XY-DemoRad Network Protocol
v. 0.9.0 (Early Access)

XY-Sensing Ltd.

Contents

1 General communication rules

The protocol operates in the client-server architecture using TCP on underlying layer. Each message exchange is initiated by the client with a command message. The server responds with response. The client should not disconnect until receiving the response. Lack of the response is considered a protocol violation. The server automatically disconnects when there is no message from the client after establishing a TCP connection or after responding to previous command message. After receiving the command message it will not disconnect before sending the response. The client has a freedom managing the TCP connection to send several commands one after another. However it should handle the case, when server disconnects where there is too long interval between the commands. The timeout is 1 second, but it may be changed in the future. It is not advisable for the client implementation to assume the server will not disconnect before that time.

2 Message Format

Messages are exchanged in text format using ASCII encoding. Each message consists of minimum two lines delimited with a newline character, where the last one is always empty, indicating end of message. Therefore, every message is concluded with double newline character.

2.1 Command

The general format of command message is as follows:

1. Single line with the command name - mandatory.
2. Several lines with command arguments - optional. Format of this part depends on particular command.
3. Single empty line - mandatory.

Listing below presents example command.

```
1 set
2 bandwidth 1e9
3 carrier 24e9
4 prf 1e3
5
```

In this example, the command consists of: line 1 with the command name, lines 2-4 with the command arguments and last line is empty, to indicate the end of the message.

2.2 Response

The general format of the response is as follows:

1. Command name with execution status - required. Execution status will be covered later in this section.

2. Response contents - optional. Generally format of this part depends on particular command and it's execution status.

3. Single empty line - mandatory.

Below example command response is presented:

```
1 set partial
2 bandwidth coerced 2e9
3 carrierr unknown
4 prf set
5
```

It consists of: first line indicating, that the command execution partially succeeded. Lines 2-4 contain details and the last line is empty, indicating the end of the message.

2.2.1 Execution status

The first line of the response has the format independent of the particular command. It consists of the command name and the status. The status may be one of the following:

ok The command execution succeeded completely.

error The command execution failed. The response contents contain human-readable error description.

partial The command execution partially succeeded. The response contents is command-dependent with specified format to indicate, which part of the command succeeded and which part of the command failed.

unknown The command with given name was not found, hence it could not be executed.

3 Commands

Each command may be defined in very different way. However two commands, `get` and `set` have special meaning, since they are responsible for writing and reading single values from the sensor, further referred to as parameters.

3.1 `set` command

Command name: `set`
This command is responsible for setting a new value for one or more parameters.

Arguments

The arguments contain a single line for each parameter. The line has format: `parameter value`, where `parameter` is the name of the parameter and `value` is a text representation of its value.

Response

The command response consists of the same number of lines as its arguments. Each line refers to one of the parameters in the same order as they appeared in the command. Lines have format: `[parameter] set|coerced|error|unknown| badFormat [message]`, where `parameter` is the name of the parameter and optional `message` has a meaning dependent on the status of the command execution for given parameter.

Status has following meaning:

set The parameter has been successfully set to provided value.

coerced The parameter has been successfully set, however the value was coerced. The `message` in the line contains actual value set.

error The parameter has not been set due to an error. The `message` contains error description.

unknown Requested parameter name has not been recognized.

badFromat Either the value format is invalid or the whole argument line could not be parsed. Is the `parameter` is present in the response line, it indicates the former case. Otherwise the latter has occurred.

The command execution status is

ok All parameters has been set with requested or coerced values.

partial One or more parameter has failed to be set to requested or coerced value, however one or more succeeded.

error No parameter has been successfully set.

3.2 `get` comamnd

Command name: `get`
This command is responsible for retrieving a value of one or more parameters.

Arguments

The arguments contain a single line for each parameter. Each line should contain only the parameter's name.

Response

The command response consists of the same number of lines as its arguments. Each line refers to one of the parameters in the same order as they appeared in the command. The line contains only the returned value if the parameter has been recognized, or has format `parameter unknown`, where `parameter` is the parameter's name.

The command execution status is

ok All parameters has been read.

partial One or more parameter has failed to be read, however one or more succeeded.

error No parameter has been successfully read.

3.3 `start` command

Command name: `start`
This command enables the sensor: configures and starts waveform generation, enables RF front-end and starts IF signal streaming. Issuing this command when the sensor is running will not cause error.

Arguments

The commands takes no arguments.

Response

The command execution status is

ok Sensor has been successfully configured and started running.

error An error occurred during configuration and the sensor has not started operation.

3.4 `stop` command

Command name: `stop`
This command disables the sensor: stops waveform generation, disables RF front-end and stops IF signal streaming. Issuing this command when the sensor is not running will not cause error.

Arguments

The commands takes no arguments.

Response

The command execution status is

ok Sensor has been successfully stopped.

error An error occurred during execution of the command.

4 Parameters

4.1 List of parameters

The list of available parameters is provided below:

4.1.1 `who`

Access type: read-only
Value type: string
Purpose: The parameter identifies the device as XY-DemoRad sensor and provides information about the firmware version.
Format: `XY-DemoRad_v<major>.<minor>.<patch>_b<build>`, where `major`, `minor`, `patch` and `build` are parts of version number. For example, version 0.9.0, build 1 has version string: `XY-DemoRad_v0.9.0_b001`.

4.1.2 `bandwidth`

Access type: write-only
Value type: numeric
Purpose: Sets bandwidth of the waveform. Accepts values in Hz.

4.1.3 `carrier`

Access type: write-only
Value type: numeric
Purpose: Sets center frequency of the waveform. Accepts values in Hz.

4.1.4 `prf`

Access type: write-only
Value type: numeric
Purpose: Sets pulse repetition frequency of the waveform. Accepts values in Hz.

4.2 `attenuators`

Access type: write-only
Value type: bool (1 or 0)
Purpose: Sets the attenuators state. If enabled the 20 dB attenuation in IF signal chain is enabled.

4.2.1 `status`

Access type: read-only
Value type: string
Purpose: Indicates status of the XY-DemoRad sensor.
Format: `running|ready`

4.2.2 `minFrequency`

Access type: read-only
Value type: numeric
Purpose: Used to obtain minimum frequency within current hardware capabilities of the sensor. Returns values in Hz.

4.2.3 `maxFrequency`

Access type: read-only
Value type: numeric
Purpose: Used to obtain maximum frequency within current hardware capabilities of the sensor. Returns values in Hz.

5 Examples

Below some examples of common operations may be found.

5.1 Device identification

To identify the device, the client should read the `who` parameter:

```
1 get
2 who
3
```

The sensor will provide a response:

```
1 get ok
2 XY-DemoRad_v0.9.0_b001
3
```

with actual version of the software. After verification of the version, frequency limits of the hardware may should be checked:

```
1 get
2 minFrequency
3 maxFrequency
4
```

The response will contain frequency limits the hardware is capable of achieving, for example:

```
1 get ok
2 22500000000
3 26900000000
4
```

5.2 Typical operation

For typical operation, after identifying the sensor, the client needs only to set parameters of the radar and start the operation. Below is an example command for configuring the sensor:

```
1 set
2 carrier 24000000000
3 bandwidth 1000000000
4 prf 1000
5 attenuators 0
6
```

and then only `start` comand is needed:

```
1 start
2
```

The sensor should operate after issuing the last command. To stop its operation the `stop` command should be issued:

```
1 stop
2
```

After the command sensor stops the operation.