Uppsala University

Proof of concept

- Using an Raspberry Pi in Earth Science studies

Jakob Papirov

2014-02-03

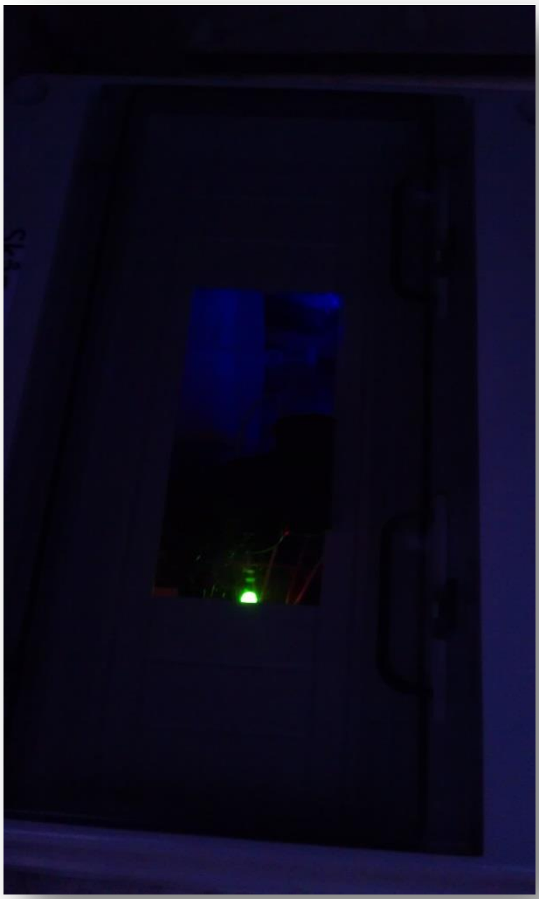


Table of Contents

Abstract

This project will show that it’s quite possible to successfully use a Raspberry Pi (RPI) in Earth science studies, more specifically in temperature measurements using only online resources and ordering the vital components online. To test whether a RPI can be used in this fashion temperature readings obtained from a temperature sensor (DS18B20) were compared with the department’s (Pt100) readings. The result will show that the DS18B20 is measuring 0.1 – 0.2 ^{o}C higher than the Pt100. A concern might be the low accuracy of the DS18B20 at ± 0.5 ^{o}C compared to the Pt100’s ± 0.1 ^{o}C, but the low of only 3 % of the RPI set-up compared to the department’s is something to think about.

1. Introduction

This section will describe the background to the project and the project goals.

1.1. Background

New technologies are continuously being developed, sometimes to suit a particular need, sometimes they spring out of other projects, but sometimes they are developed just because we, humans, can.

In the case of the Raspberry Pi (RPI), it's a mix of several things, on the one hand the founders behind the Raspberry Pi Foundation wanted to promote computer science in schools and on the other hand, the time was just right to make this a reality, because processors used in mobile phones had reached a level that could be used in a project such as this (Raspberry Pi Foundation, 2013). The first RRI was released February 29:th 2012 (Wikipedia, 2014) and within a year over one million units had been sold (Raspberry Pi Foundation, 2013).

I had heard of the RPI on several occasions and at one point I was curious if it had been used in any scientific work. I did a bit of searching online, but I didn't find any in Google nor in Google Scholar. In addition, since I am studying the hydrology/hydrogeology master programme I was interested if the RPI can be used in that field of study as I can see many advantages with the RPI.

1.2. Project goal

The goal of the project is to see if a Raspberry Pi can be successfully used in Earth science and particularly in hydrology (micrometeorology). The technology has been out for about 1.5 years (at the date of writing) but I haven't come across any mentionable usages of these devices in Earth science in general and hydrology in particular.

What I really wanted to study was if there was any application of the RPI in hydrology/hydrogeology? However since my general question was if the RPI could be used in Earth science at all, I decided to focus on something that I thought is easier to test. So I decided that measuring temperature would be the most suitable thing to do, as there should be plenty of information about that online.

In fact it was soon understood that there was quite a lot of different things that were being studied as the same time in my project.

The main idea was to see if this project could be done using only online resources as for documentation but also use some departmental hardware resources. Part of the project was to study how easy or difficult it would be to set up a temperature measuring system by ordering the necessary parts online and assembling them by following tutorial(s) and other information found online as well.

The second aspect of this project was to study how accurate the measurements from the RPI are compared to a scientific procedure used at Geocentrum.

Thirdly, was the RPI suitable as a logger and controller device, including programming and durability.

Lastly, I wanted to compare the price of my set-up and the one used by the department, because I find it relevant to take the cost into account when comparing the temperature results.

2. Method

This section describes in detail how the equipment is set up, basic information about the equipment and how the data was measured and then analysed for both the weather station and the RPI & temperature sensor. The idea is that my procedure should be reproducible, by following my main resource (Kirk, 2012), the bibliography and general procedure of finding relevant information online.

The main equipment used in this project can be seen in Table 1; Components were ordered online from Adafruit (Adafruit, 2013) and Sparkfun (Sparkfun, 2013).

Table 1. Table of the equipment that were bought for the project as well as the price.

Equipment nr	Item	Quantity	Price	Currency
1	Adafruit Assembled Pi Cobbler Breakout + Cable for Raspberry Pi (QTY: 1)	1	7.95	$
2	Temperature sensor (QTY: 1)	1	9.95	$
3	Wires (pack)	1	6.00	$
4	Button (QTY: 1)	1	0.35	$
5	Resistors (pack)	4	7.95	$
6	Breadboard (QTY: 1)	1	5.00	$
7	UBEC DC/DC Step-Down (Buck) Converter (QTY: 1)	1	9.95	$
8	Diffused Green 5mm LED (pack)	1	4.00	$
9	Diffused Red 5mm LED (pack)	1	4.00	$
11	USB-µUSB cable	1	70	SEK

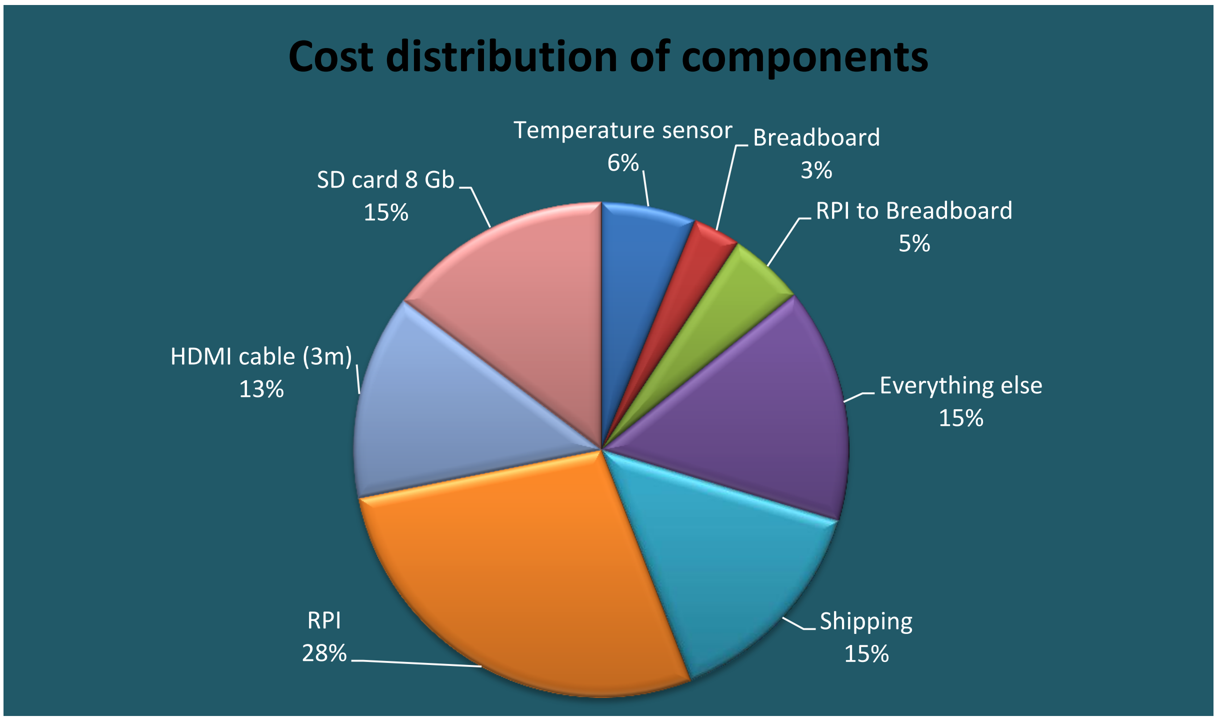


Figure 1. Pie-chart showing the price distribution of equipment from Table 1, but also including an RPI from Table 2 and SD card.

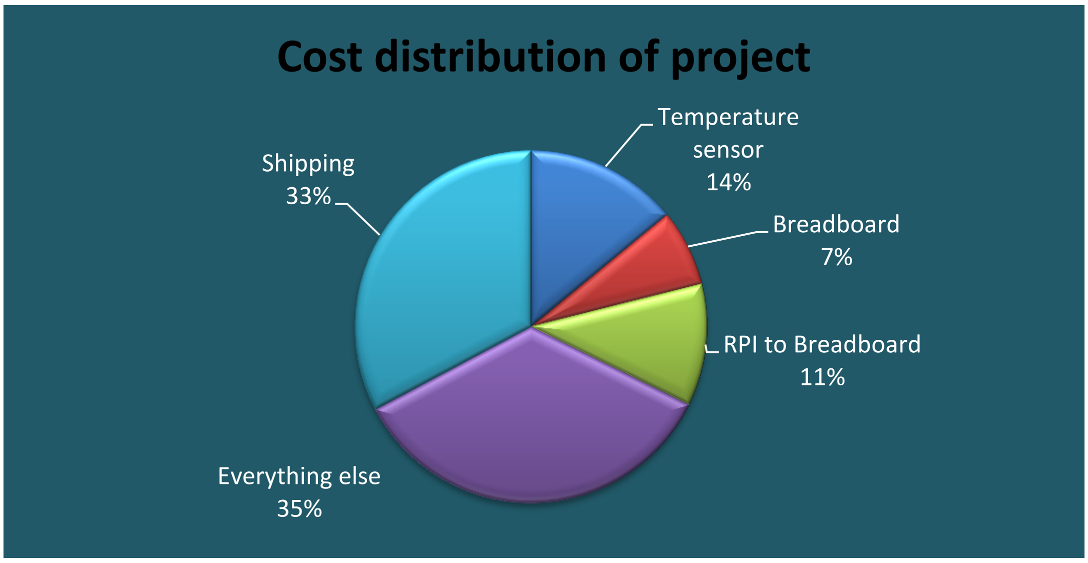


Figure 2. The actual cost of components bought, note that the RPI is not included.

However beside the main equipment there are a number of items that were essential in this project, seen in Table 2.

Table 2. Additional equipment essential for the project.

Equipment nr	Item name	QTY
12	Raspberry Pi Model B	1
13	Mobile Battery Pack	1
14	HDMI cable (2 m)	1
15	(DVI-HDMI adapter)	1
16	Monitor	1
17	Keyboard	1
18	Computer mouse	1
19	Enclosure (RPI)	1
20	Enclosure (entire set-up)	1
21	Desiccant	Varying amount (3-4)
22	Voltmeter	1
23	RPI charger	1
24	Adhesive tape	1 roll
25	SD card 16 Gb	1

2.1. The weather station

2.1.1. Temperature sensor

The temperature measurements carried out today at the department (Geocentrum) is part of the same time series as the one started by Anders Celsius' professor (Erik Burman) in 1722, in Uppsala (Bergström, 2013). The temperature sensor used today is a Pt100 (Bergström, 2013).

Pt100 stands for Platinum-100, which means that it is made of the element platinum with a resistance of 100 Ω (at 0 ^{o}C). This thermometer is of the type called resistance thermometer. A resistance thermometer uses a known correlation between the changes in resistance of material (platinum) with temperature (Wikipedia, 2014). The accuracy of the thermometer used by the department is 0.01 ^{o}C (Bergström, 2013).

2.1.2. Set-up

The Pt100 is connected to a CR1000 data logger from Campbell scientific (via a multiplexer). The temperature is measured every 10 seconds and an average over 10 minutes is then calculated by the logger program. The logger then sends the data via cable to a server at the department and new data is available for use every 24 hours (Bergström, 2013).

The main components required to set-up a system such as the one at the department are illustrated in Table 3 (Campbell Scientific, 2013), for price comparison I have added a Pt100 sensor from Omega (Omega, 2014).

Table 3. The minimum required components to set up a temperature measuring system. Note price difference on sensors between Omega and Campbell Scientific.

Equipment nr	Item name	Price	QTY	Purpose
1	CR-1000	£ 1071	1	Logger
2	Pt100	£ 109	1	Sensor (Campbell Scientific)
3	LoggerNet	£ 400	1	Software
4	Multiplexer	£ 399	1	Extension
5	Pt100	£ 48	1	Sensor (Omega)

2.2. Raspberry Pi + Temperature sensor (DS18B20)

2.2.1. Raspberry Pi

There are two models available, and since I have used model B, that is the one I am going to write about. The Raspberry Pi (RPI) has a SoC (System on a Chip) that includes an ARM processor of 700 MHz, a GPU (Graphics Processing Unit) and 512 Mb or RAM (Random Access Memory). It doesn't come with a data storage device but supports a SD card to store the OS (Operating System) and any files. The RPI also has an Ethernet controller and two USB 2.0 ports. According to the website RPI requires an input current of 5 V, 700 mA and 3.5 W (a standard smartphone charger should be enough). For video output the RPI has an HDMI connector. The RPI does not include a real time clock, which means that the time and date is only updated when an internet connection is present. In case it isn't, the operating system time will be continued from the date and time the last time it was on (Wikipedia, 2014).

Right after booting up the OS, it is necessary to check if there are any updates to the OS (software update) as well as downloading the necessary code that will allow the RPI to communicate with the DS18B20 (kernel update). In addition, it is a good idea to download the RPi.GPIO which is a library that will simplify programming of the GPIO pins in python (Kirk, 2012).

2.2.2. Temperature sensor

The temperature sensor used in this project was the DS18B20 from Dallas Semiconductor (owned today by Maxim Integrated). It is a digital thermometer requiring only one wire to send and receive data. In total it had three wires, with the other two being Ground and Power Supply Voltage (3.3 V) (Dallas Semiconductor, u.d.).

The measuring range is between -55 ^{o}C and +125 ^{o}C. The default accuracy without any calibration is $\pm$ 0.5 ^{o}C and the resolution can be programmed between 9 to 12 bits, meaning a resolution of 0.5 ^{o}C, 0.25 ^{o}C, 0.125 ^{o}C or 0.0625 ^{o}C, where the default is 0.0625 ^{o}C (Dallas Semiconductor, u.d.).

As the DS18B20 is a digital thermometer its output is in binary form, instead of voltage, as does an analogue thermometer. The conversion from binary output to temperature is done by the RPI. There is already written code that will allow the RPI to communicate with the DS18B20 and convert the binary data input to decimal temperature output (in Celsius) (Dallas Semiconductor, u.d.).

2.2.3. Set-up

Figure 4 and Figure 5 is intended to illustrate how my experiment was configured and how it looked like in the field. A closer inspection of the actual wiring can be seen as a simplified diagram in Figure 3. The figure is supposed to give a sense of how the temperature sensor, the button and the two LEDs were connected to the RPI via the Pi Cobbler and how the resistors were used (see Figure 10 and Table 6 in the Appendix on p. 22, for more detail).

Initially my plan was to use a 12 V car-like battery to power the RPI, but I was not able to connect the power source to the UBEC and to the RPI, so instead I used my own battery, which is a battery pack for smartphones and has a µUSB connector that the RPI uses.

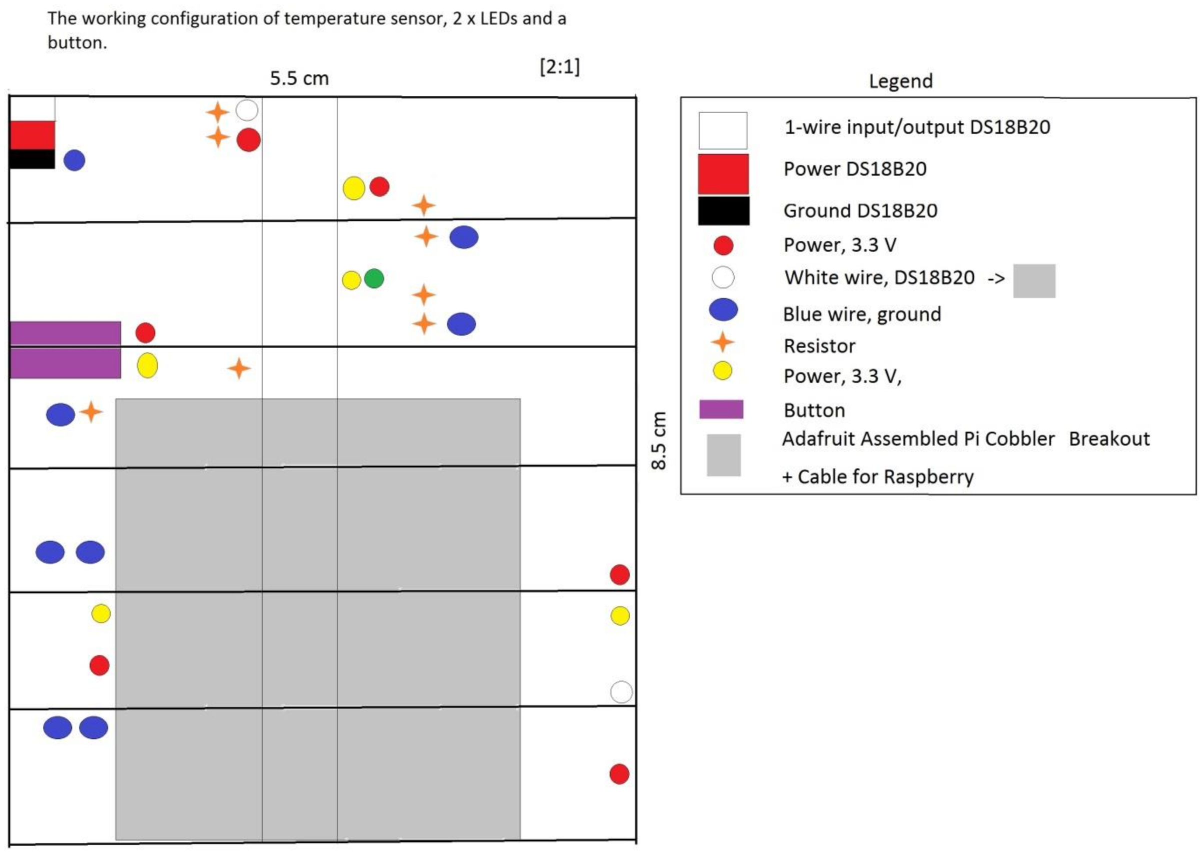

The working configuration of temperature sensor, 2 x LEDs and a button.
5.5 cm
[2:1]
8.5 cm
Legend
1-wire input/output DS18B20
Power DS18B20
Ground DS18B20
Power, 3.3 V
White wire, DS18B20 ->
Blue wire, ground
Resistor
Power, 3.3 V,
Button
Adafruit Assembled Pi Cobbler Breakout + Cable for Raspberry

Figure 3. A simplified diagram, showing the how the necessary wires and components were configured on the breadboard. For a more detailed view please see Figure 10 and Table 6 in the Appendix on p. 22.

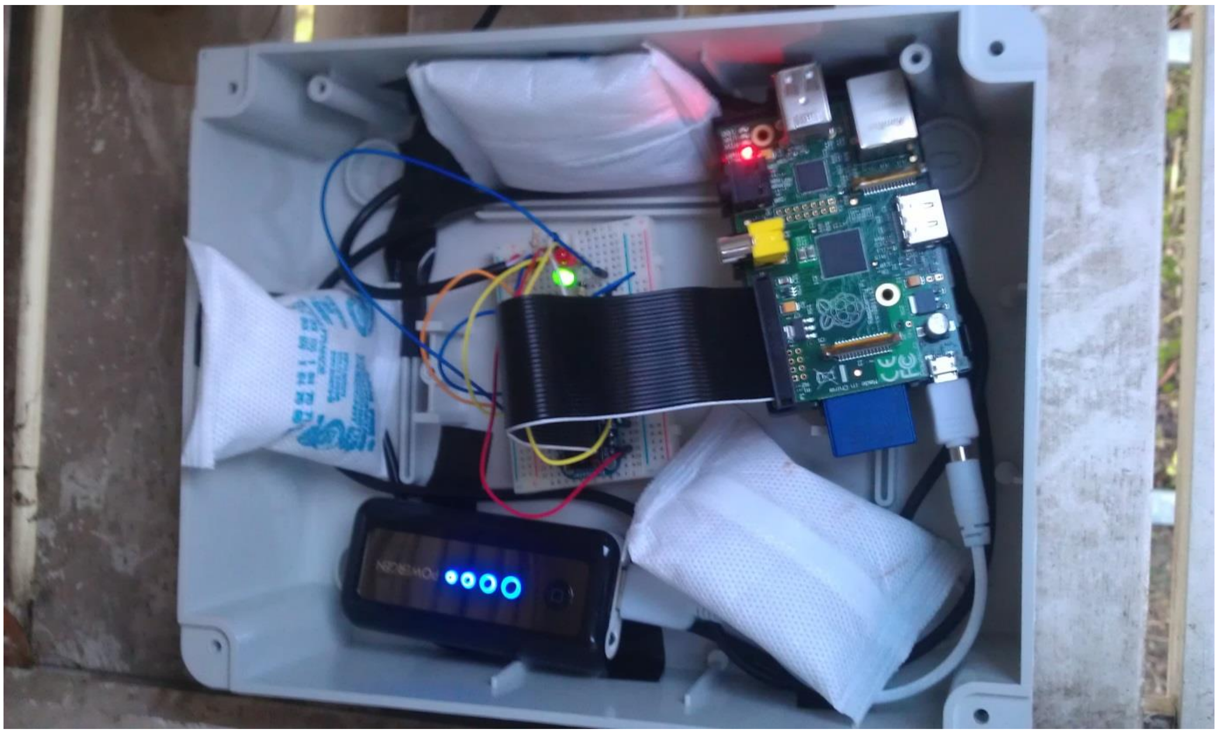


Figure 4. Here we see how everything looks like during a "run". The "Mobile Power Pack" is fully charged and on (blue lights, RPI is on (red light) and the program is taking readings (green light).

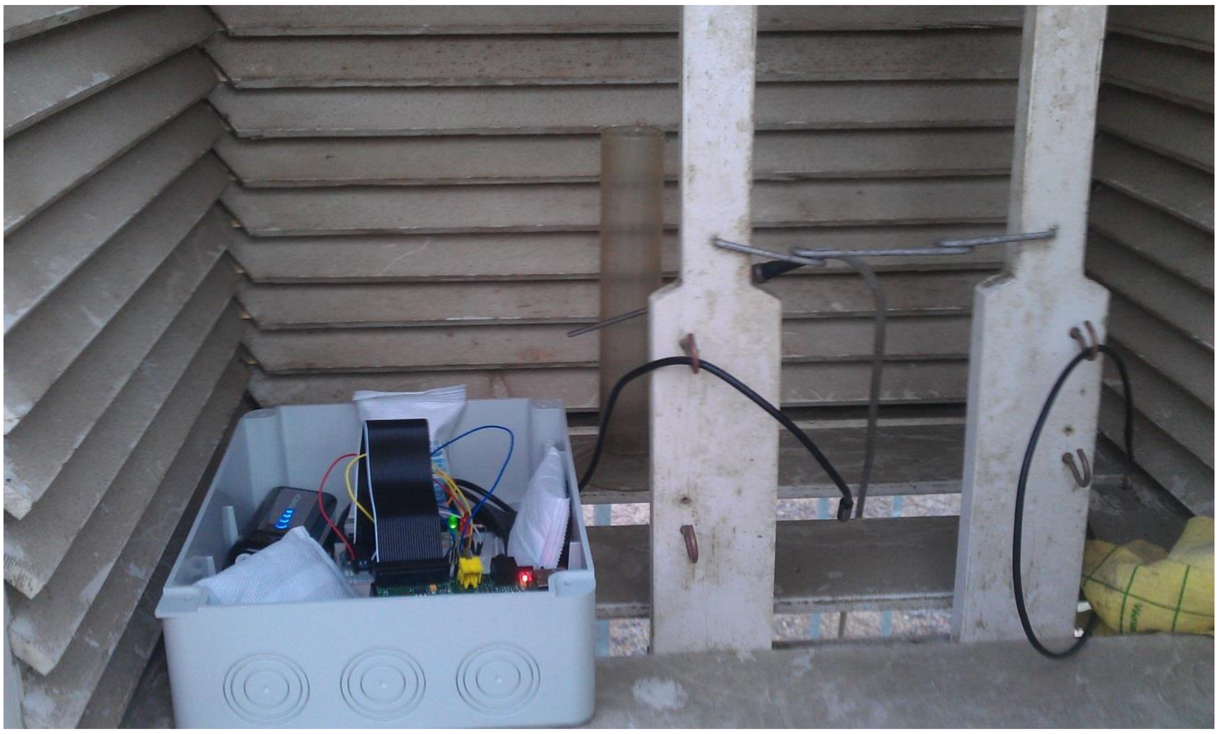


Figure 5. In this view, the temperature sensor (DS18B20) can be seen to the left (centre in figure) of the Pt100 (to the right in the figure), and is a side view of Figure 4.

2.3. Obtaining the data

2.3.1. Programming the RPI

A program was necessary in order to make the DS18B20 to take a temperature reading and then send it to the RPI which will convert the binary data to actual temperature data and then store it on the SD card. The program was written in python following a tutorial, but some modification was done to suit my needs. The program had to be started from the command line (Kirk, 2012).

The basic procedure of the python program was to light up an LED (Red), wait for a physical button press, then turn off the LED (Red) and turn on LED (Green) and start collecting temperature data once every ten seconds and write it to a file (folder name is a timestamp). Upon another button press the Green LED would turn off and the program would close the file and not take any more measurements (Kirk, 2012).

There are a few different operating systems (OS) available for the RPI; all of them are based on Linux to some extent. I have used Raspbian which is an adaptation from debian to suit the RPI. Python is officially the programming language to use on the RPI and comes pre-installed on Raspbian. The installation of the OS is done manually by writing an image to the SD card (Raspberry Pi Foundation, 2014). It is easiest done on another computer (I did it on a netbook running windows 7).

2.3.2. Collecting the data

The temperature measured was saved as a temperature.log-file on the RPIs SD card. The measurements was taken and stored on the log file once every 10 seconds, totalling 4801 data points and c:a 30 kb. The easiest way to collect the data from the RPI was to connect the RPI to the home network and then logging onto it via WinSCP (windows) and retrieving the data files.

2.4. Data analysis

The data was analysed with Mathworks MATLAB R2012a. The data was averaged over 10 minutes to provide a good resolution, reduce noise and to lower the data size and speed up the processing. Collecting rate of 1 measurement per 10 seconds and then averaging the data over 10 minutes is the same procedure as is being done at the department. No timestamp was included in the .log file because it wouldn't be correct, would increase the file size and since the time interval was known and the starting time was known since it is stored as folder name; the time can easily be re-constructed as a vector in MATLAB.

There was however some extra work that had to be done. My initial plan was to measure the temperature in one go for three days (72 hours) but since I wasn't able to use the 12 V power source and I used my own battery I couldn't measure the temperature for that long because of the limited power of the battery. In the beginning of the project I had attempted to calculate an estimate of how long my battery would last and I got a figure of about 10 hours (which is why I decided to use the car battery). In my first run I decided to leave my set-up overnight and see how long I was able to measure temperature, unfortunately I got bad data from that run (which I tracked down to be loose sensor wiring), so I then decided to do three measurements over the days and measure over different time periods during each day.

Since I now had three datasets I had to re-write the MATLAB code to accommodate for that and in addition I realized (upon a first quick plot) that there was some time-constant effect going on and so I had to remove data points until the sensor had settled down. That is however not part of this project so I will not include anything about that.

The solution I came up with to obtain a 10-minute average of the dataset was to reshape the dataset-vector with the number of data points in each column being the sought after average, which in my case is 60, but the column number is unknown. That led to a situation where I had to sacrifice data points in order for my solution to work. I feel that my solution was elegant, efficient and in total I had to remove c:a 100 ± 20 data points (2 ± 0.4 %). That made constructing my time-vectors a bit complicated which unfortunately forced me to manually pick out the rows from each dataset.

Because I had three datasets I had to extract the corresponding temperature subsets from the weather station (Pt100) dataset, and again I ran into the problem with not knowing the time.

3. Results

In this section the results of my temperature measurements will be presented as well as my data analysis results.

3.1. Results from the weather station

Temperature is graphed between 2013-12-18 and 2013-12-20. The graph is continuous (time and temperature).

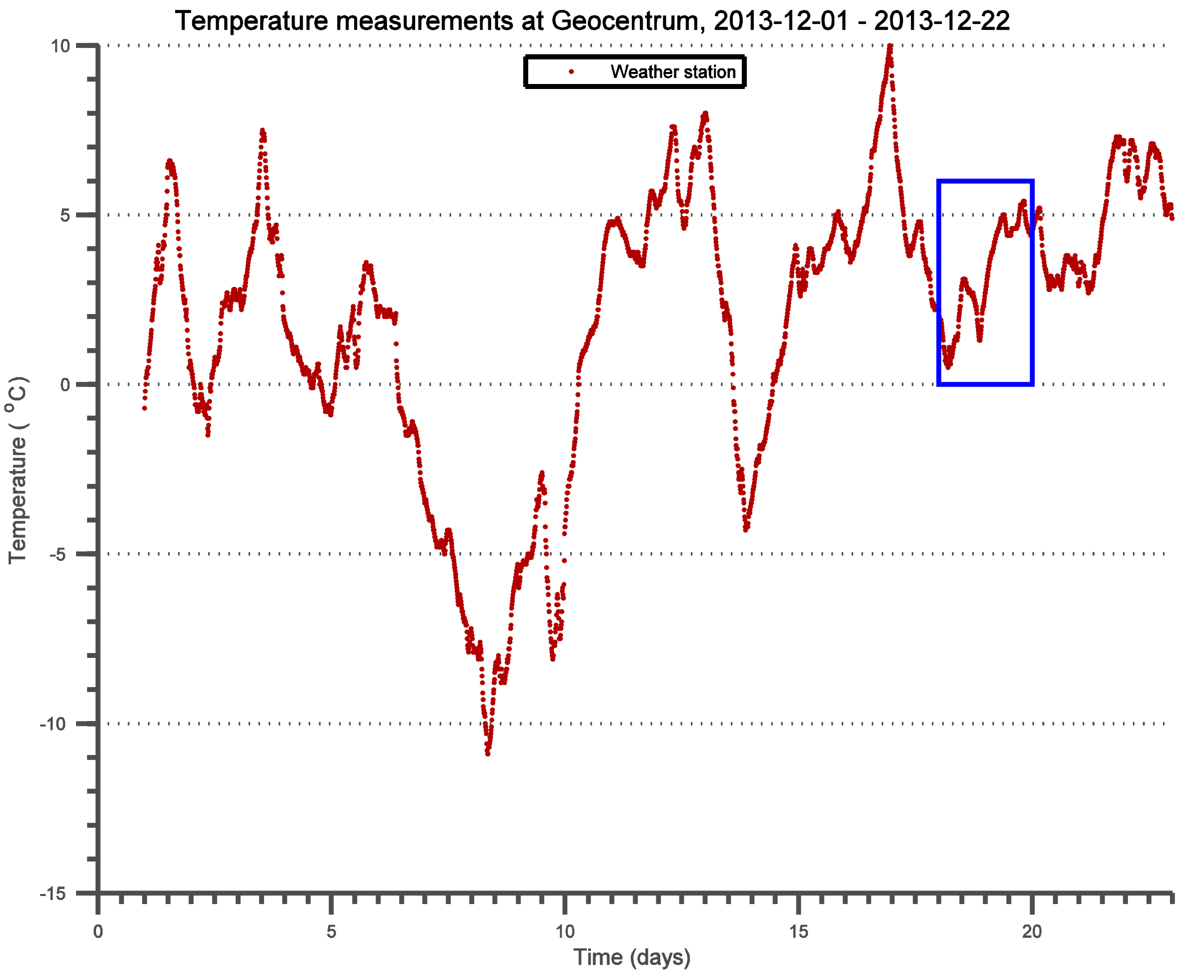


Figure 6. Time-series of temperature, measured by the Pt100 sensor at Geocentrum 2013-12-01 to 2013-12-22.

Even though I don't attempt to draw any conclusions from my project on general temperature trends it is good to have a background to see where my measurements fit in. This can be seen in Figure 6, which shows the temperature changes over time during most part of December. The maximum is around 10 °C and the minimum is around -11 °C. Right before my three-day measuring period (18:th – 20:th) the temperature was at the highest and was going down and within my measuring period the temperature stayed around 1 and around 5 °C.

3.2. Results using the Raspberry Pi & DS18B20

Temperature is graphed between 2013-12-18 and 2013-12-20 and will show that the DS18B20 is constantly measuring a higher temperature than the Pt100.

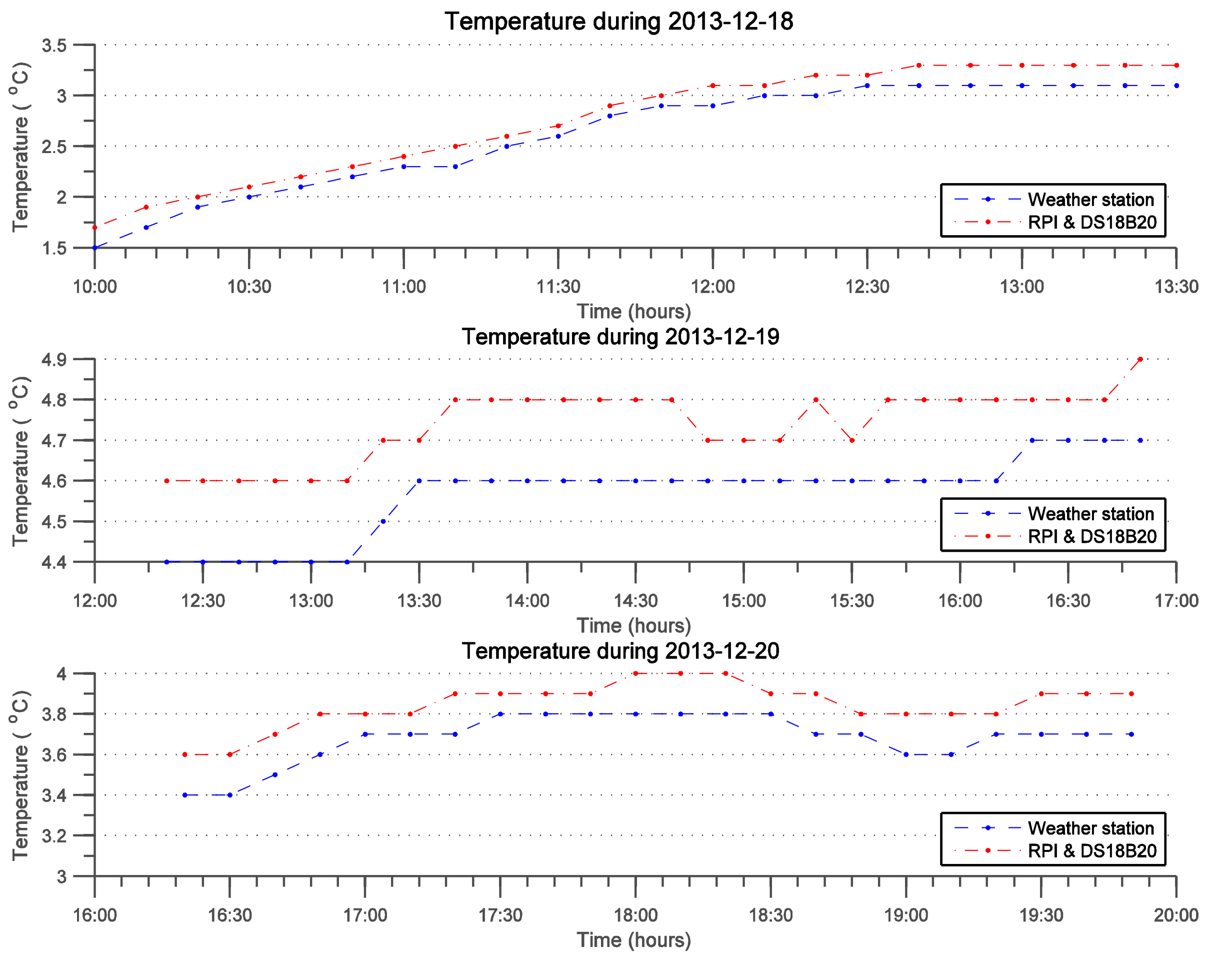
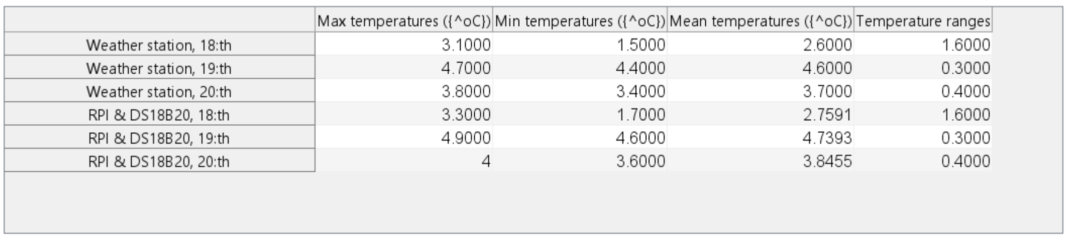


Figure 7. Three subplots where each plot corresponds to one measurement occasion but during different time-periods in a day.

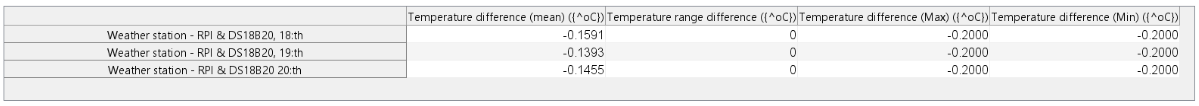
Zooming in and taking a look at my measurements, seen in Figure 7, it is clear that the red line shows consequently higher values than the blue one. It is also clear that the temperature varies during the course of the day and is highest during the second measurement occasion (19:th).

Table 4. A table with simple descriptive statistics of each dataset.



	Max temperatures ({^oC})	Min temperatures ({^oC})	Mean temperatures ({^oC})	Temperature ranges
Weather station, 18:th	3.1000	1.5000	2.6000	1.6000
Weather station, 19:th	4.7000	4.4000	4.6000	0.3000
Weather station, 20:th	3.8000	3.4000	3.7000	0.4000
RPI & DS18B20, 18:th	3.3000	1.7000	2.7591	1.6000
RPI & DS18B20, 19:th	4.9000	4.6000	4.7393	0.3000
RPI & DS18B20, 20:th	4	3.6000	3.8455	0.4000

Table 5. A follow-up table from Table 4 which compares its statistics.



	Temperature difference (mean) ({^oC})	Temperature range difference ({^oC})	Temperature difference (Max) ({^oC})	Temperature difference (Min) ({^oC})
Weather station - RPI & DS18B20, 18:th	-0.1591	0	-0.2000	-0.2000
Weather station - RPI & DS18B20, 19:th	-0.1393	0	-0.2000	-0.2000
Weather station - RPI & DS18B20 20:th	-0.1455	0	-0.2000	-0.2000

From Table 4 and Table 5 it is interesting to note that both the minimum and the maximum temperature measured by the DS18B20 is 0.2 ^{o}C higher than the Pt100. In addition, the temperature ranges are identical, but there is a very slight difference in the mean values calculated, but again the DS18B20 is showing the higher values.

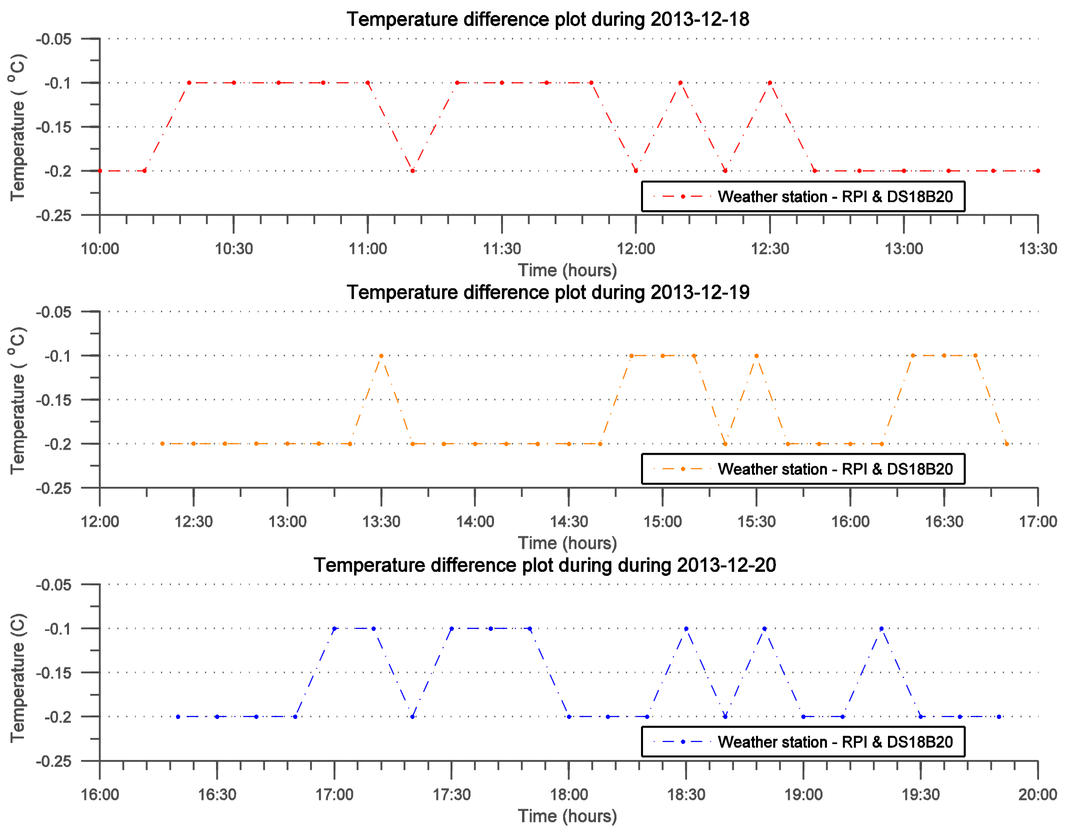
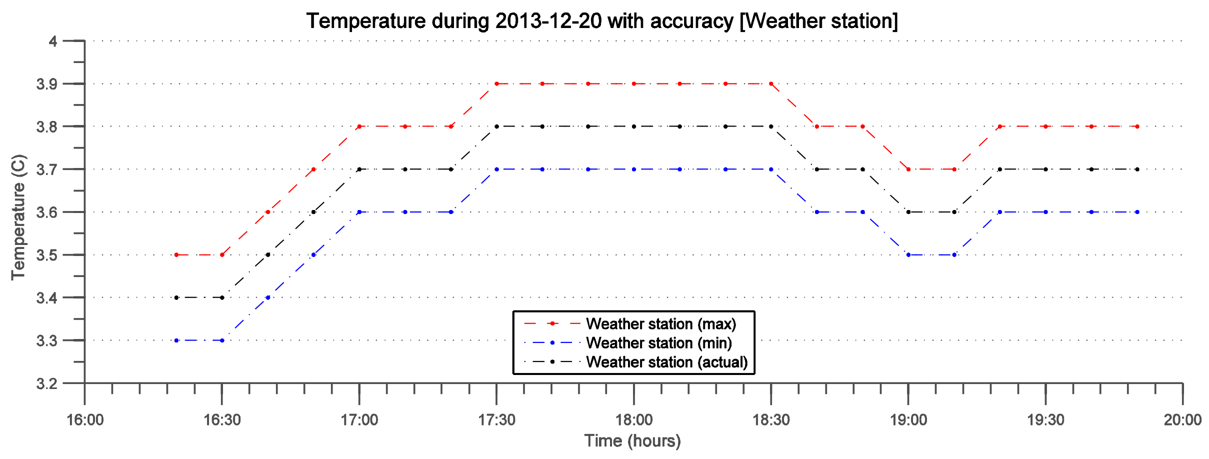


Figure 8. Subplots with the weather station temperature subsets subtracted by the DS18B20 measurements.

I wanted to take a closer look what the possible difference might be between my sensor readings the Pt100, so I made the difference plot shown in Figure 8. It simply shows how much the temperature values obtained from the Pt100 differs from the DS18B20 by taking each Pt100 value and subtracting it with a DS18B20 value.

What is visible is that the DS18B20 always records a higher value than the Pt100, and this difference is discrete in the way that it's either 0.1 ^{o}C or 0.2 ^{o}C higher. It is difficult to see any real pattern between each data subset, but perhaps if one looks at the peaks and counts the number of -0.1 ^{o}C peaks and divides by the total number of points, then the top subplot might be considered to show best agreement with the Pt100 data as the ratio is higher than the other two (the bottom subplot is only slightly better than the middle one).



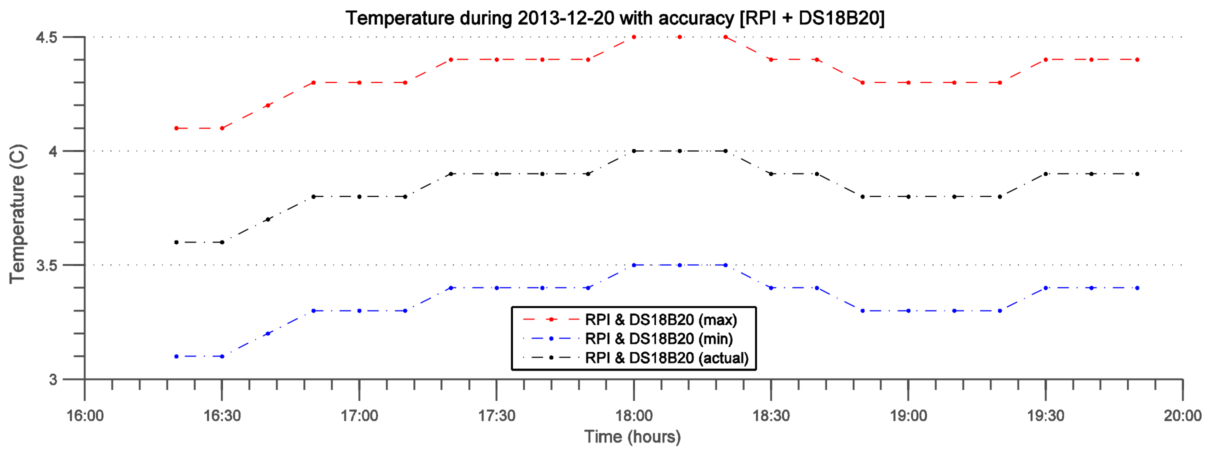


Figure 9. Temperature measurements by the weather station (PT100) and the DS18B20 but with their respective accuracies included.

The biggest difference between the DS18B20 and the Pt100 is clearly seen in Figure 9, due to the 5 times better accuracy of the Pt100 sensor compared to the DS18B20. The Pt100 sensor has an accuracy of ± 0.1 ^{o}C and the DS18B20 has an accuracy of ± 0.5 ^{o}C (within the range -10 ^{o}C to +85 ^{o}C).

4. Discussion

In this section the different parts presented in this project will be compared and discussed.

4.1. Programming

The CR1000 is programmed using CRBasic, which is a programming language developed specifically for the purpose of logging data.

Since the python language is the official programming language for the RPI, it comes pre-installed in Raspbian (the OS); the main reason is probably that it's open source, meaning that it is free. It is a modern high-level, general-purpose programming language that is also cross-platform, giving it a great deal of flexibility. The main advantage with python are the many libraries that are available with code written by the community and for the community, as with RPi.GPIO and the kernel-code to the RPI, simplifying writing code that uses the GPIO and allowing communication with many hardware devices connected to the RPI through the GPIO. Community programming does have its down-sides compared to commercial software. In my project I came in contact with that aspect when I was trying to connect the DS18B20 to the breadboard and take a temperature reading from the RPI. I was sure that my wiring was correct, even though I had not followed the instruction to the letter, because for instance it does not matter which ground pin (on item 1, Table 1) I use or which power pin I use, and also because there was nothing mentioned in the instruction that I had to use a specific pin. However I discovered online on the RPI forum that I had to connect the data wire (or input/output wire) to pin #4, as it had be hard-coded in the RPi.GPIO code, and when I made the changes, I was able to take a temperature measurement.

The up-side using CRBasic is that since its proprietary of Campbell Scientific, you can expect a certain level of stability, functionality and you have the comfort knowing that there is a support team to help you. You can also take courses where you learn how to use the software and program specific sensors.

An advantage with using python is that you have a higher degree of flexibility, for instance, one can code so that a new file is generated with name being equal to the time it was first created (or in my case create a folder with the name of the timestamp) for each run.

4.2. Comparing the results

There are many different ways to compare data. For this project, I chose to do a simple comparison between the two datasets, illustrated graphically, a table with some basic statistics, a difference plot and a plot with accuracies.

From my results a few initial question arises, why does the DS18B20 sensor constantly record a higher temperature, specifically + 0.1 or 0.2 ^{o}C, than the Pt100? How come it's constant? Why does it never show a negative value? Unfortunately, I have no explanations to the questions and in the DS18B20 datasheet/manual I haven't found any clues either. A possibility might be that the temperature differences are not that discrete at all, but that instead rounding of values is the cause, however an inspection of the code doesn't suggest it, but it might still be the case.

Regarding the accuracy, since that's an intrinsic property of the sensor it can't be changed, however if the sensor would be calibrated using the information my study has shown, and then it would show almost a perfect fit with the Pt100 sensor.

4.2.1. Discussion of errors

As my project was to compare the temperature readings between my set-up and the one by the department, the most accurate way was simply to put my device in the same box as the Pt100 was in. That way no external error sources were possible from the beginning. However, since I had to walk from Geocentrum to the weather station with my enclosure containing my experiment there was a question of when the experiment was actually started. To solve this I wrote down the time and date when I pressed the button on the PRI to start measuring temperature and the date and time when after I had placed my enclosure inside the wooden box.

Something I noticed right away during my initial data analysis was that the DS18B20 data has showed much higher temperatures than the Pt100 even after I had removed the values corresponding to me walking from Geocentrum to the weather station (8 – 10 minutes). I could probably have written code in MATLAB that would remove the additional data points automatically, or removing it manually in the pre-analysis code, but instead I decided not to remove it because the number of data points was slightly higher than 60, meaning one 10-minute data point and the rest would average out in the next 10-minute data point. In addition, since I had chosen to use the *reshape* command in MATLAB, I was forced to remove data points anyway, and I didn't want to remove more data points than I had to.

4.3. CR-1000 vs. the RPI

As with the CRBasic programming language, the CR-1000 is built with the research market in mind and thus once again, one should expect a certain level of build-quality. This includes amongst other things lightning strike tolerance. In addition, the CR-1000 can connect up to 16 Single Ended (SE, analogue) sensors or 16 Differential (Diff, analogue) sensors. In addition several digital ones can be connected as well and using a multiplexer (like an extension cord, but for loggers), even more sensors can be connected at the same time (Campbell Scientific, 2012).

The RPI on the other hand is very small and light but given its specifications it will not stand a lightning strike, but the community is such that someone might build one. Even though more and/or bigger breadboards can be connected to each other in series and thus allowing for multiple sensors being connected, the entire set-up will be quite clunky and I am sure a wire will get loose when working with so many of them.

Perhaps the biggest advantage of the RPI is its cheap and abundant storage. Before I began my measurements I wanted to get a feeling of how much storage my data would require, and from my calculations I ended up with 101 kb (8 byte á character), but in reality it was only 30 kb. So that might not be a problem in this case, but I can see a situation where the 4 Mb of available memory on the CR-1000 might require a data collection trip/visit which would not be necessary on the RPI. In addition, the RPI should allow an installation of a database system such as MySQL which could store data from multiple measurement occasions.

The CR-1000, however has the advantage of having a Real Time Clock (RTC) (Campbell Scientific, 2012), which as mentioned before, the RPI lacks due to price-competitiveness, that means that the CR-1000 will automatically log date and time, something that would have simplified things for me if the RPI had.

4.4. Cost vs. accuracy

From Figure 1, the main thing to note is that the cost of the RPI is only about 30 % of the total theoretical cost of the components needed for the project. However in both Figure 1 and Figure 2, equipment such as a monitor and keyboard and mouse were not included. From Figure 2 the importance of shipping cost emerges quite clearly (≈35 %) as well as how small part the cost for sensor actually is (≈15 %).

Given that the total cost of my project was ≈ 600 SEK (of purchased equipment) and the total cost of items 1 – 4 from Table 3 is ≈ 21 000 SEK, my project makes up ≈ 3 % of the departments cost.

4.5. Possible expansion of project

There are a number of things that could have been done in order to provide greater accuracy of measurements, simplifying the experiment and expanding the scope of the project.

If for instance the temperature sensor would have been calibrated, temperature reading taken at known temperature (0 ^{o}C or 100 ^{o}C), and/or another temperature sensor would have been used, then perhaps the accuracy would have been improved.

To simplify the experiment, changes to the boot-up routine could have been done in order to be able to start the python program during boot-up and so no carrying of the device would be necessary. That would eliminate something going wrong on the way to the experiment site. In addition, a GPRS transmitter & receiver would have been handy, because then the program could be started over the internet, quite possibly from a smartphone. Furthermore, the log-file could be fetched as well, and updated code (python) could be sent to the RPI and started remotely (although the point of this experiment was to use a push-button, so physical presence would still be required).

Given more time, the problem with interfacing the 12 V battery with the RPI might have been solved or not used at all, and instead focus had turned to try out a more powerful "mobile power pack".

To expand the project one could have in addition measured the power usage, using a voltmeter, to study how much power a project such as this would actually require in field conditions. It is worth nothing here that using the Ethernet and/or the USB ports is the biggest power-drain.

4.6. Possible future applications

The things mentioned in 4.5 (p. 17) can be expanded upon and used generally in other projects involving the RPI and sensors. Also, if possible instead of a GPRS transmitter & receiver an usb-wifi adapter can be used or an Ethernet connection. Regarding the power supply, if possible solar cells would be a very good alternative and/or using car batteries in series, where one used-up battery can be switched with a fresh one without having to restart the measurement readings.

5. Conclusion

It is easy to say that the Pt100 is a better set-up than the RPI & DS18B20 because the latter's results are not as good as the former, with main emphasis on the accuracy. However, the question one should ask is, "in what setting is it better?" The obvious answer to that is that since the Pt100 is part of a scientific research project accuracy is the most important factor. But even there it can vary; if you want to measure climate change then an accuracy of ± 0.5 ^{o}C can't be acceptable, but in other situations, it might be enough, such as measuring temperature in the work space. In addition, if the DS18B20 would be calibrated using this study as a basis then it would improve the reliability of the data greatly.

From another aspect; if the goal is to get a general picture of temperature then the DS18B20 is better because it's cheaper and probably simpler to set-up if one is unfamiliar with CRBasic and the CR-1000. As has been mentioned before, the accuracy is the main problem, if that could be solved than the possibilities increase. I think for home use the DS18B20 is very suited. Another aspect is that since the RPI & DS18B20 set-up is so cheap and can be bought in bulk many experiments can be made covering a larger area than what might possible with only one or two set-ups with CR-1000 and Pt100. Lastly, I believe that in remote locations and/or regions where funds are difficult to find the RPI & DS18B20 is a very suitable set-up.

I consider my project successful in that I have demonstrated that a temperature measuring system consisting of a Raspberry Pi, a temperature sensor (the DS18B20) and smartphone battery pack can be used to accurately measure temperature on pair with current methods.

6. Bibliography

Adafruit, 2013. *Adafruit.* [Online]
Available at: www.adafruit.com
[Accessed November 2013].

Adafruit, 2013. *DS18B20 Temperature Sensing.* [Online]
Available at: http://learn.adafruit.com/adafruits-raspberry-pi-lesson-11-ds18b20-temperature-sensing/hardware
[Accessed December 2013].

Adafruit, n.d. *Raspberry Pi lessons: Network setup.* [Online]
Available at: http://learn.adafruit.com/adafruits-raspberry-pi-lesson-3-network-setup/overview
[Accessed December 2013].

Adafruit, n.d. *UBEC.* [Online]
Available at: http://www.adafruit.com/products/1385#Description
[Accessed December 2013].

Anon., n.d. *Electrical Formulas.* [Online]
Available at: http://www.engineeringtoolbox.com/electrical-formulas-d_455.html
[Accessed November 2013].

Bergström, H., 2013. *Background to the weather station* [Interview] (December 2013).

Bergström, H., 2013. *Climate and climate variations (2).* Uppsala: s.n.

calebzulawski, 2012. *How to portably power your raspberry pi with a battery.* [Online]
Available at: http://www.thefruitycomputer.com/forums/tutorials/article/17-how-to-portably-power-your-raspberry-pi-with-a-battery/
[Accessed December 2013].

Calebzuwaski, 2013. *How to portably power your Raspberry Pi with a battery.* [Online]
Available at: http://www.raspberrypiforums.com/forums/tutorials/article/17-how-to-portably-power-your-raspberry-pi-with-a-battery/
[Accessed 1 December 2013].

Campbell Scientific, 2012. *CR 1000 measurement & control datalogger,* s.l.: s.n.

Campbell Scientific, 2013. *European Price List, Data acquisition for science, industry and research,* s.l.: s.n.

Dallas Semiconductor, n.d. *Preliminary DS18B20 Programmable 1-Wire (R) Digital Thermometer,* s.l.: s.n.

Kirk, M., 2012. *Raspberry Pi & Temperature Sensor.* [Online]
Available at: http://www.cl.cam.ac.uk/projects/raspberrypi/tutorials/temperature/
[Accessed 1 December 2013].

Ladyada, n.d. *solder-it.* [Online]
Available at: http://learn.adafruit.com/adafruit-pi-cobbler-kit/solder-it
[Accessed 1 December 2013].

Maxim Integrated, n.d. *DS18B20 Datasheet.* [Online]
Available at: http://www.maximintegrated.com/datasheet/index.mvp/id/2812
[Accessed December 2013].

Mullins, R., 2012. *Operating a Simple Switch and LED on the Raspberry Pi.* [Online]
Available at: http://www.cl.cam.ac.uk/projects/raspberrypi/tutorials/turing-machine/two.html
[Accessed December 2013].

Olly Andrade, E. J. A. L. D. B., n.d. *Buttons and Switches.* [Online]
Available at: http://www.cl.cam.ac.uk/projects/raspberrypi/tutorials/robot/buttons_and_switches/
[Accessed December 2013].

Omega, 2014. *Short RTD Probe.* [Online]
Available at: http://www.omega.co.uk/pptst/PR-20.html
[Accessed 23 January 2014].

Raspberry Pi forum, 2013. *Enclosure for sub-zero temperature.* [Online]
Available at: http://www.raspberrypi.org/phpBB3/viewtopic.php?f=40&t=57428
[Accessed December 2013].

Raspberry Pi Foundation, 2013. *About us.* [Online]
Available at: http://www.raspberrypi.org/about
[Accessed 23 January 2014].

Raspberry Pi Foundation, 2014. *Downloads.* [Online]
Available at: http://www.raspberrypi.org/downloads
[Accessed November 2013].

Raspbery Pi Foundation forums, 2013. *Source code to w1-therm?.* [Online]
Available at: http://www.raspberrypi.org/phpBB3/viewtopic.php?f=37&t=40044&start=25
[Accessed 10 December 2013].

Sparkfun, 2013. *Sparkfun.* [Online]
Available at: www.sparksfun.com
[Accessed November 2013].

Wikipedia, 2013. *General Purpose Input/Output.* [Online]
Available at: General_Purpose_Input/Output
[Accessed 23 January 2014].

Wikipedia, 2014. *Raspberry Pi.* [Online]
Available at: http://en.wikipedia.org/wiki/Raspberry_Pi
[Accessed 23 January 2014].

Wikipedia, 2014. *Resistance thermometer.* [Online]
Available at: http://en.wikipedia.org/wiki/Resistance_thermometer
[Accessed 23 January 2014].

Appendix

The working configuration of temperature sensor, 2 x LEDs and a button.

[2:1]

5.5 cm

8.5 cm

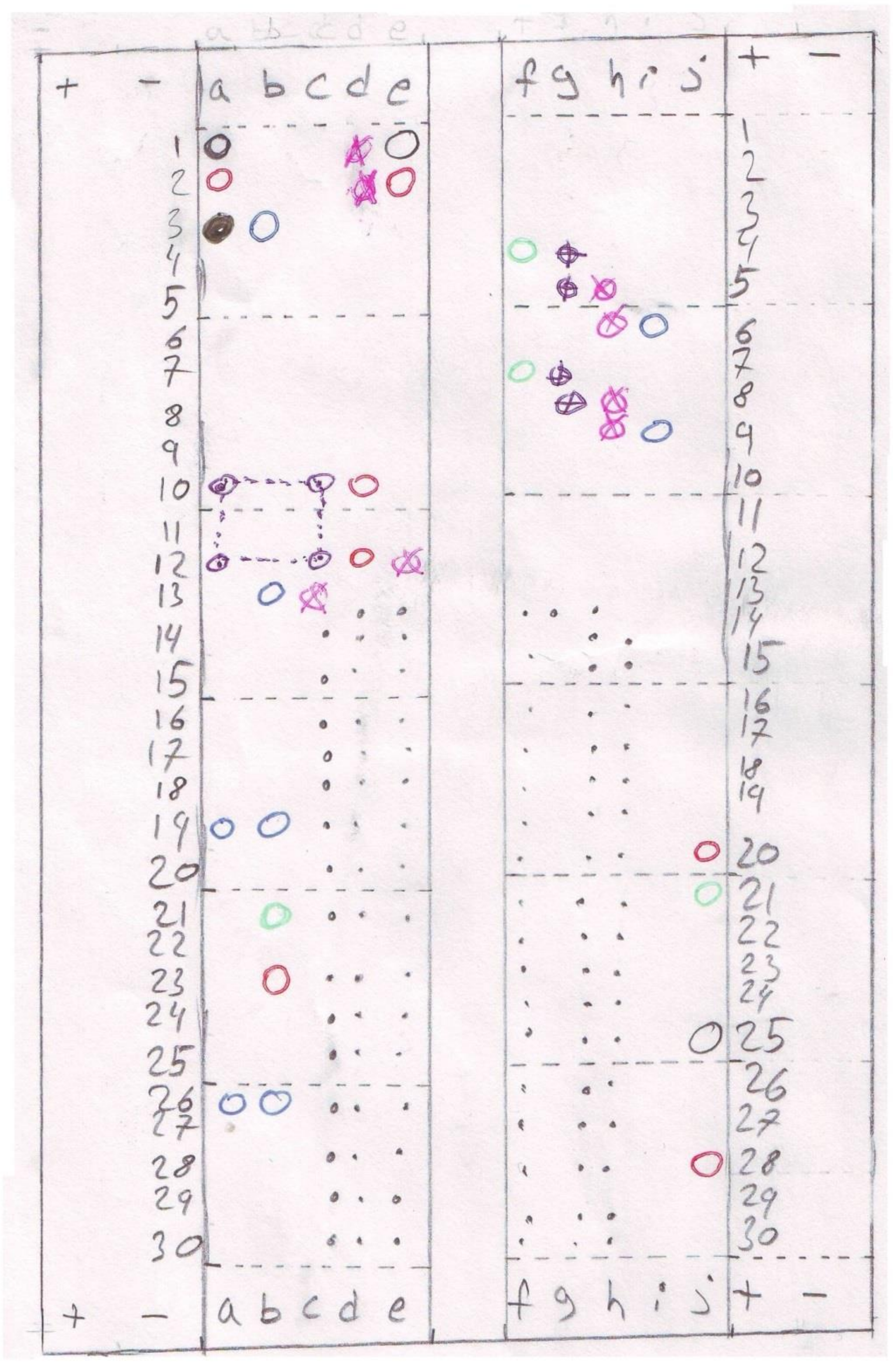


Figure 10. A hand-drawn schematic of how the different components are connected on the breadboard.

Table 6. A table showing how the different components are connected on the breadboard.

Purpose	Input	Output	Purpose	Input	Output
1-wire (white)	1a	(1e)	Output (yellow)	4f	21j
3.3 V power (red)	2a	(2e)	LED (red)	4g	5g
Ground (black)	3a	(3b)	220 Ω resistor	5h	6h
470 Ω resistor	1d	2d	Ground (blue)	6i	26a
Input (white)	1e	25j	Output (yellow)	7f	21b
Power (red)	2e	28j	LED (green)	7g	8g
Ground (blue)	3b	26b	220 Ω resistor	8h	9h
Button	(10a, 10d)	(12a, 12d)	Ground (blue)	9i	19b
Power (red)	10d	20j			
Input (orange)	12d	23b			
Ground (blue)	13b	19a			