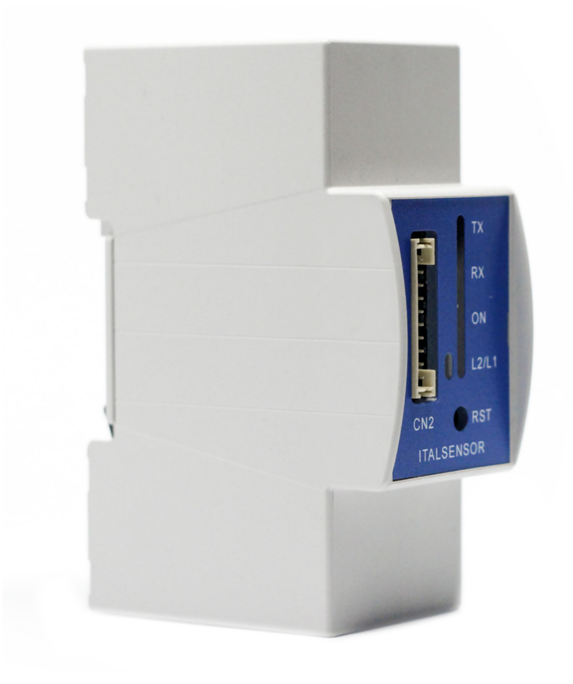
FIRST START UP AND GENERAL CHARACTERISTICS

ITALSENSOR teka

info@italsensor.com | www.italsensor.com

MyIoT it makes things easy

INDEX



MyIoT it makes things easy

FIRST START UP AND GENERAL CHARACTERISTICS

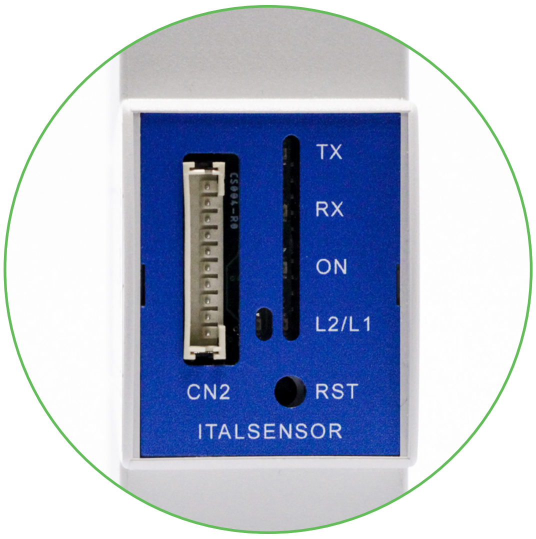
The module, on demand, may be provided already equipped with microSDHC memory card, with preloaded Raspbian "Stretch" operating system, and with the battery used by the RTC module.
To open the housing follow these guidelines.

- As first step, in order to open the housing, remove the DIN rail locking hook from the bottom case part of the housing.
- The bottom enclosure part will be gently removed from upper part of the housing making leverage by means of a flat head screwdriver
- Electronics is pulled out from housing.
- insert a CR1025 battery for RTC Module (Attention: take care to get negative battery terminal in touch with printed circuit board and positive battery terminal in touch with the battery housing otherwise the RTC circuitry will be damaged!)
- Insert the microSDHC into the appropriate slot (locking system).
- Insert the Raspberry Common Module into the appropriate slot.
- If additional output lines are used, pre-wired female connector has to be inserted in the expansion card and the upper panel of the housing should be pulled out.
  (ATTENTION/TAKE CARE: unlatch the expansion pcb card from main printed circuit board before inserting the output wired connector in order to avoid any mechanical damage of the card contacts due to pressure on the card itself)
- The module is inserted back in the housing, first pulling the cables (if used) through the opening at the top of the housing, then inserting and locking the screen panel/board
- Close the housing with the bottom part
- Insert the hook into the bottom case and lock the module on a DIN rail.
- Module is powered following connection diagram screened on housing wall.
- Initially the blue led (ON) located on the output board/card will start to flash (will flash also in case the module is supplied without inserting Compute Model board/card).
- After few seconds the blue led (ON) stops flashing, this means that boot process has been completed successful, Compute Module board/card can be therefore reach through a LAN connection (e.g. using WinSCP to manage File transfer to and from the module or using PuTTY for SSH connection and access to Compute Module console. With the WinSCP tool it is also possible to make SSH connection directly from the tool itself so no need to install also the PuTTY tool).
- RPICM3 module can be supplied with microSDHC e RTC battery preinstalled:
  - microSDHC 8 GB, class 10
  - Operating System Raspbian "Stretch"
  - Pre-configured static IP: 192.168.2.228 (file /etc/dhcpcd.conf)
  - SSH enable also for root (user: root, password: ital)
  - server web Apache2, PHP7 (v 7.0.33) and database MariadB (v 10.1.37) already uploaded and operating
  - Serial interface ttyAMA0 already set up to be directly used from Compute Module,
    output is on the RS485 isolated interface
  - RTC already set up and operating (CR1025, Li/MnO2 battery, already installed)
  - utility rpicm3extended preloaded and ready for immediate use of the module
  - examples of use (additional outputs, watchdog management, RS485 use, ...) written in Python
- Power supply voltage DC 9÷28V (protection against reverse polarity through resettable fuse)
- RS485 interface half-duplex from 1200 up to 115200 baud with automatic direction control and fail-safe mechanism
- n°1 Ethernet LAN 10/100 port conforming to IEEE802.3/802.3u standard
- n°2 USB2.0 ports
- n°1 internal micro USB port for programming eMMC memory for CM3 module (see the Raspberry official forum for details)
- n°8 output ports, not opto-isolated, with push-pull drivers, 3-state function can be activated on two groups of 4 lines, overtemperature protection, maximum permitted output current per channel 50 mA, maximum output frequency 50 kHz. Power supply for this interface must be applied and shall be equal to 5V, the 0V line is in common with the main module 0V and must necessarily, externally connected to the module itself.

ATTENTION/TAKE CARE: currently protection against reverse polarity on power supply is not implemented

It will be sooner available a push-pull interface with same basic features, but opto-isolated and with power supply range extended to 5÷30V.

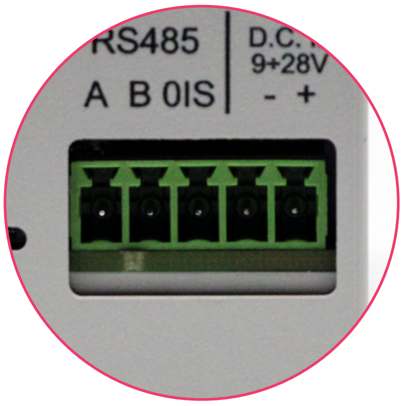
UPPER PANEL



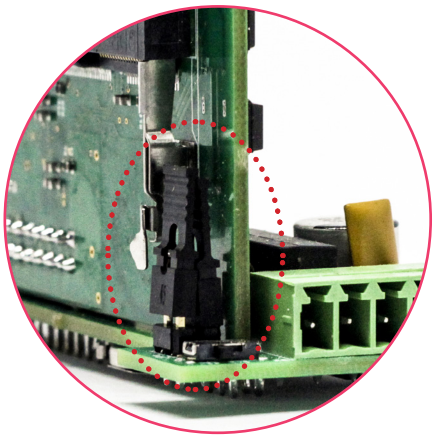
TX:	green led (data transmission on RS485)
RX:	red led (data reception on RS485)
ON:	blue led (Compute Module activity)
L1:	yellow led (user utility / use of consumer)
L2:	yellow led (user utility / use of consumer)
RST:	RST button access (see text)
CN2:	expansion connector with n°8 push-pull outputs (1...8) and dedicated power supply lines (pin 9:0V, pin 10:+5V)

POWER CONNECTOR / RS485 INTERFACE

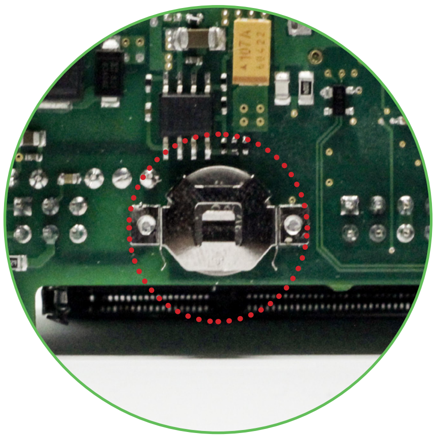
Power supply module connector and RS485 interface (0IS: isolate reference RS485 regarding the 0V of the module).The little opening at the left of the connector is an access to a switch that allows to insert (if moved toward the power connector) a terminator of 100 ohm between the RS485 A and B lines.
Inside the module, by means of 2 jumpers, is possible to turn on and off the 620 ohm pull-up and pull-down resistors on the A and B RS485 lines.



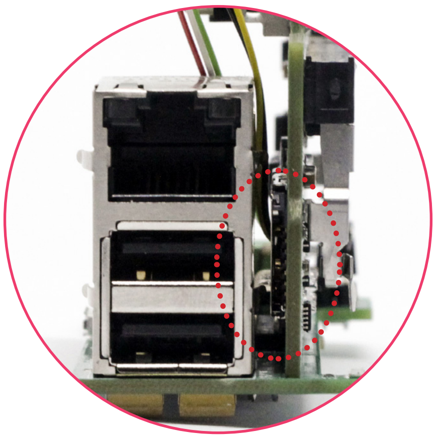
RS485 Interface Jumper Setting



RTC Battery



microSHDC



NOTES FOR FIRST USE

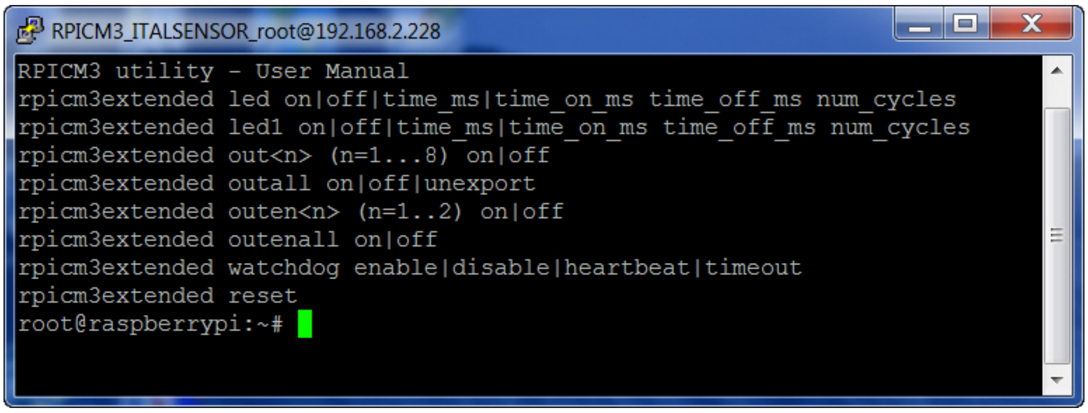
Within folder:

```
/home/test
```

There are some examples written in Python (verified/checked with 2.7.13 version) these can be used as starting point for writing of the own applications.

All the module functionalities can be managed through a dedicated system application which can be executed directly from the command line. In order to show the online guide type:

```
rpicm3extended
```


RPICM3_ITALSENSOR_root@192.168.2.228

```
RPICM3 utility - User Manual
rpicm3extended led on|off|time_ms|time_on_ms time_off_ms num_cycles
rpicm3extended led1 on|off|time_ms|time_on_ms time_off_ms num_cycles
rpicm3extended out<n> (n=1...8) on|off
rpicm3extended outall on|off|unexport
rpicm3extended outen<n> (n=1..2) on|off
rpicm3extended outenall on|off
rpicm3extended watchdog enable|disable|heartbeat|timeout
rpicm3extended reset
root@raspberrypi:~#
```

To manage outputs on expansion module there are the following options:

- led: turn (on) o turn (off) led L1
- led1: turn (on) turn (off) led L2
- out<n>: set to "1" (5V) or to "0" (0V) output line n (n=1...8)
- outall: set to "1" (5V) or to "0" (0V) all the output line
- outen<n>: enable (on) or disable (off) first or second group of the 4 output lines
- outenall: enable (on) or disable (off) all the outputs placing them on high impedance (3-state)

EXAMPLE PROGRAMS FOR VERIFYING THE FUNCTIONS OF THE MODULE

FILE (/home/test)	USAGE
testout.py	expansion interface outputs and led demo
tstGPIO18.py	set the GPIO18 line to high in order to activate the Common Module reset procedure after 60s
tstGPIO25.py	Read the GPIO25 line state (status of RST button)
tstHWWDT.py	Hardware watchdog usage example

Each demo can be interrupted during execution anytime by pressing the CTRL+C key combination. Major details are written into the comments contained in the files.

RST BUTTON

Activating or not RST button for less than 4s, the microcontroller will activate or deactivate respectively GPIO25 line (operating can be verified with the demo script tstGPIO25.py).
If RST is pressed for more than 4s blue led (ON) starts a sequence of 2 flash, if at the end of this period of time the button is released the microcontroller will wait for 60s, on expiry of this period the Compute Module reset procedure will start.
If RST button continues to be pressed for at least other 10s the blue led flashes quickly and the release of the button will cause the immediate reset of Compute Module.

SERIAL RS485 COMMUNICATION

Allowable baud rate between 1200 and 115200 baud. Protected and isolated interface with a fail-safe mechanism, the direction of transmission is automatically managed by microcontrollers present in the module. Two example files are available:

FILE (/home/test)	USAGE
tstserial_ma.py	Serial communication test on RS485 (master)
tstserial_sl.py	Serial communication test on RS485 (slave)

To execute the RS485 demonstration code wo modules must be interconnected.
The **tstserial_ma.py** script shall be executed on the first module, identified as master; this script start to carry out a transmission of a character or a series of characters included between a start and end code, after transmission it will wait for an answer from the slave module. The **tstserial_sl.py** script shall be executed on the second module, identified as slave; the script awaits and immediately send back (echo) the received character.
The connection between the two modules is made by wiring respectively pin A and B of the master module connector with the respective A and B of the slave module

To ensure a consistent transmission/reception you have to set the same baud rate value for both scripts.

A short help guide will be displayed if the example codes are launched without argument.

The RS485 serial transmission / reception sample scripts have been written to show how to handle binary data type, so they can be used as a reference not only for implementing ASCII-based data transmission but more generally for binary data transmission.

Example 1: continuous transmission of single value/character

- To transmit character “A” at the rate of 115200 baud with a pause of 1s use on master module the command:

```
python tstserial_ma.py 115200 65 65 1
```

- To activate reception on slave module use the command:

```
python tstserial_sl.py 115200
```

Example 2: continuous transmission on a range of values/characters

- To transmit at a rate of 115200 baud with a break of 100ms between subsequent sending of all the characters with ASCII code between 0 e 255 use on master module the following command:

```
python tstserial_ma.py 115200 0 255 0.1
```

- To activate reception on slave module use the following command:

```
python tstserial_sl.py 115200
```

RTC MODULE

RTC module (based on integrated circuit MCP79410) is an hardware clock separated from the system clock and have the purpose to retain date and time information even when the module is not powered. It interfaces with Compute Module through I2C lines of the module itself that are therefore reserved for this specific use.

To read data and time stored in RTC module write:

```
sudo hwclock -r
```

If you need to set date and time of the RTC equal to the system time (visible with command `date`), write:

```
sudo hwclock -w
```

SUMMARY OF THE GPIO PORTS AND EXPANSION CONNECTOR LINES

Some GPIO ports of Compute Module are used for specific purposes:

GPIO Compute Module	Direction	Function
GPIO2		Line (SDA) used for RTC module
GPIO3		(SCL) Line used for RTC module
GPIO23		Reserved: for future use
GPIO24		Reserved: for future use
GPIO14	OUT	TXD - Serial transmission line
GPIO15	IN	RXD - Serial reception line
GPIO16	OUT	1: led LD1 ON / 0: led LD1 OFF
GPIO34	OUT	1: led LD2 ON / 0: led LD2 OFF
GPIO25	IN	1:RST activated / 0: RST rest
GPIO18	OUT	1: activate the Compute Module reset procedure after 60”
GPIO22	OUT	1: activate the hardware watchdog functionality
GPIO27	OUT	Heartbeat signal for hardware watchdog, a change of state at least every 60”, is required
GPIO17	IN	set to high from microcontroller when, with the hardware watchdog activated (GPIO22), the microcontroller do not see at least one state change of the GPIO27 line within an interval time of 60s, after a further waiting of 60s the Compute Module will be reset (RUN signal forced to 0)
GPIO11	OUT	1: activate the 1,2,3,4 output lines / 0: force the output lines 1,2,3,4 in 3-state
GPIO21	OUT	1: activate the 5,6,7,8 output lines / 0: force the output lines 5,6,7,8 in 3-state

Internal GPIO ports

GPIO Compute Module	Direction	Function	Pin CN2
GPIO36	OUT	Set the output state of the line 1	1
GPIO20	OUT	Set the output state of the line 2	2
GPIO37	OUT	Set the output state of the line 3	3
GPIO35	OUT	Set the output state of the line 4	4
GPIO38	OUT	Set the output state of the line 5	5
GPIO13	OUT	Set the output state of the line 6	6
GPIO12	OUT	Set the output state of the line 7	7
GPIO26	OUT	Set the output state of the line 8	8
-	-	Power supply line of the expansion output card pin 9	9
-	-	Power supply line of the expansion output card +Vin=5V pin 10	10

Assignments for managing outputs interface and CN2 expansion connector

EXTENSION INTERFACE

The 0V power line relating the output interface is in common with the 0V line of the module, output drivers in this release are not optoisolated from the Common Module GPIO, but the interface, capable of working at 5V, is thermally protected, with a maximum current of 50mA per channel with a switching frequency up to 50 kHz.