Radiometer Physics
A Rohde & Schwarz Company

RPG-FMCW-94 Cloud Radar
(Operation and Software Manual)

RPG-FMCW-94(35)-SP/DP
94 (35) GHz W(K)-band Doppler Cloud Radar
Operation and Software Guide (Version 5.56)

Radiometer Physics
A Rohde & Schwarz Company

Edited by©: Th. Rose, Radiometer Physics GmbH, Germany

Document Change Log

Date	Issue/Rev	Change
25.10.2015	00/01	Work
30.10.2015	01/00	Release
03.08.2016	01/01	Adding chapter 2.11 as a description of polarimetric observation data
16.11.2016	01/02	Update for software version 2.0 and adding installation chapter
10.01.2016	01/02	Update for software version 2.1, adding quicklooks
19.06.2017	01/03	Introduction of LV0 / LV1 file version 3.5
06.07.2017	01/04	Retrieval post processor implemented
13.09.2017	01/05	File version 3.5 extension (sample file pointers)
17.10.2017	01/06	Radiometer interfacing (thermodynamic profiles)
11.11.2017	01/07	Splitting installation manual from instrument manual
09.05.2020	01/08	Update of LV0/LV1 file formats
21.07.2020	01/09	Adding section about data server
06.10.2020	01/10	File formats for ABSCAL.CLB and MDFs added to appendix A
15.10.2020	01/11	Chapter for SMTP server (E-Mail alerts) added
09.02.2021	01/12	Hardware specification removed
17.02.2021	01/13	Description of MBF timing display added
07.04.2021	01/14	UPS control chapter added
11.04.2021	01/15	New Data Server command added (DS_GET_RAD_SAMPLE)
14.04.2021	01/16	File format for I/Q data added
02.05.2021	01/17	File format for MBF added
02.05.2021	01/18	New Data Server command added (DS_GET_HKD_SAMPLE)
20.05.2022	01/19	Updated logo on the title page

Table of Contents

1 Theory of Operation

1.1 The Radar Equation

The radar range equation represents the physical dependences of the transmit power and the wave propagation up to the receiving of the echo-signals. The power P_R returning to the receiving antenna is given by the radar equation, depending on the transmitted power P_T, slant range R and the reflecting characteristics of the target (described as the radar cross-section σ). At known sensitivity of the radar receiver the radar equation determines the maximum range theoretically achieved by a given radar setup. Furthermore, one can assess the performance of the radar setup with the radar equation.

In this context we assume that electromagnetic waves propagate through the atmosphere under ideal conditions, i.e. without dispersion.

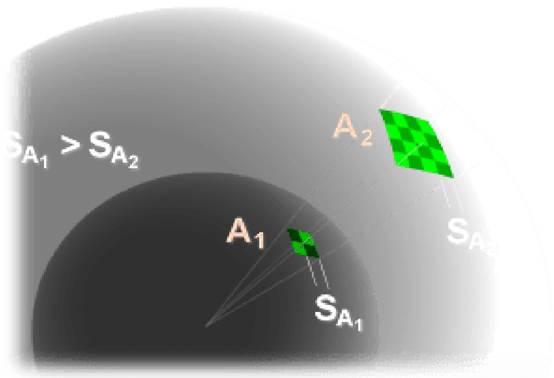


Figure 1: Non-directional power density diminishes as geometric spreading of the beam.

When microwave energy is emitted by an ideal isotropic radiator, it propagates uniformly in all directions. Therefore, areas with the same power density are forming spheres ($A= 4 \pi R^2$) around the radiator. The power density on the surface of a sphere is inversely proportional to the square of the sphere's radius.

The non-directional power density is given by:

$$P_d = \frac{P_T}{4\pi R^2} \quad , P_T = transmitter\ power$$

The transmitter's antenna gain G_T leads to an amplification of the power density along the beam axis

$$\tilde{P}_d = P_d G_T$$

Typical antenna gains of high frequency radars are in the order of 10^5 or higher.
At the target location, the amount of reflected power is determined by the target's scattering cross section σ:

$$P_r = \frac{P_T G_T}{4\pi R^2} \sigma$$

On its way back to the receiver, the reflected power is diluted in the same way as the power travelling from the transmitter to the target with the effective antenna aperture A replacing σ in the formula above and assuming that the scattering target is acting as an isotropic radiator. The power P_R received by the radar is then given as:

$$P_R = \frac{P_T G_T}{(4\pi)^2 R^4} \sigma A$$

Using the well-known relationship between antenna gain and effective aperture

$$A = \frac{G_R \lambda^2}{4\pi}$$

we end up with the general form of the radar equation (assuming $G_T = G_R$):

$$\frac{P_R}{P_T} = \frac{{G_T}^2 \lambda^2}{(4\pi)^3 R^4} \sigma \qquad (1.1.1)$$

The details of the scattering process are completely contained in the cross section σ. In the case of a point target and assuming the target size to be large compared to the radar's wavelength, σ is proportional to the geometrical cross section. For instance, a metal sphere of radius r has a cross section $\sigma = \pi r^2$.

If the target is a volumetric scattering medium (like a cloud) with the transmitter power density distribution close to a Gaussian distribution (the overlap of the RPG-FMCW radar's antenna beam pattern with a Gaussian pattern is 95%), the scattering volume is given by

$$V = \pi \left(\frac{\theta R}{2}\right)^2 \frac{\delta R}{2\ln(2)} \qquad (1.1.2)$$

δR is the range resolution and Θ is the antenna's half power beam width in radians. The reflectivity in cloud radar applications is often expressed in terms of an equivalent Rayleigh back scattering cross section:

$$\sigma = \frac{\pi^5}{\lambda^4} K^2 \sum_i^V {D_i}^6 \qquad (1.1.3)$$

The D_i are the various particle diameters in the scattering volume V and the sum is taken over this volume. K is given by the complex dielectric constant ε of the medium as:

$$K^2 = \left(\frac{\varepsilon - 1}{\varepsilon + 2}\right)^2$$

For pure water, |K| is close to 0.86 at 90 GHz.
It is common practice in cloud radar applications to define the equivalent reflectivity Z_e, which is wavelength independent and can be used to compare reflectivity data from radars operating at different frequencies:

$$Z_e = \frac{10^{18} \cdot \sum_i^V {D_i}^6}{V} \;, \qquad [Z_e] = \frac{mm^6}{m^3} \qquad (1.1.4)$$

Combining equations (2.1.1), (2.1.2), (2.1.3) and (2.1.4) we get:

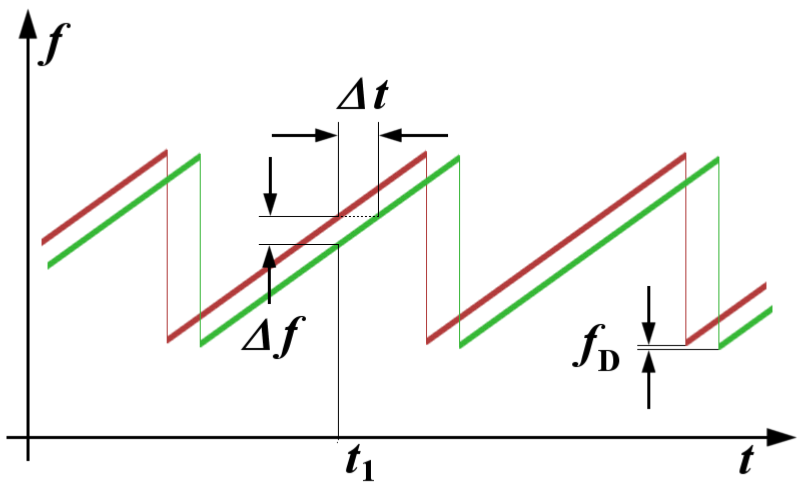
$$\frac{P_R}{P_T} = \frac{{G_T}^2\lambda^2}{(4\pi)^3R^4}\frac{V\pi^5K^2}{10^{18}\lambda^4}Z_e = \frac{{G_T}^2\theta^2\pi^3K^2}{5.12\cdot 10^{20}\ln(2)\,\lambda^2}\frac{\delta R}{R^2}Z_e = C_R\frac{\delta R}{R^2}Z_e \qquad (1.1.5)$$

C_R is the radar constant summarizing the optical radar parameters (G_T, Θ, λ) and dielectric properties of the medium (K^2, assumed to be constant).

1.2 FMCW Radar Operation

In contradiction to traditional pulsed radars, FMCW radars (Frequency Modulated Continuous Wave) do not require a concentrated high-power transmission of several kW peak power. Instead, the transmitter power is emitted continuously, but with linearly varying frequency. The advantages are:

- Low transmitter power in the order of a few Watts. Radar networks in urban areas are possible due to ease of deployment approval for low power transmitters.
- 100% integration duty cycle
- Transmitter can be realized in solid state technology, leading to long lifetime and low cost
- Radar calibration is more accurate compared to pulsed radars, because the transmitter power can be determined more accurately.
- No high-power magnetron tubes and high voltage power supplies are required, reducing the overall radar power consumption.
- High flexibility of range/Doppler resolution variation along the ranging path
- Lower costs for the overall radar



The FMCW radar is continuously emitting saw tooth frequency chirps as indicated by the red curve in the diagram above:

Here the green curve represents the reflected echo of a point source as received by the radar. The time delay between the two curves is proportional to the target distance R:

$$\Delta t = \frac{2R}{c} \;, \qquad c = speed\ of\ light$$

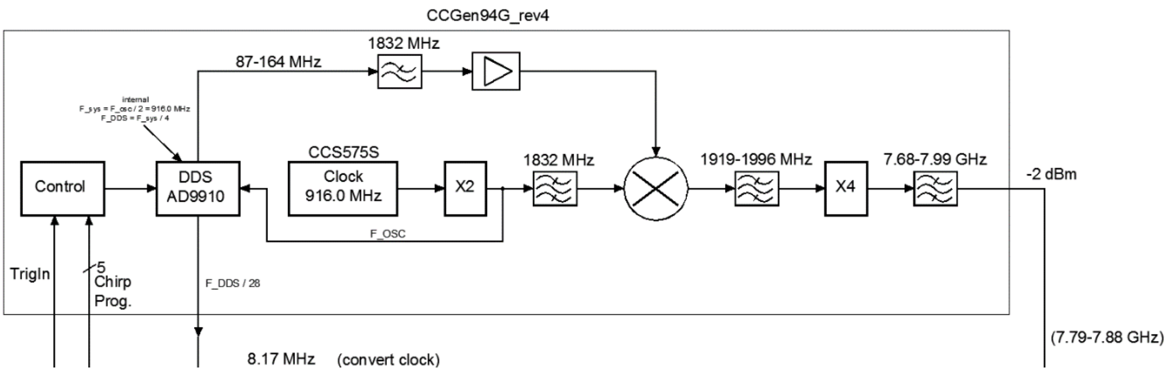
The frequency mixer inside the radar measures the frequency difference Δf (IF spectrum) which is given by the chirp's ramp slope S = Δf / Δt:

$$\Delta f = f_{IF} = \frac{2S}{c}\ R\,, \qquad S = \frac{B}{T_c} \qquad (1.2.1)$$

B = chirp bandwidth, T_c = chirp duration. The frequency resolution δf for measuring f_{IF} is 1 / T_c. Therefore

$$\frac{f_{IF}}{\delta f} = \frac{2B}{c}\ R = \frac{R}{\delta R}\,, \qquad \delta R = \frac{c}{2B} \quad (range\ resolution) \qquad (1.2.2)$$

In RPG's FMCW radars the chirp sequences are generated by a chirp generator module based on a DDS (Direct Digital Synthesizer):



The DDS is clocked by a fundamental oscillator of frequency f_0 = 916 MHz. The chirp generator generates an output signal f_{cg} in the range 7.79 GHz to 7.88 GHz which is then multiplied by 12 to become the transmitter signal at W-band (94 GHz). f_{cg} is derived from the DDS output frequency f_{DDS} by the following relation:

$$f_{cg} = 4 \cdot (2f_0 + f_{DDS})$$

as can be seen from the chirp generator layout above. The DDS's ramp generator uses an internal time base

$$\Delta T_{RG} = \frac{4}{f_0} \qquad (1.2.3)$$

and a frequency resolution (referenced to the radar's transmitter output at W-band) of

$$\Delta f_{RG} = \frac{48 \cdot f_0}{2^{32}}$$

In order to generate frequency ramps of slope S as in equation (1.2.1), the generator can be programmed with an integer step size number S_z:

$$S = S_z \frac{\Delta f_{RG}}{\Delta T_{RG}} = S_z \frac{3{f_0}^2}{2^{30}}$$

S_z can be computed by using equation (1.2.1):

$$S_z = \frac{2^{29} c}{3{f_0}^2} \frac{f_{IF}}{R}$$

The factor f_{IF} / R is mapping the IF frequency range to the distance range. When a range resolution δR shall be achieved, the number of ramp steps N can be derived from (1.2.2):

$$\delta R = \frac{c}{2 S_z \Delta f_{RG} N}$$

Because of (1.2.3) the chirp duration is then fixed to be

$$T_c = \frac{4N}{f_0} \qquad (1.2.4)$$

The mixer's IF voltage signal is sampled by a fast ADC board and directly converted to binary code. The sampling clock for the ADC board is generated by the chirp generator as well to ensure a perfect synchronization of ramp generation and ADC sampling. The generator's sampling frequency is given by

$$f_{SR} = \frac{f_0}{4M} \quad , \qquad M = 4 * (n+1) \ \ , \ \ n = integer\ number \geq 1$$

This limits the sampling rate to a maximum of f_0 / 32 = 28.625 MHz.
The number of samples in a single chirp N_{FFT} is converted to a digital frequency spectrum by an FFT (Fast Fourier Transform):

$$T_c = \frac{N_{FFT}}{f_{SR}} \quad , \qquad N_{FFT} = \frac{N}{M} \qquad (using\ (1.2.4))$$

1.3 Power Spectrum

The mixer output voltage signal is digitized with a sampling rate f_{SR} and the output spectrum is determined by transforming the voltage time series via a Fourier transform for a limited period of N_{FFT} samples:

$$V(t) \quad ==> \quad S(f_{IF}) \qquad (1.3.1)$$

Due to the discrete form of Parceval's theorem the mean power of the time series and spectrum over the sampling interval is identical:

$$P_m = \frac{1}{N_{FFT}} \sum_{k=1}^{N_{FFT}} |V_k|^2 = \frac{1}{{N_{FFT}}^2} \sum_{n=1}^{N_{FFT}} |S_n|^2 \qquad (1.3.2)$$

Because V_k is a sequence of real numbers, the following symmetry applies to the complex Fourier components:

$$S(-f_{IF}) = [S(f_{IF})]^* \quad , \quad f_{IF} \leq \frac{f_{SR}}{2} \quad (Nyquist\ Limit) \qquad (1.3.3)$$

Therefore, the total power sum on the right side in (1.3.2) can be written as:

$$P_m = \frac{2}{{N_{FFT}}^2} \sum_{n=1}^{N_{FFT}/2} |S_n|^2 = \sum_{n=1}^{N_{FFT}/2} P_n \qquad (1.3.4)$$

Thus the power spectrum is computed from the Fourier transform in the following way:

$$P_n = \frac{2|S_n|^2}{{N_{FFT}}^2} \quad for \quad 1 \leq n \leq \frac{N_{FFT}}{2} \qquad (1.3.5)$$

This power spectrum (also called the periodogram estimate) is only defined at positive frequencies below the Nyquist frequency limit.
The power normalized ranging Fourier transform follows from (1.3.5):

$$\tilde{S}_n = \frac{S_n}{N_{FFT}} \quad for \quad 1 \leq n \leq N_{FFT} \qquad (1.3.6)$$

1.4 Doppler Spectrum

Sampling a chirp sequence leads to a power spectrum P_n (see equation (1.3.3)) with n corresponding to a certain IF frequency f_{IF}, which is linearly related to a range R via equation (1.2.1). For one final ranging sample, consisting of several hundred range bins (one for each altitude layer), a few thousand chirps (meaning power spectra) are integrated over time. Therefore, for each range bin n (corresponding to a certain altitude layer) there is a time series of normalized spectral Fourier components

$$\tilde{S}_n(t_i) \quad , \quad t_0 = 0\ s,\ t_1 = T_c,\ t_2 = 2T_c, \ldots,\ t_k = kT_c \qquad (1.4.1)$$

The time series sampling interval is T_c. Another Fourier transform (often called the Doppler or 2D transform) over this time series reveals Doppler frequency shifts caused by moving particles in the scattering volume at the altitude related to range bin n. These Doppler frequency shifts linearly depend on the particle velocities v:

$$\frac{f_d}{f_t} = \frac{2 \cdot v}{c} \quad , \qquad f_t = transmitter\ frequency \quad (1.4.2)$$

The Doppler transform is given by:

$$\tilde{S}_n(t) \quad ==> \quad DT_n(f_d) \qquad (1.4.3)$$

with maximum Doppler frequency $f_{dm} = 1/2T_c$. If D_{FFT} is the number of samples in (1.4.1), the Doppler frequency resolution is:

$$\delta f_d = \frac{2 \cdot f_{dm}}{D_{FFT}} \qquad (1.4.4)$$

In contradiction to the ranging FFT in (2.3.1), the input time series for the Doppler FFT is not an array of real, but complex numbers. Therefore, the symmetry relation in (1.3.3) does not hold. The transform in (2.4.3) consists of D_{FFT} independent complex numbers:

$$DT_n(f_d)\ , \quad -f_{dm} \le f_d \le f_{dm} \qquad (1.4.5)$$

The Doppler frequency spectrum is then computed as the normalized power spectrum of the transform in (2.4.3):

$$DS_n(f_d) = \frac{|DT_n(f_d)|^2}{D_{FFT}{}^2}\ , \quad -f_{dm} \le f_d \le f_{dm} \qquad (1.4.6)$$

It should be noticed, that the integral over the Doppler spectrum equals P_n in (1.3.5):

$$P_n = \sum_{-f_{dm}}^{f_{dm}} DS_n(f_d) \qquad 12.4.7)$$

There is a restriction for the maximum Doppler velocity $f_{dm} = 1/2T_c$, which is caused by the fact that T_c cannot be chosen to be arbitrarily short. This problem is often referred to as the ‘Doppler dilemma’ in radar technology and is relevant for both, FMCW and pulsed radars. After a frequency chirp has been sent into the atmosphere, its echo will take about 70 µs for a maximum range of 10 km to complete the round trip. During that period, the radar should not emit another chirp (or pulse in the case of pulsed radars), because otherwise the echoes of two adjacent chirps will overlap in time and cannot be separated anymore. After this 70 µs ‘dead time’ the chirp has to last for at least another 30 µs to give time for measuring an FFT sample. Therefore, the minimum chirp repetition time is about 100 µs corresponding to a maximum Doppler velocity f_{dm} of 5 kHz. By using equation (1.4.2) we end up with the following restriction for v_m:

$$v_m = \frac{c\, f_{dm}}{2\, f_t} \qquad (1.4.7)$$

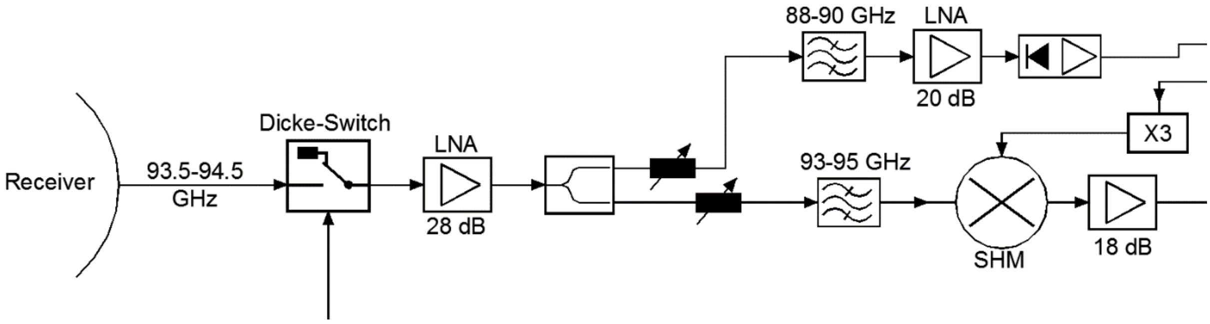
This leads to a v_m of 75 m/s for an X-Band radar ($f_t = 10\ GHz$), 21.5 m/s for a K-Band radar ($f_t = 35\ GHz$) and 8 m/s for a W-Band radar ($f_t = 94\ GHz$). Beyond this velocity range the Doppler spectrum becomes prone to aliasing effects, meaning that spectral components exceeding the maximum velocity will be mirrored to the opposite side of the spectrum. This effect can be easily corrected when observing in zenith direction by starting at the cloud top, where the particle velocities are close to zero and processing the spectra from top to bottom while maintaining the continuity of moving directions. The real problem occurs if the radar is scanning in elevation without having a range bin with a zero reference spectrum to start with. Another obvious problem occurs with very broad Doppler spectra exceeding the total velocity range of $2 \cdot v_m$.

1.5 Radar Receiver Calibration

P_n in (1.3.5) is the power spectrum at the input of the ADC board. Finally we are interested in the power spectrum P_{Rn} incident to the radar antenna (see equation (1.1.5)). The factor between P_n and P_{Rn} is called the receiver Gain:

$$P_n = G_{rn}\, P_{Rn} \qquad (1.5.1)$$

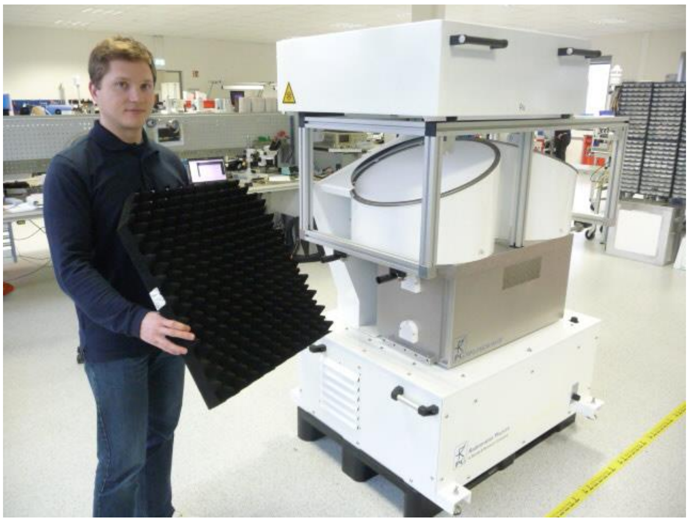
The receiver gain should also include all losses in the feed horn and antenna system.
The radar receiver is equipped with a Dicke switch calibration device located at the receiver input:



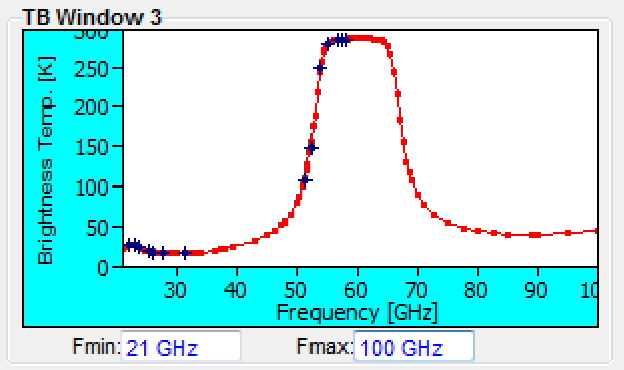
This device can switch the receiver input between the receiver antenna and an internal black body termination (attenuator), which is stabilized to a constant physical temperature T_{DS}.

In an absolute radar calibration the receiver is sequentially terminated with two absolute calibration standards in front of the receiver's Cassegrain antenna system. This can be realized in two ways:

1. Using two external black body targets, one at ambient temperature T_H and one at liquid nitrogen temperature (LN2) T_C.



2. Using the (cloud free) sky as a cold target and get the sky brightness temperature from another measurement, for instance a RPG-HATPRO radiometer. This way the calibration of the other instrument can be transferred to the radar. For the hot (ambient) target the radar can use the internal Dicke switch, which has been calibrated by a previous absolute calibration with an external ambient target (the Dicke switch is considered to be a secondary standard). In this scenario the relatively large external targets as used in 1. are not required at all.



Spectrum retrieval output (red curve) of a RPG-HATPRO radiometer for the frequency range [20 GHz, … , 100 GHz]. The 90 GHz brightness temperature can be used to calibrate the RPG-FMCW-94 cloud radar.

Whatever method is chosen, both of them result in a hot / cold calibration with two calibration standards at temperatures T_H and T_C.

A special feature of RPG-FMCW-94 cloud radars is that they include a passive broad band (2 GHz bandwidth) channel operated at a centre frequency of 89 GHz. This channel is very useful to provide information about the integrated liquid water path (LWP) in cloud observations. The frequency spacing from the radar channels around 94 GHz is wide enough to allow for an effective low pass filtering, protecting the 89 GHz radiometer from the strong signals of the radar operation at 94 GHz.
Therefore, an absolute calibration has to calibrate all radar channels plus the passive 89 GHz channel (in the case of an RPG-FMCW-94 cloud radar).

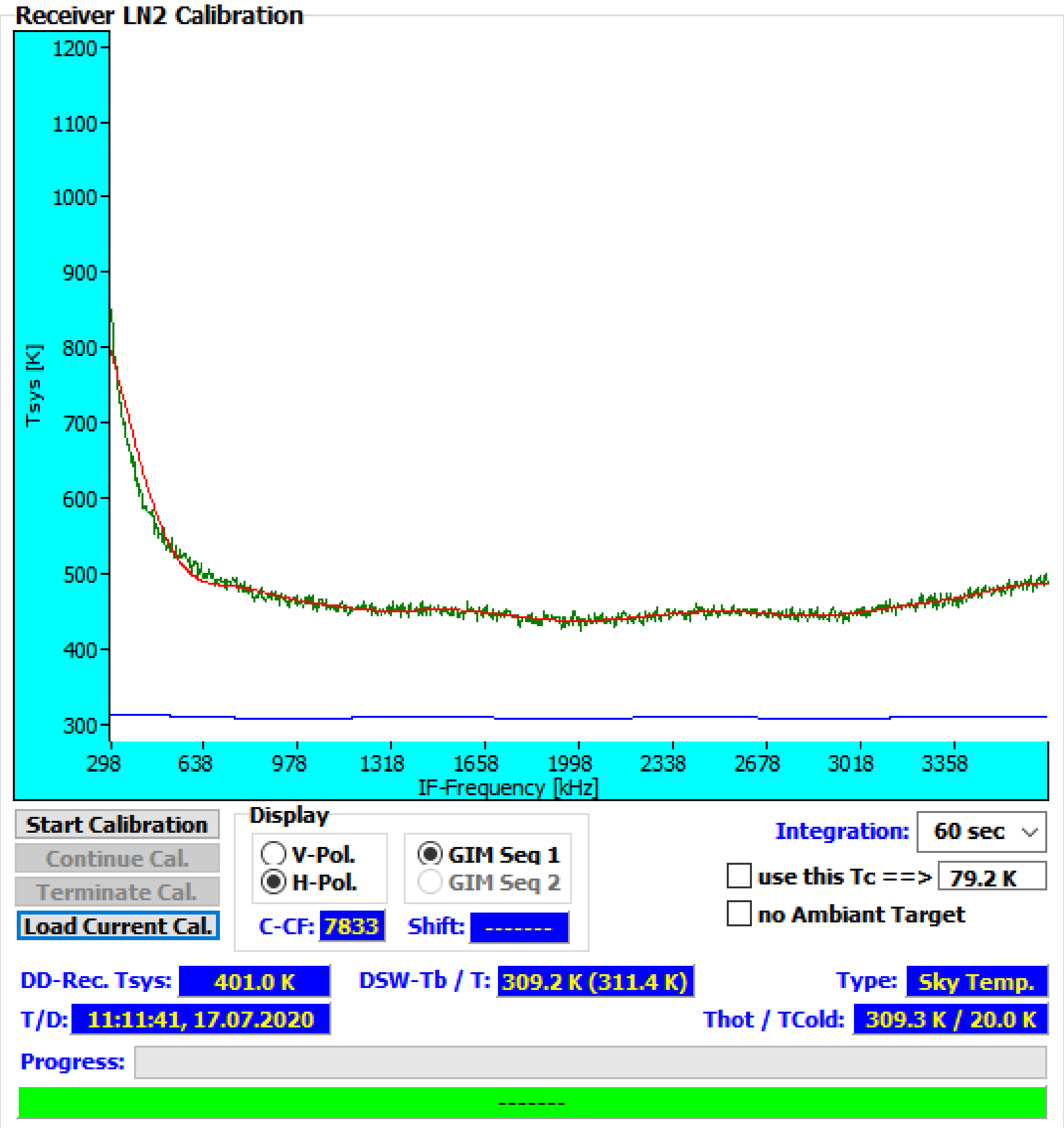
The first step is to determine the system noise temperature of each individual radar channel n (meaning for each bin in the IF spectrum) and the passive channel:

$$P_{nH} = G_n(T_{rn} + T_H) \quad , \quad P_{pH} = G_p(T_p + T_H) \qquad (1.5.2a)$$
$$P_{nC} = G_n(T_{rn} + T_C) \quad , \quad P_{pC} = G_p(T_p + T_C) \qquad (1.5.2b)$$

The index n represents all radar IF channels while the index p represents the passive 89 GHz channel. The T_{rn} are the radar receiver channel's system noise temperatures (the intrinsic noise contribution from the radar receiver channels) and T_p is the corresponding temperature for the passive broad band channel at 89 GHz. G_n and G_p are the receiver gains and P_{nH} , P_{nC} and P_{pH} , P_{pC} are the measured power spectrum numbers of equation (1.5.1) measured on the hot (ambient temperature) and cold targets respectively. From equations (1.5.2a) and (1.5.2b) all receiver gains and system noise temperatures can be computed.
We have now separated the radiometric contributions of the receiver from those of the external signals (sky, LN2 target, ambient target).
In a second step the internal Dicke switch is turned on (it terminates the receiver input with the internal black body standard at temperature T_D). By using the receiver's system noise and gain parameters calculated in the first step, the Dicke switch's brightness temperature T_{DS} can be determined. This temperature should be very close to the physical Dicke switch temperature:

$$T_{DS} = \frac{P_{pD}}{G_p} - T_p \qquad (1.5.3)$$

Both temperatures T_D and T_{DS} are stored in the calibration file for later use.



The red line represents a fit to each radar channel's system noise temperature.

The gain factors G_n and G_p are not the receiver Gain factors referred to in equation (1.5.1) because they link the power spectrum values to brightness temperatures but not to the power levels measured at the receiver's antenna input. The transformation of the brightness temperature gain factors to power related gain factors of equation (1.5.1) can be performed by applying the Rayleigh Jeans approximation. With $\tilde{P}_{nT}$ being the power received by the radar receiver channel n at the antenna input when the receiver is terminated by a target of brightness temperature T, this power is given as:

$$\tilde{P}_{nT} = k_B B_n T \quad , \quad B_n = \frac{f_{SR}}{N_{FFT}} = \frac{1}{T_c} \quad (radar\ channel\ bandwidth) \qquad (1.5.4)$$

with k_B being the Boltzmann constant. The Rayleigh Jeans approximation validity is justified for the high brightness temperatures observed by the radar $(k_B T \gg h\nu \ or \ T \gg 4.5\ K\ \ at\ \ 94\ GHz)$ at frequencies below 100 GHz. The required gain factors G_{rn} of equation (1.5.1) can then be computed from the brightness temperature related gain factors G_n:

$$G_{rn} = \frac{P_{nT}}{\tilde{P}_{nT}} = \frac{G_n}{k_B B_n} \qquad (1.5.5)$$

It should be noticed that the G_n factors' unit is $[W/K]$ while the G_{rn} are dimensionless (as usual, in the order of 80 dB). The G_{rn} factors include ALL corrections due to antenna and feed horn losses, as well as the complete chain of receiver amplifiers and mixer conversion losses. No other corrections need to be applied to link the measured power spectrum levels of each radar bin to the real power received at the antenna input within the bin's bandwidth B_n. The radar equation in (1.1.5) can be written in the more detailed form for a single bin of the IF spectrum with P_n being the received power in this bin (S_{rn} is the radar scaling factor):

$$\frac{P_n}{P_T} = \frac{G_{rn} C_R}{F_R(R)} Z_e \lll==\ggg Z_e = \frac{P_n F_R(R)}{P_T G_{rn} C_R} \equiv S_{rn} P_n F_R(R) \; with \; F_R(R) \equiv \frac{R^2}{\delta R} L_{ovlp}(R) \quad (1.5.6)$$

L_{ovlp} is the beam overlap loss given in equation (2.8.1) and $F_R(R)$ is the range factor.
During radar operation, the receiver periodically (every 10 minutes) performs a 'zero calibration' cycle by turning the Dicke switch on and measuring one sample on the Dicke switch termination black body instead of the sky (the transmitter is turned off during this cycle). Because this calibration is a single point calibration, the radar software assumes the T_n to be constant over time. The G_{rn} are readjusted according to equations (1.5.2a, using T_{DS} instead of T_H) and (1.5.6). This way a long term stability of the radar's receiver accuracy can be maintained. **We recommend a repetition of the absolute calibration every 6 months in order to re-calibrate the radar's channel system noise temperatures T_n.**
The integration time on each of the calibration targets during the absolute calibration process is 60 seconds. According to the radiometer formula, the receiver noise measured in K for a single IF bin of typically 4 kHz bandwidth and about 500 K system noise temperature is:

$$\Delta T_n = \frac{T_n}{\sqrt{60\,s \cdot 4000\,Hz}} \approx 1\,K\,RMS \;=\gg\; \Delta T_{p-p} \approx 4\,K \qquad (1.5.7)$$

The peak-to-peak noise is in the order of 4 K only, showing that an absolute calibration with passive black body targets of brightness temperatures in the range [77K, 300K] is well suited to calibrate a radar receiver with high accuracy. It should be noticed, that the receiver sensitivity achievable for radars operated at K-band or X-band is significantly higher compared to W-band. Therefore, the proposed method is even better suited for these radars, assuming a careful receiver design is focusing on receiver sensitivity optimization.

1.6 Radar Sensitivity Limit

The radiometer formula also provides the radar's sensitivity limit when the temperature standard deviation on the left side of equation (1.5.7) is converted to power using the Rayleigh-Jeans approximation. The brightness temperature T_n in (1.5.7) for the sensitivity limit is composed of the radar receiver noise temperature T_{rn} of equations (1.5.2) and the passive atmospheric emission at operating frequency. The later contribution can be derived from the passive receiver channel sky brightness temperature T_{sky}:

$$\Delta T_n = \frac{T_{rn} + T_{sky}}{\sqrt{\tau \cdot B_n}} \qquad (1.6.1)$$

Here τ is the chirp sequence integration time and B_n is the channel bandwidth of (1.5.4). In order to derive the sensitivity limit SL_n in Z_e units we combine equations (1.5.4) and (1.5.6) to get

$$\frac{2\,k_B B_n \Delta T_n}{P_T} = \frac{C_R}{F_R(R)}\,SL_n \quad \ll=\gg \quad SL_n = \frac{2\,k_B B_n \Delta T_n}{P_T C_R} F_R(R) \qquad (1.6.2)$$

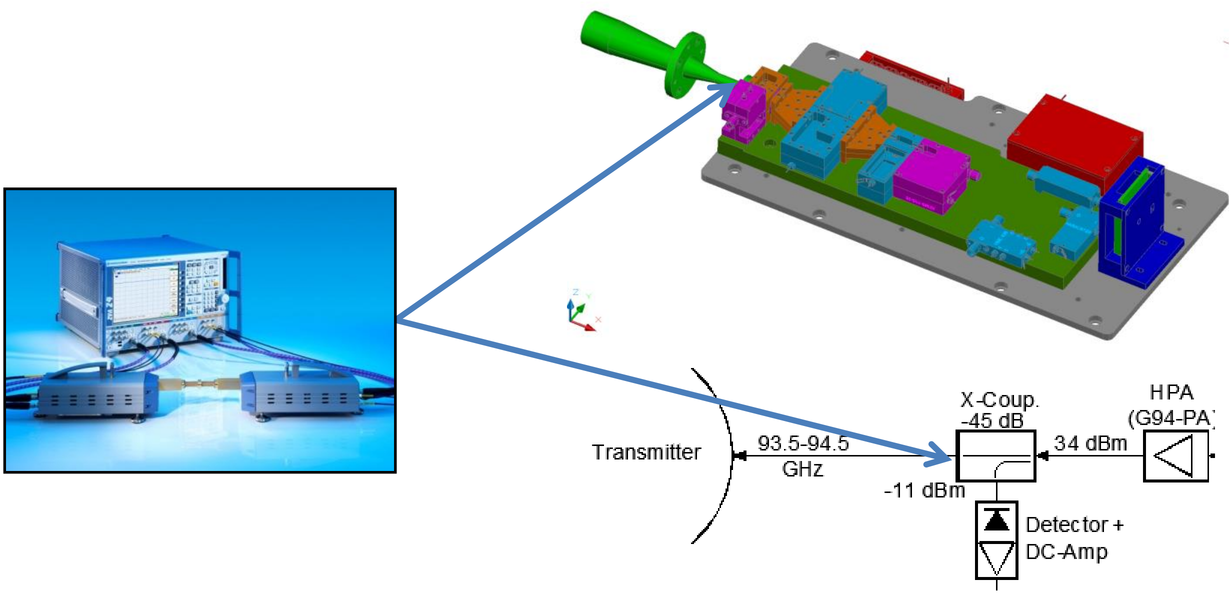
Here we have set the detection threshold to $2 \cdot \Delta T_n$.
Substituting (1.6.1) into (1.6.2) yields:

$$SL_n = \frac{2\,k_B \sqrt{B_n} F_R(R)}{P_T C_R \sqrt{\tau}} \left(T_{rn} + T_{sky}\right) \quad (1.6.2)$$

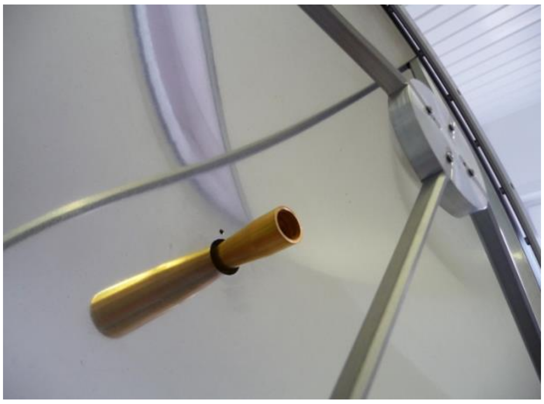
The SL_n array is stored to the level 0 and level 1 data files for each radar sample. The array is calculated independently for different polarisations because the T_{rn} are related to separate receivers in dual pol. radars.

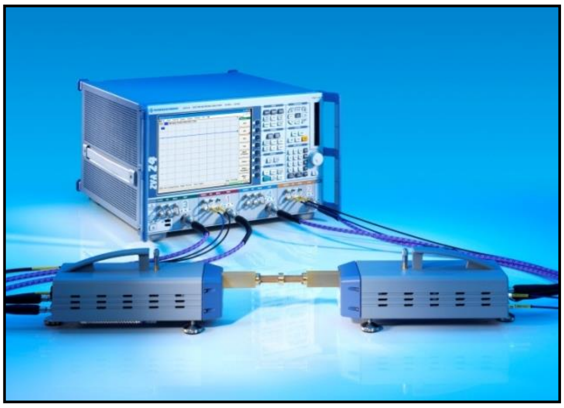
1.7 Radar Transmitter Calibration

In order to complete the radar's internal calibration, also the transmitter power needs to be characterized as good as possible. In addition, the transmitter power should be continuously monitored over time to ensure that changes in output power are captured and taken into account in the radar calibration procedure.

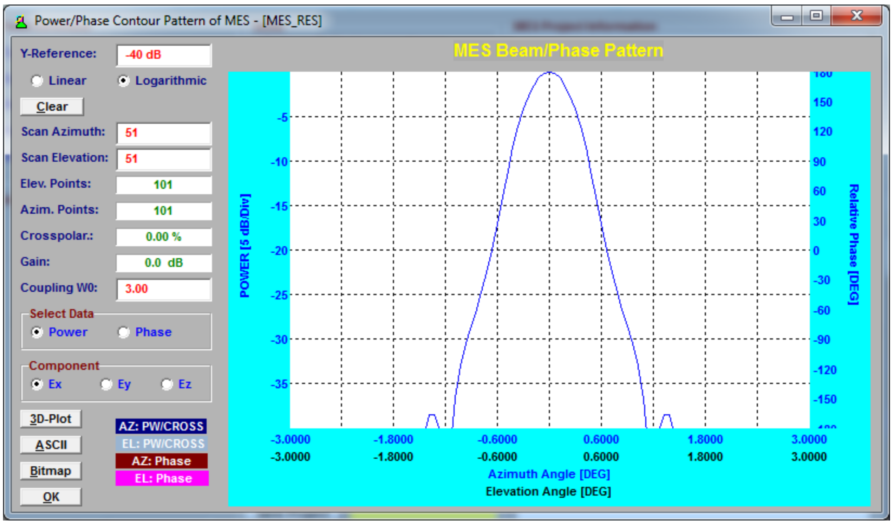


In RPG radars, the receiver and transmitter RF components are all thermally stabilized within a few hundred mK. This ensures a stable transmitter power generation and power measurement. Once thermally stabilized, the maximum output power is determined by a retraceable precision Rhode & Schwarz W-band power head (accuracy 0.1 dB). A small fraction of the transmitter power is coupled to an RF detector for power monitoring. Care was taken to ensure that this detector always operates in its linear detection regime. While the transmitter power is measured by the power head, the detector reading is monitored to determine its calibration factor. This measurement does not take into account the feed horn and antenna losses. These losses need to be determined separately.

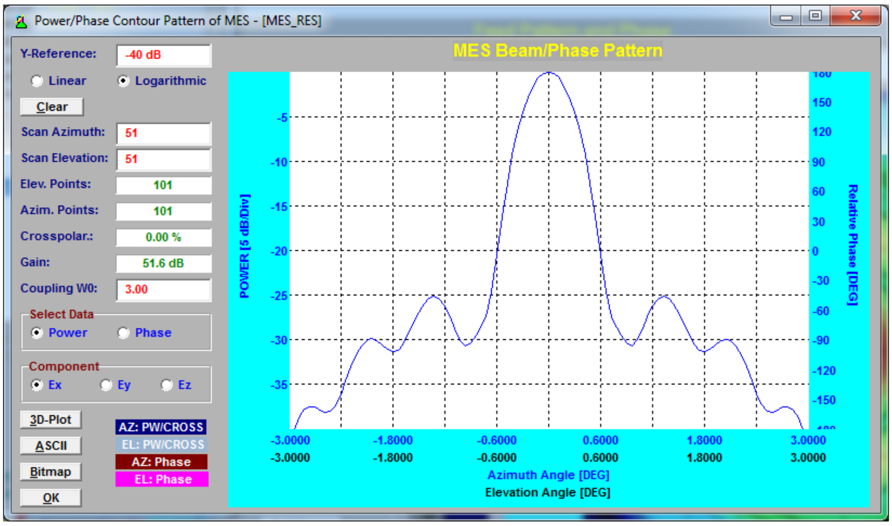




The feedhorn insertion loss has been measured by a Rhode & Schwarz Network Analyzer to be 0.3 +/- 0.05 dB ($L_f = 1.072$).



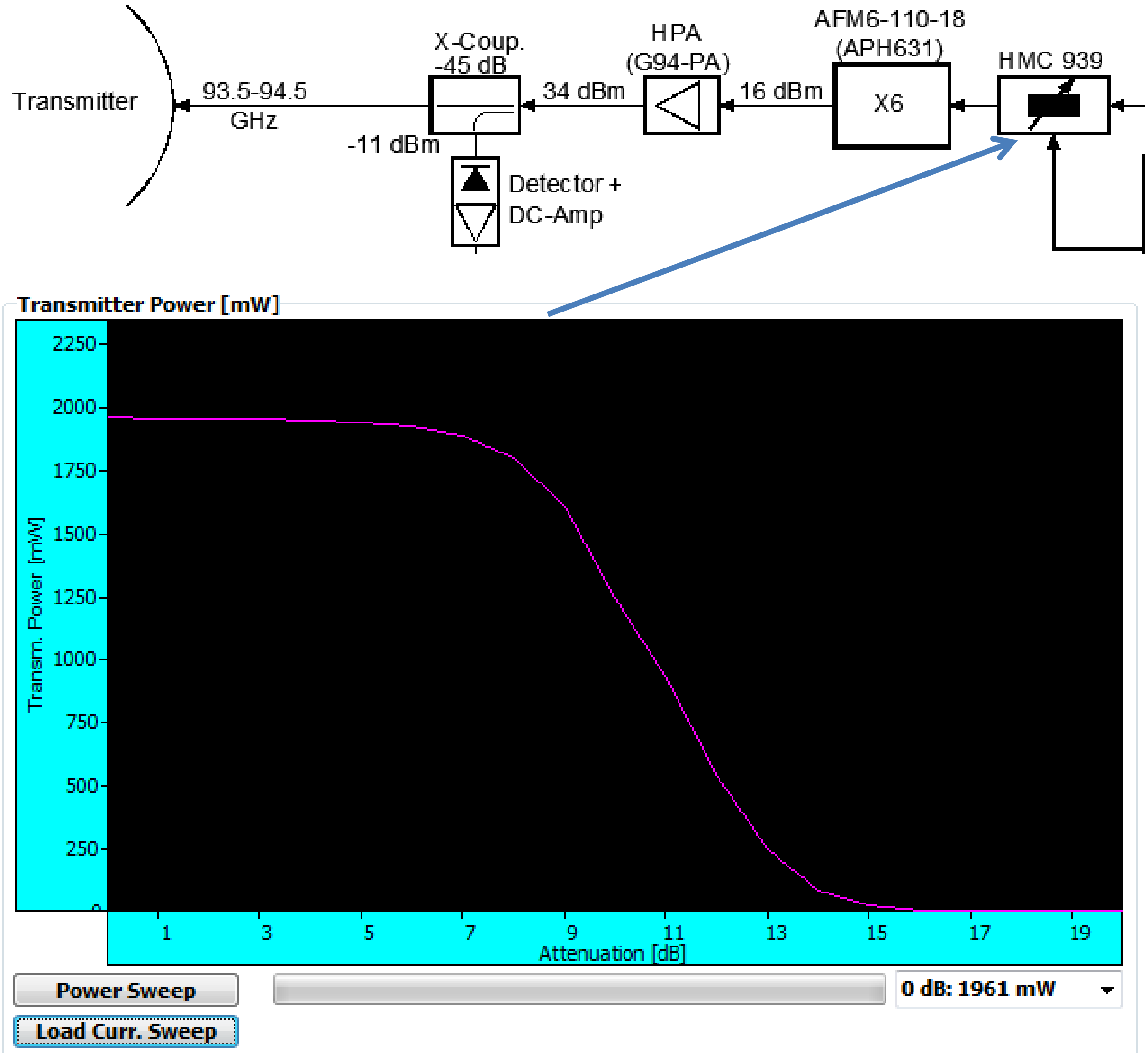
Antenna pattern without sub-reflector blockage



Antenna pattern including sub-reflector blockage

The Cassegrain antenna system losses are dominated by sub-reflector blocking. A detailed analysis reveals a power loss of 11% ($L_{sr} = 1.11$) by the sub-reflector shadowing the output beam.
In addition, the sub-reflector changes the beam pattern compared to the unblocked antenna. The Cassegrain antenna gain needs to be corrected accordingly.

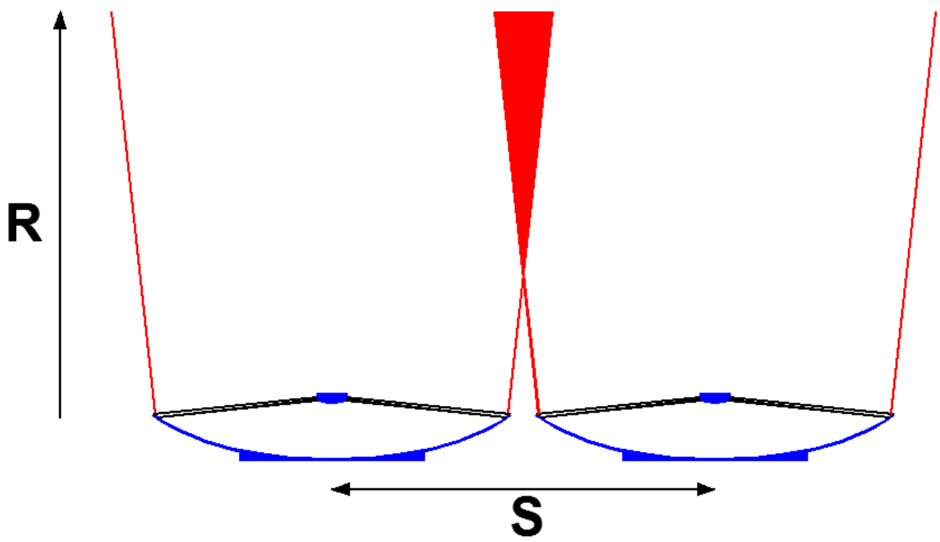
The radar receiver is operated under a high dynamic range of input signal levels. On one hand its sensitivity must be as high as possible to detect faint objects like liquid water clouds at high altitudes. On the other hand the radar should also respond linearly to precipitation events where large droplets located close to the radar reflect a high fraction of the transmitter power back to the receiver. In order to cope with the high dynamic range requirements, the radar's transmitter power can be reduced in digital steps by a maximum of 16 dB.



For this purpose a digital 5 bit attenuator is included in the transmitter's signal path. During radar operation the input signal level to the receiver is measured and the transmitter power is automatically adjusted to protect the receiver from saturation effects.
A power sweep is performed every hour to monitor changes in output power. These changes are then used to tune P_T in the radar equation (1.5.6).

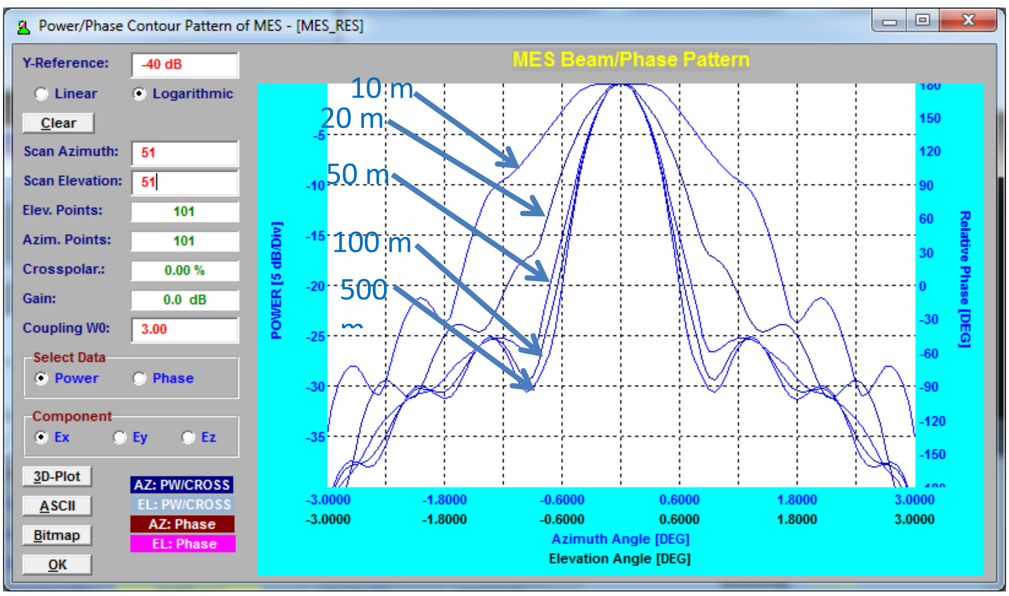
1.8 Beam Overlap Correction

All FMCW-radars with high dynamic range are realized as bi-static systems with a separate antenna for transmitter and receiver. This is mandatory because the transmitter is always active during the receiver operation and a power transfer from the transmitter to the receiver must be attenuated by at least 80 dB to protect the receiver from saturation. If the receiver would share the same antenna with the transmitter, it would be technically impossible to separate the transmitted power from the received power by a 80 dB suppression factor.



The obvious disadvantage of the bi-static configuration is the imperfect beam overlap close to the radar. If the beam pattern strongly overlaps with a Gaussian beam pattern (which is fulfilled by RPG's radar antennas to 95%), the loss caused by an incomplete overlap is:

$$L_{ovlp} = e^{\frac{2\ln(2)S^2}{\theta^2 R^2}} \qquad (1.8.1)$$



Beam pattern variation in the near field region.

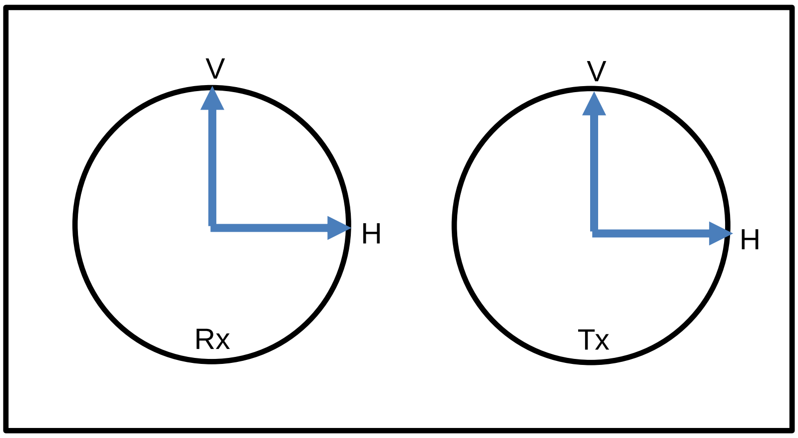
For an antenna separation S of 568 mm and a HPBW of 0.48° (RPG-FMCW-94 parameters) this leads to a loss of 10% at 300 m distance from the radar. However, this loss is not critical because it only affects reflections from objects close to the radar which are relatively strong. The RPG-FMCW radars are capable of detecting reflectivity levels of -60 dBz down to 50 m altitude.

Another important issue for all radars (not only bi-static systems) is related to the antenna far field, because the antenna gain used in the radar equation (1.1.5) is usually calculated or measured in the far field region. In the antenna's near field region the antenna gain G_T deviates from the far field value. Therefore, in order to determine the radar's lowest ranging limit, the far field distance should be checked. This limit is about 50 m for the RPG-FMCW-94 radar.

1.9 Cloud radar polarimetry

1.9.1 Reference Polarisation Basis

For an analysis of polarization states and their transformation a reference basis has to be chosen. Usually the Cartesian reference basis formed by radar antenna feeds is used. The Cartesian reference bases used throughout this manual are shown in the figure below. When the radar is pointed to 0° elevation, i.e. the transmitted signal propagates parallel to the ground, the orientations of unit vectors, forming the basis, correspond to horizontal and vertical directions. Therefore, the unit vectors and corresponding polarization channels are denoted as “horizontal” (H) and “vertical” (V), respectively. The reference basis are related to the radar antennas and therefore at 90° elevation (radar pointing to zenith), the denotations H and V are kept the same, although the correspondence with respect to the ground is violated.



1.9.2 Representation of Polarisation States

It is known that polarization states and transformations of electromagnetic waves can be described with the Jones formalism. Taking into account that the coherency bandwidth of the radar hardware is wider than the transmitted signal bandwidth (narrowband approximation), the polarization state of a radar signal in the far field (plane wave approximation) can be represented in the Cartesian basis by the following Jones vector up to an absolute phase shift:

$$\mathbf{e} = \begin{bmatrix} \dot{E}_H \\ \dot{E}_V \end{bmatrix} = \begin{bmatrix} E_H \\ E_V e^{j\Delta\Phi} \end{bmatrix}. \qquad (1.11.1)$$

$\dot{E}_H$ and $\dot{E}_V$ are complex projections of the electric field to the unit vectors forming the reference basis, E_H and E_V are the magnitudes of $\dot{E}_H$ and $\dot{E}_V$, respectively, $\Delta\Phi = \Phi_H - \Phi_V$ is the phase difference between $\dot{E}_H$ and $\dot{E}_V$, Φ_H and Φ_V are the absolute phases of $\dot{E}_H$ and $\dot{E}_V$, respectively.

The wave $\mathbf{e}$ can be represented in an arbitrary orthogonal polarization basis $(\mathbf{e}_1; \mathbf{e}_2)$:

$$\mathbf{e}' = \mathbf{Re}, \qquad (1.11.2)$$

where $\mathbf{R}$ is a 2 x 2 rotational operator. Columns of the matrix $\mathbf{R}$ represent unit vectors $\mathbf{e}_1$ and $\mathbf{e}_2$ in the original basis and in general has the following form:

$$\mathbf{R} = \begin{bmatrix} \dot{R}_1 & -\dot{R}_2^* \\ \dot{R}_2 & \dot{R}_1^* \end{bmatrix}, \quad (1.11.3)$$

Where $\dot{R}_1$ and $\dot{R}_2$ are Cayley-Klein Parameters:

$$\dot{R}_1 = \cos\varepsilon\cos\theta - j\sin\varepsilon\sin\theta$$
$$\dot{R}_2 = \cos\varepsilon\sin\theta - j\sin\varepsilon\cos\theta$$

Where ε and θ are ellipticity and orientation angles of the new basis with respect to the original one. Note that the determinant of the matrix $\mathbf{R}$ is 1, i.e. the transformation Eq. (1.11.2) does not change the power of the wave $\mathbf{e}$.

1.9.3 Polarimetric Configurations of the Radar

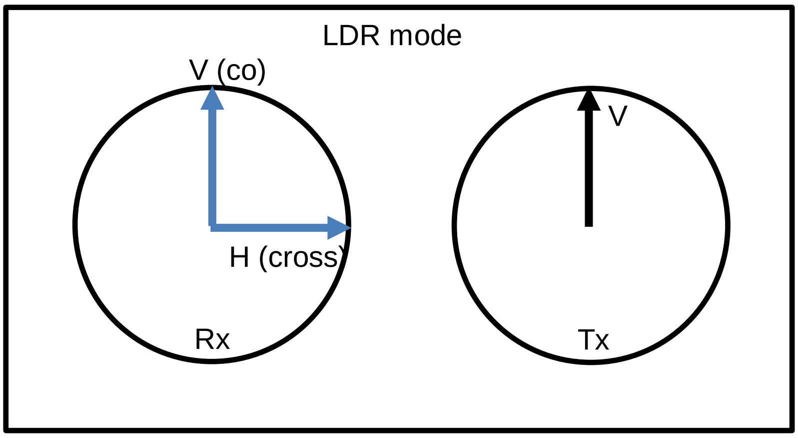
There are two polarimetric options available for RPG cloud radars: Linear depolarization ratio (LDR) mode and Simultaneous Transmission Simultaneous Reception (STSR) mode. In the LDR mode the transmitter signal $\dot{E}_\Sigma$ is radiated from the vertical channel:

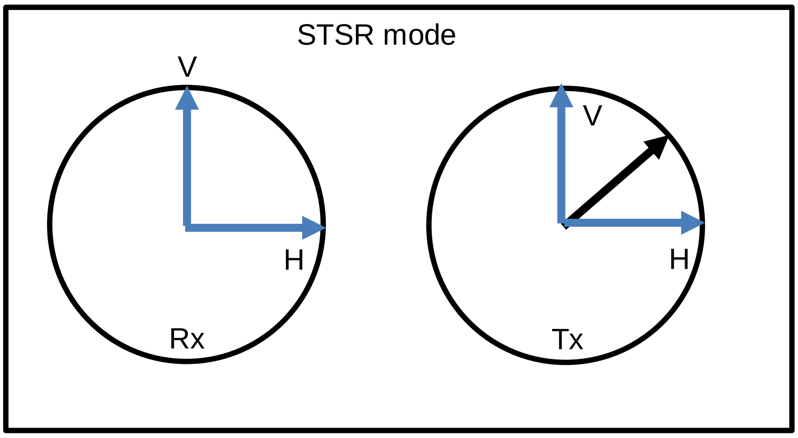
$$\mathbf{e}_t = \dot{E}_\Sigma \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \quad (1.11.4)$$

In the STSR mode the transmitter antenna is rotated by $\theta = 45°$ clockwise and therefore, the radar transmits horizontal and vertical components simultaneously:

$$\mathbf{e}_t = \begin{bmatrix} \cos 45° & \sin 45° \\ -\sin 45° & \cos 45° \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} \dot{E}_\Sigma = \frac{\dot{E}_\Sigma}{\sqrt{2}} \begin{bmatrix} 1 \\ 1 \end{bmatrix} \quad (1.11.5)$$

As the phase difference $\Delta\Phi$ is 0 deg, the transmitted wave has linear polarization with 45° orientation.





1.9.4 Statistical Properties of the Received Signals

In both polarimetric configurations the radar has two coherent receivers and thus allows for simultaneous reception and analysis of the complex amplitudes $\dot{E}_H$ and $\dot{E}_V$ of received signals in the horizontal and vertical channels, respectively. Using a sequence of $\dot{E}_H$ and $\dot{E}_V$ for a certain range bin, complex spectra $\dot{S}_H(f_k)$ and $\dot{S}_V(f_k)$ are calculated as explained in chapters 1.3 and 1.4 of the manual. Here f_k is the Doppler frequency shift of the spectrum line k, the dot over a denotation stands for the complex number. In the following, only a single spectral component is considered for brevity.

In general, the orthogonal components $\dot{S}_H$ and $\dot{S}_V$ of the received electromagnetic wave $\mathbf{e}_r$ vary with time. Statistical properties of the signals in the horizontal and vertical channels can be described using the spectral form of the coherency matrix $\mathbf{B}$:

$$\mathbf{B} = \langle \mathbf{e}_r \mathbf{e}_r^{\dagger} \rangle = \left\langle \begin{bmatrix} \dot{S}_H \\ \dot{S}_V \end{bmatrix} \begin{bmatrix} \dot{S}_H^* & \dot{S}_V^* \end{bmatrix} \right\rangle = \begin{bmatrix} \langle \dot{S}_H \dot{S}_H^* \rangle & \langle \dot{S}_H \dot{S}_V^* \rangle \\ \langle \dot{S}_V \dot{S}_H^* \rangle & \langle \dot{S}_V \dot{S}_V^* \rangle \end{bmatrix} = \begin{bmatrix} B_{hh} & \dot{B}_{hv} \\ \dot{B}_{vh} & B_{vv} \end{bmatrix} \quad (1.11.6)$$

† is the Hermitian transpose sign, * is complex conjugation sign, <> is averaging over time. The main-diagonal components of the coherency matrix $B_{hh} = |\dot{S}_h|^2$ and $B_{vv} = |\dot{S}_v|^2$ correspond to power spectra in the horizontal and vertical receiving channels, respectively. The off-diagonal components $\dot{B}_{hv}$ and $\dot{B}_{vh}$ are covariancies of the signals in the vertical and horizontal receiving channels. In the single polarization configuration of the radar only the element B_{vv} is calculated and analyzed. A dual polarization configuration performs the computation of the elements B_{hh} , B_{vv} , and $\dot{B}_{hv}$. Taking into account that $\dot{B}_{hv} = \dot{B}_{vh}^*$, storing of the element $\dot{B}_{vh}$ is not required. Note, even though the receiving polarization basis is the same for LDR and STSR modes, the polarization states of the transmitted signals in these modes are different and therefore the elements of the coherency matrix have different information content.

A coherency matrix $\mathbf{B}$ can be decomposed to a sum of stable $\mathbf{K}$ and fluctuating $\mathbf{J}$ parts:

$$\mathbf{B} = \mathbf{J} + \mathbf{K} \quad (2.11.7)$$

$$\mathbf{K} = \langle \mathbf{e}_r \rangle \langle \mathbf{e}_r \rangle^{\dagger} = \begin{bmatrix} \langle \dot{S}_H \rangle \langle \dot{S}_H^* \rangle & \langle \dot{S}_H \rangle \langle \dot{S}_V^* \rangle \\ \langle \dot{S}_V \rangle \langle \dot{S}_H^* \rangle & \langle \dot{S}_V \rangle \langle \dot{S}_V^* \rangle \end{bmatrix}. \qquad (1.11.8)$$

In the case of non-coherent scattering, which is the case when cloud particles are observed, absolute phases of $\dot{S}_H$ and $\dot{S}_V$ are distributed uniformly from 0 to 2π. Therefore, their mean values and $\mathbf{K}$ are equal to 0. Note, that in the case of point scatterers or external coherent sources of radiation, $\mathbf{K}$ can differ from 0. A coherent polarization coupling produced by a radar antenna of poor quality can also bias the elements of the matrix $\mathbf{K}$.

As the coherency matrix $\mathbf{B}$ contains only the fluctuating part, the elements B_{ij} can be presented as follows:

$$B_{hh} = \sigma_h^2$$
$$B_{vv} = \sigma_v^2$$
$$\dot{B}_{hv} = \sigma_h \sigma_v \dot{R}_{hv}$$
$$\dot{B}_{vh} = \sigma_h \sigma_v \dot{R}_{hv}^*$$

Where σ_h and σ_v are standard deviations of the horizontal and vertical components, respectively, and $\dot{R}_{hv}$ is the complex correlation coefficient. Absolute value of $\dot{R}_{hv}$ shows statistical relation between the horizontal and vertical components, while its argument corresponds to the phase shift between horizontal and vertical components $\Delta\Phi$.

1.9.5 Non-Polarized and Fully-Polarized Coherency Matrix Components

The coherency matrix $\mathbf{B}$ can be decomposed as follows:

$$\mathbf{B} = A\mathbf{I} + \begin{bmatrix} C & \dot{D} \\ \dot{D}^* & F \end{bmatrix}, \qquad (1.11.9)$$

with the condition:

$$CF - |\dot{D}|^2 = 0. \qquad (1.11.10)$$

Here $\mathbf{I}$ is the 2 x 2 unit matrix. Parameters $A, \dot{D}, F$ can be found as follows:

$$A = \frac{1}{2}\left(\mathrm{Sp}\mathbf{B} - \left[\mathrm{Sp}^2\mathbf{B} - 4\det\mathbf{B}\right]^{1/2}\right),$$
$$C = \frac{1}{2}\left(B_{hh} - B_{vv} + \left[\mathrm{Sp}^2\mathbf{B} - 4\det\mathbf{B}\right]^{1/2}\right),$$
$$F = \frac{1}{2}\left(B_{vv} - B_{hh} + \left[\mathrm{Sp}^2\mathbf{B} - 4\det\mathbf{B}\right]^{1/2}\right).$$

The horizontal and vertical components of the first term of the Eq. (1.11.9) are not correlated ($|\dot{R}_{hv}|$=0) and therefore this term corresponds to the non-polarized part. The second term describes the fully-polarized part as the corresponding correlation $|\dot{R}_{hv}|$ is equal to 1. The degree of polarization can be found as a ratio of the non-polarized power to the total power of the electromagnetic wave:

$$\mu = \frac{2A}{\mathrm{Sp}\mathbf{B}} \quad (1.11.11)$$

1.9.6 Coherency Matrix in Different Basis

Using Eqs. (1.11.2) and (1.11.6) the coherency matrix $\mathbf{B}$ can be represented in another polarization basis using the following unitary transformation:

$$\mathbf{B}' = \left\langle \mathbf{Re}_r (\mathbf{Re}_r)^\dagger \right\rangle = \left\langle \mathbf{R}\mathbf{e}_r \mathbf{e}_r{}^\dagger \mathbf{R}^\dagger \right\rangle = \mathbf{RBR}^\dagger \quad (1.11.12)$$

Note, that transformation Eq. (1.11.12) does not change the non-polarized part as it is proportional to the unity matrix.

1.9.7 STSR sensitivity issue

In LDR mode most of the received power is concentrated in the vertical (co-polarized) receiving channel. The power in the horizontal (cross-polarized) channel is about 10 to 30 dB lower. Taking into account that the noise power is the same in both orthogonal channels, the total signal power is not calculated. Only the signal power in the vertical channel is used for scatter detection. In the case of the STSR configuration, the signal generated by the transmitter is “split” to two channels. Therefore, the amount of power received in the vertical channel is reduced by a factor of 2 in comparison to the single polarization and LDR configurations. This leads to a sensitivity loss of 3 dB if only the vertical channel is used for the reflectivity calculation.
The cloud radar sensitivity corresponds to a threshold above the mean noise power density:

$$T = Q\sigma_n,$$

Where σ_n is the standard deviation of the noise power density, and Q is a parameter related to a certain false alarm rate (typically 3 to 4).
In the STSR-configuration, non-coherent averaging of signals in the horizontal and vertical channels gains only 1.5 dB in sensitivity:

$$\sigma_{nav} = \frac{\sigma_n}{\sqrt{2}}.$$

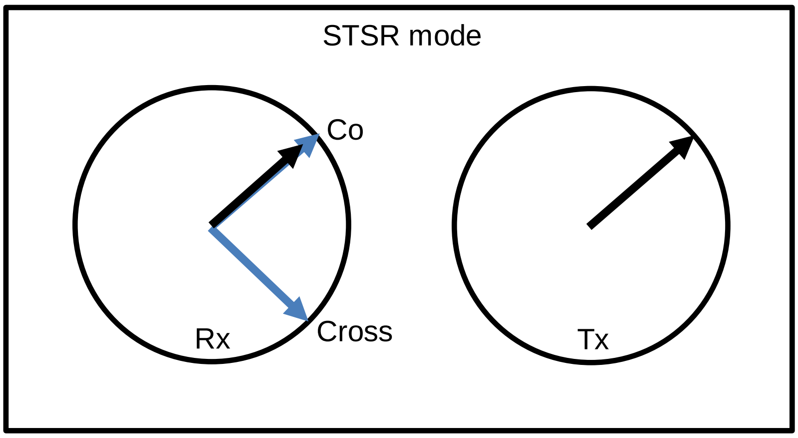
Taking into account that cloud particles do not significantly de-correlate cloud radar signals, the maximization of the power in one of the channels applying Eq. (1.11.12) can gain up to 3 dB in sensitivity. As the radar has the transmitting differential phase of 0°, the maximum power of the returned signal concentrated in the single component corresponds to the ±45° reference basis:

$$\mathbf{B}' = \begin{bmatrix} \cos 45^\circ & -\sin 45^\circ \\ \sin 45^\circ & \cos 45^\circ \end{bmatrix} \begin{bmatrix} B_{hh} & \dot{B}_{hv} \\ \dot{B}_{vh} & B_{vv} \end{bmatrix} \begin{bmatrix} \cos 45^\circ & \sin 45^\circ \\ -\sin 45^\circ & \cos 45^\circ \end{bmatrix} =$$

$$\frac{1}{2} \begin{bmatrix} B_{hh} + B_{vv} - 2\,\mathrm{Re}\,\dot{B}_{hv} & B_{hh} - B_{vv} + 2i\,\mathrm{Im}\,\dot{B}_{hv} \\ B_{hh} - B_{vv} - 2i\,\mathrm{Im}\,\dot{B}_{hv} & B_{hh} + B_{vv} + 2\,\mathrm{Re}\,\dot{B}_{hv} \end{bmatrix} = \begin{bmatrix} B_{xx} & \dot{B}_{xc} \\ \dot{B}_{cx} & B_{cc} \end{bmatrix}$$

For spherical cloud particle scattering the polarization state of the scattered wave is not changed, $B_{hh} = B_{vv}$.
Taking into account Eq. (1.9.10):

The new basis is illustrated in the figure below.



It can be seen, that the element B_{cc} contains all the power of the fully polarized part while the power of the non-polarized part (contains noise and depolarized signal) is still equally splitted to B_{xx} and B_{cc}. This recovers the sensitivity loss up to 3 dB. Therefore,

$$B_{CC} = \frac{1}{2}\left(B_{hh} + B_{vv} + 2\operatorname{Re}(B_{hv})\right) \qquad (1.11.13).$$

contains the combined spectral information with fully recovered sensitivity in STSR-mode.

1.9.8 Polarimetric products

Using the coherency matrix the following spectral polarimetric parameters are calculated for Doppler components where the signal is detected:

Equation	LDR-mode parameter	STSR-mode parameter
1. $(B_{hh} - P_n)/(B_{vv} - P_n)$	Spectral linear depolarization ratio (sLDR)	Spectral differential reflectivity (sZ$_{DR}$)
2. $\dfrac{\lvert\dot{B}_{hv}\rvert}{\sqrt{(B_{hh} - P_n)(B_{vv} - P_n)}}$	Spectral co-cross-channel correlation coefficient (ρ_{cx})	Spectral correlation coefficient (ρ_{hv})
3. $\arg[\dot{B}_{hv}]$	Spectral co-cross-channel differential phase (Φ_{cx})	Spectral differential phase (Φ_{DP})
4. $\dfrac{B_{hh} + B_{vv} - 2\operatorname{Re}[\dot{B}_{hv}] - 2P_n}{B_{hh} + B_{vv} + 2\operatorname{Re}[\dot{B}_{hv}] - 2P_n}$	Not applicable	Spectral slanted linear depolarization ratio (sSLDR)
5. $\dfrac{\lvert B_{hh} - B_{vv} + 2i\operatorname{Im}[\dot{B}_{hv}] - 2P_n\rvert}{\sqrt{(B_{hh} + B_{vv} - 2\operatorname{Re}[\dot{B}_{hv}] - 2P_n)(B_{hh} + B_{vv} + 2\operatorname{Re}[\dot{B}_{hv}] - 2P_n)}}$	Not applicable	Spectral co-cross-channel correlation coefficient in slanted basis

Table 1.9.1: Spectral polarimetric variables

P_n in the formulas above is the mean noise power density.

Summation of the coherency matrix elements over the spectral lines with the detected signal yields the following integrated polarimetric variables:

Equation	LDR-mode parameter	STSR-mode parameter
1. $\sum(B_{hh}-P_n)/\sum(B_{vv}-P_n)$	Linear depolarization ratio (LDR)	Differential reflectivity (Z_{DR})
2. $\dfrac{\left\lvert\sum \dot{B}_{hv}\right\rvert}{\sqrt{\sum(B_{hh}-P_n)\sum(B_{vv}-P_n)}}$	Co-cross-channel correlation coefficient (ρ_{cx})	Correlation coefficient (ρ_{hv})
3. $\arg\left[\sum \dot{B}_{hv}\right]$	Spectral co-cross-channel differential phase (Φ_{cx})	Spectral differential phase (Φ_{DP})
4. $\dfrac{\sum\left(B_{hh}+B_{vv}-2\,\mathrm{Re}\left[\dot{B}_{hv}\right]-2P_n\right)}{\sum\left(B_{hh}+B_{vv}+2\,\mathrm{Re}\left[\dot{B}_{hv}\right]-2P_n\right)}$	Not applicable	Spectral slanted linear depolarization ratio (SLDR)
5. $\dfrac{\sum\left\lvert B_{hh}-B_{vv}+2i\,\mathrm{Im}\left[\dot{B}_{hv}\right]-2P_n\right\rvert}{\sqrt{\sum\left(B_{hh}+B_{vv}-2\,\mathrm{Re}\left[\dot{B}_{hv}\right]-2P_n\right)\sum\left(B_{hh}+B_{vv}+2\,\mathrm{Re}\left[\dot{B}_{hv}\right]-2P_n\right)}}$	Not applicable	Spectral co-cross-channel correlation coefficient in slanted basis

Table 1.9.2: Integrated polarimetric variables

In the STSR mode two additional parameters are calculated. Using the spectral lines induced only by Rayleigh scattering (criterium has to be developed using observations) the specific differential phase shift and differential attenuation are calculated, respectively:

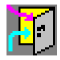
$$K_{DP}=\frac{d\left\langle \arg\left[\sum \dot{B}_{hv}\right]\right\rangle}{2dR} \qquad (1.11.13)$$

$$A=\frac{d\left(\sum(B_{hh}-P_n)/\sum(B_{vv}-P_n)\right)}{2dR} \qquad (1.11.14)$$

Here R is the range.

2 Radar Software

The following conventions are used in this software description:

- Messages generated by the program that have to be acknowledged are printed in red. Example: ***No connection to Radar PC!***
- Button labels are printed in green: ***Delete Program***
- Messages that have to be answered by ***Yes*** or ***No*** are printed in light blue: ***Overwrite the existing file?***
- Labels are printed in grey: ***UTC***
- Names of group boxes are printed in blue. Example: ***El / Az Positioner***
- Names of tags are printed in violet: ***Radar Monitoring***
- Names of menus are printed in black: ***File Transfer***
- Labels of Entry-Boxes are printed in light blue: ***Const. Elev. Angle***
- When a speed button shall be clicked, this is indicated by its symbol: 
- Hints to speed buttons are printed in brown: ***License Manager***
- Selections from list boxes are printed in magenta: ***DD-Rec. TBs***
- Selections from radio buttons or check boxes are printed in dark green: ***Ze***
- File names are printed in orange: ***MyFileName***
- Directory names are printed in dark blue: ***C:\Programs\RPG-FMCW***

2.1 Host Software Installation

2.1.1 Host PC Hardware Recommendations

The hardware recommendations for the host PC (H-PC) are:

- Intel I5 multi-core CPU or larger
- 4 GB free RAM minimum
- High resolution graphics screen (at least 768 vertical points)
- Ethernet interface (mandatory)

2.2.1 Directory Tree

In order to install the host software from a flash drive the following steps should be performed:

- Start the Windows® operating system (make sure you have Administrator rights)
- Start the Windows Explorer®
- Insert the Radar Software flash drive (included in the delivery package)
- In Windows Explorer® click on the flash drive
- Click on the ***RPG-FMCW***-folder (flash drive) and drag the whole folder to ***MY_DIRECTORY*** (user selectable). Notice: <u>Do NOT drag the ***RPG-FMCW***-folder to the Windows Desktop. The host software will not work properly from the Desktop folder</u>.
- Browse to the ***RPG-FMCW***-folder on the hard drive and localize the executable ***FMCW_H.EXE***. Create a shortcut of this program on the Windows® Desktop

- Right-click on the shortcut (Desktop) and select 'Properties->Compatibility'. Check the checkbox 'Run as Administrator' and exit with 'Apply'.

The directory tree contains a few files the user should notice.
Example: If '***MY_DIRECTORY***' is the directory ***D:\Programs*** the complete tree should look as:

```
D:
 |---Programs
        |---RPG-FMCW-H          root directory, contains the executable FMCW_H.EXE
            |---CONFIG          configuration file FMCW_H.CFG, chirp table CHIRP.TBL
            |---DATA            directory for data file archiving (can be changed)
            |---LICENSE         license file LicID.DAT.
            |---LOG             TCP-IP log file Get_IP.LOG, file access error log file
            |                   FileAccessErrors.LOG, pointing map log files
            |---MDF_MBF         directory for measurement definition files (MDF)
            |---netCDF32        netCDF DLLs for converting binary files to netCDF
            |---RADAR PC        Radar PC executable FMCW_R.EXE, SW updates
            |---RETRIEVAL       retrieval directory
               |---ATN          attenuation retrievals (ASCII versions)
               |---BINARY       retrieval binary versions
               |---LWP          LWP (liquid water path) retrievals (ASCII versions)
               |---SPC          spectral retrievals (ASCII versions)
               |---TMR          mean radiation temperature (needed for sky tipping)
            |---SCAN            solar scan files (beam pattern measurements)
            |---SW-UPDATES      directory for automatic software updates
            |---TEMP            temporary files, used during data file archiving
```

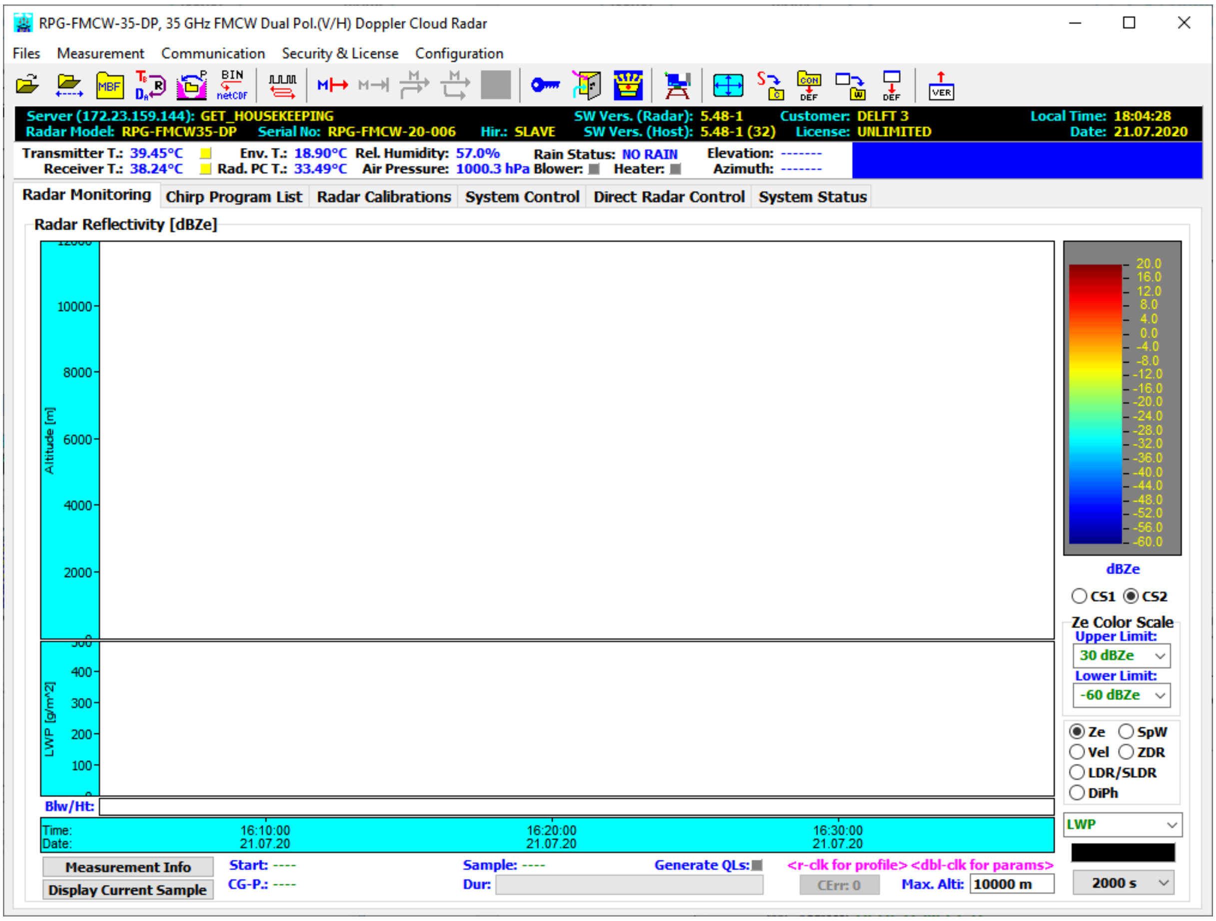
The Host PC (H-PC) is considered to be the 'external' PC. It communicates with the Radar PC (R-PC) inside the instrument. The R-PC collects the radar data and evaluates it (FFTs, Doppler spectra, calculation of moments, reflectivity corrections, etc.). The data link between H-PC and R-PC is via Ethernet. The radar can be plugged into a network (fixed IP) and is then accessed by the H-PC from any network location.
The R-PC has a similar directory structure as the H-PC:

```
C:
 |---RPG-FMCW-R             root directory, contains the executable FMCW_R.EXE,
     |                      previous executable version FMCW_R.OLD
     |                      executable launcher Spawn_FMCW_R.EXE
     |                      DLLs for data acquisition and digital boards
     |---CALIB              calibration files ABSCAL.CLB, ABSCAL.HIS, zero cal.
     |                      files X_ZERO.CAL, anti-alias filter correction
     |                      RecLpfCorr.DAT
     |---CHIRP_GEN          chirp generator files for re-programming chirp table
     |---CONFIG             configuration file FMCW_R.CFG, chirp table CHIRP.TBL
     |                      abs. calib. chirp definition file ABSCAL.CHP
     |                      pointing chirp definition file FASTSCAN.CHP
     |                      positioner baudrate and drive parameters POSI.BDR,
     |                      POSI.PAR
     |---DATA               directory for data file archiving
     |---LOG                TCP-IP log file Get_IP.LOG, FIFO reset log file
     |                      FIFO_RES.LOG, pointing map log files
     |---MDF                directory for measurement definition files (MDF)
```

|---***RETRIEVAL*** retrieval files (neural network)
  |---***ATN*** atmospheric attenuation retrieval file (ASCII)
  |---***BINARY*** binary retrieval file versions
  |---***LWP*** integrated liquid water path (LWP) retrieval
  |---***SPC*** spectrum retrieval file
  |---***TMR*** Tmr retrieval for sky tipping calibrations
|---***TEMP*** temporary files, used during data file archiving

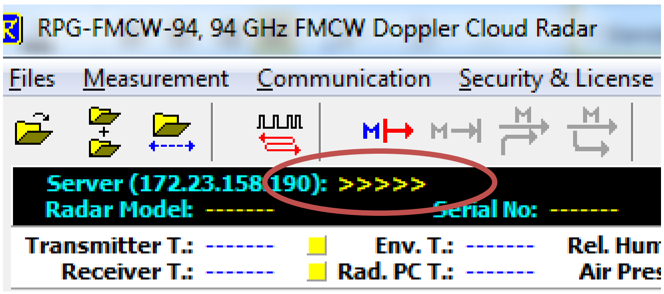
2.2 Getting Started

After the host software is installed according to section 2.1, click  on the desktop to execute it:

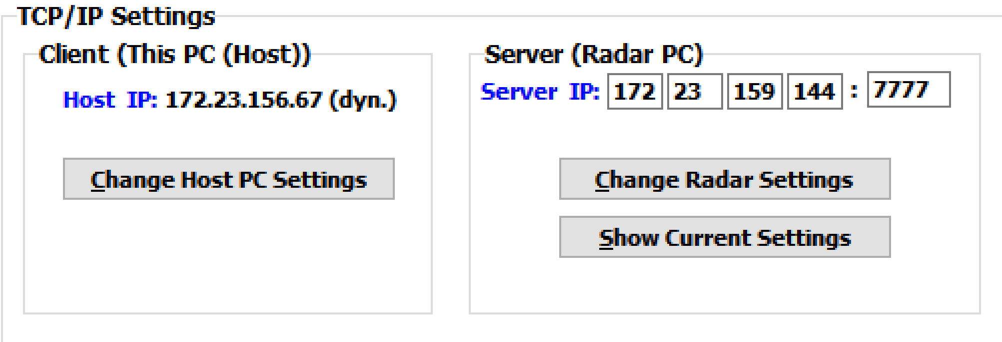


The screen is showing the ***Radar Monitoring*** register tag. On top of the register tags, environmental parameters (surface sensor data), position and blower status are displayed. The black panel summarizes the radar ID information, as model number, polarisation, customer code, software version and license status.

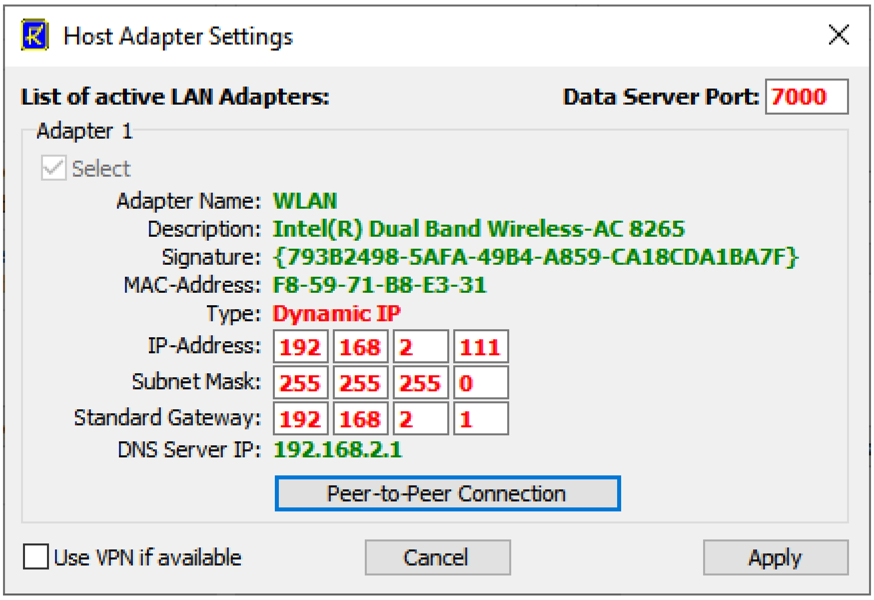
As soon as the H-PC application starts, it is looking for an Ethernet connection to a radar, assuming the H-PC is connected to a network, router or switch. When a connection cannot be established, the TCP-IP command entry in the black top panel is filled with search indicators:

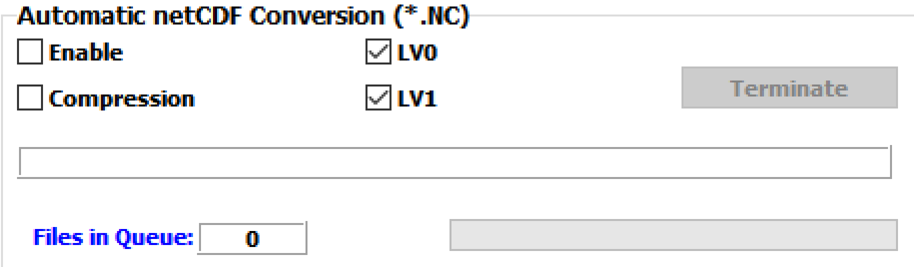


The Host assumes a radar (Server) with a certain IP connected to the network or directly connected (peer-to-peer connection). This IP is defined in the ***TCP / IP Settings*** box under the ***System Control*** tag:



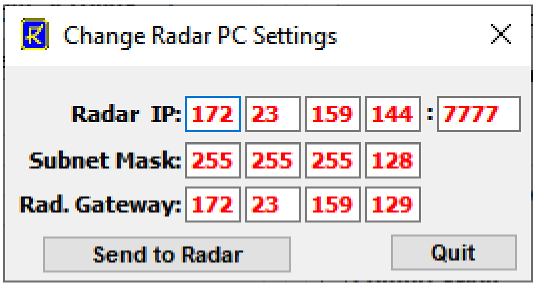
When a new radar is shipped, its IP setting is 192.168.0.1:7777 (default), subnet mask 255.255.255.0. This IP must be entered to the fields right of the label ***Server IP:***. Because the radar's subnet mask is 255.255.255.0, the host IP should be in the same subnet, e. g. 192.168.0.x (x can be any number except for 0 and 1).
The H-PC IP settings may be changed from within the radar application when clicking ***Change Host PC Settings*** (you must run the host software with Administrator rights for this command):





If the IP address you enter here is available within the network the host is connected to, the ***Apply*** command will automatically change the host IP accordingly. An alternative, of course, is the standard procedure using the Windows IP setting menu.
The data server port is used in applications where the Host PC software is remotely controlled via TCP-IP (see section about Data Server).

Once connected to the radar, its IP setting is modified remotely with ***Change radar settings***.



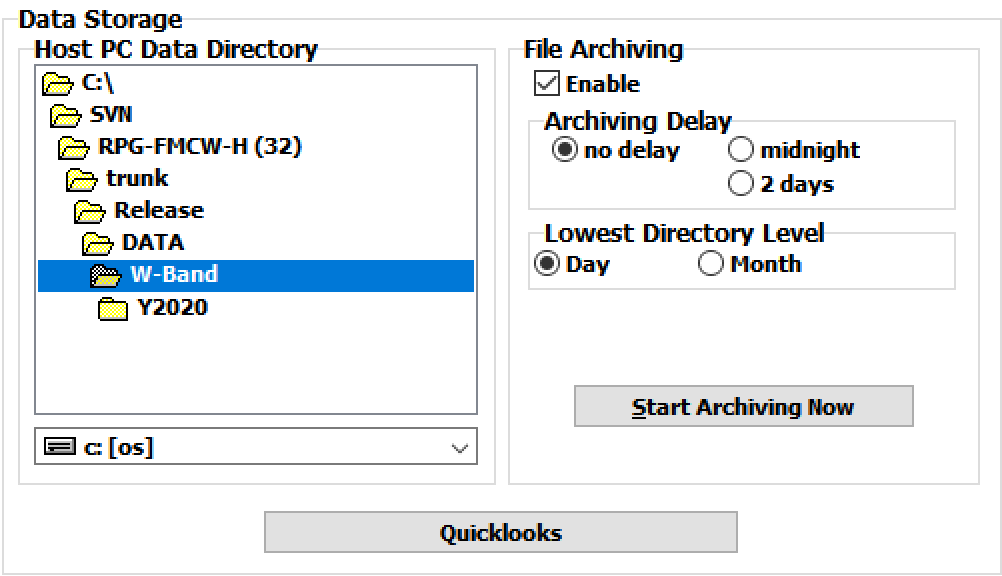
The rest of the ***System Control*** panel is sub-divided into the following groups:

- Data Storage
- El / Az Positioner
- Blower / Heater
- Radar PC Control
- InterLAN Control
- Automatic netCDF Conversion

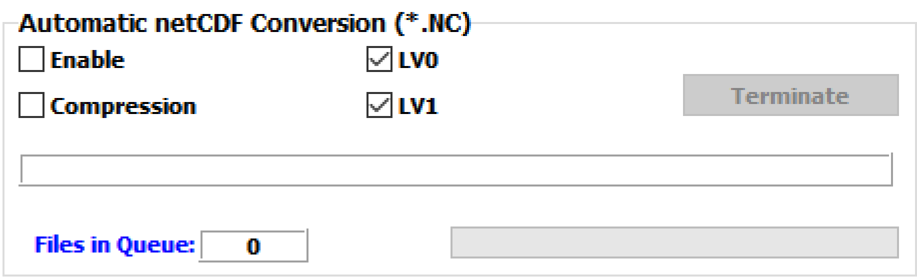
2.3 Data Storage

During measurements, the recorded radar data is automatically stored to the directory selected in the ***Data Directory*** box (in binary format).
All formats are stored to the same data directory.
Data archiving is a useful feature to prevent the data directory from being populated with ten thousands of files, which may overload the operating system. MS operating systems cannot handle many (in the order of ten thousands) files in a single directory. If ***Enable*** is checked, the program automatically creates sub-directories in the data directory and stores the data files according to the year, month and day they were created. For example, a file ***16111623_P01.LV1*** would be stored in a directory ***…\RPG-FMCW-H\Data\Y2016\M11\D16***\ if ***daily*** is checked or in ***…\RPG-FMCW-H\Data\Y2016\M11*** if monthly is checked. Archiving, if enabled, is executed without delay, after midnight or after 2 days, depending on the radio button selection. If the user wants to immediately archive data files, he may click the ***Start Archiving Now*** button.

Data files are created every hour, if a measurement is run in UNLIMITED mode.



Next to the standard binary data format (described in Appendix A), the well know netCDF format is also available:

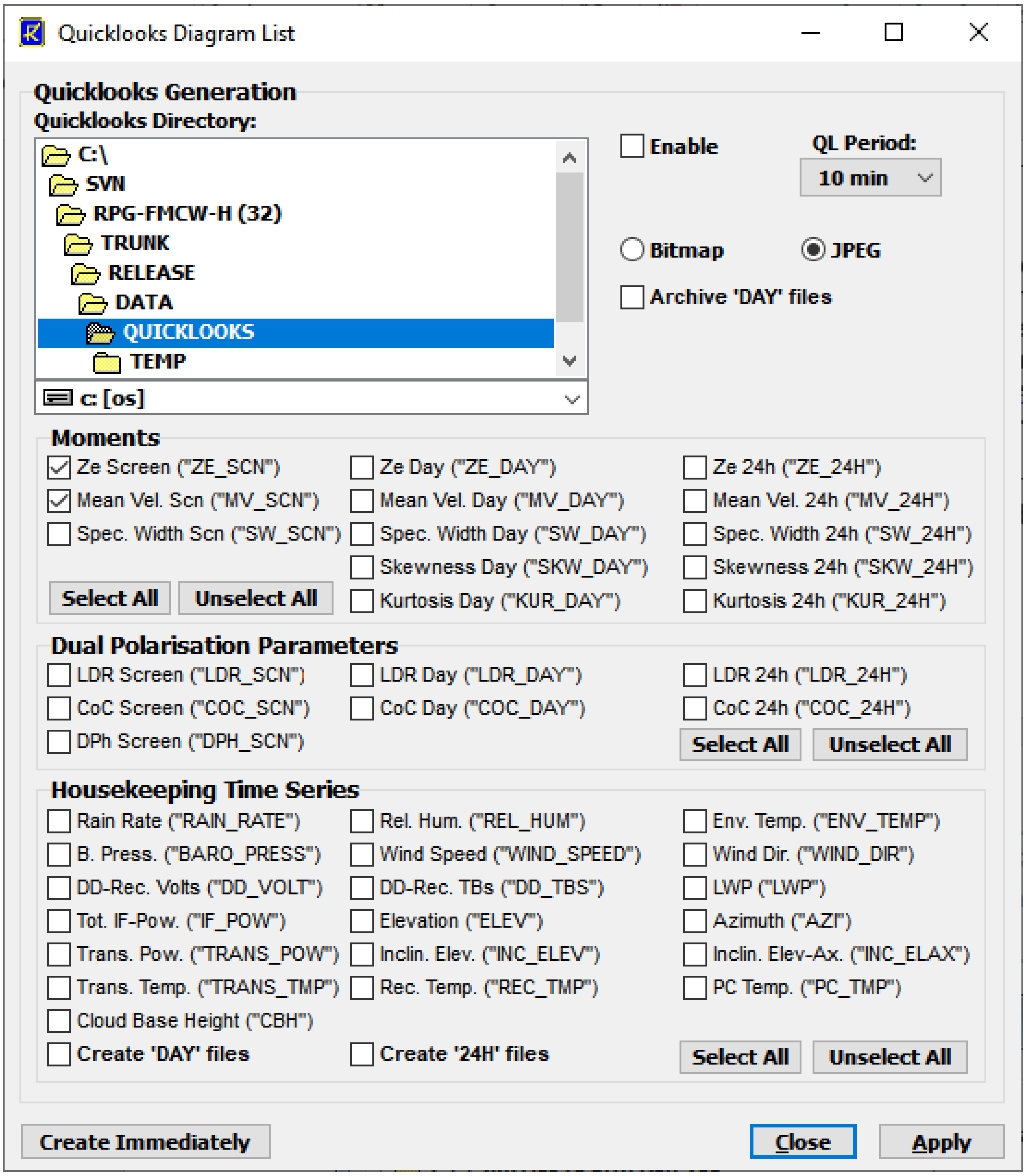


2.3.1 Quick-looks Generation

In order to create quick-looks of radar reflectivity, mean velocity, etc., the user may open the related menu with ***Quicklooks***.
In the upper left corner the quick-looks file path is specified. Quick-looks generation is enabled / disabled by checking / unchecking ***Enable***. The updating period is set in the combo-box labelled **QL Period:** and ***Bitmap*** and ***JPEG*** file formats can be selected.
Within the **Moments** box the moments Ze, Mean Vel., Spec. Width, Skewness and Kurtosis can be selected as “Screen” (SCN), “Day” (DAY) and “24h” (24H) versions:

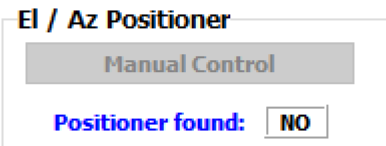
- The SCN-versions display the original current monitoring screens for Ze and mean Doppler velocity.
- The DAY-versions display all data collected during the current day.
- The 24H-versions display all collected data during the last 24 hours.

For DAY- and 24H-versions the data is taken from the archive. Therefore, in order to process these versions, data archiving must be enabled and “File Concatenation” must be disabled.
Additionally, all housekeeping data time series are available as quicklooks.
The names given in brackets are the associated filenames.

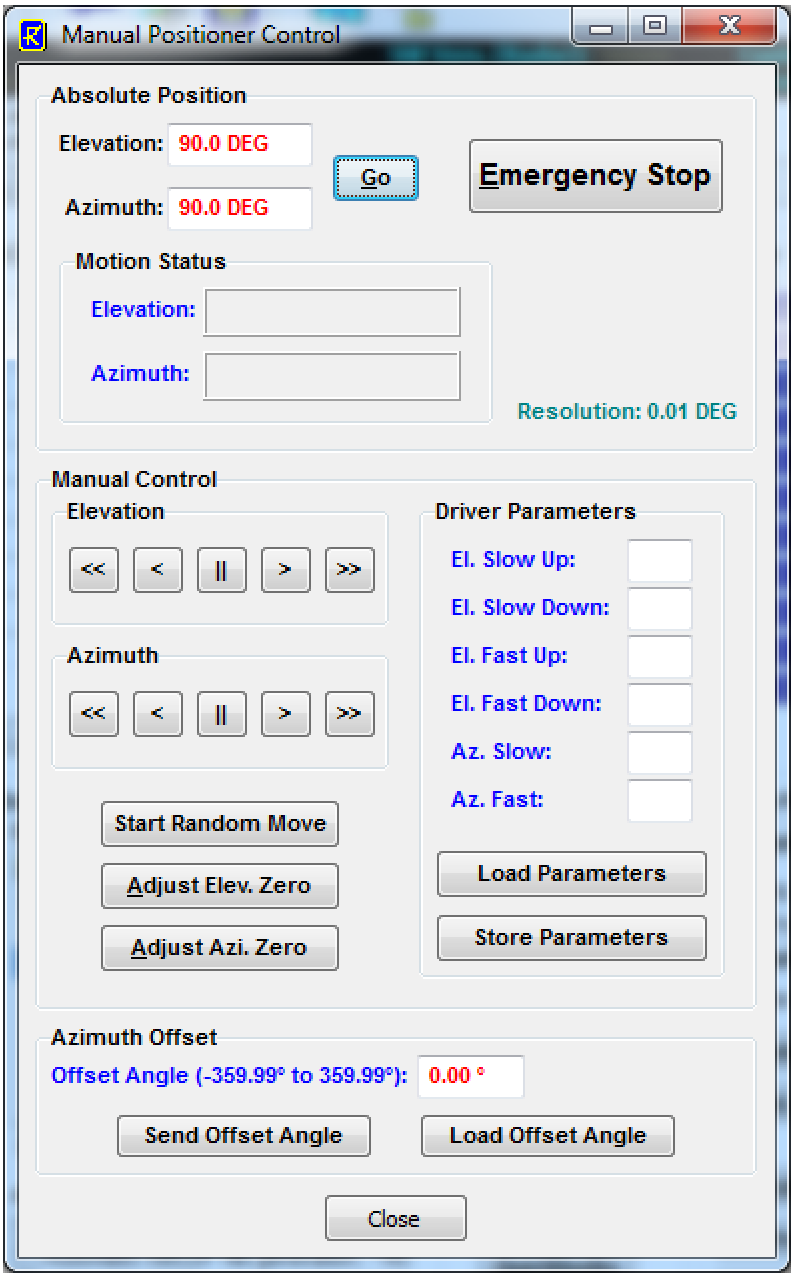


2.4 El / Az Positioner

When the radar is mounted to a scanner, the user may manually control its movements:



The angular stepper resolution is 0.02°.

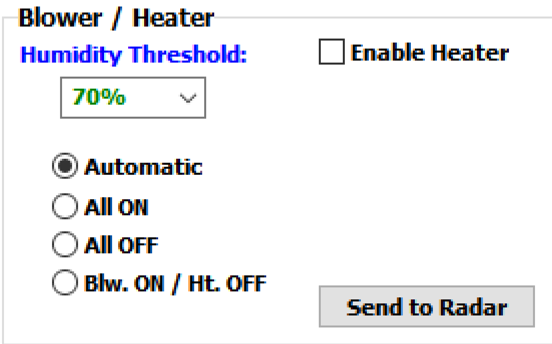


Elevation movements are defined as follows:

- Zenith: +90°
- Horizontal: 0° and 180°

The azimuth value range is set from 0° to 360° (clockwise from top view).
With the **<<**, **<**, **||**, **>**, and **>>** keys the positioner can be moved without specifying a target angle. Slow and fast motions are selected and stopped by clicking the **||** button. In emergency cases any movement can be immediately terminated by clicking the ***Emergency Stop*** button.
The user should not modify the driver parameters in order to maintain a smooth positioner motion.

2.5 Blower / Heater Control

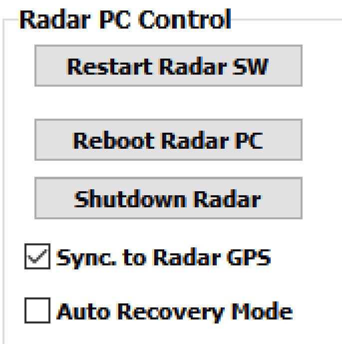


The radome blowers and air heater modules are controlled in the following ways:

- Automatic: The blowers / heater are turned on during rain events or when the surface relative humidity exceeds the value listed in the humidity threshold combo box. A threshold of 70% is recommended.
- All ON: Enforces the blowers / heater to turn on immediately
- All OFF: Enforces the blowers / heater to turn off immediately
- Blw. ON / Ht. OFF: Only turns the blowers on and leaves the heaters off.

The heater modules may also be disabled completely when unchecking ***Enable Heater*** (this function is currently not supported because heater modules are a possible future extension to the rain mitigation system).
The settings only become active if the parameters are ***Send to Radar***, when connected.

2.6 Radar PC Control



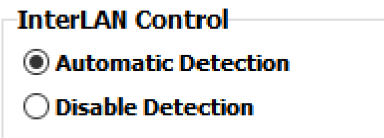
The radar PC can be remotely rebooted or shut down. The shutdown feature should be used whenever the radar shall be turned off (power off). Only the controlled shutdown procedure will ensure that all open files are closed and all interfaces are initialized.
Restarting only the radar software without resetting the radar PC is useful after software updates, when a new radar software version has been transferred and shall be started.
The H-PC system clock may be synchronized to the radar's GPS clock. This feature replaces a time server. The H-PC is synchronized every hour when a radar is connected and ***Sync. to Radar PC*** is checked.
When checking ***Auto Recovery Mode***, the user enables the radar to automatically restart a running measurement after it has been interrupted by a power failure. The host PC software

permanently scans the Ethernet connection for the radar and will automatically reconnect to it to monitor and archive the next measurement samples.

2.7 InterLAN Control

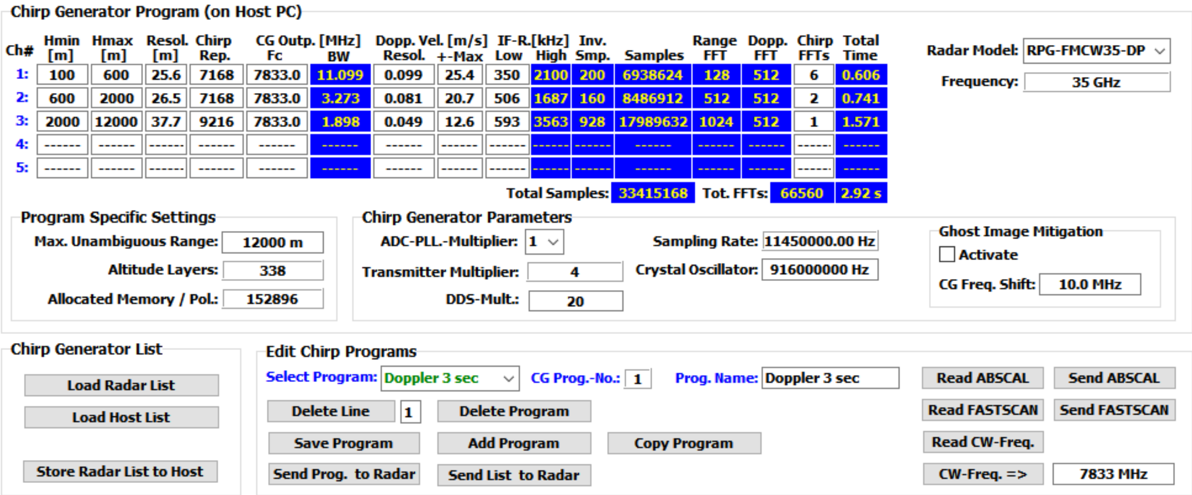
RPG radars can be combined with other radars of different frequency to form a dual frequency system (typically combined on a single scanner) like a K-Band / W-Band pair:

This configuration is realized in a Master / Slave setup, where the Master radar controls all measurements and triggers the Slave radar for sampling. This way, sampling of the two radars is synchronized to better than 50 ms. The radar PCs provide a second Ethernet adapter (the first one is used to communicate with the Host PC(s)). The two radars are linked by a TCP-IP cable in a peer-to-peer configuration, called InterLAN. The radar operating on the higher frequency band automatically becomes the Master (the W-band radar in the example above) instrument while the lower frequency radar (K-band in the example above) becomes the Slave.
Before linking two radars with each other, the user must inform the radar PCs to scan the InterLAN connector for a second radar. This is enabled by selecting the Automatic Detection option within the **InterLAN Control** box:



2.8 The Chirp Program List

As explained in chapter 1, the FMCW radar transmits a continuous sequence of saw-tooth frequency modulation chirps. The chirp parameters, like duration and slope, determine the ranging resolution and range mapping to the IF band. The user can define multiple chirp sequences for different altitude ranges. This is advantageous, because it allows for the optimization of range resolution, sensitivity, maximum Doppler velocity and Doppler resolution for different altitude layers. For instance, a higher range resolution of a few meters is often desirable within the planetary boundary layer (PBL), while a coarser resolution is acceptable (e.g. 30 m) above the PBL which increases sensitivity at longer distances from the radar (larger scattering volumes). The integration time for each altitude range can be individually adjusted by controlling the chirp repetition in each sequence.



A set of chirp sequences (a sequence is a repetition of identical chirps, max. 5 sequences per program) is defined as a program. Up to 10 programs are possible, for example to realize different sampling rates.

Within a chirp program, each line defines a chirp sequence and the white fields can be edited while the blue fields are parameters resulting from the edited settings. Each sequence has a minimum and maximum range and adjacent sequences must meet without overlaps or gaps between them (meaning the maximum range of a sequence must be the minimum range of the next sequence). The minimum range is 50 m, the maximum 20.000 m.

The **Fc** parameter represents the chirp generator's center frequency in MHz. It is multiplied by a factor of 12 (W-Band radar) or 4 (K-Band radar) for the radar's transmitter output. The chirp generator frequency bandwidth is listed in the BW-column. Other columns are the **Resol.**-column (range resolution), the **Dopp. Vel. Resol**.-column (Doppler velocity resolution), the **+-Max**-column (maximum unambiguous Doppler velocity), the **Chirp Rep.**-column (chirp repetition, used to control the sequence duration) and the **IF-R Low**-column (the low IF limit in kHz, must be higher than 300 kHz).

According to equations (1.2.4), (1.3.3), (1.4.3) and (1.4.4) the different parameters depend on each other and are automatically adjusted when changes are made to editable fields.

With the ***Load Radar List*** command, the current chirp table is loaded from the radar (if connected to it). It is not possible to edit this table directly. When the host application is started and a radar PC is found to connect, the radar's chirp table is automatically loaded, which is indicated by the table's box caption ***Chirp Generator Program (on Radar)***. The user may overwrite the table with the table stored locally on the H-PC (***Load Host List***) in the ***…\CONFIG*** directory (***CHIRP.TBL***). Only the local table can be edited.

The ***Edit Chirp Programs*** box summarizes commands for building and editing chirp tables. The uppermost combo box is listing the different table programs. Each program has a name and number (1-10). If a new program shall be added, a new name should be entered ('Prog.

Name'). Then a click on ***Add Prog*** will open a blank program table. With ***Copy Prog*** an existing program can be copied to a new program number and edited afterwards. After all changes to a program have been finished, ***Save Program*** stores these changes to the ***CHIRP.TBL*** file.

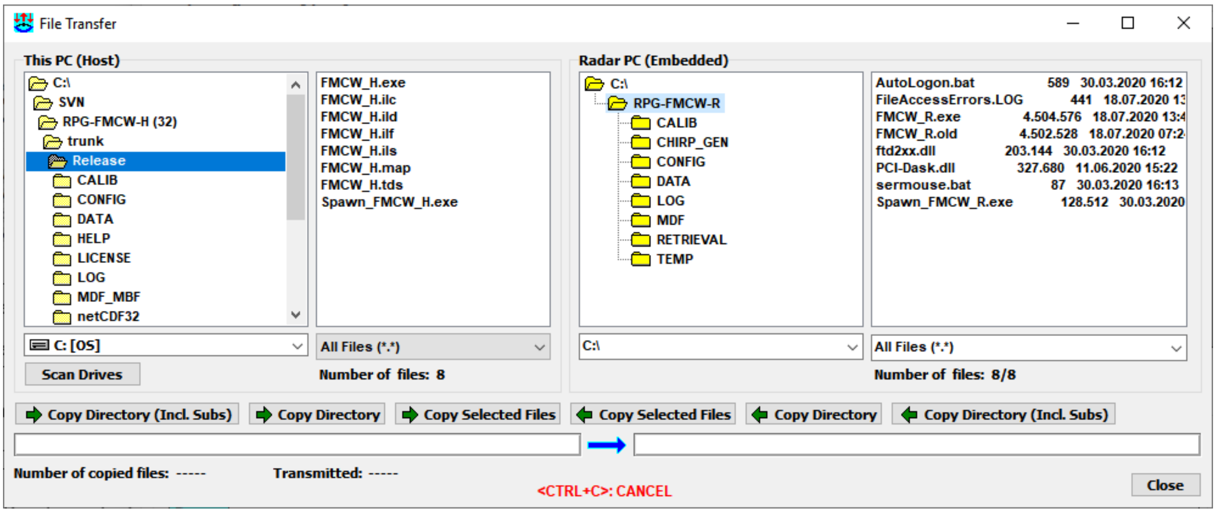
A modified table is sent to the radar with ***Send List to Radar***, overwriting the existing chirp programs on the chirp generator. The overwriting process may take a minute. After reprogramming, the radar sends a log file of the successfully modified chirp sequences. This command can only be executed by the Administrator.

If you want to send a single program to the radar, use the ***Send Prog. to Radar*** button.

2.9 Exchanging Data Files

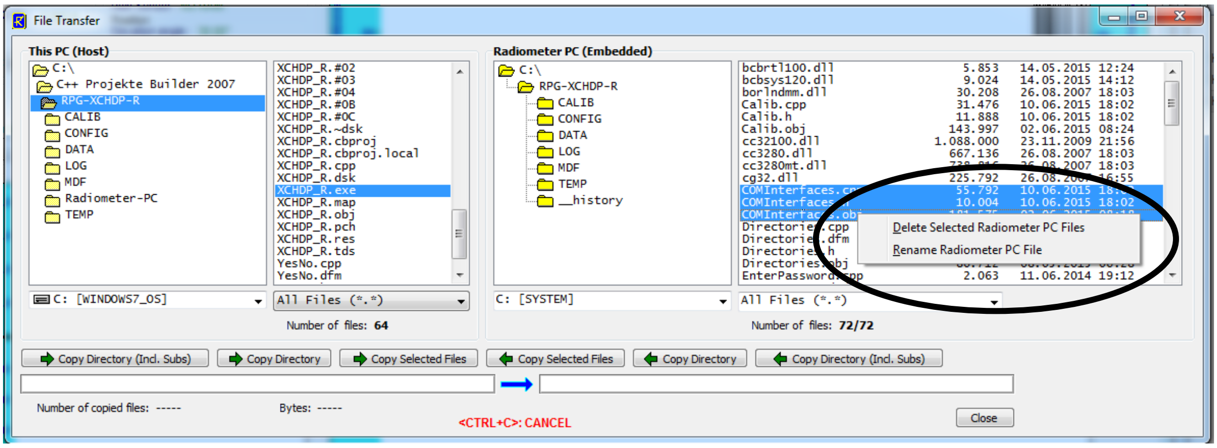
The radar PC is running a Windows®7 or Windows®10 operating system. The radar software excecutable ***FMCW_R.EXE*** is stored in the root directory ***C:\RPG-FMCW-R***. Write processes to this directory or its sub-directories are password protected. When upgrading the radiometer software, the new executable ***FMCW_R.EXE*** has to transferred to this root directory (administrator rights are required). **Overwriting *FMCW_R.EXE* with a corrupted software or deleting this executable file will disable all radar functions and requires to restore the executable directly on the radar PC without host access (using the direct access to the radar PC)!**

To get access to the radiometer directories click (***File Transfer*)**.

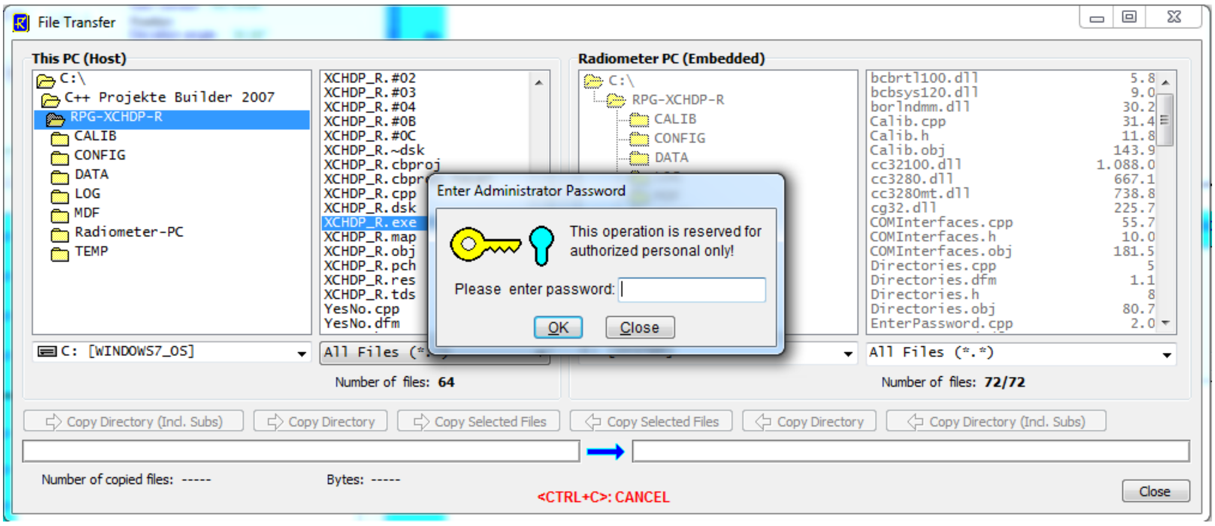


File transfer menu.

With the **File Transfer** menu, the user copies backup data files from the radar's hard disk to the host computer. If file backup is enabled for a measurement, the instrument stores all data files in its data directory ***C:\RPG-FMCW-R\DATA***. This data can then be down-loaded after or during a measurement.
When copying files from one PC to the other, select them within the source directory, while marking a directory on the destination PC and click ***Copy Selected Files***. Alternatively, a complete directory, with or without its sub-directories may be copied (***Copy Directory***, ***Copy Directory (Incl. Subs)***).
Delete files or directories by marking them and clicking the right mouse button:



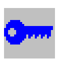
From the displayed drop-down list, the user may select '***Delete Selected Radar PC Files***' or '***Rename Radar PC File***'. These functions are also available for directories.
If the user tries to send files to the ***C:\RPG-FMCW-R*** directory (the root directory), a password must be entered:



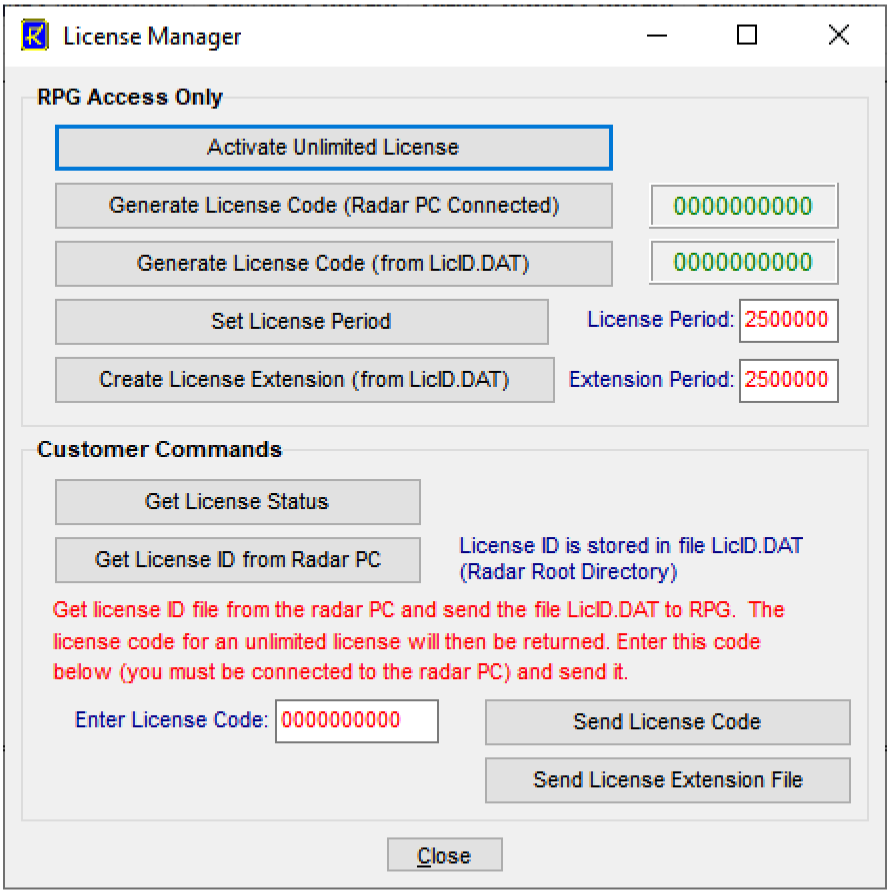
The H-PC is asking for the entry of the Administrator password (APW) to complete the desired action. This mechanism prevents unauthorized users from overwriting important R-PC system files.

2.10 The License Manager

RPG's radars are delivered with a preliminary limited license of 30 days. Without activating an unlimited license, the radar terminates measurement execution when the limited license is expired. The common procedure to install a permanent license is the following:

Invoke the **License Manager** by clicking the  button.
The user retrieves license status information with the ***Get License Status*** command. The license type (limited or unlimited) as well as the expiration date and time will be displayed. In order to obtain an unlimited license, the following steps have to be performed:

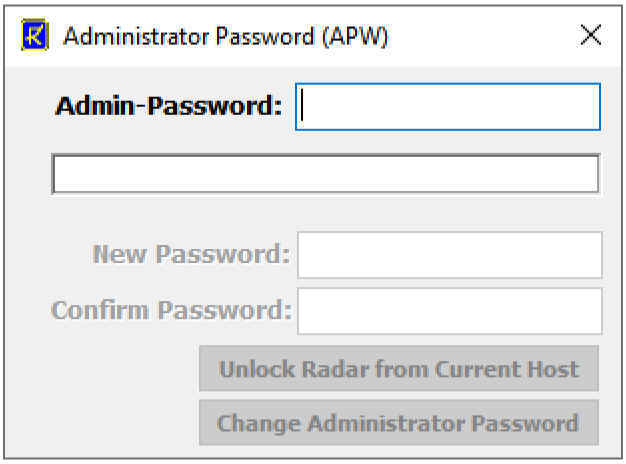
1. Connect to the radar and click the ***Get License ID from Radar PC*** button. The license ID code is then written to the file ***LicID.DAT*** stored in the H-PC ***…\LICENSE*** directory (see section 2.1).
2. Send the 'LicID.DAT' file to RPG (by e-mail to info@radiometer-physics.de). Then the 10 digit license code will be returned (also by e-mail).
3. Enter the 10 digit license code to the edit box in the license manager and click ***Send License Code***. The license manager will inform the user, if the unlimited license installation was successful or not. If not successful, please contact RPG again.
4. It is possible to extend a limited license. The procedure is as described above. Send the ***LicID.DAT*** file to RPG by e-mail and receive a license extension file ***LicCode.EXT***. Store this file to the H-PC ***…\LICENSE*** directory (see section 2.1) and click the ***Send License Extension File*** button in the license manager menu.

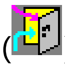


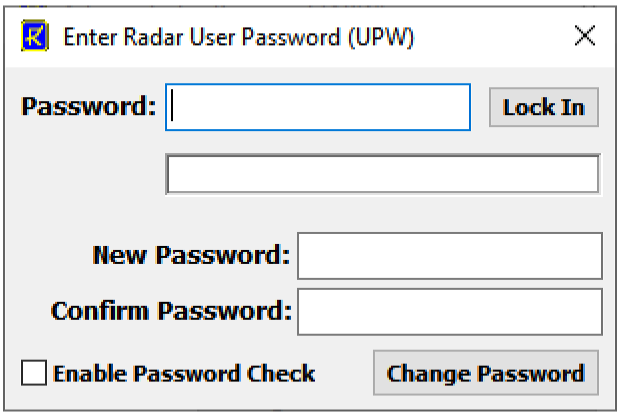
License manager menu.

2.11 Administrator Password (APW)

System critical operations, for instance software updates or chirp table overwrites, are password protected. A new radar is shipped with the default Administrator password (APW) ***Administrator***. Note that the APW in this context is NOT the Administrator Password of the Windows® operating system. The APW referred to here is ONLY relevant for the radar application. The APW can be modified by clicking the  button:



Before a new APW is defined, the old one must be entered first. The procedure is identical to the change of Windows® account passwords. The administrator password is stored on the radar PC (if connected) with ***Change Administrator Password***.
The host application will only ask for the APW when the host is connected to a radar. In stand-alone mode (without a connection) no password checking is applied.
When a connection to the R-PC is established, the radar will reserve this connection to the first client connecting to it. This first client is the real host. All other clients connecting after the host (secondary clients) will be informed that the R-PC is occupied. The locking to the host client remains valid for 5 minutes when the TCP-IP traffic between R-PC and H-PC has stopped. After this period, the R-PC is free for other clients to connect. If the R-PC is locked to a host client and a secondary client tries to connect to the radar, the host locking can be unlocked on the secondary client by the Administrator with ***Unlock Radar from Current Host***.
The radar is usually connected to a network. Therefore, theoretically everyone who knows the radar's IP address within this network can connect to it. In order to establish a protection mechanism, the Administrator may define a Radar User Password (UPW) ():



When the APW is entered as password, all disabled fields become enabled. The Administrator can then define a UPW, enable or disable the UPW checking and change the UPW, assuming the H-PC is connected to the R-PC.
The same menu is used for radar user lockin. If UPW checking is enabled on the radar, the UPW-window pops up for UPW entry the first time the user connects to the radar. When a valid UPW is entered (***Lock In***), the radar accepts the user for connection.

2.12 Software Updates

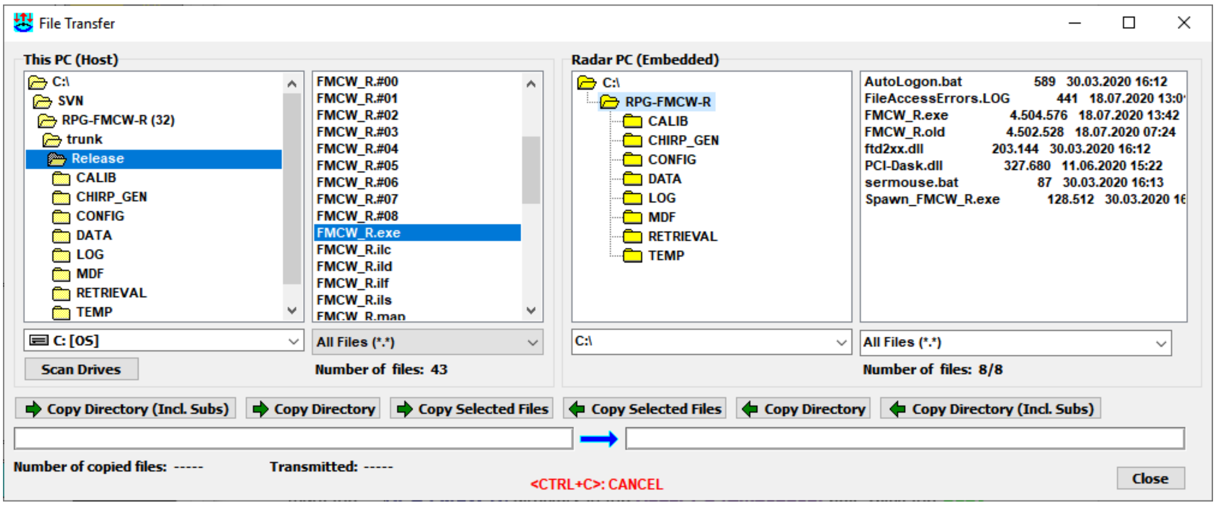
It may be desirable to update the H-PC and R-PC software version from time to time, in order to add advanced features to the data processing or to correct software bugs.
The radar SW is running on an embedded PC and is named ***FMCW_R.EXE***. This file is located in the radar's root directory ***C:\RPG-FMCW-R*** (see section 2.1).
The host SW name is ***FMCW_H.EXE*** and it is located in the application's root directory (…***RPG-FMCW***) on the H-PC.

For a SW update, the following steps are required:

1. Step: Save the old software versions
a) Create a directory to save the old software versions (e.g. ***C:\MyPath\Save***).

b) Connect the H-PC to the radar and enter the File Transfer Menu ( ). On the left side (H-PC, 'This PC (Host)') browse to the directory for saving the files (e.g. C:\MyPath\ SAVE) and on the right side (Radar) in the …***RPG-FMCW-R*** directory mark the ***FMCW_R.EXE*** file. Then click ***←Copy Selected Files***.
c) Locate the ***FMCW_H.EXE*** file in the …***RPG-FMCW*** directory on the H-PC and copy this file to the ***C:\MyPath\Save*** directory (by using the Operating System File Explorer).

2. Step: Overwrite the old versions by the new ones
a) Copy the new version of ***FMCW _R.EXE*** (the R-PC software) to an arbitrary directory on your host PC (e.g. …***RPG-FMCW\RADAR PC***). In the file transfer menu, browse to that directory. Mark the ***FMCW _R.EXE*** file in the file list within the ***This PC (Host)*** box and mark the …***RPG-FMCW-R*** directory in the ***Radar PC (Embedded)*** box. Click the ***Copy Selected Files→*** button. Because you are now going to overwrite a file in the radar's root directory, you must enter the Administrator password to proceed.

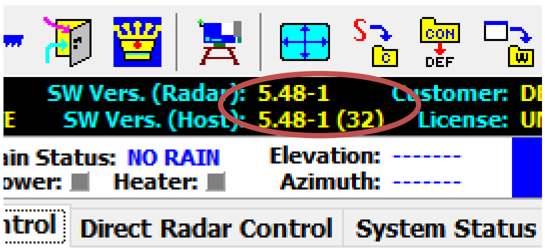


File Transfer Menu during software update procedure.

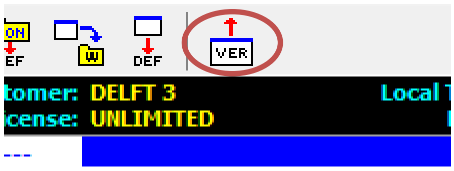
b) Reload the R-PC software (see below) to run the new ***FMCW_R.EXE*** version. Wait for approximately 10 seconds until the R-PC has restarted ***FMCW_R.EXE***. The H-PC will automatically re-connect to the radar.
c) Terminate ***FMCW_H.EXE*** on the host and overwrite it by the new version.
d) Execute ***FMCW_H.EXE*** and reconnect to the radar.

Reloading R-PC software (***System Control*** ***register tag)***

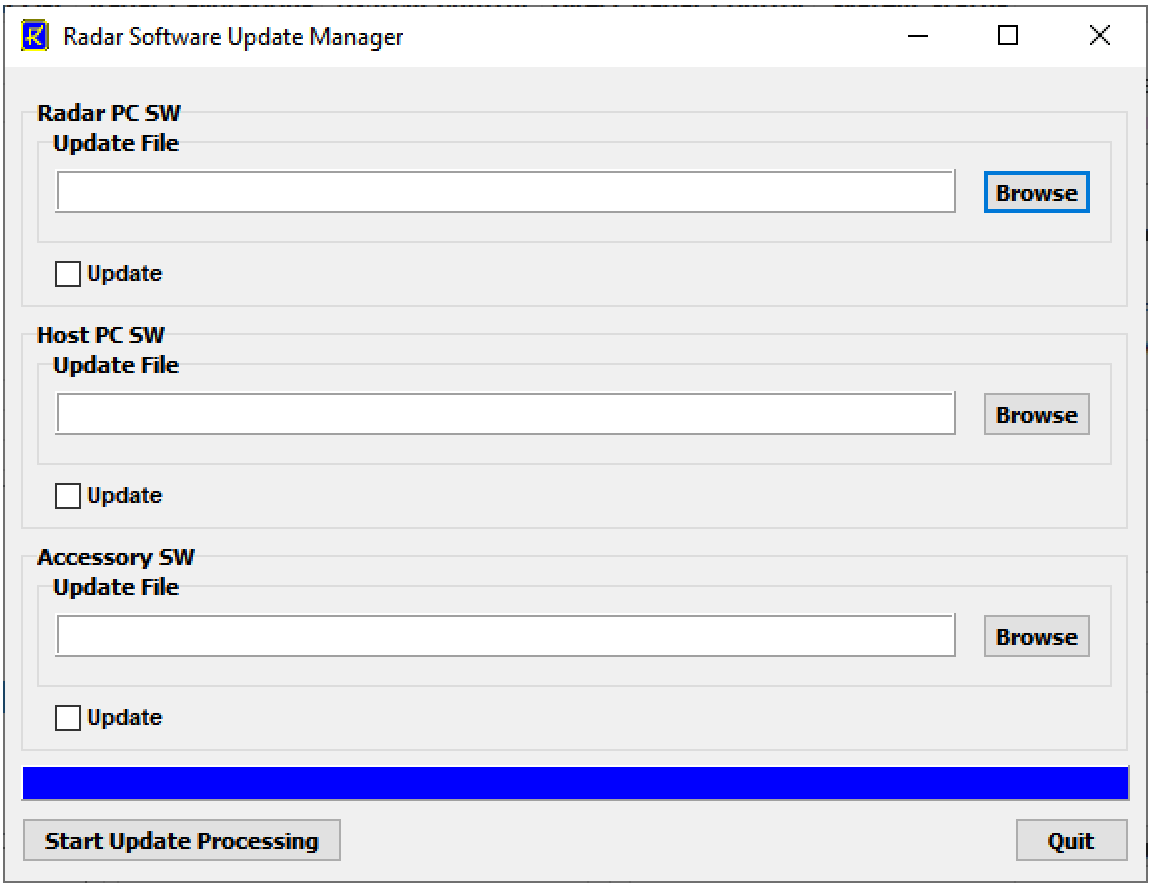
The software upgrade is finished. You can confirm the successful upgrade by reading the software version numbers of the embedded R-PC and the H-PC in the black status line on top of the application screen. Both version numbers should be identical.



Alternatively, one can use the SW-Update Manager:



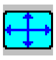
The new SW version files ***FMCW_H.EXE*** and ***FMCW_R.EXE*** should be stored to arbitrary directories on the local Host PC first. Within the SW-Update Manager, you browse to these directories for each file, check the associated checkboxes (**Update**) and start the automatic update process with ***Start Update Processing***.

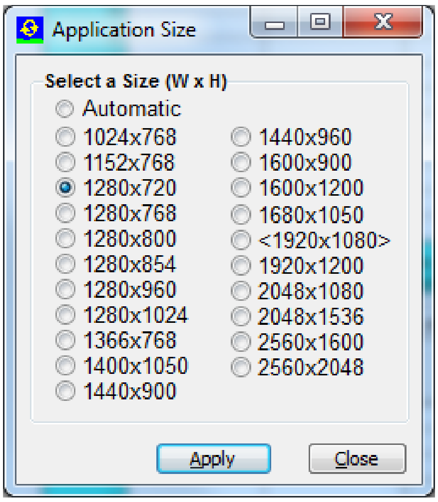


SW update manager menu for automatic SW version updates.

2.13 Application Size on Screen

The host SW can scale the application to almost any common screen size. This may be useful if you want to run a beamer to display the host application in the beamer's screen resolution and size.

Click  to enter the **Application Size** menu:

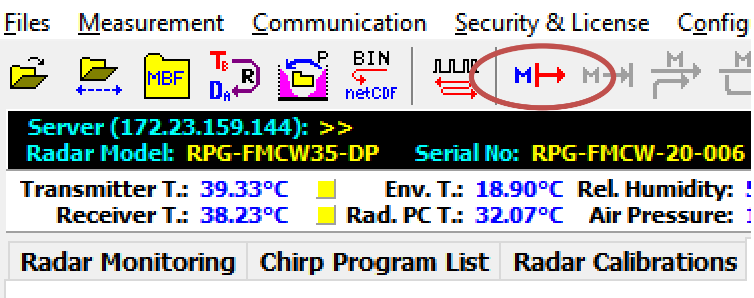


The screen size in brackets <…> is the host's maximum screen size but you can select any (smaller) sizes to fit the application on other screens (like a beamer). The application sizing can be done any time, even during a running measurement.

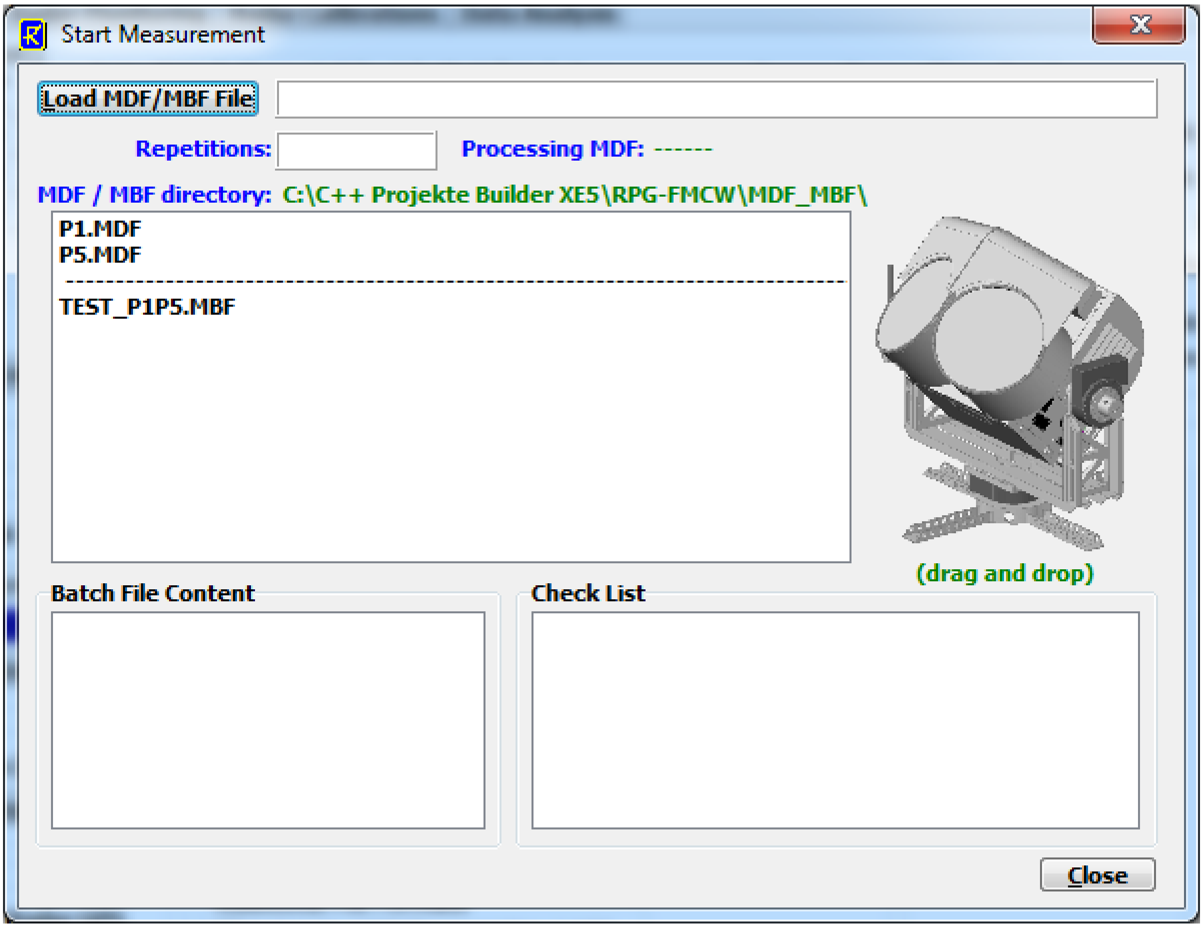
2.14 Starting Measurements

Before a measurement can be started on the radar PC, a measurement definition file (MDF) needs to be created first, containing all details of the measurement setup. This file is then sent to the radar for execution. Please refer to section 2.13.1 to learn how to create MDFs.

When a host successfully connects to the radar and the radar is in STANDBY mode, the radar is ready to start a measurement. This status is indicated by the enabled button in the application's shortcut panel:

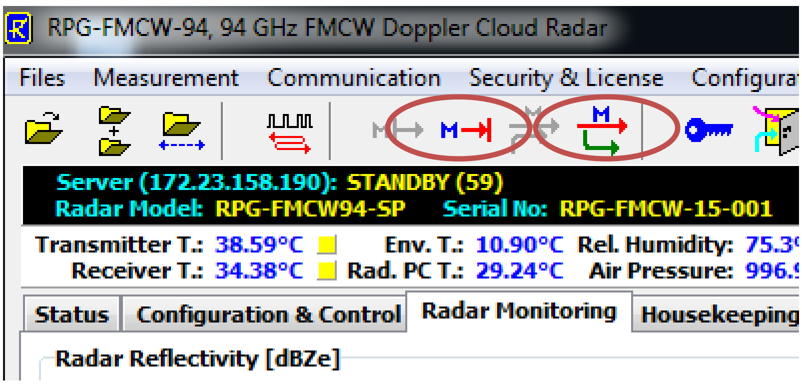


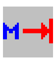
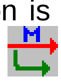
When an MDF or MBF is loaded (***Load MDF/MBF File***), its contents and repetition factor are displayed. In addition some pre-checks are performed, e.g. radar configuration, MDF version number, availability of chirp program number, etc. A variety of other checks ensure that no erroneous command data is sent.
When the consistency check of a MDF is finished, the test result is displayed in the ***Check List***. The batch can only be sent to the radiometer if all consistency checks have finished with the status OK. Then the MBF is transmitted automatically.
The H-PC 'remembers' the directory where MDFs and MBFs are stored from a previous ***Load MDF/MBF File*** command. This directory is listed in green. In the MDF / MBF list, MDFs are separated from MBFs by a dashed line. Dragging a file from the list and dropping it on the radar image on the right (or simply double clicking the file) is starting the measurement, if the consistency checks have been passed successfully. In this case the measurement launcher is closed automatically.

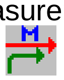
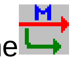
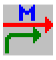


Measurement Launcher

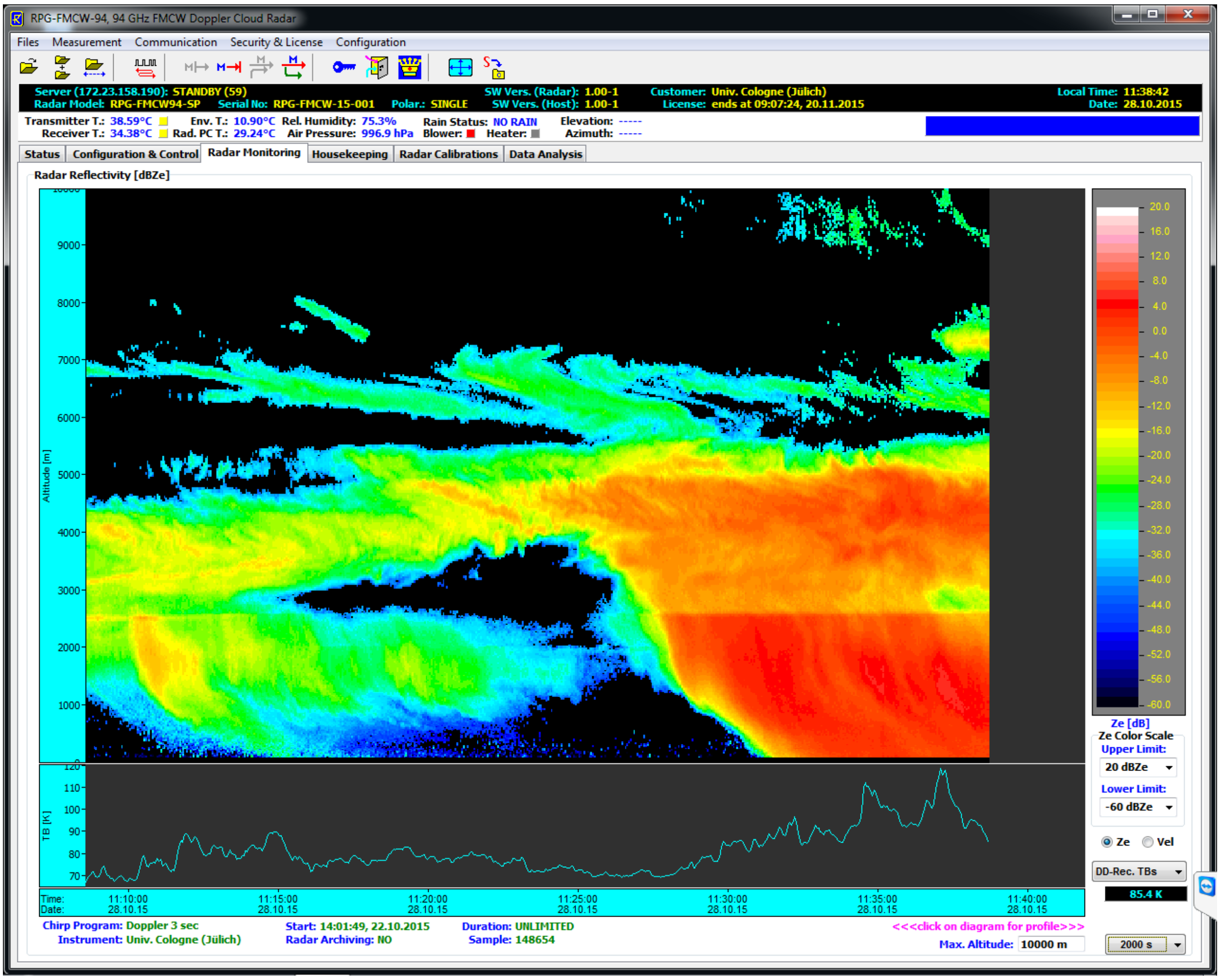
Once a measurement has been started, the control buttons in the shortcut panel change in the following way:



The  button is used to terminate the running measurement on both, the radar and the host, while the  button enables the host to drop off the measurement and leave the radar alone to continue. In both cases all monitored data samples are stored and the associated files are closed.

If the radar is running a measurement and the host connects to it, the H-PC realizes the active status and enables the  button for the host to jump on the measurement and start monitoring it. The  and  buttons do not affect the radar activities during a measurement, but act as host monitoring toggle switches.

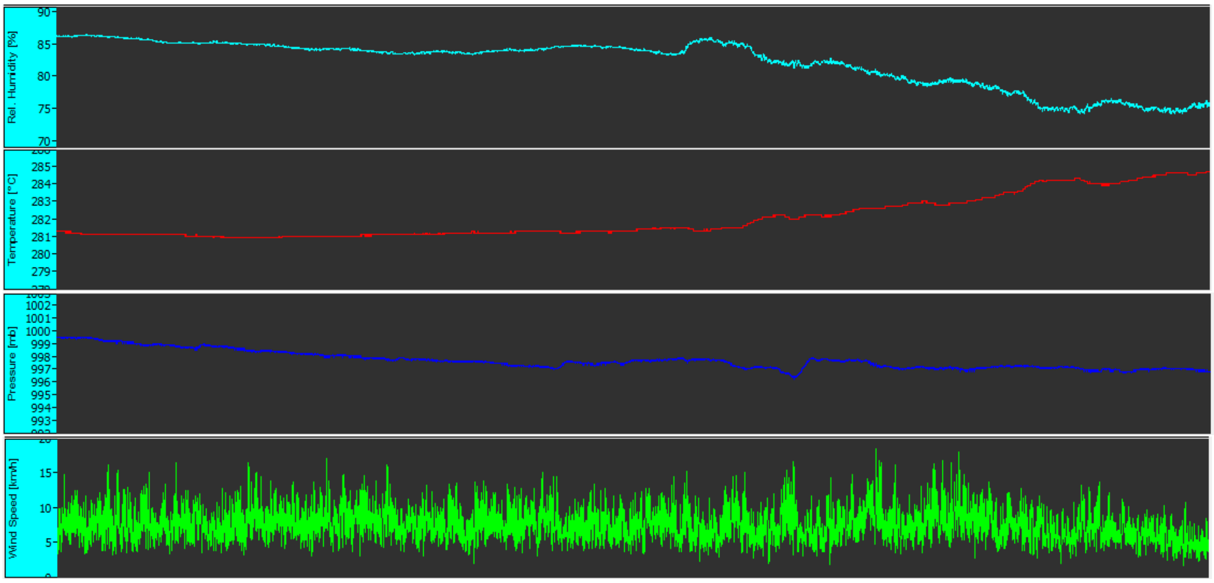
The radar profiles are displayed in the register page ***Radar Monitoring*** which acts as a real time display. In the main graphics area a color coded time series of reflectivity (in dBZe) or mean velocity (in m/s) is shown. Two radio buttons ***Ze*** and ***Vel*** switch between the two alternatives (higher moments can be displayed within the 'Open File' menu). The color coding limits are user adjustable.



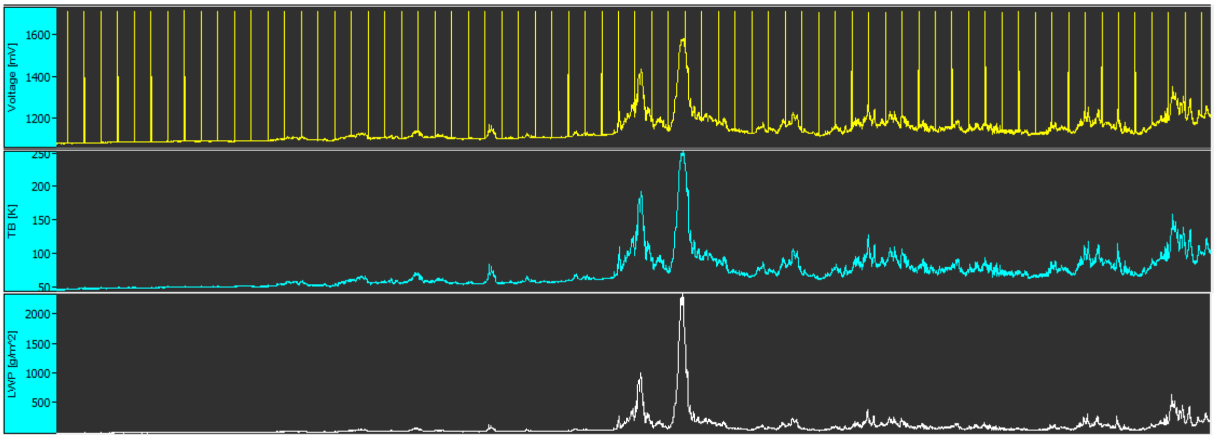
***Radar Monitoring* register page**

Underneath the main display a switchable time series of different useful parameters is plotted. The desired parameter is selected from a combo box on the right side of the time series.

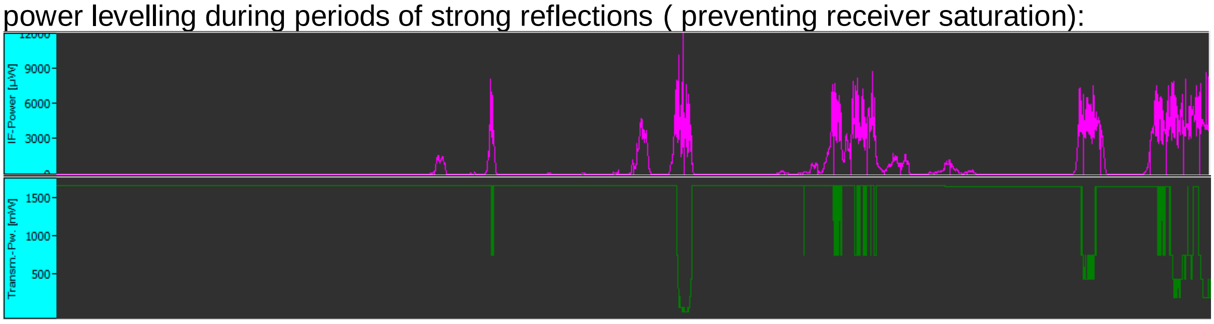
The radar is equipped with a weather station, providing information about environmental temperature, rel. humidity, barometric pressure, wind speed / direction and rain / snow rate:



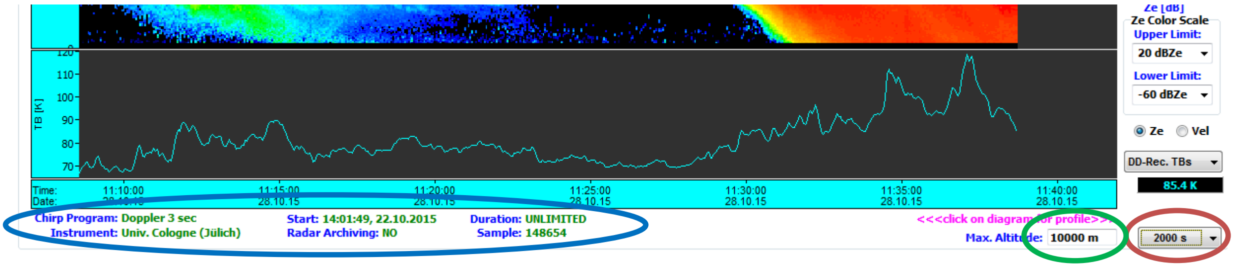
Another time series group is related to the direct detection passive channel at 89 GHz (W-band radar) or 31.4 GHz (K-band radar), which is intended for retrieving LWP. Implemented are the DDR (Direct Detection Receiver) detector voltages, brightness temperatures T_B and LWP:



Additionally, information about the IF power level at the ADC board input (end of IF chain) as well as the transmitter power level are presented. The later one demonstrates the automatic power levelling during periods of strong reflections (preventing receiver saturation):



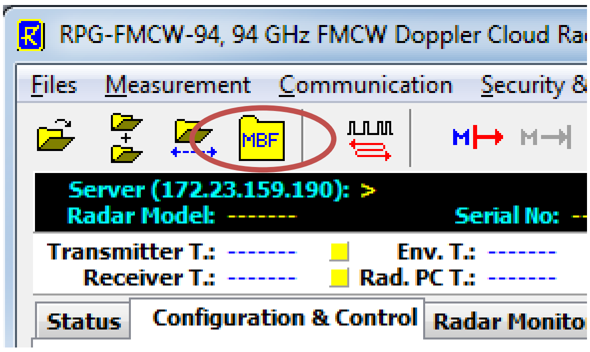
The time series' time span is set in another combo box (red ellipse) at the bottom line of the screen. In addition, the maximum vertically displayed altitude in the main screen area can be modified (green ellipse):



Additional information about the chirp program in use, the measurement start and duration, the customer code and radar PC archiving status is plotted (blue ellipse).

2.14.1 Creating MDFs and Batch Files (MBFs)

In order to create a measurement definition file (MDF), which later is sent to the radar PC, click on the icon below:



This launches the MDF window, composed of a register of four tags that summarize different aspects of a measurement.

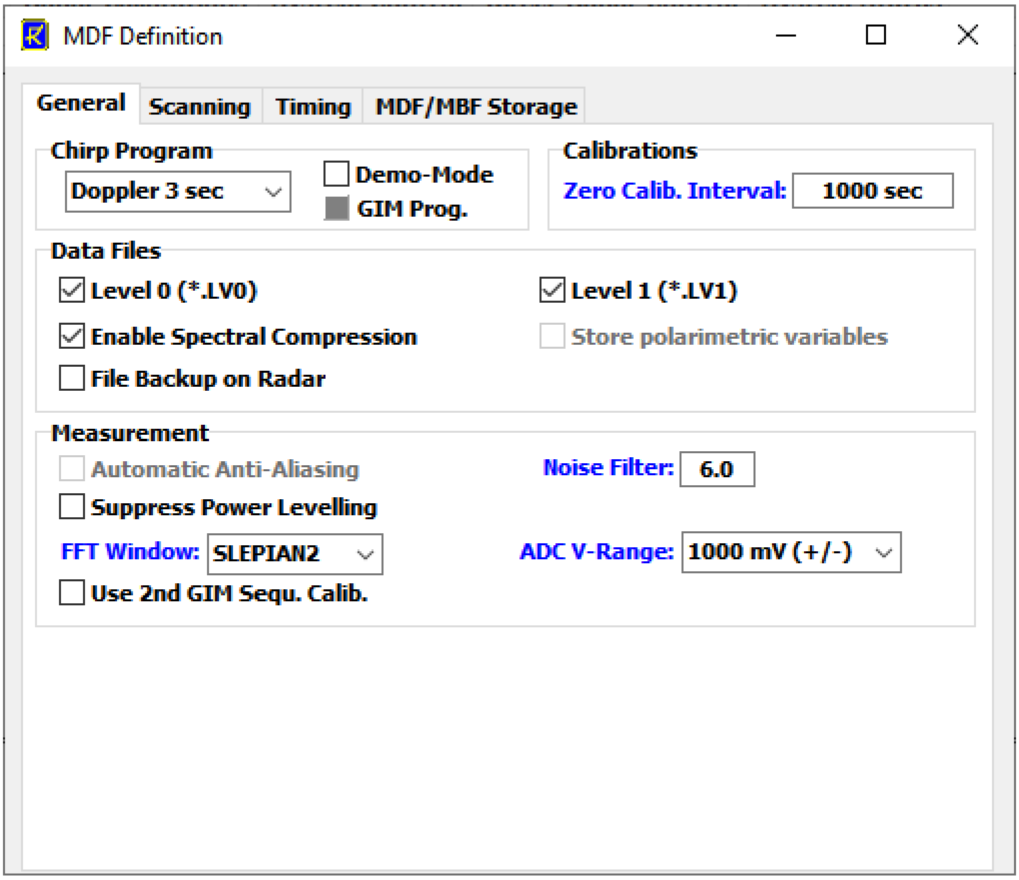
The ***General*** tag opens a setup menu with general measurement parameters:

Chirp Program: Selection of a radar's chirp program for the MDF. A single MDF can only define one of the available chirp programs.

Calibrations: Here the user specifies the period for zero calibrations. These are automatic calibrations performed inside the radar. As explained in the calibration chapter, the radar is using a Dicke Switch reference target for frequent calibrations of the radar receiver's gain, which may drift within a couple of hours. In a zero calibration, the radar's transmitter is switched off while the Dicke Switch is closed to terminate the receiver input with a known radiometric temperature. Then the radar integrates on the Dicke Switch for one sample duration and recalibrates the gains of all radar IF channels as well as of the passive direct detection channel. We recommend setting the zero calibration period to about 1000 seconds. If the passive channel (used for integrated liquid water measurements) is not required in the measurement, the zero calibration period can be set to be much longer, for instance 3600 seconds or more.

Data Files: This box contains parameters related to data file storage. The radar produces data files of different levels (LV0 = raw data, LV1 = pre-processed data, refer to Appendix A for more details). The storage of these data levels is enabled / disabled by associated checkboxes.

While a measurement is running, the radar PC stores data files to a temporary directory. When a Host connects to it, the Host PC is starting the download of all files within the temporary radar PC directory. The radar PC then deletes the files after download, if **File Backup** is not active. Otherwise, the radar PC archives the files on its internal HD.
A usefull feature is Spectral Compression, which reduces the file size by up to an order of magnitude. Appendix A2.1 includes further details about spectral compression.
For dual polarization radars, there are additional spectral variables available, described in detail in section 1.11. These variables are providing valuable cloud property information in a variety of applications or scan patterns. Additional storage of these variables may increase the LV0 file size by a factor of 2 to 3. Alternatively, polarimetric variables and spectra can be produced in a post-processing step from LV0 spectra (V / H pol. and covariance spectra).

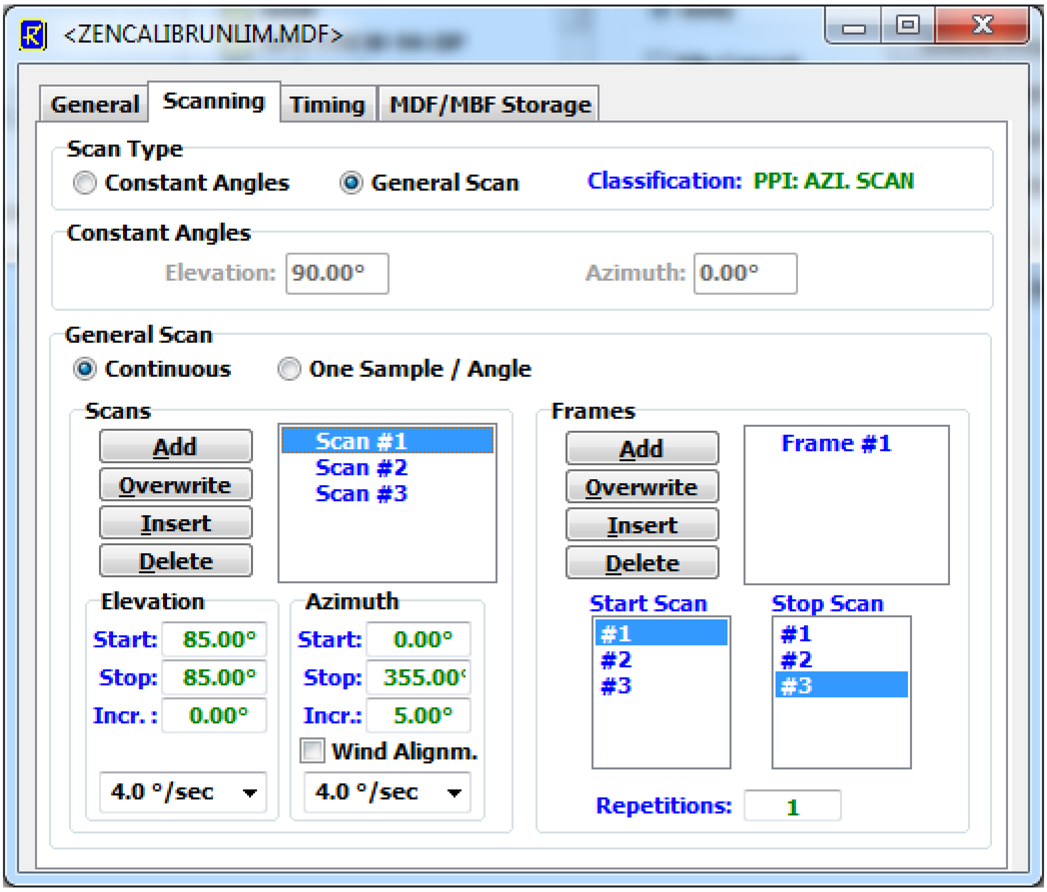


'General' tag of MDF definition window.

Measurement: In chapter 1 we explained the problem of aliasing in Doppler spectra, which is caused by insufficient chirp repetition rates (the 'Doppler dilemma'). The problem exists for pulsed radars as well. The radar software includes an algorithm to anti-alias the Doppler spectra but some users may want to apply their own methods. This is why anti-aliasing is optional.
Noise Filter: In spectral compression mode, the underground noise power is subtracted from Doppler spectra and the noise floor is characterized by a standard deviation (STD). The noise filter factor is the number of STDs the radar is using as a threshold for valid data. Noise filter settings in the range of [3.0,…,7.0] are recommended for effective noise suppression and may depend on the chirp program in use. Note that a higher noise filter threshold reduces the radar's sensitivity. **By setting the noise filter to 0.0, the noise cancelling is disabled and the radar transmits only raw spectra (no noise level removed, all spectra stored, no anti-aliasing).**

As mentioned before, the radar may automatically adjust the transmitter power if strong reflections (strong rain events) lead to a receiver saturation. In order to disable power levelling, check **Suppress Power Levelling**.
The FFT window is also user selectable. We recommend the SLEPIAN2-window, which offers a good compromise between good resolution and FFT leakage suppression. The ADC voltage range works best in the +/- 1.0 V interval for optimum dynamic range and sufficiently low spurious signal level of the ADC board. In cases where spurious signal suppression is not important and a maximum dynamic range without power levelling is desired, the ADC range might be changed to +/- 2.0 V or even +/- 5.0 V.
The ***Scanning*** tag opens a setup menu summarizing all information required in scanning measurements:

Scan Type: Here the user selects between observations of constant elevation / azimuth angles and more complicated 'general' scans. Please note that if azimuth angles >0.0° or elevation angles ≠ 90° are selected, an elevation / azimuth positioner must be installed for moving the radar to the desired positions.



'Scanning' tag of MDF definition window.

When selecting ***General Scan***, the definition of arbitrary elevation and azimuth scan patters are possible. If the optional elevation / azimuth scanner is not available, the radar ignores all angle definitions.
The radar movements consist of elementary scans from a start angle to a stop angle with a certain increment angle. These scans are numbered as Scan#1, Scan#2, ...

The radar does not execute single scans but only frames of scans. Each frame has a start scan and a stop scan (these can be identical) which form a 'loop' of scans that may be

repeated arbitrarily. The concept of having two levels of movement definitions allows for the definition of complex scan procedures.
Define a frame by selecting one of the scans in the start scan list and then clicking on one in the stop scan list. After entering the repetition number, ***Add*** (or ***Insert***) add (or insert) a frame to the frame list. It is possible to edit a frame definition using the ***Overwrite*** command or to delete it with ***Delete***. Three examples illustrate how a frame executes:

1) Start: Scan#4, stop: Scan#6, repetitions: 3 ⇒
   Scan#4,Scan#5,Scan#6,Scan#4,Scan#5,Scan#6,Scan#4,Scan#5,Scan#6
2) Start: Scan#4, stop: Scan#2, repetitions: 2 ⇒
   Scan#4,Scan#3,Scan#2,Scan#4,Scan#3,Scan#2
3) Start: Scan#2, stop: Scan#2, repetitions: 1 ⇒
   Scan#2

The radar may perform sample measurements **continuously** while moving or in **One Sample / Angle** mode. When **continuous** scanning is selected, the increment elevation and azimuth angles of a scan definition are ignored, while in **One Sample / Angle** mode the radar does not measure while moving but only after reaching a discrete position in a scan.
Whit active ***Wind Alignment***, the positioner will ignore all azimuth settings but aligns the azimuth direction in parallel to the current wind direction.
Scanning speeds of elevation and azimuth axis can be set independently between 0.5°/sec and 5.0°/sec.

The ***Timing*** tag opens a setup menu summarizing all information related to measurement start time, trigger modes and termination options.
If the measurement has a well-defined end time (automatic measurement termination, ***LIMITED*** mode) the radar needs a 'Base' filename in order to create an MDF specific filename. In ***UNLIMITED*** mode the radar creates file names automatically:

YYMMDD_HHmmSS_PXX_MCC.LV0 and

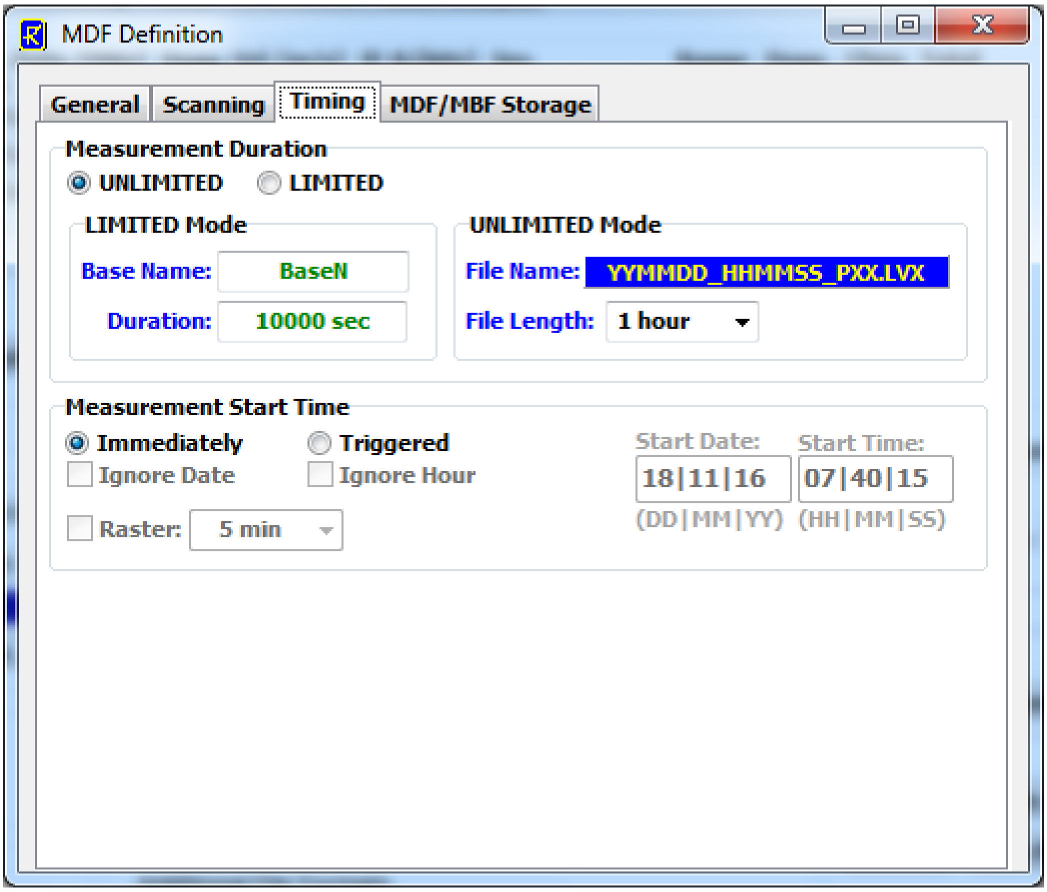
YYMMDD_HHMMSS_PXX_MCC.LV1

YY = Year, MM = Month, DD = Day, HH = Hour, mm = Minute, SS = Second, XX = chirp program number, MCC = measurement classification code (see Appendix A4). In ***LIMITED*** mode, the file names are changed:

BaseN_YYMMDD_HHmmSS_PXX_MCC.LV0 and

BaseN_YYMMDD_HHMMSS_PXX_MCC.LV1

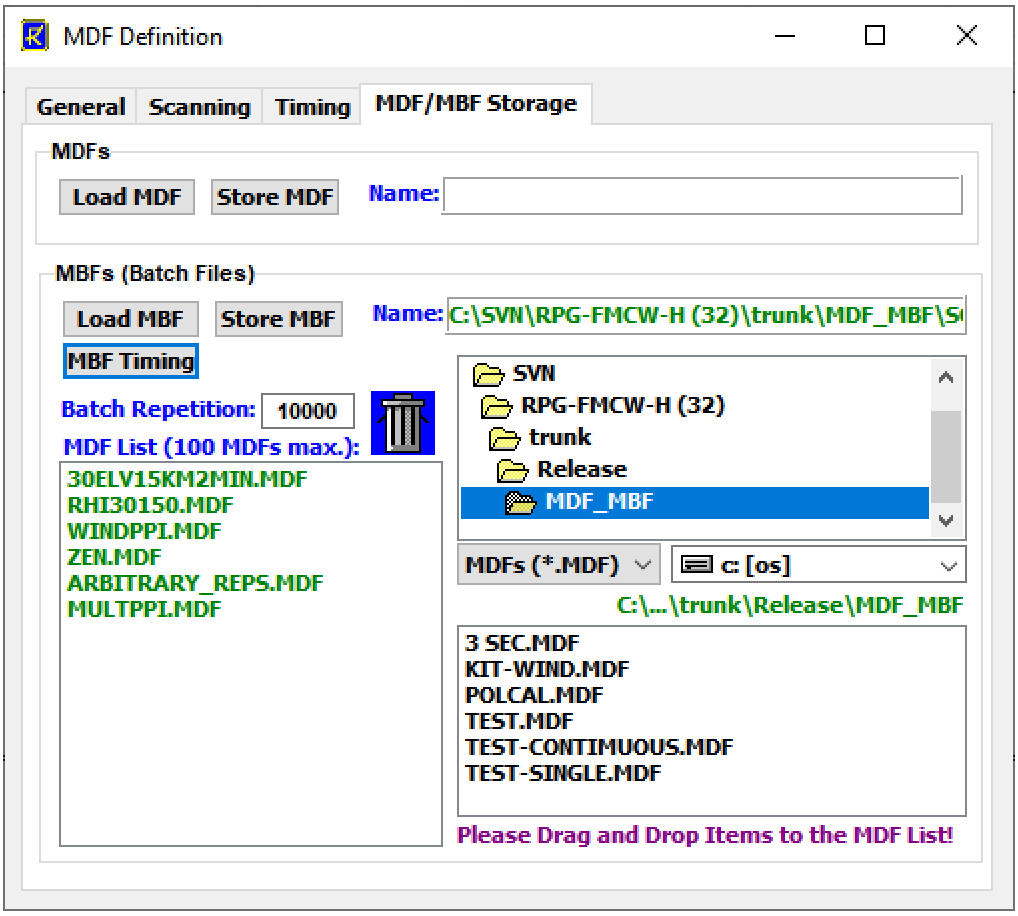
In ***LIMITED*** mode the measurement duration (starting from the trigger point) has to be specified, while in ***UNLIMITED*** mode the file length (in hours) can be selected. It we recommend to use a file length of one hour for unlimited measurements, because data files can reach an enormous size.



'Timing' tag of MDF definition window.

Start time and end time are important parameters for a measurement setup. There are two ways of triggering a measurement: Immediately after launching the MDF or at a certain time and date. Using a start time before the current time is equivalent to an immediate start. If the measurement start is triggered to a certain time, the check boxes ***Ignore Date*** and ***Ignore Hour*** allow for a date or hour independent triggering. This is particularly useful in a repeated multiple MDF batch measurement, where MDFs are repeated multiple times. A triggering to a certain date / time would trigger the MDF only once but not repeatedly. E.g. if ***Ignore Hour*** is checked (assuming ***Triggered*** mode is activated) and the 'Start Time' entry is set to 22|36|15, the measurement is triggered to 00:15:00, 01:15:00, 02:15:00, …, ignoring the current date and hour. If a more frequent trigger is required, one can use the ***Raster*** feature combined with a raster period. This mode assumes a start time of 00|00|00 and uses a raster period. For instance, if the period is 10 minutes and the current time is 10|17|32, the start trigger occurs at 10|20|00, 10|30|00, 10|40|00, etc.

The ***MDF / MBF Storage*** tag opens a setup menu for storing the MDF parameters to a certain filename, loading an existing MDF file or creating batch files (MDFs).
It is possible to send a *single* MDF **directly** to the radar. Multiple MDFs are combined in a MBF (measurement batch file).

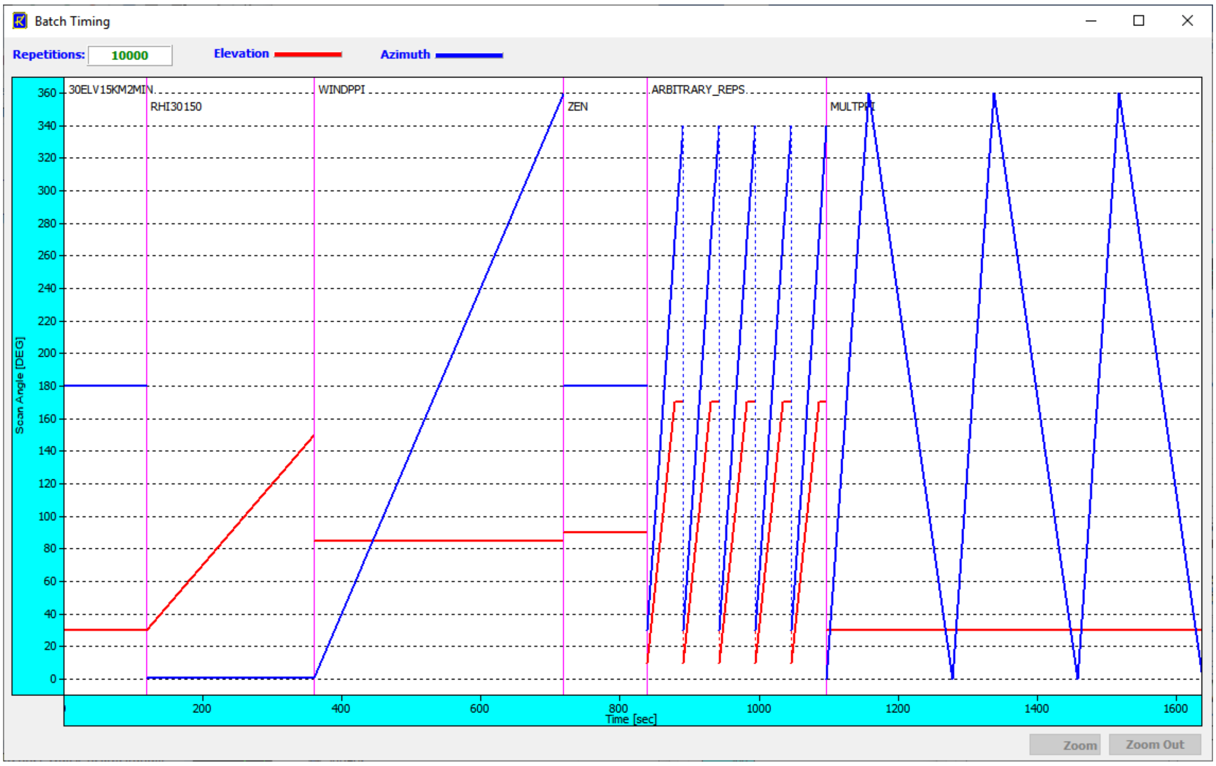


'MDF/MBF Storage' tag of MDF definition window.

MDFs in a batch file execute sequentially in the order listed in the MDF list. The batch repetition number has the same meaning as the frame repetition factor for scanning: The MDF list forms a loop, which is repeated an arbitrary number of times. This offers the user a flexibility of combining different measurement tasks, which would otherwise not be compatible in a single MDF, for instance when combining different scan patterns (RHI, PPI, etc.) with zenith observations and multiple repetition cycles. The solution is to define different MDFs for each individual measurement task (e.g. a PPI scan) and combine them in a batch file with a certain repetition factor. The only restriction for MDF definitions in multi-MDF batches is that all MDFs in a batch list must be ***LIMITED*** mode MDFs.

It is a good practice to store all MDFs in one directory (e.g. ***…\RPG-FMCW\MDF_MBF***). All MDFs in the selected directory are shown in the box in the lower right corner. From this list, the user may select each MDF he wants to add or insert to the MDF batch list by dragging the desired MDFs to the MDF batch list box. MDFs can be deleted from the MDF batch list by dragging them to the waste bin. Store your measurement batch files (MBFs) in a single directory (like ***…\RPG-FMCW\MDF_MBF***).

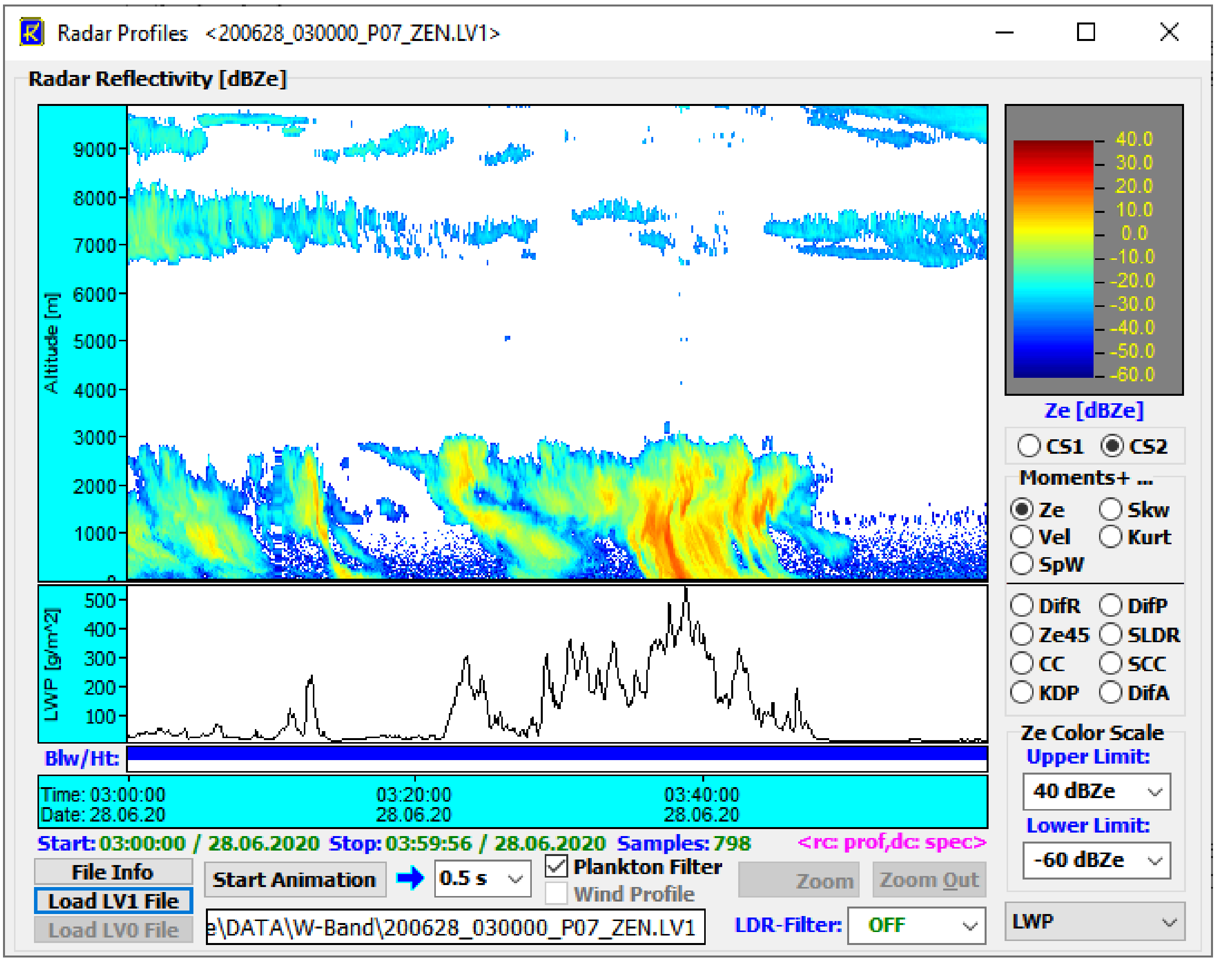
A useful feature is the visualisation of the MBF timing. When multiple scanning MDFs are combined, the time series of elevation and azimuth positions can be inspected by clicking **MBF Timing**. For the example MBF displayed above, the following display is shown:



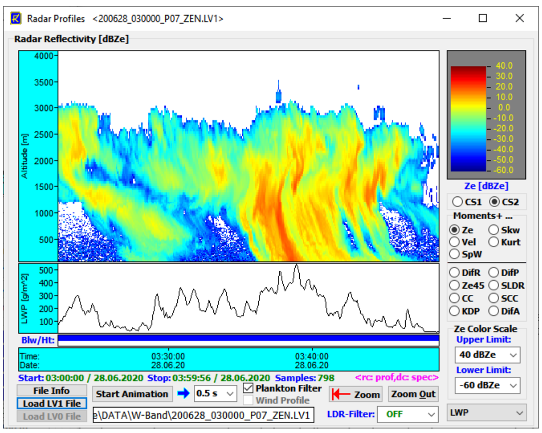
The various MDFs are separated by vertical magenta lines and the associated MDF names are plotted on the diagram top. Azimuth angles are displayed in blue, elevation in red. Dotted vertical lines indicate movements without radar sampling as can be seen in “ARBITRARY_REPS”.

2.15 Open Data Files

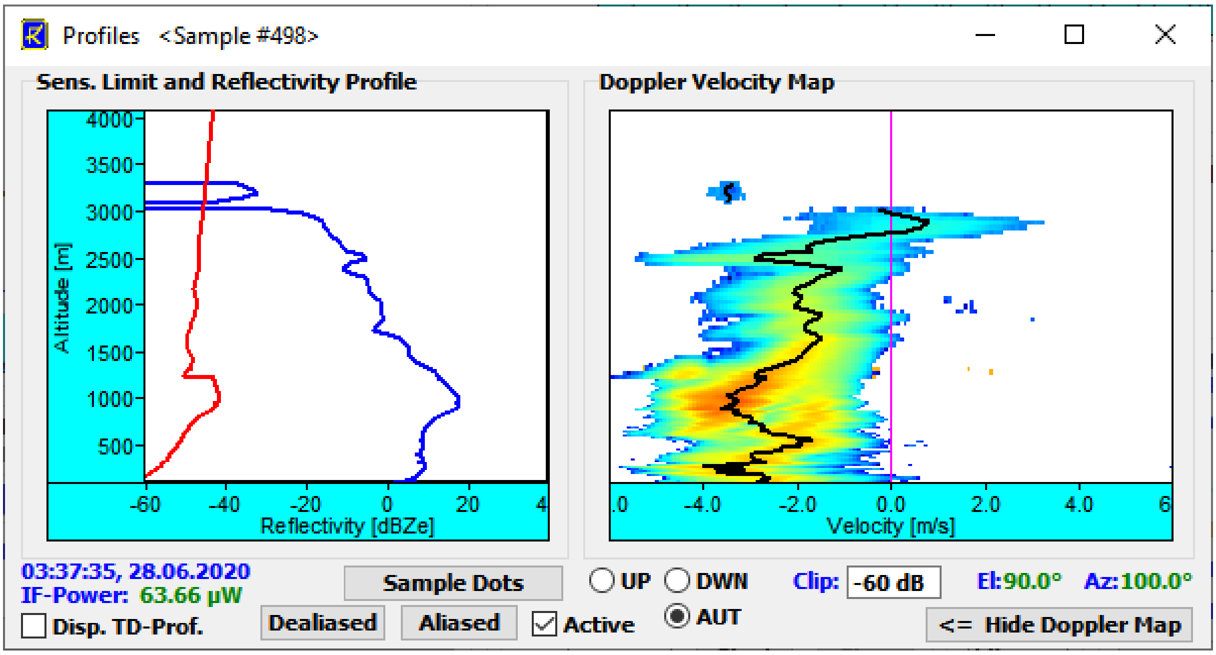
Existing data files are loaded and inspected by clicking the (Open Radar Data File) button. With ***Load Level 1 File***, select a *.LV1 file:



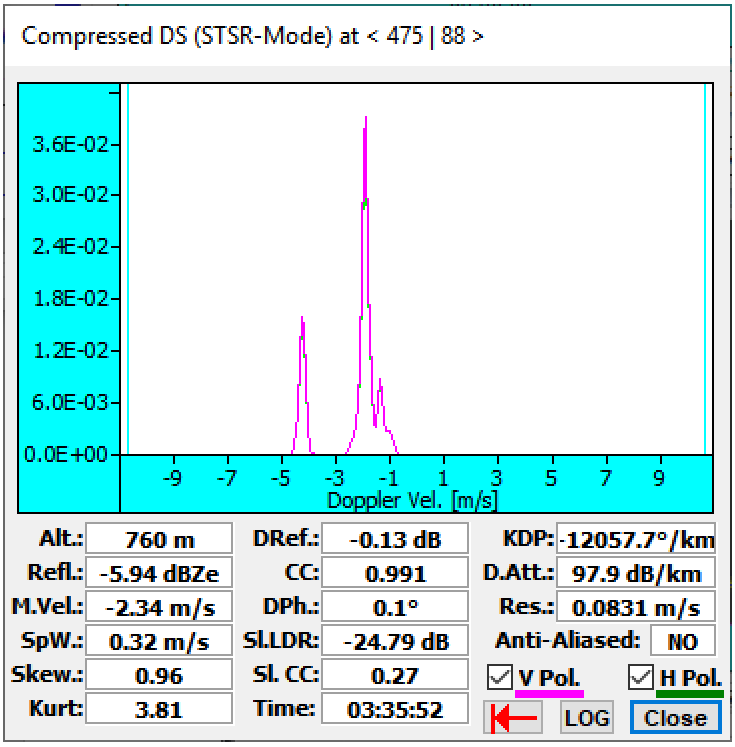
Multiple radio buttons allow for switching the display between all moments like reflectivity ***Ze***, Doppler velocity ***Vel***, spectral width ***SpW***, skewness ***Skw*** and kurtosis ***Kurt***. For dual polarization radars, additional parameters like LDR, differential phase, differential ZDR, correlation coefficient and other parameters are available.
The user may zoom into and out of the displayed data:



Clicking ***Zoom Out*** zooms back one step while |← ***Zoom*** zooms back to the maximum range.
A right click on the main display shows the reflectivity profile together with the sensitivity limit in red (see equation (1.5.7)) and the color coded Doppler map.



A double click on a non-empty range cell plots the Doppler spectrum of that cell:



Vertical Ze-profile and Doppler spectrum

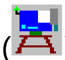
The structure of a level 1 data file is listed in appendix A.

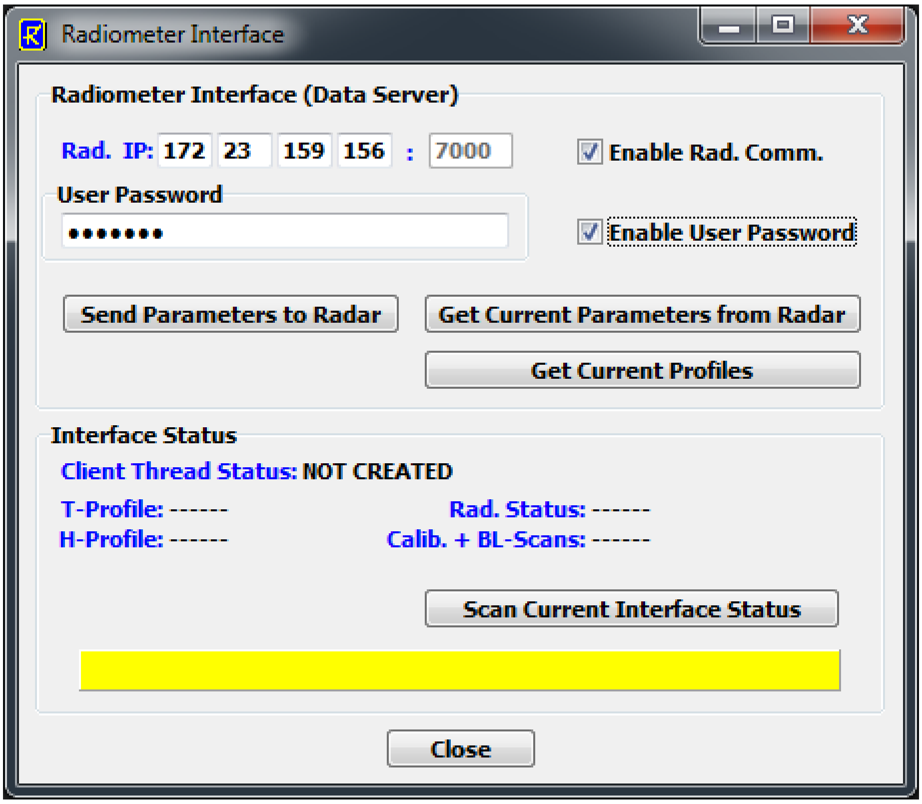
2.15 Connecting the Radar to RPG-Radiometers

RPG radars provide a passive channel close to the radar observation frequency in order to derive useful parameters as LWP (Integrated Liquid Water Path). Other data of interest in combination with cloud radar data are thermodynamic profiles (temperature and humidity profiles). Passive RPG microwave instruments like the RPG-HATPRO provide such products and do not require a matched beam of radar and passive microwave sensors.

Connect the two instruments to the same network. The passive radiometers provide a data server interface (RDS = Radiometer Data Server) for downloading the currently measured profiles online by external software. The radar software provides the data interface to locate a radiometer within the network and automatically downloads the newest temperature and humidity profiles when the connected radiometer is a profiler and currently running a measurement. If the radar does not detect a radiometer, it creates standard atmosphere profiles tuned by the met staton’s surface parameters.

The thermodynamic profiles are stored to both, level 0 and level 1 data files, together with the radar samples data.

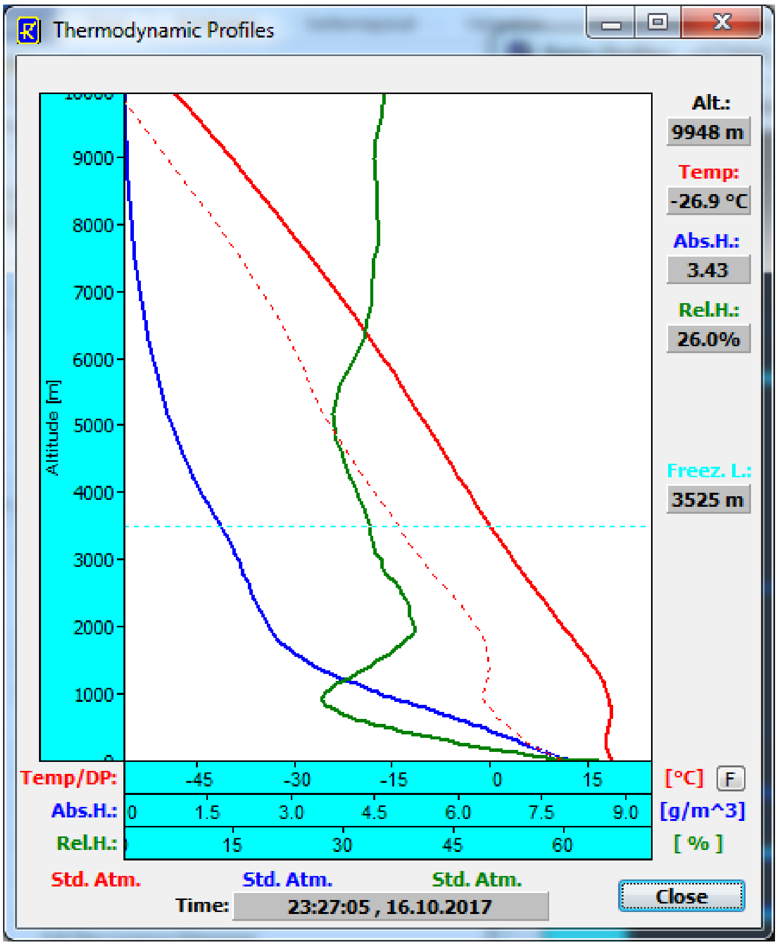
In order to establish a connection between the radar and a passive radiometer like the RPG-HATPRO, define interface parameters by starting the ***Radiometer Interfacing*** menu ():



Radiometer Interfacing menu.

The most fundamental parameter is the RDS’ IP address within the common network. It is important to understand that only the radar PC is connecting to the RDS, but NOT the radar Host PC itself! Therefore, the thermodynamic profiles are stored to the radar’s data files without the need for being connected to a Host PC.

The RDS provides a fixed port address (typically 7000). It also utilizes a User Password (UPW) to authorize access to radiometer data. When the radar downloads profiles from the radiometer, it needs this UPW (only if password checking is enabled). The UPW and enabling password checking are both set on the radiometer PC and cannot be changed via the radar interface. The UPW entered to the dedicated box in the ***Radiometer Interfacing*** menu is NOT necessarily identical to the UPW defined for the radar Host PC → radar PC communication. If password checking is activated for the RDS, the checkbox ***Enable User Password*** must be checked and the valid password should be entered. The communication between the radar PC and a radiometer is enabled / disabled (***Enable Rad. Comm.*** checkbox).
The current interfacing settings are loaded with ***Get Current Parameters from Radar*** and stored to the radar PC by ***Send Parameters to Radar***. If a communication to the RDS is established, this is indicated by a corresponding message in the yellow message field and the latest profiles are displayed by clicking ***Get Current Profiles***. In order to continuously check for the Interface status, the ***Scan Current Interface Status*** button is clicked. The status checking automatically stops when the menu is exited.
During measurements, the current radar reflectivity profiles and Doppler maps are displayed within the ***Radar Reflectivity Profile*** window, which contains the checkbox ***Display Thermodynamic Diagrams***. The same checkbox is found in the ***Reflectivity and Sensitivity Profile*** window when opening existing data files from the datra archive: The display looks like this:



Thermodynamic Profiles Display.

The following profiles are shown:

- temperature profile (in red)
- dew point profile (in dotted red)
- absolute humidity profiles (in blue)
- relative humidity profile (in green)

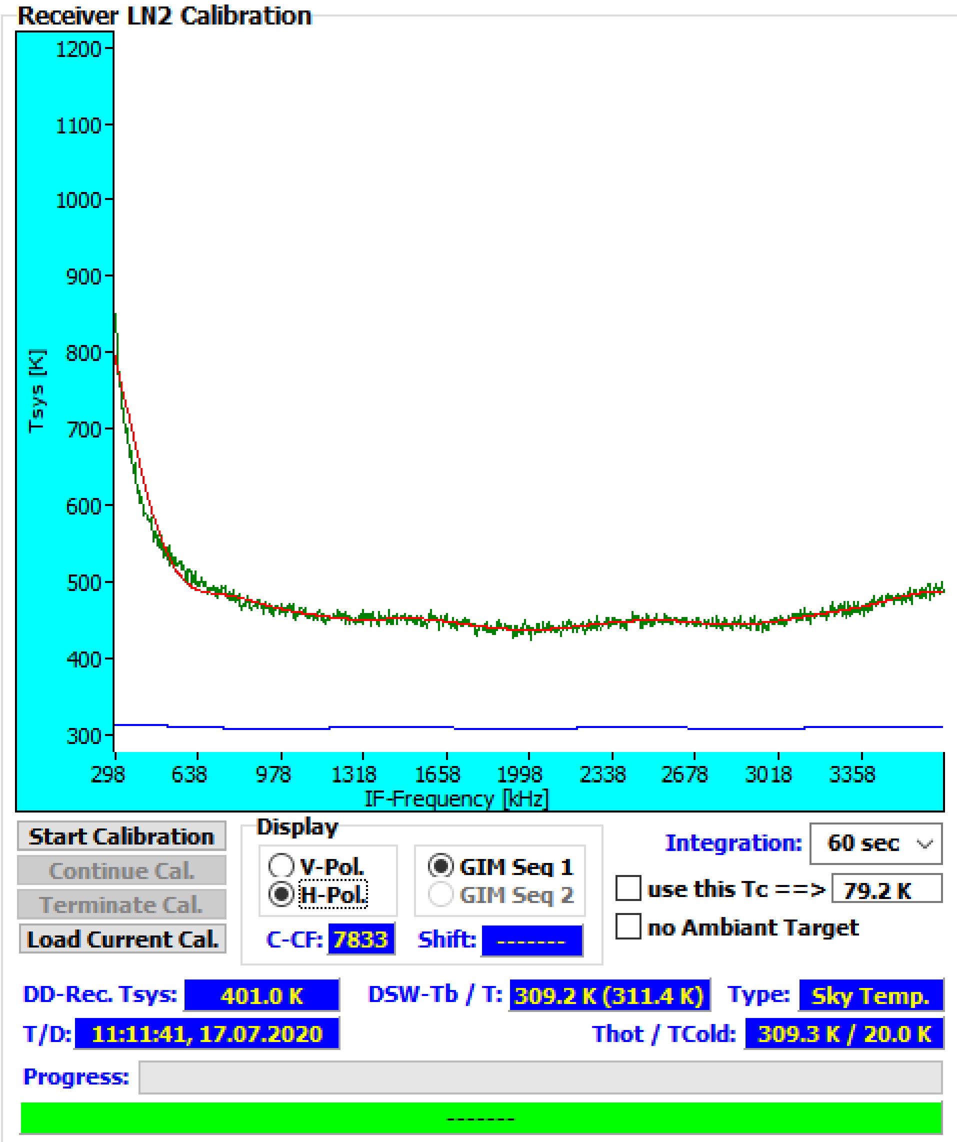
The freezing layer is indicated as a light dotted blue line. The temperature axis uint can be selected to be °C, °F or K. Profile types, as standard atmosphere (Std. Atm.) and radiometer profile (Radiom. Prof.) are shown below the diagram along with the profile sample time. When running the mouse cursor over the profile diplay, a display of cursor coordinates is listed on the right side of the diagram.
The thermodynamic profiles are important information for the level 2 processor (separate software product available at RPG), which uses humidity profiles for the correction of radar signal gas absorption and temperature profiles for the classification of hydro meteors, the determination of ice particle types in certain altitudes and detection of undercooled liquid water. The freezing level should also be consistent with the melting layer height detected by the radar.

2.16 Calibration Menus

The ***Radar Calibrations*** register page summarizes all radar calibrations described in chapters 1.5, 1.6 and 1.8.
The receiver absolute calibration has been described in detail in section 1.5. It can be executed remotely from the host PC, if it is connected to the radar and no measurement is running. The radar is assuming an LN2 cooled cold calibration target as long as the ***use this Tc*** box is not checked. If checked, the temperature entry (in K) right to the ➔ arrow is used for the cold target temperature. This applies if the cloud free sky shall be scanned instead of the LN2 target, but it requires the knowledge of the sky temperature from a measurement of a different instrument, for instance a profiler like the RPG-HATPRO-G4.
The absolute calibration is started by clicking the ***Start Calib.*** button. The calibration status is continuously monitored in the message line. When a new target needs to be placed in front of the receiver, the calibration pauses and waits for the user to click ***Continue Cal.***.
The calibration is divided in three steps:

- Integration on ambient target (1 minute)
- Integration on cold target (1 minute)
- Integration on Dicke switch (1 minute)

After all steps have passed, the calibration is finished and its results are displayed graphically for each radar IF bin as well as for the passive 89 GHz channel.



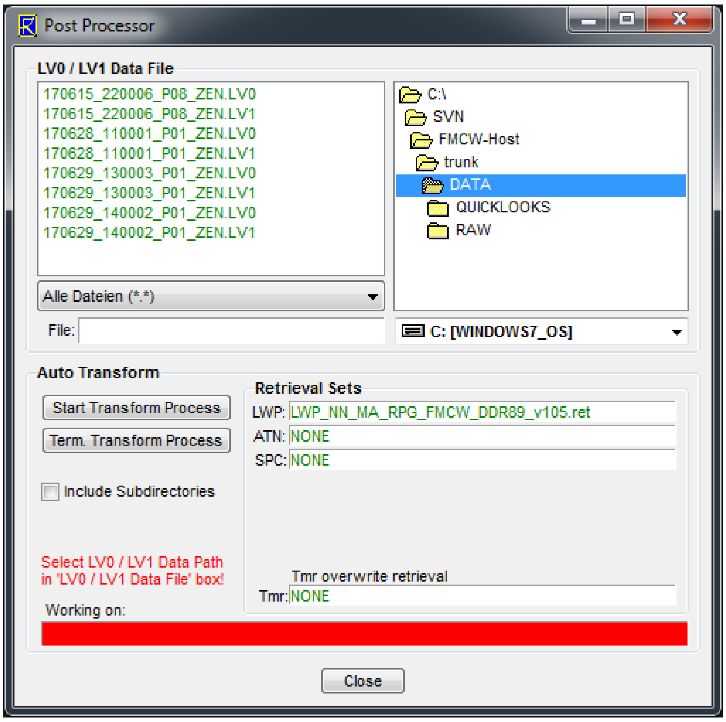
Absolute receiver calibration box

2.17 Post Processor

RPG cloud radars offer a broad band passive channel (31.4 GHz for K-band radar, 89 GHz for W-band radar) observing the same angular volume as the radar itself. The brightness temperature measured in the passive channel can be used to estimate the integrated liquid water (LWP = Liquid Water Path) in clouds which is very useful when the radar data is quantitatively evaluated in terms of liquid water clouds. It provides an independent consistency check for radar algorithms that derive liquid water profiles.

The basis of the LWP calculation is a retrieval that has been developed by radiative transfer computations of a large number of atmospheric conditions to estimate the sky brightness temperature Tb as a function of atmospheric state. The passive channel measures the Tb and therefore the retrieval is finally needed to perform the backward step of deriving atmospheric quantities from Tb and some surface parameters (temperature, rel. humidity, barometric pressure). RPG’s retrievals are all neural network (NN) based algorithms. The NN parameters are stored in ASCII files delivered with the instrument.

The retrieval files need to be stored in a sub-folder ***RETRIEVAL*** (see section 2.1.2 “Directory Tree”) which again has individual sub-folders for different retrieval types. For instance, a LWP-retrieval would be stored in the directory ***…\RPG-FMCW\RETRIEVAL\LWP***.
All data files (LV0 and LV1) contain a Tb-time series that can be processes by a retrieval which then stores the result (e.g. LWP) to the same data file. This processing can be automated and the Host PC software provides a tool for that, the **Post Processor**. The tool opens when clicking  :



Retrieval post processor menu

First of all the user marks the directory where the data files are located (upper right corner). Then it is required to select one of the installed retrievals. For instance, in order to select a LWP-retrieval, click on the white box right to the **LWP:** label within the **Retrieval Sets** box. The list of installed LWP retrievals is then displayed. Now choose one of them. It will be shown next to the **LWP:** label.
The post processor is an efficient tool when a huge amount of files shall be processed. It is possible to process a data directory including all its sub-directories. In order to do so, check the **Include Subdirectories** box. Then start the processing with ***Start Transform Process***. A background process is handling the task without interfering with measurements or other SW activities and can be terminated any time (***Term.Transform Process***).

2.18 Data Server

For script-based control, the Host can act as a server, which accepts some elementary commands via TCP-IP and directly executes them without manual intervention.
The following assumes that the Host PC is connected to a network and is assigned a valid IP address (IP-A). The data server is reached through a port set within the menu **Host Adapter Settings**, for instance 7000, thus **IP-A:7000**.

A data stream sent to the server starts with a single byte representing the command to be executed by the server. This is the list of currently implemented commands (data transfer commands in blue, MDF commands in red and control commands in green):

Code	Description
170	**DS_TERM_MEAS: Terminate the measurement currently run on the radar**
171	**DS_START_MEAS: Starts a measurement with a locally stored MDF / MBF**
172	**DS_TRANS_MDF: Starts a measurement with a MDF transmitted remotely**
173	**DS_UPDATE_STAT: Returns current radar operation status information**
174	**DS_LIST_MDF_DIR: Returns all MDF / MBF filenames in MDF_MBF dir.**
175	**DS_RADAR_ID: Returns radar system information**
176	**DS_INSTALL_MDF: Installs MDF in standard directory MDF_MBF**
177	**DS_GET_RAD_SAMPLE: Returns current radar sample data**

Table 4.1: List of data server commands

The command byte is followed by a 4 byte password code (PWC). The PWC represents the user password defined on the radiometer PC (can be set by the Administrator, see section 2.10). The PWC is calculated the following way:

1. The password is first converted to uppercase
2. The PWC is then the sum of the ASCII code of each character multiplied by the square of its position in the password

Example: The User Password shall be ‘RadUser’. Then the PWC is:

PWC = 1 * ‘R’ + 4 * ‘A’ + 9 * ‘D’ + 16 * ‘U’ + 25 * ‘S’ + 36 * ‘E’ + 49 * ‘R’ = 10891

With ‘R’ = 82 (ASCII code of character R), etc.
The PWC must be sent as little endian (least significant byte first), which is the standard for Microsoft operating systems. The correct byte sequence starting a measurement with user password ‘RadUser’ would be (decimal): 171 139 42 0 0.
If there is no user password activated on the radar PC, the PWC is not evaluated by the server and can be an arbitrary 4 byte integer number. Note, that even in this case an arbitrary 4 byte PWC must be sent (for example four zero bytes).
When a user password is activated, the PWC will be evaluated. Does the PWC match the user password code on the radiometer, the server responds with the repetition of the command code (if the command code represents a valid command) followed by the requested data block. If the PWC does not match, the server only responds with the PW_ERROR byte (0xFF = 255). When an invalid command code is sent, the server responds with a single INVALID_COM (0xFD = 253) byte.

When a valid command byte followed by the correct PWC was sent to the server, the first response byte is the command byte (as a confirmation) followed by the data block which is described in the following sections.

2.18.1 DS_TERM_MEAS command (170)

This command is used to terminate a running measurement on the radar. Only the command code is sent without any additional data.

Variable Name	Type	# Bytes	Description
ComByte	byte	1	=170
PWC	integer	4	password code

Return Byte1:
ComByte (=170)
Return Status (Byte2):
0: Host is not connected to radar
1: no measurement runnung, STANDBY mode
2: running measurement will be terminated
3: cannot terminate: zero calibration running
4: no measurement: absolute calibration running
5: cannot terminate: transmitter power calibration running
6: Host is connected to Slave radar! Slave cannot terminate measurement

2.18.2 DS_START_MEAS command (171)

This command starts a new measurement MDF / MBF, if the radar is in STANDBY mode. Because the command requires the definition of a MDF / MBF filename, the command structure is extended compared to the commands described so far. The command structure is as follows:

Variable Name	Type	# Bytes	Description
ComByte	byte	1	=171
PWC	integer	4	password code
PathStr	char	PathLen+1	Null terminated Path string to MDF / MBF on Host PC (local MDF / MBF), PathLen is the number of characters in PathStr

If **PathStr** is just the name of a MDF / MBF file (like 'Zenith_Home_V87.MDF'), the file path is assumed to be the default MDF / MBF directory (…\RootDir\MDF_MBF). If the MDF / MBF is located in a different directory, the full path needs to be specified in **PathStr**, for instance C:\MyMDFs\ Zenith_Home_V87.MDF.
The server responds in the following way:

Return Byte1:

ComByte (=171)
Return Status (Byte2):
0: Host is not connected to radar
1: radar in STANDBY mode: starting measurement
2: radar not in STANDBY mode
3: specified MDF/MBF path not valid
4: Host is connected to Slave radar. Slave cannot start measurement

2.18.3 DS_TRANS_MDF command (172)

This command starts a new measurement MDF, if the radar is in STANDBY mode. The command requires the definition of a full path to a MDF file on the remote PC, which is accessing the data server. The command structure is as follows:

Variable Name	Type	# Bytes	Description
ComByte	byte	1	=172
PWC	integer	4	password code
PathStr	char	PathLen+1	Null terminated path string to MDF file on remote PC, PathLen is the number of characters in PathStr
FileLen	integer	4	MDF file length in Bytes
Data	char	FileLen	MDF file contents

Return Byte1:
ComByte (=172)
Return Status (Byte2):
0: Host is not connected to radar
1: radar in STANDBY mode: starting measurement
2: radar not in STANDBY mode
3: Host is connected to Slave radar. Slave cannot start measurement

2.18.4 DS_UPDATE_STAT command (173)

This command returns the current measurement status and the radar hierarchy code. The command structure is as follows:

Variable Name	Type	# Bytes	Description
ComByte	byte	1	=173
PWC	integer	4	password code

Return Byte1:
ComByte (=173)
Return Status (Byte2): Connected:
0: Host is not connected to radar
1: Host is connected

Return Status (Byte3): Measurement Status (only if Byte2=1):
1: radar in STANDBY mode
2: measurement running
3: zero calibration running
4: absolute calibration running
5: transmitter power calibration running

Return Status (Byte4): Hirarchy (only if Byte2=1):
0: single radar
1: dual radar; radar is in Master mode
2: dual radar; radar is in Slave mode
For script-based control, the Host can act as a server, which accepts some elementary commands via TCP-IP and directly executes them without manual intervention.

2.18.5 DS_LIST_MDF_DIR command (174)

This command returns all MDF and MBF filenames stored in the standard MDF_MBF directory. The command structure is as follows:

Variable Name	Type	# Bytes	Description
ComByte	byte	1	=174
PWC	integer	4	password code

Return Structure:

Variable Name	Type	# Bytes	Description
ComByte	byte	1	=174
FilCnt	integer	4	Number of filenames
FnStr(1)	char	StrLen+1	Null terminated path string of first MDF filename
…	…	…	…
FnStr(FilCnt)	char	StrLen+1	Null terminated path string of last MDF filename

2.18.6 DS_RADAR_ID command (175)

This command returns radar system information. The command structure is as follows:

Variable Name	Type	# Bytes	Description
ComByte	byte	1	=175
PWC	integer	4	password code

Return Structure:

Variable Name	Type	# Bytes	Description
ComByte	byte	1	=175
Connected	byte	1	0: Host isnot connected to radar 1: Host is connected to radar
The following data is only valid, if Connected = 1			
SWVersNum	float	4	Radar PC software version number
SWSubVersNum	int	4	Radar PC software sub-version number
RadModel	int	4	Radar Model number: 0: 94 GHz single pol. radar 1: 94 GHz dual pol. radar 2: 35 GHz single pol. radar 3: 35 GHz dual pol. radar
FabrYear	int	4	Year of radar fabrication
FabriNo	int	4	Fabrication number (during fabrication year)
CustomerStr	char	StrLen+1	Null terminated string of customer ID
LicEndT(1)	int	4	End of license time (0 = UNLIMITED)
ScanAvail	byte	1	Scanner availability (0=NO, 1=YES)
DualPol	byte	1	Dual polarization mode: 0: single pol. radar 1: dual pol. radar (LDR mode) 2: dual pol. radar (STSR mode)
IFRngMin	float	4	IF frequency range min. frequency [Hz]
IFRngMax	float	4	IF frequency range max. frequency [Hz]
RadarWl	float	4	Radar operating wavelength [m]
AntDiam	float	4	Antenna diameter [m]
AntSep	float	4	Antenna pitch [m]
AntGain	float	4	Antenna gain [linear]
AntHPBW	float	4	Antenna half power beam width [DEG]
SubRefBlk	float	4	Sub-reflector blocking [%]
RecGainV	float	4	V-pol. receiver gain [dB]
RecGainH	float	4	H-pol. receiver gain [dB]
FFTWin	int	4	FFT window: 0: rectangle window 1: Parzen window 2: Blackman window 3: Welch window 4: Slepian2 5: Slepian3
AutoRecov	byte	1	Auto recovery mode (0:NO, 1:YES)
AbsCAlAvail	byte	1	Absolute receiver calibration availability: 0: NO 1: YES
PowSupDiag	byte	1	Power suppy diagnostic feature: 0: not installed 1: installed
AzimOffs	float	4	Azimuth offset angle [DEG]
StatusIL	byte	1	InterLAN status:

			0: automatic IL detection 1: IL detection disabled
RadIP	char	StrLen+1	Radar IP null terminated string

(1) The time is expressed in number of seconds since 1.1.2001, 00:00:00

2.18.7 DS_INSTALL_MDF command (176)

This command installs a new MDF to the standard MDF_MBF directory. The command structure is as follows:

Variable Name	Type	# Bytes	Description
ComByte	byte	1	=176
PWC	integer	4	password code
Fn	char	StrLen+1	null terminated filename string (not including path) of new MDF
FileLen	integer	4	Number of bytes in MDF file
MDFDat	byte	FileLen	MDF file contents

2.18.8 DS_GET_RAD_SAMPLE command (177)

This command returns the complete latest radar sample data (LV0 + LV1). The command structure is as follows:

Variable Name	Type	# Bytes	Description
ComByte	byte	1	=177
PWC	integer	4	password code

Return Structure:

Variable Name	Type	# Bytes	Description
ComByte	byte	1	=177
MeasRunning	byte	1	0: measurement not running 1: measurement running
The following data is only valid, if MeasRunning = 1			
SampIdx	int	4	Sample index number on radar PC
LV0HeaderLen	int	4	LV0 header data length in bytes
CGProg	int	4	chirp generator program number
RadModel	int	4	Radar Model number: 0: 94 GHz single pol. radar 1: 94 GHz dual pol. radar 2: 35 GHz single pol. radar 3: 35 GHz dual pol. radar

ProgNameStr	char	StrLen+1	Null terminated string of chirp program name
CustomerIDStr	char	StrLen+1	Null terminated string of customer ID
RadFr	float	4	Radar frequency [GHz]
AntSep	float	4	Antenna pitch [m]
AntDiam	float	4	Antenna diameter [m]
AntGain	float	4	Antenna gain [linear]
AntHPBW	float	4	Antenna half power beam width [DEG]
Cr	float	4	Radar constant Cr
DualPol	byte	1	Dual polarization mode: 0: single pol. radar 1: dual pol. radar (LDR mode) 2: dual pol. radar (STSR mode)
CompEna	byte	1	Doppler spectra compression 0: not compressed 1: compressed, no spectral polarim. var. 2: compressed, spectral polarim. var. stored
AntiAlias	byte	1	Anti Aliasing (0: not enabled, 1: enabled)
SampDur	float	4	Sample duration [sec]
GPSLat	float	4	GPS latitude
GPSLong	float	4	GPS longitude
CalInt	int	4	period for automatic zero calibrations in number of samples
RAltN	int	4	number of radar ranging layers
TAltN	int	4	number of temperature profile layers
HAltN	int	4	number of humidity profile layers layers
SequN	int	4	number of chirp sequences
RAlts[ ]	float	4 x RAltN	ranging altitude layers [m]
TAlts[ ]	float	4 x TAltN	temp. profile altitude layers (only if TAltN>0)
HAlts[ ]	float	4 x HAltN	hum. profile altitude layers (only if HAltN>0)
SpecN[ ]	int	4 x SequN	number of samples in Doppler spectra of each chirp sequence
RngOffs[ ]	int	4 x SequN	chirp sequence start index in altitude layer array
ChirpReps[ ]	int	4 x SequN	number chirps within a sequence
SeqIntTime[ ]	float	4 x SequN	effective sequence integration time [sec]
dR[ ]	float	4 x SequN	chirp sequence range resolution [m]
MaxVel[ ]	float	4 x SequN	max. Doppler velocity [m/s] for each chirp sequence (unambiguous)
ChirpCenterFr[ ]	float	4 x SequN	chirp centre frequency [MHz] at radar transmitter output
InstCalPar	int	4	Calibration period [sec]
SampRate	int	4	ADC sampling rate [Hz]

MaxRange	int	4	maximum unambiguous range [m]
SupPowLev	byte	1	flag indicating, if power levelling has been used (0=yes, 1=no)
SpkFilEna	byte	1	flag indicating, if spike/plankton filter has been used (1=yes, 0=no)
PhaseCorr	byte	1	flag indicating, if phase correction (dual. pol. radars) has been used (1=yes, 0=no)
RelPowCorr	byte	1	flag indicating, if relative power correction (dual. pol. radars) has been used (1=yes, 0=no)
FFTWindow	byte	1	FFT window in use: 0 = SQUARE 1 = PARZEN 2 = BLACKMAN 3 = WELCH 4 = SLEPIAN2 5 = SLEPIAN3
FFTInputRng	int	4	ADC input voltage range (+/-) [mV]
NoiseFilt	float	4	noise filter threshold factor (multiple of STD in Doppler spectra)
CTProg	int	4	Current chirp table program number
LimitedMeas	byte	1	Measurement limitation flag: 0: UNLIMITED 1: LIMITED
The following data is only valid, if LimitedMeas = 1			
EndOfMeas(1)	int	4	End of measurement time
ScanType	byte	1	Scan type: 0: constant angle 1: general scan
ScanMode	byte	1	Scan mode: 0: continuous scan 1: discrete scan
ScanDur	int	4	Approx. scan duration in seconds
BaseFn	char	StrLen+1	null terminated string of base filename
ArchData	byte	1	Data archiving flag: 0: no archiving 1: archiving enabled
DiskFull	byte	1	Disk full flag: 0: not full 1: disk full
SupPowLev	byte	1	Suppress power levelling flag: 0: no suppression 1: suppress power levelling
MeasTrig	byte	1	Measurement triggering flag: 0: not triggered 1: measurement triggered
MeasStartT(1)	int	4	Measurement start time

MCC	byte	1	Measurement classification code: 0: zenith observation 1: constant elevation observation 2: RHI scan 3: RHI scan with azimuth wind dir. 4: PPI scan 5: mixed scan (elev. and azi. changing)
StoreLV0	byte	1	LV0 data storage (0: NO, 1:YES)
StoreLV1	byte	1	LV0 data storage (0: NO, 1:YES)
RepIdx	int	4	Current repetition number in MBF
MDFIdx	int	4	Current MDF number
RepN	int	4	Total number of repetitions
MDFsN	int	4	Number of MDFs in MBF
BatchFn	char	StrLen+1	null terminated string of MBF filnename
MDFFn(1)	char	StrLen+1	null terminated string of MDF1 filnename
…	…	…	…
MDFFn(MDFsN)	char	StrLen+1	null terminated string of MDF(MDFsN)
End of Header			
LV0 Data			
SampT(1)	unsigned int	4	Sample time
SampMs	int	4	Milliseconds of sample time
QF	byte	1	Quality flag: Bit 1: ADC saturation Bit 2: spectral width too high Bit 3: no transm. power leveling
RR	float	4	rain rate [mm/h]
RelHum	float	4	rel. humidity [%]
EnvT	float	4	environmental temp. [K]
BaroP	float	4	barometric pressure [hPa]
WS	float	4	wind speed [km/h]
WD	float	4	wind direction [°]
DDVolt	float	4	direct detection channel voltage [V]
DDTb	float	4	direct detection brightness temp. [K]
LWP	float	4	liquid water path [g/m^2]
PowIF	float	4	IF power at ADC [µW]
Elev	float	4	elevation angle [°]
Azi	float	4	aszimuth angle [°]
Status	float	4	mitigation status flags: 0/1: heater switch (ON/OFF) 0/10: blower switch (ON/OFF)
TransPow	float	4	transmitter power [W]
TransT	float	4	transmitter temp. [K]
RecT	float	4	receiver temp. [K]
PCT	float	4	PC temp. [K]
SkyTb	float	4	Sky brightness temp. [K]: TB[v] + TB[h] / 10000
InclEl	float	4	Inclination angle of elevation zenith

			position [DEG]
InclElAx	float	4	Inclination angle of elevation axis (perpendicular to InclEl_1) [DEG]
TPr[ ]	float	TAltN x 4	temp. profile [K]
AHPr[ ]	float	HAltN x 4	abs. hum. Profile [g/m^3]
RHPr[ ]	float	TAltN x 4	rel. hum. profile [%]
SLv[ ]	float	RAltN x 4	linear sensitivity limit in Ze units for vertical polarisation
PrMsk[ ]	byte	RAltN	mask array of occupied range cells: 0: range cell not occupied 1: range cell occupied
The loop over all range bins (loop index n) starts here			
The following data is only stored, if PrMsk[n]=1			
BlockN[n]	byte	1	number of compression blocks in spectrum
MinBlkIdx[n] [ ]	short int	BlockN[n] x 2	minimum index field of blocks in spectrum
MaxBlkIdx[n] [ ]	short int	BlockN[n] x 2	maximum index field of blocks in spectrum
VSpec[n] [ ]	float	BlockN[n] x (MaxBlkIdx[n][] - MinBlkIdx[n][]+1) x 4	compressed V-pol. Doppler spectrum, linear Ze, loop over BlockN[n] blocks
The following data is only stored, if AntiAlias=1			
AliasMsk[n]	byte	1	mask indicating, if anti-aliasing has been applied (=1) or not (=0)
MinVel[n]	float	4	minimum velocity in Doppler spectrum [m/s]
LV1 Data			
The loop over all range bins (loop index n) starts here			
The following data is only stored, if PrMsk[n]=1			
Ze[n]	long long int	8	1e7 * reflectivity in dBZe units for vert. pol. in range bin n
MeanVel[n]	long long int	8	1e7 * mean velocity [m/s] for vert. pol. in range bin n
SpecWidth[n]	long long int	8	1e7 * spectral width [m/s] for vert. pol. in range bin n
Skewness[n]	long long int	8	1e7 * spectral skewness for vert. pol. in range bin n
Kurtosis[n]	long long int	8	1e7 * spectral kurtosis for vert. pol. in range bin n
The following data is only stored, if DualPol > 0 (dual pol. radar)			
RefRat[n]	long long int	8	1e7 * Differential reflectivity in range bin n, table 2.11.2 (product 1.), [dB]
CorrC[n]	long long int	8	1e7 * Correlation coefficient in range

			bin n, table 2.11.2 (product 2.), [0,…,1]
DiffPh[n]	long long int	8	1e7 * differential phase in range bin n, table 2.11.2 (product 3.), [rad]
The following data is only stored, if DualPol = 2 (STSR mode)			
Ze45[n]	long long int	8	1e7 * Combined (v/h) reflectivity in dBZe units in range bin n
SLDR[n]	long long int	8	1e7 * slanted LDR in range bin n, table 2.11.2 (product 4.), [dB]
SCorrC[n]	long long int	8	1e7 * slanted correlation coefficient in range bin n, table 2.11.2 (product 5.), [0,…,1]
KDP[n]	long long int	8	1e7 * specific differential phase shift in range bin n, see equ. (2.10.13), [rad / km]
DiffAtt[n]	long long int	8	1e7 * differential attenuation in range bin n, see equ. (2.10.14), [dB / km]

(1) The time is expressed in number of seconds since 1.1.2001, 00:00:00

2.18.9 DS_GET_HKD_SAMPLE command (178)

This command returns the latest HKD sample data. The command structure is as follows:

Variable Name	Type	# Bytes	Description
ComByte	byte	1	=178
PWC	integer	4	password code

Return Structure:

Variable Name	Type	# Bytes	Description
ComByte	byte	1	=178
GPSFlag	byte	1	0: GPS clock not found 1: GPS clock found
The following data is only valid, if GPSFlag = 1			
PosStat	byte	1	0: positioning… 1: position valid
TimeStat	byte	1	0: radar not synchronized yet 1: radar synchronized
If PosStat = 1			

GPSlongitude	float	4	Longitude
GPSlatitude	float	4	Latitude
GPSPosTime(1)	int	4	Time of positioning
		If TimeStat = 1	
GPSSyncTime(1)	int	4	Time of radar synchronization
RadarTime(1)	int	4	Time on radar (UTC)
MetFound	byte	1	0: Met Station not found 1: Met Station found
		If MetFound = 1	
EnvTemp	float	4	Environmental Temp. [K]
Pressure	float	4	Barometric Pressure [hPa]
RelHum	float	4	Relative Humidity [%]
WindSp	float	4	Wind Speed [km/h]
WindDir	float	4	Wind Direction [GEG], 0 = North
ScannerFound	byte	1	0: scanner not found 1: scanner found
		If ScannerFound = 1	
Elev	float	4	Elevation [DEG]
Azi	float	4	Azimuth [DEG]
RecTemp	float	4	Receiver Temp. [K]
TransTemp	float	4	Transmitter Temp. [K]
PCTemp	float	4	Radar PC Temp. [K]
RainStat	byte	1	0: not raining 1: raining
HeatSwitch	int	4	0: heater off 1: heater on
BlowSwitch	int	4	0: blower off 1: blower on
RadDriveCnt	int	4	Number of radar drives
		If RadDriveCnt > 0	
FreeMem[ ]	int	4 x RadDriveCnt	Free memory array [MByte]
TotMem[ ]	int	4 x RadDriveCnt	Total memory [MByte]
IncEl	float	4	Elevation inclination [DEG]
IncElAx	float	4	Elevation axis inclination [DEG]
MeasMode	byte	1	0: STANDBY 1: Sampling 2: Waiting for Trigger

			3: Scanning 4: zero calibration 5: Transmitter power cal.
Hirarchy	byte	1	0: none (single radar) 1: MASTER 2: SLAVE
If Hirarchy = 1 (MASTER)			
SlModelNo	int	4	Slave Model number: 0: RPG-FMCW94-SP 1: RPG-FMCW94-DP 2: RPG-FMCW35-SP 3: RPG-FMCW35-DP
SLError	byte	1	0: No Error 1: License expired 2: insufficient memory for meas. 3: Transmitter power too low 4: No calibration found 5: Slave timeout
MeasRun	byte	1	0: STANDBY 1: Measurement running
HDDOverFlow	byte	1	0: HDD no overflow 1: HDD overflow
AlarmCode	byte	1	0: OK 1: Radr PC overheated

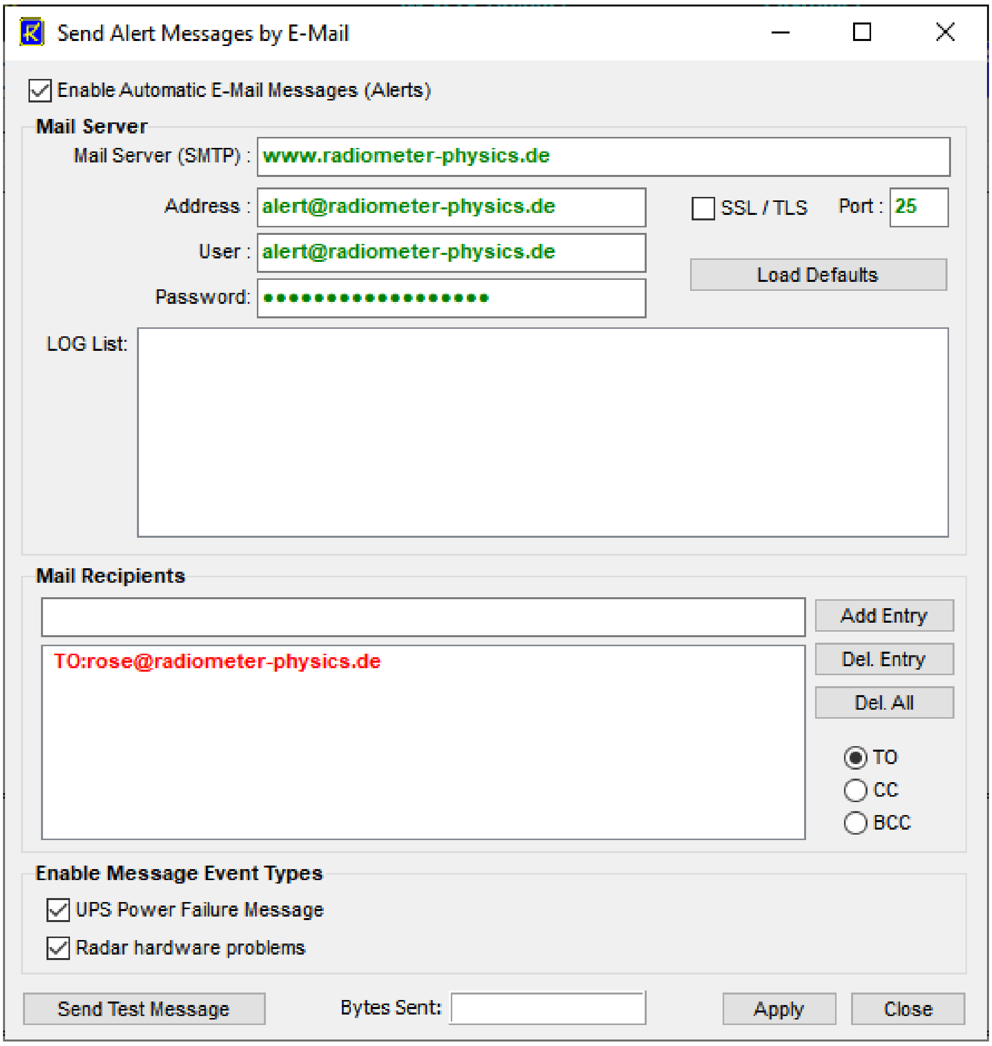
(1) The time is expressed in number of seconds since 1.1.2001, 00:00:00

2.19 Automatic E-Mail Warning Messages

While a measurement is running, several events that may require interaction by the user are of major interest. These events could be:

- Hardware malfunction of a radar component
- Freezing of the Host software for unknown reason
- A UPS system has signaled a power loss
- The Host has lost the connection to the radar

The host SW offers the possibility to send automatic e-mails to pre-defined addresses via a configurable SMTP server. Click to enter the SMTP server definition menu:



The first step is to define the SMTP server details. The user may configure a server of his choice (enter the server, a user address with user and password and the port number for the server). The default mail server that is listed when clicking ***Load Defaults*** is no longer available.
Then enter the address of a mail recipient as a message target and add it to the mail list (***Add Entry***). Before adding an address, the copy status may be selected (TO, CC, BCC).

The message types can be enabled or disabled separately. Mail server settings, the mailing list and event types are stored permanently by clicking ***Apply***.
The communication may be tested by sending a test message (***Send Test Message***). Automatic e-mails must be enabled by checking ***Enable Automatic E-Mail Messages (Alerts)***.

2.19 UPS Control

At locations with frequent electrical power blackouts, it is recommended to use a UPS system (Unbreakable Power Supply), in order to keep radar measurements continuously running (short blackouts of < 10 minutes) or to provide a controlled way of measurement soft shutdown.
In principle, any UPS can be used for this purpose. However, in combination with RPG radars, several aspects need to be considered:

1. Power consumptions:

W-band radar:	300 Watts
K-band radar:	700 Watts
Blower system:	500 Watts

   From these numbers it follows, that an efficient use of a UPS requires the capability of handling about 2 kW of electrical power.
2. Control: A soft shutdown, meaning an intervention of the radar software in advance of a UPS inverter shutdown, requires interfacing compatibility between the UPS and the RPG software. The RPG software only interfaces to UPS systems from **Alpha Technologies** (FXM 650 / 1100 / 2000 UPS). Another advantage (besides the possiblility of measurement soft shutdowns) is the more efficient UPS usage by turning off the blower power consumption during battery mode.

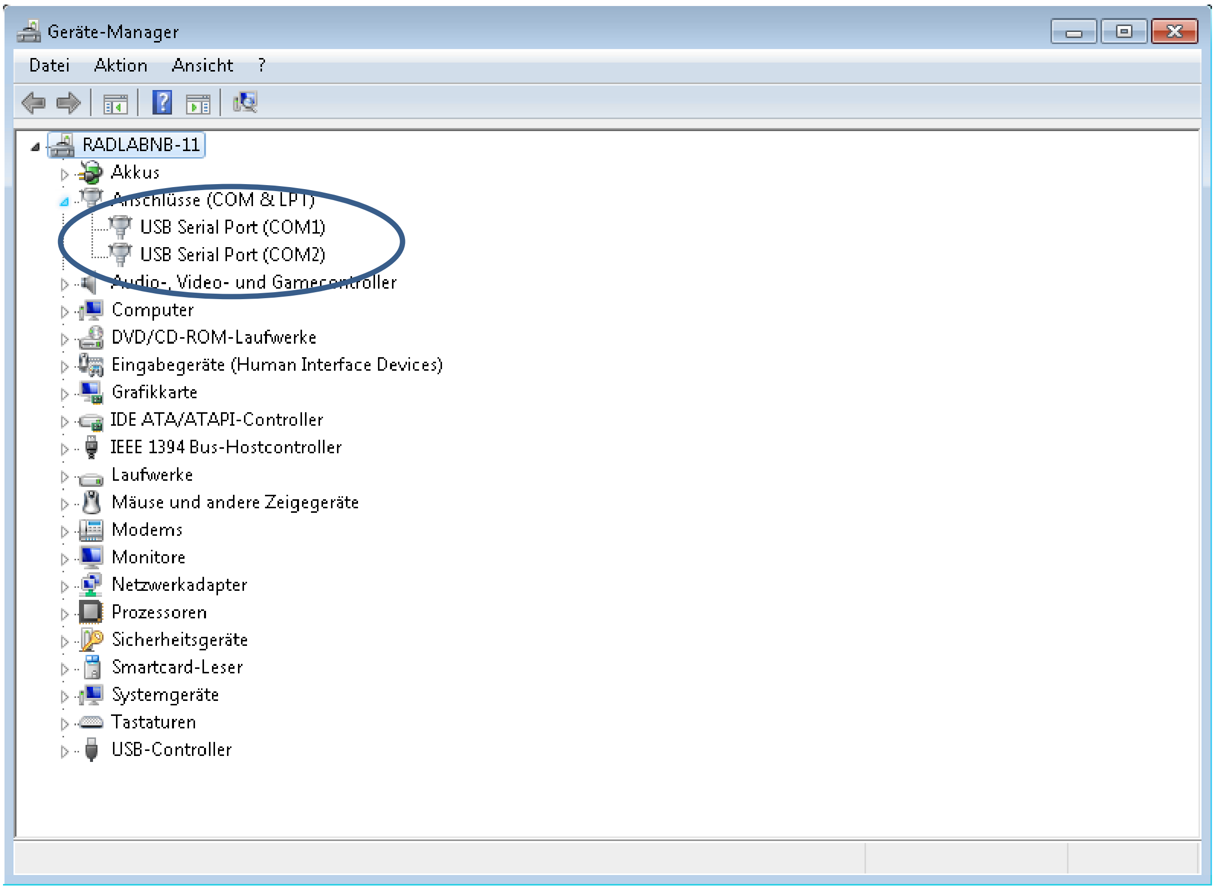
The FXM2000 from Alpha Industries is the most appropriate UPS in combination with RPG radars. It operates a battery chain of 48 Volts / DC and provides several options for communication. The connection to the radar Host PC application is realized by a slow RS-232 interface (2400 baudrate), which is the UPS standard interface. All functionality (monitoring / maintenance) can be remotely controlled via the UPS's command line system by using a Hyper Terminal (fully described in the UPS Installation & Operation Manual). The RPG Host PC software is operating this command line system.
Alpha Industries provides an optional LAN interface (FXM Communication Module), which allows to manually control the UPS from a network via a web-browser (HTML). This is a convenient way of changing UPS parameters on the fly, or to allow a visual inspection of the current UPS input / output line state. But it is not well suited for interfacing the radar software.

2.19.1 Installing the Host PC Connection

Modern PCs do not provide RS-232 interfaces anymore. Therefore, a USB-to-RS232 converter must be used. When such a converter is plugged into one of the PC's USB sockets, the OS (Operating System) automatically assigns a COM number to it, which is kind of unpredictable for applications. To overcome this problem, FTDI (one of the major providers of USB-to-RS232 converter chips) has developed a DLL with functions, that identify a specific FTDI chip with its associated COM number. Each FTDI converter chip has an on-board EEPROM, holding information about its device ID. Once this ID is defined, an application can identify the COM interface number via the associated device ID. The radar's

Host PC SW uses this mechanism to connect to the UPS. The procedure only works, if a USB-to-RS232 converter from FTDI is used, which is the most common brand on the market.

The FTDI driver and DLL can be downloaded from https://ftdichip.com/drivers/vcp-drivers/
When the converter is connected to the Host PC, the device manager will list the device:



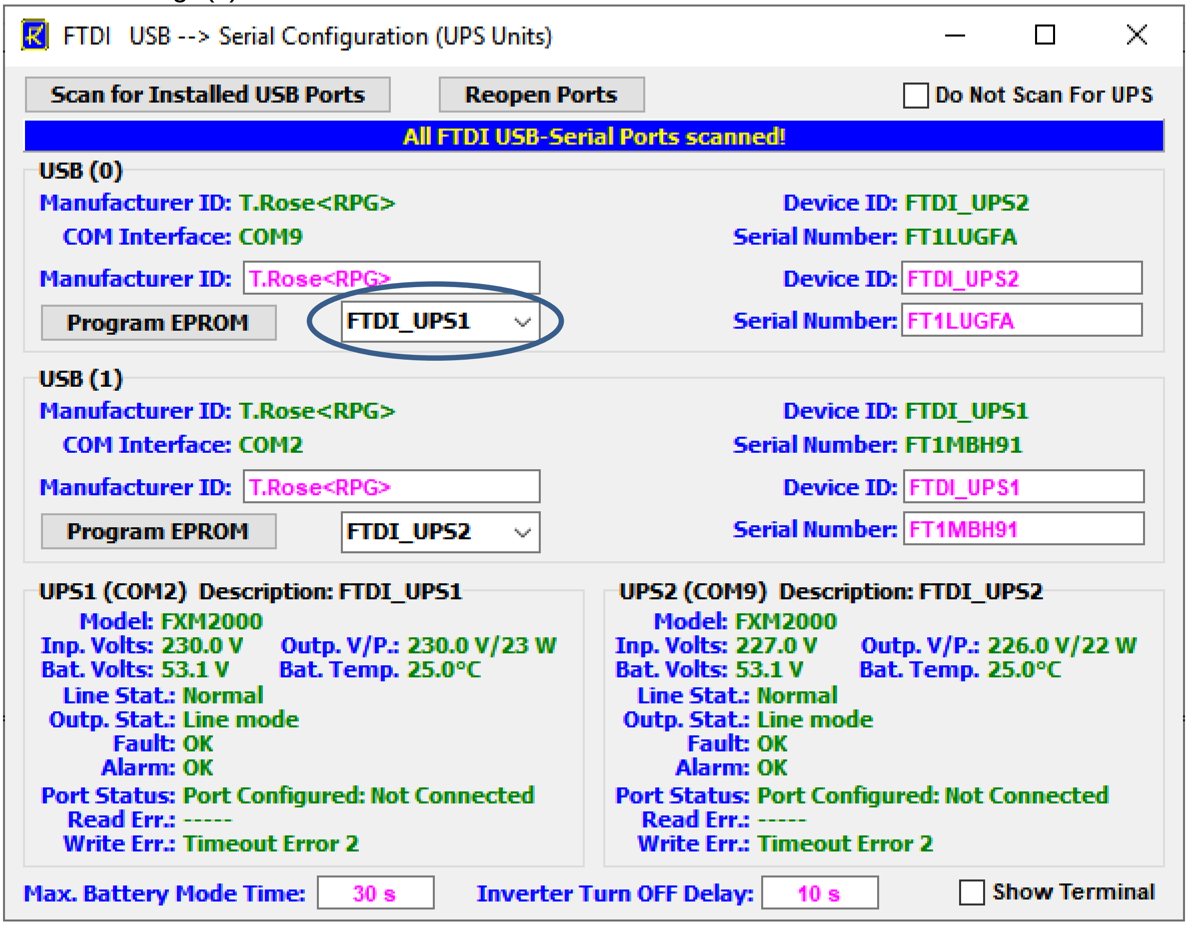
Use standard Windows procedures to install the FTDI driver and copy the correct FTD2XX.DLL file for your OS to the root directory (where FMCW_H.EXE is stored). Reboot the Host PC to get the driver registration completed and restart the radar application.
Up to two UPS controllers can be handled by the radar application (e.g. two FX2000 units). This may be required when operating a dual frequency radar (W-band + K-band radar combination on scanner) and each AC phase is buffered by a separate FX2000.

2.19.2 Configuring the Host PC Connection

When clicking the UPS icon 

from the shortcut menu, the **FTDI USB → Serial Configuration** menu opens.
The first step is to set the FTDI device ID. Click the **Scan for Installed USB Ports** button and confirm to close all open ports. The program will then scan the Host PC for all installed FTDI devices and list their parameters in the **USB(0)** and **USB(1)** boxes. <u>Make sure there are not more than two FTDI devices installed on the Host PC!</u>
The COM interface is listed, together with the current device ID and manufacturer ID (in green). You can leave the manufacturer ID as it is, but select **FTDI_UPS1** from the combo-

box. The new device ID shows up in the edit box (magenta). Then click **Program EPROM** to save the change(s) on the FTDI device.



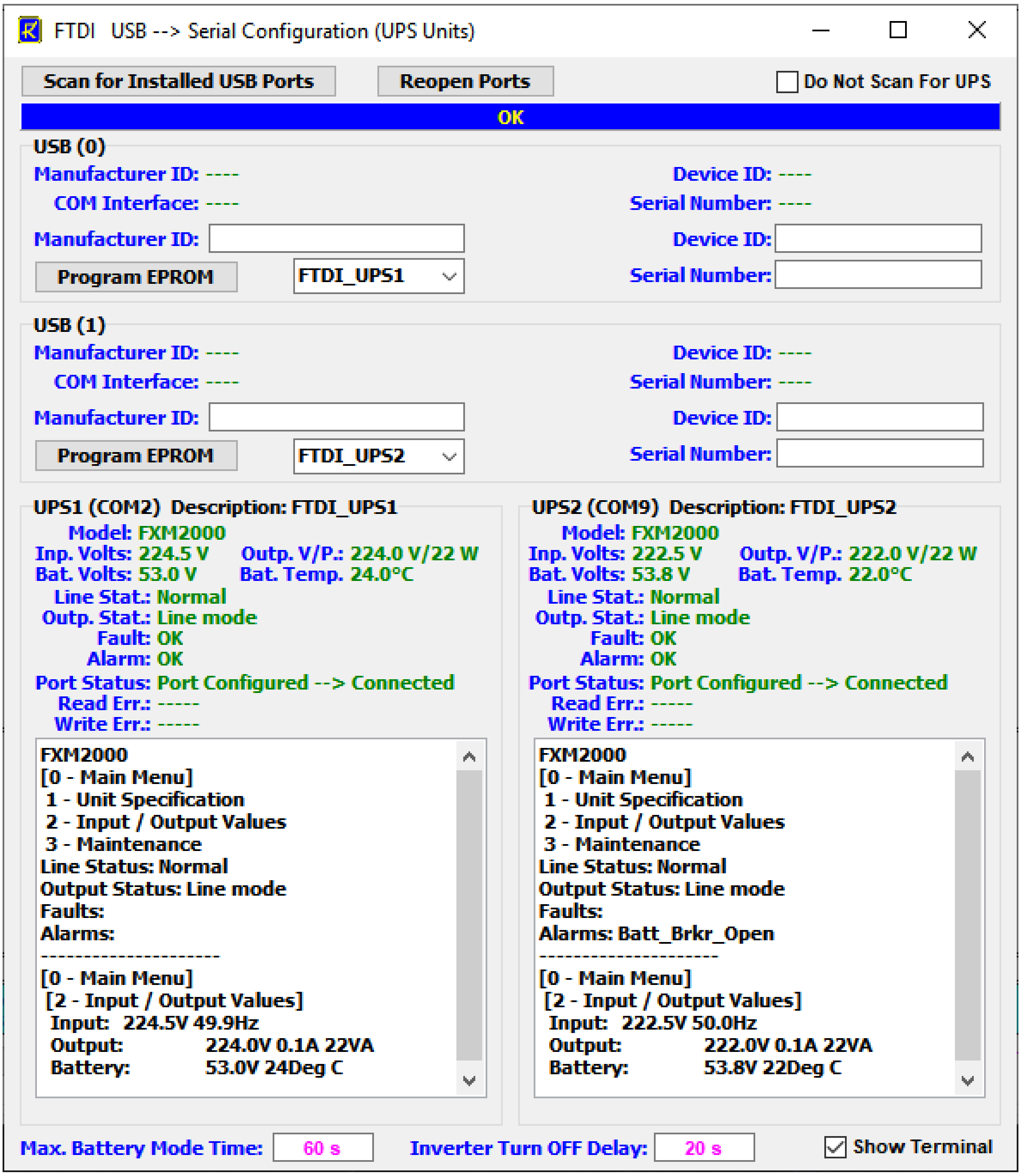
Configure the USB device ID.

Click the **Scan for Installed USB Ports** button again to see if the EPROM has been really updated. Repeat this step for a possible second FTDI device, but select **FTDI_UPS2** then as the device ID..

Now click **Reopen Ports** to connect to the RS232-interface(s). When the UPS is powered on and connected to the Host PC, there will be a display of the current UPS parameters. Next time the radar application is started, the USB devices are automatically detected and the correct COM ports are opened and connection to the UPS system(s) is established. The parameters are updated approx. every 5 seconds.

If you want to monitor the UPS command line communication, check the **Show Terminal** checkbox.

When a dual frequency radar is operated, the Host PC is monitoring each radar from a different root directory (the Host PC radar application is executed twice in different root directories). If both applications would scan for the UPS controllers, an access conflict occurs, because each RS232-interface can be opened by one application only. Therefore, the application monitoring the Slave radar (usually the K-band radar) should be disabled from scanning the USB devices. This is accomplished by checking the **Do Not Scan For UPS** checkbox. After this checkbox is checked, terminate the application and restart it. The application will then not attempt to access the RS232interface(s), so that the Master radar application can take over the full control alone (here the checkbox remains unchecked).



Monitoring the UPS command line communication.

The editable **Maximum Battery Mode Time** is the period in seconds, for which the battery maintains the output line power after an input power blackout (in Battery Mode), until the shutdown procedure starts. After this period, the radar application initiates a UPS inverter shutdown, which is delayed by the time period given in **Inverter Turn OFF Delay** (also editable). The delay is needed for the radar to perform a soft shutdown of a running measurement. During a soft shutdown, the current MDF is terminated and all collected data is stored to the radars internal HD. After that, the radar PC is shut down. If the Recover Flag is set, the MDFs are not removed from the radar, so that it can restart the measurement when the power returns.

When in Battery Mode, **the Max. Battery Mode Time** is only fully used if – in the meantime - there is no output line status change from “Battery mode” to “Battery mode, low bat warning”. If the low battery warning occurs, the **Inverter Turn OFF Delay** starts immediately.

When the UPS Line Status turns to “Blackout”, the host PC’s application shuts down the blower (if it is turned ON), in order to save battery buffering time.

The user may optimize the **Maximum Battery Mode Time** to bridge a large as possible blackout period. 10 minutes or more is a reasonable period, depending on the radar’s power consumption and battery capacity / temperature. The **Inverter Turn OFF Delay** should be >30 seconds to leave enough time for the radar PC to shut down.

Appendix A (Binary File Formats)

A2: LV0-Files (.LV0) Version 2.0*

(this file structure is used by RPG-FMCW-94-SX radars since Nov. 2016)

Variable Name	Type	# Bytes	Description
File Code	**int**	**4**	**LV0-File ID (=789346), Version 2.0**
Header starts here			
HeaderLen	**int**	**4**	**header length in bytes (not including HeaderLen)**
ProgNo	**int**	**4**	**chirp program number in chirp table**
ModelNo	**int**	**4**	**=0: 94 GHz single pol. radar =1: 94 GHz dual pol. radar**
ProgName	**char**	**Len(ProgName)+1**	**null terminated char string of chirp program name**
CustName	**char**	**Len(CustName)+1**	**null terminated char string of customer name**
Freq	**float**	**4**	**radar frequency [GHz]**
AntSep	**float**	**4**	**separation of both antenna axis (bistatic configuration), [m]**
AntDia	**float**	**4**	**antenna diameter, [m]**
AntG	**float**	**4**	**linear antenna gain**
HPBW	**float**	**4**	**antenna half power beam width [°]**
Cr	**float**	**4**	**radar constant, defined by equ. (2.1.5)**
DualPol	**char**	**1**	**=0: single pol. radar =1: dual pol. radar, LDR conf. =2: dual pol. radar, STSR mode (see section 2.11.3)**
CompEna	**char**	**1**	**spectral compression flag: 0: not compressed 1: spectra compressed 2: spectra compressed and spectral polarimetric variables are stored in the file**
AntiAlias	**char**	**1**	**0: Doppler spectra are not anti-aliased 1: Doppler spectra have been anti-aliased**
SampDur	**float**	**4**	**sample duration [sec]**
GPSLat	**float**	**4**	**GPS latitude**
GPSLong	**float**	**4**	**GPS longitude**
CalInt	**int**	**4**	**period for automatic zero**

			calibrations in number of samples
RAltN	int	4	number of radar ranging layers
TAltN	int	4	number of temperature profile layers
HAltN	int	4	number of humidity profile layers layers
SequN	int	4	number of chirp sequences
RAlts[ ]	float	4 x RAltN	ranging altitude layers
TAlts[ ]	float	4 x TAltN	temp. profile altitude layers (only if TAltN>0)
HAlts[ ]	float	4 x HAltN	hum. profile altitude layers (only if HAltN>0)
Fr[ ]	int	4 x RAltN	range factors (see equ. (2.5.6))
SpecN[ ]	int	4 x SequN	number of samples in Doppler spectra of each chirp sequence
RngOffs[ ]	int	4 x SequN	chirp sequence start index in altitude layer array
SeqAvg[ ]	int	4 x SequN	number of averaged chirps within a sequence
SeqIntTime[ ]	float	4 x SequN	effective sequence integration time [sec]
dR[ ]	float	4 x SequN	chirp sequence range resolution [m]
MaxVel[ ]	float	4 x SequN	max. Doppler velocity [m/s] for each chirp sequence (unambiguous)
Header ends here			
TotSamp	int	4	total number of samples
Sample 1 starts here			
SampBytes _1	int	4	length of sample 1 [bytes], not including SampBytes _1
Time_1[(1)]	unsigned int	4	time of sample 1 [sec]
MSec_1	int	4	milliseconds of sample 1 [msec]
QF_1	char	1	quality flag of sample 1: Bit 1: ADC saturation Bit 2: spectral width too high Bit 3: no transm. power leveling
RR_1	float	4	rain rate of sample 1 [mm/h]
RelHum_1	float	4	rel. humidity of sample 1 [%]
EnvTemp_1	float	4	environm. Temp. of sample 1 [K]
BaroP_1	float	4	barometric press. of sample 1 [hPa]
WS_1	float	4	wind speed of sample 1 [km/h]
WD_1	float	4	wind direction of sample 1 [°]
DDVolt_1	float	4	direct detection channel voltage of sample 1 [V]
DDTb_1	float	4	direct detection brightness temp. of sample 1 [K]

LWP_1	**float**	**4**	**liquid water path of sample 1 [g/m^2]**
PowIF_1	**float**	**4**	**IF power at ADC of sample 1 [µW]**
Elev_1	**float**	**4**	**elevation angle of sample 1 [°]**
Azi_1	**float**	**4**	**aszimuth angle of sample 1 [°]**
Status_1	**float**	**4**	**mitigation status flags of sample 1, 0/1: heater switch (ON/OFF) 0/10: blower switch (ON/OFF)**
TransPow_1	**float**	**4**	**transmitter power of sample 1 [W]**
TransT_1	**float**	**4**	**transmitter temp. of sample 1 [K]**
RecT_1	**float**	**4**	**receiver temp. of sample 1 [K]**
PCT_1	**float**	**4**	**PC temp. of sample 1 [K]**
Res_1[ ]	**float**	**3 x 4**	**reserved, sample 1**
TPr_1[ ]	**float**	**TAltN x 4**	**temp. profile, sample 1**
AHPr_1[ ]	**float**	**HAltN x 4**	**abs. hum. profile, sample 1**
RHPr_1[ ]	**float**	**HAltN x 4**	**rel. hum. profile, sample 1**
PNv_1[ ]	**float**	**RAltN x 4**	**total IF power in v-pol. measured at ADC input**
PNh_1[ ]	**float**	**RAltN x 4**	**total IF power in h-pol. measured at ADC input (only if DualPol>0)**
SLv_1[ ]	**float**	**RAltN x 4**	**linear sensitivity limit in Ze units for vertical polarisation**
SLh_1[ ]	**float**	**RAltN x 4**	**linear sensitivity limit in Ze units for horizontal polarisation (only if DualPol>0)**
PrMsk_1[ ]	**char**	**RAltN**	**mask array of occupied range cells of sample 1 0: range cell not occupied 1: range cell occupied**
The loop over all range bins of sample 1 (loop index n) starts here			
The following data is only stored if PrMsk_1[n]=1			
SpecBytes_1[n]	**int**	**4**	**number of bytes of following spectral block**
The following data is only stored if CompEna=0 (spectra contain noise floor)			
VSpec_1[n]	**float**	**4 x SpecN**	**full Doppler spectrum (incl. noise), vertical pol., linear Ze**
The following data is only stored if DualPol >0 (dual pol. radar)			
HSpec_1[n]	**float**	**4 x SpecN**	**full Doppler spectrum (incl. noise), horizontal pol., linear Ze**
ReVHSpec _1[n]	**float**	**4 x SpecN**	**full covariance spectrum , real part (see equ. 2.11.6), linear Ze**
ImVHSpec _1[n]	**float**	**4 x SpecN**	**full covariance spectrum , imaginary part (see equ. 2.11.6) linear Ze**
The following data is only stored if CompEna>0 (spectral compression enabled)			

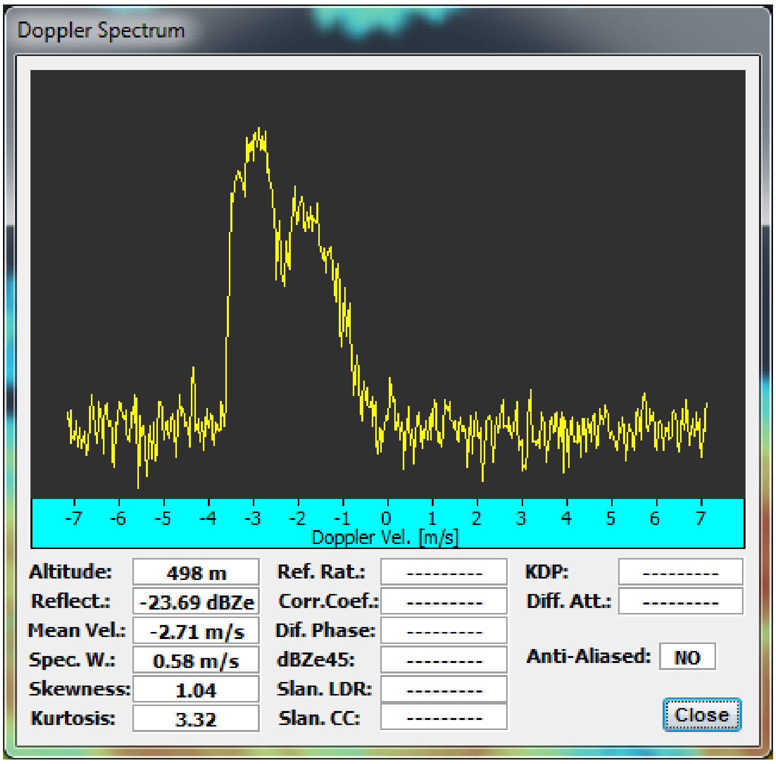
BlockN_1[n]	char	1	number of blocks in spectra
MinBkIdx_1[n]	short int	2 x BlockN_1[n]	minimum index of blocks in spectra
MaxBkIdx_1[n]	short int	2 x BlockN_1[n]	maximum index of blocks in spectra
VSpec_1[n]	float	4 x BlockN_1[n] x (MaxBkIdx_1[n] - MinBkIdx_1[n] +1)	compressed Doppler spectrum, vertical pol., linear Ze
The following data is only stored if DualPol >0 (dual pol. radar)			
HSpec_1[n]	float	4 x BlockN_1[n] x (MaxBkIdx_1[n] - MinBkIdx_1[n] +1)	compressed Doppler spectrum, horizontal pol., linear Ze
ReVHSpec_1[n]	float	4 x BlockN_1[n] x (MaxBkIdx_1[n] - MinBkIdx_1[n] +1)	compressed covariance spectrum , real part (see equ. 2.11.6), linear Ze
ImVHSpec_1[n]	float	4 x BlockN_1[n] x (MaxBkIdx_1[n] - MinBkIdx_1[n] +1)	compressed covariance spectrum , imaginary part (see equ. 2.11.6), linear Ze
The following data is only stored if CompEna=2 (include polar. spectral variables)			
RefRat_1[n]	float	4 x BlockN_1[n] x (MaxBkIdx_1[n] - MinBkIdx_1[n] +1)	compressed spectral differential reflectivity, see table 2.11.1 (product 1.), [dB]
CorrCoeff_1[n]	float	4 x BlockN_1[n] x (MaxBkIdx_1[n] - MinBkIdx_1[n] +1)	compressed spectral correlation coefficient, see table 2.11.1 (product 2.), [0,…,1]
DiffPh_1[n]	float	4 x BlockN_1[n] x (MaxBkIdx_1[n] - MinBkIdx_1[n] +1)	compressed spectral differential phase, see table 2.11.1 (product 3.), [rad]
The following data is only stored if DualPol =2 (dual pol. radar in STSR mode)			
SLDR_1[n]	float	4 x BlockN_1[n] x (MaxBkIdx_1[n] - MinBkIdx_1[n] +1)	compressed spectral slanted LDR, see table 2.11.1 (product 4.), [dB]
SCorrCoeff_1[n]	float	4 x BlockN_1[n] x (MaxBkIdx_1[n] - MinBkIdx_1[n] +1)	compressed spectral slanted correlation coefficient, see table 2.11.1 (product 5.), [0,…,1]
KDP_1[n]	float	4	specific differential phase shift, see equ. (2.10.13), [rad / km]
DiffAtt_1[n]	float	4	differential attenuation, see equ. (2.10.14), [dB / km]
The following data is only stored if CompEna>0 (spectral compression enabled)			
VNoisePow_1[n]	float	4	integrated Doppler spectrum noise power in v-pol., [Ze]
The following data is only stored if DualPol >0 (dual pol. radar)			
HNoisePow_1[n]	float	4	integrated Doppler spectrum noise power in h-pol., [Ze]
The following data is only stored if AntiAlias =1 and CompEna>0 (spectra are anti-aliased and compressed)			
AliasMsk_1[n]	char	1	mask indicating, if anti-aliasing has been applied (=1) or not (=0)

MinVel_1[n]	**float**	**4**	**minimum velocity in Doppler spectrum [m/s]**
The loop over all range bins of sample 1 ends here			
Sample 1 ends here			
…	**…**	**…**	**…**
Sample TotSamp starts here			
SampBytes _ TotSamp	**int**	**4**	**length of sample TotSamp [bytes], not including SampBytes _ TotSamp**
…	**…**	**…**	**…**

(1) The time is expressed in number of seconds since 1.1.2001, 00:00:00

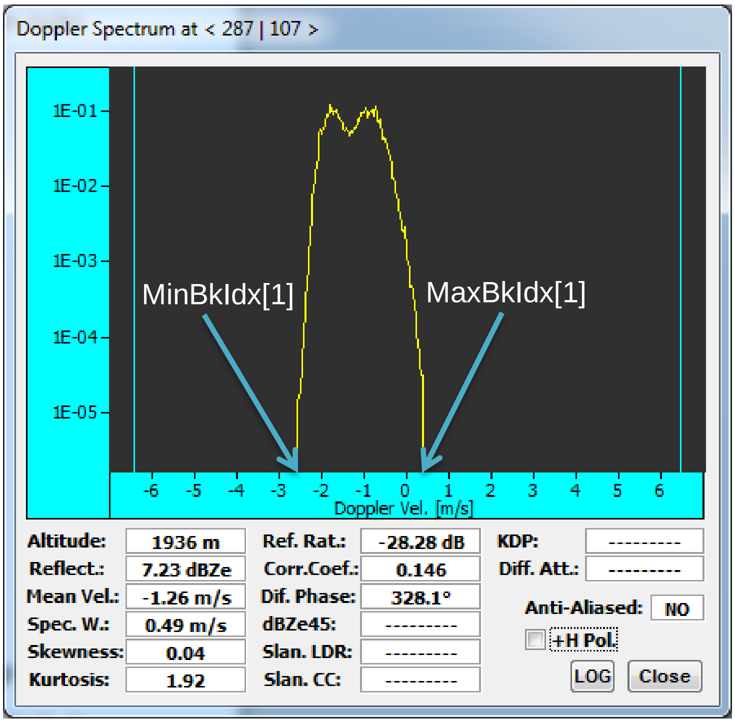
A2.1 Spectral Data Compression

In LV0 files the data compression of spectra can be enabled (CompEna>0). If CompEna=0, the spectral information is not noise stripped and not compressed (number of spectral samples = SpecN):



Uncompressed Doppler spectrum with unremoved noise level, shown in log scale

If the spectra are compressed (CompEna>0), the noise floor is removed and only the spectral information exceeding the noise underground is stored to the file. This reduces the file size extremely (by a factor of 5 – 10) and is also recommended, if the radar data has to be distributed over heavily loaded or slow networks. A typical compressed spectrum looks like this:



Compressed Doppler spectrum with removed noise underground, shown in log scale

The example spectrum above contains only a single block (BlockN =1). The block is specified by its minimum sample index (MinBkIdx[1]) and its maximum sample index (MaxBkIdx[1]) in the spectrum. With the spectral resolution

$$R_S = \frac{2\, MaxVel}{SpecN}$$

the velocity at an index k in the range [MinBkIdx[1],…, MaxBkIdx[1]] can be computed as:

$$V(k) = MinVel + k\, R_S$$

Of course a spectrum may have more than a single block (BlockN > 1). In this case MinBkIdx[] and MaxBkIdx[] are arrays of length BlockN and the minimum and maximum indices of block number n is given by MinBkIdx[n] and MaxBkIdx[n]. Each block contains exactly (MaxBkIdx[n] - MinBkIdx[n] + 1) spectral samples.

A2.2 LV0-Files (*.LV0) Version 3.5

(this file structure is used by RPG-FMCW-94-SX radars since July 2017)

In order to allow for a more detailed data analysis when transforming LV0-files to the highest data level 2 (cloud physical parameters), an extended LV0 header has been added to LV0 data files. This contains more details about chirp generator parameters for expert use that are usually stored in the chirp table. Because the chirp table can be modified by the user, it is more consistent to copy the parameters of a chirp program to the data file itself.

Variable Name	Type	# Bytes	Description
File Code	**int**	**4**	**LV0-File ID (=889346), Version 3.5, (=1889346 for consistency**

			checked LV0 file)
Header starts here			
HeaderLen	int	4	header length in bytes (not including HeaderLen)
StartTime[(1)]	unsig. int	4	time of first sample in file
StopTime[(1)]	unsig. int	4	time of last sample in file
CGProg	int	4	chirp generator program number
ModelNo	int	4	=0: 94 GHz single pol. radar =1: 94 GHz dual pol. radar =2: 35 GHz dual pol. radar =3: 35 GHz dual pol. radar
ProgName	char	Len(ProgName)+1	null terminated char string of chirp program name
CustName	char	Len(CustName)+1	null terminated char string of customer name
Freq	float	4	radar frequency [GHz]
AntSep	float	4	separation of both antenna axis (bistatic configuration), [m]
AntDia	float	4	antenna diameter, [m]
AntG	float	4	linear antenna gain
HPBW	float	4	antenna half power beam width [°]
Cr	float	4	radar constant, defined by equ. (2.1.5)
DualPol	char	1	=0: single pol. radar =1: dual pol. radar, LDR conf. =2: dual pol. radar, STSR mode (see section 2.11.3)
CompEna	char	1	spectral compression flag: 0: not compressed 1: spectra compressed 2: spectra compressed and spectral polarimetric variables are stored in the file
AntiAlias	char	1	0: Doppler spectra are not anti-aliased 1: Doppler spectra have been anti-aliased
SampDur	float	4	sample duration [sec]
GPSLat	float	4	GPS latitude
GPSLong	float	4	GPS longitude
CalInt	int	4	period for automatic zero calibrations in number of samples
RAltN	int	4	number of radar ranging layers
TAltN	int	4	number of temperature profile layers
HAltN	int	4	number of humidity profile layers layers

SequN	**int**	**4**	**number of chirp sequences**
RAlts[ ]	**float**	**4 x RAltN**	**ranging altitude layers**
TAlts[ ]	**float**	**4 x TAltN**	**temp. profile altitude layers (only if TAltN>0)**
HAlts[ ]	**float**	**4 x HAltN**	**hum. profile altitude layers (only if HAltN>0)**
Fr[ ]	**int**	**4 x RAltN**	**range factors (see equ. (2.5.6))**
SpecN[ ]	**int**	**4 x SequN**	**number of samples in Doppler spectra of each chirp sequence**
RngOffs[ ]	**int**	**4 x SequN**	**chirp sequence start index in altitude layer array**
ChirpReps [ ]	**int**	**4 x SequN**	**number chirps within a sequence**
SeqIntTime[ ]	**float**	**4 x SequN**	**effective sequence integration time [sec]**
dR[ ]	**float**	**4 x SequN**	**chirp sequence range resolution [m]**
MaxVel[ ]	**float**	**4 x SequN**	**max. Doppler velocity [m/s] for each chirp sequence (unambiguous)**
ChanBW [ ]	**float**	**4 x SequN**	**bandwidth of individual radar channel in the sequence [Hz]**
ChirpLowIF [ ]	**int**	**4 x SequN**	**lowest IF frequency in the sequence [Hz]**
ChirpHighIF [ ]	**int**	**4 x SequN**	**highest IF frequency in the sequence [Hz]**
RangeMin [ ]	**int**	**4 x SequN**	**minimum altitude (range) of the sequence [m]**
RangeMax [ ]	**int**	**4 x SequN**	**maximum altitude (range) of the sequence [m]**
ChirpFFTSize [ ]	**int**	**4 x SequN**	**ranging FFT size, must be power of 2**
ChirpInvSamples [ ]	**int**	**4 x SequN**	**number of invalid samples at beginning of chirp**
ChirpCenterFr [ ]	**float**	**4 x SequN**	**chirp centre frequency [MHz] at radar transmitter output**
ChirpBWFr [ ]	**float**	**4 x SequN**	**chirp bandwidth [MHz] at radar transmitter output**
FFTStartInd [ ]	**int**	**4 x SequN**	**start index of sequence in FFT array**
FFTStopInd [ ]	**int**	**4 x SequN**	**stop index of sequence in FFT array**
ChirpFFTNo [ ]	**int**	**4 x SequN**	**number of FFT range layers in one chirp (usually = 1)**
SampRate	**int**	**4**	**ADC sampling rate [Hz]**
MaxRange	**int**	**4**	**maximum unambiguous range [m]**
SupPowLev	**char**	**1**	**flag indicating, if power levelling has been used (0=yes, 1=no)**
SpkFilEna	**char**	**1**	**flag indicating, if spike/plankton**

			filter has been used (1=yes, 0=no)
PhaseCorr	**char**	**1**	**flag indicating, if phase correction (dual. pol. radars) has been used (1=yes, 0=no)**
RelPowCorr	**char**	**1**	**flag indicating, if relative power correction (dual. pol. radars) has been used (1=yes, 0=no)**
FFTWindow	**char**	**1**	**FFT window in use: 0 = SQUARE 1 = PARZEN 2 = BLACKMAN 3 = WELCH 4 = SLEPIAN2 5 = SLEPIAN3**
FFTInputRng	**unsigned short**	**2**	**ADC input voltage range (+/-) [mV]**
SWVersion	**unsigned short**	**2**	**Software version * 100**
NoiseFilt	**float**	**4**	**noise filter threshold factor (multiple of STD in Doppler spectra)**
InstCalPar	**int**	**4**	**Calibration period [sec]**
Reserved	**int**	**4 x 24**	**reserved 96 Bytes for future use**
Reserved	**unsign. int**	**4 x 5000**	**space for 5000 time stamps to single samples**
Reserved	**unsign. int**	**4 x 5000**	**space for 5000 file pointers to single samples**
Header ends here			
TotSamp	**int**	**4**	**total number of samples**
Sample 1 starts here			
SampBytes _1	**int**	**4**	**length of sample 1 [bytes], not including SampBytes _1**
Time_1(1)	**unsig. int**	**4**	**time of sample 1 [sec]**
MSec_1	**int**	**4**	**milliseconds of sample 1 [msec]**
QF_1	**char**	**1**	**quality flag of sample 1: Bit 1: ADC saturation Bit 2: spectral width too high Bit 3: no transm. power leveling**
RR_1	**float**	**4**	**rain rate of sample 1 [mm/h]**
RelHum_1	**float**	**4**	**rel. humidity of sample 1 [%]**
EnvTemp_1	**float**	**4**	**environm. Temp. of sample 1 [K]**
BaroP_1	**float**	**4**	**barometric press. of sample 1 [hPa]**
WS_1	**float**	**4**	**wind speed of sample 1 [km/h]**
WD_1	**float**	**4**	**wind direction of sample 1 [°]**
DDVolt_1	**float**	**4**	**direct detection channel voltage of sample 1 [V]**
DDTb_1	**float**	**4**	**direct detection brightness temp. of sample 1 [K]**

LWP_1	float	4	liquid water path of sample 1 $[g/m^2]$
PowIF_1	float	4	IF power at ADC of sample 1 [µW]
Elev_1	float	4	elevation angle of sample 1 [°]
Azi_1	float	4	aszimuth angle of sample 1 [°]
Status_1	float	4	mitigation status flags of sample 1, 0/1: heater switch (ON/OFF) 0/10: blower switch (ON/OFF)
TransPow_1	float	4	transmitter power of sample 1 [W]
TransT_1	float	4	transmitter temp. of sample 1 [K]
RecT_1	float	4	receiver temp. of sample 1 [K]
PCT_1	float	4	PC temp. of sample 1 [K]
SkyTB_1	float	4	Sky brightness temp. [K]: TB[v] + TB[h] / 10000
InclEl_1	float	4	Inclination angle of elevation zenith position [DEG]
InclElAx_1	float	4	Inclination angle of elevation axis (perpendicular to InclEl_1) [DEG]
TPr_1[ ]	float	TAltN x 4	temp. profile, sample 1
AHPr_1[ ]	float	HAltN x 4	abs. hum. profile, sample 1
RHPr_1[ ]	float	HAltN x 4	rel. hum. profile, sample 1
PNv_1[ ]	float	RAltN x 4	total IF power in v-pol. measured at ADC input
PNh_1[ ]	float	RAltN x 4	total IF power in h-pol. measured at ADC input (only if DualPol>0)
SLv_1[ ]	float	RAltN x 4	linear sensitivity limit in Ze units for vertical polarisation
SLh_1[ ]	float	RAltN x 4	linear sensitivity limit in Ze units for horizontal polarisation (only if DualPol>0)
PrMsk_1[ ]	char	RAltN	mask array of occupied range cells of sample 1 0: range cell not occupied 1: range cell occupied
The loop over all range bins of sample 1 (loop index n) starts here			
The following data is only stored if PrMsk_1[n]=1			
SpecBytes_1[n]	int	4	number of bytes of following spectral block
The following data is only stored if CompEna=0 (spectra contain noise floor)			
TotSpec_1[n] (2)	float	4 x SpecN	full Doppler spectrum (incl. noise), linear Ze
The following data is only stored if DualPol >0 (dual pol. radar)			
HSpec_1[n]	float	4 x SpecN	full Doppler spectrum (incl. noise), horizontal pol., linear Ze

ReVHSpec _1[n]	**float**	**4 x SpecN**	**full covariance spectrum , real part (see equ. 2.11.6), linear Ze**
ImVHSpec _1[n]	**float**	**4 x SpecN**	**full covariance spectrum , imaginary part (see equ. 2.11.6) linear Ze**
The following data is only stored if CompEna>0 (spectral compression enabled)			
BlockN_1[n]	**char**	**1**	**number of blocks in spectra**
MinBkIdx_1[n] [BlockN_1[n]]	**short int**	**2 x BlockN_1[n]**	**minimum index of blocks in spectra**
MaxBkIdx_1[n] [BlockN_1[n]]	**short int**	**2 x BlockN_1[n]**	**maximum index of blocks in spectra**
TotSpec_1[n][ ](2)	**float**	**4 x BlockN_1[n] x (MaxBkIdx_1[n][ ] - MinBkIdx_1[n][ ] +1)**	**compressed Doppler spectrum, linear Ze loop over BlockN_1[n] blocks**
The following data is only stored if DualPol >0 (dual pol. radar)			
HSpec_1[n][ ]	**float**	**4 x BlockN_1[n] x (MaxBkIdx_1[n][ ] - MinBkIdx_1[n][ ] +1)**	**compressed Doppler spectrum, horizontal pol., linear Ze loop over BlockN_1[n] blocks**
ReVHSpec_1[n][ ]	**float**	**4 x BlockN_1[n] x (MaxBkIdx_1[n][ ] - MinBkIdx_1[n][ ] +1)**	**compressed covariance spectrum , real part (see equ. 2.11.6), linear Ze loop over BlockN_1[n] blocks**
ImVHSpec_1[n][ ]	**float**	**4 x BlockN_1[n] x (MaxBkIdx_1[n][ ] - MinBkIdx_1[n][ ] +1)**	**compressed covariance spectrum , imaginary part (see equ. 2.11.6), linear Ze loop over BlockN_1[n] blocks**
The following data is only stored if CompEna=2 (include polar. spectral variables)			
RefRat_1[n][ ]	**float**	**4 x BlockN_1[n] x (MaxBkIdx_1[n][ ] - MinBkIdx_1[n][ ] +1)**	**compressed spectral differential reflectivity, see table 2.11.1 (product 1.), [dB] loop over BlockN_1[n] blocks**
CorrCoeff_1[n][ ]	**float**	**4 x BlockN_1[n] x (MaxBkIdx_1[n][ ] - MinBkIdx_1[n][ ] +1))**	**compressed spectral correlation coefficient, see table 2.11.1 (product 2.), [0,…,1] loop over BlockN_1[n] blocks**
DiffPh_1[n][ ]	**float**	**4 x BlockN_1[n] x (MaxBkIdx_1[n][ ] - MinBkIdx_1[n][ ] +1)**	**compressed spectral differential phase, see table 2.11.1 (product 3.), [rad] loop over BlockN_1[n] blocks**
The following data is only stored if DualPol =2 (dual pol. radar in STSR mode)			
SLDR_1[n][ ]	**float**	**4 x BlockN_1[n] x**	**compressed spectral slanted**

		(MaxBkIdx_1[n][ ] - MinBkIdx_1[n][ ] +1)	LDR, see table 2.11.1 (product 4.), [dB] loop over BlockN_1[n] blocks
SCorrCoeff_1[n][ ]	float	4 x BlockN_1[n] x (MaxBkIdx_1[n][ ] - MinBkIdx_1[n][ ] +1)	compressed spectral slanted correlation coefficient, see table 2.11.1 (product 5.), [0,…,1] loop over BlockN_1[n] blocks
KDP_1[n][ ]	float	4	specific differential phase shift, see equ. (2.10.13), [rad / km]
DiffAtt_1[n][ ]	float	4	differential attenuation, see equ. (2.10.14), [dB / km]
The following data is only stored if CompEna>0 (spectral compression enabled)			
TotNoisePow_1[n][(2)]	float	4	integrated Doppler spectrum noise power [Ze]
The following data is only stored if DualPol >0 (dual pol. radar)			
HNoisePow_1[n]	float	4	integrated Doppler spectrum noise power in h-pol., [Ze]
The following data is only stored if AntiAlias =1 and CompEna>0 (spectra are anti-aliased and compressed)			
AliasMsk_1[n]	char	1	mask indicating, if anti-aliasing has been applied (=1) or not (=0)
MinVel_1[n]	float	4	minimum velocity in Doppler spectrum [m/s]
The loop over all range bins of sample 1 ends here			
Sample 1 ends here			
…	…	…	…
Sample TotSamp starts here			
SampBytes _ TotSamp	int	4	length of sample TotSamp [bytes], not including SampBytes _ TotSamp
…	…	…	…

[(1)] The time is expressed in number of seconds since 1.1.2001, 00:00:00
[(2)] The total spectrum is identical to the vertical polarisation spectrum for single polarized or LDR-mode radars. In the case of dual polarized STSR-mode radars, this spectrum represents the combined spectrum given in equation (2.11.13).

A.3 LV1-Files (.LV1), Version 3.5*

(file structure used by RPG-FMCW-94-SX radars since June 2017)

Variable Name	Type	# Bytes	Description
File Code	int	4	LV1-File ID (=889347), Version 3.5, (=1889347 for consistency checked LV1 file)
Header starts here			
HeaderLen	int	4	header length in bytes (not

			including HeaderLen)
StartTime(1)	unsig. int	4	time of first sample in file
StopTime(1)	unsig. int	4	time of last sample in file
CGProg	int	4	chirp generator program number
ModelNo	int	4	=0: 94 GHz single pol. radar =1: 94 GHz dual pol. radar
ProgName	char	Len(ProgName)+1	null terminated char string of chirp program name
CustName	char	Len(CustName)+1	null terminated char string of customer name
Freq	float	4	radar frequency [GHz]
AntSep	float	4	separation of both antenna axis (bistatic configuration), [m]
AntDia	float	4	antenna diameter, [m]
AntG	float	4	linear antenna gain
HPBW	float	4	antenna half power beam width [°]
DualPol	char	1	=0: single pol. radar =1: dual pol. radar, LDR conf. =2: dual pol. radar, STSR mode (see section 2.11.3)
SampDur	float	4	sample duration [sec]
GPSLat	float	4	GPS latitude
GPSLong	float	4	GPS longitude
CalInt	int	4	period for automatic zero calibrations in number of samples
RAltN	int	4	number of radar ranging layers
TAltN	int	4	number of temperature profile layers
HAltN	int	4	number of humidity profile layers layers
SequN	int	4	number of chirp sequences
RAlts[ ]	float	4 x RAltN	ranging altitude layers
TAlts[ ]	float	4 x TAltN	temp. profile altitude layers (only if TAltN>0)
HAlts[ ]	float	4 x HAltN	hum. profile altitude layers (only if HAltN>0)
SpecN[ ]	int	4 x SequN	number of samples in Doppler spectra of each chirp sequence
RngOffs[ ]	int	4 x SequN	chirp sequences start index array in altitude layer array
SeqAvg[ ]	int	4 x SequN	number of averaged chirps within a sequence
SeqIntTime[ ]	float	4 x SequN	effective sequence integration time [sec]
dR[ ]	float	4 x SequN	range resolution array for chirp sequences
MaxVel[ ]	float	4 x SequN	max. Doppler velocity [m/s] for

			each chirp sequence (unambiguous)
SupPowLev	char	1	flag indicating, if power levelling has been used (0=yes, 1=no)
SpkFilEna	char	1	flag indicating, if spike/plankton filter has been used (1=yes, 0=no)
PhaseCorr	char	1	flag indicating, if phase correction (dual. pol. radars) has been used (1=yes, 0=no)
RelPowCorr	char	1	flag indicating, if relative power correction (dual. pol. radars) has been used (1=yes, 0=no)
FFTWindow	char	1	FFT window in use: 0 = SQUARE 1 = PARZEN 2 = BLACKMAN 3 = WELCH 4 = SLEPIAN2 5 = SLEPIAN3
FFTInputRng	unsigned short	2	ADC input voltage range (+/-) [mV]
SWVersion	unsigned short	2	Software version * 100
NoiseFilt	float	4	noise filter threshold factor (multiple of STD in Doppler spectra)
Header ends here			
TotSamp	int	4	total number of samples
Sample 1 starts here			
SampBytes _1	int	4	length of sample 1 [bytes], not including SampBytes _1
Time_1[(1)]	unsig. int	4	time of sample 1 [sec]
MSec_1	int	4	milliseconds of sample 1 [msec]
QF_1	char	1	quality flag of sample 1: Bit 1: ADC saturation Bit 2: spectral width too high Bit 3: no transm. power leveling
RR_1	float	4	rain rate of sample 1 [mm/h]
RelHum_1	float	4	rel. humidity of sample 1 [%]
EnvTemp_1	float	4	environm. Temp. of sample 1 [K]
BaroP_1	float	4	barometric press. of sample 1 [hPa]
WS_1	float	4	wind speed of sample 1 [km/h]
WD_1	float	4	wind direction of sample 1 [°]
DDVolt_1	float	4	direct detection channel voltage of sample 1 [V]
DDTb_1	float	4	direct detection brightness temp. of sample 1 [K]
LWP_1	float	4	liquid water path of sample 1

			[g/m²]
PowIF_1	float	4	IF power at ADC of sample 1 [µW]
Elev_1	float	4	elevation angle of sample 1 [°]
Azi_1	float	4	aszimuth angle of sample 1 [°]
Status_1	float	4	mitigation status flags of sample 1, 0/1: heater switch (OFF/ON) 0/10: blower switch (OFF/ON) 0/100: T-prof. from radiometer (NO/YES) 0/1000: H-prof. from radiometer (NO/YES)
TransPow_1	float	4	transmitter power of sample 1 [W]
TransT_1	float	4	transmitter temp. of sample 1 [K]
RecT_1	float	4	receiver temp. of sample 1 [K]
PCT_1	float	4	PC temp. of sample 1 [K]
SkyTB_1	float	4	Sky brightness temp. [K]: TB[v] + TB[h] / 10000
InclEl_1	float	4	Inclination angle of elevation zenith position [DEG]
InclElAx_1	float	4	Inclination angle of elevation axis (perpendicular to InclEl_1) [DEG]
TPr_1[ ]	float	TAltN x 4	temp. profile, sample 1
AHPr_1[ ]	float	HAltN x 4	abs. hum. profile, sample 1
RHPr_1[ ]	float	HAltN x 4	rel. hum. profile, sample 1
SLv_1[ ]	float	RAltN x 4	linear sensitivity limit in Ze units for vertical polarisation
SLh_1[ ]	float	RAltN x 4	linear sensitivity limit in Ze units for horizontal polarisation (only if DualPol>0)
PrMsk_1[ ]	char	RAltN	mask array of occupied range cells of sample 1 0: range cell not occupied 1: range cell occupied
The loop over all range bins of sample 1 (loop index n) starts here			
The following data is only stored if PrMsk_1[n]=1			
Ze_1[n]	float	4	linear reflectivity in Ze units for vert. pol. in range bin n of sample 1
MeanVel_1[n]	float	4	mean velocity [m/s] for vert. pol. in range bin n of sample 1
SpecWidth_1[n]	float	4	spectral width [m/s] for vert. pol. in range bin n of sample 1
Skewn_1[n]	float	4	spectral skewness for vert. pol. in range bin n of sample 1
Kurt_1[n]	float	4	spectral kurtosis for vert. pol. in

			range bin n of sample 1
The following data is only stored if DualPol >0 (dual pol. radar)			
RefRat_1[n]	float	4	Differential reflectivity in range bin n of sample 1, table 2.11.2 (product 1.), [dB]
CorrC_1[n]	float	4	Correlation coefficient in range bin n of sample 1, table 2.11.2 (product 2.), [0,…,1]
DiffPh_1[n]	float	4	differential phase in range bin n of sample 1, table 2.11.2 (product 3.), [rad]
The following data is only stored if DualPol =2 (dual pol. radar in STSR mode)			
Reserved	float	4	Reserved for future use
SLDR_1[n]	float	4	slanted LDR in range bin n of sample 1, table 2.11.2 (product 4.), [dB]
SCorrC_1[n]	float	4	slanted correlation coefficient in range bin n of sample 1, table 2.11.2 (product 5.), [0,…,1]
KDP_1[n]	float	4	specific differential phase shift in range bin n of sample 1, see equ. (2.10.13), [rad / km]
DiffAtt_1[n]	float	4	differential attenuation in range bin n of sample 1, see equ. (2.10.14), [dB / km]
The loop over all range bins of sample 1 ends here			
Sample 1 ends here			
…	…	…	…
Sample TotSamp starts here			
SampBytes _ TotSamp	int	4	length of sample TotSamp [bytes], not including SampBytes _ TotSamp
…	…	…	…

(1) The time is expressed in number of seconds since 1.1.2001, 00:00:00

A.3.1 LV1-Files (*.LV1), Version 4.0

(file structure used by RPG-FMCW-94-SX radars since June 2019)

Variable Name	Type	# Bytes	Description
File Code	int	4	LV1-File ID (=889348), Version 4.0, (=1889348 for consistency checked LV1 file)
Header starts here			
HeaderLen	int	4	header length in bytes (not including HeaderLen)
StartTime(1)	unsig. int	4	time of first sample in file

StopTime[(1)]	**unsig. int**	**4**	**time of last sample in file**
CGProg	**int**	**4**	**chirp generator program number**
ModelNo	**int**	**4**	**=0: 94 GHz single pol. radar =1: 94 GHz dual pol. radar**
ProgName	**char**	**Len(ProgName)+1**	**null terminated char string of chirp program name**
CustName	**char**	**Len(CustName)+1**	**null terminated char string of customer name**
Freq	**float**	**4**	**radar frequency [GHz]**
AntSep	**float**	**4**	**separation of both antenna axis (bistatic configuration), [m]**
AntDia	**float**	**4**	**antenna diameter, [m]**
AntG	**float**	**4**	**linear antenna gain**
HPBW	**float**	**4**	**antenna half power beam width [°]**
DualPol	**char**	**1**	**=0: single pol. radar =1: dual pol. radar, LDR conf. =2: dual pol. radar, STSR mode (see section 2.11.3)**
SampDur	**float**	**4**	**sample duration [sec]**
GPSLat	**float**	**4**	**GPS latitude**
GPSLong	**float**	**4**	**GPS longitude**
CalInt	**int**	**4**	**period for automatic zero calibrations in number of samples**
RAltN	**int**	**4**	**number of radar ranging layers**
TAltN	**int**	**4**	**number of temperature profile layers**
HAltN	**int**	**4**	**number of humidity profile layers layers**
SequN	**int**	**4**	**number of chirp sequences**
RAlts[ ]	**float**	**4 x RAltN**	**ranging altitude layers**
TAlts[ ]	**float**	**4 x TAltN**	**temp. profile altitude layers (only if TAltN>0)**
HAlts[ ]	**float**	**4 x HAltN**	**hum. profile altitude layers (only if HAltN>0)**
SpecN[ ]	**int**	**4 x SequN**	**number of samples in Doppler spectra of each chirp sequence**
RngOffs[ ]	**int**	**4 x SequN**	**chirp sequences start index array in altitude layer array**
SeqAvg[ ]	**int**	**4 x SequN**	**number of averaged chirps within a sequence**
SeqIntTime[ ]	**float**	**4 x SequN**	**effective sequence integration time [sec]**
dR[ ]	**float**	**4 x SequN**	**range resolution array for chirp sequences**
MaxVel[ ]	**float**	**4 x SequN**	**max. Doppler velocity [m/s] for each chirp sequence (unambiguous)**

SupPowLev	char	1	flag indicating, if power levelling has been used (0=yes, 1=no)
SpkFilEna	char	1	flag indicating, if spike/plankton filter has been used (1=yes, 0=no)
PhaseCorr	char	1	flag indicating, if phase correction (dual. pol. radars) has been used (1=yes, 0=no)
RelPowCorr	char	1	flag indicating, if relative power correction (dual. pol. radars) has been used (1=yes, 0=no)
FFTWindow	char	1	FFT window in use: 0 = SQUARE 1 = PARZEN 2 = BLACKMAN 3 = WELCH 4 = SLEPIAN2 5 = SLEPIAN3
FFTInputRng	unsigned short	2	ADC input voltage range (+/-) [mV]
SWVersion	unsigned short	2	Software version * 100
NoiseFilt	float	4	noise filter threshold factor (multiple of STD in Doppler spectra)
InstCalPar	int	4	Calibration period [sec]
Reserved	int	4 x 24	reserved 96 Bytes for future use
Reserved	unsign. int	4 x 5000	space for 5000 time stamps to single samples
Reserved	unsign. int	4 x 5000	space for 5000 file pointers to single samples
Header ends here			
TotSamp	int	4	total number of samples
Sample 1 starts here			
SampBytes _1	int	4	length of sample 1 [bytes], not including SampBytes _1
Time_1[(1)]	unsig. int	4	time of sample 1 [sec]
MSec_1	int	4	milliseconds of sample 1 [msec]
QF_1	char	1	quality flag of sample 1: Bit 1: ADC saturation Bit 2: spectral width too high Bit 3: no transm. power leveling
RR_1	float	4	rain rate of sample 1 [mm/h]
RelHum_1	float	4	rel. humidity of sample 1 [%]
EnvTemp_1	float	4	environm. Temp. of sample 1 [K]
BaroP_1	float	4	barometric press. of sample 1 [hPa]
WS_1	float	4	wind speed of sample 1 [km/h]
WD_1	float	4	wind direction of sample 1 [°]
DDVolt_1	float	4	direct detection channel voltage

			of sample 1 [V]
DDTb_1	float	4	direct detection brightness temp. of sample 1 [K]
LWP_1	float	4	liquid water path of sample 1 [g/m²]
PowIF_1	float	4	IF power at ADC of sample 1 [µW]
Elev_1	float	4	elevation angle of sample 1 [°]
Azi_1	float	4	aszimuth angle of sample 1 [°]
Status_1	float	4	mitigation status flags of sample 1, 0/1: heater switch (OFF/ON) 0/10: blower switch (OFF/ON) 0/100: T-prof. from radiometer (NO/YES) 0/1000: H-prof. from radiometer (NO/YES)
TransPow_1	float	4	transmitter power of sample 1 [W]
TransT_1	float	4	transmitter temp. of sample 1 [K]
RecT_1	float	4	receiver temp. of sample 1 [K]
PCT_1	float	4	PC temp. of sample 1 [K]
SkyTB_1	float	4	Sky brightness temp. [K]: TB[v] + TB[h] / 10000
InclEl_1	float	4	Inclination angle of elevation zenith position [DEG]
InclElAx_1	float	4	Inclination angle of elevation axis (perpendicular to InclEl_1) [DEG]
TPr_1[ ]	float	TAltN x 4	temp. profile, sample 1
AHPr_1[ ]	float	HAltN x 4	abs. hum. profile, sample 1
RHPr_1[ ]	float	HAltN x 4	rel. hum. profile, sample 1
SLv_1[ ]	float	RAltN x 4	linear sensitivity limit in Ze units for vertical polarisation
SLh_1[ ]	float	RAltN x 4	linear sensitivity limit in Ze units for horizontal polarisation (only if DualPol>0)
PrMsk_1[ ]	char	RAltN	mask array of occupied range cells of sample 1 0: range cell not occupied 1: range cell occupied
The loop over all range bins of sample 1 (loop index n) starts here			
The following data is only stored if PrMsk_1[n]=1			
Ze_1[n]	float	4	linear reflectivity in Ze units for vert. pol. in range bin n of sample 1
MeanVel_1[n]	float	4	mean velocity [m/s] for vert. pol. in range bin n of sample 1
SpecWidth_1[n]	float	4	spectral width [m/s] for vert. pol.

			in range bin n of sample 1
Skewn_1[n]	float	4	spectral skewness for vert. pol. in range bin n of sample 1
Kurt_1[n]	float	4	spectral kurtosis for vert. pol. in range bin n of sample 1
The following data is only stored if DualPol >0 (dual pol. radar)			
RefRat_1[n]	float	4	Differential reflectivity in range bin n of sample 1, table 2.11.2 (product 1.), [dB]
CorrC_1[n]	float	4	Correlation coefficient in range bin n of sample 1, table 2.11.2 (product 2.), [0,…,1]
DiffPh_1[n]	float	4	differential phase in range bin n of sample 1, table 2.11.2 (product 3.), [rad]
The following data is only stored if DualPol =2 (dual pol. radar in STSR mode)			
Reserved	float	4	Reserved for future use
SLDR_1[n]	float	4	slanted LDR in range bin n of sample 1, table 2.11.2 (product 4.), [dB]
SCorrC_1[n]	float	4	slanted correlation coefficient in range bin n of sample 1, table 2.11.2 (product 5.), [0,…,1]
KDP_1[n]	float	4	specific differential phase shift in range bin n of sample 1, see equ. (2.10.13), [rad / km]
DiffAtt_1[n]	float	4	differential attenuation in range bin n of sample 1, see equ. (2.10.14), [dB / km]
The loop over all range bins of sample 1 ends here			
Sample 1 ends here			
…	…	…	…
Sample TotSamp starts here			
SampBytes _ TotSamp	int	4	length of sample TotSamp [bytes], not including SampBytes _ TotSamp
…	…	…	…

(1) The time is expressed in number of seconds since 1.1.2001, 00:00:00

A4. Filename Conventions

With software version 2.0 or later the filenames are fulfilling the following naming conventions:

Un-concatenated filenames contain the exact date and time of the first data sample in the format

YYMMDD_HHmmSS

Where YY is the year -2000, MM is the month, DD is the day, HH is the hour, mm is the minute and SS is the second.

All data files are starting with header sections depending on the chirp program that is used during the MDF measurement. When data files shall be concatenated, the program number of the two files to be concatenated needs to be the same. Therefore, the program number is always a part of the filename:

YYMMDD_HHmmSS_PXX

XX is the chirp generator program number.

All measurements are classified by the scan type. The following classification codes are defined:

ZEN : zenith observation
CEL : constant elevation and azimuth, but elevation is not equal to 90°
RHI : RHI scan (elevation scan with constant azimuth)
RHW : RHI scan with azimuth angle defined by wind direction
PPI : PPI scan (azimuth scan with constant elevation)
MIX : scan with mixed elevation and azimuth variation, neither PPI nor RHI

The measurement classification code is also part of all filenames:

YYMMDD_HHmmSS_PXX_MCC

MCC is one of the measurement classification codes defined above.

Level 0 and level 1 data files are distinguished by their file extensions:

YYMMDD_HHmmSS_PXX_MCC.LV0 for level 0 data files and
YYMMDD_HHmmSS_PXX_MCC.LV1 for level 1 data files.

The last filename convention is related to ***UNLIMITED*** and ***LIMITED*** mode measurements (see section 3.13.1). In ***UNLIMITED*** mode, the filename remains unchanged and a new file is created every hour. In ***LIMITED*** mode, the standard filename is preceded by the base name BasName defined in the measurement's MDF (see section 3.13.1) and the file is only closed when the associated MDF has finished:

BasName _YYMMDD_HHmmSS_PXX_MCC.LV0 for level 0 data files and
BasName _YYMMDD_HHmmSS_PXX_MCC.LV1 for level 1 data files.

This helps to distinguish between different MDFs within a batch (MBF) execution where only ***LIMITED*** mode MDFs are allowed.

When data files are concatenated to daily files, the resulting daily filename is reduced:

BasName _YYMMDD _PXX_MCC.LV0 for level 0 data files (***LIMITED*** mode) and
BasName _YYMMDD _PXX_MCC.LV1 for level 1 data files (***LIMITED*** mode).

or

YYMMDD _PXX_MCC.LV0 for level 0 data files (***UNLIMITED*** mode) and
YYMMDD _PXX_MCC.LV1 for level 1 data files (***UNLIMITED*** mode).

A5. Calibration File Format (ABSCAL.CLB)

Variable Name	Type	# Bytes	Description
File Code	int	4	File ID (=4594960)
PolN	char	1	Number of polarisations
GIMShift	int	4	Ghost Image Frequency Shift [Hz] * 10
ChirpCenterFr	float	4	Chirp generator center frequency [MHz]
CalTime	int	4	Calibration time (number of seconds since 1.1.2001, 00:00:00)
CalType	int	4	Calibration type: 1: LN2 2: Sky
BaroPress	float	4	Barometric pressure [hPa]
THot	float	4	Hot Load temperature [K]
TCold	float	4	Cold Load temperature [K]
TDickeSW	float	4	Dicke Switch brightness temperature [K] (vertical channel)
TDickeSWRef	float	4	Dicke Switch reference temperature [K]
TsysDD	float	4	Direct detection receiver system temperature [K]
AbsCalCount	int	4	Number of calibrated IF channels
AbsCalFMin	float	4	Minimum frequency in IF band [Hz]
AbsCalDelF	float	4	Frequ. step in calibrated radar IF-band [Hz]
TsysRadv[ ]	float	4 x AbsCalCount	Tsys array of radar IF channels (vertical pol.)
TDickeSWRadv[ ]	float	4 x AbsCalCount	Dicke Switch TB array of radar IF channels (vertical pol.)
RXGv	double	8	Mean receiver gain (vertical pol.)
TsysRadh[ ]	float	4 x AbsCalCount	Tsys array of radar IF channels (horizontal pol.)
TDickeSWRadh[ ]	float	4 x AbsCalCount	Dicke Switch TB array of radar IF channels (horizontal pol.)
RXGh	double	8	Mean receiver gain (horizontal pol.)

A6. Measurement Definition Files (.MDF)*

Variable Name	Type	# Bytes	Description
File Code	int	4	File ID (=48856)
CGProgNo	int	4	Chirp generator program number 1000: Demo Mode
ZeroCalInt	int	4	Zero calibration interval [sec]
Lev0Ena	char	1	Level 0 data storage flag: 0: off 1: 0n
Lev1Ena	char	1	Level 1 data storage flag: 0: off 1: 0n
SpecCompEna	char	1	Spectral compression flag: 0: off 1: 0n
SpecParLV0Ena	char	1	Store spectral parameters in LV0 file: 0: off 1: 0n
FileBackupEna	char	1	File backup on radar PC: 0: off 1: 0n
AntiAliasEna	char	1	Anti-aliasing of Doppler spectra: 0: off 1: 0n
PowLevSupEna	char	1	Transmitter power levelling suppression: 0: off 1: 0n
NoiseThresh	float	4	Detection noise threshold [sigma]
ScanType	char	1	Scan type: 0: constant angles 1: general scan
If ScanType = 0 (constant angles)			
ConstEl	float	4	Constant elevation angle [DEG]
ConstAz	float	4	Constant azimuth angle [DEG]
If ScanType = 1 (general scan)			
ScanMode	char	1	Scan mode: 0: continuous 1: discrete, stop at each sample
ScanCnt	int	4	Number of scan entries
ScanStartEl[ ]	float	4 x ScanCnt	Start elevation angles [DEG]
ScanStoptEl[ ]	float	4 x ScanCnt	Stop elevation angles [DEG]
ScanIncEl[ ]	float	4 x ScanCnt	elevation increment angles [DEG]

ScanSpeedEl[ ]	float	4 x ScanCnt	elevation speed array [DEG / sec]
ScanStartAz[ ]	float	4 x ScanCnt	Start azimuth angles [DEG]
ScanStoptAz[ ]	float	4 x ScanCnt	Stop azimuth angles [DEG]
ScanIncAz[ ]	float	4 x ScanCnt	azimuth increment angles [DEG]
ScanSpeedAz[ ]	float	4 x ScanCnt	azimuth speed array [DEG / sec]
AzWindAlign[ ]	char	1 x ScanCnt	Azimuth wind alignment flag: 0: off 1: on
FrameCnt	int	4	Number of frame entries
FrameStartScn[ ]	int	4 x FrameCnt	Frame start scan number
FrameStopScn[ ]	int	4 x FrameCnt	Frame stop scan number
FrameRep[ ]	int	4 x FrameCnt	Frame repetition number
Timing	char	1	Timing flag: 0: limited measurement 1: unlimited measurement
If Timing = 0 (LIMITED measurement)			
Duration	int	4	Measurement duration [sec]
BaseNmLen	int	4	Length of base name string (including string termination 0)
If Timing = 1 (UNLIMITED measurement)			
FileLen	int	4	File length [hours]
MeasStart	char	1	Measurement start flag: 0: immediately 1: triggered
If MeasStart = 1 (triggered measurement)			
StartTime	int	4	Start time [seconds since 1.1.2001, 00:00:00)
TrigCond	int	4	Trigger condition: 1: ignore date 2: ignore hour Others: raster interval [sec]
WindFunc	int	4	FFT windowing function: 0: rectangle 1: Parzen 2: Blackman 3: Welch 4: Slepian2 5: Slepian3
ADCVoltRng	int	4	ADC voltage range [volts *1000]

A7. Measurement Batch Files (.MBF)*

Variable Name	Type	# Bytes	Description

File Code	int	4	File ID (=23988557)
EntryCnt	int	4	Number of entries (MDFs)
Loop over EntryCnt MDFs (index n)			
MDFFnLen[n]	int	4	Length of null terminated MDF string (full path)
MDFFn[n][ ]	char	MDFFnLen [n]	null terminated MDF string (full path)
MDFLen[n]	int	4	Length of null terminated MDF string
MDF[n][ ]	char	MDFLen [n]	null terminated MDF string
BatRep	int	4	Batch repetition factor

A8. Solar Scan Files (.SSF)*

Variable Name	Type	# Bytes	Description
Scan Mode	char	0	Scanning mode: 0: azimuth scan 1: elevation scan
ScanCnt	int	4	Number of scan points
ScanWidth	float	4	Half scan width [DEG]
SunAz	float	4	Solar azimuth (of last point)
SunEl	float	4	Solar elevation (of last point)
TBv[ ]	double	8 x ScanCnt	Vetical polarisation brightness temp. array
TBh[ ]	double	8 x ScanCnt	Horizontal polarisation brightness temp. array
InclSlave	char	1	Slave data: 0: not included 1: included
If InclSlave = 1 (Slave data included)			
TBv[ ]	double	8 x ScanCnt	Vetical polarisation brightness temp. array of Slave radar
TBh[ ]	double	8 x ScanCnt	Horizontal polarisation brightness temp. array of Slave radar

A9. Chirp Table Files (CHIRP.TBL)

Variable Name	Type	# Bytes	Description
File Code	int	4	File ID (=12345679)
TblProgCnt	int	4	Number of programs in table
Data of Program 1			
ProgNm[ ]	char	15	Null terminated string of program name
CGProgNo	int	4	Chirp generator program number

DDSMult	int	4	CG DDS multiplier
ExtFundFreq	int	4	Fundamental frequency (crystal oscillator) [Hz]
SeqCnt	int	4	Number of chirp sequences
TotSamples	int	4	Total number of IF voltage samples
TotFFTs	int	4	Total number of FFTs to process the program
AltCnt	int	4	Number of altitude layers
IntUse	int	4	reserved
MaxBufCnt	int	4	Maximum number of FIFO transfer buffers
ProgDuration	float	4	Program duration [sec]
ChirpReps[ ]	int	4 x SeqCnt	Number of chirp repetitions in sequence
RangeMin[ ]	int	4 x SeqCnt	Minimum range in sequence [m]
RangeMax[ ]	int	4 x SeqCnt	Maximum range in sequence [m]
RangeRes[ ]	float	4 x SeqCnt	Range resolution [m]
ChirpCenterFr[ ]	float	4 x SeqCnt	Chirp center frequency at CG output [MHz]
ChirpBWFr[ ]	float	4 x SeqCnt	Chirp bandwidth at CG output [MHz]
ChirpLowIF[ ]	int	4 x SeqCnt	Chirp low IF frequency limit [Hz]
ChirpHighIF[ ]	int	4 x SeqCnt	Chirp high IF frequency limit [Hz]
TotSequSamp[ ]	int	4 x SeqCnt	Total number of IF voltage samples in sequence
SequenceTime[ ]	float	4 x SeqCnt	Sequence duration [sec]
InvalSamples[ ]	int	4 x SeqCnt	Number of invalid voltage samples in a chirp
ChirpFFTSize[ ]	int	4 x SeqCnt	Number of voltage samples in an FFT
ChirpFFTCnt[ ]	int	4 x SeqCnt	Number of FFTs in a chirp
RGLowLimit[ ]	int	4 x SeqCnt	DDS ramp generator low limit
RGHighLimit[ ]	int	4 x SeqCnt	DDS ramp generator high limit
RGStepSize[ ]	int	4 x SeqCnt	DDS ramp generator step size
RGStepRate[ ]	int	4 x SeqCnt	DDS ramp generator step rate (usually = 1)
RangeOffset[ ]	int	4 x SeqCnt	offset index of chirp sequence in altitude array
FFTStartInd[ ]	int	4 x SeqCnt	start index of sequence range in FFT
FFTStopInd[ ]	int	4 x SeqCnt	stop index of sequence range in FFT
AltLevels[ ]	float	4 x AltCnt	Altitude layers [m]
ModelNo	char	1	Radar model number: 0: RPG-FMCW94-SP 1: RPG-FMCW94-DP

			2: RPG-FMCW35-SP 3: RPG-FMCW35-DP
MaxRng	**int**	**4**	**Maximum unambiguous range [m]**
DopplerFFTSize[ ]	**int**	**4 x SeqCnt**	**Doppler spectrum FFT size**
DopplerRes[ ]	**float**	**4 x SeqCnt**	**Doppler spectral resolution [m/s]**
DopplerMax[ ]	**float**	**4 x SeqCnt**	**Doppler Niquist limit [m/s]**
PLLMult	**int**	**4**	**ADC board PLL multiplier**
GIMShift	**int**	**4**	**ghost image mitigation frequ. shift [*10 MHz]. If zero, GIM is disabled**
Reserved[ ]	**char**	**32**	**Reserved space for later use disabled**
Data of Program 2			
…	**…**	**…**	**…**
Data of Program TblProgCnt			
…	**…**	**…**	**…**

A10. I/Q Data Files (.IQ)*

Variable Name	Type	# Bytes	Description
File Code	**int**	**4**	**File ID (=889349)**
SampCnt	**int**	**4**	**Number of samples in file**
PolN	**byte**	**1**	**Number of polarisations**
The following data is only valid, if PolN=2			
DualPol	**byte**	**1**	**=0: single pol. radar =1: dual pol. radar, LDR conf. =2: dual pol. radar, STSR mode (see section 2.11.3)**
PhaseCorr	**byte**	**1**	**Dual pol. phase correction flag**
AmpCorr	**byte**	**1**	**Dual pol. amp. Correction flag**
FFTWindow	**byte**	**1**	**FFT window in use: 0 = SQUARE 1 = PARZEN 2 = BLACKMAN 3 = WELCH 4 = SLEPIAN2 5 = SLEPIAN3**
SequN	**int**	**4**	**Number of sequences**
ChirpReps[ ]	**int**	**4 x SequN**	**Number of chirp repetitions in sequence**
FFTStartInd[ ]	**int**	**4 x SequN**	**start index of sequence range in FFT**
FFTStopInd[ ]	**int**	**4 x SequN**	**stop index of sequence range in FFT**

RangeMin[ ]	**int**	**4 x SequN**	**Minimum range in sequence [m]**
RangeMax[ ]	**int**	**4 x SequN**	**Maximum range in sequence [m]**
RangeRes[ ]	**float**	**4 x SequN**	**Range resolution [m]**
ChirpFFTSize[ ]	**int**	**4 x SequN**	**Number of voltage samples in an FFT**
Loop over SampCnt samples (index n)			
Loop over polarisations (index p)			
Loop over chirp sequences (index s)			
Loop over chirp repetitions (index r)			
Loop over range bins (index b): [FFTStartInd[s],…,FFTStopInd[s]]			
IQr[n][p][s][r][b]	**float**	**4**	**real part**
IQi[n][p][s][r][b]	**float**	**4**	**imaginary part**
total of (FFTStopInd[s]- FFTStartInd[s]+1) complex numbers			