VAISALA

USER'S MANUAL

RVP900™ Digital Receiver and Signal Processor

M211322EN-F

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

CHAPTER 1
ABOUT THIS MANUAL

This manual provides technical information for installing, operating, and maintaining the RVP900 Digital IF Receiver and Signal Processor.

1.1 Version Information

Manual Code	Description
M211322EN-E	This manual. Fifth version. October 2015
M211322EN-D	Previous manual. Fourth version. September 2014
M211322EN-C	Previous manual. Third version. November 2013
M211322EN-B	Previous manual. Second version. March 2013
M211322EN-A	Previous manual. First version. June 2012

1.2 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
M211321EN	RVP8 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.

1.3 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.

`prompt`—Some features of the RVP900 operate by displaying questions and waiting for you to type an answer. The text of prompts is displayed in bold, monospace type.

1.4 Safety

The Vaisala RVP900 is delivered to you has been tested and approved as shipped from the factory. Note the following precautions:

CAUTION	Do not modify the unit. Improper modification can damage the product or lead to malfunction.

1.4.1 ESD Protection

Electrostatic Discharge (ESD) can cause immediate or latent damage to electronic circuits. Vaisala products are adequately protected against ESD for their intended use. However, it is possible to damage the product by delivering electronic discharges when touching, removing, or inserting any objects inside the equipment housing.

To make sure you are not delivering high static voltages yourself:

- Avoid touching exposed connectors unnecessarily.

- Handle ESD sensitive components on a properly grounded and protected ESD workbench.
- When an ESD workbench is not available, ground yourself to the equipment chassis with a wrist strap and a resistive connection cord.
- If you are unable to take either of the above precautions, touch a conductive part of the equipment chassis with your other hand before touching ESD sensitive components.
- Always hold the boards by the edges and avoid touching the component contacts.

1.5 Regulatory Compliances

For information on the performance and environmental test standards.

1.6 WEEE Compliance

DECLARATION OF CONFORMITY in relation to Directive 2002/96/EC, Waste Electrical and Electronic Equipment (WEEE).

The RVP900 manufactured by Vaisala complies fully with the requirements of Directive 2002/96/EC on the Waste Electrical and Electronic Equipment (WEEE).

1.6.1 Recycling

Vaisala has implemented return facilities for all products that we bring to market. All RVP900 components should be returned to the following address for recycling:

Vaisala Inc.
194 South Taylor Ave.
Louisville, CO 80027
(303) 499–1701

RVP900 components should not be disposed of in landfills.

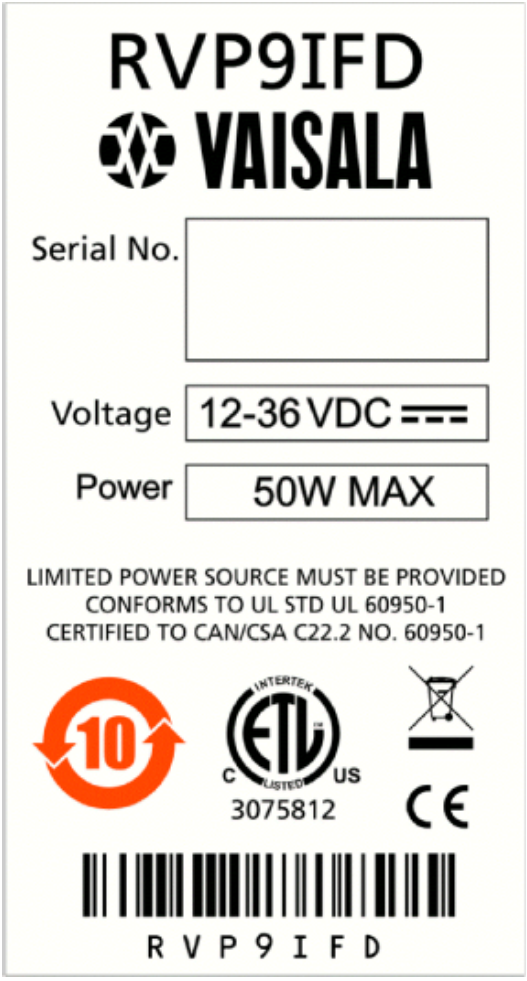
RVP900 components are marked with the end-of-life, not for landfill disposal symbol in accordance with European Standard EN 50419.

1.7 RoHS Compliance

The RoHS Directive 2002/95/EC restricts the use of six hazardous materials found in electrical and electronic products. All applicable products in the EU market after July 1, 2006 must pass RoHS compliance. The maximum permitted concentrations are 0.1% or 1000 ppm (except for cadmium, which is limited to 0.01% or 100 ppm) by weight of homogeneous material.

1.7.1 China RoHS Compliance

The China RoHS Directive requires disclosure (not removal) of the 6 EU RoHS substances for those products included in the "List". Disclosure can be at the component or at the sub assembly level, but it has to be in the prescribed format, in Chinese, as detailed in the document "Marking for the control of Pollution Caused by Electronic Information Products". There are product marking requirements and a calculation of the "Environmentally Friendly Use Period" to be calculated.



1.8 Trademarks

Vaisala and the Vaisala logo are registered trademarks of Vaisala Oyj in the United States and/or other countries.

All other company, product names, and brands used herein may be the trademarks or registered trademarks of their respective companies.

1.9 License Agreement

All rights to any software are held by Vaisala or third parties. The customer is allowed to use the software only to the extent that is provided by the applicable supply contract or Software License Agreement.

1.10 Warranty

Visit our Internet pages for more information and for our standard warranty terms and conditions: www.vaisala.com/warranty.

Any such warranty may not be valid in case of damage due to normal wear and tear, exceptional operating conditions, negligent handling or installation, or unauthorized modifications. See the applicable supply contract or Conditions of Sale for details of the warranty for each product.

1.10.1 Hardware Limited Warranty

Vaisala warrants its RVP900 Digital IF Receiver and Signal Processor 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.

CHAPTER 2

INTRODUCTION AND SPECIFICATIONS

2.1 RVP900 Lineage

The Vaisala product line has a three decade history of innovative, high-quality signal processing products. The history of Vaisala products is similar to the 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 second trip echo filtering.
1996	RVP7	>200	First commercial processor to implement fully digital IF processing for weather radar.
2003	RVP8	>400	First digital receiver/signal processor to be implemented using an open hardware and software architecture on standard PC hardware running a Linux operating system. Public APIs are provided so that customers may implement their own custom processing algorithms.

Year	Model	Units Sold	Major Technical Milestones
2009	RVP900		First IF digital receiver as a networked device to a signal processing PC running a Linux operating system. The PC bus-less architecture allows the fastest PCs to be used in signal processing role creating more real-time processing possibilities. The RVP900 contains all the functionality of the RVP8 PCI cards and IFD on a single printed circuit board.

Much of the proven, tested, and documented software from the highly-successful RVP8 (written in C) is ported directly to the new RVP900 architecture. This allows Vaisala to reduce time-to-market and produce a high-quality, reliable system. The RVP900 provides new capabilities for weather radar systems that, until now, were not available outside of the research community.

2.2 Dual Frequency Receive Options

For example, the RVP900 IF Digital Receiver (IFDR) performs 38.4 billion multiply accumulate cycles per second, which is a fivefold increase over the RVP8. Parallel Finite Impulse Response (FIR) filter processing blocks have been created to allow simultaneous dual frequency receive strategies to be deployed. With the advanced digitally synthesized transmitter function, this allows for new processing techniques still in the research realm for weather radar applications, such as alternating dual frequency staggered PRTs and pulse compression with off-frequency short pulses to fill in the near range data.

2.3 Open Hardware and Software Design

Compared to previous processors that were built around proprietary DSP chips and PCI card technology, the RVP900 is implemented around a single FPGA and acts as a networked device connected through a CAT5e ethernet to the latest in PC server technology. Eliminating the dependency of multiple PCI slots on the host computer allows continued access to latest improvement in processor speed, bus bandwidth, and the availability of low-cost compatible hardware and peripherals. The performance of an entry level RVP900 PC (currently dual quad-core 2.33 GHz Intel Xeon processors) is approximately five times faster than the fastest RVP8 ever produced (with dual 3.0 GHz Pentium processors).

The RVP900 IFDR produces digital I and Q data. The digital I and Q data is given to a PC server to perform the processing using pulse pairs, Fourier

transforms, or random phase techniques. Since the IFDR is a networked device, the digital I and Q data can be received by parallel signal processors in real-time. This allows a hardware topology that has many advantages that are yet to be explored.

Aside from the open hardware approach, the RVP900 has an open software approach; it runs in a Linux operating system. The code is structured, and public APIs are provided, so that research customers can modify or replace existing algorithms, or write their own software using the RVP900 software structure as a foundation to build on.

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 by purchasing standard, low-cost PC components from local sources.

2.4 Standard LAN Interconnection for Data Transfer or Parallel Processing

For communication with the outside world, the RVP900 supports a standard 10/100/1000 BaseT Ethernet. For most applications, the IRIS/Radar software is installed on the same PC. Moment results (Z, T, V, and W) are transferred internally; however, the 100 BaseT Ethernet is used to transfer moment results (Z, T, V, and W) to third-party applications host computer (for example, a product generator). The gigabit Ethernet is also sufficiently fast enough to allow UDP broadcast of the I and Q values for 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 RVP900.

2.5 System Configuration Concepts

The hardware building blocks of an RVP900 system are:

- **RVP901™ IF Digitizer Receiver (IFDR)**—A separate sealed unit from electrical interference and environmental conditions. It is usually mounted inside the receiver cabinet, but the new multi-functionality allows new opportunities of locating the device. The IFDR contains all the functionality of the RVP8 PCI cards and IFDR within the same footprint.

  The primary input to the IFDR is the received IF signal. The IFDR has five identical 16-bit A/D converters to sample the transmit pulse and up to four receiver channels. An external clock may be used to phase

lock the A/D conversion with the transmit pulse (not used for magnetron systems); however, the internal clock of the IFDR is so stable the unit can be used as the reference clock for the entire radar system.

IF transmit waveforms are synthesized by the IFDR and can be output over two BNC connectors. The IF transmit waveforms are programmable in phase, frequency, and amplitude. In the simplest case, it might supply the Coherent Local Oscillator (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.

The RVP901 IFDR can handle miscellaneous digital input and output, such as triggers, polarization switch controls, pulse width control, and more. Each of these I/O lines is a general purpose, uncommitted, static-protected signal that is directly controlled by the FPGA. Their specific functions is defined at the user-level in future software releases.

The IFDR is connected to the RVP902 Signal Processor by a CAT5e cable, which can be up to 25 m in length.

- **RVP902 Signal Processor**—A robust 1U rack-mounted PC chassis with dual quad-core Intel Xeon motherboard, two hot-swappable hard drives, DVD/RW, keyboard, mouse, and optional monitor for local diagnostic work. Redundant fans and a remote Intelligent Platform Management Interface (IPMI) are also included.
- **Expansion Panels**—A means for the signal processor to interface with other sub-systems of a radar. The RVP901 is designed with a high number of generic I/O capability to interface with these expansion panels. Currently, an expansion panel is available for the TDWR system. In the future, the RCP8 panel has the option to connect to the RVP901.

Compared to the previous generations architecture, this approach of consolidating all functionality into one printed circuit board eliminates four components. This increases reliability by reducing the number of hardware devices that could fail. It also decreases the life-time costs of operating a radar by lowering the cost of spares and maintenance. Typically, Vaisala supplies turn-key systems, although some OEM customers who produce many systems can purchase just the RVP901 component and integrate it themselves. This allows OEM customers to put their own custom "stamp" on the processor and even their own custom software.

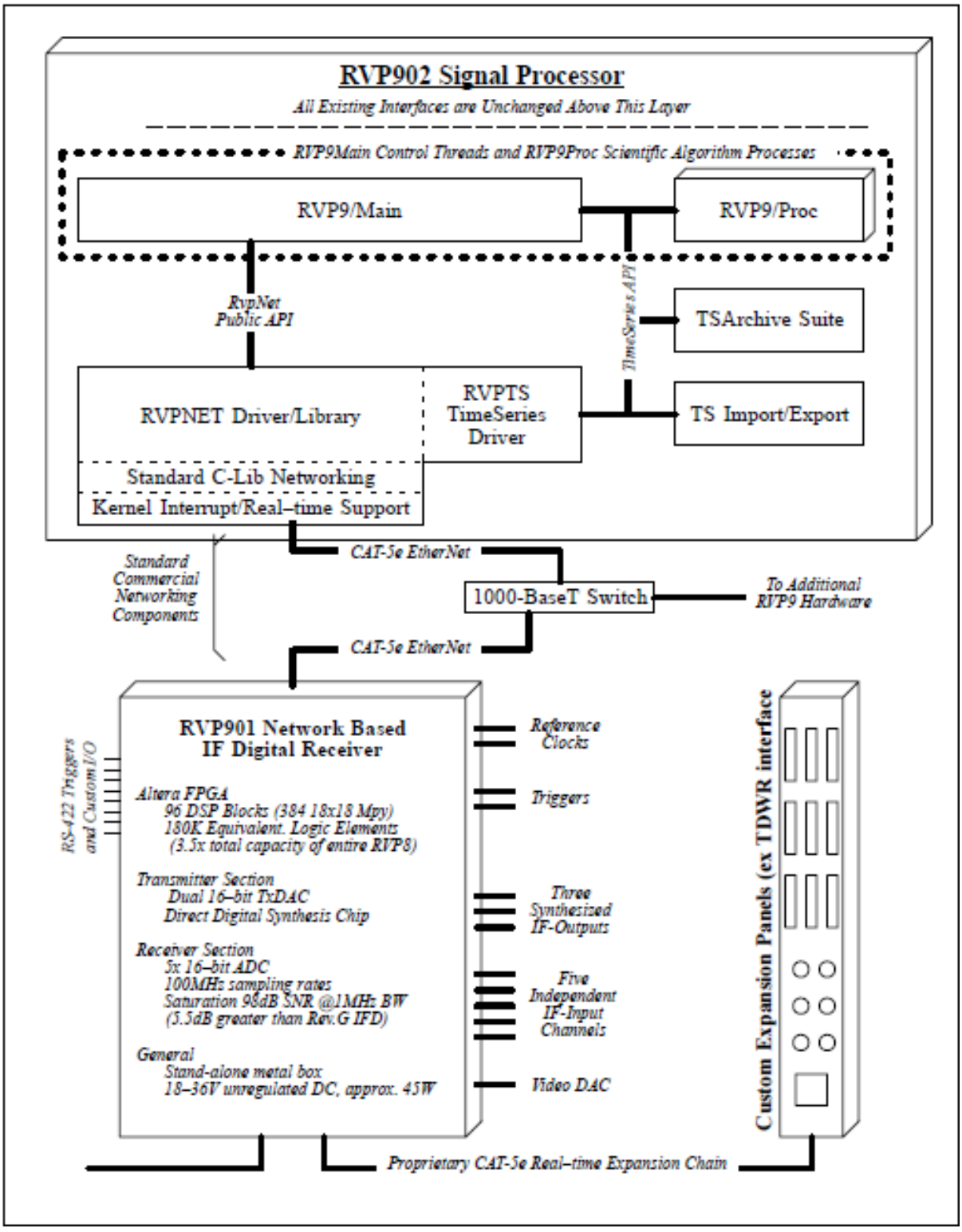


Figure 1 RVP900 System Concept

See on page 22 through on page 24 for examples of typical RVP900 configurations.

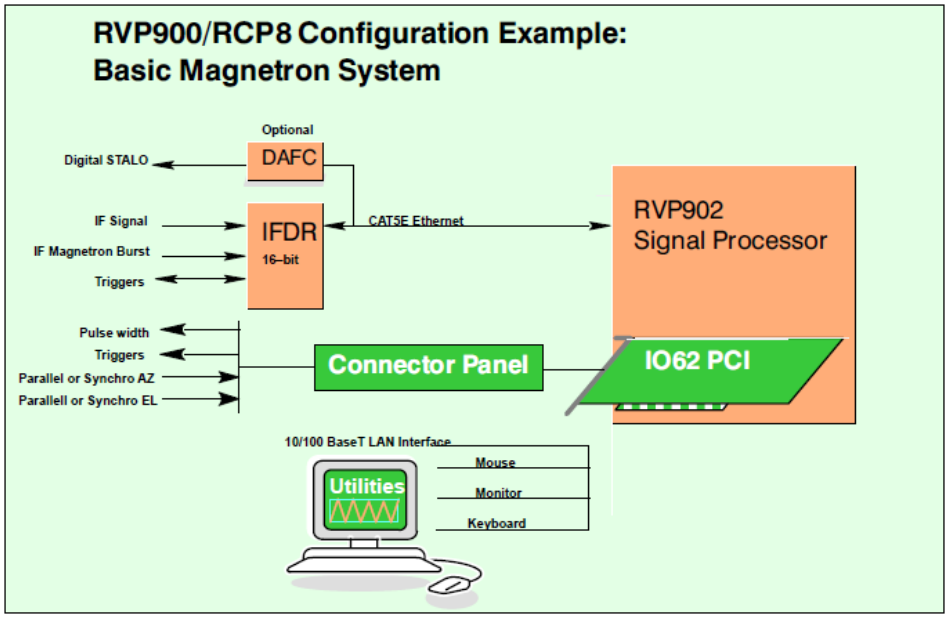


Figure 2 Example 1: Basic Magnetron System

The building blocks required to construct the basic system are:

- **RVP901 IFDR**—IDR installed in the radar receiver cabinet. This can be located up to 25 m from the RVP902 main chassis. The Digital Automatic Frequency Control (DAFC) is an option to interface to a digitally controlled STALO. The RVP900 provides full AFC control with burst pulse auto-tracking.
- **I/O-62 PCI Card**—This card is still available for additional triggers, parallel, synchro or encoder AZ and EL angle inputs, pulse width control, spot blanking control output, and more. These signals are brought in through the connector panel.
- **RVP902 Signal Processor**—1U, 19 in, rack-mounted computer with two quad-core Intel Xeon processors (PC) running a Linux operating system.

Figure 2 shows a basic magnetron system. The RCP8 I/O-62 PCI card continues to be used for generic input/output until the next generation of back panel is developed. This system has approximately five times the processing power of the fastest RVP8 ever produced (with dual 3.0 GHz Pentium processors), so that it is capable of performing DFT processing in 4200 range bins with advanced algorithms such as random phase second trip echo filtering and recovery.

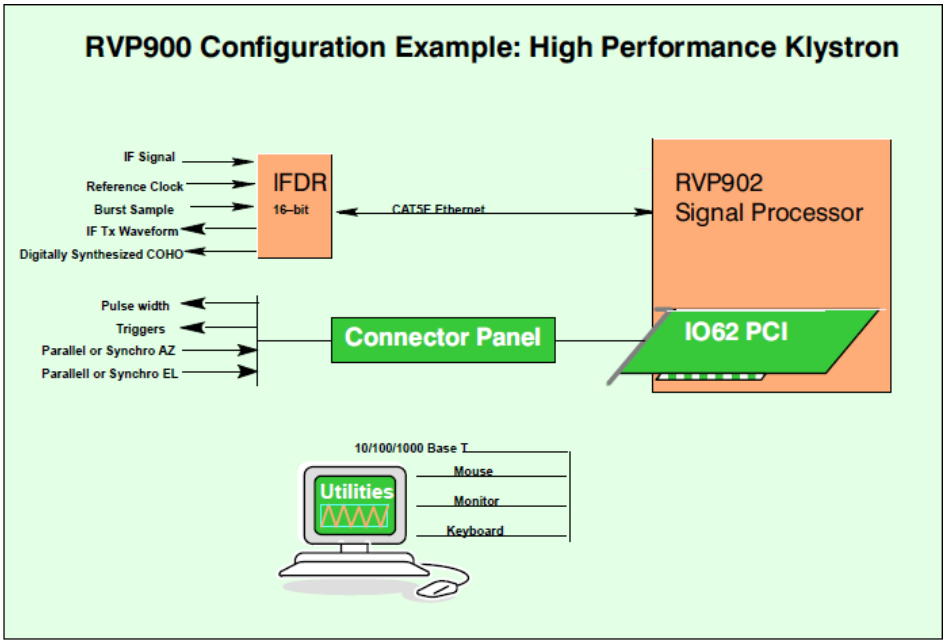


Figure 3 Example 2: Klystron System with Digital Tx

In Figure 3, the IFDR can receive a master clock from the radar system (for example, the COHO), or act as the reference clock. This ensures that the entire system is phase locked. The IFDR provides the digital Tx waveform. As compared to the example in Figure 2, no additional hardware is required. The Tx waveform generation functionality requires an optional software license installed.

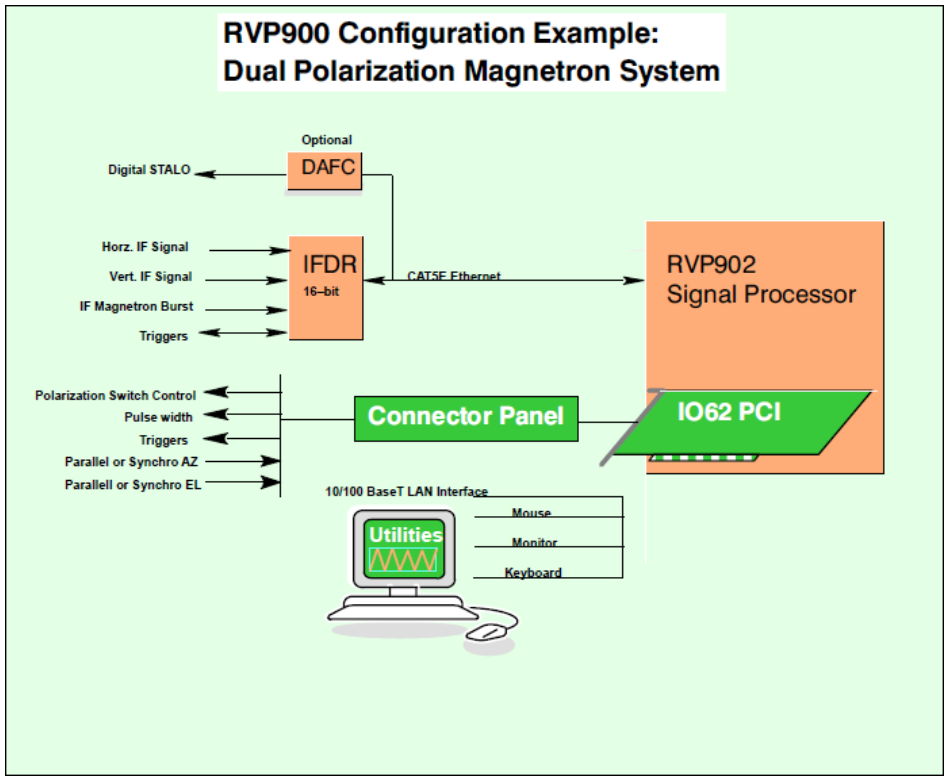


Figure 4 Example 3: Dual Polarization Magnetron System

In Figure 4, a two receive channel on the RVP901 IFDR is used. The receive signals, in both channels, are at the same IF frequency. Each channel has its own 16-bit ADC converter, which are phase locked to the master clock. As seen in the single polarization example in Figure 2, no additional signal processing hardware is required to convert to a dual-polarization solution. The functionality is an optionally licensed feature.

The RVP900 supports calculation of the complete covariance matrix for dual pol, including Z_{dr}, PhiDP (K_{dp}), RhoHV, LDR, and more. Which of these variables is available depends on whether the system is a single-channel switching system (alternate H and V), a (Simultaneous Transmit and Receive (STAR) system, or a dual-channel switching system (co- and cross-receivers).

2.6 RVP901 IFDR

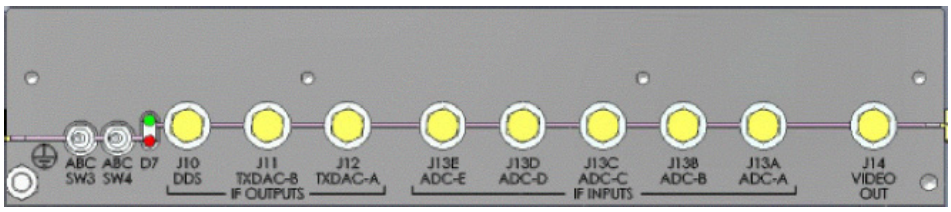


Figure 5 RVP901 IFDR IF Digital Receiver

The 16-bit IFDR is a sealed unit for optimum low-noise performance. The unit is carefully grounded and shielded to make the cleanest possible digital capture of the input IF signal. Because of this, the IFDR achieves the theoretical minimum noise level for the A/D converters.

The possible inputs to the IFDR are:

- IF video signals—There are four A/D converters used for received waveforms. Single polarization radars receive on ADC-A input. For dual-polarization radars, ADC-A is used for the primary polarization (usually H) and ADC-B for the secondary (usually V). The extra channels allow very wide dynamic range (WDR) applications for single and dual-polarization radars.
- The IF burst pulse sample for magnetron or IF COHO for Klystron is received over ADC-E.
- Optional reference clock for system synchronization. For a Klystron system, the COHO can be input. Magnetron systems do not require this signal.
- Trigger input or output is available on two 5 V 50Ω driver/receivers.

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 +8 dBm for the strongest expected input signal. A fixed attenuator or IF amplifier can be used to adjust the signal level to be in this range.

Digitizing is performed for both the IF signal and burst/COHO channels from a user-selectable range of 50 to 100 MHz at 16-bits resolution. This provides 92 dB 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. The five individual A/D converters are time synced within 1 nanoseconds. This ensures sampling in multiple channels is of the nearly equivalent targets.

All communication to the main RVP902 server chassis goes over a special CAT5e type cable. The major volume of data is the I and Q samples and some status indicators.

The RVP900 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 RVP900. A digital interface (±10 V) is supported.

2.6.1 Digital Receiver Function

The RVP901 receives the analog receive waveforms and digitizes IF samples. The advantage of this design is that the receiver electronics (LNA, RF mixer, IF preamp, and IFDR) can be located as far as 25 m away from the RVP902 server chassis. This makes it possible to choose optimum locations for both the IFDR and the RVP902, for example, the IFDR could be mounted on the antenna, and the processor box is in a nearby equipment room.

A remarkable amount of computing power is resident on the IFDR, in the form of an FPGA that can execute 38.4 billion multiply/accumulate cycles per second. This allows the use of multiple FIR filter arrays to run simultaneously. The FPGA serve as the first stage of processing of the raw IF data samples. Its job is to perform the down-conversion, band pass, and deconvolution steps that are required to produce (I,Q) time series. The time series data are then transferred over a Gigabit Ethernet connection to the RVP902 server for final processing.

The FIR filter array can buffer as much as 80 microseconds of 100 MHz IF samples, and then compute a pair of 2880-point dot products on those data every 0.83 microseconds. This could be used to produce over-sampled (I,Q) time series having a range resolution of 125 m and a bandwidth as narrow as 30 Khz. The same computation can also yield independent 125 m time series data from an 80 microseconds compressed pulse, whose transmit bandwidth was approximately 1 MHz.

Finer range resolutions are also possible, down to a minimum of 25 m. A special feature of the RVP900 is that the bin spacing of the (I,Q) data can be set to any desired value between 25 m and 2000 m. Range bins are placed accurately to within +2.2 m 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 m) 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 are interfaced to the same piece of hardware. Being the sampling time is highly coherent, ZDR biases do not

occur in high reflectivity gradients. The 16-bit I and Q resolution is passed to the RVP902 server for both H and V.

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.

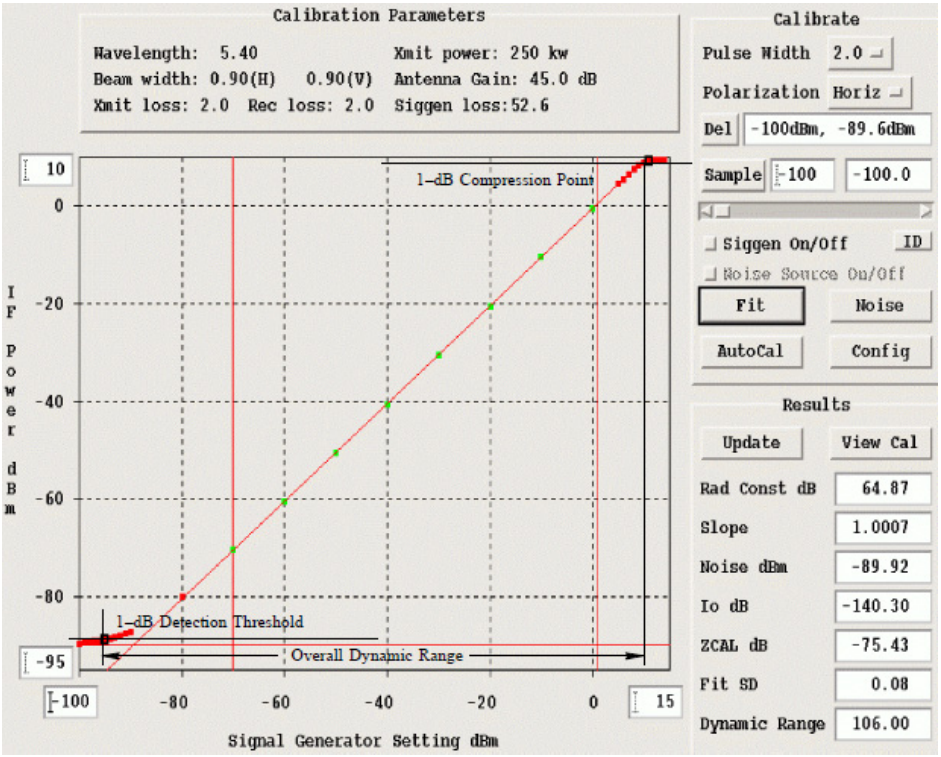


Figure 6 Calibration Plot for RVP901

Figure 6 shows a calibration plot for a 16-bit IFDR with the digital filter matched to a 2 microseconds pulse. The performance in this case is >105 dB dynamic range.

The RVP900 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 RVP900. 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 cancellation 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 linearizion algorithm used in the RVP900 provides an extra 3 dB to 4 dB of dynamic range by accounting for the effects of saturation.

The RVP900 provides the same comprehensive configuration and test utilities as in the RVP8. These utilities can be run either locally or remotely over the network.

2.6.1.1 Digital IF Band Pass Design Tool

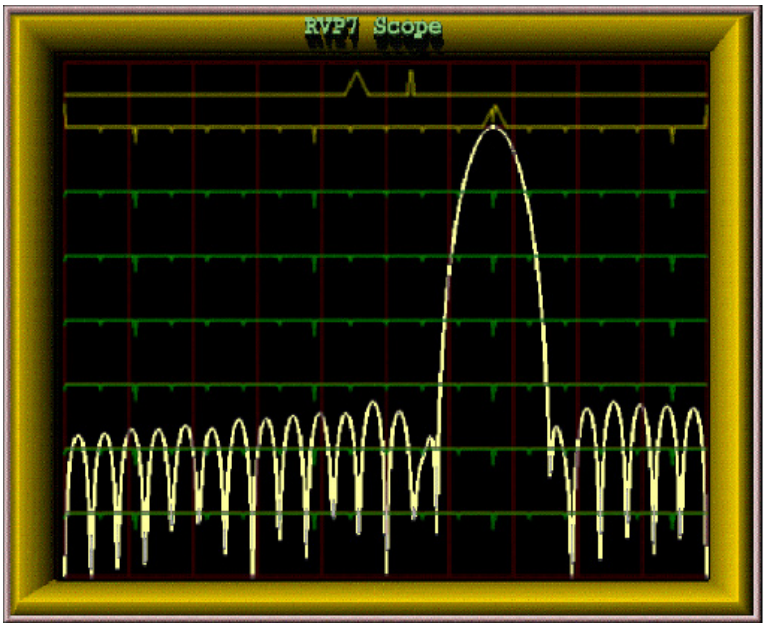


Figure 7 Digital IF Band Pass Design Tool

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.

2.6.1.2 Burst Pulse Alignment Tool

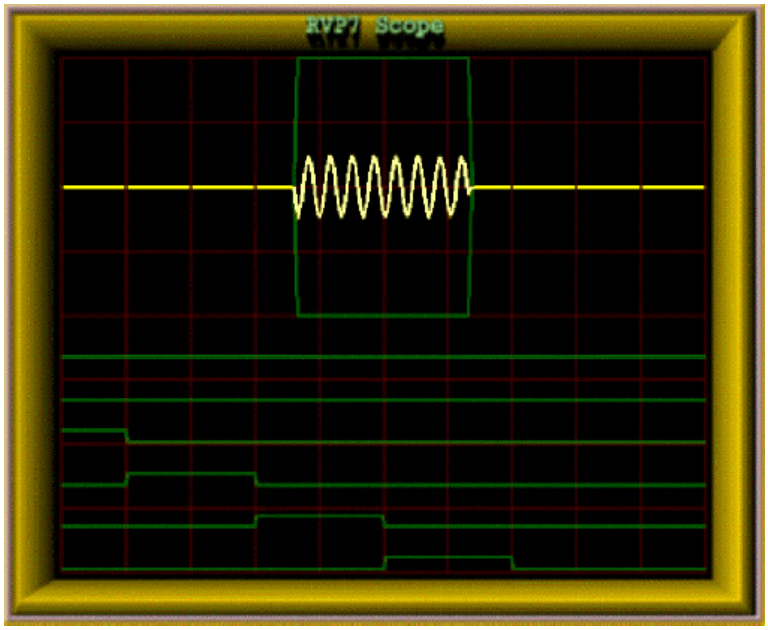


Figure 8 Burst Pulse Alignment Tool

The quality assessment of the transmit burst pulse and its precise alignment at range zero are easy to do, either manually using this tool or automatically using the burst pulse auto-track feature. This performs a 2D search in both time and frequency space if a valid burst 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.

2.6.1.3 Received Signal Spectrum Analysis Tool

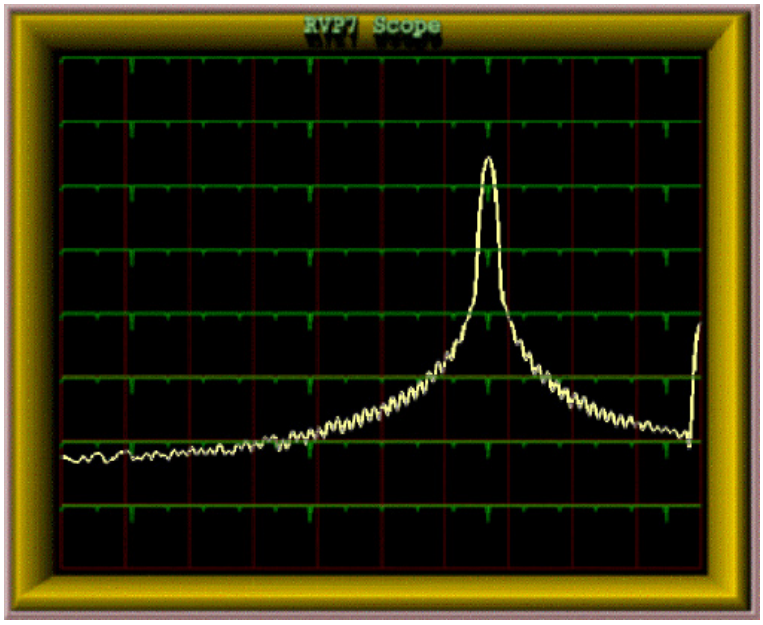


Figure 9 Received Signal Spectrum Analysis Tool

The RVP900 provides plots of the IF signal versus range as well as spectrum analysis of the signal as shown in Figure 9.

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 be done from a central maintenance facility through the network. For a multi-radar network, this results in substantial savings in equipment, time, and labor.

2.6.2 Digital Transmitter Function

Many of the exciting new meteorological applications for the RVP900 are made possible by its ability to function as a digital radar receiver and transmitter simultaneously. The RVP901 IFDR synthesizes an output waveform that is centered or offset from 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 RVP900 can even correct for waveform distortion by adaptively "per-distorting" the transmit waveform, based on the measured transmit burst sample.

The IFDR has two SMA output for the IF Tx waveform. The digital IF waveforms are generated by a 16-bit DAC with 65 dB SNR from 10 MHz

to 95 MHz. In addition, there is a third 12-bit A/D video channel for output for an auxiliary signal or clock up to 1.2 MHz.

The RVP900 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. The transmitter synthesizes an IF waveform for up-conversion to RF. The beauty of this approach is that the RVP900 has complete control over both halves of the radar, making possible a whole new realm of matched Tx/Rx processing algorithms. For example:

- **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 RVP900 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, which 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 illuminate its targets using longer pulses at lower power. The problem 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 RVP900 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 RVP900 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 agility can also be combined with pulse compression to remedy the blind spot, at close ranges, while the long pulse is being transmitted.
- **COHO synthesis**—The RVP900 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.

2.6.3 RVP902 Signal Processing Computer

The dual quad-core Intel Xeon-based computer acts as the host to the Linux operating system and provides all of the computing resources for processing the I/Q values that are generated by the RVP901 IFDR. Dual hot-swappable hard drives, using a RAID 1 configuration, help to ensure longevity of the processor, reducing system interrupts. Standard keyboard, mouse, and monitor connections are available, along with 10/100/1000 Base Ethernet port. The system does not require that a keyboard, mouse, or monitor be connected, which is typically the case at an unattended site.

Computers are available from many vendors, at various speeds. Typically the computer is equipped with 2 GB RAM. The RVP902 chassis has four drive bays. A DVD/RW is also provided for software maintenance.

NOTE	The latest versions of the RVP900 software and documentation can be downloaded for free from the Vaisala web site.

The RVP902 computer also plays host for RVP900 Utilities, which provide test, configuration, control, and monitoring software as well as built-in, online documentation.

2.7 Analog Versus Digital Radar Receivers

2.7.1 What is a Digital IF Receiver?

A digital IF receiver accepts the analog IF signal (typically 30 MHz or 60 MHz), processes it, and outputs a stream of WDR 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 WDR
- 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. They 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 about 1997, weather radars were not supplied with digital receivers. Today, 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 and RVP8, which are 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, >95 dB depending on pulse width), without having to use AGC circuits, which are complex to build, calibrate, and maintain. Other advantages include:

- Lower initial cost by eliminating virtually all IF receiver components
- Lower life-cycle cost due 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.

2.7.2 Magnetron Receiver Example

For a typical analog receiver of a magnetron system (see Figure 10 on page 35 (top portion)), the received RF signal from the LNA is first mixed with the STALO (RF-IF). The resulting IF signal is applied to one of several band pass 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 logarithmic amplifier (LOG), having a dynamic range of 80 dB to 100 dB, 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; therefore, a separate linear channel is still required.

The linear amplifier is fed from the other half of the band pass 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 IAGC attenuator must be calibrated so that the "I" and "Q" samples can 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. Each mixer must be supplied with an accurate 0 degree and 90 degree version of the COHO. The later is usually obtained by sampling a portion of the transmitted pulse, and then phase locking a COHO that continues to "ring" afterward. Phase locked COHOs 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 AFC loop. The AFC relies on a 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.

For the RVP900 digital receiver (see Figure 10 on page 35, bottom portion), the only old parts 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 IFDR on SMA coax cables. These cables constitute the complete interface to the radar's internal signals; no other connections are required within the receiver cabinet.

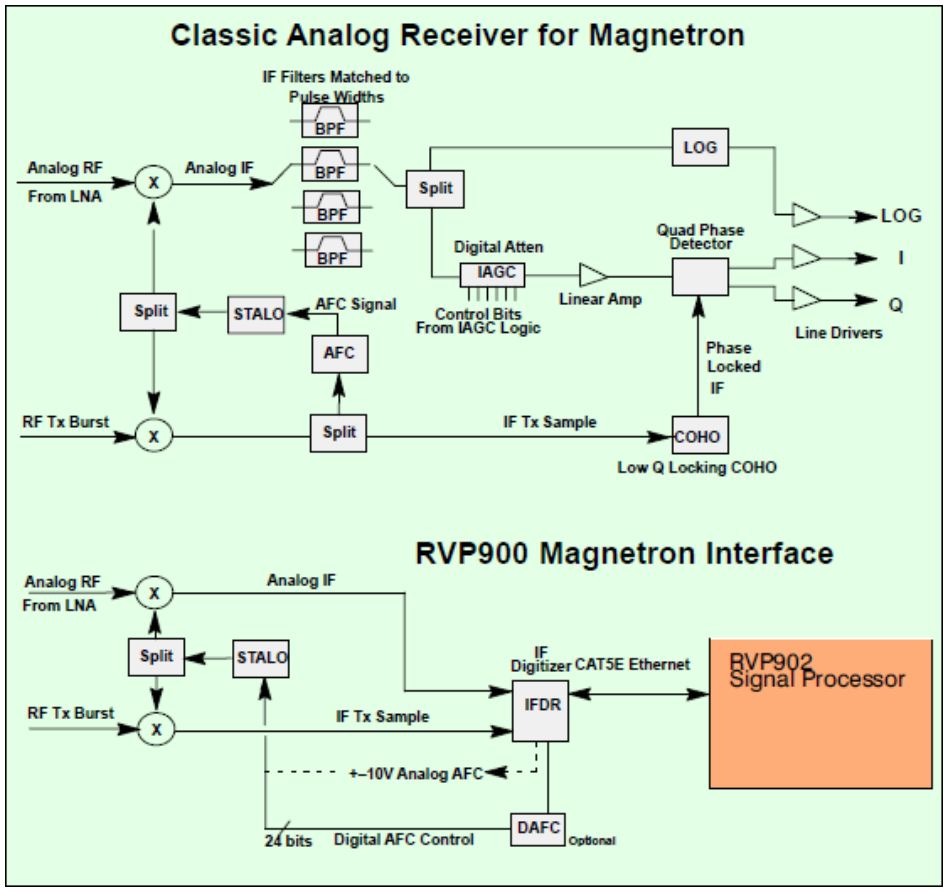


Figure 10 Analog vs Digital Receiver for Magnetron Systems

2.7.3 Klystron or TWT Receiver and Transmit RF Example

For a typical analog receiver of a klystron system (see Figure 11 on page 36 (top portion)), 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 second 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 RVP900 transmitter function 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.

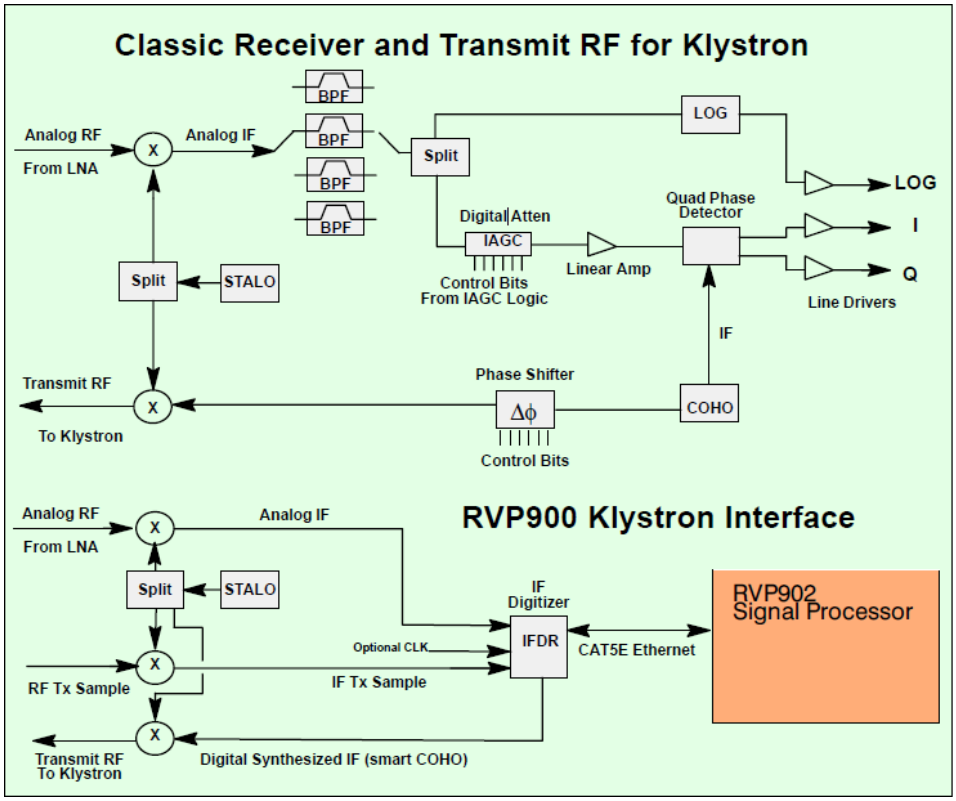


Figure 11 Analog vs Digital Receiver for Klystron Systems

2.8 RVP900 IF Signal Processing

2.8.1 IFDR Data Capture and Timing

The RVP900 design concept is to provide a next generation signal processor on a single board. The architecture is bus-less and processor independent. The design relies on common networking components as the interface for extension and communication.

The digitized IF and burst pulse samples are multiplexed onto the fiber channel link, which provides the digital data to the RVP902/main board at approximately 540 Mbps. The 14-bit samples are encoded for transmission over a fiber channel link. This optical link allows the IFDR to be as far as 100 m away from the RVP902/main board, and provides an added degree of noise immunity and isolation.

The uplink input from the RVP902/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 sampling clock in the IFDR is selected to be very stable. The sample clock serves a similar function to the COHO on a traditional Klystron system; it is the master time keeper. The IFDR sample clock is used to phase lock the entire RVP900; the Rx, Tx, miscellaneous I/O are all phase locked to the IFDR sample clock.

2.8.2 Burst Pulse Analysis for Amplitude/Frequency/Phase

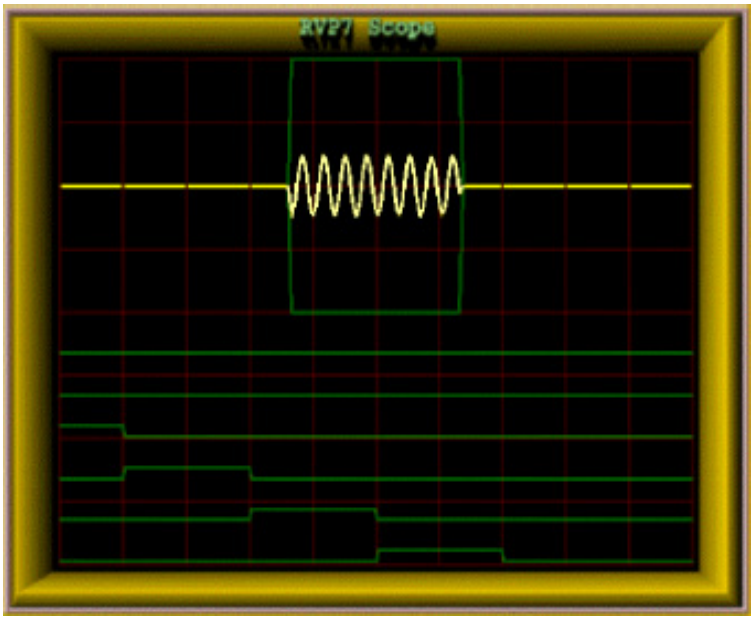


Figure 12 Burst Pulse Analysis for Amplitude/Frequency/Phase

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 de-aliasing, 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 pulse-to-pulse transmitted power. In addition, a warning is issued if the burst pulse amplitude falls below a threshold value.

The burst pulse data stream is first analyzed by an adaptive algorithm to locate the burst pulse power envelope (for example, 0.8 ?sec). The

algorithm does a coarse search for the burst pulse in the time/frequency domain (by scanning the AFC). It 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 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 through the uplink to the IFDR in the receiver cabinet.

2.8.3 RVP901 Functional Block Diagram and IF to I/Q Processing

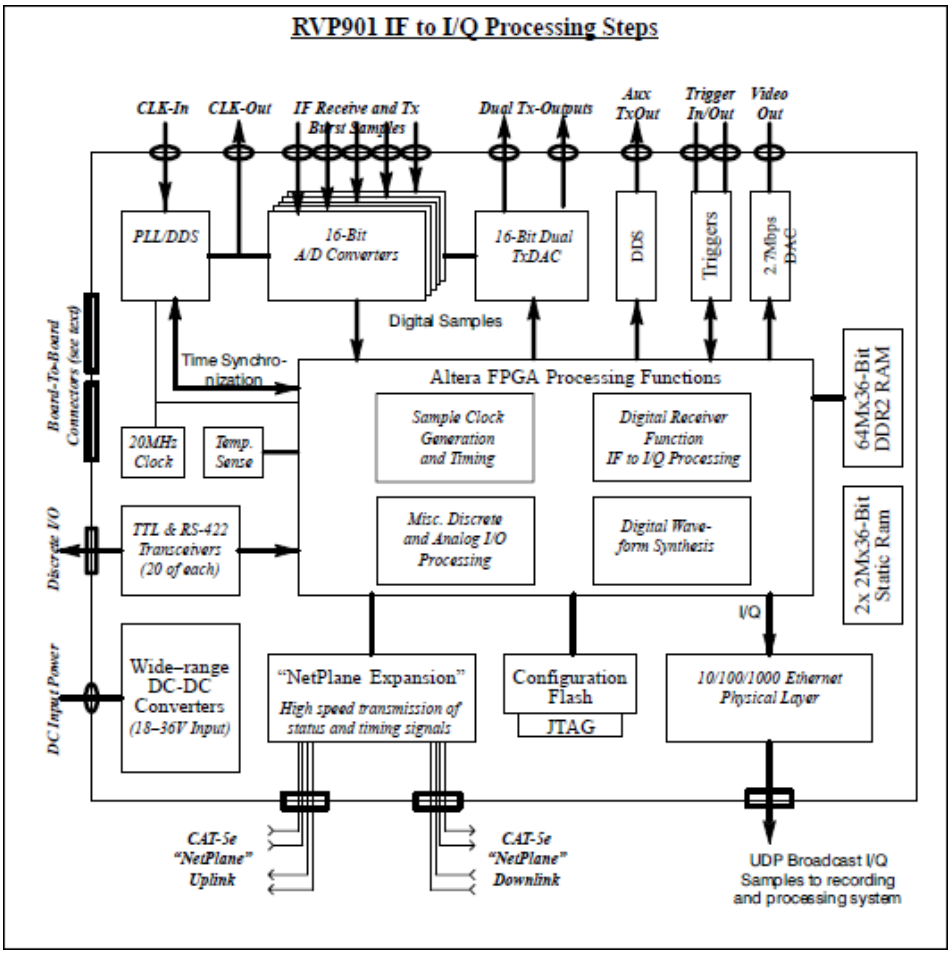


Figure 13 IF to I/Q Processing Steps

The RVP901 IFDR is the first commercial signal processor to perform parallel FIR filtering, simultaneously on each channel. This allows frequency agility receiving functions within the bandwidth of the analog receiver. This frequency agility on receive unifies the transmit frequency agility, introduced with the RVP900, allowing more advanced signal processing concepts to be introduced in commercial weather radars.

The RVP901 IFDR performs the initial processing of the IF digital data stream and outputs "I" and "Q" data values to the host computer through the CAT5e Ethernet. 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 the pulse width
- 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 IFDR for digital 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, pulse width, gate spacing, etc.) is supplied from the host computer.

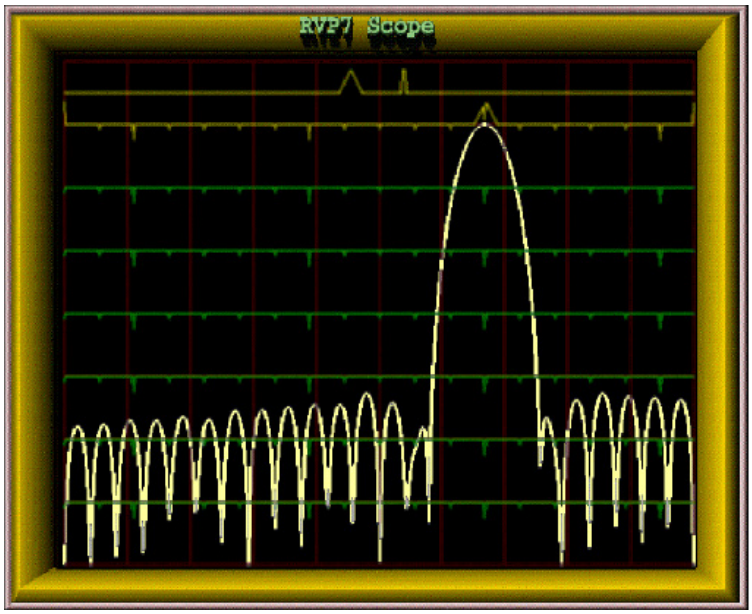


Figure 14 Digital IF Band Pass

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 RVP900 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 RVP900. Three different interference rejection algorithms are supported.

The RVP901 IFDR places the WDR “I” and “Q” samples directly on the Ethernet line, where they are sent to the processor section of the PC (for example, dual Intel Xeon processors on a motherboard). The I/Q values are then processed on the Intel Xeon processors to extract the moment information (Z, V, W, and optional polarization parameters).

2.9 RVP900 Weather Signal Processing

The processing of weather signals by the RVP900 is based on the algorithms used in RVP8 and RVP7. However, the performance of the RVP900 has a different approach to some of the processing algorithms, especially the frequency domain spectrum processing. All of the algorithms start with the WDR I and Q samples that are obtained from the IFDR over the Ethernet.

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

- Quantitative Rainfall Measurement
- Vertical Wind Profiling
- Z_{dr} 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 RVP900 has several major processing modes options:

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

The RVP900 performs discrete Fourier transforms (DFT) and fast Fourier transforms (FFT). FFT is more computationally efficient than DFT, but the sample size is limited to a power of two (16, 32, 64, etc.) This is too restrictive on the scan strategy for a modern Doppler radar since this means, for example, that a 1 degree azimuth radial must be constructed from exactly 64 input I/Q values. The RVP900 has the processing power such that when the sample size is not a power of two, a DFT is performed instead of an FFT.

2.9.1 General Processing Features

Figure 15 shows a block diagram of the processing steps.

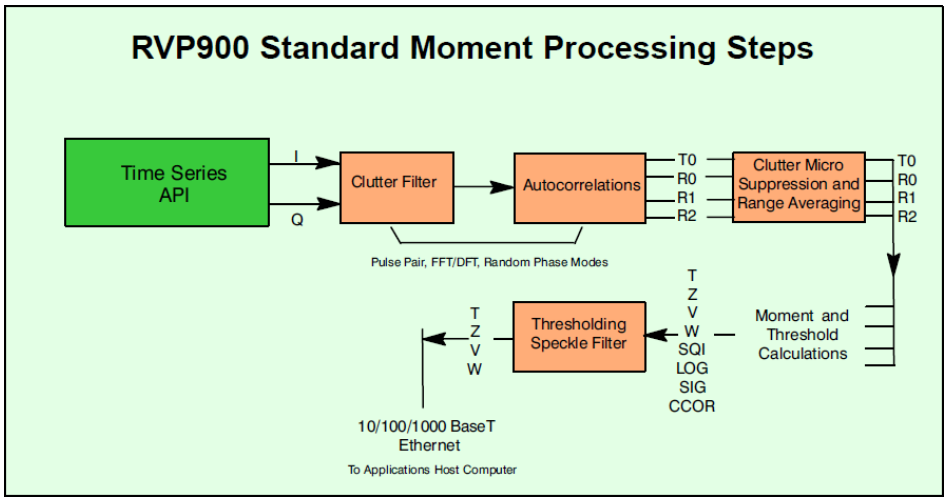


Figure 15 I/Q Processing for Weather Moment Extraction

The use of the R2 lag provides improved estimation of signal-to-noise ratio and spectrum width. Processors that do not use R2 cannot effectively measure the SNR and spectrum width.

2.9.1.1 Autocorrelations

The autocorrelations R0, R1, and R2 are produced by Pulse Pair, Random Phase, and DFT/FFT modes. However, the way that they are produced is different for the three modes, particularly with regard to the filtering that is performed:

- **Pulse Pair Mode**—Filtering for clutter removal may be performed in the time domain or frequency domain. Traditional IIR type clutter filters are available in the time domain. However, the frequency domain filter is much more adaptable. Clutter filtering can be optionally performed in the frequency domain, and then inverse FFT

or DFT may be performed to return to time domain after clutter removal. Autocorrelations are then computed in the time domain.

- **DFT/FFT Mode**—Filtering for clutter is performed in the frequency domain using both fixed width filters and the Gaussian Model Adaptive Processing (GMAP) technique. Autocorrelations are computed from the inverse transform.
- **Random Phase**—Filtering for clutter and second trip echo is performed in the frequency domain by adaptive algorithms. Autocorrelations are computed from the inverse transform.

2.9.1.2 Time (Azimuth) Averaging

The autocorrelations are based on input "I" and "Q" values over a selectable number of pulses between 8 and 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, for example, 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).

2.9.1.3 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 RVP900 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 can 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.

2.9.1.4 Range Averaging and Clutter Microsuppression

To improve the accuracy of the reflectivity measurements, the RVP900 can perform range averaging. When this is done, autocorrelations from consecutive range bins are averaged, and the result is treated as 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 bins to 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.

2.9.1.5 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 application.

2.9.1.6 Thresholding

The RVP900 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.

2.9.1.7 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 (three azimuth rays by three range bins) are supported.

2.9.1.8 Velocity Unfolding

A special feature of the RVP900 processor is the ability to "unfold" mean velocity measurements based on a dual PRF algorithm. In this technique, two different radar PRFs 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 RVP900 measures the PRF to determine which rate (high or low) was present on a given processing interval. It then unfolds based on either a 2:3, 3:4, or 4:5 frequency ratio. Table 1 on page 45 provides typical unambiguous velocity intervals for a variety of radar wavelengths and PRFs.

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	

2.9.2 RVP900 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 time domain notch filter, frequency domain fixed, or variable width filters. IIR filters of various selectable widths are available for either 40 dB 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 sent to the range averaging and moment extraction steps.

2.9.3 RVP900 DFT/FFT Processing

The DFT/FFT mode allows clutter cancellation to be performed in the frequency domain. DFT is used in general, with FFTs used if the requested sample size is a power of two.

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

2.9.4 Random Phase Processing for Second 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[3] separates the first and second trip echoes so that:

- In nearly all cases, the second trip echo can be removed from the first trip, even in the case of overlapped first and second trip echoes. The benefit is a clean first trip display.
- The second trip echoes can be recovered and placed at their proper range at first trip/second trip signal ratios of up to 40 dB difference for overlapped echoes. Because of the WDR of weather echoes, this power limit is sometimes 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. Magnetron radars have a naturally random phase. For Klystron radars, a digitally controlled precision IF phase shifter

is required. The RVP900 provides an 8-bit RS422 output for the phase shifter.

2.9.5 Polarization Mode Processing

Polarization processing uses a time domain autocorrelation approach to calculate the various parameters of the polarization co-variance matrix, that is, Z_{dr}, LDR, PhiDP, RhoHV, PhiDP (K_{dp}), etc. In addition, the standard moments T, V, Z, and 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, STAR, or dual channel switching. Vaisala is licensed by US National Severe Storms Laboratory (NSSL) to use the STAR hardware and processing techniques and algorithms.

Polarization measurements require special calibration of the Z_{dr} and LDR offsets. The use of a clutter filter for the polarization variables can sometimes bias the derived parameters. Because of this, the user decides whether or not to use filtered or unfiltered time series.

2.9.6 Output Data

The RVP900 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-bit or 16-bit format. An 8-bit format is preferred over a 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%. A 16-bit format is sometimes used by research customers for data archive purposes. Time series and DFT are always 16-bit formats. All data formats are documented in Chapter 7, Host Computer Commands, on page 255.

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

2.10 RVP900 Control and Maintenance Features

2.10.1 Radar Control Functions

The RVP900 also performs several important radar control functions:

- Trigger generation—Up to eight programmable triggers
- Pulse width control (four states controlled by four bits)
- Angle/data synchronization—To collect data at precise azimuth intervals (for example, every 0.5 degree, 1 degree, 1.5 degrees) based on the AZ/EL angle inputs
- Phase shifter—To control the phase on legacy Klystron systems. New or upgraded Klystron or TWT systems can use the RVP900/Tx card to provide very accurate phase shifting
- Z_{dr} switch control—For horizontal/vertical or other polarization switching scheme
- AFC output (digital) based on the burst pulse analysis for magnetron systems

Pulse width and trigger control are both built into the RVP900. Four TTL output lines can be programmed to drive external relays that control the transmitter pulse width. The internal trigger generator drives eight 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. Precisely delayed and jitter-free strobes and gates can be easily produced. For each pulse width, there is a corresponding maximum trigger rate that can be generated; however, the RVP900 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 RVP900 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:

- 2D AZ/EL sector blanking areas can be defined in the RVP900
- 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

2.10.2 Power-Up Setup Configuration

The RVP902 stores on disk an extensive set of configuration information. The purpose is to define the exact configuration on startup. The setup information can be accessed and modified using either the 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. The

RVP902 has the capability to flash the RVP901 software. This allows upgrading to new versions, as needed, by the user.

During the boot process, the RVP901 is first loaded with a factory configurable diagnostic image. The diagnostic image initializes devices that need initialization, and leaves the board in a reset condition. It verifies that the RDA software and setup configuration is present and uncorrupted. Once validated, the diagnostic image boots the RDA application. The application image runs some basic memory self-tests before starting.

The RDA version can be updated in the field with minimal risk. The RVP902 software provides a configuration interface. If, for some reason, the upgrade was interrupted or not completed successfully resulting in an invalid image, the unit stays in diagnostic mode on the next reboot to allow the user to recover from the failure. If the RDA software boots, but is unresponsive, the user may force the unit into diagnostic mode by sticking a paper clip in the inner most hole (SW1) in the finned side of the enclosure. After entering the diagnostic boot mode, the user may re-flash the RVP901 software from the signal processor to recover from a corrupted image.

The platform provides monitoring of voltage and temperature conditions. It also monitors the locked conditions of the various PLLs on the FPGA to verify that the analog and digital clocks on board are in good health.

The RVP902 IP address is user-configurable. The outer most hole (SW2) in the finned side of the enclosure allows the user to reset the IP address to 10.0.1.254, in the event the user can not remember what was programmed. For more information about flashing the software and setting the IP address, see Chapter 3, Hardware Installation, on page 61.

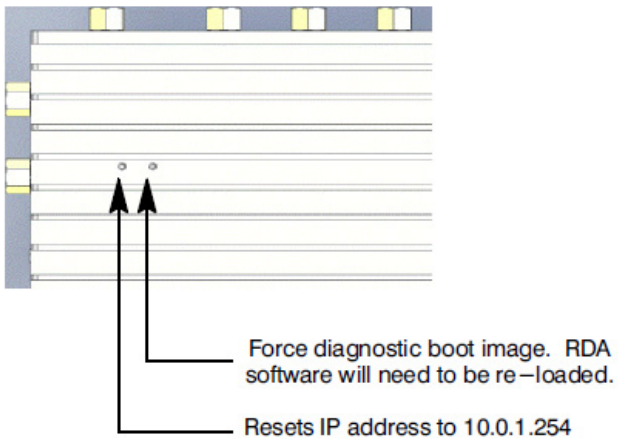


Figure 16 Reset IP Address

2.11 Support Utilities and Application Software

The RVP900 system includes a complete set of tools for the calibration, alignment, and configuration of the RVP900. 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 also included.
- **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 RVP900 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 RVP900 configuration files
- **zauto**—Calibration utility for use with a test signal generator

These tools can be run locally on the RVP900 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 RVP900. The standard X-Window export is supported, but requires more bandwidth.

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

- **IRIS/Radar**—Runs on the same or separate PC. It interfaces to the RVP900 internally or by 100 BaseT Ethernet. IRIS/Radar controls both the RVP900 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 (Windows or Linux) running Netscape or Internet Explorer.
- **IRIS/Display**—Displays 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. Both IRIS/Analysis and IRIS/Radar have all of the capabilities of IRIS/Display in addition to their own functions; any IRIS system can display products.

2.12 System Network Architecture

The RVP900 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.

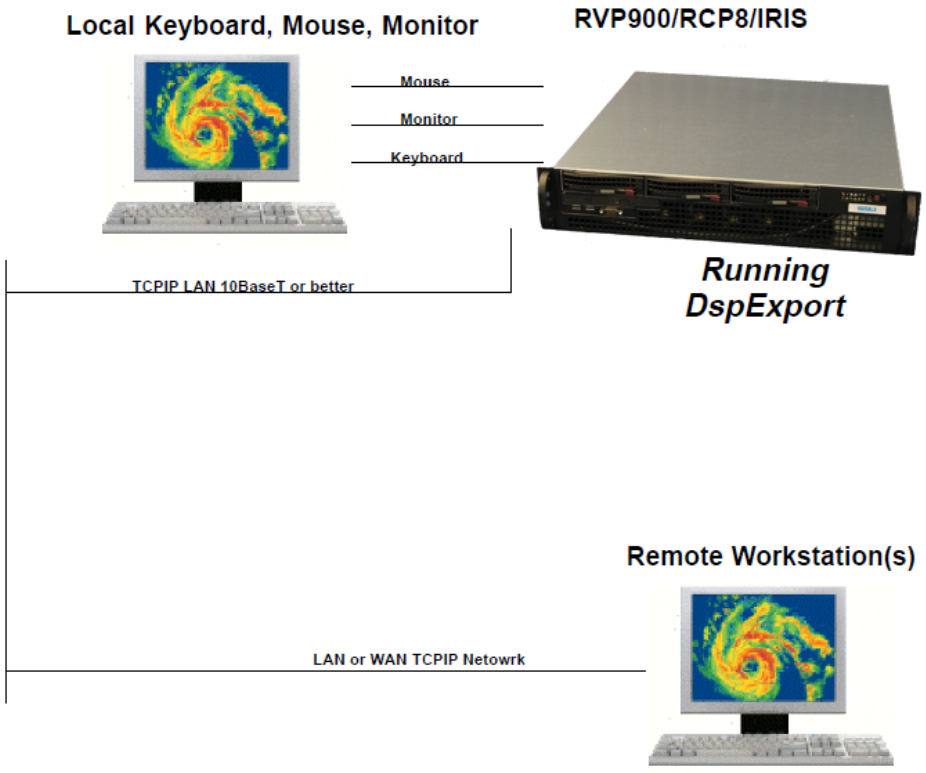


Figure 17 Network Architecture

The "dsp lib" runs locally on the RVP900. A utility, called DspExport, exports the library over the network using a TCP/IP socket. Typically the controlling application is on the same computer, but DspExport may be exported to a remote host radar control workstation (RCW) on the network. If this workstation is running the IRIS software, at least a 10BaseT connection is recommended.

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

2.13 Open Architecture and Published API

Vaisala recognizes that certain users may want to write their own signal processing algorithms, which run on the RVP900. The RVP900 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 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 the modification of 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 occurs for user-supplied code. The API allows the redefining of how the standard parameters (dBZ, Velocity, Width, PHIDP, etc.) are computed from the incoming (I,Q) time series. Brand new parameter types, which are not included in the basic RVP900 data set, may also be created.

The standard scientific algorithms are not made public in this model. The interface hooks and development tools are provided, so that users can add their own software extensions to the RVP900 framework. Many of the library routines that are fundamental to the RVP900 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 use the signal processing software supplied with the RVP900, the new open software architecture approach is 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.

2.14 RVP901 Technical Specifications

2.14.1 RVP901 IF Receiver Functions

Input Signals:

- IF Received Signal: 50Ω ,+ 8.0 dBm full-scale, +20dBm absolute max
- IF Burst or COHO: 50Ω, +8.0 dBm full-scale, +20dBm absolute max
- Optional Reference Clock: 7.5 MHz to 100 MHz, -20dBm to 6 dBm

IF Ranges:

- 5 MHz to 120 MHz

Linear Dynamic Range

- 85 dB to >105 dB, depending on pulse width/bandwidth filter

A/D Conversion:

- Resolution: 16 bit with jitter <1.0 picosec
- Sampling rate: 50 MHz to 100 MHz (software selectable)

AFC Output:

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

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 GUI.

Impulse Response:

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

Range Resolution:

Minimum bin spacing of 25 m, selectable in N*8.33 m 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 m.

Maximum Range:

Up to 1024 km

Number of Range Bins:

Full unambiguous range at minimum resolution or 4200 range bins (whichever is less)

RVP901 to RVP902 Link:

Uses shielded CAT5e cable, jumbo 8192-byte packets

Data Output via PCI Bus:

16-bit floating I and Q values

14-bit raw IF samples

2.14.2 RVP901 Digital Waveform Synthesis

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

TxDAC Analog Output Waveform Characteristics:

- Two independent, digitally synthesized, analog output waveforms (SMA). 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 +13 dBm into 50Ω
- 16-bit interpolating TxDAC provides >65 dB Signal-to-Noise Ratio
- IF center frequency selectable from 5 MHz to 65 MHz
- Signal bandwidth as large as 15 MHz for wideband/multiband Tx applications; bandwidth is adjustable in software

- Continuous or pulse modulated output with bandwidth limiting on pulse modulation output
- Precise phase shifting with transient bandwidth limiting
- Total harmonic distortion less than -74 dB
- Waveform pre-emphasis compensates for both static and dynamic Tx nonlinearities

DDS Analog Output Waveform Characteristics:

- Direct Digital Synthesis of analog waveforms that has simpler modulation requirements than are possible with TxDACs
- Can drive up to +13 dBm into 50Ω
- Outputs frequencies from 5 MHz to 105 MHz

2.14.3 Miscellaneous Discrete and Analog I/O

- There are two identical 51-pin MicroD connector to support miscellaneous I/O
- Includes D/A, A/D, discrete inputs and outputs (TTL, RS-485/422, etc.). See Table 2. RVP901 51-pin Micro-D Summary on page 57.
- I/O pin assignment mapping
- ESD protection using low capacitance TVS diode. Series resistors are added to the TTL pins to provide over voltage protection.

TTL I/O:

- 20 dedicated 5 V TTL lines
- Drive capability of ± 32 mA
- Over voltage, ESD, ETF, and lightening protection

RS-485/422 IO:

- 20 pairs with optional 120Ω termination
- Maximum data rate of 20 Mbps
- Over voltage, ESD, ETF, and lightening protection

Analog Inputs:

- Six analog differential inputs
- ± 10 V DC or low frequency signals
- Settle to within 0.1% of full scale value in 800 nanoseconds
- ADC conversion rate once every 0.5 μsec

Analog Outputs:

- +5 V 400 mA total combined maximum current
- -5 V 150 mA combined maximum current

Table 2 RVP901 51-pin Micro-D Summary

Pin #	Signal Type	Signal Name on J6	Signal Name on J3
1, 19	RS-485/422	GPDIFF_PIN_LP/N[10]	
2, 20	RS-485/422	GPDIFF_PIN_LP/N[11]	
3, 21	RS-485/422	GPDIFF_PIN_LP/N[12]	
4, 22	RS-485/422	GPDIFF_PIN_LP/N[13]	
5, 23	RS-485/422	GPDIFF_PIN_LP/N[14]	
6, 24	RS--485/422	GPDIFF_PIN_LP/N[15]	
7, 25	RS-485/422	GPDIFF_PIN_LP/N[16]	
8, 26	RS-485/422	GPDIFF_PIN_LP/N[17]	
9, 27	RS-485/422	GPDIFF_PIN_LP/N[18]	
10, 28	RS-485/422	GPDIFF_PIN_LP/N[19]	
11	TTL	TTLIO_PIN[10]	TTLIO_PIN[0]
12	TTL	TTLIO_PIN[11]	TTLIO_PIN[1]
13	TTL	TTLIO_PIN[12]	TTLIO_PIN[2]
14	TTL	TTLIO_PIN[13]	TTLIO_PIN[3]
15	TTL	TTLIO_PIN[14]	TTLIO_PIN[4]
16	TTL	TTLIO_PIN[15]	TTLIO_PIN[5]
17	TTL	TTLIO_PIN[16]	TTLIO_PIN[6]
36	TTL	TTLIO_PIN[17]	TTLIO_PIN[7]
38	TTL	TTLIO_PIN[18]	TTLIO_PIN[8]
40	TTL	TTLIO_PIN[19]	TTLIO_PIN[9]
42, 43	Analog Mux ± 10V	AMUX_POS_PIN/ AMUX_NEG_PIN[3]	AMUX_POS_PIN/ AMUX_NEG_PIN[0]
46, 47	Analog Mux ± 10V	AMUX_POS_PIN/ AMUX_NEG_PIN[4]	AMUX_POS_PIN/ AMUX_NEG_PIN[1]
50, 51	Analog Mux ± 10V	AMUX_POS_PIN/ AMUX_NEG_PIN[5]	AMUX_POS_PIN/ AMUX_NEG_PIN[2]
44	5V supply out	V_5P0_GPIO	V_5P0_GPIO
48	-5V supply out	V_N5P0_GPIO	V_N5P0_GPIO

Table 2 RVP901 51-pin Micro-D Summary

Pin #	Signal Type	Signal Name on J6	Signal Name on J3
18, 29, 30, 31, 32, 33, 34, 35, 37, 39, 41, 45, 49	GND	GND	GND

2.14.4 RVP900 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 dB to 5 dB)

Primary Processing Modes:

- Poly-Pulse Pair (PPP)
- DFT
- Random or phase coded second trip echo filtering/recovery
- Optional polarization with full co-variance matrix (Z_{dr}, PHIDP, LDR, RHOHV, etc.)
- Optional pulse compression

Processing Options:

- IIR Clutter filters (40 dB and 50 dB) or in pulse pair mode
- Fixed, adaptive width, and GMAP clutter filters in DFT and phase coded second 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) or 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 four pulse widths

Data Outputs:

- dBZ—Calibrated equivalent radar reflectivity, 8 bits or 16 bits
- V—Mean radial velocity, 8 bits or 16 bits
- W—Spectrum width, 8 bits 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: Z_{dr}, PHIDP, RHOHV, LDR, RHO, 8 bits 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—Removes single-bin targets such as aircraft or noise; fills isolated missing pixels as well

2.14.5 RVP900 Input/Output Summary

Ethernet Input/Output from Host Computer:

Data output of calibrated dBZ, V, and W during normal operation. Full I/Q time series 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 through the Ethernet interface.

AZ/EL Angle Input Options:

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

Trigger Output:

Up to 12 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

2.14.6 Physical and Environmental Characteristics

Packaging:

- Motherboard configuration: 1U rack mount
- Custom PC configurations available or packaged by customer
- Dimensions of standard 1U chassis:

  48.3 cm (W) x 50.3 cm (L) x 4.3 cm (H)
  19 in (W) x 19.8 in (L) x 1.7 in (H)
- Dimensions IF Digitizer:

  16.9 cm (W) x 24.3 cm (L) x 8.2 cm (H)
  6.6 in (W) x 9.6 in (L) x 3.2 in (H)

Input Power:

- RVP901: 100 VAC to 240 VAC, 47 Hz to 63 Hz auto-ranging or 18 VDC to 36 VDC
- Main Chassis: 100 VAC to 240 VAC, 50 Hz to 60 Hz auto-ranging

Power Consumption:

- RVP902/Main Processor: 1300 W
- RVP901 IFDR IF Digitizer: 45 W

Environmental:

- RVP901 with DC Option: -40°C to 50°C (-40°F to 122°F), 0% to 95% R.H. (non-condensing) with a minimum of 20 cubic feet per minute of air-flow.
- RVP901 with AC Option: -40°C to 45°C (-40°F to 122°F), 0% to 95% R.H. (non-condensing) with a minimum of 20 cubic feet per minute of air-flow.
- RVP902: 10°C to 35°C (50°F to 95°F), 8% to 90% R.H. (non-condensing) at altitudes less than 2000 meters.

Reliability:

MTBF>50,000 hours (at 25°C/77°F), <1 hour MTTR

CHAPTER 3

HARDWARE INSTALLATION

3.1 Overview and Input Power Requirements

This chapter describes how to install the RVP900 hardware. This includes mechanical installation and siting, electrical specifications of the interface signals, system-level considerations, and the standard connector panel.

There are two major modules supplied with the RVP900:

- RVP901 IF Digital Receiver; typically mounts in the radar receiver cabinet.
- RVP902 Main Chassis; usually mounted in 19 in EIA rack

Much of the RVP900 I/O is configured through software. This makes the unit very flexible. Since there is no custom wiring, internal jumpers, or oscillators, it is easy to insert spare modules. The software configuration of the I/O is described in Appendix A, Serial Status Formats, on page 337. The physical installation of the hardware is described in Appendix B, RVP900 Packaging, on page 343

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

3.2 RVP901 IFDR Installation

3.2.1 RVP901 IFDR Safety Precautions

WARNING	If IFDR is equipped with AC power option and installed in a 220V application, a fuse must be included in the neutral input path of the end product for compliance with IEC 60601-1:2005.

WARNING	IFDR must be installed in a suitable fire enclosure to meet EN60950-1 requirements.

WARNING	The IFDR mains power must 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 IFDR case mounting screw, and then brought to the power supply ground connection.

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

WARNING	Never open the IFDR for field-level services. Thermal conductive material does not bind properly when unit is reassembled. **Opening RVP901 voids all warranties.**

3.2.2 IFDR Introduction

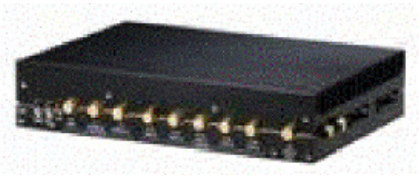


Figure 18 RVP901 IFDR

The IFDR 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. 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. Redundant fans are attached to supply airflow across the heat sink. The fans power is supplied from the AC to DC power supply or directly from the DC source. The RVP901 can be ordered without fans. In this case, the user needs to provide greater than 20 cubic feet per minute of airflow directly across the RVP901 heat sink. See Appendix B, RVP900 Packaging, on page 343 for a drawing of the power harness needed to supply power to both the RVP901 and fans.

The IFDR replaces all of the IF receiver components that are found in a traditional analog receiver system:

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

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 are other receiver modifications required to match the IFDR 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 may want to consider purchasing 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.

3.2.3 IFDR Power, Size, and Mounting Considerations

The IFDR is designed so that mounting brackets uses the same dimensions as previous RVP7 and RVP8 generation IFDs. It is a compact sealed module with dimensions 26.8 cm x 17.6 cm x 4.8 cm (10.5 in x 6.9 in x 1.9 in). The unit is designed to be mounted on edge, so that the 26.8 cm x 4.8 cm surface is flush with the back of the receiver cabinet, with a 17.6 cm protrusion into the cabinet. The unit is typically placed where a traditional LOG receiver, or previous generation RVP IFD, would be installed.

The IFDR is cooled by direct conduction through its metal enclosure. One side of the IFDR serves as a heat sink. The hotter chips mounted on the printed circuit board are bonded to the heat sink. The IFDR should be positioned, so that a minimum of 20 cubic feet per minute of airflow can freely convect around it. The ambient air temperature should be within a range of -40C to +50C (-40F to 122F) with DC supply power option and -40C to +45C (-40F to 113F) with AC supply option.

The power module is separate unit. The IFDR is delivered with the power supply module attached to the IFDR mechanical housing. The power supply module can be removed and mounted nearby in the radar cabinet. The power supply, fans, and bracket add 8.4 cm (3.3 in) of overall width to the receiver module. Power modules are available for 100 VAC to 240 VAC at 47 Hz to 63 Hz or 18 VDC to 36 VDC. The AC power supply is a low noise, low ripple, auto-switching unit. The DC input voltage may be unregulated. If DC voltage is selected as an option; three different fan assemblies are available. These fan options allow DC voltages from 12 VDC to 18 and 18 VDC to 28 VDC.

The IFDR has an internal regulator to supply the various digital circuits the voltages need. The internal regulated power modules are sized to provide several Watts more than is required by the RVP901. A ferrite choke around the supply wires, near the terminal strip, is also recommended.

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

The IFDR communicates with a host computer through Ethernet. The 10/100/1000 Mbit physical interface supports full and half duplex configurations. It has an Auto-MDIX feature, which eliminates the need to worry about cross-over versus straight cables. The platform provides support for both TCP and UDP packets. The default IP address, shipped with each system, is 10.0.1.254. The IFDR supports jumbo packets.

NOTE Vaisala recommends the UDP packet sizes be set to 8192 on the host computer.

Vaisala recommends using a shielded CAT5e cable (certified to ? 350 MHz), having shielded RJ45 plugs on each end. Ensure the Ethernet connectors on the host computer and IFDR make contact with the metal on the shielded cable, which provides DC return path. This design prevents ground loop currents from flowing between units, even when they are plugged into different AC/Mains.

3.2.4 IFDR I/O Summary

The connectors on the IFDR are labeled and described in Table 3. IFDR Connectors on page 65.

Table 3 IFDR Connectors

RVP901 Connector Panel Summary			
J-ID	**Label**	**Type**	**Description**
J1	DC IN	MINIFIT 2X2	Power Supply Input
J2	ETHERNET	RJ-45	Ethernet connection to host computer
J3	MISC I/O-B	51-Pin Micro-D	Each Micro-D has an identical pinout
J4	UP LINK	RJ-45	Gigabit serial Link
J5	DN LINK	RJ-45	Same as J4
J6	MISC I/O-A	51-Pin Micro-D	Same as J3
J7	CLK IN	SMA	Reference Clock Input
J8	TRIG-B	SMA	General purpose trigger I/O
J9	CLK OUT	SMA	Reference Clock Output
J10	DDS	SMA	Direct Digital Synthesizer Output
J11	TxDAC-B	SMA	Direct Transmit IF output
J12	TxDAC-A	SMA	Same as J11
J13A	ADC-A	SMA	Direct IF Input. IF-1 for single channel or primary polarization
J13B	ADC-B	SMA	Direct IF Input. IF-2 for secondary polarization
J13C	ADC-C	SMA	Direct IF Input
J13D	ADC-D	SMA	Direct IF Input
J13E	ADC-E	SMA	Direct IF Input. Burst Sample Input
J14	VIDEO OUT	SMA	Video DAC output, synthesizes simple video waveforms
J15	TRIG-A	SMA	General purpose trigger I/O or DAFC interface
SW1		Switch	Tactile, momentary on switch. Restores factory boot image
SW2		Switch	Tactile, momentary on switch
SW3	ABC	Switch	Three position toggle switch
SW4	ABC	Switch	Three position toggle switch

3.2.5 IFDR Status Indicators

The IFDR is packaged in a tight metal enclosure for maximum noise immunity. Two LEDs provide status information for the IFDR and status of the communication links to the RVP902. These LEDs have the same interpretation as the RVP7 and RVP8 LED indicators.

Table 4 IFDR LED Indicators

Red (Uplink)	Green (Ready)	Meaning
Blink	Blink	Reset sequence (power-up or from uplink)
On	On	Normal Operation (two-way communication with IFDR and computer are both okay)

3.2.6 IFDR Input A/D Saturation Levels

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

- 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 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 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 +8 dBm. 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 IFDR, and have an insertion loss of 0.5 dB to 1.0 dB. Thus, the input saturation level is +8.5 dBm to +9.0 dBm, 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 -10 dBm to +4 dBm, measured as described in Section C.15 Burst Pulse Alignment on page 374. This is important for making a precise phase measurement on each pulse.

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 RVP900 employs a statistical linearization algorithm to derive correct power levels from targets that are as much as 6 dB 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 IFDR Clock Subsystem

The RVP901 provides a flexible, fully programmable, low jitter clock generator used in sampling the IF inputs and generating the IF outputs.

- **Master Clock Source**—Clock reference can be provided by a 20 MHz on-board oscillator. An external clock reference may also be provided to the RVP901 through CLK-IN (J7).
- **IF Clock Frequency**—The sampling clock frequency is fully programmable between 50 MHz to 100 MHz (see Section 3.2.8 Choice of A/D Sample Rate and Tx Synthesis Rate on page 69) with microhertz (?Hz) resolution. The clock frequency can be chosen independently of the original reference clock frequency that produces it; they do not have to be small integer multiples of each other.
- **Clock Jitter**—IF clock jitter is sub-picosecond allowing the system to maximize the benefits of the 16-bit A/D converters.

The master clock source is software-configurable between the on-board 20 MHz reference or an external source. The external clock option allows the IFDR to be phase locked to a standard system reference; however, the external clock is not a requirement. The internal reference oscillator is a high-quality oscillator, but is not temperature compensated. The internal 20 MHz reference frequency stability is 20 ppm over extended temperature range of -40°C to +85°C (-40°F to +176°F). Its jitter is sub–picosecond.

The IFDR sampling clock is derived from the master clock source, using a novel architecture recommended by Analog Devices. The architecture minimizes jitter, while allowing full flexibility in selecting sampling frequencies between 50 MHz to 100 MHz. The output clock runs at the same frequency as the sampling clock.

When the RVP900 is used in a klystron system, or in any type of synchronous radar, the radar COHO is supplied to the IFDR, so that the sampling clock 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 IFDR internal stable sampling clock, which is user selectable. The internal sampling clock is not affected by the application of the COHO. A/D

samples of the COHO are obtained at the fixed sampling rate, and the (I,Q) data are digitally locked downstream in the RVP900 IF-to-I/Q processing chain (see Figure 13 on page 39). 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 RVP900 is used in the above manner within a synchronous radar system. Both concerns are the result of the IFDR sampling clock being asynchronous with the radar system clock. These concerns are:

- **RVP900 Generates the Radar Trigger**—The trigger signals supplied by the RVP900 are synchronous with the IFDR data sampling clock. This is accomplished by a clock recovery PLL that provides on-board timing, which is identical to the sampling clock in the IFDR. Since the IFDR sampling clock is asynchronous with the radar clocks, the RVP900 trigger outputs are similarly asynchronous. The result is that each transmitted pulse envelope is triggered independently of the COHO phase. The transmitted pulse is still synchronous, but the precise alignment of the amplitude modulated envelope varies.

  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 RVP900 to reject ground clutter and to process multi-mode return signals. For this reason, a synchronous radar system, which is triggered using the RVP900 triggers, still performs optimally using the standard digital COHO locking techniques. In spite of this, some system designers may still prefer that the amplitude envelope be locked to the COHO.
- **RVP900 Receives the Existing Radar Trigger**—When an external trigger is supplied to the RVP900, 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 IFDR selectable sampling clock. The result is that 27.8 nanoseconds of jitter is introduced in the placement of the RVP900 range bins relative to the transmitted pulse.

  The effect of this synchronization jitter is that targets appear to be fluctuating in range by approximately 4.2 m. Although this is small, relative to the range bin spacing, and 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 ?sec pulse may be limited to approximately 43 dB as a result of this 27.8 nanoseconds range jitter. This falls quite short of the usual

expectations of a synchronous radar system in which clutter rejection of 55 dB to 60 dB should be attainable.

The solution to either of the these concerns is to provide some means for the IFDR internal sampling clock to be phase locked to the radar system. If the RVP900 provides the radar triggers, then those triggers become synchronous with the radar COHO. If the RVP900 receives an external trigger, then its range bin clock is synchronous with that external trigger, and there is no synchronization jitter in the range bins.

The IFDR has the option of locking its sampling clock to an external system clock reference through the CLK-IN SMA input. This results in an RVP900 that is fully synchronous with the existing radar timing.

3.2.8 Choice of A/D Sample Rate and Tx Synthesis Rate

The internal system clock, which samples the IF input signals and synthesizes the Tx output waveforms, can be configured to run at any frequency between 50 MHz and 100 MHz. The setup questions in the Mc menu (see Section 4.2.1 Mc — Top Level Configuration on page 104) select the sampling clock frequency and whether the clock is derived from a stable on-board crystal oscillator or from the external CLK-IN SMA reference.

The sample clock frequency is a far-reaching parameter that affects many components of the radar and signal processing system. The choice of frequency is likely to be different for each radar facility, because it represents a trade-off of the following considerations:

- **A/D Quantization Noise and Dynamic Range**—The inherent SNR of the A/D converter chip is spread over a Nyquist band, whose width is determined by the sampling frequency (see Section 3.2.10 IF Bandwidth and Dynamic Range on page 71). As the sampling frequency increases, the A/D quantization noise that is contained within a given Rx bandwidth decreases, which means that the RVP900 becomes "more quiet". The dynamic range varies linearly with sampling frequency; therefore, the RVP900 has 3 dB greater dynamic range at 100 MHz versus 50 MHz clock.
- **Quantization of Trigger Timing and Range Bin Placement**—Triggers generated by the RVP900 are specified by their start time in microseconds, width in microseconds, and polarity. Triggers are always produce that are ±0.5 clocks of these ideal values. However, if you want the triggers to be precisely aligned down to the exact clock edge, the sample clock frequency should be chosen so that trigger edges fall on integer multiples of the clock period. Similarly, the range

bin spacing is specified in meters and are always within half a clock period of the ideal value. The bins can also be placed precisely in range, by choosing a clock period that is an integer multiple of the desired spacing.

- **Maximum Length of FIR Downconversion Filters—**The FIR filters that compute (I,Q) time series from raw IF samples must process those samples at the acquisition clock rate. A filter of a given length in microseconds must contain a greater number of taps (coefficients) as the sample rate increases. For very long filters (greater than 40 ?sec), it may sometimes be necessary to limit the clock rate, in order to achieve the desired impulse response length. The Mt<n> and Ps menus are helpful in determining the maximum length filter that can be achieved for a given RVP900 processing mode (affected by single/dual polarization, range bin spacing, etc).
- **Null Frequency Bands in Synthesized Tx Output**—When a synthesized Tx waveform is generated by the RVP900, there are certain constraints on the output IF frequency and clock frequency. The output frequency can not be within the interval of frequencies 0.4 f to 0.6 f, where "f" is the sample clock rate.

The RVP900 setup menus are helpful in cross-checking many of the above constraints. However, it is ultimately the responsibility of the system installer to choose a sample clock frequency that achieves the best set of trade-offs at each radar site.

3.2.9 External Pre-Trigger Input

Users may supply the RVP900 with their own CMOS-level pre-trigger for installations in which adequate trigger control already exists. The trigger input is provided on the IFDR TRIG-A or TRIG-B SMA connector J8 and J15, or on a TTL or RS-422 I/O line on J3 or J6. The choice is made in the *softplane.conf* file.

The trigger input uses CMOS levels (1.5 V maximum low, 2.5 V minimum high) for improved noise immunity. The SMA inputs may also be driven as high as +100 V or as low as -100 V, without damage. This makes it easier to connect to existing high-voltage trigger distribution systems. The rising or falling edge of this external trigger signal is interpreted by the RVP900 as the pretrigger point. The actual pulse width of the signal does not matter. The delay to range zero is configured through the TTY Setups (see Section 4.2.5 Mt<n>— Triggers for Pulsewidth #n on page 117). The other trigger outputs are then synchronized to the input trigger. The synchronization jitter between the user pretrigger and the other trigger outputs are less than the period of the A/D sampling clock, for example, 10 nanoseconds at a 100 MHz rate.

Trigger jitter can be improved in the case of coherent systems, by phase locking the IFDR to the same reference clock used to generate the external triggers (typically the COHO). The improved IF samples may provide as much as 10 dB of additional sub-clutter visibility.

3.2.10 IF Bandwidth and Dynamic Range

The RVP900 performs best with a wide bandwidth IF input signal. A wideband signal can be made free of phase distortions within the (relatively narrow) matched passband of the received signal. The RVP900 uses an external analog anti-aliasing filter at each of its IF and Burst inputs. These filters block frequencies that would otherwise alias into the matched filter passband. The anti-alias filters have a nominal passband width of 14 MHz. There are options of providing anti-alias filters centered at 30 MHz, 57.5 MHz, and 60 MHz. Therefore, the nominal pass band width for the filter centered at 60 MHz is from 53 MHz through 57 MHz. This is the recommended operating bandwidth for the IF signal, although the RVP900 still works successfully with lesser IF bandwidth.

At 72 MHz sampling rate, the quantization noise introduced by LSB uncertainties is spread over an 36 MHz bandwidth. For an ideal 16-bit A/D converter that saturates at +8 dBm, the effective quantization noise level is:

$$+8dBm - 20log_{10}(2^{16}) - 10log_{10}\left(\frac{36MHz}{1MHz}\right) = -103dBm \ \ (at\ 1MHz\ BW)$$

If samples, from this ideal converter, are processed with a digital filter having a bandwidth of 1 MHz, then an input signal at –103 dBm has a signal-to-noise ratio of 0 dB. A narrower FIR passband (corresponding to a longer transmitted pulse) decreases the quantization noise even further, so that 0 dB SNR achieves at an even lower input power.

In practice, achieving the quantization noise level of an ideal converter becomes more difficult when increasing the number of bits. The 16-bit Analog Devices AD9446 chip has been measured to have a wideband SNR of 79 dB, which is 24 dB less than the 103 dB range expected for an ideal converter. The above calculation for noise density thus becomes:

$$+8dBm - 79dB - 10log_{10}\left(\frac{36MHz}{1MHz}\right) = -87dBm(at\ 1MHz\ BW)$$

The RVP900 receiver power monitor, described in Section 5.5 Pr — Plot Receiver Waveforms on page 163, shows a filtered power level of approximately -87 dBm, when the FIR bandwidth is 1 MHz and the IFDR inputs are terminated in 50Ω.

The inverse correspondence between filter bandwidth and the 0 dB 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 79 dB. However, by utilizing the extra bandwidth of the converter, the RVP900 is able to extend the dynamic range to approximately 103 dB.

To understand this, begin with the 95 dB interval between the converter's +8 dBm saturation level and the -87 dBm 0 dB SNR level at 1 MHz bandwidth. Add to this:

- 4 dB for the statistical linearization that is performed on signals that exceed the saturation level. The RVP900 can recover signal power accurately, even when the A/D converter is driven beyond saturation. Velocity data is also valid, but spectral width may be overestimated.
- 4 dB for usable dynamic range below the 0 dB SNR level. In practice, a coherent signal at -4 dB SNR can easily be measured when 25 or more pulses are used.

The overall dynamic range at 1 MHz bandwidth (approximately 1 µsec transmit pulse) is 95+4+4 = 103 dB. For a 0.5 µsec pulse, the dynamic range is reduced to 100 dB, but it increases to 106 dB for a 2.0 µsec pulse. An actual calibration curve demonstrating this performance is shown in Figure 19 on page 73, for which the RVP900 digital bandwidth was set to 0.53 MHz. An external signal generator, with steps of 1 dB, were used to measure the compression and detection thresholds.

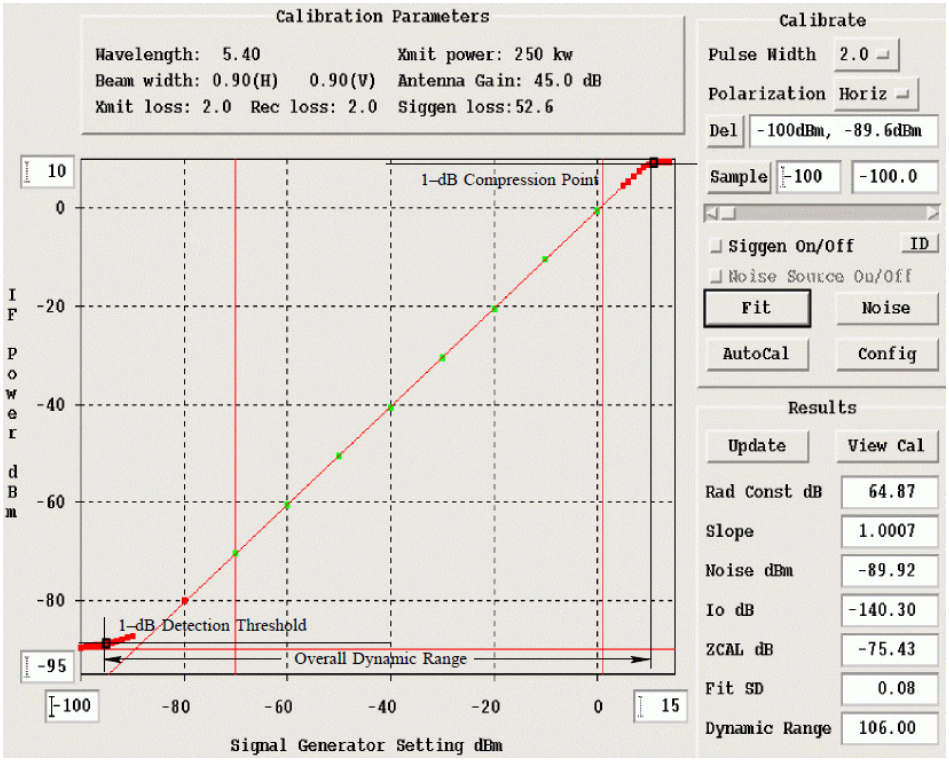


Figure 19 Calibration Plot for a Stand-Alone 16-Bit IFDR

3.2.11 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 IFDR.

Assume that the thermal noise (*kT*) of the system is -114 dBm/MHz, and that the noise figure of the LNA/Mixer is 2 dB. We want to bring this -112 dBm/MHz noise level up into the working range of the IFDR, so that the received echoes can be optimally processed. However, in trying to select the required gain, we realize that we must make a trade-off between preserving the receiver sensitivity that has been established by the LNA, and preserving the overall dynamic range of the IFDR. This is the exact same trade-off that is made in traditional multi-stage analog receiver systems that include a WDR LOG receiver.

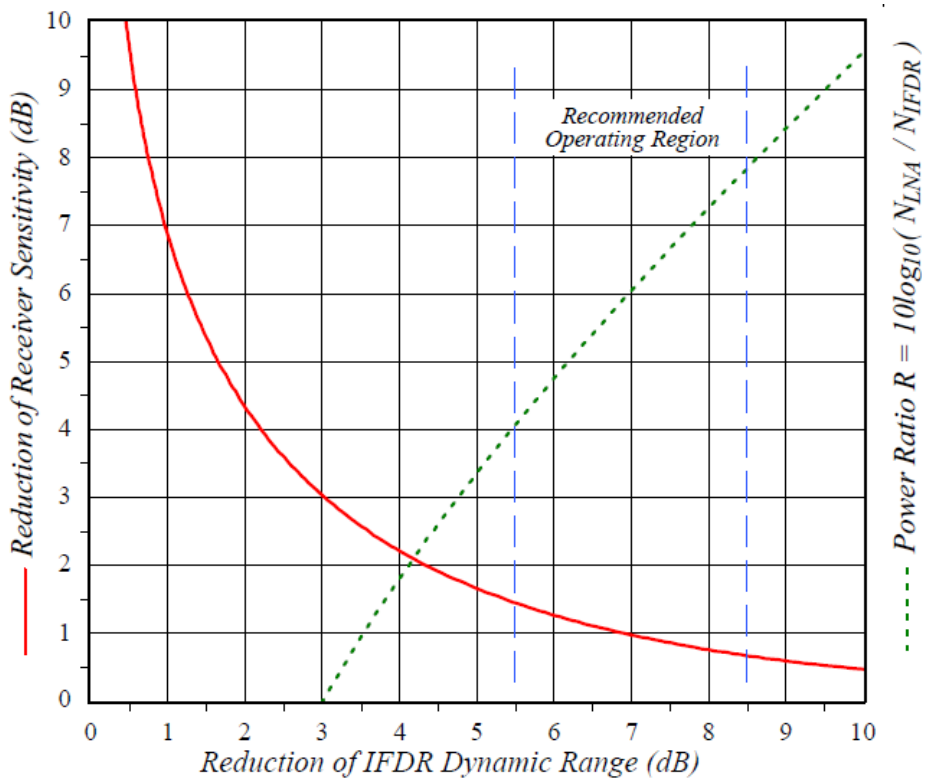


Figure 20 Trade-off Between Dynamic Range and Sensitivity

The solid red curve in on page 77 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_{IFDR} 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. N_{IFDR} 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:

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

Similarly, the reduction of RVP900 dynamic range is the amount by which the IFDR quantization noise is increased over its stand-alone value:

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

Both of these quantities depend only on the ratio of the two powers; therefore, the two equations define a parametric relationship in the dimensionless variable R = (N_{LNA} / N_{IFDR}) . on page 77 was created by sweeping the value of R from 1/9 to 9. The solid red curve shows the locus of (*?DynamicRange*, *?Sensitivity*) points, and the dashed green curve shows *R* (expressed in dB) as a function of *?DynamicRange*. For example, when the LNA noise power is equal to the IFD noise power, *R* is 1.0 (0 dB) and there is a 3 dB 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 1.4 dB and 0.65 dB. The attendant loss of dynamic range falls between 5.5 dB and 8.5 dB, 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 RVP900 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 need to pick an operating point that makes the most of preserving that investment.

on page 77 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 the need for sensitivity versus dynamic range. For this example, we allow a 1 dB loss of sensitivity from the theoretical limit of the LNA/Mixer, and assume a bandwidth of 0.5 MHz.
2. For a 1 dB loss of sensitivity, the *?DynamicRange* is first determined from the solid red curve as 7 dB. The required noise ratio *R* is then read vertically on the dashed green curve as 6.1 dB.
3. The RF/IF gain must bring the front-end thermal noise at -112 dBm/MHz up to a level that is 6.1 dB higher than the IFD noise density of -87 dBm/MHz. The gain is: (-87 dBm/MHz +6 dB) - (-112 dBm/MHz) = 31 dB. This gain does not depend on bandwidth, and therefore is correct for all pulse width/bandwidth combinations.

4. The dynamic range for the complete system at 0.5 MHz bandwidth may now be calculated as 106 dB - 6 dB = 100 dB.
5. After assembling all of the RF and IF components, we can check whether we achieved the correct gain, by verifying a 7 dB rise (independent of bandwidth) in RVP900 filtered power, when the IF-Input cable is connected, versus disconnected.

When designing your RF and IF components, keep in mind that the final amplifier driving the IFDR must be capable of driving up to, perhaps, +14 dBm, so that signals above saturation can still be correctly measured.

3.2.12 IF Gain Based on System Noise Figure

The previous section described how to compute the front-end RF/IF gain based on the desired trade-off 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. It 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 20 dB delivers -90 dBm of output power within a 1 MHz bandwidth when its input is terminated. We would expect the Boltzman thermal input noise, at -114 dBm/MHz, to produce -94 dBm, from an ideal 20 dB amplifier, under the same conditions. The noise figure of the real amplifier is +4 dB, (-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 RVP900.

- To calculate the gain of the IFDR, we apply a calibrated mid-power signal generator directly to its IF-Input, and use the Pr plot (Section 5.5 Pr — Plot Receiver Waveforms on page 163) to print the measured power. For a wide range of analog input power levels, the RVP900 reports the exact same measured digital power; therefore the overall analog/digital gain is 1.0 (0 dB).
- To calculate the noise figure of the IFDR, we set the receiver bandwidth to 1 MHz (Section 5.4.2 Available Subcommands Within Ps on page 151), terminate the IF-Input in 50Ω, and use the Pr plot, this time to examine the in-band thermal noise power. The measured

noise level is around -87 dBm. Since an ideal unity gain amplifier has produced a noise power of -114 dBm in an equivalent bandwidth, the noise figure of the IFDR is 27 dB.

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 2 dB noise figure (noise factor 1.58), and we want to know what gain it must have such that, when cascaded into the IFDR, the overall noise figure is 3 dB. The 27 dB noise figure of the IFDR is equivalent to a noise factor of 501, therefore we have:

$$3dB = 10\log_{10}\left[1.58 + \left(\frac{501 - 1}{G^1_{lin}}\right)\right]$$

from which we solve $G^1_{\text{lin}} = 1204$ (30.8 dB). This agrees with the 31 dB of gain that was computed in the example of the previous section for the same RF/IF components and desired overall performance.

3.2.13 Choice of Intermediate Frequency

The RVP900 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 2 MHz away from any multiple of half sampling rate. At 72 Mhz sample, the multiple are nominally 18 MHz, 36 MHz, 54 MHz, 72 MHz, and 90 MHz). The valid frequency bands are thus:

6-16 MHz, 20-34 MHz, 38-52 MHz, 56-70 MHz, 74-88 MHz

There are many reasons for staying clear of the Nyquist frequency multiples. Most of these considerations apply to all types of digital processors, and are not specific to the RVP900.

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

Another problem that arises with a 35 MHz IF on a magnetron system, is the RVP900 computation of AFC. If the processor can not distinguish 37 MHz from 35 MHz, then it can not tell the difference between the STALO being correctly on frequency, versus being 2 MHz too high. The symmetric AFC tracking range is reduced to the very small interval 34 MHz to 36 MHz.

For similar reasons (that is, transition band width), the digital FIR filter 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 36 MHz is aliased into DC, we are left with the contradictory requirements of a zero very close to the edge of the filter's passband.

3.3 RVP902 Main Chassis

3.3.1 RVP902 Main Chassis Overview

Figure 21 RVP902 Main Chassis

The RVP902 main chassis is available in a variety of forms, depending on the customer requirements. A standard RVP900 system is packaged in a SuperMicro SuperChassis 825MTQ-R700UB which contains at least the following:

- Dual Intel Xeon Quad Core CPU
- Two 500 GB Hard Drives
- 8 GB RAM
- 1 PCI Expansion Slot (used with RCP8 combination)
- DVD/CDROM Drive
- 2 USB, and 1 DB9 Serial on the front panel
- 2 USB, and 1 DB9 Serial on the back panel
- PS/2 KB and Mouse ports on the back panel
- 4 Full-height Expansion Slots
- 3 Low-profile Expansion Slots
- 2 Gigabit LAN interfaces

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

The standard RVP902 chassis is a 19 in, EIA, 2U rackmount unit, 18 in (45 cm) deep. The chassis is usually mounted in a nearby equipment rack on rack slides (provided as standard). A shielded Ethernet cable is provided to connect the IFDR to the main chassis.

The power requirements are 100 VAC to 240 VAC, 50 Hz to 60 Hz, 10–4 Amps. There are dual-redundant power supplies, with two line cords.

WARNING Due to redundant power supplies, RVP902 has a High Leakage Current. To meet EN60950-1 safety standards, the RVP902 chassis ground point must make a low impedance connection to the earth ground.

3.3.3 Power-Up Details

When the RVP902 is powered-up or reset, the host Linux PC goes through an automated boot process that ultimately starts the RVP902 application. The RVP902 then runs extensive internal diagnostics. In most cases, there is no display connected to the RVP902 to monitor the boot sequence. For

troubleshooting, it is useful to connect a display to view any error messages, or you can view the startup log in */usr/iris_data/log/rvp9.log*.

3.3.4 Socket Interface

The RVP902 is configured to listen on a network port. It is ready to interface to a host computer through the network using a program called DspExport. When IRIS/Radar is installed onto the same RVP902 computer, it is already configured to communicate with RVP902 processes through the native interface, bypassing the socket interface. The RVP902 also comes with some built-in utilities such as setup, dspx, and ascope. These utilities are described in the *IRIS Utilities Manual*. Because the RVP902 can only have one program controlling it at a time, use of a local program like dspx blocks network access, and vice versa.

3.3.4.1 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 establishes a connection to the RVP902. At this point, multiple connections are allowed. It only handles the "INFO", "SETU" and "OPEN" commands. Once the "OPEN" command is sent, an exclusive connection for I/O to the RVP902 is established. If a second OPEN request comes in while the first is still active, it fails, and returns the message "Device allocated to another user". To see if it is running on your RVP902, type:

```
$ 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.

3.3.4.2 Source Examples

The source code for DspExport, and for the dsp library, is supplied on the RVP902 release cdrom. This is always installed as part of the install procedure, discussed in *IRIS Software Installation Manual*. DspExport is in ${IRIS_ROOT}/src/rda/dsp, and the dsp library is in ${IRIS_ROOT}/base/dsp_lib. In the library, example code is provided, 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.

3.3.4.3 Socket Protocol

The socket interface basically supports all the "Host Computer Commands" in Chapter 7, Host Computer Commands, on page 255. 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 responds 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 always includes a string indicating what the error was. For Ack, there is optional data following.

On initial socket connection request DspExport provides 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 looks something to:

```
Ack|CanCompress=1,Model=RVP900,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. This means that the server only responds to INFO and OPEN commands. DspExport supports only the six commands:

- Read Command (READ)
- Write Command (WRIT)
- Read Status Command (STAT)
- Set Information Command (INFO)
- Read data available command (RDAV)
- Open the connection for I/O (OPEN)

3.3.4.3.1 Read Command (READ)

Example: "READ|100|" means read 100 bytes from the RVP900. Since the RVP900 interface is a 16-bit word interface, these read sizes should always be even. It always replies with a "Ack|" followed by 100 bytes of binary data, or with a "Nak|"; in other words there can be no partial reads.

3.3.4.3.2 Write Command (WRIT)

Example: "WRIT|<data>", where <data> is some binary data. This data is written to the RVP900. Again, the data size should be even.

3.3.4.3.3 Read Status Command (STAT)

Example: "STAT|" reads the status bits back from the RVP900. This is a 1 bit value, set to 1 if the RVP900 has data available in its output buffer. It returns either "Ack|0", "Ack|1", or a "Nak". This is the equivalent of the dspr_status() call in the dsp library.

3.3.4.3.4 Set Information Command (INFO)

Example:
"INFO|ByteOrder=LittleEndian,WillCompress=1,Version=7.32". This command can be used to inform RVP902 DspExport about the host computer. Current options available are:

- ByteOrder—Informs DspExport of the byte order of the host computer. This is needed because all the data read or written to/from the RVP900 is in 16-bit words. If the host computer has a different byte order from the RVP900, DspExport byte swaps the data.
- WillCompress—Informs 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 replies 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 RVP900 reads, because the reader always specifies the read size.
- Version—Sends the IRIS version.

3.3.4.3.5 Read Data Available Command (RDAV)

Example: "RDAV|100|2|" means read up to 100 bytes of data from the RVP902 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.

3.3.4.3.6 Open the Connection for I/O (OPEN)

Example: "OPEN" means switch from open for "info only" mode to open for I/O. If the signal processor is in use by another device, you get an error in response to this command. Multiple clients are allowed to connect for info only, but only one can do I/O. If you run DspExport with the -803 command line option, you get the legacy behavior, which means that every connection automatically sends 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.

3.3.4.3.7 Read Z Cal Information (RCAL)

Example: "ZCAL" means read the dsp_refl_cal structure from the RVP900 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 want to use the RVP900.

3.3.4.3.8 Reset Kernel FIFOs (RKFF)

Example: "RKFF|2|" means reset the kernel FIFOs on the RVP900. The argument specifies which direction FIFOs to reset.

3.3.4.3.9 Read Setup Information (SETU)

Example: "SETU" means read the dsp_manual_setup structure from the RVP900 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 want to use the RVP900.

3.3.4.3.10 Write Z cal Information (WCAL)

Example: "WCAL|..." writes the dsp_refl_cal structure to the RVP900 to be saved there.

3.3.4.3.11 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 or RVP900. 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/RVP900 in regards to the FIFO reset command. This is the RVP900 command 0x008C (see Reset (RESET) on page 307). The RVP900 is unable to read incoming commands if the output FIFO is entirely full. Therefore, if you put the RVP900 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 Digital AFC Module (DAFC) is a small, self-contained circuit board, which can passively "eavesdrop" on the RVP900 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 in x3.75 in) assembly diagram of the board is shown in Figure 22 on page 85. It can be installed in the radar system as a bare board, or packaged into a small metal enclosure. The RVP900 uplink transmission come from J15 port of the IFDR. A question within dspx settings sets the function of J15 port between generalized trigger function or DAFC uplink.

Digital I/O Model – 0:Common, 1:TDWR, 2:WSR88D : 0
  Trig–A SMA – 0:Trig1, 1:PBit1, 2:DAFC : 2
  Trig–B SMA – 0:Trig2, 1:PBit2, 2:ExtIn : 0

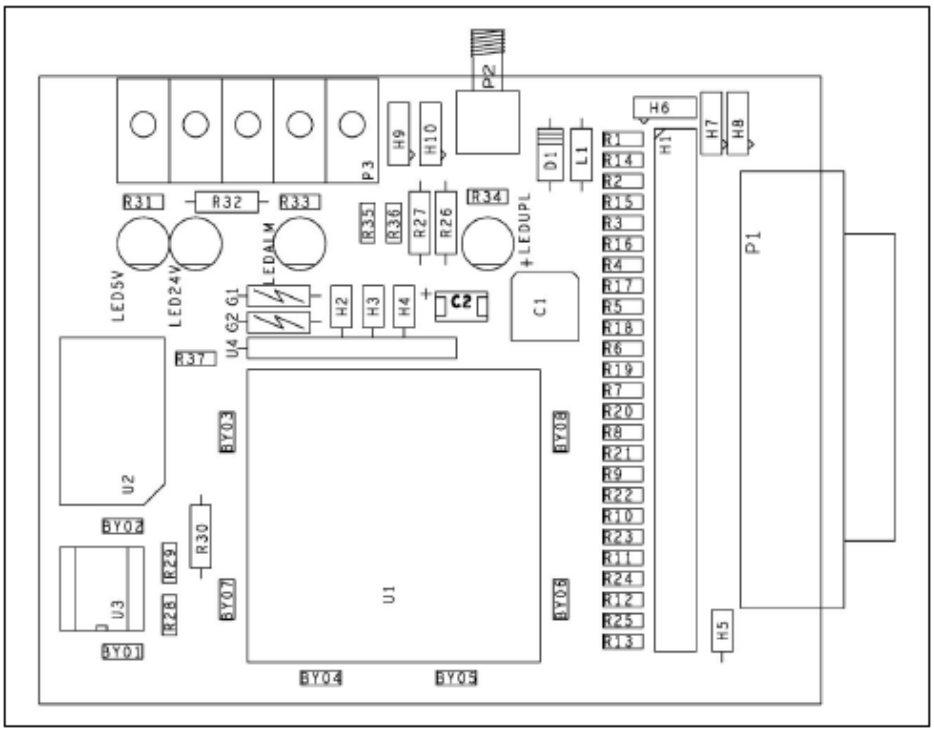


Figure 22 Assembly Diagram of the DAFC

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:

- 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.
- 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.

The digital output lines are made available as TTL levels on a 25-pin female "D" connector (P1). There are 130? resistors (R1 through 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 0.6 V maximum requires that the STALO inputs be pulled up to +5 with nothing less than (approximately) 1.2K?. Put another way, the low level input current of the receiving device should not exceed 4.5 mA. Most STALOs, that we have seen, use 5-20K? pull-up resistors, so this should not be a problem.

All 25 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 (summarized in Table 5 on page 87). 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 RVP900 system, whose uplink uses the older style 16-bit AFC uplink format. In this case, you have to make most or all of the pin assignments using wirewrap

wire to connect each bit to its corresponding pin. This is somewhat tedious, but hopefully one of the three formats is a reasonable starting point for doing the wiring. By far, the most preferable solution is to use the Pinmap uplink protocol (available since Rev.19), which allows for complete software mapping of all 25 external pins.

Table 5 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, +5 V, and +24 V 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 3 K/6 K resistive terminators are also available at H5 for pulling pins up to approximately +3.3 V, 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 +5 V 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 +24 V (approximately) 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 +5 V and +24 V. Terminal block Pin #1 is +5 V, Pin #2 is +24 V, 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 does not affect the uplink in any way. 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 indicates 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.

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", the crystal frequency should be equal to the IFD 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 Section 4.2.6 Mb — Burst Pulse and AFC on page 126 and Section 6.1.3 Automatic Frequency Control (AFC) on page 187.

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 through 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 6 on page 88.

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 +5 V and +24 V pins, and two grounds are provided. An "alarm" output is also available.

Table 6 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	Bit-0
11	6	Bit-2	12	19	Bit-1
13	7	Bit-3	14	20	Bit-10
15	8	Bit-11	16	21	Bit-4
17	9	Bit-9	18	22	Bit-5

Table 6 Pinout for the CTI MVSR-xxx STALO (Continued)

Ribbon Pin	"D" Pin	Function	Ribbon Pin	"D" Pin	Function
19	10	Bit-8	20	23	Bit-6
21	11	Bit-7	22	24	Ground
23	12	Bit-12	24	25	Bit-13
25	13	Inhb	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 +5 V and +24 V to the STALO, and should be wire wrapped to the internal power posts. The STALO and DAFC power, are then supplied externally through the terminal block on the DAFC.

Sixteen jumpers should be installed to connect the Control and Status lines, that is, pins 4, 6–13, 18–23, and 25. We 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 MHz to 6020 MHz in 100 KHz steps. In this example, we assume that we need an AFC frequency span of 5580 MHz to 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 × 100KHz) = 5580MHz
5200MHz + (4000 × 100KHz) = 5600MHz

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

3.4.2 Example of a MITEQ MFS-05.00–05.30–100K–10MP STALO

The electrical interface for this STALO uses a 25-pin “D” connector with the pin assignments (Table 7 on page 90).

Table 7 Pinout for the MITEQ MFS–xx.xx–xx.xx–100K–xxMP Synthesizers

Ribbon Pin	"D" Pin	Function	Ribbon Pin	"D" Pin	Function
1	1	Ground	2	14	Ground
3	2	+20 VDC	4	15	+20 VDC
5	3	+5.2 VDC	6	16	+5.2 VDC
7	4	Test Point	8	17	10 MHz (1)
9	5	TTL Alarm	10	18	1 MHz (8)
11	6	Phase Voltage	12	19	1 MHz (4)
13	7	100 MHz (8)	14	20	1 MHz (2)
15	8	100 MHz (4)	16	21	1 MHz (1)
17	9	100 MHz (2)	18	22	100 MHz (8)
19	10	100 MHz (1)	20	23	100 MHz (4)
21	11	10 MHz (8)	22	24	100 MHz (2)
23	12	10 MHz (4)	24	25	100 MHz (1)
25	13	10 MHz (2)	26	--	--

First configure the DAFC pins themselves. Pins 1 and 14 are ground, and are connected with wirewrap wire to the nearby ground posts. Pins 2 and 15 are connected with a wirewrap wire to the +24 V posts, and pint 3 and 16 are connected with wirewrap wire to the +5 V posts. The Alarm on pin 5 is wired to the alarm post. Pins 7 through 13 and pins 17 through 25 all are signal pins, so we plug in a jumper for each of these 16 pins. We use pinmap uplink protocol, so H3 and H4 are removed. Also, remove H5 and leave H1 on.

In this example, we assume that we wish to control the STALO in 100 KHz steps from 5.1330 GHz to 5.1830 GHz. This can be done with the following setups from the Mb section:

```
AFC span- [-100%,+100%] maps into [ 1330 , 1830 ]
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:GND   Pin04:GND   Pin05:GND
Pin06:GND   Pin07:15    Pin08:14    Pin09:13    Pin10:12
Pin11:11    Pin12:10    Pin13:09    Pin14:GND   Pin15:GND
Pin16:GND   Pin17:08    Pin18:07    Pin19:06    Pin20:05
Pin21:04    Pin22:03    Pin23:02    Pin24:01    Pin25:00
FAULT status pin (0:None): 5, ActLow: YES
```

3.5 IFDR DAFC Uplink Protocol

This section describes the interface from the IFDR to the DAFC. The interface is actually backward compatible to the legacy RVP7. 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

3.5.1 Using the Legacy IFD Coax Uplink

In the previous RVP7 processor, the Coax Uplink was the IFD single line of communication from the main processor board. All of the information that was needed by the IFD arrived through this uplink. 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 DAFC.

The RVP900 uses a single CAT5e Uplink/Downlink cable between the IFDR and and computer server. 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 IFDR. Any hardware that used to be attached to the RVP7 coax uplink can still be driven from this new IFDR 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 V and +12 V (open circuit) electrical levels. The output impedance of the uplink driver is approximately 55? . When the cable is terminated in 75?, the overall positive voltage swing is approximately 8.6 V.

The electrical characteristics of the uplink have been optimized for balanced "groundless" reception. The recommended eavesdropping circuit is shown in Figure 23 on page 92, 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 33K? isolation resistors; no direct ground attachment is made to the shield. The 500? resistors provide the local ground reference, and the 47K? resistor supplies a bias to shift the unipolar uplink signal into a bipolar range for the comparator.

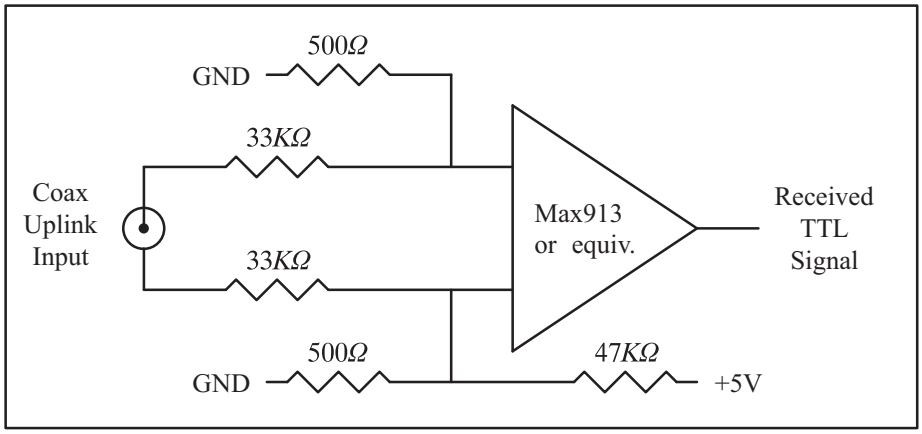


Figure 23 Recommended Receiving Circuit for the Coax Uplink

The uplink signal, shown in Figure 24 on page 92, is periodic at the radar pulse repetition frequency, and conveys two distinct types of information to the IFDR. 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.

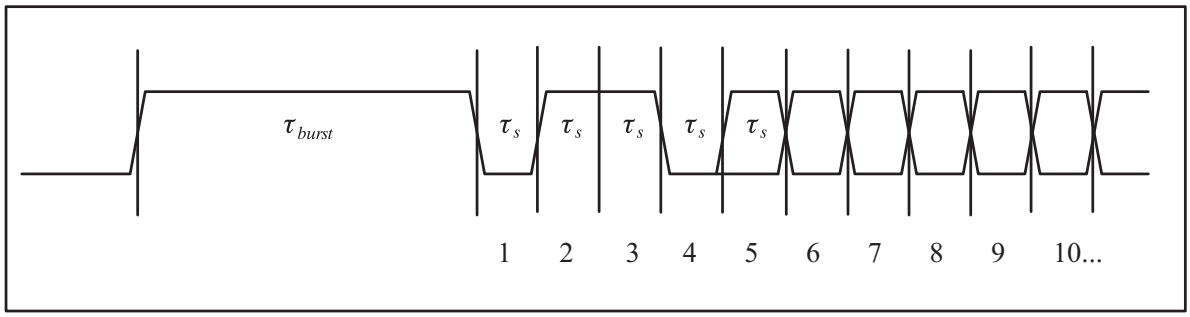


Figure 24 Timing Diagram of the IFD Coax Uplink

The first part of each pulse sequence is a variable length "burst window", which is centered on the transmitted pulse and has a duration $?_{burst}$ approximately 800 nanoseconds greater than the length of the current FIR matched filter. The burst window defines the interval of time during which the IFDR transmits digitized burst pulse samples, rather than digitized IF samples, on its downlink. The exact placement and width of the burst window depends on the trigger timing and digital filter specifications that

the user has chosen, usually through 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 IFDR. 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 8 on page 93 provides the interpretation of the serial data bits.

Table 8 Bit Assignments for the IFDR Coax Uplink

Bit(s)	Meaning
1 to 4	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.
5 to 20	16-bit, multi-purpose data word, MSB is transmitted first (see below)
21	Reset Request. This bit sets in just one transmitted sequence whenever an RVP900 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)
23 to 24	Diagnostic select bits. These are used by the RVP900 power-up diagnostic routines; they are both 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 IFDR.

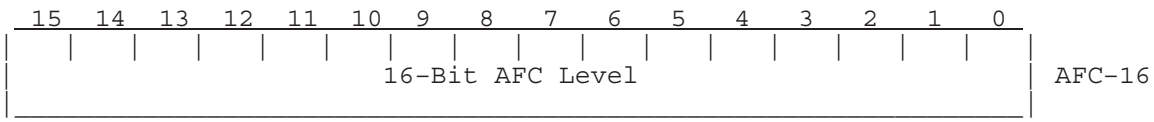
The period $?_s$ of the serial data is $(128/f_{aq})$, where f_{aq} is the acquisition clock frequency given in the Mc section of the RVP900 setup menu. For the default clock frequency of 71.9502 MHz, the period of the serial data is 1.779 ?sec. The logic that is receiving the serial data should first locate the center of the first data bit at $(0.5 \times ?_s)$ past the falling edge at the end of the burst window. Subsequent data bits are then sampled at uniform $?_s$ 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 therefore, 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 5 MHz or 10 MHz clock were available, then sampling at 1.8 µsec intervals (1:85 error) would be fine. Similarly, one could sample at 1.75 µsec based on a 4 MHz or 8 MHz 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

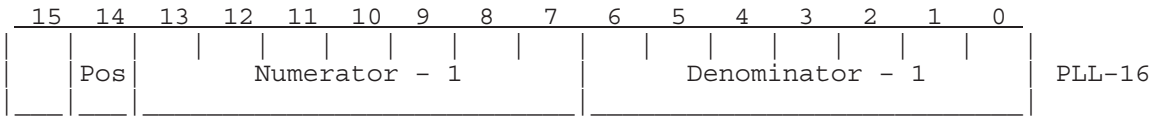
The serial 16-bit data word has several different interpretations according to how the RVP900 has been configured, and whether Bit #22 of the uplink stream is set or clear. The evolution of these different formats has been in response to new features being added to the IFDR (Section 3.2 RVP901 IFDR Installation on page 62), and the production of the DAFC module (Section 3.4 Digital AFC Module (DAFC) on page 84).

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 remains available to guarantee compatibility with older equipment.



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

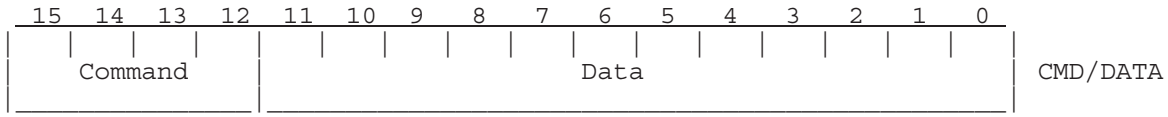
When the IFDR is jumpered for phase locking to an external reference clock, then Bit #22 is clear and the data word conveys the PLL clock ratio, and the Positive/Negative deviation sign of the Voltage Controlled Crystal Oscillator (VCXO). This format is commonly used with klystron systems, especially when the RVP900 is locking to an external trigger.



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. 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 are not 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 Section 4.2.6 Mb — Burst Pulse and AFC on page 126); the 12-bit data with CMD=3 then holds 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. 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 IFDR. 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 150 KHz to 900 KHz band, with negligible residual power above 1.4 MHz. 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 72 MHz single channel
		1 : Half-band 36 MHz dual channel
	Data<9>	0 : Low Half-band 1: High Half-band

CHAPTER 4
TTY NONVOLATILE SETUPS

The RVP900 provides an interactive setup menu that is accessed either from a serial TTY, or from the host computer interface. Most of the RVP900 operating parameters are viewed and modified with this menu, and the settings are saved in non-volatile RAM, so that they take immediate effect on start-up. This permits custom trigger patterns, pulse width 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 are examined in detail (both in the time and frequency domain) and the digital FIR filter is 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
```

The following prompt displays:

```
$dspx
Digital Signal Processor 'Chat'
Checking for code upgrades...Okay
(Type ^C to exit Chat Mode)
```

The interactive setup menu is invoked by pressing **ESC** on the TTY. If the key cannot be found on the keyboard, you can sometimes use **CTRL** + **[** to generate the ESC code. RVP900 responds with the following banner and command prompt.:

```
                Vaisala Incorporated, USA
RVP9 Digital IF Signal Processor V13.1 IRIS-8.13.1
-------------------------------------------------
RVP9>
```

The banner identifies the RVP900 product, and displays the RVP900 software version (for example, V13.1) and IRIS software version (for example, IRIS 8.13.1). This information is important when RVP900 support is required. It is also displayed on the printout of the **V** command.

The **Q** command exits from the menus and reloads the RVP900 with the changed set of current values.

NOTE It is important to quit from the menus before attempting to resume normal RVP900 operation. Portions of the RVP900 command interpreter remain running while the menus are active (so that the TTYOP command works properly), but the processor as a whole does not function until the menus are exited.

From the command prompt, typing **help** or ? to get 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 Virtual Oscilloscope
   Pa - Tx Pulse Ambiguity Diagram
   Pb - Burst Pulse Timing
   Ps - Burst Spectra and AFC
   Pr - Receiver Waveforms
   P+ - Visual Test Pattern
V: View Card and System Status
   Vz - Like 'V', but resets internal states
   Vp - Show Processing and Threshold values
?: Print all Menu Commands (this list)
   ?? - Print all Current Setup Settings
```

```
@: Display/Change the Current Major Mode
Q: Quit
```

4.1.1 Factory, Saved, and Current Settings

- *current settings*—Collection of setup values with which the RVP900 is presently operating
- *saved settings*—Collection of values stored in non-volatile RAM. The saved settings are restored (made current) each time the RVP900 starts
- **S** command—Saves the current settings into the non-volatile RAM
- **R** command—Restores those non-volatile values so that they become the current settings
- **F** command—Initializes the current settings with factory default values of RVP900. The factory default values do not correspond to any user installation, and they are not meant to be applied, in normal situations. **F** followed by **S** saves factory defaults in non-volatile RAM, so the user's site specific power up settings are overwritten, irreversibly, and the RVP900 powers up in its manufacturing mode. **This step is not recommended**.

RVP900 retains all of its saved settings when new software releases are installed. The new version of the code automatically uses all of the previous saved values; however, if RVP900 detects a new setup parameter, it is set to a factory default value. A warning is printed whenever this occurs (see Section 4.1.2 V and Vz – View Card and System Status on page 100).

There is also support for intermediate minor releases of RVP900 code. Each software release has a major version number (for example, 8.13), plus a minor version number for intermediate "official" releases (for example, 8.13.2). The minor number starts at zero at the time of each “official” release, and then increments until the next “official” release. RVP900 includes the minor release number (if it is not zero) in the printout of the **V** command. Similarly, the minor release number of the code that was last saved in the non-volatile 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 RVP900 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 and Vz – View Card and System Status

The **V** command displays internal diagnostics. This information is for inspection only, and cannot be changed from the TTY. If invoked as **Vz**, the various counters are cleared, so that subsequent **V** commands show what has accumulated since the last **Vz**. The view listing displays:

```
Configuration and Internal Status
-------------------------------
```

and then prints the following lines:

```
RVP9 Digital IF Signal Processor V13.1(Pol) IRIS-8.13.1
```

This line is the revision level of the RVP900 software and IRIS version.

```
Settings were last saved using V12.3
```

This line displays which version of RVP900 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.

```
RVP9 started at: 13:07:33 3 JUN 2012
Current time is: 13:14:03 3 JUN 2012
```

These lines provide information about when the RVP900 was started, current system time, and implicitly, the uptime.

```
CPU-Type: Pentium(R) 4 Hyperthreaded
```

This line displays processor information.

```
IPP-Library: libippsw7.so v4.0 4.0.19.77
```

This line displays information about the Intel libraries used for RVP900 processing.

```
Diagnostics: PASS
```

If errors were detected by the startup diagnostics, then an error bitmask is shown on the first line. `PASS` indicates that no errors were detected.

```
Processes and Threads:
RVP9Proc-0  – PID:6917  Priority:10 Policy:RealTimeRR
RVP9Proc-1  – PID:6918  Priority:10 Policy:RealTimeRR
```

```
 Chat/Plot  –  PID:6909  Priority:10 Policy:RealTimeRR
  Watchdog  –  PID:6909  Priority:10 Policy:RealTimeRR
 Burst/AFC  –  PID:6909  Priority:10 Policy:RealTimeRR
  HostCmds  –  PID:6909  Priority:11 Policy:RealTimeRR
    Angles  –  PID:6909  Priority:12 Policy:RealTimeRR
  RtCtrl-0  –  PID:6909  Priority:12 Policy:RealTimeRR
  RtCtrl-1  –  PID:6909  Priority:12 Policy:RealTimeRR
   IQ-Data  –  PID:6909  Priority:13 Policy:RealTimeRR
```

The `Process and Threads` list displays RVP900 processes and their related priority. All RVP900 processes/threads should be running under RealTimeRR policy to guarantee adequate attention from the processors.

```
Shared library build dates:
RVP9/Main/Core:   Mon  Apr  2 16:01:12 EDT 2012
RVP9/Main/Open:   Mon  Apr  2 16:01:13 EDT 2012
RVP9/Main/Site:   Mon  Apr  2 16:01:12 EDT 2012
RVP9/Proc/Core:   Mon  Apr  2 16:01:14 EDT 2012
RVP9/Proc/Open:   Mon  Apr  2 16:01:16 EDT 2012
RVP9/Proc/Site:   Mon  Apr  2 16:01:14 EDT 2012
```

This section provides RVP900 developers with information about code resources.

```
GPS:Inused
```

This line shows the status of optional GPS time synchronization of the RVP900 triggers and range bins.

```
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 IFDR 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 139.

```
IFD Temperature – Chassis: 37C (99F), FPGA: 38C (100F) )
```

This line shows the case temperature and FPGA chip core temperature, both in Centigrade and Fahrenheit degrees.

```
Receiver mode: 0 (Standard single channel)
```

This line shows the presently operating receiver mode.

```
TrigRAM using 4.8% of 588-KBytes, TrigCount:8777651
```

`TrigRAM` provides resource information for those who are implementing custom waveforms. `TrigCount` is simply a cumulative count of the number of triggers since the last **Vz** or overall reset.

4.1.3 Vp – View Processing and Threshold Values

The **Vp** command displays internal parameters that affect the moment processing within the RVP900. This information is for inspection only, and can not be changed from the TTY.

```
Threshold Settings for All Data Parameters
          LOG      CSR      WSP      SQI      PMI        TCF (Equation)
        -------  -------  -------  -------  -------  ------------------------------------
   DBZ:  0.75dB  -18.0dB  5.0 dB   0.149    0.449  0x8888 ( LOG & CSR )
   DBT:  0.75dB  -18.0dB  5.0 dB   0.149    0.449  0xFFFF ( All Pass )
   VEL:  0.75dB  -18.0dB  5.0 dB   0.398    0.449  0xC0C0 ( CSR & SQI )
   WID:  0.75dB  -18.0dB  5.0 dB   0.398    0.449  0xA000 ( LOG & SQI & SIG )
   ZDR:  0.75dB  -18.0dB  5.0 dB   0.398    0.449  0xAAAA ( LOG )
   KDP:  0.75dB  -18.0dB  5.0 dB   0.398    0.449  0xC0C0 ( CSR & SQI )
 PHIDP:  0.75dB  -18.0dB  5.0 dB   0.398    0.449  0xC0C0 ( CSR & SQI )
 RHOHV:  0.75dB  -18.0dB  5.0 dB   0.398    0.449  0xC0C0 ( CSR & SQI )
   SQI:  0.75dB  -18.0dB  5.0 dB   0.398    0.449  0xFFFF ( All Pass )
  LDRH:  0.75dB  -18.0dB  5.0 dB   0.398    0.449  0xAAAA ( LOG )
  RHOH:  0.75dB  -18.0dB  5.0 dB   0.398    0.449  0xC0C0 ( CSR & SQI )
  PHIH:  0.75dB  -18.0dB  5.0 dB   0.398    0.449  0xC0C0 ( CSR & SQI )
  LDRV:  0.75dB  -18.0dB  5.0 dB   0.398    0.449  0xAAAA ( LOG )
  RHOV:  0.75dB  -18.0dB  5.0 dB   0.398    0.449  0xC0C0 ( CSR & SQI )
  PHIV:  0.75dB  -18.0dB  5.0 dB   0.398    0.449  0xC0C0 ( CSR & SQI )
HCLASS:  0.75dB  -18.0dB  5.0 dB   0.398    0.449  0xC0C0 ( CSR & SQI )
   SNR:  0.75dB  -18.0dB  5.0 dB   0.398    0.449  0xFFFF ( All Pass )
  DBZA:  0.75dB  -18.0dB  5.0 dB   0.149    0.449  0x8888 ( LOG & CSR )
  DBTA:  0.75dB  -18.0dB  5.0 dB   0.149    0.449  0xFFFF ( All Pass )
```

The five threshold parameters are LOG (receive power above noise), CSR, Weather Signal Power (WSP), SQI, and Polarimetric Met Index (PMI). The Threshold Control Flags (TCF) column shows both numeric and symbolic forms. See Section 7.3 Setup Operating Parameters (SOPRM) on page 259 for details regarding the threshold parameters and TCF values.

4.1.4 @ – 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 RVP900 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.

4.2 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 displayed 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 new values are entered. In all cases, the numbers are checked to be within reasonable bounds, and an error message (listing those bounds) is displayed if the limits are exceeded.

NOTE	Changes to the settings (generally) do not take effect until after the **Q** command is typed, at which point the RVP900 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, you 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 type **M** to make the changes starting from that point, and then type **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 IFDR.

```
IP address of networked RVP9/IFD: 10.0.1.254
```

This is the ethernet address of the remote IFDR module. For best results, the IFDR should be attached to a dedicated ethernet port of the host computer and with no routers, switches or hubs in between.

```
Maximum ethernet incoming UDP frame length : 8192
```

The (I,Q) time series data from the IFDR are transmitted to the host computer through ethernet UDP packets. Depending on the receiver configuration (dual-pol, bin spacing, etc.) the time series data rate may be as high as 50 MBytes/second. To improve the efficiency of packet transmission and reception it is very helpful to employ "jumbo" packets, that is, packets that are longer than the typical 1500-byte standard. The RVP900 allows jumbo packets up to 8192 bytes to be used.

Limits: 250 to 8192 bytes. Use, for example, "ifconfig mtu 8192 eth0" to set the receiver Maximum Transfer Unit size on the host computer. Also make sure that any inline ethernet switches are also configured to transmit the jumbo packets.

```
Receive buffer size for incoming UDP packets : 1500000
```

The UDP time series packets are send-and-forget, meaning that the IFDR assumes that every transmitted packet is correctly received by the RVP900 host computer. This assumption is reliable as long as there is sufficient buffering of the incoming data to prevent overruns within the Linux environment. In most cases the 1.5 MByte default provides reliable reception, but this can be increased to 5 MBytes if "UDP acquisition ID mismatch" errors are ever seen in the RVP900 diagnostic log.

Limits: 100000 to 5000000 bytes

```
IFD synthesized system clock: 72.0000000 MHz
```

This is the frequency of the acquisition sampling clock in the IFDR module. The clock is synthesized on-board using a specialized low-jitter PLL that allows any frequency to be generated merely by choosing it here to the nearest 0.1 Hz.

Limits: 50 MHz to 100 MHz

```
IFD clock is derived from an external reference: YES
External input reference: 10 MHz
```

The synthesized system clock can be derived from an internal crystal oscillator or from an externally applied reference clock. When the external source is used, its frequency is specified here.

```
Live angle input – 0:None, 1:SimRVP, 2:SimIFD, 3:TAGs, 4:S/D,
5:RtCtrl : 0
```

This setting is used to configure the input of live antenna angles. If angle data are supplied outside of the RVP900, e.g., by and RCP8 making direct calls to the IRIS antenna library, then select None. For test purposes, the SimRVP and SimIFD options implement an antenna simulator either in the host computer or the remote IFDR. Parallel angle inputs can be used through TAGs, and 3-wire synchros can hookup using the S/D option. Lastly, RtCtrl should be selected when custom software has been written to deduce antenna angles from the real–time state machine controller.

```
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
```

If you have selected TAGs, then you have these options. The incoming TAG input bits may be selectively inverted through each of the 16-bit words. The values are displayed in Hex. Setting a bit causes 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 RVP900 was in an awkward form such as unsigned integer tenths of degrees, i.e., 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 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.

```
Co-Polarized signal is always on the primary Rx: Yes

# Rx Mode Description
- ----------------------
0 Standard single channel
1 --Reserved--
2 Legacy RVP8/2005 WDN compatibility
3 Standard dual channel

Default receiver mode: 0
```

The IFDR can operate in one of several fundamental modes for the acquisition of (I,Q) time series data. For details, seeSection 6.1.2 RVP900 Receiver Modes on page 185.

NOTE The receiver mode is chosen in the Mc menu, but changes do not take effect until they are saved and the RVP900 is restarted.

4.2.2 Mp — Processing Options

```
Default Spectral Window- 0:User, 1:Rect, 2:Hamming,
3:Blackman : 0
```

Whenever power spectra are computed by the RVP900, the time series data are multiplied by a (real) window prior to computation of the Fourier Transform (DFT). You may use whichever window has been selected through SOPRM word #10 "`0:User`", or force a particular window.

In case of IRIS application, the setting "`0:User`" refers to the window defined in the Setup Utility block RVP "Signal Processing Options". In case of ascope application, the setting "`0:User`" enables the button "Spect Win". At RVP900 startup, "`0:User`" refers to the settings saved in rvp9.conf block 'opprm.filter' bits 9,10,11.

```
Allow continuous sizes for power spectra: YES
```

The power spectra computed within the RVP900 are normally not constrained to be powers of two in length. However, answering NO re-applies that constraint to mimic the behavior of older processors

```
R0/R1/R2 Processing- 0:Never, 1:User, 2:Always : 1
```

Controls whether R0/R1/R2 or R0/R1 estimates are used to compute the spectrum width (see Section 6.3.5 Spectrum Width Algorithms on page 219). Selecting "0" unconditionally disables the R2 algorithms, regardless

of what the host computer requests in the SOPRM command. Similarly, selecting "2" unconditionally enables R2 processing. Bit-7 of SOPRM word #2 is the host computer's interface to this function when the "`1:User`" case is selected (see Section 7.3 Setup Operating Parameters (SOPRM) on page 259). The Vaisala radar application (IRIS) uploads R0/R1/R2. Ascope uploads the button "R2 algorithms" setting in the block Gen Setup. At RVP900 startup, the user setting is retrieved from rvp9NV.opprm.iflags.

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

Controls whether "cluttery" bins are rejected prior to being averaged in range. Bit-8 of SOPRM word #2 is the host computer's interface to this function when the "`1:User`" case is selected (see Section 7.3 Setup Operating Parameters (SOPRM) on page 259). The functionality is dependent on the application settings for CCOR thresholds, in addition (see Section 6.3.2 Range Averaging and Clutter Microsuppression on page 215). The Vaisala radar application (IRIS) requests for Clutter Microsuppression. Ascope uploads the button "Clutter Microsuppression" setting in the block Gen Setup.

```
PPP autocorels from DFTs - 0:Never, 1:User, 2:Always : 1
```

When autocorrelation terms are computed from power spectra, the results differ from a strict time-domain calculation in that an “end-around” term is introduced as a result of circular rather than linear convolution. Requesting PPP-style autocorrelations correct this spurious term when the computations are done through DFTs. Bit-11 of SOPRM word #10 is the host computer's interface to this function when the "`1:User`" case is selected (see Section 7.3 Setup Operating Parameters (SOPRM) on page 259). The Vaisala radar application (IRIS) does not request PPP-style autocorrelations, while the ascope requests it. At RVP900 startup, the user setting is retrieved from rvp9NV.opprm.iflags.

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

This question allows you to choose whether the RVP900 unfolds velocities using a simple ($V_{high} - V_{low}$) algorithm, rather than the improved algorithm described in Section 6.7 Dual PRF Velocity Unfolding on page 240. Bit–11 of SOPRM word #10 is the host computer's interface to this function when the "`1:User`" case is selected (see Section 7.3 Setup Operating Parameters (SOPRM) on page 259).

NOTE This setup question is included for research customers only. The improved unfolding algorithm should still be used in all operational systems because of its lower variance. The Vaisala radar (IRIS) and ascope applications upload the improved unfolding method ("`0:Never`").

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

This question allows you to choose whether the RVP900 attempts to run its standard processing algorithms, even when a custom trigger pattern has been selected through the SETPWF command. Generally, it does not make sense to do this, so the default setting is "0:Never". Bit–12 of SOPRM word #10 is the host computer's interface to this function when the "`0:Never`" case is selected (see Section 7.3 Setup Operating Parameters (SOPRM) on page 259).

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

This parameter provides an additional 6 dB 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 RVP900 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. This behavior only occurs 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 hold-off between successive rays, expressed as a percentage of the number of pulses contributing to each ray. Choosing 100% (the default) produces rays whose input data do not overlap at all, that is, whose rate is exactly the PRF divided by the sample size. Choosing 0% gives 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 RVP900 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 +4 dBm). 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 is applied. Choosing 0 dB disables the algorithm entirely.

Limits: 0 dB to 5 dB

```
Apply amplitude correction based on Burst/COHO: YES
  Time constant of mean amplitude estimator: 70 pulses
```

The RVP900 can perform pulse–to–pulse amplitude correction of the digital (I,Q) data stream based on the amplitude of the Burst/COHO input. For details, see Section 6.1.7 Correction for Tx Power Fluctuations on page 194.

Limits: 10 pulses to 500 pulses

```
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
```

These questions only appear if you have selected WDR in the **Mc** (see Section 4.2.1 Mc — Top Level Configuration on page 104). 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, i.e., unaffected by eventually approaching the noise floor of the LoGain channel.

The RVP900 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) is set in GPARM Immediate Status Word #4.

```
Interference Filter  0:None, Alg.1, Alg.2, Alg.3: 1
Threshold parameter C1: 10.00 dB
Threshold parameter C2: 12.00 dB
```

The RVP900 can optionally apply an interference filter to remove impulsive-type noise from the demodulated (I,Q) data stream. For details, see Section 6.1.5 Interference Filter on page 189.

```
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 unfold 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 are 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 RVP900 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, the slopes are both zero, so that the surveillance data is 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.

```
T/Z/V/W computed from: H-Xmt:YES V-Xmt:NO
```

This controls 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 only appears when dual simultaneous receivers have been configured.

```
Polarimetric Power Params - NoiseCorrected:YES
Polarimetric Correlations - NoiseCorrected:YES
```

These configurations specify whether noise correction are applied to the dual-pol measurements.

```
PhiDP - Negate: NO , Offset:0.0 deg
```

You can configure the sign and offset corrections for $?_{DP}$.

```
Polarimetric Attenuation Correction:Yes
```

This is to correct for intervening weather attenuation. It uses the change of angle in DP . The tuning numbers are taken from the *dualpol.conf* file.

```
KDP computation – 0:LSQ, 1:Weighted LSQ, 2:Cubic Splines
```

Choose what kind of K_{dp} computation you would like to use. "LSQ" means a least square fit to a linear function. "Weighted LSQ" means a weighted least square fit to a linear function. FIR filter coefficients are used as weights. "Cubic Splines" means smoothing and numerical derivatives calculation using a CSU's "Adaptive estimation of specific differential phase" algorithm.

```
KDP – LSQ Length: 4.00 km
```

For LSQ K_{dp} computation, select the length here.

```
KDP – LSQ Weights(FIR) Width: 4.00 km
```

For Weighted LSQ K_{dp} computation, select the FIR width here.

```
KDP – Standard Smoothing Factor: 0.10
KDP – Adaptive Smoothing Factor: 1.10
```

For cubic spline K_{dp} computation: Smoothing factor for the first, non–adaptive, stage of cubic splines processing. Then the smoothing factor for the second, adaptive, stage of the cubic splines processing.

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

A typical installation might use H-Xmt:YES, V-Xmt:No, Co-Rcv:YES, Cx-Rcv:NO. Including both transmissions will decrease the variance of (T/Z/V/W); but most 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.

```
Melting height: 3000 meters
```

The melting height is used in the HydroClass calculations, as well as recorded with the data. This is the height above the radar, so you need to include the altitude of the radar when computing. Normally this is set automatically by the controlling application.

```
Enable noise power based correction of Z0: No
```

If set to 'Yes', the RVP900 adjusts the calibration reflectivity value Z0 when the current noise level changes from the level measured when the calibration was done. For details, see Section 6.3.3.1 Noise Correction to Reflectivity Calibration on page 217.

4.2.3 Mf — Clutter Filters

```
Default residual clutter LOG noise margins:
  Baseline : 0.15 dB/dB for Clutter/Noise above 10dB
  HiSignal : 1.00 dB/dB for Clutter/Noise above 50dB
```

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 20 dB of total power above receiver noise, and then apply a clutter correction of 19 dB, we are left with an apparent weather signal power of +1 dB above noise. However, the uncertainty of this +1 dB residual signal is much greater than that of a pure weather target at the same +1 dB 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 0.15 dB/dB, the LOG threshold would be increased by 19x0.15 = 2.85 dB. This helps eliminate noisy speckles from the corrected reflectivity data.

```
Spectral Clutter Filters
_____________________________
  Window -1:Default  0:Rectangular 1:Hamming
  Code    2:Blackman 3:ExBlackman  4:VonHann  5:Adaptive

Filter #1 Type:0(Fixed)           Win:1  WidthPts:1  EdgePts:2

Filter #2 Type:0(Fixed)           Win:2  WidthPts:2  EdgePts:2

Filter #3 Type:0(Fixed)           Win:2  WidthPts:3  EdgePts:3

Filter #4 Type:1(Variable)        Win:2  WidthPts:3  EdgePts:2 HuntPts:3
```

```
Filter #5 Type:3 (Gaussian Model) Win:-1 Spectrum width: 0.200 m/sec
Filter #6 Type:3 (Gaussian Model) Win:-1 Spectrum width: 0.300 m/sec
Filter #7 Type:3 (Gaussian Model) Win:-1 Spectrum width: 0.500 m/sec
```

These questions define the clutter filters that operate on power spectra during the DFT-type major modes (PPP, FFT, and RPH). Filter #0 is reserved as "all pass", and cannot be redefined here. For filters #1 through #7, enter a digit to choose the filter type, followed by however many parameters that type requires. For details, see in Section 6.2.5 Clutter Filtering Approaches on page 203.

Fixed Width Filters (Type 0)

These are defined by two additional 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 RVP900 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 Section 6.2.5 Clutter Filtering Approaches on page 203). 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 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. The secondary SQI level is computed by multiplying the primary user-supplied SQI threshold by the SLOPE, and adding the OFFSET. See also Section 6.8.3 Tuning for Optimal Performance on page 247.

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

```
Whitening Parameters for Tx:SZ(8/64)
------------------------------------
Max power mismatch across octants: 4.0 dB
High power rejection threshold: 8.0 dB
Maximum KEY phase error: 12.0 deg
```

These parameters tune the phase whitening process for SZ(8/64) transmissions.

4.2.4 Mt — General Trigger Setups

These questions are accessed by typing Mt with no additional arguments. They configure general properties of the RVP900 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 RVP900, 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 external pretrigger, rather than by one of the RVP900 trigger outputs. In other words, answer "YES" if the transmitter would continue running fine even if the RVP900 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.

```
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 during noise measurement : NO
```

The RVP900 can inhibit the subset of blankable trigger lines whenever a noise measurement is taken. This is 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 RVP900 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 (see Section 6.1.4 Burst Pulse Tracking on page 188), you 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.

```
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 RVP900 range selection will compensate for the additional waveguide length to within plus–or–minus half a bin, and works properly at all range resolutions.

4.2.5 Mt<n>— Triggers for Pulsewidth #n

These questions are accessed by typing **Mt**, with an additional argument giving the pulse width number. They configure specific trigger, transmit waveform, and FIR filter properties for the indicated pulse width 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. Similarly, 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 RVP900 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 RVP900 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 µsec to 5000 µsec Width limits: 0 µsec to 5000 µsec

```
Maximum number of Pulses/Sec: 2000.0
Maximum instantaneous 'PRF' : 2000.0 (/Sec)
```

These are the PRF protection limits for this pulse width.

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 pulse width. Note that the maximum instantaneous PRF can not be set lower than the maximum number of pulses per second.

PRF limits: 50 Hz to 20000 Hz

```
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 RVP900 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 µsec to 1000 µsec

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

The range resolution of the RVP900 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 IFDR Acquisition Clock (see Section 4.2.1 Mc — Top Level Configuration on page 104).

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" appears.

Limits: 10 m to 1000 m (exact limits depend on other Rx setup values)

```
Tx Intermediate Frequency: 30.0000 MHz
Rx Intermediate Frequency: 30.0000 MHz
```

The radar intermediate frequency can be selected separately for each pulse width, so that different width pulses could occupy different frequency bands if desired.

The Tx and Rx intermediate frequencies are allowed to be different from each other, so that the RF up-conversion chain for transmission could be different from the down-conversion chain for reception.

Limits: 6 MHz to 72 MHz

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

The RVP900 computes "I" and "Q" using a digital FIR 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 pulse width. 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 amount 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 toward zero on each end; hence, the power contributed by the overlapping edge samples is minimal. Vaisala recommends beginning with an impulse response length of 1.2–1.5 times the pulse width or bin spacing, whichever is greater.

The maximum FIR 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. The filter length can be as long as 110 ?sec at 125 m resolution in single-polarization mode, and up to 20 ?sec at 16-meter resolution. In dual-polarization mode, the limits are 80 ?sec at 125 m and 20 ?sec at 32 m.

More precisely, in single-polarization mode the maximum filter length (in meters) is 190 times the range bin spacing (also in meters), subject to the added constraints that the number of coefficient taps cannot exceed 8000 total and 62 per range step. For example, if we want a filter that is 3.8 km (25.3 ?sec) long, then the range resolution can be no less than 3.8 km/190 = 20 m. At the RVP900 maximum 100 MHz AQ clock this filter will require (25.3 ?sec)(100 MHz) = 2530 taps to compute, which fits within the 8000-tap limit.

In dual-polarization mode the "190" in the above formula is replaced with "95".

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

This estimator is mostly used with the **Pb** (plotting commands); see Section 5.3.2 Available Subcommands Within Pb on page 145.

```
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 band pass filter. The actual bandwidth of the final FIR filter depends on the filter's impulse response length and the design window used in the process. The actual 3 dB bandwidth is:

- 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 6 dB (not 3 dB) attenuation point. Hence, the 3 dB 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 3 dB bandwidth is shown there, so that it can be compared with the ideal prototype bandwidth.

Limits: 0.05 MHz to 10.0 MHz

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

These are the hardware control bits for this pulse width. 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 levels are assigned by default when the RVP900 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 RVP900 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 RVP900 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 Section 4.2.1 Mc — Top Level Configuration on page 104).

Limits: -500 µsec to 500 µsec

```
Polarization switch point for POLAR1: -1.00 usec
Polarization switch point for POLAR2:  1.00 usec
```

The RVP900 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 the polarization output lines is set by optionally inverting the signals in *softplane.conf*.

Limits: -500 µsec to 500 µsec

4.2.5.1 Special Options for Tx Synthesis

Several of the dialogs described in the previous section are modified when the RVP900 is equipped with an IFDR is configured for Tx waveform synthesis in the **Mz** menu. In this case, each of the RVP900 four pulse widths can select an entirely different type of transmit waveform and associated matched receiver.

For example, PW–0 and PW–1 could transmit conventional 0.5 µsec and 2.0 µsec CW pulses that are received using the bandpass filters described in Section 4.2.5 Mt<n>— Triggers for Pulsewidth #n on page 117. But within this same system, PW–2 and PW–3 could be further configured as, perhaps, 20 µsec and 40 µsec 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 pulse width for each one.

The following questions appears in the **Mt<n>** menu (immediately after the Rx Intermediate Frequency question) when digital Tx waveforms are being synthesized.

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

The RVP900 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 RVP900 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 RVP900 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 RVP900 **Ps** plot) would show an overall spectrum width equal to this desired value.

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

Waveforms are synthesized by the RVP900 using a 16-bit TxDAC followed by an analog bandpass filter centered at the midpoint of the IF interval. The analog filter is necessary to remove out-of-band components from the TxDAC, but has a side effect of introducing a bandwidth limitation within the IF passband. The result is that the shape of very short pulses (on the order of 100 nanoseconds) will be dominated by the impulse response of the analog filter rather than by the exact digital synthesis. In practice the Tx pulse width should be longer than 300 nanoseconds order that the final analog signal be a faithful reproduction of the intended waveform.

```
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 RVP900 convention. This offset question is provided so that the Tx output waveform can be shifted in time to compensate for whatever delays are present in the radar's IF/RF electronics.

```
Hybrid pulse chained PW index
```

If this pulse configuration is for a Hybrid Pulse, then configure that here. In a hybrid pulse, a long compressed pulse is followed immediately by a shorter (usually fixed frequency) pulse. Data from the second shorter pulse is used for near ranges, while the longer pulse is used for farther ranges. Specify here the index number of the second pulse (origin 0). Use -1 to mean no second pulse. You configure the second pulse separately using a different **Mt <n>** command.s.

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 RVP900 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 Section 5.6.2 Available Subcommands Within Pa on page 171.

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 pulse width, 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 pulse width 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 33 on page 170, 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 pulse width 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.

4.2.6 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.

```
Tx Intermediate Frequency: 30.0000 MHz
Rx Intermediate Frequency: 30.0000 MHz
```

These are the centers of the transmit and receive intermediate frequency bands. Although the Tx and Rx intermediate frequencies are generally identical, they are chosen separately so that the RF up-conversion chain for transmission could be different from the down-conversion chain for reception.

The usable Tx and RX bandwidth is delineated by 4MHz safety zones on either side of integer multiples of half the IFDR synthesized clock sampling frequency. The values entered here implicitly define which of those alias bands is being used.

Limits: 6 MHz to 72 MHz

```
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 RVP900.

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

This question controls whether the RVP900 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 IFDR 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 IFDR 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 Section 5.3 Pb — Plot Burst Pulse Timing on page 143; 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 pulse widths.

Limits: -60 dBm to +10 dBm

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

You may choose the window that is used in the design of the FIR matched filter and 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 40 dB 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 fourth 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, 10 KHz. 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 s to 120 s

```
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 long list of questions appears 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. For details, see Section 4.2.6.1 AFC Motor/Integrator Option on page 134.

```
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 s to 300 s

```
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 RVP900 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 RVP900 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 KHz 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 control that is applied to the AFC is specified here in “D-Units”, i.e., arbitrary units ranging from –100 to +100 corresponding to the complete span. Since the D–Unit corresponds in a natural way to a percentage scale, the shorter “%” symbol is sometimes used.

AFC feedback is 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 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 30 KHz results 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 becomes 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 Section 4.2.6.1 AFC Motor/Integrator Option on page 134 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 RVP900 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 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.
- An output protocol is selected for the bit pattern that was produced by encoding the numeric value. The bits may be sent to the IFDR and converted to an analog voltage, or they may be retransmitted by the IFDR as a serial stream to be received by an outboard DAFC module (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 allows 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 RVP900/DAFC module. For example, the default table shown above 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. 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 RVP900/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. The standard RVP900/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.

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

This question enables the Burst Pulse Tracking algorithm that is described in Section 6.1.4 Burst Pulse Tracking on page 188. 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 ±20 µsec; 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 35 MHz, then choosing fifteen subintervals might be a good choice. The hunt procedure would then be able to get within 2.3 MHz 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 RVP900 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.

```
Enable burst power based correction of Z0: NO
```

With this feature enabled, the processor will adjust the value of the calibration reflectivity (Z0) to compensate for long term changes in the transmit power. We do this by assuming that the burst pulse power is proportional to the transmit power. At calibration time, we record the burst pulse power. If this number is available and provided to the RVP900 at processing time, then the processor can notice the difference and adjust the calibration. The burst pulse power must be above the "Minimum power for valid burst pulse" value specified above. The calibration changes are done only at the beginning of a processing mode change, which means once per task in IRIS.

The corrected calibration reflectivity numbers to do this are stored with the time series. To get the correction applied, you need to turn on the OPTS_ZCAL flag in the SOPRM command (see XARG 6 in Section 7.3 Setup Operating Parameters (SOPRM) on page 259).

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

The RVP900 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 RVP900 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 3 dB 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.

4.2.6.1 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 RVP900 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 20 KHz/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.

4.2.7 M+ — Debug Options

A collection of debugging options has been added to the RVP900 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 RVP900 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 RVP900 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

```
Nyquist sign flip of plotted IF samples: NO
```

This question asks whether IF samples should be plotted with Nyquist flipping (multiplication by +1,–1,+1,–1...). If answered YES, then the "Pr" and "Pb" plots will modify their rendering of IF samples. As a reminder within those menus, the text "(NyFlip)" will appear in the plot text heading whenever sample flipping is being applied. Note that other plots within these menus, such as spectra and LOG magnitude, are not affected by the Nyquist flipping, nor are any live parameters derived from the IF samples. Only the visual appearance of the IF samples themselves is affected.

```
Available WSR98D RVP test points:

0:<no-output>    1:RF-Gate          2:RF-Pulse-Start
3:RF-Drive       4:Filament-Reset   5:Post-Charge
6:Mod-Charge     7:Mod-Discharge    8:Trig-Charge
9:Rx-Prot-Cmd    10:Rx-Prot-Rsp    11:Update-In
12:Udate-Time1   13:Ph-Coho-Sel    14:Uplink-Clk
15:Uplink-Data   16:Uplink-Sel
==> TP-1 and TP-2 are presently: (0,0)
```

This question only appears on system configured with the WSR98D control panel. It allows the user to configure which signals are output on the two RVP controlled Test Points on the control panel. A value of 0 means no output. You should set these to zero if the test points are not in use to save driver power. Many of these signals are WSR98D triggers. The "Update" signals are triggers used to control the timing of when the control panel outputs are updated. The "Uplink" signals are signals used on the uplink cable between the IFDR and the control panel.

4.2.8 Mz — Transmissions and Modulations

These questions are used to configure the phase modulation codes that may be used to control the phase of a coherent transmitter. See also Section 4.2.1 Mc — Top Level Configuration on page 104 and Section 4.2.5 Mt<n>— Triggers for Pulsewidth #n on page 117 where related phase control questions are found.

```
Provide phase modulation of transmitted pulses: YES
  Number of binary angle phase bits to use : 8
  Phase angle to apply when idle: 0.00 deg (0x0)
  Modulation - 0:None, 1:Random, 2:Custom, 3:SZ(8/64) : 0
```

Selects whether the Tx waveforms synthesized by the IFDR will have phase modulation applied to them.

```
Chan A - 0:Unused, 1:FixedFreq, 2:TxWaveform : 1
  FreeRunning fixed frequency : 60.00000 MHz
  Output CW power level : 0.0 dBm
```

```
  Apply pulse-to-pulse phase modulation: NO
    Fixed relative phase offset : 0.000 Deg
```

Selects the type of waveform to generate on the Tx-A SMA output. In this example a 60MHz fixed frequency CW sinewave is generated at 0.0 dBm power level. The CW wave can optionally be phase modulated from pulse to pulse, and can be offset by a fixed phase amount.

```
Chan B - 0:Unused, 1:FixedFreq, 2:TxWaveform : 2
  Output power level: 0.0 dBm, Peak:YES
  Apply pulse-to-pulse phase modulation: NO
```

Selects the type of waveform to generate on the Tx-B SMA output. Here, a pulsed transmit waveform will be generated having a peak output level of 0.0dBm. Phase modulation can optionally be applied. The details of the pulse itself are defined in the Mt<n> menu for each pulse width (see Section 4.2.5 Mt<n>— Triggers for Pulsewidth #n on page 117).

```
FM Chirp manual spectrum flattener: 0.00 %-per-MHz
```

Transmit waveforms generated by the IFDR are corrected for inherent sin(f)/f amplitude-vs-frequency response of the TxDAC, resulting in a flatter FM chirp waveform. In addition, this setup question lets you supply adhoc frequency band compensation. The units of this parameter are: percentage amplitude compensation per megaHertz of deviation from the IF center frequency. Positive values result in a relative boost of higher frequencies across the time span of the waveform. When the overall frequency response of your Tx/Rx is not flat, you can tune this parameter to null out the error based on feedback from either the Ps spectrum plot or oscilloscope display of the Tx pulse.

CHAPTER 5
PLOT-ASSISTED SETUPS

The IFDR 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 RVP900 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 RVP900 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. The RVP900 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 carry out the graphical checkup and alignment procedures remotely through a network.

5.1 P+ — Plot Test Pattern

The RVP900 can produce a simple test pattern to verify that the display software is working properly. From the TTY monitor enter the **P+** command. This prints the message **Plotting Test Pattern...** on the TTY and then produce the plot shown in Figure 25 on page 140

This display is actually an overlay of six different strokes:

- Bottom line
- Middle line
- Top line
- Line sloping up
- Line sloping down
- 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.

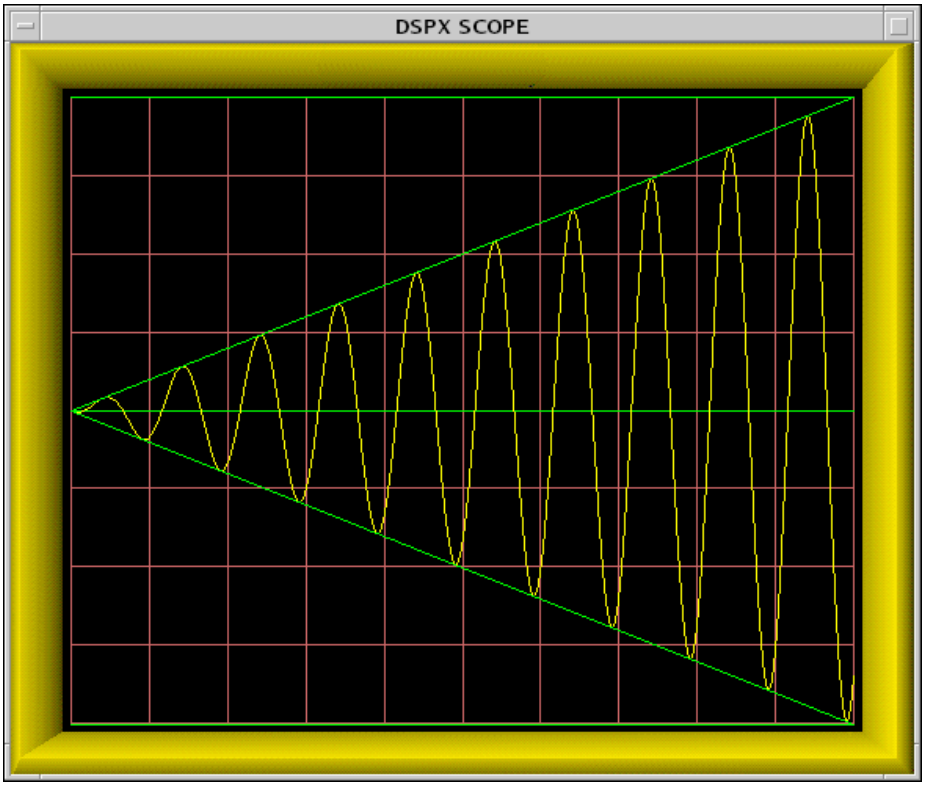


Figure 25 The Test Pattern Display

When you are satisfied that the plot is being drawn correctly, type **Q** or press **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 RVP900 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 report "No Trigger" on the TTY status line whenever the external trigger is expected, but missing.

The TTY screen scrolls upward each time a new subcommand is executed, so that a history of information lines and command activity are 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 RVP900 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** widens 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..."** is printed. Subsequently, typing another "**.**" single steps to the next data update, but the plot and printout still remain frozen.

Typing **Q** or **ESC** exits 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 where the only purpose is to alter the appearance of the display, for example, zoom, stretch, etc. These subcommands make no changes to the actual working RVP900 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 are 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 RVP900 digital front end. You may 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 pulse width that the radar supports.

1. Use **Mb** to set the systems intermediate frequency (see Section 4.2.6 Mb — Burst Pulse and AFC on page 126).
2. Use **Mt** to choose the PRF and pulse width (see Section 4.2.4 Mt — General Trigger Setups on page 114). 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 RVP900 triggers that are used by 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 Section 4.2.5 Mt<n>— Triggers for Pulsewidth #n on page 117). Make an initial guess of FIR filter length as 1.5 times the pulse width.
4. Use **Pb** to slew the start times of all triggers so that the burst pulse is properly sampled (see Section 5.3 Pb — Plot Burst Pulse Timing on page 143). 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 Section 5.4 Ps — Plot Burst Spectra and AFC on page 148). 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 pulse width should also work for all others.
7. Use **Pr** to verify that targets are being detected with good sensitivity (see Section 5.5 Pr — Plot Receiver Waveforms on page 163).

Sometimes it is useful to run the **Pb** and **Ps** commands with samples from the IF-Input of the IFDR, rather than from the Burst-Input. Similarly, it is sometimes useful to view the **Pr** plots on samples of Burst data. Within all three plotting commands the “%” key can be used to toggle among all five of the IF input SMA connectors on the IFDR.

5.3 Pb — Plot Burst Pulse Timing

For magnetron radars, the RVP900 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 ultimately resembles Figure 26 on page 143, 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.

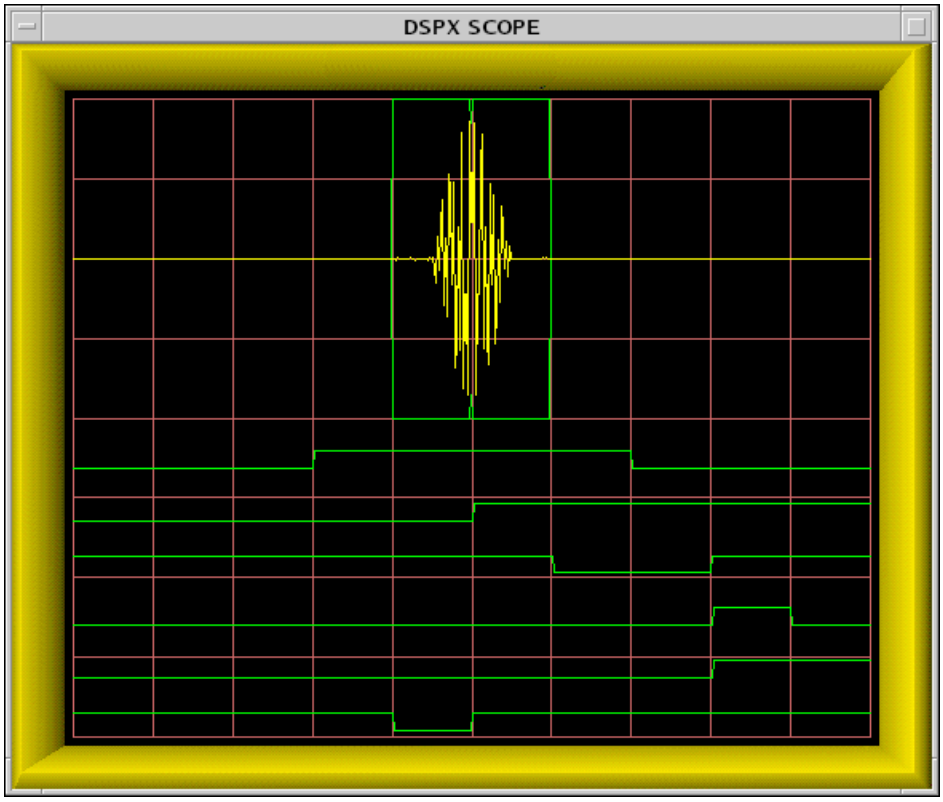


Figure 26 Successful Capture of the Transmit Burst

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 IFDR burst input are plotted (somewhat brighter) within the sample window.

The RVP900 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 triggers that are output by the RVP900. The number of triggers plotted match the number of user-defined output triggers set in the **Mt** menu, with Trigger #1 being at the top. They are drawn in their correct polarity and timing relative to each other, and relative to the burst sample window. The sample window is always drawn in the center of the overall time span. Depending on the PlotSpan and location of the trigger edges, triggers that do not vary within the plotted time span appear as flat lines.

The RVP900 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 is 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 left/right
T/t          Plot time span Up/Dn
Z/z          Amplitude zoom
> and <      Start/Stop logging to rvp9-pb.log
%            Toggle between IF input channels
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, the full FIR window is used; however, shortening the AFC interval results 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 RVP900 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) varies, and this causes 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".

> and < The live plotted data can be logged to the `$(IRIS_LOG)/rvp9-pb.log` text file; one line per plot.

% Toggle between all five IFDR IF-Input sources.

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 are found, then the previous trigger slew and AFC are not changed.

5.3.3 TTY Information Lines Within Pb

The TTY information lines resemble:

```
Zoom:x2, PlotSpan:5 usec, FIR:1.36 usec (49 Taps)
IF:Ch5(TXB) 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 is not 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 fourth order correlation model. The control parameters for this model are set using the **Mb** command (Section 4.2.6 Mb — Burst Pulse and AFC on page 126).

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 RVP900 trigger outputs. This number varies as the **L** and **R** subcommands cause the triggers to slew left and right. If the radar transmitter is directly fired by an external pretrigger, the pretrigger delay (in the form "**PreDly:6.87**") is printed instead.

BPT This shows the present value of timing slew (measured in microseconds) being applied to track the burst. The slew is zero initially when the RVP900 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 pulse width. Repeat the following procedure for each pulse width that you plan to use. Each iteration is independent.

It is first necessary to setup the proper relative timing for all RVP900 triggers that are being used. The various trigger output lines are completely interchangeable, and each may be assigned to any function within the radar system. For example, Trigger #1 might be the transmitter pretrigger, Triggers #3 and #4 might synchronize external displays, and Triggers #2, #5, and #6 might be unused.

Choose an initial impulse response length of 1.5 times the transmit pulse width. This length is refined later when the matched filter is designed (See Section 5.4 Ps — Plot Burst Spectra and AFC on page 148). 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 is still 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 IFDR A/D converter. The maximum analog signal level is +8 dBm. 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. Vaisala recommends a peak signal level between -3 dBm and +4 dBm, 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 helps decide the minimum energy threshold for a valid burst pulse, which is needed in Section 4.2.6 Mb — Burst Pulse and AFC on page 126.

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 band pass 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 RVP900 eliminates all of these restrictions via its **Ps** command, which plots the burst spectrum, designs the band pass 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 27 on page 149. 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.

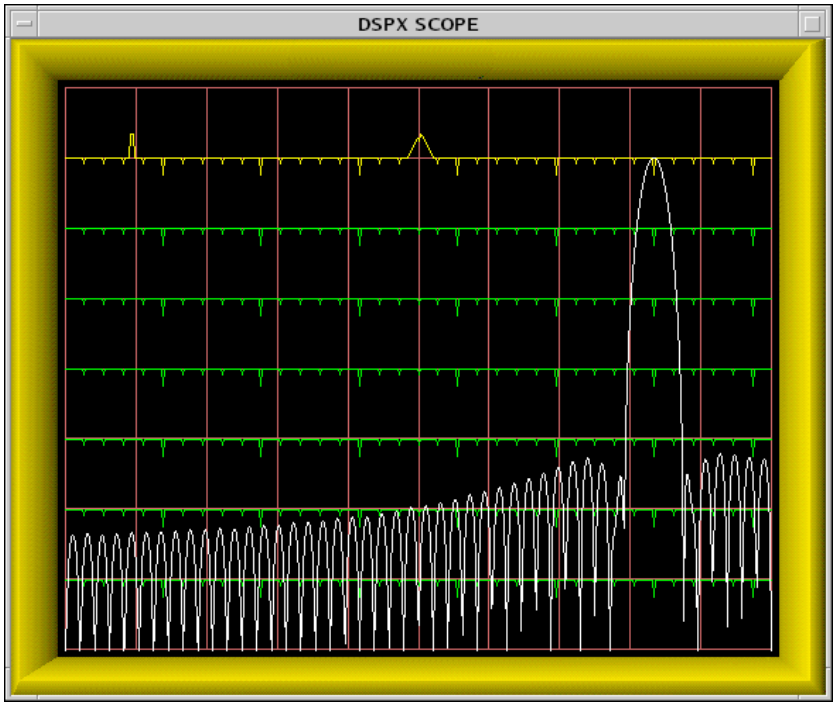


Figure 27 Example of a Filter With Excellent DC Rejection

The horizontal axis of the spectrum plot represents frequency. The overall span from the left edge to the right edge is half the acquisition system clock frequency selected in the **Mc** menu.

The exact endpoints of the plot depend on which alias band the radar's intermediate frequency falls in. For example, with a 72 MHz acquisition clock, a 30 MHz IF would imply a horizontal axis range of DC to 36 MHz, whereas a 60 MHz IF would make the range 36 MHz to 72 MHz. 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 36 MHz, either the left side or the right side will always be a multiple of 36 MHz. This is important to remember when designing the matched filter, since fixed DC offsets in the A/D converters appear aliased at these 72 MHz 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 5 MHz; minor marks are in between and represent 1 MHz steps. The power spectrum example in Figure 27 on page 149 is from a system 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, 30 MHz.

Two types of spectra can be plotted on the screen: the frequency response of the FIR filter and 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 27 on page 149 is an example of a single filter response plot, whereas Figure 28 on page 160 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 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 1 KHz steps, and the upper case commands use 100 KHz 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 RVP900/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 Section 6.1.1 FIR (Matched) Filter on page 183 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 RVP900 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 70 dB 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 Section 5.3.3 TTY Information Lines Within Pb on page 146

BW Indicates the actual 3 dB 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.

Pwr Indicates the average power in the full burst sample window.

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 Section 5.4.4 Computation of Filter Loss on page 156.

AFC Indicates the level and status of the AFC voltage at the IFDR 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 are printed to the right of the AFC level to show the encoded bit pattern which corresponds to that level. This only appears when the RVP900 deduces that a special digital format is being used, that is, 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 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. The AFC loop remains idle.

(Wait) The burst pulse has become valid just recently, but the AFC loop is idle until the transmitter stabilizes.

(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 RVP900 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 RVP900 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 dB to 1.8 dB range, depending on the FIR length and other design criteria.

As an example, consider how the RVP900 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 band pass 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 band pass 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 0.44 dB, 1.11 dB, and 3.31 dB 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 RVP900 as follows:

- Using the **Mt0** command, setup a T_o = 0.5 µsec trigger pulse from the RVP900 in the vicinity of range zero, and use that trigger to gate a signal generator whose output is applied to the IFRD Burst Input. Also setup 125 m 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 band pass 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 4 MHz wide and centered at the radars IF. There should also be weaker lobes spaced 2 MHz

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 4 MHz, 2 MHz, and 1 MHz wide at their 3 dB points. You should see filter losses reported that are very close to the theoretical values for the ideal band pass filter.

In the above analysis we have assumed that S(*f*) is the idealized power spectrum of a continuous time signal. Of course, the RVP900 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 RVP900 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 RVP900 IF filter, and the integrals are performed over the Nyquist frequency band that is implied by the IFDR 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 0 dB whenever the Tx waveform appears to have zero spectral width, that is, 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 RVP900 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 RVP900s 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 band pass 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 3 dB pad would seem to produce a 3 dB 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 increases 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 24 MHz and 30 MHz, then the filter losses for each channel would use the frequency intervals 21 MHz to 27MHz and 27 MHz to 33 MHz, 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 IFDR.

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.

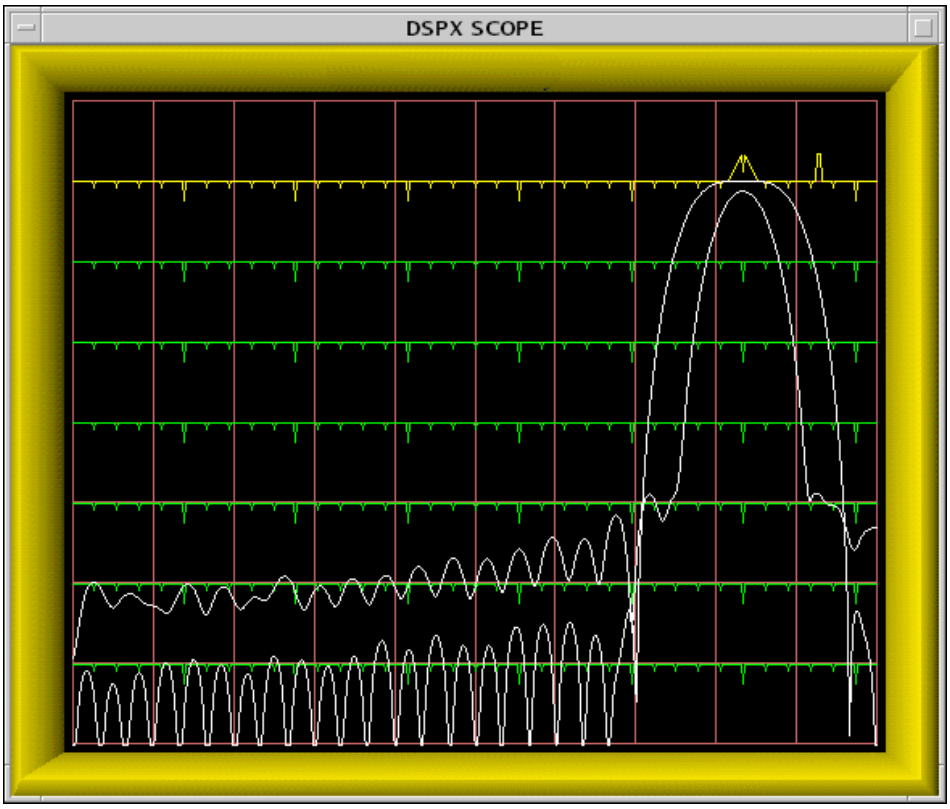


Figure 28 Example of a Poorly Matched Filter

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 pulse width. 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 -65 dB (see discussion below).
- If there are conspicuous interference spikes at particular frequencies, try to adjust the location of the filter's 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 28 on page 160 shows an example of a filter that is poorly matched to the pulse. 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 requires a few minutes of hunting with the I, W, and N keys. Every time you press any of these keys the RVP900 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 RVP900 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**—Type the **$** command and let the RVP900 do all of the work (See description in Section 5.4.2 Available Subcommands Within Ps on page 151).

The offset error of the IFDR A/D converter is at most 10mV, that is, -27 dBm into its 50? input. If we wish to achieve 90 dB of dynamic range below the converters +8dBm saturation level, then we expect usable "I" and "Q" values to be obtainable from a (sub-LSB) input signal at -82 dBm. This is 55 dB below the interference that would result from the worst-case A/D offset. But a weak input signal at -82 dBm would still be damaged by

even an equal level of DC interference. Therefore, adding another 10 dB safety margin, we get -65 dB 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 -27 dBm worse-case A/D offset.

Figure 29 on page 162 shows a 60 MHz filter with particularly poor (-42 dB) DC rejection. The frequency range of the plot is 36 MHz to 72 MHz; therefore, 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 27 on page 149 that has a true 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.

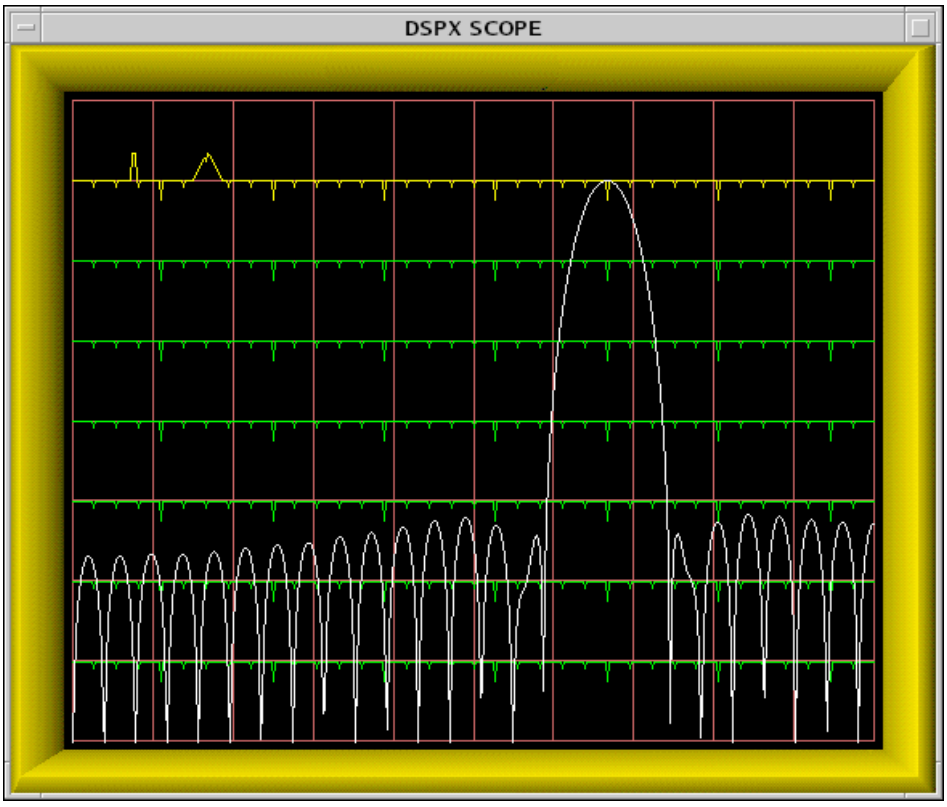


Figure 29 Example of a Filter With Poor DC Rejection

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 IFDR 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 30 on page 163. 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 plotted both as raw IF samples and as the LOG of the detected power using the FIR filter for the current pulse width.

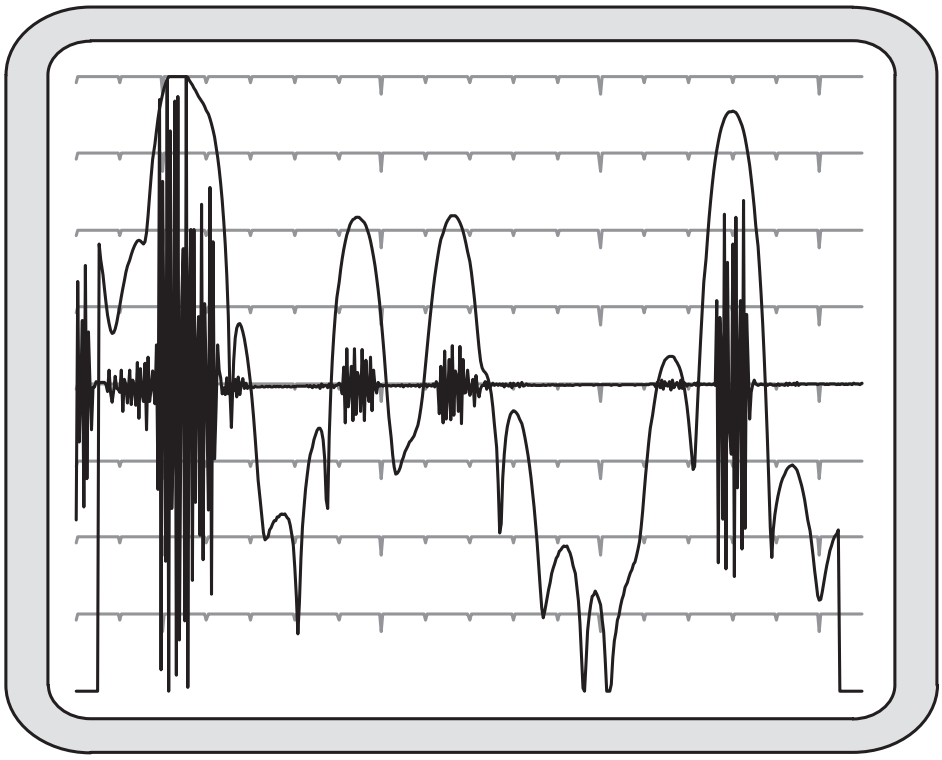


Figure 30 Example of Combined IF Sample and LOG Plot[6]

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 10 mv would appear as a several-hundred-count offset in the 16-bit A/D range. 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 RVP900 preserves the DC term in the **Pr** sample plot so that its presence is not forgotten.

The "AC" amplitude of the IF samples increases wherever targets are present. On top of these samples is drawn the detected power on a logarithmic scale. Each horizontal line represents a 10 dB 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 30 on page 163 as an example, the two equal-power targets just to the left of center are approximately 18 dB 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 6 dB 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 72 MHz sampling rate the spacing between LOG samples are a mere 4.17 m. 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 31 on page 165. 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 10 dB steps. The entire vertical axis is used so that an

overall span of 80 dB is visible. This particular plot was made with the time span set to 50 µsec, and with a 1 m 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.

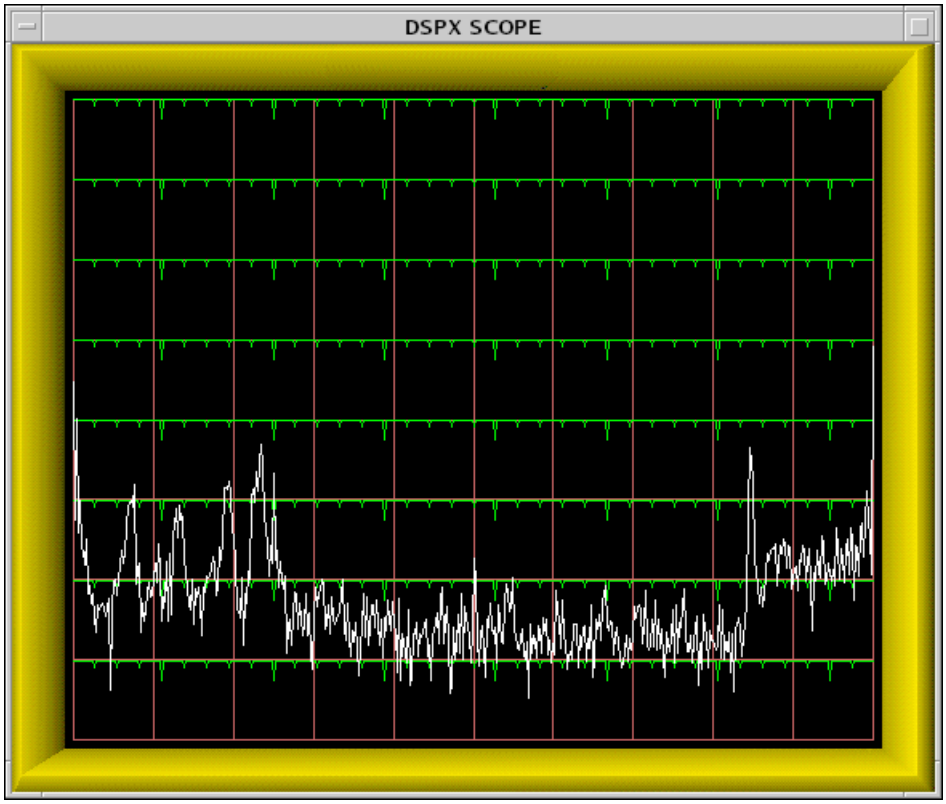


Figure 31 Example of a Noisy High Resolution Pr Spectrum

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 RVP900 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
> and <   Start/Stop logging to rvp9-pr.log
%         Toggle between IF input channels
-         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 6 dB 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: Received Samples, Received Samples and LOG Power, and Received Power Spectrum.

> and < The live plotted data can be logged to the `$(IRIS_LOG)/rvp9-pr.log` text file, one line per plot.

% Toggle between all five IFDR IF-Input sources.

5.5.3 TTY Information Lines Within Pr

The TTY information lines 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.

IF Indicates which of the five IF-Input sources (SMA edge connectors) is providing the data being plotted.

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 pulse width. 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 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

The RVP900 is able 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 ?sec to 80 ?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 RVP900 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 pulse width options, and examine and optimize the range/time sidelobes of your waveform.

5.6.1 Interpreting the Ambiguity Plots

Figure 32 on page 169 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 pulse width is 30 ?sec, bandwidth is 3 MHz, PSL is -61.2 dB and ISL is -50.8 dB, Doppler shift ± 50 KHz.

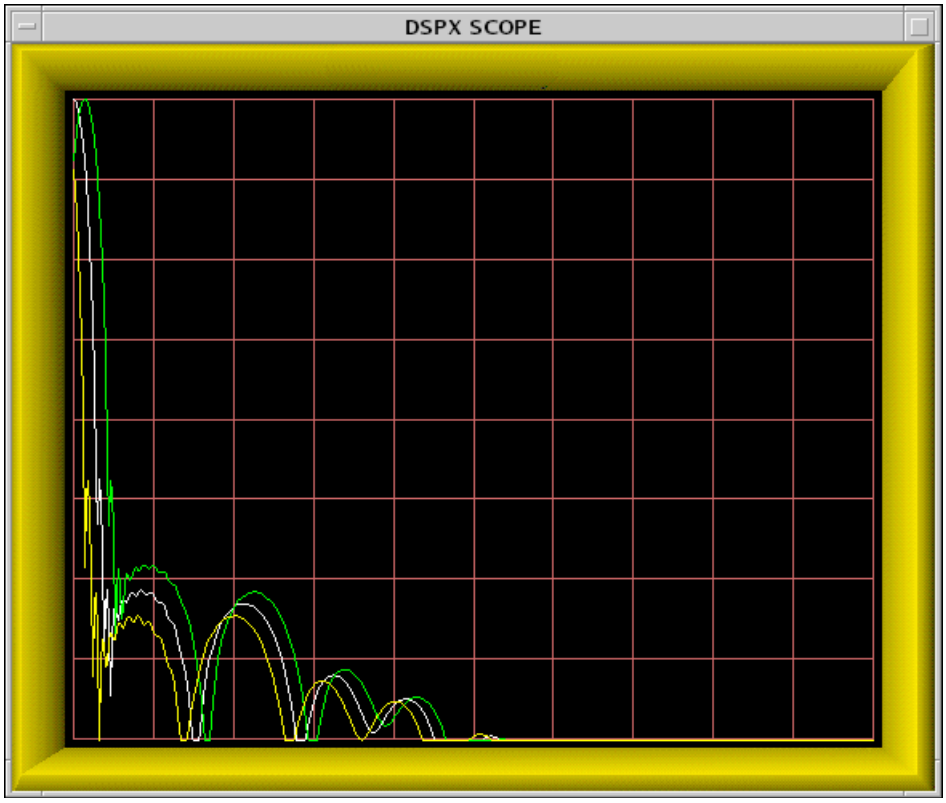


Figure 32 Ambiguity Diagram of a Compressed Tx Pulse

Also shown in yellow and green are the Tx/Rx responses when the overall waveform is modified by a 50 KHz 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 33 on page 170. 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.

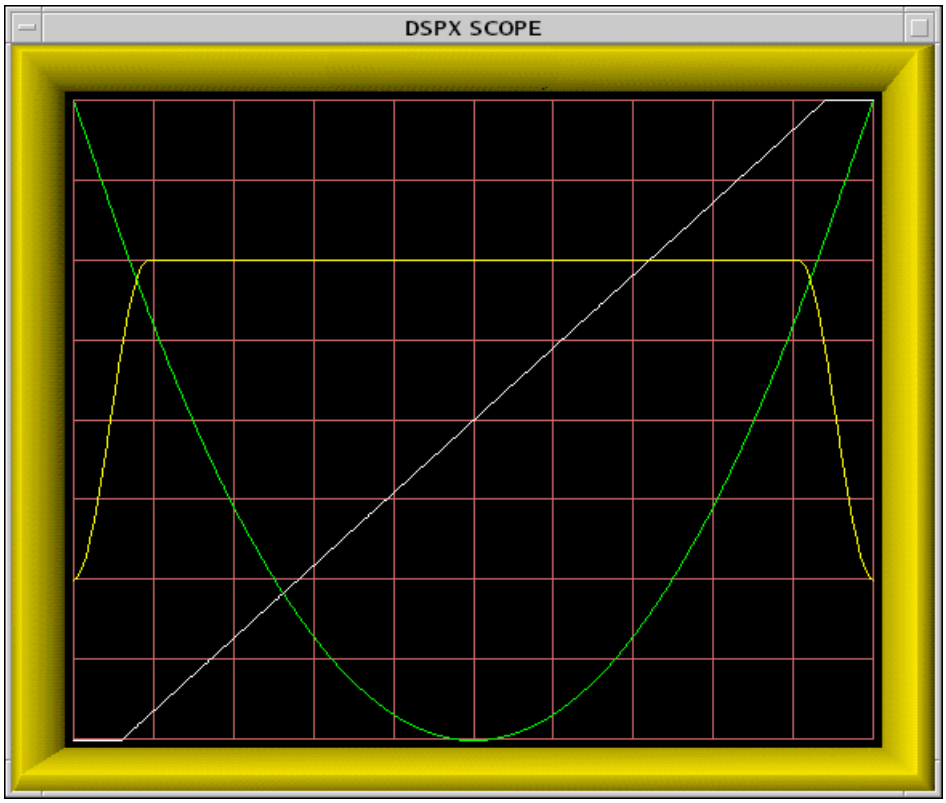


Figure 33 Frequency, Phase and Amplitude of a Compressed Tx Pulse

From Figure 33 on page 170 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 Special Options for Tx Synthesis on page 123 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 RVP900 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 RVP900 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 IFDR 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 RVP900/Rx card is able to deconvolve the compressed information.

To setup the test, simply connect the Channel #1 or Channel #2 output of the RVP900/Tx card to the IF-Input of the IFDR. Use whichever

RVP900/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.

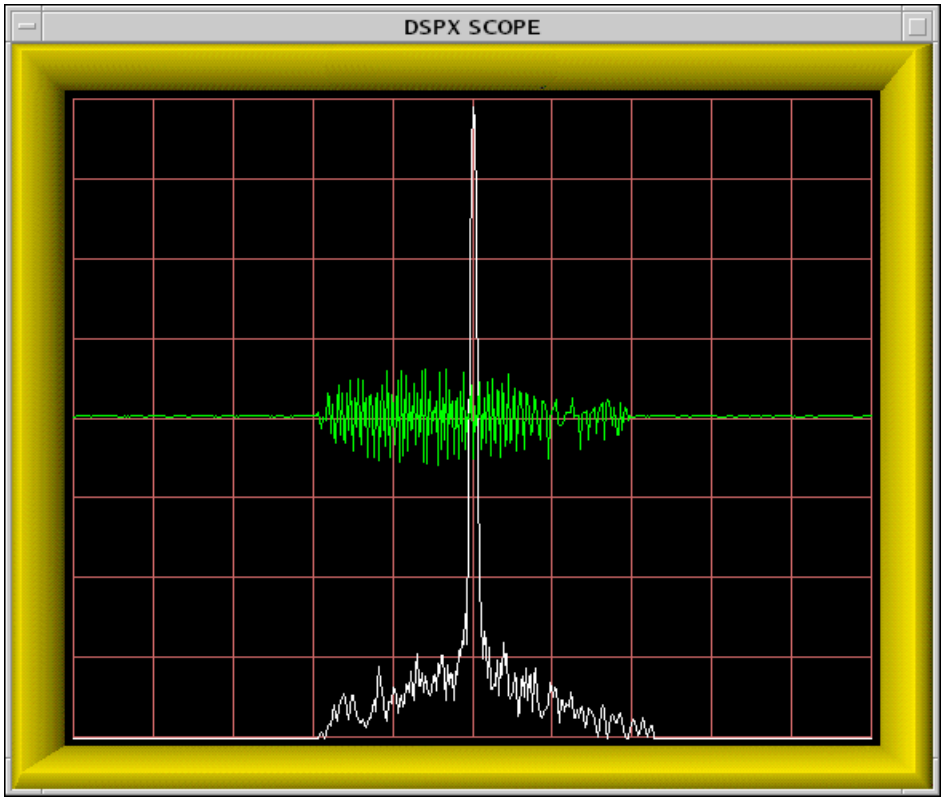


Figure 34 IFDR Sampling of Optimized Compressed Tx Waveform

Figure 34 on page 176 shows an actual **Pr** plot of a 40µ sec, 5MHz optimized waveform generated by the RVP900/Tx card and fed into the IFDR. 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 band pass filters on the RVP900/Tx and IFDR. 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 35 on page 177 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.

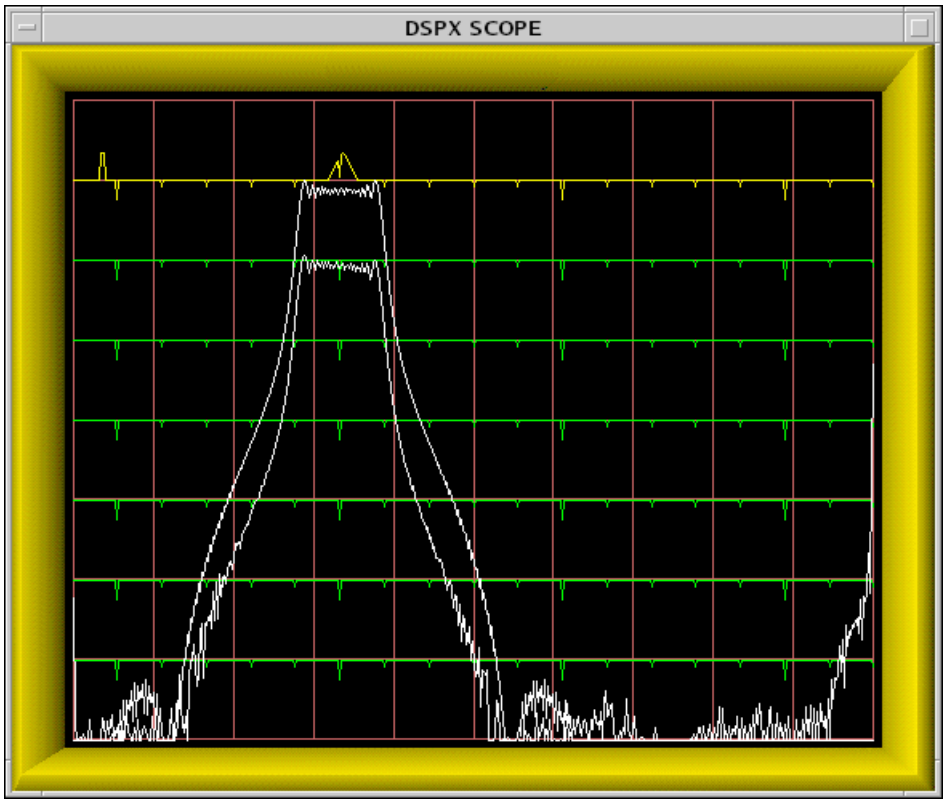


Figure 35 Ideal and Actual Linear-FM Spectrum Displayed in Ps Plot

CHAPTER 6
PROCESSING ALGORITHMS

NOTE Optional dual polarization processing algorithms are described in *IRIS and RDA Dual Polarization User's Manual*.

This chapter describes the processing algorithms implemented within the RVP900 signal processor. The discussion is confined to the mathematical description of these algorithms. Figure 36 on page 182 shows the overall process by which the RVP900 converts the IF signal into corrected reflectivity, velocity, and width. Table 9 on page 181 summarizes the quantities that are measured and computed by the RVP900. 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 RVP900 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 - ? and + ? 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 9 Algebraic Quantities Within the RVP900 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 36 on page 182, 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

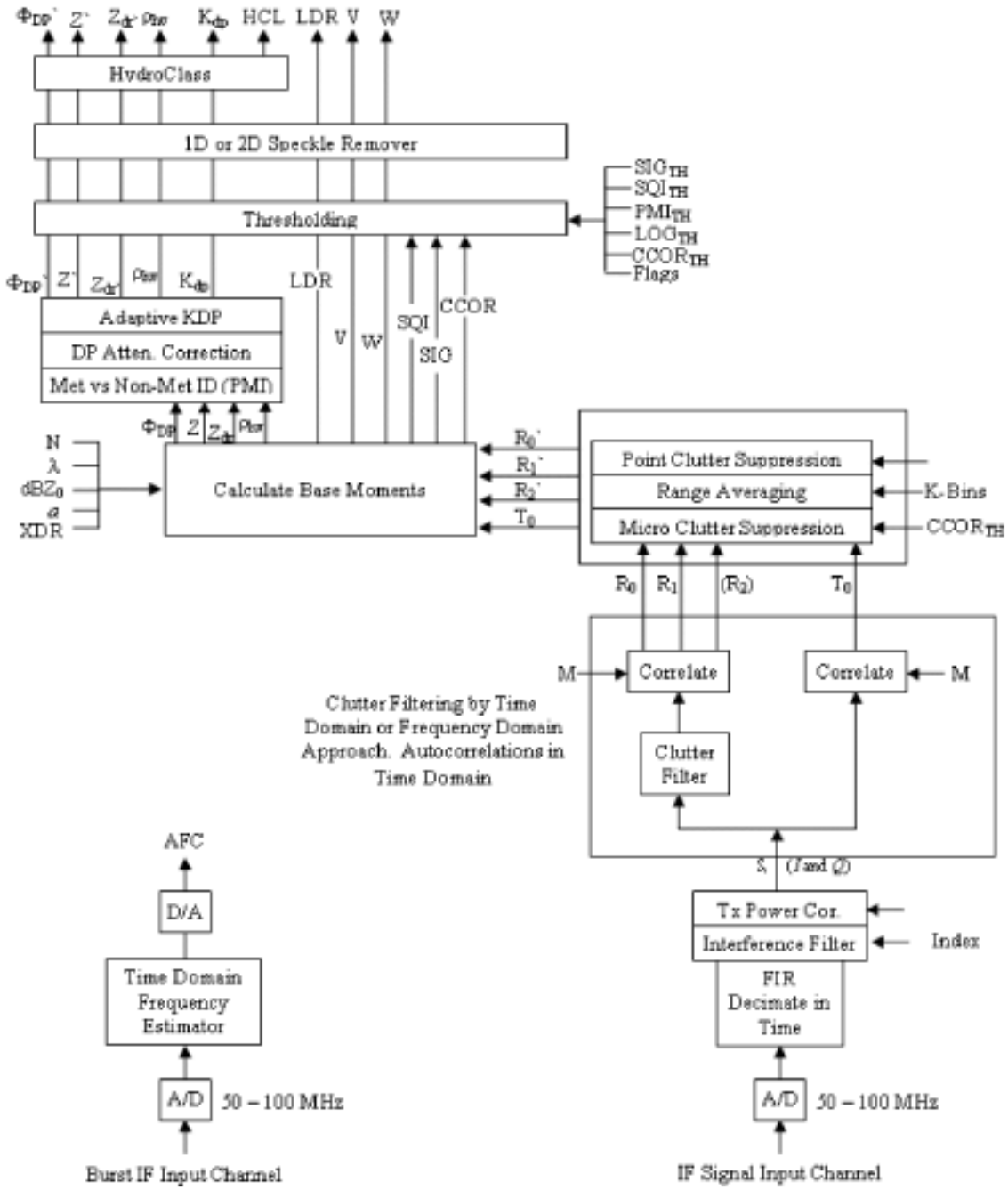


Figure 36 Flow Diagram of RVP900 Processing

6.1 IF Signal Processing

The starting point for all computations within the RVP900 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 36 MHz), which makes possible the digital implementation of functions that are traditionally performed by discrete components in an analog receiver. The RVP900 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 13 on page 39 for a block diagram of the IF processing that is performed.

6.1.1 FIR (Matched) Filter

The RVP900 implements a digital version of the "matched" filter that is found in the traditional analog radar receiver. The equivalent FIR filter is designed using an interactive graphical procedure described in Section 5.4 Ps — Plot Burst Spectra and AFC on page 148. 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 RVP900/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 RVP900/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 p_n. 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 IFDR digital transmitter board), then the samples should be from the modulated COHO.

The f^{i}_{n} 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 IFDR 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:

$$f^{i}_{(N-1)-n} = l_{(N-1)-n} \times \sin\left[\frac{\pi}{4} + 2\pi\frac{f_{IF}}{f_{SAMP}}(((N-1)-n) - \frac{N-1}{2})\right]$$

$$f^{i}_{(N-1)-n} = l_n \times \cos\left[\frac{\pi}{4} + 2\pi\frac{f_{IF}}{f_{SAMP}}\left(n - \frac{N-1}{2}\right)\right]$$

$$f^{i}_{(N-1)-n} = f^{q}_{n}$$

Thus, the coefficients needed to compute I are merely the reversal of the coefficients needed to compute Q; if you know f^{i}_{n}, then you also know f^{q}_{n}.

6.1.2 RVP900 Receiver Modes

The RVP900 supports several receiver configurations which allow you to choose the best style of IF processing for your particular site. The following table summarizes the options from the **Mc** menu:

```
# Rx Mode Description
- ----------------------
0 Standard single channel
1 --Reserved--
2 Legacy RVP8/2005 WDN compatibility
3 Standard dual channel
```

- **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 ?95 dB. The (I,Q) data are computed from IF samples.
- **Mode-1: Reserved**—This mode is currently reserved for future development, as they become necessary.
- **Mode-2: 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) IFDR inputs. A nominal channel separation of 25 dB to 30 dB might be used to achieve an overall dynamic range of up to 110 dB.
- **Mode-3: Dual-Rx From Two IF Inputs**—This is the standard dual polarization mode in which the “H” and “V” channels are fed into two separate IF inputs, both using the same intermediate frequency.

6.1.2.1 Discussion of Wide Dynamic Range Mode-2

When the RVP900 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 IFDR.

The first of these is the amount of signal level separation between the high gain and the low gain IFDR inputs. There is an absolute minimum and absolute maximum channel separation that still allows the IFDR 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 is 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 IFDR. Generally, the total dynamic range of the receiver is set by the LNA. For example, if we are considering a 1µ sec pulse (1 MHz bandwidth), the dynamic range of the LNA may be

about 105 dB, and the dynamic range of a single channel of the IFDR is about 84 dB (from -78 dBm to +6 dBm). In this case, the minimum separation would be 21 dB. 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 IFDR minus the separation. In this case, the overlap is (84 dB to 21 dB) = 63 dB.

- *The absolute maximum separation* of the channels is the dynamic range of a single channel of the IFDR. In the above example this would be 84 dB. 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 trade-off of several factors. If the separation value is too low, the IFDRs 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 IFDR 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 dB to 30 dB range provide a good balance of the above criteria. In the case of a 1 μsec pulse this results in an overlap interval of approximately 55 dB to 63 dB, which is sufficient for good IFDR 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 (105 dB 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 IFDR. If the LNA has a noise figure of 1dB and we are using a 1 μsec pulse, the minimum detectable signal at the LNA input is -113 dBm, and thus the maximum signal is 105 dB above this, or -8 dBm. 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 17 dB, the mixer has a conversion loss of 7 dB, there is 1 dB miscellaneous losses and 3 dB loss in the power splitter, then the signal level at the output of the power splitter is (-113 + 18 - 7 - 1 - 3) = -106 dBm for the minimum signal, and -1 dBm for the maximum signal. In the low gain channel, we need to bring the -1 dBm up to the maximum input value of the IFDR (+ 6 dBm). To do this we need about 8 dB of amplification (7 dB plus one more deciBel to account for the anti-alias filter loss of the IFDR). If we assume 25 dB of channel separation, on the high gain channel we require about +33 dB of amplification. Finally, this tells us that on the low gain channel, the minimum and maximum signals presented to the IFDR are (-106 + 8) = -98 dBm and (-1 + 8) = 7 dBm. For the high gain channel, the signal levels are (-106 + 33) = -73 dBm and (-1 + 33) = +32 dBm. As +32 dBm is above the maximum input level tolerated by the IFDR, the amplifier on the high gain channel must limit its output to less than +16 dBm. Thus an amplifier with an output saturation value of between +10 dBm and +15 dBm 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 MHz or 60 MHz away from the magnetron frequency. The maximum tuning range of the AFC feedback is approximately 7 MHz 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 IFDR. It is important that the system's IF frequency is at least 4 MHz away from any multiple of half the digital sampling frequency, that is, 18 MHz, 36 MHz, 54 MHz, or 72 MHz.

The RVP900 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 RVP900 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, 18 MHz to 36 MHz), 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* (see Section 4.2.6 Mb — Burst Pulse and AFC on page 126). This prevents the introduction of transients into the burst analysis algorithms each time the pulse width changes.

Additional information about using AFC can be found in Section 3.4 Digital AFC Module (DAFC) on page 84 and Section 4.2.6 Mb — Burst Pulse and AFC on page 126.

6.1.4 Burst Pulse Tracking

The RVP900 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 RVP900 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 RVP900 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 2D search in time and frequency to locate the burst; searching across a ±20 µsec 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 RVP900 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 (Section 7.26 Hunt for Burst Pulse (BPHUNT) on page 326) 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} , and P_n

where:

$$P_n = 10\log_{10}(I_n^2 + Q_n^2)$$

If the three pulse powers have the property that:

$$|P_{n-1} - P_{n-2}| < C_1 \text{ and } |P_n - P_{n-1}| > C_2$$

(Alg.1[5])

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 RVP900 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 (Section 7.23 Configure Interference Filter (CFGINTF) on page 324) 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 (Section 4.2.2 Mp — Processing Options on page 106).

$$|P_{n-1} - P_{n-2}| < C_1 \text{ and } P_n - P_{n-1} > C_2$$

(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 = +10 dB, 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 10 on page 191, 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 2 dB. 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 10 Algorithm Results for +16 dB Interference

C1,C2	Alg.1 Missed/False		Alg.2 Missed/False		Alg.3 Missed/False		Alg.3, C1+=2 dB 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%

A false alarm in actual precipitation echo affects the values of moments calculated at the gate of the false alarm. For example, the measured power and estimates of reflectivity, subsequently, become slightly lower. Table 11 on page 191 maps the rates of false alarms in Table 10 on page 191 into mean relative changes of reflectivity (in dB).

Table 11 Impact of False Alarms on Reflectivity Estimates

Alg.1 False	Bias (dB)	Alg.2 False	Bias (dB)	Alg.3 False	Bias (dB)	Alg.3, +2 dB False	Bias (dB)
10.91%	..	4.06%	..	3.48%	..	4.15%	..
6.57%	..	2.42%	..	1.71%		1.92%	..
5.09%	..	1.81%	..	1.16%	..	1.28%	..
4.01%	..	1.42%	..	0.79%	..	0.85%	..
3.14%	..	1.06%	..	0.51%	..	0.54%	..
2.53%	..	0.85%	..	0.33%	..	0.35%	..
2.07%	..	0.67%	..	0.22%	..	0.23%	..
1.70%	..	0.54%	..	0.14%	..	0.15%	..

Table 11 Impact of False Alarms on Reflectivity Estimates

1.21%	..	0.35%	..	0.06%	..	0.06%	..
0.65%	..	0.14%	..	0.01%	..	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 10 on page 191:

- 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 16 dB stronger than the weather, there was a close trade-off between the "Missed" and "False" error rates.

However, Table 12 on page 192 shows the results that would be obtained if the interference dominates by 26 db.

Table 12 Algorithm Results for +26 dB Interference

C1,C2	Alg.1 Missed/False		Alg.2 Missed/False		Alg.3 Missed/False		Alg.3, C2+=5 dB 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%

Table 12 Algorithm Results for +26 dB Interference (Continued)

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 RVP900 is able to recover the signal power of targets that saturate the IF-Input A/D converter by as much as 4 deciBels to 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 37 on page 194 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 +10 dBm versus +5 dBm, and thus has been extended by 5 dB.

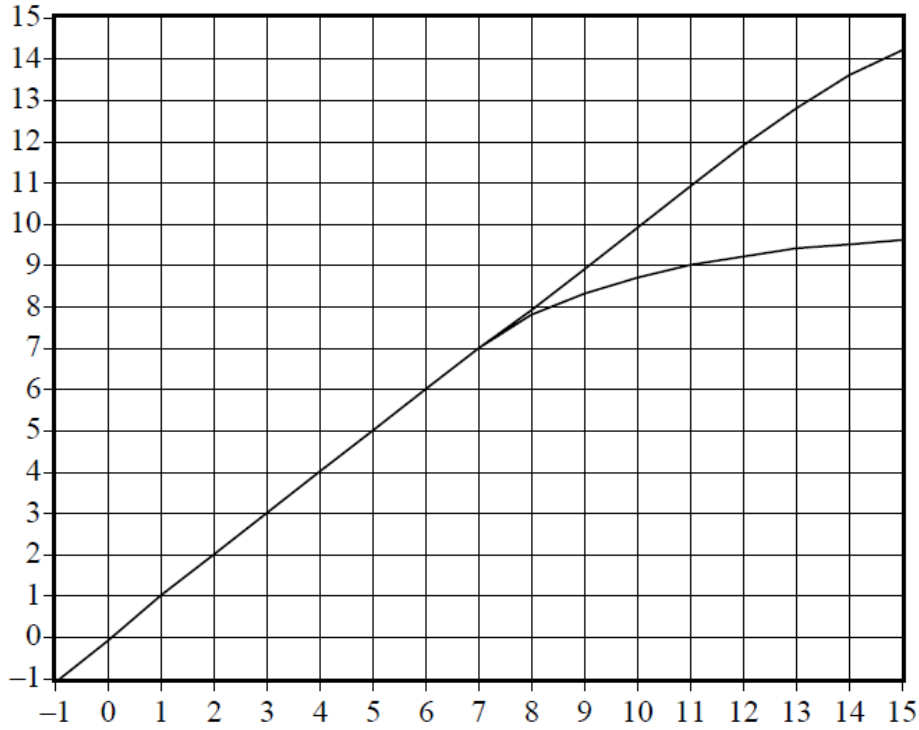


Figure 37 Linearization of Saturated Signals Above + 8 dBm

6.1.7 Correction for Tx Power Fluctuations

The RVP900 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 Section 4.2.2 Mp — Processing Options on page 106). 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 gives excellent results in almost all cases.

Whenever the RVP900 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 RVP900 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 RVP900 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 ±5 dB. If the burst power in a given pulse is more than 5 dB above the mean, or less than 5 dB 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 0.3 dB).

Instantaneous amplitude correction is a unique feature of the RVP900 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 0.02 dB 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:

$$s_m = [I_m + jQ_m] \text{ for } m = 1, 2, 3, \ldots, M$$

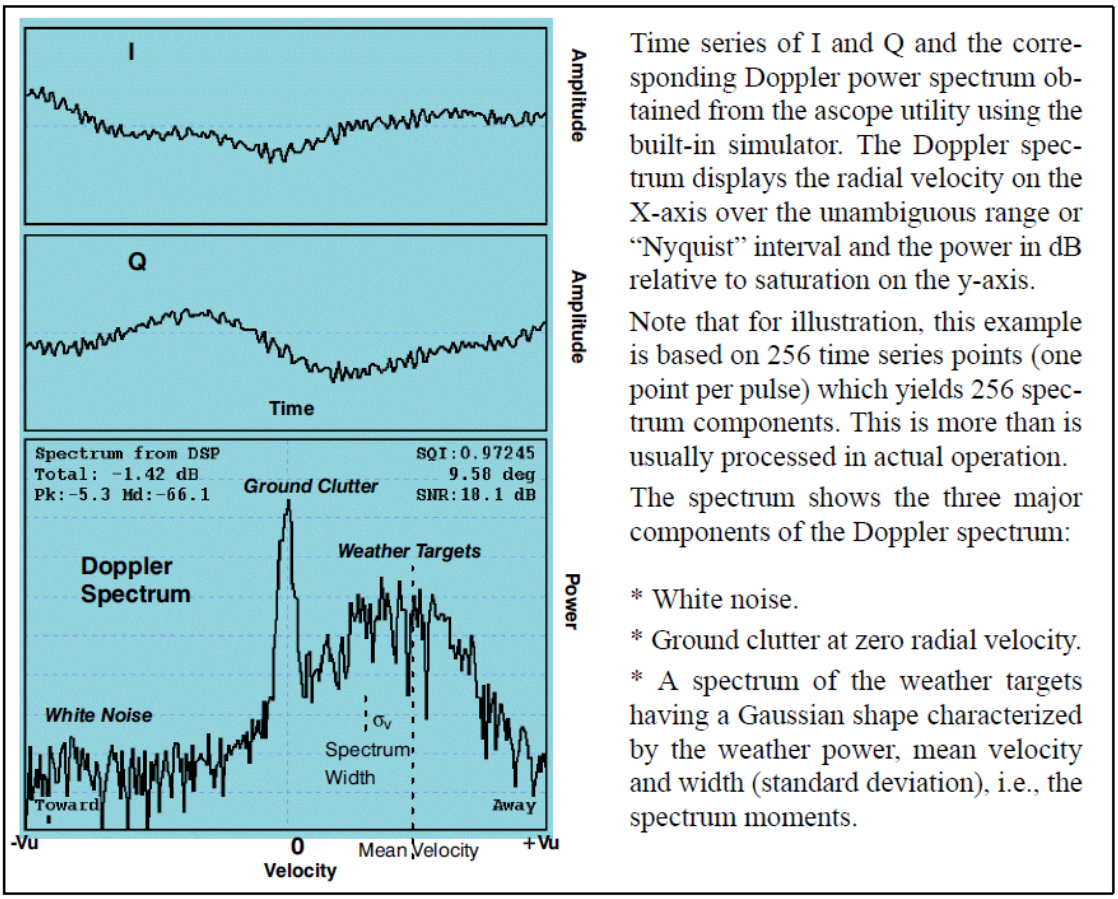
where "j" is - $1^{1/2}$. The time series, are the starting point for all calculations performed within the RVP900. There are several excellent references on the details of I and Q processing.2 The top part of Figure 38 on page 197 shows I and Q values for a simulated time series using the ascope utility.

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 second 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 RVP900 supports the concept of "major modes" or processing modes to process the time series. Currently the following major modes are supported:

- 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 Gaussian Model Adaptive Processing (GMAP) algorithms.
- 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 particular US WSR88D legacy requirement.



Time series of I and Q and the corresponding Doppler power spectrum obtained 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), i.e., the spectrum moments.

Figure 38 Time Series and Doppler Power Spectrum Example

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 38 on page 197 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 ?, 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. 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 RVP900 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. The "gain" of the window is set to preserve the total power.

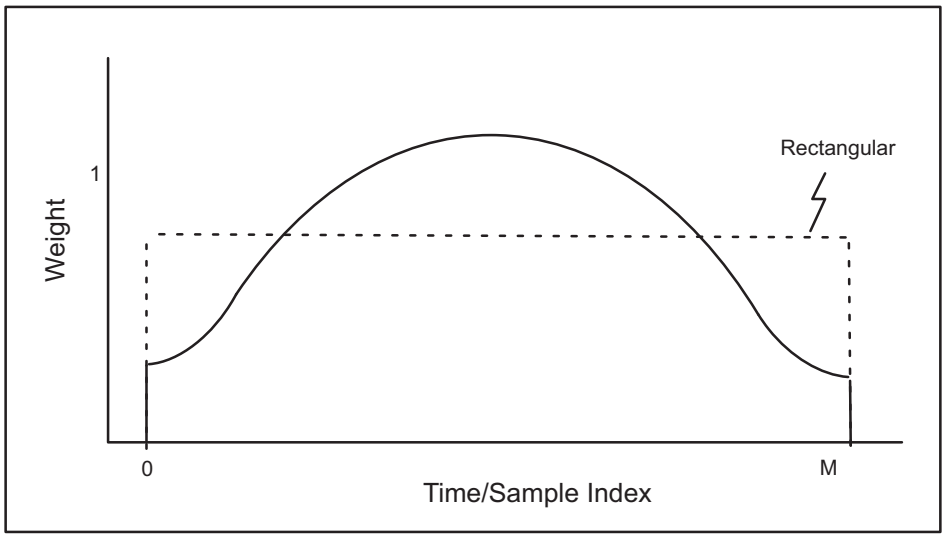


Figure 39 Typical Form of Time Series Window

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 40 on page 200 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.

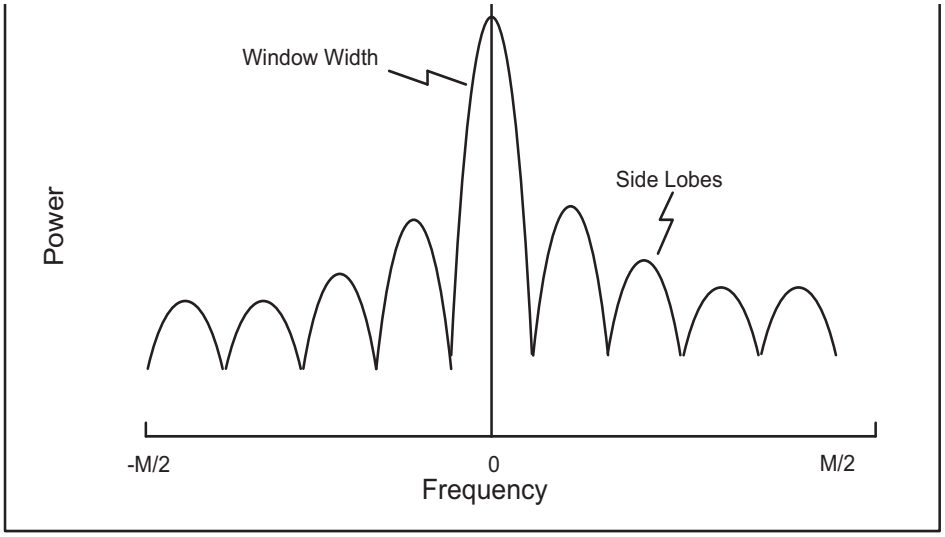


Figure 40 Impulse Response of Typical Window

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 RVP900 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 RVP900 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 38 on page 197. 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 Ray Synchronization on Angle Boundaries

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 Section 7.3 Setup Operating Parameters (SOPRM) on page 259). However, when the RVP900 is aligning its processed rays to AZ/EL angle boundaries (see Section 7.15 Load Antenna Synchronization Table (LSYNC) on page 312) 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, suppose the RVP900 is operating at 1 KHz PRF, 20-deg/sec scan rate, 1-degree ray synchronization, and a Sample Size of 80. Then, if the LSYNC Dyn bit is set, rays will consist of a full 80-pulses ending at each angle and extending back 30-pulses into the previous angle sector. However, when Dyn is clear, there will be 50 pulses used for each ray (not 80) and those pulses will exactly fill the scanning angle sector.

6.2.5 Clutter Filtering Approaches

Each major mode implements clutter filtering as follows:

- FFT Major Mode uses frequency domain clutter filters, including GMAP. The power spectra are restored by interpolating across the gap of removed spectral points.
- 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. The fixed width and variable width spectral clutter filters can be used in PPP mode; however, the power spectra and the spectra of cross-correlations are not interpolated.
- Random Phase Mode uses frequency domain clutter filters, including GMAP. The power spectra are restored by interpolating across the gap of removed spectral points.
- Batch Mode uses a simple DC removal for the small batch clutter filter. The high PRF large batch is then processed using frequency domain clutter filters, including GMAP. The power spectra are restored by interpolating across the gap of removed spectral points.

Earlier vintages of radar signal processors sometimes used an IIR (infinite impulse response) filter for clutter rejection. 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 RVP900, 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 RVP900. 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 RVP900 are configured using the **mf** setup command (Section 4.2.3 Mf — Clutter Filters on page 112). These are:

- Type 0: Fixed width filters with interpolation
- Type 1: Variable width single slope adaptive processing
- Type 2: Reserved (not used at this time)
- Type 3: GMAP

6.2.5.1 Fixed Width Clutter Filters

This filter, illustrated in Figure 41 on page 204, 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.

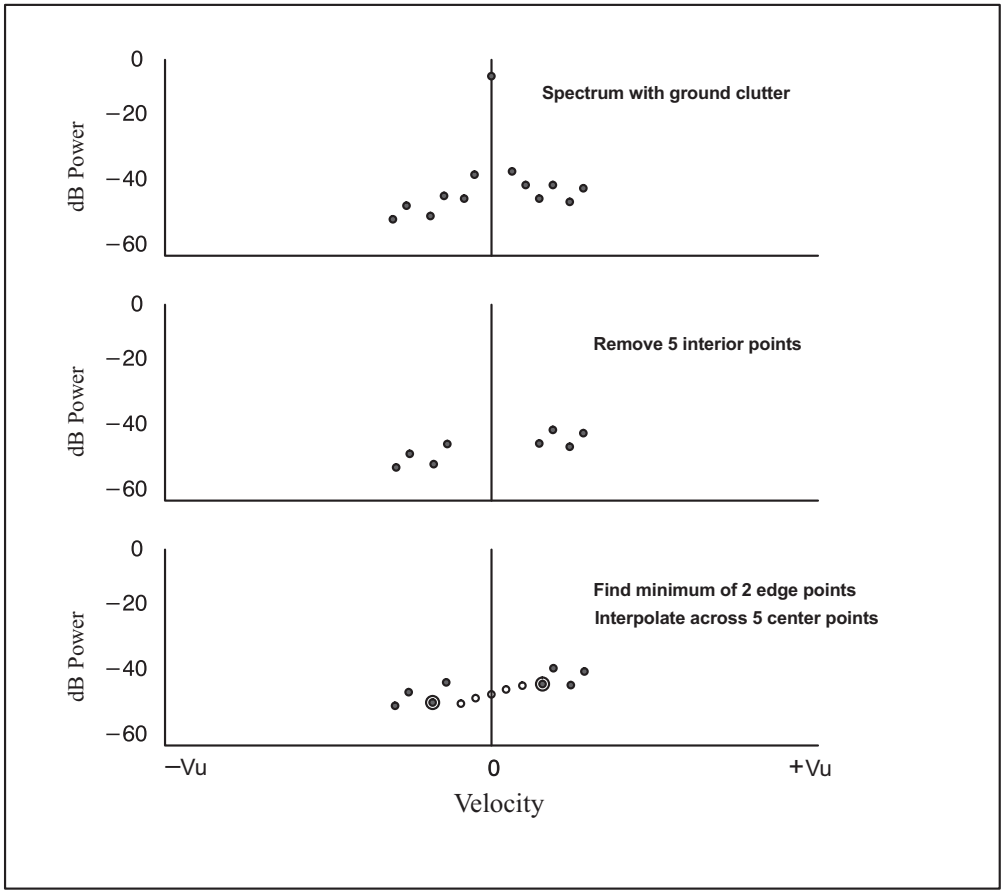


Figure 41 Example of Fixed Width

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.

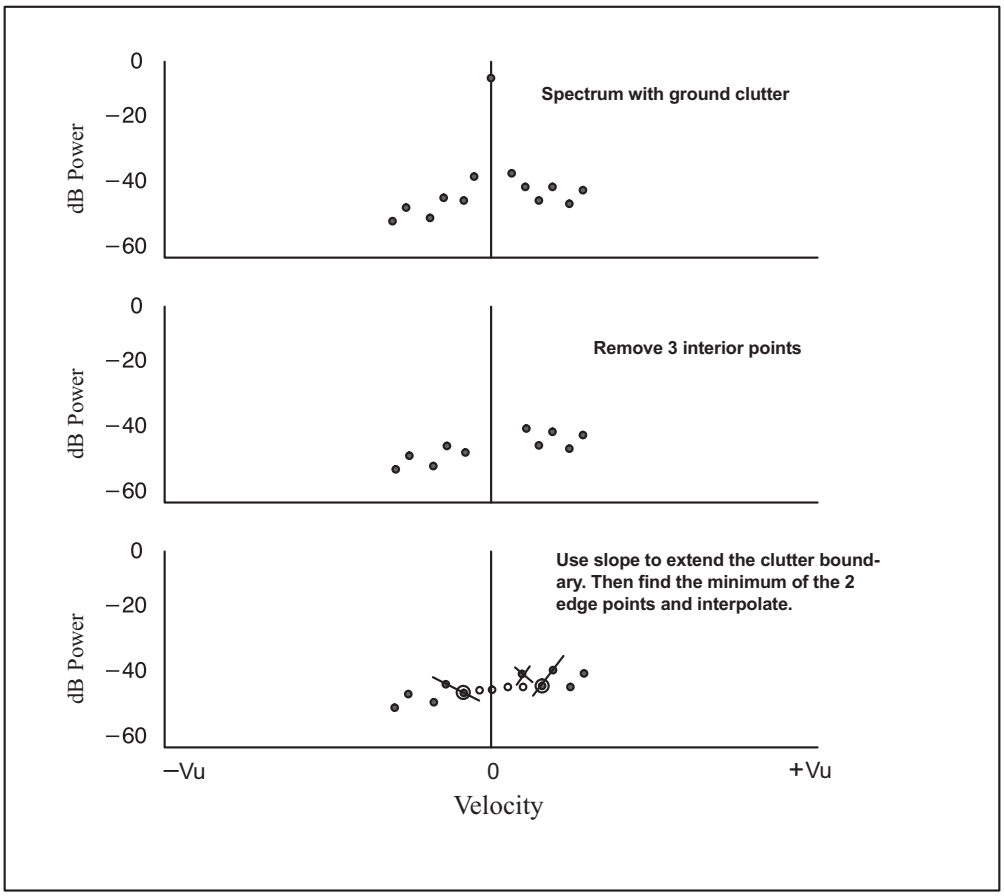


Figure 42 Variable Width Clutter Filter

In Figure 42 on page 205, 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 GMAP

GMAP is a new adaptive technique developed at Vaisala that is possible on a high-speed processor such as the RVP900. 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.

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 in Figure 43 on page 207.

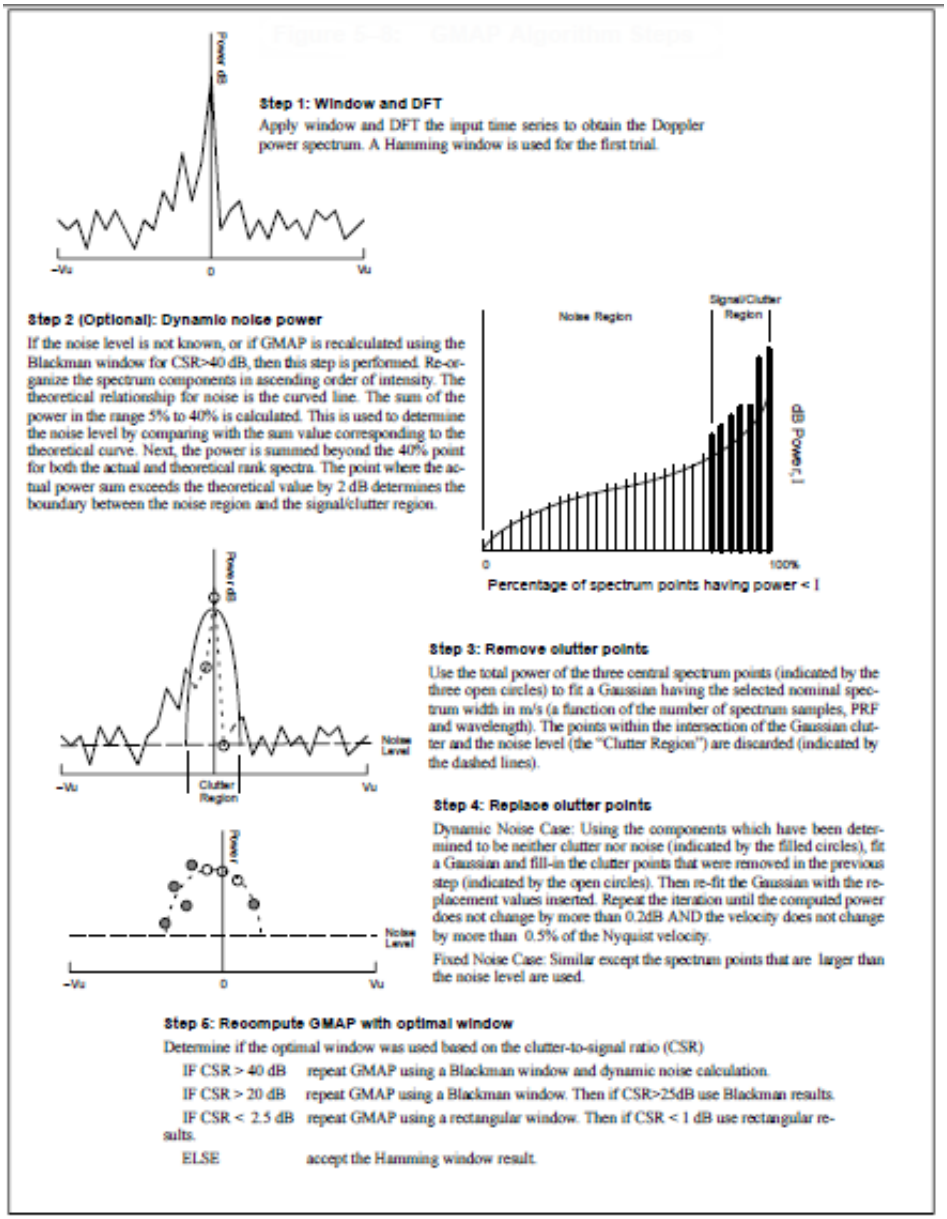


Figure 43 GMAP Algorithm Steps

- **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.

As mentioned in Section 6.2.2 Frequency Domain Processing-Doppler Power Spectrum on page 198, 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 43 on page 207 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: in the presence of very strong clutter, even a small amount of phase noise causes the spectrum noise level to increase, and 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 43 on page 207, 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 43 on page 207. 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.[2] 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 43 on page 207 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 43 on page 207. 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 RVP900 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 44 on page 213 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.

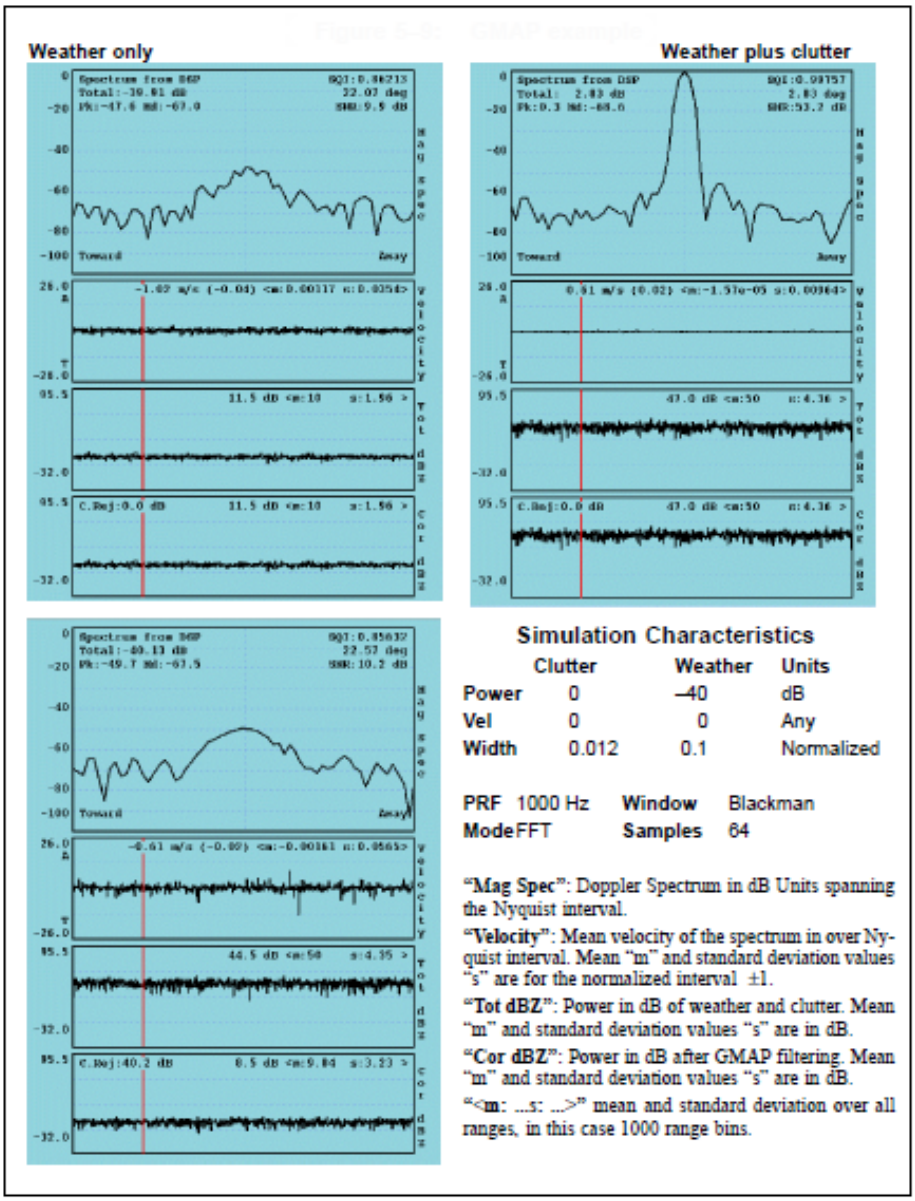


Figure 44 GMAP Example

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-meteorological target of very narrow range. These are either small strong targets, such as airplanes, ships, or other moving objects.

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 Doppler filtered power exceeds the normal level before and after in range. In other words, flag all bins if:

$$[(R_0(r) > TCM*R_0(r - Offset)) \text{ and } (R_0(r) > TCM*R_0(r + Offset))]$$

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:

$$R_0(r) = \left(\frac{r_{end} - r}{r_{end} - r_{start}}\right) * R_0(r_{start}) + \left(\frac{r - r_{start}}{r_{end} - r_{start}}\right) * R_0(r_{end}) \text{ , where } r = r_{start}, r_{end}$$

Note that the same thing is done for all the filtered autocorrelations and crosscorrelations (lagged moments R_1, R_2, and dual-polarization crosscorrelations) as soon as any of the filtered autocorrelation R_{0H}, R_{0V}, or R_{0HV} exceeds the threshold. The unfiltered autocorrelations (total power) T_{0H}, T_{0V}, and T_{0HV} are conserved.

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 RVP900 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.

Figure 45 Radar Equation

$$Z = CSr^2 = \left[\frac{Cr_0^2 N}{g_r g_t}\right]\left[\frac{r^2}{r_0^2}\right]\left[\frac{T_0 - N}{N}\right]$$

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 49 on page 232). The parameter I_o is the measured noise power at IF with appropriate calibration for the system gain. Calibration of the RVP900 involves defining the radar constant C and measuring the value of I_o. This is discussed in detail in Section 6.5 Reflectivity Calibration on page 231.

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 RVP900 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 Section 6.3.7 Clutter Correction (CCOR threshold) on page 221.

6.3.3.1 Noise Correction to Reflectivity Calibration

The dBZ0 number in the above equation is the number which sets the sensitivity of the radar. Lower numbers mean greater sensitivity. In Figure 45 on page 216 it was assumed that the noise level at calibration time is the same as the noise level at run time. And that any changes in measured noise level were due to changes in receiver gain not sensitivity.

Modern digital receivers and low-noise amplifiers are very gain stable, and this is generally not true. One example of a relatively large noise level variation is the thermal noise from the relatively warm earth and atmosphere. So, the bottom degree or so of elevation has a different noise level, and thus a different sensitivity, and thus a different dBZ0. Normally we calibrate while aiming the antenna up in the air in a direction away from the surface, or the sun. For this discussion, let us define two new noise values:

N sub c = Noise level at calibration time.
N sub r = Noise level at ray processing time.

If we answer yes to "Enable noise power based correction of Z0" in the Mp non-volatile setup section, then Figure 45 on page 216 becomes:

$$Z = CSr^2 = \left[\frac{Cr_0^2\ Nc}{g^r g^t}\right]\left[\frac{Nr}{Nc}\right]\left[\frac{r^2}{R_0^2}\right]\left[\frac{T_0 - Nr}{Nr}\right]$$

Figure 46 New Radar Equation

In this equation, the dBZ0 is the term $\left[\frac{Cr_0^2 Nc}{g^r g^t}\right]\left[\frac{Nr}{Nc}\right]$. The dBZ0 fed into the RVP900 is the basic $\left[\frac{Cr_0^2 Nc}{g^r g^t}\right]$, while the processor modifies the value by the ratio *Nr/Nc*. Values read out by the GPARM command, etc. will be the modified value.

Note that this is the only place where the calibration-time noise level Nc is used in the processing. It is possible for this value to be unknown, in which case it is set to "nan" internally. In this case, if you request the corrected values, the correction will not be applied, and you will get the message "Cannot enable Z0 noise correction because calibration is missing" in the rvp log file.

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,

$$V = \frac{\lambda}{4\pi\tau_s}\theta_1 \text{ where } \theta_1 = \arg\{R_1\}\ .$$

? is the radar wavelength, τ_s is the sampling time (1/PRF). $?_1$ is constrained to be on the interval [-?, ?]. When $?_1 = \pm ?$, 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 RVP900 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 $V_u + ?V$, then the velocity calculated by the RVP900 is - $V_u + ?V$, which is $2V_u$ away from the true mean velocity.

For 8-bit outputs, rather than calculating the absolute velocity in scientific units, the RVP900 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 RVP900, 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.

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 RVP900 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 [-?, ?]. Thus, for example, a variance of $?^2/25$ 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 RVP900:

$$W = \frac{\sqrt{Variance}}{\pi}$$

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 RVP900 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 through 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 RVP900 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 RVP900 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 RVP900 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
PMI	Polarimetric Meteo Index	PMI > 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:

LOG — This is a measure of signal strength that is usually used for the thresholding of reflectivity data. The default LOG threshold value is 0.75 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 0.4.

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 -18 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 5 dB. If R_2 processing is used, this can usually be reduced to 5 dB for width thresholding.

PMI — The PMI is typically used to reject echoes that are identified as inconsistent with the preferred hypothesis of precipitation. Identification is based on combined interpretation of polarimetric measurements of reflectivity, differential reflectivity, differential phase and co-polar correlation coefficient by the HydroClass pre-classifier. The default threshold value is 0.45.

The following are the default threshold combinations for each of the parameters that can be selected for output from the RVP900:

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:

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 approximately1 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 approximately - 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. If you average more pulses or ranges, then the threshold level can usually be reduced.

PMI — To optimize the PMI level, consider data acquired in the mode of (H+V) in PPP processing. Having first optimized the previous four Doppler qualifiers, inspect echo classification data of DB_HCLASS and seek for gates declared as "NoMet", which are unlikely of meteorological origin. These bins may appear as "NoMet" data in your display, or DB_HCLASS data might be readily thresholded, depending on your color scales and the HydroClass configuration. In order to get the other data types thresholded in the same fashion, activate PMI as a thresholding mechanism in task configuration. The PMI threshold value to 0.45 implies the same strength of suppression to other selected data types as seen in DB_HCLASS. It is possible to recover more precipitation data, typically at edges of precipitation and the most far echoes (virga) by reducing the PMI threshold, as appropriate. In these customizations, the behavior of DB_HCLASS remains unchanged.

When thresholding dual pol, 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 Section 4.2.3 Mf — Clutter Filters on page 112). 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 dual pol or 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 dual pol and 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, which is the loss of LOG data within regions of high shear, even though, for instance, the reflectivity itself was strong. You can experiment with applying a secondary SQI threshold to help clean up the LOG data, without introducing any significant black holes.

6.4.3 Speckle Filters

A speckle filter is a final pass over each output ray, in which isolated bins are removed. There are two speckle removers in the RVP900:

- 1D single-ray speckle filter (default)—This is used for any output parameter.

- 2D 3x3 speckle filter—If enabled, this is used for any output parameter.

The 1D and 2D speckle filters are enabled using the soprm command. Input 2, as documented in Section 7.3 Setup Operating Parameters (SOPRM) on page 259. These can both be turned on, if desired, and in this case, the 1D filter is applied first.

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 they are not affected by the speckle filters. There are two primary benefits to 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 speckle filter actually performs data filling of "missing speckles" as well as eliminating isolated speckle bins. The following sections describe the algorithms.

6.4.3.1 1D Speckle Filter

A ray is the basic azimuth unit of the RVP900 (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 Figure 47.

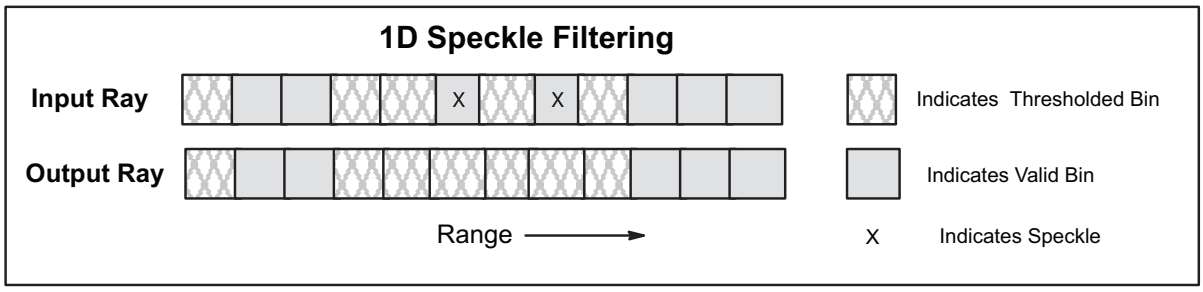


Figure 47 1D Speckle Filtering

There are two independent 1D speckle removers: one for the reflectivity data (dBT, dBZ, Z_{dr}, and LDR when using dual-pol), and one for the Doppler data (V, W, PhiDP, RhoHV, and K_{dp} when using dual-pol). 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, it is recommended that you switch both speckle filters off. When either speckle filter is switched on, it is applied in HydroClass.

6.4.3.2 2D 3x3 Speckle Filter

The 2D 3x3 speckle 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, Velocity, and Width data are candidates for this filtering step; all other parameters are processed using the default 1D speckle filter.

The rules for the 2D 3x3 speckle filter are as follows:

	Center Point Action	
	Assign Threshold	**Else**
Valid Center Point	If there are none or only one other valid point in the 3x3.	Do Nothing. Pass the center point value as-is.
Thresholded Center Point	If there are five or fewer valid neighbors in the 3x3.	If there are six or more valid neighbors in the 3x3, average to fill the center point.

The 2D 3x3 filter performs two functions:

- Filling by interpolation
- Thresholding of isolated noise bins

Some examples are shown graphically in the Figure 48.

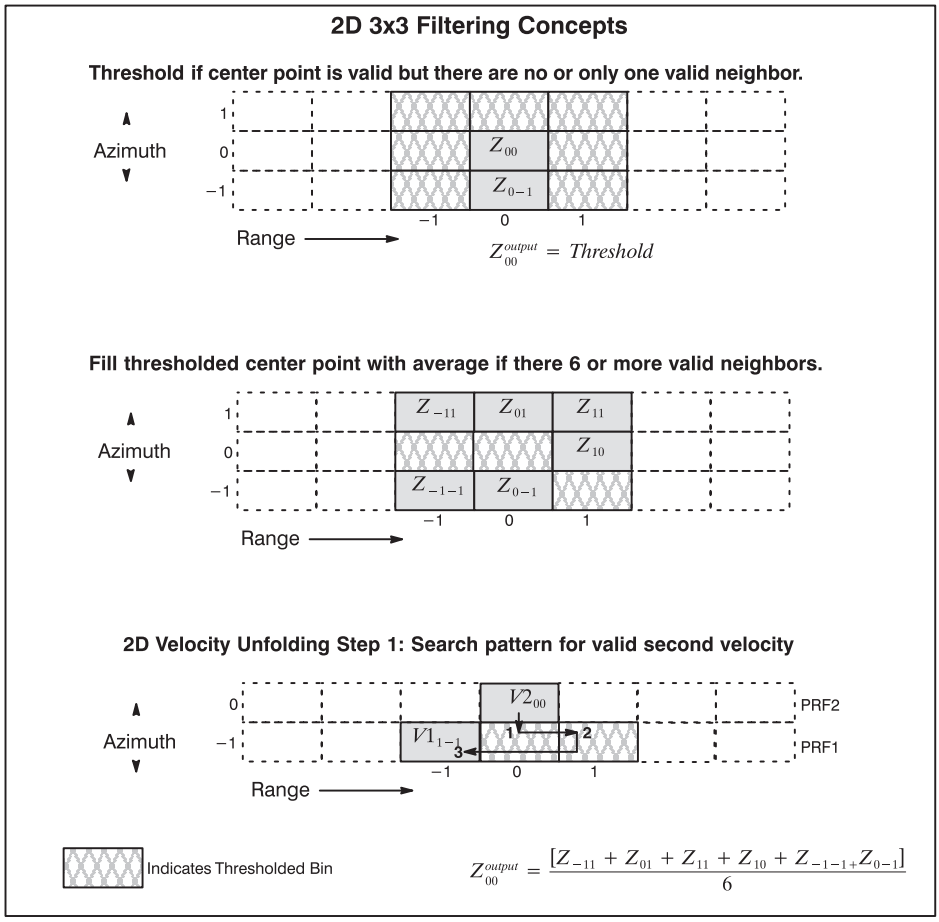


Figure 48 2D 3x3 Filtering Concepts

For all the parameters except velocity, 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 2D 3x3 speckle filter can be used with or without the 1D speckle filtering, however; the data is cleaner if both filters are applied.

The 2D 3x3 speckle filter has some interesting properties when combined with other algorithms.

6.4.3.2.1 Dual-PRF Unfolding

Dual-PRF velocity unfolding is computed within the 3x3 filter whenever both are enabled. There are two steps to the process:

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.

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 second trip range bins. At a constant PRF, the second 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 Section 6.3.3 Reflectivity on page 215 required the calibration reflectivity dBZ_0 . This section describes it's derivation. You can use the **zauto** utility to perform the calibration. (refer to the *IRIS/RDA Utilities Manual*.)

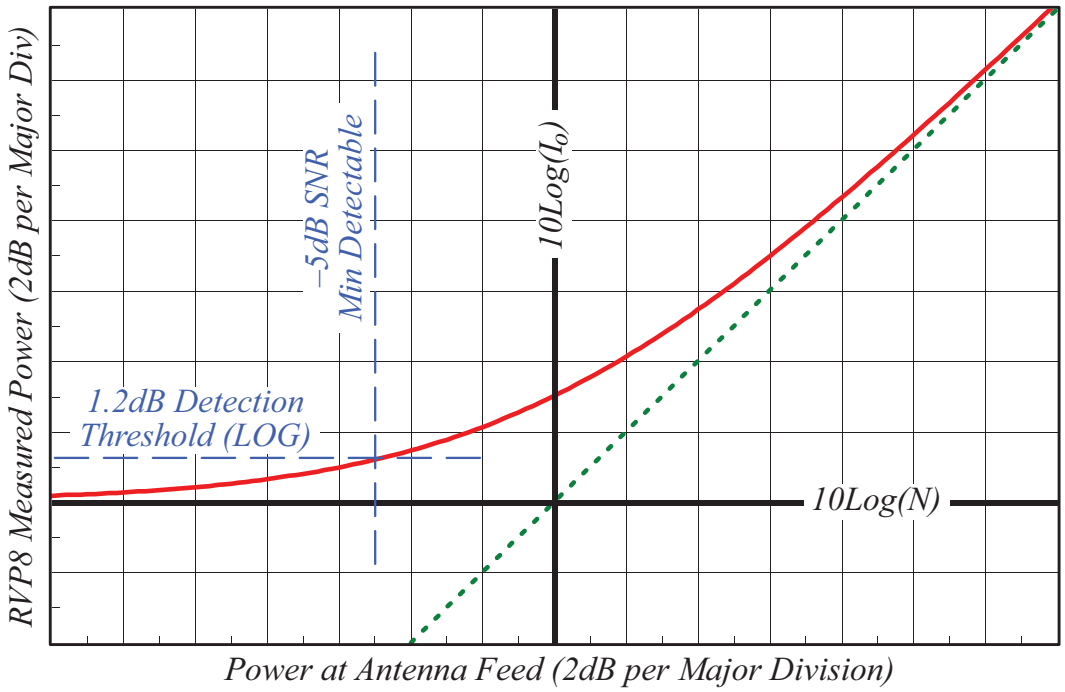


Figure 49 Model Intensity Curve

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 49 on page 232. 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 RVP900. 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.

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 RVP900 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 49 on page 232. 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 49 on page 232. 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 IFDR 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_o 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 (see Section 6.5.3 Treatment of Losses in the Calibration on page 235).

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 50 on page 236 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 IFDR 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.

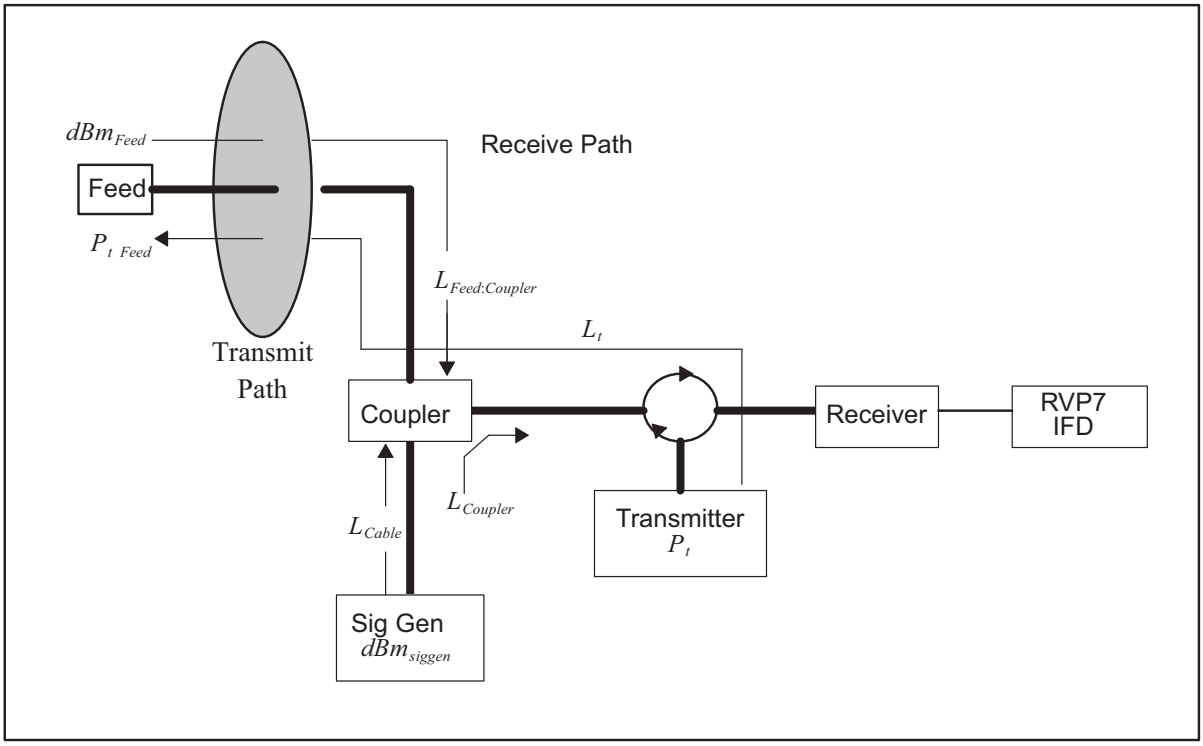


Figure 50 Illustration of Losses that Affect LOG Calibration

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,

?	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.
?	Horizontal half-power full beamwidth.
?	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 .

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:

?	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
?	Horizontal half-power beamwidth in degrees	1 degree
?	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:

$$dBZ_0 = 10\log[Cr_0^2 I_0] = 10\log[(6.76 \times 10^6)(1)^2(3.16 \times 10^{-11})]$$

$$= -36.7dB(mm^6 m^{-3})$$

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 RVP900.

The RVP900 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 20 KHz) 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 20 KHz); 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 RVP900 "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 RVP900 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 RVP900 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 RVP900 "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 RVP900 **Mc** setup menu. There is 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 RVP900 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 ? operating at a fixed sampling period $?_s$ = 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 ? for a given $?_s$. If the unambiguous range interval is made sufficiently large by increasing $?_s$, then the resulting velocity span may be unacceptably small.

The RVP900 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 $?_l$(for the low-PRF) or $?_h$ (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 $?_l$ and $?_h$ 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 $?_h$ 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 [- ?, ?] interval are returned to that interval by appropriate additions of ± 2? . These angles correspond to the ordinary single-PRF Doppler velocity measurements, and the ± 2? 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 ? to be the difference between the two measured phases then:

$$\phi = \theta_l - \theta_h = \frac{4\pi}{\lambda}[\tau_l - \tau_h]V$$

which can be interpreted as a phase angle within the unfolded interval:

$$V_{u\,unfold} = \frac{\lambda}{4(\tau_l - \tau_h)}$$

Now if $?_l$ and figure $?_h$ 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 [- ?, ?] , 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 RVP900 uses it only as a rough estimate in determining how to unfold the individual velocity measured from each PRF.

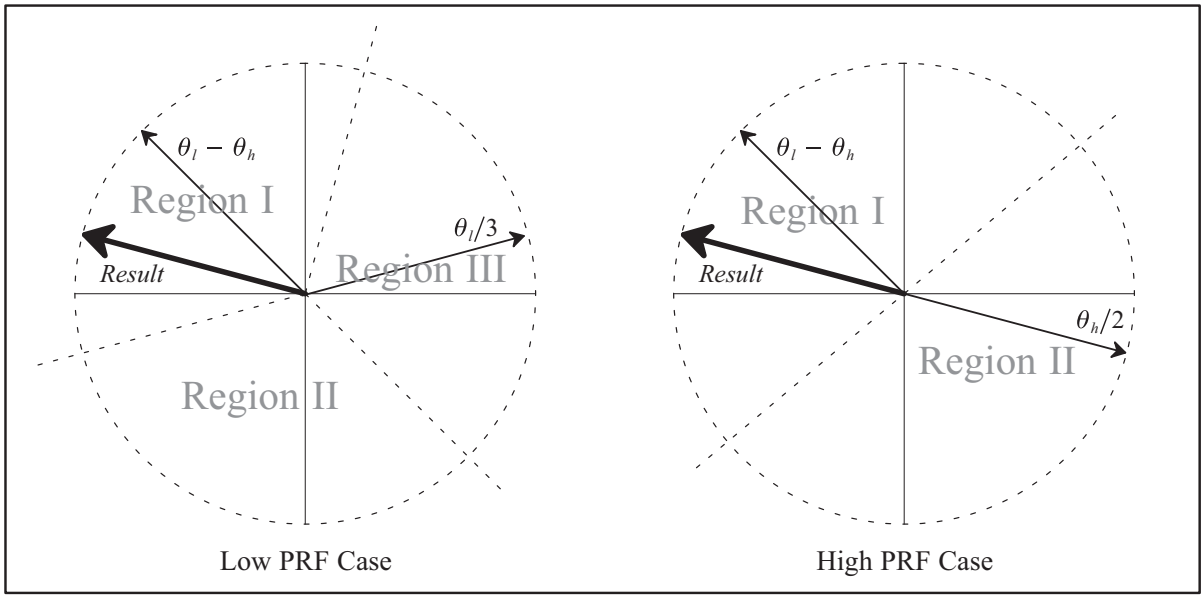


Figure 51 Dual PRF Concepts

This technique is illustrated in Figure 51 on page 242. 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 RVP900 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 RVP900 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 RVP900 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 RVP900 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 RVP900 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 RVP900 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 52 on page 245 shows seven possible scope traces (and their associated probabilities) for the RVP900 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.

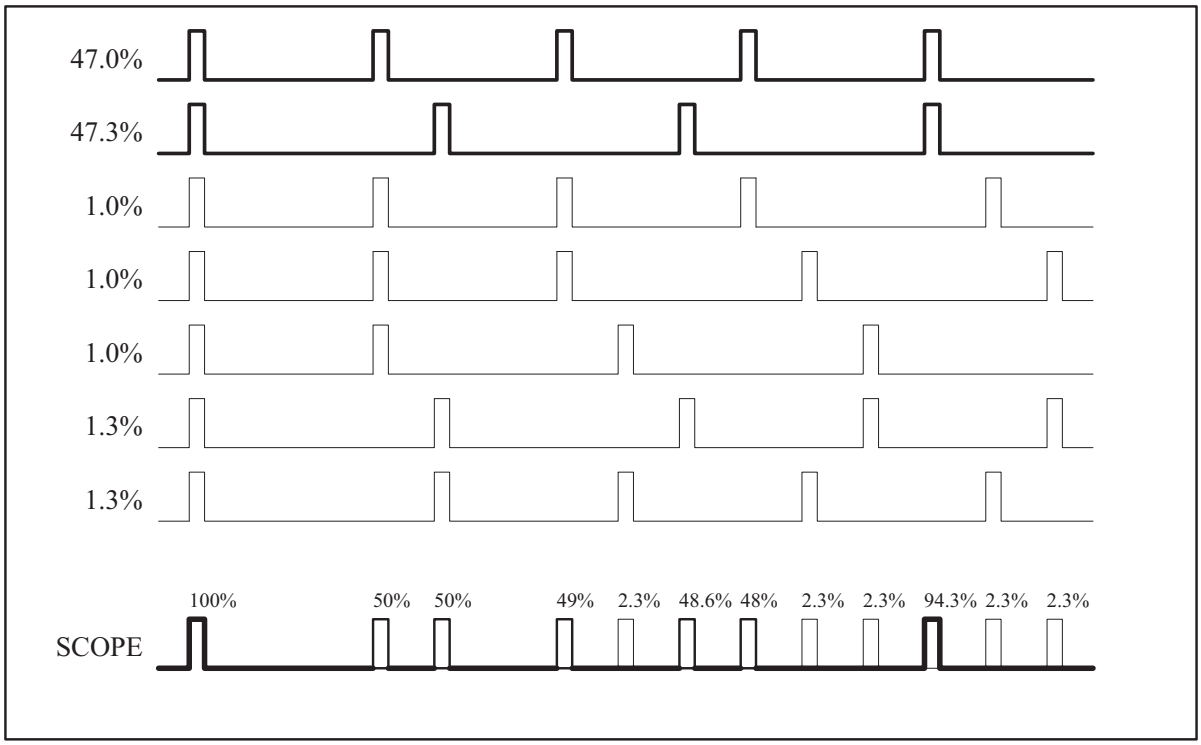


Figure 52 Example of Dual PRF Trigger Waveforms

6.8 Random Phase Second 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 RVP900 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 RVP900 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 RVP900 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 RVP900.

6.8.2 Algorithm

Figure 53 on page 250 shows a schematic of the data processing for random phase. The figure shows the Doppler spectra for the first and second 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 first and second 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 second trip echo interference. The ideal second 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 first or second 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 RVP900 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 first and second 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 RVP900 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. 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. However, in Random Phase mode, if SQI is applied to reflectivity or dual pol data, we need to relax this convention, specially in random phase processing, 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 and dual pol data in all the processing modes. The RVP900 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 secondary 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 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 reflectivity and dual pol data alone.

The *Slope* and *Offset* values should be adjusted so that the density of speckles in 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 trade-off of speckles versus sensitivity for your system in all modes of operation. Even with proper adjustment, it is normal for *dual pol*, *dBZ* and *dBT* data to show "holes" in regions of weather that have high turbulence or shear when SQI threshold is applied to that data. 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 dB to 35 dB 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 RVP900 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 RVP900 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 15 dB, then the algorithm would generally be blind to second-trip weather in the range interval from 100 km to 117.8 km.

The explanation for this can be found in the $1/r^2$ geometric correction for weather echo intensity. At ranges less than 17.8 km, the first trip weather would generally dominate the second trip weather by more than 15 dB. Thus, the initial 17.8 km ring of second trip data would be rejected by the CSR criteria. However, if the CSR were increased to 30 dB, then the size of this missing ring would be reduced to only 3.2 km.

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 second 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 second trip seams that occur for each ray. In fact, this smoothing of the second trip seam makes the dual PRF random phase mode work even better than the single PRF random phase.

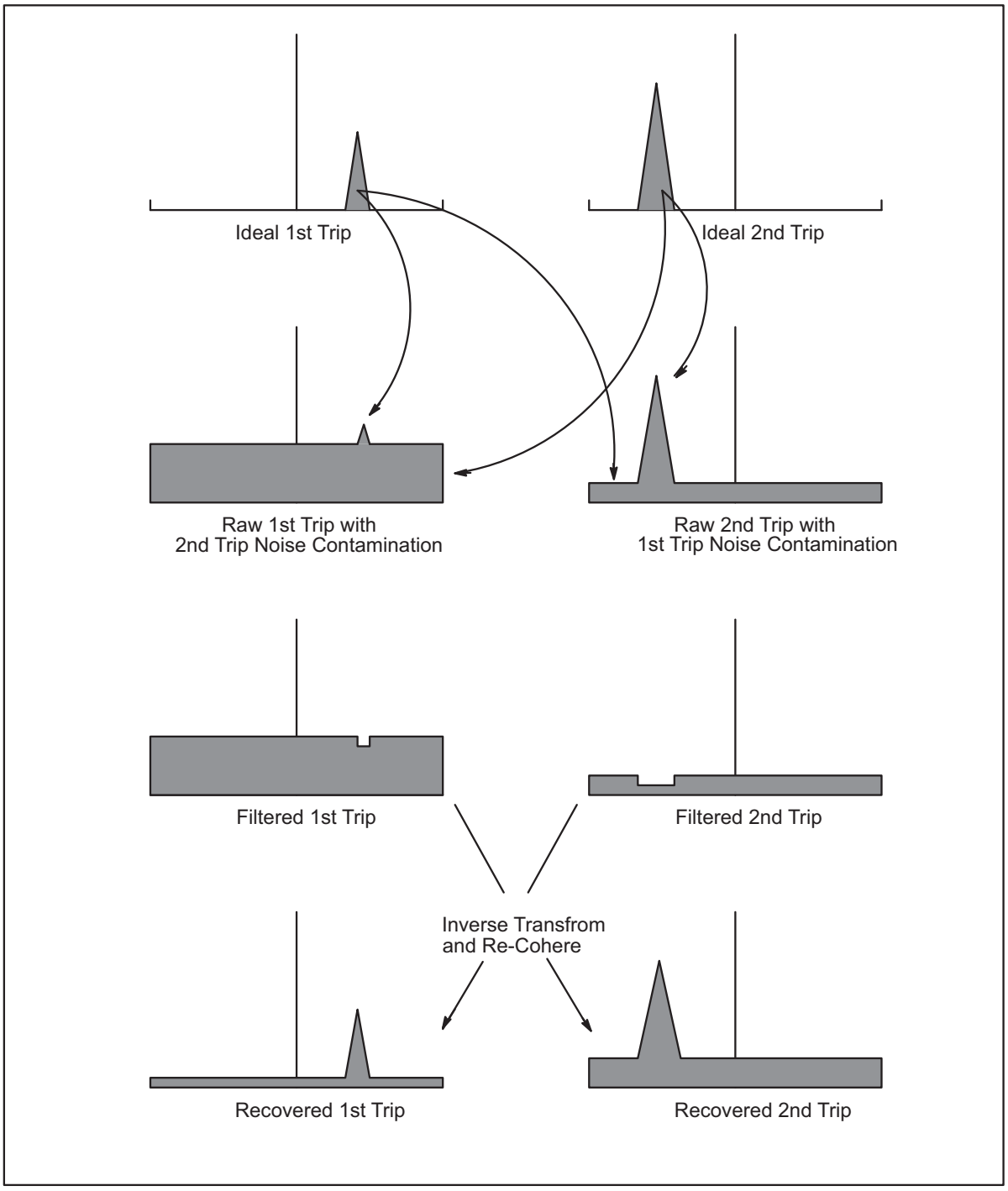


Figure 53 Random Phase Processing Algorithm

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 RVP900 processing algorithms. These tests are performed 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 C, Installation and Test Procedures, on page 355.

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 $?_{bin}$ within a set of pulses that are separated by $?_s = 1/PRF$. 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 RVP900 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$(M\ ?_s)$ 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 300 km at 250 Hz 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 300 km, where V_u is the unambiguous Doppler velocity in meters/sec. Note that these phase difference calculations have made no assumptions about the RVP900 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 $?_{bin}$ was smaller than $?_s$, 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 RVP900 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:

$$\left|s^n_A\right| = \{S_A, S_A, S_A, S_A, S_a \ldots\}, \left|s^n_B\right| = \{S_B, 0, S_B, 0, S_B \ldots\}$$

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, 400 Hz. The reason for not choosing 100% depth is that we would loose the Burst phase reference when the amplitude became smallest. The 26 dB 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 RVP900 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:

$$SQI = \frac{S^2_A}{S^2_A + \left(\frac{1}{2}\right) S^2_B}$$

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 Section 6.9.1 Linear Ramp of Velocity with Range on page 251 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 RVP900 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 RVP900 are in the form of 16-bit words.

Before attempting to program the RVP900, 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 RVP900 passed all internal checks. Since all internal RVP900 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 RVP900 and can be monitored using GPARM.

The RVP900 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 RVP900 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 RVP900 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 RVP900 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 RVP900 with the post-processing and display of that data by the user. In this case, RVP900 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.

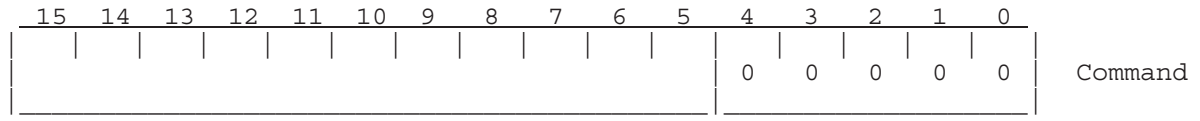
When the output FIFO is full and the RVP900 has the next word ready for output, there is another way that the idle wait loop can be exited, that is, if the processor detects that the user is performing a write I/O cycle. Since the user should have been reading data by now, the presence of a write cycle is taken to mean that some more important condition has arisen. As such, the wait loop is terminated and the RVP900 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 RVP900 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 RVP900 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 RVP900 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 Section 4.2.5 Mt<n>— Triggers for Pulsewidth #n on page 117), 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.

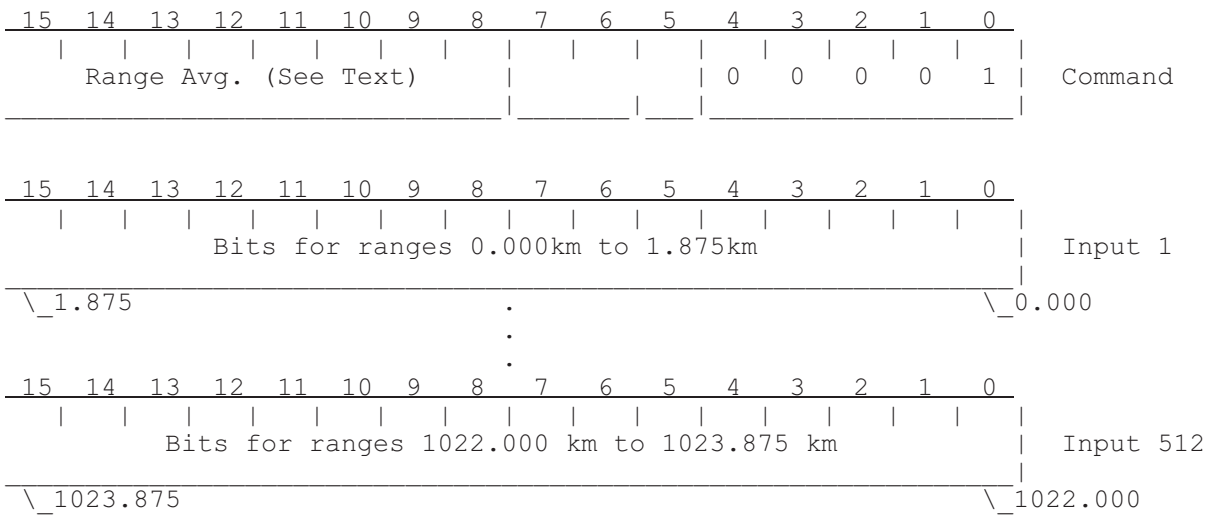
According to the range bins that are selected in the mask, the signal processor computes and stores internally a range normalization table which is later used to convert receiver intensity levels into reflectivity levels in dBZ. Note that the LRMSK command implicitly specifies the number of bins to be processed and output. The maximum bin count is

3072, though depending on the computational intensity of the configuration, the RVP900 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.

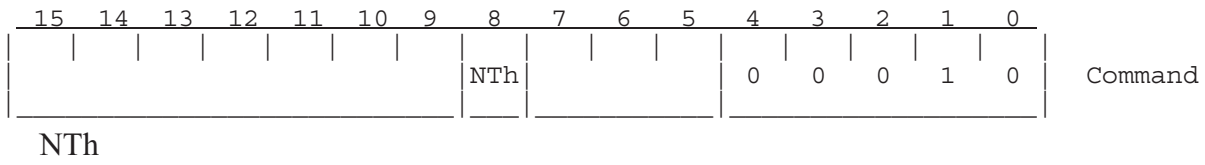
For example, suppose 100 bits are selected in the range mask and no averaging is elected. Then parameters are computed at those 100 ranges, and 100 bins of data are output. If the averaging were set to one, rather than zero, samples would still be taken at the same ranges, but pairs of bins would be 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 RVP900 forces the averaging to zero and turns on a single bin at zero range.



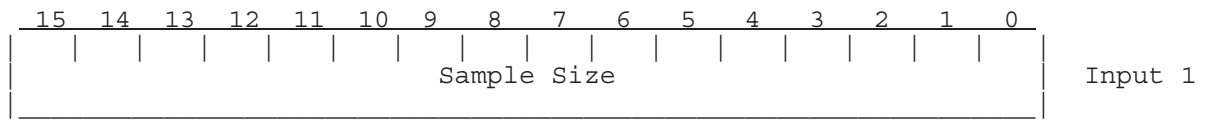
7.3 Setup Operating Parameters (SOPRM)

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.

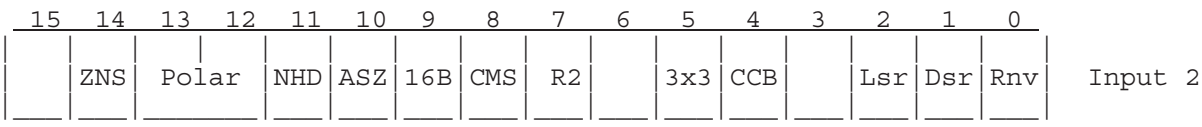


NTh

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 Section 7.29 Set Individual Thresholds (THRESH) on page 329), 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.

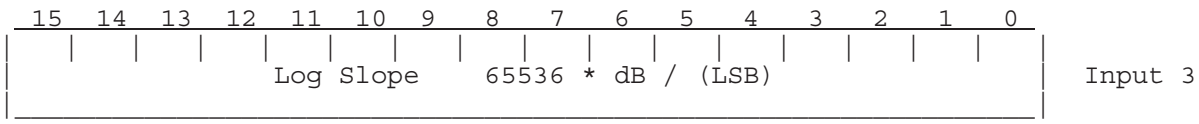
CMS — Enables Clutter Microsuppression, in which individual range bins are rejected (based on excessive clutter) prior to being averaged together in range.

R2 — Use three lag (R0/R1/R2) algorithms for width, signal power, and clutter correction.

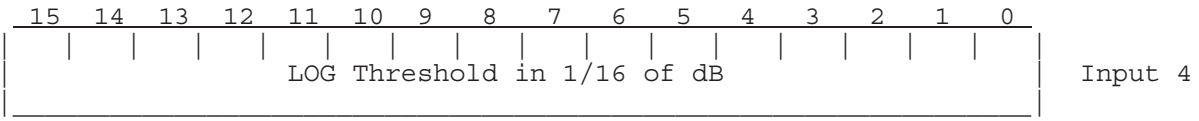
3×3 — 3x3 Switches on the 3x3 2D output filter (see Section 6.4.3 Speckle Filters on page 227). The RVP900 automatically handles all of the pipelining overhead associated with running the 3×3 filter, that is, valid output data are always obtained in response to every PROC command.

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	Lsr 1D reflectivity speckle remover. When set, range speckles in dBT, dBTa, dBZ, dBZa, SNR, ZDR, LDRH, and LDRV are removed.
Dsr	Dsr 1D Doppler speckle remover. When set, range speckles in V, W, PhiDP, PhiH, PhiV, RhoHV, RhoH, RhoV, and KDP are removed.
Rnv	Range normalization of reflectivity data. This bit also enables intervening gas attenuation correction.

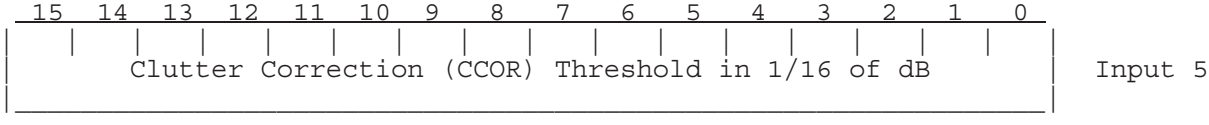
HClass is 1D speckle filtered when either Lsr or Dsr is set.



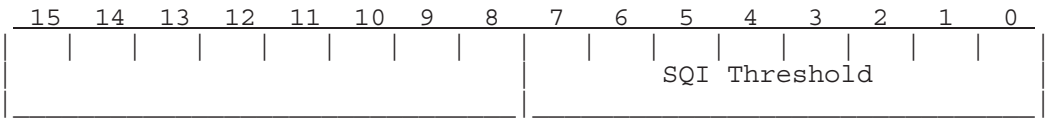
This number defines the multiplicative constant that converts the signal power in dB to the units of the 12-bit "Log of power in sample" time series outputs. One fourth of this slope is used to generate the "Log of Measured Noise Level" output from GPARM (word 6). The recommended value to use here is 0.03 (1966). This gives a dynamic range of 122 dB in 12 bits.



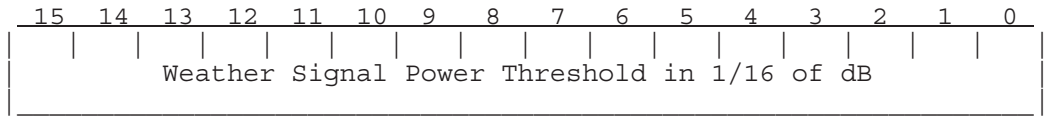
Reflectivity values below this level can result in thresholding of data, if the threshold control flags (see below) include LOG Noise bits. The threshold value is always non-negative, and the comparison test is described in section Section 6.4 Thresholding on page 224.



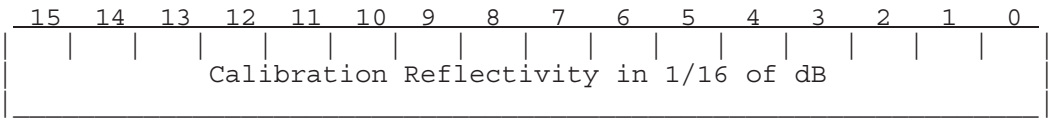
The clutter correction threshold is a bound on the computed log receiver adjustment for clutter. These corrections (in dB) are always negative. Any clutter correction which is more negative than the above value can result in thresholding of data.



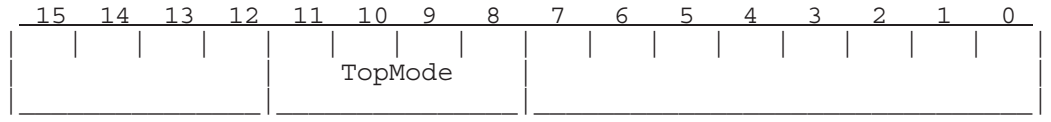
The Signal Quality Index (SQI) threshold is an unsigned binary fraction in the range 0 to 255/256. When the SQI for a range bin falls below the stated value, it may result in thresholding of data. An analogous Polarimetric Meteo Index (PMI) can be set by the command THRESH (see Section 7.29 Set Individual Thresholds (THRESH) on page 329).



Weather Signal Power (SIG) is an estimate of the SNR of the weather component of the received signal. When the SIG (see Section 6.3.8 Weather Signal Power (SIG Threshold) on page 223) 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 RVP900. Although the processing algorithms that are used in each top level mode are quite different, the RVP900 command set works in a uniform way in all modes.

0000 Polarimetric Processing or PPP Mode is a combined time domain and frequency domain approach that is used primarily for dual polarization applications; data are processed in batches of pulses (see Section 6.2 Time Series (I and Q) Signal Processing on page 195).

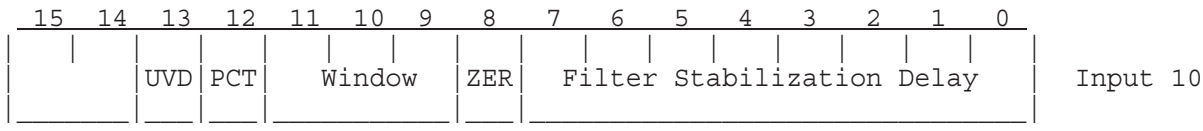
0001 FFT Processing Mode is a dedicated frequency-domain approach; data are processed in batches of pulses (see Section 6.2.2 Frequency Domain Processing- Doppler Power Spectrum on page 198).

0010 Random Phase Processing Mode. Data from first and second trips are dealiased in range based on knowledge of the radar transmitter phase (see Section 6.8 Random Phase Second Trip Processing on page 245).

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 Section 6.6 Dual PRT Processing Mode on page 238).

0101 DPRT-2 Processing Mode. The trigger generator produces alternate short and long pulses, and Doppler autocorrelations are computed using both pairs (see Section 6.6 Dual PRT Processing Mode on page 238).

11XX Four codes reserved for custom user modes.



The RVP900 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 RVP900 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.

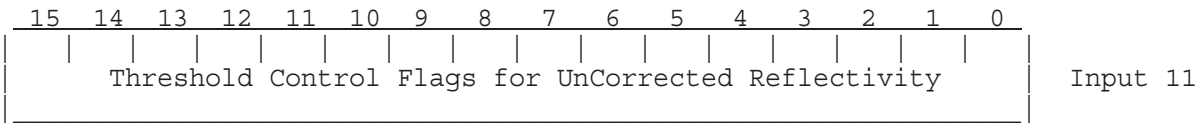
ZER If set, then the clutter filter's internal state variables are zeroed prior to waiting the delay time. For some signal conditions, this may give better results than allowing the filter to naturally flow into the new data.

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.

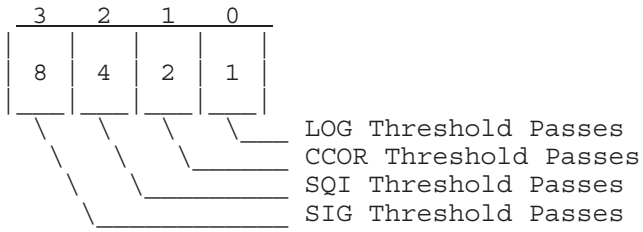
Window Selects the type of window that is applied to time series data prior to computing power spectra via a DFT. Choices are: 0:Rectangular, 1:Hamming, 2:Blackman, 3:Exact Blackman, 4:VonHann.

PCT — If set, the RVP900 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 Section 6.7 Dual PRF Velocity Unfolding on page 240.



These flags select which legacy 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. Further quality tests such as the Polarimetric Meteo Index exist, and they can be configured for each data type (see Section 7.29 Set Individual Thresholds (THRESH) on page 329). 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:

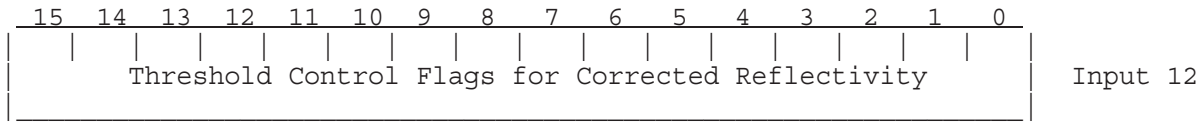
Value	Criteria
FFFF	All Pass (Thresholds disabled)
0000	All Fail (No data are passed)
AAAA	LOG
8888	LOG and CSR
A0A0	LOG and SQI
8080	LOG and CSR and SQI
F0F0	SQI
FAFA	SQI or LOG
C0C0	SQI and CSR
F000	SQI and SIG
C000	SQI and SIG and CSR
FFF0	SQI or SIG
CCC0	(SQI or SIG) and CSR

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.

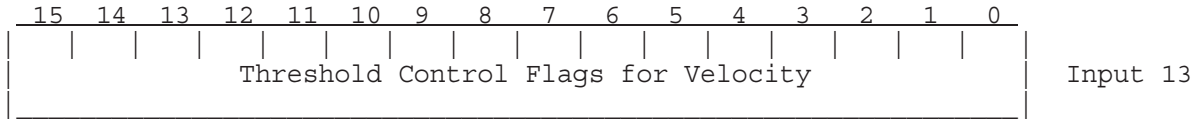
For example:

```
(SQI or SIG) and CSR    =    (F0F0 or FF00 ) and CCCC
                        =    (FFF0) and CCCC
                        =    CCC0
```

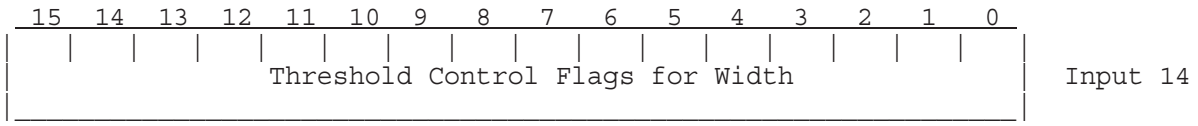
which corresponds with one of the examples given above.



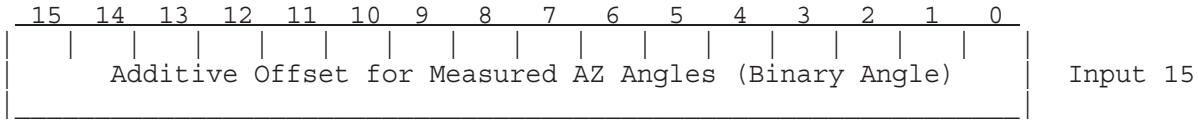
See Description for Input #11.

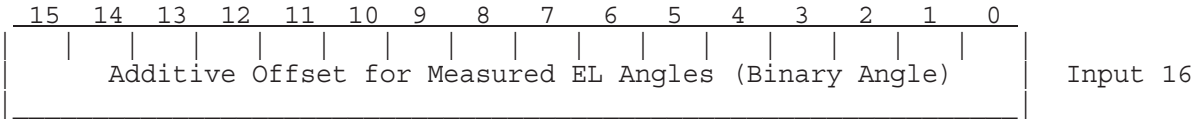


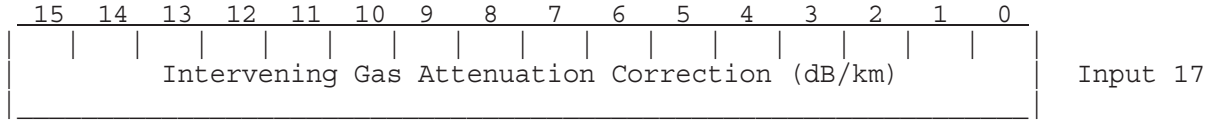
See Description for Input #11.



See Description for Input #11.







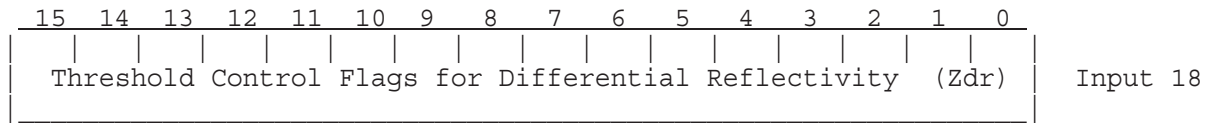
Gas attenuation correction attempts to compensate for overall (two-way) beam losses due to absorption by atmospheric gasses. The correction is linear with range, and is added to the data along with range normalization. Therefore, clearing the RNV bit in Word #2 above disables the correction. Of course, 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:

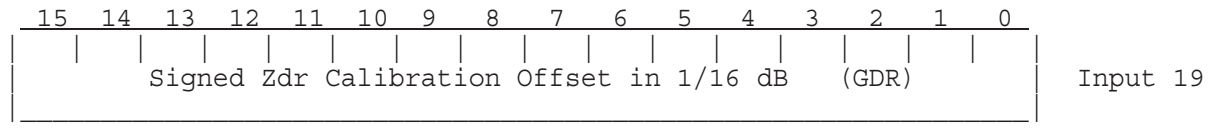
0 ? N ? 10000 G = N / 100000

else G = 0.1 + (N - 10000)/10000

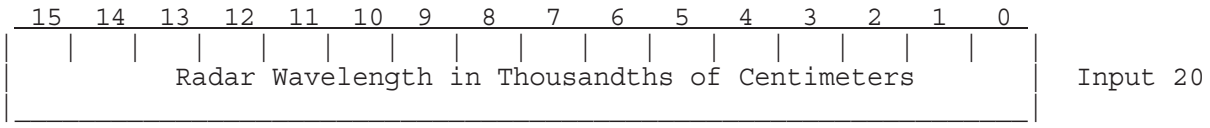
This format is backward compatible with the previous linear format for all values between 0.0 and 0.1 dB/km; but it extends the upper range of values from 0.65535 up to 5.6535. These larger attenuation corrections are needed for very short wavelength radars.



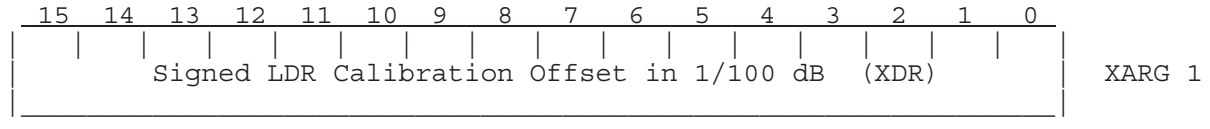
See Description for Input #11.



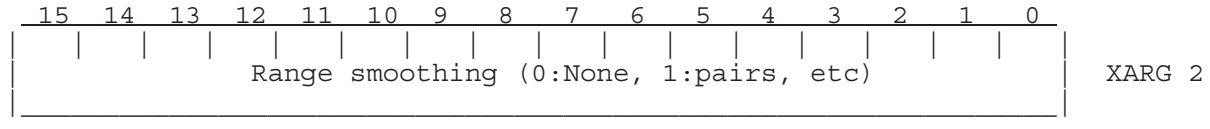
When differential reflectivity is computed there is a possibility that radar asymmetries will introduce a bias in the Zdr values, that is, that Zdr will be non-zero even when observing purely spherical targets. This calibration offset permits nulling out this effect. The GDR offset accounts for the overall Tx/Rx gain imbalance between the two channels of the radar.



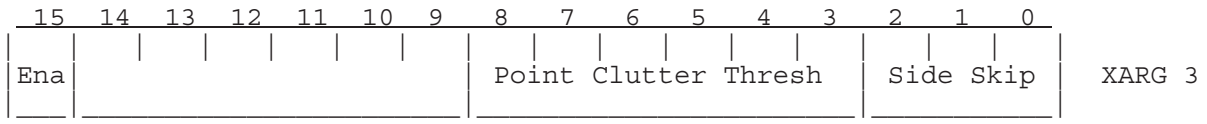
The radar wavelength is used in the calculation of 16-bit velocity and width data, to convert from Nyquist units to absolute physical units.



The XDR offset is used in the Linear Depolarization Ratio equations, and is the differential receiver gain between the two channels. Unlike the GDR offset (used for Z_{dr}), the gain difference does not depend on differential transmit power.

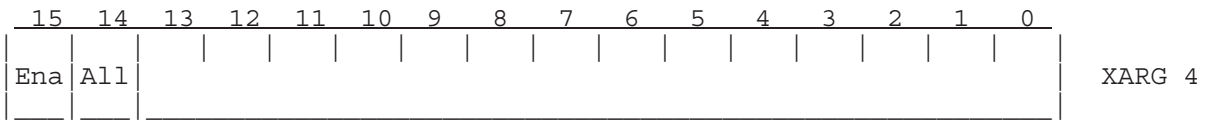


Range smoothing can be performed on raw moment data prior to the computation of scientific parameters. The number of bins to sum together is given here. This should generally be an odd integer so that no range bias is introduced by the smoothing operation.



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.

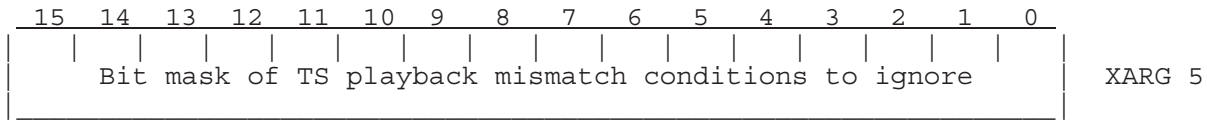
Ena — This bit is set to enable point clutter detection. Flag bits will then be reported in the "Flg" output data type of the PROC command.



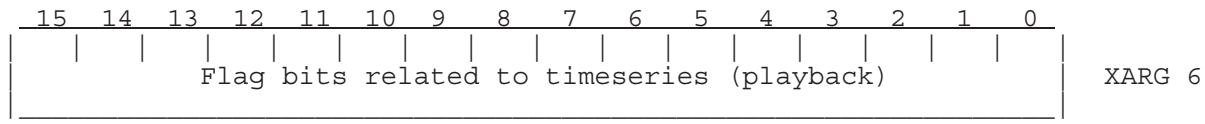
Point clutter censoring is configured with this word.

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.

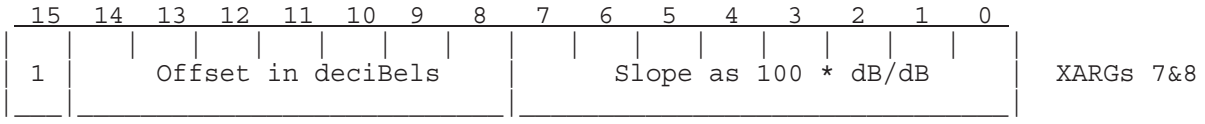
All — Optionally expand the reported detection flags to show the entire replaced interval, not just the original detected bins. This gives a more honest view of the data bins that have been altered.



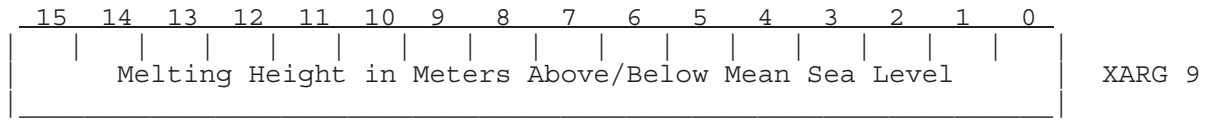
This word is a combination of MMTS_xxx bits specifying what types of mismatches are okay (do not cause an all-zero ray to be produced) during PROC command processing of timeseries data that are played back from an external source into the RVP900.



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 RVP900 defaults from the **Mf** menu.



During time series playback, the height recorded with the time series is used instead of this value.

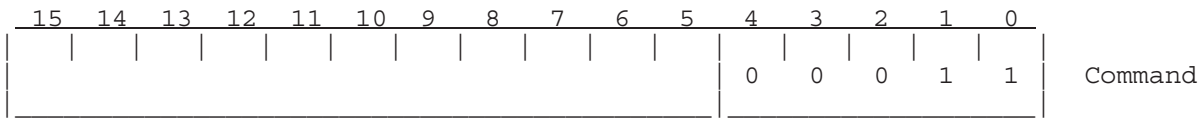
The default (power-up) values for the above parameters are listed below. Both the scientific units and the integer-input required by the command to set up that value are given. Most of these defaults will likely be reasonable for a wide variety of radars.

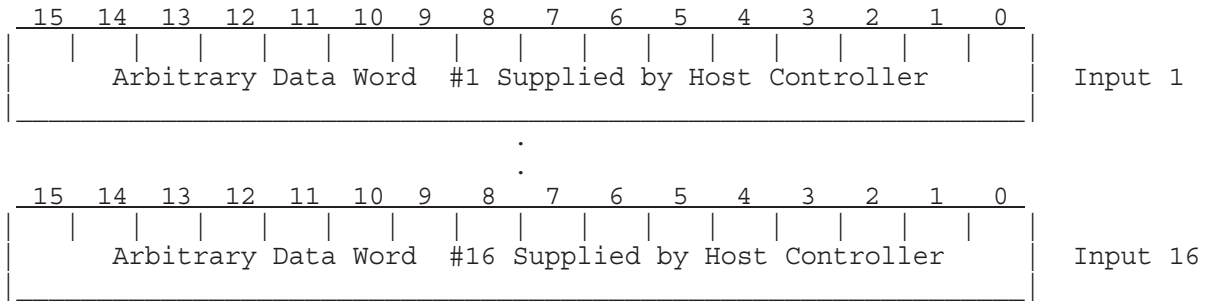
Table 13 Default Values For Operating Parameters

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

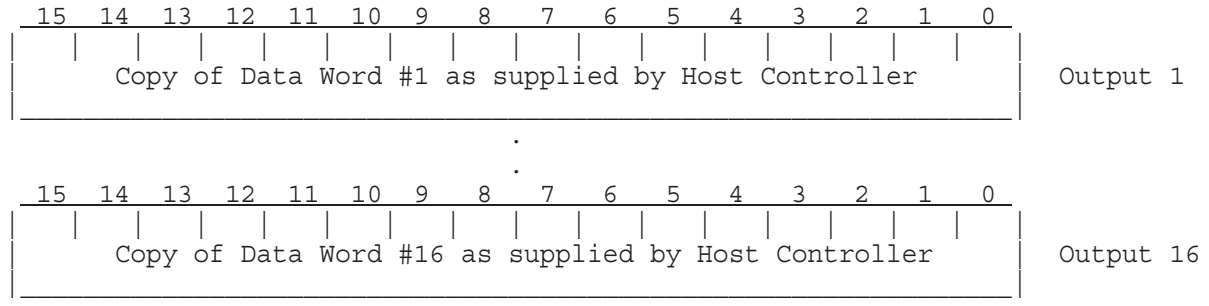
7.4 Interface Input/Output Test (IOTEST)

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).Section 7.20 Pass Auxiliary Arguments to Opcodes (XARGS) on page 320



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.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
											0	0	1	0	0	Command
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	Output 1
.																
.																
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Output 16

7.6 Sample Noise Level (SNOISE)

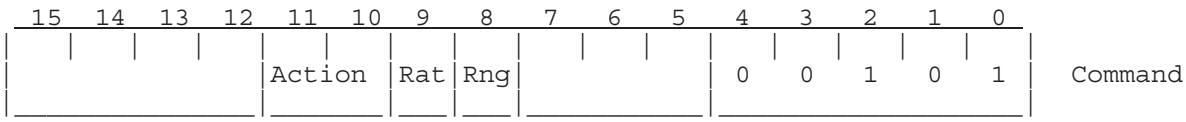
This command is used to estimate the current noise level from the receiver, so that the noise can be subtracted from subsequent measurements. Data are sampled for 256 pulses at 256 bins, beginning at a selectable range and spaced by the range resolution at that pulse width. The internal trigger generator 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 32 km 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 RVP900. The power-up sampling range is 250 km (input value of 250), and the power-up trigger rate is 200 Hz (input value of 30000). These initial values persist until such time as they are altered here. 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 992 km. The trigger rate, resulting from a given input, is 6 MHz divided by the input value, that is, the input value is the trigger period in 0.1667 ?sec increments. Keep in mind that the given rate is bounded against the minimum PRT allowed for the current radar pulse width.

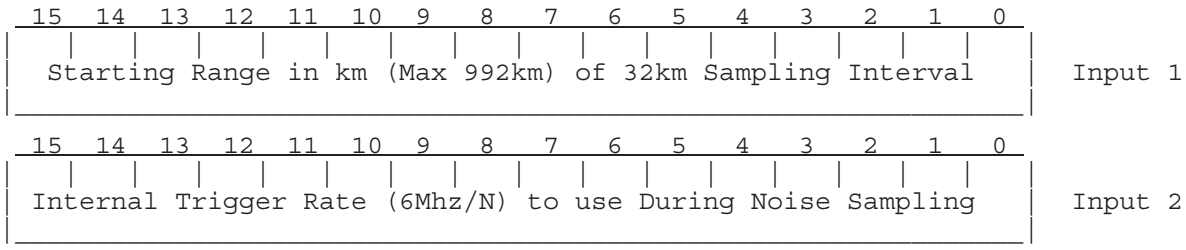
The SNOISE command bounds the requested starting range of the noise sampling interval. This is to insure that the noise samples fit within the specified PRT, and within the range mask hardware RAM. The RVP900 sets an error bit when an improper range is requested.

The SNOISE command should be reissued, 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 RVP900 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 power-up noise levels with the **Mt** setup command (see Section 4.2.5 Mt<n>— Triggers for Pulsewidth #n on page 117). The RVP900 does not automatically take a noise sample as part of its initialization procedure.

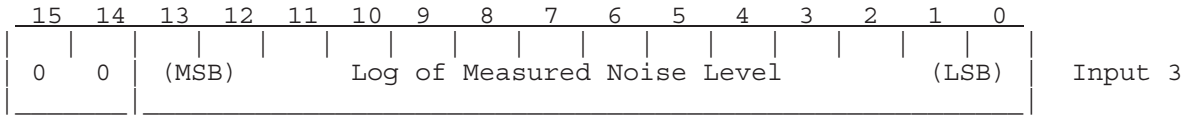
The measured offsets are stored internally for all subsequent uses inside the RVP900. The offset values may be inspected through the GPARM command, as may the current range and rate values themselves. Whenever the range or rate are changed, the user must make sure that the new trigger rate allows at least 32 km 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.



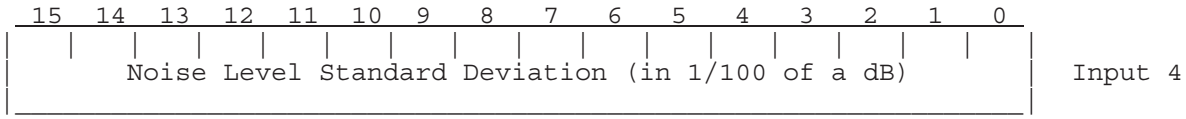
Rng	If 1, then the range in input word 1 is taken as the starting noise range for this and all subsequent SNOISE calls.
Rat	If 1, then the trigger rate in input word 2 is taken as the noise rate for this and all subsequent SNOISE calls.
Action	Specifies what action is carried out by the command.
0	Compute 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 to 50 are changed to reflect the new noise settings. If the current transmit pulse is a hybrid pulse, and XARGS arguments are supplied, then the additional arguments set the noise level for the second pulse.
2	Do not compute a noise sample, but rather, restore the powerup noise defaults.



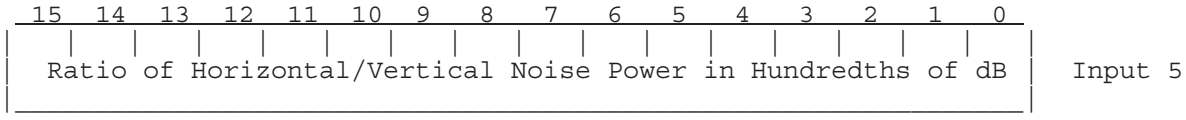
The following input words are optional, only if Action=1.



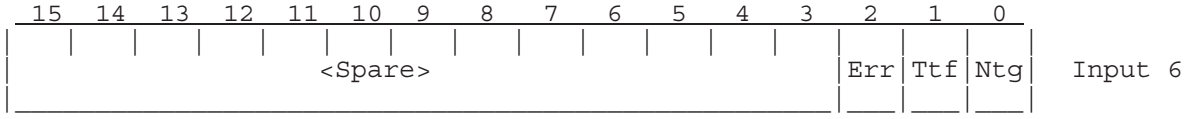
The is the same number as GPARM output 6. See the discussion in Sections Section 7.7 Initiate Processing (PROC) on page 275 and Section 7.9 Get Processor Parameters (GPARM) on page 290.



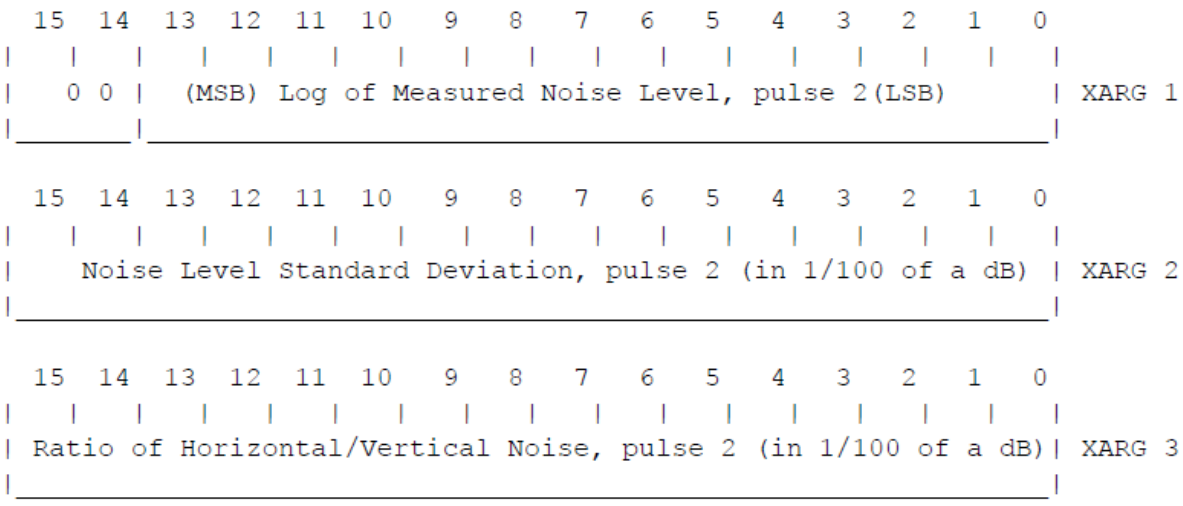
This is the same number as the GPARM output 49.



This is the same number as the GPARM output 50.



The following XARGS input words are optional, only if Action=1 and hybrid pulse.



These fault bits are outputted 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 RVP900 are described in Chapter 2, Introduction and Specifications, on page 17. It also describes the proper use and application of the RVP900 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 can be used to terminate the free running mode with no other consequences.

- **Time Series mode**—The processor acquires, processes, and outputs one ray of time series samples in response to each PROC command. Similar to Synchronous mode above. 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 RVP900 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 to change the velocity scale on any displays being generated. 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, Z_{dr}, and K_{dp} 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 generally misses weak signals. Vaisala recommends 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 0.01 dB (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. 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 RVP900 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:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
ARC	Z	T	V	W	ZDR	Unfold		KDP	0	1	0	0	1	1	0	Command

The Command word format for Free Running Doppler Mode is:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
ARC	Z	T	V	W	ZDR	Unfold		KDP	1	0	0	0	1	1	0	Command

Either of these may be augmented by an optional XARG word (see Section 7.20 Pass Auxiliary Arguments to Opcodes (XARGS) on page 320).

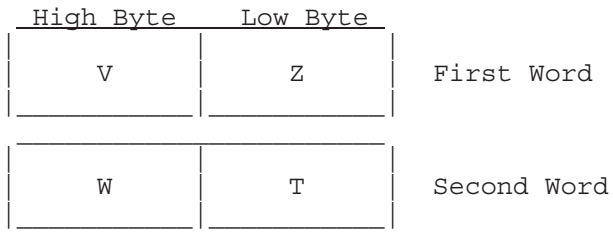
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
					HCL	FLG	(Tx Vert) Phi	Rho	Ldr	(Tx Horz) Phi	Rho	Ldr	SQI	RHV	PDP	XARG 1

Unfold Selects Dual-PRF 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 Section 7.3 Setup Operating Parameters (SOPRM) on page 259). The same SOPRM word configures the RVP900 for single or dual polarization. The later is required for K_{dp}, 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:

$$V_{m/sec} = V_{Nyquist} \times (N-128) / 127.5$$

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 2:3, 3:4, and 4:5 unfolding.

16-Bit Velocity Format—Mean velocity in meters per second (m/s) is computed from the unsigned word N as:

$$V_{m/sec} = (N-32768) / 100$$

The overall range is from 327.67 m/s to +327.66 m/s in one cm/s steps as follows:

0 : Indicates velocity data is not available at this range
1 : -327.67 m/s (towards the radar)
32768 : 0.00 m/s
65534 : +327.66 m/s (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:

$$W_{Nyquist} = N / 256$$

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 per second (m/s) is computed from the unsigned word N as:

$$W_{m/sec} = N / 100$$

The overall range is from 0.01 m/s to 655.34 m/s in one cm/s steps as follows:

0 : Indicates width data is not available at this range
1 : 0.01 m/s
65534 : 655.34 m/s
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 = (N–64)/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 = (N–32768) / 100

The overall range is from -327.67 dB to +327.66 dB in 1/100 dB 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 -7.935 dB to +7.935 dB in 1\16 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:

$$KDP \times \lambda = 0.25 \times 600^{\left[\frac{N-129}{126}\right]}$$

The conversion equation for negative values (codes from 1 to 127) is:

$$KDP \times \lambda = -0.25 \times 600^{\left[\frac{127-N}{126}\right]}$$

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 $?_{DP}$ data.

8-Bit $?_{DP}$ Format—The phase angle in degrees is computed on a 180-degree interval from the unsigned byte N as:

$$\Phi_{DP} (mod\ 180) = 180\ (N - 1) / 254$$

0 : Indicates no $?_{DP}$ data available at this range
1 : 0.00 deg
254 : 179.29 deg
255 : Reserved Code

16-Bit $?_{DP}$ Format The phase angle in degrees is computed on a 360-degree interval from the unsigned word N as:

$$\Phi_{DP} (mod\ 360) = 360\ (N - 1) / 65534$$

0 : Indicates no $?_{DP}$ data available at this range
1 : 0.00 deg
65534 : 359.995 deg
65535 : Reserved Code

RHV — Selects dual polarization correlation coefficient $?_{HV}$ data.

8-Bit $?_{HV}$ 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 $?_{HV}$ data available at this range
1 : 0.0000 (dimensionless)
2 : 0.0629
253 : 0.9980
254 : 1.0000
255 : Reserved Code

16-Bit $?_{HV}$ Format—The correlation coefficient is computed on the interval 0.0 to 1.0 linearly from the unsigned word N as:

0 : Indicates no $?_{HV}$ 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 ($?_{HV}$).

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 45 dB. The overall range is from -45.0 dB to +5.6 dB in 0.2 dB 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 ($?_{HV}$).

PHI — Selects the cross channel differential phase. This parameter uses the same 8-bit and 16-bit angular data formats as PDP ($?_{DP}$).

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

HCLASS — Hydrometeor Classification (HydroClass) parameter. There are several possible classification schemes. The choice is made in the *dpolapp_*-band.conf* file, where * is C (for C-band radars) and S (for S-band radars). The legacy Meteo classifications (up to IRIS/RDA 8.12.6) are:
0 No measurement available
1 Non-meteorological target
2 Rain
3 Wet snow
4 Snow
5 Graupel
6 Hail

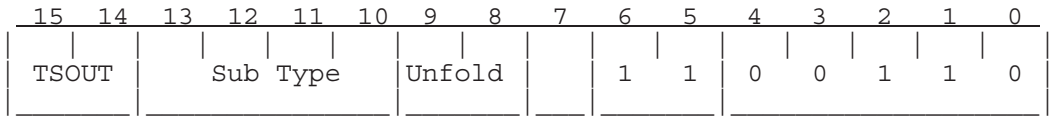
Higher bits of the HCLASS data fields may contain results from further methods of clasification, see *sig_data_types.h* and HCLASS data description in the *IRIS Programmers Manual*.

SNR — Signal-to-Noise ratio on the primary (horizontal) channel. Uses the same storage format as Z

Ta — Total power in the alternative polarization receive channel (usually vertical). Uses the same storage format as T.

Za — Clutter corrected reflectivity in the alternative polarization receive channel (usually vertical). Uses the same storage format as Z.

The Command word format for Time Series Mode is:



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 "Sub Type" are:

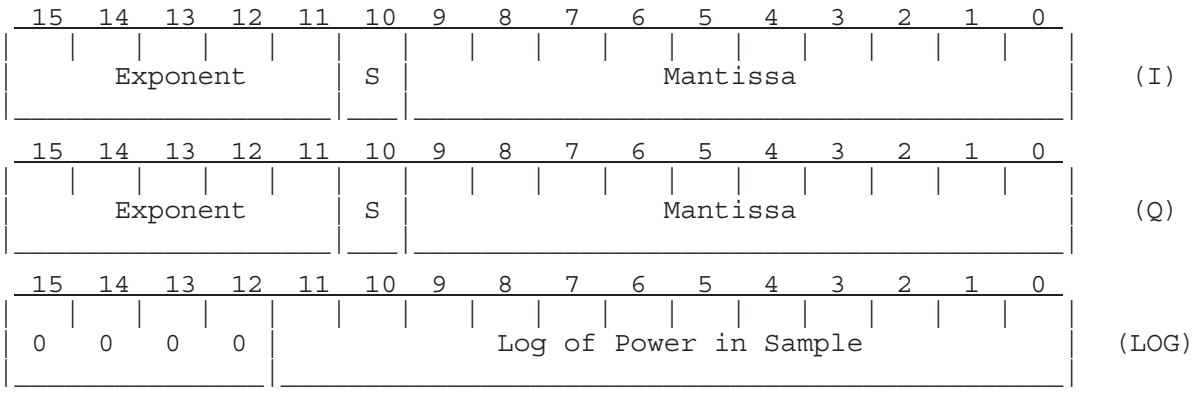
0: Raw First Trip	4: Raw Second Trip
1: Whitened First Trip	5: Whitened Second Trip
2: Cleaned First Trip	6: Cleaned Second Trip
3: Final First Trip	7: Final Second Trip

When the TSOUT bits select "Time Series" then, a further choice may be needed to select the time series from the first or second pulse when using the Hybrid Pulse Compression scheme. For the Random Phase major mode the possible values of "Sub Type" are:

0: Main pulse time series
1: Second pulse time series (if hybrid pulse)

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:



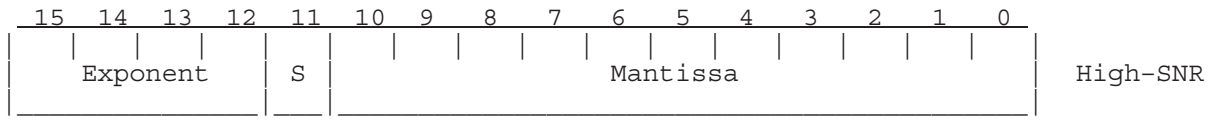
To convert these "legacy format" floating I and Q samples to voltages: First create a 12-bit signed integer in which bits zero through nine are copied from the Mantissa field, and bits ten and eleven are either 01 or 10 depending on whether S is 0 or 1. Then, multiply this number by 2**(exponent-40), 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 190 dB. The large dynamic range is necessary to cover the full range of data. In summary:

$$Voltage = V_{MAX} \times (Sign, Mantissa) \times 2^{[Exponent - 40]}$$

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 6 dB improvement in SNR, that is, a commensurate improvement in sub-clutter visibility of -78 dB versus -72 dB.



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 66 dB to 72 dB and we can assign a mean value of 69 dB. Another 9 dB 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 3 dB 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 6 dB (one more bit) to the overall dynamic range and large-signal SNR. In the floating format we really only gain 3 dB (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 time series 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 IFDR, +6.0 dBm for 14-bit IFDR

V_{MAX} = 0.5309 Volts for 12-bit IFDR, 0.6310 V for 14-bit IFDR

Slope = "Log Power Slope" word 3 of SOPRM command. 0.03 recommended.

For backwards compatibility, the RVP900 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:

High Byte	Low Byte	
Q Sample	I Sample	First Word
Zero	Log Power	Second Word

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 (P_{MAX}). Thus, the spectral output terms are almost always negative.

The time series that are output by the RVP900 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" results 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" still produces 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 RVP900. 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 RVP900 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 Section 7.9 Get Processor Parameters (GPARM) on page 290) 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.

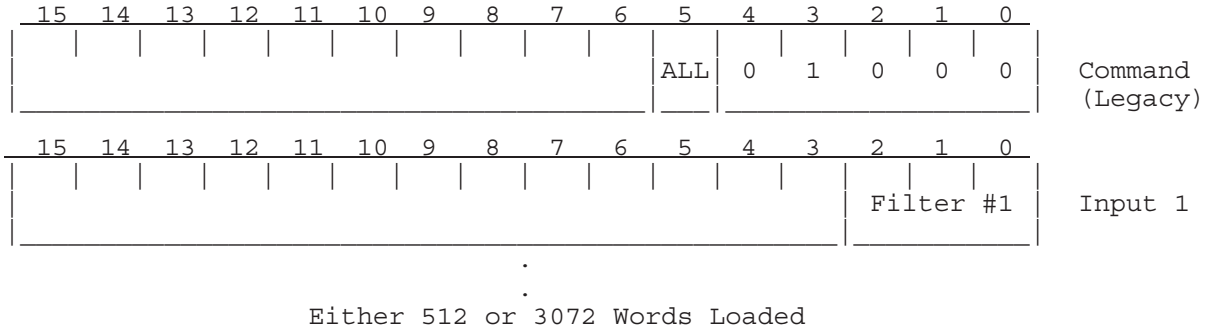
7.8 Load Clutter Filter Flags (LFILT)

A special feature of the RVP9 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 Section 7.2 Load Range Mask (LRMSK) on page 257) 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 L3K bit is set in the Command, then 3072 words are loaded, otherwise only 512 words are loaded. In both cases the last filter choice is replicated for all bins further in range.



Enhanced Version

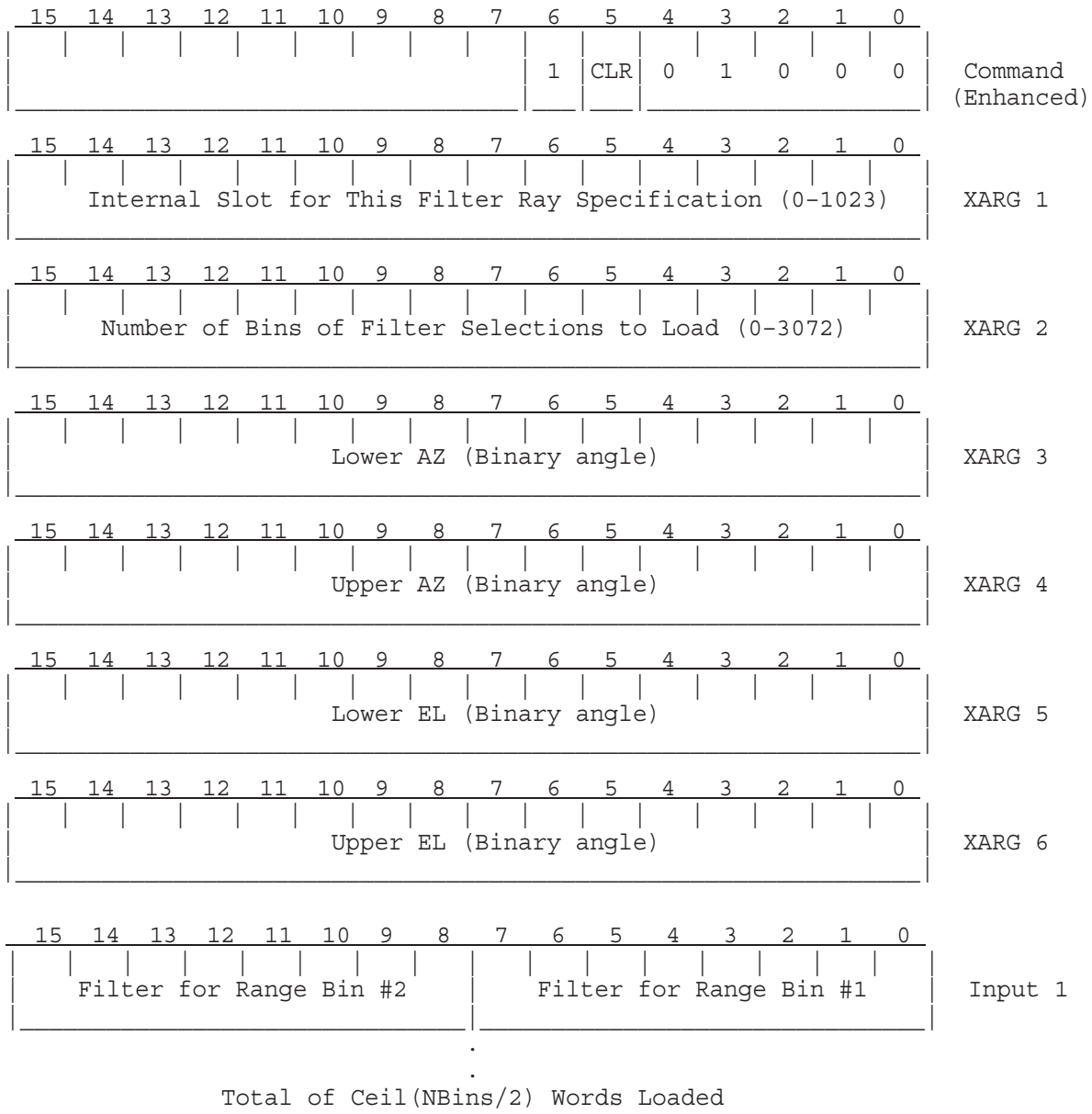
The RVP900 supports an enhanced version of the LFILT command that provides "Clutter Maps", that is, much greater flexibility by allowing filter choices to depend on antenna angle as well as range. This lets you specify a 2D or even 3D table of clutter filter selections that are dynamically selected during live data processing.

The RVP900 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 RVP900 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 RVP900 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 14 on page 290.

Table 14 RVP900 Status Output Words

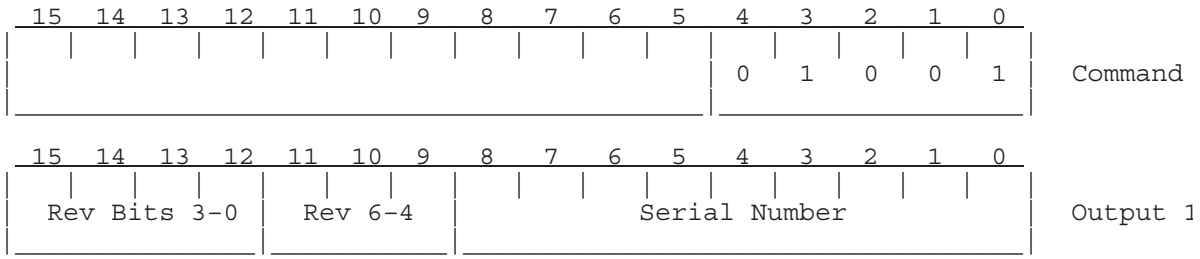
Word	Description
1	Revision/Serial number

Table 14 RVP900 Status Output Words (Continued)

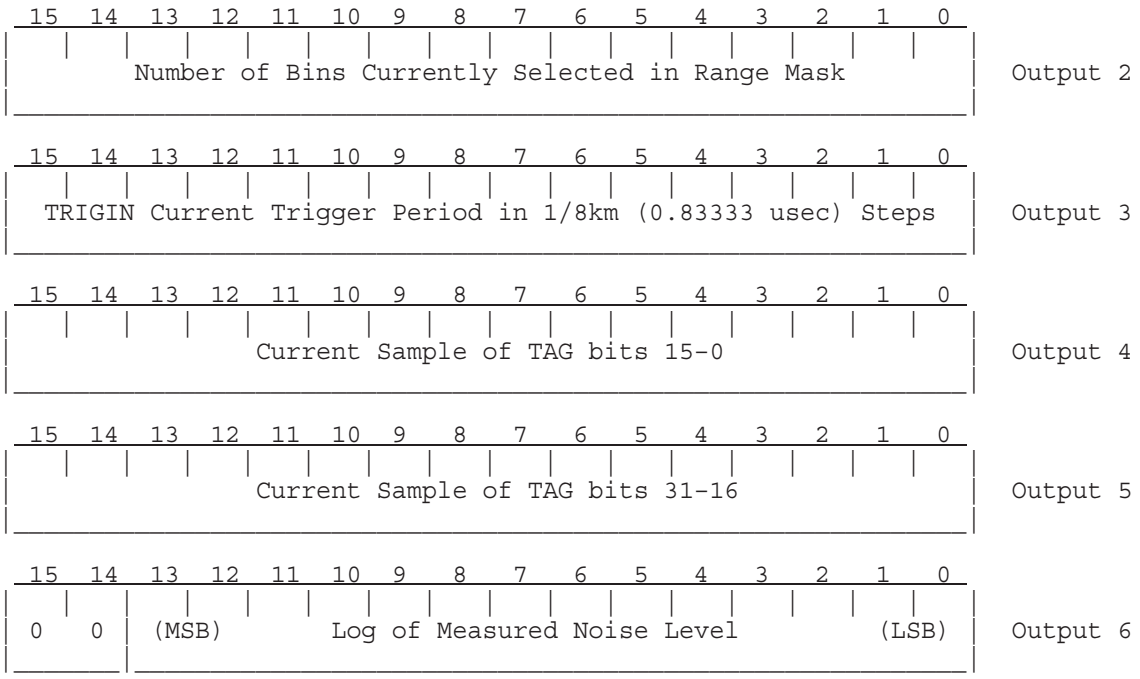
Word	Description
2	Number of Range Bins
3	Current trigger period
4	Current TAG00 - TAG15
5	Current TAG16 - TAG31
6	Log of Measured Noise Level
7	"I" Channel DC Offset
8	"Q" Channel DC Offset
9	Latched Processor Status
10	Immediate Status Word #1
11	Diagnostic Register A
12	Diagnostic Register B
13	Number 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
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)

Table 14 RVP900 Status Output Words (Continued)

Word	Description
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

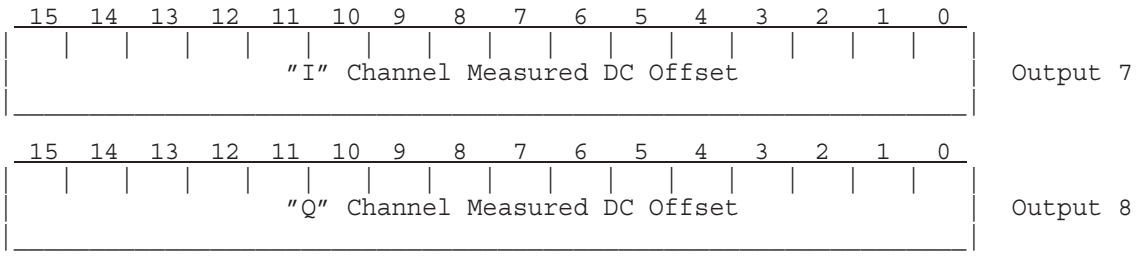


The revision and serial numbers of the particular RVP900 board are accessible here. This information is useful when computer software is being designed to handle a variety of signal processor revisions. The revision number is seven bits total; four of which are still in the high four bits of the word for compatibility with an older format.

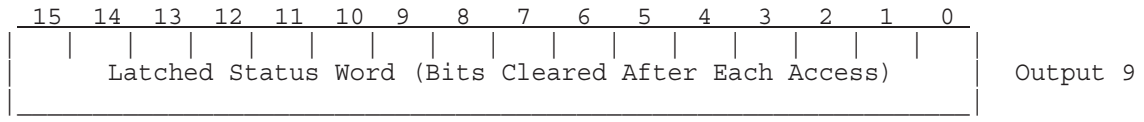


This value is scaled four times higher than the time series LOG format (see the discussion in Section 7.7 Initiate Processing (PROC) on page 275). To convert to dBm, use the equation:

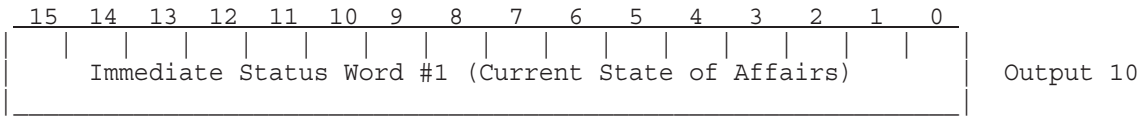
$$dBm = P_{MAX} + Slope \times [(Value/4) - 3584]$$



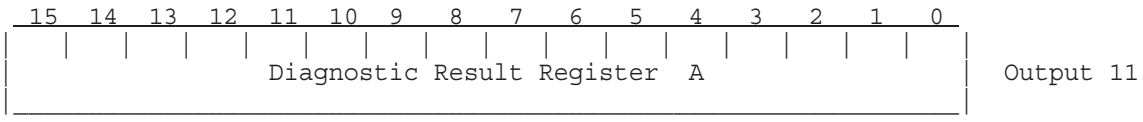
These two words convey the measured "I" and "Q" DC offsets from the last noise sample. The output format is either signed 16-bit values in which ±32767 represent ±1.0 (legacy format), or packed time series values using the High-SNR encoding format. Bit-9 of GPARM Word-59 shows which format to use.



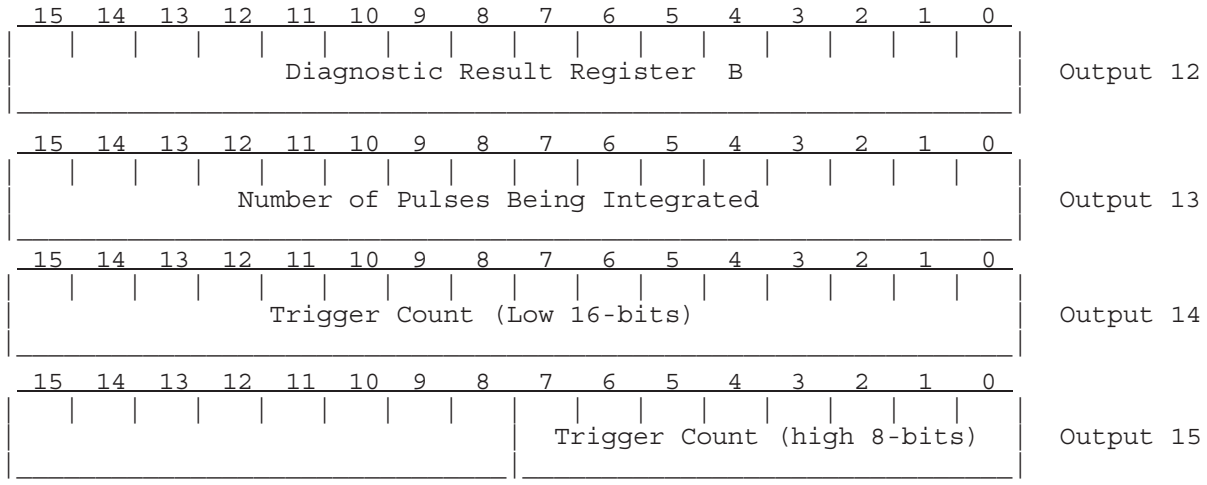
Bit 0	No Trigger during noise measurement
Bit 1	Trigger too fast during noise measurement, that is, some of the noise sample bins were positioned past the trigger range
Bit 2	No trigger during PROC command
Bit 3	PRT varied by more than 10 ?sec within portions of a processing interval that should have 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	Error 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



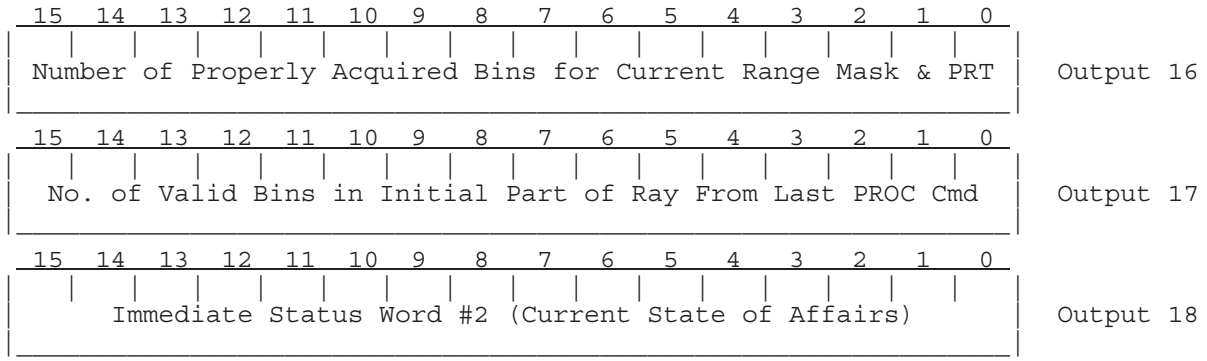
Bit 0	No trigger, or, more than 50 ms since last trigger.
Bit 1	Error in loading trigger angle table (See LSYNC Command)
Bit 2	PWINFO command is disabled.
Bit 3	Angle sync input is BCD (Else binary angle)
Bit 4	Angle sync is on elevation axis (Else azimuth axis)
Bit 5	Angle sync is enabled
Bit 6	Angle 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 RVP900/PROC compute processes minus one
Bit 14	DSP supports Power Spectrum output



Bit 0	Error loading config/setup files
Bit 1	Error attaching to antenna library
Bit 2	Problem when forking compute processes
Bit 3	Signals raised during startup
Bit 4	RVP900 running without ’root’ privileges
Bit 5	Problem creating daemon process
Bit 6	Inconsistent setup values detected
Bit 7	Ethernet MTU does not support requested frame size
Bit 8	Processor is running in Test/Debug mode
Bit 9	Insufficient kernel buffering for incoming UDP packets



The trigger count is a running tally of the number of triggers received by the RVP900 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 IFDR systems: Bits 4,5,7, and 11 are set if either IFDR fails:

Bit 4 Unused

Bit 5 Unused

Bit 7 IFDR 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

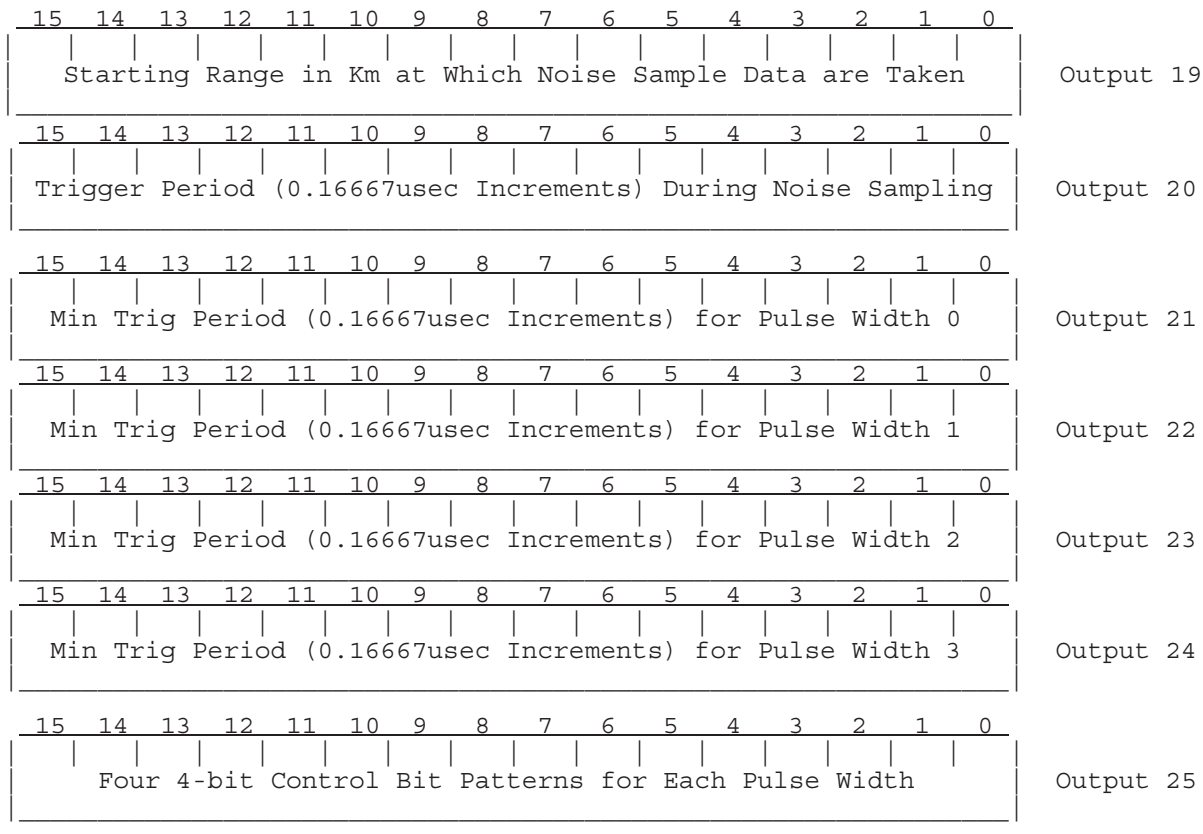
Bit 11 IFDR test switches are not in their normal operating position

Bit 12 Set according to whether the RVP900 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.

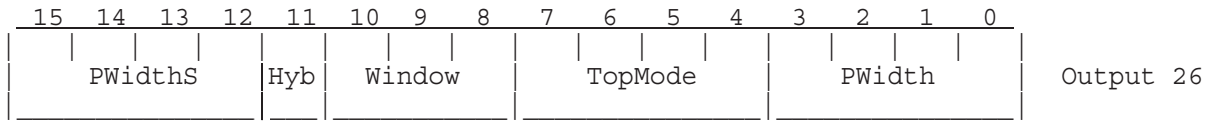
Bit 13 Missing signal at IFDR #1 Burst Input

Bit 14 Reserved (zero)

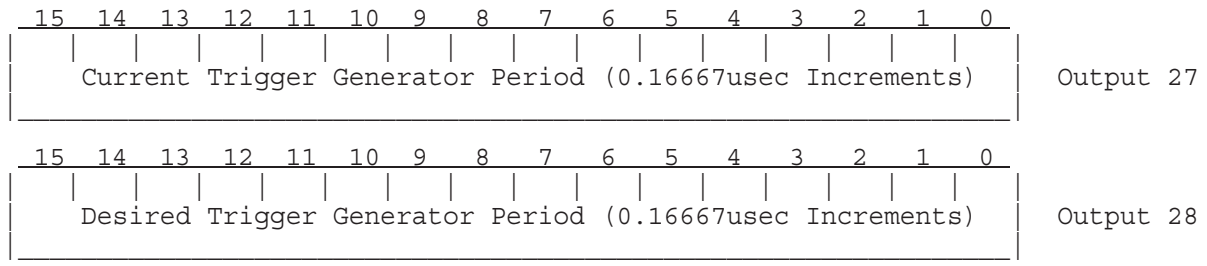
Bit 15 Set when valid burst power is detected, but the center-of-mass lies outside of the aperture sub-window that defines the portion of the pulse used for AFC analysis. This error bit effectively flags when the burst pulse has drifted out of its optimal placement within the sampling window.



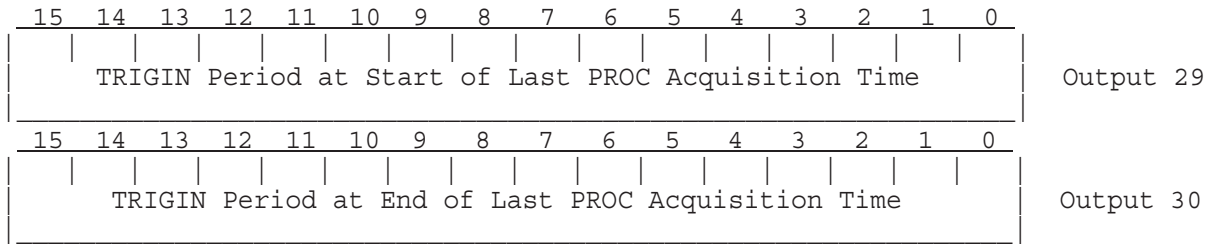
See PWINFO command, input word #1, for definition of these bits.



PWidth	Currently selected radar pulse width
TopMode	Major Mode (See SOPRM Input #9)
Window	Spectral Window Choice (See SOPRM Input #10)
PWidthS	Pulse width of second pulse in hybrid transmit waveform
Hyb	Bit indicating second pulse in use in hybrid transmit waveform



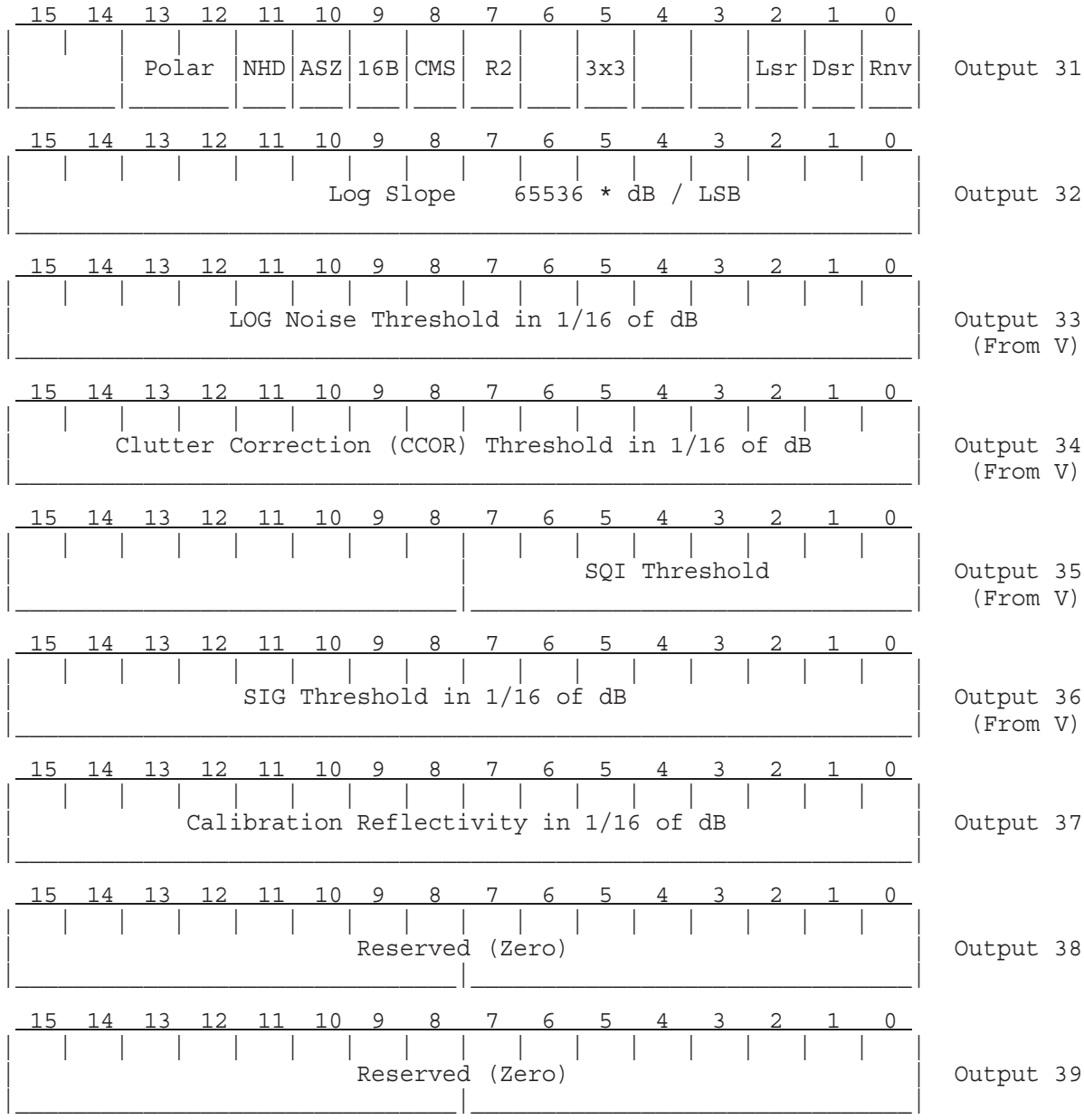
The desired trigger generator rate is that which was selected in the most recently issued SETPWF command (or power-up rate if SETPWF was never issued). The current rate may be different from the desired rate due to bounding against limits for the current pulse width, or being in an odd ray cycle during dual-PRT processing. The measured PRTs are forced to 0xFFFF (the maximum unsigned value) whenever the external trigger is expected, but missing.

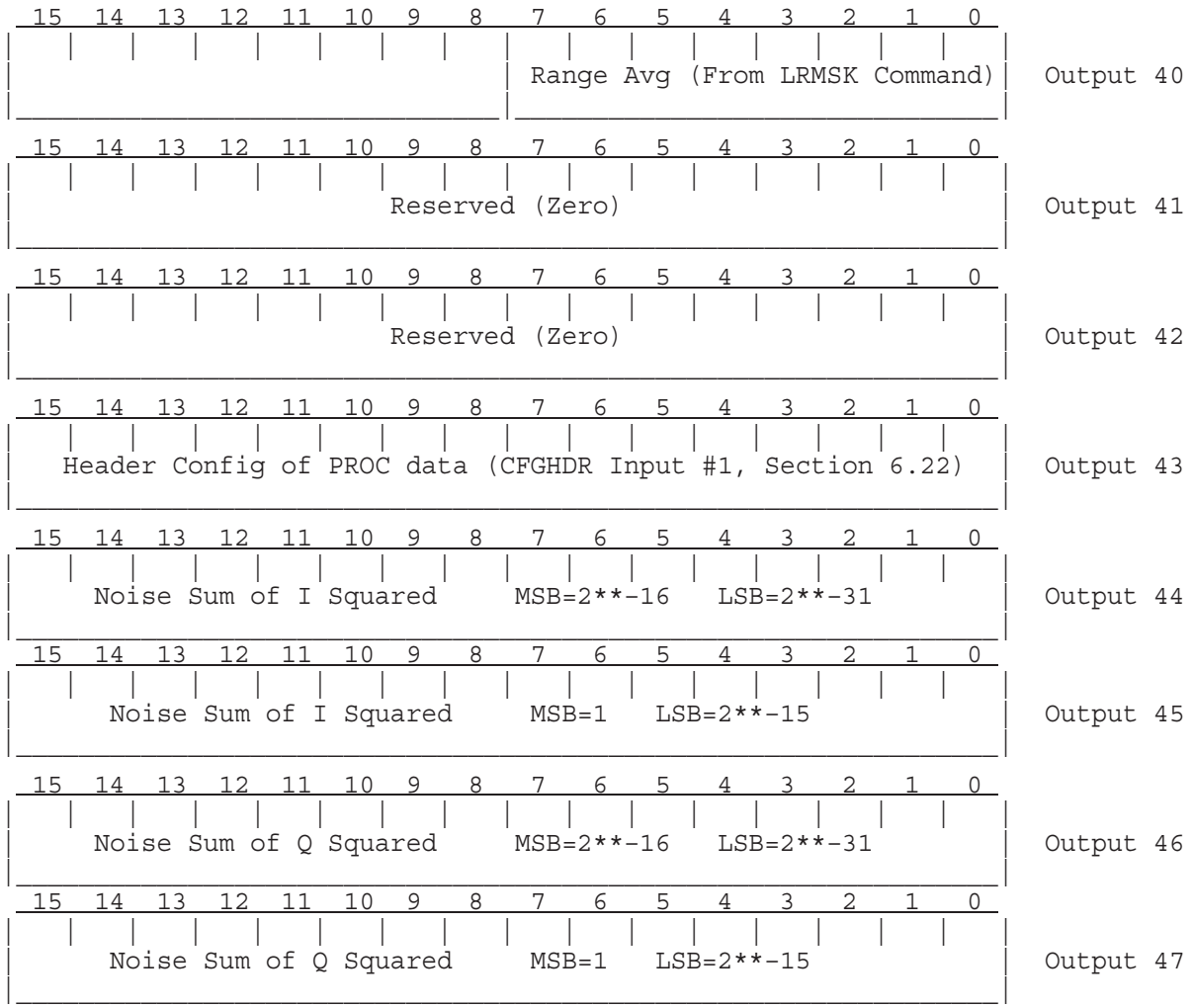


The PRTs from the start and end of the last ray are the actual measured values whenever possible, that is, when non-simulated data are being processed, and we either have an external trigger, or an internal trigger that

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 Section 7.3 Setup Operating Parameters (SOPRM) on page 259 for additional notes on each of these parameters. Since the threshold levels for each data parameter can be different (see Section 7.29 Set Individual Thresholds (THRESH) on page 329), words 33 through 36 are taken from the velocity parameter.





To compute the noise power in dBm from Words 44 through 47, first calculate:

$$N_I = (Word\ 45) \times 2^{-15} + (Word\ 44) \times 2^{-31}$$

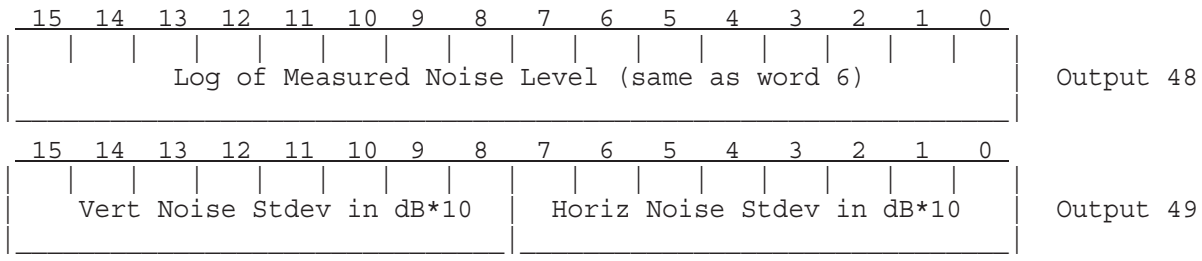
$$N_Q = (Word\ 47) \times 2^{-15} + (Word\ 46) \times 2^{-31}$$

from which we obtain:

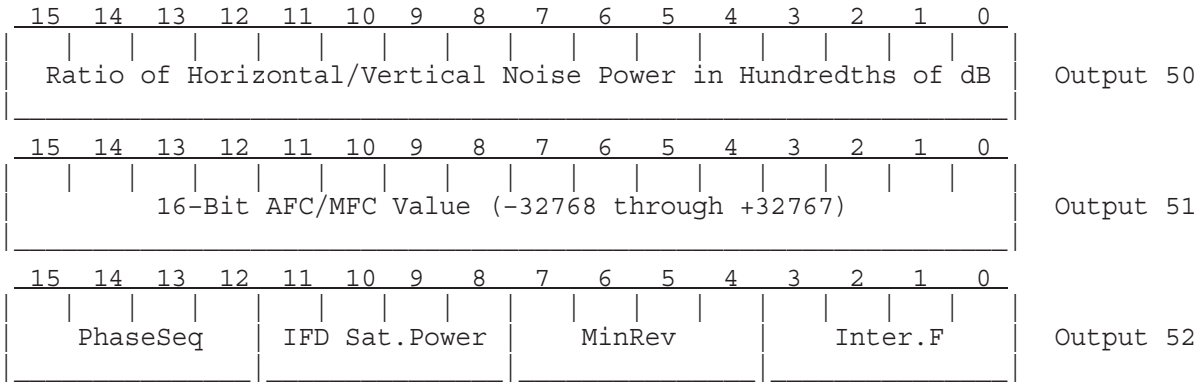
$$dBm = P_{MAX} + 10\log_{10}(N_I + N_Q) - 3dB$$

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

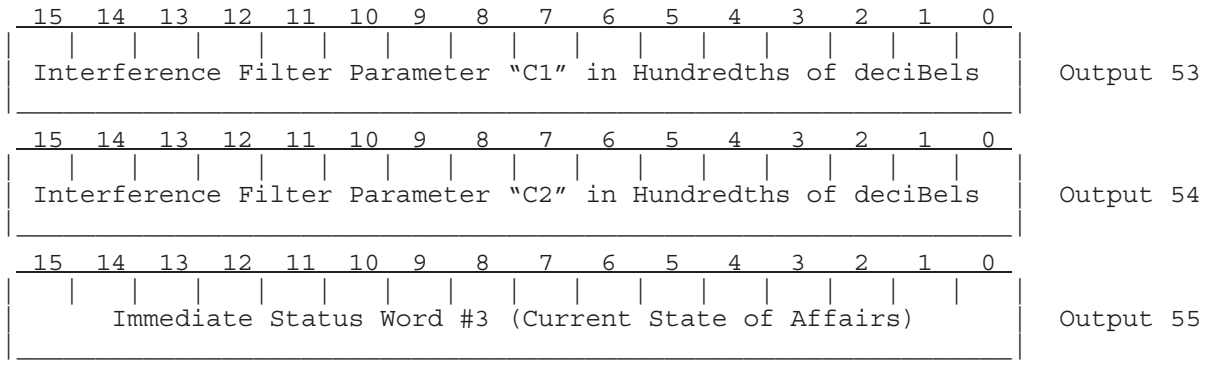
in the presence of a strong test signal. Since "I" and "Q" are inherently balanced in the RVP900, these output words are no longer of much value.



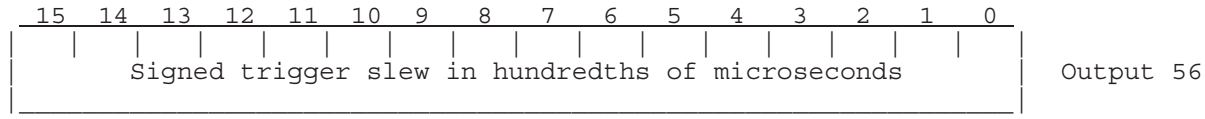
The noise standard deviations for each receive channel are normalized to the mean power. The values reported here hover around 0 dB for ordinary exponentially distributed noise in which the standard deviation scales directly with the mean.



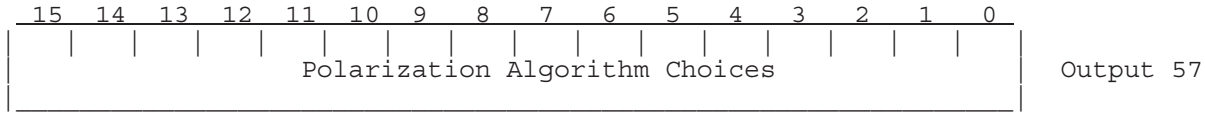
Inter.F	Specifies which interference filter is running. Zero means "none"; see Section 6.1.5 Interference Filter on page 189 for a description of the interference filter algorithms.
MinRev	Minor revision level of the RVP900 code that is currently running
IFDR Sat.Power (P_{MAX})	Input power required to saturate the IF-Input A/D converter for the IFDR that is currently attached 0: +4.5 dBm 1: +6.0 dBm
PhaseSeq	Tx Phase modulation sequence (see Section 7.27 Configure Phase Modulation (CFGPHZ) on page 327)



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	Unused
Bits 8-11	User-defined Major Modes 1 through 4 are supported



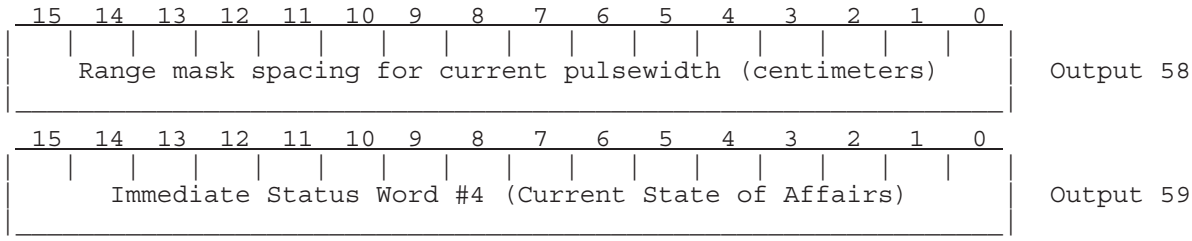
This is the same format that is used by the SETSLEW command to set the current trigger slew (see Section 7.25 Set Trigger Timing Slew (SETSLEW) on page 326).



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 Z_{dr} 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 High-SNR packed (I,Q) format

Bit 5 Trigger sequence truncated due to insufficient pattern memory

Bit 6 Time series data source is external to the RVP900

Bit 7 WSR88D "Batch" mode is supported

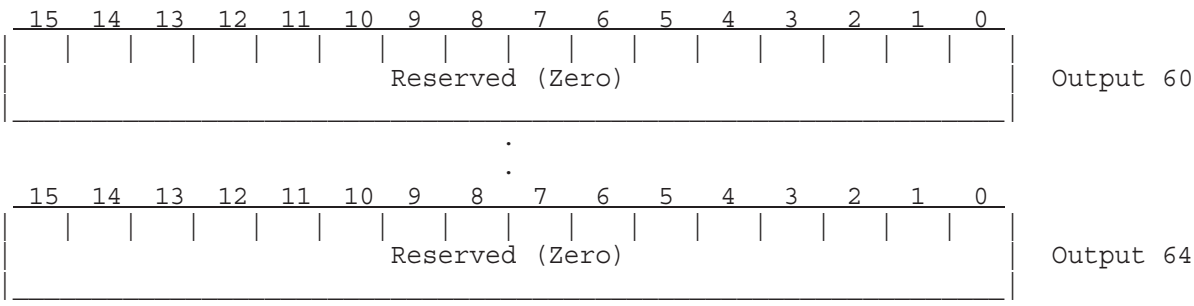
Bit 8 Major mode refuses to use external trigger

Bit 9 GPARM outputs #7 and #8 use Hi-SNR format, else linear

Bit 10 Receiver protection fault

Bit 11 IFDR 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

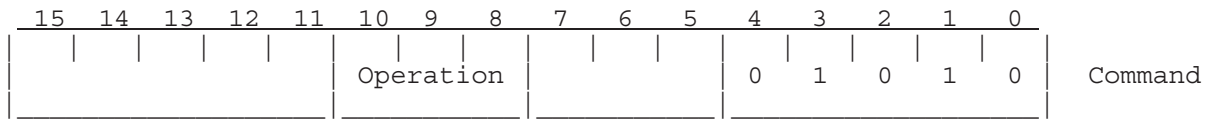


7.10 Load Simulated Time Series Data (LSIMUL)

This command is provided as a diagnostic for proper functioning of the RVP900 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 RVP900 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 RVP900 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 RVP900 to replicate them internally.

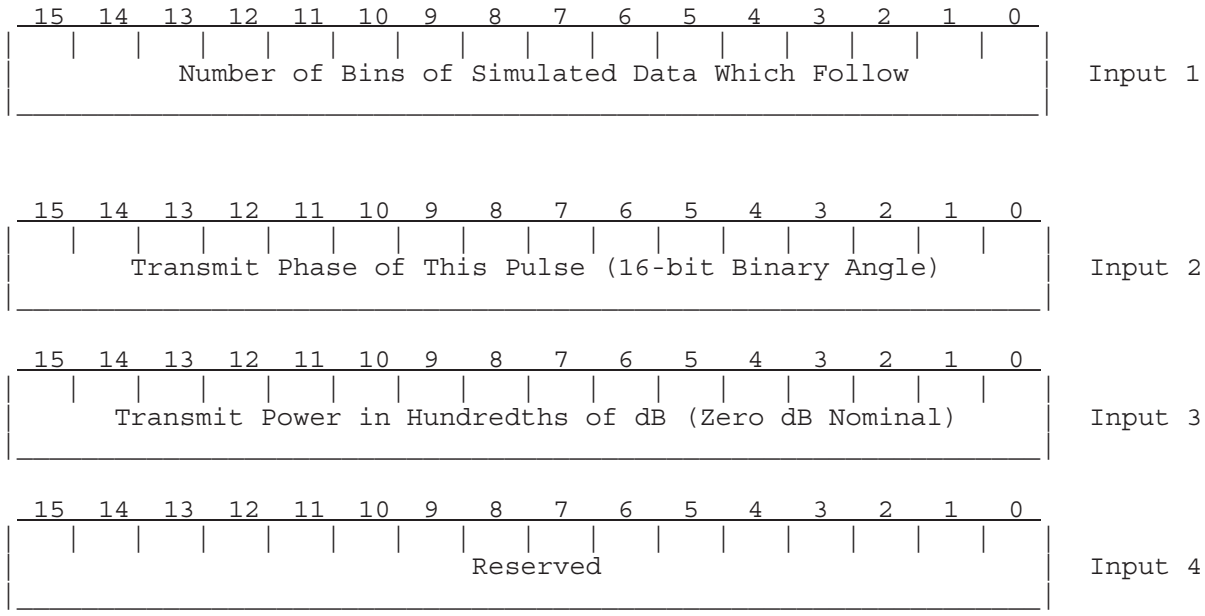


The available operations are:

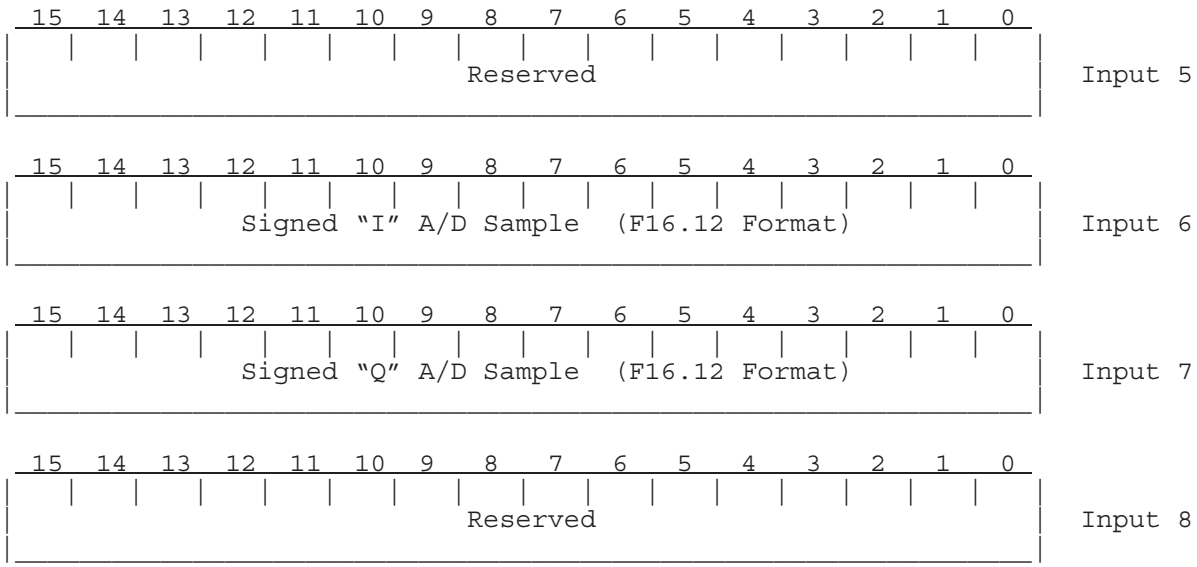
0 Disable the use of simulated data. RVP900 returns to acquisition and processing of live data from the A/D converters.

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 RVP900 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.

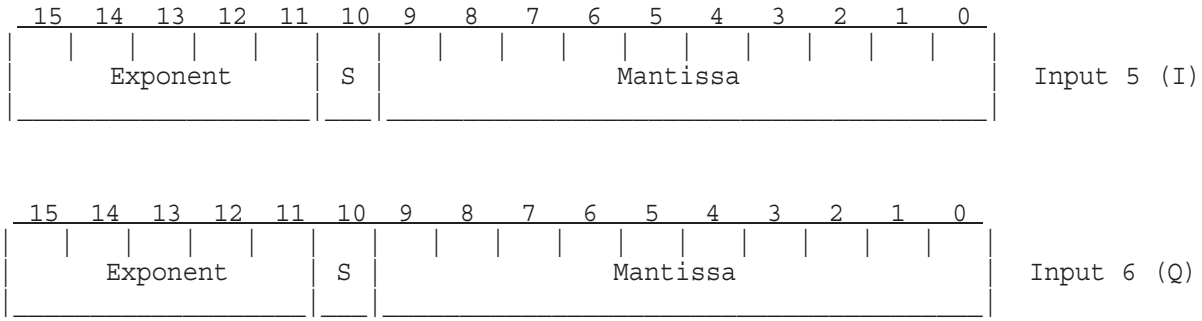
2 or 3 Load one pulse of data samples beginning with the following 4-word header, and continuing with an array of items each representing a single instantaneous sample of (I,Q) data. You may specify one or more bins to be loaded, and the RVP900 will replicate these data as necessary in order to fill 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-RVP900) 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.

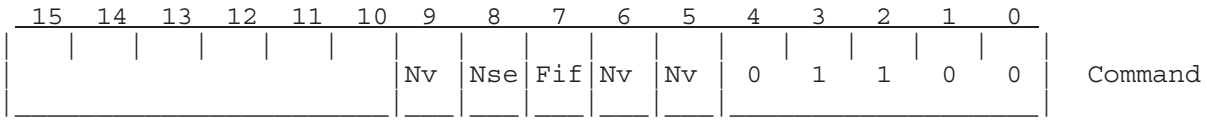


In Format #3 (new in the RVP900) each bin within the pulse is represented by two 16-bit floating point words having the exact same format as the packed (I,Q) time-series data that are output by the PROC command (See Section 7.7 Initiate Processing (PROC) on page 275). Compared to the legacy 4-word format, this 2-word format uses half the I/O bandwidth, has superior 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 RVP900 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 Section 4.2.5 Mt<n>— Triggers for Pulsewidth #n on page 117).

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 RVP900 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 Section 4.2.1 Mc — Top Level Configuration on page 104), because the interactive trigger setup procedure described in Section 5.3 Pb — Plot Burst Pulse Timing on page 143 is the preferred method of defining all RVP900 triggers and timing. TRIGWF should not be used in any new code applications that drive the RVP900.

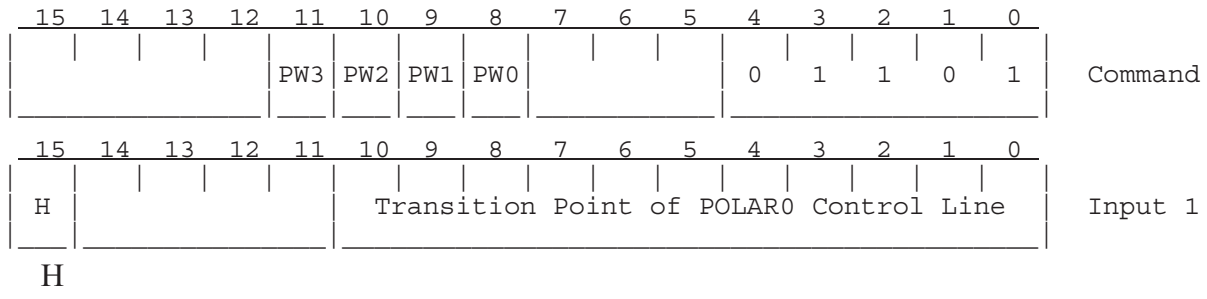
The RVP900 has a built-in trigger generator that can synthesize six independent digital output waveforms, each having arbitrary shape and being active anywhere in a window centered around zero-range. The six trigger outputs can be defined by a 2048-word by 6-bit table which is loaded from the user computer. 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 RVP900 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.

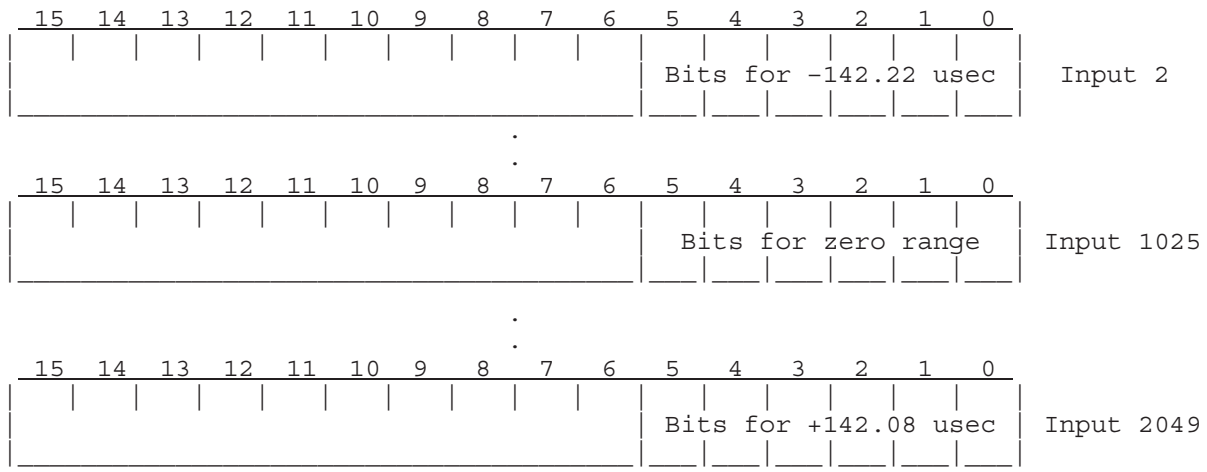
Trigger waveforms are loaded using the TRIGWF command. Four bits in the command word (PW0 through PW3) select which pulse widths will receive the new waveforms. On power-up, all four pulse widths are initialized to user-selected waveforms.

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 RVP900. 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.



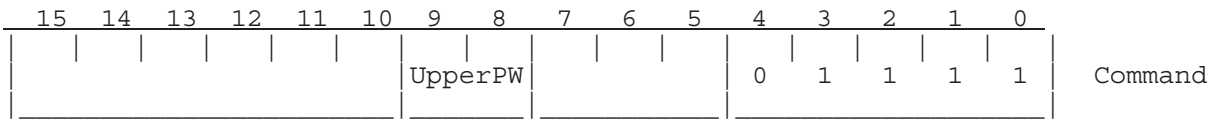
7.13 Define Pulse Width Control and PRT Limits (PWINFO)

The RVP900 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 pulse width 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.

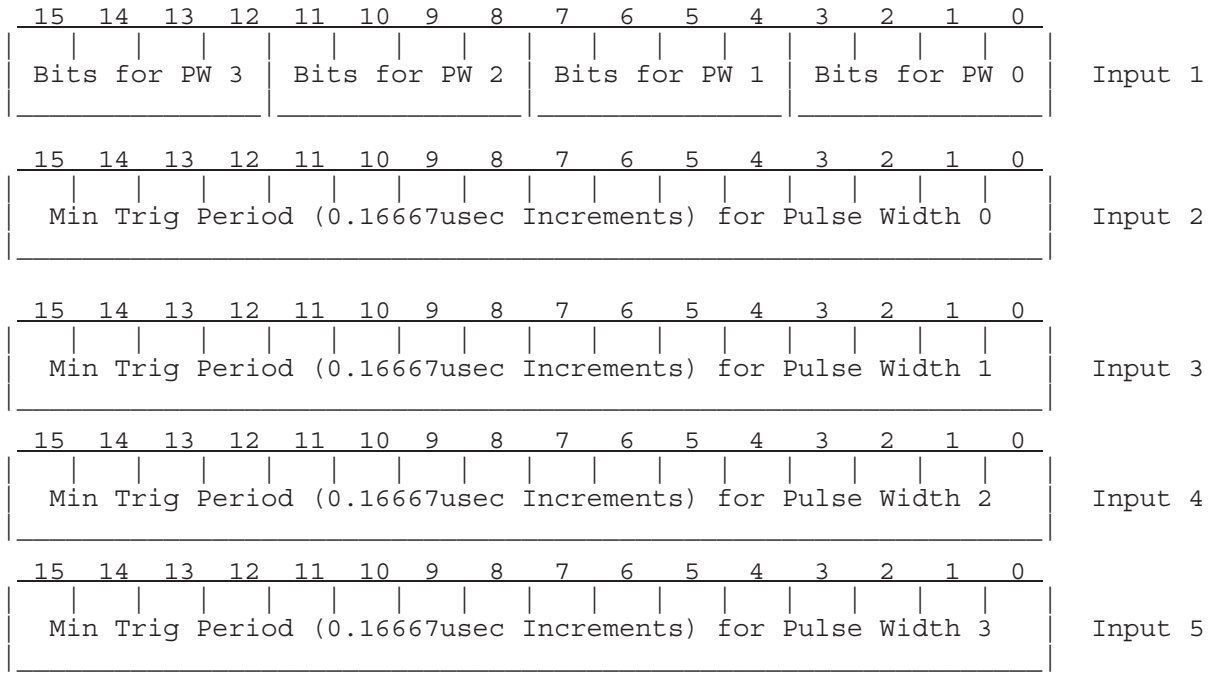
The RVP900 drives four TTL output lines (PWBW0 – 3) which are intended to control the radar pulse/bandwidth hardware. Typically this control is via relays or solid-state switches in the transmitter and receiver. The user decides what state the four lines assume for each pulse width code. This is done 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 pulse width 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 PRTs 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 RVP900 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.

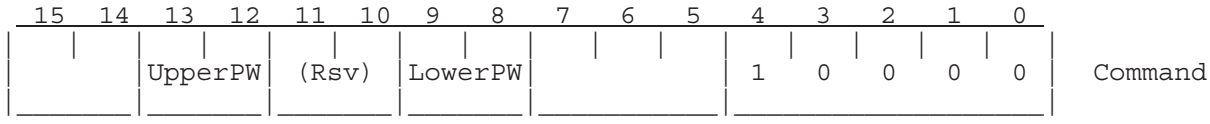


UpperPW Upper two bits of the four 4-bit pulse widths being defined.



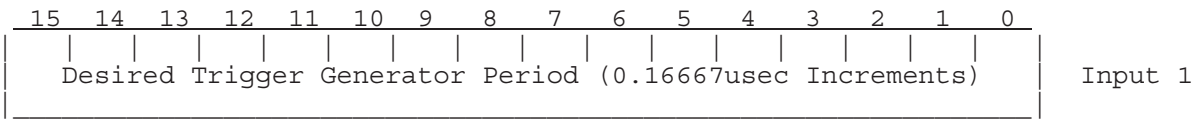
7.14 Set Pulse Width and PRF (SETPWF)

This command selects the pulse width 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.

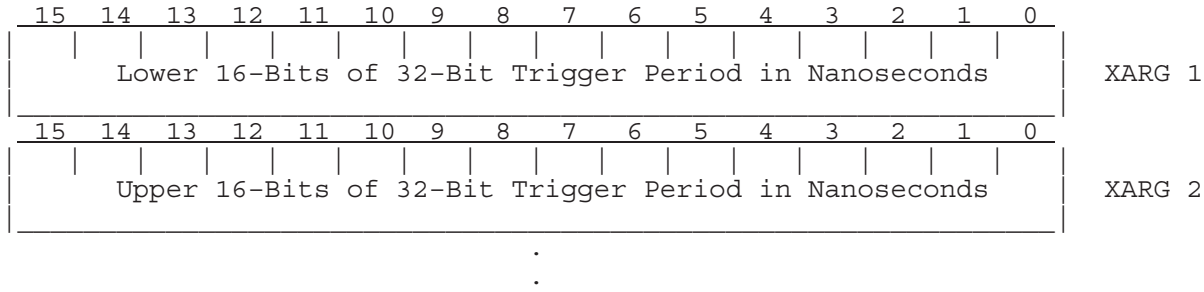


UpperPW Upper two bits of overall 4-bit pulse width selection

LowerPW Lower two bits of overall 4-bit pulse width 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 RVP900 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 RVP900 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. On power-up the angle table is initialized to 360 values corresponding to half-integer-valued degrees from 0.5 to 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 RVP900 will automatically produce an output ray after one second of waiting, even if no trigger angles have been crossed. This is to avoid time-outs with the host computer when the antenna is not moving at all.

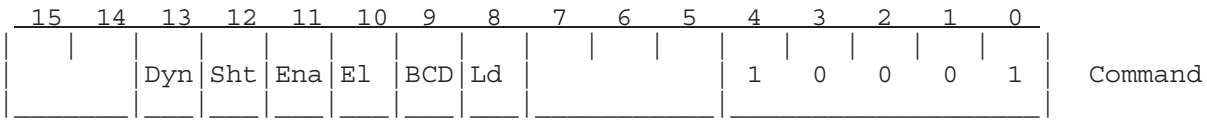
To use the synchronization mode, the trigger angle table is first loaded using the LSYNC command. The user chooses the number of table entries and then writes the required number of words to the RVP900. The angles must be supplied in a clockwise strictly increasing order, and they must neither reach 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 Section 7.9 Get Processor Parameters (GPARM) on page 290) 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.

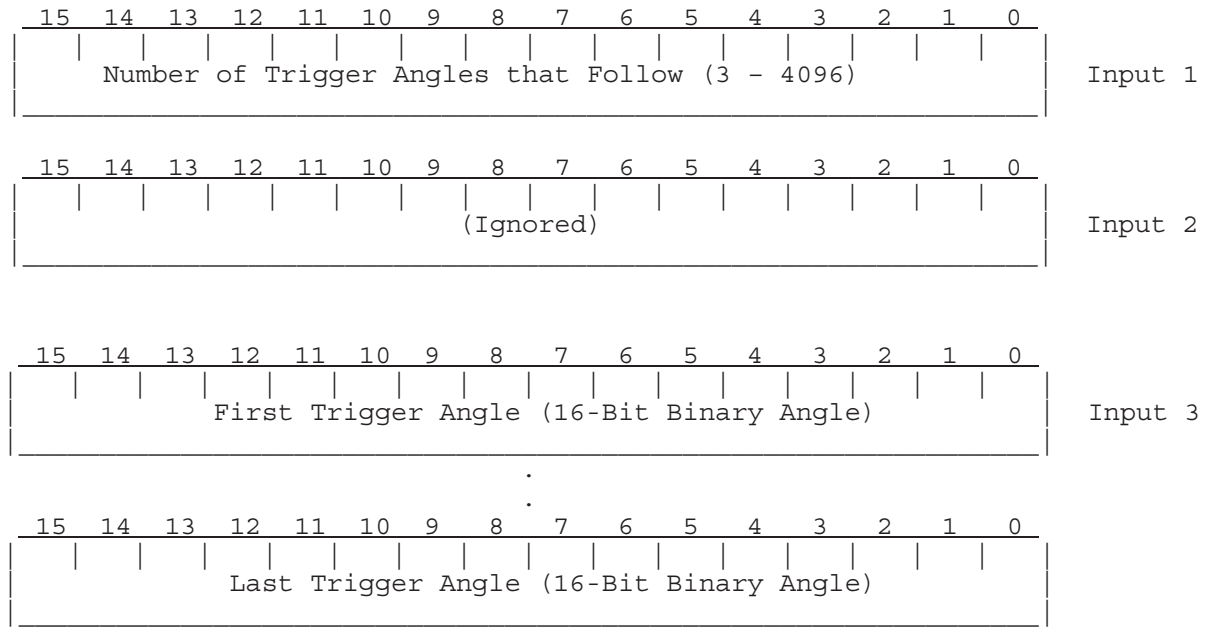
The angle sync algorithm is dynamic and works as follows. Each ray begins immediately upon the user's request, or upon completion of the previous ray when in continuous processing mode. At the start of the ray, the RVP900 finds the pair of sync angles that enclose the previous trigger angle. The current 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

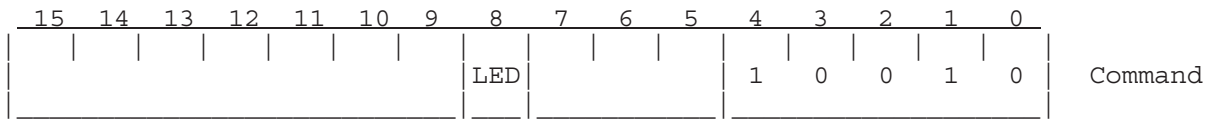


Dyn	If set, then synchronize the endpoints of each ray but let their angular width vary by whatever is required to collect the SOPRM number of pulses. If clear, then the angular width of each ray is fixed (between successive table entries) by adjusting the pulse count and reinterpreting the SOPRM sample 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:



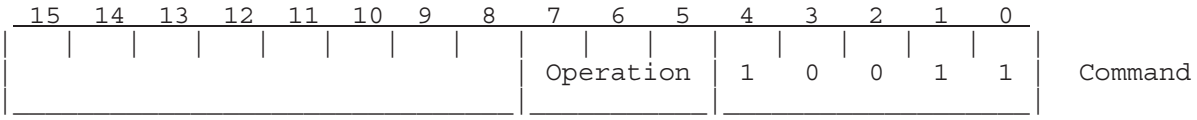
7.16 Set/Clear User LED (SLED)

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 RVP900 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 RVP900 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:

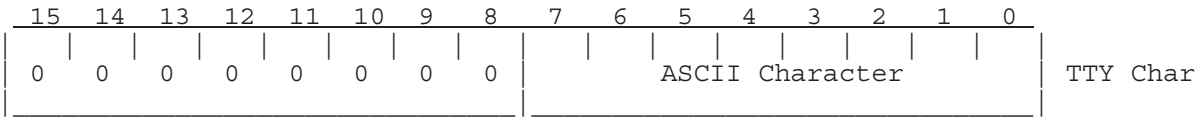
0 Sends the ASCII character in the upper byte of the word to the RVP900 as if it had been typed on the setup TTY keyboard.

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.

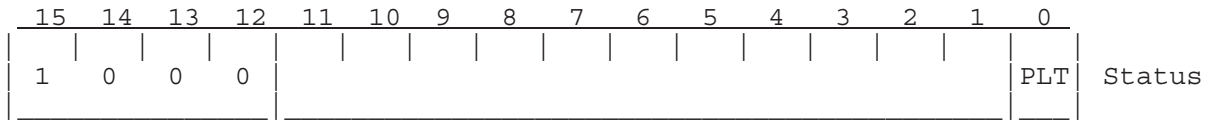
2 Disable the scope plotting output data.

Any of the following types of data may be output by the RVP900 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 RVP900 are output in the low byte of the word, with the upper byte set to zeros.

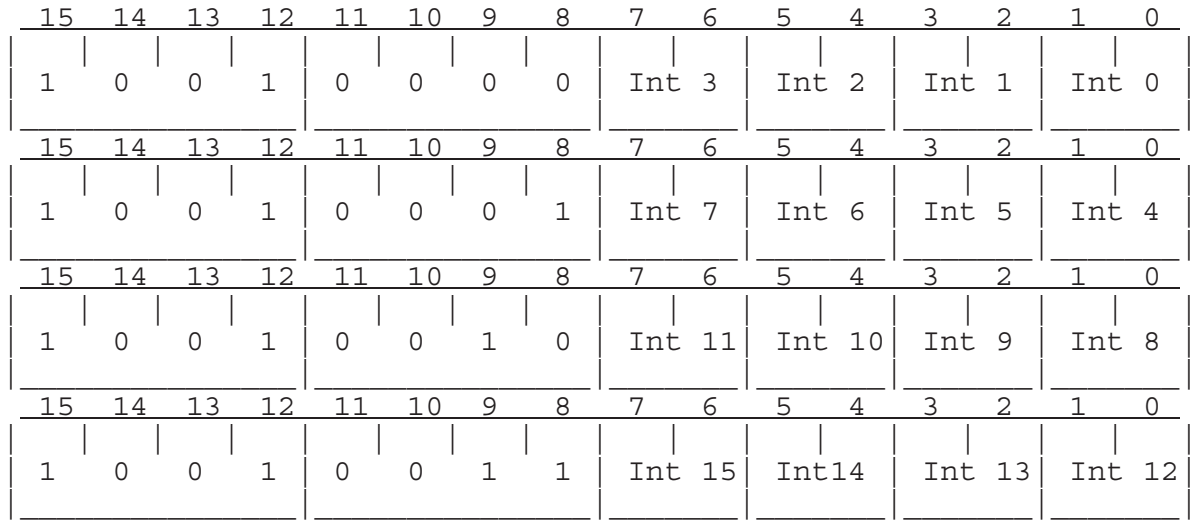


The status of the plotting modes is given in the following word.

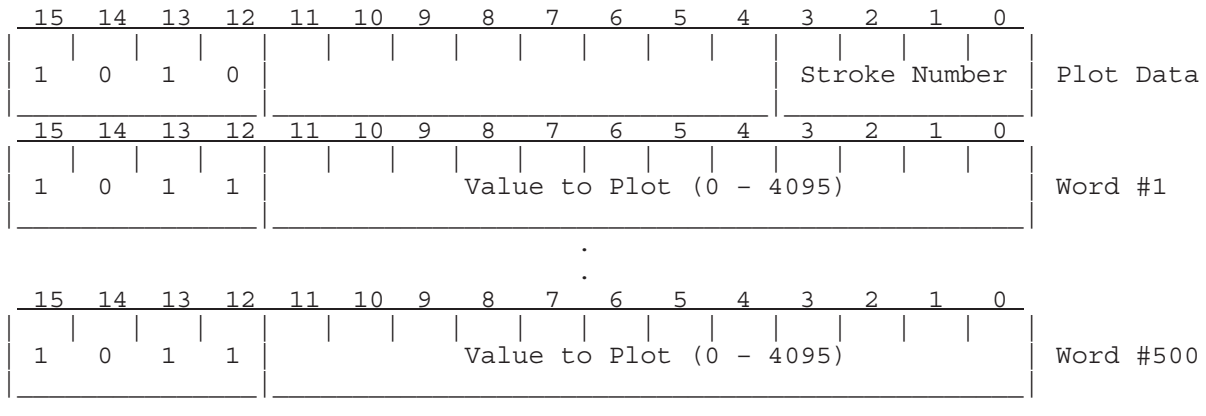


PLT Indicates that a scope plot is being drawn now.

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.

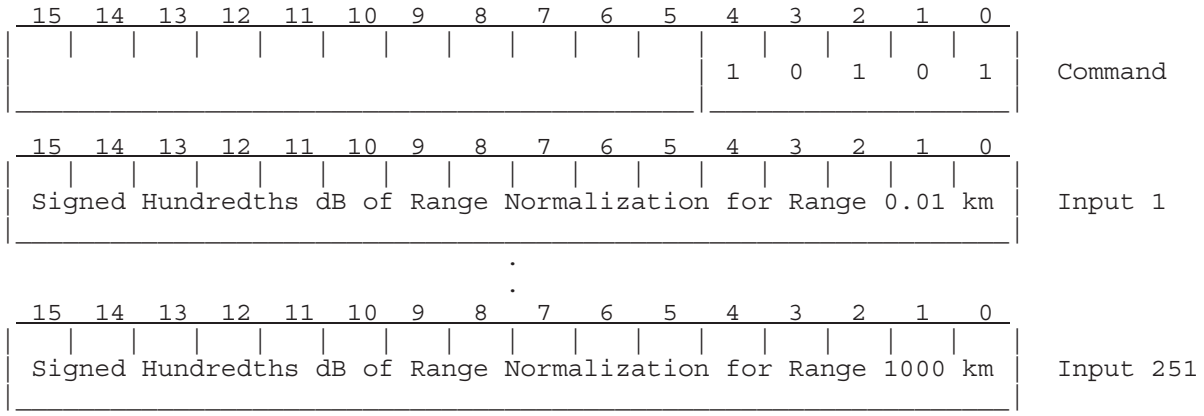


7.18 Load Custom Range Normalization (LDRNV)

Reflectivities computed by the RVP900 are ordinarily corrected for range effects by adding an offset in deciBels equal to 20 log(R/1 km), 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.

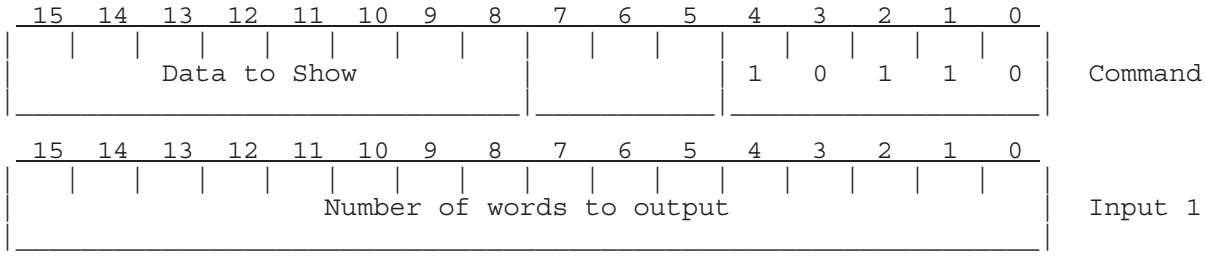
LDRNV loads a 251-word custom correction table holding values in hundredths of deciBels over five decades of log(range) from 0.01km to 1000km. There are 50 table entries per decade of range. Thus, the range in kilometers corresponding to an input word #N is 10[{{N-1}<devide>50}-2], and the default correction table (automatically used on power-up) is simply 40(*N* - 101) . The table values are stored and interpolated whenever the RVP900 loads a new range mask (see Section 7.2 Load Range Mask (LRMSK) on page 257), 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 Section 7.3 Setup Operating Parameters (SOPRM) on page 259) 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 RVP900 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 4 ms 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 pulse width
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 (Section 7.29 Set Individual Thresholds (THRESH) on page 329). 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 *RVP98D* 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 RVP9SpecFiltIO* 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 rvp9.h.
16	Returns the identifiers of the currently active HydroClass classifier algorithms. The identifiers are encoded in *sig_data_types.h*
17	Returns the nickname of the currently active HydroClass configuration. The nickname can be used to label the possible customizations made to echo identification settings in the d*polapp_*-band.conf* file. It is impractical to save all modified parameters as metadata. The string of 16 characters is reported in a sequence of eight words, each reporting two characters.

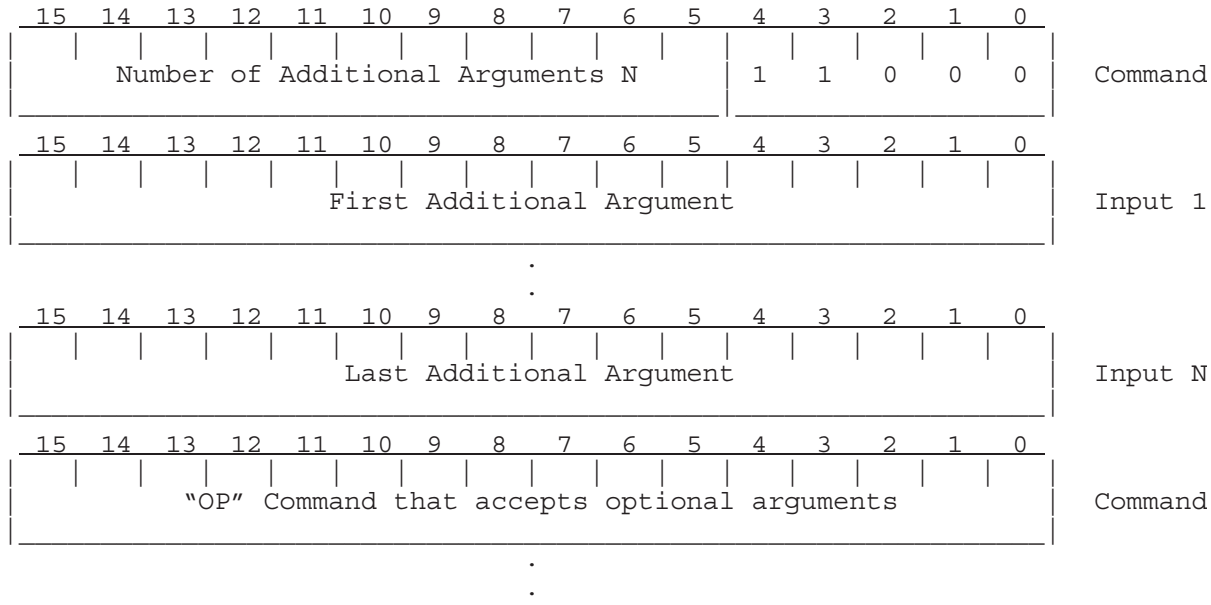
7.20 Pass Auxiliary Arguments to Opcodes (XARGS)

This command provides a backward compatible mechanism for supplying additional (optional) arguments to other opcodes. The command may be used freely in the RVP900 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

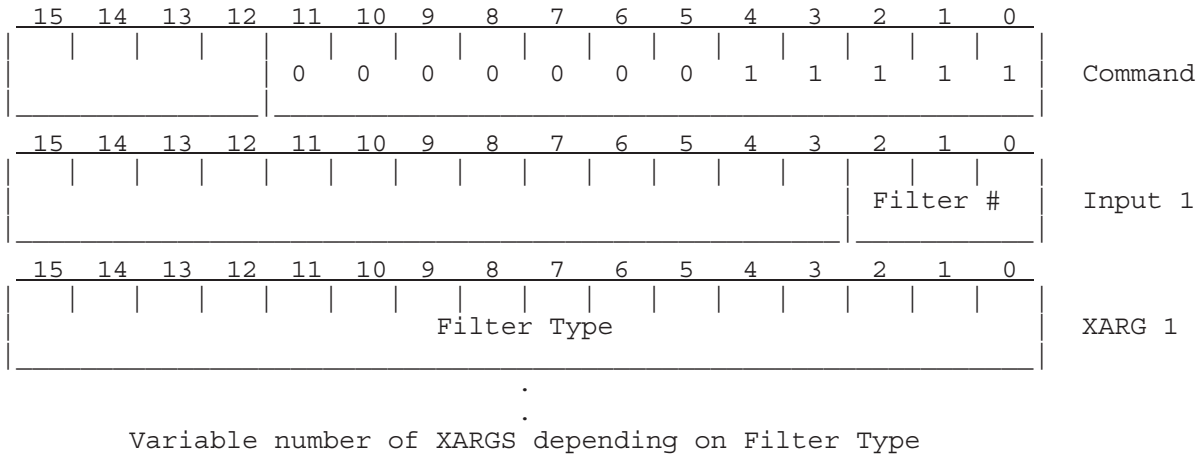
its upper 11-bits. This is followed by the array of between 0 and 2047 additional arguments. At this point the XARGS command is finished and the "OP" command is fetched as the next instruction. "OP" will execute normally, except that the additional arguments from XARGS can be picked up after its own 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 RVP900 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 rvp900SpecFiltIO* (See *include/rvp900.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.

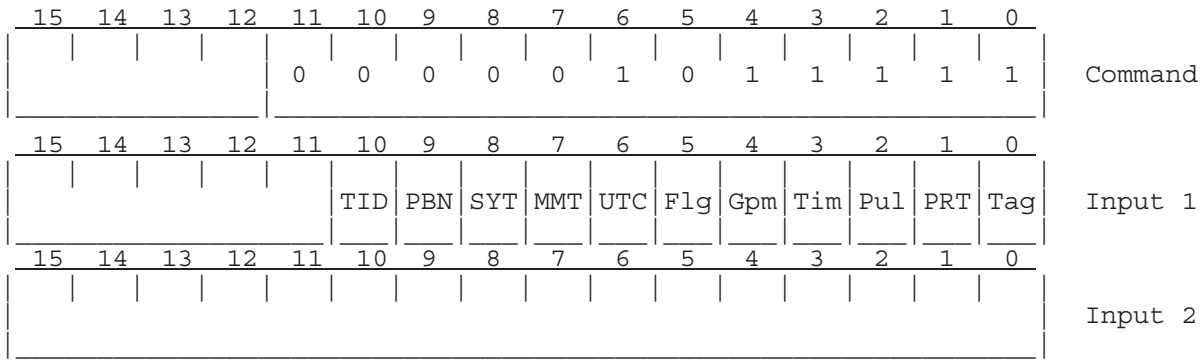
Type:3 SPFILT_GMAP Gaussian Model Adaptive Processing Spectral Filter

This is the RVP900 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 RVP900 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.

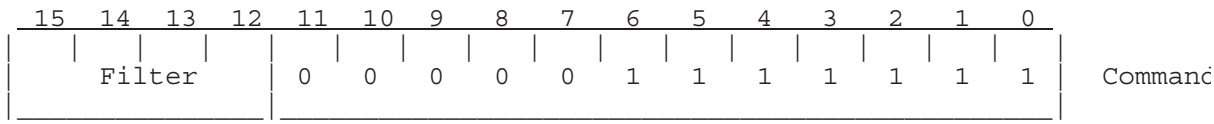
Also, the RVP900 will output all zeroed data whenever a ray contains any blanked pulses.

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 RVP900 configuration). See the MMTS_* flags in dsp.h.
SYT	IFDR 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 rvp900TaskID_IO* (14 words total)
PedINU	Auxiliary pedestal and INU information encoded as *struct rvp900AuxPedINU*, giving full angle and position information for moving platform systems (18 words total).

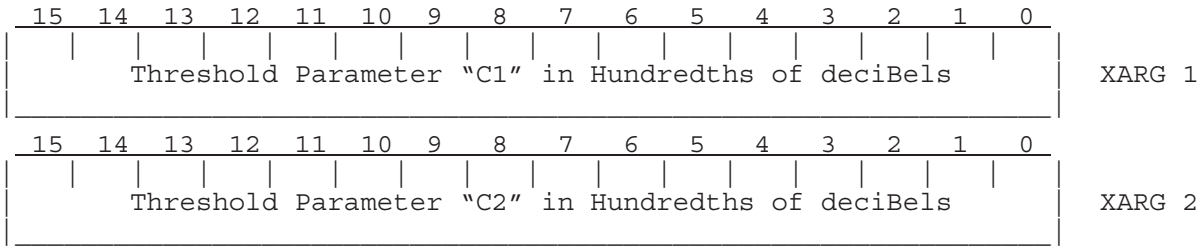
7.23 Configure Interference Filter (CFGINTF)

The RVP900 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 Section 7.20 Pass Auxiliary Arguments to Opcodes (XARGS) on page 320).

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.

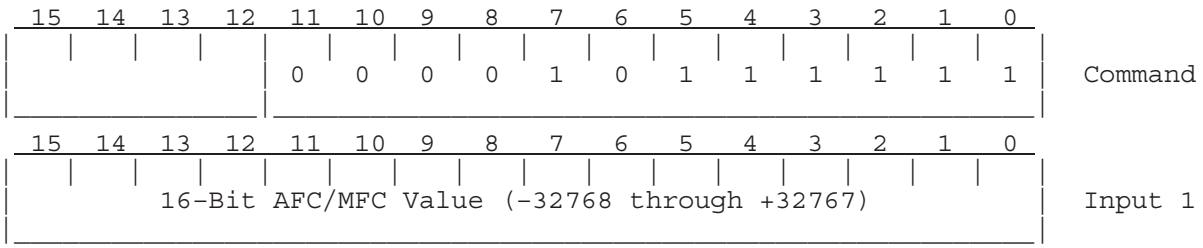


Filter — Chooses which interference algorithm should be run. See Section 6.1.5 Interference Filter on page 189 for a description of the available algorithms.
0: None (Interference filtering is disabled)
1: Alg.1 (Traditional JMA Algorithm)
2: Alg.2 (Alg.1 optimized for additive interference)
3: Alg.3 (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.



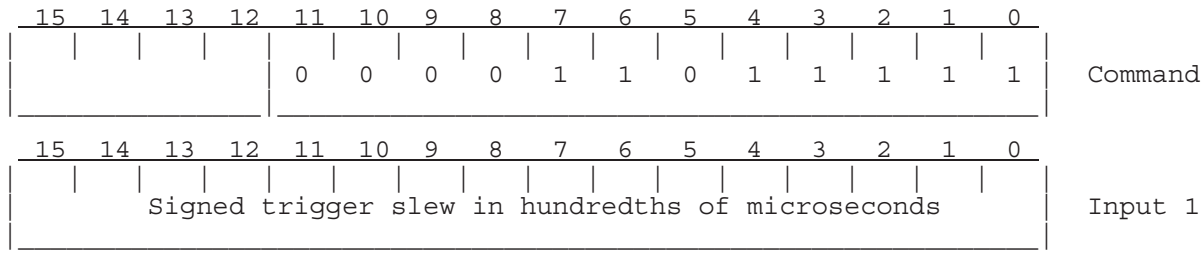
7.24 Set AFC level (SETAFC)

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 RVP900 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.



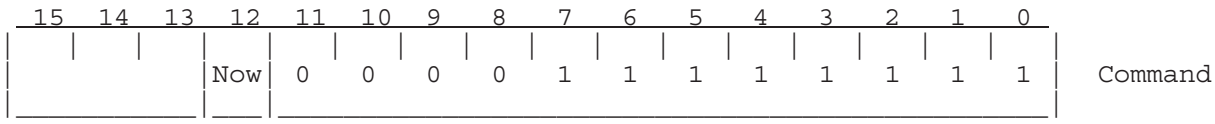
7.25 Set Trigger Timing Slew (SETSLEW)

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 -327.68 µsec to +327.67 µsec. The actual span permitted by the RVP900 is ±20 µsec. 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 RVP900 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.

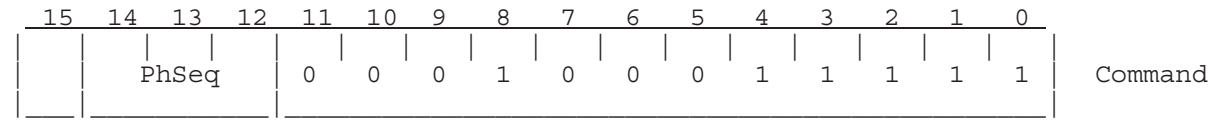


Now — Forces the hunt procedure to be started even if the burst pulse is already present. Normally the procedure will only be started when the burst pulse is missing at the time BPHUNT is given.

7.27 Configure Phase Modulation (CFGPHZ)

This command configures the RVP900 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 Section 4.2.8 Mz — Transmissions and Modulations on page 137.



PhSeq=0 — Selects **No Modulation**. The RVP900 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 RVP900 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 RVP900 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.

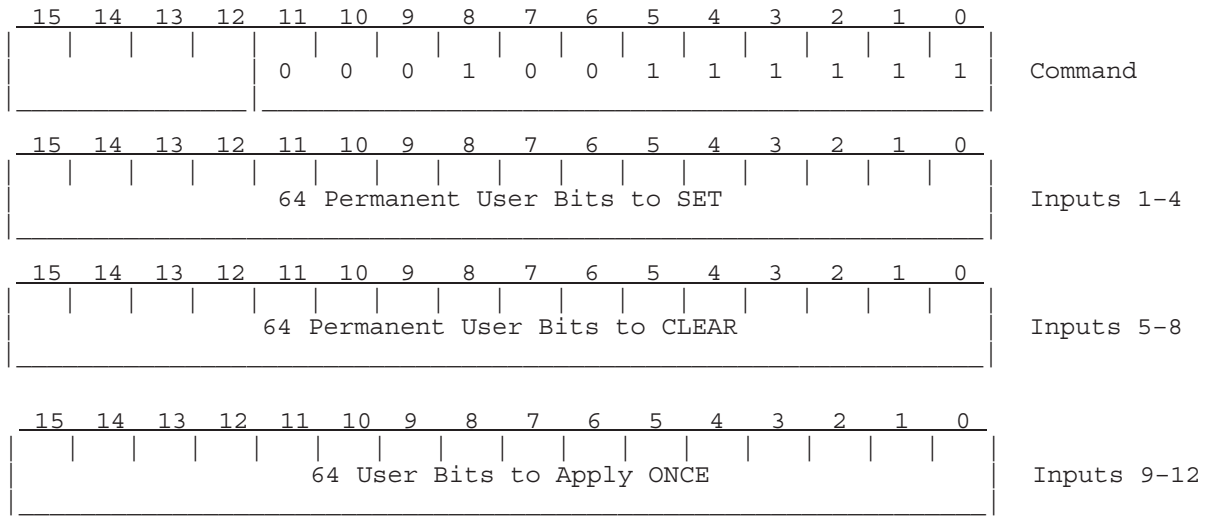
PhSeq=3 Selects the **SZ(8/64)** sequence. This is a systematic code[4], which does a nice job separating and recovering first and second trip echoes in "Random Phase" mode. It will usually perform better than a truly random transmit sequence, especially when the processing interval is fairly short (as little as 32-pulses). With no XARGS, the RVP900 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 RVP900 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.

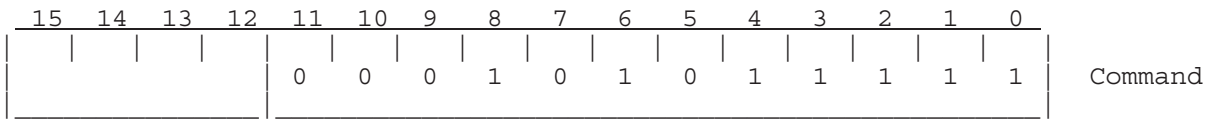
A FIFO history of the permanent bits is maintained so that the bits can be associated with the data being acquired right now as the UIQBITS opcode is executed. Each 16-bit command arg specifies bits to Set/Clr in successive bytes of the structure. This allows user code to safely change some bits without affecting others.

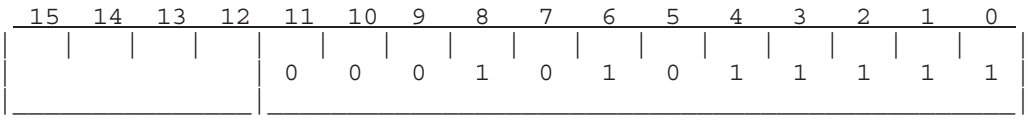
The user bits from separate calls will never be collapsed into a single pulse header, even if the header and bit times indicate that they could. This means that each UIQBITS opcode will always result in at least one pulse header being tagged with exactly those data. This is generally what you want, since no other exact outcome could be guaranteed based on time-of-arrival alone.



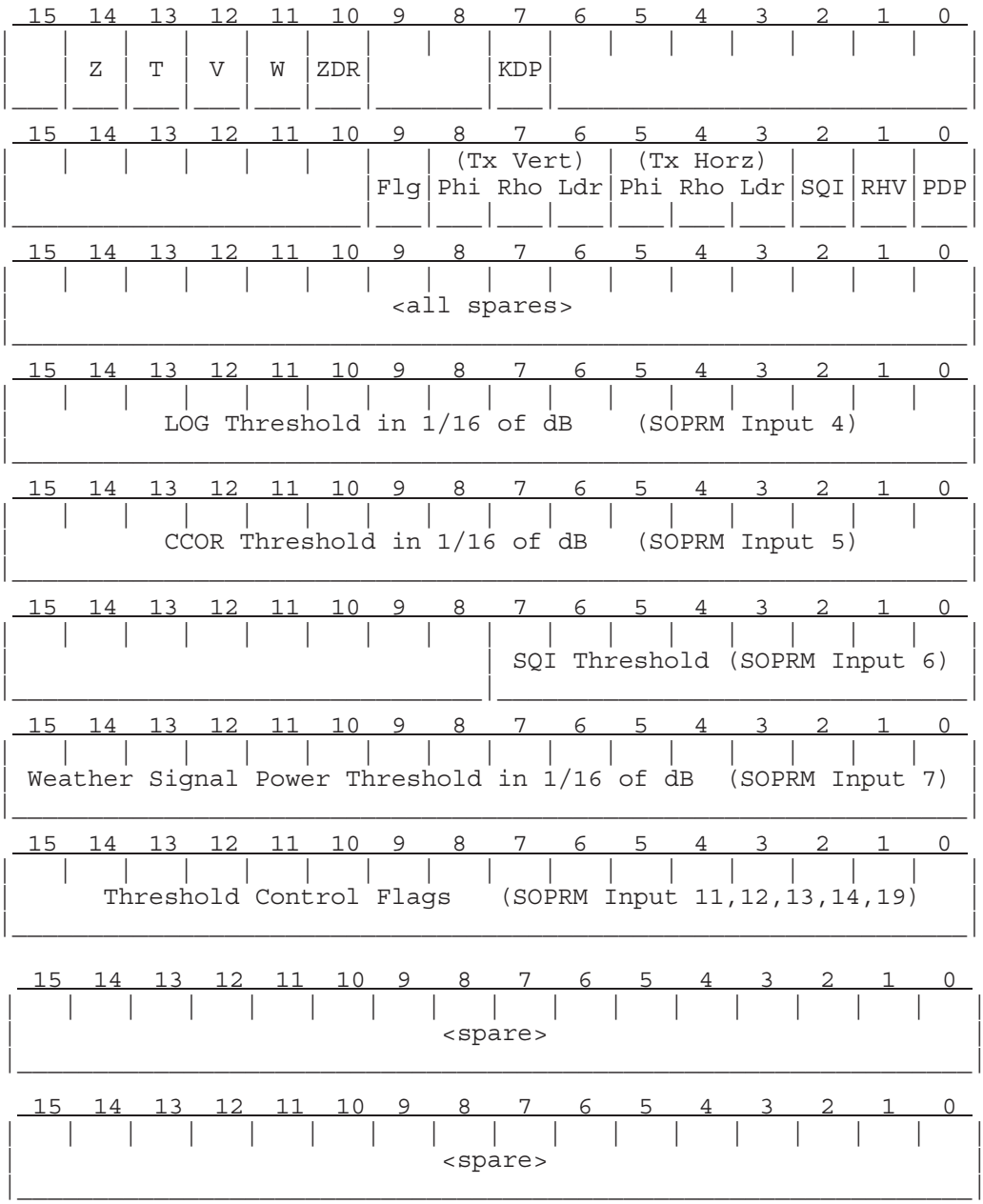
7.29 Set Individual Thresholds (THRESH)

The SOPRM command in Section 7.3 Setup Operating Parameters (SOPRM) on page 259 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 (Section 7.19 Read Back Internal Tables and Parameters (RBACK) on page 319).





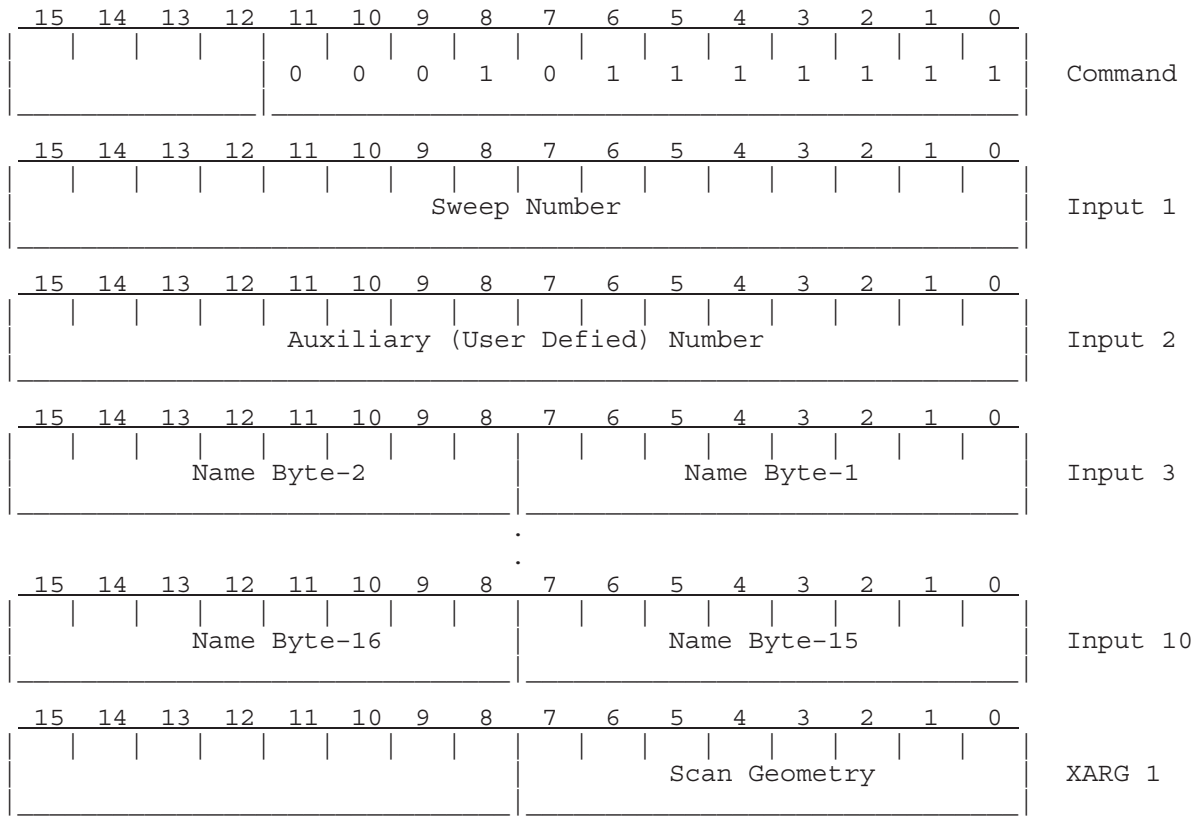
The first three words supply a mask that indicates which data types are being set:



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 RVP900. This naming information then becomes associated with these data, and is available in the pulse information structures (*struct rvp900PulseInfo*) that are read from the Timeseries API. The *iAqMode* field of the pulse headers (*struct rvp900PulseHdr*) will be incremented each time a TASKID opcode is received, but the continuous flow of (I,Q) data from the RVP900/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.

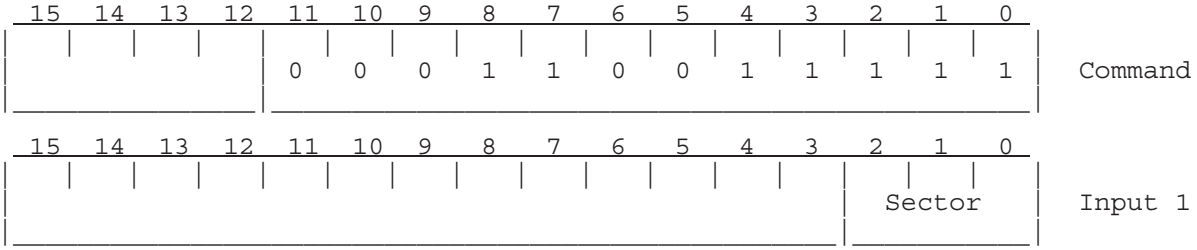


7.31 Define PRF Pie Slices (PRFSECT)

This command supplements the SETPWF command (Section 7.14 Set Pulse Width and PRF (SETPWF) on page 311) 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.

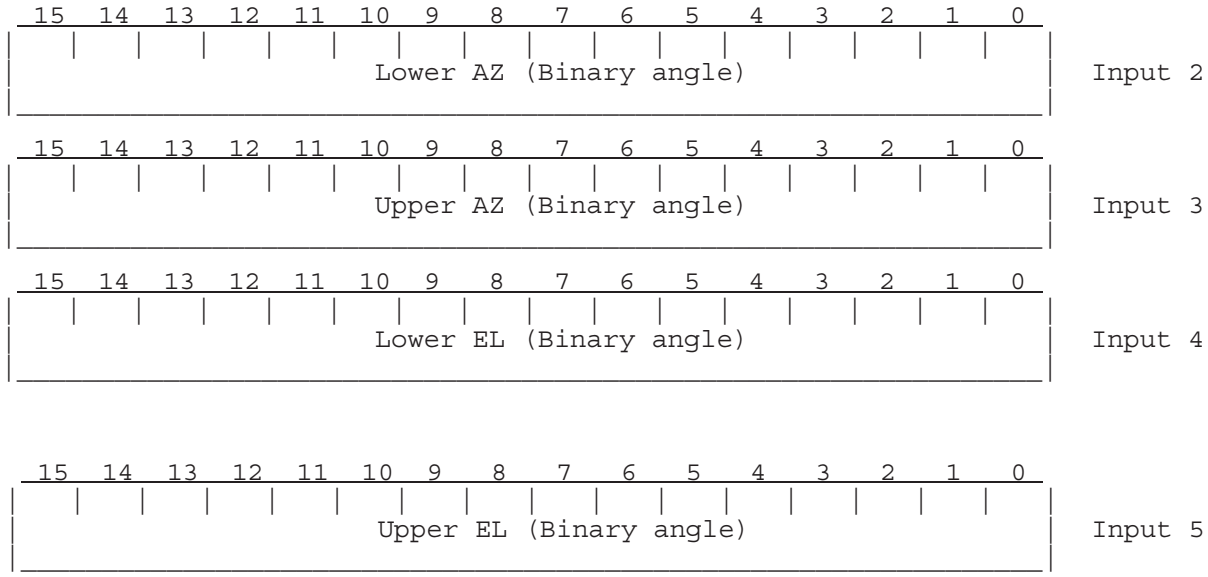
The SETPWF opcode completely erases any alternate sectors that have been setup so far. Thus, the PRFSECT command can only be used after SETPWF has established the pulsewidth and default trigger rate for the entire AZ/EL scan volume.



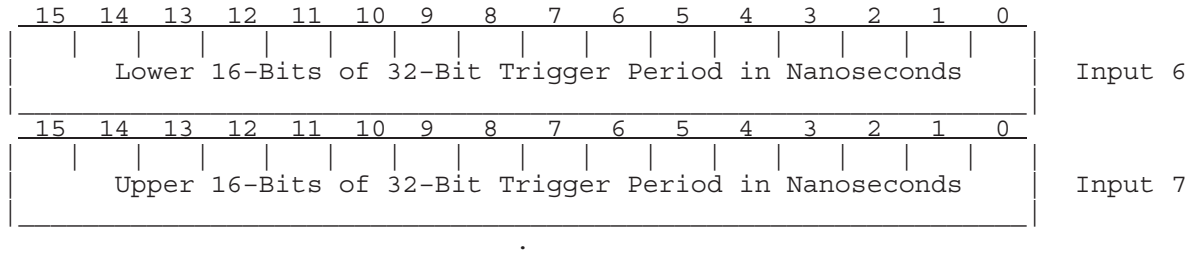
Sector — Selects which sector number is being defined to have an alternate trigger pattern. This is simply an arbitrary index from 0-7.

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.

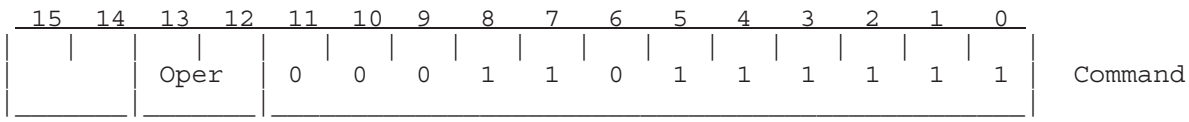


.

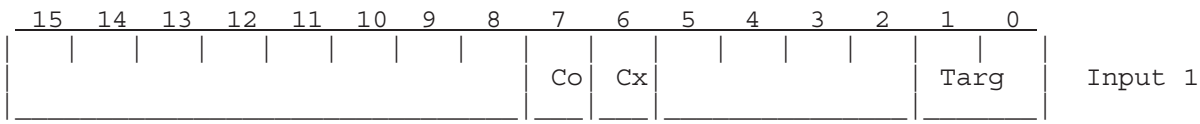
7.32 Configure Target Simulator (TARGSIM)

The RVP900 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.

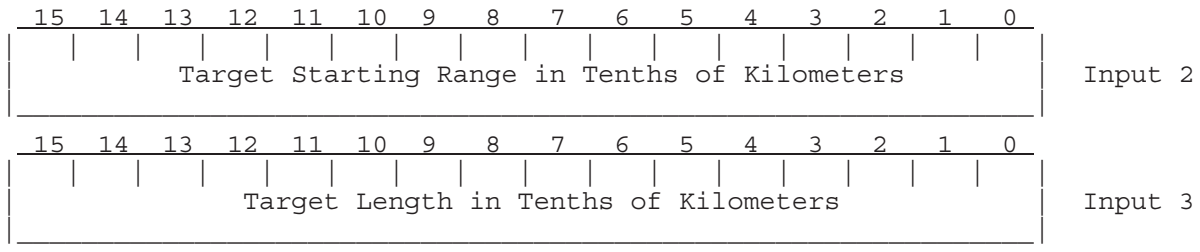
The target simulator can be used with both live and simulated (I,Q) data (See LSIMUL opcode in Section 7.10 Load Simulated Time Series Data (LSIMUL) on page 304). In the former case, it allows you to overlay simulated physical targets on top of real physical targets from the radar receiver.

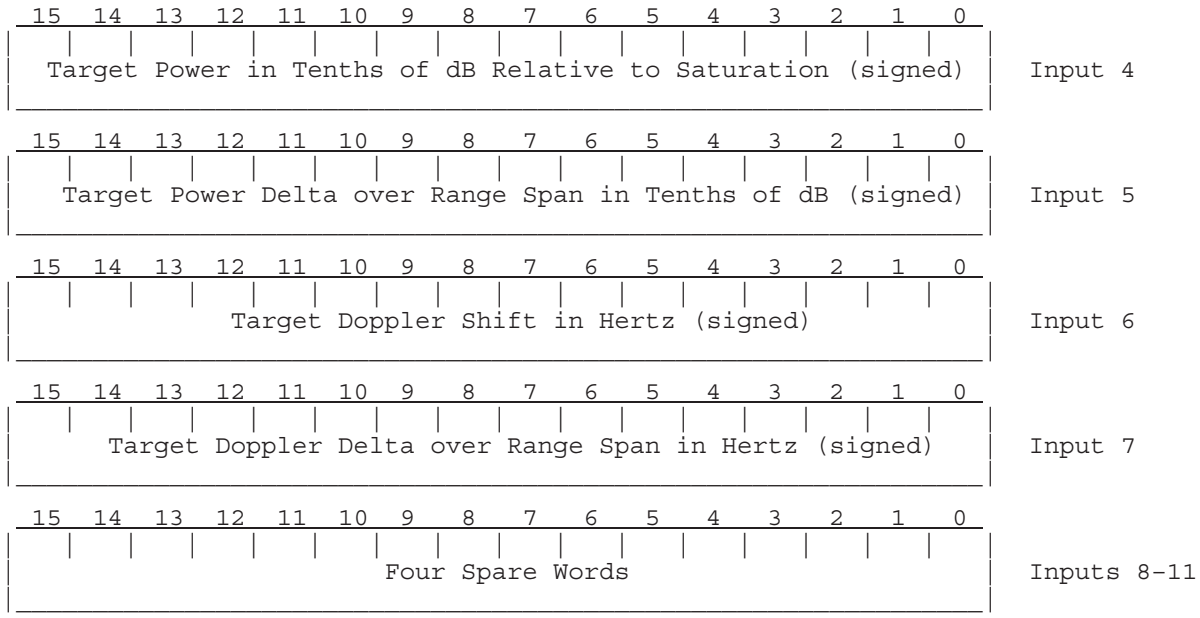


Oper=0	Disable all target simulation activity (no additional args)
Oper=1	Enable simulation of all defined targets (no additional args)
Oper=2	Define a new simulated target (arg list follows)



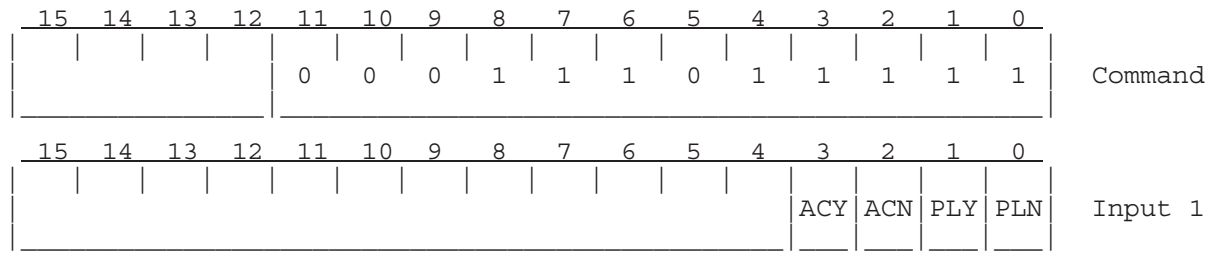
Targ	Targ Which target is being defined
Co/Cx	Place simulated target in Co-Pol and/or Cross-Pol Rx channels





7.33 Set Burst Pulse Processing Options (BPOPTS)

Some burst pulse processing options can be set by this command.

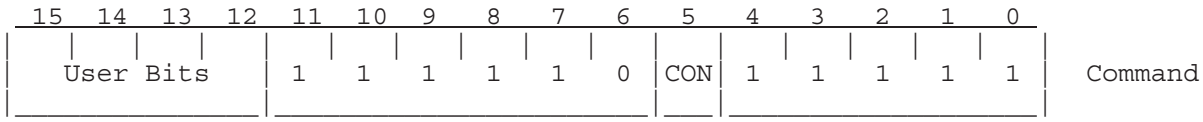


PLY/N These bits affect whether the RVP900 will phase lock its (I,Q) data to the measured burst pulse. The "PLY" and "PLN" bits force "Yes" and "No" responses. If both bits are clear or both bits are set, then no change will be made.

ACY/N These bits affect whether the RVP900 applies pulse-to-pulse amplitude correction to its (I,Q) data. The Yes/No bits behave the same as PLY/PLN.

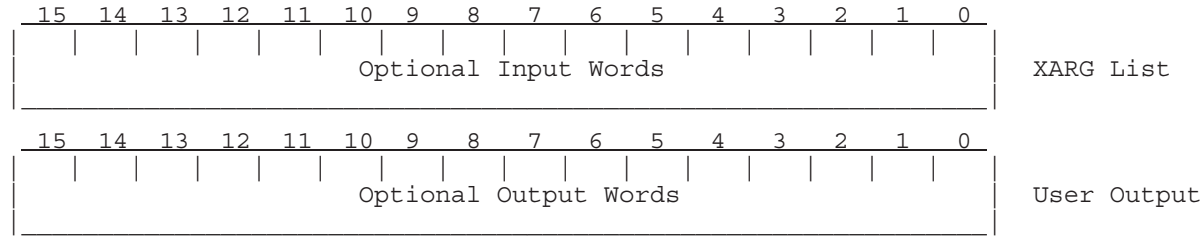
7.34 Custom User Opcode (USRINTR and USRCONT)

These opcodes are part of the open software extensions to the RVP900, 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 RVP900 IQ data acquisition thread proceeds continuously while the opcode is executed. If clear, then the IQ stream is interrupted prior to handling the call.



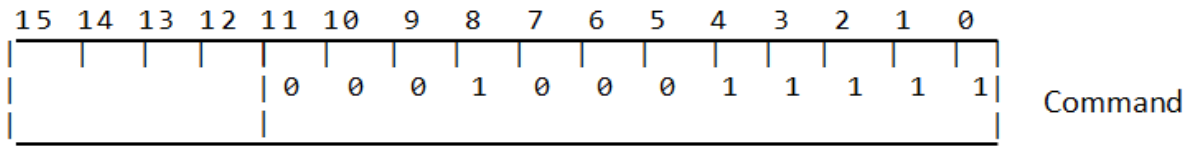
7.35 Load Melting Layer Specification (MLSPEC)

The RVP900 allows to get melting layer specific information from the latest melting layer product created in IRIS. The information includes the melting layer height and the uncertainties. The specifications of the melting layer are set as a function of range, azimuth, and elevation based on the information coming from the configuration of the task and either the melting layer product parameters used in the latest melting product created or the melting layer product default parameters.

The RVP900 can use spatially variable melting layer (ML) altitudes, which may be preloaded for each interval of data processing (PROC). The ML altitudes are referenced to the Mean Sea Level, and estimate the top of ML.

The input data are the maps of melting layer altitudes projected into sweep cones of the data sweeps, to be carried out by OpCommand PROC.

The command is an extended OP command and is defined as follows:



The lowest 5 bits correspond to the default 0X1F for extended OP commands and the 7 middle bits are the actual value for the XOP_MLSPEC command (0X10).

The application SW provides the input data, for example by converting the IRIS standard Cartesian MLHGT products into the curved Earth co-ordinates of the sweeps configured for the radar task. Typically, a map for a collection of sweeps (a volume) is uploaded.

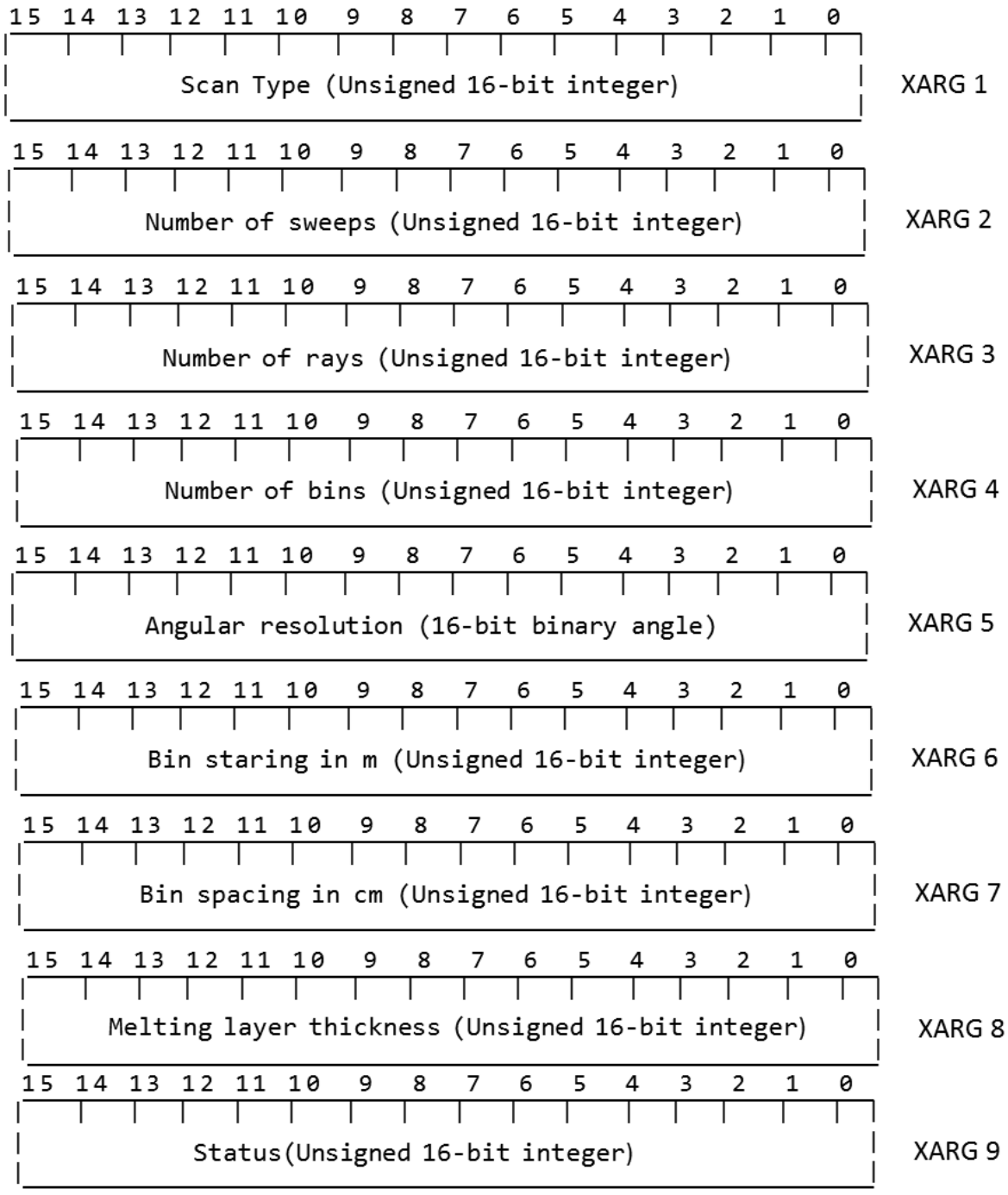
Application SW adjusts the spatial resolution of the map by down scaling of gates so that it fits in the buffer managed in RDA:

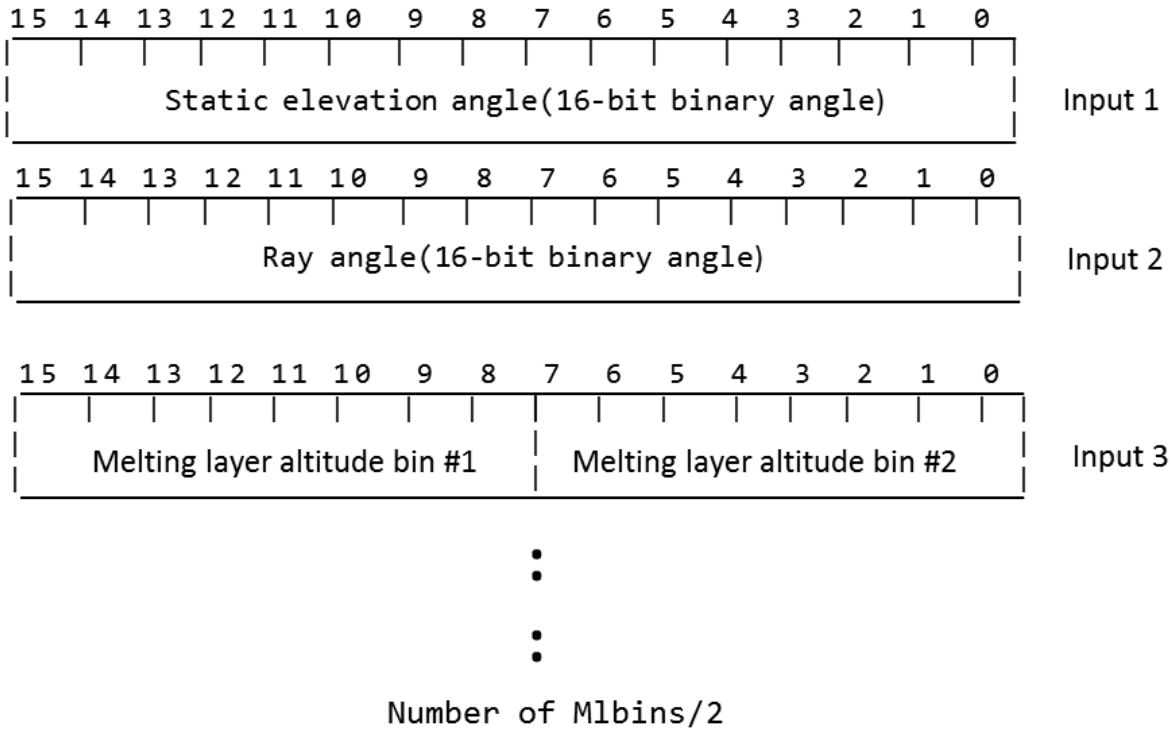
DPOLAPP_MAX_ML_SWEEPS*DPOLAPP_MAX_ML_RAYS*DPOLAPP_MAX_ML_BINS

The command also checks the maximum number of sweeps and rays (DPOLAPP_MAX_ML_SWEEPS = 30 and DPOLAPP_MAX_ML_RAYS=90). If the buffer does not exist or exceeds its limits, it is flushed.

The RVP900 supports command that provides a mean to feedback the melting layer information from IRIS into the RDA. So that, the melting layer altitude information is specified for every antenna angle as well as range. This way melting layer height can be used in different live data processing applications such as Hydroclass.

The RVP900 maintains an internal array of up to 1024 different filter versus- range tables, each of which is keyed to a particular angle (EL, for PPIs and AZ, for RHIs). Each XOP_MLSPEC command uploads the complete sweep. Then, for each live bin in every ray being processed, the RVP900 obtains the melting layer at the midpoint AZ/EL of the ray.


15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
Scan Type (Unsigned 16-bit integer)
XARG 1
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
Number of sweeps (Unsigned 16-bit integer)
XARG 2
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
Number of rays (Unsigned 16-bit integer)
XARG 3
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
Number of bins (Unsigned 16-bit integer)
XARG 4
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
Angular resolution (16-bit binary angle)
XARG 5
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
Bin staring in m (Unsigned 16-bit integer)
XARG 6
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
Bin spacing in cm (Unsigned 16-bit integer)
XARG 7
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
Melting layer thickness (Unsigned 16-bit integer)
XARG 8
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
Status(Unsigned 16-bit integer)
XARG 9



The XARGS statements allow flexibility regarding backward compatibility. They include the current task and the product configuration settings. The command uploads the melting layer height and thickness for every volume.

Scan type

The melting layer information can be uploaded to the RDA for both RHI and PPI scans

Number of sweeps

Is the minimum between the number of sweep of the running task and the DPOLAPP_MAX_ML_SWEEPS.

Number of rays

The number of rays is calculated as follows:

$$\#\ rays = \frac{360000}{iRES}$$

where iRES is the desired angular resolution expressed as an integer number of thousands of degrees.

Number of bins

Corresponds to the number of output bins

Angular resolution

Is the binary angle of the corresponding angle in degrees $\theta = \frac{360}{nMLrays}$

Bin starting

Range of the first bin in centimeters

Bin Spacing

It is the product of the number of output bins by the step divided by the number of bins.

Melting layer thickness

Corresponds to the USER setting of the melting layer thickness in the product configuration

Status

Indicates the status of the DSP features.

APPENDIX A
SERIAL STATUS FORMATS

The RVP900 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. For more details, see Section 7.9 Get Processor Parameters (GPARM) on page 290. 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 15 Internal BITE Packet (RVP900 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 = RVP900 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 15 Internal BITE Packet (RVP900 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 21–27 D6 = 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 15–21 D6 = 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 15 Internal BITE Packet (RVP900 to Host) (Continued)

Char	Function
11	Immediate Status 22–28 D6 = GP18,D11 = IFDR 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 = IFDR PLL is not locked to external reference D1 = GP18,D6 = <spare> D0 = GP18,D5 = IFDR 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 RVP900D4 = 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 15 Internal BITE Packet (RVP900 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 15 Internal BITE Packet (RVP900 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 RVP900 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. For more details, see Section 7.9 Get Processor Parameters (GPARM) on page 290.

Table 16 Internal QBITE Packet (RVP900 to Host)

Char	Function
1	SYNC Byte (AF Hex)
2	Identification byte (User Choice)
3–4	Burst pulse frequency, IFDR #1
5–6	Burst pulse frequency, IFDR #2
7–8	Burst pulse power, IFDR #1
9–10	Burst pulse power, IFDR #2
11–12	Noise level, IFDR #1
13–14	Noise level, IFDR #2
15–16	Chassis temperature, IFDR #1
17–18	Chassis temperature, IFDR #2
19–20	FPGA temperature, IFDR #1
21–22	FPGA temperature, IFDR #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

RVP900 PACKAGING

This section describes the general features of the packaging and the electrical specifications and cabling of these units.

B.1 RVP900 Processor Components

A complete RVP900 processor is built from the following components:

- RVP902 Main Computer (see TBD)
- RVP901 IFDR Module (IF Digital Receiver) (see Section B.4 IFDR Module on page 344)
- Optional DAFC (Digital AFC) (see Section B.5 Optional DAFC on page 347)
- Optional TDWR Custom Back Panel (see TBD)

B.2 Safety

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.

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 IFDR module unless a properly grounded wrist strap is worn.

B.3 Main Computer

Need description here

B.4 IFDR Module

The IFDR module mounting brackets are the same dimensions as previous RVP8 generation IFDs. The compact sealed module has the following dimensions: 26.8 cm x 17.6 cm x 4.8 cm (10.5 in x 6.9 in x 1.9 in). The module can be mounted on edge with the 26.8 cm x 4.8 cm (10.5 in x 1.9 in) surface flush to the back of the receiver cabinet with 17.6 cm (6.9 in) protrusion into the cabinet. The module is typically placed where a traditional LOG receiver, or a previous generation RVP IFD, would be installed.

The IFDR is cooled by direct conduction through its metal enclosure. One side of the IFDR serves as a heat sink. The hotter chips mounted on the printed circuit board are bonded to the heat sink. The IFDR should be positioned so that a minimum of 20 cubic feet per minute of air can freely convect around it. The ambient air temperature should be within a range of -40C to +50C (-40F to 122F) with DC supply power option and -40C to +45C (-40F to 113F) with AC supply option.

The power module is separate unit. The IFDR is delivered with the power supply module attached to the IFDR mechanical housing. The power supply module could be removed and mounted nearby in the radar cabinet. The power supply, fans, and bracket add 8.4 cm (3.3 in) of overall width to the receiver module. Power modules are available for 100 to 240 VAC at 47 to 63 Hz or 18 to 36 VDC. The AC power supply is a low noise, low ripple, auto-switching unit. The DC input voltage may be unregulated. If DC voltage is selected as an option, different fan assemblies are available. These fan options allow DC voltages from 12 to 18 VDC, 18 to 28 VDC, and 28 to 36 VDC.

The IFDR has an internal regulator to supply the various digital circuits the voltages needed. The internal regulated power modules are sized to provide several Watts more than is required by the RVP901 itself. A ferrite choke, around the supply wires, near the terminal strip is also recommended.

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 IFDR on the opposite side of the power supply. The filters and mounting bracket add 2.0 cm (0.8 in) of overall width.

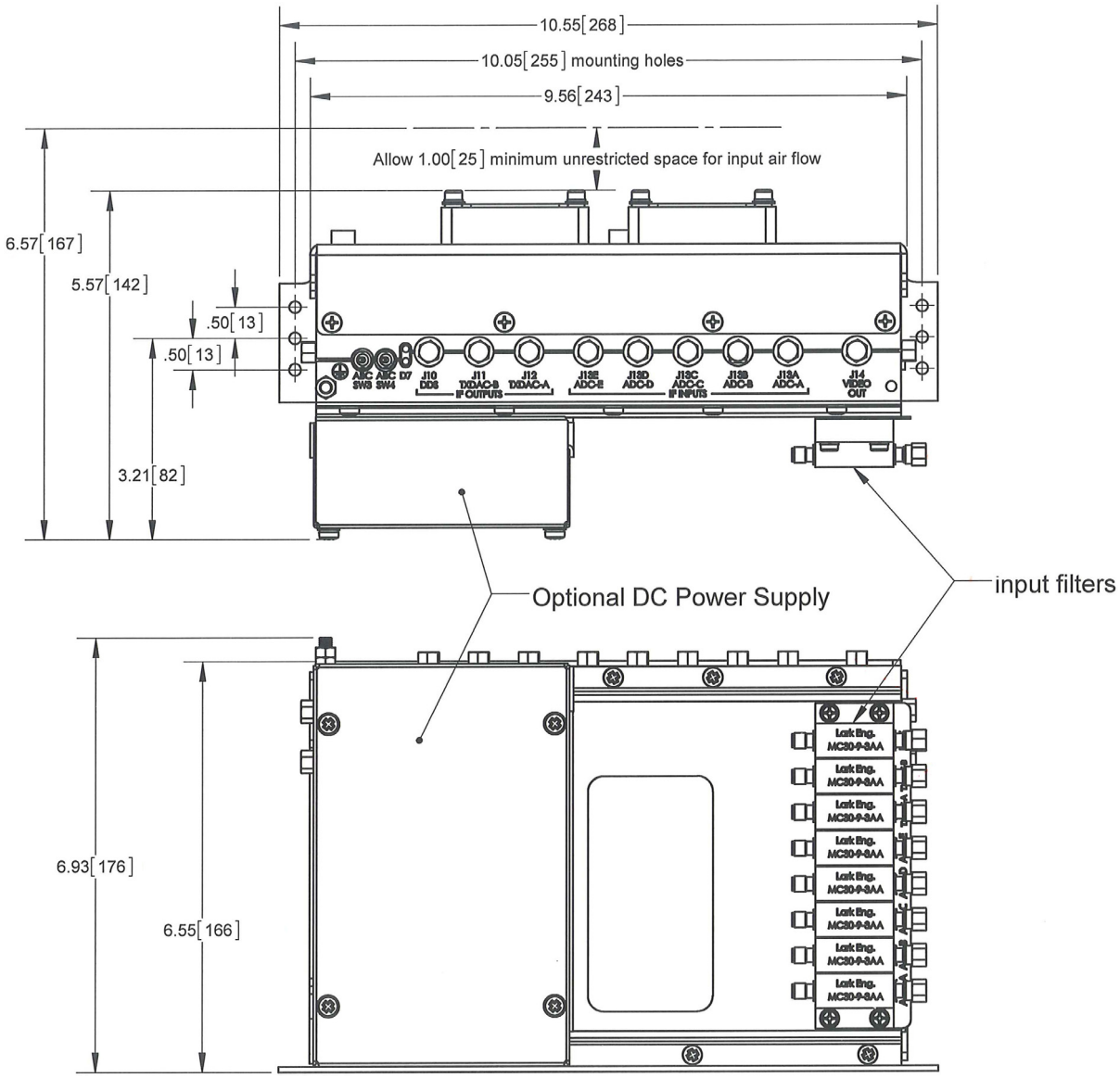


Figure 54 IFDR - Top and Front Face

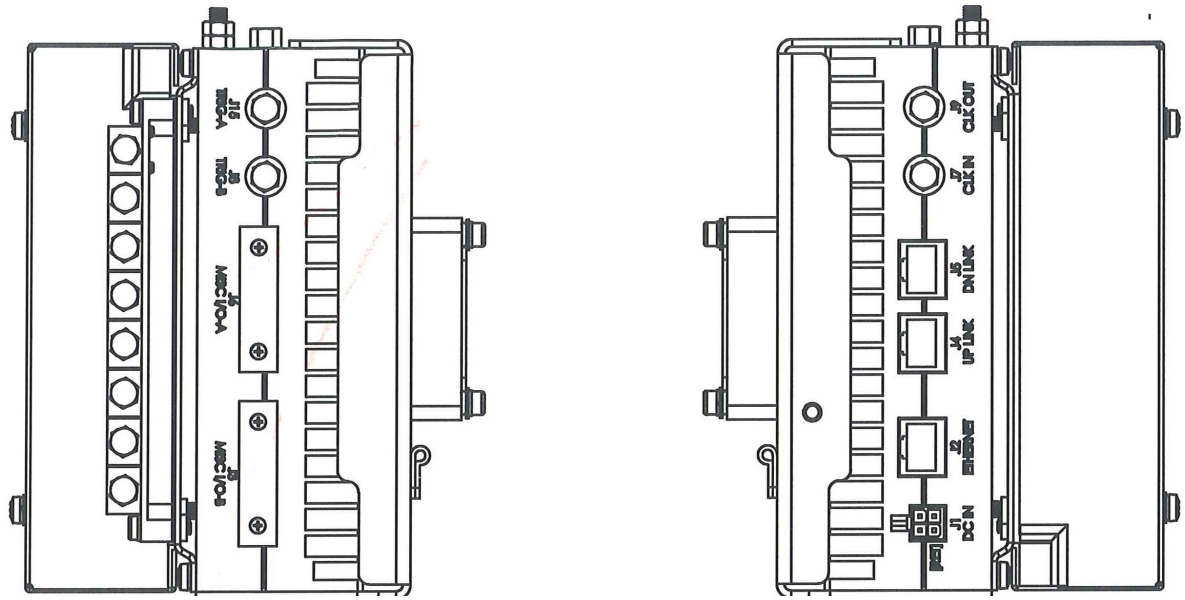


Figure 55 IFDR - Right and Left Sides

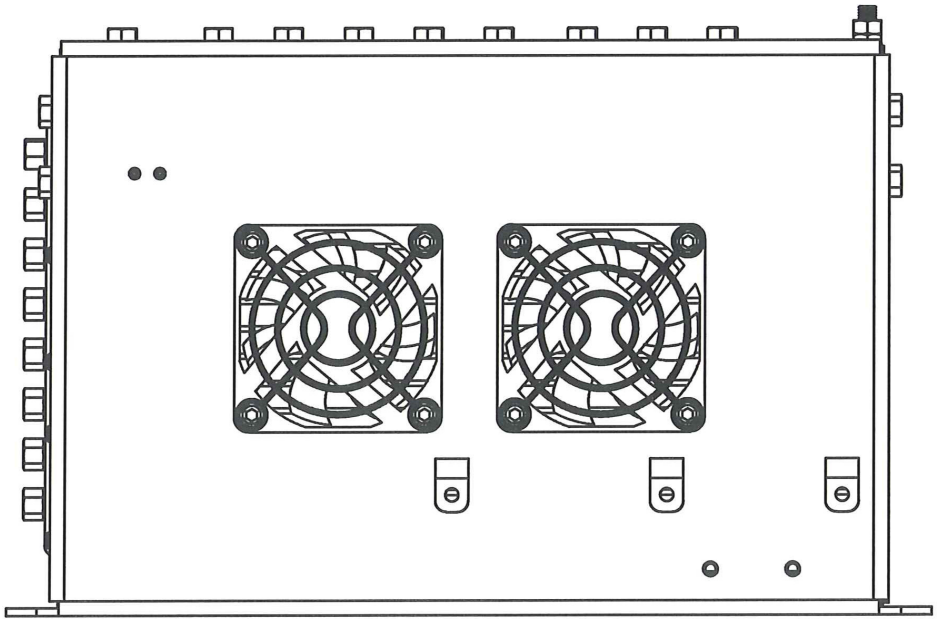


Figure 56 IFDR - Fan Side (Heat Sink)

B.4.1 Generic I/O Interconnect Breakout Cable

A generic interconnect cable is available to breakout each of the 51-pin micro "D" connectors on the IFDR into a standard 25-pin and 37-pin

female “D” connector. The cable wiring and internal signal names are shown in Table 17. The cable provides 10 RS-422 signals, 10 TTL signals, three differential analog input A/D signals, and +5V and -5V auxiliary power from the IFDR. Standard cable length is 1 meter.

Table 17 Generic Interconnect cable for IFRD Analog/Digital I/O

D Connectors					Index	
51-Pin	25- Pin	37- Pin	Signal Name	Softplane Signal	J3	J6
1/19	1/14	—	GPDIFF_PIN_LP/N	Comm.diff	0	10
2/20	2/15	—	GPDIFF_PIN_LP/N	Comm.diff	1	11
3/21	3/16	—	GPDIFF_PIN_LP/N	Comm.diff	2	12
4/22	4/17	—	GPDIFF_PIN_LP/N	Comm.diff	3	13
5/23	5/18	—	GPDIFF_PIN_LP/N	Comm.diff	4	14
6/24	6/19	—	GPDIFF_PIN_LP/N	Comm.diff	5	15
7/25	7/20	—	GPDIFF_PIN_LP/N	Comm.diff	6	16
8/26	8/21	—	GPDIFF_PIN_LP/N	Comm.diff	7	17
9/27	9/22	—	GPDIFF_PIN_LP/N	Comm.diff	8	18
10/28	10/23	—	GPDIFF_PIN_LP/N	Comm.diff	9	19
18	13	—		GND		
11/29	—	1/20	TTLIO_PIN/GND	Comm.ttl/GND	0	10
12/30	—	2/21	TTLIO_PIN/GND	Comm.ttl/GND	1	11
13/31	—	3/22	TTLIO_PIN/GND	Comm.ttl/GND	2	12
14/32	—	4/23	TTLIO_PIN/GND	Comm.ttl/GND	3	13
15/33	—	5/24	TTLIO_PIN/GND	Comm.ttl/GND	4	14
16/34	—	6/25	TTLIO_PIN/GND	Comm.ttl/GND	5	15
17/35	—	7/26	TTLIO_PIN/GND	Comm.ttl/GND	6	16
36/37	—	8/27	TTLIO_PIN/GND	Comm.ttl/GND	7	17
38/39	—	9/28	TTLIO_PIN/GND	Comm.ttl/GND	8	18
40/41	—	10/29	TTLIO_PIN/GND	Comm.ttl/GND	9	19
42/43	—	12/30	AMUX_POS_PIN	AMUX P0/N0	0	3
			AMUX_NEG_PIN			
46/47	—	14/32	AMUX_POS_PIN	AMUX P1/N1	1	4
			AMUX_NEG_PIN			
50/51	—	16/34	AMUX_POS_PIN	AMUX P2/N2	2	5
			AMUX_NEG_PIN			
44/45	—	18/36	V_5P0_GPIO/GND	+5V/GND		
48/49	—	19/37	V_N5P0_GPIO/GND	-5V/GND		

B.5 Optional DAFC

The Digital Automatic Freq Control (DAFC) module is used on the RVP900 for magnetron systems to interface to a digitally controlled STALO. The DAFC is driven from the Trig-A trigger output SMA of the IFDR module. DC power needs be provided by running discrete wires, but

+5 VDC is all that is required to run the DAFC. If you want to supply the STALO power through the ribbon cable between the DAFC and STALO, connect the +24 VDC pin to a suitable power supply. Otherwise, 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 30 cm of the STALO, if possible. For details on the DAFC, including pin assignment examples for some commercial STALO, see the RVP900's Installation chapter (xref TBD).

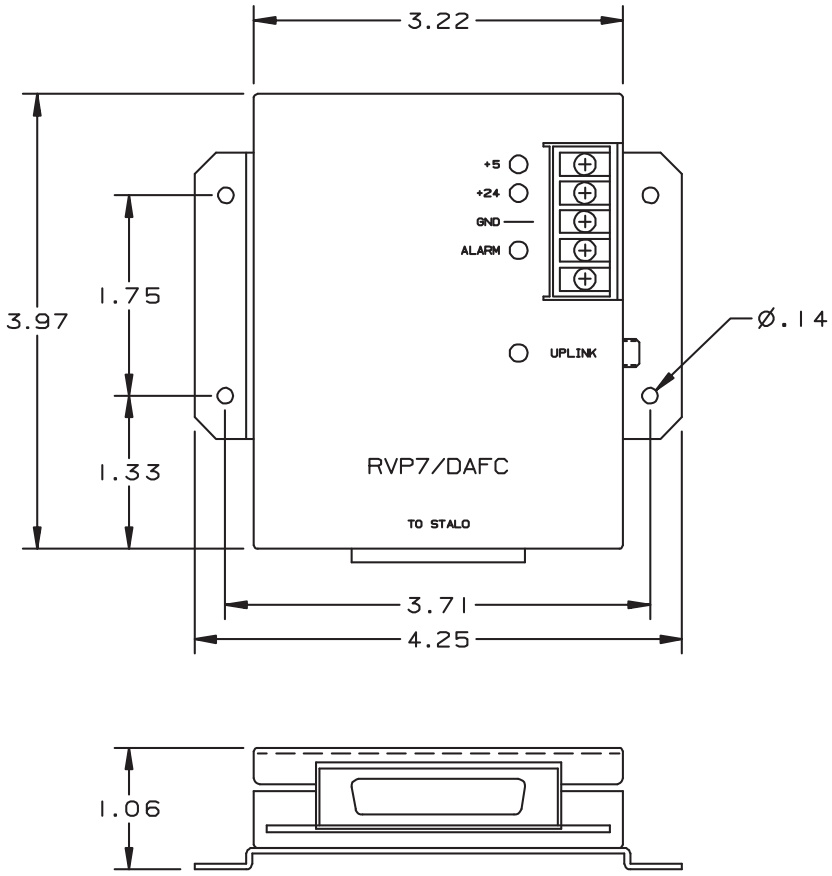


Figure 57 View of DAFC Module

B.6 Optional TDWR Custom Back Panel

The RVP900 can be supplied with a custom back panel that connects to the specific electrical signals of the FAA Terminal Doppler Weather Radar (TDWR). The back panel connects to the two IFDR 51-pin micro-D I/O connectors using a pair of breakout cables shown in Table 17 on page 347.

The signals assigned to the back panel's 25-pin I/O connectors are shown in Tables TBD to TBD. Each line in the tables generally describes a pair of signals that should be twisted together for best signal integrity. A common ground is provided on Pin-25 of all eight connectors.

Table 18 J1 "Filter Amp #1"

Pin	Type	Dir	Signal Name	Comment
1/14	TTL	Out	RFSW/GND	Command, RF Upconvert Switch
2/15	TTL	Out	RFTST/GND	Command, RF Test Switch
3/16	TTL	Out	RFTSTPP/GND	Command, RF Pilot Pulse Switch
4/17	TTL	Out	RFTSTTER/GND	Command, RF Test Term Switch
5/18	TTL	In	RFXTALFM/GND	Status, Crystal Fault Monitor
25	—	—	GND	Common Ground

Table 19 J2 "Filter Amp #2"

Pin	Type	Dir	Signal Name	Comment
2/15	TTL	In	MULTRF/GND	Status, Multiplier Fault
3/16	TTL	In	RFLOFM/GND	Status, RF LO Generator Fault
4/17	TTL	In	RFTSTFM/GND	Status, RF Test Fault
5/18	TTL	In	RFFM/GND	Status, RF Generator Fault
25	—	—	GND	Common Ground

Table 20 J3 "Pedestal"

Pin	Type	Dir	Signal Name	Comment
3/16	RS-422	In	ACP/ACPn	Antenna Control Pulse
4/17	RS-422	In	ARP/ARPn	Antenna Reset Pulse
25	—	—	GND	Common Ground

Table 21 J4 "Transmitter"

Pin	Type	Dir	Signal Name	Comment
4/17	RS-422	Out	RFAMPGT/RFAMPGTn	Transmitter Trigger
5/18	RS-422	Out	BPIGT/BPIGTn	Transmitter Trigger
6/19	RS-422	Out	PFNGT/PFNGTn	Transmitter Trigger
25	—	—	GND	Common Ground

Table 22 J5 "STC #1"

Pin	Type	Dir	Signal Name	Comment
5/18	RS-422	Out	STCTRG/STCTRGn	STC Real–time Trigger
6/19	RS-422	Out	STCCLK/STCCLK/n	STC Real–time Clock
7/20	RS-422	Out	STCBDCLK/STCBDCLKn	STC Serial Loadup Clock
25	—	—	GND	Common Ground

Table 23 J6 "STC #2"

Pin	Type	Dir	Signal Name	Comment
6/19	RS-422	Out	STCDATA/STCDATAn	STC Serial Loadup Data
7/20	RS-422	In	TRLFLT/TRLFLTn	TRL Fault
8/21	RS-422	In	STCFLT/STCFLTn	STC Fault
25	—	—	GND	Common Ground

Table 24 J7 "Stability Monitor"

Pin	Type	Dir	Signal Name	Comment
7/20	RS-422	In	STABFLT1/STABFLT1n	Stability Monitor Fault #1
8/21	RS-422	In	STABFLT2/STABFLT2n	Stability Monitor Fault #2
9/22	RS-422	In	STABBSY/STABBSYn	Stability Monitor Busy
10/23	RS-422	Out	XMTSTABEN/XMTSTABENn	Enable Stability Monitor
25	—	—	GND	Common Ground

Table 25 J8 "REX Power"

Pin	Type	Dir	Signal Name	Comment
8/10	Analog	In	REX5VA/REX5VAn	REX +5V Power Monitor #1
21/23	Analog	In	REX5VB/REX5VBn	REX +5V Power Monitor #2
9/11	Analog	In	REX18VA/REX18VAn	REX +18V Power Monitor #1
22/24	Analog	In	REX18VB/REX18VBn	REX +18V Power Monitor #2
25	—	—	GND	Common Ground

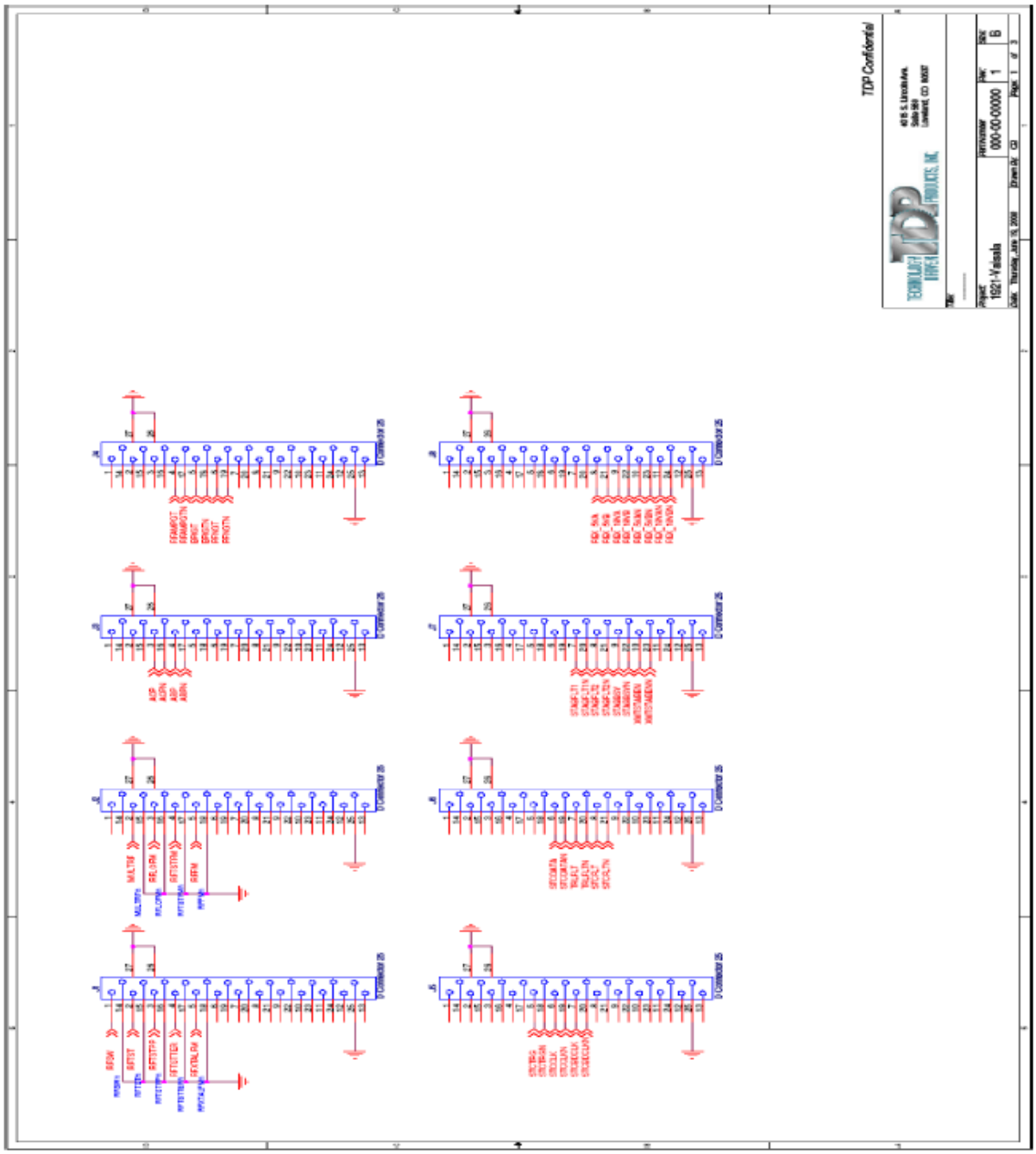


Figure 58 J1 to J9 Wiring Diagrams

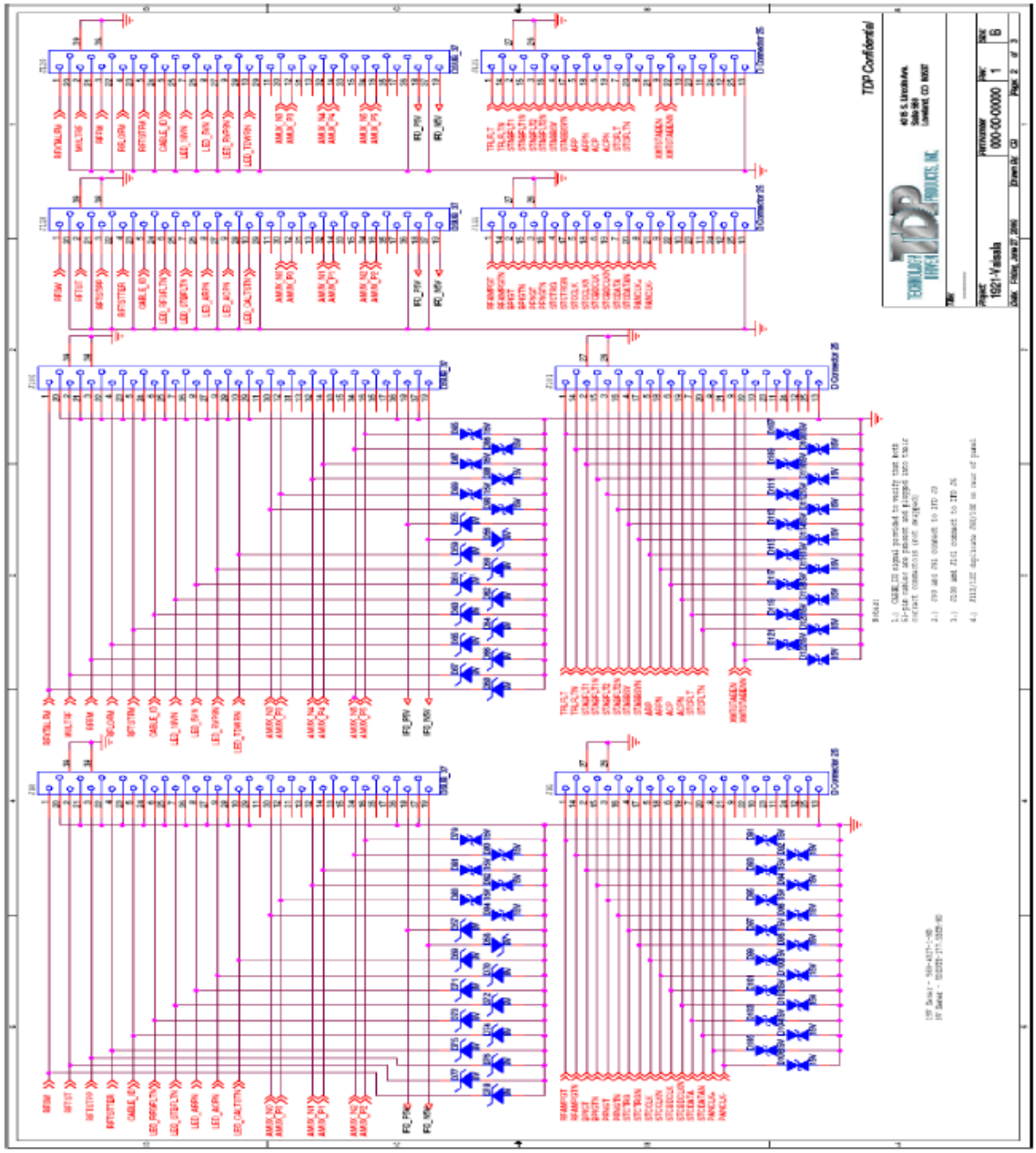


Figure 59 J90 to J111 Wiring Diagrams

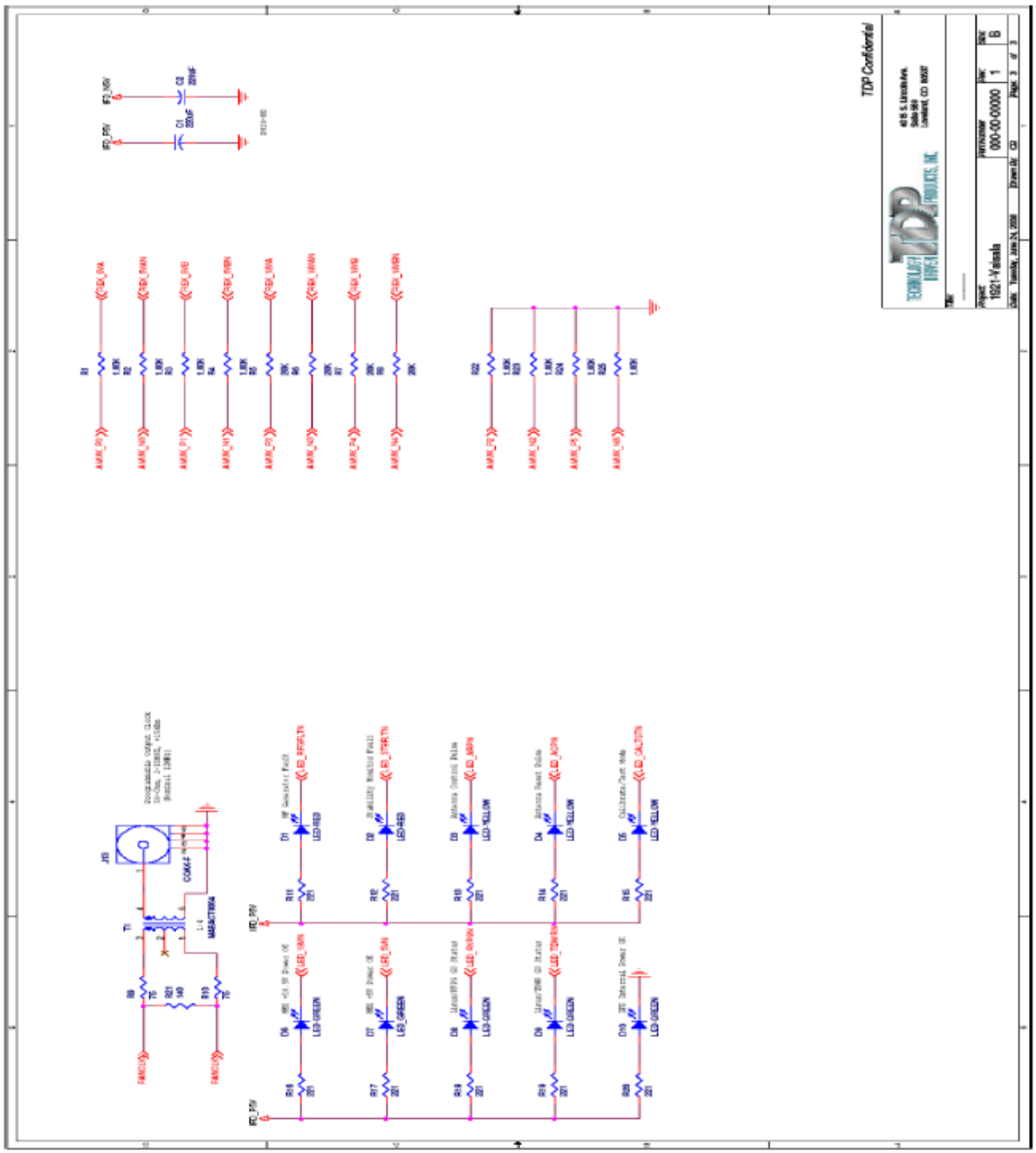


Figure 60 J13 Wiring Diagram

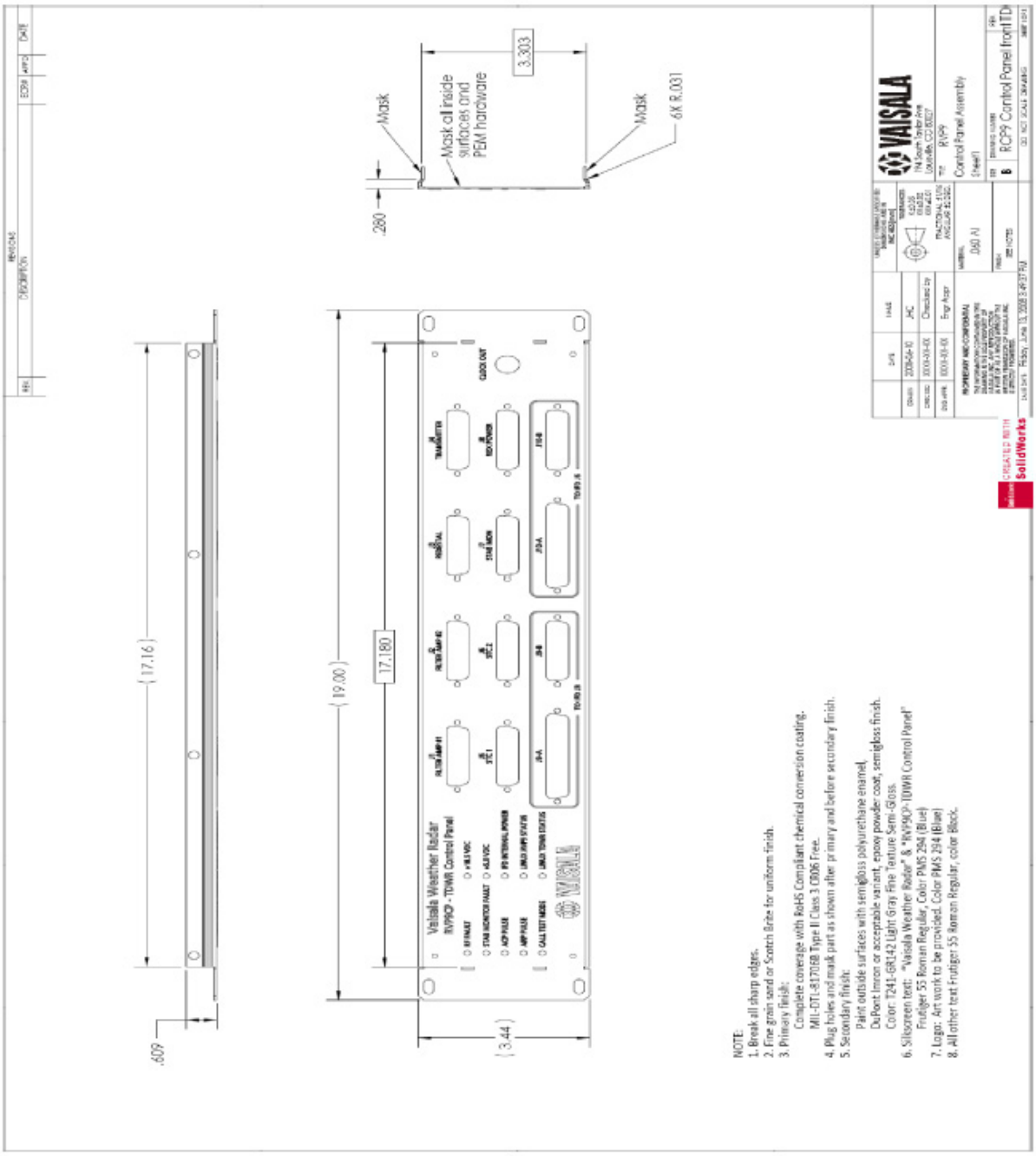


Figure 61 Control Panel Assembly

APPENDIX C

INSTALLATION AND TEST PROCEDURES

Customer:		
Serial No.	**Main:**	
	IFDR:	
Delivery Date:		
Radar Mfg./Type:		
Customer Engineer:		
Vaisala Engineer:		

C.1 Overview

These installation and test procedures are designed to assist Vaisala field engineers and customers with the installation and testing of the RVP900 on a radar system. Because these test procedures 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 *RVP900 Digital IF Receiver and Signal Processor User's Manual*. Do not write in the manual since it will be replaced with an upgrade; 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 and test procedures, your RVP900 is ready to connect to your software application, such as the Vaisala IRIS system. There are additional configuration and calibration procedures before using the RVP900 with the software.

Section C.1.1 Test Checklist is a checklist of the installation and test procedures. Contact Vaisala at support@vaisala.com if you have any problems or comments regarding this product or procedures.

C.1.1 Test Checklist

- ❑ C.2 Installation Check
- ❑ C.3 Power Up Check
- ❑ C.4 Setup Terminal
- ❑ C.5 Setup "V" Command (Internal Status)
- ❑ C.6 Setup "Mc" Command (Board Configuration)
- ❑ C.7 Setup "Mp" Command (Processing Options)
- ❑ C.8 Setup "Mf" Command (Clutter Filters)
- ❑ C.9 Setup "Mt" Command (General Trigger Setup)
- ❑ C.10 Initial Setup of Information for Each Pulse Width
- ❑ C.11 Setup "Mb" Command (Burst Pulse and AFC)
- ❑ C.12 Setup "M+" Command (Debug Options)
- ❑ C.13 Setup "Mz" Command (Transmitter Phase Control)
- ❑ C.14 Ascope Test
- ❑ C.15 Burst Pulse Alignment
- ❑ C.16 Bandwidth Filter Adjustment
- ❑ C.17 Digital AFC (DAFC) Alignment (Optional)
- ❑ C.18 MFC Functional Test and Tuning (Optional)
- ❑ C.19 AFC Functional Test (Optional)
- ❑ C.20 Input IF Signal Level Check
- ❑ C.21 Calibration and Dynamic Range Check
- ❑ C.22 Receiver Bandwidth Check
- ❑ C.23 Receiver Phase Noise Check
- ❑ C.24 Hardcopy and Backup of Final Setups
- ❑ C.25 RVP901 TxDAC Stand-alone Bench Test

Test Passed

For Customer_____ Date_____

For Vaisala_____ Date_____

C.2 Installation Check

Test Goal

Verify that the RVP900 is properly connected to the radar system and document some of the basic radar characteristics. There are differences for TWT/Klystron versus magnetron radar systems.

- ❑ Check that RVP901 Digital Receiver is properly connected to AC/DC line voltage according to supplied power converter.

Test Procedure

- ❑ RVP901 mounted in the radar receiver cabinet or other convenient location.
- ❑ RVP901 IF input connection. IF Frequency _____MHz.
- ❑ RVP902 Signal Processor chassis installed in (circle one): Rack Tabletop
- ❑ Distance between RVP901 and RVP902 Chassis _____
- ❑ Check that CAT5E ethernet cable is connected to J2 "Ethernet" of RVP901 and other end is connected to ETH0 port of RVP902 server chassis.
- ❑ Verify trigger connections.

For magnetron systems only, verify the following:

- ❑ IF burst pulse connected to J13 "ADC–E" of RVP901.

For Klystron or TWT systems only, verify the following:

- ❑ IF COHO is connected to J13 "ADC–E" of RVP901.
  Tx License Key Installed (circle one): Yes No

Test Passed

For Customer_____ Date_____

For Vaisala_____ Date_____

C.3 Power Up Check

Test Goal

Verify that the RVP901 and RVP902 properly powered up.

Test Procedure

The display terminal of the RVP902 shows summary power up messages.

1. Apply power to the RVP902 (or reset it) and the RVP901.
2. Verify the following:

- ❑ When power is applied to the RVP901, the red and green lights blink and then stay on.
- ❑ When the CAT5E cable is disconnected from either the RVP902 server or RVP901, the red light remains on and the green light turns off.

Test Passed

For Customer_____ Date_____

For Vaisala_____ Date_____

C.4 Setup Terminal

Test Goal

Verify that the TTY Setups are accessible and functioning properly.

Special Test Equipment

- Keyboard and mouse
- Monitor (KVM) are installed locally or on a remote computer (with DspExport running)

Test Procedure

Follow the procedure in Section 4.1 Overview of Setup Procedures on page 97 to access the TTY setups.

1. As operator, type "dspx".
2. Press the **Esc** button and verify that the RVP9 banner appears.
3. Type "Help" or "?" and verify that the list of commands displays.
4. Type "q" or "quit" to exit the menus.

Test Passed

For Customer_____ Date_____

For Vaisala_____ Date_____

C.5 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

- See Section 4.1.2 V and Vz – View Card and System Status on page 100

Test Procedure

1. Enter the TTY setups through dspx.
2. Issue the "V" command to display the internal status.

❑ Status information is correct per Section 4.1.2 V and Vz – View Card and System Status on page 100 and shows no faults.

NOTE	We will record the final values of all the settings at the end of the installation.

Test Passed

For Customer_____ Date_____

For Vaisala_____ Date_____

C.6 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

- See Section 4.2.1 Mc — Top Level Configuration on page 104

Test Procedure

1. Enter the TTY setups and type the "Mc" command.
2. Set all the values, as required, for your operation.

❑ Parameters set.

Test Passed

For Customer_____ Date_____

For Vaisala_____ Date_____

C.7 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

- See Section 4.2.2 Mp — Processing Options on page 106

Test Procedure

1. Enter the TTY setups and type the "Mp" command.
2. Set all the values, as required, for your operation.

❑ Parameters set.

Test Passed

For Customer_____ Date_____

For Vaisala_____ Date_____

C.8 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

- See Section 4.2.3 Mf — Clutter Filters on page 112

Test Procedure

1. Enter the TTY setups and type the "Mf" command.
2. Set all the values, as required, for your operation.

❑ Parameters set.

Test Passed

For Customer_____ Date_____

For Vaisala_____ Date_____

C.9 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 RVP900 can output up to 10 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 pulse width. The final timing adjustments are configured in Section C.15 Burst Pulse Alignment on page 374.

In the chart below, enter the purpose of each trigger, the nominal start time, and pulse width. Start times are relative to range zero (middle of the burst pulse). We recommend a nominal pulse width of 3 microseconds. This chart is used in Section C.10 Initial Setup of Information for Each Pulse Width on page 367.

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	Pulse Width
0	_____	_____usec	_____usec
1	_____	_____usec	_____usec
2	_____	_____usec	_____usec
3	_____	_____usec	_____usec
4	_____	_____usec	_____usec
5	_____	_____usec	_____usec
6	_____	_____usec	_____usec
7	_____	_____usec	_____usec
8	_____	_____usec	_____usec
9	_____	_____usec	_____usec

Reference

- See Section 4.2.4 Mt — General Trigger Setups on page 114

Test Procedure

1. Enter the TTY setups and type the "Mt" command.
2. Set all the values, as required, for your operation.

NOTE	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_____

C.10 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	The final values of trigger timing, FIR filter impulse response length, and bandwidth are 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 RVP900 supports up to four pulse widths (coded 0 to 3), although most transmitters typically support only two pulse widths. Record, in the chart 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

- See Section 4.2.5 Mt<n>— Triggers for Pulsewidth #n on page 117.

Test Procedure

1. Enter the TTY setups and type the "Mt #" command, once for each pulse width.
2. Enter the start time and widths for each trigger as documented in the chart in Section C.9 Setup "Mt" Command (General Trigger Setup) on page 365.

3. For all unused triggers, set the width to zero.
4. Enter the Maximum PRF from the chart above.
5. Set the initial impulse response length to 1.5 times the pulse width, and the initial pass bandwidth to the inverse of the pulse width.

❑ Parameters set.

Test Passed

For Customer_____ Date_____

For Vaisala_____ Date_____

C.11 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	An external AFC can be used rather than the RVP900 AFC, but is not recommended.

Klystron or TWT-based Systems

The COHO is measured instead of the burst pulse.

NOTE	Klystron systems that use a phase shifter should input the phase shifted COHO into the IFD, so that the RVP900 can digitally lock to the actual transmitted phase. For Klystron systems the AFC feedback loop is not used.

Special Test Equipment

- KVM connected

Reference

- See Section 4.2.6 Mb — Burst Pulse and AFC on page 126.

Test Procedure

1. Enter the TTY setups and type the "Mb" command.
2. Set all the values as required.

❑ Parameters set.

Test Passed

For Customer_____ Date_____

For Vaisala_____ Date_____

C.12 Setup "M+" Command (Debug Options)

Test Goal

Verify that the TTY setups for the Debug Options section are properly configured for the customer application.

Background

The RVP900 supports several test features that are configured in this section. For operational systems, the simulation features should be turned off. Vaisala recommends that the LEDs be set to "1:Go/Proc" so that the front panel red LED flashes during each processing cycle.

Special Test Equipment

- KVM connected

Reference

- See Section 4.2.7 M+ — Debug Options on page 136

Test Procedure

1. Enter the TTY setups and type the "M+" command.
2. Set all the values as required.

❑ Parameters set.

Test Passed

For Customer_____ Date_____

For Vaisala_____ Date_____

C.13 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

- See Section 4.2.8 Mz — Transmissions and Modulations on page 137.

Test Procedure

1. Enter the TTY setups and type the "Mz" command.
2. Set all the values as required.

❑ Parameters set.

Test Passed

For Customer_____ Date_____

For Vaisala_____ Date_____

C.14 Ascope 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 the radar and RVP900.

Special Test Equipment

- KVM connected

Reference

- See the **TTY Nonvolatile Setups** chapter in the *IRIS Utilities Manual.*

Test Procedure

1. As operator, type the "ascope" command and verify that the ascope utility displays correctly and starts to update.
2. Configure a "DEFAULT" startup and save it.
3. Exit and restart ascope.
4. Verify that the default startup is properly restored.

Test Passed

For Customer_____ Date_____

For Vaisala_____ Date_____

C.15 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

- See Section 5.3 Pb — Plot Burst Pulse Timing on page 143

Test Procedure

1. Use ascope to select the desired pulse width at a save PRF.
2. In the TTY setups, use the "Mb" command to turn off burst pulse tracking.
3. Set the transmitter to radiate.
4. 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.
5. Adjust the width of the burst pulse window using the I/i command to be slightly larger than the burst pulse (for example, ~50%).
6. Verify that the burst pulse power is in the range +1 to -12 dBm per the tabular display on the setup terminal.
7. Record the burst pulse power:

   Pulsewidth #0: _____ dBm

   Pulsewidth #1: _____ dBm

   Pulsewidth #2: _____ dBm

   Pulsewidth #3: _____ dBm
8. 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 RVP901). On a magnetron radar, if the AFC is not working, it is possible the IF frequency is outside the RVP901 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, the radar should be serviced by an experienced technician. If the burst pulse 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 by installing a fixed attenuator or amplifier.

Test Passed

For Customer_____ Date_____

For Vaisala_____ Date_____

C.16 Bandwidth Filter Adjustment

Test Goal

Set the band width filter for each pulse width.

Special Test Equipment

- KVM connected

Reference

- See Section 5.4 Ps — Plot Burst Spectra and AFC on page 148

Test Procedure

1. Enter the "Ps" command 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).
2. 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 -66 dB.
3. Repeat for each pulse width that is used (use the "Mt" command 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_____

C.17 Digital AFC (DAFC) Alignment (Optional)

Test Goal

Verify that the RVP900 DAFC output controls the STALO over the correct span.

Special Test Equipment

- Setup TTY

Reference

- See Section 3.4 Digital AFC Module (DAFC) on page 84

Background

The RVP900 implements an AFC based on the measurement of the burst pulse frequency. The DAFC connects to the TRIG–A sma port of the RVP901. It translates the AFC requests from the RVP900 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

1. Use the setup terminal to set the Digital AFC span as required in Section C.11 Setup "Mb" Command (Burst Pulse and AFC) on page 369.
2. Use the setup terminal and display scope in the Pb (plot burst) mode to verify that the burst pulse is properly centered. Any pulse width can be used.
3. Set to MFC using the "=" command, and adjust the control to the lowest setting using the "D" command. Record the results in the chart below.
4. Raise the control using “U” to within 0.1 MHz of the IF frequency. Record the results in the chart below.
5. Raise the control using “U” to the highest setting. Record the results in the chart below.

6. Verify that sufficient span is covered, and the power at the end points is sufficiently high to run the AFC loop.

	Voltage	Frequency
Midpoint:	_____A/D	_____A/D
Lower limit:	_____A/D	_____A/D
Upper limit:	_____A/D	_____A/D

Test Passed

For Customer_____ Date_____

For Vaisala_____ Date_____

C.18 MFC Functional Test and Tuning (Optional)

Test Goal

Verify that the Manual Frequency Control (MFC) is functioning properly.

NOTE	Skip this test if you are not using the RVP900 AFC.

Special Test Equipment

- KVM connected

Reference

- See Section 5.4 Ps — Plot Burst Spectra and AFC on page 148

Test Procedure

1. Enter the "Ps" command (Plot burst spectrum and AFC).
2. 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.
3. 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 reversed, then go to the "Mb" command menu and change the question "Burst frequency increases with increasing AFC voltage".
4. Using the U/u and D/d commands, verify the limits of the AFC tuning and fill in the chart below:

AFC %	Measured Freq (MHz)
-100%	_____
0%	_____
+100%	_____

   The 0% AFC value should be within approximately ±0.2 MHz of the center IF frequency (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.
5. Toggle the MFC mode to AFC by typing the "=" symbol and verify that the terminal indicator changes from "Manual" to "AFC".
6. Exit the Ps menu.

Test Passed

For Customer_____ Date_____

For Vaisala_____ Date_____

C.19 AFC Functional Test (Optional)

Test Goal

Verify that the AFC properly tracks the burst pulse frequency.

Special Test Equipment

- KVM connected

Reference

- See Section 5.4 Ps — Plot Burst Spectra and AFC on page 148

Test Procedure

1. Use the setup terminal to enter the Ps mode and observe the output on a display scope.
2. Verify the following:

- ❑ Verify the system is in AFC mode by checking that the text on the terminal for the AFC % output says "AFC".
- ❑ Verify 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" command); for example, in the range 29.985 to 30.015 MHz. If it is not in this range, verify that it moves within this range.
- ❑ Turn radiate off for 10 minutes and then turn the radiate back on. Observe to see if 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 to see if 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 pulse width 1.
  - ❑ For pulse width 2.
  - ❑ For pulse width 3.

Test Passed

For Customer_____ Date_____

For Vaisala_____ Date_____

C.20 Input IF Signal Level Check

Test Goal

Verify that the input signal level is optimized for the RVP901. This is done by observing the power in the noise using the "Pr" command.

Special Test Equipment

- KVM connected

Reference

- See Section 5.5 Pr — Plot Receiver Waveforms on page 163

Test Procedure

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.

1. Set the transmitter to radiate and elevate the antenna to >45 degrees to minimize the effects 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.
2. Use the setup terminal to enter the "Pr" command and use 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
3. Now remove the cable connecting the IF signal into the IFD. Again record the powers:
   Total: _____ dBm Filtered: _____ dBm
4. 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 IF Gain and System Performance on page 73).
5. 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_____

C.21 Calibration and 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 100 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.

Special Test Equipment

- KVM connected
- RF signal generator

Reference:.

- See Section 5.5 Pr — Plot Receiver Waveforms on page 163

Test Procedure

1. Run the radar and test signal generator for 20 minutes to allow proper warm-up of the system prior to the test. This allows the AFC to stabilize.
2. 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.
3. Use the setup terminal to enter the calibration gains, losses, and transmit power values.
4. Use the "ps" command in the dspx utility to put the RVP900 in manual AFC control by pressing the "=" key.

5. Connect the test signal generator to inject a signal at RF ahead of the LNA.
6. Set the signal generator to a value that is approximately 20 dB above noise and observe the scope plot.
7. Adjust the frequency of the test signal generator to make the frequency of the spectrum to the correct IF frequency. This can also be done by achieving the highest Filtered: _____ dBm
8. Open the ZAUTO utility and perform a calibration. Begin at +10-dBm and decrease the SIGGEN power by 10 dB for each sample. At the saturation point and noise floor use 1 dB steps to carefully define the roll-offs.
9. Select the lower and upper bounds of the linear portion of the receiver.
10. Press the **FIT** button within the ZAUTO utility.
11. Turn off the SIGGEN RF output and press the **NOISE** button.
12. Verify that the receiver dynamic range is greater than or equal to 95 dB.
13. 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.
14. Use the "ps" command in the dspx utility to put the RVP900 back in automatic AFC control by pressing the "=" key.
15. Record RVP900 calibration values.

NOTE	Do not increase the signal generator power, such that Total Power exceeds the Safe Total Power Limit for Pr command display +12 dBm, damage to the RVP900 A/D convertor could result.

Test Passed

For Customer_____ Date_____

For Vaisala_____ Date_____

C.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 RVP900. 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 (Section C.21 Calibration and Dynamic Range Check on page 381). For dual polarization systems, you expect to get a narrower bandwidth.

Special Test Equipment

- KVM connected
- RF signal generator

Reference

- See Section 5.5 Pr — Plot Receiver Waveforms on page 163

Test Procedure

1. Connect the test signal generator to inject a signal at RF ahead of the LNA.
2. 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.
3. Set the signal generator power to a value that is approximately 60 dB above noise and observe the scope plot.
4. Adjust the frequency of the test signal generator in 1 MHz increments to cover the whole range of the scope plot. Mark the total power measured on the plot in Figure 62 on page 384.
5. Verify that the 3dB point gives approximately 14 MHz of bandwidth.

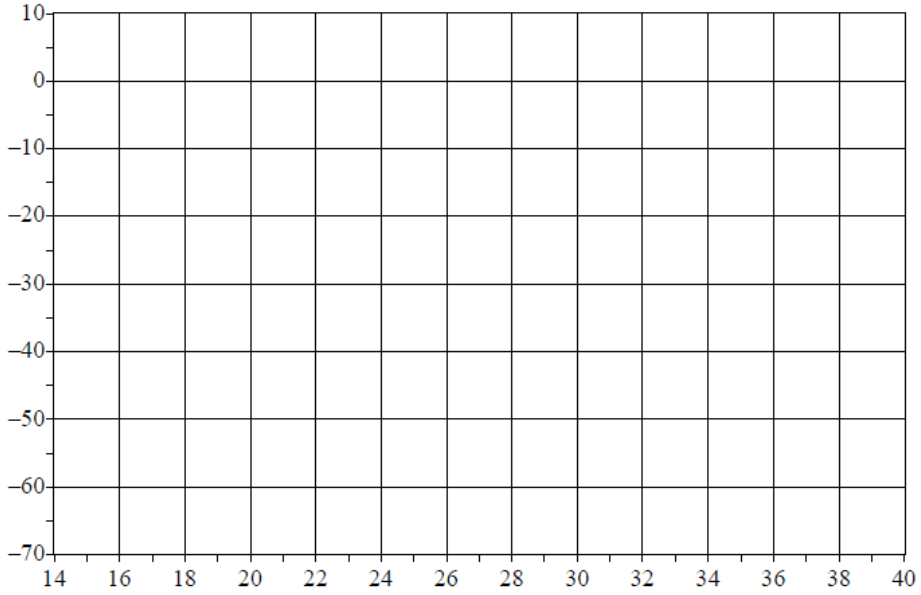


Figure 62 Graph of Total Power vs. IF Frequency

Test Passed

For Customer_____ Date_____

For Vaisala_____ Date_____

C.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 (refer to the *IRIS Utilities Manual*)
- Known clutter targets with no weather signal

Test Procedure

1. Configure the radar for normal operation expect pointing at a known clutter target.
2. 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
3. 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.
4. Select the range bin of the target. Record the Az, El, range and 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:_____
   Az:_____ El:_____ Range:_____ Phase Noise:_____
5. Try minor changes in Az, El, and Range to get the lowest phase noise. The goal is less than 1 degree within 20 km.

Test Passed

For Customer_____ Date_____

For Vaisala_____ Date_____

C.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

1. Start script logging with commands "cd /usr/sigmet/config/listings", "script RVP900.26feb09".
2. Enter the TTY setups and type the "??" command to list all the TTY setups.
3. Exit dspx and script logging with "exit".
4. 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
- rvp900.conf
- softplane_dsp.conf

Backup of /usr/sigmet/config directory

Test Passed

For Customer_____ Date_____

For Vaisala_____ Date_____

C.25 RVP901 TxDAC Stand-alone Bench Test

Test Goal

Verify that the RVP901 TxDAC electrical I/O is working properly in an isolated environment.

Special Test Equipment

- IRIS dspx utility
- IF signal generator
- Voltmeter

Test Procedure

1. Begin by running dspx and temporarily reverting to factory settings with "f".
2. In the Mb menu, choose an Intermediate Frequency that matches the RVP901 analog output filters (for example, 60 MHz) and a Blackman window.

   ```
   Receiver Intermediate Frequency: 60.0000 MHz
   Design/Analysis Window- 0:Rect, 1:Hamming, 2:Blackman :
   2
   ```

3. Enter the Pr menu and type “RRRTTZ” to move the starting plot range to 30 km, the plot interval to 20 msec, and zoom factor to x2.
4. Type two space characters to switch to the spectral display plot and exit the plot for now with “q”.
5. Setup the following in the Mz menu:

   ```
   Chan A - 0:Unused, 1:FixedFreq : 1, 2:TxWaveform: 1
   FreeRunning fixed frequency : 60.00000 MHz
   Output power level : 0.0 dBm
   Apply pulse-to-pulse phase modulation: NO
   Fixed relative phase offset : 0.000 Deg
   ```

   The output frequency should match the IF previously set in Mb.
6. Connect the RVP901 TxDAC-A output to the ADC-A port of the IFD. It should be connected directly to the IFDR SMA input connector; not through any bandpass filter.
7. Verify the plot shows a single strong spectral line at the selected Intermediate Frequency, and that any spurious signals are down at least 60 dB.
8. Repeat the steps 5 through 7 on the TxDAC-B output port.
9. Apply a 0 dBm SigGen waveform at the selected reference frequency (for example, 10 MHz) to the CLK IN BNC input of the RVP901.

10. Verify the V command shows Tx/Clk:Okay, indicating the RVP901 DAC is locking properly.
11. Reduce the SigGen output to -20 dBm and verify the locking is still okay.
12. Vary the frequency by 10 KHz (0.01 MHz) and check that the lock is lost.
13. Return to the center frequency and verify that lock returns.

Test Passed

For Customer_____ Date_____

For Vaisala_____ Date_____

APPENDIX D

RVP900 DEVELOPER'S NOTES

This appendix describes the software environment that is provided to third-party developer's who wish to customize the RVP900 (and RVP8) algorithms to meet their particular needs. This information only applies to the few organizations who have signed the Vaisala Software Developer's License Agreement. The vast majority of operators and users can skip this entire appendix, as it is not relevant to your operational and scientific needs.

D.1 Organization of the RDA Software

The Vaisala RVP900 is an open-architecture radar signal processor that uses Gigabit Ethernet to interface to the IFDR, which samples the data. The RVP900 software runs under standard RedHat Linux, and is developed and maintained using standard GNU tools (for example, gcc, gdb, make, etc.) Therefore, the RVP900 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 RVP900 is a solid open environment for many years to come.

The larger RDA software collection, which includes the RVP900 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
  * |-- dsp           DspExport daemon
  * |-- 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
    |-- pcicards      Board-level support for RVP8/Rx, RVP8/Tx and I/O-62 cards
  * |-- rcp8          Root point of RCP8
  * |   |-- core        CORE RCP8 code
  * |   |-- open        OPEN RCP8 code
  * |   `-- site        SITE RCP8 code (customized by developers)
  * |-- rdautils      Rdadiasg program
    |-- 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 compiling RVP8 compute processes
    |-- rvp9main      Root point of RVP900 main threads
    |   |-- core        CORE RVP8 code (not delivered to customers)
  * |   |-- open        OPEN RVP8 code (available to developers)
  * |   `-- site        SITE RVP8 code (customized by developers)
  * |-- rvp9proc      Root point of compiling RVP900 compute processes
    |-- rvpxproc      Root point of compute code shared between RVP8 and RVP900
    |   |-- core        CORE PROC code (not delivered to customers)
  * |   |-- open        OPEN PROC code (available to developers)
  * |   `-- site        SITE PROC code (customized by developers)
    |-- ts            Root point of Time Series tree
  *     |-- archfmt     time series archive output formatting library
        |-- archive     tsarchive GUI code (not delivered to customers)
        |-- archlib     Library for TS related functions (not delivered)
        |-- exec        tsexec code, does the work of tsarchive (not delivered)
  *     |-- export      tsexport and tsimport code
  *     |-- switch      tsswitch code
  *     |-- view        tsview code
```

Figure 63 RDA Software Collection Tree

All software in each of the directories marked with an asterisk * is provided to licensed developers.

The open portions of software contain all of the hooks that scientific programmers need to add/modify the processing algorithms. This comprises about 15% of the roughly 105 thousand lines of code that make up the complete RDA system. Vaisala does 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.; the material that might unfairly be used to clone the RDA products.

D.2 RVP Overall Code Organization

The remainder of this appendix focuses on the RVP8 portion of the RDA development tree. The various RVP8 internal APIs allows 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. See Figure 64 on page 392 for the RVP8 hardware and software organization.

- **PCI Board Firmware**—The code that runs within the Field Programmable Gate Array (FPGA) chips on the PCI cards. The FPGA code 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 500 ms interval, all of the jobs get done with no loss of data.
- **Linux Kernel Module**—This module is *insmod* at system boot time, 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**—The 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 (see Chapter 7, Host Computer Commands, on page 255).

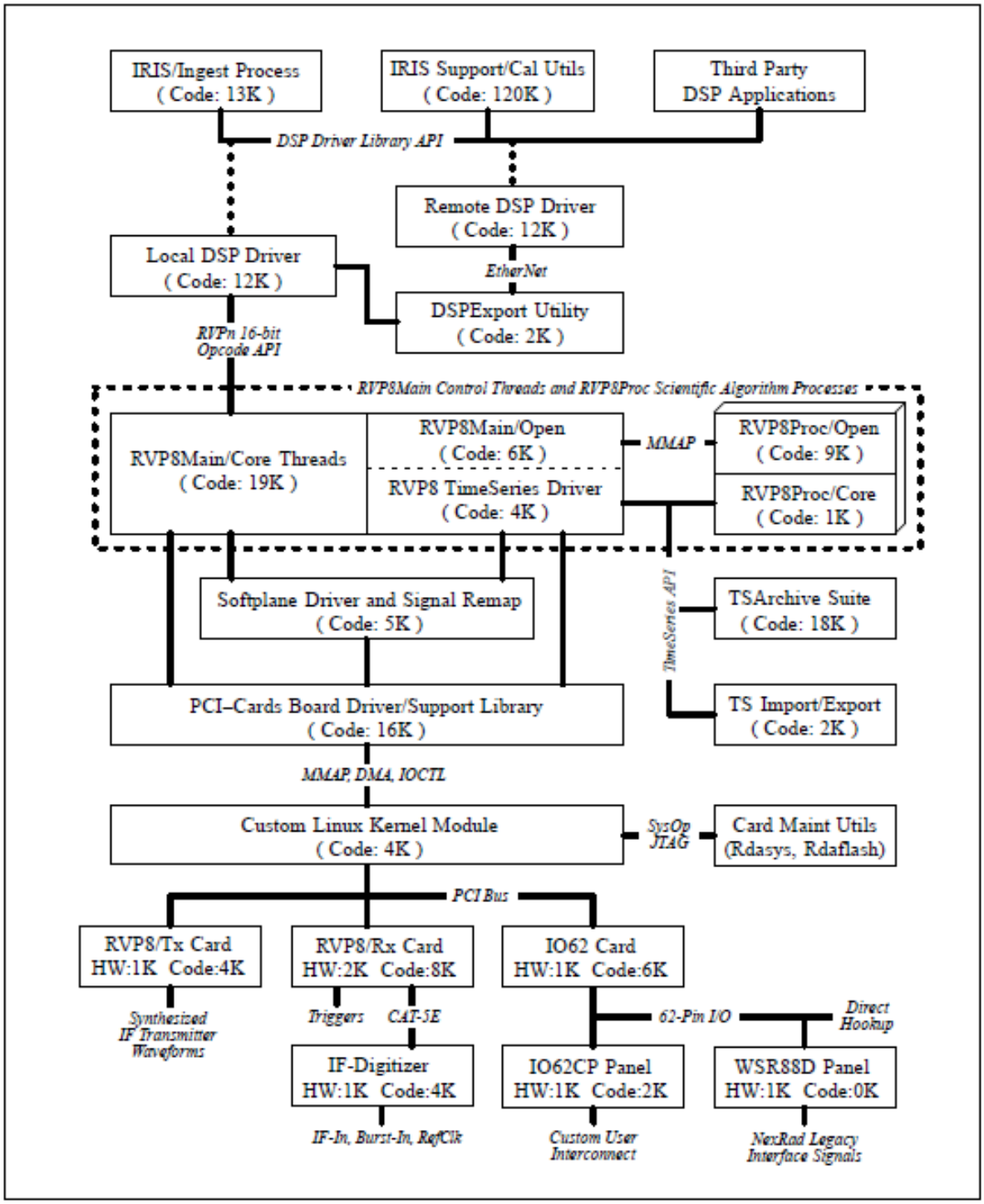


Figure 64 RVP8 Hardware and Software Organization

- **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 6, Processing Algorithms, on page 179) within the RVP8. These are the actual "number crunchers" that 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 create one RVP8/Proc process per available CPU on an SMP machine. Most of your customization of the RVP8 algorithms occur in these processes.

Virtually all of the RVP8/Proc code is released to licensed developers as source code. There are open APIs for the RVP8/Main that allow you to build custom trigger patterns, phase sequences, etc., on the transmitter side. The kernel source is available so that developers can run within their own customized Linux environment.

D.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 are using the exact same rvp8main/open and rvp8proc/open source directories that our licensed developers receive. All bug fixes, enhancements, new processing modes, and maintenance work is 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. 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 only be used 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 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 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.

D.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 *Vaisala 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 replaces 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
```

You are rebuilding your code under the new header files that were just installed during the upgrade. 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 *Vaisala RDA Release Notes* to find out what changes need to be made. The RVP8 will not let you run with incompatible headers. Some You may receive a 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>
```

D.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 finds the appropriate kernel module for your system and installs it automatically. However, if you are running with a custom Linux kernel, you have to rebuild your RDA kernel module whenever its version has changed since the last time that you built it. The RVP8 does not run with incompatible modules and reports an error:

```
RVP8/Main: KernelMod mismatch
<Found SIGMET RDA kernel module V3.3, but V3.2 is required>
```

1. Install the headers, source, and objects from the release cdrom
2. Copy the rda source over to your development tree.
3. Compile.

NOTE	If you would like to cross-compile for a different kernel version, 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 is automatically loaded on the next boot, or it can be manually installed (again, as root) using:

```
# rdasys stop
# rdasys start
```

D.3 Debugging and Profiling Your Code

Although the complete RVP8 is a rather complex multi-thread and multi-process system, it is still a user-level application running under the Linux operating system. Most of the custom code that you develop can be debugged using tools that are already familiar to Linux/C/GNU programmers.

D.3.1 Monitoring Opcode/Data Activity: -exposeIO

The RVP8 is generally controlled by higher level application such as ascope, iris, dspx, etc., which communicates with the RVP8 through the *IRIS DSP Driver Library* using the opcodes (see Chapter 7, Host Computer Commands, on page 255). 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 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 arrives, along with the two input words, which are expected by

that opcode. No output words are generated by SNOISE. The GPARM opcode is received and the 64 output words that it produces are displayed.

Watching the opcode-level activity of the signal processor is helpful when debugging both custom driver code and custom opcode handlers that you might write for the RVP8.

D.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 Direct Memory Access (DMA) bus cycles that are initiated by the RVP8/Main threads. This is an implementation detail that developer's can 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
```

We see 193 pulses of packed 32-bit (I,Q) data words being transferred from the first Rx card, followed 3 ms 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. 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.

D.3.3 Showing Coherent Processing Intervals: -showCPIs

Coherent Processing Intervals (CPIs) 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. Here ia sample printout (while running *ascope)*:

```
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 (ms). In the example, the PRF was 500 Hz, hence the 64 pulses span a total time of 126 ms. 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.

D.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 is 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 Section D.5 Real-Time Control of the RVP8 on page 405 and 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(-)
```

We see the first invocation requests a callback in 100000 counts of the 1 MHz timer. The "Rq" (request) field of timer #0 shows the *iTimerWait* duration. The *iTVWaitOnNext* field causes the 1 MHz 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 1 MHz timer (2.5 ms), and a delay of five counts on the Trigger timer. The RVP8 trigger is selected as the clock source for timer #1.

NOTE	The clock source messages are only printed 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 2.5 ms; 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 2.5 ms expires.

D.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 well-crafted tools that are provided on all Linux systems.

Code that you write for the RVP8/Main threads can be debugged using:

```
$ cd /home/operator/src/rda/rvp8main
$ make -j2
$ cd open
$ ddd rvp8
```

The SITE and OPEN code is first compiled in the private development tree, and then we 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 causes the Makefiles to build non-optimized code, which generally behaves more smoothly in a symbolic debugger.

You can 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 causes *ddd* to break on the first executable line after all of the dynamic library references have been resolved. This is a good way to get started, because you may 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* are also sent to the RVP8/Proc child threads. They are likely to cause them to behave improperly. Use this technique only for quick ventures into the main threads, and preferably with -procs 0.

Debugging the RVP8/Proc code is similar, except that we do not want it to automatically start from the main threads. Therefore, in one X-Term window type:

```
$ rvp8 -noFork
```

which starts up the RVP8/Main threads, but pauses at the point that the RVP8/Proc process would normally be created. The *-noFork* forces the subprocess count to one, as if you had included *-procs 1* on the original command line.

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 the debugger. Typing "run" in *ddd* starts the subprocess, and when it has finished its initializations, the RVP8/Main continues 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 the *ddd* "run" command. 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 does not interfere with the other.

D.3.6 Finding Memory Leaks with valgrind

The *valgrind* profiler is 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*.

Valgrind can be downloaed from http://valgrind.kde.org. *Valgrind* is easy to use, because you run it on the executable being debugged in the same way that *ddd* was used in the previous section. There are no special compiling or linking requirements. To debug the RVP8/Proc process, use:

```
$ cd /usr/sigmet/src/rda/rvp8proc/open
$ valgrind —tool=memcheck rvp8proc
```

D.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 need to make sure that only one of them is forked by including "-procs 1" in the original startup command. Otherwise, each sub-process attempts to create the same *gmon.out* and chaos follows. If you forget to specify the solitary process option, the RVP8 forces it upon you anyway, but with a warning message. The RVP8 does 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 allows *gprof* to find its symbols, for example:

```
$ cd /home/operator/src/rda/rvp8proc/open
<Run the RVP8 for a while...>
$gprof rvp8proc -I ../site
```

D.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 containing about 100 lines of boilerplate top-level definitions. Creating your new major mode begins by copying this prototype code into separately named files in

the *rvp8main/site* and *rvp8proc/site* areas, and then making 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 ) {}
```

For execution, call your custom initialization routines from one of these predefined user stubs:

```
void mtInitMajorMode_user1( void ) {initMajorMode_myCustomMode() ; }
```

D.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    ;
} ;
```

Users of object-oriented languages 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 code organization was patterned after the elegant and intricate implementation of driver module replacement within Linux.

The *initMajorMode()* routine, which creates each major mode in the first place, is not part of this list because it is separately invoked (see Section D.4 Creating New Major Modes from Old Ones on page 402) 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*.

Here are a few fundamentals:

- **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 when *exitMajorMode()* is called; that is, if *exitProcSection()* needs to be invoked, that always happens first.
- **initProcSection_f** and **exitProcSection_f** —Pair of routines that are 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. The major mode itself does not change at these transitions, but these functions allow the current major mode to interact with the timeseries data acquisition as needed.

D.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 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 depends on the kernel version, the type of motherboard being used, and the nature of the other processes that are competing for CPU time. Vaisala has found that intense disk and network I/O are among the worst contributors to high scheduling latency, sometimes amounting to as much as 25 ms of variation. These uncertainties can usually be absorbed by proper coding of the thread.

D.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 is 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 occurs, and each can be based on:

- A specified number of counts of a free running 1 MHz 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 occurs 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 1 MHz counts, then at 1 KHz PRF the callback occurs in 5 ms. The callback sequence can be terminated at any time within the current PROC section 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 events 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.

D.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. 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 later controls 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 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 ;
    ```

The interrupt latencies are absorbed differently in these two cases. For angle syncing, 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 compute a shorter future time in such cases. For Dual-PRF triggers, we 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.

D.5.3 Example: Real-Time 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*:

1. Change the default USER1 major mode init routine to call the demo histogram:

   ```
   void mtInitMajorMode_user1( void ) {
   initMajorMode_demohist() ; }
   ```

2. Include the header file consisting of a single line prototype for the above function:

   ```
   #include "demohist.h"
   ```

3. Compile, install, and run your changes with:

   ```
   $ cd /usr/sigmet/src/rda/rvp8main
   $ make clean
   $ make -j2
   $ rvp8 -noDiags
   ```

   The RVP8 starts up normally.

4. Run the **ascope** utility and request USER1 major mode from the *Gen Setup* menu.

   You should 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 ms or every five triggers, whichever is fastest. Moreover, a histogram is printed every ten seconds showing the distribution of times. Some interesting things to try:

- Type a low PRF (~500 Hz) into **ascope** and verify that the interrupts cluster around 2.5 ms. Then type a high PRF (~5 KHz) and verify that histogram peaks at 1 ms.
- 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.

- Verify that selecting some other major mode in **ascope** results in the messages:

```
Exiting from demohist PROC section
Exiting DemoHist major mode
```

D.6 Customizing the (I,Q) Data Stream

D.6.1 Defining the FIR Matched Filter

D.6.2 Applying Raw Pulse Corrections

D.6.3 Inserting UserIQ Header Bits

D.7 Customizing the Front Panel Display

D.8 Adding Custom DSP/Lib Opcodes

D.9 Using the Softplane for Physical I/O

D.9.1 Softplane Programmer's Model

D.9.2 Reducing Unnecessary PCI Traffic

D.10 Handling Live Antenna Angles

D.11 Creating Custom Trigger Sequences

D.11.1 Defining Trigger Waveshapes

D.11.2 Defining Trigger PRT Sequences

D.11.3 Polarization and Phase Control

D.11.4 Example: Adding PRT Micro-Stagger

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, modify *rvp8main/site/mt_user.c* as described in Section D.5.3 Example: Real-Time Interrupt Histogram on page 408. In addition, 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 1 KHz PRF and with a delayed sweep of 1 ms, you should see a flurry of "second bang" triggers whose leading edges jitter over a 5 s 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 100 Hz in order to limit the rate at which the live headers were printed. A 72 MHz IFD was used in this measurement, hence the nominal time to the previous and next 100 Hz pulses would be 720000 clock ticks. However, because of the micro-stagger, the actual interpulse PRTs are seen to vary.

D.12 Determining CPI's and Ray Boundaries

D.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*. Refer to 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.

D.13.1 Reader and Writer Clients

The Timeseries API is entirely stateless and passive from a reader client's point of view. 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 100–400 ms delayed from their actual time of arrival. As such, it is not 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 checking at 10–20 Hz presents a negligible CPU load, does not risk losing any data, and matches the PCI/DMA burst transfers.

D.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 must also specify the client type you are. Readers are generic, but for writers, there are several choices because we need to switch sources, and we 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

D.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.

If the RVP is reconfigured in some way, this causes 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.

D.13.4 Using Memory Bandwidth Effectively

It will be a waste of effort to read each pulse of data individually. So a smart reader sleeps 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.

D.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 E

TIME SERIES RECORDING

E.1 Overview

The time series (TS) recording feature of the RVP900 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 RVP900 or on a separate archive host through a gigabit network.

The time series recording is built on several software components:

- **RVP900 TS API**—Where time series reside in memory in the RVP900.
- **Tsexport**—Grabs time series from the TS API and sends them over the network through 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 RVP900 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 RVP900 processing algorithms.
- **Tsview Utility**—Used to see diagnostic printouts of stored time series files. The 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. Customers are free to write their own TS record/playback software and still take advantage of the network features such as **tsimport** and **tsexport**.

E.2 TS Record/Playback Software Architecture

E.2.1 General Architecture

The TS record and playback features on a local RVP900 and a remote archive host, share a common software architecture. The most general case is shown in Figure 65.

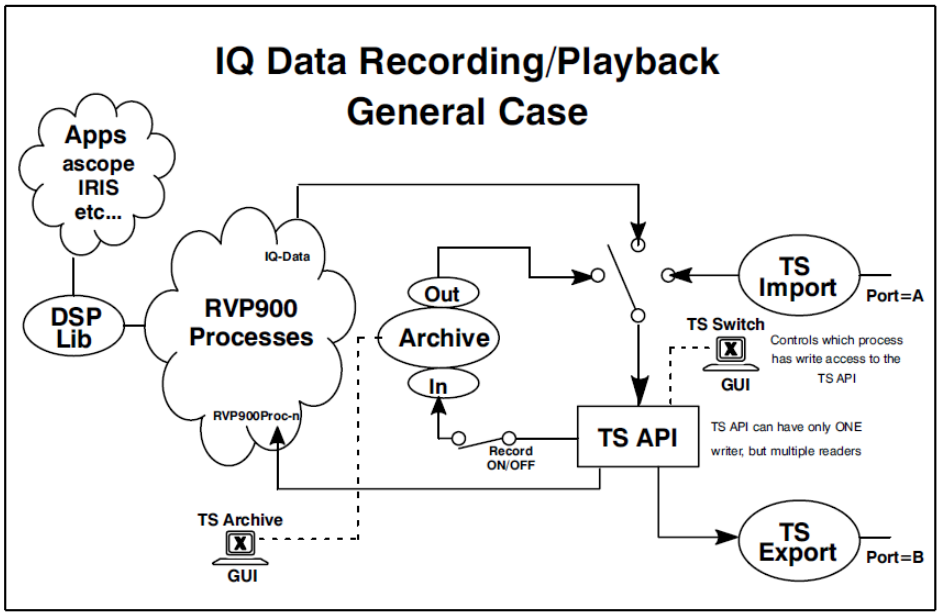


Figure 65 IQ Data Recording/Playback General Case

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 RVP900 and a separate archive host operating in record or playback mode.

E.2.2 Description of Processes

The following are descriptions of the various processes shown in Figure 65:

- **TS API**—Central to the architecture, 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 RVP900

  - From a local disk archive
  - From a remote network host (via **tsimport**)

- **Tsswitch—**GUI that allows the user to select the sole TS writer from among these three possible sources. This GUI is described in Section E.4 TS Switch Utility on page 423.
- **Tsarchive**—Process that 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 Section E.5 TS Archive Utility on page 424.
- **Tsimport and Tsexport**—Processes that provide the ability to receive/send time series over a network. 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 two 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, with little or no competing traffic.

  The output is through 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 Section E.3 Installation & Configuration on page 420.
- **RVP900 Processes**—Collection of processes that are only present on an actual RVP900 machine. The two important functions are:

  - IQ–Data: writes real time TS to the TS API. These are collected from an IFDR.
  - RVP9Proc-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. A remote archive host will most likely not be an RVP900. In this case, these processes do not exist.

E.3 Installation & Configuration

Vaisala recommends the use of TS recording in two configurations. The simplest (but least flexible) configuration is where the RVP900 also serves as the archive host.

NOTE	A separate partition should be created to prevent system crashes.

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, the RVP900 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. The UDP ports for the sending and receiving system also have to be configured.

E.3.1 Required Software

The TS archive software is part of the Vaisala's RDA release, which is installed by default on all RVP900s and RCP8s, but it is not installed on IRIS systems.

RVP900/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

E.3.2 Configuring UDP Ports

The UDP ports can be configured by editing the tsimport and tsexport scripts found in /usr/sigmet/config_template/rc/d/. These files should be copied to /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 ports.

Vaisala recommends the following ports:

RVP900	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 RVP900 end, so you need to edit the archive host's 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.

E.3.3 Configuring Automatic Startup of tsimport and 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
```

E.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 to the end:

```
net.core.rmem_default = 1000000
net.core.rmem_max = 4000000
```

Reboot for this to take effect.

E.3.5 tsimport and tsexport from the Command Line

From the command line, tsimport and tsexport can also be run. They 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:<broadcast–address>
   –daemon – Run as daemon
   –debug – Print diagnostics
   –port:<port> – Specify the port number to use
   All other options ignored
```

E.4 TS Switch Utility

To start the tsswitch utility as operator, type:

```
$tsswitch
```

In Section E.2 TS Record/Playback Software Architecture on page 418, 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:

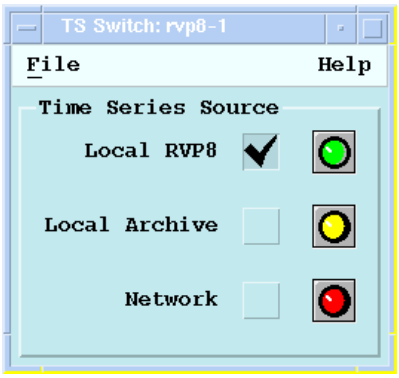


Figure 66 TS Switch Utility

- **Local RVP900**—Real time IQ from the IFDR. This setting is available only on an RVP900. This would be the choice for normal data processing, local RVP900 archive of real time IQ or export of real time IQ over the network.
- **Local Archive**—Used to extract time series from the local disk archive. On an RVP900 these time series could be processed. On a separate archive host, these time series could be sent to an RVP900 for processing or perhaps a custom user application.
- **Network**—Used to collect time series from a networked RVP900 or archive host, via the TS Import process. This setting can be used either by an RVP900 (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, but not currently selected. If you select this source, it changes 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 remains red until it is enabled.

In the case of a separate archive host, the "Local RVP900" choice would always show as red, since there is not a local RVP900.

The "Local Archive" case and the "Network" case are possible for both an RVP900 and a separate archive host.

E.5 TS Archive Utility

To start the utility, type (as operator):

```
$tsarchive
```

The **tsarchive** utility enables the end user to record and playback archived data. The utility provides a GUI that includs access to the TS switch utility, playback and record modes, and a complete archive file inventory for managing disk space.

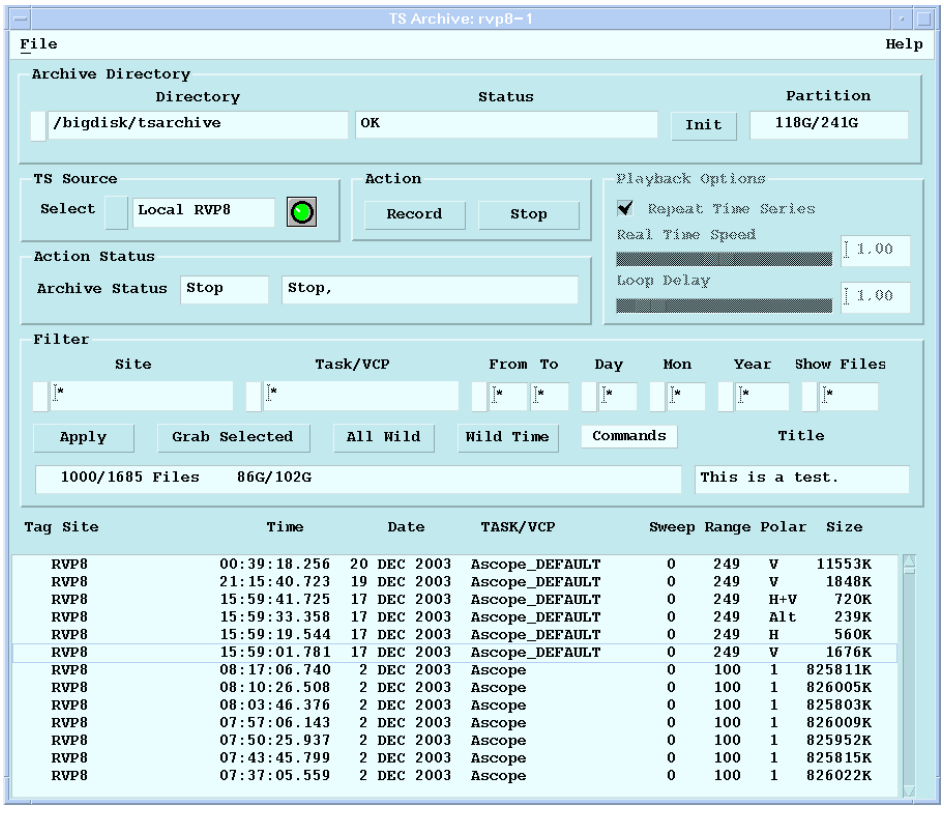


Figure 67 TS Archive Utility

E.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.

E.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 Section E.4 TS Switch Utility on page 423.

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 updates with TS file information. The 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 is only enabled when Local Archive has been selected from the TS Source Area.

- **Repeat Time Series**—If selected, repeats 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.

E.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 volume control procedure (VCP)) 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. A pop-up list of values is available.

Day, Month, Year

Enter a day, month. or year to narrow the list of products by date. A pop-up list of values is available.

Show Files

The show files field lets you control how many files to include in the list.

Apply

Click the **Apply** button to update the file list, based on your selection criteria. The **Apply** button is only enabled if the **Filter** button is selected. If the **Apply** button is grayed out, click the **Filter** button.

Grab

If a product is selected, the **Grab** button inserts a selected files information into the filter fields.

All Wild

Click the **All Wild** button to return all the fields to the wildcard character.

Wild Time

Click the **Wild Time** button to only change 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 the "Delete" command, every TS archive file could be deleted An "Are you sure" prompt is provided to verify that you want to do this.

E.5.4 TS Archive Log Area

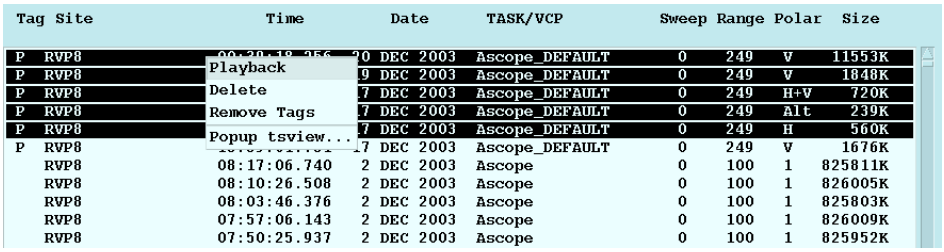


Figure 68 TS Archive Log Area

Tag & Right-Click Menu

When you right-click on any selected file or group of files (as shown in Figure 68), you are prompted with four commands:

- **Playback**—Marks the files with a P in the **Tag** Column. When the **Playback** button is selected, only the tagged files are 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 is selected for deletion.
- **Remove Tags**—Clears the D or P tags.
- **Popup tsview**—Provides easy access to file information in another window.

Site Field

The site field displays which TS site was used to create the data.

Time and Date Fields

The time and date fields display the UTC time and date when the data were acquired.

Task/VCP Field

The task/vcp field displays the name of the associated tasks or ASCOPE file used for controlling the RVP8.

Sweep Field

The sweep field displays the number of sweeps (for example, 360 degrees) in the Task/VCP.

Range and Polar Fields

The range and polar fields display the range and polarization of the TS data.

Size

The size field displays the size of the file.

E.6 Specific Software Application Examples

There are four specific example applications described here:

- TS recording on a local RVP900, see Section E.6.2 Case 1: TS Recording on a Local RVP900 on page 431.
- TS recording on a separate archive host, see Section E.6.3 Case 2: TS Recording on Separate Archive Host on page 432.
- TS playback on a local RVP900, see Section E.6.4 Case 3: TS Playback on a Local RVP900 on page 433.
- TS playback from a separate archive host to an RVP900, see Section E.6.5 Case 4: TS Playback from a Separate Archive Host to an RVP900 on page 434.

These are the only applications that are supported. In the example descriptions, unused processes are omitted for clarity.

For reference, the case of an RVP900 in normal operation is shown in Figure 69 on page 430.

E.6.1 RVP900 in Normal Real-Time Operation

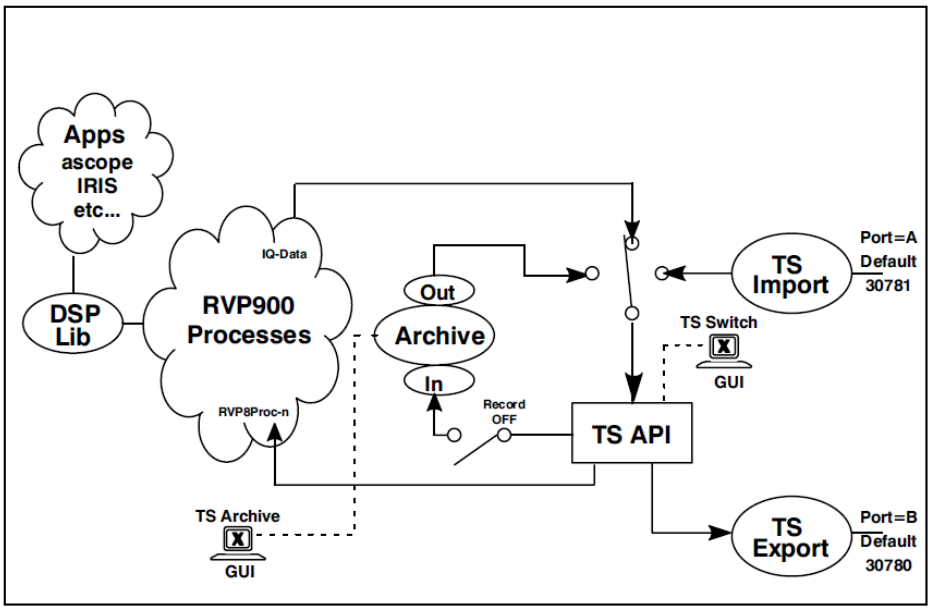


Figure 69 RVP900 in Normal Real-Time Operation

In this case, the TS Switch is set to write real-time data from the IQ-Data process to the TS API. The RVP9Proc-n extracts time series data and processes it. The configuration information is obtained from and data are passed to the various user applications through the DSP Lib functions. The TS Export process, if started, can extract data simultaneously from the TS API.

Utility Settings

TS Switch: Local RVP900
TS Archive: N/A

E.6.2 Case 1: TS Recording on a Local RVP900

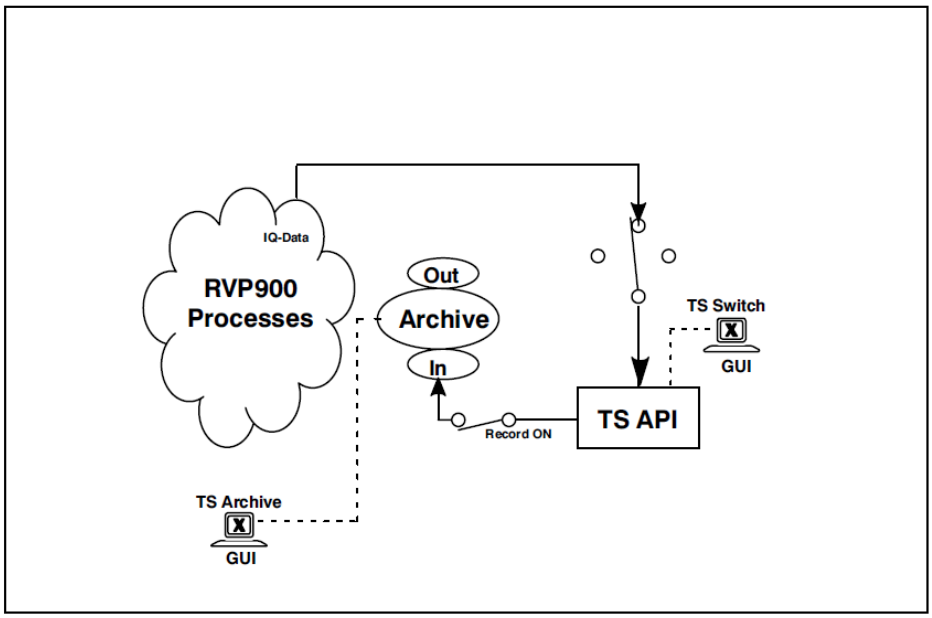


Figure 70 TS Record on Local RVP900

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. The RVP900 may still do its normal data processing tasks, as shown in Section E.6.1 RVP900 in Normal Real-Time Operation on page 430, while TS data are being recorded.

Utility Settings

TS Switch: Local RVP900
TS Archive: Record

This configuration records to a local disk on the RVP900. Typically only a 20 GB disk is provided with a standard RVP900 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.

E.6.3 Case 2: TS Recording on Separate Archive Host

This is the recommended recording configuration for TS recording.

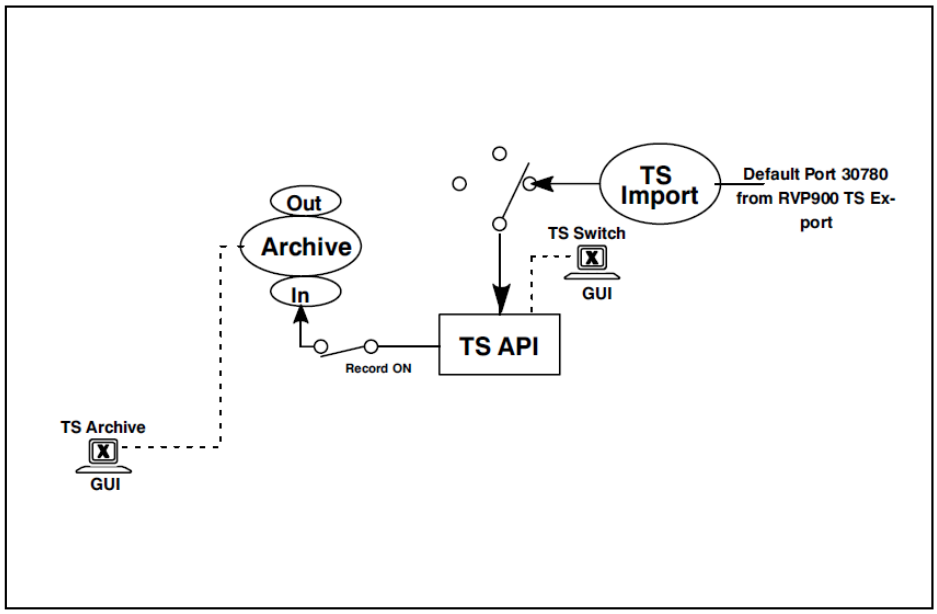


Figure 71 TS Record on Separate Archive Host

In this case, the 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 through UDP broadcast by the TS Export process on a networked RVP900.

Utility Settings

RVP900
TS Switch: Local RVP900
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 RVP900. There can be multiple archive hosts on the network.

E.6.4 Case 3: TS Playback on a Local RVP900

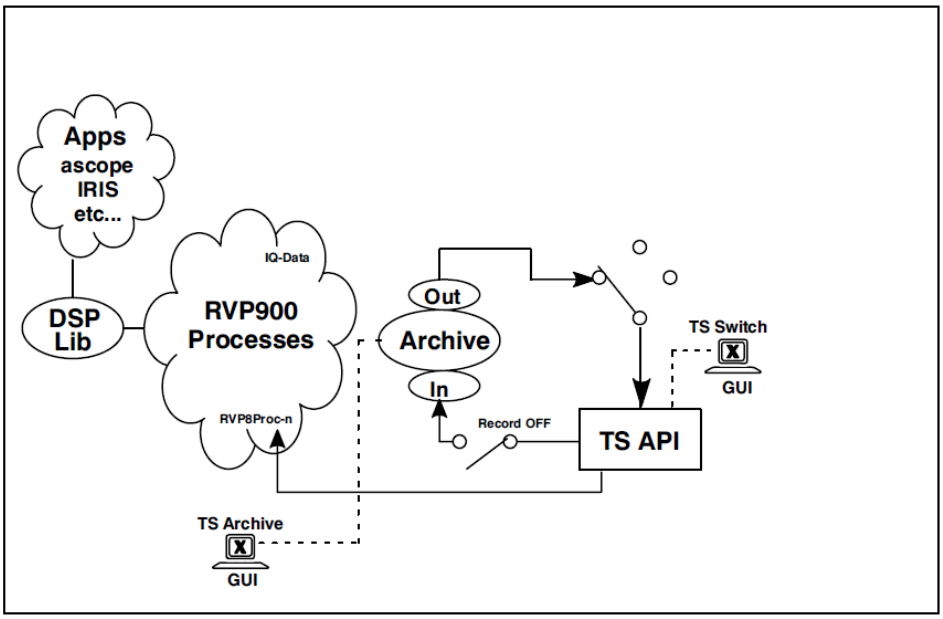


Figure 72 TS Playback on Local RVP900

In this case, the TS Switch is set to write data from the TS Archive to the TS API. The RVP9Proc-n processes then reads the time series data from the API. This is essentially identical to the real-time normal operation case shown in Figure 69 on page 430, except that the archive is the source rather than the real-time operation.

Utility Settings

RVP900
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.

E.6.5 Case 4: TS Playback from a Separate Archive Host to an RVP900

This is the recommended mode of operation.

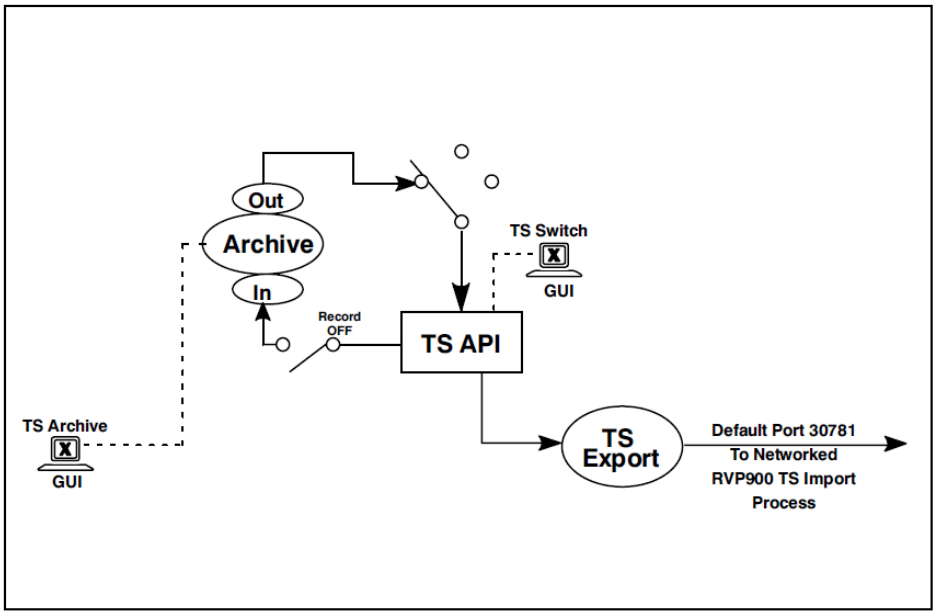


Figure 73 TS Playback from a Separate Archive Host

In this case, the TS Switch is set to write data from the TS Archive to the TS API. TS Export sends these through a UDP broadcast over the network to an RVP900 for processing.

Utility Settings

RVP900
TS Switch: Network
TS Archive: N/A
Archive Host
TS Switch: Local Archive
TS Archive: Play

The diagram for the corresponding RVP900 would be identical to Figure 72 on page 433, except the TS Switch would be set to write data from the TS Import process.

E.6.6 Quick Guides

The following steps should be used in conjunction with the software specific application examples described in Section E.6.1 through Section E.6.5 .

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 (right-click and select Playback) and mark them for playback.
- Select any playback options.
- Click the **Playback** button.

E.7 Ascope Playback Features

The ascope utility is a full-featured, stand-alone signal processor configuration and plotting utility. When an RVP900 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, refer to the *IRIS Utilities Manual*. Figure 74 illustrates the differences in the meanings of various menu fields when the RVP900 is in playback mode.

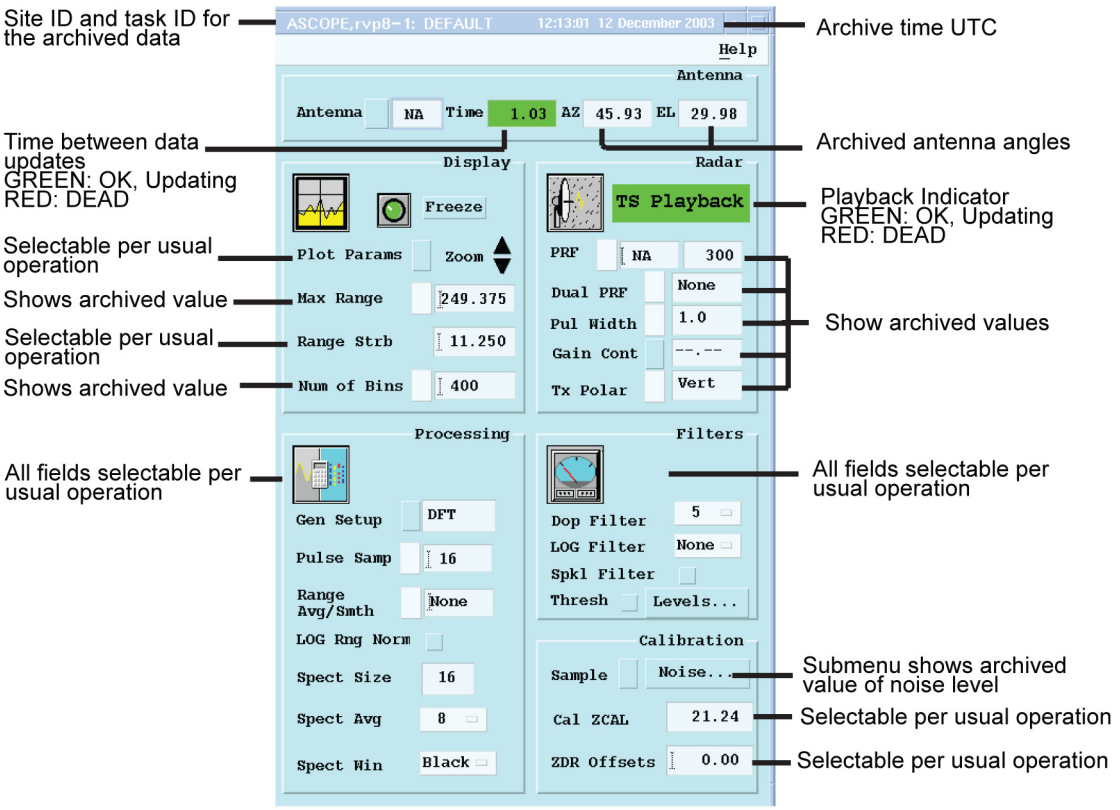


Figure 74 Ascope Differences during RVP900 TS Playback

When the RVP900 is in playback mode, versus processing real-time data, the major difference is that some parameters were fixed at the time, when the data were recorded. Some 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. An example is the processing major mode and clutter filtering. The user may freely select these while the RVP900 is in playback mode.

There are several ways to use ascope during RVP900 playback:

- The archive can be on the local RVP900.
- The archive can be on a separate archive host.

It does not matter where ascope is run, that is, either on the local RVP900 or on a networked host computer through DspExport.

E.7.1 Archive on Local RVP900

Utility Settings

RVP900
TS Switch: Local Archive
TS Archive: Play

Network Note

In most cases, you are not sitting up at the radar, so you 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
- 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, then start ascope. Ascope starts up in the playback mode and updates 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.

E.7.2 Archive on Separate Archive Host

Utility Settings

RVP900
TS Switch: Network
TS Archive: N/A
Archive Host
TS Switch: Local Archive
TS Archive: Play

E.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

"Source of recorded angles" = "RVP Tags"
"Source of recorded time" = "RVP Tags"

General pop-up > Modes and Protocols

"Timezone for data recording" = "UTC"

General pop-up > Scanning Options

"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 displays a 2-digit number containing the playback version after the site code. Original data with a playback version of zero, does not include the number. The ingest filenames all have appended a "V" followed by the 2-digit playback number.

To play back data:

1. Launch tsarchive on your RVP900 machine.
2. Click the **TS Source** button to launch the tsswitch GUI.
3. In tsswitch, select "Local Playback".
4. In the tsarchive Playback Options section, clear the **Repeat** button.
5. Set the **Version** to a unique non-zero value.

6. Select from the inventory the file or files you wish to playback, and set the **P** bit for those.
7. If you are playing back a multiple sweep volume scan, there is one file per sweep.
8. Now using the **IRIS** menu bar, bring up the **Task Scheduler** menu. Make sure that it says **Passive** on the top menu bar.
9. Select the task that matches the data.
10. Select **Go/Schedule** for that task; it toggles to **running.** Wait for data to arrive.
11. On the **tsarchive**, click the **Playback** button.
12. On the **Task Scheduler** menu, set the task to **Stop (when done)**, so that it stops after one time.

E.9 TS Viewing Utility (tsview)

E.9.1 Overview

The tsview utility allows users to specify a TS file and obtain printout of header information and time series data on a terminal screen. This is used to view header information to see how the data were collected, or to verify that IQ data were recorded. Since the tsview source code is provided, 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 Section E.5.4 TS Archive Log Area on page 428.

E.9.2 Starting tsview and Sample Session

E.9.2.1 Starting tsview

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.

NOTE Before starting tsview, use the `cd` command to go to the directory where the archive files are located. This saves you from having to specify the full path and to allow the X-window center button and copy/paste technique to be used to avoid typing file names.

1. In the archive directory, use the `ls` command to view the files.
2. Run `tsview <pathname> <-options>`. For example, `$ tsview -help`.

   To see the various options, see Section E.9.3.1 -help on page 441.

E.9.2.2 Sample Session

In this session, you 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)
              ask Name:Ascope_DEFAULT
          Sweep Number:0
                    . . .
```

See Section E.10 TS Record Data Format on page 443 for details of the header format and information content. See Section E.9.3 Tsview Command Line Options on page 441 for the file naming convention.

E.9.3 Tsview Command Line Options

E.9.3.1 -help

Gives a list of available 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)
```

E.9.3.2 pathname and file naming convention

This is the directory and name of the TS file.

NOTE	Before starting tsview, use the `cd` command to go to the directory where the archive files are located. This saves you from having to specify the full path and to allow the X-window center button and copy/paste technique to be used to avoid typing file names.

In the archive directory, use the `ls` command to view the files that you want by date and time using standard UNIX options for `ls`. The asterisk * is a wild card. The file names are very long, so you do not want to type them. Instead, highlight the file name and click the middle button to copy the text to the terminal cursor location.

The file name format is designed to make it easy to identify TS files. In this 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
```

The following is a description of some of the fields:

- `site`—Site name typed into the setup program on the RVP900 that generated the data. Both `site` and `taskname` fields are preprocessed

to remove characters that would mess up the filename, such as unprintable characters, space and `/`.

- `taskname`—Identification of the application configuration 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 (in the example, `Ascope_DEFAULT`). In the case of a custom user application, the user can specify an appropriate ID for the configuration so that it is archived appropriately.
- `sweep`—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 RVP900 operation, there is no concept of a sweep and the sweep number is set to 0. For RHI scanning, the concept of a sweep is the same, except that it is an elevation sweep rather than azimuth sweep.
- `polarization`—This refers to the transmit polarization. There are four choices: H, V, H+V (simultaneous H and V transmit) and ALT (alternating H and V transmit).

E.9.3.3 -count:N

Each file consists of the IQ data for all range bins for each pulse in the sweep. `-count:N` is how many pulses to display. See the example in Section E.9.3.4 -data. The count is set to 1, that is, only the information for one pulse is displayed.

E.9.3.4 -data

Use the `-data` option to see the actual IQ values for each range bin.

This example has the time series values for 400 bins for the 101st pulse:

```
$ 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
```

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.

E.9.3.5 -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 this example, 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)
```

E.9.3.6 -skip:N

Use `-skip:N` to specify how many pulses to skip before starting the terminal listing. In this example, we wanted the 101st pulse, so we specified `skip:100`.

E.9.3.7 -verbose

To see the detailed information contained in the headers. This was enabled in the example session (see Section E.9.2.2 Sample Session).

E.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 RVP900. In RVP900 nomenclature, this is called the "Acquisition Mode" (stored in a structure called the "rvptsPulseInfo"). Each time something is changed, such as the PRF, the acquisition mode changes, and a new file is created. If there are no changes, the files are arbitrarily written every 200000 pulses.

TS files consist of an interesting mixture of ASCII headers and binary data, shown in Table 26 on page 444. They start with the "rvptsPulseInfo" structure. This is followed by possibly many pulses. Each pulse has a rvptsPulseHdr structure followed by an array of 16-bit binary data.

Table 26 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 full magnitude with a value of 1. This represents +8dBm on the IFDR, but may change in future revisions. Floating point numbers are packed into 16-bit words using "High SNR" packed floating format (see Section 7.7 Initiate Processing (PROC) on page 275). These 16-bit words are always stored in the little-endian byte order that is native to the Intel processor chips common on PCs, which is the reverse of "Network order" used on sockets. The tsview displays the (I,Q) samples in power and angle format:

$$Power - 6dBm + 10 \times \log_{10}[I^2 + Q^2]$$

$$Angle - atan2(Q, I)$$

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 it is a dual polarization receiver system, this is duplicated for the second receiver (the iVIQPerBin field in the pulse hdr). The total number of bytes of data is:

$$Bytes - 2 \times 2 \times iNumVecs \times iVIQPerBin$$

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=RVP900	Site name of RVP900
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	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 RVP900
fDBzCalibCx	dBZ0 for second polarization
fNoiseCalib[2]	Noise level at calibration, [2 polarizations]
fBurstCalib	Burst power at calibration
iAntStatusMask	Mask of what antenna status bits are available
fPWidthUSecPulse2	Width of pulse 2
fDBzCalibPulse2[2]	dBZ0 pulse 2 [2 polarizations]
fNoiseCalibPulse2[2]	Noise level at calibration, [2 polarizations]
fBurstCalibPulse2	Burst power at calibration
iFlags	Bit 1 set if Hybrid Pulse recorded
fNoiseDBmPuse2[2]	Current noise level for pulse 2 [2 pols.]
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 ms
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	
iAntStatus	Mask of current antenna status bits
iVecOffsetPulse2	Offset in the TS data to pulse 2
rvptsPulseHdr end	

APPENDIX F

RCP902 WSR98D PANEL

F.1 OVERVIEW

This appendix describes the functionality and architecture of the interface between the RCP902 WSR98D panel and RVP900 system (IFDR). It briefly summarizes the system, placing special emphasis on functionality, including identification of key hardware and software components, as they relate to the interface. If more than one external system is part of the interface being defined, then include additional sections at this level for each additional external system.

F.2 Safety Considerations

F.2.1 ESD Protection

Electrostatic Discharge (ESD) can cause immediate or latent damage to electronic circuits. Vaisala products are adequately protected against ESD for their intended use. However, it is possible to damage the product by delivering electronic discharges when touching, removing, or inserting any objects inside the equipment housing.

To make sure you are not delivering high static voltages yourself:

- Avoid touching exposed connectors unnecessarily.
- Handle ESD sensitive components on a properly grounded and protected ESD workbench.
- When an ESD workbench is not available, ground yourself to the equipment chassis with a wrist strap and a resistive connection cord.

- If you are unable to take either of the above precautions, touch a conductive part of the equipment chassis with your other hand before touching ESD sensitive components.
- Always hold the boards by the edges and avoid touching the component contacts.

F.3 Regulatory Compliances

This equipment complies with the standard listed in Table 27.

Table 27 Regulatory Compliances

Listing	Standard
Electrical Safety	
CE Mark	EN60950-1
EMC	61000:
US EMC	FCC part 15, subpart B, Class A, EN55011
EU: Emissions	IEC6100-6-2 EMC Emissions
EU: Immunity	IEC6100-6-4 EMC Immunity Testing

F.3.1 DC Power Conditions for Use

These conditions must be met by the customer installation for the unit to be considered safe in DC power application:

- DC power as rated on the label or in the manual; 28V ±1V
- Temperatures: -20°C to +55°C (-4°F to 131°F)
- In order to satisfy CE Mark safety requirements, the system integrator must provide a means of removing power to the unit without the use of a tool. Common options to meet the requirement are:
  - Install a quick disconnect box located within view of the panel.
  - Use a power cord with a fuse holder that can be opened without a tool. The panel has a 5 A fuse internally, so an 8 A, 125 V fuse on the cord would not interfere with operation.
- To guarantee immunity and emissions performance, the power cable should be shielded with a minimum shielding of 85% optical coverage. The shield should make 360 degrees/circumferential electrical contact with the metal case of the connectors at each end. If this is not possible, ferrites maybe necessary to block noise from entering the panel to achieve acceptable emissions performance. Power and ground cables should also be twisted and able to carry a maximum of 1.0 A total.

WARNING WSR98D Power Brick provides 2250V of basic isolation against shock. Higher levels of protection will need to be provided at the system level by providing a SELV or equivalent protection at the power input of the panel per the IEC/UL/EN 60950-1 standard. The system integrator is responsible for providing an SELV compliant input to guaranteeing a safe operating environment.

F.3.2 WEEE Compliance

DECLARATION OF CONFORMITY in relation to Directive 2002/96/EC, Waste Electrical and Electronic Equipment (WEEE).

The RVP900 manufactured by Vaisala complies fully with the requirements of Directive 2002/96/EC on the Waste Electrical and Electronic Equipment (WEEE).

F.3.2.1 Recycling

Vaisala has implemented return facilities for all products that we bring to market. All RVP900 components should be returned to the following address for recycling:

Vaisala Inc.
194 South Taylor Ave.
Louisville, CO 80027
(303) 499–1701

RVP900 components should not be disposed of in landfills.

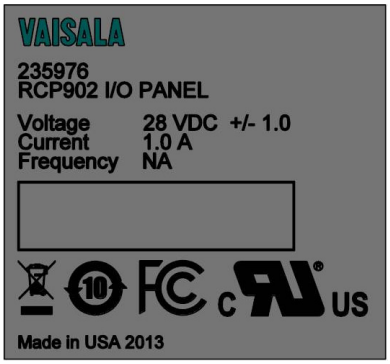
RVP900 components are marked with the end–of–life, not for landfill disposal symbol in accordance with European Standard EN 50419.

F.4 RoHS Compliance

The RoHS Directive 2002/95/EC restricts the use of six hazardous materials found in electrical and electronic products. All applicable products in the EU market after July 1, 2006 must pass RoHS compliance. The maximum permitted concentrations are 0.1% or 1000 ppm (except for cadmium, which is limited to 0.01% or 100 ppm) by weight of homogeneous material.

F.4.1 China RoHS Compliance

The China RoHS Directive requires disclosure (not removal) of the 6 EU RoHS substances for those products included in the "List". Disclosure can be at the component or at the sub assembly level, but it has to be in the prescribed format, in Chinese, as detailed in the document "Marking for the control of Pollution Caused by Electronic Information Products". There are product marking requirements and a calculation of the "Environmentally Friendly Use Period" to be calculated.



F.5 RCP902 WSR98D Panel Architecture

The RVP902 server connects to the RVP900 IFDR, which connects to the RCP902 WSR98D panel. The RCP902 WSR98D panel connects to the radar. An separate external power supply connects to RCP902 WSR98D panel.

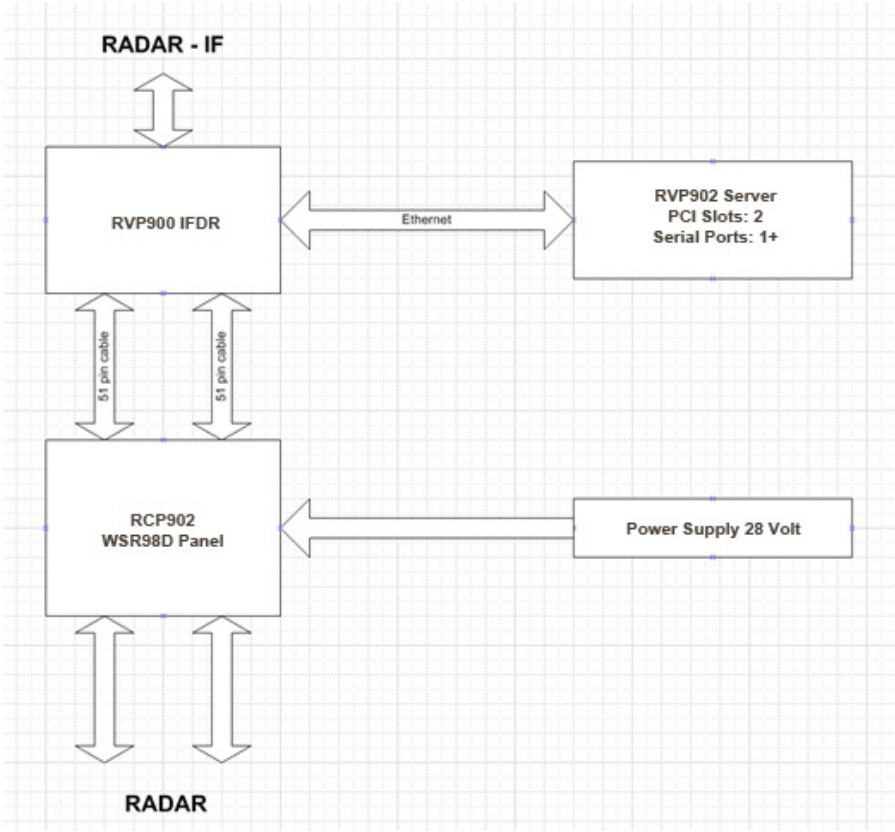


Figure 75 RCP902 Architecture

The RVP902 software allows you to set and configure the RCP902 WSR98S panel (with IRIS8.13.1 or higher) for the specific radar needs.

The RCP902 WSR98D is designed to connect to WSR98D radar interfaces.

F.6 Physical Interface

F.6.1 Overall Size

The overall size of the RVP902 WSR98D panel is equal to or less than the overall size of the RVP8/RCP8 ORDA panel.

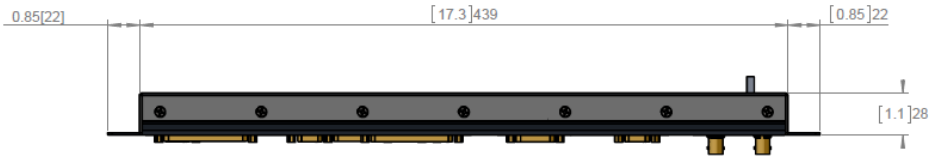


Figure 76 RCP902 WSR98D Top Panel Dimensions

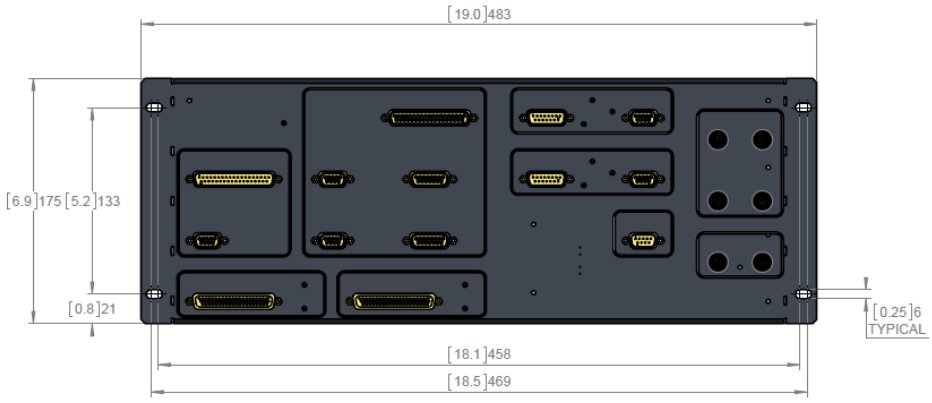


Figure 77 RCP902 WSR98D Back Panel Dimensions

F.6.2 Mounting Dimensions

The mounting dimensions of the RVP902 WSR98D panel are 4U 19 in EIA rack mountable; the same as the RVP8/RCP8 ORDA panel. The RVP902 WSR98D panel is capable of being mounted in the same location an RVP8/RCP8 ORDA panel; without modification to the WSR98D radar.

F.6.3 Connector Locations

All the connectors on the RVP902 WSR98D panel that interface with the WSR98D radar are located in the similar locations as the RVP8/RCP8 ORDA panel.

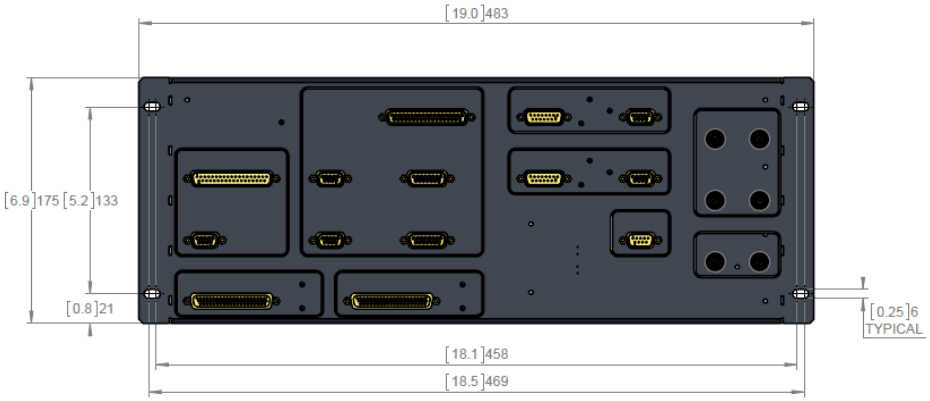


Figure 78 RCP902 WSR98D Connectors (Modify without dimensions)

F.6.4 Modifications on RCP902 WSR98D Panel

The following modifications have been made to the RCP902 WSR98D panel from the RVP8/RCP8 ORDA panel:

- J5, J6, J16, J17, J18-A, J19, J24, and J25 connectors have been removed and their functions obsoleted
- All the connectors on the back side (*-B) have been removed
- J18 connector has been added for external power

F.7 Electrical Interfaces

There are several classes of signals for the RCP902 WSR98D panel. The signal descriptions have a format to identify relational information.

- Signals have pull-up /down depending on initial state and to maintain level during boot conditions
- RS-485 signals:
  - Outputs have a 5 V unidirectional TVS device
  - Inputs have 15 V bidirectional TVS device
  - Inputs have Termination resistors on input to buffer
- TTL signals have a 5 V unidirectional TVS device
- BNC Test Point Outputs:
  - Signal can drive a 50Ω, 5 V signal

- TTL input signals should have a 5 V unidirectional TVS device on them
- TTL output signals should have a 5 V unidirectional TVS device on them

- BNC Analog Inputs:
  - LOG_VIDEO signal (AMUX0) is a 0.5 to 2.5 analog signal with very low frequency/DC characteristics
  - Spare test input (AMUX1) is connected to resistive network and capacitor to allow flexibility in maximum signal. As configured, it will accept ±3V input low frequency signal.

Table 28 RCP902 WSR98D Interface to Radar

Connector Size	Designator	TYPE	Connector Size	Designator	TYPE
D-SUB STR 37 POSITION	J3	PLUG	BNC 50 OHM	J20	JACK
D-SUB STR 9 POSITION	J4	RCPT	BNC 50 OHM	J21	JACK
D-SUB STR 37 POSITION	J7	RCPT	BNC 50 OHM	J22	JACK
D-SUB STR 9 POSITION	J8	RCPT	BNC 50 OHM	J23	JACK
D-SUB STR 15 POSITION	J9	RCPT	BNC 50 OHM	J26	JACK
D-SUB STR 9 POSITION	J10	RCPT	BNC 50 OHM	J27	JACK
D-SUB STR 15 POSITION	J11	RCPT			
D-SUB STR 15 POSITION	J12	PLUG			
D-SUB STR 9 POSITION	J13	RCPT			
D-SUB STR 15 POSITION	J14	PLUG			
D-SUB STR 9 POSITION	J15	RCPT			
D-SUB STR 9 POSITION	J18	PLUG			

Table 29 Signal Entries

Entry	Definition
NC	No connection at pin
GND	Ground
H	Signal is forced to a high state on power up
L	Signal is forced to a low state on power up
_N	Negative signal of an RS-422 signal pair
_P	Positive signal of an RS-422 signal pair

NOTE

- If a signal is an output, the **Initial Condition** column contains the power-on state of that signal. This value may change once RVP900 boots and processes on host computer start.

- All RS-422 signals are capable of driving a 100 ohm or greater terminated line.

- All RS-422 inputs are terminated with a 100 ohm resistor.

- To guarantee immunity and emissions performance, the power cables should be shielded with a minimum shielding of 85% optical coverage. The shield should make 360 degrees/circumferential electrical contact with the metal case of the connectors at each end. For best signal quality, twisted pairs should be used for all differential signals.

F.7.1 J3 - Transmitter Triggers (Tx TRIGS)

Signal Type: DB37P

Signal	Connections	Initial Condition
1	NC	
2	RF_PULSE_START_TRG_N	L
3	RF_DRIVE_TRG_N	L
4	FILAMENT_RESET_TRG_N	L
5	POST_CHARGE_TRG_N	L
6	MOD_CHARGE_TRG_N	L
7	MOD_DISCHARGE_TRG_P	H
8	NC	
9	SHORT_BEAM_PULSE_SEL_P	H
10	SHORT_RF_PULSE_SEL_P	H
11	PULSE_RATE_IN_P0	H
12	PULSE_RATE_IN_P1	H
13	PULSE_RATE_IN_P2	H
14	TRIGGER_CHARGE_TRG_N	L
15	NC	
16	NC	
17	NC	
18	NC	
19	NC	
20	NC	
21	RF_PULSE_START_TRG_P	H
22	RF_DRIVE_TRG_P	H
23	FILAMENT_RESET_TRG_P	H
24	POST_CHARGE_TRG_P	H
25	MOD_CHARGE_TRG_P	H

Signal	Connections	Initial Condition
26	MOD_DISCHARGE_TRG_N	L
27	NC	
28	SHORT_BEAM_PULSE_SEL_N	L
29	SHORT_RF_PULSE_SEL_N	L
30	PULSE_RATE_IN_N0	L
31	PULSE_RATE_IN_N1	L
32	PULSE_RATE_IN_N2	L
33	TRIGGER_CHARGE_TRG_P	H
34	NC	
35	NC	
36	NC	
37	NC	

F.7.2 J4 - Receiver Protector (Rx PROT)

Connector Type: DB09S

Signal	Connections	Initial Condition
1	VCC3_OUT*	
2	RCVR_PROTECT_RSP_P	
3	RCVR_PROTECT_RSP_N	
4	RCVR_PROTECT_CMD_P	H
5	RCVR_PROTECT_CMD_N	L
6	NC	
7	GND	
8	GND	
9	NC	
*	VCC3_OUT is designed to provide 200 mA of current from the 5V supply on the panel.	

F.7.3 J7 - RF Generator (RF-GEN)

Connector Type: DB37S

Description: RF generator control, status, and phase modulation

Signal	Connections	Initial Condition
1	NC	
2	NC	
3	NC	
4	CHANFAIL_N	
5	CHANFAIL_P	
6	PHASEBIT_N4	H
7	PHASEBIT_P4	L

Signal	Connections	Initial Condition
8	PHASEBIT_N3	H
9	PHASEBIT_P3	L
10	PHASEBIT_N2	H
11	PHASEBIT_P2	L
12	PHASEBIT_N1	H
13	PHASEBIT_P1	L
14	OPFREQREFFL_N	
15	OPFREQREFFL_P	
16	COHOREFFAIL_P	
17	NC	
18	NC	
19	RFGATE_P	L
20	NC	
21	NC	
22	GND	
23	GND	
24	PHCOHOSEL_N	H
25	PHCOHOSEL_P	L
26	PHMODFAIL_N	
27	PHMODFAIL_P	
28	PHASEBIT_N7	H
29	PHASEBIT_P7	L
30	PHASEBIT_N6	H
31	PHASEBIT_P6	L
32	PHASEBIT_N5	H
33	PHASEBIT_P5	L
34	COHOREFFAIL	
35	NC	
36	NC	
37	RFGATE_N	H

F.7.4 J8 - RF Test Selection (RF-TEST SEL)

Connector Type: DB09S

Description: Controls the four position diode switch for the RF test signal selection.

Signal	Connections	Initial Condition
1	DRSIGPOS_P3	L
2	DRSIGPOS_N2	H
3	DRSIGPOS_P2	L
4	DRSIGPOS_N1	H

Signal	Connections	Initial Condition
5	DRSIGPOS_P1	L
6	DRSIGPOS_N3	H
7	DRSIGPOS_N4	H
8	DRSIGPOS_P4	L
9	GND	

F.7.5 J9 - Attenuator Control (ATTEN)

Connector Type: DB15S

Description: Controls the 7-bit attenuator.

Signal	Connections	Initial Condition
1	DRIVESIG8DB_N	L
2	DRIVESIG8DB_P	H
3	DRIVESIG4DB_N	L
4	DRIVESIG4DB_P	H
5	DRIVESIG2DB_N	L
6	DRIVESIG2DB_P	H
7	DRIVESIG1DB_N	L
8	DRIVESIG1DB_P	H
9	GND	
10	DRIVESIG40DB_N	L
11	DRIVESIG40DB_P	H
12	DRIVESIG32DB_N	L
13	DRIVESIG32DB_P	H
14	DRIVESIG16DB_N	L
15	DRIVESIG16DB_P	H

F.7.6 J10 - Noise Source (NOISE SRC)

Connector Type: DB09S

Description: Controls the noise source and RF test signal injection point.

Signal	Connections	Initial Condition
1	NC	
2	NOISECTRLIN_N	H
3	NOISECTRLIN_P	L
4	P5ION_N	H
5	P5ION_P	L
6	NC	
7	GND	
8	GND	

Signal	Connections	Initial Condition
9	NC	

F.7.7 J11 - RF Test Switch (RF-TEST SW)

Connector Type: DB15S

Description: Controls the 10-position RF test switch.

Signal	Connections	Initial Condition
1	DRSIGBIT_N4	H
2	DRSIGBIT_P4	L
3	DRSIGBIT_N3	H
4	DRSIGBIT_P3	L
5	DRSIGBIT_N2	H
6	DRSIGBIT_P2	L
7	DRSIGBIT_N1	H
8	DRSIGBIT_P1	L
9	NC	
10	NC	
11	NC	
12	GND	
13	GND	
14	NC	
15	NC	

F.7.8 J12 - DAU Serial I/O (SERIAL-IN)

Connector Type: DB15P

Description: DAU serial line from the 98D.

Signal	Connections	Initial Condition
1	NC	
2	NC	
3	NC	
4	NC	
5	NC	
6	NC	
7	NC	
8	NC	
9	NC	
10	NC	
11	DAU_TX	
12	NC	

Signal	Connections	Initial Condition
13	NC	
14	DAU_RX	
15	GND	

F.7.9 J14 - DCU Serial I/O (SERIAL-IN)

Connector Type: DB15P

Description: DCU serial line from the WSR98D.

Signal	Connections	Initial Condition
1	NC	
2	NC	
3	NC	
4	NC	
5	NC	
6	NC	
7	NC	
8	NC	
9	NC	
10	NC	
11	DCU_TX	
12	NC	
13	NC	
14	DCU_RX	
15	GND	

F.7.10 COAX

I/O	Pins	Signal Name
O	J20.1	RVP9_TP1
GND	J20.2	GND
O	J21.1	RVP9_TP2
GND	J21.2	GND
O	J22.1	RCP_TP1
GND	J22.2	GND
O	J23.1	RCP_TP2
GND	J23.2	GND
I	J26.1	LOGVID_IN_AN
I	J26.2	LOGVID_IN_AP
I	J27.1	SPARE_AMUX_IN_P
I	J27.2	SPARE_AMUX_IN_N

F.7.11 J26 - LOG Video Input (RF TEST-IN)

Connector Type: 50Ω BNC

Description: Baseband detected pulse:

- Analog input range: 0.5 V to 2.5 V, low frequency/DC
- Input Impedance: 90 ohms
- Settling time: Input has a settling time of 7 usec
- Gain: 1 V/V nominal
- Circuit description: Input is treated as a differential input. The circuit has an active input electronics to isolate signal and signal ground from the panel and RVP9IFDR ground noise. If the ground reference drifts from the panel's signal ground, the gain deviates from its nominal value. Use a well-shielded cable if noise enters the signal or ground reference; it influences readings.

F.7.12 J27 - Spare Analog Input (SPARE)

Connector Type: 50 ohm BNC

Description: Spare, maybe use instead of or in addition to:

- Analog input range: +/- 3V, low frequency/DC
- Input Impedance: 100 ohms
- Settling time: Input has a settling time of 500 usec
- Gain: 1 V/V

- Circuit description: Ciruit has resistive network to provide input impedance. Due to its simplicity, it has small temperature sensitivity; results may vary on the order of a 1 mV/C.

F.7.13 J20, J21, J22, J23 - RVP901 Digital Test Points

Connector Type: 50 ohm BNC

Description: Programmable test point outputs from the RVP900 are capable of driving 5 V, 50 ohm load. J20 and J21 are RVP controlled test points showing real-time trigger information. J22 and J23 are RCP controlled test points that are configurable in the softplane file. For reasonable signal integrity, a 50 ohm load should be provided or a connector should be probed directly with an active high-impedance probe.

F.7.14 J18 - Panel Power Input (+28V POWER)

Connector Type: DB09P

Description: Nominal 28 V Input Supply. Expected power to be less than 15 W

NOTE	* NC on J18 pin 3, per customer request, one pin left un-connected.

F.8 RVP900 Interface to the RCP902 WSR98D Panel

The RCP902 WSR98D panel has two connectors, which are defined by connection to the RVP900.

Table 30 RVP900 Signal Types

Type	Description
GPDIFF_PIN_LP/N	RS-422 signals
TTLIO_PIN/GND	TTL signals
AMUX_P0/AMUX_N0	Differential analog input A/D signals
+5V	+5VDC
-5V	-5VDC
GND	Ground

For more information, refer to drawing DRW240510.

F.8.1 RCP902 WSR98D I/O Interconnect Breakout

The interconnect cables breakout each of the 51-pin micro "D" connectors on the IFDR into standard 62-pin female "D" connectors to the RCP902 WSR98D panel. Table 31 and Table 32 provide the cable wiring and internal signal names on the IFDR and WSR98D. The standard cable length is 2 m.

If extending the length of this interface by additional cabling, a cable with good isolation between the twisted signal pairs is required to maintain the signal integrity, guarantee valid logic levels on TTL signals, and low noise on analog signals. The Vaisala provided wire meets the MIL-DTL-22759/11 specification, which is insulated with PTFE. If additional shielding between wire pairs is feasible it should also be considered.

Table 31 RCP902 WSR98D Interconnect Cable for Misc I/O A Connection

IFDR J6 - 51 Pin	IFDR Signal Name	WSR98D J1 - 62 Pin	WSR98D Signal Name
1/19	GPDIFF_PIN_LP/N0	1/23	TRIGGER_CHARGE_TRG_P/N
2/20	GPDIFF_PIN_LP/N1	2/24	MOD_DISCHARGE_TRG_P/N
3/21	GPDIFF_PIN_LP/N2	3/25	MOD_CHARGE_TRG_P/N
4/22	GPDIFF_PIN_LP/N3	4/26	POST_CHARGE_TRG_P/N
5/23	GPDIFF_PIN_LP/N4	5/27	FILAMENT_RESET_TRG_P/N
6/24	GPDIFF_PIN_LP/N5	6/28	RF_DRIVE_TRG_P/N
7/25	GPDIFF_PIN_LP/N6	7/29	RF_PULSE_START_TRG_P/N
8/26	GPDIFF_PIN_LP/N7	8/30	RCVR_PROTECT_CMD_P/N
9/27	GPDIFF_PIN_LP/N8	9/31	RCVR_PROTECT_RSP_P/N
10/28	GPDIFF_PIN_LP/N9	10/32	RVP_UP_RSP_P/N
11/29	TTLIO_PIN0/GND	11/33	CONN_LED_LO
12/30	TTLIO_PIN1/GND	12/34	SPARE_TTL11
13/31	TTLIO_PIN2/GND	13/35	SPARE_TTL12
14/32	TTLIO_PIN3/GND	14/36	SPARE_TTL13
15/33	TTLIO_PIN4/GND	15/37	SHORT_RF_PULSE_SEL
16/34	TTLIO_PIN5/GND	16/38	SHORT_BEAM_PULSE_SEL
17/35	TTLIO_PIN6/GND	17/39	RVP_TP1
18	GND	22	
36/37	TTLIO_PIN7/GND	18/40	RVP_TP2
38/39	TTLIO_PIN8/GND	19/41	NC
40/41	TTLIO_PIN9/GND	20/42	NC
42/43	AMUX_P0/AMUX_N0	54/55	AMUX_5V
44/45	V_5P0/GND +5V	47/48	J1 OK LED

Table 31 RCP902 WSR98D Interconnect Cable for Misc I/O A Connection

IFDR J6 - 51 Pin	IFDR Signal Name	WSR98D J1 - 62 Pin	WSR98D Signal Name
46/47	AMUX_P1/AMUX_N1	57/58	SPARE_AMUX_P/N
48/49	V_N5P0/GND -5V	49/50	NC
50/21	AMUX_P2/AMUX_N2	60/61	NC

Table 32 RCP902 WSR98D Interconnect Cable for Misc I/O B Connection

IFDR J3 - 51 Pin	IFDR Signal Name	WSR98D J2 - 62 Pin	WSR98D Signal Name
1/19	GPDIFF_PIN_LP/N10	1/23	PHCOHOSEL_P/N
2/20	GPDIFF_PIN_LP/N11	2/24	RFGATE_P/N
3/21	GPDIFF_PIN_LP/N12	3/25	RVP_UP_CLK_P/N
4/22	GPDIFF_PIN_LP/N13	4/26	RVP_UP_DATA_P/N
5/23	GPDIFF_PIN_LP/N14	5/27	RVP_UP_TIME1_P/N
6/24	GPDIFF_PIN_LP/N15	6/28	RVP_UP_TIME2_P/N
7/25	GPDIFF_PIN_LP/N16	7/29	RVP_UP_SEL_P/N
8/26	GPDIFF_PIN_LP/N17	8/30	RVP_UP_SPARE_P/N
9/27	GPDIFF_PIN_LP/N18	9/31	RCP_TP1_P/N
10/28	GPDIFF_PIN_LP/N19	10/32	RCP_TP2_P/N
11/29	TTLIO_PIN0/GND	11/33	CHANFAIL
12/30	TTLIO_PIN1/GND	12/34	OPFREQREFFL
13/31	TTLIO_PIN2/GND	13/35	COHOREFFAIL
14/32	TTLIO_PIN3/GND	14/36	PHMODFAIL
15/33	TTLIO_PIN4/GND	15/37	DRSIGBIT1
16/34	TTLIO_PIN5/GND	16/38	DRSIGBIT2
17/35	TTLIO_PIN6/GND	17/39	DRSIGBIT3
18	GND	22	
36/37	TTLIO_PIN7/GND	18/40	DRSIGBIT4
38/39	TTLIO_PIN8/GND	19/41	CONN_LED_LO
40/41	TTLIO_PIN9/GND	20/42	NC
42/43	AMUX_P0/AMUX_N0	54/55	LOG_VIDEO_P/N
44/45	V_5P0/GND +5V	47/48	J2 OK LED
46/47	AMUX_P1/AMUX_N1	57/58	NC
48/49	V_N5P0/GND -5V	49/50	NC
50/21	AMUX_P2/AMUX_N2	60/61	NC

F.9 Software Control/Status

Many of the signals on the WSR98D panel are driven real-time by the RVP900 IFDR controlled by the RVP900 process or on a sampled basis by either the RVP900 or RCP8 processes. If controlled by the RCP8 process, they are mapped to logical variables in the softplane.conf file.

F.9.1 Logical Variables

For logic mappings to be valid, the settings in Table 33 must be defined in the softplane.conf file.

Table 33 Software Control/Status Variable

Signal Name	Controlling Process	Programmable Logic Variable	Conditions
		J3	
NC			
RF_PULSE_START_TRG_P/N	RVP	trigger 2	
RF_DRIVE_TRG_P/N	RVP	trigger 6	
FILAMENT_RESET_TRG_P/N	RVP	trigger 5	
POST_CHARGE_TRG_P/N	RVP	trigger 7	
MOD_CHARGE_TRG_P/N	RVP	trigger 4	
MOD_DISCHARGE_TRG_P/N	RVP	trigger 3	
SHORT_BEAM_PULSE_SEL_P/N	RCP	cAux[27]	
SHORT_RF_PULSE_SEL_P/N	RCP	cAux[28]	
PULSE_RATE_IN_P/N0	RVP		Latched in on falling edge of trigger 10
PULSE_RATE_IN_P/N1	RVP		Latched in on falling edge of trigger 10
PULSE_RATE_IN_P/N2	RVP		Latched in on falling edge of trigger 10
TRIGGER_CHARGE_TRG_P/N	RCP	trigger 8	
		J4	
RCVR_PROTECT_RSP_P	RCP	sAux[30]	
RCVR_PROTECT_CMD_P	RVP / RCP	trigger 9	cAux[24] forces RX Protect Mode
		J7	
PHASEBIT_P/N7	RVP		Latched in on rising edge of trigger 10
PHASEBIT_P/N6	RVP		Latched in on rising edge of trigger 10
PHASEBIT_P/N5	RVP		Latched in on rising edge of trigger 10
PHASEBIT_P/N4	RVP		Latched in on rising edge of trigger 10

Table 33 Software Control/Status Variable

Signal Name	Controlling Process	Programmable Logic Variable	Conditions
PHASEBIT_P/N3	RVP		Latched in on rising edge of trigger 10
PHASEBIT_P/N2	RVP		Latched in on rising edge of trigger 10
PHASEBIT_P/N1	RVP		Latched in on rising edge of trigger 10
CHANFAIL_P/N	RCP	sAux[61]	
OPFREQREFFL_P/N	RCP	sAux[60]	
COHOREFFAIL_P/N	RCP	sAux[58]	
PHMODFAIL_P/N	RCP	sAux[63]	
PHCOHOSEL_P/N	RCP/RVP	cAux[49], trigger 11	cntPhaseForce = "true", "false"
RFGATE_P/N	RVP	trigger 1	
		J8	
DRSIGPOS_P/N4	RCP	cAux[42]	
DRSIGPOS_P/N3	RCP	cAux[43]	
DRSIGPOS_P/N2	RCP	cAux[44]	
DRSIGPOS_P/N1	RCP	cAux[45]	
		J9	
DRIVESIG40DB_P/N	RCP	cAux[31]	
DRIVESIG32DB_P/N	RCP	cAux[32]	
DRIVESIG16DB_P/N	RCP	cAux[33]	
DRIVESIG8DB_P/N	RCP	cAux[34]	
DRIVESIG4DB_P/N	RCP	cAux[35]	
DRIVESIG2DB_P/N	RCP	cAux[36]	
DRIVESIG1DB_P/N	RCP	cAux[37]	
		J10	
NOISECTRLIN_P/N	RCP	cAux[47]	
P5ION_P/N	RCP	cAux[46]	
		J11	
DRSIGBIT_P4	RCP	cAux[38]	
DRSIGBIT_P3	RCP	cAux[39]	
DRSIGBIT_P2	RCP	cAux[40]	
DRSIGBIT_P1	RCP	cAux[41]	
		J20	
RVP9_TP1	RVP		dspx debug options
		J21	

Table 33 Software Control/Status Variable

Signal Name	Controlling Process	Programmable Logic Variable	Conditions
RVP9_TP2	RVP		dspx debug options
J22			
RCP9_TP1	RCP	rcpTP1Value	rcpTP1Enabled = "true"
J23			
RCP9_TP2	RCP	rcpTP2Value	rcpTP2Enabled = "true"
LED			
GO	RVP / RCP	RCP ~cAux[64]	Both RVP and RCP process must be running, toggling their GO bits for this LED to blink
DCU OK	RCP	~cAux[66]	
DAU OK	RCP	~cAux[65]	
INT POWER			When on 5V power on panel has reached acceptable levels.
J1 OK			When on 5V IFDR signal pin is mated on J1
J2 OK			When on 5V IFDR signal pin is mated on J2
CONN OK			When on IFDR is in WSR98D mode and J1 and J2 cables are installed in the right order
IFDR			
Radiate Command	RCP	cAux[29]	
Rf Test Command	RCP	cAux[26]	
CW Test Command	RCP	cAux[25]	
Radiate Status	RCP	sAux[29]	
Rf Test Status	RCP	sAux[26]	
CW Test Status	RCP	cAux[25]	
Rx Protect Status	RCP	cAux[30]	

F.9.2 Monitoring Analog Inputs

There are hooks in the WSR98D Set IO RPC to enable sampling of the analog inputs. There were no customer requirements on how to integrate the sampling in the RVP900/RCP8 processes, but a sample code is provided on how to enable and control the sampling process and retrieve the averaged voltage values. One of three inputs can be selected at a time.

A user-defined RVP900 trigger enables/disables the triggering. When the trigger is high, samples are taken at a user-defined sampling rate, which is defined in usec. The voltage is averaged over a user-defined number of triggers. At the completion of the specified number of triggers, an average value is calculated. The value is held until the user reads the value with the WSR98D Get IO RPC, at which point a new acquisition is started, if the number of samples is not set to zero. To monitor the Log Video Input, the channel number should be set to 0, to monitor the Spare input, it should be set to 4, and the 5V supply should be set to 3.

APPENDIX G

RVP900 SPECIFICATION FOR ASR9-WSP WITH RCP903 ASR9-WSP PANEL

G.1 OVERVIEW

This appendix describes the functionality and architecture of the ASR9 WSP, weather signal processor, using the RVP900 hardware and software implemented using the RVP902-WSP Processor, RCP903 ASR9 Panel, and RVP901-WSP signal processor. It briefly summarizes the system, placing special emphasis on functionality, including identification of key hardware and software components, as they relate to the interface.

The ASR9 WSP functionality is designed to provide a direct replacement for the original RxNet7 / RVP7 implementation carrying forward form, fit, and function of the original system wherever physically and logically possible.

G.2 Safety Considerations

G.2.1 ESD Protection

Electrostatic Discharge (ESD) can cause immediate or latent damage to electronic circuits. Vaisala products are adequately protected against ESD for their intended use. However, it is possible to damage the product by delivering electronic discharges when touching, removing, or inserting any objects inside the equipment housing.

To make sure you are not delivering high static voltages yourself:

- Avoid touching exposed connectors unnecessarily.
- Handle ESD sensitive components on a properly grounded and protected ESD workbench.
- When an ESD workbench is not available, ground yourself to the equipment chassis with a wrist strap and a resistive connection cord.
- If you are unable to take either of the above precautions, touch a conductive part of the equipment chassis with your other hand before touching ESD sensitive components.
- Always hold the boards by the edges and avoid touching the component contacts.

G.3 Regulatory Compliances

This equipment complies with the standard listed in Table 34.

Table 34 Regulatory Compliances

Listing	Standard
Electrical Safety	
CE Mark	EN60950-1
EMC	EMC 61000-3-2 and 3-3, Flicker and Harmonics
US EMC	EN55011 Radiated and Conducted Emissions for ISM products
EU: Emissions	EN61000-6-2-2005 Electromagnetic compatibility for industrial environments, including the following subcategories: • EN61000-4-2, ESD • EN61000-4-3, Radiated Immunity • EN61000-4-4, EFT / Burst • EN61000-4-5, Surge • EN61000-4-6, Conducted Susceptibility • EN61000-4-11, Dips / Dropouts

G.3.1 Power Conditions for Use

These operating conditions must be met by the customer installation for the unit to be considered safe in DC power application:

- RVP900-WSP Subsystem
  - Components must be installed in a suitable fire enclosure to meet EN60950-1 requirements

- Maximum Elevation: 2000 meters
- AC Input: 120V AC, 60 Hz
- Ferrites provided for the WSP and COM interface cables must be installed to guarantee compliance with EMC standards listed.
- Chassis ground connections between each component and building ground must be installed to meet EN60950 requirements.

- RVP901-WSP (IF Digital Receiver)
  - Operating Temperature: -40C to 45C with fans installed and AC supply mounted on enclosure. If fans are removed, customer is responsible for validating operational temperature range under these conditions.
  - Maximum Wattage: 50W
  - Red LED Indicator on D7 indicates normal operation when illuminated. When blinking shows unit is operational but not running data. When ON solid indicates operational and running data. LED behavior backward compatible with RVP7 behavior.
- RCP902-WSP (Processor Computer)
  - Operating Temperature: 10C to 35C
  - High Leakage Current: Due to the redundant power supplies the unit poses a high leakage current.To meet EN60950-1 provided yellow/green chassis ground bonding wire must be installed between chassis of RCP902-WSP and RCP903 Shelf Assembly and building ground.
- RVP903 (Panel and Shelf Assembly)
  - Operating Temperature: -20C to 55C . Unit has shown to be functionally operation down to -20C. However RS-422 receivers are only rated to 0 C, part was selected to maintain full compatibility with original design.
  - Maximum Wattage: 24W
  - Red LED Indicator on D2 indicates normal operation when illuminated. When blinking shows unit is operational but not running data. When ON solid indicates operational and running data. LED behavior backward compatible with RVP7 behavior.

G.3.2 WEEE Compliance

DECLARATION OF CONFORMITY in relation to Directive 2002/96/EC, Waste Electrical and Electronic Equipment (WEEE).

The RVP900-WSP manufactured by Vaisala complies fully with the requirements of Directive 2002/96/EC on the Waste Electrical and Electronic Equipment (WEEE).

G.3.2.1 Recycling

Vaisala has implemented return facilities for all products that we bring to market. All RVP900-WSP components should be returned to the following address for recycling:

Vaisala Inc.
194 South Taylor Ave.
Louisville, CO 80027
(303) 499–1701

RVP900-WSP components should not be disposed of in landfills.

RVP900-WSP components are marked with the end–of–life, not for landfill disposal symbol in accordance with European Standard EN 50419.

G.4 RoHS Compliance

The RoHS Directive 2002/95/EC restricts the use of six hazardous materials found in electrical and electronic products. All applicable products in the EU market after July 1, 2006 must pass RoHS compliance. The maximum permitted concentrations are 0.1% or 1000 ppm (except for cadmium, which is limited to 0.01% or 100 ppm) by weight of homogeneous material.

G.4.1 China RoHS Compliance

The China RoHS Directive requires disclosure (not removal) of the 6 EU RoHS substances for those products included in the “List”. Disclosure can be at the component or at the sub assembly level, but it has to be in the prescribed format, in Chinese, as detailed in the document “Marking for the control of Pollution Caused by Electronic Information Products”. There are product marking requirements and a calculation of the “Environmentally Friendly Use Period” to be calculated.

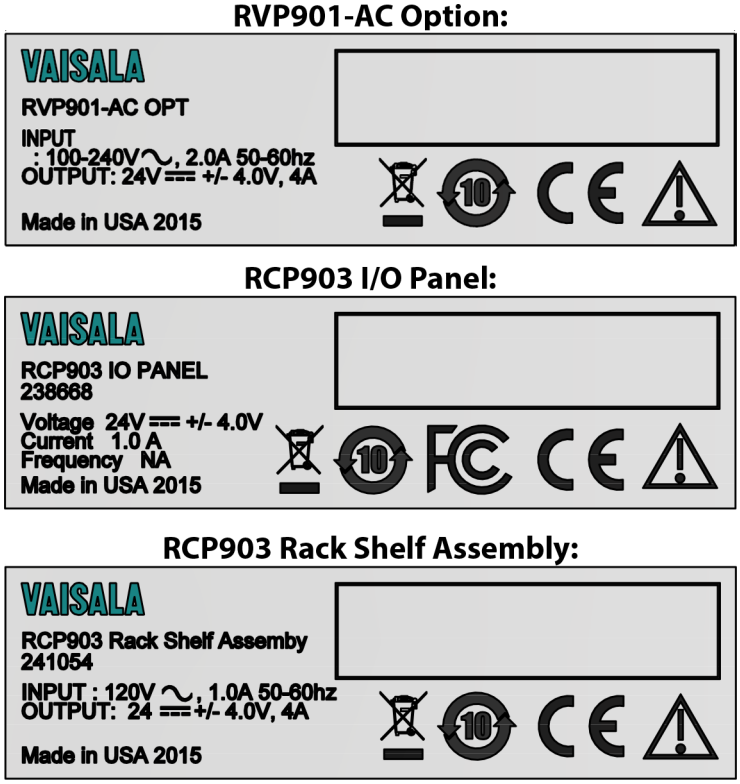


Figure 79 RVP901-AC and RCP903 Product Markings

G.5 ASR9 WSP with RVP900 Panel Architecture

The original ASR9 WSP solution was implemented with the Sigmet RxNet7 Model ASR9/RIM-1 hardware and an RVP7 signal processor. The original design used an embedded general purpose computer built by Ampro Corporation that utilized an ISA bus to connect to the ASR9 specialized radar interface module (RIM) hardware. The hardware was implemented in a set of programmable logic devices (PLDs) and 4 MB of general purpose RAM. The following figure shows a block diagram of the original system.

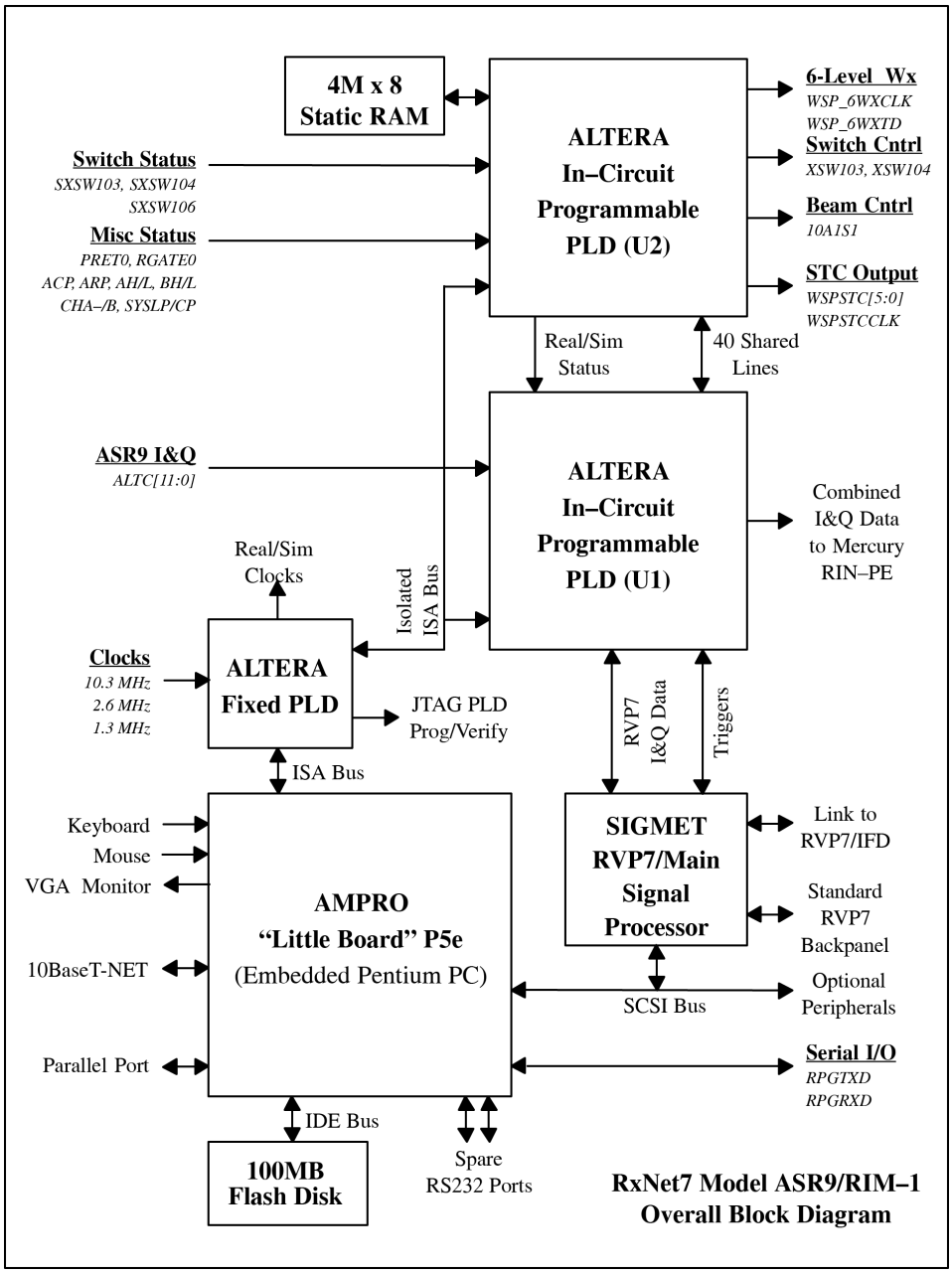


Figure 80 ASR9 WSP with RVP7 Architecture

The RVP7/main signal processor is replaced by the RVP901-WSP IF Digital Receiver. The RVP901-WSP meets or exceeds the specifications of the RVP7 and is designed to mount in a similar fashion as the RVP7, although its footprint is slightly larger. The RVP7 fiber optic link carrying the RVP I and Q interface to the RxNet7 is replaced by an Ethernet interface to the RVP902-WSP Processor. The RVP902-WSP Processor also replaces the Ampro embedded computer to provide the same RIM API control and status functions as the original implementation.

The ASR9 WSP connections on the original RxNet7 front panel have been preserved and implemented on the RCP903 ASR9 Panel on J1 and J2 and connects to the existing WSP interface hardware in the ASR9 radar. The panel also preserved the serial interfaces provided by the RxNet7 on J3 and J4. The RIN-PE connection to the Mercury computer have been obsoleted and the I/Q data path for the ASR9 will now pass to the RVP902-WSP Processor over Ethernet on J5. The required trigger and header information is provided to the RVP901-WSP using its general purpose I/O interface from J6 on the RCP903 ASR9 Panel. This allows the RVP902-WSP Processor to synchronize the ASR9 and RVP900 data streams.

In addition to the RCP903, RVP901-WSP, and RVP902-WSP hardware, an Ethernet switch is provided to allow connections from the RVP901-WSP and RCP903 ASR9 Panel to communicate with ETH1 on the RVP902-WSP Processor, a AC/DC power supply to power the RCP903 ASR9 Panel and an AC power strip to power the RCP903 supply and RVP902-WSP Processor are provided. All of the components in Bay 3 are designed to fit in the same area as the original RxNet7 solution. The following figure shows the block diagram for the new RVP900 base ASR9 WSP solution and a table with the RVP900-WSP components is shown in Table 35.

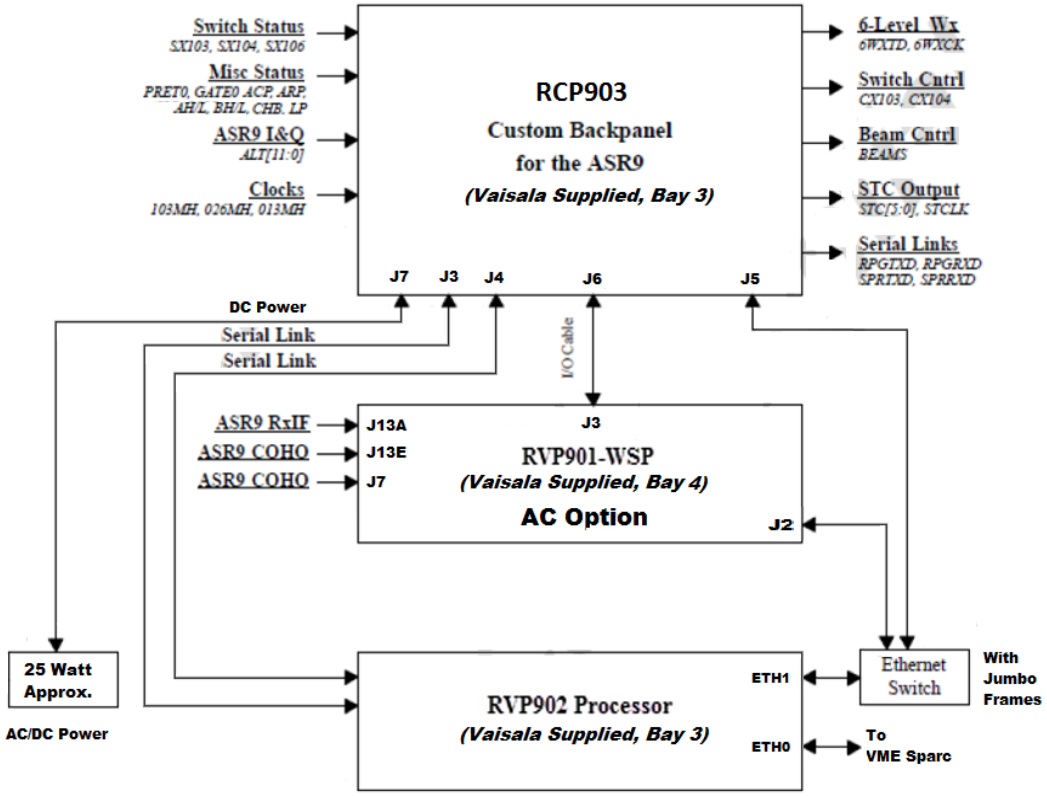


Figure 81 ASR9 WSP with RVP900 Architecture

Table 35 RVP900-WSP Components

Item Code	Description
RVP901-WSP	IF Digital Receiver configured for ASR9-WSP
RVP902-WSP	Processor Computer, RVP900 Signal Processor
RCP903	Panel Assembly for ASR9-WSP
RDA900	Software, RVP900
RVP905	Manual Set

G.5.1 RVP901-WSP Signal Processor

The RVP901-WSP signal processor is Vaisala's standard product offering. It is configured the following way to allow usage in the ASR9 WSP application:

- 5 input channels with up to 100 MHz 16-bit sampling
  - 30 MHz filter on J13A / ADC-A port targeted at a 31.07 MHz IF Frequency
  - The COHO is filtered with a 30 MHz IF filter and split to drive the ADC-E and CLK-IN port as in the original RVP7 implementation.

- General Purpose IO
  - GPIO Differential signal 0 is configured as the Pre-trigger
  - GPIO Differential signals 2, 4 and 6 are configured as inputs and used as a serial data interface to allow the RCP903 panel to provide header information
- I/Q data is output over Ethernet using UDP with command and status functions set using TCP/IP.
  - I / Q data is sent over Ethernet processed and past to the Time Series Interface. Header information is used as a tag to synchronize the RVP900 I/Q data with the ASR9 data stream.
- AC Power option
- 24V Fan

G.5.2 RVP902-WSP Processor

The RVP902-WSP Processor is Vaisala's standard product offering with a Xeon X8DTU motherboard running CentOS 6.4. The RVP900 Radar utilities are equivalent to similar utilities in the RVP7 / RxNet7 platform. The computers key features are.

- Dual Xeon E5620 Intel 2.4GHz Quad-Core Processor with 1333 MHz front side bus
- 2U 19 Rack Mounted Chassis
- WDC 2TB 7.2K SATA 6Gbps 3.5" HDD
- 16 GB Memory
- Dual 500 GB 7.2 RPM SATA Hard Disk
- 4x External 2.0 USB: 2 in the front, 2 in the back
- PS/2 keyboard and mouse
- 2 serial interfaces (1 front, 1 back)
- 2 Ethernet (10/100/1000)
- 700 W Redundant Power Supplies
- VGA Port
- 4x PCI Expansion Slots: 1x PCIe 2.0 x 8 and 3x PCI-X
- DVD-RW
- 0,1,5,10 Software RAID
- RoHs
- 10 to 35°C Operating Temperature

G.5.3 RCP903 ASR9-WSP Custom Panel

The RCP903 is designed to be form, fit, and function compatible with the original RxNet7 implementation. Carrying forward the same physical interfaces and logical functionality as the original implementation. Its main features are as follows:

- CPLD and FLASH memory to configure the panel and load the application
- Altera Cyclone IV FPGA and 16 MB of SRAM to replace the RxNet7 PLDs and SRAM
- 100/1000 Ethernet Capability running TCP/IP for control and status and UDP packets for I/Q data
- WSP#1 and #2 connectors with backward compatible pin out
- Serial Port #1 and #2 with backward compatible pass through functionality
- Pre-trigger and tagging interface to RVP901-WSP
- 24V DC Supply Input (1A max current)
- 8 Indicator LEDs
    - Backward compatible 2x 6-Level Weather Indicators and 4x Test LEDs
    - 2x RCP903 Status Indicators

G.6 RCP903 ASR9-WSP Panel Physical Interfaces

G.6.1 Overall Size

The overall size of the RCP903 ASR9-WSP Panel is equal to or less than the overall size of the RVP7 implementation.

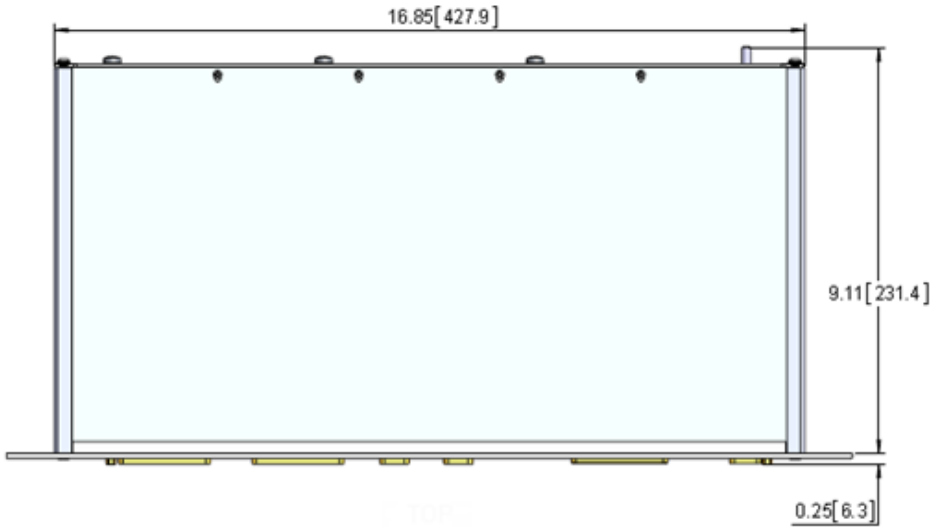


Figure 82 Top Panel Dimensions

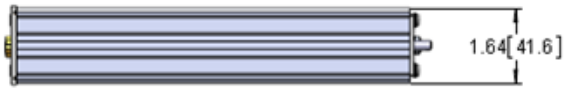


Figure 83 Side Panel Dimensions

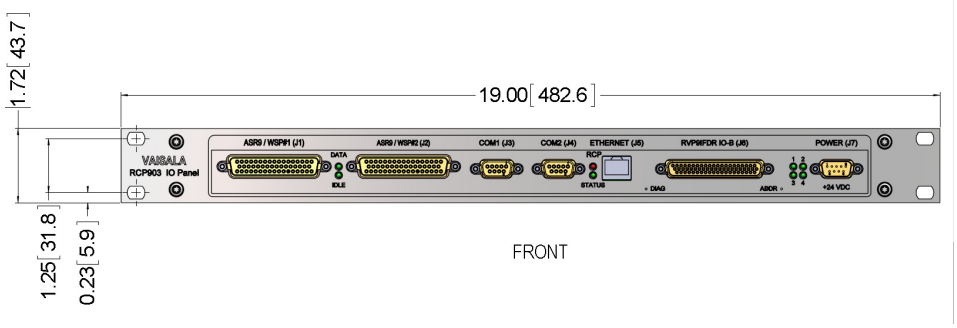


Figure 84 Front Panel Dimensions

Figure 85 Back Panel Dimensions

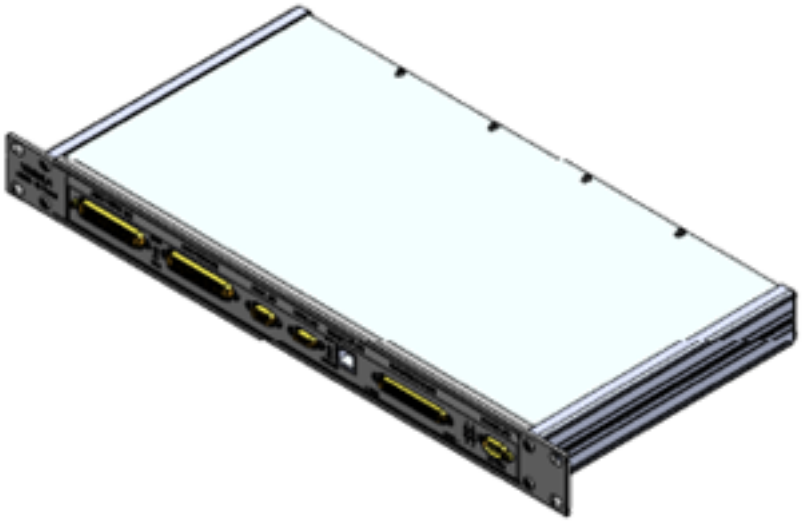


Figure 86 Panel Perspective View

G.6.2 Mounting Dimensions

The mounting dimensions of the RCP903 ASR9-WSP Panel alone are 1U - 19 in EIA rack mountable. The RCP903 solution include a 1U shelf mounted directly behind the panel. The rear mounting shelf includes a connector strip with a switch, a power supply for the panel, an Ethernet switch, and the associated cabling. The RCP903 ASR9-WSP Panel is capable of being mounted in the same location as the RVP7 solution in the ASR9 cabinets, without significant modification to the ASR9.

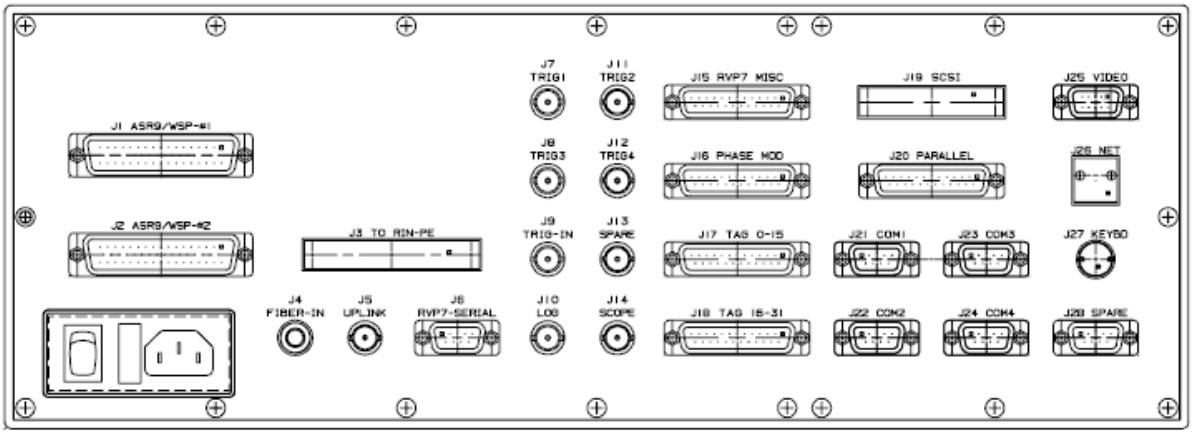


Figure 87 RxNet7 Front Panel

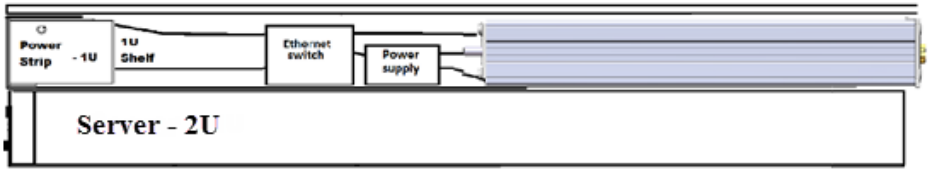


Figure 88 Rack Side Perspective View

G.6.3 Connector Locations

All the connectors on the RCP903 ASR9-WSP Panel that interface with the radar are located on the front of the mounting position.

NOTE	This is intended to mount from the rear of the rack.

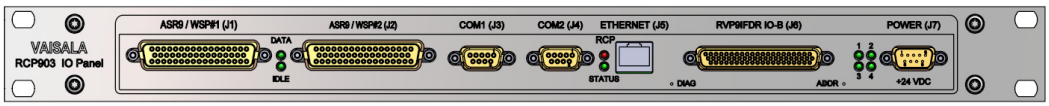


Figure 89 Panel Connector Locations

G.6.4 RCP903 Shelf

RCP903 Shelf is mounted directly behind the panel. The rear mounting shelf includes:

- Connector strip with a switch (Hammond, 1583H6B1BK POWER STRIP)
- Power supply for the panel Integrated power Designs, (REL-110-1006-WT-CHCO, 115 VAC IN, 24VDC)
- Ethernet switch (MOXA, EDS-G205-T or EDS-G205-1GTXSFP-T)
- CBL210355 Cable DC, RCP903 Shelf, DC Supply RCP903

G.7 Electrical Interfaces

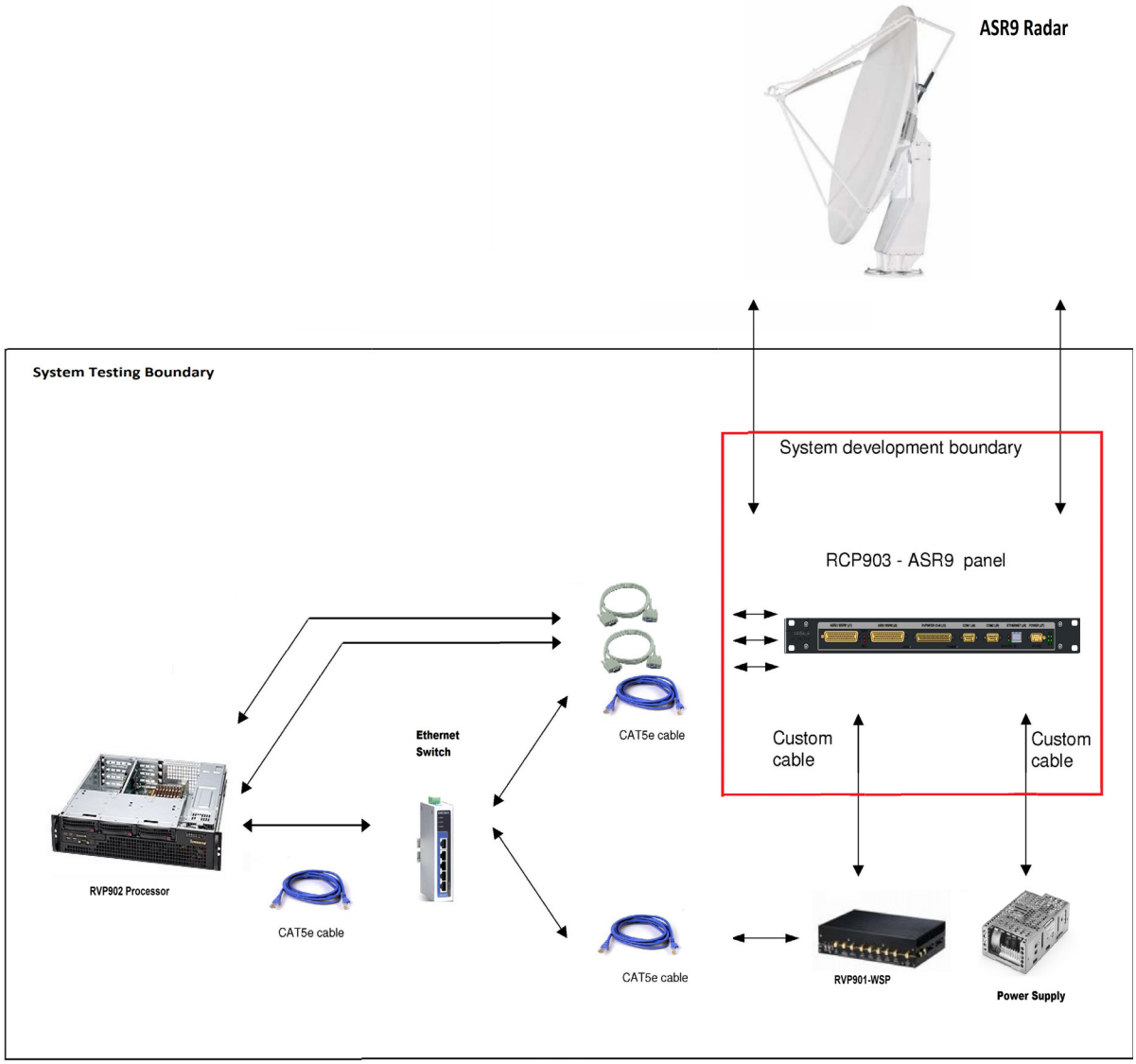


Figure 90 System Testing Boundary

G.7.1 Interconnect Cabling

There are several classes of signals for the RCP903 ASR9-WSP Panel. The signal descriptions have a format to identify relational information.

- The RVP902-WSP Processor attaches to the Ethernet switch using:
  - 2 - 242525 DB9 M to DB9 F Null Modem Reverser Cable and
  - 1 - 220525 Ethernet Cable (Ethernet CAT5E).
- The RCP903-ASR9 WSP Interface Panel attaches to the Ethernet switch using:

  - 1 - 220525 Ethernet Cable (Ethernet CAT5E).
- The RVP901-WSP attaches to the Ethernet switch using:
  - 1 - 231167 Ethernet Cable (Ethernet CAT5E).
- The RVP901-WSP attaches to the Interface Panel using:
  - 1 - Vaisala Cable CBL210393 for Data I/O interface.
- The External 24 volt power supply attaches to the RCP903 ASR9 Interface panel using:
  - 1 - CBL210355, a DC power cable with a 9 pin DB connector.

CBL210393 is an I/O interconnect cable with a 51-pin micro "D" connector for connecting to the RVP901-WSP. This interconnect cable will have a standard 62-pin female "D" connector to attach to the Interface Panel. The standard cable length is 3 m. If the length of the CBL210393 cable needs to be extended, a cable with good isolation between the twisted signal pairs to maintain signal integrity, guarantee valid logic levels on TTL signals, and low noise analog signal, will be provided by Vaisala. The current cable meets the MIL-DTL-22759/11 specification, which is insulated with PTFE.

CBL210355 is a DC power cable with a 9 pin "D" connector for connecting to the Interface panel and an Unmanaged Ethernet Switch. If the length of the CBL210355 cable needs to be extended, a cable with good isolation between the twisted power pairs will be provided by Vaisala.

220525 Ethernet Cable is a CAT5E cable to connect to the Panel via a network switch or directly from the RVP902-WSP Processor.

231167 Ethernet Cable is a CAT5E cable to connect to the RVP901-WSP.

G.7.2 RVP901-WSP

Table 36 RVP901-WSP Interface to ASR9 Radar

Signal	RVP901-WSP Connector
COHO	J7
COHO	J13E
IF	J13A (31.07 MHz)

The RVP901-WSP is configured with the AC Option and with 2 - 30 MHz Filters specified.

Table 37 RVP901-WSP SMA Connector Summary

J-ID	Label	Type	Description	Use
J7	CLK IN	SMA	Reference Clock Input	Y
J8	TRIG-B	SMA	General purpose trigger I/O	N

Table 37 RVP901-WSP SMA Connector Summary

J-ID	Label	Type	Description	Use
J9	CLK OUT	SMA	Reference Clock Output	N
J10	DDS	SMA	Direct Digital Synthesizer output	N
J11	TxDAC-B	SMA	Direct Transmit IF output	N
J12	TxDAC-A	SMA	Same as J11	N
J13A	ADC-A	SMA	Direct IF Input. IF-1 for single channel	Y
J13B	ADC-B	SMA	Direct IF Input. IF-2 for 2nd pol.	N
J13C	ADC-C	SMA	Direct IF Input	N
J13D	ADC-D	SMA	Direct IF Input	N
J13E	ADC-E	SMA	Direct IF Input. Burst Sample Input	Y
J14	VIDEO OUT	SMA	Video DAC output	N
J15	TRIG-A	SMA	General purpose trigger I/O or DAFC intf	N

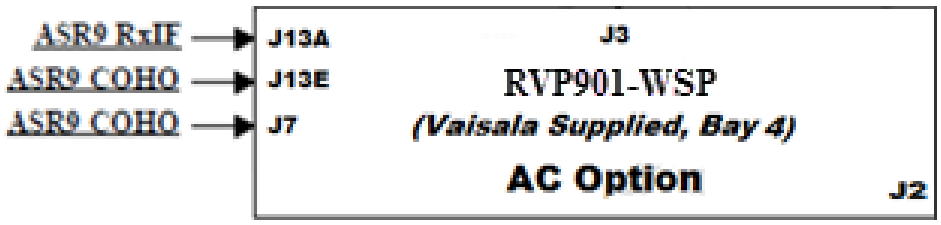


Figure 91 Vaisala Supplied, Bay 4

G.7.3 RCP903 ASR9-WSP Panel Interfaces

Table 38 RCP903 ASR9-WSP Panel Connectors Defined for Customer's Systems Implementation

Connector Size	Designator	Type	Destination
D-SUB STR 50 POSITION	J1	RCPT	ASR9 / WSP#1
D-SUB STR 50 POSITION	J2	RCPT	ASR9 / WSP#2
D-SUB STR 9 POSITION	J3	RCPT	COM1
D-SUB STR 9 POSITION	J4	RCPT	COM2
RJ45	J5	-	ETHERNET
D-SUB STR 62 POSITION	J6	RCPT	RVP901-WSP IO-A
D-SUB STR 9 POSITION	J7	PLUG	POWER (+24V)

G.7.4 ASR9-WSP Panel Indicators and Switches

The RCP903 Panel has several ASR9-WSP specific indicators as well as some general status indicator and control switches. Table 39 summarizes their functionality.

Table 39 ASR9-WSP Panel Indicators and Switches

Label	Location	Type	Description
DATA	Between J1 and J2	Green LED	When enabled, blinks when data is being transmitted on the 6-Level Weather Interface
IDLE	Between J1 and J2	Green LED	When enabled, blinks when idle patterns are being transmitted on the 6-Level Weather Interface
RCP	Between J4 and J5	Red LED	RCP and Status signals blink slowly in unison when boot image is running. When application image is running, status indicator is solid green and RCP indicator is blinking red.
STATUS	Between J4 and J5	Green LED	
1, 2, 3, 4	Between J6 and J7	Green LED	Programmable to display ASR9-WSP test or debug status.
DIAG	Between J5 and J6	Momentary Push Button	When pressed will force the panel into boot mode.
ADDR	Between J6 and J7	Momentary Push Button	When pressed will force the IP address of the panel to its default value, 10.0.1.253

G.7.5 J1, ASR9 Interface WSP #1

Table 40 Pin-out for J1 ASR9 / WSP #1

Pin Number	Ribbon Cable Number	Signal Name	Type	Direction	Description
1	1	WP_PRG_RXD	RS-232	Input	Primary Serial link to ASR9
34	2	WP_RPG_TXD	RS-232	Output	Primary Serial link to ASR9
18,2	3,4	WP_6WX_TD_P/N	RS-422	Output	6-level WX Sync Data
35,19	5,6	WP_CX103_P/N	RS-422	Output	Switch #103 control
3,36	7,8	WP_CX104_P/N	RS-422	Output	Switch #104 control
20,4	9,10	NC_WP_SPARE0_P/N	Spare	No Connect	Spare
37,21	11,12	NC_WP_SPARE1_P/N	Spare	No Connect	Spare
5,38	13,14	WP_SX103_P/N	RS-422	Input	Switch #103 Sense
22,6	15,16	WP_SX104_P/N	RS-422	Input	Switch #104 Sense
39,23	17,18	NC_WP_SPARE2_P/N	Spare	No Connect	Spare
7,40	19,20	WP_SX106_P/N	RS-422	Input	Swithc #106 Sense
24,8	21,22	NC_WP_SPARE3_P/N	Spare	No Connect	Spare
41,25	23,24	WP_CHB_P/N	RS-422	Input	Channel B (vs. A) Status
9,42	25,26	WP_LP_P/N	RS-422	Input	Linear Polarization Status
26,10	27,28	WP_BEAMS	RS-422	Output	Beam Selection
43,27	29,30	WP_AHI_L_P/N	RS-422	Input	A Hi Beam Status
11,44	31,32	WP_BHI_L_P/N	RS-422	Input	B Hi Beam Status
28,12	33,34	WP_STC05_P/N	RS-422	Output	STC Data Bit #5
45,29	35,36	WP_STC04_P/N	RS-422	Output	STC Data Bit #4
13,46	37,38	WP_STC03_P/N	RS-422	Output	STC Data Bit #3
30,14	39,40	WP_STC02_P/N	RS-422	Output	STC Data Bit #2
47,31	41,42	WP_STC01_P/N	RS-422	Output	STC Data Bit #1
15,48	43,44	WP_STC00_P/N	RS-422	Output	STC Data Bit #0
32,16	45,46	WP_STCLK_P/N	RS-422	Output	STC Clock
49,33	47,48	WP_6WX_CLK_P/N	RS-422	Output	6-level WX Sync Clock
17	49	WP_SPR_RXD	RS-232	Input	Spare Serial Link to ASR9
50	50	WP_SPR_TXD	RS-232	Output	Spare Serial Link to ASR9

G.7.6 J2, ASR9 Interface WSP #2

Table 41 Pin-out for J2 ASR9 / WSP #2

Pin Number	Ribbon Cable Number	Signal Name	Type	Direction	
1,34	1,2	WP_PRET0_P/N	RS-422	Input	Pretriggger time zero
18,2	3,4	WP_GATE0_P/N	RS-422	Input	Range Gate zero
35,19	5,6	NC_WP_SPARE4_P/N	Spare	No Connect	Spare
3,36	7,8	WP_ALT00_P/N	RS-422	Input	I/Q Data Bit #0
20,4	9,10	WP_ALT01_P/N	RS-422	Input	I/Q Data Bit #1
37,21	11,12	WP_ALT02_P/N	RS-422	Input	I/Q Data Bit #2
5,38	13,14	WP_013MHZ_CLK	RS-422	Input	1.3 MHz Clock
22,6	15,16	NC_WP_SPARE5_P/N	Spare	No Connect	Spare
39,23	17,18	NC_WP_SPARE6_P/N	Spare	No Connect	Spare
7,40	19,20	NC_WP_SPARE7_P/N	Spare	No Connect	Spare
24,8	21,22	WP_026MHZ_CLK	RS-422	Input	2.6 MHz Clock
41,25	23,24	NC_WP_SPARE8_P/N	Spare	No Connect	Spare
9,42	25,26	WP_103MHZ_CLK	RS-422	Input	10.32 MHz Clock
26,10	27,28	WP_ACP_P/N	RS-422	Input	Antenna Control Pulse
43,27	29,30	WP_ARP_P/N	RS-422	Input	Antenna Reset Pulse
11,44	31,32	NC_WP_SPARE9_P/N	Spare	No Connect	Spare
28,12	33,34	WP_ALT03_P/N	RS-422	Input	I/Q Data Bit #3
45,29	35,36	WP_ALT04_P/N	RS-422	Input	I/Q Data Bit #4
13,46	37,38	WP_ALT05_P/N	RS-422	Input	I/Q Data Bit #5
30,14	39,40	WP_ALT06_P/N	RS-422	Input	I/Q Data Bit #6
47,31	41,42	WP_ALT07_P/N	RS-422	Input	I/Q Data Bit #7
15,48	43,44	WP_ALT08_P/N	RS-422	Input	I/Q Data Bit #8
32,16	45,46	WP_ALT09_P/N	RS-422	Input	I/Q Data Bit #9
49,33	47,48	WP_ALT10_P/N	RS-422	Input	I/Q Data Bit #10
17,50	49,50	WP_ALT11_P/N	RS-422	Input	I/Q Data Bit #11

G.7.7 J3 and J4 RS-232 Interfaces to RVP902-WSP Processor

Table 42 Pin-out for J3 and J4 RS-232 Serial Interface

Pin Number	Signal Name	Type	Direction	Description
1	No Connect	RS-232	NC	
2	RXD (TXD on WSP interface)	RS-232	Input	Receive Data
3	TXD (RXD on WSP interface)	NC	Output	Transmit Data
4	No Connect	NC	Output	
5	GND	GND	GND	Ground
6	No Connect	NC	NC	
7	No Connect	NC	Output	
8	No Connect	NC	NC	
9	No Connect	NC	NC	

G.7.8 J5, Ethernet Interface

RJ45 Ethernet Interface provides 100/1000 Base-T communication with the RVP902-WSP Processor direct or via a network switch. This interface requires Jumbo Frames. RJ-45 signal interface cable side pin out.

Table 43 Pin-out for J6 RJ-45 Interface

J6 Pin Number	Signal Name	Description
1	TRP1+	Positive side of Twisted Pair A
2	TRP1-	Negative side of Twisted Pair A
3	TRP2+	Positive side of Twisted Pair B
4	TRP3+	Positive side of Twisted Pair C
5	TRP3-	Negative side of Twisted Pair C
6	TRP2-	Negative side of Twisted Pair B
7	TRP4+	Positive side of Twisted Pair D
8	TRP4-	Negative side of Twisted Pair D

G.7.9 J6, RVP901-WSP Misc IO A to RCP903 ASR9-WSP Panel

Table 44 RVP900 SIGNAL TYPES IN CBL210313

Type	Destination
GPDIFF_PIN_LP/N	RS-422 signals
TTLIO_PIN/GND	TTL signals
AMUX_P0/AMUX_N0	Differential analog input A/D signals
+5V	+5VDC
-5V	-5VDC
Gnd	Ground

Table 45 CBL210313 RCP903 INTERCONNECT CABLE TO RVP901-WSP MISC IO PORT A (J3)

RCP903 Pin Number	Signal Name	Type	Direction	RVP901 IFDR Pin Number	Logical Connection
1,22	IFDR_PRETO	RS-422	Output	1,19	DIFF0 P,N
2,23	IFDR_HDR_INFO 0_P/N	RS-422	Output	2,20	DIFF1 P,N
3,24	IFDR_HDR_INFO 1_P/N	RS-422	Output	3,21	DIFF2 P,N
4,25	IFDR_HDR_INFO 2_P/N	RS-422	Output	4,22	DIFF3 P,N
5,26	IFDR_HDR_INFO 3_P/N	RS-422	Output	5,23	DIFF4 P,N
6,27	IFDR_HDR_INFO 4_P/N	RS-422	Output	6,24	DIFF5 P,N
7,28	IFDR_HDR_INFO 5_P/N	RS-422	Output	7,25	DIFF6 P,N
8,29	IFDR_HDR_INFO 6_P/N	RS-422	Output	8,26	DIFF7 P,N
9,30	IFDR_HDR_INFO 7_P/N	RS-422	Output	9,27	DIFF8 P,N
10,31	IFDR_HDR_INFO 8_P/N	RS-422	Output	10,28	DIFF9 P,N
11,32	IFDR_HDR_INFO 9_P/N	TTL 5V	Output	11,29	TTL0, GND
12,33	IFDR_HDR_INFO 10_P/N	TTL 5V	Output	12,30	TTL1, GND
13,34	IFDR_HDR_INFO 11_P/N	TTL 5V	Output	13,31	TTL2, GND
14,35	IFDR_HDR_INFO 12_P/N	TTL 5V	Output	14,32	TTL3, GND
15,36	IFDR_HDR_INFO 13_P/N	TTL 5V	Output	15,33	TTL4, GND
16,37	IFDR_HDR_INFO 14_P/N	TTL 5V	Output	16,34	TTL5, GND
17,38	IFDR_HDR_INFO 15_P/N	TTL 5V	Output	17,35	TTL6, GND
18,39	IFDR_HDR_INFO 16_P/N	TTL 5V	Output	36,37	TTL7, GND
19,40	IFDR_HDR_INFO 17_P/N	TTL 5V	Output	38,39	TTL8, GND
20,41	IFDR_HDR_INFO 18_P/N	TTL 5V	Output	40,41	TTL9, GND
49,50	V_5Po	TTL 5V	Output	42,43	AMUX0 P,N
53,54	V_3Po	TTL 5V	Output	46,47	AMUX1 P,N
57,58	V_1P2	TTL 5V	Output	50,51	AMUX2 P,N
43,44	IFD_PRESENT	TTL 5V	Input	44,45	+5V, GND
45,46	NC_V_N5Po	TTL 5V	No Connect	48,49	-5V ,GND
62	GND	TTL 5V	Output	18	GND

G.7.10 J7, Power Interface (DC)

The power supply sub-system is designed with the following assumptions about the input power source.

- Input Voltage: Nominal 24V ±4V
- Maximum Power Consumption: 24W
- Maximum input noise 1.0Vp-p max at 50-120 Hz, 100 mVp-p max 120-300 kHz
- Low Voltage Directive Compliant (SELV)
- Panel power connector is a DB9-M. Power cord should have a female DB9 connector, DB9-F
- Cable to power connector should use 18-gauge wire with power and ground twisted. If the cable is not shielded a ferrite should be added as close to the connector as possible with the cable looped through twice.

Table 46 Pin-out for J7 Power Input

J6 Pin Number	Signal Name	Description
1	PWR_IN	24V nominal input power
2	PWR_IN	24V nominal input power
3	NC	
4	PWR_RETURN	Return path (ground) for input power
5	PWR_RETURN	Return path (ground) for input power
6	PWR_IN	24V nominal input power
7	PWR_IN	24V nominal input power
8	PWR_RETURN	Return path (ground) for input power
9	PWR_RETURN	Return path (ground) for input power

G.8 ASR9 RIM Software API

The original RxNet7 Radar Interface Module (RIM) API functionality has been ported to the RVP902-WSP and modified to use the new RCP903 Panel Ethernet Interface for control and status functionality.

In addition to the original RIM commands, new functions have been added that allow monitoring of board status and manufacturing tests. New commands are marked with a * next to their name. Commands that are no longer needed because those physical interfaces no longer exist have been deprecated. Deprecated commands are listed at the end of the section.

This section gives an overview of the functionality implemented. Details about API functionality can be found in the code comments.

The RIM API includes the following functionality:

Table 47 RIM API Board Status Functions

Function Name	Description
rim_board_status*	Returns the boards complete operational status of all hardware interfaces
rim_board_status_clear*	Clears persistent error status bits from designated hardware interface
rim_asr9_flash_status*	Returns the status of the flash memory contents
rim_asr9_brd_status*	Returns the status of the boards voltage and temperature
rim_board_revlevel	Returns the hardware revision level
rim_software_revlevel	Returns the software revisions level
rim_pldreset	Reboots the board returning it to its original power on state
rim_led_clrset	Controls the test output LED state
rim_board_sw_status_set*	Configures the embedded processors performance monitor
rim_board_sw_status_get*	Returns status of the embedded processors performance
rim_board_sw_errlog_set*	Configures the boards error message logging functionality
rim_board_sw_errlog_get*	Gets error log messages
rim_testport_set*	Configures the signal test port header
rim_testport_get*	Gets the current configuration of the signal test port header

Table 48 RIM API 6-Level Weather Functions

Function	Description
rim_sio_reset	Resets the 6-level weather hardware and status and loads default line state
rim_sio_setclock	Sets the baud rate and clock phase of the 6-level weather interface
rim_sio_setidle	Sets the idle pattern of the 6-level weather interface
rim_sio_sendbuf	Sends data to be transmitted to the 6-level weather interface
rim_sio_readbuf*	Reads data back from 6-level weather interface if it is in loopback mode
rim_sio_leds	Places the 6-level weather LEDs in normal or override operation
rim_sio_setconfig*	Sets all of the 6-level weather interface configuration parameters in one call
rim_sio_getconfig*	Gets current value of all of the 6-level weather interface parameters
rim_sio_getstatus*	Gets current status of 6-level weather interface

Table 49 RIM API Diagnostic IO Functions

Function	Description
rim_set_out_state*	Configures forced state and output value the user specified outputs
rim_get_out_state*	Gets the forced state and output value of all outputs
rim_get_io_state*	Reads the current logic level on all inputs and output pins
rim_set_line_state	Forces a single output to the user defined level
rim_get_line_state	Reads the logic level at the pin of the input or output requested
rim_set_line_state_cmd	Returns the line being forced by rim_set_line_state
rim_get_line_state_cmd*	Returns the line state of the user defined output that would be forced in override mode

Table 50 RIM API Clock Functions

Function	Description
rim_asr9_clk_align	Aligns the ASR9 8x clock edge to the 1x clock, compensating for any delays in the system
rim_asr9_clk_set	Sets the phase of the ASR9 8x clock
rim_asr9_clk_get	Gets the phase of the ASR9 8x clock
rim_asr9_clk_status*	Gets the current status of the ASR9 and RCP903 clocks

Table 51 RIM API Azimuth Functions

Function	Description
rim_asr9_getaz*	Gets the current antenna position and rotation count

Table 52 RIM API STC Functions

Function	Description
rim_asr9_stcsel	Sets the STC select map bits
rim_asr9_stcsel_get	Gets the STC select map bits
rim_asr9_stcloadsect	Loads the STC map data for a user defined sector and map
rim_asr9_stcreadsect*	Reads the STC map data for a user defined sector and map
rim_asr9_stc_dataphase_get	Gets the STC data phase relative to the ASR9 1x clock

Table 52 RIM API STC Functions

Function	Description
rim_asr9_stc_dataphase_set	Sets the STC data phase relative to the ASR9 1x clock
rim_asr9_stc_clkphase_get	Gets the STC data clock relative to the ASR9 1x clock
rim_asr9_stc_clkphase_set	Sets the STC data clock relative to the ASR9 1x clock
rim_asr9_stc_status*	Gets the current status of the STC interface

Table 53 RIM API Data Processing Controls

Function Name	Description
rim_asr9_set_run	Sets the RCP903 processing state
rim_asr9_set_run_cmd	Gets the RCP903 processing state that is currently being requested
rim_asr9_get_run	Gets the RCP903 processing state that is currently running
rim_asr9_set_udp_addr*	Sets the UDP address and port information for the IQ data stream
rim_asr9_get_udp_addr*	Gets the UDP address and port information for the IQ data stream
im_asr9_iqdata_status*	Gets the current status of the IQ Data Interface

Table 54 RIM API Simulated Controls

Function	Description
rim_asr9_asr9_simclocks	Sets the simulation state of the ASR9 clocks
rim_asr9_asr9_simclocks_get	Gets the simulation state of the ASR9 clocks
rim_asr9_simantenna	Sets the simulation state of the antenna signals (WP_ACP, WP_ARP)
rim_asr9_simantenna_get	Gets the simulation state of the antenna signals (WP_ACP, WP_ARP)
rim_asr9_simiq	Sets the simulation state of the I/Q data
rim_asr9_simiq_get	Gets the simulation state of the I/Q data
srim_asr9_simiq_data_get*	Gets the simulation values of the I/Q data
rim_asr9_simtrigs	Sets the simulation state of the PRET0
rim_asr9_simtrigs_get	Gets the simulation state of the PRET0
rim_asr9_swst_sim_set	Sets the operational mode and simulation override value of a single ASR9 status input
rim_asr9_swst_sim_get	Gets the operational mode and simulation override value of a single ASR9 status input
rim_asr9_swcnt_sim_set	Sets the operational mode and simulation override value of a single ASR9 control output

Table 54 RIM API Simulated Controls

Function	Description
rim_asr9_swcnt_sim_get	Gets the operational mode and simulation override value of a single ASR9 control output
rim_asr9_103104sts_sim_set	Sets the operational mode and simulation override values for the WP_SW103 and WP_SW104 input levels
rim_asr9_103104sts_sim_get	Gets the operational mode and simulation override values for the WP_SW103 and WP_SW104 input levels
rim_asr9_103104cnt_sim_set	Sets the operational mode and simulation override values for the WP_CX103 and WP_CX104 output levels
rim_asr9_103104cnt_sim_get	Gets the operational mode and simulation override values for the WP_CX103 and WP_CX104 output levels

Table 55 Deprecated Functions

Function Type	Function Names	Reason for Deprecation
Mutex functions	Rim_mutex_define, rim_mutex_lock, rim_mutex_unlock.	Mutex functionality exists in the light weight remote procedure call routines. (lwrpc)
IO functions	Rim_port_init, rim_port_read, rim_port_write, rim_pld_serial, rim_board_allow.	Hardware interface no longer exists.
Memory access functions	Rim_mem_read, rim_mem_write, rim_memset, rim_mem_read_nomutex, rim_mem_write_nomutex, rim_memset_nomutex.	Memory interface no longer exists.
RINPE functions	Rim_set_rinpe_run, rim_set_rinpe_run_cmd, rim_get_pinpe_run.	RINPE hardware interface no longer exists replaced by Ethernet interface.
RVP simulation functions	Rim_rvp7_simiq, rim_rvp7_simiq_get.	RVP9 simulation functionality does not exist.

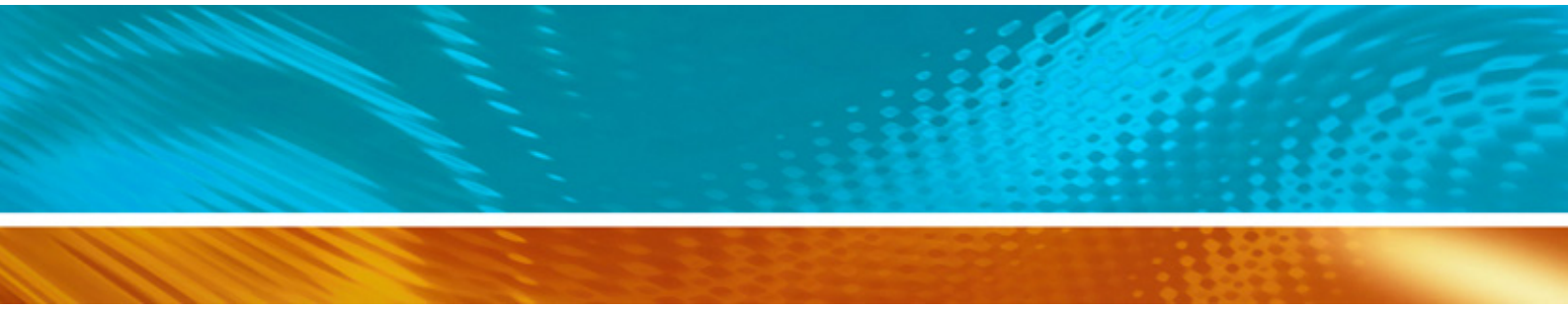
APPENDIX H
ACRONYMS

API	Application Program Interface
COHO	Coherent Local Oscillator
CSR	Clutter-to-signal ratio
DAFC	Digital AFC Module
DFT	Discrete Fourier Transforms
FFT	Fast Fourier Transforms
FPGA	Field Programmable Gate Array
FIR	Finite Impulse Response
GMAP	Gaussian Model Adaptive Processing
IFDR	Intermediate Frequency Digital Receiver
LRMSK	Load Range Mask
NSSL	National Severe Storms Laboratory
PLD	Programmable Logic Device
PMI	Polarimetric Met Index
PPP	Poly-Pulse Pair
RCW	Radar Control Workstation
RF	Radio Frequency
RPG	Radar Product Generator
SNR	Signal-to-noise ratio
SQI	Signal Quality Index
STALO (RF-IF)	Stable Local Oscillator
STAR	Simultaneous Transmit and Receive
TCF	Threshold Control Flags
VCXO	Voltage Controlled Crystal Oscillator
WSP	Weather System Processor

APPENDIX I
REFERENCES AND CREDITS

1. Dazhang, T., S.G. Geotis, R E. Passarelli Jr., A.L. Hansen, and C.L. Frush, 1984: Evaluation of an Alternating–PRF Method for Extending the Range of Unambiguous Doppler Velocity. *Preprints of the 22nd Conference on Radar Meteorology*, American Meteorological Society, 523-527.
2. Doviak, R.J., and D.S. Zrni?, 1993: Doppler Radar and Weather Observerations. *Academic Press*, San Diego.
3. Joe, P., and A. Siggia, 1995: Second Trip Unfolding by Phase Diversity Techniques. *Preprints of the 27th Conference on Radar Meteorology,* American Meteorological Society, 770-772.
4. Sachidananda, M., D.S. Zrni?, and R.J. Doviak, 1997: Signal Design and Processing Techniques for WSR–88D Ambiguity Resolution. *National Severe Storms Laboratory Report, Part 1*, Norman, OK, 100 pp.
5. The JMA internal specification for Interference Filter algorithm for use on Chitose airport Doppler weather radar is the basis for Alg.1 described in Interference Filter on page 189
6. Environment Canada – Aldergrove BC, kindly supplied the snapshot of receiver data that is plotted in Figure 30 on page 163

INDEX



www.vaisala.com

4*M211322EN*