Internship report

Calibration of the PowerIt

Jakob Sawatzki

June 2021

Abstract

The PowerIt board is responsible for controlling the parts which supply power via enable pins and measuring the current and voltage in the wires. An improved method for the calibration is presented along with some other important findings. For the calibration the already existing software had to be improved, software had to be written to determine the calibration parameters and also a way to write those parameters on to the PowerIt.

Contents

1 Goal

The PowerIt can read out the voltage and the current in its wires but those values are raw data and need to be calibrated. When calibrated one can read out that data easily and monitor the PowerIt properly.
The goal of this internship was to find a way to take raw data from the PowerIt and use this data to calibrate it. This included finding a proper fit function for the data, and figuring out how to write the calibration parameters back onto the PowerIt as well as finding a way to justify the results. In addition the calibration process must be as short as possible but also reliable.

2 Introduction

The research of the group Electronic Vision(s) involves all hardware components as well as software solutions around the Wafer-system BrainScaleS.
The different parts are provided with 1.8V (HICANNs)[1] and 9.6V (FPGAs)[2]. The power

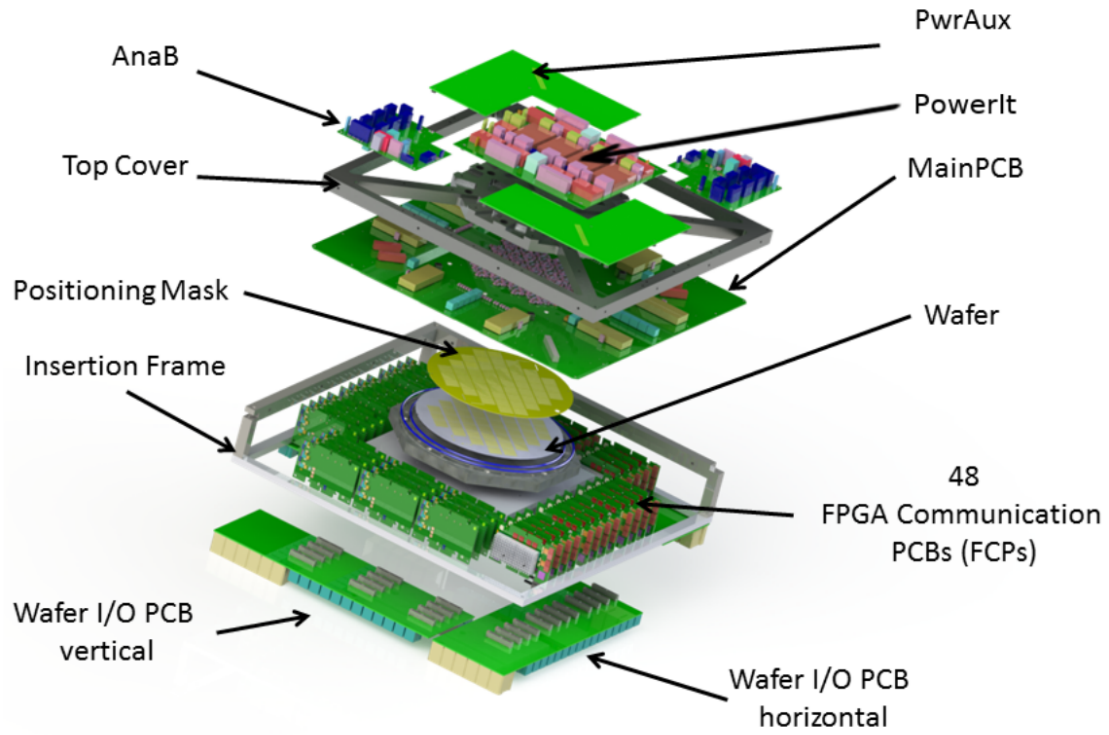


Figure 1: The BrainScaleS system split up into its components, as well as the communication- and powersupply. The PowerIt is responsible for supplying power for the HICANNs, FPGAs, the Wafer-system and all the other components.

consumption can reach up to 2kW per Wafer-system. For this a special board is used, the

[1]High Input Count Analog Neural Network
[2]Field Programmable Gate Array

PowerIt. In addition a Wafer-system includes a Raspberry Pi which controlles the whole system, amongst other things the PowerIt board.

3 The PowerIt

The PowerIt board and its components are subject of this internship. The core of the PowerIt is a micro controller of the firm STMicroelectronics [3].

If one looks at the schematic depicition one can see the placing of the PowerIt in the whole

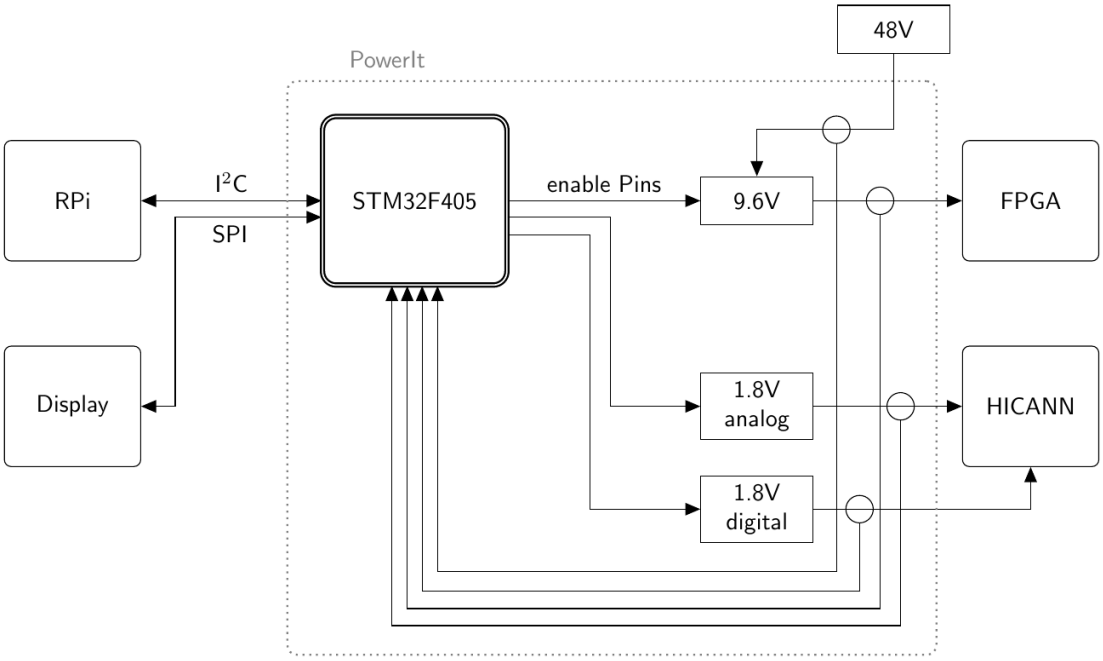


Figure 2: Connections of the STM32-Chip, here visible are the 1.8V and 9.6V voltage controller which are regulated by the enable pins, the measuring points for voltage and current of the board and the controlling with the Raspberry Pi.

system and a simplified representation of the tasks of the chip:

- control of the parts responsible for supplying power via enable-pins
- measuring the current and voltage in the wires of the 48V input and the 9,6V ouput as well as the 1.8V analog and digital output with an internal ADC
- communication with the monitoring system via an I^2C-Bus [1].

4 Experimental setup

For the calibration of the PowerIt a variety of external equipment is necessary. Firstly a bench power supply (EA-PS 8080-120 link) to provide the power for the PowerIt, secondly an electronic load (EA-EL 9080-400 link) and thirdly an external voltmeter (Keithley 2100 6 1/2 Digit

Multimeter link) were used. In addition a Raspberry Pi microcomputer was connected to all the equipment.
To now calibrate the PowerIt the bench power supply could be controlled to sweep through a voltage range, or similarly the electronic load could sweep through different current draw settings.
The initial calibration runs were done with one PowerIt board, B11. It was used to develope and optimise the calibration process and for further testing three other boards (B08, B11, B26) have been used later, too.
The boards were mounted onto the top cover of an empty insertion frame and connected to the other equipment by wires. The wiring is discussed in 7. in greater detail.

5 The four stages of the work

In the following of this report I go through four different stages of my work, I motivate each stage, explain what I did in some detail for each step and present the most important findings.

To gather the data necessary for the calibration a software written in python was already written and in a working condition with a couple of bugs and some missing features. One bug was that in the output file elements from a numpy array were tried to display which resulted in unreadable lines of numbers and letters. That was due to the fact elements of the type float64 can't be displayed in a yaml file [4]. Another bug was a problem with anchors and aliases in the output file with which a smooth continuous work was hindered. So the bugs were fixed and during the whole work always new features were added finalising the software.
In addition a software was created to evaluate the output of the calibration. This software automatically creates plots for each measurement to showcase the values measured by the PowerIt versus the values measured by either the multimeter or the electronic load, depending on the measurement. In addition a file is created with the calibration parameters and the corresponding board stored in it with which the board can be calibrated.

5.1 Optimise the calibration procedure

The most important part now was to shorten the time which it takes to take the data. This was addressed by looking at how much data was gathered during one run of calibration and get its amount to a minimum without sacrificing accuracy. For that the amount of data taken for each step in the sweep of the bench power supply through a voltage range or the sweep of the electric load through different current settings was reduced. The first calibration run was done with 50 measurements during each step. That lead to a total duration of about one hour including rewiring. Then the data was fitted with a polynomial function of second degree. When looking at the fit it showed that some of the fit parameters had rather big uncertanties. That lead to more calibration runs with different amount of measurements for each step of the sweep. The second prominent thing in the data of the first run and also in the following runs was that the uncertainty of the fit parameters of the function fitted to the data from the 48V ouput was a lot higher than the uncertainty of all the other fit parameters (see table 1). This trend was found for various amounts of measuring points.

These findings lead to two key aspects to focus on. First as mentioned before the number of data points and second which fit function should be used for the calibration. The fit function is a tool to determine how good the data is which was taken. It was important to make sure a well working solution was found before reducing the number of data points because with less data the accuracy decreases, too. For all the data that was already taken different function were

	current 48V Measurement	voltage 48V Measurement
0th degree	-2.47687 ± 0.2192	-2.41828 ± 7.44255
1st degree	74.09332 ± 2.00067	31.43227 ± 8.62564
2nd degree	-10.57006 ± 3.93448	-1.0079 ± 2.49666
		voltage 10V Measurement
0th degree		-0.30846 ± 0.42978
1st degree		4.24536 ± 0.35478
2nd degree		-0.04875 ± 0.07314
	current 1.8V analog Measurement	voltage 1.8 analog Measurement
0th degree	-31.61333 ± 0.05678	0.03311 ± 0.01023
1st degree	79.03144 ± 0.13343	0.96391 ± 0.01153
2nd degree	-0.29402 ± 0.07261	0.0098 ± 0.00324
	current 1.8V digital Measurement	voltage 1.8V digital Measurement
0th degree	-30.14358 ± 0.0507	0.01886 ± 0.01678
1st degree	77.72135 ± 0.1198	0.97999 ± 0.01889
2nd degree	-0.36466 ± 0.06535	0.00529 ± 0.00529

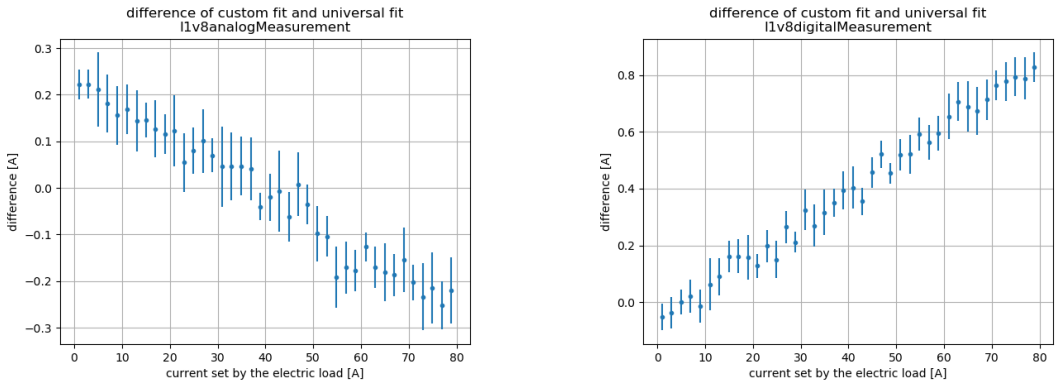
Table 1: Fit parameters of a quadratic function from one of the first measurements. Note that the error for both the current and voltage measurements at the 48V input are big.

fitted. The already used polynomial function of second degree, but also a linear function and a qubic function.
When looking at the paramters of the qubic fit (see table 3) one can see huge numbers with some having even greater errors. That means a qubic fit is inaccurate and not an alternative to the quadratic fit. Then again some are close to zero, that occurs only for parameters of second and third degree. That implies a linear fit is enough here.
If one compares the parameters of the quadratic and the linear fit (see table 3) one finds that for the first and zeroth degree the parameters are similar and the parameters of second degree in the quadratic fit are close to zero or considerably smaller than the parameters for the term of first degree in the case of the current measurement of the 48V input which implies again that a linear fit is sufficient. This is further comfirmed by the fact that the errors of the parameters of the linear fit are very small.
One can quickly see that the linear function is best fit function, fitting the quadratic and subsequently the qubic function is a case of overfitting. Therefore for the rest of the calibration a linear fit for all data sets was chosen.
Now that this problem was solved it was a question of how little data is necessary without sacrificing accuracy. In the beginning, as mentioned before, 50 measurements during each step in the sweeps of the bench power supply and electric load was done. This number got cut down to 35 measurements when taking data from the 48V input and 20 for the other outputs. This cut down in data points lead to a reduction of the time down to about 30 minutes while still keeping accurate calibration parameters. The reason why two different numbers where chosen is because small errors get accumulated and amplified by a factor of 8 before they are read by the PowerIt in the case of the 48V input [2] which can be seen if one compares Figure 6, 7 and 8 in the appendix. That leads to a lower accuracy and more data is necessary to compensate that.

5.2 Determine the calibration parameters

As mentioned in 5.1 a python script was used to evaluate the data taken for the calibration. The output file with the calibration parameters is a .yaml file and can therefore both be read easily

and used for the calibration.
The script takes the data and fits it with a linear function. The calibration parameters and its errors are stored in the output file as well as the number of the PowerIt in order to later match the PowerIt to the correct file more easily.
It was considered to create one universal calibration for all PowerIT boards. To test if that is an



(a) analog output (b) digital output

Figure 3: The current at the 1.8V output calibrated with the custom calibration parameters substracted from the current at the same output calibrated with the universal calibration parameters. The orange line is the difference of both fit functions. This data was taken with the PowerIt board B11.

option from all four available PowerIt boards calibration parameters were determined and every parameter of the universal calibration was an average of the four parameters of the four PowerIt boards. Then each of the four boards were calibrated with the universal calibration parameters and a test was executed by doing a calibration run with the calibrated PowerIt boards. The resulting data was then fitted again and substracted from the results of a reference data set. The reference data set was data taken from the same board but with its custom calibration parameters. If the difference of the results is close to zero one can say only one set of calibration parameters is needed.
Even though some results show almost no difference, the results for measurement of the current at both the 1.8V analog and digital output are off by a few hundred Milliampere and even more at the 48V input (see graphs 3a and 3b).
As a result each PowerIt needs to go through a run of data collection in order to determine individual calibration parameters, universal calibration parameters are not a viable option.

5.3 Getting the calibration parameters onto the PowerIt

In order to write the calibration parameters onto the PowerIt a python class which already existed was extended by the necessary functions. The class can by found in the file powerit.py which can be found on gerrit, repository: power-calibration-sw, change set: 14176. The functions added are for reading the calibration parameters from the PowerIt and writing new calibration parameters onto the PowerIt, as well as reading calibration parameters from a yaml-file.
With the python script give_param.py one can now write the calibration parameters onto the board. The script takes one argument in form of the yaml-file in which the parameters are stored.

```
./give_param.py <YAMLFILE>
```

5.4 Checking the calibration parameters

To test if the calibration was successful a second calilbration run can be executed with the calibrated PowerIt. When evaluating this data a linearity should be visible in the output file. An example can be seen in the appendix B.3 in Figure 13, 14, 15, 16. One can tell very well if the calibration went well when looking at the new fit parameters. When the parameter of 1st degree equals 1 and the one of zeroth degree equals 0 then the calibartion was successful.

6 Further conclusions

During the process of optimisation some other thing were found which are briefly discussed in this section since they are not as important to the work but still need to be metioned.

6.1 Swapping resistors

After optimising the calibration process with only one board, other boards were tested as well in order to find out if the optimisation works for different hardware just as well. But before that could be done parts of the board had to be swapped or removed and the latest firmware had to be flashed on (see BrainScaleS-Wiki).

6.2 Temperature measurement

Another thing the PowerIt measured besides the voltages and currents at the in- and outputs is the temperature of the board. During the internship many runs were done in different circumstances. There is for once the duration of a run, but also a fan was attached to the system which could be turned on or off externally.
When the fan was kept turned off the PowerIt reached temperatures of more than 30 C° during the calibration. But with the fan turned on the temperature quckly dropped down to room temperature again.
An increase in temperature had no impact on the duration of the calibration and also the accuracy of the measurements wasn't effected. That means the fan can be turned of when calibrating it.

6.3 Potentiometer settings

Table 2 was created with which one can check which setting of the potentiometer corresponds to which output voltage. Both a theoretical value was determined and as an example experimental values from one calibration run are shown. Both those values are similar, that's useful. For the column of the experimental values the last two entries are missing. That's because in the calibration process the potentiometer is only set to 240 and not higher but in theory it can go up to 256.

7 Instruction for the Calibration

For the calibration of the PowerIt one needs a couple of software and for the hardware side a power supply, an electric load and a multimeter, as well as cables to attach all of that to the

potisetting	theo. V	exp. V	potisetting	theo. V	exp. V
0	2.026	2.022	140	1.7040	1.707
10	1.9943	1.991	150	1.6881	1.692
20	1.9645	1.962	160	1.6728	1.677
30	1.9362	1.935	170	1.6581	1.662
40	1.9095	1.909	180	1.6441	1.648
50	1.8841	1.885	190	1.6305	1.634
60	1.8691	1.861	200	1.6175	1.622
70	1.8372	1.839	210	1.6049	1.609
80	1.8154	1.817	220	1.5928	1.597
90	1.7946	1.797	230	1.5811	1.585
100	1.7748	1.778	240	1.5698	1.575
110	1.7559	1.759	250	1.5589	
120	1.7378	1.741	256	1.5526	
130	1.7205	1.724			

Table 2: A table to quickly see what potentiometer setting (potisetting) correspnds to what output voltage (V), both theoretical (theo.) and experimental (exp.) values on any board.

board. Here it helps to use multiple cables for the multimeter since it saves time and effort. Also two Allen wrenches size (insert size here) are needed for the screws.

7.1 Setting up the calibration

At first one should connect the wires as depicted in Figure 4, but connect the multimeter to the 48 volts input. This setup will be used for the first three of seven measurements. Those are the voltage measurement of the 48 volts input and the 10 volts output as well as the current measurement of the 48 volts input.
When the PowerIt is all wired up one can turn on the power supply, the electric load and the multimeter and start the software calib.py which can be found on gerrit, repository: power-calibration-sw, change set: 14176 on the Raspberry Pi as a super user, as an argument the software needs the name of the PowerIt, e.g. B11.

```
sudo ./calib.py B11
```

During the first measurement the raspberry sets the power supply to 46 volts and lets it go up to 51.5 volts in steps of 0.5 volts and on each step the raspberry takes 35 data points from the power supply, the multimeter and the PowerIt itself. After that is done the software stops and waits for the user to comfirm that it can continue. Before confirming one has to connect the 10 volts output with the multimeter.
After confirming the software can start with the next measurement. The raspberry once again takes data points following the same principle as before only that it now takes 20 data points per setting of the power supply.
The third measurement needs no rewiring, one can start this one right away. Here the electric load feeds the 10V ouput a set voltage of 9.6V and it varies the current it feeds, starting at 1A and going up to 29A in steps of 1A. The data taken is the setting of the powersupply and what current the PowerIt measures, both 35 times for each current.

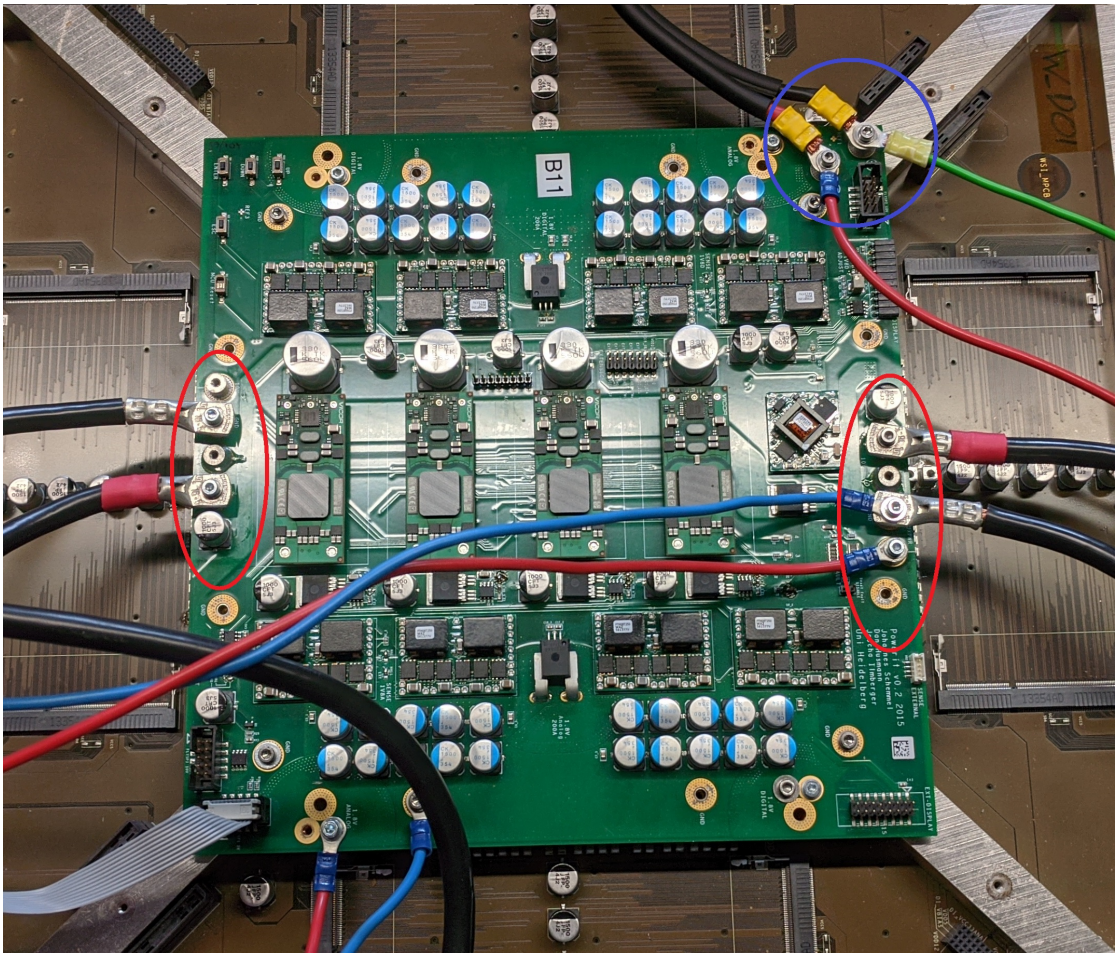
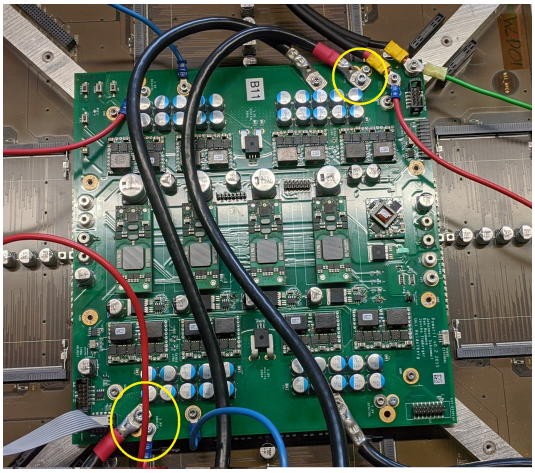
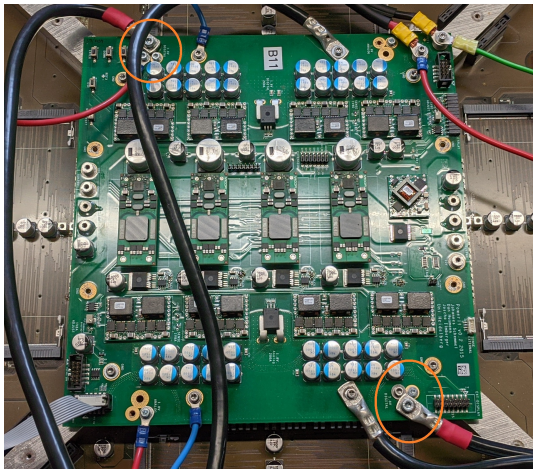


Figure 4: experimental setup for the first three measurements. The 48V input is circled in blue, the two 10V outputs are circled in red



(a) 1.8 volts analog measurements with the analog outputs circled in yellow.



(b) 1.8 volts digital measurements with the digital outputs circled in orange.

Figure 5: experimental setup for the last four measurements

7.2 rewiring for the last measurements

Once that is done one needs to do some more rewiring. Next the raspberry takes data from the 1.8 volts analog output. The rewiring needs to be done as depicted in Figure 5a. Basically the electric load needs to be connected to the 1.8V analog output first, later to the digital output. Also the cables for the multimeter need rewiring. This setup can be used for the next two measurements without doing any rewiring, but be careful at all time that the multimeter is connected to the correct output.
In the first run the voltage of the 1.8 volts analog output is varied by having a potentiometer varying its resistance. That is accomplished by setting the potentiometer to values between 0 and 240 in steps of 10 and each setting correlates with a different resistance. The voltage of the power module and the voltage measured by the multimeter are both taken 20 times per setting by the raspberry.
The current on the 1.8 volts analog output is measured in the same way as the 10 volts output. For the last two measurements only a small rewiring has to be done. As one can see in Figure 5b only two cables need to be moved and also the multimeter needs to be now connected to the 1.8 volts digital output. Here the execution is the same as for the analog output.

7.3 creating the calibration parameters

Once that completed successfully one now can take the output file and determine the calibration parameters. The output file has the form 'yyyymmdd_hhmmss_powerit_test'. To determine the calibration parameters one needs to execute the software analysis.py which can be found on gerrit, repository: power-calibration-sw, change set: 14176. It takes one argument, the file which was just created.

```
./analysis.py <output_file>
```

This creates eight graphs, one for each measurement that was taken plus a plot that shows the temperature of the PowerIt during the calibration. In addition a file is created which holds the calibration parameters.

7.4 evaluation of the calibration

To evaluate the calibration we can go back to the beginning and redo the calibration just like before but this time we can type the following instead.

```
sudo ./calib.py --check yes --file <calibparameter_file>
```

With those two flags the calibration is shorter, only 10 measurement points per setting and the data of the PowerIt after the calibration is read out. When the fitting parameters of each measurement are at about 0 for fitparam0deg and 1 for fitparam1deg the calibration was successful.

8 Conclusion

The goal of this internshsip was to finish the calibration process for the PowerIt and to optimize it. The first part was successfully done which is justified by the resulting calibration parameters, which are all close to 1 for the parameter of first degree and close to 0 for the parameter of zeroth degree.
The optimisation was done in multiple steps. The first step was to optimize the fit function. It was found that a linear fit is better than a quadratic fit. In all cases the parameters for the term of second degree was close to zero or considerably smaller than the parameters for the term of first degree in the case of the parameters of the current of the 48V input.
Fitting the data with a linear function left the parameters for the term of first and zeroth degree almost untouched, also an indication a linear fit is enough, but also decreased the error of those values further.
This shows that a linear fit is reasonable here for determining the calibration parameters.
The qubic fit function produced the worst calibration parameters. That is not an alternative.

The next step in optimizing the calibration process was to reduce the number of measurements per step in the sweeps of the bench power supply and the settings of the electronic load respectively. Different measurement runs showed that the number of data points per step can be cut down to 20 and 35 respectively while keeping accurate calibration parameters. The reason why one needs more data points for the 48V input is because the data here is far more inaccurate. As one can see in Figure 6, 7 and 8 in the appendix, the error is larger, which is due to the fact that small errors get accumulated and amplified by a factor of 8 in the case of the 48V input[2].

A python script was written to get the calibration parameters onto the board. The script takes the output file from the calibration process as an argument and writes the parameters from there onto the PowerIt.

To check if the calibration was successful one can redo the calibration process with the calibrated board and check the results. When the parameter of 1st degree equals 1 and the one of zeroth degree equals 0 then the calibartion was successful.

Furthermore before a calibration can be done with a PowerIt one has to check if the commissioning has be done and if the latest firmware is flashed onto the board. This can be checked here.

References

[1] Silicon Labs. *C8051F3xx and C8051F41x Datasheet*. 2013. URL: https://www.silabs.com/.

[2] Patrick Nisblé. *Calibration and Regulation of the Power Supply in the BrainScaleS System*. 2018.

[3] STMicroelectronics. *STM32F405xx Datasheet (DM00037051)*. 2020. URL: https://www.st.com/content/st_com/en.html.

[4] *yaml file*. URL: yaml.org.

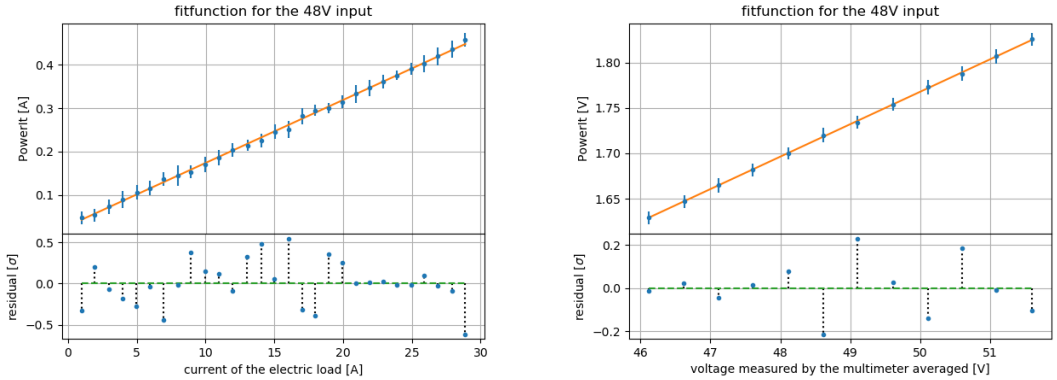
A Tables

	linear	quadratic	qubic
current 1.8V analog Measurement			
0th degree	-31.564 ± 0.028	-31.90 ± 0.07	414.272
1st degree	78.712 ± 0.029	79.52 ± 0.16	-451.3491
2nd degree		-0.45 ± 0.09	131 ± 5
3rd degree			-47.0 ± 2.6
current 1.8V digital Measurement			
0th degree	-29.791 ± 0.026	-30.10 ± 0.06	440 ± 9066176
1st degree	76.916 ± 0.027	77.69 ± 0.14	-473 ± 9066176
2nd degree		-0.42 ± 0.08	125.51 ± 5.26
3rd degree			-44.9 ± 2.6
current 48V Measurement			
0th degree	-2.14 ± 0.14	-2.04 ± 0.26	-102 ± 19812612
1st degree	69.44 ± 0.52	68.3 ± 2.4	138 ± 19812612
2nd degree		2 ± 5	142 ± 19
3rd degree			-179 ± 31
voltage 10V Measurement			
0th degree	0.007 ± 0.007	0.13 ± 0.24	474 ± 5880
1st degree	3.997 ± 0.003	3.90 ± 0.20	-470 ± 5880
2nd degree		0.02 ± 0.04	-0.04 ± 0.09
3rd degree			0.009 ± 0.018
voltage 1.8 analog Measurement			
0th degree	0.0021 ± 0.0014	0.030 ± 0.020	392.81322
1st degree	0.9987 ± 0.0008	0.97 ± 0.023	-391.79504
2nd degree		0.009 ± 0.006	-0.020 ± 0.013
3rd degree			0.005 ±0.004
voltage 1.8V digital Measurement			
0th degree	0.0007 ± 0.0012	0.030 ± 0.017	-835.88186
1st degree	0.9994 ± 0.0007	0.966 ± 0.020	836.8992
2nd degree		0.009 ± 0.006	-0.01946
3rd degree			0.00538
voltage 48V Measurement			
0th degree	-0.11 ± 0.12	2 ± 4	-250 ± 204801
1st degree	28.34 ± 0.07	26 ± 5	279 ± 204800
2nd degree		0.6 ± 1.4	-1 ± 3
3rd degree			0.3 ± 0.9

Table 3: Fit data from an exemplary calibration run. One can see that the parameters of the linear fit have the smallest uncertainty. Furthermore most of the parameters of second and third degree close to zero in both the quadratic and qubic fit. Both these observations indicate that a linear fit is the best option from those three.

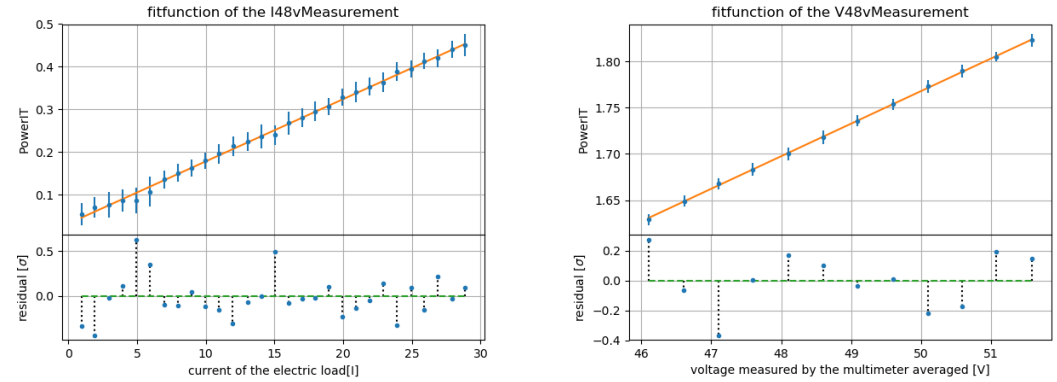
B Graphs

B.1 48V input



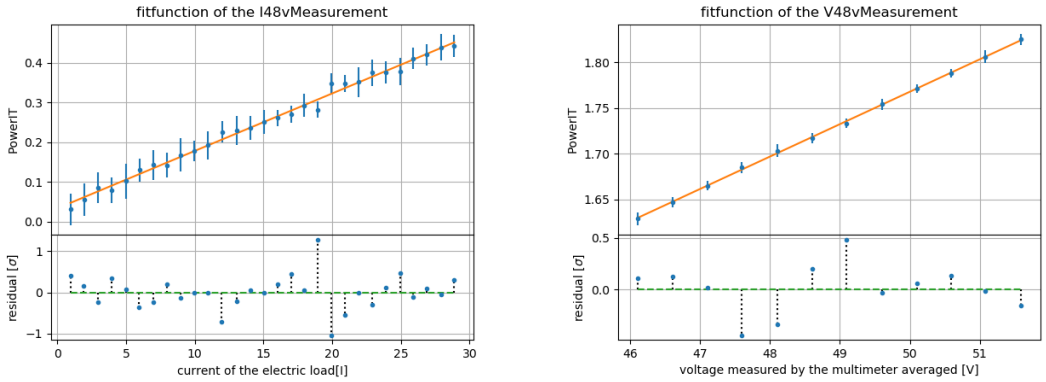
(a) current (b) voltage

Figure 6: 50 measurement points per step in one sweep at the 48V output.



(a) current (b) voltage

Figure 7: 35 measurement points per step in one sweep at the 48V output.

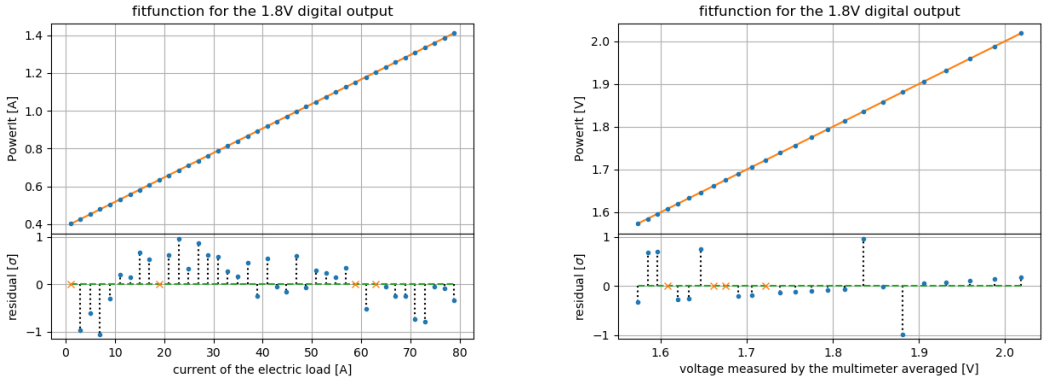


(a) current (b) voltage

Figure 8: 20 measurement points per step in one sweep at the 48V output.

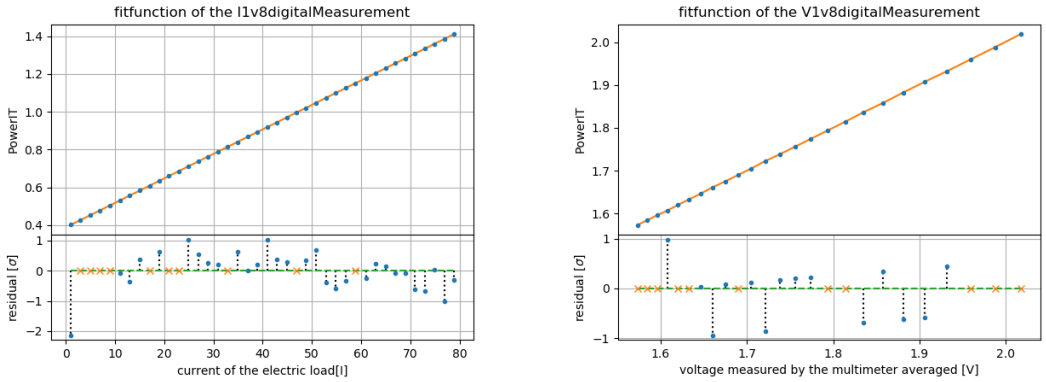
B.2 1.8V digital output

Note: in each subplot where the residuals are shown the orange x implies that the value in the upper subplot has no error in y-direction. That also means for that value no conlusion about its accuracy can be made because no information is given about how far away the data point is from the fit and having more orange x's in the plot doesn't mean a better result was archieved.



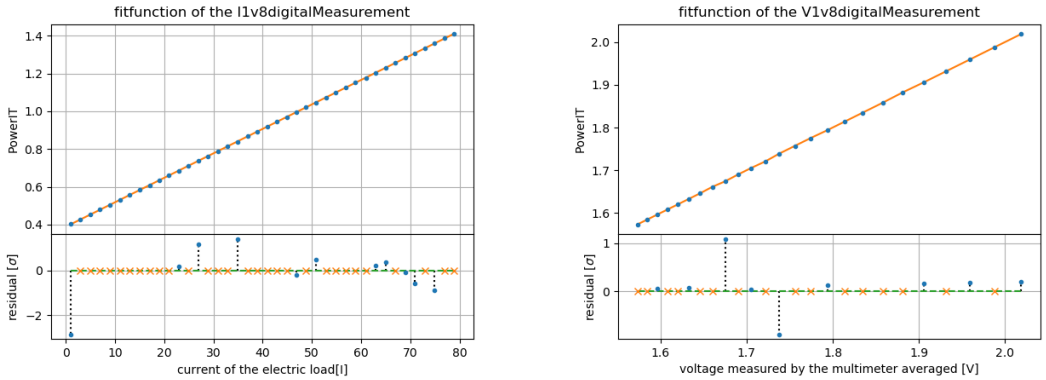
(a) current (b) voltage

Figure 9: 50 measurement points per step in one sweep at the 1.8V digital output.



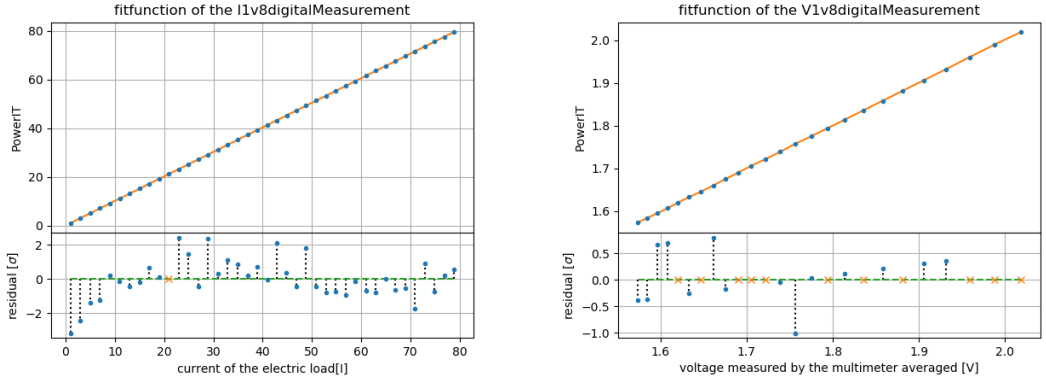
(a) current (b) voltage

Figure 10: 35 measurement points per step in one sweep at the 1.8V digital output.



(a) current (b) voltage

Figure 11: 20 measurement points per step in one sweep at the 1.8V digital output.

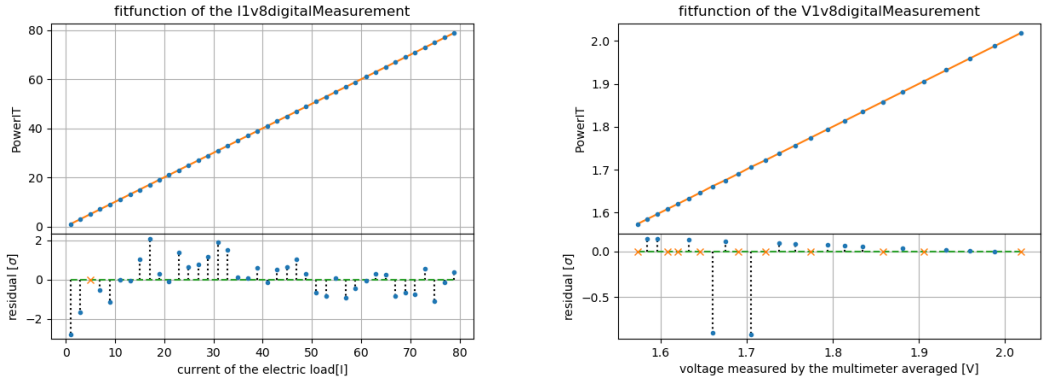


(a) current (b) voltage

Figure 12: 10 measurement points per step in one sweep at the 1.8V digital output.

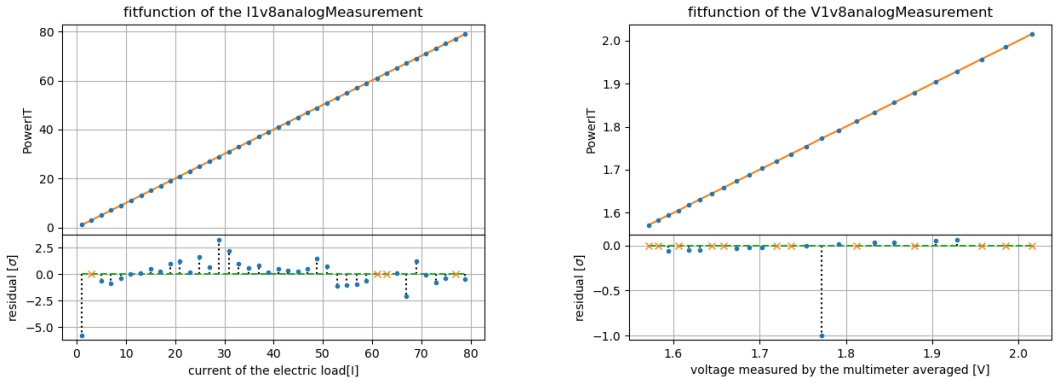
B.3 calibrated board

Note: in each subplot where the residuals are shown the orange x implies that the value in the upper subplot has no error in y-direction. That also means for that value no conlusion about its accuracy can be made because no information is given about how far away the data point is from the fit and having more orange x's in the plot doesn't mean a better result was archieved.



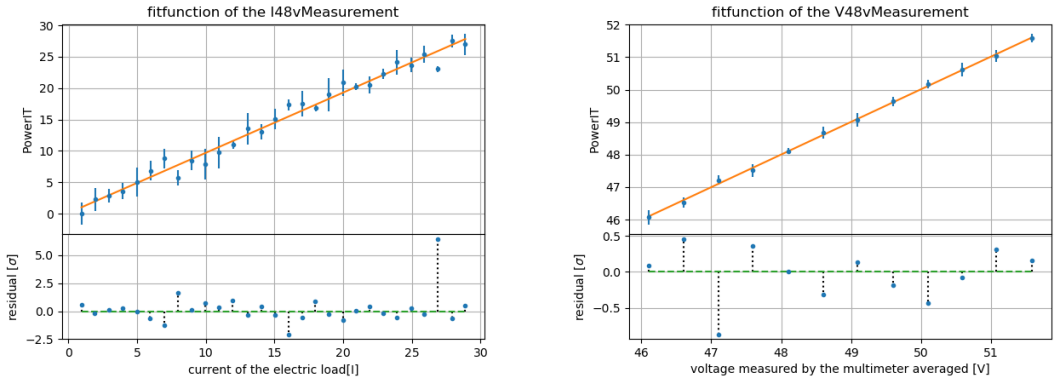
(a) current (b) voltage

Figure 13: Calibrated 1.8V digital ouput.



(a) current (b) voltage

Figure 14: Calibrated 1.8V analog output.



(a) current (b) voltage

Figure 15: Calibrated 48V input.

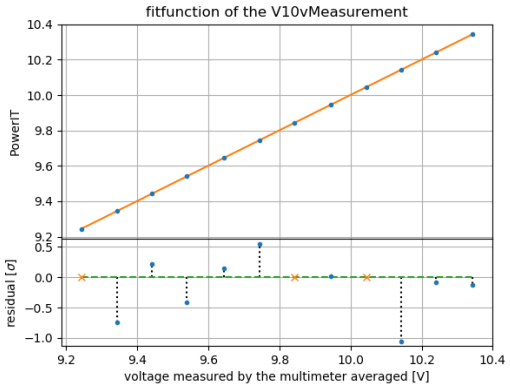
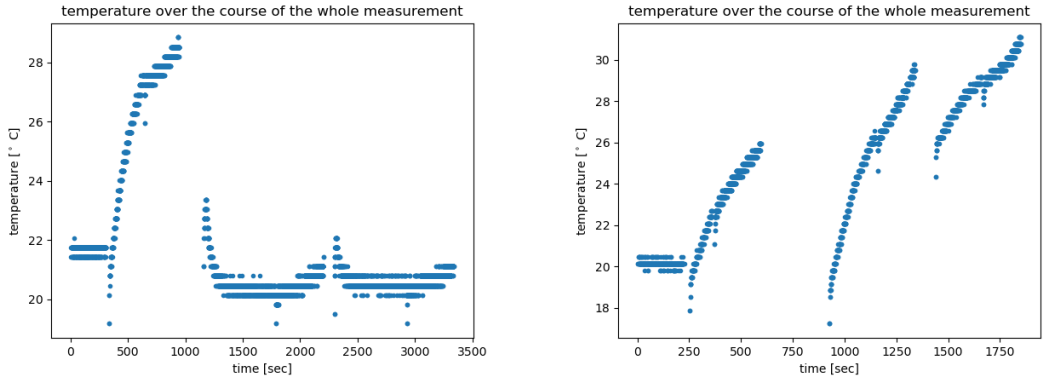


Figure 16: Calibrated 10V output, current.

B.4 temperature



(a) fans were turned on after 17 minutes

(b) no fans were turned on

Figure 17: Two recordings of the temperature. One has no fan turned on, the other one after the first 17 minutes as it can be seen by the sudden drop of the temperature.