Bridgeport Instruments, LLC, 11740 Jollyville Rd., Ste 500, Austin, TX 78759,
www.BridgeportInstruments.com

By **Michael Momayezi**
Phone **512-553-9933**
Fax **512-553-9934**
email **momayezi @bridgeportinstruments.com**

MCA-2000 Users Manual

V. 1.2, July, 2022

Summary

The MCA-2000 is a family of low-cost multi-channel analyzer (MCA) for measuring gamma-ray radioactivity with a scintillator.

The PMT-2000 includes a PMT operating voltage supply and plugs directly onto a photomultiplier.

The SIPM-2000 includes the SiPM operating voltage supply and plugs directly onto an SiPM-array.

Using its embedded 32-bit ARM processor, it provides accurate gamma-ray spectroscopy and gain stabilization together with many advanced features such as automatic background subtraction and alarm computations.

This document describes how to use the MCA-2000 from the graphical user interface and from Python scripts. Since all commands are formatted as human readable JSON strings, developers will find they can program the MCA-2000 in any language they want.

Feature summary

- Histograms: Either two 2K×32 or one 4K×32
- Up to 560kcps histogramming rate
- PMT-2000: Support for positive and negative operating voltage and different PMT pinouts.
- Automatic sample vs background histogramming with alarm function and programmable region of interest.
- Two-bank list mode with programmable arrival time resolution.
- Radiation Portal mode with automatic background tracking and programmable alarm
- Built-in neural network (perceptron) for pulse shape discrmination.
- Software updates via USB
- Only 150mW (5V@30mA) power consumption, USB and serial interfaces with 3.3V RS232 and RS485.

Table of Contents

Fig. 1: Two of the MCA-2000 variants, one for PMT and one for SiPM. PMT-2000 MCA with high voltage unit on the left. On the right: SiPM-2000 MCA+detector assembly with SiPM array and 50mm NaI(Tl) crystal in detachable housing.

Compact SiPM-detector: Note that the SiPM-2000 electronics, including the SiPM-array all fit into the top part with the orange rim. The bottom grey aluminum housing is detachable and contains the 50mm NaI(Tl) scintillator.

1. Supporting documentation

Open-source software: The software is open source and mostly written in Python. For Windows, Bridgeport Instruments provides a software installer, which by default will create a C:\BPISoftV3 directory. That folder includes Python 3.7 with three added packages: ZMQ (www.zeromq.net), wxPython and Matplotlib v 3.2. ZMQ is used to implement the client-server behavior and it is accessible from more than 40 programming languages. Matplotlib is used for the graphics interface, and wxPython is used to create a traditional user interface with pull down menus.

The wxMCA folder that contains the MCA software can be placed anywhere on the hard disk. Inside is a folder called documentation and the best starting page is wxMCA/documentation/english/introduction/introduction.html which you can open in any web-browser. This set of linked documents contains a description of every control variable and every data field used by the MCA-2000.

2. Getting started

2.1 Using the USB interface

For USB communication the MCA-2000 uses libusb0.1 or libusb1.0 (Linux) and libusb_win32 on Windows 10. The Windows 10/11 libusb dll is digitally signed. When the MCA hardware is present during the installation, the installer will automatically link the MCA-2000 to libusb_win32.

However, it is also possible to simply copy the BPISoftV3 folder onto the C: drive and install the Windows USB-driver manually. This needs to be done only once, and Windows will later recognize any other MCA-2000, by its USB vendor ID of 0x1FA4 and the USB product ID of 0x0102 (PMT-2000) or 0x202 (SIPM-2000).

Open the Windows Device Manager and connect the MCA-2000 to a USB port. Wait until it appears in the Device Manager as an unknown device of type pmt2k or sipm2k.

Open the C:\BPISoftV3 folder and launch zadig-2.4.exe. Use Options/List All Devices to refresh the device list and select the device with the BPI vendor ID of 0x1FA4. To the right of the green arrow, select libusb-win32 (v 1.2.6.0) and click the big 'Install Driver' button below. Note that it may take up to 20 seconds for the screen to update - be patient. Once the 'Driver Installation Successful' message appears you can close the window. In the Device Manager you will now see the MCA-2000 listed under 'libusb-win32 devices'.

2.2 Launching the software

First launch the MCA Data Server. Look

inside the wxMCA folder. Under Windows, double-click on run_mds.cmd. Under Linux, launch wxMCA/mds/mds_server.py making sure you use a python 3.6 or higher installation that also includes ZMQ, wxPython and Matplotlib.

If the MDS reports no MCA found, the OS may have been slow in enumerating the MCA on the USB bus. Kill the MDS and launch it again.

Then launch the User Interface. Under Windows, double-click on run_wxMCA.cmd. Under Linux, launch wxMCA/wxGUI /MCA_Main.py from run_wxMCA.sh

The items in the menu bar are mostly self-explanatory. You can view results and edit instrument settings in a spread-sheet environment. To send the changed values to the instrument, you need to click on File →to MCA.

3. Energy Spectrum

The MCA provides fast and accurate measurements of energy spectrum and count rates. To acquire an energy histogram, select Display→Histogram from the menu bar. In response, a histogram panel will open. On that panel use [New] to erase the old histogram and count rate data and begin a new acquisition. The panel does not automatically refresh, so click the "Refresh" button to get an updated energy spectrum.

Use [Save] to append the energy histogram and count rate data to a default data data file or a file of your choice. The default file name is of the form xyz_histo_status.json where xyz is the detector serial number. By default that file is located in wxMCA/user/mca2k/data.

3.1 Sample spectrum

Low noise, high-quality spectra: Both, the PMT-2000 and the SiPM-2000 deliver energy spectra using low-noise front-end electronics. As a result one can set exceptionally low minimum trigger thresholds to achieve a wide dynamic range of measured energies of > 200:1

In the images below, the MCA electronic gain was adjusted to provide a maximum measurable energy of 1600keV. For both, the PMT and the SiPM sensors, the minimum trigger threshold was below 8keV – for a dynamic range of >200:1

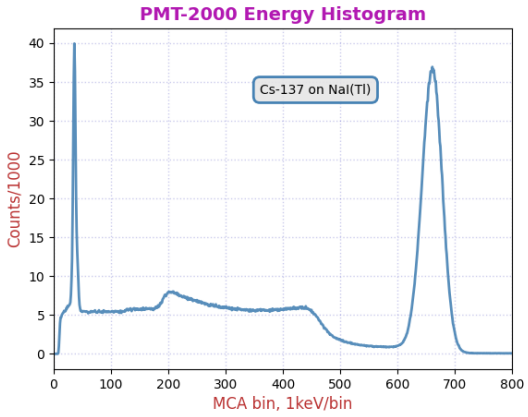


Fig. 2: Energy histogram of Cs-137 on a 50mm NaI(Tl) detector, captured with a PMT-2000.

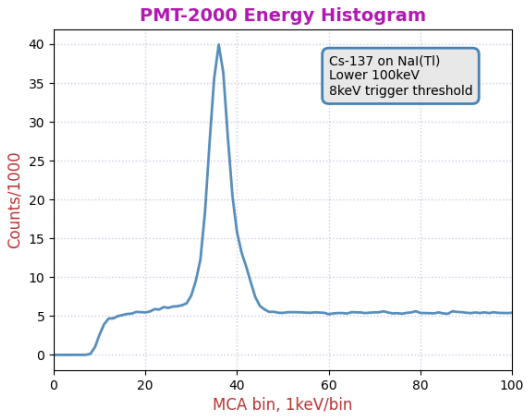


Fig. 3: The same energy histogram as in the previous figure, but with a focus on the low-energy region.

High-quality spectra with SiPM: We achieve the same performance on a 50mm NaI(Tl) crystal read out by a 5.76cm^2 array of SiPM. Again, the low minimum trigger thresholds yields an accessible dynamic range of measured energies of > 200:1

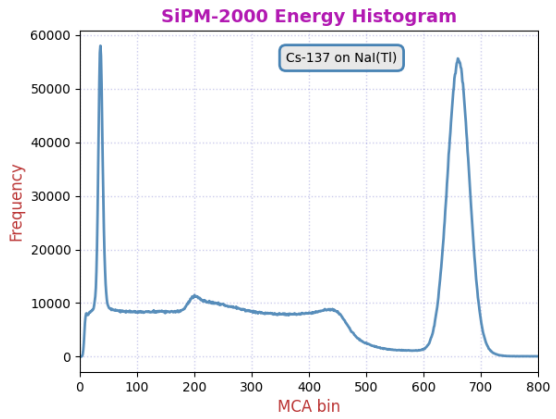


Fig. 4: Energy histogram of Cs-137 on a NaI(Tl) detector captured with a PMT-2000.

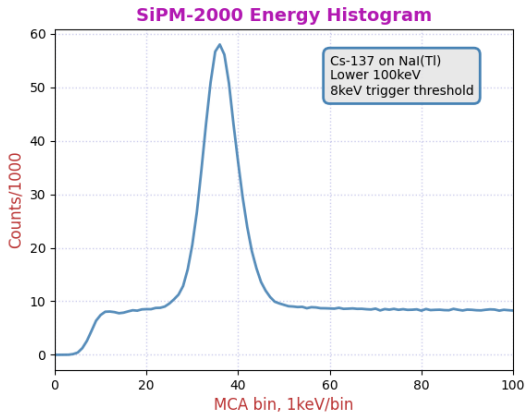


Fig. 5: The same energy histogram as in the previous figure, but with a focus on the low-energy region.

3.2 Adjusting operating voltage

You need to set the electronic gain and adjust the operating voltage to set the maximum measurable energy.

Then adjust the digital gain to achieve the desired MCA calibration in keV per bin

The maximum measurable energy is independent of the number of histogram bins used. Users need to adjust the operating voltage to change the gain of the SiPM or PMT. In the table below we recommend maximum measurable energies for different applications.

Max. Energy	*Comment*
1.0MeV	High gain for measuring low energies down to 20keV. Good for measuring contamination with Cs-137 and Cs-140 (fresh fission product), or Radium isotopes.
2.0MeV	Measure NORM up to K-40 at 1460keV
3.0MeV	Wide range covers all naturally occurring gamma-rays, even Tl-208 at 2615keV

Table 1: Cs-137 calibration for different tasks; NORM: Naturally occurring radioactive materials

How to determine the maximum measurable energy. In the (fpga_ctrl) display set amplitude to 1 to measure the pulse height. The maximum measurable pulse height is 900. Start a new histogram and record the peak position (P) of a known isotope; eg E=662keV of Cs-137. The maximum measurable energy is computed as

$$E_{\max} = E \cdot \frac{900}{P}$$

Hence, putting the pulse height peak of Cs-137 at 187mV will result in a maximum measurable enery of 3.18MeV.

There are four discrete electronic gains that can be applied. In (fpga_ctrl), select gain_select = 1, 2, 4, or 8 for increasingly higher electronic gains. Note that the highest electronic gain will slow down the input amplifier response and is usually only used for slow scintillators, such as CsI(Na).

For a continuous and smooth gain change, vary the operating voltage to move the pulse height peak. Access the operating voltage control by opening the (arm_ctrl) display. Enter a new operating voltage in the cal_ov field and hit enter. Use File→To MCA to write the voltage to the MCA.

Go back to the (Histogram) display and click (New) to start a new acquisition. If you change the operating voltage by 1%, the PMT gain will change by about 6%. For the SiPM-2000 the gain equation is

$$\text{gain} = 1.0\text{e}6 \cdot (V - 28.6\text{V})$$

Once the pulse height peak is in the right

position and therefore the maximum energy has been set, keep the operating voltage and the electronic gain constant.

Now adjust the digital gain to achieve the desired MCA calibration in keV per bin. Once the operating voltage has been established, set **amplitude** back to 0, and start a new histogram

Determine the measured peak position (P) and use the desired peak position (E) to determine the new ***digital_gain***

$$dg_{\text{new}} = dg \cdot \frac{E}{P}$$

With that the detector has been calibrated. See the calibration section 4 for a much more detailed description.

3.3 Count rate measurement

Observe how the count rate accuracy improves with measurement time. The MCA reports count rates together with their statistical errors. This gives users a useful tool. Manually, or programmatically, they can end a measurement precisely when the desired accuracy has been reached, which saves time and money.

The error (in %) is computed from the number of events as 100*2/sqrt(N), where N is the number of events. This is called the statistical 2-σ error. The true count rate lies within this error range with a 95% probability.

Note that the MCA corrects the recognized count rate using the known dead time per event. Hence the reported count rate is greater than the recognized count rate. The reported count rate is also an accurate estimate of the true number of counts per second. In fact the reported count rate should match the true input count rate with about 1% accuracy (systematic error) for input count rates up to 500kcps. The statistical count rate error is computed correctly using the number of recognized events.

3.4 Histogramming speed

The MCA has a non-extendable dead time equal to the hold_off_time per recognized event. For NaI(Tl) between 5°C and 65°C the MCA-2000 recommends a fixed hold off time for NaI of 1.2µs. The resulting throughput is shown in the next figure.

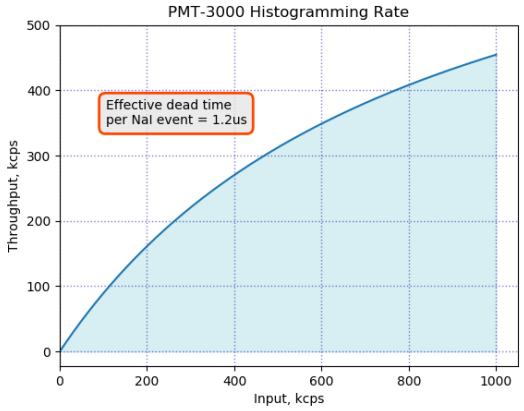


Fig. 6: Expected histogramming rate vs input count rate for a hold_off_time = 1.2µs; (ie typical for NaI(Tl) at room temperature.

4. Calibration

4.1 Theory of operation

The MCA-2000 uses a switched-gain input amplifier stage followed by an ADC with a 1V input range. The ADC samples the amplifier output voltage at a rate of typically 24MSPS.

For proper operation, the amplifier output voltage must fall within the 1.0V input range of the ADC, as shown in the two figures below.

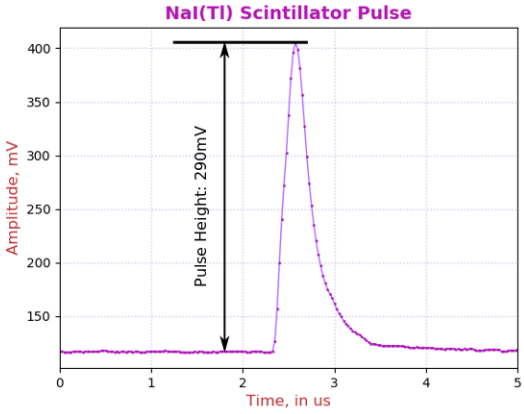


Fig. 7: A regular NaI(Tl) pulse.

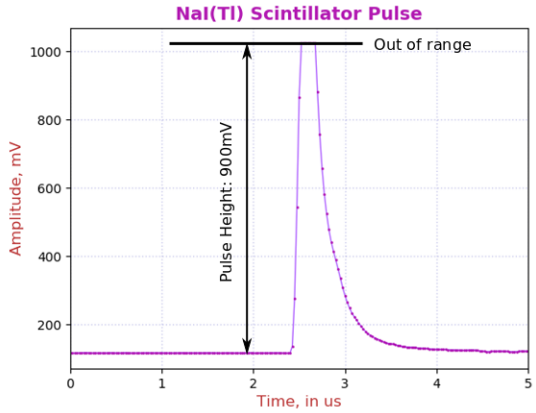


Fig. 8: An out of range NaI(Tl) pulse. Note how it cannot reach beyond 1023mV.

Pulse height is a measure of the difference between the maximum pulse voltage and the baseline from which it rises.

There is a built-in DC-offset of around 128mV and the maximum measurable voltage is shown as 1.023V. Hence the maximum practical signal pulse height range is around 900mV.

The first step of calibration at a chosen electronics gain, is to make sure that the maximum wanted energy can be measured by the ADC. At a fixed electronics gain, the applied operating voltage will determine the maximum measurable energy.

A very common calibration is to set the operating voltage such that the full-energy peak of Cs-137 (661.66keV) corresponds to a pulse height of 187mV. The maximum measurable energy is then 900mV/187mV×662keV=3.18MeV.

After adjusting the high voltage to set the maximum measurable energy, the user can now vary the digital gain to map the entire energy spectrum into an energy histogram of the desired size and calibration.

For example, we assume 24MHz ADC sampling rate and the recommended integration time of 1.2µs for NaI(Tl). We also assume that the Cs-137 662keV peak has an average pule height of 187mV. In that case, choose a digital gain of 13000 to place the peak in the energy histogram at 662keV, ie have an MCA calibration of 1.0 keV per bin.

Below we describe the calibration steps in detail.

4.2 Calibration file

Calibration controls are in autocal.json. This file can be found in the ./rad_config /Matplotlib_gui/config/ folder. Its contents are explained in the table below.

Variable	*Description*
cal_energy	The energy, in keV, for the calibration peak; eg 662 for Cs-137.
cal_pulse_height	The pulse height for the calibration peak; eg 187 for Cs-137 and 3.2MeV maximum measurable energy.
auto_update	0→ Off; 1→ Apply the new operating voltage and restart the mcaafter user clicks on "Calibrate" button.
keV_bin	Desired MCA calibration, as keV per MCA bin.
gain_exp	PMT's have a power law for their gain vs voltage function: $$\frac{gain}{gain_0} = \left(\frac{HV}{HV_0}\right)^{\text{gain\_exp}}$$ Typical values are 5.6 for R6231, 6.0 for CR105 and 7.5 for many other 10-stage PMT. For SiPM this parameter is ignored.
mass	The mass of the NaI crystal, in kg; used for computing the deposited dose rate from the count rate and deposited energy.

Fig. 8: The autocal.json data explained.

For use with Broadcom SiPM we use the following gain formula:

$$\text{gain} = 1.0\text{e}6 \cdot (V - 28.6\text{V})$$

The 28.6V are the sum of the typical 27V break through voltage and the 1.6V offset of the SiPM-2000 input amplifier.

You can manually calibrate the detector, in which case you have more freedom on how to do it. If you want to use the (Calibrate) button on the (Histogram) panel, then the software will always pick the tallest peak in the histogram, beyond 50 MCA bins, to use in the calibration computations. This works best for isotopes with a well-separated medium-energy peak; eg Cs-137 and Na-22. Co-60 is not recommended for the software-assisted calibration.

4.3 Step 1 of the calibration

First, adjust the operating voltage to set the maximum measurable energy. Open the(arm_ctrl) display and set the amplitude control to 1. Start a new histogram acquisition on the (Histogram) display by clicking on(New) . Wait until there are around 1000 counts in the peak maximum. Then click (Calibrate) on the (Histogram) display.

After 1 second, you will receive a message with the suggested new operating voltage. The software has applied the new operating voltage already if ***auto_update*** is set to 1.

There is some variation between the SiPM or PMT and you may have to repeat a second time to achieve the desired accuracy. Note that even if you do not know the exact gain exponent of your PMT, the process will still converge. It may just take another iteration or two.

Here is how to compute the ***cal_pulse_height (cph)*** :

$$cph = 900 \cdot \frac{\text{cal\_energy}}{E_{\max}}$$

4.4 Step 2 of the calibration

Now change the digital gain to achieve the desired MCA calibration in keV per MCA bin. From here on, the operating voltage and the electronic gain ***gain_select*** should be kept constant. Instead we use the ***digital_gain*** from the (fpga_ctrl) display.

Set the amplitude control to 0 to return to normal energy measurements, and start a new histogram acquisition. Wait for sufficient accuracy and click (Calibrate) .

This time, you will receive a notice of the new ***digital_gain*** .

The new value was computed from a linear formula

$$dg_{\text{new}} = dg \cdot \frac{\text{cal\_energy}}{E_{\text{peak}}}$$

and it will be accurate on the first try.

However, keep in mind that there is a statistical uncertainty in determining the actual energy of the histogram peak. You can compute the 1-σ error from the fwhm (in %) and the number of net counts (N) above background in the peak as

$$\varepsilon = \frac{\text{fwhm}}{2.3655 \cdot \sqrt{N}}$$

5. Gain Stabilization

All MCA-2000 offer gain stabilization that keeps the gain, the trigger threshold and the maximum measurable energy constant. NaI(Tl) is the most commonly used scintillator, and it also one of the most challenging. Its brightness and pulse shape change with temperature, while the application engineer wants to keep constant not only the MCA gain (keV per bin) but also the trigger threshold and maximum measurable energy that is independent of temperature.

All MCA-2000 offer a user-programmable set of lookup tables to change the operating voltage and the digital gain to achieve this goal. In addition, for NaI(Tl) the MCA also vary the hold-off time with temperature to avoid retriggering on the tail end of the same pulse.

Since code and data needed for the gain stabilization are embedded in the MCA, no user software is required and the calibration moves with the detector.

The PMT-2000 also offers LED-based gain stabilization to counter PMT aging. Vacuum photomultiplier tubes lose gain over time, even under light to moderate loads. For example the popular R6231 PMT loses half its gain after an accumulated anode charge of about 50C. Distributed over one year, this equivalent to an average anode current of just 1.6µA, with a 6% gain loss per month. With a built-in LED that injects light into the back of the photomultiplier, the PMT-2000 can gain-stabilize on the LED response and counteract the PMT aging.

5.1 General theory of operation

All types of gain stabilization use lookup tables, and users can supply their own. The standard code includes a 64-long float array with three look up tables **(LUT)**. The MCA frequently determines the temperature and adjust the operating parameters according to the values found in the lookup tables.

The LUT depend on the scintillator and on one DSP setting, namely the integration time. For NaI(Tl) the LUT supplied by the factory use an integration time of 1.2µs. For the PMT-2000 and NaI(Tl) the LUT supplied by the factory was derived for an integration time of 1.2µs. That value remains fixed over the entire temperature range, as it provides the best energy resolution for small and large energies at all temperatures. For the SiPM-2000 and NaI we suggest a fixed integration time of 1.5µs.

But the hold-off time has to be varied. NaI(Tl) pulses become very long when the crystal temperature falls below 0°C. Consequently, the hold-off time has to be increased at low temperatures to avoid retriggering on the tail end of the same pulse. Since this phenomenon is determined by the scintillator physics, it has been implemented as an immutable function in the MCA, instead of a lookup table. The function is used if arm_ctrl["fields"]["cal_scint"]==1.

The implemented hold-off vs time function for NaI(Tl) is

$$\max(1.25, (0.634 + 1.27 \cdot \exp(-T/32.26)))$$

in µs and with T being the temperature in °C.

How the LUT are constructed. Developers may find that they need to construct the correct LUT for their detectors, as they may use a different scintillator or a different PMT. To this end the detectors needs to be run through a temperature cycle. The detector is recalibrated to desired specification at each temperature and the operating voltage, digital gain and possibly the LED value are recorded. Using interpolation one then constructs the LUT at the fixed temperature steps.

Keep in mind that many crystals don't allow for rapid temperature changes or they will crack. 10°C per hour is a safe bet for a 50mm NaI. Stay below 10°C per hour for a 75×75mm NaI, and always follow the manufacturer's recommendations.

Secondly, it takes a while for a detector to attain steady state. The vacuum PMT needs an hour and the crystal characteristic time depends on the material and the size. For a 50mm NaI it takes about half an hour, for a 76mm NaI it takes 1.5hrs. Hence, it is necessary for a precise measurement, to maintain the detector at a fixed temperature for 2 to 4 hours before moving to the next.

Thermal equilibrium is necessary. Keep in mind that the PMT and the scintillator both have different temperature coefficients. If the two are not in equilibrium, there is no easy way to provide gain stabilization. Hence an outdoor detector should be packaged with good thermal insulation to achieve thermal time constants of 2 hours or more – to ensure that all components are in a steady state situation. However, care must be taken not to insulate the electronics. Even if it only dissipates 300mW, that small amount of power stills needs to drain away to avoid self heating.

When using SiPM: SiPM-based systems react more quickly, because the SiPM-array has very little thermal mass. In uncooled systems its temperature may be close to the crystal or halfway between the electronics temperature and the crystal temperature, depending on the assembly.

In cooled systems the SiPM-array is well insulated from the crystal and its temperature remains constant as long as the Peltier cooling loop remains in regulation.

As a result, insulation for SiPM-based systems needs to be, at the minimum, just good enough to avoid cracking the crystal through temperature shock.

arm_cal registers and fields	
Index: name	***Description***
0: lut_len	Number of entries in the LUT; default is 19, 2..19 are allowed
1: lut_tmin	Minimum temperature in the lookup table; Typically -30°C
2: lut_dt	Temperature step size in the lookup table; Typically 5°C
[3:22]: lut_ov	Change of operating voltage vs temperature
[23:42]: lut_dg	Change of digital gain vs temperature
[43:62]: lut_led	Change of LED target vs temperature
63: lut_mode	int(lut_mode)&0x1 → lock bit, set to 1 to prevent the user from reading the arm_cal data from the MCA.

The arm_cal registers and fields.

5.2 Gain stabilization summary

Standard software recognizes these gain stabilization modes (gsm):

- 0 ⇒ Off; Use when calibrating a detector.
- 7 ⇒ Suspend; Keep all parameters as they are.
- 1 ⇒ Use temperature lookup for change of operating voltage and digital gain; adjust hold-off if required.
- 2 ⇒ Compare measured LED value to expected value and adjust operating voltage accordingly. Use LUT for digital gain and compute hold-off time as needed.

arm_ctrl for detector calibration		
Name	***gsm***	***Description***
cal_ov	0,1,2	Operating voltage when the detector was calibrated
cal_temp	1,2	Temperature (in deg C) at which the detector was calibrated
cal_dg	1,2	Digital gain when the detector was calibrated
cal_scint	1,2	Scintillator type; 1⇒ NaI(Tl), adjust hold-off time vs temperature.
cal_target	2	Target value for response to LED; used with gain_stab=2

The arm_ctrl registers and fields concerning detector calibration; The gsm column lists the gain stablization modes for which this parameter is used.

LUT needed for detector calibration		
Name	***gsm***	***Description***
lut_ov	1	Operating voltage
lut_dg	1,2	Digital gain
lut_led	2	LED average

The lookup tables (LUT) needed detector calibration; The gsm column lists the gain stabilization modes for a given LUT is used.

5.3 Gain stab. mode: gsm = 0

Off. Gain stabilization is turned off and the target operating voltage the MCA will set is the one requested in arm_ctrl["fields"]["cal_ov"].

5.4 Gain stab. mode: gsm = 7

Suspend. This is different from gsm=0. During gsm $\neq 0$ the gain stabilization algorithm may have changed the target operating voltage to a temperature dependent value that is different from arm_ctrl["fields"]["cal_ov"]. Setting gsm=0 would revert to that original value, while gsm=7 will leave the value unchanged. The same applies to the digital gain and the hold-off setting.

5.5 Gain stab. mode: gsm = 1

This mode relies on just the look up tables and the measured temperature. It is implemented for all MCA-2000. The units ship with LUT determined for NaI(Tl) at an integration time of 1.25µs

5.6 Gain stab. mode: gsm = 2

This mode adjusts the voltage using an LED measurement It is only implemented for the PMT-2000 series. The units ship with LUT determined for NaI(Tl) at an integration time of 1.25µs

The driving circuit for the LED used in the gain stabilization is electronically temperature compensated, but the compensation is not perfect. Since also the scintillator brightness changes with temperature, the ratio of the LED value and the full energy peak from a gamma-ray will be temperature dependent. Hence, there needs to be a lookup table to tell by how much the LED target value needs to be shifted as the temperature changes, in order to keep a full-energy gamma-peak constant.

Use an LED when expecting PMT aging. The purpose of the LED is to counteract PMT aging. This applies mostly to remotely installed detectors where a frequent recalibration is not possible. Whenever a detector can be recalibrated monthly and only sees light to moderate loads (anode current < 1µA with round the clock operation), then an LED system is not required.

Since aging is not a concern for SiPM, the SiPM-2000 series does not offer an LED option.

Adjusting the LED. In the arm_ctrl panel you can set the led_width parameter. Keep the LED pulses short, between 3µs and 6µs. The ARM processor runs the LED at a fixed rate of 20Hz. Setting led_width=0 turns the LED off.

The ARM processor periodically updates the led_val, when it has acquired enough LED pulses to compute an accurate average. The number of pulses to average is set by the cal_events parameter. When the detector has been calibrated, read the LED value from the led_val field on the (arm_status) display and enter it into the cal_target field on the (arm_ctrl) display.

LED-based gain stabilization When the gain stabilization relies on measuring the LED value, the ARM will perform a gain update every time a sufficient number of led events have been measured, cf cal_events.

Keep the LED pulses short, between 3µs and 6µs. When gain stabilization is off, you can see the LED values histogrammed. Mostly they will fall into a narrow peak of with an "energy resolution" of 1% to 2% fwhm. At high input count rates from a source (> 20kHz) you will also see a small fraction of events classified as LED that are a pile up of an LED pulse with a gamma-ray. These are rare, but they do slightly affect the accuracy of the LED average measurement. Keeping the LED pulses short helps to suppress these unwanted events.

The LED pulse is shorter than the drive pulse. The LED light pulse about 0.9µs shorter than the drive pulse. LED pulses can be viewed in the pulse display. They appear as rectangular pulses.

Usually the LED pulse height is adjusted at the factory to about 200mV when the detector is calibrated to have a maximum energy range of 3.2MeV. This allows operation with a higher gain and a lower energy range of down to 1MeV max, while keeping the LED pulse within ADC range.

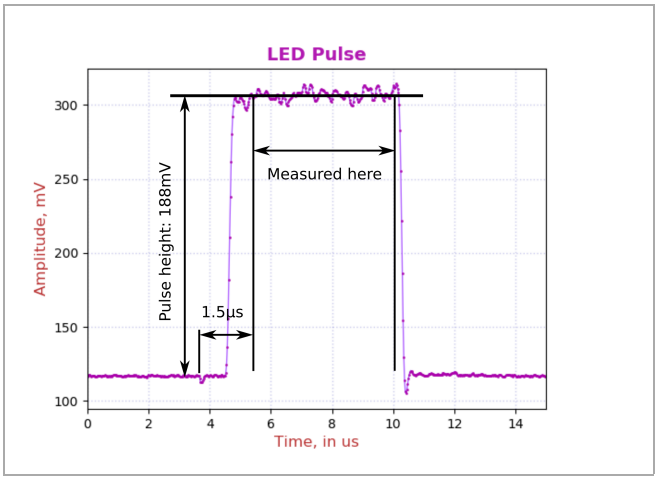


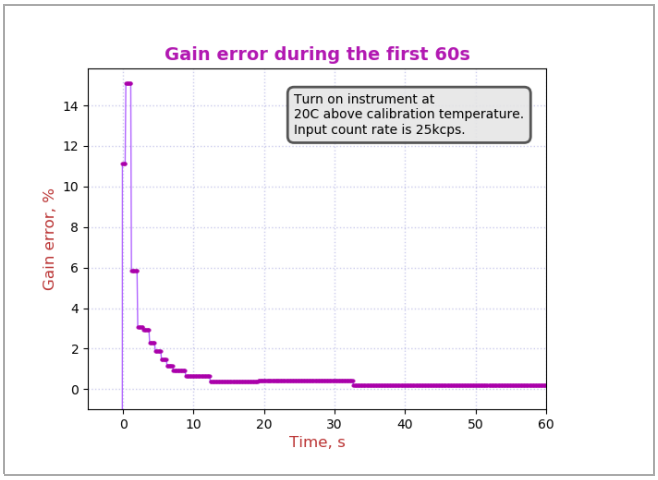
Fig. 9: A wide LED pulse to demonstrate how it is being measured.

5.7 Dynamical performance for gsm = 2

The gain error falls to < 1% in 10s. If an instrument that was calibrated at room temperature is left in a hot vehicle and turned on at a temperature much different from the calibration temperature, its initial gain settings will be all wrong for the new temperature. For gsm=1 the adjustment will be immediate, as the instrument only requires one temperature lookup to adjust to the new situation.

For LED-based gain stabilization (gsm=2)

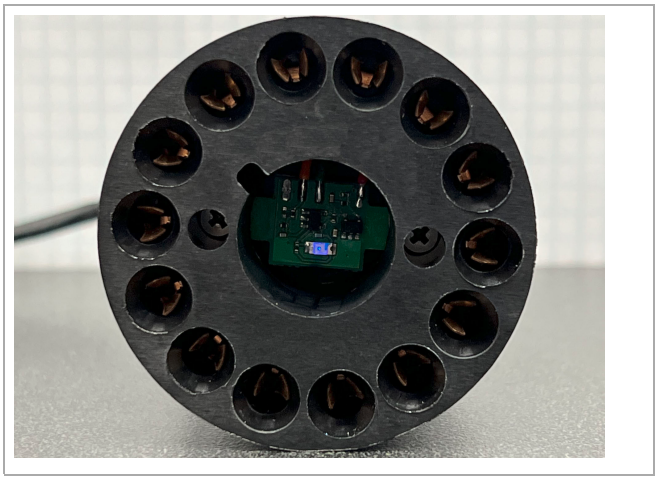
there is a short time lag. The instrument needs to measure the new LED value and then step by step adjust the high voltage to make the LED value meet the computed LED target. As shown in the figure below, this process takes less than 10 seconds. In the beginning the LED blinks at 1000Hz, and the high voltage is updated about every 1.0s. Once the LED value is close to the LED target (within 1%) the instrument gradually reduces the LED blinking frequency to 20Hz to reduce the load on the PMT and avoid accelerated PMT-aging.



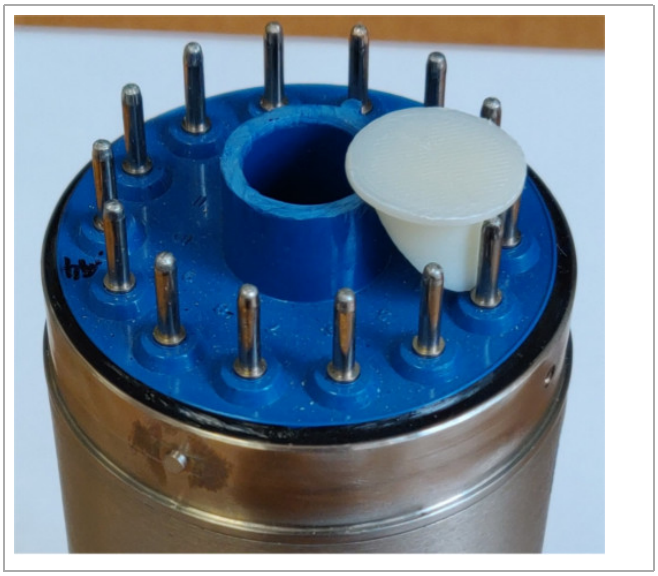
This is the response of the gain stabilization algorithm to turning on the detector at a temperature that is different by 20°C from the calibration temperature. Within 10 seconds the gain error falls back to below 1%.

5.8 Detectors and LED

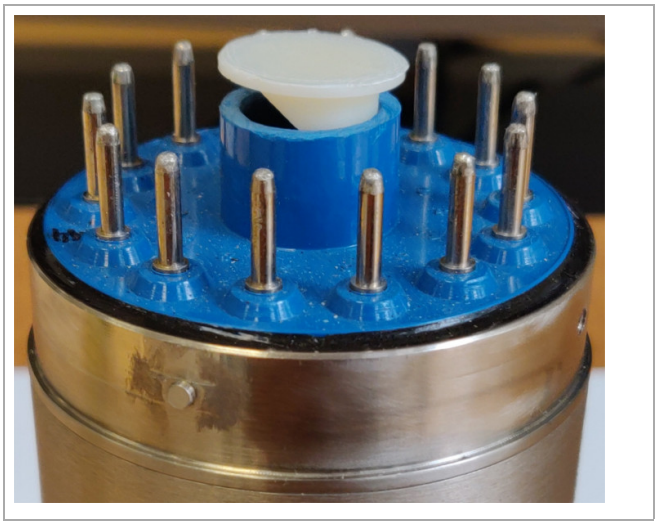
LED in the MCA not the detector: The PMT-2000 can be equipped with an LED for the purpose of gain stabilization. The LED is mounted on the underside of the high voltage unit. The LED shines its light through the back of the PMT.



View into a high voltage base with embedded blue LED.



Detector with an opening for the LED. The white light diffuser is shown on the side.



Here the white light diffuser is partially inserted.

Retrofitting is possible: Most detectors can be retrofitted to accept the LED light. Carefully cut open the back of the PMT socket. Once the MCA is plugged onto the PMT, the assembly is light tight again. In integral detectors the PMT is glued directly to the crystal and the environmental seal is usually between the side of the PMT and the magnetic shield. Hence, opening up the back of the PMT does not cause a problem.

Adjusting the diffuser: First keep the LED off `led_width=0` and adjust the gain of the detector as desired. Then, insert the diffuser, turn the LED on `led_width=2µs`, and acquire a few pulses. You will see a rectangular LED pulse with a width that can be controlled with the `led_width` control. For now we are only concerned about the pulse height.

In the MCA the LED is placed off-center and the light diffuser is asymmetric. Turning the light diffuser changes the amount of LED light transmitted into the PMT. You have to unplug the PMT-2000 from the PMT in order to turn the light diffuser. Note that you have to power down the MCA before plugging and unplugging – disconnect the USB cable to be sure. Never hot-plug the PMT-2000. It would damage the device.

Aim for LED pulses of around 200mV ±50mV above baseline, cf Fig. 9. Then glue the light diffuser into the opening. For precise operation, the diffuser must be securely glued in, otherwise it will change its position during temperature cycles.

5.9 PMT-2000 Gain Accuracy

Gain control with 15-bit DAC: In the PMT-2000 we use 15-bit of a 16-bit DAC to control the PMT voltage up to 1500V. Discretization errors in the gain due to the DAC differential non-linarity are limited to less than 0.1%.

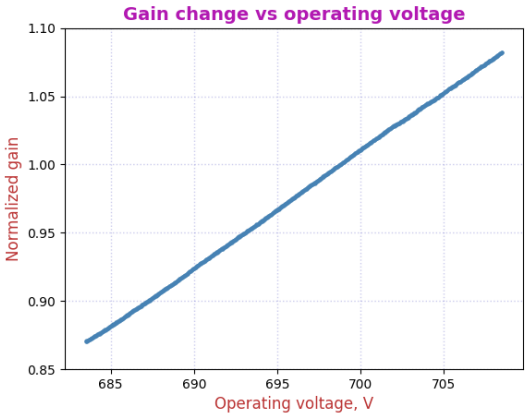


Fig. 10: Gain change of the PMT in response to a smooth change of the requested operating voltage. The voltage was increased in 25mV steps, and the peak position was measured at each step.

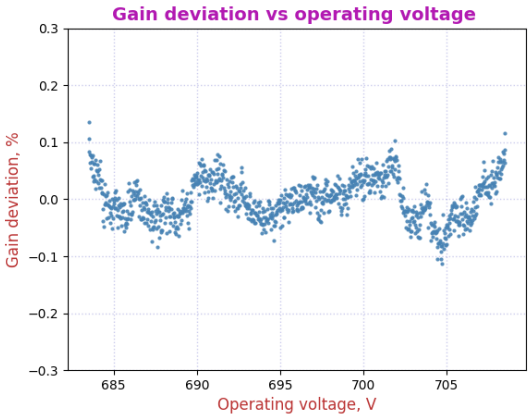


Fig. 11: Gain deviation from a smooth fit derived from the data in the previous figure. The gain error due to DAC nonlinearity is below ±0.1%.

6. Sample vs Background

The instrument provides automatic background subtraction. To accurately measure the radioactivity of a weak source, or a weakly contaminated sample can be a complicated process. However, in all cases the process requires a precise measurement of the background and the ability to correctly subtract that background spectrum from the sample spectrum. And the MCA-2000 greatly simplifies this task.

The MCA-2000 stores sample and background data on the instrument and reports the difference. From the dashboard, open the "Sample − Bck" panel. From top to bottom you will see three spectrum displays. The one at the top is the background spectrum. Initially this will be empty. The middle panel has the sample spectrum, and initially it will show the previously acquired spectrum. Finally, the bottom panel has the difference spectrum, which initially equals the middle spectrum.

To acquire a background spectrum, remove all unwanted radioactive materials from near the detector and click on "New" next to the top spectrum panel. You can increase the accuracy of the results by measuring the background spectrum 4 times longer than a typical sample spectrum. Click "Ref" next to the top spectrum to update the display.

Once you have counted the background long enough, click on "New" next to the middle spectrum panel. This starts a new sample

spectrum acquisition and stops the background acquisition. The background spectrum will now have stopped and remain unchanged.

If you simply acquire a sample spectrum under the same conditions that you acquired the background spectrum, you will see a similar spectrum grow; click "Ref" next to the middle spectrum to update the display.

Every time the display is updated, the software reads three spectra from the MCA: the background, the sample and the difference spectrum. You will notice, that even in the absence of any new radioactivity the difference spectrum is not exactly a flat, empty spectrum. The reason for that is that the gamma-rays arrive randomly, and there is therefore a natural statistical variation between acquired spectra.

The instrument computes probabilities and alarms. The MCA computes the probability that the count rate in the difference spectrum was caused by nothing but background. The message box next to the difference spectrum shows the result of that computation. You can confine this computation to a region of interest (ROI) by adjusting roi_low and roi_high in the alarm panel. For instance you can increase the sensitivity to Cs-137 by setting [roi_low, roi_high] to [580, 780]. The ROI is entered in units of raw histogram bins. For instance at 2keV/bin, the previous [roi_low, roi_high] would be [290, 390].

Finally, you can set an alarm threshold. When the probability that the measured sample radioactivity is caused by nothing but background is too low, the MCA can raise an alarm. The alarm threshold can be set in alarm_ctrl via the alarm_thr variable. For example, a value of 1.0e-3 is interpreted as a probability of 1 in 1000.

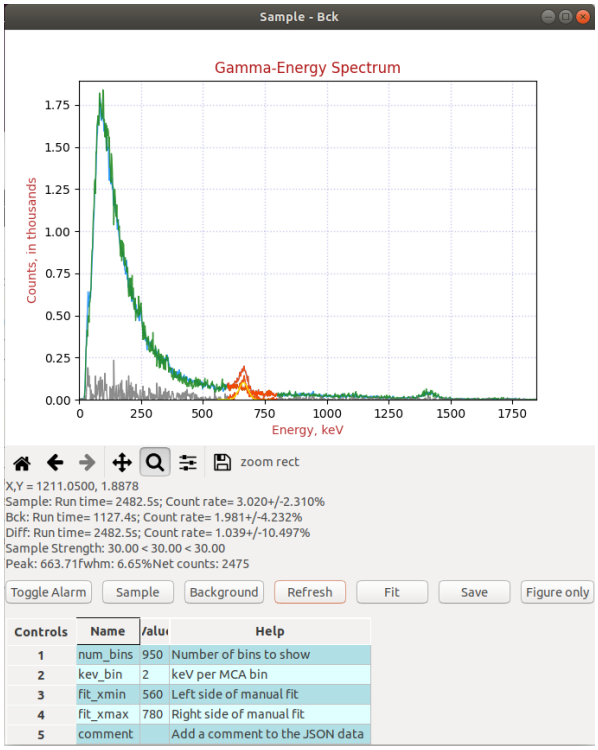


Fig. 12: Example of a background histogram and a nearly identical sample histogram (blue and green).

In figure 12 we show an example of a natural background histogram (green) and a sample histogram (blue), with a weak Cs-137 source at 2.5m distance.

The run-time corrected difference spectrum is shown in grey. The region of interest used for the alarm computation is indicated in orange.

The instrument reports the signal strength and confidence interval. The excess of count rate over background caused by a small amount of radioactivity is called the signal. Here the signal strength is reported as $-log10()$ of the above-mentioned probability. The maximum signal strength that will be reported is 30.

The instrument also reports the 95% confidence interval for the measured source strength. Users can thus make an informed choice on how to set alarm levels and thresholds for taking a decision. The mathematics is described in the math section below.

7. Mathematics of Errors and Alarms

This section is for the reader with an interest in the mathematics that is used by the instrument.

7.1 Count rates and their errors

When measuring count rates, the instruments counts events and the elapsed time. Systematic errors for measuring the run time are very small and are ignored here. The dominant variation come from the fact that the number of events during a given time interval is Poisson-distributed. When the total number of events counted is greater than 100, we can approximate the resulting Poisson distribution with Gaussian normal distribution with an average of (μ = number of events) equal to the number of events and a standard deviation of $\sigma = \sqrt{\mu}$ For a given measurement the instrument reports the 2-σ relative count rate error

$$\epsilon = 2/\sqrt{\mu}$$

Dead time does not contribute to statistical error. The instrument knows the incurred dead time per recognized event and computes a live time as run time minus dead time. The reported count rate is events/live_time.

7.2 Background subtraction

Background time can be different from sample time. The instrument can subtract a background spectrum from a sample spectrum even when the two have been acquired for different amounts of time. The calculation for the difference histogram bins is

$$\mathrm{diff} = \mathrm{sample} - \frac{\mathrm{sample\_time}}{\mathrm{back\_time}} \cdot \mathrm{back}$$

Here, diff, sample, and back are the histogram bin contents of the three energy spectra.

Count rates are measured by summing all events within the alarm region of interest, and dividing by the respective live times. Count rate errors are computed from the number of events in the region of interest.

The difference count rate is computed as a direct difference, $R_D = R_S - R_B$, from the dead-time corrected sample and background rates.

The count rate error for the difference spectrum has to be computed from the two uncorrelated errors of the sample and the background counting:

$$\epsilon_d = \frac{\sqrt{(R_S \cdot \epsilon_S)^2 + (R_B \cdot \epsilon_B)^2}}{R_S - R_B}$$

Notice that for very small difference count rates the resulting relative error can be quite large. Further, a difference rate with a large relative error, eg 1.0cps ± 200%, simply means that the result is compatible with a difference of 0cps.

7.3 Computing probabilities

For a given sample measurement time, T_S, the MCA knows how many background events to expect on average: $N_B = T_S \cdot R_B$. It compares that number to the number of actually measured events, N_S. Using correct Poisson statistics, not a Gaussian approximation, it calculates the probability that N_S could have been caused by the known background: $P = P(N \geq N_S | N_B)$. If $N_S \gg N_B$, this probability will be very small. If it falls below the given alarm probability (**alarm_thr** in the alarm panel), the MCA can raise an alarm. In other words, a stronger signal causes a lower probability.

However, a user might prefer to use a measure of the signal strength that increases with the signal strength. Hence we define a signal strength as $A = \log_{10}(1/P)$

For the confidence interval of the signal strength, the instrument reports a narrower measure. It computes the two opposites we get by assuming 1-σ errors in the background count and the sample count, but pointing in opposite directions. The resulting rectangular two-dimensional confidence interval contains about 71% of all cases.

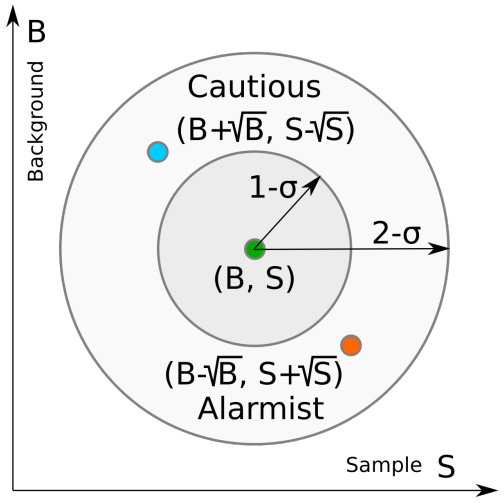


Fig. 13: Two dimensional confidence interval.

8. Radiation Portal Monitor

The MCA can act as a radiation portal monitor with background tracking and a programmable alarm function. In a Radiation Portal Monitor (RPM) the data acquisition unit must perform a number of tasks:

- Continuous background measuring;
- Deliver an alarm within a few seconds after the time of closest approach by a source;
- Keep a few seconds of alarm history so that a polling host will not miss an alarm;
- Automatically reset the system on a continuous alarm to remain operational;
- Support a programmable alarm threshold based on a false-alarm rate;
- Support an adjustable background averaging time, eg 30s for a big gamma-ray detector vs 5 minutes for a neutron detector,
- Be able to recognize passing sources without the aid of an occupancy detector.

The MCA-2000 implements all that functionality within its embedded 32-bit ARM processor. An alarm can be sent as digital pulse with programmable width. The output is fed by an OpAmp that is short-circuit proof. The chip can 50mA at 3V or 5V – enough to drive an LED light or a buzzer. Alarms can also be read via USB or the serial communications interface.

8.1 Theory of operation

In the discussion below, times are given in units of time slices. Typically a time slice is 100ms long and the processor perform one RPM computation step every time slice. In some software versions, time slices can be set in multiples of 50ms.

At first the instrument measures the background. On start, after power on or a reset, the ARM processor begins to measure backgrounds. Until backgrounds are know with sufficient precision, the ability to alarm is disabled. This wait time is user programmable, cf **wait** in the RPM panel. Typically the wait time is a fraction of the background averaging time, eg 20% to 50%. A shorter wait time yields an active RPM earlier at an elevated risk for a false alarm until a full background averaging time has passed.

As long as there is no alarm, the events counted during one time slice are considered background events. Using the region of interest **(roi_low, roi_high)** in the RPM panel allows the user to constrain the attention on a part of the energy spectrum.

The background counts per time slice are averaged using geometric averaging. Using the $w = 1/Bck\_avg$ from the alarm panel the processor applies the formula: $B_{n+1} = B_n + w \cdot (N - B_n))$. The background averages are then stored in a 128-long FIFO. This way the instrument can look back 128 time slices to find an untainted background after an alarm has occurred.

We have $0 \leq w \leq 1$ for the weight. If the background at time $t = 0$ changes from B_0 to B_1, The background average responds to a step function with an exponential function of $B(t) = (B_1 - B_0) \cdot (1 - \exp(-t/\tau)$ where $\tau = 1/w$. The standard deviation of the averaged background, ie its noisiness, improves as if $1/w = \mathrm{Bck\_avg}$ samples had been averaged.

What happens when a passing source causes an alarm? The instrument continuously

computes a moving window sum of the last L time slices and compares that sum to the number of expected background counts during L time slices. For every time slice the instrument computes the probability that the observed counts could have been caused by the known background, cf the mathematics section above. It computes $P = P(N \geq N_L | N_B)$, with $N_B = L \cdot B$. If that probability is less than ***epsilon*** on the alarm panel, it will trigger an alarm.

An alarm is typically raised no later than $L/2$ time slices after the closest encounter. For a very strong signal, that alarm may be raised right at the leading edge of the L period; for a weak signal the alarm may be raised about $L/2$ time slices after the source has passed the point of closest approach. In many applications the allowed latency $L/2$ is about 2 seconds, which allows for a 4s summation time. At a time slice length of 100ms this means $L = 40$. In those applications, the time during which the radiation signal is detectable is 4s to 8s and and simulations show that 4s to 6s summation times produce the highest sensitivity; ie lowest minimum detectable activity.

No missed alarms. The instrument keeps a history of the alarm status with a maximum length of 128 time slices. This is controlled by the ***history*** parameter in the alarm panel. A polling device, via serial interface or USB, will be informed if there was an alarm present in the last ***history*** time slices. This way a polling host may have a a latency of 128 time slices (12.8s) and still will not miss an alarm.

As long as there is an alarm present in the alarm history FIFO, the instrument will suspend background updates and use the oldest background average in its memory as the best estimator of the background.

Automatic device reset ensures the instrument remains functional when the background suddenly increases. Consider the case of a radiation detection backpack. The wearer walks into a room where radioactive material is present, and this causes an alarm. After ***history*** time slices, the instrument automatically resets and starts to accept the elevated radiation level as the new background. After a ***wait*** period (30s typically) the instrument will again be ready to alarm if the operator suddenly encounters an even higher radiation level.

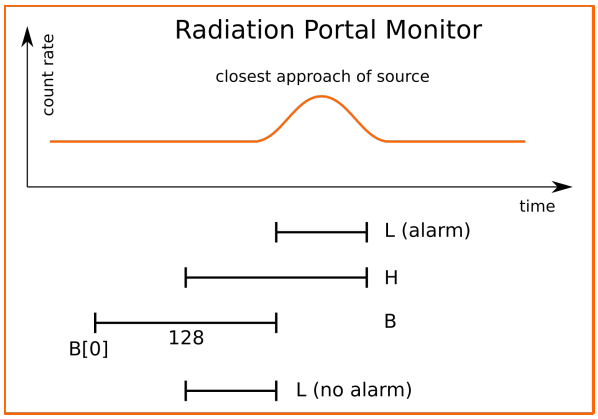


Fig. 14: Programmer's model of the Radiation Portal Monitor software.

9. Two-channel Logger

A chart recorder for rapidly changing situations: The MCA-2000 can log two parameters from the arm_status array in time steps as small as 50ms. Those data include status data (such as temperature or operating voltage) or radiation data such as counts per time step and alarm probabilities.

The arm_logger is implemented in an 8kB general purpose region of memory. For example users can log the change in operating voltage in response to a rapid change of count rate – to verify that the gain remains stable in such circumstances. Or the user may log net counts over background and the resulting alarm probability – to study the performance of the portal monitor alarm function. In case custom software is implemented on device the logger size may be reduced, or the logger may be omitted altogether.

The logger can be used to monitor rapidly changing parameters in time intervals ranging from 50ms to 12.75s. The logger creates running logs for two parameters. The parameters are chosen by the their index in the arm_status register. The parameter index is also indicated in the wxMCA arm_status table.

See the software documentation for details.

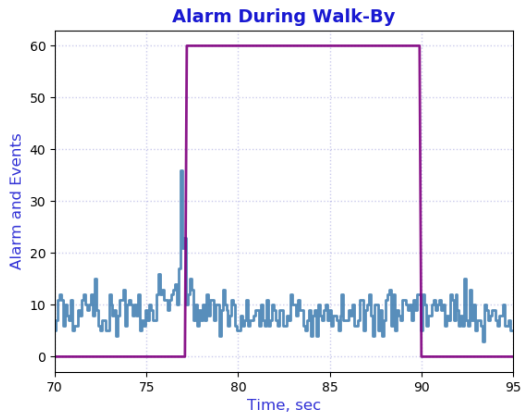


Fig. 15: Logger example: Events per 100ms time slice and result of the built-in alarm computation while a check source is moving past the detector.

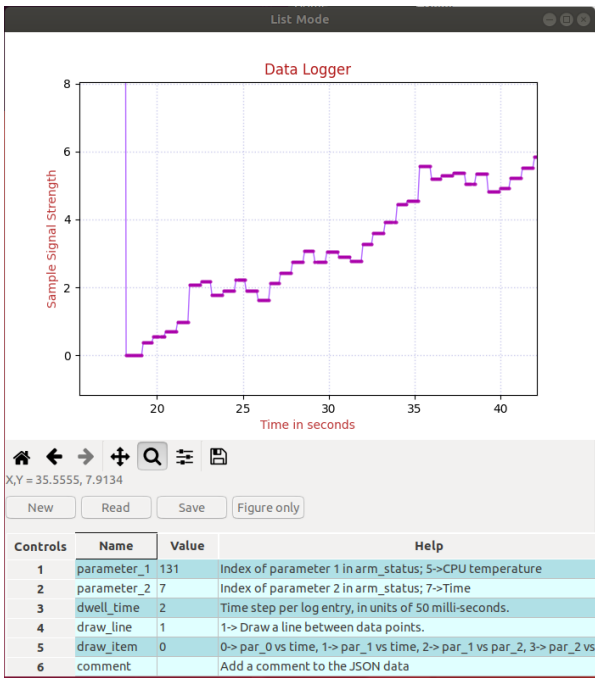


Fig. 16: Logger example: After having measured the background, a new sample run is started at T=13s. The sample is slightly more radioactive than the background and the certainty of that rises with time, cf "Sample vs Background" section.

10. List mode

Lossless event-by-event data acquisition: The MCA-2K devices (PMT-2000 and SiPM-2000) have a built-in dual-bank list mode capability. Each bank operates independently and has room for 511 events. While one memory bank is actively acquiring data, the other is stopped and can be read out by the client. This ensures loss-less list mode data acquisition up until the USB data bandwidth limit, of typically 125kcps.

List mode operation: The user writes ***clear_listmode*** and ***lm_buffer*** to arm_ctrl in the same command, to at the same time select the active buffer and clear the event counter of that buffer. The next read is then automatically directed to the inactive buffer.

In typical operation, the user switches buffers at a fixed rate and reads the inactive buffer. Note that when a buffer is cleared, only the event counter is set to zero. The event data in the buffer are not zeroed out to avoid dead time. The arm_listmode class takes that into account and only reports data for as many events as are indicated by the event counter in the first word of the listmode data buffer.

Alternative operation: If the user wants to always read complete buffers with 511 events, they can start an acquisition as before and frequently check the listmode_done bit in results resgister RR2. The MDS reports that in arm_status["user"]["listmode_done"].

When the bit is set, switch data acquisition the other buffer, and read the data from the now inactive buffer.

Dependig on count rates and how often the listmode_done bit is checked, some data loss may occur in this kind of operation. Custom versions of the firmware may allow for automatic memory bank switching when one data bank fills up.

11. Summation Weights

Normally the MCA-2000 measures a pulse energy by first subtracting the DC-baseline, and then by summing all ADC samples over an integration time. All ADC samples are weighed equally. But the MCA-2000 offers a more fine-tuned control: $E = \sum_k y_k \cdot w_k$

Improve the performance of certain scintillators. In some unusual scintillators the energy resolution can be improved if the summation weights w_k are not all equal. One example is SrI(Eu) where in big crystals the self-absorption of its own scintillation light can create significantly different pulse shapes for the same amount of energy deposited in the crystal; say for 662keV. The scintillator grower may have a recommendation, or the user can apply iterative or machine learning methods to find an optimized set of summation weights.

Most often, however, summation weights will be advantageously used to create powerful pulse shape discrimination algorithms as discussed in the section.

Within the FPGA, 1024 consecutive summation weights are stored, which covers integration times up to 8.53µs (@120MHz ADC speed) or up to 51.2µs (@20MHz ADC speed).

Ignore if not needed. By default all summation weights are set to 32767. For the purpose of energy measurement, the weights are considered to be unsigned 16-bit integers, with a range from 0 to 65535. Unless this feature is used, users can completely ignore this.

There is only one set of weights. When psd_on=0, the weights are treated as uint16_t and are used in the computation of the energy sum. When psd_on=1, the weights are used for the psd sum instead of the energy sum.

The software ships with a big examples section, where the user can find short python scripts to program custom weights into the FPGA, and even store them in the ARM processor's non-volatile memory.

12. Pulse Shape Discrimination

PSD is controlled by the user. Both MCA-2000 support a very general, patented, pulse shape discrimination method that gives the user great control. The user can define 16-bit signed weights to apply to each ADC sample in a pulse after triggering. $P = \sum_k y_k \cdot w_k$. For each event the firmware classifies the event as being type 0 or 1.

When PSD is turned on, the firmware separates type 0 and 1 events into two separate halves (2Kx32) of the available histogram memory. Type 0 events are always histogrammed in the lower half, while type-1 events are recorded in the upper half. For both types of events there is a separate event counter to measure count rates.

When PSD is turned on, the firmware separates type 0 and 1 events into two separate halves (2Kx32) of the available histogram memory. Type 0 events are always histogrammed in the lower half, while type-1 events are recorded in the upper half. For both types of events there is a separate event counter to measure count rates.

This feature is compatible with loss-less histogram acquisition where the histogram is split into two separate banks. In that case there will be four 1Kx32 spectra.

See wxMCA/examples on how to program the weights. Consult the examples folder of the software to find code examples that write summation weights to the FPGA of the MCA-2000 and to the non-volatile memory of the ARM processor.

Factory default: The factory default for the weights is 32767, which is very nearly the same as 1.0. When psd_on=0, the weights are used for regular energy measurements, and having all weights to be equal and near 1.0, is a reasonable default.

When psd_on=1, the weights are used for pulse shape discrimination (PSD). The default set will not provide any PSD, and the user must program a new set of weights. This is discussed in detail below.

12.1 Where does the pulse begin?

In order to select the weights it is helpful to see exactly where a pulse starts, ie to which ADC sample the first weight is applied. To this end set `psd_on=1` and `trace_mode=0`. Then acquire a set of 10 pulses.

You will notice that the pulse peaks all occur at the same time. The PMT-2000 issues its internal trigger when the signal reaches its peak. The weights summation starts at sample 0.

The distance between sample 0 and the pulse peak is 8 ADC samples on the PMT-2000 and 16 ADC samples on the SiPM-2000.

12.2 A theoretical example

Here we consider a theoretical example to explain how the PSD feature works. In the next section we show a practical case.

Consider a phoswich detector in which beta-particles create short pulses, and gamma-rays create long pulses. Here we just consider a much simplified version using two triangular pules shapes. For simplicity, we also assume that the trigger point is exactly at the start of the rising edge.

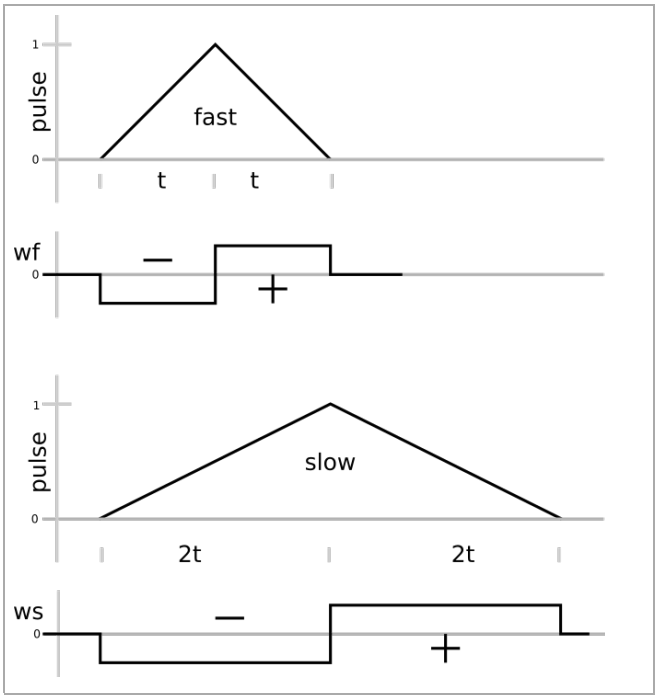


Fig. 18:Pulse shape discrimination using summation weights.

Consider applying the weights wf to the fast pulse. Because of the symmetry, the sum will be zero. But the same wf weights applied to the slow pulse will create P>0.

Similarly, applying the weights ws to the slow pulse will result in P=0. But the same ws weights applied to the fast pulse will create P<0.

Hence, using a weight function that is between wf and ws will map fast pulses to P<0 and slow pulse to P≥0. The firmware uses 0 as the decision point.

12.3 Weights format

Weights are signed 16-bit integers. When pulse shape discrimination is turned on, the weights are used as indicated in table 3. Since most of the control and GUI software treats the FPGA controls, and the summation weights as unsigned 16-bit, consult the table below to correctly encode the weights values.

PSD Weight	*uint16_t value*
-1	32768, 0x8000
-0.5	49152, 0xC000
-0.25	57344, 0xE000
0	0, 0x0
0.25	8192, 0x2000
0.5	16384, 0x4000
1-1/32768	32767, 0x7FFF

Table 3: Encoding PSD weights as 2's complement unsigned int 16. The largest possible positive value is represented by 0x7FFF=32767. The most negative possible value is represented by 32786=0x8000.

12.4 PSD controls and usage

Set **psd_on=1** to turn on pulse shape discrimination. Set **psd_sel=0** to see a split energy histogram with one type of radiation histogrammed in the lower half, and the other in the upper half. Set **psd_sel=1** to histogram the PSD sums used for the decision making, instead of histgramming event energies. To only histogram the energies of one type of pulses, set **psd_reject=1**. For unbiased operation, set **PUT=0**

With these standard settings, and after having loaded the weights, you will see events being classified as ***psd_left*** or ***psd_right***. In the PSD histogram, when ***psd_sel=1***, they will show up left and right of bin 1024. In the energy histogram, when ***psd_sel=0***, they will again be histogrammed left or right of bin 1024. When ***psd_sel=0, psd_reject=1*** the instrument will accumulate a 2K histogram of only ***psd_left*** events.

Fine tuning PSD: In some applications it will be useful to tune the pulse shape discrimination (PSD) by changing the decision threshold. For instance, a user might want to reduce a false positive rate for neutrons (caused by gamma-rays) at the expense of reducing the neutron detection efficieny.

To this end the ***PUT pileup*** parameter is repurposed when ***psd_on=1.*** Its 2 bytes ***put_low=PUT[0:7]*** and ***put_high=PUT[8:15]*** are considered to be signed bytes and are used as follows.

The PSD values, ***psd_value***, shown in the PSD

spectrum when ***psd_on=1, psd_select=1.*** are 11-bit 2's complement signed numbers. When plotted like a histogram, zero is mapped into bin 1024.

For each event, the FPGA computes the logical conditions ***psd_left=(psd_value/8)<psd_low***, ***psd_right=(psd_value/8)≥psd_high***

When ***psd_on=1, psd_select=0*** the MCA-2000 creates a split energy histogram consisting of two 1K histograms. The lower histogram corresponds to events classified as ***psd_left*** while the higher histogram records events for which the ***psd_right*** condition holds true.

Using ***PUT≠0*** one can tighten or loosen the PSD cuts.

The user should select ***psd_low≤psd_high*** in order to avoid that for any given event both conditions are true. The FPGA logic only histograms events for which exactly one of the two conditions is fulfilled. Events with ***psd_low=0 & psd_high=0*** or ***psd_low=1 & psd_high=1*** will not be histogrammed.

Using psd_reject: In a NaI-PVT phoswich detector, tpically only the gamma-ray energies (pulses from the NaI) need to be histrogrammed, as there is no useful information in the β-energy spectrum. In this case, set ***psd_reject=1.*** Only pulses for which the ***psd_left=1*** condition holds will be histogrammed – and with the full 2K histogram.

Swapping psd_low and psd_high: It depends on the weights chosen for the perceptron which type of pulses will be considered ***psd_left*** or ***psd_right***. For instance, in a NaI-PVT phoswich, plastic scintillator events might be classified as ***psd_left*** and be histogrammed on the low side of the split energy histogram. If the user wants the NaI pulses histogrammed in the low part of the spectrum instead, they need to change the sign on all weights. If the weights are loaded from a table of 16-bit unsigned integers, simply replace each weight ***w*** with ***w = 65536-w***.

12.5 PMT-2000 β/γ Example

Precision spectroscopy and ADC speed. To achieve precision spectroscopy, the PMT-2000 input amplifier slows down the scintillator pulses to achieve the best possible energy resoultion. However, the signal remains fast enough to support efficient puse shape discrimination. When using a β/γ phoswich made of a plastic scintillator (PVT) and a NaI-detector, the plastic scintillator pulse is still much faster than the NaI light pulse, which makes pulse shape discrimination an easy task in this case. Below we show the performance of such a PVT/NaI phoswich detector.

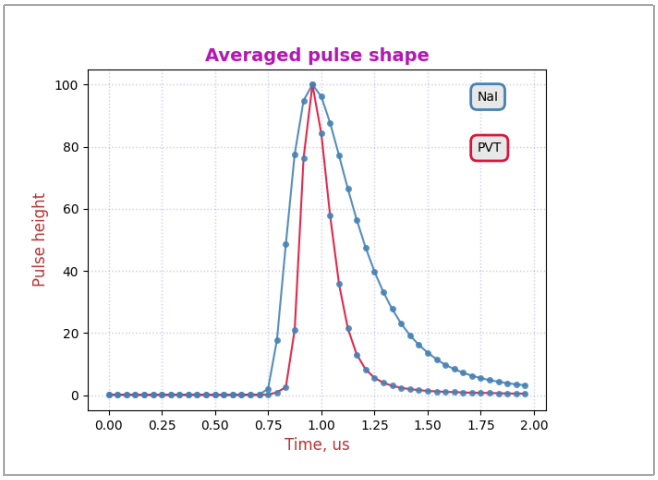


Fig. 19: Average of 1000 beta and gamma traces, acquired on a PMT-2000 at 24MHz digitization rate with a PVT/NaI phoswich detector.

Observe in fig. 19 how the β-pulses (red) are on average faster than the γ-pulses (blue). Their pulse width (fwhm) is about 250ns vs 500ns for the γ-pulses from the NaI. The difference is clearly visible. Consult the PMT-3000 user's manual to learn how to manually construct the pulse shape discrmination weights.

Consult the PMT-3000 user's manual to learn how to manually construct the pulse shape discrmination weights. Here we focus on using weights that have been computed using any of the many commercial tools to train a 1-layer neural network; ie a perceptron.

The MCA-2000 provides a tool for the developer to judge the performance of the pulse shape discrimination. In the arm_controls set **psd_on=1** and **psd_sel=1**. In the **Histogram** panel set the number of MCA bins to 2048, to see the entire spectrum. Keep in mind the PSD decision sums are centered around bin 1024 in this display.

With these settings you will see a 2K histogram of the PSD sums. Events to the left of bin 1024 will fall in one class, and events in bins ≥ 1024 will fall into the other. The shape of the PSD

histogram on either side of the divide is not of much interest. The only thing that counts is that the right and left side of the spectrum are separated by a clear gap.

Fig. 20 below shows an example. This one was acquired using a weak Cs-137 γ-source and a Tl-204 β-source. In the PSD histogram you can clearly see the two classes of events, well separated by a gap at the middle of the 2K histogram.

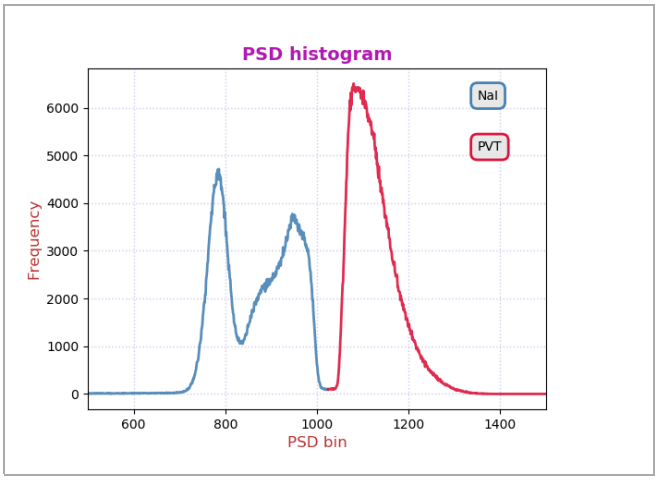


Fig. 20: Pulse shape discrimination (PSD) histogram for a combination of a Cs-137 and Tl-204 source using the PVT/NaI detector + PMT-2000. In this case, the NaI pulses are mapped onto the left side of the spectrum.

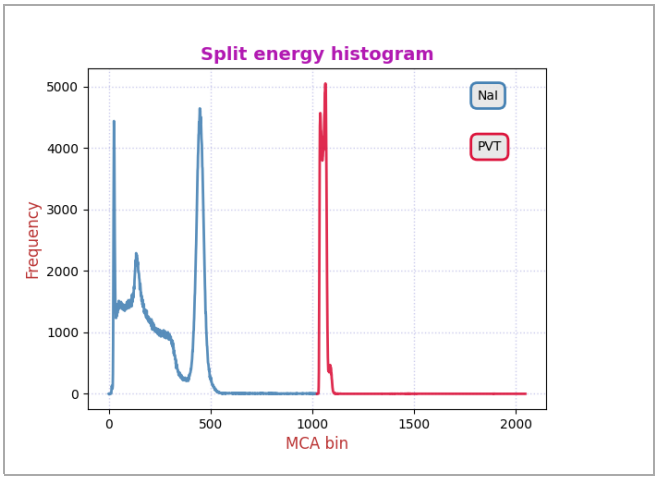


Fig. 21: Split energy histogram when ***psd_on=1, psd_sel=0***.

13. Advanced Capabilities

On-board flash memory: The MCA-2000 has an on-board 32MB flash memory that is used to store the FPGA configuration. It can also be used to store thousands of histograms, or even variants of the ARM processor code.

13.1 Code memory

FPGA firmware variants: In the MCA-2000 we have reserved space for eight FPGA configurations. With custom ARM software and FPGA firmware, an application could choose to boot the FPGA with different configurations.

13.2 Data memory

Record data without host computer: There is room for about 4000 2K histograms in the SPI flash. This can be used to record a large number of data in the device, and have the data still be present after power was turned off. Note that this type of recording does not require the presence of a host computer.

For example, an application could record a histogram once per minute for a record length of 66 hours, ie nearly 3 days.

The SPI flash has an endurance of 100,000 write cycles. In the above example, that corresponds to 750 years.

Collaboration: Contact Bridgeport Instruments to discuss how these capabilities can be put to use for your application.

14. Software upgrades

Upgrading and updating software: The MCA-2000 ARM code as well as the FPGA configuration can be upgraded or updated in the field via its USB connection. This feature allows to deploy code with bug fixes (updates) as well as the delivery of completely new custom software with application specific features and capabilities. All files are fully encrypted. They can be delivered to the developer, and even to the enduser customer without compromising software security. Once loaded into the MCA, the software can not be read back.

14.1 FPGA upgrade

Updating from the application: When the user receives a new FPGA configuration, as an encrypted file, they can load that file using the appropriate script in examples/mca2k /using_mds/arm_spi/ The example will have a name similar to manage_fpga.py Copy the fpga file into the data folder of the example and follow the instructions given in the python file.

14.2 ARM code upgrade

Using the bootloader: When the user receives a new ARM executable, as an encrypted file, they can load that file using the boot loader. This is a two-step process.

Step 1 Invalidate App: Using the examples in the examples/software_update folder, first invalidate the running application by executing the invalidate_app.py file. This sets a bit in non-volatile memory. Then power cycle the device or force a CPU restart (arm_restart.py)

Step 2 Uploading code: After the ARM CPU reboots, the MCA will be recognized by the computer's operating system as a USB device with Bridgeport's vendor ID, but now with a new product ID, namely PID=0x1000. Close the MCA Data Server.

On Windows: If you are performing the procedure for the first time on a Windows machine and are using libusb0.1, you must run wxMCA/zadig.exe and install libusb_win32 as the device driver. From then on Windows will remember the driver for this new device. On Linux no action is necessary.

Relaunch MDS: Launch the MCA Data Server again. It will now report a PID of 0x1000, and it will report the same serial number that the MCA already had when shipped. The serial number does not change in the process.

Uploading code: Copy the encrypted arm executable to the software_update/data folder. Edit the file name in the update_arm.py file and execute that file. A typical code is uploaded in 10 seconds, but the maximum time for the biggest possible code size is 80 seconds.

Relaunch MDS: After the code has been uploaded, the computer will register a disconnect and reconnect series of events on the USB bus. This happens because the updated application relaunches the USB interface. Exit the MDS and launch the MCA Data Server again. It will now report a PID of the MCA, and the same serial number as before.

Step 3 Validate App: Test the new application to make sure it works, then validate the application. Execute the validate_app.py file. It issues a command to clear the invalid/valid bit. With that done, the internal bootloader will automatically boot into the application after the next power cycle.

Risks: There is no risk to permanently disable the device. When BPI distributes an update, BPI will also distribute a safe fall back code. If for some reason the processor fails to boot into the application after the code update, simply power cycle the device. Since its program memory has ben declared invalid, the boot loader will simply wait for the next attempt to upload code via USB.

15. Analog

15.1 PMT-2000

The PMT-2000 combines a PMT operating voltage supply and an MCA data acquisition board with a structure as shown in the figure below.

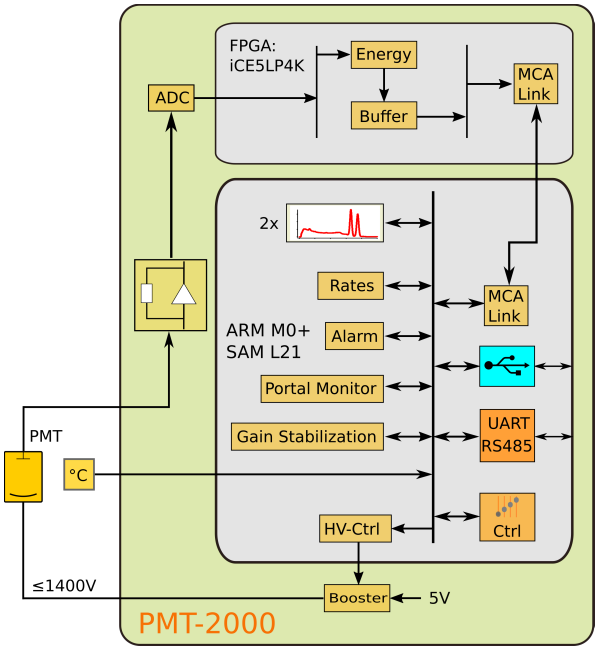


Fig. 24: The components of the PMT-2000.

The analog input of the PMT-2000 uses an I-to-V converter with four programmable gain resistors as shown in the figure below. A gain_select of 0, 1, 2, 3, and 4 creates a transimpedance of 100Ω, 430Ω, 1100Ω, 3400Ω, and 10100Ω, respectively.

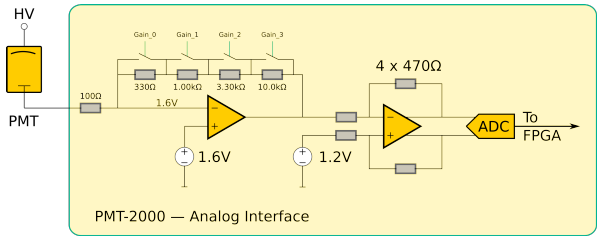


Fig. 25: The input amplifier of the PMT-2000.

15.2 SiPM-2000

The SiPM-2000 combines a very low-noise SiPM operating voltage supply and an MCA data acquisition board with a structure as shown in the figure below.

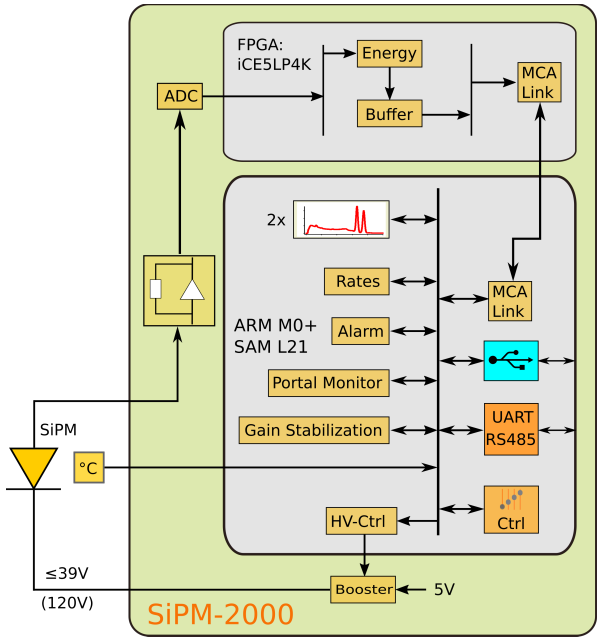


Fig. 26: The components of the SiPM-2000.

The analog input of the SiPM-2000 uses an I-to-V converter with four programmable gain resistors as shown in the figure below. A gain_select of 0, 1, 2, 3, and 4 creates a transimpedance of 5Ω, 20Ω, 55Ω, 155Ω, and 505Ω, respectively.

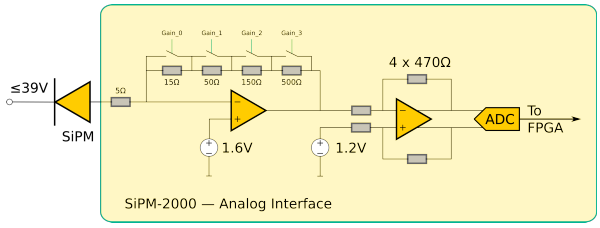


Fig. 27: The input amplifier of the SiPM-2000.

16. Mechanical

16.1 8-Pin Connector

The PMT-2000 uses a Bulgin PX0447 mini-B USB connector for power and USB communication. Mating cables are the Bulgin PX0441 (USB-A) and PX0442 (USB mini-A) series cables. They come in lengths from 2m to 4.5m.

The PMT-2000 uses a Switchcraft EN3P8MPX 8-pin connector for GPIO and serial UART communication. The mating connector is part of the EN3C8 series.

Fig. 28: PMT-2000 diemsions and EN3P8 (8-pin GPIO) connector.

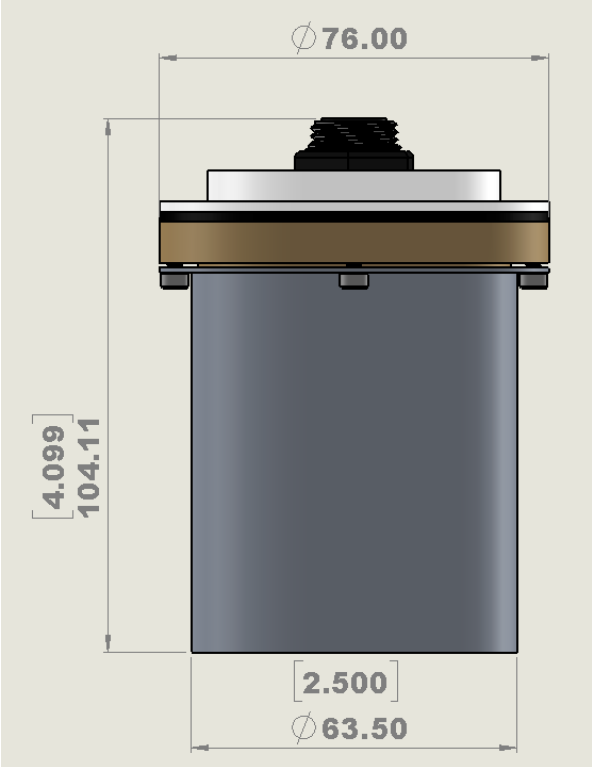


Fig. 29: SiPM-2000 diemsions. The SiPM-2000 board exposes the same conectors as the PMT-2000.

Fig. 30: Pin numbers on the EN3P8 (8-pin GPIO) connector. The connectors and pinouts are the same for the PMT-2000 and the SiPM-2000.

Connector J2, EN3P8MXPKG		
#	***Name***	***Description***
1	SWD_IO	Software Debug Data
2	SWD_RST	Software Debug Reset
3	Pin3	N_Out50# or RS485_A
4	Pin4	GPIO_A3 or N_Out33 or PA2 (SiPM)
5	Pin5	N_Out50 or RS485_B
6	VD50	+ 5V power input (when not using USB)
7	GND	Ground
8	SWD_CLK	Software Debug Clock

Front side connectors pin out

17. Product and Part Numbers

17.1 Product numbers

The PMT-2000 is part of a product series that all contain an ARM M0+ 32-bit processor for communication and slow control, such as gain stabilization. In the -1000 series the MCA is implemented by software in the ARM processor. In the -2000 series, the MCA is implemented in the ARM processor, but the real-time signal processing is performed in the FPGA, which achieves higher throughput. In the -3000 series the MCA is implemented independently within an FPGA for very high-speed operation. The -3000 series also uses a waveform-digitizing ADC and offers detailed pulse capture and real time pulse shape discrimination. The fixed, lower, discretization speed of the -2000 series MCA still allows for excellent pulse shape discrimination in, for example, neutron detectors.

The devices are available for vacuum photomultiplier tubes (PMT) and Si-photomultipliers (SiPM). In both cases, the device generates the operating voltage for the photo-sensor from the incoming 5V.

P/N	***Sensor***	***FPGA***
PMT-2000	PMT	Yes
SIPM-2000	SiPM	Yes

Table : Part numbers.

17.2 USB-ID

On the USB bus devices are recognized by their Vendor ID (VID), Product ID (PID) and Serial Number (SN). The vendor ID for Bridgeport Instruments is 0x1FA4. The Product ID's are shown in the table below. Within a product the serial number is fixed, unless BPI makes a custom device that requires a non-standard driver. Note that simple extensions, such as adding a variable to the controls, does not require a new driver.

The BPI software recognizes individual devices by the unique serial number burnt into each ARM processor. The device reports that when the host reads ***arm_version***. The serial number communicated in response to USB setup commands is fixed for each part, to avoid that the host keeps adding every new device to an ever longer list of devices requiring a designated USB driver.

P/N	*PID*	*SN*
PMT-2000	0x0102	pmt2k0001
SIPM-2000	0x0202	sipm2k0001

Table : Product ID and USB bus serial numbers.

17.3 Device serial numbers

Each ARM processor has an immutable 128-bit unique serial number, which can be printed as a 32-character hexadecimal string. The MCA Data Server always uses the complete 32-character string to identify the device.

The MCA printed circuit board carries a unique printed identifier. It carries a 4-digit part number and revision; eg PN1380/A for the PMT-2000. Further it carries a running serial number of the form YYMMnnnn, which encodes the production date as year since 2000, followed by the month of the year. The remainder is the running PCB number of this production run.

The ARM processor reports a hardware version of the form: PPPPYYMMV, where PPPP is the 4-digit part number, YYMM are the production date, and V indicates a production variant.