COMMUNICATION INTERFACE V1.12

DISCLAMER

WARNING: This information comes with absolutely no warranty. If used improperly you may fry your device. Gamma-Scout GmbH & Co. KG is not responsible for any issues caused by this. Information herein may be incorrect or imprecise and can change without notice.

DOCUMENTATION

The GS can talk to other devices (e.g. PCs) via a serial interface. On ancient devices this is a RS-232 interface (identifiable by the 9 pin D-SUB plug under the I/O port cover) while on newer devices an FTDI chip is used to implement a UART over USB interface (identifiable by the USB type B plug). In any case, the host side will talk to either a real (RS-232) or virtual (FTDI) serial port.

The parameters are:

FW < 6.00:	2400,7,e,1
6.00 <= FW < 6.90:	9600,7,e,1
FW >= 6.90:	460800,7,e,1

The communication with the GS is based on a simple protocol where the host sends a single command character to the device. Depending on this command character, the device immediately performs the corresponding action or (e.g. for commands which set the date and/or time) it expects a parameter block consisting of more characters in order to complete the command.

All data (sent or received) is readable cleartext ASCII (apart from CR and LF).

Depending on the firmware, there are different requirements regarding the delay between the transmissions of the various characters (see the FW specific notes for details).

The characters sent to the device will not be echoed - we recommend to set the local echo to ”on“ in the terminal programme. Additionally, in case the whole communication will be written to a file, it is recommended to record the data sent to the device as well in order to ease subsequent analysis.

ADVICE: It is strongly advised not to play around with undocumented characters or character sequences as this could lead to unexpected and non-reversible results which might render the device unusable.

FIRMWARE VERSIONS AFTER 2.0 AND BEFORE 6.00

Talking to devices with firmware < 6.00 is only possible when the PC mode had been activated by pressing the PC button. There **must** be a subsequent delay of at least 550 ms for **all** characters being sent to the device. If this rule is not obeyed, characters might get lost (resulting in ignored commands or badly transferred parameter blocks).

Commands available in PC mode:

'v'	returns FW version
'b'	dumps protocol memory
'z'	resets protocol pointer to beginning of memory
'u'	prepares setting of the time in the form ”hhmm“
'd'	prepares setting of the date in the form ”DDMMYY“
't'	FW 3.03 and up: returns ”AN“ if this is an ”Alert“ device, else ”AUS“

In online mode (only for online devices):

's' requests online status and pulses
'0'-'9' sets online interval ('0' only available in FW 5.43)

FIRMWARE VERSIONS AFTER 6.00

Starting with FW 6.00, modes (Standard, PC, Online) can be changed remotely and more commands have been implemented. After having sent a command character, there **must** be a delay of at least 550 ms before the next character (new command or first char of a parameter block) may be sent. Possibly subsequently sent characters of a parameter block may be sent at full speed - nevertheless, we recommend to put a short delay of 1 or 2 ms in between them.

Starting with FW 6.00, the following commands are implemented (mode dependent):

Standard mode:

'v' returns current mode ("Standard")
'P' switches to PC mode (same as pressing PC button)
'O' switches to "Classic" online mode (only for online devices)
'R' switches to doserate online mode (only for online devices)
'D' switches to dose online mode (only for online devices)

All online modes (only for online devices):

'v' returns current mode ("Online x")
'P' switches to PC mode
'X' exits online mode

Additionally in „Classic" online mode (only for online devices):

's' requests online status and pulses
'0'-'9' sets online interval

PC mode up to FW 6.1x:

'v' returns: FW version, SN, number of used bytes in the protocol memory, date and time
'c' dumps internal conversion data
'z' resets protocol pointer to beginning of memory
'i' !!! prepares a cold start and resets the device !!!
't' prepares setting of date and time in the form "DDMMYYhhmmss"
'b' dumps protocol memory
'X' exits PC mode

Changes in PC mode starting with FW 6.90:

'v' returns: FW version, CPU version, SN, number of used bytes in the protocol memory, date and time
'i' is gone (too dangerous) and was replaced by:
'N' warmstart

Changes in PC mode starting with FW 7.03:

'v' returns: FW version, SN, number of used bytes in the protocol memory, date and time

Changes in PC mode starting with FW 7.04:

'h' returns internal h-data
's' returns internal s-data

The official toolboxes can save the device communication to a so-called dump file. In case you are going to cook your own read-out software and want to create compatible dump files we recommend to proceed as following:

» Record everything sent to and received from the device. Do this in the chronologically correct order.

» Send 'P' to enter PC Mode. For older device the PC mode must be activated on the ' by pressing the PC button.

» Send 'v' to determine the device's firmware.

» If supported issue the 'h' and 's' commands.

» If supported issue the 'c' command.

» Issue the 'b' command to receive the protocol dump.

» Exit PC mode with 'X' (or by pressing the radiation button on the device)

Note 1: The data returned by 'c', 'h' and 's' (where applicable) is useless for the end user but should be included to maintain compatibility with the official toolboxes.

Note 2: When seeking vendor support regarding dumps and/or protocol data you probably will receive a reply *only* if the dump file has been created as described above.

PROTOCOL DATA:

The protocol dump is organised in lines. The format differs depending on the FW version:

For FW 5.99 and lower:

Example:

```
 0000 92 94 03 ff ff ff ff ff ff ff ff ff ff ff ff ff
 0010 ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff
...
 00f0 ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff
 0100 fe 55 19 12 07 10 f0 ff 2d 03 1e c1 f4 00 1e 00
 0110 17 00 10 00 16 00 15 00 14 00 15 00 0a 00 1c 00
...
```

First comes the address (16 Bit), then 16 protocol bytes (8 Bit) - all separated by spaces. The first 256 (0x100) bytes have to be ignored. So for the example above, the first byte of real protocol data is 0xFE, followed by 0x55, followed by 0x19, and so on...

For FW 6.00 and higher:

Example:

```
f5ef5915120114f500f5ee05000013f5ef5915120114f505f5eec0000290f50a10
0013001800 1d0014001800240024001100160017001100170013001e001f001587
...
```

There are no preceding addresses or spaces. Each line contains 33 bytes, with first 32 bytes being protocol data. The last byte contains the mod256 checksum of all preceding bytes of this line. It can be used for integrity checks (or has to be ignored otherwise). For the above example, the first protocol byte is 0xF5, followed by 0xEF, followed by 0xF5. The last byte (0x10) is the checksum.

Interpretation:

After stripping off unused junk (header, address, spaces or checksum) the real protocol data remains. It consists of either 2 byte pulse entries or special codes. It must be interpreted byte by byte. Special codes start with a byte with its high-nibble being 0xF. If the high-nibble is not 0xF, this byte will be the first byte of a 2 byte pulse entry.

This 2 byte pulse entry represents the number of pulses collected during the last protocol interval (or the number of pulses during an out-of-band protocol interval, see below). The highest 5 bits contain the exponent, the lower 11 bits the mantissa. E.g.:

0x3E27 = %0011_1110_0010_0111 =
2^7 (exponent) * 1575 (mantissa) = 201600

THE SPECIAL CODES CONSIST OF ONE OR MORE BYTES. THEIR MEANING DEPENDS ON THE FIRMWARE VERSION:

For FW < 6.00:

0xF0 User selected a protocol interval of 1 week
0xF1 User selected a protocol interval of 1 day
0xF2 User selected a protocol interval of 1 hour
0xF3 User selected a protocol interval of 10 minutes
0xF4 User selected a protocol interval of 1 minute
0xFC Dose rate overflowed (> 1000 uSv/h) during the current protocol interval at least once.
0xFE Timestamp, 5 bytes following: mmhhDDMMYY
0xFF Out-of-band protocol interval. This occurs when the user changed the protocol interval. 2 bytes following - indicating the number of minutes the last protocol interval was active (before having been interrupted due to the interval change). Normally followed by a 2 byte pulse entry.

For 6.00 <= FW <= 6.016:

0xF0 User selected a protocol interval of 1 week
0xF1 User selected a protocol interval of 3 days
0xF2 User selected a protocol interval of 1 day
0xF3 User selected a protocol interval of 12 hours
0xF4 User selected a protocol interval of 2 hours
0xF5 User selected a protocol interval of 1 hour
0xF6 User selected a protocol interval of 30 minutes
0xF7 User selected a protocol interval of 10 minutes
0xF8 User selected a protocol interval of 5 minutes
0xF9 User selected a protocol interval of 2 minutes
0xFA User selected a protocol interval of 1 minute
0xFB User selected a protocol interval of 30 seconds
0xFC User selected a protocol interval of 10 seconds
0xFD Dose rate overflowed (> 1000 uSv/h) during the current protocol interval at least once.
0xFE Timestamp, 5 bytes following: mmhhDDMMYY
0xFF Out-of-band protocol interval, see above. The duration is now given in multiples of 10 seconds (not minutes).

For 6.017 < FW < 6.90:

There are only two 0xF? special codes left:

0xFA Dose rate overflowed (> 1000 uSv/h) during the current protocol interval at least once.

0xF5 Generic special code. The following event byte determines the meaning:

0x00-0x0C protocol interval:
0x00 User selected a protocol interval of 1 week
0x01 User selected a protocol interval of 3 days
0x02 User selected a protocol interval of 1 day
0x03 User selected a protocol interval of 12 hours
0x04 User selected a protocol interval of 2 hours
0x05 User selected a protocol interval of 1 hour
0x06 User selected a protocol interval of 30 minutes
0x07 User selected a protocol interval of 10 minutes
0x08 User selected a protocol interval of 5 minutes
0x09 User selected a protocol interval of 2 minutes
0x0A User selected a protocol interval of 1 minute
0x0B User selected a protocol interval of 30 seconds
0x0C User selected a protocol interval of 10 seconds
0xEE Out-of-band protocol interval, see above
0xEF Timestamp, 5 bytes following: mmhhDDMMYY
0xF0-0xFE debug flags, must be ignored

6.90 <= FW < 7.01 were not released to public

For 7.01 <= FW <= 7.09:

The device now supports stopping the protocol. Since a protocol interval of 0 (zero) now denotes a stopped protocol, the remaining protocol intervals have shiftet +1.

Additionally, if a different protocol interval had been chosen by the user and the currently elapsed time is less than the duration of the newly selected interval, no Out-of-band event will be recorded (just the switch to the new protocol interval when it has elapsed).

A new event byte (0xED) was created to support 6 byte timestamps. Support for the older event byte for 5 byte timestamps (0xEF) might be dropped in future for 6.90+ FW.

Special codes 0xFA, 0xFB and 0xFC were converted into single bits whose sum will be added to 0xF8 if it is non-zero. This saves space in the protocol area for the price of using special codes 0xF9 with 0xFF.
The bits used are:

Bit 0: Dose rate overflowed (> 1000 uSv/h) during this protocol interval at least once.
Bit 1: Dose alarm fired at least once during this protocol interval.
Bit 2: Dose rate alarm fired at least once during this protocol interval.

resulting in:

0xF9 Dose rate overflowed
0xFA Dose alarm fired
0xFB Dose alarm fired + Dose rate overflowed
0xFC Dose rate alarm fired
0xFD Dose rate alarm fired + Dose rate overflowed
0xFE Dose rate alarm fired + Dose alarm fired
0xFF Dose rate alarm fired + Dose alarm fired + Dose rate overflowed

0xF5 Generic special code. The following event byte determines the meaning:

0x00-0x0D protocol interval:
0x00 User disabled the protocol
0x01 User selected a protocol interval of 1 week
0x02 User selected a protocol interval of 3 days
0x03 User selected a protocol interval of 1 day
0x04 User selected a protocol interval of 12 hours
0x05 User selected a protocol interval of 2 hours
0x06 User selected a protocol interval of 1 hour
0x07 User selected a protocol interval of 30 minutes
0x08 User selected a protocol interval of 10 minutes
0x09 User selected a protocol interval of 5 minutes
0x0A User selected a protocol interval of 2 minutes
0x0B User selected a protocol interval of 1 minute
0x0C User selected a protocol interval of 30 seconds
0x0D User selected a protocol interval of 10 seconds
0xED Timestamp, 6 bytes following: ssmmhhDDMMYY
0xEE Out-of-band protocol interval, see above
0xEF Timestamp, 5 bytes following: mmhhDDMMYY

0xF8 This is a new internal special byte. The following size byte denotes the amount of additionally following bytes (including this size byte but not including the special byte 0xF8) which have to be ignored and skipped over in the protocol stream.

Starting with FW 7.10:

New event byte indicating which conversion data set is active:

0xEA Standard conversion data set (Cs137) active
0xEB Alternative conversion data set (Co60) active

GAMMA-SCOUT GmbH & Co. KG

Kirchweg 49
D-50858 Köln
Germany

Tel.: +49 (0) 221 / 500 618 80
info@gamma-scout.com
www.gamma-scout.com

Stand Juni 2024