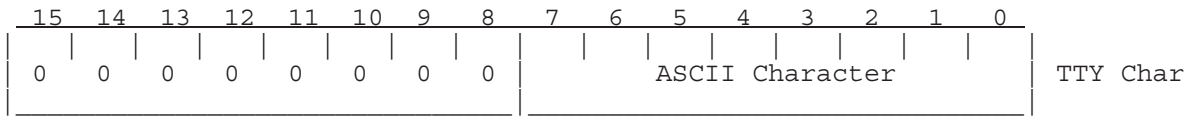
0 Sends the ASCII character in the upper byte of the word to the RVP8 as if it had been typed on the setup TTY's keyboard.

1 Allow scope plotting data to be output whenever a plot is being drawn. All relevant status and data words are output once upon each receipt of this command. Subsequently, status and data will be output only when a change has taken place.

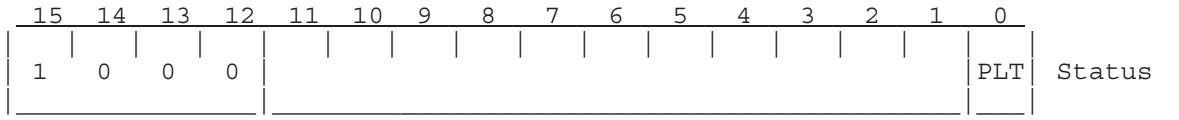
2 Disable the scope plotting output data.

Any of the following types of data may be output by the RVP8 while the TTY monitor is running. The order of arrival of each data type is indeterminate, but all multi-word sequences will always be output as contiguous words.

Individual "TTY" characters generated by the RVP8 are output in the low byte of the word, with the upper byte set to zeros.

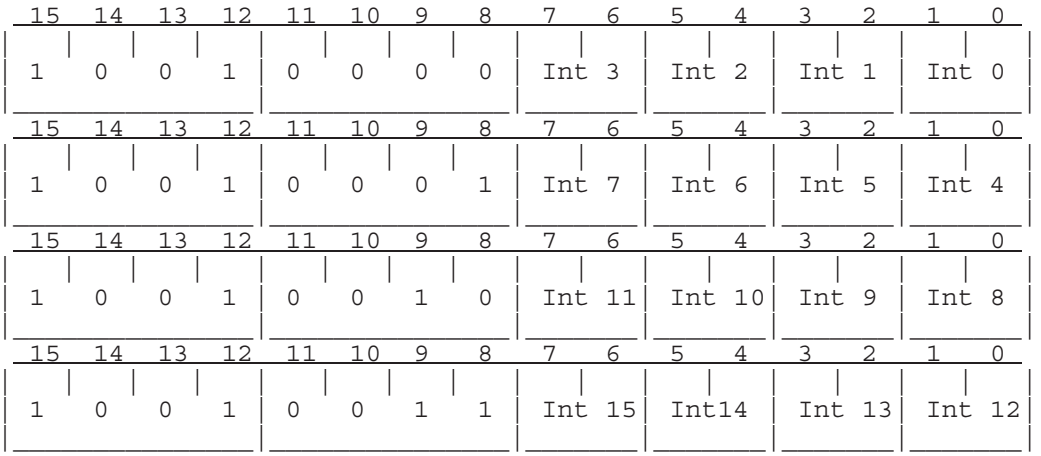


The status of the plotting modes is given in the following word.

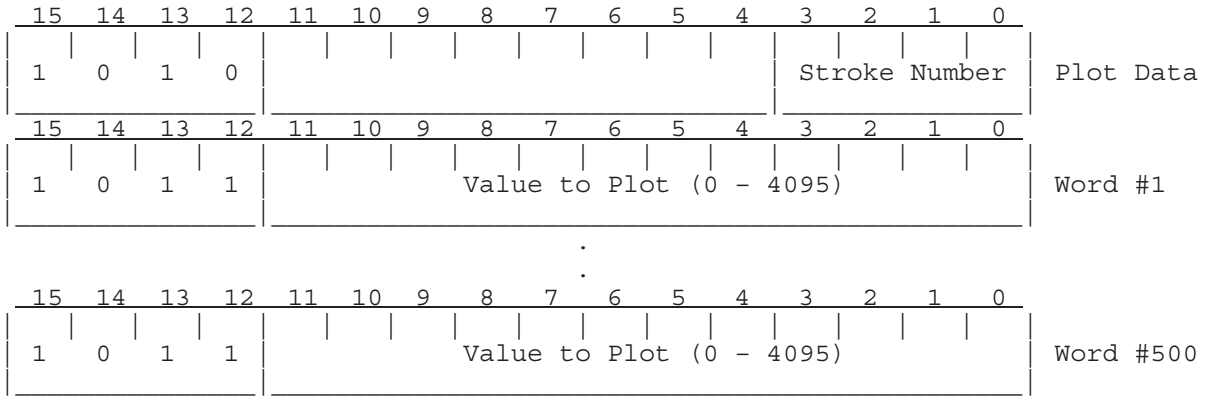


PLT Indicates that a scope plot is being drawn now.

The 2-bit intensities of each of 16 possible strokes of data is given in the following 4-word sequence. An intensity of zero represents "OFF"; one, two and three are successively brighter.



The data for each stroke of the plot is given by the following sequence of 501 words.

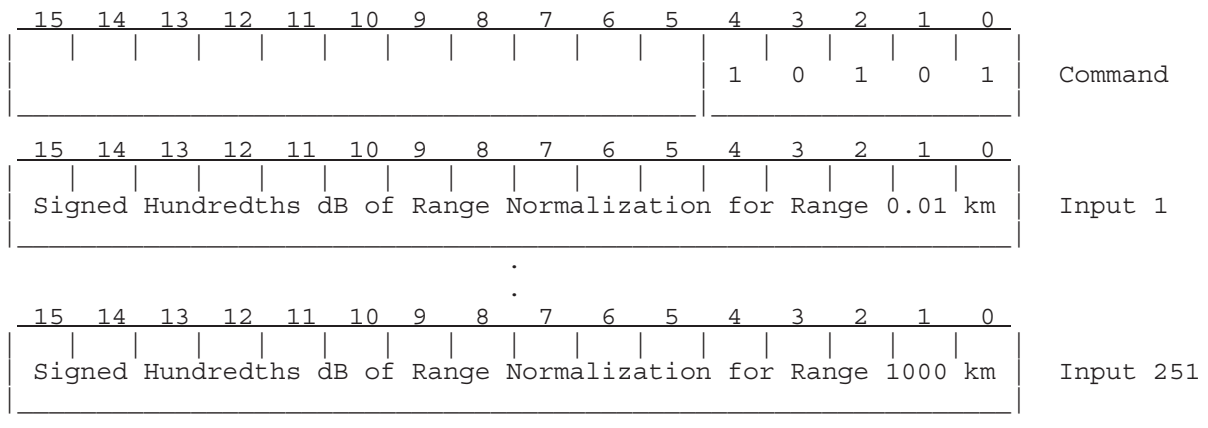


7.18 Load Custom Range Normalization (LDRNV)

Reflectivities computed by the RVP8 are ordinarily corrected for range effects by adding an offset in deciBels equal to 20 log(R / 1km), where R is the range in kilometers. This correction is based on a simple filled beam geometry, and is sufficiently accurate for most meteorological observations. The LDRNV command is provided for applications in which an alternate custom range correction is required, for example, if the radar receiver's LNA were to be driven by an external user-supplied STC waveform.

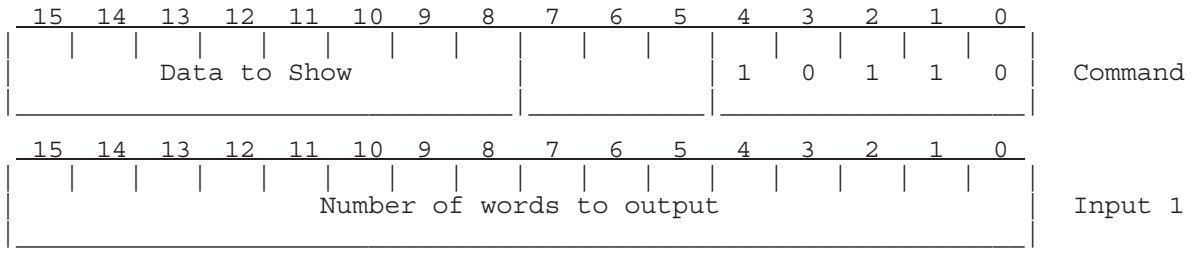
LDRNV loads a 251-word custom correction table holding values in hundredths of deciBels over five decades of log(range) from 0.01km to 1000km. There are 50 table entries per decade of range. Thus, the range in kilometers corresponding to an input word #N is 10[{{N-1}<devide>50}-2], and the default correction table (automatically used on power-up) is simply $40(N - 101)$. The table values are stored and interpolated whenever the RVP8 loads a new range mask (See LRMSK), at which point custom values for the actual user ranges are computed. The LDRNV command need be issued only once, but it must be done prior to choosing the working set of range bins.

The linear intervening gas attenuation correction (See SOPRM) is always added to the reflectivity data, regardless of whether default or custom range normalization is in effect. If this is undesirable, the intervening gas slope should be set to zero.



7.19 Read Back Internal Tables and Parameters (RBACK)

This command permits some of the RVP8 internal tables to be read back for confirmation and diagnostic purposes. This command would not generally be used during normal data acquisition and processing.



The data that can be returned are:

0 Full operational parameter table from last SOPRM command.

1 Ray history array consisting of six words per ray for the last 40 rays (in reverse time order) that were processed. Each six-word group holds

   a. Actual number of samples that went into the ray
   b. Time since the last ray (in tenths of ms)
   c. Ending azimuth TAG bits
   d. Ending elevation TAG bits
   e. Starting azimuth TAG bits
   f. Starting elevation TAG bits

2 Angle sync table from last LDSYNC command.

3 *Reserved (was AGC table)*

4 Filter selection array from the last LFILT command. This returns the filter selection codes, one per word, for all range bins described by filter slot #0.

5 *Reserved (was STC table)*

6 Custom range normalization from last LDRNV command.

7 Samples of the TAG input lines at 4ms intervals. The sampling begins at the moment the RBACK command is received, and continues until the output count is reached. Each 32-bit sample is output as a pair of 16-bit words:

   a. Azimuth (TAG bits 0 15)
   b. Elevation (TAG bits 16 31)

8	Doppler clutter filter coefficients (Same format as for LFCOEFS command)
9	*Reserved (was LOG clutter filter coefficients)*
10	Range mask spacing in cm for each pulsewidth
11	Current value of UIQ bits from Set/Clr all prior operations
12	Individual threshold configuration for each data type. This allows read back of the threshold table set with the THRESH command (7.29 Set Individual Thresholds (THRESH) on page 348). Outputs 7 words per data type. Datatypes in the order specified in the selection mask.
13	Extended parameter information defined in *struct dspExParmIO*.
14	Minimum and Maximum values of the optional I/O-62 A/D converter, sampled over at least one complete pulse period. Sixteen bit signed outputs represent the full range of the A/D converter. When the *RVP88D* backpanel is connected, the first Min/Max pair samples the *LOGVideo* input, the second pair samples the *CathodePulse* input, and the third and fourth pairs sample internal levels.
15	Returns an array of *struct rvp8SpecFiltIO* structures for each of the non-zero clutter filter definitions, beginning with #1. This is the same format used by the LFSPECS command to define each individual clutter filter. The order is as defined in the PPRMS_N_* #defines in rvp8.h.

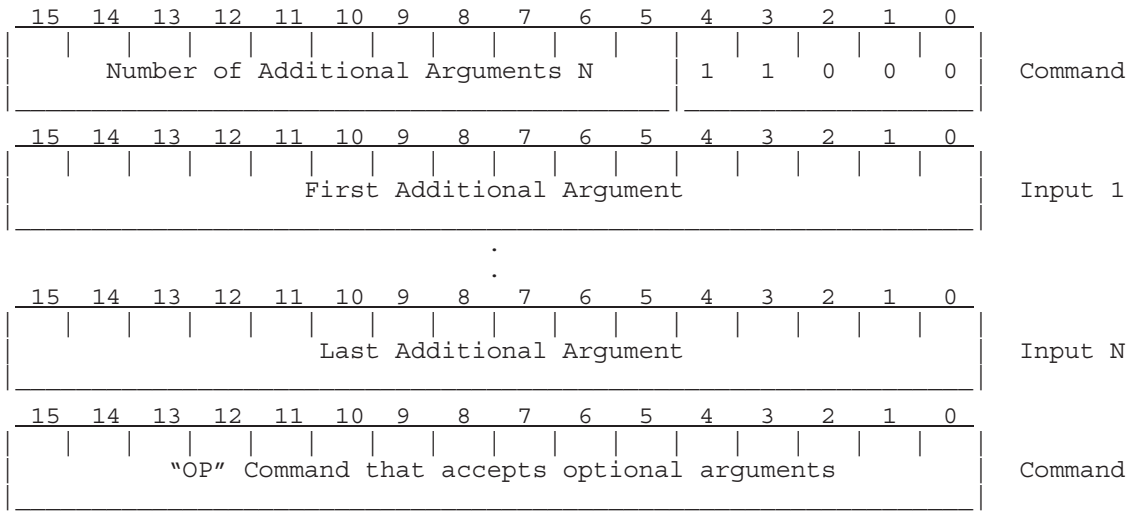
7.20 Pass Auxiliary Arguments to Opcodes (XARGS)

This command provides a backward compatible mechanism for supplying additional (optional) arguments to other opcodes. The command may be used freely in the RVP8's instruction stream, even if the opcode being modified does not expect any optional arguments. XARGS will be a NOP in that case.

To supply optional arguments to another opcode "OP", the XARGS command is first executed with the additional argument count encoded in its upper 11-bits. This is followed by the array of between 0 and 2047 additional arguments. At this point the XARGS command is finished and the "OP" command is fetched as the next instruction. "OP" will execute normally, except that the additional arguments from XARGS can be picked up after its own input list has been read to completion.

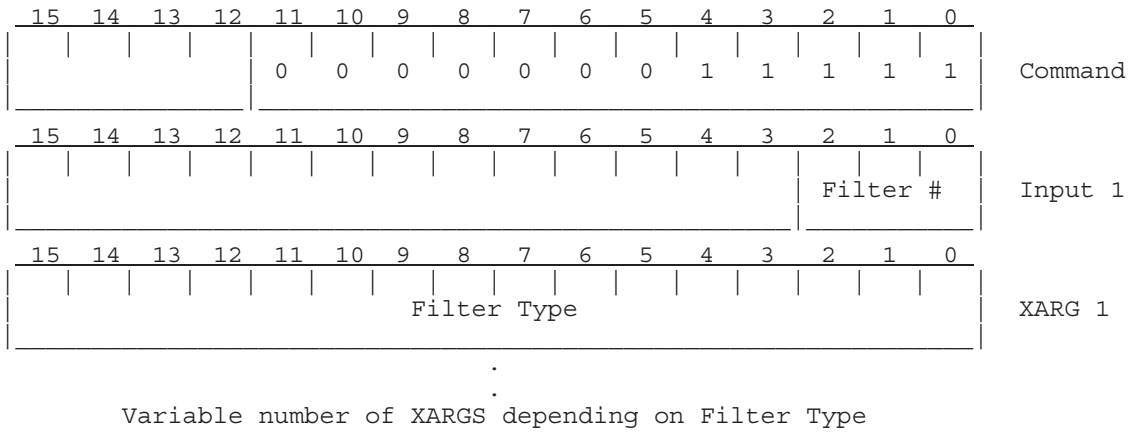
XARGS affects only the opcode that immediately follows it. The entire list of optional arguments is discarded after "OP" executes, even if "OP" did

not use some or all of the list. However, if "OP" is yet another XARGS command, then the additional arguments that it supplies will be appended to the first set. In this way, XARGS can supply an arbitrarily large number of additional arguments.



7.21 Load Clutter Filter Specifications (LFSPECS)

The RVP8 allows seven different clutter filters (plus the fixed all-pass filter) to be resident at once, so that an appropriate filter can be selected and applied to each processed ray based on Range, Azimuth, and/or Elevation. The LFSPECS command allows this suite of filters to be redefined on the fly.



The "Filter Number" tells which filter definition slot is being modified, and the "Filter Type" conveys the type of clutter filter to construct. The command is then followed by additional XARGs that give the specific filter parameters. Beginning with the Filter Type, the complete XARG list *is a struct rvp8SpecFiltIO* (See *include/rvp8.h*) for each of the following filter types.

Type:0 SPFILT_FIXED Fixed Width Spectral Filter

Legacy clutter filter inherited from the RVP6/7, specified by a width parameter telling how many points to remove (center zero velocity point, plus one side), plus an "Edge Points" parameter telling how many points to minimize on each side of the gap to compute the end points of a linear interpolation to fill the gap.

Type:1 SPFILT_VARIABLE Variable Width Spectral Filter

Similar to the fixed width filter except that the width parameter is interpreted as a minimum width, and a third parameter indicates the maximum width. The actual clutter gap width will be dynamically determined at each bin based on the slopes of the spectral terms. Linear interpolation of the gap (based on "Edge Points") is the same as above.

Type:2 SPFILT_VARLSQ Variable Width / Quadratic interpolation

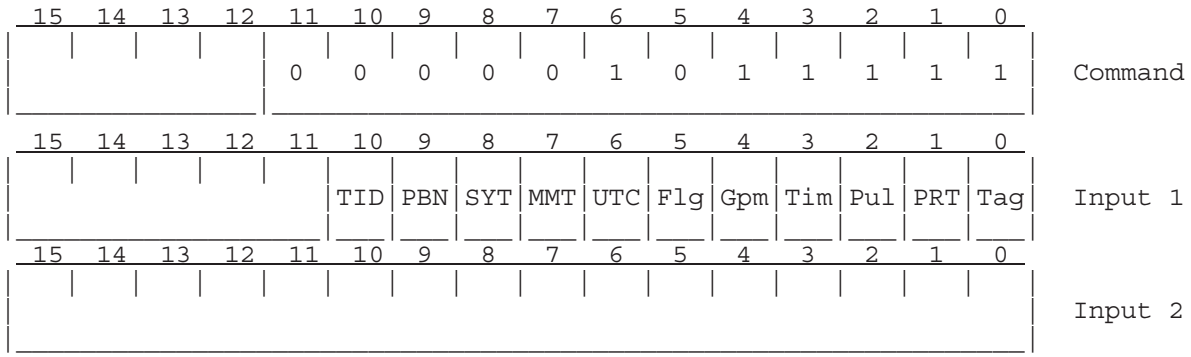
Similar to the variable width filter except that quadratic gap interpolation is used. This filter is experimental and should not be used.

Type:3 SPFILT_GMAP Gaussian Model Adaptive Processing Spectral Filter

This is the RVP8's most advanced clutter filter, combining the best techniques for determining the clutter gap width and restoring whatever low-velocity spectral points are removed. This filter is characterized by a single parameter, which is the assumed clutter width expressed as a physical velocity.

7.22 Configure Ray Header Words (CFGHDR)

The processed data that are output by the PROC command may contain optional header words that give additional information about each ray. This command configures the set of words that makeup each header. There are (up to) thirty two different choices of words or groups of words to include, as indicated by the bit mask following the command. Setting a bit requests that those words be included in the header, and be placed in the order implied by the sequence of the bits. Leaving all bits clear will suppress the header entirely; though this can also be done without changing the configuration via the NHD (No-Headers) bit in SOPRM Input #2.



Tag

Four words containing two 32-Bit TAG samples, one from the beginning and one from the end of the ray:

Word #1	TAG150	Start of Ray
Word #2	TAG3116	Start of Ray
Word #3	TAG150	End of Ray
Word #4	TAG3116	End of Ray

- When the RVP8 is operating in dual PRF mode, bit zero of the "start" TAG word is replaced with a flag indicating that the ray's PRF was low (0) or high (1).
- When trigger blanking is enabled, bit zero of the "end" TAG word is replaced with a flag indicating that the trigger was blanked (0) or normal (1). Note that the data within a ray are considered to be invalid if any of the pulses that were used to compute the ray were blanked.

Also, the RVP8 will output all zeroed data whenever a ray contains any blanked pulses.

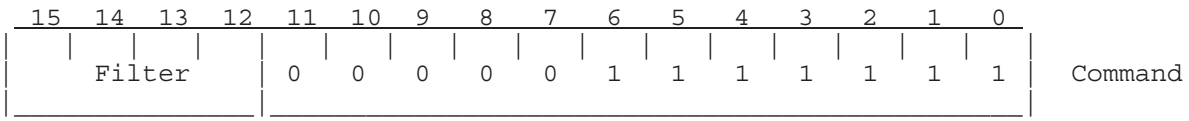
PRT	PRT (Pulse Repetition Time) measured at the end of the ray. Same format as GPARM Word #30. The measured PRT's are forced to 0xFFFF (the maximum unsigned value) whenever the external trigger is expected but missing.
Pul	Number of pulses that were used to compute the ray.
Tim	Milliseconds universal time (0999), sampled at the end of the ray.
Gpm	GPARM. Sends a copy of the 64-word GPARM output with each ray.
Flg	Ray Flag word: Bit 0: Dual PRF is in the low PRF state Bit 1: Trigger is blanked for this ray Bit 2: This ray is from one of the PRFSECT special sectors Bits46: Tells which PRFSECT when Bit-2 is set
UTC	3-word universal time, sampled at the beginning of the ray Word 1: Milliseconds (0999) Word 2: Low 16-bits of 32-bit UTC time Word 3: High 16-bits of 32-bit UTC time
MMT	MisMatched Timeseries bits (playback versus RVP8 configuration). See the MMTS_* flags in dsp.h.
SYT	IFD system clock time at the beginning of the ray Word 1: Low 16-bits of 32-bit clock counter Word 2: High 16-bits of 32-bit clock counter
PBN	Timeseries playback version number
TID	Task ID encoded as *struct rvp8TaskID_IO* (14 words total)
PedINU	Auxiliary pedestal and INU information encoded as *struct rvp8AuxPedINU*, giving full angle and position information for moving platform systems (18 words total).

7.23 Configure Interference Filter (CFGINTF)

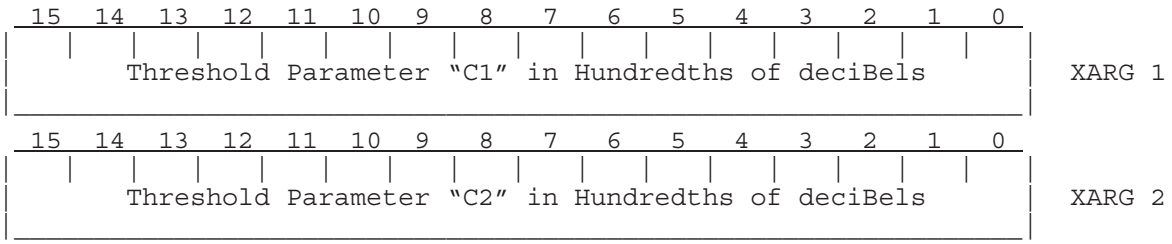
The RVP8 can optionally apply an interference filter to its incoming (I,Q) data stream, with the goal of rejecting occasional and sparse interference from other (usually man-made) signal sources. The CFGINTF command is used to choose which filtering algorithm will be applied, and to configure its operation via additional XARGS parameters (See 7.20 Pass Auxiliary Arguments to Opcodes (XARGS) on page 339).

If the XARGS are not supplied, then the filter parameters will simply retain their previous values. Thus, CFGINTF with no XARGS can be used to turn

the interference filters On/Off without making any other changes to their threshold constants. Likewise, if only XARG 1 is supplied, then that single threshold value will be used for both C1 and C2.

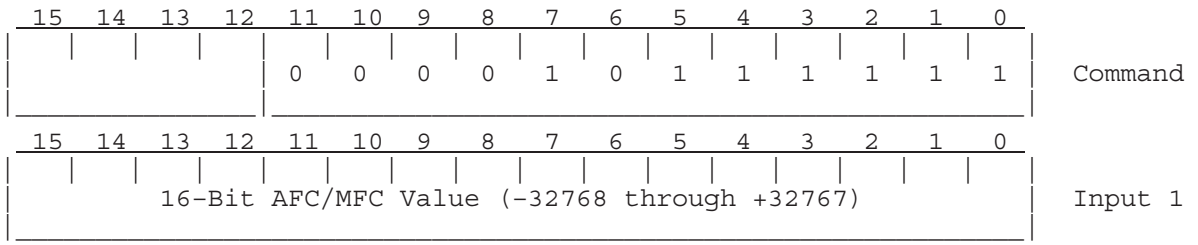


Filter — Chooses which interference algorithm should be run. See 6.1.5 Interference Filter on page 212 for a description of the available algorithms.
0: None (Interference filtering is disabled)
1: Alg.1 (Traditional JMA Algorithm)
2: Alg.2 (Alg.1 optimized for additive interference)
3: Alg.3 (Alg.2 with better statistics)
We recommend that you choose Alg.3 for general operational use. The other algorithms are included mostly for historical reasons.



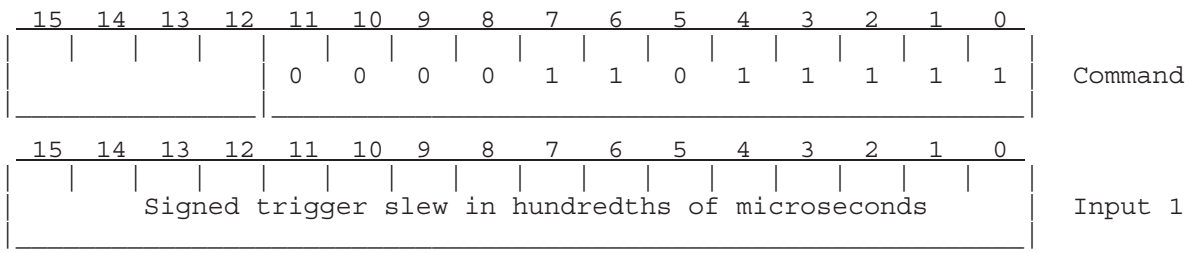
7.24 Set AFC level (SETAFC)

This command sets the AFC level to a given value. The signed 16-bit span is identical to GPARM Output #51 which shows the present AFC level, that is, corresponding to the -100% to +100% AFC range that is defined in the **Mb** menu. The RVP8 will automatically convert the new level into whatever analog or digital AFC output format has been configured. The only exception is for the Motor/Integrator type of AFC loop, in which case this command does nothing.



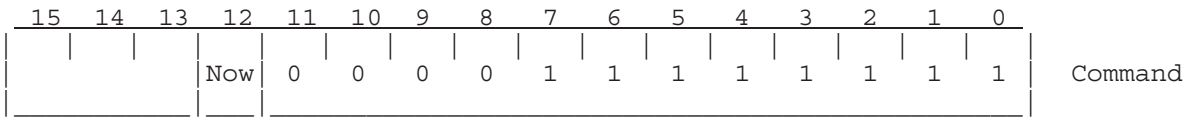
7.25 Set Trigger Timing Slew (SETSLEW)

The Mt menu allows you to select a subset of triggers that can be slewed "left" and "right" in order to place the burst pulse accurately at range zero. This command allows you to manually set the present amount of slew. The input argument is in hundredths of microseconds, that is, ranging from -327.68µsec to +327.67µsec. The actual span permitted by the RVP8 is ±20µsec. This is the same format used in GPARM Output #56 which shows the present slew value.



7.26 Hunt for Burst Pulse (BPHUNT)

This command starts up the internal procedure to hunt for a missing burst pulse when we are uncertain of both its time and frequency. Depending on how the hunting process has been configured in the **Mb** menu, the whole procedure may take several seconds to complete. The RVP8's host computer interface remains completely functional during this time, but any acquired data would certainly be questionable. GPARM status bits in word #55 indicate when the hunt procedure is running, and whether it has completed successfully.

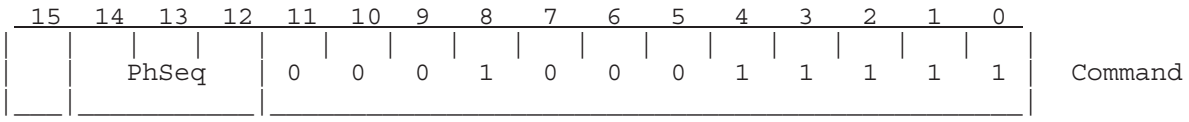


Now — Forces the hunt procedure to be started even if the burst pulse is already present. Normally the procedure will only be started when the burst pulse is missing at the time BPHUNT is given.

7.27 Configure Phase Modulation (CFGPHZ)

This command configures the RVP8 phase control output lines, which determine the relative phase of each transmitted pulse. In some cases the phase sequence that is chosen will also have side effects elsewhere in the processor, for example, different algorithms may be used in Random Phase mode according to the transmit sequence that is requested.

Some of the phase sequences chosen by CFGPHZ also expect additional arguments to have been supplied by the XARGS command. Phase sequences are expressed as a list of *N* 16-bit binary angles representing the desired phase sequence. The sequence is assumed to be periodic with period *N*. The **Mz** command defines the correspondence between phase codes and phase angles, and is described in 4.2.10 Mz — Transmissions and Modulations on page 155.



PhSeq=0 — Selects **No Modulation**. The RVP8 outputs a constant default phase request as defined in the **Mz** menu.

PhSeq=1 — Selects a **Random Phase** sequence. This is also the default phase modulation that will be output following power-up. From the set of valid phase codes that are defined in the **Mz** setup section, a random code is automatically chosen for each pulse. Each code has an equal probability of being chosen each time, and the choice is independent of any previous state. No XARG words accompany this command.

PhSeq=2 Selects a **User Defined** sequence. If no XARGS have been supplied, then the RVP8 outputs the default idle phase that is defined in **Mz**. If XARGS are supplied, then they are interpreted as a sequence of 16-bit binary angles. The RVP8 will make the best match between each desired angle and the closest realizable angle that the phase modulation hardware can produce. The maximum length of the sequence is 1024 pulses.

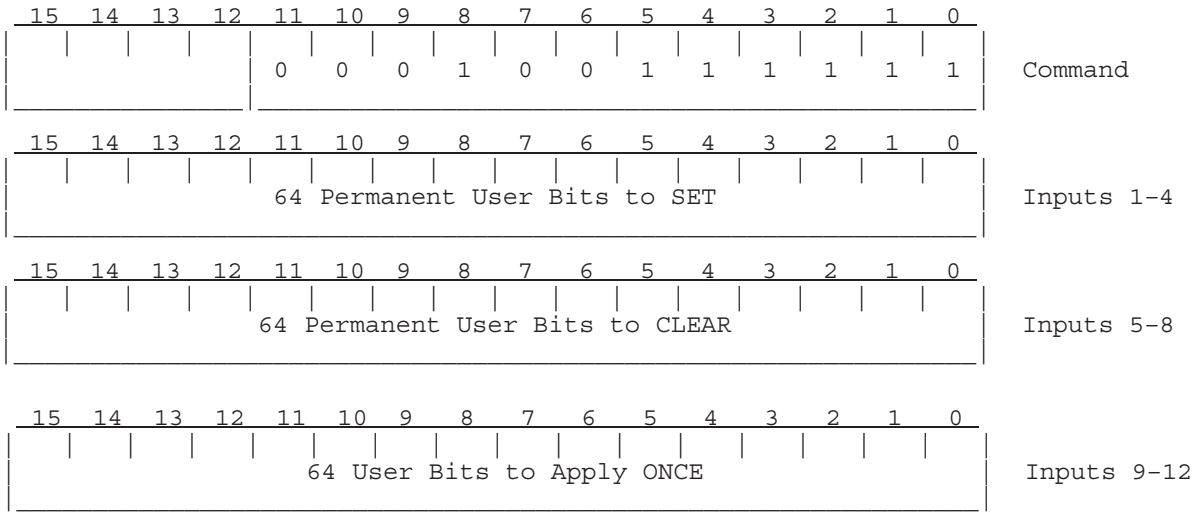
PhSeq=3 Selects the **SZ(8/64)** sequence. This is a systematic code due to Sachidananda and Zrnic, which does a nice job separating and recovering first and second trip echoes in "Random Phase" mode. It will usually perform better than a truly random transmit sequence, especially when the processing interval is fairly short (as little as 32-pulses). With no XARGS, the RVP8 automatically generates the phase sequence using the closest realizable angles that the phase modulation hardware can produce. This is the recommended way to invoke SZ(8/64) coding. However, you may also supply your own 32-pulse angle sequence.

7.28 Set User IQ Bits (UIQBITS)

Load user-specified bits that will be included with the pulse headers in the RVP8 TimeSeries API data stream. The permanent Set/Clr bits are updated in the signal processor and retain their value from the last time they were defined. These bits are then repeated into all pulse headers. The ONCE bits, however, are transitory and will appear in only one pulse header each time they are set.

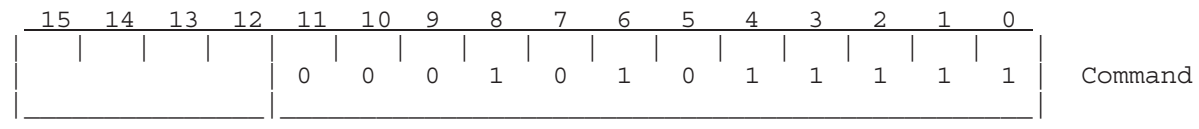
A FIFO history of the permanent bits is maintained so that the bits can be associated with the data being acquired right now as the UIQBITS opcode is executed. Each 16-bit command arg specifies bits to Set/Clr in successive bytes of the structure. This allows user code to safely change some bits without affecting others.

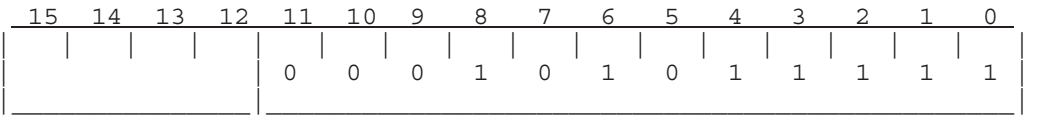
The user bits from separate calls will never be collapsed into a single pulse header, even if the header and bit times indicate that they could. This means that each UIQBITS opcode will always result in at least one pulse header being tagged with exactly those data. This is generally what you want, since no other exact outcome could be guaranteed based on time-of-arrival alone.



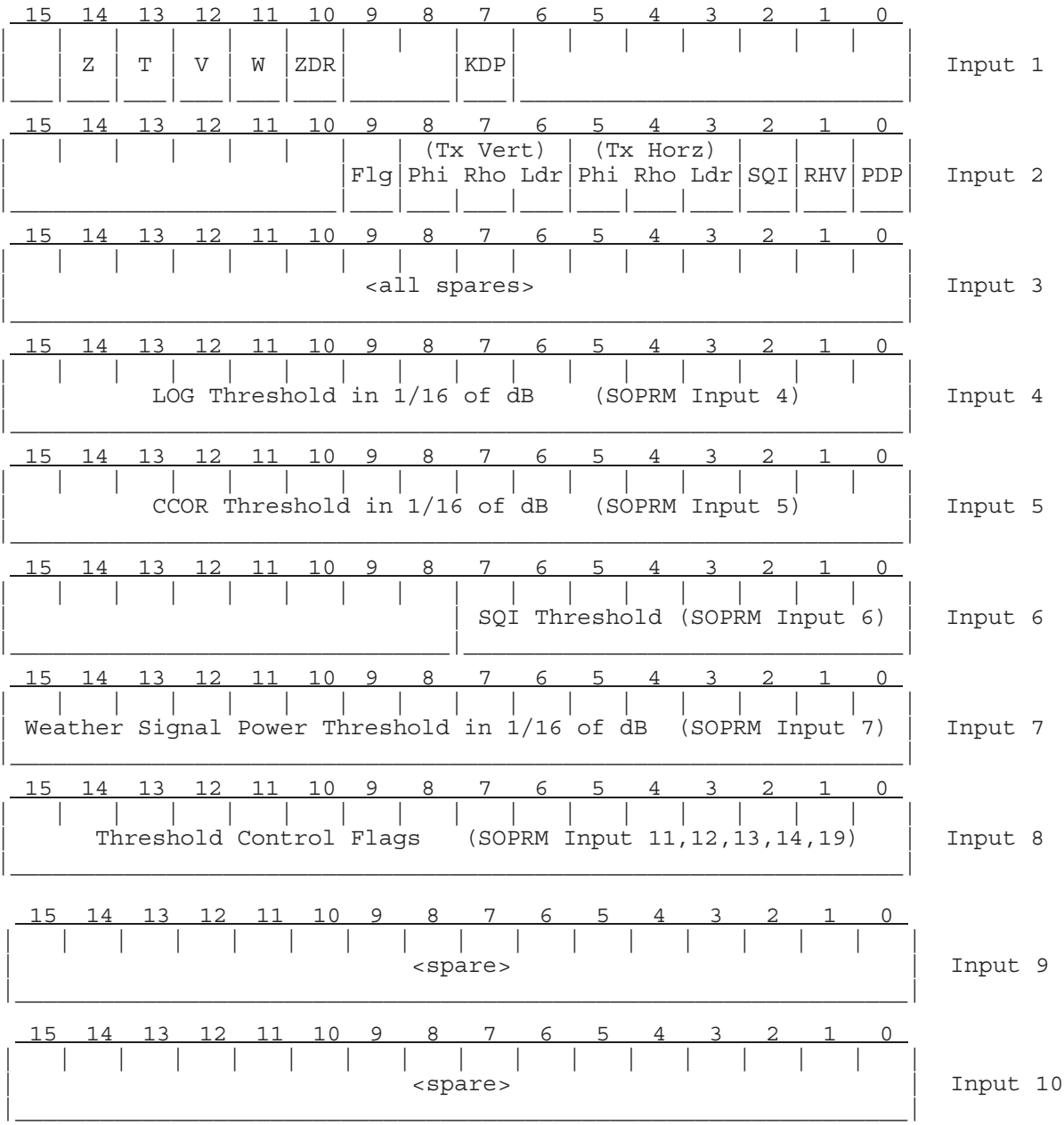
7.29 Set Individual Thresholds (THRESH)

The SOPRM command in 7.3 Setup Operating Parameters (SOPRM) on page 279 allows you to configure four threshold numbers used by all data types, and to select the threshold control flags for five of the data types. See that section for detailed documentation on how the thresholds work. Use the THRESH command if you wish to apply different threshold numbers to different data types. Using this command you can individually set the thresholds and mask used for each data type, or for groups of data types. Note that the GPARM command will read out the threshold numbers set for velocity. To read back the numbers for each data type use the RBACK command (7.19 Read Back Internal Tables and Parameters (RBACK) on page 338).





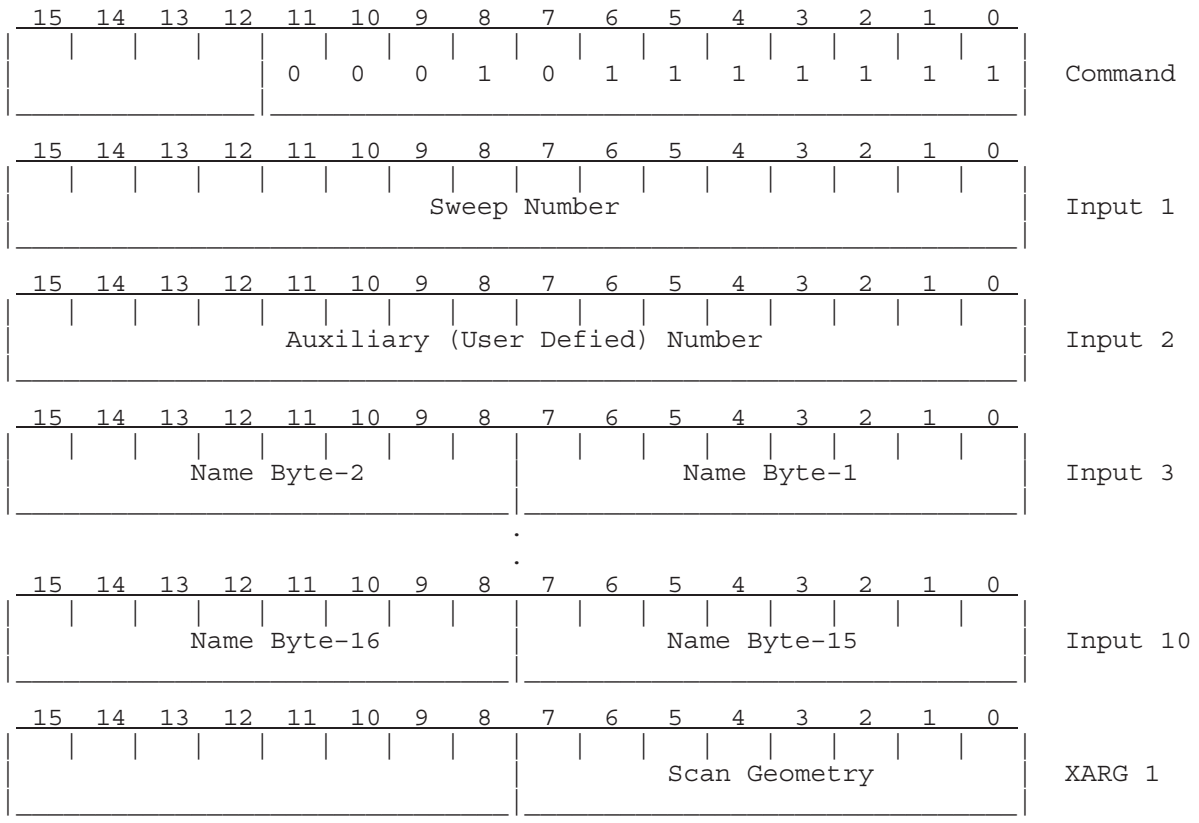
The first three words supply a mask that indicates which data types are being set:



7.30 Set Task Identification Information (TASKID)

This command allows the user to "name" the (I,Q) data that are currently being acquired by the RVP8. This naming information then becomes associated with these data, and is available in the pulse information structures (*struct rvp8PulseInfo*) that are read from the Timeseries API. The *iAqMode* field of the pulse headers (*struct rvp8PulseHdr*) will be incremented each time a TASKID opcode is received, but the continuous flow of (I,Q) data from the RVP8/Rx card(s) will not be disturbed in any way.

The TASKID command defines a 16-character Null-terminated name, along with a 16-bit sweep number and 16-bit auxiliary (user defined) number. You may use all sixteen characters of the name, as it is stored internally in seventeen slots. The "Sweep Number" and "Scan Geometry" (one of SCAN_xxx parameters) should be filled in with values best approximating those notions. "Auxiliary Number" may be filled in with any value that you find meaningful.

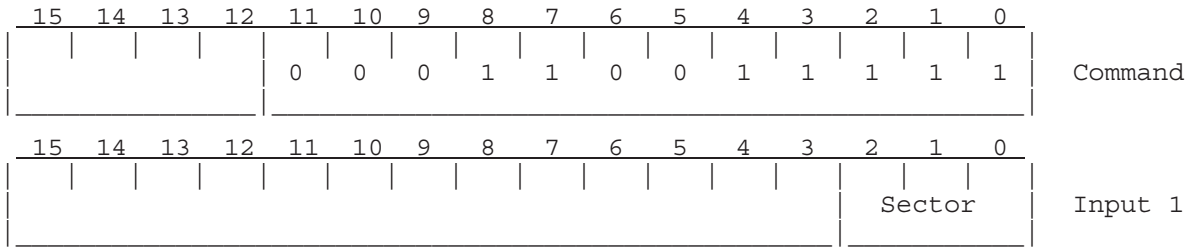


7.31 Define PRF Pie Slices (PRFSECT)

This command supplements the SETPWF command (7.14 Set Pulse Width and PRF (SETPWF) on page 330) and allows an alternatetrigger PRF to be generated within prescribed AZ/EL sectors. As many as eight different trigger sectors can be defined by invoking PRFSECT for each separate region. The trigger pattern will then automatically change whenever the antenna enters any of these regions, but the timeseries data will remain continuous and uninterrupted throughout each change. The motivation behind PRFSECT is that it allows a complete volume scan to run with PRFs that have been optimized to the radar echoes in all directions. Some caveats should be observed:

- Dual-PRF unfolding can not possibly work properly at PRF sector boundaries, so we recommend not using Dual-PRF and PRFSECT at the same time.
- When PRF sectors are used in conjunction with angle sync'ing, it is best to set the PRF sector boundaries at the midpoint between individual sync angles. This will prevent the PRF seam from bobbling between two adjacent sync angles.

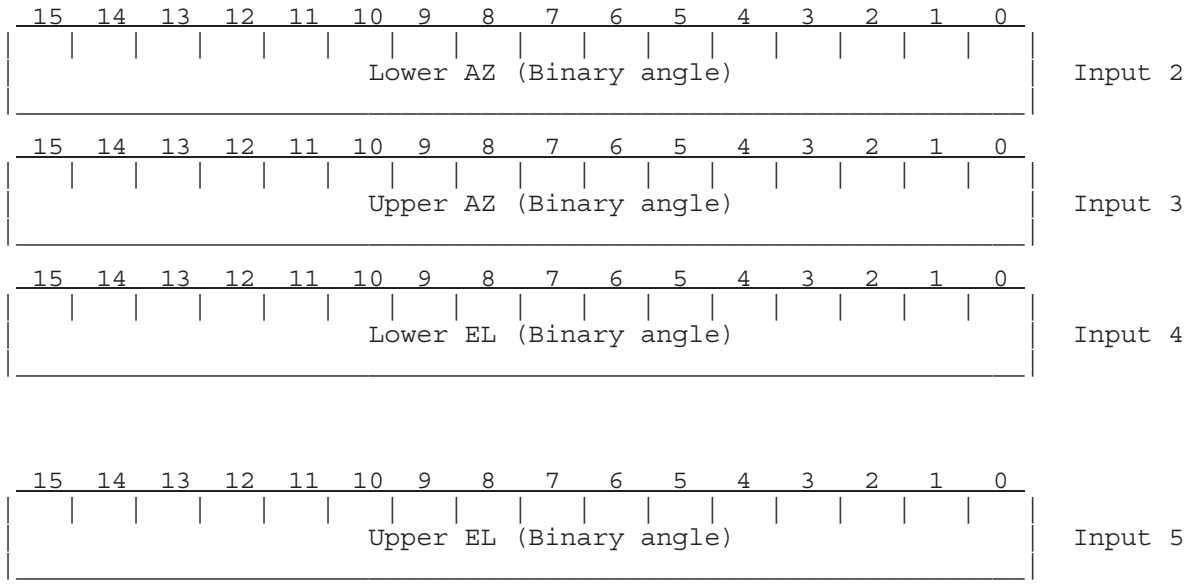
The SETPWF opcode completely erases any alternate sectors that have been setup so far. Thus, the PRFSECT command can only be used after SETPWF has established the pulsewidth and default trigger rate for the entire AZ/EL scan volume.



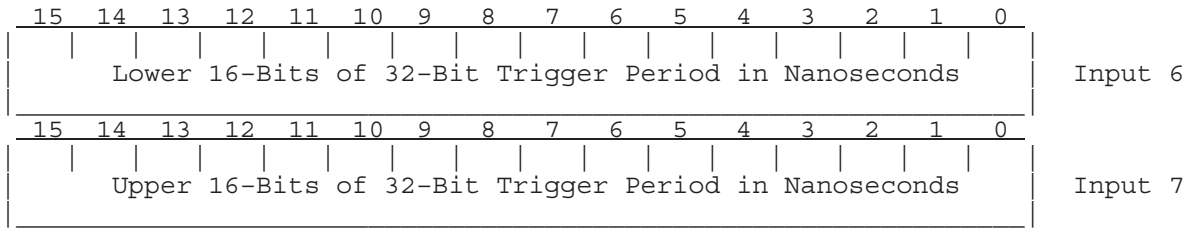
Sector — Selects which sector number is being defined to have an alternate trigger pattern. This is simply an arbitrary index from 0-7.

The following four arguments define a solid sector in azimuth and elevation within which the alternate trigger pattern will be used in preference to the default (SETPWF) pattern. When the current AZ/EL angle pair is contained in more than one defined sector, then the trigger pattern from the lower numbered sector will be used. Note that the sector bounds are inclusive, that is, they include the Upper/Lower AZ/EL

boundaries themselves. This convention makes it simpler to define several contiguous regions without generating slivers in between.



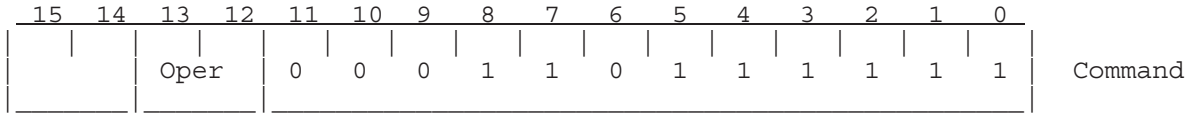
The following two arguments specify a trigger period in the same manner as the optional form of the SETPWF command.



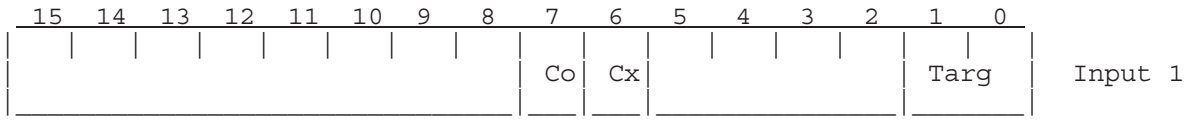
7.32 Configure Target Simulator (TARGSIM)

The RVP8 contains a built-in target simulator tool that can test and debug processing algorithms that work with multiple trip returns. Several real physical targets can be simulated, each having a range span measured in kilometers, a Doppler shift in Hertz, and an echo power relative to the saturation level of the receiver. The echoes are placed in range exactly according to how they have been illuminated by whatever sequence of pulses have been transmitted so far. Multiple trip returns and range folding are all modeled correctly.

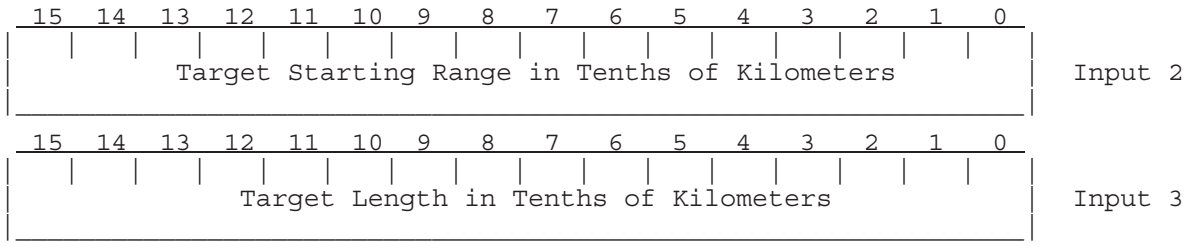
The target simulator can be used with both live and simulated (I,Q) data (See LSIMUL opcode in 7.10 Load Simulated Time Series Data (LSIMUL) on page 323). In the former case, it allows you to overlay simulated physical targets on top of real physical targets from the radar receiver.

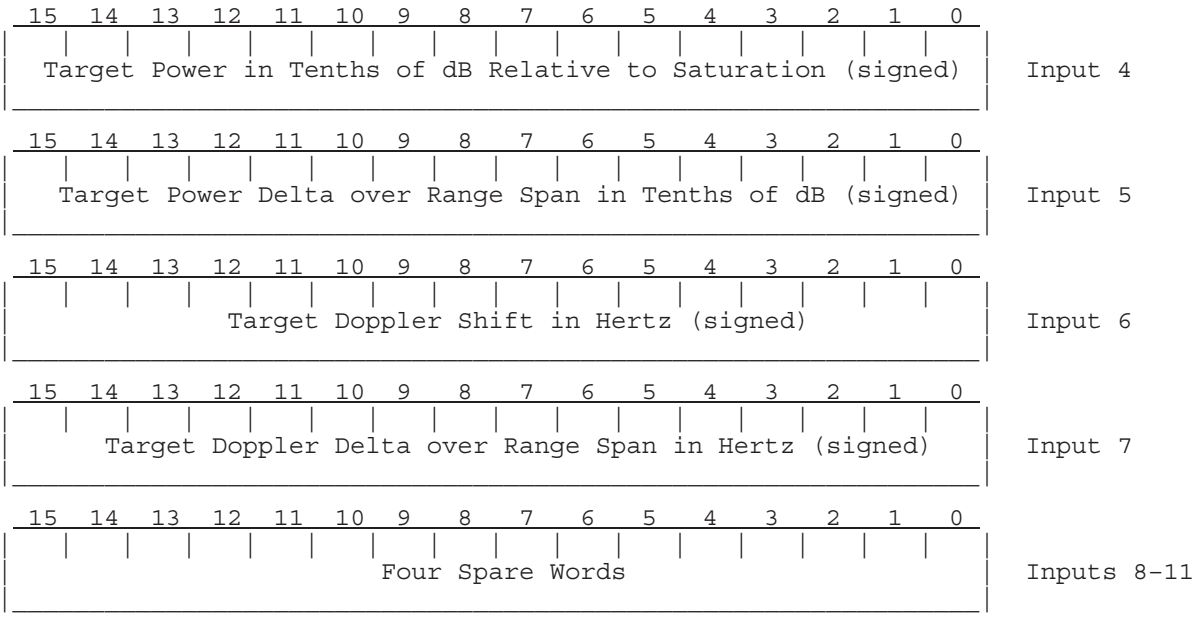


Oper=0 Disable all target simulation activity (no additional args)
Oper=1 Enable simulation of all defined targets (no additional args)
Oper=2 Define a new simulated target (arg list follows)



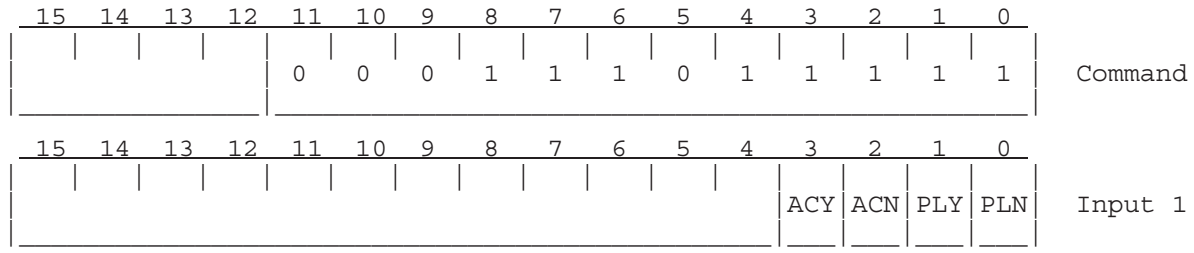
Targ Targ Which target is being defined
Co/Cx Place simulated target in Co-Pol and/or Cross-Pol Rx channels





7.33 Set Burst Pulse Processing Options (BPOPTS)

Some burst pulse processing options can be set by this command.

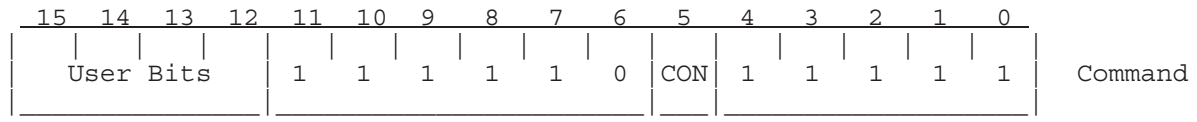


PLY/N — These bits affect whether the RVP8 will phase lock its (I,Q) data to the measured burst pulse. The "PLY" and "PLN" bits force "Yes" and "No" responses. If both bits are clear or both bits are set, then no change will be made.

ACY/N — These bits affect whether the RVP8 applies pulse-to-pulse amplitude correction to its (I,Q) data. The Yes/No bits behave the same as PLY/PLN.

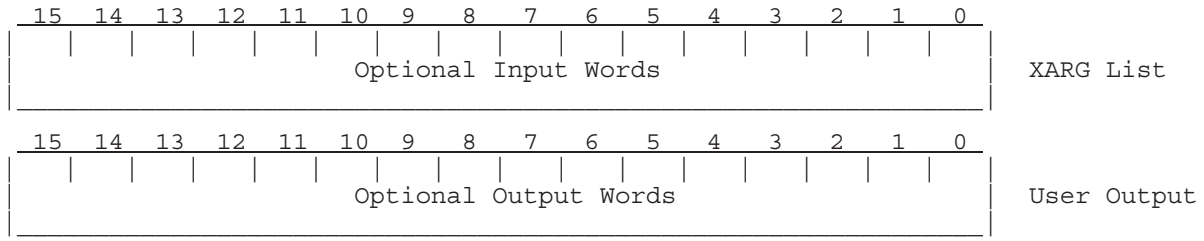
7.34 Custom User Opcode (USRINTR and USRCONT)

These opcodes are part of the open software extensions to the RVP8, which allow custom opcodes to be defined for each major mode of operation. Arguments may be passed into a custom opcode handler as an XARG list. Likewise, an optional array of words returned from that handler will appear after the command executes.



UserBits — Four additional bits defined by the user to help subdivide the opcode functions if desired.

CON — If set, then the RVP8's IQ data acquisition thread proceeds continuously while the opcode is executed. If clear, then the IQ stream is interrupted prior to handling the call.



APPENDIX A
SERIAL STATUS FORMATS

The RVP8 can optionally generate this “internal” BITE packet. Most of these bits are copies of data available from the GPARM command. Those bits are labelled with “GP” followed by the word number and bit number. See section 7.9 Get Processor Parameters (GPARM) on page 309 for more details. The identification byte is selectable, so that conflicts with other BITE packets can be avoided. The 64 auxiliary status variables, labeled S[0:63], may optionally be assigned to electrical input pins on an I/O-62 card using the softplane.conf file.

Table 21 Internal BITE Packet (RVP8 to Host)

Char	Function
1	SYNC Byte (C0 Hex)
2	Identification byte (User Choice)
3	Diagnostic Results 0–6 D6 = GP11D6 = Error loading config/setup files D5 = GP11,D5 = IO62 card #2 failure D4 = GP11,D4 = IO62 card #1 failure D3 = GP11,D3 = Tx card #2 failure D2 = GP11,D2 = Tx card #1 failure D1 = GP11,D1 = Rx card #2 failure D0 = GP11,D0 = Rx card #1 failure
4	Diagnostic Results 7–13 D6 = GP11,D13 = <spare> D5 = GP11,D12 = <spare> D4 = GP11,D11 = RVP8 running without root privileges D3 = GP11,D10 = Signals raised during startup D2 = GP11,D9 = Error in softplane configuration D1 = GP11,D8 = Problem forking compute process D0 = GP11,D7 = Error attaching to antenna library

Table 21 Internal BITE Packet (RVP8 to Host) (Continued)

Char	Function
5	Diagnostic Results 14–20 D6 = GP12,D4 = <spare> D5 = GP12,D3 = <spare> D4 = GP12,D2 = <spare> D3 = GP12,D1 = <spare> D2 = GP12,D0 = <spare> D1 = GP11,D15 = <spare> D0 = GP11,D14 = <spare>
6	Diagnostic Results 21–27D6 = GP12,D11 = <spare> D5 = GP12,D10 = <spare> D4 = GP12,D9 = <spare> D3 = GP12,D8 = <spare> D2 = GP12,D7 = <spare> D1 = GP12,D6 = <spare> D0 = GP12,D5 = <spare>
7	Shutdown Conditions 28–34 D6 = <spare> D5 = <spare> D4 = <spare> D3 = GP12,D15 = <spare> D2 = GP12,D14 = <spare> D1 = GP12,D13 = <spare> D0 = GP12,D12 = <spare>
8	Immediate Status 0–7 D6 = GP10,D6 = Angle sync not interruptible D5 = GP10,D5 = Angle sync enabled D4 = GP10,D4 = Angle sync on elevation D3 = GP10,D3 = Angle sync is BCD D2 = GP10,D2 = PWINFO command is disabled D1 = GP10,D1 = Error loading trigger angle table D0 = GP10,D0 = No trigger

9	Immediate Status 8–14
	D6 = GP10,D13 = # compute processes –1 (bit 1)
	D5 = GP10,D12 = # compute processes –1 (bit 0)
	D4 = GP10,D11 = Current unfolding mode (bit 1)
	D3 = GP10,D10 = Current unfolding mode (bit 0)
	D2 = GP10,D9 = DSP supports 16-bit floating time series
	D1 = GP10,D8 = DSP has full IAGC hardware support
	D0 = GP10,D7 = Angle sync is dynamic

10	Immediate Status 15–21D6 = GP18, D4 = IFD uplink cable failure D5 = GP18,D3 = DSP supports DPRT–1 algorithms D4 = GP18,D2 = <spare> D3 = GP18,D1 = DSP supports random phase algorithms D2 = GP18,D0 = DSP supports FFT algorithms D1 = GP10,D15 = <spare> D0 = GP10,D14 = DSP supports power spectrum output

Table 21 Internal BITE Packet (RVP8 to Host) (Continued)

Char	Function
11	Immediate Status 22–28 D6 = GP18,D11 = IFD test switches are not in normal position D5 = GP18,D10 = AFC status (bit 2) D4 = GP18,D9 = AFC status (bit 1) D3 = GP18,D8 = AFC status (bit 0) D2 = GP18,D7 = IFD PLL is not locked to external reference D1 = GP18,D6 = <spare> D0 = GP18,D5 = IFD downlink cable failure
12	Immediate Status 29–35 D6 = GP55,D2 = Burst pulse hunting is enabled D5 = GP55,D1 = Burst pulse frequency changes can be made D4 = GP55,D0 = Burst pulse timing changes can be made D3 = GP18,D15 = Burst at incorrect range D2 = GP18,D14 = <spare) D1 = GP18,D13 = Missing signal at IFD #1 burst D0 = GP18,D12 = Trigger blanking is enabled
13	Immediate Status 36–42 D6 = GP55,D9 = User-defined Major mode #2 supported D5 = GP55,D8 = User-defined Major mode #1 supported D4 = GP55,D7 = Problem with digital transmitter clock D3 = GP55,D6 = Count not generate the requested phases D2 = GP55,D5 = DSP supports DPRT-2 algorithms D1 = GP55,D4 = Last burst pulse hunt was unsuccessful D0 = GP55,D3 = Burst pulse hunt is running now
14	Immediate Status 43–49 D6 = GP59,D0 = Power spectra size matches sample size D5 = GP55,D15 = <spare> D4 = GP55,D14 = <spare> D3 = GP55,D13 = <spare> D2 = GP55,D12 = <spare> D1 = GP55,D11 = User-defined Major mode #4 supported D0 = GP55,D10 = User-defined Major mode #3 supported
15	Immediate Status 50–56D6 = GP59, D7 = WSR88D Batch mode is supportedD5 = GP59, D6 = Time series data source is external to RVP8D4 = GP59, D5 = Trigger sequence truncatedD3 = GP59, D4 = Using High-SNR packed (I,Q) formatD2 = GP59, D3 = PRT altered to fit trigger patternD1 = GP59, D2 = Trigger pattern altered to fit PRTD0 = GP59, D1 = PROC spectra size matches sample size
16	Immediate Status 57–63 D6 = GP59,D14 = <spare> D5 = GP59,D13 = <spare> D4 = GP59,D12 = <spare> D3 = GP59,D11 = <spare> D2 = GP59,D10 = Receiver protection fault D1 = GP59,D9 = GP outputs #7&8 use Hi-SNR format D0 = GP59,D8 = Major mode refused to use external trigger

Table 21 Internal BITE Packet (RVP8 to Host) (Continued)

Char	Function
17	Immediate Status 64–70 D6 = D5 = D4 = D3 = D2 = D1 = D0 = GP59,D15 = <spare>
18	Latched Status 0–6 D6 = GP9,D6 = Command received while FIFO full D5 = GP9,D5 = FIFO overflow during last PROC command D4 = GP9,D4 = <spare>D3 = GP9, D3 = PRT varied by more than 10 microseconds D2 = GP9,D2 = No trigger during PROC command D1 = GP9,D1 = Trigger too fast during noise measurement D0 = GP9,D0 = No trigger during noise measurement
19	Latched Status 7–14 D6 = GP9,D13 = <spare> D5 = GP9,D12 = <spare> D4 = GP9,D11 = Measured phase sequence is invalid D3 = GP9,D10 = Error in LSIMUL command protocol D2 = GP9,D9 = Error in last LRMSK command D1 = GP9,D8 = <spare> D0 = GP9,D7 = Error detected during last SNOISE command
20	Latched Status 15–21 D6 = <spare> D5 = <spare> D4 = <spare> D3 = <spare> D2 = <spare> D1 = GP9,D15 = Invalid processor configuration D0 = GP9,D14 = <spare>
21	SOPRMS Status 0–6 D6 = GP31,D6 = <spare> D5 = GP31,D5 = 3x3 filtering enabled D4 = GP31,D4 = <spare> D3 = GP31,D3 = <spare> D2 = GP31,D2 = Reflectivity speckle remover on D1 = GP31,D1 = Doppler speckle remover on D0 = GP31,D0 = Reflectivity is range normalized, else SNR
22	SOPRMS Status 7–13 D6 = GP31,D13 = Polarization bit 1: 0=Horiz, 1=Vert D5 = GP31,D12 = Polarization bit 0: 2=Alternating, 3=Dual D4 = GP31,D11 = Disables header output D3 = GP31,D10 = Use any spectrum size D2 = GP31,D9 = Output is in 16-bit format D1 = GP31,D8 = Enable clutter microsupression D0 = GP31,D7 = Use 3-lag processing for widths

Table 21 Internal BITE Packet (RVP8 to Host) (Continued)

Char	Function
23	SOPRMS Status 14–21 D6 = <spare> D5 = <spare> D4 = <spare> D3 = <spare> D2 = <spare> D1 = GP31,D15 = <spare> D0 = GP31,D14 = <spare>
24	Status Bits 6 5 4 3 2 1 0
25	Status Bits 13 12 11 10 9 8 7
26	Status Bits 20 19 18 17 16 15 14
27	Status Bits 27 26 25 24 23 22 21
28	Status Bits 34 33 32 31 30 29 28
29	Status Bits 41 40 39 38 37 36 35
30	Status Bits 48 47 46 45 44 43 42
31	Status Bits 55 54 53 52 51 50 49
32	Status Bits 62 61 60 59 58 57 56
33	Status Bits 63
34-43	<spare>
44	END OF MESSAGE (FF Hex)

The RVP8 can optionally generate this “internal” QBITE packet. These values are copies of data available from the GPARM command. Regular GPARM values are labelled with “GP” followed by the word number. Those in the dspExParmIO structure are labelled with “EX” followed by the word number. See section 7.9 Get Processor Parameters (GPARM) on page 309 for more details.

Table 22 Internal QBITE Packet (RVP8 to Host)

Char	Function
1	SYNC Byte (AF Hex)
2	Identification byte (User Choice)
3–4	Burst pulse frequency, IFD #1
5–6	Burst pulse frequency, IFD #2
7–8	Burst pulse power, IFD #1
9–10	Burst pulse power, IFD #2
11–12	Noise level, IFD #1
13–14	Noise level, IFD #2
15–16	Chassis temperature, IFD#1
17–18	Chassis temperature, IFD#2
19–20	FPGA temperature, IFD#1
21–22	FPGA temperature, IFD#2
23–24	AFC setting
25–26	Burst timing slew
27–28	Current PRF
29–30	Current pulse width
31–62	<spare>
63	END OF MESSAGE (FF Hex)

APPENDIX B

OPTIONAL DUAL POLARIZATION- ZDR, PHIDP, KDP, LDR, ...

B.1 Overview of Dual Polarization

Polarization measurements can provide additional information that can be used to determine more accurate measurements of rainfall or, in some cases, infer particle type such as hail or graupel. The fundamental basis for polarization is that raindrops, particularly larger ones, are not spherical — they are oblate (flattened) such that the horizontal axis is longer than the vertical axis. This means that raindrops will respond differently, for example, to vertical and horizontal polarization of the electric field vector. Because of this, and for technical reasons, most polarization radars use horizontal and vertical polarization. For a review of polarization techniques and variables, refer to Doviak and Zrnic (1993) Section 8.5.

Fundamentally a polarization radar measures amplitude and phase in the same manner as a conventional radar. The new information is that the amplitude and phase can be measured at more than one polarization. The differences in amplitudes and phases measured at different polarizations contain information on the presence or absence of non-spherical scatterers such as large flattened drops. For convenience, some of the basic polarization variables are described below:

- **ZDR: Differential Reflectivity**

  In the case of amplitude (power) measurements, the larger horizontal axis of drops causes the power measured at horizontal polarization (of the electric field) to be larger than the power measured at vertical polarization. The ratio of the reflectivity factors ZH/ZV expressed in dB is given the name ZDR or differential reflectivity. It is generally positive in rain (that is, >1) and is usually less than about 5 dB. When the rainfall rate is large, there are typically more large drops so that ZDR is larger. Low ZDR and high dBZ indicates the presence of hail

which is perhaps tumbling with no preferred orientation. ZDR, because it is a ratio of powers, is not sensitive to the radar calibration as long as the overall gain of the H and V channels is the same (or calibrated).

- **PhiDP and KDP: Differential Phase and Specific Differential Phase**

  In the case of phase measurement, the speed of propagation is also affected by the asymmetry of the larger drops. Because of the longer dimension of the horizontal axis of drops, the medium is effectively more dense for horizontal than for vertical polarization so that the speed of light is reduced for horizontal polarization. This causes the horizontal wavelength to be slightly compressed (more phase cycles per unit distance) in comparison with the vertical wavelength which leads to a phase difference between horizontal and vertical. The phase difference FH-FV is called FDP differential phase shift. FDP increases with range since the phase shifts faster (more frequency cycles per unit distance) for the compressed horizontal microwaves as compared to the faster vertical microwaves. The range derivative of the differential phase, that is, the change of phase per unit distance, is called KDP or the specific differential phase. KDP is almost directly proportional to the rainfall rate so that it has the potential for improving precipitation rate measurements as compared to traditional Z–R relationship measurements which can be highly inaccurate.

- **LDR: Linear Depolarization Ratio**

  Some advanced polarization radars can transmit at one polarization and receive simultaneously in two channels, usually the co–polarized and cross–polarized components. For example, when transmitting horizontal, both horizontal (co–polarized) and vertical (cross–polarized) are received by two separate channels. In the case of vertical or horizontal, the ratio of the power Zcross / Zco is called the linear depolarization ratio or LDR. The amount of incident radiation that is depolarized by a particle depends on the particle shape and orientation (for example, canting angle with respect to horizontal). Perfectly spherical particles do not depolarize either horizontal or vertical polarization so that LDR is zero. Particles that are wet, tumbling and irregularly shaped will give larger LDR values. Therefore, LDR values in rain tend to be small, for example, less than -25dB. Larger values of LDR can occur in the bright band or in the presence of hail.

  A radar and antenna system must be optimized to measure LDR by assuring that the antenna, feed and supporting struts and radome are not themselves depolarizing the transmitted and received radiation. This is called "cross–pol isolation". The integrated cross–pol isolation of the antenna pattern must be better than about 30 dB for LDR measurement since -20 dB is a large LDR.

- **[RHOHV, PHIDP] [RHOH, PHIH] [RHOV, PHIV]: Correlation Variables**

  There are several correlation functions that can be calculated depending on the capabilities of the radar. These are generally complex having both an amplitude and phase. These are all normalized so that a perfect correlation magnitude is 1 and perfectly decorrelated is 0.

  RHOHV and PHIDP are the magnitude and phase of the correlation between the horizontal and vertical co-polarized channels. These are available on H/V switching systems or on systems that transmit simultaneous H and V. As discussed in a preceding paragraph, PHIDP can be used to infer precipitation rate. RHOHV in rain is typically very close to 1 (0.98). RHOHV values can be reduced in the case of irregularly shaped, randomly oriented, wet tumbling particles. Thus RHOHV has information on the particle type.

  RHOH and PHIH are the magnitude and phase of the correlation between the co-polarized and cross-polar channels for H transmission and simultaneous H and V reception. RHOV and PHIV denote the cross–channel correlation magnitude and phase for vertical transmission. These are available on dual-channel receiver with transmit either fixed or alternating. The information content of the cross-pol correlations is the topic of current research.

B.2 Radar System Considerations

A polarization radar is characterized by how it transmits and how it receives. For simplicity we will assume that the radar uses horizontal and/or vertical polarization. However, other polarization pairs could be used (for example, right and left circular polarization).

Transmit Modes

- **Fixed** (horizontal or vertical) - this can be controlled by a switch or the radar can be simply fixed to transmit a single polarization. If a switch is used, it can be a simple slow waveguide switch rather than a fast switch (pulse-to-pulse).
- **Alternating** (horizontal and vertical) - in this case the radar alternates pulse-to-pulse between horizontal and vertical. A high-power fast switch is used to switch the polarization between the two channels.
- **Simultaneous** (horizontal and vertical) - horizontal and vertical are transmitted simultaneously.

Receive Modes

- **Single-channel receiver** used only for alternating transmission. The receiver typically receives the co–polarized radiation (transmit H and receive H then transmit V and receive V).
- **Dual-channel receiver** receives two channels (H and V) simultaneously.

The table below summarizes the various transmit and receive cases and the polarization variables that are available for each. Note that standard parameters are available for all cases (dBT, dBZ, V and W). The RVP8 supports all of these cases.

Table 23 Transmitter Types

	Transmitter Type			
Receiver Type	**Fixed H**	**Fixed V**	**Alternating H&V**	**Simultaneous H+V**
Single-Channel	ConventionalRadar	Conventional Radar	ZDR RHOHV PHIDP and KDP	Not applicable
Dual-Channel	LDRH RHOH PHIH	LDRV RHOV PHIV	LDRH LDRV RHOH RHOV PHIH PHIV ZDR RHOHV PHIDP and KDP	ZDR RHOHV PHIDP and KDP *(STAR mode)*

The fixed single channel cases are conventional radars rather than polarization radars. The case of simultaneous H+V transmission and a single radar does not make physical sense. The other cases provide various polarization measurements. The fixed dual-channel cases allow the Optional Dual Polarization cross-polarization LDR and the co-pol/cross-pol correlation amplitude and phase to be measured (for example, RHOH and PHIH). The simultaneous H+V transmission and dual-channel reception is sometimes called the STAR mode (simultaneous transmit and receive). This allows the co–pol measurements to be made (ZDR, RHOHV, PHIDP and KDP). The alternating transmission dual-channel receiver allows both the co–pol and the cross-pol measurements to be made, that is, it is the most complete.

Summary of Radar System Characteristics

The RVP8 supports all of these modes, but most polarization radar systems do not. As mentioned before, the measurement of cross-pol parameters such as LDR (fixed or alternating transmission and dual-channel reception) requires a radar system that has been optimized for cross–pol isolation, for example, an offset feed antenna and no radome. By removing the feed, support struts and radome from the path of the radiation, the cross–pol isolation can be improved.

The single-channel alternating method has been used in several polarization radars for ZDR measurement. The advantage of this approach is that it is relatively easy to modify a conventional radar by simply adding a dual port feed and a high-power fast switch above the antenna rotary joints. The disadvantage is that the switch is costly and will eventually fail.

For these reasons, the STAR mode has come into recent use. No switch is required and the components are fairly reliable. The disadvantage of the approach (as it is usually implemented) is that a dual rotary joint and dual waveguides are required to duct both the H and the V through the antenna pedestal up to the antenna feed. In spite of this, the STAR mode offers perhaps the best approach for upgrading an existing radar or for factory installation on a new radar of conventional design.

B.3 RVP8 Dual-Channel Receiver Approach

Dual IFD and Rx Card

For dual-dual channel receivers, the RVP8 uses 2 IFD's and 2 RVP8/Rx cards. An example configuration is described and illustrated in 2.1 System Configuration Concepts on page 20.

Reference Clock to IFD

It is critical that both IFD's be phase-locked to a common reference clock. This clock, or a derivative frequency of the clock such as a COHO frequency, is input into each IFD to provide an absolute phase reference.

The RVP8 IFD phase-locks its sampling crystal to the reference clock input. Trigger generation by the RVP8 will also be phase locked to the reference clock. The reference clock must be in the range 2 to 60 MHz at 0 to -10 dBm and stable in phase to 10–7. The RVP8 IFD must be specially configured with a locking crystal to enable this feature. Vaisala will either factory install the modification or assist the customer in performing the modification and supply the necessary components. See 3.2.12 IFD Reference Clock Input (Optional) on page 82 for more information on installing and configuring to use an external reference clock.

Vaisala Antenna Mounted Receiver (AMR)

The AMR converts a single polarization to a dual polarization system. It can be used on new radars or field upgrades of older systems. The AMR supports both STAR mode and LDR mode for H-only transmit. In the AMR, the RVP8 and other receiver components are packaged in a box that is mounted above the elevation axis of the antenna. Communication to/from the box is accomplished using a wireless LAN. For more information on the AMR, see the AMR Technical Description at www.vaisala.com.

B.4 Overview of Processing Algorithms

The RVP8 supports four polarization modes summarized in the table below. For each case, the standard moments (T, Z, V and W) are calculated as well. The notation for the outputs used here is similar to that in standard usage (for example, Doviak and Zrnic). However, for LDR we use the notation LDRH to indicate that this is the LDR for horizontal transmission. The notation RHOH and PHIH is used to indicate the magnitude and phase of the covariance between the co- and cross–polarized channels for H transmit.

Table 24 Supported Polarization Modes and Outputs

Case	Transmit	Receive	Processing Mode	Polarization Outputs
1	Fixed Horizontal or Fixed Vertical	Dual-Channel	PPP only	LDRH RHOH PHIH or LDRV RHOV PHIV
2	Simultaneous H+V (STAR Mode)	Dual-Channel	PPP or ZDR for FFT, Random Phase and DPRT1&2	ZDR PHIDP KDP RHOHV
3	Alternating H/V	Single-Channel	PPP only	ZDR PHIDP KDP RHOHV
4	Alternating H/V	Dual-Channel	PPP only	LDRH RHOH PHIH LDRV RHOV PHIV ZDR PHIDP KDP RHOHV

Input Receiver Sample Notation

For the discussion of polarization, we will adopt the notation used by Doviak and Zrnic. The received signal for pulse n from a single range bin shall be denoted as:

s^{n}_{hh}	Receive h: Transmit h	Horizontal co-polar signal
s^{n}_{vh}	Receive v: Transmit h	Horizontal cross–polar signal
s^{n}_{vv}	Receive v: Transmit v	Vertical co–polar signal
s^{n}_{hv}	Receive h: Transmit v	Vertical cross-polar signal

The pulse index is now indicated by the superscript as opposed to the subscript. The first subscript indicates the received polarization while the second subscript indicates the transmit polarization. If the transmit is the same as the received polarization, then this is called the co–polarized signal. If the transmit and receive are different then this is called the cross–polarized signal.

These variables are complex and are the same is the "s_n" notation used earlier, for example we can write:

$$S_{hh}^{n} = I_{hh}^{n} + jQ_{hh}^{n}$$

to show the relationship to the received I and Q values. Either filtered and unfiltered versions of the samples can be selected for processing. However, for convenience we will drop the snotation for filtered samples.

Notation and Model for Correlations

The pulse pair processing mode is used for all of the polarization calculations, except that ZDR-only processing for the STAR case can be done in either FFT or random phase as well as pulse pair. As with the standard moments, the autocorrelations form the basis for the processing of the polarization variables.

The autocorrelations are computed in a manner identical to the standard moments, for example, in pulse pair mode, the autocorrelations for the horizontal transmit co-polar channel are:

$$T_{ohh} = \frac{1}{M}\sum_{n=1}^{M} S_{hh}^{n} * S_{hh}^{n}$$

$$R_{ohh} = \frac{1}{M}\sum_{n=1}^{M} s'^{n}_{hh} * s'^{n}_{hh}$$

$$R_{1hh} = \frac{1}{M-1}\sum_{n=1}^{M-1} s'^{n}_{hh} * s'^{n+1}_{hh}$$

$$R_{2hh} = \frac{1}{M-2}\sum_{n=1}^{M-2} s'^{n}_{hh} * s'^{n+2}_{hh}$$

What is different is that for polarization systems, this processing can be applied in up to four separate channels (s_{hh}, s_{vh}, s_{vv} and s_{hv}). The physical

model for the channel powers is identical to the model used for the standard moment cases, that is,

Co-Channel Power **Cross-Channel Power**

$$R_0^{hh} = g_h^r g_h^t S_{hh} + N_h \qquad R_0^{vh} = g_v^r g_h^t S_{vh} + N_v$$

$$R_0^{vv} = g_v^r g_v^t S_{vv} + N_v \qquad R_0^{hv} = g_h^r g_v^t S_{hv} + N_h$$

Here S is used to denote the actual backscatter average power to the radar which, when multiplied by the appropriate transmitter and receiver gains, yields the actual measured power. Sometimes in comparing powers in two channels (for example, ZDR and LDR) we will need to know the relative gains of the two channels. However, in many calculations, the relative gains cancel-out and in these cases the algorithms are implemented assuming all the gains are equal to 1.

In the algorithm descriptions, we will often use the notation common in the literature that (for example):

$$R_{ohh} = \frac{1}{M}\sum_{n=1}^{M} s'^{n}_{hh} * s'^{n}_{hh} = < \left| s'_{hh} \right|^2 >$$

Noise Bias in Channel Powers and Correlations and Optional Correction

The average noise powers Nv and Nh are assumed to be receiver noise only. These bias the autocorrelations at lag zero, that is, the channel power measurements. Autocorrelations at lags 1 and 2 are not biased by noise, while cross channel correlations are biased by finite noise. Cross-channel phases are smeared by finite noise, while unbiased, assuming that the noises in the two channels are independent (a good assumption).

The channel noise values are measured directly by the RVP8 during noise sampling. The use of these measurements for correcting for the effects of finite noise is configured in the TTY setups. The choices are made with the mp Nonvolatile setup settings "Polarimetric Power Params – NoiseCorrected:YES/NO" and "Polarimetric Correlations – NoiseCorrected:YES/NO"

If "Polarimetric Power Params – NoiseCorrected" is enabled, the ZDR and LDR are computed from channel powers, subtracted for noise levels. The sensitivity is enhanced to observe weather effects in weak echoes (SNR << 10 dB). At the SNR low limit, ZDR approach 0 dB, as soon as noise levels

are properly set. With no noise correction, ZDR values in weak signal regions are biased, and approach to the ratio of channel noise levels. A finite noise level may set a limit for LDR observations.

If "Polarimetric Power Correlations – NoiseCorrected" is left disabled, RHOHV, RHOH, and RHOV approach zero in weak echoes, while setting "YES" makes them approach unity. Phase measurements PHIDP, KDP, PHIH, PHIV are not affected by these settings.

Clutter Filtering

The polarization variables are computed from data filtered for clutter. The settings are the same, and user configurable, as for standard moments. The filter is applied identically for each polarimetric channel.

B.5 Case 1: Fixed Transmit: Dual-Channel Receiver

Input Receiver Samples

In fixed mode the radar is configured (either permanently or by means of a switch) to transmit either vertical or horizontal polarization with dual-channel reception of both the co- and cross–channel polarizations, for example, transmit horizontal and receive both horizontal (co) and vertical (cross) polarizations.

The received samples in the two transmit cases are

Transmit Horizontal

$$\left[S_{hh}^{1}:S_{vh}^{1}\right]\left[S_{hh}^{2}:S_{vh}^{2}\right]\left[S_{hh}^{3}:S_{vh}^{3}\right]\ldots\left[S_{hh}^{M}:S_{vh}^{M}\right]$$

or

Transmit Vertical

$$\left[S_{vv}^{1}:S_{hv}^{1}\right]\left[S_{vv}^{2}:S_{hv}^{2}\right]\left[S_{vv}^{3}:S_{hv}^{3}\right]\ldots\left[S_{vv}^{M}:S_{hv}^{M}\right]$$

Calculation of the Polarization Measurands

The processing in this mode is done by pulse pair algorithm. The user may select a clutter filter.

The polarization measurands for the two transmit cases are as follows:

Transmit Horizontal	**or**	**Transmit Vertical**
$LDRH = 10LOG\left[\frac{S_{vh}}{S_{hh}}\right]$	or	$LDRV = 10LOG\left[\frac{S_{hv}}{S_{vv}}\right]$
$= 10LOG\left[\frac{<\lvert S_{vh}\rvert^2>-N_v}{<\lvert S_{hh}\rvert^2>-N_h}\right]-XDR$	or	$= 10LOG\left[\frac{<\lvert S_{hv}\rvert^2>-N_h}{<\lvert S_{vv}\rvert^2>-N_v}\right]+XDR$
$RHOH = \lvert\rho_h\rvert$	or	$RHOV = \lvert\rho_v\rvert$
$PHIH = \arg\lvert\rho_h\rvert$	or	$PHIV = \arg\lvert\rho_v\rvert$

Here, the H and V average channel powers are computed as follows with optional noise correction, for example,

Co- $g_h^r g_h^t S_{hh} = <\lvert S_{hh}\rvert^2> - N_h$ or $g_v^r g_v^t S_{vv} = <\lvert S_{vv}\rvert^2> - N_v$

Cross- $g_v^r g_h^t S_{vh} = <\lvert S_{vh}\rvert^2> - N_h$ or $g_h^r g_v^t S_{hv} = <\lvert S_{hv}\rvert^2> - N_h$

The complex covariance ρ (used above) is:

for H transmit $\rho_h = \frac{< S_{vh} S_{hh}^* >}{\sqrt{S_{vh} S_{hh}}}$ or for V transmit $\rho_h = \frac{< S_{hv} S_{vv}^* >}{\sqrt{S_{hv} S_{vv}}}$

Fortunately, the algorithms do not require us to know all of the individual gain terms. They cancel in the calculation of r so are taken as =1 in the implementation. However, the differential receiver gain LDR Offset must be known from calibration to calculate
LDR:

$dB\ Value\ is\ XDR = 10LOG\ xdr$ *where the linear value is* $xdr\ \frac{g_v^r}{g_h^r}$

B.6 Case 2: Simultaneous Dual Transmit and Receive (STAR)

Input Receiver Samples

In this mode there is simultaneous transmit and receive of both vertical and horizontal polarization. For each pulse there is a measurement of the amplitude in each channel, for example,

$$\left[S_{hh}^{1}:S_{vv}^{1}\right]\left[S_{hh}^{2}:S_{vv}^{2}\right]\left[S_{hh}^{3}:S_{vv}^{3}\right]\ldots\left[S_{hh}^{M}:S_{vv}^{M}\right]$$

We will assume that M samples are collected for processing, that is, Note that even though there is cross–polarized radiation received in each channel, this cross–polar contribution can be neglected since the co-polarized received signal is much stronger.

Calculation of the Polarization Measurands

The processing in this case is done by pulse pair mode. However both FFT and random phase processing can be performed if only ZDR and standard moments are requested for ouput. In any mode, the user may select a clutter filter.

The RVP8 calculates the following polarization parameters:

$$ZDR = 10LOG\left[\frac{S_{hh}}{S_{vv}}\right]$$

$$ZDR = 10LOG\left[\frac{<|S_{hh}|-N_h}{<|S_{vv}|-N_v}\right]+GDR$$

$$RHOHV = |\rho_{hv}(0)|$$

$$PHIDP = \arg[\rho_{hv}(0)]$$

KDP based on least squares fit to PHIDP (see B.9 KDP Calculation on page 377) where the following definitions are used:

$$g_h^r g_h^t S_{hh} = <|S_{hh}|^2> - N_h \qquad g_v^r g_v^t S_{vv} = <|S_{vv}|^2> - N_v$$

The noise powers the two channels are denoted as Nh and Nv . The noise corrections to Shh and Svv are optionally configured in the TTY setups. The ZDR Offset is the total (transmit and receive) differential channel gain. It must be calibrated for the system.

$$dB\ Value\ is\ \ ZDR\ Offset\ =\ 10LOG\ offset\ where\ the\ linear\ value\ is\ \ offset\ =\ \frac{g_v^r g_v^t}{g_h^r g_h^t}$$

The correlation function is computed from:

$$\rho_{hv}(0)\ =\ \frac{< S_{vv} S_{hh}^{*} >}{\sqrt{S_{hh} S_{vv}}}$$

The gain terms cancel in the calculation of ρ so in the implementation they are simply assumed to be =1.

B.7 Case 3: Alternating H/V Transmit: Single Receiver

Input Receiver Samples

This is the traditional ZDR radar with a high-power fast switch that alternates between horizontal and vertical on each pulse. The switch is made just prior to the transmit pulse so that the transmitter radiates and then receives at a single polarization for each pulse. Thus the samples are:

$$S_{hh}^1 \quad S_{vv}^2 \quad S_{hh}^3 \quad \ldots \quad S_{vv}^{M+1}$$

For the discussion below we will assume that there are M+1 total samples with M/2 horizontal pulses indexed by (1, 2, 3...M–1) and M/2+1 vertical pulses indexed at (2, 4, 6, ...M). Note that the processor does not assume that the first pulse in a sequence is horizontal.

Calculation of the Polarization Measurands

The processing is done in pulse pair with optional clutter filter.

The RVP8 calculates the following:

$$ZDR\ =\ 10LOG\left[\frac{S_{hh}}{S_{vv}}\right]$$

$$ZDR = 10LOG\left[\frac{<|S_{hh}|^2> - N_h}{<|S_{vv}|^2> - N_v}\right] + GDR$$

$$PHIDP = \frac{1}{2}\arg\,[R_a R_b^*]$$

$$RHOHV = \frac{|\rho_{hv}(T_s)|}{[\rho_{hv} 2T_s]^{0.25}}$$

KDP based on least squares fit to PHIDP (see B.9 KDP Calculation on page 377). where the following definitions are used:

$$|\rho_{hv}(T_s)| = \frac{|R_a| + |R_b|}{2\sqrt{S_{hh} S_{vv}}}$$

$$\rho(2T_s) = \frac{\sum_{n=1}^{M/2-1} (s^*_{hh}\,[2n-1]s_{hh}[2n+1] + s^*_{vv}[2n]s_{vv}[2n+2])}{(M/2-1)(s_{hh}+s_{vv})}$$

$$R_a = \frac{1}{M/2}\sum_{n=1}^{M/2} s^{2n-1*}_{hh}\; s^{2n}_{vv}$$

$$\text{and } R_b = \frac{1}{M/2}\sum_{n=1}^{M/2} s^{2n*}_{vv}\; s^{2n+1}_{hh}$$

The calculation of the channel powers ($<|s_h|^2>$ and $<|s_{vv}|^2>$) is done using alternating pulses in this case. Note that in the calculation of Rb, the RVP8 uses the extra M+1 sample. The gain terms cancel in the calculation of r so in the implementation they are simply assumed to be =1.

B.8 Case 4: Alternating H/V Transmit: Dual Receiver

Input Receiver Samples

This is the most comprehensive case of polarization operation since it permits calculation of all of the polarization measurands. In this case the transmitter alternates pulse-to-pulse between horizontal and vertical polarization and the dual-channel receiver provides measurement of both the co- and the cross-polarized return, that is,

$$\left[S^1_{hh}:S^1_{vh}\right]\left[S^2_{vv}:S^2_{hv}\right]\left[S^3_{hh}:S^3_{vh}\right]\left[S^3_{hh}:S^3_{hv}\right]\cdots\left[S^{M+1}_{vv}:S^{M+1}_{hv}\right]$$

We will assume that M+1 samples are collected for processing (an extra sample is required for the calculation Rb per B.7 Case 3: Alternating H/V Transmit: Single Receiver on page 374).

Calculation of the Polarization Measurands

The RVP8 calculates the following:

Co-polar channel measurements	
ZDR, PHIDP, RHOHV	Identical to alternating case B.7 Case 3: Alternating H/V Transmit: Single Receiver on page 374.
Cross-polar channel measurements	
LDRH, RHOH, PHIH LDRV, RHOV, PHIV	Identical to fixed case B.5 Case 1: Fixed Transmit: Dual-Channel Receiver on page 371.

The co-polar channel measurements are exactly as they are for the alternating single-receiver case. The cross–polar measurements are calculated using fixed case algorithms except they are calculated for *both* H and V polarizations.

B.9 KDP Calculation

In all modes that compute PHIDP, the signal processor can also be configured to compute KDPthe specific differential phase in units of degrees per km. This is the range derivative of PHIDP. There are two techniques that have been used to obtain this:

- The smoothed range derivative.
- The slope from a least squares fit.

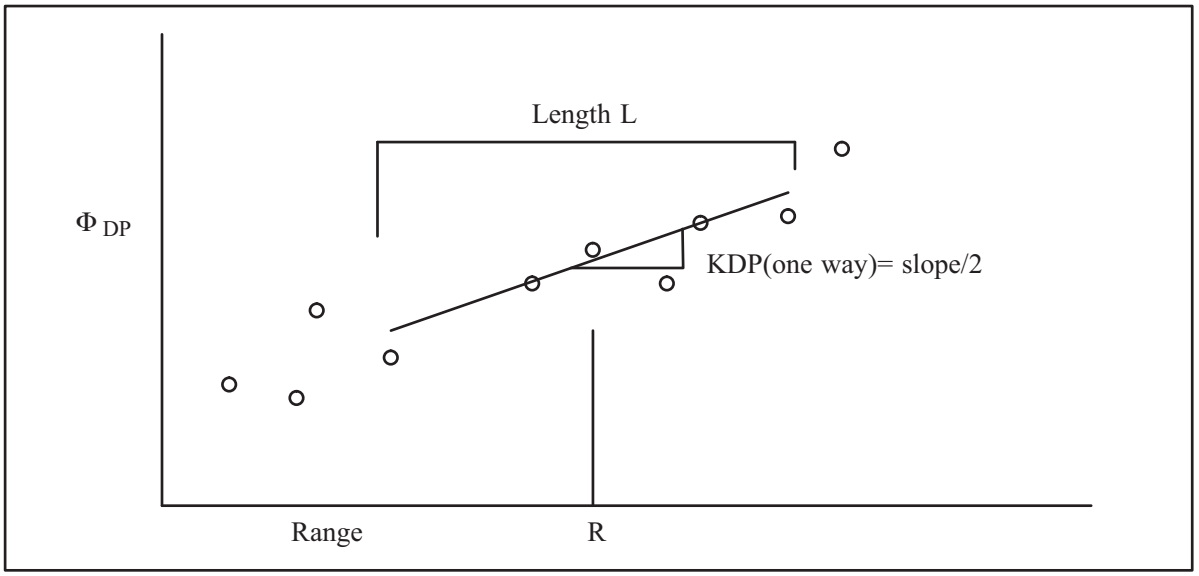


Figure 47 Least Squares KDP calculation

The graph shows the thresholded differential phase vs range. This is the starting point for the algorithm. The length scale L is selectable by the user in the TTY setups (mp section, KDP Length in km, default 5.00 km). The KDP value for a bin at range R is computed from a least squares fit that includes points that are within +-L/2 as indicated in the figure. PHIDP is output by the processor on the unambiguous interval of 0 to 180 degrees. Before fitting, the points are first unfolded to a common interval by starting at the left-most point and then moving right assuming that a difference of more than 1/2 the unambiguous interval is the result of folding. Since it is the slope that is of interest, the absolute interval is not critical, as long as the points are in a common interval.

After fitting, the slope is obtained which corresponds to the 2-way KDP since it is based on the 2-way measurement of PHIDP . To be consistent with most values in the literature, the slope value is divided by 2 so that the final output is the one-way KDP in degrees per km (with a wavelength scaling in the data format).

This procedure is repeated for each bin. Thus if the bin spacing is 250 m, the output bin spacing of KDP will be 250 m. It is required that there be at least 50% of the possible number of bins present in the interval L to calculate a valid KDP, else the KDP is set to the threshold value. Since the input PHIDP values are already thresholded, the only additional threshold on KDP is this 50% rule.

B.10 Standard Moment Calculations (T, Z, V, W)

Overview

Standard moments are available for all four of the polarization cases. Since there can be up to four different channels of time series input, there are several choices for computing the standard moments. For example, in the STAR mode (Case 2), the standard moments can be computed from:

- s_{hh} samples
- s_{vv} samples
- Average of the results from the s_{hh} and s_{vv} samples

The third case is handled by averaging the individual channel correlations, and then using the average correlations in the standard moment processing. The averaging must take into account the differential gain of the channels.

The selection of which method to use is made in setup. There are four questions posed in the mp section:

```
T/Z/V/W computed from: H-Xmt:YES V-Xmt:YES

T/Z/V/W computed from: Co-Rcv:YES Cx-Rcv:NO
```

The first two questions are used to specify that *given a choice* between vertical and horizontal transmit, which transmit polarization to use. Thus for the fixed H or V case where there is only one transmit polarization, this question does not apply. The processor will simply use samples for the polarization is transmitted.

The second two questions are used to specify that *given a choice* between using the co- or cross-polar receivers which one shall be used. This question applies only to systems that can measure LDR, that is, fixed or alternating transmit, dual-channel receiver systems).

The tables in the sections below summarize the standard moment calculations for each of the four modes and how to configure the four TTY setup responses. Note that these are the only supported modes. Some

combinations of responses are unsupported. For example, it is not supported to answer both Co-Rcv: NO and Cx-Rcv: NO.

The top of each table identifies the transmitter/receiver case and what samples are available. The notation HH signifies that the s_{hh} samples are available. The tables use "—" to indicate that either a YES or NO response will cause the same result, that is, the RVP does not care what response is made. In cases where averaging is performed, the type of weighting used is indicated.

Model for standard moment autocorrelations

The model for the moment autocorrelation calculations is as follows (using Ro as an example):

$$R_0^{hh} = g_h^r g_h^t S_{hh} + \bar{N}_h \qquad R_0^{vh} = g_v^r g_h^t S_{vh} + \bar{N}_v$$

$$R_0^{vv} = g_v^r \, g_v^t \, S_{vv} + \bar{N}_v \qquad R_0^{hv} = g_h^r \, g_v^t \, S_{hv} + \bar{N}_h$$

where:

R_0^{hh}, R_0^{vh}, R_0^{vv}, R_0^{hv}	Are the autocorrelations if the samples at lag zero.
S_{hh}, S_{vh}, S_{vv}, S_{hv}	The average power returned from the scatterers.
g^r_h,g^r_v	Receiver gains for horizontal and vertical receive.
g_h^t, g_v^t	Transmitter gains for horizontal and vertical transmit.
N_h, N_v	Measured noise power of the samples.

In other words, the power that is measured in a channel has two components:

- Backscattered power from the targets that is effected by the transmitter and receiver channel gains.
- Receiver noise which is measured by the RVP8 during noise sampling. In the case of R1 and R2 autocorrelations, the model is similar except that there is no noise bias.

In the case of R1 and R2 autocorrelations, the model is similar except that there is no noise bias.

Calibration Parameters

For dBZ calculations, a calibration constant is required, that is, the dBZ_o value in 6.5 Reflectivity Calibration on page 252. Depending on the polarization case and the technique selected for standard moment calculation, it may also be required to have ZDR Offset and LDR Offset, that is,

- ZDR Offset- The ratio of the total gains (transmit/receive) of the two co–receive channels.
- LDR Offset- The ratio of the receiver gains in a dual receiver system. This is not required for the Case 2: STAR or the Case 3: Alternating Single–Channel.

The RVP8 supports a single calibration reflectivity dBZ_o. In all cases it is assumed that the $dBZ_o$is for the horizontal co–receive (HH) channel. The only exception is for fixed vertical polarization, in which the algorithm assumes that the calibration is for the vertical co-receive (VV) channel. LDR Offset and ZDR Offset are also downloaded and used to adjust the dBZ_o as required depending on the user's selection for the standard moments. For example, in STAR mode, if the user selects dBZ to be computed from the VV channel, the dBZ_o for the HH and a ZDR Offset adjustment are used to calculate the dBZ in the VV channel.

The remainder of this section discusses the standard moment calculation options for the each polarization case. For a discussion of how to calibrate LDR Offset and ZDR Offset see B.12 Calibration Considerations on page 390.

Case 1H: Fixed Horizontal Transmit, Dual Channel Receive– (HH, VH)8				
dBZo from HH Channel	**TTY Setup Question Responses**			
Calculate T, Z, V, W from:	**HXmt**	**VXmt**	**CoRcv**	**CxRcv**
HH (co) (Recommended)	—	—	YES	NO
VH (LDR Offset^{-1} weighting)	—	—	NO	YES
HH+VH (xdr^{-1} weighting)	—	—	YES	YES

HH Channel (co–pol)

This is the recommended channel for the case of linear polarization. The reason is that for linear polarization, the co–polar channel will have the strongest signal. Processing is identical to a conventional radar.

VH Channel (cross–pol)

This choice would be used for circular or elliptic transmit polarization. Since the algorithm assumes that dBZo is from the co–polar channel, xdr is used to adjust the autocorrelations as follows:

$$T_0 = xdr^{-1} T_0^{vh}$$

$$R_{(0)} = xdr^{-1} R_0^{vh}$$

$$R_1 = xdr^{-1} R_1^{vh}$$

$$R_2 = xdr^{-1} R_2^{vh}$$

$$N = xdr^{-1} N_v$$

These adjusted autocorrelations are then used as per the standard moment processing for a conventional radar. To illustrate this, consider the example of reflectivity processing. The radar equation can be written as (see 6.3.3 Reflectivity on page 238):

$$Z^{vh} = C S_{vh} r^2 = \left[\frac{C r_0^2 N_v}{g_v^r g_h^t}\right]\left[\frac{r^2}{r_0^2}\right]\left[\frac{T_0^{vh} - N_v}{N_v}\right], \text{ where } T_0^{vh} = g_v^r g_h^t S_{vh} - N_v$$

$$= \left[\frac{C r_0^2 N_h}{g_h^r g_h^t}\right]\left[\frac{r^2}{r_0^2}\right]\left[\frac{g_h^r}{g_v^r}\right]\left[\frac{T_0^{vh} - N_v}{N_h}\right]$$

The third term is simply 1/XDR so that we can write:

$$Z^{vh} = \left[\frac{C r_0^2 N_h}{g_h^r g_h^t}\right]\left[\frac{r^2}{r_0^2}\right]\left[\frac{xdr^{-1} T_0^{vh} - xdr^{-1} N_v}{N_h}\right]$$

In this case, the first term is the dBZ_o for the HH channel. Thus we can use the dBZ_o for the HH channel to calibrate the cross-channel, if we first adjust the cross–channel noise and power by 1/xdr and then normalize by Nh. The reflectivity calculation assumes that the calibrated xdr value compensates for any differences in the radar constant between the two channels, that is, we do not need to have separate radar constants for the two channels.

HH+VH Channels

This choice would be used for elliptic transmit polarizations that give comparable return signal in both the co- and cross-channels. The approach is to obtain average autocorrelation functions as follows:

$$T_0 = \frac{T_0^{hh} + xdr^{-1} T_0^{vh}}{2}$$

$$R_0 = \frac{R_0^{hh} + xdr^{-1} R_0^{vh}}{2}$$

$$R_1 = \frac{R_1^{hh} + xdr^{-1} R_1^{vh}}{2}$$

$$R_2 = \frac{R_2^{hh} + xdr^{-1} R_2^{vh}}{2}$$

$$N = \frac{N_h + xdr^{-1} N_v}{2}$$

These adjusted autocorrelations are then used as per the standard moment processing for calibration with respect to the HH channel.

Case 1V: Fixed Vertical Transmit and Dual Channel Receive– (VV, HV)				
dBZo from VV Channel	**TTY Setup Question Responses**			
Calculate T, Z, V, W from:	HXmt	VXmt	CoRcv	CxRcv
VV (co)	—	—	YES	NO
HV (xdr weighting)	—	—	NO	YES
VV+HV (xdr weighting)	—	—	YES	YES

This is the only case for which the calibration constant dBZ_o for the VV channel should be downloaded to the signal processor.

VV Channel (co–pol)

This is the recommended channel for the case of linear polarization. The reason is that for linear polarization, the co-polar channel will have the strongest signal. Processing is identical to a conventional radar.

HV Channel (cross–pol)

This choice would be used for circular or elliptic transmit polarization when most of the return is in the cross-pol channel. Since the algorithm assumes that dBZo is from the co-polar channel, xdr is used to adjust the autocorrelations as follows:

$$T_0 = xdr\ T_0^{hv}$$

$$R_0 = xdr\ R_0^{hv}$$

$$R_1 = xdr\ R_1^{hv}$$

$$R_2 = xdr\ R_2^{hv}$$

$$N = xdr\ N_h$$

These adjusted autocorrelations are then used as per the standard moment processing with dBZo calibrated with respect to the VV channel.

VV+HV Channels

This choice would be used for elliptic transmit polarizations that give comparable return signal in both the co- and cross-channels. The approach is to obtain average autocorrelation functions as follows:

$$T_0 = \frac{T_0^{vv} + xdr\ T_0^{hv}}{2}$$

$$R_0 = \frac{R_0^{vv} + xdr\ R_0^{hv}}{2}$$

$$R_1 = \frac{R_1^{vv} + xdr\ R_1^{hv}}{2}$$

$$R_2 = \frac{R_2^{vv} + xdr\, R_2^{hv}}{2}$$

$$N = \frac{N_v + xdr\ N_h}{2}$$

These adjusted autocorrelations are then used as input to the standard moment processing algorithms with dBZ_o calibrated with respect to the VV channel.

Case 2: Simultaneous Transmit and Receive– STAR (HH, VV) **Case 3: Alternating Transmit Single–Channel Receive (HH, VV)**				
dBZo from HH Channel	**TTY Setup Question Responses**			
Calculate T, Z, V, W from:	H–Xmt	V–Xmt	Co–Rcv	Cx–Rcv
HH	YES	NO	—	—
VV (ZDR Offset^{-1}weighting)	NO	YES	—	—
HH+VV (gdr^{-1}weighting)	YES	YES	—	—

A fundamental difference between these two cases is that for all standard moment processing choices, the STAR case has double the number of samples as compared to the single-channel alternating case. However, the processing is otherwise identical.

HH Channel

Since the HH channel is directly calibrated this is the recommended choice. Processing is identical to a conventional radar.

VV Channel

In this case, GDR is used to adjust the autocorrelations as follows:

$$T_0 = gdr^{-1} T_0^{vv}$$

$$R_0 = gdr^{-1} R_0^{vv}$$

$$R_1 = gdr^{-1} R_1^{vv}$$

$$R_2 = gdr^{-1} R_2^{vv}$$

$$N = gdr^{-1} N_v$$

These adjusted autocorrelations are then used as input to the standard moment processing algorithms with dBZo calibrated with respect to the HH channel.

HH+VV Channels

This approach gives the benefit of doubling the number of samples used for the reflectivity calculation.

$$T_0 = \frac{T_0^{hh} + gdr^{-1} T_0^{vv}}{2}$$

$$R_0 = \frac{R_0^{hh} + gdr^{-1} R_0^{vv}}{2}$$

$$R_1 = \frac{R_1^{hh} + gdr^{-1} R_1^{vv}}{2}$$

$$R_2 = \frac{R_2^{hh} + gdr^{-1} R_2^{vv}}{2}$$

$$N = \frac{N_h + gdr^{-1} N_v}{2}$$

These adjusted autocorrelations are then used as input to the standard moment processing algorithms with dBZo calibrated with respect to the HH channel.

Case 4: Alternating Dual–Channel (HH, VH, VV, HV)				
dBZo from HH Channel	**TTY Setup Question Responses**			
Calculate T, Z, V, W from:	HXmt	VXmt	CoRcv	CxRcv
HH	YES	NO	YES	NO
VH (xdr^{1}weighting)	YES	NO	NO	YES
VV (gdr^{-1}weighting)	NO	YES	YES	NO
HV (xdr/gdr weighting)	NO	YES	NO	YES
HH+VV (gdr^{-1}weighting)	YES	YES	YES	NO
HV+VH (xdr & gdr weighting)	YES	YES	NO	YES

HH Channel

Since the HH channel is directly calibrated this is the recommended choice. Processing is identical to a conventional radar.

VH Channel

Processing is identical to Case 1H: Horizontal Transmit HV Processing.

VV Channel

Processing is identical to Cases 2&3:STAR and Single Channel Alternating VV Processing.

HV Channel

The weighting in this case uses both xdr and gdr.

$$T_0 = \frac{xdr}{gdr} T_0^{hv}$$

$$R_0 = \frac{xdr}{gdr} R_0^{hv}$$

$$R_1 = \frac{xdr}{gdr} R_1^{hv}$$

$$R_2 = \frac{xdr}{gdr} R_2^{hv}$$

$$N = \frac{xdr}{gdr} N_h$$

These adjusted autocorrelations are then used as input to the standard moment processing algorithms with dBZo calibrated with respect to the HH channel.

HH + VV Channels

Processing is identical to Cases 2&3: STAR and Single Channel Alternating HH+VV Processing.

HV + VH Processing

The weighting here has to correct for both transmitter and receiver effects in order to use the HH channel dBZ_o.

$$T_0 = \frac{\frac{xdr}{gdr} T_0^{hv} + xdr^{-1} T_0^{vh}}{2}$$

$$R_0 = \frac{\frac{xdr}{gdr} R_0^{hv} + xdr^{-1} R_0^{vh}}{2}$$

$$R_1 = \frac{\frac{xdr}{gdr} R_1^{hv} + xdr^{-1} R_1^{vh}}{2}$$

$$R_2 = \frac{\frac{xdr}{gdr} R_2^{hv} + xdr^{-1} R_2^{vh}}{2}$$

$$N = \frac{\frac{xdr}{gdr} N_h + xdr^{-1} N_v}{2}$$

These adjusted autocorrelations are then used as input to the standard moment processing algorithms with dBZo calibrated with respect to the HH channel.

An example of how this weighted averaging works is given here. Suppose that we want to compute the average of the reflectivities for the VH and HV channels,

$$Z^{hv+vh} = Cr^2 \frac{S_{hv} + S_{vh}}{2}$$

$$= Cr^2 \frac{\frac{T_0^{hv} - N_h}{g_h^r g_h^t} + \frac{T_0^{vh} - N_v}{g_v^r g_h^t}}{2} = \frac{Cr^2}{g_h^r g_h^t} \frac{\left(T_0^{hv} - N_h\right)\frac{g_h^t}{g_v^t} + \left(T_0^{vh} - N_v\right)\frac{g_h^r}{g_v^r}}{2}$$

but since $xdr = \frac{g_v^r}{g_v^h}$ and $gdr = \frac{g_v^r g_v^t}{g_h^r g_h^t}$

$$Z^{vh+hv} = \frac{Cr^2}{g_h^r g_h^t}\left[\frac{\frac{xdr}{gdr}T_0^{hv} + xdr^{-1}\,T_0^{vh}}{2} - \frac{\frac{xdr}{gdr}N_h + xdr^{-1}N_v}{2}\right]$$

$$Z^{vh+hv} = \frac{Cr^2}{g_h^r g_h^t}[T_0 - N] = \left[\frac{Cr^2 N_h}{g_h^r g_h^t}\right]\left[\frac{r^2}{r_0^2}\right]\left[\frac{T_0 - N}{N_h}\right]$$

The first term in brackets is precisely dBZo for the HH channel. Thus if we average the correlations using the appropriate GDR and xdr weighting as shown above, then the average reflectivity is obtained by using conventional processing with the HH channel dBZo.

B.11 Thresholding of Polarization Parameters

The thresholding of polarization parameters by the processor eliminates bins with weak or uncertain signals. Note that the thresholding can be disabled if it is desired to see all of the data regardless of the data quality.

All of the polarization parameters are based on power ratios. The RVP8 requires that each power term in a ratio pass a signal-to-noise test similar to the log power test. For example, there are up to four different powers that can be calculated (alternating dual-channel case) so the tests for each of these are:

$$\frac{<|S_{hh}|^2>}{N_h} > N_{thresh}$$

$$\frac{<|S_{hv}|^2>}{N_h} > N_{thresh}$$

$$\frac{<|S_{vv}|^2>}{N_h} > N_{thresh}$$

$$\frac{<|S_{vh}|^2>}{N_h} > N_{thresh}$$

where the linearized threshold that is input as the dB LOG threshold, that is,

$$N_{thresh} = 10^{LOGthresh/10}$$

For example, a valid LDRH requires both a valid S_{hh} and a valid S_{vh}. The parameters RHOH and PHIH have the same requirement since they are the magnitude and phase of the cross-correlation function which is based on S_{hh} and S_{vh}.

There are two exceptions:

- **ZDR**

  ZDR requires that both Shh and Svv pass the signal-to-noise tests noted above. However, ZDR can be additionally thresholded by any of the other threshold parameters (LOG, SIG, SQI, CSR) similar to a standard moment. See *** 'TS Archive Log Area' on page 497 *** for a description of the standard moment thresholding.

- **PHIDP for single channel alternating case**

  PHIDP requires that both Shh and Svv pass the signal-to-noise tests noted above. In the single channel alternating case, PHIDP must also satisfy the additional test that the Doppler veloc-ity at the range bin must be valid, that is, not thresholded by its own criteria. This is because the algorithm for PHIDP in this case essentially subtracts the phase change due to the Dop-pler velocity. If the Doppler velocity is uncertain, the algorithm cannot produce reliable re-sults.

B.12 Calibration Considerations

Polarization systems require additional calibration as compared to conventional systems. There are three aspects to the calibration:

- dBZ_o measurement in both channels for dBZ and dBT calibration.
- gdr measurement for ZDR calibration.
- xdr measurement for LDR calibration.

These are discussed below.

dBZ_oCalibration for dBZ

The RVP8 supports separate calibration of both polarization channels. Measurement of dBZ_o for each channel of a dual polarization system is identical to the conventional radar case described in 6.5 Reflectivity Calibration on page 252. Note that for a single-channel switching system, the only difference between the horizontal and vertical signal paths occurs after the high power switch, that is, differential insertion loss of the switch itself and any differential insertion loss of the waveguides and feed after the switch. This means that for single-channel switching systems it may be sufficient to calibrate at one polarization and then adjust the calibration of the other channel by the differential gain GDR (see below).

GDR Calibration for ZDR

The ZDR Offset is the dB value of the relative gain between the co-polarized channels including both transmitter and receiver gain, that is,

$$ZDROffset = 10LOG\ \frac{g_v^r g_v^t}{g_h^r g_h^t} \quad \text{and} \quad gdr = \frac{g_v^r g_v^t}{g_h^r g_h^t}$$

GDR is input into the processor as a dB value. However, for analyses in this chapter, the linear gdr value is sometimes more convenient.

In principle, if dBZ_o could be calibrated perfectly in both channels, measurement of GDR would not be required. In practice, this is not possible because dBZ_o cannot be calibrated to an absolute accuracy sufficient for ZDR, that is, to 1/16th of a dB. Therefore, the RVP8 uses the GDR approach.

Since GDR includes both transmitter and receiver differential gains, accurate calibration requires that an actual target be observed. One way to do this is as follows:

- Set the GDR to be 0 dB using your application software (for example, for Vaisala IRIS systems in the setup utility RVP section). Disable clutter filtering for ZDR in either your application software (by selecting filter 0) or explicitly in the RVP8 TTY setups mp section.
- Place the antenna at 90 degrees elevation (vertical incidence) during moderate to heavy rain. The melting layer should be at a height that is well above the recovery zone of the T/R and in the antenna "far zone". A melting layer higher than 2 km is suggested, but the specific characteristics of the radar should be considered.
- Collect ZDR data at vertical incidence while the antenna is rotating in azimuth.
- Use a separate application program to average the ZDR values around a full 360 degrees at each range bin (height). Generate a plot of 360-average ZDR vs height.
- You should observe that the average ZDR values in regions of strong signal (>20 dB SNR) below the bright band are approximately constant with height. This is the value that should be used in your application software for GDR.
- Enter the value and repeat the calibration to verify that the average ZDR is now 0 dB.

The rationale for this approach is as follows. When viewed at vertical incidence, rain should have a ZDR of 0 dB since the drops will all appear circular. The reason for averaging over 360 degrees is to cancel-out effects from sidelobe contamination from nearby ground targets and other artifacts of the antenna/feed/radome system. For example the radome may have an obstruction light on the top. Some of these artifacts can be minimized by assuring the the weather targets are strong, that is, heavy rain is preferred for this calibration.

Offset Calibration for LDR

XDR is the dB value of the relative gain between the co- and cross-receiver channels for LDR measurements. Analogous to GDR, it is defined as the dB value of the ratio of the vertical to horizontal receiver gains, for example,

$$XDR = 10LOG\frac{g_v^r}{g_h^r} \quad \text{and} \quad xdr = \frac{g_v^r}{g_h^r}$$

Three techniques for calibration of XDR are discussed. It is recommended for the transmitter to be off for all of these methods.

- 1. **Solar method**

  Use the sun to measure LDR. The measured value of LDR is then the XDR offset. LDR should be measured in fixed mode for both LDRH and LDRV. The values should be reciprocal (for example, +1 dB and -1 dB). Use the average of the absolute value if they are not precisely reciprocal (for example, for +1.4 and -1.2 use 1.3). Finally after inputting the XDR value, retest to verify that the sun has been properly corrected to have zero LDR.
- 2. **Signal generator method with connection to waveguide**

  Connect a signal generator with a splitter to both channels and measure XDR directly. This does not account for any effects that are before the coupler (for example, waveguide, feed, radome, antenna gain).
- 3. **Linear feed horn remote radiator method**

  Use a calibrated linear feed horn with an RF source located several hundred meters from the radar. Maximize the H channel return and measure the response using the RVP8 pr command "Filtered" power in the "Primary Channel". Now rotate the feed horn to vertical and maximize the power in the "Secondary Channel". The difference in dB is XDR. Note that signal multi–path effects could bias the results from this technique.

In all cases it is recommended that for the calibration, XDR be set to 0 dB in the application user software and that the RVP8 TTY setups be configured as follows:

- Noise correction enabled for LDR and noise sample taken prior to the measurements (with care not to sample with a test signal turned–on or while looking at the sun).
- Clutter correction disabled for LDR.

APPENDIX C
RVP8/RCP8 PACKAGING

A standard RVP8/RCP8 processor consists of three separate units:

-	Main Chassis	C.1 Main Chassis General Description on page 394	**RVP8 and RCP8**
-	Connector Panel	C.2 I/O-62 and Connector Panel on page 406	**RVP8 and RCP8**
-	IFD (IF Digitizer)	C.3 IFD Module (RVP8 Only) on page 423	**RVP8 Only**

Because of the similarity of the packaging for the RVP8 and RCP8, both units are described here.

The main chassis and connector panel are located in a rack within 100m of the IFD. Typically the main chassis interfaces to a host computer via 100 BaseT Ethernet. For the RVP8, the IFD receiver module resides in the radar cabinet.

This section describes the general features of the packaging and the electrical specifications and cabling of these units. Read *CAREFULLY* the following warnings before you apply power to your system.

WARNING The Main Chassis power supply modules are NOT auto ranging. These must be set by a switch on each module for either 115/230 VAC 60/50 Hz. Verify these before applying power to the system. See C.1.3 Main Chassis Back Panel Power Section on page 401.

WARNING Turn off power to the main chassis before installing or removing any PCI boards. For safety, the line cord should be disconnected before opening either the IFD module or main chassis.

NOTE	The circuit boards contain many static sensitive components. Do not handle the boards or open the IFD module unless a properly grounded wrist strap is worn.

C.1 Main Chassis General Description

Vaisala standard main chassis is a 4U rackmount/table top enclosure (43.2 wide x 43.2 long x 17.8 cm high) or (17 wide x 17 long x 7.00 inch high) which fits a standard 19-inch EIA rack. The system comes standard with hot–swap redundant power supplies. The chassis may be equipped with either a mother board or a single-board computer depending on how the unit was purchased. The chassis is shown in the following figures.

- Front View **Figure C–1**
- Rear View **Figure C–2**
- Side view **Figure C–3**
- Internal Wiring **Figure C–4**

The front of the unit has a plasma matrix display that is used for status information. There is also a CDRW drive (for software installation and backup) and in most cases, a floppy drive as well (for configuration backup).

Two fans are mounted behind the door on the front of the enclosure. These draw ambient air in to the unit. The air flows through the unit and exits the rear. Do not block the slots or the exhaust grills on the fans. Check airflow now and then, and also check the board and fan screen for dust accumulation. If necessary, excessive dust accumulations on the board can be cleaned at a properly equipped static-free workstation with "canned air" or Chemtronics TF-Plus solvent, which can be purchased through electronics distributors.

The boards should be left in the chassis whenever the unit is shipped. This minimizes handling and static risk. Save the original packing provided for shipment.

NOTE	Prior to shipment, see www.vaisala.com and contact helpdesk@vaisala.com to obtain a returned materials authorization (RMA) and to coordinate the shipping.

A table top unit can be converted for rack mount by simply installing rack mount ears. The rack ears are installed with #8-32 flat head screws. It is

strongly recommended that the rack mount slide brackets supplied with the unit should be installed in the rack for additional structural support.

The internal cabling diagram in Figure 51 on page 399 shows how the various disk drives, power supplies, etc. are connected within the standard Main Chassis. A mother board example is shown. Use this as a guide if you have to replace internal components.

The remainder of this section describes the front and rear panel of the Main Chassis.

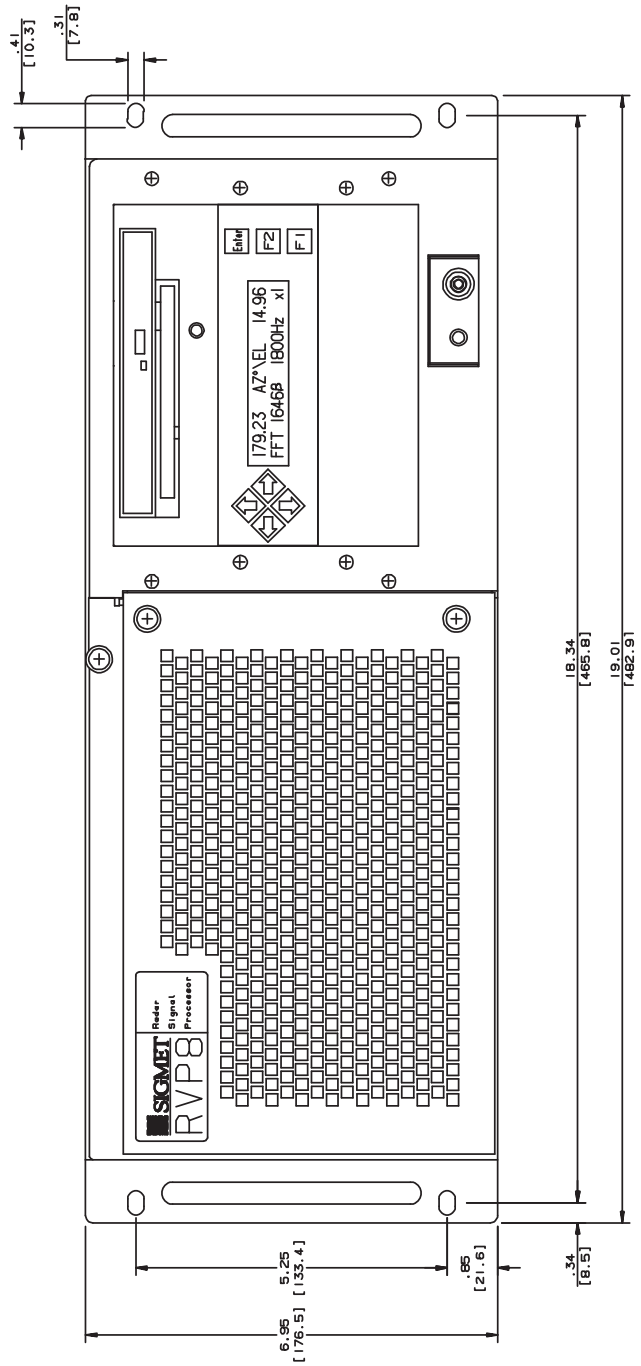


Figure 48 Main Chassis - Front Panel

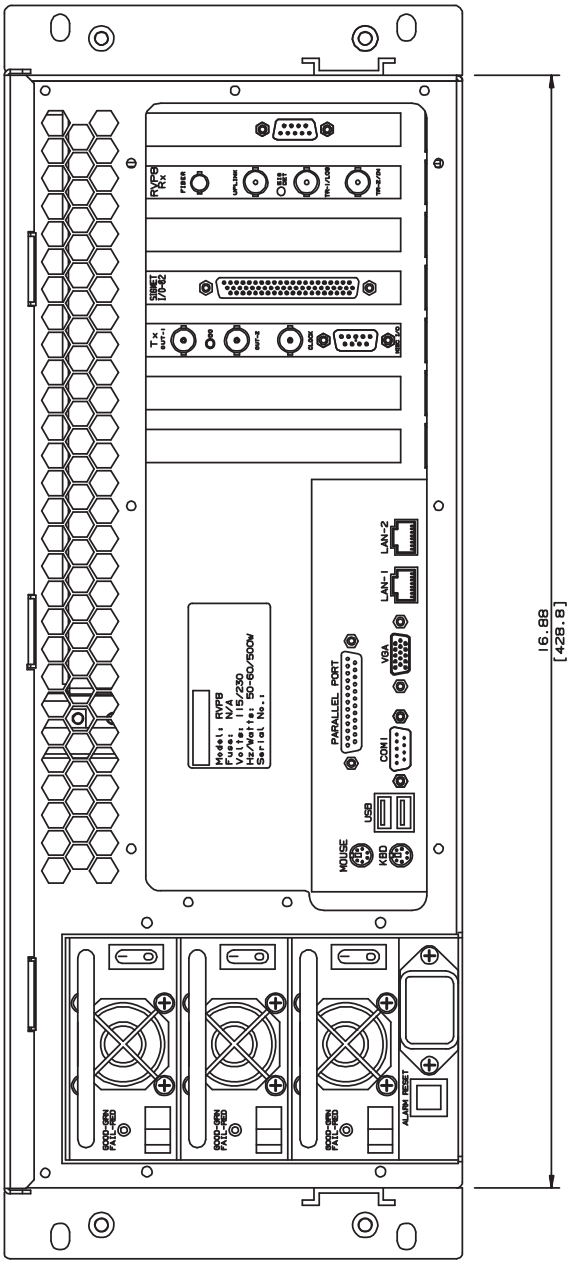


Figure 49 Main Chassis - Back Panel

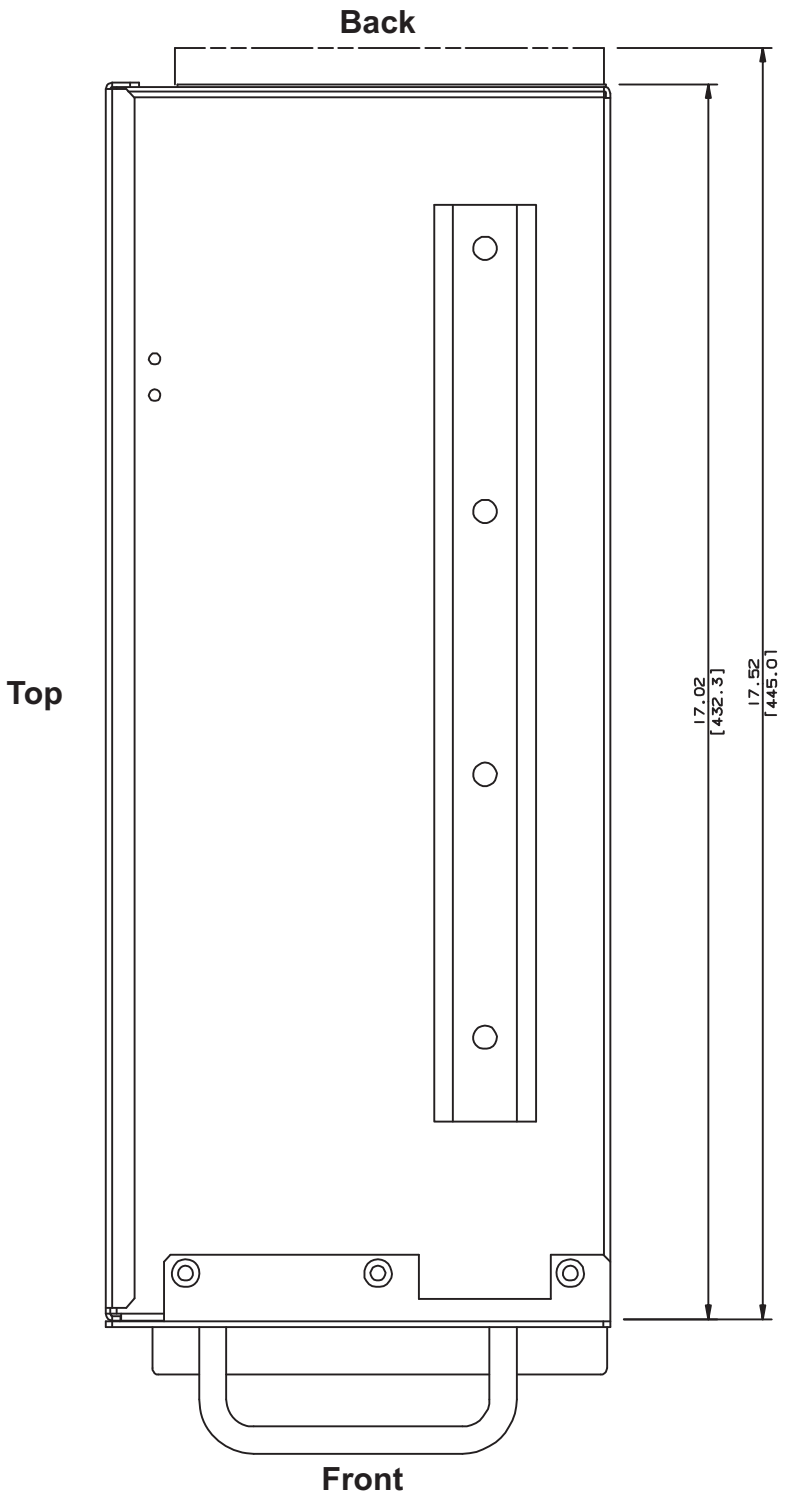


Figure 50 Main Chassis - Right Side View

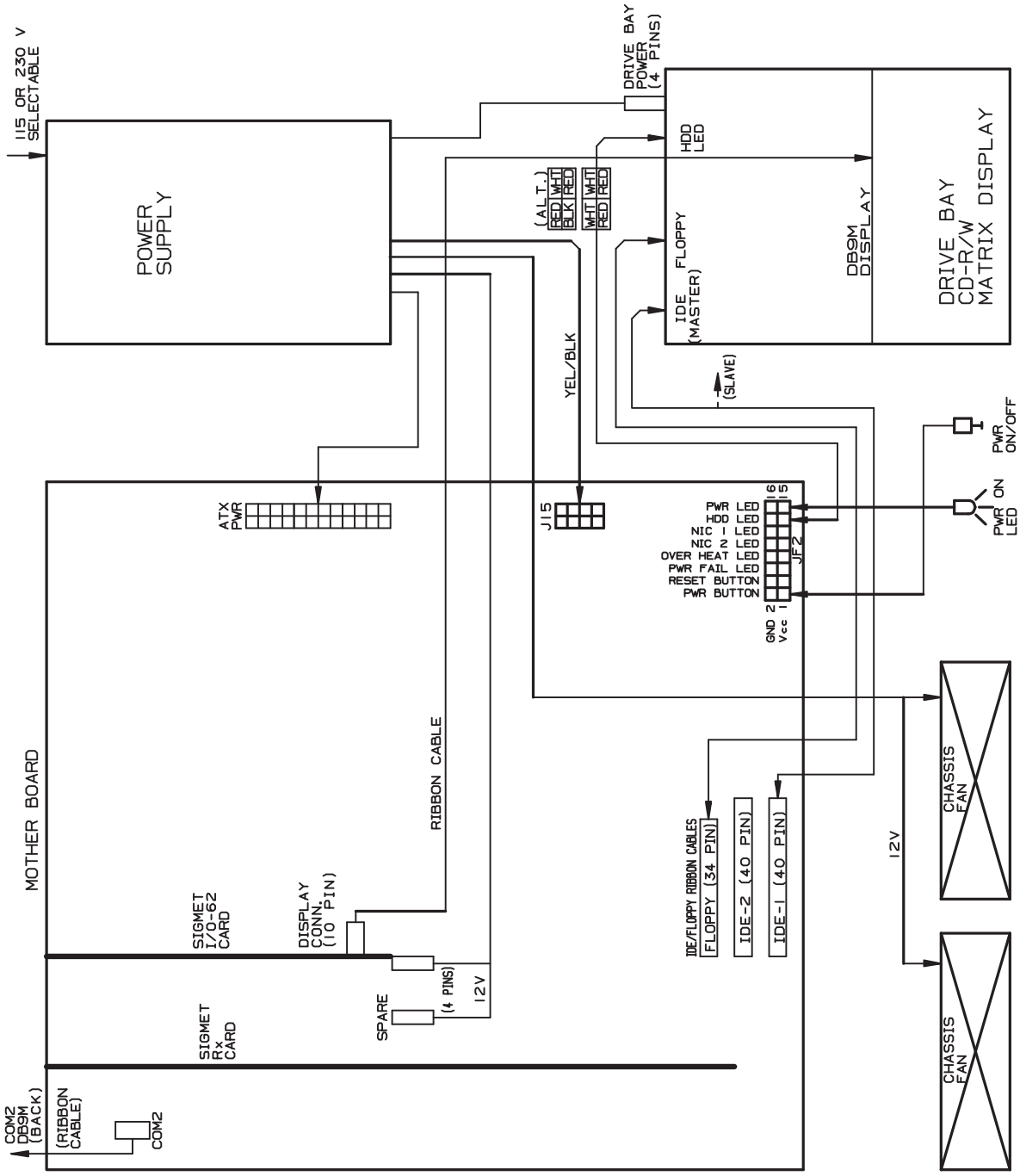
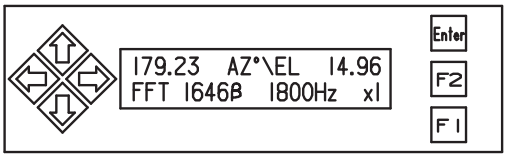


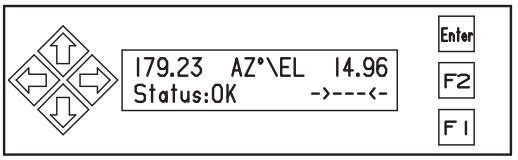
Figure 51 Main Chassis Internal Cabling

C.1.1 Main Chassis Front Panel

The front panel is shown in Figure 48 on page 396. The front panel matrix plasma display is typically connected internally by a ribbon cable to either an I/O-62 card or an RVP8/Rx card. The display is used to show status and power-up test results. The function keys beside the display are not currently used.



RVP8 Receiver/Signal Processor



RCP8 Radar Control Processor

In addition, the DVD+RW/CDRW are located on the front panel. The various activity lights are for the DVD (yellow), floppy drive (small green) and the hard disk drive (large red). The cabling diagram shows how to connect the activity lights. At the lower right of the unit is a power on/off switch and a green LED to indicate that power is on.

C.1.2 Main Chassis Back Panel

Figure 49 on page 397 shows an example of the main chassis back panel for the case of a motherboard system. There are three main sections to the Main Chassis back panel:

- **Power section**- on the left (looking from the rear) with the power entry module, alarm reset and three redundant hot-swap power supplies.
- **PC I/O section**- in the lower center with connectors for keyboard, mouse, monitor, network, etc. This is for a mother board example.
- **PCI card section**- on the right (looking from the rear) with standard PCI slots for the RVP8/RCP8 circuit cards as well as other standard commercial PCI cards that may be used (for example, a four port serial card).

Note that depending on whether your system is using a mother board or single-board computer (SBC), the appearance of these sections may be

different, but the functions are the same. These sections are described in detail in the sections below.

C.1.3 Main Chassis Back Panel Power Section

WARNING The Main Chassis power supply modules are NOT auto ranging. These must be set by a switch on each module for either 115/230 VAC 60/50 Hz. Verify these per the procedure below, before applying power to the system.

The Main Chassis back panel is equipped with a modular AC power entry device. There are three hot-swap redundant power supply modules in the system. The procedure for setting/verifying the voltage on each one is as follows:

- The unit should be powered-off. This can be assured by simply not connecting the power input cord.
- Remove the top power supply module by shifting the black release button to the right.
- Use the handle to pull the module out.
- Check the red power selector switch on the right side (rear) of the module and set it as appropriate to your line voltage (115/230).
- Re–insert the module and push the chrome handle down. This switches the module in the on "1" position.
- Repeat this procedure for the middle and lower modules (the order is not critical)).

When the system is switched–on, the LED on each module shows green to indicate that it is functioning properly. A red light indicates a failure. There is an audio "buzzer" alarm in the event that a power module is turned-off, removed or fails.

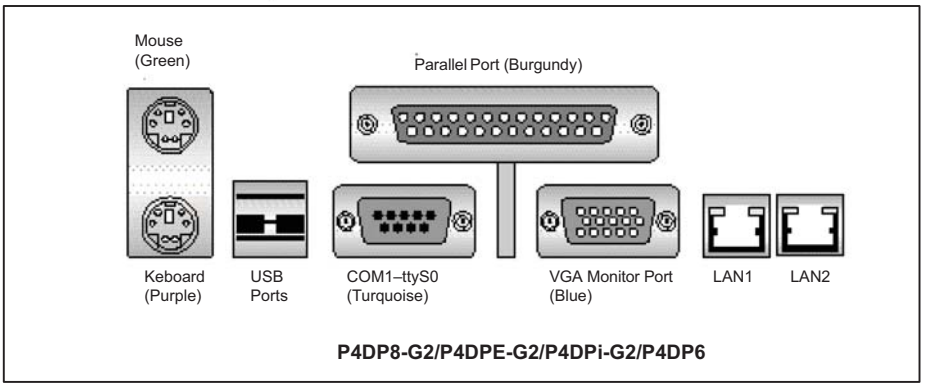
NOTE The red button next to the power entry module will reset the "buzzer" alarm.

The system will function if there is failure of any one power module, but is not rated to function with only one module, that is, if two modules fail. Each power module is equipped with internal protection for over-temperature and over-current. In the event that the protection is triggered, the module LED will show red. It can be reset by removing the module for

a minute and then inserting the module back into the system. It is best to do this with power–off to the module.

NOTE If a power module is switched on, but the LED indicator is red, then it is not functioning. The reset procedure is to turn the power off on the failed module, remove it for one minute and then re-insert it and power it back on.

C.1.4 Main Chassis Back Panel PC I/O Section



The PC I/O section shown above is where you make all of your standard PC connections. Note pins (male) are indicated by filled black circles while sockets (female) are indicated by open circles.

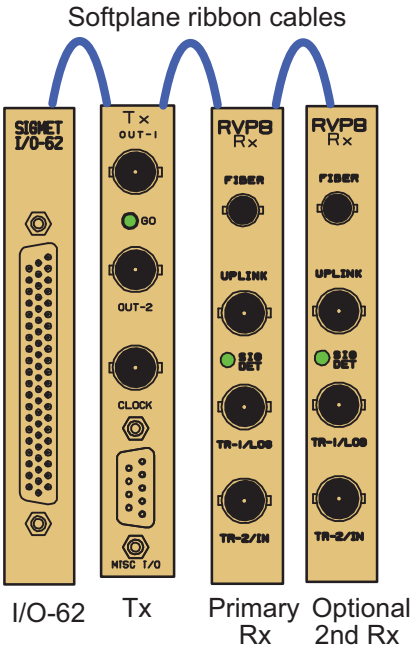
A standard keyboard and mouse are provided with the unit. VGA monitor is supplied by the customer or ordered as an option from Vaisala.

Note that LAN 1 and LAN 2 are standard RJ45 connectors. For the –G2 style mother boards the LAN port speed is 100/1000 BaseT. For the –Q they are 10/100 BaseT.

The keyboard and mouse are standard PS/2. You can use an adapter to plug a USB mouse into the circular mouse connector.

COM1 (/dev/ttyS0) is the DBM9 connector. COM2 (/dev/ttyS1) is typically installed as a separate DBM9 connector in the PCI section.

C.1.5 Main Chassis Back Panel PCI Card Section



The PCI cards are installed vertically on the right of the chassis (looking from the back). Since there are many different RVP8/RCP8 configuration options that can be ordered, there is quite a bit of variability in what PCI cards are installed. The order of the cards in the slots IS important. The table below gives the required card sequence from right to left as viewed from the back of the chassis (like the figure). There can be blank slots in between the cards as long as the order is preserved. Note, all Vaisala cards require a softplane ribbon cable as shown in the figure, except for the case of a minimal system with only a single Rx card. Note that COM2 is typically installed as a DBM9 connector on an otherwise blank panel in the PCI section.

PCI Card	Card Slot Order	Used on	Vendor	Functions
RVP8/Rx	1st	RVP8 Dual Pol System	Vaisala	Optional Secondary Rx Card. Used for vertical or crosspolarized receiver channel.
RVP8/Rx	2nd	RVP8 Standard	Vaisala	Primary Rx Card. Used for horizontal or co-polarized receiver channel.
RVP8/Tx	3rd	RVP8	Vaisala	Two waveform outputs, clock output/ input, 4 RS422 lines
I/O 62	4th	RVP8 RCP8	Vaisala	I/O to radar system control and monitoring. Usually connected to an I/ O-62 Connector Panel.
COM3N	5th	RVP8 RCP8	Market	Additional RS232C serial card (typically 4 channels).
HPIB	6th	RVP8 RCP8	Market	HPIB control for signal generator or other test equipment

Card slot order is from right-to-left when viewed from the back.

The I/O-62 is used on both the RVP8 and RCP8. It is described in C.2 I/O-62 and Connector Panel on page 406, along with the standard connector panels for the RVP8 and RCP8. See Chapter 3, Hardware Installation, on page 65 of the *RVP8 User's Manual* for a description of the connectors on the RVP8 PCI cards. The jumper settings for the I/O-62, Rx and Tx Cards are described in the tables below. Note that the high-lighted entries correspond to the default factory jumper settings.

Table 25 RVP8 and RCP8 I/O-62 Card Jumper Settings

Jumper	ID	Description	AB	BC	Not Installed
JP1	BOOT	Controls the card bootup	**X**	**X**	**Normal Boot**
JP2	JTAG	Enables on-board "flash" re-programming for code version upgrades. Other settings are reserved for Vaisala maintenance functions.	Enable Flash	Maintenance	Maintenance
JP3	TTYX0/ RSV	These jumpers assign dedicated hardware I/O lines to pins on DBF62 connector on the back of the I/O-62. The selections are made among: 1.) The two RS232 lines noted as TTY0 or 1 with transmit and receive for each). 2.) Four trigger output lines. 3.) The three contact positions of the onboard DIP relays (K1 and K2). Note the specific pins are listed in the AB column.	**Pin 47 TTYX0**	**X**	**X**
JP4	TRIG0/ TTYX0		**Pin 49 TRIG0**	TTYX0	**X**
JP5	TRIG1/ K1NC		**Pin 51 TRIG1**	K1 Normally Closed Contact	**X**
JP6	TRIG2/ K1NO		**Pin 53 TRIG2**	K1 Normally Open Contact	**X**
JP7	TRIG3/ K1CT		**Pin 55 TRIG3**	K1 Center Contact	**X**
JP8	TTYR0/ K2NC		**Pin 57 TTYR0**	K2 Normally Closed Contact	**X**
JP9	TTYX1/ K2NO		**Pin 59 TTYX1**	K2 Normally Open Contact	**X**
JP10	TTYR1/ K2CT		**Pin 61 TTYR1**	K2 Center Contact	**X**

Table 26 RVP8/Rx Card Jumper Settings

Jumper	ID	Description	AB	BC	Not Installed
JP1	LOG	Select TRIG1 out or LOG out for card BNC labeled "TR-1/LOG"	LOG Out	**TRIG1 Out**	**X**
JP2	TGV	TRIG1 output voltage	**5 V**	12 V	**X**
JP3	TIN	Select TRIG2 out or TRIG-IN for card BNC labeled "TR-2/IN"	TRIG-IN	**TRIG2 Out**	**X**
JP4	TGV	TRIG2 output voltage	**5 V**	12 V	**X**
JP5	BOOT	Controls the card bootup	**X**	**X**	**Normal Boot**
JP6	JTAG	Enables on-board "flash" re-programming for code version upgrades. Other settings are reserved for Vaisala maintenance functions.	**Enable Flash**	Maintenance	Maintenance
JP7	TERM	TRIG-IN Termination	**Un-terminated**	75 Ohm	**X**

Table 27 RVP8/Tx Card Jumper Settings

Jumper	ID	Description	AB	BC	Not Installed
JP1	BOOT	Controls the card bootup	**X**	**X**	**Normal Boot**
JP2	JTAG	Enables on-board "flash" re-programming for code version upgrades.Other settings are reserved for Vaisala maintenance functions.	**Enable Flash**	Maintenance	Maintenance
JP3A	—	Reserved for future use	**Reserved**	**X**	**X**
JP3B	—	Reserved for future use	**Reserved**	**X**	**X**

C.2 I/O-62 and Connector Panel

Figure 52 on page 409 and Figure 53 on page 410 show the I/0-62 Connector Panel for the RVP8 and RCP8. This is typically mounted on the same rack as the Main Chassis. A 1:1 62–position cable (standard 1.8 m/6 foot) connects the connector panel to the I/O-62. As shown in the figures, the cable can be connected to either the front or the back of the panel so that the cable run can be optimized. In most cases, it is recommended to connect the cable to the back of the panel to minimize the risk of physical damage to the cable.

The panel is electrically the same for both the RVP8 and RCP8. Indeed the circuit board is identical. However, the panel labelling and the softplane configurations are different.

The pin assignments to the various connectors are described in Table 28 on page 411 to Table 41 on page 423 located at the end of this section. The tables show the basic electrical properties of each pin and the default ignal assignment (if any) that is made in the factory softplane.conf file. The softplane approach provides a great deal of flexibility in assigning the I/O to the panel.

The I/O–62 PCI card provides forty multi–protocol digital interface lines at its 62–pin faceplate connector. These lines are grouped into five independent and identical blocks, each of which contains eight lines. Moreover, each of these blocks of eight lines can be further divided into four line pairs.

Each block of I/O lines can operate in one of the following modes:

- As eight TTL/CMOS single–ended outputs
- As eight TTL/CMOS single–ended inputs
- As N RS–422 differential transmitters or receivers, and (8–2N) TTL/CMOS single–ended inputs.

The assignment of electrical levels and signal directions are all made in the softplane.conf file. Users do not have to worry about how to configure each block of lines because inconsistent signal assignments will be checked and reported when the file is loaded.

All forty I/O–62 digital lines are individually protected against both overvoltage and electrostatic discharge (ESD). You may safely apply voltages between -27V and +27V to any line regardless of whether it is configured for an input or output. Likewise, external ESD ulses of 15KV (Human body model) will be safely shunted to ground at the 62–pin connector point of entry.

This wide voltage tolerance effectively makes the TTL/CMOS inputs function as wide range comparators with a 2.5V logic threshold. These inputs could be connected directly to a 24V panel bulb, for example, in order to monitor its On/Off status. Note that the line protection circuitry has a side effect of raising the output impedance of the TTL/CMOS drivers to approximately 120–Ohms. This should not ause any trouble unless the signal is heavily terminated at the receiving nd. The RS–422 drivers are not affected by the line protection, and have the standard very low output impedance.

The I/O–62 provides a variety of terminations for its digital I/O lines. The TTL/CMOS signals can optionally be pulled either to GND or to +5V through a 2.2K–Ohm resistor. Similarly, the RS–422 linepairs can optionally be terminated with a 100–Ohm resistor across each pair.

There are a few additional constraints that should be kept in mind when assigning electrical signals to a block of eight I/O–62 lines. These are:

- When TTL/CMOS pull-up/pull-down resistors are enabled, they are applied to the entire group of eight lines. This is somewhat inconsistent with using some of those same lines as RS–422.
- Similarly, when RS–422 terminators are enabled, they are applied to all four line pairs. This is completely inconsistent with using some of those same lines as TTL/CMOS.

Thus, if line termination is required, it is usually necessary to split the TTL/CMOS and RS–422 functions so that both do not appear within the same block of eight lines.

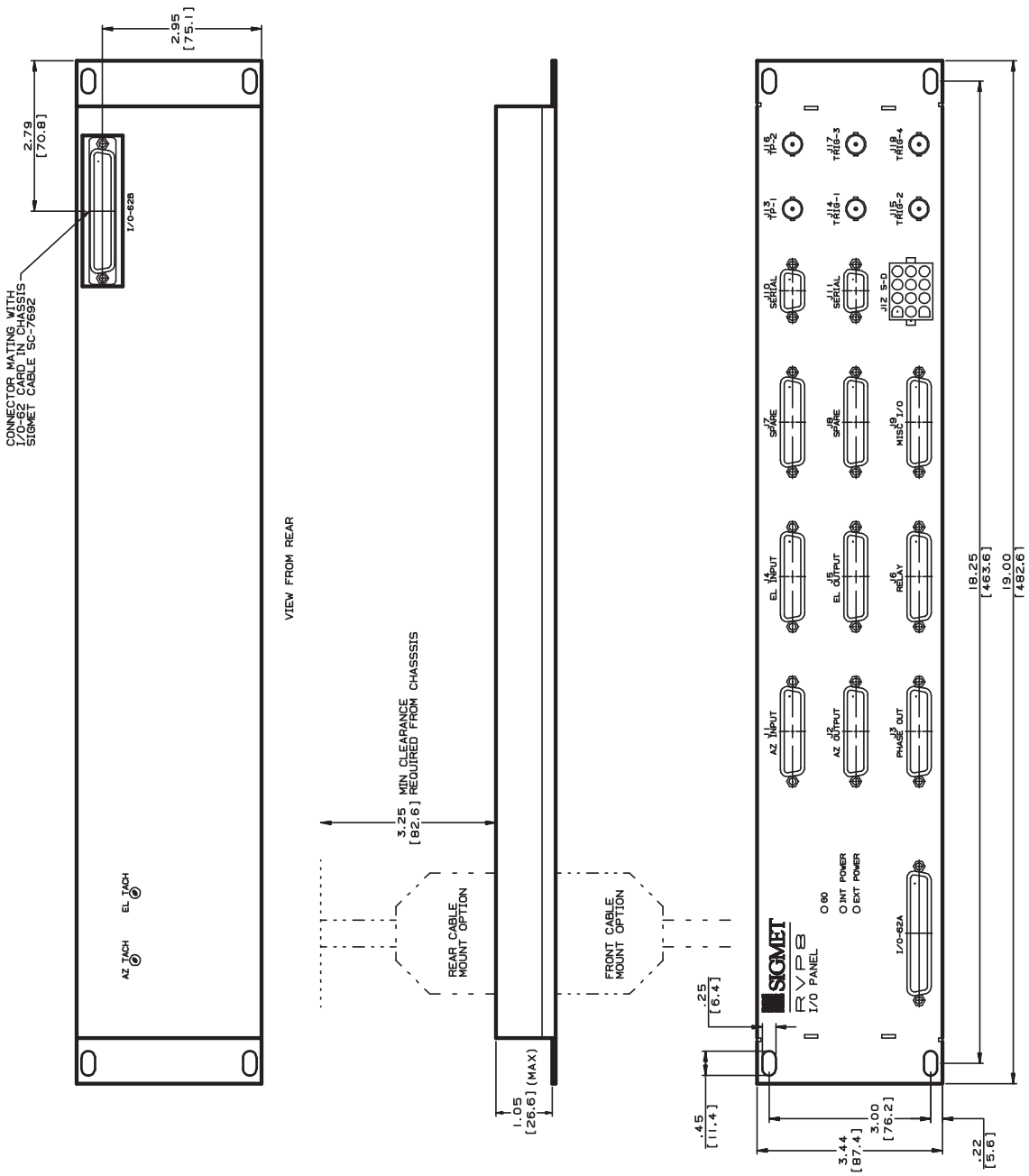


Figure 52 RVP8 I/O-62 Connector Panel

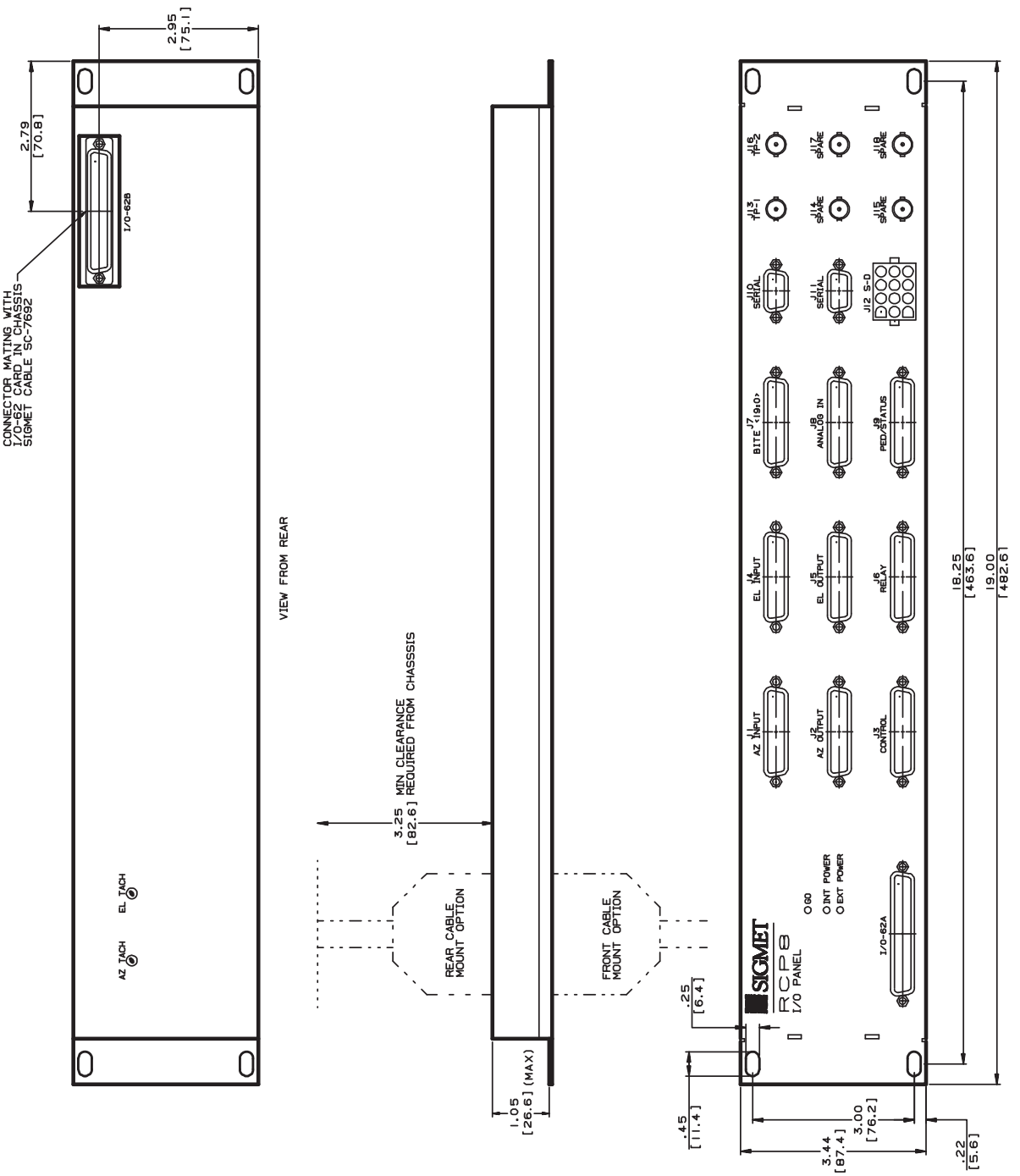


Figure 53 RCP8 I/O-62 Connector Panel

Table 28 J1 "AZINPUT"

Pin	Electrical Specification	RVP8 Signal Name	RCP8 Signal Name
1	TTL	sPedAZ[0]	sPedAZ[0]
2	TTL	sPedAZ[1]	sPedAZ[1]
3	TTL	sPedAZ[2]	sPedAZ[2]
4	TTL	sPedAZ[3]	sPedAZ[3]
5	TTL	sPedAZ[4]	sPedAZ[4]
6	TTL	sPedAZ[5]	sPedAZ[5]
7	TTL	sPedAZ[6]	sPedAZ[6]
8	TTL	sPedAZ[7]	sPedAZ[7]
9	TTL	sPedAZ[8]	sPedAZ[8]
10	TTL	sPedAZ[9]	sPedAZ[9]
11	TTL	sPedAZ[10]	sPedAZ[10]
12	TTL	sPedAZ[11]	sPedAZ[11]
13	TTL	sPedAZ[12]	sPedAZ[12]
14	TTL	sPedAZ[13]	sPedAZ[13]
15	TTL	sPedAZ[14]	sPedAZ[14]
16	TTL	sPedAZ[15]	sPedAZ[15]
17	TTL		
18	TTL		
19	TTL		
20	TTL		
21	GND		
22	GND		
23	GND		
24	GND		
25	GND		

Table 29 J2 "AZ OUTPUT"

Pin	Electrical Specification	RVP8 Signal Name	RCP8 Signal Name
1	TTL	cEarthAZ[0]	cEarthAZ[0]
2	TTL	cEarthAZ[1]	cEarthAZ[1]
3	TTL	cEarthAZ[2]	cEarthAZ[2]
4	TTL	cEarthAZ[3]	cEarthAZ[3]
5	TTL	cEarthAZ[4]	cEarthAZ[4]
6	TTL	cEarthAZ[5]	cEarthAZ[5]
7	TTL	cEarthAZ[6]	cEarthAZ[6]
8	TTL	cEarthAZ[7]	cEarthAZ[7]
9	TTL	cEarthAZ[8]	cEarthAZ[8]
10	TTL	cEarthAZ[9]	cEarthAZ[9]
11	TTL	cEarthAZ[10]	cEarthAZ[10]
12	TTL	cEarthAZ[11]	cEarthAZ[11]
13	TTL	cEarthAZ[12]	cEarthAZ[12]
14	TTL	cEarthAZ[13]	cEarthAZ[13]
15	TTL	cEarthAZ[14]	cEarthAZ[14]
16	TTL	cEarthAZ[15]	cEarthAZ[15]
17	TTL		
18	TTL		
19	TTL		
20	TTL		
21	GND		
22	GND		
23	GND		
24	GND		
25	GND		

Table 30 J3 RVP8: "PHASE OUT"; "RCP8 CONTROL"

Pin	Electrical Specification	RVP8 Signal Name	RCP8 Signal Name
1	**Configurable I/ O62 Digital Lines**:		cPWidth[0]
2			cRadiateOn
3			cServoPwr
4			cReset
5			sPWidth[0]
6			sRadiate
7			sServoPwr
8			sReset
9	RS422+		
10	RS422+		
11	GND		
12	GND		
13	GND		
14	**Configurable I/ O62 Digital Lines:**		cPWidth[1]
15			cRadiateOff
16			cTransmitPwr
17			
18			sPWidth[1]
19			
20			sTransmitPwr
21			
22	RS422-		
23	RS422-		
24	GND		
25	GND		

Notes:

1.) I/O-62 lines can be configured in **softplane.conf** in groups of 4 for the following options:

RS422 differential vs single–ended

Input or output sense

Input termination for single-ended lines can be pull-up (...Term=1), pull-down (...Term=-1) or un-terminated (...Term=0)

1.) All RCP8 status variables (starting with "s") are terminated with pull–up's in the default **softplane.conf**.

2.) All RCP8 control variables (starting with "c") are un-terminated (...Term=0).

Table 31 J4 "EL INPUT"

Pin	Electrical Specification	RVP8 Signal Name	RCP8 Signal Name
1	TTL	sPedEL[0]	sPedEL[0]
2	TTL	sPedEL[1]	sPedEL[1]
3	TTL	sPedEL[2]	sPedEL[2]
4	TTL	sPedEL[3]	sPedEL[3]
5	TTL	sPedEL[4]	sPedEL[4]
6	TTL	sPedEL[5]	sPedEL[5]
7	TTL	sPedEL[6]	sPedEL[6]
8	TTL	sPedEL[7]	sPedEL[7]
9	TTL	sPedEL[8]	sPedEL[8]
10	TTL	sPedEL[9]	sPedEL[9]
11	TTL	sPedEL[10]	sPedEL[10]
12	TTL	sPedEL[11]	sPedEL[11]
13	TTL	sPedEL[12]	sPedEL[12]
14	TTL	sPedEL[13]	sPedEL[13]
15	TTL	sPedEL[14]	sPedEL[14]
16	TTL	sPedEL[15]	sPedEL[15]
17	TTL		
18	TTL		
19	TTL		
20	TTL		
21	GND		
22	GND		
23	GND		
24	GND		
25	GND		

Table 32 J5 "EL OUTPUT"

Pin	Electrical Specification	RVP8 Signal Name	RCP8 Signal Name
1	TTL	cEarthEL[0]	cEarthEL[0]
2	TTL	cEarthEL[1]	cEarthEL[1]
3	TTL	cEarthEL[2]	cEarthEL[2]
4	TTL	cEarthEL[3]	cEarthEL[3]
5	TTL	cEarthEL[4]	cEarthEL[4]
6	TTL	cEarthEL[5]	cEarthEL[5]
7	TTL	cEarthEL[6]	cEarthEL[6]
8	TTL	cEarthEL[7]	cEarthEL[7]
9	TTL	cEarthEL[8]	cEarthEL[8]
10	TTL	cEarthEL[9]	cEarthEL[9]
11	TTL	cEarthEL[10]	cEarthEL[10]
12	TTL	cEarthEL[11]	cEarthEL[11]
13	TTL	cEarthEL[12]	cEarthEL[12]
14	TTL	cEarthEL[13]	cEarthEL[13]
15	TTL	cEarthEL[14]	cEarthEL[14]
16	TTL	cEarthEL[15]	cEarthEL[15]
17	TTL		
18	TTL		
19	TTL		
20	TTL		
21	GND		
22	GND		
23	GND		
24	GND		
25	GND		

Table 33 J6 "RELAY"

Pin	Electrical Specification	RVP8 Signal Name	RCP8 Signal Name
1	**Relay K1: CT**	**cPWidth[0]**	**cPWidth[0]**
2	**Relay K1: NO**		
3	**Relay K1: NC**		
4	Relay K2: CT	cPWidth[1]	cPWidth[1]
5	Relay K2: NO		
6	Relay K2: NC		
7	**Relay K3: CT**		
8	**Relay K3: NO**		
9	**Relay K3: NC**		
10	—		
11	GND		
12	GND		
13	GND		
14	+12VDC	**External Relay Control Power**	
15	+12VDC		
16	+12VDC		
17	+12VDC		
18	+12V Unreg		
19	Return14	**External Relay Control Returns**	
20	+12V Return15		
21	+12V Return16		
22	+12V Return17		
23	—		
24	GND		
25	GND		

WARNING To avoid possible damage to the connector panel, all external relays must be equipped with diode protection against the back EMF generated when the external relay coil is opened. Relays can be purchased with a diode installed or a diode can be added to the relay across the coil supply and return.

Notes: Internal relays K1, K2, K3 on the connector panel are dry contacts:

CT	Center contact
NO	Normally open contact
NC	Normally closed contact

Table 34 J7: "RVP8 SPARE"; "RCP8 BITE" 19:0

Pin	Electrical Specification	RVP8 Signal Name	RCP8 Signal Name
1	TTL		sAux[0]
2	TTL		sAux[1]
3	TTL		sAux[2]
4	TTL		sAux[3]
5	TTL		sAux[4]
6	TTL		sAux[5]
7	TTL		sAux[6]
8	TTL		sAux[7]
9	TTL		sAux[8]
10	TTL		sAux[9]
11	TTL		sAux[10]
12	TTL		sAux[11]
13	TTL		sAux[12]
14	TTL		sAux[13]
15	TTL		sAux[14]
16	TTL		sAux[15]
17	TTL		sAux[16]
18	TTL		sAux[17]
19	TTL		sAux[18]
20	TTL		sAux[19]
21	GND		
22	GND		
23	GND		
24	GND		
25	GND		

Table 35 J8: "RVP8 SPARE"; "RCP8 ANALOG IN"

Pin	Electrical Specification	RVP8 Signal Name	RCP8 Signal Name
1	±20VDCDifferentialAnalog Inputs Positive Side	Amux0+	Amux0+
2		Amux1+	Amux1+
3		Amux2+	Amux2+
4		Amux3+	Amux3+
5		Amux4+	Amux4+
6		Amux5+	Amux5+
7		Amux6+	Amux6+
8		Amux7+	Amux7+
9		Amux8+	Amux8+
10		Amux9+	Amux9+
11	GND		
12	GND		
13	GND		
14	±20VDCDifferentialAnalog Inputs Negative Side	Amux0-	Amux0-
15		Amux1-	Amux1-
16		Amux2-	Amux2-
17		Amux3-	Amux3-
18		Amux4-	Amux4-
19		Amux5-	Amux5-
20		Amux6-	Amux6-
21		Amux7-	Amux7-
22		Amux8-	Amux8-
23		Amux9-	Amux9-
24	GND		
25	GND		

Table 36 J9 "RVP8: MISC I/O" ; "RCP8: PED/STATUS"

Pin	Electrical Specification	RVP8 Signal Name	RCP8 Signal Name
1	**Configurable I/ O62 Digital Lines**:		sWavegpFlt
2			sInterlockFlt
3			sLocal
4			sLowerEL
5			
6			
7			
8	±6 to ±70 VDC Input		AzTach+
9	±6 to ±70 VDC Input		ElTach+
10	±10 VDC Output		AzDrive
11	GND		
12	GND		
13	GND		
14	**Configurable I/ O-62 DigitalLines**:		sAirflowFlt
15			sMagCurrentFlt
16			sStandby
17			sUpperEL
18			
19			
20			
21	±6 to ±70 VDC Input		AzTach-
22	±6 to ±70 VDC Input		ElTach-
23	±10 VDC Output		ElDrive
24	GND		
25	GND		

Notes:

1. I/O-62 lines can be configured in **softplane.conf** in groups of 4 for the following options:
   - RS422 differential vs single–ended
   - Input or output sense
   - Input termination for single–ended lines, pull-up (...Term=1),
   - pull-down (...Term=-1) or un-terminated (...Term=0)
2. All RCP8 status variables (starting with "s") are terminated with pull–up's in the default **softplane.conf**.

3. Antenna pedestal tachometer inputs are adjusted by a pot on back of the Connector Panel. The tach and drive signals are not configured in **softplane.conf**

Table 37 J10 "SERIAL"

Pin	Electrical Specification	Comment
1	GND	
2	RS232C Rx	
3	RS232C Tx	
4	—	
5	GND	
6	—	
7	—	
8	—	
9	—	

Table 38 J11 "SERIAL"

Pin	Electrical Specification	Comment
1	GND	
2	RS232C Rx	Channel 0
3	RS232C Tx	Channel 0
4	RS232C Rx	Channel 1
5	GND	
6	RS232C Tx	Channel 1
7	-12VDC @ 50mA max regulated	Regulated power supply
8	+12VDC @ 50mA max	Regulated power supply
9	+5VDC @ 50mA max	Regulated power supply

Table 39 J12 "SD"

Pin	Electrical Specification	RVP8 Signal Name	RVP8 Signal Name
1	**Nominal 90V60Hz SynchroSignals**	RefEL+	RefEL+
2		RefEL-	RefEL-
3		SyEL1	SyEL1
4		SyEL2	SyEL2
5		SyEL3	SyEL3
6	GND		
7	**Nominal 90V60Hz SynchroSignals**	RefAZ+	RefAZ+
8		RefAZ-	RefAZ-
9		SyAZ1	SyAZ1
10		SyAZ2	SyAZ2
11		SyAZ3	SyAZ3
12	GND		

The pin numbers are embossed on the J12 plastic connector but can be hard to read by eye. Facing the backpanel connector the pin arrangement is:

1. `RefEL+`
2. `RefEL-`
3. `SynEL1`
4. `SynEL2`
5. `SynEL3`
6. `Ground`
7. `RefAZ+`
8. `RefAZ-`
9. `SynAZ1`
10. `SynAZ2`
11. `SynAZ3`
12. `Ground`

The mating plug is AMP 350735–1 using Amplatch pins 350547–1. The corresponding hood comes in two identical pieces: AMP 640717–1, along with #6 x 1/2" self-tapping screw. You must use two hoods and two screws per plug.

The following table lists the maximum RMS voltage that can be applied to the backpanel's Molex SYNCHRO connector for each value of plug-in SIP resistor. The AZ channel voltages are set by SIP S1, whereas S2 sets the

EL voltage levels. These resistors are socketed, and can be changed by removing the back cover of the IO62–CP panel

S1 or S2	Max Ref(RMS)	Max SS(RMS)
47K	56V	31V
68K	81V	45V
100K	118V	66V
150K	178V	99V
220K	261V	145V

Note that the Ref inputs have somewhat lower gain than the three s inputs. This is because the precision of the S/D angle conversion is affected primarily by the precision at which the three s voltages can be measured. The backpanel therefore biases the gains so that the s voltages can be made as large as possible, that is, without the Ref voltages first filling the A/D conversion range.

The appropriate resistor is the smallest value such that the maximum S-to-S voltage of the synchro (which is angle dependent) still fits within the table range. The reference voltage should then fit easily into its corresponding maximum range. Dont worry if it doesnt; the important thing is to match the s line voltages.

For example, a traditional 90Vrms 1:1 synchro would best use the 150K resistor, whereas a 105Vrms unit would require the 220K value. Note that you can check for proper A/D conversion levels of the synchro inputs using the help view menu of the RCP8.

NOTE The synchro voltage input feature is only available on Rev.B and higher backpanels. If you are running an RCP8 with a Rev.A backpanel and would like to switch to synchro inputs, Vaisala will be happy to upgrade your panel at no cost.

Table 40 RVP8 BNC Connector Pin Assignments

Ref Designator	Label	Electrical Specification	Signal Name
J13	TP1	5V 75Ohm	
J14	TRIG1	12V 75Ohm	Trigger[1]
J15	TRIG2	12V 75Ohm	Trigger[2]
J16	TP2	5V 75Ohm	
J17	TRIG3	12V 75Ohm	Trigger[3]
J18	TRIG4	12V 75Ohm	Trigger[4]

Table 41 RCP8 BNC Connector Pin Assignments

Ref Designator	Label	Electrical Specification	Signal
J13	TP1	5V 75Ohm	
J14	SPARE		
J15	SPARE		
J16	TP2	5V 75Ohm	
J17	SPARE		
J18	SPARE		

C.3 IFD Module (RVP8 Only)

The IFD module is a small metal box which can be mounted inside the receiver cabinet. The IFD is shown in Figure 54 on page 424 and Figure 55 on page 425. Cooling of the inside components is accomplished bydirect conduction to the case. It is desirable to place the module in an environment that allows external convective cooling.

The IFD is equipped with its own auto ranging power supply (110 to 240 VAC 50/60 Hz) which is mounted on the side of the IFD. On the other side of the IFD are two anti-aliasing filters. These analog filters must be specified for the radar IF frequency. The filters have an 8 MHz bandwidth centered about the IF frequency.

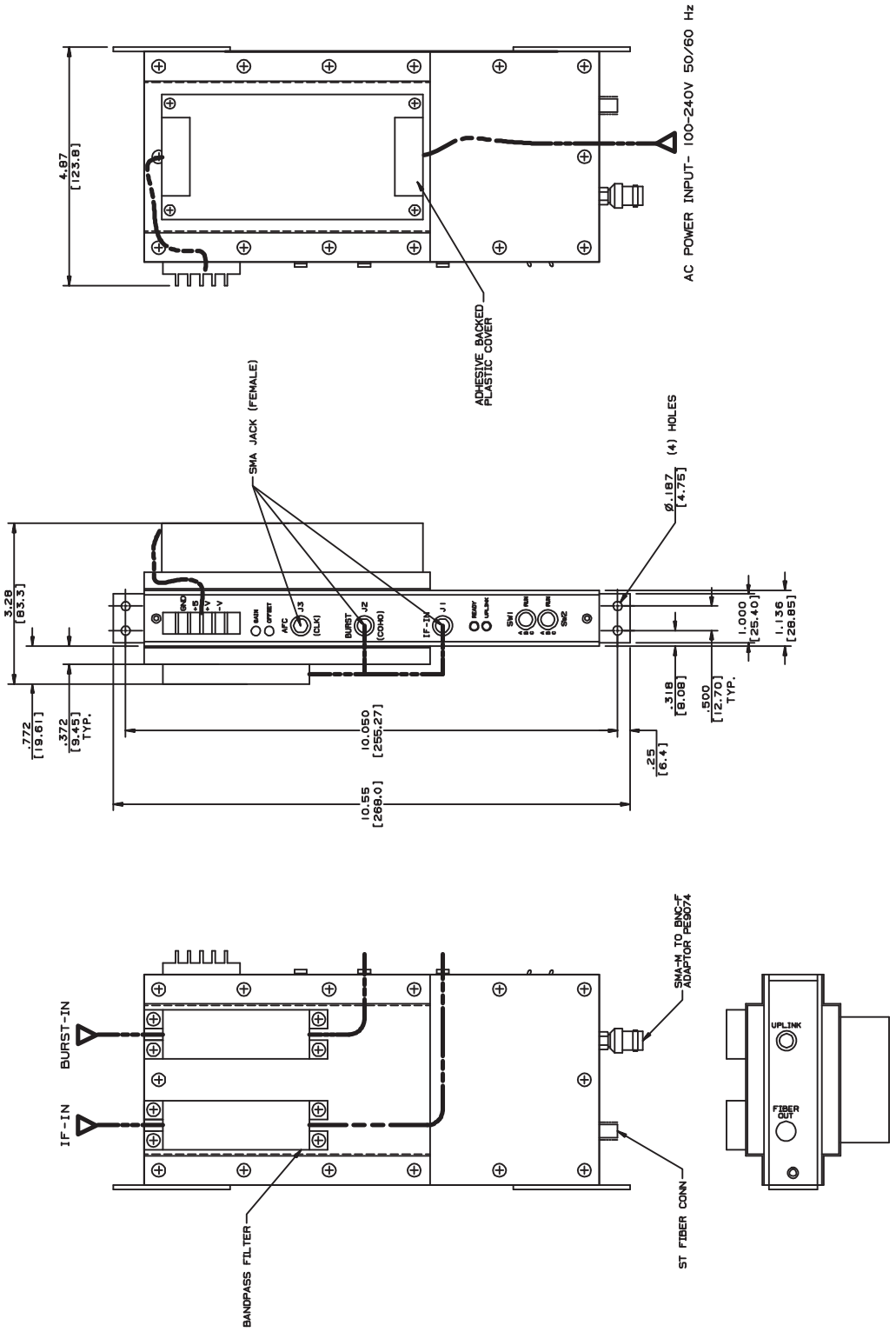


Figure 54 RVP8/IFD Module

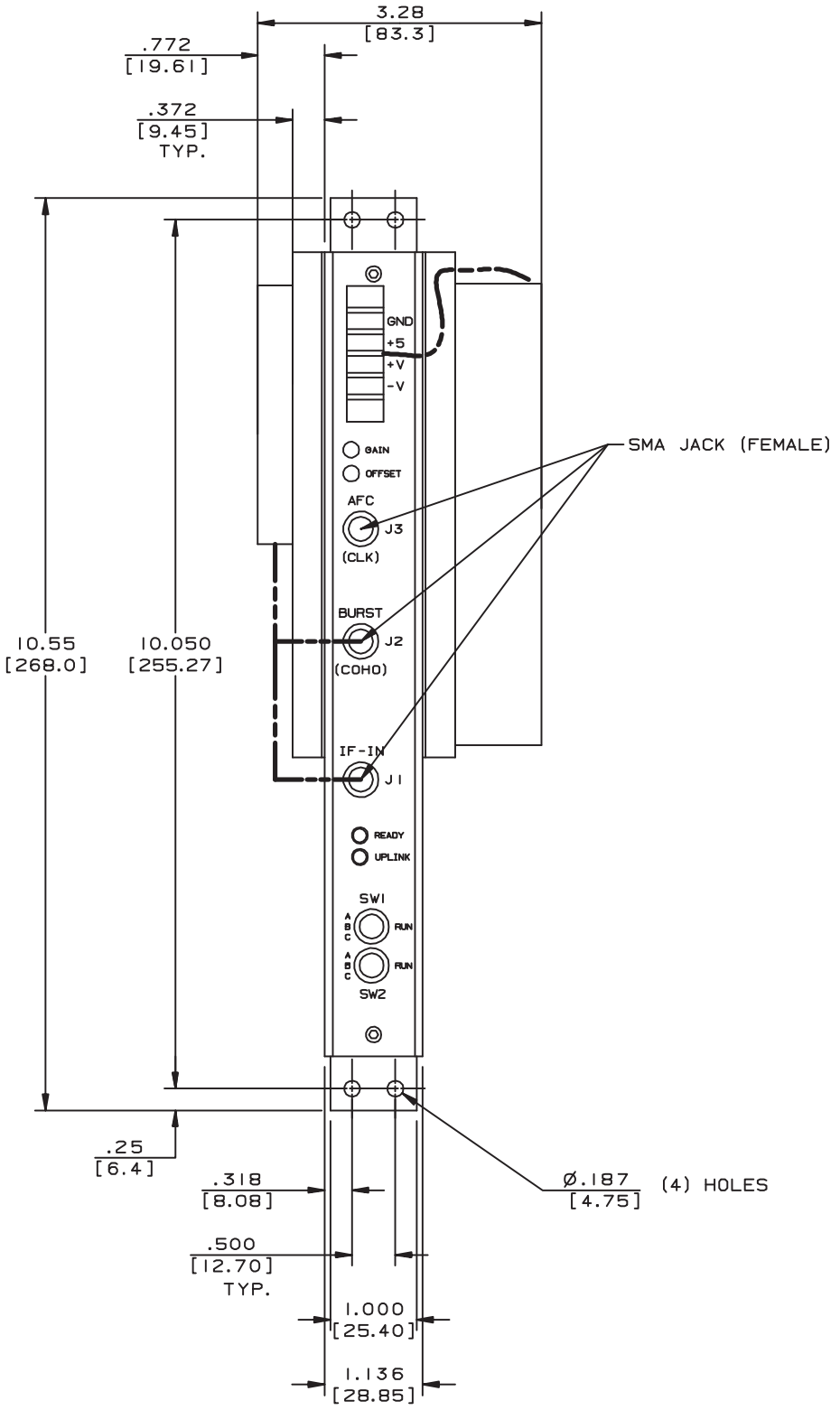


Figure 55 IFD Front Panel

C.4 DAFC Module (RVP8 only)

The Digital AFC (DAFC) module is used on RVP8 for magnetron systems to interface to a digitally controlled STALO. The DAFC "T's" off the coax uplink cable. Power can be provided by running discrete wires from the IFD. Note that +5 VDC is all that is required to run the DAFC. If you want to supply the STALO power over the ribbon cable to the IFD, you can connect the +24 VDC input to an appropriate power supply. Otherwise, you can power the STALO directly.

The DAFC outputs up to 24 TTL lines to the STALO digital control/ interface. Since these are TTL, the DAFC should be mounted within 10 to 30cm of the STALO if possible. For details on the DAFC, including pin assignment examples for some commercial STALO's, refer to the RVP8's Installation chapter.

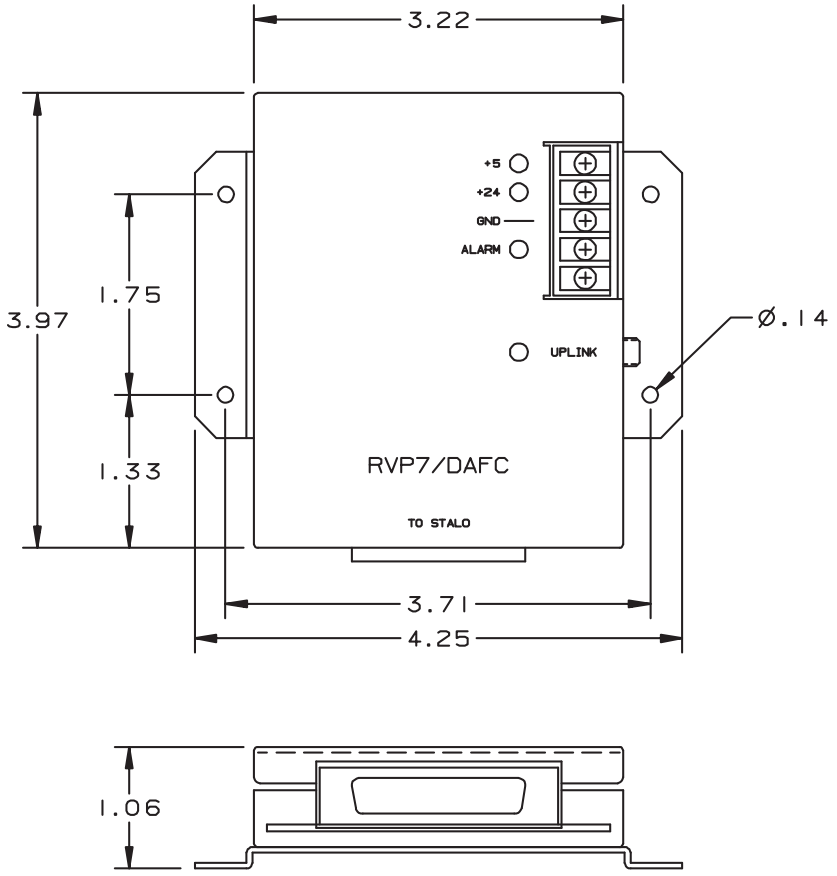


Figure 56 View of DAFC Module

APPENDIX D
INSTALLATION AND TEST PROCEDURE

Customer:		
Serial No.	**Main:**	
	IFD(s):	
Delivery Date:		
Radar Mfg./Type:		
Customer Engineer:		
Vaisala Engineer:		

Overview

This installation and test procedure is designed to assist Vaisala field engineers and customers with the installation and testing of the RVP8 on a radar system. Because the tests also function as an installation procedure, they must be completed in order. Failure to perform one step may effect later tests.

A copy of the test results should be kept either on file or with the *RVP8 User's Manual*. Do not write in the manual since it will be replaced with an upgrade, instead make a copy and mark that.

Each test should be performed and signed off when it is completed. If a test does not pass, then the problem should be remedied and the test repeated. If the test still does not pass, then an additional sheet should be added to the test explaining the variance. A supplementary test sheet is at the end of the test procedure.

After you have successfully completed the installation steps and tests in this procedure, your RVP8 will be ready for connection to your application software such as Vaisala IRIS system. There will be additional configuration and calibration to use the RVP8 with your application software.

The following page is a convenient summary check list for the tests. Use this as a check list for completing the installation and test procedure. We hope that you enjoy your new RVP8. Please contact us if you have any problems or comments regarding this product or procedure at helpdesk@vaisala.com.

Test Summary:

- D.1 Installation Check
- D.2 Power-up Check
- D.3 Setup Terminal
- D.4 Setup **V** Command (Internal Status)
- D.5 Setup **M** Command (Board Configuration)
- D.6 Setup **Mp** Command (Processing Options)
- D.7 Setup **Mf** Command (Clutter Filters)
- D.8 Setup **Mt** Command (General Trigger Setup)
- D.9 Initial Setup of Information for Each Pulse Width
- D.10 Setup **Mb** Command (Burst Pulse and AFC)
- D.11 Setup **M+** Command (Debug Options)
- D.12 Setup **Mz** Command (Transmitter Phase Control)
- D.13 Display Scope Check
- D.14 Burst Pulse Alignment
- D.15 Bandwidth Filter Adjustment
- D.16 Digital AFC Voltage Alignment
- D.17 Analog AFC Voltage Alignment
- D.18 MFC Functional Test
- D.19 AFC Functional Test
- D.20 Input IF Signal Level Check
- D.21 Dynamic Range Check
- D.22 Receiver Bandwidth Check
- D.23 Receiver Phase Noise Check
- D.24 Hardcopy of Final Setups
- D.25 IFD SigGen Test

- D.26 RVP8/Tx Card Test

All Tests Passed

For Customer _____ Date _____

For Vaisala _____ Date _____

D.1 Installation Check

Test Goal

Verify that the RVP8 is properly connected to the radar system and document some of the basic radar characteristics. There are differences for TWT/Klystron vs magnetron radar systems.

WARNING	These are factory configured for the expected voltage, but should be VERIFIED by the customer before power is applied to the system.

AC Line Voltage _____ VAC

- Check the 3 power supply modules for proper configuration per C.1.3 Main Chassis Back Panel Power Section on page 401.

Test Procedure

- IF Digitizer (IFD) mounted in the radar receiver cabinet or other convenient location.
- IF Digitizer Power Supply properly connect to AC Line voltage.
- IFD IF input connection. IF Frequency _____MHz.
- RVP8 Chassis installed in (circle one) Rack Tabletop
- Distance between IFD and RVP8 Chassis _____
- Downlink connection from IFD to RVP8 Chassis.
- Uplink connection from IFD to RVP8 Chassis.
- Trigger connections. List triggers in D.8 Setup Mt Command (General Trigger Setup) on page 434

For magnetron systems, verify the following:

- IFD IF burst pulse connected.

STALO AFC is done by (circle one) Analog connection to IFD DAFC

Other

_____

For Klystron or TWT systems only verify the following:

- IFD COHO connected to Burst Input.

Tx Card Installed (circle one) Yes No

Verify that Vaisala cards are functioning:

- As root type "service rvp8 stop"
- As operator type "rdadiags rvp8rx0", "rdadiags io62-0", "rdadiags io62cp-0" (note any messages and refer to RVP8 A.3.3)

Test Passed

For Customer _____ Date _____

For Vaisala _____ Date _____

D.2 Power-Up Check

Test Goal

Verify that the IFD and RVP8 properly powerup.

Reference:*RVP8 User's Manual*3.3.6 Power-Up Details on page 94

Test Procedure

The front panel of the RVP8 shows summary powerup messages. Apply power to the RVP8 (or reset it) and the IFD. Verify the following:

For the IFD with RVP8 power on:

- When power is applied to the IFD, the red and green lights blink and then go to on.
- When the uplink is disconnected, the red light blinks and the green light is off.
- When the fiber cable is disconnected, the red light is on and the green light is off.

For the RVP8 with the IFD power on:

- The front panel display shows the correct text per 3.3.6 Power-Up Details on page 94 indicating successfulbootup.

Test Passed

For Customer _____ Date _____

For Vaisala _____ Date _____

D.3 Setup Terminal

Test Goal

Verify that the TTY Setups can be accessed and function properly.

Special Test Equipment

Keyboard, mouse and monitor (KVM) installed locally or on remote computer (with DspExport running).

Reference:*RVP8 User's Manual,*4.1 Overview of Setup Procedures on page 114

Test Procedure

Follow the procedure in 4.1 Overview of Setup Procedures on page 114 to access the TTY setups.

- As operator, type "dspx"
- Press the ESC key and verify that the RVP8 banner appears (see 4.1 Overview of Setup Procedures on page 114).
- Type **Help** or **?** and verify that the list of commands is displayed.
- Type **q** or **quit** to exit the menus.

Test Passed

For Customer _____ Date _____

For Vaisala _____ Date _____

D.4 Setup V Command (Internal Status)

Test Goal

Verify that the TTY setups for the Internal Status section are properly reported.

Special Test Equipment: KVM installed

Reference: *RVP8 User's Manual*4.1.2 V — View Card and System Status on page 116

Test Procedure

Enter the TTY setups via dspx and issue the **V** command to display the internal status. Note that we will record the final values of all the settings at the end of the installation.

- Status information is correct per 4.1.2 V — View Card and System Status on page 116 and shows no faults.

Test Passed

For Customer _____ Date _____

For Vaisala _____ Date _____

D.5 Setup Mc Command (Board Configuration)

Test Goal

Verify that the TTY setups for the Board Configuration section are properly configured for the customer application.

Special Test Equipment: KVM connected

Reference: *RVP8 User's Manual*, 4.2.1 Mc — Top Level Configuration on page 120

Test Procedure

Enter the TTY setups and type the **Mc** command. Set all the values as required for your operation.

- Parameters set.

Test Passed

For Customer _____ Date _____

For Vaisala _____ Date _____

D.6 Setup Mp Command (Processing Options)

Test Goal

Verify that the TTY setups for the Processing Options section are properly configured for the customer application.

Special Test Equipment: KVM connected

Reference: *RVP8 User's Manual*, 4.2.2 Mp — Processing Options on page 122

Test Procedure

Enter the TTY setups and type the **Mp** command. Set all the values as required. for your operation.

- Parameters set.

Test Passed

For Customer _____ Date _____

For Vaisala _____ Date _____

D.7 Setup Mf Command (Clutter Filters)

Test Goal

Verify that the TTY setups for the Clutter Filters section are properly configured for the customer application.

Special Test Equipment: KVM connected

Reference: *RVP8 User's Manual*, 4.2.3 Mf — Clutter Filters on page 127

Test Procedure

Enter the TTY setups and type the **Mf** command. Set all the values as required for your operation.

- Parameters set.

Test Passed

For Customer _____ Date _____

For Vaisala _____ Date _____

D.8 Setup Mt Command (General Trigger Setup)

Test Goal

Verify that the TTY setups for the General Trigger Setup section are properly configured for the customer application.

Background

The RVP8 can output up to 6 different triggers. These can be delayed by different amounts, and have different pulse widths. For example trigger 0 may go to fire the transmitter, while a slightly delayed trigger 1 may be used for triggering an oscilloscope. The timing can be different for each transmitter pulsewidth. The final timing adjustments will be done later in D.14 Burst Pulse Alignment on page 439 Enter in the table below the purpose of each trigger, as well as the nominal start time and pulse width. Note that start times are relative to range zero (middle of the burst pulse). We recommend a nominal pulsewidth of 3 microseconds. This chart will be used in the next section.

Magnetron radars using an analog COHO system often have a trigger generator circuit which produces a trigger for the COHO latching and also for the transmitter pulse. This circuit should be bypassed in an upgrade.

#	Purpose	Start Time	Width
0	_____	_____usec	_____usec
1	_____	_____usec	_____usec
2	_____	_____usec	_____usec
3	_____	_____usec	_____usec
4	_____	_____usec	_____usec
5	_____	_____usec	_____usec

Special Test Equipment: KVM connected

Reference:*RVP8 User's Manual*, 4.2.4 Mt — General Trigger Setups on page 130

Test Procedure

Enter the TTY setups and type the Mt command. Set all the values as required for your operation. Note that the PRF and pulse width set here are the current values, and values used at power-up.

- Parameters set.

Test Passed

For Customer _____ Date _____

For Vaisala _____ Date _____

D.9 Initial Setup of Information for Each Pulse Width

Test Goal

Enter the initial values for the TTY Setups for each of the pulse widths. Note that the final values of trigger timing, FIR filter impulse response length and bandwidth will be adjusted later.

Background

The duty cycle of the transmitter is the product of the PRF and the pulse width in seconds. For example, a PRF of 1000 Hz and 1 microsecond pulse width is a duty cycle of 0.001. Thus a transmitter with a 0.001 duty cycle limit could function at 1000 Hz and 1 microsecond pulse width, or 500 Hz and 2 microsecond pulse widths.

The duty cycle limits of your radar should be obtained from your system documentation or radar manufacturer. The RVP8 supports up to four pulse widths (coded 0 to 3), although most transmitters typically support only two pulse widths. Record below the pulse width in microseconds and the maximum PRF that is allowed for each pulse width.

#	Pulse Width	Max PRF
0	_____ microseconds	_____ Hz
1	_____ microseconds	_____ Hz
2	_____ microseconds	_____ Hz
3	_____ microseconds	_____ Hz

Special Test Equipment: KVM connected

Reference:*RVP8 User's Manual*, 4.2.5 Mt<n>— Triggers for Pulsewidth #n on page 133

Test Procedure

Enter the TTY setups and type the Mt # command, once for each pulsewidth. Enter the start time and widths for each trigger as documented on the previous page. For all unused triggers, set the width to zero. Next enter the Maximum PRF from the table above. Set the initial impulse response length to 1.5 times the pulsewidth, and the initial pass bandwidth to the inverse of the pulsewidth.

- Parameters set.

Test Passed

For Customer _____ Date _____

For Vaisala _____ Date _____

D.10 Setup Mb Command (Burst Pulse and AFC)

Test Goal

Verify that the TTY setups for the Burst Pulse and AFC Configuration section are properly configured for the customer application.

Background: Magnetron vs Klystron Systems

Magnetron Systems: For magnetron systems, the phase and frequency of the burst pulse from the transmitter is measured at IF. The phase measurement is used for digital phase locking and 2nd trip echo filtering and recovery. The frequency measurement is used to implement an analog (+10V) AFC output to control the STALO frequency. Note that an external AFC can be used rather than the RVP8 AFC, but is not recommended.

Klystron or TWTbased Systems: The COHO is measured instead of the burst pulse. Note that Klystron systems that use a phase shifter should input the phase shifted COHO into the IFD so that the RVP8 can digitally lock to the actual transmitted phase. For Klystron systems the AFC feedback loop is not used.

Special Test Equipment: KVM connected

Reference:*RVP8 User's Manual*, 4.2.7 Mb — Burst Pulse and AFC on page 142

Test Procedure

Enter the TTY setups and type the **Mb** command. Set all the values as required.

- Parameters set.

Test Passed

Test Passed

For Customer _____ Date _____

For Vaisala _____ Date _____

D.11 Setup M+ Command (Debug Options)

Test Goal

Verify that the TTY setups for the Debug Options section are properly configured for the customer application.

Special Test Equipment: KVM connected

Background

The RVP8 supports several test features that are configured in this section. For operational systems, the simulation features should be turned off. Vaisala recommends that the LED's be set to "1:Go/Proc" so that the front panel red LED will flash during each processing cycle.

Reference:*RVP8 User's Manual*, 4.2.9 M+ — Debug Options on page 153

Test Procedure

Enter the TTY setups and type the **M+** command. Set all the values as required.

- Parameters set.

Test Passed

For Customer _____ Date _____

For Vaisala _____ Date _____

D.12 Setup Mz Command (Transmitter Phase Control)

Test Goal

Verify that the TTY setups for the Transmitter Phase Control section are properly configured for the customer application. This feature is not used for magnetron systems since these have inherent random phase that is measured, but not controlled.

Special Test Equipment: KVM connected

Reference:*RVP8 User's Manual*, 4.2.10 Mz — Transmissions and Modulations on page 155

Test Procedure

Enter the TTY setups and type the **M+** command. Set all the values as required.

- Parameters set.

Test Passed

For Customer _____ Date _____

For Vaisala _____ Date _____

D.13 A-Scope Test

Test Goal

Verify that the display ascope utility functions properly.

Background

The ascope utility provides a completely independent radar control and plotting capability. This is used extensively for testing of the radar and the RVP8.

Reference:*IRIS Utilities Manual*, Chapter 4, TTY Nonvolatile Setups, on page 113.

Special Test Equipment:

KVM connected

Test Procedure

- As operator, type the command **ascope** and verify that the ascope utility comes up cor-rectly and starts to update.
- Configure a "DEFAULT" startup and save it. Then exit and restart ascope. Verify that the default startup is properly restored.

Test Passed

For Customer _____ Date _____

For Vaisala _____ Date _____

D.14 Burst Pulse Alignment

Test Goal

Verify that the burst pulse is present and that its amplitude is sufficient. This test also aligns the burst pulse in the burst pulse sample window.

Special Test Equipment: KVM connected

Reference: *RVP8 User's Manual*, 5.3 Pb — Plot Burst Pulse Timing on page 163

Test Procedure

- Use ascope to select the desired pulsewidth at a save PRF.
- In the TTY setups, use the Mb command to turn off burst pulse tracking.
- Set the transmitter to radiate.
- Issue the Pb command to obtain the burst pulse display. Use the L/R commands to find the burst pulse. Use the l/r commands to fine tune the position of the burst pulse in the burst pulse window.
- Adjust the width of the burst pulse window using the I/i command to be slightly larger than the burst pulse (for example, ~50%).
- Verify that the burst pulse power is in the range +1 to -12 dBm per the tabular display on the setup terminal.

  Record the burst pulse power

  Pulsewidth #0: _____ dBm

  Pulsewidth #1: _____ dBm

  Pulsewidth #2: _____ dBm

  Pulsewidth #3: _____ dBm
- Repeat the above procedure for each pulse width.

In the event that the burst pulse is not found, try to detect the burst pulse on an oscilloscope connected directly to the IF burst line (ahead of the IFD). On a magnetron radar, if the AFC is not working it is possible the IF frequency is outside the IFD anti-aliasing filter bandwidth. It may be necessary to go to manual frequency control to get this to work. If no burst pulse is detected, then the radar should be serviced by an experienced technician. If the burst pulse is power is too small or large, check the status of any attenuators or amps in the burst pulse signal path. It might be necessary to adjust the gain buy installing a fixed attenuator or amplifier.

Test Passed

For Customer _____ Date _____

For Vaisala _____ Date _____

D.15 Bandwidth Filter Adjustment

Test Goal

Set the band width filter for each pulse width.

Reference: *RVP8 User's Manual*, 5.4 Ps — Plot Burst Spectra and AFC on page 168

Special Test Equipment: KVM connected

Test Procedure

- Enter the Ps command mode and view the results on the display scope. Toggle the space bar to show both the spectrum of the burst pulse and the spectrum of the bandwidth filter response. Use the Z/z command to zoom the burst spectrum plot to approximately match the height of the bandwidth filter response (which will have a smoother shape than the burst pulse).
- Use the Ww/Nn commands to adjust the width of the bandwidth filter plot to be slightly narrower than the burst pulse. Then use the w/n commands to fine tune the filter width such that the "DC–Gain:" is either "ZERO" or less than -64 dB.
- Repeat for each pulse width that is used (use mt to change pulse width) and record:

	FIR Length	Bandwidth	DC Gain
Pulsewidth 0	_____usec	_____MHz	_____ dB
Pulsewidth 1	_____usec	_____MHz	_____ dB
Pulsewidth 2	_____usec	_____MHz	_____ dB
Pulsewidth 3	_____usec	_____MHz	_____ dB

Test Passed

For Customer _____ Date _____

For Vaisala _____ Date _____

D.16 Digital AFC (DAFC) Alignment (Optional)

Test Goal

Verify that the RVP8 DAFC output controls the STALO over the correct span.

Special Test Equipment: Setup TTY

Reference:*RVP8 User's Manual*, 3.4 Digital AFC Module (DAFC) on page 99

Background

The RVP8 implements an AFC based on the measurement of the burst pulse frequency. The DAFC "T's" off the coaxial uplink cable. It translates the AFC requests from the RVP8 main chassis into digital output requests supporting up to 25 bits. A frequency control span of approximately + -7 MHz is expected.

Document the STALO frequency that is desired. This is typically the RF frequency minus 30 or 60 MHz depending on the IF of your system.

RF Transmit frequency _____MHz

STALO Frequency _____MHz

Test Procedure

- Use the setup terminal to set the Digital AFC span as required in the Mb section.
- Use the setup terminal and display scope in the Pb (plot burst) mode to verify that the burst pulse is properly centered. Any pulsewidth can be used.
- Set to MFC using the command, and adjust the control to the lowest setting using the **D** command. Record the results below.
- Raise the control using "U" to within 0.1 MHz of the IF frequency. Record the results.
- Raise the control using "U" to the highest setting. Record the results.

- Verify that sufficient span is covered, and the the power at the end points is sufficiently high to run the AFC loop.

	voltage	frequency
Midpoint:	_____A/D	_____ MHz.
Lower limit:	_____A/D	_____ MHz.
Upper limit:	_____A/D	_____ MHz.

Test Passed

For Customer _____ Date _____

For Vaisala _____ Date _____

D.17 Analog AFC Voltage Alignment (Optional)

Test Goal

Verify that the RVP8 AFC output voltage is properly adjusted to match the STALO input control voltage.

Special Test Equipment

Calibrated Oscilloscope or Voltmeter.

Setup TTY

Background

The RVP8 implements an AFC based on the measurement of the burst pulse frequency. The analog control output is an SMA labeled "AFC" on the IFD module which connects to the STALO control voltage input. The output signal is an analog voltage in the range +-10V. A frequency control span of approximately +-7 MHz (for a 30 MHz IF) is expected. Some STALOs contain a nominal frequency adjustment. The alignment procedure is different in that case.

Check the specification of the STALO and verify the following:

STALO analog control input range _____V to _____V

STALO frequency control range ± _____ MHz

Test Procedure, initial setup

- Connect a scope or digital Voltmeter to the AFC line, either at the IFD or the STALO.
- Use the setup terminal and display scope in the Pb (plot burst) mode to verify that the burst pulse is properly centered. Any pulsewidth can be used.
- Set the test switches on the IFD to output the Midpoint Voltage

SW1–A SW2–B

Test Procedure, STALO without adjustment

- Adjust the "Offset" pot by screwdriver on the IFD module until the the IF frequency dis-play on the setup terminal is approximately the desired IF frequency (for example, 30 MHz). Re-cord the results below.
- Set the test switches on the IFD to output the AFC low test voltage.

SW1–A SW2–A

The voltage on the monitoring scope or Voltmeter will decrease. The burst pulse frequency may either increase or decrease depending on the nature of the voltage control in the STALO.

- Adjust the "Gain" pot by screwdriver on the IFD module until the frequency stops changing or until the burst pulse frequency is 7 MHz off the center frequency (for example, for 30 MHz IF, either 23 or 37 MHz) whichever occurs at a higher voltage. Record the results.
- Set the test switches on the IFD to output the AFC high test voltage.

SW1–A SW2–C

- Reduce the gain slightly (CCW turn) and verify that the frequency changes such that it becomes closer to the center IF frequency. If it does not change, then continue to reduce the gain until it does. If the frequency is more than 7 MHz from the IF center frequency, then reduce the gain until the frequency is 7 MHz off. Record the results below.

Test Procedure, STALO with adjustment

- Adjust the "Offset" pot by screwdriver on the IFD module until the voltage is at the middle of the desired voltage span.
- Adjust the "Offset" pot on the STALO until the the IF frequency display on the setup terminal (Pb mode) is approximately the desired IF frequency (for example, 30 MHz). Record the results below.

- Set the test switches on the IFD to output the AFC low test voltage.

  SW1–A SW2–

  A The voltage on the monitoring scope or Voltmeter will decrease. The burst pulse frequency may either increase or decrease depending on the nature of the voltage control in the STALO.
- Adjust the "Gain" pot by screwdriver on the IFD module until low end of the desired voltage span is reached, or until the burst pulse frequency is 7 MHz off the center fre-quency (for example, for 30 MHz IF, either 23 or 37 MHz) whichever occurs at a higher voltage. Record the results below.
- Set the test switches on the IFD to output the AFC high test voltage.

  SW1–A SW2–C
- Reduce the gain slightly (CCW turn) and verify that the frequency changes such that it becomes closer to the center IF frequency. If it does not change, then continue to reduce the gain until it does. If the frequency is more than 7 MHz from the IF center frequency, then reduce the gain until the frequency is 7 MHz off. Record below.

Test Procedure, final cleanup

- Set the switches back to the run position (SW1–B and SW2–B), and disconnect the T for the Voltmeter or scope monitor. This cable can introduce a lot of noise into the system.

	voltage	frequency
Midpoint:	_____Volts	_____ MHz.
Lower limit:	_____Volts	_____ MHz.
Upper limit:	_____Volts	_____ MHz.

Test Passed

For Customer _____ Date _____

For Vaisala _____ Date _____

D.18 MFC Functional Test and Tuning (Optional)

Test Goal

Verify that the Manual Frequency Control (MFC) is functioning properly. Skip this test if you are not using the RVP8's AFC.

Reference: *RVP8 User's Manual*, 5.4 Ps — Plot Burst Spectra and AFC on page 168

Special Test Equipment: KVM connected

Test Procedure

Enter the Ps command (Plot burst spectrum and AFC).

- Use the = command to enter the MFC (manual frequency control) mode. Verify that the MFC mode is indicated by the "Manual" notation next to the AFC % output indicator on the terminal.
- Use the U/u and D/d commands and verify that these commands shift the measured IF frequency (as displayed on the TTY) either up or down. The U command should increase the frequency and the D command should decrease the frequency. If the sense is re-versed, then go to the Mb command menu and change the question "Burst frequency in-creases with increasing AFC voltage".
- Using the U/u and D/d commands, verify the limits of the AFC tuning and fill in the table below:

AFC %	Measured Freq (MHz)
100%	_____
0%	_____
+100%	_____

The 0% AFC value should be within approximately +-0.2 MHz of the center IF frequen-cy (for example, 30 MHz) . The values at +- 100% should correspond to approximately +- 7 MHz of the center IF frequency, or at the maximum span that is supported by the STALO, whichever is less.

- Toggle the MFC mode to AFC by typing the "=" symbol and and verify that the terminal indicator changes from "Manual" to "AFC". Then exit the Ps menu.

Test Passed

For Customer _____ Date _____

For Vaisala _____ Date _____

D.19 AFC Functional Test (Optional)

Test Goal

Verify that the AFC properly tracks the burst pulse frequency.

Reference:*RVP8 User's Manual*, 5.4 Ps — Plot Burst Spectra and AFC on page 168

Special Test Equipment: KVM connected

Test Procedure

Use the setup terminal to enter the Ps mode and observe the output on a display scope. Verify the following:

- Verify that the system is in AFC mode by checking that the text on the terminal for the AFC % output says "AFC".
- Verify that the frequency displayed on the setup terminal is within +-15KHz of the center IF frequency (the default value for the AFC hysteresis outer limit in the Mb com-mand). For example in the range 29.985 to 30.015 MHz. If it is not in this range then verify that it moves within this range.
- Turn radiate off for 10 minutes and then turn the radiate back on. Observe that the AFC properly tracks the magnetron frequency as the magnetron warms.
- Similarly, set the control signal to the maximum and minimum values using MFC, then turn on AFC. Observe that the AFC properly tracks back to the correct frequency.
- Perform the tests above for each pulse width and verify that the AFC properly tracks the center frequency.
- For pulsewidth 0.
- For pulsewidth 1
- For pulsewidth 2.
- For pulsewidth 3.

Test Passed

For Customer _____ Date _____

For Vaisala _____ Date _____

D.20 Input IF Signal Level Check

Test Goal

Verify that the input signal level is optimized for the IFD. This is done be observing the power in the noise using the Pr command.

Reference:*RVP8 User's Manual*, 5.5 Pr — Plot Receiver Waveforms on page 183

Special Test Equipment: KVM connected

Test Procedure

- Set the transmitter to radiate and elevate the antenna to >45 degrees to minimize the ef-fects of weather or clutter echoes (including earth noise). Be sure the antenna azimuth is pointed away from the sun or any known RF interference sources you may have.

NOTE	This entire procedure may also be performed with the transmitter off since, in theory, it is only measuring properties of the receiver. However, you may notice some noise interaction between the Tx and Rx.

- Use the setup terminal to enter the **Pr** command and the display scope to view results. Use the **Ll/Rr** commands to move out in range to a start range of 50 km so that only noise is present.

  Record the powers displayed on the setup terminal. You can use the V/v command to increase/decrease averaging of samples to make the noise measurement more stable. Total: _____ dBm, Filtered: _____ dBm
- Now remove the cable connecting the IF signal into the IFD. Again record the powers: Total: _____ dBm, Filtered: _____ dBm
- Add attenuation and/or amplification by an amount such that the Filtered noise power is approximately 6 dB higher when the signal is connected (See 3.2.8 IF Gain and System Performance on page 75).
- After verifying the above rise in noise level, disconnect the output cable from the LNA and verify that the noise drops to the same level as when the IFD IF-Input was discon-nected. This verifies that the

dominant noise is indeed coming from the LNA, and not from any of the subsequent IF amplifiers.

Test Passed

For Customer _____ Date _____

For Vaisala _____ Date _____

D.21 Dynamic Range Check

Test Goal

Verify the receiver dynamic range is in excess of 80 dB.

NOTE	This test requires the injection of an RF test signal over a 90 dB range. Damage to the LNA could occur. Check the LNA specification to verify the maximum signal that it can accept. The output from the signal generator (accounting for cable and coupler losses) should not be allowed to exceed this value.

Reference: *RVP8 User's Manual* 5.5 Pr — Plot Receiver Waveforms on page 183

Special Test Equipment:

KVM connected

RF signal generator

Test Procedure

- Run the radar and test signal generator for 20 minutes to allow proper warm-up of the system prior to the test. This will allow the AFC to stabilize.
- After warm–up is complete, turn the radiate off but leave the receiver on since the test signal generator may be damaged by the transmitter. The antenna should be elevated to 20 degrees and the azimuth should be set to point away from any known microwave sources including the sun.
- Use the setup terminal to enter the Mt command to set the pulse width to 0.
- Use the Mt 0 command to temporarily configure the FIR impulse response to 2.89 usec (**I/i** command) and 0.59 MHz bandwidth (N/n and W/w commands). These settings are for the purpose of benchmarking the receiver performance. Do not save this result since

it would override your previously configured band width and impulse response.

- Connect the test signal generator to inject a signal at RF ahead of the LNA.
- Enter the Pr mode and make the following settings:

  Use the space bar to toggle to the power spectrum plot.

  Use the **L/l** and **R/r** commands to set the start to 50 usec.

  Use the **T/t** command to set the plot span to 50 usec

  Use the **V/v** command to set averaging to 10 samples
- Set the signal generator to a value that is approximately 20 dB above noise and observe the scope plot. Adjust the frequency of the test signal generator to make the frequency of the spectrum at the correct IF frequency.

  Turn off the signal generator RF output and record the "Filtered" noise power Siggen power: none RVP8 Filtered power:_____dBm Turn on the signal generator RF with about 20 dB of signal above noise. Now reduce the power until you the Filtered power is approximately 1 dB above the noise level measured in the previous step. Verify this by toggling the signal generator RF ON and OFF. The samples will be a little noisy, but getting the signal exactly 1 dB above noise is not required. Record the signal generator setting for the 1 dB above noise power (minimum detectable power).

  Siggen power (Pmin):_____dBm RVP8 Filtered power:_____dBm
- Increase the signal generator output power by 10 dB steps until saturation of the Filtered power is observed.

NOTE	Do not increase the signal generator power such that Total Power exceeds the Safe Total Power Limit for Pr command display +10 dBm or damage to the RVP8 A/D convertor could result.

Back off the siggen power to approximately 10 dB below saturation. Now use 1 dB steps to more carefully define the saturation point to within +-1 dB (for example, for a 0.2 dB roll off). Record the signal generator setting.

Siggen power (Psat):_____dBm RVP8 Filtered power:_____dBm

The Receiver dynamic range is:

_____dB = Psat Pmin

- Verify that the receiver dynamic range is greater than or equal to 80 dB.

- Check that the signal generator frequency has not drifted by looking at the plot. If it is off by more than 0.1 MHz, retune and repeat the test.
- Exit Pr and do a restore to "restore" the saved settings.

Test Passed

For Customer _____ Date _____

For Vaisala _____ Date _____

D.22 Receiver Bandwidth Check

Test Goal

Verify the receiver bandwidth is in excess of 14 MHz.

Background

For proper functioning of the high speed A/D convertors, it is necessary that approximately 14 MHz of broadband noise is available at the IFD. This noise does not interfere with the signal to noise ratio because the bandwidth filter is applied afterwards. The bandwidth of the anti-aliasing filter should be the limiting factor. This test uses the same hookup as the previous test (D.21 Dynamic Range Check on page 448). For dual polarization systems, you expect to get a narrower bandwidth.

Reference: *RVP8 User's Manual*, 5.5 Pr — Plot Receiver Waveforms on page 183

Special Test Equipment:

KVM connected

RF signal generator

Test Procedure

- Connect the test signal generator to inject a signal at RF ahead of the LNA.
- Enter the Pr mode and make the following settings:

  Use the space bar to toggle to the power spectrum plot.

  Use the **L/l** and **R/r** commands to set the start to 50 usec.

  Use the **T/t** command to set the plot span to 50 usec

  Use the **V/v** command to set averaging to 1 sample
- Set the signal generator power to a value that is approximately 60 dB above noise and observe the scope plot. Adjust the frequency of the

test signal generator in 1 MHz steps to cover the whole range of the scope plot. Mark the total power measured on the plot below.

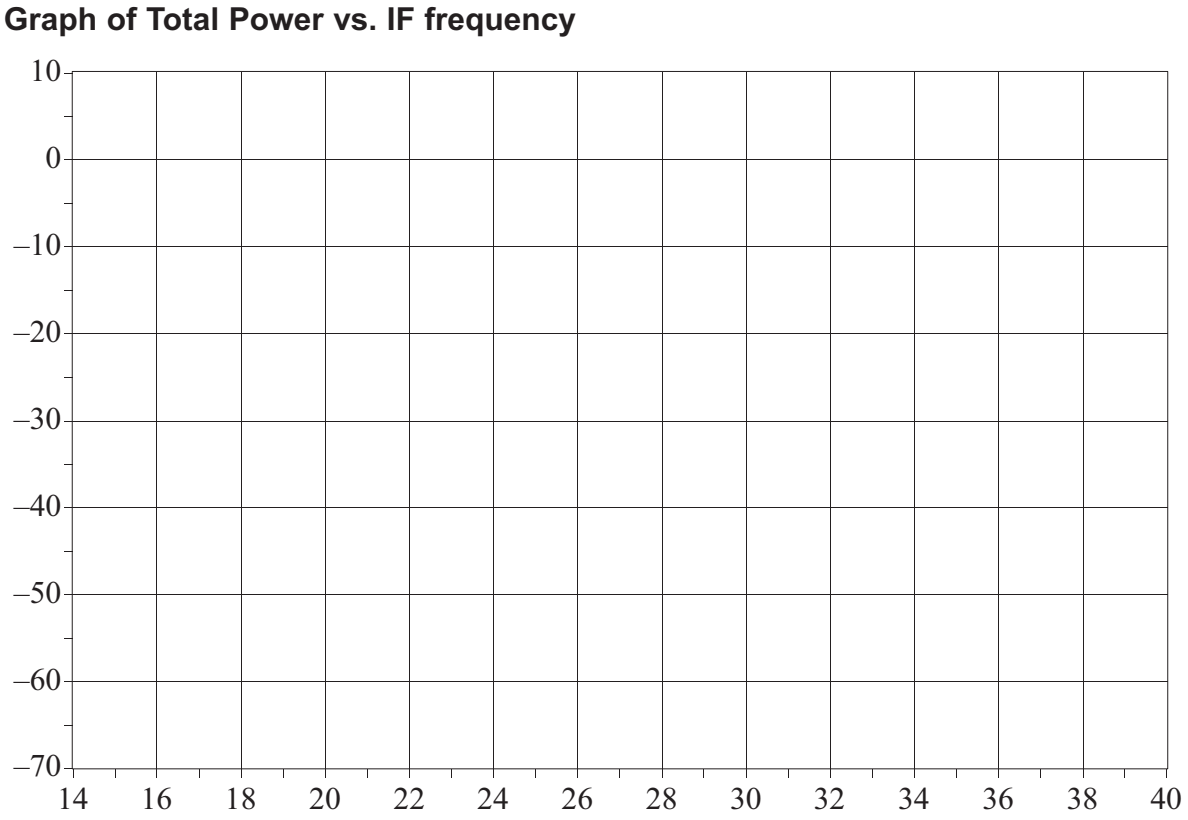
- Verify that the 3dB point gives approximately 14 MHz of bandwidth.

Test Passed

For Customer _____ Date _____

For Vaisala _____ Date _____

Graph of Total Power vs. IF frequency



D.23 Receiver Phase Noise Check

Test Goal

Verify the stability of the STALO by looking at the phase noise of a clutter target.

Background

For proper velocity calculations and for ground clutter rejection, it is required that the radar's STALO maintain a stable frequency, and that the transmitted pulse contain no amplitude or phase artifacts.

Special Test Equipment:

ascope utility per IRIS Utilities Manual Section 3.7.3

Known clutter targets with no weather signal

Test Procedure

- Configure the radar for normal operation expect pointing at a known clutter target.
- Run the ascope utility, and configure as follows: 16–bit time series, Spectrum not from DSP, spectrum size 256, Rectangular window, short pulse width, high PRF, no clutter filters. Select the maximum range and number of bins to get the maximum resolution over the target. For targets <24 km, use 201 bins, 25 km. Select the range bin of the target. Record the Az, El, range and phse noise below.
- Try minor changes in Az, El, and Range to get the lowest phase noise. The goal is less than 1 degree within 20 km.

Az: _____	El:_____	Range:_____	Phase Noise:_____
Az: _____	El:_____	Range:_____	Phase Noise:_____
Az: _____	El:_____	Range:_____	Phase Noise:_____
Az: _____	El:_____	Range:_____	Phase Noise:_____
Az: _____	El:_____	Range:_____	Phase Noise:_____
Az: _____	El:_____	Range:_____	Phase Noise:_____

Test Passed

For Customer _____ Date _____

For Vaisala _____ Date _____

D.24 Hardcopy and Backup of Final Setups

Test Goal

Make a hardcopy of all the final setups, and attach to the tests.

Special Test Equipment:

KVM connected,

Printer or ftp access to a computer that supports a printer.

TTY Setups Hardcopy Listing

- Start script logging with commands "cd /usr/sigmet/config/listings", "script RVP8.26feb01".
- Enter the TTY setups and type the **??** command to list all the TTY setups.
- Exit **dspx**, and exit script logging with **exit**.
- Print the file or ftp it to a computer that supports a printer.

/usr/sigmet/config Hardcopy Listings

Print the following files or ftp them to a computer that supports printing:

- setup_dsp.conf
- rvp8.conf
- softplane_dsp.conf

Backup of /usr/sigmet/config directory

Test Passed

For Customer _____ Date _____

For Vaisala _____ Date _____

D.25 IFD Stand-alone SigGen Bench Test

Test Goal

Verify that the RVP8/IFD electrical I/O is working properly in an isolated environment.

Background

These stand-alone production tests are are performed on every IFD prior to shipment.

Special Test Equipment:

IRIS dspx utility, IF Signal Generator, Voltmeter

Test Procedure

- Begin by running **dspx** and temporarily reverting to factory settings with 'f'.
- Enter the **Mb menu** and choose an IF frequency f_{IF} MHz that matches the center frequency of the IFDs anti-alias bandpass filter.
- Enter the **Pr** menu, and type "RRRTTVVVVV" to move the starting plot range to 30km, the plot interval to 20μ sec, and the averaging factor to 10. Also type two space charac-ters to switch to the spectral display plot.
- Attach an IF SigGen to the input of the anti-alias filter that feeds the "IF-In" IFD input.
- Set the SigGen for f_{IF} MHz at 0dBm. Verify that the filtered power is within 0.5dB of -1dBm, and that a single clean spectral line is plotted. Use the "Z" or "z" keys to zoom Up/Down as needed.
- Increase the SigGen power in 1dB steps until distortion harmonics are seen on the plot. This should occur at 6–7dBm.
- Reduce the SigGen power to -20dBm, and sweep the frequency in 1MHz steps over a 20MHz band centered on f_{IF}. Verify that the bandwidth of the anti-alias filter matches its designed value (14MHz BW for the 30, 57.5 and 60MHz filters, and 4MHz BW for the 16MHz filter).
- Switch the SigGen off, and verify that the noise floor filtered power is within the follow-ing limits. Note that a very good quality shielded test cable is required here.
  - -83dBm to -81.0dBm for IFD Rev.D
  - -86dBm to -84.0dBm for IFD Rev.E and higher

- Move the SigGen cable to the other IFD filter input, and swap the plots by flipping IFD SW2 to its "A" position. Repeat the above four tests, now on the "Burst-In" SMA port.
- Place IFD SW1 in its "A" position, and verify that the AFC output voltage varies in a negative-zero-positive pattern as SW2 is flipped from A-B-C. Use the **offset** pot to set a nominal 0-Volt output, and the **gain** pot to set a ±5V span.

D.26 RVP8/Tx Stand-alone Bench Test

Test Goal

Verify that the RVP8/Tx electrical I/O is working properly in an isolated environment.

Background

These stand-alone production tests are are performed on every RVP8/Tx prior to shipment.

Special Test Equipment: IRIS dspx utility, IF Signal Generator, RVP8/Rx & IFD

Test Procedure

- Begin by running **dspx** and temporarily reverting to factory settings with f.
- In the **Mb** menu, choose an Intermediate Frequency that matches the RVP8/Tx analog output filters (for example, 60MHz), and a Blackman window.

  ```
  Receiver Intermediate Frequency: 60.0000 MHz
  Design/Analysis Window 0:Rect, 1:Hamming, 2:Blackman : 2
  ```

- Enter the **Pr** menu, and type "RRRTTZ" to move the starting plot range to 30km, the plot interval to 20µsec, and x2 zoom factor. Also type two space characters to switch to the spectral display plot. Exit the plot for now with "q".
- Setup the following in the **Mz menu**. Note that the frequency specified in the first ques-tion should match the U9 RVP8/Tx crystal (VCXO), and the output frequency should match the IF previously set in **Mb**.

  Onboard synthesizer TxClk/VCXO: 81.00000 MHz

  ```
  Lock TxClk to an external reference source: YES
  Source 0:ClkBNC, 1:RxClk : 0
  PLL ratio of (12/27) ==> Reference at 36.0000 MHz
  ```

```
ClkBNC 0:In/HiZ, 1:In/50Z, 2:TxClk, 3:RxClk, 4:RefClk : 1
Chan A 0:Unused, 1:FixedFreq : 1
FreeRunning fixed frequency : 60.00000 MHz
Output power level : 0.0 dBm
Gate the sinewave to produce a pulsed output: NO
```

- Connect the RVP8/Tx *Chan-A* output to the IF-In port of the IFD. It should be connected directly to the IFD's SMA input connector; not through any bandpass filter.
- Verify that the plot shows a single strong spectral line at the selected Intermediate Fre-quency, and that any spurious signals are down at least 60dB.
- Repeat the above three steps on the *Chan-B* output port of the RVP8/ Tx card.
- Apply a 0dBm SigGen waveform at the selected reference frequency (for example, 36MHz) to the *Clock* BNC input of the RVP8/Tx. Verify that the **V** command shows *Tx/Clk:Okay*, indicating that the VCXO is locking properly.
- Reduce the SigGen output to 20dBm and verify that locking is still okay. Vary the fre-quency by ±10KHz (0.01MHz) and check that lock is lost. Return to the center frequen-cy and verify that lock returns.

D.27 RVP8/Rx Stand-alone Bench Test

Test Goal

Verify that the RVP8/Rx electrical I/O is working properly in an isolated environment.

Background

These stand-alone production tests are are performed on every RVP8/Rx prior to shipment.

Special Test Equipment: IRIS dspx utility, RVP8/IFD

Test Procedure

- Begin by running **dspx** and temporarily reverting to factory settings with f.

APPENDIX E

RVP8 DEVELOPER'S NOTES (DRAFT)

This appendix describes the software environment that is provided to third-party developer's who wish to customize the RVP8 (and RCP8) algorithms to meet their particular needs. This information only applies to the very few organizations who have signed the Vaisala Software Developer's License Agreement. The vast majority of RVP8 operators and users should please skip over this entire appendix, as it is not relevant to your operational and scientific needs.

E.1 Organization of the RDA Software

RVP8 is Vaisala's open-architecture radar signal processor that uses standard PCI cards that can plug into any PC motherboard or embedded PCI backplane. The RVP8 software runs under standard RedHat Linux, and is developed and maintained using standard GNU tools (*gcc*, *gdb*, *make*, etc). The RVP8 therefore builds on generic high-volume PC hardware running the Linux kernel and GNU tool set. These building blocks are all mainstream, open, active technology, thus assuring the RVP8 a solid open environment for many years to come.

The larger RDA software collection, which includes the RVP8 within it, is organized into the following tree. A brief description of the contents and purpose of each directory is given:

```
/usr/sigmet/src/rda   Standard root point for top level of RDA tree
  * |-- intelipp     Covers for Intel Integrated Performance Primitives (IPP)
    |-- jamplayer   Altera JTAG support tools for FPGA chips on PCI cards
  * |-- kernelmod   Custom RDA kernel module to support PCI cards
    |-- lib         Static libraries originating from within this tree
    |-- picards     Board-level support for RVP8/Rx, RVP8/Tx and I/O-62 cards
    |-- rcp8        Root point of RCP8
    |   |-- core      CORE RCP8 code (not delivered to customers)
  * |   |-- open      OPEN RCP8 code (available to developers)
  * |   `-- site      SITE RCP8 code (customized by developers)
  * |-- rdasubs     Common support routines for the entire RDA system
    |-- rvp8main    Root point of RVP8 main threads
    |   |-- core      CORE RVP8 code (not delivered to customers)
  * |   |-- open      OPEN RVP8 code (available to developers)
  * |   `-- site      SITE RVP8 code (customized by developers)
    |-- rvp8proc    Root point of RVP8 compute processes
    |   |-- core      CORE PROC code (not delivered to customers)
  * |   |-- open      OPEN PROC code (available to developers)
  * |   `-- site      SITE PROC code (customized by developers)
    `-- softplane   Softplane abstraction for device independent I/O

/usr/sigmet/src/ts  Standard root point for top level of Time Series tree
    |-- archive     tsarchive GUI code (not delivered to customers)
  * |-- archlib     Library for TS related function
    |-- exec        tsexec code, does the work of tsarchive (not delivered)
  * |-- export      tsexport and tsimport code
    |-- lib         Static libraries originating from within this tree
  * |-- switch      tsswitch code
  * |-- view        tsview code
```

All software in each of the directories marked with a "*" is provided to licensed developers.

Note that the open portions of software contain all of the hooks that scientific programmers would need to add/modify the RVP8 processing algorithms to their taste. Interestingly, this comprises only about 15% of the roughly 105 thousand lines of code that make up the complete RDA system. We do not release the "guts" of the RDA that handles memory management, low-level card drivers, PCI interrupts, PCI DMA operations, cache optimizations, on-board FPGA code, etc. In other words, the material that might unfairly be used to clone the RDA products, but which a legitimate scientific developer would not need to use.

E.2 RVP8 Overall Code Organization

The remainder of this appendix will focus only on the RVP8 portion of the RDA development tree. The various RVP8 internal APIs gives the programmer a great deal of abstraction from the underlying hardware; and with it, freedom from worrying about things such as kernel support, interrupts, resource allocation, timing details, and the interfaces to higher layers such as IRIS and its utilities. The RVP8 is a layered software product whose organized is shown in Figure 57 on page 460:

- *PCI Board Firmware* – This is the code that runs within the Field Programmable Gate Array (FPGA) chips on the PCI cards themselves. The FPGA code is listed in this hierarchy because it is fundamental to the overall software model, that is, *all* of the hard real-time functions of the RVP8 are implemented at the chip level on one or more PCI cards. This allows the remainder of the RVP8 to run under standard (non real-time) Linux, because no Linux process ever needs to respond to events with critically short latency. As long as there is enough average CPU time during any 500ms interval, all of the jobs will get done with no loss of data.
- *Linux Kernel Module* – This module is *insmod'*ed at system boot time, and provides all of the low-level PCI support for the RVP8 hardware (RVP8/Rx receiver card, RVP8/Tx transmitter card, I/O-62 card, etc). It also provides the FIFO interfaces to the IRIS DSP driver, and other services that can only be implemented at the kernel level.
- *PCI Card Driver Library* – This library, along with the kernel module, constitute the low level board support package for the RVP8. Most of the hardware details remain hidden below this layer. The library is also responsible for handling FPGA firmware upgrades to the PCI boards with each release.
- *Softplane Driver Library* – This layer completes the hardware abstraction and provides soft configuration support for most of the electrical I/O. The *softplane.conf* file makes the association between logical control signals and physical I/O pins.
- *RVP8/Main Threads* – This collection of Linux threads is the foundation and support core of the RVP8, but they do not run any of the actual (I,Q) data processing algorithms. They handle all of the RVP8 configuration, setup and plotting commands, run the burst pulse analysis and AFC loop, define the system triggers and timing, and install transmit waveforms and receiver FIR filter coefficients. The Timeseries API is implemented in these threads, as is the opcode command interpreter that is documented in Chapter #6 of this manual.

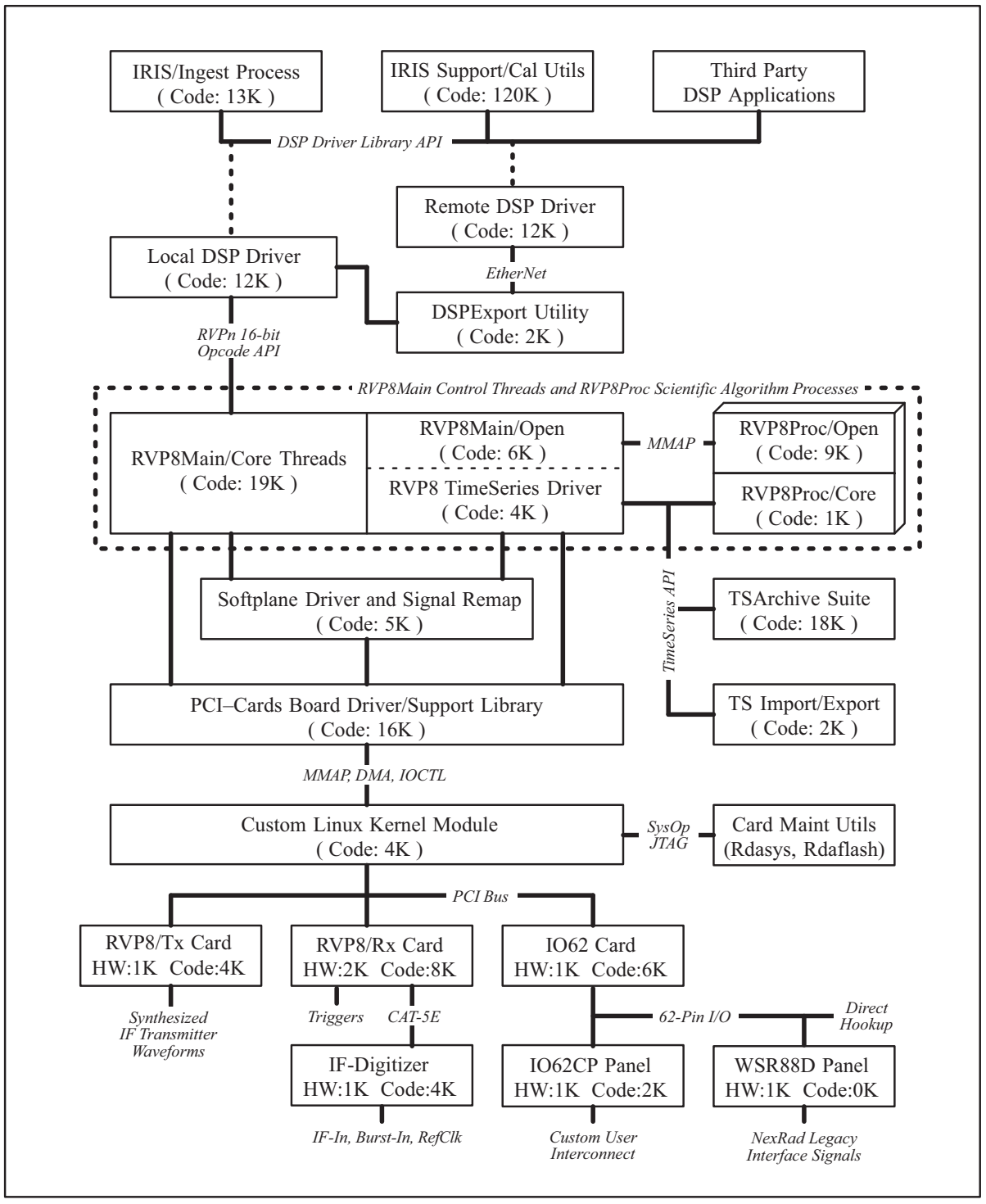


Figure 57 RVP8 Hardware and Software Organization

- *RVP8/Proc Processes* – These are N-copies of identical code that are forked by the MAIN threads and which carry out all of the scientific processing (See Chapter 5) within the RVP8. These are the actual "number crunchers" which implement most of the signal processing

algorithms that operate on raw (I,Q) data. The code is written in standard C, but uses a set of optimized Pentium/Athlon library functions for floating point FFT, convolution, filtering, dot product, etc. By calling these very fast primitives, ordinary C code is adequate for programming the processing algorithms. This makes the code very maintainable. Generally you will create one RVP8/Proc process per available CPU on an SMP machine. Most of your customization of the RVP8 algorithms will likely occur in these processes.

Virtually all of the RVP8/Proc code is released to licensed developers as source code. Likewise, there are open APIs for the RVP8/Main that allow you to build custom trigger patterns, phase sequences, etc., on the transmitter side. Finally, the kernel source is available so that developers can run within their own customized Linux environment.

E.2.1 RVP8 Software Maintenance Model

The remarkable thing about Vaisala's open source developer's model is that the open code is the actual delivered code. It is neither example code, nor an abridged form of the final delivered algorithms. When we build an official RDA release we're using the exact same rvp8main/open and rvp8proc/open source directories that our licensed developers would receive. All bug fixes, enhancements, new processing modes, and maintenance work will always be reflected directly in these source files -- there is no "other" version. This point is crucial in the model for software maintenance that includes both Vaisala code and customer code.

The CORE portion of the developer's tree contains no source files, and is delivered only as compiled binaries, whereas the OPEN portion of the tree is fully populated with the actual source files used by Vaisala to build each RVP8 release. This OPEN code should be used only as a reference for programming examples and ideas; *it should never be directly modified by the developer*. The reason this is so important is that the OPEN code may change significantly with each new release. If custom changes were made in this area, those changes would all be lost when the next RDA update was installed.

The correct way to add custom features to the RVP8 is to add them to the SITE directory. This directory is delivered simply as a collection of empty program stubs whose calling conventions are very stable. Each stub is basically a hook that can be used to define an entirely new RVP8 major mode. By replacing these stubs with custom code, custom major modes can be added to the RVP8. And since the delivered SITE stubs are simply thrown away and replaced with their customized versions, the long term maintenance of both the Vaisala portions and developer portions of the RVP8 code are assured.

E.2.2 Installing Incremental RDA Upgrades

RDA development systems are upgraded in the same manner as standard operational systems using the procedures described in the *Software Installation Manual*. The only difference is that you have a little more work to do afterward to continue running your custom code under the new RDA release.

The RDA upgrade will replace the RVP8/Main and RVP8/Proc SITE libraries with their default stubs, so you will need to rebuild your customizations. Since the executables are built from the OPEN trees, the simplest method is to copy those newly released OPEN areas alongside your existing SITE directories. Your entire development cycle can then be carried out entirely from the */home/operator* area, without ever having to compile in the */usr/sigmet* tree directly.

```
$ cd /home/operator/src/rda

$ rm –rf rvp8main/open rvp8main/core rvp8proc/open rvp8proc/
core

$ cp –pr /usr/sigmet/src/rda/rvp8main/open rvp8main

$ cp –pr /usr/sigmet/src/rda/rvp8proc/open rvp8proc

$ make clean

$ make –j2

$ make –j2 install
```

What is important here is that you are rebuilding your code under the new header files that were just installed during the upgrade. As such, you are likely to find a few minor incompatibilities in the form of changed structure definitions and/or changed function prototypes. Check the header file documentation as well as the current *RDA Release Notes* to find out whatever (usually simple) changes need to be made. The RVP8 will not let you run with incompatible headers, producing RVP8/Main startup errors such as:

```
RVP8/Main code version mismatch

Core: Version=2.12 Build=298

Open: Version=2.12 Build=298

Site: Version=2.11 Build=295
```

and RVP8/Proc errors later in the initializations such as:

```
Forking parallel compute process(es)...

RVP8/Proc–0: Requesting exit due to signal 1

RVP8/Main: UNIX Signal: Unexpected RVP8/Proc termination

RVP8/Main: RVP8/Proc version mismatch <ProcSite: Ver=2.11
Bld=295>
```

E.2.3 Rebuilding the RDA Linux Kernel Module

The RVP8 requires kernel driver support that is provided with each RDA release in files whose names resemble */usr/sigmet/bin/rda/rda–2.4.20–6smp.o*. If you are running on standard RDA hardware, the rdasys start script will find the appropriate kernel module for your system and install it automatically. However, if you are running with a custom Linux kernel, you will have to rebuild your RDA kernel module whenever its version has changed since the last time that you built it. The RVP8 will not run with incompatible modules and will report an error:

```
RVP8/Main: KernelMod mismatch

<Found SIGMET RDA kernel module V3.3, but V3.2 is required>
```

First install the headers, source, and objects from the release cdrom. Then copy the rda source over to your development tree. Then compile. If you wish to cross-compile for a different kernel version some extra work is required.

```
$ cd

$ mkdir ./src

$ cp –r /usr/sigmet/src/rda ./src/rda

$ cp /usr/sigmet/src/config.mk ./src

$ cd ./src/rda/kernelmod

$ make clean

$ make

$ sudo make install
```

The new module will be automatically loaded on the next boot, or it can be manually installed right now (again, as root) using:

```
# rdasys stop

# rdasys start
```

E.3 Debugging and Profiling Your Code

Although the complete RVP8 is a rather complex multi-thread and multi-process system, it is still "merely" a user-level application running under the Linux operating system. As such, most of the of the custom code that you develop can be debugged using tools that are already familiar to Linux/C/GNU programmers. This section presents an overview of debugging and monitoring techniques that are available to RVP8 developers.

E.3.1 Monitoring Opcode/Data Activity: –exposeIO

The RVP8 is generally controlled by some higher level application such as ascope, iris, dspx, etc., which communicates with the RVP8 via the *IRIS DSP Driver Library* using the opcodes described in Chapter 6. These layered applications provide a clean and maintainable signal processor interface, as evidenced by the RVP5/6/7/8 being largely opcode-compatible over a fifteen year period.

The complete opcode activity between the application driver and the RVP8 can be viewed by including the *–exposeIO* flag on the RVP8 startup command line. The following printout shows what happens when an "Output Test" opcode is written, followed by a "Noise Sample" command and a "Get Processor Parameters" opcode:

```
Opcode 0x0004 (OTEST)

Output Words

0: 0001 0002 0004 0008 0010 0020 0040 0080 0100 0200 0400
0800 1000 2000 4000 8000

Opcode 0x0005 (SNOISE)

Input Words

0: 0000 0000

Opcode 0x0009 (GPARM)
```

```
Output Words

0: 2000 0064 0960 9DCF 0110 0DD0 0000 0000 0000 5284 0000
0000 0040 D472 0000 0000

16: 0000 0603 020D 5DC0 012C 1D4C 1770 0BB8 8421 0210 2EE0
2EE0 0000 0000 042E 07AE

32: 000D FE20 0066 0050 FE70 0000 0000 0000 0000 0000 0001
0011 0000 0011 0000 0DD0

48: 0000 0000 00E8 01C0 03E8 04B0 0303 0000 0037 30CB 0003
0000 0000 0000 0000 0000
```

When the OTEST opcode is received it is parsed as such by the RVP8/Main *HostCmds* thread, which also generates the *exposeIO*printout. The opcode is shown in both numerical and mnemonic form, followed by the sixteen-word walking-ones pattern that results from it. The SNOISE command then arrives, along with the two input words which are expected by that opcode. No output words are generated by SNOISE. Finally, the GPARM opcode is received and the 64 output words that it produces are displayed.

Being able to watch the opcode-level activity of the signal processor can be very helpful when debugging both custom driver code and custom opcode handlers that you might write for the RVP8.

E.3.2 Showing Live Acquired Pulse Info: –showAQ

The (I,Q) data that are computed by the FIR filters on the RVP8/Rx PCI card are transferred to CPU memory by way of DMA (Direct Memory Access) bus cycles that are initiated by the RVP8/Main threads. This is an implementation detail that developer's can largely ignore, except to be aware that the radar data always arrive in discrete "chunks" and that the TimeSeries API is updated accordingly. The following printout shows the (I,Q) bus activity for a Dual-Polarization (dual RVP8/Rx cards) system when the *–showAQ* flag is included on the RVP8 startup command line.

```
AQ:193 pulses(1:1B96 – 1:1C57) 32038 words(8:2762F –
8:2F355) 157ms

AQ:192 pulses(1:1B9B – 1:1C5B) 31872 words(8:27B0A –
8:2F78A) 3ms

AQ:191 pulses(1:1C57 – 1:1D16) 31706 words(8:2F355 –
8:36F2F) 157ms
```

```
AQ:192 pulses(1:1C5B – 1:1D1B) 31872 words(8:2F78A –
8:3740A) 4ms
```

Here we see 193 pulses of packed 32-bit (I,Q) data words being transferred from the first Rx card, followed 3ms later by a similar transfer from the second Rx card. Each block of data is moved directly into the virtual address space of the RVP8/Main without requiring any cycles from the CPU itself. Then, a DMA-Done interrupt causes the *IQ-Data* thread to wakeup and unpack that card data into FLT4 values which are written to the TimeSeries API.

Do not worry too much about the other numbers that are printed from *–showAQ*. The flag's main use is to show how chunks of timeseries data are being made available to the RVP8 processing threads. If you are curious about the timing details, this is a simple way to take a look.

E.3.3 Showing Coherent Processing Intervals: –showCPIs

Coherent Processing Intervals (CPI's) are blocks of acquired pulses that have been selected as the input data for the computation of each processed ray. You can monitor how the timeseries data stream is being organized into rays by supplying the *–showCPIs* flag on the RVP8 startup command line. A sample printout while running *ascope* is shown below.

```
CPI 0: 64–Pulses (269.74, 1.49) to (271.25, 1.49) 126.00–ms
TS: 1%

CPI 1: 64–Pulses (270.51, 1.49) to (272.02, 1.49) 126.00–ms
TS: 0%

CPI 2: 64–Pulses (271.28, 1.49) to (272.79, 1.49) 126.00–ms
TS: 0%

CPI 3: 64–Pulses (272.04, 1.49) to (273.56, 1.49) 126.00–ms
TS: 1%
```

The CPIs are numbered beginning with each new ProcSection, and the number of pulses within each one is shown. The starting (AZ,EL) and ending (AZ,EL) are then printed, along with the duration of the CPI in milliseconds. In the above example the PRF was 500Hz, hence, the 64 pulses span a total time of 126ms. Finally, the "lateness" of the data being extracted from the TimeSeries API is listed as a percentage of available history depth.

If the lateness approaches 100%, that is evidence that the RVP8 is having trouble keeping up with the CPU and/or memory throughput demanded by

the current major mode. The default algorithm for blocking CPIs keeps track of whether it seems to be falling behind, and tries to gracefully skip pulses in order to catch up. Custom CPI algorithms for intensive major modes should be written with the same sort of feedback. See the code in *rvp8main/open/cpi.c* for more details and suggestions.

E.3.4 Showing RealTime Callback Timers: –showRTCtrl

The RVP8 can show the detailed activity of whatever callback timers have been registered for the current major mode. If the *–showRTCtrl* flag is supplied on the command line, then timer activity will be printed during each active ProcSection. These timer callbacks provide a simple yet powerful framework for developers to handle real–time events within the RVP8 (See E.5 Real-Time Control of the RVP8 on page 474). See also the documentation for *struct rtCtrlCBTimer*.

When no callbacks are registered, or when the registered callback does not request any further activity, the printout will simply look like this:

```
RTC –First- #0(-) #1(-) #2(-)

RTC Disabling further callbacks for this PROC section
```

The first real–time callback does not set any of the *iTVWaitOnNext* fields, and the whole timer mechanism simply goes to sleep until the next ProcSection begins. In contrast, the following printout was generated by the demonstration histogram callback that is provided in *rvp8main/open/rtctrl.c* :

```
RTC –First- #0( dV:0 F:0 Rq:100000 ) #1(-) #2(-)

RTC Setting timer #0 clock source to 0 (1MHz)

RTC 0.100068 #0( dV:100009 F:1 Rq:2500 ) #1( dV:0 F:0 Rq:5 )
#2(-)

RTC Setting timer #1 clock source to 1 (Trig)

RTC 0.002512 #0( dV:2513 F:1 Rq:2500 ) #1( dV:2 F:0 Rq:5 )
#2(-)

RTC 0.002510 #0( dV:2510 F:1 Rq:2500 ) #1( dV:3 F:0 Rq:5 )
#2(-)
```

Here we see the first invocation requests a callback in 100000 counts of the 1MHz timer. The "Rq" (request) field of timer #0 shows the *iTimerWait*

duration, and the *iTVWaitOnNext* field causes the 1MHz clock to be selected as the timer source.

That callback fires a little more than a tenth of a second later (the additional 68 sec represents the Linux Interrupt-to-Running latency, plus the time to set the clock source in the first place). On the second invocation, the demo callback requests a delay of 2500 on the 1MHz timer (2.5ms), and a delay of five counts on the Trigger timer. We also see the RVP8 trigger being selected as the clock source for timer #1. Note that the clock source messages are printed only when changes are made.

From then on the callbacks fire regularly based on activity on timers #0 and #1. The "dV" number show the "delta-value" for each timer, that is, the change in the timer's count since the previous call. The "F" field indicates whether each timer has fired (1) or is still counting up to the requested delay (0). In this case, timer #0 is regularly firing every 2.5ms; and since we only get two or three trigger counts in that much time, timer #1 never actually fires at all. If the PRF were increased, however, we would suddenly see timer #1 counting up to five triggers and then firing before the 2.5ms expires.

E.3.5 Using ddd on the Main & Proc Code

The GNU *ddd* symbolic debugger is (usually) built on top of the *dde* command line debugger. Both are mature and remarkably well crafted tools that are provided on all Linux systems.

Code that you write for the RVP8/Main threads can be debugged simply using:

```
$ cd /home/operator/src/rda/rvp8main

$ make –j2

$ cd open

$ ddd rvp8
```

Here the SITE and OPEN code is first compiled in the private development tree, and we then run the RVP8 executable that resides in the OPEN area. You may want to redefine the environment variable "OPTIMIZEFLAG=–g", that is, remove the "-O2" that is normally there. This will cause the Makefiles to build non-optimized code which generally behaves more smoothly in a symbolic debugger.

You can simply type "run" in the *ddd* execution window, but you may prefer to use "run –nod" to skip the powerup diagnostics and speed up your

development cycle. Typing "b main" ahead of time will cause *ddd* to break on the first executable line after all of the dynamic library references have been resolved. This is usually a good way to get started because you may then break on any entry point within the RVP8. Before the libraries are loaded, it is possible that *ddd* may not be able to find all of its symbols.

NOTE When the RVP8/Main is debugged in the above manner, signal events originating from ddd will also be sent to the RVP8/Proc child threads and are likely to cause them to behave improperly. Use this technique only for quick ventures into the main threads, and preferably with –procs 0. Please read on.

Debugging the RVP8/Proc code is similar, except that we do not want it to be automatically started from the main threads. Therefore, in one X-Term window type:

```
$ rvp8 –noFork
```

which will startup the RVP8/Main threads, but pause at the point that the RVP8/Proc process would normally be created. Note that *–noFork* forces the subprocess count to one, as if you had included *–procs 1* on the original command line. Then, in another window type:

```
$ cd /home/operator/src/rda/rvp8proc

$ make –j2

$ cd open

$ ddd rvp8proc
```

which builds the new RVP8/Proc code and starts up the debugger. Typing "run" in *ddd* will startup the subprocess; and when it has finished its initializations, the RVP8/Main will magically continue in the other window.

NOTE The proper way to debug the RVP8/Main builds on this technique. Run ddd rvp8 as described above, but include the –noFork flag in ddd's "run" command. Then run rvp8proc manually in another window, either with or without its own ddd. You can create two simultaneous Main and Proc ddd sessions in this manner, and signals from one will not interfere with the other.

E.3.6 Finding memory leaks with valgrind

The *valgrind* profiler can be useful if you are having runtime problems that are hard to track down. The most common problem that it solves is finding reads-before-writes, that is, when you forget to set a value in a structure somewhere, and then reference it later before actually writing into it. Malloc/Free inconsistencies are also easily diagnosed with *valgrind*.

Valgrindcan be downloaed from http://valgrind.kde.org. Valgrind is very easy to use because you simply run it on the executable being debugged in the same ways that *ddd* was used in the previous section. Moreover, there are no special compiling or linking requirements. To debug the RVP8/Proc process, for example, simply use:

```
$ cd /usr/sigmet/src/rda/rvp8proc/open

$ valgrind –tool=memcheck rvp8proc
```

E.3.7 Profiling with gprof

The GNU tools include the handy runtime profiler *gprof*. This tool works in conjunction with the C-compiler, and analyzes statistics in the *gmon.out* file that is produced when the running program exits. The RDA Makefiles are already setup to build profiled versions of either the RVP8/Main or RVP8/Proc executables. To do this, define *one* of the following environment variables:

```
export PROFILE_RVP8MAIN="1"

export PROFILE_RVP8PROC="1"
```

Then do a "make clean" and "make install" in the SITE and OPEN portions of the relevant tree. If you are profiling the RVP8 compute process you'll need to make sure that only one of them is forked by including "–procs 1" in the original startup command. Otherwise, each sub-process will attempt to create the same *gmon.out* and chaos will surely follow. If you forget to specify the solitary process option, the RVP8 will force it upon you anyway but with an accompanying warning message. Likewise, the RVP8 will not let you profile both the Main and Proc executables at the same time.

Since the *rvp8* and *rvp8proc* executables are built from their OPEN directories, running the profile analysis from within that directory will allow *gprof* to find its symbols, for example,

```
$ cd /home/operator/src/rda/rvp8proc/open
```

```
<Run the RVP8 for a while...>

$gprof rvp8proc –I ../site
```

E.4 Creating New Major Modes from Old Ones

Custom algorithms are added to the RVP8 by building on its concept of Major Modes and Output Data Types. Each Major Mode defines all of the methods that are required to compute each of the Output Data Types from raw (I,Q) data. Therefore, by allowing users to define their own Major Modes, one has all of the hooks required for full customization. You can code up your own custom algorithms by making incremental changes to one of the Vaisala models; or you can start from scratch and build something completely unique.

The best way to create a custom major mode is usually to start with the code for an existing one and incrementally modify it to include the new features. The FFT mode, for example, is defined in the files *rvp8main/open/mt_fft.c* and *rvp8proc/open/ct_fft.c* , each of which contains only about 100 lines of boilerplate toplevel definitions. Creating your new major mode would likely begin by copying this prototype code into separately named files in the *rvp8main/site* and *rvp8proc/site* areas, and then proceeding to make the desired changes.

There are four major mode slots reserved for custom user applications. The names of your new modes are defined using the **setup** utility's *RVP–>Optional Data Parameters* area. Names can be up to fifteen characters long, and whatever text you choose here will automatically appear later in pulldown menus for **ascope, iris**, etc. Your new code is invoked by modifying one of the lines of *rvp8main/site/mt_user.c* and *rvp8proc/site/ct_user.c* . These files are very simple and are shipped in deactivated form as follows:

```
/* ----------------------------

* The available user modes are shipped in an unused state.
By doing

* nothing in their INIT routines, these modes are effectively

* disabled (all function pointers remain NULL).

*/

void mtInitMajorMode_user1( void ) {}

void mtInitMajorMode_user2( void ) {}
```

```
void mtInitMajorMode_user3( void ) {}

void mtInitMajorMode_user4( void ) {}
```

All that's required for execution is that you call your custom initialization routine(s) from one of these predefined user stubs:

```
void mtInitMajorMode_user1( void ) {
initMajorMode_myCustomMode() ; }
```

E.4.1 Function Pointers are the Key to Customization

Each major mode is characterized by a set of function pointers or *methods* which define how certain critical operations are to be carried out. The main RVP8 threads are governed by the following methods:

```
struct rvp8MainMajorMode { /* Customized processing routines
*/

privateData_t *privateData ;

exitMajorMode_f *exitMajorMode ;

initProcSection_f *initProcSection ;

exitProcSection_f *exitProcSection ;

rtCtrlCBF_f *rtCtrlCBF ;

iNominalTrigSequence_f *iNominalTrigSequence ;

iCustomTrigSequence_f *iCustomTrigSequence ;

customUserOpcode_f *customUserOpcode ;

cwpulseMatchedFilter_f *cwpulseMatchedFilter ;

iTxWaveformDesign_f *iTxWaveformDesign ;

rawPulseCorrections_f *rawPulseCorrections ;

rawPulseCorrections_f *targetSimulator ;

lConfigError_f *lConfigError ;

lFindNextCPI_f *lFindNextCPI ;
```

```
lCheckupCPI_f *lCheckupCPI ;

lAssignProcsInCPI_f *lAssignProcsInCPI ;

setupProcBins_f *setupProcBins ;

lAnalyzeBurstPhseq_f *lAnalyzeBurstPhseq ;

getAzElPosBtime_f *getAzElPosBtime ;

sampleLiveAzElPos_f *sampleLiveAzElPos ;

insertLiveAzElPos_f *insertLiveAzElPos ;

frontPanelDisplay_f *frontPanelDisplay ;

} ;
```

User's of object-oriented languages will recognize this sort of structure as allowing new objects to inherit existing properties of old objects, while still permitting individual "personality" to enter as needed. Like the Linux Kernel, the RVP8 is written in standard "C" and uses function pointer replacement to accomplish customization. Much of the RVP8's code organization was patterned after the elegant and far more intricate implementation of driver module replacement within Linux.

Note that the *initMajorMode()* routine which creates each major mode in the first place is not part of this list because it is separately invoked (See E.4 Creating New Major Modes from Old Ones on page 471) in order to first establish the list. Detailed descriptions of the various methods are included along with their definitions in *rvp8main.h* and *rvp8proc.h* . A few are so fundamental that they're worth repeating here:

privateData_t : Each major mode can allocate a private data area for whatever memory resources are required during the operation of that mode. This private area is created and prepared by the *initMajorMode()* routine as part of its filling in the entire list of methods at the start of each major mode. The *exitMajorMode()* handler is responsible for releasing all private memory resources.

exitMajorMode_f : This routine releases all resources that were allocated during the initialization and execution of the major mode. The EXIT routine is called whenever the major mode changes, or whenever we are "between" modes, for example, after a TTY Chat "q" command, or a processor reset. The RVP8 is guaranteed not to be within an active PROC Section (See below) when *exitMajorMode()* is called; that is, if *exitProcSection()* needs to be invoked, that will always happen first.

initProcSection_f & **exitProcSection_f :** Pair of routines that will be called whenever the RVP8 enters and exits active timeseries acquisition and processing. The INIT routine is called when a PROC opcode is executed that causes the *IQData* thread to begin collecting data, and activates the RVP8 compute threads for the current major mode. The EXIT routine is called whenever anything interrupts those processes, for example, the execution of most other opcodes. Note that the major mode itself does not change at these transitions; rather, these functions allow the current major mode to interact with the timeseries data acquisition as needed.

E.5 Real-Time Control of the RVP8

The RVP8/Main includes a dedicated thread (named "*RT-Ctrl*") which can be programmed to handle real–time events while the processor is running. This includes activities such as altering trigger patterns in response to antenna position or other external conditions, tracking live inputs from an I/O-62 card, etc. The *RT-Ctrl* thread runs at a priority that is higher than any other RVP8 thread. This allows it to preempt other activities to achieve near real–time behavior; but as such, it should always be coded as efficiently as possible and should not do anything that could possibly become CPU bound.

Despite the high POSIX static priority of *RT-Ctrl*, its scheduling is still subject to jitter due to uncertain latencies within the Linux kernel. The magnitude of this jitter will depend on the kernel version, the type of motherboard being used, and the nature of the other processes that are competing for CPU time. We've found that intense disk and network I/O are among the worst contributors to high scheduling latency, sometimes amounting to as much as 25ms of variation. Fortunately, these uncertainties can usually be absorbed by proper coding of the thread.

E.5.1 Using the Programmable Callback Timers

The *RT-Ctrl* thread is structured around a flexible set of real–time callback timers. The thread is activated each time the *initProcSection* method is called, and it is deactivated when that PROC section is eventually exited. Thus, real–time control is available whenever live (I,Q) data are also being acquired and processed.

The **rtCtrlCBF_f** callback function is customized to handle the real–time control and sequencing tasks for the current major mode. This routine will be called once at the beginning of each data processing section. It then performs whatever work needs to be done, and requests that it be called

back at some later time. Up to three independent timing and/or event criteria can define when the callback will occur, and each can be based on:

- A specified number of counts of a free running 1MHz counter (clock time)
- A specified number of trigger pulses (trigger relative time)
- A specified number of external input line transitions (I/O event time)

The callback will occur as soon as the timeout criteria are met for any of the three timers that are activated. For example, if Timer-0 requests a callback after five triggers, and Timer-1 requests service after 10000 1MHz counts, then at 1KHz PRF the callback will occur in 5ms. Note that the callback sequence can be terminated at any time within the current PROC section simply by specifying void criteria for reentry.

Callback routines can request that a new trigger bank be loaded at the precise instant that the next timer event(s) fire. This special feature is implemented in hardware on the RVP8/Rx card, and is necessary for creating alternating trigger patterns that remain completely unaffected by Linux interrupt latencies. Precise programmable trigger control is an important application of the *RT-Ctrl* thread, and is made possible by having this hardware support at the PCI card level.

E.5.2 Example: Standard Trigger/Antenna Events

Refer to the source file *rvp8main/open/rtctrl.c* which contains the standard RVP8 code for live trigger control and angle synchronization. The software consists of these pieces:

i**nitProcSection_dflt** – This is the default routine for initial entry into each PROC section, and its primary job is deciding what kind of real–time callback responses are needed for the current RVP8 configuration. It sets up an area of private memory that will later control the real–time activity (see calling example in *rvp8main/open/mt_fft.c*).

rtCtrlCBF_dflt – This is the default real–time callback of the RT-Ctrl thread. It receives a subset of the private memory for this PROC section that was set aside for real–time control (see calling example in *rvp8main/open/mt_fft.c*), and performs one of the following:

- **Angle synchronization** – A brief history of prior antenna angles are entered into a least squares model, which is then used to predict how long it will be until the next sync angle boundary is crossed. A callback based on the 1MHz counter is then requested for that time using one of the timers:

```
cbt >iTVWaitOnNext = RTCBTV_1MHZ ;
cbt– >iTimerWait = (1000 * iFutureMS) ;
```

along with an immediate hardware assisted trigger bank change using:

```
cbi_a– >iTgBank = iBankNew ; cbt– >lTgBankAutoStart =
TRUE ;
```

- Freerunning Dual-PRF – This mode is much simpler, and simply alters the pattern of triggers after a computed number of pulses have occurred:

```
cbt– >iTVWaitOnNext = RTCBTV_TRIG ;

cbt– >iTimerWait = ceil( MAX( 1.0, fTrigs ) - 1E-4 ) ;

cbt– >iTimerWait –= cbt– >iTimerError ;
```

Note that interrupt latencies are absorbed differently in these two cases. For angle sync'ing we compute the expected crossing time based on where the antenna happens to be when the callback was entered. It does not matter if a callback is delayed, because we'll simply compute a shorter future time in such cases. For Dual-PRF triggers we likewise need to shorten the next interrupt whenever the present callback is delayed, but we do this using the *iTimerError* field that keeps track of how late (measured in timer events) we presently are.

E.5.3 Example: RealTime Interrupt Histogram

Refer to the source file *rvp8main/site/demohist.c* which contains demonstration code for a real–time callback that prints a histogram showing the scatter of callback times brought about by Linux scheduling latencies. First, arrange for this code to be attached to a user major mode by editing *rvp8main/site/mt_user.c* :

- Change the default USER1 major mode init routine to call the demo histogram:

```
void mtInitMajorMode_user1( void ) {
initMajorMode_demohist() ; }
```

- Include the header file consisting of a single line prototype for the above function:

```
#include "demohist.h"
```

- Compile, install, and run your changes with:

```
$ cd /usr/sigmet/src/rda/rvp8main
```

```
$ make clean

$ make –j2

$ rvp8 –noDiags
```

The RVP8 will startup normally. Next, run the **ascope** utility and request USER1 major mode from the *Gen Setup* menu. You should then see messages printed in the RVP8 startup X-Term indicating that the major mode has been entered, and that the PROC section has been initialized:

```
Beginning PROC section demohist code.

Target scheduling interval is 2.500 ms OR 5 triggers

Will print timing jitter histogram every 10 seconds
```

The real–time callback handler is now collecting statistics on its interrupt times, and is scheduled every 2.5 milliseconds or every five triggers, whichever is fastest. Moreover, a histogram will be printed every ten seconds showing the distribution of times. Some interesting things to try:

- Type a low PRF (~500Hz) into **ascope** and verify that the interrupts cluster around 2.5ms. Then type a high PRF (~5KHz) and verify that histogram peaks at 1ms.
- While the callback handler is running, make the RVP8 machine busy in some manner, for example, by running compilers, reading large files, etc. You should see noticeable scatter of the histogram plot as other processes compete for CPU scheduling.

Lastly, verify that selecting some other major mode in **ascope** results in the messages:

```
Exiting from demohist PROC section

Exiting DemoHist major mode
```

E.6 Customizing the (I,Q) Data Stream

E.6.1 Defining the FIR Matched Filter

E.6.2 Applying Raw Pulse Corrections

E.6.3 Inserting UserIQ Header BIts

E.7 Customizing the Front Panel Display

E.8 Adding Custom DSP/Lib Opcodes

E.9 Using the Softplane for Physical I/O

E.9.1 Softplane Programmer's Model

E.9.2 Reducing Unnecessary PCI Traffic

E.10 Handling Live Antenna Angles

E.11 Creating Custom Trigger Sequences

E.11.1 Defining Trigger Waveshapes

E.11.2 Defining Trigger PRT Sequences

E.11.3 Polarization and Phase Control

E.11.4 Example: Adding PRT Micro-Stagger

Please refer to the source file *rvp8main/site/demostag.c* which contains demonstration code for building custom trigger timing within a new major mode. A special "micro-stagger" trigger generator is implemented in which the trigger PRT varies a tiny amount (a few microseconds) from pulse to pulse. This causes multiple trip echoes to become incoherent when viewed relative to the first trip Tx phase, thus providing a very simple method of "whitening" the Doppler signature from range aliased echoes.

The code consists of two parts:

- **initMajorMode_stag** – The major mode initialization routine is a direct clone of the FFT code from *rvp8main/open/mt_fft.c* . The only difference is that the default trigger generation is superceded by the custom *iNominalTrigSequence_stag* method.
- **iNominalTrigSequence_stag** – This is a direct clone of the factory default trigger code in *rvp8main/open/txsubs.c* , except that the PRT sequence is altered by a zero-mean pseudo-random set of time staggers.

To compile and run this mode, please modify *rvp8main/site/mt_user.c* as described in E.5.3 Example: RealTime Interrupt Histogram on page 476. In addition, we'll want to import the default FFT behavior for the compute processes by changing *rvp8proc/site/ct_user.c* to:

```
void ctInitMajorMode_user1( void ) { ctInitMajorMode_fft() ;
}
```

Build and install all of these changes by compiling both the *rvp8main* and *rvp8proc trees*. You may then request this user mode from ascope and verify the micro-staggered PRTs on a delayed sweep oscilloscope. For example, at 1KHz PRF and with a delayed sweep of 1ms, you should see a flurry of "second bang" triggers whose leading edges jitter over a 5 sec span. Contrast this with the normal FFT mode which has constant interpulse spacing.

Another way to verify the staggered PRTs is to check the live timeseries data using the example source utility that is included in the *rdasubs* directory:

$ /usr/sigmet/src/rda/rdasubs/rvp8ts_example –headers

```
-SeqNum-  --AZ--    --EL--    PrevPRT   NextPRT   Bank      Wave      TX
00014289  0.00      0.00      720088    720120    0         60        00
0001428A  0.00      0.00      720120    720080    0         61        00
0001428B  0.00      0.00      720080    720056    0         62        00
0001428C  0.00      0.00      720056    719952    0         63        00
0001428D  0.00      0.00      719952    719992    0         0         00
0001428E  0.00      0.00      719992    719832    0         1         00
0001428F  0.00      0.00      719832    720040    0         2         00
```

The PRF was set to a low value of 100Hz in order to limit the rate at which the live headers were printed. A 72MHz IFD was used in this measurement, hence the nominal time to the previous and next 100Hz pulses would be 720000 clock ticks. However, because of the micro-stagger, the actual interpulse PRTs are seen to vary.

E.12 Determining CPI's and Ray Boundaries

E.13 Using the RVP TimeSeries API

The RVP TimeSeries API is the fundamental interface through which (I,Q) data are made available to all application code which requires them. This API is central to the design and operation of the RVP8 itself, and is used by the parallel compute processes to access incoming timeseries data. Because the API is used so heavily, the entry points have been reasonably stable and well debugged since the early days of the RVP8.

The TimeSeries API is provided within a larger collection of RDA support services that are located in *rda/rdasubs*, with the defining header file *include/rdasubs_lib.h* . Please see the documentation in that header file for the most recent API definitions. This file is heavily documented for this purpose. If you have any specific questions we would be happy to improve the comments.

E.13.1 Reader and Writer Clients

The timeseries API is entirely stateless and passive from a reader client's point of view, that is, it allows any number of callers to eavesdrop on the (I,Q) data as they arrive, but there are no control actions passed back in the other direction. The reason that an event driven model is not provided is that the API is fundamentally a single-writer / multiple-reader interface. There is no private state maintained for each reader client that hooks up to it. This was a design goal from the start; readers should be able to come and go willy-nilly without affecting anything else. As such, the notion of 'notify me when new data are available' would not be well defined, because 'new' would have to be a per-client notion, that is, 'new' since the last data that each particular client happened to look at.

Also, the API is really most valuable in providing random access to the recent buffered (I,Q) data. Because of the PCI/DMA buffering this is a pseudo real-time interface, and the data are typically 100–400ms delayed from their actual time of arrival. As such, it is not terribly important to be able to track the leading edge of newly arriving data with any particular precision. There are about two seconds of data buffered within the timeseries API; so merely checking at 10–20Hz presents a negligible CPU load, does not risk losing any data, and matches the PCI/DMA burst transfers.

E.13.2 Attach/Detach Details

Use rvptsAttach() to attach, and rvptsDetach() to release the connection. Always attach to unit RVPTS_UNIT_MAIN, the others are for internal RVP use. You also must specify the client type you are. Readers are generic, but for writers we have several choices because we need to switch sources, and need to know who the sources are.

See the example program rda/rdasubs/rvpts_example.c. Other programs using the Timeseries API are: ts/export/tsexport.C ts/export/tsimport.C, ts/switch/tsswitchMainWin.C, ts/exec/TsArchExec.C, ts/exec/tsclientshell.C, ts/archive/tsarchAS.C.

E.13.3 Extracting Pulses via Sequence Numbers

Because there is a ring buffer filled with time series, the first thing a reader should do is get the most recent pulse, and start reading after that. We do that with the function rvptsCurrentSeqNum(). Take a look at the source code to tsexport.C to see how this is done. To get data starting at a specific sequence number, use rvptsGetPulses(). If there are none available, then we should sleep to wait for some more.

Note that if the RVP is reconfigured in some way, this will cause a change in the "acquisition mode". An example might be that the PRF, pulse width, or transmit polarization has changed. All data read from a single call to rvptsGetPulses() will be for the same acquisition mode.

E.13.4 Using Memory Bandwidth Effectively

It will be a waste of effort to read each pulse of data individually. So a smart reader will sleep until at least, say, 10 pulses have arrived before reading them. To support this, there are function calls to find out how many pulses are available after a give sequence number, and how old a given pulse is. Look at tsexport.C to see how to use this.

E.14 Using the Intel IPP Library

The IPP software enables taking advantage of the parallelism of the single-instruction, multiple-data (SIMD) instructions that comprise the core of the MMX technology and Streaming SIMD Extensions. These technologies can vastly improve the performance of computation intensive signal processing applications. Refer to the complete *Integrated Performance Primitives Reference Manual Volume 1* which is supplied in PDF form on each RDA CDROM.

The data types used within the IPP library are, for the most part, compatible with the standard Vaisala typedefs; hence, the IPP routines can almost always be called directly from RVP8 code. You can find many examples of this simply by grep'ing the RVP8 sources for "ipps". Status codes from the IPP library can be converted into RDA MESSAGEs via *sigIppStatus()*.

The header file *intelipp_lib.h* should be used to define the IPP library entry points. Do not include the Intel headers directly because doing so will

bypass some RDA consistency checks and will make your code less maintainable.

If you are new to the Intel IPP library we provide an example file *rda/intelipp/ipp_example.c* that you can read through to get started. It allows you to run DFT's and matrix transposes of various sizes, and provides a simple method of benchmarking the IPP library on your hardware.

The IPP runtime libraries are bundled into each RDA release. Vaisala has purchased a single-user developer's license from Intel which allows one engineer to develop code which links to the IPP library, and then to distribute an unlimited number of copies of that code in executable form. Our license does not, however, allow for any additional programmers to develop their own code as an extension of Vaisala's license.

Essentially, the IPP license is a per-developer license. There are no restrictions or royalties on the distribution of compiled binaries, but each additional developer must be licensed at a cost of approximately $250/year. Some relevant text is included below from the Intel online FAQ http://www.intel.com/software/products/ipp/faq_lic.htm . The bottom line is that our community of RDA developers must individually license themselves with Intel in order to write new code that uses the IPP library.

- *What are the redistributable files?*

  In general, the redistributable files include the linkable files (.DLL and .LIB files for Windows*, .SO files for Linux*) including the Runtime Installer. With your purchase of the Intel IPP product (and updates through the support service subscription), the redist.txt file outlines the list of files that can be redistributed.

- *Do I need to buy an the Intel IPP license for each copy of our software that we sell?*

  No. there is no per copy royalty fee. Check the Intel IPP end user license agreement for more details.

- *How many copies of my company's application can redistribute the Intel IPP library files?*

  You may redistribute an unlimited number of copies of the files that are found in thedirectories defined in the Redistributables section of the end–user license agreement.

- *Are there royalty fees in using the Intel IPP?*

  No, there is no royalty fee for redistributing the Intel IPP libraries with your software. By licensing the Intel IPP product for your developers, you have redistribution rights to distribute the Intel IPP library files with your software for an unlimited number of copies. For more information please refer to the end user license agreement.

- *How many copies of the Intel IPP product do I need to secure for my project team or company?*

  The number of copies of the Intel IPP product needed is determined by the number of developers who are writing code compiling and testing using the Intel IPP API as well as the number of build machines involved in compiling and linking, thereby needing the full development tools file set of Intel IPP product.

APPENDIX F

TIME SERIES RECORDING

F.1 Overview

The time series (TS) recording feature of the RVP8 provides end-users with the ability to record and playback IQ data. In addition, operators can record time series and then use their own "off-line" programs for processing the data. The time series can be recorded directly on an RVP8 or on a separate archive host via a gigabit network.

The time series recording is built on several software components:

- **RVP8 TS API**- where time series reside in memory in the RVP8.
- **Tsexport**- grabs time series from the TS API and sends them over the network via a UDP broadcast. Typically a gigabit network is used for this.
- **Tsimport**- receives UDP TS packets and recreates the TS API on a local machine.
- **Tsarchive (licensed separately)**- records time series to a local disk. This can be on the RVP8 or a separate networked archive host. Supports both archive and playback.
- **Ascope Utility**- can be used in playback mode to view either raw time series or processed results from the RVP8 processing algorithms.
- **Tsview Utility**- used to see diagnostic print-outs of stored time series files. Note that complete source code is provided for this to assist users with writing their own applications.

Of these components, only the **tsarchive** software is licensed separately. This means that customers are free to write their own TS record/playback software and still take advantage of the the network features such as **tsimport** and **tsexport**.

In this appendix:

F.2 TS Record/Playback Software Architecture

F.2.1 General Architecture

The TS record and playback features on a local RVP8 and a remote archive host, share a common software architecture. The most general case is shown below.

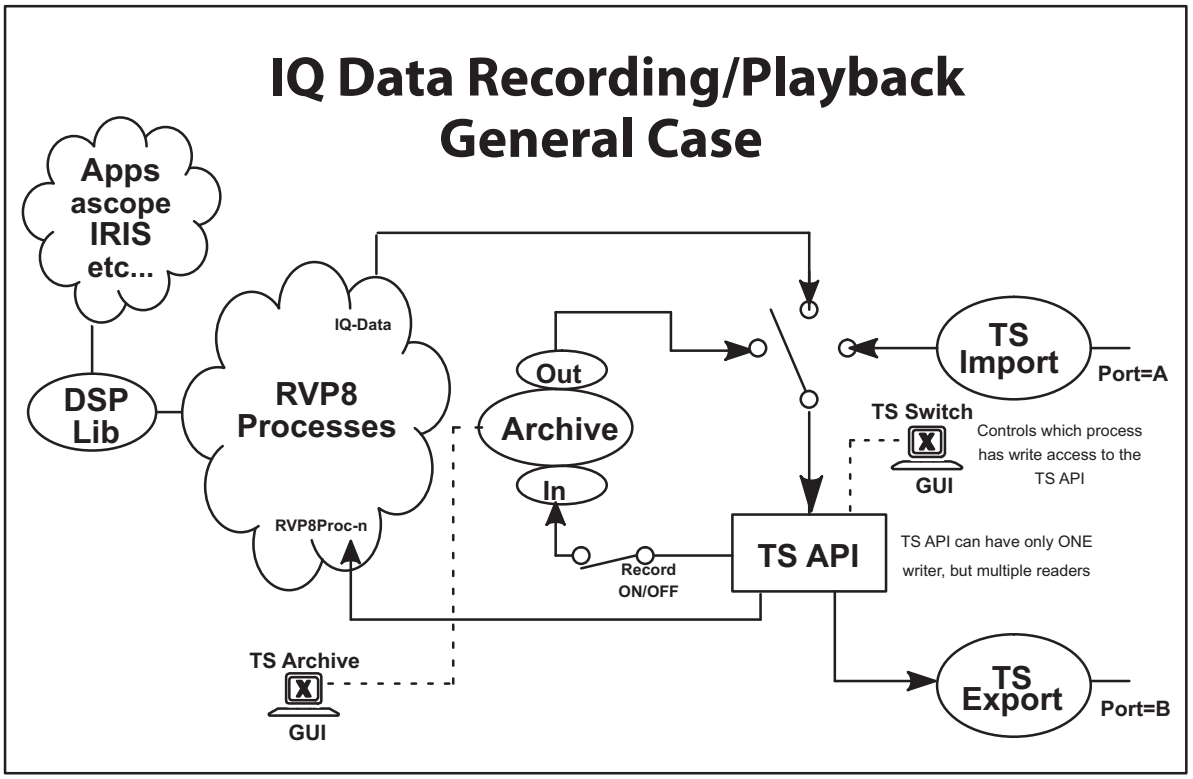


Figure 58 IQ Data Recording/Playback General Case

The structure shown above runs on a single machine. Replicas of this structure, with pieces included or excluded, can be run simultaneously on several machines to handle different scenarios, such as a separate archive host. This modular design does not really make a strong distinction between the RVP8 and a separate archive host operating in record or playback mode.

F.2.2 Description of Processes

The various process shown in the figure are described below.

TS API

Central to the architecture is the **TS API**. This is where the time series data reside in memory.

The time series can be written to the **TS API** from any one (but only one) of three sources:

- From a local RVP8
- From a local disk archive
- From a remote network host (via **tsimport**)

Tsswitch

The **tsswitch** GUI, allows the user to select the sole TS writer from among these three possible sources. This GUI is described in F.4 Tsswitch Utility on page 492.

Tsarchive

The **tsarchive** process handles both the recording and playback of data. It has its own GUI to select record/playback mode and which directory contains the local disk archive. It also shows an inventory of the TS files and has filter/search capability to locate specific groups of files (for example, by date and time). The **tsarchive** GUI record and playback features are described in F.5 Tsarchive Utility on page 493.

Tsimport and Tsexport

The **tsimport and tsexport** processes provide the ability to receive/send time series over a network. Note that either a 1000 BaseT (gigabit) or 100 BaseT Ethernet can be used depending on the typical mode of operation and the competing network traffic. For example,

1000 bins * 2 parameters/bin/pulse * 16 bits/parameter * 1000 pulses/sec = 32 Mbit/sec

Here the 2 parameters/bin/pulse are the I and Q values which are represented by 16 bits each (floating point). For dual polarization systems with two receiver channels, the data rate would be doubled. The 32 Mbit/sec basic data rate here would fit comfortable on a dedicated 100 BaseT network, that is., with little or no competing traffic.

The output is via UDP broadcast. This is a very efficient way to transfer data since there is virtually no overhead as compared to standard TCPIP. The socket ports are configured so that the **tsexport** on one system connects to the **tsimport**on another system. This configuration is described in F.3 Installation & Configuration on page 489.

RVP8 Processes

These are a collection of processes that are only present on an actual RVP8 machine. The two important functions indicated in the figure are:

- IQ–Data: writes real time TS to the TS API. These are collected from an RVP8/Rx and IFD.
- RVP8Proc-n: extracts TS from the TS API and processes the data to obtain the various moments.

These processes can be viewed using the v command in dspx. Of course a remote archive host will likely not be an RVP8. In this case these processes do not exist.

F.3 Installation & Configuration

Vaisala recommends the use of TS recording in two configurations. The simplest (but least flexible) configuration is where the RVP8 also serves as the archive host.

NOTE	A separate partition should be created to prevent system CRASHES.

However, the ideal configuration is to have a separate archive host with an increased amount of disk space and peripherals (that is tape drives). The two system configuration should be used in a high bandwidth environment. If this is not possible, then the RVP8 can be equipped with a separate disk for archive operations

If a two system configuration is used, then the TS IMPORT and TS EXPORT modules have to be turned on at boot time on both systems, and the UDP ports for the sending and receiving system have to be configured as well.

F.3.1 Required Software

The TS archive software is part of the Vaisalas RDA release which is installed by default on all RVP8s and RCP8s, but it is not installed on IRIS systems.

RVP8/RCP8	No additional software required
Archive System (IRIS)	Install RDA (make sure the keep old files button is pressed)
Archive System (w/o IRIS)	Install RDA

F.3.2 Configuring UDP Ports

The UDP ports can be configured by editing the tsimport and tsexport scripts installed in /etc/rc.d/init.d. The tsimport and tsexport files must be edited if you plan on using tsarchive with a separate archive host. The export port on the sending system must match up to the import port on the receiving system so that a connection can be made between the two port. Vaisala recommends the following ports:

RVP8	Archive Host
Tsexport: 30780	Tsexport: 30781
Tsimport: 30781	Tsimport: 30780

Each of the scripts contain extensive comments and should be edited depending on your configuration. The scripts are shipped configured for the RVP8 end, so you will need to edit the archive hosts files. You also need to edit all tsexport files to explicitly set the target IP address to which it will broadcast the time series. Vaisala recommends that you use a single target host. A broadcast address can be used, but be careful that all recipients can handle the traffic, and that there are no 10baseT network sections.

F.3.3 Configuring automatic start–up of tsimport & tsexport

As root, enter the following commands:

```
#chkconfig --add tsimport

#chkconfig --add tsexport
```

Once configured with the chkconfig command, you can start and stop these programs with the commands:

```
#service tsimport start

#service tsimport stop
```

F.3.4 Configuring Network buffering for tsimport

For **tsimport** to successfully read all the pulses of data from the network, the network read buffers must be enlarged. Do this by editing the /etc/sysctl.conf file and adding these commands:

```
net.core.rmem_default = 1000000

net.core.rmem_max = 4000000
```

Then reboot for this to take effect.

F.3.5 Tsimport and Tsexport from the command line

Tsimport and **tsexport** can also be run from the command line. Tsimport and tsexport take the following command line options:

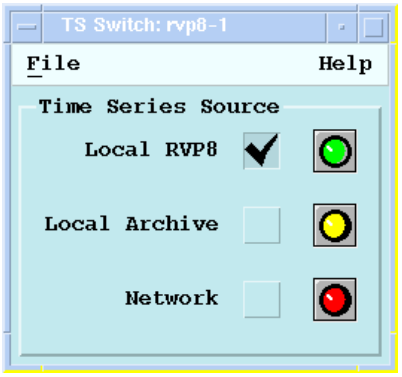
```
$ tsimport –help

tsimport command line options:

daemon              Run as daemon
debug               Print diagnostics
help                Print this list
port:<port>         Specify the port
                   number to use
All other options ignored


$ tsexport –help

tsexport command line options:

broadcast:<broadcastaddress>
daemon              Run as daemon
debug               Print diagnostics
port:<port>         Specify the port
                   number to use
All other options ignored
```

F.4 Tsswitch Utility

To start the tsswitch utility as operator, simply type:

$ tsswitch

In F.2 TS Record/Playback Software Architecture on page 486 it is described that only one of three processes may write to the **TS API**. The TS Switch utility, shown below, is used to select among the three possible sources:



Real time IQ from the RVP8/Rx Card. This setting is available only on an RVP8. This would be the choice for normal data processing, local RVP8 archive of real time IQ or export of real time IQ over the network.

Used to extract time series from the local disk archive. On an RVP8 these time series could be processed. On a separate archive host, these time series could be sent to an RVP8 for processing or perhaps a custom user application.

Used to collect time series from a networked RVP8 or archive host, via the TS Import process. This setting can be used either by an RVP8 (for playback) or an archive host (for recording).

The colored lights show the status associated with each of the sources:

- GREEN indicates that the selection of a source has been successful.
- YELLOW indicates that a source is available by not currently selected. If you select this source then it will change to green when the selection is confirmed.
- RED indicates that a source is not available. You may still select the source, but the color of the status indicator will remain red until it is enabled.

In the case of a separate archive host, the "Local RVP8" choice would always show as red, since there is not local RVP8.

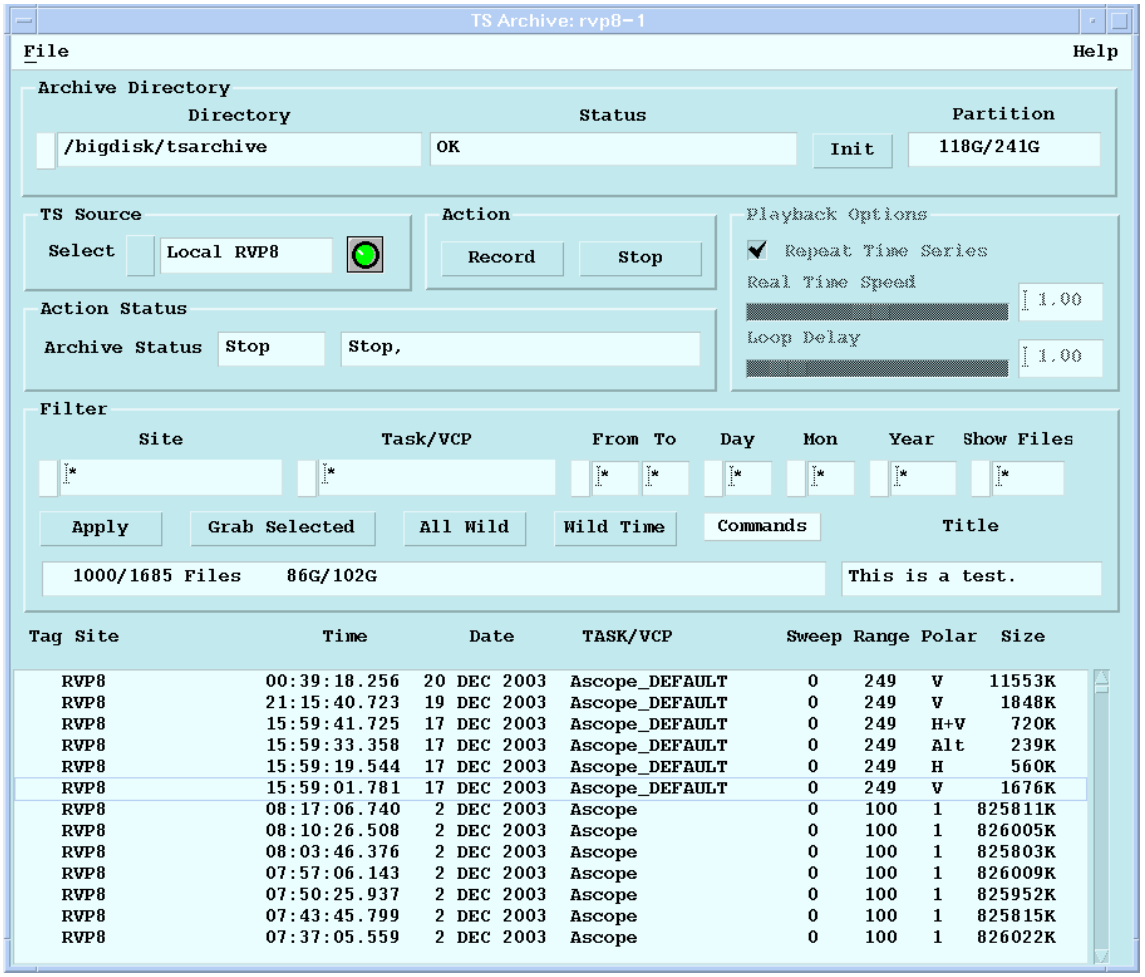
The "Local Archive" case and the "Network" case are possible for both an RVP8 and a separate archive host.

F.5 Tsarchive Utility

To start the utility, simply type (as operator):

```
$tsarchive
```

The **tsarchive** utility enables the end user to record and playback archived data. The utility provides a graphical user interface (GUI) including access to the TS switch utility, playback and record modes, and a complete archive file inventory for managing disk space.



F.5.1 Archive Directory Area

Directory

The directory section is used for selecting which archive directory to use, or to add another directory. It is recommended that you first create the directory from the command prompt.

```
$mkdir /bigdisk/tsarchive
```

Status

The Status section displays information about the chosen archive directory.

Init

The Init button initializes the directory by prompting the user for a title for the directory and also verifying the contents of the directory. Initialization requires that the directory be empty, or that it contains ONLY TS archive files. If a directory contains TS archive files, then the user will be prompted for permission to delete them. The initialization feature can ONLY delete TS data files for system security reasons. Note that if there are other non TS archive files in the directory, then these must be manually deleted using the UNIX "rm" command.

Partition

The Partition field displays the ratio of used disk space over total disk space for the selected archive directory partition. For example, if the selected archive directory is /bigdisk/tsarchive, the Partition field will show the used/total disk space for /bigdisk/. The values displayed here are for all types of files in the partition, not just the TS archive files. The filter area provides information on the amount of storage used for just the TS archive files alone.

F.5.2 TS Source

The TS Source area is used to select which source is used to record data. This button executes the TS switch utility. For more information, see F.4 Tsswitch Utility on page 492.

Action & Action Status

Once the source has been selected, data can be recorded or played back. Pushing the record button begins the data recording. The inventory at the bottom of the screen will update with TS file information. T he Playback

button is only displayed when the Local Archive source is selected. The Stop button terminates all recording or playback operations. The "Action Status" displays the current mode of the utility and provides information on what data are being recorded or played back.

Playback Options

The Playback Options section will only be enabled when Local Archive has been selected from the TS Source Area.

- **Repeat Time Series** (if selected) will repeat the selected files.
- **Real Time Speed** controls the speed at which the files are played back.
- **Loop Delay** controls the length of the pause in between each repetition.

F.5.3 Filter

The filter area of the utility provides end users with the ability to manage the files that are stored on disk. In particular, the filter functions are used to display data for a certain Site, Task and time .

Site

Enter a site ID in this field to select data from only one site, or enter the wildcard character to select data from all sites.

Task

Here Task refers to either an IRIS TASK name (sometimes called a VCP or volume control procedure) or an ascope saved configuration file name. Enter the name of a TASK in this field to narrow the list down to only those products generated by a specific TASK. You can include wildcard characters in the TASK name. A question mark (?) matches any single character; an asterisk (*) matches any sequence of zero or more characters. For example, entering a TASK name of *PPI*_? selects all hybrid TASKS that have PPI somewhere in their names.

From, To

Enter a range of hours in these two fields to narrow the list of products by time of day. You can pop up a list of ranges to choose from.

Day, Month, Year

Enter a day, month. or year to narrow the list of products by date. You can pop up a list of values to choose from.

Show Files

The Show files field lets you control how many files to include in the list.

Apply

Click on the Apply button to update the file list, based on your selection criteria. The Apply button is enabled only if the Filter button is pushed in. If the Apply button is grayed out, click on the Filter button.

Grab

If a product is selected, the Grab button inserts a selected files information into the filter fields.

All Wild

Click on the All Wild button to return all the fields to the wildcard character.

Wild Time

Click on the Wild Time button to change only the hours, month, day and year fields to the wildcard character.

Commands

Pops up the following list of operations that you can perform on the TS archive files selected by the Filter menu.

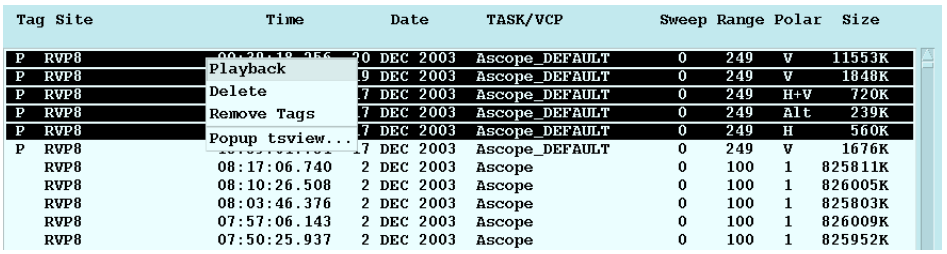
Playback Tags files for playback. A "P" shows in the left column

Delete Deletes files. A "D" shows in the left column.

Remove Tags Untags any delete or Playback tags.

CAUTION	The commands apply to all of the TS archive files that match the filter. If you put in wildcards everywhere and command "Delete", then every TS archive file could be deleted- that is why there is an "Are you sure" prompt.

F.5.4 TS Archive Log Area



Tag & Right–Click Menu

When you right–click on any selected file or group of files (as shown in the above picture), you are prompted with 4 commands: Playback, Delete, Remove Tags, and Popup Tsview.

- **Playback** marks the files with a P in the Tag Column and when the playback button is pressed, only the tagged files will be played.
- **Delete** marks the files with a D in the Tag Column and then the files are deleted. This is helpful when a large list of files has been selected for deletion.
- **Remove Tags** clears the D or P tags
- **Popup tsview** provides easy access to file information in another window. For more information see section ...

Site

The Site field displays which TS site was used to create the data.

Time & Date

The time & date fields display the UTC time & date when the data were acquired.

Task/VCP

This field displays the name of the associated tasks or ASCOPE file used for controlling the RVP8.

Sweep

This field displays the number of sweeps (for example, 360 degrees) in the Task/VCP.

Range & Polar

These fields display the range and polarization of the TS data.

Size

This field displays the size of the file.

F.6 Specific Software Application Examples

There are four specific example applications described here:

1. – TS recording on a local RVP8, see F.6.1 Case 1: TS recording on a local RVP8 on page 499
2. – TS recording on a separate archive host, see F.6.2 Case 2: TS recording on separate archive host on page 500
3. – TS playback on a local RVP8, see F.6.3 Case 3: TS playback on a local RVP8 on page 501
4. – TS playback from a separate archive host to an RVP8, see F.6.4 Case 4: TS playback from a separate archive host to an RVP8 on page 502

These are the only applications that are supported. In the example descriptions, unused processes are omitted for clarity.

For reference, the case of an RVP8 in normal operation, is shown below.

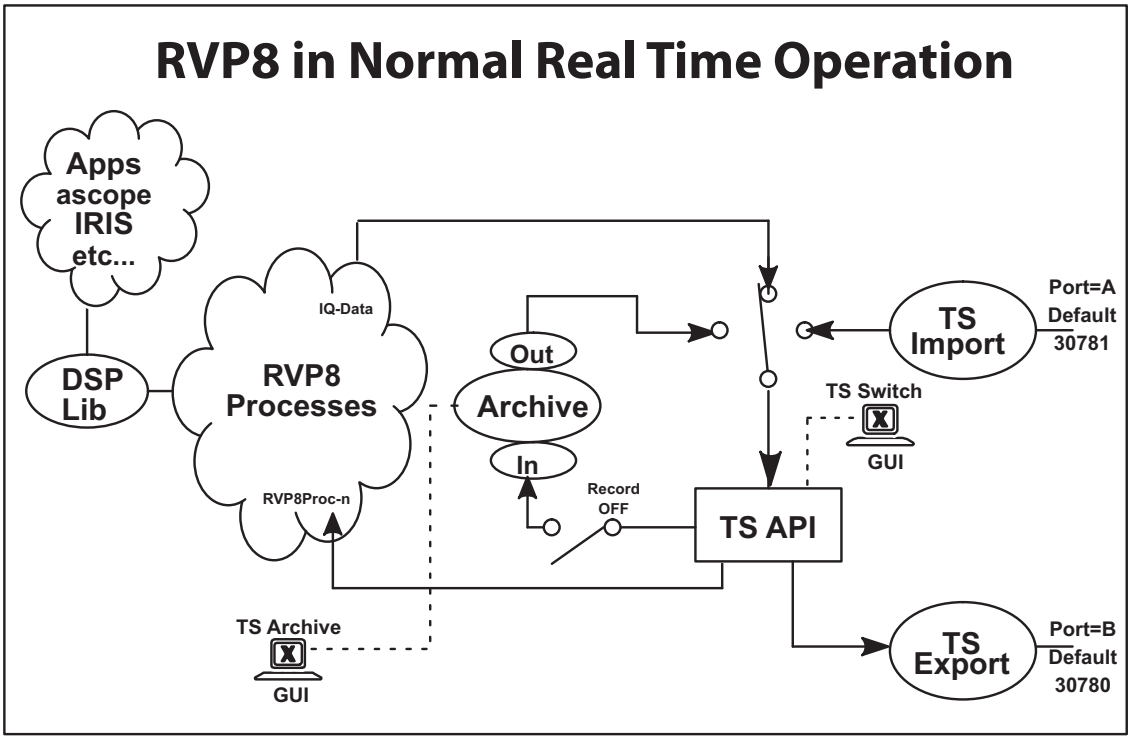


Figure 59 RVP8 in Normal Real Time Operation

In this case, the TS Switch is set to write real time data from the IQ-Data process to the TS API. The RVP8Proc-n then extract time series data and process it. Configuration information is obtained from and data are passed to the various user applications via the DSP Lib functions. Note that the TS Export process, if started, can extract data simultaneously from the TS API.

Utility Settings:

TS Switch:	Local RVP8
TS Archive:	N/A

F.6.1 Case 1: TS recording on a local RVP8

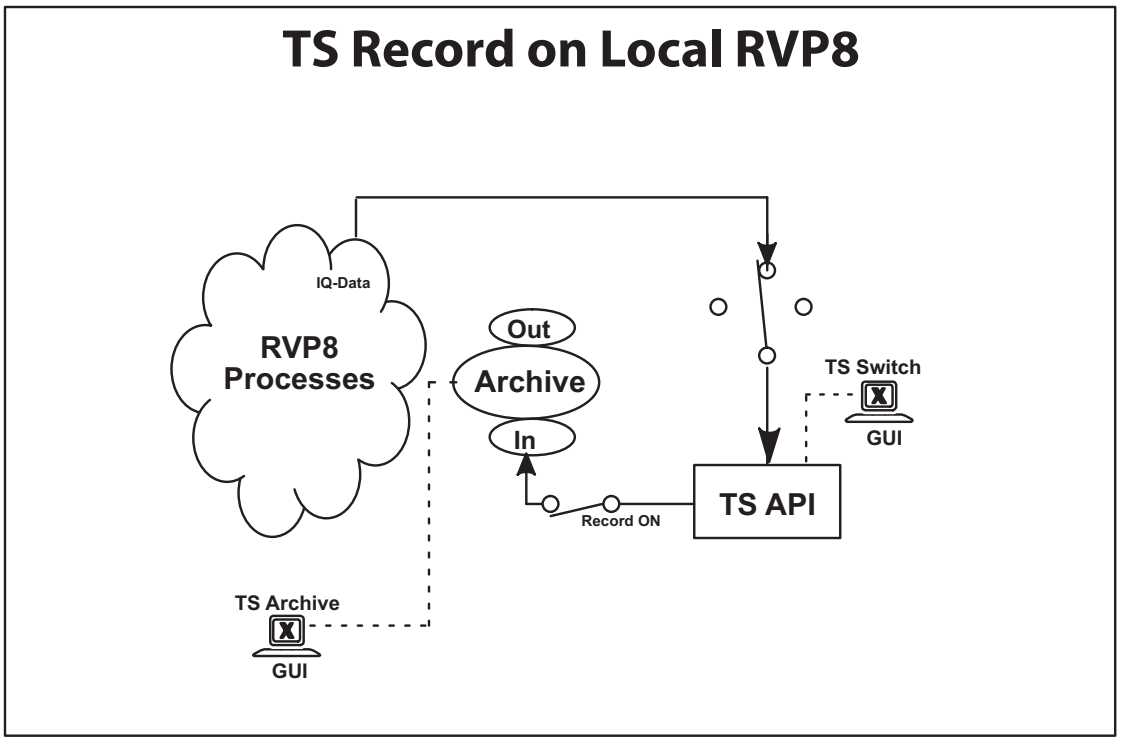


Figure 60 TS Record on Local RVP8

In this case, the TS Switch is set to place the real time IQ values from the IQ-Data Process into the TS API. These are extracted and recorded to local disk by the TS Archive process. Note that the RVP8 may still do its normal data processing tasks, as shown on the previous page, while TS data are being recorded.

Utility Settings:

TS Switch:	Local RVP8
TS Archive:	Record

This configuration records to a local disk on the RVP8. Typically only a 20 GB disk is provided with a standard RVP8 so recording capability is limited unless a large custom disk is added. In either case, the recording should be done to a dedicated data partition, NOT the "/" partition. The reason for this is that if the "/" partition fills-up, the system will crash. However, if the separate data partition fills-up, then the system will stop recording, but otherwise function normally.

F.6.2 Case 2: TS recording on separate archive host

This is the recommended recording configuration for TS recording.

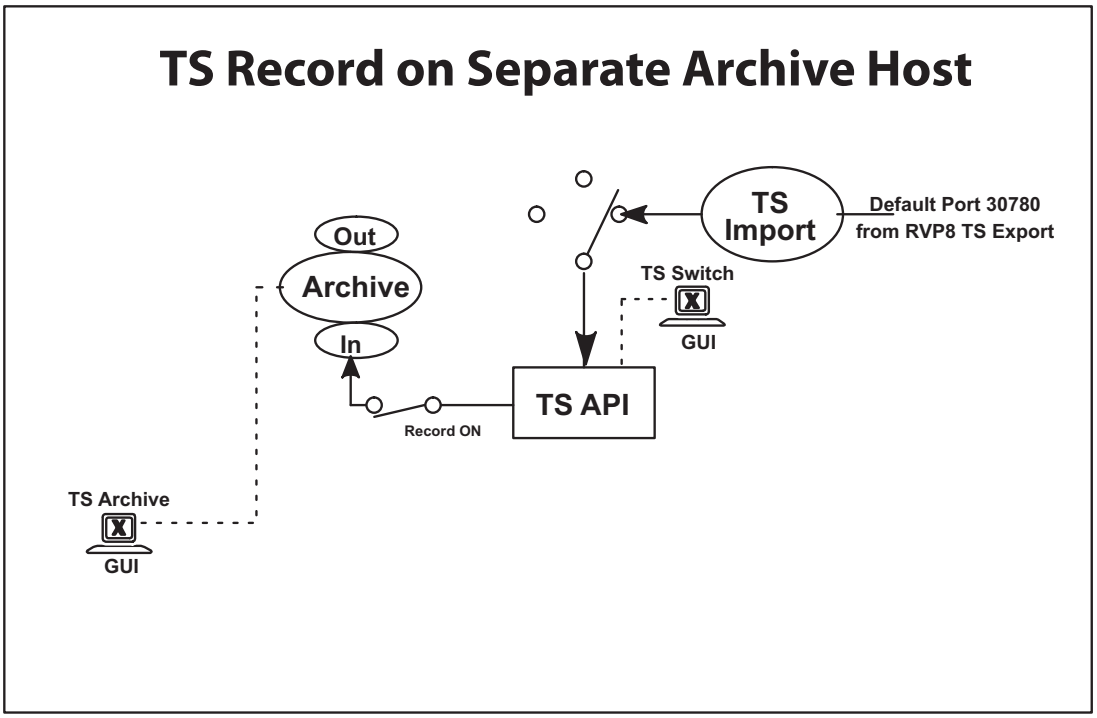


Figure 61 TS Record on Separate Archive Host

In this case TS Switch is set to write IQ data from the TS Import network source. Typically this is a 100 or 1000 Base T LAN connection. The time series are placed on the network via UDP broadcast by the TS Export process on a networked RVP8.

Utility Settings:

RVP8
TS Switch: Local RVP8
TS Archive: N/A
Archive Host:
TS Switch: Network
TS Archive: Record

The advantage of having a separate archive host is that it is easy to install a large disk that is dedicated to time series recording without having record/playback/backup operations interfere with the normal operation of an RVP8. Note that there can be multiple archive hosts on the network.

F.6.3 Case 3: TS playback on a local RVP8

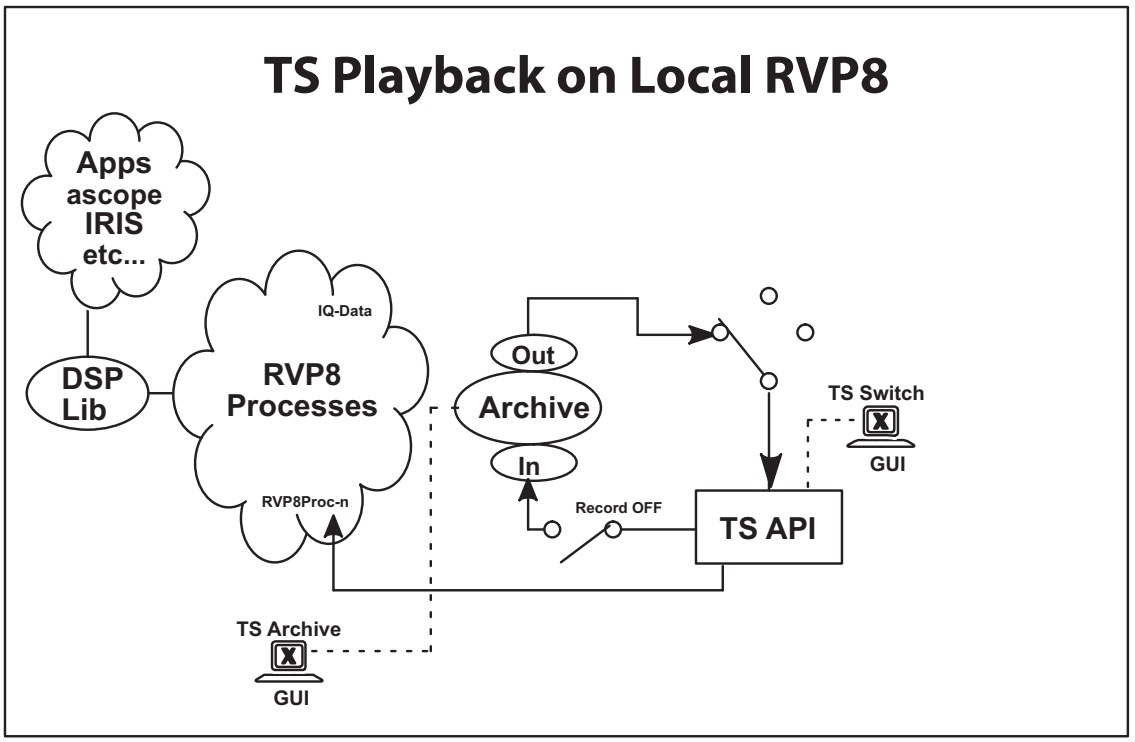


Figure 62 TS Playback on Local RVP8

Here, the TS Switch is set to write data from the TS Archive to the TS API. The RVP8Proc-n process(es) then read the time series data from the API. Note that this is essentially identical to the real time normal operation case shown at the beginning of this section except that the archive is the source rather than the real time operation.

Utility Settings:

RVP8
TS Switch: Local Archive
TS Archive: Play

The difference is that applications that use the time series must know that the data are playback rather than real time data. This information as well as all of the required housekeeping data are actually part of the time series data format.

F.6.4 Case 4: TS playback from a separate archive host to an RVP8

This is the recommended mode of operation

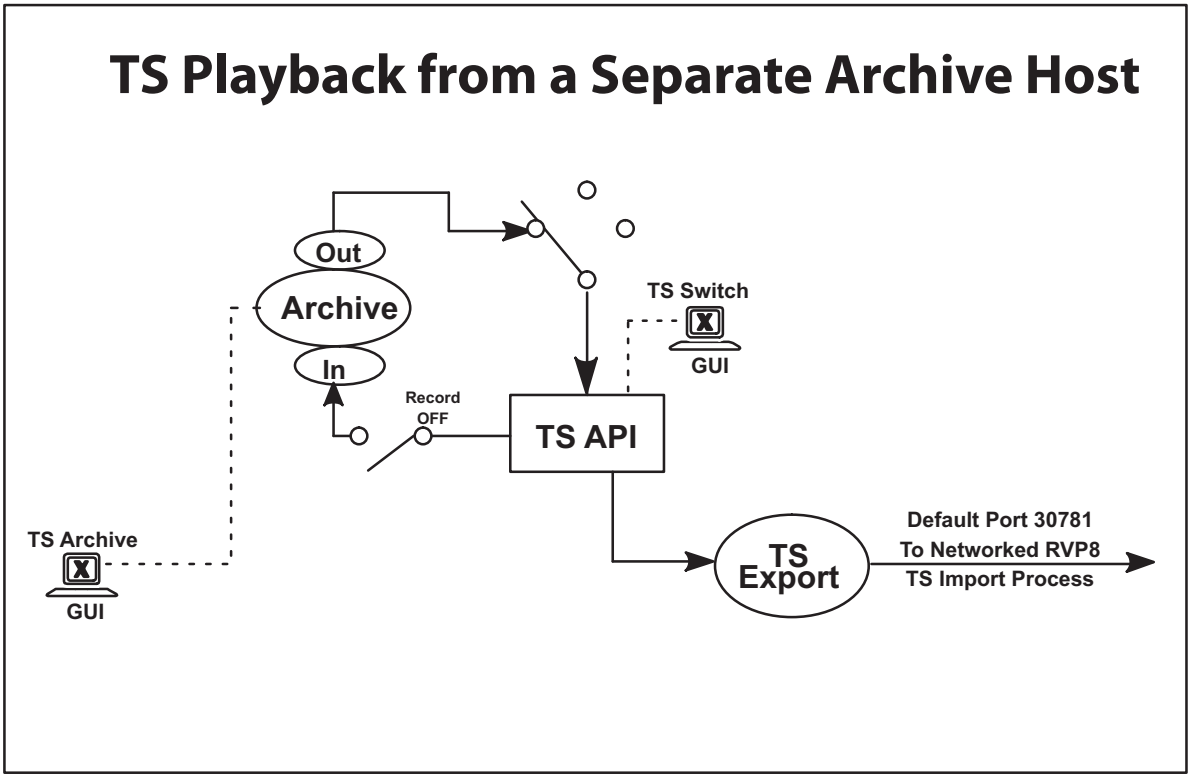


Figure 63 TS Playback from a Separate Archive Host

Here the TS Switch is set to write data from the TS Archive to the TS API. TS Export then sends these via a UDP broadcast over the network to an RVP8 for processing.

Utility Settings:

RVP8

TS Switch:	Network
TS Archive:	N/A
Archive Host:	
TS Switch:	Local Archive
TS Archive:	Play

The diagram for the corresponding RVP8 would be identical to the previous page (Case 3: Playback from Local RVP8), except that the TS Switch would be set to write data from the TS Import process.

F.6.5 Quick Guides

The following steps should be used in conjunction with the software specific application examples described in the previous sections.

TS Archive Recording Quick Guide

- Select a directory for the TS data to be written to. Make sure the desired directory already exists in the file system.
- Initialize the directory and give it a title. If the directory has already been used for archiving, then there is no reason to re–initialize it unless you want the data to be erased.
- Select the data source and make sure the light is green.
- Press record and watch for data to arrive in the log section.

TS Archive Playback Quick Guide

- Select the directory that contains the TS data to be played.
- Select Local Archive in the TS Source section.
- Make sure that the light is green.
- Select the files that you wish to playback and (right–click and select Playback) mark them for playback.
- Select any playback options.
- Press the Playback button.

F.7 Ascope Playback Features

The **ascope** utility is a full-featured, stand-alone signal processor configuration and plotting utility. When an RVP8 is in playback mode, **ascope** can be used to configure the processing of the playback data and display the results. For more information on the **ascope** utility please refer to Chapter 3 of the *IRIS Utilities Manual*. The figure below illustrates the differences in the meanings of various menu fields when the RVP8 is in playback mode.

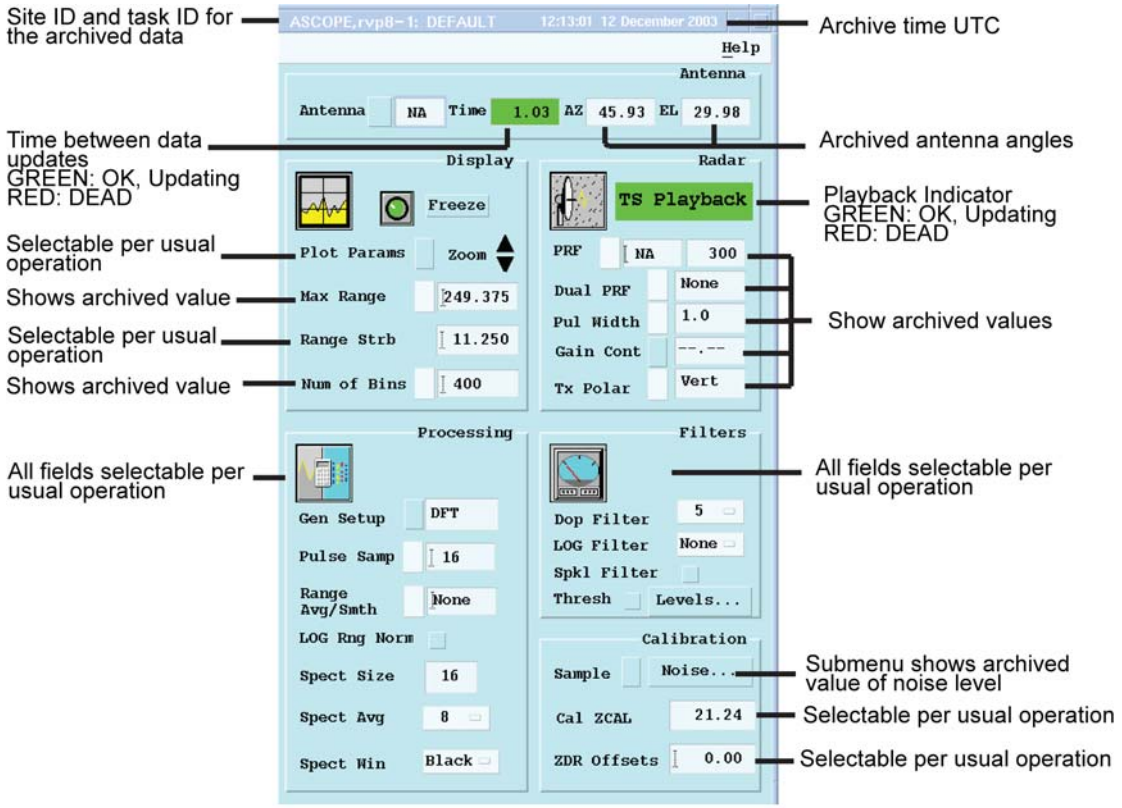


Figure 64 Ascope Differences during RVP8 TS Playback

When the RVP8 is in playback mode, vs processing real time data, the major difference is that, some parameters were fixed at the time when the data were recorded. Examples of fixed parameters are:

- Maximum range
- PRF
- Transmit polarization
- Number of range bins
- Phase coding
- Value of noise level

All of these relate to the transmit characteristics of the radar and for these, **the archived value is displayed — users may not alter the fields**. There are other parameter fields that can be changed, identical to using **ascope** for real time data, such as the parameters to configure processing and plotting, for example, the processing major mode and clutter filtering. The user may freely select these while the RVP8 is in playback mode.

There are several ways to use ascope during RVP8 playback.

- The archive can be on the local RVP8.
- The archive can be on a separate archive host.

Note that it does not matter where ascope is run, that is, either on the local RVP8 or on a networked host computer via **DspExport**.

Archive on Local RVP8

Utility Settings:

RVP8

TS Switch:	Local Archive
TS Archive:	Play

Network Note: In most cases you will not be sitting up at the radar so you will need to export the displays for these utilities over the network. There are two ways to do this:

- Easy way: Use sigterm <hostname> to open a terminal window, or manually
- rlogin and export the display with DISPLAY=<hostname>:0.0 (you may also need to type xhost + on your local workstation).

It is often convenient to get the playback going in TS Archive with "Repeat" set, and then start **ascope**. **Ascope** will start-up in the playback mode and be updating the display appropriately. You are free to select the data processing and display parameters "on-the-fly" as playback is continuing and repeating.

When **ascope** is set the way that you want it, you can use the TS Archive menu to select different files and/or restart the playback.

Archive on Separate Archive Host

In this case the various support menus would be set as follows:

RVP8
TS Switch: Network
TS Archive: N/A
Archive Host:
TS Switch: Local Archive
TS Archive: Play

F.8 TS Playback using IRIS

Successful playback of time series data using IRIS takes some configuration. First you must be running version 8.09 or later. Second, make sure the following is configured in **setup:**

Ingest pop-up, *Signal Processing and Data Storage section*

"Source of recorded angles" = "RVP Tags"

"Source of recorded time" = "RVP Tags"

General pop-up, *Modes and Protocols section*

"Timezone for data recording" = "UTC"

General pop-up, *Scanning Options section*

"Task Scheduling Control" = "Active/Passive" or "Passive"

"Passive type" = "TS–Playback"

Because IRIS sorts stored data by time and playback version number it is important to only playback the data once. In the ingest summary menu, it will display a 2-digit number containing the playback version after the site code. Original data with a playback version of zero, will not include the number. The ingest filenames will all have appended a "V" followed by the 2-digit playback number. Below is a procedure to play back data:

Launch **tsarchive** on your RVP8 machine. Push the "TS Source" button to launch the **tsswitch** GUI. In **tsswitch** select "Local Playback". Back on **tsarchive** in the "Playback Options" section, be sure to turn off the "Repeat" button. Also set the "Version" to a unique non-zero value. Select from the inventory the file or files you wish to playback, and set the "P" bit on those. If you are playing back a multiple sweep volume scan there will be one file per sweep.

Now using the IRIS menubar, bring up the Task Scheduler Menu. Be sure that it says "Passive" on the top menubar. Select the task which matches the data. Select "Go/Schedule" for that task. It will toggle to "running" and wait for data to arrive.

Back on **tsarchive**, push the "Playback" button.

Back on the Task Scheduler Menu, set the task to "Stop (when done)" so that it stops after one time.

F.9 TS Viewing Utility (tsview)

F.9.1 Overview

The `tsview` utility allows users to specify a TS file and obtain print-out of header information and time series data on a terminal screen. This capability is usually used to view header information to see how the data were collected, or to verify that IQ data were indeed recorded. Since the `tsview` source code is provided, perhaps the most important use of the `tsview` software is to serve as a model for developers building their own off-line applications for reading and processing time series data.

NOTE	If the TS archive utility has been licensed, the tsview utility can be accessed by right–clicking on the desired file in the TS inventory area. See F.5.4 TS Archive Log Area on page 497.

F.9.2 Starting tsview and Sample Session

You must have operator privilege to run tsview. The TS View utility must be installed on the same node where the IQ data are archived. Before starting `tsview` it is usually most convenient to `"cd"` to the directory where the archive files are located. This will save you from having to specify the full path and allow the X-window center button copy/paste technique to be used to avoid typing file names. Once you are in the archive directory you can issue an `"ls"` command to view the files, and then run `"tsview <pathname> <-options>."` To see the various options simply use the `-help` option, that is,

```
$ tsview -help
```

```
$ tsview          help
Options:          count:N          (only print N
                                   pulses)
                  data             (print data values)
                  help             (print this list
                                   and exit)
                  length:N         (max line length to
                                   use)
                  skip:N           (skip the first N
                                   pulses)
                  verbose          (print full header
                                   info)
                  noExit           (do not exit when
                                   done)
                  pathname
                  (all other arguments ignored)
```

A sample session is provided below. In this session you will view the pulse header information for the 101st pulse in a TS file. The archive directory is called `"/bigdisk/tsarchive"`:

`$ cd /bigdisk/tsarchive`*Change directory to location of TS files*

`$ ls *20031208.192519*`*List all files from 8 Dec 03 19:25:19*

`RVP8.20031208.192519.074.Ascope_DEFAULT.0.H.249`*One file found*

`$ tsview RVP8.20031208.192519.074* –count:1 –skip:100 –v`*run tsview*

`Pulse Info size:1148`*tsview output*

```
==============TS Pulse Info==============

Site:RVP8

Version:0

Major Mode:1 (FFT)

Polarization:1 (Vertical)

Phase mod sequence:0 (Fixed)

Task Name:Ascope_DEFAULT

Sweep Number:0
```

```
. . .
```

Details of the header format and information content are provided in F.10 TS Record Data Format on page 511. The file naming convention is discussed in the next section under the heading "pathname".

F.9.3 Tsview Command Line Options

`-help`

Gives a list of available options as shown on the previous page.

`pathname` (*and file naming convention*)

This is the directory and name of the TS file. As shown above in the sample session, it is easiest to `"cd"` to the directory where the archive files are stored, then you only need to provide the file name. Once there you can do an `"ls"` command searching for the files that you want by date and time using standard UNIX options for `"ls"`. Note that "*" is a wild card. The file names are very long so you dont want to type them. Instead, use the X–window trick of highlighting the file name and then clicking the middle button to copy the text to the terminal cursor location.

The reason why the file names are so long is that the file name format is designed to make it easy to identify TS files. In the example we used a file with the name:

```
RVP8.20031208.192519.074.Ascope_DEFAULT.0.H.249
```

The file name format is:

```
site.YYMMDD.HHMMSS.SSS.taskname.sweep.polarization.maxrange
km
```

Most of the fields are self-explanatory. Details for the less obvious fields are given below:

- **site**- This is the site name typed into the setup program on the RVP8 which generated the data. Both site and taskname fields are preprocessed to remove characters which would mess up the filename, such as unprintable characters, space and "/".
- **taskname**- This is the identification of the configuration of the application that was operating when the TS data were collected. In the case of IRIS, it is the IRIS TASK name. In the case of the **ascope** utility, it is set to "ascope_<filename>" where filename is the name of the saved ascope configuration that was used to collect the data (see example above). In the case of a custom user application, the user

could specify an appropriate ID for the configuration so that it is archived appropriately.

- **sweep**- A sweep is a full 360 degree (or partial sector in sector mode) of data, typically collected at a fixed elevation angle. Most data acquisition software packages such as IRIS, collect volume scan data this way. The sweeps are indexed 1, 2, 3, ... In the case where **ascope** is used for RVP8 operation, there is no concept of a "sweep" and the sweep number is set to "0". Note that for RHI scanning, the concept of a sweep is the same, except that it is elevation sweep rather than azimuth sweeps.
- **polarization**- This refers to the transmit polarization. There are three choices: H, V, H+V (simultaneous H and V transmit) and ALT (alternating H and V transmit).

`-count:N`

Each file consists of the IQ data for all range bins for each pulse in the sweep. "Count" is how many pulses we want to display. In the example below in the `-data` section, the count is set to 1, that is, only the information for one pulse is displayed.

`-data`

Use the `-data` option to see the actual IQ values for each range bin. Here are the time series values for 400 bins for the 101st pulse. The label on the left is the index number of the first range bin on the line. The numbers displayed are the power of the pulse in dBm, and the angle in degrees of the IQ vector.

```
$ tsview RVP8.20031208.192519.074* -count:1 -skip:100 -data

Site:RVP8

Pulse #101 at:19:25:19.406 8 DEC 2003 UTC, Az: 9.99, El:29.98

0:        (91.2,244) (91.2, 0)  (80.4,209) (87.4,149) (82.2,150) (79.9, 95)
6:        (84.2,157) (81.4,182) (81.6,100) (83.8,276) (79.2,331) (83.9,220)
12:       (83.2,202) (84.9, 53) (78.7, 52) (82.3,184) (82.7,121) (78.7,286)


              ...

390:      (90.2,244) (83.5,228) (83.7,264) (86.4,181) (85.4,182) (84.8,206)
396:      (84.5, 19) ( 118,233) (81.0,149) (96.5,256) (90.9,249)
          Bin 396    Bin 397    Bin 398    Bin 399    Bin 400
```

```
-length:N
```

Since there are a lot of bins, and you may be using a terminal window with either a large or small font, the "-length" option allows you to specify the maximum number of characters to display under the "-data" option. In the example below, the length is set to 43 characters.

```
$ tsview RVP8.20031208.192519.074* -count:1 -skip:100 -data
-length:43

Site:RVP8

Pulse #101 at:19:25:19.406 8 DEC 2003 UTC, Az: 9.99, El:29.98

0:              (91.2,244)        (91.2, 0)         (80.4,209)
3:              (87.4,149)        (82.2,150)        (79.9, 95)
```

```
-skip:N
```

Used to specify how many pulses to skip before starting the terminal listing. In the example, we wanted the 101st pulse, so we specified "skip:100".

```
-verbose
```

To see the detailed information contained in the headers. This was enabled in the example session.

F.10 TS Record Data Format

Each TS file recorded to disk contains a run of 1 or more pulses which are from the same basic configuration of the RVP8. In RVP8 nomenclature, this is called the "Acquisition Mode". The definition of an acquisition mode is stored in a structure called the "rvptsPulseInfo". Each time something is changed, such as the PRF, the acquisition mode will change, and a new file is created. If there are no changes, then files will be arbitrarily written every 200000 pulses.

TS files consist of an interesting mixture of ASCII headers and binary data as shown in Table 42 on page 512. They start with the rvptsPulseInfo structure. This is followed by possibly a great many pulses. Each pulse has a rvptsPulseHdr structure followed by an array of 16-bit binary data.

Table 42 TS File Format

<rvptsPulseInfo> Variable size, even
<rvptsPulseHdr #1> Variable size, even
Pulse Data #1 16-bit words, count from header
<rvptsPulseHdr #2> Variable size, even
Pulse Data #2 16-bit words, count from header
...

Each individual time series sample consists of 2 floating point numbers representing the I followed by the Q voltage. The values are scaled such that the full magnitude is a value of 1. This represents +6dBm on the IFD rev D, but may change in future revisions. Floating point numbers are packed into 16-bit words using "High SNR" packed floating format described in 7.7 Initiate Processing (PROC) on page 295. These 16-bit words are always stored in the little-endian byte order which is native to the Intel processor chips common on PCs, which is the reverse of "Network order" used on sockets. Note that tsview displays the (I, Q) samples in power and angle format, as follows:

$Power$ - $6dBm + 10 \times \log_{10}[I^2 + Q^2]$

$Angle - atan2(Q, I)$

The first time series sample number is from the burst pulse. This is followed by a sample from each range bin with data. The iNumVecs field in the pulse hdr indicates the total number of samples. If is is a dual polarization receiver system, then this is duplicated for the second receiver (the iVIQPerBin field in the pulse hdr). Thus the total number of bytes of data is:

$Bytes$ - $2 \times 2 \times iNumVecs \times iVIQPerBin$

Note that the number of sample can be different in each pulse within the same file. This is because the sampling stops when the next trigger arrives. If triggers are from an external source, the PRT may fluctuate.

To explain the rvptsPulseInfo structure, we give an example from the file. You can find more details in our header file rvpts.h

rvptsPulseInfo start	The structure is bracketed by start and end
iVersion=0	Structure version number
iMajorMode=1	1:FFT, 2:Random Phase (see dsp.h)
iPolarization=1	Transmit polarization: 0:H, 1:V, 2:Alt, 3:H+V
iPhaseModSeq=0	See dsp.h
taskID.iSweep=0	Application Sweep number
taskID.iAuxNum=0	Application auxiliary number
taskID.sTaskName=Ascope_DEFAULT	Application task name
sSiteName=RVP8	Site name of RVP8
iAqMode=161	Increments each time there is a change
iUnfoldMode=0	Dual-PRF flag, see PRF_* in dsp_lib.h
iPWidthCode=0	Pulse width index (0–3)
fPWidthUSec=1	Actual pulsewidth in microseconds
fAqClkMHz=35.9751	Actual pulsewidth in microseconds
fAqClkMHz=35.9751	Acquisition clock rate
fWavelengthCM=10.7	Radar wavelength in cm
fSaturationDBM=6	Saturation power of the I & Q samples
fRangeMaskRes=125	Range mask resolution in meters
iRangeMask=33825 ...	Full range mask, up to 512 16-bit numbers
fNoiseDBm=–81.6584 –81.6584	Noise samples for the 2 channels
fNoiseStdvDB=–0.00540576 –0.00540576	Standard deviation of the noise samples
fNoiseRangeKM=525	Range at which the last noise was taken
fNoisePRFHz=250	PRF at which the last noise was taken
iGparmLatchSts=0 0	Latched status from GPARM command
iGparmImmedSts=21124 8963 771 19 0 0	Immediate status from GPARM command
iGparmDiagBits=0 0 0 0	Diagnostic results from GPARM command
sVersionString=8.04.4	Version of RVP8
rvptsPulseInfo end	

The rvptsPulseHdr structure is also defined in the rvpts.h file. Here is an example:

rvptsPulseHdr start

iVersion=0

iFlags=3

Bit 0: N/A

Bit 1: Gap before this pulse

Bit 2: First pulse in trigger bank

Bit 3: Last pulse in trigger bank

Bit 4: Trig bank (possibly unchanged) is just beginning

Bit 5: Triggers were blanked on this pulse

iMSecUTC=179	The data acquisition time milliseconds
iTimeUTC=1071875957	The data acq. time in seconds since 1970, UTC
iBtime=2429100475	Ms time pulse inserted in API
iSysTime=45973182	Acq clock count of pulse acquisition
iPrevPRT=119917	Acq clock period from previous trigger
iNextPRT=119917	Acq clock period to next trigger
iSeqNum=287828	Sequence number of pulse in API
iAqMode=161	Acq Mode sequence number (8-bits)
iPolarBits=0	Polarization control bits
iTxPhase=182	Transmit phase 1 deg (16-bit binary angle)
iAz=16381	Azimuth=89.98 deg (16-bit binary angle)
iEl=179	Elevation=0.98 deg. (16-bit binary angle)
iNumVecs=401	The number of TS samples in this pulse
iMaxVecs=401	The max possible number (1+#bins requested)
iVIQPerBin=1	1 for single polarization, 2 for dual

iTgBank=0	Trigger bank number
iTgWave=0	Trigger waveform sequence number
uiqPerm.iLong=0 0	User tag bits, permanent
uiqOnce.iLong=0 0	User tag bits, one time
RX[0].fBurstMag=3.58298e–05	Burst pulse amplitude, 1=full scale 0.446Volts
RX[0].iBurstArg=45561	Burst pulse phase difference (previous–this)
RX[1].fBurstMag=0	Second receiver burst info
RX[1].iBurstArg=0	
rvptsPulseHdr end	

APPENDIX G
REFERENCES AND CREDITS

1. Tang Dazhang, et.al. (1984). Evaluation of an Alternating–PRF Method for Extending the Range of Unambiguous Doppler Velocity. *Preprints of the Conference on Radar Meteorology, 22nd, 1984* pp.523–527.
2. Joe, Passarelli and Siggia (1995). Second Trip Unfolding by Phase Diversity Techniques. *Preprints of the Conference on Radar Meteorology, 27th, 1995* pp. 770–772.
3. Doviak, R. J., and Zrnic, D. S. (1993). "Doppler Radar and Weather Observerations." Academic Press, San Diego.
4. The JMA internal specification for Interference Filter algorithm for use on Chitose airport Doppler weather radar is the basis for Alg.1 described in 6.1.5 Interference Filter on page 212.
5. Environment Canada – Aldergrove BC, kindly supplied the snapshot of receiver data that is plotted in Figure 25 on page 183.
6. Sachidananda, M., D. S. Zrnic, and R. J. Doviak, 1997: Signal design and processing techniques for WSR–88D ambiguity resolution. *National Severe Storms Laboratory* Report, Part 1, Norman, OK, 73069. July, 100 pp.

INDEX

www.vaisala.com

M211321EN