M211322EN-H

User Guide

RVP900 Digital Receiver and Signal Processor

RVP900

VAISALA

PUBLISHED BY

Vaisala Oyj
Street address: Vanha Nurmijärventie 21, FI-01670 Vantaa, Finland
Mailing address: P.O. Box 26, FI-00421 Helsinki, Finland
Phone: +358 9 8949 1
Fax: +358 9 8949 2227

Visit our Internet pages at www.vaisala.com.

© Vaisala 2017

No part of this manual may be reproduced, published or publicly displayed in any form or by any means, electronic or mechanical (including photocopying), nor may its contents be modified, translated, adapted, sold or disclosed to a third party without prior written permission of the copyright holder. Translated manuals and translated portions of multilingual documents are based on the original English versions. In ambiguous cases, the English versions are applicable, not the translations.

The contents of this manual are subject to change without prior notice.

Local rules and regulations may vary and they shall take precedence over the information contained in this manual. Vaisala makes no representations on this manual's compliance with the local rules and regulations applicable at any given time, and hereby disclaims any and all responsibilities related thereto.

This manual does not create any legally binding obligations for Vaisala towards customers or end users. All legally binding obligations and agreements are included exclusively in the applicable supply contract or the General Conditions of Sale and General Conditions of Service of Vaisala.

This product contains software developed by Vaisala or third parties. Use of the software is governed by license terms and conditions included in the applicable supply contract or, in the absence of separate license terms and conditions, by the General License Conditions of Vaisala Group.

This product may contain open source software (OSS) components. In the event this product contains OSS components, then such OSS is governed by the terms and conditions of the applicable OSS licenses, and you are bound by the terms and conditions of such licenses in connection with your use and distribution of the OSS in this product. Applicable OSS licenses are included in the product itself or provided to you on any other applicable media, depending on each individual product and the product items delivered to you.

Table of Contents

List of Figures

List of Tables

1. About This Document

1.1 Version Information

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

Table 1 Document Versions

Document Code	Description
M211322EN-H	Eigth version. This manual. May 2017
M211322EN-G	Seventh version. This manual. February 2017
M211322EN-F	Sixth version. March 2016
M211322EN-E	Fifth version. October 2015

1.2 Related Documents

Table 2 Weather Radar Documentation

Document Code	Name
M211315EN	*IRIS and RDA Software Installation Guide*
M211318EN	*IRIS Programming Guide*
M211316EN	*IRIS and RDA Utilities Guide*
M211319EN	*IRIS Product and Display Guide*
M211317EN	*IRIS Radar User Guide*
M211452EN	*IRIS and RDA Dual Polarization User Guide*
M211322EN	*RVP900 Digital Receiver and Signal Processor User Guide*
M211320EN	*Radar Control Processor RCP8 User Guide*

For information on changes made since your current release was installed, download the latest document versions and check the IRIS and RDA Release Notes from www.vaisala.com.

Vaisala encourages you to send your comments or corrections to helpdesk@vaisala.com

1.3 Documentation Conventions

WARNING! Warning alerts you to a serious hazard. If you do not read and follow instructions 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 highlights important information on using the product.

Tip gives information for using the product more efficiently.

Lists tools needed to perform the task.

Indicates that you need to take some notes during the task.

1.4 Trademarks

Vaisala® is a registered trademark of Vaisala Oyj.

Digi TransPort® is a registered trademark of Digi International Inc.

All other product or company names that may be mentioned in this publication are trade names, trademarks, or registered trademarks of their respective owners.

2. Product Overview

2.1 RVP900 Product Architecture

RVP900 provides a full suite of weather radar signal processing functionality to support implementing a new weather radar or upgrading an existing system to the latest signal processing technology.

The flexible architecture provides modular, highly configurable hardware, and an open software design with public APIs that support developing applications and algorithms using RVP900 as a foundation.

RVP900 includes the following hardware components.

RVP901 IF Digital Receiver

RVP901 IF Digital Receiver (IFDR) provides receive, transmit, and IF detector functionality in a single, compact network-attached FGPA-based product able to perform tens of billions of multiply accumulate cycles per second.

The IFDR samples up to 4 receive channels and 1 burst channel of IF inputs and computes **I** and **Q** data from them. The **I** and **Q** data is transmitted over Gigabit Ethernet to the RVP902 signal processor or third party Linux server for further processing into moments.

The fully digital transmit functionality can produce outputs as simple as a COHO for a Klytsron system or as complex as a pulse compression waveform.

See 3.1 RVP901 IF Digital Receiver (page 23).

RVP902 Signal Processor

RVP902 Signal Processor is server class computer with a dual processor multicore Xeon processor running Linux.

RVP902, running RDA software, computes the radar data moments (**Z**,**T**,**V**,**W**) from the **I** and **Q** data provided by the IFDR. The moments can be distributed internally on RVP902 or externally to other computers running IRIS or third party software.

See 3.2 RVP902 Signal Processing Computer (page 38).

DAFC Digital Automatic Frequency Control Unit

RVP901 and RVP902 use comprehensive, configurable I/O interfaces to provide DAFC functionality in a magnetron system, generate triggers, monitor status, and control the radar.

Much of this can be done through the RVP901 IFDR in combination with RVP902 and RCP8.

See 3.6 Digital AFC (DAFC) (page 51) and *Radar Control Processor RCP8 User Guide*.

2.2 Regulatory Compliances

More Information

- ASR9-WSP with RCP903 ASR9-WSP Panel Overview (page 447)
- RCP902 WSR98D Panel Architecture (page 425)

2.2.1 WEEE Compliance

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

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

2.2.2 RoHS Compliance

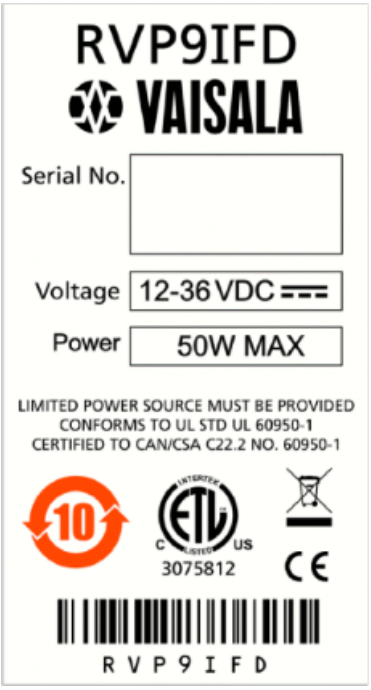
The RoHS Directive 2002/95/EC restricts the use of six hazardous materials found in electrical and electronic products. All applicable products in the EU market after July 1, 2006 must pass RoHS compliance. The maximum permitted concentrations are 0.1% or 1000 ppm (except for cadmium, which is limited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

2.2.2.1 China RoHS Compliance

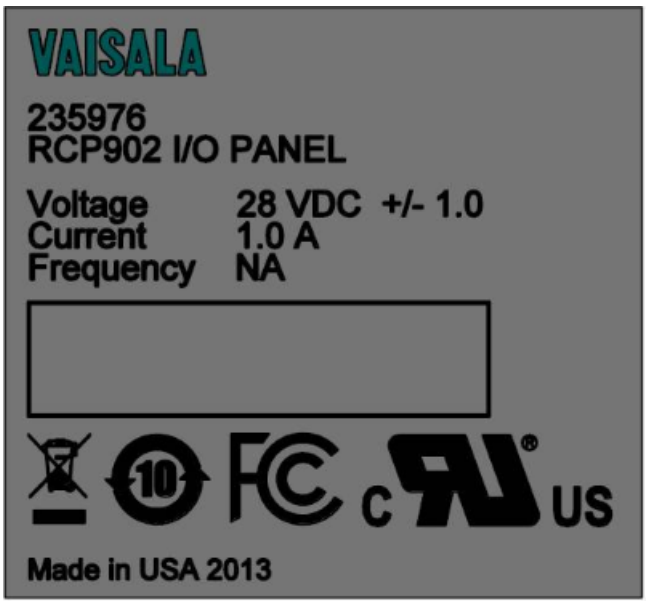
The China RoHS Directive requires disclosure (not removal) of the 6 EU RoHS substances for those products included in the "List". Disclosure can be at the component or at the sub assembly level, but it has to be in the prescribed format, in Chinese, as detailed in the document "Marking for the control of Pollution Caused by Electronic Information Products".

There are product marking requirements and a calculation of the "Environmentally Friendly Use Period" to be calculated.

RVP9IFD China RoHS Compliance

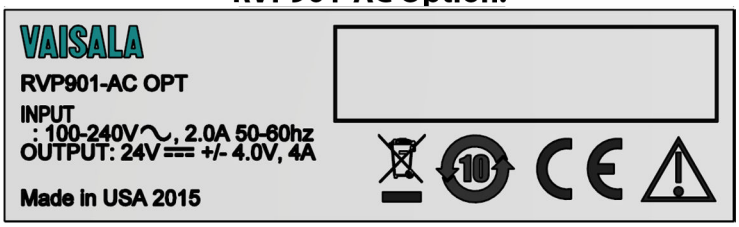


RCP902 WSR98D Panel China RoHS Compliance

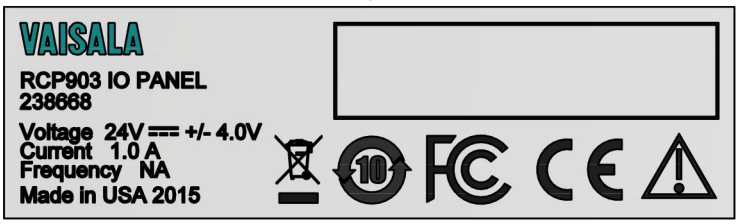


RVP901-AC and RCP903 China RoHS Compliance

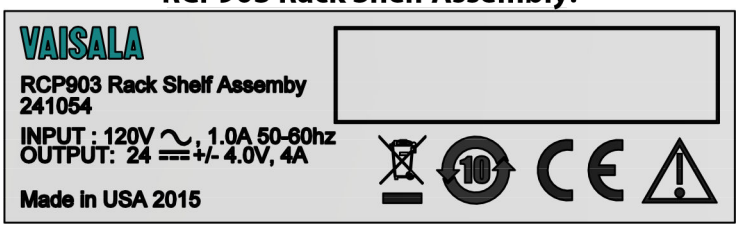
RVP901-AC Option:



RCP903 I/O Panel:



RCP903 Rack Shelf Assembly:



2.3 Safety

WARNING! Turn off power to the main chassis before installing or removing PCI boards. Disconnect the line cord should be disconnected before opening either the IFD module or main chassis.

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 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.**

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

2.3.1 ESD Protection

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

To avoid delivering high static voltages to the product:

- Handle ESD-sensitive components on a properly grounded and protected ESD workbench or by grounding yourself to the equipment chassis with a wrist strap and a resistive connection cord.
- If you are unable to take either precaution, touch a conductive part of the equipment chassis with your other hand before touching ESD-sensitive components.
- Hold component boards by the edges and avoid touching component contacts.

2.4 Product Nomenclature

Table 3 RVP900 Nomenclature

Code	Common Name
220031	Keyboard and display

Code	Common Name
RVP901	RVP Intermediate Frequency Digital Receiver (IFDR)
RVP902	Server Computer, RVP900 Signal Processor
RVP902-IO	Server with IO62 Card
RVP903	Signal Processing Computer with RCP8
RCP902, RCP903, RCP904	Panel Assemblies for WSR98D (custom IO panel)
RCP9CP-TDWR	TDWR Panel Assembly (custom IO panel)

3. Functional Description

3.1 RVP901 IF Digital Receiver

The RVP Intermediate Frequency Digital Receiver (IFDR) receives and digitizes the IF signal. The resulting digital **I** and **Q** data is sent to the RVP900 software on the radar server computer processing into data products.

RVP901 IF Digital Receiver (IFDR) is a network-attached FGPA-based product that provides the receive and transmit functionality needed in a weather radar signal processor. From the sampled IF, input it generates **I** and **Q** data that is sent to the RVP902 Signal Processor over standard Gigabit Ethernet. It provides configuration and status functionality over TCP/IP and data transfer using UDP packet formats.

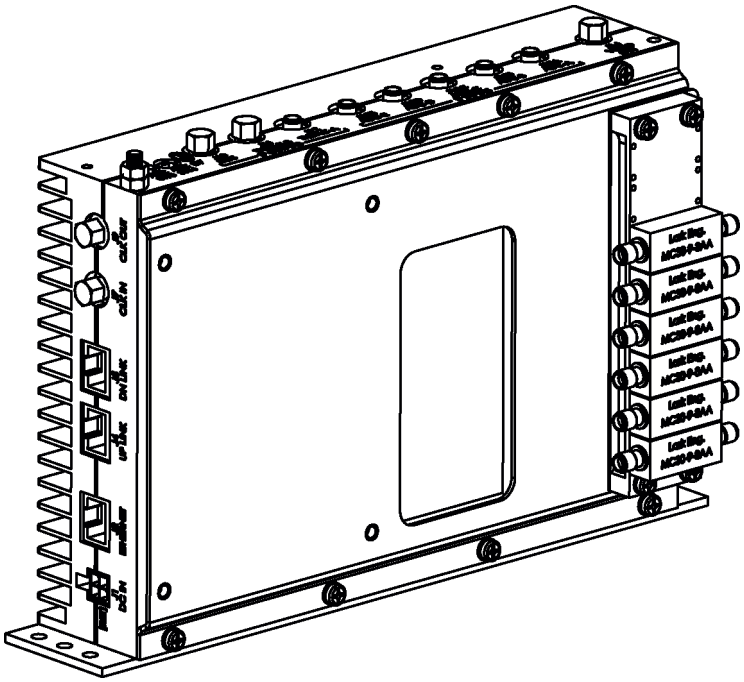


Figure 1 RVP901 IFDR

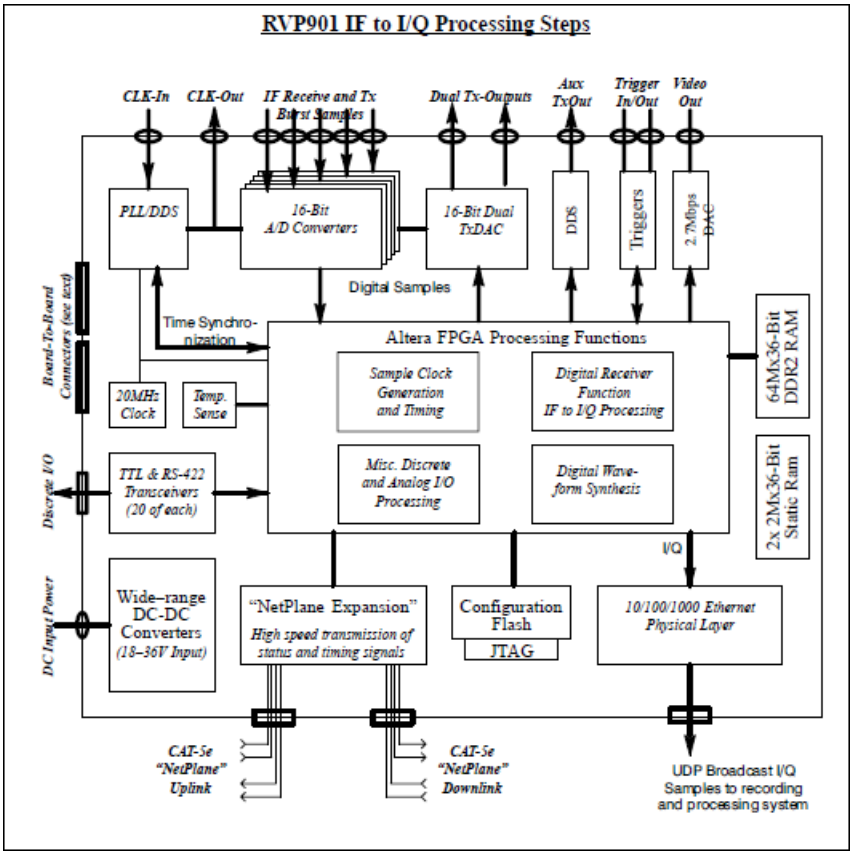


Figure 2 IF to I/Q Processing Steps

More Information

- RVP901 IFDR Hardware (page 56)
- RVP901 IFDR Specifications (page 326)
- RVP901 IFDR Technical Drawings (page 365)

3.1.1 RVP901 IFDR Data Capture and Timing

RVP901 IFDR handles digital input and output, such as triggers, polarization switch controls, and pulse width control.

Input

The primary input to the RVP901 IFDR is the received IF signal. The RVP901 IFDR has five 16-bit A/D converters to sample the transmit pulse and up to 4 receiver channels.

An external clock may be used to phase lock the A/D conversion with the transmit pulse (not used for magnetron systems). Alternatively, the stable internal clock of the IFDR can be used as the reference clock for the entire radar system.

Output

IF transmit waveforms are synthesized by the IFDR and can be output over two SMA 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.

Other applications include pulse compression and frequency agility scanning.

Timing

The sampling clock in the IFDR is selected for stability. The sample clock serves a similar function to the COHO on a traditional Klystron system; it is the master time keeper. The RVP901 IFDR sample clock is used to phase lock the entire RVP900: the Rx, Tx, miscellaneous I/O are phase-locked to the IFDR sample clock. It is and is programmable over a wide range of frequencies.

3.1.2 RVP901 IFDR IF to I and Q Processing

RVP901 IFDR performs parallel FIR filtering, simultaneously on each channel. This allows frequency agility receiving functions within the bandwidth of the analog receiver, which unifies the transmit frequency agility to provide advanced signal processing.

The 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. The frequency, phase, and amplitude of the burst pulse are also measured.

In sum, the processor performs the following functions:

- Band pass filters the IF signal using configurable digital FIR filter matched to the pulse width. This also includes:
  - Range gating and optional coherent averaging
  - Calculating `I` and `Q` quadrature values
- Calculates transmit burst sample frequency, phase, and amplitude
- Corrects `I` and `Q` phase and amplitude based on transmit burst sample
- Runs the interference rejection algorithm
- Calculates the AFC frequency error with output to IFDR for digital control of STALO (for magnetron systems)

The digital approach allows the software algorithms for these functions to be easily changed. Configuration information (for example, processor major mode, PRF, pulse width, gate spacing, and so on.) is supplied from the host computer.

More Information

- Digital IF Band Pass Design Tool (page 28)
- IF Signal Processing (page 170)

3.1.3 Digital Receiver Function

A digital receiver can perform the following functions:

Accept the transmit pulse burst sample to measure its frequency, phase, and power

RVP901 receives the analog receive waveforms and digitizes IF samples.

The receiver electronics (LNA, RF mixer, IF preamp, and IFDR) can be up to 25 m away from the RVP902 server chassis. This makes it possible to optimize the locations of 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.

Amplitude measurement and correction of transmitted pulse (from burst sample)

The IFDR computing power, in the form of an FPGA that can execute 38.4 billion multiply/accumulate cycles per second, allows the use of multiple finite impulse response (FIR) filter arrays to run simultaneously.

The FPGA serve as the first stage of processing of the raw IF data samples. It performs 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 up to 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 can produce over-sampled (`I`,`Q`) time series with 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. In RVP900, the bin spacing of the (`I`,`Q`) data can be set to any value between 25 m ... 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 0.

Dual-polarization radars that are capable of simultaneous reception for both horizontal and vertical channels are interfaced to the same piece of hardware. Because 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.

The following figure 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.

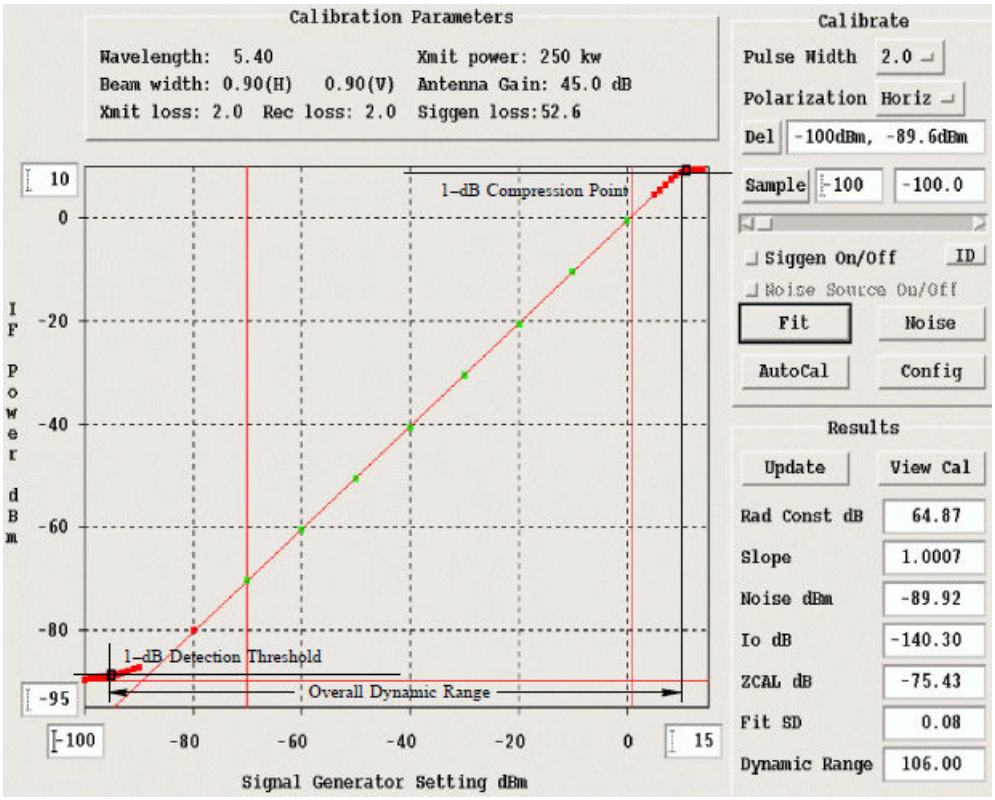


Figure 3 Calibration Plot for RVP901

Frequency measurement for AFC output (from burst sample)

See 7.2.3 Automatic Frequency Control (AFC) (page 174).

I and Q calculation over a wide dynamic range (WDR)

The RVP902 signal processing computer processes the I and Q values to obtain the moment data (for example, Z, V, W, or polarization variables).

IF band pass filtering

See 3.1.3.2 Digital IF Band Pass Design Tool (page 28).

Phase measurement and correction of transmitted pulse for magnetron systems (from burst sample)

The following table shows the RVP900 real-time signal corrections to the I/Q samples from the Rx.

Table 4 Real-time Signal corrections to I/Q Samples

Correction	Description
Amplitude Correction	RVP900 computes a running average of the transmit pulse power in the magnetron burst channel in real-time. 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 measured phase. The coherency achievable is better than 0.1° by this technique.
Large Signal Linearization	When an IF signal saturates, there is still considerable information in the signal since only the peaks are clipped. The proprietary large signal linearization algorithm used in RVP900 provides an extra 3 ... 4 dB of dynamic range by accounting for the effects of saturation.

The configuration and test utilities can run either locally or remotely over the network.

Advantages

- Achieve a wide linear dynamic range (for example, >95 dB depending on pulse width), without using AGC circuits, which are complex to build, calibrate, and maintain
- Dual or multiple IF multiplexing
- Improved remote monitoring down to the IF level
- Lower initial cost by eliminating most IF receiver components
- Lower life-cycle cost due to reduced maintenance
- Programmable band pass filter
- Selectable IF frequency
- Software-controlled AFC with automatic alignment
- Software-controlled modules can be adapted to many radars and operational requirements.

3.1.3.1 Dual Frequency Receive Options

RVP900 IF Digital Receiver (IFDR) performs 38.4 billion multiply accumulate cycles per second.

Parallel finite impulse response (FIR) filter processing blocks permit the deployment of simultaneous dual frequency receive strategies.

The digitally synthesized transmitter function allows for 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.

3.1.3.2 Digital IF Band Pass Design Tool

The digital matched filter computes `I` and `Q` interactively using a TTY and oscilloscope for graphical display.

You can choose the filter's passband width and impulse response length, and RVP constructs the filter coefficients. You can display the frequency response of the filter and compare it to the frequency content of the 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. To minimize the risk of substantial interference to a weather radar system, RVP supports interference rejection algorithms. See 7.2.5 Interference Filter (page 176).

The RVP901 IFDR places the WDR `I` and `Q` samples on the Ethernet line, where they are sent to the computer's processor. The `I` and `Q` values are then processed to extract the moment information (`Z`, `V`, `W`, and optional polarization parameters).

Using the Digital IF Band Pass Filter

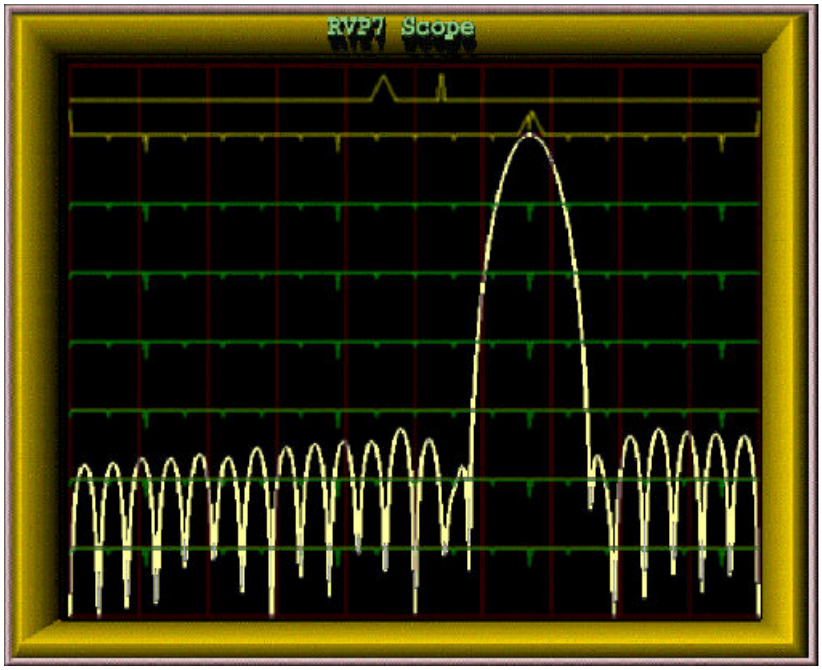


Figure 4 Digital IF Band Pass Design Tool

You can use the filter design tool to design the optimal IF filter to match each pulse width and application.

To show the filter, specify the impulse response and pass band. Use keyboard commands to widen or narrow the filter and to automatically search for an optimal filter.

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

More Information

- RVP901 IFDR IF to I and Q Processing (page 25)

3.1.3.3 Burst Pulse Alignment

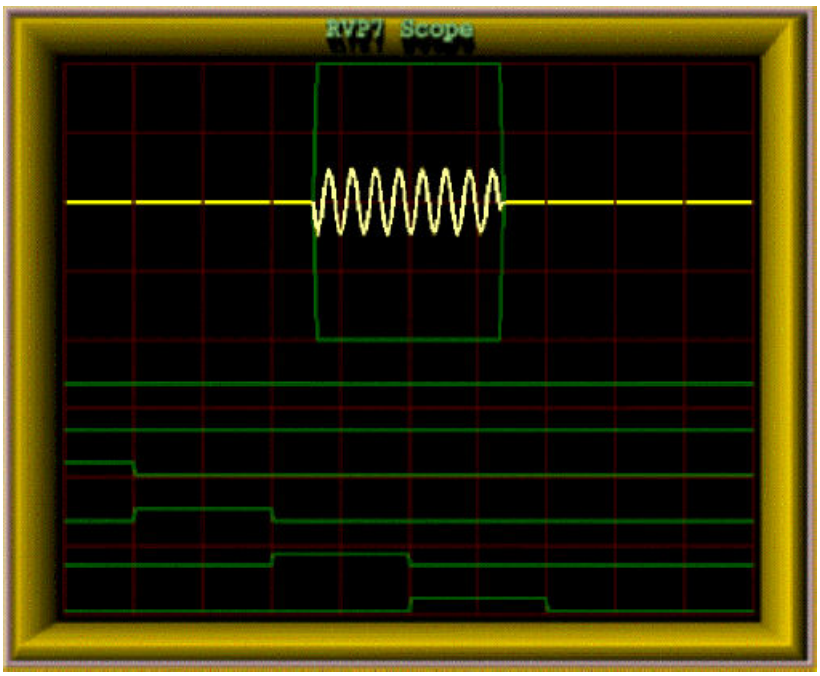


Figure 5 Burst Pulse Alignment Tool

You can check the quality assessment of the transmit burst pulse and its precise alignment at range 0:

- *Manually*, with the burst pulse alignment tool
  This is often done from a central maintenance site.
- *Automatically*, with the burst pulse auto-track feature
  This makes the automatic frequency control (AFC) robust to start-up temperature changes and pulse width changes that can affect the magnetron frequency.

Quality assessments perform a 2D search in both time and frequency space if a valid burst pulse is undetected.

3.1.3.4 Received Signal Spectrum Analysis Tool

RVP900 provides plots of the IF signal versus range as well as spectrum analysis of the signal.

You can perform detailed analysis and configuration from a central maintenance facility through the network.

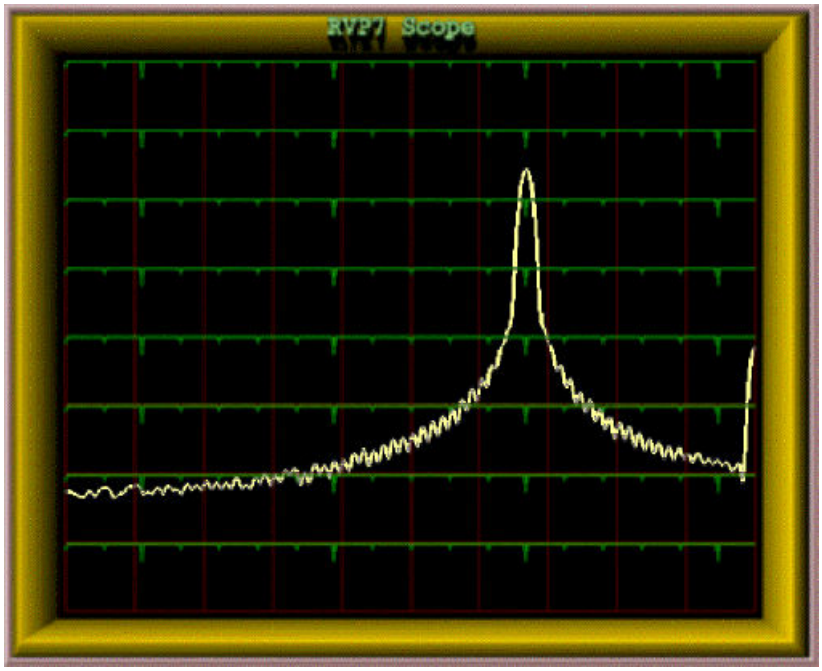


Figure 6 Received Signal Spectrum Analysis Tool

3.1.4 Digital Transmitter Function

RVP900 can function as a digital radar receiver and transmitter simultaneously. 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 transmitter can be a solid state or vacuum tube device. RVP900 can correct for waveform distortion by adaptively "pre-distorting" the transmit waveform, based on the measured transmit burst sample.

The IFDR has 2 SMA outputs for the IF Tx waveform. The digital IF waveforms are generated by a 16-bit DAC with 65 dB SNR from 10 MHz ... 95 MHz. There is also 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 complements the digital receiver in the radar system. The receiver samples an IF waveform that has been down-converted from RF. The transmitter synthesizes an IF waveform for up-conversion to RF. This gives RVP900 control over both halves of the radar, making matched Tx/Rx processing algorithms possible.

Table 5 Example Tx/Rx Processing Algorithms

Algorithm	Description
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. This requires additional hardware and cabling in the radar cabinet, and the phase/amplitude characteristics may not be precise or repeatable. RVP900 can perform precise phase modulation to any desired angle, without requiring external phase shifting hardware.
Pulse Compression	Placing radars in urban areas requires addressing strict regulations on transmit emissions. Because the peak transmit power is often limited in these areas, the weather radar must illuminate its targets using longer pulses at lower power. However, a simple long pulse lacks the ability (bandwidth) to discern targets in range. The solution is to increase the Tx bandwidth by modulating the overall pulse envelope to restore a reasonable range resolution. The fidelity of the RVP900 waveform can accomplish this without introducing the spurious modulation components that often occur when using external phase modulation hardware.
Frequency Agility	RVP900 frequency agility makes this as simple as changing the center frequency of the synthesized IF waveform. Many 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	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. This is an easy way to get a dedicated oscillator at a random frequency in the IF band.

3.1.5 Magnetron Receiver Example

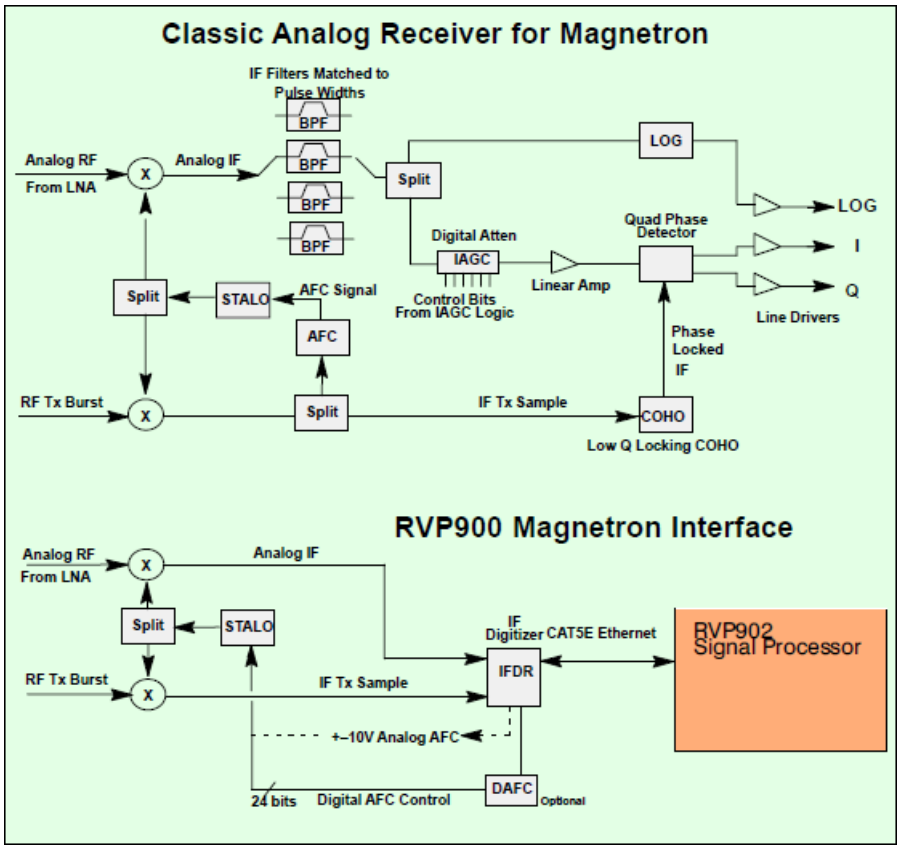


Figure 7 Analog vs Digital Receiver for Magnetron Systems

Analog Receiver - Magnetron

In a typical magnetron system analog receiver, 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.

Phase distortion within the log amplifier renders it unsuitable for making Doppler measurements. A separate linear channel is 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° and 90° version of the `COHO`. The latter is usually obtained by sampling a portion of the transmitted pulse, and then phase locking a COHO that continues to "ring" afterward. Phase locked COHO of this sort can be 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.

RVP Digital Receiver - Magnetron

For the RVP900 digital receiver, the old parts that 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 are the complete interface to the radar's internal signals. No other connections are required in the receiver cabinet.

3.1.5.1 RVP900 Example Configuration for Magnetron

The following example shows a basic magnetron system that can perform DFT processing in 4200 range bins with advanced algorithms such as random phase second trip echo filtering and recovery.

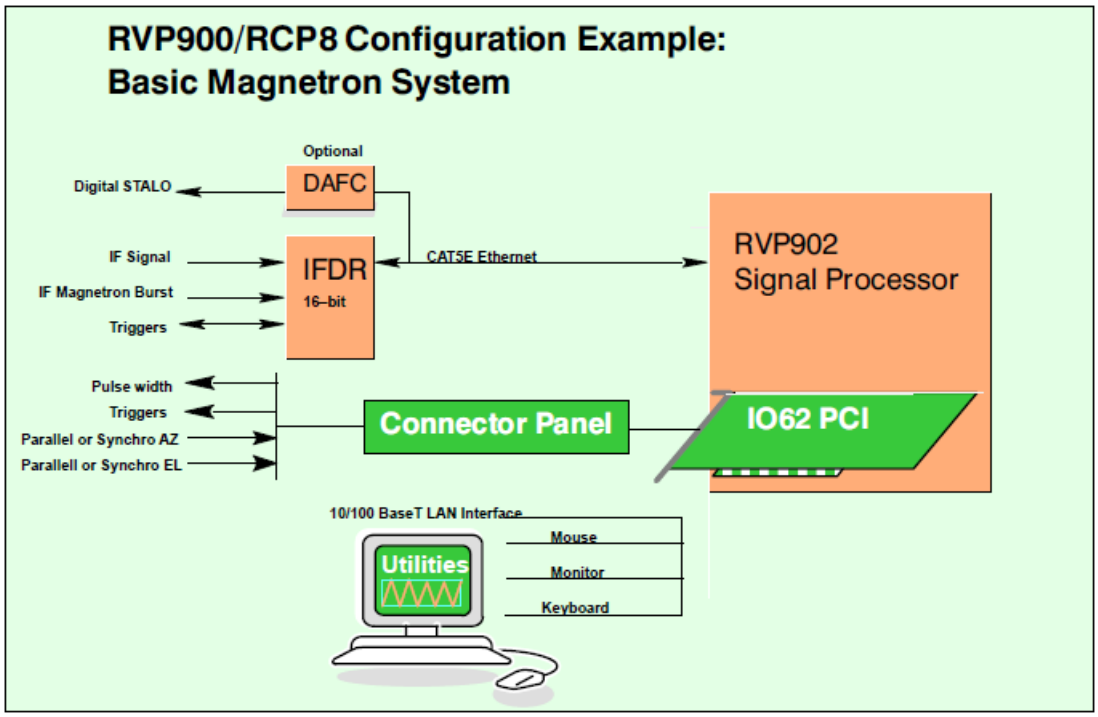


Figure 8 Basic Magnetron System

RVP901 IF Digitizer Receiver (IFDR)	IDR installed in the radar receiver cabinet up to 25 m from RVP902 main chassis. The optional Digital Automatic Frequency Control (DAFC) interfaces to a digitally controlled STALO. RVP900 provides full AFC control with burst pulse auto-tracking.
I/O-62 PCI Card	Available for additional triggers, parallel, synchro or encoder AZ and EL angle inputs, pulse width control, spot blanking control output, and more. The signals are brought in through the connector panel.
RVP902 Signal Processor	19″ rack-mounted computer with a dual processor multi-core running a Linux operating system.

3.1.5.2 RVP900 Example Configuration for Dual Polarity Magnetron

The following figure shows a dual receive channel on the RVP901 IFDR.

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. No additional signal processing hardware is required to convert to a dual-polarization solution. The functionality is an optionally licensed feature.

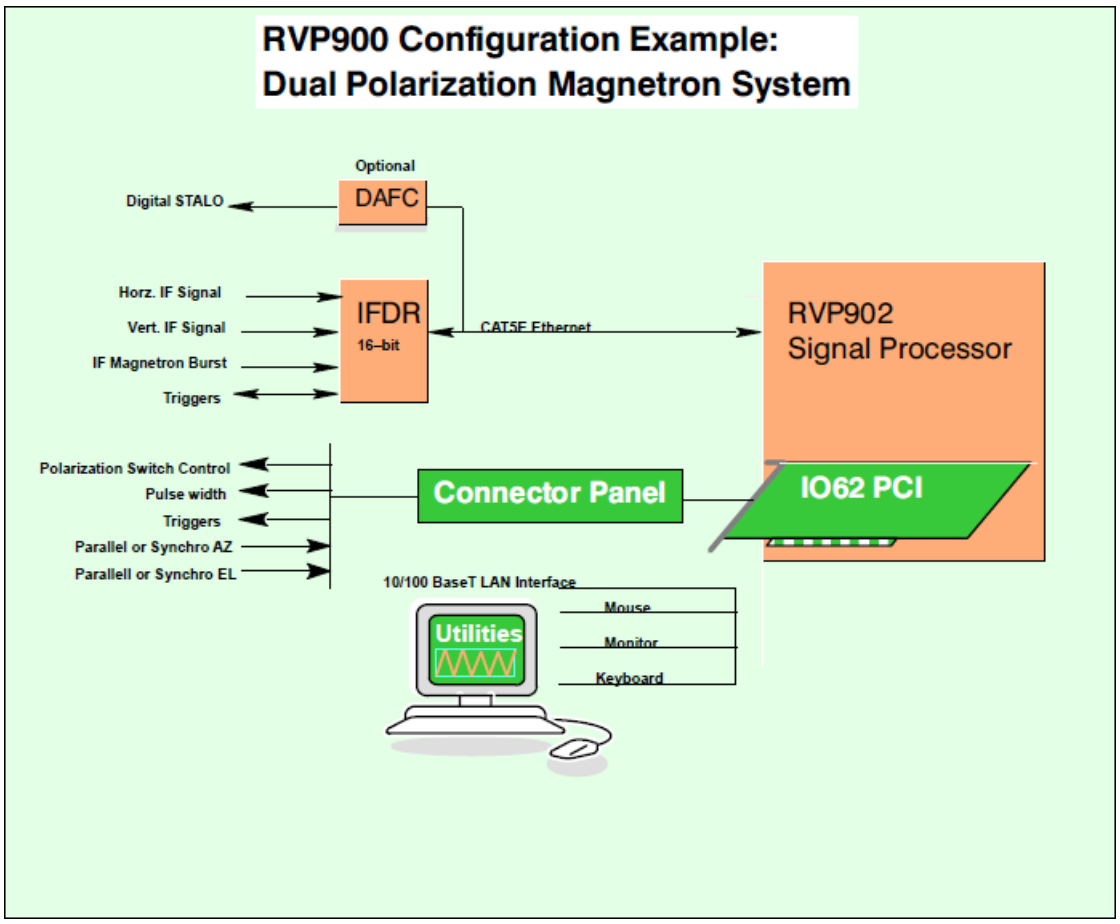


Figure 9 Dual Polarization Magnetron System

RVP900 supports calculation of the complete covariance matrix for dual pol, including, for example, Z_{dr}, **PhiDP** (K_{dp}), **RhoHV**, **LDR**. 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).

3.1.6 Klystron or TWT Receiver and Transmit RF Example

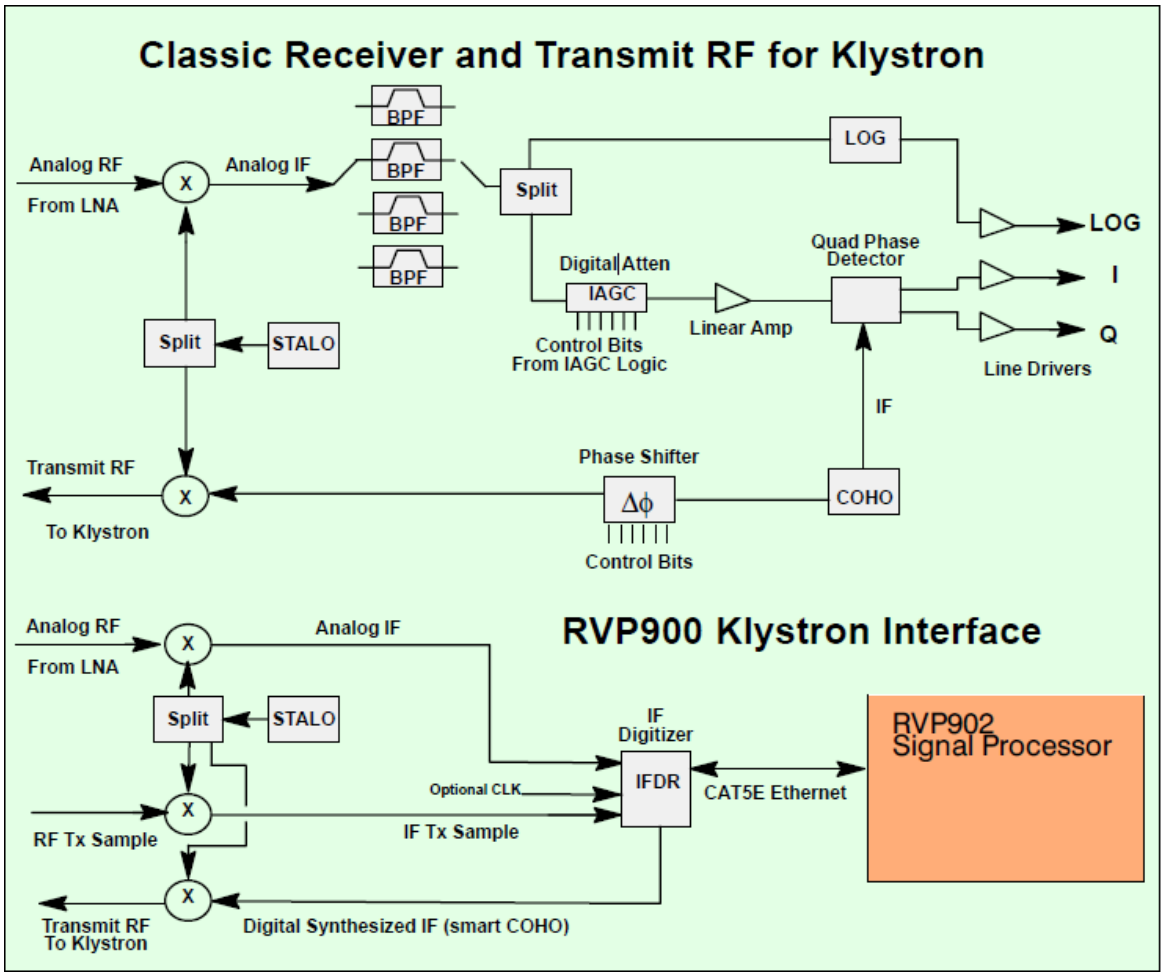


Figure 10 Analog versus Digital Receiver for Klystron Systems

Analog Receiver - Klystron

A typical analog receiver in a klystron system has a component arrangement 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, and so on.

RVP Digital Receiver - Klystron

The RVP900 transmitter function 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 support advanced algorithms, for example, range/velocity ambiguity resolution or pulse compression for low power TWT systems.

The following figure shows a Klystron system in which 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. No additional hardware is required.

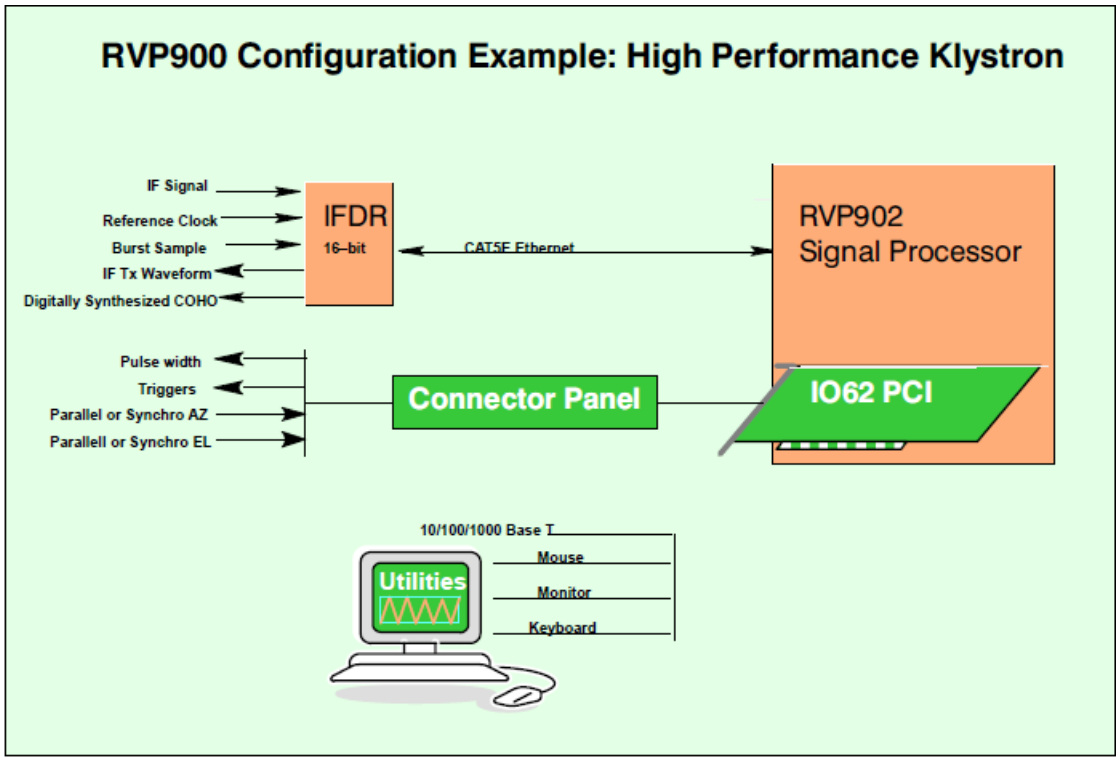


Figure 11 Klystron System with Digital Tx

Tx waveform generation requires an optional software license.

3.2 RVP902 Signal Processing Computer

The RVP902 signal processing computer hosts the Linux operating system and provides the computing resources for processing the `I/Q` values that are generated by the RVP901 IFDR.

While available, the system does not require that a keyboard, mouse, or monitor be connected, which is typically the case at an unattended site.

The RVP902 computer also hosts RVP900 utilities, for test, configuration, control, and monitoring.

More Information

- RVP902 Signal Processing Computer Specifications (page 329)

3.2.1 LAN Connection for Data Transfer or Parallel Processing

For external communication, RVP902 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.

The Ethernet is used to transfer moment results (`Z`, `T`, `V`, and `W`) to third-party application host computers such as a product generator.

RVP901 communicates over 100 Base T or Gigabit Etherent using UDP packets to send the `I` and `Q`values to RVP902 and can broadcast them - allowing for archiving or parallel processing.

The digital `I` and `Q` data produced by the IFDR 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.

3.2.2 Open Hardware and Software Design

RVP is an open-architecture radar signal processor that uses Gigabit Ethernet to interface to the IFDR, which samples the data.

RVP software runs under standard CentOS Linux, and is developed and maintained using standard GNU tools (for example, **`gcc`**, **`gdb`**, **`make`**).

Using public APIs, researchers and OEM manufacturers can modify or replace existing algorithms, or write their own software using the RVP900 software as a foundation.

Upgrades

To support software upgrades, RVP902 can flash RVP901 software.

The RDA version can be updated in the field with minimal risk. RVP902 software provides a configuration interface.

For more information, see *IRIS and RDA Software Installation Guide*.

3.2.3 RVP902 Socket Interface

RVP902 is configured to listen on a network port.

It is ready to interface to a host computer through the network using the `DspExport` program.

When IRIS/Radar is installed on the same RVP902 computer, it is pre-configured to communicate with RVP902 processes through the native interface, bypassing the socket interface.

RVP902 includes built-in utilities such as **Setup**, **dspx**, and **Ascope**. See *IRIS and RDA Utilities Guide*.

Because RVP902 can only have one program controlling it at a time, using a local program such as **dspx** blocks network access.

3.2.4 RVP902 Socket Protocol

The socket interface supports the commands in 8. Host Computer Commands (page 231).

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, consisting 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. The`Ack|` string is in the form of name/value pairs, for example:

```
Ack|CanCompress=1,Model=RVP900,Version=7.32
```

Your program can evaluate, or ignore, these keywords. `CanCompress=1` indicates that the `DspExport` computer supports compression. The host computer can then choose to use compression. 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 the following 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`**)

Table 6 Socket Protocol Commands

Command	Example
Read (**`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|`, meaning there can be no partial reads.
Write (**`WRIT`**)	Example: `WRIT|<data>`, where `<data>` is some binary data. This data is written to RVP900. The data size should be even.
Read Status (**`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 `Ack|0`, `Ack|1`, or `Nak`. This is equivalent to the `dspr_status()` call in the `dsp` library.

Command	Example
Set Information (**INFO**)	Example: `INFO|ByteOrder=LittleEndian,WillCompress=1,Version=7.32`. This command can be used to inform RVP902 `DspExport` about the host computer. Available options 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 data from normal **READ** commands is compressed. If the data 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.
Read Data Available (**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**and **STAT** commands.
Open the Connection for I and 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.
Read **Zcal** Information (**RCAL**)	Example: `ZCAL` means read the `dsp_refl_cal` structure from RVP900 and send it back in an ASCII name=value pair format. This is the structure configured by **Zauto** and **Zcal**. The configuration is served to all clients using RVP900.
Reset Kernel FIFOs (**RKFF**)	Example: `RKFF|2|` means reset the kernel FIFOs on the RVP900. The argument specifies which direction FIFOs to reset.
Read **Setup** Information (**SETU**)	Example: `SETU` means read the `dsp_manual_setup` structure from RVP900 and send it back in an ASCII `name=value` pair format. This is the structure configured in the RVP section of the **Setup** utility. The configuration is served to all clients using RVP900.
Write **Zcal** Information (**WCAL**)	Example: `WCAL|...` writes the `dsp_refl_cal` structure to RVP900 for saving.

3.2.5 Public API

To support writing your own signal processing algorithms for RVP900, the RVP900 software architecture allows you to statically link plug-in modules to the running code. The following table shows how the API supports adding software extensions to the RVP900 framework to modify some signal processing stages.

Table 7 API Support for Modifying Signal Processing

Processing Stage	API Support
Tx/Rx waveform synthesis and matched filter generation	Define transmit waveforms from pulse-to-pulse, along with the corresponding FIR coefficients that extract (`I`,`Q`) from the Tx waveform. Allows users to experiment with arbitrary waveforms for pulse compression and frequency agility.
Time series and spectra processing from (`I`,`Q`)	Modify default time series and spectra data, for example, to perform averaging or windowing in a different way.
Parameter generation from (`I`,`Q`)	• Redefine how standard parameters (`dBZ`, `Velocity`, `Width`, `PHIDP`, and so on.) are computed from the incoming (`I`,`Q`) time series. • Create new parameter types.

The standard scientific algorithms are not public. Many RVP900 library routines are also documented and can be called by user code, but the source to these routines is not generally released. Development tools must be purchased separately.

More Information

› Customizing RVP Software (page 367)

3.3 RVP900 Weather Signal Processing

Radar signal processor (RVP) software triggers radar measurement by producing the trigger signal for the transmitter using the intermediate frequency digital receiver (IFDR) unit in the receiver.

After the IFDR unit has digitized the received echo signal into samples (`I` and `Q` data), RVP processes the data in the radar server computer using computations such as:

- Converting the received signal amplitude into calibrated radar reflectivity values.
- Doppler processing to filter out ground clutter and compute radial velocities.
- Polarimetric processing to classify the measured hydrometers and to apply attenuation correction.

The end product of the RVP process is a radar ray, where selected radar data from a certain short time interval is stored as a function of range.

The following figure shows a block diagram of moment processing.

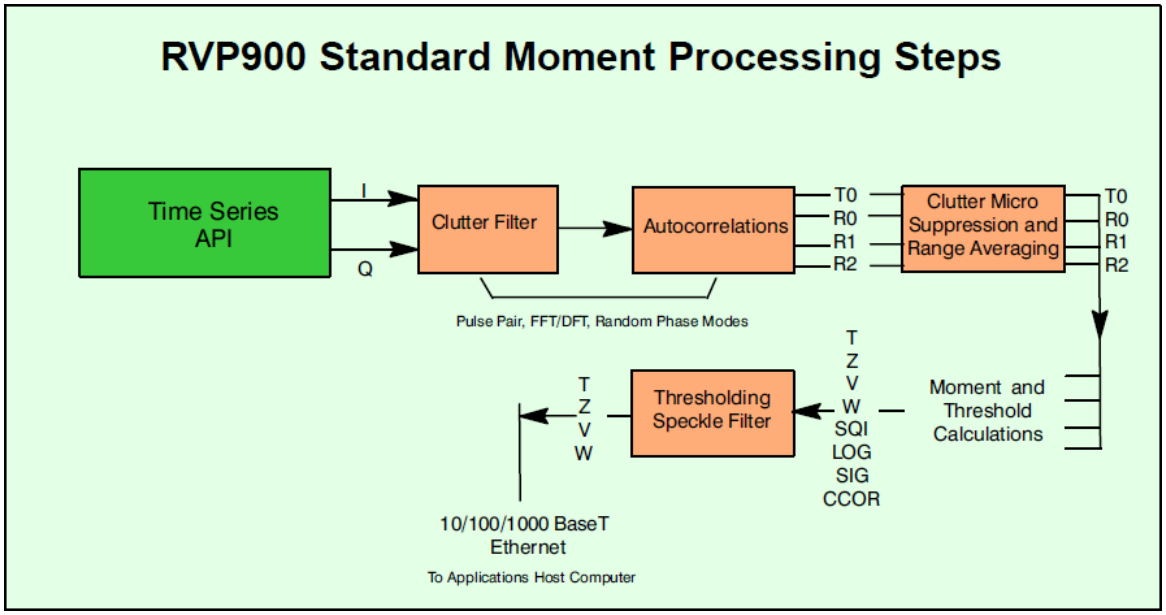


Figure 12 `I/Q` Processing for Weather Moment Extraction

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

Parameters configure signal processing, such as pulse repetition frequency, range resolution, and Doppler filter parameters. You can select these either directly when running RVP as a standalone or through the IRIS software when IRIS controls the RVP process during automatic weather radar measurements.

Applications

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 and microburst detection
- Gust front detection
- Particle identification
- Target detection and tracking
- General weather monitoring

Processing Modes

RVP900 has several major processing modes options for obtaining basic moments:

- Pulse pair mode time domain processing
- DFT/FFT mode frequency domain processing
- Random phase mode for second trip echo filtering
- Polarization mode processing

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, and so on.) This is too restrictive on the scan strategy for a modern Doppler radar since this means, for example, that a 1° azimuth radial must be constructed from exactly 64 input `I`/`Q` values. 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.

More Information

- RVP Algorithm Overview (page 167)

3.3.1 Burst Pulse Analysis for Amplitude, Frequency, and Phase

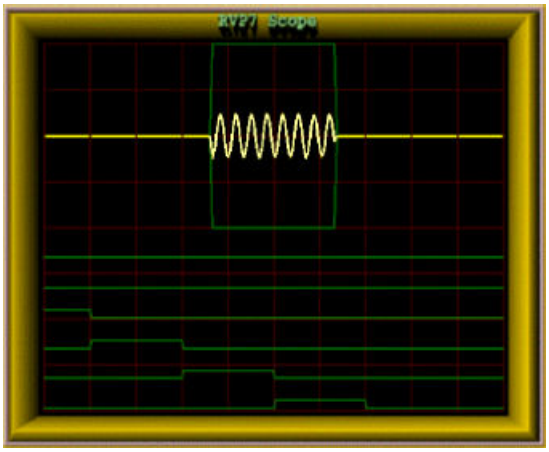


Figure 13 Burst Pulse Analysis for Amplitude/Frequency/Phase

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. A warning is issued if the burst pulse amplitude falls below a threshold value.

1. The burst pulse data stream is first analyzed by an adaptive algorithm to locate the burst pulse power envelope (for example, 0.8 µsec).
   The algorithm 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.
2. The power-weighted phase of the burst pulse and the total burst pulse power is computed.
3. 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° RMS, as measured on nearby clutter targets. For Klystron systems, the phase locking is better than 0.1° 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.

3.3.2 Time (Azimuth) Averaging

Autocorrelations are based on input `I` and `Q` values over a selectable number of pulses between 8 ... 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° 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.

3.3.3 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, RVP900 samples 32 digital input `TAG` lines from the RVP data structure: 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. 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. Angle values may be binary or BCD.

3.3.4 Range Averaging and Clutter Microsuppression

Range averaging can improve the accuracy of the reflectivity measurements.

When this is done, autocorrelations from consecutive range bins are averaged, and the result is treated as a single bin. This means that the host computer must process fewer range bins.

Range averaging of the autocorrelations may be performed over 2 ...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.

More Information

- Performing Range Averaging and Clutter Microsuppression (page 198)

3.3.5 Moment Extraction

The autocorrelations are the basis for 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`, but without ground clutter

These standard parameters are output to the host application.

3.3.6 Threshold Processing

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`.

3.3.7 Speckle Filter

A speckle filter is a final pass over each output ray, in which isolated, single bins of velocity, width, or intensity are removed.

There are two speckle removers in 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.

This eliminates single pixel speckles, which allows the thresholds to be reduced for greater sensitivity with fewer false alarms (speckles).

More Information

‣ Speckle Filter Processing (page 208)

3.3.8 Velocity Unfolding

The RVP900 processor can unfold mean velocity measurements based on a dual PRF algorithm.

In this technique, 2 radar PRFs are used for alternate `N`-pulse processing intervals. The internal trigger generator automatically produces the correct dual-PRF trigger. An external trigger can also be applied.

For external triggers, the `ENDRAY_` output line indicates when to switch rates. RVP900 measures the PRF to determine which rate (high or low) was present on a given processing interval. It then unfolds based on a 2:3, 3:4, or 4:5 frequency ratio.

Table 8 Example Unambiguous Velocity Intervals

PRF1	PRF2	Unambiguous Range (km)	Unambiguous Velocity (m/s)			Unfolding
			3 cm Wavelength	5 cm Wavelength	10 cm Wavelength	
500	*	300	3.75	6.25	12.50	None
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	

3.3.9 Autocorrelations

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

Table 9 Autocorrelation Mode

Mode	Description
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 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.

Mode	Description
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.

3.3.9.1 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.

3.3.9.2 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.

3.3.9.3 Random Phase Processing for Second Trip Echo

Second trip echoes can be a serious problem for applications that operate 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[1)] separates the first and second trip echoes so that:

- In most 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 wide dynamic range of weather echoes, this power limit is sometimes exceeded.

1) *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.*

The technique requires that the phase of each pulse be random. Digital phase correction is applied in the processor for the first and second trips. The adaptive filter 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. RVP900 provides an 8-bit RS422 output for the phase shifter.

3.3.9.4 Polarization Mode Processing

Polarization processing uses a time domain autocorrelation approach to calculate the parameters of the polarization co-variance matrix, that is, **ZDR**,**LDR**, **PhiDP**, **RhoHV**, **PhiDP** (**Kdp**), and so on.

The standard moments **T**, **V**, **Z**, and **W** are also calculated.

The available parameters and the algorithms 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 calibration of the **ZDR** and **LDR** offsets. The use of a clutter filter for the polarization variables can sometimes bias the derived parameters. Because of this, you must decide whether to use filtered or unfiltered time series.

3.3.10 Output Data

RVP900 output data for standard moment calculations include 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. 16-bit format is preferred for most applications. Time series and DFT are always 16-bit formats.

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

More Information

- Host Computer Command Overview (page 231)

3.4 Radar Control Functions

Table 10 Radar Control Functions

Function	Description
Trigger generation	Up to 12 programmable triggers.

Function	Description
Pulse width control	Four states controlled by 4 bits.
Angle/data synchronization	Used to collect data at precise azimuth intervals (for example, every 0.5°, 1°, 1.5°) based on the AZ/EL angle inputs.
Phase shifter	Used to control the phase on legacy Klystron systems. New or upgraded Klystron or TWT systems can use the RVP900/Tx card to provide accurate phase shifting.
Z_{dr} switch control	For horizontal/vertical or other polarization switching scheme.
AFC digital output	Based on the burst pulse analysis for magnetron systems

Pulse Width and Trigger Control

Four TTL output lines can be programmed to drive external relays that control the transmitter pulse width.

The internal trigger generator drives 8 separate lines - each can be programmed to produce a desired waveform. The waveforms are stored in RAM, and can be modified interactively by the 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. The processor measures the trigger period between pulses, so that user software can monitor it as needed.

RVP900 can also operate from an external user-supplied trigger.

Trigger Blanking

In trigger blanking, one or more (selectable) of the transmit triggers can be inhibited. Using one of the following techniques, trigger blanking is used to avoid interference with other electronic equipment and to protect nearby personnel from radiation hazard:

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

3.5 Configuration and Monitoring

The radar server computer (RVP902) stores the setup configuration on a disk.

You can access setup information locally or remotely, over the network. For multiple radar networks, you can manage the configuration centrally by copying tested master configuration files to the network radars.

Boot-up Process

During the boot process, 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 basic memory self-tests before starting.

Monitoring

The platform monitors:

- voltage
- temperature
- locked conditions of the PLLs on the FPGA to verify that the analog and digital clocks are in good health.

IP Address Configuration

The RVP902 IP address is user-configurable.

If you forget the programmed IP address, you can use the outer most hole (SW2) in the finned side of the enclosure to reset the IP address to 10.0.1.254.

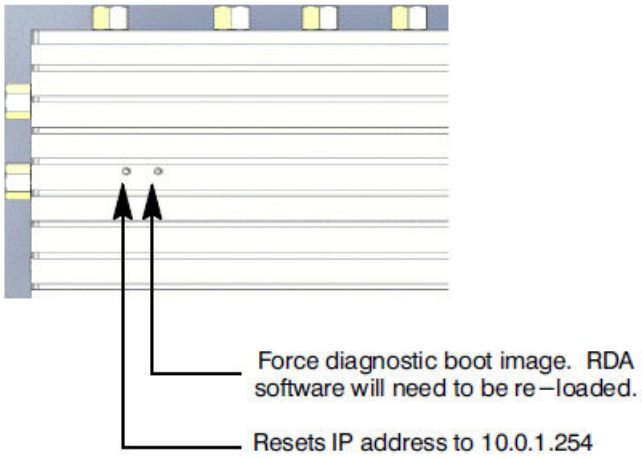


Figure 14 Reset IP Address

3.6 Digital AFC (DAFC)

The optional Digital AFC Module (DAFC) is a small, self-contained circuit board that passively listens to RVP900 serial uplink transmissions. Its generates a set of digital AFC control lines that could be applied, for example, to a custom STALO frequency synthesizer.

Vaisala recommends using the DAFC board in systems that require an AFC because it offers advantages over other methods of frequency control:

- The stability of a synthesized STALO can be made much greater than that of a tunable cavity oscillator. 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 be difficult.
- 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. The DAFC board can supply up to 24 output control lines.

More Information

- Automatic Frequency Control (AFC) (page 174)
- Installing DAFC (page 76)
- Mb — Burst Pulse and AFC (page 95)

3.7 Expansion Panels

Expansion panels allow the signal processor to interface with other radar sub-systems. RVP901 is designed with generic I/O capability to interface with these expansion panels.

3.8 Utilities and Applications

RVP900 includes applications and utilities for the calibration, alignment, and configuration. These can be run locally on RVP900 or over the network from a central maintenance facility.

For more information, see *IRIS and RDA Utilities Guide*.

Table 11 RVP Utilities

Utility	Description
Ascope	Used to control the signal processor manually and to display moments, times series, and Doppler spectra. Includes a signal simulator able to produce first and second trip targets. Can record and playback of time series and moments.
dspx	An ASCII, text-based program for accessing and controlling the signal processor, including accessing local setup menus
DspExport	Exports RVP900 to another workstation over the network. This allows utilities on a remote network to run locally, instead of exporting the utility display window over the network. 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.
Setup	Interactive GUI for creating and editing RVP900 configuration files
Zauto	Calibration utility for use with a test signal generator

Table 12 Radar Application Software

Application	Description
IRIS/Radar	The package provides complete local and remote control/monitoring, data processing, and communication for a radar system. • Runs on the same or separate PC. • Interfaces to RVP900 internally or by 100 BaseT Ethernet. • Controls RVP900 and RCP8 radar/antenna control processor.
IRIS/Analysis	Functions as a radar product generator (RPG) to provide outputs such as CAPPI, rain accumulations, echo tops, automatic warning and tracking, and so on. Optional software packages are provided for certain applications: wind shear and microburst detection, hydrometeorology with raingage calibration and subcatchments, composite, dual Doppler, and 3D Display. Runs on a separate PC, often at a central site. One IRIS/Analysis can support up to 20 radar systems.
IRIS/Focus	Provides IRIS displays to network users on standard PC (Windows or Linux) running standard browsers.
IRIS/Display	Displays products sent to it and, with password authorization, can serve as a remote control and monitoring site for networked radar systems. Includes features such as looping, cross-section, track, local warning, and annotation. IRIS/Analysis and IRIS/Radar include the capabilities of IRIS/Display. Any IRIS system can display products

3.9 Network Architecture

RVP900 can run on a network to support remote control and monitoring, subject to the user's security restrictions.

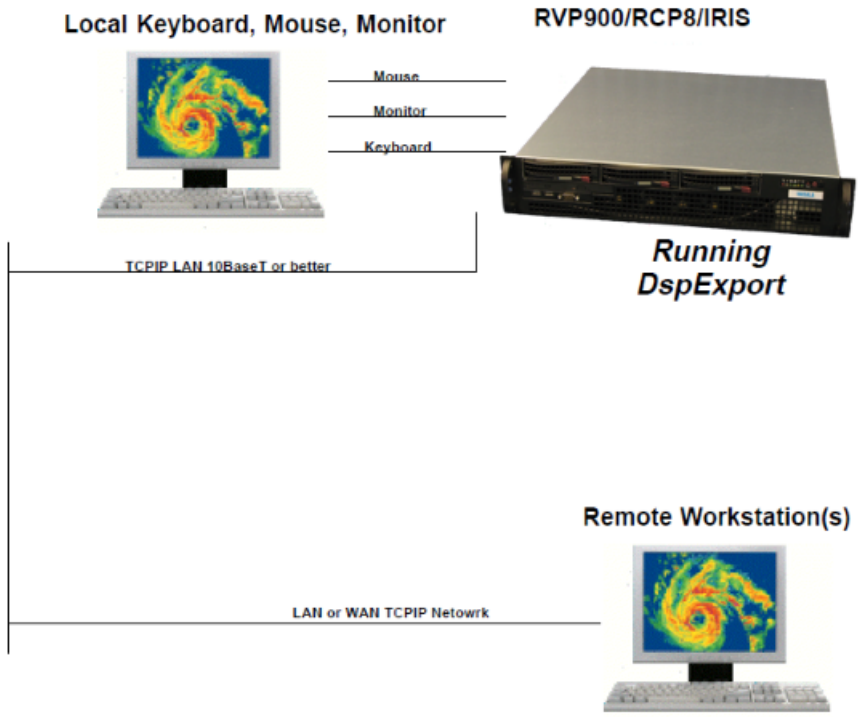


Figure 15 Network Architecture

The `dsp lib` runs locally on RVP900. The **DspExport** utility exports the library over the network through a TCP/IP socket. Typically the controlling application is on the same computer, but **DspExport** can also export to a remote host radar server computer on the network. If this workstation runs IRIS software, at least a 10BaseT connection is recommended.

A remote workstation can also use **DspExport** for configuration, monitoring, and diagnostic testing.

4. RVP Hardware

4.1 Hardware System Overview

RVP900 hardware installation includes mechanical installation and siting, electrical specifications of the interface signals, system-level considerations, and the standard connector panel.

The major modules supplied with RVP900 are:

- RVP901 IF Digital Receiver (IFDR), which is typically mounted in the radar receiver cabinet.
- RVP902 main chassis, which is usually mounted in a 19″ EIA rack

Much of the RVP900 I/O is configured through software. Since there is no custom wiring, internal jumpers, or oscillators, it is easy to insert spare modules. See E. Serial Status Formats (page 413).

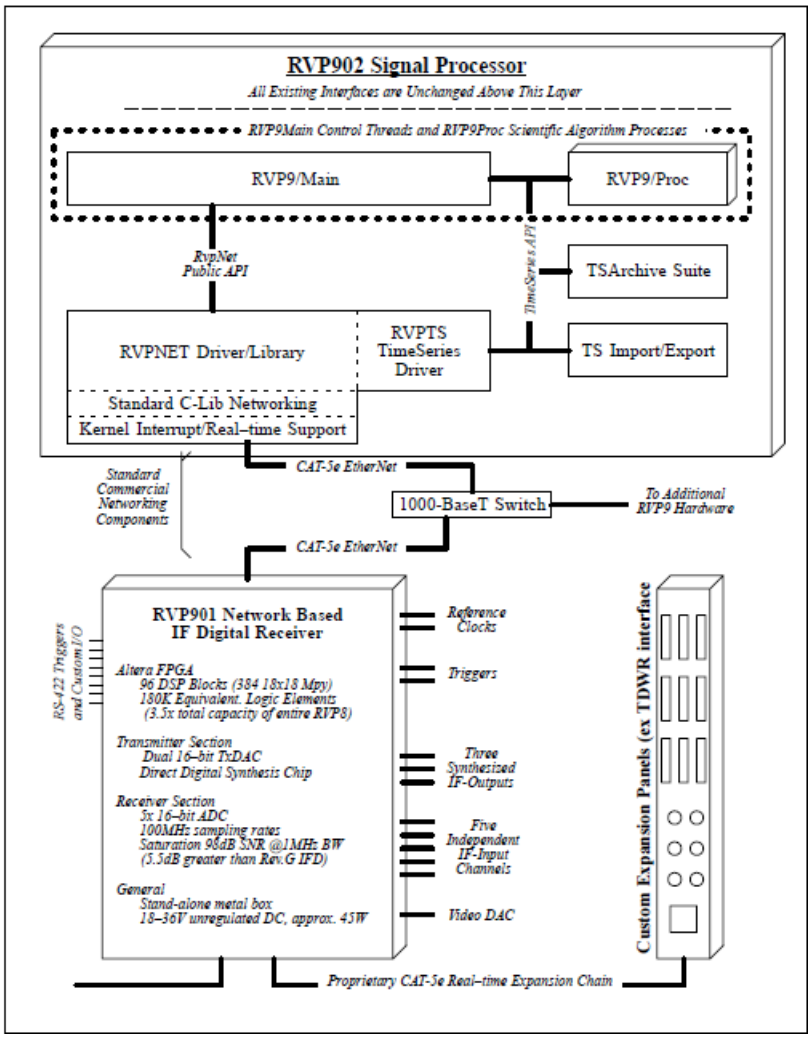


Figure 16 RVP900 System Concept

Vaisala usually supplies turn-key systems. Some OEM customers purchase just RVP901 and integrate it themselves while customizing the processor and software.

4.2 RVP901 IFDR Hardware

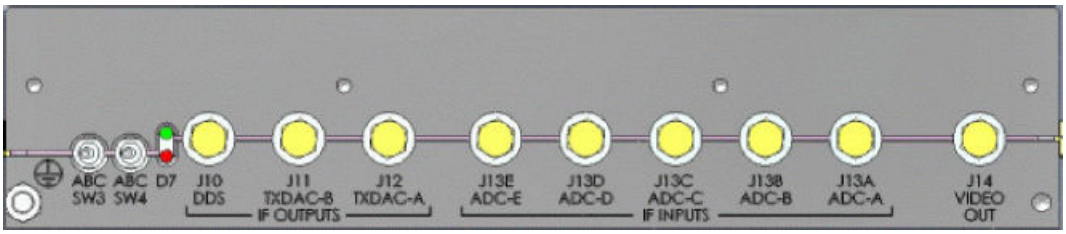


Figure 17 RVP901 IFDR

The RVP901 IFDR module is typically placed where a traditional LOG receiver would be installed.

The module can be mounted on edge with the 26.8 cm x 4.8 cm (10.5 in x .9 in) surface flush to the back of the receiver cabinet with 17.6 cm (6.9 in) protrusion into the cabinet.

The RVP901 IFDR is packaged in a tight metal enclosure for maximum noise immunity.

More Information

‣ RVP901 IF Digital Receiver (page 23)

4.2.1 Upgrading IF Receiver with RVP901 IFDR

The RVP901 IFDR replaces the IF receiver components in a traditional analog receiver system:

- AFC circuit
- AGC or IAGC circuit
- COHO (on magnetron systems)
- Band pass filters
- Line drivers for base band video
- LOG receiver
- `QuadpPhase` detector

Removing old components can be time consuming. Many customers choose to bypass them and leave them in place.

In some cases, other receiver modifications are required to match RVP901 IFDR signal input specifications. For example, IF attenuators or an IF amplifier may be required.

Read the document and red-line your system schematics to reflect any changes to the receiver.

If you are upgrading an older system, consider purchasing a new STALO to improve Doppler performance.

4.2.2 RVP901 IFDR Power, Size, and Mounting Considerations

Table 13 IFDR Installation Considerations

Consideration	Description
Communication	The IFDR communicates with a host computer through Ethernet. The 10/100/1000 Mbit physical interface supports supports 100 Base-T full duplex and Gigabit Ethernet options. 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. Vaisala recommends the UDP packet sizes be set to 8192 on the host computer. To comply with industry standards, use 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.
Cooling	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. One side of the RVP901 IFDR serves as a heat sink. The hotter chips mounted on the printed circuit board are bonded to the heat sink. Position the RVP901 IFDR so that a minimum of 20 cubic feet per minute of air can freely convect around it. If you order RVP901 without fans, you must provide provide > 20 cubic feet per minute of airflow across the RVP901 heat sink. The ambient air temperature should be within a range of -40 ... +50 °C (-40 ... 122 °F) with DC supply power option and - 40 °C ... +45 °C (-40 °F ... 113°F) with AC supply option.
Filters	Reserve mounting space for the external, analog, anti-alias filters. The 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.
Internal regulator	Internal regulator is an isolated DC-DC converter taking a 12 ... 36V input and regulating it down to provide supply the voltages needed by internal circuits. A ferrite choke, around the supply wires, near the terminal strip is also recommended.
Mounting	The IFDR 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, was installed.

Consideration	Description
Power module	The power module is an auto-ranging AC-DC converter generating 24V to drive the RVP901 and external fans. The RVP901 IFDR is delivered with the power supply module attached to the IFDR mechanical housing. You can remove the power supply module and mount it 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 ... 240 VAC at 47 ... 63 Hz or 18 ... 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, 3 fan assemblies are available. These fan options allow DC voltages from 12 ... 18 VDC and 18 ... 28 VDC. The IFDR has an internal regulator to supply the digital circuits the voltages need. The internal regulated power modules are sized to provide several Watts more than is required by RVP901. A ferrite choke around the supply wires, near the terminal strip, is also recommended. For information on the power harness needed to supply power to the RVP901 and fans, see 9.3 RVP901 IFDR Specifications (page 326) and B. RVP901 IFDR Technical Drawings (page 365).

4.2.3 RVP901 IFDR I/O Connectors

Table 14 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

J-ID	Label	Type	Description
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

4.2.3.1 Generic I/O Interconnect Breakout Cable

All communication to the main RVP902 server chassis is over an Ethernet cable. Also, two general purpose I/O ports on J3 and J6 provide additional system control and monitoring.

The major volume of data is the **I** and **Q** samples and some status indicators.

A generic interconnect cable is available to breakout each of the 51-pin micro **D** connectors on the IFDR to a standard 25-pin and 37-pin female **D** connector. A similar 62-pin cable is used with IO62 card.

The following table shows the cable wiring and internal signal names. The cable provides 10 RS-422 signals, 10 TTL signals, three differential analog input A/D signals, and +5 V and -5 V auxiliary power from the IFDR. Standard cable length is 1 meter.

J3 and J6 are identical ports each providing the following signals.

Table 15 Generic Interconnect cable for IFDR 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

D Connectors					Index	
51-Pin	25-Pin	37- Pin	Signal Name	Softplane Signal	J3	J6
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		

4.2.4 RVP901 Miscellaneous Discrete and Analog I/O Connectors

Table 16 Discrete and Analog I/O

Function	Description
Misc	• 2 identical 51-pin MicroD connectors to support miscellaneous I/O • Includes D/A, A/D, discrete inputs and outputs (TTL, RS-485/422, and so on.). See the following table. • 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	• 6 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 17 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]

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

4.2.5 RVP901 IFDR Status Indicators

Two LEDs provide status information for the RVP901 IFDR and status of the communication links to RVP902.

Table 18 RVP901 IFDR LED Indicators

Red (Uplink)	Green (Ready)	Description
Blink	Blink	Reset sequence (power-up or from uplink). When in boot or diagnostic mode the red and green LEDs have a slow blink.
On	On	Normal Operation (two-way communication with RVP901 IFDR and computer are both okay). When the weather radar application is booted the green light is solid and the red light blinks to indicate that it is ready to communicate with RVP902 software but no services have been started. The red light is solid when the `rvp900` service on the RVP902 server has connected and configured the unit for normal operation.

4.2.6 RVP901 IFDR Inputs

All RVP901 IFDR 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.

You can use a fixed attenuator or IF amplifier to adjust the signal level to be in this range. The maximum signal level is 20 dBm.

Table 19 RVP901 IFDR Input Options

Input	Description
IF signals	Four A/D converters are 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 wide dynamic range (WDR) applications for single and dual-polarization radars.
IF burst pulse sample for magnetron IF COHO for Klystron	Received over ADC-E.
Optional reference clock	Used for system synchronization in Klystron systems. The COHO can be input. Magnetron systems do not require this signal.
Trigger input or output	Available on two 5 V 50Ω driver/receivers.

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

RVP900 provides AFC support for tuning the STALO of a magnetron system. Alternatively, the magnetron can be tuned by a motorized tuning circuit controlled by RVP900.

4.2.7 Defining RVP901 IFDR Input A/D Saturation Levels

Two analog signals must be supplied to the RVP901 IFDR:

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

Both inputs are on SMA connectors.

The IF signal should be driven by the front-end mixer/LNA/IF-Amp. components, similar to how a LOG receiver would be installed. The magnetron burst pulse, or klystron COHO reference, is also derived in the same manner as a traditional analog receiver.

IF Input and Burst Input

The A/D input saturation level for the IF input and burst input is +8 dBm.

In most 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 ... 1.0 dB. Thus, the input saturation level is +8.5 dBm ... +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 ... +4 dBm. This is important for making a precise phase measurement on each pulse.

See A.16 Burst Pulse Alignment (page 350).

IF Receiver Input

For the IF receiver input, it is permissible (in fact, desirable) to occasionally exceed the A/D input saturation level at the strongest targets, however inputs must not exceed 20 dBm. RVP900 uses a statistical linearization algorithm to derive correct power levels from targets that are as much as 6 dB above saturation.

Establish the IF signal level by weak-signal and noise considerations, rather than by working backwards from the saturation level.

4.2.8 Configuring the RVP901 IFDR Clock Subsystem

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

Table 20 Clock Generator

Source	Description
Master Clock Source	Clock reference can be provided by a 20 MHz on-board oscillator. An external clock reference may also be provided to RVP901 through CLK-IN (J7). The master clock source is software-configurable between the on-board 20 MHz reference or an external source. The external clock option allows the RVP901 IFDR to be phase locked to a standard system reference; however, the external clock is not a requirement. Recommended operating frequency is 7.5 ... 100 MHz and amplitudes -20 ... 6 dBm. The input clock has a 50 Ω input impedence. 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 ... +85°C (-40°F ... +176°F). Its jitter is sub–picosecond.

Source	Description
IF Clock Frequency	The sampling clock frequency is fully programmable 50 MHz ... 100 MHz with microhertz (µHz) resolution. You can choose the clock frequency independently of the original reference clock frequency that produces it; they do not have to be small integer multiples of each other. See 4.2.8.1 Choosing A/D Sample Rate or Tx Synthesis Rate (page 67) The RVP901 IFDR sampling clock is derived from the master clock source. The architecture minimizes jitter, while allowing full flexibility in selecting sampling frequencies 50 MHz ... 100 MHz. The output clock runs at the same frequency as the sampling clock.
Clock Jitter	IF clock jitter is sub-picosecond allowing the system to maximize the benefits of the 16-bit A/D converters.

When RVP900 is used in a klystron system, or a synchronous radar, the radar COHO is supplied to the RVP901 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 RVP901 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 3.1.2 RVP901 IFDR IF to I and Q Processing (page 25)). 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 concerns that may occur when RVP900 is used in the above manner within a synchronous radar system. Both concerns are the result of the RVP901 IFDR sampling clock being asynchronous with the radar system clock.

Table 21 Clock Generator Concerns in a Synchronous Radar System

Concern	Description
RVP900 Generates the Radar Trigger	The trigger signals supplied by RVP900 are synchronous with the RVP901 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 RVP901 IFDR. Since the RVP901 IFDR sampling clock is asynchronous with the radar clocks, 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 RVP900 to reject ground clutter and to process multi-mode return signals. For this reason, a synchronous radar system, which is triggered using 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.

Concern	Description
RVP900 Receives the Existing Radar Trigger	When an external trigger is supplied to 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 RVP901 IFDR selectable sampling clock. The result is that 27.8 nanoseconds of jitter is introduced in the placement of RVP900 range bins relative to the transmitted pulse. The effect of this synchronization jitter is that targets appear fluctuate 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 field measurements, we have found that sub-clutter visibility of a µsec 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 ... 60 dB should be attainable.

The solution to these concerns is to provide a way for the RVP901 IFDR internal sampling clock to be phase locked to the radar system. If RVP900 provides the radar triggers, then those triggers become synchronous with the radar COHO. If 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 RVP901 IFDR can lock 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.

4.2.8.1 Choosing A/D Sample Rate or 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 ... 100 MHz.

The setup questions in the **Mc** menu 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. See 5.2.2 Mc — Top-level Configuration (page 104).

The sample clock frequency 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.

Table 22 Sample Clock Frequency Considerations

Consideration	Description
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 4.2.10 IF Bandwidth and Dynamic Range (page 69). As the sampling frequency increases, the A/D quantization noise that is contained within a given Rx bandwidth decreases, which means that RVP900 becomes more quiet. The dynamic range varies linearly with sampling frequency. RVP900 has 3 dB greater dynamic range at 100 MHz versus 50 MHz clock.

Consideration	Description
Quantization of Trigger Timing and Range Bin Placement	Triggers generated by RVP900 are specified by their start time in microseconds, width in microseconds, and polarity. Triggers are always produced that are ±0.5 clocks of these ideal values. 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 Down-conversion 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 µsec), 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 gmode (affected by single/dual polarization, range bin spacing, and so on).
Null Frequency Bands in Synthesized Tx Output	When RVP900 generates a synthesized Tx waveform , the output frequency cannot be within the interval of frequencies 0.4 ... 0.6 **f**, where **f** is the sample clock rate.

Use the RVP900 setup menus to cross-check the above constraints. Note that the system installer must choose a sample clock frequency that achieves the best set of trade-offs at each radar site.

4.2.9 Configuring External Pre-Trigger Input

You may supply RVP900 with your own CMOS-level pre-trigger for installations in which adequate trigger control already exists.

1. In the *softplane.conf* file, configure the trigger input.
   The trigger input is provided on the RVP901 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 trigger input threshold is 1.3 V and can tolerate a 5 V maximum input when DC coupled. If AC coupled and 50 Ω terminated, it can take a -6 ... 8 dBm input signal. 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 RVP900 as the pretrigger point. The pulse width of the signal does not matter.
2. Configure the delay to **range0** in the TTY Setups.
   See 5.2.6 Mt<n> — Triggers for Pulsewidth n (page 117).
3. Synchronize the other trigger outputs 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.

4. In coherent systems, you can improve trigger jitter by phase locking the RVP901 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.

4.2.10 IF Bandwidth and Dynamic Range

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. 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 ... 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:

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

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:

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

The RVP900 receiver power monitor shows a filtered power level of approximately -87 dBm, when the FIR bandwidth is 1 MHz and the IFDR inputs are terminated in 50 Ω. See 6.7 Pr — Plot Receiver Waveforms (page 154)

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, RVP900 can 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. 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.

The following figure shows a calibration curve demonstrating this performance, for which the RVP900 digital bandwidth was set to 0.53 MHz. An external signal generator, with steps of 1 dB, was used to measure the compression and detection thresholds.

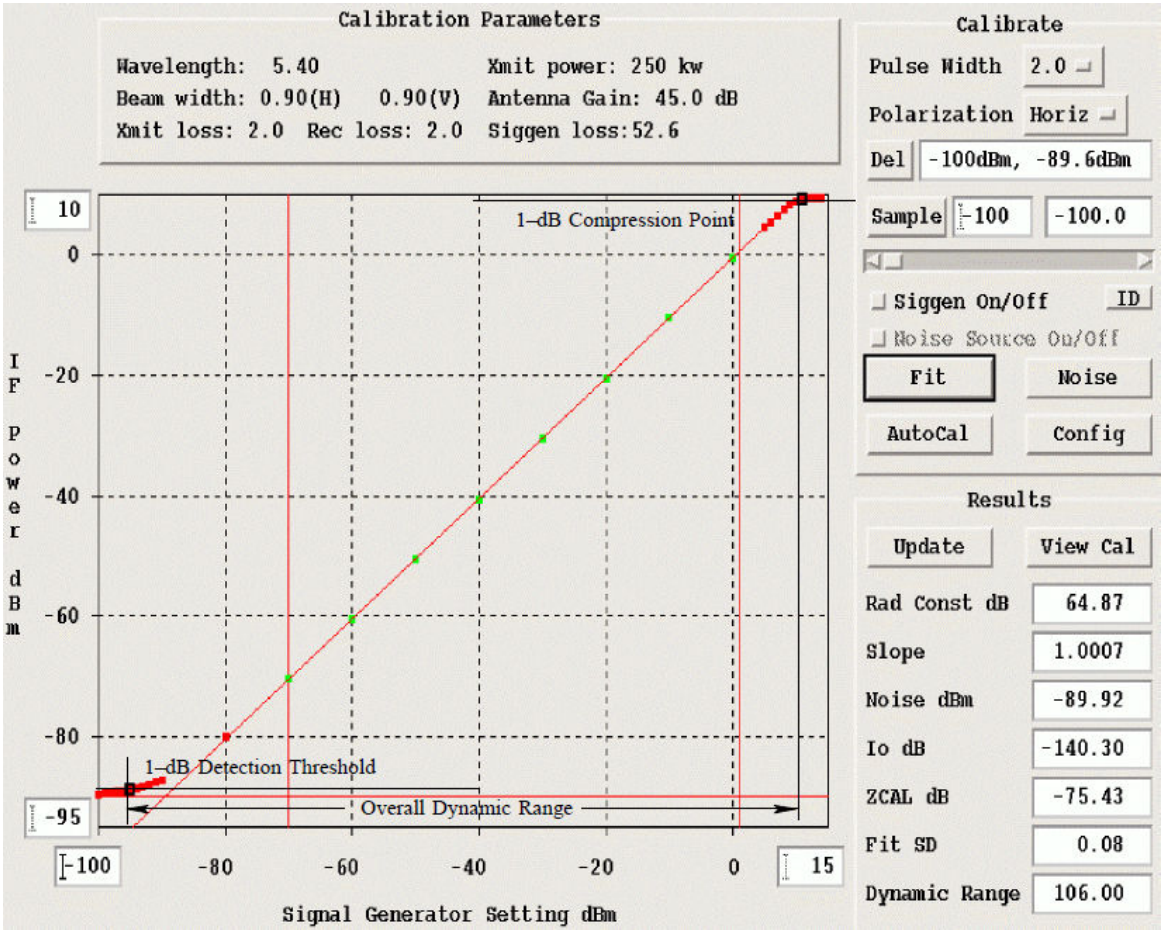


Figure 18 Calibration Plot for a Stand-Alone 16-Bit IFDR

4.2.11 Configuring IF Gain Based on System Performance

When considering IF gain and system performance for a complete radar receiver, we assume that an LNA/Mixer has already been selected that offers an appropriate balance between price and noise figure. The total RF/IF gain between the antenna waveguide and the IFDR must still be determined.

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 RVP901 IFDR so that the received echoes can be optimally processed.

When selecting the required gain, 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 same trade-off that is made in traditional multi-stage analog receiver systems that include a WDR (wide dynamic range) LOG receiver.

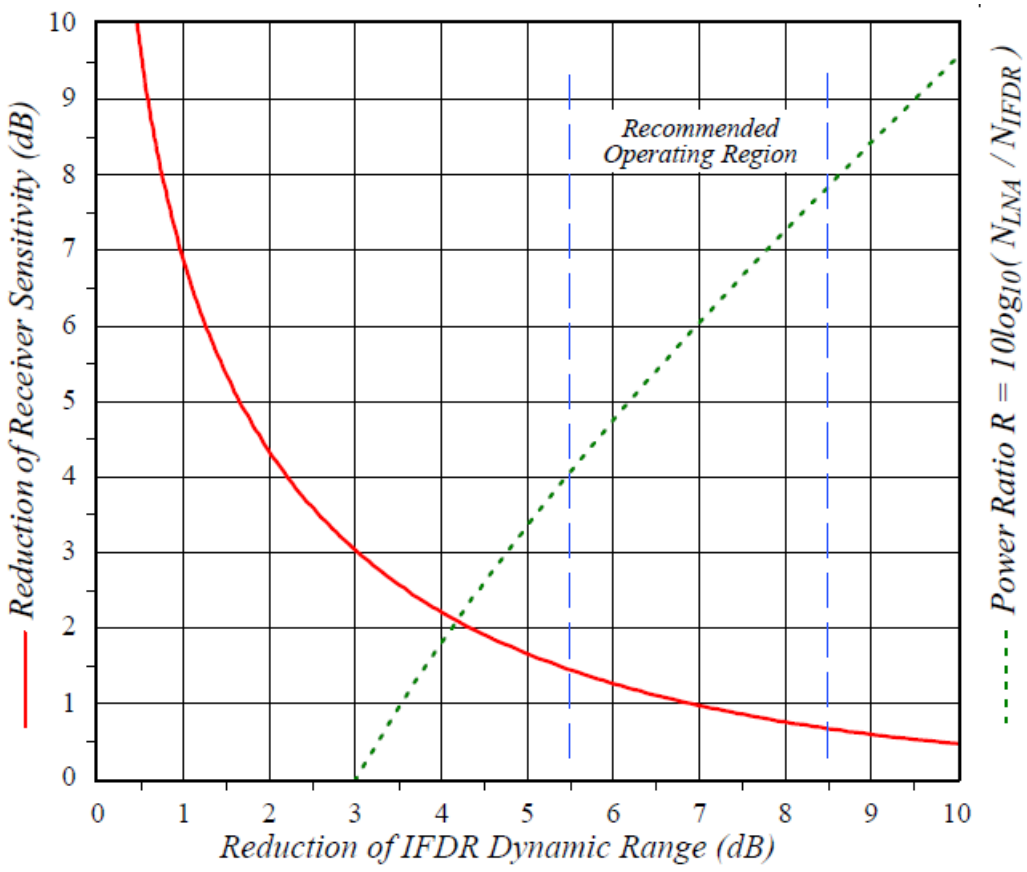


Figure 19 Trade-off Between Dynamic Range and Sensitivity

The solid red curve in the figure shows that the 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.
  N_{LNA} has its origins in the fundamental thermal noise of the receiving system.
- 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.

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:

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

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 quantities depend only on the ratio of the two powers. The two equations define a parametric relationship in the dimensionless variable $R = (N_{LNA} / N_{IFDR})$. The previous figure was created by sweeping the value of R from 1/9 to 9. The solid red curve shows the locus of (ΔDynamicRange, ΔSensitivity) points, and the dashed green curve shows R (in dB) as a function of ΔDynamicRange. For example, when the LNA noise power is equal to the IFD noise power, R is 1.0 (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 1.4 dB ... 0.65 dB. The attendant loss of dynamic range falls between 5.5 dB ... 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 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.

When designing your RF and IF components, remember that the final amplifier driving the RVP901 IFDR must be capable of driving up to, perhaps, +14 dBm, so that signals above saturation can be correctly measured.

For example, you can use the calculation shown in the previous figure 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, read the **ΔDynamicRange** from the red curve as 7 dB.
3. Read the required noise ratio **R** vertically on the dashed green curve as 6.1 dB.
4. Check that the RF/IF gain brings 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 does not depend on bandwidth, and therefore is correct for all pulse width/bandwidth combinations. The gain is:

   ```
   (–87 dBm/MHz+6 dB) – (–112 dBm/MHz) = 31 dB
   ```

5. Calculate the dynamic range for the complete system at 0.5 MHz bandwidth as 106 dB - 6 dB = 100 dB.
6. After assembling all of the RF and IF components, check whether you have the correct gain by verifying a 7 dB rise (independent of bandwidth) in RVP900 filtered power, when the IF- Input cable is connected and disconnected.

4.2.12 Configuring IF Gain Based on System Noise Figure

Every amplifier can be partially characterized by its gain **G** and noise figure **F**.

When computing the front-end RF/IF gain based on the system noise figure, gain is measured by injecting a test signal at the mid- power range of the amplifier and measuring the ratio of Output/Input power.

Noise figure 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 RVP900.

- To calculate the gain of the RVP901 IFDR, we apply a calibrated mid-power signal generator directly to its IF-Input, and use the **Pr** plot to print the measured power.
  For a wide range of analog input power levels, RVP900 reports the same measured digital power; therefore the overall analog/digital gain is 1.0 (0 dB).
  See 6.7 Pr — Plot Receiver Waveforms (page 154).
- To calculate the noise figure of the RVP901 IFDR, we set the receiver bandwidth to 1 MHz, terminate the IF-Input in 50 Ω, 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 RVP901 IFDR is 27 dB.
  See 6.6.2 Ps Subcommands (page 142).

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 the linear representations of the noise figures that were expressed in decibels:

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

If 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 RVP901 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:

$$G^1_{lin} = 1204(30.8dB)$$

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.

4.2.13 Choosing Intermediate Frequency

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 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 cannot distinguish 37 MHz from 35 MHz, then it cannot 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.

4.3 Installing RVP902 Main Chassis

WARNING! RVP902 has two power supplies.
Disconnect power from both supplies when powering down or servicing unit.

1. Mount the chassis in a nearby equipment rack on rack slides.
2. Using the shielded Ethernet cable to connect the IFDR to the main chassis.
3. Connect the power.
   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 Rev C and D have High Leakage Current 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.

More Information

- RVP902 Signal Processing Computer Specifications (page 329)

4.3.1 Powering-up RVP902

1. Power-up and boot RVP901.
2. Start or reset RVP902.
3. If RVP901 is not already powered-up and booted, the `rvp900` process cannot start.
   The host Linux PC goes through an automated boot process that ultimately starts the RVP902 application.

4. While RVP902 runs extensive internal diagnostics, for troubleshooting purposes, you can connect a display to view any error messages or monitor the startup log in */usr/iris_data/log/rvp9.log*.
   In most cases, there is no display connected to RVP902 to monitor the boot sequence.

4.4 Installing DAFC

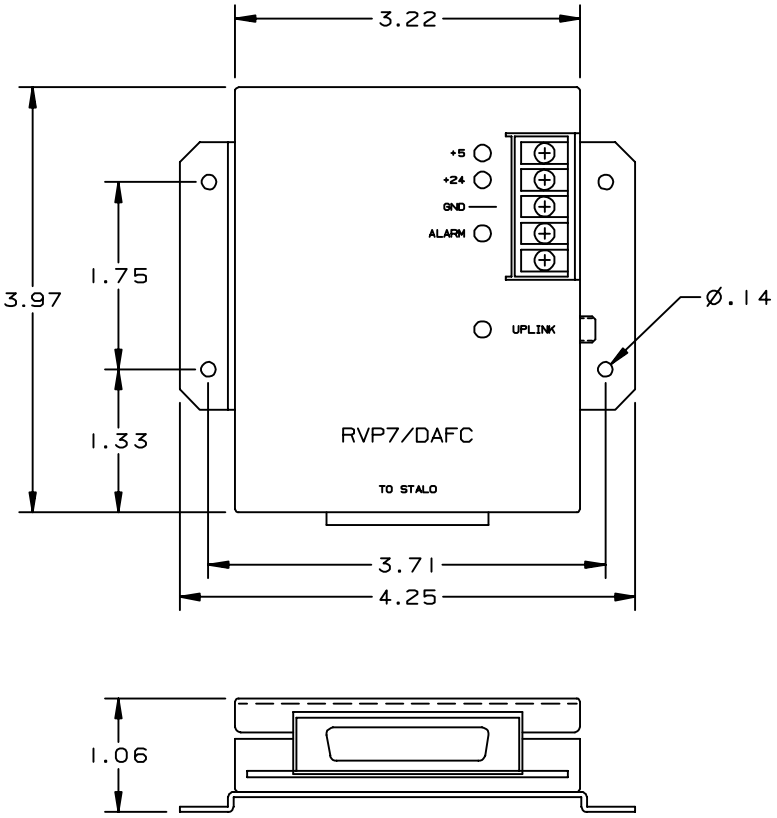


Figure 20 DAFC Module

The following figure shows assembly diagram of the board (3 in x 3.75 in). It can be installed in the radar system as a bare board, or packaged in a small metal enclosure.

The RVP900 uplink transmission comes from J15 port of the IFDR. A question in the **dspx** 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
```

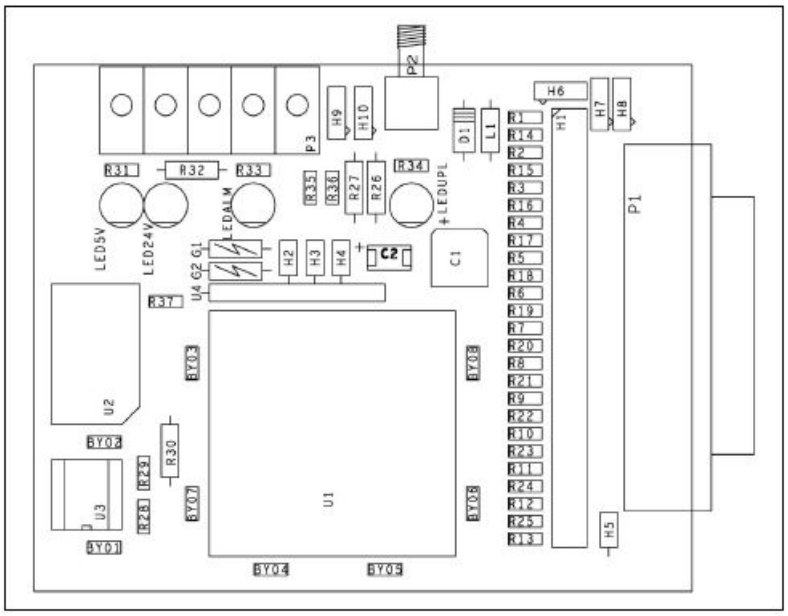


Figure 21 DAFC Assembly Diagram

DAFC Interface to STALO

The DAFC module is used on 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 (+5 VDC) must be provided by running discrete wires.

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, mount the DAFC within 30 cm of the STALO, if possible.

DAFC Pin Assignments and Jumper Selections

The digital output lines are made available as TTL levels on a 25-pin female "D" connector (P1). There are 130Ω 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 ... 20KΩ 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 (see the following table). The first three table entries describe three fixed mappings of the traditional AFC-16 uplink format on the pins of the 25-pin "D" connector. One of these choices must be used when 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. The recommended solution is to use the Pinmap uplink protocol (available since Rev.19), which allows for complete software mapping of all 25 external pins.

Table 23 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! Only install jumpers 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 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 ($0.75 \times f_{aq}$).

More Information

- Digital AFC (DAFC) (page 51)

4.4.1 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 following table shows the resulting pinout.

The STALO frequency is controlled by a 14-bit binary integer, whose LSB has a weight of 100 KHz. 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 24 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
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:

- 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. Wire-wrap them to the internal power posts.
- Supply the STALO and DAFC power externally through the terminal block on the DAFC.
- Install 16 jumpers installed to connect the Control and Status lines, that is, pins 4, 6–13, 18–23, and 25.
- H3 and H4 are removed because we use pinmap uplink protocol.
- Remove H2 because we us an x1 on-board crystal

The STALO has an output frequency range from 5200 MHz ... 6020 MHz in 100 KHz steps. In this example, we assume that we need an AFC frequency span of 5580 MHz ... 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   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 frequency limits 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.

4.4.2 Example: 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 25 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	--	--

Configure the DAFC pins:

- 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.
- H3 and H4 are removed because we use pinmap uplink protocol.
- Remove H5
- 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
```

4.4.3 Using the Legacy IFD Coax Uplink

In older RVP versions, such as RVP8 and RVP7, the coax uplink was the IFD single line of communication from the main processor board. Information needed by the IFD arrived through this uplink, including information useful to parts of the radar system such as DAFC.
While the legacy coax uplink protocol is no longer used directly, to support backward compatibility, the waveform is synthesized as an output from the IFDR. Any hardware previously attached to the coax uplink can be driven from this 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 Ω. When the cable is terminated in 75 Ω, the overall positive voltage swing is approximately 8.6 V.

The electrical characteristics of the uplink are optimized for balanced groundless reception. The recommended eavesdropping circuit is shown the following figure, and consists of a high-speed comparator (Maxim **MAX913** or equivalent) and input conditioning resistors. The shield and the center conductor of the coax uplink feed the comparator through 33 KΩ isolation resistors, no direct ground attachment is made to the shield. The 500 Ω resistors provide the local ground reference, and the 47 KΩ resistor supplies a bias to shift the unipolar uplink signal to a bipolar range for the comparator.

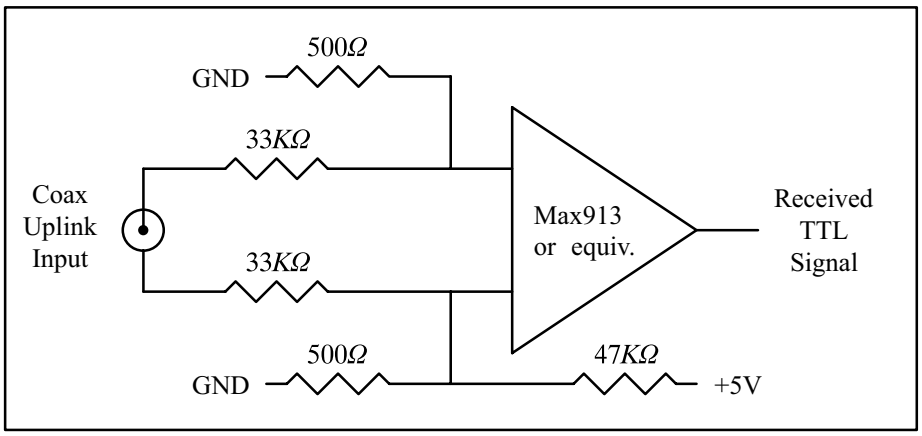


Figure 22 Recommended Receiving Circuit for the Coax Uplink

The uplink signal shown in the following figure is periodic at the radar pulse repetition frequency, and conveys two distinct information types to the IFDR. The signal is normally low (to minimize driver and termination power), but begins a transition sequence at the beginning of each transmitted pulse.

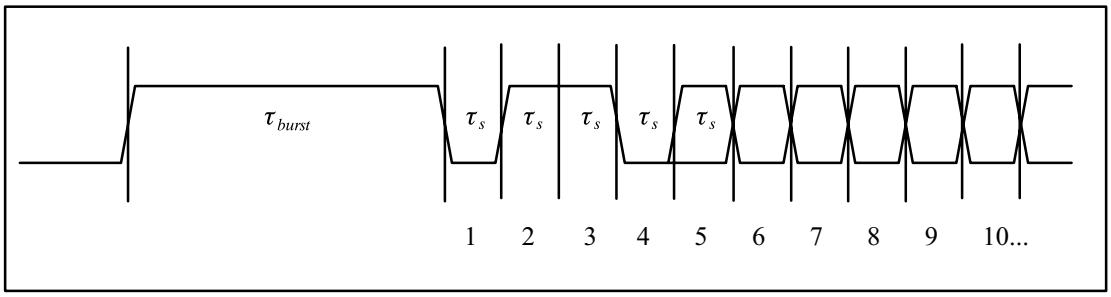


Figure 23 Timing Diagram of the IFD Coax Uplink

The first part of each pulse sequence is a variable length burst window that is centered on the transmitted pulse and has a duration τ_{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 chosen trigger timing and digital filter specifications, usually through the **Pb** and **Ps** plot setup commands.

After the burst window, there is a fixed-length sequence of 25 serial data bits that convey information from the IFDR. The following table shows the interpretation of the serial data bits.

Table 26 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. The first 0 in this pattern marks the end of the variable length burst window. You can check the other three bits to verify that a valid bit sequence is being received.
5 to 20	16-bit, multi-purpose data word, MSB is transmitted first
21	Reset Request. This bit sets in one transmitted sequence when 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
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 usually follows the "Downlink Detect" LED on the IFDR.

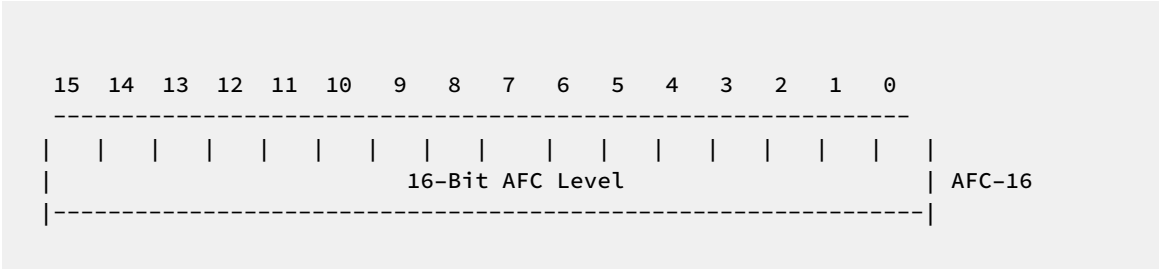
The period τ_s of the serial data is $(128/f_{aq})$, where f_{aq} is the acquisition clock frequency given in the **Mc** section of the RVP900 setup menu. For the default clock frequency of 71.9502 MHz, the period of the serial data is 1.779 µsec. The logic that receives the serial data should first locate the center of the first data bit at $(0.5 \times \tau_s)$ past the falling edge at the end of the burst window. Subsequent data bits are sampled at uniform τ_s intervals.

The 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. This flexibility helps when designing 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, you 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

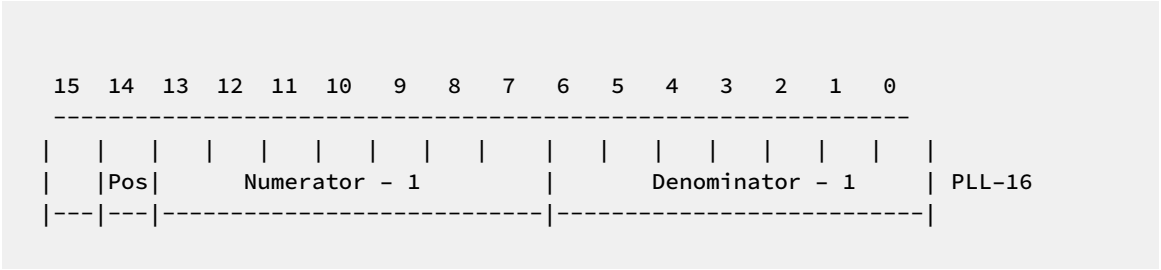
The serial 16-bit data word has different interpretations depending on the RVP900 configuration and whether `Bit #22` of the uplink stream is set or clear. The evolution of these different formats is in response to the evolution of the IFDR and DAFC modules. See 4.2 RVP901 IFDR Hardware (page 56) and 3.6 Digital AFC (DAFC) (page 51).

Orignally, the purpose of the uplink data word was 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. While it is available to support backwards compatibility, this format is not recommended for new hardware designs.



Level	Description
01111111111111111	(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 cannot be interleaved for use at the same time, since there would be no way to distinguish them at the receiving end.

An expanded format has been defined to handle 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 are available through 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. All 25-bits of pin information are transferred synchronously to the output pins only when `CMD=3` is received. This makes sure that momentary invalid patterns are not produced upon arrival of `CMD=1` or `CMD=2` when the output bits are changing.

Digital AFC Pinmap Commands

The `CMD=1`, `CMD=2`, `CMD=3` digital AFC pinmap commands are the recommended replacement to the original AFC-16 format.

If you only need 12-bits of linear AFC, map the AFC range to the -2048 ... +2047 numeric span, and select binary coding format (See 5.2.1 Mb — Burst Pulse and AFC (page 95)). The 12-bit data with `CMD=3` then holds the required values.

To get a full 16-bit value, use a -32768 ... +32767 span and extract the full word from `CMD=2` and `CMD=3`. Other combinations of bit formats and number of bits (up to 25) are also possible.

`CMD=4` also controls some IFDR internal features.

Table 27 Digital AFC Pinmap Commands

Command	Data Value	Description
`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

Command	Data Value	Description
CMD=4	Data<4:0>	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 most of the energy is in the 150 ... 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. Built-in noise generator level IF-input and burst-input selection
	Data<6:5>	Bits <6:5> permit the IF-input and burst-input signals to be reassigned on the downlink. 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

5. TTY Non-volatile Setups

5.1 Using the TTY Setup Menu

You can view and modify most RVP900 operating parameters with the TTY setup menu. For example:

- Make TTY connections
- Save and restore configurations
  For example, you can configure custom trigger patterns, pulse width control, matched FIR filter specs, PRF, and so on, in the field.
- Access graphical setup and monitoring procedures that use an ordinary oscilloscope as a synthesized visual display.
  The burst pulse and receiver waveforms are displayed in the time and frequency domain.
  The digital FIR filter matches the characteristics of the transmitted pulse.

1. To access the TTY menu:
   a. In the serial TTY or host computer interface, type: **dspx**
   The following prompt is shown:

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

   b. On the TTY, press ESC.
   RVP900 lists the RVP and IRIS software versions and shows a command prompt:

```
Vaisala Incorporated, USARVP9 Digital IF Signal Processor V13.1
IRIS-8.13.1
-------------------------------------------------
RVP9>
```

2. To view RVP and IRIS software versions, type **V**
3. For a list of available commands, type **help**

4. To exit the menu and reload RVP900 with the changed set of current values, type: **Q**
   Settings are saved in non-volatile RAM so they take immediate effect on start-up.

You must exit the menus using the **Q** before resuming normal RVP operation. Portions of the RVP command interpreter continue to run while the menus are active, but the processor does not function until you exit the menus.

5. To return to the Linux command prompt, press **SHIFT > C**.

5.1.1 Factory, Saved, and Current Settings

RVP settings included:

- Current settings—Collection of setup values with which RVP900 is currently operating
- Saved settings—Collection of values stored in non-volatile RAM. The saved settings are restored (made current) each time RVP900 starts

RVP900 retains the saved settings when new software releases are installed. The new version of the code automatically uses all of the previous saved values. If RVP900 detects a new setup parameter, it is set to a factory default value and a warning is printed if this occurs. See 5.1.2 V and Vz – View Card and System Status (page 91).

Table 28 Settings Commands

Command	Description
S	Saves the current settings into the non-volatile RAM
R	Restores those non-volatile values so that they become the current settings.
F	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 S	Saves factory defaults in non-volatile RAM. The site-specific power up settings are overwritten, irreversibly, and RVP900 powers up in its manufacturing mode. *This is not recommended.*
V	Shows the currently running versions of RVP and IRIS software. Each software release has a major version number (for example, 8.13), plus a minor version number for intermediate releases (for example, 8.13.2). The minor number starts at 0 at the time of each official release, and increments until the next official release.

You must exit the menus using the **Q** before resuming normal RVP operation. Portions of the RVP command interpreter continue to run while the menus are active, but the processor does not function until you exit the menus.

5.1.2 **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 counters are cleared, so that subsequent **V** commands show what has accumulated since the last **Vz**.

The view listing displays:

```
Configuration and Internal Status
---------------------------------
RVP9 Digital IF Signal Processor V13.1(Pol) IRIS-8.13.1
```

The displayed 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.

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

Then, information about when the RVP900 was started, current system time, and implicitly, the uptime.

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

Then information about the processor and the Intel libraries used for RVP900 processing.

```
CPU-Type: Pentium(R) 4 Hyperthreaded
IPP-Library: libippsw7.so v4.0 4.0.19.77
```

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

```
Diagnostics: PASS
```

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

```
Processes and Threads:
RVP9Proc-0  -   PID:6917Priority:10 Policy:RealTimeRR
RVP9Proc-1  -   PID:6918  Priority:10 Policy:RealTimeRR
  Chat/Plot   -   PID:6909Priority:10 Policy:RealTimeRR
  Watchdog    -   PID:6909Priority:10 Policy:RealTimeRR
  Burst/AFC   -   PID:6909Priority:10 Policy:RealTimeRR
  HostCmds    -   PID:6909Priority:11 Policy:RealTimeRR
  Angles      -   PID:6909Priority:12 Policy:RealTimeRR
  RtCtrl-0    -   PID:6909Priority:12 Policy:RealTimeRR
  RtCtrl-1    -   PID:6909Priority:12 Policy:RealTimeRR
  IQ-Data     -   PID:6909  Priority:13 Policy:RealTimeRR
```

This section provides RVP900 developers with information about code resources.

```
Shared library build dates:
RVP9/Main/Core:MonApr216:01:12EDT2012
RVP9/Main/Open:MonApr216:01:13EDT2012
RVP9/Main/Site:MonApr216:01:12EDT2012
RVP9/Proc/Core:MonApr216:01:14EDT2012
RVP9/Proc/Open:MonApr216:01:16EDT2012
RVP9/Proc/Site:MonApr216:01:14EDT2012
```

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

```
GPS:Inused
```

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 fourth order correlation model. See 6. Plot-assisted Setups (page 131).

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

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

This line shows the case temperature and FPGA chip core temperature, both in °C and °F.

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

This line shows the presently operating receiver mode.

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

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

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

5.1.3 Vp – View Processing and Threshold Values

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

The 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 numeric and symbolic forms.

```
Threshold Settings for All Data Parameters
       LOG       CSR       WSP      SQI      PMI      TCF (Equation)
       ---       ---       ---      ---      ---      -------------

DBZ:    0.75dB    -18.0dB    5.0dB    0.149    0.449    0x8888( LOG& CSR )
DBT:    0.75dB    -18.0dB    5.0dB    0.149    0.449    0xFFFF( AllPass )
VEL:    0.75dB    -18.0dB    5.0dB    0.398    0.449    0xC0C0( CSR& SQI )
WID:    0.75dB    -18.0dB    5.0dB    0.398    0.449    0xA000( LOG& SQI &SIG )
ZDR:    0.75dB    -18.0dB    5.0dB    0.398    0.449    0xAAAA( LOG )
KDP:    0.75dB    -18.0dB    5.0dB    0.398    0.449    0xC0C0( CSR& SQI )
PHIDP:  0.75dB    -18.0dB    5.0dB    0.398    0.449    0xC0C0( CSR& SQI )
RHOHV:  0.75dB    -18.0dB    5.0dB    0.398    0.449    0xC0C0( CSR& SQI )
SQI:    0.75dB    -18.0dB    5.0dB    0.398    0.449    0xFFFF( AllPass )
LDRH:   0.75dB    -18.0dB    5.0dB    0.398    0.449    0xAAAA( LOG)
RHOH:   0.75dB    -18.0dB    5.0dB    0.398    0.449    0xC0C0( CSR& SQI )
PHIH:   0.75dB    -18.0dB    5.0dB    0.398    0.449    0xC0C0( CSR& SQI )
LDRV:   0.75dB    -18.0dB    5.0dB    0.398    0.449    0xAAAA( LOG )
RHOV:   0.75dB    -18.0dB    5.0dB    0.398    0.449    0xC0C0( CSR& SQI )
PHIV:   0.75dB    -18.0dB    5.0dB    0.398    0.449    0xC0C0( CSR& SQI )
HCLASS: 0.75dB    -18.0dB    5.0dB    0.398    0.449    0xC0C0( CSR& SQI )
SNR:    0.75dB    -18.0dB    5.0dB    0.398    0.449    0xFFFF( AllPass )
DBZA:   0.75dB    -18.0dB    5.0dB    0.149    0.449    0x8888( LOG& CSR )
DBTA:   0.75dB    -18.0dB    5.0dB    0.149    0.449    0xFFFF( AllPass )
```

More Information

- Setup Operating Parameters (SOPRM) (page 235)

5.1.4 @ - Displaying and Changing Current Major Mode

The **@** command provides developers with a simple way of switching modes enabling on–the–fly testing of code.

For information on the top level RVP900 see **SOPRM** command word #9 in 8.4 Setup Operating Parameters (SOPRM) (page 235). This question allows you to use the mode that has been selected by that command, or to force the use of a particular mode.

5.2 Managing Settings with **M** Menus

You can use the **M** menus to view and modify current settings.

Type **??** to print the entire set of questions

The **M** menu works from the current parameter values, not from the saved values in non-volatile RAM. If the host computer has modified some current values, you see these changes as you browse the setup list.

Vaisala recommends making incremental changes to saved settings.

1. Type **R** to restore the saved values.
2. Type **M** to make the changes starting from that point.
   The current value of each parameter is displayed on the screen. The TTY pauses for input at the end of the line:
   - Press RETURN to move to the next parameter and leave the present parameter unchanged.
   - Type **U** to move back up in the list, and
   - Type **Q** to exit the list at any time.
   - Type a number or a YES/NO response to change the parameter's value, and displays the line again with the new value.
     An error message is displayed if entries are invalid.
3. Type **S** to save the new values.

You must exit the menus using the **Q** before resuming normal RVP operation.
Portions of the RVP command interpreter continue to run while the menus are active, but the processor does not function until you exit the menus.

More Information

- Mb — Burst Pulse and AFC (page 95)
- Mc — Top-level Configuration (page 104)
- Mf — Clutter Filters (page 106)
- Mp — Processing Options (page 108)
- Mt — General Trigger Setups (page 114)
- Mt<n> — Triggers for Pulsewidth n (page 117)
- Mz — Transmissions and Modulations (page 126)
- M+ Debug Options (page 128)

5.2.1 `Mb` — Burst Pulse and AFC

Type **`Mb`** to view and manage parameters that influence the phase and frequency analysis of the burst pulse, and the operation of the AFC feedback loop.

Frequency and Power Parameters

The first lines list the centers of the transmit and receive intermediate frequency bands. Although the Tx and Rx intermediate frequencies are usually the same, they are chosen separately so that the RF up-conversion chain for transmission can be different from the down-conversion chain for reception.

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

Limits: 6 MHz ... 72 MHz

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

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.

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

The `PhaseLock` parameter controls whether the RVP900 locks the phase of its synthesized `I` and `Q` data to the measured phase of the burst pulse.

- Type `YES` for an operational magnetron system, since the transmitter's random phase must be known to recover Doppler data.

- Type **NO** for non-phase modulated klystron systems in which the IFDR sampling clock is locked to the COHO.
  NO is also useful for bench testing. In these **NO** cases, the phase of **I** and **Q** is determined relative to the stable internal sampling clock in the IFDR module.

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

The **Minimum power for valid burst pulse** parameter 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. For information on the reporting burst pulse power, see 6.5 Pb — Plot Burst Pulse Timing (page 134). The value entered here should be, perhaps, 8 dB less. This makes sure that burst pulses are 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 pane contains only the active portion of the burst, and thus a mean power measurement is meaningful. The full FIR pane includes 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 ... +10 dBm

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

Choose the analysis window used in the design of the FIR matched filter and the presentation of the power spectra for the scope plots. Choices are:

- Rectangular
  The rectangular window is included as a teaching tool. Do not use it for operation.
- Hamming
  The Hamming window is the best overall choice.
- Blackman
  The Blackman window is useful if you want to see plotted spectral components that are more than 40 dB below the strongest signal present. It is useful in the **Pr** plot when a long span of data are available.
  FIR filters designed with the Blackman window has greater stopband attenuation than those designed with the Hamming window, but the wider main lobe may be undesirable.

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

The burst frequency estimator uses a fourth order correlation model to estimate the center frequency of the transmitted pulses. Each burst pulse typically occupies approximately 1 microsecond. The frequency estimate feeding the AFC loop must be accurate to, perhaps, 10 KHz. This accuracy cannot be achieved using just 1 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 ... 120 s

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

AFC and MFC Parameters

AFC is required in magnetron systems 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.

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

The following list appears only if AFC and MFC functions have been enabled.

You can configure the AFC servo loop 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 0 when the loop is locked; and that the voltage be proportional to frequency error while tracking. See 5.2.1.1 AFC Motor and Integrator Option (page 102).

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

AFC Loop

After turning on a magnetron transmitter, it can take 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 begins running.

Limits: 0 ... 300 s

```
Wait time before applying AFC: 10.0 sec
AFC hysteresis –– Inner: 5.0 KHz, Outer: 15.0 KHz
```

These are the frequency error tolerances for the AFC loop. The loop applies active feedback when the outer frequency limit is exceeded, but holds 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 remains constant most of the time; making small and brief adjustments only occasionally as the need arises.

In general, the AFC feedback loop is active only when RVP900 is not processing data rays. This is because the Doppler phase measurements are seriously degraded when the AFC control voltage makes a change. To avoid this, the AFC loop can only run 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. Use the following parameter to place an upper bound on the frequency error that is tolerated during sustained data processing. AFC is guaranteed to be applied when this limit is exceeded.

Limits: 15 ... 4000 KHz

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

The following parameters control the feedback computations of the AFC loop.

The control that is applied to the AFC is specified here in **`D-Units`**, that is, arbitrary units ranging from –100 ... +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 slow drifts in the radar system, so response times of this magnitude are reasonable.

```
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
```

Note 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. For information about these slope and slew rate parameters, see 5.2.1.1 AFC Motor and Integrator Option (page 102) .

```
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
```

AFC Output Process

The RVP900 AFC implementation has been generalized so that there is no difference between configuring an analog loop and a digital loop. The AFC feedback loop parameters are set the same way in each case. The only difference is the model for how the AFC information is made available to the outside world. Many types of interfaces and protocols are possible according to how these three questions are answered.

AFC output follows these steps:

1. 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 ... +32767 results in 16–bit AFC, and 0 ... 999 might be appropriate for 3–digit BCD; but any other span could also be selected from the full 32–bit integer range.
2. 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 includes in the list of supported formats most 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 encodes to `0x00419999 Hex`.
3. 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. This flexibility allows easy hookup to any STALO synthesizer.

Pinmap Uplink Protocol Parameters

These parameters 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 can 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 a power or ground connection. The latter must be done with 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 parameter allows you to choose any pin, however, because it does not know what hardware may be listening on the uplink and what its constraints might be.

```
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
```

Burst Parameters

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) results in an increase in the estimated burst frequency. If the AFC loop is completely unstable, try reversing this parameter.

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

This parameter enables the burst pulse tracking algorithm (see 7.2.4 Burst Pulse Tracking (page 175)). 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 Burst Pulse Tracking: YES
```

These parameters 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 parameters are needed to define the hunting process in time. The hunt in frequency is a different matter. The overall frequency range is always be the full -100% … +100% AFC span; but the number of sub-intervals 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% are tried, with a one–quarter second wait time before checking for a valid burst at each AFC setting.

```
Enable Time/ Freq  hunt for missing burst: No Number of frequency intervals to
search: 5  Settling time for each frequency hop: 0.25 sec
```

Choose the number of AFC intervals so that the hunt procedure can deduce an initial AFC level that is within a few MHz of the correct value. The normal AFC loop then takes over 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 sub-intervals 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 are suppressed in that case.

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 included as a work around; the correct way for a burst pulse hunt to occur is through an explicit request from the host computer which "knows" when the pulse really should be present. Blindly hunting cannot be done because there are many reasons why the burst pulse may legitimately be missing, for example, during a radar calibration.

```
Automatically hunt immediately after being reset: YES Repeat auto hunt every:
60.00 sec
```

The automatic hunt for the burst pulse is always run at least once when 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.

With this feature enabled, the processor adjusts 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. During calibration, we record the burst pulse power. If this number is available and provided to 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, turn on the `OPTS_ZCAL` flag in the **`SOPRM`** command. See `XARG 6` in 8.4 Setup Operating Parameters (SOPRM) (page 235).

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

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 is produced when RVP900 is in dual–receiver mode. The pulse is the sum of 2 transmit pulses at the primary and secondary intermediate frequencies. To make the simulation more realistic, the signal strengths are unequal; the primary pulse is 3 dB stronger than the secondary pulse.

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

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

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

More Information

- Automatic Frequency Control (AFC) (page 174)
- Digital AFC (DAFC) (page 51)

5.2.1.1 AFC Motor and Integrator Option

The `AFC Servo- 0:DC Coupled, 1:Motor/Integrator` question 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 must be overcome.

The `DC-Coupled` AFC loop questions 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
```

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
```

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. The **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 100 KHz produces the full–scale (100 **D–Units**) AFC output. But this is modified by the minimum and maximum slew requests as follows:

- A 0 **D–Unit** output is always be produced when AFC is locked.
- When AFC is tracking, the output drive is always at least ±15 **D–Units**. Set this minimum non–zero drive to the sustaining drive level of the motor actuator, that is, the minimum drive that keeps the motor turning.
- When AFC is tracking, the output drive never exceeds ±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, then the normal AFC parameters (minimum slew, maximum slew, feedback slope) all resume working properly. The algorithm operates as follows:

1. When AFC correction is applied, RVP900 calculates how long it would take to reach the desired IF frequency at the present rate of change.
   For example, if we are 1 MHz away from the desired IF frequency, and the measured rate of change of the IF burst frequency is 20 KHz/sec, then it is 50 seconds until the loop reaches equilibrium.
2. When the AFC loop is in Track–Mode, but the time to equilibrium is greater than 2 minutes, then the **minimum slew** parameter is slowly increased.
   The idea is to gradually increase the starting motor drive when it appears that the IF frequency is not converging toward the correct value, that is, the motor is stuck.
3. When the frequency begins to change, such that the desired IF would be reached in less than two minutes, the **minimum slew** parameter is immediately put back to its correct setup value.
4. 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. When MFC is enabled in the **Ps** command, it directly controls 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.

5.2.2 `Mc` — Top-level Configuration

You can use this set of commands to configure general properties of the IFDR.

This is the Ethernet address of the remote IFDR module. For best results, attach the IFDR to a dedicated Ethernet port of the host computer with no routers, switches, or hubs in between.

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

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, and so on.) the time series data rate may be as high as 50 MBytes/second. To improve the efficiency of packet transmission and reception, use jumbo packets, that is, packets that are longer than the typical 1500-byte standard. RVP900 allows jumbo packets up to 8192 bytes.

Limits: 250 ... 8192 bytes. Use, for example, **`ifconfig mtu 8192 eth0`** to set the receiver `Maximum Transfer Unit` size on the host computer. You must also configure any inline Ethernet switches to transmit the jumbo packets.

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

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. Usually the 1.5 MByte default provides reliable reception, but this can be increased to 5 MBytes if `UDP acquisition ID mismatch` errors are listed in the RVP900 diagnostic log.

Limits: 100000 ... 5 000 000 bytes

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

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

Limits: 50 MHz ... 100 MHz

```
IFD synthesized system clock: 72.0000000 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.

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

This setting is used to configure the input of live antenna angles. If angle data are supplied outside RVP900, for example 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.

Select `RtCtrl` when custom software has been written to deduce antenna angles from the real–time state machine controller.

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 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 16 bits), to which the offset angle is finally added.

```
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
```

For example, assume the elevation angle input to RVP900 is in an awkward form such as unsigned integer tenths of degrees, that is, `0x0000` for 0°, `0x000a` for 1°, 0x0e06 for -1°, and so on. 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 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. See 7.2.2 RVP900 Receiver Modes (page 172).

You can select the receiver mode in the **Mc** menu, but changes only take effect when you save them and restart RVP900.

5.2.3 Mf — Clutter Filters

When a clutter correction is applied to the reflectivity data, you must increase the **LOG** noise threshold slightly 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.

```
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
```

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 increases by **19 x 0.15 = 2.85 dB**. This helps eliminate noisy speckles from the corrected reflectivity data.

The **Spectral Clutter Filters** 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** to **#7**, enter a digit to choose the filter type, followed by however many parameters that type requires. See 7.3.4 Clutter Filtering Approaches (page 187).

```
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
```

Fixed Width Filters (Type 0)

These are defined by the following additional parameters:

Width
Sets the number of spectral points that are removed around the zero velocity term. A **Width** of 1 removes the DC term. A **Width** of 2 removes the DC term plus 1 point on either side. A **Width** of 3 removes DC plus 2 points on either side, and so on. Spectral points are removed by replacing them with a linear interpolating line.

EdgeMinPts
The endpoints of the line are determined by taking the minimum of **EdgeMinPts** past the removed interval on each side.

Variable Width, Single Slope (Type 1)

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 **Width** and **EdgeMinPts** parameters in the **Mf** menu, except that the **Width** is now interpreted as a minimum width.

The **Hunt** parameter allows you to choose how far to extend the notch beyond **Width** to capture the clutter power. Setting **Hunt=0** converts a variable–width filter to a fixed– width filter.

The algorithm for extending the notch width is based on the slope of adjacent spectral points.

- The beginning (**Width–1**) points away from 0, 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 run with a fixed **Width=3** filter, try experimenting with a variable **Width=2** and **Hunt=1** filter.
- If the original fixed width occasionally fails, but you are reluctant to increase it to cover those rare cases, try selecting a variable **Width=2** and **Hunt=2** filter.

In general, make your variable filters "wider" by increasing **Hunt** rather than by increasing **Width**. This preserves more flexibility in how they can adapt to whatever clutter is present.

Gaussian Model Adaptive Processing (GMAP) (Type 2)

This is the most advanced form of clutter filtering and moment estimation. See 7.3.4 Clutter Filtering Approaches (page 187).

For GMAP processing, you must specify the spectrum width of clutter. Note that the algorithm is not very sensitive to the exact value. You should configure several widths 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 measurements of the clutter width while the antenna is rotating using, for example, the **Ascope** utility or application software such as IRIS.

```
Whitening Parameters for  Tx :Random
```

The values in the next line define a secondary **SQI** threshold that is traditionally used to qualify **LOG** data in the Random Phase processing mode. The secondary **SQI** threshold is applied uniformly in all processing modes when reflectivity data are specified as being thresholded by **SQI**.

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

The secondary **SQI** level is computed by multiplying the primary user-supplied **SQI** threshold by the **SLOPE**, and adding the **OFFSET**. See 7.9.2 Tuning for Optimal Performance (page 226).

Limits: **SLOPE**: 0.0 ... 2.0, **OFFSET** -2.0 ... 1.0

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

```
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
```

5.2.4 Mp — Processing Options

When RVP900 computes power spectra, the time series data are multiplied by a (real) window before computing the Fourier Transform (DFT).

You can select the window through **SOPRM** word #10 **0:User**, or force a particular window:

- In the IRIS application, the setting **0:User** refers to the window defined in the **Setup** utility block RVP signal processing options.
- In the **Ascope** application, the setting **0:User** enables the **Spect Win** button.

At RVP900 start up, **0:User** refers to the settings saved in *rvp9.conf* block **opprm.filter** bits 9,10,11.

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

The power spectra computed within RVP900 are normally not constrained to be powers of two in length. Answer **NO** to reapply the constraint to mimic the behavior of older processors

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

Spectrum Width

The following parameters control whether **R0/R1/R2** or **R0/R1** estimates are used to compute the spectrum width (see 7.4.5 Spectrum Width Algorithms (page 202) and 8.4 Setup Operating Parameters (SOPRM) (page 235)):

- Select **0** to unconditional disable the **R2** algorithms, regardless of what the host computer requests in the **SOPRM** command.
- Select **2** to unconditionally enable **R2** processing. Bit-7 of **SOPRM** word #2 is the host computer's interface to this function when the **1:User** case is selected.
- IRIS radar application uploads **R0/R1/R2**.

Ascope uploads the **R2 algorithms** button setting in the block *Gen Setup*. At RVP900 start up, the user setting is retrieved from *rvp9NV.opprm.iflags*.

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

Clutter Microsuppression

The following parameter controls whether "cluttery" bins are rejected before 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 8.4 Setup Operating Parameters (SOPRM) (page 235)). The functionality is dependent on the application settings for CCOR thresholds, in addition (see 3.3.4 Range Averaging and Clutter Microsuppression (page 45)).

IRIS requests clutter microsuppression. **Ascope** uploads the **Clutter Microsuppression** button setting in the block *Gen Setup*.

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

Autocorrelation

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 8.4 Setup Operating Parameters (SOPRM) (page 235)). The IRIS does not request PPP-style autocorrelations, while **Ascope** requests it. At RVP900 start up, the user setting is retrieved from *rvp9NV.opprm.iflags*.

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

Velocity and Processing Algorithms

The following parameter allows you to choose whether RVP900 unfolds velocities using a (V_{high}– V_{low}) algorithm, instead of the improved algorithm described in 7.8 Dual PRF Velocity Unfolding (page 219). Bit– 11 of **SOPRM** word #10 is the host computer's interface to this function when the "**1:User**" case is selected (see 8.4 Setup Operating Parameters (SOPRM) (page 235)).

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

The following parameter allows you to choose if RVP900 attempts to run its standard processing algorithms, even when a custom trigger pattern has been selected through the **SETPWF** command. Usually, this does not make sense, 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 8.4 Setup Operating Parameters (SOPRM) (page 235)).

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

Additional SNR

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

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

Free-running Rays

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 can establish a minimum hold-off between successive rays, expressed as a percentage of the number of pulses contributing to each ray:

- 100% (default) produces rays whose input data do not overlap at all, that is, whose rate is exactly the PRF divided by the sample size.
- 0% gives the unregulated behavior in which no minimum overlap is enforced and rays can be quickly produced.

Limits: 0 ... 100%

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

Saturation Algorithm

RVP900 uses a statistical saturation algorithm that estimates the real signal power correctly even if the IF receiver is over-driven (that is, for input power levels above +4 dBm). The algorithm extends 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 is applied. Choose 0 dB to disable the algorithm.

Limits: 0 dB ... 5 dB

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

Amplitude Correction

RVP900 can perform pulse–to–pulse amplitude correction of the digital (**I**,**Q**) data stream based on the amplitude of the Burst/COHO input. See 7.2.7 Correction for Tx Power Fluctuations (page 180).

Limits: 10 pulses ... 500 pulses

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

Channel Separation

The following parameters appear if you select **WDR** in the **Mc** (see 5.2.2 Mc — Top-level Configuration (page 104)).

You must determine the **Channel separation** and **Overlap/Interpolate interval** from the **Pr** printout. Sweep a signal generator across the shared power region of the two channels to determine a representative channel separation, along with the size of the overlap region at the top of the **HiGain** channel within which that separation remains steady and constant, that is, unaffected by eventually approaching the noise floor of the **LoGain** channel.

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, 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**.

```
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
```

Interface Filter and Unfolding

RVP900 can optionally apply an interference filter to remove impulsive-type noise from the demodulated (**I**,**Q**) data stream. See 7.2.5 Interference Filter (page 176).

```
Interference Filter 0:None , Alg.1, Alg.2, Alg.3: 1 Threshold parameter C1:
10.00 dB
Threshold parameter C2: 12.00 dB
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 a general implementation of a Lo/Hi Surveillance/Doppler PRF unfolding scheme that provides the legacy features as special cases. The parameters are defined as follows:

- The maximum range to unfold (in km) 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 is somewhat noisy. You can 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 0 means **No averaging**, a value of 1 averages 3 points, and so on.
- 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 threshold is specified in decibels.
- The fundamental RVP900 operating parameters (PRF, Sample Size, and so on) 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 0, 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 entry points **dspw_batchSetup()** and **dspw_batchSetup()**. These use the custom opcode that is defined separately by each major mode. The **customUserOpcode_batch()** is a useful model building these entry points.

Standard Parameters in Multi-Polarization Systems

The following parameters control how the standard parameters (**Total Reflectivity**, **Corrected Reflectivity**, **Velocity**, and **Width**) are computed in a multiple polarization system. Applying **YES** with **H-Xmt** or **V-Xmt** means that data from those transmit polarizations is used when there is more than one choice available. These selections only apply to the Alternating and Simultaneous transmit modes.

Applying **YES** to **Co-Rcv** or **Cx-Rcv** means that received data from the co-channel or cross- channel is used. The receiver question appears when dual simultaneous receivers have been configured.

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

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

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

You can configure the sign and offset corrections to correct for intervening weather attenuation. It uses the change of angle in DP. The tuning numbers are taken from the *dualpol.conf* file.

```
PhiDP  – Negate:  NO , Offset:0.0  deg
Polarimetric Attenuation Correction :Yes
```

Choose what kind of K_{dp}computation you would like to use, where:

`LSQ`
: Least square fit to a linear function.

`Weighted LSQ`
: Weighted least square fit to a linear function. FIR filter coefficients are used as weights.

`Cubic Splines`
: Smoothing and numerical derivatives calculation using a CSU's Adaptive estimation of specific differential phase algorithm.

```
KDP computation – 0 :LSQ, 1:Weighted LSQ, 2:Cubic Splines
KDP – LSQ Length: 4.00 km
```

For `LSQ` K_{dp} computation, select the length:

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

For `Weighted LSQ` K_{dp} computation, select the FIR width:

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

For `Cubic Splines` K_{dp} computation: Smoothing factor for the first, non– adaptive, stage of cubic splines processing followed by 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 uses **H-Xmt:YES, V-Xmt:No, Co-Rcv:YES, Cx-Rcv:NO**. Including both transmissions decreases the variance of (**T/Z/V/W**), but most researchers prefer excluding **V-Xmt** because that is more standard in the literature.
If your polarizations are such that the main power is returned on the cross channel, then you probably want **Co- Rcv:NO** and **Cx-Rcv:YES**.

Melting Height

The melting height is used in **HydroClass** calculations and is recorded with the data.

This is the height above the radar, so must include the altitude of the radar when computing. Normally this is set automatically by the controlling application.

```
Melting height: 3000 meters
```

Noise Correction

If the noise correction is set to **Yes**, RVP900 adjusts the calibration reflectivity value **Z0** when the current noise level changes from the level measured when the calibration was done.

See 7.4.3.1 Noise Correction to Reflectivity Calibration (page 200).

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

5.2.5 Mt — General Trigger Setups

To configure the general properties of the RVP900 trigger generator, type **Mt** with no additional arguments.

You can check the **Pulse Repetition Frequency** and **Transmit pulse width: 0** of the internal trigger generator (Limits: 50 ... 20000Hz) and the **Transmit pulse width: 0** (Limits: 0 ... 3).

```
Pulse Repetition Frequency: 500.00 Hz
Transmit pulse width: 0
```

External Pretrigger

When an external pretrigger is applied to the RVP900 **TRIGIN** input, 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.

```
Use external pretrigger : NO
  PreTrigger active on rising edge: YES
  PreTrigger is synchronous with IFD AQ clock: No
```

Answer the `PreTrigger fires the transmitter directly` sub-question according to whether the radar transmitter is directly fired by the external pretrigger, rather than by one of the RVP900 trigger outputs. Answer `YES` if the transmitter can continue running even if the RVP900 `TRIGIN` signal is 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 is affected rather than the start times of the six output triggers.

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

Output Triggers

The number of user–defined output triggers is limited to 12 (including polarization output controls).

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

Polarization Output Controls

The next setting defines the number of polarization output controls.

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

Blankable Triggers

Modify the next setting to reduce erroneous transmissions into physical obstructions.

RVP900 can inhibit the subset of blankable trigger lines when a noise measurement is taken. This is forced when trigger blanking (based on `TAG0`) is enabled, but it can also be selected in general through this question. Since noise triggers must be blanked when trigger blanking is enabled, this question only appears if trigger blanking is disabled.

These settings permit the state of the triggers during noise measurements to be consistent and known, regardless of whether the antenna is in a blanked sector.

```
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
```

To choose whether you use blanked noise triggers all the time, use the following parameter.

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

To choose whether you use blanked noise triggers when changing pulsewidth, use the following parameter.

```
Blank output triggers when changing pulsewidth : YES
    Times to wait (miliseconds) before/after PW change: 0 0
```

Transmitter- and Receiver-related Triggers

You can specifiy which triggers are transmitter-related and which are receiver-related.

Answer **YES** or **NO** responses for each trigger line, for the two polarization control lines, and for the timing of the phase control lines.

Answer **NO** for any trigger that is involved with the pre–fire timing of the transmitter.

If you enable the burst pulse tracker, you probably want to assign a **YES** to some of your triggers so that they remain fixed relative to the burst itself. See 7.2.4 Burst Pulse Tracking (page 175).

The trigger outputs are defined by:

- Which RVP900 trigger outputs are timed relative to the transmitter pre–fire sequence
  You can move these triggers left/right using the **L/R** keys in the **Pb** plot.
  These triggers are also skewed in response to Burst Pulse Tracking. See 7.2.4 Burst Pulse Tracking (page 175).
- Which RVP900 trigger outputs are relative to the received target ranges
  These triggers remain fixed relative to "receiver range zero", and are not affected by the **L/R** keys or by tracking.

For the trigger start times, move triggers that fire the transmitter, either directly or indirectly as a group when hunting for the burst pulse and moving it to the center of the FIR window. Fix triggers that function as range strobes relative to range zero, that is, the center of that window, and the center of the burst. This distinction is important when the transmitter's pre–fire delay drifts with time and temperature.

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

Use the next setting to compensate for the offset in range that is due to the length of waveguide connecting the transmitter, antenna, and receiver. 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 compensates for the additional waveguide length to within plus–or–minus half a bin, and works properly at all range resolutions.

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

5.2.6 `Mt<n>` — Triggers for Pulsewidth n

To configure pulse width triggers, type **`Mt`** with an additional argument for the pulse width number.

You can configure specific trigger, transmit waveform, and FIR filter properties for the indicated pulse width.

Triggers

Trigger parameters list the starting times (in microseconds relative to range zero), the widths (in microseconds), and the active sense of each trigger generated by the internal trigger generator.

Set a width to `0` to inhibit 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 ... +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. If you do not want to use this term, enter this multiplier as `0`, and is omitted from the printout.

In the following example, `Trigger #2` is a 10.0 µsec active–high pulse with a leading edge that occurs precisely halfway between the zero–range of every pair of pulses. Similarly, `Trigger #6` is a 2.0 µsec active–low pulse with a falling edge that is nominally 5.0 µsec prior to range zero, but that 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 the trigger period.

```
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
```

Some subtleties of these variable start times are:

- The **PRT** multipliers can only be used with the RVP900 internal trigger generator. The **PRT**–relative start times are completely disabled when 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 is drawn cross–hatched at a location computed according to the current PRF. The cross–hatching serves as a reminder that the location of that trigger may vary from its 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 when RVP900 generates multiple–PRT triggers, for example, during DPRT mode, or during dual–PRF processing.
  Multipliers from 0.0 ... +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)**) positions 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.

In some situations, waveforms that do not fit are zeroed (not output) to preserve the desired period. This means that you can define triggers with large positive start times, and they come into existence only when the PRF is low enough to accommodate them. It applies when:

- The trigger period is internally determined, that is, the external pretrigger input is not being used.
- The overall span of the six trigger definitions combined does not fit into that period.

For example, if **Trigger #2** is defined as a 200.0 µsec pulse starting at +400.0 µsec:

- The trigger is suppressed if the PRF is 2000 Hz.
- The trigger is present at a PRF of 1000 Hz.
- If a trigger does not completely fit within the overall period it is suppressed.

This means that although the +400.0µsec start time is still valid at 2000Hz, the entire 200.0 µsec pulse would not fit, and the pulse is eliminated.

Start limits: -5000 ... 5000 µsec

Width limits: 0 ... 5000 µsec

PRF Protection Limits

When defining the PRF protection limits for this pulse width, consider:

- In the `Maximum number of Pulses/Sec` question, 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 2 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 cannot be set lower than the maximum number of pulses per second.

PRF limits: 50 Hz ... 20000 Hz

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

Range Zero

The `range zero` is the time at which the signal from a target at zero range appears at the radar receiver outputs.

This parameter adjusts the delay from the active edge of the external trigger to range zero.

This delay must be correct when RVP900 operates 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, the signals are adjusted to accomplish the alignment of range zero.

See 5.2.1 Mb — Burst Pulse and AFC (page 95) and A.12 Checking Burst Pulse and AFC with Setup Mb Command (page 346).

Limits: 0.1 µsec ... 1000 µsec

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

Range Resolution

The RVP900 range resolution 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 5.2.2 Mc — Top-level Configuration (page 104).

The ranges that are selected by the bit mask in the **LRMSK** command are spaced according to the range resolution you define 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 cannot 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 ... 1000 m (exact limits depend on other Rx setup values)

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

Intermediate Frequency

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 ... 72 MHz

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

Filter Length

RVP900 computes **I** and **Q** using a digital FIR matched filter. Define the length of that filter (in microseconds) here.

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

Table 29 Filter Length Considerations

Consideration	Description
Transmitted pulse width	The filter length should be at least as long as the transmitted pulse width. If it were shorter, 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.
Range bin spacing	The filter length should be at least as long as the range bin spacing. The goal 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.
Out-of-band noise	The filter length 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. This improves the **SNR** of weak signals.

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 0 on each end. 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 µsec at 125 m resolution in single-polarization mode, and up to 20 µsec at 16- meter resolution. In dual-polarization mode, the limits are 80 µsec at 125 m and 20 µsec 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 µsec) 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 requires (25.3 µsec)(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 Frequency Estimator

This estimator is mostly used with the **Pb** (plotting commands). See 6.5.2 Pb Subcommands (page 136).

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

FIR–Filter Prototype Passband Width

The passband width of the ideal lowpass filter is used to design the matched FIR band pass filter.

The bandwidth of the final FIR filter depends on the filter's impulse response length and the design window used in the process. The 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.

You must tune this parameter using the TTY output and interactive visual plot from the **Ps** command. The 3 dB bandwidth is shown in the plot for comparison with the ideal prototype bandwidth.

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

Limits: 0.05 MHz ... 10.0 MHz

Pulse Width Hardware Control

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 ... 15 (input must be typed in decimal)

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

Noise Levels

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. These questions are intended for applications in which RVP900 must operate with a reasonable default value, up until the time that an **SNOISE** command is 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.

- 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, and when the **RESET** opcode is issued with **Bit #8** set.
- The current noise level is revised when the **SNOISE** opcode is issued.

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

-or-

```
Current noise level: -75.00  dBm Powerup noise level: -75.00 dBm
```

Phase Processing Transition Time

This is the transition time of the RVP900 phase control output lines during random phase processing modes.

Select the switch point should so that there is adequate settling time prior to the burst/COHO phase measurement on each pulse.

This question appears if the **PHOUT[0:7]** lines are configured for phase control. See 5.2.2 Mc — Top-level Configuration (page 104).

Limits: -500 µsec ... 500 µsec

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

Polarization Switch

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. These questions set the transition points of the control signals.

The values are in microseconds relative to range zero. The same units define the start times of the six user triggers.

To optionally set the logical sense of the polarization output lines, invert the signals in the *softplane.conf* file.

Limits: -500 µsec ... 500 µsec

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

5.2.6.1 Tx Synthesis Options

Several **Mt** dialogs are modified when RVP900 is equipped with an IFDR that is configured for Tx waveform synthesis in the **Mz** menu. See 5.2.7 Mz — Transmissions and Modulations (page 126).

When digital Tx waveforms are synthesized, the following questions appear in the **Mt<n>** menu after the **Rx Intermediate Frequency** question.

In this case, each of the RVP900 four pulse widths can select a different type of transmit waveform and associated matched receiver.

For example, to make it easy for application software such as **Ascope** to transparently switch between radically different Tx waveforms by requesting a different pulse width for each one:

- **PW–0** and **PW–1** could transmit conventional 0.5 µsec and 2.0 µsec CW pulses that are received using the bandpass filters.
  See 5.2.6 Mt<n> — Triggers for Pulsewidth n (page 117).
- **PW–2** and **PW–3** could be configured as 20 µsec and 40 µsec compressed non–linear FM waveforms.

Tx Waveform

RVP900 supports three standard Tx waveforms:

- Conventional fixed– frequency CW pulse
  The **CWPulse** can be used as a pulsed Doppler waveform in all the same way that a Klystron or Magnetron system with a traditional pulse forming network would be used.
- Linear and non–linear FM chirp
  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.

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

The RVP900 internal APIs permit developers to create arbitrary waveforms for transmission. The types listed above are the out–of–the–box selections that are standard on RVP900 processors.

The next 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 the frequency modulation and amplitude modulation used by the waveform. A spectrum analyzer (or the RVP900 **Ps** plot) shows 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 removes 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) is 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 for that the final analog signal to be a faithful reproduction of the intended waveform.

The Tx waveform is normally synthesized with its center lined up with range 0. 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.

```
Bandwidth of transmit pulse: 3.25 MHz Pulselength of transmit pulse: 15.00
usec
Zero offset of transmit pulse: 0.00 usec
```

Configuring a pulse configuration for a hybrid pulse 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 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>** commands.

Check this transmit pulse timing offset in the **Pb** plot by making sure that the Tx burst is centered within the FIR data window.

```
Hybrid pulse chained PW index
```

RVP900 uses 3 real–valued tuning parameters to make the synthesis of complex waveforms more flexible. You can alter and fine-tune each waveform class up to 3° of freedom, making it possible for a single class (for example, the non–linear FM class) to generate a huge variety of waveforms. These adjustable constants also form the basis of the automatic waveform optimization procedure for the **Pa** command. See 6.8.2 Available Subcommands In Pa (page 161).

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

CWPulse Class

The **CWPulse** class of waveforms do not use the tuning parameters because the Tx waveform is determined by the desired bandwidth and pulse width, that is, there are no remaining degrees of freedom to adjust. These questions do not appear in the **CWPulse** case.

```
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
```

Linear and Non-Linear FM Class

The linear FM class is specified by the bandwidth and pulse width values, and does not reference any tuning parameter.

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.
  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.
  See 6.8.1 Interpreting Ambiguity Plots (page 159).

- The class of non–linear FM curves always pass through the points (- 1,-,1), (0,0), and (1,1).
  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. 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 results 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.

Table 30 Linear FM Class Examples

Example	Description
`P1 = 0.0, P2 = 0.0, P3 = 1.0`	`P1` and `P2` place the FM breakpoint at the origin. Because the FM curve passes through that point, the response reverts to linear FM. `P3` indicates that amplitude modulation should not be applied until the very end of the pulse, and thus does not occur at all. The resulting waveform is 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. 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 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 neatly within their prescribed bandwidth limits. We recommend using non–linear FM waveforms that combine chirp limits and amplitude modulation in this manner.

5.2.7 `Mz` — Transmissions and Modulations

Use these questions to configure the phase modulation codes that may be used to control the phase of a coherent transmitter.

Phase Modulation

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

```
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
```

Waveform Types

Define the type of waveform to generate on the Tx-A SMA output. In this example a 60 MHz 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 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
```

Transmit Waveforms

Select the type of waveform to generate on the Tx-B SMA output.

In the following example, a pulsed transmit waveform is generated with a peak output level of 0.0dBm. Phase modulation can optionally be applied.

The details of the pulse are defined in the **Mt<n>** menu for each pulse width. See 5.2.6 Mt<n> — Triggers for Pulsewidth n (page 117).

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

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, the next question lets you supply adhoc frequency band compensation. The units of this parameter are: percentage amplitude compensation per MHz 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.

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

More Information

- Mc — Top-level Configuration (page 104)
- Mt<n> — Triggers for Pulsewidth n (page 117)

5.2.8 M+ Debug Options

RVP900 debugging options help you develop and debug application code. You can also use these questions to put RVP900 in unusual operating states. These options are usually disabled during normal radar operation.

This is the noise level that is assumed when simulated **I** and **Q** data are injected into RVP900 with 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 ... 0dB

```
Noise level for simulated data: -50.0 dB
```

This question asks whether IF samples should be plotted with Nyquist flipping (multiplication by +1,–1,+1,–1...).

If **YES**, the **Pr** and **Pb** plots modify their rendering of IF samples. In those menus, the text **(NyFlip)** appears in the plot text heading when sample flipping is being applied. Other plots in these menus, such as **spectra** and **LOG magnitude**, are unaffected by the Nyquist flipping, nor are any live parameters derived from the IF samples. Only the visual appearance of the IF samples is affected.

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

WSR98D Test Points

This question appears on systems configured with the WSR98D control panel for choosing the signals output on the RVP-controlled test points. A value of **0** means no output. To save driver power, set these to **0** if the test points are not in use.

Many of these signals are WSR98D triggers:

- **Update** signals are triggers used to control the timing of when the control panel outputs are updated.
- **Uplink** signals are signals used on the uplink cable between the IFDR and the control panel.

```
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)
```

6. Plot-assisted Setups

6.1 Plot-assisted Setup Overview

The IFDR receiver module replaces most IF components in a traditional analog receiver. The alignment procedures for those analog components are usually very tedious, and require continued maintenance.

RVP900 improves the alignment procedure by providing an interactive graphical alignment procedure for burst pulse detection, Tx/Rx phase locking, and calibration of the AFC feedback loop with tools for:

- Viewing samples of the burst pulse and receiver waveform to examine their frequency content, design an appropriate matched filter, and observe live operation of the AFC.
- Checking the spectral purity of the transmitter on a regular basis to check for unwanted noise or harmonics.

RVP900 can track and modify the initial settings so that proper operation is maintained even with changes in temperature and aging of the microwave components.

To access the Plot-Assisted Setups, type the **P** commands in the TTY setup interface.

You can use the **dspx** utility to display the plots on the workstation screen, and you can perform graphical checkup and alignment procedures remotely through a network.

RVP900 supports opcodes that allow the host computer to monitor the data being plotted.

6.2 Plot Command Conventions

The **Pb**, **Ps**, and **Pr** commands have a similar structure to their TTY user interface.

Each command begins by printing a list of subcommands that are valid in that context. The subcommands are single keystrokes that RVP900 executes immediately when they are typed, you do not need to press ENTER. The available subcommands are different for each plot command. 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 change when a subcommand is issued.
  The first line is printed once.
- The second line is live and reflects the current status of the burst input, the IF input, or the AFC output.
  The second line is continually overprinted. This makes it appear as a live status line with up-to-date values.

The **Pb**, **Ps**, and **Pr**commands report `No Trigger` on the TTY status line when the external trigger is expected, but missing.

Table 31 Plot Command Conventions

Convention	Description
Display commands	Most plot commands support subcommands that alter the appearance of the display, for example, zoom, stretch, and so on. These subcommands make no changes to the working RVP900 calibrations. Display settings are stored in nonvolatile RAM, like the other setup parameters. Previous display settings are restored when you restart each plot command. If the initial list of subcommands disappears off the top, type **?** to force a reprint.
Exit	To exit the plot command and return to the TTY main menu, type **Q** or ESC. All other keys return the plot command to its normal live updating, but the key is otherwise discarded (for example, subcommand keys are not executed while exiting from single step mode).
Lower case commands Upper case commands	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, the **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.
Single Step mode	By default, the graphical display and TTY status lines are updated with fresh data several times per second. If you wish to freeze a plot for further analysis or comparison enter **Single Step** mode by typing: **.**" (period) This causes the display and TTY status lines to freeze in their present state, and prints the message **"Paused..."** is printed. To move to the next data update, retype **.**" (period). The plot and printout remain frozen at the next data update.
Scrolling	The TTY screen scrolls upward each time a new subcommand is executed to show a history of information lines and command activity. Press ENTER to scroll the display up at any time.
%	Press **%** to toggle between the IF input SMA connectors on the IFDR.

6.3 Configuring RVP900 Digital Front End

You can use the **Pb**, **Ps**, and **Pr** commands to configure the RVP900 digital front end.

You may run the commands at any time. The following procedure is for first-time setups.

Sometimes it is useful to run plot commands with samples from the IF-Input of the IFDR, rather than from the Burst-Input.
Press **%** to toggle between the IF input SMA connectors on the IFDR.

1. Use **Mb** to set the systems intermediate frequency.
   See 5.2.1 Mb — Burst Pulse and AFC (page 95).
2. Use **Mt** to choose the PRF and pulse width. Also, choose the range resolution, as it may constrain the design of the matched filter later.
   See 5.2.5 Mt — General Trigger Setups (page 114).
3. Use **Mt0, Mt1,** and so on to set the relative timing of the RVP900 triggers used by the radar.
   Do not worry about the absolute values of the trigger start times. Set their polarity and width, and their start times relative to each other.
   Make an initial guess of FIR filter length as 1.5 times the pulse width.
   See 5.2.6 Mt<n> — Triggers for Pulsewidth n (page 117).
4. Use **Pb** to slew the start times of all triggers so that the burst pulse is properly sampled.

   Refine the impulse response length if necessary so that all samples easily fit within the display window.
   See 6.5 Pb — Plot Burst Pulse Timing (page 134).
5. Use **Ps** to design the matched FIR filter.
   See 6.6 Ps — Plot Burst Spectra and AFC (page 139).
   a. 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 has changed, return to **Pb** to verify that the burst is still sampled properly
6. Continue using **Ps** and **Mb** to tune the AFC feedback loop.
   The settings that work for one pulse width should also work for the others.
7. Use **Pr** to verify that targets are being detected with good sensitivity.
   See 6.7 Pr — Plot Receiver Waveforms (page 154).
8. Repeat the procedure for each pulse width supported by the radar.

6.4 **P+** — Plot Test Pattern

You can use the **P+** command to generate a simple test pattern and verify that the display software works properly.

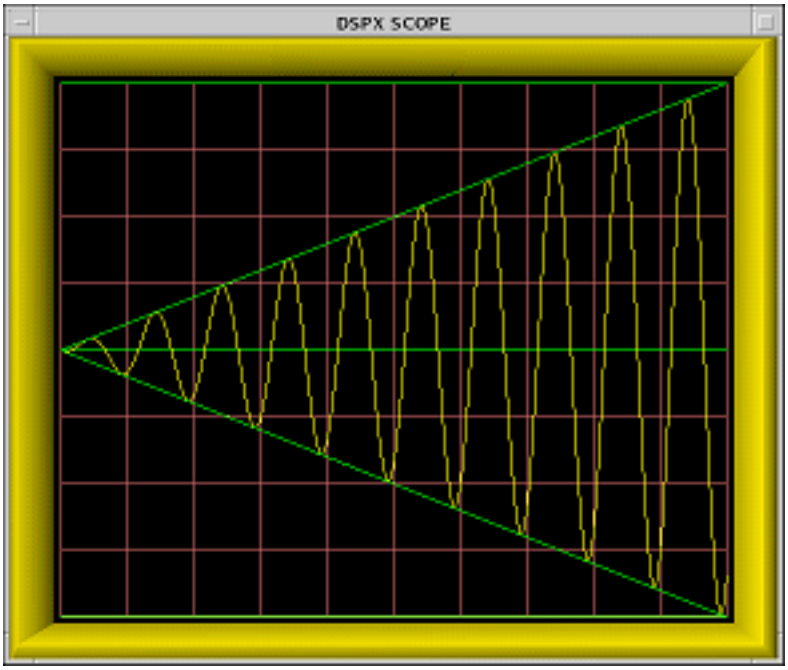


Figure 24 Test Pattern Display

1. In the TTY monitor type: **P+**
   This prints the message `Plotting Test Pattern...` on the TTY and then produces the plot shown in the following figure. The display is an overlay of the following strokes:
   - Bottom line
   - Middle line
   - Top line
   - Line sloping up
   - Line sloping down
   - Sine wave pattern that changes phase with each plot so that it appears to radiate from the left side of the display.
2. When you are satisfied that the plot is drawn correctly, type **Q** or press ESC to return to the TTY monitor.

6.5 **Pb** — Plot Burst Pulse Timing

For magnetron radars, 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.

Use the **Pb** command to adjust the trigger timing and A/D sampling window so that the burst pulse is correctly measured.

6.5.1 Interpreting the Burst Timing Plot

The following figure shows an example of a burst timing display plot of a successful capture of the transmitter's burst pulse.

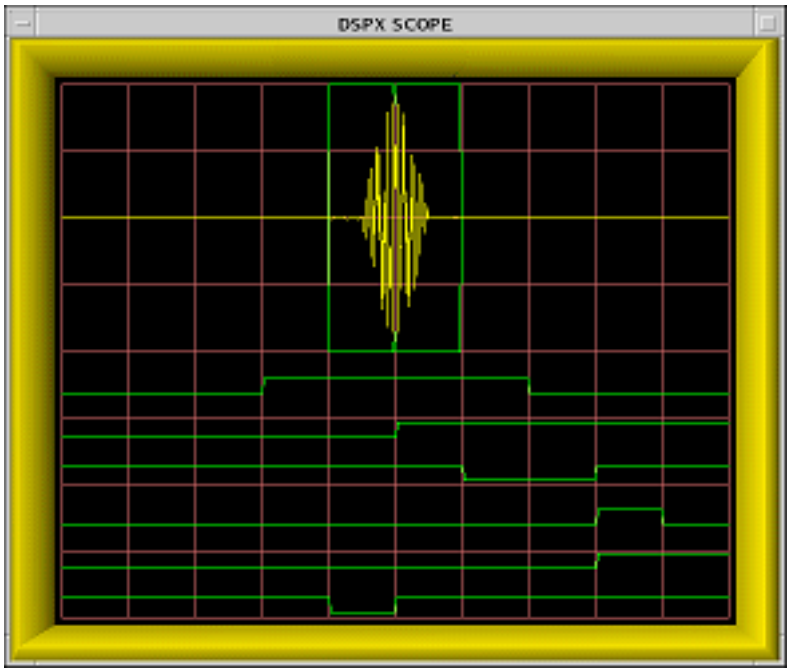


Figure 25 Successful Capture of the Transmit Burst

Time

The horizontal axis of the display represents time. The duration is determined by the impulse response length of the matched FIR filter.

The same FIR coefficients that compute `I` and `Q` are used to compute the reference phase vectors for the burst pulses.

The upper portion of the plot shows the sampling window where the burst pulse is measured.

On the TTY, the overall time span from the left edge to the right edge is listed as `PlotSpan`.

Plot

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.

The A/D samples of the IFDR burst input are plotted (somewhat brighter) in the sample window.

AFC

You can independently choose a sub-interval of burst pulse samples that are used by the AFC frequency estimator.

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 transmitted pulse, and thus, the samples contributing to the frequency estimate includes 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 misled by these edges and might not tune to the optimum center frequency if they were included.

Triggers

The lower portion of the plot shows the triggers output by RVP900. The number of triggers plotted match the number of user-defined output triggers set in the **Mt** menu, with `Trigger #1` 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.

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.

6.5.2 **Pb** Subcommands

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.

Table 32 **Pb** Subcommands

Command	Description
I/i	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	Raise/lower the aperture/start of the subwindow of burst pulse samples for AFC. If you do not use these commands, the full FIR window is used. Shortening the AFC interval results in two sample windows being drawn on the plot. Position the smaller AFC window positioned in the center portion of the transmitted pulse, where the burst amplitude and frequency are fairly stable.
L/l & R/r	Shift the 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 6 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	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	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 the IFDR IF-Input sources.
B/b	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.
+	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.

6.5.3 TTY Information Lines In **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
```

Table 33 **Pb** TTY Information Lines

Information Line	Description
`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. 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.
`IF`	Indicates which of the 5 IF-input sources (SMA edge connectors) is providing the data being plotted.
`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. See 5.2.1 Mb — Burst Pulse and AFC (page 95).
`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`	Shows the present value of timing slew (measured in microseconds) being applied to track the burst. The slew is 0 initially when RVP900 is first powered up, meaning that the normal trigger start times are all being used.

6.5.4 Adjusting Burst Pulse Timing

You must calibrate the burst pulse timing separately for each pulse width. Each iteration is independent.

1. Setup the relative timing for all used RVP900 triggers.
   The trigger output lines are 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.
2. 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 6.6 Ps — Plot Burst Spectra and AFC (page 139).

3. 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.
4. 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 6 triggers relative to range zero. This moves 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 lineof misplaced A/D samples. However, after a few moments of hunting, the burst pulse should appear on the display screen.
5. Fine tune the triggers so that the burst envelope is centered in the window, and adjust the amplitude zoom for a comfortable size display.
6. Isolate the clean center portion of the burst pulse 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 are derived.
7. 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 seriouslydegrade 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.
   See 5.2.1 Mb — Burst Pulse and AFC (page 95).

6.6 **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.

The **Ps** command plots the burst spectrum, designs the band pass filter, plots its frequency response, and helps align the AFC.

6.6.1 Interpreting Burst Spectra Plots

The following figures show an example of two plots from the **Ps** command. The display screen is divided into 2 areas:

- The top portion (single line) shows the present AFC level.
- The lower portion shows power spectrum plots of the burst pulse and/or the matched filter response.

The first figure is an example of a single filter response plot, while the second 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.

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 varies with signal strength but can be adjusted using the **Z** subcommand.

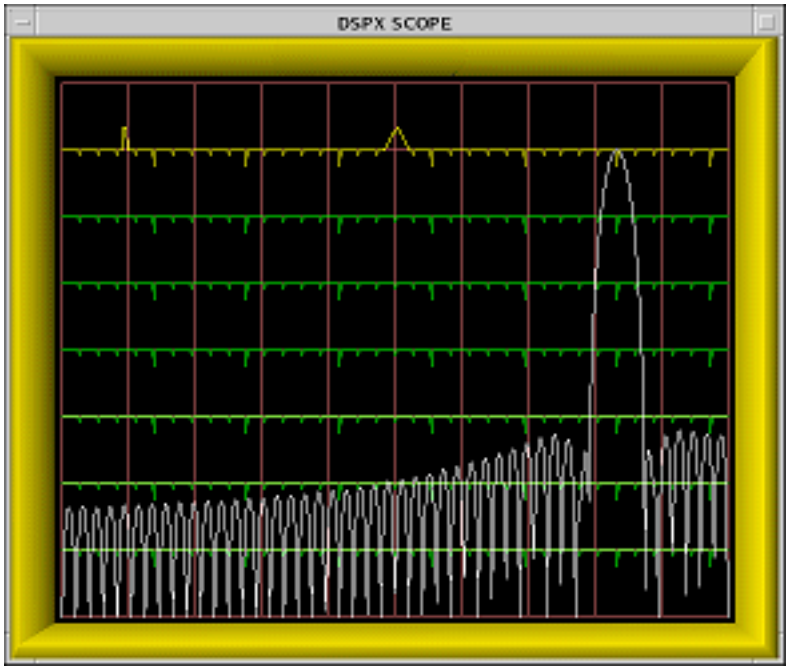


Figure 26 Example of a Filter With Excellent DC Rejection

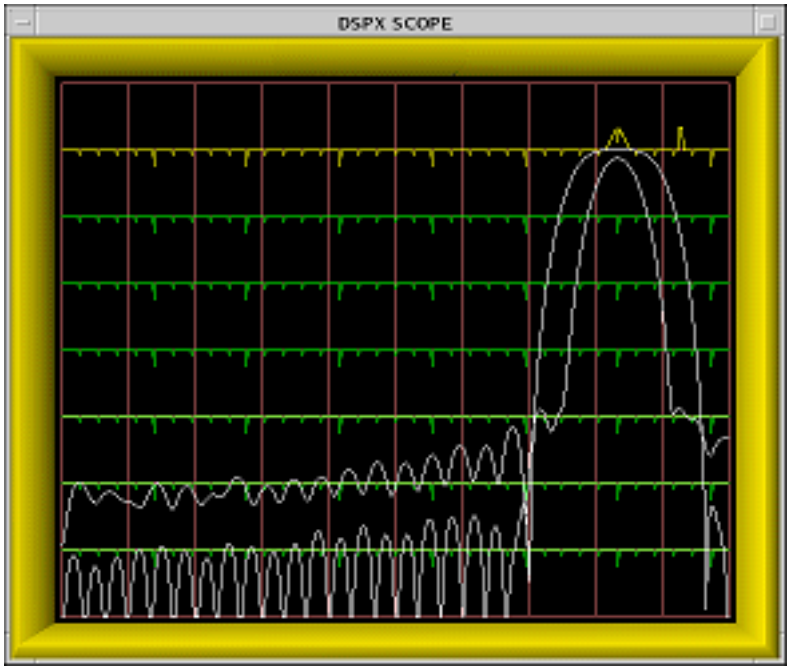


Figure 27 Example of a Filter With a Poorly Matched Filter

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 are always 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.

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.

Horizontal Axis

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.

Vertical Axis

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.

Horizontal Lines

The horizontal lines 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 in the example is from a system with an intermediate frequency of 30MHz. The left edge of the plot begins at DC, and the graph is centered on the sixth major tick, that is, 30 MHz.

Horizontal Line at the Top of the Plotting Area

The horizontal line at the top of the plotting area is marked with an upward pointing major and minor tick to 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 10 horizontal division lines visible 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 watch the AFC behavior in real-time.

Horizontal line at the Top of the Display

The horizontal line at the top of the display (above the spectra plot) indicates the 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 moves from the left edge to the right edge as the AFC varies from its minimum to maximum value. The minor tick traverses the screen at 10 times this rate.

6.6.2 **Ps** Subcommands

These subcommands change the design of the matched FIR filter, assist with calibration of the AFC loop, and alter the format of the display.

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
```

Table 34 **Ps** Subcommands

Command	Description
I/i	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 much improved by changing the FIR length by one or two taps. Experiment with this as you design your filter.
N/n & W/w	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 shifts the exact locations of the filter's zeros, and possibly improve the DC rejection.
U/u & D/d	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**.
=	Toggle MFC mode on and off with the **=** key. A warning is printed if the **Ps**command is exited while MFC is enabled, and you are given another chance to re-enable AFC.

Command	Description

|	To enter the AFC test submode, type the **|** key.
	The following list of keybindings are shown, and remain in effect until you exit the test mode 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 76`
	`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 AFC information is 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 are output instead of the normal formatted AFC value. The walking-ones test is useful as an oscilloscope diagnostic. To return to it at any time, type: **W**.
	Th3 test pattern shifts a single `1` down through the AFC word, making a transition approximately every 4ms. This helps ring out and test the wiring for digital AFC installations.
	Type `0` ... `9` or `A`... `O` to 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 a pattern, you can toggle a 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 in the pin-map table in the **Mb** menu; or whether it represents those pins directly.
	The printed hex test value is followed either by `Bits` or `Pins`.
	• When in `Pins` mode, the "0" key toggles `Pin-1`, the "1" key toggles `Pin-2`, and so on.
	The `Pins` mode is useful when you are doing the initial electrical tests of the wiring of each pin.
	• 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.
	After you have verified the pin wiring and created the **Mb** mapping table, then the `Bits` mode allows you to test the complete digital AFC interface.

Command	Description
#	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, when performing a radar sphere calibration. For coefficient definitions, see 7.2.1 FIR (Matched) Filter (page 170).
$	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. 1. Use the **I/i** and **N/n & W/w** subcommands to manually move to that starting point. 2. Type **$** to print a dialog line in which the search span length and width are chosen. You may keep the indicated values or type in new ones. The search begins when the spans are accepted. The search procedure may require a few seconds to a few minutes, depending on the length and width spans that are being scanned. During this time, a progress message is printed showing the length and width currently under examination. You may type **Q** to abort the search and retain the original filter settings. When the search completes normally, it prints "**Done**" and replaces the old filter settings with the best ones that could be found. In dual-receiver mode, the **$** command searches for a filter that minimizes the maximum width and DC offset at both receivers intermediate frequencies. The final filter is the one that has the best simultaneous performance at both IFs.
V/v	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	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.
< 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. Type **%** to toggle the plot of the FIR filters frequency response, and the printout of its DCGain. The plotted spectrum and printed power levels are always based on the sum of all input signals, and do not change with **%**.

6.6.3 TTY Information Lines Within **Ps**

The TTY information lines 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)
```

Table 35 **Ps** TTY Information Lines

Information Line	Description
`Navg`	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 updates more slowly.
`FIR`	The length of the impulse response of the matched FIR filter. See 6.5.3 TTY Information Lines In Pb (page 137).
`BW`	The 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`	The filter's 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 do not propagate into the synthesized `I` and `Q` values.
`Freq`	The mean frequency of the burst.
`Pwr`	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 (It would not be possible to compute the filter loss when the burst waveform is missing). See 6.6.4 Computing Filter Loss (page 147).

Information Line	Description
AFC	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 appears when RVP900 detects a special digital format. That is, when the back panel 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 ... +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 modes shown to the right of the numerical value(s), and can take 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. If 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.

6.6.4 Computing 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 detects 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 increases 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 ... 1.8 dB range, depending on the FIR length and other design criteria.

For 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 by the shape of the modulation envelope. For a pulse of length T_o to have unit-energy it must have an amplitude of : $\frac{1}{\sqrt{T_0}}$

By centering this pulse at time zero the power spectrum is easily computed using a real-valued integral:

$$S(f) = \left(\int_{\frac{T_0}{2}}^{\frac{T_0}{2}} \frac{1}{\sqrt{T_0}} \cos(2(\pi f t)t) dt \right)^2 = \frac{\sin^2(\pi f t)}{\pi^2 f^2 T_0}$$

where **f** is the frequency in Hz. This is the familiar "synch" function, whose main frequency lobe extends from $1/T_o$ to $1/T_o$ Hz, 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 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:

$$dB_{loss} = -10\log_{10}\left(\int_{\frac{-B_w}{2}}^{\frac{B_w}{2}} S(f) df \right)$$

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 makes 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 assume 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 becomes more complicated than the above example in which we integrated an idealized filter response over an idealized power spectrum.

Let $\widehat{B}(f)$ 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 $\widehat{C}(f)$ 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 $\widehat{B}(f)$ according to its sampled measurement of the transmitted waveform; however it can calculate $\widehat{C}(f)$ internally based on an idealized sinusoid. The reported filter loss is then:

$$dB_{loss} = -10\log_{10}\left(\frac{\int |H(f)|^2 \widehat{B}(f)\,df}{\int \widehat{B}(f)\,df} \div \frac{\int |H(f)|^2 \widehat{C}(f)\,df}{\int \widehat{C}(f)\,df}\right)$$

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 $\widehat{C}(f)$ have a constant value and need only be computed once. They serve to normalize the $\widehat{B}(f)$ integrals in such a way that the filter loss evaluates to 0 dB when 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 when 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 should never look like a pure tone as long as RVP900 can "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 is underestimated because the Tx spectrum appears to be narrower than it really is.

The RVP900 calculation of digital filter loss is 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 ... 27MHz and 27 MHz ... 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 to `single`, and disconnecting the secondary burst input to the IFDR.

6.6.5 Adjusting Plot Burst Spectra and AFC

1. Make sure you have successfully captured the burst pulse with the **Pb** command.
2. Type the **Ps** command.
3. Use the space bar to display the burst spectrum plot by itself, and use the **Z** key to shift the entire graph into view.
   The plot shows the frequency content of the transmitted pulse.
   a. Check that the plot shows a clean main power lobe centered at the receivers intermediate frequency.
   b. Check the spectrum for spurious harmonics, excessive width, and other out-of-band noise.
   c. Adjustment the transmitter to give a sharper main lobe or reduce spurious noise.

4. Begin designing the matched FIR filter.
   Use the space bar 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.

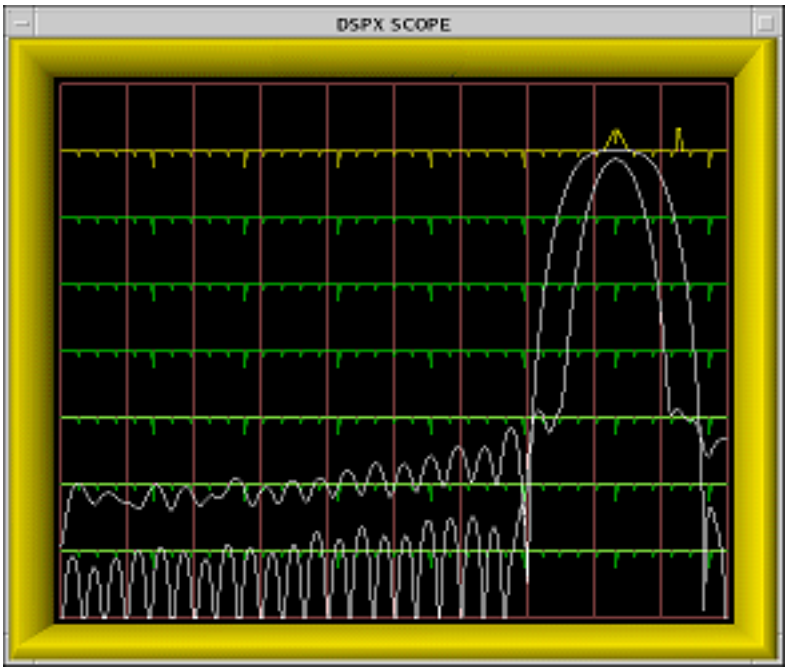


Figure 28 Example of a Poorly Matched Filter

5. 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 reduces 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.
   - 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.

6. Optimize the performance of the FIR filter.
   The filter should not pass any frequencies that do not contain useful information from the original transmitted pulse. If anything, choose a filter whose width is slightly narrower than the bursts spectral width.
   The previous figure 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.
   You can optimize the FIR filter manually or automatically:

   - **Manual Method**
     Defining a nearly optimal filter requires a few minutes of hunting with the I, W, and N keys. Each time you press any of these keys, RVP900 designs a new FIR filter from scratch, and displays the results.
     Even though you must still control two degrees of freedom (length and bandwidth), the RVP900 design calculations perform 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 6.6.2 Ps Subcommands (page 142).

Example

The offset error of the IFDR A/D converter is at most 10mV, that is, -27 dBm into its 50 Ω input.

To achieve 90 dB of dynamic range below the converters +8 dBm saturation level, 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.

The following figure 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 previously that has a true zero at DC. In general, a poor filter can be converted into a "nearly" good filter by making only incremental changes to the impulse response length and/or desired bandwidth.

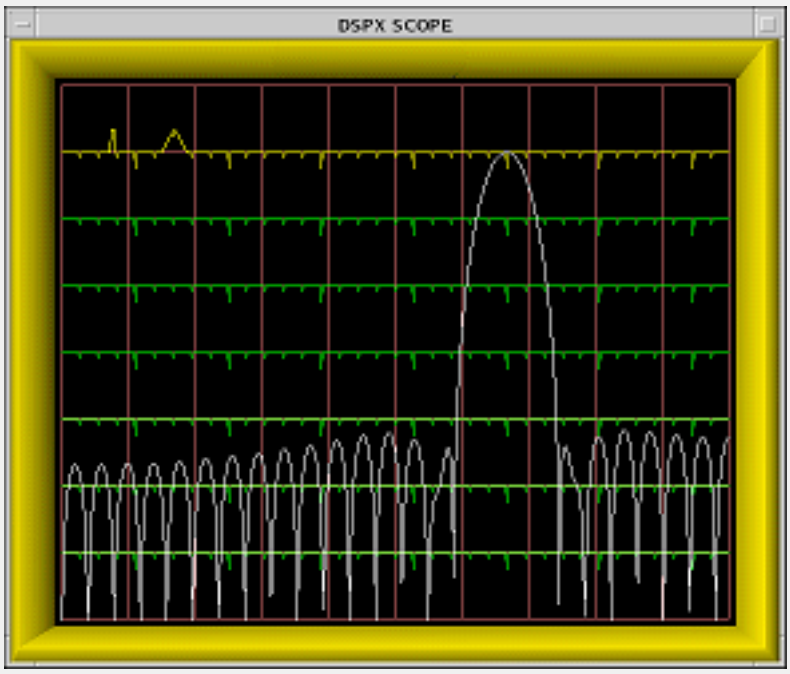


Figure 29 Example of a Filter With Poor DC Rejection

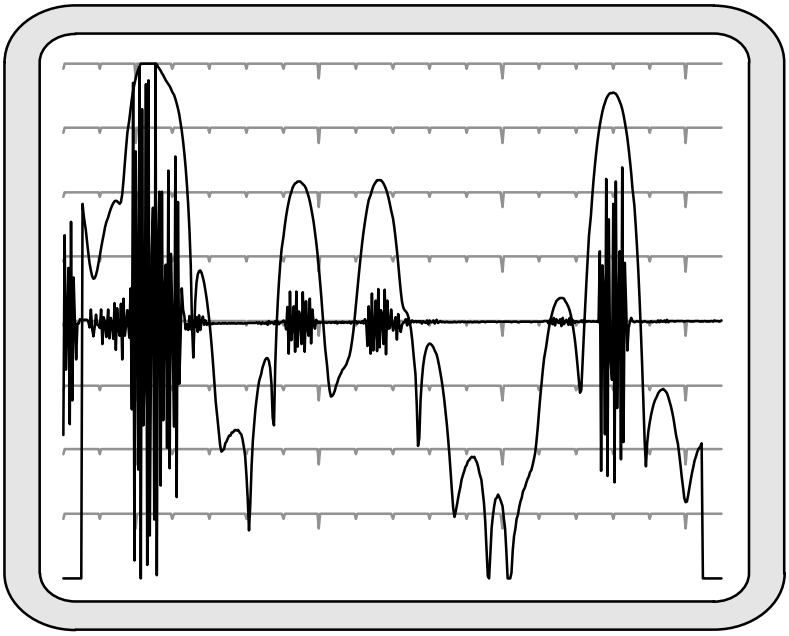
6.7 **Pr** — Plot Receiver Waveforms

The **Pb** and **Ps** commands are used to analyze the signal that is applied to the **Burst-In** connector of the IFDR module.

You must now use the **Pr** command to check the 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.

6.7.1 Interpreting Receiver Waveform Plots

The following figure shows an example of a plot from the **Pr** command. 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.



1) Environment Canada – Aldergrove BC

Figure 30 Example of Combined IF Sample and LOG Plot

The IF samples are plotted on a linear scale as signed quantities, with 0 appearing at the center line of the scope. Any DC offset present in the A/D converter is not removed, and is 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 is much less than this worst case value; but 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 when 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 the previous figure 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 $10(^{-18/20}) = 0.13$, 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 shifts 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. The LOG plot gives a very detailed view of received power versus range. Of course, successive LOG points are 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 0, you 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; the latter half of the burst pulse is visible when the plot begins at range 0.

Noisy High Resolution Pr Spectrum

The following figure shows a **Pr** display with 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).
- 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.

In this example, the time span is set to 50 µsec, with a 1 m antenna attached to the IF input so that a broad range of signals (radio stations, electrical noise, and so on) would be detected.

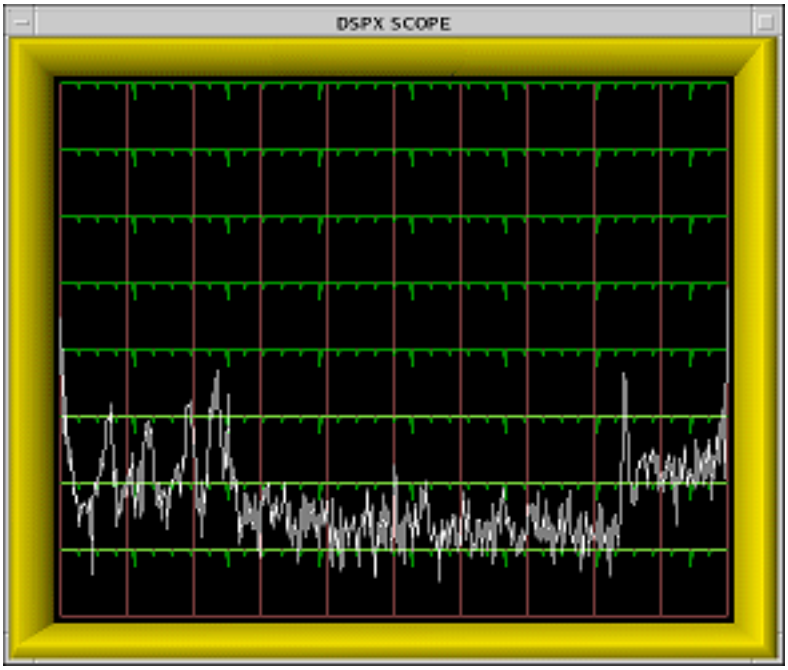


Figure 31 Example of a Noisy High Resolution Pr Spectrum

The purpose of the **Pr** power spectrum is to check for spurious interference in the IF signal from the radar receiver. View the spectrum with the transmitter turned off, and with the starting range moved out so that the burst samples are not mixed 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 be quite fine to make it possible to discern any interfering frequencies with some detail.

The **Pr** spectrum plot shows a 0 Hz peak from any DC offset in the A/D converter, and is consistent with how the DC offset is presented in the **Pr** sample plot.

Both 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.

6.7.2 Available Subcommands In **Pr**

These subcommands change the start time and span of the IF sampling window, and alter the format of the display.

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
```

Table 36 **Pr** Subcommands

Command	Description
L/l & R/r	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	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	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	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 the IFDR IF-Input sources.

6.7.3 TTY Information Lines In **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
```

Table 37 **Pr** TTY Information Lines

Information Line	Description
Zoom	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 is not be quite this high. Use larger zoom levels to see the weaker samples more clearly. You may zoom in powers of two up to x128.
Navg	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 updates more slowly.
Start	The starting time of the IF sample window relative to range zero. The time is shown both in microseconds and in kilometers.
Span	Time span of the IF sample window in microseconds.
IF	Which IF-Input sources (SMA edge connectors) is providing the data being plotted.
Total	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	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	The RMS power within the passband of the FIR filter using only the raw IF samples in the 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.

The **Total**, **Filtered**, and **MidSamp** values represent true RMS power (that is, variance), and not merely a sum-of-squares. Any DC offset present in the A/D converter does not affect these power levels.

6.8 **Pa** — Plot Tx Waveform Ambiguity

RVP900 can make radar observations using compressed pulse waveforms. This opens many new opportunities for using low-power solid-state transmitters that employ very long pulse lengths (20 ... 80 µsec). 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 required to make good use of these long transmit pulses is also much more complex. To help with this, RVP900 provides the **Pa** (plot ambiguity) command in which compressed transmit waveforms can be designed, studied, and optimized. In the **Pa**, plots you can experiment with different waveform designs, try out bandwidth and pulse width options, and examine and optimize the range/time sidelobes of your waveform.

6.8.1 Interpreting Ambiguity Plots

The following figure 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 µsec, bandwidth is 3 MHz, PSL is -61.2 dB and ISL is -50.8 dB, Doppler shift ± 50 KHz.

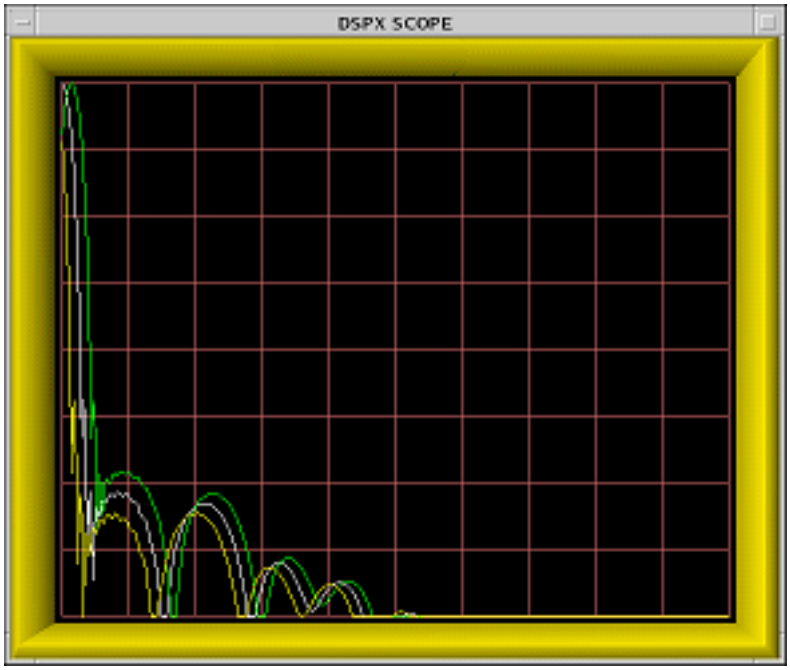


Figure 32 Ambiguity Diagram of a Compressed Tx Pulse

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.

Frequency, Phase, and Amplitude of a Compressed Tx Pulse

The following figure shows an alternate form of **Pa** plot of the same Tx waveform. The horizontal axis again represents time, but now spans the entire duration of the pulse. Three plots are drawn, and 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.
- 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.

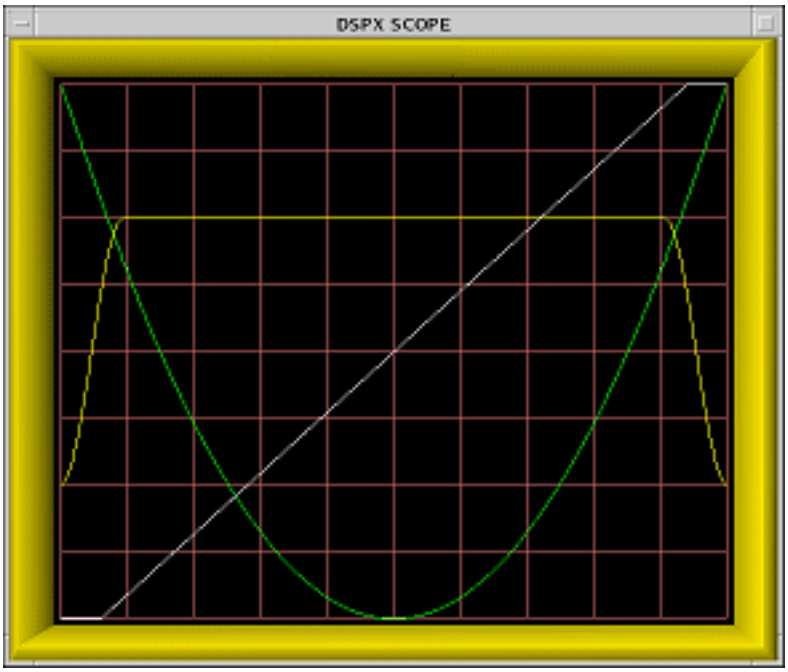


Figure 33 Frequency, Phase and Amplitude of a Compressed Tx Pulse

The figure shows 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 label `Non-Linear FM`.

The central chirp is contained within a 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.

6.8.2 Available Subcommands In **Pa**

Pa subcommands change the bandwidth, pulse width, and shaping parameters of the transmit waveform, and alter the format of the display:

```
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
```

Table 38 **Pa** Subcommands

Command	Description
S/s & L/l	The shorter and longer commands decrease or increase the time duration (that is, pulse width) of the transmitted pulse. The lower case commands shift in 0.05 µsec steps. The upper case commands use 1 µsec steps.
N/n & W/w	The narrower and wider commands decrease or increase the bandwidth of the transmitted pulse. The lower case commands shift in 10KHz steps. The upper case commands use 200 KHz steps.
1/2/3	Type one of these numbers to choose which of the 3 waveform tuning parameters are altered by the **D** and **U** commands.
D/d & U/u	Decrease or increase the waveform tuning parameter that has been chosen by the most recent **1**, **2**, or **3** command. The lower case commands shift in 0.001 (dimensionless) steps. The upper case commands use 0.05 steps. Present values of all 3 parameters are printed each time a Down/Up key is pressed, for example: `Tuning parameters: 0.9000 1.0000 0.1770` See 5.2.6.1 Tx Synthesis Options (page 123).

Command	Description
V/v	Shows how the overall compressed pulse Tx/Rx system responds to the effects of target Doppler shift. You can introduce frequency shifts as large as 100 KHz in order to see their effect on the Range/Time side lobes. The **Pa** plot shows 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** side lobe plot is 80dB. Usually this gives plenty of room to examine the properties of the waveform. For very wide dynamic range pulses, you can shift the plot up/down in 10dB steps using this command.
$	Designs an optimal compressed waveform. For a given pulse width and bandwidth of the Tx waveform, you can try 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.

$ Command

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 can 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
```

requests that the parameter be searched using 200 evenly spaced grid points lying between 0.9000 ... 0.9500 inclusive.

After you have entered all 3 parameter spans, RVP900 begins searching for the optimum waveform. Progress messages are printed on the TTY, and the plot updates each time a better waveform is discovered. Follow the messages to check if the search is converging.

The process normally runs to completion on its own. If the search takes too long, or you change your mind about which intervals to search, type **Q** to exit immediately.

When prompted with: `Keep this waveform? [Y]`:

- Type **Y** to keep the optimized tuning parameters that were just discovered and overwrite the starting values.

- Type **N** to discard the search results and return to the original settings.

6.8.3 TTY Information Lines In **Ps**

The TTY information lines resemble:

```
BW :3.40MHz  PW:29.99usec PSL:61.2dB ISL:51.3dB
TxLoss:0.5dB RxLoss:2.4dB
```

Table 39 **Ps** TTY Information Lines

Information Line	Description
BW	Bandwidth of the Tx waveform (MHz).
PW	Pulse width (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).

Information Line	Description
`RxLoss`	The `RxLoss` is a measure of how much information is discarded 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: $dB_{loss} = -10\log_{10}\left(\frac{\left\|\int T(t)Rtdt\right\|^2}{\left\|\int\left\|T(t)\right\|^2 dt\left\|Rtdt\right\|^2\right\|^2}\right)$ 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 0 dB. 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, and so on) of the ideal matched filter is used to achieve the desired sidelobe levels. The 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 when a window is applied.

Given a compressed transmit waveform, RVP900 designs the appropriate mismatch Rx filter automatically, using an optimized Blackman window in all cases. Developers can also access the internal APIs directly to design any desired transmit waveform along with the associated FIR filter to receive it.

6.8.4 Bench Testing Compressed Waveforms

Once the Tx waveform has been designed, you can inject it into the IFDR for testing with the **Pr** command.

This verifies that the analog waveform is generated properly checks that the matched filtering on the RVP900/Rx card can deconvolve the compressed information.

1. Connect the Channel #1 or Channel #2 output of the RVP900/Tx card to the IF-Input of the IFDR.
2. In the **Mz** menu, use the RVP900/Tx channel that has been configured for waveform synthesis.
   See 5.2.7 Mz — Transmissions and Modulations (page 126).

3. Set the zero offset of the transmitter pulse in the **Mt<n>** menu to, for example, 50 µsec. This shifts the waveform out in range so the **Pr** plot can show it.
   See 5.2.6 Mt<n> — Triggers for Pulsewidth n (page 117).
   The following example shows a **Pr** plot of a 40 µsec, 5 MHz optimized waveform generated by the RVP900/Tx card and fed into the IFDR.
   In the example, the ideal Tx waveform has a Peak Sidelobe Level (PSL) of - 76.7dB and an Integrated Sidelobe Level (ISL) of -62.3 dB. 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 5 MHz signal bandwidth.
   The sampled analog waveform is not quite identical to the ideal waveform.

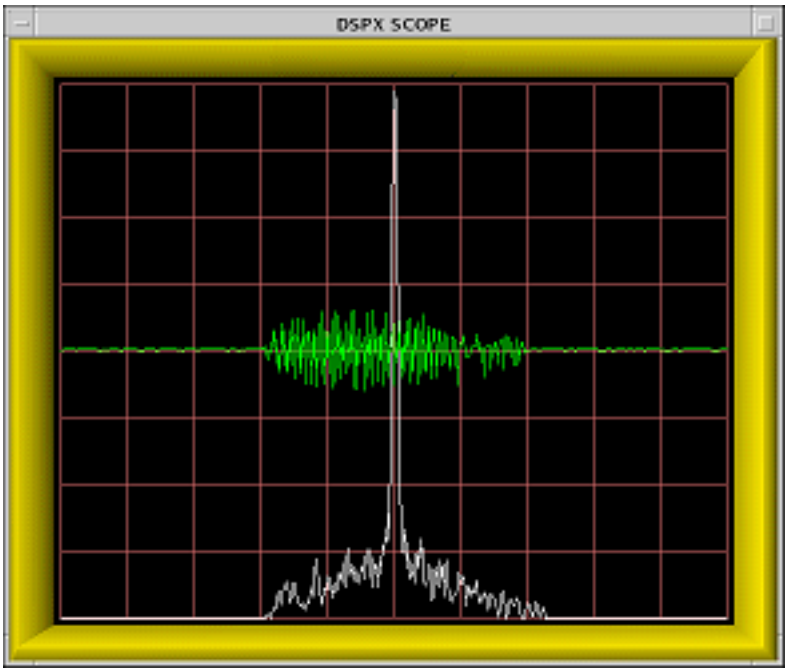


Figure 34 IFDR Sampling of Optimized Compressed Tx Waveform

4. If needed, use the **Ps** plotting command to examine the ideal transmit spectrum and received spectrum of compressed pulses.
   The following example shows a 60 MHz, 40 µsec linear FM pulse having a bandwidth of 2 MHz. The energy in the pulse is sharply contained within and uniformly distributed over the 2 MHz frequency interval centered on the IF carrier.
   This shows the ability of synthesized transmit waveforms to remain cleanly in their allocated bounds.

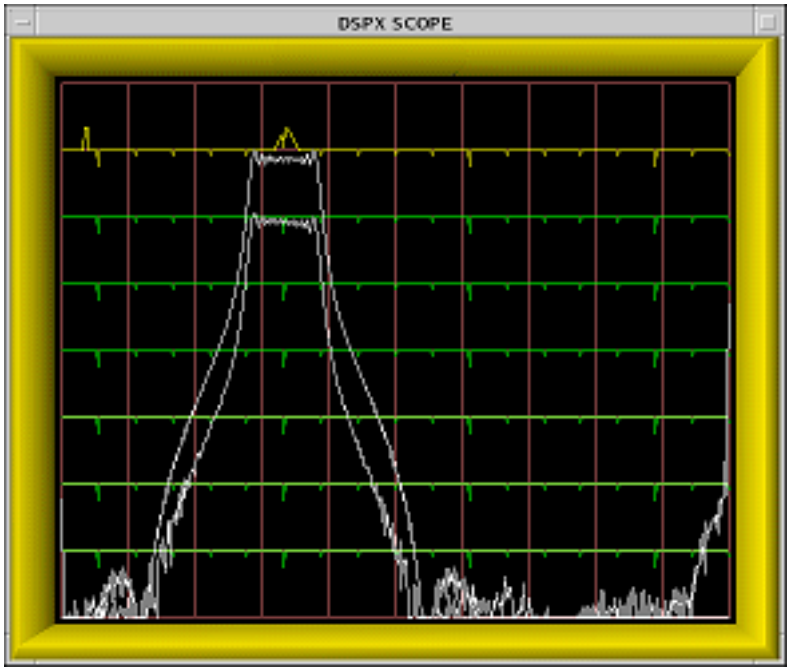


Figure 35 Ideal and Actual Linear -FM Spectrum Displayed in Ps Plot

7. Processing Algorithms

7.1 RVP Algorithm Overview

For information on dual polarization processing algorithms, see *IRIS and RDA Dual Polarization User Guide*.

The discussion on processing algorithms implemented within the RVP900 signal processor is confined to the mathematical description of these algorithms.

It is often 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 here are defined in terms of the operations performed on the **s**'s, rather than the **I** and **Q**'s. The use of the complex terms leads to a more concise mathematical expression of the signal processing techniques being used.

During operation, the complex arithmetic is broken down into its real-valued component parts in order to be computed by RVP900. For example, the complex product:

```
s = W × 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 use three additional 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:

$$|s| = \sqrt{\mathrm{Re}\,al\{s\}^2 + Imag\{s\}^2}$$

The signed (CCW positive) angle made between the positive real axis and the above vector is:

$$\measuredangle = Arg\{s\} = \arctan\left[\frac{Imag\{s\}}{\mathrm{Re}\,al\{s\}}\right]$$

where this angle lies between - π and + π and the signs of `Real{}` and `Imag{}` determine the proper quadrant. Note that this angle is real, and is uniquely defined as long as `|s|` is non-zero. When `|s| = 0`, `Arg{s}` is undefined.

Finally, the complex conjugate of `s` is the value obtained by negating the imaginary part of the number, that is:

$$s^* = \mathrm{Re}\,al\{s\} - jImage\{s\}$$

`Arg{s*} = -Arg{s}`.

More Information

- RVP900 Weather Signal Processing (page 42)

7.1.1 Measured Quantities

The following table summarizes the quantities that are measured and computed by RVP900.

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. There is no need to include a range subscript in this data notation.

Table 40 Algebraic Quantities Within the RVP900 Processor

Quantity	Description	Type
`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`

Quantity	Description	Type
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`

7.1.2 IF Signal Conversion Process

The following figure shows the overall process by which the RVP900 converts the IF signal into corrected reflectivity, velocity, and width, including:

- 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
- Algorithms for ambiguity resolution (dual PRF, dual PRT, random phase)
- Calibration and testing

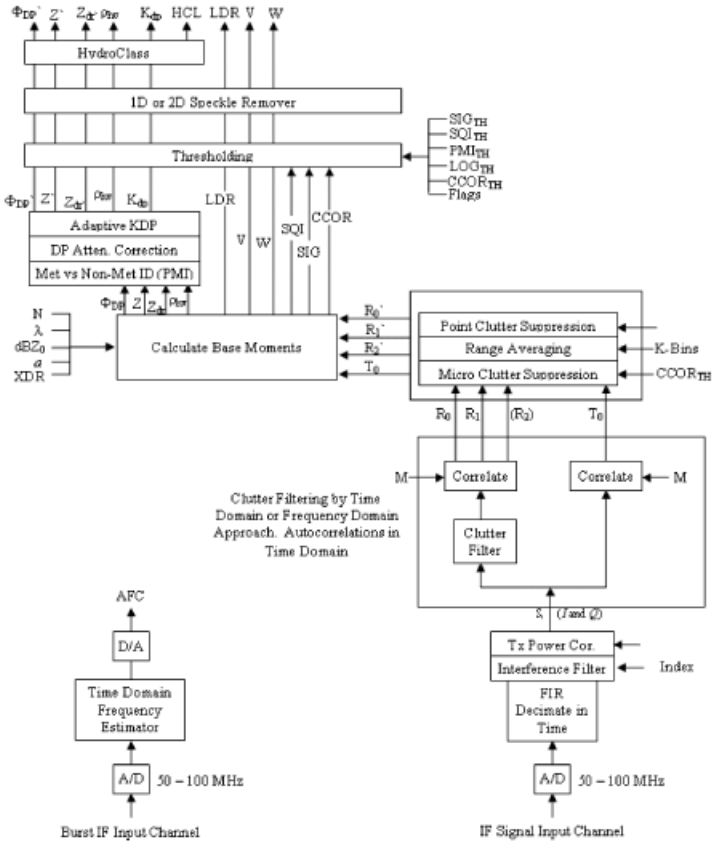


Figure 36 RVP 900Processing Flow Diagram

7.2 IF Signal Processing

The starting point for RVP900 computations are the instantaneous IF-receiver samples $\mathbf{p_n}$ and, the instantaneous burst-pulse or **COHO** reference samples $\mathbf{b_n}$.

These data are available at a very high sampling rate (up to 100 MHz), which makes possible the digital implementation of functions that are traditionally performed by discrete components in an analog receiver.

More Information

- RVP901 IFDR IF to I and Q Processing (page 25)

7.2.1 FIR (Matched) Filter

RVP900 implements a digital version of the matched filter that is found in the traditional analog radar receiver.

For information on designing the digital matched filter using the graphical user interface, see 6.6 Ps — Plot Burst Spectra and AFC (page 139).

The filter length (number of taps), center frequency, and bandwidth are adjustable. The design procedure computes two sets of filter coefficients f_n^i and f_n^q such that the instantaneous quadrature samples at a given bin are:

$$I = \sum_{N=0}^{N-1} f_n^i \times p_n, Q = \sum_{N=0}^{N-1} f_n^q \times p_n$$

where `N` is the length of the filter. The input samples p_n are centered on the range bin to which the (`I`, `Q`) pair is assigned. Note that some p_n may overlap among adjacent bins. That is, the filter length may be greater than the bin spacing. The 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 RVP900 using a flexible FGPA that can perform billions of sums of products per second.

Reference Phase

The reference phase for each transmitted pulse is computed using the same two FIR sums, except that b_n is substituted for p_n.

For magnetron systems, the `N` b_n samples are centered on the transmitted burst.

For klystron systems, the `N` b_n samples may be obtained from the burst pulse (recommended) or from the CW COHO. If the klystron is phase modulated by an external phase shifter (instead of the IFDR digital transmitter board), the samples should be from the modulated COHO.

Coefficients

The f_n^i coefficients are computed as:

$$f_n^i = l_n \times \sin\left[\frac{\pi}{4} + 2\pi\frac{f_{IF}}{f_{SAMP}}\left(n - \frac{N-1}{2}\right)\right],\ n = 0 \ \ldots\ N-1$$

where f_{IF} is the radar intermediate frequency, f_{SAMP} is the IFDR 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_n^q 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 chosen to have a valuable overall symmetry property when `n` is replaced with `(N-1)-n`, that is, the sequence is reversed:

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

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

$$f_{(N-1)-n}^i = f_n^q$$

The coefficients needed to compute **I** are the reversal of the **i** coefficients needed to compute **Q**. That is, if you know f_n^i, then you also know f_n^q.

7.2.2 RVP900 Receiver Modes

Use the **Mc** menu to select the receiver configuration that defines IF processing for your site.

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

Table 41 Receiver Mode Configuration Options

Mode	Description
Mode-0: Standard Single Channel	This is the most common mode 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	Reserved for future development.
Mode-2: Extra Wide Dynamic Range	Radars with high performance front-end LNAs can preserve the full benefit of that investment by running 2 separate IF signals to the Primary (**HiGain**) and Secondary (**LoGain**) IFDR inputs. A nominal channel separation of 25 dB ... 30 dB can be used to achieve an overall dynamic range of up to 110 dB.
Mode-3: Dual-Rx From Two IF Inputs	Standard dual polarization mode. The **H** and **V** channels are fed to 2 separate IF inputs using the same intermediate frequency.

7.2.2.1 Wide Dynamic Range Mode-2

When RVP900 is used as an extended dynamic range receiver, you must make some important decisions regarding setting up the RF/IF levels that drive the IFDR.

Separation Value

First, decide on 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 outside of these absolute limits. valuable receiver dynamic range is lost.

- The absolute minimum channel separation 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, for 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 (-78 … +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 is 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 channel separation 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 0. We begin using one as soon as the other has begun to saturate.

There can be a large difference between the absolute minimum and maximum signal level separations. You must consider additional criteria 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 operate very close to their noise floors. If the separation is too high, 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 … 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 … 63 dB, which is sufficient for good IFDR transitions and also leads to receiver components that are practical to build.

Receiver

Once you have chosen a separation value, you must consider how to build the receiver to achieve this value.

The basic receiver is 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 must 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. An amplifier with an output saturation value of +10 dBm ... +15 dBm should be used.

7.2.3 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.

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 ... 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. The AFC feedback loop typically has a time constant of several seconds or more.

Most 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 . This prevents the introduction of transients into the burst analysis algorithms each time the pulse width changes.

More Information

- Mb — Burst Pulse and AFC (page 95)
- Digital AFC (DAFC) (page 51)

7.2.4 Burst Pulse Tracking

RVP900 can 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, and so on) are tracked and nulled out during normal operation; so that fixed targets remain fixed in range, and clean Tx phase measurements are always available on every pulse.

The burst pulse tracker feedback loop changes the trigger timing in response to the measured position of the burst. Timing changes are generally made only when 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 is done immediately. Also, there is an approximately 5 ms interruption in the normal trigger sequence when 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 also include a combined hunt mode that tracks down a missing burst pulse when we are uncertain of both its time and frequency. This coarse-tuning mode is 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 it is either:

- 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 must change in order to bring the center of the pulse back to the center of the window.
- 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 in those intervals then the Burst Pulse Tracker and AFC loops are 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 procedure may take several seconds to complete. The RVP902 host computer interface remains 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.

More Information

‣ Hunt for Burst Pulse (BPHUNT) (page 308)

7.2.5 Interference Filter

The interference filter is an optional processing step that can be applied to the raw (**I**,**Q**) samples from the FIR filter chips. The filter can 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 3 most recent pulses:

$P_{n-2},\ P_{n-1},\ and\ \ P_n$

where:

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

If the 3 pulse powers have the property that:

$\left|P_{n-1} - P_{n-2}\right| < C_1 \quad and \quad \left|P_{n-1} - P_{n-2}\right| > C_2$

(Alg. 1)[1)]

then ($\mathbf{I_n}$, $\mathbf{Q_n}$) is replaced by ($\mathbf{I_{n-1}}$, $\mathbf{Q_{n-1}}$). Here $\mathbf{C_1}$ and $\mathbf{C_2}$ 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 $\mathbf{C_1} = \mathbf{C_2}$; but 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 2 clear pulses in between. Interference that tends to arrive in bursts are not rejected.

Two variations on the fundamental algorithm are also defined. The **CFGINTF** command allows you to choose algorithms to use, and to tune the 2 threshold constants. You may also do this from the **Mp** setup menu. See 8.24 Configure Interference Filter (CFGINTF) (page 306) and 5.2.4 Mp — Processing Options (page 108).

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

(Alg. 2)

$\left|P_{n-1} - P_{n-2}\right| < C_1 \quad and \quad \left|P_n - LinAvg\left(P_{n-1},\ P_{n-2}\right)\right| > C_2$

(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. When power levels are compared in the algorithms, any power that is less than the noise level is first set equal to that noise level. This makes the filters more robust and properly tunable, so that interference is more successfully rejected on top of blank receiver noise.

1) *The JMA internal specification for Interference Filter algorithm for use on Chitose airport Doppler weather radar is the basis for* ***Alg.1***

Optimum values for C_1 and C_2 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 16 dB stronger than the weather. The interference arrived in isolated single pulses with a probability of 2%.

The algorithm performance is summarized in the first three columns of the following table, 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 42 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. The following table maps the rates of false alarms to mean relative changes of reflectivity (in dB).

Table 43 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%	..
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 does not do 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 let the missed percentage fall where it may.

The following table shows the results obtained if the interference dominates by 26 db. To summarize the numerical results in the table:

- The **Missed** rates of **Alg.1** and **Alg.2** are identical, but the **False** rate of **Alg.1** is much higher. **Alg.1** 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} alone. 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.

Optimum tuning depends 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.

Table 44 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%

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%

Note that we can 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 "clean" weather virtually untouched. That is, we would have a safe filter that only removes fairly strong interference. You could leave such a filter running at all times without too much worry about side effects.

7.2.6 Large-Signal Linearization

RVP900 can 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 it is still possible to deduce information about the signal.

The following figure shows 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 has been extended by 5 dB.

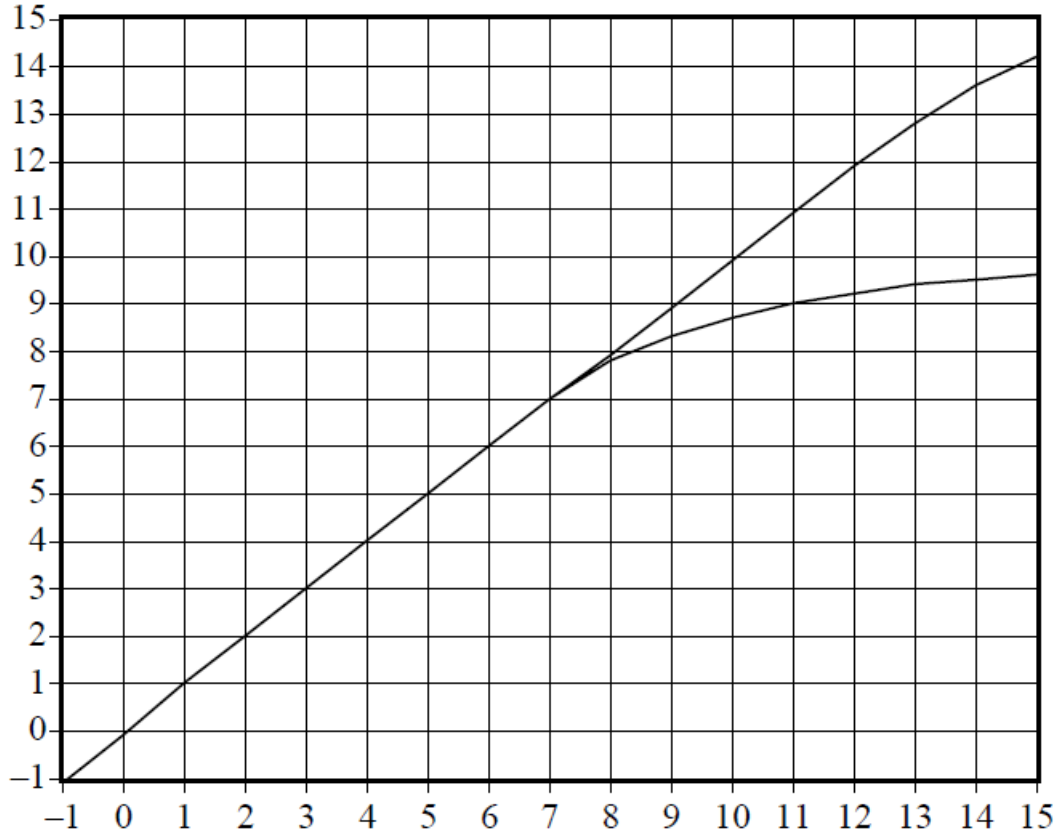


Figure 37 Linearization of Saturated Signals Above + 8 dBm

7.2.7 Correction for Tx Power Fluctuations

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 5.2.4 Mp — Processing Options (page 108).

A short time constant settles faster, but is not be as thorough in removing amplitude variations (since the mean itself varies). Longer time constants do a better job, but require a few seconds before valid data is available when the transmitter is first turned on. The default value of 70 gives excellent results in almost all cases.

When RVP900 enters a new internal processing mode (time series, FFT, PPP, and so on), 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. Valid corrected data are produced even when RVP900 alternates rapidly between different data acquisition tasks, for example, in a multi-function ASCOPE display. The additional pipeline delay does not affect the high-speed performance when 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. The amplitude correction feature "gets out of the way" when receiver-only tests are being performed.

The maximum applied correction 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 easily contained within this interval (it is typically less than 0.3 dB).

Instantaneous amplitude correction is a unique feature of RVP900 digital receiver. Bench tests with a signal generator reveal that an amplitude modulated waveform having 2.0 dB of pulse-to-pulse variation is reduced to less than 0.02 dB RMS of (`I`,`Q`) variation after applying the amplitude correction.

7.3 Time Series Signal Processing

Radar time series data (also called linear "video" or `I` and `Q`) processing is done to obtain the meteorologically significant moment parameters: reflectivity, total power, velocity, width, signal quality index, clutter power correction, and optional polarization variables.

The time series synthesized by the FIR filter consist of an array of complex numbers:

$$s_m = [I_m + jQ_m] \; for \quad m = 1, 2, 3, \ldots, M$$

where `j` is - $1^{1/2}$.

The time series are the starting point for all calculations performed in RVP900.

The following figure shows the components of the Doppler spectrum, that is, white noise, weather signal, and ground clutter. Other target types such as sea clutter, birds, insects, aircraft, surface traffic, second trip echo, and so on may also be present.

- The top part shows the `I` and `Q` values for a simulated time series using the **Ascope** utility.
- The bottom part shows an example of a Doppler power spectrum for the time series shown in the upper part of the figure.

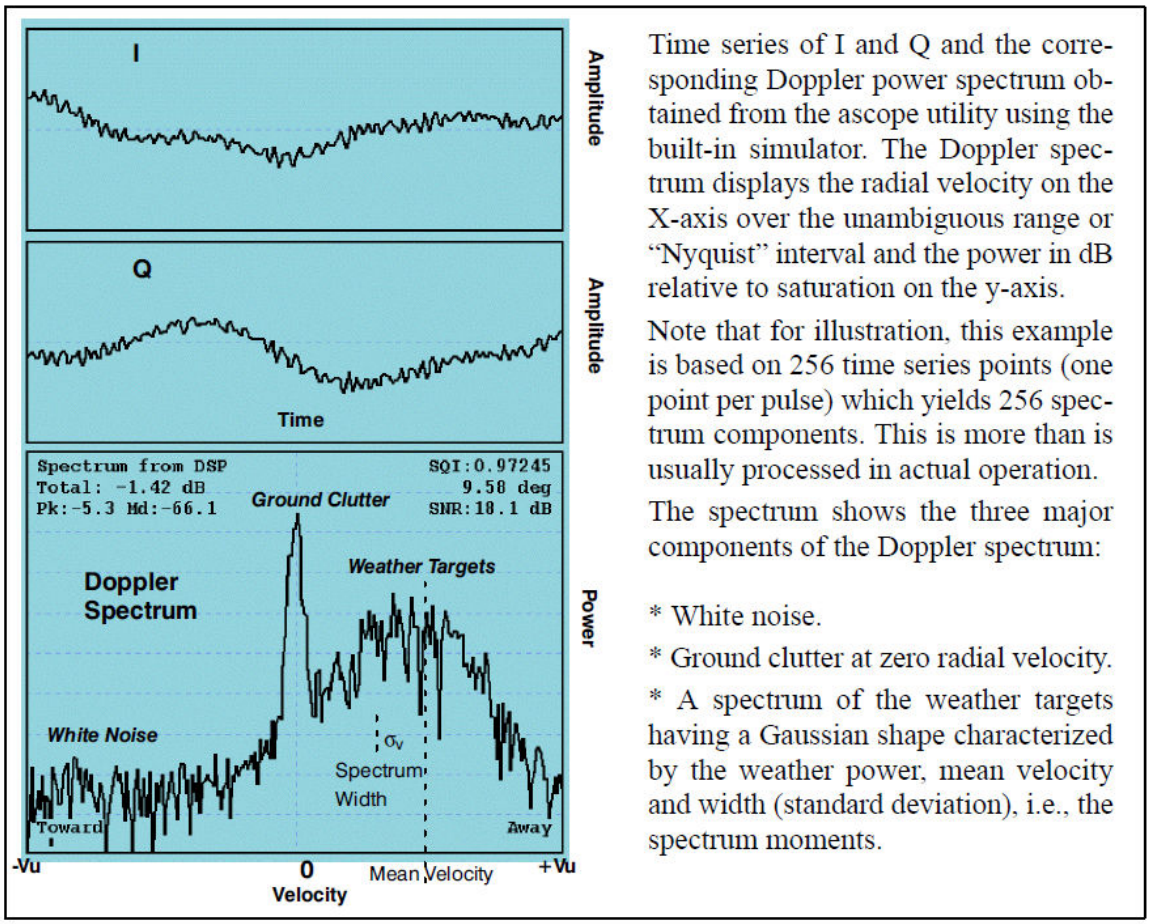


Figure 38 Time Series and Doppler Power Spectrum Example

Time Series Signal Processing Categories

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

Support Time Series Processing Modes

RVP900 supports "major modes" or processing modes to process the time series, including:

- 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 and Random Phase Mode or RPHASE are frequency domain approaches 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° of scanning) to determine which ranges are likely contaminated by second trip echo. This was developed to support a particular US WSR88D legacy requirement.

7.3.1 Frequency Domain Processing- Doppler Power Spectrum

The Doppler power spectrum (also known as Doppler spectrum) is the easiest way to visualize the meteorological information content of the time series.

The Doppler power spectrum is obtained by taking the magnitude squared of the input time series, that is, for a continuous time series,

$$S(\omega) = |f\{s(t)\}|^2$$

Here `S` denotes the power spectrum as a function of frequency ω, and `f` denotes the Fourier transform of the continuous complex time series `s(t)`. The Doppler power spectrum is real-valued since it is the magnitude squared of the complex Fourier transform of `s(t)`.

In practice, a pulsed radar operates with discrete rather than continuous time series. That is, there is an `I` and `Q` value for each range bin for each pulse. In this case we use the discrete Fourier transform or DFT to calculate the discrete power spectrum.

When we have `2n` input time series samples (for example, 16, 32, 64, 128, ...), we use the fast Fourier transform algorithm (FFT), which is significantly faster than the full DFT.

The DFT has the form:

$$S_k = |DFT_k\{w_m s_m\}|^2 = \left| \sum_{m=0}^{M} w_m s_m m^{e^{-j\left(2\left(\frac{\pi}{M}\right)\right)mk}} \right|^2$$

Typically a weighting function or "window" wm is applied to the input time series s_m to mitigate the effect of the DFT assumption of periodic time series. RVP900 supports different windows such as the Hamming, Blackman, Von Han, exact Blackman, and the rectangular window for which all spectral components are weighted equally.

The following figure shows the typical form of a spectrum window to illustrate how the edge points of the time series are de-emphasized and the center points are over emphasized. The dashed line corresponds to the rectangular window. The gain of the window is set to preserve the total power.

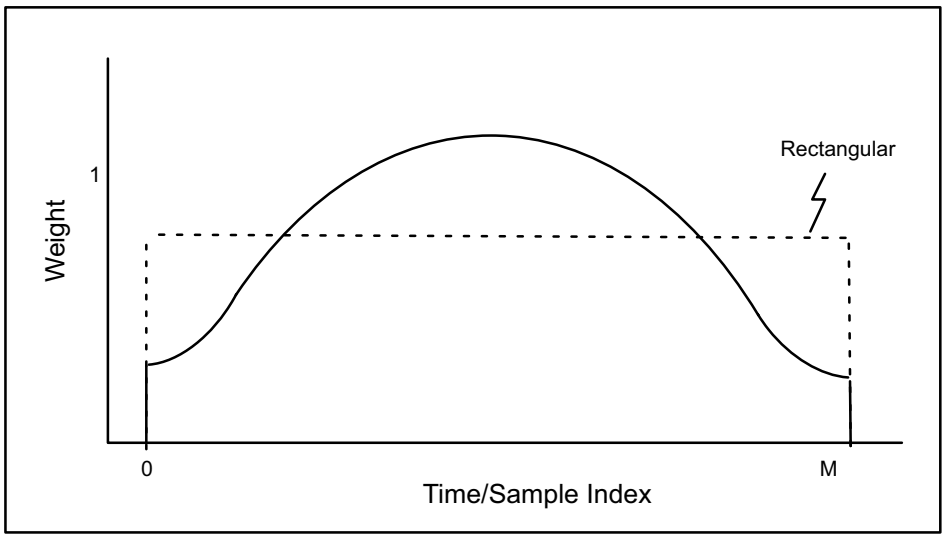


Figure 39 Typical Form of Time Series Window

Although 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 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 the following figure 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.

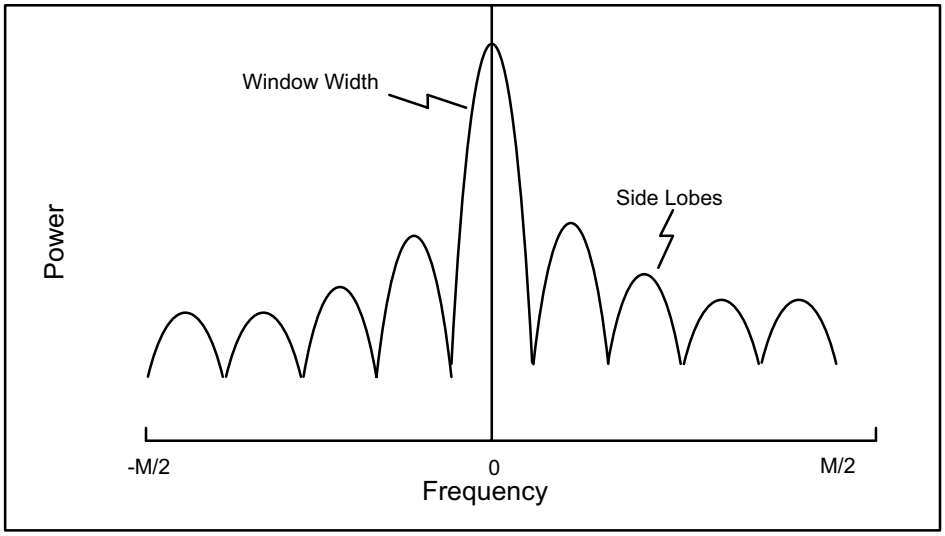


Figure 40 Impulse Response of Typical Window

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.

The best approach is to use the least aggressive window possible in order to contain the clutter power that is present. That is, an adaptive approach is the best.

7.3.2 Autocorrelation for Moment Estimations

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. 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.

Table 45 Time Domain Calculation of Autocorrelations and Corresponding Physical Models

Parameter and Definition	Physical Model
$T_0 = \frac{1}{M}\sum_{n=1}^{M} s_n{}^* s_n$	$g^r g^t (S + C) + N$
$R_0 = \frac{1}{M}\sum_{n=1}^{M} s'_n{}^* s'_n$	$g^r g^t S + N$
$R_1 = \frac{1}{M-1}\sum_{n=1}^{M-1} s'_n{}^* s'_{n+1}$	$g^r g^t S + e^{j\pi V' - \frac{\pi^2 W^2}{2}}$
$R_2 = \frac{1}{M-2}\sum_{n=1}^{M-2} s'_n{}^* s'_{n+2}$	$g^r g^t S + e^{j\pi V' - 2\pi^2 W^2}$

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. `gr` and `gt` 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 are ignored in the discussion of these. T_0 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_0 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 Figure 38 (page 182). `W` is the spectrum 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 R_1 and R_2 lags are complex, this yields, effectively, 5 equations in 5 unknowns using the constraint provided by the argument of R_1. This closed system of equations can be solved for the unknowns which is the basis for calculating the moments from the autocorrelations.

7.3.3 Ray Synchronization on Angle Boundaries

The exact value of **M** that is used for each time average is generally the sample size that is selected by the **SOPRM** command. See 8.4 Setup Operating Parameters (SOPRM) (page 235).

However, when RVP900 aligns its processed rays to AZ/EL angle boundaries (See 8.16 Load Antenna Synchronization Table (LSYNC) (page 293)) the 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** is never greater than the **SOPRM** Sample Size, but it may sometimes be less. For example, if 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 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 are 50 pulses used for each ray (not 80) and those pulses exactly fill the scanning angle sector.

7.3.4 Clutter Filtering Approaches

The following table shows how each major mode implements clutter filtering.

Table 46 Clutter Filtering in Major Modes

Mode	Clutter Filtering
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	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.

Some earlier vintages of radar signal processors 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 are significantly attenuated by the filter.

High-speed processors such as RVP900 offer 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 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 frequency domain filters available in the RVP900 are configured using the **mf** setup command (See 5.2.3 Mf — Clutter Filters (page 106)):

- 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

7.3.4.1 Fixed Width Clutter Filters

This filter, shown in the following figure, 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.

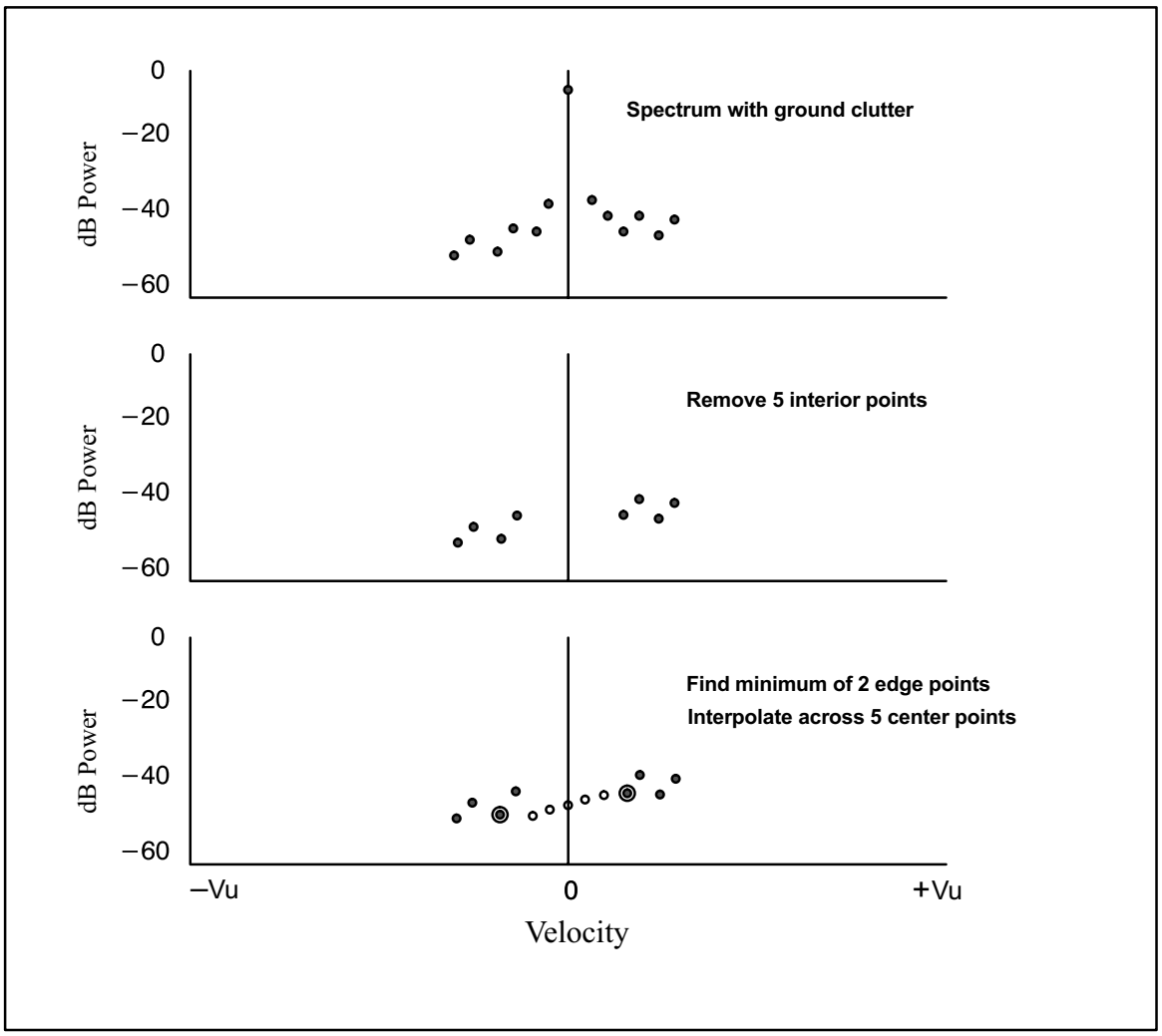


Figure 41 Fixed Width Clutter Filter Examples

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.

7.3.4.2 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.

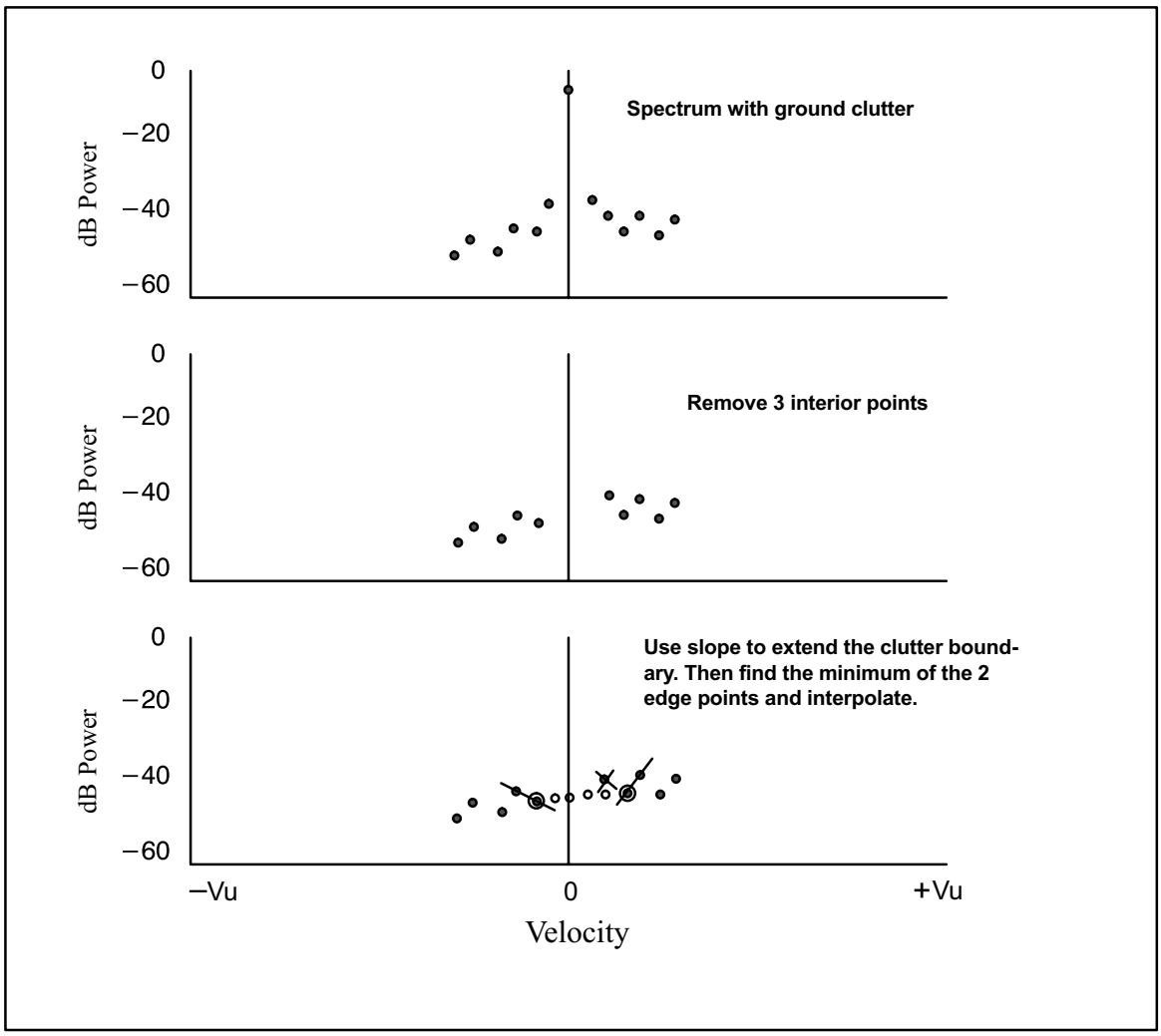


Figure 42 Variable Width Clutter Filter

In the previous figure, 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 "hunts". 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.

7.3.4.3 GMAP

GMAP processing 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, 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 the following assumptions about the model for clutter, weather, and noise:

- The spectrum width of the weather signal is greater than that of the clutter. This is a fundamental assumption required for 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 following figure shows the steps used to implement the GMAP approach.

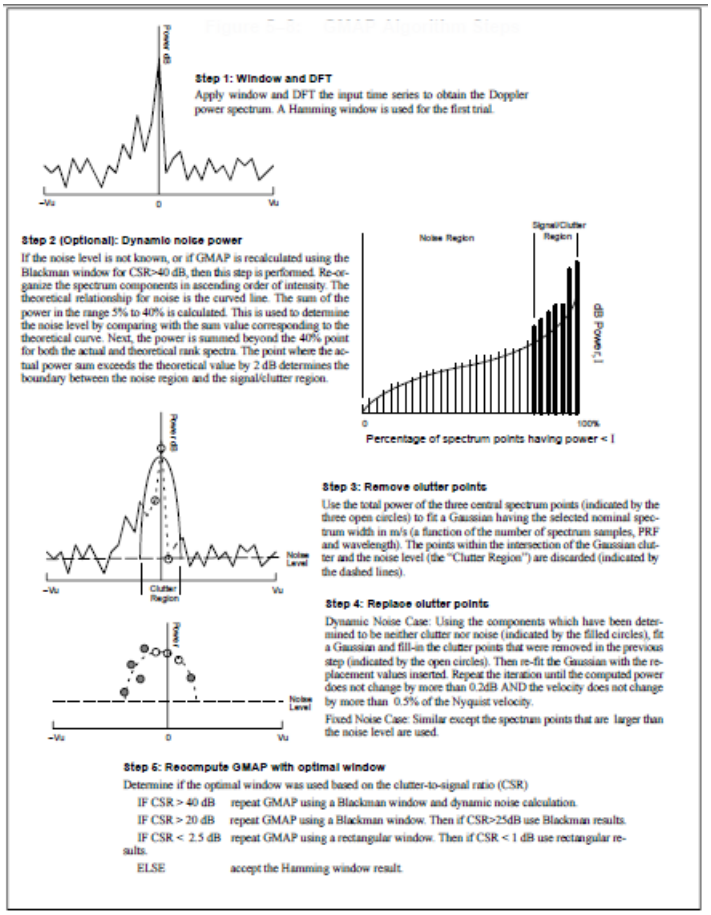


Figure 43 GMAP Algorithm Steps

Table 47 GMAP Algorithm Steps

GMAP Step	Description
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 7.3.1 Frequency Domain Processing- Doppler Power Spectrum (page 183), 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.

GMAP Step	Description
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 differs 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. 50 dB clutter targets 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 the figure, 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 the figure. 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 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

GMAP Step	Description
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 so 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 is 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 behavior, 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 is often incorrect. Since GMAP determines the no-filter case automatically and then processes accordingly, a clutter map is not required.

GMAP Step	Description
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 GMAP 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 "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 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 should 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 ... 0.5.

A good practice is to make a scan on a clear day while using ascope or other utility and observe the width of the clutter for your scan rates. You must turnoff the clutter filtering to do this (select `filter 0` for the all pass filter).

Implementation Example

GMAP has undergone extensive evaluation for use in the US WSR88D ORDA network upgrade (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.

The following figure 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 on 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 number 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

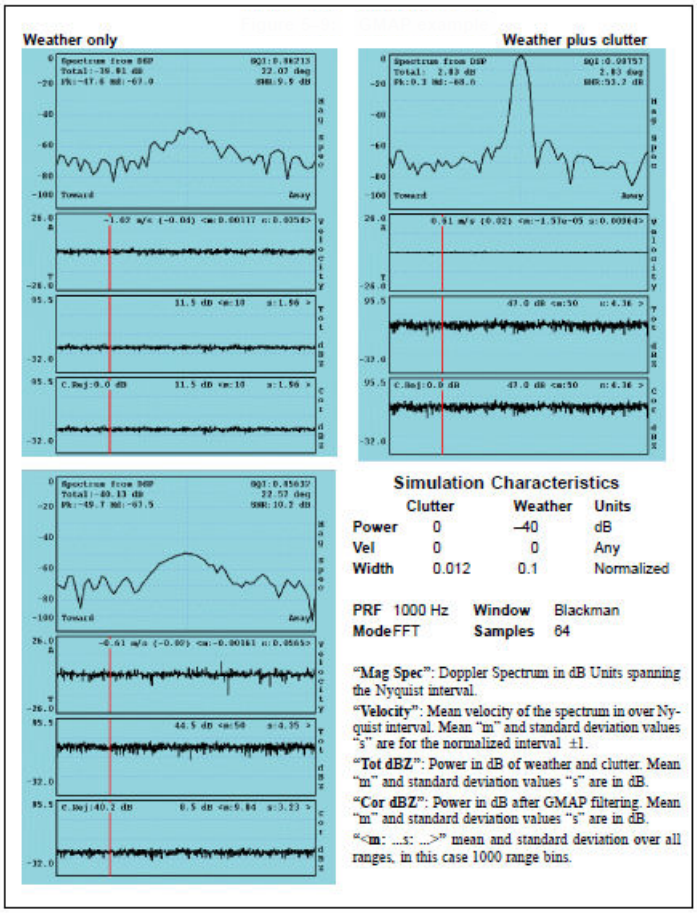


Figure 44 GMAP Example

7.4 Autocorrelation R(n) Processing

7.4.1 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:

$$\left[\left(R_0(r) > TCM * R_0(r - Offset)\right) \quad and \quad \left(R_0(r) > TCM * R_0(r + Offset)\right)\right]$$

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:

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

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.

7.4.2 Performing 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 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.

More Information

- Range Averaging and Clutter Microsuppression (page 45)

7.4.3 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 a dB version of the familiar radar equation for distributed targets. The relationship between the measured autocorrelation function, the received signal and the noise can be expressed as:

$$T_0 = g^t g^r S + N$$

where g^t and g^r represent the transmitter and receiver gains, S is the average back scattered power from the targets and N is the measured average noise power. Neglecting attenuation and the contribution of ground clutter (for the moment), the radar equation can be written as.

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

where **C** is the radar constant and **ro** is a reference range which we later set to 1 km. This is identical to the first three terms of the dB version of the equation with the definition that:

$$Z_0 = \frac{Cr_0^2 N}{g^r g^t} = Cr_0^2 I_0 \quad where \ I_0 = \frac{N}{g^r g^t}$$

where:

Z_0
: 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.

I_o
: Measured noise power at IF with appropriate calibration for the system gain.

The measurement of **I_o** is based on the measurement of the system noise at the time of calibration. However, if the receiver gain changes 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**, that is:

$$Z = CSr^2 = \left[\frac{Cr_0^2 N}{g^r g^t}\right]\left[\frac{r^2}{r_0^2}\right]\left[\frac{kT_0 - kN}{kN}\right]$$

Thus the **k**'s cancel to give us the same result for **Z**. This makes the approach robust to system gain fluctuations. As long as the system sensitivity (noise figure) does not change, then the system does not require re-calibration.

Calibrating RVP900 involves defining the radar constant **C** and measuring the value of **I_o**. See 7.6 Reflectivity Calibration (page 211).

The following table shows the individual terms in the dB form of the equation.

Table 48 Terms in the dB Equation Format

Term	Term Name	Description
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.
2nd Term	**dBZ_o**: Calibration Reflectivity (see discussion above)	**dBZ_o** is the minimum detectable dBZ at a reference range **r_o =1 km**

Term	Term Name	Description
3rd Term	`20 log r` : Range Normalization	This term is the range normalization expressed in dB form. $\left[\frac{r}{r_0}\right]^2$
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. See 7.4.7 Clutter Correction (CCOR threshold) (page 203).

7.4.3.1 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 $Z = CSr^2 = \left[\frac{Cr_0^2N}{g^r g^t}\right]\left[\frac{r^2}{r_0^2}\right]\left[\frac{T_0 - N}{N}\right]$ 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 the new radar equation becomes:

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

In this equation, the `dBZ0` is the term $\left[\frac{Cr_0^2Nc}{g^r g^t}\right]\left[\frac{Nr}{Nc}\right]$

The dBZ0 fed into RVP900 is the basic $\left[\frac{Cr_0^2Nc}{g^r g^t}\right]$, while the processor modifies the value by the ratio `Nr/Nc`. Values read out by the **GPARM** command, and so on are the modified value.

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 is not applied, and you get the message **"Cannot enable Z0 noise correction because calibration is missing"** in the *rvp* log file.

7.4.4 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,

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

λ is the radar wavelength, τ_s is the sampling time (1/PRF). θ_1 is constrained to be on the interval [-π, π]. When $\theta_1 = \pm\pi$, then $V = \pm V_u$ where the unambiguous velocity is ,

$$V_u = \frac{\lambda}{4\tau_s}$$

If the absolute value of the true velocity of the scatterers is greater than V_u, then the velocity calculated by 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:

$$V_u + \Delta V$$

then the velocity calculated by the RVP900 is:

$$-V_u + \Delta V$$

which is $2V_u$ away from the true mean velocity.

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

7.4.5 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

Use the **SOPRM** command to select the approach.

7.4.5.1 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 [-π, π]. For example, a variance of $\Pi^2/25$ would be obtained from a Gaussian spectrum having a standard deviation equal to one fifth of the total width of the plotted spectral distribution. For scientific purposes, the spectrum width (standard deviation) is more physically meaningful than the variance, since it scales linearly with the severity of wind shear and turbulence. For these reasons, the width **W** is output by RVP900:

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

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 .

7.4.5.2 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 defined as in 7.4.5.1 R0, R1 Width Algorithm (page 202).

7.4.6 Signal Quality Index (SQI threshold)

RVP900 can 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 ... 0.5.

7.4.7 Clutter Correction (CCOR threshold)

The following content is based on legacy implementation in which R_1 , and R_2 are defined in terms of PPP (pulse-pair-processing).
Current major modes use FFT (DFT) processing instead.

In addition to calculating the R_0, R_1 and optional R_2 autocorrelation terms, which are based on filtered time series data, RVP900 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.

7.4.7.1 R0, R1 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.

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.

7.4.7.2 R0, R1 Clutter Correction Using Clutter Signal and Noise Power Calculations

In this case there is enough information to compute the clutter signal and noise power independently. The algorithm for CCOR is:

$$CCOR_{est} = 10\log\left[\frac{S}{C+S}\right] = 10\log\left[\frac{1}{CSR+1}\right]$$

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.

7.4.8 Weather Signal Power (SIG Threshold)

The **SIG** parameter provides an estimate of the weather signal-to-noise ratio in dB for thresholding.

The **SIG** calculation is different depending on the whether the optional R_2 autocorrelation is computed.

Table 49 SIG Calculation

Autocorrelation	Equation	Description
R_0, R_1	$SIG = 10\log\left[\frac{T_0 - N}{N}\right] + CCOR$	Represents the SNR after the removal of clutter. The **CCOR** value is that described for R_0, R_1 in 7.4.7 Clutter Correction (CCOR threshold) (page 203).
R_0, R_1 ,R_2	$SIG = 10\log\left[\frac{2\pi S}{R_0 - 2\pi S}\right] + CCOR$	The SIG is computed based on the SNR. The signal power **S** is determined as described in 7.4.7 Clutter Correction (CCOR threshold) (page 203).

7.4.9 (Signal+Noise) to Noise Ratio (LOG Threshold)

The **LOG** threshold parameter is 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 + 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)*

7.5 Thresholding

RVP900 can accept or reject incoming data based on derived properties of the signals themselves.

Typically, rejected data are not displayed by the user's software, making for clean weather presentations.

7.5.1 Threshold Qualifiers

For data quality control, each RVP900 output parameter can be qualified, that is, accepted or rejected for output, based on threshold criteria:

Table 50 Threshold Quality 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

Each qualification criterion can be switched on and off independently, and the threshold levels (for example, $\mathbf{SQI}_{thresh}$) can each be set independently. Also, 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 following table shows the threshold qualifiers.

Table 51 Threshold Qualifiers

Threshold Qualifier	Description	Default Value
LOG	A measure of signal strength that is usually used for the thresholding of reflectivity data.	0.75 dB
SQI	Typically used for velocity and width thresholding since it is a measure of the coherency. It is a number between 0 ... 1 (dimensionless), where 0 is perfect white noise and 1 is a pure tone (perfect Doppler signal).	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. Threshold values lower than the default (more negative) reject fewer clutter bins. Threshold values closer to 0 reject more clutter bins.	-18 dB.
SIG	Typically used only for thresholding the spectrum width to assure that the signal power is strong enough for an accurate width measurement. If R_2 processing is used, this can usually be reduced to 5 dB for width thresholding.	5 dB
PMI	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.	0.45

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

Table 52 Default Threshold Combinations

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

7.5.2 Adjusting Threshold Qualifiers

When 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.

The following table shows some tips for optimizing the levels for the default criteria.

Table 53 Adjusting Threshold Qualifiers

Qualifier	Adjustment
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 are 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 effects 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.

Secondary SQI Threshold

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 in a slope and offset from the primary user value. See 5.2.3 Mf — Clutter Filters (page 106).

The secondary threshold is more permissive (lower valued), and is traditionally used to qualify LOG data only in the Random Phase processing mode.

The secondary **SQI** threshold is applied uniformly in all processing modes when 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.

7.5.3 Speckle Filter Processing

A speckle filter is a final pass over each output ray, in which isolated, single bins of velocity, width, or intensity are removed.

There are two speckle removers in 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.

This eliminates single pixel speckles, which allows the thresholds to be reduced for greater sensitivity with fewer false alarms (speckles).

Both 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. The benefits of using a speckle filter are:

- Displays look "cleaner" to observers
- Thresholds can be set slightly more sensitive without increasing the number of noise pixels

The 1D and 2D speckle filters are enabled using the **soprm** command, **Input 2** (see 8.4 Setup Operating Parameters (SOPRM) (page 235). If both are turned on, the 1D filter is applied first.

More Information

- Speckle Filter (page 46)

7.5.3.1 1D Speckle Filter

A ray is the basic azimuth unit of RVP900 (for example, 1°) 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". .

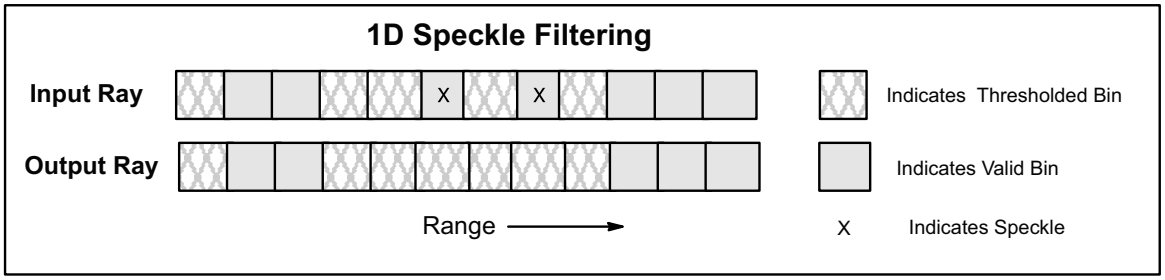


Figure 45 1D Speckle Filtering Algorithm

There are two independent 1D speckle removers:

- One for the reflectivity data (`dBT`, `dBZ`, `ZDR`, and `LDR` when using dual-pol).
- One for the Doppler data (`V`, `W`, `PhiDP`, `RhoHV`, and `KDP` when using dual-pol).

Each 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`.

7.5.3.2 2D 3x3 Speckle Filter

The 2D 3x3 speckle filter performs data filling of "missing speckles" as well as eliminating isolated speckle bins.

The 2D 3x3 speckle filter examines 3 adjacent range bins from 3 successive rays to assign a value to the center point. For each output point, its 8 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 are as follows:

Table 54 2D 3x3 Speckle Filter Rules

	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

The following figure shows some examples.

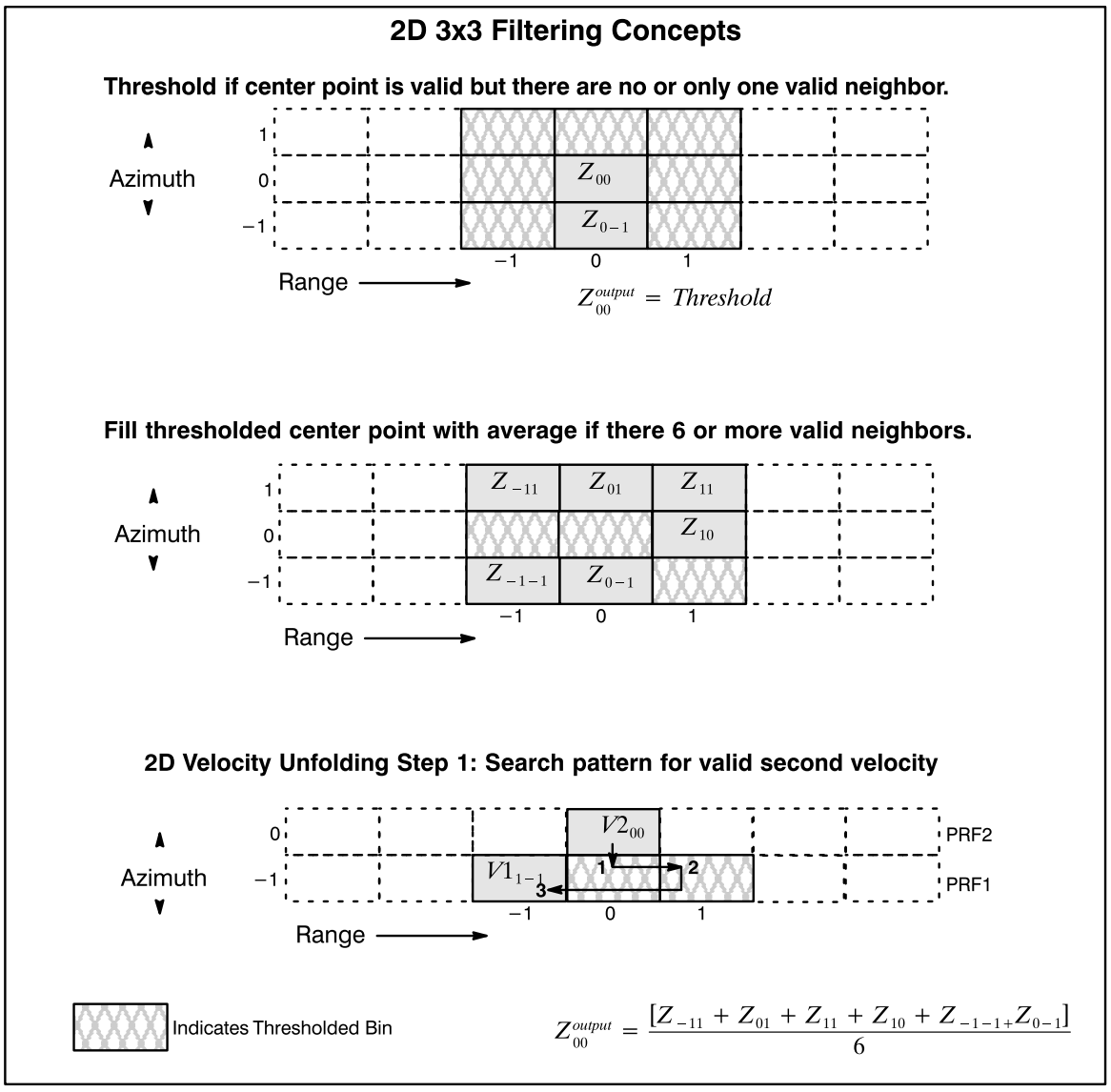


Figure 46 2D 3x3 FilteringConcepts

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.

7.5.3.2.1 Dual-PRF Unfolding

Dual-PRF velocity unfolding is computed within the 3x3 filter when 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 Figure 46 (page 210). This larger selection of alternate-PRF bins makes it more likely that the algorithm finds the pairs of Low/High PRF data that are required for unfolding.
2. The unfolded velocities are subjected to standard 3x3 filtering.

7.5.3.2.2 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 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.

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 is also be filled-in using the nearest neighbor. The 2D filter mitigates much of the damage that is caused at the 2nd trip seam to make a nearly seamless display.

7.6 Reflectivity Calibration

The calculation of reflectivity described in 7.4.3 Reflectivity (page 198). Here we look at its derivation.

You can use the **Zauto** utility to perform the calibration. See *IRIS and RDA Utilities Guide*.

7.6.1 Plot Method for Calibration of Io

This approach generates the curve (red) that determines the value of I_o.

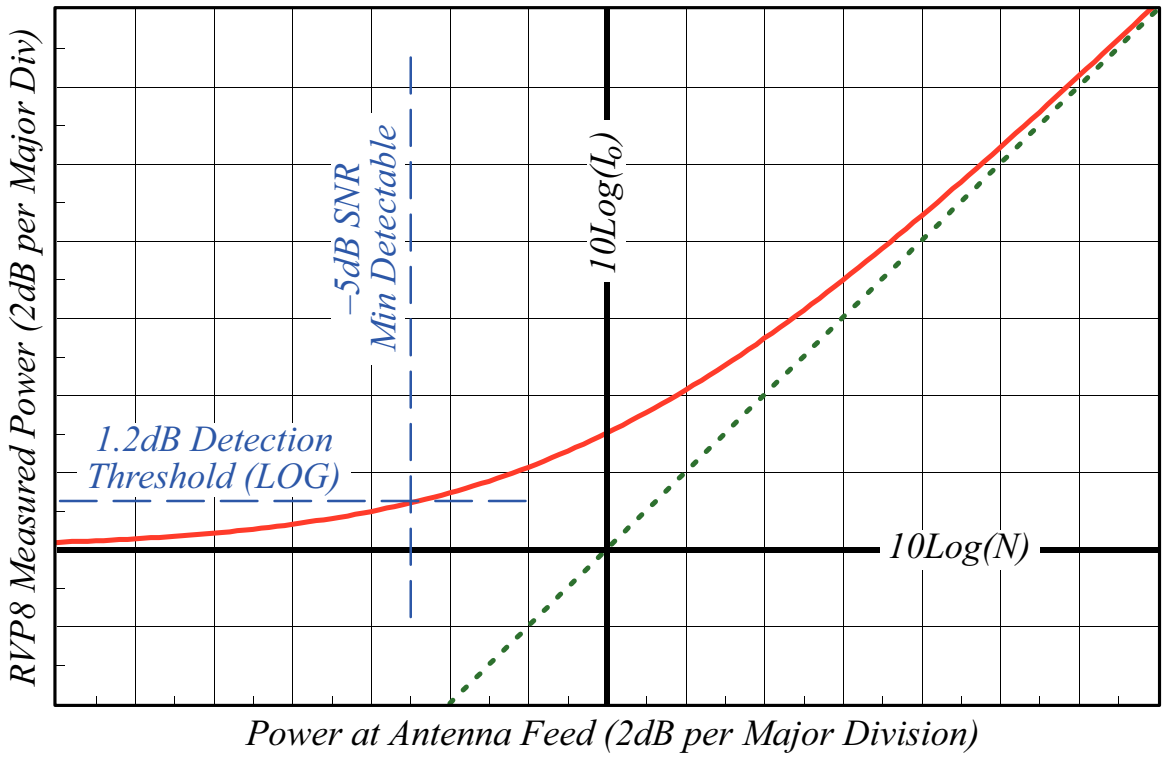


Figure 47 Model Intensity Curve - Power at Antenna Feed (2dB per Major Division)

The 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 versus the inserted power at the antenna feed, similar to that in shown in the figure.

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.

To understand why this geometric construction yields the value of I_o, Let G_{dB} represent the overall gain of the RF and IF components leading up to 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:

$$\text{(Red)} \quad 10\log_{10}(P_{OUT}) = GdB + 10\log_{10}(P_{IN} + I_0)$$
$$\text{(Green)} \quad 10\log_{10}(P_{OUT}) = GdB + 10\log_{10}(P_{IN})$$

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 0:

$$10\log_{10}(N) = GdB + 10\log_{10}(I_0)$$

Intersecting this measured noise level with the green straight line gives:

$$GdB + 10\log_{10}(I_0) = GdB + 10\log_{10}(P_{IN})$$

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 produces 0 dB SNR, that is, signal power equal to noise power. Do not confuse this with the minimum detectable power P_{MDS} which typically is several dB lower, depending on processor settings. In the above example, a 1.2 dB **LOG** detection threshold is shown (horizontal blue line) for the received signal. If RVP900 applies sufficient range and time averaging so that thermal noise alone produces very few false alarms above 1.2 dB, then P_{MDS} are a 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 the figure. 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.

CAUTION! Verify with the radar manufacturer that no damage can occur to the signal generator if the transmitter is running during the calibration.

1. Raise the antenna up a few degrees to avoid ground thermal noise
2. Insert signals at steps of 5 or 10 dB over the entire range of the system
3. Draw the plot shown in the previous figure.
   You can use fine resolution steps at the ends of the scale to observer the details of the roll off.

   For IRIS users, you can do this in the **Zauto** utility.

4. Tune the frequency of the signal generator using the setup command **pr**, and displaying the received signal spectrum.
   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. Use the `Log of Measured Noise Level (Word 6)` from **GPARM**.
   This procedure averages many samples together.
5. Turn it all the way down and make one more sample to measure the noise level **N**.
   You can obtain I_o from the intercept of the horizontal line at **N** and the straight line fit to the linear portion of the curve.

6. Correct the value for losses.
   See 7.6.3 Treatment of Losses in Calibration (page 215).

7.6.2 Single-Point Direct Method for Calibration of I_o

- Signal generator output calibrated in absolute dBm
- Power meter for checking the signal generator calibration

This calibration method uses the TTY setup commands.

1. Use the power meter to check the calibration of the signal generator.
2. Turn off the radiation and connect the signal generator to the test signal injection point.
3. Raise the antenna to at least 20°, and set the azimuth to point away from any known RF sources including the sun.
4. Select the pulse width using the **Mt** command.
   See 5.2.5 Mt — General Trigger Setups (page 114).
5. 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
```

6. 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 signal generator's RF signal frequency so that it produces the precise IF frequency (for example, IF frequency of 30 MHz).
7. Turn the signal generator off and record the *Filtered* power level.
   Because of large averaging, it takes several seconds for the average to stabilize.
8. 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.
   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 power equals the noise power.
9. Correct the value for losses.
   See 7.6.3 Treatment of Losses in Calibration (page 215).

7.6.3 Treatment of Losses in Calibration

When calibrating the dBm level of the test signal, you must account for any losses that may occur between the antenna feed and the injection point, and in the cable and coupler that connect the signal generator to the injection point.

The following figure shows the nomenclature of the losses that are involved in the calibration.

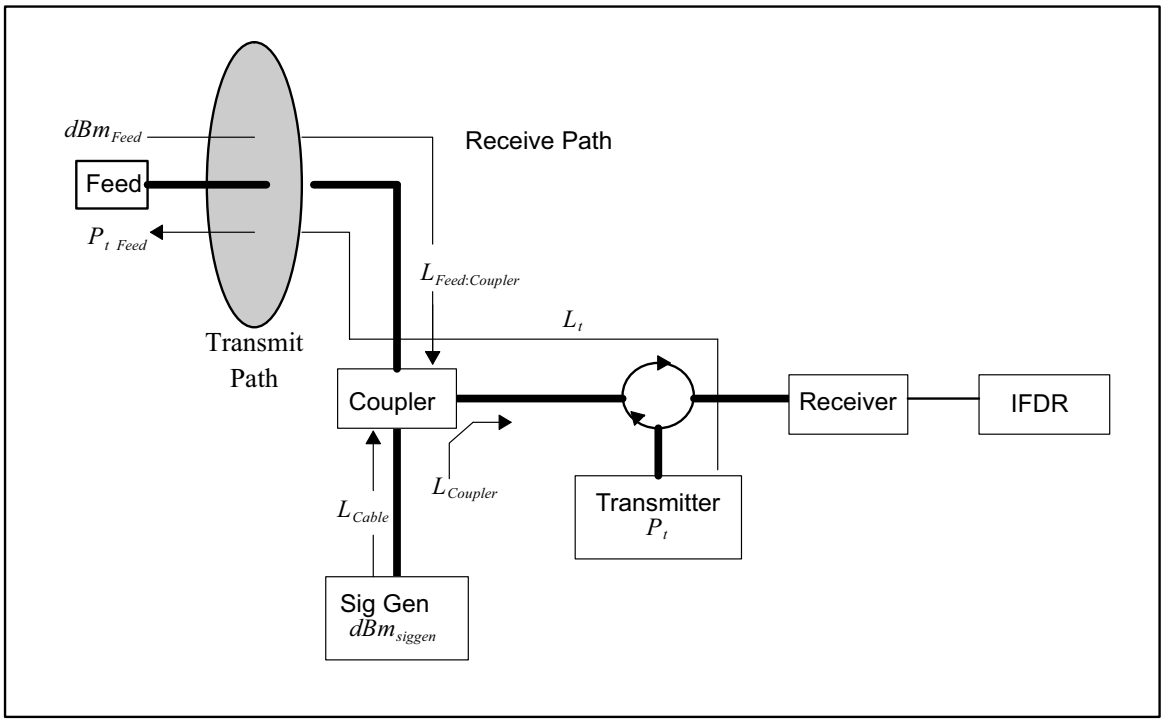


Figure 48 Overview of Losses that Affect LOG Calibration

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	Relationship	Value
Loss between the feed and the coupler	$dBL_{Feed:Coupler}$	3 dB
Loss caused by the coupler	$dBL_{Coupler}$	30 dB
Loss in the cable from siggen to coupler	dBL_{Cable}	2 dB

If the test signal generator output is -50 dBm, the injected power is

$$dBm_{Injected} = -50 - [30 + 2] = -82\ dBm$$

The equivalent power at the feed is then 3 dB more than this:

$$dBm_{Feed} = -82 + 3 = -79\ dBm$$

During the calibration, there are several ways to handle the losses using these equations. For example:

- Correct each signal generator value for losses so that the calibration plot shows IFDR measured power against the received power at the feed.
  This is recommended for manual calibration.
- Plot the signal generator values directly and correct the intercept power $\mathtt{I_0}$ for losses so that it is properly referenced to power at the feed.
  This is the approach used by the IRIS **Zauto** utility. See *IRIS and RDA Utilities Guide*.

7.6.4 Determining dBZo

The calibration reflectivity is determined from the radar equation as follows:

$$dbZ_0 = 10\log\left[Cr_0^2 I_0\right]$$

where $\mathtt{I_o}$ is in **mW** (corrected for receive losses), the reference range $\mathtt{r_o}$ is 1 km, and the radar constant **C** is:

$$C = \frac{2.69 \times 10^{16}\lambda^2}{P_t \tau\theta\phi G^2} L_t$$

where:

λ
: Radar wavelength in cm.

P_t
: Transmitted peak power in kW.

L_t
: Transmit loss (for example, 3 dB corresponds to $L_t = 2$)

τ
: Pulse width in microseconds.

θ
: Horizontal half-power full beamwidth.

φ
: Vertical half-power full beamwidth.

G
: Antenna gain (dimensionless) on beam axis.

The radar constant is determined from the characteristics of your radar (check with the manufacturer if you are unsure of the values). Note that transmit losses are accounted for in the radar constant, while receiver loss is usually included in the calculation of $\mathtt{I_0}$.

If the value of $\mathbf{I_0}$ calculated above was not based on loss-corrected dBm values, correct $\mathbf{I_0}$ as follows:

$$dBI_{0\,corrected} = dBI_0 - dBL_{Coupler} - dBL_{Cable} + dBL_{Feed:Coupler}$$

Example Calculation of dBZo

Use this sample calculation to check your arithmetic. The radar parameters:

Table 55 Radar Parameters

Parameter	Description	Value
λ	Radar wavelength in cm.	5 cm
P_t	Transmitted power in kW.	500 kW
L_t	Transmit loss.	2 (3 dB)
τ	Pulse width in microseconds.	1 microsecond
θ	Horizontal half-power beamwidth in degrees.	1°
φ	Vertical half-power beamwidth in degrees.	1°
G	Antenna gain (dimensionless) on beam axis.	19,953 (43.0 dB)

The radar constant for this example is,

$$C = \frac{2.69 \times 10^{16}\lambda^2}{P_t\tau\theta\phi G^2}L_t = \frac{\left(2.69 \times 10^{16}\right)(5)^2}{\left(500\right)\left(1\right)\left(1\right)(19,953)^2}(2.0) = 6.76 \times 10^6\left[mm^6m^{-3}km^{-2}mW^{-1}\right]$$

Assume that $\mathbf{I_0}$ with loss correction is calculated to be -105 dBm (3.16×10^{-11} mW), then dBZ_0 is:

$$dBZ_0 = 10\log\left[Cr_0^2I_0\right] = 10\log\left[\left(6.76 \times 10^6\right)(1)^2\left(3.16 \times 10^{11}\right)\right] = -36.7dB\left(mm^6m^{-3}\right)$$

You can download this value to the signal processor using the **SOPRM** command.

7.7 Dual PRT Processing Mode

These modes were originally implemented in the RVP7 processor, but have not yet been ported to RVP900.

RVP 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 are organized so that the first pulse of a short PRT pair always comes first.

7.7.1 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 is generally quite large (up to 20 KHz); and the reciprocal of this "effective instantaneous PRF" determines the trigger's short PRT interval. In this way, all subsequent physical calculations are scaled correctly, for example, unambiguous velocity, maximum first trip range, and so on, are all

supposed to be based on the short PRT interval. The host computer must be configured so that it can request 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 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. RVP900 returns the GPARM "Invalid Processor Configuration" bit if "R2" is requested in DPRT mode. The error bit is also returned if the number of pulses requested (sample size) is not even. All other error conditions are the same as FFT mode.

CAUTION! Since the RVP900 "Maximum number of Pulses/Sec" is usedto enforce the duty cycle limit, it is essential that it not be overwritten by the host computer's upper PRF limit, which typically is much higher.
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.

You can run Dual-PRF velocity unfolding within the DPRT-1mode. The short PRT varies in the selected 3:2, 4:3, or 5:4 ratio, but the overall duty cycle remains constant. The combination of Dual-PRF and DPRT-1 is very effective in extending the radar's unambiguous velocity interval.

7.7.2 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, RVP900 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 when 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.

7.8 Dual PRF Velocity Unfolding

For a radar of wavelength λ operating at a fixed sampling period τ_s = **1/PRF**, the unambiguous velocity and range intervals are given by:

$$V_u = \frac{\lambda}{4\tau_s} \; and \quad R_u = c\frac{\tau_s}{2}$$

where **c** is the speed of light. Often these intervals do not fully cover the span of velocity and range that one would like to measure. The problem is generally worse for short wavelength radars, since that unambiguous velocity span is directly proportional to λ for a given τ_s. If the unambiguous range interval is made sufficiently large by increasing τ_s, then the resulting velocity span may be unacceptably small.

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 (that is, unfold) the mean velocity measurement from each individual period. The Dual PRF trigger pattern consists of alternating (**N**+**k**)-pulse intervals where the period in each interval is either τ_l (for the low-PRF) or τ_h (for the high-PRF). Here **N** is the sample size, and **k** represents a delay that permits the clutter filter to equilibrate to the new PRF after each change. The clutter filter impulse response lengths vary according to which filter is selected.

The two trigger periods τ_l and τ_h must be chosen in either a 3:2, 4:3, or 5:4 ratio. These ratios give factors of two, three, and four times velocity expansion over the τ_h period alone. The unfolding algorithm makes use of the following results. Suppose that the radar observes a target with mean velocity V at each of the two trigger periods. The measured phase angles for the R_1 autocorrelations at the two PRFs are:

$$\theta_l = \frac{4\pi V \tau_l}{\lambda} \; and \; \theta_h = \frac{4\pi V \tau_h}{\lambda}$$

where angles outside the basic [- Π, Π] interval are returned to that interval by appropriate additions of ± 2Π . These angles correspond to the ordinary single-PRF Doppler velocity measurements, and the ± 2Π uncertainties reflects the fact that each measurement is folded into its own unambiguous interval:

$$V_{ul} = \frac{\lambda}{4\tau_l} \quad and \quad V_{uh} = \frac{\lambda}{4\tau_h}$$

If we define ϕ to be the difference between the two measured phases then:

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

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

$$V_u unfold = \frac{\lambda}{4(\tau_l - \tau_h)}$$

Now if τ_l and τ_h are in a 3:2 ratio, then:

$$\tau_l - \tau_h = \frac{\tau_l}{3} = \frac{\tau_h}{2}$$

and thus:

$$V_u unfold = 3V_{ul} = 2V_{uh}$$

The angle Ø represents a velocity phase angle in [- Π, Π] , but with respect to an enlarged unambiguous interval. By 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.

Velocity Estimator

In practice, the unfolded angle Ø is not in itself a suitable velocity estimator. This is because 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.

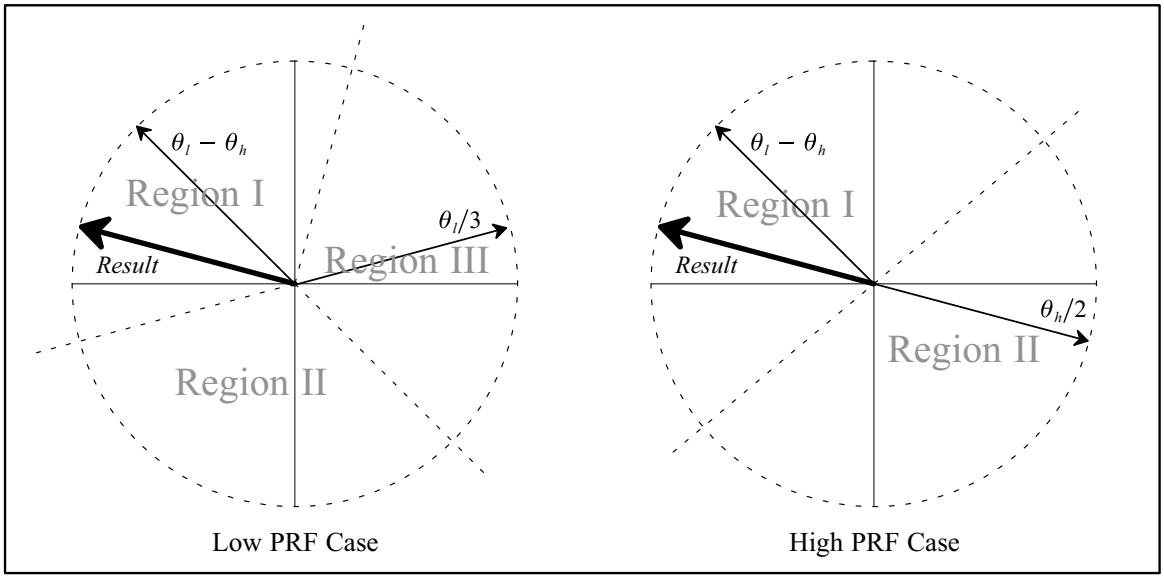


Figure 49 Dual PRF Concepts

This technique is illustrated in the previous figure. 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 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

$$\frac{\theta_l}{3}, \frac{\theta_l}{3}, + \frac{2\pi}{3} \; or \; \frac{\theta_l}{3} + \frac{4\pi}{3}$$

depending on which one falls into the acceptance `Region 1`. The resultant angle is the same in each case.

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, 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.

Width Data

The width data are 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 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.

A Few Words of Caution Regarding Dual PRF Processing

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. You must ensure 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 has 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 .

These clutter filter "images" are a 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.
To minimize their impact, turn the clutter filter off at far ranges where little clutter is expected and use 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. You must 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 your host computer.

Example

The following example shows 7 possible oscilloscope 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 labeled **SCOPE** is the composite of these traces, and is what is shown on an oscilloscope.

Note 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.

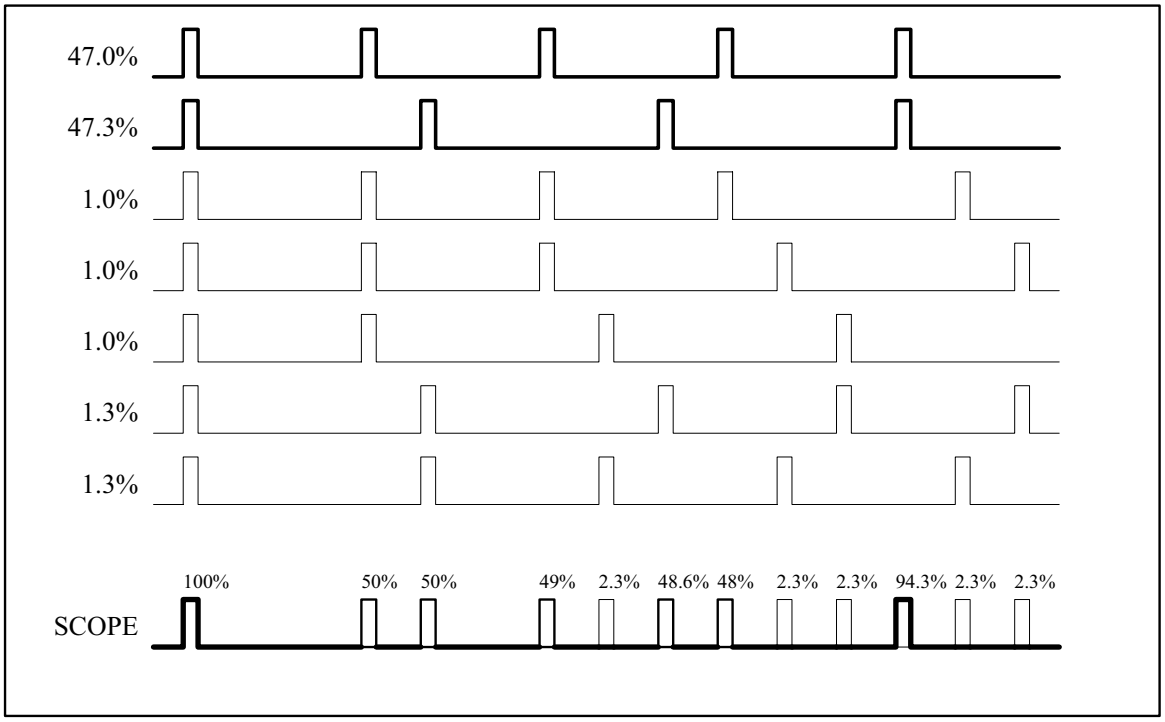


Figure 50 Example of Dual PRF Trigger Waveforms

7.9 Random Phase Second Trip Processing

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 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.

RVP900 has optional random phase processing for the filtering and recovery of second trip echoes. While details of the technique are proprietary to Vaisala, we describe the general principle and the configuration options to optimize the algorithm performance.

The information that is used to separate the first and second trip echoes is the phase.

Magnetron Radars

For a magnetron radar, the phase of each pulse is different. 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.

RVP900 measures the phase of the transmitted pulse and the phase locking is done digitally as opposed to the traditionally locking COHO.

Klystron Radars

For a Klystron radar, the phase is controlled by RVP900 through 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 RVP900.

7.9.1 Random Phase Second Trip Processing Algorithm

The following figure shows a schematic of the data processing for random phase. The figure shows the Doppler spectra for the first and second trip in the 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).

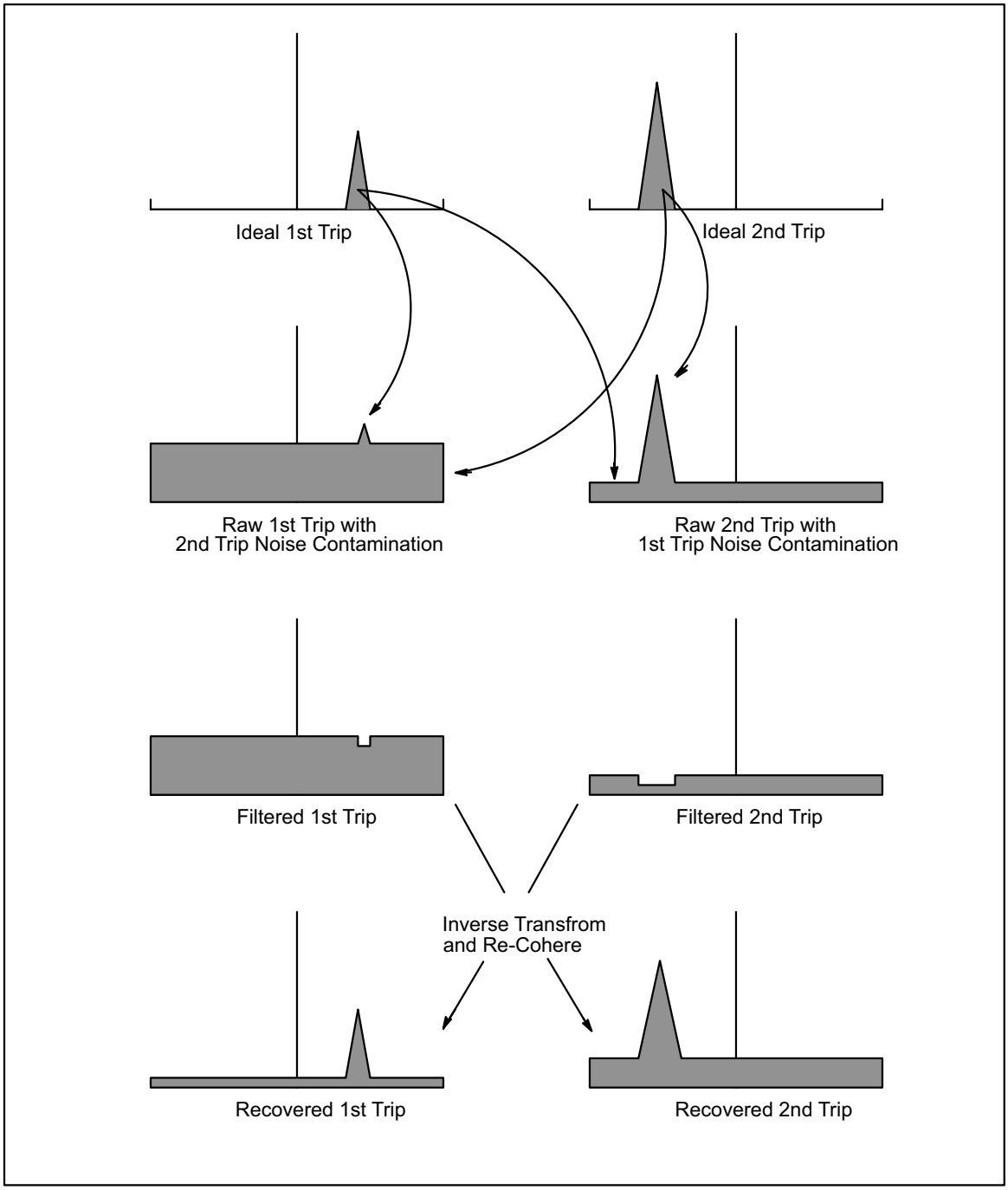


Figure 51 Random Phase Processing Algorithm

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. The echoes 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 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 has 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 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 is dominated by the coherent echo, that is, the other trip phase information is contaminated. For this reason, the filtering should effect as small a region of the spectrum as possible.

7.9.2 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, and so on.

However, a few parameters are special to random phase mode, 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, especially 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, cannot be placed in either trip.

An SQI threshold is required to qualify reflectivity and dual pol data in all the processing modes. RVP900 defines a secondary SQI threshold $\mathtt{SQI}_2$ that is computed from the standard threshold value 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 usually match 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 ... 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.

RVP900 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 also be used to place similar limits on the depth of visibility of the adaptive filters.

As an example, if 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 ... 100 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, there is an abrupt ring of missing data in the beginning of the second trip. If set too high, there are 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 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.

For more information, see 7.9.1 Random Phase Second Trip Processing Algorithm (page 224).

7.10 Signal Generator Algorithm Testing

The IF signal generator tests that can be used to verify the RVP900 processing algorithms.

Perform these tests after adding new algorithms or major modes to the processor.

You can use the test descriptions to debug your system, or better understand how they work.

More Information

- Installation and Test Procedure Overview (page 335)

7.10.1 Linear Ramp of Velocity with Range

Suppose that a continuous-wave IF waveform has an instantaneous frequency f (t)in Hertz (cycles/sec). Consider a range bin located at time

τ_{bin} within a set of pulses that are separated by τ_s = 1/PRF. The phase measured at that bin on the n^{th} pulse is the integral of the frequency within that pulse starting from range zero (since RVP900 is phase-locked to range 0):

$$\Phi_n = \int_{n\tau_s}^{n\tau_s + \tau_{bin}} f(t)\,dt$$

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 is independent of both T_0 and n. Thus, a linear FM input signal produces a fixed (I,Q) phase difference from pulse-to-pulse at any given range. The magnitude of the phase difference is proportional to the range, and the slope is($M\ \tau_s$) cycles for each second of delay in range. For example, if the test signal generator is sweeping 100 KHz every two seconds, then the velocity observed at a range of 300 km at 250 Hz PRF is:

$$\Phi_{n+1} - \Phi_n = \left(\frac{100KHz}{2\text{sec}}\right) \times \left(\frac{1}{250}\text{sec}\right) \times (300km) \times \left(\frac{6.6\mu\text{sec}}{1km}\right) = 0.40\ \ cycles$$

We would thus observe a velocity of (0.8 × Vu) at 300 km, where Vu 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.

This simple FM signal generator also produces valid second trip velocities that can be seen during Random Phase processing. This follows from the above analysis because we've never assumed that τbin was smaller than τs, that is, it is fine for the range bin to be located in any higher-order trip.

7.10.2 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 is:

$$\left(1.0\frac{\text{deg}\,ree}{km}\right) \times \left(\frac{1}{360}\frac{cycles}{\text{deg}\,ree}\right) \times \left(299792\frac{km}{\text{sec}\,ond}\right) = 833\frac{cycles}{\text{sec}\,ond}$$

7.10.3 Verifying RHOH, RHOV, and RHOHV

These terms measure the normalized cross-channel covariance in a polarization radar. They all are computed having the form:

$$RHOAB = \frac{< S_A^n\ S_B^{n}* >}{\sqrt{< S_A^2\ S_B^2 >}}$$

Where the S_A^n and S_B^n 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_A^n and S_B^n samples are coming from two signal generators installed on a dual-receiver system, and that only the B-Channel is AM modulated so that:

$$\left|S_A^n\right| = \{S_A, S_A, S_A, S_A, S_a ...\},\ \left|S_B^n\right| = \{S_B, 0, S_b, 0, S_B ...\}$$

Then the above estimator reduces to:

$$RHOAB = \frac{\left(\frac{1}{2}\right) S_A S_B}{\sqrt{S_A^2 \times \left(\frac{1}{2}\right)\ S_B^2}} = 07.07$$

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. We do not choose 100% depth because we would lose 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 see the RHOAB terms varying with range; reaching a high value of 1.00, and a low value of 0.707. The plots are 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 is not affect the p terms, but it does have an interesting effect on SQI. If (T,Z,V,W) are computed from both channels combined, then the SQI is:

$$SQI = \frac{S_A^2}{S_A^2 + \left(\frac{1}{2}\right) S_B^2}$$

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 0.

The linear FM ramp (see 7.10.1 Linear Ramp of Velocity with Range (page 228)) can also be used as a test of RHOAB in a dual-receiver system. With one siggen modulated and the other fixed, one receive channel appears to rotate 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 observe `RHOAB = 0` at that range. The plot of RHOAB shows a characteristic `sin (x)/x` behavior as a function of range.

8. Host Computer Commands

8.1 Host Computer Command Overview

RVP includes a set of digital commands that the host computer must use to set up and control the RVP900 processor for recording data.

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`, and so on. If the command produces output, those words are listed as `Output 1`, `Output 2`, and so on.

Often each word is broken down into independent fields, each consisting of one or more bits. All data transferred to or from RVP900 are in the form of 16-bit words.

Unless otherwise noted, the command descriptions describe set bits.

More Information

- Output Data (page 49)

8.1.1 Setting-up Data Acquisition and Processing

Although the internal RVP900 tables and parameters are set to reasonable values on power-up, you usually need to modify the default information to meet your needs.

1. On power-up, issue **`IOTEST`** to ensure that the interface connections are all intact.
2. Check the diagnostic result registers from **`GPARM`** to verify that RVP900 passes all internal checks.
3. Establish trigger and pulse width using the **`SETPWF`** commands.
4. Chose range bin placement and processor options using **`LRMSK`** and **`SOPRM`**.
5. Take receiver noise samples with **`SNOISE`**.
   The noise levels are not automatically sampled on power-up, you must issue **`SNOISE`** at least once.
6. If clutter filters are needed, issue **`LFILT`**.
7. If data rays are to be synchronized with antenna motion, issue **`LSYNC`** to specify a table of antenna angles.

8. When the setups are complete, issue **PROC** commands to collect, process, and output the data.

To monitor errors detected during the execution of commands using the **GPARM** command.

8.1.2 First-in-first-out (FIFO) buffer

RVP900 contains a 4096-word first-in-first-out (FIFO) buffer through which all output data flow.

The FIFO holds each sequential word generated by RVP900 until the user is ready to accept it. When reading from the processor, the host can fall behind by as many as 4096 words before performance slows.

RVP900 writes to the FIFO at full speed as long as it is not full. Internal processing is not affected by the exact speed at which user I/O occurs.

Writing to the FIFO continues as long as the average I/O rate on, perhaps 10 ms intervals, and matches the average rate at which data are being produced.

Full Output FIFO

The sequence of events changes when the FIFO is full:

1. 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.
2. RVP900 waits and does not proceed with its internal processing until space becomes available.

Despite the slowdown in performance, you alway obtains correct data no matter how long it takes to read it. You can take advantage of this 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, you can read these words at your maximum rate, and the overall system automatically runs at the slower of those two speeds.

Write Cycle and Full Output FIFO

If the output FIFO is full and the RVP900 has the next word ready for output, the idle wait loop can be exited 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 understood to mean that a more important condition has arisen. The wait loop terminates and 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 processor continues to execute all commands correctly, but that their output is discarded.

The discarded output data are not lost. The data are eventually replaced with an equal number of zeros. Each time RVP900 discards an output word, it increments an internal 24-bit count. When FIFO space becomes available, the processor replaces the missing data with zero-valued placeholders.

Writing when the FIFO is full can be useful if the new command is a **RESET** which calls for clearing the output FIFO. When the **RESET** is processed, all past and present output data are discarded, leaving the RVP900 output section empty. This is useful when the processor has pending output data that the user wants to discard.

8.2 No-Operation (**NOP**)

NOP is useful when a number of words must be flushed through RVP900 without side effects.

This single-word instruction is ignored by the signal processor.

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|                              |           | 0   0   0   0   0 |  Command
|------------------------------|-------|---|-------------------|
```

8.3 Load Range Mask (**LRMSK**)

LRMSK informs the signal processor of the ranges at which data are to be collected.

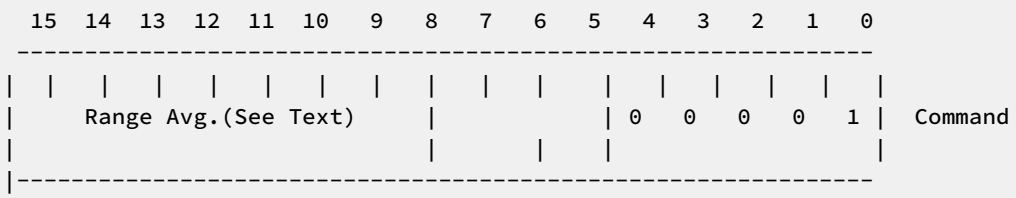
An arbitrary set of range bins are selected through an 8192- bit mask. The Nth bit in the mask determines whether data are acquired and processed at a range equal to:

`RES x (N-1)`

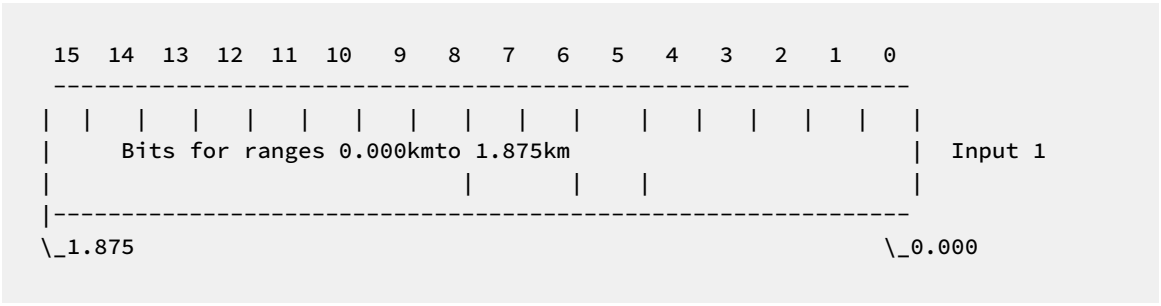
Table 56 **LRMSK** Parameters

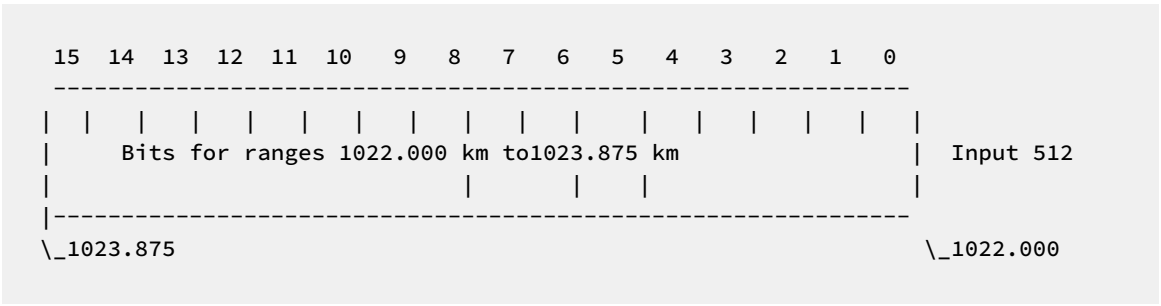
Range Parameters	Description
Range resolution	Specified by a TTY setup question , in the range 25 ... 1000 meters. Any collection of ranges may be chosen from integer multiples of that distance. See 5.2.6 Mt<n> — Triggers for Pulsewidth n (page 117).

Range Parameters	Description
Range mask	The range mask is passed to RVP9 packed in 512 16-bit words. In each group of 16: • The least significant bit of each packed word represents the nearest range. • The most significant bit represents the furthest range. 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 **LRMSK** implicitly specifies the number of bins to be processed and output. The maximum bin count is 4200, though depending on the computational intensity of the configuration, RVP9 may be able to compute fewer bins. If the number of bins selected in the bit mask exceeds the maximum, the trailing bins are truncated. If the new mask does not specify any active bins, then a single bin at range 0 is forced on. The default power-up mask selects 256 bins equally spaced by 1.0 km starting from 0 range.
Range averaging	**LRMSK** determines range averaging. The upper byte of the command controls how many consecutive bins are grouped together. For example: • `0`: No averaging For example, if 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. • `1`: Pairs of samples are averaged If the averaging is to 1, samples are taken at the same ranges, but pairs of bins are averaged together and only 50 ranges would result. Note that the parameters are averaged by summing the autocorrelations for each bin. The individual samples that go into each average are taken according to the bits that are set in the mask, except that they are grouped together so that only one net bin results from the several data samples. Note that the limitation of 4200 sampled ranges applies to the bin count prior to averaging. • `255`: 256 terms are summed The range normalization value associated with the averaged bin is computed according to the midpoint of the first and last sample. The command discards incompletely averaged bins. Continuing from the previous example, if the averaging were set to 2 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.



In the following examples, the ranges listed in Inputs 1 ... 512 assume a 125 km range bin.

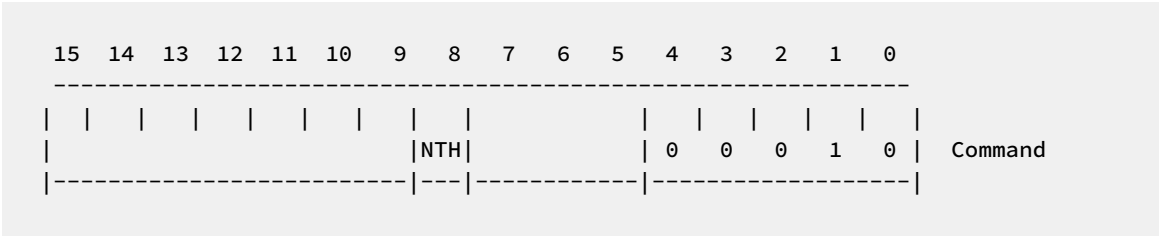




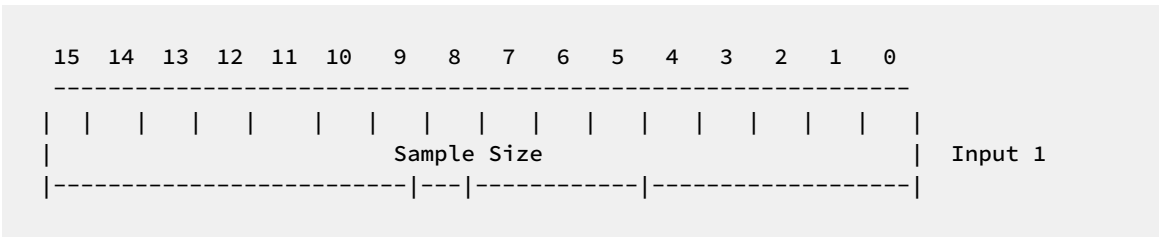
8.4 Setup Operating Parameters (**SOPRM**)

Use **SOPRM** to configure the signal processor. You must issue **SOPRM** when any of the parameters in the list change.

The default parameter list consists of 20, 16-bit input words. These can be followed by optional **XARG** parameters as needed.



If `Nth` is `1`, then no threshold values are set. This means the system ignores input words `4`, `5`, `6`, `7`, `11`, `12`, `13`, `14`, and `18`. This is usually used with the **THRESH** command when setting individual thresholds. See 8.30 Set Individual Thresholds (THRESH) (page 311).



The sample size is continually adjustable from 1 ... 256 pulses. In the alternating polarization mode, the sample size must be even, if an odd value is entered, it is rounded up by one.

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|  |ZNS| Polar |NHD |ASZ|16B|CMS|R2 |   |3x3|CCB|   |Lsr|Dsr|Rnv|  Input 2
|--|---|-------|----|---|---|---|---|---|---|---|---|---|---|---|-
```

Each single-bit field selects whether the given processing or threshold option is enabled (`1`) or disabled (`0`).

Table 57 **SOPRM** Threshold Options

Threshold Option	Description
`ZNS`	If `Rnv` is `0` (no range normalization), you can set `ZNS` to have 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 the inclusion of header words in the processed data that are output by the **PROC** command. See also 8.23 Configure Ray Header Words (CFGHDR) (page 304).
`ASZ`	The `Any Spectrum Size` bit requests that DFT processing algorithms, clutter filters, spectral output, and so on. 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. The **PROC** command's archive format always holds 8-bit data, regardless of the 16B setting. This gives the option of extracting 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 3 lag (`R0/R1/R2`) algorithms for width, signal power, and clutter correction.

Threshold Option	Description
3×3	Switches on the 3x3 2D output filter. See 7.5.3 Speckle Filter Processing (page 208). RVP900 automatically handles all of the pipelining overhead associated with running the 3×3 filter. 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 **Lsr** or **Dsr** is set.

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|              Log Slope65536 * dB / (LSB)                      |  Input 3
|---------------------------------------------------------------|
```

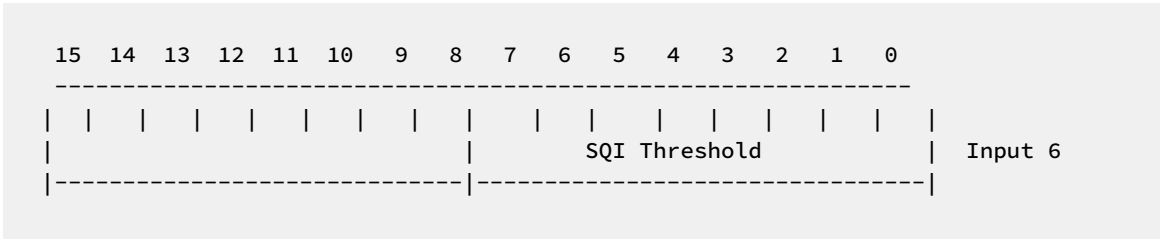
This number defines the multiplicative constant that converts the signal power in dB to the units of the 12-bit **Log of power in sample** time series outputs. One fourth (1/4) of this slope is used to generate the **Log of Measured Noise Level** output from **GPARM** (word 6). The recommended value is 0.03 (1966). This gives a dynamic range of 122 dB in 12 bits.

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|              LOG Threshold in 1/16 of dB                      |  Input 4
|---------------------------------------------------------------|
```

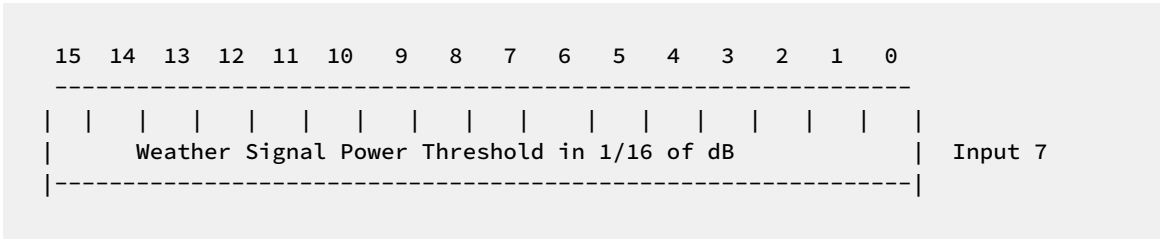
Reflectivity values below this level can result in thresholding of data, if the threshold control flags (see below) include **LOG Noise** bits. The threshold value is always non-negative. For the comparison test, see 7.5 Thresholding (page 205).

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|       Clutter Correction (CCOR) Threshold in 1/16 of dB       |  Input 5
|---------------------------------------------------------------|
```

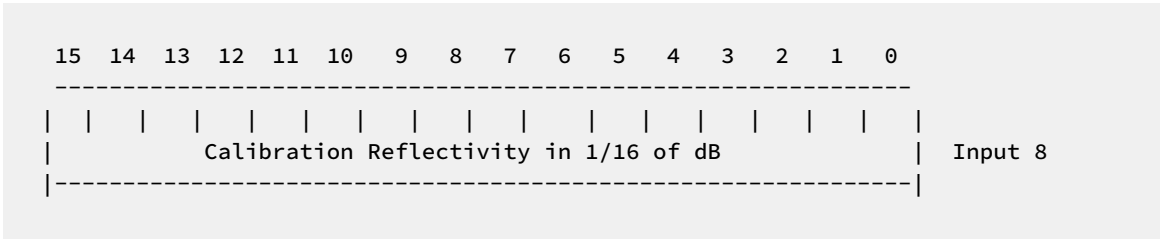
The clutter correction threshold is a bound on the computed log receiver adjustment for clutter. These corrections (in dB) are always negative. Any clutter correction which is more negative than the above value can result in thresholding of data.



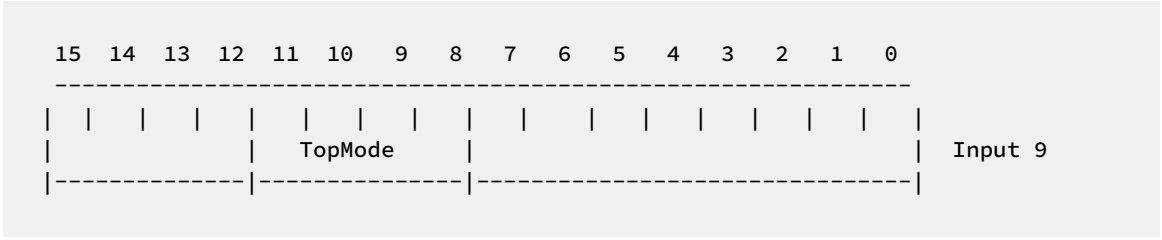
The signal quality index (SQI) threshold is an unsigned binary fraction in the range 0 ... 255/256. When the SQI for a range bin falls below the stated value, it may result in thresholding of data. An analogous Polarimetric Meteo Index (PMI) can be set by the **THRESH** command. See 8.30 Set Individual Thresholds (THRESH) (page 311).



Weather signal power (**SIG**) is an estimate of the SNR of the weather component of the received signal. When the **SIG** falls below this comparison value, it may result in data thresholding. See 7.4.8 Weather Signal Power (SIG Threshold) (page 204).



The calibration reflectivity is referenced to 1.0 kilometers.



The **TopMode** bits select the overall data acquisition and processing mode for RVP900. Although the processing algorithms used in each top level mode are different, the RVP900 command set works in a uniform way in all modes.

Table 58 TopMode Bits

Bits	Description
0000	Polarimetric processing (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 7.3 Time Series Signal Processing (page 181).
0001	FFT processing mode is a dedicated frequency-domain approach; data are processed in batches of pulses. See 7.3.1 Frequency Domain Processing- Doppler Power Spectrum (page 183).
0010	Random phase processing mode. Data from first and second trips are dealiased in range based on knowledge of the radar transmitter phase. See 7.9 Random Phase Second Trip Processing (page 223).
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 7.7 Dual PRT Processing Mode (page 217).
0101	DPRT-2 processing mode. The trigger generator produces alternate short and long pulses, and Doppler autocorrelations are computed using both pairs. See 7.7 Dual PRT Processing Mode (page 217).
11XX	Reserved for custom user modes.

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|  |   |   |   |    |   |   |   |   |   |   |   |   |   |   |   |
|  |   |UVD|PCT|  Window    |ZER|   Filter Stabilization Delay  |  Input 10
|--|---|---|---|----|---|---|---|---|---|---|---|---|---|-------|
```

The RVP900 clutter filters are controlled by this word.

Table 59 RVP Clutter Filter Controls

Control	Description
Delay	This delay is introduced prior to processing the next ray of data when 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.

Control	Description
`Window`	Selects the type of window that is applied to time series data prior to computing power spectra with a DFT. Choices are: • 0:Rectangular • 1:Hamming • 2:Blackman • 3:Exact Blackman • 4:VonHann
`PCT`	If set, RVP900 attempts to run its standard processing algorithms even when a custom trigger pattern has been selected with the **`SETPWF`** command.
`UVD`	Unfold velocities using a simple (Vhigh Vlow) algorithm, rather than the standard algorithm described in 7.5.3.2.1 Dual-PRF Unfolding (page 210) .

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|     Threshold Control Flags for UnCorrected Reflectivity      |  Input 11
|---------------------------------------------------------------|
```

These flags select which legacy threshold comparisons result in uncorrected reflectivity being accepted or rejected at each bin. There are 4 test comparisons made at each range, as described above for input words `4`, `5`, `6`, and `7`. Further quality tests such as the Polarimetric Meteo Index can be configured for each data type. See 8.30 Set Individual Thresholds (THRESH) (page 311).

Each test either passes and produces a code of `1`, `2`, `4`, and `8` respectively, or fails and produces a code of `0`. The sum of the codes for each of the 4 tests is a number between 0 and 15, which can also be interpreted as the following four-bit binary number:

```
  3   2   1   0
-----------------
| 8 | 4 | 2 | 1 |
|---|---|---|---|
   \   \   \   \_______ LOG Threshold Passes
    \   \   \__________ CCOR Threshold Passes
     \   \_____________ SQI Threshold Passes
      \________________ SIG Threshold Passes
```

The individual bits of the `Threshold Control Flag` word specify whether data are to be accepted (`1`) or rejected (`0`) in each of the possible combination of threshold outcomes. The pattern of bits in the flag word represents a truth table for a given logical function of the four threshold outcomes.

The following examples show values of the `Flag` word for the stated combinations of acceptance criteria:

Table 60 Example Flag Values With Acceptance Criteria Combinations

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

One 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.

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|     Threshold Control Flags for Corrected Reflectivity        |  Input 12
|---------------------------------------------------------------|
```

See description for **Input 11**.

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|     Threshold Control Flags for Velocity                      |  Input 13
|---------------------------------------------------------------|
```

See description for **Input 11**.

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|     Threshold Control Flags for Width                         |  Input 14
|---------------------------------------------------------------|
```

See description for **Input 11**.

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|     Additive Offset for Measured AZ Angles (Binary Angle)     |  Input 15
|---------------------------------------------------------------|
```

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|     Additive Offset for Measured EL Angles (Binary Angle)     |  Input 16
|---------------------------------------------------------------|
```

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|     Intervening Gas Attenuation Correction (dB/km)            |  Input 17
|---------------------------------------------------------------|
```

Gas attenuation correction attempts to compensate for overall (two-way) beam losses due to absorption by atmospheric gases. 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. Gas attenuation compensation can be turned off when RNV is on by setting a slope of 0.0 dB/km.

An attenuation of G db/km is encoded into the unsigned 16-bit word **N** as follows:

$0 \leq N \leq 10000 \;\; G = \frac{N}{100000}$

else

`G = 0.1 + (N – 10000)/10000`

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

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|    Threshold Control Flags for Differential Reflectivity(Zdr) |  Input 18
|---------------------------------------------------------------|
```

See description for **`Input 11`**.

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|    Signed Zdr Calibration Offset in 1/16 dB(GDR)              |  Input 19
|---------------------------------------------------------------|
```

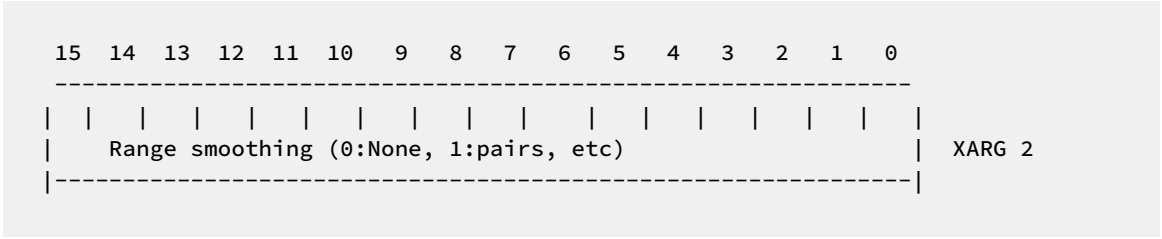
When differential reflectivity is computed there is a possibility that radar asymmetries introduce a bias in the **`Zdr`** values, that is, that **`Zdr`** is non-zero even when observing purely spherical targets. This calibration offset permits nulling out this effect. The GDR offset accounts for the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

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|    Radar Wavelength in Thousandths of Centimeters             |  Input 20
|---------------------------------------------------------------|
```

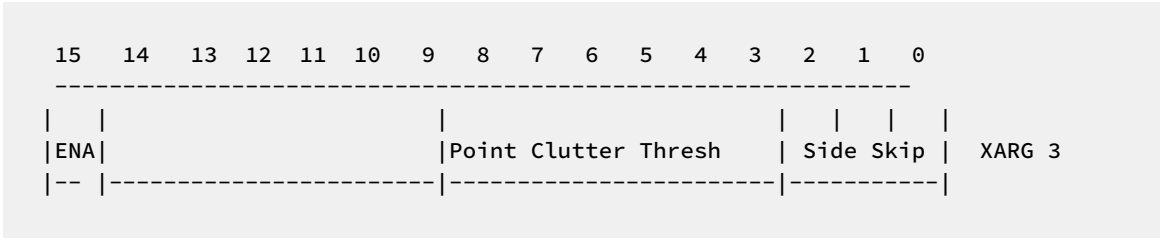
The radar wavelength is used in the calculation of 16-bit velocity and width data, to convert from Nyquist units to absolute physical units.

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|    Signed LDR Calibration Offset in 1/100 dB(XDR)             |  XARG 1
|---------------------------------------------------------------|
```

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 **`Zdr`**), the gain difference does not depend on differential transmit power.

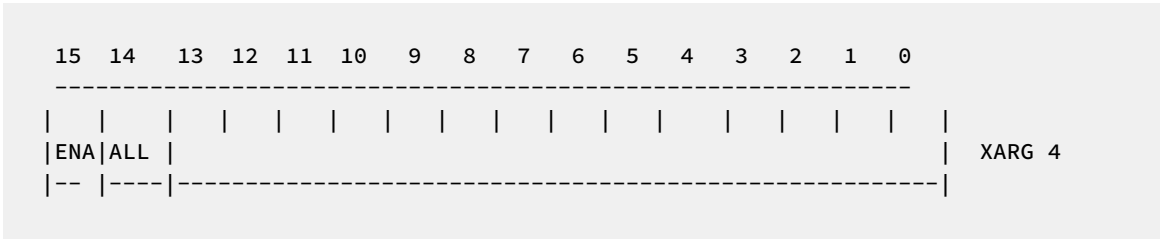


Range smoothing can be performed on raw moment data prior to the computation of scientific parameters. The number of bins to sum together is given here. This should generally be an odd integer so that no range bias is introduced by the smoothing operation.



Point clutter detection is configured with this word. A bin is flagged as containing clutter if its power exceeds that of its two neighboring bins by more than the detection threshold (in decibels). Up to 7 bins may optionally be skipped on each side of the central bin prior to making these two comparisons.

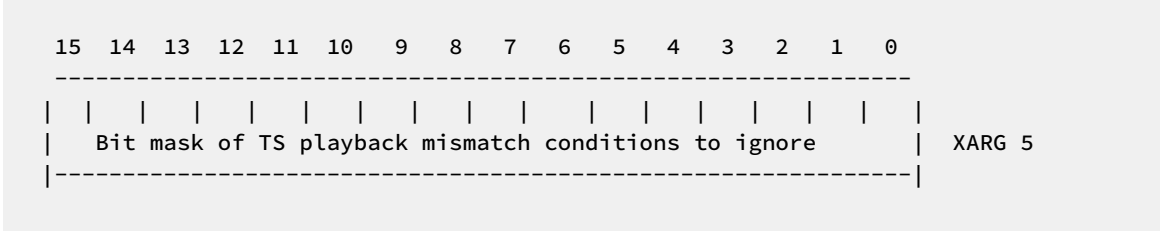
`Ena`
: This bit is set to enable point clutter detection. Flag bits are reported in the `Flg` output data type of the **PROC** command.



Point clutter censoring is configured with this word.

`Ena`
: Set to enable point clutter censoring. Raw moment data containing point clutter are 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, defined in *dsp.h*, 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 RVP900.

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|   Flag bits related to timeseries (playback)                  |  XARG 6
|---------------------------------------------------------------|
```

Combination of `OPTS_xxx` bits, , defined in *dsp.h*, which modify details of time series behavior.

```
 15   14   13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |    |   |   |   |   |   |    |   |   |   |   |   |   |     |
| 1 |    Offset in deciBels       |  Slope as 100 * dB/dB       |  XARG 7&8
|-- |-----------------------------|-------------------|---------|
```

These 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 0 restores the RVP900 defaults from the **Mf** menu.

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|   Melting Height in Meters Above/Below Mean Sea Level         |  XARG 9
|---------------------------------------------------------------|
```

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

The MSB is complemented in this signed number. This means that a value of `0` is the most negative value and that the burst power was unknown. This is used to compute the long term burst power calibration adjustment.

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|    Burst power at time of calibration in 1/100 of dBm         |  XARG 10
|---------------------------------------------------------------|
```

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 are suitable for most radars.

Table 61 Default Values For Melting Height Operating Parameters

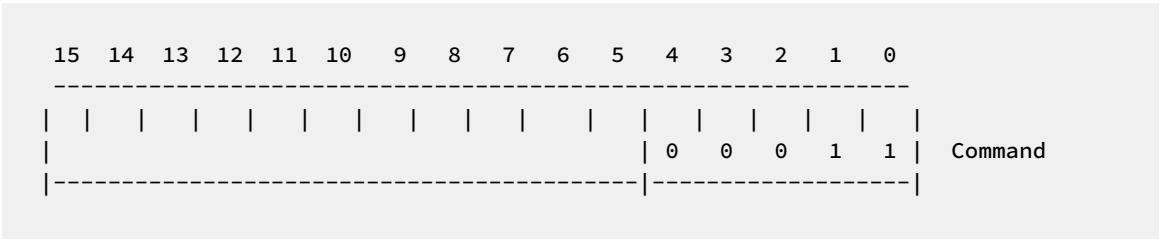
Parameter	Scientific Units	Input
Sample Size	25 pulses	25
Flag Word		0007 Hex
Log Slope	0.03 dB/LSB	1966
LOG Threshold	0.5 dB	8
CCOR Threshold	25.0 dB	400
Signal Quality Index Threshold	0.5 (dimensionless)	128
SIG Threshold	10.0 dB	160
Calibration Reflectivity	22.0 dBZ	352
Gas Attenuation	0.016 dB/km	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 and CSR	8888 Hex
Velocity Thresh. Control Flag	SQI and CSR	C0C0 Hex
Width Thresh. Control Flag	SQI and CSR and SIG	C000 Hex
Zdr Refl. Thresh. Control Flag	LOG	AAAA Hex
AZ/EL Angle Offsets	0°	0000 Hex
Altitude of radar	0 meters MSL	0

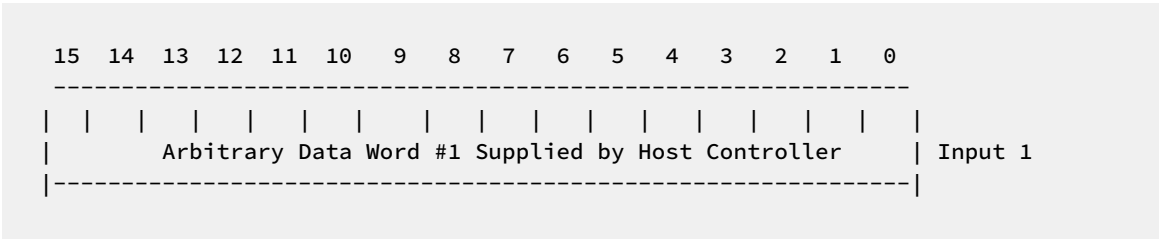
8.5 Interface Input/Output Test (**IOTEST**)

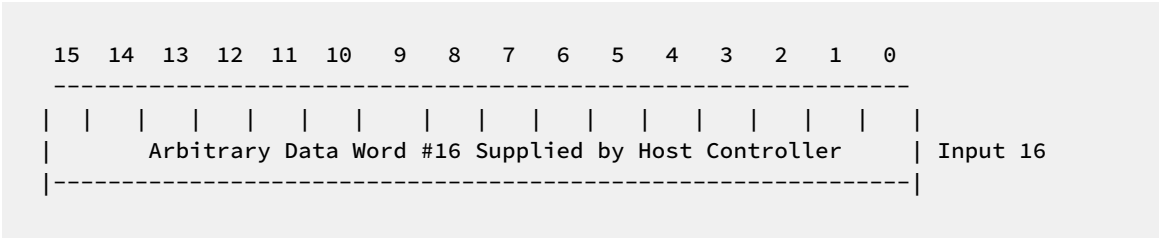
Use **IOTEST** to test the input and output data busses of the signal processor interface.

When issued, the command causes 16 words to be read from the host controller, after which the same 16 words are printed.

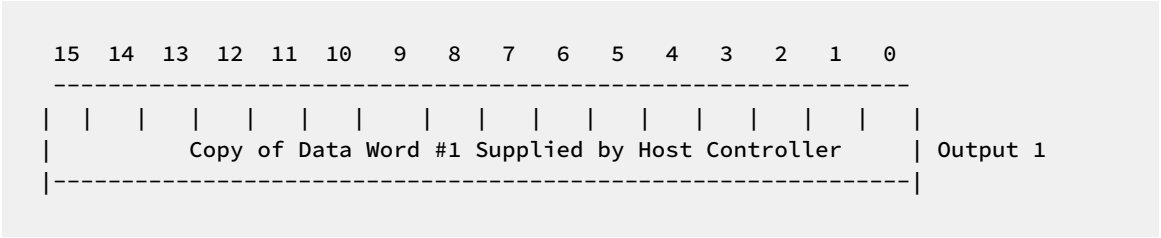
Typically, the controller supplies a "barber pole" input sequence consisting, for example, of successive powers of two. If all of the output words are correct, one may conclude that there are no malfunctioning bits in the interface hardware

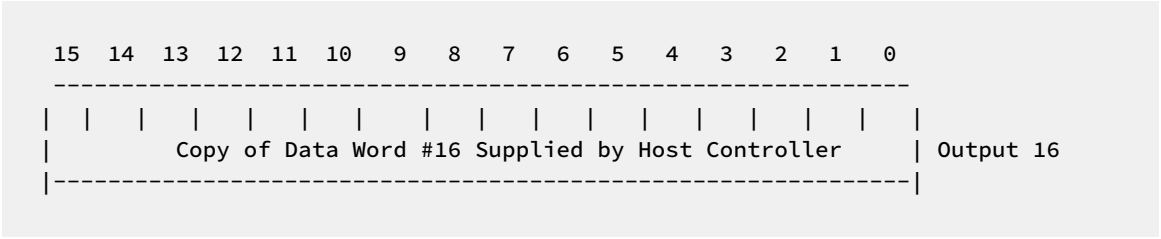






The **IOTEST** command can also process and echo up to 128 additional **XARGS** data words. See 8.21 Pass Auxiliary Arguments to Opcodes (XARGS) (page 302).





8.6 Interface Output Test (**OTEST**)

Use **OTEST** to test the integrity of the data being output by the signal processor. The command causes 16 words to be output consisting of successive powers of two starting from one.

By verifying whether each output word is correct, you can isolate malfunctioning bits in the interface data bus.

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 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   1 |  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
|-----------------------------------------------------------------|
```

8.7 Sample Noise Level (**SNOISE**)

Use **SNOISE** 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 the user's responsibility to make sure 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 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 µsec increments. Note 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 make sure the noise samples fit within the specified PRT, and within the range mask hardware RAM. RVP900 sets an error bit when an improper range is requested.

Reissue the **SNOISE** command periodically to compensate for drift in the RF and A/D systems.

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 5.2.6 Mt<n> — Triggers for Pulsewidth n (page 117)). RVP900 does not automatically take a noise sample as part of its initialization procedure.

The measured offsets are stored internally for all subsequent uses in RVP900. The offset values may be inspected through the **GPARM** command, as may the current range and rate values themselves. When 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.

```
  15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
  ---------------------------------------------------------------
 |  |   |   |   |   |   |   |   |   |    |   |   |   |   |   |   |
 |              |   Action  |Rat|Rng|        | 0   0   1   0   1 |  Command
 |--------------|-----------|---|---|--------|-------------------|
```

`Rng`
: If 1, the range in input word 1 is taken as the starting noise range for this and all subsequent **SNOISE** calls.

`Rat`
: If 1, the trigger rate in input word 2 is taken as the noise rate for this and all subsequent **SNOISE** calls.

`Action`
: Specifies what action is carried out by the command.

`0`
: Compute a new noise sample based on the present IFD input signals.

`1`
: Do not compute a noise sample, but rather, read new noise values from the host computer and use them for subsequent processing. Four additional input words supply the noise information, and **GPARM** words `6`, `9`, and `44` .. `50` 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.

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|   Starting Range in km (Max 992km) of 32km Sampling Interval  |  Input 1
|---------------------------------------------------------------|
```

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|  Internal Trigger Rate (6Mhz/N) to use During Noise Sampling  |  Input 2
|---------------------------------------------------------------|
```

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

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|0   0 | (MSB)        Log of Measured Noise Level(LSB)          |  Input 3
|------|-----------------------------------------------------------|
```

The is the same number as **GPARM** output 6. See 8.8 Initiate Processing (PROC) (page 251) and 8.10 Get Processor Parameters (GPARM) (page 267).

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|      Noise Level Standard Deviation (in 1/100 of a dB)        |  Input 4
|---------------------------------------------------------------|
```

This is the same number as the **GPARM** output `49`.

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|  Ratio of Horizontal/Vertical Noise Power in Hundredths of dB |  Input 5
|---------------------------------------------------------------|
```

This is the same number as the **GPARM** output `50`.

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|               <Spare>                           |Err|Ttf|Ntg|  Input 6
|-------------------------------------------------|---|---|---|
```

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

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|0   0 |  (MSB) Log of Measured Noise Level, puse 2 (LSB)       |  XARG 1
|------|---------------------------------------------------------|
```

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|   NOISE Level Standard Deviation, pulse 2 (in 1/100 of a dB)  |  XARG 2
|------|---------------------------------------------------------|
```

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| Ratio of Horizontal/Vertical Noise, pulse 2 (in 1/100 of a dB)|  XARG 3
|------|---------------------------------------------------------|
```

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.

8.8 Initiate Processing (**PROC**)

The **PROC** command controls the processing and output of radar data.

PROC is a single-word command that specifies the type of processing to be performed, and the type of output to be generated.

The following table shows the modes available in the command word.

Table 62 **PROC** Modes

Mode	Description
Synchronous mode	The processor acquires, processes, and outputs one ray in response to each **PROC** command. Processing begins after each command is 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. 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 8 and 9. For Doppler data either a 2:3, 3:4, or 4:5 PRF unfolding ratio may be selected. RVP900 performs the unfolding steps internally, so mean velocity is output with respect to the larger unambiguous interval. No additional velocity processing is needed except to change the velocity scale on any generated displays.

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 6 bits of the **PROC** command. Packed archive output is selected by setting the ARC bit. Individual byte or word display output is selected by setting any or all of the Z, T, V, W, Zdr, and Kdp command word bits. When more than one of these bits is set, the output array consists of all of the bins for the leftmost selected parameter, followed by all of the bins for the next selected parameter, and so on. 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.

In time series mode, there are three output data formats available. For backwards compatibility, there is an 8-bit integer format, in which the 8 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 8.21 Pass Auxiliary Arguments to Opcodes (XARGS) (page 302).

```
  15   14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |    |    |   |  Tx Vert  |  Tx Horz  |   |   |   |
|                   |HCL |FLG|Phi Rho Ldr|Phi Rho Ldr|SQI|RHV|PDP|  XARG1
|---|---|---|---|----|----|---|---|---|---|---|---|---|---|---|---|
```

The **Unfold** parameter selects the dual-PRF unfolding scheme:

```
00 : No Unfolding    01 : Ratio of 2:3
10 : Ratio of 3:4    11: Ratio of 4:5
```

The **ARC** selects the archive output format in which four data bytes (see 8-Bit descriptions below) are packed in two output words per bin as follows:

```
 HighByte  Low Byte
--------------------
|        |         |
|   V    |    Z    |
|        |         |
--------------------

--------------------
|        |         |
|   W    |    T    |
|        |         |
--------------------
```

The remaining data parameters are available in both 8-Bit and 16-bit formats, according to **SOPRM** `Command Input word #2` (see 8.4 Setup Operating Parameters (SOPRM) (page 235)). The same **SOPRM** word configures RVP900 for single or dual polarization. The latter is required for **Kdp**, **PDP**, and **RHV** to be computed properly.

Table 63 **PROC** 8-bit and 16-bit Data Formats

Parameter	Description	8-bit Format	16-bit Format
V	Selects radial velocity data.	Mean velocity, expressed as a fraction of the unambiguous velocity interval, is computed from the unsigned byte **N** as: $V_{\frac{m}{\sec}} = V_{Nyquist} \times \frac{(N-128)}{127.5}$ **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.	Mean velocity in meters per second (m/s) is computed from the unsigned word **N** as: $V_{\frac{m}{\sec}} = \frac{(N-32768)}{100}$ **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

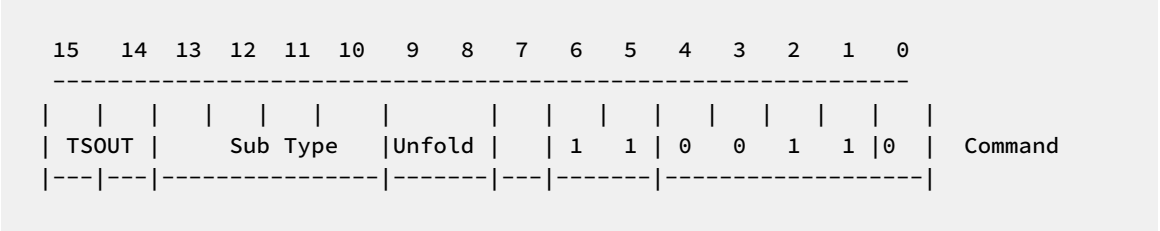
Parameter	Description	8-bit Format	16-bit Format
W	Selects spectral width data.	Spectral width is computed from the unsigned byte N as: $W_{Nyquist} = \frac{N}{256}$ The overall range is a fraction between 1/256 to 255/256 of the unambiguous interval. The code of zero indicates that width data was not available at this range.	Spectral width in meters per second (m/s) is computed from the unsigned word N as: $W_{\frac{m}{\sec}} = \frac{N}{100}$ 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.	The level in decibels is computed from the unsigned byte N as: $dBZ = \frac{(N-64)}{2}$ 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	The level in decibels is computed from the unsigned word N as: $dBZ = \frac{(N-32768)}{100}$ 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	

Parameter	Description	8-bit Format	16-bit Format
ZDR	Selects differential reflectivity data.	The level in decibels is computed from the unsigned byte N as: $dBZ = \frac{(N128)}{16}$ 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	Same as 16-bit decibel format for Z.
KDP	Selects dual polarization specific differential phase data.	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. The maximum value is 150.0 deg*cm/km. A code of zero represents no data 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: $KDP \times \lambda = -0.25 \times 600\left[\frac{127-N}{126}\right]$	Same as 16-bit decibel format for Z, except that the units are hundredths of degrees per kilometer. No weighting by wavelength is introduced.

Parameter	Description	8-bit Format	16-bit Format
PDP	Selects dual polarization differential phase PDP data.	The phase angle in degrees is computed on a 180° interval from the unsigned byte **N** as: $\phi DP(\mathrm{mod}\,180) = 180\frac{(N-1)}{254}$ **0** Indicates no PDP data available at this range **1** 0.00° **254** 179.29° **255** Reserved Code	The phase angle in degrees is computed on a 360° interval from the unsigned word **N** as: $\phi DP(\mathrm{mod}\,360) = 360\frac{(N-1)}{65534}$ **0** Indicates no PDP data available at this range **1** 0.00° **65534** 359.995° **65535** Reserved Code
RHV	Selects dual polarization correlation coefficient RHV data.	The correlation coefficient is computed on the interval 0.0 ... 1.0 using a square root weighting of the unsigned byte **N** as: $P_{HV} = \sqrt{\frac{(N-1)}{253}}$ **0** Indicates no RHV data available at this range **1** 0.0000 (dimensionless) **2** 0.0629 **253** 0.9980 **254** 1.0000 **255** Reserved Code	The correlation coefficient is computed on the interval 0.0 to 1.0 linearly from the unsigned word **N** as: $P_{HV} = \frac{(N-1)}{65533}$ **0** Indicates no RHV 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**.	

Parameter	Description	8-bit Format	16-bit Format
LDR	Selects Linear Depolarization Ratio, measured either on the horizontal receive channel while transmitting vertically, or on the vertical receive channel while transmitting horizontally.	The level in decibels is computed from the unsigned byte N as: `dB = -45.0 + (N-1) / 5` This spans an asymmetric interval around zero decibels, and allows for cross channel isolation as large as 45 dB. The range is from -45.0 ... 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	Same as 16-bit decibel format for Z.
RHO	Selects Signal Quality Index data	This dimensionless parameter uses the same 8-bit and 16-bit data formats as RHV.	
PHI	Selects the cross channel differential phase.	This parameter uses the same 8-bit and 16-bit angular data formats as PDP.	
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	

Parameter	Description	8-bit Format	16-bit Format
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 classification. See sig_data_types.h and HCLASS data description in *IRIS Programming Guide*.	
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 shown below.		
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. The following table shows the values for the random phase major mode the possible values of **Sub Type**.

Table 64 **TSOUT** Random Phase Major Mode Values

First Trip Value	First Trip Description	Second Trip Value	Second Trip Description
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

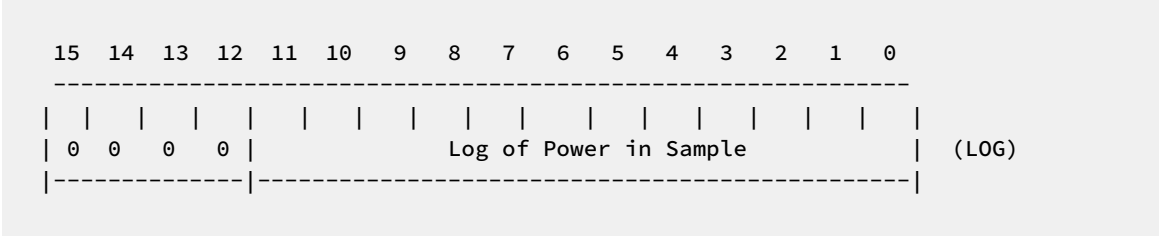
When the **TSOUT** bits select **Time Series** then, a further choice may be needed to select the time series from the first or second pulse when using the Hybrid Pulse Compression scheme. For the Random Phase major mode the possible values of **Sub Type** are:

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**, and so on up to pulse **#N**. 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:

1. Create a 12-bit signed integer in which bits 0 ... 9 are copied from the `Mantissa` field, and bits 10 and 11 are either `01` or `10` depending on whether `S` is `0` or `1`.
2. Multiply this number by 2**(exponent-40), where the exponent field is interpreted as an unsigned 5-bit integer.
3. 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:

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

The resulting voltage span is ± 4 × V_{MAX}. The extra factor of 4 is built into the format so that transient excursions above the full scale input voltage can be encoded properly.

A `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 0.

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 0, 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 vary from 2048 ... 4095. The SNR therefore varies from 66 dB ... 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 scales 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 time series (`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 time series output words, the `Log of Power in Sample`, is provided 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}
: +4.5 dBm for 12-bit IFDR, +6.0 dBm for 14-bit IFDR, +8.0 dBm for 16-bit IFDR

V_{MAX}
: 0.5309 V for 12-bit IFDR, 0.6310 V for 14-bit IFDR, 0.7934 for 16-bit IFDR

`Slope`
: `Log Power Slope` word 3 of the **`SOPRAM`** command. 0.03 is recommended.

For backwards compatibility, RVP900 produces a 8-bit fixed point time series format. Because of the limited dynamic range available, this only shows strong signals, and is not recommended for use. The `I`, `Q`, and `Log power` triplets are packed in two 16-bit output words as follows:

```
 HighByte    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 0 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 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. If you must 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`) produces data in one of two formats according to the `Sum H+V Time Series` question in the **Mp** setup section:

- `Yes` produces summed time series from both channels, but spectra from the DSP is 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 selected in the **Processing/Gen-Setup** window.
- `No` produces the usual (`BxN`) time series output samples, except that the first half of these samples is the first half of the `H` data in their normal order. This is followed by a zero sample if (`BxN`) is odd; followed by the first half of the `V` data, also in their normal order.

Only the first halves of the individual `H` and `V` sample arrays are output by RVP900. As an example, if you select 25 bins and 100 pulses, then the output data consists 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 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 indicates when time series overflows occur. In such cases, the correct number of words are still output, but they are all 0 after the point at which overflow was detected. See 8.10 Get Processor Parameters (GPARM) (page 267)

8.9 Load Clutter Filter Flags (**LFILT**)

In the RVP9 processor, you can choose any available clutter filters 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. You can choose a highly rejective filter for near ranges, and a less rejective, or perhaps no filter, 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 8.3 Load Range Mask (LRMSK) (page 233)) is using the full number of bins. In such cases, the later filter codes are 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. In sum, each filter code is associated with a particular bin number, not with a particular range. The correspondence between bin numbers and 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 4200 words are loaded, otherwise only 512 words are loaded. In both cases the last filter choice is replicated for all bins further in range.

```
  15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|                                       |ALL| 0   1   0   0   0 |  Command
|---------------------------------------|---|-------------------|  (Legacy)
```

```
  15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|                                                   | Filter #1 |  Input 1
|---------------------------------------------------|-----------|
Either 512 or 4200 Words Loaded
```

Enhanced Version

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.

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`) is 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° 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 is 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, if:

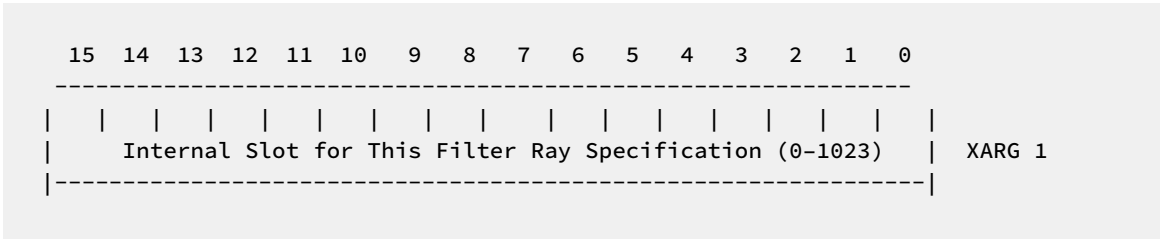
- `Slot-0` defines a default 360° filter-versus-range table.
- `Slot-1` defines special filtering within 0 … 90° that is further modified by ° filters in a 40 … 50° span.

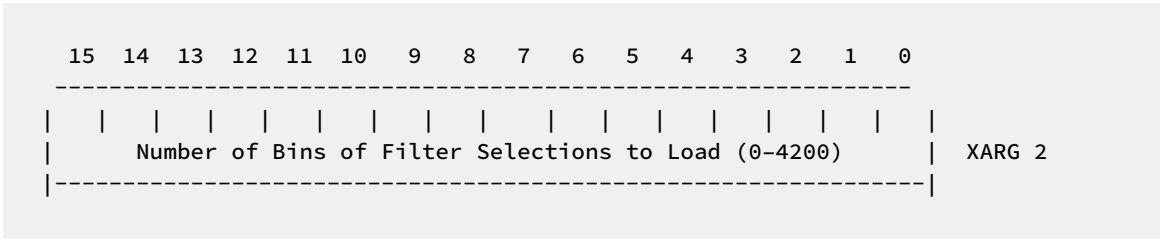
Then a 45° ray would then be filtered according to `Slot-2`, a 60° ray would use `Slot-1`, and a 100° ray would use `Slot-0`.

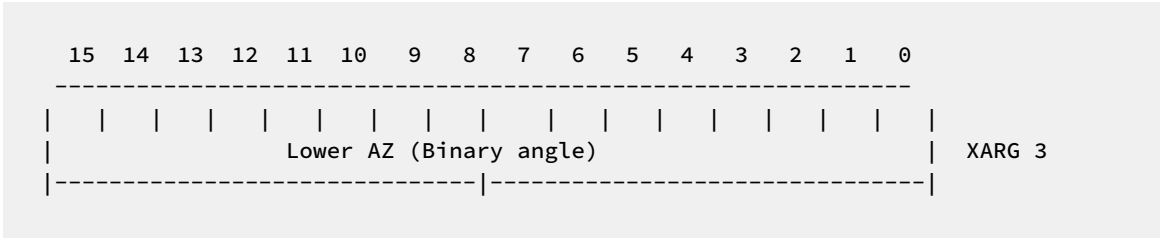
If the `CLR` bit is set in the opcode, then no additional arguments follow and the entire internal filter array (all slots) is invalidated. The result is that no clutter filter is applied to any of the processed data, regardless of range, `AZ`, or `EL`. Moreover, loading a given slot with a table consisting of 0 bins of filter data invalidates just that slot. This allows some data to be removed from the table without having to resort to a complete `CLR` operation.

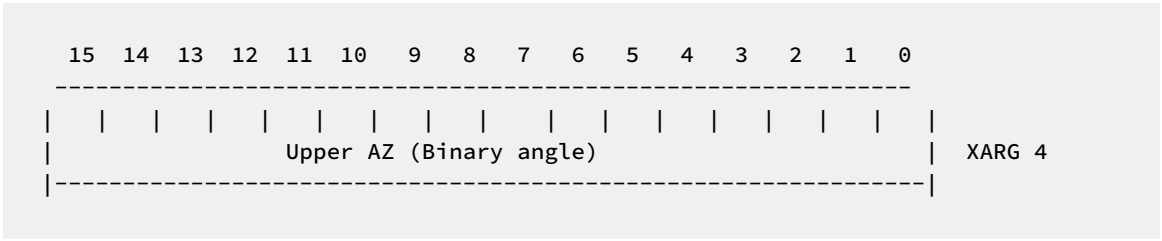
The legacy **LFILT** command is equivalent to calling the enhanced command first with the `CLR` bit set, followed by a second call that writes the legacy filter choices into slot `#0` using `AZ`/`EL` limits that cover all of space. Thus, the legacy behavior is obtained as a special case of the enhanced mechanism.

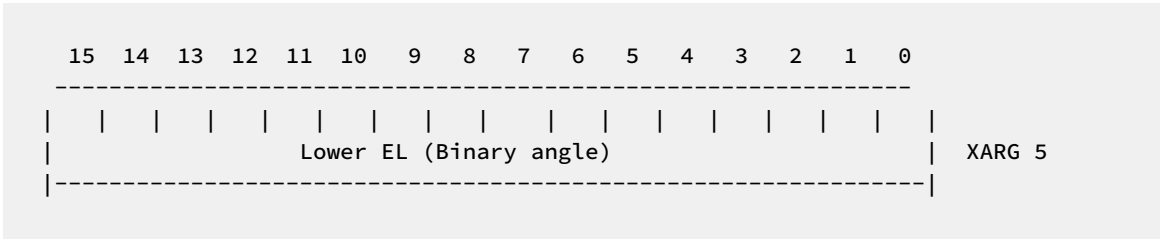
```
  15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|                                     1 |CLR| 0   1   0   0   0 |  Command
|---------------------------------------|---|-------------------|  (Enhanced)
```

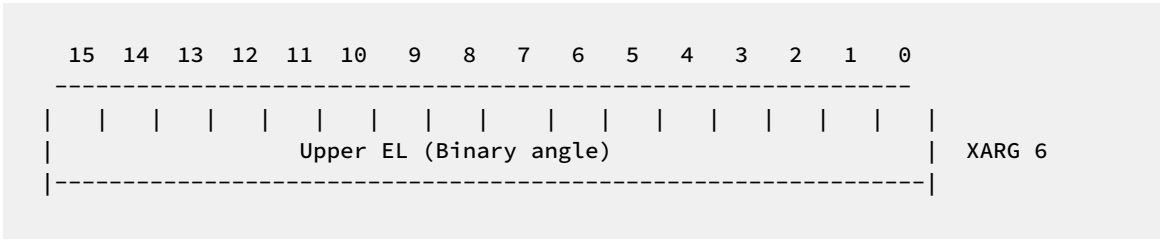

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
Internal Slot for This Filter Ray Specification (0-1023)
XARG 1


15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
Number of Bins of Filter Selections to Load (0-4200)
XARG 2


15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
Lower AZ (Binary angle)
XARG 3


15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
Upper AZ (Binary angle)
XARG 4


15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
Lower EL (Binary angle)
XARG 5


15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
Upper EL (Binary angle)
XARG 6

```
  15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|     Filter for Range Bin #2   |   Filter for Range Bin #1     |  Input 1
|-------------------------------|-------------------------------|

Total of Ceil(NBins/2) Words Loaded
```

8.10 Get Processor Parameters (**GPARM**)

Use **GPARM** to access status information from the RVP900 processor.

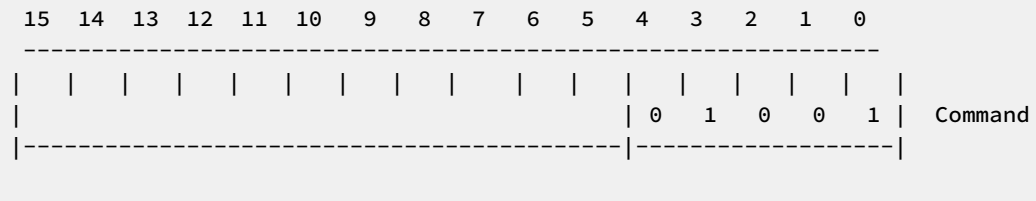
64 words are always transferred. Some words are reserved for future compatibility and are read as zeros.

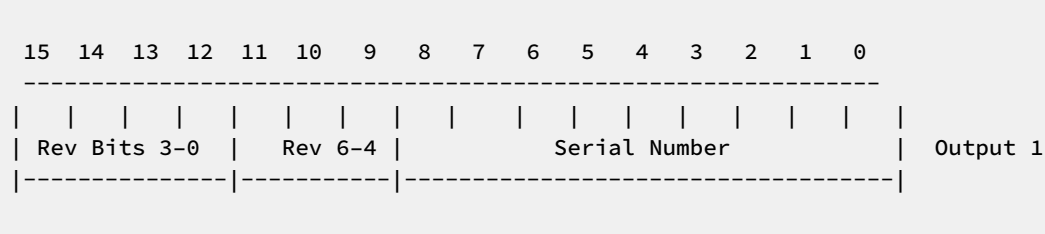
Table 65 RVP900 Status Output Words

Word	Description
1	Revision/Serial number
2	Number of Range Bins
3	Current trigger period
4	Current `TAG00` - `TAG15`
5	Current `TAG16` - `TAG31`
6	Log of Measured Noise Level
7	`I` Channel DC Offset
8	`Q` Channel DC Offset
9	Latched Processor Status
10	Immediate Status Word #1
11	Diagnostic Register A
12	Diagnostic Register B
13	Number of Pulses/Ray
14	Trigger Count (Low 16-bits)
15	Trigger Count (High 8-bits)
16	No. of Properly Acquired Bins
17	No. of Properly Processed Bins
18	Immediate Status Word #2
19	Noise Range in Km

Word	Description
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)
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

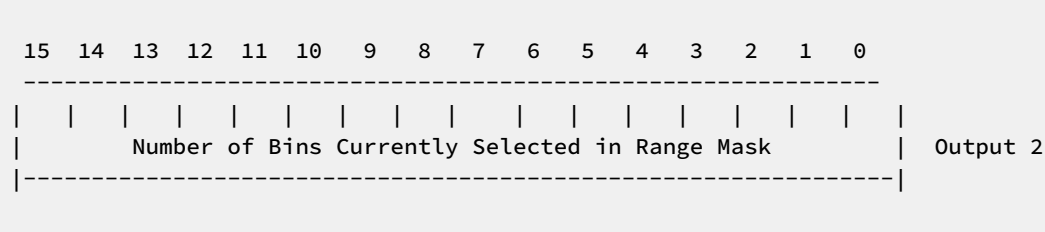
Word	Description
53	Interference Filter C1 Constant
54	Interference Filter C2 Constant
55	Immediate Status Word #3
56	Burst Tracking Slew
57	Polarization Algorithm Choices
58	Range Mask Spacing
59	Immediate Status Word #4
60	Reserved
61	Reserved
62	Reserved
63	Reserved
64	Reserved





Shows the revision and serial numbers of the RVP900 board. This information is useful when computer software is designed to handle many signal processor revisions.

The revision number is 7 bits total, 4 of which are in the high four bits of the word for compatibility with an older format.



```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|   TRIGIN Current Trigger Period in 1/8km (0.83333 usec) Steps  |  Output 3
|---------------------------------------------------------------|
```

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|                 Current Sample of TAG bits 15-0               |  Output 4
|-------------------------------|-------------------------------|
```

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|             Current Sample of TAG bits 31-16                  |  Output 5
|---------------------------------------------------------------|
```

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 0   0 | (MSB)        Log of Measured Noise Level(LSB)          |  Output 6
|-------|-------------------------------------------------------|
```

This value is scaled four times higher than the time series LOG format (see 8.8 Initiate Processing (PROC) (page 251)). To convert to dBm, use the equation:

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

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|            "I" Channel Measured DC Offset                     |  Output 7
|---------------------------------------------------------------|
```

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|            "Q" Channel Measured DC Offset                     |  Output 8
|-------------------------------|-------------------------------|
```

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.

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|     Latched Status Word (Bits Cleared After Each Access)       |  Output 9
|----------------------------------------------------------------|
```

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 µsec 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.

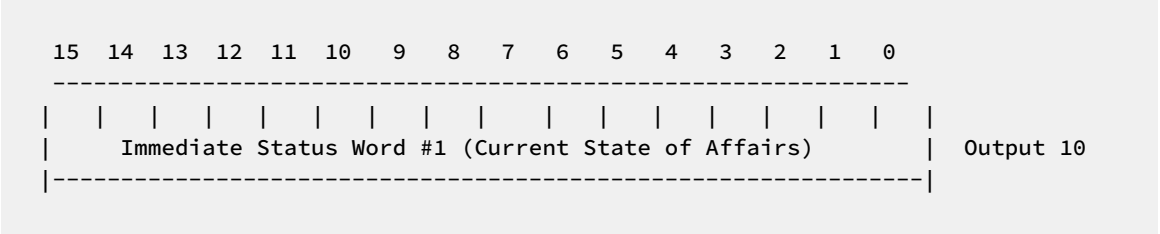
Bit 10
Error in **LSIMUL** command protocol

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 8.16 Load Antenna Synchronization Table (LSYNC) (page 293).

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

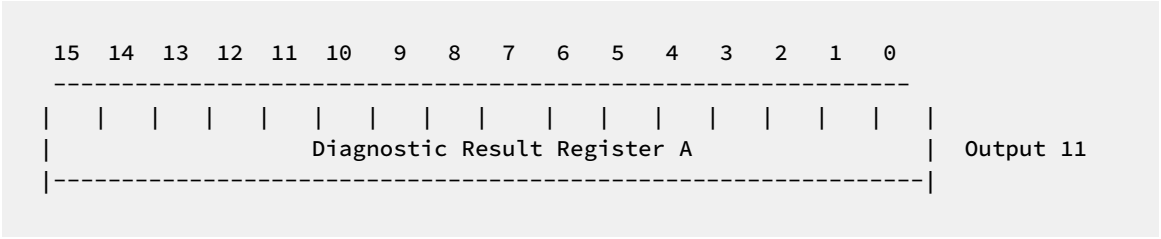
Current unfolding mode

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`
: RVP running without `root` privileges

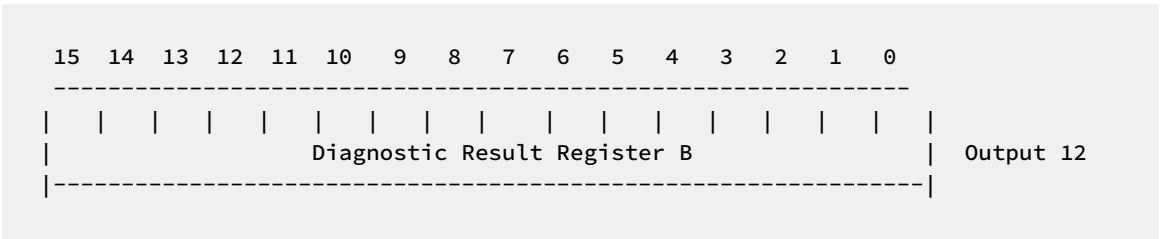
`Bit 5`
: Problem creating daemon process

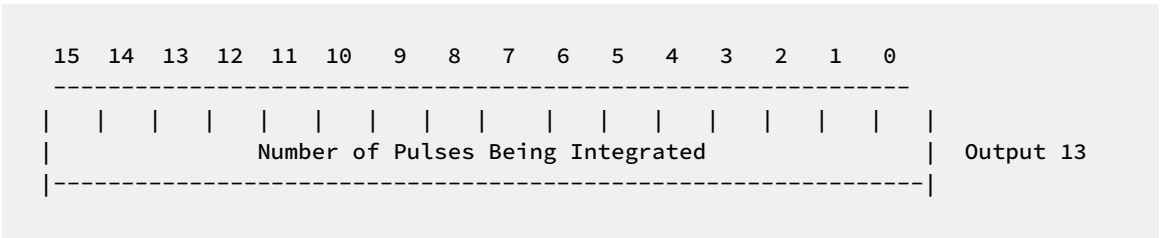
`Bit 6`
: Inconsistent setup values detected

`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





```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|                Trigger Count (Low 16-bits)                    |  Output 14
|---------------------------------------------------------------|
```

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|                Trigger Count (high 8-bits)                    |  Output 15
|---------------------------------------------------------------|
```

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.

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| Number of Properly Acquired Bins for Current Range Mask & PRT  |  Output 16
|---------------------------------------------------------------|
```

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|    No. of Valid Bins in Initial Part of Ray From Last PROC Cmd |  Output 17
|---------------------------------------------------------------|
```

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|    Immediate Status Word #2 (Current State of Affairs)         |  Output 18
|---------------------------------------------------------------|
```

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 flags when the burst pulse has drifted out of its optimal placement within the sampling window.

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|    Starting Range in Km at Which Noise Sample Data are Taken   |  Output 19
|---------------------------------------------------------------|
```

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| Trigger Period (0.16667usec Increments) During Noise Sampling  |  Output 20
|---------------------------------------------------------------|
```

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|   Min Trig Period (0.16667usec Increments) for Pulse Width 0   |  Output 21
|---------------------------------------------------------------|
```

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|   Min Trig Period (0.16667usec Increments) for Pulse Width 1   |  Output 22
|---------------------------------------------------------------|
```

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|   Min Trig Period (0.16667usec Increments) for Pulse Width 2   |  Output 23
|---------------------------------------------------------------|
```

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|  Min Trig Period (0.16667usec Increments) for Pulse Width 3    |  Output 24
|---------------------------------------------------------------|
```

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|    Four 4-bit Control Bit Patterns for Each Pulse Width        |  Output 25
|---------------------------------------------------------------|
```

For definitions of these bits, see **`input word #1`** in 8.14 Define Pulse Width Control and PRT Limits (PWINFO) (page 290) .

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|  PWidthS      |Hyb|  Window   |     TopMode    |    PWidth     |  Output 26
|---------------|---|-----------|----------------|---------------|
```

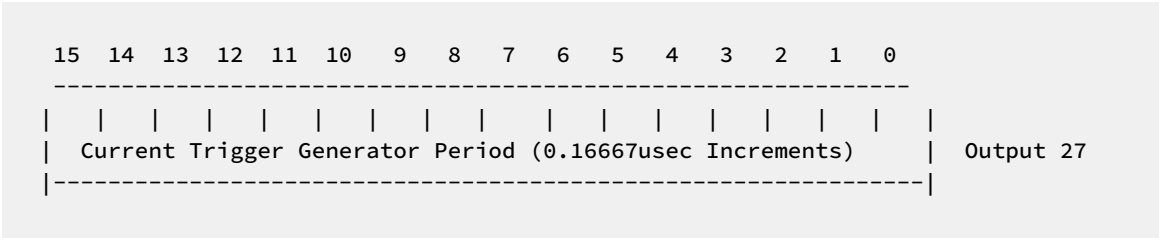
`PWidth`
Currently selected radar pulse width

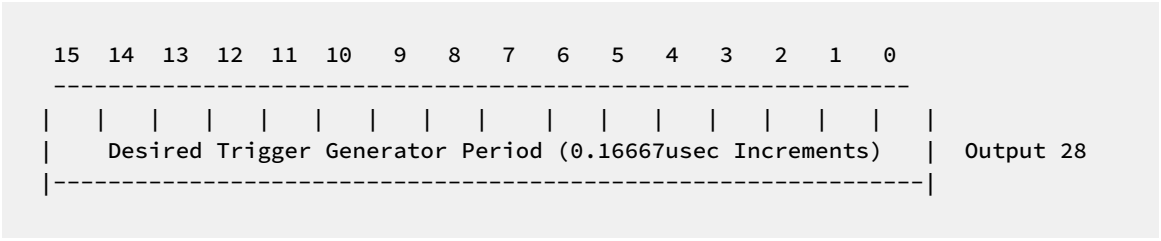
`TopMode`
: Major Mode. See **`Input #9`** in 8.4 Setup Operating Parameters (SOPRM) (page 235).

`Window`
: Spectral Window Choice. See **`Input #10`** in 8.4 Setup Operating Parameters (SOPRM) (page 235).

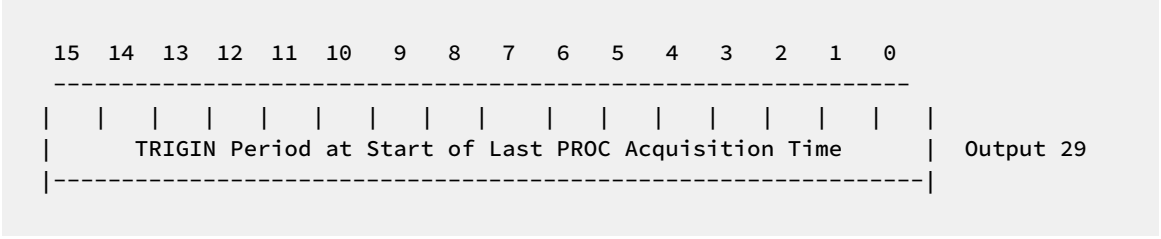
`PWidthS`
: Pulse width of second pulse in hybrid transmit waveform

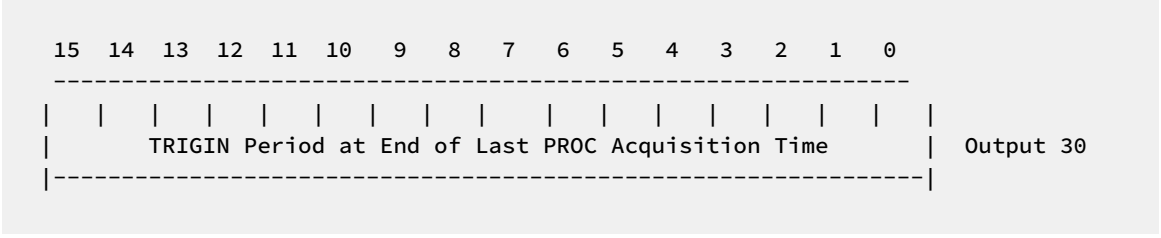
`Hyb`
: Bit indicating second pulse in use in hybrid transmit waveform





The desired trigger generator rate is that which was selected in the most recently issued **`SETPWF`** command (or power-up rate if **`SETPWF`** was not 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) when the external trigger is expected, but missing.





The PRTs from the start and end of the last ray are the measured values when 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`.

`Output 31 ... Output 37` are the current processing and threshold parameters set by **`SOPRM`**. See 8.4 Setup Operating Parameters (SOPRM) (page 235).

Since the threshold levels for each data parameter can be different (see 8.30 Set Individual Thresholds (THRESH) (page 311)), words `33 ... 36` are taken from the velocity parameter.

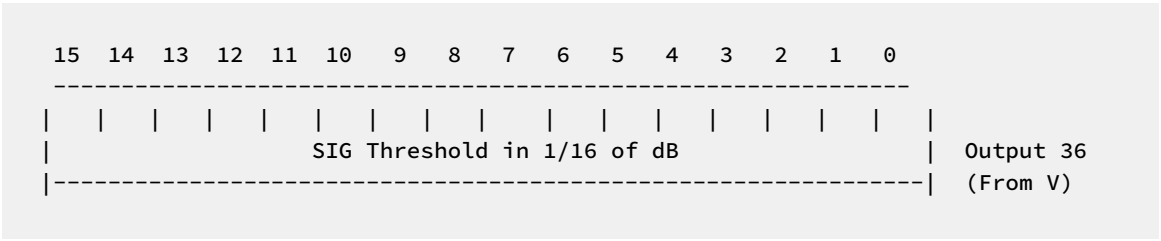
```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|       | Polar |NHD|ASZ|16B|CMS| R2 |   |3x3|   |   |Lsr|Dsr|Rnv|  Output 31
|-------|-------|---|---|---|---|----|---|---|---|---|---|---|---|
```

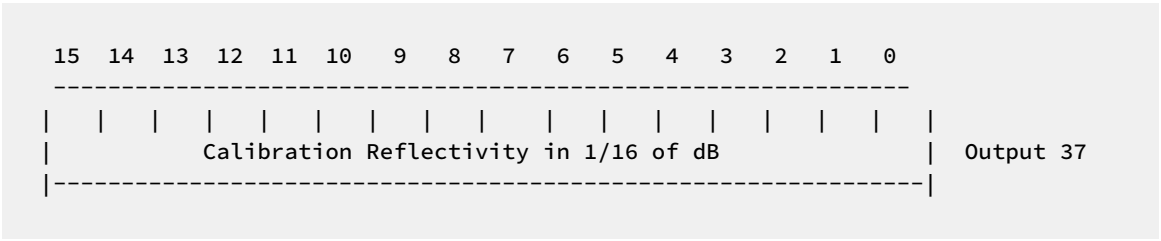
```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|                     Log Slope65536 * dB / LSB                 |  Output 32
|---------------------------------------------------------------|
```

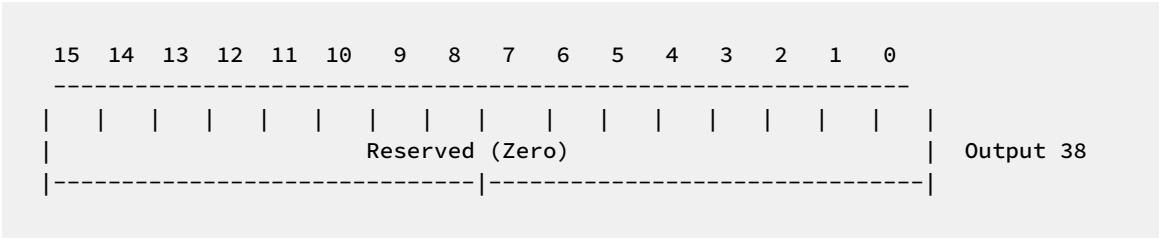
```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|                  LOG Noise Threshold in 1/16 of dB            |  Output 33
|---------------------------------------------------------------|  (From V)
```

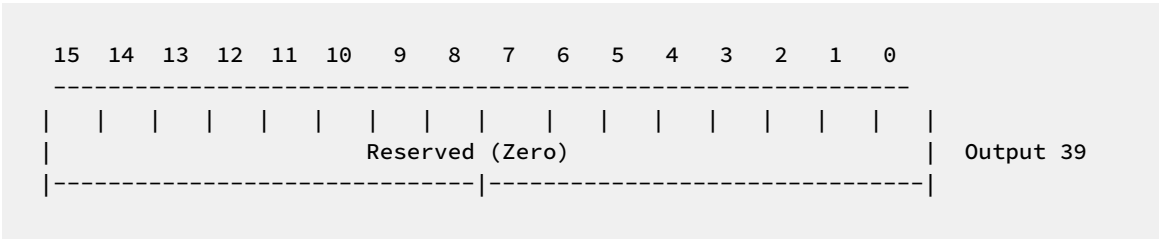
```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|       Clutter Correction (CCOR) Threshold in 1/16 of dB       |  Output 34
|---------------------------------------------------------------|  (From V)
```

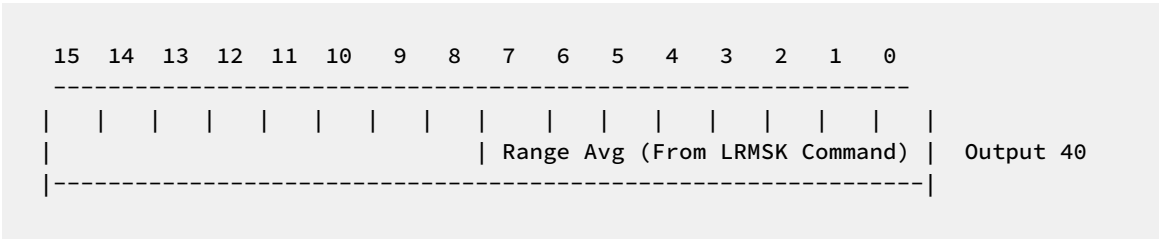
```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|                             |SQI Threshold                    |  Output 35
|---------------------------------------------------------------|  (From V)
```

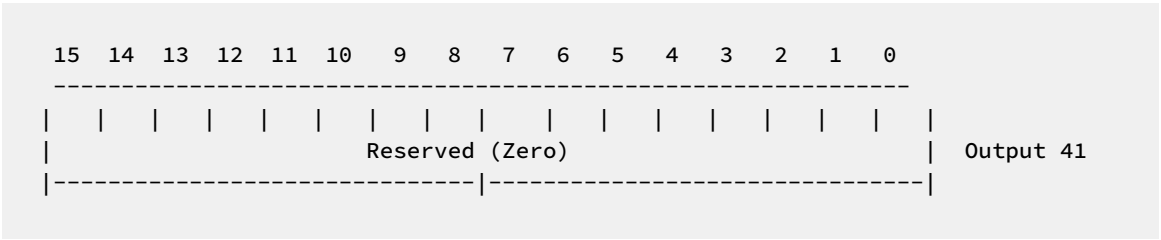

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
SIG Threshold in 1/16 of dB
Output 36
(From V)


15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
Calibration Reflectivity in 1/16 of dB
Output 37


15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
Reserved (Zero)
Output 38


15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
Reserved (Zero)
Output 39


15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
Range Avg (From LRMSK Command)
Output 40


15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
Reserved (Zero)
Output 41

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|                      Reserved (Zero)                          |  Output 42
|-------------------------------|-------------------------------|
```

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|        Header Config of PROC data (CFGHDR Input #1)           |  Output 43
|---------------------------------------------------------------|
```

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|           Noise Sum of I Squared   =2**-16LSB=2**-31          |  Output 44
|---------------------------------------------------------------|
```

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|           Noise Sum of I Squared   MSB=1LSB=2**-15            |  Output 45
|---------------------------------------------------------------|
```

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|          Noise Sum of Q Squared  MSB=2**-16LSB=2**-31          |  Output 46
|---------------------------------------------------------------|
```

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|          Noise Sum of Q Squared  MSB=1LSB=2**-15              |  Output 47
|---------------------------------------------------------------|
```

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

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

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

from which we obtain:

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

The four integer values become rather small and severely quantized when the noise power drops to low values. Previously, these words helped balance the individual gain of the `I` and `Q` channels in 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.

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|        Log of Measured Noise Level (same as word 6)           |  Output 48
|---------------------------------------------------------------|
```

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|    Vert Noise Stdev in dB*10  |  Horiz Noise Stdev in dB*10    |  Output 49
|---------------------------------------------------------------|
```

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.

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| Ratio of Horizontal/Vertical Noise Power in Hundredths of dB   |  Output 50
|---------------------------------------------------------------|
```

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|           16-Bit AFC/MFC Value (-32768 through +32767)         |  Output 51
|---------------------------------------------------------------|
```

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |
| PhaseSeq      | IFD Sat.Power |    MinRev      |    Inter.F   |  Output 52
|---------------------------------|----------------|--------------|
```

`Inter.F`
Specifies which interference filter is running. Zero means "none". For information on interference filter algorithms, see 7.2.5 Interference Filter (page 176).

`MinRev`
Minor revision level of the RVP900 code that is currently running

`IFDR Sat.Power ( PMAX )`
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
- 2: +8.0 dBm

`PhaseSeq`
Tx Phase modulation sequence. See 8.28 Configure Phase Modulation (CFGPHZ) (page 309).

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|    Interference Filter Parameter “C1” in Hundredths of dB      |  Output 53
|----------------------------------------------------------------|
```

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|    Interference Filter Parameter “C2” in Hundredths of dB      |  Output 54
|----------------------------------------------------------------|
```

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  |
|       Immediate Status Word #3 (Current State of Affairs)     |  Output 55
|-------------------------------|---------------|--------------|
```

`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 ... 4 are supported

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|         Signed trigger slew in hundredths of microseconds      |  Output 56
|---------------------------------------------------------------|
```

This is the same format that is used by the **SETSLEW** command to set the current trigger slew. See 8.26 Set Trigger Timing Slew (SETSLEW) (page 308).

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|               Polarization Algorithm Choices                  |  Output 57
|---------------------------------------------------------------|
```

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

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|         Range mask spacing for current pulsewidth (cm)         |  Output 58
|---------------------------------------------------------------|
```

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|    Immediate Status Word #4 (Current State of Affairs)        |  Output 59
|---------------------------------------------------------------|
```

`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 RVP900

`Bit 7`
: WSR88D Batch mode is supported

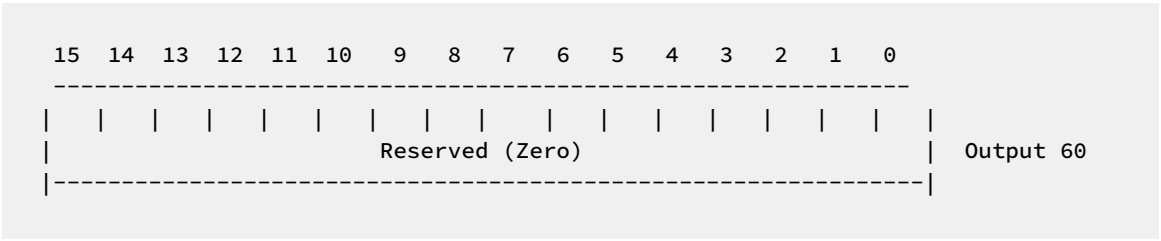
`Bit 8`
: Major mode refuses to use external trigger

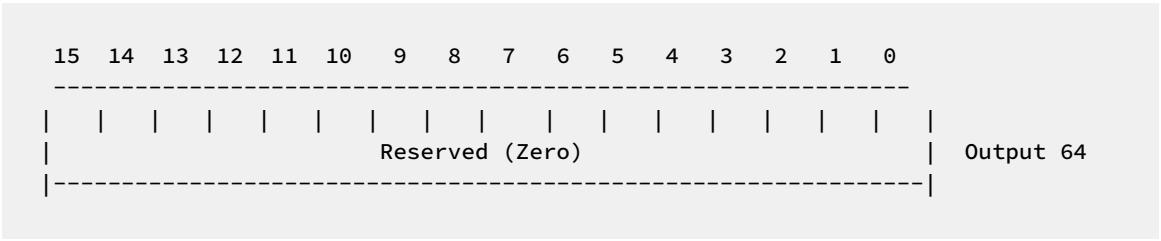
`Bit 9`
: **GPARM** outputs `#7` and `#8` use Hi-SNR format, else linear

`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





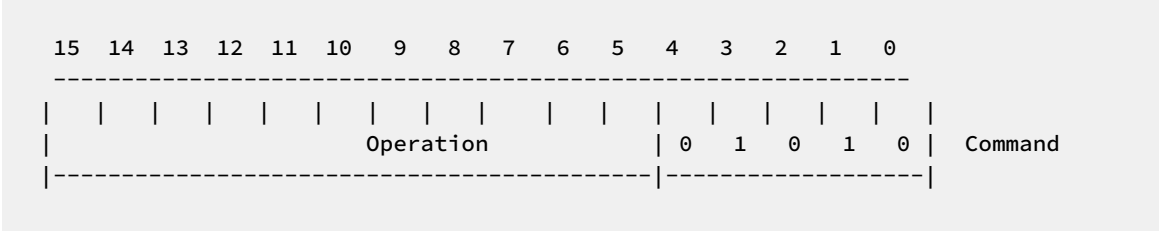
8.11 Load Simulated Time Series Data (**LSIMUL**)

LSIMUL acts 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 waits for `N` (`N` = sample size) **LSIMUL** commands (with `Operation=2`) prior to outputting each ray. The arrival of any other command during that time causes the simulated data mode to be exited, and error bit `#10` is 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 use the data supplied in the next `N` (`N` = sample size) **`LSIMUL`** with `Operation= 2`.
The receiver noise and offset levels which are internally maintained by 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. 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 RVP900 replicates these data as necessary in order to fill out the entire count of acquired bins. If the number of bins is 0, then a zero-valued sample is applied for all channels.

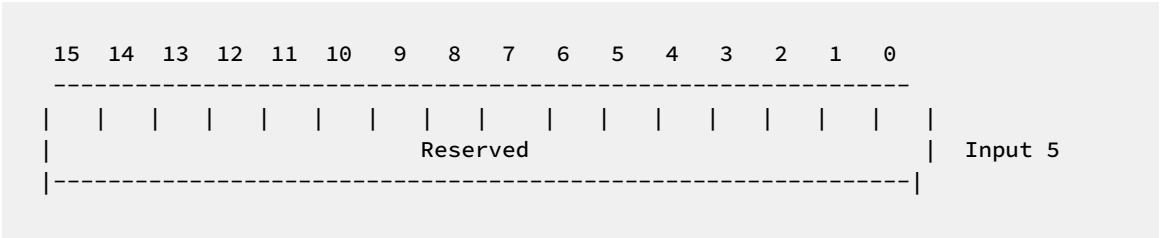
```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|          Number of Bins of Simulated Data Which Follow        |  Input 1
|---------------------------------------------------------------|
```

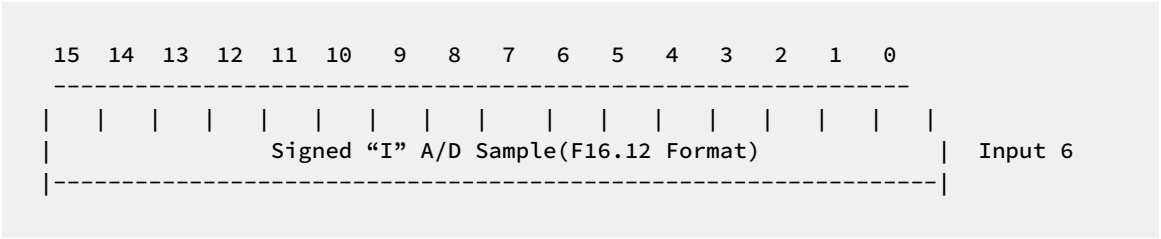
```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|         Transmit Phase of This Pulse (16-bit Binary Angle)    |  Input 2
|-----------------------------------------------|---------------|
```

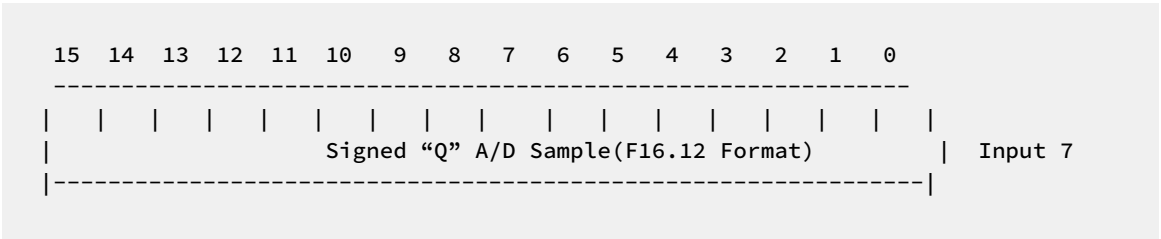
```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|        Transmit Power in Hundredths of dB (Zero dB Nominal)   |  Input 3
|---------------------------------------|---|-------------------|
```

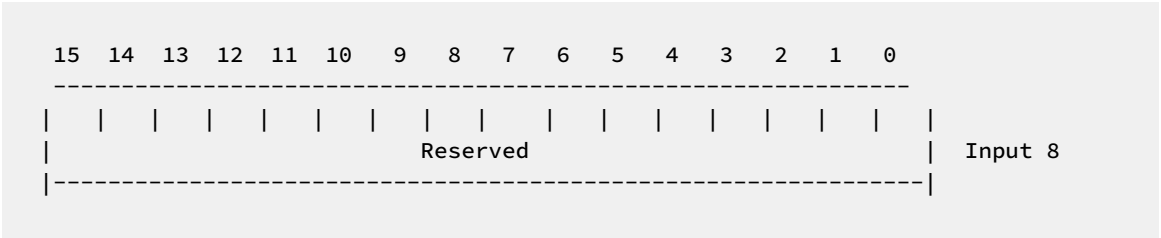
```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|                          Reserved                             |  Input 4
|-----------------------------------------------|---------------|
```

In the legacy format #2 (RVP5-RVP900) each bin within the pulse is represented by four 16-bit fixed point words. The total number of words loaded is (4+4B), where B is the bin count specified in **Word #1**. This takes account the 4 header words, plus 4 words for every bin being defined.



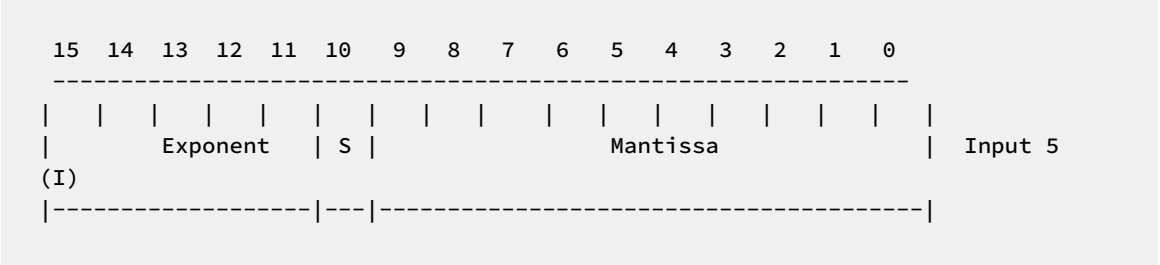






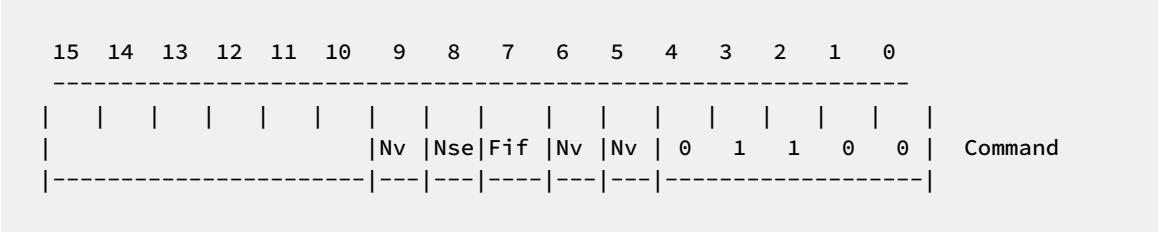
In format #3 each bin within the pulse is represented by two 16-bit floating point words having the exact same format as the packed (I,Q) time-series data that are output by the **PROC** command . See 8.8 Initiate Processing (PROC) (page 251).

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**.



8.12 Reset (**RESET**)

The **RESET** command permits resetting either the entire RVP900 processor, or selected portions. Flags in the command word determine the action to be taken.



`Nv`
: Reloads configuration from the saved nonvolatile settings.

`Nse`
: Reset the receiver noise levels to the power-up default value for all pulsewidths as defined in the **Mt** setup questions. See 5.2.6 Mt<n> — Triggers for Pulsewidth n (page 117).

`Fif`
: Remove any data currently in the output FIFOs. This permits flushing output data that was left from a previous command, so that new output can be read from scratch. See 8.1.2 First-in-first-out (FIFO) buffer (page 232).

8.13 Define Trigger Generator Waveforms (**TRIGWF**)

CAUTION! Do not use **TRIGWF** in any new code applications that drive RVP900. Use the interactive trigger setup procedure to define RVP900 triggers and timing. See 6.5 Pb — Plot Burst Pulse Timing (page 134).

TRIGWF is obsolete . It is included for backward compatibility with RVP6. The code is disabled by default.

RVP900 has a built-in trigger generator that can synthesize 6 independent digital output waveforms, each having arbitrary shape and being active anywhere in a window centered around zero-range.

The 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 6 trigger lines during each radar pulse.

The outputs can be used for transmitter triggers, scope triggers, range strobes, PLL gates, and so on. The writable waveform table is unique, in that the detailed timing of trigger and related control signals can be easily adjusted in software, without having to resort to reprogramming **PROM**s. This makes it possible for user software to edit the trigger timing interactively.

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

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|               |PW3|PW2|PW1|PW0|           | 0   1   1   0   1 |  Command
|---------------|---|---|---|---|-----------|-------------------|
```

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| H |               | Transition Point of POLAR0 Control Line   |  Input 1
|---|---------------|-------------------------------------------|
```

The **H** bit defines the sense of the control line when horizontal polarization is selected.

Inputs **2 ... 2049**, indicate bits output every 7.195 MHz and **Input 1025** corresponds to the time at which data at range zero are sampled.

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|                                       | Bits for -142.22 usec |  Input 2
|---------------------------------------|-----------------------|
```

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|                                       |  Bits for zero range  |  Input 1025
|---------------------------------------|-----------------------|
```

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|                                       | Bits for +142.08 usec |  Input 2049
|---------------------------------------|-----------------------|
```

8.14 Define Pulse Width Control and PRT Limits (**PWINFO**)

RVP900 can control the radar transmitter's pulse width and corresponding receiver bandwidth.

There are 16 pulse/bandwidth codes, numbered `0` ... `15`. The association between codes and pulse widths is determined by the needs and capabilities of each radar. In some cases, code `0` can represent 0.25 microsecond pulse width, and in other cases it can represent 2.0 microseconds, and some radars may use all sixteen codes while others provide fewer options.

The **PWINFO** command defines what happens for each pulse width code. **SETPWF** defines selects which code is used.

The **PWINFO** command loads four codes at a time according to the `UpperPW` bits: `00` loads codes `0` ... `3`, `01` loads codes `4` ... `7`, and so on.

RVP900 drives four TTL output lines (**PWBW0 – 3**) which control the radar pulse/bandwidth hardware. Typically this control is through relays or solid-state switches in the transmitter and receiver. The user decides what state the four lines assume for each pulse width code 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 limits the transmitter duty cycle to a safe value under all conditions. **PWINFO** sets up these minimum PRTs using words 2 ... 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 if that **SETPWF** accidentally selects one of the unused states. If the internal trigger generator is not being used, then the PRT limits no longer affect the trigger rate and transmitter protection becomes the responsibility of the user hardware.

You can turn off the pulse/bandwidth mechanism 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 accidentally be changed by the host computer. When this is done RVP900 still reads the five input words, but no changes are made to the pulse width and PRT information. The command I/O behaves the same way, whether enabled or disabled.

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|                      |UpperPW|           | 0   1   1   1   1 |  Command
|----------------------|---|---|-----------|-------------------|
```

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| Bits for PW 3 | Bits for PW 2 | Bits for PW 1  | Bits for PW 0 |  Input 1
|---------------|---------------|----------------|---------------|
```

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|   Min Trig Period (0.16667usec Increments) for Pulse Width 0   |  Input 2
|---------------------------------------------------------------|
```

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|   Min Trig Period (0.16667usec Increments) for Pulse Width 1   |  Input 3
|---------------------------------------------------------------|
```

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|   Min Trig Period (0.16667usec Increments) for Pulse Width 2   |  Input 4
|---------------------------------------------------------------|
```

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|   Min Trig Period (0.16667usec Increments) for Pulse Width 3   |  Input 5
|---------------------------------------------------------------|
```

8.15 Set Pulse Width and PRF (**SETPWF**)

SETPWF 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 16 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.

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|       |UpperPW| (Rsv) |LowerPW|           | 1   0   0   0   0 |  Command
|---------------|-------|---|---|-----------|-------------------|
```

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

LowerPW
: Lower two bits of overall 4-bit pulse width selection

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|    Desired Trigger Generator Period (0.16667usec Increments)   |  Input 1
|---------------------------------------------------------------|
```

When `Input #1` is zero, 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. RVP900 generates triggers with 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.

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|       Lower 16-Bits of 32-Bit Trigger Period in Nanoseconds    |  XARG 1
|---------------------------------------------------------------|
```

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|       Upper 16-Bits of 32-Bit Trigger Period in Nanoseconds    |  XARG 2
|---------------------------------------------------------------|
```

8.16 Load Antenna Synchronization Table (**LSYNC**)

RVP900 can operate in a mode where radar data are acquired in synchronization with the antenna motion along the azimuth or elevation axis. This means that the user computer does not need to separately monitor the antenna angles and request each data ray individually.

To use this mode, `TAG0-15` must be wired to receive azimuth angles and `TAG15-31` must be wired to receive elevation. Angle input may be 16-bit binary angles or 4-digit BCD. This synchronization mode is the only one that ascribes meaning to TAG inputs (usually they are only passed on to the user computer as ancillary information).

Antenna synchronization is done using a table of trigger angles. This table, which contains 3 ... 4096 angles, defines the angle boundaries for each processed ray. The trigger angles do not need to be uniformly spaced nor must they span the full 360° rotation. This gives flexibility in the choice of angles. For example, if local obstructions cause shadows in the radar image, then those regions can be skipped by omitting table entries in their vicinity.

Likewise, as the antenna rotates, data can be acquired within one or more sectors by specifying the appropriate sets of contiguous bearings at the desired angular resolution. On power-up, the angle table is initialized to 360 values corresponding to half-integer-valued degrees from 0.5 ... 359.5°.

The synchronization algorithm works automatically with clockwise or counterclockwise antenna rotation, and can tolerate any sequence of changes in direction, for example, if the antenna scans a sector, or turns erratically. The trigger angles do not have to be hit exactly to start each new ray – the antenna only needs to move across them. This minimizes the possibility of losing data due to missing codes in the angle encoders. RVP900 automatically produces an output ray after one second of waiting, even if no trigger angles have been crossed. This prevents time-outs with the host computer when the antenna is not moving.

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|       |Dyn|Sht|Ena|El |BCD|Ld |           | 1   0   0   0   1 |  Command
|-------|---|---|---|---|---|---|---|--------|----|--------------|
```

`Dyn`
: If set, the endpoints of each ray are synchronized. Their angular width can vary by what is required to collect the **SOPRM** number of pulses.
If clear, 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 are 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, where the pulse count is less than expected due to encountering a feature in the CPI (usually a trigger transition) that would normally result in the entire ray being discarded.
When this bit is set, the user's code must check the 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. Otherwise, `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:

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|         Number of Trigger Angles that Follow (3 – 4096)       |  Input 1
|---------------------------------------------------------------|
```

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|                          (Ignored)                            |  Input 2
|---------------------------------------------------------------|
```

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|          First Trigger Angle (16-Bit Binary Angle)            |  Input 3
|---------------------------------------------------------------|
```

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|          Last Trigger Angle (16-Bit Binary Angle)             |  Input 3
|---------------------------------------------------------------|
```

You must configure the system to use the synchronization mode.

1. Use the **LSYNC** command to load the trigger angle table.
   a. Choose the number of table entries.
   b. Write the required number of words to RVP900.
      Supply the angles clockwise in a strictly increasing order. They must neither reach nor pass 0° by the table's end. The first value may be 0.
      Use binary angle representation, where Bit 15 represents 180°, Bit 14 represents 90°, and so on.
   c. Set the **Ld** bit command word to indicate that a new table size and set of angles are being loaded.
   d. Set a flag bit to be set if errors are detected when loading the table of angles.

   See 8.10 Get Processor Parameters (GPARM) (page 267).

2. Enable synchronized operation by setting the **Ena** command bit. Set or clear, `EL` and `BCD` according to your needs.
   These bits may be used independently of reloading the table values. Thus, antenna synchronization may be turned on and off without having to reload the table each time. If there are errors when the table was last loaded, the processor ignores the `Ena` bit and synchronization is forced off.
   Once enabled, **PROC** commands are 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 automatically aligned with the trigger angles.
   The angle sync algorithm is dynamic and works as follows:
   a. Each ray begins immediately upon the user's request, or upon completion of the previous ray when in continuous processing mode.
   b. At the start of the ray, RVP900 finds the pair of sync angles that enclose the previous trigger angle.
   c. The current ray then runs until the antenna passes outside of either limit, at which point processing for that ray is terminated.
   d. Once this happens, a new trigger angle is assigned based on which limit was crossed.
3. In the `Sample Size` field of the **SOPRM** command, specify the maximum number of pulses present in each ray during angle syncing .
   This is the number of pulses that used when `Dyn=1`.
   When `Dyn=0`, the number of pulses may be less if a trigger angle is crossed before the full pulse count can be accumulated

8.17 Set or Clear User LED (**SLED**)

SLED 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 5. TTY Non-volatile Setups (page 89)), in which case this command has no effect.

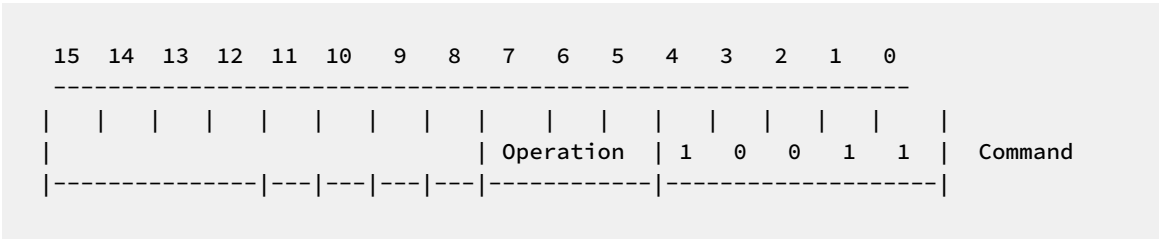
```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|                         |LED|           | 1   0   0   1  0  |  Command
|-------------------------|---|-----------|-------------------|
```

8.18 TTY Operation (**TTYOP**)

TTYOP 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 TTY. Whatever you can do in the TTY, you can also do with this command.

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:

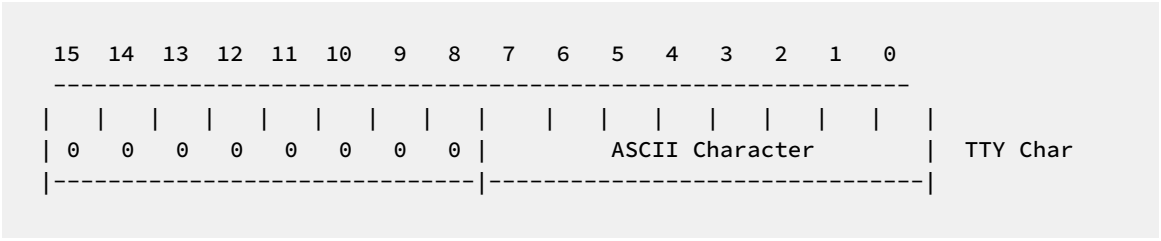
`0`
: Sends the ASCII character in the upper byte of the word to the RVP900 as if it had been typed on the setup TTY keyboard.

`1`
: Allow scope plotting data to be output when a plot is being drawn. All relevant status and data words are output once upon each receipt of this command. Subsequently, status and data is output only when a change has taken place.

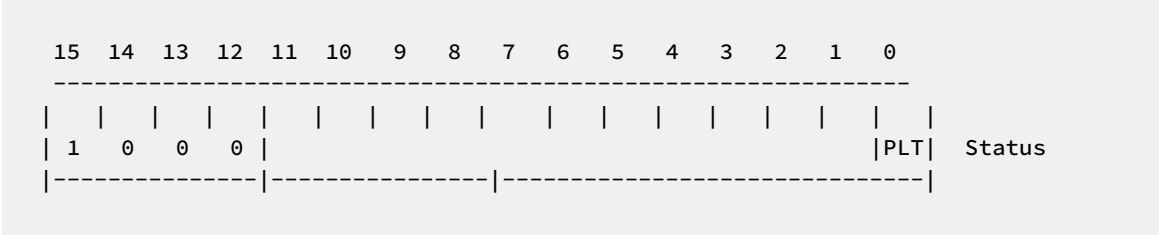
`2`
: Disable the scope plotting output data.

Any of the following types of data may be output by RVP900 while the TTY monitor is running. The order of arrival of each data type is indeterminate, but all multi-word sequences are always be output as contiguous words.

Individual TTY characters generated by 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 **0** represents OFF; **1**, **2**, and **3** are successively brighter.

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 1   0   0   1 | 0   0   0   0 | Int 3  | Int 2 | Int 1 | Int 0 |
|---------------|---------------|--------|-------|-------|-------|
```

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 1   0   0   1 | 0   0   0   1 | Int 7  | Int 6 | Int 5 | Int 4 |
|---------------|---------------|--------|-------|-------|-------|
```

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 1   0   0   1 | 0   0   1   0 | Int 11 | Int 10| Int 9 | Int 8 |
|---------------|---------------|--------|-------|-------|-------|
```

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 1   0   0   1 | 0   0   1   1 | Int 15 | Int 14| Int 13| Int 12|
|---------------|---------------|--------|-------|-------|-------|
```

The data for each stroke of the plot is given by the following sequence of 501 words.

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 1   0   1   0 |                               | Stroke Number |  Plot Data
|---------------|-----------------------------------------------|
```

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 1   0   1   1 |          Value to Plot (0 – 4095)             |  Word #1
|---------------|-----------------------------------------------|
```

```
15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 1   0   1   1 |           Value to Plot (0 – 4095)            |  Word #500
|---------------|---------------|--------|-------|-------|-------|
```

8.19 Load Custom Range Normalization (**LDRNV**)

Reflectivities computed by RVP900 are usually 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 for applications that require an alternate custom range correction, for example, if an external user-supplied STC waveform drives the radar receiver's LNA.

LDRNV loads a 251-word custom correction table holding values in hundredths of decibels over 5 decades of `log(range)` from 0.01km ... 1000km. There are 50 table entries per decade of range. 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 `40(N - 101)`.

The table values are stored and interpolated when RVP900 loads a new range mask. Custom values for the user ranges are then computed. See 8.3 Load Range Mask (LRMSK) (page 233).

The **LDRNV** command must be issued only once, before choosing the working set of range bins.

The linear intervening gas attenuation correction (see 8.4 Setup Operating Parameters (SOPRM) (page 235)) is always added to the reflectivity data, regardless of whether default or custom range normalization is enabled. If this is undesirable, set the intervening gas slope to 0.

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|                                           | 1   0   1   0   1 |  Command
|-------------------------------------------|-------------------|
```

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|  Signed Hundredths dB of Range Normalization for Range 0.01 km |  Input 1
|---------------------------------------------------------------|
```

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|  Signed Hundredths dB of Range Normalization for Range 1000 km |  Input 251
|---------------------------------------------------------------|
```

8.20 Read Back Internal Tables and Parameters (**RBACK**)

RBACK permits some RVP900 internal tables to be read back for confirmation and diagnostic purposes. This command is not generally used during normal data acquisition and processing.

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|             Data to Show                  | 1   0   1   1   0 |  Command
|---------------|---|---|---|---|-----------|-------------------|
```

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|                 Number of words to output                     |  Input 1
|-------------------|-------------------------------------------|
```

Table 66 Data Returned by RBACK

Data Number	Description
0	Full operational parameter table from last **SOPRM** command.

Data Number	Description
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: • Number of samples that went into the ray • Time since the last ray (in tenths of ms) • Ending azimuth TAG bits • Ending elevation TAG bits • Starting azimuth TAG bits • 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: • Azimuth (TAG bits 0–15) • 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. See 8.30 Set Individual Thresholds (THRESH) (page 311). 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. 16-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 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`.

Data Number	Description
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 *`dpolapp_*-band.conf`* file. It is impractical to save all modified parameters as metadata. The string of 16 characters is reported in a sequence of 8 words, each reporting 2 characters.

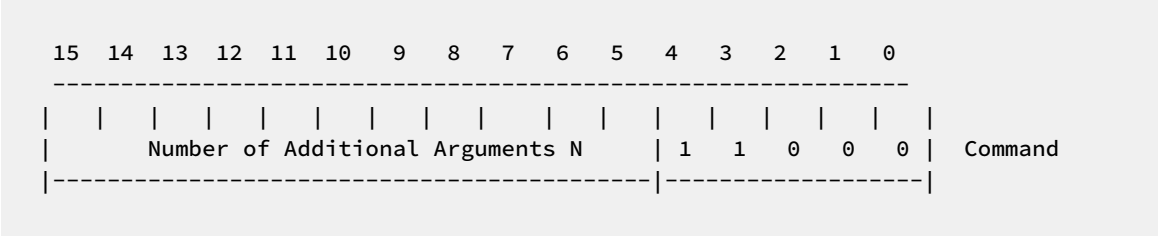
8.21 Pass Auxiliary Arguments to Opcodes (**XARGS**)

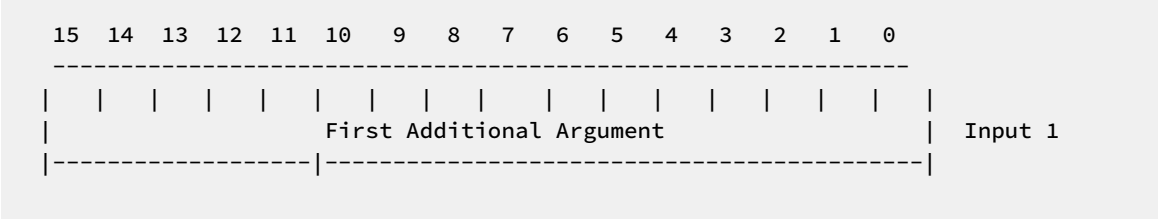
XARGS provides a backward compatible mechanism for supplying additional (optional) arguments to other opcodes. The command may be used freely in the RVP900 instruction stream, even if the opcode being modified does not expect any optional arguments. **XARGS** is a **NOP** in that case.

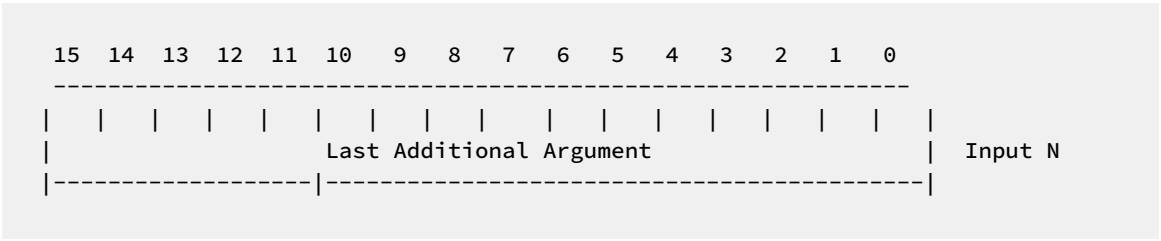
To supply optional arguments to another opcode **OP**, the **XARGS** command is first executed with the additional argument count encoded in its upper 11-bits. This is followed by the array of 0 ... 2047 additional arguments. At this point the **XARGS** command finishes and the **OP** command is fetched as the next instruction. **OP** executes normally, except that the additional arguments from **XARGS** can be picked up after its own input list has been read to completion.

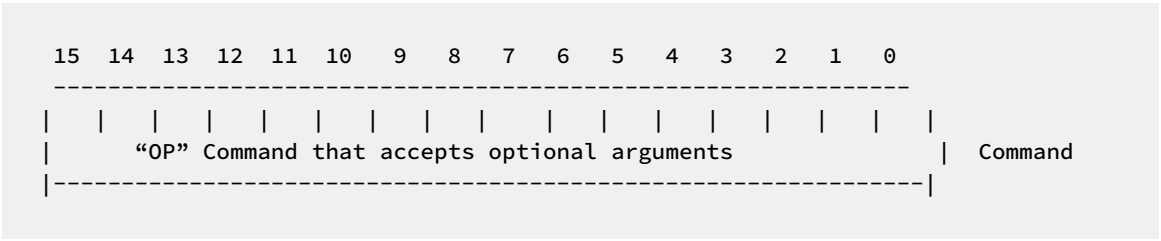
XARGS affects only the opcode that immediately follows it. The entire list of optional arguments is discarded after **OP** executes, even if **OP** did not use some or all of the list.

However, if **OP** is another **XARGS** command, then the additional arguments that it supplies are appended to the first set. In this way, **XARGS** can supply an arbitrarily large number of additional arguments.



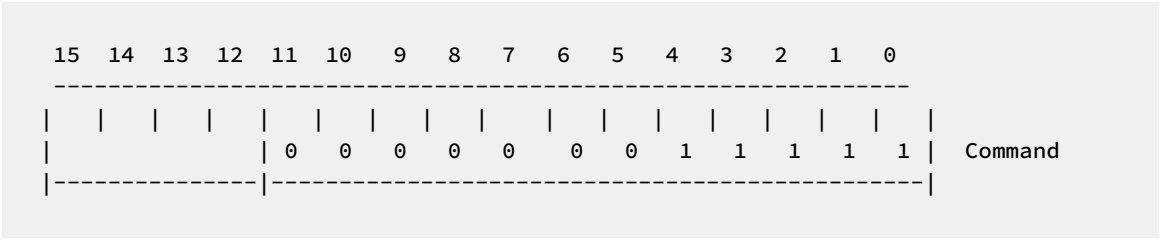


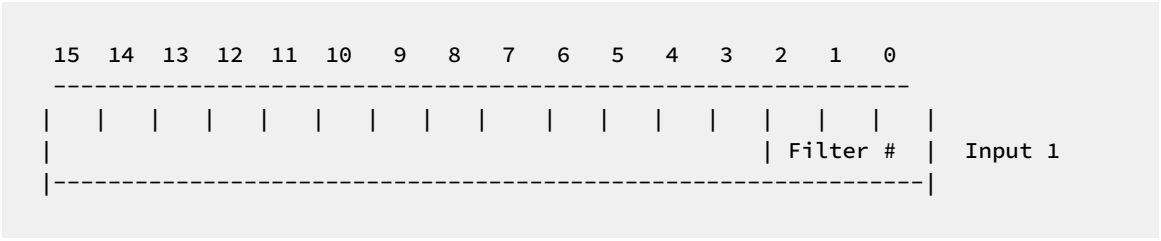


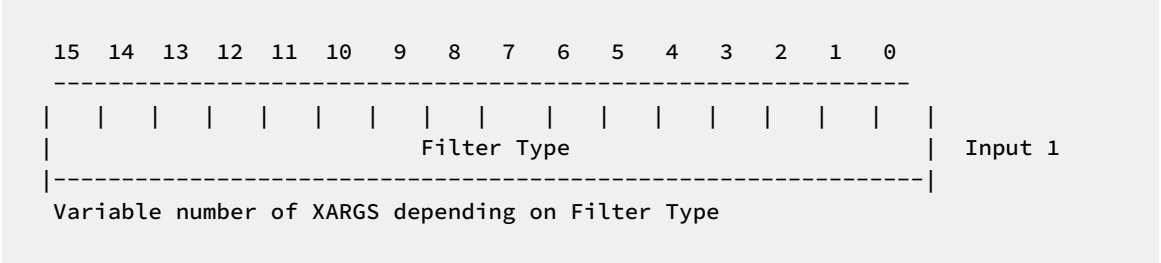


8.22 Load Clutter Filter Specifications (**LFSPECS**)

RVP900 allows 7 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 #` tells which filter definition slot is being modified, and the `Filter Type` tells the type of clutter filter to construct.

The command is 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 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 clutter gap width is 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

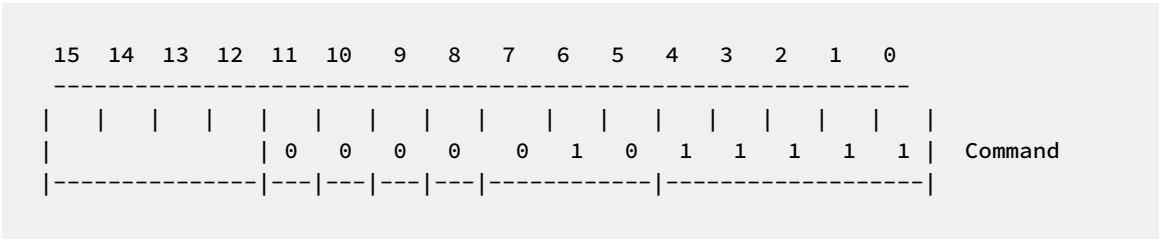
This is the RVP900 most advanced clutter filter, combining the best techniques for determining the clutter gap width and restoring whatever low-velocity spectral points are removed.

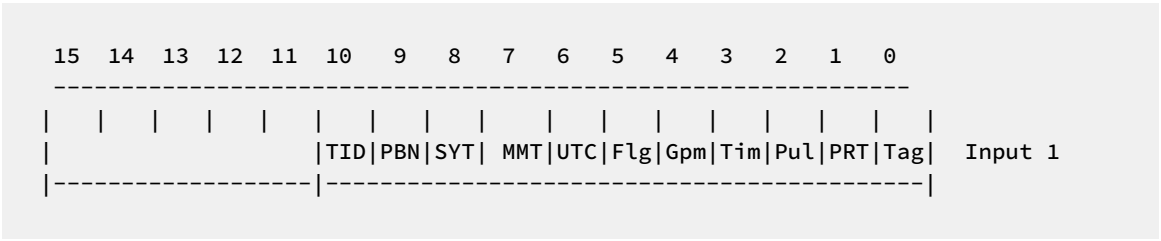
This filter is characterized by a single parameter, which is the assumed clutter width expressed as a physical velocity.

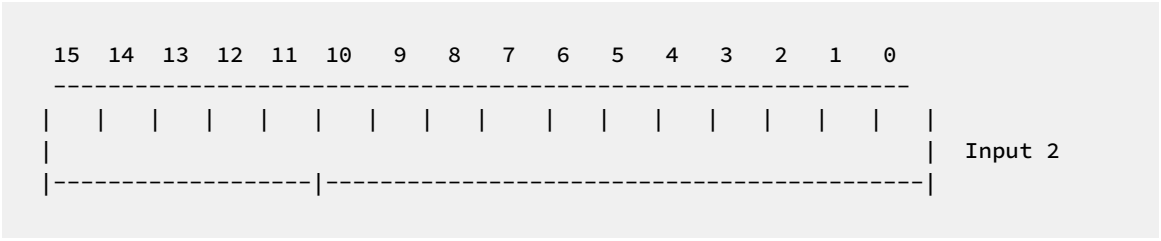
8.23 Configure Ray Header Words (**CFGHDR**)

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) 32 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 suppresses the header; though this can also be done without changing the configuration in 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 RVP900 operates 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.

  RVP900 outputs all zeroed data when 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 PRTs are forced to `0xFFFF` (the maximum unsigned value) when the external trigger is expected but missing.

`Pul`

Number of pulses 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`
: Mis-matched 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).

8.24 Configure Interference Filter (**`CFGINTF`**)

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 is applied, and to configure its operation with additional **`XARGS`** parameters. See 8.21 Pass Auxiliary Arguments to Opcodes (XARGS) (page 302).

If the **XARGS** are not supplied, the filter parameters retain their previous values. **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 **XARGS** is supplied, then that single threshold value is used for both C1 and C2.

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| Filter        | 0   0   0   0    0   1   1   1   1   1   1   1 |  Command
|---------------|-------------------------------------------------|
```

Filter chooses which interference algorithm should be run. See 7.2.5 Interference Filter (page 176).

- 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)

Vaisala recommends Alg.3 for general operational use. The other algorithms are included mostly for historical reasons.

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|      Threshold Parameter C1 in Hundredths of decibels         |  XARG 1
|---------------------------------------------------------------|
```

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|      Threshold Parameter C2 in Hundredths of decibels         |  XARG 2
|---------------------------------------------------------------|
```

8.25 Set AFC level (**SETAFC**)

SETAFC 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 … +100% AFC range that is defined in the **Mb** menu. See 5.2.1 Mb — Burst Pulse and AFC (page 95).

RVP900 automatically converts the new level to the configured analog or digital AFC output format. The exception is for the Motor/Integrator type of AFC loop, for which this command does nothing.

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|               | 0   0   0   0    1   0   1   1   1   1   1   1 |  Command
|---------------|-----------------------------------------------|
```

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|      16-Bit AFC/MFC Value (-32768 through +32767)              |  Input 1
|-------------------|-------------------------------------------|
```

8.26 Set Trigger Timing Slew (**SETSLEW**)

The **Mt** menu allows you to select a subset of triggers that can be slewed left and right to place the burst pulse accurately at range zero.

SETSLEW allows you to manually set the present amount of slew. The input argument is in hundredths of microseconds, that is, ranging from - 327.68 .. +327.67 µsec. The permitted span is ±20 µsec.

This is the same format used in `GPARM Output #56` which shows the present slew value.

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|               | 0   0   0   0    1   1   0   1   1   1   1   1 |  Command
|---------------|-----------------------------------------------|
```

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|        Signed trigger slew in hundredths of microseconds      |  Input 1
|---------------------------------------------------------------|
```

8.27 Hunt for Burst Pulse (**BPHUNT**)

The **BPHUNT** command allows the host computer to initiate hunt mode when it knows or can sense that a burst pulse should be present.

BPHUNT starts 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 procedure may take several seconds to complete. The RVP900 host computer interface remains functional during this time, but any acquired data is questionable.

GPARM status bits in `word #55` indicate when the hunt procedure is running, and whether it has completed successfully.

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
-----------------------------------------------------------------
|  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|          |Now| 0   0   0   0    1   1   1   1   1   1   1   1 |  Command
|--------------|---|---|---|---|------------|------------------|
```

Now forces the hunt procedure to start even if the burst pulse is already present.

Normally the procedure only starts when the burst pulse is missing at the time **BPHUNT** is given.

More Information

- Burst Pulse Tracking (page 175)

8.28 Configure Phase Modulation (**CFGPHZ**)

CFGPHZ configures RVP900 phase control output lines, which determine the relative phase of each transmitted pulse.

In some cases the chosen phase sequence has 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 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. See 5.2.7 Mz — Transmissions and Modulations (page 126).

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
-----------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|   |  PhSeq    | 0   0   0   1    0   0   0   1   1   1   1   1 |  Command
|---|-----------|-------------------------------------------------|
```

`PhSeq=0`
: Selects `No Modulation`. 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 is 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 **XARGS** words accompany this command.

`PhSeq=2`
: Selects a `User Defined` sequence. If no **XARGS** have been supplied, 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.
RVP900 makes 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[1)] that separates and recovers first and second trip echoes in `Random Phase` mode. It usually performs better than a truly random transmit sequence, especially when the processing interval is fairly short (as little as 32-pulses). With no **XARGS**, 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. You may also supply your own 32-pulse angle sequence.

8.29 Set User IQ Bits (**UIQBITS**)

UIQBITS loads user-specified bits that are 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 in all pulse headers. The `ONCE` bits are transitory and 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 are 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 always result in at least one pulse header being tagged with exactly that data. This is generally what you want, since no other exact outcome could be guaranteed based on time-of-arrival alone.

1) *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.*

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|               | 0   0   0   1   0   0   1   1   1   1   1   1 |  Command
|---------------|-----------------------------------------------|
```

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|                 64 Permanent User Bits to SET                 |  Inputs 1-4
|---------------------------------------------------------------|
```

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|                 64 Permanent User Bits to CLEAR               |  Inputs 5-8
|---------------------------------------------------------------|
```

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|                 64 Permanent User Bits to ONCE                |  Inputs 9-
12
|---------------------------------------------------------------|
```

8.30 Set Individual Thresholds (**THRESH**)

The **SOPRM** command allows you to configure 4 threshold numbers used by all data types, and to select the threshold control flags for 5 of the data types. See 8.4 Setup Operating Parameters (SOPRM) (page 235).

Use the **THRESH** command if you wish to apply different threshold numbers to different data types. You can individually set the thresholds and mask used for each data type, or for groups of data types.

The **GPARM** command reads out the threshold numbers set for velocity. To read back the numbers for each data type use the **RBACK** command. See 8.20 Read Back Internal Tables and Parameters (RBACK) (page 300).

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|               | 0   0   0   1   0   1   1   1   1   1   1   1 |  Command
|---------------|-----------------------------------------------|
```

The first 3 words supply a mask that indicates which data types are being set:

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|   | Z | T | V | W |ZDR|       |KDP |                           |  Input 1
|---|---|---|---|---|---|-------|----|---------------------------|
```

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|                       |Flg|Phi Rho  Ldr|Phi Rho Ldr|SQI|RHV|PDP|  Input 2
|-----------------------|---|---|----|---|---|---|---|---|---|---|
```

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|                        <all spares>                           |  Input 3
|---------------|-----------------------------------------------|
```

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|          LOG Threshold in 1/16 of dB(SOPRM Input 4)           |  Input 4
|---|---|---|---|---|---|-------|----|---------------------------|
```

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|          CCOR Threshold in 1/16 of dB(SOPRM Input 5)           |  Input 5
|---|---|---|---|---|---|-------|----|---------------------------|
```

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|         SQI   Threshold in 1/16 of dB(SOPRM Input 6)          |  Input 6
|---|---|---|---|---|---|-------|----|--------------------------|
```

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|   Weather Signal Power Threshold in 1/16 of dB(SOPRM Input 7)  |  Input 7
|---|---|---|---|---|---|-------|----|--------------------------|
```

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|       Threshold Control Flags(SOPRM Input 11,12,13,14,19)      |  Input 8
|---|---|---|---|---|---|-------|----|--------------------------|
```

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|                      <spare>                                  |  Input 9
|---------------------------------------------------------------|
```

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|                      <spare>                                  |  Input 10
|---------------------------------------------------------------|
```

8.31 Set Task Identification Information (**TASKID**)

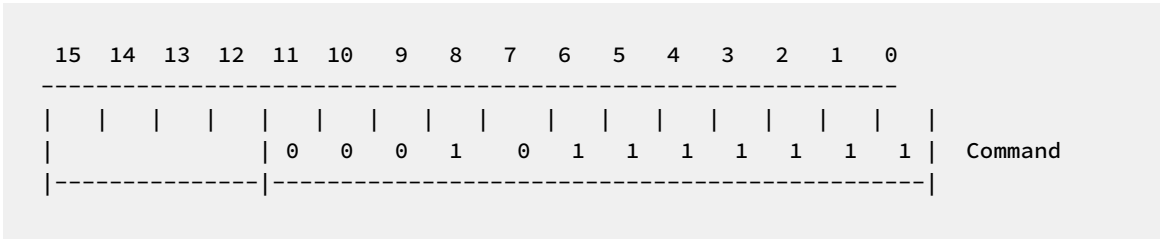
TASKID allows you to name the (**I**,**Q**) data that are currently being acquired by RVP900.

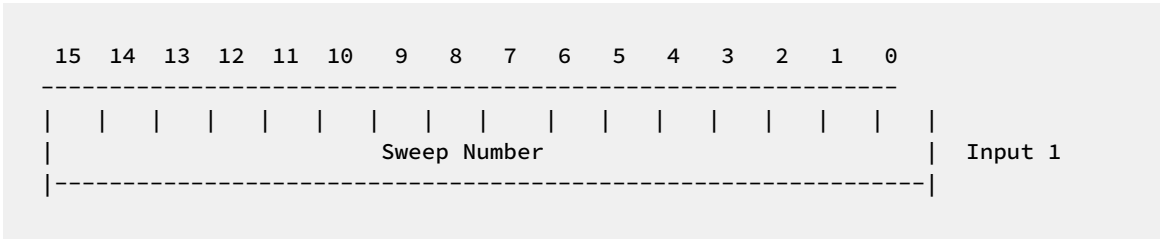
This naming information then becomes associated with these data, and is available in the pulse information structures (**struct rvp900PulseInfo**) that are read from the **Timeseries** API. The **iAqMode** field of the pulse headers (**struct rvp900PulseHdr**) are incremented each time a **TASKID** opcode is received, but the continuous flow of (**I**,**Q**) data from the RVP900/Rx card(s) are disturbed.

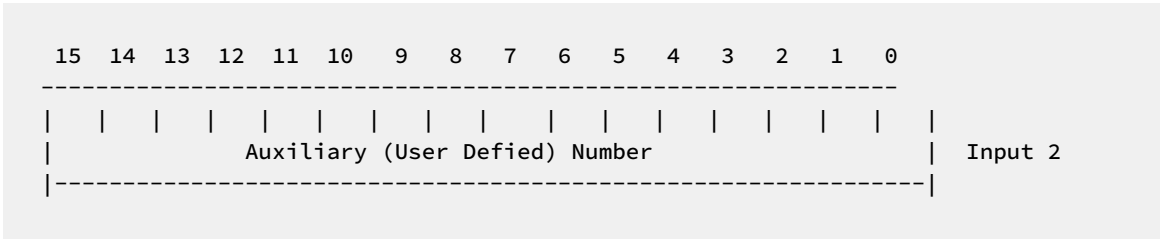
The **TASKID** command defines a 16-character Null-terminated name, along with a 16-bit sweep number and 16-bit auxiliary (user defined) number.

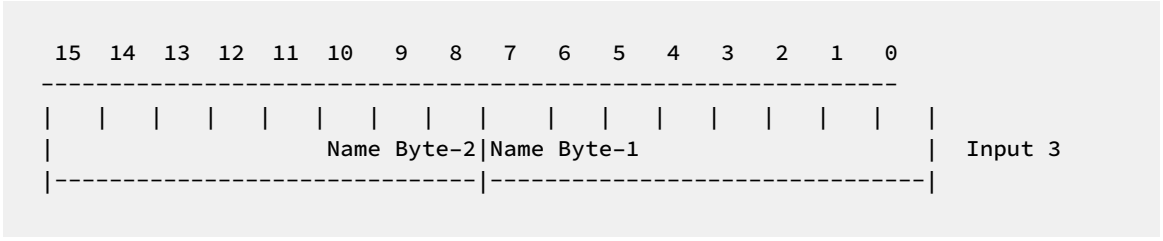
You may use all 16 characters of the name, as it is stored internally in 17 slots.

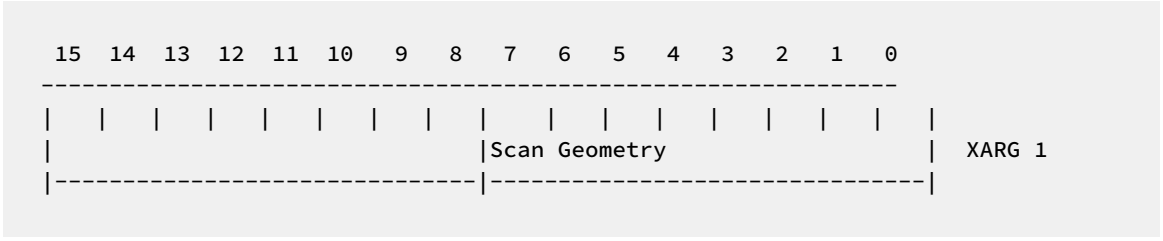
The `Sweep Number` and `Scan Geometry` (one of `SCAN_xxx parameters`) should be filled in with values best approximating those notions. `Auxiliary Number` may be filled in with any value that you find meaningful.











8.32 Define PRF Pie Slices (**PRFSECT**)

This command supplements the **SETPWF** command and allows an alternate trigger PRF to be generated within prescribed AZ/EL sectors. See 8.15 Set Pulse Width and PRF (SETPWF) (page 292).

Up to 8 trigger sectors can be defined by invoking **PRFSECT** for each separate region. The trigger pattern then automatically changes when the antenna enters any of these regions, but the timeseries data remains continuous and uninterrupted throughout each change. **PRFSECT** allows a complete volume scan to run with PRFs that have been optimized to the radar echoes in all directions. Note the following caveats should be observed:

- Dual-PRF unfolding cannot work properly at PRF sector boundaries. 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 prevents the PRF seam from bobbling between two adjacent sync angles.

The **SETPWF** opcode deletes any alternate sectors that have been setup so far. Thus, you can only use the **PRFSECT** command after **SETPWF** has established the pulse width and default trigger rate for the entire AZ/EL scan volume.

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|               | 0   0   0   1    1   0   0   1   1   1   1   1 |  Command
|---------------|-----------------------------------------------|
```

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|                                                 |Sector     |  Input 1
|-------------------------------------------------|-----------|
```

`Sector` selects which sector number is defined to have an alternate trigger pattern. This is an arbitrary index from 0 … 7.

The following 4 arguments define a solid sector in azimuth and elevation within which the alternate trigger pattern is used in preference to the default (**SETPWF**) pattern. When the current `AZ/EL` angle pair is contained in more than one defined sector, the trigger pattern from the lower numbered sector is used.

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.

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|               Lower AZ (Binary angle)                         |  Input 2
|---------------------------------------------------------------|
```

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|               Upper AZ (Binary angle)                         |  Input 3
|---------------------------------------------------------------|
```

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|              Lower EL (Binary angle)                          |  Input 4
|---------------------------------------------------------------|
```

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|              Upper EL (Binary angle)                          |  Input 5
|---------------------------------------------------------------|
```

The following arguments specify a trigger period in the same manner as the optional form of the **SETPWF** command.

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|     Lower 16-Bits of 32-Bit Trigger Period in Nanoseconds     |  Input 6
|---------------------------------------------------------------|
```

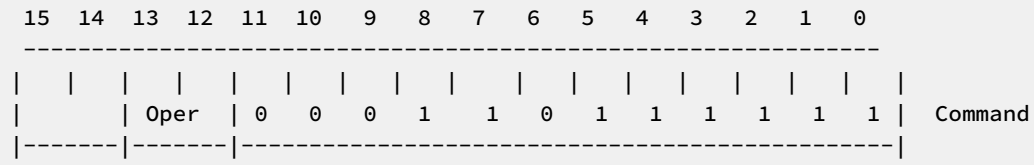
```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
 ---------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|     Upper 16-Bits of 32-Bit Trigger Period in Nanoseconds     |  Input 7
|---------------------------------------------------------------|
```

8.33 Configure Target Simulator (**TARGSIM**)

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 8.11 Load Simulated Time Series Data (LSIMUL) (page 285).

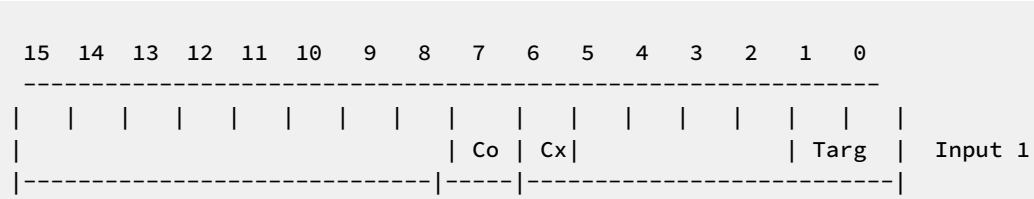
In the simulated case, you can overlay simulated physical targets on top of real physical targets from the radar receiver.



`Oper=0`
: Disable all target simulation activity (no additional arguments)

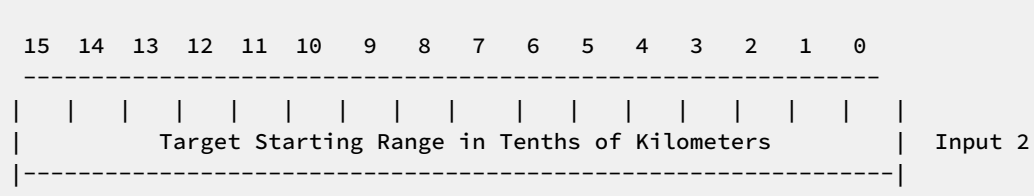
`Oper=1`
: Enable simulation of all defined targets (no additional arguments)

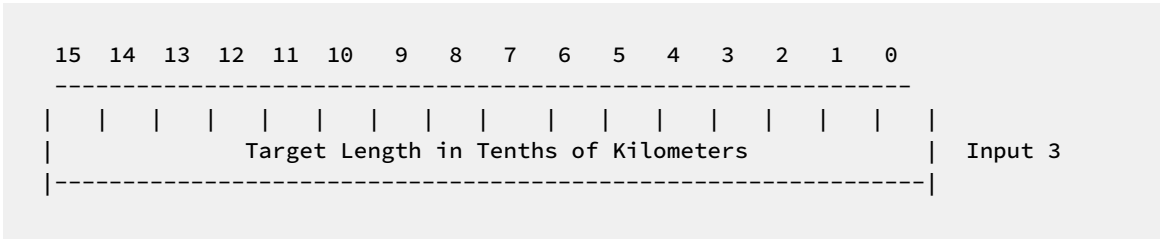
`Oper=2`
: Define a new simulated target (arguments list follows)

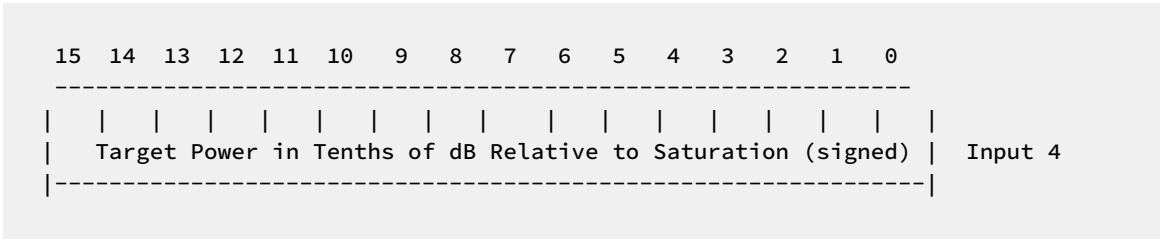


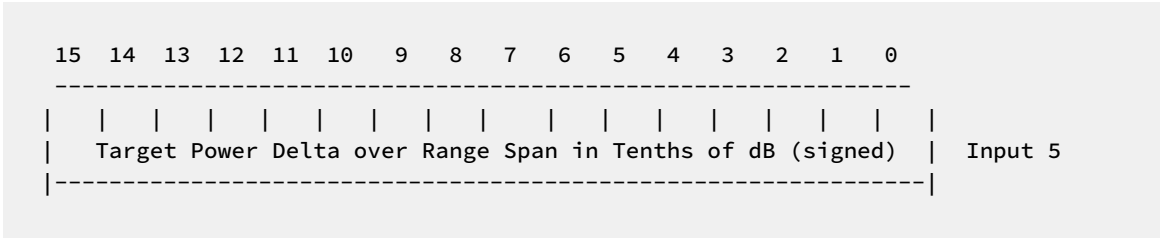
`Targ`
: Which target is being defined

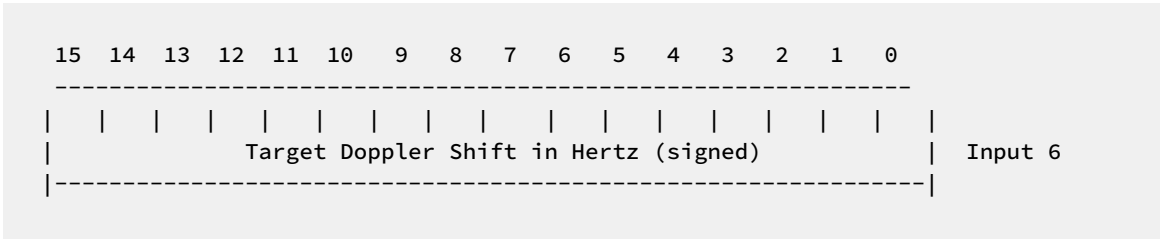
`Co/Cx`
: Place simulated target in Co-Pol and/or Cross-Pol Rx channels

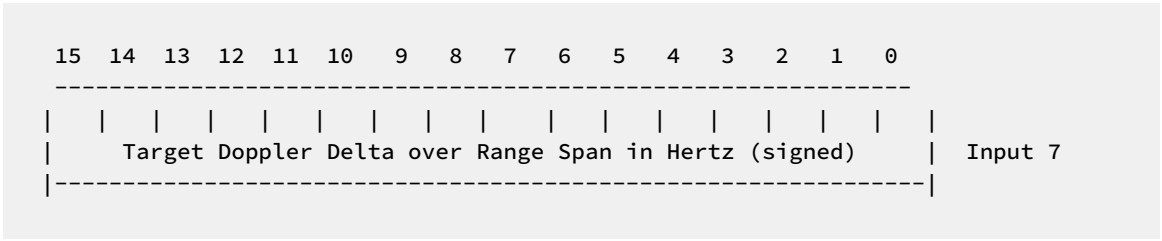


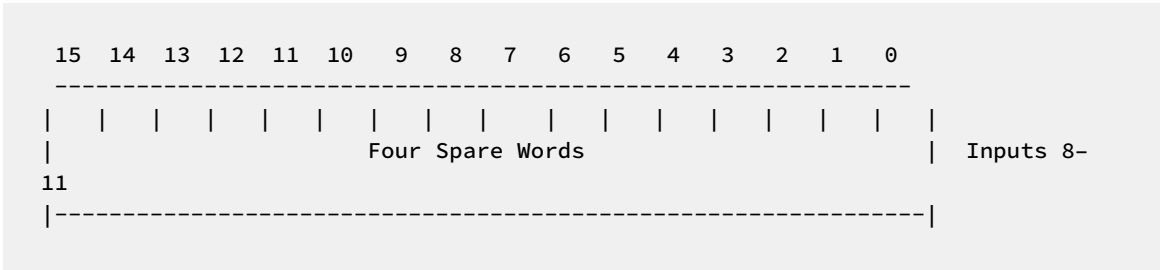

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
Target Length in Tenths of Kilometers
Input 3


15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
Target Power in Tenths of dB Relative to Saturation (signed)
Input 4


15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
Target Power Delta over Range Span in Tenths of dB (signed)
Input 5

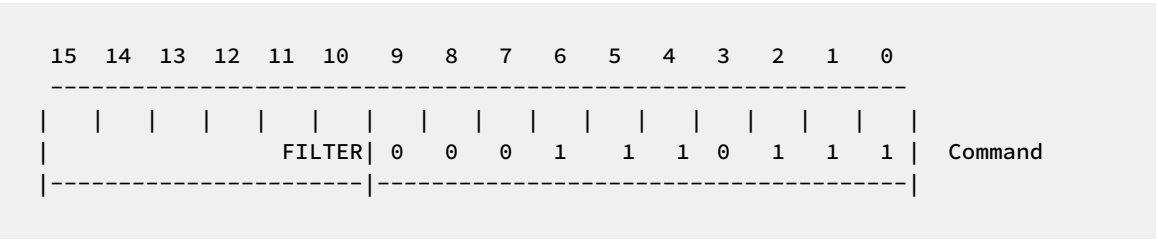

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
Target Doppler Shift in Hertz (signed)
Input 6

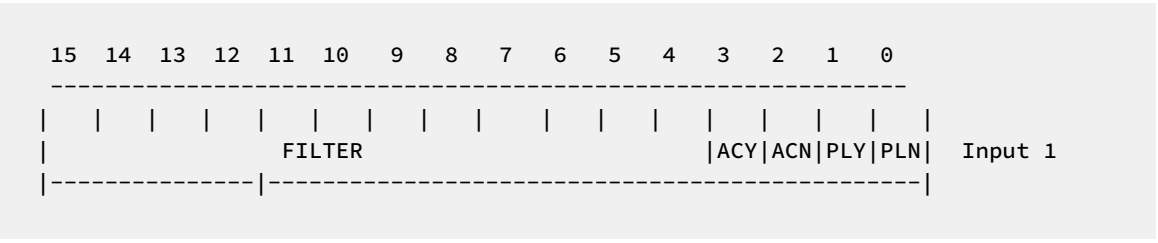

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
Target Doppler Delta over Range Span in Hertz (signed)
Input 7


15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
Four Spare Words
Inputs 8-
11

8.34 Set Burst Pulse Processing Options (**BPOPTS**)

Use this command to set burst pulse processing options.





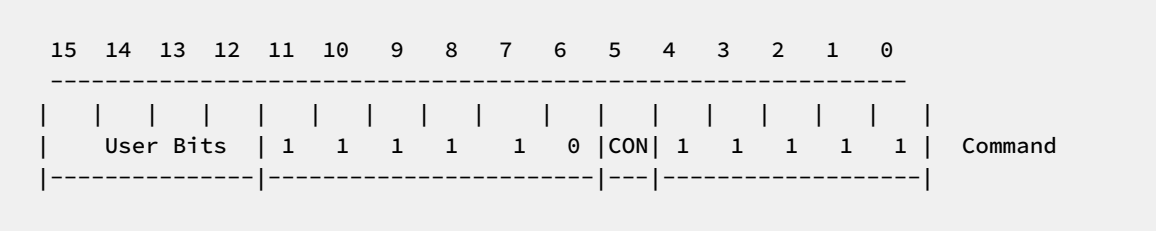
`PLY/N`
: Affects whether RVP900 phase locks its (**I**,**Q**) data to the measured burst pulse. The **PLY** and **PLN** bits force **Yes** and **No** responses. If both bits are clear or both bits are set, no change is made.

`ACY/N`
: Affects whether RVP900 applies pulse-to-pulse amplitude correction to its (**I**,**Q**) data. The **ACY** and **ACN** bits force **Yes** and **No** responses. If both bits are clear or both bits are set, no change is made.

8.35 Custom User Opcode (**USRINTR** and **USRCONT**)

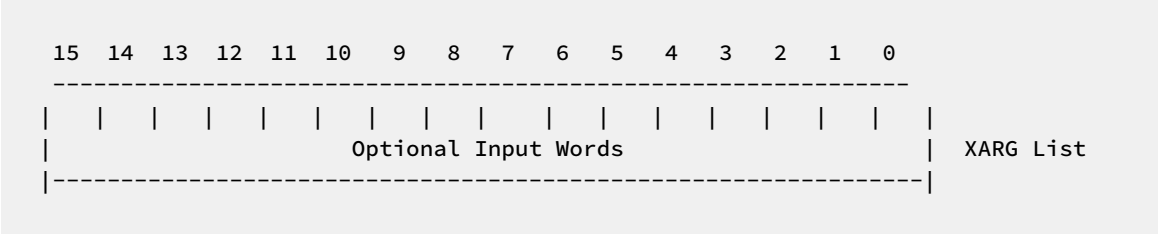
These opcodes are part of the open software extensions that allow you to define custom opcodes for each major mode of operation.

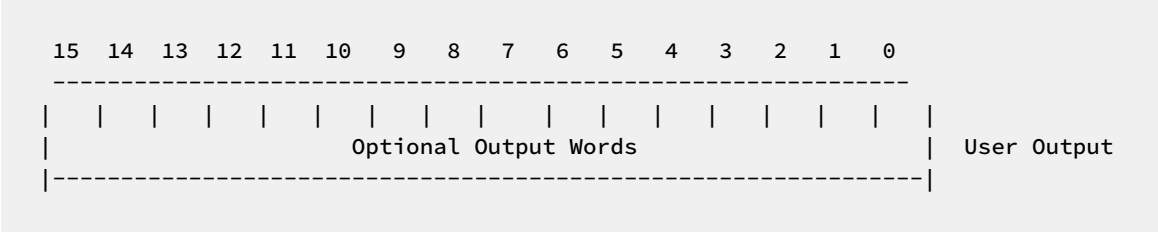
Arguments may be passed into a custom opcode handler as an **XARG** list. An optional array of words returned from that handler appears after the command executes.



`UserBits`
: Four additional bits defined by the user to help subdivide the opcode functions.

CON
If set, the **I**,**Q** data acquisition thread proceeds continuously while the opcode is executed.
If clear, the **I**,**Q** stream is interrupted before handling the call.





8.36 Load Melting Layer Specification (**MLSPEC**)

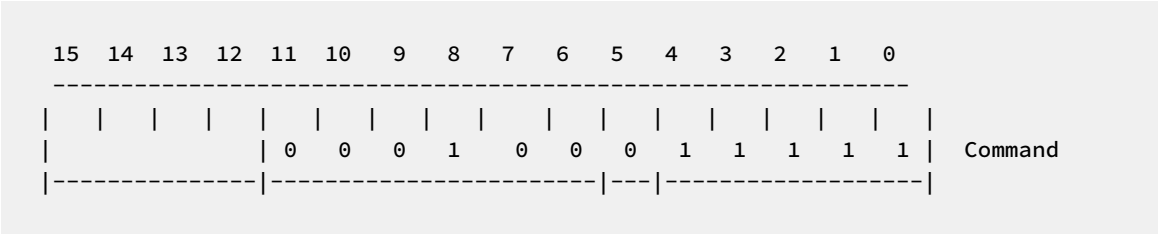
RVP900 allows you to get melting layer-specific information, including the melting layer height and the uncertainties, from the latest melting layer product created in IRIS.

The melting layer specifications 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.

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 (MSL), 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 the **PROC** command.

The command is an extended **OpCommand** and is defined as follows:



The lowest 5 bits correspond to the default **0X1F** for extended **opcode** commands and the 7 middle bits are the value for the **XOP_MLSPEC** command (**0X10**).

The application software provides input data by, for example, converting the IRIS standard cartesian MLHGT products to the curved Earth coordinates of the sweeps configured for the radar task. Typically, a map for a collection of sweeps (a volume) is uploaded.

The application software 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*DPO LAPP_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.

RVP900 supports a command that provides a way to feedback the melting layer information from IRIS to the RDA. The melting layer altitude information is specified for each antenna angle and range. This way, the melting layer height can be used in different live data processing applications such as `HydroClass`.

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 processed ray processed, 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  (Unsigned 16-bit integer)         |  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 for 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`
: 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 $\#rays = \frac{360000}{ires}$

where `iRES` is the desired angular resolution expressed as an integer number of thousands of degrees.

`Number of bins`
: The number of output bins.

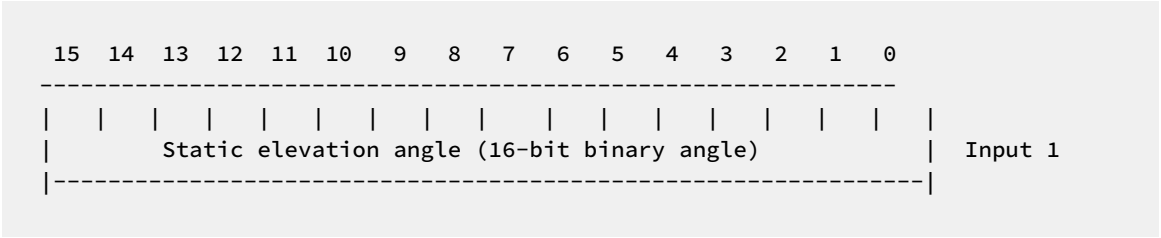
`Angular resolution`
: The binary angle of the corresponding angle in degrees, $\theta = \frac{360}{nMLrays}$

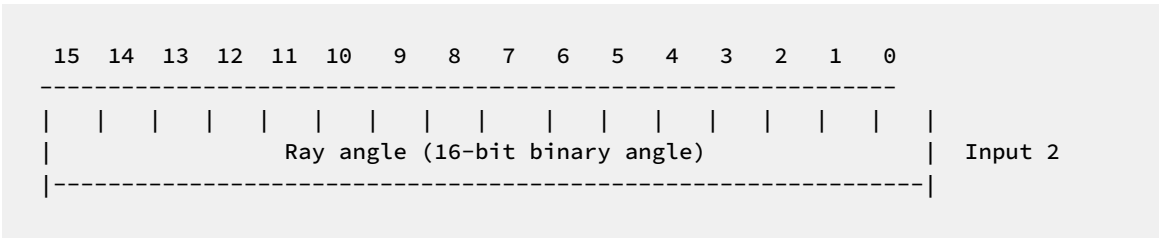
`Bin starting`
: Range of the first bin in centimeters.

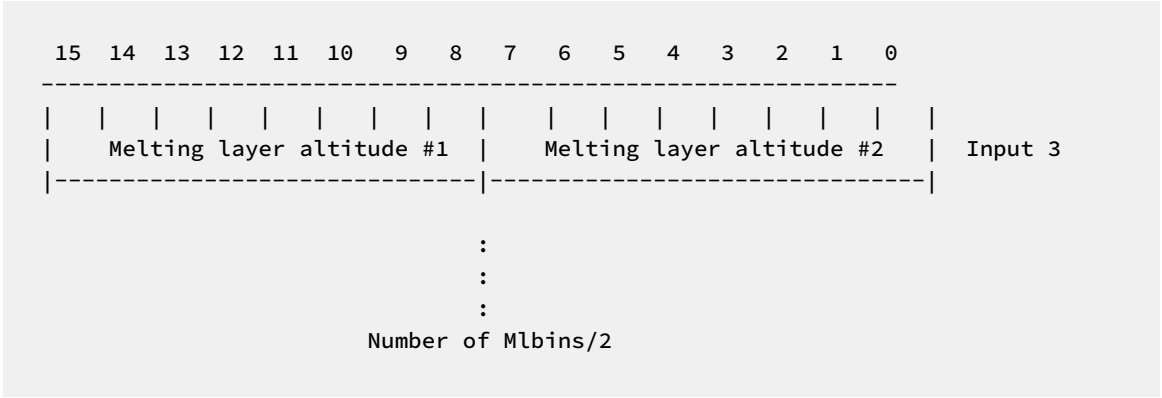
`Bin Spacing`
: The product of the number of output bins by the step divided by the number of bins.

`Melting layer thickness`
: The user setting of the melting layer thickness in the product configuration.

`Status`
: Indicates the status of the DSP features.






15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
Melting layer altitude #1
Melting layer altitude #2
Input 3
Number of Mlbins/2

9. Technical Data

9.1 RVP900 Processing Algorithms

Table 67 Processing Algorithms

Algorithm	Description
I/Q signal correction options	• Amplitude jitter correction based on running average of transmit power from burst pulse • Interference correction for single pulseinterference • Saturation correction (3 ... 5dB)
Primary processing modes	• Poly-Pulse Pair (PPP) • DFT • Random or phase coded second trip echofiltering/recovery • Optional polarization with full co-variance matrix (**Zdr**, **PHIDP**, **LDR**, **RHOHV**, and so on.) • Optional pulse compression
Processing options	• Azimuth averaging: 2 ... 1024 pulses • Corrections for gaseous attenuation and1/R2 • Dual polarization: Alternating, simultaneous, H-only, V-only • Fixed, adaptive width, and GMAP clutter filters in DFT and phase coded second tripmode • High sensitivity Rhv STAR mode processing: >3dB improvement in detectability • Pulse integration up to1024 • Range de-aliasing: phase coding method (random phase for magnetron) or frequency coding method (not available for magnetron) • Scan angle synchronization for data acquisition • Up to 4 pulse widths • 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
Data outputs (8 and 16 bit)	**Zh**, **Zv**, **Zhv**, **V**, **W**, **SQI**, **ZDR**, **LDR**, **RHOHV**, **PHIDP**, and **KDP**
Optional data outputs	**HCLASS**, **I/Q**
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 havingvery strong clutter; typically applied to mean velocity, width, and dBZ • Speckle Filter—Removes single-bin targets such as aircraft or noise and fills isolated missing pixels

9.2 RVP900 Input and Output Summary

Table 68 I/O Summary

Function	Description
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
Ethernet I/O 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 custom application using a public API. Signal processor configuration and verification read-back is performed through the Ethernet interface.
Input from RVP901 IFDR	• 16-bit **I**/**Q** samples • Optional dual-channel **I**/**Q** samples (for example, for polarization systems or dual frequencysystems)
Optional Polarization Control	RS-422 differential control for polarization switch
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).

9.3 RVP901 IFDR Specifications

Table 69 IF Band Pass Filter

Function	Description
IF band pass filter	Programmable Digital FIR with software selectable bandwidth. Built-in, filter design software with user interface.

Table 70 IF Inputs

Characteristic	Description
Input Signals	• IF Received Signal: 50 Ω ,+ 8.0 dBm full-scale, +20 dBm absolute max • IF Burst or COHO: 50 Ω, +8.0 dBm full-scale, +20 dBm absolute max • Optional Reference Clock: 7.5 MHz ... 100 MHz, -20 dBm ... 6 dBm
IF range	5 ... 120 MHz
Saturation level	+8.0 dBm @ 50 Ω
Dynamic Range (dependent on matched filter)	90 ... >105 dB

Characteristic	Description
Optional single and dual polarization wide dynamic range	>120 dB
A/D conversion	• Resolution: 16 bit with jitter <1.0 picosec • Sampling rate: 50 ... 100 MHz (software selectable)
Sampling rate	50 ... 100 MHz
Master clock jitter	<1.0 picosec
Multiply/accumulate cycles per second	38.4 billion Hz
Pulse repetition frequency (PRF)	50 Hz ... 20 KHz, , +0.1%, continuously selectable
Impulse response	Up to 80 µsec These very long filters are used 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.
Minimum range resolution	15 meters (accuracy of ±1.5 m)
Maximum range	Up to 1024 km
Maximum number of range bins	Full unambiguous range at minimum resolution or 4200 range bins (whichever is less)

Table 71 Phase Stability

Characteristic	Description
Klystron:	Better than 0.1°
Magnetron (for 1.0 microsecond pulse):	Better than 0.5°

Table 72 IF Waveform Generator

Characteristic	Description
Two 16-bit **TxDAC** outputs	5 ... 65 MHz >65 dB SNR +13dBm @ 50Ω
TxDDS output	5 ... 105 MHz >65 dB SNR +13dBm @ 50Ω

Table 73 RVP901 IFDR I/O Specifications

Characteristic	Description
AFC output	• Digital AFC (DAFC) with up to 24 programmable output bits • Automatic 2D (time/frequency) burst pulse search and fine-tracking algorithms
Data output through Ethernet	• 16-bit floating `I` and `Q` values • 16-bit raw IF samples
RS-422	20 differential line pairs
TTL/CMOS Lines	20 series terminated
Analog input	6 differential pairs ±10V Signal must be low frequency with steps settling to 0.1% in 800 nanoseconds and a maximum sampling rate of 0.5 µsec.
RVP901 to RVP902 link	The IFDR is connected to RVP902 Signal Processor by a CAT5e cable (up to 25 m in length), jumbo 8192-byte packets. Each digital I/O line is configurable with over-voltage, ESD, and EFT protection.

Table 74 RVP901 IFDR Physical Specifications

Characteristic	Description
Dimensions	26.8 cm x 17.6 cm x 4.8 cm (10.5 in x 6.9 in x 1.9 in)
Required ambient air temperature (DC power supply)	-40 ... +50 °C (-40 ... 122 °F) Non-condensing humidity 9 ... 95 %
Required ambient air temperature (AC power supply)	-40 ... +45 °C (-40 ... 113 °F) Non-condensing humidity 9 ... 95 %
Input power	Available for 100 ... 240 VAC at 47 ... 63 Hz or 18 ... 6 VDC. • AC power supply: is a low noise, low ripple, auto-switching unit. • DC input voltage may be unregulated. DC voltage available fan assembly options: 12 ... 36 VDC, and 18 ... 28 VDC. The required DC input requires an SELV Compliant source in the in the 12 ... 36 V input range. The recommended power rating DC input is 100 W with max input nosie of 1.0 Vp-p @ 50 ... 100 Hz and 100 mVp-p 120 ... 200kHz.
Internal regulator	An isolated supply with a 12 ... 36V wide ranging input. It has an 10 A fuse with reverse and over voltage protection and in-rush current limiting. Sized to provide several Watts more than is required by RVP901.
Installation	IFDR has a gasketed ruggedized enclosure to minimize electrical interference and mechanical stress. The IFDR is usually mounted inside the receiver cabinet, but the multi-functionality provides flexibility for placing the device. If ordered without fans must provide a minimum of 20 cubic feet per minue of air flow across RVP901 enclosure fins.

More Information

- RVP901 IF Digital Receiver (page 23)

9.4 RVP901 Digital Waveform Synthesis

Table 75 Digital Waveform Synthesis

Function	Description
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	• 2 independent, digitally synthesized, analog output waveforms (SMA). These outputs are electrically identical and logically independent IF waveform synthesizers that can produce phase modulated CW signals, finite duration pulses, compressed pulses, and so on. • Can drive up to +13 dBm into 50Ω • 16-bit interpolating TxDAC provides >65 dB Signal-to-Noise Ratio • IF center frequency selectable 5 MHz ... 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 5 MHz ... 105 MHz

9.5 RVP902 Signal Processing Computer Specifications

Table 76 RVP02 Signal Processing Computer Specifications

Characteristic	Description
Chassis	• 3U Short-depth rackmount chassis (depth 18.9"), Advantech HPC-7320 Chassis • Rack rail kit to accommodate 19" racks of depths 18" ... 36"
Cooling	• Fan: 2 (8cm/57CFM) + 1 (6cm/27.72CFM) • Air filter

Characteristic	Description

I/O	• PCI slot to allow for IO-62 card installation
	• DVD +/-RW drive for software installation
	• 2 Serial Ports accessed from the back
	• 4 10/100/1G Ethernet ports accessed from the back
	• 3.0 USB ports:
	  • 4 ports accessed from the back
	  • 2 ports accessed in the front
	• 2.0 USB ports:
	  • 2 ports accessed from the back
	• PS/2 Single Connector with brake out cable for monitor and keyboard

Memory and processing	• 512 GB Solid State Drive configured with RAID 1 (mirrored) • 16 G of DDR4 Memory • 2 XEON E5-2609 v3 Processors with 6 Cores, 1.9G Hz clock speed, 6.4GT/s QPI Speed, 15M of Cache
Operating environment	• Temperature: 0 ... 40° C (32 ... 122° F) • Humidity: 10 ... 95% @ 40° C, non-condensing • Vibration (5...500Hz): 1 Grms • Shock: 10 G (with 11 ms duration, half sine wave)
Non-operating environment	• Temperature: -40 ... 70° C (-40 ... 156° F) • Humidity: 10 ... 95% @ 60° C, non-condensing • Vibration (5...500Hz): 2 G
Physical characteristics	• Dimensions (W * H * D): 426.4 x 132.2 x 480 mm (16.79″ x 5.2″ x 18.9″) • Weight: 12 kg
Power	Dual redundant power supplies auto-ranging 100 ... 240V AC

More Information

- RVP902 Signal Processing Computer (page 38)
- Installing RVP902 Main Chassis (page 75)

9.6 RVP902 Safety Compliance

Table 77 RVP902 Safety Compliance

Item	Standard
Radiated and Conducted Emissions.	FCC Part 15 Class A ICES-003, AS/NZS & CISPR32 EN55011 Class A Group 1
Harmonics and Flicker	EN61000-3-2 & 3-3
Electromagnetic compatibility	EN61000-6-2 - 2005
ESD	EN61000-4-2
Radiated Immunity	EN61000-4-3

Item	Standard
EFT / Burst	EN61000-4-4
Surge	EN61000-4-5
Conducted Susceptibility	EN61000-4-6
Dips / Dropouts	EN61000-4-11
Product Safety	CE Mark - EN60950-1

9.7 RVP900 Spare Parts

Table 78 RVP900 Spare Parts

Part Number	Description
223326SP	AC Supply Assembly Spare part RVP900
DRW229161SP	DC FAN
DRW228541SP	DC Power Cable with fans
DRW229805SP	DC Power Cable without fans
RCP8-CPSP	IO62 Panel RCP8
RVP8-IOSP	IO62 Card
DRW229162	RVP9 Cable Interface Panel
RVP902-IO	Radar server computer and RVP8-IO

Table 79 RVP902-IO Spare Parts

Part Number	Description
248382SP	Redundant power
248384SP	1TB SSD
248386SP	512GB SSD
248392SP	System fan (2 pcs)

9.8 Physical and Environmental Characteristics

Table 80 Physical and Environmental Characteristics

Characteristic	Description

Packaging	• 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)
	See also Table 76 (page 329).

Input power	RVP901 IFDR: • 100 VAC ... 240 VAC • 47 Hz ... 63 Hz auto-ranging or 18 VDC ... 36 VDC RVP902 Main Chassis: • 100 VAC ... 240 VAC • 50 Hz ... 60 Hz auto-ranging
Power consumption	• RVP902/Main Processor: 1300 W • RVP901 IFDR maximum power consumption: 50 W • RVP901 IFDR IF Digitizer: 50 W
Environmental	• RVP901 with DC Option: -40°C ... 50°C (-40°F ... 122°F), 0% ... 95% R.H. (non-condensing) with a minimum of 20 cubic feet per minute of air-flow. • RVP901 with AC Option: -40°C ... 45°C (-40°F ... 122°F), 0% ...95% R.H. (non-condensing) with a minimum of 20 cubic feet per minute of air-flow. • RVP902: 10°C ... 35°C (50°F ... 95°F), 8% ... 90% R.H. (non-condensing) at altitudes less than 2000 m for Rev C and D, and at altitudes less than 5000 m for Rev E.
Reliability Digital receiver	MTBF>50,000 hours (at 25°C/77°F) <1 hour MTTR

9.9 Type Plate

The type plates identify RVP902 and RVP902-IO parts.

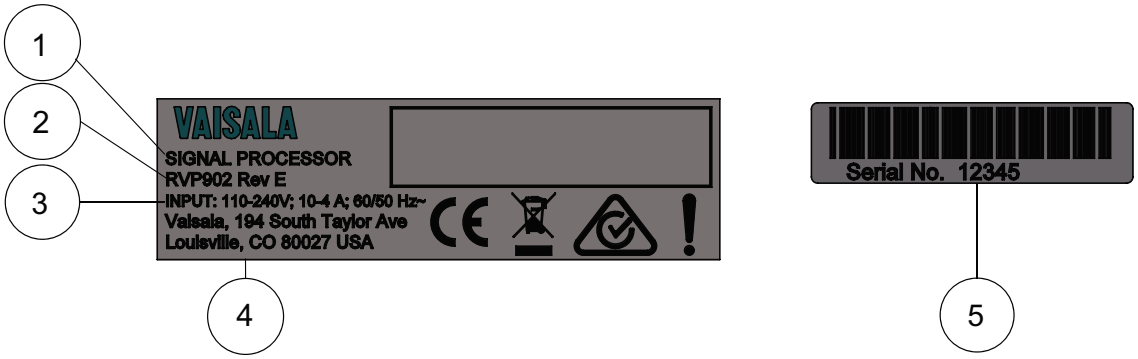


Figure 52 RVP902 Type Plate

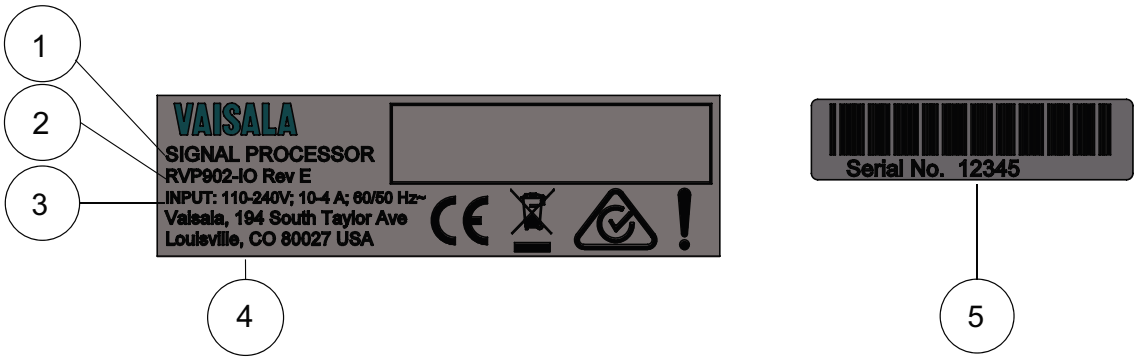


Figure 53 RVP902-IO Type Plate

1 Part name
2 Product code
3 Voltage, current, and frequency
4 Place of manufacture
5 Serial number in bar code

Appendix A. Installation and Test Procedures

A.1 Installation and Test Procedure Overview

Record all values when instructed. Save the values for future reference.

Customer	
Main Serial Number	
IFDR Serial Number	
Delivery Date	
Radar Mfg./Type	
Customer Engineer	
Vaisala Engineer	

These installation and test procedures are for Vaisala field engineers and customers installing and testing of RVP900 on a radar system.

Complete these procedures in the order described. Failure to perform one step may effect later tests.

Keep a copy of the test results either on file or with this manual. Do not write in this manual since it is replaced with an upgrade. Make a copy.

Sign-off on each test when it is complete. If a test does not pass, remedy the problem and repeat the test. If the test still does not pass, add a sheet to the test explaining the variance. A supplementary test sheet is at the end of the test procedure.

When 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. Perform the additional configuration and calibration procedures before using the RVP900 with the software.

Contact Vaisala technical support if you have problems or comments regarding this product or procedures.

A.2 Test Checklist

Complete the installation checklist after installing the RVP.

Record all values when instructed. Save the values for future reference.

Table 81 RVP Test Checklist

Task	Checked OK/Not OK	Remarks
A.3 Installation Check (page 337)		
A.4 Power Up Check (page 338)		
A.5.1 Checking Terminal Setup (page 340)		
A.6 Checking Internal Status with Setup V Command (page 340)		
A.7 Checking Board Configuration with Setup Mc Command (page 341)		
A.8 Checking Processing Options with Setup Mp Command (page 342)		
A.9 Checking Clutter Filters with Setup Mf Command (page 343)		
A.10 Checking General Trigger Setup with Setup Mt Command (page 343)		
A.11 Initial Setup of Information for Each Pulse Width (page 345)		
A.12 Checking Burst Pulse and AFC with Setup Mb Command (page 346)		
A.13 Checking Debug Options with the M+ Command (page 347)		
A.14 Checking Transmitter Phase Control with Setup Mz Command (page 348)		
A.15 Ascope Test (page 349)		
A.16 Burst Pulse Alignment (page 350)		
A.17 Bandwidth Filter Adjustment (page 351)		
A.18 Digital AFC (DAFC) Alignment (Optional) (page 352)		
A.19 MFC Functional Test and Tuning (Optional) (page 354)		
A.20 AFC Functional Test (Optional) (page 355)		
A.21 Checking Input IF Signal Level (page 356)		
A.22 Checking Calibration and Dynamic Range (page 358)		
A.23 Checking Receiver Bandwidth (page 359)		
A.24 Checking Receiver Phase Noise (page 361)		
A.25 Hardcopy and Backup of Final Setups (page 362)		
A.26 RVP901 TxDAC Stand-alone Bench Test (page 363)		

Task	Checked OK/Not OK	Remarks
Test Passed		
For Customer	Date:	
For Vaisala	Date:	

A.3 Installation Check

Test Goal

Verify that the RVP900 is properly connected to the radar system and document some basic radar characteristics. There are differences for TWT/Klystron and magnetron radar systems.

Checklist

Record all values when instructed. Save the values for future reference.

Table 82 RVP Installation Checklist

Task	Checked OK/Not OK	Remarks
RVP901 Digital Receiver is properly connected to AC/DC line voltage according to supplied power converter.		
RVP901 is 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: _____		
CAT5E Ethernet cable is connected to J2 **`Ethernet`** of RVP901 and the other end is connected to **`ETH1`** port of the RVP902 Server chassis.		
Trigger connections verified		
Magnetron Systems		
IF burst pulse is connected to J13 **`ADC-E`** of RVP901.		

Task	Checked OK/Not OK	Remarks
Klystron Systems		
IF COHO is connected to J13 **ADC–E** of RVP901.		
Tx license key is installed (circle one): • Yes • No		
Test Passed		
For Customer	Date:	
For Vaisala	Date:	

A.4 Power Up Check

Test Goal

Verify that RVP901 and RVP902 properly power up.

To run the test, apply power to RVP902 (or reset it) and RVP901.

Checklist

Table 83 Power Up Checklist

Task	Checked OK/Not OK	Remarks
When power is applied to RVP901: • Red and green lights (D7) may blink on boot and then green light is solid • The red light is solid if software is running on server and blink if software is not running.		
When the CAT5E cable is disconnected from either RVP902 server or RVP901: • Green light (D7) remains on. • Red light blinks indicating it has lost connection to the software.		
Test Passed		
For Customer	Date:	
For Vaisala	Date:	

A.5 Terminal Setup Check

Test Goal

Verify that the TTY Setups are accessible and functioning properly.

Test Equipment

- Keyboard andmouse
- Monitor (KVM) are installed locally or on a remote computer (with DspExportrunning)

Test Procedure

Follow the steps in A.5.1 Checking Terminal Setup (page 340).

Checklist

Table 84 Terminal Set Up Checklist

Task	Checked OK/Not OK	Remarks
Procedure in A.5.1 Checking Terminal Setup (page 340)completed successfully.		
Test Passed		
For Customer	Date:	
For Vaisala	Date:	

A.5.1 Checking Terminal Setup

1. To access the TTY menu:
   a. In the serial TTY or host computer interface, type: **dspx**
   The following prompt is shown:

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

   b. On the TTY, press ESC.
   RVP900 lists the RVP and IRIS software versions and shows a command prompt:

```
Vaisala Incorporated, USARVP9 Digital IF Signal Processor V13.1
IRIS-8.13.1
--------------------------------------------------
RVP9>
```

2. To view RVP and IRIS software versions, type **V**
3. For a list of available commands, type **help**
4. To exit the menu and reload RVP900 with the changed set of current values, type: **Q**
   Settings are saved in non-volatile RAM so they take immediate effect on start-up.

You must exit the menus using the **Q** before resuming normal RVP operation. Portions of the RVP command interpreter continue to run while the menus are active, but the processor does not function until you exit the menus.

5. To return to the Linux command prompt, press **SHIFT > C**.

A.6 Checking Internal Status with Setup **V** Command

Test Goal

Verify that the TTY setups for the Internal Status section are properly reported.

Test Equipment

KVM installed

Test Procedure

1. Enter the TTY setups through **dspx**.
2. Issue the **V** command to display the internal status.

Checklist

Table 85 Internal Status Setup Checklist

Task	Checked OK/Not OK	Remarks
Test procedure completed successfully.		
Status information is correct and shows no faults. See 5.1.2 V and Vz – View Card and System Status (page 91).		
Test Passed		
For Customer	Date:	
For Vaisala	Date:	

A.7 Checking Board Configuration with Setup **Mc** Command

Test Goal

Verify that the TTY setups for the Board Configuration section are properly configured for the customer application.

Test Equipment

KVM installed

Test Procedure

1. Enter the TTY setups through **dspx**.
2. Issue the **Mc** command to display the internal status.
3. Set all the values, as required, for your operation.

Checklist

Table 86 Board Configuration Checklist

Task	Checked OK/Not OK	Remarks
Test procedure completed successfully.		

Task	Checked OK/Not OK	Remarks
Status information is correct and shows no faults. See 5.2.2 Mc — Top-level Configuration (page 104).		
Parameters have been set.		
Test Passed		
For Customer	Date:	
For Vaisala	Date:	

A.8 Checking Processing Options with Setup **Mp** Command

Test Goal

Verify that the TTY setups for the **Processing Options** section are properly configured for the customer application.

Test Equipment

KVM connected

Test Procedure

1. Enter the TTY setups through `dspx`.
2. Issue the **Mp** command.

Checklist

Table 87 Processing Options Setup Checklist

Task	Checked OK/Not OK	Remarks
Test procedure completed successfully.		
See 5.2.4 Mp — Processing Options (page 108).		
Test Passed		
For Customer	Date:	
For Vaisala	Date:	

A.9 Checking Clutter Filters with Setup **Mf** Command

Test Goal

Verify that the TTY setups for the Clutter Filters section are properly configured for the customer application.

Test Equipment

KVM connected

Test Procedure

1. Enter the TTY setups through **dspx**.
2. Issue the **Mf** command to display the internal status.
3. Set all the values, as required, for your operation.

Checklist

Table 88 Clutter Filters Setup Checklist

Task	Checked OK/Not OK	Remarks
Test procedure completed successfully.		
Parameters set. See 5.2.3 Mf — Clutter Filters (page 106).		
Test Passed		
For Customer	Date:	
For Vaisala	Date:	

A.10 Checking General Trigger Setup with Setup **Mt** Command

Test Goal

Verify that the TTY setups for the General Trigger Setup section are properly configured for the customer application.

Background

RVP900 can output up to 12 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 A.16 Burst Pulse Alignment (page 350).

In the following table, 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 µsec.

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	_____ _____ _____	_____ µsec	_____ µsec
1	_____ _____ _____	_____ µsec	_____ µsec
2	_____ _____ _____	_____ µsec	_____ µsec
3	_____ _____ _____	_____µsec	_____ µsec
4	_____ _____ _____	_____ µsec	_____ µsec
5	_____ _____ _____	_____ µsec	_____ µsec
6	_____ _____ _____	_____µsec	_____ µsec
7	_____ _____ _____	_____µsec	_____µsec
8	_____ _____ _____	_____µsec	_____ µsec
9	_____ _____ _____	_____µsec	_____ µsec

Test Procedure

1. Enter the TTY setups through **`dspx`**.
2. Issue the **`Mt`** command to display the internal status.
3. Set all the values, as required, for your operation.

ThePRF and pulse width set here are the current values and the values used at power up.

Checklist

Table 89 General Trigger Setup Checklist

Task	Checked OK/Not OK	Remarks
Test procedure completed successfully.		
Parameters set See 5.2.5 Mt — General Trigger Setups (page 114).		
Test Passed		
For Customer	Date:	
For Vaisala	Date:	

A.11 Initial Setup of Information for Each Pulse Width

Test Goal

Enter the initial values for the TTY Setups for each of the pulse widths.

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. RVP900 supports up to four pulse widths (coded 0 to 3), although most transmitters typically support only 2 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	_____ µsec	_____ Hz
1	_____ µsec	_____ Hz
2	_____ µsec	_____ Hz

#	Pulse Width	Max PRF
3	_____ µsec	_____ Hz

Test Procedure

1. Enter the TTY setups through **dspx**.
2. Issue the **Mt #** command, once for each pulse width.
3. Enter the start time and widths for each trigger as shown in 5.2.6 Mt<n> — Triggers for Pulsewidth n (page 117).
4. For all unused triggers, set the width to 0.
5. Enter the Maximum PRF from the chart above.
6. 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.

Checklist

Table 90 Pulse Width Information Checklist

Task	Checked OK/Not OK	Remarks
Parameters are set. See 5.2.6 Mt<n> — Triggers for Pulsewidth n (page 117)		
Test Passed		
For Customer	Date:	
For Vaisala	Date:	

A.12 Checking Burst Pulse and AFC with Setup **Mb** Command

Test Goal

Verify that the TTY setups for the Burst Pulse and AFC Configuration section are properly configured for the customer application.

Background: 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 second trip echo filtering and recovery.

The frequency measurement is used to implement an analog (±10V) AFC output to control the STALO frequency.

An external AFC can be used instead of RVP900 AFC, but it is not recommended.

Background: Klystron or TWT-based Systems

The COHO is measured instead of the burst pulse.

Klystron systems that use a phase shifter should input the phase shifted COHO into the IFD, so that RVP900 can digitally lock to the transmitted phase.
For Klystron systems, the AFC feedback loop is unused.

Test Equipment

KVM connected

Test Procedure

1. Enter the TTY setups through **dspx**.
2. Issue the **Mb** command, once for each pulse width.
3. Set all the values as required.

Checklist

Table 91 Burst Pulse and AFC Set Up Checklist

Task	Checked OK/Not OK	Remarks
Parameters are set. See 5.2.1 Mb — Burst Pulse and AFC (page 95)		
Test Passed		
For Customer	Date:	
For Vaisala	Date:	

A.13 Checking Debug Options with the M+ Command

Test Goal

Verify that the TTY setups for the Board Configuration section are properly configured for the customer application.

Background

RVP900 supports several test features that are configured in this section.

Vaisala recommends that the LEDs be set to **`1:Go/Proc`** so that the front panel red LED flashes during each processing cycle.

For operational systems, turn off the simulation feature.

Test Equipment

KVM connected

Test Procedure

1. Enter the TTY setups through **`dspx`**.
2. Issue the **`M+`**.
3. Set all the values, as required.

Checklist

Table 92 Debug Options Checklist

Task	Checked OK/Not OK	Remarks
Test procedure completed successfully.		
Parameters are set. See 5.2.8 M+ Debug Options (page 128).		
Parameters have been set.		
Test Passed		
For Customer	Date:	
For Vaisala	Date:	

A.14 Checking Transmitter Phase Control with Setup **`Mz`** Command

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.

Test Equipment

KVM connected

Test Procedure

1. Enter the TTY setups through **dspx**.
2. Issue the **Mz** command.

Checklist

Table 93 Transmitter Phase Control Setup Checklist

Task	Checked OK/Not OK	Remarks
Test procedure completed successfully.		
Parameters are set. See 5.2.7 Mz — Transmissions and Modulations (page 126).		
Test Passed		
For Customer	Date:	
For Vaisala	Date:	

A.15 Ascope Test

Test Goal

Verify that the **Ascope** utility functions properly.

Background

The **Ascope** utility provides independent radar control and plotting for testing the radar and RVP900.

Test Equipment

KVM connected

Test Procedure

1. Enter the TTY setups through **dspx**.
2. Issue the **ascope** command.
3. Verify that the **Ascope** utility launches and that the data display begins updating.
4. Configure a **DEFAULT** startup and save it.
5. Exit and restart **Ascope**.
6. Verify that the default startup is properly restored.

Checklist

Table 94 Ascope Setup Checklist

Task	Checked OK/Not OK	Remarks
Test procedure completed successfully.		
Parameters are set. See TTY nonvolatile setups in *IRIS and RDA Utilities Guide*.		
Test Passed		
For Customer	Date:	
For Vaisala	Date:	

A.16 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.

Test Equipment

KVM connected

Test Procedure

1. Use **Ascope** to select the desired pulse width at a saved PRF.
2. Enter the TTY setups through **dspx**.
3. Issue the **Mb** and turn off burst pulse tracking.
4. Set the transmitter to radiate.
5. Issue the **Pb** and turn off burst pulse tracking.
   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.
6. Adjust the width of the burst pulse window using the **I/i** command to be slightly larger than the burst pulse (for example, ~50%).
7. Verify that the burst pulse power is in the range +1 ... -12 dBm per the tabular display on the setup terminal.

8. Record the burst pulse power:

Pulse width #	
0	_____ dBm
1	_____ dBm
2	_____ dBm
3	_____ dBm

9. Repeat the procedure for each pulse width.

If you cannot find the burst pulse, try to detect the burst pulse on an oscilloscope connected directly to the IF burst line (ahead of RVP901).

On a magnetron radar, if the AFC does not work, the IF frequency may be outside RVP901 anti-aliasing filter bandwidth. To get it to work you may need to go to manual frequency control. If no burst pulse is detected, an experienced technician must service the radar. If the burst pulse power is too small or large, check the status of the attenuators or amps in the burst pulse signal path. You may need to adjust the gain by installing a fixed attenuator or amplifier.

Checklist

Table 95 Burst Pulse Alignment Checklist

Task	Checked OK/Not OK	Remarks
Test procedure completed successfully.		
Parameters are set. See 6.5 Pb — Plot Burst Pulse Timing (page 134).		
Parameters have been set.		
Test Passed		
For Customer	Date:	
For Vaisala	Date:	

A.17 Bandwidth Filter Adjustment

Test Goal

Set the band width filter for each pulse width.

Test Equipment

KVM connected

Test Procedure

1. Enter the TTY setups through **dspx**.
2. Issue the **Ps** command and view the results on the display scope.
3. Toggle the space bar to show both the spectrum of the burst pulse and the spectrum of the bandwidth filter response.
4. Use the **Z/z** command to zoom the burst spectrum plot to approximately match the height of the bandwidth filter response (which have a smoother shape than the burst pulse).
5. 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.
6. Repeat for each pulse width that is used (use the **Mt** command to change pulse width) and record:

Pulse width #	FIRLength	Bandwidth	DC Gain
0	_____ µsec	_____ MHz	_____ dBm
1	_____ µsec	_____ MHz	_____ dBm
2	_____ µsec	_____ MHz	_____ dBm
3	_____ µsec	_____ MHz	_____ dBm

Checklist

Table 96 Bandwidth Filter Setup Checklist

Task	Checked OK/Not OK	Remarks
Test procedure completed successfully.		
Parameters are set. See 6.6 Ps — Plot Burst Spectra and AFC (page 139).		
Test Passed		
For Customer	Date:	
For Vaisala	Date:	

A.18 Digital AFC (DAFC) Alignment (Optional)

Test Goal

Verify that RVP900 DAFC output controls the STALO over the correct span.

Background

RVP900 implements an AFC based on the measurement of the burst pulse frequency. The DAFC connects to the TRIG–A sma port of 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 Equipment

Setup TTY

Test Procedure

1. Enter the TTY setups through **dspx**.
2. Use the setup terminal to set the Digital AFC span. See A.12 Checking Burst Pulse and AFC with Setup Mb Command (page 346).
3. 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.
4. Set to MFC using the **=** command, and adjust the control to the lowest setting using the **D** command.
   Record the results in the following table.
5. Raise the control using **U** to within 0.1 MHz of the IF frequency.
   Record the results in the following table.
6. Raise the control using **U** to the highest setting.
   Record the results in the following table.
7. 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

Checklist

Table 97 DAFC Setup Checklist

Task	Checked OK/Not OK	Remarks
Test procedure completed successfully.		

Task	Checked OK/Not OK	Remarks
Parameters are set. See 3.6 Digital AFC (DAFC) (page 51).		
Test Passed		
For Customer	Date:	
For Vaisala	Date:	

A.19 MFC Functional Test and Tuning (Optional)

Test Goal

Verify that the Manual Frequency Control (MFC) is functioning properly.

Skip this test if you are not using the RVP900 AFC.

Test Equipment

KVM connected

Test Procedure

1. Enter the TTY setups through `dspx`.
2. Type the **Ps** command (Plot burst spectrum and AFC).
3. Type 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.
4. 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, go to the **Mb** command menu and change the question **Burst frequency increases with increasing AFC voltage**.

5. Using the **U/u** and **D/d** commands, verify the limits of the AFC tuning and fill in the following table:

AFC %	Measured Frequency (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.
6. Toggle the MFC mode to AFC:
   a. Type: =
   b. Verify that the terminal indicator changes from **Manual** to**AFC**.
7. Exit the **Ps** menu.

Checklist

Table 98 MFC Setup Checklist

Task	Checked OK/Not OK	Remarks
Test procedure completed successfully.		
Parameters are set. See 6.6 Ps — Plot Burst Spectra and AFC (page 139).		
Test Passed		
For Customer	Date:	
For Vaisala	Date:	

A.20 AFC Functional Test (Optional)

Test Goal

Verify that the AFC properly tracks the burst pulse frequency.

Test Equipment

KVM connected

Test Procedure

1. Enter the TTY setups through `dspx`.
2. Enter the **Ps** mode and observe the output on a display scope.

3. 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 ±15 KHz 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 ... 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.
   - 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.
4. Perform the tests listed above for each pulse width and verify that the AFC properly tracks the center frequency:
   - For pulse width 0.
   - For pulse width 1.
   - For pulse width 2.
   - For pulse width 3.

Checklist

Table 99 AFC Setup Checklist

Task	Checked OK/Not OK	Remarks
Test procedure completed successfully.		
Parameters are set. See 6.6 Ps — Plot Burst Spectra and AFC (page 139).		
Test Passed		
For Customer	Date:	
For Vaisala	Date:	

A.21 Checking Input IF Signal Level

Test Goal

Verify that the input signal level is optimized for RVP901 by observing the power in the noise using the **Pr** command.

You can perform this procedure with the transmitter off since, in theory, it is only measures receiver properties. However, you may notice some noise interaction between the Tx and Rx.

Test Equipment

KVM connected

Test Procedure

1. Set the transmitter to radiate and elevate the antenna to >45° to minimize the effects of weather or clutter echoes (including earth noise).
   Make sure the antenna azimuth points away from the sun or any known RF interference sources.
2. Enter the TTY setups through **dspx**.
3. Enter the **Pr** command and use the display scope to view results.
4. Use the **Ll/Rr** commands to move out in range to a start range of 50 km, so that only noise is present.
5. Record the powers displayed on the setup terminal. Use the **V/v** command to increase/decrease averaging of samples to make the noise measurement more stable.
   Total: _____dBm
   Filtered:_____dBm
6. Remove the cable connecting the IF signal into the IFD. Record the powers again:
   Total: _____dBm
   Filtered:_____dBm
7. 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 4.2.11 Configuring IF Gain Based on System Performance (page 71).
8. Verify the rise in noise level.
9. Disconnect the output cable from the LNA and verify that the noise drops to the same level as when the IFD IF-Input was disconnected.
   This verifies that the dominant noise comes from the LNA, and not from any of the subsequent IF amplifiers.

Checklist

Table 100 Input IF Signal Level Setup Checklist

Task	Checked OK/Not OK	Remarks
Test procedure completed successfully.		
Parameters are set. See 6.7 Pr — Plot Receiver Waveforms (page 154).		
Test Passed		
For Customer	Date:	
For Vaisala	Date:	

A.22 Checking Calibration and Dynamic Range

Test Goal

Verify the receiver dynamic range is in excess of 80 dB.

CAUTION! 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 do not allow the output from the signal generator (accounting for cable and coupler losses) to exceed this value.

CAUTION! To avoid damage to the RVP901 (IFDR), do not increase the signal generator power such that the `Total Power` exceeds the `Safe Total Power Limit` for the **`Pr`** command display more than +12dBm.

Test Equipment

KVM connected

RF signal generator

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. Turn the radiate off, but leave the receiver on since the test signal generator may be damaged by the transmitter.
3. Elevate the antenna to >20° and point the azimuth away from any known microwave sources including the sun.
4. Use the setup terminal to enter the calibration gains, losses, and transmit power values.
5. Enter the TTY setups through **`dspx`**.
6. Use the **`ps`** command in the dspx utility to put RVP900 in manual AFC control by pressing **=** .
7. Connect the test signal generator to inject a signal at RF ahead of the LNA.
8. Press **%** to select the IF input under test.
9. Set the signal generator to a value that is approximately 20 dB above noise and observe the scope plot.
10. Adjust the frequency of the test signal generator to make the frequency of the spectrum to the correct IF frequency.
    You can also do this by using **`Pr`** to monitor the `Filtered` reading and selecting the frequency setting with the highest dBm reading.
    `Filtered`: _____dBm

11. Open the **ZAUTO** utility and perform a calibration:
    a. Begin at +10-dBm and decrease the `SIGGEN` power by 10 dB for each sample.
    b. At the saturation point and noise floor use 1 dB steps to carefully define the roll-offs.
    c. Select the lower and upper bounds of the linear portion of the receiver.
    d. Select **FIT**.
    e. Turn off the `SIGGEN RF` output and select **NOISE**.
    f. Verify that the receiver dynamic range is greater than or equal to 95 dB.
    g. Check that the signal generator frequency has not drifted by looking at the plot.
    h. If it is off by more than 0.1 MHz, retune and repeat the test.
12. In the `dspx` utility, put RVP900 back in automatic AFC control by typing the **`ps`** command and pressing =.
13. Record RVP900 calibration values.

Checklist

Table 101 Calibration and Dynamic Checklist

Task	Checked OK/Not OK	Remarks
Test procedure completed successfully.		
Parameters are set. See 6.7 Pr — Plot Receiver Waveforms (page 154).		
Test Passed		
For Customer	Date:	
For Vaisala	Date:	

A.23 Checking Receiver Bandwidth

Test Goal

Verify the receiver bandwidth is in excess of 14 MHz.

Background

For proper functioning of the high speed A/D convertors, RVP requires approximately 14 MHz of broadband noise.

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 A.22 Checking Calibration and Dynamic Range (page 358).

For dual polarization systems, you can expect a narrower bandwidth.

Test Equipment

KVM connected

RF signal generator

Test Procedure

1. Connect the test signal generator to inject a signal at RF ahead of the LNA.
2. Enter the TTY setups through **dspx**.
3. 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 µsec.
   - Use the **T/t** command to set the plot span to 50 µsec.
   - Use the **V/v** command to set averaging to 1 sample.
4. Set the signal generator power to a value that is approximately 60 dB above noise and observe the scope plot.
5. 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 the following graph.
6. Verify that the 3 dB point gives approximately 14 MHz of bandwidth.

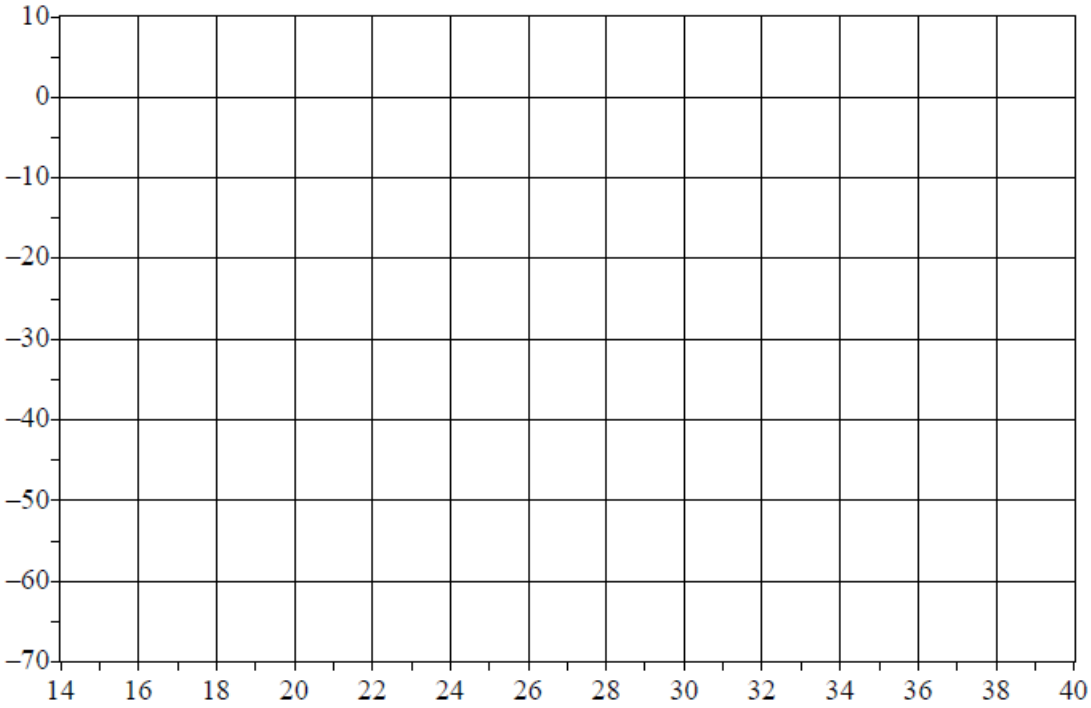


Figure 54 Total Power and IF Frequency

Checklist

Table 102 Receiver Bandwidth Checklist

Task	Checked OK/Not OK	Remarks
Test procedure completed successfully.		
Parameters are set. See 6.7 Pr — Plot Receiver Waveforms (page 154).		
Test Passed		
For Customer	Date:	
For Vaisala	Date:	

A.24 Checking Receiver Phase Noise

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, radar's must STALO maintain a stable frequency, and that the transmitted pulse must not contain any amplitude or phase artifacts.

Test Equipment

Ascope utility. See *IRIS and RDA Utilities Guide*.

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.

5. Try minor changes in **Az**, **El**, and Range to get the lowest phase noise. The goal is less than 1° within 20 km.
6. 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:_____

Checklist

Table 103 Receiver Phase Noise Checklist

Task	Checked OK/Not OK	Remarks
Test procedure completed successfully.		
Test Passed		
For Customer	Date:	
For Vaisala	Date:	

A.25 Hardcopy and Backup of Final Setups

Test Goal

Make a hardcopy of all the final setups and attach to the tests.

Test Equipment

KVM connected

Printer or ftp access to a computer that supports a printer

TTY Setups Hardcopy Listing

1. Start script logging with the commands:

```
cd /usr/sigmet/config/listings
script RVP900.26feb09
```

2. Enter the TTY setups and type the **??** command to list all TTY setups.
3. Exit **dspx** and script logging by typing: **exit**
4. Print the file or ftp it to a computer that supports a printer.

/usr/sigmet/config Hardcopy Listings

1. Print the following files or ftp them to a computer that supports printing:
   - *setup_dsp.conf*
   - *rvp900.conf*
   - *softplane_dsp.conf*

Backup */usr/sigmet/config* directory

Checklist

Table 104 Backup and Hardcopy Checklist

Task	Checked OK/Not OK	Remarks
Test procedure completed successfully.		
Test Passed		
For Customer	Date:	
For Vaisala	Date:	

A.26 RVP901 TxDAC Stand-alone Bench Test

Test Goal

Verify that the RVP901 TxDAC electrical I/O is working properly in an isolated environment.

Test Equipment

IRIS **dspx** utility

IF signal generator

Test Procedure

1. Enter the TTY setups through **dspx**.
2. Run **dspx** and temporarily revert to factory settings with **f**.

3. 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
```

4. Enter the **Pr** menu and type **RRRTTZ** to move the starting plot range to 30 km, the plot interval to 20 µsec, and zoom factor to x2.
5. Type 2 space characters to switch to the spectral display plot and exit the plot by typing: **q**
6. In the **Mz** menu, setup the following:

```
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**.
7. 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.
8. 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.
9. Repeat the steps 5 to 8 on the TxDAC-B output port.
10. Apply a 0 dBm SigGen waveform at the selected reference frequency (for example, 10 MHz) to the CLK IN BNC input of RVP901.
11. Verify the **V** command shows `Tx/Clk:Okay`, indicating the RVP901 DAC is locking properly.
12. Return to the center frequency and verify that lock returns.

Checklist

Table 105 RVP901 TxDAC Stand-alone Bench Test Checklist

Task	Checked OK/Not OK	Remarks
Test procedure completed successfully.		
Test Passed		
For Customer	Date:	
For Vaisala	Date:	

Appendix B. RVP901 IFDR Technical Drawings

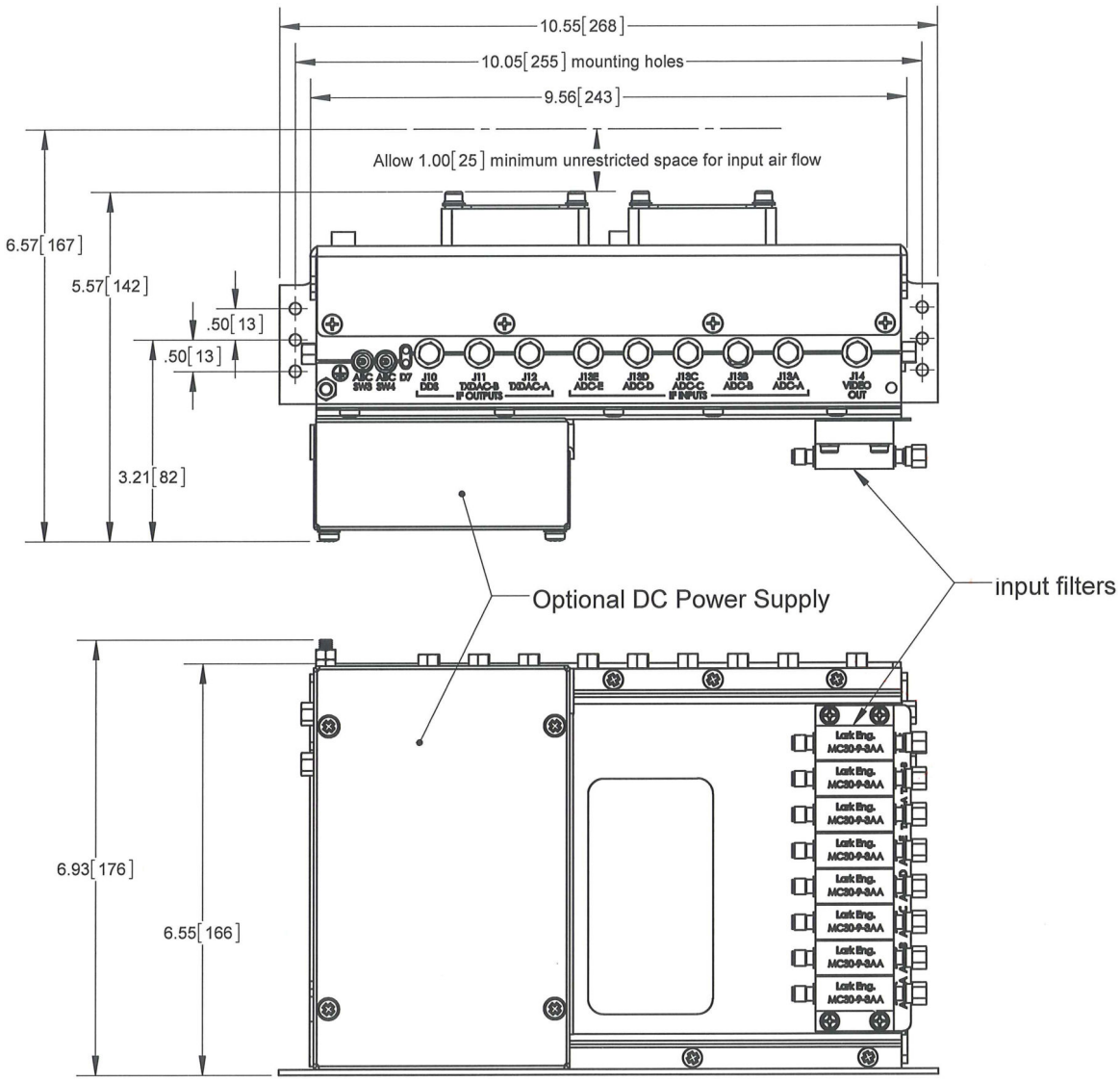


Figure 55 RVP901 IFDR - Top and Front Face

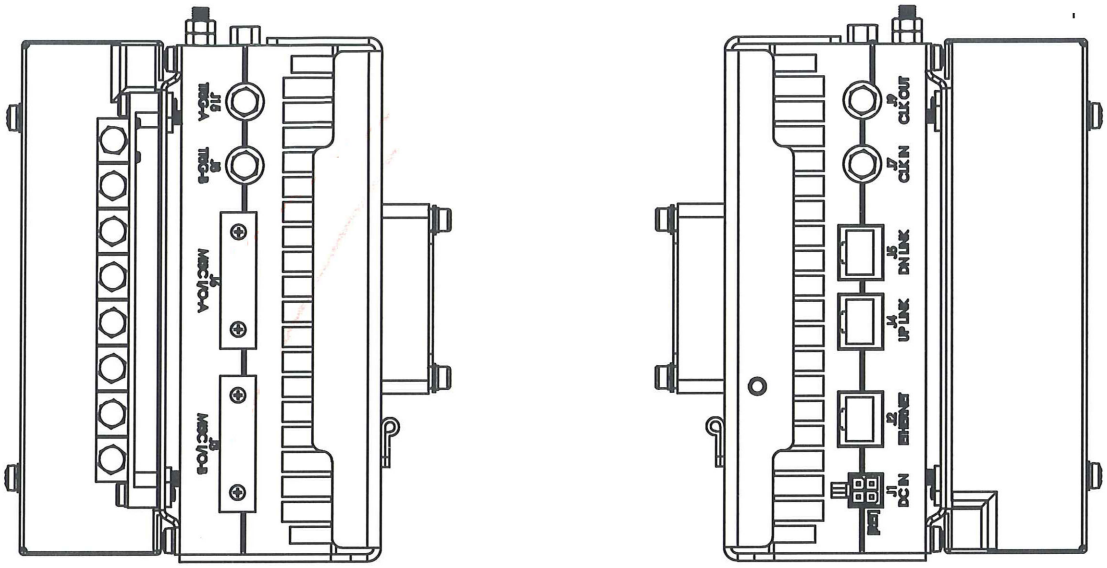


Figure 56 RVP901 IFDR - Right and Left Sides

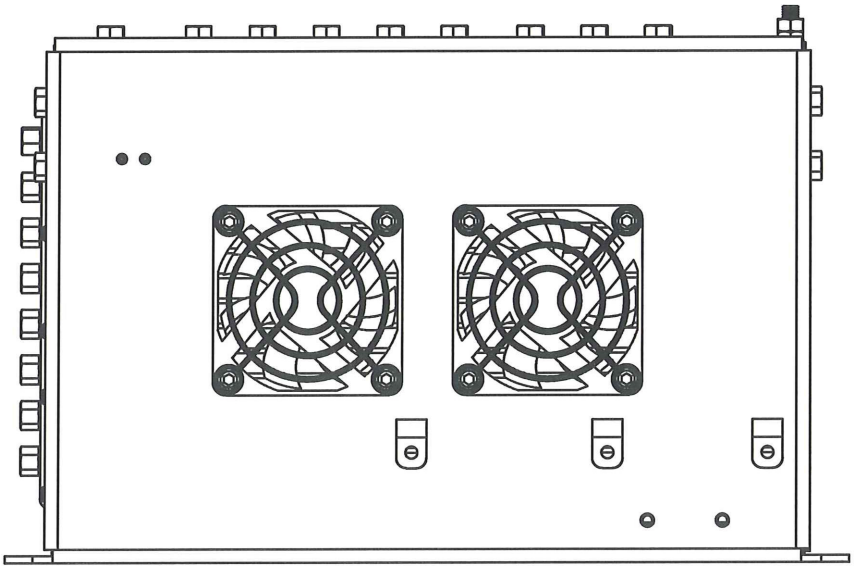


Figure 57 RVP901 IFDR - Fan Side (Heat Sink)

More Information

- RVP901 IF Digital Receiver (page 23)

Appendix C. RVP900 Developer Notes

C.1 Customizing RVP Software

Vaisala provides a software environment to third-party developers wishing to customize RVP algorithms.

This information only applies to organizations who have signed the Vaisala Software Developer's License Agreement.
This information is not relevant to the operational and scientific needs of most operators and users.

RVP is an open-architecture radar signal processor that uses Gigabit Ethernet to interface to the IFDR, which samples the data.

Using public APIs, researchers and OEM manufacturers can modify or replace existing algorithms, or write their own software using the RVP900 software as a foundation.

RVP software runs under standard CentOS Linux, and is developed and maintained using standard GNU tools (for example, **gcc**, **gdb**, **make**).

More Information

- Public API (page 42)

C.2 RVP Code Organization

RVP internal APIs provide abstraction from the underlying hardware. This means developers do not need to worry about kernel support, interrupts, resource allocation, timing details, and the interfaces to higher layers such as IRIS and its utilities.

Table 106 RVP Hardware and Software Organization

Component	Description
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. All of the RVP real-time functions are implemented at the chip level on one or more PCI cards. This allows the remainder of RVP 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.

Component	Description
Linux Kernel Module	This module is **insmod** at system boot time, provides all of the low-level PCI support for RVP hardware (Rx receiver card, Tx transmitter card, I/O-62 card, and so on.) It also provides the FIFO interfaces to the IRIS DSP driver and other services implemented at the kernel level.
PCI Card Driver Library	The library, along with the kernel module, constitute the low-level board support package for RVP. Most hardware details are 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.
RVP/Main Threads	This collection of Linux threads is the foundation and support core of RVP, but they do not run any of the (**I**,**Q**) data processing algorithms. They handle RVP 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 8.1 Host Computer Command Overview (page 231).
RVP/Proc Processes	These are **N**-copies of identical code that are forked by the **MAIN** threads and which carry out the scientific processing in RVP. These 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, and so on. 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 **RVP/Proc** process per available CPU on an SMP machine. Most customizations of the RVP algorithms occur in these processes. See 7.1 RVP Algorithm Overview (page 167).

Most **RVP/Proc** code is released to licensed developers as source code. There are open APIs for the **RVP/Main** that allow you to build custom trigger patterns, phase sequences, and so on, on the transmitter side. The kernel source is available so that developers can run within their own customized Linux environment.

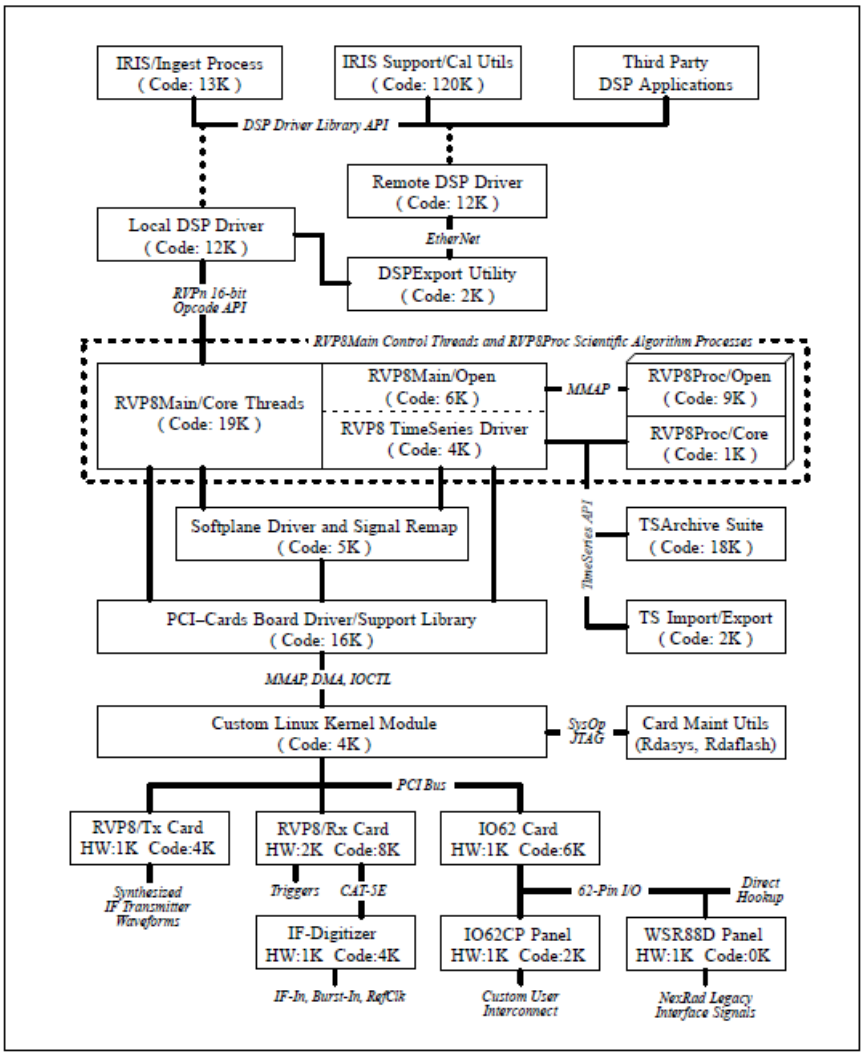


Figure 58 RVP Hardware and Software Organization

C.2.1 RVP Software Maintenance Model

In the Vaisala open source developer model, the open code is the delivered code. It is not example code, nor an abridged form of the final delivered algorithms.

- The *CORE* portion of the developer tree contains no source files, and is delivered only as compiled binaries.
- The *OPEN* portion of the tree is fully populated with the source files used by Vaisala to build each RVP release.
  Use the *OPEN* code as a reference for programming examples and ideas.
  Be aware that that *OPEN* code can change significantly with each release. Changes may be lost when you install the next RDA update.
- Use the *SITE* directory to add custom features to RVP.
  This directory is a collection of empty program stubs whose calling conventions are very stable. Each stub is a hook that can be used to define a new major mode. Replace these stubs with custom code to add custom major modes to RVP.

Customizations are typically done in the `rvp9main/open`, `rvpxproc/open`, `rvp9main/site`, and `rvpxproc/site` trees.

RVP source code is in the *`rvpxproc`* tree.

C.2.2 Installing Incremental RDA Upgrades

RDA development systems are upgraded using the procedures in *IRIS and RDA Software Installation Guide*. 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 *`RVP9/Main`* and *`RVP9/Proc SITE`* libraries with their default stubs, so you must rebuild your customizations.

Since the executables are built from the *`OPEN`* trees, the simplest method is to copy the new *`OPEN`* areas alongside your existing *`SITE`* directories. Your development cycle can then be carried out from the */`home/operator`* area, without compiling in the */`usr/sigmet`* tree directly.

```
$  cd  /home/operator/src/rda
$  rm  -rf rvp9main/open rvp9main/core rvp9proc/open rvp9proc/core
$  cp  -pr /usr/sigmet/src/rda/rvp9main/open rvp9main
$  cp  -pr /usr/sigmet/src/rda/rvp9proc/open rvp9proc
$ make clean
$ make -j2
$_make -j2 install
```

During the upgrade, you rebuild your code under the new header files that were installed during the upgrade. You may find a few minor incompatibilities in the form of changed structure definitions 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. RVP does not let you run with incompatible headers.

You may receive a *`RVP9/Main`* startup errors such as:

```
RVP9/Main code version mismatch Core: Version=2.12 Build=298 Open:
Version=2.12 Build=298 Site: Version=2.11 Build=295
```

and *`RVP9/Proc`* errors later in the initializations such as:

```
Forking parallel compute process(es)... RVP9/Proc-0: Requesting exit due to
signal 1
RVP9/Main: UNIX Signal: Unexpected RVP9/Proc termination RVP9/Main: RVP9/Proc
version mismatch <ProcSite: Ver=2.11 Bld=295>
```

C.3 Debugging and Profiling Your Code

While RVP is a complex multi-thread and multi- process system, it is a user-level application running under the Linux operating system.

You can debug most of your custom code using tools familiar to Linux/C/GNU programmers.

C.3.1 Monitoring Opcode/Data Activity: –exposeIO

RVP is generally controlled by higher level application such as **Ascope**, **IRIS**, and **dspx**, which communicate with RVP through the IRIS DSP Driver Library using the opcodes (see 8.1 Host Computer Command Overview (page 231)). These layered applications provide a clean and maintainable signal processor interface.

The complete opcode activity, between the application driver and RVP, can be viewed by including the **-exposeIO** flag on RVP startup command line. The following printout shows the output example when you type an **Output Test** opcode, followed by a **Noise Sample** command and a **Get Processor Parameters** opcode:

```
Opcode 0x0004 (OTEST)
Output Words0:0001 0002 0004 0008 0010 0020 0040 0080 0100 0200 0400 0800
10002000 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 RVP9/Main HostCmds thread, which also generates the exposeIO printout. The opcode is shown in both numerical and mnemonic form, followed by the 16-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 RVP.

C.3.2 Showing Live Acquired Pulse Information: **-showAQ**

The (`I`,`Q`) data computed by the FIR filters are transferred to CPU memory using direct memory access (DMA) bus cycles initiated by `RVP9/Main` threads.

The radar data arrive in discrete chunks and the `TimeSeries` API is updated accordingly.

To show live acquired pulse information for debugging, include the **-showAQ** command option in the RVP startup command line.

The following example shows the output (`I`,`Q`) bus activity for a dual polarization system in which 3 UDP packets were sent to transfer the data.

`UDP` column
: Record ID that increments each time a pulse is sent.

`Frags` column
: Shows the number of UDP packets sent to transfer the last pulse of data.

`Bins` column
: Number of bins requested multiplied by two, because this is a dual polarization system.

`PRT` column
: Pulse repletion time for the last and the next pulses.

`Az` and `El` columns
: Azimuth and elevation reported by the IFDR.

`iMisc` column
: You may ignore this information.

```
UDP:1    Frags:8150/8150/120  Bins:4086/0  PRT:324000/324000  iMisc:3e945c85
Az:178.32 El:148.05
UDP:7    Frags:8150/8150/120  Bins:4086/0  PRT:324000/324000  iMisc:52aa61c1
Az:145.02 El:139.38
UDP:17   Frags:8150/8150/120  Bins:4086/0  PRT:324000/324000  iMisc:6a356e76
Az:164.34 El:163.44
UDP:27   Frags:8150/8150/120  Bins:4086/0  PRT:324000/324000  iMisc:7b267be2
Az:161.64 El:163.22
UDP:37   Frags:8150/8150/120  Bins:4086/0  PRT:324000/324000  iMisc:7ce87bd0
Az:114.69 El:136.73
UDP:47   Frags:8150/8150/120  Bins:4086/0  PRT:324000/324000  iMisc:5c976dc6
Az:148.14 El:153.97
UDP:57   Frags:8150/8150/120  Bins:4086/0  PRT:324000/324000  iMisc:661e61e9
Az:173.00 El:173.93
UDP:67   Frags:8150/8150/120  Bins:4086/0  PRT:324000/324000  iMisc:74b673e8
Az:175.36 El:173.93
UDP:77   Frags:8150/8150/120  Bins:4086/0  PRT:324000/324000  iMisc:733d7451
Az:130.34 El:154.59
UDP:87   Frags:8150/8150/120  Bins:4086/0  PRT:324000/324000  iMisc:548062a8
Az:143.92 El:138.08
UDP:97   Frags:8150/8150/120  Bins:4086/0  PRT:324000/324000  iMisc:6a676e83
Az:164.33 El:163.37
UDP:107  Frags:8150/8150/120  Bins:4086/0  PRT:324000/324000  iMisc:7b357c03
Az:161.99 El:163.31
UDP:117  Frags:8150/8150/120  Bins:4086/0  PRT:324000/324000  iMisc:7c757b96
Az:119.01 El:138.88
UDP:127  Frags:8150/8150/120  Bins:4086/0  PRT:324000/324000  iMisc:5c8e6dbe
Az:148.25 El:154.04
UDP:137  Frags:8150/8150/120  Bins:4086/0  PRT:324000/324000  iMisc:651f60d4
Az:173.38 El:174.20
UDP:147  Frags:8150/8150/120  Bins:4086/0  PRT:324000/324000  iMisc:74e87431
Az:176.00 El:174.43
```

Use **-showAQ** with **verbose** to show information on every packet received as well as a report for every pulse (instead of every 10 pulses).

C.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 RVP startup command line. Here is a sample printout (while running **Ascope**):

```
CPI0: 64-Pulses (269.74,1.49) to (271.25,1.49) 126.00-ms  TS:1%
CPI1: 64-Pulses (270.51,1.49) to (272.02,1.49) 126.00-ms  TS:0%
CPI2: 64-Pulses (271.28,1.49) to (272.79,1.49) 126.00-ms  TS:0%
CPI3: 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 in each one is shown. The starting (**AZ**,**EL**) and ending (**AZ**,**EL**) are 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%, RVP has 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.

For more information, see the code in *rvp9main/open/cpi.c*.

C.3.4 Showing RealTime Callback Timers: -showRTCtrl

RVP can show the detailed activity of the 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 handling real-time events within RVP. For information on **struct rtCtrlCBTimer**, see C.5 Real-Time Control of RVP (page 381).

When no callbacks are registered, or when the registered callback does not request any further activity, the printout looks 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 timer mechanism goes to sleep until the next **ProcSection** begins.

In contrast, the following printout was generated by the demonstration histogram callback that is provided in *rvp9main/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(–)
```

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 5 counts on the trigger timer. The RVP trigger is selected as the clock source for timer #1.

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 regularly fires every 2.5 ms; and since we only get two or three trigger counts in that much time, timer #1 never fires at all. If the PRF were increased, however, we would see timer #1 counting up to 5 triggers and then firing before the 2.5 ms expires.

C.3.5 Using ddd on Main and Proc Code

The GNU **ddd** symbolic debugger is (usually) built on top of the **dde** command line debugger. Both tools are provided on all Linux systems.

1. To debug *RVP9/Main* code:
   a. (Optional) Type **b main** to cause **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 RVP. Before the libraries are loaded, it is possible that **ddd** cannot find all its symbols.
   b. Type:

```
$ cd /home/operator/src/rda/rvp9 main
$ make -j2
$ cd  open
$ ddd rvp9main
```

   The **SITE** and **OPEN** code is first compiled in the private development tree and then the RVP executable that resides in the **OPEN** area runs.
   c. To redefine the environment variable **OPTIMIZEFLAG=-g**, 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.
   d. Do one of the following:
   - Type **run** in the **ddd** execution window
   - Use **run -nod** to skip the powerup diagnostics and speed up your development cycle.

2. To debug *RVP9/Proc* code, in an X-Term window type:

```
$ rvp9 -noFork
```

which starts **RVP9/Main** threads, but pauses at the point that the **RVP9/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.

We do not want **RVP9/Proc** code to automatically start from the main threads.

When **RVP9/Main** is debugged in this way, signal events originating from **ddd** are also sent to **RVP9/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**.

3. In another terminal window:
   a. To builds the new **RVP9/Proc** code and start the debugger, type:

```
$  cd /home/operator/src/rda/rvp9proc
$ make -j2
$ cd open
$ ddd rvp9proc
```

   b. To start the subprocess, in in **ddd**, type; **run**
   When it has finished initializations **RVP9/Main** continues in the other window.
4. To debug *RVP9/Main* the complete way:
   a. Run **ddd rvp9** as described above, but include the **-noFork** flag in the **ddd run** command.
   b. Run **rvp9proc** manually in another window, either with or without its own **ddd**.

   You can create two simultaneous **Main** and **Proc** **ddd** sessions in this manner. Signals from one does not interfere with the other.

C.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 writing into it. Malloc/Free inconsistencies are also easily diagnosed with **valgrind**.

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 **RVP9/Proc** process, use:

```
$ cd /usr/sigmet/src/rda/rvp9proc/open
$ valgrind –tool=memcheck rvp9proc
```

You can download **valgrind** from http://valgrind.kde.org. .

C.3.7 Profiling with gprof

The GNU tools include the runtime profiler **gprof**.

This tool works with the C-compiler, and analyzes statistics in the ***gmon.out*** file that is produced when the running program exits.

1. The RDA ***Makefiles*** are setup to build profiled versions of either **RVP9/Main** or **RVP9/Proc** executables. Define one of the following environment variables:

   ```
   export PROFILE_RVP9MAIN="1"
   export PROFILE_RVP9PROC="1"
   ```

   If you are profiling RVP compute process, 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***.
   RVP does not let you profile both the **Main** and **Proc** executables at the same time.

2. Perform a **make clean** and **make install** in the **SITE** and **OPEN** portions of the relevant tree.
3. Run the profile analysis from the RVP and **rvp9proc** executables from their **OPEN** directories to allow **gprof** to find its symbols.
   For example:

   ```
   $ cd /home/operator/src/rda/rvp9proc/open
   <Run RVP for a while...>
   $ gprof rvp9proc – I .. /site
   ```

C.4 Creating New Major Modes from Old Ones

Developers can add custom algorithms to RVP by building on its concept of major modes and output data types. Each major mode defines all the methods required to compute each output data type from raw (**I**,**Q**) data.

Allowing users to define their own major modes provides all hooks required for full customization. You can code your own custom algorithms by making incremental changes to one of the Vaisala models, or you can start from scratch and build something 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 *rvp9main/open/mt_fft.c* and *rvpxproc/open/ct_fft.c*, each containing boilerplate top-level definitions.

1. To start creating your new major mode, copy the prototype code to separately named files in the *rvp9main/site* and *rvp9proc/site* areas, and then make the desired changes.
   There are 4 major mode slots reserved for custom user applications.
2. Define the names of your new modes using **Setup > RVP > Optional Data Parameters**. Names can be up to 15 characters long, and the text you choose automatically appears in utility menus such as **Ascope**, **IRIS**.
3. To invoke your new code, modify one of the lines of *rvp9main/site/mt_user.c* and *rvpxproc/site/ct_user.c*.
   These simple files 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 ) {}
```

4. During execution, call your custom initialization routines from one of the predefined user stubs:

```
void mtInitMajorMode_user1( void ) {initMajorMode_myCustomMode() ; }
```

C.4.1 Function Pointers for 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 RVP threads are governed by the following methods:

```
struct rvp9MainMajorMode { /* 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       ;
} ;
```

This structure allows new objects to inherit existing properties of old objects, while still permitting individual personality to enter as needed. RVP is written in standard C, and uses function pointer replacement to accomplish customization.

The `initMajorMode()` routine, which creates each major mode in the first place, is not part of this list because it is separately invoked (see C.4 Creating New Major Modes from Old Ones (page 379)) in order to first establish the list.

For descriptions of the methods, see *`rvp9main.h`* and *`rvp9proc.h`*.

Table 107 Summary of Customization Methods

Method	Description
`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 ofmethods at the start of each major mode. The `exitMajorMode()` handler is responsible for releasing all private memory resources.

Method	Description
`exitMajorMode_f`	This routine releases all resources that were allocated during the initialization and execution of the major mode. The **EXIT** routine is called if the major mode changes, or if we are between modes, for example, after a TTY Chat **q** command, or a processor reset. RVP is guaranteed not to be within an active **PROC** section when `exitMajorMode()` is called. That is, if `exitProcSection()` must be invoked, that always happens first.
`initProcSection_f` `exitProcSection_f`	Pair of routines called if RVP 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 RVP compute threads for the current major mode. The **EXIT** routine is called if 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.

C.5 Real-Time Control of RVP

RVP9/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, and so on. The **RT-Ctrl** thread runs at a priority that is higher than any other RVP 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. 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.

C.5.1 Using 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 when live (**I**,**Q**) data are 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 3 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 feature is implemented in hardware on the RVP901 IFDR, 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 IFDR-level.

C.5.2 Standard Trigger and Antenna Event Example

See the *rvp9main/open/rtctrl.c* file for the standard RVP code for live trigger control and angle synchronization.

The example consists of `initProcSection_dflt` and `rtCtrlCBF_trigs_dflt`.

The interrupt latencies are absorbed differently for each case:

- For angle syncing, the expected crossing time is computed based on the location of the antenna when the callback was entered. If a callback is delayed, a shorter future time is computed.
- For dual PRF triggers, the next interrupt must be shortened. When the callback is delayed, the `iTimerError` field tracks the length of the delay (measured in timer events).

`initProcSection_dflt`

This is the default routine for the initial entry into each **PROC** section. It primarily chooses the real-time callback responses for the current RVP configuration.

The routine sets up an area of private memory that later controls the real-time activity. See the calling example in *rvp9main/open/mt_fft.c*.

`rtCtrlCBF_trigs_dflt`

The real time callback strategies for controlling the trigger generator include:

- *Angle synchronization trigger algorithms*
  The following parameters define the maximum tracking rate for angle syncing:

```
#define SYNCSPAN
16  /* Number of sync angle machine states */
#define SYNCPERIOD 150000  /* Callback period in microseconds */
```

  By default the **SYNCSPAN** of 16 applies ± 8 sync angles to be tracked with a call-back period of 0.15 msec. For angles spaced 1°, the maximum antenna speed can be calculated to be:

```
*  Degree
*  ( 8 Angles ) * ( 1 ------ ) / ( 0.15 sec )  =  53 deg/sec
*  Angle
```

- *Ray-by-ray processing*
  Configures the RVP901 state based on pulse counts that are not angle synchronized. For single PRF, the RVP901 state sets the PRF, counts for the desired number of triggers and repeats.
  For dual PRF, depending on the programming, it stays at the current PRF or switches to a new one.

C.6 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 improve the performance of computation intensive signal processing applications.

The data types used in the IPP library are mostly compatible with the standard Vaisala **`typedefs`**. The IPP routines can almost always be called directly from RVP code. Status codes from the IPP library can be converted to RDA messages with **`sigIppStatus()`**.

To see examples, use **`grep`** to find RVP sources for **`ipps`**.

Use the header file **`intelipp_lib.h`** to define the IPP library entry points. Do not include the Intel headers directly because this bypasses some RDA consistency checks and makes your code less maintainable.

To get started with IPP, read the example file, **`rda/intelipp/ipp_example.c`**. It allows you to run DFT's and matrix transposes of sizes, and provides a simple method of benchmarking the IPP library on your hardware.

For more information, see https://software.intel.com/en-us/intel-ipp.

Licensing

The IPP runtime libraries are in each RDA release.

Vaisala has purchased a single-user developer's license from Intel which allows one engineer to develop code then links to the IPP library, and then to distribute an unlimited number of copies of that code in executable form. The license does not allow any additional programmers to develop their own code as an extension of Vaisala's license.

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 with Intel. in order to write new code that uses the IPP library.

Appendix D. Time Series Recording

D.1 Time Series Overview

Time series (TS) recording enables recording and playing back (**I**,**Q**) data.

Operators can record time series and then use their own off-line programs to process the data.

The time series can be recorded directly on an RVP or on a separate archive host through a gigabit network.

Table 108 Time Series Software Components

Software Component	Description
RVP900 TS API	Where time series reside in memory in 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	Records time series to a local disk. This can be on RVP or a separate networked archive host. Supports both archive and playback. (licensed separately)
Ascope utility	Can be used in playback mode to view either raw time series or processed results from RVP900 processing algorithms.
Tsview utility	Used to view diagnostic printouts of stored time series files. The complete source code is provided to assist users with writing their own applications.

Only the **Tsarchive** is licensed separately. You can write your own TS record/playback software and still take advantage of the network features such as **Tsimport** and **Tsexport**.

D.2 TS Record and Playback Software Architecture

The TS record and playback features on a local RVP900 and a remote archive host share a common software architecture. The following figure shows the most general case.

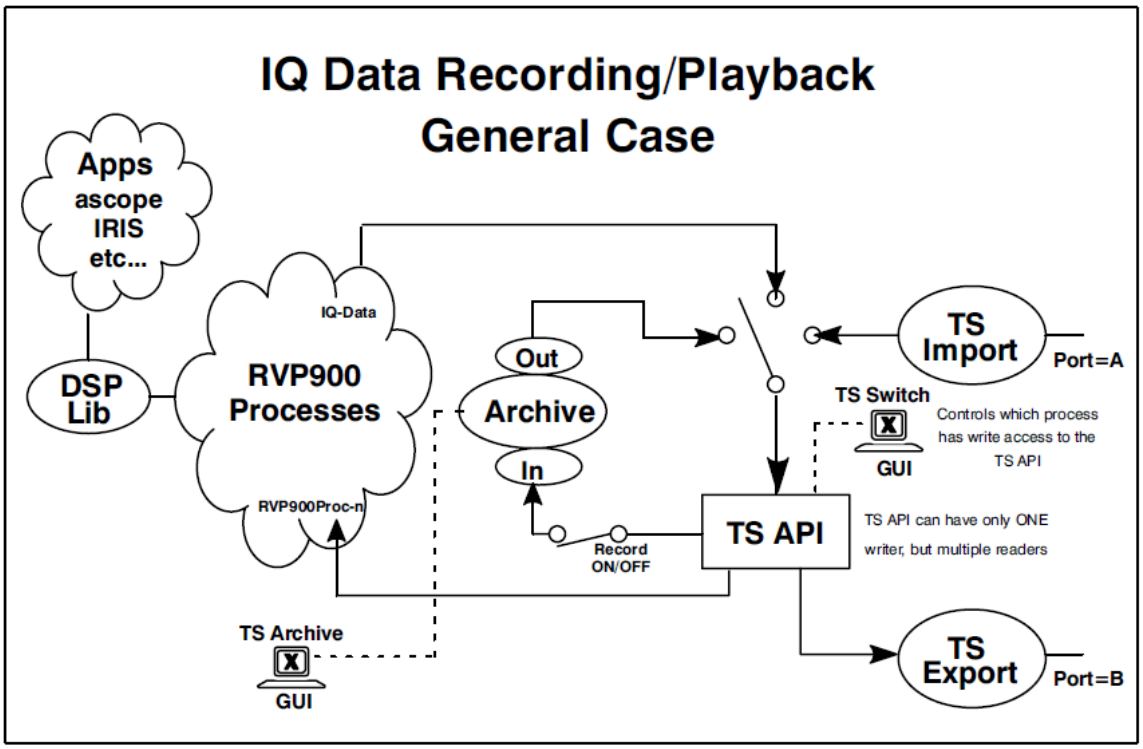


Figure 59 IQ Data Recording/Playback General Case

The structure shown 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 make a strong distinction between RVP900 and a separate archive host operating in record or playback mode.

Table 109 Description of `IQ` Data Recording/Playback General Case

Process	Description
`TS API`	Where the time series data reside in memory. The time series can be written to the TS API from one of the following sources: • From a local RVP900 • From a local disk archive • From a remote network host (with `tsimport`)
`tsswitch`	GUI that allows the user to select the sole TS writer from among these three possible sources. See D.5 TS Switch Utility (page 391).
`tsarchive`	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. `Tsarchive` also shows an inventory of the TS files and has a filter/search for locating specific groups of files (for example, by date and time). See D.6 TS Archive Utility (page 393).

Process	Description
`tsimport` `tsexport`	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` In the example, `2 parameters/bin/pulse` are the `I` and `Q` values, which are represented by 16 bits each (floating point). For dual polarization systems with two receiver channels, the data rate would be doubled. The 32 Mbit/sec basic data rate here would fit comfortably 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. See D.4 Installing and Configuring TS Recording (page 389).
RVP900 Processes	Collection of processes that are only present on an RVP900 machine. The 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 moments. To view these processes, use the **v** command in **dspx**. A remote archive host is most likely not from RVP900. In this case, these processes do not exist.

D.3 Using RVP TimeSeries API

The `TimeSeries` API is the interface through which (`I`,`Q`) data are made available to the application code that requires them.

This API is central to the design and operation of RVP and is used by the parallel computer processes to access incoming time series data.

The `TimeSeries` API is provided within a larger collection of RDA support services in *`rda/rdasubs`*. For API definitions, see the header file documentation.

D.3.1 Reader and Writer Clients

The `TimeSeries` API is 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. 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 checked.

The API is most valuable in providing random access to the recent buffered (`I`,`Q`) data and can buffer a couple of seconds of data within the `TimeSeries` API. This means that programs using the API only need to check the data every 50 to 100 msec without the risk of losing any data. This has only a minimal load on the CPU.

D.3.2 Attach and Detach Details

Use `rvptsAttach()` to attach and `rvptsRelease()` to release the connection.

Always attach to unit `RVPTS_UNIT_MAIN`. The others are for internal RVP use. You must also specify your client type. 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.

For commands, see *`program /rda/tsapi_lib/`*. The same directory includes the *`rvpts_example.c`* example.

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.3.3 Extracting Pulses with Sequence Numbers

Because there is a ring buffer filled with time series, a reader should first get the most recent pulse, and start reading after that. We do that with the function `rvptsCurrentSeqNum()`.

Check 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 RVP is reconfigured, 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()` is for the same acquisition mode.

D.3.4 Using Memory Bandwidth Effectively

Reading each pulse of data individually is inefficient. Instead, a smart reader sleeps until, for example, 10 pulses have arrived before reading them.

To support this, function calls find out how many pulses are available after a give sequence number, and how old a given pulse is.

For more information, see `tsexport.C`.

D.4 Installing and Configuring TS Recording

Vaisala recommends using TS recording in two configurations. The simplest (but least flexible) configuration is where RVP900 also serves as the archive host.

Create a separate partition should to prevent system crashes.

The ideal configuration is with 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, RVP900 can be equipped with a separate disk for archive operations.

In two system configurations, the `tsimport` and `tsexport` modules must be turned on at boot time on both systems. The UDP ports for the sending and receiving system must also be configured.

D.4.1 Required Software for TS Recording

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.

Table 110 Required Software for TS Recording

System Setup	Required Software
RVP900/RCP8	No additional software required
Archive System (IRIS)	Install RDA You must select the **Keep old files** option during installation.
Archive System (with out IRIS)	Install RDA

D.4.2 Configuring UDP Ports for TS Recording

You can configure the UDP ports by editing the `tsimport` and `tsexport` scripts in */usr/sigmet/config_template/rc.d/*.

Copy the files to */etc/rc.d/init.d/*.

If you plan to use `tsarchive` with a separate archive host, you must edit the `tsimport` and `tsexport` files. The export port on the sending system must match the import port on the receiving system so that a connection can be made between the two ports.

Table 111 Recommended UDP Ports for TS Recording

RVP900	Archive Host
TSEXPORT: 30780	TSEXPORT: 30781
TSIMPORT: 30781	TSIMPORT: 30780

Each script contain extensive comments and should be edited to suit your configuration.

The scripts are shipped configured for the RVP900 end, so you must edit the archive host's files.

You must also edit all **tsexport** files to explicitly set the target IP address to which it broadcasts the time series. Vaisala recommends that you use a single target host. A broadcast address can be used, but make sure that all recipients can handle the traffic and that there are no **10baseT** network sections.

D.4.3 Configuring Automatic Startup of tsimport and tsexport

1. Login as **root**.
2. Use the **chkconfig** command to add the processes:
   - **CentOS6**:

```
#chkconfig --add tsimport
#chkconfig --add tsexport
```

   - **CentOS7**:

```
#systemctl enable tsimport.service
#systemctl enable tsexport.service
```

3. You can now start and stop these programs with the commands:
   - **CentOS6**:

```
#service tsimport start
#service tsimport stop
```

   - **CentOS7**:

```
#systemctl start tsimport
#systemctl stop tsimport
```

D.4.4 Configuring Network Buffering for tsimport

For **tsimport** to successfully read the pulses of data from the network, you must enlarge the network read buffers.

1. Add the following commands to the end of the */etc/sysctl.conf* file:

```
net.core.rmem_default = 1000000
net.core.rmem_max = 4000000
```

2. Reboot the system.

D.4.5 Running tsimport and tsexport from the Command Line

You can run **tsimport** and **tsexport** from the command line using 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
```

D.5 TS Switch Utility

Only one of processes may write to the **TS API**. Use the **TS Switch** utility to select among the available sources.

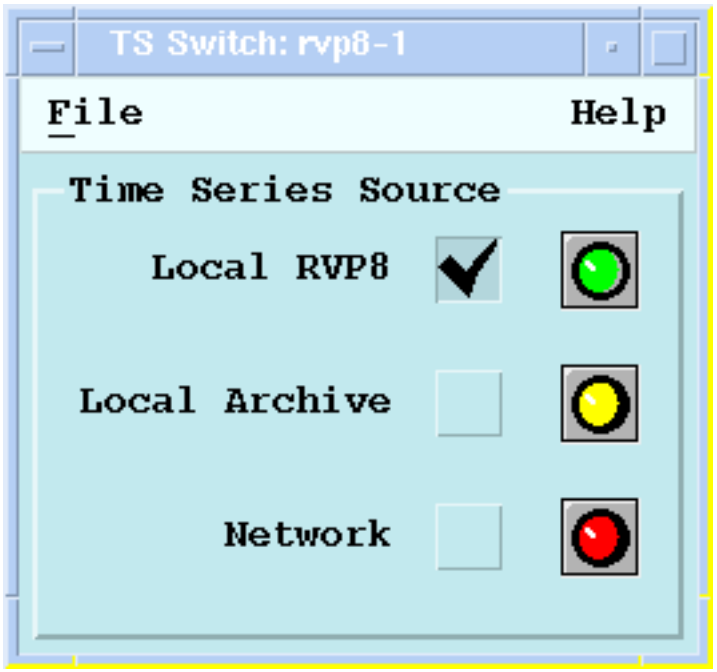


Figure 60 TS Switch Utility

Local RVP900 Real time IQ from the IFDR. This setting is available only on an RVP. Select this option for normal data processing, local RVP900 archive of real time `IQ` or to export real time `IQ` over the network.

Local Archive Used to extract time series from the local disk archive. On an RVP, these time series could be processed. On a separate archive host, these time series could be sent to an RVP or a custom user application for processing.

Network Used to collect time series from a networked RVP900 or archive host, using 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 always shows red, because there is not a local RVP.

1. To start the `tsswitch` utility as operator by typing: **`$tsswitch`**
2. Select the process.
   The **Local Archive** case and the **Network** case are possible for both an RVP and a separate archive host.

D.6 TS Archive Utility

Use the **TS Archive** utility to record and playback archived data.

The utility provides access to the **TS Switch** utility, playback and record modes, and a complete archive file inventory for managing disk space.

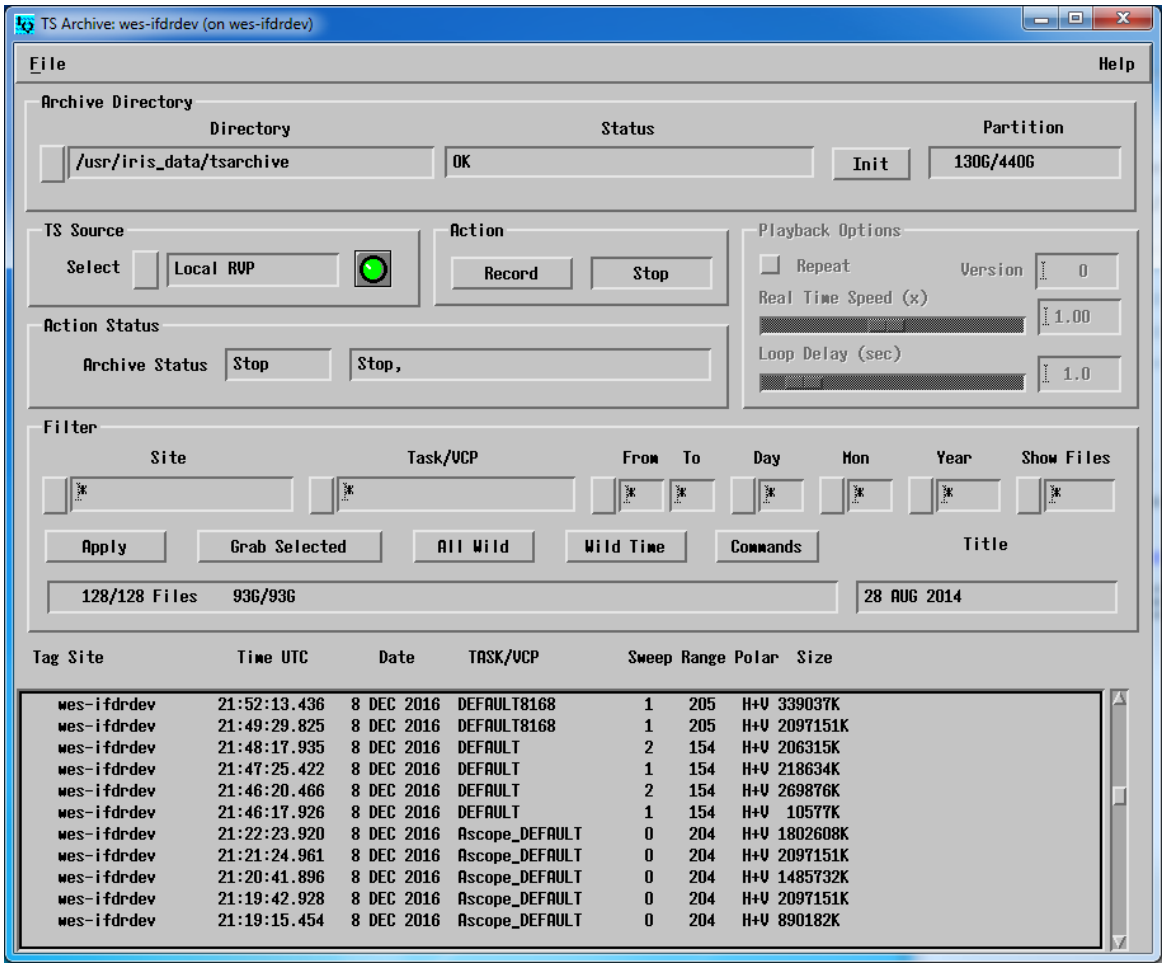


Figure 61 TS Archive Utility

1. Log in as operator.
2. Start the **TS Archive** utility by typing: **`$tsarchive`**

3. Under **Archive Directory**:
   - Use the **Directory** section to select which archive directory to use, or to add another directory.

     Vaisala recommends you first create the directory from the command prompt.

     ```
     $mkdir /bigdisk/tsarchive
     ```

   - Check **Status** to see information about the chosen archive directory.
   - Select **Init** to initialize the directory. You are prompted to type a title for the directory and to verify the contents of the directory.
     The directory must be empty or contain ONLY `tsarchive` files.
     If a directory contains `tsarchive` files, you are prompted for permission to delete them.

     For system security reasons, **Init** can only delete TS data files.
     If you must delete other, non-`tsarchive` files in the directory, you must use the UNIX **`rm`** command to manually delete them.

   - Check **Partition** to view 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 shows the used/total disk space for */bigdisk/*.
     The values displayed include all file types in the partition.
4. Select **TS Source** to launch the **TS Switch** utility and chose the source used to record data.
   See D.5 TS Switch Utility (page 391).
   You can now record or play back data.
   a. Select **Record** to start data recording. The inventory at the bottom of the screen updates with TS file information.
   b. If the **Local Archive** source is selected, you can select **Playback** to play previously recorded data.
      - Select **Repeat Time Series** to repeat the selected files.
      - Select **Real Time Speed** to control the speed at which the files are played back.
      - Select **Loop Delay** to control the length of the pause in between each repetition.
   c. Select **Action Status** to display the current mode of the utility and provide information on what data are being recorded or played back.
   d. Select **Stop** to terminate recording or playback.

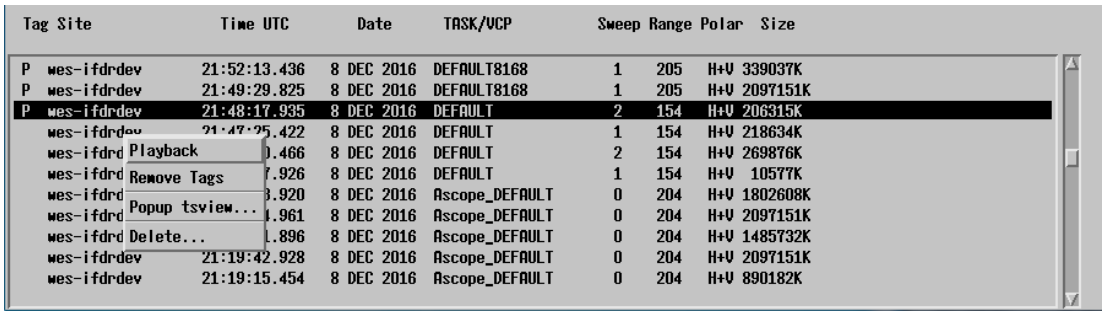
5. Use **Filter** area to manage files that are stored on a disk, including displaying data for a certain site, task, and time.

Option	Description
Site	Enter a site ID to select data from only one site, or enter the wildcard character to select data from all sites.
Task	A **Task** refers to either an IRIS task name (also 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 name. A question mark (?) matches any single character, an asterisk (*) matches any sequence of zero or more characters. For example, enter a task name of `*PPI*_?` to select all hybrid tasks with `PPI` somewhere in their names.
From, To	Enter or select a range of hours in these two fields to narrow the list of products by time of day.
Day, Month, Year	Enter or select a day, month. or year to narrow the list of products by date.
Show Files	Control how many files to include in the list.
Apply	Update the file list based on your selection criteria.
Grab	Select a product and then select **Grab** to insert a selected file information in the filter fields.
All Wild	Return all the fields to the wildcard character.
Wild Time	Change the hours, month, day, and year fields to the wildcard character.
Commands	Shows the following operations that you can perform on the `tsarchive` files selected from **Filter**: • **Playback**: Tags files for playback. A **P** appears in the left column. • **Delete**: Deletes files. A **D** appears in the left column. • **Remove Tags**: Untags any **Delete** or **Playback** tags.  **CAUTION!** The commands apply to all of the `tsarchive` files that match the filter. If you put in wildcards everywhere and give the**Delete** command, every TS archive file could be deleted.

6. View TS Archive logs in the **Log** area.

Archive Log Column	Description
Site	TS site used to create the data.
Time and **Date**	UTC time and date when the data were acquired.
Task/VCP	Name of the associated tasks or **Ascope** file used for controlling RVP.
Sweep	Number of sweeps (for example, 360°) in the Task/VCP.
Range and **Polar**	Range and polarization of the TS data.
Size	Size of the file.

7. Right-click a file or group of files in the **Log** area to show the following commands:



- **Playback**: Marks the files with a **P** in the **Tag** column. When selected, only the tagged files are played.
- **Delete**: Marks the files with a **D** in the **Tag** column, and deletes the files. This is helpful when you select a large list of files for deletion.
- **Remove Tags**: Clears the **D** or **P** tags.
- **Popup tsview**: Provides file information in another window.

D.7 Software Application Examples

RVP supports four uses of the time series application:

- TS recording on a local RVP900, see D.7.2 TS Recording on a Local RVP900 (page 398).
- TS recording on a separate archive host, see D.7.3 TS Recording on Separate Archive Host (page 398).
- TS playback on a local RVP900, see D.7.4 TS Playback on a Local RVP900 (page 400).
- TS playback from a separate archive host to an RVP900, see D.7.5 TS Playback from a Separate Archive Host to an RVP900 (page 401).

In the example descriptions, unused processes are omitted for clarity.

For reference, see D.7.1 RVP900 in Normal Real-Time Operation (page 397).

D.7.1 RVP900 in Normal Real-Time Operation

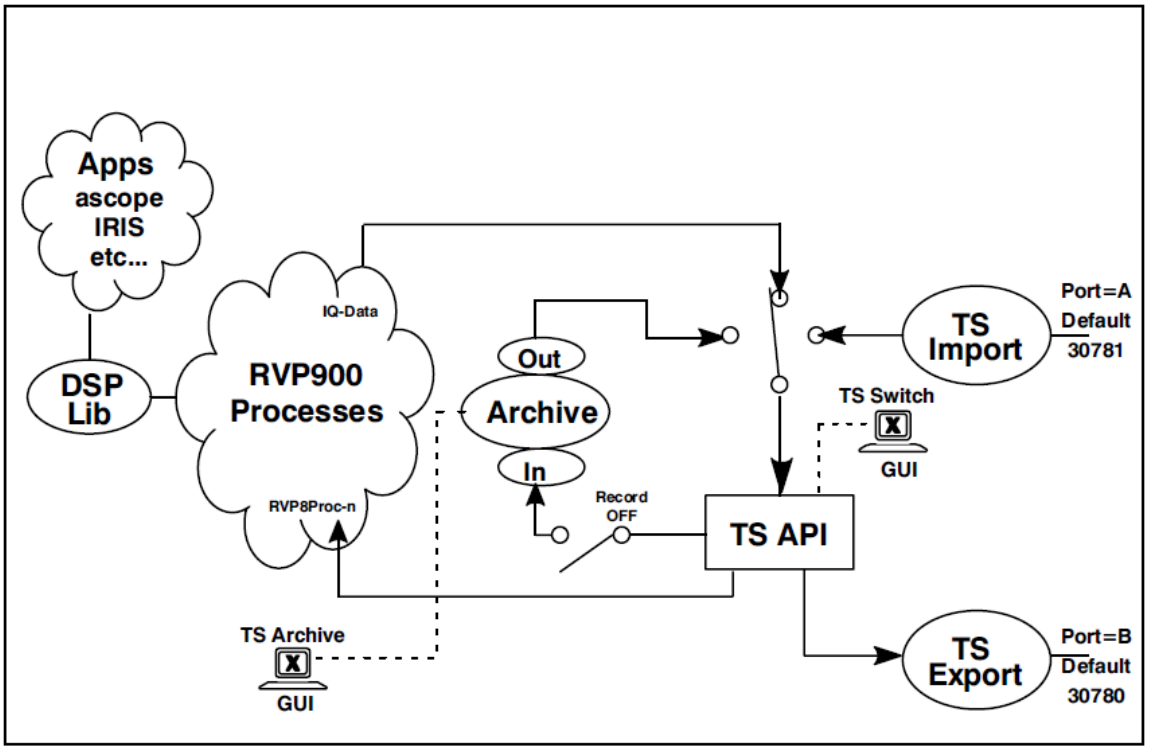


Figure 62 RVP900 in Normal Real-Time Operation

In this case, the **TS Switch** is set to write real-time data from the IQ-Data process to the `TS API`.

`RVP9Proc-n` extracts time series data and processes it. The configuration information is obtained from and data are passed to user applications through the `DSP Lib` functions. The `tsexport` process, if started, can extract data simultaneously from the `TS API`.

Utility Settings

- **TS Switch**: **Local RVP900**
- **TS Archive**: N/A

D.7.2 TS Recording on a Local RVP900

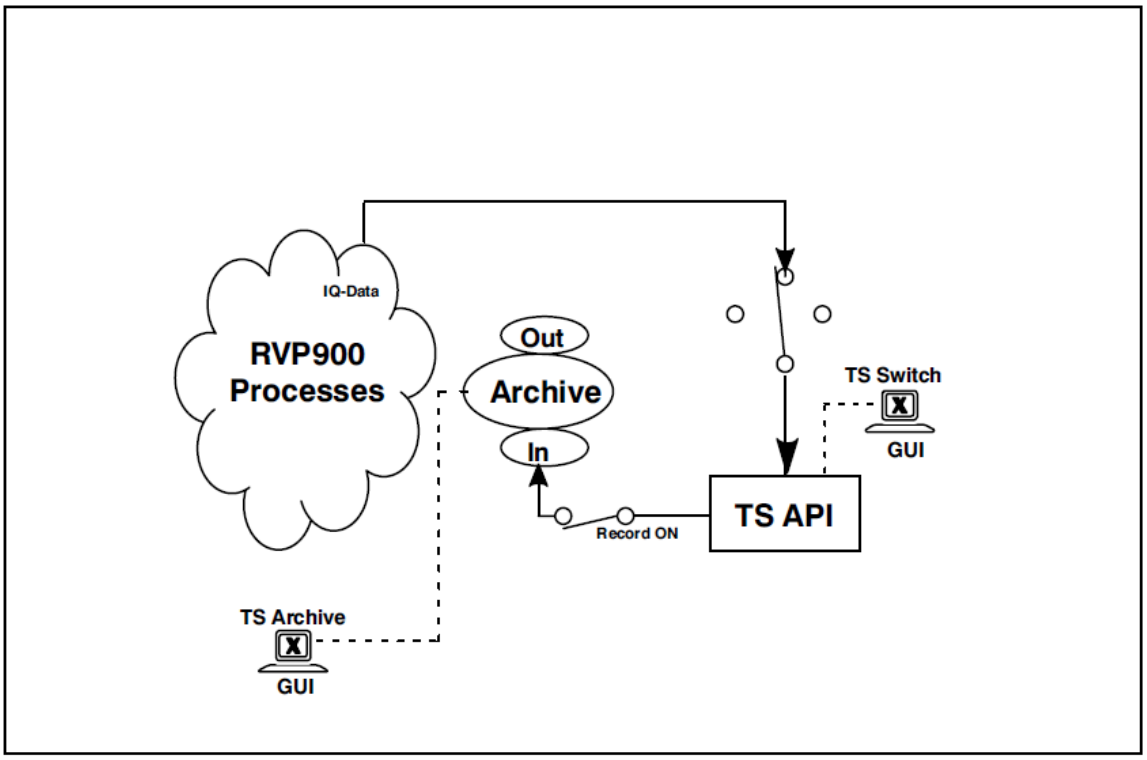


Figure 63 TS Record on Local RVP900

In this example, 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 `tsarchive` process.

While TS data are being recorded, RVP900 may still do its normal data processing tasks, as shown in D.7.1 RVP900 in Normal Real-Time Operation (page 397) .

Utility Settings

- **TS Switch**: **Local RVP900**
- **TS Archive**: **Record**

This configuration records to a local disk on RVP900. Recording to a dedicated data partition, NOT the "/" partition. This is because if the "/" partition fills-up, the system crashes.

If the separate data partition fills-up, then the system stops recording, but otherwise functions normally.

D.7.3 TS Recording on Separate Archive Host

This is the recommended recording configuration for TS recording.

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.

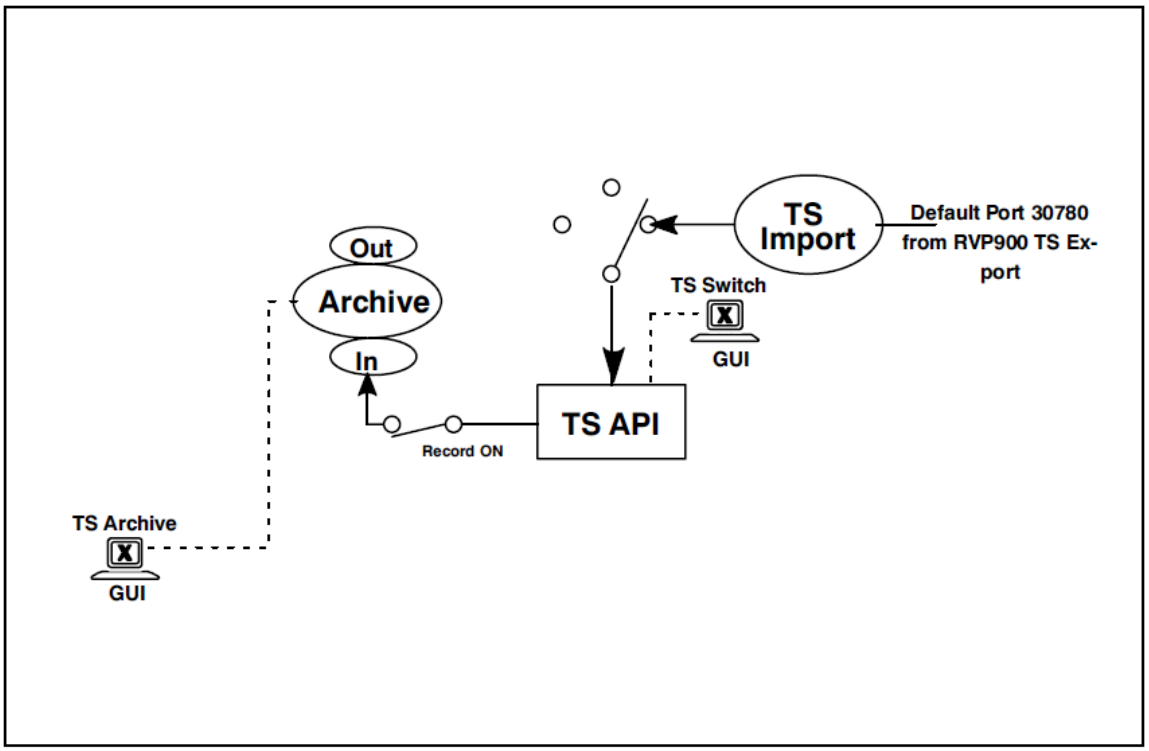


Figure 64 TS Record on Separate Archive Host

In this case, the **TS Switch** is set to write `IQ` data from the `tsimport` 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 `tsexport` process on a networked RVP900.

Utility Settings

RVP900:

- **TS Switch**: **Local RVP900**
- **TS Archive**: N/A

Archive Host:

- **TS Switch**: **Network**
- **TS Archive**: **Record**

D.7.4 TS Playback on a Local RVP900

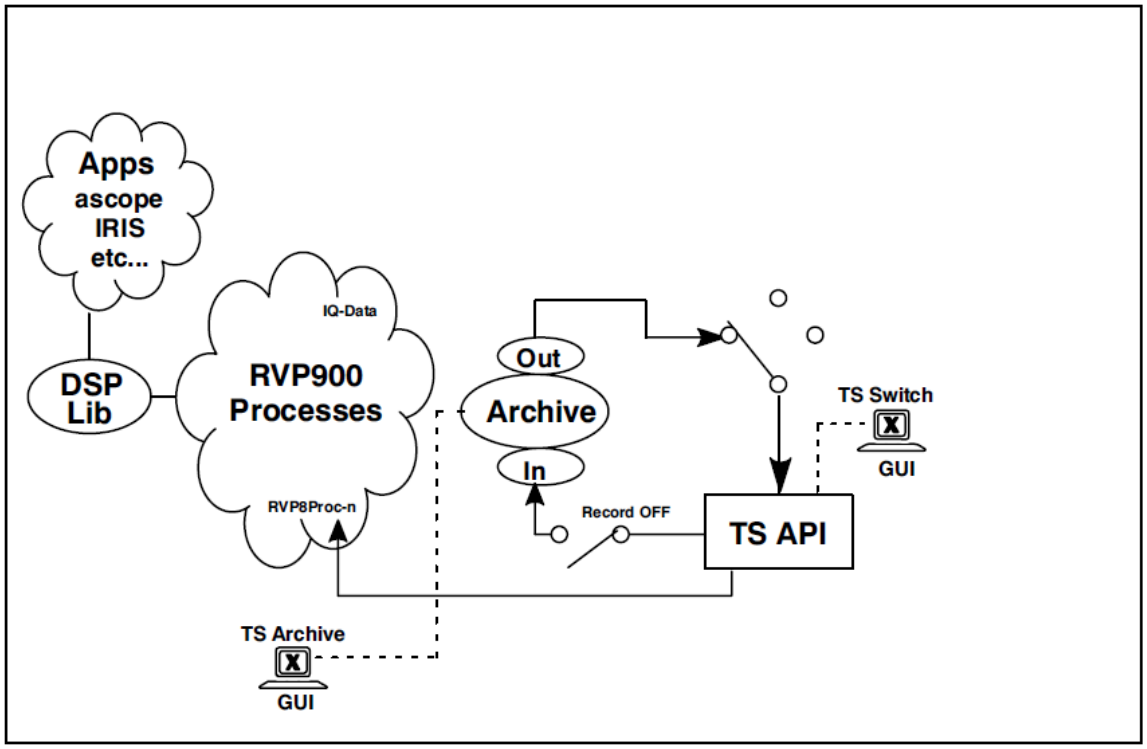


Figure 65 TS Playback on Local RVP900

In this example, the **TS Switch** is set to write data from the `tsarchive` to the `TS API`. `RVP9Proc-n` processes then reads the time series data from the API.

This is the same as the real-time normal operation case shown in D.7.1 RVP900 in Normal Real-Time Operation (page 397), except that the archive is the source rather than the real-time operation.

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 part of the time series data format.

Utility Settings

RVP900:

- **TS Switch**: **Local Archive**
- **TS Archive**: **Play**

D.7.5 TS Playback from a Separate Archive Host to an RVP900

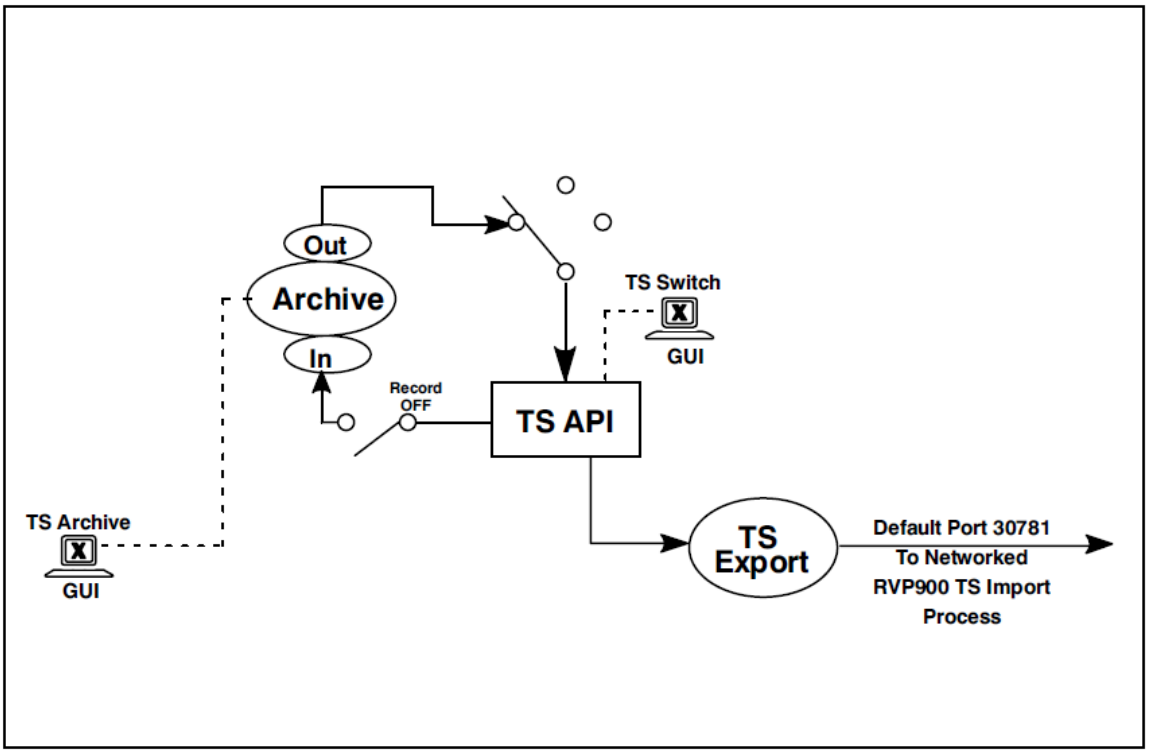


Figure 66 TS Playback from a Separate Archive Host

This is the recommended mode of operation.

In this example, the **TS Switch** is set to write data from the `tsarchive` to the `TS API`. `tsexport` sends these through a UDP broadcast over the network to an RVP900 for processing.

The diagram for the corresponding RVP900 would be identical to the one shown in this example, except the **TS Switch** would be set to write data from the `tsimport` process.

Utility Settings

RVP900:

- **TS Switch**: **Network**
- **TS Archive**: N/A

Archive Host:

- **TS Switch**: **Local Archive**
- **TS Archive**: **Play**

D.7.6 TS Archive Recording Quick Guide

1. Select a directory for the TS data to be written to.
   Make sure the desired directory already exists in the file system.
2. 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.
3. Select the data source and make sure the light is green.
4. Select **Record** and watch for data to arrive in the log section.

D.7.7 TS Archive Playback Quick Guide

1. Select the directory that contains the TS data to be played.
2. Under **TS Source**, select **Local Archive**.
3. Make sure the light is green.
4. Select the files that you wish to playback (right-click and select **Playback**) and mark them for playback.
5. Select any playback option.
6. Select **Playback**.

D.8 Ascope Playback Features

The **Ascope** utility is a stand-alone signal processor configuration and plotting utility. When an RVP900 is in playback mode, you can use **Ascope** to configure the processing of the playback data and display the results. See *IRIS and RDA Utilities Guide*.

The following figure shows the differences in the meanings of menu fields when RVP900 is in playback mode.

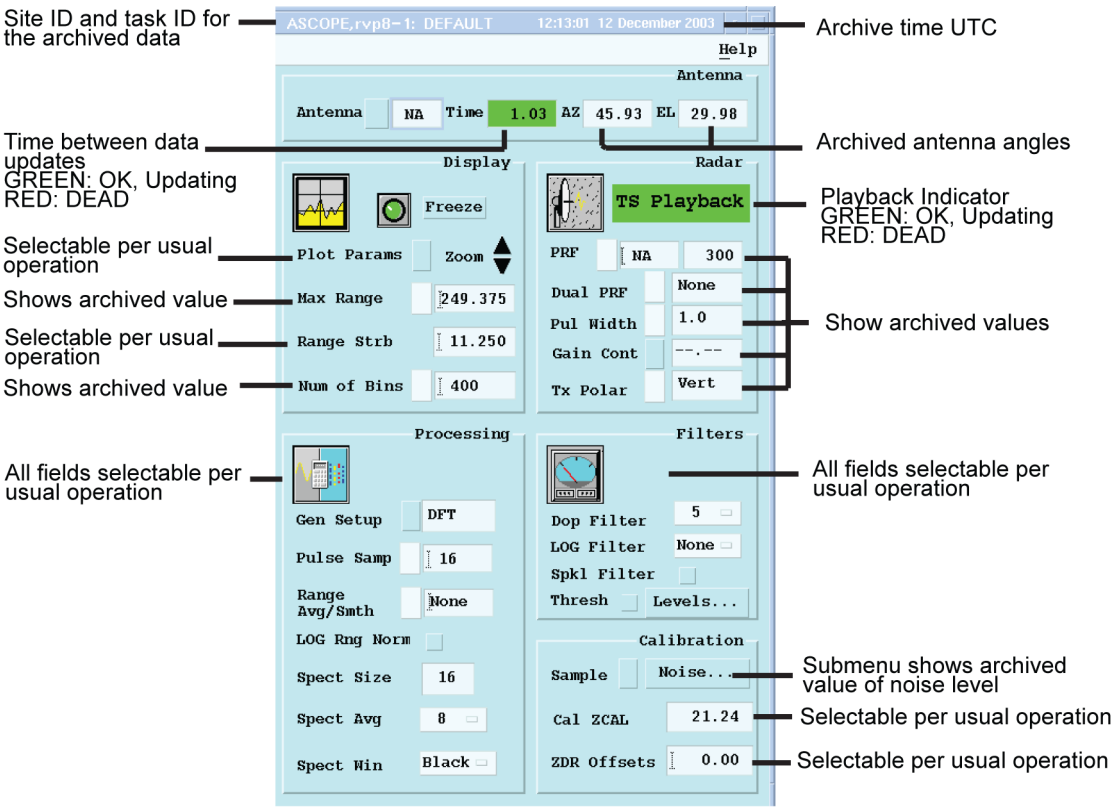


Figure 67 Ascope Differences during RVP900 TS Playback

When RVP900 is in playback mode, instead of 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

These paranters relate to the transmit characteristics of the radar and for these, the archived value is displayed. You cannot alter the fields.

There are other parameter fields that can be changed, just like when you use **Ascope** with real-time data, such as the parameters to configure processing and plotting. For example, you can change processing major mode and clutter filtering parameters while RVP900 is in playback mode.

When using **Ascope** during RVP900 playback, the archive can be on the local RVP900 or on a separate archive host.

It does not matter where **Ascope** runs. It can be on the local RVP900 or on a networked host computer through `DspExport`.

D.8.1 Archive on Local RVP900

Utility Settings

The support menus settings for archiving on a local RVP are:

- **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 by doing one of the following:

- 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 **`tsarchive`** with **Repeat** set, then start **Ascope**.

Ascope starts 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 files or restart the playback.

D.8.2 Archive on Separate Archive Host

The utility settings for archiving on a separate archive host are:

RVP900:

- **TS Switch**: **Network**
- **TS Archive**: N/A

Archive Host:

- **TS Switch**: **Local Archive**
- **TS Archive**: **Play**

D.9 TS Playback Using IRIS

Successful playback of time series data requires IRIS version 8.09 or later and you must make configure some settings in the IRIS user interface.

Table 112 Configuration Requirements for IRIS Playback

IRIS Location	Setting
Ingest > Signal Processing and Data Storage	**Source of recorded angles** = **RVP Tags** **Source of recorded time** = **RVP Tags**

IRIS Location	Setting
General > Modes and Protocols	**Timezone for data recording** = **UTC**
General > Scanning Options	**Task Scheduling Control** = **Active/Passive** or **Passive** **Passive type** = **TS–Playback**

When the configuration is complete, you are ready to use IRIS to play back data.

1. Launch `tsarchive` on your RVP900 machine.
2. Select **TS Source** to launch the `tsswitch`.
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.
   If you are playing back a multiple sweep volume scan, there is one file per sweep.
7. Use the **IRIS** menu bar to bring up the **Task Scheduler** menu.
   Make sure it says **Passive** on the top menu bar.
8. Select the task that matches the data.
9. Select **Go/Schedule** for that task.
   It toggles to **running**.
   Wait for data to arrive.
10. On the `tsarchive`, select **Playback**.
11. On the **Task Scheduler** menu, set the task to **Stop (when done)**, so that it stops after one time.

Because IRIS sorts stored data by time and playback version number, it is important to only playback the data once.

The **Ingest Summary** menu displays a 2-digit number containing the playback version after the site code.

Original data with a playback version of 0, does not include the number.

The ingest filenames have a **V** appended followed by the 2- digit playback number.

D.10 TS View Utility

Use the `tsview` utility to specify a TS file and obtain printout of header information and time series data on a terminal screen.

This is useful for checking how the data were collected, or to verify that **IQ** data were recorded.

Developers can use **tsview** software as a model when building their own off-line applications for reading and processing time series data.

If you have a licence for the **tsarchive**, right click a file TS inventory area to access the **tsview** utility.
See D.6 TS Archive Utility (page 393).

D.10.1 Starting tsview

You must have radarop privileges to run **tsview**. The **TS View** utility must be installed on the same node where the **IQ** data are archived.

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 options, see D.10.3 Tsview Command Line Options (page 407).

D.10.2 Starting tsview Sample Session

In this session, you view the pulse header information for the 101st pulse in a TS file. The archive directory is */bigdisk/tsarchive*:

```
//Change directory to location of TS files
$ cd /bigdisk/tsarchive

//List all files from 8 Dec 03 19:25:19
$ ls  *20031208.192519*

//One file found
RVP8.20031208.192519.074.Ascope_DEFAULT.0.H.249

//run tsview
$ tsview  RVP8.20031208.192519.074* -count:1 -skip:100 -v

//tsview output
Pulse Info size :1148
==============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
. . .
```

For information on the header format and information content, see D.11 TS Record Data Format (page 409).

For the file naming convention, see D.10.3 Tsview Command Line Options (page 407).

D.10.3 Tsview Command Line Options

-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)
```

pathname

This is the directory and name of the TS file.

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
```

The fields are as follows:

`site`
: Site name typed in the setup program on RVP900 that generated the data. `site` 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. `taskname` fields are preprocessed to remove characters that would mess up the filename, such as unprintable characters, space and `/`.

`sweep`
: A full 360° (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 is the transmit polarization. There are four choices: `H`, `V`, `H+V` (simultaneous H and V transmit) and `ALT` (alternating H and V transmit).

`-count:N`

Each file consists of the `IQ` data for all range bins for each pulse in the sweep.

Use `-count:N` to define how many pulses to display.

In the example in the **`-data`** section, the count is set to 1, that is, only the information for one pulse is displayed.

`-data`

Use the **`-data`** option to see the `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.

-length:N

The **-length:N** 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:1 00 -data
-length :43  Site:RVP8
Pulse #101 at :19:25:19.406 8 DEC 2003 UTC, Az: 9.99, El:29.98
0: (-91.2 ,244 ) (-91.2, 0) (-80.4,209)
3: (-87.4 ,149 ) (-82.2,150) (-79.9, 95)
```

-skip:N

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**.

-verbose

Use **-verbose** to see the detailed information contained in the headers. This was enabled in the example session. See D.10.2 Starting tsview Sample Session (page 406).

D.11 TS Record Data Format

Each TS file recorded to disk contains a run of 1 or more pulses, which are from the same basic RVP900 configuration. In RVP900 nomenclature, this is called the **Acquisition Mode** (stored in the **rvptsPulseInfo** structure). Each time something changes, 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 ASCII headers and binary data, shown in the following table. 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 113 TS File Format

File Component	Description
<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

File Component	Description
...	

Each time series sample consists of 2 floating point numbers representing the **I** and **Q** voltages. The values are full magnitude with a value of 1. This represents +8 dBm 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 8.8 Initiate Processing (PROC) (page 251).

The 16-bit words are 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:

$$\text{Power} = 6\text{dBm} + 10 \times \log_{10}[I^2 + Q^2]$$
$$\text{Angle} - \text{atan2}(Q, I)$$

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:

$$\text{Bytes} = 2 \times 2 \times \text{iNumVecs} \times \text{iVIQPerBin}$$

The number of samples can be different in each pulse in 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, see the following example (for more information, see the **rvpts.h** header file):

```
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           Pulse width 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. For 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 E. Serial Status Formats

RVP900 can optionally generate an internal **BITE** packet. Most of these bits are copies of data available from the **GPARM** command. Those bits are labelled with **GP** followed by the word number and bit number. See 8.10 Get Processor Parameters (GPARM) (page 267)

The identification byte is selectable, to avoid conflicts with other **BITE** packets. 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 114 Internal BITE Packet (RVP900 to Host)

Char	Function
1	**SYNC** Byte (**C0** Hex)
2	Identification byte (User Choice)
3	Diagnostic Results 0–6 D6 = GP11, D6 = 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
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>**

Char	Function
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

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 = Could 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 50–56 D6 = GP59, D7 = WSR88D Batch mode is supported D5 = GP59, D6 = Time series data source is external to RVP900 D4 = GP59, D5 = Trigger sequence truncated D3 = GP59, D4 = Using High-SNR packed (I,Q) format D2 = GP59, D3 = PRT altered to fit trigger pattern D1 = GP59, D2 = Trigger pattern altered to fit PRT D0 = GP59, D1 = **PROC** spectra size matches sample size

Char	Function
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 and #8 use Hi-SNR format D0 = GP59, D8 = Major mode refused to use external trigger
17	Immediate Status 64–70 D6 = <spare> D5 = <spare> D4 = <spare> D3 = <spare> D2 = <spare> D1 = <spare> 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>

Char	Function
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
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)

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.

Table 115 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

Appendix F. Softplane.conf

F.1 Configuring the *softplane.conf* File

For manual configuration, configure the ***softplane.conf*** file to define pin–by–pin assignment of I/O functions to connectors on the I/O-62 connector panel.

The file is a commented plain text ASCII file. Since the RVP and RCP8 have virtually no jumpers, or wirewrap, all I/O configuration on the I/O-62 connector panel is done by software approach according to this file.

The file is in the ***IRIS_CONFIG*** directory, typically ***/usr/sigmet/config*** (this is the default directory that is factory installed).

The factory configurations are also available in the ***/usr/sigmet/template/init*** directory so that you can always return to the factory defaults.

1. Login as **radarop**
2. Type: **$ cd /usr/sigmet/config**
3. Launch one of the text editors provided in the system:
   - **$ gedit softplane.conf**
     A user friendly editor with keyboard commands and mouse support when you are in X–Windows. It us a little easier to learn than **vi**.
   - **$ vi softplane.conf**
     The generic UNIX editor available on every UNIX system and familiar to many users.

F.2 *Softplane.conf* Organization and Syntax

The ***softplane.conf*** file defines the I/O pins on each connector on the PCI cards and on the connector panel. There are two primary definitions for each pin:

- Physical Interface — the electrical properties (RS422 output, analog input, TTL output, and so on).
- Logical Interface — the internal variable name that is associated with each pin.

The syntax of the file is:

- # at the beginning of a line indicates a comment.
  These are used for internal documentation. If you make changes, comment them, for example:

```
# TTL I/O on J7
 #
 # Modification by REP on 2 Apr 03
 # Added new interlock input on connector panel J7 pin07
 ...
```

- The top part of the file provides a list of internal variables names that are used to define the logical interface to the softplane.
  These are divided into status inputs (also called indicators) and control outputs (also called requests).
  For example, **sPedAZ0** corresponds to the LSB of a digital azimuth angle relative to the antenna pedestal. The following tables provide a summary of the available status and control variable names.

Important: This table is subject to change

- Each definition line in the file starts with the keyword text:

```
# splConfig...
```

- The first uncommented line of the file indicates the version of the IRIS support software that was last used to machine–generate the file.
  This is an information-only field for traceability purposes and is not edited. For example:

```
# splConfig.sVersion = "7.32"
```

On the TTL connectors (J1, J2, J4, J5, J7), each connector must be exclusively used for **INPUT** (s vars) or **OUTPUT** (c vars). You cannot mix these on an individual connector.

Summary of *softplane.conf* Status and Control Bits	**Description**
Control Output	Meaning/Interpretation
cPedAZ[15:0]	16 bits of antenna azimuth angle relative to the pedestal (fixed base system)
cPedEL[15:0]	16 bits of antenna elevation angle relative to the pedestal (fixed base system)
cEarthAZ[15:0]	16 bits of antenna azimuth angle relative to the earth (moving platform)
cEarthEL[15:0]	16 bits of antenna elevation angle relative to the earth (moving platform)
cServoPwr	To control servo power on
cCabinetRelay	To control a relay signal
cTransmitPwr	Request transmit power on
cPWidth[3:0]	Request one of four pulse widths
cTrigBlank	Trigger blanking signal
cRadiateOn	Request radiate on
cRadiateOff	Request for radiate off
cReset	Request a reset of external equipment
cIrisMode[2:0]	Request the application software (e.g., IRIS) to switch to 1 of 8 operating modes.
cAux[63:0]	Arbitrarily assigned output requests
true	Internal logic variable
false	Internal logic variable
sPedAZ[15:0]	16 bits of antenna azimuth angle relative to the pedestal (fixed base system)
sPedEL[15:0]	16 bits of antenna elevation angle relative to the pedestal (fixed base system)
sServoPwr	Servo power on indicator

Summary of *softplane.conf* Status and Control Bits	Description
sLocal	Antenna local mode indicator, usually tied to an external local/remote switch.
sStandby	Radar ready to radiate indicator
sLowerEL	Lower limit switch indicator
sUpperEL	Upper limit switch indicator
sTransmitPwr	Transmitter cabinet power on indicator
sTransmitLocal	Transmitter local mode indicator, usually tied to an external local/ remote switch.
sPWidth[3:0]	Indicator of the current pulse width
sTrigBlank	Indicator that trigger blanking is requested, usually from an external source.
sRadiate	Radiate on indicator
sAirflowFlt	Cooling airflow fault indicator
sWavegpFlt	Wave guide pressure fault indicator
sInterlockFlt	Master interlock fault indicator
sMagCurrentFlt	Transmitter overload fault indicator
sReset	Request for reset coming from external source
sIrisMode[2:0]	Information on which operating mode is active in the application software
sAux[319:0]	Arbitrary status indicators

- Each piece of hardware is identified as either in use or not in use.

```
splConfig.Io62[0].InUse = 1 if in use
     splConfig.Io62[0].InUse = 0 if unused or not installed
```

- The I/O-62 is the only I/O device supported by the softplane.

- Specify the method of connecting to the I/O-62, for example:

```
splConfig.Io62[0].sExtPanel = "DIRECT"
```

The options are:

Connection Type	softplane Descriptor
Direct connect to I/O-62 via 62 pin connector	`DIRECT`
I/O-62 Connector Panel (used for RVP8 and RCP8)	`IO62CP`
WSR88D connector panel, RVP8 portion	`RVP88D`
WSR88D connector panel, RCP8 portion	`RCP88D`

- The assignments for each connector and each pin are then made.
For convenience, these are usually grouped together by connector.
For example if, Pin 1 of connector J1 on the I/O-62 connector panel is assigned to be the LSB of the input azimuth angle, then:

```
# TTL/CMOS on J1
 #
 splConfig.Io62[0].Opt.Cp.J1.pin01 = "sPedAZ[0]"
 ...
```

- The notation `""` indicates that no assignment is made.

```
# BNC testpoint
monitors#splConfig.Io62[0].Opt.Cp.J13_BNC = ""
```

In the example above, the pin name is `J13_BNC`.

- Put a ~ in front of a logic variable to invert the variable.

```
splConfig.Io62[0].Opt.Cp.J1.pin03 = "~sPedAZ[2]"
```

Check the */usr/sigmet/config_template/init* directory for other examples of softplane configurations.

F.3 Testing, Backup, and Calibration

After software installation and before calibration, check system performance and check for installation errors.

If DSP calibrations are off, radar data may be unavailable.

For more information on testing and backing-up the system components, see 1.2 Related Documents (page 15).

Appendix G. RCP902 WSR98D Panel

G.1 RCP902 WSR98D Panel Regulatory Compliances

Table 116 RCP902 WSR98D Panel Regulatory Compliances

Listing	Standard
Electrical Safety	
CE Mark	EN60950-1
EMC	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

G.2 RCP902 WSR98D Panel Architecture

RVP902 server connects to the RVP900 IFDR, which connects to the RCP902 WSR98D panel.

The RCP902 WSR98D panel connects to the radar. A separate external power supply connects to RCP902 WSR98D panel.

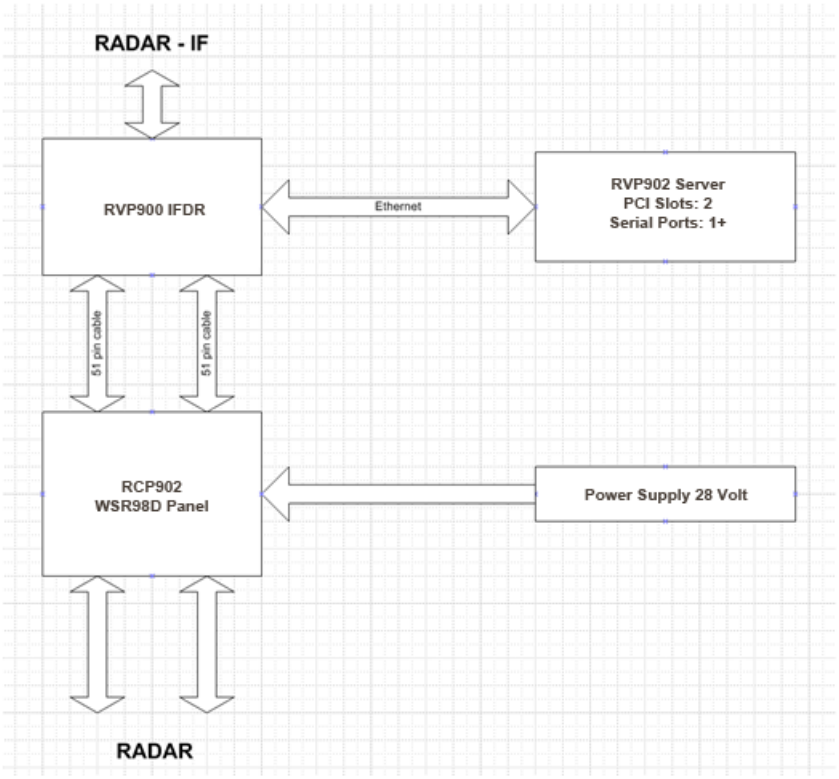


Figure 68 RCP902 Architecture

RVP902 software allows you to set and configure the RCP902 WSR98D panel (with IRIS8.13.1 or higher) for specific radar needs.

The RCP902 WSR98D connects to WSR98D radar interfaces.

More Information

- Regulatory Compliances (page 18)

G.3 DC Power Conditions - RCP902 WSR98D

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 ... +55 °C (-4 °F ... 131 °F)

- Inorder 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 thepanel.
  - Use a power cord with a fuse holder that can be opened without atool. 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°/circumferential electrical contactwith 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 must 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.

G.4 RVP902 WSR 98D Dimensions

The mounting dimensions of the RVP902 WSR98D panel are 4U 19 in EIA rack mountable.

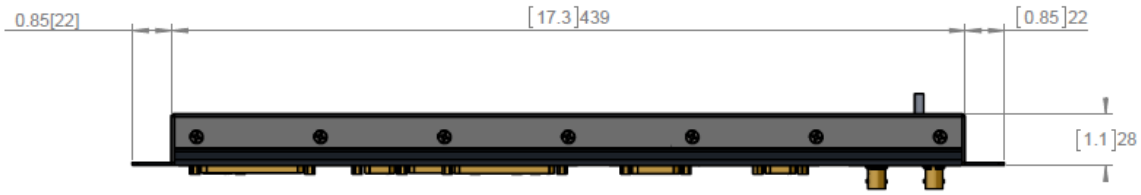


Figure 69 RCP902 WSR98D Top Panel Dimensions

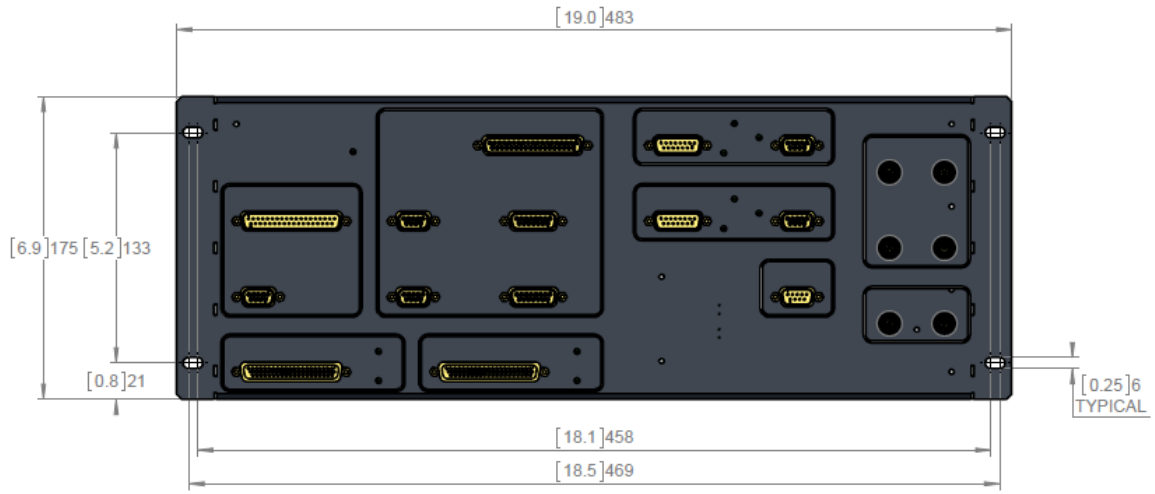


Figure 70 RCP902 WSR98D Back Panel Dimensions

G.5 RCP902 WSR98D Connector Locations

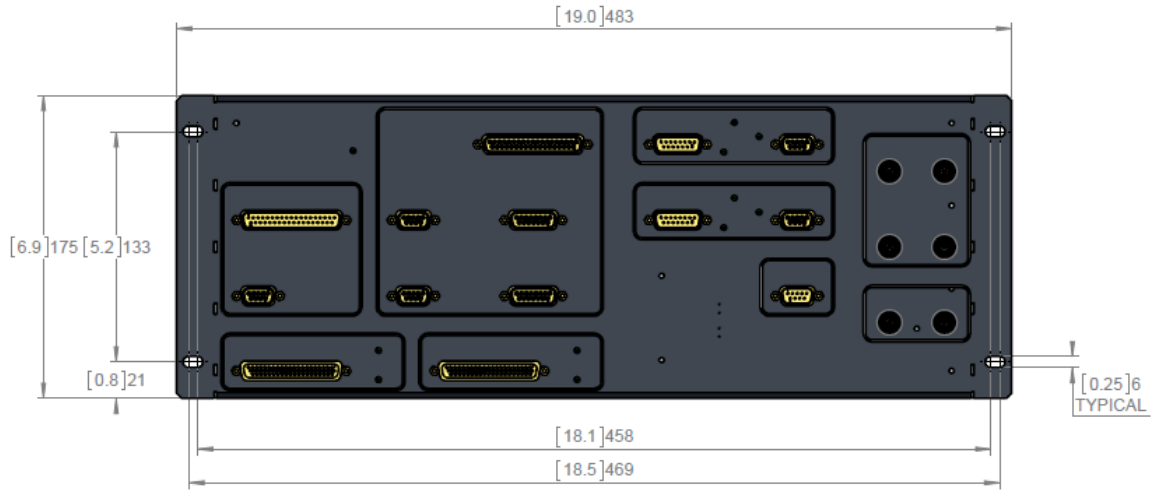


Figure 71 CP902 WSR98D Connectors

G.6 RCP902 WSR98D Panel Modifications

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
- J18 connector has been added for external power
- All the connectors on the back side (*-B) have been removed

G.7 RCP902 WSR98D Electrical Interfaces

There are several signals classes for the RCP902 WSR98D panel. The signal description formats identify relational information.

- Signals have pull-up /down depending on initial state and to maintain level during boot conditions
- RS-485signals:
  - 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 50W, 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 ... 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 accepts ±3V input low frequencysignal.

Table 117 RCP902 WSR98D Interface to Radar

Connector Size	Designator	TYPE	Connector Size	Designator	TYPE
D-SUB STR 37 POSITION	J3	PLUG	BNC 50 Ω	J20	JACK
D-SUB STR 9 POSITION	J4	RCPT	BNC 50 Ω	J21	JACK
D-SUB STR 37 POSITION	J7	RCPT	BNC 50 Ω	J22	JACK
D-SUB STR 9 POSITION	J8	RCPT	BNC 50 Ω	J23	JACK
D-SUB STR 15 POSITION	J9	RCPT	BNC 50 Ω	J26	JACK
D-SUB STR 9 POSITION	J10	RCPT	BNC 50 Ω	J27	JACK

Connector Size	Designator	TYPE	Connector Size	Designator	TYPE
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 118 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

- 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 Ω or greater terminated line.
- All RS-422 inputs are terminated with a 100 Ω 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°/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.

G.7.1 J3 - Transmitter Triggers (Tx TRIGS)

Table 119 J3 - 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
26	MOD_DISCHARGE_TRG_N	L
27	NC	
28	SHORT_BEAM_PULSE_SEL_N	L
29	SHORT_RF_PULSE_SEL_N	L

Signal	Connections	Initial Condition
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	

G.7.2 J4 - Receiver Protector (Rx PROT)

Table 120 J4 - Rx PROT, Signal Type: DB09S

Signal	Connections	Initial Condition
1	VCC3_OUT[1)]	
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	

1) *VCC3_OUT provides 200 mA of current from the 5 V supply on the panel.*

G.7.3 J7 - RF Generator (RF-GEN)

RF−GEN proivdes RF generator control, status, and phase modulation.

Table 121 J7 - RF-GEN, Connector Type: DB37S

Signal	Connections	Initial Condition
1	NC	
2	NC	
3	NC	

Signal	Connections	Initial Condition
4	CHANFAIL_N	
5	CHANFAIL_P	
6	PHASEBIT_N4	H
7	PHASEBIT_P4	L
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	

Signal	Connections	Initial Condition
37	RFGATE_N	H

G.7.4 J8 - RF Test Selection (RF-TEST SEL)

RF−TEST SEL controls the four position diode switch for the RF test signal selection.

Table 122 J7 - RF-GEN, Connector Type: DB09S

Signal	Connections	Initial Condition
1	DRSIGPOS_P3	L
2	DRSIGPOS_N2	H
3	DRSIGPOS_P2	L
4	DRSIGPOS_N1	H
5	DRSIGPOS_P1	L
6	DRSIGPOS_N3	H
7	DRSIGPOS_N4	H
8	DRSIGPOS_P4	L
9	GND	

G.7.5 J9 - Attenuator Control (ATTEN)

ATTEN controls the 7-bit attenuator.

Table 123 J9 - ATTEN, Connector Type: DB15S

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

Signal	Connections	Initial Condition
11	DRIVESIG40DB_P	H
12	DRIVESIG32DB_N	L
13	DRIVESIG32DB_P	H
14	DRIVESIG16DB_N	L
15	DRIVESIG16DB_P	H

G.7.6 J10 - Noise Source (NOISE SRC)

NOISE SRC controls the noise source and RF test signal injection point.

Table 124 J10 - NOISE SRC, Connector Type: DB09S

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	
9	NC	

G.7.7 J11 - RF Test Switch (RF-TEST SW)

RF-TEST SW controls the 10-position RF test switch.

Table 125 J10 - NOISE SRC, Connector Type: DB15S

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

Signal	Connections	Initial Condition
7	DRSIGBIT_N1	H
8	DRSIGBIT_P1	L
9	NC	
10	NC	
11	NC	
12	GND	
13	GND	
14	NC	
15	NC	

G.7.8 J12 - DAU Serial I/O (SERIAL-IN)

SERIAL-IN is the DAU serial line from the 98D.

Table 126 J10 - NOISE SRC, Connector Type: DB15P

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	
13	NC	
14	DAU_RX	
15	GND	

G.7.9 J14 - DCU Serial I/O (SERIAL-IN)

SERIAL-IN is the DCU serial line from the WSR98D.

Table 127 J10 - NOISE SRC, Connector Type: DB15P

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	

G.7.10 J18 - Panel Power Input (+28 V POWER)

Connector type: DB09P

+28 V POWER is the nominal 28 V Input Supply. Expected power is less than 15 W

NC on J18 pin 3, per customer request, one pin left un-connected.

G.7.11 J20, J21, J22, J23 - RVP901 Digital Test Points

Connector type: 50 Ω BNC

Programmable test point outputs from RVP900 can drive a 5 V, 50 Ω 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.conf* file.

For reasonable signal integrity, a 50 Ω load should be provided or a connector should be probed directly with an active high-impedance probe.

G.7.12 J26 - LOG Video Input (RF TEST-IN)

`RF TEST-IN` is the baseband detected pulse.

Table 128 J26 - RF TEST-IN, Connector Type: 50Ω BNC

Value	Description
Analog input range	0.5 V ... 2.5 V, low frequency/DC
Input Impedance	90 Ω
Settling time	Input has a settling time of 7 µsec
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.

G.7.13 J27 - Spare Analog Input (SPARE)

SPARE is the baseband detected pulse.

Table 129 J27 - SPARE, Connector Type: 50Ω BNC

Value	Description
Analog input range	+/- 3 V, low frequency/DC
Input Impedance	100 Ω
Settling time	Input has a settling time of 500 µsec
Gain	1 V/V
Circuit description	Circuit 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.

G.7.14 COAX

Table 130 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

G.8 RVP900 Interface to the RCP902 WSR98D Panel

The RCP902 WSR98D panel has two connectors, which are defined by connection to RVP900.

Table 131 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	+5 VDC
−5V	-5 VDC
GND	Ground

For more information, see drawing DRW240510.

G.8.1 RCP902 WSR98D I and 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.

The folllowing tables 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 with 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. Consider using additional shielding between wire pairs, if feasible.

Table 132 RCP902 WSR98D Interconnect Cable for Miscellaneous 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

IFDR J6 - 51 Pin	IFDR Signal Name	WSR98D J1 - 62 Pin	WSR98D Signal Name
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
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 133 RCP902 WSR98D Interconnect Cable for Miscellaneous 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

IFDR J3 - 51 Pin	IFDR Signal Name	WSR98D J2 - 62 Pin	WSR98D Signal Name
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

G.9 WSR98D Software Control and Status

Many 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 RVP900 or RCP processes.

If controlled by the RCP process, they are mapped to logical variables in the *softplane.conf* file.

G.9.1 Logical Variables

For logic mappings to be valid, the following settings must be defined in the *softplane.conf* file.

Table 134 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	

Signal Name	Controlling Process	Programmable Logic Variable	Conditions
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
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	RVP / RCP	cAux[49], trigger 11	cntPhaseForce = "true", "false"

Signal Name	Controlling Process	Programmable Logic Variable	Conditions
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			

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 5 V power on panel has reached acceptable levels.
J1 OK			When on 5 V IFDR signal pin is mated on J1
J2 OK			When on 5 V 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 correct 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]	

Signal Name	Controlling Process	Programmable Logic Variable	Conditions
`Rx Protect` status	RCP	`cAux[30]`	

G.9.2 Monitoring Analog Inputs

Hook in the WSR98D Set IO RPC to enable sampling of the analog inputs.

A sample code shows 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 µsec.

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, set the channel number to 0.

To monitor the Spare input, set the channel number to 4, and set the 5 V supply to 3.

Appendix H. RCP903 ASR9-WSP Panel

H.1 ASR9-WSP with RCP903 ASR9-WSP Panel Overview

The ASR9 WSP(weather signal processor) uses RVP900 hardware and software as well as the RVP902-WSP processor, RCP903 ASR9 panel, and RVP901-WSP signal processor.

The ASR9 WSP (weather signal processor) is designed to provide a direct replacement to the orignal RxNet7/RVP7 implemtnation carrying forward form, fit, and function of the original system wherever physically and logically possible.

More Information

‣ Regulatory Compliances (page 18)

H.2 RCP903 ASR9-WSP Panel Regulatory Compliances

Table 135 RCP903 ASR9-WSP Panel 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

H.3 Power Conditions for Use - RCP903 ASR9-WSP

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: -40 °C ... 45 °C 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: 10 °C ... 35 °C
- 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: -20 °C ... 55 °C. Unit has shown to be functionally operation down to -20 °C. 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.

H.4 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. A block diagram of the original system is shown in the following figure.

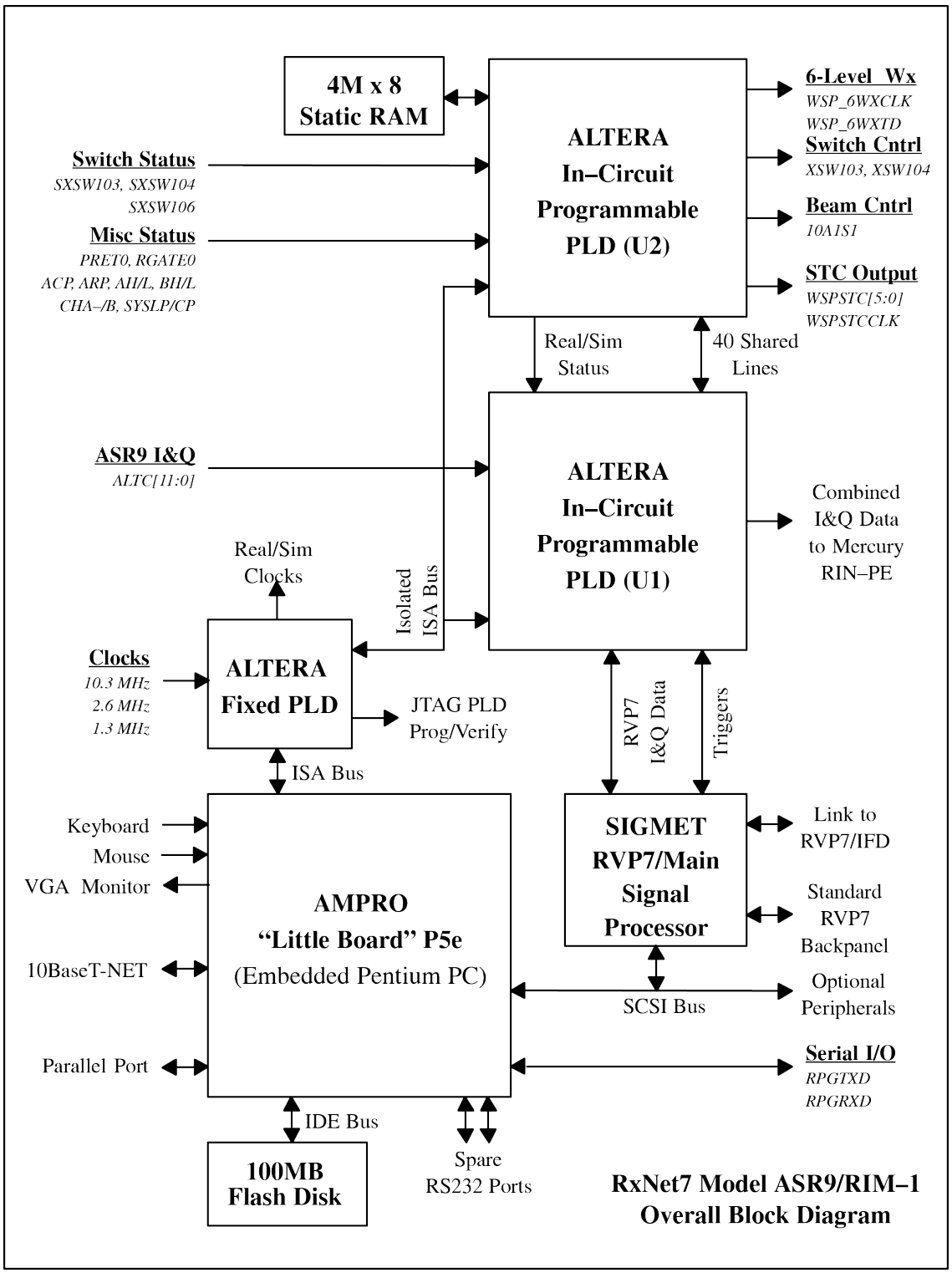


Figure 72 ASR9 WSP with RVP7 Architecture

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 block diagram for the new RVP900 base ASR9 WSP solution is shown in the following figure and a table with the RVP900-WSP components is shown in the following table.

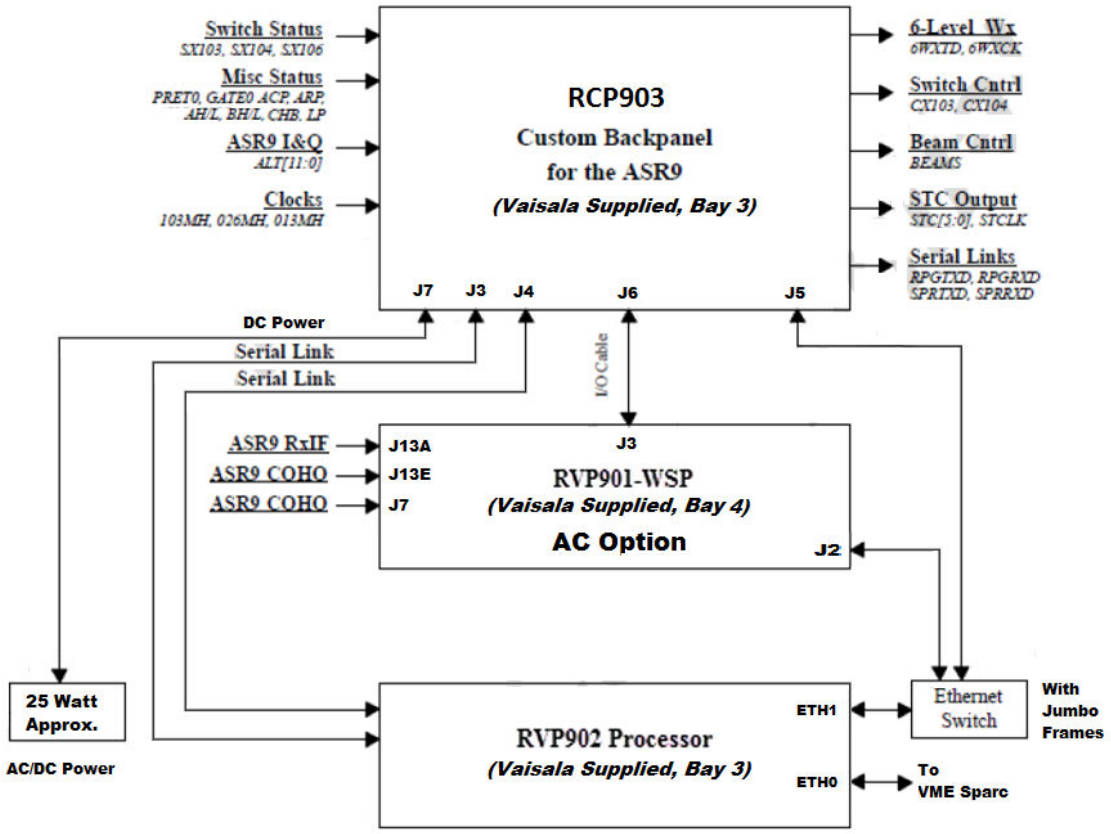


Figure 73 ASR9 WSP with RVP900 Architecture

Table 136 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

H.4.1 RVP901-WSP Signal Processor Customized for ASR9 WSP

The RVP901-WSP signal processor is Vaisala's standard product offering. When used with ASR9 WSP, it is configured as follows.

Table 137 RVP901-WSP Customizations for ASR9 WSP

Feature	Description
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	

H.4.2 RVP902-WSP Processor Customized for ASR9 WSP

RVP902-WSP Processor is Vaisala's standard product offering with a Xeon X8DTU motherboard running CentOS 6.4. RVP900 Radar utilities. The 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 ... 35 °C operating temperature

H.4.3 RCP903 ASR9-WSP Custom Panel

The main features of RCP903 are as follows:

- CPLD and FLASH memory to configure the panel and load the application
- Altera Cyclone IV FPGA and 16 MB of SRAM

- 100/1000 Ethernet 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 (1 A max current)
- 8 Indicator LEDs
  - Backward compatible 2x 6-Level weather indicators and 4x test LEDs
  - 2x RCP903 status indicators

H.5 RCP903 ASR9-WSP Panel Physical Interfaces

H.5.1 RCP903 ASR9-WSP Panel Dimensions

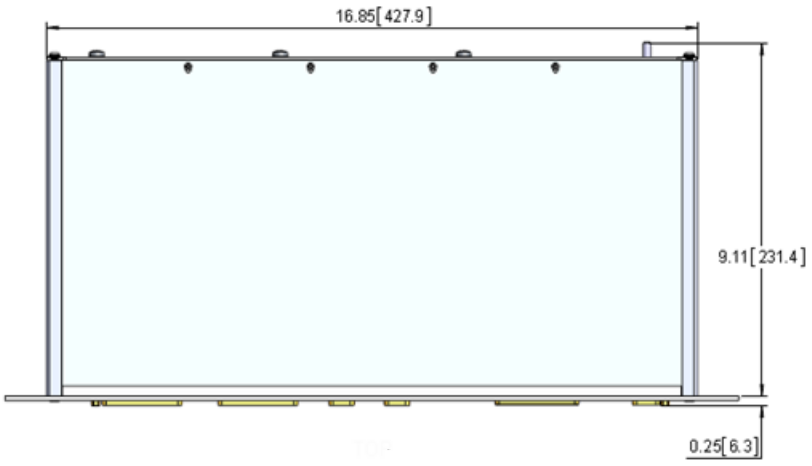


Figure 74 Top Panel Dimensions

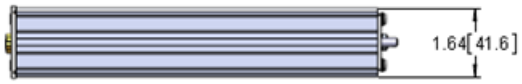


Figure 75 Side Panel Dimensions

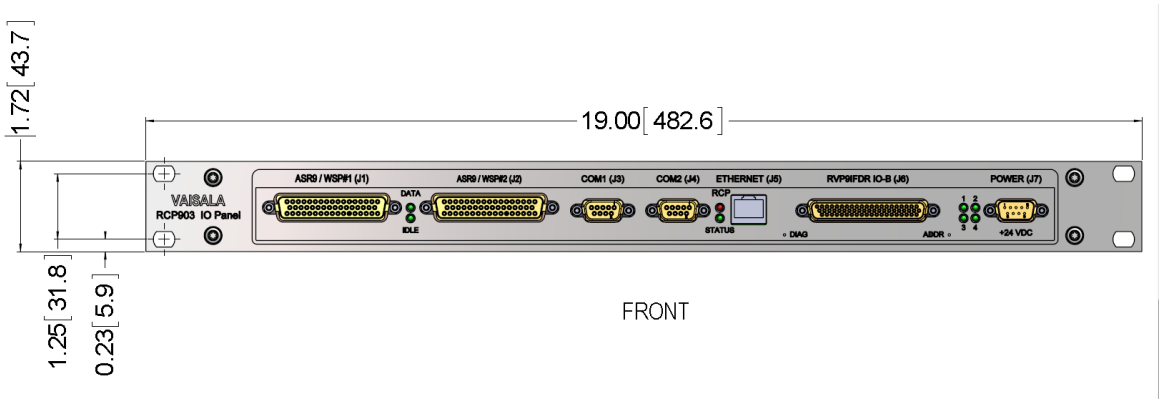


Figure 76 RCP903 Front Panel Dimensions

Figure 77 RCP903 Back Panel Dimensions

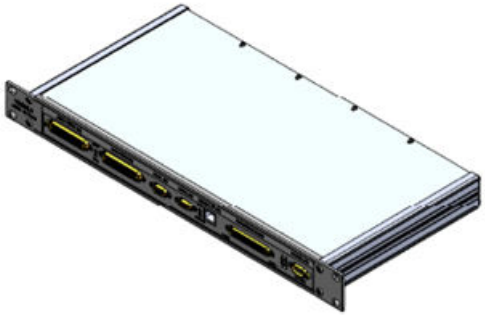


Figure 78 RCP902 Panel Perspective View

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.

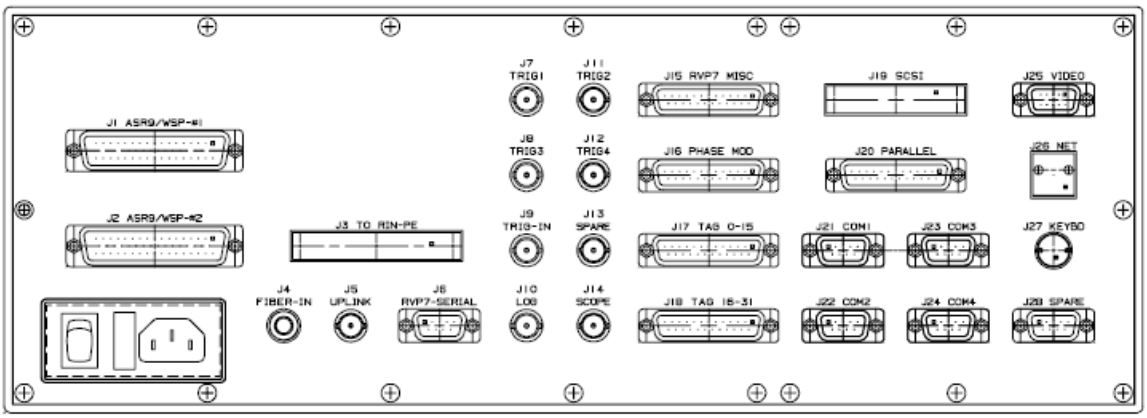


Figure 79 RxNet7 Front Panel

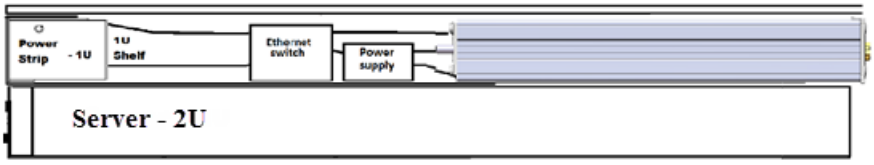
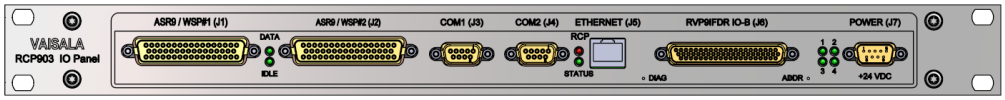


Figure 80 RCP903 Rack Side Perspective View

H.5.2 ASR9-WSP Connector Locations

All the connectors on the RCP903 ASR9-WSP Panel that interface with the radar are on the front of the mounting position.



FRONT

Figure 81 RCP903 ASR9-WSP Panel

H.5.3 RCP903 Shelf

Figure 82 RCP903 Shelf

This is intended to mount from the rear of the rack.

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

H.6 RCP903 ASR9-WSP Electrical Interfaces

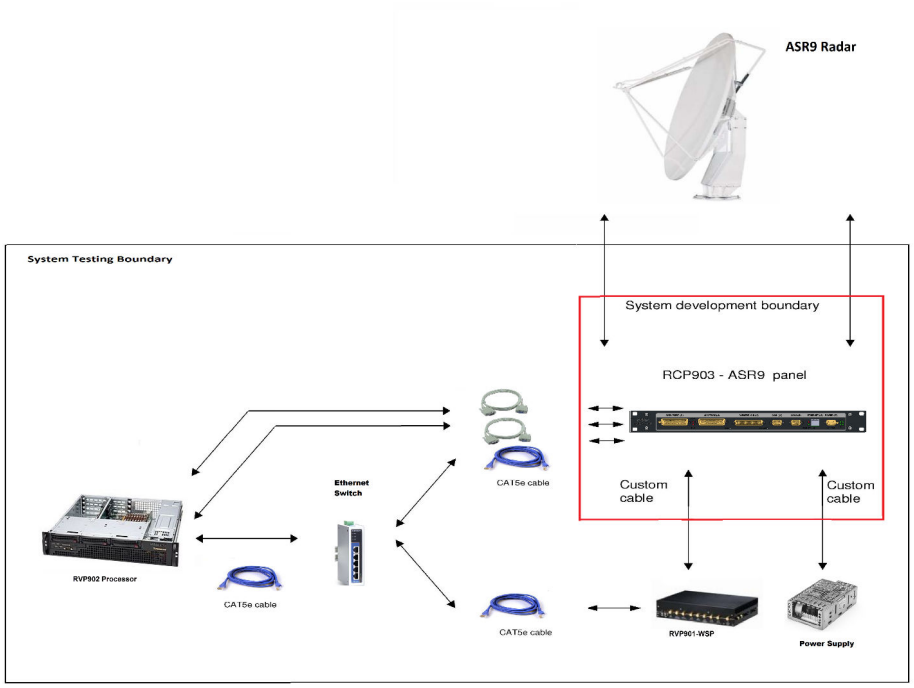


Figure 83 System Testing Boundary

H.6.1 RCP903 ASR9-WSP Interconnect Cabling

There are several classes of signals for the RCP903 ASR9-WSP Panel. The signal descriptions format identifies relational information.

- RVP902-WSP Processor attaches to the Ethernet switch using:
  - 2 - 242525 DB9 M to DB9 F Null Modem Reverser Cable
  - 1 - 220525 Ethernet Cable (Ethernet CAT5E)
- The RCP903-ASR9 WSP Interface Panel attaches to the Ethernet switch using:
  - 1 - 220525 Ethernet Cable (Ethernet CAT5E).
- RVP901-WSP attaches to the Ethernet switch using:
  - 1 - 231167 Ethernet Cable (Ethernet CAT5E).
- 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 RVP901-WSP. This interconnect cable has 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, are 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 is provided by Vaisala.

220525 Ethernet Cable is a CAT5E cable to connect to the Panel through a network switch or directly from the RVP902-WSP processor.

231167 Ethernet Cable is a CAT5E cable to connect to RVP901-WSP.

H.6.2 RVP901-WSP to ASR9 Radar

Table 138 RVP901-WSP Interface to ASR9 Radar

Signal	RVP901-WSP Connector
COHO	J7
COHO	J13E
IF	J13A (31.07 MHz)

RVP901-WSP is configured with the AC Option and with 2 - 30 MHz filters specified.

Table 139 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
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

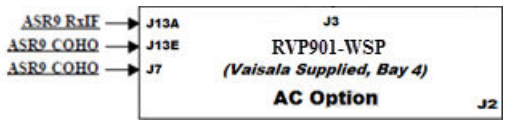


Figure 84 Vaisala Supplied, Bay 4

H.6.3 RCP903 ASR9-WSP Panel Interfaces

Table 140 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

Connector Size	Designator	Type	Destination
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)

H.6.4 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 141 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, forces the panel into boot mode.
ADDR	Between J6 and J7	Momentary Push Button	When pressed, forces the IP address of the panel to its default value, 10.0.1.253

H.6.5 J1 - ASR9 Interface WSP #1

Table 142 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_PRG_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	Switch #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

Pin Number	Ribbon Cable Number	Signal Name	Type	Direction	Description
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

H.6.6 J2 ASR9 Interface WSP #2

Table 143 Pin-out for J2 ASR9 / WSP #2

Pin Number	Ribbon Cable Number	Signal Name	Type	Direction	Description
1,34	1,2	WP_PRET0_P/N	RS-422	Input	Pretrigger 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

Pin Number	Ribbon Cable Number	Signal Name	Type	Direction	Description
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

H.6.7 J3 and J4 RS-232 Interfaces to RVP902-WSP Processor

Table 144 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	

H.6.8 J5 - Ethernet Interface

RJ45 Ethernet Interface provides 100/1000 Base-T communication with RVP902-WSP Processor direct or through a network switch. This interface requires Jumbo Frames. RJ-45 signal interface cable side pin out.

Table 145 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

H.6.9 J6 - RVP901-WSP Misc IO A to RCP903 ASR9-WSP Panel

Table 146 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	+5 VDC
−5V	-5 VDC
Gnd	Ground

Table 147 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_PRET0	TTL 5V	Output	1,19	DIFF0 P,N
2,23	IFDR_HDR_INFO 0_P/N	TTL 5V	Output	2,20	DIFF1 P,N
3,24	IFDR_HDR_INFO 1_P/N	TTL 5V	Output	3,21	DIFF2 P,N
4,25	IFDR_HDR_INFO 2_P/N	TTL 5V	Output	4,22	DIFF3 P,N
5,26	IFDR_HDR_INFO 3_P/N	TTL 5V	Output	5,23	DIFF4 P,N

RCP903 Pin Number	Signal Name	Type	Direction	RVP901 IFDR Pin Number	Logical Connection
6,27	IFDR_HDR_INFO 4_P/N	TTL 5V	Output	6,24	DIFF5 P,N
7,28	IFDR_HDR_INFO 5_P/N	TTL 5V	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	ANALOG	Output	42,43	AMUX0 P,N
53,54	V_3PO	ANALOG	Output	46,47	AMUX1 P,N
57,58	V_1PO	ANALOG	Output	50,51	AMUX2 P,N
43,44	IFD_PRESENT	TTL 5V	Input	44,45	+5V, GND
45,46	NC_V_N5PO	ANALOG	No Connect	48,49	-5V, GND
62	GND	GND	Output	18	GND

H.6.10 J7 - Power Interface (DC)

The power supply sub-system is designed with the following assumptions about the input power source:

- Input voltage: Nominal 24 V ±4 V
- Maximum power consumption: 24 W
- Maximum input noise: 1.0 Vp-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 148 Pin-out for J7 Power Input

J6 Pin Number	Signal Name	Description
1	PWR_IN	24 V nominal input power
2	PWR_IN	24 V 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	24 V nominal input power
7	PWR_IN	24 V nominal input power
8	PWR_RETURN	Return path (ground) for input power
9	PWR_RETURN	Return path (ground) for input power

H.7 ASR9 RIM Software API

The RxNet7 Radar Interface Module (RIM) API functionality has been ported to RVP902-WSP and modified to use the RCP903 Panel Ethernet Interface for control and status functionality.

New functions (marked with a ⋆) allow monitoring of board status and manufacturing tests.

Commands that are no longer needed because those physical interfaces no longer exist have been deprecated. See Table 157 (page 470).

For more information on the functions, see the code comments.

Table 149 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 150 RIM API 6-Level Weather Functions

Function Name	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

Function Name	Description
`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 151 RIM API Diagnostic IO Functions

Function Name	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 152 RIM API Clock Functions

Function Name	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 153 RIM API Azimuth Functions

Function Name	Description
`rim_asr9_getaz*`	Gets the current antenna position and rotation count

Table 154 RIM API STC Functions

Function Name	Description
`rim_asr9_stcsel`	Sets the STC select map bits

Function Name	Description
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
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 155 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 I/Q data stream
rim_asr9_get_udp_addr*	Gets the UDP address and port information for the I/Q data stream
rim_asr9_iqdata_status*	Gets the current status of the I/Q Data Interface

Table 156 RIM API Simulated Controls

Function Name	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

Function Name	Description
`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
`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 157 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 I. TDWR Customizations

I.1 TDWR Technical Drawings

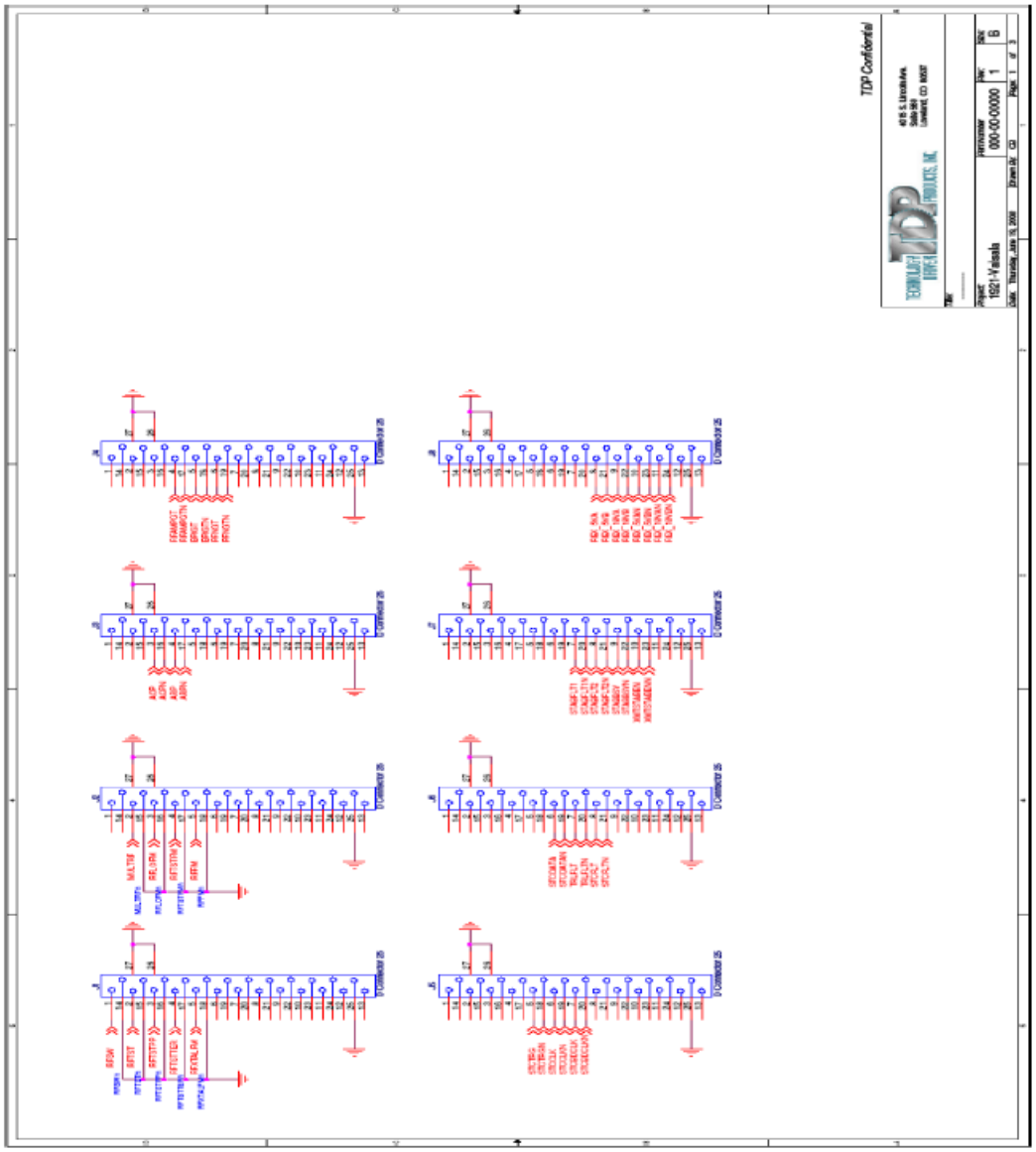


Figure 85 J1 to J9 Wiring Diagrams

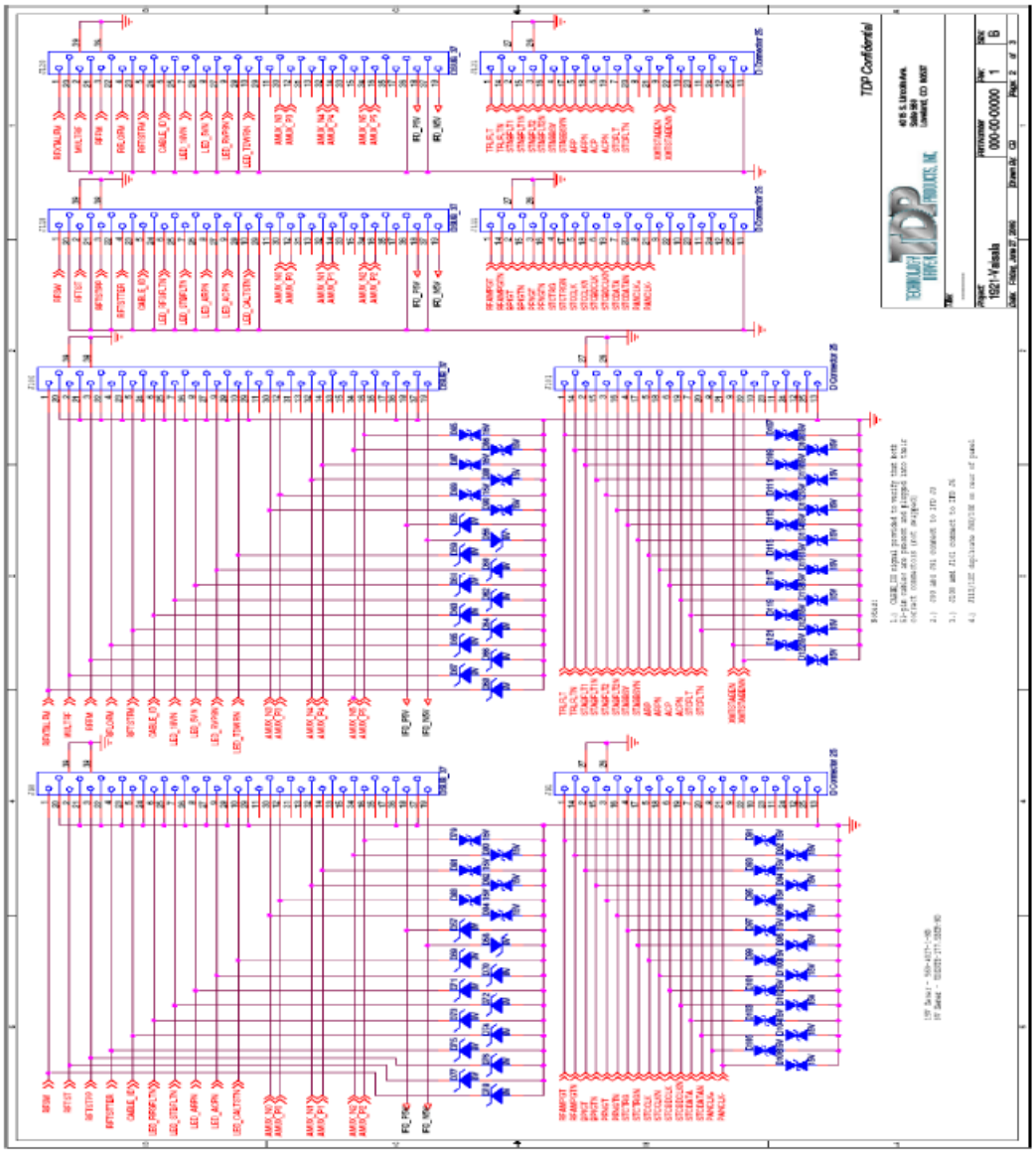


Figure 86 J90 to J111 Wiring Diagrams

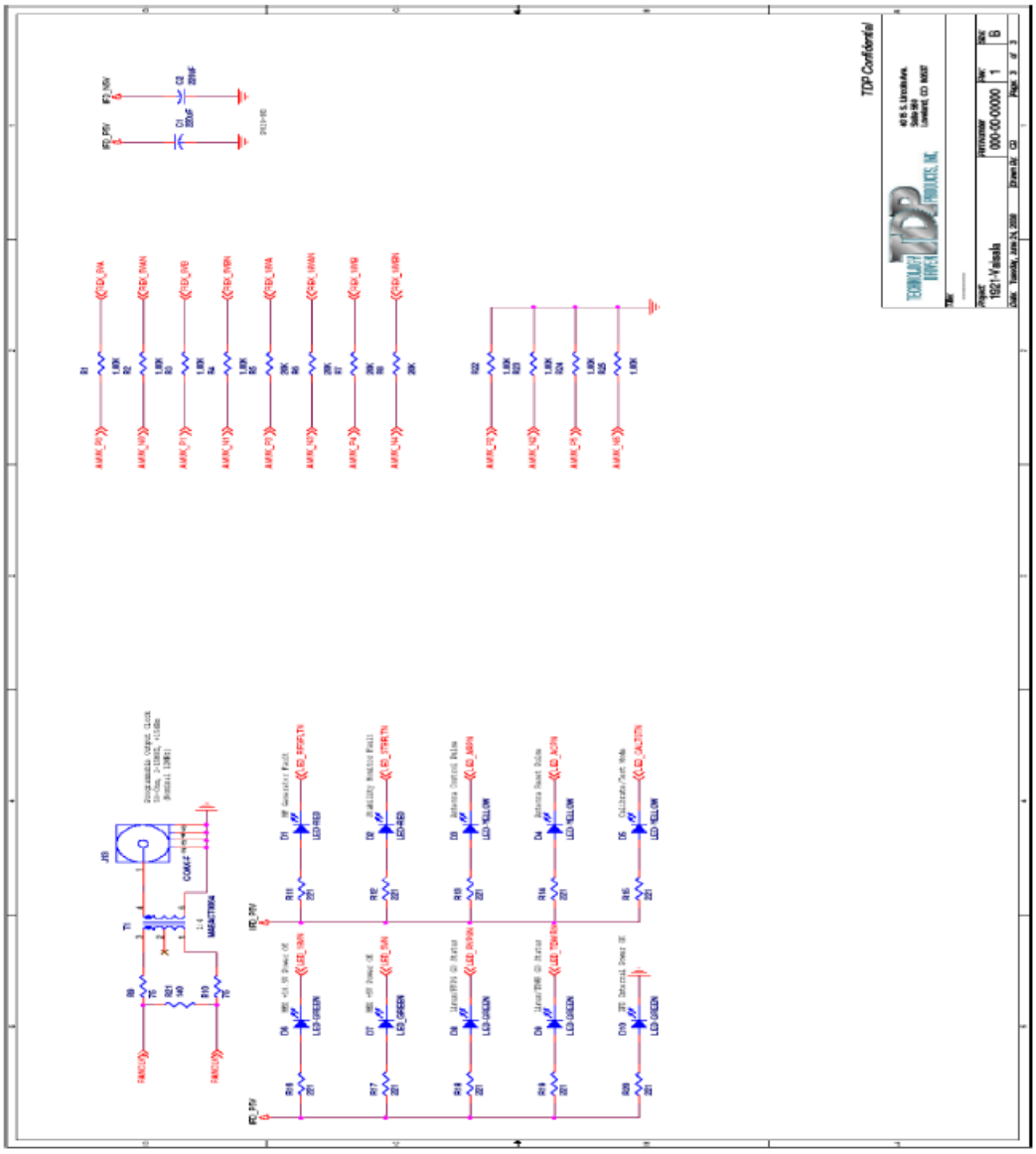


Figure 87 J13 Wiring Diagram

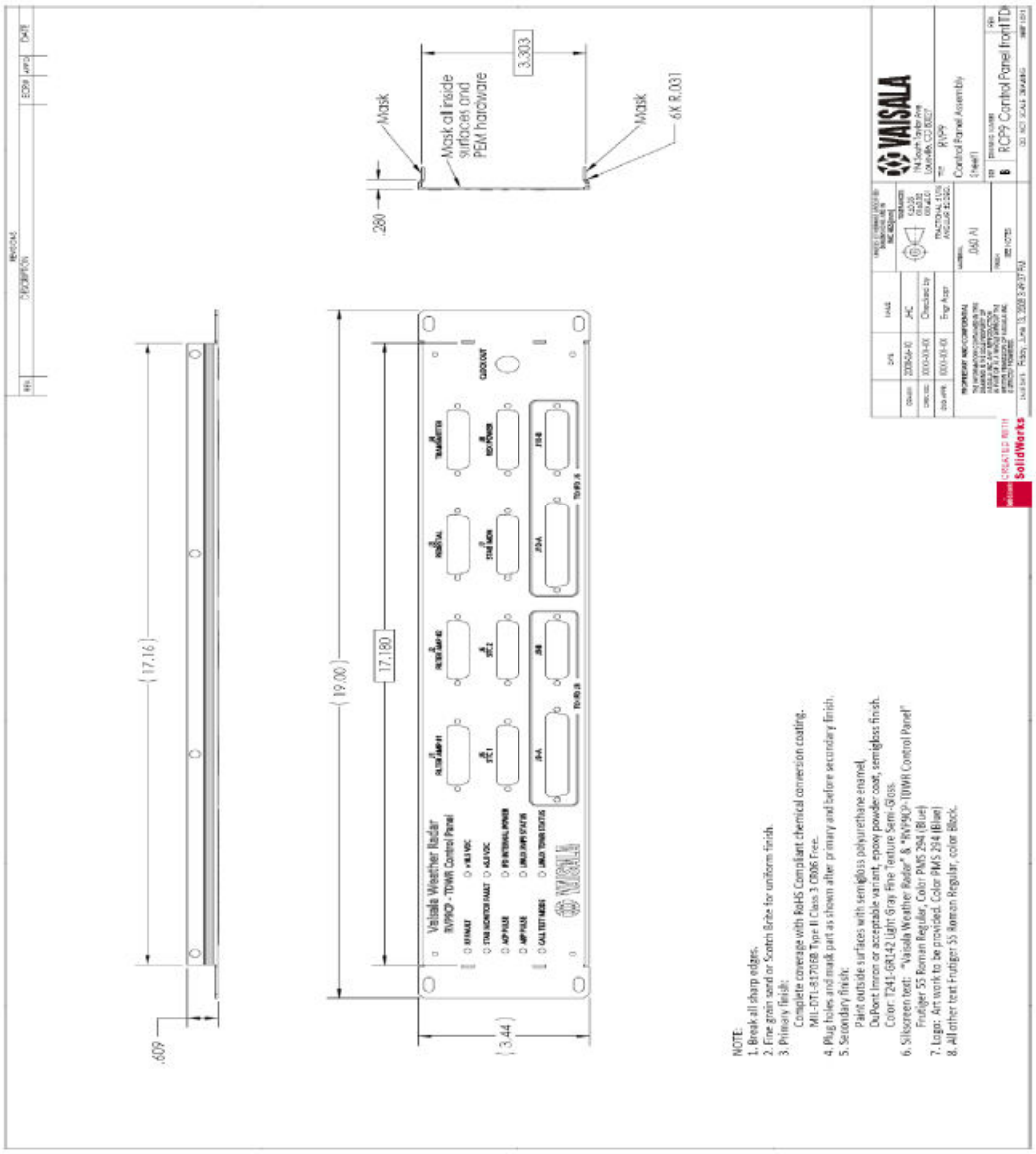


Figure 88 Control Panel Assembly

More Information

- TDWR Custom Back Panel (page 474)

I.2 TDWR Custom Back Panel

RVP900 can be supplied with an optional 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. See 4.2.3.1 Generic I/O Interconnect Breakout Cable (page 60).

The following tables show the signals assigned to the back panel's 25-pin I/O connectors. 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 8 connectors.

Table 158 J1 Filter Amp #1

Pin	Type	Direction	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 159 J2 Filter Amp #2

Pin	Type	Direction	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 160 J3 Pedestal

Pin	Type	Direction	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 161 J4 Transmitter

Pin	Type	Direction	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 162 J5 STC #1

Pin	Type	Direction	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 163 J6 STC #2”

Pin	Type	Direction	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 164 J7 Stability Monitor

Pin	Type	Direction	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

Pin	Type	Direction	Signal Name	Comment
25	—	—	GND	Common Ground

Table 165 J8 REX Power

Pin	Type	Direction	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

More Information

- TDWR Technical Drawings (page 471)

I.3 Softplane.conf File: RVP900 TDWR Panel Example

The following example shows RVP900 IFDR-related excerpts from the ***softplane.conf*** file.

```
#
#
# Softplane Configuration File
#
# The following general purpose control and status signals:
#
...
# -------------------- RVP9IFD #0 --------------------
#
# If you change the in-use flag, run 'softplane -resave' to rev the choices.
#
splConfig.Rvp9[0].lInUse = 1
# The remote backpanel type must be one of the following:
#    Common : Direct connections using the 'Common I/O' model
#    TDWR : MIT/LL TDWR custom breakout panel and protocols
# If you change this, run 'softplane -resave' to rev the choices.
#
splConfig.Rvp9[0].sNetPanel = "TDWR"
softplane.conf file: RCP8 example
```

The following shows an example from the beginning and some excerpts from the *`softplane.conf`* file.

To display the file on the terminal, give the command: **`cat`**.

```
Softplane Configuration File

The following general purpose control and status signals can be routed to/from
any available hardware pin. The '~' prefix character may be used for signal
inversion.
Control Outputs    Status Inputs
cPedAZ[15:0]    sPedAZ[15:0]
cPedEL[15:0]    sPedEL[15:0]
cEarthAZ[15:0]    sServoPwr
cEarthEL[15:0]    sLocal
cServoPwr    sStandby
cCabinetRelay    sLowerEL
cTransmitPwr    sUpperEL
cPWidth[3:0]    sTransmitPwr
cTrigBlank    sTransmitLocal
cRadiateOn    sPWidth[3:0]
cRadiateOff    sTrigBlank
cReset    sRadiate
cIrisMode[2:0]    sAirflowFlt
cAux[80:0]    sWavegpFlt
true    sInterlockFlt
false    sMagCurrentFlt
    sReset
    sIrisMode[2:0]

sAux[319:0]splConfig.sVersion = "8.00"
# ------------------- IO62 Slot #0 --------------------
#
splConfig.Io62[0].lInUse = 1
# The remote backpanel type should be one of the following:
# Direct : Direct I/O with IO62 connector itself
# IO62CP : Standard IO62-CP connector panel
# RCP88D : RCP8 portion of WSR88D panel
# RVP88D : RVP8 portion of WSR88D panel
#
splConfig.Io62[0].sExtPanel = "IO62CP"# TTL/CMOS on J1
#
splConfig.Io62[0].Opt.Cp.J1.pin01 = "sPedAZ[0]"
splConfig.Io62[0].Opt.Cp.J1.pin02 = "sPedAZ[1]"
splConfig.Io62[0].Opt.Cp.J1.pin03 = "sPedAZ[2]"
splConfig.Io62[0].Opt.Cp.J1.pin04 = "sPedAZ[3]"
splConfig.Io62[0].Opt.Cp.J1.pin05 = "sPedAZ[4]"
splConfig.Io62[0].Opt.Cp.J1.pin06 = "sPedAZ[5]"
splConfig.Io62[0].Opt.Cp.J1.pin07 = "sPedAZ[6]"
splConfig.Io62[0].Opt.Cp.J1.pin08 = "sPedAZ[7]"
splConfig.Io62[0].Opt.Cp.J1.pin09 = "sPedAZ[8]"
splConfig.Io62[0].Opt.Cp.J1.pin10 = "sPedAZ[9]"
splConfig.Io62[0].Opt.Cp.J1.pin11 = "sPedAZ[10]"
splConfig.Io62[0].Opt.Cp.J1.pin12 = "sPedAZ[11]"
splConfig.Io62[0].Opt.Cp.J1.pin13 = "sPedAZ[12]"
splConfig.Io62[0].Opt.Cp.J1.pin14 = "sPedAZ[13]"
splConfig.Io62[0].Opt.Cp.J1.pin15 = "sPedAZ[14]"
```

```
splConfig.Io62[0].Opt.Cp.J1.pin16 = "sPedAZ[15]"
splConfig.Io62[0].Opt.Cp.J1.pin17 = ""
splConfig.Io62[0].Opt.Cp.J1.pin18 = ""
splConfig.Io62[0].Opt.Cp.J1.pin19 = ""
splConfig.Io62[0].Opt.Cp.J1.pin20 = ""# TTL/CMOS on J2
#
splConfig.Io62[0].Opt.Cp.J2.pin01 = "cEarthAZ[0]"
splConfig.Io62[0].Opt.Cp.J2.pin02 = "cEarthAZ[1]"
splConfig.Io62[0].Opt.Cp.J2.pin03 = "cEarthAZ[2]"
. . .
splConfig.Io62[0].Opt.Cp.J2.pin15 = "cEarthAZ[14]"
splConfig.Io62[0].Opt.Cp.J2.pin16 = "cEarthAZ[15]"
splConfig.Io62[0].Opt.Cp.J2.pin17 = ""
splConfig.Io62[0].Opt.Cp.J2.pin18 = ""
splConfig.Io62[0].Opt.Cp.J2.pin19 = ""
splConfig.Io62[0].Opt.Cp.J2.pin20 = ""# TTL/CMOS on J4
#
splConfig.Io62[0].Opt.Cp.J4.pin01 = "sPedEL[0]"
splConfig.Io62[0].Opt.Cp.J4.pin02 = "sPedEL[1]"
splConfig.Io62[0].Opt.Cp.J4.pin03 = "sPedEL[2]"
. . .
splConfig.Io62[0].Opt.Cp.J4.pin15 = "sPedEL[14]"
splConfig.Io62[0].Opt.Cp.J4.pin16 = "sPedEL[15]"
splConfig.Io62[0].Opt.Cp.J4.pin17 = ""
splConfig.Io62[0].Opt.Cp.J4.pin18 = ""
splConfig.Io62[0].Opt.Cp.J4.pin19 = ""
splConfig.Io62[0].Opt.Cp.J4.pin20 = ""# TTL/CMOS on J5
#
splConfig.Io62[0].Opt.Cp.J5.pin01 = "cEarthEL[0]"
splConfig.Io62[0].Opt.Cp.J5.pin02 = "cEarthEL[1]"
splConfig.Io62[0].Opt.Cp.J5.pin03 = "cEarthEL[2]"
. . .
splConfig.Io62[0].Opt.Cp.J5.pin15 = "cEarthEL[14]"
splConfig.Io62[0].Opt.Cp.J5.pin16 = "cEarthEL[15]"
splConfig.Io62[0].Opt.Cp.J5.pin17 = ""
splConfig.Io62[0].Opt.Cp.J5.pin18 = ""
splConfig.Io62[0].Opt.Cp.J5.pin19 = ""
splConfig.Io62[0].Opt.Cp.J5.pin20 = ""# TTL/CMOS on J7
#
splConfig.Io62[0].Opt.Cp.J7.pin01 = "sAux[0]"
splConfig.Io62[0].Opt.Cp.J7.pin02 = "sAux[1]"
splConfig.Io62[0].Opt.Cp.J7.pin03 = "sAux[2]"
splConfig.Io62[0].Opt.Cp.J7.pin04 = "sAux[3]"
splConfig.Io62[0].Opt.Cp.J7.pin05 = "sAux[4]"
splConfig.Io62[0].Opt.Cp.J7.pin06 = "sAux[5]"
splConfig.Io62[0].Opt.Cp.J7.pin07 = "sAux[6]"
splConfig.Io62[0].Opt.Cp.J7.pin08 = "sAux[7]"
splConfig.Io62[0].Opt.Cp.J7.pin09 = "sAux[8]"
splConfig.Io62[0].Opt.Cp.J7.pin10 = "sAux[9]"
splConfig.Io62[0].Opt.Cp.J7.pin11 = "sAux[10]"
splConfig.Io62[0].Opt.Cp.J7.pin12 = "sAux[11]"
splConfig.Io62[0].Opt.Cp.J7.pin13 = "sAux[12]"
splConfig.Io62[0].Opt.Cp.J7.pin14 = "sAux[13]"
splConfig.Io62[0].Opt.Cp.J7.pin15 = "sAux[14]"
splConfig.Io62[0].Opt.Cp.J7.pin16 = "sAux[15]"
```

```
splConfig.Io62[0].Opt.Cp.J7.pin17 = "sAux[16]"
splConfig.Io62[0].Opt.Cp.J7.pin18 = "sAux[17]"
splConfig.Io62[0].Opt.Cp.J7.pin19 = "sAux[18]"
splConfig.Io62[0].Opt.Cp.J7.pin20 = "sAux[19]"
#
# Eight IO62 line pairs on J3
#
splConfig.Io62[0].Opt.Cp.J3_01_14.lRS422 = 0
splConfig.Io62[0].Opt.Cp.J3_01_14.iTerm = 0
splConfig.Io62[0].Opt.Cp.J3_01_14.pinPos = "cPWidth[0]"
splConfig.Io62[0].Opt.Cp.J3_01_14.pinNeg = "cPWidth[1]"
splConfig.Io62[0].Opt.Cp.J3_02_15.lRS422 = 0
splConfig.Io62[0].Opt.Cp.J3_02_15.iTerm = 0
splConfig.Io62[0].Opt.Cp.J3_02_15.pinPos = "cRadiateOn"
splConfig.Io62[0].Opt.Cp.J3_02_15.pinNeg = "cRadiateOff"
splConfig.Io62[0].Opt.Cp.J3_03_16.lRS422 = 0
splConfig.Io62[0].Opt.Cp.J3_03_16.iTerm = 0
splConfig.Io62[0].Opt.Cp.J3_03_16.pinPos = "cServoPwr"
splConfig.Io62[0].Opt.Cp.J3_03_16.pinNeg = "cTransmitPwr"
splConfig.Io62[0].Opt.Cp.J3_04_17.lRS422 = 0
splConfig.Io62[0].Opt.Cp.J3_04_17.iTerm = 0
splConfig.Io62[0].Opt.Cp.J3_04_17.pinPos = "cReset"
splConfig.Io62[0].Opt.Cp.J3_04_17.pinNeg = ""
splConfig.Io62[0].Opt.Cp.J3_05_18.lRS422 = 0
splConfig.Io62[0].Opt.Cp.J3_05_18.iTerm = 1
splConfig.Io62[0].Opt.Cp.J3_05_18.pinPos = "sPWidth[0]"
splConfig.Io62[0].Opt.Cp.J3_05_18.pinNeg = "sPWidth[1]"
splConfig.Io62[0].Opt.Cp.J3_06_19.lRS422 = 0
splConfig.Io62[0].Opt.Cp.J3_06_19.iTerm = 1
splConfig.Io62[0].Opt.Cp.J3_06_19.pinPos = "sRadiate"
splConfig.Io62[0].Opt.Cp.J3_06_19.pinNeg = ""
splConfig.Io62[0].Opt.Cp.J3_07_20.lRS422 = 0
splConfig.Io62[0].Opt.Cp.J3_07_20.iTerm = 1
splConfig.Io62[0].Opt.Cp.J3_07_20.pinPos = "sServoPwr"
splConfig.Io62[0].Opt.Cp.J3_07_20.pinNeg = "sTransmitPwr"
splConfig.Io62[0].Opt.Cp.J3_08_21.lRS422 = 0
splConfig.Io62[0].Opt.Cp.J3_08_21.iTerm = 1
splConfig.Io62[0].Opt.Cp.J3_08_21.pinPos = "sReset"
splConfig.Io62[0].Opt.Cp.J3_08_21.pinNeg = ""
# Two RS-422 Tx/Rx chips on J3
#
splConfig.Io62[0].Opt.Cp.J3_09_22 = ""
splConfig.Io62[0].Opt.Cp.J3_10_23 = ""
# Seven IO62 line pairs on J9
#
splConfig.Io62[0].Opt.Cp.J9_01_14.lRS422 = 0
splConfig.Io62[0].Opt.Cp.J9_01_14.iTerm = 1
splConfig.Io62[0].Opt.Cp.J9_01_14.pinPos = "sWavegpFlt"
splConfig.Io62[0].Opt.Cp.J9_01_14.pinNeg = "sAirflowFlt"
splConfig.Io62[0].Opt.Cp.J9_02_15.lRS422 = 0
splConfig.Io62[0].Opt.Cp.J9_02_15.iTerm = 1
splConfig.Io62[0].Opt.Cp.J9_02_15.pinPos = "sInterlockFlt"
splConfig.Io62[0].Opt.Cp.J9_02_15.pinNeg = "sMagCurrentFlt"
splConfig.Io62[0].Opt.Cp.J9_03_16.lRS422 = 0
splConfig.Io62[0].Opt.Cp.J9_03_16.iTerm = 1
```

```
splConfig.Io62[0].Opt.Cp.J9_03_16.pinPos = "sLocal"
splConfig.Io62[0].Opt.Cp.J9_03_16.pinNeg = "sStandby"
splConfig.Io62[0].Opt.Cp.J9_04_17.lRS422 = 0
splConfig.Io62[0].Opt.Cp.J9_04_17.iTerm = 1
splConfig.Io62[0].Opt.Cp.J9_04_17.pinPos = "sLowerEL"
splConfig.Io62[0].Opt.Cp.J9_04_17.pinNeg = "sUpperEL"
splConfig.Io62[0].Opt.Cp.J9_05_18.lRS422 = 0
splConfig.Io62[0].Opt.Cp.J9_05_18.iTerm = 0
splConfig.Io62[0].Opt.Cp.J9_05_18.pinPos = ""
splConfig.Io62[0].Opt.Cp.J9_05_18.pinNeg = ""
splConfig.Io62[0].Opt.Cp.J9_06_19.lRS422 = 0
splConfig.Io62[0].Opt.Cp.J9_06_19.iTerm = 0
splConfig.Io62[0].Opt.Cp.J9_06_19.pinPos = ""
splConfig.Io62[0].Opt.Cp.J9_06_19.pinNeg = ""
splConfig.Io62[0].Opt.Cp.J9_07_20.lRS422 = 0
splConfig.Io62[0].Opt.Cp.J9_07_20.iTerm = 0
splConfig.Io62[0].Opt.Cp.J9_07_20.pinPos = ""
splConfig.Io62[0].Opt.Cp.J9_07_20.pinNeg = ""
# Relays and relay drivers on J6
#
splConfig.Io62[0].Opt.Cp.J6_IntRelay1 = "cPWidth[0]"
splConfig.Io62[0].Opt.Cp.J6_IntRelay2 = "cPWidth[1]"
splConfig.Io62[0].Opt.Cp.J6_IntRelay3 = ""
splConfig.Io62[0].Opt.Cp.J6_ExtRelay1 = ""
splConfig.Io62[0].Opt.Cp.J6_ExtRelay2 = ""
splConfig.Io62[0].Opt.Cp.J6_ExtRelay3 = ""
splConfig.Io62[0].Opt.Cp.J6_ExtRelay4 = ""
# BNC testpoint monitors
#
splConfig.Io62[0].Opt.Cp.J13_BNC = ""
splConfig.Io62[0].Opt.Cp.J16_BNC = ""
# BNC trigger drivers direct from IO62 PCI card.
# Special signals 'trigger[8:1]' may also be used here.
#
splConfig.Io62[0].Opt.Cp.J14_BNC = ""
splConfig.Io62[0].Opt.Cp.J15_BNC = ""
splConfig.Io62[0].Opt.Cp.J17_BNC = ""
splConfig.Io62[0].Opt.Cp.J18_BNC = ""
# RS232 TTY transmitters/Receivers from IO62
#
splConfig.Io62[0].Opt.Cp.TTY0_Tx = ""
splConfig.Io62[0].Opt.Cp.TTY1_Tx = ""
splConfig.Io62[0].Opt.Cp.TTY0_Rx = ""
splConfig.Io62[0].Opt.Cp.TTY1_Rx = ""
# ------------------- IO62 Slot #1 --------------------
#
splConfig.Io62[1].lInUse = 0
# ------------------- IO62 Slot #2 --------------------
#
splConfig.Io62[2].lInUse = 0
# ------------------ IO62 Slot #3 --------------------
#
splConfig.Io62[3].lInUse = 0
```

Appendix J. References and Credits

- 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.
- Doviak, R.J., and D.S. Zrni, 1993: Doppler Radar and Weather Observerations. *Academic Press*, San Diego.
- 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.
- 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.
- The JMA internal specification for Interference Filter algorithm for use on Chitose airport Doppler weather radar is the basis for **`Alg.1`** in 7.2.5 Interference Filter (page 176).
- Environment Canada – Aldergrove BC, kindly supplied the snapshot of receiver data that is plotted in Figure 30 (page 154).

Glossary

AFC
automatic frequency control. See also, DAFC (page 486).

AGC
Automatic Gain Control. The gain of the linear channel video signals is adjusted based on an estimate of the next signal level. For example using the average power of the last few pulses at that range.

Antenna gain
The measurement of an antenna's ability to amplify the incoming microwave signals in a particular direction.

Antenna Utility
IRIS utility for manually controlling and monitoring the radar and antenna.

API
application programming interface

Ascope Utility
A diagnostic and test utility used for aligning and testing the radar and signal processor.

Azimuth
A direction in terms of the 360° compass; north at 0°, east at 90°, south at 180°, and west at 270°.

BITEX
Built-in test equipment software provided by IRIS.

Burst channel
Channel used for storing a sample of the transmitted microwave pulse.

CAN-BUS controller area network
Multicast shared serial bus standard, robust in noisy environments.

Clutter
Unwanted echoes on a radar screen. Microwaves reflected from, for example, ground, buildings, sea, insects, or birds.

Co-polar
The transmitted and received signals have the same polarization.

Coherency
Capability of a system to keep a stable phase reference from pulse to pulse.

COHO
coherent local oscillator

Cross-polar
The transmitted and received signals have orthogonal polarizations.

CSR
clutter-to-signal ratio

DAFC
Digital Automatic Frequency Control.

DFT
Discrete Fourier Transforms

Doppler spectrum width
The standard deviation of the Doppler spectrum in m/s. The spectrum width is a measure of the shear and turbulence in the radar pulse volume at a given range. See also Doppler velocity spectrum (page 486).

Doppler velocity (v)
Reflectivity-weighted average velocity of targets in the pulse volume. Determined by phase measurements from a large number of successive pulses. It is assumed that raindrops and other particles are advected with the wind and have no own motion except their falling velocity. Also called radial velocity.

Doppler velocity spectrum
The spectrum of the power returned as a function of the Doppler velocity (towards or away from the radar). The mean of the Doppler velocity spectrum is the mean velocity computed by the signal processor. The standard deviation of the Doppler spectrum is the spectrum width (in m/s).

Double conversion
Received signal frequency is down-converted in two steps; first to higher intermediate frequency (IF1) and then to final intermediate frequency (IF2).

Dual conversion
See Double conversion (page 486).

Dual polarization
Property of dual pol radars. Dual pol radars (also known as polarimetric radars) can transmit both horizontally (H) and vertically (V) polarized microwaves. In olden days this happened alternating with a high-speed switch, nowadays simultaneous dual pol is preferred. Comparing of H and V gives information on hydrometeors.

Elevation
The vertical pointing angle of the antenna; 0° is horizontal, 90° is vertical.

ENR
Excess Noise Ratio.

FFT
An RVP signal processing mode based on based on Fast Fourier Transform. See also PPP (page 488).

FIR
finite impulse response

FPGA
field programmable gate array

Gain
The measurement of a device's ability to amplify incoming signals.

GC/AP
Ground clutter and anomalous propagation

GMAP
Gaussian model adaptive processing

IF
Intermediate Frequency.

IFDR
Intermediate Frequency Digital Receiver.

Ingest file
A disk file of raw polar coordinate data that is collected during the execution of a TASK. Ingest files are used for subsequent product generation. See also RAW product (page 488).

IRIS
Interactive Radar Information System. A suite of software tools for configuring, calibrating, and operating a complete weather radar system.

JPOLE
Joint Polarization Experiment

K_{dp}
Specific Differential Phase Shift. The range derivative of the differential phase (PhiDP) expressed in degrees per km. It is nearly linearly proportional to the rainfall rate.

LDR
Depolarization ratio. Ratio between the reflectivity received at vertical polarization, but transmitted with horizontal polarization, and the reflectivity at horizontal polarization. Depolarization of the horizontal polarized pulse is caused by asymmetric particles which are not aligned horizontal. Depolarization of rain is normally very small (approx. -40 dB). It is high (-20 to -10 dB) for melting snow and water coated hail or graupel.

LLWAS
Low Level Windshear Alert System.

LNA
Low noise amplifier.

LRMSK
load range mask

Melting Layer
Melting layer or ML (height). Altitude at which falling snow starts to melt. This forms the top of the Bright Band. Also called the 0°C height.

MLHGT
Melting Level Height Detection

MSL
Mean sea level. An average level for the surface of the sea or ocean.

NSSL
National Severe Storms Laboratory, Oklahoma, USA

NWP
Numerical weather prediction

PLD
programmable logic device

PMI
Polarimetric Met Index

Polarization
Polarization of electromagnetic radiation is defined by the direction of the electric field. In traditional weather radars only horizontal polarization is used. In a dual-polarization weather radar both horizontal and vertical polarizations can be transmitted and received simultaneously.

PPP
Poly-Pulse Pair. An RVP signal processing mode based on pulse pair correlations. See also FFT (page 486)

RAW product
Spherical coordinate data product obtained directly from the raw ingest data. The data are stored in compressed format so they can be recorded on tape or sent to a workstation for further processing.

RCP
Radar Control Processor. Takes care of, for example, antenna movements.

RDA
Radar Data Acquisition software used in RVP900

RF
Radio frequency

RPG
radar product generator

RS
Rule strength

RVP900
Digital IF Receiver and Signal Processor. A floating-point programmable signal processor which directly samples the IF signal from a radar.

RX
Receiver

Signal processor
A programmable device for digitizing and processing video signals from the radar receiver.

Signal Quality Index (SQI)
: The autocorrelation of the received signal at lag 1 divided by lag zero. This is a number in the range [0,1] where 1 is the perfect Doppler point target and 0 is white noise. Typically used for thresholding velocity and width at a level of ~0.3...0.4.

SNR
: signal-to-noise ratio

STALO
: Stable Local Oscillator. A very stable radio frequency local oscillator used in radar for mixing received signals to produce an intermediate frequency signal.

STAR mode
: Simultaneous Transmit And Receive. The operational mode of a dual polarization radar where the radar transmits simultaneously horizontally and vertically.

TCF
: threshold control flags

TX
: Transmitter

VCXO
: voltage-controlled crystal oscillator

WSP
: weather system processor

Index

Hardware Limited Warranty

Vaisala warrants its RVP900 Digital IF Receiver and Signal Processor to function according to the hardware User Guide 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.

For standard warranty terms and conditions, see www.vaisala.com/warranty.

Please observe that 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. Please see the applicable supply contract or Conditions of Sale for details of the warranty for each product.

Technical Support

Contact Vaisala technical support at helpdesk@vaisala.com. Provide at least the following supporting information:

- Product name, model, and serial number
- Name and location of the installation site
- Name and contact information of a technical person who can provide further information on the problem

For more information, see www.vaisala.com/support.

Recycling

Vaisala has implemented return facilities for all products that we bring to market. Return all RVP900 and RVP900-WSP components to the following address for recycling:
Vaisala Inc., 194 South Taylor Ave. Louisville, CO 80027
(303) 499–1701
Do not dispose of RVP900 components in landfills.
RVP900 and RVP900-WSP components are marked with the end-of-life, not for landfill disposal symbol in accordance with European Standard EN 50419.

VAISALA

www.vaisala.com