Raspberry Pi for Slow Control at the beam telescopes.

Manuel Morgado, University Simon Bolivar, Venezuela.

September 7, 2016

Abstract

This report contains a general review about the summer project on DESY in the summer school of 2016. Test Beam Telescopes are an invaluable tool for the improvement and development of the silicon detectors for experiments in High Energy Physics as ATLAS at LHC. In order to bring more information about the conditions of each test, it will be implemented a slow controller hardware which is responsible to measure different conditions in the test beam. This is achieved by implementing two sensors (BMP180 and SHT21), connected to a Raspberry Pi 2 (+B) which is also connected to the data acquisition system of the telescope (EUDAQ).

Contents

1 Introduction

The beam telescopes are fundamental for the research and the development of projects which studying the performance of a prototype sensor whose purpose is the position sensitive particle detection. Which is often used for medical applications, detectors with a high radiation tolerance and high resolution and low material requirements.

The guidelines for the development of the EUDET project established in the 6th framework programme of the EU Integrated Infrastructure Initiative, allows to have a huge range of different setups using a beam telescope, which consist of six pixel detector planes equipped with fine-pitch MIMOSA 26 sensors, a mechanism for precise positioning of the device under test (**DUT**), a Trigger Logic Unit (**TLU**) which provide trigger capabilities and a data acquisition system (**EUDAQ**).

These telescopes are designed to sense to keep the best performance in the track resolution even at the rather low particle energies of up to 6 GeV at the DESY test beam facilities.

2 Beam Telescopes

Some environment parameters as the temperature, humidity and pressure are important at the moment of characterizing new sensors in the test beam telescope. This could be accomplished through the use of the slow control devices. This project will make a first approach to integrate the Raspberry Pi as slow control device in the EUDAQ framework and also in the Telescope hardware.

2.1 EUDET-type Beam Telescope

In the figure 1 we can see the beam telescope area (DATURA beam telescope with the big red magnet) where all hardware and software is launched. Also there are some points where the parameters (Humidity, Temperature and Pressure) can be measured (outside of the bench, between planes and between planes and DUT).

2.2 EUDAQ Framework

The architecture of EUDAQ consists in a set of interconnected processes. *Run Control*(RC) asks for addresses to *Data Collector*(DC) and *Log Collector*(LC) in order to report it to the producers and the monitor. Then the producers send its data to DC and the log data to the LC. The DC handles and merges the data from the producers and then is saved in a file and also is shown on the Monitor. This is shown in figure 2

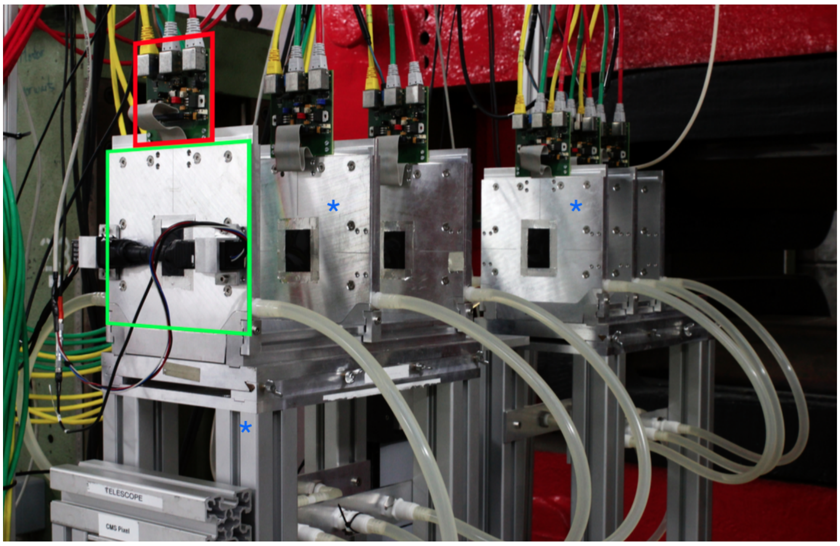


Figure 1: *DATURA beam telescope. Possible places to measure environment conditions (blue asterisks), aluminum frames (green) and auxiliary boards (red).*

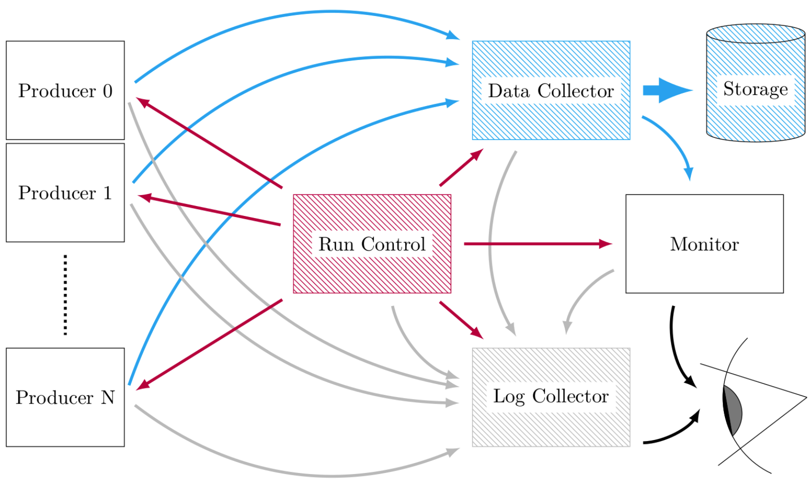


Figure 2: *EUDAQ framework scheme.*

3 Raspberry Pi and Implementation

The main goal is to obtain data from the sensor using the Raspberry Pi and then integrate it with the EUDAQ framework in the sense of all data can be merge in a single file.

In figure 3 it is shown how is the general set up of the slow control. In the next sections, each component will be explained.

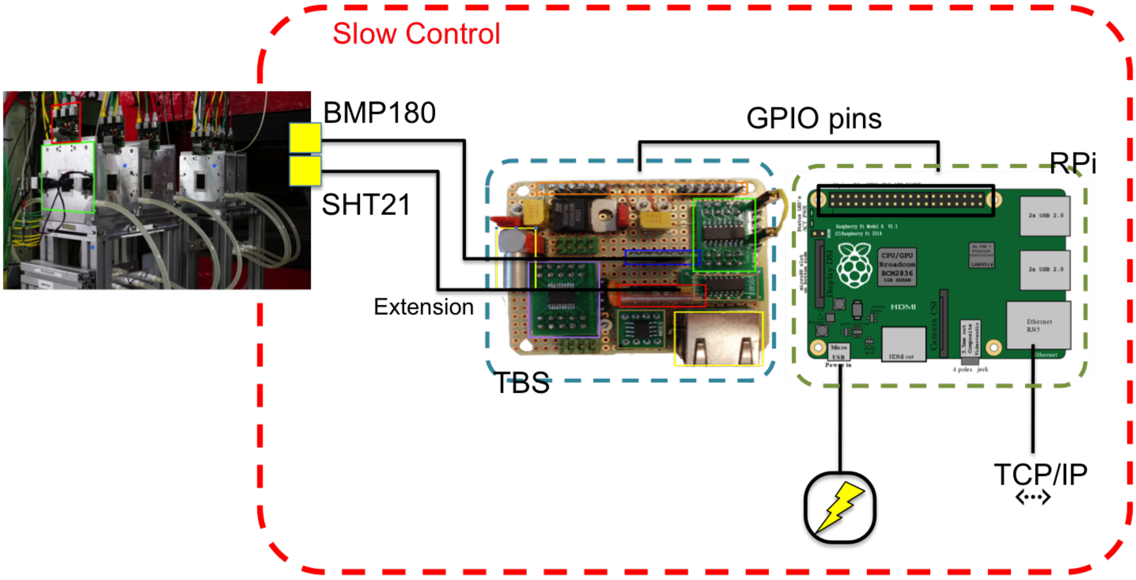


Figure 3: *General overview of the hardware in the testbeam.*

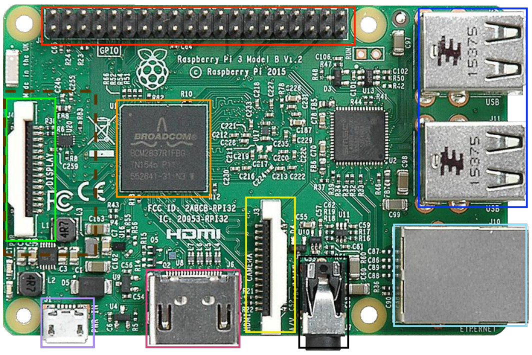
3.1 Hardware

The Raspberry Pi (**RPi**) is a device which is capable to plug to different peripherals as mouse, screen and keyboard, in a few words is a tiny computer (credit card size) [1]. That is also pretty nice, because could be possible to use this feature in order to connect more peripherals and more components that enrich the data to the user.

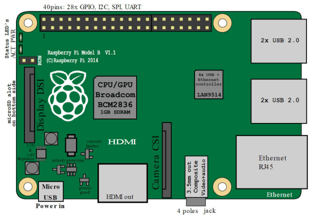
RPi 2 model B was the one used for this project and it has 4 USB, 1 micro-usb, 1 Ethernet, 1 HDMI, 1 combined 3.5mm audio jack and composite video ports also 29 *General Purpose Input Output* (**GPIO**) pins, *Micro SD* card slot, *Camera interface* (**CSI**) and *Display interface* (**DSI**). The processor consist in a 900MHz quad-core ARM Cortex-A7 (BCM 2836 32bit) with 1GB of RAM. (See figure 4)

There are exist many *OS* for the raspberry board, however we decide to install *Raspbian* [2]. So, in order to start to work with the raspberry without any OS already installed it is recommended to follow the next tutorial. In the case, the raspberry has already a *OS* that should not appear any issue to use the procedures presented in the previous section.
Two sensors (SHT21 and BMP180) are connected to the GPIO ports 3 and 2 (9 and 8 respectively in WiringPi library notation) and they can be used through I^{2}C protocol connection (Inter Integrated Circuit or I^{2}C is a serial computer bus developed in order to communicate with devices and peripherals of lower speed) through some libraries in C^{++}

[1]More information about Raspberry Pi:`https://www.raspberrypi.org/help/faqs/#introWhatIs`
[2]More information about Raspberry Pi *OS*:`https://www.raspberrypi.org/downloads/`



(a) Raspberry 3 Model B photo. Each color correspond to the same components of the RPi that we used for the slow control.

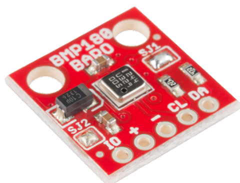


(b) Raspberry 2 Model B schematic.

Figure 4: Caption place holder

(wiringPi) and python (SMBus). The sensor SHT21 (figure 5a) is capable of measure temperature and humidity and the sensor BMP180 (figure 5b) is capable to measure temperature and pressure. In this sense, is important to communicate with each one in order to obtain data of the three parameters (Humidity, Temperature and Pressure) in the test beam area.

(a) *Sensor SHT21* (Humidity & Temperature). 6 GPIO pins connections.



(b) *Sensor BMP180* (Pressure & Temperature). 5 GPIO pins connections.

Figure 5: Sensors

Also it was used a extra board (see figure 6) which was developed in DESY by Torsten Külper for communicate the RPi with EUDAQ (Test Beam Shield, **TBS** in short) and implement it as a Slow control (Slow producer), more details can be read it in the documentation [3].

3.2 Software

As it was said before the RPi is basically a computer, therefore it need an operative system as *Raspbian*, also in order to control and get data from the sensors, it is necessary to use some libraries, depending of which language will be used to control it. In the case of C^{++} the library is called *WiringPi*, whereas for Python, it is necessary to lead the *SMBus* and *RPi.GPIO* libraries. (For more technical detail information read

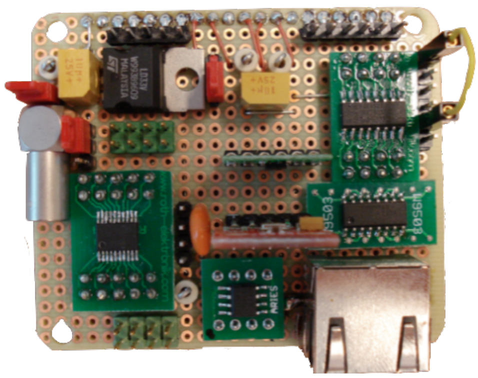


Figure 6: *Add-on board (top view). TBS is connected with RPI via GPIO port.*

the documentation [1].)

For each sensor, it was developed a couple of scripts (Python and C^{++}), here the pseudocode for each sensor. For SHT21 the pseudocode is presented in the *Algorithm Nr. 1*
This algorithm build the SHT21 class by example that is represented in the scheme of the figure 7,

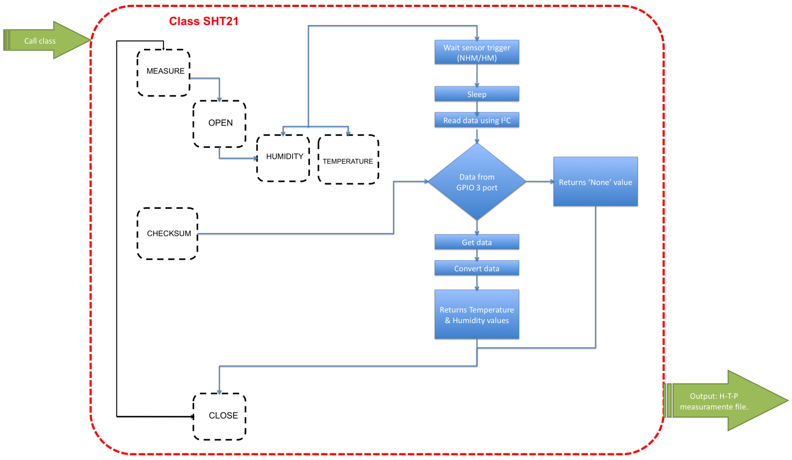


Figure 7: *Pseudocode* for sensor SHT21.

Note: all codes are in this github repository `https://github.com/manuelmorgado/RPi_HTP.git`

4 Results.

In order to show the sensor is getting data, we measure the temperature in two sites. One was with the sensor attached directly at the board and the other one was using a cable extension developed by my self. The difference can be noticed very easily in figure 8.

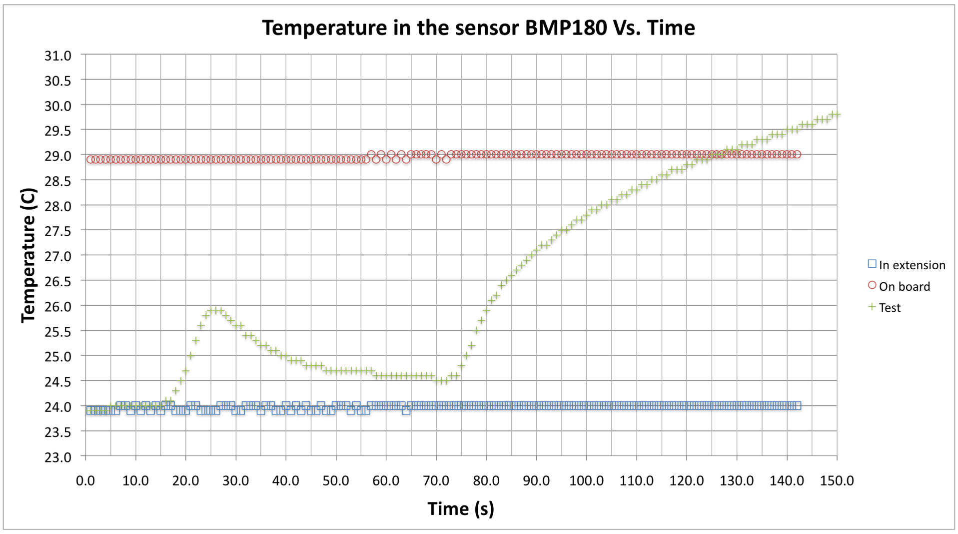


Figure 8: Temperature measurament using the BMP180 sensor. Attached directly in the TBS (red), with a cable extension (blue) and testing data.

Figure 9: Extension connections for the Test beam shield. For 6 and 5 pins.

and the extension is in the figure 9

5 Conclusions

I have tested the hardware and developed software for it. The major output is the understanding of the hardware which is documented [6] for the the Telescope Team.

6 Future Work

For future works it is suggested to develop the same task using the sensor **Bosch BME280**, which can measure the temperature, humidity and pressure at the same time, in order to avoid the need of use a class or maybe 2 different codes. Here the datasheet of this sensor.

Other recommendation is to add an monitoring system (**e.g** IR heat camera) of the complete bench of the test beam in order to have a other reference of this parameters. And finally but not less important is the made a benchmark between the new idea of *Slow Producer* and the Classical idea of the *Slow Control System* in order to integrate in EUDAQ properly.

Output: Temperature and Humidity values.

begin

-Call the class I^2C in order to emulate I^2C bus using GPIO.

//Define functions of the class.

function MEASURE(`self, dev, scl,sda`)
// **Self**: is the device. **dev**: is the bus. **scl**: is the clock. **sda**: is the data.
-Open device.
-Get temperature (from temperature function).
-Get humidity (from humidity function).
-Close device.
Return (temperature, humidity)
end function

function OPEN(`self, dev, scl, sda`)
-Open I^2C object.
-Write softreset.
-Sleep 50 ms.
end function

function TEMPERATURE(`self`)
-Writer trigger (NHM).
-Sleep (66 ms).
-Write data.
if *there is data from gpio pin 2* **then**
-Return data converted as a number
else
-Save 'None'.
end
end
end function

function HUMIDITY(`self`)
-Writer trigger (NHM).
-Sleep (25 ms).
-Read data from gpio pin 3..
if *there is data from gpio pin 2* **then**
-Return data converted as a number
else
-Return 'None'.
end
end
end function

function CLOSE(`self`)
-Close I^2C object.
end function
//Standar function.
-Use CHECKSUM function

end

Algorithm 1: *Pseudocode* of the SHT21 sensor. (Python)

References

[1] WiringPi

[2] Raspberry Pi Documentation

[3] Torsten's Documentation

[4] E. Corrin. "EUDAQ Software Manual". Edition (2010).

[5] H. Jansen, S. Spannagel, J. Behr, A. Bulgheroni, G. Claus, E. Corrin, D. G. Cussans, J. Dreyling-Eschweiler, D. Eckstein, T. Eichhorn, M. Goffe, I. M. Gregor, D. Haas, C. Muhl, H. Perrey, R. Peschke, P. Roloff, I. Rubinskiy, M. Winter. "Performance of the EUDET-type beam telescopes. ". arXiv:1603.09669v2.(2016)

[6] M. Morgado. "Documentation DESY Summer Project 2016. Slow Control with Raspberry PI for Test Beam Telescope.". (2016). `https://github.com/manuelmorgado/RPi_HTP.git`