External Interface Control Document

for the

GQ Electronics LLC

GMC-300 Model

Geiger-Muller Counter

Revision History

| Revision | Date | Description |
| --- | --- | --- |
| A | 04/30/2012 | Initial Release. Produced by Phil Gillaspy. |
| B | 02/15/2013 | Set time/date commands added. |
|  |  |  |
|  |  |  |
|  |  |  |

This documentation contained herein is licensed to the public domain under terms of the GNU GPL reproduced here.

Copyright (C) 2012, 2013 Phil Gillaspy

This documentation is free: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This documentation is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

Additionally, the author, Phil Gillaspy, requires that no copyright notice nor acknowledgement to the author is necessary for any derivative software or documentation using any part or all of this documentation herein.

Table of Contents

1. Interface Overview 2

1.1 Document Overview 2

1.2 Acronyms 2

1.3 Notations 2

2.0 Physical Interface 3

3. 0 Commands and Returned Data 4

3.1 Get Version 4

3.2 Get Serial Number 4

3.3 Get CPM 4

3.3 Get CPS 5

3.4 Get Battery Voltage 5

3.5 Get History Data 5

3.6 Turn On CPS 7

3.7 Turn Off CPS 8

3.8 Get Configuration Data 8

3.9 Erase Configuration Data 11

3.10 Write Configuration Data 11

3.11 Update Configuration Data 11

3.12 Set Year 12

3.13 Set Month 12

3.14 Set Day 12

3.15 Set Hour 13

3.16 Set Minute 13

3.17 Set Second 13

3.18 Send Key 14

3.19 Power Off 14

# 1. Interface Overview

This document defines a protocol to support communication between a GQ GMC Geiger Counter and a computer host via a USB port. The protocol allows a host computer to send a command to a GQ GMC geiger counter and to receive data in return. The physical medium is USB, however, the connection is utilized by imitating traditional RS-232 communications. The protocol, generally speaking, consists of transmitting a command consisting on an ASCII string of characters possibly intermixed with raw binary data serving as parameters of the command. Data is optionally returned by the GQ GMC geiger counter in response to the command from the host computer. The returned data has no start/stop delimiters, consisting only of a sequence of N bytes. The returned data is, principally, raw binary data although some returned data may include ASCII characters. Both the command data parameters and the returned data are transmitted most significant byte first.

## 1.1 Document Overview

The GQ GMC Geiger Counter Communication Protocol is presented in Section 3. Each command is described in a table with content defining the command and the returned data, if any. Optional information may be included to help clarify the command and response.

## 1.2 Acronyms

CPS counts per second

CPM counts per minute

MSB most significant byte

LSB least significant byte

## 1.3 Notations

[X] square brackets specify that quantity X is hex byte value,

square brackets are not part of command

# 2.0 Physical Interface

The software on the host computer is expected to use a USB-to-Serial device driver in order to communicate with the geiger counter. The USB-to-Serial device driver is commonly available on most operating systems.

The USB-to-Serial device driver must be configured with the following characteristics.

| Baud Rate | Data Bits | Parity | Stop Bits | Control |
| --- | --- | --- | --- | --- |
| 57600 | 8 | None | 1 | None |

A standard USB 2.0 cable connects the host computer to the GQ GMC geiger-muller counter.

# 3. 0 Commands and Returned Data

## 3.1 Get Version

The Get Version command retrieves the GQ GMC's model number and firmware revision.

| Command | Returned Data | Comment | Firmware Support |
| --- | --- | --- | --- |
| <GETVER>>  no command parameters | 14 ASCII characters | 7 byte model number & 7 byte revision, e.g.,  “GMC-300Re 2.11” | GMC-300 Rev 2.10 or later |

## 3.2 Get Serial Number

The Get Serial Number command retrieves the GQ GMC's 14 character serial number.

| Command | Returned Data | Comment | Firmware Support |
| --- | --- | --- | --- |
| <GETSERIAL>>  no command parameters | [B1][B2][B3][B4][B5][B6][B7] | 7 bytes binary data are returned, each nibble is a single hex digit of a 14 character serial number | GMC-300 Rev 2.10 or later |

## 3.3 Get CPM

The Get CPM command retrieves the current counts per minute value.

| Command | Returned Data | Comment | Firmware Support |
| --- | --- | --- | --- |
| <GETCPM>>  no command parameters | [MSB][LSB] | Example, 00 1C, returned value is 28 counts per minute | GMC-300 Rev 2.10 or later |

## 3.3 Get CPS

The Get CPS command retrieves the current counts per second value.

| Command | Returned Data | Comment | Firmware Support |
| --- | --- | --- | --- |
| <GETCPS>>  no command parameters | [MSB][LSB] | Example, 00 01, returned value is 1 count per second | GMC-300 Rev 2.15 or later |

## 3.4 Get Battery Voltage

The Get Battery Voltage command retrieves the voltage level of the GQ GMC's internal rechargeable battery.

| Command | Returned Data | Comment | Firmware Support |
| --- | --- | --- | --- |
| <GETVOLT>>  no command parameters | [B] | Voltage is equal to B converted to real number divided by 10.0, for example, B = 0x62 converts to 9.8V | GMC-300 Rev 2.10 or later |

## 3.5 Get History Data

The Get History Data retrieves the history data from the GQ GMC's internal flash memory.

| Command | Returned Data | Comment | Firmware Support |
| --- | --- | --- | --- |
| <SPIR[A2][A1][A0][L1][L0]>> | [L1][L0] number of binary bytes | A2,A1,A0 form 24 bit address.  L1,L0 form 16 bit data length | GMC-300 Rev 2.10 or later |

The user requests the data given the starting address and length of data requested from the GQ GMC's 64k byte buffer. The data logging can be set to record either CPM or CPS and the data can be recorded either every second, minute, or hour. The length parameter is the number of bytes to be read, but note that the history data itself is intermixed with special tag data indicating either (1) date/timestamp, (2) double byte data sample, or (3) ASCII tag. So the number of bytes read is not the number of data samples. You don't know the number of data samples until you have parsed the history data returned.

The date/timestamp embedded into the history buffer is a string of the form

[55][AA][00][YY][MM][DD][HH][MM][SS][55][AA][DD] where

55AA is start of sequence marker,

00 is the enumeration code that the date/timestamp follows,

YY is year, MM is month,

HH is 24 hour time, MM is minutes, SS is seconds,

55AA is end of sequence marker,

DD is indicator of saved data type where

0 = off (history is off),

1 = CPS every second,

2 = CPM every minute,

3 = CPM recorded once per hour.

The double byte data sample string is [55][AA][01][DH][DL] where

55AA is start of sequence marker,

01 is the enumeration code that two byte data follows,

DH is the MSB, DL is the LSB.

There is no 55AA end of sequence marker.

This particular special tag represents a data sample whose value exceeded 255 and thus needed two bytes. A history buffer of two byte data samples would look something like 55AA02a355AA027555AA01b8 etc. If the radiation is continuously high, then all data will be > 255 and the number of data samples in the history buffer will greatly diminish (decreased by factor of 2).

The ASCII label string is [55][AA][02][LL][CC][CC][CC][CC].... where

55AA is start of sequence marker,

02 is the enum code indicating that the tag ASCII string follows,

LL is the number of ASCII characters in the label,

CC is an ASCII character, there being LL number of ASCII characters.

There is no 55AA end of sequence marker.

The history data itself is recorded as either CPS or CPM. If CPS, the data byte is typically a sequence of 00, 01's, 02's, and occasionally a 03. If CPM, the data is typically anywhere from 0x0a (10 decimal) up to 0x1e (30 decimal). But remember that any time the count per time exceeds 255, then the special [55][AA][DH][DL] sequence kicks in.

The history data continues until either one or both of the date/timestamp or label tag special sequences are inserted into the history buffer.

If the user requests an area of the history buffer which has no data as yet recorded, then the start of the history buffer is used, in other words, it is as if the user requested address zero.

If the user requests a small area of the history buffer, then chances are high that the data retrieved will have no date/timestamp embedded. It is then impossible to know when the data was recorded. The 64K history buffer is divided into 4K blocks and it is guaranteed that there will be a date/timestamp somewhere within each 4K block. For these reasons, the user should request no less than 4K bytes on 4K byte boundary. Note that 4K bytes is also the maximum request allowed. So for all practical purposes, all get history commands should be 4K bytes. It would be wise to collect all 64K of the history buffer for parsing.

The following is a typical history buffer log of counts per second (leading 0 is suppressed). Note that the date/timestamp is embedded about half way through. The timestamp of the data preceding the date/timestamp is unknown until a further search backward in the history buffer discovers the previous date/timestamp.

1 0 1 1 0 0 0 1 0 1 0 0 0 0 2 0 0 0 0 0 0 2 0 0 0 2 0 0 0 0 0 0

0 1 0 0 0 0 0 0 1 0 0 2 0 0 1 0 1 0 2 0 0 0 1 2 1 0 0 0 1 0 1 1

1 1 0 0 0 0 0 0 1 1 0 1 0 0 0 1 0 4 1 0 0 0 0 1 0 0 2 2 0 2 0 0

0 2 1 1 0 0 1 1 0 0 1 1 2 0 0 3 2 0 1 0 0 0 2 0 0 2 0 0 0 0 0 0

0 0 1 2 0 0 0 55 aa 0 c 4 1 11 1f a 55 aa 1 1 0 0 0 0 0 0 1 0 0 0 1 1

0 1 0 0 0 0 0 0 0 0 0 0 0 1 0 1 1 1 1 1 0 0 0 1 0 1 0 0 0 0 0 1

1 0 0 0 0 0 0 0 1 0 0 1 0 2 1 0 1 2 1 1 0 1 1 0 0 0 0 0 0 0 0 0

0 0 1 0 0 0 0 0 0 2 0 0 0 0 0 1 1 3 0 0 2 0 0 0 0 0 0 1 1 0 1 0

The following is a typical history buffer log of counts per minute (leading 0 is suppressed) changed over to counts per minute. Note that there are two date/timestamps(2012/April 2/17:15). The first indicates the current logging is set to counts per second, but the second changes that to counts per minute. Also notice, there is a single two byte sample (55AA021b). The 0xff data indicates an area of the history buffer that has no data recorded.

0 0 0 0 0 0 1 0 0 0 1 1 0 0 0 0

1 3 0 0 0 0 0 1 1 0 0 2 0 0 0 0

0 0 55 aa 0 c 4 2 11 e 34 55 aa 1 55 aa

0 c 4 2 11 e 35 55 aa 2 1b 16 12 18 18 16

14 13 18 13 18 24 a6 ff ff ff ff ff ff ff ff ff

ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff

## 3.6 Turn On CPS

The Turn On CPS command enables the GQ GMC to send counts per second data to the host computer every second automatically. This automated return data will continue until the Turn Off CPS command is received by the GQ GMC.

| Command | Returned Data | Comment | Firmware Support |
| --- | --- | --- | --- |
| <HEARTBEAT1>> | [MSB][LSB] | A 14 bit unsigned integer is returned once a second. The highest two order bits should be set to zero (i.e., masked out). | GMC-300 Rev 2.10 or later |

## 3.7 Turn Off CPS

The Turn Off CPS command disables the GQ GMC from automatically transmitting the CPS once per second.

| Command | Returned Data | Comment | Firmware Support |
| --- | --- | --- | --- |
| <HEARTBEAT0>> | None | There is no returned data. | GMC-300 Rev 2.10 or later |

## 3.8 Get Configuration Data

The Get Configuration Data command retrieves the total 256 bytes of the GQ GMC's configuration data.

| Command | Returned Data | Comment | Firmware Support |
| --- | --- | --- | --- |
| <GETCFG>> | [X] ….., 256 bytes | Returned data is all 256 bytes of configuration data. | GMC-300 Rev 2.10 or later |

The configuration data consists of the following parameters in the order listed.

| Byte Number | Parameter | Description |
| --- | --- | --- |
| 0 | CFG\_PowerOnOff |  |
| 1 | CFG\_AlarmOnOff |  |
| 2 | CFG\_SpeakerOnOff |  |
| 3 | CFG\_GraphicModeOnOff |  |
| 4 | CFG\_BackLightTimeoutSeconds |  |
| 5 | CFG\_IdleTitleDisplayMode |  |
| 6 | CFG\_AlarmCPMValueHiByte |  |
| 7 | CFG\_AlarmCPMValueLoByte |  |
| 8 | CFG\_CalibrationCPMHiByte\_0 |  |
| 9 | CFG\_CalibrationCPMLoByte\_0 |  |
| 10 | CFG\_CalibrationSvUcByte3\_0 |  |
| 11 | CFG\_CalibrationSvUcByte2\_0 |  |
| 12 | CFG\_CalibrationSvUcByte1\_0 |  |
| 13 | CFG\_CalibrationSvUcByte0\_0 |  |
| 14 | CFG\_CalibrationCPMHiByte\_1 |  |
| 15 | CFG\_CalibrationCPMLoByte\_1 |  |
| 16 | CFG\_CalibrationSvUcByte3\_1 |  |
| 17 | CFG\_CalibrationSvUcByte2\_1 |  |
| 18 | CFG\_CalibrationSvUcByte1\_1 |  |
| 19 | CFG\_CalibrationSvUcByte0\_1 |  |
| 20 | CFG\_CalibrationCPMHiByte\_2 |  |
| 21 | CFG\_CalibrationCPMLoByte\_2 |  |
| 22 | CFG\_CalibrationSvUcByte3\_2 |  |
| 23 | CFG\_CalibrationSvUcByte2\_2 |  |
| 24 | CFG\_CalibrationSvUcByte1\_2 |  |
| 25 | CFG\_CalibrationSvUcByte0\_2 |  |
| 26 | CFG\_IdleDisplayMode |  |
| 27 | CFG\_AlarmValueuSvByte3 |  |
| 28 | CFG\_AlarmValueuSvByte2 |  |
| 29 | CFG\_AlarmValueuSvByte1 |  |
| 30 | CFG\_AlarmValueuSvByte0 |  |
| 31 | CFG\_AlarmType |  |
| 32 | CFG\_SaveDataType | 0 = OFF, 1 = CPS, 2 = CPM,  3 = CPM averaged per hour |
| 33 | CFG\_SwivelDisplay |  |
| 34 | CFG\_ZoomByte3 |  |
| 35 | CFG\_ZoomByte2 |  |
| 36 | CFG\_ZoomByte1 |  |
| 37 | CFG\_ZoomByte0 |  |
| 38 | CFG\_SPI\_DataSaveAddress2 | 1st of 3 bytes forming a 24 bit address in the history buffer of the starting sample of the most recent data logging run. |
| 39 | CFG\_SPI\_DataSaveAddress1 |  |
| 40 | CFG\_SPI\_DataSaveAddress0 |  |
| 41 | CFG\_SPI\_DataReadAddress2 |  |
| 42 | CFG\_SPI\_DataReadAddress1 |  |
| 43 | CFG\_SPI\_DataReadAddress0 |  |
| 44 | CFG\_nPowerSavingMode |  |
| 45 | CFG\_nSensitivityMode |  |
| 46 | CFG\_nCounter\_Delay\_HiByte |  |
| 47 | CFG\_nCounter\_Delay\_LoByte |  |
| 48 | CFG\_nVoltageOffset |  |
| 49 | CFG\_Max\_CPM\_HiByte |  |
| 50 | CFG\_Max\_CPM\_LoByte |  |
| 51 | CFG\_nSensitivityAutoModeThreshold |  |
| 52 | CFG\_Save\_DateTimeStamp6 | 1st of 6 bytes of Date/Timestamp. The 6 bytes of date/timestamp are always placed at end of configuration data. 1St byte is last two digits of year. |
| 53 | CFG\_Save\_DateTimeStamp5 | month of year |
| 54 | CFG\_Save\_DateTimeStamp4 | day of month |
| 55 | CFG\_Save\_DateTimeStamp3 | hour of day |
| 56 | CFG\_Save\_DateTimeStamp2 | minute of hour |
| 57 | CFG\_Save\_DateTimeStamp1 | second of hour |
| 58 | CFG\_MaximumBytes | Value is always 0xff.  CFG\_MaximumBytes is always placed at end of configuration data. |
| 59 | Spare |  |
| 60 | Spare |  |
| 61 | Spare etc through byte number 255 |  |

## 3.9 Erase Configuration Data

The Erase Configuration Data causes all 256 bytes of configuration data to be reset to factory default values. Individual parameters of the configuration data cannot be erased or reset.

| Command | Returned Data | Comment | Firmware Support |
| --- | --- | --- | --- |
| <ECFG>> | [AA] | 0xAA is returned data. | GMC-300 Rev 2.10 or later |

## 3.10 Write Configuration Data

Write Configuration Data assigns a byte of configuration data as specified by the address of the parameter to the value given in the command.

| Command | Returned Data | Comment | Firmware Support |
| --- | --- | --- | --- |
| <WCFG[A0][D0]>> | [AA] | 0xAA is returned data. [A0] is offset of byte in configuration data. [D0] is the assigned value of the byte. | GMC-300 Rev 2.10 or later |

Only one byte of configuration data can be written at a time. Multibyte configuration parameters will require multiple invocations of this command.

## 3.11 Update Configuration Data

The Update Configuration Data causes changes to the configuration data to take effect. Any number of changes to configuration data can be made using Write Configuration Data command, but none of those changes cause the GQ GMC to change operation until after GQ GMC receives the Update Configuration Data command.

| Command | Returned Data | Comment | Firmware Support |
| --- | --- | --- | --- |
| <CFGUPDATE>> | [AA] | 0xAA is returned data. | GMC-300 Rev 2.15 or later |

## 3.12 Set Year

The Set Year command sets the year for the internal real time clock of the GQ GMC.

| Command | Returned Data | Comment | Firmware Support |
| --- | --- | --- | --- |
| <SETDATEYY[D0]>> | [AA] | 0xAA is returned data.  D0 is the value of the year in hexadecimal. For example, 0x60 is the year 96. The century is implied from context. | GMC-300 Rev 2.23 or later |

## 3.13 Set Month

The Set Month command sets the month of the year for the internal real time clock of the GQ GMC.

| Command | Returned Data | Comment | Firmware Support |
| --- | --- | --- | --- |
| <SETDATEMM[D0]>> | [AA] | 0xAA is returned data.  D0 is the value of the month in hexadecimal. For example, 0x0C is the 12th month. | GMC-300 Rev 2.23 or later |

## 3.14 Set Day

The Set Day command sets the day of the month for the internal real time clock of the GQ GMC.

| Command | Returned Data | Comment | Firmware Support |
| --- | --- | --- | --- |
| <SETDATEDD[D0]>> | [AA] | 0xAA is returned data.  D0 is the value of the day in hexadecimal. For example, 0x1C is the 28th day of month. | GMC-300 Rev 2.23 or later |

## 3.15 Set Hour

The Set Hour command sets the hour of the day for the internal real time clock of the GQ GMC.

| Command | Returned Data | Comment | Firmware Support |
| --- | --- | --- | --- |
| <SETTIMEHH[D0]>> | [AA] | 0xAA is returned data.  D0 is the value of the hour in hexadecimal. For example, 0x12 is the 18th hour of the day. Hour is a 24 hour clock counting from 0 to 23. | GMC-300 Rev 2.23 or later |

## 3.16 Set Minute

The Set Minute command sets the minute of the hour for the internal real time clock of the GQ GMC.

| Command | Returned Data | Comment | Firmware Support |
| --- | --- | --- | --- |
| <SETTIMEMM[D0]>> | [AA] | 0xAA is returned data.  D0 is the value of the minute in hexadecimal. For example, 0x25 is the 37th minute of the hour. Minutes count from 0 to 59. | GMC-300 Rev 2.23 or later |

## 3.17 Set Second

The Set Second command sets the second of the minute for the internal real time clock of the GQ GMC.

| Command | Returned Data | Comment | Firmware Support |
| --- | --- | --- | --- |
| <SETTIMESS[D0]>> | [AA] | 0xAA is returned data.  D0 is the value of the second in hexadecimal. For example, 0x33 is the 51st second of the minute. Seconds count from 0 to 59. | GMC-300 Rev 2.23 or later |

## 3.18 Send Key

The Send Key emulates the pushbuttons on the front panel of the GQ GMC. There are four such keys, named S1, S2, S3, & S4. See the GQ GMC User Manual for further clarification.

| Command | Returned Data | Comment | Firmware Support |
| --- | --- | --- | --- |
| <KEY0>> for key S1  <KEY1>> for key S2  <KEY2>> for key S3 <KEY3>> for key S4 | None |  | GMC-300 Rev 2.10 or later |

## 3.19 Power Off

The Power Off command causes the GQ GMC to turn off.

| Command | Returned Data | Comment | Firmware Support |
| --- | --- | --- | --- |
| <POWEROFF>> | None |  | GMC-300 Rev 2.10 or later |