RFbeam Microwave GmbH

data sheet

K-MD2

Engineering Sample

radar transceiver
with integrated signal processing

Features

- 24 GHz FMCW radar with digital signal processing
- Angle of arrival in azimuth/elevation
- Serial target list output
- Detection distance: 100 m for persons/200 m for cars
- Distance range: 0–250 m, 1 m resolution
- Speed range: ± 130 km/h, 1 km/h resolution
- Angle range: ± 9.1° (elevation) ± 16.4° (azimuth), 0.1° resolution
- Compact size: 120 × 72 × 15 mm

Applications

- Traffic analysis and classification
- Intersection management
- Security systems
- Object speed measurement systems
- Measurement and research applications
- Industrial sensors

Description

The K-MD2 is a high-end 3D radar transceiver with three receiving channels and a low phase noise, PLL controlled transmitter. The target information from the three receive antennas is digitized and the high speed digital signal processing performs range and doppler FFT's with an update rate of 20 measurements per second. Using the serial interface, many operating parameters such as frequency, bandwidth and repetition rate can be adjusted. Results are available in target list format as well as in raw range-doppler matrices. Ethernet and a serial communication interfaces are included.

Blockdiagram

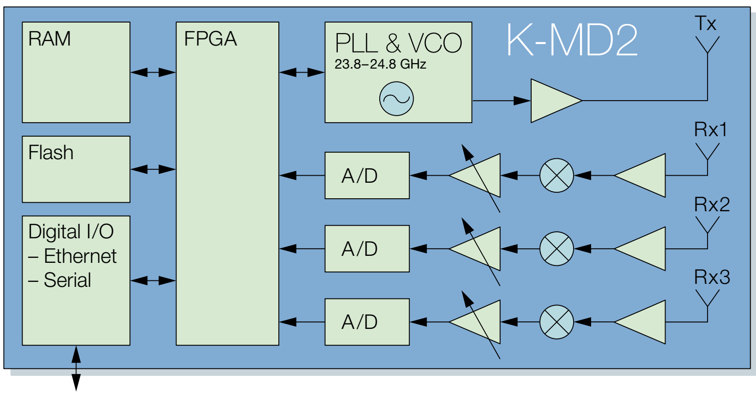


Figure 1: Blockdiagram

Characteristics

Parameter	Conditions/Notes	Symbol	Min	Typ	Max	Unit
Operating conditions						
Supply voltage		V_{cc}	10	12	16	V
Supply current		I_{cc}		600		mA
Connector type				JST S8B-PH-SM4-TB		
Operating temperature		T_{op}	-20		+85	°C
Storage temperature		T_{st}	-20		+105	°C
Transmitter						
Transmitter frequency		f_{TX}	23.800		24.800	GHz
Output power	EIRP	P_{TX}		+19		dBm
Output power deviation	f_{TX} = 24.000 .. 24.250 GHz	ΔP_{TX}		±1		dB
Maximum frequency error	V_{cc} = 12 V, -20 °C .. +85 °C	Δf_{Error}		50		ppm
Phase noise	@100 kHz	P_N			-90	dBc
Spurious Emissions	According to ETSI 300 440	P_{Sport}			-30	dBm
Antenna						
TX antenna gain	F_{TX} = 24.125 GHz	G_{Ant}		15		dBi
Polarisation				Vertical		
Horizontal –3dB beamwidth	E-Plane	W_{φ}		30		°
Vertical –3dB beamwidth	H-Plane	W_{θ}		20		°
RX 1,2,3 antenna gain		G_{Ant}		15		dBi
Horizontal –3dB beamwidth	E-Plane	W_{φ}		30		°
Vertical –3dB beamwidth	H-Plane	W_{θ}		20		°
Reciever						
Receiver sensitivity		P_{RX}		-132		dBm
Overall sensitivity	σ = 1 m², r = 100 m, S/N = 6 dB	S		-151		dBc
Signal Processing						
Modulation				FMCW		
Range processing				256 point FFT		
Velocity processing				256 point FFT		
Update rate				50		ms
Output						
Serial output			3.3 V Level, 115200 Bit/s			
Ethernet output	RJ-45 Connector		10		1000	MBit/s
Body						
Outline dimensions				120 × 72 × 15		mm
Weight				165		g

TABLE OF CONTENTS

ANTENNA DIAGRAM CHARACTERISTICS

This diagram shows the relative output power in both azimuth and elevation directions.

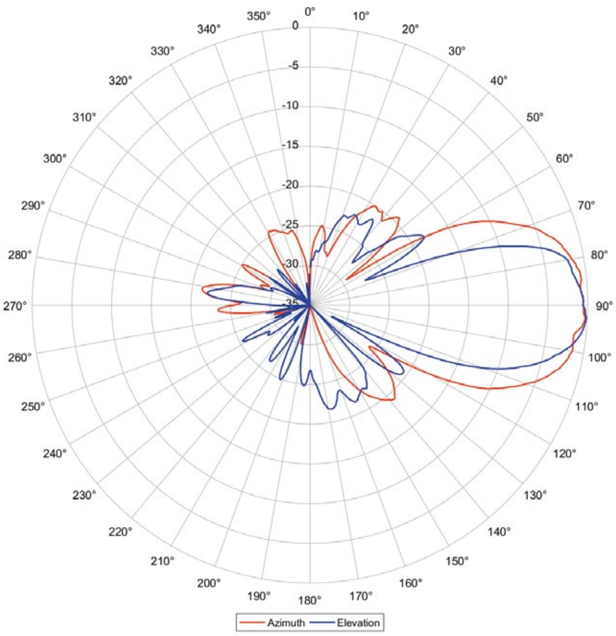


Figure 2: TX relative antenna pattern

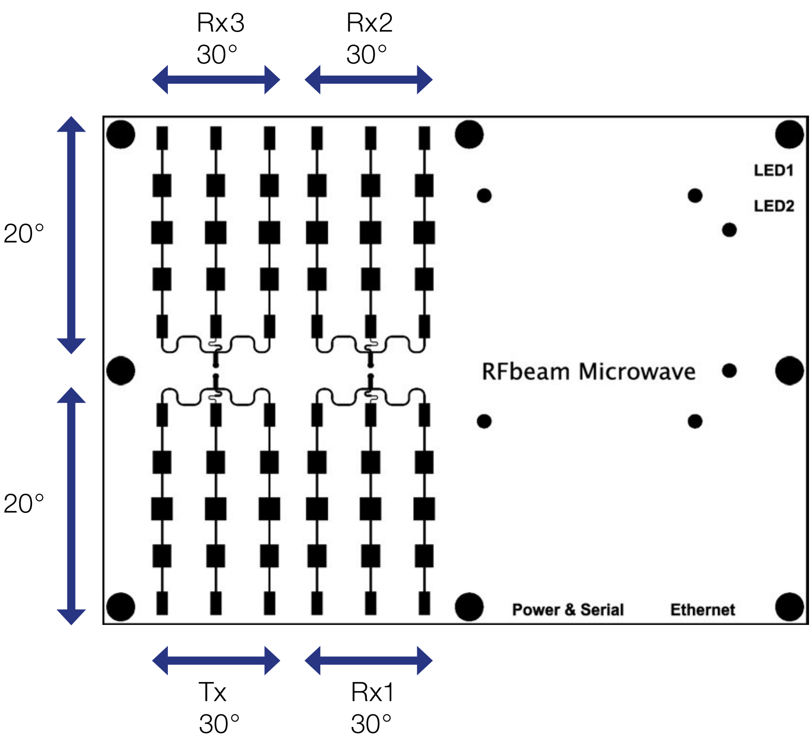


Figure 3: Beamwidth and antenna order

PIN CONFIGURATION AND FUNCTIONS

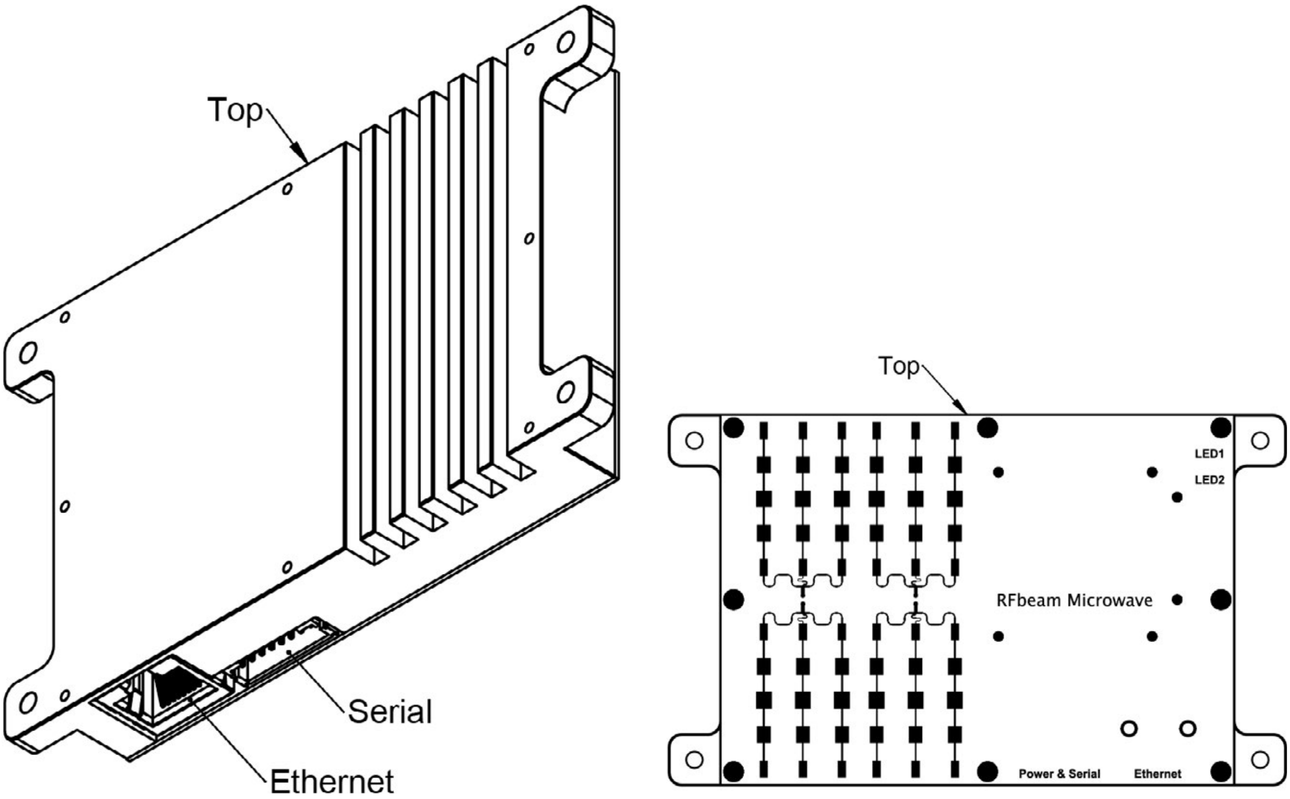


Figure 4: Direction and connectors

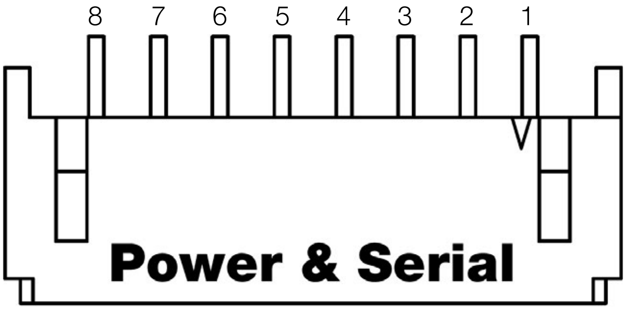


Figure 5: Power and serial connector

Pin 1: VCC +12 V/0.8 A
Pin 2: VCC +12 V/0.8 A
Pin 3: VCC +12 V/0.8 A
Pin 4: Serial TX, 3.3 V Level
Pin 5: Serial RX, 3.3 V Level
Pin 6: GND
Pin 7: GND
Pin 8: GND

THEORY OF OPERATION

The K-MD2 is configured to perform Frequency Modulated Continuous Wave (FMCW). With this modulation the radar receives information about static and moving objects.

In a typical FMCW radar system, the carrier frequency is modulated with a digitally generated linear ramp. At each step in the ramp, the received signal is demodulated and the phase is sampled.

Ramp

The K-MD2 is configured to use a saw tooth modulation for the ramp. It samples 260 points and ignores the first 4 samples. This means only the last 256 samples are used for processing. Ignoring the first four samples will be done to eliminate the discontinuity at start of the ramp.

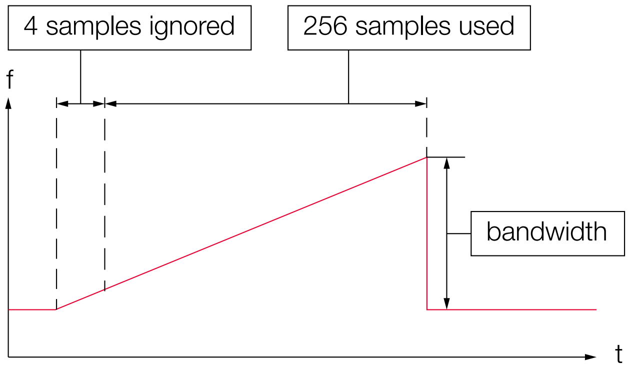


Figure 6: Ramp generation and sampling

Frame

The K-MD2 is configured to sample 257 chirps per frame but disregards the first chirp and uses only 256 chirps. Ignoring the first chirp is necessary to exclude disturbances in the first chirp which are caused by jumping to the start frequency.

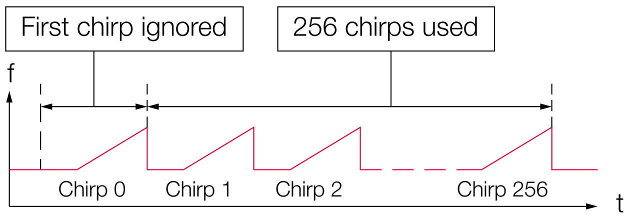
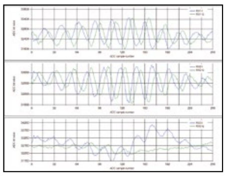


Figure 7: Frame generation

Processing

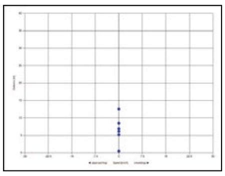


Raw ADC Data (RADC)

- Samples ADC data from all 3 RX antennas
  (1 Frame = 256 Samples x 256 Chirps)

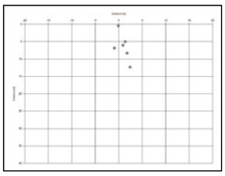
Mean Range-Doppler (RMRD)

- Calculates the range-doppler map for all 3 RX antennas
- Averages the range-doppler map of all 3 RX antennas



Raw Target (PDAT)

- Find targets over threshold
- Range compensation
- Filtering with an exponential moving average



Tracking Target (TDAT)

- Uses an alpha-beta-gamma tracker to track targets
- Assigns objects to tracking channels
- Predicts temporary lost objects

Figure 8: Signal processing workflow

APPLICATION INFORMATION

The K-MD2 controls the Radar Front End (RFE) and receives demodulated signals. It processes the signals to detect objects, sending the results to the main system using a serial or ethernet interface. There is a powerful GUI available to visualize the signals and target lists and change the settings.

> Even though the K-MD2 is able to drive a bandwidth of 1 GHz in most countries the K-Band is limited to a bandwidth of 250 MHz.

Speed and Range Settings

There are several predefined range and speed settings selectable. However there is also the possibility to change the range and speed settings out of the presets.

Range

$$\Delta R = 150\,MHz \cdot \left(\frac{N+M}{N \cdot B_w}\right)$$

$$R_{max} = (N-1) \cdot \Delta R$$

ΔR = Range resolution [m]
N = Used samples
M = Ignored samples
B_w = Bandwidth [MHz]
R_{max} = Maximal range [m]

Max. Range [m]	Resolution [m]	Ramp bandwidth [MHz]	Sampling bandwidth [MHz]	Start frequency [MHz]
40.0491302	0.15705541	970	955.0769231	23800
70.1221232	0.27498872	554	545.4769231	23848
100.122825	0.39263853	388	382.0307692	23931
150.572311	0.59047965	258	254.0307692	23996
200.245651	0.78527706	194	191.0153846	24028
249.023438	0.9765625	156	153.6	24047

Table 1: Range presets

Speed

$$\Delta V = \frac{\lambda \cdot clk \cdot 3.6}{2 \cdot N \cdot (S_{clk} \cdot (N+M) + delay)}$$

$$V_{max} = \Delta V \cdot \left(\frac{N}{2} - 1\right)$$

ΔV = Speed resolution [km/h]
λ = Wavelength = 0.012426 m
clk = Core clock = 38461538 Hz
S_{clk} = Core clocks per sample = 12
N = Used samples
M = Ignored samples
$delay$ = Initial delay of chirp [clk]
V_{max} = Maximum speed [km/h]

Max. speed [km/h]	Resolution [km/h]	Initial delay [clk]
30.0008448	0.23622712	11106
50.004923	0.39373955	5415
80.0135017	0.63002757	2214
100.021565	0.78757138	1147
120.020253	0.94504136	436

Table 2: Speed presets

Target Detection

With a set of parameters the number of raw targets can be influenced. It can be decided how a peak over threshold is reported. Usually an object results not only as one peak in the range-doppler map but rather as some peaks side by side.

Background Update (PSBU)

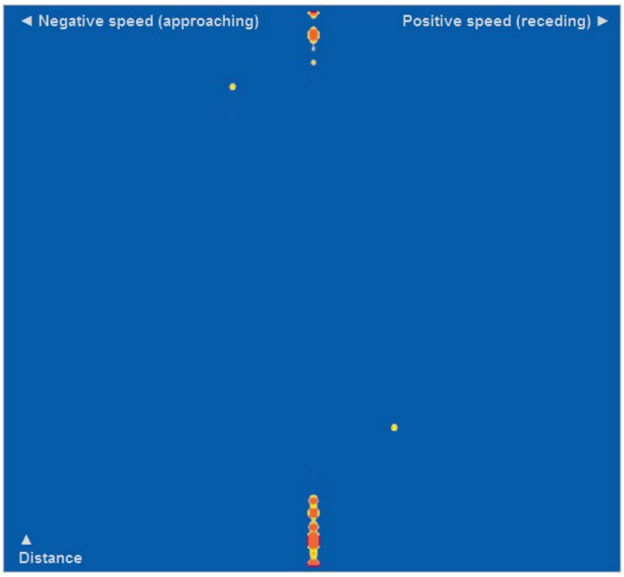
The running moving average is maintained for the mean range-doppler map, which provides an exponential moving average (EMA) against which each measurement is compared. The coefficient for the exponential decay is set using the PSBU command.

When debugging the performance, it is useful to set the 'Background update' to zero so that no clutter or interference is removed. It should be reset to a typical value of 128 for normal operation. The value sets the coefficient for the exponential moving average against which new signals are compared. Higher values cause the average to update more slowly so that new signals are reported for longer.

The following formula shows how the exponential moving average filter is built:

$$S(t) = \alpha \cdot Y(t) + (1-\alpha) \cdot S(t-1)$$

$S(t)$ = *Value of the EMA at any time period t*
$Y(t)$ = *Value at a time period t*
α = *1/(background update)*



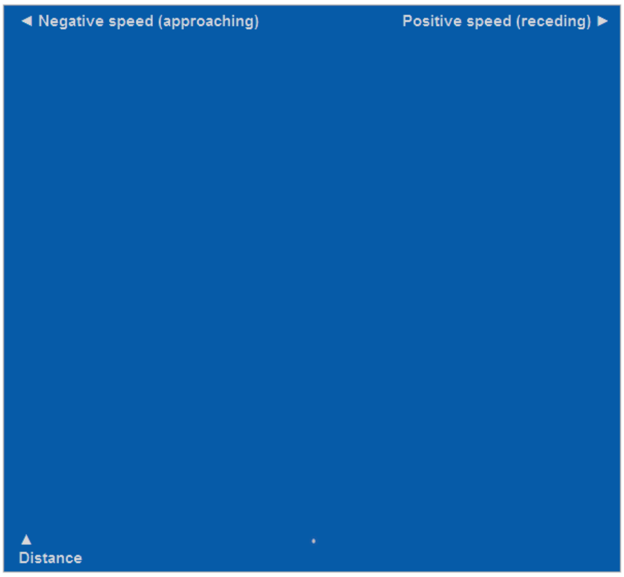


Figure 9: Background update = 0 on the left and background update = 128 on the right

Peak Threshold (PSPT)

If a value in the range-doppler map is higher than its neighbouring values in range and doppler speed (it is a local peak) and it is more than a specified amount higher than the exponential moving average (it is a new signal), it is detected as a peak. The required increase in signal strength is set using the PSPT command.

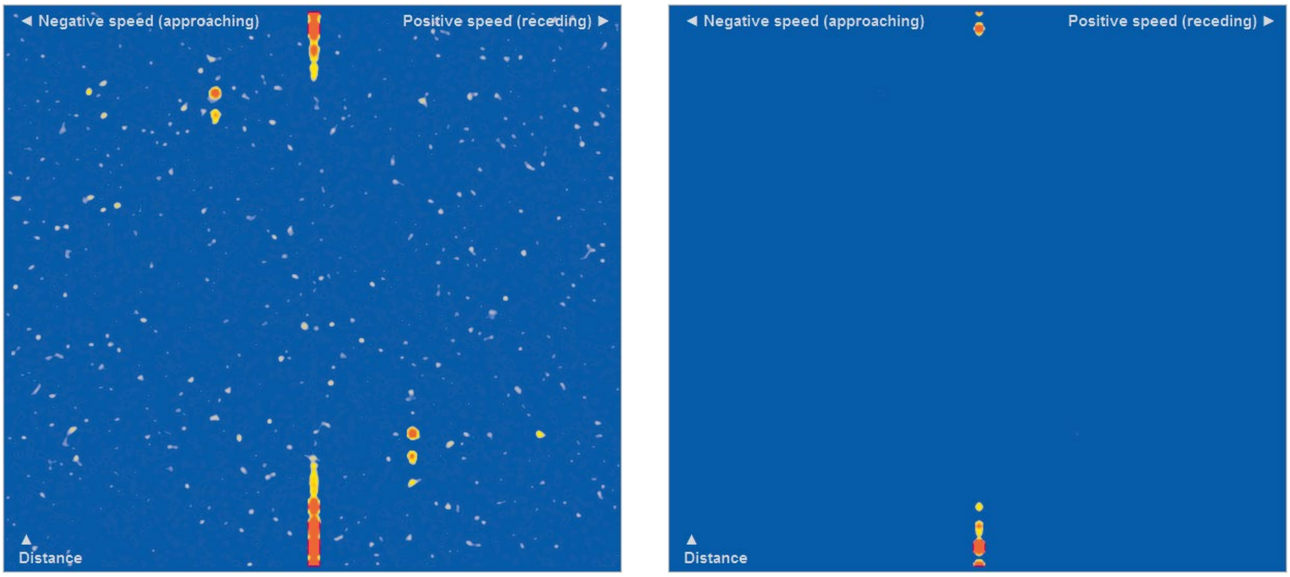


Figure 10: Peak threshold = 400 on the left and threshold = 4000 on the right

Range Compensation (PSRC)

The peak threshold can be adjusted to allow for the expected change in signal with range. The adjustment is made by dividing the threshold by the range to the power of the compensation value. 0.0 results in no adjustment. 1.0 is a linear decrease in threshold with range etc. The threshold at the centre range is not adjusted, so the threshold is increased for shorter ranges and decreased for longer ranges.

$$T_{cmp} = \frac{T_{peak}}{R_{bin}^{\,R_{cmp}}}$$

T_{cmp} = *Compensated threshold*
T_{peak} = *Peak threshold*
R_{bin} = *Range bin*
R_{cmp} = *Range compensation*

Smoothing (PSSM)

The mean range-doppler map can be smoothed with a 3x3 Gaussian window.

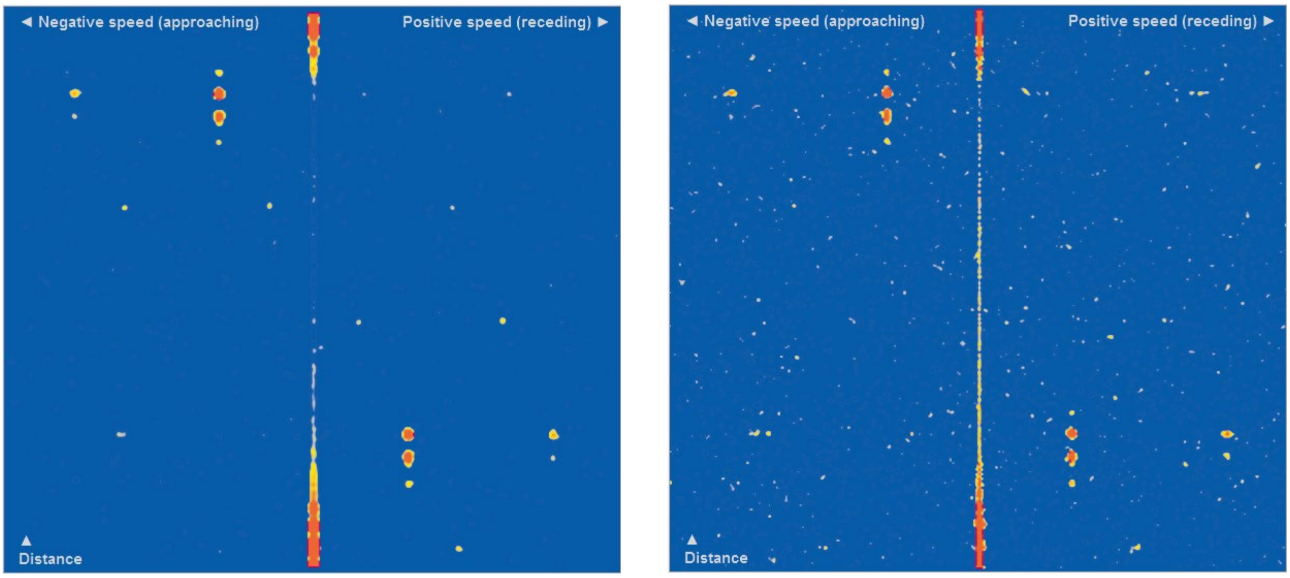


Figure 11: “Smoothing = On” on the left side and “Smoothing = Off” on the right side

Tracking

With a set of parameters the number of tracked targets can be influenced. On each frame the range, speed and direction of every track is updated. An alpha-beta-gamma tracker is used, whereby the speed is updated with the current acceleration estimate, and the range is updated with the current speed estimate.

Track Life (PSBL, PSTL)

The list of detected peaks is compared with the updated list of known tracks. If a peak matches an existing track, it is marked as associated and the life of the track is increased. If no peak is found for the track, the life of the track is decreased. If there is no existing track for the peak a new track is created. Once the life of a track reaches a minimum threshold it is reported as a confirmed track. The life of a track is limited, in that it will be lost if no detections are associated with it for the maximum track life. Note that the track will no longer be reported once the track life reaches the minimum threshold.

Range and Speed Jitter (PSRJ, PSSJ)

Every new detected target is compared to the known tracks. If the range and the speed of the new target are within the jitter values around the track's speed and range the target is assigned to the track and the track is updated.

Assume Constant Speed (PSCS)

When assuming constant speed is on, the tracking algorithm allows only small changes in speed between each measurement. This is useful when objects are moving slowly or are moving towards or away from the radar at high speed (e.g. looking along a road). It can result in tracks being lost if the speed changes quickly.

When assuming constant speed is off, the tracking algorithm allows large changes in speed. This is useful when objects are moving across the field of view such that their speed towards the radar changes rapidly as they pass the radar (e.g. looking across a road). It can result in tracks being split if the speed information is noisy.

Direction Threshold

The '`direction_error_threshold`' is used when comparing new detections to existing tracks. The track will follow the detection that most closely matches the predicted range, speed and direction of the track. If all of the detections have a difference in direction greater than the '`direction_error_threshold`', the strongest detection with a matching range and speed is used.

The value should be set to zero if no multiple object with a similar range and speed are expected, so the strongest signal (within the jitter thresholds) is always tracked.

If multiple objects with the same range and speed are expected (e.g. when looking along a multi-lane road) the value can be:

- set small (e.g. 5 degrees) so that the direction is only used if it matches the track reasonably well, otherwise the strongest signal is used; or
- set high (e.g. 45 degrees) so that the detection with the closest match in direction is always used (even if there is a big difference).

> Using smaller values can result in tracks being merged. Using larger values can result in tracks being split.

Interpretation Example

The direct output of the radar processing is the mean range-doppler map. It is the result of 2D FFT over all ranges and all speeds.

The following example demonstrates a road situation. In the following figure, the range-doppler map on the left side shows the speed and the distance of the objects. Zero speed is placed in the middle of the x-axis. The distance is increasing on the y-axis. Negative speeds (approaching objects) are shown to the left of zero speed and positive speeds (receding objects) are shown to the right of zero speed.

The blue and the black cars are driving away from the radar and the white car is approaching. The blue car is driving twice as fast as the black car.

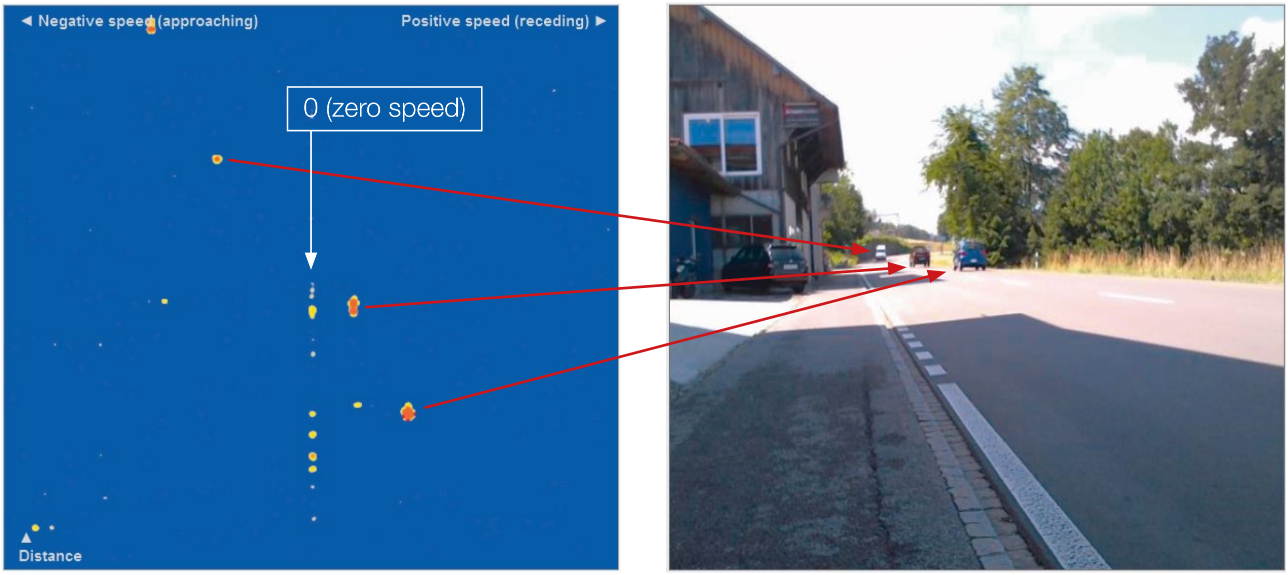


Figure 12: Interpretation example of a road situation

RADAR SYSTEM

For the processing of the digital signals, the radar uses a Zynq XC7Z020 system on a chip (SoC). The SoC is mounted on a TE720 adapter board from Trenz electronic. It mainly consists out of a processing system (PS) part, which contains two Cortex-A9 processors, and the programmable logic (PL) part, which contains the FPGA part. A Linux operating system is running on the processing system.

Start-Up

The start-up script `/etc/init.d/rcS` runs automatically when the radar is powered. This script performs the following steps:

1. Mounts the file systems in `/etc/fstab`
2. Look for `/mnt/rcS` and refer to this script if it is present
3. Start the Telnet daemon
4. Start the FTP daemon
5. Start the SSH daemon
6. Run the file `/mnt/init.sh` if it is present, otherwise
7. Run the file `/opt/init.sh`

Users should put any additional start-up actions in the file `/mnt/init.sh`:

```
cp /opt/init.sh /mnt/init.sh
```

```
vi /mnt/init.sh
```

The default `/opt/init.sh` script will display the software version on the console and start the sensor server running in the background.

> Changing anything on the file system can result in an irreparable state where the radar has to be re-programmed by RFbeam Microwave GmbH.

Network Interface

The radar uses a static IP address (default 192.168.16.2 / 255.255.255.0). The sensor server listens for connections on the specified TCP port (default 6172). The radar will continually send messages to any connected client and accept command messages at any time. Command messages are processed between frames.

The messages generated by the radar can exceed the speed of the TCP/IP stack. When this happens the frame rate of the radar will drop so that all messages are transmitted. This can be prevented by limiting the type of messages transmitted using the DSF0 command.

The IP address and netmask can be changed by Telnet using the ifconfig command:

```
ifconfig eth0 192.168.1.5
netmask 255.255.255.0
```

Serial Interface

Specification	Value
Voltage	3.3V
Baudrate	115200
Data bits	8
Stop bits	1
Parity	No
Hardware flow control	No

Table 3: Serial interface specifications

The radar continually sends messages via the serial interface and accepts command messages at any time. Command messages are processed between frames. Serial commands are processed after network commands.

The messages generated by the radar can exceed the speed of the serial interface. When this happens the frame rate of the radar will drop so that all messages are transmitted. This can be prevented by limiting the maximum size of messages sent via the serial interface with the `serial_max_length` configuration key. Messages exceeding this length will not be sent via the serial interface, but they will still be sent via the network interface.

Logging

The sensor server outputs messages to the console and a log file (/opt/sensor_server.log). The level of the messages can be different on the console and in the log file. The levels are:

Level	Description
0	No messages (silent operation)
1	Errors (These are usually fatal, and the software will exit)
2	Warnings (These are usually non-fatal, but are not expected)
3	Notes (These are expected and should only occur at start-up or initialisation)
4	Trace (These messages provide additional information to the user)
5	Debug (These messages provide information to development engineers)

Table 4: Level of log messages

The level specified includes all messages from lower levels. The default level for the log file is Notes and the default level for the console is Trace. Care should be taken not to set the level for the log file too high in case the file system runs out of space.

File System

> ⚠ Changing anything on the file system can result in an irreparable state where the radar has to be reprogrammed by RFbeam Microwave GmbH.

File	Description
`/etc/init.d/rcS`	This file is always run, but will refer to `/mnt/rcS` if it exists.
`/mnt/rcS`	(optional) This file is run instead of `/etc/init.d/rcS`
`/opt/init.sh`	This is the default start-up script that starts the sensor server
`/mnt/init.sh`	(optional) This file is run instead of `/opt/init.sh`
`/opt/sensor_server`	This is the executable. It reads the configuration file(s).
`/opt/sensor_server.conf`	This is the default configuration file for the sensor server.
`/mnt/sensor_server.conf`	This is the user configuration file. Any entries override those in the default configuration file.
`/opt/sensor_server.log`	This is the default output log file.
`/opt/roms_*`	Files in this folder are loaded on to the processor.
`/opt/versions`	This file shows the software and firmware versions that were used to create the boot image.
`/etc/fstab`	This file defines the mount points, including `/mnt`
`/tmp`	This is a ram disk, used for updating the firmware.

Table 5: Radar file system

Configuration

The sensor server configuration is loaded from files containing key/value pairs. The key and value must be separated by an equals sign:

key = value

The default configuration file `/opt/sensor_server.conf` is always loaded. If the user configuration file `/mnt/sensor_server.conf` exists, it is loaded as second and any values overwrite the value loaded from the default configuration file.

The following table describes the system parameter which can be changed by the user in the `/mnt/sensor_server.conf`.

> ⚠ System parameter which aren't described in this table are system dependent and must not be changed. Changing of these parameter can result in an irreparable state where the radar has to be reprogrammed by RFbeam Microwave GmbH.

Key	Description
`log_level_file`	The level for messages in the log file.
`log_level_stdout`	The level for messages displayed on the terminal.
`log_filename`	The name and location of the log file.
`datasink_timeout`	Maximum time to allow for receiving commands.
`datasink_filter`	The types of message sent by the sensor server.
`datasink_port`	The network TCP port.
`serial_device`	The serial interface.
`serial_baud`	The serial interface baud rate.
`serial_max_length`	The maximum length of messages sent via the serial interface.
`chirp_setup_time`	Clocks from PLL trigger to ADC sample.
`chirp_hold_time`	Clocks from ADC sample to PLL trigger.
`chirp_num_sweeps`	Number of sweeps in the FMCW modulation scheme.
`chirp_num_repeats`	Number of times the sweeps are repeated.
`chirp_num_repeats_ignored`	Number of repeats where the ADC samples are not stored.
`chirp_num_steps`	Number of steps in each FMCW sweep.
`chirp_num_steps_ignored`	Number of steps in the first sweep where the ADC sample is not stored.
`chirp_initial_delay`	Number of clocks between the first PLL trigger and the first ADC sample.
`chirp_start`	The initial frequency of the FMCW sweep.
`chirp_bandwidth`	The bandwidth of the FMCW sweep.
`radar_rx_gain`	The receiver VGA gain in dB

Table 6: System parameter

The following table describes the processing parameter which can be changed by the user in the /mnt/sensor_server.conf.

Processing parameter which aren't described in this table are system dependent and must not be changed. Changing these parameters can result in an irreparable state where the radar has to be reprogrammed by RFbeam Microwave GmbH.

Key	Description
mean_channels	The number of receiver channels to average for the mean range-doppler map.
peak_threshold	The signal threshold for peak detection.
max_num_peaks	The maximum number of peaks to find.
background_update	The number of frames to average for the background range-doppler map.
range_compensation	The rate at which the peak threshold is adjusted with range, relative to mid-range.
min_range	The minimum range cell for finding peaks.
max_range	The maximum range cell for finding peaks.
min_speed	The minimum doppler bin for finding peaks.
max_speed	The maximum doppler bin for finding peaks.
smoothing	Smooth the mean range-doppler map with a 3x3 Gaussian window.
azimuth_ref_channel	The reference receiver channel for Azimuth direction calculation.
azimuth_alt_channel	The alternate receiver channel for Azimuth direction calculation.
azimuth_offset	The phase offset for the azimuth channel.
azimuth_ratio	The ratio between azimuth phase and reported direction.
elevation_ref_channel	The reference receiver channel for Elevation direction calculation.
elevation_alt_channel	The alternate receiver channel for Azimuth direction calculation.
elevation_offset	The phase offset for the elevation channel.
elevation_ratio	The ratio between elevation phase and reported direction.
max_num_tracks	The maximum number of tracks to report. The internal list length is the same as the maximum number of peaks to find.
max_range_jitter	Maximum difference in range bins between predicted range and range of peak.
max_speed_jitter	Maximum difference in doppler bins between predicted speed and speed of peak.
direction_error_threshold	Maximum difference in degrees between predicted direction and direction of peak.
min_track_life	Minimum number of frames with tracked detection before track is reported.
max_track_life	Maximum number of frames without tracked detection before track is lost.
track_history	Number of previous frames to use in tracking.
stationary_objects	Report tracks that have never moved.
constant_speed	Assume constant speed in tracking algorithm.

Table 7: Processing parameter

Firmware Update

The processor can boot from an image stored in SPI flash. The firmware is loaded from the boot image and then a file system is unpacked containing the software. The current version of the firmware and software is shown in the /opt/versions file.

The flash is split into two sections, with the boot image stored in the first section. The second section is formatted with a JFFS2 file system and mounted to /mnt.

The boot image can be updated by transferring the new boot image to /tmp and writing the image to the first section of the flash with flashcp:

```
flashcp -v /tmp/BOOT.bin /dev/mtd0
```

After the device has been powered off and on, the new firmware is loaded.

Application Layer

Client-Server

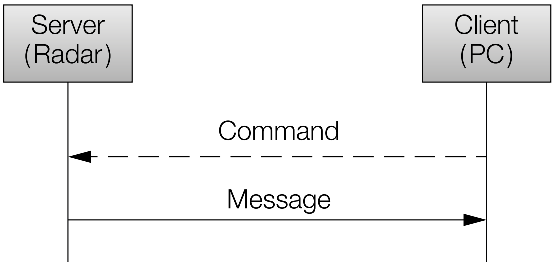


Figure 13: Client-Server model

The communication is based on a client-server model. There are two types of packets transmitted. Commands are sent from client to server and messages are sent from server to client.

Presentation Layer

All commands and messages sent have the format described in table x.

Description	Datatype	Length
Header The header describes the command or message type (e.g. RADC, RMRD, …)	ASCII character	4
Payload Length The payload length is always sent even if the payload is zero. It is sent as little endian (LSB first).	UINT32	4
Payload The payload is message and command dependent. If the payload includes datatypes (e.g. UINT16, INT32, …) then they are sent as little endian (LSB first). This data is typically padded to a multiple of 4 bytes.	Binary data	4

Table 8: Packet format

Overview Messages and Commands

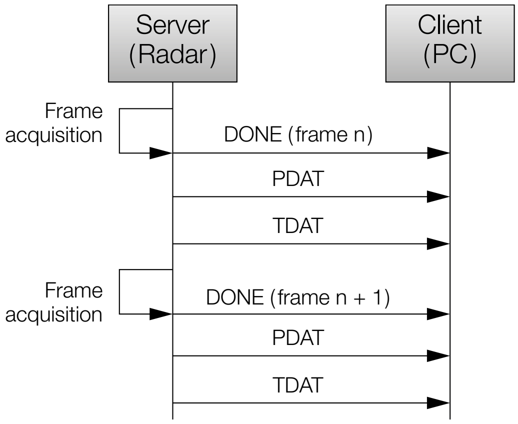


Figure 14: Cyclic message output

The server acquires frames in a fixed cycle of 50 ms. After every frame cycle it outputs all enabled messages. The messages DONE, PDAT and TDAT are enabled by default.

The table x shows the possible messages – see the chapter 'Messages' for details.

Header	Payload Length	Description
DONE	0	Frame done
RADC	786432	Raw ADC values
RMRD	262144	Mean range-doppler map
PDAT	0-2400	The array of detected raw targets
TDAT	0-8800	The array of tracked targets
RPRM	12	Radar parameter structure
PPRM	56	Processor parameter structure
GBYE	0	Indicates that the sensor server exits

Table 9: Application messages

The table Application commands shows the possible application commands, see the chapter Commands for details.

Header	Payload Length	Description	Default
INIT	0	Run the initialization sequence and load the configuration files	–
RSID	4	Radar initial delay [clk]	436
RSSF	4	Radar start frequency [MHz]	24028
RSBW	4	Radar bandwidth [MHz]	194
RSRG	4	Radar receiver gain [dB]	20
PSMC	4	Receivers to include in mean range-doppler map	7 (0b111)
PSPT	4	Processor peak detection threshold	1000
PSRC	4	Processor range compensation for threshold	0.0
PSNP	4	Processor maximum number of peaks	200
PSBU	4	Processor background update rate	128
PSBR	4	Processor peak detection minimum range	2
PSTR	4	Processor peak detection maximum range	200
PSBS	4	Processor peak detection minimum speed	0
PSTS	4	Processor peak detection maximum speed	100
PSSM	4	Processor smooth mean range-doppler map	1
PSNT	4	Processor maximum number of tracks to report	20
PSRJ	4	Processor maximum range jitter	2
PSSJ	4	Processor maximum speed jitter	3
PSBL	4	Processor minimum track life	5
PSTL	4	Processor maximum track life	15
PSTH	4	Processor length of history for tracking	10
PSSO	4	Processor report stationary objects	1
PSCS	4	Processor assume constant speed for tracking	1
DSF0	4	Disable messages with given header	–
DSF1	4	Enable messages with given header	–
GBYE	0	Disconnect but leave the sensor server running	–
STOP	0	Stop the sensor server	–

Table 10: Application commands

If a command is rejected a warning will be displayed on the console and in the log file. No response is provided via the network or serial interface to accept or reject the command.

The INIT and STOP commands are acknowledged with a DONE message so that the client knows when the command has been completed. When sending the GBYE command the client should read any outstanding messages before disconnecting.

Messages

This chapter provides detailed information about the messages of the K-MD2.

Header	Payload Length	Description	Payload: Description	Datatype	Length
DONE	0	Frame done	–		
RADC	786432	Raw ADC values	RX1: Chirp 0: Sample 0-255: I-Channel, Q-Channel … to Chirp 255: Samples 0-255: I-Channel, Q-Channel	UINT16	262144
			RX2: Chirp 0: Sample 0-255: I-Channel, Q-Channel … to Chirp 255: Samples 0-255: I-Channel, Q-Channel	UINT16	262144
			RX3: Chirp 0: Sample 0-255: I-Channel, Q-Channel … to Chirp 255: Samples 0-255: I-Channel, Q-Channel	UINT16	262144
RMRD	262144	Mean range-doppler map. Averaged magnitudes of all three RX antennas.	Range 0: Speed 0-255 … to Range 255: Speed 0-255	UINT32	262144
PDAT	0-2400	The array of detected raw targets	Range bin of mean range-doppler map [bin]	UINT16	2
			Speed bin of mean range-doppler map [bin]	UINT16	2
			Azimuth direction [° x 100]	INT16	2
			Elevation direction [° x 100]	INT16	2
			Magnitude of peak	UINT16	2
			Reserved, Used internally	UINT16	2
TDAT	0-8800	The array of tracked targets	Identification number of track	INT32	4
			Life [frame]	INT32	4
			Range bin of mean range-doppler map [bin]	FLOAT	4
			Speed bin of mean range-doppler map [bin]	FLOAT	4
			Doppler acceleration [bin/frame]	FLOAT	4
			Azimuth direction [°]	FLOAT	4
			Reserved, Used internally	FLOAT	4
			Elevation direction [°]	FLOAT	4
			Number of micro doppler peaks	FLOAT	4
			Magnitude of peak	FLOAT	4
			Reserved, Used internally	FLOAT	4
RPRM	12	Radar parameter structure	Radar initial delay [clk]	UINT16	2
			Radar start frequency [MHz]	UINT16	2
			Radar bandwidth [MHz]	UINT16	2
			Radar receiver gain [dB]	UINT16	2
			Reserved, Used internally	UINT16	2
			Reserved, Used internally	UINT16	2

<table>
<tr><th>Header</th><th>Payload Length</th><th>Description</th><th colspan="3">Payload</th></tr>
<tr><td rowspan="24">PPRM</td><td rowspan="24">56</td><td rowspan="24">Processor parameter structure</td><td>Description:</td><td>Datatype</td><td>Length</td></tr>
<tr><td>Processor peak detection threshold</td><td>UINT32</td><td>4</td></tr>
<tr><td>Reserved, Used internally</td><td>UINT32</td><td>4</td></tr>
<tr><td>Processor maximum number of peaks</td><td>UINT16</td><td>2</td></tr>
<tr><td>Processor background update rate</td><td>UINT16</td><td>2</td></tr>
<tr><td>Processor range compensation for threshold</td><td>FLOAT</td><td>4</td></tr>
<tr><td>Processor peak detection minimum range [bin]</td><td>UINT16</td><td>2</td></tr>
<tr><td>Processor peak detection maximum range [bin]</td><td>UINT16</td><td>2</td></tr>
<tr><td>Processor peak detection minimum speed [bin]</td><td>UINT16</td><td>2</td></tr>
<tr><td>Processor peak detection maximum speed [bin]</td><td>UINT16</td><td>2</td></tr>
<tr><td>Processor smooth mean range-doppler map
0x00 = Off 0x01 = On</td><td>UINT16</td><td>2</td></tr>
<tr><td>Reserved, Used internally</td><td>UINT16</td><td>2</td></tr>
<tr><td>Processor maximum number of tracks to report</td><td>UINT16</td><td>2</td></tr>
<tr><td>Processor maximum range jitter [bin]</td><td>UINT16</td><td>2</td></tr>
<tr><td>Processor maximum speed jitter [bin]</td><td>UINT16</td><td>2</td></tr>
<tr><td>Processor minimum track life [frame]</td><td>UINT16</td><td>2</td></tr>
<tr><td>Processor maximum track life [frame]</td><td>UINT16</td><td>2</td></tr>
<tr><td>Direction error threshold [° x 100]</td><td>INT16</td><td>2</td></tr>
<tr><td>Processor length of history for tracking [frame]</td><td>UINT16</td><td>2</td></tr>
<tr><td>Processor report stationary objects
0x00 = Off 0x01 = On</td><td>UINT16</td><td>2</td></tr>
<tr><td>Processor assume constant speed for tracking
0x00 = Off 0x01 = On</td><td>UINT16</td><td>2</td></tr>
<tr><td>Reserved, Used internally</td><td>UINT16</td><td>2</td></tr>
<tr><td>Range scaling factor [m]</td><td>FLOAT</td><td>4</td></tr>
<tr><td>Speed scaling factor [m/s]</td><td>FLOAT</td><td>4</td></tr>
<tr><td>GBYE</td><td>0</td><td>Indicates that the sensor server exits</td><td colspan="3">–</td></tr>
</table>

Table 11: Application messages

Commands

The commands are sent to the radar to set configurations. The payload length may be zero or four bytes. This chapter provides detailed information about the commands of the K-MD2.

Header	Description	Datatype	Length
INIT	Run the initialization sequence and load the configuration files	-	0
RSID	Radar initial delay [clk]. The delay before the ramp starts in every chirp. The core clock is 38461538Hz. 0-32768 See chapter "Speed and Range Settings" for more information	UINT16	4
RSSF	Radar start frequency [MHz] 23800–24800 See chapter "Speed and Range Settings" for more information	UINT16	4
RSBW	Radar bandwidth [MHz] 100–1000 See chapter "Speed and Range Settings" for more information	UINT16	4
RSRG	Radar receiver gain [dB] 0–80	UINT8	4
PSMC	Receivers to include in mean range-doppler map 0x01 = RX1 0x02 = RX2 0x04 = RX3 Can be "OR" bit coded	UINT8	4
PSPT	Processor peak detection threshold 0–32768 See chapter "Peak Threshold" for more information	UINT16	4
PSRC	Processor range compensation for threshold 0.0–5.0 See chapter "Range Compensation" for more information	FLOAT	4
PSNP	Processor maximum number of peaks 0–200	UINT8	4
PSBU	Processor background update rate 0, 2, 4, 8, 16, 32, 64, 128, 256 See chapter "Background Update" for more information	UINT16	4
PSBR	Processor peak detection minimum range [bin] 2–254	UINT8	4
PSTR	Processor peak detection maximum range [bin] 2–254	UINT8	4
PSBS	Processor peak detection minimum speed [bin] 0–126	UINT8	4
PSTS	Processor peak detection maximum speed [bin] 0–126	UINT8	4
PSSM	Processor smooth mean range-doppler map 0x00 = Off 0x01 = On See chapter "Smoothing" for more information	UINT8	4
PSNT	Processor maximum number of tracks to report 0–200	UINT8	4
PSRJ	Processor maximum range jitter [bin] 0–10 See chapter "Range and Speed Jitter" for more information	UINT8	4
PSSJ	Processor maximum speed jitter [bin] 0–10 See chapter "Range and Speed Jitter" for more information	UINT8	4
PSBL	Processor minimum track life [frame] 0–255 See chapter "Track Life" for more information	UINT8	4
PSTL	Processor maximum track life [frame] 0–255 See chapter "Track Life" for more information	UINT8	4

Header	Description	Datatype	Length
PSTH	Processor length of history for tracking [frame] 0–255	UINT8	4
PSSO	Processor report stationary objects 0x00 = Off 0x01 = On	UINT8	4
PSCS	Processor assume constant speed for tracking 0x00 = Off 0x01 = On See chapter “Assume Constant Speed” for more information	UINT8	4
DSF0	Disable messages with given header (e.g. RADC, PDAT, …)	ASCII character	4
DSF1	Enable messages with given header (e.g. RADC, PDAT, …)	ASCII character	4
GBYE	Disconnect but leave the sensor server running	-	0
STOP	Stop the sensor server	-	0

Table 12: Application commands

OUTLINE DIMENSIONS

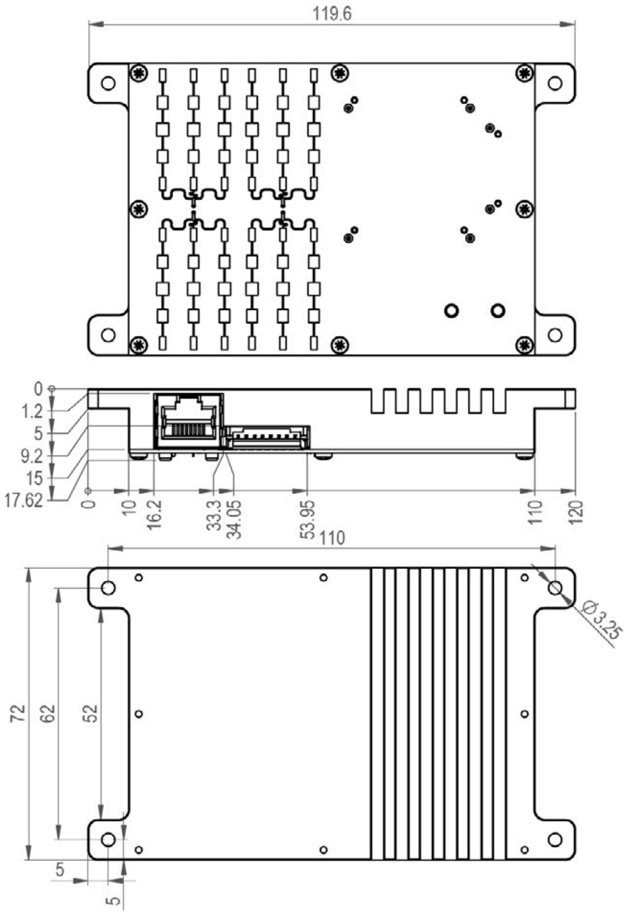


Figure 15: Outline dimensions in milimeter

ORDER INFORMATION

The ordering number consists of different parts with the structure below.

Product	–	Customer	–	HW variant	Supply	–	SW variant
= K-MD2		**= RFB**		**= 01** for engineering sample	**= F** for 12 V version		**= 01** for engineering sample

Figure 16: Ordering number structure

Ordering number	Description	Release date
K-MD2-RFB-01F-01	K-MD2 Engineering Sample	06/2018

Table 13: Available ordering numbers

DELIVERY CONTENT

- K-MD2 Engineering Sample
- Power supply with adapter cable
- Ethernet cable
- Memory stick containing:
  - Installer for “K-MD2 Control Panel”
  - Firmware and configuration files
  - Source code of Control Panel available as Microsoft Visual Studio 2017 project

VERSIONING

There are different version descriptions for software and firmware:

TE0720-1xx_boot.bin → Application firmware boot image
TE0720-1xx_sensor_server.conf → Firmware configuration file
K-MD2_CTP-RFB-01xx.exe → PC control panel software

REVISION HISTORY

06/2018 – Revision A: Initial Version