ED1608 Application note #004

Application note for the ED2010 extension board

Product description for ED2010 I/O expander for ED1608 module

The ED1608 Sigfox/LoRa Module is developed as a general purpose module that is able to integrate functionality for various applications and communicate with the widely available Low Power Wide Area Networks (LPWAN) SigFox and LoRa.

There are many sensors and interfaces already present on the ED1608 itself, but there will always be applications that need sensors that are not implemented on the main PCB. In this case extension boards can be used to expand the ED1608 functionality with the required extra interfaces.
This document only describes the extra features/connections with the ED2010 board, for standard information see the "Technical Product Description" on our website: https://www.1m2m.eu/manuals/ .

Product feature overview

The ED2010 adds a number of functionalities to the ED1608.

1. A high precision dual 24 bit Analog to Digital converter that can be used for measuring very small signals. Examples are given for Pt100, Pt1000, load cells and a generic analog input.

2. An efficient power supply that can be used to use the device on a voltage of 6 to 28VDC instead of 3 to 5 VDC. This circuit also switches between battery and main supply depending on input voltage.

3. Two high side switches (2A, 5 to 28VDC) for inductive, capacitive and resistive loads like relais, lamps etc.

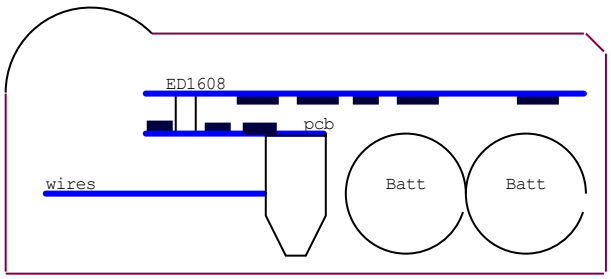
4. Two 3.3V universal digital ports with several pull-up, pull-down and pulse counting and generating options.

5. A 18 lead screw terminal for connecting sensors and relays.

All inputs and outputs are ESD protected.

Mechanical overview

The ED2010 is mounted on the ED1608 main board by a 8 pin connector and two double sided foam blocks. It offers a screw terminal interface, but at the expense of some battery space.
When mounted on the ED1608 PCB the 4 cell battery pack cannot be used inside the standard enclosure. Instead a 2 cell pack can be used for applications that require a battery backup or use little power.



ED1608 Application note #004

Application note for the ED2010 extension board

Connections

J3 Power supply and power outputs

1	VHigh	External power supply. 6.0 to 28VDC max Normally 12V to 24V
2	Ground	Ground for pin 1
3	AuxOut1	High power high side switch #1
4	Ground	Ground for pin 3
5	AuxOut2	High power high side switch #2
6	Ground	Ground for pin 5

J2 HiRes Analog inputs

1	Bridge P	For loadcell bridge positive *
2	Bridge A	For loadcell bridge A *
3	Bridge B	For loadcell bridge B *
4	Bridge N	For loadcell bridge negative *
5	AnIn3 P	Auxiliary analog input positive *
6	AnIn3 N	Auxiliary analog input positive *

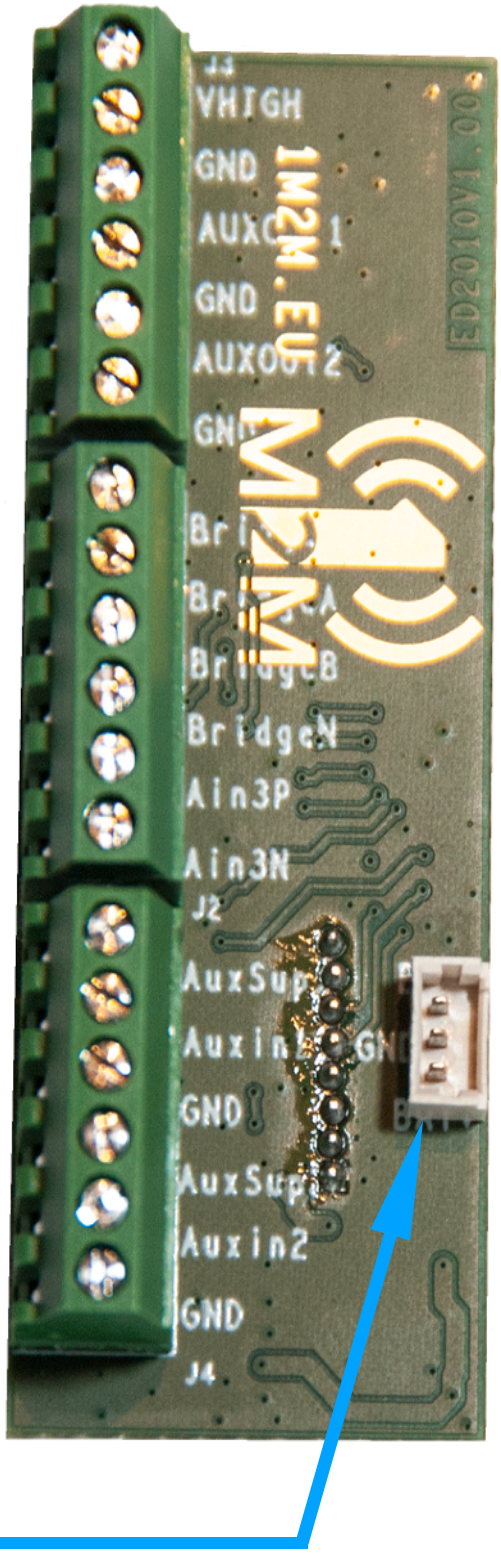
* depending on use case

J4 Digital inputs and sensor interface

1	AuxSup1	Supply (3.3V 20mA) for external sensor #1. Can be switched off
2	AuxIn1	Digital input #1. Has several options for pull up, pull down and pulse counting or debouncing
3	Ground	Ground for pin 1 and pin 2
4	AuxSup2	Supply (3.3V 20mA) for external sensor #2. Can be switched off
5	Auxin2	Digital input #1. Has several options for pull up, pull down and pulse counting or debouncing
6	Ground	Ground for pin 4 and pin 5

K1 Battery

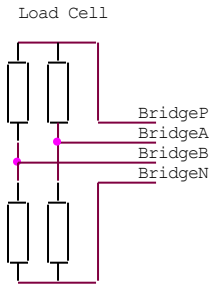
1	VBatt	Battery voltage 3.2 VDC min, 5.3 VDC max
2	Ground	Ground for pin 1
3	VBatt	Same as pin 1



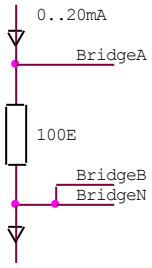
Interface description

HiRes analog inputs (Connector J2)

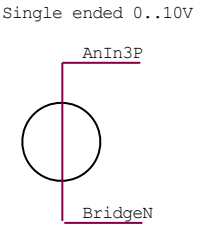
These inputs support several types of analog sensors. Below are some examples but many more configurations are possible. Please contact us if you need something else.



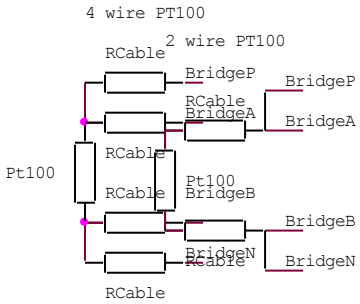
Load cell interface for measuring weights



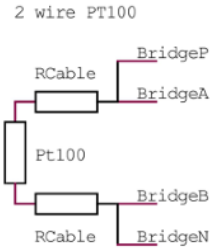
0 to 20mA interface



Single ended 0..10V interface



4-wire Pt100 or Pt1000 interface



2-wire Pt100 or Pt1000 interface

Digital Inputs and sensor interface (Connector J4)

This interface is highly configurable.

Either as 2 independent single digital sensor interfaces for use as switches, pulse detectors, frequency detectors, pulse duration, OneWire etc. or as a cooperating double digital interface like an external IIC bus or serial uart. Voltage levels are 3.3V.

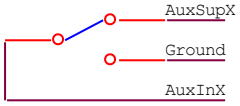
The AuxIn1 and AuxIn2 pins can be configured to pull up or pull down with resistors of 100K, 25K, 20K, 4K7 and 3K9. They can count pulses up to 1 MHz, or debounce noisy mechanical switches.

For maximum reduction of power consumption, a “level keep” function is available.

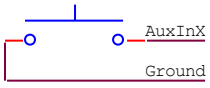
This pulls the pin down with 25KOhm after it was forced low, or up with 25KOhm after it was forced high.

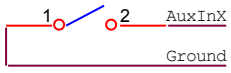
The effect is that a switch can be read reliably without any extra current consumption.

Example:

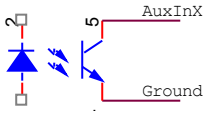


Some examples for Inputs:

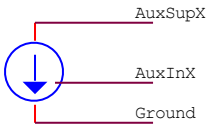




Push button Switch

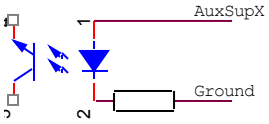


Optocoupler input

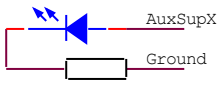


Flowmeter with pulse counter output (hall effect or mechanical).
AuxSupX supplies 3.3V at a maximum of 20mA and can be shut down if necessary.

Some examples for outputs

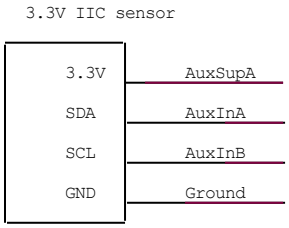


Optocoupler Output

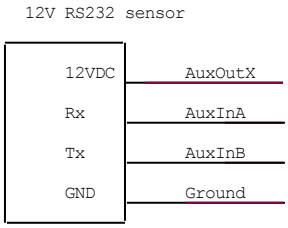


LED output

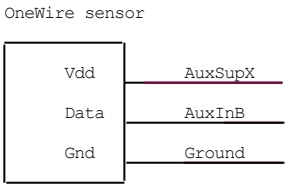
Some examples for serial interfaces



IIC based sensor, 3.3V power supply



RS232(TTL 3.3V) sensor with 12V power supply



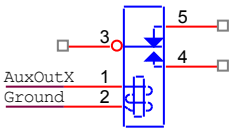
OneWire sensor with or without Vdd

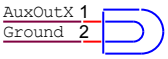
Power interface (Connector J3)

Two pins are for connecting an external power supply. Range is 6 to 28 VDC, normally 12V or 24V is applied. This voltage is converted to an internal 5V that supplies the ED1608. If this voltage is not available then the battery voltage is routed to the ED1608.

The power supply pin also powers the two auxiliary power outputs. These are high side switches that can route the (12V or 24V) power input to the AuxOut1 and AuxOut2 pins.
Maximum current is 2A and the outputs are protected against overload and over-temperature.

Examples :





Relay or valve

Lamp 12V/24V <2A

In case of power interruption, all analog interfaces, timers, counters and radio's will keep working normally as long as the battery is not drained.
The only exception are the power outputs. They use the external power supply, not the internal battery, so if there is no external supply they cannot output any voltage.

Messages for Load cell

Payload example: 9A2808FC00000000000A000A

Online json payload decoder:

https://www.1m2m.eu/services/getpayload?Company=1m2m&payload=9A2808FC00000000000A000A

RETURNS:

{
"MsgID":"Scale",
"Temp":"23.00",
"Humidity":"40",
"Weight":"0",
"DioCnt1":"10",
"DioCnt2":"10"
}

Uplink:

The GPS information is compressed in exactly the same way as in the alive and moving messages

Defines of the variables:

```
typedef struct {                //
    byte  MsgId;                //  MsgIDScale
    byte  Humidity;             //  humidity 0..99%
    int16 Temperature;          //  Temperature in 0.01 C
    int32 Weight;               //  raw value of analog voltage from
                                    ADC
    word  DioCnt1;              //  counter for DIO1
    word  DioCnt2;              //  counter for DIO2
  } TScaleMsg;                  //

#define MsgIDGenSens        (0x02) //
#define MsgIDScale          (0x9A) //

void SendScaleMessage()                                          //
{                                                                //
  TScaleMsg Msg;                                                 //
  Msg.MsgId       = MsgIDScale;                                  //
  Msg.Humidity    = RHDHumidity;                                 //
  Msg.Temperature = RevEndian16(RHDTemp);                        //
  Msg.Weight      = RevEndian32(RawLoadCell);                    //
  Msg.DioCnt1     = RevEndian16(Get2010DIO1Counter());           //
  Msg.DioCnt2     = RevEndian16(Get2010DIO2Counter());           //
  WrCStr(chMon,"\r\n");WrWhite("Scale message queued");          //
  SendMsg(&Msg);                                                 //
}
```

```
void SendGenSensMessage()                                          //
{                                                                  //
  TGenSensMsg Msg;                                                 //
  Msg.MsgId    = MsgIDGenSens;                                     //
  Msg.Status   = 0;                                                //
  Msg.BaromBar = RevEndian16(BaroPressure);                        //
  Msg.Humidity = RHDHumidity;                                      //
  Msg.Temp     = RevEndian16(RHDTemp);                             //
  Msg.VibAmp   = 0; // FFTMax1.Amplitude;                          //
                    // limited to 4095/16
  Msg.VibFreq  = 0; // FFTMax1.Freq;                               //
                    // 6.5 Hz per unit
  Msg.LevelX   = 0; // angle(GravX,GravY,GravZ);                   //
  Msg.LevelY   = 0; //angle(GravY,GravX,GravZ);                    //
  Msg.LevelZ   = 0; //angle(GravZ,GravX,GravY);                    //
  WrCStr(chMon,"\r\n");WrWhite("GenSens message queued");          //
  SendMsg(&Msg);                                                   //
}                                                                  //
```

```
typedef struct {       //
  byte   MsgId;        // Message Identification Value = 0x02
  byte   Status;       // Content Depends on Message ID
                       // ==for future use
  int16  BaromBar;     // Air Pressure in mBar
  int16  Temp;         // in 0,01 degC
  byte   Humidity;     // Relative Humidity in %
  int8   LevelX;       // Inverse Sinus of Spirit Level in Deg
                       // X-Dir -128 = -90 Degr .. +127 = +90 Degr
  int8   LevelY;       // Inverse Sinus of Spirit Level in Deg
                       // Y-Dir -128 = -90 Degr .. +127 = +90 Degr
  int8   LevelZ;       // Inverse Sinus of Spirit Level in Deg
                       // Z-Dir -128 = -90 Degr .. +127 = +90 Degr
  uint8  VibAmp;       // Amplitude of Vibration Detected == Future
  uint8  VibFreq;      // Approx. Frequency of Vibration Detected
                       // in Hz = Future
} TGenSensMsg;         //
```

For testing purposes there are web tools for coding and decoding available on:
https://www.1m2m.eu/webtools.php