DOE/SC-ARM/TR-113

ARM

CLIMATE RESEARCH FACILITY

Scanning ARM Cloud Radar (X/Ka/W-SACR) HANDBOOK

June 2012

U.S. DEPARTMENT OF ENERGY | Office of Science

Work Supported by the U.S. Department of Energy
Office of Science, Office of Biological and Environmental Research

DOE/SC-ARM/TR-113

Scanning ARM Cloud Radar (X/Ka/W-SACR)

K Widener
N Bharadwaj
K Johnson

June 2012

Work supported by the U.S. Department of Energy,
Office of Science, Office of Biological and Environmental Research

Acronyms and Abbreviations

AGL	above ground level
ARM	Atmospheric Radiation Measurement (Climate Research Facility)
C band	frequencies between 4 GHz and 8 GHz
dB	decibel
dBi	antenna gain referenced to isotropic radiator
dBm	decibel referenced to 1 mW
dBZ	reflectivity
DMF	Data Management Facility
DQO	Data Quality Office (ARM)
EIKA	Extended Interaction Klystron Amplifier
GHz	gigahertz (10^9 Hz)
Hz	hertz
Ka band	frequencies between 26.5 GHz and 40 GHz
KA-SACR	Ka-band scanning ARM cloud radar
KAZR	Ka-band ARM Zenith Radar
K_{DP}	specific differential phase
kW	kilowatt
m	meter
MHz	megahertz (10^6 Hz)
mW	milliwatt
NSA	North Slope of Alaska
PPI	Plan Position Indicator (type of radar scan)
PRF	pulse repetition frequency
RF	radio frequency
RHI	Range Height Indicator (type of radar scan)
SACR	scanning ARM cloud radar
SGP	Southern Great Plains
TWP	Tropical Western Pacific
TWT	Traveling Wave Tube
TWTA	Traveling Wave Tube Amplifier
VAP	value-added product
WACR	W-band ARM cloud radar
W band	frequencies between 75 GHz and 110 GHz
W-SACR	W-band scanning ARM cloud radar
X band	frequencies between 8 GHz and 12 GHz
X-SACR	X-band scanning ARM cloud radar
Z_{DR}	differential reflectivity
ρ_{HV}	correlation coefficient between H and V polarizations
ϕ_{DP}	differential phase

Contents

Figures

Tables

1.0 General Overview

1.1 Introduction

Figure 1. Ka/W-SACR at the Southern Great Plains.

The scanning ARM cloud radar (SACR) is a polarimetric Doppler radar consisting of three different radar designs based on operating frequency. These are designated as follows:

- X-band SACR (X-SACR)
- Ka-band SACR (Ka-SACR)
- W-band SACR (W-SACR).

There are two SACRs on a single pedestal at each site where SACRs are deployed. The selection of the operating frequencies at each deployed site is predominantly determined by atmospheric attenuation at the site. Because RF attenuation increases with atmospheric water vapor content, ARM's Tropical Western Pacific (TWP) sites use the X-/Ka-band frequency pair. The Southern Great Plains (SGP) and North Slope of Alaska (NSA) sites field the Ka-/W-band frequency pair. One ARM Mobile Facility (AMF1) has a Ka/W-SACR and the other (AMF2) has a X/Ka-SACR.

1.2 X-band SACR (X-SACR)

1.2.1 Transmitter

Type:	TWTA
Center frequency:	9.710 GHz
Peak power output:	20.0 kW
Pulse width:	100 ns–40 μs

Polarization:	dual-polarization, simultaneous H and V
Maximum duty cycle:	1%
PRF:	maximum 10 kHz
Manufacturer:	CPI

1.2.2 Receiver

Type:	dual-channel digital
Dynamic range:	> 80 dB
Noise figure:	4.5 dB
Sampling rate:	120 MHz
Decimation factor:	Adjustable
Video bandwidth:	Adjustable
Processing software:	custom development
Manufacturer:	Mercury Computers

1.2.3 Antenna/Pedestal

Antenna Type:	direct feed parabolic reflector
Diameter:	1.82 m
3 dB beam width:	1.40°
Gain:	42.0 dBi
Cross polarization isolation:	-30 dB
2-way radome loss:	<0.2 dB
Pedestal Type:	azimuth over elevation
Azimuth scan rate:	up to 36°/s
Elevation scan rate:	up to 20°/s
Pedestal manufacturer:	Orbit Industries

1.3 Ka-band SACR (Ka-SACR)

1.3.1 Transmitter

Type:	EIKA
Center frequency:	35.3 GHZ
Peak power output:	2.0 kW
Pulse width:	50 ns–13 μs
Polarization:	transmit horizontal linear
Maximum duty cycle:	5%
PRF:	up to 10 kHz
Manufacturer:	CPI Canada with CPI Beverly modulator

1.3.2 Receiver

Type:	dual-channel receiver
Dynamic range:	> 80 dB
Noise figure:	3.5 dB
Sampling rate:	120 MHz
Decimation factor:	adjustable
Video bandwidth:	adjustable
Processing software:	custom
Manufacturer:	Mercury Computer

1.3.3 Antenna/Pedestal

Type:	Cassegrain parabolic reflector
Antenna diameter:	1.82 m
3 dB beam width:	0.33°
Gain:	53.5 dBi
Cross polarization isolation:	-27 dB
2-way radome loss:	1.5 dB
Pedestal type:	elevation over azimuth
Azimuth scan rate:	up to 36°/s
Elevation scan rate:	up to 20°/s

1.4 W-band SACR (W-SACR)

1.4.1 Transmitter

Type:	EIKA
Center frequency:	94.0 GHz
Peak power output:	1.7 kW
Pulse width:	50 ns–2 μs
Polarization:	transmit horizontal linear
Maximum duty cycle:	1%
PRF:	up to 20 kHz
Manufacturer:	CPI Canada with CPI Beverly modulator

1.4.2 Receiver

Type:	dual-channel receiver
Dynamic range:	>80 dB
Noise figure:	6.0 dB
Sampling rate:	120 MHz
Decimation factor:	adjustable
Video bandwidth:	adjustable

1.4.3 Antenna/Pedestal

Type:	Cassegrain parabolic reflector
Antenna diameter:	0.9 m
3 dB beam width:	0.33°
Gain:	53.5 dBi
Cross polarization isolation:	-27 dB
2-way radome loss:	1.5 dB
Pedestal type:	elevation over azimuth
Azimuth scan rate:	up to 36°/s
Elevation scan rate:	up to 20°/s

2.0 Contacts

2.1 Mentor

Kevin Widener
Pacific Northwest National Laboratory
P.O. Box 999
Richland, WA 99352
Phone: (509) 375-2487
Fax: (509) 375-6736
Email: kevin.widener@pnnl.gov

Nitin Bharadwaj
Pacific Northwest National Laboratory
P.O. Box 999
Richland, WA 99352
Phone: (509) 372-4267
Fax: (509) 375-6736
Email: nitin@pnnl.gov

Karen Johnson (for data issues)
Environmental Sciences Department
Brookhaven National Laboratory
Upton, NY 11973
Phone: (631) 344-5952
Fax: (631) 344-2060
Email: kjohnson@bnl.gov

2.2 Vendor / Instrument Developer

ProSensing, Inc.
107 Sunderland Road
Amherst, MA 01002
Website: http://www.prosensing.com

3.0 Deployment Locations and History

Table 1. SACR deployment locations.

Site	Location	Bands	Latitude	Longitude	Altitude	Date Installed
SGP/C1	Billings, OK	Ka/W	36.605300 N	97.486294 W	316	March 2011
NSA/C1	Barrow, AK	Ka/W	71.322953 N	156.615839 W	7	August 2011
TWP/C1	Manus, PNG	X/Ka	2.061062 S	147.425297 E	4	November 2011
TWP/C3	Darwin, AUS	X/Ka	12.424518 S	130.891527 E	29.9	July 2011
AMF1	Cape Cod, MA	Ka/W	42.030503 N	70.049339 W	51	September 2012
AMF2	TBD	X/Ka				

Figure 2. Ka/W-SACR in Barrow, Alaska.

Figure 3. X/Ka-SACR on Gan Island, Maldives.

Figure 4. X/Ka-SACR at Darwin, Australia.

Figure 5. X/Ka-SACR on Manus Island, Papua New Guinea.

4.0 Near-Real-Time Data Plots

See http://plot.dmf.arm.gov/plotbrowser/.

5.0 Data Description and Examples

Since the SACR is a scanning cloud radar, there are many different types of scans that the SACR can perform. Scan strategies are defined for each site that alternate between the following types of scans:

- RHI (Range Height Indicator) scans are scans in which the azimuth axis is held constant while the elevation axis is changed. An RHI scan can go horizon-to-horizon over 180 degrees of elevation or a subset of that. Figure 6. RHI "domer" scan shows one type of RHI scanning scenario that ARM currently uses, also known as a "hemispherical sky RHI." It is a series of horizon-to-horizon RHIs in which the azimuth is incremented. This type of scan provides a good indication of the 3D cloud field. This will be the most common scan for the SACRs.

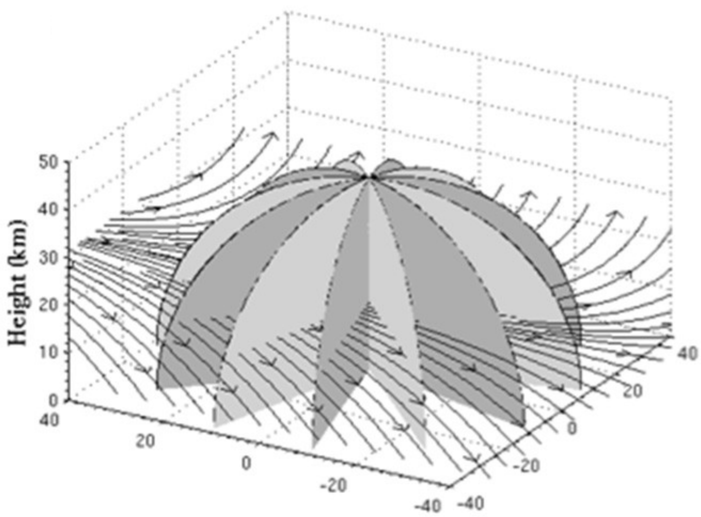

Figure 6. RHI "domer" scan.

- **PPI** (Plan Position Indicator) scans are typically thought of when thinking about weather radars. These are scans in which the elevation is scanned for 360 degrees, and then the elevation is incremented. Figure 7. PPI scan shows a scan with four different elevations.

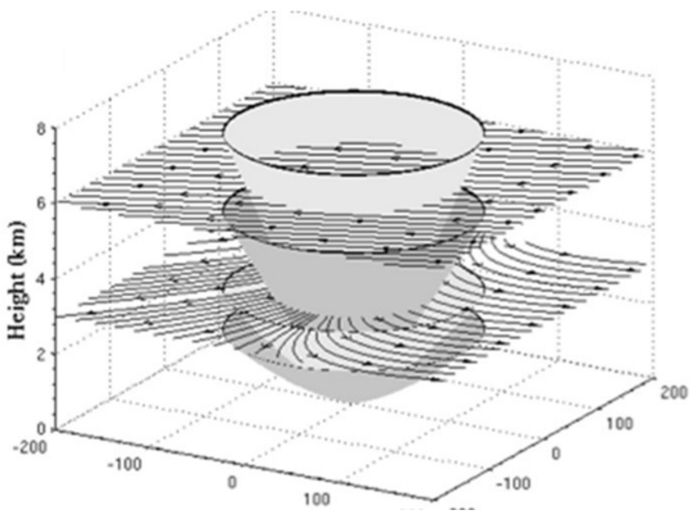

Figure 7. PPI scan.

- **Vertical pointing** – during part of the measurement period, ARM plans on operating the SACR in a vertical-pointing mode to obtain zenith cloud profiles similar to the Ka-band ARM zenith radar (KAZR) and W-band ARM cloud radar (WACR). This mode will be important to provide a calibration reference for the KAZR and WACR. See Figure 8. Vertical-pointing.

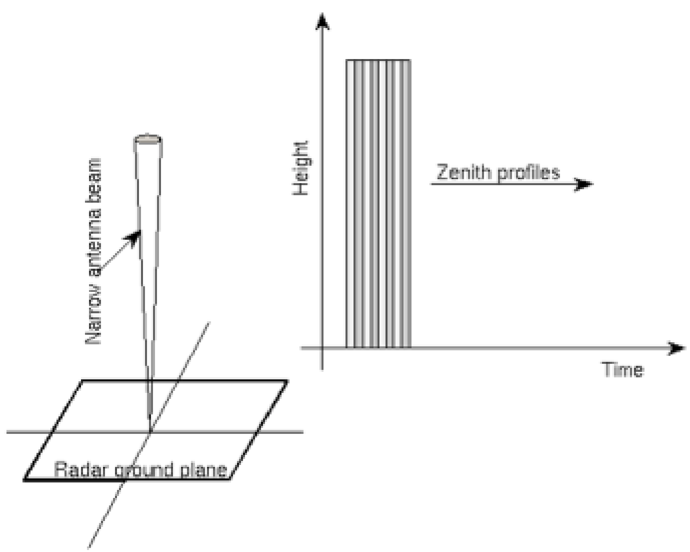

Figure 8. Vertical-pointing mode.

- **Sector –** RHI scans that occur at only one azimuth. In one type of sector scan, the azimuth is selected so that it is perpendicular to the wind direction at a desired altitude, the so-called "cross-wind RHI scan." See Figure 9. Sector scan for an example of this type of scan. A second type of sector scan positions the azimuth to be along the wind direction at a selected altitude. The azimuth directions are currently not determined automatically but must be entered manually.

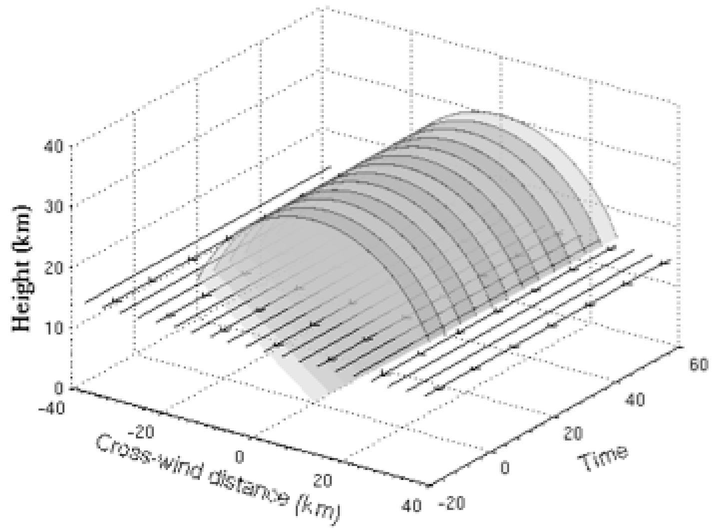

Figure 9. Sector scan.

- **Calibration** – one of the great benefits of having a scanning cloud radar with a relatively narrow beamwidth is the ability to look at a fixed target of known radar cross-section to provide an absolute calibration point for the SACR. In this mode, the SACR points at a predetermined fixed azimuth and elevation for the corner reflector.

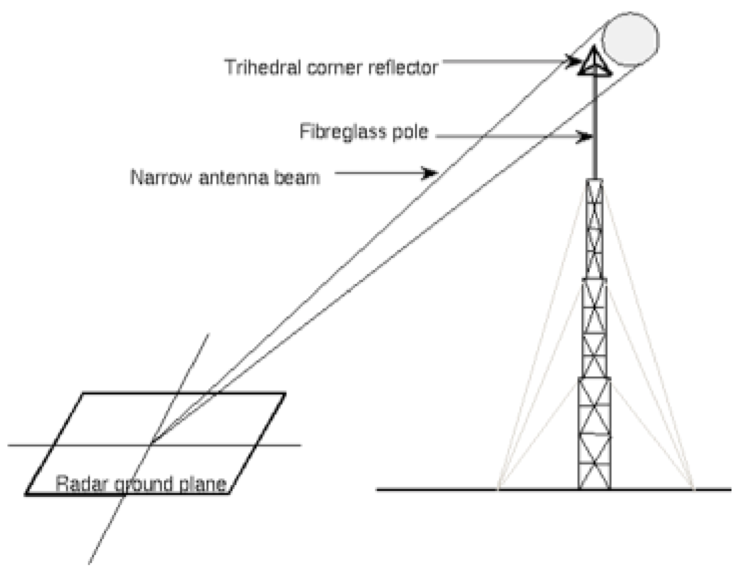

Figure 10. Calibration target.

5.1 Data File Contents

SACR moments data are available from the ARM Data Archive in the following datastreams:

Scan Description	Datastream Name		
	Ka-band	**W-band**	**X-band**
Along-Wind RHI	Kaawrhi	wawrhi	xawrhi
Boundary-Layer RHI	Kablrhi	wblrhi	xblrhi
Cross-Wind RHI	Kacwrhi	wcwrhi	xcwrhi
Hemispherical Sky RHI	Kahsrhi	whsrhi	xhsrhi
PPI	Kappi	wppi	xppi
RHI (general)	Karhi	wrhi	xrhi
VPT	Kavpt	wvpt	xvpt

The SACR datastreams at the archive conform to CfRadial format standards. For more information on this standard, refer to: http://www.ral.ucar.edu/projects/titan/docs/radial_formats/CfRadialDoc.pdf

5.1.1 Ka/W-SACR RHI File Contents

```
int base_time ;
    base_time:string = "2-Feb-2012,16:44:45 GMT" ;
    base_time:long_name = "Base time in Epoch" ;
    base_time:units = "seconds since 1970-1-1 0:00:00 0:00" ;
double time_offset(time) ;
    time_offset:long_name = "Time offset from base_time" ;
    time_offset:units = "seconds since 2012-02-02 16:44:45 0:00" ;
double time(time) ;
    time:long_name = "time in seconds since volume start" ;
    time:units = "seconds since 2012-02-02T16:44:45Z" ;
    time:standard_name = "time" ;
int qc_time(time) ;
    qc_time:long_name = "Quality check results on field: Time in seconds since volume start" ;
    qc_time:units = "unitless" ;
    qc_time:description = "This field contains bit packed values which should be interpreted as listed. No
    bits set (zero) represents good data." ;
    qc_time:bit_1_description = "Delta time between current and previous samples is zero." ;
    qc_time:bit_1_assessment = "Indeterminate" ;
    qc_time:bit_2_description = "Delta time between current and previous samples is less than the
    delta_t_lower_limit field attribute." ;
    qc_time:bit_2_assessment = "Indeterminate" ;
    qc_time:bit_3_description = "Delta time between current and previous samples is greater than the
    delta_t_upper_limit field attribute." ;
    qc_time:bit_3_assessment = "Indeterminate" ;
    qc_time:delta_t_lower_limit = 0.01 ;
```

```
        qc_time:delta_t_upper_limit = 30. ;
        qc_time:prior_sample_flag = 0 ;
        qc_time:comment = "If the \'prior_sample_flag\' is set the first sample time from a new raw file will
        be compared against the time just previous to it in the stored data. If it is not set the qc_time value for
        the first sample will be set to 0." ;
float range(range) ;
        range:long_name = "Range to measurement volume" ;
        range:units = "meters" ;
        range:standard_name = "projection_range_coordinate" ;
        range:spacing_is_constant = "true" ;
        range:meters_to_center_of_first_gate = " 549.620" ;
        range:meters_between_gates = "  24.982" ;
        range:axis = "radial_range_coordinate" ;
float azimuth(time) ;
        azimuth:long_name = "Azimuth angle from true north" ;
        azimuth:units = "degrees" ;
        azimuth:standard_name = "beam_azimuth_angle" ;
        azimuth:axis = "radial_azimuth_coordinate" ;
float elevation(time) ;
        elevation:long_name = "Elevation angle from horizontal" ;
        elevation:units = "degrees" ;
        elevation:standard_name = "beam_elevation_angle" ;
        elevation:axis = "radial_elevation_coordinate" ;
float scan_rate(time) ;
        scan_rate:long_name = "Scan rate, positive for clockwise or increasing elevation" ;
        scan_rate:units = "degrees per second" ;
        scan_rate:meta_group = "instrument_parameters" ;
        scan_rate:standard_name = "antenna_angle_scan_rate" ;
int antenna_transition(time) ;
        antenna_transition:long_name = "Antenna transition indicator, 1 if between sweeps, otherwise 0" ;
        antenna_transition:units = "unitless" ;
        antenna_transition:standard_name = "antenna_is_in_transition_between_sweeps" ;
short reflectivity(time, range) ;
        reflectivity:long_name = "Equivalent reflectivity factor" ;
        reflectivity:units = "dBZ" ;
        reflectivity:standard_name = "equivalent_reflectivity_factor" ;
        reflectivity:_FillValue = -9999s ;
        reflectivity:scale_factor = 0.002766339f ;
        reflectivity:add_offset = -65.01484f ;
        reflectivity:comment = "To unpack field, multiply values by the scale_factor attribute, then add the
        add_offset attribute." ;
        reflectivity:coordinates = "elevation azimuth range" ;
short mean_doppler_velocity(time, range) ;
        mean_doppler_velocity:long_name = "Mean Doppler velocity" ;
        mean_doppler_velocity:units = "meters per second" ;
        mean_doppler_velocity:standard_name = "radial_velocity_of_scatterers_away_from_instrument" ;
```

```
        mean_doppler_velocity:_FillValue = -9999s ;
        mean_doppler_velocity:scale_factor = 0.0006447607f ;
        mean_doppler_velocity:add_offset = -10.20541f ;
        mean_doppler_velocity:comment = "To unpack field, multiply values by the scale_factor attribute,
        then add the add_offset attribute." ;
        mean_doppler_velocity:coordinates = "elevation azimuth range" ;
short spectral_width(time, range) ;
        spectral_width:long_name = "Spectrum width" ;
        spectral_width:units = "meters per second" ;
        spectral_width:standard_name = "spectrum_width" ;
        spectral_width:_FillValue = -9999s ;
        spectral_width:scale_factor = 0.0002320935f ;
        spectral_width:add_offset = 0.f ;
        spectral_width:comment = "To unpack field, multiply values by the scale_factor attribute, then add
        the add_offset attribute." ;
        spectral_width:coordinates = "elevation azimuth range" ;
short snr(time, range) ;
        snr:long_name = "Signal-to-noise-ratio" ;
        snr:units = "dB" ;
        snr:standard_name = "signal_to_noise_ratio" ;
        snr:_FillValue = -9999s ;
        snr:scale_factor = 0.00305341f ;
        snr:add_offset = -25.58069f ;
        snr:comment = "To unpack field, multiply values by the scale_factor attribute, then add the
        add_offset attribute." ;
        snr:coordinates = "elevation azimuth range" ;
short linear_depolarization_ratio(time, range) ;
        linear_depolarization_ratio:long_name = "Linear depolarization ratio H" ;
        linear_depolarization_ratio:units = "dB" ;
        linear_depolarization_ratio:standard_name = "log_linear_depolarization_ratio_h" ;
        linear_depolarization_ratio:_FillValue = -9999s ;
        linear_depolarization_ratio:scale_factor = 0.001795543f ;
        linear_depolarization_ratio:add_offset = -38.76717f ;
        linear_depolarization_ratio:comment = "To unpack field, multiply values by the scale_factor
        attribute, then add the add_offset attribute." ;
        linear_depolarization_ratio:coordinates = "elevation azimuth range" ;
float pulse_width(time) ;
        pulse_width:long_name = "Pulse width" ;
        pulse_width:units = "seconds" ;
        pulse_width:meta_group = "instrument_parameters" ;
        pulse_width:standard_name = "transmitter_pulse_width" ;
float nyquist_velocity(time) ;
        nyquist_velocity:long_name = "Unambiguous Doppler velocity" ;
        nyquist_velocity:units = "meters per second" ;
        nyquist_velocity:meta_group = "instrument_parameters" ;
        nyquist_velocity:standard_name = "unambiguous_doppler_velocity" ;
```

```
float unambiguous_range(time) ;
    unambiguous_range:long_name = "Unambiguous range" ;
    unambiguous_range:units = "meters" ;
    unambiguous_range:meta_group = "instrument_parameters" ;
    unambiguous_range:standard_name = "unambiguous_range" ;
int n_samples(time) ;
    n_samples:long_name = "Number of samples used to compute moments" ;
    n_samples:units = "unitless" ;
    n_samples:meta_group = "instrument_parameters" ;
    n_samples:standard_name = "number_of_samples_used_to_compute_moments" ;
int volume_number ;
    volume_number:long_name = "Volume number" ;
    volume_number:units = "unitless" ;
    volume_number:standard_name = "data_volume_index_number" ;
float frequency(frequency) ;
    frequency:long_name = "Operating frequency" ;
    frequency:units = "Hz" ;
    frequency:meta_group = "instrument_parameters" ;
    frequency:standard_name = "radiation_frequency" ;
int sweep_number(sweep) ;
    sweep_number:long_name = "Sweep number" ;
    sweep_number:units = "count" ;
float prt(sweep) ;
    prt:long_name = "Pulse repetition time" ;
    prt:units = "seconds" ;
    prt:meta_group = "instrument_parameters" ;
    prt:standard_name = "pulse_repetition_time" ;
float fixed_angle(sweep) ;
    fixed_angle:long_name = "Target angle for sweep" ;
    fixed_angle:units = "degrees" ;
    fixed_angle:standard_name = "target_fixed_angle" ;
int sweep_start_ray_index(sweep) ;
    sweep_start_ray_index:long_name = "Index of first ray in sweep, 0-based" ;
    sweep_start_ray_index:units = "count" ;
int sweep_end_ray_index(sweep) ;
    sweep_end_ray_index:long_name = "Index of last ray in sweep, 0-based" ;
    sweep_end_ray_index:units = "count" ;
char follow_mode(sweep, string_length_4) ;
    follow_mode:long_name = "Follow mode" ;
    follow_mode:units = "unitless" ;
    follow_mode:options = "none, sun, vehicle, aircraft, target, manual" ;
    follow_mode:meta_group = "instrument_parameters" ;
    follow_mode:standard_name = "follow_mode_for_scan_strategy" ;
char sweep_mode(sweep, string_length_24) ;
    sweep_mode:long_name = "Sweep mode" ;
    sweep_mode:units = "unitless" ;
```

```
    sweep_mode:comment = "possible values: sector, coplane, rhi, vertical_pointing, idle,
    azimuth_surveillance, elevation_surveillance, sunscan, pointing, manual_ppi, manual_rhi" ;
char prt_mode(sweep, string_length_6) ;
    prt_mode:long_name = "PRT mode" ;
    prt_mode:units = "unitless" ;
    prt_mode:meta_group = "instrument_parameters" ;
    prt_mode:options = "fixed, staggered, dual" ;
    prt_mode:standard_name = "transmit_pulse_mode" ;
char polarization_mode(sweep, string_length_10) ;
    polarization_mode:long_name = "Polarization mode" ;
    polarization_mode:units = "unitless" ;
    polarization_mode:meta_group = "instrument_parameters" ;
    polarization_mode:options = "horizontal, vertical, hv_alt, hv_sim, circular" ;
    polarization_mode:standard_name = "transmit_receive_polarization_mode" ;
float radar_antenna_gain_h ;
    radar_antenna_gain_h:long_name = "Nominal radar antenna gain, h channel" ;
    radar_antenna_gain_h:units = "dB" ;
    radar_antenna_gain_h:meta_group = "radar_parameters" ;
    radar_antenna_gain_h:standard_name = "nominal_radar_antenna_gain_h_channel" ;
float radar_antenna_gain_v ;
    radar_antenna_gain_v:long_name = "Nominal radar antenna gain, v channel" ;
    radar_antenna_gain_v:units = "dB" ;
    radar_antenna_gain_v:meta_group = "radar_parameters" ;
    radar_antenna_gain_v:standard_name = "nominal_radar_antenna_gain_v_channel" ;
float radar_beam_width_h ;
    radar_beam_width_h:long_name = "Radar beam width, horizontal channel" ;
    radar_beam_width_h:units = "degrees" ;
    radar_beam_width_h:meta_group = "radar_parameters" ;
    radar_beam_width_h:standard_name = "half_power_radar_beam_width_h_channel" ;
float radar_beam_width_v ;
    radar_beam_width_v:long_name = "Radar beam width, vertical channel" ;
    radar_beam_width_v:units = "degrees" ;
    radar_beam_width_v:meta_group = "radar_parameters" ;
    radar_beam_width_v:standard_name = "half_power_radar_beam_width_v_channel" ;
float radar_measured_transmit_power_h(time) ;
    radar_measured_transmit_power_h:long_name = "Radar measured transmit power, h channel" ;
    radar_measured_transmit_power_h:units = "dBm" ;
    radar_measured_transmit_power_h:meta_group = "radar_parameters" ;
    radar_measured_transmit_power_h:standard_name = "radar_transmit_power_h_channel"
char platform_type(string_length_6) ;
    platform_type:long_name = "Platform type" ;
    platform_type:units = "unitless" ;
    platform_type:standard_name = "platform_type" ;
char instrument_type(string_length_6) ;
    instrument_type:long_name = "Instrument type" ;
    instrument_type:units = "unitless" ;
```

```
    instrument_type:standard_name = "type_of_instrument" ;
char primary_axis(string_length_6) ;
    primary_axis:long_name = "Primary axis of rotation" ;
    primary_axis:units = "unitless" ;
    primary_axis:standard_name = "primary_axis_of_rotation" ;
char time_coverage_start(string_length_21) ;
    time_coverage_start:long_name = "Time corresponding to first radial in file" ;
    time_coverage_start:units = "unitless" ;
    time_coverage_start:standard_name = "data_volume_start_time_utc" ;
char time_coverage_end(string_length_21) ;
    time_coverage_end:long_name = "Time corresponding to last radial in file" ;
    time_coverage_end:units = "unitless" ;
    time_coverage_end:standard_name = "data_volume_end_time_utc" ;
int r_calib_index(time) ;
    r_calib_index:long_name = "Index for the calibration that applies to each ray" ;
    r_calib_index:units = "unitless" ;
    r_calib_index:standard_name = "calibration_data_array_index_per_ray" ;
char r_calib_time(r_calib, string_length_21) ;
    r_calib_time:long_name = "Time of the calibration that applies to each ray" ;
    r_calib_time:units = "ns" ;
    r_calib_time:standard_name = "radar_calibration_time_utc" ;
float r_calib_pulse_width(r_calib) ;
    r_calib_pulse_width:long_name = "Pulse width for this calibration" ;
    r_calib_pulse_width:units = "unitless" ;
    r_calib_pulse_width:standard_name = "radar_calibration_pulse_width" ;
float r_calib_two_way_radome_loss_h(r_calib) ;
    r_calib_two_way_radome_loss_h:long_name = "Two-way radome loss" ;
    r_calib_two_way_radome_loss_h:units = "dB" ;
    r_calib_two_way_radome_loss_h:meta_group = "radar_calibration" ;
    r_calib_two_way_radome_loss_h:standard_name =
    "radar_calibration_two_way_waveguide_loss_h_channel" ;
float r_calib_xmit_power_h(r_calib) ;
    r_calib_xmit_power_h:long_name = "Transmit power" ;
    r_calib_xmit_power_h:units = "dBm" ;
    r_calib_xmit_power_h:meta_group = "radar_calibration" ;
    r_calib_xmit_power_h:standard_name = "calibrated_radar_xmit_power_h_channel" ;
float r_calib_receiver_gain_hc(r_calib) ;
    r_calib_receiver_gain_hc:long_name = "Receiver gain for horizontal copolar channel 1" ;
    r_calib_receiver_gain_hc:units = "dB" ;
    r_calib_receiver_gain_hc:meta_group = "radar_calibration" ;
    r_calib_receiver_gain_hc:standard_name = "calibrated_radar_receiver_gain_co_polar_channel" ;
float r_calib_receiver_gain_vx(r_calib) ;
    r_calib_receiver_gain_vx:long_name = "Receiver gain for vertical cross-polar channel 2"
    r_calib_receiver_gain_vx:units = "dB" ;
    r_calib_receiver_gain_vx:meta_group = "radar_calibration" ;
```

```
        r_calib_receiver_gain_vx:standard_name = "calibrated_radar_receiver_gain_v_cross_polar_channel"
        ;
float r_calib_noise_hc(r_calib) ;
        r_calib_noise_hc:long_name = "Noise floor power for horizontal copolar channel1" ;
        r_calib_noise_hc:units = "dBm" ;
        r_calib_noise_hc:meta_group = "radar_calibration" ;
        r_calib_noise_hc:standard_name = "calibrated_radar_receiver_noise_h_co_polar_channel" ;
float r_calib_noise_vx(r_calib) ;
        r_calib_noise_vx:long_name = "Noise flor for vertical cross-polar channel 2" ;
        r_calib_noise_vx:units = "dBm" ;
        r_calib_noise_vx:meta_group = "radar_calibration" ;
        r_calib_noise_vx:standard_name = "calibrated_radar_receiver_noise_v_cross_polar_channel" ;
float r_calib_noise_source_power_h(r_calib) ;
        r_calib_noise_source_power_h:long_name = "Noise power in noise source region for horizontal
        copolar channel 1" ;
        r_calib_noise_source_power_h:units = "dBm" ;
        r_calib_noise_source_power_h:meta_group = "radar_calibration" ;
        r_calib_noise_source_power_h:standard_name = "radar_calibration_noise_source_power_h_channel"
        ;
float r_calib_noise_source_power_v(r_calib) ;
        r_calib_noise_source_power_v:long_name = "Noise power in noise source region for vertical cross-
        polar channel" ;
        r_calib_noise_source_power_v:units = "dBm" ;
        r_calib_noise_source_power_v:meta_group = "radar_calibration" ;
        r_calib_noise_source_power_v:standard_name = "radar_calibration_noise_source_power_v_channel"
        ;
float r_calib_radar_constant_h(r_calib) ;
        r_calib_radar_constant_h:long_name = "Radar constant, horizontal copolar channel 1" ;
        r_calib_radar_constant_h:units = "dB" ;
        r_calib_radar_constant_h:meta_group = "radar_calibration" ;
        r_calib_radar_constant_h:standard_name = "calibrated_radar_constant_h_channel" ;
float r_calib_radar_constant_v(r_calib) ;
        r_calib_radar_constant_v:long_name = "Radar constant, vertical cross-polar channel 2" ;
        r_calib_radar_constant_v:units = "dB" ;
        r_calib_radar_constant_v:meta_group = "radar_calibration" ;
        r_calib_radar_constant_v:standard_name = "calibrated_radar_constant_v_channel" ;
double latitude ;
        latitude:long_name = "Latitude" ;
        latitude:units = "degrees_north" ;
double longitude ;
        longitude:long_name = "Longitude" ;
        longitude:units = "degrees_east" ;
double altitude ;
        altitude:long_name = "Altitude" ;
        altitude:units = "meters" ;
        altitude:standard_name = "altitude" ;
```

```
double altitude_agl ;
    altitude_agl:long_name = "Altitude above ground level" ;
    altitude_agl:units = "meters" ;
    altitude_agl:standard_name = "altitude_above_ground_level" ;
float lat ;
    lat:long_name = "North latitude" ;
    lat:units = "degree_N" ;
    lat:valid_min = -90.f ;
    lat:valid_max = 90.f ;
float lon ;
    lon:long_name = "East longitude" ;
    lon:units = "degree_E" ;
    lon:valid_min = -180.f ;
    lon:valid_max = 180.f ;
float alt ;
    alt:long_name = "Altitude above mean sea level" ;
    alt:units = "m" ;
```

5.1.2 Ka/W-SACR PPI File Contents

```
int base_time ;
        base_time:string = "3-Feb-2012,16:55:46 GMT" ;
        base_time:long_name = "Base time in Epoch" ;
        base_time:units = "seconds since 1970-1-1 0:00:00 0:00" ;
double time_offset(time) ;
        time_offset:long_name = "Time offset from base_time" ;
        time_offset:units = "seconds since 2012-02-03 16:55:46 0:00" ;
double time(time) ;
        time:long_name = "time in seconds since volume start" ;
        time:units = "seconds since 2012-02-03T16:55:46Z" ;
        time:standard_name = "time" ;
        int qc_time(time) ;
        qc_time:long_name = "Quality check results on field: Time in seconds since volume start" ;
        qc_time:units = "unitless" ;
        qc_time:description = "This field contains bit packed values which should be interpreted as listed.
        No bits set (zero) represents good data." ;
        qc_time:bit_1_description = "Delta time between current and previous samples is zero." ;
        qc_time:bit_1_assessment = "Indeterminate" ;
        qc_time:bit_2_description = "Delta time between current and previous samples is less than the
        delta_t_lower_limit field attribute." ;
        qc_time:bit_2_assessment = "Indeterminate" ;
        qc_time:bit_3_description = "Delta time between current and previous samples is greater than the
        delta_t_upper_limit field attribute." ;
        qc_time:bit_3_assessment = "Indeterminate" ;
        qc_time:delta_t_lower_limit = 0.01 ;
        qc_time:delta_t_upper_limit = 30. ;
        qc_time:prior_sample_flag = 0 ;
```

```
        qc_time:comment = "If the \'prior_sample_flag\' is set the first sample time from a new raw file
        will be compared against the time just previous to it in the stored data. If it is not set the qc_time
        value for the first sample will be set to 0." ;
float range(range) ;
        range:long_name = "Range to measurement volume" ;
        range:units = "meters" ;
        range:standard_name = "projection_range_coordinate" ;
        range:spacing_is_constant = "true" ;
        range:meters_to_center_of_first_gate = " 549.620" ;
        range:meters_between_gates = "  24.982" ;
        range:axis = "radial_range_coordinate" ;
float azimuth(time) ;
        azimuth:long_name = "Azimuth angle from true north" ;
        azimuth:units = "degrees" ;
        azimuth:standard_name = "beam_azimuth_angle" ;
        azimuth:axis = "radial_azimuth_coordinate" ;
float elevation(time) ;
        elevation:long_name = "Elevation angle from horizontal" ;
        elevation:units = "degrees" ;
        elevation:standard_name = "beam_elevation_angle" ;
        elevation:axis = "radial_elevation_coordinate" ;
float scan_rate(time) ;
        scan_rate:long_name = "Scan rate, positive for clockwise or increasing elevation" ;
        scan_rate:units = "degrees per second" ;
        scan_rate:meta_group = "instrument_parameters" ;
        scan_rate:standard_name = "antenna_angle_scan_rate" ;
int antenna_transition(time) ;
        antenna_transition:long_name = "Antenna transition indicator, 1 if between sweeps, otherwise 0"
        ;
        antenna_transition:units = "unitless" ;
        antenna_transition:standard_name = "antenna_is_in_transition_between_sweeps" ;
short reflectivity(time, range) ;
        reflectivity:long_name = "Equivalent reflectivity factor" ;
        reflectivity:units = "dBZ" ;
        reflectivity:standard_name = "equivalent_reflectivity_factor" ;
        reflectivity:_FillValue = -9999s ;
        reflectivity:scale_factor = 0.002287041f ;
        reflectivity:add_offset = -47.64025f ;
        reflectivity:comment = "To unpack field, multiply values by the scale_factor attribute, then add
        the add_offset attribute." ;
        reflectivity:coordinates = "elevation azimuth range" ;
short mean_doppler_velocity(time, range) ;
        mean_doppler_velocity:long_name = "Mean Doppler velocity" ;
        mean_doppler_velocity:units = "meters per second" ;
        mean_doppler_velocity:standard_name = "radial_velocity_of_scatterers_away_from_instrument"
        ;
        mean_doppler_velocity:_FillValue = -9999s ;
        mean_doppler_velocity:scale_factor = 0.0006447607f ;
        mean_doppler_velocity:add_offset = -10.20541f ;
        mean_doppler_velocity:comment = "To unpack field, multiply values by the scale_factor
        attribute, then add the add_offset attribute." ;
```

```
        mean_doppler_velocity:coordinates = "elevation azimuth range" ;
short spectral_width(time, range) ;
        spectral_width:long_name = "Spectrum width" ;
        spectral_width:units = "meters per second" ;
        spectral_width:standard_name = "spectrum_width" ;
        spectral_width:_FillValue = -9999s ;
        spectral_width:scale_factor = 0.0001108419f ;
        spectral_width:add_offset = 0.f ;
        spectral_width:comment = "To unpack field, multiply values by the scale_factor attribute, then
        add the add_offset attribute." ;
        spectral_width:coordinates = "elevation azimuth range" ;
short snr(time, range) ;
        snr:long_name = "Signal-to-noise-ratio" ;
        snr:units = "dB" ;
        snr:standard_name = "signal_to_noise_ratio" ;
        snr:_FillValue = -9999s ;
        snr:scale_factor = 0.002848917f ;
        snr:add_offset = -17.23047f ;
        snr:comment = "To unpack field, multiply values by the scale_factor attribute, then add the
        add_offset attribute." ;
        snr:coordinates = "elevation azimuth range" ;
short linear_depolarization_ratio(time, range) ;
        linear_depolarization_ratio:long_name = "Linear depolarization ratio H" ;
        linear_depolarization_ratio:units = "dB" ;
        linear_depolarization_ratio:standard_name = "log_linear_depolarization_ratio_h" ;
        linear_depolarization_ratio:_FillValue = -9999s ;
        linear_depolarization_ratio:scale_factor = 0.00143829f ;
        linear_depolarization_ratio:add_offset = -33.57011f ;
        linear_depolarization_ratio:comment = "To unpack field, multiply values by the scale_factor
        attribute, then add the add_offset attribute." ;
        linear_depolarization_ratio:coordinates = "elevation azimuth range" ;
float pulse_width(time) ;
        pulse_width:long_name = "Pulse width" ;
        pulse_width:units = "seconds" ;
        pulse_width:meta_group = "instrument_parameters" ;
        pulse_width:standard_name = "transmitter_pulse_width" ;
float nyquist_velocity(time) ;
        nyquist_velocity:long_name = "Unambiguous Doppler velocity" ;
        nyquist_velocity:units = "meters per second" ;
        nyquist_velocity:meta_group = "instrument_parameters" ;
        nyquist_velocity:standard_name = "unambiguous_doppler_velocity" ;
float unambiguous_range(time) ;
        unambiguous_range:long_name = "Unambiguous range" ;
        unambiguous_range:units = "meters" ;
        unambiguous_range:meta_group = "instrument_parameters" ;
        unambiguous_range:standard_name = "unambiguous_range" ;
int n_samples(time) ;
        n_samples:long_name = "Number of samples used to compute moments" ;
        n_samples:units = "unitless" ;
        n_samples:meta_group = "instrument_parameters" ;
        n_samples:standard_name = "number_of_samples_used_to_compute_moments" ;
```

```
int volume_number ;
        volume_number:long_name = "Volume number" ;
        volume_number:units = "unitless" ;
        volume_number:standard_name = "data_volume_index_number" ;
float frequency(frequency) ;
        frequency:long_name = "Operating frequency" ;
        frequency:units = "Hz" ;
        frequency:meta_group = "instrument_parameters" ;
        frequency:standard_name = "radiation_frequency" ;
int sweep_number(sweep) ;
        sweep_number:long_name = "Sweep number" ;
        sweep_number:units = "count" ;
float prt(sweep) ;
        prt:long_name = "Pulse repetition time" ;
        prt:units = "seconds" ;
        prt:meta_group = "instrument_parameters" ;
        prt:standard_name = "pulse_repetition_time" ;
float fixed_angle(sweep) ;
        fixed_angle:long_name = "Target angle for sweep" ;
        fixed_angle:units = "degrees" ;
        fixed_angle:standard_name = "target_fixed_angle" ;
int sweep_start_ray_index(sweep) ;
        sweep_start_ray_index:long_name = "Index of first ray in sweep, 0-based" ;
        sweep_start_ray_index:units = "count" ;
int sweep_end_ray_index(sweep) ;
        sweep_end_ray_index:long_name = "Index of last ray in sweep, 0-based" ;
        sweep_end_ray_index:units = "count" ;
char follow_mode(sweep, string_length_4) ;
        follow_mode:long_name = "Follow mode" ;
        follow_mode:units = "unitless" ;
        follow_mode:options = "none, sun, vehicle, aircraft, target, manual" ;
        follow_mode:meta_group = "instrument_parameters" ;
        follow_mode:standard_name = "follow_mode_for_scan_strategy" ;
char sweep_mode(sweep, string_length_24) ;
        sweep_mode:long_name = "Sweep mode" ;
        sweep_mode:units = "unitless" ;
        sweep_mode:comment = "possible values: sector, coplane, rhi, vertical_pointing, idle,
        azimuth_surveillance, elevation_surveillance, sunscan, pointing, manual_ppi, manual_rhi" ;
char prt_mode(sweep, string_length_6) ;
        prt_mode:long_name = "PRT mode" ;
        prt_mode:units = "unitless" ;
        prt_mode:meta_group = "instrument_parameters" ;
        prt_mode:options = "fixed, staggered, dual" ;
        prt_mode:standard_name = "transmit_pulse_mode" ;
char polarization_mode(sweep, string_length_10) ;
        polarization_mode:long_name = "Polarization mode" ;
        polarization_mode:units = "unitless" ;
        polarization_mode:meta_group = "instrument_parameters" ;
        polarization_mode:options = "horizontal, vertical, hv_alt, hv_sim, circular" ;
        polarization_mode:standard_name = "transmit_receive_polarization_mode" ;
float radar_antenna_gain_h ;
```

```
		radar_antenna_gain_h:long_name = "Nominal radar antenna gain, h channel" ;
		radar_antenna_gain_h:units = "dB" ;
		radar_antenna_gain_h:meta_group = "radar_parameters" ;
		radar_antenna_gain_h:standard_name = "nominal_radar_antenna_gain_h_channel" ;
	float radar_antenna_gain_v ;
		radar_antenna_gain_v:long_name = "Nominal radar antenna gain, v channel" ;
		radar_antenna_gain_v:units = "dB" ;
		radar_antenna_gain_v:meta_group = "radar_parameters" ;
		radar_antenna_gain_v:standard_name = "nominal_radar_antenna_gain_v_channel" ;
	float radar_beam_width_h ;
		radar_beam_width_h:long_name = "Radar beam width, horizontal channel" ;
		radar_beam_width_h:units = "degrees" ;
		radar_beam_width_h:meta_group = "radar_parameters" ;
		radar_beam_width_h:standard_name = "half_power_radar_beam_width_h_channel" ;
	float radar_beam_width_v ;
		radar_beam_width_v:long_name = "Radar beam width, vertical channel" ;
		radar_beam_width_v:units = "degrees" ;
		radar_beam_width_v:meta_group = "radar_parameters" ;
		radar_beam_width_v:standard_name = "half_power_radar_beam_width_v_channel" ;
	float radar_measured_transmit_power_h(time) ;
		radar_measured_transmit_power_h:long_name = "Radar measured transmit power, h channel" ;
		radar_measured_transmit_power_h:units = "dBm" ;
		radar_measured_transmit_power_h:meta_group = "radar_parameters" ;
		radar_measured_transmit_power_h:standard_name = "radar_transmit_power_h_channel" ;
	char platform_type(string_length_6) ;
		platform_type:long_name = "Platform type" ;
		platform_type:units = "unitless" ;
		platform_type:standard_name = "platform_type" ;
	char instrument_type(string_length_6) ;
		instrument_type:long_name = "Instrument type" ;
		instrument_type:units = "unitless" ;
		instrument_type:standard_name = "type_of_instrument" ;
	char primary_axis(string_length_6) ;
		primary_axis:long_name = "Primary axis of rotation" ;
		primary_axis:units = "unitless" ;
		primary_axis:standard_name = "primary_axis_of_rotation" ;
	char time_coverage_start(string_length_21) ;
		time_coverage_start:long_name = "Time corresponding to first radial in file" ;
		time_coverage_start:units = "unitless" ;
		time_coverage_start:standard_name = "data_volume_start_time_utc" ;
	char time_coverage_end(string_length_21) ;
		time_coverage_end:long_name = "Time corresponding to last radial in file" ;
		time_coverage_end:units = "unitless" ;
		time_coverage_end:standard_name = "data_volume_end_time_utc" ;
	int r_calib_index(time) ;
		r_calib_index:long_name = "Index for the calibration that applies to each ray" ;
		r_calib_index:units = "unitless" ;
		r_calib_index:standard_name = "calibration_data_array_index_per_ray" ;
	char r_calib_time(r_calib, string_length_21) ;
		r_calib_time:long_name = "Time of the calibration that applies to each ray" ;
		r_calib_time:units = "ns" ;
```

```
        r_calib_time:standard_name = "radar_calibration_time_utc" ;
float r_calib_pulse_width(r_calib) ;
        r_calib_pulse_width:long_name = "Pulse width for this calibration" ;
        r_calib_pulse_width:units = "unitless" ;
        r_calib_pulse_width:standard_name = "radar_calibration_pulse_width" ;
float r_calib_two_way_radome_loss_h(r_calib) ;
        r_calib_two_way_radome_loss_h:long_name = "Two-way radome loss" ;
        r_calib_two_way_radome_loss_h:units = "dB" ;
        r_calib_two_way_radome_loss_h:meta_group = "radar_calibration" ;
        r_calib_two_way_radome_loss_h:standard_name =
        "radar_calibration_two_way_waveguide_loss_h_channel" ;
float r_calib_xmit_power_h(r_calib) ;
        r_calib_xmit_power_h:long_name = "Transmit power" ;
        r_calib_xmit_power_h:units = "dBm" ;
        r_calib_xmit_power_h:meta_group = "radar_calibration" ;
        r_calib_xmit_power_h:standard_name = "calibrated_radar_xmit_power_h_channel" ;
float r_calib_receiver_gain_hc(r_calib) ;
        r_calib_receiver_gain_hc:long_name = "Receiver gain for horizontal copolar channel 1" ;
        r_calib_receiver_gain_hc:units = "dB" ;
        r_calib_receiver_gain_hc:meta_group = "radar_calibration" ;
        r_calib_receiver_gain_hc:standard_name = "calibrated_radar_receiver_gain_co_polar_channel" ;
float r_calib_receiver_gain_vx(r_calib) ;
        r_calib_receiver_gain_vx:long_name = "Receiver gain for vertical cross-polar channel 2" ;
        r_calib_receiver_gain_vx:units = "dB" ;
        r_calib_receiver_gain_vx:meta_group = "radar_calibration" ;
        r_calib_receiver_gain_vx:standard_name =
        "calibrated_radar_receiver_gain_v_cross_polar_channel" ;
float r_calib_noise_hc(r_calib) ;
        r_calib_noise_hc:long_name = "Noise floor power for horizontal copolar channel1" ;
        r_calib_noise_hc:units = "dBm" ;
        r_calib_noise_hc:meta_group = "radar_calibration" ;
        r_calib_noise_hc:standard_name = "calibrated_radar_receiver_noise_h_co_polar_channel" ;
float r_calib_noise_vx(r_calib) ;
        r_calib_noise_vx:long_name = "Noise flor for vertical cross-polar channel 2" ;
        r_calib_noise_vx:units = "dBm" ;
        r_calib_noise_vx:meta_group = "radar_calibration" ;
        r_calib_noise_vx:standard_name = "calibrated_radar_receiver_noise_v_cross_polar_channel" ;
float r_calib_noise_source_power_h(r_calib) ;
        r_calib_noise_source_power_h:long_name = "Noise power in noise source region for horizontal
        copolar channel 1" ;
        r_calib_noise_source_power_h:units = "dBm" ;
        r_calib_noise_source_power_h:meta_group = "radar_calibration" ;
        r_calib_noise_source_power_h:standard_name =
        "radar_calibration_noise_source_power_h_channel" ;
float r_calib_noise_source_power_v(r_calib) ;
        r_calib_noise_source_power_v:long_name = "Noise power in noise source region for vertical
        cross-polar channel" ;
        r_calib_noise_source_power_v:units = "dBm" ;
        r_calib_noise_source_power_v:meta_group = "radar_calibration" ;
        r_calib_noise_source_power_v:standard_name =
        "radar_calibration_noise_source_power_v_channel" ;
```

```
float r_calib_radar_constant_h(r_calib) ;
        r_calib_radar_constant_h:long_name = "Radar constant, horizontal copolar channel 1" ;
        r_calib_radar_constant_h:units = "dB" ;
        r_calib_radar_constant_h:meta_group = "radar_calibration" ;
        r_calib_radar_constant_h:standard_name = "calibrated_radar_constant_h_channel" ;
        float r_calib_radar_constant_v(r_calib) ;
        r_calib_radar_constant_v:long_name = "Radar constant, vertical cross-polar channel 2" ;
        r_calib_radar_constant_v:units = "dB" ;
        r_calib_radar_constant_v:meta_group = "radar_calibration" ;
        r_calib_radar_constant_v:standard_name = "calibrated_radar_constant_v_channel" ;
double latitude ;
        latitude:long_name = "Latitude" ;
        latitude:units = "degrees_north" ;
double longitude ;
        longitude:long_name = "Longitude" ;
        longitude:units = "degrees_east" ;
double altitude ;
        altitude:long_name = "Altitude" ;
        altitude:units = "meters" ;
        altitude:standard_name = "altitude" ;
double altitude_agl ;
        altitude_agl:long_name = "Altitude above ground level" ;
        altitude_agl:units = "meters" ;
        altitude_agl:standard_name = "altitude_above_ground_level" ;
float lat ;
        lat:long_name = "North latitude" ;
        lat:units = "degree_N" ;
        lat:valid_min = -90.f ;
        lat:valid_max = 90.f ;
float lon ;
        lon:long_name = "East longitude" ;
        lon:units = "degree_E" ;
        lon:valid_min = -180.f ;
        lon:valid_max = 180.f ;
float alt ;
        alt:long_name = "Altitude above mean sea level" ;
        alt:units = "m" ;
```

5.1.3 X-SACR Data File Contents

The X-SACR, unlike the Ka/W-SACR, transmits both polarizations simultaneously. This allows the determination of the following dual-polarization parameters:

- Differential Reflectivity (Z_{DR}) – the ratio of horizontal and vertical radar returns. It gives a good estimation of the shape of the precipitation particles, which leads to an indication of the size of the precipitation particles.
- Correlation Coefficient (ρ_{HV} or RHOHV) – the correlation between the horizontal and vertical radar returns. This can provide a good indication on where there is mixed-phase precipitation, i.e., both ice and liquid water. Values near 1 indicate homogeneous precipitation.

- Differential Phase (ϕ_{DP} or PHIDP) – the total differential phase between the horizontal and vertical radar returns.
- Specific Differential Phase (K_{DP}) – the range derivative of the differential phase.

These parameters will be included in a future release of the ingested X-SACR data file. In the initial release, the data file fields are essentially the same as for the Ka and W-SACR files.

5.1.4 Primary Variables and Expected Uncertainty

Table 2. Primary variables.

Variable	Description	Uncertainty
reflectivity_copol	Co-polarization reflectivity (time, height) in dBZ	3 dBZ
reflectivity_xpol	Cross-polarization reflectivity (time, height) in dBZ	3 dBZ
mean_doppler_velocity_copol	Co-polarization radial Doppler velocity (time,height) in m/s	0.1 m/s
mean_doppler_velocity_xpol	Cross-polarization radial Doppler velocity (time, height) in m/s	0.1 m/s
spectral_width_copol	Co-polarization spectral width (time, height) in m/s	0.1 m/s
spectral_width_xpol	Cross-polarization spectral width (time, height) in m/s	0.1 m/s

The uncertainty related to Doppler variables listed in Table 2 is for zenith-pointing mode. The uncertainty for scanning mode is to be determined at each frequency band.

5.2 Annotated Examples

To be determined.

5.3 User Notes and Known Problems

To be determined.

5.4 Frequently Asked Questions

What is the meteorological radar range equation?

This is the equation to determine the reflectivity and is usually given in decibels of Z or dBZ.

$$Z = 10\log\left(\frac{1024\ln(2)\lambda^2 R^2 P_r L_a L_{sys}}{10^{-18} c\tau\pi^3 G_0^2 |K_w|^2 \theta_{3dB}^2 P_t}\right) \qquad dBZ$$

where:

Z = reflectivity (dBZ)
λ = wavelength (m)
R = range (m)
P_r = received power (watts)
L_a = two-way atmospheric loss
L_{sys} = radar system losses
c = speed of light (m/s)
τ = pulse width (s)
G_o = antenna gain
$/K_w/^2$ = index of refraction factor for liquid water at 0 C
θ_{3dB} = antenna beamwidth
P_t = transmit power (watts)

What dielectric factor for water is used to compute reflectivity in the radar range equation $/K_w/^2$?

The dielectric factor of water is a function of frequency and temperature of water drops as shown in Figure 11.

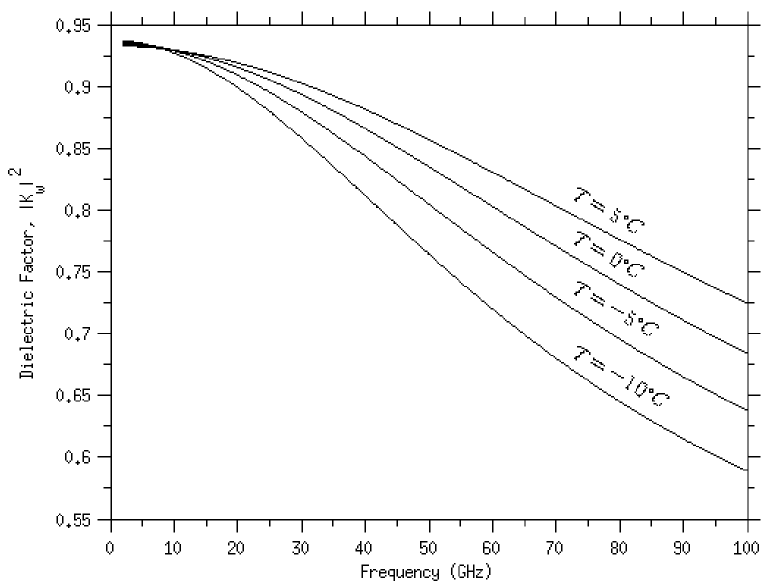

Figure 11. Dielectric function of water as a function of frequency and temperature.

The dielectric factor of water at 0° C is used for the computation of equivalent reflectivity factor. The values of dielectric factor used are 0.93 for X-SACR, 0.88 for Ka-SACR, and 0.70 for W-SACR.

6.0 Data Quality

To be determined.

6.1 Data Quality Health and Status

The Data Quality Office (DQO) website has links to several tools for inspecting and assessing SACR data quality:

- DQ Explorer
- DQ Plot Browser
- NCVweb: Interactive web-based tool for viewing ARM data.

Plots of reflectivity, Doppler radial velocity, and Doppler spectral width provide a good indicator of whether the system is operational or not.

6.2 Data Reviews by Instrument Mentor

Instrument mentors review SACR data in the following ways:

- Routine review for nominal operation, usually daily Monday–Friday
- When requested by site operations
- When requested by the site scientist team
- When requested by an ARM data translator
- When requested by a data user
- When notified automatically by the SACR's built-in test (BIT) email messages.

6.3 Data Assessments by Site Scientist / Data Quality Office

To be determined.

6.4 Value-Added Procedures and Quality Measurement Experiments

There are no value-added products (VAPs) for SACR at this time. There are plans to produce a "corrected moments" product in the coming year, which will include velocity and range dealiasing and water vapor attenuation correction. After that, a gridded moments product with cloud boundaries will be produced.

7.0 Instrument Details

Instrument details will be listed in the "SACR Operations Manual," currently in progress.

7.1 Detailed Description

The following three figures show block diagrams for the three radar bands being used in SACR.

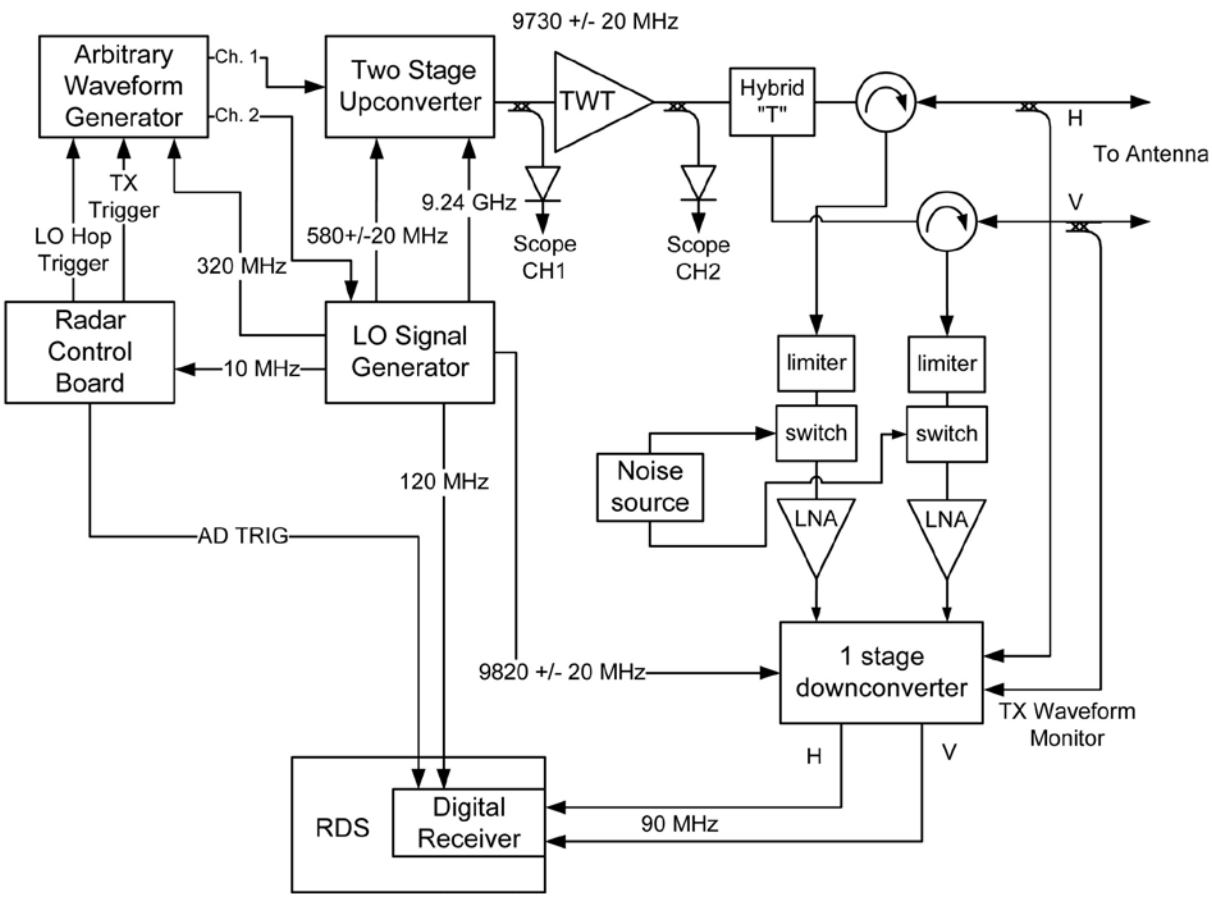

Figure 12. X-SACR block diagram.

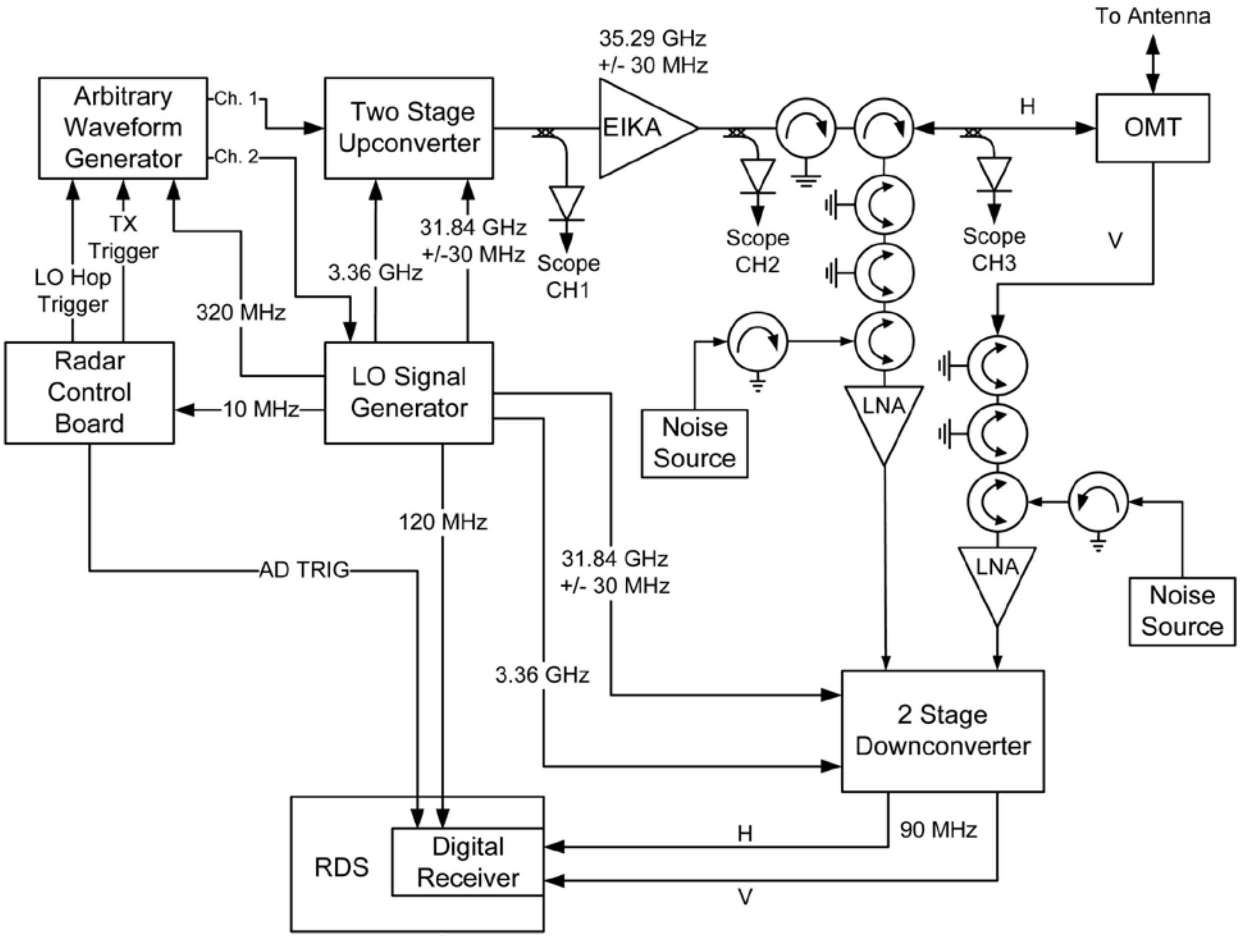

Figure 13. Ka-SACR block diagram.

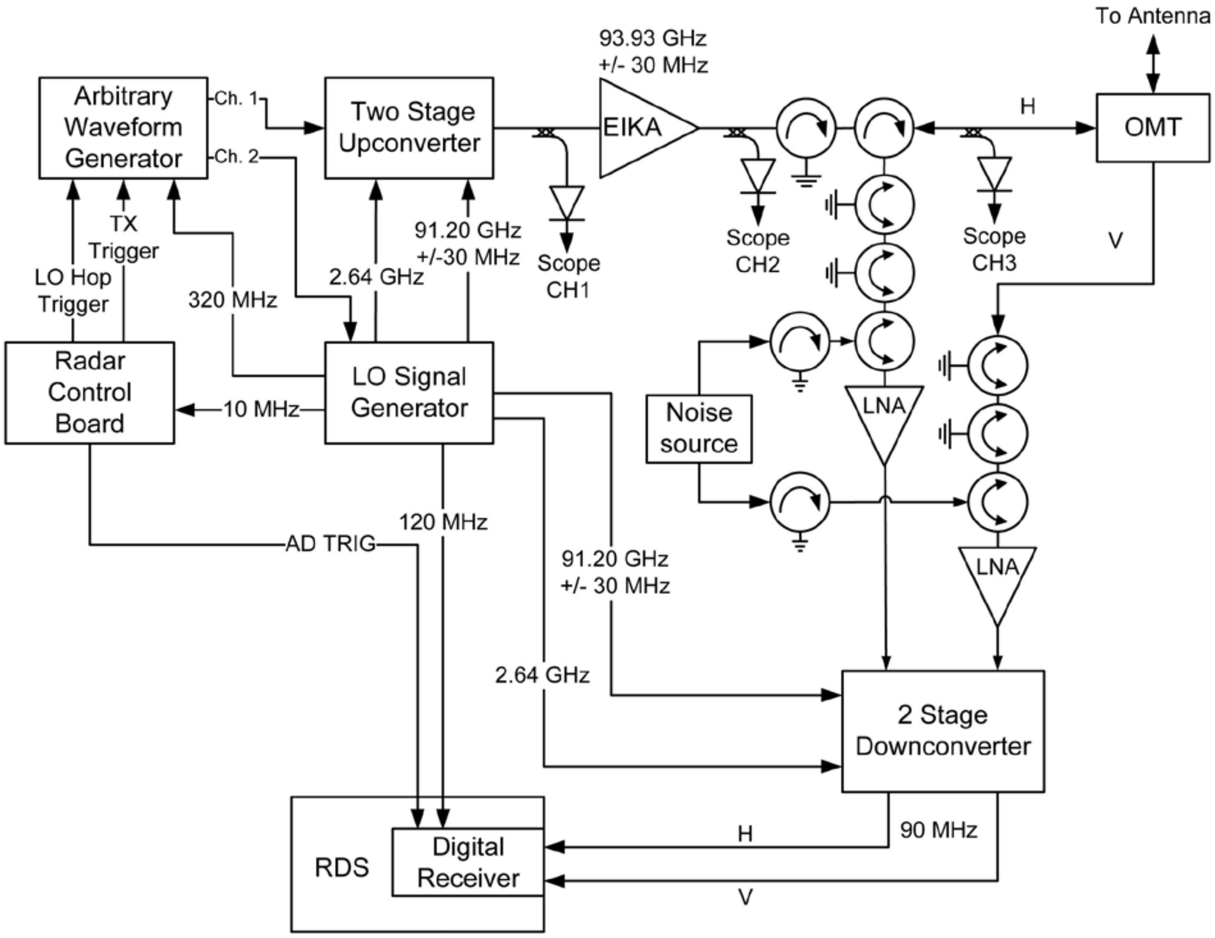

Figure 14. W-SACR block diagram.

7.2 Theory of Operation

The theory of operation will be listed in the "SACR Operations Manual," currently in progress.

7.3 Calibration

The SACRs have the advantage of having narrow beamwidths antennas on a pedestal, making the use of a trihedral corner reflector calibration target possible. For Ka/W-SACRs, a single corner reflector is used. X/Ka-SACR installations require a separate corner reflector for each radar band.

Calibration information will be listed in the "SACR Operations Manual," currently in progress. Additional information is available in the ARM Common Calibration Database (CCDB); access is limited to ARM Facility personnel.

7.4 Operation and Maintenance

Operation and maintenance information will be listed in the "SACR Operations Manual," currently in progress.

7.5 Citable References

Bharadwaj, N, K Widener, A Koontz, and K Johnson. “Data Specification for ARM Scanning Radars”, Draft ARM Technical Report May 2010

Bringi, VN and V Chandrasekar. 2001. “Polarimetric Doppler Weather Radar.” Cambridge University Press.

Doviak, RJ and DS Zrnic. 1993. “Doppler Radar and Weather Observations.” 2nd Edition, Academic Press.

Mead, J. “Scanning ARM Cloud Radar (SACR) System Description and Operations Manual.”. In process.