MUSICR1: 8 channel Multiple Use IC for SiPM anode readout

Software Manual

Issue:	Stable
Revision:	2.6
Reference:	MUSICR1
Created:	29th June 2016
Last modified:	8th May 2019
Prepared By:	SiUB

Table of contents

1 Introduction

1.1 Description

This is a software for controlling MUSIC boards from the command line. For this purpose, a list of commands allows the user to write from and read to the MUSIC board.

1.2 Computer requirements

There are different compilations of the software for different operating system versions, so you want to check out that is the one suitable for you. The **"music"** is available for linux users and also for windows 7. For Linux users, we strongly recommend the usage of version 16 or higher of Ubuntu.

The MUSIC board employs an FTDI chip to establish the communication with the computer. In linux, in order to be able to connect to the music board it's necessary to configure the system to grant access to the current user to the board through the usb interface. This configuration must be done as root in the machine. In order to do so create a file

```
/etc/udev/rules.d/99-ftdi.rules
```

with the following line:

```
SUBSYSTEMS=="usb", ATTRS{idProduct}=="6010", ATTRS{idVendor}=="0403",
RUN+="/bin/sh -c 'echo -n  $kernel:1.0 >>
/sys/bus/usb/drivers/ftdi_sio/unbind'",  RUN+="/bin/sh -c 'echo -n
$kernel:1.1 >> /sys/bus/usb/drivers/ftdi_sio/unbind'", GROUP="music"
```

Keep in mind that the contents must be a single line. Once the file exists, create a group called music by running:

```
groupadd music
```

And add yourself to the group

```
useradd -G music yourusername
```

To verify that your user is in the group you can run:

```
id yourusername
```

and group music should appear in the list of groups for that user. Once everything is set up, reboot the machine to make sure the new configuration is properly loaded.

NOTE: The music board might end in unknown state whenever an abrupt exit of the software occurs (e.g., killing the process or CTRL+C command). In this scenario, reconfigure the music board by using properly any of the commands available in the software.

1.3 Summary of all available comments

A list of all included commands in the MUSIC ASIC is here detailed.

Commands	Action
version	Show current software version.
list_boards	Show connected boards.
upload_firmware	Upload a new firmware.
version_firmware	Show firmware version.
config	Configure MUSIC functionalities through CLI.
config_file	Configure MUSIC with a calibration file (enables advanced mode).
show_config	Shows current MUSIC chip configuration.
threshold_scan	Performs a threshold scan.
photon_count	Performs a Photon Count Rate (PCR) scan (or Dark Count Rate).
spi_fer	Performs MUSIC SPI Frame Error Rate (FER) test.

2 Measurement procedures

2.1 Initial considerations

The MUSIC software enables the possibility to perform several analyses on the SIPM anode readout. Next it is detailed the procedure and commands to do the different tests. Observe that the default configuration of the ASIC enables the single ended analog output and the summation of all channels.

Connect the power supply and the usb connection. Note that the board can be configured using only the power supply coming from the USB connection, although some USB cables/computer might not provide enough current to the board to properly feed the ASIC. Hence, use always a power supply source besides of the usb cable.

Lastly, an example of the output of any of the commands used hereafter can be seen in section 3 and 4.

2.2 Analog Summation Readout.

The following example shows how to perform summation analog readout with a specific configuration and how to save the current configuration of the ASIC to a file.

1. Connect the power supply and the usb connection.
2. Load the firmware.
    a. Switch U20 set to ON. The firmware is loaded directly from the FLASH.
    b. Switch U20 set to OFF. Use the following command to upload a new firmware.

> **music *upload_firmware*** -f music_firmware.rbf

3. Configure the required parameters to operate the ASIC. An example with a commonly used configuration of the MUSIC ASIC is here depicted.
    a. Set the Calibration file (Mandatory): MUSIC_M01.vdc

b. Configure summation mode for channels: 0 3 6.
c. Enable the required channels: For example: 0 3 6
d. Enable the pole zero cancellation with R = 3, C=5 and low attenuation ON.
e. Enable high transimpedance.

music *config* -f MUSIC_M01.vdc -u 0 3 6 -e 0 3 6 -p 3 5 -a -t

4. Observe in the oscilloscope the summation readout using pin P11 or the SMAs Co3-Co7.
5. Repeat step 3 to change the current configuration of the ASIC as many times as needed. Note that he enable input might not be the same as the summation outputs. For instance, we might enable all inputs in the 7 SiPM array, but we want only to obtain the summation in the center of the array (channel 3, i.e. SiPM Si4 from board), then the command to configure the ASIC will be:

music *config* -f MUSIC_M01.vdc -u 3 -e 0 1 2 3 4 5 6 -p 3 5 -t

2.3 Analog Single Ended Readout.

The following example shows how to perform single ended analog readout with a specific configuration.

1. Connect the power supply and the usb connection.
2. Load the firmware.
   a. Switch U20 set to ON. The firmware is loaded directly from the FLASH.
   b. Switch U20 set to OFF. Use the following command to upload a new firmware.

music *upload_firmware* -f music_firmware.rbf

3. Configure the required parameters to operate the ASIC. An example with a commonly used configuration of the MUSIC ASIC is here depicted.
   a. Set the Calibration file (Mandatory): MUSIC_M01.vdc
   b. Configure analog mode (default).
   c. Enable the required channels: For example: 2 4 5
   d. Enable the pole zero cancellation with R = 4, C=18 and low attenuation ON.
   e. Enable high transimpedance.

music *config* -f MUSIC_M01.vdc -e 2 4 5 -p 4 18 -a -t

4. Observe in the oscilloscope the analog output readout using pin P13 or the SMA external board connected to P13.
5. Repeat step 3 to change the current configuration of the ASIC as many times as needed.

2.4 Digital Single Ended Readout

The following example shows how to perform single ended digital readout with a specific configuration and how to save the current configuration of the ASIC to a file.

1. Connect the power supply and the usb connection.
2. Load the firmware.

a. Switch U20 set to ON. The firmware is loaded directly from the FLASH.
b. Switch U20 set to OFF. Use the following command to upload a new firmware.

music *upload_firmware* -f music_firmware.rbf

3. Configure the required parameters to operate the ASIC. An example with a commonly used configuration of the MUSIC ASIC is here depicted.
   a. Set the Calibration file (Mandatory): MUSIC_M01.vdc
   b. Enable all channels (In order to properly obtain the thresholds for all channels).
   c. Disable Pole Zero Cancellation (default).
   d. Set transimpedance gain to low (default).
   e. Configure the digital mode.

music *config* -f MUSIC_M01.vdc -e 0 1 2 3 4 5 6 7 -d

4. Perform a threshold scan in order to obtain the required thresholds, so the circuit is configured near the transition. Observe that the thresholds depend on the desired configuration of the ASIC and whether the SIPM is switched ON or not.

music *thresholdScan* -r thScan_results.txt

This command will give an output similar to: -b 5 -v 215 214 213 218 211 214 216 215
Where -b 5 says that the optimum VBG must be set to 5, and the -v followed by those numbers tells a recommended good threshold value to see the digital signal.

5. Use the thresholds and the VBG voltage obtained to reconfigure the ASIC.
   a. Enable the desired channels: For example, 1, 6.

music *config* -f MUSIC_M01.vdc -e 1 6 -d -b 5 -v 215 214 213 218 211 214 216 215

6. Observe in the oscilloscope the digital output readout using pin P13 or the SMA external board connected to P13.
7. Repeat step 3 to change the current configuration of the ASIC as many times as needed in order to configure the thresholds at the desired values.

2.5 Single Photon Time Resolution. Dark Count Rate

The MUSIC software includes a photon counting or a dark count rate (DCR) implemented in the FPGA, depending if the SiPM is under light conditions or at the dark. The DCR is required to properly configure the threshold of the digital signal to obtain the Single Photon Time Resolution (SPTR) or Jitter.

1. Connect the power supply and the usb connection.
2. Load the firmware.
   a. Switch U20 set to ON. The firmware is loaded directly from the FLASH.
   b. Switch U20 set to OFF. Use the following command to upload a new firmware.

music *upload_firmware* -f music_firmware.rbf

3. Configure the digital signal as output. The MUSIC board includes an SMA output connection (U4) which shows one selected digital output hannel. This output has the advantage that it does not include uncoupling capacitors and thus the DC level of the digital signal can be easily seen (it outputs the same signal as Pin P13).
    a. Enable channel 4
    b. Configure channel 4 as output via the SMA U4.
    c. Configure the digital mode.

> **music *config*** -f MUSIC_M01.vdc -e 4 -m 4 -d

4. Prior to perform the DCR, the transition of the discriminators must be found. Hence, configure the ASIC with the required specifications of the signal
    a. Set the Calibration file (Mandatory): MUSIC_M01.vdc
    b. Configure the digital mode.
    c. Enable all channels.
    d. Disable Pole Zero Cancellation.
    e. Set transimpedance gain to high.

> **music *config*** -f MUSIC_M01.vdc -e 0 1 2 3 4 5 6 7 -d -t

Observe that step 4 and 5 can be done at the same time.

> **music *config*** -f MUSIC_M01.vdc -e 0 1 2 3 4 5 6 7 -d -t -m 4

5. Perform a threshold scan in order to obtain the required thresholds, so the circuit is configured near the transition. Observe that the thresholds depend on the desired configuration of the ASIC and whether the SIPM is switched ON or not.

> **music *threshold_scan*** -r thScan_results.txt

This command will give an output similar to: -b 5 -v 215 214 213 218 211 214 216 215

6. Use the thresholds and the VBG voltage obtained to reconfigure the ASIC.
    a. Enable the desired channels: For example 1, 6.

> **music *config*** -f MUSIC_M01.vdc -e 1 6 -d -b 5 -v 215 214 213 218 211 214 216 215

7. Observe that the thresholds obtained with the prior command might be too close to the noise level depending on the SIPM signal and thus an excessive number of transitions will be observed in the oscilloscope. If this happens, decrease 1 or 2 LSBs the thresholds previously obtained: For instance, if we decrease 1 LSB the threshold of channel 4, the command will be:

> **music *config*** -f MUSIC_M01.vdc -e 1 6 -d -b 5 -v 215 214 213 218 <u>210</u> 214 216 215

8. Now that we have the channel with the lowest time threshold, now we are going to perform the DCR stairplot. The DCR is obtained using the photon counting functionality of the music software.

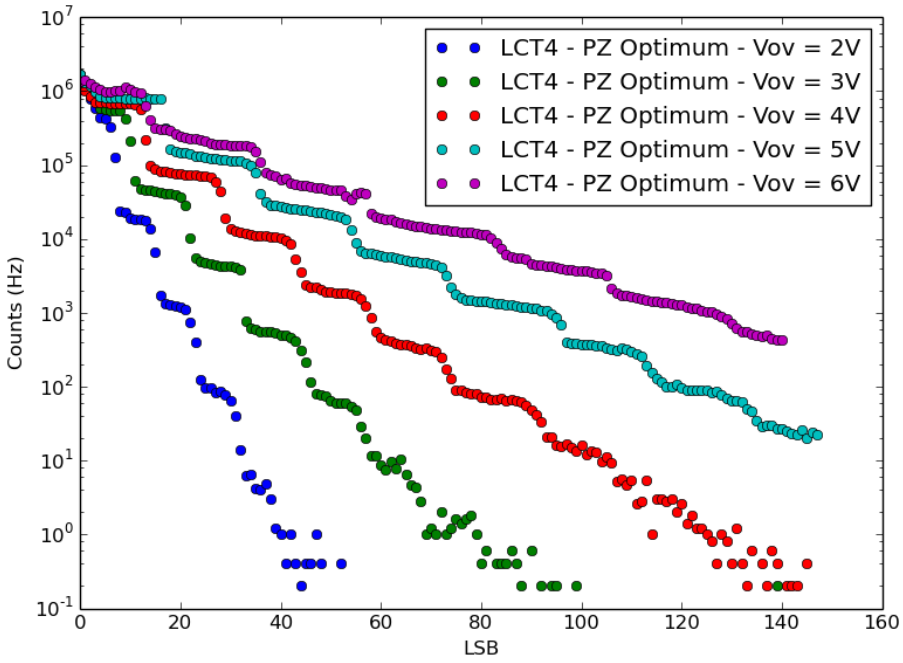
a. The photon counting counts the number of pulses from the time signal in a certain time window. That number is the Counts (Hz) for a certain threshold (our **210** correspond to the **0** from the plot).

b. The threshold is decreased in steps of 1 LSB (**210,209,208,207…, that corresponds to 0,1,2,3… on the plot**) and we count the number of pulses (photons)

music *photon_count* -r phCounts.txt -v 50 210

c. The previous function will provide a file containing the amount of photons counted in each channel for each threshold in the range previously defined.

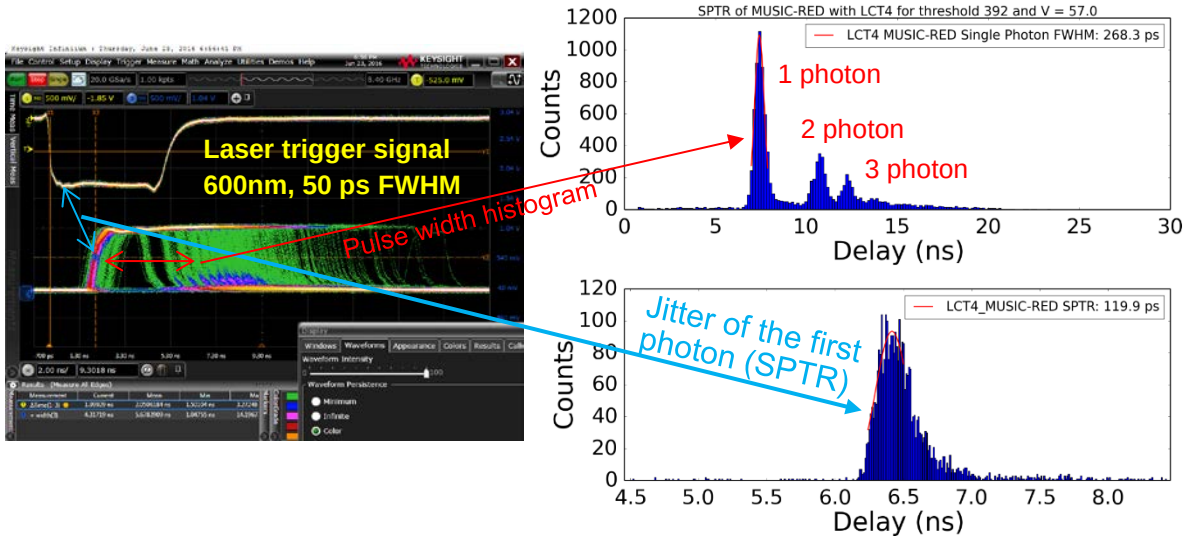
d. If we plot the amount of photons per threshold for the channel 4, we will be able to obtain something similar the plot below. In this example, the DCR has been performed for different bias voltage, i.e., for a different level of dark counts.



9. Now that we have the DCR plot and we can distinguish the plateau for the 1st, 2nd, 3rd photoelectron. So, for the SPTR we must put our time threshold on the middle of the first plateau, from the previous plot we can see that for Vop at 6V, the time threshold should be 5 or 6, so we can configure 204 or 205.

music *config* -f MUSIC_M01.vdc -e 1 6 -d -b 5 -v 215 214 213 218 205 214 216 215

10. Once we have the channel configured, we just need to perform the SPTR. We calculate the delay from a certain trigger and our time signal (referred as jitter), as well as the width of the time signal. This procedure should be done acquiring N samples of this delay from the signals visualize in the oscilloscope and thus the acquisition method directly depends on the oscilloscope used for each user.



2.6 Photon Counting

The MUSIC software includes a Photon counter implemented in the FPGA.

1. Repeat steps 1-9 to configure the digital channels as explained in the previous section with the difference that here the SiPM has to be switched on before configuring the ASIC. Hence, set the bias voltage above the breakdown of the SiPM (step 3 in the previous section).
2. Repeat step 10 from the previous section with the difference that here we can select the threshold that represents the 1st, 2nd, 3rd, or any other plateau depending on the photons that we want to count. If we want to count single photons and above, we will select the 1st plateau, if want to count at least two photons and above, we will select the 2ns plateau, if want to count at least three photons and above, we will select the 2ns plateau and so on.

2.7 Analog Readout with channel gain equalization.

The mismatch variations cause that each channel might have a slightly different gain. The MUSIC ASIC can correct these differences by modifying the input voltage (Voffset) at the anode of the SIPM. Note that equalization should be done for both the summation and the single-ended analog readout. This section describes the procedure to equalize all channels.

1. Connect the power supply and the usb connection.
2. Load the firmware.
    a. Switch U20 set to ON. The firmware is loaded directly from the FLASH.
    b. Switch U20 set to OFF. Use the following command to upload a new firmware.

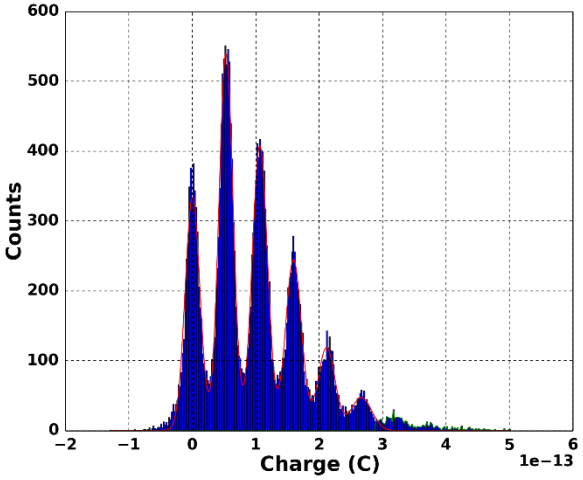
music *upload_firmware* -f music_firmware.rbf

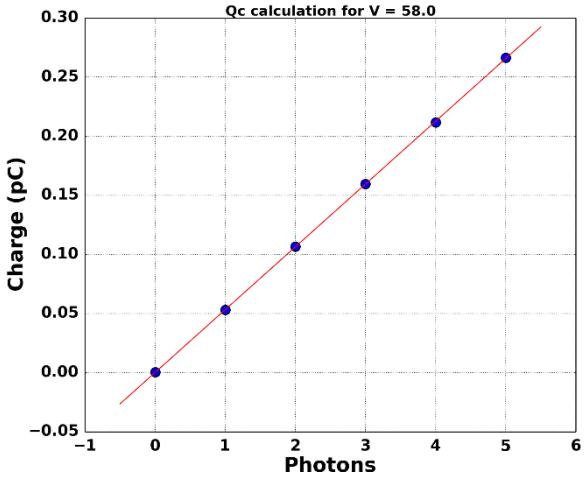
3. Configure the required parameters to operate the ASIC. Enable all channels to perform the equalization
    a. Set the Calibration file (Mandatory): MUSIC_M01.vdc
    b. Configure summation mode for channels: 0 1 2 3 4 5 6.
    c. Enable the required channels: For example: 0 1 2 3 4 5 6.

d. Enable the pole zero cancellation with R = 4, C=18 and low attenuation ON.
e. Enable high trans-impedance.

music *config* -f MUSIC_M01.vdc -u 0 3 6 -e 0 3 6 -p 4 18 -a -t

4. Observe in the oscilloscope the summation readout using pin P11 or the SMAs Co3-Co7.
5. Compute the gain of each individual channel. You can do the following procedure for each channel to obtain the charge gain:
   a. Modify the optical system, (attenuators and laser signal) in order to visualize only the first photons.
   b. Acquire several pulse signals (at least 1000 samples) from the Oscilloscope.
   c. Compute the charge of each signal and obtain the charge distribution of the samples obtained. Note that output of the summation is an output voltage, so the charge can be obtained by integrating the voltage output and dividing it by the gain of the circuit (referred to the Data Sheet document to see the different gain options).
   d. Fitting the plot "charge vs amount of counts", you can obtain the charge of each photoelectron, in other words, the charge gain.





6. Once we have obtained the charge gain of each channel, we select one channel (for instance, the one with the gain closer to the average) and we modify the voffset of all other channels in order to increase or decrease the charge gain and equalize it with respect to the selected reference channel. As an example, we fied as reference channel 3 (with an Voffset configured with the default value of 255) and then we modify the other channels to equalize the charge gain.

music *config* -f MUSIC_M01.vdc -u 0 3 6 -e 0 3 6 -p 4 18 -a -t -o 248 257 247 255 246 243 258

7. Repeat steps 5-6 until we properly equalize the different channels.

2.8 Save/Load configuration to/from a file

1. Save the current configuration of the ASIC for future uses.

music show_config -o MUSIC_USER_CONF.calib

2. Load a previous configuration of the ASIC.

```
music config_file -f MUSIC_USER_CONF.calib
```

2.9 Configuration of the ASIC using the ADVANCE mode.

The simplified command line interface permits to easily configure the MUSIC ASIC, as previously detailed. An advanced mode is also available to configure any parameter of the ASIC design (the description of all parameters can be found in the datasheet of the ASIC).

1. The advanced mode is activated loading a calibration file (MUSIC_USER_CONF.calib) with the auto reload option enabled.

```
music config_file -f MUSIC_USER_CONF.calib -a
```

2. Open in a text editor the configuration file MUSIC_USER_CONF.calib.
3. Apply any change in the configuration file and save it. Once save it, the software automatically send to the ASIC the new configuration.
4. The advanced mode can be exit just by ending of the interactive session in the command line, i.e., killing the process (CTRL + C).

2.10 Output DC voltage.

The DC voltage at each of the analog outputs is already calibrated for any combination of the pole zero cancellation or gain. However, if the user aims to change the DC values at the output, they can be modified using the advanced mode detailed in section 2.9. The summation can be changed by modifying the VDC_LG and the VDC_HG and the single ended channels can be modified by changing the VDC_CH. These parameters can be modified from 0 to 255. In order to see the DC level at each output, the easiest method is the employment of a multi-meter. The DC output voltage of the summation channels can be seen in pin P11 and for the case of the single-ended analog channels they can be measured using pin P13.

3 Firmware and board Commands

The following sections describe the available commands to configure the board and the firmware.

3.1 Version

The version command displays the current version of the software. The version corresponds to the last git commit ID of the source code.

Command:

version

Input parameters:

<none>

Generic options:

Help (-h, --help)
Shows the help message.

Board (-b,--board) $argument
Indicates in which MUSIC board serial number the configuration will take place. This parameter is mandatory when two or more MUSIC boards are connected to the same USB host.
Log (-l, --loglevel) $argument
Activates log messages to provide further information about the software commands. The different log levels are: trace, debug, verbose ,info, warning, error, fatal, disable)
Header command (-F, --logformat) $argument
Use character '-' to disable any header printed in the command line windows.
Load command from file (-C, --args_file) $argument
File containing the arguments to be parsed to this command. Same format as the command line flags.

Output:

This command prints on the console the version identifier so we can track out the associated source code.

Example:

```
$ music version -F -
Current version is 493914c6.
```

3.2 List Boards

List boards command displays the list of powered MUSIC boards currently connected to the computer. Note that the communication hardware on the board includes 2 ports per device. For all subsequent commands the last letter can be omitted when referring to the board. See examples below.

Command:

list_boards

Input parameters:

<none>

Generic options:

Help (-h, --help)
Shows the help message.
Board (-b,--board) $argument
Indicates in which MUSIC board serial number the configuration will take place. This parameter is mandatory when two or more MUSIC boards are connected to the same USB host.
Log (-l, --loglevel) $argument
Activates log messages to provide further information about the software commands. The different log levels are: trace, debug, verbose ,info, warning, error, fatal, disable)
Header command (-F, --logformat) $argument
Use character '-' to disable any header printed in the command line windows.
Load command from file (-C, --args_file) $argument
File containing the arguments to be parsed to this command. Same format as the command line flags.

Output:

This command prints on the console a list of all connected MUSIC boards.

Example:

```
$ music list_boards -F -
Board MUSIC0001 : MUSIC Test Board (B-MUS-T-000-01)
   Channel: MUSIC0001A
   Channel: MUSIC0001B
```

3.3 Upload Firmware

This command allows to upload a new firmware to the MUSIC board. This will override the current firmware running on the FPGA but not the one stored in the on-board flash memory. Therefore, this command must be issued before performing any test if the firmware to be used is newer than the one stored into the on-board flash memory.

Command:

upload_firmware

Input parameters:

Firmware (-f,--firmware)
Indicates the path where the binary (.rbf) is located.

Generic options:

Help (-h, --help)
Shows the help message.
Board (-b,--board) $argument
Indicates in which MUSIC board serial number the configuration will take place. This parameter is mandatory when two or more MUSIC boards are connected to the same USB host.
Log (-l, --loglevel) $argument

Activates log messages to provide further information about the software commands. The different log levels are: trace, debug, verbose ,info, warning, error, fatal, disable)
Header command (-F, --logformat) $argument
Use character '-' to disable any header printed in the command line windows.
Load command from file (-C, --args_file) $argument
File containing the arguments to be parsed to this command. Same format as the command line flags.

Output:

This command prints on the console the upload status of the new firmware. If no errors occur, then it shows the firmware identifier, build number, build date and git version.

Example:

```
> music upload_firmware -f music_fw.rbf -F -
Trying to upload music_fw/output/music_fw.rbf to board MUSIC0000
Firmware file music_fw/output/music_fw.rbf is 123562 bytes long.
Uploaded firmare at 686891 bytes/sec
Firmware: MUSIC MABI (SC+FC) Build: v1.0@2016/10/21 17:33
Version: 82494081
```

3.4 Firmware Version

The version command displays the current version of the firmware, build number and build date. The version corresponds to the last git commit ID of the source code.

Command:

version_firmware

Input parameters:
<none>

Generic options:

Help (-h, --help)
Shows the help message.
Board (-b,--board) $argument
Indicates in which MUSIC board serial number the configuration will take place. This parameter is mandatory when two or more MUSIC boards are connected to the same USB host.
Log (-l, --loglevel) $argument
Activates log messages to provide further information about the software commands. The different log levels are: trace, debug, verbose ,info, warning, error, fatal, disable)
Header command (-F, --logformat) $argument
Use character '-' to disable any header printed in the command line windows.
Load command from file (-C, --args_file) $argument
File containing the arguments to be parsed to this command. Same format as the command line flags.

Output:

This command shows the firmware identifier, build number, build date and git version.

Example:

```
$ music_linx64 version_firmware -F -
Firmware:  MUSIC  MABI  (SC+FC)  Build:  v1.0@2016/10/21  17:33  Version:
```

4 Functional Commands

The following commands are used to configure the different operation modes of the MUSIC ASIC.

4.1 Config

This command allows to configure MUSIC chip electronics. It is important to highlight that this command will set several parameters to the default value if they are not specified, but this default configuration does not necessary need to be the same as the configuration of the ASIC after a reset or power on of the board.

Command:

config

Input parameters:

VDC Calibration file (-f, --vdcfile) $argument

Specify the Music Single-Ended VDC calibration file name (*.vdc). Observe that each PCB board comes with an already calibrated .vdc file. If needed, a user can modify this file using the advanced mode detailed in section 2.9.

Optional input parameters:

Readout Input Channels (-e, --chan_enable) $list_arguments

The list of MUSIC readout channels enabled (0-7).

Summation mode (-u, --sumchans) $list_arguments

The list of MUSIC channels to be summed (0-7). It this option is not specified, all summation channels are switched off.

Single Ended operational mode (-d, --digital)

When specified, MUSIC Single-Ended outputs (VoSE[7:0]) are configured as digital, otherwise is configured as analog outputs(DEFAULT).

Selectable SMA digital output (-m, --measchan) &argument

Select one MUSIC SE output channel (0-7) to be output in digital mode via SMA U4. Note that this output does not have uncoupling capacitances in its path.

Pole Zero (-p, --polezero) $R_value $Cap_value

When specified, MUSIC Pole Zero cancellation is enabled. This command requires two values: first, the resistance ladder (0-7) and second the capacitance ladder (0-31).

Low Attenuation Ladder (-a, --lowlad)

When specified, Pole-Zero attenuation is reduced (true). By default, low attenuation is set to false.

High gain transimpedance mode (-t, --hightrans)

When specified, single-ended and differential channels are configured in high trans-impedance mode. By default, the low trans-impedance mode is configured.

Input voltage configuration (-o, --voffchannels) $list_arguments

Select per-channel the anode input voltage VOFFSET (0-511). 8 arguments (one for each channel) are required to use this configuration option.

Voltage threshold at comparator (-v, --vthchannels) $list_arguments

Select per-channel the fine voltage threshold (Vth). 8 arguments (one for each channel) are required to use this configuration option.

Reference voltage in comparator threshold (-g, --vbg) &argument

Selects the reference voltage (VBG) at the comparator (0-7), permits to extende the range of threshold at the comparator (Consult the datasheet for further information).

SPI frequency (-s, --freqspi) $argument
Configures the frequency of the SPI master in MHz. Choose a number between 0.12 and 30.

Generic options:

Help (-h, --help)
Shows the help message.
Board (-b,--board) $argument
Indicates in which MUSIC board serial number the configuration will take place. This parameter is mandatory when two or more MUSIC boards are connected to the same USB host.
Log (-l, --loglevel) $argument
Activates log messages to provide further information about the software commands. The different log levels are: trace, debug, verbose ,info, warning, error, fatal, disable)
Header command (-F, --logformat) $argument
Use character '-' to disable any header printed in the command line windows.
Load command from file (-C, --args_file) $argument
File containing the arguments to be parsed to this command. Same format as the command line flags.

Output:

This command shows a message informing that MUSIC was configured properly.
If the vdc configuration file does not exist, it will show a message telling that it was not able to load a .vdc file. In case that there are communication problems between the MUSIC FPGA and the integrated circuit error messages will appear. In this case it is recommended to run an SPI Frame Error Rate (FER) test.

Example:

```
$ music config -f ./calib/MUSIC_M01_VDC_CF.vdc -e 0 1 2 3 -d -m 3 -u 0 2
-p 3 5 -a -t -o 240 243 244 234 243 240 247 233 -g 6 -v 134 132 131 128
139 125 137 123 -F -
```

```
GLOBAL CONFIGURATION:
  - Transimpedance = high
  - Pole/Zero = enabled
    - LOW_AT_LAD = enabled
    - RLAD_PZ    = 3
    - CAP_PZ     = 5
  - Bandgap Voltage (VBG) = 6
  - Readout input channel enable:
    - Enable [0] = on
    - Enable [1] = on
    - Enable [2] = on
    - Enable [3] = on
    - Enable [4] = off
    - Enable [5] = off
    - Enable [6] = off
    - Enable [7] = off
  - Voffset values:
    - Voff [0] = 240
    - Voff [1] = 243
    - Voff [2] = 244
    - Voff [3] = 234
    - Voff [4] = 243
    - Voff [5] = 240
    - Voff [6] = 247
    - Voff [7] = 233
SUM CONFIGURATION:
  - ch[0] = enabled
  - ch[1] = disabled
  - ch[2] = enabled
  - ch[3] = disabled
  - ch[4] = disabled
  - ch[5] = disabled
  - ch[6] = disabled
  - ch[7] = disabled

SINGLE ENDED CHANNELS' CONFIGURATION:
  - Output type   = digital
  - Selected digital SMA Output: Ch 3
  - Threshold values:
    - Vth [0] = 134
    - Vth [1] = 132
    - Vth [2] = 131
    - Vth [3] = 128
    - Vth [4] = 139
    - Vth [5] = 125
    - Vth [6] = 137
    - Vth [7] = 123
```

4.2 Config_file and interactive mode

This command allows to configure the MUSIC chip electronics from a calibration file. An example file can be found in Appendix: MUSIC IC Calibration File example. If the file exists but it is not formatted properly (a variable name is missing or miswritten) a warning message will appear.

This command also allows to configure MUSIC chip electronics interactively from a calibration file (). This allows user to tune chip parameter in real time. User can modify the specified calibration time and each time that the file is saved (Ctrl+S in most of the text/code editors) chip parameters are updated. An example of a MUSIC Calibration File can be found in Appendix: MUSIC IC Calibration File example.

Command:

config_file

Input parameters:

Calibration file (-f, --file) $argument

Indicates the path where the calibration file (.calib) is located. Appendix A shows an example of this calibration file. This parameter is MANDATORY.

Optional input parameters:

Interactive mode (-a, --auto_reload)
When the interactive or advanced mode is enabled, open the calibration file in a text editor. All changes saved in the configuration file will be automatically send to the ASIC. The configuration of the ASIC is updated every time the file is saved and a message will appear to inform that the Calibration File has been modified. In case of SPI communication problems, error messages wil appear.
SPI frequency (-s, --freqspi) $argument
Configures the frequency of the SPI master in MHz. Choose a number between 0.12 and 30.

Generic options:

Help (-h, --help)
Shows the help message.
Board (-b,--board) $argument
Indicates in which MUSIC board serial number the configuration will take place. This parameter is mandatory when two or more MUSIC boards are connected to the same USB host.
Log (-l, --loglevel) $argument
Activates log messages to provide further information about the software commands. The different log levels are: trace, debug, verbose ,info, warning, error, fatal, disable)
Header command (-F, --logformat) $argument
Use character '-' to disable any header printed in the command line windows.
Load command from file (-C, --args_file) $argument
File containing the arguments to be parsed to this command. Same format as the command line flags.

Example:

The following example shows the advanced mode and the re-configuration of the ASIC after 5 modifications.

```
$ music config_file -f calib/CF_MUSIC08R1_LCT5.calib -a -F -
Last MUSIC configuration modified on Wed Oct 26 16:18:46 2016
Last MUSIC configuration modified on Wed Oct 26 16:19:28 2016
Last MUSIC configuration modified on Wed Oct 26 16:19:29 2016
Last MUSIC configuration modified on Wed Oct 26 16:19:45 2016
Last MUSIC configuration modified on Wed Oct 26 16:20:03 2016
```

4.3 Show_config

Show Config command downloads the current MUSIC configuration and shows them into the screen.

Command:

show_config

Input parameters:

Save Configuration (-o, --dump_config_file) $file
Dump current configuration to the specified file.

Optional input parameters:

Hexadecimal (-X, --hex)
Show values in hexadecimal

Generic options:

Help (-h, --help)
Shows the help message.
Board (-b,--board) $argument
Indicates in which MUSIC board serial number the configuration will take place. This parameter is mandatory when two or more MUSIC boards are connected to the same USB host.
Log (-l, --loglevel) $argument
Activates log messages to provide further information about the software commands. The different log levels are: trace, debug, verbose ,info, warning, error, fatal, disable)
Header command (-F, --logformat) $argument
Use character '-' to disable any header printed in the command line windows.
Load command from file (-C, --args_file) $argument
File containing the arguments to be parsed to this command. Same format as the command line flags.

Output:

MUSIC registers configuration.

Example:

```
$ music show_config

* Channel configuration

+---------+-------------------------+-----------------------------------------------------------------------------+
| Channel | Input state register    | Channel register                                                            |
| Channel | EN_CH | DIS_IN | V_OFFSET | HL | EN_CH_SW | EN_PZ | EN_DRV_SE | V_TH | HL_COMP | EN_COMP_SW | EN_PZ_COMP |
+---------+-------+--------+----------+----+----------+-------+-----------+------+---------+------------+------------+
|       0 |     1 |      0 |      255 |  1 |        1 |     1 |         1 |  120 |       1 |          1 |          1 |
+---------+-------+--------+----------+----+----------+-------+-----------+------+---------+------------+------------+
|       1 |     1 |      0 |      255 |  1 |        1 |     1 |         1 |  120 |       1 |          1 |          1 |
+---------+-------+--------+----------+----+----------+-------+-----------+------+---------+------------+------------+
|       2 |     1 |      0 |      255 |  1 |        1 |     1 |         1 |  120 |       1 |          1 |          1 |
+---------+-------+--------+----------+----+----------+-------+-----------+------+---------+------------+------------+
|       3 |     1 |      0 |      255 |  1 |        1 |     1 |         1 |  120 |       1 |          1 |          1 |
+---------+-------+--------+----------+----+----------+-------+-----------+------+---------+------------+------------+
|       4 |     1 |      0 |      255 |  1 |        1 |     1 |         1 |  120 |       1 |          1 |          1 |
+---------+-------+--------+----------+----+----------+-------+-----------+------+---------+------------+------------+
|       5 |     1 |      0 |      255 |  1 |        1 |     1 |         1 |  120 |       1 |          1 |          1 |
+---------+-------+--------+----------+----+----------+-------+-----------+------+---------+------------+------------+
|       6 |     1 |      0 |      255 |  1 |        1 |     1 |         1 |  120 |       1 |          1 |          1 |
+---------+-------+--------+----------+----+----------+-------+-----------+------+---------+------------+------------+
|       7 |     1 |      0 |      255 |  1 |        1 |     1 |         1 |  120 |       1 |          1 |          1 |
+---------+-------+--------+----------+----+----------+-------+-----------+------+---------+------------+------------+
* Global configuration

+- Pole zero conf --------------+
| LOW_AT_LAD | CAP_PZ | RLAD_PZ |
+------------+--------+---------+
|          1 |     12 |       5 |
+------------+--------+---------+
+- Bias voltages ---------------------------------+
| VDC_LG | VDC_HG | VDC_CH | VCM | VBG_ADJ | V_LIM |
+--------+--------+--------+-----+---------+-------+
|    144 |    144 |    111 | 129 |       6 |    38 |
+--------+--------+--------+-----+---------+-------+
+- Bias currents ------------------------------------------------------------------------------------------+
| IB_COMP | IB_OP_SE | IB_OP_DIFF | IB_AB_SE | IB_AB_DIFF | IB_AB_PZ | IB_PAIR | IB_IN | IB_PZ_BUF |
+---------+----------+------------+----------+------------+----------+---------+-------+-----------+
|       3 |        3 |          3 |       15 |         15 |        4 |       4 |    25 |         3 |
+---------+----------+------------+----------+------------+----------+---------+-------+-----------+
+- Sum block ----------------------------------------------------+
| HL_SUM | EN_DIFF_DRV_HL | EN_BYPASS_HL | EN_PZ_HL | EN_CH_SUM |
+--------+----------------+--------------+----------+-----------+
|      3 |              3 |            0 |        3 |       255 |
+--------+----------------+--------------+----------+-----------+
```

4.4 Threshold Scan

This command measures the comparator threshold for each of the MUSIC digital outputs. First, it looks for the optimal reference voltage region (parameter VBG with a range between 0 to7) and then it sweeps the DAC output which generates V_{TH} from 0 to 511 counts. The output of this test is a CSV file where it can be seen the transition between VoSE[i] from 0 to 1 (S-curve).

Command:

threshold_scan

Input parameters:

Results File (-r,--results) $filename

Indicates the file where threshold scan measurements will be stored. MANDATORY

Optional input parameters:

Duty Cycle Measurement Window (-w,--dutywindow) $argument
Select the duty cycle analysis window (in us). The maximum window MUST not exceed $2^{24}/100$ ~ 166.000 us. The wider the window, the finer the measurements.

Generic options:

Help (-h, --help)
Shows the help message.
Board (-b,--board) $argument
Indicates in which MUSIC board serial number the configuration will take place. This parameter is mandatory when two or more MUSIC boards are connected to the same USB host.
Log (-l, --loglevel) $argument
Activates log messages to provide further information about the software commands. The different log levels are: trace, debug, verbose ,info, warning, error, fatal, disable)
Header command (-F, --logformat) $argument
Use character '-' to disable any header printed in the command line windows.
Load command from file (-C, --args_file) $argument
File containing the arguments to be parsed to this command. Same format as the command line flags.

Output:

This command will compute the optimal bandgap voltage (VBG) for the current MUSIC configuration and then it will show the threshold scan progress. An example of the output file can be found in Appendix: Threshold Scan results file. The output is essentially a 512x8 matrix with the normalized output level (values between 0 and 1) for each of the 512 thresholds and for each of the 8 MUSIC digital channels.

The output of this command gives a voltage threshold corresponding to the transition between 0 and 1, i.e., it gives the threshold value corresponding to a normalized output of 0.5. This threshold corresponds to the point where the same amount of '0's' and '1's' is detected. This transition corresponds to point where the dc voltage of the electronics and the configured threshold is rather close to the noise level. Notice that there are two noise components: the intrinsic electronics noise (observable when SiPM HV is off , i.e., the SiPM is not connected to the input) and SiPM noise produced by dark counts. Lastly, the options to be used in the 'config' command to configure the selected thresholds and the reference voltage can be seen at the end of the output printed.

Observe that the threshold scan must be performed with the digital mode enabled, otherwise the threshold cannot be obtained. An error message will appear in case the single ended channels are set to analog.

```
E MUSIC Single-Ended outputs are not configured in digital mode. Add '-d'
option when configuring MUSIC electronics.
```

Converting V_{TH} in DAC counts to Volts:

- Bandgap Voltage from Post-layout simulations as a function of VBG[2:0].
  vbgVoltageListMillivolts = [487.22 , 730.92 , 974.62 , 1218.3 , 1462.0 , 1705.7 , 1949.4 , 2436.8]
- vbg is a number between 0 and 7.
  vbgMillivolts = vbgVoltageListMillivolts[vbg]
- vth fine component depends on DAC counts.
  vthFineMillivolts = 1637.79 - 3.1445*vthDacCounts
- vth depends on fine vth and vbg.
  vthVolts = (1.5*vbgMillivolts - 0.5*vthFineMillivolts) / 1000

Example:

```
$ music threshold_scan -r thScanResults.txt 5 12 -m 4 -w 10000 -F –

+- Global configuration ------------------+
| LOW_AT_LAD | CAP_PZ | RLAD_PZ | VBG_ADJ |
+------------+--------+---------+---------+
|          1 |      5 |       3 |       5 |
+------------+--------+---------+---------+
+- Channel configuration------------------+
| Channel | ENABLED | MODE | EN_PZ | V_TH |
+---------+---------+------+-------+------+
|       0 |       1 |  DIG |     1 |  511 |
+---------+---------+------+-------+------+
|       1 |       1 |  DIG |     1 |  511 |
+---------+---------+------+-------+------+
|       2 |       1 |  DIG |     1 |  511 |
+---------+---------+------+-------+------+
|       3 |       1 |  DIG |     1 |  511 |
+---------+---------+------+-------+------+
|       4 |       0 |  DIG |     1 |  511 |
+---------+---------+------+-------+------+
|       5 |       0 |  DIG |     1 |  511 |
+---------+---------+------+-------+------+
|       6 |       0 |  DIG |     1 |  511 |
+---------+---------+------+-------+------+
|       7 |       0 |  DIG |     1 |  511 |
+---------+---------+------+-------+------+
Performing Bandgap Voltage (VBG) scan...
The optimal VBG is 5

Vth = {   0 ,  63 } ..............................................................
Vth = {  64 , 127 } ..............................................................
Vth = { 128 , 191 } ..............................................................
Vth = { 192 , 255 } ..............................................................
Vth = { 256 , 319 } ..............................................................
Vth = { 320 , 383 } ..............................................................
Vth = { 384 , 447 } ..............................................................
Vth = { 448 , 511 } ..............................................................

Digital test ended after 2.6435 seconds.

+- Optimal Vth parameters -----------------------------+
| CH 0 | CH 1 | CH 2 | CH 3 | CH 4 | CH 5 | CH 6 | CH 7 |
+------+------+------+------+------+------+------+------+
|  423 |  425 |  428 |  425 |  451 |  451 |  450 |  453 |
+------+------+------+------+------+------+------+------+

Command line options friendly copy&paste:
-g 5 -v 423 425 428 425 451 451 450 453
```

4.5 Photon/Dark Count Rate (PCR/DCR)

This command measures the photon count rate of MUSIC VoSE[7:0] digital outputs. User can provide a certain range of V_{TH}'s and configure the time window where Photon Count Rate (PCR) is measured. The aim of displaying counters is nothing but noticing user about Photon Count Rate (PCR) changes without having to monitor digital outputs with an oscilloscope. By default, the software will output the photon counter of the last 10s window. Lastly, observe that this command can be used for both photon or dark count rate, the only change depends on whether the SIPM is switch on or off.

Command:

photon_count

Input parameters:

Results File (-r,--results)
Indicates the file where photon count measurements will be stored. MANDATORY

Optional input parameters:

V_{TH} Range (-v,--vthrange)
Select the range (inclusive in both bounds) of channel comparator V_{TH} to perform the Photon Count V_{TH} sweep. When no specified, the test is performed once with the current chip V_{TH} configuration.
Photon Count Measurement Window (-n,--nseconds)
Select the photon count analysis window (in seconds). The wider the window, the finer the measurements.

Output:

This command shows the average PCR per channel in Hz every 1 second for each of the V_{TH} values provided. An example of the output file can be found in Appendix: Photon Count results file. The output is essentially an Nx8 matrix with the normalized output level for each the N thresholds and for each of the 8 MUSIC digital channels. The header specifies the first V_{TH} and the last V_{TH}.

Example:

```
$ music config -f ./calib/MUSIC_M01_VDC_CF.vdc -e 0 1 2 3 4 5 6 7 -d
$ music threshold_scan -r kk.txt -w 1000
$ music config -f ./calib/MUSIC_M01_VDC_CF.vdc -e 0 4 -d -g 5 -v 358 349 350 349
353 355 358 351 -F –
$ music photon_count -r phCountRes.txt -n 3 -v 10 12 -F -
+- Global configuration ------------------+
| LOW_AT_LAD | CAP_PZ | RLAD_PZ | VBG_ADJ |
+------------+--------+---------+---------+
|          1 |     12 |       5 |       5 |
+------------+--------+---------+---------+
+- Channel configuration------------------+
| Channel | ENABLED | MODE | EN_PZ | V_TH |
+---------+---------+------+-------+------+
|       0 |       1 |  DIG |     0 |  358 |
+---------+---------+------+-------+------+
|       1 |       1 |  DIG |     0 |  349 |
+---------+---------+------+-------+------+
|       2 |       1 |  DIG |     0 |  350 |
+---------+---------+------+-------+------+
|       3 |       1 |  DIG |     0 |  349 |
+---------+---------+------+-------+------+
|       4 |       1 |  DIG |     0 |  353 |
+---------+---------+------+-------+------+
|       5 |       1 |  DIG |     0 |  355 |
+---------+---------+------+-------+------+
|       6 |       1 |  DIG |     0 |  358 |
+---------+---------+------+-------+------+
|       7 |       1 |  DIG |     0 |  351 |
+---------+---------+------+-------+------+
Setting VTH to 348 DAC counts...
Starting Photon Count Rate (PCR) measurement...
+- Photon counter Rate (PCR) per channel (Hz) for   3 seconds ----+
| CH 0 | CH 1 | CH 2 | CH 3 | CH 4 | CH 5 | CH 6 | CH 7 | Seconds |
+------+------+------+------+------+------+------+------+---------+
| 2.5e6|    0 |    0 |    0 | 4.5e6|    0 |    0 |    0 | 1.04858 |
+------+------+------+------+------+------+------+------+---------+
| CH 0 | CH 1 | CH 2 | CH 3 | CH 4 | CH 5 | CH 6 | CH 7 | Seconds |
+------+------+------+------+------+------+------+------+---------+
| 2.3e6|    0 |    0 |    0 | 4.2e6|    0 |    0 |    0 | 2.09715 |
+------+------+------+------+------+------+------+------+---------+
| CH 0 | CH 1 | CH 2 | CH 3 | CH 4 | CH 5 | CH 6 | CH 7 | Seconds |
+------+------+------+------+------+------+------+------+---------+
| 2.7e6|    0 |    0 |    0 | 4.8e6|    0 |    0 |    0 | 3.00941 |
Setting VTH to 349 DAC counts...
Starting Photon Count Rate (PCR) measurement...
+- Photon counter Rate (PCR) per channel (Hz) for   3 seconds ----+
| CH 0 | CH 1 | CH 2 | CH 3 | CH 4 | CH 5 | CH 6 | CH 7 | Seconds |
+------+------+------+------+------+------+------+------+---------+
| 2.4e6|    0 |    0 |    0 | 4.5e6|    0 |    0 |    0 | 1.04858 |
+------+------+------+------+------+------+------+------+---------+
| CH 0 | CH 1 | CH 2 | CH 3 | CH 4 | CH 5 | CH 6 | CH 7 | Seconds |
+------+------+------+------+------+------+------+------+---------+
| 2.9e6|    0 |    0 |    0 | 4.8e6|    0 |    0 |    0 | 2.09715 |
+------+------+------+------+------+------+------+------+---------+
| CH 0 | CH 1 | CH 2 | CH 3 | CH 4 | CH 5 | CH 6 | CH 7 | Seconds |
+------+------+------+------+------+------+------+------+---------+
|    0 |    0 |    0 |    0 | 4.9e6|    0 |    0 |    0 | 3.00941 |
+------+------+------+------+------+------+------+------+---------+
Setting VTH to 350 DAC counts...
Starting Photon Count Rate (PCR) measurement...
+- Photon counter Rate (PCR) per channel (Hz) for   3 seconds ----+
| CH 0 | CH 1 | CH 2 | CH 3 | CH 4 | CH 5 | CH 6 | CH 7 | Seconds |
+------+------+------+------+------+------+------+------+---------+
| 1.3e6|    0 |    0 |    0 |5.5e6 |    0 |    0 |    0 | 1.04858 |
+------+------+------+------+------+------+------+------+---------+
| CH 0 | CH 1 | CH 2 | CH 3 | CH 4 | CH 5 | CH 6 | CH 7 | Seconds |
+------+------+------+------+------+------+------+------+---------+
| 1.4e6|    0 |    0 |    0 |6.1e6 |    0 |    0 |    0 | 2.09715 |
+------+------+------+------+------+------+------+------+---------+
| CH 0 | CH 1 | CH 2 | CH 3 | CH 4 | CH 5 | CH 6 | CH 7 | Seconds |
+------+------+------+------+------+------+------+------+---------+
| 1.5e6|    0 |    0 |    0 | 6.3e6|    0 |    0 |    0 | 3.00941 |
+------+------+------+------+------+------+------+------+---------+
```

4.6 TDC calibration

Acquires ToT pulses for each of the 16 HR-FlexToT channel outputs and calibrates each of the TDC bins, assuming that input pulses are random and follow a uniform random distribution. If this is not possible, the calibration pulse can be used instead (SMA with the TDC_CALIB_PULSE label). When the calibration pulse is used, the 16 TDC channels will receive the same input pulse. An external Pulse Pattern Generator (PPG) is required (3.3V single-ended CMOS) to generate random pulses (> 6-ns pulse width). It is important to highlight that this procedure should only be done in case that there were evidences that TDC calibration is not valid for the current setup conditions, or after a firmware upgrade. Otherwise, please use the TDC-FPGA calibration file provided.

Command:

tdc_calib

Input parameters:

Results File (-o,--calib_outfile)

The output calibration file containing the required information for the TDC-FPGA to convert acquired events into picoseconds units. The file not only contains a lookup table for each of the sampling channels, but also information regarding the calibration conditions.

Optional input parameters:

Number of events (-e,--events)

The number of total events to read before stopping acquisition. One event corresponds to a pulse, where rising edge is the Time-of-Arrival, and the difference between rising and falling edge is the pulse width.

Acquisition time (-m,--msecs)

The acquisition time of the test in milliseconds. Note that either number of events or acquisition time must be provided.

Raw dump file (-D,--raw_dump_file)

Dumps the data coming from the USB in raw format file. It can be useful to trace and report errors.

Checkpoint byte in data frame (-f,--frames_between_checks)

A checkpoint byte is added every N data frames coming from the USB acquisition system. This byte is useful to check data alignment and to ensure that no data is lost in the transmission. The minimum value is 16 and the maximum is 4080. The default value does not need to be changed.

Acquisition rate (-M,--max_rate_kevts)

The maximum acquisition rate can be limited in case that the acquisition system host is not powerful enough (e.g. Raspberry Pi or similar). Apply this filter only if the number of checkpoint issues (use the -l debug option for debugging) grows substantially (hundreds of errors per second).

Minimum pulse width (-W,--min_pwidth)

This option prevents the FPGA to send TDC events the pulse of which is lower than a certain value. This is very useful to reduce the capture file size and also to prevent USB DAQ throttling. The value is specified in nanoseconds.

USB packet size (-p,--packet_size)

Length (in bytes) of the USB packet to be read. Big packet sizes are more efficient in terms of throughput.

Enable TDC-FPGA channels (-E,--en_channels)

This option lets the user to individually enable/disable TDC-FPGA channels. When using this option, a list of 16 values with either 0 or 1 value must be provided. Channel 0 corresponds enable corresponds to the first value, while the last value for channel 15. All channels are enabled by default.

Calibration with external pulse (-P,--calib_pulse)

When this option is provided, all the TDC channels will rely on TDC_CALIB_PULSE SMA input port. This can be very useful to calibrate the board with an external pulse generator.

Maximum bin width (-n,--max_bin)

Modifies the maximum bin width computed by the calibration software to accept the acquisition step as good. If maximum bin width is not fulfilled, TDC-FPGA sampling clock frequency will increase on the next calibration iteration until maximum frequency is reached.

Maximum bin width standard deviation (-d,--max_std)

Modifies the maximum bin width standard deviation allowed by the calibration software.

Carry chain minimum stages (-i,--min_clearance)

Establishes the minimum number of stages of the 64-carry chain which its delay is higher than the clock period. It is recommendable that a margin of, at least, 3 carry stages.

Clock output ON (-k,--clk_mon)

When this option is provided, a copy of the TDC-FPGA internal reference clock is output though TDC_CLK_MON_SE SMA output port and TDC_CLK_MON LVDS output port. This is useful to synchronize two boards with the same reference clock.

External reference clock (-K,--clk_ext)

When this option is provided, TDC-FPGA will use an external reference clock.

FPGA global clock coarse counter external reset (-c,--coarse_rst)

When this option is provided, TDC_COARSE_RST SMA port is used to reset the TDC-FPGA global coarse counter. This is useful to synchronize different boards.

FPGA sampling range (-r,--freq_range)

Mandatory. The range of TDC-FPGA internal sampling frequencies to be tested. Lower frequency values save power consumption, but makes the 64 fast-carry chains for the given clock frequency (events will be lost), while higher frequencies will solve the problem. Note that frequencies higher than 450 MHz may produce clock violations. Thus, the typical adjust range goes from 380 to 420 MHz.

Frequency steps (-t,--freq_steps)

The step size of the calibration test in MHz. For low values (< 5 MHz), the optimal frequency can be optimized, but the calibration time will increase dramatically. Moreover, not all the frequencies can be synthesized due to internal PLL limitations (M/N factor).

Info rate (-s,--events_progress)

When the acquisition is set by the number of events, this option lets user to select every how many events the INFO message will appear with the current number of captured events.

Output:

Example:

This example calibrates the all the FPGA-TDC channels with 100000 events generated by an external PPG connected to TDC_CALIB_PULSE SMA port. As for the optimization values, the clearance is set to 3, and the range of internal PLL frequencies to test ranges from 380 to 420 MHz:

```
hrflextot tdc_calib -e 100000 -o hrft_calib.txt -P -i 3 -r 380 420
```

4.7 TDC acquisition

Acquires ToT pulses for each of the 16 HR-FlexToT channel outputs.

Command:

tdc_acq

Input parameters:

Calibration file (-I,--calib_file)
The calibration file generated by tdc_calib command, which contains a lookup table to convert acquired events into picoseconds units. It is important to highlight that each board has its own calibration file, and this can also vary after a firmware upgrade.
Output file (-o,--output_file)
The name of the CSV file containing the acquired events. The file contains as many rows as events, and each row contains the Event ID, the HR-FlexToT channel number (from 0 to 15), the timestamp (in ps) corresponding to the ToA, and finally the pulse width (in ps). The software automatically fixes the FPGA ToA counter overflow which occurs every ~100 ms. Note that, unless the coarse counter external reset capability is enabled, the first ToA timestamp will be an arbitrary value. A second file with the suffix .uncalib.txt is also generated. It contains the TDC information as it arrives from the FPGA. This file is very useful for troubleshooting purposes.

Optional input parameters:

Total number of events (-e,--events)
The number of total events to read before stopping acquisition. One event corresponds to a pulse, where rising edge is the Time-of-Arrival, and the difference between rising and falling edge is the pulse width.
Test acquisition time (-m,--msecs)
The acquisition time of the test in milliseconds. Note that either number of events or acquisition time must be provided.
External TDC calibration pulse (-P,--calib_pulse)
When this option is provided, all the TDC channels will rely on TDC_CALIB_PULSE SMA input port. This can be very useful to calibrate the board with an external pulse generator.
Dump raw data on file (-D,--raw_dump_file)
Dumps the data coming from the USB in raw format file. It can be useful to trace and report errors.
Frames between checks (-f,--frames_between_checks)
A checkpoint byte is added every N data frames coming from the USB acquisition system. This byte is useful to check data alignment and to ensure that no data is lost in the transmission. The minimum value is 16 and the maximum is 4080. The default value does not need to be changed.
Maximum acquisitioin rate (-M,--max_rate_kevts)
The maximum acquisition rate can be limited in case that the acquisition system host is not powerful enough (e.g. Raspberry Pi or similar). Apply this filter only if the number of checkpoint issues (use the -l debug option for debugging) grows substantially (hundreds of errors per second).
Minimum pulse width (-W,--min_pwidth)

This option prevents the FPGA to send TDC events the pulse of which is lower than a certain value. This is very useful to reduce the capture file size and also to prevent USB DAQ throttling. The value is specified in nanoseconds.

USB packet size (-p,--packet_size)

Length (in bytes) of the USB packet to be read. Big packet sizes are more efficient in terms of throughput.

Enable channels (-E,--en_channels)

This option lets the user to individually enable/disable TDC-FPGA channels. When using this option, a list of 16 values with either 0 or 1 value must be provided. Channel 0 corresponds enable corresponds to the first value, while the last value for channel 15. All channels are enabled by default.

Reference output clock ON (-k,,--clk_mon)

When this option is provided, a copy of the TDC-FPGA internal reference clock is output though TDC_CLK_MON_SE SMA output port and TDC_CLK_MON LVDS output port. This is useful to synchronize two boards with the same reference clock.

FPGA global clock coarse counter external reset (-c,--coarse_rst)

When this option is provided, TDC_COARSE_RST SMA port is used to reset the TDC-FPGA global coarse counter. This is useful to synchronize different boards.

Info rate (-s,--events_progress)

When the acquisition is set by the number of events, this option lets user to select every how many events the INFO message will appear with the current number of captured events.

Example:

This example acquires during 5 seconds from the 16 FPGA-TDC channels. The calibration file is *hrft_calib.txt* and the converted events are stored into *tdc_data.txt file*.

```
hrflextot tdc_acq -s 5000 -i hrft_calib.txt -o tdc_data.txt
```

5 Logs

This software provides a flexible log capability which enables debugging at different levels. There are different log levels:

Level	Description
global	Generic level that represents all levels. Useful when setting global configuration for all levels.
trace	Information that can be useful to back-trace certain events - mostly useful than debug logs.
debug	Informational events most useful for developers to debug application. Only applicable if NDEBUG is not defined (for non-VC++) or _DEBUG is defined (for VC++).
fatal	Very severe error event that will presumably lead the application to abort.
error	Error information but will continue application to keep running.
warning	Information representing errors in application but application will keep running.
info	Mainly useful to represent current progress of application.

The default log level is **info**. To change the log level, the following option has to be specified:

All the logs displayed on the screen are also available in the path ./logs/myeasylog.log. This path is relative to the current working directory where MUSIC software was launched.

6 Revision history

1. 29 – June – 2016: First version of the software manual.
2. 27 – October – 2016: Second version of the music software. Includes description of experimental procedures.
3. 17 – January – 2017 : Added computer requirements: Installation of FTDI drivers.
4. 8 – March – 2017 : Changes in how the machine has to be configured
5. 16 – March – 2017 : Fix typo
6. 23 – May – 2017 : Added a note to tell the user to reconfigure the music board after an abrupt exit of the software.
7. 02 – Dic – 2017: modified format, tables were not properly placed.

A. Appendix: MUSIC IC Calibration File example

```
############################################################
#############   MUSIC CALIBRATION FILE   ###################
############################################################

#

#File Version     : v1
#Date (YYYY/MM/DD): 2016/05/25
#Time (HH:MM)     : 18:28
#MUSIC Chip ID    : 0

############################################################
# MUSIC Bias Registers Parameters:
############################################################

LOW_AT_LAD     = 1
VDC_LG         = 120
VDC_HG         = 105
VDC_CH         = 118
VCM            = 129
IB_COMP        = 3
VBG_ADJ        = 5
IB_OP_SE       = 3
IB_AB_SE       = 15
HL_SUM         = 3
EN_DIFF_DRV_HL = 3
EN_BYPASS_HL   = 0
EN_PZ_HL       = 3
EN_CH_SUM      = 255
V_LIM          = 38
CAP_PZ         = 12
RLAD_PZ        = 5
IB_AB_DIFF     = 15
IB_OP_DIFF     = 3
IB_PAIR        = 4
IB_AB_PZ       = 4
IB_IN          = 25
IB_PZ_BUF      = 3

############################################################
#Channel #0 Parameters:
############################################################

EN_CH[0]      = 1
DIS_IN[0]     = 0
V_OFFSET[0]   = 255
HL[0]         = 1
EN_CH_SW[0]   = 1
EN_PZ[0]      = 1
EN_DRV_SE[0]  = 1
V_TH[0]       = 454
HL_COMP[0]    = 1
EN_COMP_SW[0] = 1
EN_PZ_COMP[0] = 1

############################################################
#Channel #1 Parameters:
############################################################

EN_CH[1]      = 1
DIS_IN[1]     = 0
V_OFFSET[1]   = 255
HL[1]         = 1
EN_CH_SW[1]   = 1
EN_PZ[1]      = 1
EN_DRV_SE[1]  = 1
V_TH[1]       = 500
```

```
HL_COMP[1]    = 1
EN_COMP_SW[1] = 1
EN_PZ_COMP[1] = 1

############################################################
#Channel #2 Parameters:
############################################################

EN_CH[2]      = 1
DIS_IN[2]     = 0
V_OFFSET[2]   = 255
HL[2]         = 1
EN_CH_SW[2]   = 1
EN_PZ[2]      = 1
EN_DRV_SE[2]  = 1
V_TH[2]       = 55
HL_COMP[2]    = 1
EN_COMP_SW[2] = 1
EN_PZ_COMP[2] = 1

############################################################
#Channel #3 Parameters:
############################################################

EN_CH[3]      = 1
DIS_IN[3]     = 0
V_OFFSET[3]   = 255
HL[3]         = 1
EN_CH_SW[3]   = 1
EN_PZ[3]      = 1
EN_DRV_SE[3]  = 1
V_TH[3]       = 120
HL_COMP[3]    = 1
EN_COMP_SW[3] = 1
EN_PZ_COMP[3] = 1

############################################################
#Channel #4 Parameters:
############################################################

EN_CH[4]      = 1
DIS_IN[4]     = 0
V_OFFSET[4]   = 255
HL[4]         = 1
EN_CH_SW[4]   = 1
EN_PZ[4]      = 1
EN_DRV_SE[4]  = 1
V_TH[4]       = 120
HL_COMP[4]    = 1
EN_COMP_SW[4] = 1
EN_PZ_COMP[4] = 1

############################################################
#Channel #5 Parameters:
############################################################

EN_CH[5]      = 1
DIS_IN[5]     = 0
V_OFFSET[5]   = 255
HL[5]         = 1
EN_CH_SW[5]   = 1
EN_PZ[5]      = 1
EN_DRV_SE[5]  = 1
V_TH[5]       = 120
HL_COMP[5]    = 1
EN_COMP_SW[5] = 1
EN_PZ_COMP[5] = 1

############################################################
#Channel #6 Parameters:
############################################################
```

```
EN_CH[6]      = 1
DIS_IN[6]     = 0
V_OFFSET[6]   = 255
HL[6]         = 1
EN_CH_SW[6]   = 1
EN_PZ[6]      = 1
EN_DRV_SE[6]  = 1
V_TH[6]       = 120
HL_COMP[6]    = 1
EN_COMP_SW[6] = 1
EN_PZ_COMP[6] = 1

############################################################
#Channel #7 Parameters:
############################################################

EN_CH[7]      = 1
DIS_IN[7]     = 0
V_OFFSET[7]   = 255
HL[7]         = 1
EN_CH_SW[7]   = 1
EN_PZ[7]      = 1
EN_DRV_SE[7]  = 1
V_TH[7]       = 240
HL_COMP[7]    = 1
EN_COMP_SW[7] = 1
EN_PZ_COMP[7] = 1
```

B. Appendix: Threshold Scan results file

An example of the results from the threshold scan is here depicted. The threshold scan shows the normalized output value for all the 512 thresholds, omitted thresholds corresponds to all 0's at the beginning of the file or all 1's at the end of the file.

```
#VBG = 5
#NSamples = 10000000
0       0       0       0       0       0       0       0
0       0       0       0       0       0       0       0
0       0       0       0       0       0       0       0
.
.
.
.
0       0       0       0       0       0       0       0
0       0       0       0       0       0       0       0
0       0       0       0       0       0       0       0
0       0       0.0006447       0       0       0       0       0
0       0       0.108747        0       0       0       0       0
0       0       0.544242        0       0       0       0       0
0       0       0.967696        0       0       0       0       0
0       0       0.999989        0       0       0       0       0
0       2.01e-05        1       3e-07   1.86e-05        0       0       0
0       0.082009        1       0.0007941       0.0224623       0       0.0008474       0
0       0.370774        1       0.0787876       0.693127        0       0.636837        0
0       0.962392        1       0.0918962       0.983915        0       0.979759        0
0       0.999982        1       0.526327        0.999993        0       0.999991        0
0       1       1       0.988971        1       0       1       0
0.001579        1       1       1       1       2.5e-06 1       0
0.136714        1       1       1       1       0.0049678       1       0
0.788312        1       1       1       1       0.289417        1       0
0.995304        1       1       1       1       0.969005        1       0
1       1       1       1       1       0.999881        1       0
1       1       1       1       1       1       1       7.3e-06
1       1       1       1       1       1       1       0.0783801
1       1       1       1       1       1       1       0.80279
1       1       1       1       1       1       1       0.99973
1       1       1       1       1       1       1       1
1       1       1       1       1       1       1       1
1       1       1       1       1       1       1       1
.
.
.
.
1       1       1       1       1       1       1       1
1       1       1       1       1       1       1       1
1       1       1       1       1       1       1       1
```

C. Appendix: Photon Count results file

```
#PCR results file (multiple VTH).
#Capture Time (s) = 3
#VTH Init = 340
#VTH End = 360
3.85e+06 3.95e+06 3.93e+06 3.85e+06 3.83e+06 3.99e+06 3.76e+06 3.78e+06
3.87e+06 4.01e+06 4.01e+06 3.91e+06 3.90e+06 4.06e+06 3.74e+06 3.86e+06
3.98e+06 4.05e+06 4.00e+06 3.94e+06 3.94e+06 4.11e+06 3.88e+06 3.92e+06
4.04e+06 4.15e+06 4.08e+06 4.05e+06 4.00e+06 4.15e+06 3.94e+06 4.00e+06
4.66e+06 4.99e+06 5.06e+06 4.94e+06 4.65e+06 5.08e+06 4.04e+06 4.76e+06
5.22e+06 5.34e+06 5.44e+06 5.33e+06 5.06e+06 5.66e+06 4.47e+06 5.14e+06
5.78e+06 5.62e+06 5.65e+06 5.56e+06 5.51e+06 1.48e+06 4.75e+06 5.43e+06
1.00e+06 443942   490561   387755   405244   1.58e+06 5.76e+06 270126
3.84e+06 2.55e+06 2.60e+06 2.51e+06 2.52e+06 4.63e+06 2.81e+06 2.41e+06
3.42e+06 1.43e+06 1.43e+06 1.43e+06 1.43e+06 4.19e+06 2.69e+06 1.43e+06
613512   6.08e+06 6.07e+06 6.05e+06 6.08e+06 1.22e+06 1.55e+06 6.04e+06
244673   261056   260686   259948   259904   256848   1.98e+06 259630
324222   316597   316857   316482   316376   317177   638918   316382
621171   621214   621227   621154   621052   621211   627536   621060
841841   841959   841956   841871   841765   841930   841928   841767
1.09e+06 1.09e+06 1.09e+06 1.09e+06 1.09e+06 1.09e+06 1.09e+06 1.09e+06
1521.89  1627.89  1654.14  1559.11  1442.14  1636.2   1540.83  1450.12
5828.05  5912.78  5925.41  5842.34  5807.44  5917.77  5817.41  5790.17
340.93   456.567  460.223  414.367  401.407  429.32   377.815  381.802
349.57   456.567  459.558  429.984  408.385  434.636  337.94   388.78
552.599  654.613  668.569  610.418  606.431  634.011  529.007  596.13
```