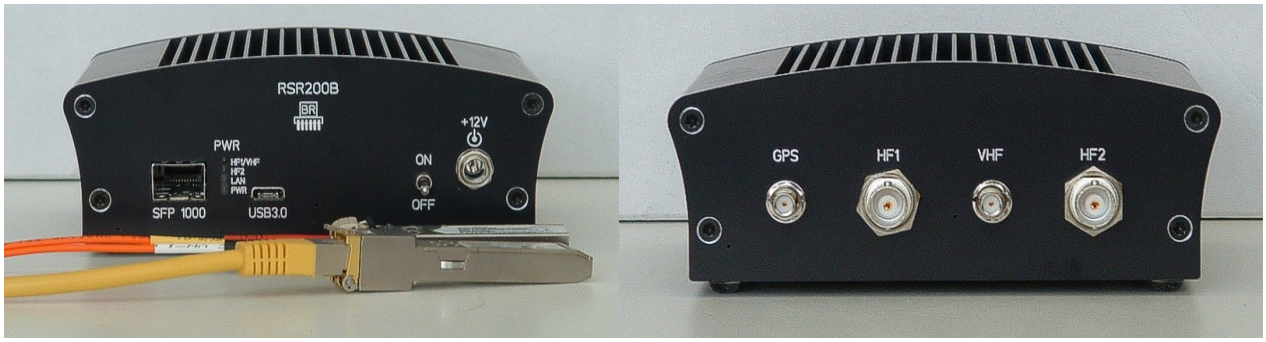
Data Protocols

Direct Digitizing Black Box Receiver

RSR200



Version: 0.3
Created: 25.11.2025
Last changed: 10.01.2026

Content

Change log

Version No.	Description	Firmware
01	Initial release	221
02	- Changed polarity of bit 2 variable no. 5 "Switch" of the command "Set variable 16 bit value" - Increased resolution of the ADC clock frequency value to 0.1 MHz in the command "Set ADC clock".	222
03	- Updated DSP Schematic	222

1. Basics

The RSR200 possesses a USB and a network (LAN) interface for connection to one or two PCs. Data is transferred in both directions via these interfaces:

RSR200 → PC: IQ data from the ADC and subsequent signal processing, measured values, confirmations for executed control commands

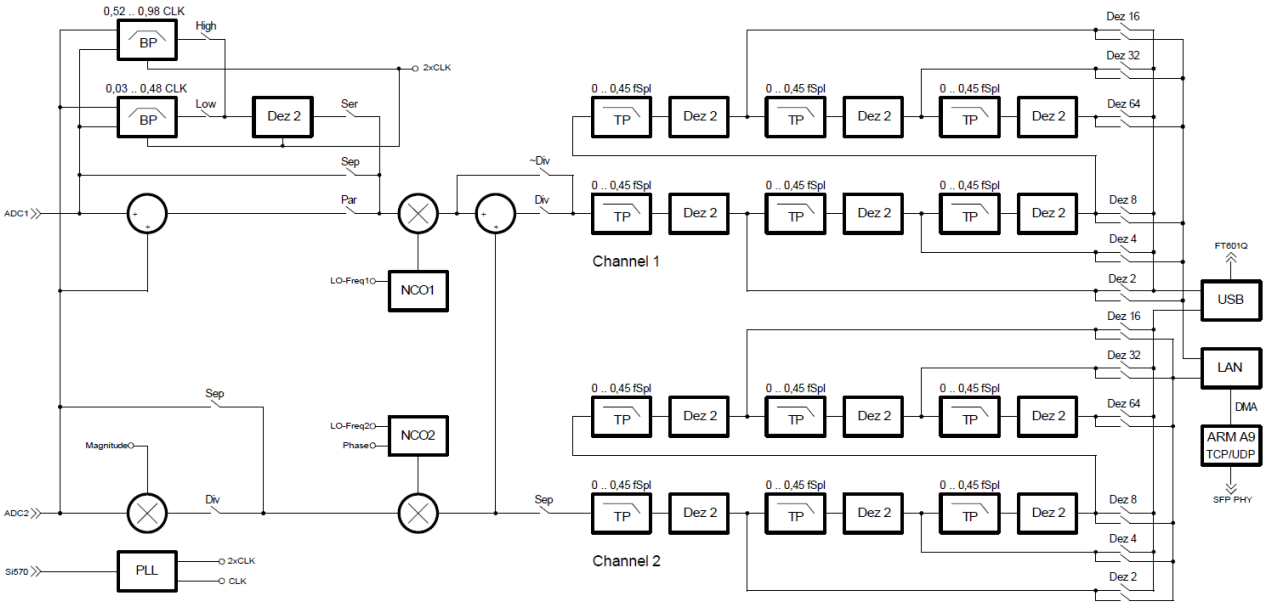
PC → RSR200: Control commands, firmware files

The data content is largely the same for both interfaces. The formatting of the data (number of bytes transferred, arrangement of the data in the transferred bytes) varies between the interfaces and the protocols used.

Data transmission from the RSR200 to the PC is optimized for the highest possible speed (maximum bandwidth of the IQ data). The data is transmitted in the form of packets and blocks (several related packets). The packets contain user (IQ) and control data (commands). Packets or blocks are sent regularly from the RSR200 to the PC (“streaming”). The packet / block rate depends on the amount of user data generated in the RSR200 (sampling rate) and other settings (e.g., bit width of the data, number of channels). The PC must be able to receive all packets / blocks within the sampling rate time interval, otherwise data loss will occur.

For a description of the RSR200’s functionality, refer to its manual and, in particular, the block diagram. The explanation of the control elements provides information on useful setting options and the command structure in the data packets.

Values in the data packets are displayed in decimal (simple numerical values) or hexadecimal in C notation (preceded by 0x).



Hardware for digital signal processing inside the FPGA of the RSR200.

2. Interfaces

2.1 USB

The USB interface operates according to the USB 3.0 standard (5 Gbit). The RSR200 uses the FT601Q circuit from FTDI. The PC software should use the API created by the manufacturer for this circuit. It contains a library with routines for connecting and transferring data from a USB 3.0-compliant port on the PC to the RSR200.

The RSR200 operates the FT601Q with the following settings:

- One bulk transmit endpoint (0x82) with 4096 bytes of double buffer memory

- One bulk receive endpoint (0x02) with 512 bytes of buffer + FPGA-internal 264 bytes of buffer.

After switching on, the RSR200 immediately begins processing ADC data and filling the transmit memory. As soon as a buffer is full, the endpoint is marked as “Ready” and the corresponding routine in the PC (one of the read commands of the FT601Q API) can read the buffer. This must be done in less time than it takes for the second buffer memory in the RSR200 to fill up. This enables continuous transmission (streaming) without any data loss (one buffer is filled while the other is being read by the PC). The time required to fill a buffer is calculated from the sampling rate, the bit width, and the number of channels of user data.

Note: Reading a packet, processing the data, and requesting data again requires computing time on the PC. The FTDI API includes options for reading multiple packets with minimal overhead. To do this, the pipe for reading endpoint 0x82 can be set to “streaming” and asynchronous reading (“overlapped” mechanism) can be used. The read command should be assigned as many individual buffers of 4096 bytes as possible. For minimum overhead, waiting for the buffers to fill should not be interrupted (parameter Overlapped.Result = True). However, this will cause the program to freeze if errors occur!

A packet = a block (a buffer content) of the USB interface has the following structure:

Byte no.	Value	Description
0 : 3	LSB packet counter : MSB packet counter	Consecutive packet counter 32 bit
4 : 4083	1st Data byte : 4080th Data byte	4080 Bytes IQ data
4084	Signed Byte	Temperature in °C
4085 4086	LSB GPS MSB GPS, OV	Current value of the frequency correction in Hz as a signed 14-bit value, OV1, OV2
4087	Command no.	Number of the currently transmitted command
4088 : 4095	LSB command : MSB command	8 bytes current command

The data is 32-bit word-oriented. The alignment is “little endian”; low-order bytes are transferred first or are located at lower addresses. Packet size = block size = 4096 bytes = 1024 words.

The first data word contains a packet counter, which is incremented by 1 in each subsequent packet. This is followed by 1020 words = 4080 data bytes with IQ data (see below). The end of the packet contains measured values and the current command from the RSR200 to the PC (see section Commands).

Each packet contains 4080 bytes of IQ data. The number of samples contained therein varies depending on the settings. The data is organized as follows, depending on the set resolution and number of channels.

2.1.1 Data arrangement at single-channel mode 16 bits

IQ data 1-channel 16 bit: 1020 samples

Byte no.	Value	Description
0	LSB I data	16 bit Inphase data 1st sample in packet
1	MSB I data	
2	LSB Q data	16 bit quadrature data 1st sample in packet
3	MSB Q data	
:	:	:
4076	LSB I data	16 bit Inphase data 1020th sample in packet
4077	MSB I data	
4078	LSB Q data	16 bit quadrature data 1020th sample in packet
4079	MSB Q data	

2.1.2 Data arrangement at dual-channel mode 16 bits

IQ data 2-channel 16 bit: 510 samples

Byte no.	Value	Description
0	LSB I data	16 bit Inphase data 1st sample channel 1
1	MSB I data	
2	LSB Q data	16 bit quadrature data 1st sample channel 1
3	MSB Q data	
4	LSB I data	16 bit Inphase data 1st sample channel 2
5	MSB I data	
6	LSB Q data	16 bit quadrature data 1st sample channel 2
7	MSB Q data	
:	:	:
4072	LSB I data	16 bit Inphase data 510th sample channel 1
4073	MSB I data	
4074	LSB Q data	16 bit quadrature data 510th sample channel 1
4075	MSB Q data	
4076	LSB I data	16 bit Inphase data 510th sample channel 2
4077	MSB I data	
4078	LSB Q data	16 bit quadrature data 510th sample channel 2
4079	MSB Q data	

2.1.3 Data arrangement at single-channel mode 24 bits

IQ data 1-channel 24 bit: 680 samples

Byte no.	Value	Description
0	LSB I data	24 bit Inphase data 1st sample in packet
1	:	
2	MSB I data	
3	LSB Q data	24 bit quadrature data 1st sample in packet
4	:	
5	MSB Q data	
:	:	:
4074	LSB I data	24 bit Inphase data 680th sample in packet
4075	:	
4076	MSB I data	
4077	LSB Q data	24 bit quadrature data 680th sample in packet
4078	:	
4079	MSB Q data	

2.2 LAN

The LAN interface operates according to the IEEE 802.3 (Ethernet) standard with a data rate of 1 Gbit. Data is transferred using the standard TCP and UDP protocols. After switching on, the RSR200 sets up a TCP server and waits for incoming connections. The server operates on port 55557. The IP address is stored non-volatile in the RSR200. The default address is 191.168.1.10 and can be set to any value using a control command. After switching on, the RSR200 runs a DHCP client for approx. 5 seconds. During this time, the address can be changed dynamically (for the duration of the RSR200's switch-on time) via the DHCP service.

The TCP server initially only sends confirmations (short TCP packets without IQ data) for incoming commands (see section Commands). It can be switched to streaming mode via a command. It then regularly sends larger blocks of user and control data (as user data becomes available and the blocks are filled). It can be switched back to packet mode via command.

In addition to the TCP server, a UDP service can be set up via a command. The UDP service operates on port 55558 and always receives/sends individual packets. In streaming mode, as many logically related individual packets are sent as are necessary to transmit a complete LAN data block (IQ and command data).

Note: TCP and UDP transceivers are built directly onto the TCP-IP stack “lwip220” in the RSR200 without an operating system (minimal overhead). This allows them to achieve almost the maximum possible net data rate of just under 1 GBit. The PC must be able to receive this data rate, otherwise data will be lost. The implementation of the RSR200’s “ExtIO-DLL” operates with asynchronous TCP or UDP sockets from the Microsoft WINSOCK2 API and thus achieves a data rate of approx. 900 Mbit on an idle PC (no other programs and no further signal processing).

The LAN interface always operates with blocks containing the same number of samples (130560) and thus with a variable block length depending on the set resolution and number of channels (see below). Data is generally byte-oriented, values greater than 1 byte are little-endian oriented. The IQ samples are positioned at the beginning of a block. They are followed by a continuous 32-bit block counter (incremented by 1 in each subsequent block). The block counter is contained again in the next 4 bytes, but inverted (ones’ complement). This is followed by 8 bytes for synchronization purposes (always the same in each block). The end of the block contains commands from the RSR200 to the PC (see section Commands).

The block counters and synchronization bytes can be used to identify the block (and, in the case of UDP, also the packet) boundaries in a continuous data stream (reception of any number of bytes from the sockets). With the help of the counters and the known fixed length (depending on the operating mode) of the blocks, it is also possible to check for missing data and thus for the quality of the transmission (interference, interruptions, or exceeding the possible transmission rate). Error checking of the data and possible correction (checksums, FEC, etc.) is not implemented at a higher level.

2.2.1 TCP data transmission

TCP allows data blocks of (almost) any size to be sent. The TCP stack handles all data processing, such as packet generation, dividing large amounts of data into individual TCP packets, error detection/correction, etc. For the RSR200, relatively large data blocks are therefore defined as the smallest unit (no individual packets). This requires relatively little softwareoverhead.

TCP operates with data checking and acknowledgment, which enables relatively secure data transmission (automatic retries in case of errors). However, this requires additional overhead and reduces the data rate. Especially in the case of errors, the data stream is interrupted until the faulty packet has been retransmitted correctly. This hinders continuous streaming of real-time data (data overflow with losses at the sender).

Note: There are many different control parameters for the TCP protocol that can be used to fine-tune the acknowledgment system (e.g., buffer sizes, wait times, number of packets acknowleged simultaneously, and much more). Most of these work automatically within the TCP stack or the PC operating system. The programmer is required to program optimal values for the stack settings (if accessible) in order to achieve high data rates.

2.2.1.1 TCP block structure at single-channel mode 16 bits

TCP block 1-channel 16 bit: 522704 bytes

Byte no.	Value	Description
0	LSB I data	16 bit Inphase data 1st sample in block
1	MSB I data	
2	LSB Q data	16 bit quadrature data 1st sample in block
3	MSB Q data	
:	:	:
522236	LSB I data	16 bit Inphase data 130560th sample in block
522237	MSB I data	
522238	LSB Q data	16 bit quadrature data 130560th sample in block
522239	MSB Q data	
522240	LSB block counter	Consecutive block counter 32 bit
:	:	
522243	MSB block counter	
522244	LSB inverted block counter	Consecutive block counter 32 bit, inverted
:	:	
522247	MSB inverted block counter	
522248	0x78	1st Synchronization word
522249	0x56	
522250	0x34	
522251	0x12	
522252	0xF0	2nd Synchronization word
522253	0xDE	
522254	0xBC	
522255	0x9A	
522256	Signed Byte	Temperature in °C
522257	LSB GPS	Current value of the frequency correction in Hz as a
522258	MSB GPS, OV	signed 14-bit value, OV1, OV2
522259	Command no.	Number of the currently transmitted command block
522260	LSB command amount	32 bit count of following commands
:	:	
522263	MSB command amount	
522264	Command	1st Byte (instruction) 1st command
:	:	:
522703	Data byte	Last possible byte for last command

2.2.1.2 TCP block structure at dual-channel mode 16 bits

TCP block 2-channel 16 bit: 1045408 bytes

Byte no.	Value	Description
0	LSB I data	16 bit Inphase data 1st sample channel 1
1	MSB I data	
2	LSB Q data	16 bit quadrature data 1st sample channel 1
3	MSB Q data	
4	LSB I data	16 bit Inphase data 1st sample channel 2
5	MSB I data	
6	LSB Q data	16 bit quadrature data 1st sample channel 2
7	MSB Q data	
:	:	:
1044472	LSB I data	16 bit Inphase data 130560th sample channel 1
1044473	MSB I data	
1044474	LSB Q data	16 bit quadrature data 130560th sample channel 2
1044475	MSB Q data	
1044476	LSB I data	16 bit Inphase data 130560th sample channel 2
1044477	MSB I data	
1044478	LSB Q data	16 bit quadrature data 130560th sample channel 2
1044479	MSB Q data	
1044480	LSB block counter	Consecutive block counter 32 bit
:	:	
1044483	MSB block counter	
1044484	LSB inverted block counter	Consecutive block counter 32 bit, inverted
:	:	
1044487	MSB inverted block counter	
1044488	0x78	1st Synchronization word
1044489	0x56	
1044490	0x34	
1044491	0x12	
1044492	0xF0	2nd Synchronization word
1044493	0xDE	
1044494	0xBC	
1044495	0x9A	
1044496	Signed Byte	Temperature in °C
1044497	LSB GPS	Current value of the frequency correction in Hz as a
1044498	MSB GPS, OV	signed 14-bit value, OV1, OV2
1044499	Command no.	Number of the currently transmitted command block
1044500	LSB command amount	32 bit count of following commands
:	:	
1044503	MSB command amount	
1044504	command	1st Byte (instruction) 1st command
:	:	:
1045407	Data byte	Last possible byte for last command

2.2.1.3 TCP block structure at dual-channel mode 24 bits

TCP block 1-channel 24 bit: 784784 bytes

Byte no.	Value	Description
0	LSB I data	24 bit Inphase data 1st sample in block
1	:	
2	MSB I data	
3	LSB Q data	24 bit quadrature data 1st sample in block
4	:	
5	MSB Q data	
:	:	:
783354	LSB I data	24 bit Inphase data 130560th sample in block
783355	:	
783356	MSB I data	
783357	LSB Q data	24 bit quadrature data 130560th sample in block
783358	:	
783359	MSB Q data	
783360	LSB block counter	Consecutive block counter 32 bit
:	:	
783363	MSB block counter	
783364	LSB inverted block counter	Consecutive block counter 32 bit, inverted
:	:	
783367	MSB inverted block counter	
783368	0x78	1st Synchronization word
783369	0x56	
783370	0x34	
783371	0x12	
783372	0xF0	2nd Synchronization word
783373	0xDE	
783374	0xBC	
783375	0x9A	
783376	Signed Byte	Temperature in °C
783377	LSB GPS	Current value of the frequency correction in Hz as a
783378	MSB GPS, OV	signed 14-bit value, OV1, OV2
783379	Command no.	Number of the currently transmitted command
783380	LSB command amount	32 bit count of following commands
:	:	
783383	MSB command amount	
783384	command	1st Byte (instruction) 1st command
:	:	:
784783	Data byte	Last possible byte for last command

2.2.2 UDP data transmission

Depending on the implementation of the stack, UDP allows data blocks of (almost) any size to be sent. However, the effort required to fragment a block into UDP packets is considerable with the lwip220 stack used and reduces the possible data rate. In the RSR200, the LAN data block is therefore divided into individual UDP packets by RSR200 firmware with a length that does not require further fragmentation (1458 bytes).

UDP packets are not checked for loss or transmission errors and are therefore not automatically repeated in the event of errors. No acknowledgments are sent back from the receiver to the sender. The receiver does not have to wait for these acknowledgments. This means that UDP allows continuous streaming without data stalls in the event of errors. However, if errors occur, the data in the affected packets is damaged or irretrievably lost. It is therefore recommended to use UDP only for IQ data and not for the possible transmission of control commands.

Note: The RSR200 sends individual UDP packets with commands only for requesting the version number when streaming is not enabled. In streaming mode, however, measured values and command confirmations are also sent in the LAN block.

Depending on the block length, a certain number of individual UDP packets that are logically related (a “LAN block”) are sent. After reception, the individual packets must be reassembled into a block according to their packet numbers. Each packet has the following structure:

UDP packet

Byte no.	Value	Description
0	LSB packet number	16 bit number of the packet in the block (from 0)
1	MSB packet number	
2	Data	1st Data byte in the packet (consecutive from byte 0 of
:	:	the block)
1457	Data	Last data byte in the packet

The packet begins with a 16-bit packet number (position of the packet in the block, starting with 0 at the beginning of the block), followed by 1456 bytes of consecutive data from the block. There is no relation between the bytes and sample boundaries. Only in the reassembled block certain byte numbers correspond to certain beginnings and ends of samples (see TCP block).

Except for the last packet, all packets contain IQ data. The last packet contains the last IQ data up to the 130560th sample of the block and, in the remaining part, the control commands of the block.

2.2.2.1 UDP block structure for single-channel mode 16 bits

UDP block 1-channel 16 bit: 359 packets = 523422 bytes

Byte no.	Value	Description
0	1st Byte of the 1st packet	Begin 1st UDP packet
:	:	:
1457	1458th Byte of the 1st packet	End 1st UDP packet
:	:	:
521964	1st Byte of the last packet	Begin 359th UDP packet
:	:	:
523421	1458th Byte of the last packet	End 359th UDP packet

359. Packet in block 1-channel 16 bit

Byte no.	Value	Description
0	0x66	16 Bit number of the packet: 358
1	0x01	
2	Data	1st Data byte in the packet
:	:	:
993	Data	Last data byte in the packet and block
994	LSB block counter	Consecutive block counter 32 bit
:	:	
997	MSB block counter	
998	LSB inverted block counter	Consecutive block counter 32 bit, inverted
:	:	
1001	MSB inverted block counter	
1002	0x78	1st Synchronization word
1003	0x56	
1004	0x34	
1005	0x12	
1006	0xF0	2nd Synchronization word
1007	0xDE	
1008	0xBC	
1009	0x9A	
1010	Signed Byte	Temperature in °C
1011	LSB GPS	Current value of the frequency correction in Hz as a
1012	MSB GPS, OV	signed 14-bit value, OV1, OV2
1013	Command no.	Number of the currently transmitted command block
1014	LSB command amount	32 bit count of following commands
:	:	
1017	MSB command amount	
1018	command	1st Byte (instruction) 1st command
:	:	:
1457	Data byte	Last possible byte for last command

2.2.2.2 UDP block structure at dual-channel mode 16 bits

UDP block 2-channel 16 bit: 718 packets = 1046844 bytes

Byte no.	Value	Description
0	1st Byte of the 1st packet	Begin 1st UDP packet
:	:	:
1457	1458th Byte of the 1st packet	End 1st UDP packet
:	:	:
1045386	1st Byte of the last packet	Begin 718th UDP packet
:	:	:
1046843	1458th Byte of the last packet	End 718th UDP packet

718th Packet in block 2-channel 16 bit

Byte no.	Value	Description
0	0xCD	16 Bit number of the packet: 717
1	0x02	
2	Data	1st Data byte in the packet
:	:	:
529	Data	Last data byte in the packet and block
530	LSB block counter	Consecutive block counter 32 bit
:	:	
533	MSB block counter	
534	LSB inverted block counter	Consecutive block counter 32 bit, inverted
:	:	
537	MSB inverted block counter	
538	0x78	1st Synchronization word
539	0x56	
540	0x34	
541	0x12	
542	0xF0	2nd Synchronization word
543	0xDE	
544	0xBC	
545	0x9A	
546	Signed Byte	Temperature in °C
547	LSB GPS	Current value of the frequency correction in Hz as a
548	MSB GPS, OV	signed 14-bit value, OV1, OV2
549	Command no.	Number of the currently transmitted command block
550	LSB command amount	32 bit count of following commands
:	:	
553	MSB command amount	
554	Command	1st Byte (instruction) 1st command
:	:	:
1457	Data byte	Last possible byte for last command

2.2.2.3 UDP block structure for single-channel mode 24 bits

UDP block 1-channel 24 bit: 539 packets = 785862 bytes

Byte no.	Value	Description
0	1st Byte of the 1st packet	Begin 1st UDP packet
:	:	:
1457	1458th Byte of the 1st packet	End 1st UDP packet
:	:	:
784404	1st Byte of the last packet	Begin 539th UDP packet
:	:	:
785861	1458th Byte of the last packet	End 539th UDP packet

539th Packet in block 1-channel 24 bit

Byte no.	Value	Description
0	0x1A	16 Bit number of the packet: 538
1	0x02	
2	Data	1st Data byte in the packet
:	:	:
33	Data	Last data byte in the packet and block
34	LSB block counter	Consecutive block counter 32 bit
:	:	
37	MSB block counter	
38	LSB inverted block counter	Consecutive block counter 32 bit, inverted
:	:	
41	MSB inverted block counter	
42	0x78	1st Synchronization word
43	0x56	
44	0x34	
45	0x12	
46	0xF0	2nd Synchronization word
47	0xDE	
48	0xBC	
49	0x9A	
50	Signed Byte	Temperature in °C
51	LSB GPS	Current value of the frequency correction in Hz as a
52	MSB GPS, OV	signed 14-bit value, OV1, OV2
53	Command no.	Number of the currently transmitted command block
54	LSB command amount	32 bit count of following commands
:	:	
57	MSB command amount	
58	Command	1st Byte (instruction) 1st command
:	:	:
1457	Data byte	Last possible byte for last command

3. Commands

To operate the RSR200, it is necessary to transmit command data in addition to IQ data. The RSR200 offers the following options for this purpose:

- Sending data from the RSR200 to the PC using individual TCP or UDP packets with streaming disabled.
- Sending command data embedded in the data blocks in streaming mode (always with USB) from the RSR200 to the PC.
- Sending individual USB, TCP, or UDP packets from the PC to the RSR200.

Individual packets can be sent to the RSR200 at any time. To do this, a corresponding connection must be established in the PC program and the command packet sent via this connection. For USB, the corresponding write commands of the FTDI API can be used. The data must be sent to endpoint 0x02. For LAN, a TCP socket (or a UDP socket, recommended for UDP streaming only) must be configured and the data sent via this socket. Usually, the same sockets that are required to receive data from the RSR200 can be used. Command packets must always be sent with exactly the specified length (32-bit word-wise for USB, byte-wise for LAN).

The RSR200 sends individual packets only via TCP (exception: version numbers on request also via UDP) and only when streaming is disabled. In all other cases, the command data is embedded in the IQ data blocks.

3.1 Handling of commands embedded in the data stream

Embedded commands must be extracted from the data blocks and processed as needed. The difficulty here is that data is constantly being transferred during streaming, which means that command data is also constantly being sent from the RSR200. However, the PC program must be able to recognize whether this data contains a command that needs to be processed or contains older data already processed. For this purpose, each block contains a command number. The process of command transmission from the RSR200 to the PC during streaming is as follows:

- A command is written by the RSR200 to the corresponding command data area of the blocks. This data area is sent to the PC each time the block is transmitted. This means that the same command data always arrives at the PC until the RSR200 writes new data to the data area.
- After writing command data, the RSR200 sets the command number to a new value (usually incremented by 1, but this is not guaranteed). A new command (to be processed by the PC) is considered valid when the PC program recognizes the new command number in the currently received block.
- At this point, the program must extract the command data belonging to the new number from the block and process it (at some point). The command data remains valid as long as the command number remains unchanged. This can be the case for many blocks, but the next block may already contain a new command with a new number.

A USB block always contains only one command. A LAN block, on the other hand, can contain several commands. The number of commands contained in the LAN block is specified in the 32-bit area “Command amount.” The PC program must process the individual commands in the command block one after the other until all of them have been processed. It can be important to adhere to the sequence!

Commands sent by the PC to the RSR200 must also always contain a command number (see section “Commands PC → RSR200”). The choice of this number is left to the PC program. It is used in the RSR200 exclusively for the creation of confirmation commands. It is used there to identify which received command is being confirmed.

3.2 Commands RSR200 → PC

All commands sent by the RSR200 are responses to commands that the PC has previously sent to the RSR200 (confirmations of command execution or of requested data). A command consists of a 4-byte header, followed by a command byte and a precisely defined number of data bytes for that command. The header is structured as follows.

Command header: 4 bytes

Byte no.	Value	Description
0	Signed Byte	Temperature in °C
1	LSB GPS	Current value of the frequency correction in Hz as a signed 14-bit value, OV1, OV2
2	MSB GPS, OV	
3	Command no.	Number of the currently transmitted command

The first byte contains the current temperature measurement (core temperature of the RSR200's FPGA) as a signed 8-bit value (range -128 to +127°C).

The second and third bytes contain a signed 14-bit value (LSB bit 0 to bit 7, MSB bit 8 to bit 13) for the current frequency correction of the RSR200. There are two possibilities (see command "Set ADC clock" from PC to RSR200):

- When internal frequency control is switched on, this value reflects the currently required correction value (control variable) for the control. It describes how far the ADC clock oscillator is from the target frequency and is therefore corrected by this value so that the target frequency is generated. The resolution is 0.5 Hz / LSB.

- When frequency control is switched off, this value indicates the deviation of the currently generated frequency from the target frequency (MEASUREMENT – TARGET). The resolution is 0.1 Hz / LSB.

Note: The internal frequency measurement and control uses the built-in GPS receiver (1pps signal) as the time base. If GPS reception is not possible, the highest possible negative value is output (0x2000). This value is also output if no valid value is available for other reasons (e.g., frequency has just been switched, but no new measured value is available yet) .

The two most significant bits of the third byte indicate the overload status of the ADC (bit 6: channel 1, bit 7: channel 2). A set bit indicates overload, a reset bit indicates operation in the linear range.

Note: The measured values for temperature and frequency correction, as well as the bits for the overload indicators, are updated in the data stream as they become available internally. The update rate is not fixed in relation to the ADC clock frequency, the block rate, or similar times determined by signal processing.

The fourth byte contains the number of the command currently being sent by the RSR200. As soon as this value changes from one block to the next, the rest of the command data contains a new command. Blocks with the same command number may or may not contain the same data/commands! **The change in command number represents the trigger signal for evaluating the command data of the block with the new number.**

The command header is followed by a specific number of bytes containing the instruction included in the command with the associated data. A USB block always contains only one command, which immediately follows the header. Commands are always sent via USB in streaming mode.

A LAN block can contain several commands. Therefore, the header is followed by a 4-byte field that specifies the number of commands as an unsigned 32-bit value (see LAN block data structure).

Individual LAN packets (only the "Report LAN version number" command if not in streaming mode) always contain only one command and therefore no counter before the command bytes.

Below is the structure of the various commands.

Confirmation: 8 bytes

Byte no.	Value	Description
0	0x00	Command: Acknowledgement
1	0x00	
2	0x00	
3	0x00	
4	LSB command received	32 Bit command no. of the received command,
:	:	the correct execution of which is hereby confirmed.
7	MSB command received	

General confirmation of the successful / correct execution of a command sent by the PC for which no data feedback is expected.

Special confirmation: 8 bytes

Byte no.	Value	Description
0	Command	Executed command
1	1st Data byte	Feedback data
2	2nd Data byte	
3	3th Data byte	
4	LSB command received	32 bit command no. of the received command,
:	:	the processing of which is hereby confirmed.
7	MSB command received	

Special confirmation of commands that expect data feedback. The first byte contains the command sent by the PC. The following 3 bytes contain additional information. This can be a simple “OK” (value 0) or “Not OK” (value not equal to 0) in the first byte, or the feedback of processed data (equal to the sent data: successfully processed, not equal to the sent data: data changed to this value).

Report version numbers USB: 8 bytes

Byte no.	Value	Description
0	0x12	Command: Read version numbers
1	LSB S/N	24 bit serial number of the device
2	:	
3	MSB S/N	
4	LSB Firmware	4 Digit hexadecimal value firmware version
:	:	
7	MSB Firmware	

This command is sent by the RSR200 in response to the request to transmit the version numbers via USB.

Caution! USB packets are only sent when USB streaming is turned on!

Report version numbers LAN: 12 bytes

Byte no.	Value	Description
0	LSB length	Length of the command: 12
:	:	0
3	MSB length	0
4	0x12	Command: Read version numbers
5	LSB S/N	24 bit serial number of the device
6	:	
7	MSB S/N	
8	LSB Firmware	4 Digit hexadecimal value firmware version
:	:	
11	MSB Firmware	

This command is sent by the RSR200 in response to the request to transmit the version numbers via LAN. The command is always sent as a single packet via the protocol through which the request for transmission was received (TCP or UDP). It does not contain the received command number and, unlike the other commands, is 12 bytes long.

3.2 Commands PC → RSR200

Commands sent from the PC to the RSR200 are used to set the RSR200 to different operating modes. It is also possible to transfer an FPGA configuration file (“firmware”) containing all information on the hardware configuration of the FPGA and the software of the integrated processors.

The commands are always sent as individual packets via USB, TCP, or UDP. In the case of LAN, unused bytes are not sent (package length is shortened accordingly). Confirmations are always sent back by the RSR200 via the interface through which the command was received. The RSR200 can process the following commands.

Command Reset: 8 bytes

Byte no.	Value	Description
0	LSB command no.	32 bit number of this command
:	:	
3	MSB command no.	
4	0xB2	Instruction: Reset
5	0x00	
6	0x00	
7	0x00	

This command causes a soft reset. The RSR200 loads firmware from flash and starts executing it.

Command Read version numbers: USB 8 bytes, LAN 6 bytes

Byte no.	Value	Description
0	LSB command no.	32 bit number of this command
:	:	
3	MSB command no.	
4	0x12	Instruction: Read version numbers
5	Byte	Repeat counter
6	n/a	No meaning / Not transmitted via LAN
7	n/a	No meaning / Not transmitted via LAN

The RSR200 responds by outputting the device version numbers (command “Report version numbers”).

Command Start stream: USB 8 bytes, LAN 7 bytes

Byte no.	Value	Description
0	LSB command no.	32 bit number of this command
:	:	
3	MSB command no.	
4	0x15	Instruction: Start streaming of the data
5	Port	Interface: 0 = UDP, 1 = TCP, 2 = USB
6	Size	Block length for LAN: 7 = 1-channel 16 bit, 15 = 2-channel 16 bit, other values = 1-channel 24 bit
7	n/a	No meaning / Not transmitted via LAN

The command starts streaming mode (continuous transmission of blocks) for the specified interface. The required block length for LAN must correspond to the current setting of the RSR200 (resolution and number of channels, see command “Set interface”).

This command is not acknowledged by the RSR200 with a confirmation command.

Note: USB streaming is automatically started by the RSR200 after each power-up / reset.

Note: If the resolution and/or channel number of the LAN interface are changed after the stream has started (command “Set interface”), a new command is required to restart the stream.

Command Stop stream: USB 8 bytes, LAN 7 bytes

Byte no.	Value	Description
0 : 3	LSB command no. : MSB command no.	32 bit number of this command
4	0x16	Instruction: Stop streaming of data
5	Port	Interface: 0 = UDP, 1 = TCP, 2 = USB
6	Byte	Repeat counter
7	n/a	No meaning / Not transmitted via LAN

Streaming mode of the specified interface is stopped.

This command is not acknowledged by the RSR200 with a confirmation command.

Caution! Stopping streaming via the USB interface closes the sender endpoint completely! No more data or commands can be sent from the RSR200 to the PC! Reception via USB (e.g., restarting the stream) is still possible.

Command Set ADC clock: 8 bytes

Byte no.	Value	Description
0 : 3	LSB command no. : MSB command no.	32 bit number of this command
4	0xF2	Instruction: Set value for ADC clock
5	LSB ADC-CLK	LSB ADC clock in 0.1 MHz
6	Bit 0 .. 6: MSB ADC-CLK Bit 7: GPS-Dis	MSB ADC clock Turn off GPS frequency control
7	Byte	Repeat counter

The command sets the clock frequency of the ADC circuits. The desired clock frequency must be within the possible range of the device (RSR200B: 700 … 2000). Bit 7 of byte no 6 “GPS-Dis” indicates the type of frequency control: Value = 0: Internal control on, different value: Control off.

Attention: In case the MSB of ADC-CLK is zero, the LSB is treated as a 1 MHz resolution value (range 70 … 200).

This command is acknowledged with a special confirmation. The actual clock frequency is specified in the first (LSB) and second (bit 0 .. 6 = MSB) data byte of the confirmation. The GPS Disable value is specified in bit 7 of the second byte.

Note: When the internal frequency control is switched on, the RSR200 outputs the current correction value with a resolution of 0.5 Hz; when the control is switched off, it outputs the current deviation between MEASUREMENT and TARGET with a resolution of 0.1 Hz.

Command Set frequency generators or IP address: USB 12 bytes, LAN 11 bytes

Byte no.	Value	Description
0	LSB command no.	32 bit number of this command
:	:	
3	MSB command no.	
4	0xB0	Instruction: Set the value for generators / IP
5	Channel	Set value: 0 = LO frequency channel 1 1 = LO frequency channel 2 2 = LO frequency both channels 9 = Magnitude and phase channel 2 10 = IP address 32 bit binary format other values: no effect
6	LSB data	LSB 32 bit value or LSB 16 bit magnitude
7	(MSB)	at magnitude: 16 bit value MSB
8	(LSB)	at phase: 16 bit value LSB
9	MSB data	MSB 32 bit value or MSB 16 bit phase
10	Byte	Repeat counter
11	n/a	No meaning / Not transmitted via LAN

The command sets the mixing oscillators of one or both channels or magnitude and phase of channel 2. It is also used to transmit a new IP address.

After execution, this command is acknowledged with a special confirmation. The first data byte contains the “Channel” parameter (byte 5), the second byte contains the success message (0 = OK, other value = not executed).

Notes: Frequencies are interpreted as signed 32-bit values with 1 Hz resolution. It is the responsibility of the sender to transmit suitable values. The RSR200 calculates the necessary data for setting the generators internally. Overflows (e.g., frequency higher than maximum ADC clock frequency) are programmed into the generators without correction.

The magnitude for influencing the second channel in diversity mode is interpreted as an unsigned 16-bit value (max. 0xFFFF = 8-1 LSB times amplification).

The phase for influencing the second channel in diversity mode is interpreted as a signed 16-bit value (0x8000 = -180° ... 0x7FFF = +180°-1 LSB).

The 32-bit value for an IPv4 address (e.g., obtained from the string using the inet_addr() function) is stored non-volatile in the RSR200. It is used as the default address from the next power-up / reset.

Command Set variable 16 bit value: USB 12 bytes, LAN 9 bytes

Byte no.	Value	Description
0	LSB command no.	32 bit number of this command
:	:	
3	MSB command no.	
4	0xF5	Instruction: Set value for 16 bit variable
5	Variable no.	Number of the variable to be changed
6	LSB data	LSB 16 bit value for variable
7	MSB data	MSB 16 bit value for variable
8	Byte	Repeat counter
9	n/a	No meaning / Not transmitted via LAN
10	n/a	No meaning / Not transmitted via LAN
11	n/a	No meaning / Not transmitted via LAN

The command is used to set various values that can be represented with a 16-bit (2-byte) data word. The possible settings are listed below.

Variable no.	Description	Permissible value range
0	CLK frequency correction in Hz	-3276.8 .. +3276.7 Hz
1	Attenuator ADC1	LSBits: 0 ... 35 (-7 .. +28 dB), bit 7 = 1: ADC2 as well
2	Attenuator ADC2	LSBits: 0 ... 35 (-7 .. +28 dB), MSBits: 0
3	Antenna control HF1/VHF	LSB 9 bit UART data word for antenna (see control units) +
4	Antenna control HF2	MSB 9 bit UART data word for antenna (see control units)
5	Switch	LSB: Bits for switch, MSB: 0 Bit 0: 0 = ADC2 CLK in phase, 1 = ADC2 CLK inverted Bit 1: 0 = ADC1 to HF1, 1 = ADC1 to VHF Bit 2: 0 = ADC2 to ADC1 (parallel), 1 = ADC2 to HF2 Bit 3: Remote power supply HF1/VHF, 0 = off, 1 = on Bit 4: Remote power supply HF1/VHF, 0 = +12 V, 1 = Control Bit 5: Remote power supply HF2, 0 = off, 1 = on Bit 6: Remote power supply HF2, 0 = +12 V, 1 = Control Bit 7: Preamplifier VHF, 0 = off, 1 = on
6	Antenna control HF1/VHF	LSB 16 bit UART data word for frequency (see control units) +
7	Antenna control HF2	MSB 16 bit UART data word for frequency (see control units)
8 .. 255	n/a	not defined

This command is acknowledged by the RSR200 with a special confirmation. The first data byte of the confirmation contains the variable number, the second (LSB) and third (MSB) contain the 16-bit value actually used in the RSR200 (possibly changed to permissible values).

Command Set data transmission: USB 12 bytes, LAN 9 bytes

Byte no.	Value	Description
0 : 3	LSB command no. : MSB command no.	32 bit number of this command
4	0xB4	Instruction: Set interface
5	Interface	1: USB, 2: LAN, 3: only set DSP, others: not defined
6	Port Mode	Byte for port mode, at interface = 3: 0
7	DSP Mode	Byte for DSP mode
8	Byte	Repeat counter
9	n/a	No meaning / Not transmitted via LAN
10	n/a	No meaning / Not transmitted via LAN
11	n/a	No meaning / Not transmitted via LAN

The command allows you to configure an interface in terms of the data rates, resolutions, and channel numbers used. The following options are available for the Port Mode and DSP Mode values.

1 Byte port mode

Bit number	Description	Permissible value range
0 1 2	3 bit decimation rate D	0 (rate = 2) .. 5 (rate = 64), others not allowed, Decimation rate = 2exp(D+1)
3	Channel selection	If bit 4 = 0: 0 = Data from ADC 1, 1 = from ADC 2 If bit 4 = 1: 0 = ADC1 at channel 1. ADC2 at channel 2 1 = ADC1 at channel 2, ADC2 at channel 1
4	Channel count	0 = single channel, 1 = dual channel
5	Bit width IQ data	0 = 24 bit, 1 = 16 bit (must always be with dual channel)
6	n/a	not defined
7	n/a	not defined

1 byte DSP mode

Bit number	Description	Permissible value range
0 1	2 bit operation mode	0 = ADC1 and ADC2 independent (port mode bit 4 must be 1!) 1 = ADC1 + ADC2 parallel (Addition of Data) 2 = ADC1 + ADC2 serial (CLK ADC2 must be inverted!) 3 = ADC1 + ADC2 parallel (Diversity)
2	n/a	0
3	Side band	0 = lower side band for serial, 1 = upper side band
4	n/a	0
5	n/a	0
6	n/a	0
7	n/a	0

This command is acknowledged by the RSR200 with a special confirmation. The first data byte of the confirmation contains the acknowledgment. If the acknowledgment is 0, the interface continues to run immediately with the changed settings. If the acknowledgment is not 0, the interface must be closed, reinitialized, and reconnected.

Caution! When switching the mode for the LAN interface, the RSR200 automatically stops streaming mode (if enabled) and sets the interface to the requested data. Streaming mode must then be restarted with a separate “Start stream” command, specifying a block size that matches these settings. As long as the stream is switched off, no confirmation can be received (it is included from the first block of the new stream).

3.3 Notes on RSR200 operation

After switching on or a reset (soft reset or serious error), the RSR200 is in the following default state:

- ADC CLK is undefined, approximately 125 MHz
- Operating mode: Parallel (ADC1 + ADC2)
- Decimation rates: 16
- Ports: Single-channel 24-bit, data to USB (switched to streaming mode)
- Adjustable values (frequencies, attenuators, etc.) set to zero or minimum value
- Inputs: HF1 to ADC1 and ADC2, remote power supplies / antenna control devices off
- Frequency correction via GPS active

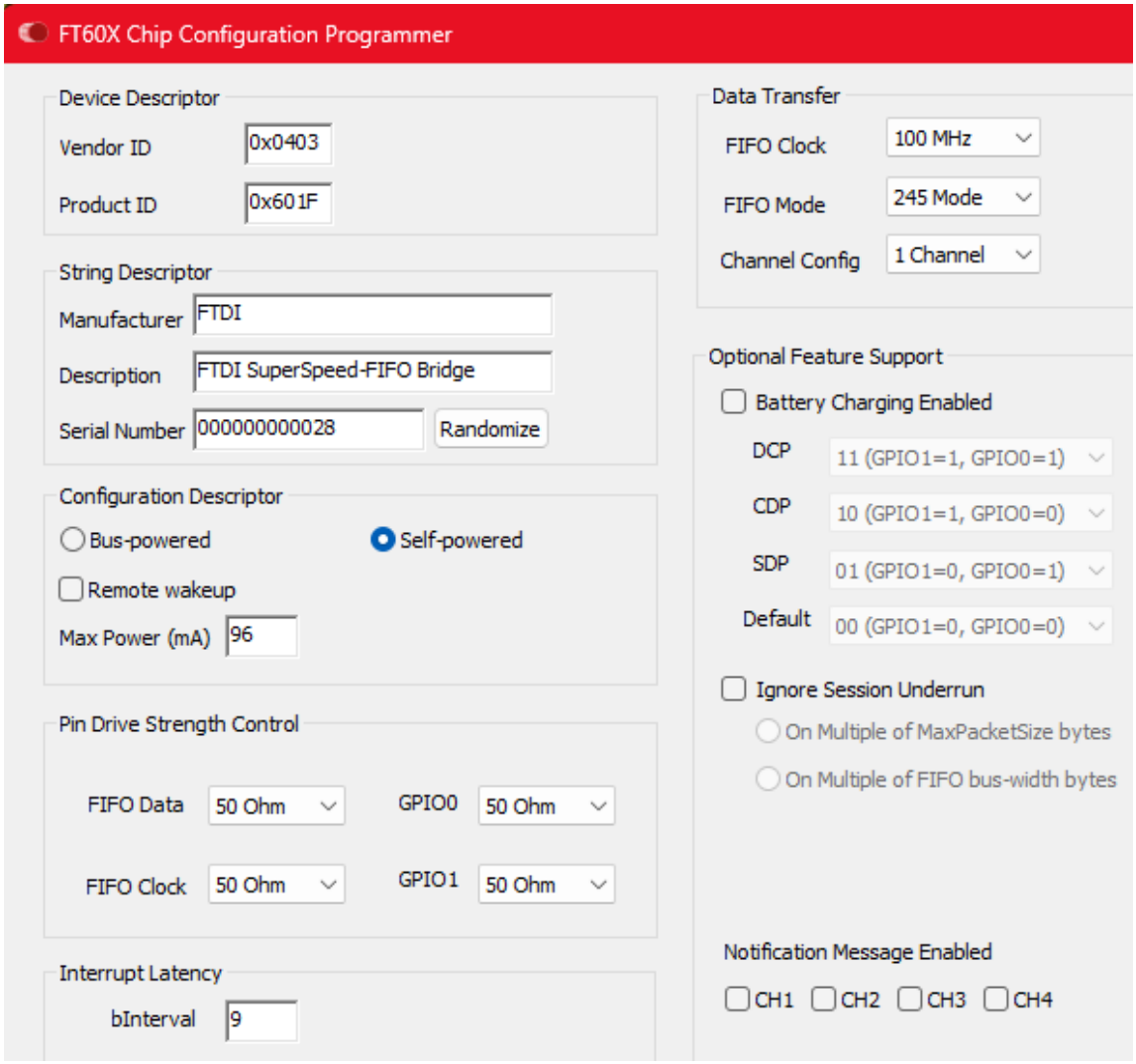
Except for the IP address, no set values are permanently stored in the RSR200.

3.3.1 Using the USB interface

The buffer memory of the USB interface begins to fill immediately after the device is initialized. The command area is initialized with command no. 0 and the command areas are undefined. As soon as measured values (temperature and GPS) are available, they are continuously written to the block. A change to these values is not a command and therefore does not result in a change to the command no.

Each change to the command number represents the transmission of a new command. The data of the current command remains valid (across any number of packets) until a new command number is sent.

To establish a connection from the PC to the RSR200, the FTDI API offers various options (preferably “FT_OPEN_BY_DESCRIPTION”).The following image shows the configuration of the FT601Q. For use with the serial number, this setting can be changed as desired (preferably to the serial number of the RSR200). **All other parameters must not be changed!**



The following procedure is recommended for using the USB interface.

Switch on device / reset
|
Wait approx. 3 seconds
|
Initialize USB port according to FTDI API (create port handler)
|
Read 4096 bytes from endpoint 0x82 (check for success), note command no.
|
Send command “Read version numbers” to endpoint 0x02 (8 bytes)
|
Continuously read 4096 bytes from endpoint 0x82, check command no.
|
If the number changes, check whether the “USB version numbers” command was received. Note/display version numbers as needed.

However, the IQ data from the USB blocks can be used safely here. To do this, the program must continuously receive all data from the RSR200 and process it according to the block structure. Commands with the desired settings can be sent to the RSR200 independently of this. The following back and forth sequence of commands and confirmations is recommended:

Generate command number (e.g., continuous 32-bit counter)
|
Compile data for command and send packet to RSR200
|
Wait for confirmation (if required for command) from RSR200 (block processing)
|
Evaluate confirmation: Reception OK or error, check feedback data (processed as desired or changed by RSR200), etc.
|

Generate and send next command.

Commands for which no individual confirmation is to be evaluated (however, this is always recommended!) can be sent as a command block with a common command number. The individual commands must follow each other without gaps in the transmission data. Due to the fixed command lengths, the RSR200 can read and process all commands one after the other. The maximum block length is 264 bytes.

After sending a command block, you must wait for confirmation of the last command (which must be one requesting confirmation) before new commands can be sent.

3.3.2 Using the LAN interface with TCP

After switching on / resetting, the RSR200 attempts to obtain an IP address (IPv4) from a DHCP server for approx. 5 seconds. If this is successful, this address is used for the duration of the device's operation. If not, the address stored in the device is used (factory setting 192.168.1.10).

The RSR200 configures a TCP server that waits for incoming connections on port 55557. A valid request leads to the establishment of the connection. Only one connection to a client is possible. Once the connection is established, the client can send commands to the RSR200. As long as the TCP server is not switched to streaming mode, it responds with individual packets only to the request to output the LAN version numbers.

Upon receiving the “Start stream” command for TCP, the server begins sending LAN blocks. The command must specify the correct size of the blocks according to the interface setting (execute beforehand using the “Set interface” command). The client can receive the blocks in their entirety and process the data and commands they contain.

The blocks must be checked for correct data arrangement at the block boundaries and arranged correctly if necessary. The block counters and synchronization data available in each block can be used for this purpose.

At the end of each block, there is space for commands from the RSR200 to the TCP client, depending on the block size. Command blocks of various sizes may be present in this space. The number of commands contained is specified in the block. The client must evaluate the block as soon as it detects a change in the command number. The data (commands) in the block are only valid for this block. Subsequent blocks with the same command number may already contain other (new, still invalid / incomplete) data!

The following procedure is recommended for using the LAN interface with TCP.

Switch on device / reset
|
Wait approx. 6 seconds (RSR200 receives address via DHCP or sets the internally stored address).
|
Configure a TCP socket for IP data (“AF_INET”) and socket streaming, as well as port 55557 and IP address of the RSR200.
|
Establish connection to the RSR200’s TCP server.
|
Send command “Read version numbers” to the server (6 bytes)
|
Receive response “Version numbers LAN” (12 bytes). Note / display version numbers as needed.

The PC can now send further commands to the RSR200 via the TCP socket. To receive data from the RSR200, the command “Start stream” must be sent for TCP. The interface must first be configured with the “Set data transmission” command. After that, processing of the blocks can begin. Command processing should be carried out according to the back and forth principle as described for USB (Command → Confirmation → Next command...). If commands will be sent faster than one LAN block can be received due to sampling rate, the confirmations will be bundled in the next LAN block (“Command Amount” will be larger than 1).

Streaming mode can be interrupted at any time by the command “Stop stream” (7 bytes). When restarting with “Start stream,” the correct block size must be used (may have changed in the meantime), otherwise the RSR200 will send blocks with the wrong length.

3.3.3 Using the LAN interface with UDP

Data transfer with UDP packets may enable a higher data rate (no acknowledgments from the PC required, no streaming interruptions in the event of faulty packets). However, data errors or losses may occur that cannot be detected or corrected. UDP operation is therefore only recommended for the data direction RSR200 → PC. The RSR200 sends commands in each block, whereby a command is usually repeated many times. A complete loss of a command is therefore very unlikely. Minor losses of IQ data are often acceptable.

The RSR200 can receive UDP packets on port 55558 under its current IP address (see TCP configuration). This is only recommended for initializing the connection using the “Read version numbers” command. The RSR200 extracts the sender's data from the UDP packet and from then on always sends UDP packets (if enabled as stream) to this UDP partner. All other commands during operation should be sent from the PC via TCP.

The following procedure is recommended for using the LAN interface with UDP.

Switch on device / reset
|
Wait approx. 6 seconds (RSR200 receives address via DHCP or sets the internally stored address).
|
Configure a TCP socket for IP data (“AF_INET”) and socket streaming, as well as port 55557 and IP address of the RSR200.
|
Establish connection to the RSR200’s TCP server.
|
Send ”Read version numbers” command to TCP server (6 bytes)
|
Send ”Read version numbers” command via UDP to port 55558 (6 bytes)
|
Receive ”LAN version numbers” responses (12 bytes each). Compare/note/display version numbers as needed.

The PC can now send further commands to the RSR200 via UDP or, preferably, TCP (more secure!). To receive data from the RSR200 via UDP, the “Start stream” command must be sent as a UDP packet. The interface must first be configured with the “Set data transmission” command. After that, processing of UDP data can begin. The UDP packets should be received individually and assembled into the final LAN block.

UDP stream operation can be aborted at any time using the “Stop stream” command (7 bytes). The stream can be restarted using either TCP or UDP.

3.3.4 Command repeats

The PC → RSR200 commands contain a byte labeled “Repeat counter.” The default value is 0.To increase transmission reliability, the PC program can check whether a command sent to the RSR200 was executed within an acceptable time or not (loss of the command packet, error in the RSR200, loss of confirmation, etc.). If an expected confirmation for a command does not arrive within a reasonable time, the command can be repeated one or more times. For a repeated command, the command number should be retained and only the repeat counter should be incremented.

When receiving repeated commands, the RSR200 can check whether it has already executed this command (and only the PC has not received the confirmation). In this case, the command is not executed again; only a new confirmation is sent.

Note: In firmware version 220, the RSR200 does not check the repeat counter. New commands should be sent instead of repeating commands. The consequences of any repeated execution of the same commands must be taken into account.

3.3.5 Antenna control

The RSR200 contains two RSW control units for controlling active antennas. The control unit for channel 1 is connected either to input HF1 or to input VHF (if activated), depending on the channel connection to the respective input. Control unit 2 is always connected to HF2 (if activated).

Each control unit can either output a fixed voltage of 12 V (pure remote power supply for any antenna) or an RS-232 modulated voltage for controlling special antennas. The data for controlling the antennas can be transferred using the command “Set 16-bit variable value”. Depending on the antenna to be controlled, the data must correspond to the format and value range expected by the antenna. Various variables are available for the command (e.g., variables 6 and 7 for directional control of an RLA4 / 5 on HF1 / VHF and HF2). The required data can be found in the descriptions of the antennas or the RSW2, 3, or 4 control units.

Note: When activating the control units, the supply voltage of the RSR200 must be at least 0.5 V above the remote supply voltage. The current for supplying the antenna(s) is taken directly from the supply voltage via analog voltage regulators. This leads to additional heating of the RSR200 in accordance with the generated power loss! The supply voltage of the RSR200 should therefore not be much higher than the remote supply voltage + 0.5 V. It should be well stabilized and low in noise / interference.

3.3.6 Using the GPS receiver

The RSR200 has an integrated GPS receiver that outputs a 1pps signal (1 pulse per second). The accuracy of the pulse is determined by the atomic clocks of the received satellites and the inherent sources of error in data transmission and signal processing in the receiver. It is specified as better than 15 ppm for the receiver used, but is typically well below 1 ppm when reception is good. This signal is therefore suitable as a time base for accurate measurement and possible correction of the ADC clock frequency.

Other data such as position determinations or similar from the GPS receiver are not processed.

The RSR200’s ADC clock frequency is variable (command “Set ADC clock”). The clock oscillator used is controlled by a digital word. This data word can be used to compensate for deviations of the oscillator frequency from the set target frequency.

To do this, the RSR200 measures the oscillator frequency using the 1 pps signal as a time base in two different intervals:

- 2-second interval (two 1 pps pulses): Resolution 0.5 Hz.
- 10-second interval (ten 1 pps pulses): Resolution 0.1 Hz.

Using the “Set ADC clock” command (GPS-Dis parameter), the RSR200 can be switched to two different operating modes for using this measurement data.

- Internal control: The measurement data is used to calculate a correction value, which is used to retune the clock oscillator by exactly this value so that it operates at the target frequency. The correction value therefore represents the current deviation of the free-running oscillator without correction. It is output in the data blocks at 2-second intervals (0.5 Hz resolution).
- No control: The oscillator runs freely without being corrected independently by the RSR200. The measurement of the current deviation (actual value minus target point) is output in the data blocks at 10-second intervals (0.1 Hz resolution).

The RSR200 accepts correction data for the clock oscillator at any time using the command “Set 16-bit variable value” (variable no. 0) in 0.1 Hz resolution. Together with the evaluation of the measurement data sent by the RSR200, this can be used to set up an external (PC-controlled) control system.

Note: The readjustment of the clock oscillator by means of a digital control word causes a frequency jump with each new setting, corresponding to the difference between the previous setting and the new setting. This jump can be noticeable in the signal and cause interference. Readjustments of the clock oscillator should therefore only be made in very small steps (unless a large deviation, e.g., after a new frequency setting, needs to be corrected quickly).

3.3.7 Firmware update

Attention! The use of commands to update the device's firmware should only be attempted if it is possible to reprogram the original firmware at any time in accordance with the manufacturer's specifications. This requires a “Xilinx Platform Cable USB II” emulator connected to the internal programming port of the RSR200, as well as Xilinx development software and an original firmware

file (“golden image”) from the RSR200 manufacturer!

The firmware of the RSR200 consists of the hardware configuration of the FPGA and the software for the processors integrated in the FPGA. Both can be reprogrammed together into the RSR200's internal flash memory via a single binary file.

There are always two images of this binary file in the flash memory:

- A version image that can be deleted and reloaded via an interface (firmware that is loaded and executed after power-up / reset).
- A “golden image” that can only be changed via the internal programming port (reserved file in case the version file is not available or damaged).

The “golden image“ is permanently burned into the device by the manufacturer. It is essentially an older, well-tested, and reliable version of the firmware. It enables the device to start up if no valid version file is loaded (check the version of the “golden image“ and use the corresponding PC software!).

The version image (usually with the extension “.Bit”) can be deleted and re-programmed via the data connection from the PC. The following commands are available for this purpose.

Command Enable firmware update: USB 12 bytes, LAN 9 bytes

Byte no.	Value	Description
0	LSB command no.	32 bit number of this command
:	:	
3	MSB command no.	
4	0x0B	Instruction: Enable update
5	LSB S/N	Serial number of RSR200
6	:	
7	MSB S/N	
8	Byte	Repeat counter, not analyzed
9	n/a	No meaning / Not transmitted via LAN
10	n/a	No meaning / Not transmitted via LAN
11	n/a	No meaning / Not transmitted via LAN

This command enables the update process. The RSR200 responds with the special confirmation containing the same data (serial number). If the numbers doesn’t match the process remains disabled. This command must be followed by the “Start firmware update” command. Any other command will disable the update process.

Command Start firmware update: USB 12 bytes, LAN 9 bytes

Byte no.	Value	Description
0	LSB command no.	32 bit number of this command
:	:	
3	MSB command no.	
4	0x0C	Instruction: Start update
5	LSB length	Length of bit file in bytes
6	:	
7	MSB length	
8	Byte	Repeat counter, not analyzed
9	n/a	No meaning / Not transmitted via LAN
10	n/a	No meaning / Not transmitted via LAN
11	n/a	No meaning / Not transmitted via LAN

The length of the binary file to be transferred must be specified in bytes.

After receiving this command, the RSR200 deletes the version file. When the deletion process is complete, it sends a special confirmation with 0 = OK or 4 = error in the first data byte of the confirmation. If OK, it expects the data from the version file from this point on. If there are errors, the process is aborted.

Caution! After an aborted firmware update, the status of the version file is undefined. Restarting the

RSR200 could succeed with the start of the “Golden image“ or get stuck in the damaged version file. Therefore, aborting the update should always end with the successful deletion or transfer of the version image. If necessary, restart the update or at least the deletion before resetting the device and execute it successfully!

Command Data for firmware update: USB 264 bytes, LAN 263 bytes

Byte no.	Value	Description
0 : 3	LSB command no. : MSB command no.	32 bit number of this command
4 5	0x0D Packet number	Instruction: Data update Continuous 7 bit packet counter, bit 7 always 1
6 : 261	1st Byte of the packet : 256th Byte of the packet	Continuous data from the bit file
262 263	Byte n/a	Repeat counter No meaning / Not transmitted via LAN

This command transfers a block of 256 bytes of the binary file. The RSR200 stores the data in flash memory and acknowledges with a special confirmation.

The first data byte of the confirmation contains the packet number (= packet OK) or error code 4 (error during transmission/saving, abort!) or code 0 (last packet, everything OK). The second (LSB) and third (MSB) data bytes contain a 16-bit checksum. It is generated by continuously adding all bytes of the binary file received so far in binary form (overflow 16 bits truncated). It can be checked on the PC against a similarly generated checksum to ensure data integrity.

Notes: **Caution! See command “Start firmware update“.**

The transfer of the firmware bytes must start at **the end** of the file and proceed backwards to the beginning! The RSR200 always expects full packets of 256 bytes (with the bytes in ascending order within the packet). Therefore, the first packet must be filled with empty bytes (0xFF) at the end if the binary file is not exactly a multiple of 256 bytes long. The last packet must contain the 1st to 256th byte of the binary file.

The confirmations for the packets are sent as commands in the data stream as usual. Therefore, streaming mode must be enabled. To relieve the PC of unnecessary work (and thus increase the security of data transmission), no processing of IQ data should take place during the update.

After a successful update, the RSR200 continues to operate normally. Only after a restart (switching on / reset), the new version file is loaded and executed.