RDA 8.01.14 Release Notes (26 Aug 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-8.01.12 generated on 12 August 2003. If you are upgrading from an earlier release, please read those notes also.

Release Changes

1. Added the following files to the rda release: bin/audiox, bin/audio, and bin/sigxbell. This prevents a warning message each time **bitex** is started.

RVP8 New Features

1. Several new features have been added to the RVP8 TimeSeries API.

    - It is now possible to have several timeseries API "units" running simultaneously on a given machine. This is an important step toward supporting standalone applications that import/export timeseries data in a unified manner.
    - The "writer side" of the API is now fully broken out so that applications other than the RVP8 can create and manage their own APIs. Please see documentation for *rvp8tsCreate()*, *rvp8tsAllocateDataAreas()*, and *rvp8tsInsertPulseHdr()*.
    - The arguments to *rvp8tsAttach()* have changed slightly, and the API no longer supports the obscure ability to manually supply the mapped areas. To attach to the standard RVP8 timeseries as a reader, please change your code to:

      ```
      TS = rvp8tsAttach( &istatus, RVP8TS_UNIT_MAIN, FALSE ) ;
      ```

    - The API unit that holds the data for each receive channel does not need to be the same as the unit that actually presents those data. This allows sharing of data across Rx domains without the need for any memory copies.

2. The red LED on the inside top edge of the RVP8/Rx PCI card now flashes when there is PCI bus activity to that card. This allows you to "see" the bursts of (I,Q) data as they are transferred off the card.

3. There's now a new Appendix F in the *RVP8 User's Manual* which documents the RDA developer's environment. This appendix is currently a "work in progress".

4. The Intel IPP library covers in `rda/intelipp` have been added to the developer's source.

5. The RVP8 and RCP8 executables are now built from the OPEN portion of the tree, rather than from the CORE area. This gives developers much more control over their execution environment, e.g., for doing profiling and debugging.

6. The RVP8's windowed FIR matched filter design software is now available to developers in `rvp8proc/open/if_filter.c`. Also the new function pointer *matchedFilterCoefs()* can be used to completely customize how the filters are built in each major mode.

7. A new "@" command has been added to the RVP8 menus to allow you to view or change the current major mode of the processor. This is handy when you want to test the *init* and *exit* code of new major modes.

RCP8 Bug Repairs

1. Repaired a bug that was introduced in the 8.01.11 release that would cause the antenna positions to get stuck for a few seconds when crossing zero degrees.

RDA 8.01.12 Release Notes (12 Aug 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–8.01.7 generated on 7 July 2003. If you are upgrading from an earlier release, please read those notes also.

RVP8 Bug Repairs

1. The block of IPCS shared memory identified as *rvp8_ctshare* was not being freed properly when the RVP8 would exit. This could cause the RVP8 to not restart properly following an RDA upgrade in which no reboot was done in between.

2. Isolated softplane *true* and *false* output level assignments would sometimes not be setup properly upon startup.

3. Improvements have been made in the handling of the internal real–time control callback function *rtCtrlCBF()*. It is now possible on each invocation of the callback to request a different clock source for each of the available timers. Previously, spurious counts would sometimes be registered when clock sources were changed at any time other than the very first call.

4. A second and related timer bug was also repaired in which the callback function would sometimes be invoked with none of the *lTimerFired* fields set to *true*. The root cause was a hardware FPGA clock–domain bug, so you will need to reflash all of your PCI cards to repair it. Code has also been added to the RVP8 to signal this condition as an internal error -- hopefully, you will never see this signal.

5. A documentation error was repaired for the *iBurstArgs* array in *struct rvp8PulseHdr*. These binary angles represent the burst phase of the previous pulse minus that of the current pulse. The previous definition had the order (hence sign) reversed. No changes have been made in the computation of these angles, so any code that already uses them can remain “as is”.

6. Random Phase (RPH) mode second-trip signal recovery was not working properly due to the documentation error described in #5. Although the RPH algorithms were working properly with simulated data from *ascope*, they produced incorrect results on live radar data from the RVP8/IFD. This discrepancy accounts for why it took us several weeks to finally understand the problem and track it down.

7. An obscure race condition was repaired in the coordination of the RVP8 compute threads that would produce very infrequent missing halves of parameter rays and/or spurious error messages on the RVP8 startup console.

8. The data from the internal ray band simulator (enabled via the **M+** menu) were not being output properly when “archive” data format was selected for the PROC command. The simulated data were correct in all other output modes.

RVP8 New Features

1. The maximum number of range bins allowed within the RVP8 has been increased from 2048 to 3072. This change only affects third-party developers who program the RVP8 directly; the IRIS bin limit still remains at 2048.

2. Support for dedicated real–time signals on the MIT/LL prototype TDWR panel is now more fully integrated into the RVP8. A new backpanel type:

   ```
   splConfig.Io62[0].sExtPanel = "TDWRV1"
   ```

   can now be selected in your softplane.conf file. Selecting this option, and then running "softplane –resave" will create blank stanzas for all of the signals that are accessible via C code. Pins that are not shown in the file have the following hardwired assignments:

   - J9–1,14 RS-422 Trigger #1 output
     J9–2,15 RS-422 Trigger #2 output
     J9–3,16 RS-422 Trigger #3 output
   - J9–4,17 RS-422 STC trigger output
     J9–5,18 RS-422 STC clock output
     J9–6,19 RS-422 STC data clock output
     J9–7,20 RS-422 STC data output
   - J3–5,18 RS-422 Antenna Reference Pulse (ARP) input
     J3–6,19 RS-422 Antenna Change Pulse (ARP) input

   Please save a copy of your old configuration file prior to making this conversion, because the "–resave" operation will install blank stanzas the very first time it is run. The new file contents will look very similar to a standard IO62CP backpanel, and with some minor editing you can convert your old stanzas into new ones.

3. The association that is maintained between Linux system time and RVP8/Rx acquisition clocks is now more tolerant of minor frequency differences between the two. Previously, the RVP8 would report *Rx clock skew* if the normalized frequency error was more than 0.0001 (one part in ten thousand). This required that the Linux time be accurate to eight seconds per day, which could not always be guaranteed. The new threshold is 0.0007, corresponding to one minute of error per day.

4. The RVP8 now performs a "courtesy update" of the softplane control output lines. This is for the benefit of user code that may have made changes to "slow" output bits and did not want to bother updating them right away via *splLogicCntPhysIO()*.

5. The RVP8 now makes more graceful tradeoffs when generating out-of-spec internal triggers, i.e., trigger patterns from the **Mt<n>** menus that do not fit within the current trigger period. The issue here is that the defined trigger patterns may extend to far negative or positive times, and the overall span of the pattern may be greater than the desired PRT. This is now handled as follows:

   - All triggers that are active during negative times (before range zero) are always generated exactly as defined, even if that requires increasing the PRT to fit them.

For example, if one of the user triggers has a start time of –500µsec, then PRF requests above 2KHz (approximately) will be bounded so that this trigger can be properly output. The idea is that negative triggers most likely represent system pretriggers with important timing relationships that must be preserved.

- On the other hand, triggers with positive start times will be truncated in an attempt to preserve the requested PRT, i.e., priority is now given to providing the exact PRT rather than the exact trigger pattern. The idea is that positive triggers probably are strobes and markers that do not directly affect the operation of the radar transmitter.

Two bits have been added to GPARM Immediate Status Word #4 (Output word #59) to indicate when the RVP8 is running with an altered trigger pattern, altered PRT, or both.

6. New command line option **–showRTCtrl** causes the RVP8 to print diagnostic messages showing activity of its real–time callback timers. This can be handy when debugging your custom callbacks.

7. Bits were added to the CFGHDR (Configure Ray Header) command so that each ray can now include a complete *gparm* structure, as well as a miscellaneous status bit word.

8. The SNOISE (Sample Noise) command is now able to set the RVP8 noise levels, in addition to measuring them as it has always done. This allows user code to install a trusted noise level for each pulsewidth, rather than always having to run with the most recently measured value.

RCP8 New Features

1. The RCP8 now has the ability to reject AZ/EL position samples that are “momentarily” incorrect. The incoming position data are filtered so as to remove any measurement that would imply an antenna velocity greater than the maximum allowed speed. In other words, if the antenna could not possibly have moved to a newly sampled position from its last trusted (Time, Position), then that new angle sample is discarded. The result is that normal valid position data are accepted “as is” without being modified in any way; but momentary jumps in position are discarded.

   Note that an abrupt jump in position *will* eventually be accepted as a new and trusted measurement, but only after it has persisted long enough that the antenna could have gotten there by moving at its maximum velocity from its last trusted location.

   The setup parameters that affect the operation of this filter are:

   - The tachometer calibration (Level, Speed) from the **axis** menu, along with the maximum velocity from the **veservo** menu. Together, these define the maximum angular velocity of the antenna in degrees/sec. The velocity threshold used by the filter is then set 20% higher than this value to safeguard against false hits.
   - The number of bits for angle input from the **axis** menu. This parameter is needed to distinguish erroneous position hops from those that are merely due to normal angle quantization.

There are no additional setup questions required to configure this “Pop Angle” filter; it gets all of its setup information from existing menu questions, and is always enabled on both axes.

IRIS Bug Repairs

1. Several bugs were repaired in ASCOPE’s generation of simulated phase sequences. The most observable error was that the sequences were discontinuous at the boundaries between groups of pulses that contribute to each ray.

IRIS New Features

1. The **ascope** utility now reports the altered trigger pattern and altered PRT status bits as described above. If you suddenly begin seeing these warnings, it may be a clue that your trigger definitions are inconsistent with the PRFs being used.

RDA Bug Repairs

1. A fairly serious Linux bug has been present in all recent 2.4.18+ SMP kernels, in which PCI configuration space bus cycles would not work reliably on dual-processor machines. We have recoded our PCI cards and kernel module to work around this problem, and expect that some of the intermittent RVP8/RCP8 crashes that have been reported will likely be resolved. Please let us know what you find.

RDA New Features

1. New test/debug utility ***rtnice*** *priority pid* can be used to set a given real–time priority and SCHED_RR policy for the indicated process ID. This is not an operational utility; it is only intended for software developers.

RDA 8.01.7 Release Notes (7 Jly 2003)

We're very happy to provide this first set of 'official' release notes covering changes made to the RDA (Radar Data Acquisition) software including primarily the RVP8 and RCP8. The last release was RDA–8.01.2 generated on 26 May 2003.

Important: Beginning with this release, RCP8 systems using tachometer analog inputs will only work with Rev.B and higher backpanels. If you are running an RCP8 with a Rev.A backpanel and tachometer inputs, SIGMET will be happy to upgrade your panel at no cost. Please do not install this software release until you actually have your new backpanel in hand.

RVP8 Bug Repairs

1. When spectra were being computed within the RVP8, the data window selection (Hamming, Rectangular, etc.) would sometimes not be set properly from ASCOPE.

2. The *iTgWave* field of the RVP8 timeseries pulse headers was not incrementing properly. It represents the trigger waveform sequence number within a bank of trigger waveforms, but was stuck at zero.

3. Minor bug fixed for PROC spectral output in which zero power terms were expressed as 0x0000 rather than 0x8000. It is almost impossible to actually exhibit this bug on real data.

4. A bug was repaired in the UIQBITS opcode wherein stale elements could sometimes clog the internal UIQ FIFO. The problem could arise if UIQ bits got written several tens of seconds prior to the PROC command that begins the actual (I,Q) data acquisition.

RVP8 New Features

1. GPARM output word #59 is now a fourth immediate status word. Bit zero indicates that the RVP8 has been configured to use continuous spectra sizes (DFTs versus FFTs) for internal parameter computations whenever possible. In other words, the spectra size will match the sample size exactly, rather than being constrained to the greatest power of two that is less than or equal to the sample size.

   Bit one of the new GPARM word indicates that any size spectrum can be output as well. This bit will only be set if the user has answered *YES* to the Mp setup question *Allow continuous sizes for power spectra*, and if Bit-10 of the SOPRM flag word (also a new bit) is set, indicating that continuous sized output spectra are desired. Also, ASCOPE is now able to plot any size power spectrum from RVP8 data.

2. Two new data window types, Exact Blackman and VonHann, have been added to the RVP8 and to dsp.h. An additional bit is now used in SOPRM word #10 to hold the 3-bit window code that used to be only two bits wide. The UVD bit, which thus far is unused in IRIS, has been moved to Bit-13 of that word to make room.

3. The RVP8's maximum sample size has been raised from 256 to 1024, with a corresponding change also made to ASCOPE.

4. The maximum amount of time that the RVP8 is willing to wait when trying to assemble the next Coherent Processing Interval (CPI) is now the greater of one second, and 1.2 times the expected time to acquire the present number of pulses at the present PRF. Previously only the 1-second timeout was used, which might not be long enough when very long CPIs (e.g., 1024 pulses) are being collected. The 1-second component of the timeout is really most important during angle syncing, when the rate of finding new CPIs is determined entirely by angle rotation rate, and not at all by PRF.

5. The default timeout of the interface between the IRIS DSP driver and the signal processor to which it is attached has been increased from four seconds to six seconds. This is to help when acquiring long pulse sequences at low PRFs.

6. The Random Phase (RPH) major mode is now fully implemented for second-trip echo recovery. A new statistical whitening filter has been incorporated that is much more effective at removing multi-mode coherent power than was previously done in the RVP7. This filter will be invoked whenever the transmit phase sequence is random. Note that RPH mode also handles SZ(8/64) systematic phase codes using a completely different whitening and reconstruction filter. The RVP8's present implementation of SZ(8/64) is only intended for research evaluation, pending refinement of the techniques and clarification of licensing issues.

   One particularly nice feature of RPH mode in the RVP8 versus RVP7 is that it now works equally well with either an internal or external trigger. The first/second trip bins are sorted properly in both cases, whereas RPH mode in the RVP7 would only work from an internal trigger.

7. The default value of the *Minimum freerunning ray holdoff* in the **Mp** setup menu has been changed from 100% to 50%. This starting value will typically be more useful when tuning up overlapped freerunning performance in your system.

8. A new function *genericBinMoments()* in *rvp8proc/open/binmoments.c* has been introduced to coordinate the calculation of autocorrelation moments for all major modes. In that same directory, support routines for many spectral-based algorithms have been moved to *spectra.c*, and code for phase modulated modes can be found in *phasemod.c*.

9. Error signals generated by the RVP8 are now tagged with the time and date.

10. The programmer's model for RVP8 realtime callback timers has changed slightly. The two previous RTCBTV_ACP and RTCBTV_ARP macros have been replaced with three general purpose input selectors RTCBTV_IOSHR0/1/2. These select one of three shared status inputs from the I/O-62 card as the counter/timer clock sources.

    The assignment of physical I/O lines to these shared clocks is made by choosing from a fixed list of hookup models in softplane.conf. Typing *softplane –resave* will cause the following lines to appear, or you may type them in yourself:

```
# Hookup model for shared status/counter inputs.  Options are:
#   MIT/LL-TDWR-V1 : MIT/LL Custom TDWR patchpanel, Ver.1
#
splConfig.Io62[0].Opt.Cp.SharedStsModel = ""
```

Choosing *MIT/LL-TDWR-V1* as the model will configure the I/O-62 card and backpanel for the ACP and ARP assignments assumed by the prototype MIT/Lincoln TDWR patch panel. ACP arrives on *splConfig.Io62[x].Opt.Cp.J3_06_19.pinPos*, and attaches to RTCBTV_IOSHR2; ARP is on the corresponding ...*J3_05_18*... pin, and attaches to ...SHR1. These hardware lines should be configured as RS-422 inputs by setting the *lRS422* and *iTerm* fields to '1', and attaching some free *sAux[]* bits to the lines themselves. This will also allow you to monitor the ACP/ARP line levels in those auxiliary status bits.

WSR88D RVP8/RCP8 Updates

1. The polarity of the receiver protect command and response signals has been inverted for the past year due to an error in translating the pin assignments for the ORDA backpanel. This has been fixed in firmware.

2. The trigger control firmware has been completely rewritten to handle the required startup, shutdown, and test modes in a fully general and interlocked manner.

   - The Radiate-On sequence always begins with a MOD_CHARGE trigger.
   - The Radiate-Off sequence always ends with a TRIGGER_CHARGE and MOD_DISCHARGE trigger.
   - The RFD test mode is supported in which the RF_PULSE_START, RF_GATE, and RF_DRIVE triggers are present.
   - The CW test mode is supported in which RF_GATE is held active.
   - Within any of the above modes, the RX_PROT_CMD trigger can optionally be held active to protect the receiver at all times, not just when transmitting.

   The Radiate-On, RFD-Test and CW-Test modes can only be entered when Radiate is OFF, i.e., all transitions between these three states will always pass through the Radiate-Off state. If one particular mode is in effect, the request for that mode must be removed before any other modes can be entered. This helps insure that the radar hardware are protected against bad sequencing requests.

   To use these new features, please add the following to the RCP88D section of softplane.conf:

```
splConfig.Io62[0].Opt.Rcp88d.TxB2B.bit00 = "cAux[29]"
splConfig.Io62[0].Opt.Rcp88d.TxB2B.bit01 = "cAux[26]"
splConfig.Io62[0].Opt.Rcp88d.TxB2B.bit02 = "cAux[25]"
splConfig.Io62[0].Opt.Rcp88d.TxB2B.bit03 = "cAux[24]"
splConfig.Io62[0].Opt.Rcp88d.TxB2B.bit04 = "true"
splConfig.Io62[0].Opt.Rcp88d.TxB2B.bit05 = "cAux[49]"
```

```
splConfig.Io62[0].Opt.Rcp88d.RxB2B.bit00 = ”sAux[29]”
splConfig.Io62[0].Opt.Rcp88d.RxB2B.bit01 = ”sAux[26]”
splConfig.Io62[0].Opt.Rcp88d.RxB2B.bit02 = ”sAux[25]”
```

Control bits C29/C26/C25 request Radiate/RFD/CW modes respectively, and the mode that is actually running (after qualifying the requests) is reported back in S29/S26/S25. At most one of the status bits will ever be active at once, even if multiple and/or contradictory requests are being made. C24 is a request to hold the receiver protect command active at all times. If the transmitter is radiating it will continue to run normally, but only as long as the receiver protect response also remains active. Lastly, C49 forces the PHCOHOSEL control line to the state mapped into `TxB2B.bit04`. This control line will eventually have some other default behavior, but for now it can at least be forced On/Off from the RCP8.

3. Several improvements were made to the WSR88D DCU serial thread:

   - The RCP8 is now much more tolerant of out-of-sequence responses to its BIT and SelfTest packet requests. Previously, whenever the RCP8 would request those data from the DCU it expected them to arrive immediately following the next antenna record. However, observations with a real DCU show that there is sometimes a delay of one or even two packets intervals. The new algorithm will accept BIT/ST replies within 350ms of the time they were requested, and it will never queue more than one such request at a time. The one-at-a-time constraint is necessary because the DCU reply packets can not necessarily be distinguished from each other solely from their content.
   - The RCP8 DCU simulator has been modified to introduce a random one or two packet delay in its response to BIT requests. This was done to test the above new feature.
   - The SelfTest #1 requests from the RCP8 now consist of a walking ones pattern that cycles through the four command bytes that will be echoed by the DCU. The special BIT and AZ/EL bits retain their meaning, hence the walking cycle repeats with a period of 28 (32 bits in the four bytes, minus four bits having fixed-meaning).

RCP8 New Features

1. The “help view” command is now implemented at the top-level RCP8 prompt. An example printout is shown below.

```
RCP> help view
Board Configuration and Status
------------------------------
RCP8 Radar Control Processor V2.1
  Settings were last saved using V2.1
  RCP8 started at: 21:53:06 31 MAY 2003
  Current time is: 21:53:10 31 MAY 2003

Physical hardware inventory:
```

```
  Found PCI Card I/O-62  - Rev.B  Serial:1841  Code:12 (/dev/rda/
io62-0)
   \--> IO62CP Backpanel - Rev.B  Serial:1822  Code:3

Parallel execution threads:
     CS-Tick - PID:20454  tId:1026
      Servos - PID:20455  tId:2051
    Watchdog - PID:20456  tId:3076
    Host-RCV - PID:20457  tId:4101
    Host-XMT - PID:20458  tId:5126

Shared library build dates:
  RCP8/Core: Sat May 31 20:54:52 EDT 2003
  RCP8/Open: Sat May 31 20:54:45 EDT 2003
  RCP8/Site: Sat May 31 20:54:45 EDT 2003

AZ Axis - Pos:   0.00  Vel:  0.0  Synchro Ref: 0% Syn: 0% (FAULT)
EL Axis - Pos:  23.81  Vel:  0.0  Synchro Ref:34% Syn:78%
```

The code version, start time and current time are first listed, followed by an inventory of all SIGMET hardware (PCI cards and backpanels) being used. The process IDs, thread IDs and names of all parallel execution threads is shown, along with the build dates of the Core/Open/Site RCP8 shared libraries. Summary information about each antenna axis is also printed. If synchro inputs are being used, then both the reference voltage and S1/S2/S3 voltages are shown as a percentage of full scale A/D value. A synchro will be faulted if either of these levels drops below 10% or rises above 95%.

2. The RCP8 now implements 3-wire synchros as an optional method for measuring both position and velocity. The synchro voltages for both AZ and EL are applied to the 12-pin Molex connector on the IO62/CP backpanel. This connector uses the same wiring that was used for the synchro inputs to the old RCP02; so, for upgrades, you can simply move your existing cable from one to the other.

   The Synchro-to-Digital (S/D) conversion is implemented entirely in FPGA code on the I/O-62 card, and in software running in the RCP8. No additional hardware is required to begin using synchro inputs on your system. New setup questions in the “Axis” command might be set as follows:

```
Angle input signal source: Synchro
  Synchro reference frequency: 60 Hz
  Shutdown for invalid synchro voltages: YES
Angle offset from true orientation:   0.00 deg
Use tachometer voltage to estimate velocity: NO
  Synchro Tach -- Full scale speed:  80.00 deg/sec
```

   The first two questions establish that a 60Hz synchro will be used for angle (position) input on this axis. Moreover, the voltages on the two “Ref” and three S1/S2/S3 lines will be checked for validity, and the RCP8 will shutdown if these voltages drop below 10% or rise above 95% of full-scale A/D values. Note that the present voltage levels can be checked in the “Help View” menu. The angle offset from true orientation is set to zero in this case, but you can use it to null out any fixed position error.

   When synchros are used for position input you can still use tachometers for velocity input in the usual way. However, if tachometers are not available, an excellent

alternative is to use the velocities that are generated by the S/D conversion process itself. The S/D converter is implemented as a Type-II tracking servo that provides zero position error at any velocity whenever the acceleration is zero. The internal velocity that is maintained during this process can be used by the RCP8 in place of a physical tachometer. When doing this, you choose the velocity that will correspond to 100 T-Units of virtual tachometer level. Simply choose an upper bound that is equal to the fastest spin rate you ever intend to use.

The following table lists the maximum RMS voltage that can be applied to the backpanel's Molex SYNCHRO connector for each value of plug-in SIP resistor. The AZ channel voltages are set by SIP S1, whereas S2 sets the EL voltage levels. These resistors are socketed, and can be changed by removing the back cover of the IO62–CP panel

S1 or S2	Max Ref(RMS)	Max S–S(RMS)
47K	56V	31V
68K	81V	45V
100K	118V	66V
150K	178V	99V
220K	261V	145V

Note that the 'Ref' inputs have somewhat lower gain than the three 'S' inputs. This is because the precision of the S/D angle conversion is affected primarily by the precision at which the three 'S' voltages can be measured. The backpanel therefore biases the gains so that the 'S' voltages can be made as large as possible, i.e., without the 'Ref' voltages first filling the A/D conversion range.

The appropriate resistor is the smallest value such that the maximum S-to-S voltage of the synchro (which is angle dependent) still fits within the table range. The reference voltage should then fit easily into its corresponding maximum range. Don't worry if it doesn't; the important thing is to match the 'S' line voltages.

For example, a traditional 90Vrms 1:1 synchro would best use the 150K resistor, whereas a 105Vrms unit would require the 220K value. Note that you can check for proper A/D conversion levels of the synchro inputs using the 'help view' menu of the RCP8.

Important: The synchro voltage input feature is only available on Rev.B and higher backpanels. If you are running an RCP8 with a Rev.A backpanel and would like to switch to synchro inputs, SIGMET will be happy to upgrade your panel at no cost.

3. The RCP8 now supports the serial protocol for the Kavouras TCU (Transmitter Control Unit).
The following questions have been added to the 'site custom' menu:

```
Kavouras TCU Interface (Radiate/BITE): YES
  Serial port: /dev/ttyS0
  ID of TCU standard BITE packets: 0x09
  ID of Quantitative BITE packets: 0x0A
  Simulator port: /dev/ttyS1
```

Simply choose the serial port, and ID for the BITE and Q–BITE packets that will be associated with the TCU. Note that the serial baud rate, parity, and stop bits are fixed at 9600/Odd/2 because they are fixed by the TCU. The built-in simulator can be used for debugging the main code, and you can watch the live I/O from a real TCU using the 'Monitor SIO' command followed by 'Raw rTcu'.

The standard BITE packet for the TCU is 13 bytes long, and maps the 64 TCU status bits into the first 64 packet bits. The timeout bit (no TCU communication) appears in the MSB of Byte #12. These seventy bits of BITE status (10 words of 7 bits apiece) are mapped into status bits S64–S133. If you want to use any of the TCU status bits in a logic equation, those are the variables to grab.

The Q–BITE packet is 27 bytes long and holds 12 14–bit values. The first two are the 'Max strike' and 'Current strike' counts from the TCU status packets, and the next eight are from the temperature report. The last two slots (11 and 12) are unused for now.

The TCU is reset using the standard BITE resetting mechanism. A BITEX reset that is directed at either the BITE or Q–BITE unit will send a reset command to the physical TCU. In addition, a rising edge on Control Variable C63 will also reset the TCU.

The TrPower and Radiate control/status bits are the only ones needed for the TCU, giving you the states OFF, STANDBY, and RADIATE. When you twiddle those two control bits the appropriate commands will be sent to the TCU. Likewise, status from the TCU will set the TrPower and Radiate status bits appropriately.

4. The RCP8 can now apply drive compensation to an antenna that is not mechanically balanced. The result is that even a badly unbalanced axis can be properly stabilized without requiring any readjustment of the antenna counterweights. The new setup questions in the 'axis' menu are:

```
Use drive compensation for unbalanced antenna: YES
  Neutral droop position of antenna:  35.0 deg
  Drive required 90–deg off neutral:  51.4 D–Units
```

The model being assumed here is that the center of mass of the unbalanced antenna is offset some distance from the axis of rotation. Thus, when no other forces are applied, the antenna will tend to droop to some neutral angle that puts that center of mass directly below that axis. In the above example, the neutral droop angle is 35 degrees, i.e., no motor drive is required to 'hold' the antenna at that position.

Once we know the neutral droop angle *N*, the drive that is required to compensate for the imbalance when positioned at some angle *P* is simply $D \sin(P–N)$, where *D* is the drive that would be required to hold the axis 90-degrees away from the neutral point. The second setup question asks for that value *D*.

As an example, suppose we have the unbalanced antenna mentioned above that naturally returns to 35 degrees whenever no drive is applied. We wish to compensate for this, and have manually determined that a drive level of –28 D-Units is required to hold the antenna down when it is positioned at two degrees. The drive that would be required at 90 degrees offset is therefore $-28 / \sin(2–35)$ = 51.4 D-Units, which we then enter into the second setup question.

5. In the “site misc“ menu you can now choose the duration of the unsafe interval following the release of any shutdown condition. Previously this had been hard coded as one second. There has also been some confusion about using the **cReset** and **sReset** variables. Think of the RCP8 ’reset module’ as a black box, capable of resetting the RCP8 from shutdowns, and having the following inputs and outputs:
    - INPUT: A reset sequence can be initiated from the host computer serial line, e.g., via the RESET button in the ANTENNA utility.
    - INPUT: The **sReset** status line can also cause a reset sequence to begin on its rising edge. If **sReset** is hooked to an external status input via softplane.conf, then that external input will trigger a reset. If **sReset** is modified in logic equations, then the assigned value will be what causes the reset to take place (again, on its rising edge).
    - OUTPUT: Whenever a reset sequence begins for any reason, **cReset** is a control output that goes high for exactly one second. In other words, **cReset** is merely an output flag that is produced by the reset module. You can ignore it entirely, or hook it to some other hardware that also needs to be informed when a reset is occurring.
6. Error signals generated by the RCP8 are now tagged with the time and date.
7. The serial data stream diagnostic printout from the *Raw* command within *Monitor SIO* are now formatted with a newline whenever there is a change in the stream source. This makes the data much more readable when multiple streams are being displayed.

IRIS New Features

1. Many changes have been made to ASCOPE, making it much more powerful as a signal modelling and analysis tool.
    - When ASCOPE starts up it negotiates with the DSP to determine if continuous spectra sizes will be supported. If both agree, then any size spectra can be requested and processed, not just powers of two. Also, when specta are computed internally from timeseries, ASCOPE is now capable of computing DFTs (rather than FFTs) of any length.
    - The maximum sample size has been raised from 256 pulses to 1024 pulses.
    - The MagSpec plot now shows peak and median spectrum power, as well as the SNR of the signal according to the R1/R2 estimator.
    - Addition of Exact Blackman and VonHann windows.
    - Repaired bug in which ASCOPE would crash if A/D data were requested along with timeseries data. Also repaired many other minor bugs.
    - The parameter plots now provide printouts of mean and standard deviation of each scientific parameter. These numbers will be displayed whenever the new ’–stats’ startup flag is given, or whenever simulated data are being used.

- The data simulator now provides separate timeseries for every bin that is being processed. Along with the statistics mentioned above, this provides a very convenient way to study the bias and uncertainty of processing algorithms on simulated data.
- The width of the simulated data can now be set in increments of 0.001 (normalized units), rather than 0.01.
- When both dBZ and dBT are displayed, the dBZ window will contain a field showing the mean clutter suppression, averaged over all bins in the ray.

Anticipated New Features

The following items did not quite make it into this release, but are expected soon.

- Entry points to supply ancillary data in the RVP8 TimeSeries API. This would include, for example, the current range mask and other housekeeping information.
- Refinement of RVP8 spectral clutter filters, specifically in preparation for evaluation as alternates to legacy time-domain filters.
- Ability to select playback spectrum in ASCOPE.

Open Issues

1. When the RVP8 or RCP8 is running in the background and the X-Server is started via **xinit** or **startx**, it is fairly likely that interrupts from the SIGMET PCI cards will become disturbed in such a way that the RVP8/RCP8 must be restarted. We've added additional kernel logging (to */var/log/messages*) to try to diagnose this problem, but it has not yet been resolved. There are a few recommendations for workarounds:
   - Generally the RVP8/RCP8 are black box "appliances" and will not even run X. In these cases there is no problem at all.
   - If you want to run X, then please rearrange your startup scripts so that the RVP8 or RCP8 are started after X starts, rather than before.
   - The bug is only present on SMP (multi-processor) kernels. If you run a non-SMP kernel you can freely start/stop X whenever you like; but you will only be running with one CPU.

Note: This has been repaired as of 7 August 2003