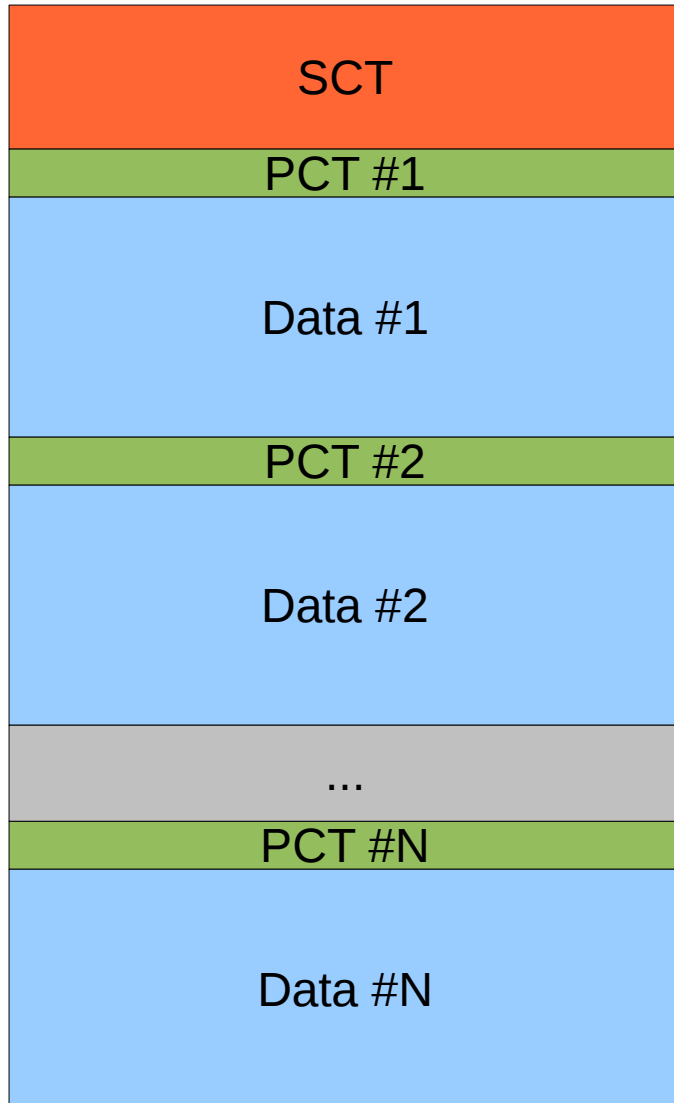


RIQ Data Format

- VIPIR data: Raw In-phase and Quadrature



- SCT: Sounding Configuration Table
 - Instrument settings (output)
 - Instrument control (input)
- PCT: Pulse Configuration Table
 - Pulse-to-pulse variable settings
- Data Block
 - 16 bit values
 - In-phase and Quadrature
 - 8 receivers
 - Requested number of range gates

Sounding Configuration Table

- Binary object used to control the radar and record the instrument settings
- C or FORTRAN structure

```
! Top level Sounding Configuration Table, Version 1.2
!  
TYPE :: SCTtype  
INTEGER(KIND=4) :: magic           ! magic number 0x51495200  
INTEGER(KIND=4) :: sounding_table_size ! bytes in sounder configuration structure (this file)  
INTEGER(KIND=4) :: pulse_table_size  ! bytes in pulse configuration structure  
INTEGER(KIND=4) :: raw_data_size     ! bytes in raw data block (one PRI)  
REAL(KIND=4)    :: struct_version    ! Format Version Number.  Currently 1.2  
INTEGER(KIND=4) :: start_year        ! Start Time Elements of the ionogram (Universal Time)  
INTEGER(KIND=4) :: start_daynumber   !  
INTEGER(KIND=4) :: start_month       !  
INTEGER(KIND=4) :: start_day         !  
INTEGER(KIND=4) :: start_hour        !  
INTEGER(KIND=4) :: start_minute      !  
INTEGER(KIND=4) :: start_second      !  
INTEGER(KIND=4) :: start_epoch       ! epoch time of the measurement start.  
CHARACTER(128)  :: readme            ! Operator comment on this measurement  
INTEGER(KIND=4) :: decimation_method ! If processed, 0=no process (raw data)  
REAL(KIND=4)   :: decimation_threshold ! If processed, the threshold value for the given method  
CHARACTER(128) :: user               ! user-defined  
  
TYPE(STATIONtype)  :: station      ! Station info substructure  
TYPE(TIMINGtype)   :: timing       ! Radar timing substructure  
TYPE(FREQUENCYtype) :: frequency   ! Frequency sweep substructure  
TYPE(RECEIVERtype) :: receiver     ! Receiver settings substructure  
TYPE(EXCITERtype)  :: exciter      ! Exciter settings substructure  
TYPE(MONITORtype)  :: monitor      ! Built In Test values substructure
```

SCT Station

```
TYPE :: STATIONtype
CHARACTER(64)  :: file_id      ! name of station settings file
CHARACTER( 8)  :: ursi_id      ! URSI standard station ID code
CHARACTER(32)  :: rx_name      ! Receiver Station Name
REAL(KIND=4)  :: rx_latitude  ! Position of the Receive array reference point [degrees North]
REAL(KIND=4)  :: rx_longitude !                               [degrees East]
REAL(KIND=4)  :: rx_altitude  ! meters above mean sea level
INTEGER(KIND=4) :: rx_count    ! Number of defined receive antennas
CHARACTER(32),DIMENSION(32) :: rx_antenna_type ! Rx antenna type text descriptors
REAL(KIND=4),DIMENSION(3,32) :: rx_position    ! X,Y,Z = (East,North,Up) Positon [m] of each Rx
REAL(KIND=4),DIMENSION(3,32) :: rx_direction  ! X,Y,Z = (East,North,Up) Direction of each Rx
REAL(KIND=4),DIMENSION(32)   :: rx_height     ! Height above ground [m]
REAL(KIND=4),DIMENSION(32)   :: rx_cable_length ! physical length of receive cables [m]
REAL(KIND=4)   :: frontend_atten ! Front End attenuator setting
CHARACTER(32)  :: tx_name        ! Transmitter Station Name
REAL(KIND=4)  :: tx_latitude    ! Position of the Transmit Antenna reference point [degrees North]
REAL(KIND=4)  :: tx_longitude   !                               [degrees East]
REAL(KIND=4)  :: tx_altitude    ! meters above mean sea level
CHARACTER(32)  :: tx_antenna_type ! Tx antenna type text descriptors
REAL(KIND=4),DIMENSION(3) :: tx_vector ! tx antenna direction vector [m]
REAL(KIND=4)   :: tx_height      ! antenna height above reference ground [m]
REAL(KIND=4)   :: tx_cable_length ! physical length of transmit cables [m]
INTEGER(KIND=4) :: drive_band_count ! Number of antenna drive bands
REAL(KIND=4),DIMENSION(2,64) :: drive_band_bounds ! drive bands start/stop in kHz
REAL(KIND=4),DIMENSION(64)   :: drive_band_atten ! antenna drive atteunuation in dB
INTEGER(KIND=4) :: rf_control  ! -1 = none, 0 = drive/quiet, 1 = full, 2 = only quiet, 3 = only atten
CHARACTER(32)   :: ref_type    ! Type of reference oscillator
CHARACTER(32)   :: clock_type  ! Source of absoulte UT timing
CHARACTER(128)  :: user        ! Spare space for user-defined information
END TYPE STATIONtype
```

SCT Timing

```
! Timing of the measurement
TYPE :: TIMINGtype    ! Time values are in microseconds unless otherwise indicated
CHARACTER(64)        :: file_id          ! Name of the timing settings file
REAL(KIND=4)         :: pri              ! Pulse Repetition Interval (PRI) (microseconds)
INTEGER(KIND=4)      :: pri_count        ! number of PRI's in the measurement
INTEGER(KIND=4)      :: ionogram_count   ! repeat count for ionogram within same data file
REAL(KIND=4)         :: holdoff          ! time between GPS 1 pps and start
REAL(KIND=4)         :: range_gate_offset ! true range to gate 0
INTEGER(KIND=4)      :: gate_count       ! Number of range gates, adjusted up for USB blocks
REAL(KIND=4)         :: gate_start       ! start gate placement [us], adjusted
REAL(KIND=4)         :: gate_end         ! end gate placement [us], adjusted
REAL(KIND=4)         :: gate_step        ! range delta [us]
REAL(KIND=4)         :: data_start       ! data range placement start [us]
REAL(KIND=4)         :: data_width       ! data pulse baud width [us]
INTEGER(KIND=4)      :: data_baud_count  ! data pulse baud count
CHARACTER(64)        :: data_wave_file   ! data baud pattern file name
COMPLEX(KIND=4),DIMENSION(1024) :: data_baud ! data waveform baud pattern
INTEGER(KIND=4)      :: data_pairs       ! number of IQ pairs in waveform memory
REAL(KIND=4)         :: cal_start        ! cal range placement start [us]
REAL(KIND=4)         :: cal_width        ! cal pulse baud width [us]
INTEGER(KIND=4)      :: cal_baud_count   ! cal pulse baud count
CHARACTER(64)        :: cal_wave_file    ! alternative baud pattern file name
COMPLEX(KIND=4),DIMENSION(1024) :: cal_baud ! cal waveform baud pattern
INTEGER(KIND=4)      :: cal_pairs        ! number of IQ pairs in waveform memory
CHARACTER(128)       :: user             ! Spare space for user-defined information
END TYPE TIMINGtype
!
```

SCT Frequency

```
TYPE :: FREQUENCYtype ! Values are in kilohertz unless otherwise indicated
CHARACTER(64)  :: file_id      ! Frequency settings file
REAL(KIND=4)   :: base_start   ! Initial base frequency
REAL(KIND=4)   :: base_end     ! Final base frequency
INTEGER(KIND=4):: base_steps   ! Number of base frequencies
INTEGER(KIND=4):: tune_type    ! Tuning type flag: 1=log, 2=linear, 3=table, 4=ShuffleMode
REAL(KIND=4),DIMENSION(8192) :: base_table ! Nominal or Base frequency table
REAL(KIND=4)   :: linear_step  ! Linear frequency step [kHz]
REAL(KIND=4)   :: log_step     ! Log frequency step, [percent]
CHARACTER(64)  :: freq_table_id ! Manual tuning table filename
INTEGER(KIND=4):: tune_steps   ! all frequencies pre-ramp repeats
INTEGER(KIND=4):: pulse_count  ! pulset frequency vector length
INTEGER(KIND=4),DIMENSION(256) :: pulse_pattern ! pulset frequency vector
REAL(KIND=4)   :: pulse_offset ! pulset offset [kHz]
INTEGER(KIND=4):: ramp_steps   ! pulsets per B-mode ramp (ramp length,base freqs per B-block)
INTEGER(KIND=4):: ramp_repeats ! repeat count of B-mode ramps
REAL(KIND=4),DIMENSION(8192):: drive_table ! base frequencies attenuation/silent table
CHARACTER(128) :: user        ! Spare space for user-defined information
END TYPE FREQUENCYtype
```

SCT Receiver and Exciter

! Receiver Settings

TYPE :: RECEIVERTYPE

```
CHARACTER(64)    :: file_id      ! Frequency settings file
INTEGER(KIND=4)  :: rx_chan      ! Number of receivers
INTEGER(KIND=4), DIMENSION(16) :: rx_map ! receiver-to-antenna mapping
INTEGER(KIND=4)  :: word_format  ! 0=big endian fixed, 1=little endian, 2=floating_point
INTEGER(KIND=4)  :: cic2_dec     ! DDC filter block
INTEGER(KIND=4)  :: cic2_interp  ! DDC filter block
INTEGER(KIND=4)  :: cic2_scale   ! DDC filter block
INTEGER(KIND=4)  :: cic5_dec     ! DDC filter block
INTEGER(KIND=4)  :: cic5_scale   ! DDC filter block
CHARACTER(32)    :: rcf_type     ! text descriptor of FIR filter block
INTEGER(KIND=4)  :: rcf_dec      ! decimation factor for FIR filter block
INTEGER(KIND=4)  :: rcf_taps     ! number of taps in FIR filter block
INTEGER(KIND=4), DIMENSION(160) :: coefficients ! Receiver filter coefficients
REAL(KIND=4)     :: analog_delay ! analog delay of receiver, us
CHARACTER(128)   :: user        ! Spare space for user-defined information
END TYPE RECEIVERTYPE
```

!

! Exciter Settings

TYPE :: EXCITERTYPE

```
CHARACTER(64)    :: file_id      ! Frequency settings file
INTEGER(KIND=4)  :: cic_scale    ! DUC filter block
INTEGER(KIND=4)  :: cic2_dec     ! DUC filter block
INTEGER(KIND=4)  :: cic2_interp  ! DUC filter block
INTEGER(KIND=4)  :: cic5_interp  ! DUC filter block
CHARACTER(32)    :: rcf_type     ! text descriptor of FIR filter block
INTEGER(KIND=4)  :: rcf_taps     ! number of taps in FIR filter block
INTEGER(KIND=4)  :: rcf_taps_phase ! number of taps in FIR filter block
INTEGER(KIND=4), DIMENSION(256) :: coefficients ! Receiver filter coefficients
REAL(KIND=4)     :: analog_delay ! analog delay of exciter/transmitter, us
CHARACTER(128)   :: user        ! Spare space for user-defined information
END TYPE EXCITERTYPE
```

SCT monitor and PCT

! System status and Built-In-Test info

TYPE :: MONITORtype

```
INTEGER(KIND=4),DIMENSION(8) :: balun_currents ! As read prior to ionogram
INTEGER(KIND=4),DIMENSION(8) :: balun_status ! As read prior to ionogram
INTEGER(KIND=4),DIMENSION(8) :: front_end_status! As read prior to ionogram
INTEGER(KIND=4),DIMENSION(8) :: receiver_status ! As read prior to ionogram
INTEGER(KIND=4),DIMENSION(2) :: exciter_status ! As read prior to ionogram
CHARACTER(512) :: user ! Spare space for user-defined information
END TYPE MONITORtype
```

! Pulse Configuration Table Version 1.2

TYPE :: PCTtype

```
INTEGER(KIND=4) :: record_id ! Sequence number of this PCT
REAL(KIND=8) :: pri_ut ! UT of this pulse
REAL(KIND=8) :: pri_time_offset ! Time read from system clock, not precise.
INTEGER(KIND=4) :: base_id ! Base Frequency counter
INTEGER(KIND=4) :: pulse_id ! pulse set element for this PRI
INTEGER(KIND=4) :: ramp_id ! ramp set element for this PRI
INTEGER(KIND=4) :: repeat_id ! ramp repeat element for this PRI
INTEGER(KIND=4) :: loop_id ! Outer loop element for this PRI
REAL(KIND=4) :: frequency ! Frequency of observation (kHz)
INTEGER(KIND=4) :: nco_tune_word ! Tuning word sent to the receiver
REAL(KIND=4) :: drive_attenuation ! Low-level drive attenuation [dB]
INTEGER(KIND=4) :: pa_flags ! Status flags from amplifier
REAL(KIND=4) :: pa_forward_power ! Forward power from amplifier
REAL(KIND=4) :: pa_reflected_power ! Reflected power from amplifier
REAL(KIND=4) :: pa_vswr ! Voltage Standing Wave Ratio from amplifier
REAL(KIND=4) :: pa_temperature ! Amplifier temperature
INTEGER(KIND=4) :: proc_range_count ! Number of range gates kept this PRI
REAL(KIND=4) :: proc_noise_level ! Estimated noise level for this PRI
CHARACTER(64) :: user ! Spare space for user-defined information
END TYPE PCTtype
```