
# PRESENTATION OF EMRB TESTING

This report present the results of the Environmental Monitoring Readout Board tests

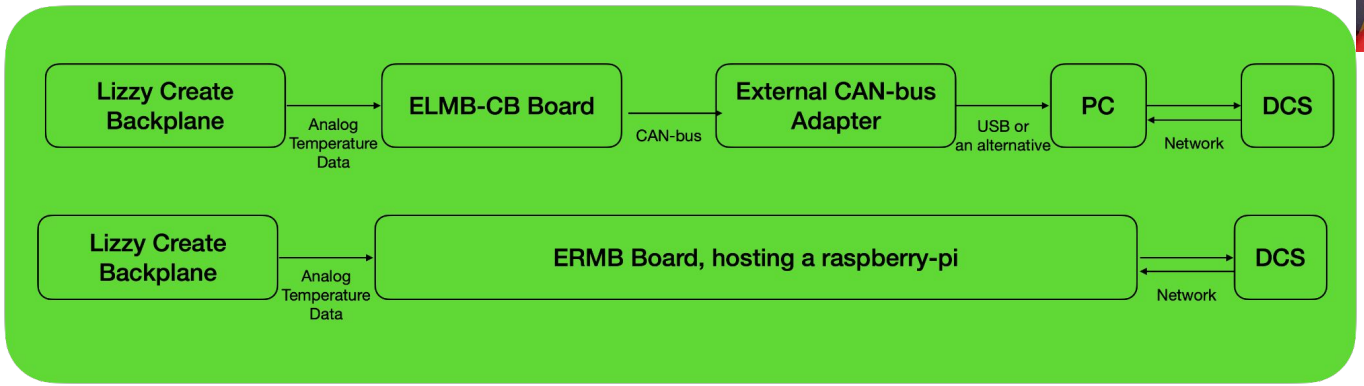
Realised by Valentin TIETZ
In Collaboration With:
Ismet Siral
Carlos Solans
Marcos Vazquez
David Margin Florez Rubio

# What's EMRB?

**Environmental Monitoring Readout Board**

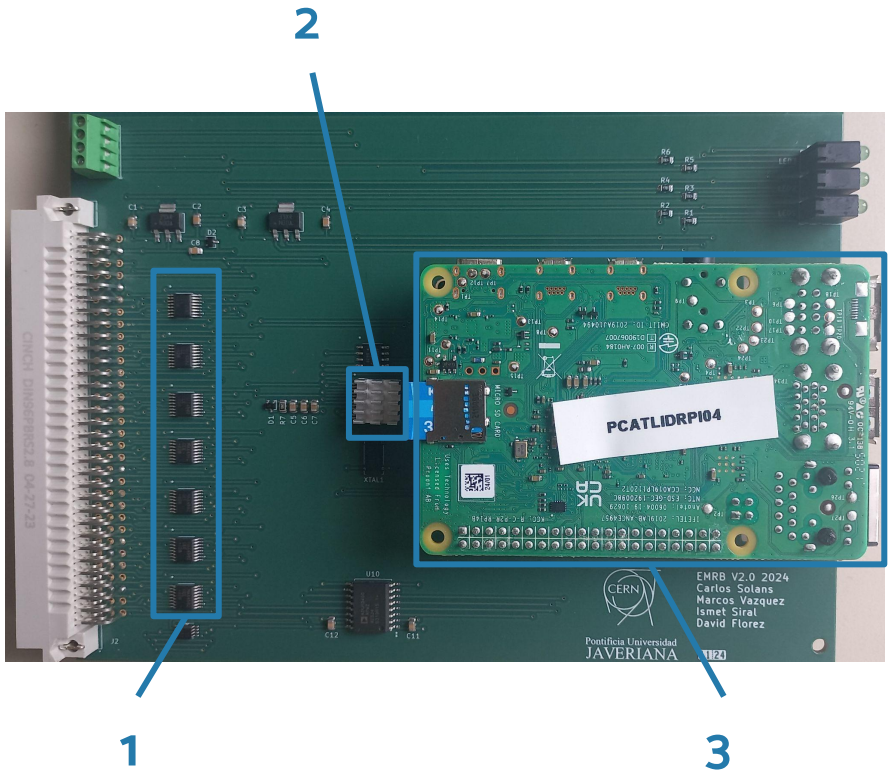
- EMRB was first introduced in: https://indico.cern.ch/event/1374100/contributions/5775078/attachments/2795842/4876959/ERMBBoard-Proposal.pdf

- The goal is to simplify the readout of the Lizzy creates, by replacing the ELMB

# Functioning

1. Temperature value measured and transmitted to the input of one of the EMRB multiplexers
2. The ADC converter reads the values from the multiplexers and converts the analog value into a digital value
3. The converted value is transmitted by the A/D converter to the Raspberry PI
4. The Raspberry PI then runs a server that can directly interface with the DCS



# SPI communication testing

**We select multiplexer 0 and ADC channel 3**

**We run a program to read the pin value every second.**

**Results on the right**

- First : registers configuration
- Second : We increase the value of the voltage from 0V to Vmax
- Third : We decrease it to 0V

```
itkdcs@pcatlidrpi04:~/emrbsw/emrb/src $ test_ermb_board
Just sent 0xFF..FC -> ffffff
just set the Reset system bit to 1 -> ffffff
just set the Reset system bit to 0 -> ffffff
Attempting read config register: 40
Setting config registers after init ffffff
Attempting read config register: 7000
Communication Check: Reading Config Register 7000
Mult: 0 Chn: 3 ADC Reading: 0
Mult: 0 Chn: 3 ADC Reading: ffffe9
Mult: 0 Chn: 3 ADC Reading: e9
Mult: 0 Chn: 3 ADC Reading: e9
Mult: 0 Chn: 3 ADC Reading: 1fa8e8
Mult: 0 Chn: 3 ADC Reading: 1c87e8
Mult: 0 Chn: 3 ADC Reading: 5af5e8
Mult: 0 Chn: 3 ADC Reading: a2d0e8
Mult: 0 Chn: 3 ADC Reading: e618e8
Mult: 0 Chn: 3 ADC Reading: ffffe9
Mult: 0 Chn: 3 ADC Reading: ffffe9
Mult: 0 Chn: 3 ADC Reading: fc93e8
Mult: 0 Chn: 3 ADC Reading: b44be8
Mult: 0 Chn: 3 ADC Reading: 9cd4e8
Mult: 0 Chn: 3 ADC Reading: 7dc8e8
Mult: 0 Chn: 3 ADC Reading: 7cd3e8
Mult: 0 Chn: 3 ADC Reading: 573de8
Mult: 0 Chn: 3 ADC Reading: 2841e8
Mult: 0 Chn: 3 ADC Reading: e9
Mult: 0 Chn: 3 ADC Reading: e9
```

Registers Configuration

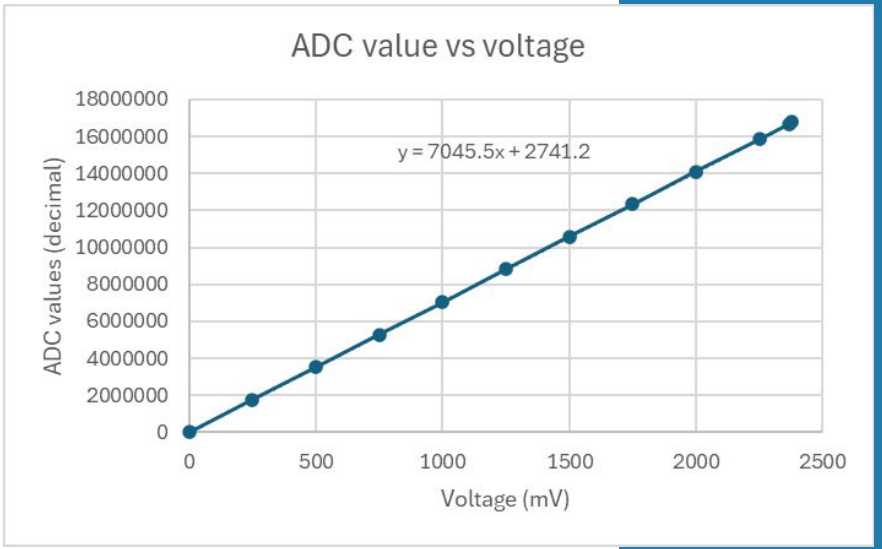
Voltage increase

Voltage decrease

# Conversion test linearity

**We verify that the ADC values as a function of input voltage is a linear function**

| Voltage (mV) | ADC values (hex) | ADC values (dec) |
|---|---|---|
| 0 | E9 | 233 |
| 250 | 1AE4D6 | 1762519 |
| 500 | 35C8F3 | 3524851 |
| 750 | 50AC8B | 5287051 |
| 1000 | 6B8E5B | 7048796 |
| 1250 | 8674D0 | 8811729 |
| 1500 | A158DA | 10574042 |
| 1750 | BC3C4D | 12336205 |
| 2000 | D7200E | 14098446 |
| 2250 | F20138 | 15860024 |
| 2370 | FE84E9 | 16680169 |
| 2380 | FFFFE9 | 16777193 |



# Pins readout testing

When we read several multiplexers one by one, we can observe that if we inject a voltage value on one of them, the next will also read a value. In the example below, we inject a voltage on multiplexer 0 on channel 0, and we can see that multiplexer 1 also returns a value other than ffffeb (maximum voltage value).

This problem is probably due to the software and we need to investigate it.

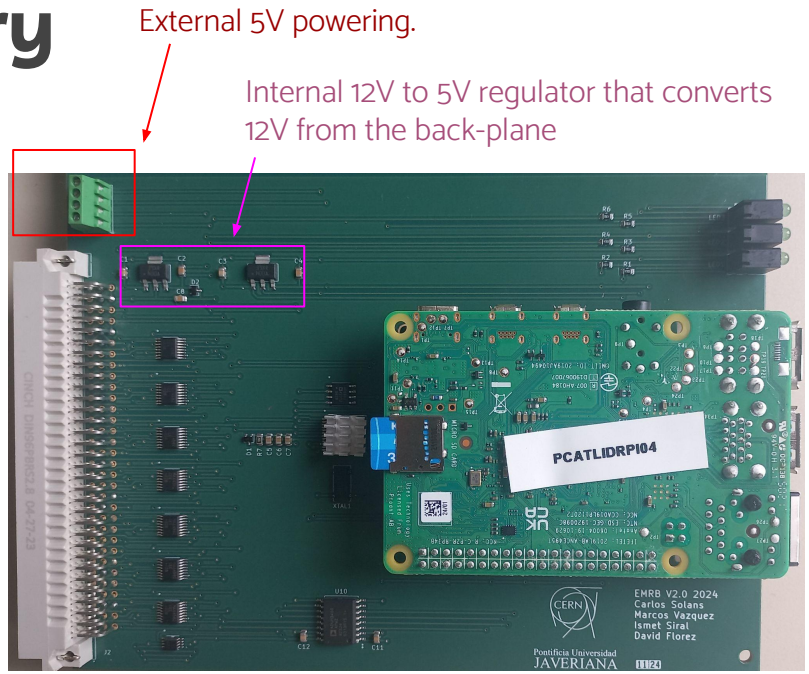
```
Mult: 0 Chn: 0 ADC Reading: d84aea
Mult: 1 Chn: 0 ADC Reading: 7899e8
Mult: 2 Chn: 0 ADC Reading: ffffeb
```

# Issues with Power Delivery

**The last remaining issue we have at hand is power delivery**

- We are currently powering this board externally using 5V
- We have noticed that that if we power the board with 12V via the backplate there is some power issues.

- This needs to be further investigated before we start mass-producing and populating these boards



# What's next?

1. **Understanding pin selection configuration**

(For the moment, whatever ADC channel is selected, we only read the B-side values, and Mult channel selection only works for 3-firsts multiplexers)

2. **Check that each multiplexer output behaves in the same way**

3. **Solving Power Delivery issues**

4. **Start building the OPC server so that we can start transferring the the readout out ADC values out.**

# Backups

# Pin selection mapping

**We carry out the multiplexer pin selection**

3 first multiplexers example

| Multiplexor | Mult_channel | SEL_4 | SEL_3 | SEL_2 | SEL_1 | ADC_channel |
|---|---|---|---|---|---|---|
| U1 | SCB_A_1 | 0 | 0 | 0 | 0 | 00 |
| | SCB_A_2 | 0 | 0 | 0 | 1 | 00 |
| | SCB_A_3 | 0 | 0 | 1 | 0 | 00 |
| | SCB_A_4 | 0 | 0 | 1 | 1 | 00 |
| | SCB_B_1 | 0 | 0 | 0 | 0 | 10 |
| | SCB_B_2 | 0 | 0 | 0 | 1 | 10 |
| | SCB_B_3 | 0 | 0 | 1 | 0 | 10 |
| | SCB_B_4 | 0 | 0 | 1 | 1 | 10 |
| U2 | SCB_A_5 | 0 | 0 | 0 | 0 | 01 |
| | SCB_A_6 | 0 | 0 | 0 | 1 | 01 |
| | SCB_A_7 | 0 | 0 | 1 | 0 | 01 |
| | SCB_A_8 | 0 | 0 | 1 | 1 | 01 |
| | SCB_B_5 | 0 | 0 | 0 | 0 | 11 |
| | SCB_B_6 | 0 | 0 | 0 | 1 | 11 |
| | SCB_B_7 | 0 | 0 | 1 | 0 | 11 |
| | SCB_B_8 | 0 | 0 | 1 | 1 | 11 |
| U3 | SCB_A_9 | 0 | 1 | 0 | 0 | 00 |
| | SCB_A_10 | 0 | 1 | 0 | 1 | 00 |
| | SCB_A_11 | 0 | 1 | 1 | 0 | 00 |
| | SCB_A_12 | 0 | 1 | 1 | 1 | 00 |
| | SCB_B_9 | 0 | 1 | 0 | 0 | 10 |
| | SCB_B_10 | 0 | 1 | 0 | 1 | 10 |
| | SCB_B_11 | 0 | 1 | 1 | 0 | 10 |
| | SCB_B_12 | 0 | 1 | 1 | 1 | 10 |

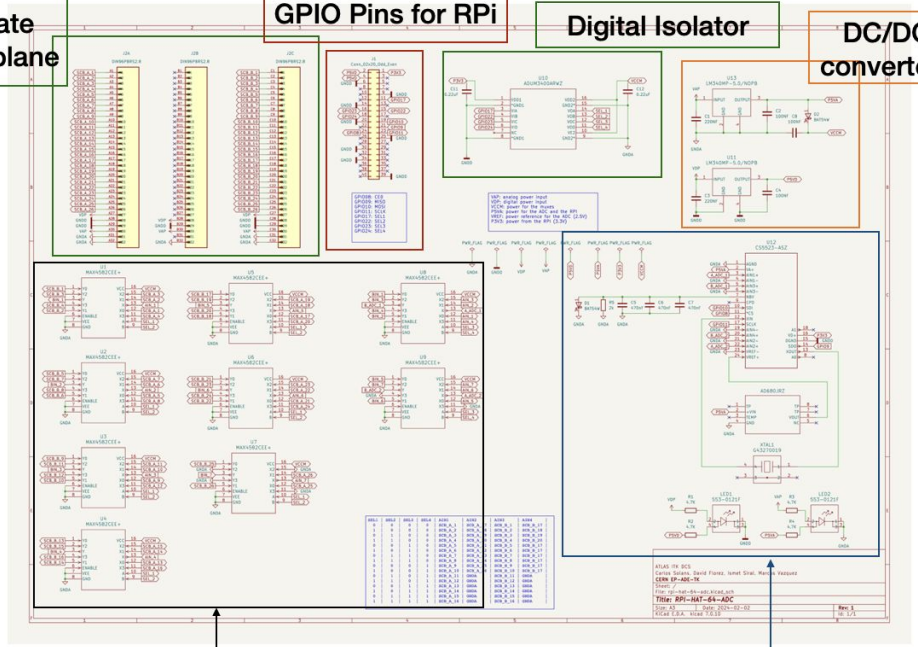
# The Design of the ERMB Board

Create Backplane

GPIO Pins for RPi

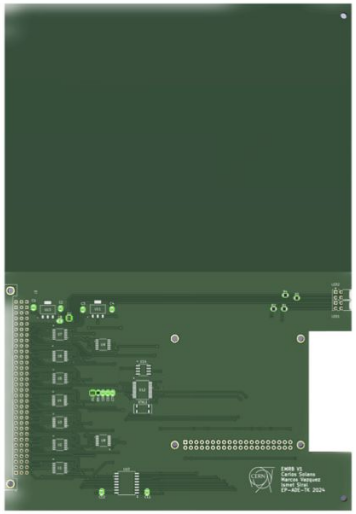
Digital Isolator

DC/DC converters



Multiplexers for choosing the different Analog data-lines

ADC System



The basic idea behind the board is:

- Use multiplexers to select different analog signals from the create backplane
  - Controlled by the RPi
- Use the ADC system to read/sample the analog channels.
- Using SPI protocol, readout the digitized values with RPi.
- Interface with the DCS system via network on RPi

*Old Design

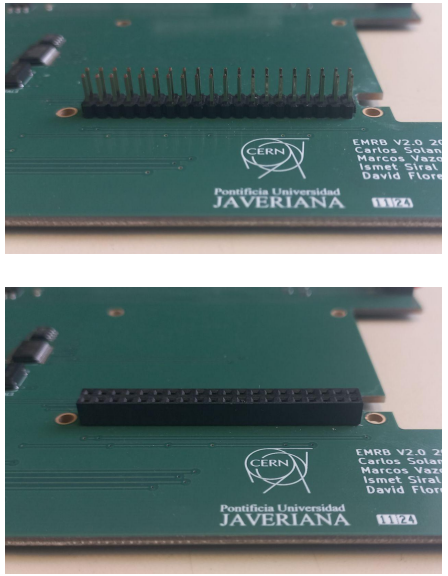
# SPI communication fixing

**Check MOSI / MISO pins on the Raspberry PI**

- MOSI signal emitted and received by ADC convertor
- No MISO signal

**Check continuity of ADC tracks and power supplies**

**Board change / soldering a new Raspberry connector**



# SPI communication fixing

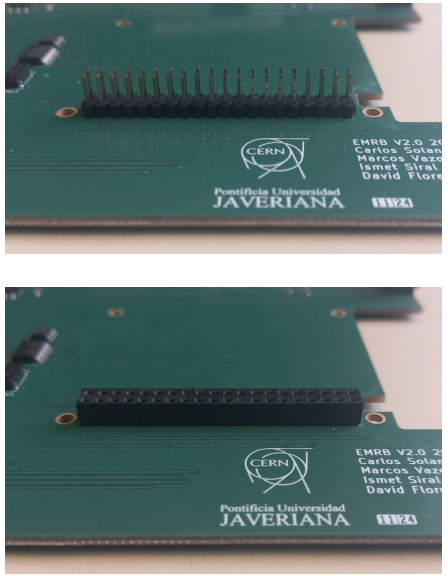
**Check MOSI / MISO pins on the Raspberry PI**

- MOSI signal emitted and received by ADC convertor
- No MISO signal

**Check continuity of ADC tracks and power supplies**

**Board change / soldering a new Raspberry connector**

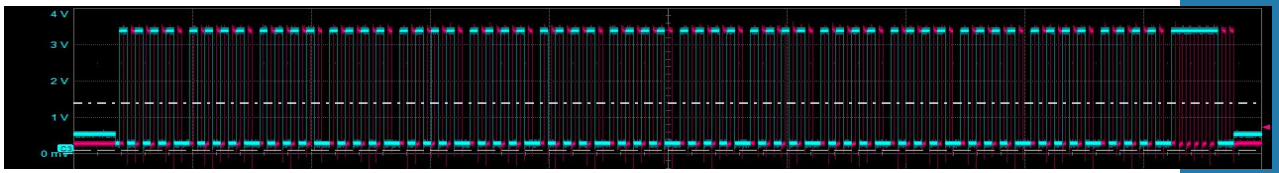
**Miracle, it works!!**



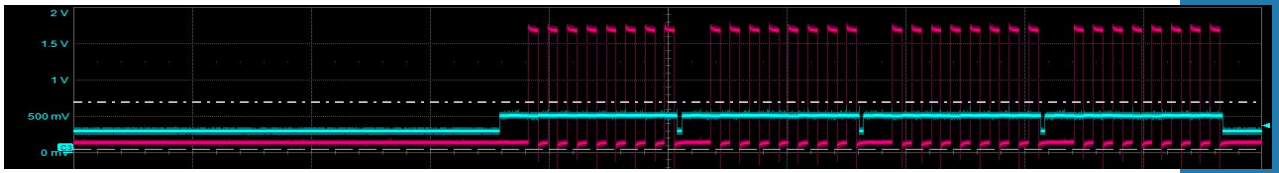
# Fixing SPI communication

**Check MOSI / MISO pins on the Raspberry PI**

- MOSI signal sent and received by ADC converter
- No MISO signal




*MOSI (blue) and CLK (pink) signals (15 * oxAA + FC)*




*MISO (blue) and CLK (pink) signals*

# SPI communication fixing

**Check MOSI / MISO pins on the Raspberry PI**

- MOSI signal emitted and received by ADC convertor
- No MISO signal

**Check continuity of ADC tracks and power supplies**

# What's next?

1. **Understanding pin selection configuration**

(For the moment, whatever ADC channel is selected, we only read the B-side values, and Mult channel selection only works for 3-firsts multiplexers)

2. **Understand why eight MSB = e9 or e8**

3. **Check that each multiplexer output behaves in the same way**

4. **Start building the OPC server so that we can start transferring the the readout out ADC values out.**