TCP commandsto remote management of FDM-S1/S2/DUO devicesbased on the new FDMSW2 TCP server ver 0.11.

Any application client can set up a communication with FDMSW2's server through a TCP connection or Web HTTP connection. FDMSW2's web http server is active on any IP address.

The client can manage FDM device in remote mode by sending string commands. The commands are the same for both the TCP and HTTP servers. Here there is an index of all command implemented:

Command Description

SR	Get/Toggleselected virtual receiver state on FDM device										Parameters: *P1(input) Index for channels/data-streams on FDM (1 digit) Index is coded with 1 char ranges from '0' char to channel number-1 char. *P2(input) Index for virtual receiver on FDM (1 digit) Index is coded with 1 char ranges from '0' char to maximum virtual receiver number-1 char *P3(input) Set to '1' to toggle virtual receiver state. Any other char value does nothing. *P3(output) State of selected virtual receiver(1 digit). State is coded with 1 char ranges from '0' to '2':
Set	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	
	S	R	P1	P2	P3='1'	;					
Get	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	
	S	R	P1	P2	;						
Answer	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	
	S	R	P1	P2	P3	;					

											‘0’: receiver is off ‘1’: receiver is on, but not active ‘2’: receiver is on and active

SR command modifies or readseach virtual receiver state on a specified data-stream. The maximum virtual receiver number is 4 for each data channel, then receiver index ranges from 0 to 3.

The change of virtual receiver state is very important because a selected receiver must be active to do:

- The change of receiver locked to central frequency state (see LF set command description)
- The change of receiver demodulation (see MD set command description)

Besides, a selected receiver must be on but it can be not active to do:

- The correct reading of S-meter value (see SM and RX commands descriptions)

To modify state on a selected virtual receiver, the string “SRP1P21;” must be sent to TCP server: P1 is the index value for data channel on FDM and P2 is the index value for selected receiver. Then:

- If P2 receiver on P1 data channel is off (state code is 0) => the command switches P2 receiver to on and active state
- If P2 receiver on P1 data channel is on (state code is 1) => the command switches P2 receiver to on and active state
- If P2 receiver on P1 data channel is on and active (state code is 2) => the command switches P2 receiver to off state

If a selected receiver is on and active (state code is 2), when a SR command is sent to server to set a new active receiver, the old selected receiver is set to on state (state code is 1) because only one receiver can be in active state at the same time.

<u>Examples for ‘Get’/’Set’</u>

SR command acts as RX1, RX2, RX3, RX4 buttons on FDMSW2 user interface, sofor example it will be compared the FDMSW2 user interface operations and the equivalent TCP interface commands.

The default states at power on for each virtual receiver are (see at Figure 1a and 1b):

- Receiver number 1 is on and active => its state value is 2
- Receiver number 2, 3, 4 are off => their state values are 0

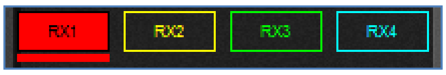


Figure 1a: Virtual receiver 1 selected and active.

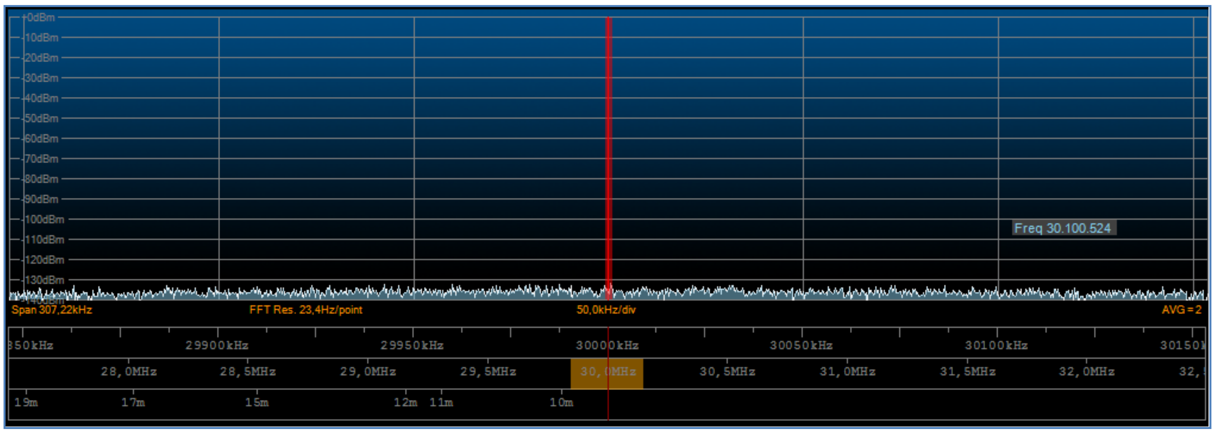


Figure 1b: Virtual receiver 1 active,displayed on spectrum.

In Figures 1a and 1b there is FDMSW2 user interface configuration for virtual receiver states. To know the same states configuration by SR get command (supposing to be on first data stream), it must be sent:

First channel, first virtual receiver:

Get-string to server TCP: "SR00;"

Answer-string from server TCP: "SR002;"=> '2' value means the receiver 1 is on and active

First channel, second virtual receiver:

Get-string to server TCP: "SR01;"

Answer-string from server TCP: "SR010;"=> '0' value means the receiver 2 is off

First channel, third virtual receiver:

Get-string to server TCP: "SR02;"

Answer-string from server TCP: "SR020;"=> '0' value means the receiver 3 is off

First channel, fourth virtual receiver:

Get-string to server TCP: "SR03;"

Answer-string from server TCP: "SR030;"=> '0' value means the receiver 4 is off

To change active receiver to the number 3, click on RX3 button of FDMSW2 user interface, and all states will change as (see Figure 2a and 2b):

- Receiver number 3 is on and active => its state value is 2
- Receiver number 1 is on and not active => its state value is 1
- Receiver number 2, 4 are off => their state values are 0

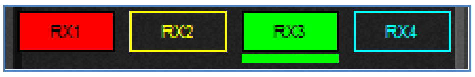


Figure2a: Virtual receiver 3 selected and active, virtual receiver 1 on.

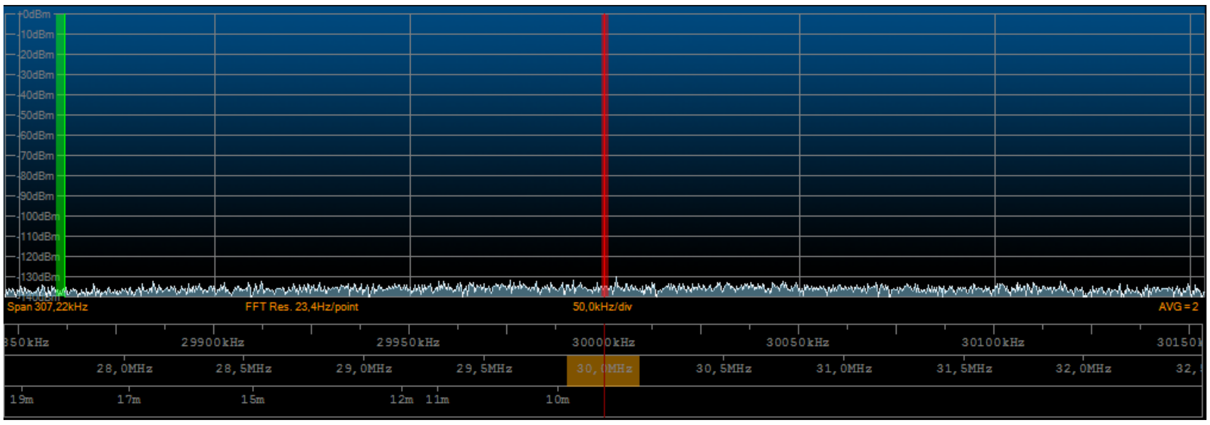


Figure 2b: Virtual receiver 3 active, virtual receiver 1 on, displayed on spectrum.

To use SR set command in order to change virtual receiver number 3, it must be sent (supposing to be on first data stream):

First channel, third virtual receiver:

Set-string to server TCP: "SR021;"=> '1' value toggles the third receiver state to on and active and puts the first receiver to on and not active state

Answer-string from server TCP: "SR021;"=> '1' value has no mean in this case

Now, to know all states, SR get commands are:

First channel, first virtual receiver:

Get-string to server TCP: "SR00;"

Answer-string from server TCP: "SR001;"=> '1' value means the receiver 1 is on

First channel, second virtual receiver:

Get-string to server TCP: "SR01;"

Answer-string from server TCP: "SR010;"=> '0' value means the receiver 2 is off

First channel, third virtual receiver:

Get-string to server TCP: "SR02;"

Answer-string from server TCP: "SR022;"=> '2' value means the receiver 3 is on and active

First channel, fourth virtual receiver:

Get-string to server TCP: "SR03;"

Answer-string from server TCP: “SR030;”=> ‘0’ value means the receiver 4 is off

To change active receiver to the number 2, click on RX2 button on FDMSW2 user interface, and all states will change as (see Figure 3a and 3b):

- Receiver number 2 is on and active => its state value is 2
- Receiver number 1 is on and not active => its state value is 1
- Receiver number 3 is on and not active => its state value is 1
- Receiver number 4 is off => its state value is 0

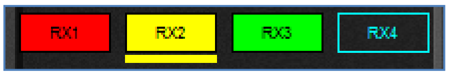


Figure3a: Virtual receiver 2 selected and active, virtual receiver 1 and 3 on.

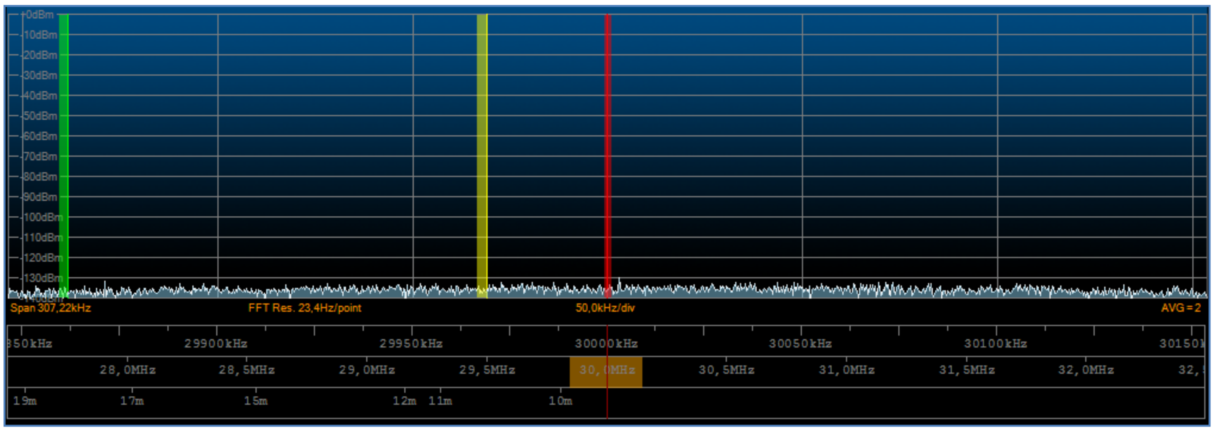


Figure 3b: Virtual receiver 2 active, virtual receiver 1 and 3 on, displayed on spectrum.

To use SR set command in order to change virtual receiver number 2, it must be sent (supposing to be on first data stream):

First channel, second virtual receiver:

Set-string to server TCP: “SR011;”=> ‘1’ value toggles the second receiver state to on and active and puts the third receiver to on and not active state

Answer-string from server TCP: “SR011;”=> ‘1’ value has no mean in this case

Now, to know all states, SR get commands are:

First channel, first virtual receiver:

Get-string to server TCP: “SR00;”

Answer-string from server TCP: “SR001;”=> ‘1’ value means the receiver 1 is on

First channel, second virtual receiver:

Get-string to server TCP: “SR01;”

Answer-string from server TCP: “SR012;”=> ‘2’ value means the receiver 2 is on and active

First channel, third virtual receiver:

Get-string to server TCP: “SR02;”

Answer-string from server TCP: “SR021;”=> ‘1’ value means the receiver 3 is on

First channel, fourth virtual receiver:

Get-string to server TCP: “SR03;”

Answer-string from server TCP: “SR030;”=> ‘0’ value means the receiver 4 is off

To turn off the number 3 receiver, it must be on and active: click on RX3 buttonto activate receiver (see Figure 4) and click on RX3 buttonto turn it off (see Figure 5a and 5b). States will change as:

First click on RX3 button

- Receiver number 3 is on and active => its state value is 2
- Receiver number 1 is on and not active => its state value is 1
- Receiver number 2 is on and not active => its state value is 1
- Receiver number 4 is off => its state value is 0

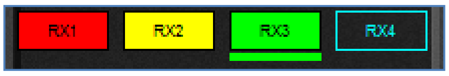


Figure 4: Virtual receiver 3 selected and active, virtual receiver 1 and 2 on (spectrum is the same as in Figure 3b).

To use SR set command in order to change virtual receiver number 3 state, it must be sent (supposing to be on first data stream):

First channel, third virtual receiver:

Set-string to server TCP: “SR021;”=> ‘1’ value toggles the third receiver state to on and active and puts second receiver to on and not active state

Answer-string from server TCP: “SR021;”=> ‘1’ value has no mean in this case

Now, to know all states, SR get commands are:

First channel, first virtual receiver:

Get-string to server TCP: “SR00;”

Answer-string from server TCP: “SR001;”=> ‘1’ value means the receiver 1 is on

First channel, second virtual receiver:

Get-string to server TCP: “SR01;”

Answer-string from server TCP: “SR011;”=> ‘1’ value means the receiver 2 is on

First channel, third virtual receiver:

Get-string to server TCP: “SR02;”

Answer-string from server TCP: “SR022;”=> ‘2’ value means the receiver 3 is on and active

First channel, fourth virtual receiver:

Get-string to server TCP: “SR03;”

Answer-string from server TCP: “SR030;”=> ‘0’ value means the receiver 4 is off

Second click on RX3 button

- Receiver number 3 is off => its state value is 0
- Receiver number 1 is on and active => its state value is 2
- Receiver number 2 is on and not active => its state value is 1
- Receiver number 4 is off => its state value is 0

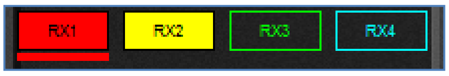


Figure 5a: Virtual receiver 3off, virtual receiver 1 selected and active, virtual receiver 2 on.

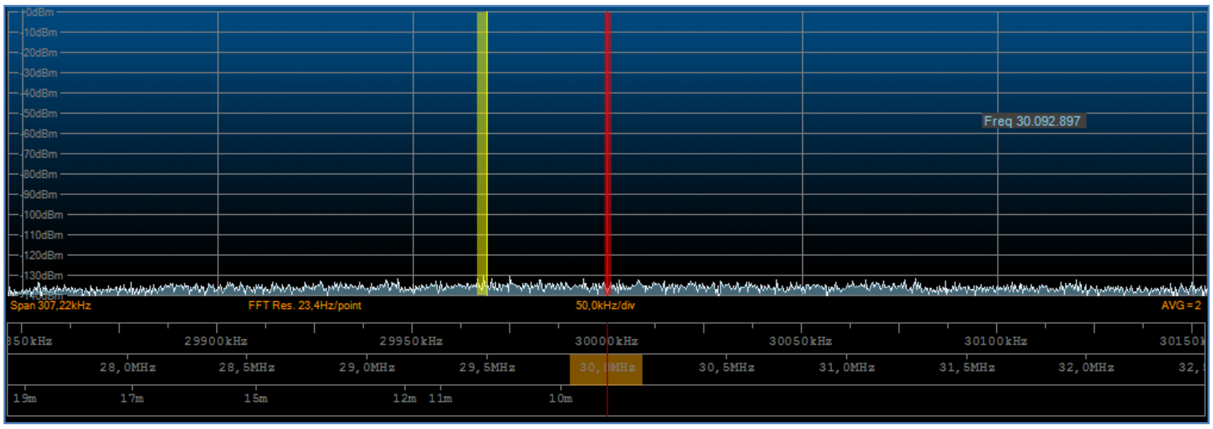


Figure 5b: Virtual receiver 1 active, virtual receiver 2 on, displayed on spectrum.

To use SR set command in order to change virtual receiver number 3, it must be sent (supposing to be on first data stream):

First channel, third virtual receiver:

Set-string to server TCP: “SR021;”=> ‘1’ value toggles the third receiver state to offand puts first receiver to on and active state

Answer-string from server TCP: “SR021;”=> ‘1’ value has no mean in this case

Now, to know all states, SR get commands are:

First channel, first virtual receiver:

Get-string to server TCP: "SR00;"

Answer-string from server TCP: "SR002;"=> '2' value means the receiver 1 is on and active

First channel, second virtual receiver:

Get-string to server TCP: "SR01;"

Answer-string from server TCP: "SR011;"=> '1' value means the receiver 2 is on

First channel, third virtual receiver:

Get-string to server TCP: "SR02;"

Answer-string from server TCP: "SR020;"=> '0' value means the receiver 3 is off

First channel, fourth virtual receiver:

Get-string to server TCP: "SR03;"

Answer-string from server TCP: "SR030;"=> '0' value means the receiver 4 is off

---

CF	Get/Set Central frequency on FDM device										Parameters: *P1(input) Index for channels/data-streams on FDM (1digit) Index is coded with 1 char ranges from ‘0’ char to channel number-1 char. *P2(input) Unused for this command and set to ‘0’ *P3(input/output) Central Frequency (11 digits) isan unsigned integer value expressed in Hz. Central Frequency is coded with 11 chars, one char for each digit of frequency value.
Set	***1***	***2***	***3***	***4***	***5***	***6***	***7***	***8***	***9***	***10***	
	C	F	P1	P2	P3	P3	P3	P3	P3	P3	
	11	***12***	***13***	***14***	***15***	***16***	***17***	***18***	***19***	***20***	
	P3	P3	P3	P3	P3	;					
Get	***1***	***2***	***3***	***4***	***5***	***6***	***7***	***8***	***9***	***10***	
	C	F	P1	P2	;						
Answer	***1***	***2***	***3***	***4***	***5***	***6***	***7***	***8***	***9***	***10***	
	C	F	P1	P2	P3	P3	P3	P3	P3	P3	
	11	***12***	***13***	***14***	***15***	***16***	***17***	***18***	***19***	***20***	
	P3	P3	P3	P3	P3	;					

CF command modifies or reads local oscillator frequency on FDM, or more local oscillator frequencies if FDM hardware configuration has more than one data-stream. For FDM-DUO central frequency is the same as VFOA frequency, thenthis command is used to get or set VFOA frequency.

Examples for ‘Set’

Example: Setting 1170000 Hz central frequency on FDM configuration at 192kHz with one channel or data-stream:

Set-string to server TCP: “CF0000001170000;”

Answer-string fromserver TCP: “CF0000001170000;”

Example: Setting 1170000 Hz central frequency on FDM configuration at 384kHz with twochannels or data-streams:

First channel

Set-string to server TCP: “CF0000001170000;”

Answer-string from server TCP: “CF0000001170000;”

Second channel

Set-string to server TCP: “CF1000001170000;”

Answer-string from server TCP: “CF1000001170000;”

=> P1 value ‘0’ means first channel;

=> P1 value ‘1’ means second channel;

=> P2 value is always ‘0’, unused for this command.

Examples for ‘Get’

Example: Getting 1170000 Hz central frequency on FDM configuration at 192kHz with one channel or data-stream:

Get-string to server TCP: “CF00;”

Answer-string from server TCP: “CF000001170000;”

Example: Getting 1170000 Hz central frequency on FDM configuration at 384kHz with two channels or data-streams:

First channel

Get-string to server TCP: “CF00;”

Answer-string from server TCP: “CF000001170000;”

Second channel

Get-string to server TCP: “CF10;”

Answer-string from server TCP: “CF100001170000;”

=> P1 value ‘0’ means first channel;

=> P1 value ‘1’ means second channel;

=> P2 value is always ‘0’, unused for this command

LF	Get/Setlocked stateon selected virtual receiver of FDM device										Parameters: *P1(input) Index for channels/data-streams on FDM (1 digit) Index is coded with 1 char ranges from '0' char to channel number-1 char. *P2(input) Index for virtual receiver on FDM (1 digit) Index is coded with 1 char ranges from '0' char to maximum virtual receiver number-1 char *P3(input/output) Locked state value.It is coded with 1 char equal to: '0': virtual receiver is in unlocked state '1': virtual receiver is locked to central frequency '2': virtual receiver is locked to absolute frequency
Set	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	
	L	F	P1	P2	P3	;					
Get	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	
	L	F	P1	P2	;						
Answer	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	
	L	F	P1	P2	P3	;					

LF command modifies or reads locked statefor a selected virtual receiver on a specified data-stream. The maximum virtual receiver number is 4 for each data channel, then receiver index ranges from 0 to 3. This command acts as Lock To CF button or Lock ABS button in FDMSW2 user interface.

LF set-command modifies the locked state only if selected virtual receiver is active; then, before using this command, get information about the state for the virtual receiver and modify it as active if you want to change locked to central frequency state (see SR command).

To set an active receiver as locked to CF or locked to absolute frequency, make sure that it is in unlocked state before locking; if it isn't, set it in unlocked state.

Locked state can be:

- Unlocked: receiver is not locked to central frequency and it is not locked to absolute frequency; P3 parameter in LF-set command and LF-answer is '0';
- Locked to Central Frequency: receiver is locked to central frequency and P3 is '1';
- Locked to Absolute Frequency: receiver is locked to absolute frequency and P3 is '2';

Examples for 'Set'

Example: Setting locked to central frequency state for active virtual receiver of the first channel on FDM configuration at 384kHz with two channels or data-streams (suppose the active receiver is the second one in the first data stream):

First channel, second virtual receiver

Set-string to server TCP: "LF010;"

Answer-string from server TCP: "LF010;"=>'0' value means receiver will be set to unlocked state

Set-string to server TCP: "LF011;"

Answer-string from server TCP: "LF011;"=>'1' value means receiver will be set to locked to central frequency state

Example: Setting locked to absolute frequency state for active virtual receiver of the second channel on FDM configuration at 384kHz with two channels or data-streams (suppose the active receiver is the third one in the second data stream):

Second channel, third virtual receiver

Set-string to server TCP: "LF120;"

Answer-string from server TCP: "LF120;" =>'0' value means receiver will be set to unlocked state

Get-string to server TCP: "LF122;"

Answer-string from server TCP: "LF122;" =>'2' value means receiver will be set to locked to absolute frequency

Examples for 'Get'

Example: Getting locked state for active virtual receiver of the first channel on FDM configuration at 384kHz with two channels or data-streams (suppose the active receiver is the second one in the first data stream):

First channel, second virtual receiver

Get-string to server TCP: "LF01;"

Answer-string from server TCP: "LF011;" =>'1' value means receiver is locked to central frequency

Example: Getting locked state for active virtual receiver of the second channel on FDM configuration at 384kHz with two channels or data-streams (suppose the active receiver is the third one in the second data stream):

Second channel, third virtual receiver

Get-string to server TCP: "LF12;"

Answer-string from server TCP: "LF122;" =>'2' value means receiver is locked to absolute frequency

----

SN	Get/Set SNAP state on FDM device										Parameters: *P1(input) Index for channels/data-streams on FDM (1 digit) Index is coded with 1 char ranges from '0' char to channel number-1 char. *P2(input) Unused for this command and set to '0' *P3(input/output) SNAP state value. It is coded with 1 char equal to: '0': SNAP mode off '1': SNAP mode on
Set	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	
	S	N	P1	P2	P3	;					
Get	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	
	S	N	P1	P2	;						
Answer	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	
	S	N	P1	P2	P3	;					

SN command modifies or reads SNAP statefor a specified data-stream on FDM device.

Examples for 'Set'

Example: Set SNAP modeon FDM configuration at 192kHz with one channel or data-stream:

Set-string to server TCP: "SN001;"

Answer-string from server TCP: "SN001;"

Example: Set SNAP modeon FDM configuration at 384kHz with two channels or data-streams:

First channel

Set-string to server TCP: "SN001;"

Answer-string from server TCP: "SN001;"

Second channel

Set-string to server TCP: "SN101;"

Answer-string from server TCP: "SN101;"

=> P1 value '0' means first channel;

=> P1 value '1' means second channel;

=> P2 value is always '0', unused for this command.

Examples for 'Get'

Example: Get SNAP modeon FDM configuration at 192kHz with one channel or data-stream:

Get-string to server TCP: “SN00;”

Answer-string from server TCP: “SN001;”

Example: Get SNAP modeon FDM configuration at 384kHz with two channels or data-streams:

First channel

Get-string to server TCP: “SN00;”

Answer-string from server TCP: “SN001;”

Second channel

Get-string to server TCP: “SN10;”

Answer-string from server TCP: “SN101;”

=> P1 value ‘0’ means first channel;

=> P1 value ‘1’ means second channel;

=> P2 value is always ‘0’, unused for this command

FX	Get/Set Tuning Frequency on virtual receiver of FDM device										Parameters: *P1(input) Index for channels/data-streams on FDM (1digit). Index is coded with 1 char ranges from ‘0’ char to channel number-1 char *P2(input) Index for virtual receiver on FDM (1digit) Index is coded with 1 char ranges from ‘0’ char to maximum virtual receiver number-1 char *P3(input/output) Tuning Frequency (11 digits) is an unsigned integer value expressed in Hz. Tuning Frequency is coded with 11 chars, one char for each digit of the frequency value.
Set	***1***	***2***	***3***	***4***	***5***	***6***	***7***	***8***	***9***	***10***	
	F	X	P1	P2	P3	P3	P3	P3	P3	P3	
	11	***12***	***13***	***14***	***15***	***16***	***17***	***18***	***19***	***20***	
	P3	P3	P3	P3	P3	;					
Get	***1***	***2***	***3***	***4***	***5***	***6***	***7***	***8***	***9***	***10***	
	F	X	P1	P2	;						
Answer	***1***	***2***	***3***	***4***	***5***	***6***	***7***	***8***	***9***	***10***	
	F	X	P1	P2	P3	P3	P3	P3	P3	P3	
	11	***12***	***13***	***14***	***15***	***16***	***17***	***18***	***19***	***20***	
	P3	P3	P3	P3	P3	;					

FX command modifies or reads tuning frequency for a selected virtual receiver on a specified data-stream. The maximum virtual receiver number is 4 for each data channel, then receiver index ranges from 0 to 3.

FX set-command doesn’t modify local oscillator frequency on FDM (it is not a CF command!), but when the virtual receiver selected is locked to Central Frequency, FX set-command acts as a CF set-command.

Then, before using FX set-command, get information about locked to CF state for each virtual receiver and modify it if it is needed for tuning (see LF command).

Examples for ‘Set’

Example: Setting different tuning frequency for each virtual receiver on FDM configuration at 384kHz with two channels or data-streams:

First channel, first virtual receiver

Set-string to server TCP: “FX0000001174000;”

Answer-string from server TCP: “FX0000001174000;”

First channel, second virtual receiver

Set-string to server TCP: “FX0100001175000;”

Answer-string from server TCP: “FX0100001175000;”

First channel, third virtual receiver

Set-string to server TCP: “FX0200001176000;”

Answer-string from server TCP: “FX0200001176000;”

First channel, fourth virtual receiver

Set-string to server TCP: “FX0300001177000;”

Answer-string from server TCP: “FX0300001177000;”

Second channel, first virtual receiver

Set-string to server TCP: “FX1000001166000;”

Answer-string from server TCP: “FX1000001166000;”

Second channel, second virtual receiver

Set-string to server TCP: “FX1100001165000;”

Answer-string from server TCP: “FX1100001165000;”

Second channel, third virtual receiver

Set-string to server TCP: “FX1200001164000;”

Answer-string from server TCP: “FX1200001164000;”

Second channel, fourth virtual receiver

Set-string to server TCP: “FX1300001163000;”

Answer-string from server TCP: “FX1300001163000;”

<u>Examples for ‘Get’</u>

Example: Getting tuning frequency for each virtual receiver on FDM configuration at 384kHz with two channels or data-streams:

First channel, first virtual receiver

Get-string to server TCP: “FX00;”

Answer-string from server TCP: “FX0000001174000;”

First channel, second virtual receiver

Get-string to server TCP: “FX01;”

Answer-string from server TCP: “FX0100001175000;”

First channel, third virtual receiver

Get-string to server TCP: “FX02;”

Answer-string from server TCP: “FX0200001176000;”

First channel, fourth virtual receiver

Get-string to server TCP: “FX03;”

Answer-string from server TCP: “FX0300001177000;”

Second channel, first virtual receiver

Get-string to server TCP: “FX10;”

Answer-string from TCP: “FX1000001166000;”

Second channel, second virtual receiver

Get-string to server TCP: “FX11;”

Answer-string from TCP: “FX1100001165000;”

Second channel, third virtual receiver

Get-string to server TCP: “FX12;”

Answer-string from TCP: “FX1200001164000;”

Second channel, fourth virtual receiver

Get-string to server TCP: “FX13;”

Answer-string from TCP: “FX1300001163000;”

=> P1 value ‘0’ means first channel;

=> P1 value ‘1’ means second channel;

=> P2 value ‘0’ means first receiver, ‘1’ means second receiver, ‘2’means third receiver, ‘3’ means fourth receiver;

FS	Get/Set Frequency Step on virtual receiver of FDM device									
Set	***1***	***2***	***3***	***4***	***5***	***6***	***7***	***8***	***9***	***10***
	F	S	P1	P2	P3	P3	P3	P3	P3	P3
	11	***12***	***13***	***14***	***15***	***16***	***17***	***18***	***19***	***20***
	P3	P3	P3	P3	P3	;				
Get	***1***	***2***	***3***	***4***	***5***	***6***	***7***	***8***	***9***	***10***
	F	S	P1	P2	;					
Answer	***1***	***2***	***3***	***4***	***5***	***6***	***7***	***8***	***9***	***10***
	F	S	P3	P3	P3	P3	P3	P3	P3	P3
	11	***12***	***13***	***14***	***15***	***16***	***17***	***18***	***19***	***20***
	P3	P3	P3	P3	P3	;				

Parameters:
*P1(input)
Index for channels/data-streams on FDM (1 digit).
Index is coded with 1 char ranges from ‘0’ char to channel number-1 char
*P2(input)
Index for virtual receiver on FDM (1 digit)
Index is coded with 1 char ranges from ‘0’ char to maximum virtual receiver number-1 char

P3 parameter is Frequency Stepvalueexpressed in Hz if command is a Get command.
P3 parameter isan Increment / Decrement value equal to +1 or -1 if command is a Set Command.

P3 is a signed integer type value coded with 11 digits.

*P3s(input/output)
Sign of Frequency Stepor Increment/Decrement value.
Sign is coded with 1 char equals to ‘+’ or ‘-‘
*P3i(input/output)
Integer part of Frequency Stepor Increment /Decrement value (10 digits).
Integer part is coded with 10 chars, one char for each digit of the integer part of Frequency Stepor Increment/Decrement value.

FS command modifies or reads frequency stepfor a selected virtual receiver on a specified data-stream. The maximum virtual receiver number is 4 for each data channel, then receiver index ranges from 0 to 3.

FS get-command returns frequency step for a selected virtual receiver expressed in Hz. get-command is independent from state of virtual state receiver (if it is off, on or active).

Frequency step are fixed in the FDMSW2 software: all possible steps are allocated in a vector displayed in the table below:

Frequency Step Vector										
index	***0***	***1***	***2***	***3***	***4***	***5***	***6***	***7***	***8***	***9***
[Hz]	10	25	50	100	250	500	1000	2000	3000	4500
index	***10***	***11***	***12***	***13***	***14***	***15***	***16***	***17***	***18***	***19***
[Hz]	5000	7500	9000	10000	12500	25000	50000	100000	125000	150000

Then the current frequency step is associated to a index value of the frequency step vector in the software.

To set a new frequency step you must to insert the increment value +1 or -1 in P3 field to increment or decrement index value of current frequency step

For example if current frequency step is 500Hz, associated software index is equal to 5.If set command has +1 in P3 field,index is incremented and the new frequency step is 1 kHz; if set command has -1in P3 field, index is decremented the new frequency step is 250 Hz.

FS set-command modify frequency step only if the selected virtual receiver in the command is active. If it is not active, server answer is “???”.

Then before using set-command, get information about the state for the virtual receiver and modify it as active if you want to change frequency step (see SR command).

Examples for ‘Get’

Example: Getting frequency stepfor each virtual receiver on FDM configuration at 384kHz with two channels or data-streams (suppose the first receiver frequency step is 1kHz, the second receiver frequency step is 10kHz, the third receiver frequency step is 10 Hz and the fourth receiver frequency step is 150 kHz)

First channel, first virtual receiver

Get-string to server TCP: “FS00;”

Answer-string from server TCP: “FS00+0000001000;”

First channel, second virtual receiver

Get-string to server TCP: “FS01;”

Answer-string from server TCP: “FS01+0000010000;”

First channel, third virtual receiver

Get-string to server TCP: “FS02;”

Answer-string from server TCP: “FS02+0000000010;”

First channel, fourth virtual receiver

Get-string to server TCP: “FS03;”

Answer-string from server TCP: “FS03+0000150000;”

Second channel, first virtual receiver

Get-string to server TCP: “FS10;”

Answer-string from TCP: “FS10+0000001000;”

Second channel, second virtual receiver

Get-string to server TCP: “FS11;”

Answer-string from TCP: “FS11+0000010000;”

Second channel, third virtual receiver

Get-string to server TCP: “FS12;”

Answer-string from TCP: “FS12+0000000010;”

Second channel, fourth virtual receiver

Get-string to server TCP: “FS13;”

Answer-string from TCP: “FS13+0000150000;”

=> P1 value ‘0’ means first channel;

=> P1 value ‘1’ means second channel;

=> P2 value ‘0’ means first receiver, ‘1’ means second receiver, ‘2’means third receiver, ‘3’ means fourth receiver;

Examples for ‘Set’

Example: Setting different frequency stepfor each virtual receiver on FDM configuration at 384kHz with two channels or data-streams:

First channel, first virtual receiver (must be active)

Set-string to server TCP: “FS00+0000000001;”

Answer-string from server TCP: “FS00+0000000001;”

To verify, use get-command

Get-string to server TCP: “FS00;”

Answer-string from server TCP: “FS00+0000002000;”=>incremented from 1 kHz

First channel, second virtual receiver (must be active)

Set-string to server TCP: “FS01-0000000001;”

Answer-string from server TCP: “FS01-0000000001;”

To verify, use get-command

Get-string to server TCP: “FS01;”

Answer-string from server TCP: “FS01+0000009000;”=>decremented from 10 kHz

First channel, third virtual receiver (must be active)

Set-string to server TCP: “FS02-0000000001;”

Answer-string from server TCP: “FS02-0000000001;”

To verify, use get-command

Get-string to server TCP: “FS02;”

Answer-string from server TCP: “FS02+0000000010;”=> min limit reached 10 Hz

First channel, fourth virtual receiver (must be active)

Set-string to server TCP: “FS03+0000000001;”

Answer-string from server TCP: “FS03+0000000001;”

To verify, use get-command

Get-string to server TCP: “FS03;”

Answer-string from server TCP: “FS03+0000150000;”=>max limit reached 150 kHz

Second channel, first virtual receiver (must be active)

Set-string to server TCP: “FS10+0000000001;”

Answer-string from server TCP: “FS10+0000000001;”

To verify, use get-command

Get-string to server TCP: “FS10;”

Answer-string from server TCP: “FS10+0000002000;”=>incremented from 1 kHz

Second channel, second virtual receiver (must be active)

Set-string to server TCP: "FS11-0000000001;"

Answer-string from server TCP: "FS11-0000000001;"

To verify, use get-command

Get-string to server TCP: "FS11;"

Answer-string from server TCP: "FS11+0000009000;"=>decremented from 10 kHz

Second channel, third virtual receiver (must be active)

Set-string to server TCP: "FS12-0000000001;"

Answer-string from server TCP: "FS12-0000000001;"

To verify, use get-command

Get-string to server TCP: "FS12;"

Answer-string from server TCP: "FS12+0000000010;"=> min limit reached 10 Hz

Second channel, fourth virtual receiver (must be active)

Set-string to server TCP: "FS13+0000000001;"

Answer-string from server TCP: "FS13+0000000001;"

To verify, use get-command

Get-string to server TCP: "FS13;"

Answer-string from server TCP: "FS13+0000150000;"=>max limit reached 150 kHz

----

TX	Get/Set transmission Enable on virtual receiver of FDMDUO device										Parameters: *P1(input) Index for channels/data-streams on FDMDUO (1 digit). Index is coded with 1 char ranges from '0' char to channel number-1 char *P2(input) Index for virtual receiver on FDMDUO (1 digit) Index is coded with 1 char ranges from '0' char to maximum virtual receiver number-1 char *P3(input/output) Transmission state on selected virtual receiver. It is coded with 1 char equals to '1' to enable transmission, '0' to disable transmission.
Set	1	2	3	4	5	6	7	8	9	10	
	T	X	P1	P2	P3	;					
Get	1	2	3	4	5	6	7	8	9	10	
	T	X	P1	P2	;						
Answer	1	2	3	4	5	6	7	8	9	10	
	T	X	P1	P2	P3	;					

This command is available only for FDMDUO devices.

TX-set command enables or disables the transmission of a selected virtual receiver on a specified data-stream. TX-get command obtains if a selected virtual receiver is in transmission or not. The maximum virtual receiver number is 4 for each data channel, then receiver index ranges from 0 to 3. This command acts as TX button for each virtual receiver in FDMSW2 user interface (Note: to see TX button on user interface, each receiver must be on or active).

TX set-command also changes receiver state: it puts the selected virtual receiver as active to enable or disable transmission.

Examples for 'Set'

Example: Enable transmission on second virtual receiver of the first channel on FDMDUO configuration at 192kHz with two channels or data-streams. Suppose that the active receiver is the first one in the first data stream and the others are off.

To know receivers' state, use SR command:

1. First channel, first virtual receiver:

   Get-string to server TCP: "SR00;"

   Answer-string from server TCP: "SR002;"=> '2' value means the receiver 1 is on and active

2. First channel, second virtual receiver:

   Get-string to server TCP: "SR01;"

   Answer-string from server TCP: "SR010;"=> '0' value means the receiver 2 is off

3. First channel, third virtual receiver:

   Get-string to server TCP: “SR02;”

   Answer-string from server TCP: “SR020;”=> ‘0’ value means the receiver 3 is off

4. First channel, fourth virtual receiver:

   Get-string to server TCP: “SR03;”

   Answer-string from server TCP: “SR030;”=> ‘0’ value means the receiver 4 is off

Now enable the transmission on first channel, second virtual receiver:

Set-string to server TCP: “TX011;”=>‘1’ value means receiver will be set totransmission

Answer-string from server TCP: “TX011;”

To know modified receivers’ state, use SR command:

1. First channel, first virtual receiver:

   Get-string to server TCP: “SR00;”

   Answer-string from server TCP: “SR001;”=> ‘1’ value means the receiver 1 is on

2. First channel, second virtual receiver:

   Get-string to server TCP: “SR01;”

   Answer-string from server TCP: “SR012;”=> ‘2’ value means the receiver 2 is on and active

3. First channel, third virtual receiver:

   Get-string to server TCP: “SR02;”

   Answer-string from server TCP: “SR020;”=> ‘0’ value means the receiver 3 is off

4. First channel, fourth virtual receiver:

   Get-string to server TCP: “SR03;”

   Answer-string from server TCP: “SR030;”=> ‘0’ value means the receiver 4 is off

<u>Examples for ‘Get’</u>

Example: Getting if the virtual receivers of the first channel on FDMDUO configuration at 192kHz with two channels or data-streams are in transmission or not:

1. First channel, first virtual receiver

   Get-string to server TCP: “TX00;”

Answer-string from server TCP: “TX000;” =>‘0’ value means that transmission is disabled

2. First channel, second virtual receiver

   Get-string to server TCP: “TX01;”

   Answer-string from server TCP: “TX011;” =>‘1’ value means that transmission is enabled

3. First channel, third virtual receiver

   Get-string to server TCP: “TX02;”

   Answer-string from server TCP: “TX020;” =>‘0’ value means that transmission is disabled

4. First channel, fourth virtual receiver

   Get-string to server TCP: “TX03;”

   Answer-string from server TCP: “TX030;” =>‘0’ value means that transmission is disabled

----

MD	Get/Set demodulation on active virtual receiver of FDM device										Parameters: *P1(input) Index for channels/data-streams on FDM (1 digit). Index is coded with 1 char ranges from '0' char to channel number-1 char *P2(input) Index for virtual receiver on FDM (1 digit) Index is coded with 1 char ranges from '0' char to maximum virtual receiver number-1 char *P3(input/output) Code for demodulation mode on FDM(1digit). Demodulation is coded with 1 char: '0': CW '1': CW SH+ '2': CW SH- '3': USB '4': LSB '5': AM '6': FM '7': DRM '8': WB FM '9': SYNC AM
Set	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	
	M	D	P1	P2	P3	;					
Get	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	
	M	D	P1	P2	;						
Answer	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	
	M	D	P1	P2	P3	;					

MD	Get/Set demodulation on active virtual receiver of FDM										Parameters: *P3(input/output) Code for demodulation on FDM (2 digits). Demodulation is coded with 2 chars "10": DSB "11": RTTY "12": RTTY "13": CW NW "14": ECSS
Set	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	
	M	D	P1	P2	P3	P3	;				
Read	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	
	M	D	P1	P2	;						
Answer	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	
	M	D	P1	P2	P3	P3	;				

MD command modifies or reads demodulation for the selected and active virtual receiver on a specified data-stream. The maximum virtual receiver number is 4 for each data channel, then receiver index ranges from 0 to 3.

MD set-command acts on demodulation only if selected virtual receiver is active; then, before using this command, get information about the state for the virtual receiver and modify it as active if you want to change demodulation (see SR command).

Examples for 'Set'

Example: Setting AM demodulation for active virtual receiver of the first channel on FDM configuration at 384kHz with two channels or data-streams (suppose the active receiver is the second one in the first data stream):

First channel, second virtual receiver

Set-string to server TCP: “MD015;”

Answer-string from server TCP: “MD015;” => ‘5’ is AM demodulation

Example: Setting DSB demodulation for active virtual receiver of the second channel on FDM configuration at 384kHz with two channels or data-streams (suppose the active receiver is the third one in the second data stream):

Second channel, third virtual receiver

Get-string to server TCP: “MD1210;”

Answer-string from server TCP: “MD1210;” =>“10” is DSB demodulation

<u>Examples for ‘Get’</u>

Example: Getting demodulation for active virtual receiver of the first channel on FDM configuration at 384kHz with two channels or data-streams (suppose the active receiver is the second one in the first data stream):

First channel, second virtual receiver

Get-string to server TCP: “MD01;”

Answer-string from server TCP: “MD015;” => ‘5’ is AM demodulation

Example: Getting demodulation for active virtual receiver of the second channel on FDM configuration at 384kHz with two channels or data-streams (suppose the active receiver is the third one in the second data stream):

Second channel, third virtual receiver

Get-string to server TCP: “MD12;”

Answer-string from server TCP: “MD1210;” =>“10” is DSB demodulation

----

SM	Get S-meter value on the open virtual receiver of FDM device										Parameters: *P1(input) Index for channels/data-streams on FDM (1 digit). Index is coded with 1 char ranges from ‘0’ char to channel number-1 char *P2(input) Index for virtual receiver on FDM (1 digit) Index is coded with 1 char ranges from ‘0’ char to maximum virtual receiver number-1 char *P3(output) S-meter parameter (4 digits) S-meter parameter is coded with 4 chars: “0000”: S0 “0002”: S1 “0003”: S2 “0004”: S3 “0005”: S4 “0006”: S5 “0008”: S6 “0009”: S7 “0010”: S8 “0011”: S9 “0012”: S9+10 “0014”: S9+20 “0016”: S9+30 “0018”: S9+40 “0020”: S9+50 “0022”: S9+60
Set											
Get	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	
	S	M	P1	P2	;						
Answer	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	
	S	M	P1	P2	P3	P3	P3	P3	;		

SM command reads s-meter value (coded for S-parameter) for the selected or active virtual receiver on a specified data-stream. The maximum virtual receiver number is 4 for each data channel, then receiver index ranges from 0 to 3.

SM get-command reads the correct S-meter value only if selected virtual receiver is on (or active); then, before using this command, get information about the state for the virtual receiver and modify it as on if you want to read S-meter value (see SR command).

Examples for ‘Get’

Example: Getting S-meter parameter for virtual receiver of the first channel on FDM configuration at 384kHz with two channels or data-streams (suppose the open receiver is the second one in the first data stream):

First channel, second virtual receiver

Get-string to server TCP: “SM01;”

Answer-string from server TCP: “SM010016;” =>“0012” is S9+30 value

Example: Getting s-meter parameter for virtual receiver of the second channel on FDM configuration at 384kHz with two channels or data-streams (suppose the open receiver is the third one in the second data stream):

Second channel, third virtual receiver

Get-string to server TCP: “SM12;”

Answer-string from server TCP: “SM120002;” =>“0002” is S1 value

----

RX	Get S-meter value on the open virtual receiver of FDM device [dBm]										Parameters: *P1(input) Index for channels/data-streams on FDM (1 digit). Index is coded with 1 char ranges from ‘0’ char to channel number-1 char *P2(input) Index for virtual receiver on FDM (1 digit) Index is coded with 1 char ranges from ‘0’ char to maximum virtual receiver number-1 char P3 parameter is S-meter valueexpressed in dBm. P3 is a float type value coded with 11 digits. *P3s(output) Sign of S-meter value. Sign is coded with 1 char equals to ‘+’ or ‘-‘ *P3i(output) Integer part of S-meter value (3 digits). Integer part is coded with 3 chars, one char for each digit of the integer part of S-meter value. *P3.(output) Fractional separator is 1 char equals to ‘.’ *P3f(output) Fractional part of s-meter value (6 digits). Fractionalpart is coded with 6 chars,one char for each digit of the fractional part of s-meter value.
Set											
Get	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	
	R	X	P1	P2	;						
Answer	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	
	R	X	P1	P2	P3s	P3i	P3i	P3i	P3.	P3f	
	11	*12*	*13*	*14*	*15*	*16*	*17*	*18*	*19*	*20*	
	P3f	P3f	P3f	P3f	P3f	;					

RX command reads s-meter value in dBm for the selected or active virtual receiver on a specified data-stream. The maximum virtual receiver number is 4 for each data channel, then receiver index ranges from 0 to 3.

RX get-command reads the correct S-meter value only if selected virtual receiver is on (or active); then, before using this command, get information about the state for the virtual receiver and modify it as on if you want to read S-meter value (see ST command).

Examples for ‘Get’

Example: Getting s-meter parameter for virtual receiver of the first channel on FDM configuration at 384kHz with two channels or data-streams (suppose the open receiver is the second one in the first data stream):

First channel, second virtual receiver

Get-string to server TCP: “RX01;”

Answer-string from server TCP: “RX01-038.880020;”

Example: Getting s-meter parameter for virtual receiver of the second channel on FDM configuration at 384kHz with two channels or data-streams (suppose the open receiver is the third one in the second data stream):

Second channel, third virtual receiver

Get-string to server TCP: “RX12;”

Answer-string from server TCP: “RX12-117.885685;”

GS-2	Get 1024 points of frequency spectrum [dBm]										Parameters: *P1(input) Index for channels/data-streams on FDM (1 digit). Index is coded with 1 char ranges from '0' char to channel number-1 char *P2(input) Fixed to '2', (it doesn't means virtual receiver index!) Pn parameters have n index ranges from3 a 1026. Pn is an averaged spectrum point expressed in dBm. It is a float type value coded with 11 digits. *Pns(output) Sign of spectrum point. Sign is coded with 1 char equals to '+' or '-' *Pni(output) Integer part of spectrum point (3 digits). Integer part is coded with 3 chars, one char for each digit of the integer part of spectrum point. *Pn.(output) Fractional separator is 1 char equals to '.' *Pnf(output) Fractional part of spectrum point (6 digits). Fractionalpart is coded with 6 chars,one char for each digit of the fractional part of spectrum point.
Set											
Get	1	2	3	4	5	6	7	8	9	10	
	G	S	P1	P2='2'	;						
Answer	1	2	3	4	5	6	7	8	9	10	
	G	S	P1	P2='2'	P3s	P3i	P3i	P3i	P3.	P3f	
	11	12	13	14	15	16	17	18	19	20	
	P3f	P3f	P3f	P3f	P3f	P4s	P4i	P4i	P4i	P4.	
	21	22	23	24	25	26	….	….	….	….	
	P4f	P4f	P4f	P4f	P4f	P4f	….	….	….	….	
	x1	x2	x3	x4	x5	x6	x7	x8	x9	(x+1)0	
	Pns	Pni	Pni	Pni	Pn.	Pnf	Pnf	Pnf	Pnf	Pnf	
	(x+1)1	….	….	….	….	….	….	….	….	….	
	Pnf	….	….	….	….	….	….	….	….	….	
	….	….	….	….	….	….	….	….	….	11269	
									….	;	

"GS02" command reads 1024 spectrum points in dBm on the first data-stream for the FDM configuration. Each point is a float type and it is coded with 11 chars (one char for digit). Then the server answerisa string with four chars for header ("GS02"), plus 1024*11=11264 charsfor spectrum points and command end char ";". Total answer length is 11269chars.

To get the spectrum of second data stream (if FDM configuration has more than one data channel), use "GS12" command.

If second channel is not supported for the selected hardware configuration, the server answer string is "???"

<u>Examples for 'Get'</u>

Example: Getting 1024 spectrum point of the first channel on FDM configuration at 384kHz with two channels or data-streams

Get-string to server TCP: "GS02;"

Answer-string from server TCP: "GS02……..-062.715000-074.800000;" (headerand last two spectrum points)

Example: Getting 1024 spectrum point of the second channel on FDM configuration at 384kHz with two channels or data-streams

Get-string to server TCP: "GS12;"

Answer-string from server TCP: "GS12……..-062.715000-074.800000;" (header and last two spectrum points)

GS-3	Getspectrumconfigurationparameters									
Set										
Get	1	2	3	4	5	6	7	8	9	10
	G	S	P1	P2='3'	;					
Answer	1	2	3	4	5	6	7	8	9	10
	G	S	P1	P2= '3'	P3s	P3i	P3i	P3i	P3i	P3i
	11	12	13	14	15	16	17	18	19	20
	P3i	P3i	P3i	P3i	P3i	P4s	P4i	P4i	P4i	P4i
	21	22	23	24	25	26	27	28	29	30
	P4i	P4i	P4i	P4i	P4i	P4i	P5s	P5i	P5i	P5i
	31	32	33	34	35	36	37	38	39	40
	P5i	P5i	P5i	P5i	P5i	P5i	P5i	P6s	P6i	P6i
	….	….	….	….	….	….	….	….	103	104
	….	….	….	….	….	….	….	….	P11i	P12s
	105	106	107	108	109	110	111	112	113	114
	P12i	P12i	P12i	P12i	P12i	P12i	P12i	P12i	P12i	P12i
	115	116	117	118	119	120	121	122	123	124
	P13s	P13i	P13i	P13i	P13i	P13i	P13i	P13i	P13i	P13i
	125	126	127	128	129	130	131	132	133	134
	P13i	;								

Parameters:
*P1(input)
Index for channels/data-streams on FDM (1 digit).
Index is coded with 1 char ranges from '0' char to channel number-1 char
*P2(input)
Fixed to '3', (it doesn't means virtual receiver index!)

Pn parameters have n index from 3 a 13:each parameter is a integer type coded with 11 digits.
P3(output): isspectrum channel index
P4(output): is frequency sampling [Hz]
P5(output): is spectrum point number
P6(output): is displayed spectrum point number
P7(output): is start displayed spectrum index
P8(output): is stop displayed spectrum index
P9(output): is central frequency or local oscillator frequency [Hz]
P10(output): is start displayed spectrum frequency [Hz]
P11(output): is stop displayed spectrum frequency [Hz]
P12(output): reserved for spectrum offset level, but not implemented yet
P13(output): point number for spectrum average.

*Pns(output)
Sign of spectrum parameter.
Sign is coded with 1 char equals to '+' or '-'
*Pni(output)
Unsigned integer value of spectrum parameter (10 digits)
Unsigned integer value is coded with 10 chars, one char for each digit of the value of spectrum parameter.

"GS03" command reads 11 spectrum parameters on the first data-stream for the FDM configuration. Each value is an integer type and it is coded with 11 chars (one char for digit). Then the server answer is a string with four chars for header ("GS03"), plus 11*11=121 chars for spectrum points and command end char ";". Total answer length is 126 chars.

To get the spectrum parameters of second data stream (if FDM configuration has more than one data channel), use "GS13" command.

If second channel is not supported for the selected hardware configuration, the server answer string is "???"

Examples for 'Get'

Example: Getting 11 spectrum parameters of the first channel on FDM configuration at 384kHz with two channels or data-streams

Get-string to server TCP: "GS03;"

Answer stringfrom server TCP: "GS03+0000000000+0000384000+0000016384+0000001024+0000001638+0000014746+0001170000-0000076805+0000076805+0000000000+0000000002;"

P3: Data stream/data channel index : +0000000000 => 0

P4: Frequency sampling: +0000384000 =>384kHz

P5: Evaluated spectrum point number: +0000016384=> 16384 points

- Spectrumresolutionis: 384000 / 16384 = 23.4375 Hz

P6: Displayed spectrum point number: +0000001024=> 1024 points

P7: Start displayed spectrum index: +0000001638=> 1638

P8: Stopdisplayed spectrum index: +0000014746=> 14746

P9: Central frequency: +0001170000=> 1.170 MHz

P10: Start displayed spectrum frequency: -0000153609=> -153.609 kHz

P11: Stopdisplayed spectrum frequency: +0000153609=> +153.609 kHz

- Spectrumspanis: +153.609 - (-153.609) = 307.218 kHz
- Spectrumbandwitdhis: [+153609 - (-153609)] / 384000 = 0.8, 80%

P12: Reserved for spectrum offset level, but not implemented yet: +0000000000 => 0

P13: Point number for evaluating spectrum average: +0000000002 => 2

Example: Getting 11 spectrum parameters of the second channel on FDM configuration at 384kHz with two channels or data-streams

Get-string to server TCP: "GS13;"

Answer stringfrom server TCP:
"GS13+0000000001+0000384000+0000016384+0000001024+0000001638+0000014746+0001170000-0000076805+0000076805+0000000000+0000000002;"

P3: Data stream/data channel index : +0000000001 =>1

P4: Frequency sampling: +0000384000 =>384kHz

P5: Evaluated spectrum point number: +0000016384=> 16384 points

- Spectrumresolutionis: 384000 / 16384 = 23.4375 Hz

P6: Displayed spectrum point number: +0000001024=> 1024 points

P7: Start displayed spectrum index: +0000001638=> 1638

P8: Stopdisplayed spectrum index: +0000014746=> 14746

P9: Central frequency: +0014008000=>14.008 MHz

P10: Start displayed spectrum frequency: -0000153609=> -153.609 kHz

P11: Stopdisplayed spectrum frequency: +0000153609=> +153.609 kHz

- Spectrumspanis: +153.609 - (-153.609) = 307.218 kHz
- Spectrumbandwitdhis: [+153609 - (-153609)] / 384000 = 0.8, 80%

P12: Reserved for spectrum offset level, but not implemented yet: +0000000000 => 0

P13: Point number for evaluating spectrum average: +0000000002 => 2

GS-4	Get 1024 points of frequency spectrum [dBm] converted to short on +/-180 dBm range										Parameters: *P1(input) Index for channels/data-streams on FDM (1 digit). Index is coded with 1 char ranges from '0' char to channel number-1 char *P2(input) Fixed to '4', (it doesn't means virtual receiver index!) Pn parameters have n index from 3 a 1026. *Pn(output): Averaged spectrum sample in short type format.
Set											
Get	***1***	***2***	***3***	***4***	***5***	***6***	***7***	***8***	***9***	***10***	
	G	S	P1	P2='4'	;						
Answer	***1***	***2***	***3***	***4***	***5***	***6***	***7***	***8***	***9***	***10***	
	G	S	P1	P2='4'	P3	P4	P5	P6	P7	P8	
	11	***12***	***13***	***14***	***15***	***16***	***17***	***18***	***19***	***20***	
	P9	P10	P11	P12	P13	P14	P15	P16	P17	P18	
	....	**....**	**....**	**....**	**....**	**....**	**....**	**....**	**....**	***x(n+2)***	
	....	**....**	**....**	**....**	**....**	....	....	....	....	Pn	
	....	**....**	**....**	**....**	**....**	**....**	***1026***	***1027***	***1028***	***1029***	
							P1024	P1025	P1026	;	

"GS04" command reads 1024 spectrum points on the first data-stream for the FDM configuration. Each point is a short type and corresponds to one char. Then the server answer is a string with four chars for header ("GS04"), plus 1024chars for spectrum points and command end char ";". Total answer length is1029chars.

To get the spectrum of second data stream (if FDM configuration has more than one data channel), use "GS14" command.

If second channel is not supported for the selected hardware configuration, the server answer string is "???"

Notice that each char is two bytes long because one char is the spectrum point in short format without any type of coding. In Figure 6 there is an example of spectrum reception with Hercules client: the client sends a get command to TCP server (in Figure 6 the command is green-marked) and the data answerfrom server is a string where the header "GS04" and the end ";" are in chars format and the 1024 spectrum points are in short format (in Figure 6 header string and end char are blue-marked).If the client is set to receive data in byte, there are 8 bytes for header, 2048 bytes for spectrum points and 2 bytes for end char. The client must code header and end char as strings, and for each spectrum point the operations listed below must be done:

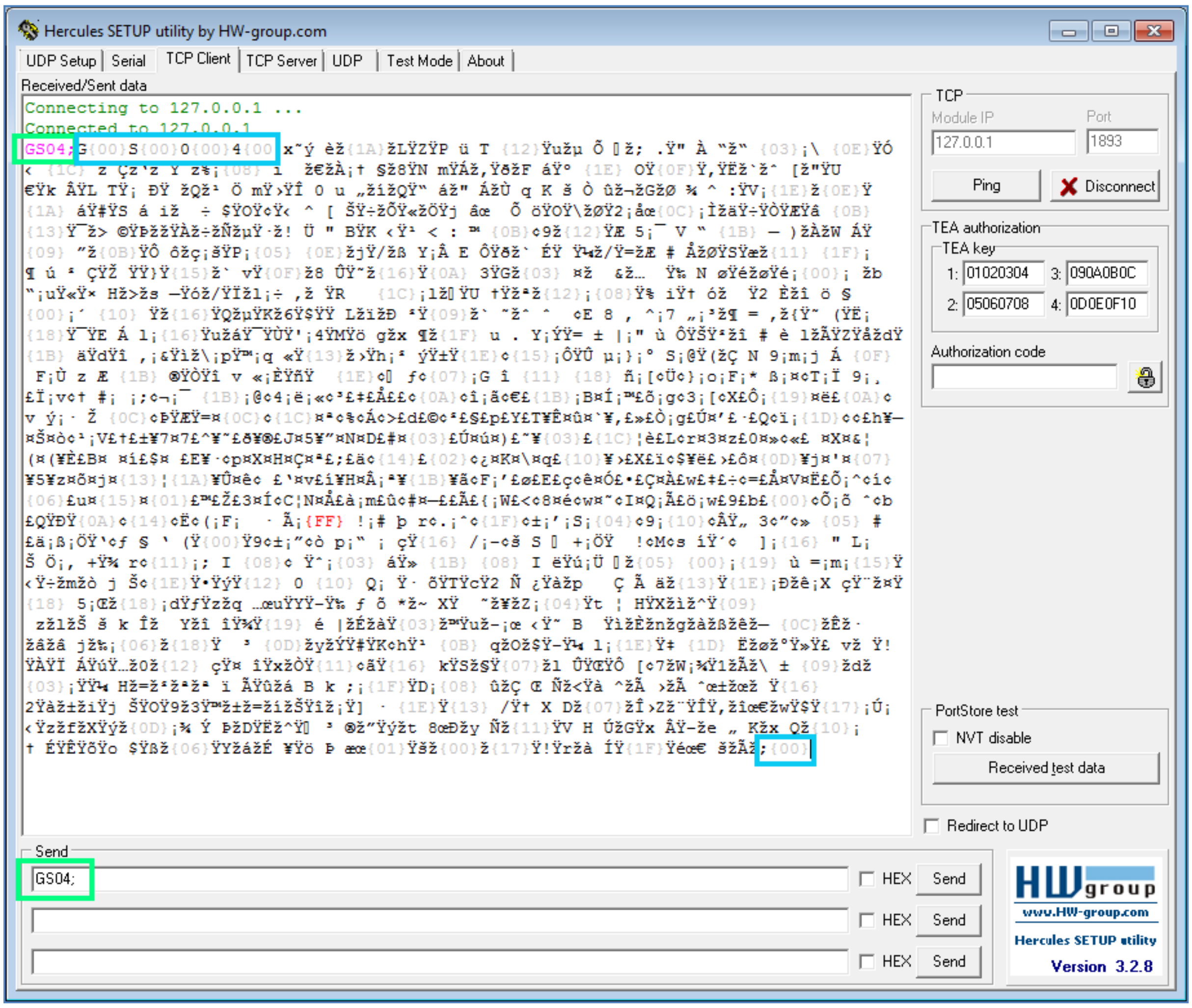
1. Declare `b_value` variable as a two byte array and fill with bytes of a single spectrum point.
2. Convert the array in a short type variable:

```
value = BitConverter.ToInt16(b_value, sizeof(short));
```

3. Convert in dBm value:

```
value_dBm = offset_dBm+ (Convert.ToDouble(value) / (Math.Pow(2.0,15.0)))*(180.0);
```

The variable `offset_dB` is filled with the value obtained from decoding P12 parameter in the answer of "GS03;" command (decoding for P12 parameter is not specified because it is not implemented yet).



```
Connecting to 127.0.0.1 ...
Connected to 127.0.0.1
GS04;G{00}S{00}0{00}4{00}
```

Hercules SETUP utility by HW-group.com

TCP: Module IP 127.0.0.1, Port 1893

TEA key: 1: 01020304, 2: 05060708, 3: 090A0B0C, 4: 0D0E0F10

Send: GS04;

Figure6: Example of spectrum reception with client Hercules on first data channel.

<u>Examples for 'Get'</u>

Example: Getting 1024 spectrum point of the first channel on FDM configuration at 384kHz with two channels or data-streams

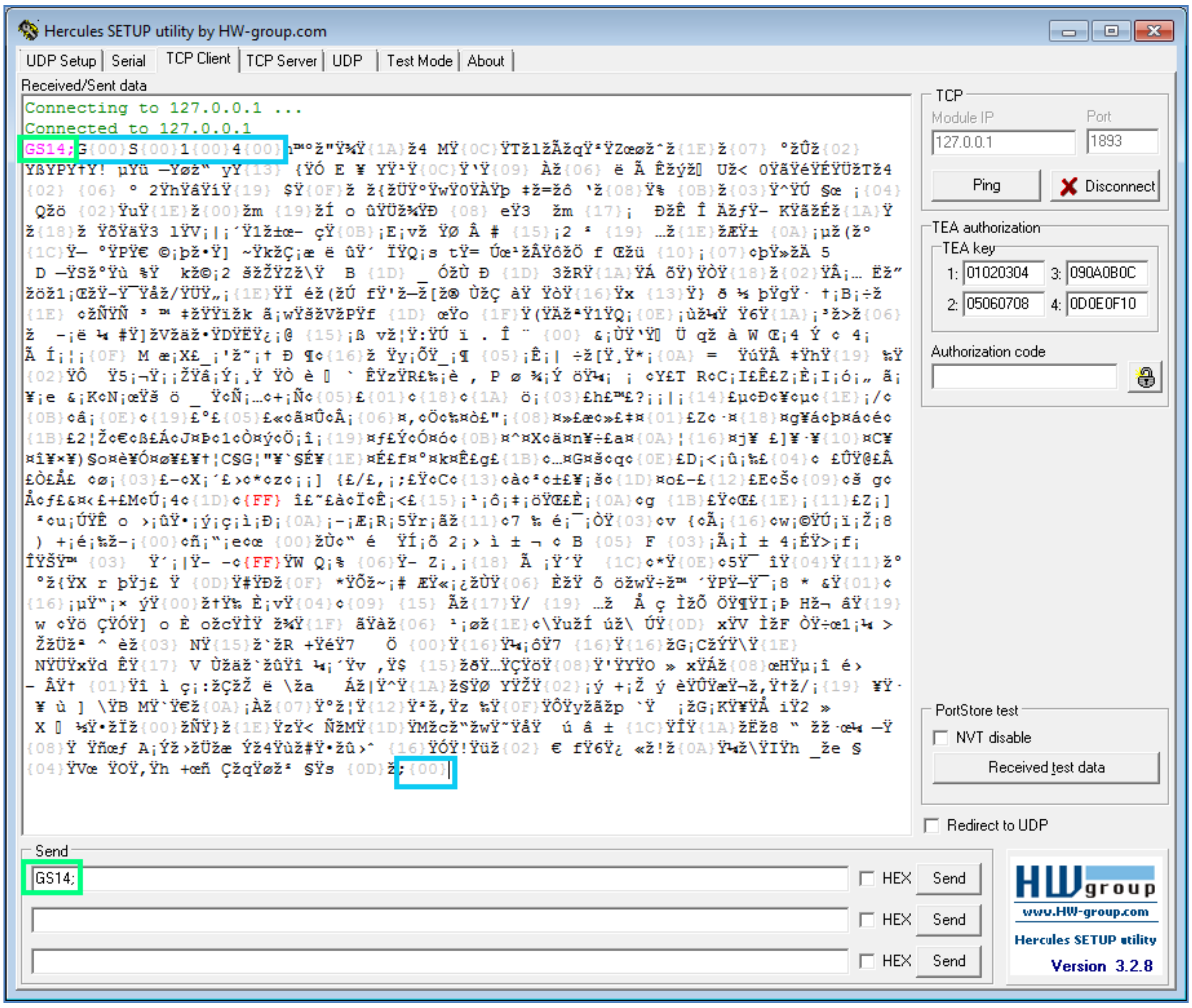
Get-string to server TCP: "GS04;"

Answer-string from server TCP: "GS04.....................................;" (see Figure 6)

Example: Getting 1024 spectrum point of the second channel on FDM configuration at 384kHz with two channels or data-streams

Get-string to server TCP: "GS14;"

Answer-string from server TCP: "GS14.....................................;" (see Figure 7)



```
Connecting to 127.0.0.1 ...
Connected to 127.0.0.1
GS14;G{00}S{00}1{00}4{00}
```

Figure 7 Example of spectrum reception with client Hercules on second data channel.

Specification for the correct FDM tuning

It is strongly recommended to set all virtual receiver into unlocked state or locked to absolute frequency state before changing FDM local oscillator frequency or virtual receiver tuning frequency.

In order to do this, follow the operations listed below:

1. Activate virtual receiver 1 and put it into unlocked/locked to absolute frequency state
   a. Send “SR001;” => receiver 1 is active
   b. Send “LF000;” => receiver 1 is unlocked
   c. Send “LF002;” => receiver 1 is locked to absolute frequency
2. Activate virtual receiver 2 and put it into unlocked/locked to absolute frequency state
   a. Send “SR011;” => receiver 2 is active
   b. Send “LF010;” => receiver 2 is unlocked
   c. Send “LF012;” => receiver 2 is locked to absolute frequency
3. Activate virtual receiver 3 and put it into unlocked/locked to absolute frequency state
   a. Send “SR021;” => receiver 3 is active
   b. Send “LF020;” => receiver 3 is unlocked
   c. Send “LF022;” => receiver 3 is locked to absolute frequency
4. Activate virtual receiver 4 and put it into unlocked/locked to absolute frequency state
   a. Send “SR031;” => receiver 4 is active
   b. Send “LF030;” =>receiver 4is unlocked
   c. Send “LF032;” => receiver 4 is locked to absolute frequency
5. Set local oscillator frequency to the desired central frequency (for example 14008000 Hz)
   a. Send “CF0000014008000;”
6. Get start and stop spectrum frequencies
   a. Send “GS03;” => P10 parameter is start frequency and P11 parameter is stop frequency; for example, if the 384kHz configuration is set, start frequency is -153609 Hz and stop frequency is 153609 Hz.
7. Tune virtual receivers in the range of (central frequency + start frequency) to (central frequency + stop frequency) only if receivers are in unlocked state.
   a. Send “FX0000014048000;” => receiver 1 is tuned at 14.048MHz
   b. Send “FX0100014088000;” => receiver 2 is tuned at 14.088MHz
   c. Send “FX0200013988000;” => receiver 2 is tuned at 13.988MHz
   d. Send “FX0300013948000;” => receiver 3 is tuned at 13.948MHz

   If receivers are locked to absolute frequency,tuning can be done out of spectrum start and stop frequency limits (there is no need of point 6) .

RC	Get/Set recording on first channel/data stream of FDM device										Parameters:
Set	***1***	***2***	***3***	***4***	***5***	***6***	***7***	***8***	***9***	***10***	*P1(input)
	R	C	P1	P2	P3	P4	P5	P6	P7	P8	Index for channels/data-streams on FDM (1 digit).
	11	***12***	***13***	***14***	***15***	***16***	***17***	***18***	***19***	***20***	It fixed to ‘0’ to means first channel
	P9	P10	P11	P12	P13	P14	P15	P16	P17	P18	*P2(input)
	….	**….**	**….**	**….**	**….**	**….**	**….**	**….**	**….**	***n+2***	Fixed to ‘0’, unused to set virtual receiver
	….	….	….	….	….	….	….	….	….	Pn	
	….	**….**	**….**	**….**	**….**	**….**	**….**	***68***	***69***	***70***	
	….	….	….	….	….	….	….	P66	P67	;	
Get	***1***	***2***	***3***	***4***	***5***	***6***	***7***	***8***	***9***	***10***	Pn parameters have n index from 3 to67.
	R	C	P1	P2	;						
Answer	***1***	***2***	***3***	***4***	***5***	***6***	***7***	***8***	***9***	***10***	*P3(Input/output): Recording state on first channel/data-stream. It is coded with 1 char equals to ‘1’ to means recording is on, ‘0’ to means recording is off.
	R	C	P1	P2	P3	;					*P4 to*P67(Input): Name of file to record data; it must be until 64-char long

RC-set command starts and stops recording on first channel/data stream of FDM device, with P1 parameter fixed to ‘0’ char.

- P3 parameter is ‘1’ to start recording, ‘0’ to stop recording
- P4 to P67 is recording file name without ‘.wav’ file termination (FDMSW2 automatically includes recording file number and ‘.wav’ termination). File name must be at least 1 char long and it must not exceed 64 chars.

RC-get command reads recording state on first channel/data stream of FDM device with P1 parameter fixed to ‘0’ char.

- P3 parameter is ‘1’ when a recording is on, ‘0’ if there is no recording

Virtual receiver index parameter (P2) is not used and it is fixed to ‘0’ char.

To know recording settings see the last configuration on FDMSW2: click on “SET”button to open “Setup” panel and select “Recording” tab.Or execute right click on rec-button to open “REC Options” panel on main form.

Examples for ‘Set’

Example: Start recording on first channel

Set-string to server TCP: “RC001test;”=> recording starts on ‘test_000.wav’ file

Answer-string from server TCP: “RC001;”

<u>Examples for ‘Get’</u>

Example: Recording state on first channel

Get-string to server TCP: “RC00;”

Answer-string from server TCP: “RC001;” => there is a recording

<u>Examples for ‘Set’</u>

Example: Stop recording on first channel

Set-string to server TCP: “RC000test;” => recording stops on ‘test_xyz.wav’ file

Answer-string from server TCP: “RC000;”

<u>Examples for ‘Get’</u>

Example: Recording state on first channel

Get-string to server TCP: “RC00;”

Answer-string from server TCP: “RC000;” => there is no recording

Default path to save .wav file is set in the last recording configurations.

ST	Get device info									
Get	1	2	3	4	5	6	7	8	9	10
	S	T	P1	P2	;					
Answer	1	2	3	4	5	6	7	8	9	10
	S	T	P1	P2=’0’	P3	P3	P3	P3	;	
Answer	1	2	3	4	5	6	7	8	9	10
	S	T	P1	P2=‘1’	P3	P3	P3	P3	P3	…
	11	12	13	14	15	16	17	18	19	20
	…	…	…	…	…	…	…	…	…	P3
	21	22	23	24	25	26	27	28	29	30
	…	…	…	…	…	…	…	…	…	…
	…	…	…	…	35	36	37			
	…	…	…	…	P3	P3	;			
Answer	1	2	3	4	5	6	7	8	9	10
	S	T	P1	P2=‘2’	P3	P3	P3	P3	P3	…
	11	12	13	14	15	16	17	18	19	20
	…	…	…	…	…	…	…	…	…	P3
	21	22	23	24	25	26	27	28	29	30
	…	…	…	…	…	…	…	…	…	…
	…	…	…	…	35	36	37			
	…	…	…	…	P3	P3	;			

Parameters:
*P1(input)
Not used. Forced to 0 in Answer.
*P2(input)
Sub command:
‘0’: get PID info
‘1’: get SN info
‘2’: get Device name

*P3 (output):
PID device information if sub-command is ‘0’. It is coded with 4chars equals to PID value in hexadecimal.
*P3 (output):
SN device information if sub-command is ‘1’. It is coded with 32chars equals to SN string
*P3 (output):
Device Name information if sub-command is ‘2’. It is coded with 32chars equals to device name string

ST-get command retrieves information about device hw connected to the software. In this command P1 parameter is not used, P2 parameter is used to get different information:

- P2 = ‘0’ get PID of the device connected
- P2 = ‘1’ get SN of the device connected
- P2= ‘2’ get the device name.

Examples

Example: get PID value

Get-string to server TCP: “ST00;”

Answer-string from server TCP: “ST00061C;”→`PID : 0x061C means FDM-S2 Device`

Example: get SN string

Get-string to server TCP: “ST01;”

Answer-string from server TCP: “ST0100000000000000000000SE033B_0012;”→ `SN: SE033B_0012`

Example: get name string

Get-string to server TCP: “ST02;”

Answer-string from server TCP: “ST02000000000000000000000000000FDM-S2;”→ `name: FDM-S2`

MS (Memories Set) command have 3 parameters:

P1 is the index of channels datastream on FDM (1 digit)

P2 is the sub-command (1 digit)

P3 is a parameter for the sub-command (5 digit)

Is possible to add max 60 memories with a command and this memories is memorized with a list number (from 1 to 99999).

Parameter P2 meaning: ‘0’ is Get memories command , ‘1’ is Delete memories command, ‘2’ is Add memories command.

Get and Delete command can operate with list memories. Get and Add memories operate with max 60 memories in a single TCP command.

MS	Get memories										Parameters: *P1(input) Index for channels/data-streams on FDM (1 digit). It fixed to ‘0’ to means first channel *P2(input) Sub command: 0: get command 1: delete command 2: add command *P3 (input) If ‘00000’ and P4,P5 not inserted → give number of memories; If ‘00000’ and P4,P5 inserted → give memories from start to stop index; If ‘xxxxx’ → give memories from ‘xxxxx’ list; *P4 (input) only if P3=0, means start index of get memories *P5 (input) only if P3=0, means stop index of get memories *P3(output) number of memories
Get	***1***	***2***	***3***	***4***	***5***	***6***	***7***	***8***	***9***	***10***	
	M	S	P1	P2	P3	P3	P3	P3	P3	P4	
	11	***12***	***13***	***14***	***15***	***16***	***17***	***18***	***19***	***20***	
	P4	P4	P4	P5	P5	P5	P5	;			
Answer	***1***	***2***	***3***	***4***	***5***	***6***	***7***	***8***	***9***	***10***	
	M	S	P1	P2	P3	P3	P3	P3	P3	P4	
	…	P5	…	P6	P7	P7	P7	P7	…	:	

											*P4(output) 11 digit Frequency *P5(output) 16 digit label memories *P6(output) 1 digit mode *P7(output) 5 digit list

MS get command: P2= '0'

If input P3 = '00000' and P4(input) and P5(input) not inserted in command→ get the number of memories (parameter P4 and P5 output not inserted in the answer)

Es. `MS0000000;MS0000009; → 9 memories present`

If input P3 = '00000' and P4(input) and P5(input) inserted in command→ get the number of memories from P4 to P5. If Nmax is the number of memories present, then P5 can assume the maximum value in Nmax – 1. If P4 = P5 a single memories in returned.

```
Es. MS000000000020003;
MS0000002
00000367000000ZAG ZagabriaX0000
0000036900000000VRS VrsarX0000; → get 2 memories from index 2 to index 3
```

If input P3 = 'xxxxx' P4(input) and P5(input) inserted or not in command → get memories of list 'xxxxx'

```
Es. MS0000001; or MS000000100020003;
MS0000003
00000067000000ZAG ZagabriaX0001
000001080000000CHI ChioggiaX0001
0000011700000000VCA VicenzaX0001;→ 3 memories present in list number 1
```

MS	Delete memories										Parameters:
Delete	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*P1(input) Index for channels/data-streams on FDM (1 digit). It fixed to '0' to means first channel *P2(input) Sub command: 0: get command 1: delete command 2: add command *P3 (input) If '00000' and P4,P5 not inserted → delete all memories; If '00000' and P4,P5 inserted → delete memories from start index P4 to stop index P5; If 'xxxxx' → delete list 'xxxxx'; *P4 (input) only if P3=0, means start index of
	M	S	P1	P2	P3	P3	P3	P3	P3	P4	
	11	*12*	*13*	*14*	*15*	*16*	*17*	*18*	*19*	*20*	
	P4	P4	P4	P5	P5	P5	P5	;			
Answer	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	
	M	S	P1	P2	P3	P3	P3	P3	P3	;	

											delete memories *P5 (input) only if P3=0, means stop index of delete memories *P3(output) answer give number of memories deleted

<u>MS delete command: P2= ‘1’</u>

If input P3 = ‘00000’ and P4(input) and P5(input) not inserted in command → delete all memories

```
Es. MS0100000;MS0100009; → Deleted 9 memories
```

If input P3 = ‘00000’ and P4(input) and P5(input) inserted in command → delete memories from P4 to P5 index. If Nmax is the number of memories present, then P5 can assume the maximum value in Nmax – 1. If P4 = P5 a single memories in deleted.

```
Es. MS010000000040006;MS0100003; → Deleted memories from index 4 to index
6 (3 memories)
```

If input P3 = ‘xxxxx’ and P4(input) and P5(input) inserted or not in command → delete P3 list memories

```
Es. MS0100001; or MS010000100020008; MS0100003; → Delete 3 memories of list
1
```

MS	Add memories										Parameters: *P1(input) Index for channels/data-streams on FDM (1 digit). It fixed to ‘0’ to means first channel *P2(input) Sub command: 0: get command 1: delete command <u>2: add command</u> *P3 (input) Number of memories to add *P4 (input) 11 digit Freq *P5 (input) 16 digit Label *P6 (input) 1 digit mode *P3(output) answer give number of memories added *P4(output) answer give number of list associated
Add	***1***	***2***	***3***	***4***	***5***	***6***	***7***	***8***	***9***	***10***	
	M	S	P1	P2	P3	P3	P3	P3	P3	P4	
	11	***12***	***13***	***14***	***15***	***16***	***17***	***18***	***19***	***20***	
	P4	P4	P4	P4	P4	P4	P4	P4	P4	P4	
	21	***22***	***23***	***24***	***25***	***26***	***27***	***28***	***29***	***30***	
	P5	P5	P5	P5	P5	P5	P5	P5	P5	P5	
	31	***32***	***33***	***34***	***35***	***36***	***37***	**…**	**…**		
	P5	P5	P5	P5	P5	P5	P6	…	;		
Answer	***1***	***2***	***3***	***4***	***5***	***6***	***7***	***8***	***9***	***10***	
	M	S	P1	P2	P3	P3	P3	P3	P4	P4	
	P4	P4	P4	;							

<u>MS add command: P2= ‘2’</u>

Command can add max 60 memories, the answer give the number of memories added on P3 (if a memory already exist this memory is not inserted) and the number of list associated on P4.

```
Es.
MS0200002
000002670000000ZAG ZagabriaX
00000369000000000VRS VrsarX;MS020000200001;added 2 memories on list 1
```

Note. The Code for demodulation mode on FDM (1 digit) is:

‘0’: CW
‘1’: CW SH+
‘2’: CW SH-
‘3’: USB
‘4’: LSB
‘5’: AM
‘6’: FM
‘7’: DRM
‘8’: WB FM
‘9’: SYNC AM
‘X’: not used