EBMM3 documentation

Release 2.2.0-dev3217

Sercalo Microtechnology Ltd

Apr 02, 2025

CONTENTS:

EBMM3 (Evaluation Board for Magnetic Mirrors) is an electronics module to interface Sercalo MEMS magnetic Mirror product family as illustrated below. The MEMS is connected to the EBMM3. The user communicates with the EBMM3 to control the MEMS via a command line interface (cli). It allows the user to:

- Build reference patterns
- Set a control strategy
- Read sensor values
- Configure the MEMS

CHAPTER

ONE

GETTING STARTED

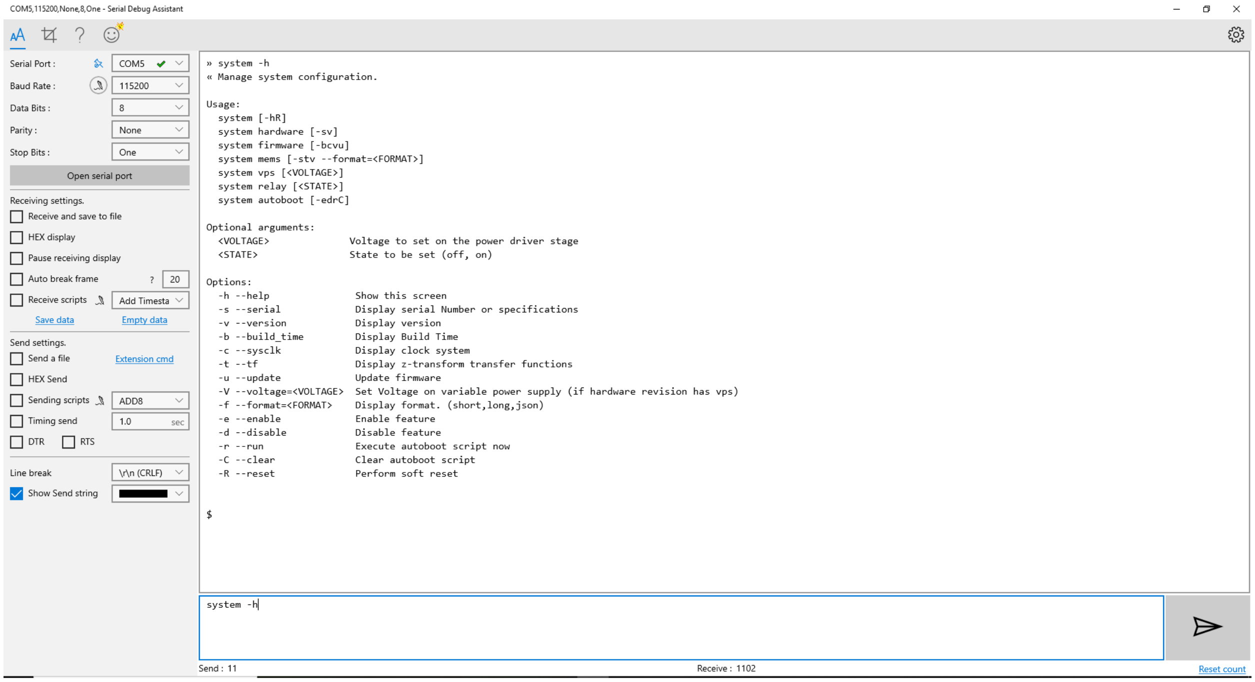
Please follow this step to connect your hardware:

1. Connect the 5V AC adaptor to the back-side of the EBMM3
2. Connect your Computer to the usb port next to the power connector.
3. Connect the MEMS to the 16-pin on the front panel.
4. Press the reset button or trigger a reset by sending command `system -R`.

Once the electronics is switch on, the communication between the user and the EBMM3 is achieved using the Command Line Interface (CLI) through the USB port (see *CLI Standard Usage* section).

1.1 Hyperterminal installation and configuration (EBMM3 using USB interface)

The user need an hyperterminal to be able to communicate with the EBMM3 product using the CLI interface. Sercalo suggests to use Serial Debug Assistant on Windows 10/11 and cutecom on Ubuntu.



COM5,115200,None,8,One - Serial Debug Assistant

Serial Port : COM5 ✓
Baud Rate : 115200
Data Bits : 8
Parity : None
Stop Bits : One

Receiving settings.
□ Receive and save to file
□ HEX display
□ Pause receiving display
□ Auto break frame ? 20
□ Receive scripts Add Timesta

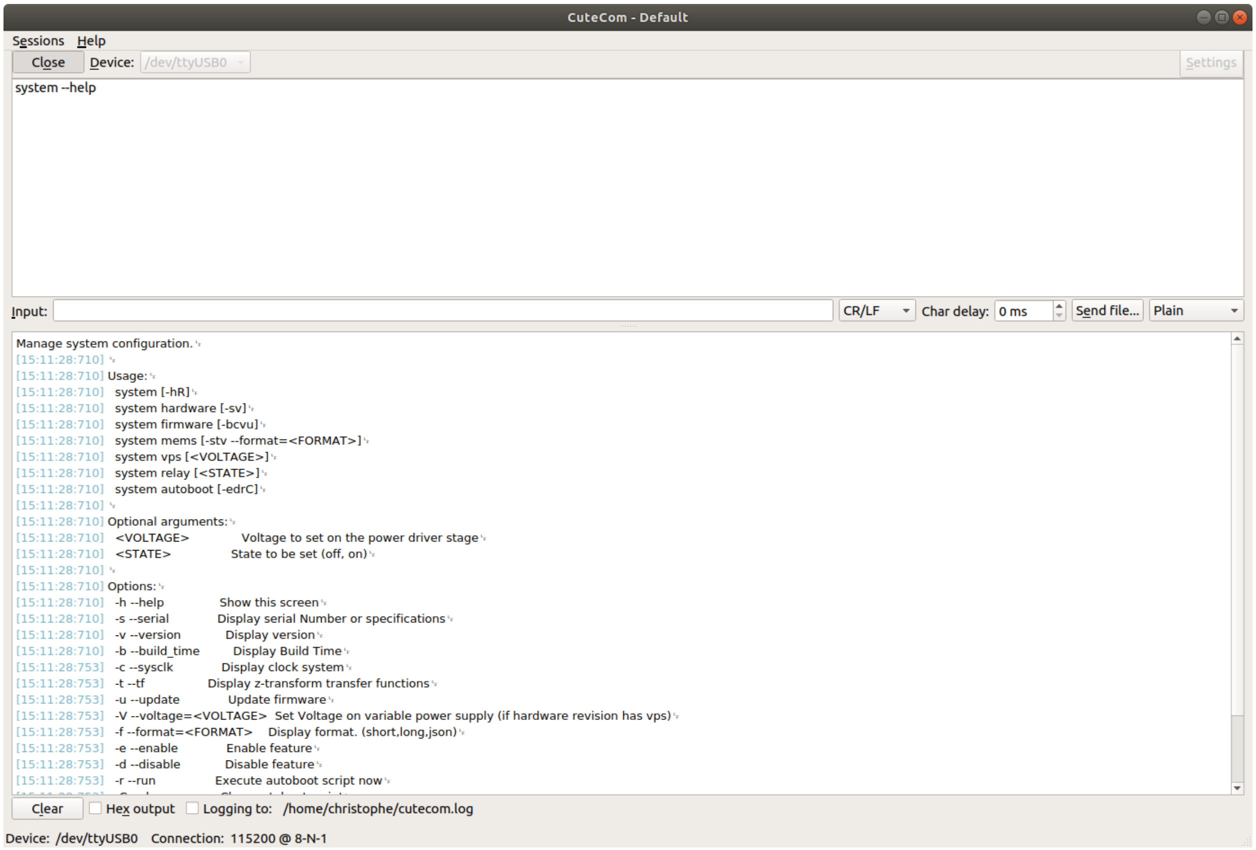
Send settings.
□ Send a file
□ HEX Send
□ Sending scripts ADD8
□ Timing send 1.0 sec
□ DTR □ RTS

Line break \r\n (CRLF)
☑ Show Send string

```
» system -h
« Manage system configuration.

Usage:
  system [-hR]
  system hardware [-sv]
  system firmware [-bcvu]
  system mems [-stv --format=<FORMAT>]
  system vps [<VOLTAGE>]
  system relay [<STATE>]
  system autoboot [-edrC]

Optional arguments:
  <VOLTAGE>              Voltage to set on the power driver stage
  <STATE>                State to be set (off, on)

Options:
  -h --help               Show this screen
  -s --serial             Display serial Number or specifications
  -v --version            Display version
  -b --build_time         Display Build Time
  -c --sysclk             Display clock system
  -t --tf                 Display z-transform transfer functions
  -u --update             Update firmware
  -V --voltage=<VOLTAGE>  Set Voltage on variable power supply (if hardware revision has vps)
  -f --format=<FORMAT>    Display format. (short,long,json)
  -e --enable             Enable feature
  -d --disable            Disable feature
  -r --run                Execute autoboot script now
  -C --clear              Clear autoboot script
  -R --reset              Perform soft reset


$
```

```
system -h
```

Send : 11 Receive : 1102



Device: /dev/ttyUSB0

system --help

Input:

CR/LF Char delay: 0 ms Plain

```
Manage system configuration.
[15:11:28:710]
[15:11:28:710] Usage:
[15:11:28:710]   system [-hR]
[15:11:28:710]   system hardware [-sv]
[15:11:28:710]   system firmware [-bcvu]
[15:11:28:710]   system mems [-stv --format=<FORMAT>]
[15:11:28:710]   system vps [<VOLTAGE>]
[15:11:28:710]   system relay [<STATE>]
[15:11:28:710]   system autoboot [-edrC]
[15:11:28:710]
[15:11:28:710] Optional arguments:
[15:11:28:710]   <VOLTAGE>             Voltage to set on the power driver stage
[15:11:28:710]   <STATE>               State to be set (off, on)
[15:11:28:710]
[15:11:28:710] Options:
[15:11:28:710]   -h --help             Show this screen
[15:11:28:710]   -s --serial           Display serial Number or specifications
[15:11:28:710]   -v --version          Display version
[15:11:28:710]   -b --build_time       Display Build Time
[15:11:28:753]   -c --sysclk           Display clock system
[15:11:28:753]   -t --tf               Display z-transform transfer functions
[15:11:28:753]   -u --update           Update firmware
[15:11:28:753]   -V --voltage=<VOLTAGE>  Set Voltage on variable power supply (if hardware revision has vps)
[15:11:28:753]   -f --format=<FORMAT>    Display format. (short,long,json)
[15:11:28:753]   -e --enable           Enable feature
[15:11:28:753]   -d --disable          Disable feature
[15:11:28:753]   -r --run              Execute autoboot script now
```

□ Hex output □ Logging to: /home/christophe/cutecom.log

Device: /dev/ttyUSB0 Connection: 115200 @ 8-N-1

The hyperterminal configuration is detailed below.

Parameter	value	Remark
Baudrate	115200	Default value
Line ending	\r\n	
Flow Control	None	
Data bits	8	
Parity	None	
Stop bits	1	

1.2 First application examples

A list of the available commands can be shown using the following command:

```
$ help
The available commands are listed below. To get the help of a specific command, use
↪<command> --help syntax.
system, signal, control, laser, extension, record, failsafe, memory, sensor, sudo
```

The following commands can be used to check that the hardware set up is correctly enumerated:

```
$ system hardware
System hardware version:  H61-127-00, H61-128-00
```

(continues on next page)

(continued from previous page)

```
System serial number: 00000000
System variable power supply: 9.00-15.00V
$ system mems
MEMS SN: YYYY-WW-XXXXX
MEMS PN: MM2536-2-AU
AXIS: 0, CALIBSOURCE: 2, AMAX: 5.0, DCGAIN: -35.4503, FRES: 365.29, ZETA: 0.0173645,␣
↪RES: 9.8863
AXIS: 1, CALIBSOURCE: 2, AMAX: 5.0, DCGAIN: 29.7059, FRES: 226.64, ZETA: 0.00520325,␣
↪RES: 9.73086
```

A basic use case is illustrated below as a quick start. This example draw a 1° circle using the default control strategy.

```
$ signal generate -a x -w sine -A 1 -F 20.0 -o 0 -p 0
$ signal generate -a y -w sine -A 1 -F 20.0 -o 0 -p 90
$ control strategy on
```

The following command can be called to stop the control and force the MEMS to the zero position.

```
$ control strategy off
```

Note:

- The $ are displayed in the sample code for user readability but are not part of the command.
- The lines that do not start with the prompt command are the responses of the system.

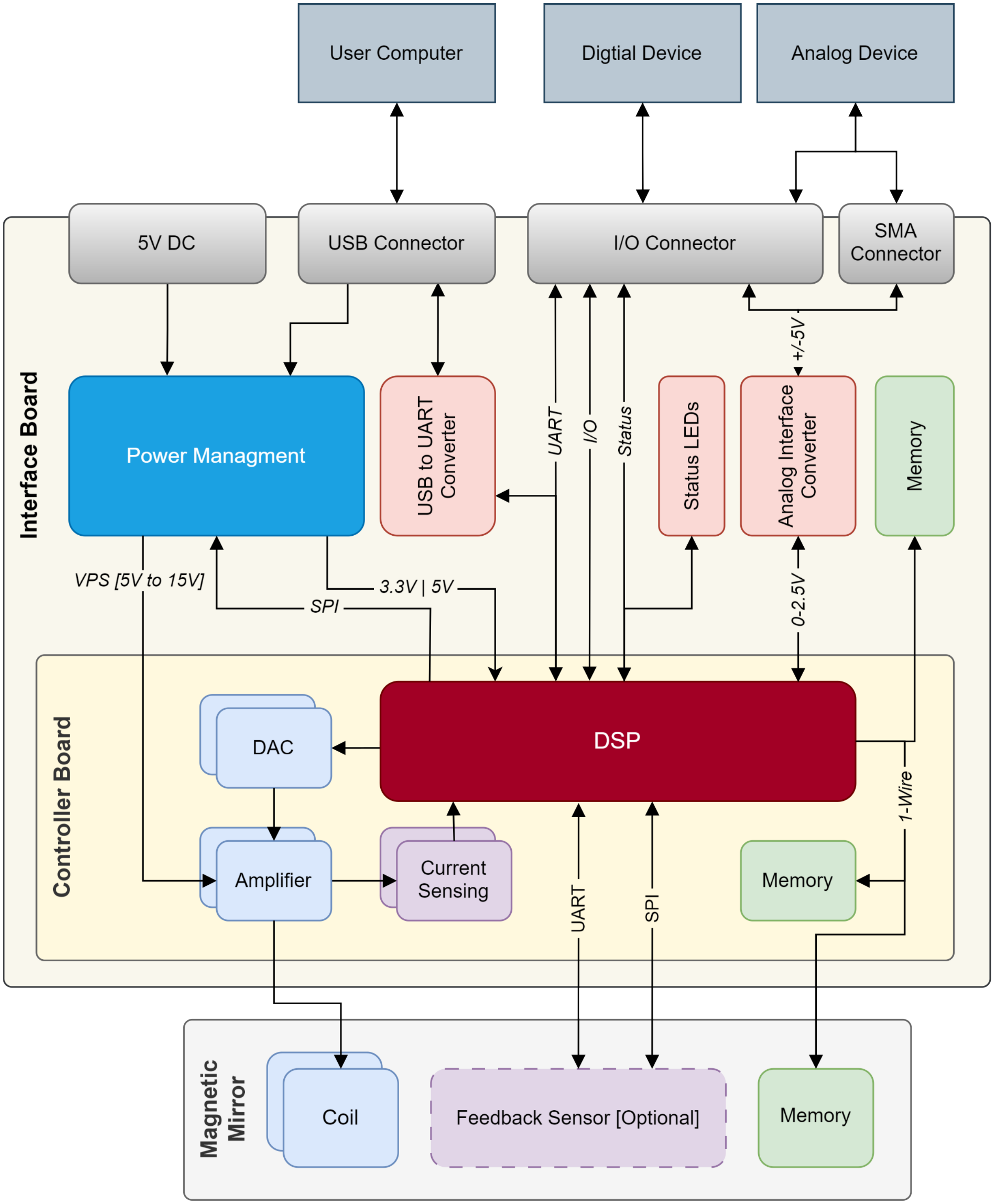
CHAPTER TWO

HARDWARE

The EBMM3 is sold as a complete development kit which includes two electronics:

- Driver board with a DSP unit and output connection for the MM mirrors.
- Interface board with USB interface, Analog IN/OUT, I/O extensions connector and Power management.

Both electronics are packaged in an enclosure. On demand, the driver board could be provided separately to integrate the controller inside a smaller package.


User Computer
Digtial Device
Analog Device
5V DC
USB Connector
I/O Connector
SMA Connector
Interface Board
Power Managment
USB to UART Converter
UART
I/O
Status
Status LEDs
+/-5V
Analog Interface Converter
Memory
VPS [5V to 15V]
3.3V | 5V
SPI
0-2.5V
Controller Board
DAC
DSP
1-Wire
Amplifier
Current Sensing
UART
SPI
Memory
Magnetic Mirror
Coil
Feedback Sensor [Optional]
Memory

2.1 Interfaces

2.1.1 UART

The main communication with the driver is done through the UART bus. One can use the I/O connector pin 5 & 6 or the USB-C connector which acts as a virtual COM port on the computer. The communication settings are listed in the table below. The system works as a unix-like CLI (Command-Line-Interface) (see *CLI Standard Usage* section).

Parameter	value	Remark
Baudrate	115200	Default value
Line ending	`\r\n`	
Flow Control	None	
Data bits	8	
Parity	None	
Stop bits	1	

2.1.2 SPI

Not available yet

2.1.3 User Programmable I/O

Five I/O pins are available to get info and trigger option during firmware execution. The `extension` module provides functionality for each I/O pins from `dig1` to `dig5` (see *User I/Os Pins Configuration* section).

2.1.4 Status Pins

Status pins are read-only I/O pins that are preconfigured with a specific output usage. On the EBMM3 Box, the status pins are displayed as LED for visual information.

Pin	Description
1	Indicate that an error occurs and all output to the MEMS were shut down. A reset is required.
2	The firmware is running and ready.
3	The controller loop is running, and the connected MM is activated.
4	The firmware is entering the control loop interruption.
5	The firmware command parser is handling UART input.

2.1.5 Laser synchronization

The laser pin is used to activate/deactivate an output laser by opening a MOSFET connected to GND. User is responsible to provide the required power. Typically, the MOSFET can handle LED current up to 200mA. The `laser` module is used to configure the laser output (see *External Laser* section).

2.1.6 Analog Input [16bit ADC]

Each axis can be controlled with a ±5V reference signal. The input signal is provided to the EBMM box from the SMA connector on the back panel or the I/O connector on the front panel. The analog signal is shaped linearly with 0V=0° and 4V=MaxTiltAngle. A negative input corresponds to a negative angle. The maximum tilt angle is defined by the connected MEMS mirror. Use module and command `signal input --source=analog` to set the reference signal as analog output (see *Analog Input* section).

2.1.7 Analog Output [12bit DAC]

An output signal could be sent from the EBMM3 box SMA connector on the back panel or the I/O connector on the front panel. Two outputs are provided as X or Y axis. Actually, this is just a naming convention as both outputs could be interchanged or provide other output type. Each output could provide analog feedback for the reference signal, the drive voltage, the measured current, or the measured feedback sensor. The `extension` module provides functionality for both analog pins `an1` (X) and `an2` (Y) (see *User I/Os Pins Configuration* section).

CHAPTER

THREE

CLI STANDARD USAGE

The EBMM3 Command-Line-Interface is the official communication channel between the user and the module. The available commands are listed in this section. Detailed use cases are available in the application dedicated section.

3.1 Typical Parameters and Arguments

CLI arguments are either in their short or long version. The short version is only a letter and start with `-`. The long version is a full word and start with `--`. All short codes are listed below with their long description. All codes are case-sensitive:

short	long	SHORT	LONG
-a	–axis	-A	–address, –amplitude
-b		-B	–baudrate
-c	–channel	-C	–clear, –coeff, –crc, –connect
-d	–td, –disable, –data	-D	–disconnect, –distance, –decimator, –delay
-e	–excitation, –enable	-E	
-f	–format	-F	–fcutoff, –fs, –feedforward, –frequency
-g		-G	–gain
-h	–help	-H	
-i	–ti, –index, –interlaced	-I	
-j		-J	
-k		-K	
-l	–loglevel, –list, –length, –ledcurrent	-L	
-m		-M	
-n	–nsample	-N	–n
-o	–offset	-O	
-p	–startperiod, –kp, –phase	-P	–prbslength
-q		-Q	
-r	–register, –run, –radius	-R	–reload, –reset
-s	–sensoralgo, –softstart, –step	-S	–seed, –source, –shape, –sync
-t	–threshold, –type, –thermalcomp	-T	–starttime, –trig, –time
-u	–sensorunit, –unit	-U	–update
-v	–value	-V	
-w	–waveform, –write	-W	
-x	–amplitudex, –alpha	-X	
-y	–amplitudey, –beta	-Y	
-z	–gamma	-Z	

3.1.1 –axis

The axis value could be x, y, 0 (=x), 1 (=y) or 2 (both). The default value is either 2 for commands that accept to set both axis at the same time or 0 for other commands.

3.1.2 –address

A memory address could either be in decimal format or hexadecimal if preceed by `0x`

3.1.3 –disable –enable

Those flags do not accept any values and can obviously not be set at the same time.

3.1.4 –delay

The delay is expressed in controller interuption loop unit. A negative delay act prior to the actual loop. For exemple, specify `--delay=1` will perform the specified action the next interuption loop. On opposite, specify `--delay=-1` will perform the specified action on the previous interuption loop (if possible).

3.1.5 –format

Most commands can display or get information in multiple format. If not specified, the default format of the command is used.

name	read	write	description
`short`	x		short text value
`long`	x		long text value
`csv`	x		comma-separated values
`json`	x	x	json formatted text
`raw`	x	x	binary data as byte array
`hex`	x	x	binary data as hex string

Those formats offer various advantages in term of visual aspect or interface communication. When speed is an important parameter, a large set of data is preferably sent with `raw` format.

3.1.6 –help

This flag is available on each module and provide information on the available commands and how to use them.

3.2 System Information

The system module is used control the system settings, such as the firmware version, hardware variant and much more. An overview of this command is presented below.

3.2.1 system module

Manage system configuration.

Usage:

```
system [-R][-h][--format=<FORMAT>]
```

-R, --reset	Reset feature
-h, --help	Show help for this command
-f <FORMAT>, --format=<FORMAT>	Display/Print format [raw | short | long | json | hex | csv]

hardware command

Display information related to this electronic

Usage:

```
system hardware [-h][--format=<FORMAT>]
```

-h, --help	Show help for this command
-f <FORMAT>, --format=<FORMAT>	Display/Print format [raw | short | long | json | hex | csv]

firmware command

Display information about the current firmware

Usage:

```
system firmware [-U][-h][--format=<FORMAT>]
```

-U, --update	Start update process
-h, --help	Show help for this command
-f <FORMAT>, --format=<FORMAT>	Display/Print format [raw | short | long | json | hex | csv]

mems command

Display information about the connected mems (if any)

Usage:

```
system mems [-R][-h][-CD --format=<FORMAT>]
```

-R, --reload	Reload the MEMS data from STM32 memory
-h, --help	Show help for this command
-C, --connect	Connect new MEMS
-D, --disconnect	Disconnect MEMS

-f <FORMAT>, --format=<FORMAT> Display/Print format [raw | short | long | json | hex | csv]

vps command

Display/Change the variable power supply

Usage:

```
system vps [-h][<VOLTAGE> --format=<FORMAT>]
```

<VOLTAGE>
Maximum voltage to set on the power driver stage [Float]

Restricted to SuperUser (see *CLI Advanced Usage*)

-h, --help Show help for this command

-f <FORMAT>, --format=<FORMAT> Display/Print format [raw | short | long | json | hex | csv]

relay command

Display/Change the status of the driver outputs

Usage:

```
system relay [-h][<STATE> --format=<FORMAT>]
```

<STATE>
State to be set [off | on]

-h, --help Show help for this command

-f <FORMAT>, --format=<FORMAT> Display/Print format [raw | short | long | json | hex | csv]

autoboot command

Display/Change the autoboot process that enable to perform custom operation at each reset.

Usage:

```
system autoboot [-h][-edrwC --format=<FORMAT>]
```

-h, --help Show help for this command

-e, --enable Enable feature

-d, --disable Disable feature

-r, --run Execute autoboot script

-w, --write Write

-C, --clear Clear feature

-f <FORMAT>, --format=<FORMAT> Display/Print format [raw | short | long | json | hex | csv]

user_setup command

Display/Change user parameters like incidence angle of the laser and distance to target

Usage:

```
system user_setup [-h][--alpha=<ALPHA> --beta=<BETA> --gamma=<GAMMA>
--distance=<DISTANCE> --format=<FORMAT>]
```

-h, --help Show help for this command

-x <ALPHA>, --alpha=<ALPHA> Incidence angle around x axis [Float]

-y <BETA>, --beta=<BETA> Incidence angle around y axis [Float]

-z <GAMMA>, --gamma=<GAMMA> Rotation angle of the setup regarding the target [Float]

-D <DISTANCE>, --distance=<DISTANCE> Distance between the MEMS and the target [Float]

-f <FORMAT>, --format=<FORMAT> Display/Print format [raw | short | long | json | hex | csv]

interface command

Display/Change the UART interface settings

Usage:

```
system interface [-h][--loglevel=<LOGLEVEL> --baudrate=<BAUDRATE> --format=<FORMAT>]
```

-h, --help Show help for this command

-l <LOGLEVEL>, --loglevel=<LOGLEVEL> Logging level (0 for minimal and 5 for maximal logging) [Integer]

-B <BAUDRATE>, --baudrate=<BAUDRATE> UART bus baudrate (default: 115200) [Unsigned]

-f <FORMAT>, --format=<FORMAT> Display/Print format [raw | short | long | json | hex | csv]

3.2.2 Examples

The following example prints the hardware version and serial number. The available options can be used to filter the output message.

```
$ system hardware
System hardware version:  H61-127-01, H61-128-01
System serial number: 24127104
System variable power supply: 5.00-15.00V
```

The following example prints the firmware revision and generic informations. The available options can be used to filter the output message.

```
$ system firmware
Firmware version: 2.0.0-dev00
```

The following example prints the MEMS specifications.

```
$ system mems -f json
{
  "PN":       "MM2536-2-AU",
```

(continues on next page)

(continued from previous page)

```
  "SN":       "2024-12-01258",
  "Actuator": {
    "X":      {
      "MaxAngle":     5,
      "Resistance":   10.024,
      "DCGains":      [30.99, 0, 0.0048],
      "Resonance":    383.6496,
      "DampingRatio": 0.004272461
    },
    "Y":      {
      "MaxAngle":     5,
      "Resistance":   8.72,
      "DCGains":      [26.26, 0, 0.0045],
      "Resonance":    232.0560455,
      "DampingRatio": 0.005340576
    }
  }
}
```

If no MEMS is detected, a warning message is displayed:

```
$ system mems
warning: no mems calibration source found. Please take the following data validity with␣
↪special care...
```

3.2.3 System Startup

The autoboot submodule allows to manage start-up script. Piece of code can be saved into non-volatile memory and are executed automatically at each start-up. The autoboot feature can be enabled/disabled using the option `--enable`/`--disable`. The script can also be cleared using the option `--clear`.

To avoid to re-initialise the device as an analog controller with the UART at each reset, the following example initialise the analog input, set the analog output to measured current and start the control strategy at start-up:

```
$ system autoboot --enable --write
ENTER LINES BATCH MODE
signal input --source=analog
extension configure analog current
control strategy on

autoboot status: enabled
autoboot script:
  -signal input --source=analog
  -extension configure analog current
  -control strategy on
```

3.3 Define the Reference Signals

The signal module allows to generate and upload custom patterns.

- The user can select customisable predefined patterns like a digital function generator [DEFAULT].
- The user can upload a customisable predefined patterns to a buffer.
- The user can upload a custom user pattern to memory.
- the user can supply a pattern using the analog input interface

General concepts about this module are detailed below:

- This command only defined the signal source and pattern. The user must then select the control strategy to move the deflection unit. (see section *MEMS Controller*)
- The amplitude number is related to the unit argument and is the peak amplitude of the signal to be generated.
- The frequency is in Hz.
- The phase in degrees is relative to the time variable.
- A soft start parameters allows a smooth transcient start-up and is defined in [sec] until the target value is achieved
- The unit parameter is related to the amplitude value. The available units are listed below. Default units is in degrees and cannot be changed unless setting the super user mode (*CLI Advanced Usage*)

unit	Command	Min,Max values	Description
MEMS tilt angle	deg	-amax,amax	MEMS mechanical tilt in degree
MEMS voltage	volt	-vpp,vpp	Voltage driven on the MEMS
Internal unit (dsp unit)	dsp	-32767,32767	microcontroller register unit (reserved for advanced use)

- vpp is the voltage of the driver. More details are available in the dedicated section *System Information*.
- amax is the maximal mems tilt angle (mechanical deflection in degrees) stored in its memory. More details are available in the dedicated section *mems command*.

3.3.1 Buffer vs Waveform

Both the buffer and waveform source can be used to generate typical AWG function like sine, triangle, square waves or dc. The main difference is how the signal is generated:

buffer

the waveform is pre-computed and filled in the internal buffer

Advantage: Easier to synchronise the x and y axes, the pattern is predictable and can typically run faster Disadvantage: Frequency precision is limited by controller frequency

waveform

the waveform is generated at each controller step.

Advantage: More flexible and it is possible to update the parameters on the run. Disadvantage: Trigger on period is less precise (could happen between two controller steps)

The waveform pattern generation is used by default and is recommendended by Sercalo for most user case.

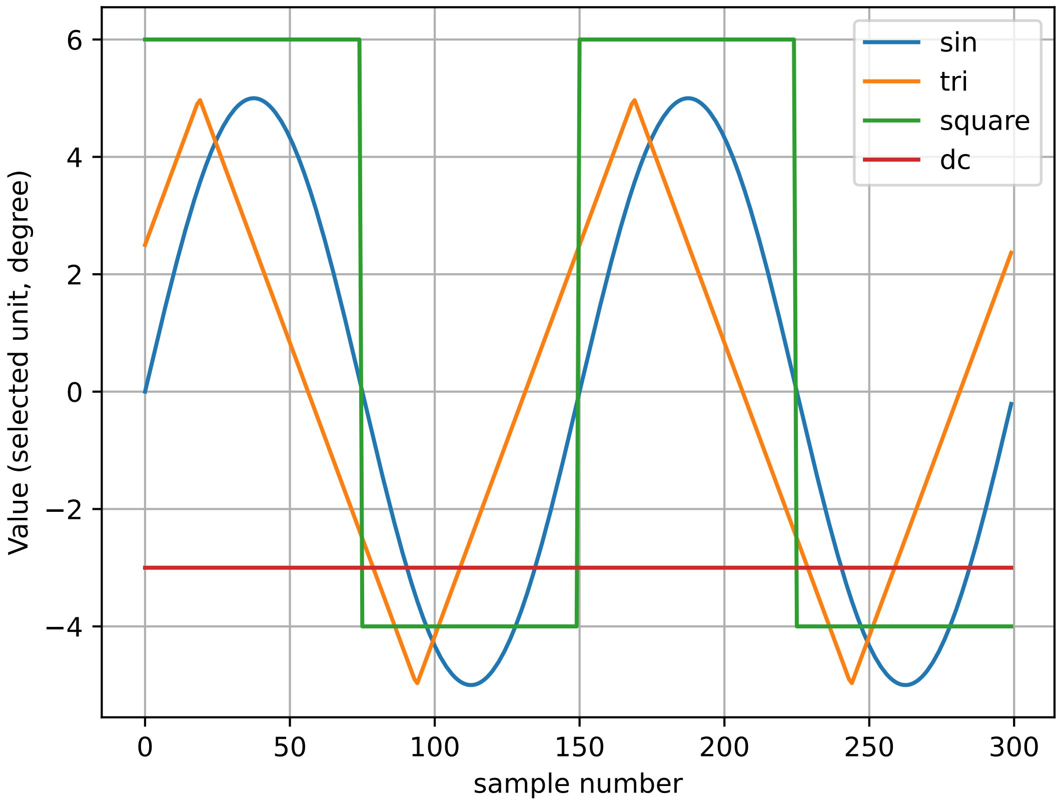
3.3.2 1D pattern generation

The `generate` subcommand allows to generate an unidimmensional reference signal. Available predefined shapes are listed below.

Shape	Command	Description
Sinewave	sine	Standard sinewave signal
Trianglewave	tri	Standard trianglewave signal
Squarewave	square	Standard squarewave
DC	dc	Constant DC signal
Sawtooth	sawtooth	Sawtooth signal
Staircase	stair	Step increasing signal

Some examples are illustrated below.

```
$ signal generate -a x -w sine -A 5.0 -F 150.0 -o 0 -p 0
$ signal generate -a x -w tri -A 5.0 -F 150.0 -o 0 -p 45
$ signal generate -a x -w square -A 5.0 -F 150.0 -o 1 -p 0
$ signal generate -a x -w dc -o -3
```



3.3.3 Soft transition

The `transition` subcommand allows to start the controller or change the waveform in a smooth way. One can control the time in seconds and shape parameter of this transition. Available shape are listed below. By default a spline curve is performed because it is the most efficient way to change from one position to another. The default time is $2\pi/F_{res}$ where F_{res} is the resonance frequency of each axis.

Shape	Command	Description
Spline Curve	spline	Smooth spline curve from start to end position (default)
Linear Curve	linear	Linear curve from start to end position

Linear curve is useful if one need to control the transition accurately. For exemple, a spiral shape could be drawn by creating a circular pattern with a linear transition:

```
$ signal generate -a x -w sine -A 1 -F 20.0 -o 0 -p 0
$ signal generate -a y -w sine -A 1 -F 20.0 -o 0 -p 90
$ signal transition -T 1 -S linear
$ control strategy on
... wait ...
$ signal generate -A 0
```

In this example, a 20Hz circle of 1° is defined with a linear 1s transition. When starting the controller, the circle will increase linearly and therefore create a spiral shape of 20 turn over 1 second. After a certain amount of time the user is executing a command to set the amplitude to zero which make the circle to return to its initial position and create a decreasing spiral.

3.3.4 User defined reference pattern (Buffer)

The `raw_data` subcommand allows to upload a custom reference pattern from a file.

Warning: This subcommand is not documented yet. Stay tuned!

3.3.5 Analog Input

The signal could also be referenced from an external analog source with the command `signal input --source=analog`. As previously mention, this command only defined the signal source. The pattern is defined by the analog input and the user must select the control strategy to move the deflection unit. (see section *MEMS Controller*), typically `control strategy feedforward`. The voltage input V_{in} is converted to an angular position a in degree. Depending on the device you are using, the voltage convertion is different. Indeed, the interface board is working from -5 to + 5V and the driver board from 0 to +2.5V.

Driver Board	Interface Board
$V_{in} = a/a_{max} + 1.25$	$V_{in} = 4 \cdot a/a_{max}$

Tip: a_{max} is the maximum tilt angle of each axis and can be checked using the `system mems` command.

3.3.6 Enable/Disable Signal

The signal output can be turn on and off with flag `--enable` and `--disable`. When the signal is disabled, the controller is using the last computed value as target. If the control strategy is `off` the last target is always 0°.

This feature is useful when one need to stop the MEMS at a specific target on the run, or to be sure to start the desired pattern at a known position.

Note: Do not mistake `control strategy on/off` and `signal --enable/disable`! The control acts on the driver loop and manage all the module, including the computation of the voltage required to reach the target angle. When off, all module are on standby mode. The signal acts on the target angle. When disabled, the target is not computed anymore and the last target is used. If control is on and signal is disabled, all modules other than signal are running (i.e. sensor, record, extension, etc.).

3.3.7 signal module

Build reference signal.

Usage:

```
signal [-h][-ed --format=<FORMAT>]
```

-h, --help Show help for this command

-e, --enable Enable feature

-d, --disable Disable feature

-f <FORMAT>, --format=<FORMAT> Display/Print format [raw | short | long | json | hex | csv]

input command

Display/Change the source of the reference pattern.

Usage:

```
signal input [-h][-ed --source=<SOURCE> --axis=<AXIS> --format=<FORMAT>]
```

-h, --help Show help for this command

-e, --enable Enable feature

-d, --disable Disable feature

-S <SOURCE>, --source=<SOURCE> Input source to generate signal [buffer | waveform | analog]

-a <AXIS>, --axis=<AXIS> Axis selection [x | y | xy]

-f <FORMAT>, --format=<FORMAT> Display/Print format [raw | short | long | json | hex | csv]

generate command

Change the reference pattern used in buffer or waveform mode by predefined pattern.

Usage:

```
signal generate [-h][-Sed --axis=<AXIS> --waveform=<WAVEFORM> --amplitude=<AMPLITUDE>
--frequency=<FREQUENCY> --offset=<OFFSET> --phase=<PHASE> --n=<N> --radius=<RADIUS>
--softstart=<SOFTSTART> --unit=<UNIT> --format=<FORMAT>]
```

-h, --help Show help for this command

-S, --sync Synchronise axes phase

-e, --enable Enable feature

-d, --disable Disable feature

-a <AXIS>, --axis=<AXIS> Axis selection [x | y | xy]

-w <WAVEFORM>, --waveform=<WAVEFORM> Waveform of the generated signal [dc | sine | tri | square | sawtooth | stair]

-A <AMPLITUDE>, --amplitude=<AMPLITUDE> Peak amplitude of the generated signal [Float]

-F <FREQUENCY>, --frequency=<FREQUENCY> Frequency of the generated signal [Hz] [Float]

-o <OFFSET>, --offset=<OFFSET> Amplitude offset of the generated signal [Float]

-p <PHASE>, --phase=<PHASE> Phase offset of the generated signal [°] [Float]

-N <N>, --n=<N> Number of staircase per period [Unsigned]

-r <RADIUS>, --radius=<RADIUS> Staircase transition smoothing factor (from 0 to 1) [Float]

-s <SOFTSTART>, --softstart=<SOFTSTART> Softstart rising time in seconds [Float]

Restricted to SuperUser (see *CLI Advanced Usage*)

-u <UNIT>, --unit=<UNIT> Unit selection [*dsp* | *volt* | deg]

-f <FORMAT>, --format=<FORMAT> Display/Print format [raw | short | long | json | hex | csv]

raw_data command

Change the reference pattern used in buffer mode with a custom pattern.

Usage:

```
signal raw_data --axis=<AXIS> [-h][-ed --unit=<UNIT> --address=<ADDRESS>
--length=<LENGTH> --crc=<CRC> --data=<DATA> --softstart=<SOFTSTART> --format=<FORMAT>]
```

-h, --help Show help for this command

-e, --enable Enable feature

-d, --disable Disable feature

-a <AXIS>, --axis=<AXIS> Axis selection [x | y | xy]

-u <UNIT>, --unit=<UNIT> Unit selection [*dsp* | *volt* | deg]

-A <ADDRESS>, --address=<ADDRESS> Starting address [Unsigned]

-l <LENGTH>, --length=<LENGTH> Length of the data [Unsigned]

-C <CRC>, --crc=<CRC> Cyclic redundancy check value. No CRC check if omitted. [Unsigned]

-d <DATA>, --data=<DATA> Data to be flashed. System will enter into batch transfer mode without this parameter. [String]

-s <SOFTSTART>, --softstart=<SOFTSTART> Softstart rising time in seconds [Float]

Restricted to SuperUser (see *CLI Advanced Usage*)

-f <FORMAT>, --format=<FORMAT> Display/Print format [raw | short | long | json | hex | csv]

transition command

Change how the reference pattern is updated when changing signal.

Usage:

```
signal transition [-h][-ed --axis=<AXIS> --time=<TIME> --shape=<SHAPE> --format=<FORMAT>]
```

-h, --help Show help for this command

-e, --enable Enable feature

-d, --disable Disable feature

-a <AXIS>, --axis=<AXIS> Axis selection [x | y | xy]

-T <TIME>, --time=<TIME> Softstart rising time in seconds [Float]

-S <SHAPE>, --shape=<SHAPE> Shape of the transit function [spline | linear | *step*]

-f <FORMAT>, --format=<FORMAT> Display/Print format [raw | short | long | json | hex | csv]

3.4 MEMS Controller

The control module allows to manage the controller settings and control algorithm to be used.

3.4.1 Select the control strategy

The `strategy` subcommand selects the control strategy. The control strategy defines the algorithm used to drive the deflection unit from the reference pattern. Available control strategies are listed in the table below. Generally it is recommended to use `control strategy on` and `control strategy off` to keep things simple.

Strategy	Valid units	Description
off	all	Disconnect the coils from the controller (open-circuit)
on	all	Use the best strategy depending on input data and connected device
direct	volts	Output the reference signal directly (function generator mode)
feedforward	deg	Use feed forward algorithm
pid	deg	Use PID algorithm (feedback sensor is needed)
rst	deg	Use RST algorithm, for development only (feedback sensor is needed)

3.4.2 control module

Manage control informations

Usage:

```
control [-h][--fs=<FS> --gain=<GAIN> --format=<FORMAT>]
```

-h, --help Show help for this command

-F <FS>, --fs=<FS> Controller sampling frequency [Hz] [Float]

-G <GAIN>, --gain=<GAIN> Controller gain (keep low for better accuracy and high for faster response) [Float]

-f <FORMAT>, --format=<FORMAT> Display/Print format [raw | short | long | json | hex | csv]

strategy command

Display/Change the controller strategy

Usage:

```
control strategy [-h][<STRATEGY> --axis=<AXIS> --format=<FORMAT>]
```

<*STRATEGY*>
Control strategy to use [off | on | *direct* | feedforward | pid | *rst*]

-h, --help Show help for this command

-a <AXIS>, --axis=<AXIS> Axis selection [x | y | xy]

-f <FORMAT>, --format=<FORMAT> Display/Print format [raw | short | long | json | hex | csv]

feedforwardconfig command

Display/Change the feedforward parameters

Restricted to SuperUser (see *CLI Advanced Usage*)

Usage:

```
control feedforwardconfig [-h][--fcutoff=<FCUTOFF> --format=<FORMAT>]
```

-h, --help Show help for this command

-F <FCUTOFF>, --fcutoff=<FCUTOFF> Cut-off frequency [Hz] [Float]

-f <FORMAT>, --format=<FORMAT> Display/Print format [raw | short | long | json | hex | csv]

pidconfig command

Display/Change the PID parameters

Restricted to SuperUser (see *CLI Advanced Usage*)

Usage:

```
control pidconfig [-h][-F --axis=<AXIS> --kp=<KP> --ti=<TI> --td=<TD>
--decimator=<DECIMATOR> --format=<FORMAT>]
```

-h, --help Show help for this command

-F, --feedforward Enable parallel feed forward filter

-a <AXIS>, --axis=<AXIS> Axis selection [x | y | xy]

-p <KP>, --kp=<KP> Proportional gain [Float]

-i <TI>, --ti=<TI> Integral gain [Float]

-d <TD>, --td=<TD> Differential gain [Float]

-D <DECIMATOR>, --decimator=<DECIMATOR> Decimator value (frec = fsample/decimator) (default: 1) [Unsigned]

-f <FORMAT>, --format=<FORMAT> Display/Print format [raw | short | long | json | hex | csv]

rstconfig command

Display/Change the RST parameters

Restricted to SuperUser (see *CLI Advanced Usage*)

Usage:

```
control rstconfig [-h][-C --axis=<AXIS> --format=<FORMAT>]
```

-h, --help Show help for this command

-C, --coeff Update RST Coefficient

-a <AXIS>, --axis=<AXIS> Axis selection [x | y | xy]

-f <FORMAT>, --format=<FORMAT> Display/Print format [raw | short | long | json | hex | csv]

3.4.3 Advanced Usage: PID controller

The controller can be tuned using the above command. A typical application example is given below (the coefficients of this example are not suitable for any MEMS, just to illustrate the syntax:

```
$ control pidctl -a x -p 1.2 -i 2.2 -d 3.3
```

The PID formula implemented in the controller is given below, assuming that k is a actual sample number and e the difference between the reference value and the sensor value:

$$K_p \cdot e[k] + T_d \cdot (e[k] - e[k-1]) + T_i \cdot \sum_{i=0}^{k} e[i]$$

One should be careful that the coefficients signification can differ from one design tool to another.

A widely used formula in matlab is given below.

$$K_p \cdot (e[k] + \frac{T_d}{h} \cdot (e[k] - e[k-1]) + h \cdot Ti \cdot \sum_{i=0}^{k} e[i])$$

The conversion formula is given below:

Parameter	Matlab	EBMM3
Kp	Kp	Kp
Ti	Ti	$h \cdot K_p \cdot T_i$
Td	Td	$\frac{K_p \cdot T_d}{h}$

A feed-forward filter can be added to the PID loop using the `-f` option.

3.5 User I/Os Pins Configuration

The extension module allows to manage configurable digital and analog pins. The user can configure those pins to synchronise the EBMM3 with external system.

The `configure` subcommand is used to select the desired extension pin and functionality. The `reset` subcommand resets all the extension pins to high impedance state.

The available digital functions are detailed in the table below. Those can be selected with pin `dig1`, `dig2`, `dig3`, `dig4` and `dig5`.

Generally the EBMM3 can output the state of a module (sync) or start/stop it with an input (trig).

func	type	description	details
hiz	OUTPUT	Set pin to high impedance	The pin is not used and set to high impedance state (default)
low	OUTPUT	Output low state	The pin is set to output and tied to GND level.
high	OUTPUT	Output high state	The pin is set to output and tied to 3.3V level.
syncx	OUTPUT	Pulse at start of x period	A pulse is generated at the beginning of each period of x axis reference signal
syncy	OUTPUT	Pulse at start of y period	A pulse is generated at the beginning of each period of y axis reference signal
synclaser	OUTPUT	Same output as Laser pin	Copy the output of the laser pin
synccontrol	OUTPUT	High when the controller is running	Indicates if the controller is off (GND) or if it's running well (3.3V).
syncclock	OUTPUT	Pulse at every new sampling computation	A pulse is generated at the beginning of the controller routine.
syncrecord	OUTPUT	High when recording is in progress	Indicates if the record acquisition is active
syncerror	OUTPUT	High when an error happens	Copy the output of the status pin 1
triglaser	INPUT	Trigger to start/stop the laser output	Copy the input value to the laser pin
trigcontrol	INPUT	Trigger to start/stop the controller	Start (3.3V) or Stop (GND) the controller.
trigrecord	INPUT	Trigger to start/stop the recorder	Start (3.3V) or Stop (GND) the recorder.
trigerror	INPUT	Trigger a user error to stop the MEMS	Generate a user failure

Tip: A sync/trig function can be applied only to a single pin. If the same function is applied to a new pin, the previous one is set to hiz and only the new pin is assigned.

The available analog functions are detailed below. Those can be selected with pin `an1` or `an2`. Alternatively both pin can be selected by specifing pin `analog`. Every function (except hiz) can be used either as described to select the best channel (x or y) to the corresponding output or by specifing exactly with channel to use. For exemple `refx` will get the output reference signal of x axis. `ref` on the other hand will get the reference signal of x if `an1` is selected, y if `an2` is selected and both channel if `analog` is used.

Sercalo recommend to use the pin `analog` with corresponding function to directly specify all channels to the corresponding output to simplify operation.

func	description	details
hiz	Set pin to high impedance	The pin is not used and set to high impedance state (default)
ref	Output reference signal	The reference signals are streamed to the output pins
pos	Sensor position	The MEMS positions measured from sensor are streamed on the output pins
drive	Output driver values	The voltages injected into the MEMS are streamed on the output pins
current	Measured Current	Measured currents are streamed on the output pins

Convertion from output voltage V_{out} to user units is done through the following equations. Be careful the interface board is working from -5 to + 5V and the driver board is from 0 to +2.5V. Depending on which device you are using, you may use a different equation.

Unit	usage	Driver Board	Interface Board
Angle [°]	refx/y, posx/y	$(V_{out} - 1.25) \cdot a_{max}$	$V_{out}/4 \cdot a_{max}$
Voltage [V]	refx/y, drivex/y	$(V_{out} - 1.25) \cdot V_{pp}$	$V_{out}/4 \cdot V_{pp}$
Current [A]	currentx/y	$(V_{out} - 1.25) \cdot 0.4A$	$V_{out} \cdot 0.1A$

Tip: Vpp value is 5V by default and can be checked using the `system vps` command. a_{max} is the maximum tilt angle of each axis and can be checked using the `system mems` command.

3.5.1 extension module

Configure extension pins

Usage:

```
extension [-R][-h][--format=<FORMAT>]
```

-R, --reset Reset feature

-h, --help Show help for this command

-f <FORMAT>, --format=<FORMAT> Display/Print format [raw | short | long | json | hex | csv]

configure command

Configure an I/O pin with a specfic function

Usage:

```
extension configure <PINID> <FUNC> [-h][--format=<FORMAT>]
```

<PINID>
Pin ID to configure [dig1 | dig2 | dig3 | dig4 | dig5 | an1 | an2 | analog]

<FUNC>
Pin feature to enable [FUNC]

-h, --help Show help for this command

-f <FORMAT>, --format=<FORMAT> Display/Print format [raw | short | long | json | hex | csv]

3.6 External Laser

The laser function is used to configure the laser output.

3.6.1 laser module

Manage laser output

Usage:

```
laser [-h][--format=<FORMAT>]
```

-h, --help Show help for this command

-f <FORMAT>, --format=<FORMAT> Display/Print format [raw | short | long | json | hex | csv]

on command

Switch on the laser

Usage:

```
laser on [-h][-S --delay=<DELAY> --format=<FORMAT>]
```

-h, --help Show help for this command

-S, --sync Synchronise axes phase

-D <DELAY>, --delay=<DELAY> Delay in milliseconds [Float]

-f <FORMAT>, --format=<FORMAT> Display/Print format [raw | short | long | json | hex | csv]

off command

Switch off the laser

Usage:

```
laser off [-h][--format=<FORMAT>]
```

-h, --help Show help for this command

-f <FORMAT>, --format=<FORMAT> Display/Print format [raw | short | long | json | hex | csv]

3.7 Recorder

The record module is used to record the runtime values. The choice of what to record could be selected from the `record register` command. A register shall be assign to a channel (up to 8 available). The memory size is automatically adjusted in function of the register and the number of channel. The available memory could be shown with `record` command.

A typical recorder setup to record the first 100 value of signal reference and sensor position with a trigger on the third period of X-channel could be set with:

```
$ record channel --enable --register=period_x
$ record channel --enable --register=signal_ref
$ record channel --enable --register=sensor_pos
$ record trigger --enable --channel=1 --type=GE --value=3
$ record acq single
# WAIT to complete before printing the record. Record status could be checked with
↪'record acq'
$ record print --format=csv
```

Important: Values are recorded only if the controller is running (see *MEMS Controller* section).

3.7.1 record module

Configure and perform recording.

Usage:

```
record [-h][--format=<FORMAT>]
```

-h, --help Show help for this command

-f <FORMAT>, --format=<FORMAT> Display/Print format [raw | short | long | json | hex | csv]

channel command

Set the channel to use for the record

Usage:

```
record channel [-h][-edl --channel=<CHANNEL> --register=<REGISTER> --format=<FORMAT>]
```

-h, --help Show help for this command

-e, --enable Enable feature

-d, --disable Disable feature

-l, --list List available features

-c <CHANNEL>, --channel=<CHANNEL> ID of the channel to use [CHANNEL]

-r <REGISTER>, --register=<REGISTER> ID of the register to use [REGISTER]

-f <FORMAT>, --format=<FORMAT> Display/Print format [raw | short | long | json | hex | csv]

register command

List the available register that can be used for the record

Usage:

```
record register [-h][-l --format=<FORMAT>]
```

-h, --help Show help for this command

-l, --list List available features

-f <FORMAT>, --format=<FORMAT> Display/Print format [raw | short | long | json | hex | csv]

trigger command

Set the trigger to use for the record

Usage:

```
record trigger [-h][-edl --channel=<CHANNEL> --index=<INDEX> --type=<TYPE>
--value=<VALUE> --delay=<DELAY> --format=<FORMAT>]
```

-h, --help Show help for this command

-e, --enable Enable feature

-d, --disable Disable feature

-l, --list List available features

-c <CHANNEL>, --channel=<CHANNEL> ID of the channel to use [CHANNEL]

-i <INDEX>, --index=<INDEX> Index of the array (for multi-index channel) [Unsigned]

-t <TYPE>, --type=<TYPE> ID of the trigger to use [TYPE]

-v <VALUE>, --value=<VALUE> Threshold value to use for the trigger [Float]

-D <DELAY>, --delay=<DELAY> Delay in sample (negative for pre-trigger) [Integer]

-f <FORMAT>, --format=<FORMAT> Display/Print format [raw | short | long | json | hex | csv]

acq command

Control the recording state

Usage:

```
record acq [-h][<ACQ> --nsample=<NSAMPLE> --decimator=<DECIMATOR> --format=<FORMAT>]
```

<ACQ>
Acquisition command [Stop | Single | Continuous]

-h, --help Show help for this command

-n <NSAMPLE>, --nsample=<NSAMPLE> Number of sample to record [Unsigned]

-D <DECIMATOR>, --decimator=<DECIMATOR> Decimator value (frec = fsample/decimator) (default: 1) [Unsigned]

-f <FORMAT>, --format=<FORMAT> Display/Print format [raw | short | long | json | hex | csv]

print command

Print the recorded data

Usage:

```
record print [-h][--channel=<CHANNEL> --nsample=<NSAMPLE> --format=<FORMAT>]
```

-h, --help Show help for this command

-c <CHANNEL>, --channel=<CHANNEL> ID of the channel to use [CHANNEL]

-n <NSAMPLE>, --nsample=<NSAMPLE> Number of sample to record [Unsigned]

-f <FORMAT>, --format=<FORMAT> Display/Print format [raw | short | long | json | hex | csv]

3.7.2 Register options

The available register options are listed below.

Value	Type	Array Size	Description
index	UInt32	1	Index of the record (from acquisition start)
ext1	UInt8	1	Extension pin 1 value
ext2	UInt8	1	Extension pin 2 value
ext3	UInt8	1	Extension pin 3 value
ext4	UInt8	1	Extension pin 4 value
ext5	UInt8	1	Extension pin 5 value
signal_ref	Float32	2	Reference signal value (in user unit)
signal_ref_raw	Int16	2	Reference signal value (in dsp unit)
period_x	UInt32	1	X-axis signal period index (from controller start)
period_y	UInt32	1	Y-axis signal period index (from controller start)
driver_coil	Float32	2	Coil input voltage [V]
driver_dac	UInt16	2	Coil input DAC [dsp]
current	Float32	2	Coil measured current [A]
current_rms	Float32	2	Coil measured RMS current [A]
sensor_adc	Int16	4	Sensor output ADC [dsp]
sensor_volt	Float32	4	Sensor output voltage [V]
sensor_norm	Float32	4	Sensor output norm [V/V]
sensor_pos	Float32	2	Sensor output position [°]
adc_i1	Float32	1	X-axis Coil input current [A]
adc_i2	Float32	1	Y-axis Coil input current [A]
x_in	Int16	1	X-axis ADC input [mV]
y_in	Int16	1	Y-axis ADC input [mV]
adc_vps	Int16	1	Power Supply ADC input [mV]
adc_vps	Float32	1	Power Supply ADC input [V]

3.7.3 Trigger options

The available trigger options are listed below.

Value	Description
IMD	Start recording immediately
RE	Start on next rising edge
FE	Start on next falling edge
REL	Start on next logic rising edge (i.e. 0 to 1)
FEL	Start on next logic falling edge (i.e. 1 to 0)
GT	Start when record is greater than a value
LT	Start when record is lower than a value
GE	Start when record is greater than or equal to a value
LE	Start when record is lower than or equal to a value

Important:

- Terminator characters are LF (Line feed, `\n`, `0x0A`) or CR+LF (carriage return and line feed, `\r\n`, `0x0D 0x0A`).

- Replies always end with a prompt symbol (`$`, or `#` in superuser mode), which indicates the system is ready to accept new commands.
- Commands, parameters, and arguments are separated by one or more spaces (ASCII `0x20`)
- Parameters are indicated by `-` followed by a letter or `--` followed by the name of the parameter.
- The CLI is divided into modules which can be listed with the command help.
- Each module function can be listed with the command `<module> --help`, with `<module>` the name of the module.
- Error messages are displayed in the console and start with the label `error` (see *Error Management*).

Tip:

- Always wait for the prompt symbol `$` (or `#` is superuser mode) before sending a new command.

CHAPTER

FOUR

CLI ADVANCED USAGE

Advanced CLI usage are mainly authorized with the `sudo` command.

Usage:

`sudo enter`
Enter super user mode

`sudo quit`
Quit super user mode

Important: Operations under sudo mode are resticted to advanced user. Some commands could lead to uncontrolled behavior of the connected device and potential non-reversible failures. If one need to perform such advanced feature, it is recommended to leave the sudo mode with command `sudo quit` as soon as the requested operations are completed.

4.1 Failure Prevention

The failsafe module allows to manage the system protections.

Caution: For a normal use of the system, it is recommended to leave the default parameters.

4.1.1 Failsafe sources

The available failsafe protection sources are listed below:

Name	Description	Settable Threshold Value
current-x/y	Measured current in the MEMS coils	Yes [A]
position-x/y	Measured tilt angle of the MEMS (required Feedback Sensor)	Yes [°]
temperature	Measured MEMS temperature (required Feedback Sensor)	Yes [°C]
overrun	Controller Frequency Overflow	No
sensor-saturation	Feedback ADC saturation	Yes (ADC value, [0-32768])
user	User generated Error (from IO pin or CLI)	No

Each source can be enabled using the `--enable` option or disabled using the `--disable` option. When available, the `threshold` option can be used to set the maximal admissible value of the failsafe source.

The `default` subcommand or the `--enable` option without command name resets all the failsafe parameters to default values. The `disableall` subcommand or the `--disable` option without command name disable all the failsafe parameters.

User can generate a virtual error with command `failsafe --trig`. The EBMM3 will enter in safe mode and stop the controller.

When a failsafe error is generated, the first status pin is enabled and the only way to get out of it is to perform a reset or call the `failsafe --reset` command.

4.1.2 failsafe module

Manage failsafe configuration and status.

Usage:

```
failsafe [-T][-h][-R][<FLAG> -ed --threshold=<THRESHOLD> --format=<FORMAT>]
```

<FLAG>
Failsafe flag ID [position-x | position-y | current-x | current-y | overrun | sensor-saturation | temperature | user]

Restricted to SuperUser (see *CLI Advanced Usage*)

-T, --trig Trig a user generated error

-h, --help Show help for this command

-R, --reset Reset feature

-e, --enable Enable feature

-d, --disable Disable feature

-t <THRESHOLD>, --threshold=<THRESHOLD> Threshold value [Float]

Restricted to SuperUser (see *CLI Advanced Usage*)

-f <FORMAT>, --format=<FORMAT> Display/Print format [raw | short | long | json | hex | csv]

angle command

Set failsafe on mems angle position

Restricted to SuperUser (see *CLI Advanced Usage*)

Usage:

```
failsafe angle [-h][-R][-ed --threshold=<THRESHOLD> --axis=<AXIS> --format=<FORMAT>]
```

-h, --help Show help for this command

-R, --reset Reset feature

-e, --enable Enable feature

-d, --disable Disable feature

-t <THRESHOLD>, --threshold=<THRESHOLD> Threshold value [Float]

-a <AXIS>, --axis=<AXIS> Axis selection [x | y | xy]

-f <FORMAT>, --format=<FORMAT> Display/Print format [raw | short | long | json | hex | csv]

current command

Set failsafe on coil maximum current

Restricted to SuperUser (see *CLI Advanced Usage*)

Usage:

```
failsafe current [-h][-R][-ed --threshold=<THRESHOLD> --axis=<AXIS> --format=<FORMAT>]
```

-h, --help Show help for this command

-R, --reset Reset feature

-e, --enable Enable feature

-d, --disable Disable feature

-t <THRESHOLD>, --threshold=<THRESHOLD> Threshold value [Float]

-a <AXIS>, --axis=<AXIS> Axis selection [x | y | xy]

-f <FORMAT>, --format=<FORMAT> Display/Print format [raw | short | long | json | hex | csv]

disableall command

Remove all failsafe behavior

Restricted to SuperUser (see *CLI Advanced Usage*)

Usage:

```
failsafe disableall [-h][-R][--format=<FORMAT>]
```

-h, --help Show help for this command

-R, --reset Reset feature

-f <FORMAT>, --format=<FORMAT> Display/Print format [raw | short | long | json | hex | csv]

4.2 Feedback Sensor

The sensor module allows to configure and get the current configuration of the MEMS sensor.

Hint: The calibration subcommand is automatically called at electronics start-up.

4.2.1 sensor module

Manage sensor configuration.

Usage:

```
sensor [-h][--format=<FORMAT>]
```

-h, --help Show help for this command

-f <FORMAT>, --format=<FORMAT> Display/Print format [raw | short | long | json | hex | csv]

calibration command

Perform a sensor calibration

Usage:

```
sensor calibration [-h][--format=<FORMAT>]
```

-h, --help Show help for this command

-f <FORMAT>, --format=<FORMAT> Display/Print format [raw | short | long | json | hex | csv]

configure command

Change sensor parameters

Restricted to SuperUser (see *CLI Advanced Usage*)

Usage:

```
sensor configure [-R][-h][--ledcurrent=<LEDCURRENT> --thermalcomp=<THERMALCOMP>
--sensoralgo=<SENSORALGO> --format=<FORMAT>]
```

-R, --reset Reset feature

-h, --help Show help for this command

-l <LEDCURRENT>, --ledcurrent=<LEDCURRENT> Current of the LED [mA] [Float]

-t <THERMALCOMP>, --thermalcomp=<THERMALCOMP> Temperature compensation algorithm to use [off | led_poly]

-s <SENSORALGO>, --sensoralgo=<SENSORALGO> Sensor algorithm to use [off | poly | lut]

-f <FORMAT>, --format=<FORMAT> Display/Print format [raw | short | long | json | hex | csv]

read command

Get the last sensor position

Usage:

```
sensor read [-h][--sensorunit=<SENSORUNIT> --format=<FORMAT>]
```

-h, --help Show help for this command

-u <SENSORUNIT>, --sensorunit=<SENSORUNIT> Sensor unit [raw | volt | norm | pos]

-f <FORMAT>, --format=<FORMAT> Display/Print format [raw | short | long | json | hex | csv]

4.3 Memory Management

The memory module allows to communicate with the different memories available on the EBMM3.

> **Attention:** This command is not made to be called by the end-user unless after a Sercalo team specific demand for maintenance purpose.

4.3.1 memory module

Manage memory informations.

Usage:

```
memory [-h][-l --format=<FORMAT>]
```

-h, --help Show help for this command

-l, --list List available features

-f <FORMAT>, --format=<FORMAT> Display/Print format [raw | short | long | json | hex | csv]

info command

Display information about the specified memory

Usage:

```
memory info <MEMORY> [-h][--format=<FORMAT>]
```

<*MEMORY*>
Memory ID [*1wire* | mems | *spi* | dbconf | ibconf | signal | *acq* | *sensor*]

-h, --help Show help for this command

-f <FORMAT>, --format=<FORMAT> Display/Print format [raw | short | long | json | hex | csv]

read command

Read and display the specified memory data

Usage:

```
memory read <MEMORY> [-h][--channel=<CHANNEL> --address=<ADDRESS> --length=<LENGTH>
--format=<FORMAT>]
```

<*MEMORY*>
Memory ID [*1wire* | mems | *spi* | dbconf | ibconf | signal | *acq* | *sensor*]

-h, --help Show help for this command

-c <CHANNEL>, --channel=<CHANNEL> Channel number (only for multi-channel memory) [Unsigned]

-A <ADDRESS>, --address=<ADDRESS> Starting address [Unsigned]

-l <LENGTH>, --length=<LENGTH> Length of the data [Unsigned]

-f <FORMAT>, --format=<FORMAT> Display/Print format [raw | short | long | json | hex | csv]

flash command

Write data to the specified memory

Restricted to SuperUser (see *CLI Advanced Usage*)

Usage:

```
memory flash <MEMORY> [-h][--channel=<CHANNEL> --address=<ADDRESS> --length=<LENGTH>
--crc=<CRC> --data=<DATA> --format=<FORMAT>]
```

<MEMORY>
Memory ID [*1wire* | mems | *spi* | dbconf | ibconf | signal | *acq* | *sensor*]

-h, --help Show help for this command

-c <CHANNEL>, --channel=<CHANNEL> Channel number (only for multi-channel memory) [Unsigned]

-A <ADDRESS>, --address=<ADDRESS> Starting address [Unsigned]

-l <LENGTH>, --length=<LENGTH> Length of the data [Unsigned]

-C <CRC>, --crc=<CRC> Cyclic redundancy check value. No CRC check if omitted. [Unsigned]

-d <DATA>, --data=<DATA> Data to be flashed. System will enter into batch transfer mode without this parameter. [String]

-f <FORMAT>, --format=<FORMAT> Display/Print format [raw | short | long | json | hex | csv]

erase command

Erase the data of the specified memory

Restricted to SuperUser (see *CLI Advanced Usage*)

Usage:

```
memory erase <MEMORY> [-h][--channel=<CHANNEL> --address=<ADDRESS> --length=<LENGTH>
--format=<FORMAT>]
```

<MEMORY>
Memory ID [*1wire* | mems | *spi* | dbconf | ibconf | signal | *acq* | *sensor*]

-h, --help Show help for this command

-c <CHANNEL>, --channel=<CHANNEL> Channel number (only for multi-channel memory) [Unsigned]

-A <ADDRESS>, --address=<ADDRESS> Starting address [Unsigned]

-l <LENGTH>, --length=<LENGTH> Length of the data [Unsigned]

-f <FORMAT>, --format=<FORMAT> Display/Print format [raw | short | long | json | hex | csv]

4.4 Characterisation Module

This characterisation module is used to perform different characterisation procedures on the MEMS and the electronics.

4.4.1 characterisation module

Perform mems characterisation procedure.

> **Warning:** This module is restricted to SuperUser (see *CLI Advanced Usage*)

Usage:

```
characterisation --axis=<AXIS> [--amplitude=<AMPLITUDE> --nsample=<NSAMPLE>
--sensorunit=<SENSORUNIT> --sensoralgo=<SENSORALGO> --offset=<OFFSET>
--decimator=<DECIMATOR>]
```

-a <AXIS>, --axis=<AXIS> Axis selection [x | y | xy]

-A <AMPLITUDE>, --amplitude=<AMPLITUDE> Peak amplitude of the generated signal [Float]

-n <NSAMPLE>, --nsample=<NSAMPLE> Number of sample to record [Unsigned]

-u <SENSORUNIT>, --sensorunit=<SENSORUNIT> Sensor unit [raw | volt | norm | pos]

-s <SENSORALGO>, --sensoralgo=<SENSORALGO> Sensor algorithm to use [off | poly | lut]

-o <OFFSET>, --offset=<OFFSET> Amplitude offset of the generated signal [Float]

-D <DECIMATOR>, --decimator=<DECIMATOR> Decimator value (frec = fsample/decimator) (default: 1) [Unsigned]

identification command

Use an excitation wavefrom to compute the MEMS parameters.

Usage:

```
characterisation identification --axis=<AXIS> --axis=<AXIS> [--excitation=<EXCITATION>
--amplitude=<AMPLITUDE> --nsample=<NSAMPLE> --sensorunit=<SENSORUNIT>
--sensoralgo=<SENSORALGO> --startperiod=<STARTPERIOD> --offset=<OFFSET>
--decimator=<DECIMATOR> --prbslength=<PRBSLENGTH> --seed=<SEED> --amplitude=<AMPLITUDE>
--nsample=<NSAMPLE> --sensorunit=<SENSORUNIT> --sensoralgo=<SENSORALGO> --offset=<OFFSET>
--decimator=<DECIMATOR>]
```

-a <AXIS>, --axis=<AXIS> Axis selection [x | y | xy]

-a <AXIS>, --axis=<AXIS> Axis selection [x | y | xy]

-e <EXCITATION>, --excitation=<EXCITATION> Excitation signal [prbs]

-A <AMPLITUDE>, --amplitude=<AMPLITUDE> Peak amplitude of the generated signal [Float]

-n <NSAMPLE>, --nsample=<NSAMPLE> Number of sample to record [Unsigned]

-u <SENSORUNIT>, --sensorunit=<SENSORUNIT> Sensor unit [raw | volt | norm | pos]

-s <SENSORALGO>, --sensoralgo=<SENSORALGO> Sensor algorithm to use [off | poly | lut]

-p <STARTPERIOD>, --startperiod=<STARTPERIOD> Begin record after a given number of signal period (useful to skip the transcient stage) [Integer]

-o <OFFSET>, --offset=<OFFSET> Amplitude offset of the generated signal [Float]

-D <DECIMATOR>, --decimator=<DECIMATOR> Decimator value (frec = fsample/decimator) (default: 1) [Unsigned]

-P <PRBSLENGTH>, --prbslength=<PRBSLENGTH> PRBS length for the PRBS generator (default: 12) [Unsigned]

-S <SEED>, --seed=<SEED> Seed for the random generator (default: 1) [Unsigned]

-A <AMPLITUDE>, --amplitude=<AMPLITUDE> Peak amplitude of the generated signal [Float]

-n <NSAMPLE>, --nsample=<NSAMPLE> Number of sample to record [Unsigned]

-u <SENSORUNIT>, --sensorunit=<SENSORUNIT> Sensor unit [raw | volt | norm | pos]

-s <SENSORALGO>, --sensoralgo=<SENSORALGO> Sensor algorithm to use [off | poly | lut]

-o <OFFSET>, --offset=<OFFSET> Amplitude offset of the generated signal [Float]

-D <DECIMATOR>, --decimator=<DECIMATOR> Decimator value (frec = fsample/decimator) (default: 1) [Unsigned]

step command

Perform a step response and measure the sensor.

Usage:

```
characterisation step --axis=<AXIS> --axis=<AXIS> [--amplitude=<AMPLITUDE>
--nsample=<NSAMPLE> --sensorunit=<SENSORUNIT> --sensoralgo=<SENSORALGO>
--offset=<OFFSET> --decimator=<DECIMATOR> --amplitude=<AMPLITUDE> --nsample=<NSAMPLE>
--sensorunit=<SENSORUNIT> --sensoralgo=<SENSORALGO> --offset=<OFFSET>
--decimator=<DECIMATOR>]
```

-a <AXIS>, --axis=<AXIS> Axis selection [x | y | xy]

-a <AXIS>, --axis=<AXIS> Axis selection [x | y | xy]

-A <AMPLITUDE>, --amplitude=<AMPLITUDE> Peak amplitude of the generated signal [Float]

-n <NSAMPLE>, --nsample=<NSAMPLE> Number of sample to record [Unsigned]

-u <SENSORUNIT>, --sensorunit=<SENSORUNIT> Sensor unit [raw | volt | norm | pos]

-s <SENSORALGO>, --sensoralgo=<SENSORALGO> Sensor algorithm to use [off | poly | lut]

-o <OFFSET>, --offset=<OFFSET> Amplitude offset of the generated signal [Float]

-D <DECIMATOR>, --decimator=<DECIMATOR> Decimator value (frec = fsample/decimator) (default: 1) [Unsigned]

-A <AMPLITUDE>, --amplitude=<AMPLITUDE> Peak amplitude of the generated signal [Float]

-n <NSAMPLE>, --nsample=<NSAMPLE> Number of sample to record [Unsigned]

-u <SENSORUNIT>, --sensorunit=<SENSORUNIT> Sensor unit [raw | volt | norm | pos]

-s <SENSORALGO>, --sensoralgo=<SENSORALGO> Sensor algorithm to use [off | poly | lut]

-o <OFFSET>, --offset=<OFFSET> Amplitude offset of the generated signal [Float]

-D <DECIMATOR>, --decimator=<DECIMATOR> Decimator value (frec = fsample/decimator) (default: 1) [Unsigned]

staticnoise command

Inject a static noise in the system and measure the sensor response.

Usage:

```
characterisation staticnoise --axis=<AXIS> [--amplitudex=<AMPLITUDEX>
--amplitudey=<AMPLITUDEY> --sensorunit=<SENSORUNIT> --sensoralgo=<SENSORALGO>
--starttime=<STARTTIME> --nsample=<NSAMPLE> --decimator=<DECIMATOR>
--amplitude=<AMPLITUDE> --nsample=<NSAMPLE> --sensorunit=<SENSORUNIT>
--sensoralgo=<SENSORALGO> --offset=<OFFSET> --decimator=<DECIMATOR>]
```

-a <AXIS>, --axis=<AXIS> Axis selection [x | y | xy]

-x <AMPLITUDEX>, --amplitudex=<AMPLITUDEX> Peak amplitude of the x axis excitation signal in volt. [Float]

-y <AMPLITUDEY>, --amplitudey=<AMPLITUDEY> Peak amplitude of the y axis excitation signal in volt [Float]

-u <SENSORUNIT>, --sensorunit=<SENSORUNIT> Sensor unit [raw | volt | norm | pos]

-s <SENSORALGO>, --sensoralgo=<SENSORALGO> Sensor algorithm to use [off | poly | lut]

-T <STARTTIME>, --starttime=<STARTTIME> Begin record after a given delay in seconds (useful to skip the transcient stage) [Float]

-n <NSAMPLE>, --nsample=<NSAMPLE> Number of sample to record [Unsigned]

-D <DECIMATOR>, --decimator=<DECIMATOR> Decimator value (frec = fsample/decimator) (default: 1) [Unsigned]

-A <AMPLITUDE>, --amplitude=<AMPLITUDE> Peak amplitude of the generated signal [Float]

-n <NSAMPLE>, --nsample=<NSAMPLE> Number of sample to record [Unsigned]

-u <SENSORUNIT>, --sensorunit=<SENSORUNIT> Sensor unit [raw | volt | norm | pos]

-s <SENSORALGO>, --sensoralgo=<SENSORALGO> Sensor algorithm to use [off | poly | lut]

-o <OFFSET>, --offset=<OFFSET> Amplitude offset of the generated signal [Float]

-D <DECIMATOR>, --decimator=<DECIMATOR> Decimator value (frec = fsample/decimator) (default: 1) [Unsigned]

CHAPTER

FIVE

ERROR MANAGEMENT

This section lists the error messages sent by the CLI.

When an error occurs a message is displayed by the CLI with the specific format:

```
error:  ERR{ID}. {Message}
```

5.1 Command parser errors

ID	Error Message
100	unmanaged error. Command returns status <value>.
101	Error while parsing command
102	Error while parsing parameter <param>.
103	Parameter <param> is missing
104	Sudo privileges are required to change parameter <param>.
105	Incorrect value for parameter <param>.
106	Command <param> is not valid.
107	Wrong parameter <param>.
108	Parameter <param> requires value.
109	Line return missing.
110	Option <param> does not take any argument.
111	Options <param> and <param> cannot be used at the same time.
112	Sudo privileges are required to perform this action.
113	Not implemented feature.
114	Sub-command <param> of <param> is not valid.
115	Missing sub-command for <param>.
116	No MEMS detected. This function requires sudo privileges or a MEMS. Please plug a MEMS and reset the board.
117	Can not start operation because control strategy is enabled. Please call "control strategy off" first.
118	Incorrect value for parameter <param>. Should be between <value> and <value>.
119	The format <param> is not implemented on this command.
120	Sudo privileges are required to set option <param> to parameter <param>.

5.2 Deflection unit power driver errors

ID	Error Message
200	Unmanaged error in driver module.
201	the requested functionality is not implemented on this board revision.
202	can not set requested voltage.

5.3 Extension manager errors

ID	Error Message
300	Unmanaged error in extension module.
301	Invalid channel selection for digital extension.
302	Invalid function selection for digital extension
303	Invalid channel selection for analog extension.
304	Invalid function selection for analog extension.
305	Invalid channel selection.

5.4 Sensor manager errors

ID	Error Message
400	Unmanaged error in sensor module.
401	No sensor detected. Check the connection and reboot.
402	Error occurred while setting the current in the sensor LED.
403	Error occurred while calibrating the sensor.
404	Error occurred while setting the sensor gain.
405	Error occurred while setting the temperature compensation algorithm.
406	Error while setting the position strategy.
407	Error while using wrong unit type.
408	Position strategy is not defined.
409	Calibration temperature not available.
410	Fail to detect the sensor.
411	Detection timeout occurred.
412	Message length exceed the maximum authorised.
413	Communication timeout occurred.

5.5 Consigne manager errors

ID	Error Message
500	Unmanaged error in consigne_manager module.
501	Error on adc conversion.
502	Error on start value.
503	Invalid waveform selection.
504	Invalid channel selection.
505	Error while reading adc value.
506	Requested amplitude value is too high.
507	Requested amplitude is higher than the Vpp voltage on the board. Decrease the amplitude or set a high VPP voltage (if available) using the system command.
508	Requested deflection angle is too high. Mems max angles are: <value> <value>
509	Request distance is too high.
510	Invalid unit selection.
511	Error while setting the unit.
512	Error while setting the source.
513	Can not use this command since input source is not configured on <param>.
514	Can not use this unit since user setup has not properly been set yet. Please use "system user_setup" first.
515	Feedback board must be used to set this strategy.
516	Invalid unit selection. Valid units are: <param>.
517	Invalid unit selection. Valid units are: <param> and <param>.
518	Invalid axis selection.
519	Can not change amplitude or offset from units <param> to <param>
520	Can not change both the phase and the frequency at the same time while the controller is running
521	Can not change the n value while the controller is running
522	The requested transition factor r is not valid (should be between 0 and 1)

5.6 Record manager errors

ID	Error Message
600	Unmanged error in record module.
601	Invalid time selection.
602	Invalid trigger value.
603	Too many samples requested.
604	Can not cancel the record in progress.
605	One extension pin must be first selected using the extension command.
606	Record in progress and cannot display results now.
607	A connected MEMS or sudo privileges are required to start a record.
608	The requested channel is disabled.
609	The recording cannot start as no channel have been selected.
610	The requested array index is not available.
611	The requested channel is not available.
612	The requested register is not available.
613	The register cannot be initialised because all channels are used.

5.7 Memory errors

ID	Error Message
700	Unmanaged error in memory module.
701	Invalid one wire device address or device not connected.
702	Error in one wire device scratch pad data.
703	Error during one wire device write: wrong data length.
704	Unmanaged error occurred during one wire memory operation.
705	Error during i2c write. Error code: <value>
706	Error during flash memory write.
707	Invalid memory selection.
708	CRC check failed. Input: <value>. Calculated: <value>.
709	Input data length is not correct. Input: <value>. Counted: <value>.
710	Trying to read outside the requested memory.
711	EEPROM wrong API version.
712	EEPROM error while reading remote hashcode.
713	EEPROM remote hashcode is 0xFF..F. The MEMS EEPROM might not be flashed.
714	Too many I2C read fail attemps.
715	Invalid MEMS image.
716	Function not implemented for the specified api.
717	Channel number not within 0 to 3.
718	Channel number not within 0 (for X) and 1 (for Y)
719	Channel number should only be the default value if the device channel has already been selected.
720	Error during json parsing.

5.8 MEMS characterisation errors

ID	Error Message
800	Unmanaged error in sensor characterisation module.
801	Invalid whitenoise reference signal selection.
802	Characterisation timeout occured.
803	Characterisation fault manager triggered.
804	Characterisation step timeout occured.
805	Characterisation step fault manager triggered.
806	Characterisation noise timeout occured.
807	Characterisation noise fault manager triggered.

5.9 Boot manager errors

ID	Error Message
900	Unmanaged error in boot module.
901	Error occurred while running boot script.
902	Autoboot script failed.
903	Autoboot end message not detected.
904	Autoboot script format is incorrect.
905	Autoboot memory error.
906	Empty script error.
907	Autoboot command not authorized.
908	Electronics is not registered

5.10 Signal shaper errors

ID	Error Message
1000	Unmanaged error in signal shaper module.
1001	Unknown signal shape.
1002	Signal shape out of range.
1003	Invalid setup combination. The ratio of fx and N is too small. A ratio fx/N greater than 0.5 is always acceptable.
1004	Angle converter angle out of range.
1005	Angle converter argument out of range.
1006	Angle converter unknown units.

CHAPTER SIX

FEEDBACK

Do not hesitate to contact Sercalo Team for additional informations, suggestions and questions.