OPEN-EYES Raspberry CM3 ACCESS CONTROL

Document history

Version	Date	Changes description
1.0	5 may 2021	first release

Index

Defines and abbreviations

ABBREVIATIONS	MEANING
CM3	Raspberry compute module versione 3
SoM	System On Module
MCU	Micro Controller Unit
AI	Artificial Intelligence
HA	Home Automation
LAN	Local Area Network
IP	Internet Protocol
IRQ	Interrupt Request
FPGA	Field Programmable Gate Array
PPE	Personal Protective Equipment
IR	Infra Red
I^2C	Inter-Integrated Circuit, interconnection bus for integrated circuits
I^2S	Integrated Interchip Sound interconnection bus for sounds devices
SPI	Serial Peripheral Interface

DISCLAIMER

The **OPEN-EYES-RPI** must be operated only by personnel qualified for the specific task and installation environment, in accordance with all relevant documentation and safety instructions. A qualified person should be capable of fully identifying all installation and operation risks and avoid potential hazards when working with this product.

The information in this document is provided in connection with **OPEN-EYES** products. No license, express or implied or otherwise, to any intellectual property right is granted by this document or in connection with the sale of this products

Despite careful verification, it cannot be excluded that this manual contains errors or inaccuracies. **OPEN-EYES** cannot be held responsible for these errors or inaccuracies and undertakes to correct them in the next edition of the manual.

OPEN-EYES Srl reserves the right to make changes to the products described in this manual at any time without notice.

Symbols

The following graphic symbols can be found in the manual:

	Warning related to installation procedures and functionality.
	Warning related to security.

Introduction

OPEN-EYES-RPI is an open platform enabled with IP connectivity and capable of performing a wide range of tasks.

The main feature of **OPEN-EYES-RPI** is the high level of flexibility allowed by sensor-rich hardware and the popular Raspberry community that enable the user to implement a wide range of innovative applications.

OPEN-EYES-RPI was built on the Raspberry CM3 module, and is characterized by a compact enclosure that can be installed either on the wall or on an articulated arm. It is equipped with a 5" display with resistive touch screen, a sophisticated sound card with DSP enabling advanced functions, interchangeable sensor slots and a video camera.

OPEN-EYES-RPI, based on open source platform and a high performance hardware potentially able to implement artificial intelligence functions related to multiple sensors with which it is equipped enable the use of new technologies with which it is possible to create new services and redefine an evolved user experience.

Some application are:

- *access control;*
- *home assistant;*
- *home speaker;*
- *kiosk front end;*

Features

Integrated

- ✗ Power source 9/24Vdc;
- ✗ 5 inches LCD display 800x480 pixels with brightness control;
- ✗ resistive touchscreen;
- ✗ 1 port Fast Ethernet 10/100Mbit/s;
- ✗ 1 port USB 2.0 external;
- ✗ 1 port USB 2.0 intern;
- ✗ 2 programmable GPIO;
- ✗ 3 axis acceleration sensor;
- ✗ temperature and humidity sensor;
- ✗ watchdog;
- ✗ advanced power control system;
- ✗ high intensity frontal LEDs;
- ✗ Raspberry Compute Module CM3 compatible;
- ✗ small size 180x100x25 mm.

Optionals

- ✗ 5/8M pixels camera;
- ✗ advanced audio;
- ✗ proximity sensors;
- ✗ thermal scanner;
- ✗ infrared matrix sensor;

Description

The **OPEN-EYES-RPI** platform consists primarily of two parts interconnected via two 10-pin connectors:

- A compact block (**BASE MODULE**) containing the processing part (Raspberry CM3 module) an audio board, two optional boards for additional sensors and a 5 inch LCD monitor with resistive touch panel;
- A back plate (**WIRING MODULE**) equipped with the connectors for wiring the LAN and the power and I/O section.

OPEN-EYES-RPI is based on Raspberry CM3 and allows to take advantage of the wide range of open source software developed for it in addition to enabling the use of the most appropriate CM3+/CM3 lite module.

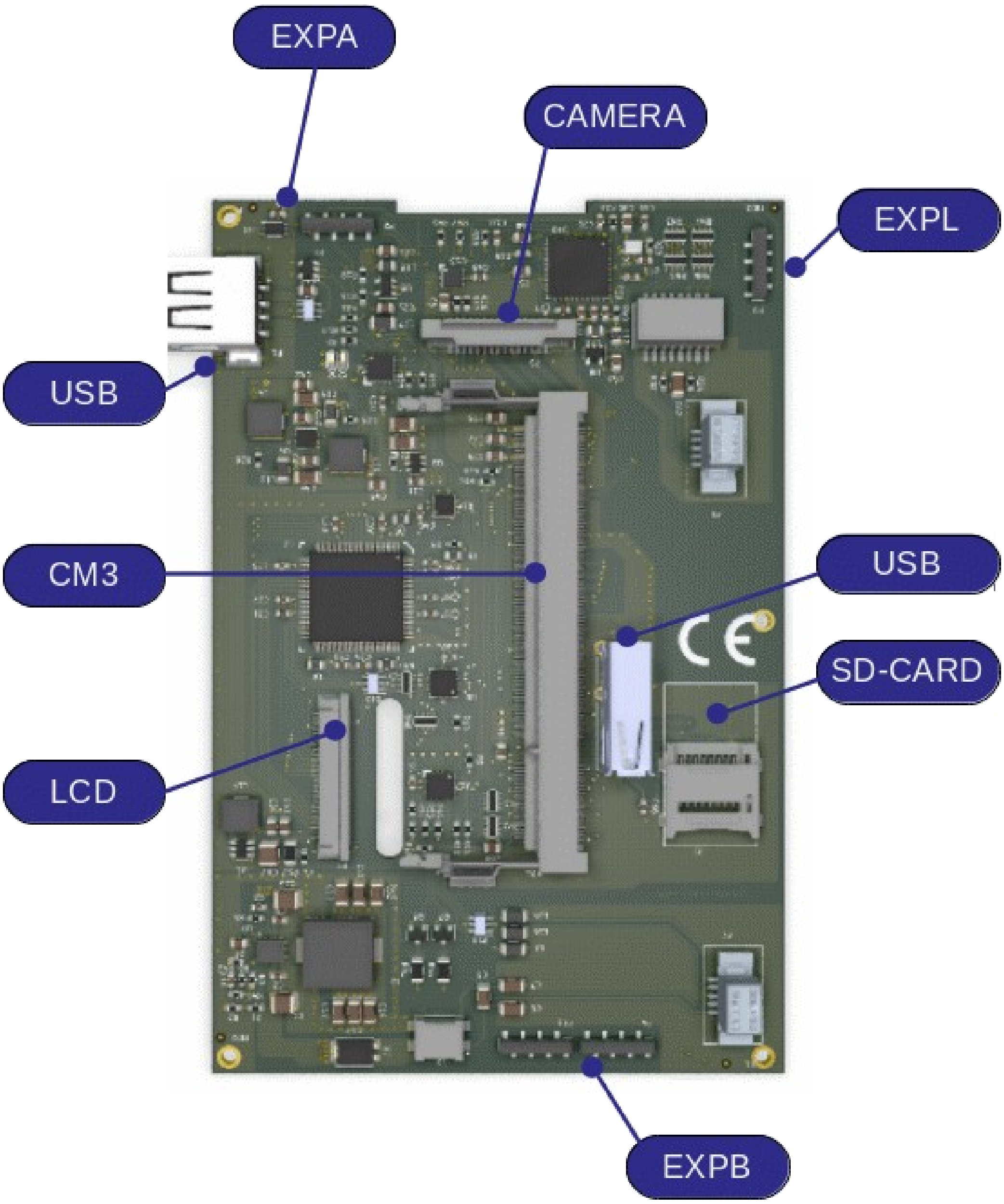
If the CM3 lite module is used (without flash memory on board) the memory card slot on the motherboard is available.

The CM3 compute module is based on a 1.2 GHz Broadcom BCM2837 ARM Cortex A53 (ARMv8) quad core processor and allows you to take advantage of the wide availability of software written for Raspberry.

Module characteristics:

- BROADCOM ® BC2837 SoC;
  - VideoCore IV GPU a 400 MHz;
  - ARM ® quad-core Cortex ® -A53 CPU Complex a 1.2 GHz;
- 1GB of RAM memory;
- Integrated temperature sensor;

Base Module



The base module consists of a main board on which the Raspberry CM3 module is installed and equipped with a series of functions and sensors that are fundamental for the functioning of the system.

Optional modules can be connected to the board, expanding the potential of the system.

Everything is enclosed in a plastic container made of recyclable material.

The additional slots are:

Camera

A 15-pin camera connector compatible with Rasperry's camera

Expansion module EXPA

On the upper side of the board is available an expansion slot, connected to the I^2C bus of the CM3. This slot is typically reserved for sensor or communication modules.

Expansion module EXPL

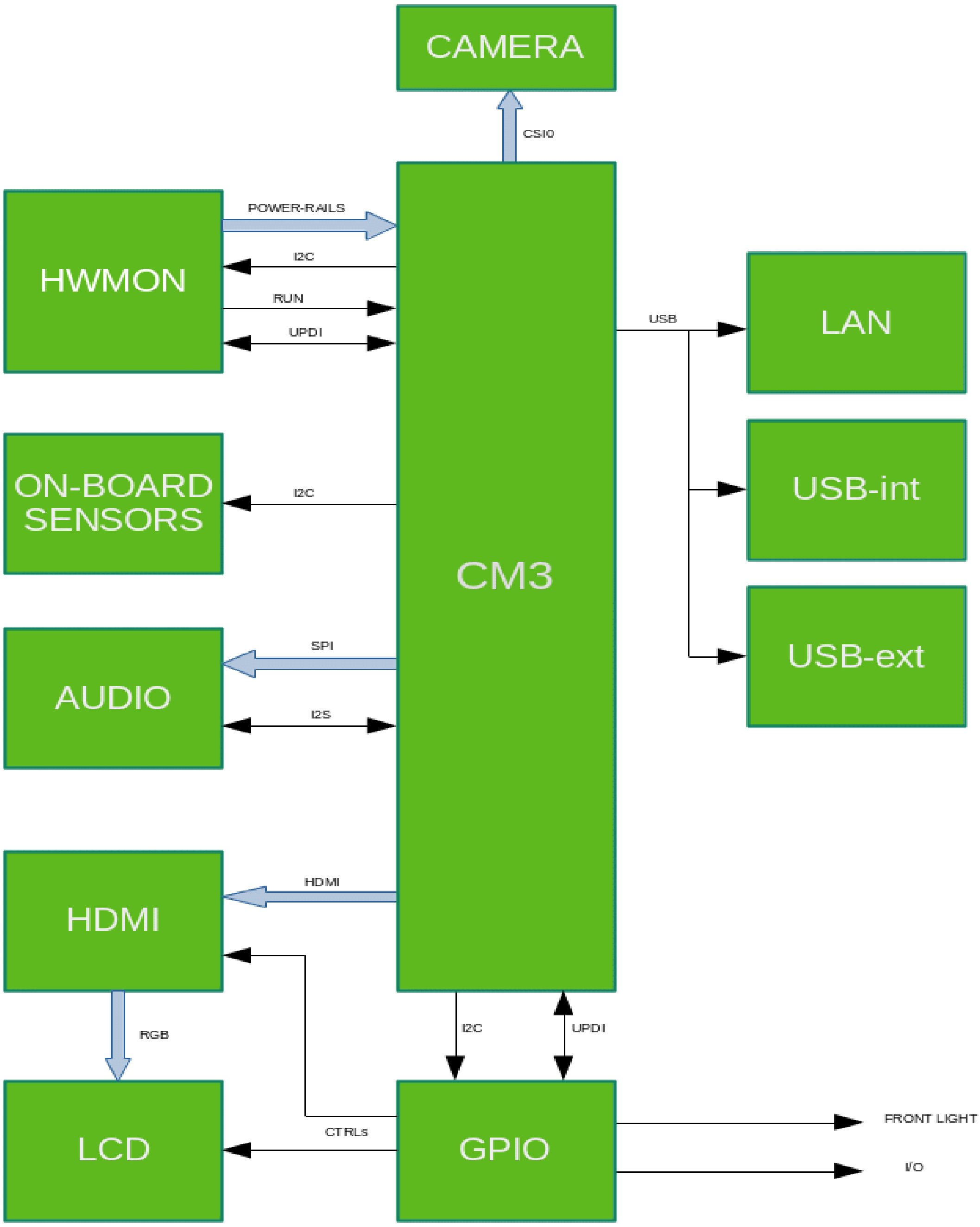
On the right side of the board a small size expansion module is reserved for proximity and movement sensors.

Expansion module EXPB

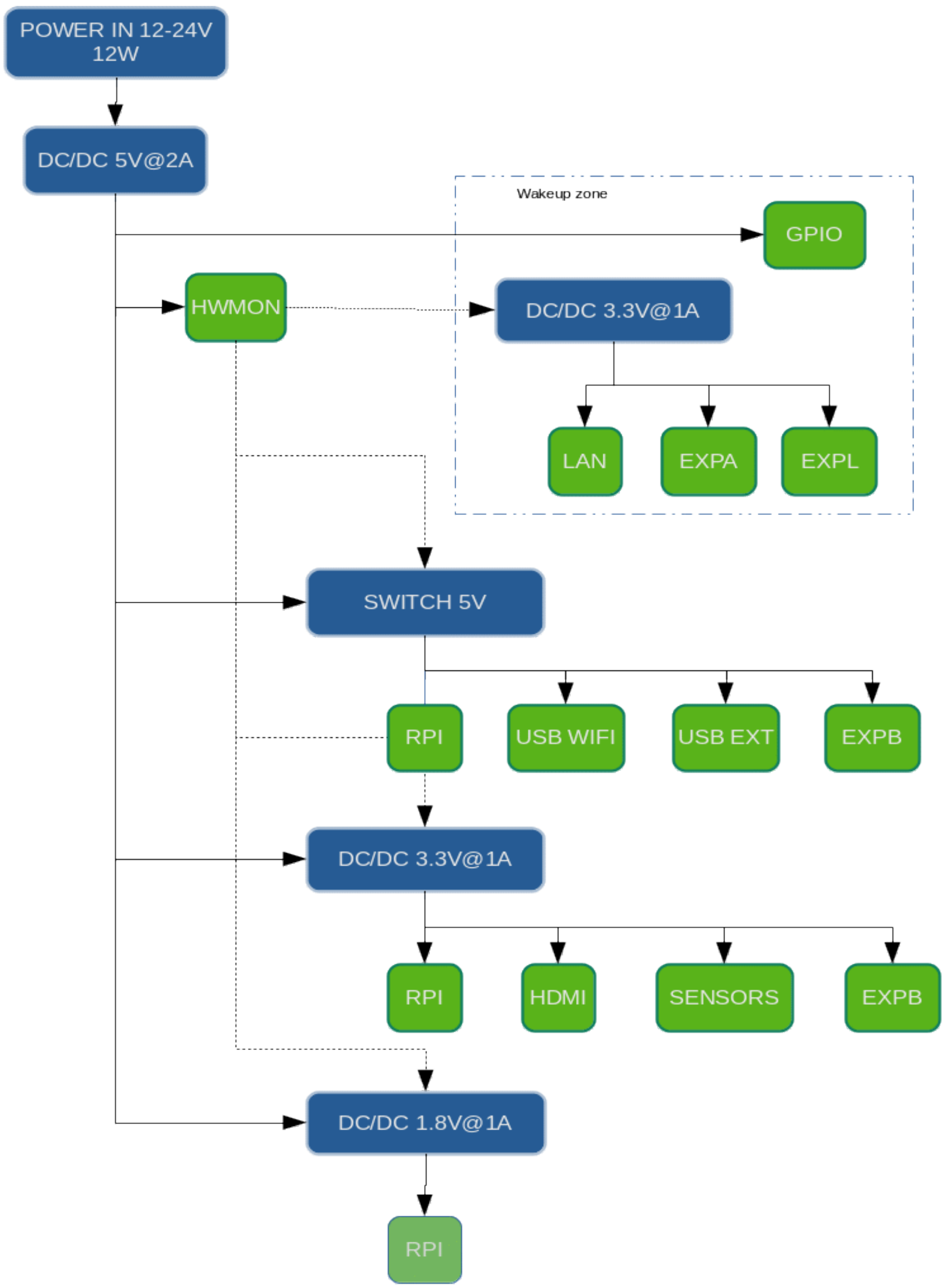
At the bottom a dual connector with a total of 16 pins is reserved for connecting a module (typically sound module) to which the SPI and I^2S interfaces of the CM3 are connected.

Finally, on the base module, a 200 pin SODIMM DDR1/DDR2 is reserved for installing the Raspberry CM3 module and an SD-card connector in case a CM3 lite module is used.

Blocks diagram



Power tree


POWER IN 12-24V
12W
DC/DC 5V@2A
Wakeup zone
GPIO
HWMON
DC/DC 3.3V@1A
LAN
EXPA
EXPL
SWITCH 5V
RPI
USB WIFI
USB EXT
EXPB
DC/DC 3.3V@1A
RPI
HDMI
SENSORS
EXPB
DC/DC 1.8V@1A
RPI

Integrated functions

On base module some basic functions are integrated as described below:

- ✗ hardware monitor;
- ✗ GPIO expansion and HDMI controls;
- ✗ temperature and humidity sensor;
- ✗ 3 axis acceleration sensor;
- ✗ USB and LAN;
- ✗ high intensity white LEDs;
- ✗ 5 inches display;
- ✗ crypto processor;

HWMON Hardware Monitor

The hardware monitor unit comprises a programmable MCU integrated on board and primarily dedicated to power control functions and Raspberry compute module CM3 boot and shutdown control process.
This unit is managed by the Raspberry via I^2C interface through a dedicated Linux driver.

The unit name is **SD109** and the matching Linux driver is **sd109-hwmon**.

Main functions:

- power on and boot management;
- power off and shutdown/halt management;
- power monitoring;
- watchdog hardware;
- status LEDs;
- pseudo real time clock functions;

Power on and power off

After a power up, the MCU, before supplying voltages to the SOC, checks that the input voltages are within the allowed limits, and activates the necessary voltages to the SOC in the correct sequence (5V --> 3,3V --> 1,8V), continuously monitoring that the voltages are correct and at the end of the power up procedure activates the SOC bootstrap sequence.

In case of scheduled shutdown (SHUTDOWN/HALT) at the end of the Linux shutdown procedure, the previous operations are performed in reverse sequence, ending with the system in a low-power state.

Power monitoring

During operation the MCU continuously reads the values of the internal voltages updating the MAX and MIN values and making them available through the Linux driver in the appropriate Linux hwmon class.

Watchdog

If enabled, it provides the SoC with a hardware watchdog module in addition to any software watchdog already provided by the Linux system, thus providing greater reliability.

The watchdog refresh mechanism is done through the I^2C interface, and in case of failure, it performs a complete cycle of power down and power up. The event is stored and made available to the Linux driver.

By default the watchdog is always disabled.

During the BOOT phase the watchdog is disabled for a time equal to WAITBOOT in order to allow a safe restart of the Raspberry SoC.

The watchdog timeout is fixed by the variable TIMEOUT.

In case of watchdog intervention, a maximum of 3 consecutive power cycles are executed. At the third power cycle without SOC activity, the system remains in HALT.

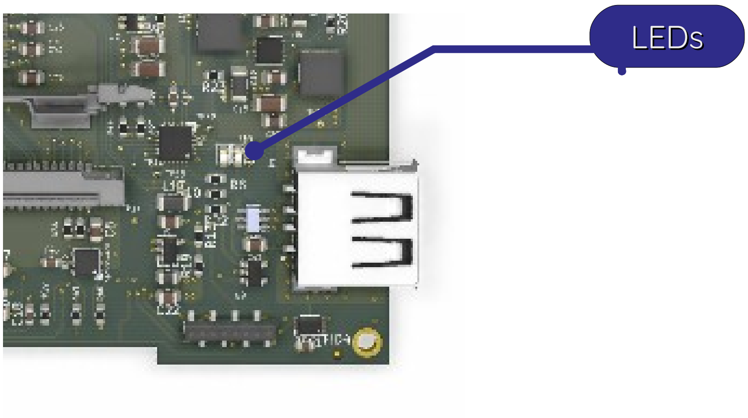
WAITBOOT and TIMEOUT values can by changed trough the overlay.

Status LEDs

Two LEDs, red and green, are positioned near the USB connector and are used to display the current MCU status for debugging purposes.

Neither LED is visible from outside.

LEDs position:



RED LED	GREEN LED	
OFF	OFF	5V or MCU boot failure
ON	ON	Error during MCU initialization
ON	slow blink	Waiting steady state voltages
ON	OFF	HALT status
Blink slow	OFF	POWERDOWN status
Blink fast	ON	Watchdog
OFF	slow blink	Raspberry shutdown or boot phase
OFF	fast blink	Normal status

slow blink = 0.5s ON 2s OFF

fast blink = 0.5s ON 0.5 OFF

WAKEUP function

Through the driver a pseudo RTC function is implemented, with the purpose of triggering a WAKEUP event at scheduled time. In short, the Raspberry can program a wake up time and go into a deep sleep mode with a very low power consumption.

Linux Driver

The hardware unit **SD109** is managed by the Linux driver **sd109-hwmon** available on github:

https://github.com/openeyes-lab/sd109-hwmon

The hardware function is to detect the unit presence on the I^2C bus and correctly register the following devices:

1. Kernel hardware monitor (ref: hwmon-kernel-api)
2. Linux Watchdog (ref: watchdog-driver-api)
3. Linux RTC

The driver is able to manage the registered device into the Linux file system through a set of MCU registers accessed by the I^2C interface, the device responds to the address 0x35 of the bus. The driver is loaded in the boot phase by the device tree overlay.

Linux driver installation

Clone the driver into a working directory:

git clone https://github.com/openeyes-lab/sd109-hwmon

cd sd109-hwmon

bash install.sh

for further information go to the README.md file

Overlay file

- **mandatory property**

compatible = "i2c,sd109"	define driver binding
reg = <0x35>	device I^2C address

- **optional property**

wdog_enabled	if present force driver to register the watchdog function. Otherwise the function is disabled.
wdog_nowayout	if present, once enabled, the watchdog can no longer be disabled.
wdog_timeout = <yy>	If watchdog is enabled a TIMEOUT value different from default (30s) can be defined.
wdog_wait = <xx>	If watchdog is enabled a WAITBOOT value different from default (90s) can be defined.
rtc_enabled	if present force driver to register the RTC function. Otherwise the function is disabled.
updi_lock = <zz>	if present force driver to take control of pin <zz> preventing the update of the MCU firmware. (1)
updi_en = <zz>	if present force driver to take control of pin <zz> enabling the update of the MCU firmware directly from driver. (1)

note:

(1) If updi_lock and updi_en are both present, updi_lock takes priority. In case are both absent the MCU UPDI pin can be managed by a user space program.

Memory map

The MCU memory map exposed on the I^2C bus consists of 32 16-bit registers, so the I2C operations supported by the device consist of a byte written by the MASTER on the I2C bus containing the address 0x35 and then a byte containing the address to be accessed. Finally a read or write operation of a 16 bit word is carried out.

Register table

ADDR	R/W	DESCRIPTION	DEFAULT
0x00	R	IDFW	0xd109
0x01	R	VERSION	0x0001
0x02	R	STATUS	-
0x06	W	COMMAND	-
0x08	W	WDOGREF	-
0x09	R/W	WDOGTIME	
0x0A	R	5V main current value	
0x0B	R	5V main minimum value	
0x0C	R	5V main maximum value	
0x0D	R	5V SOC and peripherals current value	
0x0E	R	5V SOC and peripherals minimum value	
0x0F	R	5V SOC and peripherals maximum value	
0x10	R	3,3V SOC and peripherals current value	
0x11	R	3,3V SOC and peripherals minimum value	
0x12	R	3,3V SOC and peripherals maximum value	
0x13	R	1.8V SOC and peripherals current value	
0x14	R	1,8V SOC and peripherals minimum value	
0x15	R	1,8V SOC and peripherals maximum value	
0x16	R	Vin board and peripherals current value	
0x17	R	Vin board and peripherals minimum value	
0x18	R	Vin board and peripherals maximum value	
0x1A	R/W	Current Time (sec)	
0x1B	R/W	Current Time (min)	
0x1C	R/W	Current Time (hour)	
0x1D	R/W	Current Time (day)	
0x1E	R/W	Current Time (month)	
0x1F	R/W	Current Time (year)	

STATUS

Read register with current status:

bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	-	-	-	-	-	-	-	-	-	-	-	WT	WD	BOOT		

Bits 0-2 encode the reason of last shutdown event

BOOT	SIGNAL NAME	DESCRIPTION
0	-	not allowed
1	POWERUP	the system come from a power up
2	POWEROFF	from RPI the command shutdown -P has been executed
3	REBOOT	from RPI the command shutdown -r has been executed
4	HALT	from RPI the command shutdown -h has been executed
5	-	not allowed
6	-	not allowed
7	-	not allowed

WD	DESCRIPTION
0	watchdog disabled
1	watchdog enabled

WT	DESCRIPTION
0	-
1	System restart from WAKEUP event

COMMAND

Write only register, write a CMD code on this register equals to execute the following command:

CMD	SIGNAL NAME	DESCRIPTION
0	-	not allowed
1	WDOGENABLE	enable watchdog
2	WDOGDISABLE	disable watchdog
3	POWEROFF	signal shutdown -P
4	REBOOT	signal shutdown -r
5	HALT	signal shutdown -H

WDOGREF

Execute watchdog refresh. To avoid the watchdog procedure to be triggered, the write of this register with the correct value (0xd1e) must be performed within the TIMEOUT value.

WDOGTIME

Set the watchdog timing. The register is read/write only when watchdog is disabled. Otherwise the register is read-only.

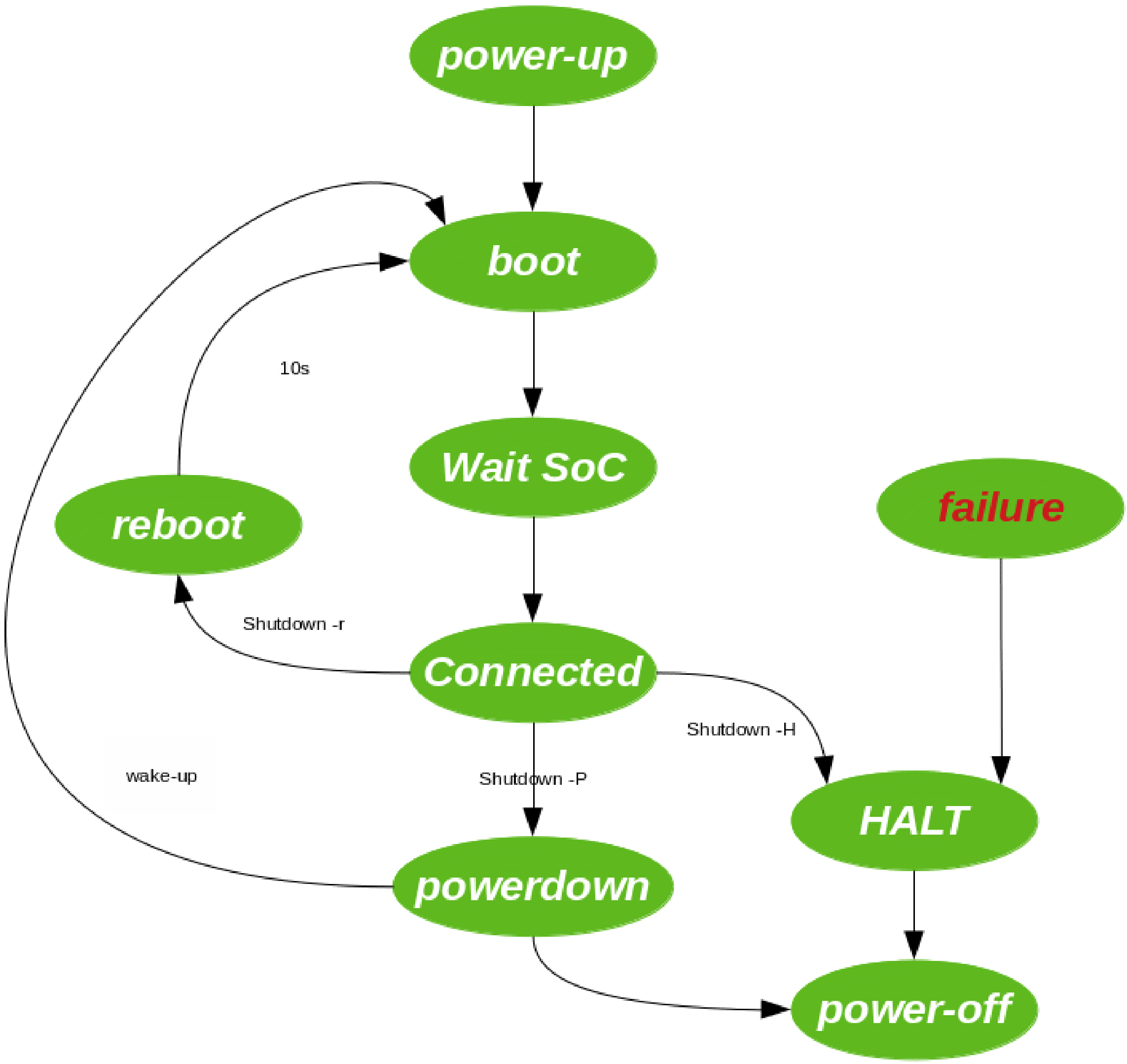
bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	BOOTWAIT								**TIMEOUT**							

BOOTWAIT define the waiting time following a BOOT before watchdog can be activated again. The time is in seconds / 5 . **BOOTWAIT**=10 is equal to 50s.

BOOTWAIT cannot be less than 9 (45s) otherwise a situation may ensue where there is not enough time for Raspberry to boot correctly.

TIMEOUT define the maximum time between two consecutive watchdog refresh after which the reboot procedure is triggered. The value is expressed in seconds, with valid values from 1 to 255 seconds.

SD109 relevant state



Power-up

Phase following the board power up. This phase has a fixed time length where voltage source must be stable. At the end of this phase the MCU firmware starts the boot phase.

Boot

Peripheral initialization (pins, I^2C, timer...) and correct power sequence of the RPI power rails. At the end of power sequence the RPI reset is released.

Wait SoC

RPI boot waiting, is the time between the release of reset and the first access to the MCU register from Raspberry

Connected

Normal status of SD109 unit.

HALT

Complete power down of Raspberry SoC and all SD109 functions. This state can be reached either by "shutdown -H" command executed to the SoC Linux system, or as a result of severe failures.

Exit from this state occurs only with a power cycle.

Powerdown

Complete power down of Raspberry SoC and SD109 functions running. This state can be reached by "shutdown -P" command executed to the SoC Linux system

The exit from this state can be done either by removing the power supply, or by a wake-up mechanism generated by a preset RTC alarm or by a hardware signal from other peripherals on the board.

Reboot

Transitional state that does not involve a power cycle of the Raspberry but simply a hard reset cycle, This state can be reached by "shutdown -r" or "reboot" command executed to the SoC Linux system.

Watchdog example

Configure a 10s timeout a long refresh time that cause the watchdog trigger.

Change directory where sd109 driver was cloned.

cd sd109/test

make

sudo ./wdog -d -t 10 -p 1000 -e

The SoC will be reset 3 times, after which the module will remain in HALT.

WakeUp example

Program a wake up alarm 180s delayed from now and execute a power down

sudo sh -c "echo +180 > /sys/class/rtc/rtc0/wakealarm"

sudo shutdown -P now

After 180s the system reboot autonomously

Hwmon example

Read internal system voltage

All value are read by the driver called from the lm_sensors package.

pi@openeyes:~ $ sensors

```
sd109-i2c-1-35
Adapter: bcm2835 (i2c@7e804000)
BOARD 5V:    +5.02 V  (min =  +4.98 V, max =  +5.08 V)
SoC 5V:      +5.01 V  (min =  +4.94 V, max =  +5.08 V)
SoC 3V3:     +3.27 V  (min =  +3.26 V, max =  +3.27 V)
SoC 1V8:     +1.79 V  (min =  +1.78 V, max =  +1.79 V)
Vin 24V:    +16.28 V  (min = +16.14 V, max = +16.48 V)

cpu_thermal-virtual-0
Adapter: Virtual device
temp1:        +31.1°C

si7006-i2c-1-40
Adapter: bcm2835 (i2c@7e804000)
temp1:        +22.4°C  (low  = +16.1°C, high = +22.4°C)
humidity1:     37.4 %RH

rpi_volt-isa-0000
Adapter: ISA adapter
in0:           N/A
```

Field Upgrade

The on field upgrade of the MCU (Microchip ATTiny817) is allowed controlling the UPDI pin from Raspberry.

The upgrade can be managed directly from Linux driver or through a separate program running in user space.

Upgrade disabled

In this case the upgrade isn't allowed, this situation can be forced adding the line:

updi_lock = <pin>

into the dts file

With this constraint the driver takes control of the UPDI pin disallowing other programs from having access to the MCU control functions.

Upgrade from kernel driver

The MCU upgrade can be done directly from Linux driver, to enable this add the line:

updi_en = 36   (see Compute module GPIO MAP)

into the sd109-hwmon.dts file inside the dts directory of the cloned driver. Then simply run again the install.sh script and reboot.

To upgrade the device simply follow:

cat sd109_verx.hex > /dev/sd109

Upgrade from user space

The driver has no indication to use the UPDI pin that is left free and so can be used by user space programs to upgrade the MCU.

GPIO - I/O controls and LCD display

The GPIO unit is built up as a programmable MCU integrated on board, dedicated to I/O and display management.

The device is controlled by the Raspberry SoC trough the I^2C interface by a Linux driver.

The unit name is **SD108** and the matching Linux driver is **sd108-gpio**.

Main functions:

1. front LEDs control;
2. 2 channel multipurpose GPIO;
3. LCD power control;
4. LCD brightness control;

Front LEDs

Two high intensity white LEDs are placed on the front side of the device. Both LEDs are seen by the operative system as independent PWM channels.

Multipurpose GPIO

Controls a pair of multipurpose I/O pins exposed on the terminal block screw. The basic version sees the two pins as standard I/O that can be configured independently as input or output.

Enhanced version of **SD108** can make more advanced features available like:

- UART TTL
- PWM
- I^2C simulated

LCD power and backlight

The Linux kernel driver manage LCD power and brightness trough Linux kernel back light core.

Linux driver

The **SD108** hardware unit is managed by the linux kernel driver **sd108-gpio** available on github:

https://github.com/openeyes-lab/sd108-gpio

The function of the driver is to detect the presence of the SD108 MCU on the I^2C bus and then register the following devices correctly:

1. Linux Kernel backlight support (ref: Backlight support)
2. Linux Kernel gpio support (ref: General Purpose Input/Output (GPIO))
3. Linux Kernel pwm support (ref: Pulse Width Modulation (PWM) interface)

The driver can manage the devices registered in the filesystem through a series of registers of the MCU visible through the I2C interface and the device responds to the address 0x36 of the bus.

The driver is loaded during the boot phase according to the device tree overlay.

Install Linux driver

Clone driver into a working directory:

git clone https://github.com/openeyes-lab/sd108-gpio

cd sd108-gpio

bash install.sh

For further information go to README.md file.

Device tree overlay file

- **mandatory property**

compatible = "i2c,sd108"	define driver binding
reg = <0x36>	device I^2C address

- **proprietà opzionali**

updi_lock = <zz>	if present force driver to take control of pin <zz> preventing the update of the MCU firmware. (1)
updi_en = <zz>	if present force driver to take control of pin <zz> enabling the update of the MCU firmware directly from driver. (1)

note:

(1) If updi_lock and updi_en are both present, updi_lock takes priority. In case are both absent the MCU UPDI pin can be managed by a user space program.

Memory map

The MCU memory map on the I^2C bus consists of sixteen 16-bit registers, so the I^2C operations supported by the device consist of a byte written by the MASTER on the I^2C bus containing the address 0x35 and then a byte containing the address to be accessed. Finally a read or write operation of a 16 bit word is carried out.

Register table

ADDR	R/W	DESCRIPTION	DEFAULT
0x00	R	IDFW	0xd108
0x01	R	VERSION	0x0001
0x03	R/W	PWMLED_ENABLE	0x0000
0x04	R/W	PWMLED_PERIOD	0x0000
0x05	R/W	PWMLED_RIGHT	0x0000
0x06	R/W	PWMLED_LEFT	0x0000
0x08	R/W	LCDBACK_PERIOD	0x0000
0x09	R/W	LCDBACK_DUTY	0x0000
0x0B	R/W	GPIO_DIR	0x0000
0x0C	R/W	GPIO1	0x0000
0x0D	R/W	GPIO2	0x0000
0x0E	W	COMMAND	
0x0F	R	STATUS	0x0000

PWMLED_ENABLE

Read/write register, enable the PWM timer related to the front leds.

bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																EN

PWMLED_PERIOD

Read/write register, define the PWM signal period in 100us ticks. This register is common to both PWM channel

bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
period x 100us																

PWMLED_RIGHT

Read/write register, define the PWM duty cycle of channel controling the front right LED.
The EN bit enable the channel, while DUTY can take the values from 0 to 100.

bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN									duty							

PWMLED_LEFT

Read/write register, define the PWM duty cycle of channel controlling the front left LED.
The EN bit enable the channel, while DUTY can take values from 0 to 100.

bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN									duty							

LCDBACK_PERIOD

Read/write register, define the PWM signal period in 1us ticks of the channel controlling the brightness of the LCD display.

bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
period x 1us																

LCDBACK_DUTY

Read/write register, define the PWM duty cycle of channel controling the brightness of the LCD display.
The EN bit enable the channel, while DUTY can take the values from 0 to 100.

bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN									duty							

GPIO_DIR

Read/write register, define function implemented on the pin.

bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IO2									IO1							

I/O type

I/O	TYPE	DESCRIPTION
0	INPUT	input signal
1	OUTPUT	output signal
2-255	-	reserved for future use

GPIO1-2

Read/write register.

bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
value																

STATUS

Read only register reporting the current status of device.

bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
															SL	AC

AC	DESCRIPTION
0	device disabled
1	device enabled

SL	DESCRIPTION
0	device in normal mode
1	device in sleep mode

COMMAND

Write only register, write a CMD code on this register equals to execute the following command:

CMD	NAME	DESCRIPTION
0x8001	SLEEP	power down device
0x8002	ACTIVE	device in normal mode

Front led control example

echo 0 > /sys/class/pwm/pwmchip12/export

echo 1000000 > /sys/class/pwm/pwmchip12/pwm0/duty_cycle

abilita pwm0 (led destro) con duty 50%

Display brightness control example

echo 5 > /sys/class/backlight/sd108/brightness

Field Upgrade

The field upgrade of the MCU (Microchip ATTiny817) is allowed, controlling the UPDI pin from Raspberry.

The upgrade can be managed directly from Linux driver or trough a separate program running in user space.

Upgrade disabled

In this case the upgrade isn't allowed, this situation can be forced adding the line:

updi_lock = 30 (see Compute module GPIO MAP)

into the dts file

With this constrain the driver take control of the UPDI pin avoiding other programs to access the MCU controls functions.

Upgrade from kernel driver

The MCU upgrade can be done directly from Linux driver, to enable this add the line:

updi_en = 30 (see Compute module GPIO MAP)

into the sd108-gpio.dts file inside the dts directory of the cloned driver. Then simply run again the install.sh script and reboot.

To upgrade the device simply follow:

cat sd108_verx.hex > /dev/sd108

Upgrade from user space

The driver has no instruction to use the UPDI pin, which is left free and can be used by user space programs to upgrade the MCU.

Temperature and humidity sensors

Together with the temperature sensor integrated in the compute module it allows to monitor the operating conditions of the apparatus and to implement predictive maintenance functions.

The sensor, consisting of an integrated SI7006, is supported by the operating system through the driver **si7006-hwmon** via the Hardware Monitoring kernel API of linux, and writes the values detected in the system directory:

/sys/class/hwmon

The driver is loaded during the boot phase according to the device tree overlay.

Linux kernel driver

The sensor is managed by the linux kernel driver **si7006-hwmon** available on github:

https://github.com/openeyes-lab/si7006-hwmon

The main driver function is to read sensor data and format them according to what is required from the hwmon Linux kernel core.

Linux Kernel hardware monitor kernel API support (ref: Linux Hardware Monitoring kernel API)

The sensor is controlled trough I^2C bus on address 0x40.

Linux driver installation

clone driver into a working directory:

git clone https://github.com/openeyes-lab/si7006.git

cd si7006-hwmon

bash install.sh

For further information go to README.md file.

Si7006 sensor use example

All value are reads by the driver called from the lm_sensors package.

```
pi@openeyes:~ $ sensors
sd109-i2c-1-35
Adapter: bcm2835 (i2c@7e804000)
BOARD 5V:     +5.02 V  (min =  +4.98 V, max =  +5.08 V)
SoC 5V:       +5.01 V  (min =  +4.94 V, max =  +5.08 V)
SoC 3V3:      +3.27 V  (min =  +3.26 V, max =  +3.27 V)
SoC 1V8:      +1.79 V  (min =  +1.78 V, max =  +1.79 V)
Vin 24V:     +16.28 V  (min = +16.14 V, max = +16.48 V)

cpu_thermal-virtual-0
Adapter: Virtual device
temp1:        +31.1°C

si7006-i2c-1-40
Adapter: bcm2835 (i2c@7e804000)
temp1:        +22.4°C  (low  = +16.1°C, high = +22.4°C)
humidity1:     37.4 %RH

rpi_volt-isa-0000
Adapter: ISA adapter
in0:            N/A
```

Touch screen sensor

Connected to the resistive touch screen panel located on the LCD, it allows to detect the pressure on the panel and to identify its position.

Compared to the capacitive touch panel, the resistive solution allows not only the use in more severe conditions, but also the operation with operator wearing gloves.

The sensor consists of an integrated **RHOM BU21026** and is supported by the operating system through the driver **bu21026-ts** via the Linux kernel input subsystem, and writes the values detected in the system directory:

/sys/class/input

The driver is loaded during the boot phase according to the device tree overlay.

Linux kernel driver

The sensor is managed via the **bu21026-ts** linux driver available on github:

https://github.com/openeyes-lab/bu21026-ts

The driver function is to detect pressure on the resistive touch panel calculating, coordinates and generating the correct event on the input device.

For information about the Linux input subsystem refer to Linux Input Subsystem kernel API (ref: Linux Hardware Monitoring kernel API)

The sensor is controlled trough I^2C bus on address 0x48

Linux driver installation

Clone driver into a working directory:

git clone https://github.com/openeyes-lab/bu21026-ts.git

cd bu21026-ts

bash install.sh

For further information go to README.md file.

3 axis accelaration sensor

It allows both the detection of the static orientation of the device and eventual transient accelerations. The sensor consists of an integrated ST LIS3DH able to detect accelerations with full scale selectable 2/4/8/16g and a maximum resolution of 12bit.

The sensor is supported by the operating system through the lis3dh driver, and like the touch sensor through the Linux kernel input subsystem, and writes the detected values in the system directory:

/sys/bus/iio

The driver is loaded during the boot phase according to the device tree overlay.

Linux kernel driver

The sensor is managed by the linux kernel driver **lis3dh-accel** available on github:

https://github.com/openeyes-lab/lis3dh-accel.git

Linux driver installation

Clone driver into a working directory:

git clone https://github.com/openeyes-lab/lis3dh-accel.git

cd lis3dh-accel

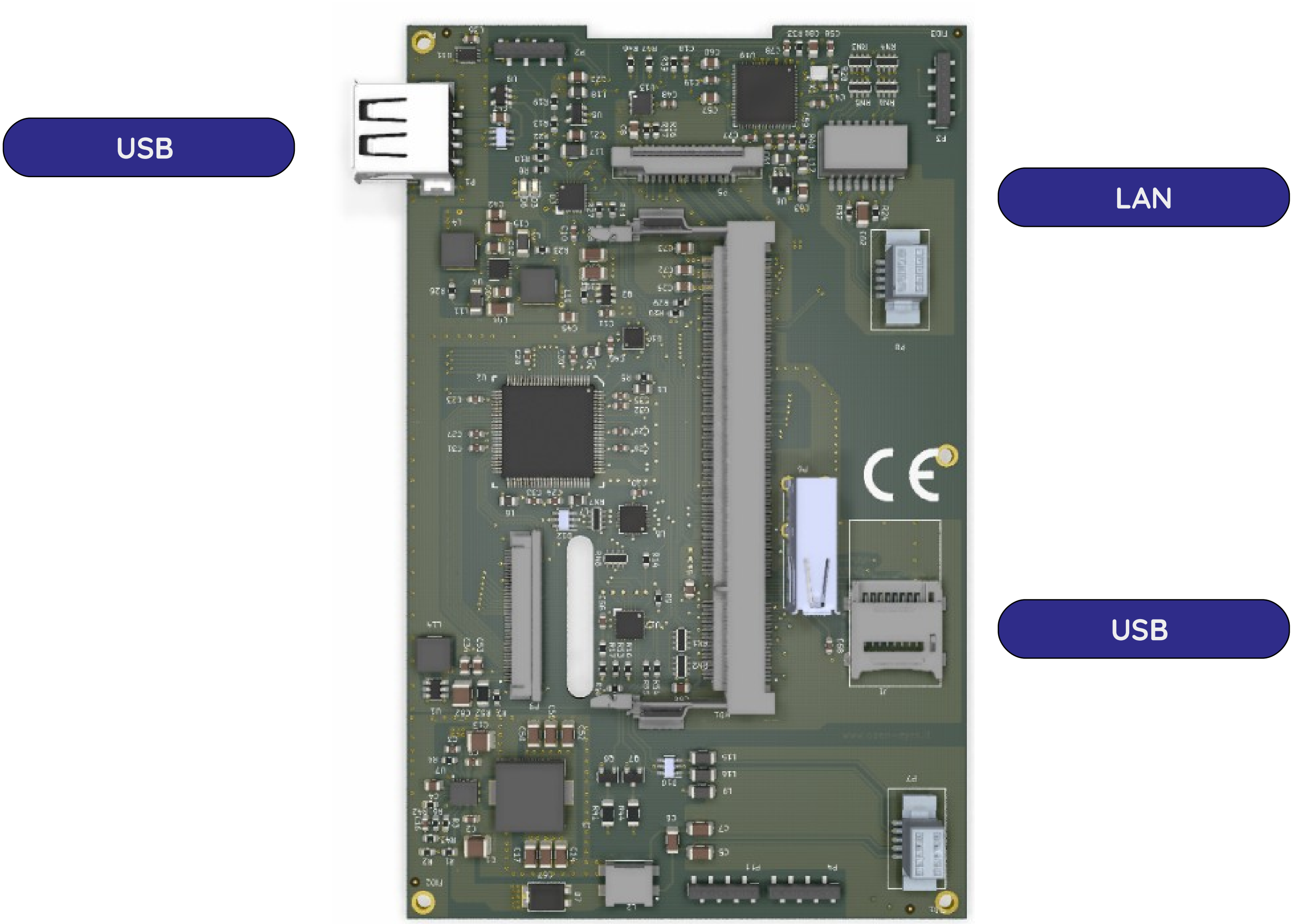
bash install.sh

For further information go to README.md file.

USB LAN

To the Raspberry CM3 SoC is connected to a component produced by MICROCHIP, the LAN9514 integrated circuit. (datasheet), fully integrated into the Rasberry platform.

LAN9114 provides support for two Hi Speed USB 2.0 interfaces and one 10/100 ethernet interface.



The LAN interface is connected to the CONNECTOR MODULE (on which there is an RJ45 connector) through the 10-pin connector. Of the two USB interfaces, the first is exposed outside, of the system; while the second is internal and allows you to equip the system with a small bluetooth or WIFI device.

5 inches HDMI display

LCD-TFT 800x480 pixel display with integrated resistive touch screen and back lighting device with adjustable intensity.

The display is connected to the CM3 through the HDMI port using an HDMI/RGB converter.

Backlighting and display enabling is managed by the operating system through the **sd108-gpio** module.

HIGH intensity LEDs

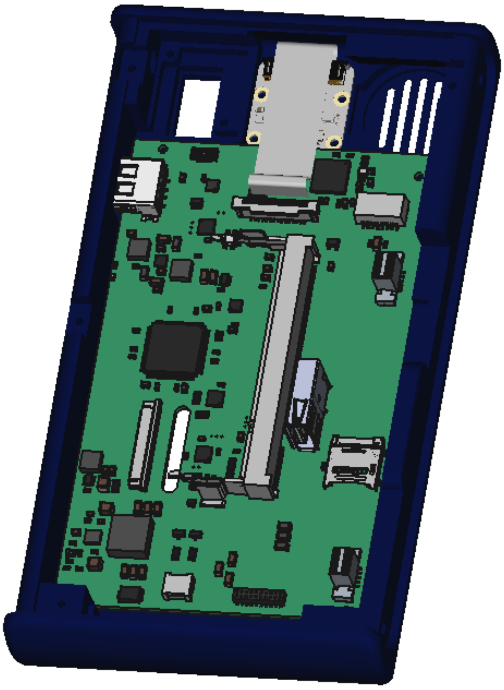
They are located on the front under the display and allow you to illuminate the area in front of the device for better use of the camera.

Compute module GPIO MAP

GPIO	SIGNAL NAME	TYPE	TERMINATION	NOTE
0	ice_done	IN	PULL-UP	audio board FPGA done pin
2	I2C1_sda	IN/OUT	PULL-UP	I^2C bus #1 data
3	I2C1_scl	OUT	PULL-UP	I^2C bus #1 timing
5	ice_reset	OUT		audio board FPGA reset pin
6	ice_select	OUT		audio board FPGA control pin
8	spi_ce0	OUT		SPI0 interface
9	spi0_miso	IN		SPI0 interface
10	spi0_mosi	OUT		SPI0 interface
11	spi0_sclk	OUT		SPI0 interface
15	expl_irq	IN	PULL-UP	IRQ slot EXPL
18	pcm_clk	OUT		I^2S timing
19	pcm_fs	OUT		I^2S frame sync
20	pcm_din	IN		I^2S data signal
21	pcm_dout	OUT		I^2S data signal
28	cam_sda	IN/OUT	PULL UP	I^2C bus #0 (camera) data
29	cam_scl	OUT	PULL UP	I^2C bus #0 (camera) timing
30	sd108 updi	IN/OUT		UPDI SD108-GPIO
32	uart_txd	OUT		AMA0 output data
33	uart_rxd	IN		AMA0 input data
36	sd109 updi	IN/OUT		UPDI SD109-HWMON
39	touch_irq	IN	PULL-UP	IRQ touch screen
40	accel_irq1	IN	PULL-UP	IRQ accelerometer
41	accel_irq2	IN	PULL-UP	IRQ accelerometer
42	expa_irq	IN	PULL-UP	IRQ slot EXPA
43	cam_mclk	OUT		
44	cam_pwdn	OUT		

Camera

At the top of the device, there is a slot for installing a standard Raspberry camera module.



The system can be equipped with the various types of cameras supported by Raspberry, here is the compatibility list:

CAMERA	CHIPSET
Raspberry Pi camera module V1.3 5Mp	OV5647
Raspberry Pi camera module V2.1 8Mp	IMX219

EXPA expansion module

It is possible to equip the **BASIC MODULE** with an optional **EXPA** module.

Optional sensors table

CODE	SENSOR	FUNCTION
TH113.1	IR image scanner	Detect of IR image
TH113.2	Termoscanner	Detect of body temperature
TH113.3	IR image scanner IR + termoscanner	Both previous sensors
SM134	weather station	Detection of ambient temperature atmospheric pressure, humidity and air quality

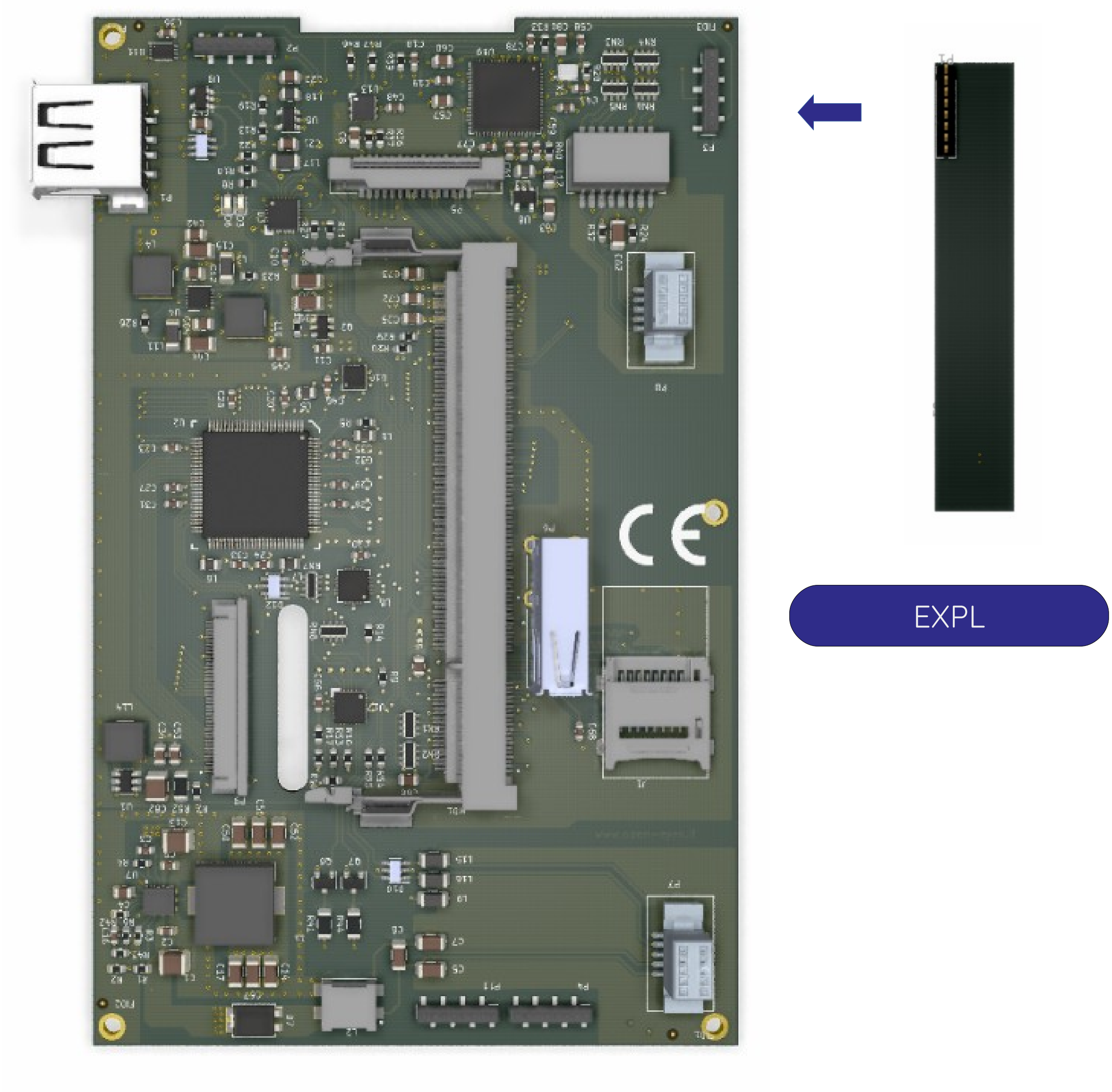
EXPA module connector

PIN	DESCRIPTION	TYPE	LEVEL
1	3,3V power input	power	3,3V
2	I^2C bus timing	INPUT	3,3V
3	I^2C bus data	I/O	3,3V
4	module enable	INPUT	3,3V
5	module irq	OUTPUT	3,3V
6	Wakeup	OUTPUT	OC
7	GND	power	
8	GND	power	

EXPL expansion module

The purpose of the **EXPL** expansion module located on the upper left side of **OPEN-EYES-RPI** is to provide the apparatus with the ability to detect the presence of objects or people in front of itself.

The **EXPL** expansion module is powered separately from the SoC CM3, and therefore can be active even with Raspberry completely unpowered and, in case of programmed events, the wake up line can force a reboot of the CM3 module.



Optional sensor table

CODE	SENSOR	FUNCTION
SP131	short range proximity sensor < 0.6m	gesture recognition
SP132	long range proximity sensor < 3m	presence detect
SL133	luminosity sensor	Ambient light detection

EXPL connector

PIN	DESCRIPTION	Tipo	Level
1	enable	INPUT	3,3V
2	GND	power	
3	3,3V power input	power	3,3V
4	I^2C bus data	I/O	3,3V
5	I^2C bus timing	I/O	3,3V
6	irq	OUTPUT	3,3V
7	Wake up	OUTPUT	Open collector
8	GND	power	

Signal description

ENABLE

Module enable signal:

- Logic level high - module enabled;
- Logic level low - module disabled;

BUS I^2C

Communication bus controlled by raspberry

IRQ

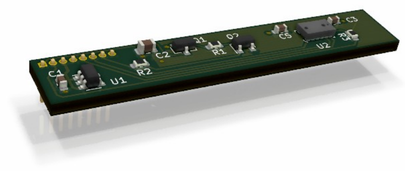
Interrupt request signal active low. It is connected directly to Raspberry and is controlled by the related Linux driver.

WAKEUP

Open collector signal connected to the **SD109-hwmon**. Can trigger a WAKEUP event for the Raspberry.

SP131 module

Equipped with an **ST microelectronics VL6180** sensor, it allows the system to measure the distance of objects within the sensor's field of view up to a distance of 60 cm



Technical specifications:

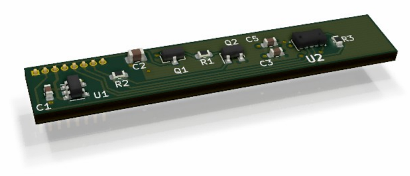
- distance from 0 to 60 cm;
- visual angle 25 degrees;
- measure error < 20mm;
- Independent of the reflection index of the object.

Applications:

- gesture recognition

SP132 module

Equipped with an **ST microelectronics VL53L3** sensor, it allows the system to detect multiple objects at a distance of up to 3m.



Technical specifications:

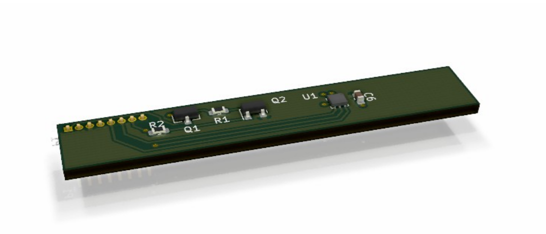
- distance from 0 to 300 cm;
- multiple object detect capacity;
- visual angle 25 degrees;

Applications:

- presence sensor;
- pass counting;

SP133 module

Equipped with a **TEXAS INSTRUMENT PT3001-Q1** Ambient Light Sensor, it allows the system to measure ambient luminosity.



Technical specifications:

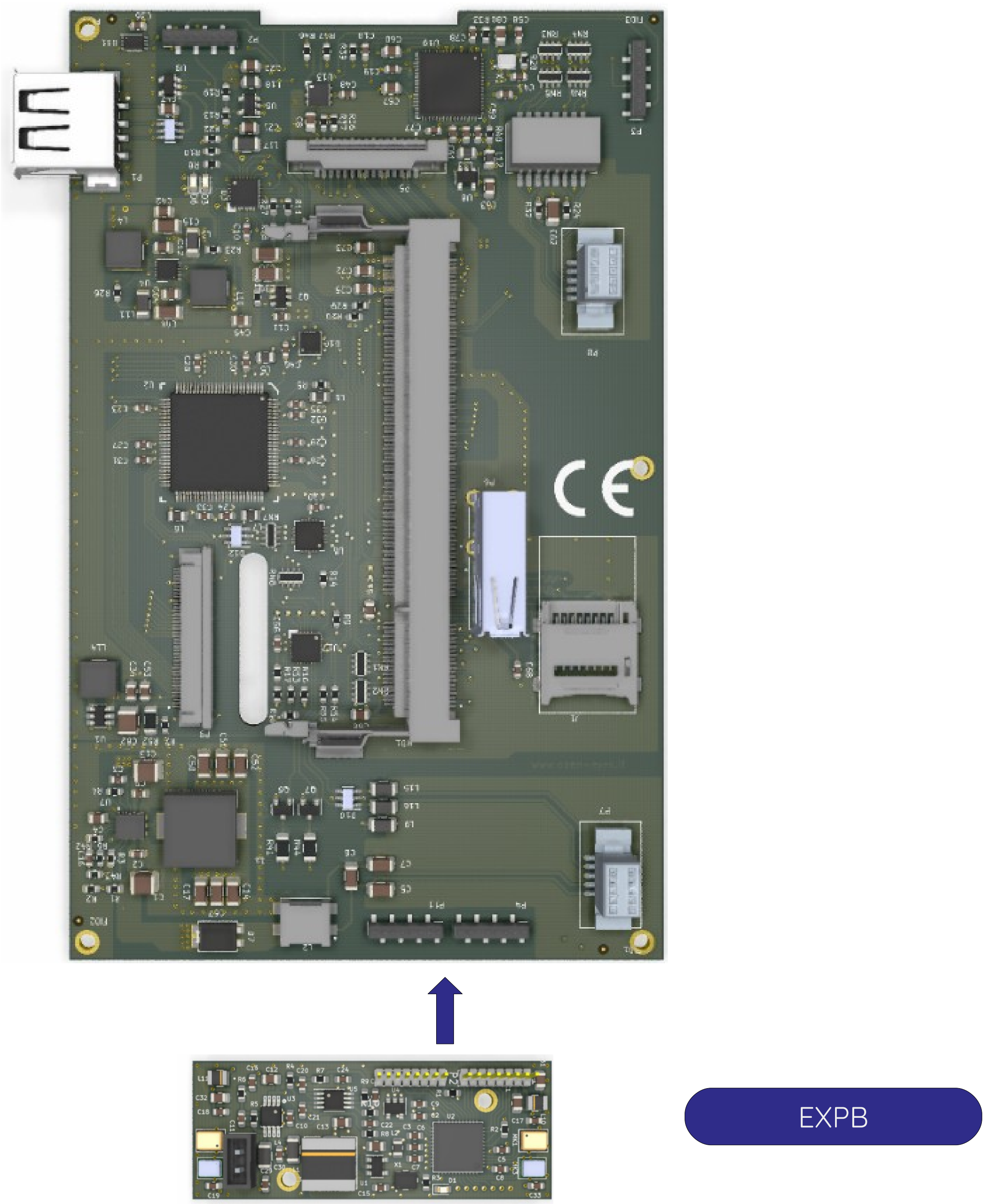
- Light measurement from 0.1 to 80k Lux

Applications:

- Display back lighting control;
- Lighting control.

EXPB expansion module

The main purpose of the lower expansion module is to provide **OPEN-EYES-RPI** with an audio interface. For this purpose the connector is provided with the SPI and I²S interfaces coming from the CM3 module.



Optional module table

CODE	MODULE	FUNCTION
AU104.1	simple audio board	single microphone
AU104.2	enhanced audio board	double microphone

EXPB connectors

EXPB-1 connector

PIN	DESCRIPTION	SIGNAL TYPE	LEVEL
1	digital power	power	3,3V
2	Ground	power	
3	GPIO00 - ICE DONE	EXPB →RPI	3,3V
4	GPIO05 - ICE RESET	RPI → EXPB	3,3V
5	GPIO19 - PCM_FS	RPI →EXPB	3,3V
6	GPIO20 - PCM_DIN	EXPB →RPI	3,3V
7	GPIO21 - PCM_DOUT	RPI → EXPB	3,3V
8	GPIO09- SPI0_MISO	EXPB →RPI	3,3V

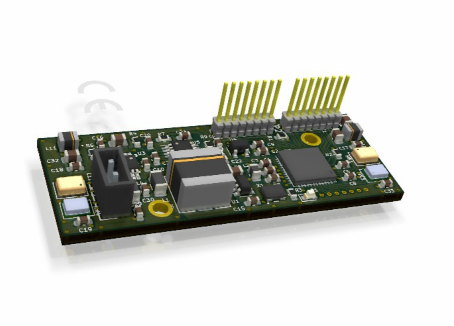
EXPB-2 connector

PIN	DESCRIPTION	SIGNAL TYPE	LEVEL
1	GPIO11 - SPI0_SCLK	RPI → EXPB	3,3V
2	GPIO06 - ICE SELECT	RPI → EXPB	3,3V
3	Digital power I/O SPI and I^2S	power	3,3V
4	GPIO10- SPI0_MOSI	RPI → EXPB	3,3V
5	GPIO08- SPI0_CS0	RPI → EXPB	3,3V
6	GPIO18 - PCM_CLK	RPI → EXPB	3,3V
7	Ground	power	
8	Analog power	power	+5V

AU104module

The **OPEN-EYES** device can be equipped with an optional sound card inserted in the lower part of the device.

The optional audio system includes one or two MEMS microphones, depending on the version, at the bottom and a speaker at the top of the device.



The displaced arrangement helps reduce echoing.

Audio base unit

It provides a microphone input with an I^2S digital interface having the following features:

PARAMETER	CONDITION	VALUE
Sensibility	94 dB SPL @ 1 kHz	-26 dBFS
Signal noise ratio (SNR)	94 dB SPL @ 1 kHz	65 dB
Harmonic distortion		<1%

One output channel with 1W speaker

Enhanced audio unit

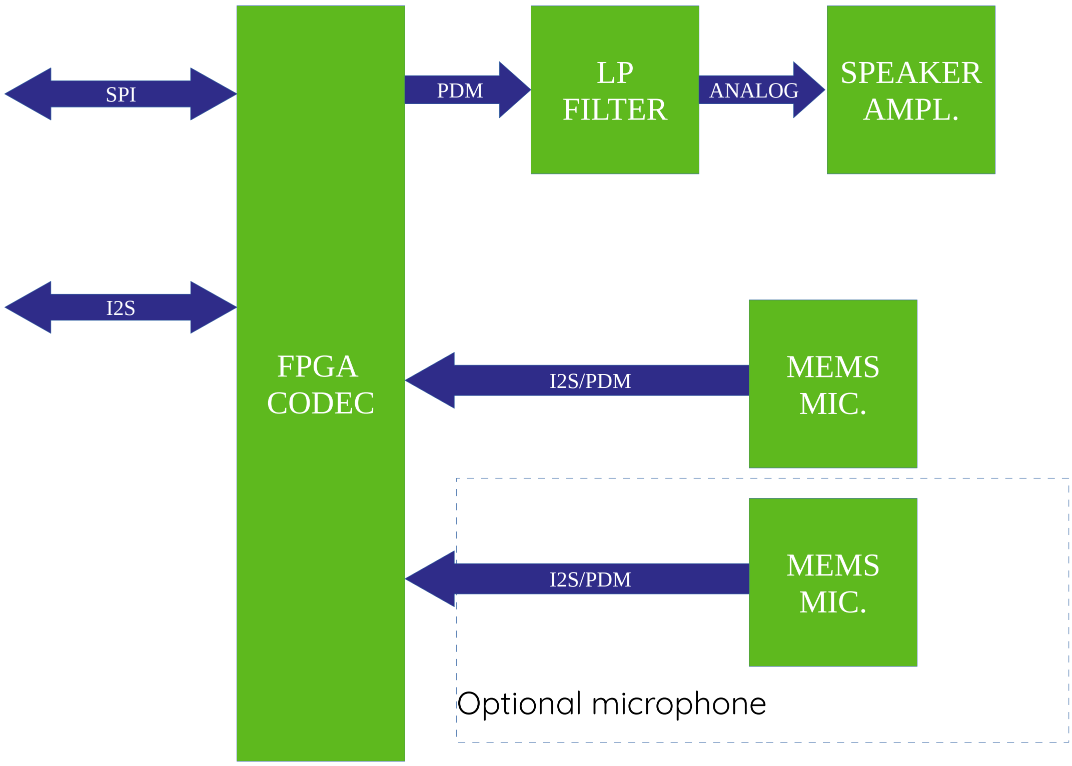
It provides two microphone inputs with PDM digital interface having the following features:

PARAMETER	CONDITION	VALUE
Sensitivity	94 dB SPL @ 1 kHz	-26 dBFS
Signal noise ratio (SNR)	94 dB SPL @ 1 kHz	69 dB
Harmonic distortion		<1%

One output channel with 2W speaker

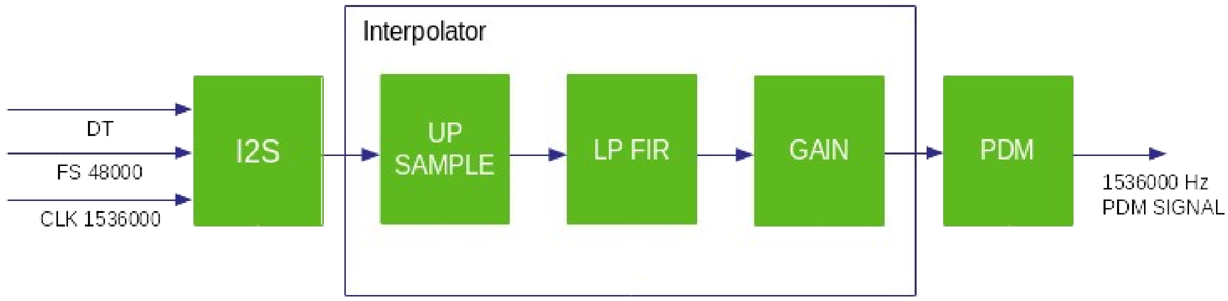
In both cases, the distinctive feature of the audio interface is that it is implemented through a programmable FPGA (Lattice UiCE40 UltraPlus UP5K) in which the IP blocks that realize the audio CODEC are instantiated.

Block diagram



Speaker CODEC filter

For the conversion of the digital signal coming from the I²S interface, we chose to convert the 16-bit digital signal into a PDM digital signal that can be easily converted into an analog signal through a simple low pass.



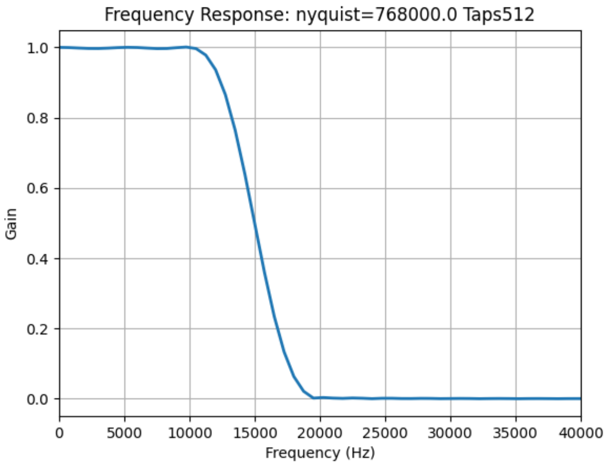
The system operates with an internal DSP frequency of 36,840,000 Hz and supports on the I²S side a symbol frequency of 48,000Hz which with two 16bit channels corresponds to a bit rate of 1,536,000 Hz.

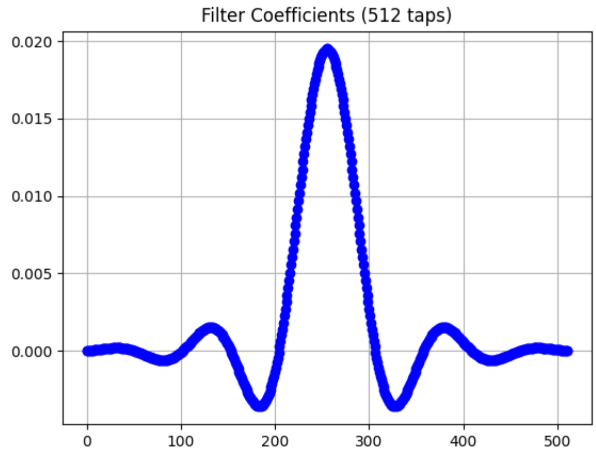
Conversion is performed by oversampling the incoming digital signal with a frequency 32 times higher, inserting null values into the added samples, and then filtering through a 512-tap FIR filter.

CARACTERISTICS	VALUE	u.m.
input signal frequency (Fi)	48000	Hz
output signal frequency (Fu)	1536000	Hz
interpolation ratio (L=Fu/Fi)	32	
digital filter lenght (T)	512	tap
polyphase filter lenght (P=T/L)	16	

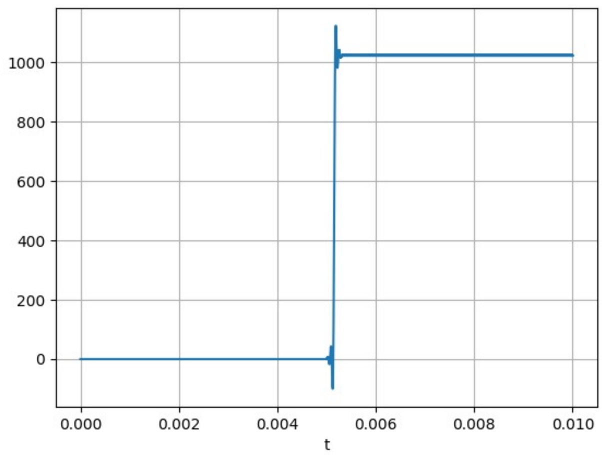
Interpolator filter caracteristics

The 512 TAP FIR filter provides 50dB out-of-band attenuation and an 8820 Hz transition band with a 15kHz cutoff frequency.

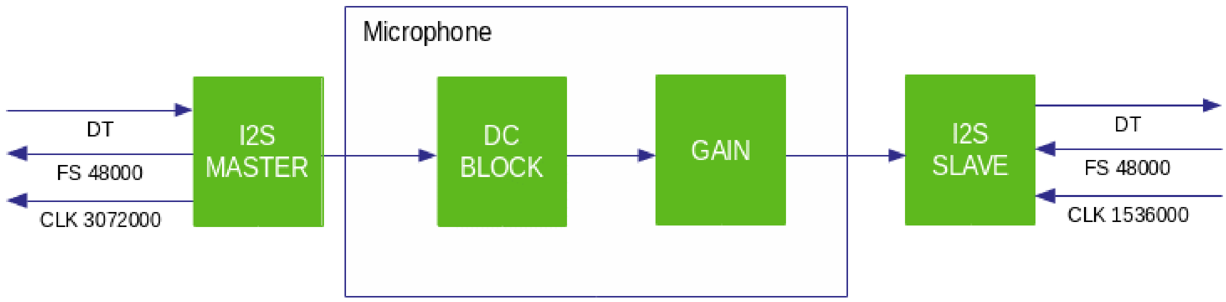




step response



Michrophone CODEC diagram



The microphone used is a **SPH0645LM4H-1 MEMS** microphone with 32-bit I2S interface of which 24 are actually used.

The digital signal received from the microphone is extracted from the I2S interface and sent to a DSP block for the elimination of the continuous component introduced by the internal microphone coding, then the signal is amplified and reduced to 16 bit to be passed to the I2S SLAVE interface on the CPU side.

FPGA available resource

As mentioned earlier, the audio CODEC is realized via VHDL IPs synthesized on a **LATTICE chipset ICE40UP5K**

The resource allocation is the following:

PARAMETER	USED	FREE	TOTAL
LUT	700	4580	5280
EBR RAM (kbits)	12	108	120
SPRAM (kbits)	0	1024	1024
DSP block	5	3	8

referring to the current version 1.0 of the CODEC, as shown in the table, over 80% of the resources remain free for the implementation of additional functions such as :

- key word detect
- ambient noise reduction
- eco cancellation

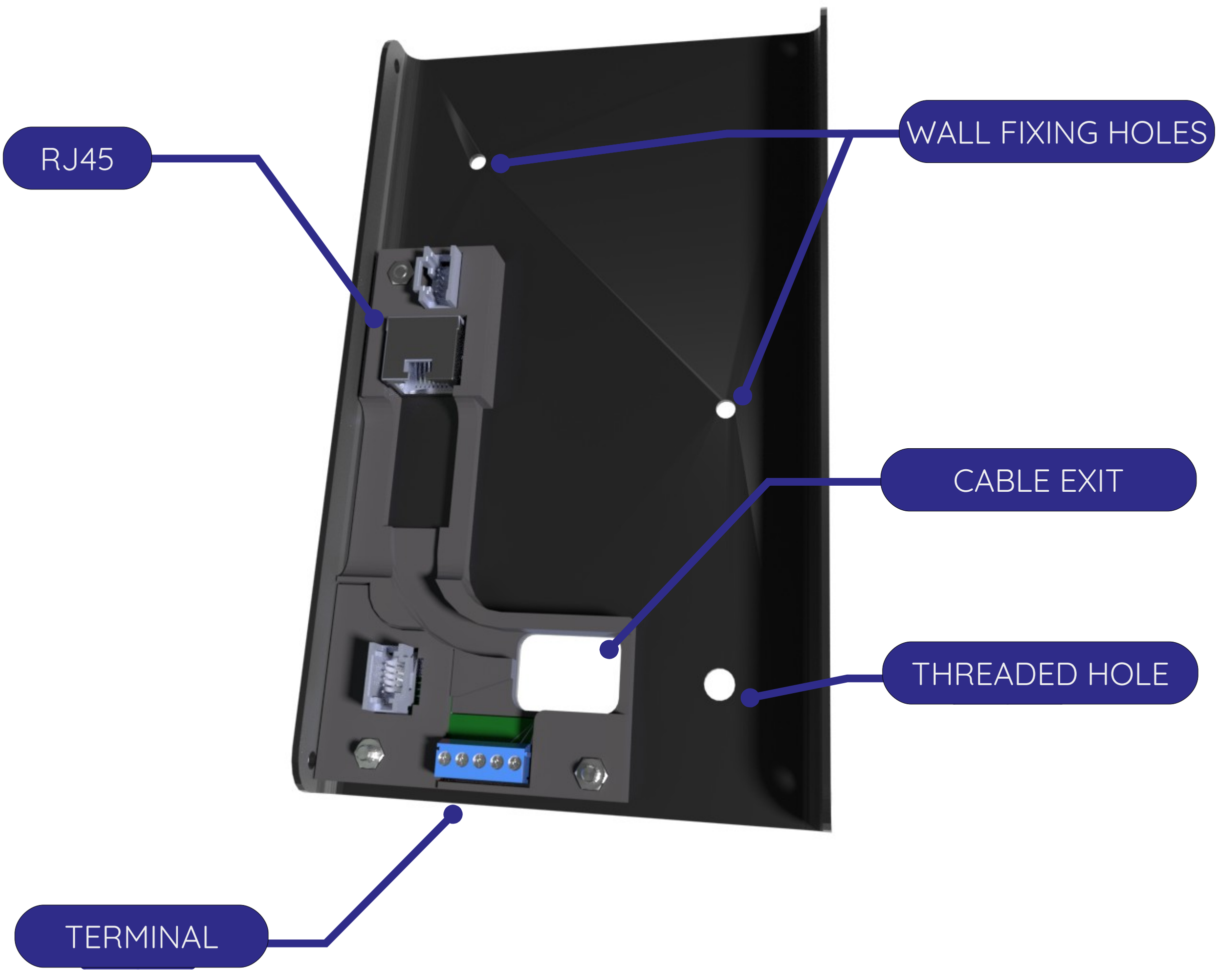
I^2C address map

ADDR	DESCRIPTION	BUS	POSITION	FUNCTION
0x18	LIS3DH	1	MODULO BASE	3-axis accelerometer
0x35	SD109	1	MODULO BASE	Hardware monitor power & wdog
0x36	SD108	1	MODULO BASE	GPIO e LCD backlight
0x40	Si7006	1	MODULO BASE	Temp/humidity
0x44	OPT3001	1	EXPL	Side-board sensor
0x48	BU21026	1	MODULO BASE	Resistive touch
0x52	VL53L3	1	EXPL	Lidar
0x52	VL6180	1	EXPL	Lidar
0x5A	MLX9014	1	EXPA	Temperature sensor
0x60	ATECC608A	1	MODULO BASE	Crypto
0x68	AMG8833	1	EXPA	IR array
0x76	BME680	1	EXPA	Air quality
-	HDMI	dedicato	MODULO BASE	HDMI master controller
0x77	BMP240	1	EXPA	Temperature/atmospheric pressure
-	CAMERA OV	dedicato	MODULO BASE	
-	CAMERA IMX	dedicato	MODULO BASE	

Connectors module

The connectors module has the purpose of making the installation easy as, once affixed to the structure, it allows to wire in a simple way the power supply, the eventual I/O and the LAN RJ45 connector. On the back plate are provided:

- ✗ two wall mounting holes;
- ✗ one 3/8 inches threaded hole for 1 threaded hole 3/8 for articulated arm mounting;
- ✗ RJ45 LAN connector;
- ✗ terminal block screw for power and GPIO cabling;
- ✗ cable exit hole;



Software platform description

Based on a Raspberry CM3 SoC **OPEN-EYES** relies on the community for OS and libraries.

OPEN-EYES releases on github, in addition to the Linux drivers necessary to operate the system, described above, also open source SDKs for the development of specific applications, including:

KIOSK demo

Application in which OPEN-EYES-RPI exposes on display a website generated via nodeJS.

WEBRTC demo

Application in which OPEN-EYES-RPI communicates via WEB RTC protocol.

Base module connectors / module connectors

Ethernet connector

RJ45 PIN	SIGNAL NAME	TE CONNECTOR	10/100	10/100/1000
1	MD0+	8	TXP+	
2	MD0-	6	TXP-	
3	MD1+	4	RXP+	
4	MD2+	1		
5	MD2-	3		
6	MD1-	2	RXP-	
7	MD3+	5		
8	MD3-	7		
	SHIELD	9		
	SHIELD	10		
	SHIELD	11		

Power connector

RJ45 PIN	SIGNAL NAME	TE CONNECTOR	10/100	10/100/1000
1	MD0+	8	TXP+	
2	MD0-	6	TXP-	
3	MD1+	4	RXP+	
4	MD2+	1		
5	MD2-	3		
6	MD1-	2	RXP-	
7	MD3+	5		
8	MD3-	7		
	SHIELD	9		
	SHIELD	10		
	SHIELD	11		

Technical specification

Power

PARAMETER	VALUE
minimum input voltage	9 V
maximum input voltage	28 V
overcurrent protection	2 A (self-resetting fuse)
overvoltage protection	500V
polarity inversion protection	present

Consumption

CONDITION	MAX CURRENT
Power down	<100 mW
Full load	<5 W

Ambient condition

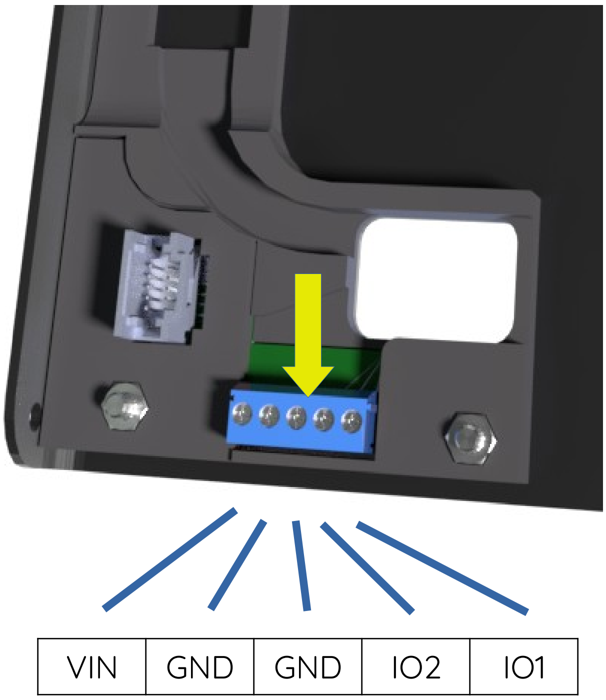
PARAMETER	VALUE
Operating Temperature	-10/+50 Celsius degrees
storage temperature	-20/+70 Celsius degrees
relative humidity	Max 90% non condensing

Installation

First of all, the back plate has to be fixed to the wall (by means of the two fixing screws) or to the articulated arm (by means of the 3/8 threaded hole).

Once you have finished attaching the back plate, proceed with connecting the LAN cable.

Then connect the power supply and any I/O signals to the appropriate terminals.

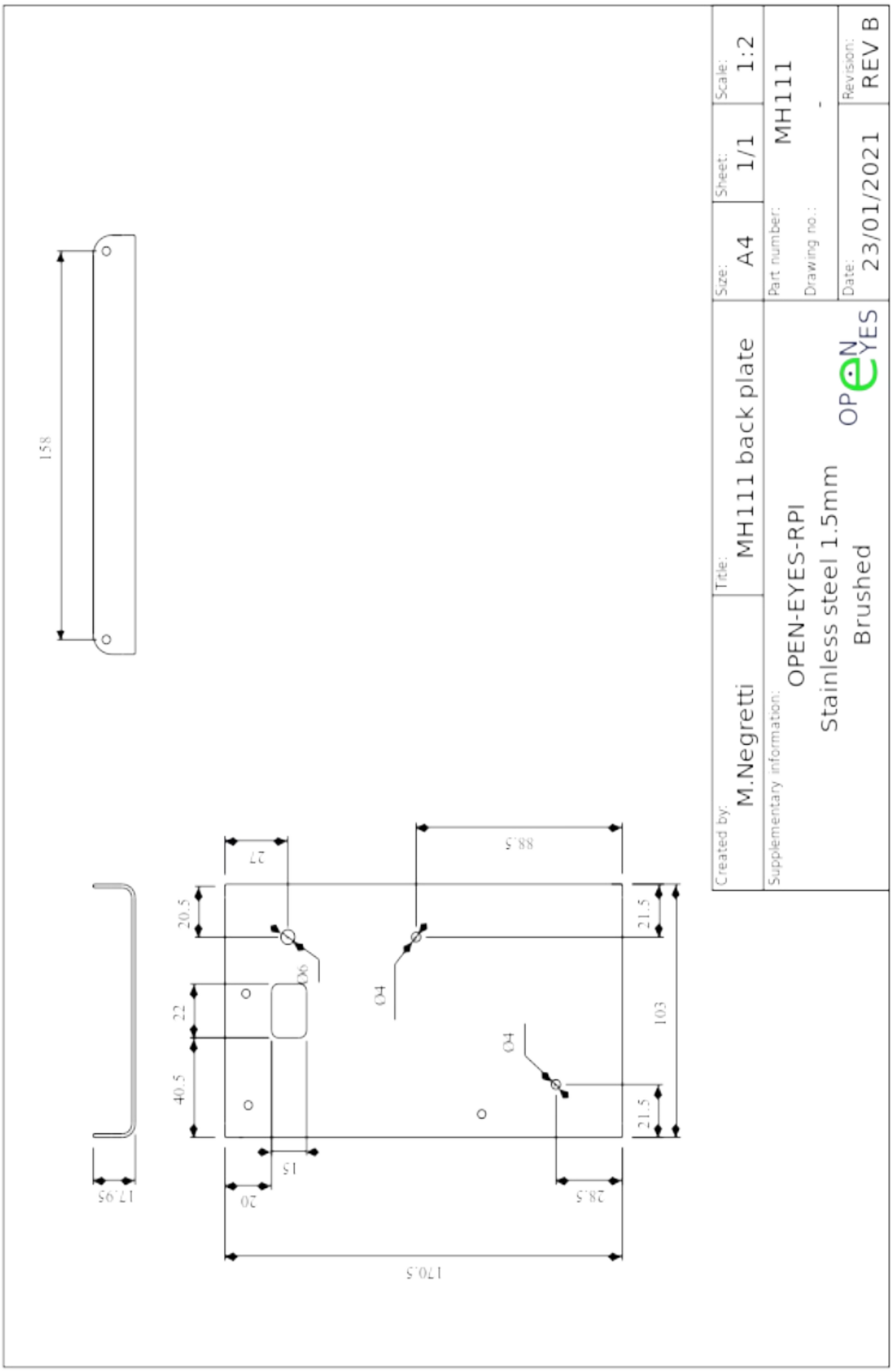


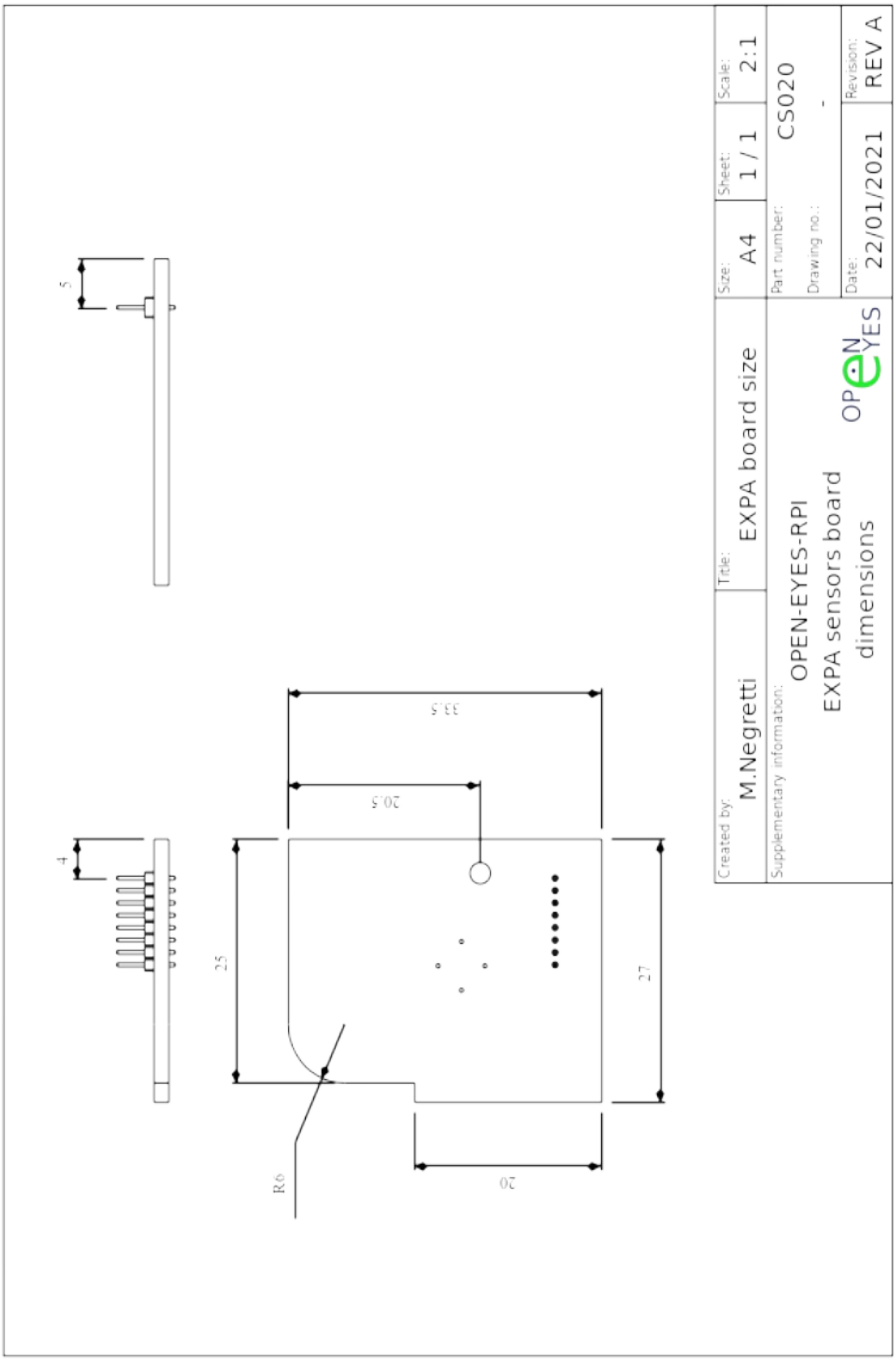
Finally, lock the cables by means of the special plate fixed with a screw.

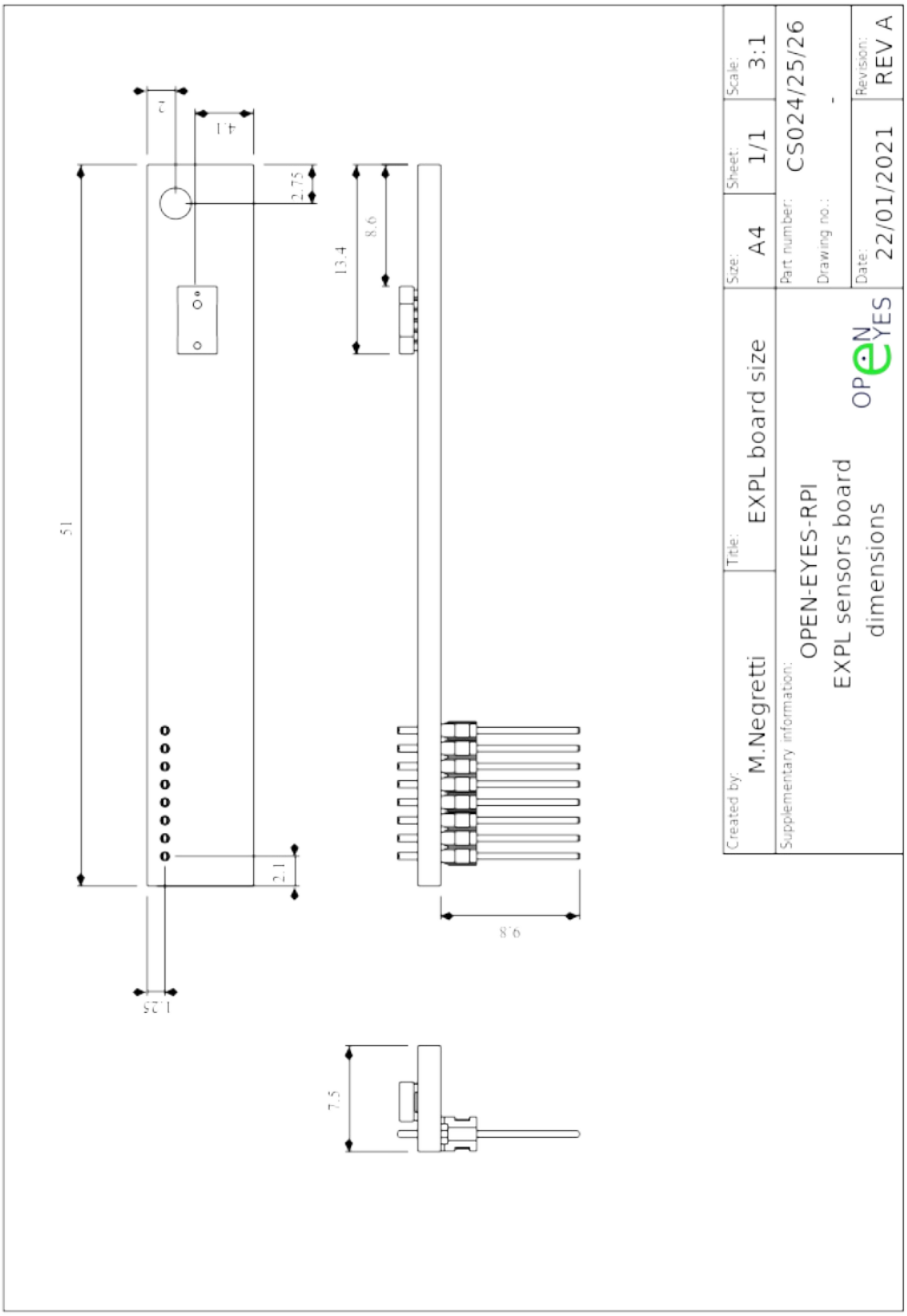
Mechanical characteristics

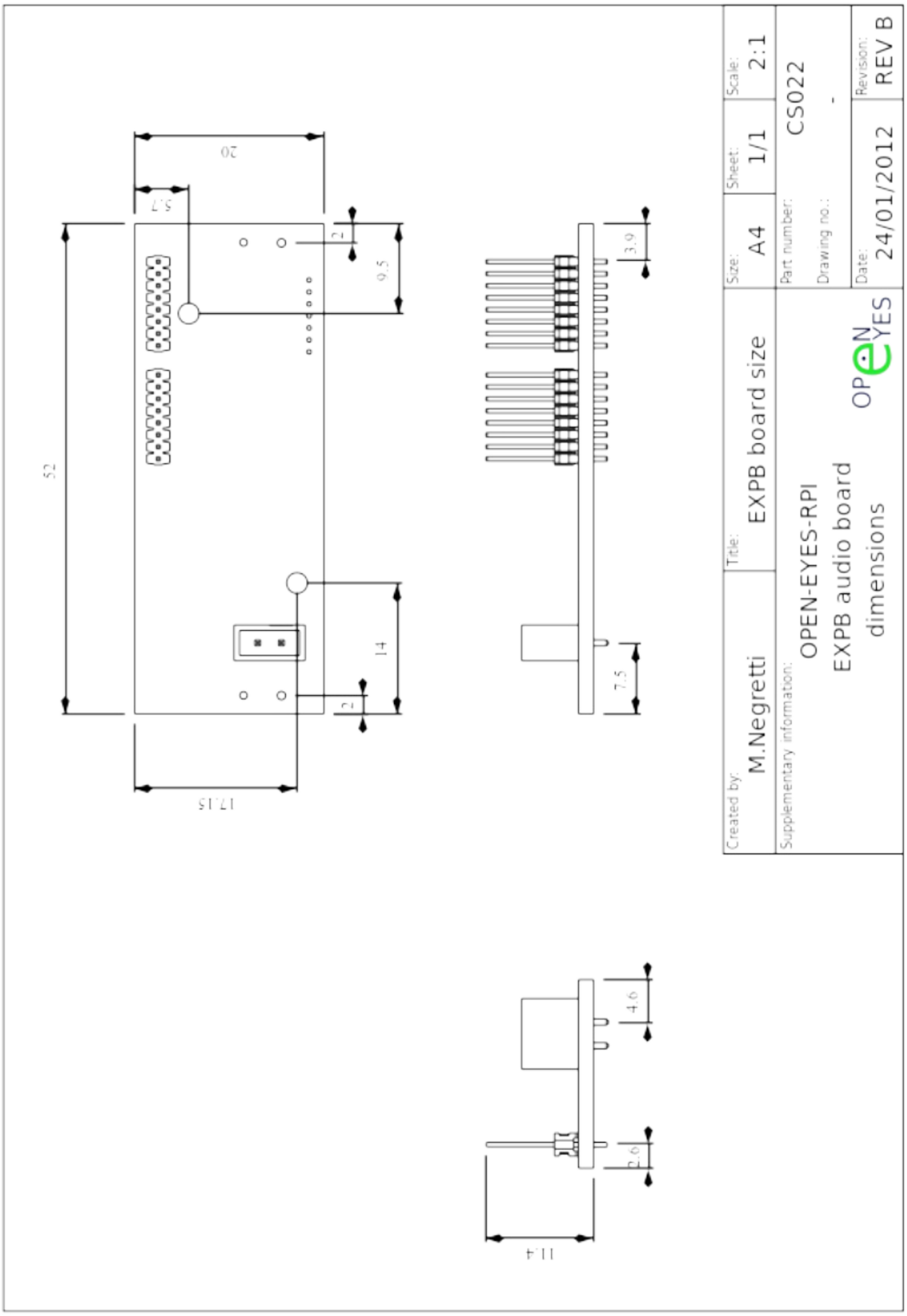
Technical sheets

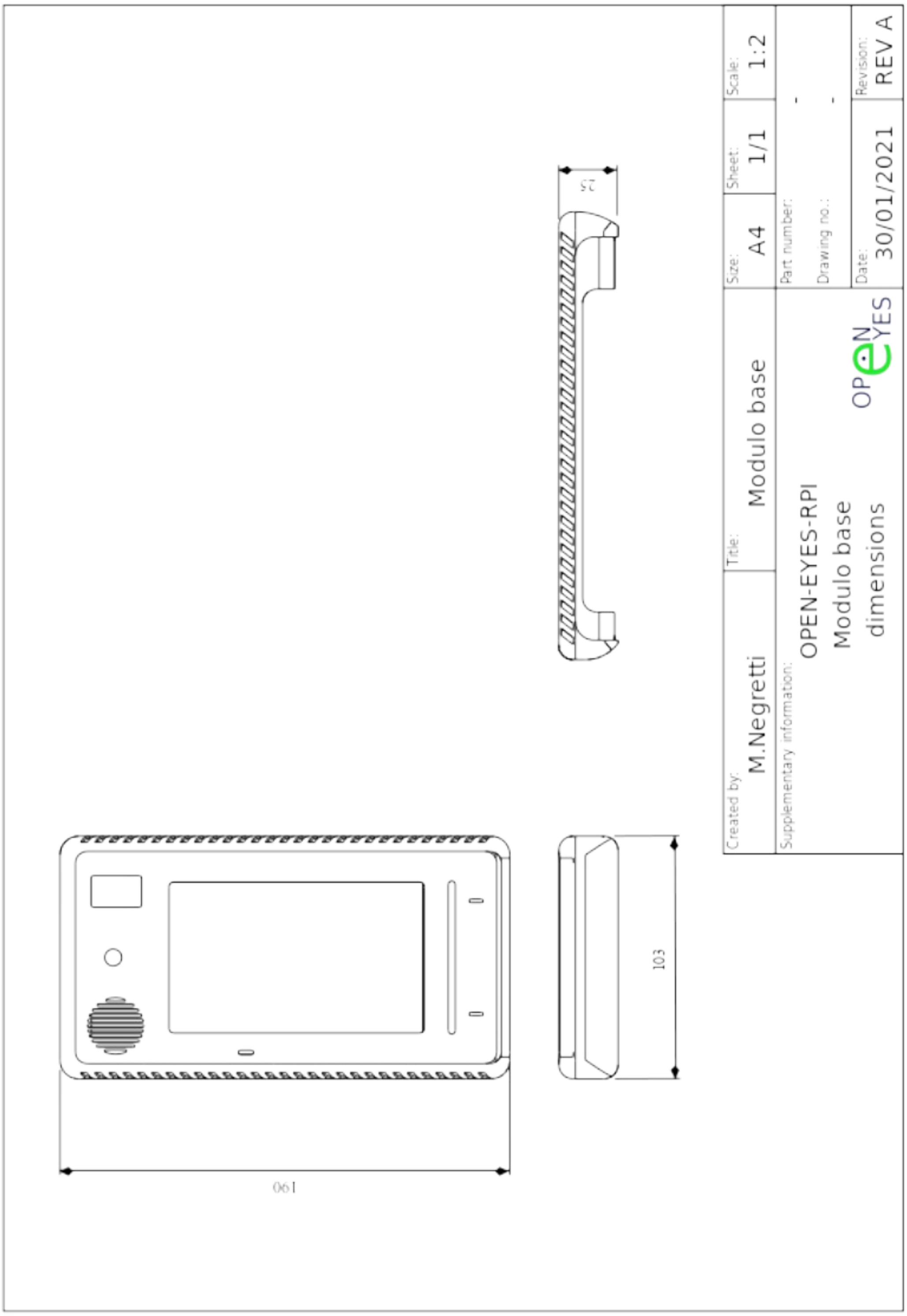
- ✗ MH111 back plate technical sheet;
- ✗ EXPA optional board dimensions technical sheet;
- ✗ EXPB optional board dimensions technical sheet;
- ✗ EXPL optional board dimensions technical sheet;
- ✗ BASE MODULE dimensions technical sheet;


Size: A4
Sheet: 1/1
Scale: 1:2
Part number: MH111
Drawing no.: -
Date: 23/01/2021
Revision: REV B
Title: MH111 back plate
Created by: M.Negretti
Supplementary information:
OPEN-EYES-RPI
Stainless steel 1.5mm
Brushed
OPEN EYES
158
27
88.5
20.5
21.5
Ø6
Ø4
22
Ø4
103
40.5
21.5
15
20
28.5
17.95
170.5


Created by:
M.Negretti
Title:
EXPA board size
Size:
A4
Sheet:
1 / 1
Scale:
2:1
Supplementary information:
OPEN-EYES-RPI
EXPA sensors board
dimensions
Part number:
CS020
Drawing no.:
-
Date:
22/01/2021
Revision:
REV A
33.5
20.5
25
27
20
R6
5
4


2
4.1
2.75
8.6
13.4
51
2.1
1.25
9.8
7.5
Created by:
M.Negretti
Title:
EXPL board size
Size:
A4
Sheet:
1/1
Scale:
3:1
Supplementary information:
OPEN-EYES-RPI
EXPL sensors board
dimensions
Part number:
CS024/25/26
Drawing no.:
-
Date:
22/01/2021
Revision:
REV A


Scale: 2:1
Sheet: 1/1
Size: A4
Part number: CS022
Drawing no.: -
Revision: REV B
Date: 24/01/2012
Title: EXPB board size
Created by: M.Negretti
Supplementary information: OPEN-EYES-RPI EXPB audio board dimensions
20
5.7
52
9.5
2
14
2
17.15
3.9
7.5
4.6
2.6
11.4


Created by:
M.Negretti
Title:
Modulo base
Size:
A4
Sheet:
1/1
Scale:
1:2
Supplementary information:
OPEN-EYES-RPI
Modulo base
dimensions
Part number:
Drawing no.:
Date:
30/01/2021
Revision:
REV A
190
103
25