GeigerLog Manual

by ullix

Version 1.5.0

May 2024

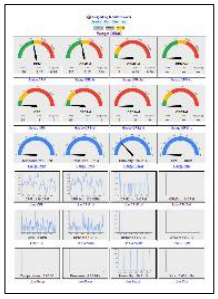
What's New in GeigerLog 1.5 ?

- **Auto-Installer** – no more fiddling with Pip
- **SerialPulse Device** – an easy way to pulse counting on **any** computer
- **RadPro Device** – an Open Source firmware enhances Geiger counters' performance
- **IoT** – more IoT support, now including TASMOTA based switchable AC plugs
- **Formula Interpreter** – much extended, with Para and Nopa functions
- **Alarm Alerts** – GeigerLog can sound an alarm and/or send you an email on alarm
- **Geomagnetic Sensor** – increasing the supported I2C family
- **Raspberry Pi** – new hardware **Raspi5** gives excellent results running GeigerLog

What's New in GeigerLog 1.4 ?

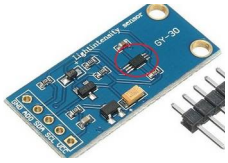
- **More things "auto"** – GeigerLog will guess more things and be right most of the time.
- **Auto-Starting** – GeigerLog can start automatically, e.g. after a computer reboot.
- **Plug and Play** extended to allow multiple GMC counters on a single computer, distinguishable by name and/or serial number.
- More **Threaded processing** to allow more parallelism to cope with ill-behaved counters.
- **Monitoring GeigerLog from your Smartphone** greatly expanded; configurable graphics can now be seen on the Smartphone

- **Widgets for remote monitoring** are offered by GeigerLog and can be used in your own, customized web page. They can even be run as SSL-secured pages.



- **PIN Diode Geiger counters** supported as I2C device.

- **Ambient Light Sensor BH 1750** (=**GY-30**) supported as I2C device.



- **Raspberry Pi – or Raspi** for short **– Support** – was greatly enhanced. You can either run GeigerLog on Raspi using Raspi specific features, or you can use the Raspi as a **satellite companion** to GeigerLog.

What's New in GeigerLog 1.3 ?

- **WiFi enablement** of GeigerLog is the most significant change in this release!

  You can now **monitor GeigerLog from your Smart-phone**, and you can manage WiFi enabled devices such as **WiFiClient Devices** (like a GMC counter) and **WiFiS-erver Devices.**

  It also allows enabled devices to **submit data wirelessly** to GeigerLog.

- The **I2C Device** support has been greatly expanded, now supporting several dongles and new I2C sensors, in particular for **CO2** measurement with the sen-sors **SCD30** and **SCD41**.

  The new combinations remain cross-platform compatible.

- A **Manu Device** has been added, which allows to enter data manually, helpful when you can't get the data electronically, like ruler measured distance of radioactive source from counter, voltage from your DVM, temperature from a home thermometer.

- **Plug and Play** for devices using a USB-To-Serial connection (like GMC counter, Gamma-Scout counter, I2C devices). Their settings can now be auto-detected.

- A **LinFit** (Linear Regression) option has been added to the Graph tools to allow quickly judging the stability of the data.

- More cleanup of the Graphical User Interface

- A lot of refactoring under-the-hood of GeigerLog.

What's New in GeigerLog 1.2.1 ?

- GeigerLog now supports all current **Gamma-Scout Counters**, including the lat-est "Online" model, the only one of the breed which allows logging.

- A **Simul Device** has been added, which creates its "counts" with a Poisson random number generator. Helpful for understanding the workings of a Geiger counter and GeigerLog.
- **Search the entire NotePad** for the occurrence of a text like 'abc' or '123.456'
- The tool **GLpipcheck** was improved

- The Easter Holidays are approaching, and fittingly, an **Easter egg** can now be found in GeigerLog. Do your best – perhaps you will see a dancing **GEIGERA**?

What's New in GeigerLog 1.2 ?

- withdrawn due to a bug with Windows 10

What's New in GeigerLog 1.1 ?

- some major **refactoring** of the code to prepare for future extensions. Most of this will hopefully not be visible to users ;-).
- GeigerLog now supports **HiDPI monitors**
  - HiDPI are High-Resolution monitors with a pixel density higher than FullHD). An example is shown for an 8k (!) monitor. See page 127.
- Editing the GMC counter's **internal configuration**
  - It allows to easily enter SSID, passwords, websites, IDs at your computer and send it to the GMC counter's internal configuration menu with a mouse click. See page 66.
- Support of **MiniMon** devices, primarily CO2 monitors
  - Currently this is supported only for the Linux operating system! See page 91.

What's New in GeigerLog 1.0 ?

Short answer: very little! That's why it became the 1.0 release.

- Installation has again become a little easier, in particular on Windows.
- A guidance for the installation of GeigerLog on a Raspberry Pi has been added to this manual, including a use example with a GMC-300E+ counter
- One thing, however, has gotten a major revision, which is the **Calibration Factor**. It is now the **inverse** of the old definition:

  new Calibration Factor = 1 / old Calibration Factor

  Why the change? To make things easier. Read more in chapter Appendix G – Calibration on page 164.

Recommended Reading on the subject from the same author:

All available on the SourceForge site under folder "**Articles**":
https://sourceforge.net/projects/geigerlog/

GeigerLog - Potty Training for Your Geiger Counter

This article is about the use of natural Potassium to give your Geiger counter a little bit of a training workout when you get tired of measuring just the background. Potassium is omnipresent on the earth, essential for all life, may already be available in or around your home or garden, and has a little bit of natural radioactivity – though well below any danger zones. I show how to best use it, taking advantage of GeigerLog on today's Geiger counter technology and software.

GeigerLog - Going Banana

Ever heard the term 'banana equivalent dose'? It refers to the Potassium content of bananas, which gives the bananas a tiny little bit of radioactivity. Nevertheless, I demonstrate that you can measure this with a Geiger counter, but it is tricky as the activity is very low and demands in-depth statistical considerations.

GeigerLog - Review Smart Geiger Pro (SGP-001)

The **Smart Geiger Pro (SGP-001)** is a semiconductor detector for radioactivity, i.e. it is NOT using a Geiger-Müller tube, and, despite its name, not a Geiger counter, but can measure radioactivity. While it is designed to plug into the headphone plug of a smartphone, the present GeigerLog version 0.9.90 allows to use it connected to a personal computer.

GeigerLog – AudioCounter Support

Some Geiger counters – especially very old ones and modern low-cost varieties – generate audio-clicks for each registered radioactive event. But even the very modern semiconductor based radioactivity detector **Smart Geiger Pro (SGP-001).** GeigerLog now fully supports those audio counters. In the article a GMC-300E+ counter, connected digitally and via audio simultaneously, demonstrates that the results are valid.

GeigerLog - Radiation-v1.1(CAJOE)-Support

The **Radiation-v1.1 (CAJOE)** Geiger counter – called **Cajoe-Counter** for short – is a low cost Geiger counter, which can only generate audio-clicks for each registered radioactive event. Die article gives some details on the counter which are not easily available, and shows how to connect GeigerLog to it.

Review of USB-To-I2C Dongles as used by GeigerLog

To connect any I2C device to a desktop or laptop computer requires some hardware, called a dongle. Several were reviewed. The best was the USB-ISS dongle.

Measuring CO2 with SCD30, SCD41, and MiniMon

CO2 measurements with GeigerLog are possible with the I2C devices SCD30 and SCD41, as well as with the MiniMon. All do what they are supposed to do. The SCD41 offers the option of an extended measurement range.

GeigerLog – Review PIN Diode Geiger Counters

PIN diodes like the **Smart Geiger Pro (SGP-001)** or the **Gamma Radiation Sensor Module (GDK-101)** semiconductor devices employed as detector for radioactivity. One of their advantages is their smallness, which, however, is also their weakness, as their sensitivity to radioactivity is much lower than classical Geiger-Müller tube equipped devices.

GeigerLog – Deadtime Correction of Geiger-Counter Events

The application of a Deadtime-Correction to a Geiger-Counter is reassessed. As a count in a Geiger tube is a so-called "**paralytic**" event, a more complex formula is needed.

(to be published)

HOWTOs

HOWTO - Using Python in a virtual Environment on Linux-v1.0

When there is a need to use multiple versions of Python, or multiple configurations, you need to Python in its own virtual environment. Can be done surprisingly easily. Here shown for Linux.

HOWTO - Using PyQt5 and matplotlib on HiDPI monitors (Python3)

The use of HiDPI monitors is still a challenge. GeigerLog can now run on 8k monitors!

HOWTO - Read-Write Permissions Serial Port on Linux

Setting the Read-Write Permissions, relevant to using USB-To-Serial connections.

HOWTO - Using Pythons Pip

Pip is an essential tool for using Python in more than the most basic application…

Table of Contents

Table of Figures

Overview

GeigerLog is a combination of data **logger**, data **presenter**, and data **analyzer**.

It is based on **Python (Version 3)**, hence it runs on Linux, Windows, Macs, and other systems. In particular it also runs on the **Raspberry Pi** (**Raspi** for short) computers, where GeigerLog takes advantage of their GPIO – connector, something not available on any other system!

GeigerLog had initially been developed for the sole use with Geiger counters, but has now become a more universal tool, which equally well handles environmental data like temperature, barometric-pressure, humidity, CO2, and light, and is ready for future sensors. In its present state it can e.g. be deployed as a monitor for a remote weather station, including monitoring of CO2, complemented with a Geiger counter to monitor radioactivity.

GeigerLog itself can be monitored by Smartphone via WiFi.

The most recent version of GeigerLog, including the manual, can be found at project GeigerLog at SourceForge: https://SourceForge.net/projects/geigerlog/.

Currently Supported Devices

GMC Devices:

GeigerLog continuous to support the Geiger counters from GQ Electronics [1]) **GMC-3xx**, **GMC-5xx**, and **GMC-6xx** line, including the variants with an additional 2nd Geiger tube, both by wire and wireless connection.

These devices can store up to several weeks of recordings in their internal memory. GeigerLog can read this internal memory.

AudioCounter Devices:

Any Geiger counter which produces audible clicks that can be fed into a computer via microphone-in or line-in can now be recorded and logged by GeigerLog. Many low-cost Geiger counters produce audio-clicks only, like the reviewed **Radiation-v1.1(CAJOE)** [2]) counter.

An audio connection also provides an alternative way to connect to the **GMC** counters, e.g. in case their USB connection fails.

In particular, GeigerLog now allows to use the interesting **Smart Geiger Pro (SGP-001)** [3]) semiconductor Geiger counter with a Personal Computer, which has so far not been possible!

---

1 GQ Electronics LLC, 1001 SW Klickitat Way,Suite 110, Seattle, WA 98134,USA, http://www.gqelectronicsllc.com/

2 GeigerLog-Radiation-v1.1(CAJOE)-Support https://sourceforge.net/projects/geigerlog/files/Articles/GeigerLog-Radiation-v1.1%28CAJOE%29-Support-v1.0.pdf/download

3 GeigerLog – Review Smart Geiger Pro (SGP-001) https://sourceforge.net/projects/geigerlog/files/Articles/GeigerLog-Review%20Smart%20Geiger%20Pro%20%28SGP-001%29-v.1.0.pdf/download

IoT Devices:

GeigerLog supports IoT devices. Both a generic type and TASMOTA based switchable AC plugs are supported.

A Raspberry Pi can be used a dataserver to transmit data via the MQTT protocol as an IoT device.

RadMon Devices:

GeigerLog supports the **RadMon+** [4]) hardware, which can provide a Geiger counter as well as an environmental sensor for temperature, barometric-pressure, and humidity.

These devices acts as IoT (Internet of Things) devices, and transmit their data wirelessly

AmbioMon Devices:

GeigerLog supports the **AmbioMon** [5]) hardware, which can provide a Geiger counter as well as an environmental sensor for temperature, barometric-pressure, humidity, and air-quality, driven by an ESP32 microprocessor.

These devices can be controlled via smartphone, and transmit their data wirelessly.

Gamma Scout Devices:

GeigerLog fully supports the **Gamma-Scout** [6]) devices ***Standard, Alert, Rechargeable,*** and ***Online,*** including logging with the Online device.

These devices can store recordings in their internal memory. GeigerLog can read this memory.

LabJack Devices:

GeigerLog supports the **Labjack** [7]) hardware U3 in combination with the ei1050 probe for temperature and humidity.

MiniMon Device:

GeigerLog supports the **MiniMon** hardware, devices for measuring in-house CO_2 levels using the infrared based NDIR technology.

Formula Device:

This device allows the use of GeigerLog's **Formula Interpreter** to create synthetic data – like "counts" by a **random number generators (Poisson, Normal, Exponential)** – as well as technical data on GeigerLog's workings.

It also allows for count rate corrections from **Paralytic** and **Non-Paralytic** counters.

---

4 DIYGeigerCounter https://sites.google.com/site/diygeigercounter/

5 This device is in development and not yet publicly available

6 https://www.gamma-scout.com/en/

7 https://labjack.com/

Manu Device:

A Manu Device allows to enter data **manually**, helpful when you can't get the data electronically, like distance of radioactive source from counter measured with a ruler, but would like to keep the data together and plot them.

WiFiClient Device:

GeigerLog acts as a **server** to which devices can connect and deliver data. Any of GQ's WiFi-enabled GMC counters can act as a WiFiClient.

WiFiServer Device:

GeigerLog acts as a **client** to request data from an external device, which is acting as a server and delivers data when a client, such as GeigerLog, is requesting them.

I2C and RaspiI2C Devices:

GeigerLog can handle I2C based sensors by two means:

- Connected via **USB-To-I2C dongles** to your laptop or desktop computer. GeigerLog supports several of them, preferred is the **USB-ISS** dongle.
- Connected to a **Raspberry Pi** computer, which natively supports the I2C protocol on its GPIO pins. GeigerLog will be running on the Raspi computer itself; no further dongle is needed.

Supported I2C devices: **LM75** (temperature), **BME280** (temperature, barometric pressure, humidity), **BH1750** (light sensor visible), **TSL2591** (light sensor visible and infrared), **SCD30** (CO2, temperature, humidity), **SCD41** (CO2, temperature, humidity), **GDK-101** (PIN diode for detection of radioactivity), **BMM150** (Geomagnetic).

More information in recent reviews of dongles and devices [8]) [9]) [10]).

RaspiPulse Devices:

A **Raspberry Pi** computer allows to count pulses being fed to its GPIO pins. These pulses must not exceed +3.3V, the upper limit for a Raspi! GeigerLog running on the latest Raspi version 5 was found to support a count rate up to CPS=10000, far more than most Geiger tubes can deliver.

SerialPulse Devices:

These consist of a **USB-To-Serial** plug, available from $1 … $5, connected to any computer running Linux, Windows, Mac, or a Raspberry Pi computer. The UART side of these plugs is fed with pulses 0 … 3.3V … 5V. As the chips in these plugs support baudrates up to Mega-Bits per

8 Review of USB-To-I2C Dongles as used by GeigerLog
https://sourceforge.net/projects/geigerlog/files/Articles/Review%20of%20USB-To-I2C%20Dongles%20as%20used%20by%20GeigerLog-v1.0.pdf/download

9 Measuring CO2 with SCD30, SCD41, and MiniMon
https://sourceforge.net/projects/geigerlog/files/Articles/Measuring%20CO2%20with%20SCD30%2C%20SCD41%2C%20and%20MiniMon-v1.0.pdf/download

10 https://sourceforge.net/projects/geigerlog/files/Articles/GeigerLog-Review%20PIN%20Diode%20Geiger%20Counters-v.1.0.pdf/download

second, there is no upper limit for possible Geiger count rates, while the computing load remains low.

Raspi as Satellite Companion to GeigerLog:

The low computing power of a Raspi may make a different approach of the use of a Raspi attractive. This is to use a Raspi as a Satellite Companion to GeigerLog running on a more powerful desktop or laptop. In this mode Raspi will only collect data and transfer them by WiFi.

The required script **GLWiFiServer** is also provided in this release of GeigerLog. See details in chapter WiFi Devices – WiFiServer Devices on page 101.

Main Operations –
Logging, Displaying, Analyzing, Monitoring

Logging:

will be done with a user defined cycle time of 0.1 sec or longer. Each logging cycle consists of:

1. reading from the connected devices
2. saving the data into a database file
3. printing the data as a numeric values to the screen
4. and displaying the data as a live, auto-updating graph

Comments can be added to the log file before, during, and after logging.

Displaying:

means that the data are shown as a **Time-Course graph**, i.e. as a plot of value versus time. The graph uses two Y-axis:

- The **left Y-axis** is reserved for Geiger counter data, and is shown in dose rate units of either CPM / CPS or µSv/h.
- The **right Y-axis** is reserved for environmental data. If temperature data are shown, the choice of units is between °C and °F.

To display variables with very different numerical values on a common scale – like temperature (e.g. 0 … 30°C) and barometric-pressure (e.g. 970 … 1030 hPa) – the variable values can be scaled for plotting, e.g. here by subtracting 1000 from the pressure. The saved value will NOT be affected.

All scales are set automatically, but can be changed manually. Time ranges can be set to plot data only within that range and to limit any quality control analysis to only those data. These ranges can be entered manually or by left/right mouse clicks. The time can be shown as Time-of-Day, or time since first record in units of sec, min, hours, days, or auto-selected in auto mode. The graphs can be stretched, shifted, and zoomed for details, and saved as pictures in various formats (png, jpg, tif, svg, ...).

Analyzing:

is supported with several **Quality Control** tests, which can be applied to the data. Beyond the standard **statistics** – as a brief summary or more elaborate statistics – a **Poisson** test can be applied to see if the Geiger counter data are valid at all, and how well they fit to a Poisson distribution. Also, a Frequency and Autocorrelation analysis by **Fast Fourier Transform (FFT)** can be done to check for any cyclic effects in any of the measured variables. **Polynomial and Exponential Regression** can be applied to test for correlation between the data sets, and extract half-lives from Geiger data.

All manipulations of the plots, and all data analysis can be done during ongoing logging without disturbing it.

Monitoring:

allows to use a Smartphone to remotely monitor the data logged by GeigerLog. This uses the **Smartphone**'s browser; no App is needed.

Introduction to GeigerLog

Installing and Starting GeigerLog

GeigerLog requires a **Python 3** environment [11]).

It is verified to run with Python 3.8, 3.9, 3.10, 3.11, and 3.12. To be conservative, **Python 3.11** is the recommended version! Earlier versions of Python will not run. Later versions may run.

NOTE: If any problems arise, follow the Step-by-Step installation instructions for your operating system **Linux, Windows, Mac,** or **Raspberry Pi** as provided in Appendix I – Installation beginning on page 167.

Installing

The software comes in a zipped package named **`geigerlog-vXYZ.zip`** (xyz is the version number, like 1.5.0). Download the package and unzip into a directory of your choice. It creates a directory '**geigerlog**', which becomes your **working directory**, plus several subdirectories.

GeigerLog needs a **Setup** before it can be used. Follow these steps:

1. Change into the GeigerLog **working directory.**
2. Edit the configuration file **`gconfig/geigerlog.cfg`** and **activate your device(s)**, like counter and/or other devices. By default, GeigerLog has none of the devices activated!

   Example:
   to use a 'GMC-300E+' Geiger counter, search for a line beginning with '# DEVICES' (all caps), scroll down to line '[GMC_Device]', and set 'GMC_Activation=yes'. That's all.

   Take the same steps for any other device you also want to use. All devices can be activated and run simultaneously. More details in chapter Configuration of GeigerLog on page 45.
3. Make sure your computer has **internet access**
4. Run: on Linux and Mac: **`./GeigerLog.sh setup`**
   `on Windows:` **`GeigerLog.bat setup`**

Starting

Once the setup has finished, s**tart GeigerLog** with these commands:

on Linux and Mac: **`./GeigerLog.sh`**
`on Windows:` **`GeigerLog.bat`**

For advanced Auto-Starting options see chapter Auto-Starting GeigerLog on page 48.

11 GeigerLog will NOT run on Python2! The last GeigerLog version running on Python version 2.X is 0.9.06. Use this if you can't use Python3, but upgrading to Python3 is strongly suggested!

The GeigerLog Window

GeigerLog has a single window with predefined usage areas. Figure 1 gives an overview.

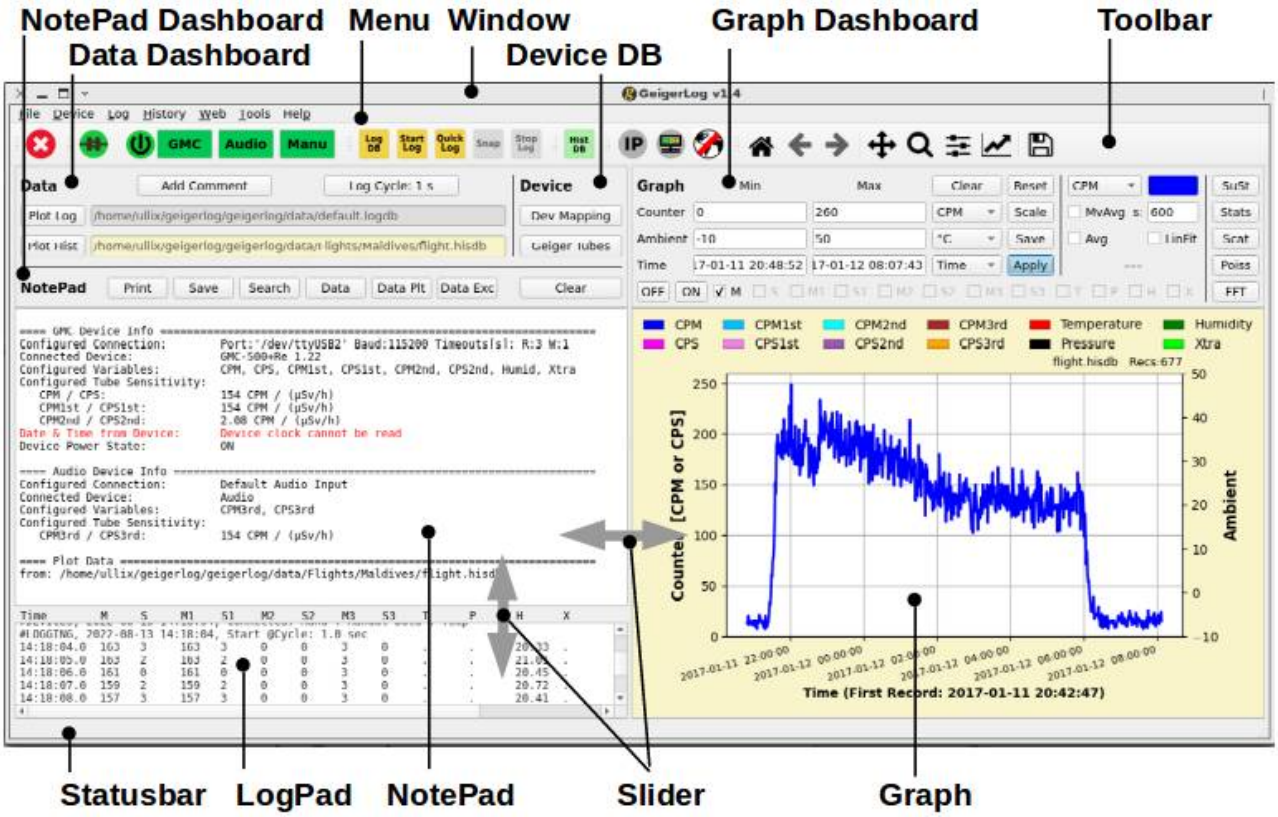


Figure 1: GeigerLog Window with Annotations

Window: The start-up window.

Menu: Menu items; some with keyboard shortcuts in the form of CTRL-X.

Toolbar: A toolbar with icons for quick mouse access to the more frequent actions.

Data Dashboard: Manage Log and/or History files, Timings, Tube Settings.

Device Dashboard: Device related settings.

NotePad Dashboard: Handling the NotePad.

Graph Dashboard: Settings to configure your graph.

Statusbar: The bottom line of the window for status and error messages.

LogPad: During logging you find here log values since last start of logging.

NotePad: A scratch-pad type of area for various textual and numeric information.

Slider: change the size of the sub-windows with your mouse to make space where you need it

Graph: Time-Course Graphs will be shown here.

Quick Tour of GeigerLog

GeigerLog is best shown with a quick demo guiding you through typical usage steps. GeigerLog can be run without a connected device in order to analyze **existing** data:

With GeigerLog running, click on menu 'History' → 'Get History from Database' and select **flight.hisdb**. The original data from an international flight from Germany to the Maldives will be loaded and displayed as a graph, showing the Time Course of CPM versus Time-of-Day.

In the Graph Dashboard, click the drop-down button currently showing **CPM**, and select **µSv/h**. The graph changes, now showing µSv/h versus Time-of-Day. Now select the Time Unit drop-down button currently showing **Time** and select **auto**. The graph switches to µSv/h versus time-since-first-record in the automatically selected unit 'hours'.

Within the graph, do a mouse-**left**-click somewhere on the vertical line near 8 h, and a mouse-**right**-click on the vertical line near 10 h. Note that the Time Min and Max fields in the **Graph** Dashboard are filled by the mouse clicks. Click the **Apply** button. The graph is zoomed-in to the descending part of the flight from about time 8 h to 10 h. You can fine tune the range with further mouse clicks, or manually edit the Time Min and Max fields, clicking Apply after changes.

Click the check button under **MovAvg**. A Moving Average is shown as an overlaid yellow-framed-line, with an averaging period of 60 sec. Since the data were collected by the Geiger counter in the 'CPM, Saving every minute' mode, which is already the average over 60 sec, no effect will be seen. Change the 60 to 600. The graph will update automatically. Now the data are averaged over 10 minutes, equal to 10 data points. Try entering numbers other than 600.

Click the **Clear** button in the **NotePad** Dashboard on the left side, then click the **Data Exc** button. Data from the beginning and the end of the flight will be printed into the NotePad.

Click the **SuSt** (Summary Statistics) button in the **Graph** Dashboard. Some brief statistics is printed into the NotePad. Click the **Stats** button for a more detailed statistics in a pop-up windows. Click **Reset**, then **hour** under **Time** drop-down button, then mouse-**left**-click on the vertical line near 10 h, and **Apply**. Then click button **Poiss**, and a 'Histogram with Poisson Fit' will be shown in a new window together with some further statistics. Click OK to close. Click button **FFT** to see an FFT analysis of the count rate data (explained later). Click OK to close.

On the toolbar click the right-most icon to save the current graph as an image file. The availability of image formats depends on your computer, but typically png, jpg, tif, and svg is available.

Click the Reset button to reset the graph to starting conditions.

The data nicely show that the background radiation, of which a good part is cosmic radiation, increases when going from ground level up to airplane cruising altitude, and up there decreases going from northern latitudes towards the equator. This is known since early last century. But at that time the radiation measuring devices had a weight of a ton mounted on a ship; today you can carry them in your shirt pocket while traveling by airplane! (Yes, the counter can be taken into the cabin.)

Running GeigerLog

This chapter explains the general approach; specific devices will be discussed later.

Establishing a GeigerLog Connection to Your Devices

To have GeigerLog interact with your device, you must **establish a connection** between them. This has two requirements: The first is the hardware between the device and the computer, the second is the software activation within GeigerLog.

The **hardware** could be based on a wire, like a USB cable or an Audio cable, or it could be based on a wireless connection, like WiFi (also called WLAN). The **software** requirement is that the GeigerLog configuration file `geigerlog.cfg` is properly defined for the devices you will use, and that you have selected the menu command: **Device → Connect Devices**. This last action establishes the needed software connection to the activated devices. Instead of using the menu command you could use the more convenient **Toggle Connection** button in the toolbar, in Figure 2 the second-from-left icon with a plug symbol.

The device icons turn green upon a successful connection to that device, red otherwise. Red devices are NOT available for logging; you may need to verify your configuration file `geigerlog.cfg`.

Figure 2: The Device Toolbar signaling the Connection Status

Devices are activated in the configuration file. Top: Before -, Bottom: After – establishing a software connection. The green devices are successfully connected, the red ones failed (here they were not physically connected for demonstration purposes)

By clicking on the device icons some info will be printed into the NotePad similar to this one for a GMC counter and an AudioCounter. Lines in **red** need your attention:

```
==== GMC Device ================================================================================
Configured Connection:        port:'/dev/geiger' baud:115200 timeoutR:3.0s timeoutW:1.0s
Connected Device:             'GMC-500+Re 1.22'
Configured Variables:         CPM1st, CPS1st, CPM2nd, CPS2nd
Configured Tube#1 Sensitivity:154   CPM/(µSv/h) (= 0.0065 µSv/h/CPM)
Configured Tube#2 Sensitivity:154   CPM/(µSv/h) (= 0.0065 µSv/h/CPM)
Date & Time from Device:      Device clock cannot be read
Device Power State:           ON
Fast Estimate Time:           60 sec. Okay


==== AudioCounter Device ========================================================================
Configured Connection:        Default Audio Input
Connected Device:             'AudioCounter'
Configured Variables:         CPM3rd, CPS3rd
Configured Tube Sensitivity:  154.0 CPM/(µSv/h) (0.0065 µSv/h/CPM)
```

You are now ready to start logging!

Device Mappings

Before you start logging, take a look at the Device Mappings**.** With the many device types now supported by GeigerLog, and the many variables available for recording, it is important to make sure that no variable is written-to by more than one device! The device mapping is shown in the NotePad upon connecting, and can be called up again by clicking the **Dev Mapping** button in the Device Dashboard, or by calling from the menu **Device → Show Device Mappings**.

This printout shows Device Mappings after a successful connection with no mapping problem:

```
==== Device Mappings ========================================================================
The configuration is determined in the configuration file geigerlog.cfg

Device    : CPM  CPS  CPM1st CPS1st CPM2nd CPS2nd CPM3rd CPS3rd Temp  Press Humid Xtra
-----------------------------------------------------------------------------------------
GMC       : -    -    M      M      M      M      -      -      -     -     -     -
Audio     : -    -    -      -      -      -      M      M      -     -     -     -
RadMon    : M    -    -      -      -      -      -      -      M     M     M     -
Mapping is valid
```

The GMC-Device counter collects Geiger counts at CPM1st, CPS1st, CPM2nd, and CPS2nd, the AudioCounter device at CPM3rd and CPS3rd, and the RadMon device collects counts at CPM, and temperature, barometric-pressure, and humidity at Temp, Press, and Humid. There are no conflicts.

However, the next example shows duplicate mappings, highlighted in red:

```
==== Device Mappings ========================================================================
The configuration is determined in the configuration file geigerlog.cfg
WARNING: Mapping problem of Variables
Variable CPM                  is mapped to more than one device
Variable Temp                 is mapped to more than one device
Variable Press                is mapped to more than one device
Variable Humid                is mapped to more than one device

Device    : CPM  CPS  CPM1st CPS1st CPM2nd CPS2nd CPM3rd CPS3rd Temp  Press Humid Xtra
-----------------------------------------------------------------------------------------
GMC       : -    -    M      M      M      M      -      -      -     -     -     -
Audio     : -    -    -      -      -      -      M      M      -     -     -     -
RadMon    : M    -    -      -      -      -      -      -      M     M     M     -
Simul     : M    M    -      -      -      -      -      -      M     M     M     M
Measurements are made on devices from top to bottom, and for each according to configuration.
If double-mapping of variables occurs, then the last measured variable will overwrite the
previous one, almost always resulting in useless data.
Also, Tube Sensitivities will likely be overwritten as well!
```

As any variable which is measured later in the log cycle, overwrites any previously measured one, there will generally be nonsense generated with such a mapping. Correct such mapping in the GeigerLog configuration file `geigerlog.cfg`.

Logging with GeigerLog

Once a connection is established, you can start logging.

NOTE: The GMC Geiger counters series, the Gamma-Scout counters series, and the AmbioMon+ + series are devices, which – beyond a logging mode – support another operating mode: **History**. This means GeigerLog can read the data these devices have stored in their internal memory. It will be explained in later chapters specific to these devices.

Logging means that GeigerLog gets fresh data from the devices, saves them in a database file, prints them on the screen, and plots them to a configurable graphic. GeigerLog then waits until the user specified cycle time has expired before it repeats the process.

This cycle time can be set by clicking the **Cycle** button in the **Data Dashboard**. A pop-up box allows you to enter a new cycle time of at least 0.1 seconds. A shorter cycle time cannot be entered. (see Error: Reference source not found on page Error: Reference source not found).

Before you can log, two things must have been done:

- First, you must have made a connection with at least one device
- Second, you must have loaded a log file

Assuming a connection has been made, we now need a log file. So click the **Log DB** icon in the toolbar to load an existing database file or define a new one by typing a file name. The toolbar will change and now also offer the **Start Log** icon. Click it to start logging. The toolbar will change again and allow only to stop the logging, or Snap (will be explained shortly). Other functions, which would interrupt logging, like exiting GeigerLog or loading the History from a counter, are also disabled during logging.

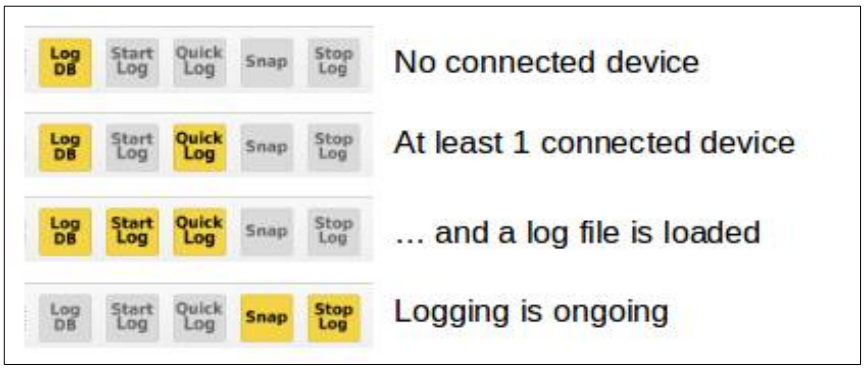


Figure 3: The Logging Toolbar's various stages

The **Quick Log** icon saves you a step by automatically using the log file `default.logdb`. However, note that this file is overwritten every time you click **Quick Log**! If you want to attach data to a previous Quick Log recording, click **Start Log** instead. **Quick Log** is very convenient if you want to just see current values, and don't care much about keeping the data.

Sometimes you may want to see fresh data right away and not wait for the next cycle. Then simply click the **Snap** icon, and GeigerLog snaps a fresh record out of order and prints it into the NotePad. Snapped records are also saved in the log database just like any other record.

The result may look like this; it uses shortcuts as in the LogPad area. A dot (“.”) represents a missing or not-configured value:

```
==== Snapped Log Record ================================================
Time        M     S     M1    S1    M2    S2    M3    S3    T      P      H      X
10:13:50.6  173   1     173   1     0     0     .     .     .      .      103.0  .
```

A note on the logging cycle when measuring Geiger counter data

The Geiger counter needs less than 1 ms (millisecond) [12]) to register and process an event which results in a count. When the counting is set to CPS (Counts per Second) the counter’s firmware sums up all events during the last second and reports this as CPS. At background radiation level there is approximately only 1 count every 3 … 4 seconds on average. But even if the count rate were much higher than background, it obviously does not make sense to sample more often than 1 second to get the ‘counts-per-**second**’. Likewise, when CPM is selected, the counts during the last minute are summed up. Hence you get all counts reaching the Geiger tube when the values are logged only once every minute.

However, this gets boring when you sit at the computer and wait for Geiger counter clicks; therefore I use a 3 second cycle time even for CPM logging just to “see some action” ;-). It also helps to watch for trends. But for long time logging you might want to set this to 60 sec or longer, and perhaps use the Moving Average (see Graph Dashboard – Visualize Your Recordings) for further smoothing the data.

This **oversampling** – sampling more often than really needed – has consequences for certain properties of the data, see Quality Control - FFT – FFT & Autocorrelation Analysis on page 53.

But keep in mind that neither oversampling nor **undersampling** – e.g. measuring a CPM value only once every 10 min – has an impact on the validity of your measured averages as long as your setup and radioactive source does not change over time. They will all be the same! This follows from the properties of Poisson distributions.

Remember: if you have set a long cycle time, and are waiting impatiently for the next reading to come up, you can always press the **Snap** button and get a reading right away!

12 Based on measurements with an oscilloscope I determined the pulse length of an GMC-300E with M4011 tube or with SBM20 tube to be about 200µs, and with SBT11A tube about 150µs, as discussed in this post: http://www.gq-electronicsllc.com/forum/topic.asp?TOPIC_ID=4598 At 200µs the maximum count rate would be under CPS=5000. However, other effects, like microprocessor cpu power, and strength of the High-Voltage generator for the Anode voltage of the Geiger tube, lower this even further.

Data Dashboard – Manage Your Recordings

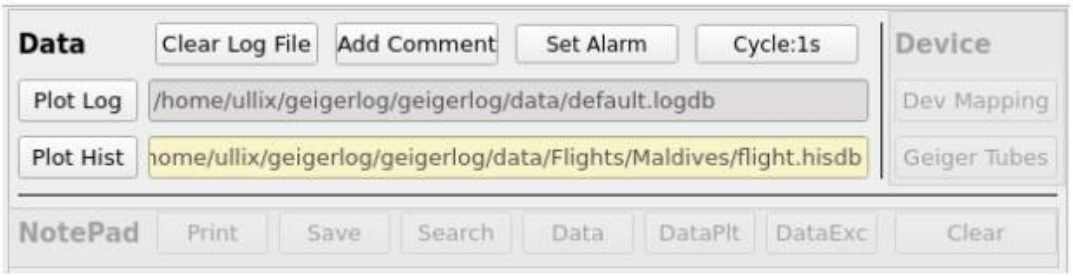


Figure 4: Data Dashboard

The Data Dashboard lets you switch between viewing the Log file and the History file. It has a button to conveniently handle the **active** file, be it Log or History, set alarms, and change the cycle time.

Database Files

One **Log** file plus one **History** file can be loaded simultaneously. Their database filenames are shown, and they can be plotted – one at a time – using the **Plot** buttons. The file with the light yellow background is the active one, currently shown in the graph.

Clear Log File

Clicking the **Clear Log File** button presents a warning, and if OK is clicked, then the contents of the current log file will be deleted, but the file remains opened for logging.

Figure 5: Clear Log File

Add Comment

Clicking the **Add Comment** button opens a dialog box allowing you to enter a comment, which will be added to the currently **active** file, which is the one on light-yellow background.

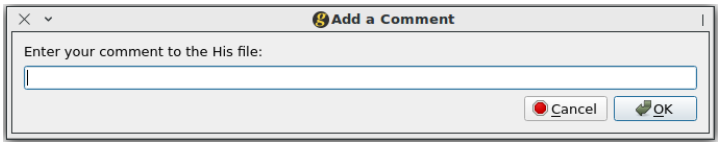


Figure 6: Add Comment to File

This is a convenience function; to enter a comment to the currently not-active file use the Log and History menus.

Set Alarm

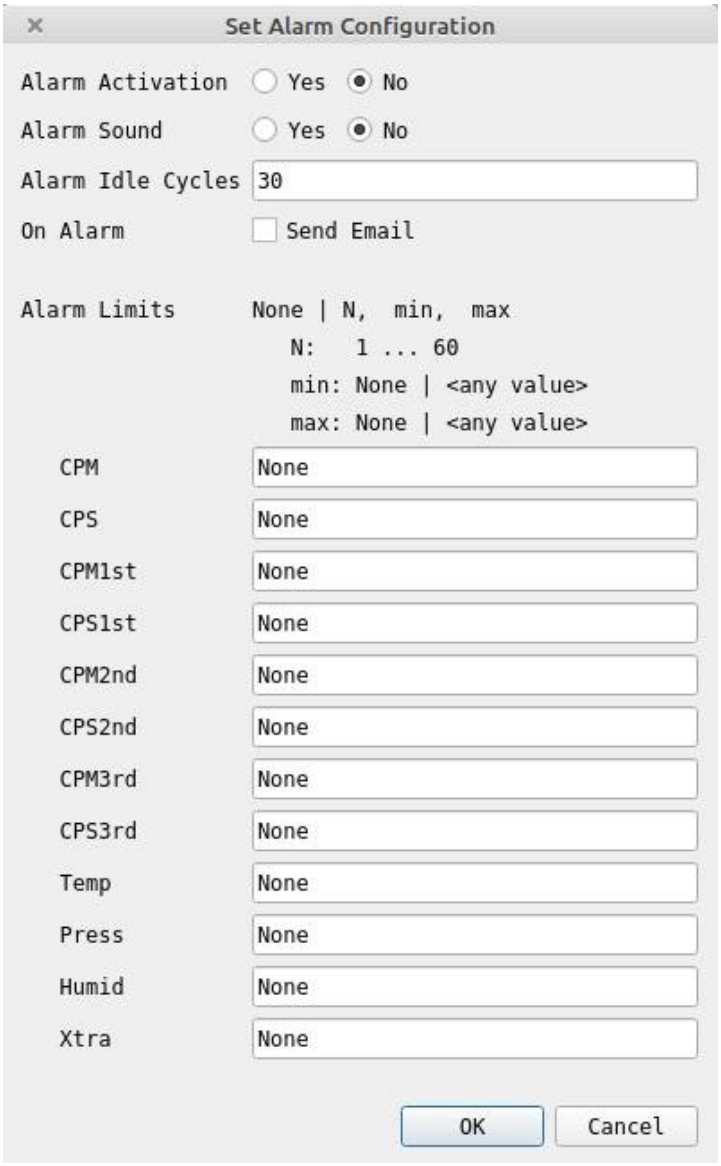


Figure 7: Set Alarm

Clicking the **Set Alarm** button opens a dialog box allowing you to enter the complete alarm configuration.

When **Alarm Activation** is No then no alarm will ever be reported. When **Alarm Sound** is No, you will never hear an alarm sound. On Yes you will hear a siren-like sound.

On **Alarm Idle Cycles** you set the first number of log cycles, during which alarm situations will be ignored. Good to allow the data to 'swing in'.

To receive an email when an alarm occurs, check the **Send Email** box. The email credentials are configured in GeigerLog's configuration file. Look there under header **[Email]** for details.

The **Alarm Limits** allow to configure each variable separately. The setting is either 'None' – then no alarm will ever be reported for this variable.

Or it is a set of 3 comma-separated values: **N, min, max:**

- **N:** the number of sequential log cycles in which an alarm-situation occurs in this variable, before an alarm is raised. E.g., with N=10, when in 9 sequential log cycles an alarm-situation occurs, but not in the 10th, then **no** alarm will be raised!
- **min:** when the variable value is lower than this value it is an alarm-situation
- **max:** when the variable value is lower than this value it is an alarm-situation

Log Cycle

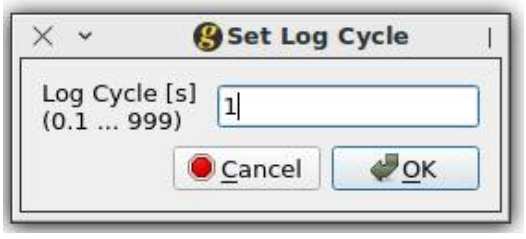


Figure 8: Set Log Cycle

The **Log Cycle** button allows to change the logging cycle time.

Clicking the button opens a dialog box where you enter the cycle time in seconds. Any number of at least 0.1 seconds is accepted; numbers less than 0.1 cannot be applied.

Device Dashboard – Manage Device Settings

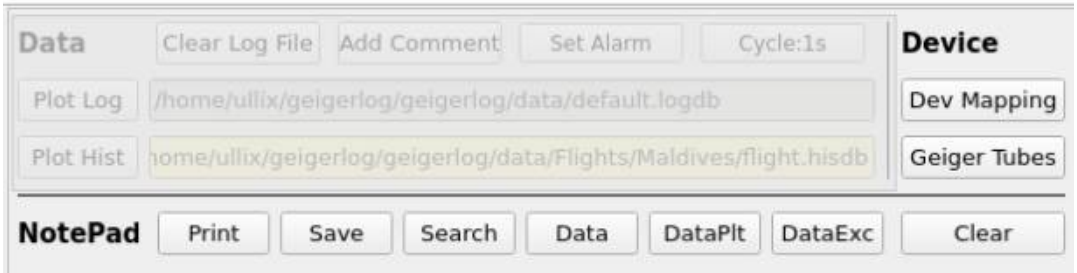


Figure 9: Device Dashboard

Show the Device Mapping and edit the Geiger Tube settings.

Device Mappings

A button to print the Device Mappings into the NotePad; see chapter Device Mappings on page 20.

Geiger Tubes

Clicking the **Geiger Tubes** button opens a dialog box allowing you to view and modify the sensitivities of the Geiger tubes as currently configured. Sensitivities are an important property of your data and impact plotting of graphs. More on this topic in chapter Appendix G – Calibration on page 164.

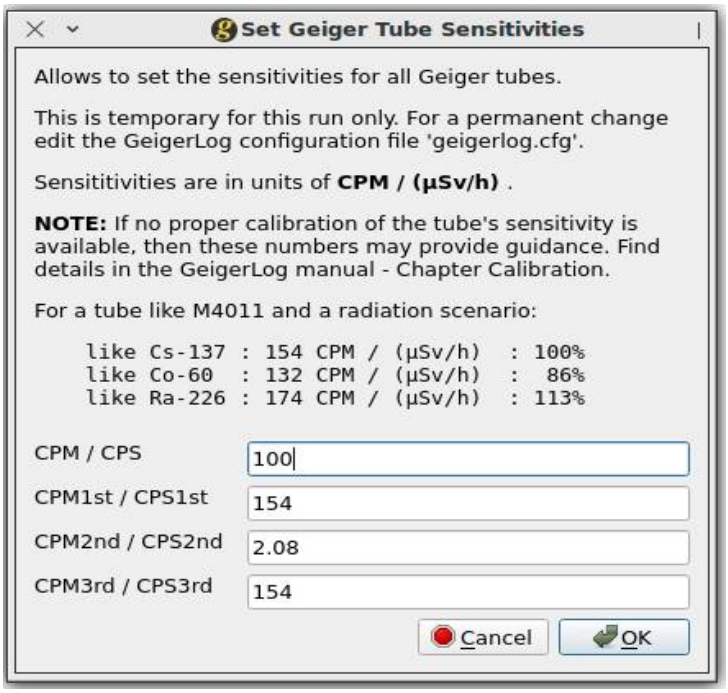


Figure 10: Set Geiger Tube Sensitivities

NotePad Dashboard – Monitor Your Recordings

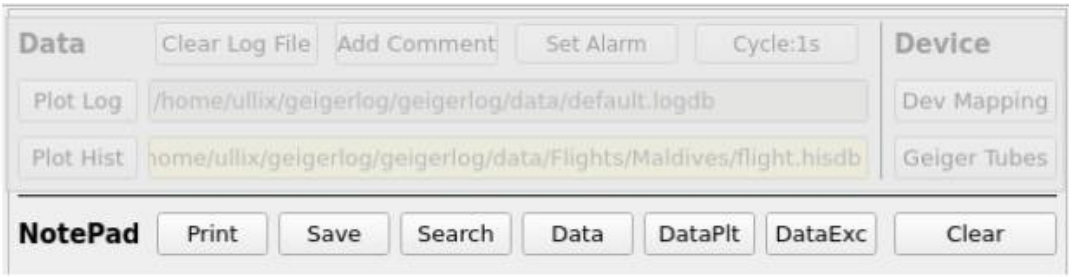


Figure 11: NotePad Dashboard

The buttons under **NotePad** provide convenience functions often used with GeigerLog.

Clear

The **Clear** button clears all content from the NotePad. It may be the most often used button ;-).

Data Exc, DataPlt, and Data

All buttons print data from the **currently active file** – Log file or History file – to the NotePad.

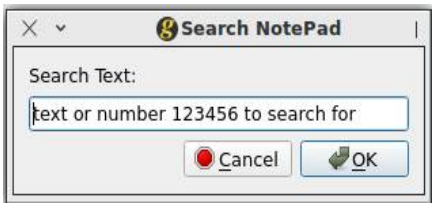
The **Data Excerpt** button prints an excerpt of the data. If the file is short, the whole file is printed, if long, then printing is limited to only a few first and last lines. This may be the second-most-often used button ;-).

The **Data Plt** button prints the data from all data points currently shown in the graph. Helpful in particular to identify individual values.

The **Data** button prints all the data in the file. This button has a second function: when your file is really large, the printing may run for a long time. This button changes into a **STOP** button, allowing you to stop any such long printouts.

Search

To search the entire NotePad for the occurrence of a text or a number. The button opens a little dialogue box up where you can enter the text to search for. It will search backwards. If it started in the middle of the NotePad and did not find the text, it will restart again from the end.

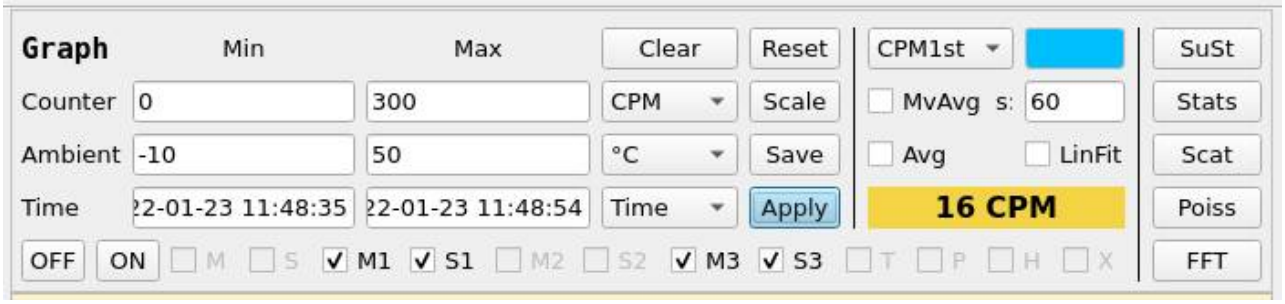


Also conveniently accessible by pressing **CTRL-F** as is the search function in most editors.

Print and Save

The **Print** button allows to print the NotePad content to paper or to a pdf file. The **Save** button will save the current content of the NotePad to a text file. It will be saved to the data directory and will be named `<filename-of-active file>.notes`.

Graph Dashboard – Visualize Your Recordings



Drawing 1: Graph Dashboard

The Graph Dashboard controls what is displayed on the graph and how it is displayed. And no matter what you do here, the logging, downloading, processing or saving of the data will never be impacted!

The graph is laid out as Time Course of your data, i.e. the horizontal X-axis shows the time, and the variables are plotted versus this time on either one of two vertical Y-axis:

- the **left Y-axis** is labeled **Counter** and is used for all Geiger counter data
- the **right Y-axis** is labeled **Ambient** and is used for environmental data temperature, barometric-pressure, humidity, and as Xtra anything else, like CO_2, air-quality, light intensity, and other.

What is displayed?

With up to 12 variables now available for display, it will often be important to reduce the number of variables displayed. The bottom row has buttons and checkboxes which allow to show or hide a variable. Depending on the active data file – be it log or history – not all variables may be available. The checkboxes of unavailable variables are grayed out and cannot be selected.

The buttons OFF, ON switch all variables OFF, or ON, resp., (unavailable variables remain OFF and unselectable). The checkboxes use shortened names for the variables to ease the overview:

M	= CPM	from any Geiger counter device
S	= CPS	from any Geiger counter device
M1	= CPM1st tube	from any Geiger counter device
S1	= CPS1st tube	from any Geiger counter device
M2	= CPM2nd tube	from any Geiger counter device
S2	= CPS2nd tube	from any Geiger counter device
M3	= CPM3rd tube	from any Geiger counter device
S3	= CPS3rd tube	from any Geiger counter device
T	= Temperature	from any device yielding ambient data
P	= Pressure	from any device yielding ambient data
H	= Humidity	from any device yielding ambient data
X	= Xtra	from any device yielding ambient data (e.g. CO2, light, Air-quality ...)

Min/Max, Apply, Clear, Reset

The graph is auto-scaled in all 3 axis so that all data fit into the graph. However, the Min and/or Max value of the X-axis and both Y-axis can be set manually.

The Min/Max values for Counter and Ambient need to be entered from the keyboard. Those for the Time can also be entered manually as e.g. '2018-07-18 14:00:41'. However, it is easier to use a mouse: with the mouse pointer resting within the graph area, do a mouse-left-click to enter the Min Time value, and a mouse-right-click to enter the Max Time value.

To apply your entries to the graph, either click the **Apply** button or hit the **Enter** key.

To clear all entries in all Min/Max boxes, click the **Clear** button.

To reset all settings in the complete Graph Dashboard to their defaults, click the **Reset** button.

Scaling

Allows to change the Scaling during a run. Generally all data are displayed as recorded, but they can be scaled for display, while still being saved unmodified. See chapter ValueScaling and GraphScaling on page 37.

One example is barometric-pressure, which is conveniently displayed as 'pressure minus 1000'. Since barometric-pressure is typically within the range of 970 … 1030 hPa, this transformation allows it to be displayed at the same scale as the other environmental variables. The values saved to the log file remain the unmodified, original values.

Another example applies when showing data with an extreme dynamic range, like light-intensity. The data can be compressed e.g. by taking the logarithm, like: log(VAR).

NOTE: You can also use the options accessible from the Graph Toolbar. Click the icon labeled 'Edit axis, curve and image parameters' and select e.g. Scale Log .

You can do more modifications via this icon, e.g. line color, line width, symbols, and more.

Save

Clicking the **Save** button will save the current graph to a file in the data directory in the "png" format. The filename is the database file name extended with a DateTime stamp, like: ".../geigerlog/data/default.logdb2022-01-13 12:27:17.png"; it will be printed into NotePad.

NOTE: Other file saving formats are accessible from the Graph Toolbar. Click the icon labeled 'Save the Figure' and select location for saving and the graph format.

Units

X-axis: The time axis can display Time-of-Day or time-since-first-record. For the latter, set the unit selector to auto for an automatic choice between day, hour, minute, second, or set the time unit manually.

Left-Y-axis: This counter axis can either show CPM/CPS or µSv/h. If both a CPM and a CPS variable are shown at the same time, their 60 fold difference may make the graph less informative. Deselecting one may be preferred.

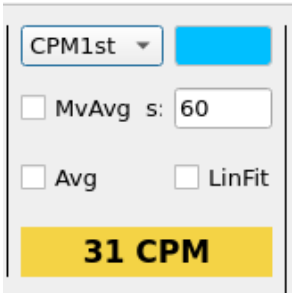
However, when µSv/h is used, the two CPM and CPS based curves will overlap!

They should also overlap if multiple counter with perhaps different tubes, or one single counter with dual tubes, as the GMC 500+ device, are used to measure a single source, but this depends on the proper use of the sensitivity values for each device and tube. This had not always been the case, see discussion in chapter Appendix G – Calibration on page 164.

Right-Y-axis: This Ambient axis is used for all environmental data temperature, barometric-pressure, humidity, and as Xtra anything else, like CO_2, air-quality, light intensity, and other.

The temperature can be displayed in units of either °C or °F.

Selected Variable



The **Selected Variable** segment is a subset of the Graph Dashboard, shown in the picture to the right.

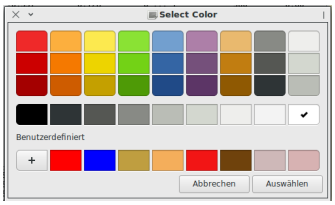
Selected Variable

In this picture **CPM1st** is the **Selected Variable**. By clicking on the drop-down box you can select one of the variables displayed, which thereby becomes the **Selected Variable.** Only variables being currently displayed in the graph can be selected, the others are grayed out in the drop-down box and cannot be selected!

This **Selected Variable** will be highlighted in the graph with a brighter color and a thicker line, while the other variables will be dimmed.

The **Selected Variable** is the variable which some of the **Tools** (see below) for analysis and quality control will use in their analysis.

Color Box



This shows the color, in which the **Selected Variable** is drawn in the graph, currently a light blue. By clicking on the Color Box, a dialogue comes up, allowing you to change the color for this variable.

Clicking the Reset button, or reloading the database, or restarting GeigerLog sets the color back to the default color.

MvAvg

If the checkbox **MvAvg** is checked then a **Moving Average** [13]) as a yellow framed line in the color of the Selected Variable will be plotted. The duration of the averaging period is taken from the entry field next to the checkbox. The default duration is 60 sec.

13 The Moving Average, sometimes also called a Rolling Average is calculated and plotted by taking N data points, calculating their arithmetic average, and plotting the result at the time point in the middle of the range. Hence, N/2 data points at both the beginning and the end of the record will not be available in the Moving Average line.

With CPS data being recorded once per second, applying a MvAvg of 60 sec to them will basically make a CPM curve out of it. For longer recording times moving averages over 600 or even 6000 may be appropriate.

Avg

If the checkbox **Avg** is checked then a yellow framed horizontal line in the color of the selected variable will be drawn at the average value of all plotted data of the **Selected Variable**.

If the **Selected Variable** is of the counter type, and these Poisson distributed data can be approximated by a Normal Distribution, two horizontal dashed lines will be drawn indicating the theoretical 95% range for the plotted data set, i.e. 95% of all data fall into this range, and 5% will be outside. If GeigerLog determines that the condition of Normal Distribution is **not** met, then **no** 95% range lines will be drawn, which is typically the case when the average is < 10 [14]).

LinFit

If the checkbox LinFit is checked then GeigerLog will calculate a **Linear Regression** to the data of the **Selected Variable,** and a straight line representing the regression will be drawn as a yellow framed line in the color of the **Selected Variable**.

If this line is moving up and down, then your data are still fluctuating too much, and you need to collect longer. Once the line remains more or less horizontal (being nearly the same as the average), you may have collected long enough. The 'Delta Plot' at the label gives you the value difference from right to left end of the LinFit line.

Last Value Box

During logging the last value of the **Selected Variable** will be shown in the **Last Value Box** as black letters on a golden colored background, in the picture showing **31 CPM.**

Display Last Values

Clicking on the **Last Value Box** opens the **Display Last Values** window showing the last values of all mapped variables, their device source, and the value they have after Graph-scaling was applied. This window is auto-updated during logging. When logging stops, the window remains open, but the values remain frozen and are shown on a gray background.

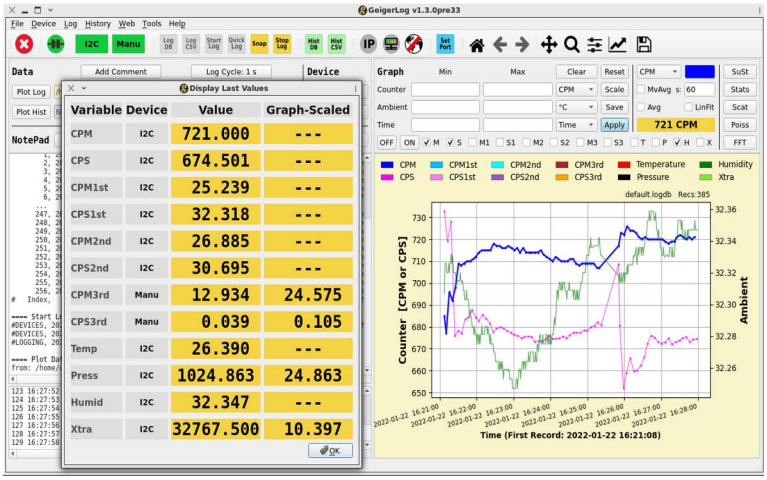
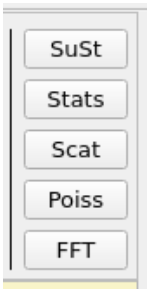


Figure 12: The Display Last Values Pop-up Window

14 For a more detailed discussion of Normal and Poisson Distributions of Geiger data see my "Potty Training for Your Geiger Counter" article on SourceForge https://SourceForge.net/projects/geigerlog .

Tools



The **SuSt**, **Stats**, **Scat, Poiss,** and **FFT** buttons are tools for the Quality Control of your data.

The **SuSt** button prints a Summary Statistics to the NotePad of all variables currently displayed, the **Stats** button opens a window with more elaborate statistics.

The **Scat** button allows to make a scatterplot of one variable against another one of the configured variables. You can also apply polynomial regressions up to order 7.

The **Poiss** button allows a Poisson test of the data; the **FFT** button does a Fast Fourier Transformation. More details in the chapter Quality Control of your Data on page 49.

The **Poiss** and **FFT** buttons act on the **Selected Variable** only and present their info in a pop-up-window.

Web Enablement

GeigerLog's Web Enablement is multifold:

- **WiFi Connections** – In addition to cable connections (USB, Audio, …) to external devices, GeigerLog can make the connection via WiFi and can act both as **Client** and as **Server**. Such WiFi-enabled devices are separately discussed in chapter WiFi Devices on page 96.
- **Remote Monitoring** – GeigerLog supports monitoring itself from remote devices, like a Smartphone or any other computer, by offering a set of web pages. Some can be configured to fit the user's need.
- **Widgets** – It also offers Widgets, which a user can embed into his own web pages.
- **Radiation World Map Updating** – GeigerLog supports updating the "Radiation World Map" gmcmap.com.

The Web Enablement requires that the computer running GeigerLog is connected to the local network, typically with WiFi access, and that you know its IP address.

Access the Web Functions

The functions can be accessed by the menu **Web** or by the toolbar icons as shown on the right. The icons represent IP Address, Remote Monitoring, Radiation World Map. The functions are inactive if their icon has a red, diagonal strike-through, and active otherwise.

IP Address

If you don't already know the IP Address of your computer running GeigerLog, then either use menu **Web → Show IP Status** or click the **IP** icon in the toolbar. The pop-up dialogue tells you – in this case – it is '10.0.0.20'. It is a property of the network, and cannot be changed by GeigerLog.

It also provides info on port use of Monitor Server and GeigerLog's WiFiClient server.

Remote Monitoring – Using the Default Server

Remote Monitoring requires to set up the Monitor Server. It is configured in GeigerLog's configuration file. By default it is set to **autostart** once GeigerLog has started.

Looking at Monitor Server Pages

In the default setting enter into the address bar of your browser:

- Regular (non-secure) server: http://IP:Port
  example: http://10.0.0.20:8081/
  or simply: 10.0.0.20:8081/

Remote Monitoring – Using an SSL Secured Server

The default is to run Monitor Server as regular, non-SSL-secured server. But your network may require otherwise. To run the monitor as an SSL secured server activate it by setting: "MonServerSSL = yes" in the GeigerLog configuration file.

Using SSL requires to provide **cryptographic keys**, stored in a PEM [15]) file. To create a PEM file use program "openssl".

- On Linux, Mac, ... : openssl is most likely installed by default; otherwise install it
- On Windows: see instruction to install openssl on Windows [16])

Once openssl is installed, issue command:

```
openssl req -x509 -newkey rsa:4096 -keyout geigerlog.pem -out geigerlog.pem -days 365 -nodes
```

and copy resulting file 'geigerlog.pem' into GeigerLog sub-directory "gweb".

Looking at Monitor Server Pages using an SSL Secured Server

In the SSL setting enter into the address bar of your browser (Note the "s" at http):

- Secure (SSL secured) server: https://IP:Port
  example: https://10.0.0.20:8081/
  Depending on browser configuration an equivalent to the default's "simple" version may or may not work!

Remote Monitoring – UTC Correction

Browsers like Firefox and Chrome have the unfortunate property of converting all Javascript time formats as if they were UTC times. Other browsers may respond differently.

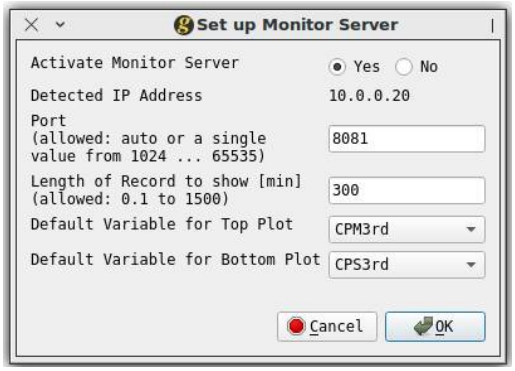
GeigerLog uses an automatic correction when set to "MonServerUTCcorr = auto" in the configuration file, but if this fails enter the proper correction. See there for details.

Remote Monitoring – Other Settings

To **autostart** the Monitor Server at every start of GeigerLog, set "MonServerAutostart = yes" in the GeigerLog configuration file.

The **Port Number** for the Monitor Server must be set by the GeigerLog user. GeigerLog uses the first available port number out of a list. The default list is 8080, 8081, 8082, 8083, 8085, 8086, 8087, 8088, 8089, but this can be reconfigured by the user.

Any ports in the range 1024 … 65535 can be given. Any lower ports, in particular the standard port 80, can NOT be used as this would require administrative privileges and could become a security issue.



15 https://www.howtogeek.com/devops/what-is-a-pem-file-and-how-do-you-use-it/

16 https://thesecmaster.com/procedure-to-install-openssl-on-the-windows-platform/

When **Yes** is checked at **Activate Monitor Server**, and OK clicked, the Monitor Server becomes active and can be accessed with any browser by simply entering <IP Address>:<Port Number> – in the case shown it would be: 10.0.0.20:8081 – and you will see the Welcome screen (1st Smartphone picture in Figure 13). Tapping the other menu items shows the other pages.

The Plot page (3rd in Figure 13) can be configured with the other entries in the Monitor Server Setup: length of the data plot in minutes, and which variables to show as top and bottom graphs. All this is pre-configured in the configuration file, but can be changed during a run from within Geiger-Log. To see the original GeigerLog graph, tap the Get GL Graph button on the Plot page (6th in Figure 13).

Figure 13: Monitor Web Pages delivered by GeigerLog

The speedometer like gauge on the 2nd from left picture shows a green-yellow-red scale, which one is trained to interpret as a ranking from safe to dangerous. But what does it actually define, which level is safe, and which dangerous? Find a discussion of the settings I have chosen in chapter On what grounds do we set the radiation safety levels? on page 140.

Widgets – Embed them in Your Webpage

Once GeigerLog Monitor Server is running, you can call up a Widget Demo by entering into your browser (remember to use “https://” when you have activated SSL!):

http://<IP Address>:<Port Number>/widget_demo

in the case just discussed it would be: http://10.0.0.20:8081/widget_demo

Viewing the source code of the web page will quickly tell you how to embed these iframe-based widgets into your own web page.

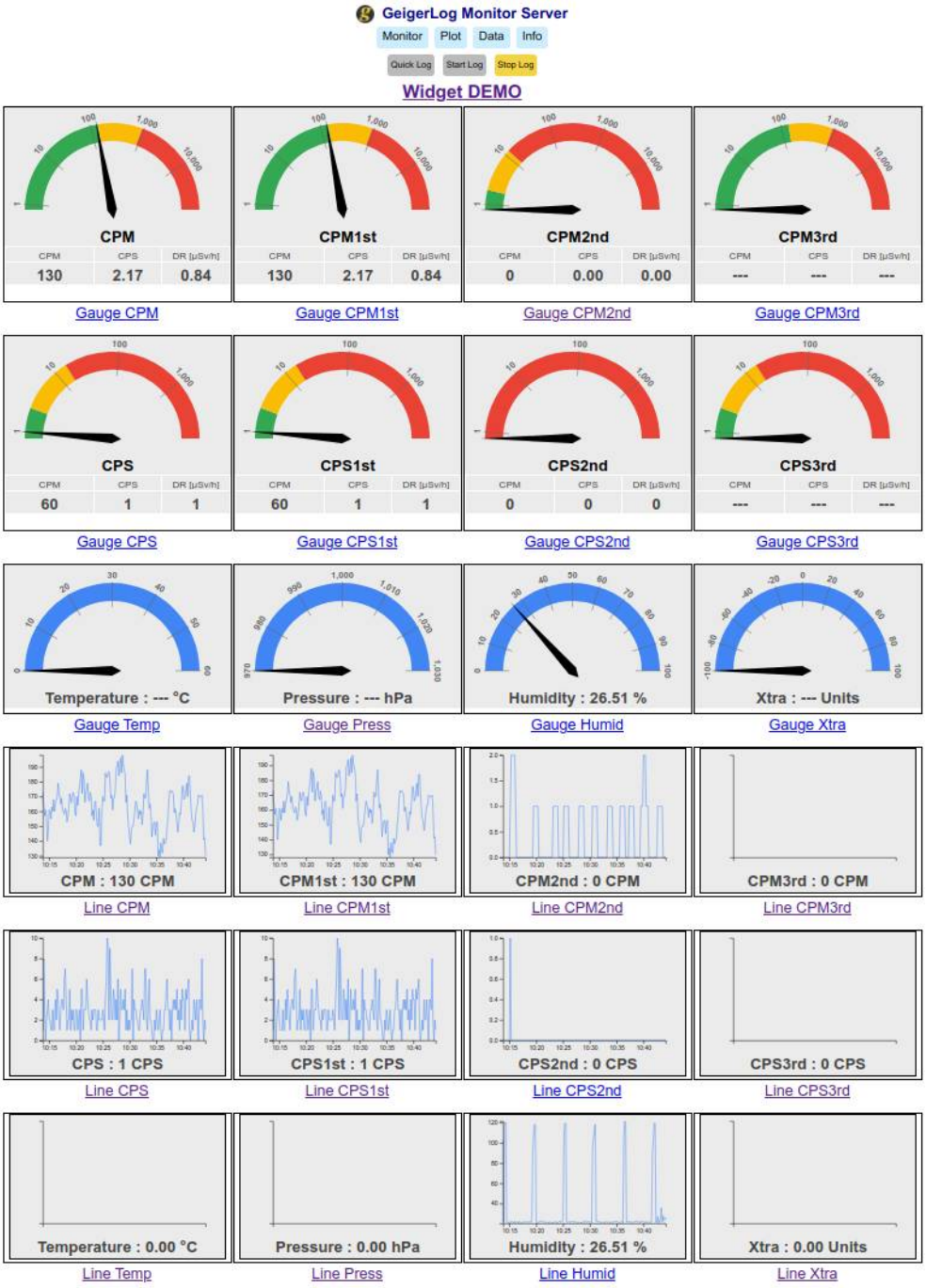


Figure 14: The standard widgets offered by GeigerLog

Radiation World Map

GQ's gmcmap.com site is the only currently supported site. But note that there are some caveats to consider when you use this site, please see Radiation World Maps on page 138!

To use this feature, GeigerLog needs access not only to the local network, but also to the internet.

If you want to contribute to gmcmap.com, you need to register there. This provides you with a UserID and a CounterID. Enter both in the GeigerLog configuration file `geigerlog.cfg` under the heading `[Worldmaps]` into the fields `GMCmapUserID` and `GMCmapCounterID`, resp..

The World Map icon in the toolbar allows to configure the updates. Upon starting GeigerLog the icon is inactive, marked by a red-strike-through. This prevents unwanted updates. Click the icon and a dialog allows you to set up the desired properties of the updater.

Check **Yes** at **Activate Updates** to activate the WorldMap (after the OK button is clicked). Enter the **Update Cycle Time** in minutes. The default is set to 1 hour, but even 12 h or 24 h do make sense. Governmental sites typically do only DAILY averages of quality controlled data [17])! The values sent to the website by GeigerLog will be averages over exactly this Update Cycle period.

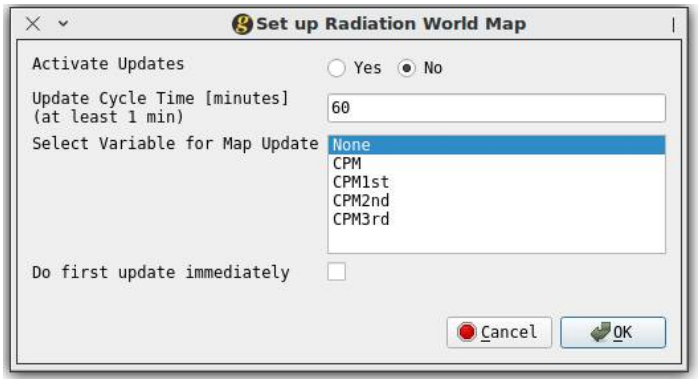


Figure 15: Set up Radiation World Map

Select the **Variable** you want to use for the update. Only CPM* variables can be selected, no CPS* or Ambient variables!

If you check to **Do first update immediately** (not recommended), the first Radiation World Map Update will be done right after clicking OK, provided that logging is ongoing. If logging is not ongoing, then GeigerLog will wait until it has started, and then do an update. This could result in an update made on a single record only! This is normally undesirable, see Radiation World Maps on page 138! However, it helps getting an immediate feedback on whether the system works correctly. I recommend to have GeigerLog log for a little while at least, and only then activate the Radiation World Map updater!

When GeigerLog does an update, it prints a brief message to the NotePad, like:

```
==== Update Radiation World Map ============================================
Sending Data   : 2021-11-10 13:37:08  CPM:109.1  ACPM:109.1  uSV:0.71
Server Response: OK
```

To stop updating click the icon and check **No** for active updates.

17 See chapter Radiation World Maps on page 138

ValueScaling and GraphScaling

GeigerLog uses a powerful **Formula Interpreter** to allow scaling of any variable, i.e. modifying the value of a variable by a formula.

There are two types of scaling : **ValueScaling** and **GraphScaling**.

ValueScaling: the measured, original value is modified and this modified value is saved. This is adequate if you know that your instrument is off by a certain amount and you correct this to save the proper value only.

ValueScaling does **not** work on any data loaded from file!

GraphScaling: The original value is saved unmodified, and the modified value is used for plotting, and only for plotting! The barometric-pressure example below is a typical application and implemented by default in the configuration file.

The GraphScaling also works on any data loaded from file as soon as they are plotted.

View and Edit Current Scaling

To view the current setting of your scaling, click menu **Tools → Scaling Dialog ...** or click the **Scaling** button in the **Graph Dashboard** to see a dialog as in Figure 16.

You can edit all entries. They become active immediately after you click OK, but remain in effect **only for the current run**. They are NOT saved to the configuration file!

If you want a permanent change, then you'd have to make the edits in the GeigerLog configuration file.

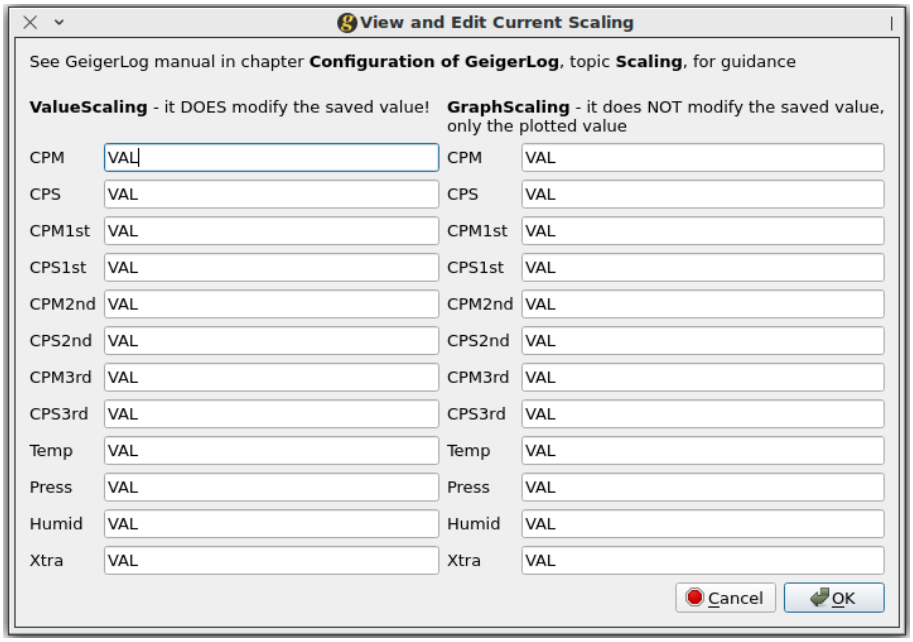


Figure 16: View and Edit Current Scaling

Formula Interpreter

Enter any valid formula, and GeigerLog will calculate the result.

Apart from basic math (+ - * /) the formula may include math functions. Any parameter to a function could be a fixed value like 123.45, or could itself be another formula. The default entry is VAL, which means no modification of the value.

Capitalization has no impact – log(val) is the same as LOG(VAL) or Log(Val) or lOG(vAl).

Valid formula examples:

- val – 1000
- SQRT(val) * 5 + 100
- EULER() ** 10 - EXP(10) # result will be close to zero
- ABS((LOG(val) + 1000) / sin(3.14 * val)) + 10.123
- MEMORYUSED()
- PARA_CORR(val, 120)

Basic functions:

+ - * /	:	basic math
**	:	raise to the power, e.g. 2**8 (=256)
VAL	:	the measured value
PI()	:	value of number Pi = 3.141592653589793
EULER()	:	value of Euler's number e = 2.718281828459045

Math functions:

EXP(x)	:	exponential
LN(x)	:	log to base e; aka natural log
LG(x)	:	log to base 10
LG2(x)	:	log to base 2
SIN(x)	:	sine
COS(x)	:	cosine
TAN(x)	:	tangent
SQRT(x)	:	square root
CBRT(x)	:	cube root
ABS(x)	:	absolute value
INT(x)	:	integer value INT(3.14) => 3

Technical functions

LOADAVG(x)	:	load average % for x=0, 1, 2; meaning average over 1, 5, 15 min
MEMORYTOTAL()	:	total memory of computer in MByte
MEMORYPERCENT()	:	percent memory of computer used by GeigerLog
MEMORYUSED()	:	memory in MByte used by GeigerLog
CPUPERCENT()	:	percent CPU usage
DURGRAPH("GRAPH")	:	duration of updating the graph in ms
DURGRAPH("MEM")	:	duration of updating the memory in ms
DBSIZE()	:	size of database in MB
NTPOFFSET()	:	the NTP (Network Time Protocol) Time server offset in ms to current computer time

Special functions:

COUNTER(x)	:	count of current records multiplied with x
LOGCYCLE()	:	length of LogCycle in sec
DELTACYCLETIME()	:	time difference to last log cycle in sec
MAKECPM(x)	:	make a CPM variable from CPS variable x
SUMVV(x, y)	:	Sum two variables into one; x=<name of variable1>, y=<name of variable2>

RadPro-specific functions

RADPROCPM()	:	uses the RadPro function 'GET tubeRate' to get a "RadPro CPM" value. **Not** recommended!
RADPROHISTLEN()	:	downloads the RadPro History and returns the len of Hist. Requires a GeigerLog LogCycle of no faster than 4 sec!
RADPROCLOCKDRIFT()	:	get the RadPro clock difference to the computer in full sec

Synthetic Data functions

POISSON(x)	:	random values from a Poisson distribution with x=mean CPS
NORMAL(x)	:	random values from a Normal distribution with x=mean CPS and StdDev=SQRT(x)
EXPONENTIAL(x)	:	random values from an Exponential distribution with x=mean CPS, returning time gap between pulses in µs.

Deadtime dependent functions

PARA_POISSON(x, y)	:	random values from a **Para**lyzing-**POISSON** distribution with x=mean, y=deadtime in µs. This is like data coming from a **Geiger tube** [18]).
NOPA_POISSON(x, y)	:	random values from a **No**n-**Para**lyzing-**POISSON** distribution with x=mean, y=deadtime in µs. Whatever detectors work like this, they do **NOT** include a Geiger tube!
PARA_CORR(x, y)	:	**Para**lyzing-Deadtime-**Corr**ection; x=value, y=deadtime in µs Correcting count rates coming from a **Geiger tube**
NOPA_CORR(x, y)	:	**No**n-**Pa**ralyzing Deadtime **Corr**ection; x=value, y=deadtime in µs Correcting count rates from devices, **excluding a Geiger tube**

Examples:

1. The temperature may need adjustments, which can be as simple as:

   T_new = T_old – 0.23

   Enter Formula:
   Temp = VAL – 0.23

2. Variables with a huge dynamic range – like light intensity – do not allow to see small values in the plot in the presence of very large values. Square root, cube root, log functions allow to compress large values. However, sometimes log may pose a problem as it cannot plot a zero value.

   X_new = SQRT (X_old) (square root)

18 https://en.wikipedia.org/wiki/Dead_time

X_new = CBRT (X_old) (cube root)
X_new = LOG (X_old) (log to base 10 – CANNOT plot zero values!)

Enter Formulas, e.g. for variable Xtra:
Xtra = SQRT(VAL)
Xtra = CBRT(VAL)
Xtra = LOG (VAL)

3. Pressure data are typically within 970 … 1030 hPa. To plot them on the same scale as other environmental parameters – like Temperature (-10 … 40 °C), Humidity (20 … 90%) – the pressure could be plotted as:

   p_new = p_old - 1000

   Enter Formula:
   Press = VAL – 1000

4. The barometric-pressure is measured at the altitude of the location of your device, but for weather stations you typically want it reduced to sea-level altitude. This formula is simple but limited to altitudes of under 50 m [19]). (sea = sea-level, alt = altitude [m], Tv = "*the mean annual normal value of virtual temperature at the station in kelvins*", chosen as 293 K:

   P_sea = P_alt * (1 + alt / 29.27 * Tv)

   For an altitude of 49 m and Tv = 293 K, enter Formula:
   Press = VAL + VAL * 49 / (29.27 * 293)

5. The barometric-pressure is measured at the altitude of the location of your device, but for weather stations you typically want it reduced to sea-level altitude. This formula is valid at any altitude, but needs T [20]): (sea = sea-level, alt = altitude [m], T = temperature [°C], T chosen as 20°C):

   P_sea = P_alt * (1 - (0.0065 * alt) / (T + 0.0065 * alt + 273.15))(-5.257)**

   For an altitude of 85 m and a temperature of 20 °C, enter Formula:
   Press = VAL*(1-0.0065*85)/(20+0.0065 * 85 + 273.15))**(-5.257)

6. If you want CPM, but your device is delivering only CPS, e.g. into variable "CPS2nd", you can get a CPM-value in variable "CPM2nd" with this formula:
   CPM2nd = MAKECPM("CPS2nd")

7. You have two devices with equal tubes and you want the sum of the counts of the two. The first device records on "CPM1st", the second one on "CPM2nd". You get the sum of the two in "CPM3rd" with this formula:
   CPM3rd = SUMVV("CPM1st", "CPM2nd")

8. To get a monotonically increasing value e.g. into variable Temp with a step-size of 3 , enter the formula:
   Temp = COUNTER(3)

---

19 "Pressure Reduction Formula", WORLD METEOROLOGICAL ORGANIZATION, CIMO/ET-Stand-1/Doc. 10, (20.November.2012), chapter 3.11.2 Low-level stations.
Quote: *"At low-level stations (namely, those at a height of less than 50 m above mean sea level), pressure readings should be reduced to mean sea level by adding to the station pressure a reduction constant C given by the following expression: C = p . Hp /29.27 Tv (3.3). Hp is the station elevation in meters; and Tv is the mean annual normal value of virtual temperature at the station in kelvins."*

20 https://keisan.casio.com/exec/system/1224575267

9. To see how demanding GeigerLog is with respect to the resources of your computer, enter the formulas:
   Temp = LOADAVG(1)
   Press = MEMORYUSED()
   Humid = CPUPERCENT()
   Xtra = DURGRAPH()

Synthetic Data functions

Three synthetic data functions are available from: Poisson, Normal, and Exponential distributions.

Poisson (mean): This is the distribution describing radioactivity, and is what a Geiger counter should deliver. It has a single parameter (mean), and has StdDev = SQRT(mean)

Normal(mean): Visually similar to Poisson, but fundamentally different. A Normal fucntion actually has a second parameter StdDev, which here is automatically set to SQRT(mean) to mimic Poisson

Exponential(mean): It is a distribution of the time gaps between Poisson pulses. This allows to build a Poisson distribution from it.

As an example, define each in the GeigerLog configuration file, and start logging:

```
…
FormulaActivation = yes
FormulaVariables = CPM, CPS, CPM1st
…
[ValueFormula]
…
CPM = Poisson(6.66)
CPS = Normal(6.66)
CPM1st = Exponential(6.66)
…
```

Results are shown in Figure 17. At a mean of 6.66 the Normal data clearly do not fit the expected Poissonian shape. Exponential data show than many count events come at the left edge, i.e. in close proximity to the previous event!

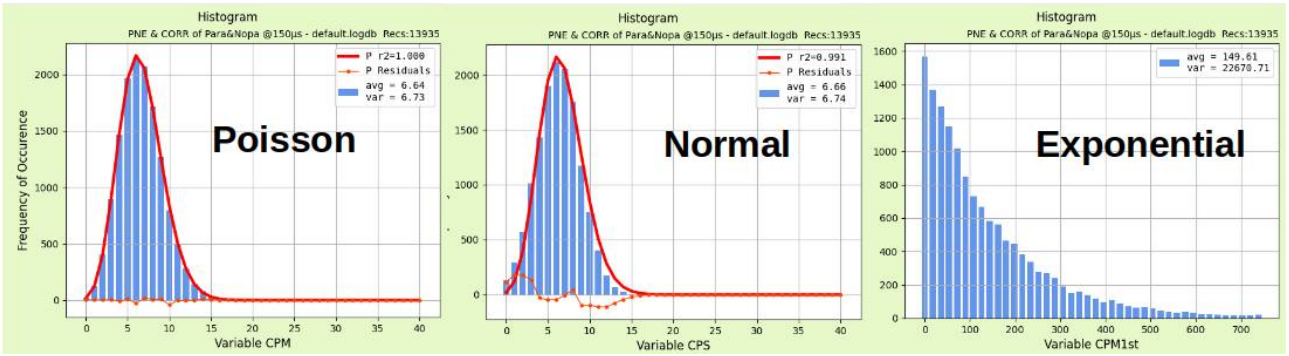


Figure 17: Using the Formula Device with Poisson(6.66), Normal(6.66), and Exponential(6.66).

Deadtime dependent functions

A radioactive particle hitting a Geiger counter basically creates a shortcut in form of a flash inside the Geiger tube. This flash lasts a moment – e.g. 150 µs (as measured for the M4011 tube) – and an electronic pulse will be registered ideally at the beginning of this flash, and forwarded to the electronic as a "count".

Obviously, a second particle arriving during this busy period, aptly called deadtime, of the flash does not trigger another count. But unfortunately it will result in elongating the shortcut flash period! Thus the deadtime for the initial pulse seems to increase!

This phenomenon is called "paralytic behavior", as it paralyzes the tube, rendering it incapable of detecting another particle event. Accordingly, functions in GeigerLog's formula interpreter are distinguished with the prefix "**PARA_**" for Paralytic, and "**NOPA_**" for Non-paralytic behavior. Geiger tubes exhibit **PARA_** behavior!

The **NOPA_POISSON(x, y)** is also a synthetic data function, drawing random values from a **POISSON** distribution with x=mean just as the default POISSON(x) function, but with the extra functionality that an event is **not** counted if it occurs within the deadtime period y (in µs) beginning with the start of the initial event.

The **PARA_POISSON(x, y)** is the same as NOPA_POISSON(x, y), except that the 2nd event also seemingly extends the deadtime of the 1st event. So, at very high data rates the events come so closely spaced, that the initial event will never end, and hence the observed count rate will be zero, thus it **paralyzes** the counter!

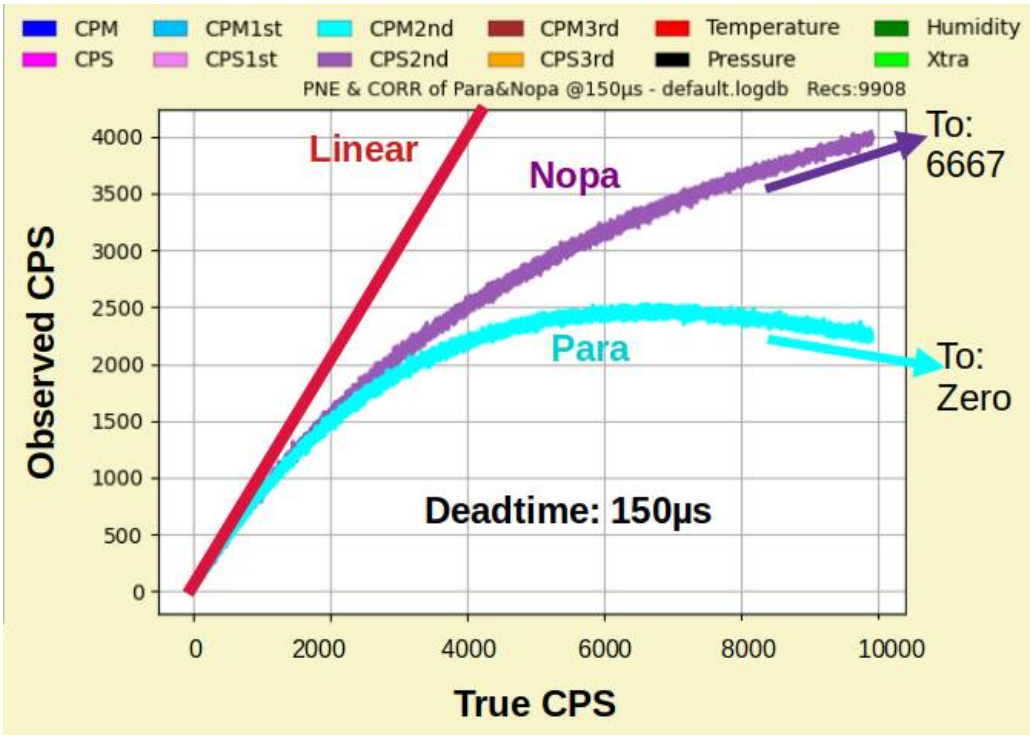


Figure 18: A run with Deadtime 150µs in the PARA and NOPA case

A simulation result with PARA and NOPA behavior and a deadtime of 150µs (as measured for M4011) is shown in Figure 18. One might naively think that the max count rate is given by the inverse of the deadtime, so CPS=6667, but far from that: at CPS=2000 the observed data deviate by

over 30%! A Geiger counter (Para case) with this tube will reach an observed maximum of CPS=2500 when the true CPS=6000; a 240% difference!

Count Rate Corrections

Now that we know what the distortion of count rates is, an obvious question comes up: to what degree can this be corrected mathematically?

In the **NOPA** case this is actually rather simple. The formula is:

CPS_true = CPS_obs / (1 - CPS_obs * deadtime[s])

For any CPS counts being recorded in variable CPS, coming from a device with a deadtime of 150 µs, you would enter the formula (but remember, this device is **NOT** a Geiger counter!):

CPS = NOPA_CORR(val, 150)

In the **Para** case, as for a Geiger counter, this is a lot more difficult. The governing formula is:

CPS_obs = CPS_true * exp(- CPS_true * deadtime)

The crux is that now we have CPS_true both inside and outside the exp() function and thus this formula cannot be analytically solved for CPS_true! The only help comes from an iterative approach. This is implemented with this formula:

CPS = PARA_CORR(val, 150)

The calculations involved are a lot more demanding for Para than for Nopa, and while this is no problem for GeigerLog on today's laptops and desktops, it is a problem for microprocessors as used e.g. in any of the Geiger counters!

Some of the latest GMC counter do have a deadtime correction, however, this is a Nopa correction, which is **the wrong one for a Geiger counter**!

These Para and Nopa corrections can of course also be applied to the simulated data in Figure 18 and the outcome is shown in Figure 19.

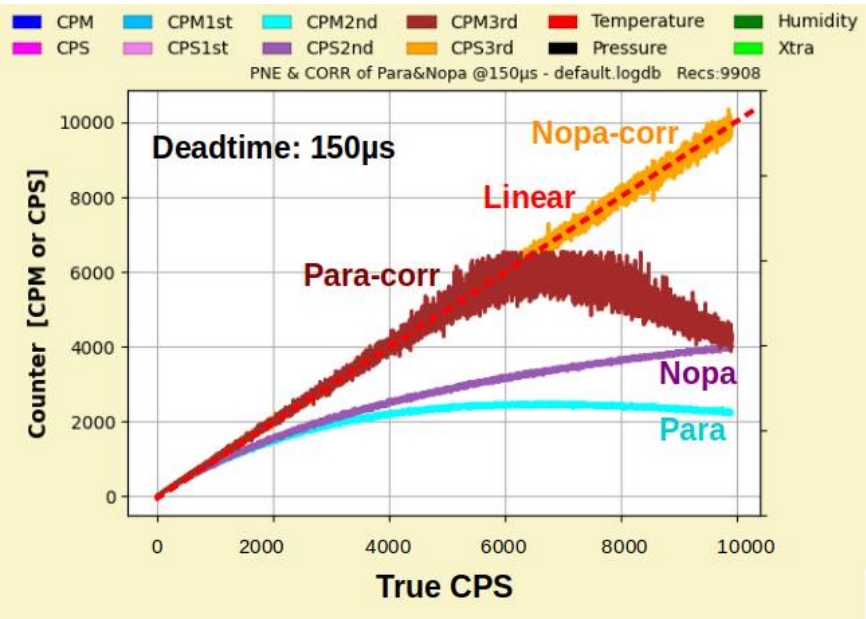


Figure 19: Correcting count rates

In this theoretical, simulated case the Nopa correction is near perfect. The Para correction fails when the maximum True CPS=~6000 has been reached, because here Para turns downward, and the iterative solution becomes ambiguous, because now two values are possible.

Handling CSV Data-Files

Loading Data from CSV formatted Files

GeigerLog stores its data in SQL-database files. Early versions of GeigerLog had used CSV (**C**omma **S**eparated **V**ariables) text files. Log files had the extension '.log', and History files the extension '.his'. Also, other programs may produce CSV files, which you might want to load into GeigerLog.

This is easily done with functions in the menu: **Log → Get Log from CSV File**, and **History → Get History from CSV File**. You will be offered to load an existing *.log, *.his, *.CSV, *.txt, or *.notes file, which will then be presented in the dialogue. An example is in Figure 20.

Either command creates the exact same database file, except that you cannot add new data to a History file, while you can for a Log file. Such appending to a Log file will most of the time not make much sense, but you do have the option. Generally a History file is the better choice.

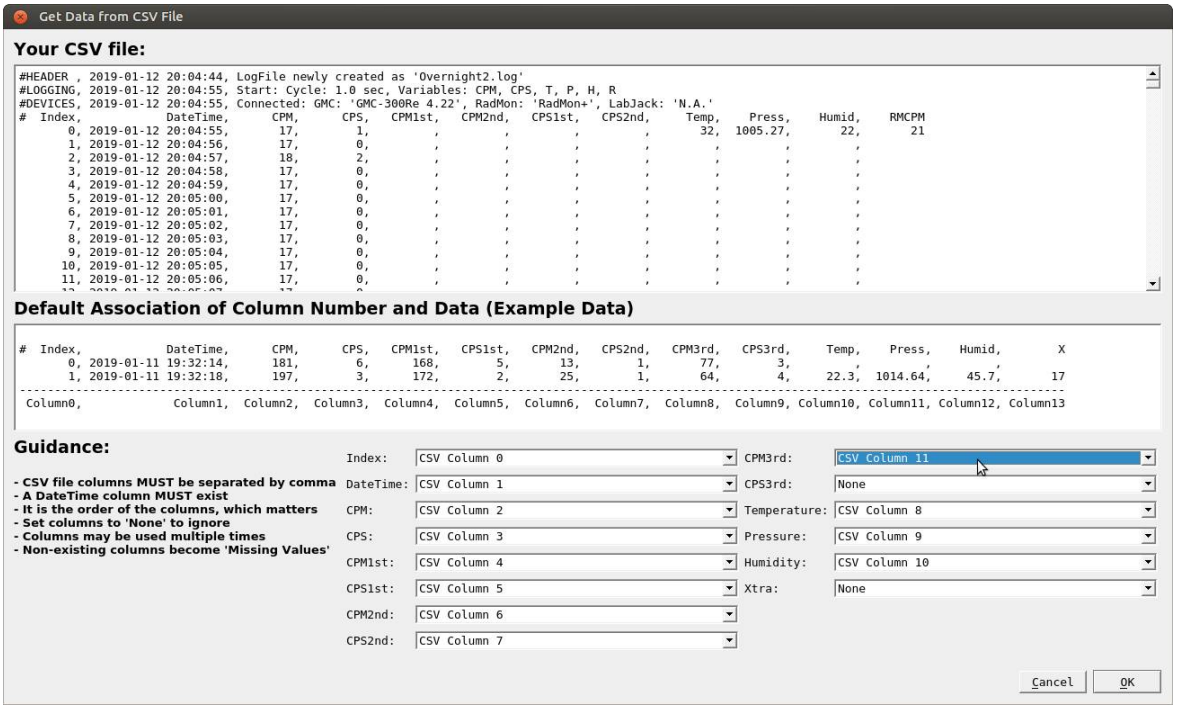


Figure 20: The "Get Data from CSV File" dialogue

The top part shows a segment from the just loaded CSV file, the middle part an example default mapping, and the bottom part allows to associate the columns of your CSV file with the variables of GeigerLog. The CSV file may have up to 20 data columns. The data columns can be associated freely with the GeigerLog variables with the exception of the DateTime column. A DateTime column must exist, it must be associated with the GeigerLog variable DateTime, and it uses the standard format 'YYYY-MM-DD hh:mm:ss", like "2021-01-23 01:23:45".

Saving Data as CSV formatted Files

Using **Log → Save Log Data into CSV file** and **History → Save History Data into CSV file** will save the respective data as a CSV file with extension 'csv'.

Configuration of GeigerLog

GeigerLog uses the configuration file `geigerlog.cfg` located in the subfolder `gconfig` in GeigerLog's working directory. This file is **required**!

If it does not exist or is not readable, a **STARTUP PROBLEM** message pops up as shown in Figure 21, and GeigerLog exits.

Figure 21: Startup Error on missing or non-readable configuration file

NOTE: To modify the configuration file avoid the **Windows** programs **NotePad** and **WordPad**, as even Microsoft confirms [21]) as they may render the file unreadable for GeigerLog. Instead, use Open Source editors like **Notepad++** [22]) or, my preference, **Geany**, available for all platforms (see Editor Geany on page 136).

On **Mac, Linux, and Raspi** any editor will work, but **Geany** is also available for these systems.

Unknown Devices

With the many different devices supported, some having model specific firmware bugs, changing firmware, new features, and more, the configuration file has provisions to allow that even an as yet unknown device can be configured so that it can be made to work with GeigerLog.

Some of the firmware issues are laid out and explained in Appendix F – Firmware Differences on page 162.

Complete

The options in the configuration completely determine the operation of GeigerLog. They will be read only once at the start of GeigerLog.

Permanent versus Temporary

Most of the options are fixed for the run, but some can be changed during a run. Those changes will, however, not be written back into the configuration file. So, they are temporary only, and at the next restart of GeigerLog the old options as laid out in the configuration file will again be used!

21 https://devblogs.microsoft.com/commandline/extended-eol-in-notepad/

22 https://notepad-plus-plus.org/

Pure Text

The configuration file is pure text, and can be edited with any editor which does pure-text editing (observe Windows-Note above!). All options and their defaults are explained within the configuration file.

Uniqueness

All options must be **unique**; any duplicate definition in the configuration file results in another startup error, showing a message box like in Figure 22, giving you detailed info on the problem, and GeigerLog exits.

Figure 22: Startup Error on Uniqueness failure

Missing Device Activation

Devices can only be worked with, if they are **activated** in the configuration file.

If none is activated, then GeigerLog cannot talk to any hardware device, but can still be used to load, show and analyze Log and History data from file.

In the default state no devices are activated; GeigerLog will alert you to the lack of activated devices with this message:

Figure 23: Missing Device Activation

Tube Sensitivities

The conversion of dose rates in CPM & CPS to the more health-relevant dose rate in µSv/h is possible with the use of factors specific for each counter and its tube. This is not a trivial task; read more on this in chapter Appendix G – Calibration.

GeigerLog uses the tube's **Sensitivity** for this, which should be given by the tube's manufacturer. It is defined in units of **CPM / (µSv/h)**. The sensitivity value tells you how many counts-per-minute a tube will generate in a **gamma**-radiating environment of 1 µSv/h. The more sensitive a tube is, the higher the sensitivity values will be! If a radiating source has a significant beta component this Sensitivity will be **NOT VALID**!

Tube sensitivities in current GMC counters range from 2 … 379 with a generic standard tube being labeled as 154 CPM / (µSv/h). A longer list is shown in chapter Appendix G – Calibration.

GeigerLog supports 4 variable-pairs CPM & CPS, and each pair is associated with one tube. The default sensitivities are:

Tube Name	Associated Variables Pair	Sensitivity [CPM / (µSv/h)]
Default tube	CPM, CPS	154.0
1st tube	CPM1st, CPS1st	154.0
2nd tube	CPM2nd, CPS2nd	2.08
3rd tube	CPM3rd, CPS3rd	154.0

These values are configured in the GeigerLog configuration file. They can be changed during a run.

To show the current setting of tube sensitivities use menu **Device → Geiger Tubes …**, or press the **Geiger Tubes** button in the **Device Dashboard** for a dialog as in Figure 24.

The four different tube sensitivities can be seen, edited, and saved for use during the present run. This allows to conveniently experiment with different tubes or settings. This redefinition can be applied at any time, and its changes will be effective immediately, but it is discarded when the program ends, or a device re-activation occurs.

For lasting changes edit the configuration file `geigerlog.cfg` in its section `[TubeSensitivities]`.

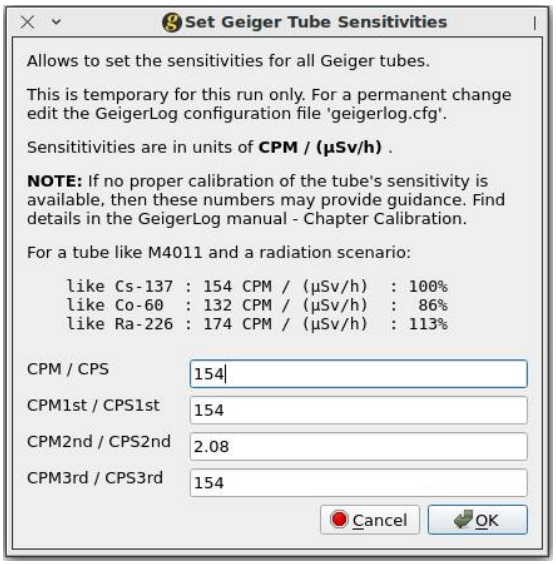


Figure 24: Set Geiger Tube Sensitivities

Auto-Starting GeigerLog

Under certain use cases it is important that GeigerLog can be auto-started from a script. E.g. this is the case when GeigerLog runs on a headless, watch-dog controlled computer e.g. as part of a weather station.

At other times it is simply convenient to use the following auto-start features. To view the options at a computer start GeigerLog with the Help option: `geigerlog -h`, for more see chapter Starting GeigerLog with Options on page 128.

You can use the option '`-l mylogfile`' (or: '`--logfile mylogfile`') and the commands `connect, load, start, quick`:

In the following sets of commands, the **'GL'** needs to be replaced with:

on Linux, Max:	./GeigerLog.sh
on Windows:	GeigerLog.bat

To autostart by connecting to the devices, loading the **last used** `logfile`, and starting logging:

```
GL connect load start
```

To autostart with Quick Log, logging data into default database file `default.logdb`:

```
GL quick
```

To autostart logging data into log file `mylogfile.logdb` :

```
GL -l mylogfile start
```

To autostart loading the log file `mylogfile.logdb` but not starting logging:

```
GL -l mylogfile load
```

To autostart and only connecting to the devices, but not starting logging:

```
GL connect
```

To autostart by connecting to the devices, loading `mylogfile.logdb`, but not starting logging:

```
GL -l mylogfile connect load
```

Quality Control of your Data

The **SuSt**, **Stats, Scat**, **Poiss**, and **FFT** buttons in the Graph Dashboard help you to check the quality of your data. In addition menu option **Tools →Show Plot Data** provides further help.

These functions do:

- **SuSt** gives a brief **Summary Statistics**
- **Stats** a more in-depth **Statistics**
- **Scat** shows a **Scatterplot** of one variable against any other with a Polynomial Regression
- **Poiss** a test for the data having **Poisson Distribution** properties
- **FFT** provides a **Fast-Fourier-Transform**
- **Show Plot Data** prints data to the NotePad, but only those shown in the plot

The first four functions use only the data and variables currently shown in the plot for their calculations! If you want to see the result for all the data in the file, click the **Reset** button in the Graph Dashboard first.

Poiss and **FFT** can only work with one variable at a time. Select this variable with the **Selected Variable** drop down box, see Graph Dashboard – Visualize Your Recordings on page 27.

Furthermore, the variable values will be used in the **units currently selected in the Graph Dashboard**. CPM and CPS values may be shown in units of CPM or CPS, or of µSv/h. Temperature may be shown in °C or °F.

Special Note on Data Quality:

When you deal with Geiger counters using Geiger tubes made from **glass**, be aware that some very strange results may be due to the **light sensitivity** of glass Geiger tubes!

See details in chapter Appendix H – Light Sensitivity of Glass Geiger Tubes on page 165.

SuSt – Summary Statistics

Clicking **SuSt** will give a printout of some summary statistics in the NotePad. It may look like this:

```
==== Summary Statistics of Variables selected in Plot ========================
File       = /home/ullix/geigerlog/geigerlog/data/default.logdb
Filesize   =  1,303,552 Bytes
Records    =     37,998 total,       31,787 shown in Plot
Time Span  =    0.69123 d total,    0.36820 d shown in Plot
Avg. Cycle =        1.6 s total,        1.0 s shown in Plot
          [Unit]   Avg ±StdDev   Variance       Range         Recs  Last Value
CPM     : [CPM]  15.91 ±3.8         14.46      6 ... 31      31787        N.A.
CPS     : [CPS]   0.27 ±0.514        0.26      0 ... 4       31787        N.A.
CPM3rd  : [CPM]  14.98 ±3.93        15.47      2 ... 32      15860        N.A.
CPS3rd  : [CPS]   0.50 ±0.712        0.51      0 ... 6       15889        N.A.
```

As a first easy check for the validity of CPM and CPS values look at Average and Variance – they should be about the same (is the case here), unless you had varying conditions during a recording.

The reason for this lies in the properties of a Poisson Distribution, which is the relevant statistics for radioactive events. For an introduction to Poisson Distribution and its statistics see my "Potty Training for Your Geiger Counter" article available on SourceForge in the Articles folder [23]).

Note that this applies ONLY when the units CPM or CPS are used, and NEVER when µSv/h is used!

Likewise, for the variables temperature, pressure, humidity, and air-quality the comparison of average and variance makes no sense!

23 https://sourceforge.net/projects/geigerlog/files/Articles/

Stats – Statistics

Clicking **Stats** will open a pop-up window showing standard statistics, which will have content like this:

```
==== Data as shown in the plot for selected variable: CPM
======================
from file: /home/ullix/geigerlog/geigerlog/data/default.log

Totals
  Filesize  =    311,786 Bytes
  Records   =        806

Variable: CPM (in units of: µSv/h)
                      % of avg
  Average   =    0.13      100%       Min  =    0.07        Max  =    0.21
  Variance  =    0.00     0.60%
  Std.Dev.  =    0.03    21.74%       LoLim=    0.10        HiLim=    0.15
  Sqrt(Avg) =    0.36   281.23%       LoLim=   -0.23        HiLim=    0.48
  Std.Err.  =    0.00     0.77%       LoLim=    0.13        HiLim=    0.13
  Median    =    0.12    97.68%       P_5% =    0.08        P_95%=    0.17
  95% Conf*)=    0.05    42.61%       LoLim=    0.07        HiLim=    0.18

  *) Approx. valid for a Poisson Distribution when Average > 10

Time
  Oldest rec    = 2018-08-19 13:49:41  (time=0 d)
  Youngest rec  = 2018-08-19 14:29:56  (time=0.028 d)
  Duration      = 2415 s   =40.25 m   =0.6708 h   =0.02795 d
  Cycle average = 3.00 s

First and last 7 records:
#HEADER , using Quick Log file: default.log
#LOGGING, 2018-08-19 13:41:50, Start with logycle: 3.0 sec
#LOGGING, 2018-08-19 13:41:50, Log variables: CPM, CPS, T, P, H, R
#LOGGING, 2018-08-19 13:41:50, Connected GMC Device: 'GMC-300Re 4.22'
#LOGGING, 2018-08-19 13:41:50, Connected RadMon Device: 'RadMon+'
#Index,           DateTime,      CPM,      CPS, Temp,   Press, Humid,  Xtra
    0, 2018-08-19 13:41:50,       16,        1, 27.2, 1012.52,    36,    14
...
 2388, 2018-08-19 15:41:14,  6778.79,    121.9,
 2389, 2018-08-19 15:41:17,  6761.01,   116.66,
 2390, 2018-08-19 15:41:20,  6773.56,   110.38,
 2391, 2018-08-19 15:41:23,  6794.47,   109.34,
 2392, 2018-08-19 15:41:26,  6803.89,    96.84,
 2393, 2018-08-19 15:41:29,   6801.8,   127.15, 29.2, 1011.5,     35,    22
 2394, 2018-08-19 15:41:32,  6796.57,   114.57,
```

Poiss – Histogram with Poisson Fit

This tool is relevant ONLY to Geiger counter data as CPM or CPS, but for these it is immensely useful!

It does NOT make sense to use it when Geiger counter data are shown in µSv/h (or any other dose rate, like mR/h or else). It also does NOT make sense to use for environmental data, like temperature, barometric-pressure, humidity, air-quality, as none of these have an underlying Poisson distribution!

The next two figures provide examples of histograms with a Poisson fit; Figure 25 for low count rates as in a background measurement, and Figure 26 for a much higher count rate.

The value **r²** (in the graph as r2) is an indicator for the goodness of a fit. **A value of r² ≥ 0.9 suggests a proper measurement.** If r² is smaller, then there may not be enough data points for a meaningful average, or some experimental error (source or counter shifted or removed during data collection ?) may have occurred.

Use the Poisson Test as an essential quality control tool for your measurement.

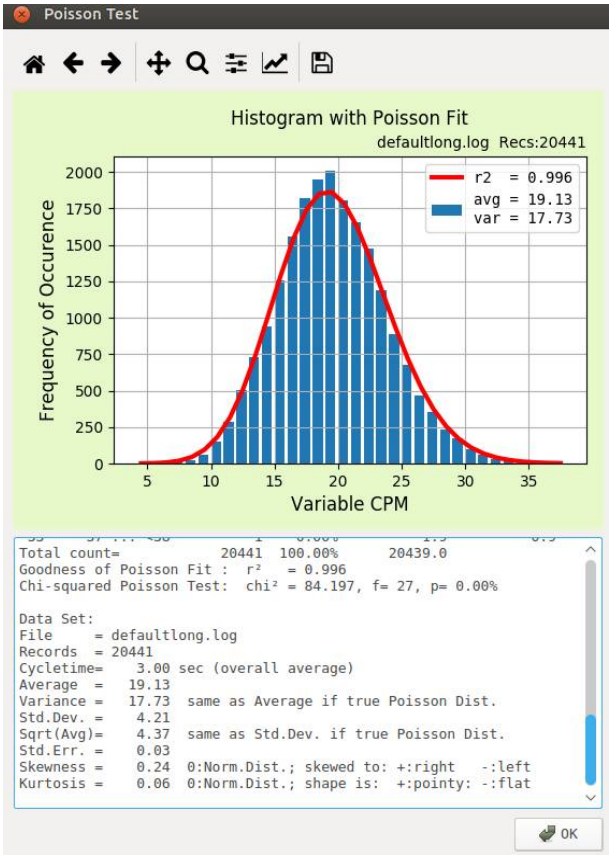


Figure 25: Histogram of Low Count Rate

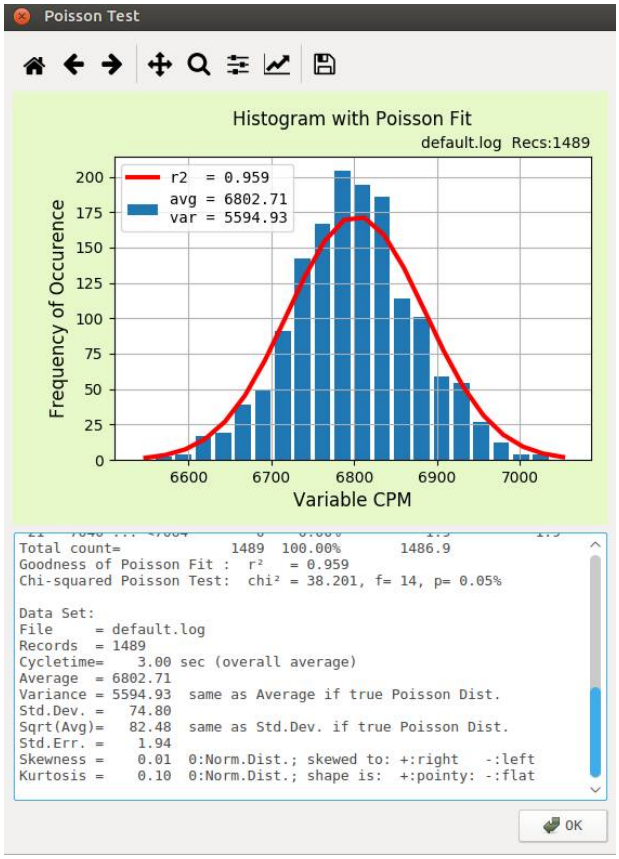


Figure 26: Histogram of High Count Rate

FFT – FFT & Autocorrelation Analysis

The FFT (Fast Fourier Transform) allows to analyze a time dependent signal, like the Count Rate, for any periodic signal hidden within the data. An example is given in Figure 27. The data were recorded by logging in the CPM mode.

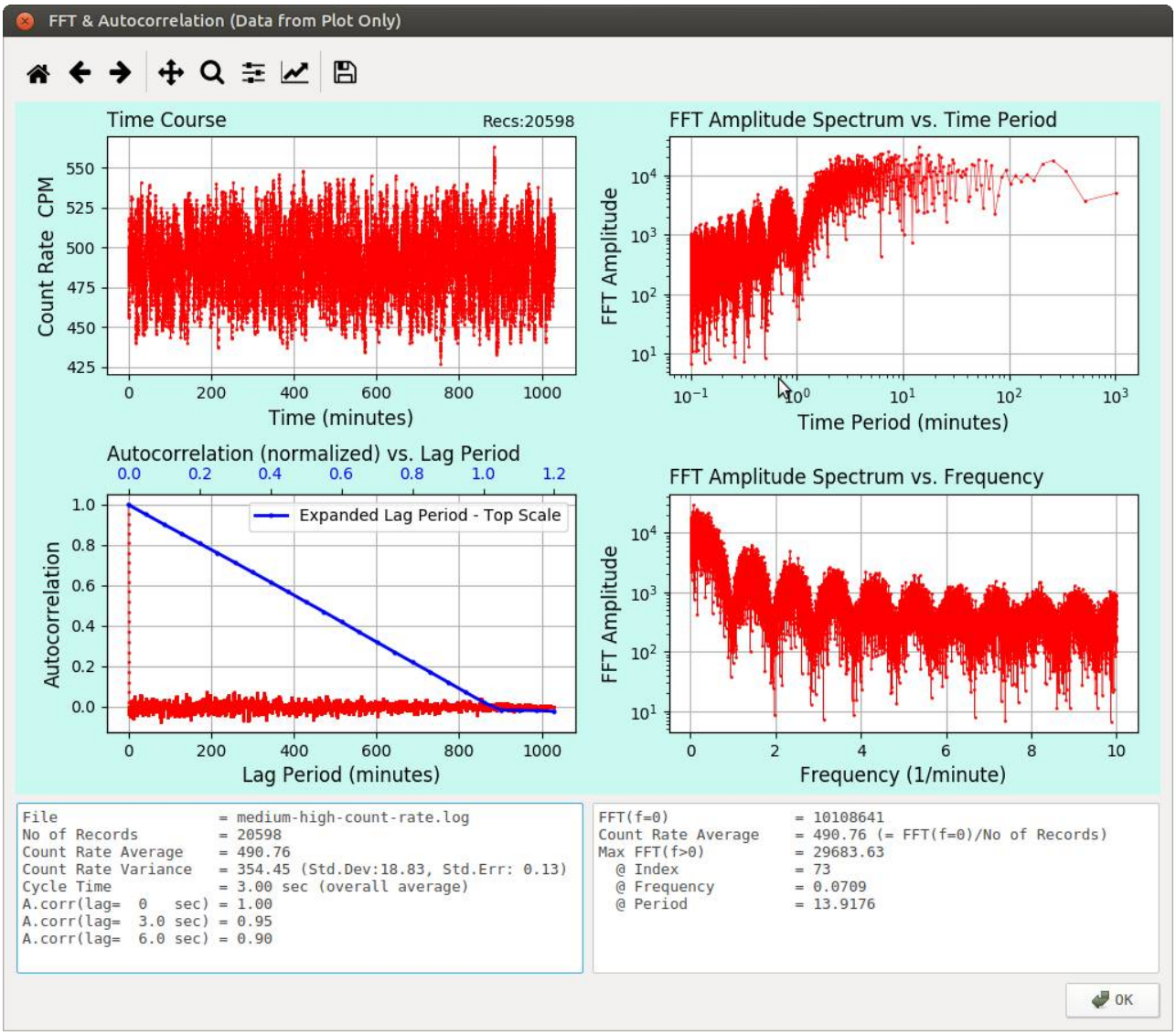


Figure 27: FFT Analysis of Medium Count Rate Measurement

With a bit of squinting at the Time Course of Count Rate vs. time (upper left panel) one may expect to find a signal with a period of 1 or more hours; at least I did. However, in the range of (upper right panel) >1 to 1000 minutes there is no such signal. Instead there is a very pronounced signal at a period of 1 min, equivalent to a frequency (bottom right panel) of 1/minute. This frequency plot clearly also shows all the harmonics of this frequency.

The effect is independent of count rate (same pattern at background count rates) and sampling time (< 30sec). At a sampling time of > 30sec, this signal would not be observable anyway due to the Nyquist limit.

The fact of a pronounced 1 **min** Period in the FFT spectrum, and the Counts per **Minute** sampling, raised the suspicion, that this was related. But, as was first considered, it has nothing to do with the Geiger counter taking a little break every minute. Rather, it is the consequence of oversampling.

In this experiment the CPM readings were taken every 3 seconds. CPM is the sum of readings during the last 60 seconds. The next reading 3s later has 3 “fresh” seconds of data, and has dropped 3 “old” seconds of data. But 57s worth of data remain unchanged. Which means that all data taken over 60s are related, strongly initially, and weakly at the end.

Such a relationship can be quantified by calculating the autocorrelation of a signal. This is shown in the bottom left diagram of Figure 27. The data are redrawn in blue vs. an expanded Lag Period (labeled on top of this panel). And, indeed, one sees the autocorrelation dropping linearly from the initial 1 (highly correlated) to the 0 (= non-correlated) at exactly 1 minute.

So, it is auto-correlated, what does it have to do with the FFT spectrum? The autocorrelation can be seen as the convolution (or folding, different name for the same thing) of a rectangle in time of length 1 minute and a Poisson distribution of the Geiger data. The FFT spectrum is then a mix of the rectangle spectrum and the Poisson spectrum.

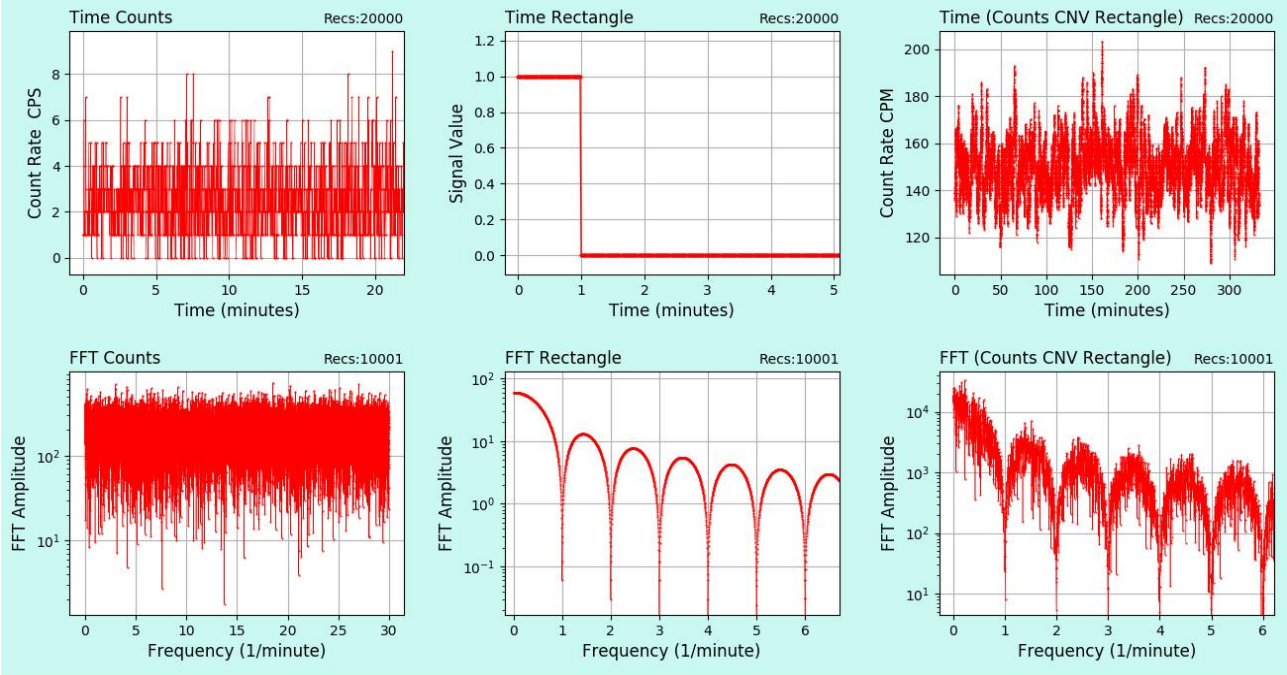


Figure 28: Demonstration of the impact of convolution on an FFT spectrum

(Synthetic data, CPS=2.5)

This can be nicely demonstrated using synthetic recording. In Figure 28 the upper panels show the signals in the time domain from Poisson White noise at average CPS=2.5 (upper left), a rectangle of 1 min at value 1 and value 0 for the remaining 19988 counts (upper middle), and the convolution of these two signals (upper right), resulting in average CPM=150. The bottom panels show the corresponding FFT spectra, white noise, a 1/min frequency and harmonics, and the mix of the two. Oversampling does no harm; but it must be accounted for when autocorrelation plays a role.

Scat – Scatter Plot

It is sometimes helpful to plot one variable against another one in an X-Y-scatter plot. You might be wondering whether temperature is correlated with humidity? Or the barometric-pressure has an influence on the temperature?

More relevant to Geiger counters was the recent observation [24]) that a GMC Geiger counter equipped with a M4011 tube (black version) was significantly sensitive to temperature, which it shouldn't be!

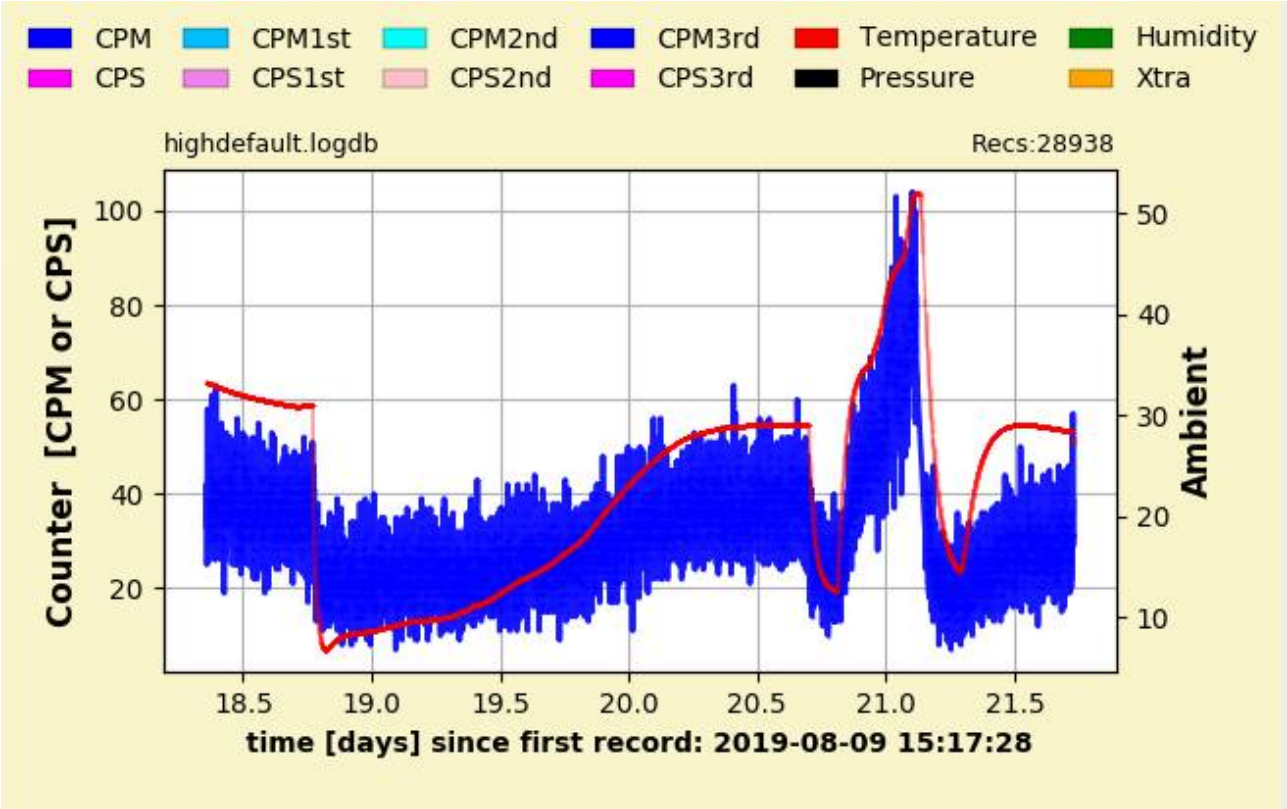


Figure 29: CPM and Temperature in a Temperature Experiment with a Geiger Counter

(GMC-300E+ with a M4011 tube - black version)

It is obvious from the time course plot in Figure 29 that CPM (blue curve) increases and decreases with temperature (red curve). The degree of this correlation can be better demonstrated by plotting a scatter plot of CPM versus temperature.

Press the Scat button. In the upcoming dialog you choose the variables for the X- and Y-axis, which in this example will be X = temperature and Y = CPM as shown in Figure 30.

You can also choose whether you want to show X, or Y, or both to include their origin (0, zero) in the plot, and whether to draw connecting lines between data points. Default is no showing of zero, and drawing connecting lines, which on most occasions is the best option.

Furthermore, you have the option of including a polynomial Least-Squares-Regression fit to the data, and can choose the order of the polynomial from "Prop" (y=m*x), 0 (equals average) to 7.

24 http://www.gqelectronicsllc.com/forum/topic.asp?TOPIC_ID=7475

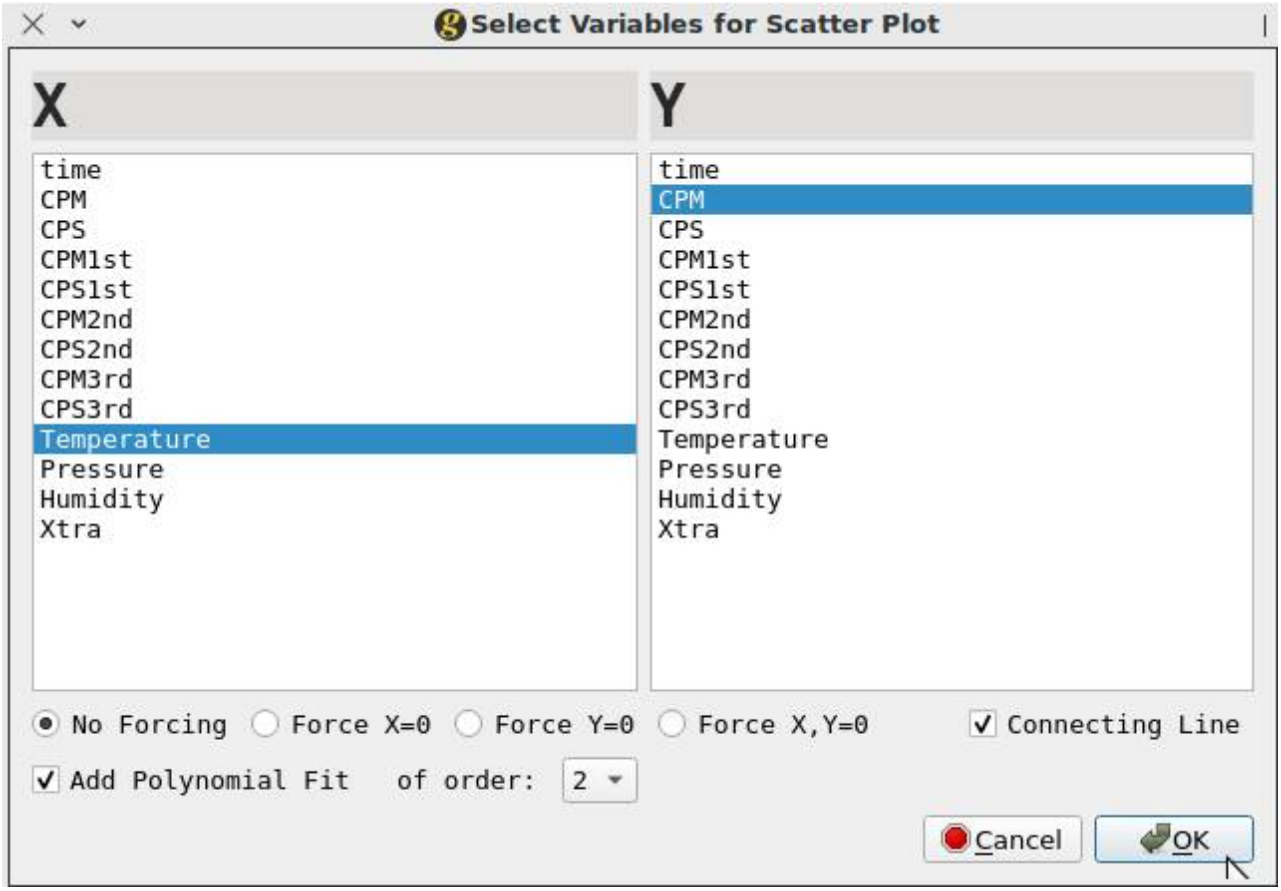


Figure 30: Select Variables for Scatter Plot

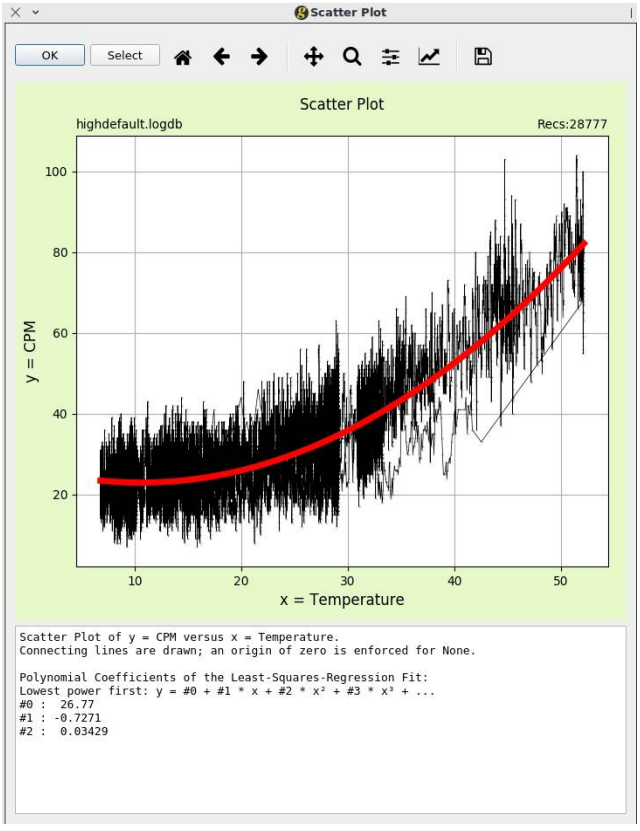


Figure 31: Scatter Plot with 2nd Order Polynomial Fit

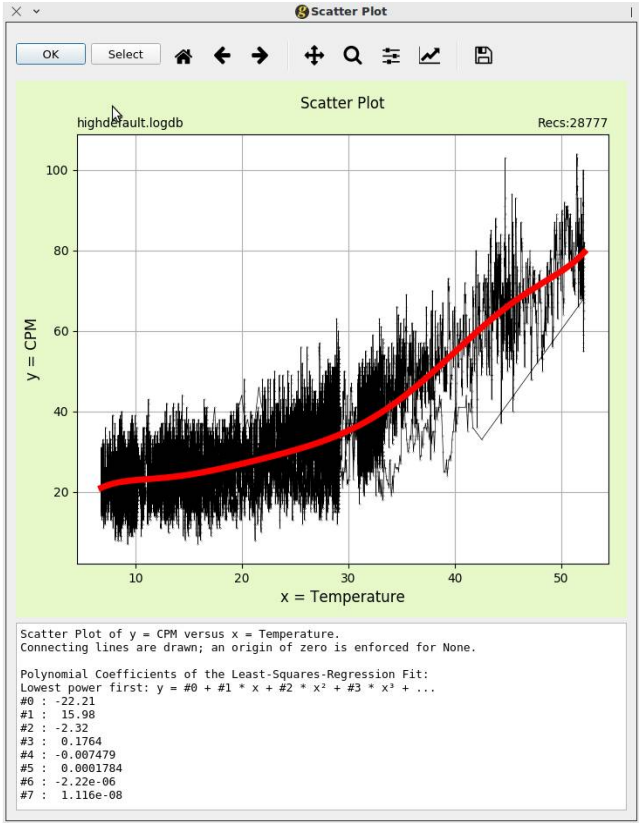


Figure 32: Scatter Plot with 7th Order Polynomial Fit

In Figure 31 you see a scatter plot of y = CPM versus x = Temperature plot in black, fitted with a polynomial of second order (also called a quadratic fit) in red.

Clicking the OK button (top left) will close the scatter plot. The Select button brings you back to the variable selection as shown in Figure 30. The choices you had just made, are maintained. Let's change the order of the fit from 2 to 7, and click OK.

Figure 32 shows the result with now a fit of the same data with a polynomial of 7th order.

With these data one can barely see a difference between the two fits. However, with other data the high order fits may show curvature which simply is meaningless! Caution must be applied to choosing an order for the fit; high orders may result in over-interpretation of data. Generally a lower order is more adequate; more than a 2nd order will rarely ever be needed nor meaningful!

Another issue which demonstrates the usefulness of a Scatter Plot was of particular relevance for this version of GeigerLog: given that we can measure Geiger counts with one single GMC-device by both the technique of digitally transmitting the data via the USB cable, and at the same time by the audio cable, the two measurements should be strongly correlated. Are they? I use the data referred to in the chapter AudioCounter Devices on page 70.

In the Select Variables dialog (Figure 30) choose the variables as X = CPM (digital data) and Y = CPM3rd (audio data). Uncheck Connecting Line, and choose a fit of order = 1, i.e. a linear fit. The result is shown in Figure 33, demonstrating the excellent correlation between the two.

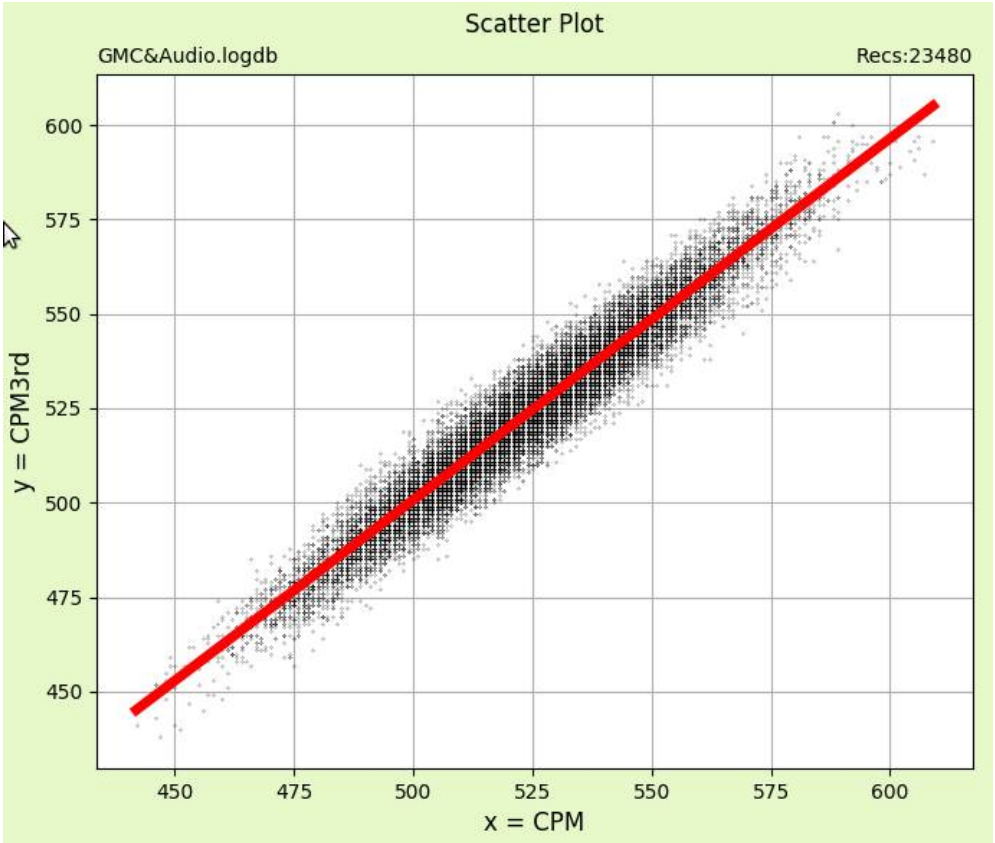


Figure 33: Scatter Plot of CPM3rd=Audio versus CPM=Digital Signal
(Using a GMC-300E+ Geiger Counter)

Show Plot Data

Sometimes you want the numerical values of your variables. In the **NotePad Dashboard** click the **Data Plt** button, or use menu **Tools→Show Plot Data** to print the Date&Time and the values of variables into the NotePad, but print only those variables currently shown in the plot, and only for the time frame selected in the plot, as shown in Figure 34. Makes it easy to inspect values within a limited range.

Print: You can print this selection on paper or as a pdf file with the **Print** button in the **NotePad Dashboard** or via menu **File → Print NotePad.**

Save: You can also save this printout to a file using the **Save** button in the **NotePad Dashboard** or menu **File → Save NotePad to File.** This will create a CSV (Comma Separated Values) file. It can be read back into any GeigerLog, which is a convenient way to forward select data to other users.

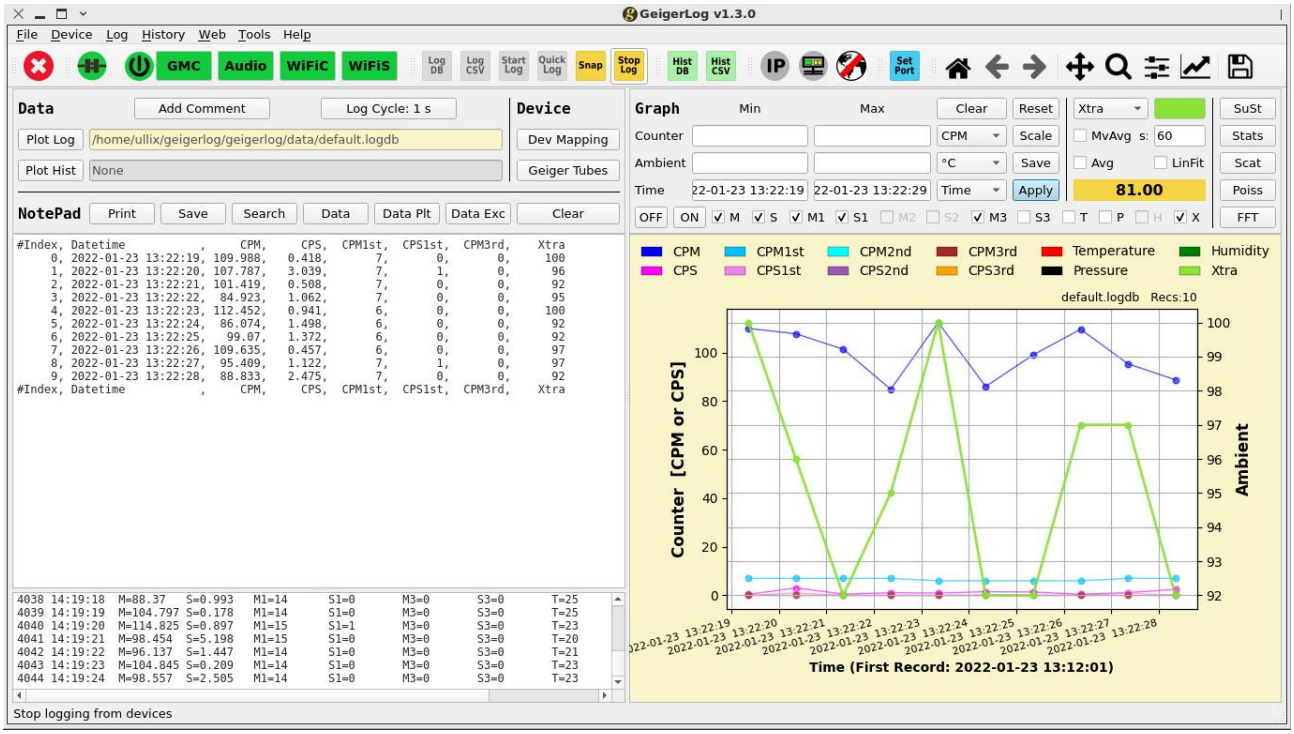


Figure 34: Show Plot Data showing only the values of data currently in the plot

Device-specific Considerations

These devices are available:

1. GMC Devices
2. AudioCounter Devices
3. Audio-To-Serial Devices
4. IoT Devices
5. RadMon Devices
6. RadPro Devices
7. Gamma-Scout Devices
8. I2C Devices
9. LabJack Devices
10. MiniMon Devices - CO2 Monitor
11. Formula Devices
12. Manu Devices

    WiFi Devices

13. WiFi Devices – WiFiClient Devices
14. WiFi Devices – WiFiServer Devices

    Raspi Devices

15. Raspi Devices – RaspiI2C Devices
16. Raspi Devices – RaspiPulse Devices
17. Raspi Devices – Raspi as Satellite to GeigerLog
18. SerialPulse Devices

GMC Devices

GMC Devices are being fully controlled by GeigerLog. All communication with the device is error-checked and corrected if possible.

Device functions like Speaker, Alarm, Saving Mode, Date&Time, Calibration, Threshold, can be read by GeigerLog, and some can also be set, which is much more convenient than clicking through the counter's menus!

Operating Modes

GMC Devices support the operating modes **Logging** and **History**.

For **Logging** a connection to GeigerLog must have been established, and the device must be powered-on. Then it is GeigerLog initiating and executing all communication and data transfer to and from the device.

History is a stand alone operation of the device. The counter must be powered-on, and then collects data and stores them in its internal memory controlled by its own microprocessor. It may remain connected to a computer and GeigerLog, but it does not have to.

However, in order to read the data from the device, it needs to be connected to GeigerLog. GeigerLog will initiate and execute the data transfer from the device. The data can then be handled in GeigerLog as if they were a logging recording (except that you cannot add more data to it).

Connecting

GeigerLog can auto-detect the presence of a GMC counter and make the correct settings, thus providing for **plug-and-play** use of **GMC counters**!

If this fails then your computer may lack an installed driver. Look for installation files for a **CH340** chip (VID:PID=0x1A86 : 0x7523) and install.

If connections still fail, then chances are something else is wrong; verify cable and plugs.

You can try to set the port manually from menu: **Device → GMC Series → Set Serial Port …**, see Figure 35. From the drop-down box select any port to try. Setting "auto" might still be your best bet.

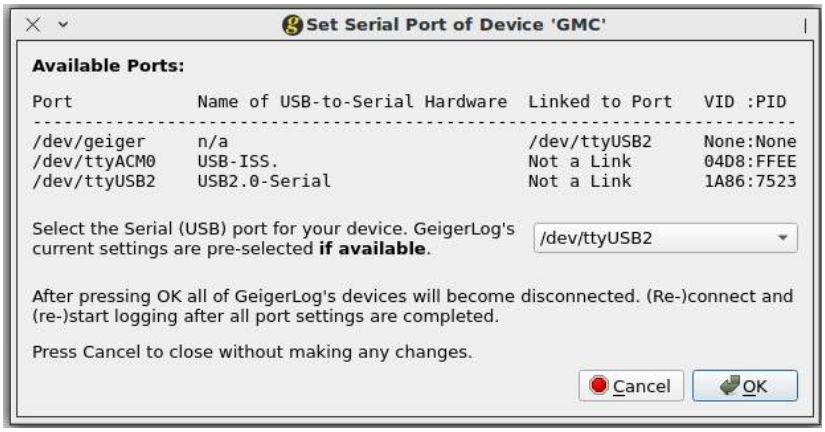


Figure 35: Set Serial Port of Device 'GMC'

If you do get a connection only with a specific port, you can pre-configure it in GeigerLog's configuration file `geigerlog.cfg` in section [GMC_Device] as GMC_usbport = <your port>.

Once you get a successful connection, the GMC icon turns green, and a confirming message is printed to the NotePad:

```
==== Connect GMC Device ============================================================
A device GMC Device was user configured for port: '/dev/ttyUSB2'
GMC Device 'GMC-500+Re 1.22' was detected at port: /dev/ttyUSB2 and baudrate: 115200
Device successfully connected
```

You're set to go.

Connecting Multiple GMC Counters to the Same Computer

While GeigerLog supports 16 different devices, only a single one of each can be connected to one instance of GeigerLog. However, you can make as many copies of the complete GeigerLog folder as you like, and each copy can run the same devices as all the others.

Of course, you also need to have separate hardware devices to run in each instance!

However, the next problem is that you have to be able to distinguish the hardware! In the case of GMC counters, this is anything but trivial, because they all share the same USB-To-Serial chip, and therefore, to the operating system, to Python, and to GeigerLog they all look the same.

I took the recent plug-and-play advances one step further and now allow to configure the required name (e.g. "GMC-500", or "GMC-300", or other) and even allow to configure the serial number. In the configuration file this looks like this:

```
# GMC_DEVICE ID:
# This forces GeigerLog to connect only when the specified GMC device is
present.
#
# Option auto becomes None, None
#
# Options: auto | < <Model>, <Serial Number> >
# examples: GMC_ID = GMC-500, F4884567890ABCDE
# GMC_ID = GMC-300, None
GMC_ID = auto
```

On "auto" all will be ignored, and GeigerLog will use whichever GMC counter it finds (provided it is activated). But I could connect both a GMC-300 and a GMC-500 counter to my computer, and run one instance of GeigerLog with one, and a second instance with the other counter.

As an aside: if I wanted the two resulting data files combined to a single one, I could use program **`Glmerger.py`**, in the **gtools** directory, to combine them.

Your GMC Geiger Counter Model

GeigerLog works the same for all GMC Devices except for some workarounds accounting for the different firmware, firmware bugs, memory sizes, calibration factors and more. It is therefore im-

portant that after you have made the connection the correct Geiger counter model and firmware is shown in the printout to the NotePad, as shown above.

If this is not the case, then you may have to customize your model by modifying the configuration file `geigerlog.cfg` in its **GMC_Device** section. Some of past problems are highlighted in Appendix F – Firmware Differences on page 162.

It is now assumed that a successful connection of GeigerLog with the Geiger counter has been made.

Powering On

For a working connection between computer and Geiger counter, the counter does not have to be switched on (powered on); it can remain off. The power for its electronics comes from the USB port, thereby also charging the battery. In this mode you can read and set various parameters of the counter, and you can download the history.

But for all new radiation measurements – be it by Logging or by History – the Geiger counter must be powered on. This power switching can be done manually directly at the device, or easier from GeigerLog (menu **Device → GMC Series → Set GMC Configuration …**). GeigerLog's GMC device power icon will change its state from Power OFF (red) to Power ON (green). The icon is gray when the device is not connected and thus GeigerLog could not determine the power state of the counter.

More convenient is to toggle the power state by clicking the Power icon on the toolbar!

Logging

Any logging is strictly controlled by GeigerLog, **not** by the counter [25])!

For every value GeigerLog wants to have, it must send a specific command to the counter. The counter answers with the data. After GeigerLog has obtained all values for one cycle, it saves them as one record, prints them to the LogPad, and displays them in the graph. Then it waits for the cycle time to expire to start asking for the next record of data.

The values which GeigerLog can ask for are:

CPM	: Counts Per Minute
CPS	: Counts Per Second

Since the release of the GMC-500+ counter, which has not just one but two Geiger tubes installed, its firmware was extended to allow reading the tubes individually. For this device the values which GeigerLog can ask for are:

CPM	: Counts Per Minute as the sum of both tubes (makes no sense [26]))
CPS	: Counts Per Second as the sum of both tubes (makes no sense)
CPM1st	: Counts Per Minute for the 1st tube, the standard tube

25 This is different from the way GQ's Dataviewer software works. DV uses the outdated heartbeat function of the counter, which only provides CPS readings, and does not allow any simultaneous other communication with the counter. Thus it is impossible to use any of the more recently introduced functions for reading more than a single tube.

26 See discussion e.g. here: http://www.gqelectronicsllc.com/forum/topic.asp?TOPIC_ID=5304

CPM2nd : Counts Per Minute for the 2nd tube, the low-sensitivity tube
CPS1st : Counts Per Second for the 1st tube, the standard tube
CPS2nd : Counts Per Second for the 2nd tube, the low-sensitivity tube

These commands work error-free on all counters, also on those with single tubes only, as well as with older firmware, but on all devices, except the GMC-500+, the answers are redundant:

CPM = CPM1st = CPM2nd
CPS = CPS1st = CPS2nd

No rule without exceptions: in the GMC-600+ counter (single tube) it remains true that CPM = CPM1st and CPS = CPS1st, but CPM2nd and CPS2nd are always 0 (zero)!

An example of a Logging with a GMC-device in combination with an AudioCounter as a 2nd device is shown in Figure 37 in chapter AudioCounter Devices.

History Background

Any GMC Geiger counter can measure the counts from radiation and store the results in its internal memory, not needing a computer connection. In the older units this memory size is 64 kB (65 536 Bytes). Newer units have an internal memory of 1 MB (1 048 576 Bytes), and the latest device, a GMC-800, has even 2 MB (2 097 152 Bytes).

The maximum storage duration depends on the storage mode – storing CPS is done every second, while storing CPM only every minute – and also strongly on the intensity of the radiation due to the storage algorithm implemented in the Geiger counter firmware, since a value of up to 255 can be stored in a single byte, while everything larger requires 5 (!) bytes!

In addition the counter stores a Time stamp of 12 Bytes. In CPS-mode it is saved every 3 min, i.e. every 180th record, in CPM-mode, the time stamp is added once per hour, i.e. every 60th record.

I tested with a GMC-300E+ counter, which has 64 kB of memory, and CPS-mode. With a radioactive source the count rate was about CPS=30. This value is well under 255, so it would result in the longest possible storage duration.

Overall, this resulted in a recording window for CPS of:

64 kB memory: **16.15 h**
1 MB memory: **258 h = 18.4 days**

After this time span the counter memory and the counter begins overwriting the oldest data.

For a CPM-mode measurement the memory would last much longer. For background radiation, typically around CPM=20...30, a single byte would suffice for storage, but e.g. for the example above with CPS=30, resulting in CPM=1800, 5 Bytes would be needed for each CPM value!

The estimate is difficult to make, but for a 64 kB memory it should last from 1 to 5 weeks.

This was seen with a GMC-300E+ and a GMC-500+ counter. No guarantee that other counters and firmware behave the same!

However, a much larger memory is not necessarily an advantage. Downloading just the 64 KB already takes about 25 sec at the fastest serial speed! Downloading 1 MB takes about 5min. This is where a faster speed would really be helpful.

In theory you could download only a portion of the memory. But since this is laid out as a ring-buffer, you'd have to know very precisely what portion of the memory you want. Typically you won't know this until after you have done the complete download and inspected the data. On top of this, a partial download may bring parsing problems (see Appendix E – GMC Device: Internal Memory, Storage Format and Parsing Strategy). So this is **not** an option.

As a 5 min download is really inconvenient, the GeigerLog protocol for the download has been modified to: the download will be stopped when 8192 bytes, each having hex value 'FF', have been read. Unfortunately, 'FF' can both be a legal value for CPM or CPS, but also signals erased or empty memory. As the memory is organized in pages of 4k bytes each, it means that 2 successive pages of 'FF' must be found. This can only be the case when nothing is written neither into these 2 pages nor into any pages beyond them. So we can safely stop downloading.

However, if the memory overflows, the ring-buffer (see Appendix E – GMC Device: Internal Memory, Storage Format and Parsing Strategy) storage principle becomes effective, and the memory is overwritten beginning at the bottom. In this situation the whole memory is filled with data, and there will never be 2 pages of empty values. Hence the whole memory will be read!

Erase Saved Data

If you don't need the content of the memory, I suggest to erase it every once in a while. Some newer counters provide a separate command in their menus to erase the memory, but older counters do not.

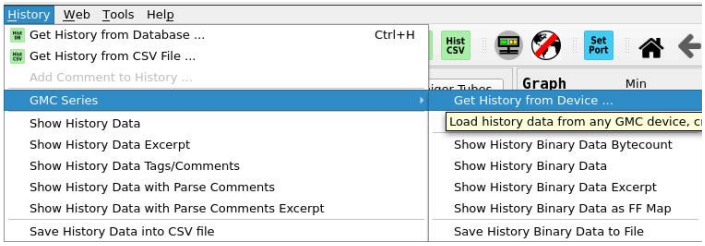
However, in both cases you can run the GeigerLog command **Device → GMC Series → Erase Saved Data …** to erase the complete History memory. The command will run about 10 sec.

History Handling

Handling the History is controlled by a button in the toolbar, and commands available in Menu – History.

The button **Hist DB** loads data from a database file created by a previous history download, see more at Menu – History. The menu command **History → Get History from CSV File** reads data from a file in CSV format and creates a database.

When a GMC device is activated then the menu History offers additional options. Once a GMC device is also connected, you can **download** the History from the device.



Note that with ongoing Logging this History Downloading cannot be done, and the respective command is grayed out, as the GMC Devices are limited computationally and stumble with parallel logging and downloading!

Additional commands in the menu History are:

History → GMC Series → Get History from GMC Binary File reads a binary history file created by an earlier version of GeigerLog, or by a different software, parses the data, and creates a regular database file. Such a History example – data collected during a long distance flight – is shown in Figure 1.

Other commands provide details about the binary data and data structure.

History → GMC Series → Save History Binary Data to File save the raw data to a file with extension 'bin'.

History Saving Mode

The GMC counters can use different strategies to save the data into the history memory, ranging from not saving at all, to saving CPS or CPM in different time intervals, or even conditional on exceeding a count threshold.

The saving mode can be switched in the dialogue box **Set GMC Configuration** from the menu: **Device → GMC Series** and select from the options under **History Saving Mode** (see Figure 36):

Mode: OFF	- OFF (no history saving)
Mode: CPS	- CPS, save every second
Mode: CPM	- CPM, save every minute
Mode: CPMh	- CPM, save hourly average
Mode: CPSTh	- CPS, save every second if exceeding threshold
Mode: CPMTh	- CPM, save every minute if exceeding threshold

I strongly **discourage using the threshold modes**, as they distort the data and may make interpretation difficult or impossible, because you loose all knowledge on Poisson properties of the data!

Assembly of the Device

No assembly needed. Just a USB 2.0 A to Mini-B Cable is needed like this: https://www.amazon.co.uk/AmazonBasics-Male-Mini-B-Cable-Feet/dp/B00NH11N5A

GeigerLog's Configuration

GeigerLog auto-detects the type of connected GMC-device and adjusts itself to match features, and correct deficiencies as well as any known firmware bugs of the connected device.

However, sometimes GQ releases a new device or new firmware without disclosing even essential changes. In those situations you may have to study the many settings in the configuration file `geigerlog.cfg` and make adjustments. Chances are good that you can make even a new device work. It may take some effort, though.

A 'Factory-Reset' is recommended to be sure of a defined starting condition.

While all settings can be modified via GeigerLog's configuration file, the exception is the baudrate used in the USB-To-Serial converter. This can **only** be changed at the counter itself. I found the factory-set baudrate of 115200 working well on a GMC-500+, whereas on a GMC-300E+ a baudrate of 115200 produced more hiccups in communication than its default of 57600.

Editing the GMC counter's internal Configuration

From the menu use **Device → GMC Series → Set GMC Configuration …** to bring up the dialogue box shown in Figure 36.

The upper half applies to all GMC counters and allows to change Power, Alarm, Speaker, History Saving Mode, and Calibration Points. The calibration points are the internal ones of the counter. GeigerLog ignores them and uses its own calibration.

The lower half is for specific counters, which have the respective features enabled. WiFi-enabled counters e.g. are the GMC-320V5, and the GMC-500/600 series counters. For all other counters this lower part is grayed out and cannot be edited.

If you have already set the WiFi related information in the GeigerLog configuration file `geigerlog.cfg`, then you fill out this form by just pressing the **Show User Configuration** button. Pressing the **Show Counter's Active Configuration** button will reset all information to what just had been read out from the counter.

Upon pressing **OK** everything will be written into the counters configuration memory.

Fast Estimate Time (FET) CAUTION !

The last option in the WiFi section is the **Fast Estimate Time** (FET). While this is not related to WiFi, it exists only with some WiFi enabled counters, like GMC-500 and 600 with newer firmware.

By default this is set to 3 seconds, which by GQ is labeled "Dynamic" mode. It was found to be the reason for completely distorted measurements, even for the creation of counts where there are none! It is discussed in depth in forum topics 9497 and 9506 [27]). Other settings are 5, 10, 15, 130, 30, 60 seconds. The algorithm is not fully disclosed by GQ, except that at 5 seconds and greater the CPM value is estimated on the basis of counts for number of seconds selected by the simple formula:

$$CPM_{FET} = \text{Sum of CPS counts during FET seconds} * (60 / FET)$$

This necessarily results in distorted Poisson distributions, and in the default Dynamic setting even in false data. A setting of FET = 60 obviously switches FET off.

It is strongly suggested to switch the FET setting OFF, i.e. set it to 60 seconds!

Unfortunately, after every Factory Reset this will be back to '3', as this is the default, and will have to be switched off again.

27 http://www.gqelectronicsllc.com/forum/topic.asp?TOPIC_ID=9497 , http://www.gqelectronicsllc.com/forum/topic.asp?TOPIC_ID=9506

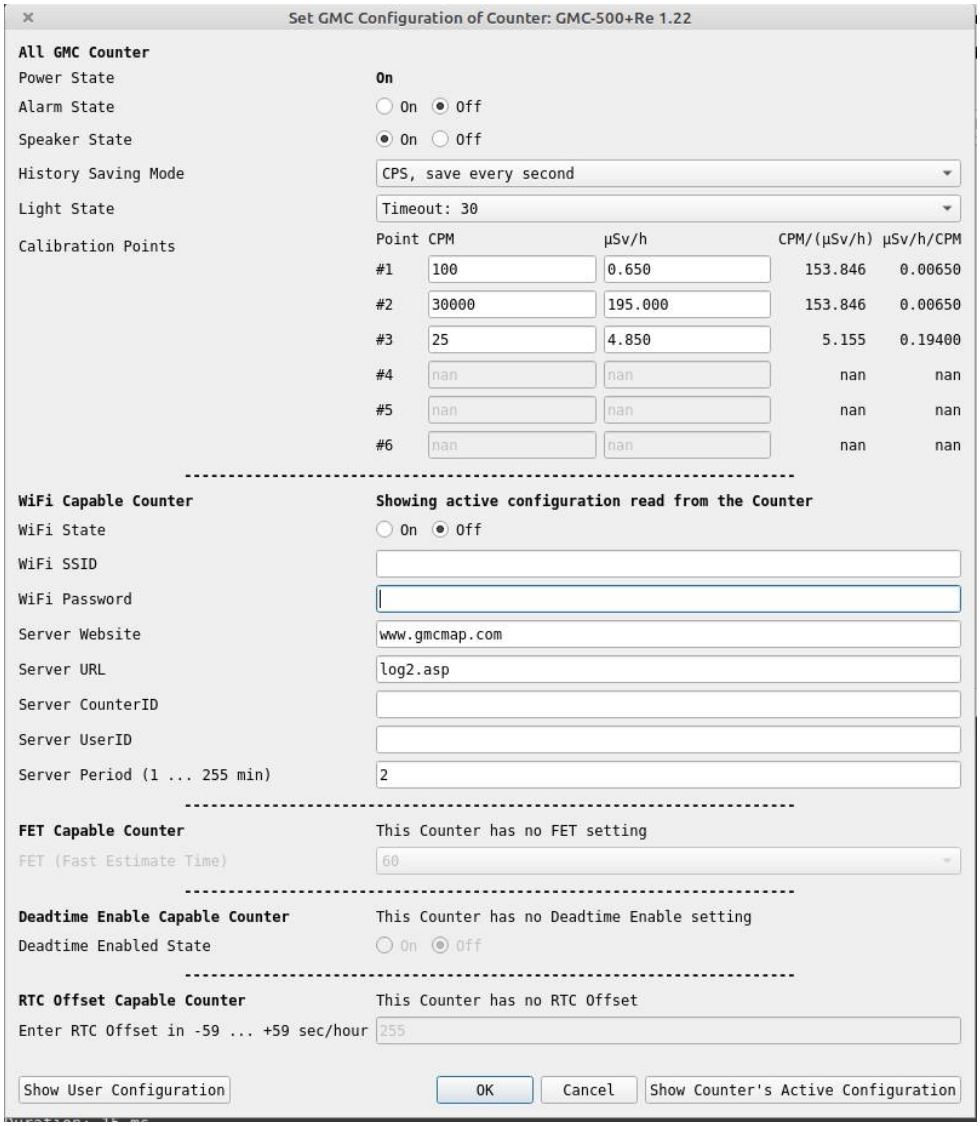


Figure 36: The dialogue to edit the GMC counter's internal configuration. This screenshot is from a GMC-500+ counter, which has 2 tubes.

WiFi Equipped GMC Counter

Out of GQ's line of Geiger counters some are WiFi equipped: series GMC-320+V5, GMC-500, GMC-600. Unfortunately, the intended use of this WiFi is only for the update of GQ's Radiation World Map on their gmcmap.com server (see Radiation World Maps on page 138, and Radiation World Map on page 36).

However, with a bit of hacking the data can now be redirected to GeigerLog, see WiFi Devices – WiFiClient Devices on page 97. Here is shown only the operation for updating the gmcmap server.

After registration you will have received a User ID (AID) and a counter ID (GID). GMC counters are limited to 3 variables named: **CPM**, **ACPM**, **uSV** [28]).

CPM is the instantaneous value of the counter's CPM.

28 https://www.gmcmap.com/AutomaticallySubmitData.asp

ACPM is an averaged value of CPM; it is calculated as the total number of counts accumulated by the counter since start (or reboot, reset) divided by the total expired time since that time-point.

This also is an unfortunate choice, as discussed in the forum [29]), because over time this value becomes less and less sensitive to changes!

uSV is calculated from CPM as CPM / 154 CPM / (µSv/h), i.e. from the instantaneous value, and NOT from a more meaningful average!

The URL, when the counters are contacting the gmcmap server, looks like:
http://gmcmap.com/log2.asp?AID=123&GID=456&CPM=41&ACPM=41.18&uSV=0.27

The gmcmap.com server will answer with:
\r\n<!-- sendmail.asp-→\r\n\r\nOK.ERR0

or, when AID and/or GID is missing or incorrect:
User is not found.ERR1.
Counter is not found.ERR2.

The counter wants to see the answer, but only the ERR0, ERR1, ERR2 words do seem to matter.

Geiger Counter Tube Sensitivities

For the general discussion please see chapter Appendix G – Calibration on page 164. For the GMC counter the situation is unfortunately more complex, which warrants this extra chapter.

GQ is using the term “calibration point” instead of Sensitivity to describe the relation of counts as CPM/CPS to dose rate in µSv/h. In their usage calibration point is nothing but the mathematical inverse of Sensitivity.

All GMC counter have 3 so called “calibration points” in their firmware. In Figure 36 you see the calibration points listed as read-out from the firmware of a GMC-500+ counter.

For single tube counter GMC-300 series all 3 points define the same “calibration factor” 0.0065 µSv/h/CPM, equivalent to Sensitivity 154 CPM/(µSv/h).

For single tube counter GMC-600 series all 3 points define the same “calibration factor” 0.00264 µSv/h/CPM, equivalent to Sensitivity 379 CPM/(µSv/h).

The tube used here is a LND7317 pancake tube, and while the tube manufacturer gives a sensitivity of 348 CPM/(µSv/h), GQ props this up to 379! GQ is not explaining why the tube is declared some 10% more sensitive once built into their counter.

For a double tube counter (GMC-500+ series) with 2 different tubes, the first 2 “calibration points” define the “calibration factor” for the 1st tube – the same 0.0065 µSv/h/CPM as for the 300 series, while the 3rd “calibration point” holds the “calibration factor” for the 2nd tube, in Figure 36 seen as 0.194 µSv/h/CPM (which is way off any realistic value, but this is a different matter).

29 See e.g. Reply#4 in http://www.gqelectronicsllc.com/forum/topic.asp?TOPIC_ID=9815

In the GeigerLog configuration file the tube sensitivities are pre-configured to the best values available (see chapter Configuration of GeigerLog on page 45).

Actually, in Figure 36 you can see not 3 but 6 “calibration points”, of which – in this example for a GMC-500 counter – the last 3 points are grayed out. This because the newest GMC counter, like the GMC-800, even have 6 “calibration points”. But, hilariously, all 6 points code the very same value of 154 CPM/(µSv/h) ;-).

AudioCounter Devices

Some Geiger counters generate audio-clicks for each registered radioactive event. For some counters – especially very old ones and modern low-cost varieties – this is the only means of indicating an event. Such counters are fully supported by GeigerLog. How to operate, connect, configure and run such devices is described in the article **GeigerLog-AudioCounter-Support** [30]).

An example for a low-cost, audio-only counter is the **Radiation-v1.1(CAJOE)** counter; see the review [31]).

But even some very advanced counter designs offer only an audio output, like in particular this also reviewed, semiconductor based radioactivity detector **Smart Geiger Pro (SGP-001)** [32]).

The article also contains a comparison of the digital measurement with the audio measurement using a Geiger counter, which can do both simultaneously, like a GMC-300E+. The results fully confirm the validity of the two different methods.

Figure 37 shows an example of a simultaneous digital and audio recording with two different devices, a **GMC-Device (GMC-300E+)** connected digitally, and the **Smart Geiger Pro (SGP-001)** device, connected as an AudioCounter.

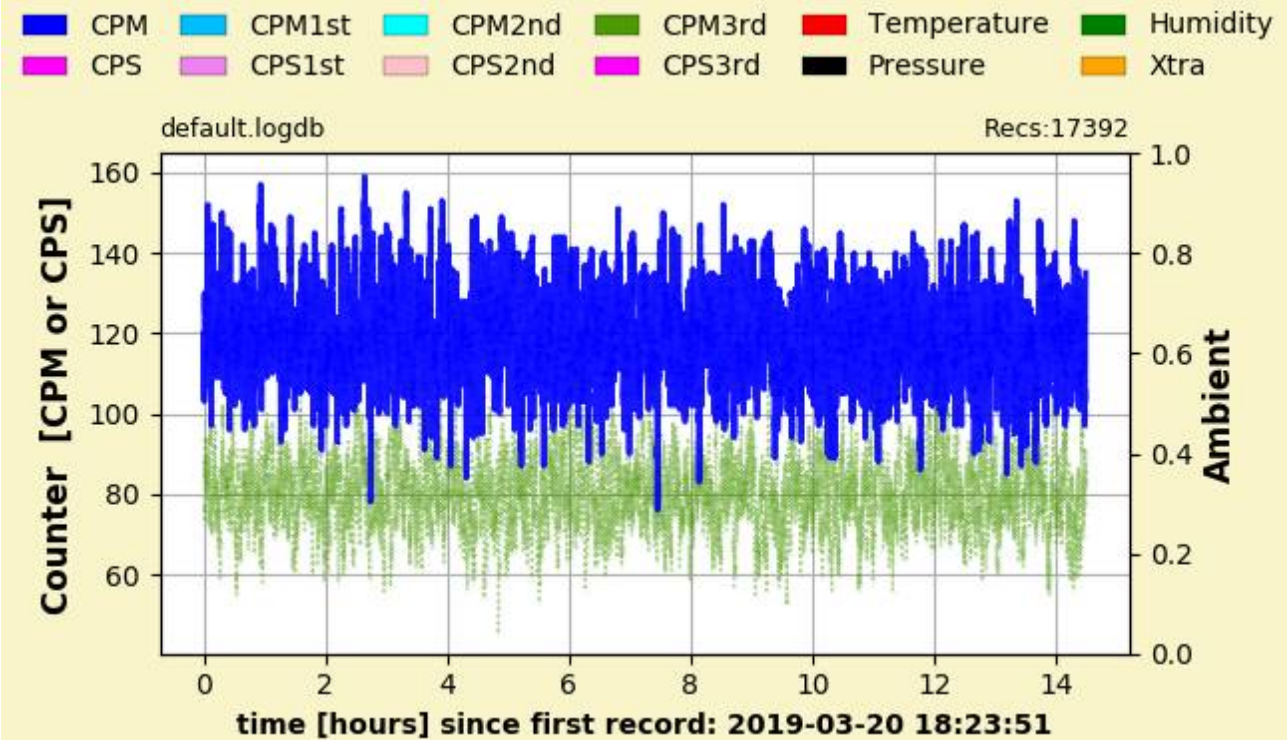


Figure 37: GMC-300E+ (blue) digital recording, and SGP-001 Device (green) audio recording

30 Read article **GeigerLog-AudioCounter-Support-v2.0.pdf** on Sourceforge: https://sourceforge.net/projects/geigerlog/files/Articles/

31 Read article **GeigerLog-Radiation-v1.1(CAJOE)-Support** on Souceforge: https://sourceforge.net/projects/geigerlog/files/Articles/

32 Read article **GeigerLog-Review Smart Geiger Pro (SGP-001)** on Souceforge; https://sourceforge.net/projects/geigerlog/files/Articles/

GeigerLog's AudioCounter on Raspi4 (Buster) and Raspi5 (Bookworm)

Raspi4 can perform using GeigerLog's AudioCounter driver, but it has its limits. That small computer is simply not powerful enough! However, the Raspi5 performed well where the Raspi4 failed!

First, since the Raspi has no **audio input**, you need a USB-Soundcard. Once plugged-in make sure to select the USB-Soundcard as audio device, and **enable Capture** in the sound settings!

I bought this device https://www.amazon.de/gp/product/B07RS11PDD (8.18 Euro) which has this chip: 'ID 1b3f:2008 Generalplus Technology Inc.' Works equally well on Raspi and on Desktop; just plug it in and select it as your audio device, no drivers needed.

Then I connected a GMC300E+ counter simultaneously both via its digital USB-To-Serial output and via its Audio output to the computer.

Raspi4

The computer running GeigerLog was either the Raspi4 or my Desktop. The summary of results is shown in Figure 38:

The upper one shows Raspi4 results: while Raspi4 easily handles GeigerLog with a digital connection to a GMC counter, it is simply overwhelmed with the audio processing.

The lower pic shows the very same setup, but now connected to my Desktop. One can barely see a load on the CPU. The data are identical within statistical scatter. The main reason for differences is that the time base is not and cannot be synchronized. In this particular example the overall average is:

Digital: **CPM=388.48**
Audio: **CPM=388.45** ;-))

Typically the differences are about 1%.

Conclusion: Raspi4 is perfectly fine for Digital, but is overwhelmed when used as AudioCounter!

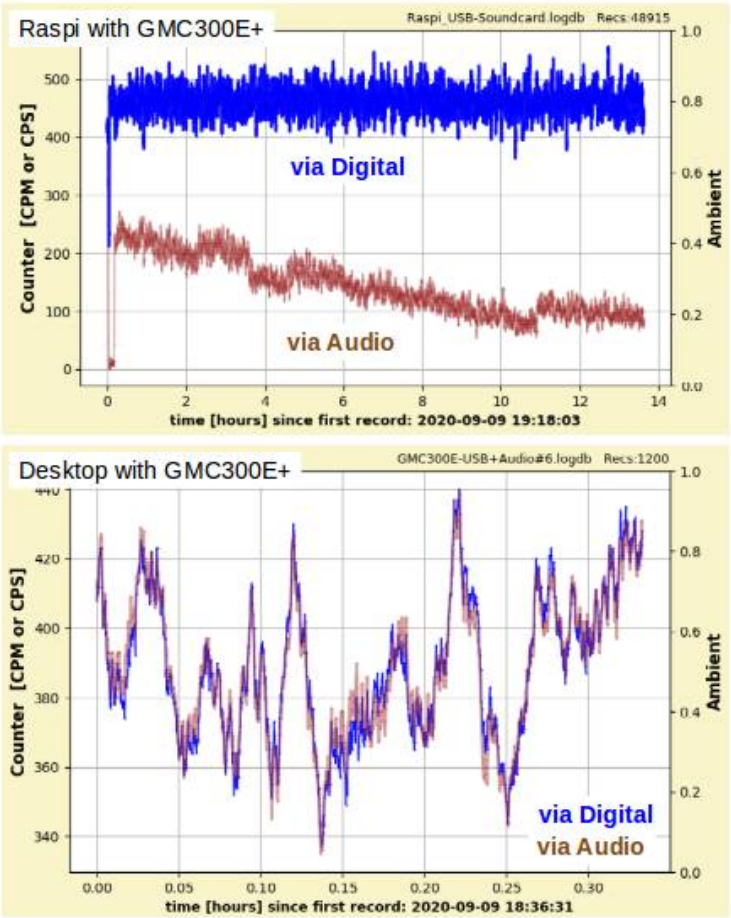


Figure 38: Comparing Raspberry Pi and Desktop Computer

Raspi5

Once the more powerful Raspi5 became available, I repeated the experiment (more details in a forum report [33]):

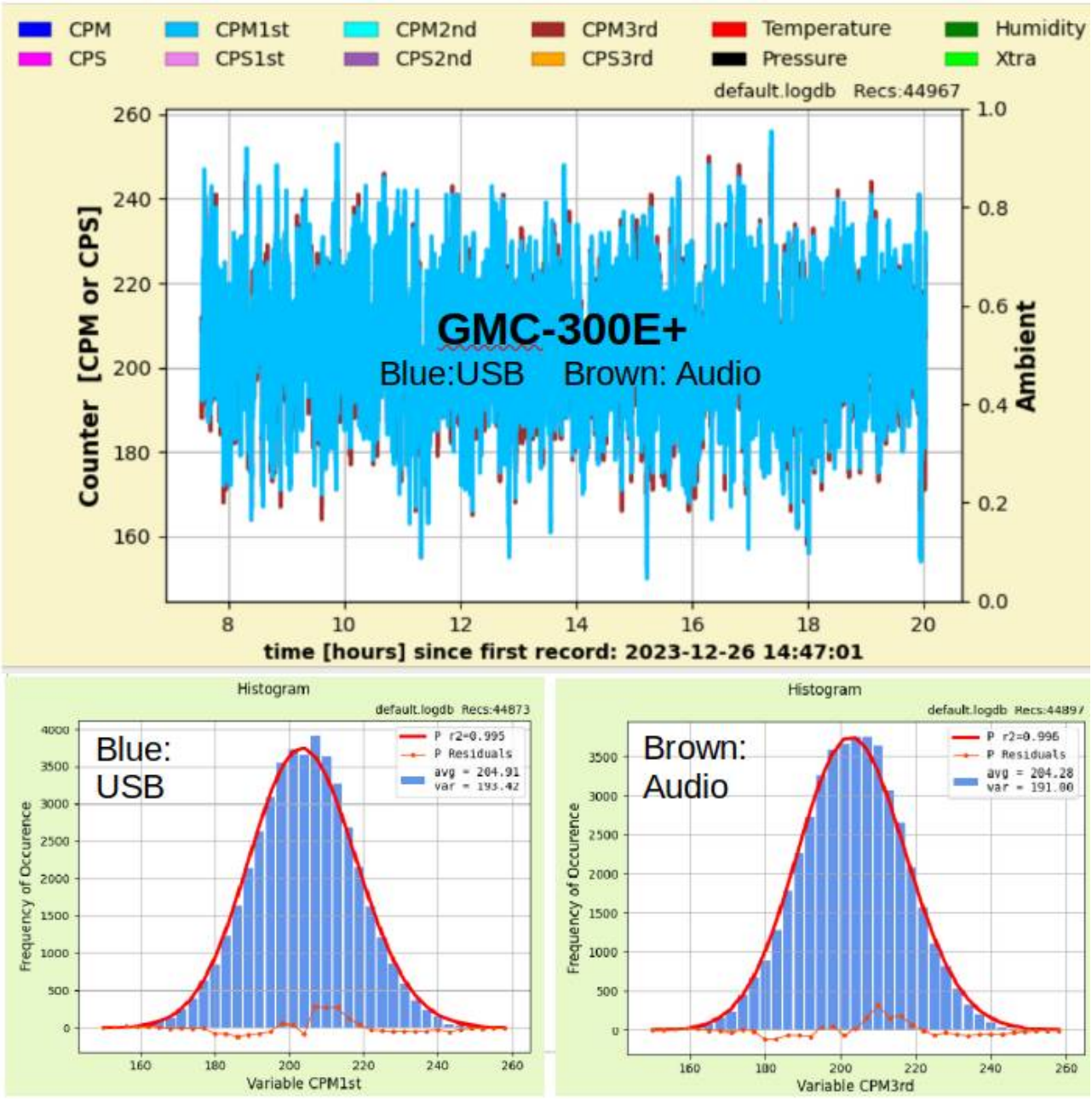


Figure 39: Raspi5 running both digital and audio connected GMC-300E+ counter

The result is obviously significantly different – there is no difference in the observed count rates for one and the same counter GMC-300E+ connected digitally and via the audio port.

Conclusion: Raspi5 is well suited both for Digital Counters and for AudioCounters!

33 https://www.gqelectronicsllc.com/forum/topic.asp?TOPIC_ID=10352

Audio-To-Serial Devices

GQ is offering a **Tool-081** [34]), which is meant to convert a regular counter audio signal to USB data.

This is the exact same technique as used with GeigerLog's **SerialPulse Device**, detailed in chapter SerialPulse Devices on page 113! The only differences are that Tool-081 has a capacitor in series to the RX pin on the UART side of an USB-To-Serial chip for some decoupling of the audio signal, and multiple times the price ;-).

Tool-081 requires audio-pulses of a certain shape and voltage, which – among the line of recent GMC counter – **ONLY** the GMC-500+ delivers, despite GQ's claim to the contrary:

> *It can be used on GMC-060, GMC-080, GMC-100, GMC-200* ***and any other new and old*** *Geiger counter with audio output only.* [34]*).*

As an aside: The related GeigerLog activity followed an initial request of GeigerLog user Bob, which triggered a lengthy discussion, also supported by user Damien, on how to create a device similar to Tool-081, which would support **all** counters whatever their audio signal was. Unfortunately, this discussion failed. If any electronically experienced user is interested to contribute: you find the topic on the discussion site of GeigerLog: **Addition of GQ TOOL-081 Audio to USB data cable as an input** [35]).

However, it could be shown that this tool would work as long as the proper pulses were provided. In Figure 40 the results from a run with the CAJOE counter [36]) and a USB-To-Serial device based on a CH340C chip and a 470nF capacitor [37]) are shown. The data have textbook quality!

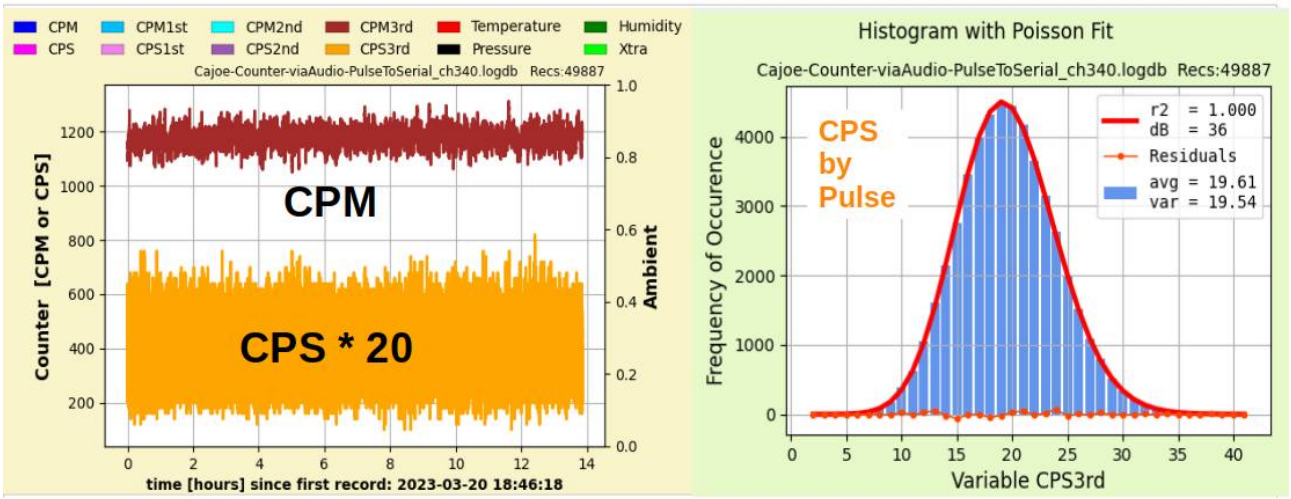


Figure 40: CAJOE Counter Pulses fed into Tool-081-like Audio-To-Pulse converter

But then, given the pulses of a CAJOE counter, the same would have been possible without a capacitor making the Tool-081 obsolete. The **SerialPulse Device** alone would have been completely sufficient!

34 https://www.gqelectronicsllc.com/comersus/store/comersus_viewItem.asp?idProduct=4544

35 https://sourceforge.net/p/geigerlog/discussion/features/thread/9654e9a11c/

36 https://sourceforge.net/projects/geigerlog/files/Articles/GeigerLog-Radiation-v1.1%28CAJOE%29-Support-v1.0.pdf/download

37 https://sourceforge.net/p/geigerlog/discussion/features/thread/9654e9a11c/?page=3#07b1

IoT Devices

IoT – Internet of Things – devices are a wide variety of devices, which are accessible through a network, in particle through the internet [38]).

Such devices include e.g. Geiger counter as the **RadMon Devices,** supported by GeigerLog and described in chapter RadMon Devices on page 76.

And they include sensors of all kinds, in particular those based on **TASMOTA**, an Open Source firmware for microprocessors of the ESP family [39]). At the last count there were some **2802** devices of that kind (as of March 2024) [40])!

GeigerLog supports – apart from the **RadMon** family – two variants of IoT devices:

- A generic client, which sends all 12 possible GeigerLog variables (CPM, CPS, CPM1st, CPS1st, CPM2nd, CPS2nd, CPM3rd, CPS3rd, Temp, Press, Humid, Xtra) as answer to a SEND request.
- A device with a TASMOTA operating system. Implemented here is a switchable AC plug which provides various parameters, like voltage, current, and power [41]).

Look at the DEMO example **`Simul_IoTClient.py`** for the generic client in folder **`gtools`**. You can start this demo and receive valid IoT data, redirected from your desk to a server in California, and back to your desk.

Connecting

There will actually never be a connection between an IoT device and GeigerLog, nor will the two ever talk to each other! That is the norm for IoT devices.

The IoT device will be configured to send its data to an IoT server – in IoT lingo a broker [42]) – into a specific folder, whenever it has data ready. GeigerLog is told the name of this broker and the name of the folder, and connects to the broker and tells him that it wants these data. Then the broker informs GeigerLog when new data are available, and GeigerLog downloads them.

To the user it looks like the two are connected, though technically they aren't.

Example: TASMOTA based Sensors

Published as "**Support for Tasmota based Sensors**" on the GeigerLog site [43]). In a 4-day run I surveyed the mains voltage of my home as shown in Figure 41. A comparison with a better instrument

38 https://en.wikipedia.org/wiki/Internet_of_things

39 https://tasmota.github.io/docs/

40 https://templates.blakadder.com/

41 https://www.amazon.de/gp/product/B0054PSI46

42 A broker will be a server in your local LAN or anywhere on the internet, which runs MQTT software, https://en.wikipedia.org/wiki/MQTT . Tested servers are based on the Eclipse Mosquitto Open Source message broker Mosquitto https://mosquitto.org/ . You can easily install one on your own computer.

43 https://sourceforge.net/p/geigerlog/discussion/features/thread/d10076b63a/

showed that the TASMOTA devices actually had a very good accuracy with respect both to voltage and power!

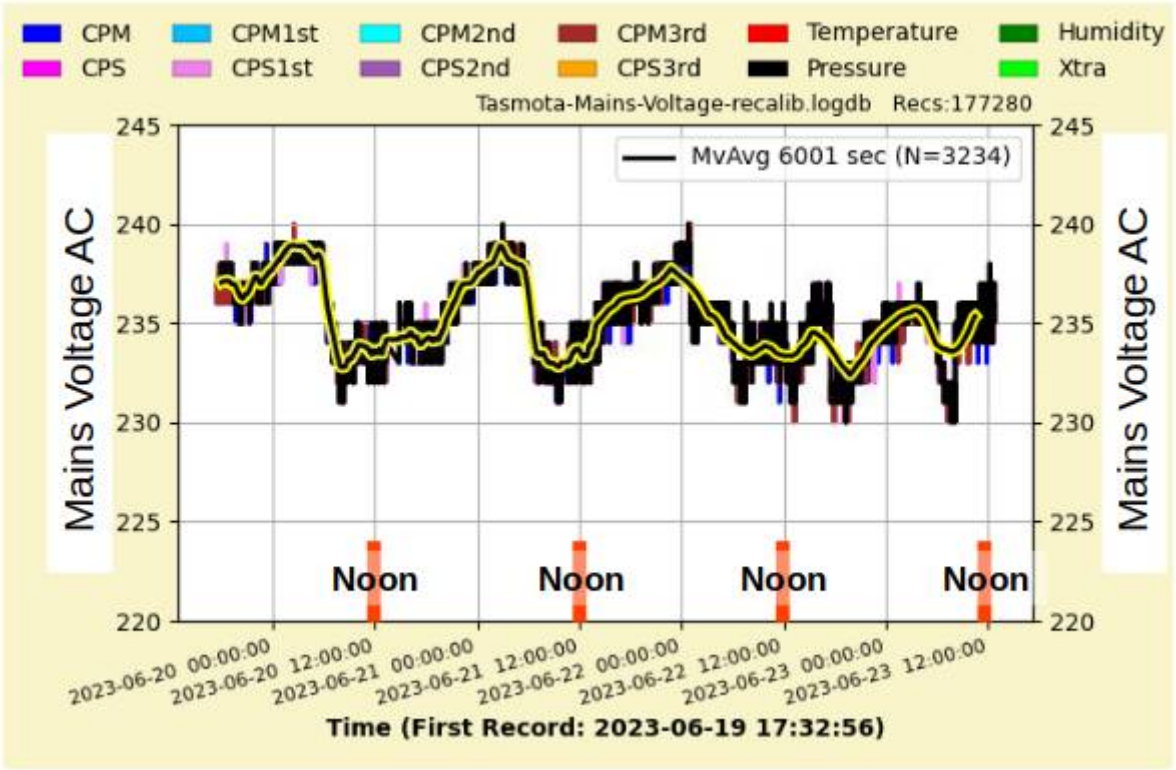


Figure 41: GeigerLog run with TASMOTA based IoT Voltage Meters

Raspberry Pi as an IoT Dataserver

In the GeigerLog folder **`raspi`** find the program **`dataserver`** which allows to connect GMC Geiger counter, various I2C sensors, and Pulse counting to the Raspi, and have Raspi transfer the data to GeigerLog as an IoT device. So, you transfer the data from and to any place on the world.

Note: as an alternative for local use the Raspi can also transfer the data via WiFi.

RadMon Devices

The RadMon devices [44]) are part of a small family of devices, of which the **RadMon+** is used here. They come as Do-It-Yourself kits; assembly is required (see below).

Operating Modes

RadMon devices acts as IoT (Internet of Things [45]) devices, and send their data wirelessly to a special IoT server. GeigerLog reads the data from that server.

You have the option of installing on the RadMon either a Geiger tube, or an environmental sensor BME280 [46]) for temperature, barometric-pressure, and humidity, or both. GeigerLog can handle up to all four variables. In the default setting it is assumed that the RadMon has both a working Geiger tube as well as a working environmental sensor.

The RadMon device needs to be software configured at the device itself.

Connecting

This is described int chapter **Connecting** for IoT Devices on page 74.

Logging

As explained above, GeigerLog cannot ask the RadMon for new data, so you must configure the RadMon and GeigerLog independently.

First the RadMon is configured, e.g. by using a smartphone. I describe my configuration:

My RadMon+ is set to collect counts from the Geiger tube for 60 seconds to determine a CPM value. Then the RadMon+ reads the data for temperature, barometric-pressure, and humidity from its BOSCH BME280 sensor. All 4 values are then sent through my wireless home network to my router and then to a broker server located in North America [47]). The total of reading the 3 environmental variables, and processing and shipping all 4 variables takes an extra time of about 7 seconds. Then the RadMon+ starts a new cycle.

GeigerLog is configured to connect to the same broker, is told what data to expect and where to find them on the server, but otherwise knows nothing about the RadMon+ device. The two sit only a few meters apart, but communicate via a 20000 km round trip of some typically 120 ms duration. A true variant of remote sensing ;-).

Of course, both RadMon+ and GeigerLog must have WiFi/network and internet access.

---

44 DIYGeigerCounter https://sites.google.com/site/diygeigercounter/

45 https://en.wikipedia.org/wiki/Internet_of_things

46 BOSCH BME280

47 Initially I used the server **iot.eclipse.org**, but it seems to have been switched off. More recently I used: **broker.hivemq.com**. However, many more such servers are publicly available for testing purposes: https://github.com/mqtt/mqtt.github.io/wiki/public_brokers

Any GeigerLog user can configure his copy of GeigerLog to access my own RadMon+ device by activating it in the configuration file [48]). Upon establishing a connection, the device should be available. However, I cannot guarantee that my RadMon+ will be always on, but there is a good chance for it.

Assembly of the Device

The RadMon+ devices come as Do-It-Yourself kits; all parts – except the tube, power supply, and a case – are delivered, but you have to solder it yourself. Some basic skill in soldering is needed, but it is not overly difficult as there are no tiny SMD parts. You definitively want to have the manual [49]) ready when you do the assembly!

Configuration

After assembly, the RadMon+ needs to be configured as explained in its manual. I found it easier to do this using my smartphone than my computer.

You have to bring the RadMon+ into its configuration mode, which confuses a little bit. From the manual I have copied the starting sequence, and inserted here as Figure 42. For the rest you can follow the instructions on the smartphone. When done, press the RST button on the RadMon+.

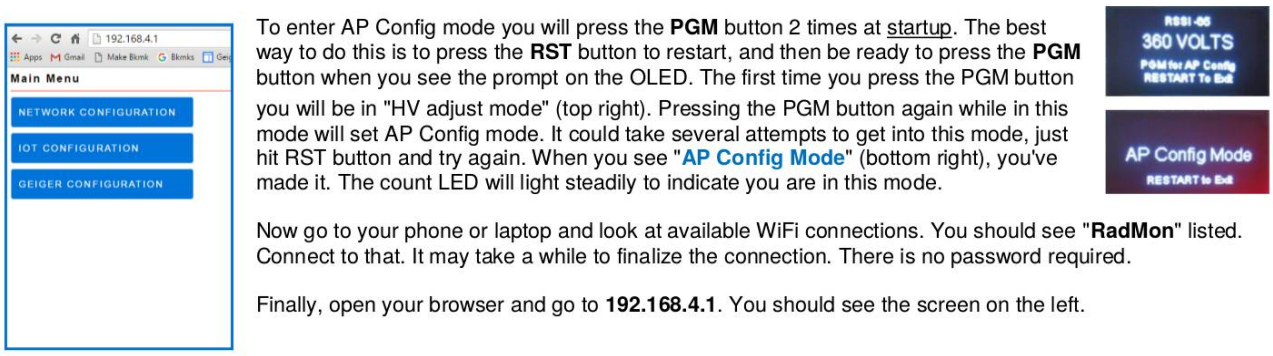


Figure 42: Activating RadMon Configuration on a Smartphone

48 Simply set: RMActivation = yes in file `geigerlog.cfg`.

49 https://www.dropbox.com/s/ypmfjw97b8qlhs1/GK%20Radmon%20Build%20and%20User%20Guide%20v2.1.pdf?dl=1

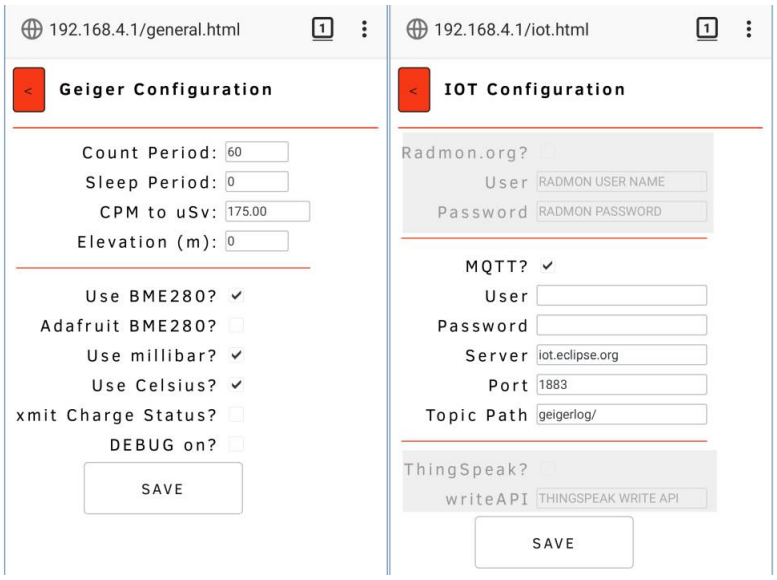


Figure 43: RadMon+ Configuration pages

The first page needs info on your own wireless LAN, so that RadMon+ can connect to it. The next two pages are shown in Figure 43 as screen shots from my smartphone:

The left page shows my current setting for the hardware, the right page for the IoT configuration; only the MQTT part is relevant. Note that the 'Topic Path' is 'geigerlog/' – ending with a slash '/'! The same must be entered into GeigerLog's configuration file (see below) as 'RMServerFolder = geigerlog/'!

Figure 44 shows the last few weeks of another Long-Term – some 9 **months** – recording using the RadMon+ device with its sensor BOSCH BME280 (Temperature, Pressure, Humidity) and a SBM20 Geiger tube. The pressure is plotted as value[hPa] – 1000, Temperature in °C, Humidity as %relative, the Geiger data are blue scatter in the background. The device is located outside in a weather-proof housing. The CPM average with a SBM20 tube is 20.1 CPM.

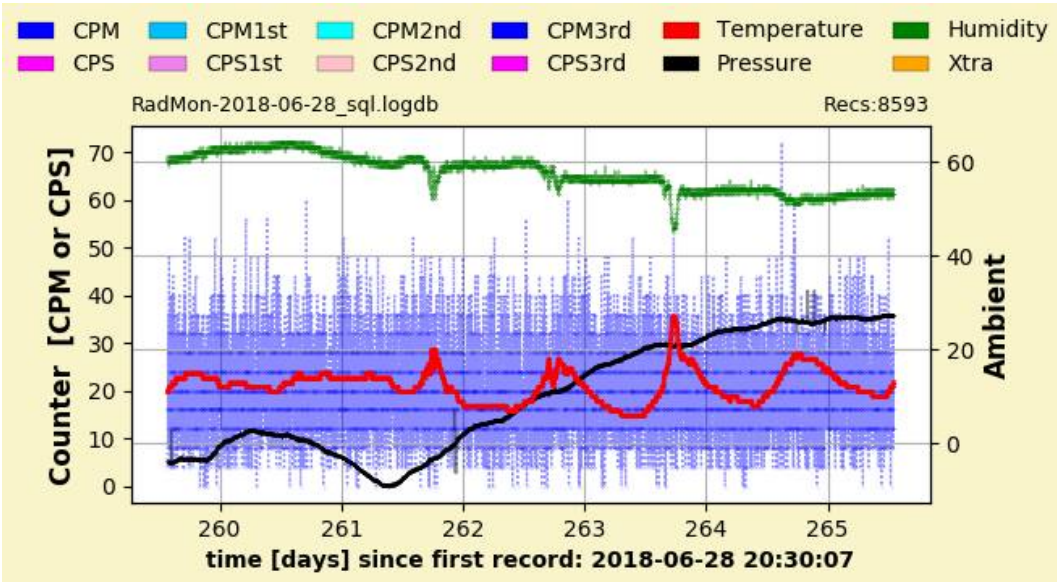


Figure 44: Very-Long-Term Recording from an Outside RadMon+Weather Station

RadPro Devices

RadPro is the name of an open-source firmware, by which various low to very low cost Geiger counters can be converted to rather intelligent devices, which allow **Logging** with GeigerLog and even on-board **History** saving and later downloading with GeigerLog!

This is a very significant achievement by developer “Gissio” [50]). The currently supported devices already form a nice list: FS2011, YT-203B, Bosean FS-600, Bosean FS-1000 and FNIRSI GC-01. The base for the use of the RadPro firmware is that the counters use a microprocessor of the STM32 or compatble family.

The installation of the firmware is not straight forward due to variations within devices – despite having the same name – and may be a bit demanding! It is, however, quite well explained in the documents [51]).

Thanks to the work of Gissio this firmware is now supported by GeigerLog. I have been able to verify operation with the **FNIRSI GC-01** device [52]). This is almost a look-alike to GQ’s GMC-800 counter [53]), but at nearly half the cost! It has a Geiger tube **J321**. Others apparently have tested this successfully with a **BOSEAN FS-600** device [54]).

Figure 45: FNIRSI GC-01

Connecting to GeigerLog

After installing the firmware switch the counter to “USB Mode”, and **ONLY THEN** connect the counter to your computer with the USB cable!

For as long as you want to communicate with the counter you must keep it at “USB Mode”! You **cannot** watch the display of the counter!

The firmware does have some limitations. To get meaningful results operate GeigerLog with a **1 sec LogCycle**, and set the counter’s “Data Logging” to “**Every minute**”.

A more in depth discussion can be found on the net [55]) [56]).

The History download can be done only when not logging; the download takes a fast 2+ sec for some 2000 … 3000 records (about 1.5 days), apparently depending on count rate.

50 https://github.com/Gissio/radpro

51 https://github.com/Gissio/radpro/tree/main/docs

52 https://de.aliexpress.com/item/1005004805929438.html

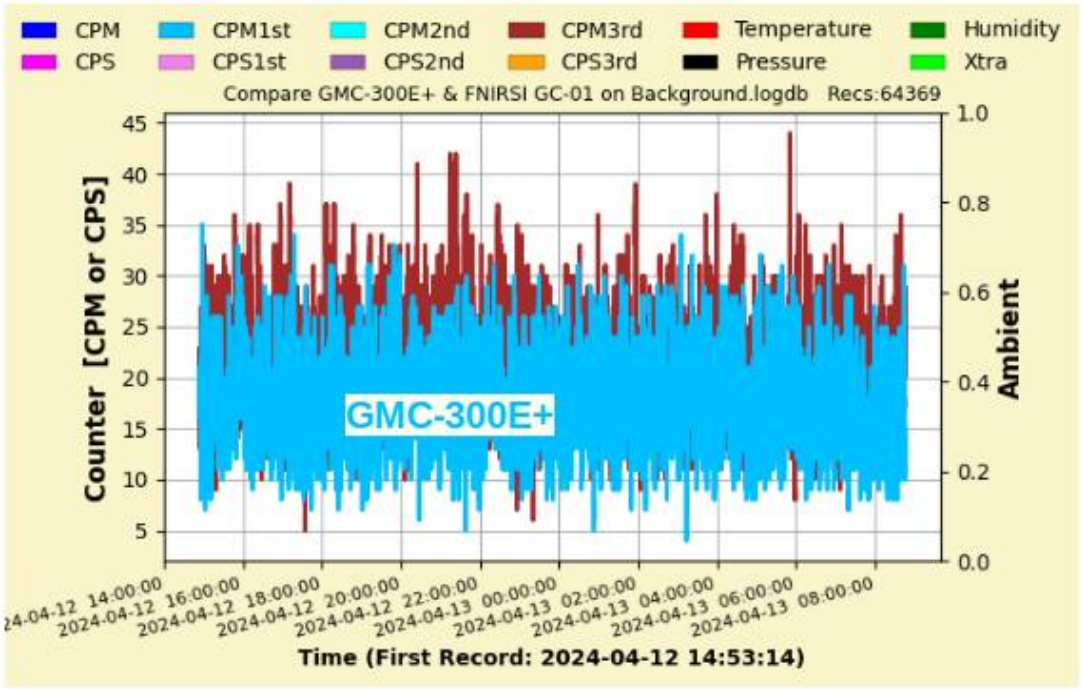
53 https://www.amazon.de/Nuklearstrahlungsdetektor-GMC-800-US-Nationalstandard-Datenspeicherung-Multifunktionsger%C3%A4t/dp/B0CKM8NML7

54 https://www.geigerzaehlerforum.de/index.php/topic,1068.msg29621.html#msg29621

55 https://sourceforge.net/p/geigerlog/discussion/features/thread/46ed425081/?limit=25#934e

56 https://github.com/Gissio/radpro/discussions/50

This FNIRSI GC-01 counter and a GMC-300E+ counter using were placed together in a dark box (to avoid light influence) and background was measured. The result is shown in Figure 46. All seems to be working well; the Poisson checks come out very well. The slightly different CPM averages could both be correct, given the different Geiger tubes and different cases – a calibration does not exist for either of the counters!



Variable [Unit]	Avg ±StdDev	±SDev%	Variance	Min ... Max
CPM1st : [CPM]	17.851 ±4.1671	±23.3%	17.365	4 ... 35
CPM3rd : [CPM]	21.226 ±4.7609	±22.4%	22.666	5 ... 44

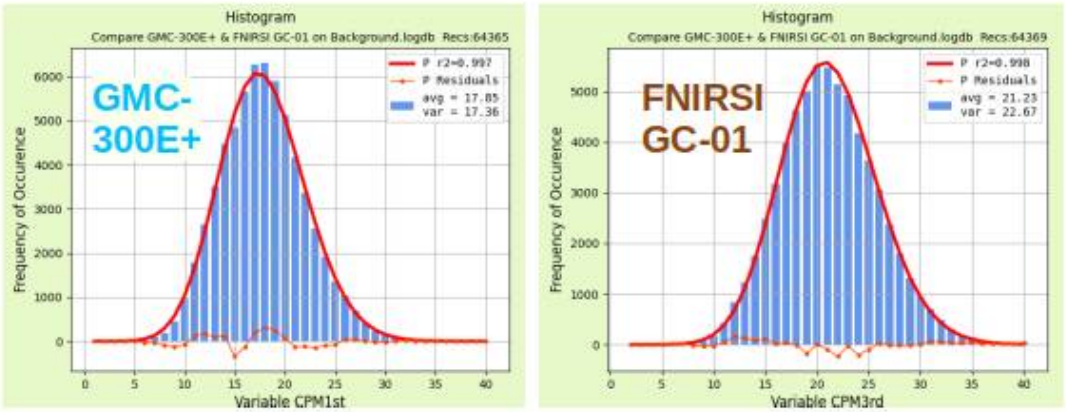


Figure 46: Geiger Counters FNIRSI GC-01 and GMC-300E+ counter, measuring background.

Gamma-Scout Devices

The company Gamma-Scout (https://www.gamma-scout.com/en/) offers these four Geiger counter models: ***Standard***, ***Alert***, ***Rechargeable,*** *and* ***Online*** [57]). From left-to-right the amazon price goes up from about 380€ to 525€ (Feb 2021).

Some hardware and firmware choices are unexpected, so will be explained below in more detail. In brief:

- All have the same type of tube, which can detect alpha, beta, and gamma radiation
- All counters allow to store a history on board, which can be downloaded to a computer
- Logging is possible **only** with the 'Online' model, and only with a cable connection
- There is no WiFi on any of these models

GeigerLog supports all models and all options, with abilities beyond what the included manufacturer's software can do!

Hardware and Firmware

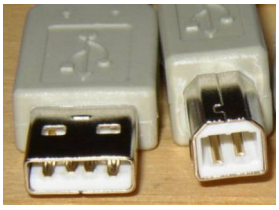
The tube

The tube of all Gamma-Scouts is an LND 712 [58]), a small tube roughly the size of a kid's thumb (49.2x15.1mm), with a sensitivity of 108 CPM/(µSv/h) for Co60, as rated by the manufacturer.

The advantage of this tube is that it has a Mica window, so it can detect alpha-particles, but the window is small with a diameter of only 9.1mm!

The counters have a convenient lever to move different shields in front of the tube's window which leaves it either completely open, blocks only alphas, or also blocks betas.

The USB-Port



The connection between computer and counter is made with an USB cable with a Type A plug on one side, and a Type B plug (the squarely one) on the other side. But electrically the connection is only a serial connection, similar to what the GMC and I2C devices use. The chip doing the conversion between USB and Serial signals is an FTDI chip (Vendor=0x0403, Product=0xd678). This chip supports USB2.0, but only in its slow mode of 12 MBit/s, which is not faster than USB1.0. But as the serial connection speed even at 460 800 baud (=0.46 MBit/s) is only a fraction of that, the chip is fast enough.

And while the serial speed is fast for a serial connection, only the Online model is that fast, the other 3 models offer only rather ancient 9600 baud!

57 https://www.gamma-scout.com/en/

58 https://www.lndinc.com/products/geiger-mueller-tubes/712/

To make matters worse, the firmware uses a 7-bit transfer mode, i.e. a single byte of 8 bit is split into two ASCII-coded bytes of 7 bit. Thus the effective transfer rate is further reduced two-fold to now 4800 baud and 230 400 baud, resp.!

Logging

The “Gamma-Scout Online” is the latest model and the only one of the four to offer an option for logging. Basically, the counter is set into a mode where it sends out count information at regular intervals. This is the only existing mode for logging.

The interval can be set to 2, 10, 30, 60, 120, and 300 seconds, so the returned data mean Counts-Per-2seconds, Counts-Per-10seconds, …, Counts-Per-300seconds. GeigerLog can handle the settings and collect, record, and display the data. As long as the interval is 60 sec or less, GeigerLog also adds the CPM data by summing up the last 60 seconds worth of counts. This CPM is true Poissonian! For any longer interval there is obviously no Poissonian way back to CPM, but you can always scale the data (e.g. divide the 300sec data by 5) by using GeigerLog’s Scaling option (see ValueScaling and GraphScaling on page 37). Though then this no longer forms a Poisson distribution.

This Gamma-Scout “Online” mode is the same as the “heartbeat” mode found in GMC counters, which, however, is outdated and not used by GeigerLog, as the GMC’s send-on-demand mode is easier to handle and much more useful.

History Memory

All 4 devices have an internal history memory of only 64k Bytes, which makes the slow transfer speed a bit more bearable ;-). All counts are stored as a 2 bytes floating point value as described in the ‘Gamma-Scout Communication Interface’ article [59]). It uses an 11-bit mantissa, and a 5-bit exponent.

The disadvantage is that 11 bits can hold values only up to 2048, so everything greater will loose precision, and while everything smaller than 255 could be coded with just a single byte, it now needs the additional, empty second byte. The advantage is that you always know how many bytes are needed for storage independent of count rate, and do not have the overhead of distinguishing between 1,2,3, and perhaps 4 byte values! This latter problem is a significant one with the algorithm used in the GMC counters (see Appendix E – GMC Device: Internal Memory, Storage Format and Parsing Strategy on page 158).

The counts will be collected over a selectable interval ranging from 10 seconds to 1 week. Note that anything faster is not possible, in particular CPS (Counts Per 1second) can NOT be set! Strangely, the 2 second interval, while available for logging, is not available for the history interval setting!

So the memory can hold some r00 values, plus some house keeping information like time stamps, intervals, alarm reports, etc. Thus with a 10 sec interval the memory is filled after about 3 days. With a 1 week interval the memory is filled after 615 years! I wonder how enticing this is for people to purchase this counter? And I wonder what they say when they have to replace the battery “after a few years” and all data in the memory are lost?

59 https://www.gamma-scout.com/wp-content/uploads/Gamma-Scout_Communication_Interface_V1.7.txt

Operating the Gamma-Scout Counter

Installation

Make sure that Gamma-Scout devices are activated in the configuration file `geigerlog.cfg` in the section `GammaScoutDevice.`

There is nothing to install, unless you computer does not have a driver for the USB-To-Serial chip. If not, you need to install a driver for the chip **FTDI (Vendor=0x0403, Product=0xd678)**.

Connecting

After the USB cable is connecting counter and computer, just press the Connect button, or use menu: **Device → Connect Devices**. The connection should be plug-and-play.

If no connection is made, then chances are something else is wrong; verify cable and plugs.

Figure 47: Set Serial Port of Device 'GammaScout'

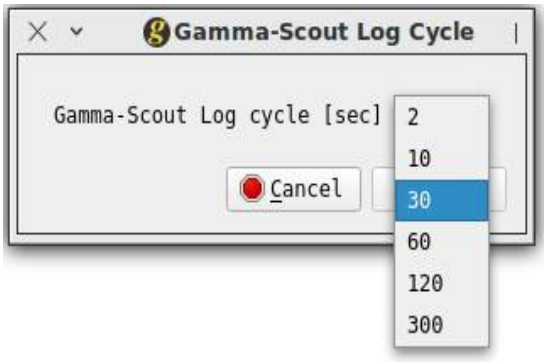
You can try to set the port manually from menu: **Device → GammaScout Series → Set Serial Port …**, see Figure 47. From the drop-down box select any port to try. Setting "auto" might still be your best bet.

If you do get a connection only with a specific port, you can pre-configure it in GeigerLog's configuration file `geigerlog.cfg` in section [GammaScoutDevice] as `GSusbport = <your port>`.

Logging

The Gamma-Scout Online must be explicitly set into its Online mode for logging. Otherwise only empty values will be collected. From the menu select: **Device → Gamma-Scout Series → Set to Online Mode**. This dialogue pops up and allows you to select the interval for logging.

The GeigerLog log cycle should be set to the same setting as the Gamma Scout. It can be set faster, but not slower. Then start logging.



History Download

From the menu choose **History → Gamma Scout Series → Get History from Device**, and the history will be downloaded and saved into a database.

If you have a *.dat file created with Gamma-Scout software (which contains a memory dump as Gamma-Scout calls a History Download) you can use **History → Gamma Scout Series → Get History from Gamma-Scout Dat File** to load this file into a GeigerLog database, just as if it were downloaded from a Gamma-Scout device.

Data Interpretation

The Gamma-Scout devices do produce only “Counts-Per-Interval” data, with intervals ranging from a minimum of 10 sec to a maximum of 1 week. Thus there is never a recording of true CPS, and true CPM results only when the interval is set at 1 minute!

By default GeigerLog maps the CPInterval counts to variable CPS3rd. If the interval is 60 sec or less, then GeigerLog creates a true CPM column mapped to variable CPM3rd by summing up the latest 60 seconds worth of counts. The interval itself is mapped to variable X.

Next lines are an excerpt of logging data done with a Gamma-Scout interval of 2 seconds. So column CPS3rd contains “Counts-Per-2seconds”, while CPM3rd contains true CPM. Of course, it takes 30 records of 2 sec each before a full 1 min of data can be summed up for the 1st CPM value!

```
#>  Index,            DateTime,  CPM3rd,  CPS3rd,       X
#  HEADER, 2021-02-20 13:15:47, LogFile newly created as 'default.logdb'
# DEVICES, 2021-02-20 13:15:47, Connected: Gamma-Scout : CPM3rd CPS3rd X
# LOGGING, 2021-02-20 13:15:47, Start: Cycle: 2 sec
        1, 2021-02-20 13:15:49,        ,     0.0,     2.0
        2, 2021-02-20 13:15:51,        ,     0.0,     2.0
        3, 2021-02-20 13:15:53,        ,     0.0,     2.0
        ...
       28, 2021-02-20 13:16:43,        ,     0.0,     2.0
       29, 2021-02-20 13:16:45,        ,     0.0,     2.0
       30, 2021-02-20 13:16:47,    14.0,     1.0,     2.0
       31, 2021-02-20 13:16:49,    14.0,     0.0,     2.0
#>  Index,            DateTime,  CPM3rd,  CPS3rd,       X
```

Both columns CPM3rd and CPS3rd can be Poisson tested, and should give a proper distribution. The above data as a graph are shown in Figure 48.

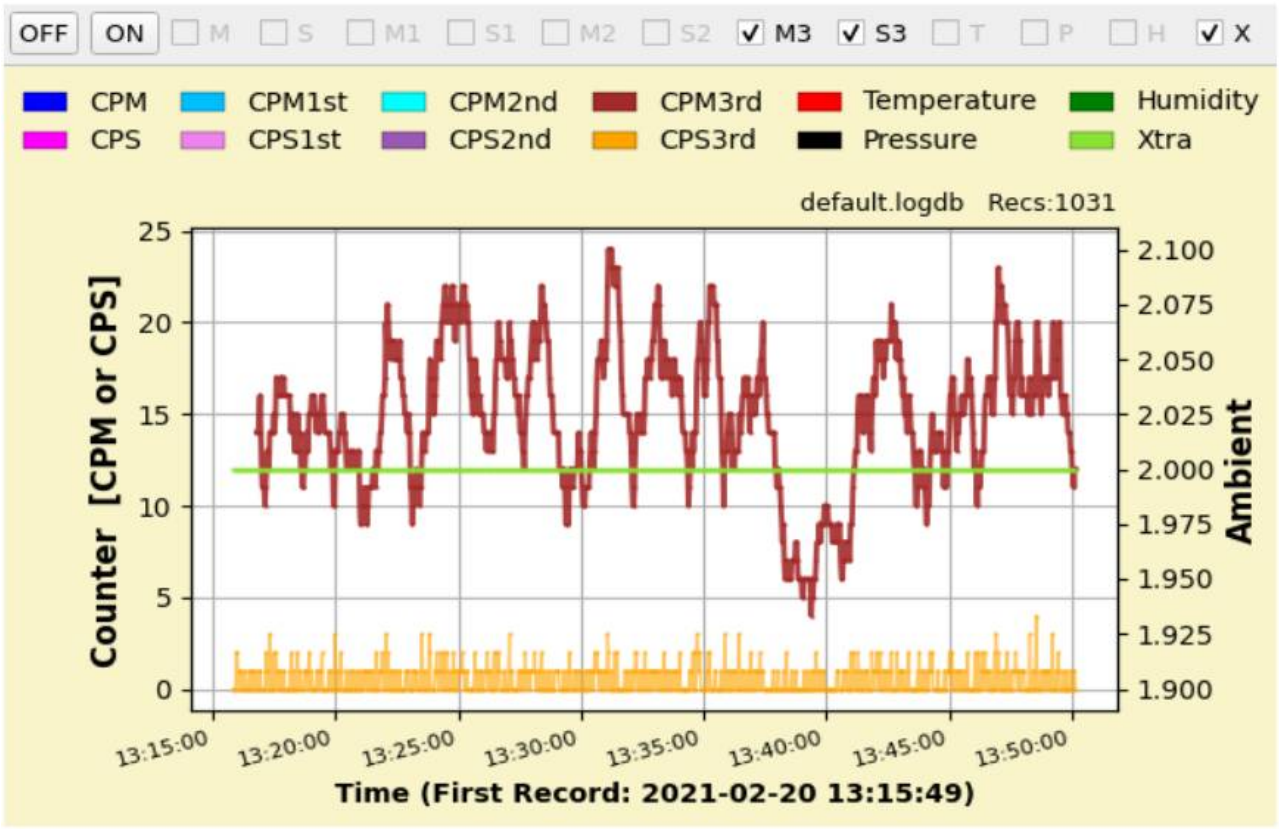


Figure 48: Gamma-Scout Online model used for logging with a 2 sec interval

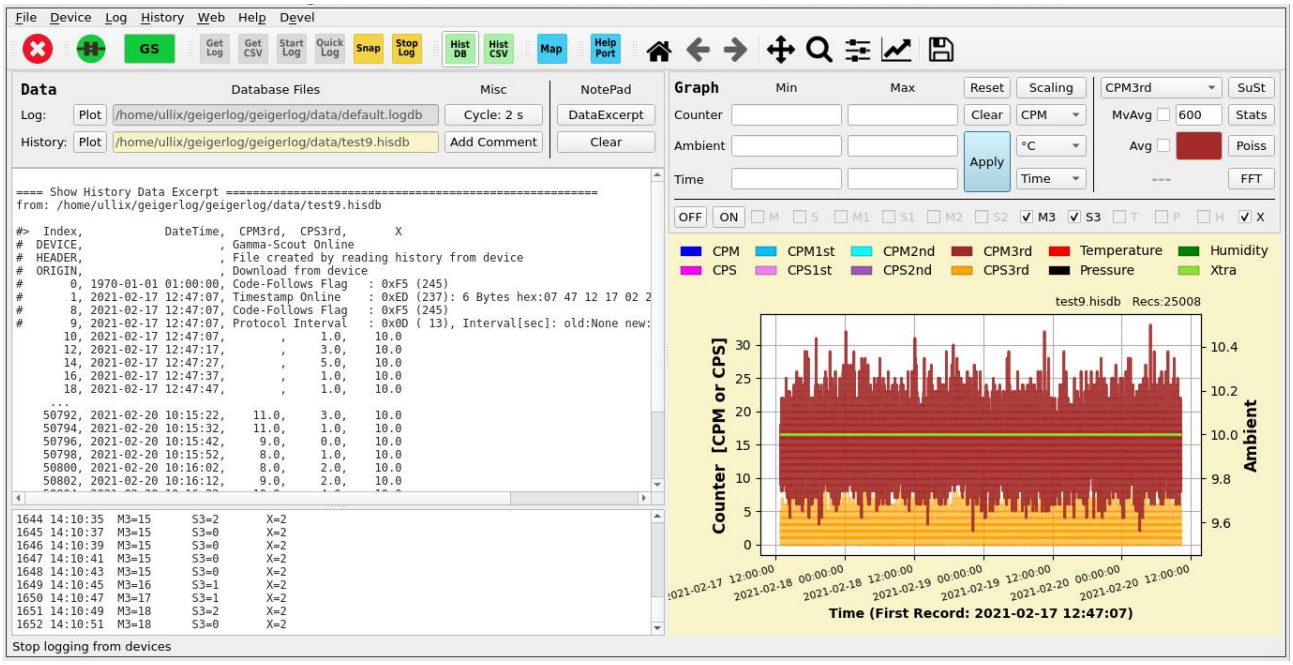
Orange: the "Counts-Per-2sec" data as delivered by the counter, brown: the CPM data calculated by GeigerLog, green: the interval of 2 sec (Ambient scale)

Counts vs. µSv/h

The conversion between a dose rate in Counts and in µSv/h is based on either CPM or CPS. Since the Gamma-Scout devices do not offer a 1 second interval, all CPS data shown as µSv/h will be too high by a factor equal to interval! CPM data calculated by GeigerLog will convert properly.

A 3-day history download from a Gamma-Scout Online device, recorded with a 10 sec interval is shown in Figure 49.

Making the Poisson tests on both data sets from Figure 49 gives the results shown in Figure 50. Both data sets are fully Poissonian.




```
==== Show History Data Excerpt ============================================================
from: /home/ullix/geigerlog/geigerlog/data/test9.hisdb

#>  Index,            DateTime,  CPM3rd,  CPS3rd,       X
#  DEVICE,                    , Gamma-Scout Online
#  HEADER,                    , File created by reading history from device
#  ORIGIN,                    , Download from device
#       0, 1970-01-01 01:00:00, Code-Follows Flag   : 0xF5 (245)
#       1, 2021-02-17 12:47:07, Timestamp Online    : 0xED (237): 6 Bytes hex:07 47 12 17 02 2
#       8, 2021-02-17 12:47:07, Code-Follows Flag   : 0xF5 (245)
#       9, 2021-02-17 12:47:07, Protocol Interval   : 0x0D ( 13), Interval[sec]: old:None new:
       10, 2021-02-17 12:47:07,         ,     1.0,    10.0
       12, 2021-02-17 12:47:17,         ,     3.0,    10.0
       14, 2021-02-17 12:47:27,         ,     5.0,    10.0
       16, 2021-02-17 12:47:37,         ,     1.0,    10.0
       18, 2021-02-17 12:47:47,         ,     1.0,    10.0
    ...
    50792, 2021-02-20 10:15:22,    11.0,     3.0,    10.0
    50794, 2021-02-20 10:15:32,    11.0,     1.0,    10.0
    50796, 2021-02-20 10:15:42,     9.0,     0.0,    10.0
    50798, 2021-02-20 10:15:52,     8.0,     1.0,    10.0
    50800, 2021-02-20 10:16:02,     8.0,     2.0,    10.0
    50802, 2021-02-20 10:16:12,     9.0,     2.0,    10.0

1644 14:10:35  M3=15       S3=2       X=2
1645 14:10:37  M3=15       S3=0       X=2
1646 14:10:39  M3=15       S3=0       X=2
1647 14:10:41  M3=15       S3=0       X=2
1648 14:10:43  M3=15       S3=0       X=2
1649 14:10:45  M3=16       S3=1       X=2
1650 14:10:47  M3=17       S3=1       X=2
1651 14:10:49  M3=18       S3=2       X=2
1652 14:10:51  M3=18       S3=0       X=2
```

Figure 49: A 3-day history download from a Gamma-Scout Online device

The recording interval was 10 sec. CP10sec: orange, CPM:brown, X (Interval):green

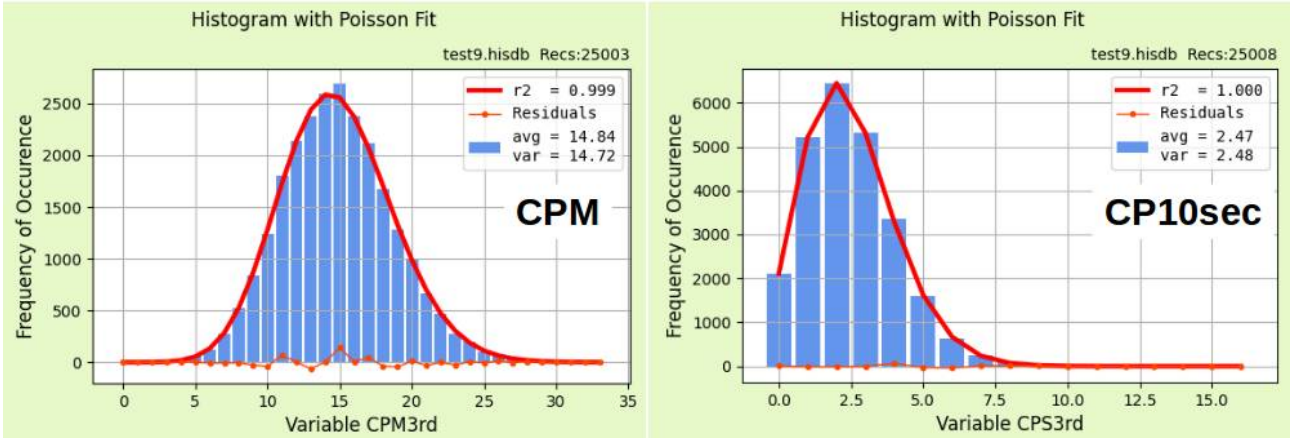


Figure 50: Poisson Tests on the data shown in above figure

I2C Devices

GeigerLog can handle I2C based devices, like sensors.

Presently it supports these I2C sensors:

- **LM75** temperature
- **BME280** temperature, barometric-pressure, humidity
- **TSL2591** light sensor visible and infrared light
- **BH1750 (=GY-30 [60])**) light sensor visible light
- **SCD30** CO2 by NDIR, temperature, humidity
- **SCD41** CO2 by Photoacoustic, temperature, humidity
- **GDK101** Geiger counter based on PIN Diodes

Such sensors require an I2C connection, which is not available on today's computers [61]). An exception to the rule is the Raspi, which does natively support I2C through its GPIO pins, see chapter Raspi Devices on page 106. A Raspi, run as WiFiServer, is an alternative to the use of dongles, see chapter Raspi Devices – Raspi as Satellite to GeigerLog on page 111.

However, USB based hardware, a so called **USB dongle**, is available for all computers, which provides this type of connection.

I have selected 4 different USB dongles and evaluated their performance with the I2C devices listed above within GeigerLog. The dongles are:

- **USB-ISS** Best
- **ELV USB-I2C** Ok
- **IOW24-DG** Poor
- **FT232H** Very poor (removed from GeigerLog)

The result is published in article **"Review of USB-To-I2C Dongles as used by GeigerLog"** available in my Article folder [62]). A follow-up assessment of devices with respect to measurement of CO2 is published in article **"Measuring CO2 with SCD30, SCD41, and MiniMon"** in the same article folder.

For the Geiger counter **GDK-101** based on a PIN Diode find a review "**GeigerLog-Review PIN Diode Geiger Counters"** in the same Article folder.

While GeigerLog can support all dongles and devices, the clear dongle champion **USB-ISS** is chosen for future work. Its advantages:

- worked with all I2C devices tested
- the fastest of all the dongles by a good margin (up to 20 fold faster!)
- cross-platform compatible (Linux, Windows, Mac, ...)

60 Typically GY-30 is the breakout-module using the chip BH 1750 http://wiki.sunfounder.cc/index.php?title=GY-30_Digital_Light_Intensity_Measuring_Module

61 Actually, it is present, but not for the regular user: enter **i2cdetect -l** (-l like list) in a terminal of a Linux computer and see the list of I2C devices already present in your computer.

62 https://sourceforge.net/projects/geigerlog/files/Articles/

- no driver installation by user (installed by default; except perhaps in old Windows systems)
- auto-configurable in GeigerLog
- similar or lower price than the competitors
- other technical advantages (supports 3.3V, bus speed up to 1MHz, easy to program)

To summarize the reviews: The 2nd fasted dongle is the ELV dongle, also cross-platform usable without user installations. The 3rd fastest is the IOW dongle, so far working only on Linux, and requiring a driver installation [63]). The 4th one is the FTD; it was removed from GeigerLog, because it failed to support one I2C device, it was difficult to install, and it was awfully slow.

Connecting

GeigerLog can auto-detect the presence of an USB-to-I2C dongle USB-ISS and make the correct settings, thus providing for **plug-and-play** use of **USB-ISS** dongles! [64]). GeigerLog will auto-configure it for best performance.

If this fails then your computer may lack an installed driver. Look for installation files for a "Microchip Technology, Inc., Devantech USB-ISS" chip (idVendor=0x04d8, idProduct=0xffee) and install.

If connections still fail, then chances are something else is wrong; verify cable and plugs.

You can try to set the port manually from menu: **Device → I2C Series → Set Serial Port …**, see Figure 51. From the drop-down box select any port to try.

Setting "auto" might still be your best bet.

Figure 51: Set Serial Port of Device 'I2C'

If you do get a connection only with a specific port, which is not auto-recognized, you can pre-configure it in GeigerLog's configuration file `geigerlog.cfg` in section [I2C] as `I2Cusbport = <your port>`.

Then you need to wire-connect your I2C devices to the dongle, configure them in the GeigerLog configuration file `geigerlog.cfg` in section **# I2C SENSORS and I2C Variables** and start GeigerLog.

63 Follow the install instructions for libiowkit-1.X https://www.codemercs.com/downloads/iowarrior/IO-Warrior_SDK_linux.zip

64 This plug-and-play will also work for the other dongles, however, in the case of IOW and FTD you'll first have to install the required, non-standard driver!

Example Setup

From left-to-right you see the MiniMon (with its backside removed), the SCD41, the BME280, the SCD30, and the USB-ISS dongle, with its USB-Type-B receptacle pointing upwards. The inset on the bottom-left serves as a size comparison for the SCD41 and BME280. The SCD41 is small, but the BME280 is tiny!

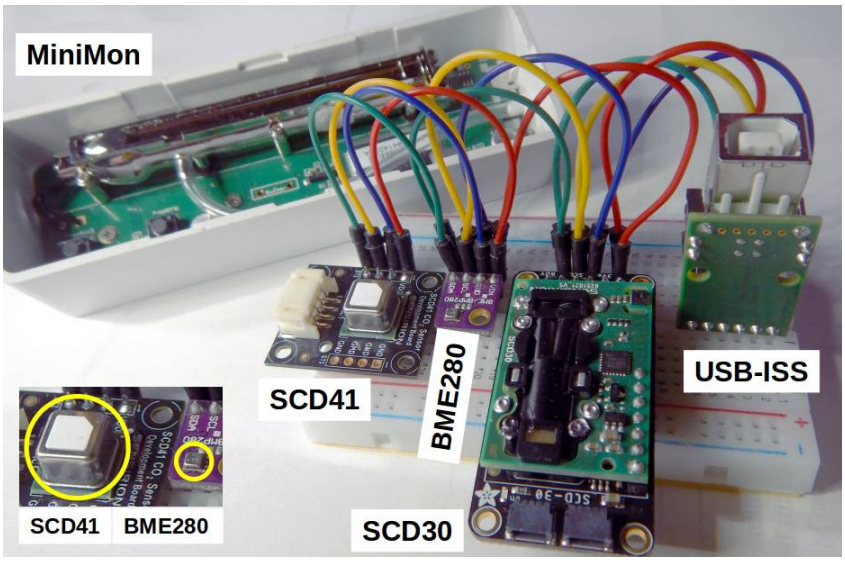


Figure 52: The assembly of the devices

Results from a 30 h run of this assembly with GeigerLog is shown in Figure 53. The bottom 3 curves are CO2 data in ppm, the top curves are Temperature data (values scaled by 50x) in °C. A detailed discussion is in the referred-to articles [62]).

These I2C devices are fast, each responding within 2 … 6 milliseconds when on the USB-ISS!

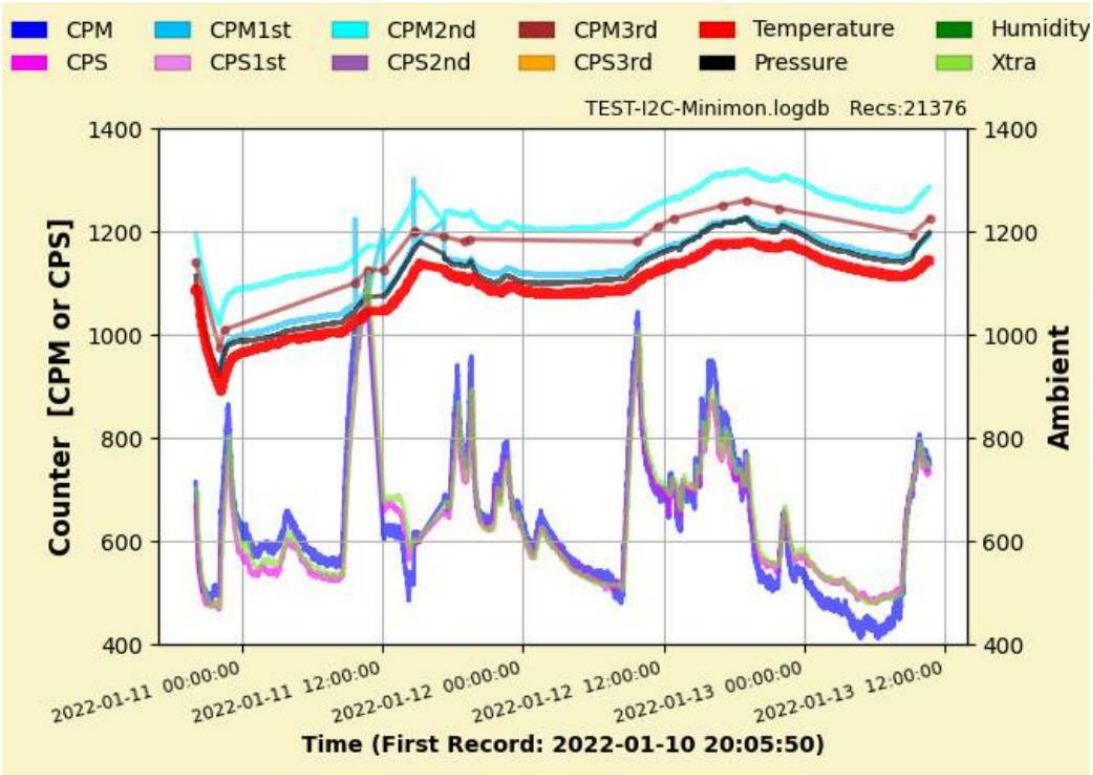


Figure 53: A run with I2C devices BME280, SCD30, SCD40 via USB-ISS, MiniMon and Manu

LabJack Devices

Running this device with GeigerLog is currently supported on Linux and it may work on Mac, but will NOT work on Windows!

Labjack is a company (https://labjack.com/) providing a range of data collection hardware, typically for laboratory and industrial purposes. Here the hardware device U3 in combination with the ei1050 probe for temperature and humidity is implemented.

Labjack provides a set of drivers and Python 3 based software to operate these devices.

For LabJack Python support see: https://labjack.com/support/software/examples/ud/labjackpython

Installation

To use the LabJack device you need the installation of (More details in the head of the Python code in file glabjack.py).
- the so called Exodriver
- the LabJackPython library
- the u3 Python package – included in the LabJackPython library package
- the ei1050 Python package – included in the LabJackPython library package

The Exodriver is here: https://labjack.com/support/software/installers/exodriver. The latest version is 2.6.0, compatible with Linux kernels from 2.6.28 onwards. Download, unzip and run:

```
$ sudo ./install.sh
```

LabJackPython 2.0.4 is available using pip, see Appendix I – Installation on page 167.

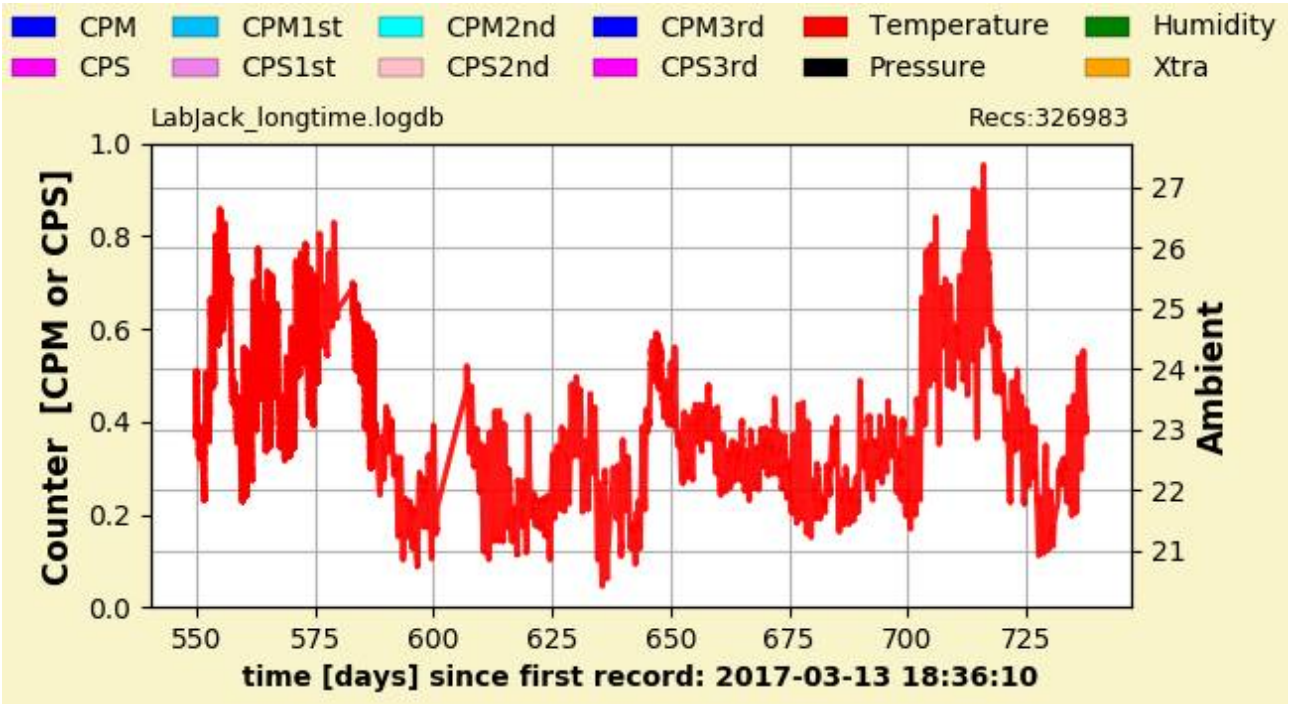


Figure 54: Long-Term Recording – 2 years – with the LabJack, showing Temperature only

MiniMon Devices - CO_2 Monitor

This device can be used with GeigerLog only when run on a Linux operating system!

This device and look-alikes are available from several companies. Mine is from **TFA-Dostmann** [65]). The original manufacturer may be **ZyAura** [66]); a (limited) datasheet is here [67]).

There is no brand-name for this device, so I created the name **MiniMon**.

Technology

MiniMon measures CO2 based on the Non-Dispersive-Infrared (NDIR) technology, which is a well established standard for measuring CO2. It uses an LED generated infrared beam, and measures the absorption of certain CO2-specific wavelengths in the air sample. The result is presented as ppm (parts per million).

For reference:

- outside air: CO2 in the low 400 ppm
- class-rooms, offices: CO2 1000-2000 ppm
- air exhaled by mammals: CO2 40 000 ppm (4%)
- Rooms, with people working in them, should be vented when CO2 reaches 1000 ppm
- The US Navy permits up to 7000 ppm on its submarines for combat-ready sailors.

MiniMon displays the CO2 concentration on a small B&W LCD screen, alternating with the display of temperature in °C. Some devices (not mine) also display relative humidity.

Installation

There is nothing to install. MiniMon has no buttons or switches and only a single plug for a micro-USB cable, which is used for power supply and data delivery. It has 3 LEDs in green, yellow, and red which are lit to indicate air quality. It is easily made a mobile device by connecting with a power bank – I used it with a 13000 mAh power bank, which was still 70% full after 1 week!

Connecting

MiniMon is used with the Linux HIDRAW (**H**uman-**I**nterface-**D**evice **Raw**) driver, which is the kernel interface for Raw Access to Human Interface Devices. These drivers are **specific to Linux**.

Once you have found MiniMon's hidraw address – explained next – you enter it into the GeigerLog configuration file in section `MiniMon` as parameter to `MiniMonOS_Device`. It will be something like `/dev/hidraw5`.

65 https://www.tfa-dostmann.de/

66 https://www.zyaura.com/product-detail/zgm053u/

67 https://www.zyaura.com/support-download/manual-zgm053u/

However, an easier way is to create a udev rule, so that you find MiniMon always under the same address, e.g. `/dev/minimon`, and can leave parameter `MiniMonOS_Device` on `auto.`

MiniMon Address – per MANUAL setting

Without plugging-in MiniMon issue the command:

```
ls -al /dev/hidraw*
```

Now plug-in MiniMon, and issue the same command. The newly appearing line, showing e.g. `/dev/hidraw5`, provides the needed hidraw address for MiniMon.

Depending on permission settings on your computer (explained in Appendix C – HOWTO deal with read and write permissions for the serial port when on Linux), you may also have to give this command:

```
sudo chmod 666 /dev/hidraw5
```

MiniMon Address – per UDEV rule (recommended)

With root permission add file `90-co2mini.rules` (included in the GeigerLog package) into folder `/etc/udev/rules.d`. To activate this udev rule, issue command:

```
sudo udevadm control --reload-rules && udevadm trigger
```

Then unplug and replug MiniMon. Your device will now always have address `/dev/minimon`.

NOTE: If you have multiple MiniMon and want to run them all simultaneously, read the file `90-co2mini.rules` for the changes to make to this udev rule.

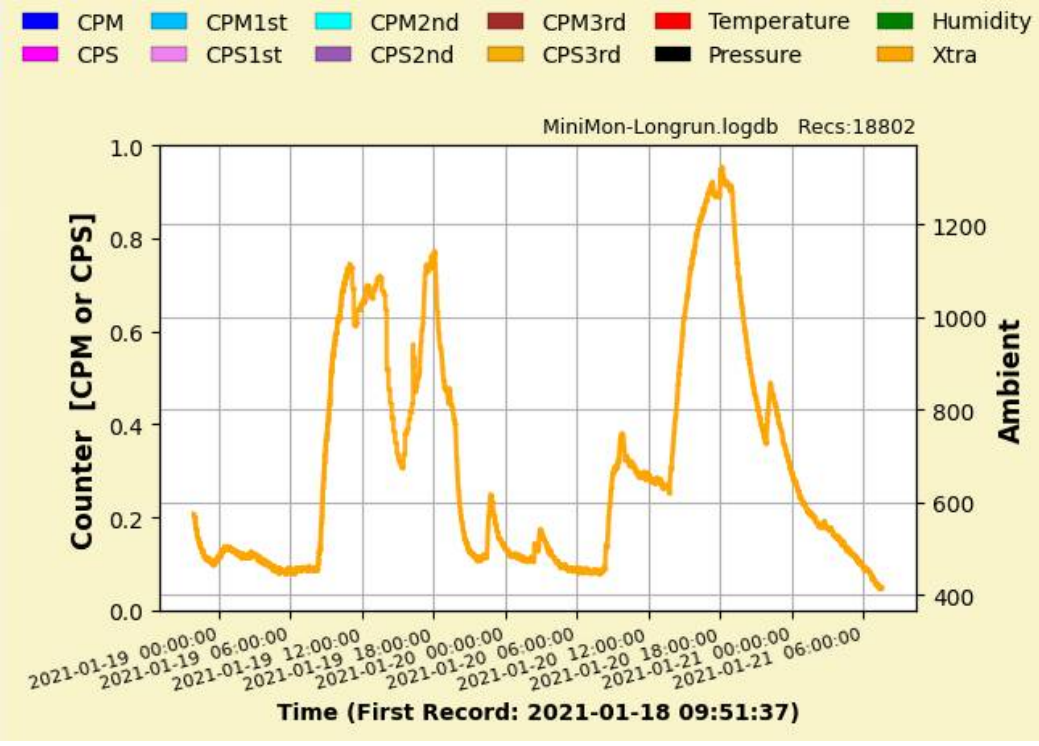


Figure 55: Measuring CO2 Concentration in an office

Logging Data

By default GeigerLog will read Temp (for temperature) and Xtra (for CO2) from MiniMon. You could also activate Humid (for humidity), but unless your device does have a humidity sensor, you would only read 0 (zero).

Figure 55 shows just the CO2 recording of an example run taken in my office. There is high-CO2 (~1000 ppm) during my working hours, fluctuating depending on opened or closed doors/windows, and going down to near 400 ppm during the night.

An in-depth comparison of this MiniMon device with two I2C based devices (SCD30, SCD41) is found in my article "**Measuring CO2 with SCD30, SCD41, and MiniMon**" [68]).

For more on CO2 measurement see also chapter I2C Devices on page 87.

History Download

The device has no internal memory, so a history download is not possible.

68 https://sourceforge.net/projects/geigerlog/files/Articles/Measuring%20CO2%20with%20SCD30%2C%20SCD41%2C%20and%20MiniMon-v1.0.pdf/download

Formula Devices

This is a device with the main purpose to support GeigerLog's **Formula Interpreter**, see ValueScaling and GraphScaling on page 37. Most often it is used to create synthetic data for some tests.

An example for very complex formulas is shown in Figure 56. (Explained in Formula Interpreter on page 38. For more functions look there.)

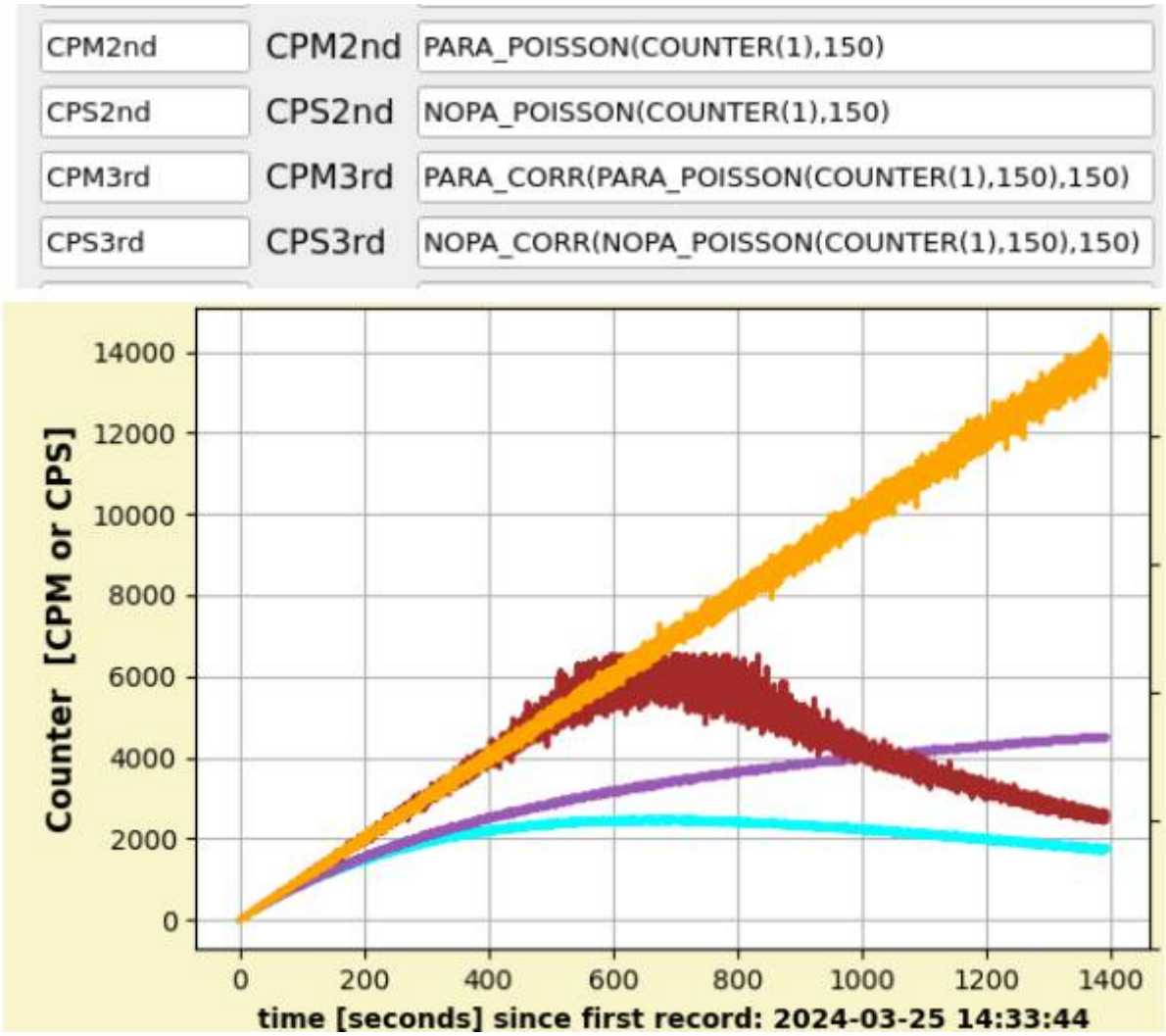


Figure 56: Evaluating the Deadtime Effect on Count Rates

Manu Devices

Manu is a plugin device which allows to enter numeric data manually, which then will be shown in the graph just like any other data. This is helpful to bring data into GeigerLog which you can't (easily) enter electronically, but want to compare them directly with other data in GeigerLog.

Examples could be:

- Study the influence of the distance between Geiger counter and radioactive source. Enter the distance via the Manu device e.g. as variable Xtra.
- Measure the Geiger tube's response to temperature. You may have a remote Temp-Sensor but it has only a manual readout (like a meat-temperature sensor for the barbecue). Enter the temp via the Manu device e.g. as variable Temp.
- You measure the capacity of a LiIon battery powering a microchip. The chip uses its own ADC to report a voltage via the GeigerLog **WiFiServer device**, and you use a precision DVM to measure true voltage manually and enter it to the **Manu device**.

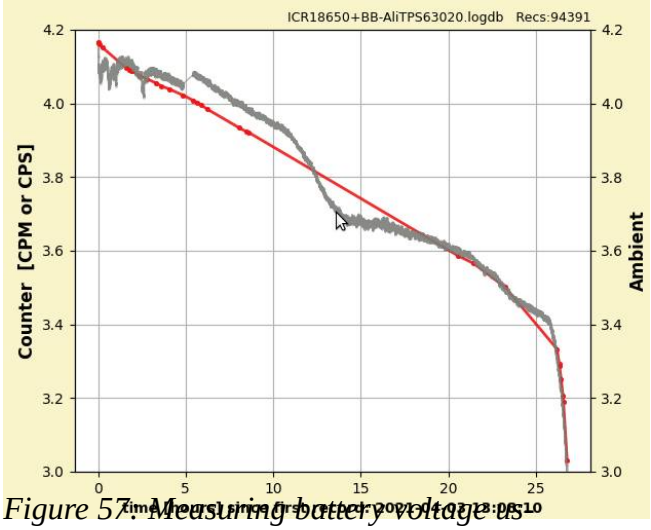


*Figure 57: Measuring battery voltage using GeigerLog **WiFiServer** and **Manu.** See text*

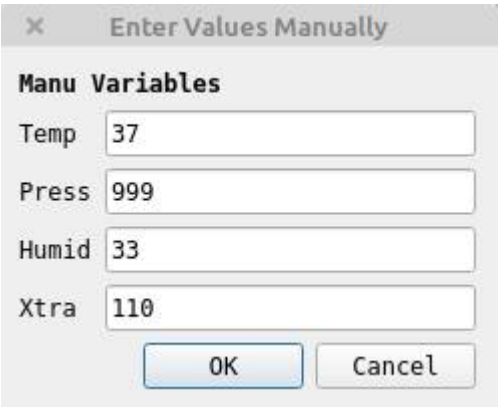
Figure 57 shows the result of the latter experiment. In gray the online-ADC voltage, in red the DVM voltage, using the variables CPM3rd and Press (my motivation for the Manu device).

Installation

There is nothing to install. Simply activate the **Manu** device in the **[Manu]** section of the GeigerLog configuration file `geigerlog.cfg` and define the variables you want to use.

Settings

Click menu **Device → Manu Series – Enter Values Manually.** In the dialogue you can enter values for all the variables you have configured for Manu, which can be up to the full set of 12 variables. The dialogue will expand to provide entry fields for all variables.



Choosing **Step** will keep the value entered for all next log cycles, appropriate when the value remains constant, like for the source-distance experiment described above. Using **Point** creates single entries, which GeigerLog connects with a line, like the red curve of Figure 57.

WiFi Devices

This applies to all web enabled devices. They could be connected either by wire (LAN cable) or wirelessly (WiFi). However, it seems highly unlikely that devices as discussed here will ever be connected by a LAN cable; so we'll limit the discussion to **WiFi connections only**.

Distance

The WiFi devices will be connected to a local WiFi, and those as well as GeigerLog can freely roam within the reach of this WiFi net. To support a greater distance between device and GeigerLog, WiFi range extenders can be used like with any other WiFi enabled device. For even greater distance your router can be set up with port forwarding technology, so that calls can even go via the internet to any place on the planet.

Speed

The devices show up in my FritzBox Mesh with speeds between 60 … 80 MBit/s. This is some 500 to 800 times fast than current connections over USB-To-Serial! A download of the 1 MByte memory of some counters – if it were possible – would take under 3 sec, and not more than 3 min as it is currently the case!

Power

One disadvantage is that WiFi needs more power than a USB / USB-To-Serial connection. In addition, the cable-connected device was charged while it was connected. Now it may have to rely more or even solely on its own battery.

However, other power delivery methods can be used, like USB Wall Chargers, solar power, power bars and others.

WiFi Devices Types

GeigerLog currently supports two types of WiFi devices:

- **WiFiClient Devices** (Example: GQ's GMC counters when WiFi enabled)
- **WiFiServer Devices** (Example: Raspberry Pi as satellite to GeigerLog)

They will be explained next.

WiFi Devices – WiFiClient Devices

These are external devices which act as **WiFiClients** and call the web server built into GeigerLog.

Example: GMC counters act as a **WiFiClient** when sending data via WiFi!

WiFiClients operate on their own schedule, as set by the user. GeigerLog, like any other server, is waiting patiently for such a call, and becomes active only when such a call arrives. The GMC counters, as example, have a default setting of once every 2 min for calling the gmcmap.com server and updating the data, and cannot be faster than once every 1 minute.

To transfer data to the server, the WiFiClient has to wrap the data values into the URL used to call the server. In web-lingo it is creating a GET call [69]).

The situation is basically the same as if you were typing a search request into the search field of your browser and then send it to the desired search engine. The search engine will look at the things you typed in your request and returns a web page as response, and then waits for the next call.

When GeigerLog receives a call, it extracts the data and answers with only a simple response, like "OK". This is its complete response “web-page” it sends. In essence, the WiFiClient has uploaded data to GeigerLog by seemingly requesting a web page, in which the WiFiClient is not interested in (and GeigerLog not delivering).

Otherwise the data are being handled by GeigerLog just like any other data, they will be saved, printed, displayed, scaled, and plotted.

Even the GMC counters’ calls can be redirected into GeigerLog, but it does take some extra effort. It will be shown below.

Installation

In the `[WiFiClientDevice]` section of the GeigerLog configuration file `geigerlog.cfg` activate the **WiFiClient** device, and change other settings as appropriate:

Any **WiFiClient Device** needs to know the IP address associated with the computer running GeigerLog. If you don’t know already, GeigerLog knows its IP Address and tells you when you click menu **Web → Show IP Status** or the **IP** icon in the toolbar**.** This is set by your home network and cannot be changed by GeigerLog. The device also needs to know the port number on which GeigerLog is listening. The default is **8000**; change if needed.

NOTE: All GMC Geiger counter can connect **only** to WiFi networks on **2.4 GHz, NOT 5 GHz**!

Device Type

GeigerLog supports two types of WiFiClient Devices: “**Generic**” and “**GMC**”. Only one can be active in a run, which is configured in the `geigerlog.cfg` configuration file. Any calls from the other one will be ignored by GeigerLog during that run.

69 https://www.w3schools.com/tags/ref_httpmethods.asp

Type 1: GENERIC:

As the name implies, this is the more universal type. The external device assembles all data into a GET command. Assuming a device which wants to transmit CPM=99, CPS=3, and Temp=24.6, this URL takes the form (for the M, S, … codes see Graph Dashboard – Visualize Your Recordings on page 27): http://<GeigerLogIP>:<GeigerLogPort>/GENERIC/?M=99&S=3&T=24.6

or, using all 12 variables with simple values and GeigerLog IP=10.0.0.20, GeigerLog Port=8000:
http://10.0.0.20:8000/GENERIC/?M=1&S=2&M1=3&S1=4&M2=5&S2=6&M3=7&S3=8&T=9&P=10&H=11&X=12

GeigerLog will answer with: OK for Generic WiFiClient Device

Type 2: GMC:

This mode accommodates the **WiFi-enabled GMC counters** by imposing certain restrictions to GeigerLog's GENERIC Type. The default behavior of the GMC counters is explained in chapter Error: Reference source not found on page Error: Reference source not found.

In GeigerLog the AID and GID parameters are ignored, can have any value, and may even be absent. Of the 3 variables: CPM, ACPM, uSV [70]) only CPM matters, because the others can be calculated by GeigerLog. Nevertheless, all 3 variables can be recorded in GeigerLog. The counter limits data update to no more than once per minute, so CPM is the only meaningful variable to send.

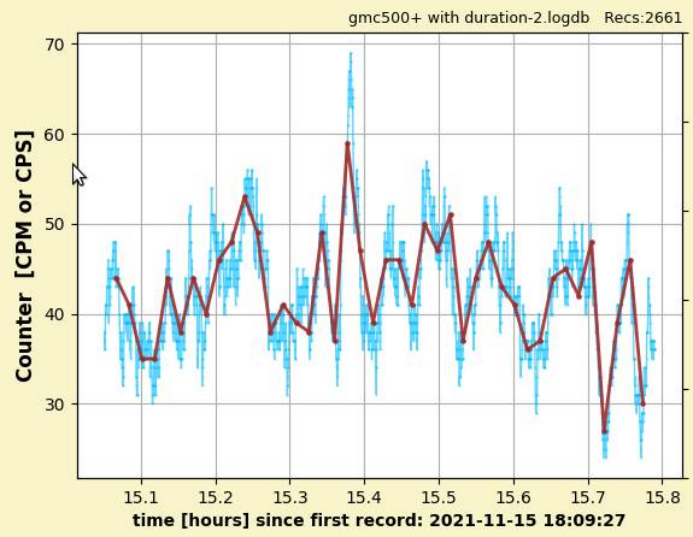


Figure 58: Running a GMC-500+ counter both via USB-Serial (blue) and WiFi (brown)

Figure 58 shows a comparison of data collected from a GMC-500+ by either a regular USB connection, or from the counter running as WiFiClient. The traces fully agree; except that the USB connection allows updates every second, while the WiFi connection only every minute.

To make this recoding possible, a **redirection** had to be applied to the GMC-500+ counter's WiFi data, which is explained in the next chapter.

70 https://www.gmcmap.com/AutomaticallySubmitData.asp

Redirection for GMC counters

It would be nice if the GMC counters could talk directly to GeigerLog via WiFi, but this is not possible.

The GMC counters have a bug and a deficiency in the counter's firmware, which makes their use a lot more difficult than it has to be. Their use with GeigerLog requires a **URL redirection** with the additional use of a **`GLrelay.py`** relaying software **.** It is explained next.

Deficiency – Port Number Problem: The firmware of a GMC counter forces the use of port=80. However, port numbers 0 … 1024 are reserved for users with Admin rights, and for security reasons it would be a bad idea to run GeigerLog as Admin, solely to open such a server port! Most likely it will even be in conflict with an active, regular web server on your PC, which almost always also listens at port=80.

Bug – "Unsafe" Operation Problem: The GMC counter's firmware forms the URL in such a way that modern servers reject their contact as a security issue. One can overcome this by explicitly putting an "Unsafe" into their configuration file [71]). But which Web server Admin would convert their server intentionally into a security risk?

GeigerLog takes a different approach. The software **`GLrelay.py`**, available in the **`gtools`** subdirectory of GeigerLog, acts as a relay between the GMC counter and GeigerLog. It it a web-server which does nothing but receiving the counter data on port 80 and sending the very same data out on a non-administrative port, like e.g. port 8000. Of course, it has to be run as **administrator**. There is little risk as it can't do anything else but bouncing the data.

To see Help information start the program with: **`python gtools/GLrelay.py -h`**

which gives:

```
Usage:  GLrelay.py [Options]

Start:  GLrelay.py      MUST start as ADMIN
Stop:   CTRL-C

Options:
    -h, --help          Show this help and exit
    -V, --Version       Show version status and exit
    -I, --GLIP          Set the IP of the computer running GeigerLog
    -P, --GLPort        Set the Port where GeigerLog will be listening

Example: start GLrelay.py with options:

    # set GeigerLog Address to '10.0.0.20:8000'
    GLrelay.py --GLIP=10.0.0.20 --GLPort=8000

or: # same
    GLrelay.py -I 10.0.0.20 -P 8000
```

71 http://httpd.apache.org/docs/2.4/de/mod/core.html see HttpProtocolOptions

Simulation software

To easily test such a WiFiClient Device a simulation software **`Simul_WiFiClient.py`** is available in the GeigerLog folder **`gtools`**, which can be run to simulate either a **GENERIC** or a **GMC** WiFiClient Device.

For Help information start with: **`python gtools/Simul_WiFiClient.py -h`**

which gives:

```
Usage:  Simul_WiFiClient.py [Options]

Start:  Simul_WiFiClient.py
Stop:   CTRL-C

Options:
  -h, --help       Show this help and exit
  -V, --Version    Show version status and exit
  -c, --cycle      Set the cycle time in sec (default 1 sec)
  -I, --TargetIP   Set the IP of the target computer
  -P, --TargetPort Set the Port of the target computer (default 8000)
  -T, --Type       Set the Type of client to simulate 'GMC' (default) or
'GENERIC’

Example: start Simul_WiFiClient.py with options:

    # set cycle time = 3 sec, and Target Address to '10.0.0.20:8000':
    Simul_WiFiClient.py --cycle 3 --TargetIP=10.0.0.20 --TargetPort=8000

or: # same, and also set Type:
    Simul_WiFiClient.py -c 3 -I 10.0.0.20 -P 8000 -T GMC
```

Run GeigerLog with its **WiFiClient Device activated**, and from a 2nd terminal start the simulation with e.g.:

`python gtools/Simul_WiFiClient.py -c 3 -I 10.0.0.20 -P 8000 -T GMC`

WiFi Devices – WiFiServer Devices

These are external devices which act as **Web Server** themselves and send data only when requested by GeigerLog, who, in this situation, is the WiFiClient. This is how most devices respond when connected with a (USB-)cable, but with a WiFiServer it is wirelessly!

Example: A **Raspberry Pi** can be made to act as a **WiFiServer**, thereby WiFi-enabling external devices, and will transfer the device's data to GeigerLog. Examples given below.

In Web terms GeigerLog has here the role of a browser and calls the WiFiServer device. In fact, you could type the same calls, which GeigerLog will use, into the web address field of any browser (Firefox, Chrome, Edge, …) and the WiFiServer Devices will respond with the same answers that they would give to GeigerLog.

We know that we can call a server on the internet by entering a request like "www.google.com" into a browser's address field. But Google does not get this request. Instead there are DNS-resolver servers in the internet, which translate this to a so called IP address 142.250.185.228. Only this is what Google sees. But that is not complete. Every server can listen on 65536 ports, numbered from 0 … 65535. This number must be given, except when it is not given, the server assumes that port 80 is meant, so the server adds the 80 to the IP address internally, and executes whatever is called for.

But every server listening on ports from 0 … 1023 must be run under administrative rights for security reasons. We don't want this extra burden, and therefore use any other port from the range 1024 … 65535. For unknown reasons – perhaps historic ones – people like port numbers in the 4000 and 8000 range. The default port number for our WiFiServer port is 4000.

What GeigerLog needs to know about the WiFiServer is its **IP address** and the **port number** at which it is listening.

Let's assume the WiFiServer device has IP address 'serverip', and port number 'port', and offers the data from its root directory subfolder 'folder' [72]). We further assume this WiFiServer provides Geiger counter CPM and CPS data from a Geiger counter, as well as Temperature, Pressure, and Humidity data from e.g. an I2C sensor BME280.

GeigerLog uses only four possible requests, which are: **id**, **lastdata**, **lastavg,** and **reset**. GeigerLog sends the http request, and the Device answers with a CSV (Comma-Separated-Values) list of numeric data.

GeigerLog: http://serverip:port/folder/**id**

Device: WiFiServerDeviceName
Example: GeigerLog WiFiServer

GeigerLog: http://serverip:port/folder/**lastdata**
http://serverip:port/folder/**lastavg**

Device: CPM, CPS, CPM1st, CPS1st, CPM2nd, CPS2nd, CPM3rd, CPS3rd, Temp, Press, Humid, Xtra
Example: 66, 1, nan, nan, nan, nan, nan, nan, 23.4, 998.1, 76.5, 620

72 The folder name may be empty (folder=""); then the data will be served from the root directory.

for: CPM=66, CPS=1, Temp=23.4 °C, Pressure=998.1 hPa, Humidity=76.5%, Xtra=620 [ppm CO2], all other values in the list are 'nan', i.e. missing values.

GeigerLog: http://serverip:port/folder/**reset**

Device: Ok

Example: Ok

The **id** call serves the double purpose to verify that the device is present, and to get its name. If the id call fails, GeigerLog will not work with this device!

The difference between **lastdata** and **lastavg** is that the former should always be the last available set of values, while the latter is an average of those values over a certain period of time calculated by the WiFiServer device itself. It is also up to the WiFiServer to define the length of this averaging period.

This is of interest if e.g. the device samples in 1 sec intervals, but the reading by GeigerLog is done in 1 minute intervals. In this case an averaging of all data over 1 min by the WiFiServer device would be appropriate.

If the WiFiServer device cannot provide all values for all variables, then '**nan**' (without quotes) must be inserted at the place of the missing variable, which stands for the computer term 'Not-A-Number', and is interpreted by GeigerLog as a missing value. GeigerLog always expects 11 commas total, and gives an error if values are missing, and ignores the whole record.

A **reset** allows the user to program a reset on the external device. GeigerLog does not verify a reset.

Installation

In the `[WiFiServerDevice]` section of the GeigerLog configuration file `geigerlog.cfg` activate the **WiFiServer device**. You must set the IP address, Port number, and folder of the external WiFiServer device. Optionally set the other parameters timeout, data type, and variables.

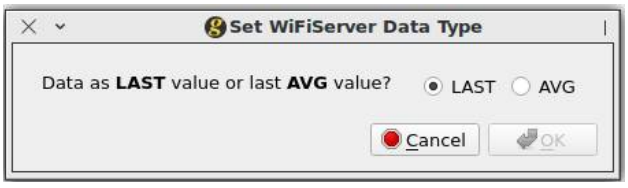
Ping WiFiServer

Pinging a server allows to verify its presence and ability to respond to your requests. A success or failure note will be printed into NotePad, which could be like:

```
==== Pinging WiFiServer IP: 10.0.0.49 ========================================
Ping Result:                  Success, time=1.63 ms
```

Set WiFiServer Data Type

Click menu **Device → WiFiServer Series → Set WiFiServer Data Type … .** In the dialogue you can select whether the **LAST** or **AVG** values will be used.



Note that the changes can only be applied when you are **not logging!** Otherwise the OK button is grayed out and inaccessible.

Examples of WiFiClient – WiFiServer Devices

Three examples are given: one uses the GeigerLog computer to create a Demo WiFiServer, one uses a Raspi acting as a WiFiServer device, and the third sets up a Demo WiFiServer on a web server using PHP coded pages.

1) The GeigerLog Computer as Demo WiFiServer Device

The easiest-to-use option: The program **DemoWiFiServer.py** is available in the **gtools** folder of GeigerLog. Configure its source code for your situation, which really only is the line WiFiServerPort = 4000 in the code of DemoWiFiServer.py (line 40ff) and the matching setting in the GeigerLog configuration file WiFiServerPort = 4000 (leaving WiFiServerIP = auto, and WiFiServerFolder = auto).

Then run GeigerLog with WiFiServer activated. From a 2nd terminal start the Demo with:

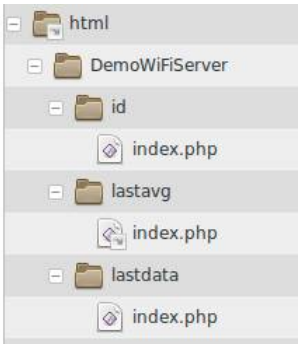
```
python gtools/DemoWiFiServer.py
```

2) Raspi as WiFiServer Device

The program GLWiFiServer needs to be installed on a Raspi and started. It converts the Raspi to a WiFiServer device. This program is currently in version 3.3 and included with GeigerLog. Find it in folder `gtools`.

For a more in-depth explanation use manual from a previous version; Download **"Using Raspi as a GeigerLog WiFiServer Device-v2.0"** [73]).

3) Demo Web Server with PHP



If you have a web server, like Apache (but any other will do as well) with PHP, running on a computer near you, you can set up a **Demo web server**, using the file structure as shown in the picture. In this case the **web server's document root** is **'html'**.

Naming all files 'index.php', but placing them in separate directories, allows to call the directory only; no need to give a filename with php extension, which may conflict with a later time use of microprocessors! Then use these simple scripts for id, lastdata, and lastavg for getting it to behave like a real WiFiServer device:

Demo Web Server with PHP – Script: id

Save file named index.php under

```
<web server document root>/DemoWiFiServer/id/index.php
```

with this content:

```
<?php
```

73 https://sourceforge.net/projects/geigerlog/files/Raspi/

```
// Sends the ID of a WiFiServer Device in the form of a simple text string.
echo "WiFiServer Simulator on Apache PHP";
?>
```

Demo Web Server with PHP – Script: lastdata

Save file named index.php under

`<web server document root>/DemoWiFiServer/lastdata/index.php`

with this content:

```
<?php
// Behaves as a WiFiServer Device yielding responses to /lastdata, /lastavg
// requests by returning a CSV string of data, like:
//    configured    : CPM, CPS, Temp, Press, Humid, Xtra
//    response like:   111.254, 1.197,,,,,,,21.000, 6.000, 48.000, 84.000
//    Commas must be put in for missing values
//
// The data are synthetic, random data. The CPM/CPS data are approximately
// Poissonian, ambient data are random over meaningful intervals, see code.

function rand_pseudo_normal($av, $sd) {
/*
* generates an approximately normal distribution based on "Box–Muller transform"
* random deviate generator.
* https://en.wikipedia.org/wiki/Box%E2%80%93Muller_transform
* @param float|int $av Mean
* @param float|int $sd Standard deviation of Mean
* @return float
*/

$x = mt_rand() / mt_getrandmax();
$y = mt_rand() / mt_getrandmax();

return sqrt(-2 * log($x)) * cos(2 * pi() * $y) * $sd + $av;
}

function rand_pseudo_poisson($av){
// gets value from normal with StdDev = sqrt(mean)
// and takes abs to make all values positive.
// Reasonably ok as Poisson with $av >= 10
return abs(rand_pseudo_normal($av, sqrt($av)));
}

$CPM = rand_pseudo_poisson(100);
$CPS = rand_pseudo_poisson(1.67);
$T = rand( 20, 25);
```

```
$P = rand(980, 1030);
$H = rand( 40, 60);
$X = rand( 80, 100);
// sending: 'CPM, CPS, Temp, Press, Humid, Xtra'.
echo sprintf("%0.3f, %0.3f,,,,,,%0.3f, %0.3f, %0.3f, %0.3f", $CPM, $CPS, $T, $P, $H,
$X);
?>
```

Demo Web Server with PHP – Script: lastavg

Save file named index.php under

`<web server document root>/DemoWiFiServer/`**`lastavg`**`/index.php`

with the exact same content as used for **lastdata**. Of course, then lastavg is the exact same as lastdata, and not an average, but at least your Demo server gives an acceptable response.

Demo Web Server with PHP – Script: reset

Save file named index.php under

`<web server document root>/DemoWiFiServer/`**`reset`**`/index.php`

with this content:

```
<?php
// Sends “Ok” in the form of a simple text string.
echo "Ok";
?>
```

Raspi Devices

GeigerLog supports **Raspberry Pi** – or **Raspi** for short – in the form of 2 devices, which will be described next:

- **RaspiI2C Devices**
- **RaspiPulse Devices**

Keep in mind that a Raspi comes in versions from Raspi1 (year 2012) to Raspi 5 (year 2024), and that later versions have increasing computing power! Some older versions may simply not have enough power to run certain features of GeigerLog (e.g. an Audio Counter).

Raspi Devices need a Raspi to run

Unsurprisingly, both the **RaspiPulse Device** as well as the **RaspiI2C Device** do function only when GeigerLog is running on a real Raspi! However, neither device crashes GeigerLog when run on a non-Raspi computer, but GeigerLog knows where it is running on, and the devices simply refuse to connect.

Things to consider when running any software on a Raspi

At first glance the Raspi is just another Linux computer, so it also runs Python and therefore it also runs GeigerLog. And indeed, it does so quite well.

However, there are two important differences, which distinguish the Raspi from other computers:

1) it has lower computing power
2) it offers **GPIO** (General Purpose Input Output) hardware pins to the user

1) Low computing power

An example for the limits imposed by low computing power is running GeigerLog's AudioCounter on an Raspi4. It does work in principle, but with somewhat elevated count rates the Raspi is overwhelmed, while a regular desktop still performs flawlessly, see the experiment reported in chapter GeigerLog's AudioCounter on Raspi4 (Buster) and Raspi5 (Bookworm) on page 71.

2) GPIO (General Purpose Input Output) Pins

GPIO pins offer options which are not easily available on desktop computers, let alone laptops. They are also accessible through high-level languages, like Python, which greatly expands the possibilities.

The pins are all digital and can be read and set; they can be logically HIGH (+3.3V), or logically LOW (0V). Libraries are available which allow to execute very complex operation with simple commands, like for SPI, I2C, UART, PCM, and single-pin interrupts.

There are two avenues for GeigerLog to use GPIO, which are both already implemented in GeigerLog:

A) Run full GeigerLog on a Raspi, and activate GeigerLog devices, which use the GPIO pins and features.

B) Build small scripts to run on Raspi, which do nothing but collect data from the GPIO and transfer them to GeigerLog via WiFi.

A) Is done by GeigerLog with the devices **RaspiI2C Device** and **RaspiPulse Device** (see below).

B) Is done with the script **`GLWiFiServer.py`** running on a Raspi. It transfers data to GeigerLog, running on a separate desk-/laptop, via WiFi, and GeigerLog is acting on the data using its **WiFiServer Device** (see chapter WiFi Devices – WiFiServer Devices on page 101). **`GLWiFiServer.py`** is available as part of the GeigerLog package in the `gtools` directory.

In both cases the Raspi can be run "**headless**", meaning having no monitor, no keyboard, and no mouse; Raspi is controlled remotely through VNC (Virtual Network Computing) and SSH.

Future

Small scripts without the overhead of – in particular – a GUI as in GeigerLog generally require **little computing power**. Then, of course, even less powerful Raspis, like the **Raspberry Pi Zero W**, might suffice as such a "data converter". They would act as WiFiServer, delivering the data via WiFi to GeigerLog.

GPIO Pin Specifications

WARNING: Raspi's GPIO data pins tolerate no more than **3.3 V** – applying 5 V to them might fry the whole Raspi!

The pins relevant to GeigerLog's current devices are shown in the Pinout picture; more details with valuable explanations in multiple languages online [74]).

74 https://de.pinout.xyz/pinout/pin11_gpio17#

Raspi Devices – RaspiI2C Devices

These devices take advantage of the I2C support natively built into Raspi.

It can be considered as an alternative to the use of dongles for I2C devices as reported in chapter I2C Devices on page 87. The Raspi is easier to program.

Supported I2C devices: **LM75** (temperature), **BME280** (temperature, barometric pressure, humidity), **BH1750** (light sensor visible), **TSL2591** (light sensor visible and infrared), **SCD30** (CO2, temperature, humidity), **SCD41** (CO2, temperature, humidity), **GDK-101** (PIN diode for detection of radioactivity), **BMM150** (Geomagnetic).

Installation

GeigerLog uses Raspi's I2C interface "I2C1", which means it uses the physical board pins 3 (for SDA) and 5 (for SCL). More details in the "**Using Raspi as a GeigerLog WiFiServer Device**" manual [75]).

In the GeigerLog configuration file simply activate the **RaspiI2C** device, and configure the sensors.

Results

Figure 59 shows an example with GeigerLog running on the Raspi4 using the RaspiI2C device, and a single sensor LM75B.

Temperature is in red and averaged temperature in black.

The Raspi4 has no difficulty delivering these values.

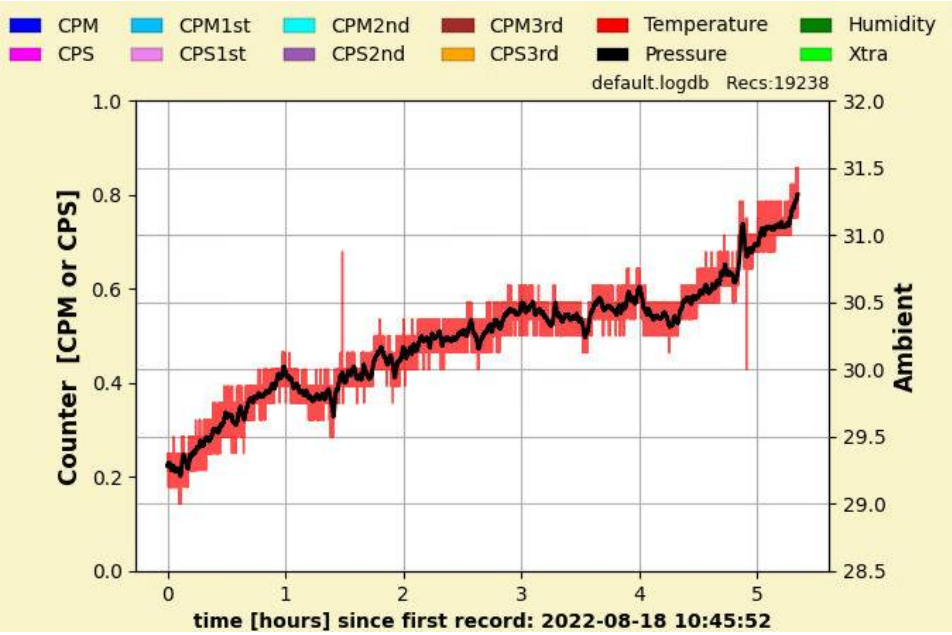


Figure 59: GeigerLog running on a Raspi using the RaspiI2C Device

75 Manual included in package: https://sourceforge.net/projects/geigerlog/files/Raspi/Using%20Raspi%20as%20a%20GeigerLog%20WiFiServer%20Device-v2.0.zip/download

Raspi Devices – RaspiPulse Devices

These devices take advantage of Raspi's ability to allow **hardware interrupts** from the GPIO pins. Any pulse arriving on the selected GPIO pin will be counted; counts in a 1 sec period are reported by GeigerLog as CPS.

Pulses may come from a variety of sensors, like tachometer, distance sensors, and other. When the pulses are coming from a Geiger counter, the Raspi becomes a full blown Geiger device, including ability to store history of any length the SD card can handle, and report it via WiFi.

The device measures CPS and calculates CPM; both values are available for any combination of the CPS/CPM variables.

However, it is only meaningful to define 1 out of CPS, CPS1st, CPS2nd, CPS3rd, as each variable would receive the same value! And likewise, it is only meaningful to define 1 out of CPM, CPM1st, CPM2nd, CPM3rd as each variable would receive the same value!

Any defined ambient variable will be ignored.

Installation

In the GeigerLog configuration file simply activate the device. Further, configure pin number, edge detection, and variables.

By default the GPIO pin used is **GPIO#17** (Boardpin #11). The pulse signal is expected to be normally HIGH (+3.3V), and LOW (~0V) during a pulse, and thus the edge detection is set to a **FALLING** edge.

Results using a Raspi5

I ran the **RaspiPulse Device** with my "Ukraine Counter"[76]). The counter uses the Russian SBM20 tube. For each Geiger event it produces a 3.3V pulse, active low, length 160µs (roughly the length of the tube deadtime). The outcome is shown in Figure 60. The result is near-perfect!

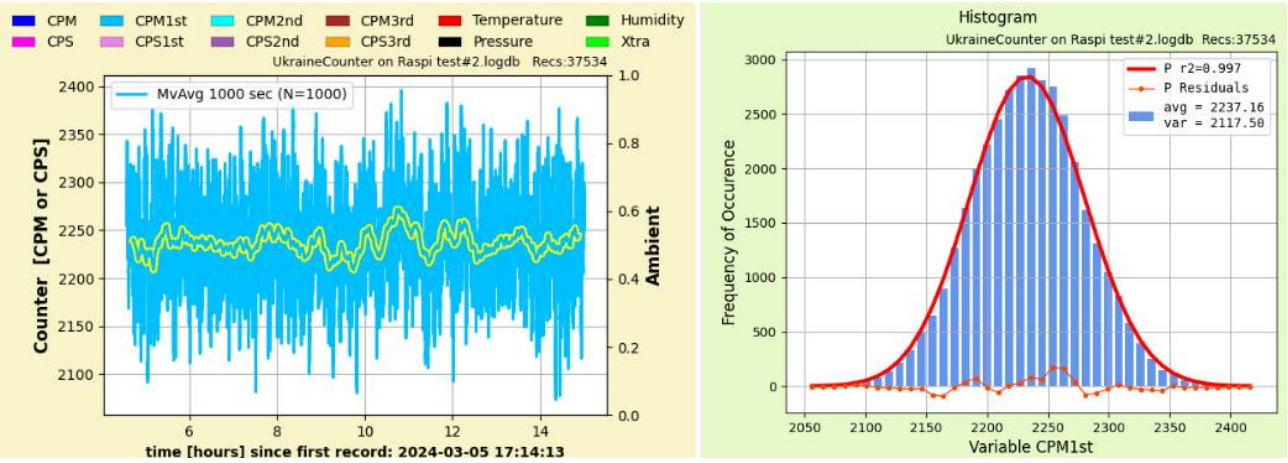


Figure 60: The RaspiPulse Device on a Raspi5 with the Ukraine Counter

76 The counter was originally developed and shipped from Ukraine, https://de.slideshare.net/IoTdevices/ggreg20-v3-datasheeteng)

Results using a Raspi4

A CAJOE [77]) Geiger counter was connected to the **Raspi4** and run overnight, results in Figure 61.

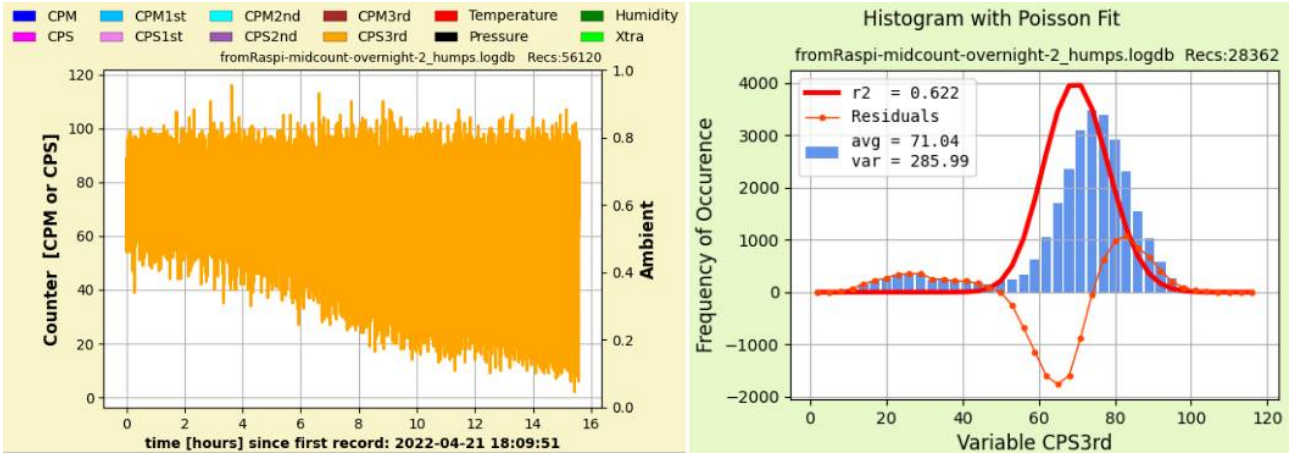


Figure 61: GeigerLog running on a Raspi using the RaspiPulse Device.

The counts are coming from a CAJOE Geiger counter.

The time course of CPS (left side) shows a constant upper value of approx. CPS=100, but the lower value is continually decreasing from approx. CPS=50 to CPS=10. A Poisson fit (right side) clearly shows a 2-hump distribution.

Not good; clearly, something has gone wrong.

The most likely explanation is that Linux is not a real-time operating system (and neither is Windows nor Mac), and so the timing needed to handle the interrupts may be missed. On top of it we are running a complex Python application, which adds its own burden of timing issues.

For a lower count rate it might still work, but there clearly are limits! It works much better when this Raspi4 is relieved from the burden of a GUI, see next chapter.

77 https://sourceforge.net/projects/geigerlog/files/Articles/GeigerLog-Radiation-v1.1%28CAJOE%29-Support-v1.0.pdf/download

Raspi Devices – Raspi as Satellite to GeigerLog

In this case the Raspi will NOT run the full GeigerLog, but instead run only a small script, acting like a satellite to GeigerLog, fulfilling only two specific tasks:

A) First task it to collect data. This can be done by a variety of methods, e.g. by counting counts on a GPIO pin, or by querying one or more I2C devices, or by querying other devices, including a GMC Geiger counter with a USB connection.

B) Second task is to send those data when GeigerLog requests them. All communication is done via WiFi.

Such a script is **`GLWiFiServer`** [78]). For a detailed description of script and results download **"Using Raspi as a GeigerLog WiFiServer Device-v2.0"** [79]). In short, it can:

1. **WiFi-enable a GQ GMC-300E+** Geiger counter – which has no native WiFi capabilities – by connecting it to a Raspi, or connect any other GMC counter to the Raspi for **improved WiFi performance**.
2. WiFi-enable I2C sensors like **LM75B** for temperature, **`BH1750`** for visible light intensity, **`GDK101`** as a solid state "Geiger" counter. More sensors will be available in future releases.
3. **WiFi-enable a Geiger counter (CAJOE) via Pulse counting** on Raspi's GPIO pins.

Results

As an example the results from a pulse-count using a CAJOE Geiger counter are shown in Figure 62.

With the count rates used here the Poisson test confirms valid data.

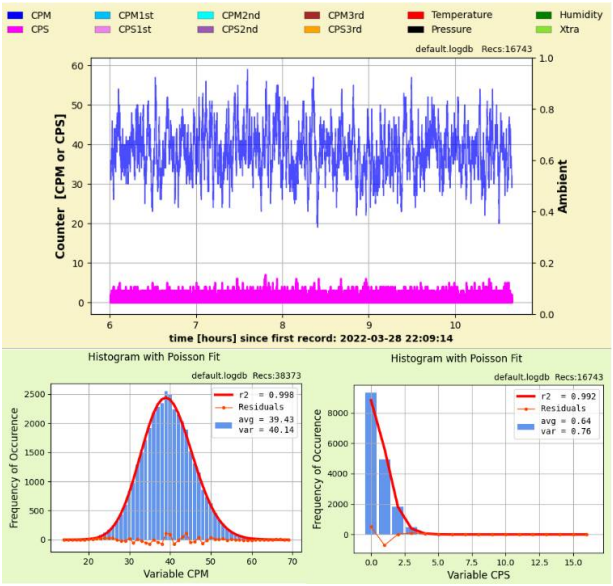


Figure 62: CAJOE Counter Result with Raspi in Pulse Counting mode – Background-Radiation

78 Latest version of GLWiFiServer.py is in GeigerLog's gtools directory.

79 https://sourceforge.net/projects/geigerlog/files/Raspi/

Next example shown in Figure 63 is created running **GLWiFiServer.py** (version 3.3) on a Raspi. Devices connected to the Raspi are: a GMC-500+ Geiger counter, an I2C sensor LM75B for temperature, and an I2C sensor BH1750 for visible Light intensity.

The data demonstrate the light sensitivity of a M4011 Geiger tube.

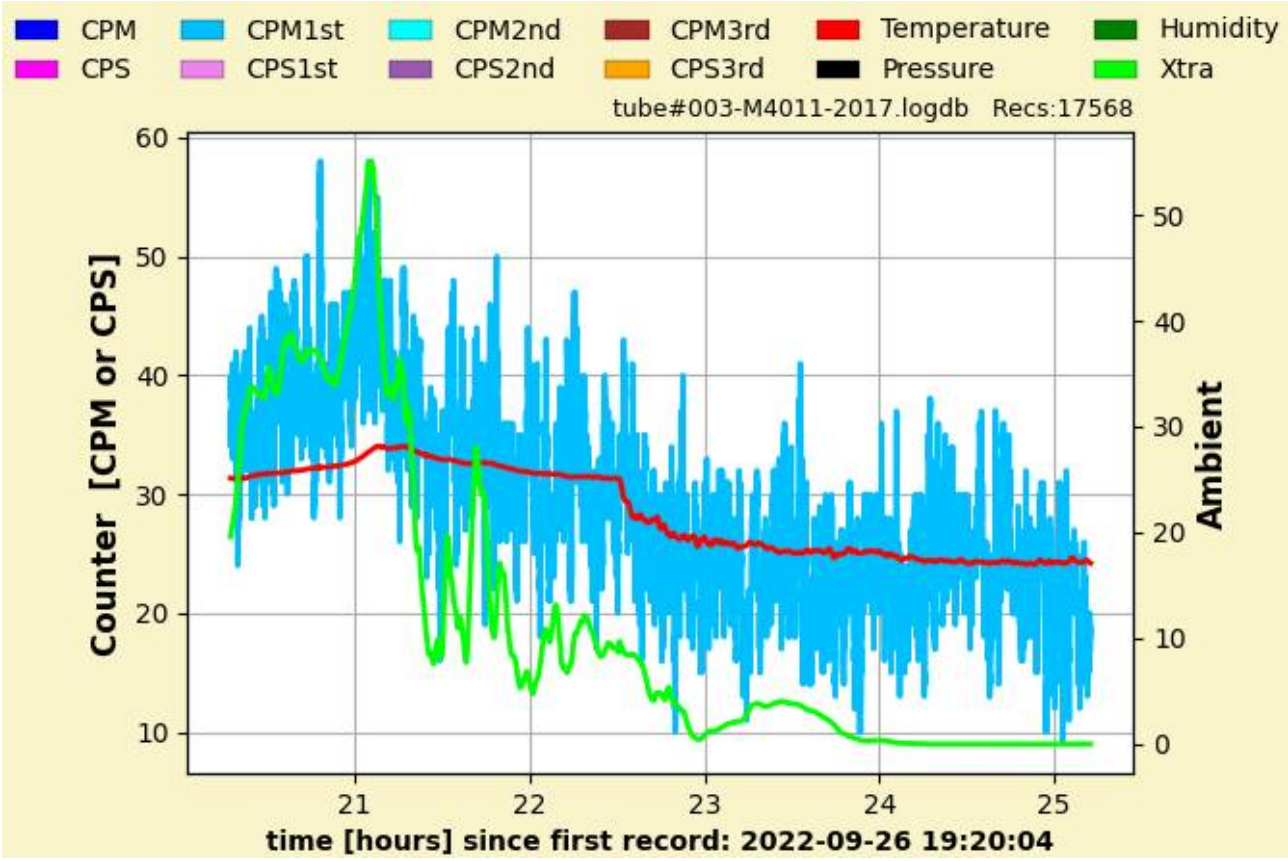


Figure 63: Raspi as WiFiServer

Blue: GMC-500+ Geiger counts, red: Temperature from I2C sensor LM75B, green: visible Light intensity from I2C sensor BH1750

SerialPulse Devices

SerialPulse Devices consist of a USB-To-Serial plug, available from $1 … $5, connected to any computer running Linux, Windows, Mac, or a Raspberry Pi computer. The RX pin on the UART side of these plugs is fed with pulses 0 … 3.3V or 0 … 5V (other voltages may also work but were not tested).

These pulses may come from Geiger counters, but also from tachometers, distance sensors, and many other devices. Each pulse triggers the sending of a single byte through the serial connection. As the chips in these plugs support baudrates up to Mega-Bits per second, there is no upper limit for possible Geiger count rates.

All what GeigerLog has to do is to count these pulses. And since these devices have their own buffer to store these bytes, GeigerLog can count them by reading at a rather leisurely pace. This also mitigates the problem which could arise from neither Linux, nor Windows, nor Mac being a Real-Time Operating system!

Using a pulse generator I could read data up to a count rate of 10 000 Hz, which is well beyond what any Geiger counter can deliver. The computing load on GeigerLog remains low. So it is also well suited to be used on low-power computers like a Raspi!

There is a huge variety of such USB-To-Serial plugs available, but all are based on one of these chips: CH340, FTDI232, CP2102, PL2303. All of them work, and work the same! I finally used this device [80]) (€3 each) for the convenience of an attached cable.

I ran this **SerialPulse Device** together with the **RaspiPulse Device** on a **Raspi5** computer.

The pulses came from my "Ukraine Counter"[81]). It uses the Russian SBM20 tube. For each Geiger event it produces a 3.3V pulse, active low, length 160µs (roughly the length of the tube deadtime). These pulses were fed simultaneously to both devices.

The outcome is shown in Figure 60. Light Blue is used for RaspiPulse data, and Brown for SerialPulse data. The radioactive source were Uranium glass beads.

Both devices and the Raspi5 passed with flying colors - all outcomes were amazingly consistent, and all Poisson checks were excellent, as the graphics show!

80 https://www.amazon.de/dp/B07Z7PPT6Y

81 The counter was originally developed and shipped from Ukraine, https://de.slideshare.net/IoTdevices/ggreg20-v3-datasheeteng)

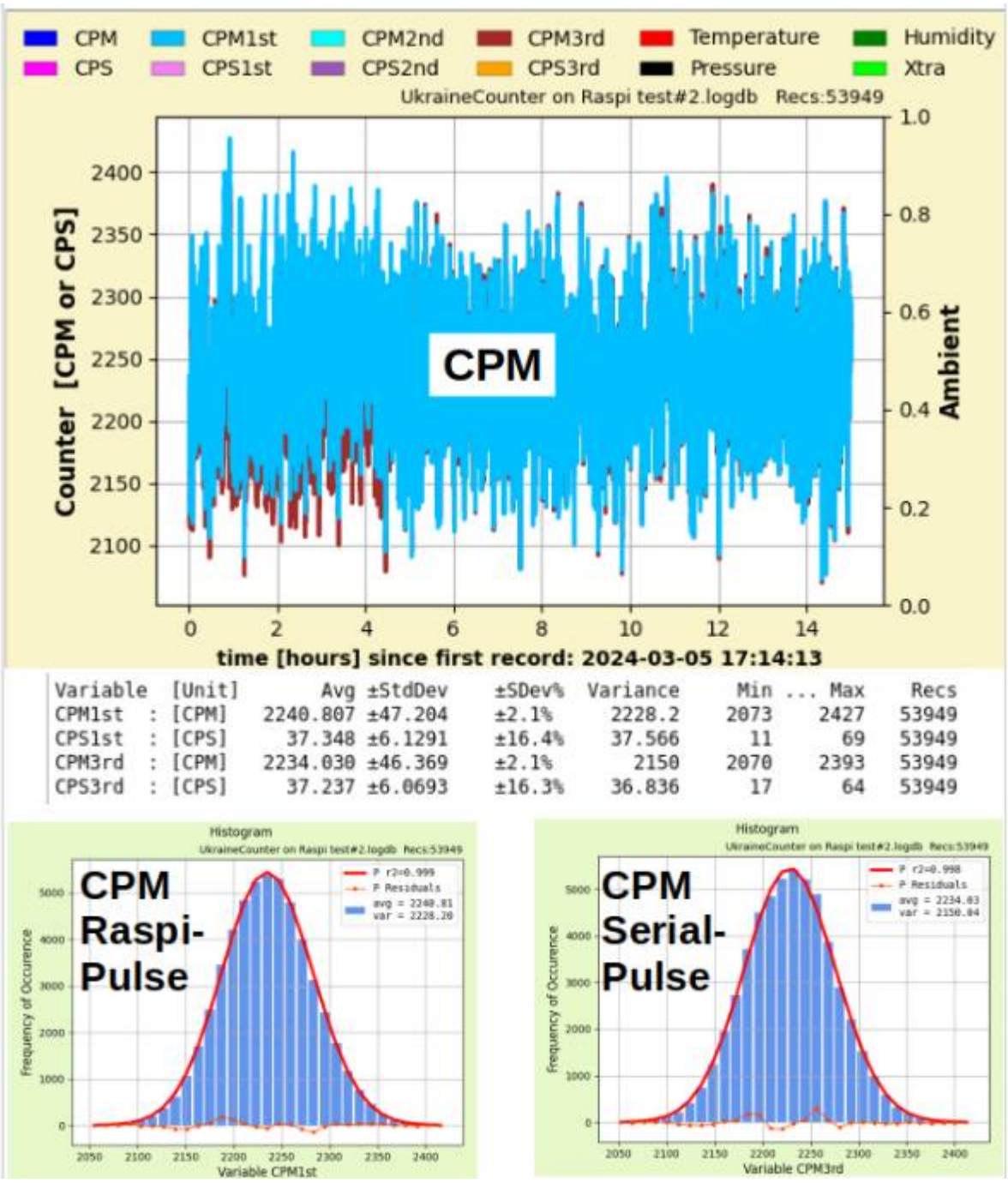


Figure 64: A Raspi5 at the Ukraine Counter with both RaspiPulse and SerialPulse Devices

The GUI – Graphical User Interface

Menus

Menu items may be grayed out when currently not selectable. Some items have keyboard shortcuts in the form of CTRL-X; see the menus for the codes to be used in lieu of the X.

Menu – File

- Clear NotePad — Delete all content from the NotePad
- Search NotePad — Search the NotePad for the occurrence of a text. Also conveniently accessible by CTRL-F!
- Save NotePad to File — Save content of NotePad as text file `<current filename>.notes`
- Print NotePad — Print content of NotePad to a hardware printer or to a pdf file
- Exit — Exit the program (will be prevented if Logging is ongoing; stop Logging first)

Menu – Device

- Connect Devices — Connect computer with devices
- Disconnect Devices — Disconnect computer from devices
- Show Device Mappings — Show the mapping of variables to the activated devices
- Geiger Tubes … — Set sensitivities for all Geiger tubes temporarily

The submenus of the individual device series show up ONLY when these devices are activated in the configuration file `geigerlog.cfg`!

- Submenu: GMC Series — Functions related to GMC Devices
- Submenu: AudioCounter Series — Functions related to AudioCounter Devices
- Submenu: RadMon Series — Functions related to RadMon Devices
- Submenu: AmbioMon Series — Functions related to AmbioMon Devices
- Submenu: Gamma-Scout Series — Functions related to Gamma-Scout devices
- Submenu: I2CSensors Series — Functions related to I2C-Sensor based devices
- Submenu: LabJack Series — Functions related to LabJack based devices
- Submenu: MiniMon Series — Functions related to MiniMon based devices

- Submenu: Simul Series — Functions related to Simul devices
- Submenu: Manu Series — Functions related to Manu devices
- Submenu: WiFiClient Series — Functions related to WiFiClient devices
- Submenu: WiFiServer Series — Functions related to WiFiServer devices
- Submenu: RaspiPulse Series — Functions related to RaspiPulse devices
- Submenu: RaspiI2C Series — Functions related to RaspiI2C devices

Menu – Device – Submenu: GMC Series

- Show Info — Prints some basic info about the device to the NotePad.
- Show Extended Info — Prints extended info about the device to the NotePad; see also Appendix D – The Device Configuration Meanings for some content included in info.
- Set GMC Configuration — Opens a dialogue to set the counter's internal configuration.
- Show Configuration Memory — Prints the device configuration as binary data in human readable format to the NotePad. Also see Appendix D – The GMC Device Configuration Meanings.
- Set Date + Time — Synchronizes computer and device time by setting the device's date and time to the computer time.
- Erase Saved Data ... — Erases the complete content of the History memory.
- Reboot — Reboots the device.
- FACTORYRESET — Does a factory reset. Your device customization is lost, and the internal memory is cleared.
- Set Serial Port ... — Select the Serial Port to be tested for connectability

Menu – Device – Submenu: AudioCounter Series

- Show Info — Prints some basic info about the device to the NotePad.
- Show Extended Info — Additional info explaining how the audio pulses are interpreted
- Plot Pulse — Plot the audio pulse recording from the AudioCounter Device. Makes GeigerLog work like a digital-storage-oscilloscope showing the audio pulses. It provides option to analyze the pulses in the audio recordings

Menu – Device – Submenu: RadMon Series

- Show Info — Prints some basic info about the device to the NotePad.
- Show Extended Info — Additional info on the device

Menu – Device – Submenu: AmbioMon Series

- Show Info — Prints some basic info about the device to the NotePad.
- Show Extended Info — Additional info on the device
- Configure — Configure settings of the AmbioMon device

Menu – Device – Submenu: Gamma-Scout Series

- Show Info — Prints some basic info about the device to the NotePad.
- Show Extended Info — Additional info on the device
- Set Date + Time — Set the Gamma-Scout internal clock to computer time
- Set to Normal Mode — The device can be manually controlled at the device
- Set to PC Mode — The device can be controlled from the computer; required for history downloads and other operations.
- Set to Online Mode — The device allows logging to GeigerLog. The device's log cycle can be set to 2, 10, 30, 60, 120, and 300 seconds.
- Reboot — Execute a reboot (Warm start) of the device.
- Set Serial Port ... — Select the Serial Port to be tested for connectability

Menu – Device – Submenu: I2CSensors Series

- Show Info — Prints some basic info about the device to the NotePad.
- Show Extended Info — Additional info on the device
- Reset System — Reset the I2C ELV dongle and sensors
- Set Serial Port ... — Select the Serial Port to be tested for connectability

Menu – Device – Submenu: LabJack Series

- Show Info — Prints some basic info about the device to the NotePad.
- Show Extended Info — Additional info on the device

Menu – Device – Submenu: MiniMon Series

- Show Info — Prints some basic info about the device to the NotePad.

Menu – Device – Submenu: Formula Series

- Show Info — Prints some basic info about the device to the NotePad.
- Set Properties — Opens a dialogue to set properties of the device

Menu – Device – Submenu: Manu Series

- Show Info — Prints some basic info about the device to the NotePad.
- Enter Values Manually — Opens a dialogue to enter values manually

Menu – Device – Submenu: WiFiClient Series

- Show Info — Prints some basic info about the device to the NotePad.

Menu – Device – Submenu: WiFiServer Series

- Show Info — Prints some basic info about the device to the NotePad.
- Show Extended Info — Additional info on the device
- Ping WiFiServer Device — Sends a “Ping” command to the device
- Set WiFiServer Data Type ... — Opens a dialogue to set data type Last or Avg
- Reset WiFiServer Device — Resetting the device

Menu – Device – Submenu: RaspiPulse Series

- Show Info — Prints some basic info about the device to the NotePad.
- Show Extended Info — Additional info on the device
- Reset RaspiPulse Device — Resetting the device

Menu – Device – Submenu: RaspiI2C Series

- Show Info — Prints some basic info about the device to the NotePad.
- Show Extended Info — Additional info on the device
- Reset RaspiI2C Device — Resetting the device

Menu – Log

- Get Log or Create New One

  Opens a dialog box where you can either select an existing file, or type in a new file name to create a new file. The file will be a database file with the extension *.logdb.

  If you select an existing file, new data will be **appended** to this file!

  After loading a file, it will always be plotted if it contains data, which can be plotted

- Get Log from CSV File

  This allows you to load log-files which were created by older versions of GeigerLog (or by completely different programs, as long as the data are in a CSV format).

  Opens a dialog box where you can select an existing file with the extension *.log or *.csv. Then another dialog box opens, which allows you associate the data columns with the variables in the present GeigerLog version.

  A new database file will the be created.

- Add Comment to Log

  Adds a comment to the log file; does not disturb logging or graphing.

- Set Log Cycle

  The log cycle in seconds can be set in a pop-up window. The cycle time must be at least 0.1s; shorter times cannot be entered

- Start Logging

  Starts logging. Requires that
  1) a connection is made to the device,
  2) the device is powered on, and
  3) a log file is loaded

  The logged values will immediately be saved to the log file, printed to the LogPad, and plotted

- Stop Logging

  Stops logging

- Quick Log

  Start logging using the file default.log. The file will be emptied, before logging starts. If you want to continue logging into a previously selected default.log file, then choose Start Logging instead

- Show Log Data

  Prints the log data to the NotePad.

- Show Log Data Excerpt

  Prints only the first and last few lines of the log, helpful for quick inspection

- Show Log Tags/Comments — Print only records from current log containing tags or comments to the NotePad
- Save Log Data into CSV file — Save all records from current log into a CSV file with extension 'csv'

Menu – History

- Get History from Database — Opens a dialog box and lets you select an existing database file, loads it and plots the data
- Get History from CSV File — Lets you select an existing CSV file – created e.g. by an earlier version of GeigerLog as a '.his' file – and saves it into a database, and then plots the data.
- Add Comment to History — Adds a comment to the history database
- Submenu: GMC Series — History functions related to GMC Devices
- Submenu: AmbioMon Series — History functions related to AmbioMon Devices
- Submenu: Gamma-Scout Series — History functions related to Gamma-Scout Devices

- Show History Data — Print history data as parsed from binary data to the NotePad
- Show History Data Excerpt — Prints the first and last few lines of the parsed data to the NotePad
- Show History Tags/Comments — Prints only those lines from the parsed data to the NotePad, which contain tags or comments. These are mostly Date&Time stamps, but also ASCII tags, which are comments entered directly at the GMC Geiger counter via its Main Menu → Save Data → Note/Location.
- Show History Data with Parse Comments — Show History Data including extended Parse Comments as created by GeigerLog
- Save History Data into CSV file — Save all records from current history into a CSV file with extension 'csv'

Menu – History – Submenu: GMC Series

- Get History from Device — Opens a dialog box where you can select either an existing file, or type in a new name to create a new file. If you select an existing file, this file will be overwritten and its

present content will be lost!

This file is a database file with the extension *.hisdb.

GeigerLog reads the data from the internal memory of the Geiger counter, and stores an exact binary copy in the database.

GeigerLog then parses the binary data and creates a log of the count rates. These data will be plotted.

- Get History from GMC Binary File

  Lets you select an existing binary file – created e.g. by an earlier version of GeigerLog or by a different program – and saves it into a database, parsed the file to create a log of the count rates, and plots the data

- Show History Binary Data Bytecount

  Show counts of bytes in history binary data

- Show History Binary Data Print history binary data in human readable form to the NotePad

- Show History Binary Data Excerpt Prints the first and last few lines of history binary data in human readable form to the NotePad

- Show History Binary Data as AA Map

  Show History Binary Data as a map highlighting the locations of bytes with AA value

- Show History Binary Data as FF Map

  Show History Binary Data as a map highlighting the locations of bytes with FF value

- Save History Binary Data to File Save the history binary data as a *.bin file, i.e. the binary data are extracted from the database and saved into a binary '.bin' file, compatible with earlier versions of GeigerLog.

Menu – History – Submenu: AmbioMon Series

- Get History Binary CAM Data from Device

  Opens a dialog box where you can select either an existing file, or type in a new name to create a new file. If you select an existing file, this file will be overwritten and its present content will be lost!

A database file with the extension *.hisdb will be created.

GeigerLog reads the data from the internal memory of the Geiger counter, and stores an exact binary copy in the database.

GeigerLog then parses the data and creates a log of the count rates. These data will be plotted.

- Get History Binary CPS Data from Device

  Opens a dialog box where you can select either an existing file, or type in a new name to create a new file. If you select an existing file, this file will be overwritten and its present content will be lost!

  A database file with the extension *.hisdb will be created.

  GeigerLog reads the data from the internal memory of the Geiger counter, and stores an exact binary copy in the database.

  GeigerLog then parses the data and creates a log of the count rates. These data will be plotted.

- Get History Binary CAM Data from File

  Getting Data from File instead from Device

- Get History Binary CPS Data from File

  Getting Data from File instead from Device

Menu – History – Submenu: Gamma-Scout Series

- Get History from Device

  Opens a dialog box where you can select either an existing file, or type in a new name to create a new file. If you select an existing file, this file will be overwritten and its present content will be lost!

  This file is a database file with the extension *.hisdb.

  GeigerLog reads the data from the internal memory of the Geiger counter, and stores an exact binary copy in the database.

GeigerLog then parses the data and creates a log of the count rates. These data will be plotted.

- Get History from GS Dat File — Lets you select an existing *.dat file created by Gamma-Scout software as a download of the counter's memory, loads it, and saves it into a database, parses the file to create a log of the count rates, and plots the data.
- Show History Dat Data — Print history Dat data, as they were downloaded from the counter's memory, to the NotePad.
- Save History Data to Dat File — Save the history data as a Gamma-Scout '.dat' file, i.e. the data as downloaded from the counter are saved in the Dat file format.

Menu – Web

- Show IP Status — Show GeigerLog's current IP and Port usage
- Set up Monitor Server … — Set up GeigerLog's Monitor server to allow monitoring GeigerLog from your Smartphone.
- Set up Radiation World Map … — Set up the cycle to upload your data from a selected variable to the Radiation World Maps, see Radiation World Maps on page 138.

Menu – Tools

- Plot Full Log — Re-Plot the full log file data (if loaded)
- Plot Full History — Re-Plot the full history file data (if loaded)
- Show SuSt (Summary Statistics) of Plot Data — Print summary statistics for all data currently shown in the plot to the NotePad
- Show Statistics of Plot Data — In a pop-up window show detailed statistics for the data currently shown and selected in the plot
- Show Scatterplot from Plot Data — Plot one variable against another in a x-y-scatter plot, using only the data currently shown in the plot
- Show Poisson Test of Plot Data — In a pop-up window show a 'Histogram with Poisson Fit' for the data currently shown and selected in the plot
- Show FFT & Autocorrelation of Plot Data — In a pop-up window show a 'FFT & Autocorrelation' analysis of the data currently shown and selected in the plot

• Scaling …	View and Edit the current value- and graph-scaling settings
• Save Graph to File	Save the current graph as a PNG file
• Show Plot Data	Show the Date&Time and variable values as currently shown in the plot
• Display Last Values	Open a window showing all variables and their values
• Fix garbled audio	(Linux Only) Fix for a Pulse-Audio bug resulting in garbled audio.

Menu – Help

• Quickstart	A very short GeigerLog Manual
• GeigerLog Manual	Opens the GeigerLog Manual
• GeigerLog GLrelay Manual	Opens the GeigerLog GLrelay Manual
• GeigerLog Calibration Guidance	Opens the GeigerLog Calibration Guidance
• Devices' Firmware Bugs	Some info on firmware bugs perhaps of relevance to the user, and workarounds
• Radiation World Maps	A brief introduction into the use of the Radiation World Maps
• Occupational Radiation Limits	Info on occupational radiation limits of USA and Germany, and links for extended info.
• Show Command Line Options	Info on options you can start GeigerLog with.
• About GeigerLog	A brief introduction to GeigerLog, as well as version and legal information

Toolbars

The individual toolbars **Main**, **Device**, **Log**, **His(tory), Web,** and **Graph** are combined into a single toolbar, see 65. If preferred, they can be separated and relocated on the screen by grabbing their vertical double-bars on the left and moving them.

Figure 65: The toolbars

Upper-Top : status is non-connected
Lower-Top : status is connected
Bottom : remainder of toolbar

The status shows all devices, activated in the configuration file, while all in green color are connected, and all in red color are not. The latter ones may not be present or had a connection failure.

The meaning of the icons in the toolbars:

- Main - Exit GeigerLog
- Device
  - Symbol ‘plug’: Toggle connection of all activated devices
  - Device buttons are shown here **only if** the respective device is activated in the configuration file `geigerlog.cfg`.
  - Symbol ‘on/off button’: belongs only to the GMC device. Green color indicates that a GMC Geiger counter is powered on (red = power-off)
  - Rectangular buttons indicate activated devices. Color = green: successfully connected, otherwise color= red. The figure shows:
  - A GMC Geiger counter is connected (green color)
  - An AudioCounter device is connected (green color)
  - An I2CSensors device is connected (green color)
  - A RadMon device is activated, but its connection failed (red color)
  - An AmbioMon device is activated, but its connection failed (red color)
  - A Labjack device is connected (green color)
  - Not shown: a Gamma-Scout device, as it had not been activated!

- Log
  - Get a Log File
  - Start Logging
  - Start a Quick Log
  - Snap a single record during logging
  - Stop Logging

- History
  - Get a History file from database

- Web
  - Show IP Status
  - Setup Monitor Server
  - Setup Radiation World Map, see Radiation World Maps on page 138.

- Graph
  - Reset original view
  - Back to previous view
  - Forward to next view
  - Pan axes with left mouse, zoom with right
  - Zoom to rectangle
  - Configure subplots
  - Save the figure
  - Edit curves, line, and axes parameters

Miscellaneous

Running GeigerLog on HiDPI Monitors

HiDPI monitors are displays with a pixel resolution higher than FullHD. FullHD typically means a pixel resolution of 1920 x 1080 (width x height), or “2k”, HiDPI monitors are typically labeled as “4k” or “8k”, meaning a width of approx. 4000 or 8000 pixels, resp. There is no fixed norm for either label, so 4k could mean e.g. 3840 x 2160, but also 4096 x 2304, and 8k could mean 7680 x 4320, or other “nearby” sizes.

The support of those HiDPI monitors by operating systems is still rudimentary and surely inconsistent, which is easily seen by the fact that some software output becomes too small to read, and requires to set “scaling” - if it exists. This triggers a different problem if part of the software supports scaling, and part not, where now the output may become too large to fit on the monitor.

GeigerLog allows to switch its adaptation to HiDPI on and off, and allows some tuning. You find the configuration file `geigerlog.cfg` in the section “Window”.

HiDPIactivation: this is set to “yes” by default. This should be ok for almost all situations, except that you must set this to “no” when GeigerLog is run on a Raspi 4 (status August 2020).

HiDPIscaleMPL: This compensates any HiDPI scaling for the graphs. For a FullHD monitor the 'scaling' set in the operating system is typically 100%. Any different scaling must be compensated.

Try it with the “auto” setting first. If this gives no good result, read the tuning instructions in the configuration file.

On an 8k monitor, with scaling set to 200%, HiDPI activated in GeigerLog, and the HiDPIscaleMPL set to 50, the screenshot in Figure 66 shows the excellent result:

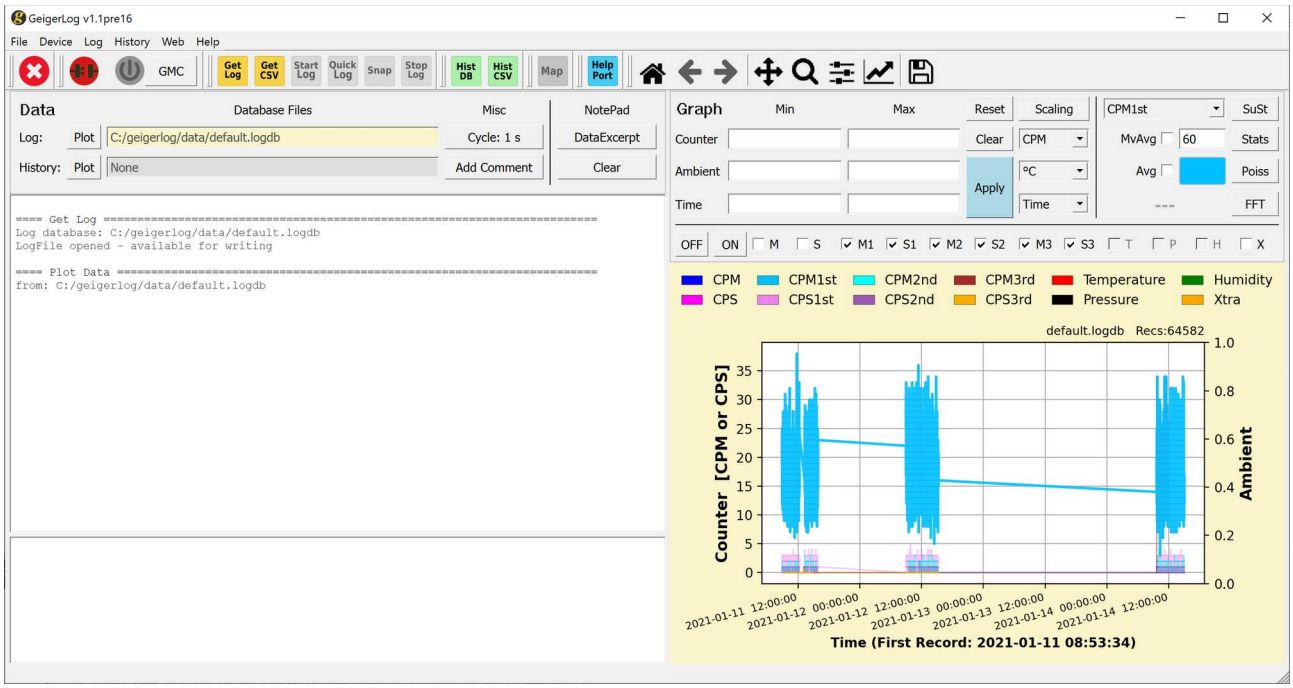


Figure 66: GeigerLog running on an 8k HiDPI monitor

Starting GeigerLog with Options

You can start GeigerLog with options. Otherwise you might prefer to customize the configuration file `geigerlog.cfg`. To see the available options, click menu **Help → Show Command Line Options** for an output as shown in Figure 67. Alternatively, start with

on Linux, Mac: `./GeigerLog.sh -h`
on Windows: `GeigerLog.bat -h`

NOTE: For Auto-Starting GeigerLog using Command Line Options see chapter Auto-Starting GeigerLog on page 48.

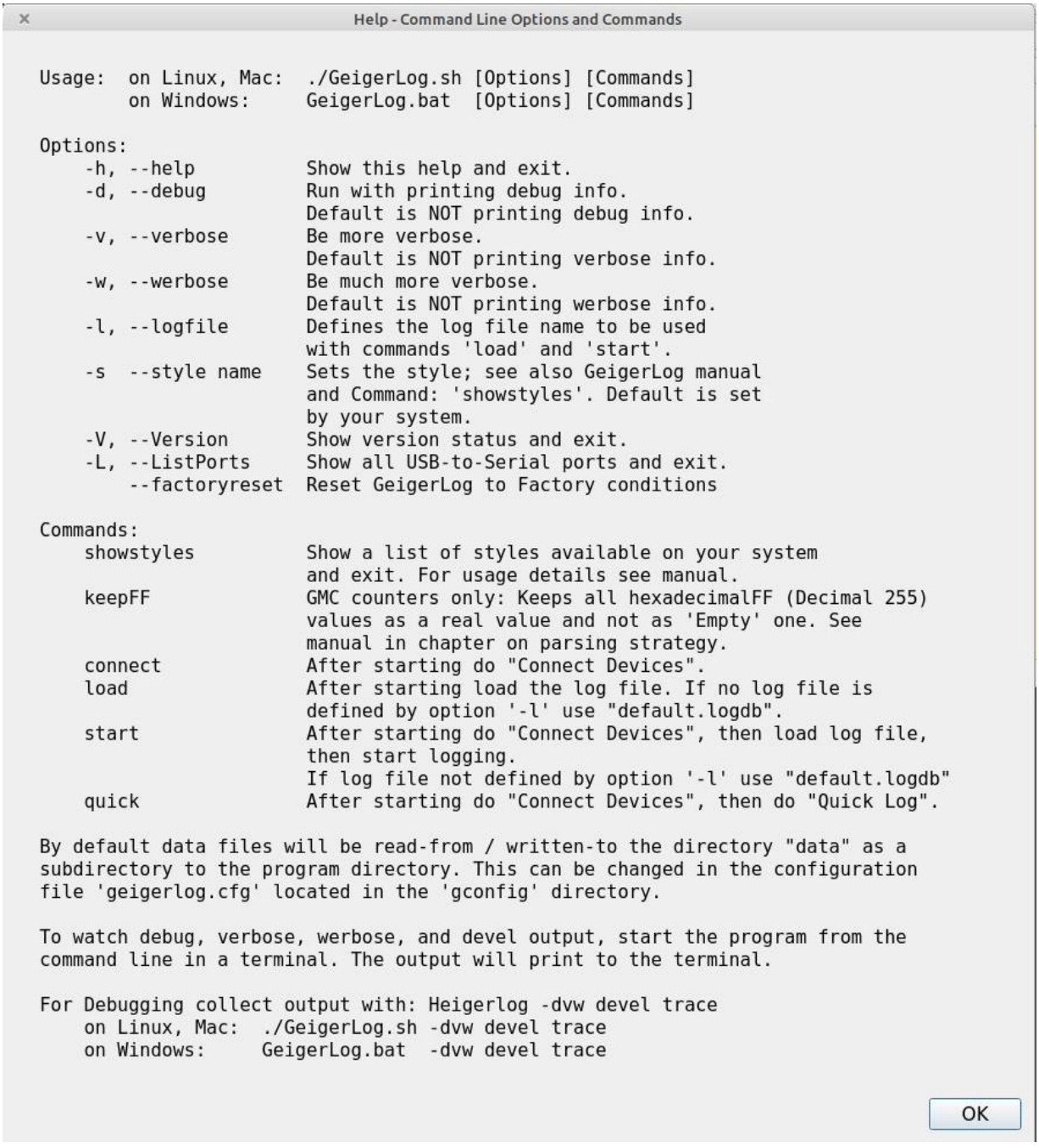


Figure 67: Command Line Options

Helpful Internal Software Tools

These internal tools come as part of GeigerLog and can be found in the GeigerLog **`gtools`** directory. None is required to run GeigerLog, but sometimes can be really helpful.

GeigerLog PipCheck

It tells you which packages required by GeigerLog are installed or are missing, and which ones have upgrades available. Very helpful if there are any problems during the installation. Change into the **`gtools`** directory, and start with:

on Linux, Mac: `./PipCheck.sh`
on Windows: `PipCheck.bat`

for output like this (note Missing and upgradeable packages):

```
---------------------- GLpipcheck.py ----------------------------------------
GLpipcheck Version:  1.2
Python Executable:   /home/ullix/geigerlog/vgl310/bin/python3
Python Version:      3.10.0a6 (default, Mar  2 2021, 02:01:08) [GCC 5.4.0 20160609]

Listing of all Pip-found packages:
   Package             Version
   ------------------- ------------
   APScheduler         3.6.3
   cachetools          4.2.2

   ... lines removed ...

   SoundFile           0.10.3.post1
   terminaltables      3.1.0
   tornado             6.1
   tzlocal             3.0

GeigerLog REQUIRED packages and their version status
   pip                 21.3.1            OK
   setuptools          58.5.3            OK
   PyQt5               5.15.6            OK
   PyQt5-sip           12.9.0            OK
   numpy               1.21.3            OK
   scipy               1.7.2             OK
   matplotlib          3.4.3             OK
   sounddevice                           MISSING
   SoundFile           0.10.3.post1      OK

GeigerLog OPTIONAL packages and their version status
   pyserial            3.5               OK   REQUIRED for GMC, Gamma-Scout, I2C Series
   paho-mqtt           1.6.0             OK   REQUIRED for RadMon Series
   LabJackPython       2.0.4             OK   REQUIRED for LabJack Series
   pip-check           2.6               OK   Recommended Pip tool

Checking for updates ... Done

GeigerLog REQUIRED packages having upgrades available:
   Package             Version           Latest     Type
   numpy               1.21.3            1.21.4     wheel

GeigerLog OPTIONAL packages having upgrades available:
   Package             Version           Latest     Type
   paho-mqtt           1.6.0             1.6.1      sdist
```

```
Mini-HOWTO:
-------------------------------------------------------------------------------
for using 'pip':
NOTE:   when 'pip' itself or 'setuptools' can be upgraded, upgrade 'pip' first, then
        'setuptools', then any others
NOTE:   when using '=', '==', '>', '<' for version definition you need to use quotation
        marks around package designation
                    like: "anypackage == 1.2.3",  "anypackage >=1, <2"

Install and/or update package 'anypackage':
                    python3 -m pip install -U anypackage

Install and/or update multiple packages at once:
                    python3 -m pip install -U anypackage 2ndpackage 3rdpackage ...
        NOTE:
        If command fails, nothing at all may have been installed!
        Continue with single-package installations.

Install package 'anypackage' in exactly version 1.2.3:
                    python3 -m pip install "anypackage == 1.2.3"

Install package 'anypackage' in version of at least 1 but not 2 or later:
                    python3 -m pip install "anypackage >=1, <2"

Remove  package 'anypackage':
                    python3 -m pip uninstall anypackage

Find    all available versions of package 'anypackage':
                    python3 -m pip install anypackage==
```

If you want to see which modules have updates available, start with command ‘latest’ on the command line, like: '**./PipCheck.sh latest**'. However, it is **NOT** recommended to update beyond the GeigerLog minimal requirements, as those latest versions have not been tested, and installing them might result in GeigerLog NOT WORKING !

If this happens, then you need to manually install the GeigerLog approved versions! Or, perhaps easier, do a fresh install of the complete GeigerLog!

GeigerLog Audio Check – GLaudiocheck.py

GLaudiocheck.py allows to check the audio signal for use as an AudioCounter input by plotting a live microphone signal. Very helpful for inspecting the audio input and setting the audio amplification.

Start with: `python /path/to/GLaudiocheck.py`

Stop with: `CTRL-C`

Its graphic output may look like in Figure 68 (a 20 sec recording sampled with 44100 samples per second for an audio signal with negative pulses):

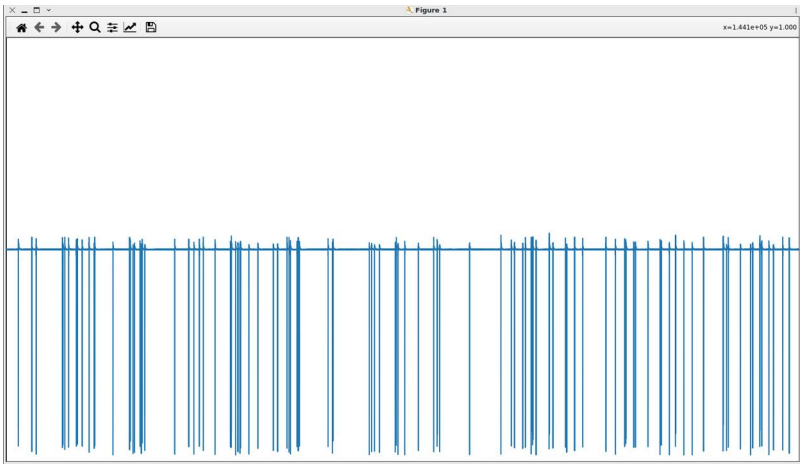


Figure 68: Output from the GLaudiocheck.py for an audio source with negative pulses

GeigerLog CSV File Merger – GLmerger.py

It allows to merge two GeigerLog created CSV files into one. It assures that the resulting CSV file has all variables set. Any records with the same DateTime stamp will be merged into a single record.

Start with:

```
python /path/to/Glmerger.py File#1 File#2 mergedfile
```

for output like this:

```
# File#1
#   Index,            DateTime, CPM,    CPS,    Temp,   Press,   Humid
        6, 2021-11-11 10:34:55,  91,    2.0,     3.0,     4.0,     5.0
        7, 2021-11-11 10:34:56, 106,    2.0,     3.0,     4.0,     5.0
        8, 2021-11-11 10:34:57,  99,    2.0,     3.0,     4.0,     5.0
        9, 2021-11-11 10:34:58, 108,    2.0,     3.0,     4.0,     5.0
       10, 2021-11-11 10:34:59,  92,    2.0,     3.0,     4.0,     5.0
       11, 2021-11-11 10:35:00,  99,    2.0,     3.0,     4.0,     5.0

#File#2
#   Index,            DateTime,  CPM1st,  CPS1st,  CPM2nd,  CPS2nd
        6, 2021-11-11 10:34:55,    43.0,     3.0,     0.0,     0.0
        7, 2021-11-11 10:34:56,    43.0,     0.0,     0.0,     0.0
        8, 2021-11-11 10:34:58,    42.0,     0.0,     0.0,     0.0
        9, 2021-11-11 10:34:58,    43.0,     1.0,     0.0,     0.0
       10, 2021-11-11 10:34:59,    44.0,     1.0,     0.0,     0.0
       11, 2021-11-11 10:35:00,    45.0,     2.0,     0.0,     0.0

#mergedfile
          DateTime,CPM,CPS,CPM1st,CPS1st,CPM2nd,CPS2nd,CPM3rd,CPS3rd,  T,   P,   H,  X
2021-11-11 10:34:55, 91,2.0,  43.0,   3.0,   0.0,   0.0,      ,      ,3.0, 4.0, 5.0,
2021-11-11 10:34:56,106,2.0,  43.0,   0.0,   0.0,   0.0,      ,      ,3.0, 4.0, 5.0,
2021-11-11 10:34:57, 99,2.0,      ,      ,      ,      ,      ,      ,3.0, 4.0, 5.0,
2021-11-11 10:34:58,108,2.0,  42.0,   0.0,   0.0,   0.0,      ,      ,3.0, 4.0, 5.0,
2021-11-11 10:34:59, 92,2.0,  43.0,   1.0,   0.0,   0.0,      ,      ,3.0, 4.0, 5.0,
2021-11-11 10:35:00, 99,2.0,  44.0,   1.0,   0.0,   0.0,      ,      ,3.0, 4.0, 5.0,
2021-11-11 10:35:01, 98,2.0,  45.0,   2.0,   0.0,   0.0,      ,      ,3.0, 4.0, 5.0,
```

You see that the 3rd row, marked in red, is the only one with a time stamp occurring only in the File#1.

Obviously, if any variable is present in both files, then one would be lost in the merging process!

GeigerLog Serial Monitor - GLsermon.py

GLsermon.py allows to send commands and receive responses via any Serial Connection.

It supports any device, but it has very helpful extra support for the GMC counter series, which may need **binary coded** commands.

Start with:

to get Help:	`GLsermon.py -h`
to set Port and Baudrate:	`GLsermon.py -P <Port> -B <Baudrate>`
specific for **GMC** counter:	`GLsermon.py -P <Port> -B <Baudrate> gmc`

Stop with: `CTRL-C`

A <Port> on Linux is: **/dev/ttyUSBx**, x = 0, 1, 2, ...
on Windows is: **COMx**, x = 0, 1, 2, ...
The <Baudrate> is: 57600, 115200, …

In the examples below you see that the simple **getcps** command gets a 4-byte answer (not 2-byte!), and that the quite complex **SPIR** command, requiring 5 binary coded bytes, can be send with ease.

A communication session with GLsermon.py connected to a GMC-500+ Geiger counter may look like this: (both input and output will be recorded to file `GLsermon.log`; you'll find it in the directory where GLsermon.py was started from.)

getver

```
>>>>> NOTE: Byte values printed to a Terminal may look incomplete! Compare
Length with expected length! <<<<<
Sending: '<GETVER>>'  (Length:9)

2021-11-08 18:30:19.775 :
Receive-Bytes:  15: GMC-500+Re 1.22
Receive-Values HEX:  47  4D  43  2D  35  30  30  2B  52  65  20  31  2E  32  32
Receive-Values DEC:  71  77  67  45  53  48  48  43  82 101  32  49  46  50  50
Receive-Values ASC:   G   M   C   -   5   0   0   +   R   e       1   .   2   2
```

getvolt

```
>>>>> NOTE: Byte values printed to a Terminal may look incomplete! Compare
Length with expected length! <<<<<
Sending: '<GETVOLT>>'  (Length:10)

2021-11-08 18:31:36.324 :
Receive-Bytes:   5: 4.9v
Receive-Values HEX:  34  2E  39  76  00
Receive-Values DEC:  52  46  57 118   0
Receive-Values ASC:   4   .   9   v
```

getcps

>>>>> NOTE: Byte values printed to a Terminal may look incomplete! Compare Length with expected length! <<<<<

Sending: '<GETCPS>>' (Length:9)

2021-11-08 18:32:38.104 :

Receive-Bytes: 4:

Receive-Values HEX: 00 00 00 00

Receive-Values DEC: 0 0 0 0

Receive-Values ASC:

spir,0,0,0,1,0

>>>>> NOTE: Byte values printed to a Terminal may look incomplete! Compare Length with expected length! <<<<<

Sending: '<SPIR#>>' (Length:12)

(Binary values ‘0’, ‘1’ are not printable, and do not show up inside ‘<SPIR#>>’)

2021-11-08 18:33:33.765 :

Receive-Bytes: 256: (Non-decodeable)

Receive-Values HEX:

```
55 AA 00 00 00 00 00 00 00 55 AA 01 55 AA 00 00 00 00 00 00 00 55 AA 01 55 AA 00 00 00 00 00 00
00 55 AA 01 01 05 00 00 00 02 00 01 00 02 00 01 00 00 00 00 02 00 00 00 00 00 00 00 01 00 00 00
00 00 02 00 00 00 01 01 00 00 01 00 00 00 01 00 00 00 00 02 00 01 00 01 00 01 01 01 01 01 01 00
00 00 00 00 00 00 01 01 00 00 00 00 00 01 00 00 00 00 00 01 00 00 00 00 00 00 00 00 00 00 00 00
```

(more data; not shown)

Helpful External Software Tools

None is required to run GeigerLog, but sometimes these tools can be really helpful.

Editor VSCode

The Microsoft Visual Studio Code editor **vscode** is available on **Linux**, **Windows**, and **Mac**. In combination with a **Python extension** (https://code.visualstudio.com/docs/languages/python) it is a very powerful, professional editor for Python. Other extensions exist for other languages, e.g. C, C++. Unfortunately, vscode has a bit of a steep learning curve.

This is the new development standard for GeigerLog.

Get it from https://code.visualstudio.com/.

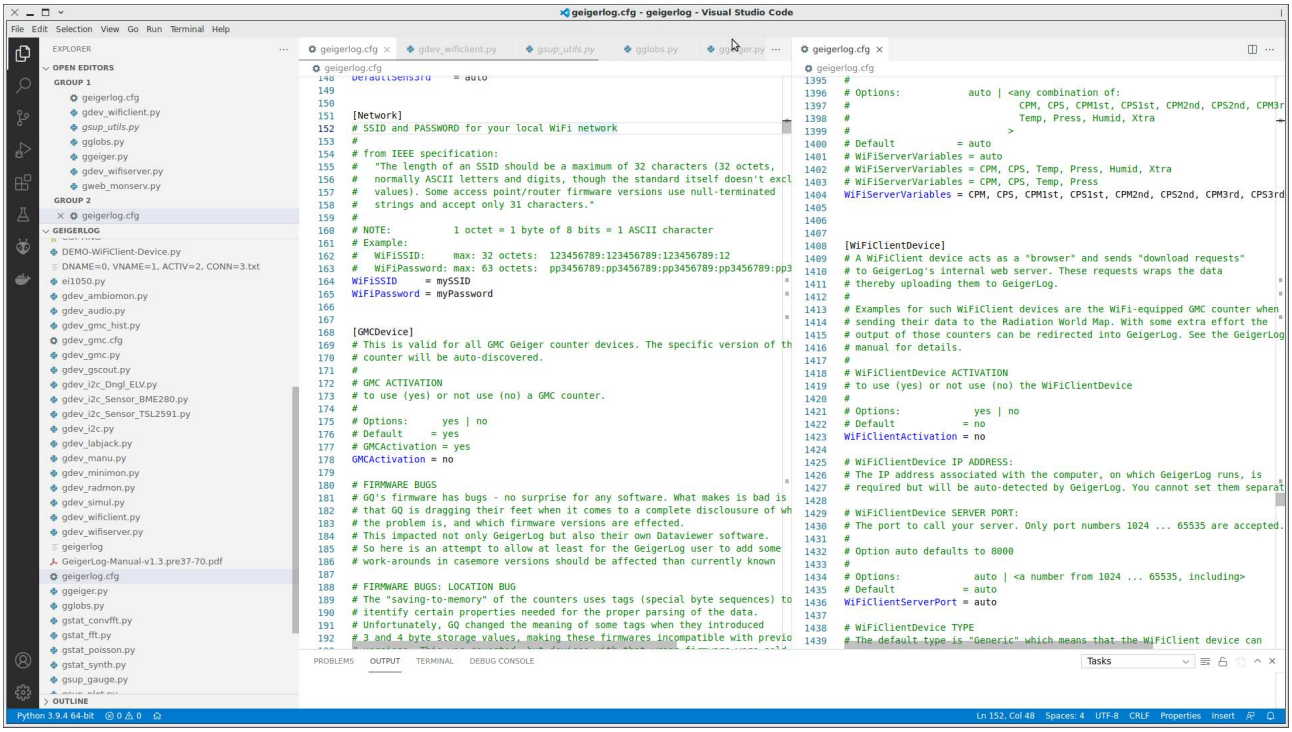


Figure 69: Editor Visual Studio code with file geigerlog.cfg opened in Split-Editor mode

Editor Geany

Even if you don't want to edit the program code, you need a proper editor to adapt e.g. the Geiger-Log configuration file to your needs, without messing with the line endings, which some Windows editors like to do.

GeigerLog had in the past been developed on the editor **Geany**, which I do recommend as a general purpose editor.

Get Geany from: https://www.geany.org/Download/Releases .

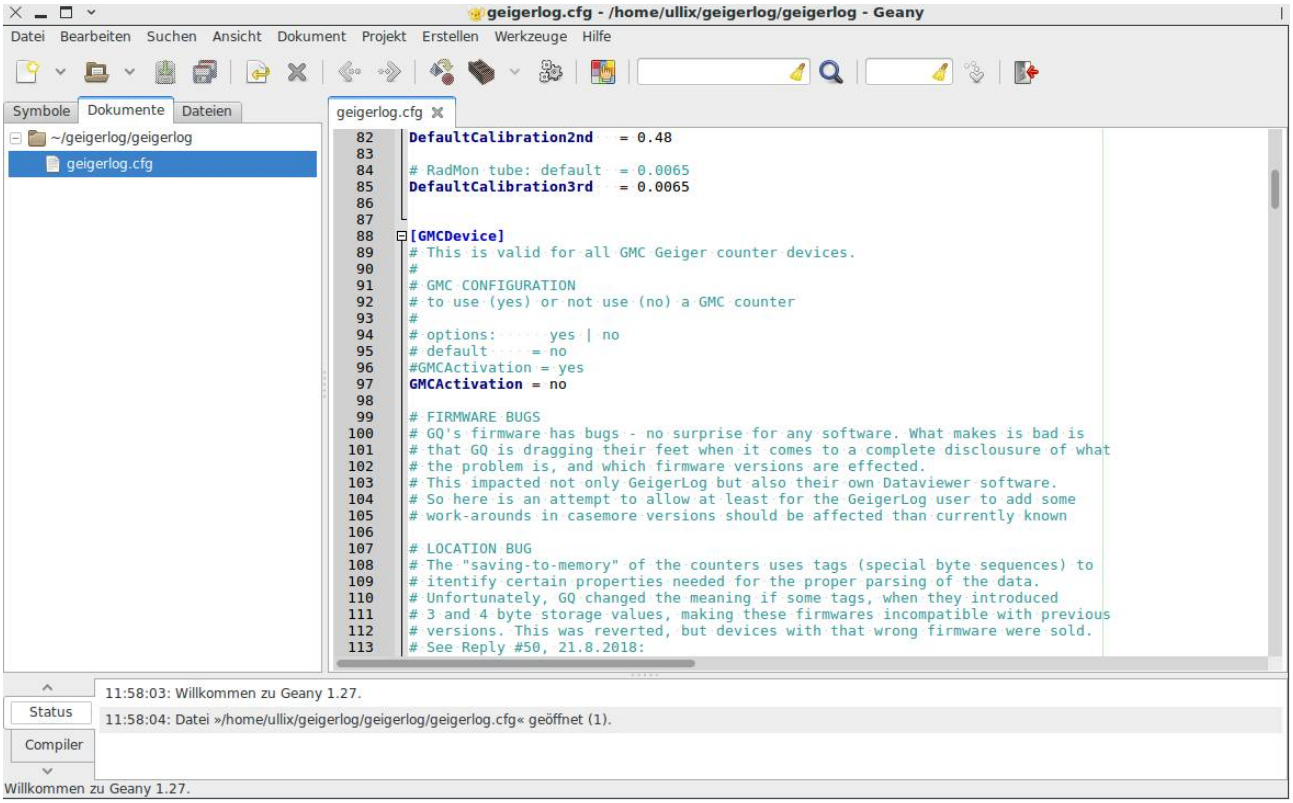


Figure 70: Editor Geany with file geigerlog.cfg opened

DB Browser for SQLite

As the Log and History files are saved as database files, you need a tool for inspecting and perhaps editing those files. The software **DB Browser for SQLite** is Open source and available for **Linux**, **Windows**, and **Mac**, and is an excellent tool for SQLite3 databases.

Download from https://sqlitebrowser.org/ .

Viewing a Log database file in this browser may look like this:

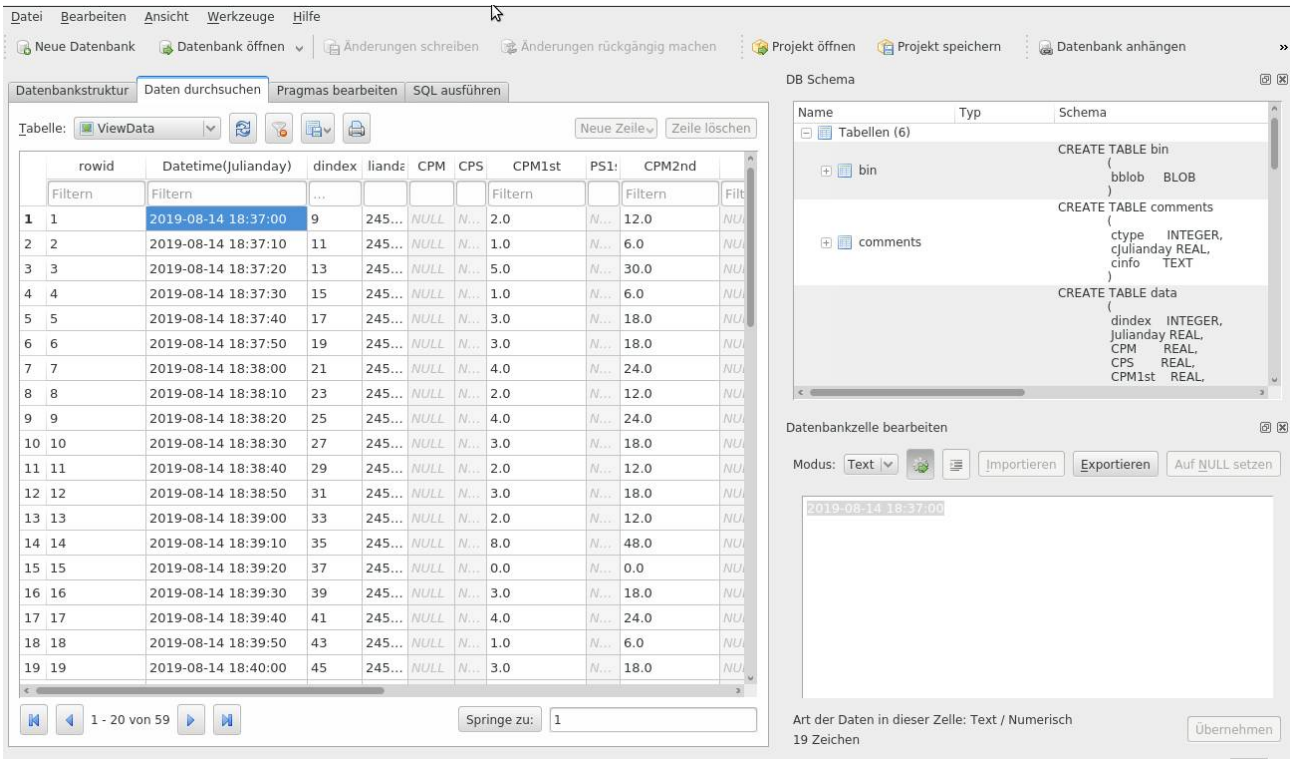


Figure 71: DB Browser for SQLite as tool to inspect Log files

Radiation World Maps

Several web sites exist which attempt to show a worldwide map of the **BACKGROUND** radioactivity, hoping to be of help to the people in case of a nuclear emergency, which will result in elevated levels of radioactivity. Some are run by governments, others by enthusiastic hobbyists.

Among the latter ones are:

- gmcmap.com – This is the one supported by GQ Electronics
- radmon.org – Has well recovered from a hacking attack
- safecast.org – Accepting radiation as well as air quality data

Currently only GQ's **GMCmap** is supported by GeigerLog; others may follow.

GQ suggests to use your WiFi equipped Geiger counter (series GMC-320+V5, GMC-500, GMC-600) to directly update their website. This is actually not such a good idea, see below.

Using GeigerLog to update the gmcmap.com site

You can support GQ's **GMCmap** using GeigerLog, and not only provide more meaningful data, but use any of their non-WiFi counters just as well. Actually, you can even use any other counter as long as it can be connected to GeigerLog.

This includes digitally connected counters, like e.g. RadMon, GammaScout, and analog-connected counters like the whole world of only-audio-click counters, including truly classic counters, the latest low-cost counters, or even the most modern solid state detectors using PIN photo-diodes [82]).

How to set this up is explained in chapter Radiation World Map on page 36.

But there are several problems associated with the **GMCmap** site in particular.

Some words of caution

CPM: The property depicted on the **GMCmap** site is CPM, which is the worst possible base on which to compare different counters, which may have different tubes and even different numbers of tubes, and therefore needing totally different factors to translate from CPM to a dose rate measured in µSv/h. This is like a worldwide reporting of temperatures as either Fahrenheit, or Celsius, or Réaumur but only giving the numbers and not telling which units are used! The only meaningful basis for comparisons is the dose rate based on units of Sievert (Sv) per time interval (Sv/h, mSv/a , µSv/h, or nSv/h).

Quality Control: As far as I can see there is no quality control of the data! Nothing prevents users from putting a strong radioactive source in front of their detector, and pushing these data to the web. In fact, you don't even need a counter, and don't even need GeigerLog, but can enter any data you wish manually! I don't want to mess with GQ's map, so I haven't tried to enter

82 See chapter AudioCounter Devices on page 70, and Review Geiger Counter "Smart Geiger Pro (SGP-001)" from https://sourceforge.net/projects/geigerlog/files/Articles/GeigerLog-Review%20Smart%20Geiger%20Pro%20%28SGP-001%29-v.1.0.pdf/download

things like CPM=9999. But if you did something like that inadvertently you would discover that there does not seem to be a way to retract any such wrongly sent data.

Poor data will quickly destroy any value of those sites.

Instantaneous CPM: It is a bit more subtle, but diminishes the data quality nevertheless. GQ's potential upload is: CPM, ACPM (Average CPM), µSv/h reading. The latter two are optional. CPM is the instantaneous CPM of the counter.

Unfortunately, Geiger counter readings fluctuate quite significantly. Thus when individual, single readings are posted, the values may be significantly higher or lower than the average, suggesting changes that don't exist. The fluctuation is largest at low count rates [83]), hence the reports of background rates are the most impacted: for a CPM=20 average background, 5% of the values can be expected to be greater than CPM=28 or smaller than CPM=10. That already is almost a 3fold difference!

Furthermore, the ACPM is averaged since the last cold start or reset of the counter. So, this value becomes less and less sensitive to any changes of the count rate! If the counter were running for a very long time, the ACPM value would not not even respond to catastrophic changes!

GeigerLog will send CPM values always as averages, but allows the user to determine the length of the period, over which data are averaged. I suggest to have values collected for at least 30 minutes, more is better, before sending anything to the maps. Daily updates are quite appropriate!

83 it decreases with 1/SQRT(count rate), see Poisson Distribution in GeigerLog - Potty Training for Your Geiger Counter

Occupational Radiation Limits

Available in menu **Help → Occupational Radiation Limits**. The exposure to radiation is strongly regulated all over the world. It is quite interesting to compare regulations in different countries. As examples, the occupational limits are given for USA and Germany.

"Occupational" refers to people working in fields with typically higher exposure to radiation compared to the average person, like medical people applying X-rays, workers in nuclear power plants, people in aviation, people in mining.

Of the many limits specified, only the yearly and lifelong exposures are given here; the links will guide you to sites with a lot more extensive specifications.

	Germany	**USA**	**USA : Ger**
Yearly exposure	20 mSv	50 mSv	2.5 x
equiv. to permanent:	2.3 µSv/h	5.7 µSv/h	
Lifelong exposure	400 mSv	2350 mSv	5.9 x
equiv. to permanent: (assuming in 40 years of work)	1.14 µSv/h	6.71 µSv/h	
	Grenzwerte	OSHA	

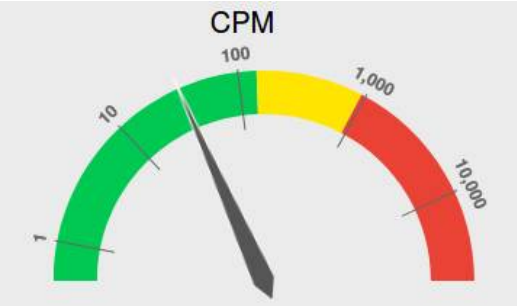
The differences are astonishing, to say the least! Over one year the US citizens are considered 2.5 times “tougher” than the Germans? And over the lifetime even 5.9 times tougher?

An even more intricate difference becomes visible when one computes dose rate per hour for yearly, lifelong and for each country: the German limit sinks twofold, while the US limit increases approx. 20%? Say what, the Germans become feebler as they age and need to be more protected, while the Americans grow even sturdier and take the extra?

This indicates a fundamentally different view between the two countries on how radiation effects health. I can’t imagine that this has anything to do with Science. It is a legal thing. Everything seems possible when facts don’t matter!

On what grounds do we set the radiation safety levels?

A gauge as shown on the right is such as easy, universally understood symbol for safety warnings with green for ok, yellow for caution, and red for danger.



But at what levels do we set the transition between the zones, from green to yellow, and from yellow to red?

With the legal situation of little help (see last chapter) from where do we take guidance?

The “natural background” could be considered a marker for a safe level – if it weren’t, mankind would not exist. The global average is 0.27 µSv/h [84]). The average dose of Finland is more than 3 times higher at 0.9 µSv/h. Even higher levels are found in the town of Ramsar (Iran), which has levels of 1.2 µSv/h average, at some locations in town even up to 15 µSv/h [85]). People live there as healthy as elsewhere. With this being a more than 50fold difference, it raises the question: which level is then “safe”?

The natural background results from the combined effects of radiation originating from the earth and from the cosmic radiation. The latter increases with altitude and at commercial aviation altitudes is much higher than the ground background. The cosmic radiation is lowest near the equator, and increases towards the poles. Thus your exposure depends on where you fly. Overall, the radiation exposure of air crews will be much higher than of other people, and the health of air crews could be an indicator for safe levels.

The US FAA has released a report in 1990 which said that cosmic radiation for domestic air travel is 6 µSv/h [86]). In a recent scientific publication [87]) an estimate was given for the average effective dose over the lifetime of aircrew at 5 … 6 µSv/h. So both estimates are close to the USA regulatory lifetime limit.

I decided for these values:

- Low = 0.9 µSv/h – transition green to yellow (Finland average)
- High = 6.0 µSv/h – transition yellow to red (Air crew and US regulatory)

These numbers nevertheless are arbitrary. They can be changed in the GeigerLog configuration file `geigerlog.cfg`, in segment DOSERATE THRESHOLDS.

These numbers are also exaggerated by the fact that they are for lifetime-long permanent exposures, but the situation for anyone experimenting with a Geiger counter is that there would at worst be only a short term higher dose exposure.

How to account for that, I don’t know. But I think there is plenty of leeway!

84 Numbers from https://en.wikipedia.org/wiki/Sievert unless otherwise stated

85 https://en.wikipedia.org/wiki/Ramsar,_Mazandaran

86 http://www2.ans.org/pubs/magazines/nn/pdfs/2000-1-3.pdf

87 From: https://academic.oup.com/jrr/article/59/suppl_2/ii1/4844965
Typical doses and dose rates in studies pertinent to radiation risk inference at low doses and low dose rates, Rühm et al., *Journal of Radiation Research*, Volume 59, Issue suppl_2, April 2018, Pages ii1–ii10, https://doi.org/10.1093/jrr/rrx093
Quote: **Air crew:** In Germany, for example, the average annual occupational effective dose to air crew was 1.9 mSv, in 2012 [28]. Maximum allowed flight hours for air crew are typically ~900 h per year, and some individuals accumulate up to 5 mSv per year or more. Assuming a maximum of ~40 working years, cumulative career doses of up to 200 mSv can therefore not be excluded. In such cases, mean effective dose rates of ~5–6 µSv/h can accumulate, which is consistent with typical dose rates at flight altitudes from secondary cosmic radiation of 2–7 µSv/h depending on altitude, latitude, and solar activity [29, 30].

Problems, Bugs, and Bug Reports

A) GeigerLog does not even start

It is best to completely re-install and setup GeigerLog. Watch the terminal for any output that may be relevant for identifying the cause of the initial problem. Make a Bug-Report on this issue if you need help.

B) GeigerLog does start Ok, but then exhibits problems

Change into GeigerLog's working directory and start GeigerLog from a terminal / command window with the option '**`-dvw devel`**', like:

on Linux, Mac: **`./GeigerLog.sh -dvw devel`**
on Windows: **`GeigerLog.bat -dvw devel`**

and look for any message during starting that may identify the problem. Start logging with command "Quick Log". Make a Bug-Report on this issue if you need help.

C) Making a Bug Report

If all attempts fail, and the problem persists, you may have found a bug. Then please file a bug report by opening a new Bug-Topic in the GeigerLog SourceForge site under:

https://sourceforge.net/p/geigerlog/discussion/general/.

If you were able to start GeigerLog as explained in **B)** it will have created a file **`geigerlog.proglog.devel`** in the `data` directory as well as a **default.logdb** database file. These two files need to be attached to your Bug Report.

References

Geiger-Müller tubes - Introduction, Centronic ISS.1 (further details unknown).
https://sourceforge.net/projects/geigerlog/files/Supporting%20Documents/
Originally downloaded April 2017 from:
https://SourceForge.net/projects/gqgmc/files/gqgmc/Geiger_Tube_theory.pdf/download
also available here: https://qa.ff.up.pt/rq2020/Bibliografia/etc/geiger_tube_theory.pdf
and other places, google for: “geiger tube theory.pdf”

Accurate Determination of the Deadtime and Recovery Characteristics of Geiger-Muller Counters, Louis Costrel, U.S. Department of Commerce, National Bureau of Standards, Research Paper RP1965, Volume 42, March 1949, Part of the Journal of Research of the National Bureau of Standards, http://nvlpubs.nist.gov/nistpubs/jres/42/jresv42n3p241_A1b.pdf

GQGMC, Documentation by Phil Gillaspy
https://sourceforge.net/projects/gqgmc

GQ-RFC1201, GQ Geiger Counter Communication Protocol, Ver 1.40, Jan-2015,
by GQ Electronics LLC, https://www.gqelectronicsllc.com/download/GQ-RFC1201.txt

GeigerLog - Potty Training for Your Geiger Counter, by ullix,
find it in the “Articles” section of: https://SourceForge.net/projects/geigerlog/

GeigerLog - Going Banana, by ullix,
find it in the “Articles” section of: https://SourceForge.net/projects/geigerlog/

Other websites offering software for Geiger counters

https://github.com/Wikilicious/pygmc

PyGMC is a Python API for Geiger-Müller Counters / Geiger Counters. It has just one dependency (pyserial) and works on multiple operating systems: Windows, OSX, Linux. PyGMC aims to be a minimalist interface - lowering the installation requirements and allowing the user to build their own tools on top of a stable package.

Appendix A – Look & Feel

GeigerLog uses some resources which exist on your computer independently from GeigerLog. This is mainly the “style”, but also the “fonts” available on a system. Both largely determine the Look & Feel of a software.

They may differ between computers even on the same version of the operating system!

Style

Generally the default style will be ok (and it is markedly improved in the current PyQt5 toolkit over the previous PyQt4 toolkit). But if it doesn’t please you, select a different one. To get a list of styles available on your computer, start GeigerLog with:

on Linux, Mac: **`./GeigerLog.sh showstyles`**
on Windows: **`GeigerLog.bat showstyles`**

Your output should be similar to this [88]):

```
Linux:                Windows, Fusion
Windows:              WindowsVista, Windows, WindowsXP, Fusion
```

To use a style, let’s say `'Windows'`, start GeigerLog like this:

on Linux, Mac: **`./GeigerLog.sh -s Windows`**
on Windows: **`GeigerLog.bat -s Windows`**

Exact spelling & capitalization of the style name is required!

Fonts

GeigerLog will select suitable fonts; they cannot be selected by the user.

88 The previously possible styles Breeze, Cleanlooks, Plastique, Windows, GTK+, and other, are no longer available under the modern PyQt5 toolkit in default installations

Appendix B – Connecting Device and Computer using a Serial Connection

NOTE: GeigerLog can auto-detect the presence of a serially connected, supported GeigerLog device, and chose all proper settings, provided the device is **activated** in the configuration file **`geigerlog.cfg`**.

All current **GMC** counters, **GammaScout** counters, **RadPro** counters, and **I2C** devices are fully **plug-and-play**. Any remaining user options are found in the chapters describing the devices.

NOTE to Linux, Mac users: Depending on your computer's settings you may still have to deal with the proper permissions for the Serial Port; see chapter: Appendix C – HOWTO deal with read and write permissions for the serial port when on Linux on page 147

BACKGROUND:

While device and computer are connected with a USB cable, their connection is actually only a classic serial connection, same as in the good old days of the teletype. The translation between USB and serial is done by an USB-To-Serial chip embedded in the electronics of the external device.

And while serial connections today are faster than decades ago, they are slow by today's standards. In the **GMC** and **I2C** devices the serial speed is in the order of 0.1 MBit/s [89]), while USB2 is nominal 480 MBit/s and USB3.2 reaches 20 000 MBit/s [90])!

It is even worse with the **Gamma-Scout** devices, which support only 0.01 MBit/s or less [91]). But because they use a transfer algorithm of 7 bit with parity, the 8 bit bytes are split over a 2 bytes transmission, thereby reducing the effective transmission speed another two-fold! Their latest model supports 460 800 baud, which, however, is also still reduced 2 fold, i.e. no better than 0.2 MBit/s, i.e. only on par with the other devices.

So, none of these devices is anywhere close to a speed champion! With respect to logging or an occasional e.g. temperature measurement, the speed is sufficient, but for other actions, in particular downloading the history stored in the device's memory, a faster speed would be very welcome.

The serial port, the baud rate, the driver, and permissions

For a successful connection between computer and device you need to know the serial port's name given to it by the computer, the baudrate of the device, have a driver installed, and perhaps need the computer's permission to read and write to the port.

89 115 200 and 57 600 baud for GMC, up to 230 400 baud for the ELV dongle used for the I2c devices

90 https://www.elektronik-kompendium.de/sites/com/1310061.htm

91 9 600 and 2 400 (!) baud

Port names:

The port names on Linux are like '/dev/ttyUSB0', '/dev/ttyUSB1', …, on Windows like ‘COM3’, ‘COM12’, ‘COM17’, …., on Mac are less predictable, can be like ‘/dev/tty.USBSERIAL/’, ‘/dev/tty.PL2303-xxx’.

But worse: on unplugging and re-plugging the cable, the names may change!

Baud rate:

The baudrate is determined by the **external** device. While the computer can successfully open the serial port with any baudrate, a communication will **only** be possible if the baudrate is set to what the external device is set to!

GeigerLog finds the proper baudrate by testing communication using known snippets.

Driver:

On Linux the driver is always part of the system.

On Windows and Mac a driver may have to be installed. Often a generic driver suffices.

Read- and Write-Permissions for the Serial Port:

Depending on computer settings, a different hurdle may exist for Linux users, as a regular user (non-administrator) may not have the read- and write-permissions to work with the serial port. See Appendix C – HOWTO deal with read and write permissions for the serial port when on Linux on page 147 for guidance.

Appendix C – HOWTO deal with read and write permissions for the serial port when on Linux

NOTE: With GeigerLog's new **plug-and-play** ability also comes the option to discover specific devices with even greater flexibility. My recommendation is now to use **Option: 2) Make yourself a member of group 'dialout'**. GeigerLog does the rest.

After you have connected a device to the USB port on a Linux system, open a terminal and run this command:

```
ls -al /dev/ttyUSB*
```

the output is like :

```
crw-rw---- 1 root dialout 188, 0 Feb 26 12:16 /dev/ttyUSB0
```

It shows that a device is connected to port '/dev/ttyUSB0' and that only the user root and all users in group dialout have read and write permissions (rw). Everybody else can neither read nor write!

Unless you are logged in as root (which you shouldn't be doing for normal work) you can only use the device if you belong to the group dialout. To see whether you do, enter in a terminal (assuming your username is 'myname'):

```
groups myname
```

giving an output listing of all groups you are a member of, like:

```
myname : myname cdrom sudo dip plugdev lpadmin
```

There is no group dialout listed, and hence you have no permission for the serial port and cannot work with the Geiger counter.

~~You have 3 options to overcome this problem, of which the 3rd is the recommended one:~~
See Note above, Option 2 is the new recommendation!

1) Change permissions

In a terminal run `'sudo chmod 666 /dev/ttyUSB0'.` Follow by `'ls -al /dev/ttyUSB0'` and you see:

```
crw-rw-rw- 1 root dialout 188, 0 Feb 26 12:34 /dev/ttyUSB0
```

Now everyone has read and write permission. Security concerns may not be relevant here, but the problem is that you have to do this every time you unplug/replug the device!

2) Make yourself a member of group 'dialout'

To become a member of the dialout group, enter in a terminal:

```
sudo usermod -a -G dialout myname
```

You will need to logout and log back in to see your new group added. Entering:

```
groups myname
```

results in:

```
myname : myname dialout cdrom sudo dip plugdev lpadmin
```

This change is permanent; it also survives a reboot.

But what if `'ls -al /dev/ttyUSB*'` gets you:

```
crw-rw---- 1 root dialout 188, 0 Feb 26 12:58 /dev/ttyUSB0
crw-rw---- 1 root dialout 188, 1 Feb 26 12:59 /dev/ttyUSB1
```

This tells you that now two USB-To-Serial devices are connected to your computer. Obviously you can't tell from this listing which one is the new and which the old one. You'll have to try it out. With even more USB-To-Serial devices connected, it becomes even more complicated. And after a reboot, the order of the devices may have changed!

3) Take advantage of udev rules

In a terminal issue (as regular user):

```
lsusb
```

to get something similar to:

```
   Bus 002 Device 002: ID 8087:8000 Intel Corp.
   Bus 002 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
   Bus 004 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
*  Bus 003 Device 004: ID 4348:5523 WinChipHead USB->RS 232 adapter with Prolifec PL 2303 chipset
   Bus 003 Device 003: ID 0424:2514 Standard Microsystems Corp. USB 2.0 Hub
*  Bus 003 Device 002: ID 1a86:7523 QinHeng Electronics HL-340 USB-Serial adapter
```

The listing shows all USB devices of the computer, of which some belong to its inner circuitry. I have marked the two USB-To-Serial adapter with an asterisk on the left; the latter one is from the Geiger counter. Its ID is 1a86:7523, the first 4 hex digits being the vendor ID, the other 4 the product ID.

A udev rule allows the computer to recognize the connection of a device by this ID, and make certain settings and configurations, like giving read and write permissions.

Create a file containing nothing but these two lines:

```
# Comment: udev rule for GQ Electronics's GMC-300 Geiger counter
SUBSYSTEM=="tty", KERNEL=="ttyUSB*", ATTRS{idVendor}=="1a86", MODE:="666", SYMLINK+="geiger"
```

and save (you must be root to do this) as file '55-geiger.rules' in directory '/etc/udev/rules.d'. Then restart your computer (or issue the command '`sudo udevadm control --reload-rules`'). Then unplug and replug your Geiger counter device. You will now always find your device at port '/dev/geiger', irrespective of how many other devices are connected, and to which /dev/ttyUSB* !

HOWEVER: The USB-ID belongs to the USB-to-Serial converter chip installed in the Geiger counter. And since (to my knowledge) GQ is using the same chip an all Geiger versions, this simple rule will not allow to distinguish between them! You'll probably have to resort to option 2 above, and figure out, which /dev/ttyUSBX with X=1, 2, 3, ... belongs to which device! Not to mention that likely a million other devices may also be using the very same chip ...

Appendix D – The GMC Device Configuration Meanings

The device configuration of the GMC-300 series is read-out as 256 bytes of binary information. Its meaning is reported here: http://www.gqelectronicsllc.com/forum/topic.asp?TOPIC_ID=4447 . However, this list is not consistent with observed values at device 'GMC-300E Plus' with firmware 'GMC-300Re 4.20'. See here for even more differences: https://SourceForge.net/projects/gqgmc/files/gqgmc/GQ-GMC-ICD.odt/download

The GMC-500 and GMC-600 series of Geiger counters have a different configuration which in addition is twice as long at 512 bytes.

GQ has recently disclosed the configuration of the 500 and 600, but some details remain unclear so far. (see: http://www.gqelectronicsllc.com/forum/topic.asp?TOPIC_ID=4948), and at least 2 firmware bugs on the 500 series were discovered (see discussion in the topic). Please, report any problems via SourceForge (see Problems, Bugs, and Bug Reports on page 142).

The following list applies only to the **300 series:**

```
CFG data Offset table. Starts from 0
Values in BOLD are read and/or set in GeigerLog
================================================
PowerOnOff, //to check if the power is turned on/off intended
AlarmOnOff, //1
SpeakerOnOff,
GraphicModeOnOff,
BackLightTimeoutSeconds,
IdleTitleDisplayMode,
AlarmCPMValueHiByte, //6
AlarmCPMValueLoByte,
CalibrationCPMHiByte_0,
CalibrationCPMLoByte_0,
CalibrationuSvUcByte3_0,
CalibrationuSvUcByte2_0, //11
CalibrationuSvUcByte1_0,
CalibrationuSvUcByte0_0,
CalibrationCPMHiByte_1,
CalibrationCPMLoByte_1, //15
CalibrationuSvUcByte3_1,
CalibrationuSvUcByte2_1,
CalibrationuSvUcByte1_1,
CalibrationuSvUcByte0_1,
CalibrationCPMHiByte_2, //20
CalibrationCPMLoByte_2,
CalibrationuSvUcByte3_2,
CalibrationuSvUcByte2_2,
CalibrationuSvUcByte1_2,
CalibrationuSvUcByte0_2, //25
IdleDisplayMode,
AlarmValueuSvByte3,
AlarmValueuSvByte2,
AlarmValueuSvByte1,
AlarmValueuSvByte0, //30
AlarmType,
SaveDataType,
SwivelDisplay,
ZooMByte3,
ZooMByte2, //35
ZooMByte1,
ZooMByte0,
```

```
    SPI_DataSaveAddress2,
    SPI_DataSaveAddress1,
    SPI_DataSaveAddress0, //40
    SPI_DataReadAddress2,
    SPI_DataReadAddress1,
    SPI_DataReadAddress0,
    PowerSavingMode,
    Reserved, //45
    Reserved,
    Reserved,
    DisplayContrast,
    MAX_CPM_HIBYTE,
    MAX_CPM_LOBYTE, //50
    Reserved,
    LargeFontMode,
    LCDBackLightLevel,
    ReverseDisplayMode,
    MotionDetect, //55
    bBatteryType,
    BaudRate,
    Reserved,
    GraphicDrawingMode,
    LEDOnOff,
    Reserved,
    SaveThresholdValueuSv_m_nCPM_HIBYTE,
    SaveThresholdValueuSv_m_nCPM_LOBYTE,
    SaveThresholdMode,
    SaveThresholdValue3,
    SaveThresholdValue2,
    SaveThresholdValue1,
    SaveThresholdValue0,
    Save_DateTimeStamp, //this one uses 6 byte space
```

The following list applies only to the 500 and 600 series: (from: http://www.gqelectronicsllc.com/forum/topic.asp?TOPIC_ID=4948)

The GMC-500 and GMC-600 still accept configuration commands same as GMC-320, no change. But GMC-500 and GMC-600 extended (added new) commands for new features.

Here is the latest configuration data structure in C code on GMC-500 and GMC-600:

```
typedef enum {
CFG_PowerOnOff,
CFG_AlarmOnOff, //1
CFG_SpeakerOnOff,
CFG_IdleDisplayMode,
CFG_BackLightTimeoutSeconds,
CFG_IdleTitleDisplayMode,
CFG_AlarmCPMValueHiByte, //6
CFG_AlarmCPMValueLoByte,
CFG_CalibrationCPMHiByte_0,
CFG_CalibrationCPMLoByte_0,
CFG_CalibrationuSvUcByte3_0,
CFG_CalibrationuSvUcByte2_0, //11
CFG_CalibrationuSvUcByte1_0,
CFG_CalibrationuSvUcByte0_0,
CFG_CalibrationCPMHiByte_1,
CFG_CalibrationCPMLoByte_1, //15
CFG_CalibrationuSvUcByte3_1,
CFG_CalibrationuSvUcByte2_1,
CFG_CalibrationuSvUcByte1_1,
CFG_CalibrationuSvUcByte0_1,
CFG_CalibrationCPMHiByte_2, //20
CFG_CalibrationCPMLoByte_2,
CFG_CalibrationuSvUcByte3_2,
CFG_CalibrationuSvUcByte2_2,
CFG_CalibrationuSvUcByte1_2,
```

CFG_CalibrationuSvUcByte0_2, //25
CFG_IdleTextState,
CFG_AlarmValueuSvByte3,
CFG_AlarmValueuSvByte2,
CFG_AlarmValueuSvByte1,
CFG_AlarmValueuSvByte0, //30
CFG_AlarmType,
CFG_SaveDataType,
CFG_SwivelDisplay,
CFG_ZooMByte3,
CFG_ZooMByte2, //35
CFG_ZooMByte1,
CFG_ZooMByte0,
CFG_SPI_DataSaveAddress2,
CFG_SPI_DataSaveAddress1,
CFG_SPI_DataSaveAddress0, //40
CFG_SPI_DataReadAddress2,
CFG_SPI_DataReadAddress1,
CFG_SPI_DataReadAddress0,
CFG_nPowerSavingMode,
Reserved_1, //45
Reserved_2,
Reserved_3,
CFG_nDisplayContrast,
CFG_MAX_CPM_HIBYTE,
CFG_MAX_CPM_LOBYTE, //50
Reserved_4,
CFG_nLargeFontMode,
CFG_nLCDBackLightLevel,
CFG_nReverseDisplayMode,
CFG_nMotionDetect, //55
CFG_bBatteryType,
CFG_nBaudRate,
Reserved_5,
CFG_nGraphicDrawingMode,
CFG_nLEDOnOff, //60
Reserved_6,
CFG_nSaveThresholdValueuSv_m_nCPM_HIBYTE,
CFG_nSaveThresholdValueuSv_m_nCPM_LOBYTE,
CFG_nSaveThresholdMode,
CFG_nSaveThresholdValue3, //65
CFG_nSaveThresholdValue2,
CFG_nSaveThresholdValue1,
CFG_nSaveThresholdValue0,

CFG_SSID_0,
//...
CFG_SSID_31 = CFG_SSID_0 + 31, //68 + 31

CFG_Password_0, //100
//...
CFG_Password_31 = CFG_Password_0 + 31, //100 + 31

CFG_Website_0, //132
//....
CFG_Website_31 = CFG_Website_0 + 31, //132 + 31

CFG_URL_0, //163
//....
CFG_URL_31 = CFG_URL_0 + 31, //163 + 31

CFG_UserID_0, //195
//.....
CFG_UserID_31 = CFG_UserID_0 + 31, //195+31

CFG_CounterID_0, //227

//....
CFG_CounterID_31 = CFG_CounterID_0 + 31, //227 + 31

CFG_Period, //259
CFG_WIFIONOFF, //260
CFG_TEXT_STATUS_MODE,

/
CFG_Save_DateTimeStamp, //this one uses 6 byte space
CFG_MaximumCFGBytes,
}EEPROMDATAT;

ZLM:
For **GMC-500**, **GMC-600** history data C code structure:(this should be same as GMC-300, no change)
In history data, it start with 0x55AA00 prefixed for timestamp and followed by the date time data. and then always followed by 0x55AA and one of the bellow data length byte.

typedef enum{
YYMMDDHHMMSS, // Time Stamp
DOUBLEBYTE_DATA, //the data are double bytes
THREEBYTE_DATA, //the data are three bytes
FOURBYTE_DATA, //the data are four bytes
LOCATION_DATA, //the data is a text string,the first byte data is the length of the text, followed by the text
TOTAL_EEPROM_SAVE_TYPE
}HistoryDataFormatMarkingT;

Also, the 0x55AA also can follow a one of following history data type:
typedef enum{
SAVEOFF,
SECONDLY, //must be save value with TOTAL_EEPROM_SAVE_TYPE
MINUTETLY, //must be save value with TOTAL_EEPROM_SAVE_TYPE
HOURLY, //must be save value with TOTAL_EEPROM_SAVE_TYPE
SaveByThresholdSecond, //only save the data if exceed the preset threshold value
SaveByThresholdMinute, //only save the data if exceed the preset threshold value
TotalSavedType
}SaveDataTypeT;

The new **GMC-800** with 6 “calibration points”.
From: http://www.gqelectronicsllc.com/forum/topic.asp?TOPIC_ID=10394, Reply #1 & #2

GMC-800

# Naming	#Comment	#in-dex
POWERONOFF,		0
AlarmOnOff,		1
SpeakerOnOff,		2
GRAPHIC_UNIT,		3
BackLightTimeoutSeconds,		4
IdleTitleDisplayMode,		5
AlarmCPMValueHiByte, //6	**inactive!**	6
AlarmCPMValueLoByte,	**inactive!**	7
CalibrationCPMHiByte_0,	**inactive!**	8
CalibrationCPMLoByte_0,	**inactive!**	9
CalibrationuSvUcByte3_0,	**inactive!**	10
CalibrationuSvUcByte2_0, //11	**inactive!**	11
CalibrationuSvUcByte1_0,	**inactive!**	12
CalibrationuSvUcByte0_0,	**inactive!**	13

CalibrationCPMHiByte_1,	**inactive!**	14
CalibrationCPMLoByte_1, //15	**inactive!**	15
CalibrationuSvUcByte3_1,	**inactive!**	16
CalibrationuSvUcByte2_1,	**inactive!**	17
CalibrationuSvUcByte1_1,	**inactive!**	18
CalibrationuSvUcByte0_1,	**inactive!**	19
CalibrationCPMHiByte_2, //20	**inactive!**	20
CalibrationCPMLoByte_2,	**inactive!**	21
CalibrationuSvUcByte3_2,	**inactive!**	22
CalibrationuSvUcByte2_2,	**inactive!**	23
CalibrationuSvUcByte1_2,	**inactive!**	24
CalibrationuSvUcByte0_2, //25	**inactive!**	25
IdleDisplayMode,		26
AlarmValueuSvByte3,	**inactive???**	27
AlarmValueuSvByte2,	**inactive???**	28
AlarmValueuSvByte1,	**inactive???**	29
AlarmValueuSvByte0, //30	**inactive???**	30
AlarmType,		31
SaveDataType,		32
SwivelDisplay,		33
ZooMByte3,		34
ZooMByte2, //35		35
ZooMByte1,		36
ZooMByte0,		37
SPI_DataSaveAddress2,		38
SPI_DataSaveAddress1,		39
SPI_DataSaveAddress0, //40		40
SPI_DataReadAddress2,		41
SPI_DataReadAddress1,		42
SPI_DataReadAddress0,		43
nPowerSavingMode,		44
nSensitivityMode, //45		45
nCOUNTER_DELAY_HiByte,		46
nCOUNTER_DELAY_LoByte,		47
nDisplayContrast,		48
MAX_CPM_HIBYTE,		49
MAX_CPM_LOBYTE, //50		50
nSensitivityAutoModeThreshold,		51
LARGE_FONT_UNIT,		52
nLCDBackLightLevel,		53
nReverseDisplayMode,		54
nMotionDetect, //55		55
bBatteryType,		56
nBaudRate,		57
nCPMSpeakerOnOffCalib,		58
GRAPHIC_MODE,		59
nLEDOnOff,		60
nHCPMCAL,		61
nSaveThresholdValueuSv_m_nCPM_HIBYTE,		62
nSaveThresholdValueuSv_m_nCPM_LOBYTE,		63
nSaveThresholdMode,		64
nSaveThresholdValue3,		65
nSaveThresholdValue2,		66

nSaveThresholdValue1,	67
nSaveThresholdValue0,	68
FAST_ESTIMATE_TIME,	69
RTC_OFFSET,	70
ALARM_VOLUME,	71
TUBE_VOLTAGE,	72
CALIBRATION_CPM_MSB_0,	73
CALIBRATION_CPM_LSB_5 = CALIBRATION_CPM_MSB_0 + 23,	96
CALIBRATION_USV0_BYTE3,	97
CALIBRATION_USV5_BYTE0 = CALIBRATION_USV0_BYTE3 + 23,	120
CLICK_SOUND,	121
SPEAKER_VOLUME,	122
VIBRATION,	123
DOSIMETER_UNIT,	124
ALARM_CPM_BYTE3,	125
ALARM_CPM_BYTE2,	126
ALARM_CPM_BYTE1,	127
ALARM_CPM_BYTE0,	128
THEME,	129
DARK_THEME_COLOR_BYTE1,	130
DARK_THEME_COLOR_BYTE0,	131

The **GMC-500 / 600 /+** with 6 “calibration points”.
From: http://www.gqelectronicsllc.com/forum/topic.asp?TOPIC_ID=10437 , Reply #5

- 6 point calibration is added with 2.44 rev.
- 500 and 600 shared the same config since 2.31
- 500+ new calibration has 6 points. 3 for tube 1, 3 for tube 2.
- old 3 point calibration addresses are not used anymore.

PowerOnOff,
AlarmOnOff,
SpeakerOnOff,
IdleDisplayMode,
BackLightTimeoutSeconds,
IdleTitleDisplayMode,
AlarmCPMValueHiByte,
AlarmCPMValueLoByte,
CalibrationCPMHiByte_0,
CalibrationCPMLoByte_0,
CalibrationuSvUcByte3_0,
CalibrationuSvUcByte2_0,
CalibrationuSvUcByte1_0,
CalibrationuSvUcByte0_0,
CalibrationCPMHiByte_1,
CalibrationCPMLoByte_1,

CalibrationuSvUcByte3_1,
CalibrationuSvUcByte2_1,
CalibrationuSvUcByte1_1,
CalibrationuSvUcByte0_1,
CalibrationCPMHiByte_2,
CalibrationCPMLoByte_2,
CalibrationuSvUcByte3_2,
CalibrationuSvUcByte2_2,
CalibrationuSvUcByte1_2,
CalibrationuSvUcByte0_2,
IdleTextState,
AlarmValueuSvByte3,
AlarmValueuSvByte2,
AlarmValueuSvByte1,
AlarmValueuSvByte0,
AlarmType,
SaveDataType,
SwivelDisplay,
ZoomByte3,
ZoomByte2,
ZoomByte1,
ZoomByte0,
SPI_DataSaveAddress2,
SPI_DataSaveAddress1,
SPI_DataSaveAddress0,
SPI_DataReadAddress2,
SPI_DataReadAddress1,
SPI_DataReadAddress0,
nPowerSavingMode,
nSensitivityMode,
nCOUNTER_DELAY_HiByte,
nCOUNTER_DELAY_LoByte,
nDisplayContrast,
MAX_CPM_HIBYTE,
MAX_CPM_LOBYTE,
nSensitivityAutoModeThreshold,
nLargeFontMode,
nLCDBackLightLevel,
nReverseDisplayMode,
nMotionDetect,
bBatteryType,
nBaudRate,
nCPMSpeakerOnOffCalib,
nGraphicDrawingMode,
nLEDOnOff,
nHCPMCAL,
nSaveThresholdValueuSv_m_nCPM_HIBYTE,
nSaveThresholdValueuSv_m_nCPM_LOBYTE,
nSaveThresholdMode,
nSaveThresholdValue3,
nSaveThresholdValue2,
nSaveThresholdValue1,
nSaveThresholdValue0,

```
SSID_0,
//...
SSID_63 = SSID_0 + 63,

Password_0,
//...
Password_63 = Password_0 + 63,

Website_0,
//....
Website_31 = Website_0 + 31,

URL_0,
//....
URL_31 = URL_0 + 31,

UserID_0,
//.....
UserID_31 = UserID_0 + 31,

CounterID_0,
//....
CounterID_31 = CounterID_0 + 31,

Period,
WIFIONOFF,
TEXT_STATUS_MODE,
FAST_ESTIMATE_TIME,

THIRD_PARTY_OUTPUT, //for sending 0xFF per count via serial. This data format used on NETC.-
com software

HIGH_VOLTAGE_LEVEL_TUBE1, // for adjusting tube 1 voltage
HIGH_VOLTAGE_LEVEL_TUBE2, // for adjusting tube2 voltage
CPM_TUBE_MODE,
CPM_TUBE_DISPLAY,
VOLTAGE_DISPLAY,
DEADTIME_ENABLE,
DEADTIME_TUBE1_HIBYTE,
DEADTIME_TUBE1_LOWBYTE,
DEADTIME_TUBE2_HIBYTE,
DEADTIME_TUBE2_LOWBYTE,
MEDIUMTHRESHOLD_HB,
MEDIUMTHRESHOLD_LB,
HIGHTHRESHOLD_HB,
HIGHTHRESHOLD_LB,
SpeakerVolume,
HV_READING,
TARGET_HV_MSB,
TARGET_HV_LSB,
HV_CALIB,

SS1_0,
SS1_5 = SS1_0 + 5,
```

```
SS2_0,
SS2_5 = SS2_0 + 5,
SS3_0,
SS3_5 = SS3_0 + 5,
SS4_0,
SS4_5 = SS4_0 + 5,

ACCURACY_DISPLAY,

DOSE_ALARM_0,
DOSE_ALARM_1,
DOSE_ALARM_2,
DOSE_ALARM_3,

LANGUAGE,

CALIBRATION_CPM_MSB_0,
CALIBRATION_CPM_LSB_5 = CALIBRATION_CPM_MSB_0 + 23,
CALIBRATION_USV0_BYTE3,
CALIBRATION_USV5_BYTE0 = CALIBRATION_USV0_BYTE3 + 23,

DOSIMETER_UNIT,
```

Appendix E – GMC Device: Internal Memory, Storage Format and Parsing Strategy

There is no official document from GQ on the storage format, except for some basic **“GQ Geiger Counter Communication Protocol”** from 2015 [92]) and a later update from 2018 [93]). But most credit for an initial description goes to **user Phil Gillaspy** [94]), who published it on his SourceForge site [95]).

Other info comes from my own reverse engineering analysis of the memory content using this GeigerLog program and a GMC-300E+ device.

The internal memory of the Geiger counters is handled like a ring-buffer. The device begins to write at the bottom, and fills the memory up. Once it reaches the top, it continues at the bottom and fills up again, overwriting the previous history. This principle in combination with the storage format creates some headaches for parsing, i.e. the method through which a log file can be created from reading and interpreting the data.

Let’s start with the memory being completely erased - like after a factory reset, or a manual ‘Erase Saved Data’ command at either the counter itself or via the GeigerLog menu. Every single byte of the memory is set to the ‘empty’ value, which is hexadecimal FF, decimal 255. One problem already: you can also have a measured value of 255 and cannot distinguish between the two!

Date & Time Stamp

Once the memory is erased, the very first thing the counter does is writing a Date&Time stamp to the memory beginning at address 0000. Then the data follow.

This Date&Time stamp is repeated in intervals depending on the chosen saving mode:

- Mode ‘CPS, save every second’ once every 10 min, or every 600 to 3000 bytes
- Mode ‘CPM, save every minute’ once every hour, or every 60 to 300 bytes
- Mode ‘CPM, save hourly average’ once every hour, or every dozen bytes. For unknown reasons the saving occurs exactly once every 1 hour + 8 … 13 seconds; this difference is ignored in GeigerLog.
- Mode ‘OFF (no history saving)’ nothing is written; not even a message that saving was switched off

The wide ranges with respect to bytes result from the fact that a count rate (CPS or CPM) of up to 255 takes one byte to store, but a higher count rate takes 5 bytes, consisting of now 2 bytes of data, preceded by a 3 byte (!) double-byte-announcing-tag! The 2 bytes now allow up to 65535 counts.

92 https://www.gqelectronicsllc.com/download/GQ-RFC1201.txt

93 http://www.gqelectronicsllc.com/download/GQ-RFC1801.txt

94 https://SourceForge.net/projects/gqgmc/files/gqgmc/GQ-GMC-ICD.odt/download

95 https://sourceforge.net/projects/gqgmc/

However, I noted an inconsistency in the readings of CPS double-byte data, which may be due to some undeclared use of the top two bits by the firmware. Therefore GeigerLog masks those two bits for CPS values, and therefore the maximum reading is 16383 counts. CPM might also be affected in the same way, but such a high reading has not been seen. Currently no GeigerLog CPM mask is effective.

Data bytes are saved at the end of the period following the Date&Time stamp. It does not matter much in the second and minute saving intervals, but in the hourly case it may matter.

The Date&Time stamp also carries the information of the saving mode. Without that you can't interpret the data, as it could have been saved every second, or every minute, or every hour, as CPS or as CPM! The saving mode is valid until the next Date&Time stamp.

If a Note/Location tag was entered at the Geiger counter device, then it will be stored after every Date&Time Stamp.

Overflow

Once the memory is filled, the bottom memory is prepared for the overflow by erasing the first page (a page = 4kB, 4096 bytes) of the memory. Again, erasing means overwriting with FF. Once this page is full, the 2nd page is erased, and so on.

The first issue to consider is that the time sequence in the memory from bottom to top is now: youngest data, followed by oldest data, which are becoming younger as you go up in memory. Therefore GeigerLog does a final sorting of all records according to time of each record determined by the parser.

Further, it is unlikely that the overflow begins with a Date&Time stamp at address 0000; instead the Date&Time stamp will come later within the regular flow of data. But since a Date&Time stamp is stringently required for the parsing, all data have to be skipped until a Date&Time stamp is found.

GeigerLog takes care of this missed overflow by linearizing the ring-buffer. Thereby those skipped data are attached to the top end of the memory copy, and will be parsed at the end.

Page Boundaries

Another issue is that deleting a page may cut through a tag, be it a Date&Time stamp, an ASCII tag, or a 5 byte double-data-byte-tag, making the left-over data uninterpretable or worse, giving them a totally different meaning. Following is an example, taken from an actual recording.

In the old recording a Date&Time stamp begins at byte index 4089 (in green; 2017-02-15 09:19:12, CPM saving every minute), and extends over the page boundary (P) into the second page. It is followed immediately by another Date&Time stamp at byte index 4101 (in blue; first 4 bytes only).

```
 4085:aa=170   4086:02=  2   4087:11= 17   4088:0e= 14 4089:55= 85
 4090:aa=170   4091:00=  0   4092:11= 17   4093:02=  2 4094:0f= 15
 4095:09=  9 P 4096:13= 19   4097:0c= 12   4098:55= 85 4099:aa=170
 4100:02=  2   4101:55= 85   4102:aa=170   4103:00=  0 4104:11= 17
```

After the page is deleted, all bytes up to the end of the page are set to 255 (in gray). The former time fragments 19 (min) and 12 (sec) become regular counts (in white) and the remainder of the

Date&Time stamp beginning at 4098 (in yellow) has now become an ASCII tag with 85 bytes of supposed ASCII code following (only 3 bytes shown) [96].

```
 4085:ff=255   4086:ff=255   4087:ff=255   4088:ff=255 4089:ff=255
 4090:ff=255   4091:ff=255   4092:ff=255   4093:ff=255 4094:ff=255
 4095:ff=255 P 4096:13= 19   4097:0c= 12   4098:55= 85 4099:aa=170
 4100:02=  2   4101:55= 85   4102:aa=170   4103:00=  0 4104:11= 17
```

There is no way to put any meaning back into these fragments, therefore all data up the next Date&Time stamp must be discarded.

Another example from an actual recording: The Date&Time stamp (in yellow, 2022-02-04 05:48:19; ignore the date being 5 years into the future, this is yet another problem of the counter firmware) extends across a page boundary. The value at 28672 (in orange) is the Saving Mode byte, which can have values of 0, 1, 2, or 3. But it is 255.

```
 28660:10=16  28661:55=85    28662:aa=170  28663:00=0    28664:16=22
 28665:02=2   28666:04=4     28667:05=5    28668:30=48   28669:13=19
 28670:55=85  28671:aa=170 P 28672:ff=255  28673:ff=255  28674:ff=255
```

The parser can only conclude that this is improper and all subsequent values until the next Date&Time stamp are made negative to mark illegitimate data. When you see negative counts – this is the reason.

The 255 value

How many of the value 255 bytes do you need to see in order to conclude that these stand for 'empty' bytes? If there are hundreds, it seems clear. But where do you set the limit? If there are only three, two, or just one, they might well be correct counts, leaving the parser no choice but to consider the next bytes as correct as well. Most of the time this is nonsense.

GeigerLog's default action is to ignore all single bytes with value 255! This results in an error when you measure counts near 255, be it CPS or CPM. Apart from changing the average, you will loose 1 second or minute, resp., in the time tag. But this is corrected with the next Date&Time stamp.

You can change this default action by starting GeigerLog with (see also menu Help → Options):

```
./geigerlog keepFF
```

This will result in all values 255 being treated as if they are correctly measured values. But most of the time this will be a mess, which needs to be corrected manually.

Correcting a Wrong History

It is an annoying procedure. The following is suggested:

1. Download the full history from the counter, and look at the graph
2. Try to zoom into the critical zone with mouse-left-click and mouse-right-click followed by Apply. Do it until you are able to read the time and count value of a relevant data point
3. Search the *.his file for this data point and note the byte index (first column)
4. Search the *.lst file for this byte index, and determine which data need to be deleted
5. Using a program able to handle binary files, delete the segment just determined in *.bin file

96 Actually, as ASCII is limited to a 7 bit code, values of 128 and greater are not ASCII code; but GeigerLog is generous and reads it as an 8 bit code. It is nonsense anyway.

6. The remaining *.bin file can now be opened and parsed again, and should result in a proper history. If not, repeat at step 2.

All Parsing Problems are Encountered by All Software!

Just to be sure: these are problems created by poor firmware! All software, including GQ's own software, is impacted by these firmware issues!

Appendix F – Firmware Differences

The firmware of the GQ Geiger counters has bugs. Nobody is surprised that software has bugs. The unpleasant part is that GQ was not the most forthcoming in disclosing these bugs after they became known.

Furthermore, the firmware is modified from model to model. So far a normal process. Though what the modifications were, was not disclosed. Of course, it is completely up to the owner of this software to decide on what to publish or not, were it not for their simultaneous promotion and marketing of their products as 'open', as done for all models including their very latest GMC-600+, quote: "GQ GMC-600 Plus provides **open** GQ RFC1201 communication protocol for easier system integration". Well, no. This document had flaws at the time of release in Jan 2015 for the then latest GMC-300 models, and today has significant differences to the real situation, despite claims to the opposite. You surely can't do any "system integration" based on this outdated document.

I was therefore very pleased that GQ had decided to come forward with helpful information, which is mostly included in the extended online discussion in this post with topic 4948:
http://www.gqelectronicsllc.com/forum/topic.asp?TOPIC_ID=4948

This has allowed to fully integrate the 500 and 600 series into GeigerLog 0.9.07!

However, during this discussion some more firmware bugs surfaced. While they don't seem to impact the function of GeigerLog, you can never be sure about what is going on as long as you have not at least understood the issues, let alone haven't solved them.

- Both Logging and History download is working on all models
- Reading the calibration factor works on all series
- Reading and Setting Geiger counter configurations like, alarm, speaker, power status, History saving mode works for all series. However, they do not work reliably, not even for the old 300 series counters: every now and then a function fails, which always turned out to be due to an unexpected timeout of the counter. This is an issue of the counter's firmware! GeigerLog attempts to correct the failure, and is mostly, but not always successful. Look for the output printed to the NotePad. Your command may have not been successful; repeat the command if it did not succeed.
- If you find a problem, and can repeat it, please, file a bug report. For instructions See chapter Problems, Bugs, and Bug Reports on page 142.

History Download issues

The history is downloaded in pages of up to 4k (4096) bytes, which is hexadecimal 1000. The download is triggered by a request from the computer to send a page of the desired size. This desired size is then logically ANDed with hex0FFF, with the consequence that no bytes are send by the Geiger counter at all because (hex1000 AND hex0FFF) = 0 !

Such is the situation with the 'GMC-300 v3.20' Geiger counter, which necessitates to limit the reading to half pages with a size of 2k.

In later models this firmware bug has been modified to a different firmware bug, whereby one byte more than requested is sent. When requesting a full 4k page of data, the firmware sends only (hex1000 AND 0FFF + 1) = 1 data point instead of 4096. The workaround is to request 4096 - 1 = 4095 data points, which results in (hex0FFF AND hex0FFF) = hex0FFF, then adds 1, resulting in hex1000, or, voilà, the full 4096 bytes.

Such is the situation with the ‘GMC-300E Plus v4.20’ and ‘GMC-320’ (assuming v4.20 firmware).

Note that this cannot be corrected by asking all counters for 2k half-pages only, as the extra byte send by the later firmware still needs to be taken care off!

In the 500 and 600 series this extra-byte modification seems to have been reversed. I don’t know how, but reading only half pages (2k) is working.

Configuration Issues

For the 300 series the configuration is stored in a memory of 256 bytes. There is confusion around the meaning of each entry (see Appendix D – The GMC Device Configuration Meanings, page 149), though most is understood.

For the 500 and 600 series the configuration is twice as long at 512 bytes, and with the recent disclosure by GQ, the meaning is now defined (http://www.gqelectronicsllc.com/forum/topic.asp?TOPIC_ID=4948).

Double-tube Counters

The history for the 500+ counters now allows to save the sum of the counts of both tubes or either the first or the second tube.

The sum of both tubes does not make any sense at all, but is the current default setting of all firmware so far. The first tube is the more sensitive tube, typically a M4011 tube, and the second tube is the much less sensitive SI3BG tube, in later models changed to the J707, see chapter Tube J707 on page 177.

The tube choice (1st or 2nd) will be auto-detected by GeigerLog from the downloaded history.

History Downloads using GQ Dataviewer and analyzing with GeigerLog

The GQ software Dataviewer may add the configuration memory of 256 or 512 bytes to the downloaded history memory, which may result in false parsing results. GeigerLog now eliminates such wrong data.

Appendix G – Calibration

Relocated to file: **GeigerLog -- Calibration Guidance-v1.5.pdf**

see: https://sourceforge.net/projects/geigerlog/files/Articles/

Appendix H – Light Sensitivity of Glass Geiger Tubes

I believe to have been the first to point out that some GMC counters are light-sensitive in this post from 2017 with a M4011 clear glass tube "**Counter counts mostly daylight, less radioactivity**" [97]). More posts were following, e.g this one showing that even a blackened M4011 tube, covered with some black plastic to keep light out, shows light-sensitivity "**Light-Sensitivity of a GMC Counter - Yet again!**" [98]). And not only M4011 tubes, also other glass tubes, like this J305 clear glass tube "**Groundhog Day in Light-Sensitivity County**" [99]). And even a video showing the light effect on a M4011 clear glass tube "**Light Sensitivity of M4011 Tubes (Video)**" [100]). And " **GMC-320 Plus - Defect Tube?**" [101]). I don't believe the list of postings is complete.

There was much discussion and speculation about the cause of this light sensitivity. Beyond assuming the "Photoelectric Effect" acting on the inner tube surface and on the gas in the tube as the cause, no conclusion was drawn on a remedy for this problem.

Science Trivia:

Einstein got his only Noble Prize for his quantum explanation of the Photoelectric Effect, and never one for his work on relativity and gravitation!

This Nobel Prize had been commemorated in Germany with a postage stamp [102]) showing electrons being kicked out of a surface by light of wavelengths from beyond-red to blue, but not from red.

User SelTec has found this very interesting scientific article on the subject: [103]). It is almost 70(!) years old and is confirming our assumptions. The (German) summary of the original article is shown below, and a Google translation follows.

User Damien found another scientific article [104]) of similar age (65 years), seemingly showing yet another mechanism, by which photo-sensitivity of glass Geiger tubes is created.

We can now conclude:

- Light sensitivity of Geiger tubes made from glass is an established fact
- Known for at least 7 decades

It surely raises the question: why using Geiger tubes made from transparent glass ?

97 http://www.gqelectronicsllc.com/forum/topic.asp?TOPIC_ID=4540
98 http://www.gqelectronicsllc.com/forum/topic.asp?TOPIC_ID=9287
99 http://www.gqelectronicsllc.com/forum/topic.asp?TOPIC_ID=9544
100 http://www.gqelectronicsllc.com/forum/topic.asp?TOPIC_ID=9862
101 http://www.gqelectronicsllc.com/forum/topic.asp?TOPIC_ID=9931
102 https://de.wikipedia.org/wiki/Photoelektrischer_Effekt
103 https://link.springer.com/article/10.1007/BF01330330
104 https://www.sciencedirect.com/science/article/abs/pii/0016003258903909

First Article:

Die Lichtempfindlichkeit von selbstlöschenden Geiger-Zählern aus Glas mit Außenkathode

Helmut Schwarz, Zeitschrift für Physik, Volume 134, Pages 540–545 (**1953**)

Zusammenfassung

Es werden einige Untersuchungen durchgeführt, um eine Erklärung für die Lichtempfindlichkeit von selbstlöschenden Geiger-Zählern aus Glas mit Außenkathode zu finden. Der Effekt äußert sich durch das Auftreten von unechten Zählimpulsen oder Doppelimpulsen bei Tageslicht. Wir konnten zwischen einem photoelektrischen Effekt an der inneren Glasoberfläche (Oberflächeneffekt) und innerhalb des Gasvolumens (Volumeneffekt) unterscheiden. Der Volumeneffekt scheint mit der Elektronenanlagerung an Sauerstoffmoleküle verbunden zu sein, welche entweder im Rohre noch verblieben oder durch Zersetzung des Alkoholdampfes beim Betrieb des Zählrohres entstanden waren. Wenn der Sauerstoff durch Lichtabsorption in einen metastabilen Zustand gerät, nimmt die Elektronenanlagerungswahrscheinlichkeit beträchtlich zu. — Der Oberflächeneffekt scheint mit der Abnahme der Austrittsarbeit des Glases bei Adsorption von z. B. Wasserstoff in Verbindung zu stehen, welcher von der Zersetzung des Alkoholdampfes herrührt.

Google Translation: I have verified the translation to be adequate.

Summary

Some research is being conducted to find an explanation for the photosensitivity of external cathode self-extinguishing glass Geiger counters. The effect manifests itself in the occurrence of spurious counting pulses or double pulses in daylight. We could distinguish between a photoelectric effect at the inner glass surface (surface effect) and within the gas volume (volume effect). The volume effect seems to be connected with the attachment of electrons to oxygen molecules, which either remained in the tube or were created by decomposition of the alcohol vapor during operation of the counter tube. When oxygen enters a metastable state by light absorption, the electron attachment probability increases considerably. — The surface effect appears to decrease with the decrease in the work function of the glass upon adsorption of e.g. B. to be connected to hydrogen, which results from the decomposition of the alcohol vapor.

Second Article:

The nature of the photosensitivity of Geiger counters

H.O.Albrecht, C.E.Mandeville, Journal of the Franklin Institute, Volume 265, Issue 6, June **1958**, Pages 473-481

Some aspects of the photosensitivity of the Geiger counter are discussed. Methods for increasing response to ultraviolet and near ultraviolet photons are described. The photosensitivity is traced to formation of a surface film upon the inner wall of the cathode; that is, a quasi-metallic alkyl coating is assumed to be formed. Relatively short wave radiation of sufficient energy to form free radicals in hydrogen and organic quenching gases is required to increase (“activate”) the sensitivity.

Appendix I – Installation

A full working environment for GeigerLog needs Python3 & Pip plus some supporting packages. If these conditions are met, GeigerLog will run on **Linux**, **Windows**, **Mac**, **Raspi**, and **other systems**!

This is the list of the supporting packages. They will be **auto-installed** by GeigerLog! A GeigerLog user may need this list only when problems come up.

GeigerLog will create its own **Python Virtual Environment**. Thus this list of packages is likely different from your default installation!

Supporting packages as of March 2024, required and verified to work with GeigerLog 1.5.0 under Python 3.11:

- pip latest version: 24.0
- setuptools latest version: 69.2.0
- PyQt5 latest version: 5.15.6
- PyQt5-sip latest version: 12.9.1
- matplotlib latest version: 3.8.3
- numpy latest version: 1.26.2
- scipy latest version: 1.10.1
- pyserial latest version: 3.5
- paho-mqtt latest version: 1.6.1
- sounddevice latest version: 0.4.6
- soundfile latest version: 0.12.1
- psutil latest version: 5.9.1
- ntplib latest version: 0.4.0
- py-cpuinfo latest version: 9.0.0

Optional, or Operating-System dependent:

- LabJackPython latest version: 2.1.0
- RPi.GPIO latest version: 0.7.1 needed for Raspi 1, 2, 3, 4
- rpi-lgpio latest version: 0.4 needed for Raspi 5
- smbus2 latest version: 0.4.3 needed for Raspi (all versions)

Older and newer versions will work most of the time, but NOT ALWAYS.

When you experience problems, make sure to have the listed versions installed!

Guidance for All Operating Systems

Python – the recommended version is Python 3.11

GeigerLog 1.5 requires **Python version 3.8 or later**. It was confirmed to work on Python 3.8, 3.9, 3.10, 3.11, and 3.12, both on Linux, Raspberry Pi and on Windows 10 and Windows 11. Python 3.11 will be supported until Oct 2027 [105]).

Python version 3.7 and earlier are no longer supported by GeigerLog. Those Python versions are past their end-of-life anyway.

If you have no Python 3 installed, look below for operating system specific installation instructions.

Pip

Pip [106]) is the **P**ackage **I**nstaller for **P**ython. For any Python installation – be it on Linux, Windows, Mac, or else – GeigerLog **DOES NEED** this program for its setup! It does not work without it.

All of the Python versions come packaged with Pip, and while Pip should normally be installed by default, make sure that it is!

64 bit versus 32 bit

Use a 64 bit installation if your operating system supports it. Any following download instructions assume 64 bit. If you have to use 32 bit, find the equivalent downloads.

Administrative rights

The default installation of Python will typically be for all users, and **has to be installed with administrative rights**.

However, GeigerLog does **NOT** need administrative rights and you should **NOT** use it!

Verify the current Python installation status on your machine

On your machine you may have installed only Python version 2.x, or only Python version 3.x, or both, or neither. Furthermore, depending on your operating system and distribution, as well as your history of installations, one of the two can probably be started with ‘python’, while the other needs to be started with ‘python2’ or ‘python3’.

To find out your situation, look at the output of these commands entered in a Command Window:

```
python -V
python2 -V
python3 -V
```

The responses will tell you which commands are valid and which one starts which version of Python. The following commands assume that ‘python’ calls the right version.

105 https://devguide.python.org/versions/

106 Pip’s Homepage: https://pypi.org/project/pip/

Verify the current Pip installation status on your machine

You'll need to make sure you have Pip available. You can check this by running:

```
python -m pip --version
```

If `Pip` isn't already installed, then first try to bootstrap it from the standard library:
(for detailed instructions see: https://packaging.python.org/tutorials/installing-packages/)

```
python -m ensurepip --default-pip
```

If pip is not installed, and the bootstrap failed, then you may have to re-install your complete Python3, see instructions specific for your operating system.

Setup GeigerLog

If both Python and Pip are installed as version 3, then you are ready to go. Follow the instructions given in Installing and Starting GeigerLog on page 16 for setup and starting.

If this fails, come back here, and walk through the next OS-specific paragraphs.

Linux – Installation

Installation of Python and Pip

If no Python is installed, or Python is older than version 3.8, then install Python Version 3.11.

If the distribution does not offer this by default, you may have to activate a suitable repository. On Ubuntu-like systems this is the **deadsnakes** repository (https://github.com/deadsnakes), which provides many new and old versions of Python:

```
sudo add-apt-repository ppa:deadsnakes/ppa)
```

If Python 3.11 is not available, try the next lower version, but at least version 3.8.

Install:

```
sudo apt-get install python3
sudo apt-get install python3-pip
```

Setup GeigerLog

If both Python and Pip are installed as version 3 you are ready to go. Follow the instructions given in Installing and Starting GeigerLog on page 16 for setup and starting of GeigerLog.

If this fails, come back here, and read the next paragraphs.

What if GeigerLog won't start?

For first help look into chapter Problems, Bugs, and Bug Reports on page 142. The problem persists? Then file a bug report, which is also outlined there.

Known Installation Problems and Solutions

Missing Programs

Some Linux distributions require to explicitly install programs from their repository, which are not installed by default. Candidates are: (thanks to user theMike!)

```
sudo apt-get install python3-setuptools
sudo apt-get install python3-dev
```

When an error "**OSError: PortAudio library not found**" [107]) comes up, you may have to install this program: (thanks to user engelbert!)

```
sudo apt-get install libportaudio2
```

Pip-Usage Impossible

Sometimes a package installed by the distribution is too old and does not allow to be updated by Pip. If Pip complains that it can't do an update, then un-install this package first with the distribu-

107 https://stackoverflow.com/questions/49333582/portaudio-library-not-found-by-sounddevice

tion tools, like for Ubuntu:

```
apt-get purge <package-name>
```

and **only then** reinstall with Pip. (Such was the case for pyserial in Mint; thanks to user theMike!)

PyQt5 Error

An obscure error message resulted from the use of PyQt5 version 5.15.0 in Python 3.6:

```
qt.qpa.plugin: Could not load the Qt platform plugin "xcb"
in "" even though it was found.
This application failed to start because no Qt platform plugin could
be initialized. Reinstalling the application may fix this problem.
```

This is highly misleading. The solution was to install **libxcb-xinerama0** from the repository:

```
sudo apt-get install libxcb-xinerama0
```

PyQt5 Error

(thanks to user Simon!) The module PyQt5 must go in sync with its sibling sip. When you see an error:

```
'the sip module implements API v12.0 to v12.8 but the PyQt5.QtWidgets module re-
quires API v12.13'
```

then try to solve it with command:

```
__venvGL1_5_0/bin/python -m pip install PyQt5-sip==12.13
```

Ubuntu Braille Error [108])

Ubuntu has a bug preventing connection to any GQ GMC counter.

It is relevant for all USB devices using VID (vendor ID) and PID (Device ID) `1A86:7523.` The issue with Ubuntu is that it assumes it is a "Braille" device (for the blind) and, ironically, blindly treats it as such. This causes the GQ GMC counter to not connect.

The fix is to comment out the `udev` rules that are responsible for this. The text file may be in two places.

```
/usr/lib/udev/85-brltty.rules
/usr/lib/udev/rules.d/85-brltty.rules
```

Find the line below and comment it out.

```
ENV{PRODUCT}=="1a86/7523/*",ENV{BRLTTY_BRAILLE_DRIVER}="bm", GOTO="brltty_usb_run"
```

Windows – Installation

This was verified with an installation of **Windows 10 and Windows 11** on two different computer.

108 https://github.com/Wikilicious/pygmc/tree/v0.7.1#known-issues

Installation of Python and Pip

Preferably install Python V3.11, though so far V3.12 has also worked. At a minimum you need Python V3.8. Download from: https://www.python.org/downloads/windows/.

You want the "**Windows x86-64 executable installer**" :

Open it **with administrative rights!**

In the installer:

- check: add Python 3.11 to Path
- select: Customize Installation
- under Optional Features:
  - check all options
  - select Next
- under Advanced Option:
  - check all options (except for the last two 'Download…' items; they are not needed)
  - leave the install path at 'C:\Program Files\Python39'
  - select Install
- once finished: ignore the option to disable path length limit and close the installer

Setup GeigerLog

If both Python and Pip are installed as version 3 you are ready to go. Follow the instructions given in Installing and Starting GeigerLog on page 16 for setup and starting of GeigerLog.

If this fails, come back here, and read the next paragraphs.

What if GeigerLog won't start?

For first help look into chapter Problems, Bugs, and Bug Reports on page 142. The problem persists? Then file a bug report, which is also outlined there.

Known Installation Problems and Solutions

When you encounter error messages like:

```
ImportError: DLL load failed: The specified package could not be
found.
```

The most likely reason is that "Microsoft Visual C++ Redistributable" is missing. Install from the Microsoft website:

https://support.microsoft.com/ms-my/help/2977003/the-latest-supported-visual-c-downloads

Mac – Installation

The following has not been tested on a Mac, but is derived from various online sources. A HOWTO for using Python on a Mac is available on this site from the Python creators: https://docs.python.org/3/using/mac.html and covers relevant topics.

To install Python version 3 see instructions under the above link. A “universal binary” build of Python, which runs natively on the Mac’s new Intel and legacy PPC CPU’s, is there available. Note the caveat on starting programs with a GUI (Graphical User Interface, which GeigerLog has) due to a quirk in Mac.

The latest Python releases for Mac are here: https://www.python.org/downloads/mac-osx/ Download the latest version of 3.11. The minimum Python version is 3.8. Preferably use 64 bit. Now with Python working, verify your installation status on your machine with the commands given in chapter Verify the current Python installation status on your machine on page 168.

Setup GeigerLog

If both Python and Pip are installed as version 3 you are ready to go. Follow the instructions given in Installing and Starting GeigerLog on page 16 for setup and starting of GeigerLog.

If this fails, come back here, and read the next paragraphs.

What if GeigerLog won’t start?

For first help look into chapter Problems, Bugs, and Bug Reports on page 142. The problem persists? Then file a bug report, which is also outlined there.

Known Installation Problems and Solutions

None reported.

Raspberry Pi – Installation

Raspberry Pi 4 with Bullseye

My hardware was a **Raspberry Pi 4 Model B with 2 GB RAM**. After initially testing it with the previous operating system named **Buster**, I had then tested it with a fresh install with the most current (as of March 2022) operating system named **Bullseye**.

The Raspi folks provide a **Raspberry Pi Imager** [109]) for Windows, Linux, and Mac, which prepares a microSD card from which you can boot the Raspi. However, this software failed to run on my Linux computer. So I did a manual installation by downloading the **Raspberry Pi OS (64-bit), Release date: January 28th 2022** [110]), and used Ubuntu's **usb-creator-gtk** to prepare the microSD card. Bullseye has Python in version 3.9.2, and no longer has a Python version 2; thus all Python commands on the Raspi with Bullseye are 'python', not 'python3'!

In contrast to what I emphasized for all other installation, you better use the Linux installer for whatever can be installed by it instead of Pip, and you better do NOT attempt to change it by using Pip! And, even stranger, some software is installed and usable, but could not be detected by Pip. This applied to PyQt5 and PyQt5-sip, but only on Buster! Quite odd.

Raspberry Pi 5 with Raspberry Pi OS (based on Debian Bookworm)

My hardware was a **Raspberry Pi 5 Model B Rev 1.0 with 4 GB RAM**. The OS was installed according to these directions [111]) and everything worked flawlessly (as of March 2023).

Debian Bookworm is strongly encouraging the use of a Python Virtual Environment, which GeigerLog is indeed now using.

Setup GeigerLog

If both Python and Pip are installed as version 3 you are ready to go. Follow the instructions given in Installing and Starting GeigerLog on page 16 for setup and starting of GeigerLog.

If this fails, come back here, and read the next paragraphs.

What if GeigerLog won't start?

For first help look into chapter Problems, Bugs, and Bug Reports on page 142. The problem persists? Then file a bug report, which is also outlined there.

Known Installation Problems and Solutions

Other than above, none reported.

109 https://www.raspberrypi.com/software/

110 https://downloads.raspberrypi.org/raspios_arm64/images/raspios_arm64-2022-01-28/2022-01-28-raspios-bullseye-arm64.zip

111 https://www.raspberrypi.com/software/

Appendix J – Using Pip

With the new GeigerLog Auto-installation process the use of Pip may no longer be necessary for the GeigerLog user, except when installation problems occur.

Using Pip has become a bit more complicated now that the distributions have begun to enforce the installations into Python specific Virtual Environments. It is recommended to use the **GeigerLog setup** command for auto-installing GeigerLog, and use Pip only if you are well familiar with it and Python Virtual Environments!

More on Pip here: https://pip.pypa.io/en/stable/.

GeigerLog's PipCheck Tool

GeigerLog's own tool **`PipCheck`** might be an easy, welcome helper when there are any installation problems. It comes as part of GeigerLog; you find it in the GeigerLog **gtools** directory. It is a tool build around Pip and simplifies the verification of your installation. It is aware of Virtual Environments. See details in chapter GeigerLog PipCheck on page 129. Start with:

on Linux, Mac: `./PipCheck.sh`

on Windows: `PipCheck.bat`

PipCheck ends by printing out a **Mini-HOWTO**, which is a good reminder for the use of standard commands with Pip.

Using Pip

Reminder: Pip needs to be executed in a Virtual Environment!

Pip is executed in a **Terminal / Command window**. This example shows the update of numpy, in this case from version 1.23.0 to version 1.23.2:

```
geigerlog$ python -m pip install -U numpy
Requirement already satisfied: numpy in lib/python3.10/site-packages (1.23.0)
Collecting numpy
  Using cached numpy-1.23.2-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (17.0 MB)
Installing collected packages: numpy
  Attempting uninstall: numpy
    Found existing installation: numpy 1.23.0
    Uninstalling numpy-1.23.0:
      Successfully uninstalled numpy-1.23.0
Successfully installed numpy-1.23.2
```

Command: Do a fresh install of package numpy:

```
python -m pip install numpy
```

Command: Do an upgrade of package numpy by using option '-U':

```
python -m pip install -U numpy
```

Command: List all versions for which an update is available:

```
python -m pip list --outdated
```

Command: Showing detailed version information of a specific package

To show details on a package, e.g. matplotlib:

```
python -m pip show matplotlib
```

the output will be like:

```
Name: matplotlib
Version: 3.3.2
Summary: Python plotting package
Home-page: https://matplotlib.org
Author: John D. Hunter, Michael Droettboom
Author-email: matplotlib-users@python.org
License: PSF
Location: /home/ullix/geigerlog/vgl38/lib/python3.8/site-packages
Requires: certifi, kiwisolver, pillow, cycler, numpy, python-dateutil, py-
parsing
Required-by:
```

Command: Looking for a specific version

To view all available package versions exclude the version number, like:

```
python -m pip install paho-mqtt==
```

which will result in an error message, which gives you the available versions, like:

```
ERROR: Could not find a version that satisfies the requirement paho-mqtt==
(from versions: 1.1, 1.2, 1.2.1, 1.2.2, 1.2.3)
ERROR: No matching distribution found for paho-mqtt==
```

Command: Installing a specific version

To install a specific python package version irrespective whether it is for the first time, for an upgrade or a downgrade, use (e.g. for the package `paho-mqtt`):

```
python -m pip install --force-reinstall paho-mqtt==1.2.1
```

Appendix K – Tubes

This is an assembly of Geiger tube information, which is of relevance to some of the Geiger counters supported by GeigerLog.

GMC-500+ – Different Tubes in some Counters

The GMC-500+ counters in the past had a **M4011** tube, plus the **SI3-BG** tube as a second low-sensitivity tube. This has been changed recently for at least some GMC-500+ counter [112]).

Quote from the post: *"After contacting the retailer, it turns out that the latest version of GMC 500+ uses a different high dose rate tube. The **SI3-BG** tube has been replaced with something called **J707**, which supposedly has better lifetime and stability."*

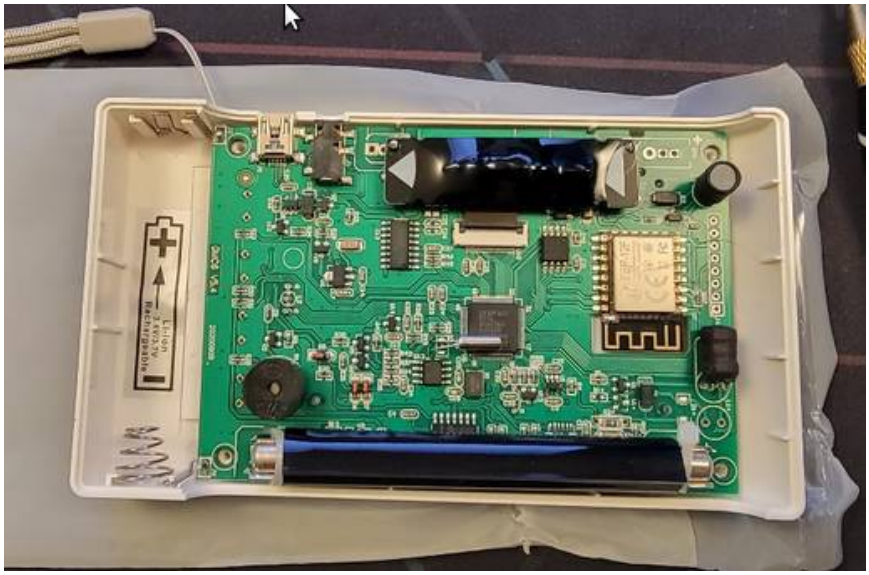


Figure 72: GMC-500+ with J707 tube

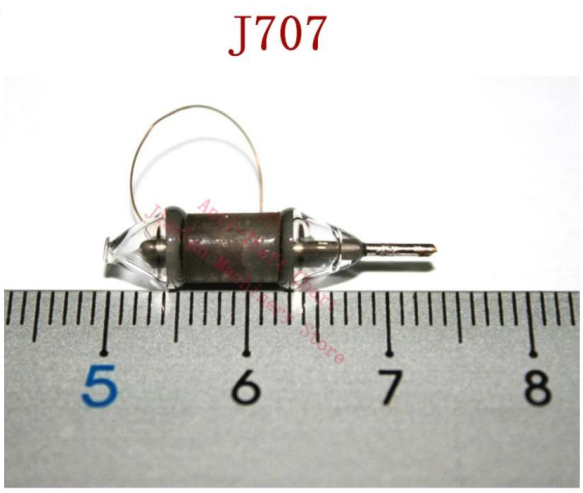
Tube J707

J707 Geiger Tube Specification: (from: [113])

Maximum length: 18mm
Maximum diameter: 6.5mm
Shell material: metal
Maximum starting voltage: 330V
Minimum flat area: 100V
Ping incline: 30%/100V
Recommended working voltage: 400V
Life: 10^10 power pulse
Detection object: β, γ rays

112 This had been reported in the GQ forum with the title: "**Is my GMC 500+ genuine?**" url: http://www.gqelectronicsllc.com/forum/topic.asp?TOPIC_ID=10006, but this URL has magically disappeared :-(

113 https://de.aliexpress.com/item/1005002406670196.html



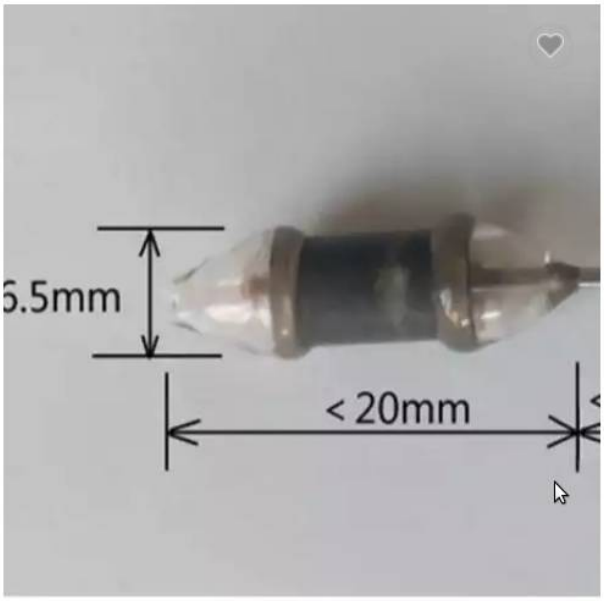


Table from [114]). The claimed sensitivity of 2.5 cps / (mR/h) (with Co-60) results in a Sensitivity of: see chapter “Deprecated Radiation Units” in article “GeigerLog Calibration Guidance” [115].

2.5 cps/(mR/h) * 60 CPM/CPS / 10 mR/µSv = **15 CPM/(µSv/h)**

Non.	maximum.	Dia	maximum	utile	Matériel	initiale	pleteau zone	de Pente	Tension de fonctionnement	gamma; (60Co) cps/	gamma; (60Co) cps/
	(Mm)	(Mm)	(Mm)	Mm		(V)	(V)	(% /100V)	(V)	MR/h)	MGy/h)
J613 & gamma;	9.5	3.5	66	30	Verre	350	100	15	420	5	0.5
J614 & gamma;	7.5	3.5	55	30	Verre	350	100	15	420	5	0.5
J401 & gamma;	13.5	10	91	50	Verre	350	100	15	420	12	1.2
J622 & gamma;	12	5	130	80	Verre	350	100	15	420	12	1.2
J403 & gamma;	23	18	263	190	Verre	350	80	10	420	120	12
J306 & gamma;	19	16	200	150	Verre	350	80	10	400	80	8
J5101	40	18	180	110	Verre	1200	200	5	1400	/	/
J142αβ	37	16	65	50	En métal	450	100	10	550	/	/
J707 & gamma;	6	5	18	6.5	En métal	330	100	30	400	2.5	0.25
J705β & gamma;	6	5	28	10	En métal	400	100	15	500	5.5	0.55
J301β & gamma;	5	2	45	10	En métal	350	100	20	400	0.8	0.08
J302β											

114 https://french.alibaba.com/product-detail/J305-M4011-glass-tube-for-Geiger-60431668386.html

115 https://sourceforge.net/projects/geigerlog/files/Articles/

Appendix L – License

GeigerLog is licensed under GPL3. The license text is available in file `COPYING` in the GeigerLog folder. If the file is missing you find a link to it in this text, which is part of all GeigerLog files:

```
################################################################################
#    This file is part of GeigerLog.                                           #
#                                                                              #
#    GeigerLog is free software: 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.                                       #
#                                                                              #
#    GeigerLog 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.                              #
#                                                                              #
#    You should have received a copy of the GNU General Public License         #
#    along with GeigerLog.  If not, see <http://www.gnu.org/licenses/>.        #
```