RDA 8.04.8 Release Notes (2 Feb 2004)

These release notes cover changes made to the SIGMET Radar Data Acquisition platform, including primarily the RVP8 and RCP8 products. The last release was RDA–8.04.5 generated on 24 December 2003. If you are upgrading from an earlier release, please read those notes also.

Important Installation Changes

1. Starting with this version, the RVP8 requires that the IRIS_TEMP and IRIS_LOG directories exit. These were not required in previous versions, and were not created on older systems shipped from the factory. So, after upgrading, please type the following commands:

```
# mkdir /usr/iris_data/temp
# mkdir /usr/iris_data/log
# chown operator /usr/iris_data/temp
# chown operator /usr/iris_data/log
```

New Features

1. It is now possible for trigger sequences defined with iNominalTrigSequence() and iCustomTrigSequence() to include looping, i.e., the last pulse in the sequence can loop back to an earlier pulse other than the first one. This is handy for defining complex trigger waveforms consisting of an initial preamble followed by some other repetitive sequence. Loops are requested by setting TGPFLG_LOOPBACKPULSE in the pulse that you wish to loop back to.
2. Some changes have been made to *struct rvp8MajorMode* :
    - The *lBlocksOfLowPRF* flag has been removed because it did not really belong in this structure. See *rda/rvp8main/open/cpi.c* which has been significantly reorganized to handle batched and other nonuniform PRT sequences.
    - Major modes now have better control over when and how their custom trigger patterns are loaded via the new *iTriggerOptions* field. Triggers can be defined as depending on other processor parameters (such as sample size) that would not normally affect them, in which case those trigger waveforms are reloaded when a PROC command first begins to use them. In addition, triggers can be setup to change only when a PROC command arrives, rather than right away as SETPWF and other similar opcodes arrive. This helps reduce external trigger thrashing during RVP8 setup operations. Lastly, external triggers can be defined as being illegal in major modes that rely so heavily on the RVP8 trigger generator that an external trigger could not possibly work correctly (See New Feature #6.)
3. The signal processing that will need to be done within any CPI, including those which contain complex trigger patterns, is now blocked out by *lAssignProcsInCPI()* . This mechanism allows several pulse intervals within each CPI to be characterized by the RVP8/Main threads for later use by the RVP8/Proc processes.

4. The *lPossibleStartCPI()* major mode routine has been removed, and that functionality has been moved into args passed into *lIntFindNextCPI_xxx()* (which are the only places where that routine is used).

5. The *ctBinMoments()* processing phase now uses a fixed distribution of range bins to each processing thread based on how the (I,Q) data can most efficiently be extracted from the timeseries API. Thus, range information is no longer passed into this routine. This change provides a uniform CPU load in major modes that have complex trigger waveforms that do not lend themselves to apriori assigned bin intervals. Related arg changes were also been made to *lSetupProcBins*() .

6. An additional error bit has been added to GPARM Word #59 (Immediate Status #4) to indicate that an external trigger has been requested in a major mode that requires an internal trigger. The **ascope** utility will popup this error whenever it occurs. Internal RVP8 code should call *lUsingExternalTrigger()* to test whether internal triggers are presently being used.

7. The RVP8 and RCP8 now have the option of resetting their attached PCI cards to their powerup state upon shutdown. Previously, the RVP8/Rx, RVP8/Tx, and I/O-62 cards would simply continue running their configured firmware procedures after the software had exited. New setup questions in the “Mc” and “Site Misc” menus allow the exit behavior to be chosen.

8. The RDA release from SIGMET now includes an optional time series recording and playback software package. See Appendix G of the *RVP8 User’s Manual* for full documentation. Included with this package are the following programs:

    - **Tsswitch** — Controls where the time series are read from.
    - **Tsexport** and **tsimport** — Transfers time series in real time over the network.
    - **Tsarchive** — Records time series to disk and alternatively selects which time series files from disk to play back.

    Because of disk space concerns, the normal operating mode is to record on a dedicated archive host. However the tsexport/import interface uses the same standard TS API, so the archiving program can also run locally. We encourage other developers to use our utilities in their applications. The **tsarchive** data format on disk is public and documented.

9. Two new fields have been added to *struct rvp8PulseHdr* to indicate the transmit phase and polarization of each recorded pulse. Note that these are the nominal requests from the RVP8 internal trigger generator, and thus, are probably not useful when an external trigger is being used.

10. The RVP8 now includes built-in software that can simulate up to four physical targets, each having a starting range and length in km, a reflected power in dB below saturation, and a Doppler shift in Hertz. These targets are properly illuminated by whatever phases and periods of transmit pulses have been sent so far (up to eight prior trips), and thus, their echoes properly show the effects of range/Doppler folding and phase modulation.

This "Target Simulator" provides a very powerful capability for testing RVP8 algorithms that work with folded and coded data. It was written specifically to test the new BATCH processing mode, but it is a fully general tool that works in all RVP8 major modes. The simulator is fully supported in the DSP driver and is available through the *Options* menu in **ascope**. The (I,Q) samples for the simulated targets are simply summed into whatever timeseries data already exist within each pulse. It works properly both with live signals from the RVP8/Rx card and with simulated (LSIMUL opcode) signals from **ascope**.

11. The RCP8 *site host* menu now allows you to choose "void" for the type of RCVnn or XMTnn packets that are exchanged with the host computer. This can be useful when simultaneously running more than one RCP8 that uses network packet I/O. The setup question *Process incoming servo control packets* has been removed since you can now get that effect by setting the XMTnn type to "void".

12. The RCP8 now starts up cleanly even when the kernel module is absent as long as the IO62 card selection is set to –1. This is handy when running/debugging the RCP8 in a limited development environment. Previously, the RCP8 would run in such cases but warnings were issued on startup.

13. The sequential algorithms that are implemented in the RVP8 compute processes have been generalized to be much more flexible. There is still a *binmoments* phase that is always the first to run; but this is now followed by an unstructured array of subsequent processing steps (up to 12), rather than the preassigned *binparams*, *rayparams*, and *rayformat* phases. You only need to rewrite a few lines of your major mode *init* code. For example, in *rvp8proc/open/ct_ppp.c* we change:

```
MMode_c.ctBinMoments = ctBinMoments_ppp ;
MMode_c.ctBinParams  = ctBinParams_ppp ;
MMode_c.ctRayParams  = ctRayParams_ppp ;
MMode_c.ctRayFormat  = ctRayFormat_ppp ;
```

in which three explicitly named phases are assigned, to the more generic:

```
MMode_c.ctBinMoments = ctBinMoments_ppp ;
MMode_c.ctAlgorPasses[0] = ctBinParams_ppp ;
MMode_c.ctAlgorPasses[1] = ctRayParams_ppp ;
MMode_c.ctAlgorPasses[2] = ctRayFormat_ppp ;
```

Thus, the organization of the existing RVP8/Proc code is not being changed in any significant way; but if you do require additional processing phases they are now easily available. This would allow you, for example, to structure your code as a larger number of simpler and more reusable processing passes.

Note that the *setupProcBins()* method is now called at the start of each one of these phases so that you have the option each time of allocating different range bin intervals to the available processors.

14. The RVP8 now supports a new DB_FLAGS data parameter type. This encodes either one byte or two bytes per bin, and allows data dependent flags to be conveyed at each individual range bin within an output ray.

15. The RVP8 now supports the WSR88D legacy "Batch" processing mode. This is actually a fully general implementation of a Lo/Hi Surveillance/Doppler PRF unfolding scheme

that provides all of the legacy features as special cases. The following new questions appear in the **Mp** menu:

```
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 :
```

The parameters are defined as follows:

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

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

16. A new control function *memoryBarrier()* is now available within RVP8/Proc. When called by a compute process it causes it to block until all others have reached that same call, at which point all processes proceed in unison. Memory barriers provide a simple way to coordinate read/modify/write operations within a single processing pass. Of course, you can also restructure you code and use additional passes to accomplish the same result; but memory barriers can be embedded within common routines and are often simpler to manage. An example can be found in *pointClutter_dflt()* .

i **Note: *memoryBarrier()* must be called uniformly from each one of the compute processes, otherwise the entire set of processes will hang. You are safe as long as it is never conditionally called based on the processor slot *iProc_c* , or any other data that are specific to a slot.**

17. Two changes have been made to the RVP8's management of WSR88D triggers:

- The *filament_sync* trigger is now active only while radiating. Previously this trigger was active all the time.
- The radiate shutdown sequence now includes a *post_charge* trigger following the final discharge of the transmitter. This is included so that the high voltage monitoring circuitry will sample and show the final discharged value.

18. The RVP8 now implements a generalized point clutter detection and removal algorithm that provides WSR88D legacy behavior as a special case. The algorithm runs on the raw moment data (prior to range averaging) and works in all major modes. A bit is set in the DB_FLAGS output data parameters telling where point clutter has been detected, and the procedure can then optionally interpolate across such bins to effectively remove them. A new driver entry point *dspw_pointClutter()* provides access to the algorithm.

19. The RCP8 now sends a Data Message Command to the DAU only when it receives a BITE_SET_VALUES request from IRIS, or when it has already received such a request and any of the control variables C54–C63 have changed since the time the last command was sent. Previously, commands were being sent regularly to the DAU, and this made it difficult to powerup the RCP8 without creating momentarily incorrect DAU states.

Bug Repairs

1. Noise levels were not being rescaled properly in the dual polarization modes of the RVP8. This would show up as peculiar side effects of changing the ZDR and LDR offsets after taking a noise sample. Many thanks to Valery Melnikov at NSSL for discovering and reporting this problem.

2. A bug was corrected in the formatting of legacy 8-bit timeseries output from the PROC command. In spite of having fixed this, we still hope that nobody is actually using this format.

3. A minor bug was repaired in which both the RVP8 and RCP8 would sometimes report an incorrect count of words changed following a *save* command.

4. Repaired a bug in the data recorded by **tsarchive**. It was always setting the flag indicating a gap in the pulse stream. This is now set only when there really is a gap.

RDA 8.04.5 Release Notes (24 Dec 2003)

These release notes cover changes made to the SIGMET Radar Data Acquisition platform, including primarily the RVP8 and RCP8 products. The last release was RDA–8.04.4 generated on 8 December 2003. If you are upgrading from an earlier release, please read those notes also.

New Features

1. The following new fields have been added to *struct gparm* :
    - Word #59 now contains a bit indicating that the TimeSeries API data source is external to the RVP8. This allows software using the DSP driver to know when the timeseries data are coming from a source other than the live RVP8/Rx card.
2. A new *iCalls_a* argument has been added to *iCustomTrigSequence()* to provide a count of the number of calls that have been issued so far. Since this routine returns successive trigger bank definitions for each successive call, it is handy to have such a count upon entry. Also, the *pBank_a* argument is now set to –1 initially, and retains its previous return value on each subsequent call.
3. Several dozen *typedef* definitions have been added within the overall RDA tree to improve the readability of the headers, prototypes, and code.
4. The routines that convert (I,Q) data between FLT4 and packed UINT2 have changed. The last argument is now a flag word that chooses the high-SNR format (as before), and also permits byte swapping of the 16-bit packed words. An underscore has been added to all four function names to insure that old code will be flagged by the compiler.
5. The RVP8 now implements “PRF Pie Slices” via the new PRFSECT opcode. This new feature allows data to be collected at different PRFs depending on where the antenna is pointed.
    - The DSP driver interface that allows you to create PRF sectors is *dspw_prfSect()*. Note that you must specify your active PRF sectors only after having first called *dsp_set_prf()* to set the fundamental default trigger rate.
    - Up to eight solid sectors in AZ/EL can be specified, and a different PRF can be selected within each one.
    - PRF Sectors can be used in all of the RVP8 major modes, including those that generate irregular or batched triggers. Major modes wishing to build on this mechanism should define custom triggers with *iCustomTrigSequence_dflt()*.
6. The *iBtime* field within the *rvp8PulseHdr* structure has been renamed *iBtimeAPI*, and now indicates the local time at which the pulse was inserted into the timeseries API. This structure element also is now filled in automatically by *rvp8tsInsertPulseHdr()*, making it unnecessary for API writer clients to have to worry about it. Note that the *iTimeUTC* and *iMSecUTC* fields still retain their meaning as the absolute UTC pulse time. Appropriate follow–up changes were also made to *rdasubs/rvp8ts_example.c.*

7. The RVP8 now computes the bin-to-bin phase that is introduced by the hardware FIR filters, and removes this effect in the *rawPulseCorrections_dflt()* routine. Although this correction is not crucial for most types of data processing, you will notice much nicer behavior in the **ascope** plots of (I,Q) versus range.

Bug Repairs

1. The GMAP clutter filter was not working properly with Dual-PRF data. The clutter width was not adjusting properly to the separate Hi/Lo PRF intervals.

2. The *PRF-Varied* GPARM error bit is now set properly by the RVP8. Previously, this bit was always clear.

RDA 8.04.4 Release Notes (8 Dec 2003)

These release notes cover changes made to the SIGMET Radar Data Acquisition platform, including primarily the RVP8 and RCP8 products. The last release was RDA–8.04.3 generated on 1 December 2003. If you are upgrading from an earlier release, please read those notes also.

Note: This release was built under RH8.0 and was not widely distributed

New Features

1. The following new elements have been added to *struct rvp8PulseInfo* :
    - Acquisition clock rate (to calibrate the *iSysTime* pulseheader counts)
    - Saturation input power corresponding to Mag(I,Q) of 1.0
    - Radar wavelength
    - Tx/Rx Polarization selection
    - Transmit phase modulation
    - PRF and Range at which noise was sampled
    - Pulsewidth code, and actual duration in microseconds
    - Copies of GPARM immediate and latched status, and diagnostic bits
    - IRIS/RDA version (full Major/Minor string) that created the data
2. The following new fields have been added to *struct gparm* :
    - Tx phase modulation is included in Word #52
3. The RBACK command now allows you to read back the entire *rvp8PulseInfo* structure that is associated with the most recent pulse of data in the Timeseries API. Use the new RBACK_PULINFO selection when calling *dspr_rback()*. Note that RBACK_TASKID has been removed since it was merely a special case of this new type.
4. An additional bit has been added to the CFGHDR command to request that a UTC time stamp (full time, plus milliseconds) be added to the ray header. The time corresponds to the first pulse of data that contributed to the ray.
5. It is now possible to choose the buffer size of the (I,Q) data that are stored in the RVP8 timeseries API. Simply add a line of the following form to your *rvp8ts.conf* file to set the number of (I,Q) vectors that are buffered:

    ```
    rvp8tsConfig.nIQDVecs = 5000000
    ```

    The default value of 2,400,000 will continue to be used whenever this line is missing from the file. That value provides roughly two seconds of buffering for data that are acquired at 125-meter resolution. However, when using finer bin spacings, or range masks that extend far beyond the “first bang”, you may find that the buffer size must

increase in order to hold a long enough duration of data. If you raise this value, you will almost certainly need to raise the maximum shared memory on your system. You can do this with a command like:

```
# echo 60000000 > /proc/sys/kernel/shmmax
```

Once you have this working, put this command into your /etc/rc.d/rc.local.

RDA 8.04.3 Release Notes (1 Dec 2003)

These release notes cover changes made to the SIGMET Radar Data Acquisition platform, including primarily the RVP8 and RCP8 products. The last release was RDA–8.04.2 generated on 19 November 2003. If you are upgrading from an earlier release, please read those notes also.

New Features

1. Several additional consistency checks have been added to the RVP8 trigger generation routines to detect and report illegal pattern and timing requests. This builds on the error flags that were recently added (see new feature #5. in the RDA-8.01.12 release notes) as follows:
    - An additional test for whether any defined triggers are being cutoff by an external trigger input that does not have sufficient pretrigger delay. This will reliably detect any subtle problems with burst pulse sampling when an external trigger is being used.
    - An error report in the **V** command whenever trigger synthesis errors of any kind have been detected.
    - A conspicuous warning in the **Pb** (Plot Burst) status line showing pattern and timing errors that might arise while triggers are being interactively shifted left and right.
2. The RVP8 now signals an error (visible in the RVP8's startup window) whenever it encounters an illegal opcode on its host computer interface. This can be handy for debugging driver software that communicates with the RVP8.
3. The *matchedFilterCoefs()* routines now return two separate coefficient arrays from which "I" and "Q" data will be computed in the RVP8/Rx card. These routines used to return only a single array since the traditional RVP8 matched bandpass filter coefficients were(and still are) time-reverse symmetric. However, relaxing this assumption is a step toward supporting more complex filtering schemes, including pulse compression.

Bug Repairs

1. Sector blanking based on the angle limits defined in the **Mt** menu was not working properly. The RVP8 was running with the setup values that were present at startup time, and would ignore typed in changes unless they were saved and the RVP8 subsequently restarted. This bug has been present for a long time.
2. The RVP8 now recovers properly when an external trigger is reapplied after having been missing for more than a few tenths of a second. The trigger timeout condition was not being exited properly, and the reapplied trigger was not being seen right away.
3. The **Mb** question *IF increases for an approaching target* now has the proper effect on the (I,Q) data. Previously, the answer to this question was being ignored.

RDA 8.04.2 Release Notes (19 Nov 2003)

These release notes cover changes made to the SIGMET Radar Data Acquisition platform, including primarily the RVP8 and RCP8 products. The last release was RDA–8.04 generated on 4 November 2003. If you are upgrading from an earlier release, please read those notes also.

New Features

1. The RVP8 and RCP8 now report the power supply current that is being drawn by the IO-62 standard backpanel. The hardware inventory listed by either product now shows:

```
Found PCI Card I/O–62  – Rev.B  Serial:1841  Code:19
 \–-> IO62CP Backpanel – Rev.B  Serial:2112  Code:3
     ( Supply Currents – Panel: 406 mA,  Relays: 38 mA )
```

   Two separate currents are shown. The "panel" value is the current flowing at the input of the DC-DC converter that produces all of the on-board supply voltages for the backpanel. The "relay" value is the current being drawn directly from the IO-62 card by any external relays (or other loads) that are attached to connector J6, plus the small amount of circuitry that drives those pins. The sum of the two numbers is the total current being supplied by the IO-62 card.

   The backpanel is driven from the +12V supply of the IO-62 PCI card, which in turn is driven either from the 4-pin Molex power connector on the board (recommended), or directly from the +12V PCI card edge finger. In the former case the total current should not exceed 1800mA (21.6 Watts), and in the latter case it should be limited to 850 mA (10.2 Watts).

2. Several new startup diagnostic bits have been added to the RVP8 GPARM Diagnostic Result Register "A". Softplane configuration errors are noted, along with any signals raised during startup or a startup that ran without *root* privileges. The following warning will be printed at the end of the startup console messages for the RVP8 (and RCP8) whenever any problems are found:

```
******************************************************
  WARNING: Configuration/Startup errors were detected
******************************************************
Initializations complete – The RVP8 is now operational.
```

   Note that RVP8 startup errors are also flagged in ascope and IRIS popup windows, so you will be aware of these conditions even if the startup messages were never seen.

3. The RVP8 can now provide direct conversion of 3-wire synchro waveforms for the AZ and EL position angles. You may directly hookup AZ/EL synchros to the 12-pin input connector on the IO-62 standard backpanel when you answer "3" to the following **Mc** setup question:

```
Live angle input – 0:None, 1:Sim, 2:TAGs, 3:S/D
```

   The following additional status line will also appear in the **V** command printout when synchro conversion is enabled:

```
Synchro Voltages: AZ Ref:0% Syn:0% (FAULT)  EL Ref:53% Syn:86%
```

For additional details on S/D conversion within the RVP, please see the RCP8 write-up in the 8.01.07 release notes, item #2.

4. The RVP8 now has a *–daemon* flag that causes the processor to run as a background daemon process.

5. Two improvements have been made to the RVP8 spectrum width estimator:

   - The R0/R1 estimator is now corrected for system noise. This removes the bias that used to be present for low SNR cases.
   - A correction is now applied to repair the spectral broadening that results when the original timeseries have been windowed. This is only a concern for narrow spectra having normalized widths less than 0.075 (or so).

6. A sitename field has been added to *struct rvp8PulseInfo* so that the original data source can be identified in the TimeSeries API. Please run **setup** on all RVP8s to set the sitename to a meaningful string.

7. It is now possible to set individual data qualification thresholds for each parameter that is computed by the RVP8. There are several components to this change:

   - A new THRESH opcode has been added that allows the threshold criteria to be set for a specific data parameter.
   - A bit has been added to the SOPRM opcode that prevents threshold values from being set by the command. If this new bit is not set, then all data types will be qualified just like they always were.
   - The *dspw_thresh()* DSP driver function still sets the threshold levels for all data types just as it always has. However, you may also call *dspw_thresh_types()* to define individual threshold criteria. The latter routine must only be called after the former.
   - The secondary SQI threshold is still defined as a *Slope* and *Offset* from the value listed in the SOPRM command and sets the SQI comparison level for dBZ and dBT. This is now the only context in which this secondary value is ever used.
   - A new readback option is added to the RBACK command to allow the threshold values to be examined for all parameters.
   - GPARM words 33-36 now return the threshold settings for the *Velocity* parameter by default.

8. Unique keys have been added in */usr/sigmet/bin/keys* for each major mode so that you can map your own custom structure (e.g., 'user1Share') across the master and compute processes with a simple call such as:

```
rdaShmemMap( "rvp8_user1", sizeof(struct user1Share),
             lCreate, lWriteable,
             &user1Addr, &user1Size, &user1Chan ) ) ;
```

The *lCreate* flag should be *TRUE* in the master init call, and *FALSE* in the init call for the compute processes.

9. The allocation of available trigger pattern RAM on the RVP8/Rx card has been doubled. This allows you to define longer and more complex banks of trigger patterns than before.

10. An error bit is now raised (Bit #5 of GPARM Word #59) when the RVP8 trigger pattern RAM is not able to hold all of the desired trigger/phase/polarization patterns. The **ascope** utility will popup this error whenever it occurs.

11. The RVP8 front panel display now shows an approximate measured PRF when supplied with an external trigger.

12. Dsp_lib now contains a new entry point **DspWriteTaskId()**. This should be used by all application programs to set the task name string and sweep number in the RVP8. With this, recorded time series data can be intelligently tagged. We encourage all developers to use this feature.

Bug Repairs

1. The RCP8 was failing to open one of its key files due to it's having incorrect protections. This can be repaired manually with:

```
# chmod 644 /usr/sigmet/bin/keys/rcp8_global
```

2. The *iSweep* and *iAuxNum* fields of the pulse information structure that is created by the RVP8 were not being initialized properly.

3. Custom trigger PRT sequences loaded via SETPWF and having a length greater than one are now working properly in the RVP8. Previously such sequences were ignored.

4. A DSP driver bug was repaired that would cause the desired polarization to sometimes not be loaded into the signal processor.

5. In Dual-Polarization Dual–IFD External-Trigger mode, the RVP8 was not setting the timing properly on the secondary RVP8/Rx card. This was mainly apparent in the **Pb** menu where the burst pulse could not be found for the secondary channel.

6. The sense of the external pretrigger input line has been corrected; it has been inverted for the entire lifetime of the RVP8. If you are using external triggers, please reverse your answer to the **Mt** setup question:

```
PreTrigger active on rising edge: Yes/No
```

RDA 8.04 Release Notes (4 Nov 2003)

These release notes cover changes made to the SIGMET Radar Data Acquisition platform, including primarily the RVP8 and RCP8 products. The last release was RDA–8.02.2 generated on 12 September 2003. If you are upgrading from an earlier release, please read those notes also.

New Features

1. The Andrew Canada ACU serial pedestal interface is now supported in the RCP8. The following new setup questions appear in the "site custom" menu:

```
Use Andrew–Canada serial pedestal interface: YES
  Serial port: io62–tty0
  Get Pos/Vel from serial status packets: YES
  ID of Andrew BITE status packets:  11(decimal)
  Map Andrew status into S[29:63] variables: NO
  Apply boresite offsets to position angles: NO
  Simulator port:
```

   When the Andrew interface is enabled you may hookup the serial lines either through a standard Linux TTY port such as "/dev/ttyS0", or through the special "io62-tty0" serializer that is built into the IO62 card firmware and is described below in New Feature #2. The RCP8 also contains a (minimal) serial simulation of a real Andrew ACU, which you can configure onto a TTY port for loopback testing.

   Normally the RCP8 will receive high-speed parallel AZ/EL angles from the Andrew ACU. However, a setup question allows the RCP8 to use the low-speed (5Hz) serial angle status information from the ACU instead. This option can be handy for testing.

2. The IO62 now supports a totally generic bidirectional serial interface that you can hookup either as RS232 or RS422 electrical levels. What makes this new feature interesting and useful is that the serial encoding and decoding of raw Tx/Rx bits is done entirely in C code living in the RDA kernel module. Thus, the RVP8 and RCP8 can now support any custom serial protocol that might come along such as (historically) the Honeywell INU HDLC stream, and the ASR9's 13-bit CD-2 stream. For now, standard asynchronous serial format is supported just to get things started.

   The serializer must be connected to electrical ports on the IO62 card and/or backpanel. This is done in the *softplane.conf* file using the following new stanza that will appear in response to a *softplane –resave* command:

```
# Pin assignments for generic serializer #0.  Options are:
#             TTY0 : Rx/Tx on TTY0 RS–232 line pair
#      Rx:12/Tx:16 : Rx/Tx on lines 12/16 (J9 Pins 1/5)
#
splConfig.Io62[0].Opt.Cp.ioSerial0_TxRx = ""
```

   Use the first option to hookup to the RS-232 lines on the 9-pin backpanel connector J10. Or, the second option can be selected for RS-422 I/O using two signal pairs on connector J9. Note that the serializer might also be implicitly hooked up to other pins when the *sExtPanel* attribute specifies a custom panel type, e.g., for custom TDWR functions.

3. The RVP8 and RCP8 both use a major and minor version numbering scheme that was inherited from their ancestors. There is quite a bit of legacy driver code that relies on these two numbers behaving in certain ways, and thus, they can not easily be replaced with the associated IRIS 3-digit values. To help reduce confusion, the RVP8 and RCP8 now at least track a common RDA 2-digit version numbering scheme, and the associated IRIS version is also shown on all startup and status banners from both products.

4. Configuration of the RVP8/Tx Digital Transmitter card has been brought out to the OPEN directory and is now callable from SITE code. The *configTxCardWaveforms()* function can be used to redefine the output frequency, power, phase modulation and amplitude modulation of the synthesized IF on either Tx BNC channel.

5. The STC support for TDWR is now complete. This includes the IO62 firmware to run the STCCLK and STCTRG lines, along with the baud rate output clock at 9600x16Hz, and the serial loadup stream. All of these signals are wired out to RS-422 pin pairs on connector J9. The STCTRG line is basically a copy of Trigger #4 (origin 1), and the 9MHz (actually, AQ/4) STCCLK signal is gated by Trigger #5. Therefore, you have complete control over the timing and duration of these STC signals simply by selecting the start and width of those two triggers in the **Mt<n>** setup section.

   There's also an API to control the STC functions, and the complete source is available in *rvp8main/open/tdwr.c*. The IO62 firmware makes nice clean transitions between the various STC states as directed by the API.

6. The arguments to *genericBinMoments()* have changed in order to make the data from multiple receivers available to the moment calculation routines. All of the information that used to be passed is still there, just in different places. Please take a look at the first few lines of *genericSpecBinMoments()* to see how to very simply adapt any of your existing moment calculation routines.

7. The RVP8's Load-Simulated-Data command (LSIMUL) now supports loading (I,Q) data in which each sample is represented by two 16-bit floating point words having the exact same format as the packed timeseries output data that are produced by the PROC command. Compared to the legacy 4-word fixed point format (inherited from the RVP5) this new 2-word option 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 following the opcode.

8. The DSP driver has been modified to use the new LSIMUL data format whenever it is available in the signal processor at hand. Also, the opcode encoding of simulated (I,Q) values is now done entirely in the driver, so that a clean floating point (FLT4) interface is presented by the driver's *dspw_sim_pulse()* routine.

9. The **ascope** signal simulator has been modified to work more cleanly with the RVP8's new LSIMUL opcode. All of the slide pots that represent power (i.e., signal, clutter, and noise) now have a positive "Off" detent at their extreme left limit of travel. Not only does this make the menu easier to read when items are switched off, it also allows you to

operate confidently with very low simulated signal levels. It was not previously possible to accurately simulate signals in the –80dB to –65dB range due to quantization limits in the legacy LSIMUL opcode. All of this is now changed, and you may freely work with the power slide pots positioned anywhere in their operating range.

10. A new packed 16-bit timeseries format is now supported in the RVP8, and offers a 6dB improvement in SNR over the legacy format that was inherited from the RVP6/7. The new format is described in Section 6.7 of the *RVP8 User's Manual*, and can be used for both PROC timeseries output and LSIMUL timeseries input. This "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 fully recovered through a formatting trick known as "soft underflow", i.e., the mantissa is allowed to become unnormalized when the exponent is zero. A new setup question:

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

    now appears in the RVP8 **Mp** setup menu, and Bit #4 of GPARM Word #59 will be set whenever the High-SNR format is being used. The *dspr_formats()* library call also returns a new *GI4S_IQPACKHISNR* bit telling which format is selected, and the various PackIQ<–>FloatIQ routines (e.g., *vecFloatIQFromPackIQ()*) take a new flag argument.

    SIGMET recommends using the new High-SNR timeseries format for future (I,Q) record and playback applications. The 78dB SNR within the top 90dB of dynamic range, along with the additional 66dB of soft underflow dynamic range should accommodate all types of weather radar transmitters and receivers that we're likely to encounter in the future.

11. An interesting side effect of LSIMUL using the packed 16-bit (I,Q) formats is that it is now very easy to investigate the operational limits of those formats. In particular, mantissa quantization effectively fixes the SNR at approximately +72dB for the legacy format and +78dB for the High-SNR format, and this limit is independent of signal strength. This can easily be verified using the SNR readout on the **ascope** MAG-Spec display based on a pure simulated tone processed with a Blackman window. Another interpretation of the SNR limit is that the sub-clutter visibility of data stored in packed format would be bounded to approximately –72dB or –78dB respectively.

12. The *rvp8tsIQDataPackCopy()* entry point has been removed from the RVP8 Timeseries API. It was a trivial pairing of *rvp8tsIQDataFLT4()* and *vecPackIQFromFloatIQ()* that offered no real added value and was insufficiently general. Please replace:

    ```
    rvp8tsIQDataPackCopy( pTS, pPack, pHdr, iRx ) ;
    ```

    with:

    ```
    FLT4 *pIQFlt = rvp8tsIQDataFLT4( pTS, pHdr, iRx ) ;
    vecPackIQFromFloatIQ( pPack, pIQFlt, (2*pHdr–>iNumVecs), TRUE ) ;
    ```

13. The RVP8 Timeseries API now presents an entirely nonvolatile calling interface which more closely reflects the typical semantics of the driver calls. Previously, the 'volatile' qualifier appeared in front of most pulse header and (I,Q) data pointers because those data actually do reside in volatile shared memory. However, from the point of view of

writer and reader clients using the API, these data are assumed to be invariant within any given processing interval. Please remove any unnecessary 'volatile' qualifiers from your code (*gcc* will flag them for you based on the new API prototypes).

14. The MAXBINS, MAXPULSES and MINPULSES macros have been renamed and are now prefixed with 'RVP8' to make their context more clear.

15. The LFILT opcode has been changed so that it loads 3072 words (the max bin count) when the ALL bit is set in the opcode. Previously only 2048 words were loaded.

16. The RCP8 now supports the Applied Systems Model 177 TWT Transmitter via a serial interface, and new setup questions have been added to the "Site Custom" menu.

17. The RCP8 now supports the Orbit Antenna Control Unit (ACU) via a serial interface, and new setup questions have been added to the "Site Custom" menu. Since the Orbit ACU only returns position information and not velocity, you should configure the RCP section of the **setup** utility so that the Angle Insertion Source is Native RCP8. This will fill in pretty decent velocity information based on curve fitting to the positions.

    The RCP8 normally only sends a position or velocity command to the ACU whenever there is a change in either the requested AZ or EL designations. However, this could mean that the most recent command might never be seen if the ACU happens to drop that single updated packet. To prevent this, AZ and EL commands are periodically sent to the ACU (one every two seconds) even when there have been no changes since the last one.

18. The writer interface to the RVP8 Timeseries API has been changed in a number of ways to prepare it for the more elaborate control requirements of Real–time/Archive/Playback applications.

    - The *rvp8tsCreate()* entry point has been removed, and *rvp8tsAttach()* has been changed to replace the last argument with a client type argument having the choices: RVP8TS_CLIENT_READER, RVP8TS_CLIENT_RVP8MAIN, RVP8TS_CLIENT_ARCHIVE, or RVP8TS_CLIENT_NETWORK. If you call it with RVP8TS_CLIENT_READER, it does exactly what it used to do when FALSE was passed for the writer argument. Attaching as a reader still returns an error if the API does not already exist; but if you're attaching as a writer then it internally calls the old create routine(s) and returns an API to which you can eventually write.
    - Writers have a new attribute state of Active or Inactive. When initially attached, a writer is Inactive. A writer can change state with *rvp8tsSetWriterState()*. It takes a logical argument to specify active/inactive. Only one client at a time can be active on a given API unit.
    - New driver calls *rvp8tsSetDesiredWriter()* and *rvp8tsGetDesiredWriter()* have been added to select a preference for which type of client should be actively writing into the API.
    - There is a new configuration file */usr/sigmet/bin/rvp8ts.conf* that contains setup information for the Timeseries API. It is automatically created with the following default contents when you first run this new code release:

```
rvp8tsConfig.sVersion = ”8.04”
rvp8tsConfig.iDesWrMainUnit = 1
rvp8tsConfig.iDesWrApp0Unit = –1
rvp8tsConfig.iDesWrApp1Unit = –1
```

The three *iDesWr...* fields name the default desired writer client for each API unit, where the integer client numbers are taken from the *RVP8TS_CLIENT_...* definitions. The default desired writer is assigned when each API unit is first created, regardless of the client type that actually did the creating. This default assignment will remain until *rvp8tsSetDesiredWriter()* is explicitly called. The “1” that is assigned to the RVP8 “MainUnit” in the automatically created file will cause an existing RVP8 machine to simply continue working the way it always has.

19. A new TASKID opcode has been added to the RVP8 to allow the DSP driver to “name” the data that are being collected at any given time. This information then flows forward through the Timeseries API via the *rvp8PulseInfo* structures. The present ID fields can be read back via the RBACK opcode.

20. The Makefiles in the OPEN portions of the rvp8main, rvp8proc, and rcp8 trees now link with the optional environment variable ${EXTRA_LIBS}. Please add any extra libraries that you need for linking in your local environment, e.g.,

```
$ export EXTRA_LIBS=“–lMySiteLibrary”
```

21. The two polarization control lines from the RVP8 can now be attached to the trigger drivers on the IO-62 card. The following comments will be added to *softplane.conf* following a *softplane –resave* command:

```
# BNC 75–Ohm trigger drivers direct from IO62 PCI card.
# Special trigger assignments may also be used here:
#    Driver–0:  ’trigger[1]’  or  ’trigger[5]
#    Driver–1:  ’trigger[2]’  or  ’trigger[6]
#    Driver–2:  ’trigger[3]’  or  ’polar[1]
#    Driver–3:  ’trigger[4]’  or  ’polar[2]
```

These assignments are also a little more flexible than they were in the past, in that there are no longer any constraints among the pairs of signals that can be assigned to each of the four drivers.

22. Programs which include *rvp8proc.h* must now also include both *intelipp_lib.h* and *rvp8.h* . As more and more structures and prototypes in the RVP8 compute process rely on the IPP definitions, it was becoming both awkward and unnecessary to maintain the conditional compilation within *rvp8proc.h* .

23. The RVP8 now flashes the message “***== RVP8 Resetting ==***” on the front panel display for one second whenever a RESET opcode is received that performs an actual system reset, i.e., not just a noise reset or FIFO reset.

24. The following routines have been renamed to make their function more clear:

```
basicBinParams()  -->  binParamsTZVW_dflt()
basicRayFormat()  -->  rayFormatTZVW_dflt()
```

We knew that a name change was in order when one developer with a particularly fine sense of humor wrote a routine called *acidicBinParams()* .

25. The *MAX_XARGS* parameter has been increased from 256 to 1200 to provide greater flexibility for developers who implement their own RVP8 opcodes.

26. The RVP8 now supports the complete set of dual-polarization processing modes that are described in Chapter 5 of the *RVP8 User's Manual*. The RVP8 can accept dual-receiver data either as two intermediate frequencies that feed a single IFD (the traditional RVP7 frequency multiplexed model), or as two identical IFs into two separate IFDs. Thus, an existing dualPol RVP7 system can easily be upgraded to an RVP8 without requiring any changes to the radar's RF/IF circuitry. New systems can be built either way according to the preferences of the system designers.

27. The **rdasys** script has been modified so that the *start* option will now reboot the FPGA chips on all of the SIGMET PCI cards. This forces these static-RAM-based chips to be reloaded from their non-volatile configuration ROM each time the kernel module is started. The change addresses a possible intermittent problem in which the RVP8/RCP8 might not restart properly after a simple Linux reboot, or even after a power On/Off in which some residual voltage was still applied to the chassis in the form of active I/O lines. A *startNoBoot* option is also provided for developers who need to retain the old *start* behavior.

28. The GMAP clutter filter has been modified to work with the RVP8's measured system noise level, rather than a noise level that is deduced independently for each bin. This allows the filter to work with signals having very wide spectral widths whose signature hardly differs from noise itself.

Bug Repairs

1. Repaired a softplane configuration problem that would cause an 11KHz periodic waveform to appear on the TTY1 RS-232 output line whenever nothing else was assigned to it.

2. Corrected a bug that would cause the RVP8 to (sometimes) crash when the "System has an antenna" setup question in **setup–>rcp** was set to "No".

3. Repaired a bug in which the RVP8 was sometimes producing (I,Q) data at ranges beyond the point that the IFD had switched over to sampling the burst pulse. This was causing a few spurious bins to sometimes appear just before the maximum range.

4. The WSR-88D DCU Self-Test-2 packet is now expected to be two (not four) bytes long. The DCU simulator that is built into the RCP8 has been changed accordingly, and the RCP8 BITE packet for these data is now 6-Bytes (not 8-Bytes) long.

5. The bandwidth of the amplitude modulation of Tx envelopes that are synthesized by the RVP8/Tx card was not being set properly in the SETPWF command. On systems using more than one pulsewidth, this would cause **ascope** to generate incorrect waveforms for

certain pulsewidth selections. The waveforms were correct, however, when selections were made via the internal setup menus. This bug has been present since the RVP8/Tx card was introduced.

6. Fixed SIGSEGV fault when *rvp8tsRelease()* was called by a reader client.