Julian David Rath

Design and Implementation of a Single-Shot Carrier-to-Envelope Phase Detector for Ultrashort Laser Pulse Trains

Master's Thesis
to achieve the university degree of
Diplom-Ingenieur
Master's degree programme: Telematik

submitted to

Graz University of Technology

Supervisor
Christian Kreiner

Institute for Technical Informatics

Technical Supervisors
Cord L Arnold
Victor Öwall

Lund University

Graz, October 2015

AFFIDAVIT

I declare that I have authored this thesis independently, that I have not used other than the declared sources/resources, and that I have explicitly indicated all material which has been quoted either literally or by content from the sources used. The text document uploaded to TUGRAZonline is identical to the present master‘s thesis dissertation.

Graz, ______________________________ ____________________________________

Date Signature

Abstract

In modern physics, lasers play a major role in contributing to the knowledge of mankind. Ultra-fast lasers provide the means to make ultra-fast processes visible with ultra-short light pulses. These lasers are complex facilities which require precise and fast measurement devices.

This thesis focuses on measuring methods of an optical experiment which produces a “Carrier Envelope Phase - Beat”. It explains the basics of all used components and methods and how the third generation of this measuring device has been developed at the University of Lund in Sweden.

This thesis starts with explaining the optical sensor element which measures the light pulses and transforms them to weak electrical signals. Further it is described how the signal processing takes place and how the data is saved on a computer to analyze it. With the results of the measurement, the Carrier Envelope Phase can be stabilized. This is important because a Carrier Envelope Phase (CEP)-stabilized laser has a more stable oscillation pattern of the electric field and a higher peak electric field strength. In other words each laser pulse has the same properties and therefore the results of experiments are more reproducible.

Zusammenfassung

In der modernen Wissenschaft spielen Laser eine große Rolle. Mit ultraschnellem Laser können sehr kurzlebige Prozesse, wie zum Beispiel chemische Reaktionen sichtbar gemacht werden. Diese Laser sind komplizierte technische Anordnungen, welche genaue und schnelle Messgeräte benötigen.

Diese Arbeit beschäftigt sich mit der Erfassung eines Lichtmusters, welches von einem Experiment mit einem solchen Laser erzeugt wird (CEP-Beat). Das Gerät ist die dritte Generation einer Serie von Messgeräten die an der Universität in Lund, Schweden, entwickelt wurden.

Am Anfang wird auf die benötigten Grundlagen eingegangen. Von dem Sensor, welcher das Lichtmuster aufnimmt und dies dann in schwache elektrische Signale konvertiert, bis hin zur Signalverarbeitung, die dann in der weiteren Folge die Daten auf einem Computer speichert, um weitere Auswertungen zu ermöglichen. Mit den Ergebnissen der Messung kann der Laser außerdem CEP-stabil gemacht werden. Dies führt zu einem stabilen Frequenzkamm, was in weiterer Folge zu einem stabileren elektrischen Feld und auch einer höheren elektrischen Feldstärke führt. In anderen Worten, Experimente liefern Puls für Puls besser reproduzierbare Ergebnisse.

Sammanfattning

Idag spelar lasrar en stor roll för modern fysik. Ultrasnabba lasrar har egenskapen att kunna mäta snabba processer med hjälp av extremt korta ljuspulser. Dessa lasrar är komplexa anordningar som kräver snabba och precisa mätningar. Detta examensarbete fokuserar på att utveckla en metoder för att mäta utsignalen av ett optiskt experiment som producerar "Carrier Envelope Phase svävning".

Arbetet förklarar grundläggande metoder och komponenter som använts. Arbetet förklarar även hur slutprodukten kom till att bli den tredje versionen av sitt slag på Lunds Tekniska Högskola.

I början av arbetet presenteras den optiska sensorn som tar emot ljuspulserna och transformerar dessa till svaga elektriska signaler. Det beskrivs sedan hur signalen behandlas och hur den sparas på en dator för vidare analys. Resultaten från mätningarna används för att stabilisera Carrier Envelope Phase. En CEP-stabiliserad laser har en mer stabil oscillations mönster såväl som maximala intensiteten det elektriska fältet. Med andra ord har varje puls samma egenskaper vilket innebär att resultaten från experimenten är mer konsekvent.

Credits

The thesis work was done at the "Lunds Tekniska Högskola (LTH)" in Sweden. All the practical work was done together with my collogue David Etienne, which I wanted to thank in the first place for his persistence and pushing forward in all situations. Namely I like to express my thanks to the following persons at Lunds University: Cord Arnold; Viktor Övall; Miguel Miranda; Anne L'Huillier; Liang Liu; Fredrik Edman; Bertil Lindvall; Martin Nilsson for their dedication and help.

At the Technical University of Graz, Christan Kreiner made it possible to write my thesis in Graz about the work I've done in Sweden.

The whole time I spent at university all my parents have been a great support. Also I thank them for never having doubts in what I have been doing. Special thanks are going to my girlfriend Teresa, always being a big idol and therefore a great implicit motivator for the whole time I spend my time in school. Coming to this point would have been impossible without the support, collaboration and knowledge of my colleges I met.

Contents

List of Figures

1. Introduction

High Speed Lasers are crucial for modern science in atomic physics and chemistry. Miguel Miranda from the Lund Laser Group has written a excellent description for what it is used in his PhD-thesis:

" What is a photographic camera flash good for? The easy answer is 'to illuminate'. There is more to it though: the duration of a camera flash is usually much shorter than the shutter speed of the camera. This allows us to take sharp pictures of fast objects with an inexpensive camera. Ultrafast science is based on a similar principle: no shutter is capable of opening and closing fast enough to "freeze" the motion of molecules breaking up and forming new ones on a chemical reaction; or, much faster, electrons "spinning" around the nucleus of an atom. The trick is to use very short light pulses (our flashes). Events like the ones described take place in times as short as femtoseconds and attoseconds, respectively. If we want to see what happens, for example, during a chemical reaction, and not only the before and after, we need a flash shorter than the time it takes to occur.

But how short is a femtosecond? And an attosecond?
A femtosecond is 0.000000000000001 seconds (or 10^{-15}*s), and an attosecond is one thousand times smaller. To put it in perspective, suppose you have a clock and that, at each second, your clock would fall behind one femtosecond. How long would it take for it to be one second off? It would take longer than thirty million years. There is a fundamental limitation to how short a light pulse can be. Light is an oscillation, or vibration, of the electric and the magnetic fields, that propagate as waves. Visible light, that our eyes can perceive, has oscillations periods of about two femtoseconds, and a light pulse cannot be shorter than that. To create even shorter pulses, we have to go higher in the frequency spectrum, towards X-rays. Light pulses with durations of some femtoseconds are nowadays generated directly from lasers. These laser pulses can then be used to interact with matter and generate light at higher frequencies, and even shorter pulses can be created, with durations of around one hundred attoseconds. "* – [22, Popular Science Summary]

This theses is about a device which is able to measure the relative CEP from such High Speed Lasers from the output of an experiment. This experiment is described in a paper from the Laser Group in Lund [11]. The experiment produces a one dimensional sinusoidal light pattern with the laser repetition rate speed as the laser runs. That means if the laser pulses 200 000 times a second, there will be 200 000 measurements per second.

In my thesis I would like to describe the device which is build to capture these sinusoidal light patterns and extract all the information needed. That implies an amplification and digitalisation of the signal. With the digital signal several mathematical operations can be performed to extract the actual data wanted.

1.1. Evolution and Challenges

Thomas Fordell and his team developed a detector which is able to measure and amplify 70 000 measurements[11] of a light pattern which was thrown on a sensor linear in space. This light pattern is generated by an optical experiment. The device (Figure 1.2) is able to measure single so called-laser pulses.

An open issue was still the feedback itself. In the digitalisation process the space of improvement is wide as well: the capture mechanism did not capture all sensor elements at the same time, where Thomas suggested to use a Sample and Hold element to solve this problem. Last but not least the maximum acquirable repetition rate should be raised to be able to do measurments the 200 Khz laser at the Lund Laser facility.

A cooperation between the Lund University and Friedrich-Schiller-Universität in Jena requested the same experiment with a laser a 4KHz repetition rate. Under time pressure another device was developed using a different sensor with a included driver, capturing all sensor elements at the same time. That sensor simplified the work and the device for the measurement in Jena was finished in time. It then was presented at the Lund Circuit Design Workshop in Sweden [10]. Finally the device was well tested with the 1 KHz laser in

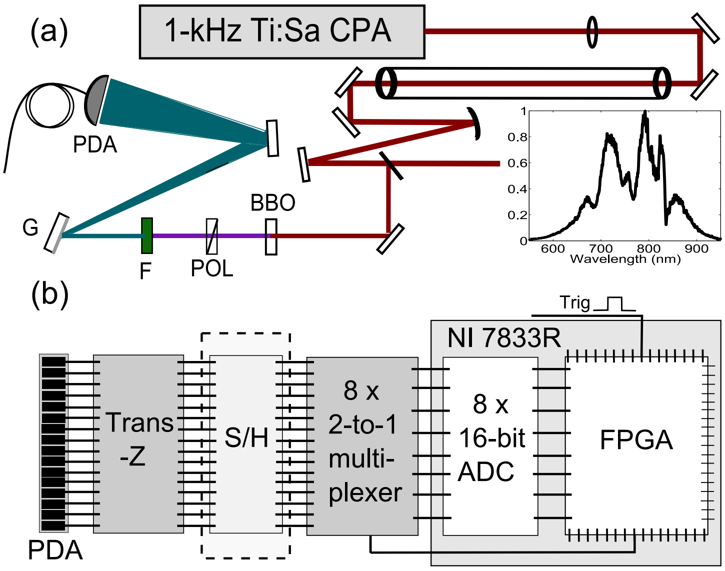


Figure 1.1.: The experimental setup [11]. It shows how the light pattern is generated by the experiment, which then is captured by a photo diode array. The data then is processed by a National Instruments FPGA.

Lund, but unfortunately it was never tested in Jena (the device can be seen in Figure 1.3).

Further on the goal was to improve the capturing process. A photo-diode-array with 35 elements and the possibility to read each of them independently similar to the one from the first iteration should ensure high frequency measurement and full control of the measurement process. That should give the possibility measuring the light pattern in the 200 KHz repetition regime and implement some of Thomas Fordells suggestions. Further using 32 instead of 16 sensor elements increases the resolution of the measurements.

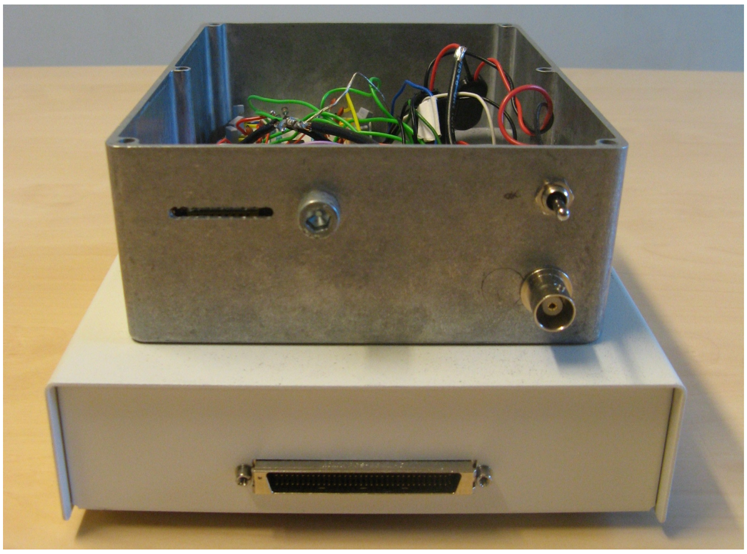


Figure 1.2.: First Generation Device from the Team around Thomas Fordell [11]

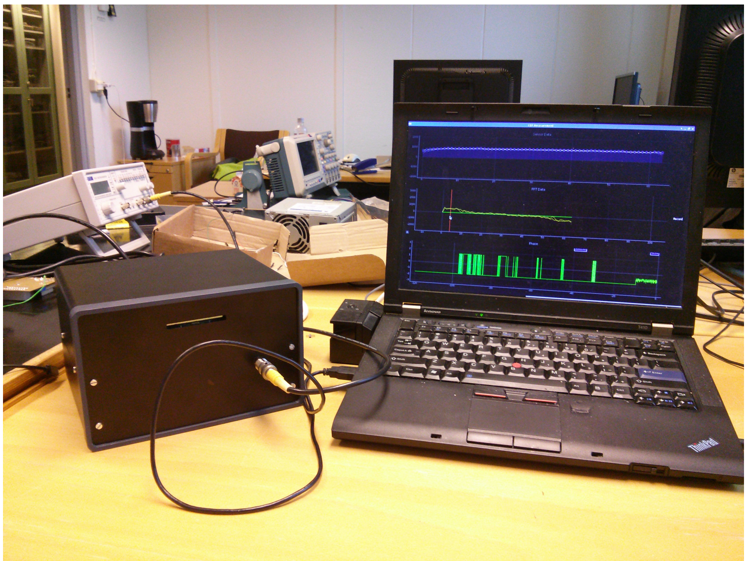


Figure 1.3.: Second iteration of the Project attached to a laptop with the visualisation. Behind the slit the sensor is placed. For test purposes the trigger is here connected to a signal generator.

2. Related Work

In this chapter all theoretical basics, concepts and building blocks which are used in the device are explained. The level of detail provided should be enough for a technical reader and for more details the according references are provided.

First the paper, on which this thesis is based on is described.

2.1. Measuring the CEP Offset

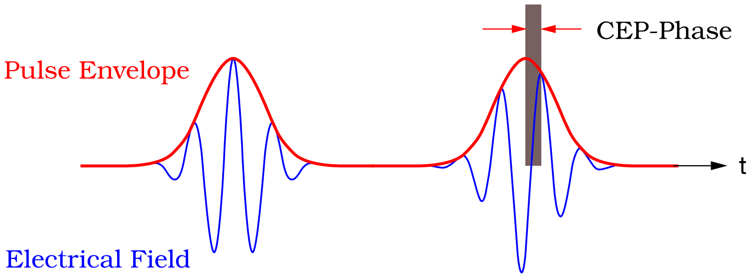


Figure 2.1.: CEO shown in the time domain with the envelope and the inner electrical field.

The pulses of ultrafast lasers consist of the obvious pulse itself. For a long time it was expected that the electrical field within exhibits strong fluctuations (Figure 2.1). To characterize and measure them was another story and was considered to be the "holy grail" for ultra fast lasers. The CEP or CEO represents the jitter of the electrical field within the pulse envelope in the time domain. Measuring the CEP directly in the time domain is hard since it is in the terahertz range [19].

A frequency comb is an optical spectrum with equidistant frequencies (Figure 2.2) resulting from its generation in a resonator. The frequency comb is just red after the cavity is broadened to the full visible spectrum by a halo-fiber element. The comb is then taken and shifted by the factor two with a f-to-2f element. The original comb and the one with the doubled frequencies are added. The laser beam is disassembled to its frequency components with a spectrometer in space. The overlapping of the two combs, which is called CEP-beat can be observed by the device presented in this thesis. The relative CEP change between the pulses can be measured with the CEP-beat.

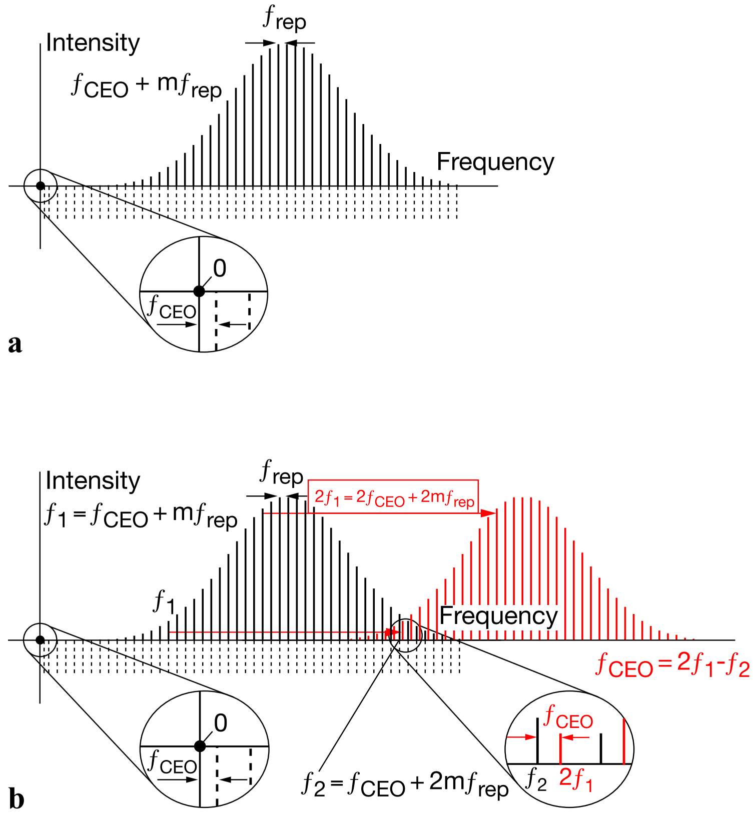

f_{rep}
Intensity
$f_{\mathrm{CEO}} + mf_{\mathrm{rep}}$
Frequency
0
f_{CEO}
a
Intensity
f_{rep}
$f_1 = f_{\mathrm{CEO}} + mf_{\mathrm{rep}}$
$2f_1 = 2f_{\mathrm{CEO}} + 2mf_{\mathrm{rep}}$
f_1
Frequency
$f_{\mathrm{CEO}} = 2f_1 - f_2$
0
f_{CEO}
f_{CEO}
$f_2 = f_{\mathrm{CEO}} + 2mf_{\mathrm{rep}}$
f_2
$2f_1$
b

Figure 2.2.: The CEP in the frequency comb [19]. In **a** the comb before the f-2f. **b** the comb after the f-2f element added with the original comb. The overlap is called CEP-beat.

From a functional point of view the system from Thomas Fordell worked like showed in Figure 2.3. The measurement part in the figure was done by a silicon based photo diode array as described in Section 2.2. The signal processing part is done by an field programmable gate array (Section 2.6. No feedback was done this time.

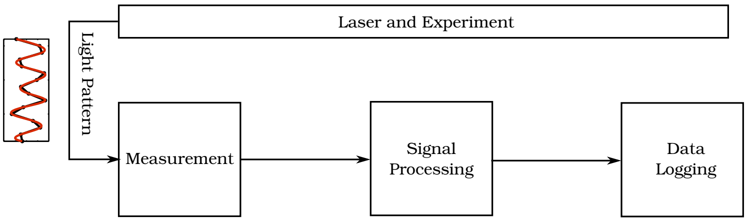


Figure 2.3.: Functional overview of the measurement setup from Thomas Fordell [11].

In the signal processing part the following steps are done the measure the relative phase shift $\Delta\varphi$:

1. Record the light pattern from the sensor (Figure 2.4). The sensor is a photo diode array on one dimensional basis. In this particular case it has 16 elements. The red line does interpolate the enveloped frequency.
2. Do a Fourier transform of the recorded light pattern (Figure 2.5). This converts the CEP-beat into the frequency domain.
3. Select an index (or frequency point) in the frequency domain. This is done by hand. Usually it is the first frequency peak after the DC part. From that point the phase-shift is computed.
4. Calculate the angle of that position. $\Delta\varphi(s) = atan\left(\frac{Im(s)}{Re(s)}\right)$ Doing this repeatedly enables one to see the relative shift from shot to shot.

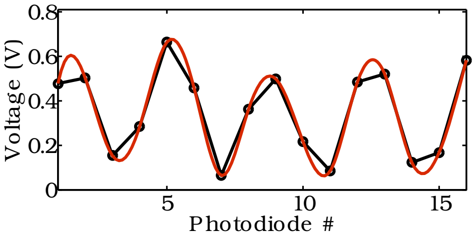


Figure 2.4.: Raw captured sensor data [11]. It represents the light input of the 16 photo diode elements of the sensor. The diodes are ordered linear 1 dimensional.

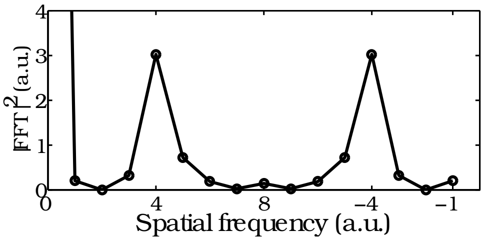


Figure 2.5.: The data [11] from 2.4 Fourier transformed. The frequency peaks can be seen. Based on the peaks the phase can be computed. Here the frequency at index 5 is used for that process.

Then it is acquired to a data recorder which is in the case of the paper a personal computer with a National Instruments PCI-card. Afterwards scientists can analyze the data off-line on their computers.

2.2. Laser Light Sensors

Acquiring data first needs a source. In this case it is electric and comes from a light sensitive photo diode sensor array. Its the same principle used in a digital camera, but just one line of "pixels", because only one dimension is interesting.

The kind of light sensor was given by the first device as described in Thomas Fordell's paper [11]. He used silicon base photo diode sensor array from Hamamatsu (Type S4111-16Q [28] Figure 2.6). To improve the measurement accuracy the project specifications required us to use the 35 element version of the same type (S4111-35Q).

In general a photo diode is a semiconductor component which converts light into a current. Its built similar to a common diode and therefor consists out of a P and a N-Layer. The material of the P-Layer is chosen to be light sensitive. Current is generated by absorbing photons in the semiconductor material. A small current is also produced when no light is present, that is called the dark current. Commonly best known photo diodes are photovoltaic power generating cells, which are just large photo-diodes.

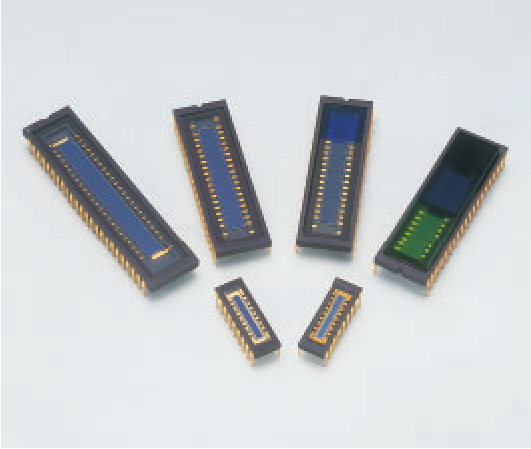


Figure 2.6.: The Hamamatsu S4111/S4114 Series Silicon Photo Diode Array (from [28])

Semiconductor Level

The PN-junction here (Figure 2.7) operates as normal diode. The P-layer just is additionally photo-sensitive, which means that light-particles can strike out a electron-hole pair. If that happens the holes are moving towards the anode and the electrons towards the cathode and a photo current is generated.

The area of the P-layer, which is the visible part of the sensor, is proportional to the sensitivity of such a sensor. Increased area means here as well more parasitical junction capacity, which lowers the reaction time (low pass filter). This effect enables it to measure short laser pulses after all.

The Hamamatsu S4111/S4114 sensor series is implemented as a ceramic DIP-housing with 40 pins (Figure 2.8). The photo diodes in the array are quite large (Figure 2.8) for photodiodes, which has its up and downsides as mentioned in section 2.2.

In the optical experiment white light is generated (spectrum in Figure 1.1). That means that the whole spectrum of visible light could be used to detect the light pattern in the output. The expected light-intensity was unfortunately not known upfront. It was assumed that this could be found out by testing amplification circuits with raw red laser light in the laboratory.

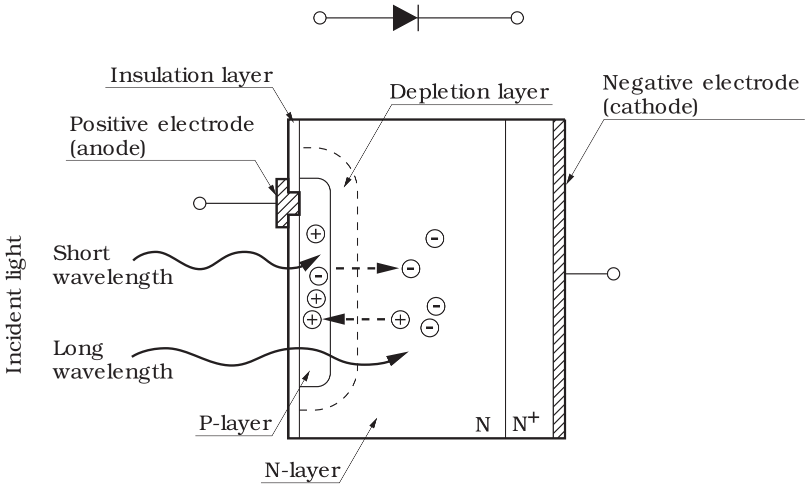


Figure 2.7.: Semiconductor level of a photo diode (from [24]). Light strikes out holes from the P-layer and then they build up a voltage on the N-layer.

This implies of course that this measurement is not linear in spectrum since the sensor is 2-3 times more sensitive in the red frequency ($\approx 750\ nm$) than in the blue range ($\approx 400\ nm$) as it can be seen in Figure 2.9. Visible light is in the range between $\approx 400\ nm$ and $\approx 750\ nm$.

2.3. Photodiode Amplifiers

To measure the light intensity falling on a photo diode there are basically two different options :

- **Photovoltaic Mode**: (Figure 2.10 on the right side) Here the photodiode is used like a solar cell. The flow of the photocurrent is restricted and due that a voltage builds up. This is precise but slow.
- **Photoconductive Mode or Reverse Voltage Mode**: (Figure 2.10 on the left side) Here the diode is driven reverse biased(= reverse voltage). This decreases the response time because the reverse bias increases the depletion area and the capacitance is decreased. That increases at

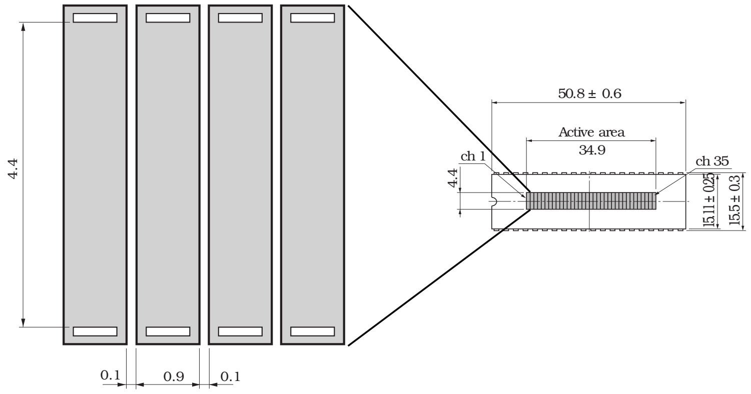


Figure 2.8.: Sensor Elementsize (from [28]). The parasitic capacitance of photo diodes is proportional to their size, which involves a lower acquisition speed with more area. The sensitivity is as well proportional to the size of the sensor. Here the width of the sensor array can be seen as well. The light beam of the experiment should be focus on that area.

the same time the dark current. As seen in the diagram (Figure 2.10) a higher linearity could be maintained as well.

A straight forward circuit now is measuring the voltage drop over a resistor after the photodiode. That implies that there is no reverse voltage and we are using the diode in voltage mode. This can be done by directly connecting a voltmeter parallel to the diode. The inner resistance (R_{sh} in the schematic) of the diode (Figure 2.11) helps to build up a voltage and so we are able to measure a voltage proportional to the light impinging on the sensor.

To use the diode in reverse voltage mode we need to apply the reverse voltage to the diode (shown in Figure 2.12), where V_R sets a voltage to the diode and the current can be measured over the shunt resistor R_L.

In practice one would use an operational amplifier [16] as transimpedance amplifier (as done in [11]). To explain this circuit we neglect the zero-point of the operational amplifier at the "-" current-wise. The only way the current I_{sc} will use is the path over the R_f where it generates a voltage drop $V = R_f I_{sc}$ since this voltage is dropped on the "-" side of the operational amplifier its output will be $-(R_f I_{sc})$.

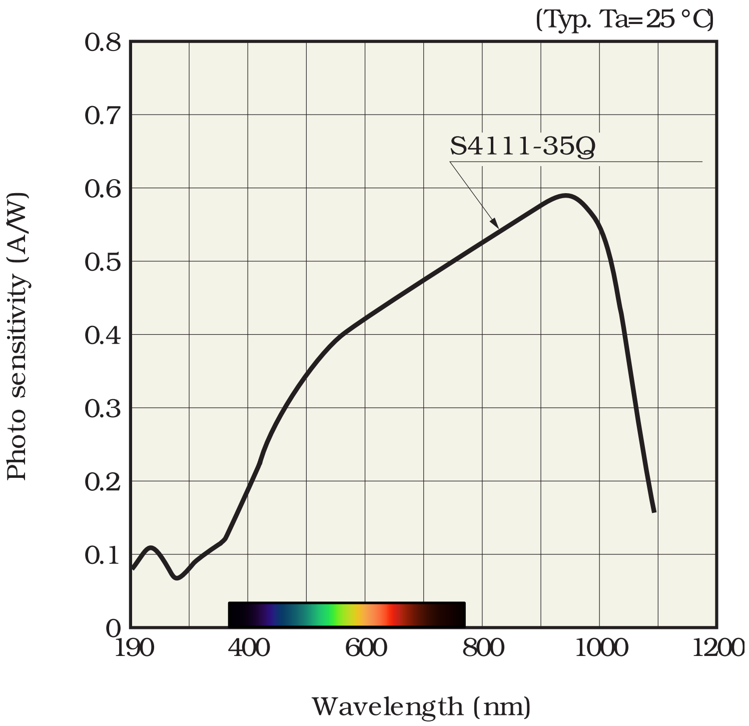


Figure 2.9.: Spectrum of the S4111-35Q (from [28]). The light produced by the experiment will be between 600 and 900 nm as seen in Figure 1.1

2.4. Analog Digital Converter

Analog Digital Converters (ADC) converts currents or in most cases analog voltage signals into digtal signals. In the digital domain the signal can then be further processed by signal processing systems as explained in section 2.6. There are different approaches how to digitalize an analog signal, but here we like to see ADCs from a more abstract view. We are interested in its specifications an most importantly its parallel acquisition capabilities, sample rate and the aperture time.

The sample rate describes how many samples the ADC can convert per time unit. Normally this value has the unit samples/second. In our case we need to aquire at least 200 000 samples/second.

Parallel acquisition means here, that if multiple analog channels are cap-

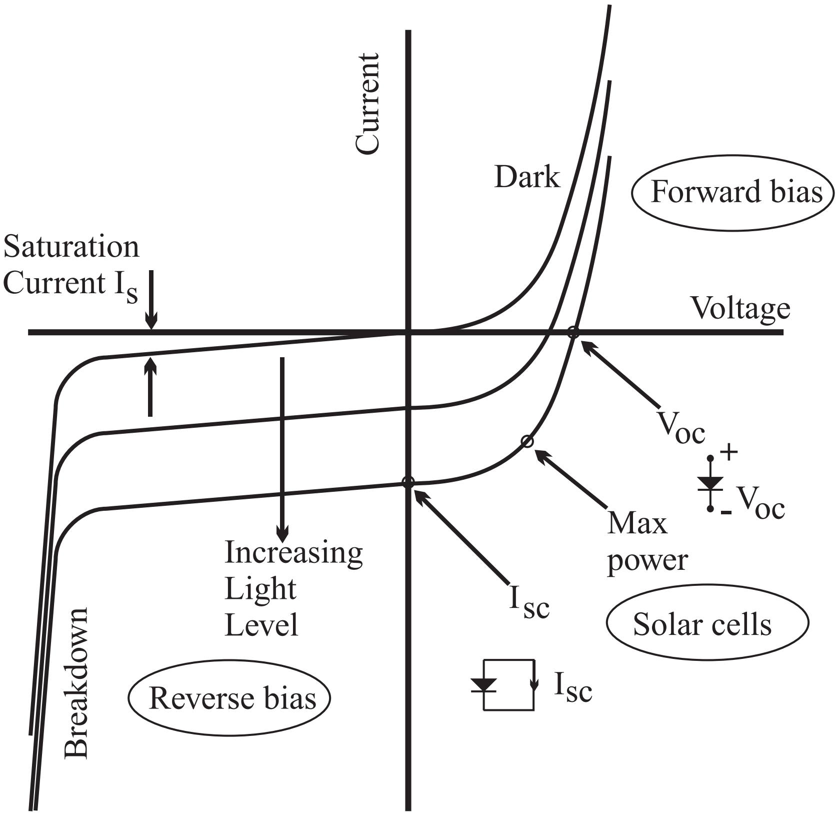
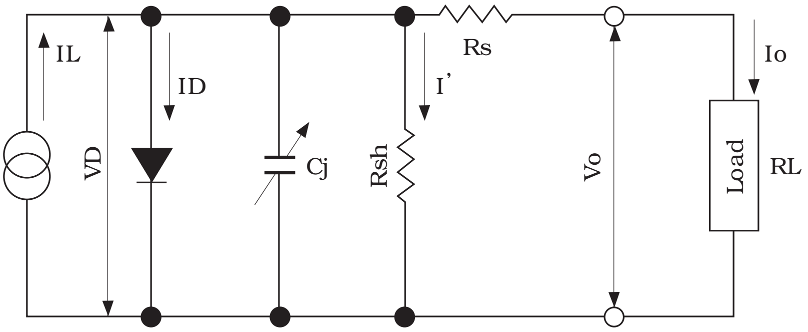


Figure 2.10.: Current vs. Voltage Characteristic [15].

tured, all of them are captured at the same time. A naive approach is to use one ADC per channel, *N* channels, *N* ADCs (Figure 2.14). Multiple sample and hold elements (A simple sample and hold element can be seen in Figure 2.16) are an alternative and implying the ADC to have *N* times more sample rate as needed by the application (Figure 2.15). The approach has one major advantage: simplification, the implementation does not need to control two different component types. The drawback might be the costs, since a sample and hold element is cheaper then a ADC.

The aperture-time is like in photography the time the ADC takes to follow the voltage without actually digitalizing it. It is normally the delay between the 'outside world' and the sample and hold element in the ADC [3] (Figure 2.17).



IL : current generated by incident light (proportional to light level)
VD : voltage across diode
ID : diode current
Cj : junction capacitance
Rsh: shunt resistance
I' : shunt resistance current
Rs : series resistance
Vo : output voltage
Io : output current

Figure 2.11.: Equivalent circuit for a photodiode [24]

Thomas Fordell used in his experiment an 8 channel parallel ADC where he converted 8 of his 16 signals at the same time, one time at the rising and one time at the falling edge of his signal. He suggested to use an additional sample and hold element to capture all signals at the same time [11]. We are not bound to the National Instruments hardware he used. That opens up the possibility to take a different approach in account: use a 32 channel ADC. This has the advantage that all of the 32 channels can captured at the same time. No additional sample and hold elements are needed, the data is directly processed by an FPGA afterwards.

2.5. Serial Peripheral Interface

For the communication between different digital components Serial Peripheral Bus is a quite popular solution. It does play a keyrole in this

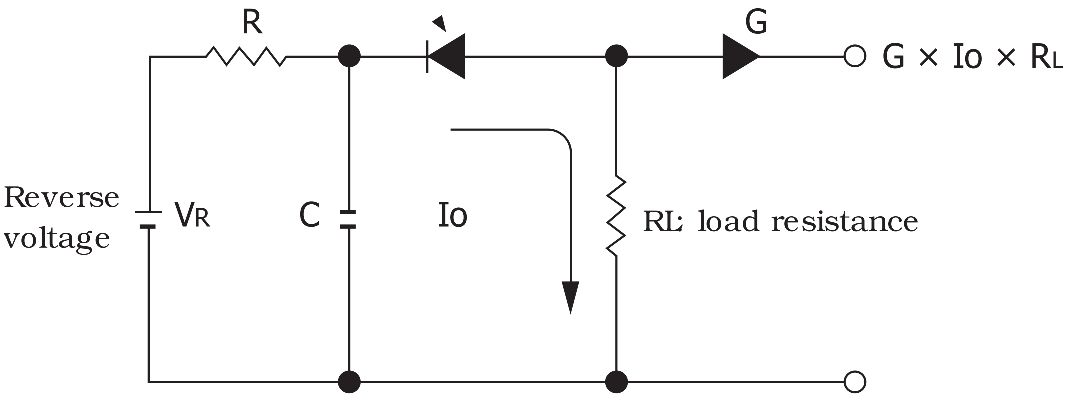


Figure 2.12.: Reverse voltage circuit [24]. The diode is connected to a voltage source and reverse-biased. This reduces the size of the depletion layer and make the diode more sensetive and reduces the parasitic capacity, but increases noise.

project as well. All digital building blocks in this project are using SPI as communication bus between them.

SPI is a simple serial bus-system originally invented by Motorola, but became a de-facto standard for a simple serial bus. It is able to transport data bi-directional up to several Mbit in full-duplex.

There are two roles in the SPI protocol, master and slave. The master sets the clock in and selects which slave it is talking to. A typically hardware setup is to use circular buffers on the slave and the master side (Figure 2.18).

In the standard configuration it uses four wired connections:

- **MISO** (Master In Slave Out) or **SDI** (Serial Data In): Line to transfer the data from the slave to the master. If the **Slave select Line:** is low for a slave it should set this pin to high impedance, to not disturb the transmission to other slaves (if there are any).
- **MOSI** (Master Out Slave In): or **SDO** (Serial Data Out): This line is used to transfer data from the master to the slave.
- **CLOCK**: The clock line is set by the master, it tells the slave when the next bit should be set to the MISO by the slave and when the next bit was set to the MOSI line. A slave only listens to it if the **Slave Select Line** is low.

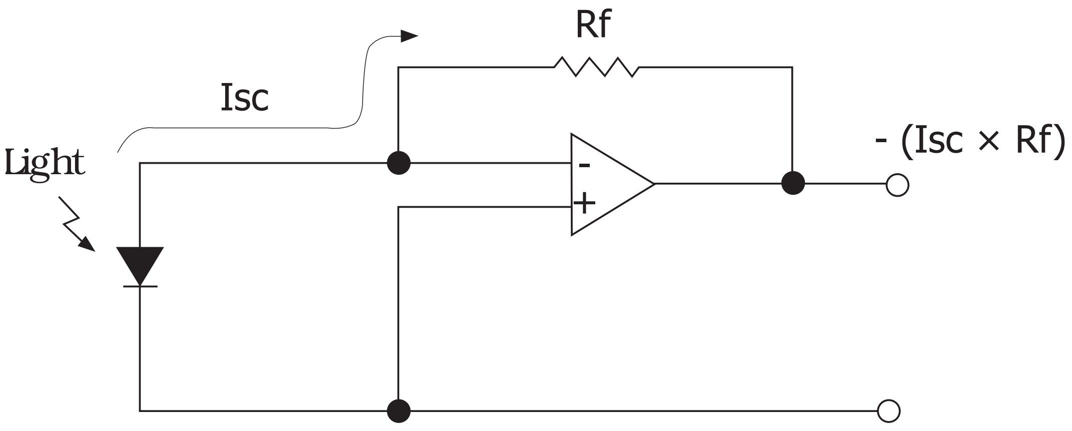


Figure 2.13.: Reverse Voltage Circuit with a Opamp used as Transimpendance Converter [24]. This is the most common circuit for fast conversion operations.

- **FS:** (Frame Sync), **SS** (Slave Select) or **CS** (chip select): Normally this is an inverted line and could have two functions: if there is just a point to point connection between master and slave it could be set to logical 'true' all the time, which means that the slave will always react to incoming clock signals. This might have the disadvantage if a clock signal is indicated by electromagnetic interference. It might also be used to synchronize a transfer, to avoid unwanted clock recognition by the slave. The signal can also be used to select one slave out of many, if needed.

The **MOSI** and **MISO** lines can exist several times to increase the bandwidth.

To connect multiple slaves to the same masters the slaves could be either daisy-chained (Figure 2.19) or just selected by the frame sync line (Figure 2.20). In the daisy chain mode, the slave is using looping the input to output with the delay of the word length [16].

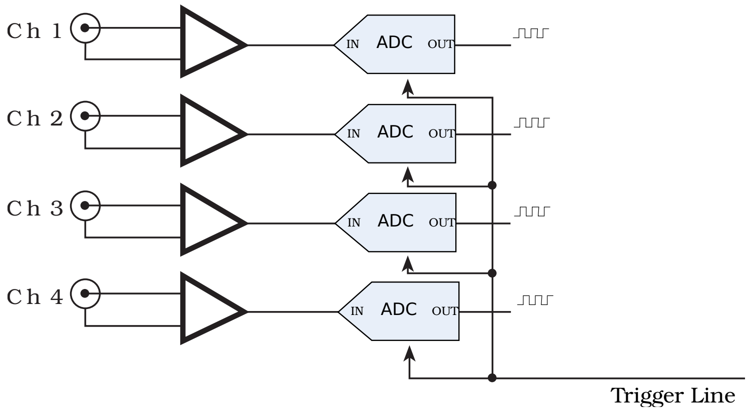


Figure 2.14.: Multiple analog channels with multiple ADCs. All channels a digitalized at the same time. Her the counterpart of the bus needs to have a parallel interface.

2.6. Digital Signal Processing Platforms

Thomas Fordell used in his work already a FPGA to do the mathematical work needed to calculate the phase per shot. This section will elaborate the differences between the different platform choices and why a FPGA is still the best solution.

The abbreviation DSP stands for Digital Signal Processing as well as for Digital Signal Processor. The first term is meant to express the task of a Signal Processor, but that job can be done by other architectures as well. These architectures are the topics of this section.

Signal Processing is not dedicated to the digital domain, it can be done in an analog circuit as well. Signal processing usually is the process of applying a mathematical function to an input signal to generate an output signal with the applied function. In the analog domain this could be just a capacitor, which might be used as high or lowpass filter, depending how it is connected to the circuit. In the digital domain the input signal needs to be digital to be processed by a platform, which is capable to process the data within the given constraints. If the example with the capacitor is directly compared to the digital system, the major constraint is the real-time ability of the digital system [31].

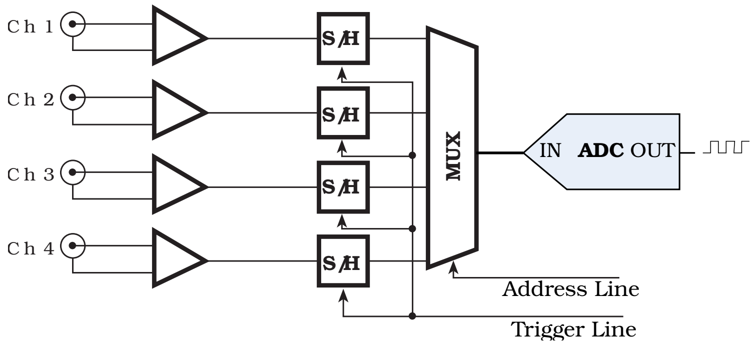


Figure 2.15.: Multiple channels with one ADC and sample and hold elements. Here the value of the channels are save analog and they are digitalized sequentially.

Additionally the input has to be digital for a processor. The digitalization is often done with a Digital Analog Converter, as described in Section 2.4. If the output signal is needed in analog form as well this could be done with a analog to digital converter.

Due the selection of the ADCs in this project a special constraint opens up: high bandwidth with 16 SPI lines running at 40 MHz need to be fed into the digital signal processing platform. Since that is a major requirement, I will discuss all platforms from that perspective here.

PC Workstations

The first thing one might think about is a common personal computer as seen in any modern western household. Even if the computing power is be enough, one would have several problems to use a standard PC for a application like this:

- **Size and Energy**: To have enough computing power to implement a software on the computer which is able to do the signal processing operations and the control task would take a relatively fast processor. That would imply a complex design in thermal and electrical manners.
- **Realtime Operating System**: The complexity of modern PC-Workstrations as well requires to use an operating system. For the described application the whole process timing has to be predictable [30]. Another

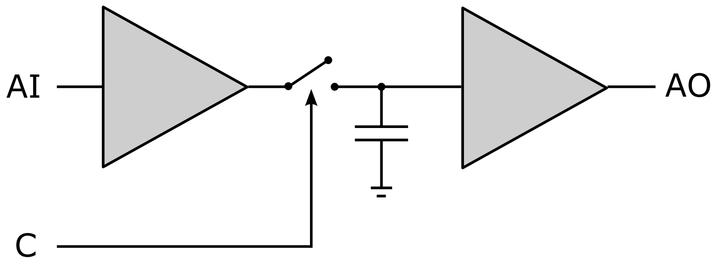


Figure 2.16.: A simplified sample and hold circuit from [7]. The first buffer is to charge the capacitor fast and not influencing front end by having a high impedance. The high impedance of the second buffer prevents the capacitor from discharging. In the time between the capturing and the actual read-out the capacitor will loose voltage since it has an internal resistance. This should be considered when working with sample and hold elements.

constraint is as well, that the calculation including the readout of the ADC and a writeback to the Digital Analog Converter (DAC) has to be done in less then $5\mu s$.

- **ADC Links**: The complexity of here chosen ADC link, which already requires 16 serial lines without control signals demands a high count of parallel accessible pins as well. That can not be provided by any common CPU.

Digital Signal Processer

With the DSP some of the CPU problems can be solved:

- **Size and Energy**: The computing power specifically for the Fast Fourier Transform (FFT) operations would be enough here.
- **Operating System and Realtime**: DSPs tend to have very slick or no operating systems. This property enables the ability to develop real-time applications with hard timing constraint much easer compared to a general purpose CPU.

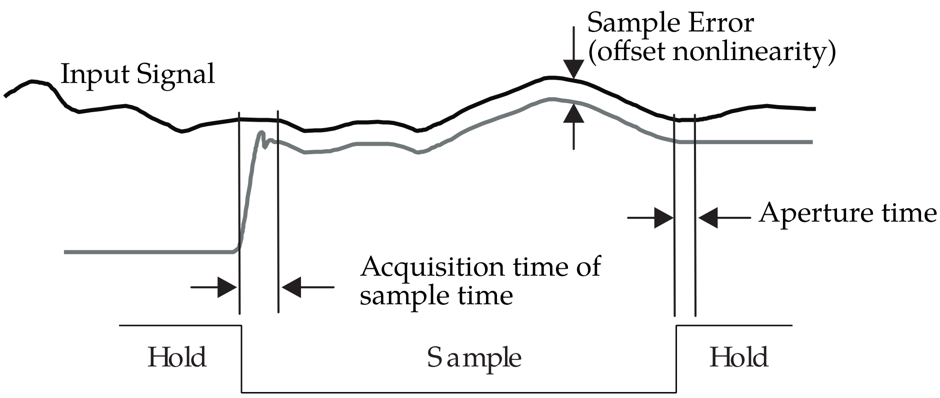


Figure 2.17.: Aperture Time in an ADC [3]. The ADC or better its built in S/H followes the signal with a cirtain offset. If it is then triggered it takes the *aperture time* until the S/H mechanism is decoupled from the external signal.

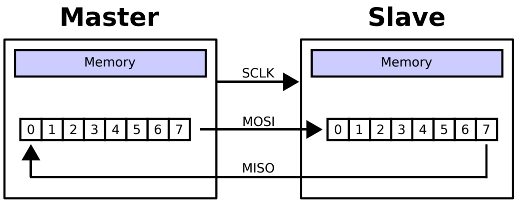


Figure 2.18.: An 8-bit Serial Peripheral Interface Bus transfer using two shift-registers: one in the master and one in the slave. After 8 single-bit transfers the master and slave will have transfered each other's register value [8].

Field Programmable Gate Array

FPGAs are integrated circuits which can be programmed after production, hence "field programmable". They consist of a huge amount of logical cells, RAM blocks and provide a hierarchal structure to logically "wire" them together.

Blocks consists out of lookup tables, which can be configured with logic tables and of course simple operations like AND, OR or NOT. Some of them have as well small memory capabilities so they could be used as registers or even register banks.

Modern Xilinx FPGAs provide blocks especially for arithmetic operations

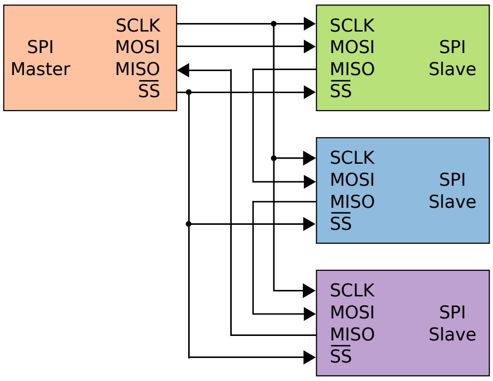


Figure 2.19.: Daisy Chaining of three SPI Slaves [5]

which are often related to digital signal processing. They provide a very fast structure for multiplications, additions and bit-shifting at the same time [17].

- **Size and Energy**: with the very generic approach of a FPGA a lat of complex tasks are solvable, if the FPGA has the appropriate size.
- **Realtime**: a typical programming style for FPGAs is to use so called Finate State Machines. That makes them predictable and deterministic in behaviour and time.
- **ADC Links**: due the possible high parallelism in the FPGA it will be no problem to feed 16 SPI channels in parallel into it.

Hardware Description Languages

To make the FPGA do something it needs to be "programmed". In terms of FPGAs that process is not called programming, it is called designing, this comes from the close relation to actual hardware design. The design the is **synthesized** to a FPGA configuration (Compiling is is the analog process in software terms). It is possible to synthesize a FPGA design to a real digital IC as well (Application-specific integrated circuit). Often FPGAs are used to do prototypes of ASICs. The design of hardware is done in a hardware description language as described below.

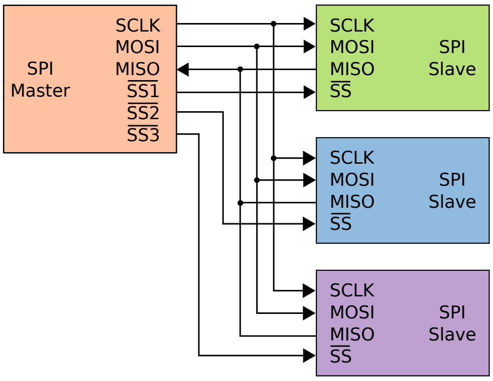


Figure 2.20.: Multiple SPI Slaves Selected by the Slave-Select(SS) Line [6]

Most commonly known hardware description languages are Very High Speed Integrated Circuit Hardware Description Language (VHDL) and Verlilog. They do have some differences when it comes to syntax, but the basic feature set of is similar [29]. They offer high abstraction levels. Not all abstraction layers are consistency supported from all synthesizers. This makes them inconvenient for testing with higher abstraction levels then Finite State Machines and Register Transfer Level.

A solution for that to use a higher abstracted language such as HLS (High Level Synthesis). Those languages allows the user to program in a imperative (e.g. C) style with no or minimal adoptions to run it the hardware itself. That implies then a lost of control as well.

One of the usual design setups for digital signal processing chains on FPGAs are done with Matlab and an emulator. A golden reference model is designed in Matlab. Then the design on the FPGA itself is done. In the end designers can see if the design in the emulator behaves like the golden reference model in Matlab. That implies a lot of "glue" code to export and import data from the simulator or Matlab.

MyHDL brings an abstraction to the hardware similar to Verilog or VHDL. But at the same time it is based on Python [12], which is a modern and robust programming environment. In combination with Scientific Python [9], Python offers more or less the same abilities as Matlab itself. So testing can be much more automated and linked, since the reference model and the hardware simulation are in the same environment.

Hardware/Software Partitioning

The engineering effort to implement even small things could be tough on FPGAs. The debugging and development effort can get enormous.

Different approaches ease development of such systems like high level synthesis (See Section 2.6) are growing. They reduce the effort of engineering such Systems, but still with more effort then just coding C. Space on the FPGA and timing constraints on a Register Transfer Level still needs to be considered. As a result the code developed for the FPGA should be as small as possible to reduce the time consumed by engeneering it.

Outsourcing as many tasks as possible to a normal CPU is a good option to reduce engineering effort of such systems. On a normal CPU compiler chains and debuggers can be used. Code can be written faster with less risk of involving bugs. This process is called software hardware partitioning. The two forces here are performance on a FPGA versus the development time on a CPU.

Resulting from this insights so called soft-core CPUs are getting popular. Those implement a CPU which can be synthized on a FPGA. This melds together the two worlds of hardware and software development and makes it easy to a partitioning which is quite dynamic as functions might moved from the soft-core to the bare metal FPGA, if the performance requires it.

For the work done here in the thesis it was decided to use a FPGA for high performance tasks and an external one board computer with Linux for the interface work.

2.7. Bus Connections Between Embedded Systems and PC Workstations

The project required to record the data to a personal computer to post process and analyze it it afterwards. Here I will discuss some different approaches to communicate between external hardware and a personal computer.

There are several ways to connect external devices to modern PCs. That does make it really hard to decide what kind of interface to use. Options for regular, modern PCs are USB 2.0, PCIe and Ethernet. All those interfaces providing a lot of bandwidth:

- Ethernet: 100 Mbit or 1000 Mbit
- USB 2.0: 480 Mbit
- PCIe 1.0: 250 Mbit per lane (Up to 16 lanes are possible)

To keep the development effort on the PC side as low as possible, PCIe and USB are no options since those would require the development of a driver on the PC side (USB provides UART emulation, there are problems with data rates higher then 196 KBit/s). A driver requires knowledge of the operating system on kernel level. That implies at the same time that a driver has to be developed for each operating system.

Ethernet on the other hand is already supported on almost every modern operating system. Operating systems normally offer the socket API for that. So do Windows and Linux. If a micro controller without operating system is used, Ethernet might be much more complex to use, as all the protocol layers have to be implemented. These are: Ethernet; MAC; IP; TCP/UDP.

High Level Protocols

TCP or UDP deliverers a stream or datagram service respectively. A higher abstraction can be reached by structuring the data in a kind of function calls. In this case the client just holds a proxy of function which are implemented on the server side. This is called “Remote Procedure Call” or Remote Procedure Call (RPC). CORBA and SUN-RPC had the leading role in the beginning in newer ages with the web so called web services became modern. Those are based on the HTTP standard. One of the is XML-RPC [34]. XML-RPC is a lightweight protocol to exchange data between different operating systems and languages and therefore a great choice for heterogeneous systems [20].

2.8. Scientific Computer Software Platforms for Laboratories

Requirements for lab-software are typically the ability to provide communication with the lab equipment, a graphical user interface and an engine to do computations. For this project software was required to display the measured data and control the parameters just in time.

Matlab

Matlab is an environment and a programming language. It offers a verity of so called toolboxes. The "Instrument Control Toolbox" provides the functionality for TCP or UDP connections to lab equipment. It provides also functions to implement graphical user interfaces.

The downside of Matlab is its pricing, a license with MATLAB and the "Instrument Control Toolbox" costs €700 for universities as of today [26].

Scientific Python

Python was already mentioned in Section 2.6. With Scientific Python [9] it provides a library to do numerical operations. It is a common programming language and has libraries to connect to network sockets as well.

Scientific Python is a free open-source software. No charges apply if used for educational or commercial projects. The whole source code is open. In case of problems the source code can be read or debugged. If a specific customisation should be made it can be done by modifying the source code.

2.9. Single Board Computer

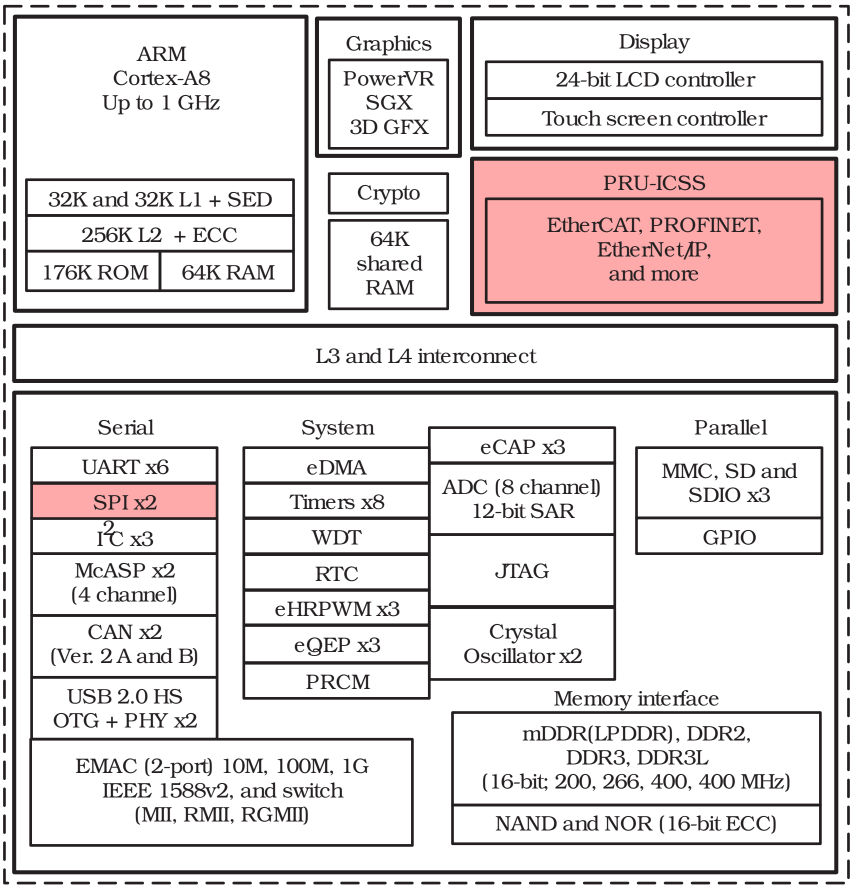


Figure 2.21.: TI Sitara System Architecture [18] with the elements marked which are particularly important for the thesis.

Released in April 23, 2013 the BeagleBone Black builds a good platform for reliable embedded systems. It is equipped with a system on chip solution from Texas Instruments, the AM3358 and offers a lot of on-board peripherals, and a pin row to build so called shields to extend it with peripherals.

The Ethernet of the BeagleBone Black is directly integrated in the SoC which is a great advantage over the Raspberry Pi.

Linux is the officially supported operating system for the BeagleBone Black. Several distributions are available for it.

The Texas Instruments AM335x series or Sitara processor is based on an ARM Cortex-A8 (architecture is on Figure 2.21). A remarkable feature of that series are the embedded Real Time Processing Units (short Programmable Realtime Unit (PRU)). These units are two microprocessors running at 200 Mhz, which can be programmed completely independent from the ARM core. That makes it possible to use them to implement time critical applications, since they do not necessarily have to be dependent on the main core. The PRUs have a small, simple instruction set, which is optimized for simple operations. The communication between the main core and PRUs can be done via the main core memory or the memory on the PRUs itself.

2.10. Algorithms Used in Project

Butterfly FFT

A Fourier transform in general converts a wave form from the time domain to the frequency domain. According to Joseph Fourier every waveform can be decomposed to its frequencies and amplitudes into its spectrum. The operation can also be reversed, called an inverse Fourier transform. This is a cruical mathematical transformation for that project.

Since we are working in the digital domain here we are using only discrete numbers leading us to the Discrete Fourier Transform (DFT).

$$F(n) = \sum_{k=0}^{N-1} x(k)^{-jk2\pi \frac{n}{N}}$$

where

$$n = [0, N)$$

With the complex coefficients (later called Twiddles as well):

$$W_N^{kn} = e^{-j\frac{2\pi}{N}kn}$$

it also can be written as:

$$F(n) = \sum_{k=0}^{N-1} x(k) W_N^{kn}$$

where

$$n = [0, N)$$

Computing all N points of a transform takes N^2 complex multiplications or in other words it has a runtime of $O(n^2)$. To reduce this the Fast Fourier Transform could be used. It has a runtime order of $O(n \log(n))$, which is much better [23]. The structure of the FFT is also optimal for hardware implementations. It reduces the amount of multiplications by simplification of the exponential terms and structuring the signal flow to buttery like structures.

Fast Trigonometric Calculations

To do all signal transformations required by the project an arctangent is needed as well. Here I will discuss two approaches of finding a solution quick in terms of computation.

With a static lookup table the memory requirements are exponential to the bit width of the input to the function. For example a sine of a 16-bit integer would take a lookup table of the size of $2^{16} = 65536$ entries. If the result should have the same resolution as the argument, the lookup table will have the size of $65536 \cdot 16 = 1048576bit$ or $128KiB$.

The COordinate Rotation DIgital Computer (CORDIC) [33] algorithm in the other hand uses a smart binary search like approach to calculate the trigonometric functions in just a few steps.

PI-Controller

To control the feedback of the device a PI-controller is used.

The PI-controller (PI means proportional, integral) is implemented to enable the possebillity of feedback to thelaser system. Lasers often provide the possebillity enable feedbach throu aanalog signal.

In continuous time a PI controller is defined as follows:

$$y(t) = K_p e(t) + K_i \int_0^t e(\tau) d\tau$$

rewritten to a discrete form it is:

$$y[n] = K_p e[n] + K_i \sum_{i=0}^{t} e[i]$$

Without a model it can be tuned with the Ziegler–Nichols [35] to reach a stable behavior.

3. Design and Implementation of CEP Shift Detection

The whole system can be seen from a functional perspective in Figure 3.1. Each block represents a function in the system. The *sensor* is responsible to acquire a signal from the light pattern given by the *laser experiment*. This acquisition outputs an analog signal which then needs to be digitalized by the *digitalisation* block. When the signal is in digital form the *digital signal processing* can take place and the steps explained in Section 2.1 can take place. In the *digital signal processing* step a PI-controller is implemented as well, which then is able to stabilize the CEP through the feedback which is outputted by the *analogize* step.

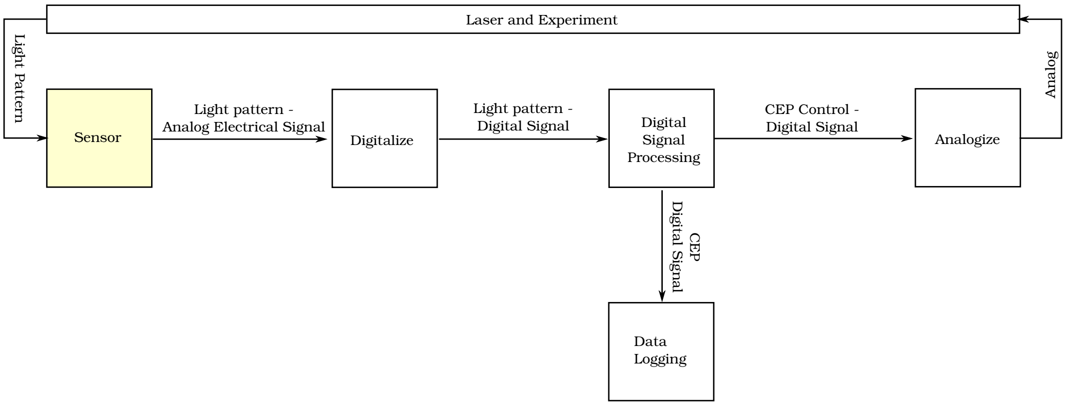


Figure 3.1.: Functional System Overview

The hardware modularisation mirrors the functional view of the system. In Figure 3.2 the similarity can be seen.

3.1. Modular Hardware Design

The PCB design was chosen to be as flexible as possible. This leads to group all components in a functional way similar to Figure 3.1. Each of those groups are exchangeable separately. In Figure 3.2 the modules and its connections are shown. Its seperated in the following hardware modules:

- The *sensor* integrated on the **amplification board**.
- The *digitalisation* is taking place on the **ADC** board.
- The **FPGA** which is responsible for the *signal processing* (FFT and so on).
- On the **IO-board** the *analogize* step takes place.

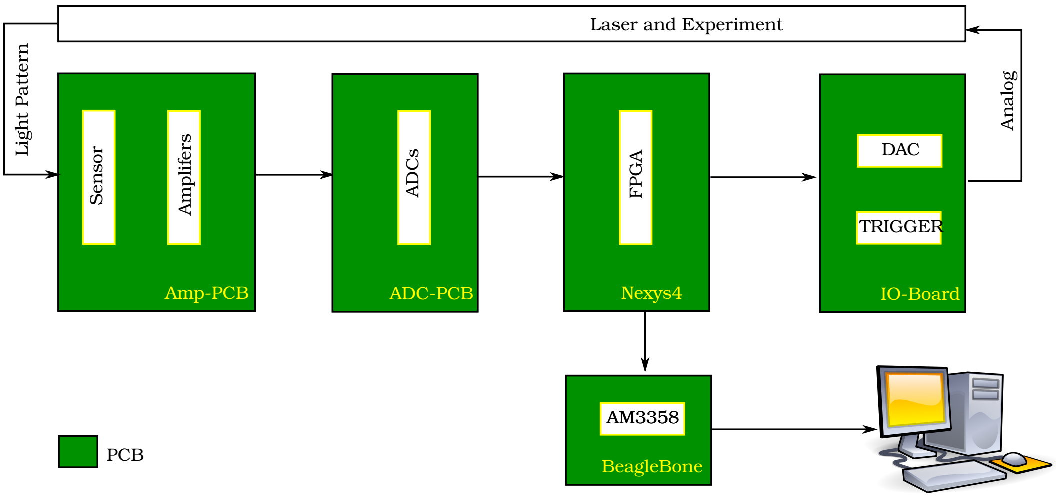


Figure 3.2.: The design of the hardware is strongly derived from the functions of the block as seen in Figure 3.1.

3.2. CEP-Beat Sensor and Amplification

The sensor block from Figure 3.1 consists on a technical level out of a silicon based photo diode (Section 2.2) array and an amplifier circuit as described in Section 2.3. On a physical level the circuitry of the amplifier and the sensor are on the same PCB.

For fast measurements the measurement is done in the *photoconductive* mode with a operational amplifer as described in Section 2.3. To ensure the proper working of the design a simulation of the selected components was done. Then a test setup was build with a bread-board and tests where done in the laser laboratory with scattered light. Based on those tested parameters a complete circuit was developed to create a PCB based on it.

Simulation and Parametrisation

The simulation of the measurement was done with a simplified model for one channel in Texas Instruments Tina [14]. Figure 3.3 shows the model. The photo diode is modeled by the current source *IG1*, as operational amplifier a Texas Instruments TLE2074 [32] is used, as recommended in [24]. The input capacitance ff the ADC is simulated by *C5*. Resistor *R2* and capacitor *C4* are together the input low-pass filter for the ADC.

Figure 3.4 shows ringing from the amplification. This indicates that the operational amplifier is driven close to its limits. A higher amplification leads to ring over to the next pulse which starts then to resonate as seen on Figure 3.5 and makes it impossible to measure the amplitude afterwards. In the implementation of Thomas Fordell the same circuit was used, but a much higher amplification. This was possible, because the repetition rate of 1kHz was much lower. Here the ringing has to decline within 5μs (Laser repition rate of 200 Khz).

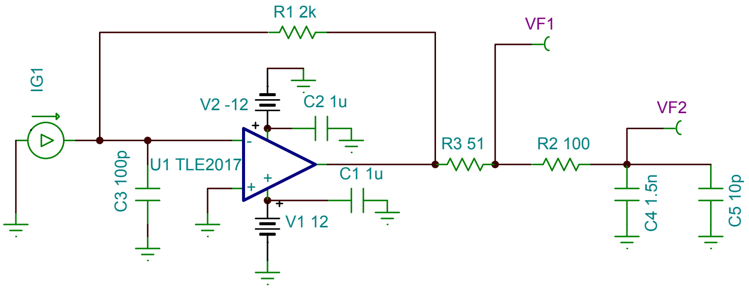


Figure 3.3.: Simulation model of the amplification circuit

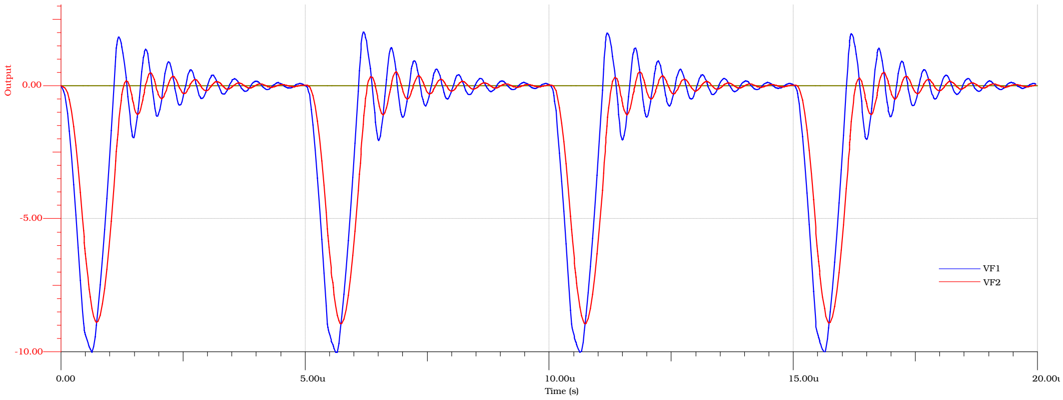


Figure 3.4.: Simulation of the amplification circuit with R1 beeing 2k (Figure 3.3). This leaves enough room to let the ringing converge to zero before the next pulse is detected.

3.2.1. Tests with the Laser

Before designing the PCB a test under lab conditions was made. In Figure 3.6 the test setup on a so called laser table can be seen. It uses the circuit from Figure 3.8 This installation has in general the effect, that the laser is visible in scattered form on the whole table. This effect was used as recommended by our supervisor to do a benchmark of the amplification circuit. The results of the measurement can be seen in Figure 3.7. The result was that the negative voltage is at $4.8V$ with a feedback of $2k$ as done in the simulation. The light impact was overestimated by factor two in the simulation, but this was still a solid starting point for the circuit development.

3.2.2. PCB Design

As described in Section 3.1 one of the design goals was to be modular. This concept is continued here: a Zero Insertion Force socket is used to mount the actual sensor on the PCB. This enables to change the sensor against another model or replace it in case of failure without any solder work.

The final circuit design can be seen in Figure 3.8. It is the same as used in the Tests.

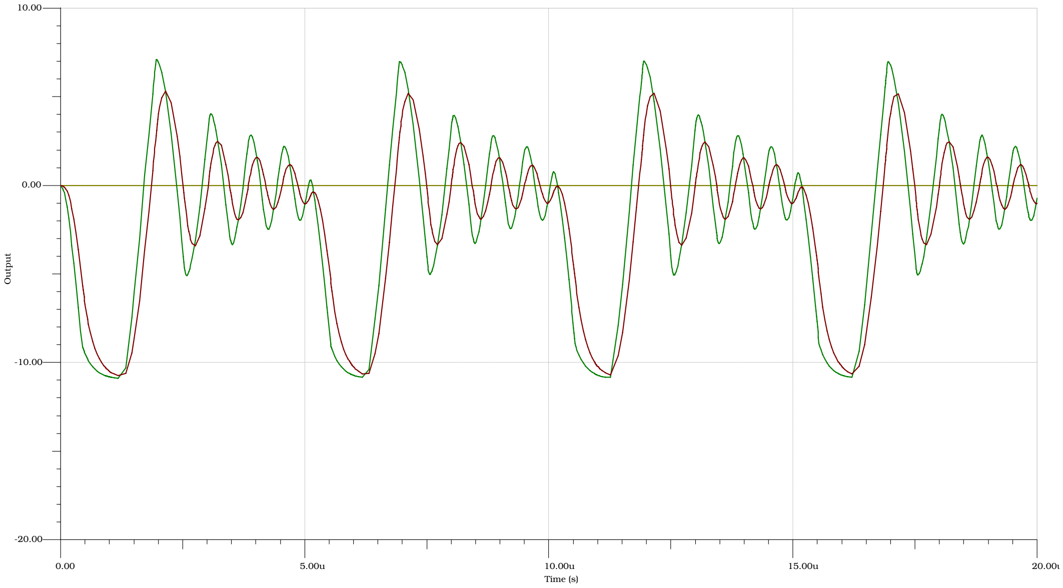


Figure 3.5.: Simulation of the amplification circuit with a feedback of 4k (Figure3.3. The amplification is to much, so it runs into the next laser pulse and makes the measurement of the maximal amplitude impossible.)

3.3. Trigger

After the amplification and before digitalizing the signal, the timing when the signal has to be digitalized is important. Regarding to this, this design is special: a series of short pulses gets digitalized with a low rate. While the repetition rate is in the range of 200kHz. The length of one pulse is in the range of μs, which is the equivalent range of Mhz speaking in frequencies. This fact is used to implement a trigger which measures the signal at the exact right time.

In Figure 3.10 the internals of the trigger can be seen. The trigger-input can external from a 5V signal. This input is secured by a Schottky-diode and then converted to a 3.3V signal by a level converter. The internal trigger is connected to a comparator, which then is connected to the output of a amplified signal from the 35th element of the sensor array. The comparison voltage is then generated by a DAC connected to the FPGA.

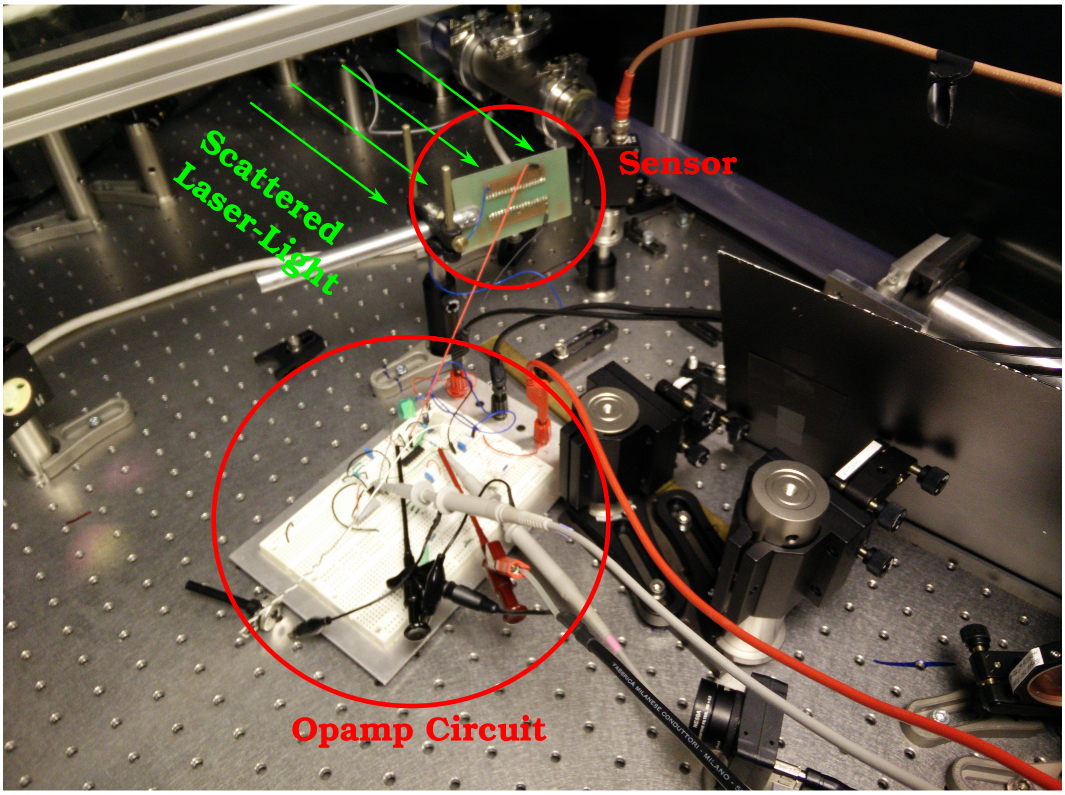


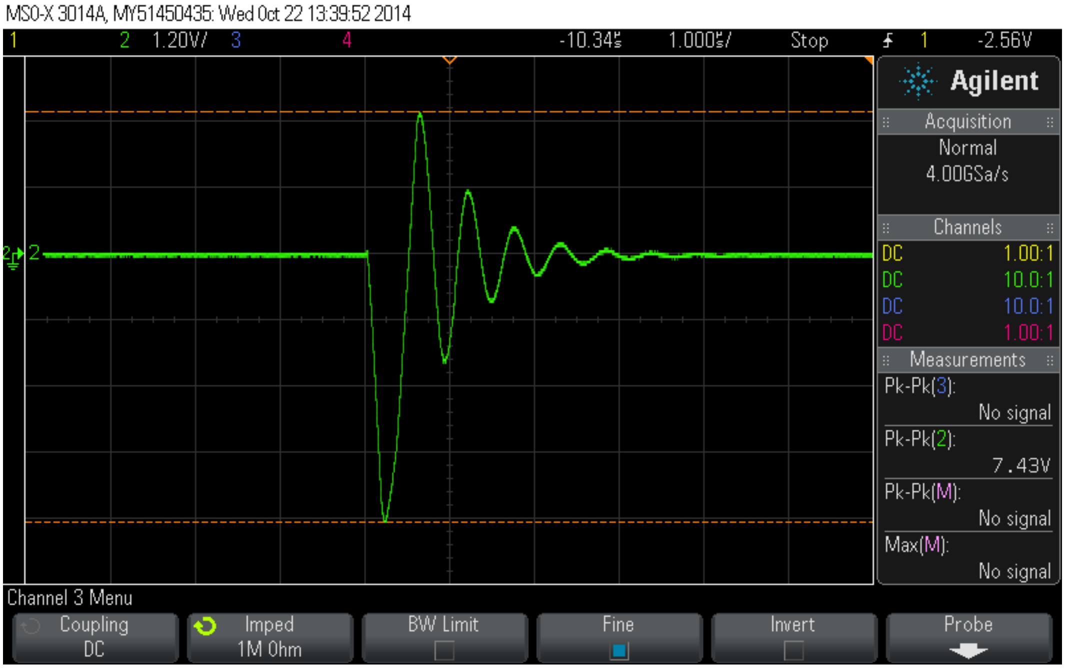
Figure 3.6.: Testsetup

To change the delay between the triggered trigger and the digitalisation the trigger signal is looped through the FPGA. The FPGA can then delay triggering if needed. An option to use route the trigger directly the ADC is given as well.

3.4. Digital Analog Conversion of CEP-Beat

The selection of the analog digital had important requirements:

- **Parallel capturing**: One of the reasons for the third iteration of the project was to capture all sensors parallel. That should have resulted in better measurement and more reliable results.



MSO-X 3014A, MY51450435: Wed Oct 22 13:39:52 2014

1 2 1.20V/ 3 4 -10.34s 1.000s/ Stop 1 -2.56V

Acquisition
Normal
4.00GSa/s

Channels
DC 1.00:1
DC 10.0:1
DC 10.0:1
DC 1.00:1

Measurements
Pk-Pk(3): No signal
Pk-Pk(2): 7.43V
Pk-Pk(M): No signal
Max(M): No signal

Channel 3 Menu
Coupling DC
Imped 1M Ohm

Figure 3.7.: Result of the measurement with a 2k resistor in the feedback. The negative voltage goes down to $-4.8V$.

- **Low Aperture Time**: The experiments from Section 3.2.1 showed us that this should be in the range of sub $0.1\mu S$ (the peak in Figure 3.7 is to be captured).
- **Sampling Rate**: A sampling rate of at least 200 $k\frac{samples}{s}$ was given by the repetition rate of the laser. Targeted was a sample-rate of 300 $k\frac{samples}{s}$ to have enough room.

The market for multi-channel-same-time-capture ADCs is pretty narrow, the choice fell on to the 8 channel 12 bit ADC Texas Instruments ADS8528 [1]. It can sample 480 $k\frac{samples}{second}$ in serial mode and has a aperture time of $5ns$. With the use of four of these ADCs, they where able to cover the requirement for 32 analog channels. It also supports a higher sample rate via a parallel interface, but that was neglected since it takes 8 digital channels for every ADC, which leads to a much higher pin usage in the interface and the higher data rate was not needed.

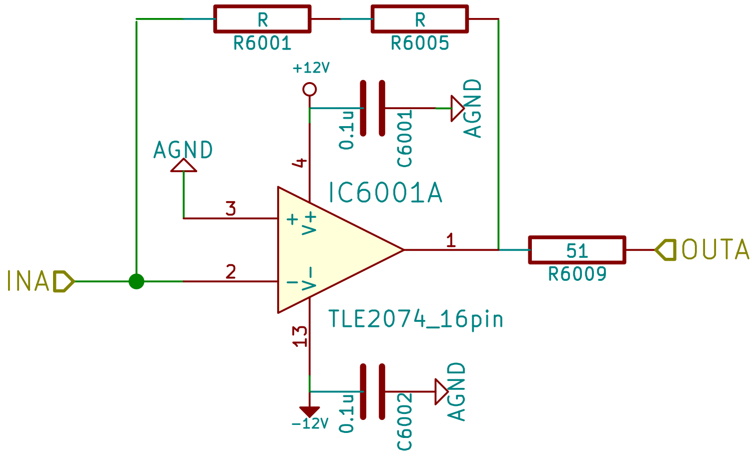


Figure 3.8.: Circuit for a amplifier of one photo diode

ADC-FPGA

An SPI-interface transports the digital data from the Analog Digital Converter to the FPGA. The selected ADC support a multichannel mode, which can be used to increase the data rate. The interface is specified to run at a maximum of $45Mhz$ which is equivalent to a data-rate of $45M\frac{bit}{s}$. Used with all four available available channels this leads to a maximum data rate of

$$45M\frac{bit}{s} \cdot 4 = 180\frac{Mbit}{s}$$

The required bandwidth is derived as the follows

- The count of ADCs, which is four
- The count of channels per ADC, which is eight.
- The laser repetition or sample rate which is $200kHz$
- The count of SPI transmission lines.
- The transmitted word-size is 16-Bit, also for the 12 Bit variant of the ADC.

$$16Bit \cdot 32Channels = 512\frac{Bit}{Sample}$$

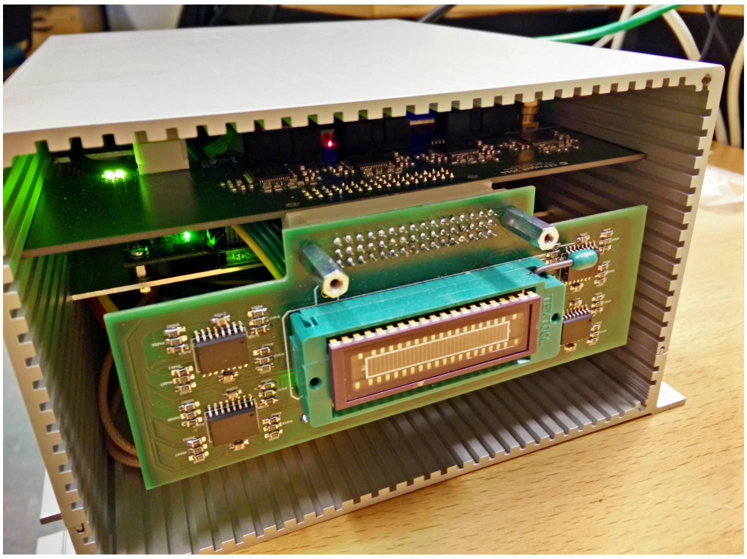


Figure 3.9.: PCB layout amplifier. The image shows the housing without the front faceplate. The amplification board is mounted parallel to front faceplate in the housing. The sensor mounted in the Zero Insertion Force Socket can be seen.

$$512 \frac{Bit}{Sample} \cdot 200k \frac{Samples}{s} = 102.4\ M \frac{Bit}{s}$$

The minimal bandwidth is $102.4\ M\frac{Bit}{s}$ if permanent data transmission is possible. To keep the possibility open the give a feedback to the laser, before the next pulse is shot the transmission time for one measurement should be as low as possible.

ADC PCB Design

In the PCB design for the ADC it was a major goal to keep close to the recommendations of Texas Instruments [1]. Their focus is to separate the digital and analog part as much as possible. This is done here with decoupling caps and two separate power-rails for the digital and analog part.

Additionally, each ADC has its own linear voltage regulator for the analog side of the ADC. Several filter and buffer capacitors are at the power lines

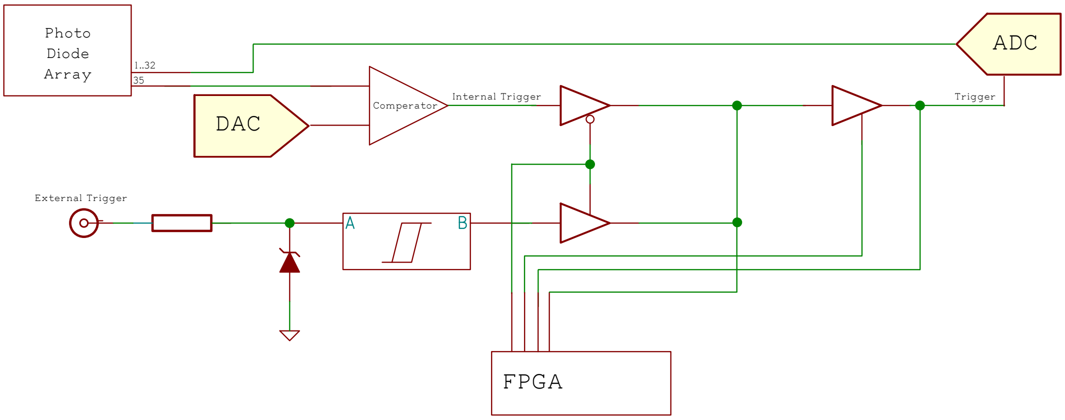


Figure 3.10.: The trigger internals

to prevent the modulation of the power line due to ADC-conversions.

3.5. Signal Processing in the FPGA

Signal processing in the FPGA is organized in different blocks (Figure 3.12) which are synchronized by handshake signals between them. The handshake signals indicate if the data from the previous block is ready to be transfered to the new block.

The main chain of blocks operate from the input to the output. These are the following:

- **ADC-SPI Driver**
- **CORDIC-Unit**
- **PI-Controller**
- **DAC-ADC**

The *ADC-SPI Driver* receives the data from the ADCs. In this case there are 4 ADCs with 4 SPI lanes each. Every ADC has 8 channels and transfers 16 bit per channel. That means it takes 32 clock cycles to transfer all channels from the ADCs to the FPGA. There is as well a trigger line which start the transfer process on the FPGA. The trigger line is connected to the ADC. After everything was received it sets a ready bit on high.

The FFT is implemented as butterfly with different states where the important are **DATA_IN**, **CALC** and **DATA_OUT**.

- **DATA_IN**: This read the data from the parallel input and stores it again in its internal buffer in the right order for the butterfly.
- **CALC**: Here the actual calculation takes place. This is done in $log_2(bits) = 5$ clocks.
- **DATA_OUT**: In this state it is written to output of the FFT block. Here the data is in the right order.

The CORDIC blocks takes the real and imaginary part of the FFT when it receives the ready flag and calculates the phase angle out of it.

3.6. CEP Control Output

Control is done with a PI controller in the FPGA(Section 2.10). It additionally has a post scaler at the output which can bit-shift the output value before it is handed to the DAC. This should enable the adaption of the output signal to different lasers. As all the other blocks it sets a signal to the "high" state when it is done with its work. This value is then received by the DAC SPI block which transfers the data to the DAC. The DAC then generates an analog signal for the laser feedback.

For digital analog conversion the Texas Instruments DAC7731 [2] was selected. It is a digital analog converter with 16 bit resolution and a output range of $[-10, 10]V$. It has a settle time of up to $5\mu s$ for a full change from the negative to the positive value. That might seem much, but in general changes are expected to be much smaller in range of several $100mV$, where the settle time is much lower as well. It is configured to the voltage range of $[-10, 10]V$ with the standard circuitry from the data sheet [2].

Feedback is done with the same model as setting the reference voltage of the internal trigger. They are connected and configured via a SPI daisy chain.

3.7. Timing behavior

With all the components described it is a good point to describe the timing behavior of the whole system.

The trigger and aperture process takes place in the sub $10nS$ domain and is therefore neglected here.

According to the data sheet it takes $1.33\mu s$ for the ADC to convert the analog value into a digital one. Afterwards the transmission takes place. The time it takes to transmit two data words is proportional to the count of the used lanes and the clocking frequency of the SPI. In this case the FPGA is the FPGA master and provides a SPI clock frequency of $25Mhz$.

$$t_{ADCSPI} = \frac{1}{25 \cdot 10^6} \cdot 32bit = 1.28\mu s \tag{3.1}$$

$$t_{ADC} = t_{ADCSPI} + t_{ADCconv} = 1.33 + 1.28 = 2.61\mu s \tag{3.2}$$

The total time needed for the ADC is then $2.61\mu S$. This leaves $2.39\mu s$ for the rest of the processing pipeline.

The FPGA runs at a clocking frequency of 100 Mhz. It could run faster but that also involves a more complex design to fit all the timing constraints. In total all blocks together take a time of $410\ nS$ (Table 3.1 for more details).

$$t_{ADC} = t_{ADCSPI} + t_{ADCconv} + t_{FPGA} = 1.33 + 1.28 + 0.41 = 3.02\mu s \tag{3.3}$$

The FPGA needs in the daisy chain configuration $1.2\mu S$ to transfer to the DACs.

$$\begin{aligned} t_{ADC} = t_{ADCSPI} + t_{ADCconv} + t_{FPGA} + t_{DACSPI} = \\ 1.33 + 1.28 + 0.41 + 1.8\mu s = 4.82\mu s \end{aligned} \tag{3.4}$$

Witch leaves the DAC $0.18\mu S$ to bring the analog value to the output. This will only work for small changes, greater changes won't fit into the $5\mu s$.

Table 3.1.: Timing of the single blocks in the FPGA

Block	Cycles	Time
ADC-SPI	2	20 nS
FFT	9	90 nS
CORDIC	21	210 nS
PID	6	60 nS
DAC-SPI	2	20 nS
Total	41	410 nS

According to the datasheet the a full bipolar change of 20V can be done in $2\mu s$. Linearly interpolated a change 1.8V can be done in $0.18\mu s$. Here more testing would be need to see what actually can be done and where the real limits are. This was not possible due the lack of time.

3.8. Communication to the PC

Signal processing is done in a FPGA. The FPGA can manage parallel tasks well (Section 2.6). This is used for the communication between the FPGA and the ADC. Further tasks of the FPGA have been as well the communication with a ordinary personal computer, where the communication should take place with Ethernet and TCP (Section 2.7).

For the first Ethernet implementation on the FPGA a soft-core from Xilinx, Microbalze was used. In the field tests this was not a stable solution, therefor a BeagleBone was used to replace the soft-core. With a working Ethernet and a working operating system it was the stable solution.

With the maximum repetition rate of $200KHz$ of the laser the following calculation is used for the bandwidth:

$$32Bit \cdot 200Khz = 6400k\frac{Bit}{s} = 6.4M\frac{Bit}{s}$$

To transfer data between the FPGA and the BeagleBone, SPI has been used since it already was implemented on the FPGA. Two interfaces are used to communicate between the FPGA and the BeagleBone (Figure 3.13). One which requires real-time data transmission, the other which is to parametrize the FPGA and has no special requirement about real time processing. Each of them is uni-directional. Real-time data transmission is done via the help of the PRU, as explained later. Parametrization is done via with the regular kernel driver called SPIDEV [25].

Unfortunately the BeagleBone hardware SPI implementation under Linux only the master mode. For the real time transmissions it is preferred to use the FPGA as master, since it can trigger transmission after each laser-shot and its followed calculation on the signal processing part of the FPGA, while using the BeagleBone as master requires polling from the BeagleBone side.

Transmit data with a certain required bandwidth is well known to be a difficult problem. Exactly this is required to get the data reliable from the FPGA to the BeagleBone. The BeagleBone especially offers the Programmable Real Time Unit for realtime tasks like that. It provides a memory interface to the main CPU. The transmission works with a flip-buffer in the PRU:

1. The FPGA finishes the phase calulculation
2. The FPGA writes the result into the SPI transmission register and triggers the transmission.
3. The PRU receives the dataword and saves it into its interal RAM
4. 1.-.3 is repeated until half of the memory is full.
5. A interrupt is triggered, which starts a routine on the ARM CPU which then reads the half of the internal RAM of the PRU
6. The PRU then can use the just read out RAM again.

To detect transmission errors and missed datasets a 16 bit counter is sent with each dataset from the FPGA to the BeagleBone.

Setting parameters in the FPGA requires no realtime requirements given, because it is done before the actual capturing process. As counterpart to the realtime SPI here the FPGA acts as slave and the Linux SPIDEV [25] drivers are used. This simplifies the programming on the Linux side since it is just writing to a file.

Protocols on the Network Layer

The communication channel between the PC Workstation and the BeageBone then is done via UDP/IP. Similar to the two SPI channels, here are two channels required as well. One is to transfer the data which should be logged to the PC, while the other one is used to configure and parametrize the "program" running on the FPGA. To enable a live view a third connection is used, which sends the raw sensor data, the FFT and as well the phase via TCP/IP to the PC.

The TCP/IP communication transfers $32 \cdot 2 \cdot 3 + 2$ bytes, that contain the raw sensor data, the real and the imaginary part of the FFT and as well the phase in 16 bin integer representation. This is used to display the according data in a plotted form to the users (Section 3.9).

XML-RPC is used to parametrize all functions on the FPGA as well to start the actual capture process or restart the capture program running on the BeagleBone. Available commands are:

- **set_config (key, value)**: To set a value in the FPGA
- **get_config (key)**: Receive the actual value
- **set_index (index)**: Sets the index of interest in the FFT
- **start (count)**: Start to record count datasets.

The set_config and get_config operations support the following keys:

- **trig_delay**: The delay of the trigger unit in $10ns$ steps.
- **atan_index**: The index in the FFT
- **pid_on**: To enable the PI controller
- **led**: The leds on the FPGA board for testing puproses
- **trig_sel**: Selection of the trigger source.
- **trig_ref**: The reference voltage of the internal trigger.
- **adc_range**: The range could be selected ether 5 Volts or 10 Volts
- **pid_kp**: The PI's K_p value.
- **atan_capture_cnt**: This value is set to start the capturing process. The number is how many shots should be recorded.
- **pid_ki**: The PI's K_i value.

The third channel is the UDP/IP transmission of the actual measurement values. They are transmitted just in the same way as the are received from the FPGA, without preprocessing them in the BeagleBone. This is a measure to put as less load as possible to the BeagleBone, because higher load increases the probability of lost data in the transmission.

3.9. PC Analysis Software

Two major goals were aimed while implementing the Graphical User Interface (GUI). The first was to ensure that the received data is written on the hard disk regardless of whatever else happens at the same time in the software. The other was to make it as easy as possible to use.

To achieve that, the software was separated into two components. One component is used to receive the data via the described protocol and will be called data recorder, whereas the other one is referred to as user interface. The user interface has the function to display the received information to the end user. Python [12] was chosen to implement the software because of the below-mentioned reasons:

- It is completely free to use for private and commercial projects.
- It is platform independent; it runs on Windows, Linux, Mac OSX as well as on real-time operating systems such as QNX.
- It offers scientific libraries, which can be used to calculate numerical problems like in Matlab [9].
- Flexibility: even though it is a high level language, it offers direct access to operating system functions such as multiprocessing [13] or the serial port [21].
- Several frameworks are available to create high sophisticated interfaces. For this project PyQT [27] in combination with PyQtGraph [4] was choosen.

The user interface has two windows: one to display the live data, and a second one to parametrize the FPGA parameters. The parameter window uses the in Section 3.8 decribed XML-RPC interface to communicate with the BeagleBone. Figure 3.14 shows the dialog running on Linux. The other window seen on Figure 3.15 represent the data which was measured in three separated lines:

- **Top**: Raw data directly from the sensor,
- **Second Line**: The FFT-transform with the imaginary and real part
- **The Last Line**: Past 120 seconds of the measured phase.

These datasets are received via TCP/IP in a frequency higher than 30Hz to look vivid. On the right, there is the additional record button with the field next to it, to enter the number of milliseconds to record. The data from this buttons is send as well by XML-RPC. To select the index for the FFT, the yellow line can be dragged and dropped to the wanted position.

The recording takes place in a separated forked process create by the python subprocess module with high process priority. It acts as a UDP/IP server and receives a whole dataset and writes it directly to a file. After this is done it does a post-checking of the integrity. A summary can be seen in Figure 3.13.

3.10. Housing

The space on a laser experimentation table is limited. It has screw holes to mount equipment on it. That laser table equipment has usually the size of centimeters or some decimeters. To not take place from running experiments we wanted to have the box as small as possible. At the same time it should be mountable as well as the other equipment.

A laser laboratory with a lot of equipment is inertly a hostile environment for electro-magnetic sensitive devices. A cooling pump to cool the laser components to low temperatures and several vacuum pumps are in the same room and cause inductive distortions. Therefore a aluminium case was chosen to ensure proper shielding of the used components.

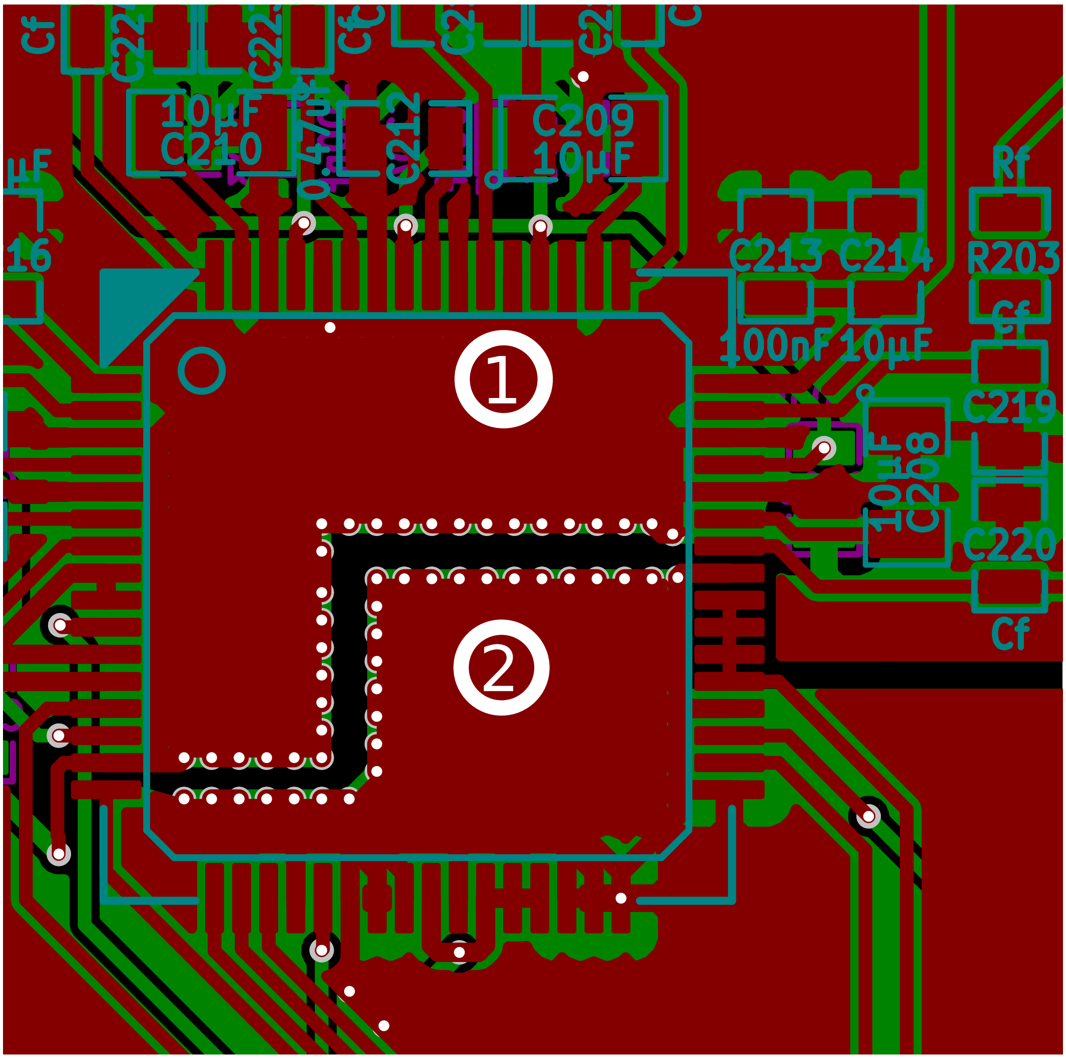


Figure 3.11.: PCB layout of the one ADC, design after the recommendations from Texas Instruments [1]. The analog side at ① is separated to the analog side ②. With through hole plating prevents modulations of the analog side with the signals from the digital side.

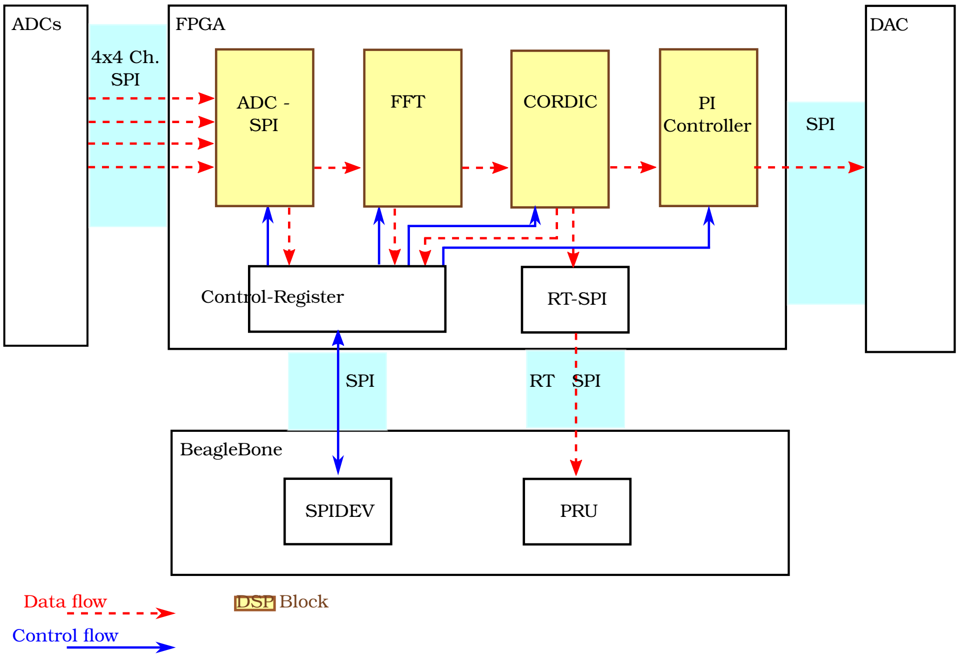


Figure 3.12.: The internal blocks of the signal processing in the FPGA

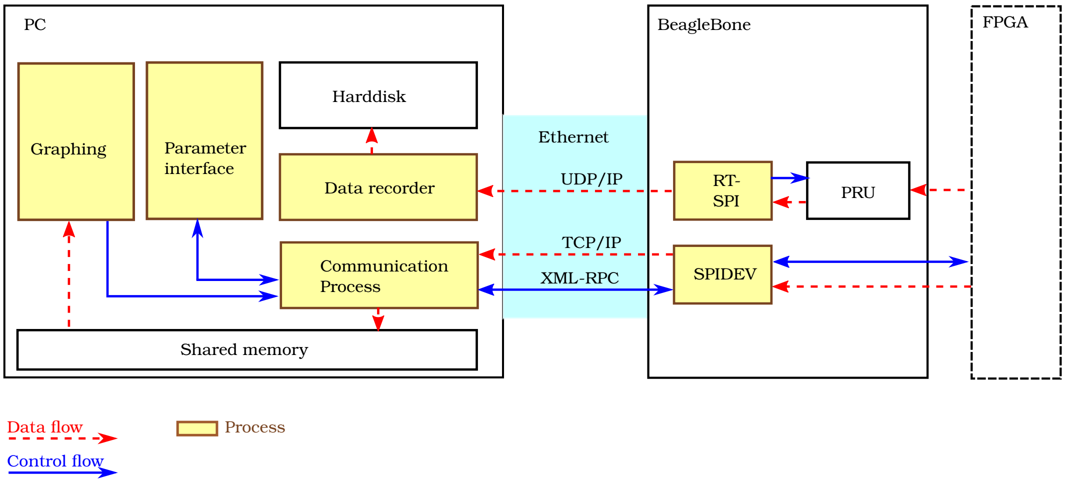
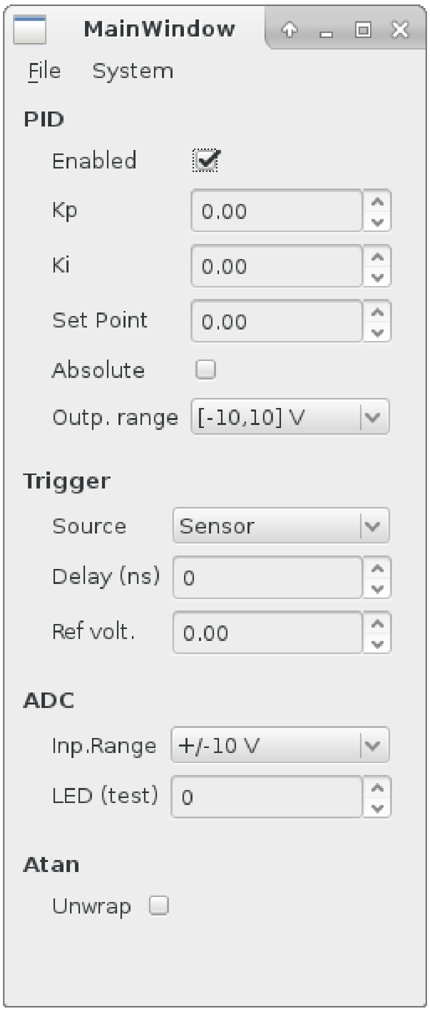


Figure 3.13.: The communication between FPGA and BeagleBone is done via two different interfaces



PID

Enabled ☑

Kp 0.00

Ki 0.00

Set Point 0.00

Absolute □

Outp. range [-10,10] V

Trigger

Source Sensor

Delay (ns) 0

Ref volt. 0.00

ADC

Inp.Range +/-10 V

LED (test) 0

Atan

Unwrap □

Figure 3.14.: The configuration part of the user interface running on Linux

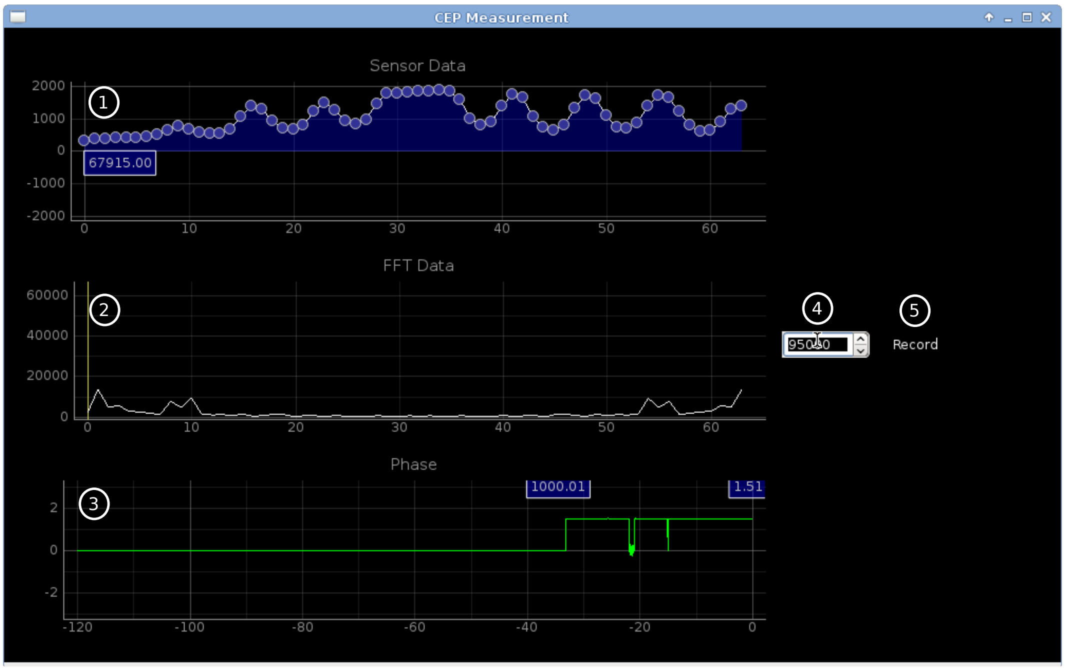


Figure 3.15.: The graphical user interface displaying live data running on Linux. ① Raw data directly from the sensor ② FFT transform of the raw data, computed by the FPGA ③ the phase change of the last 120 seconds ④ the number of samples which are recorded when pressing the record button ⑤

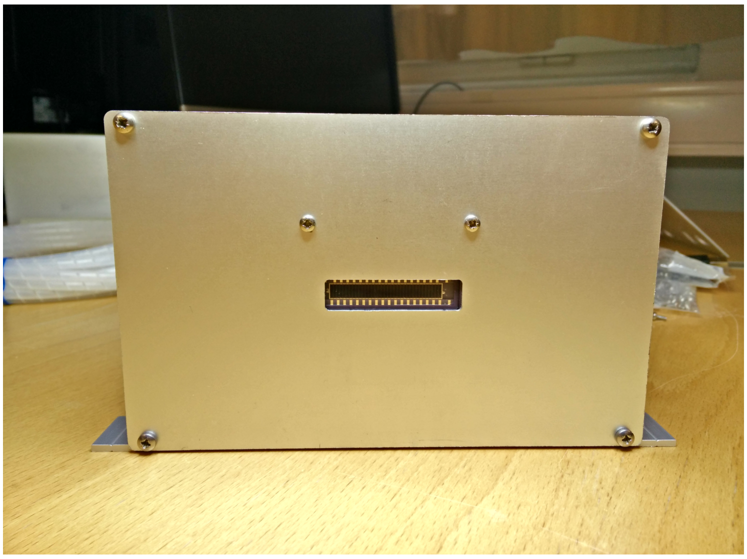


Figure 3.16.: Front of the Housing

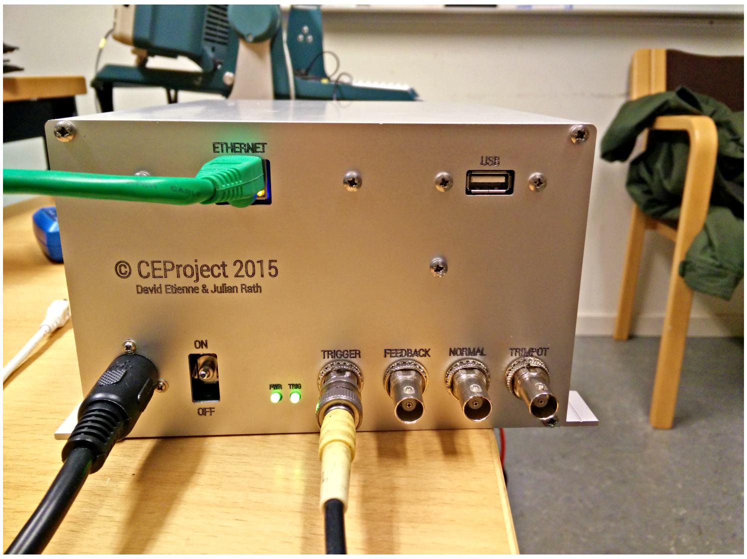


Figure 3.17.: Back of the Housing

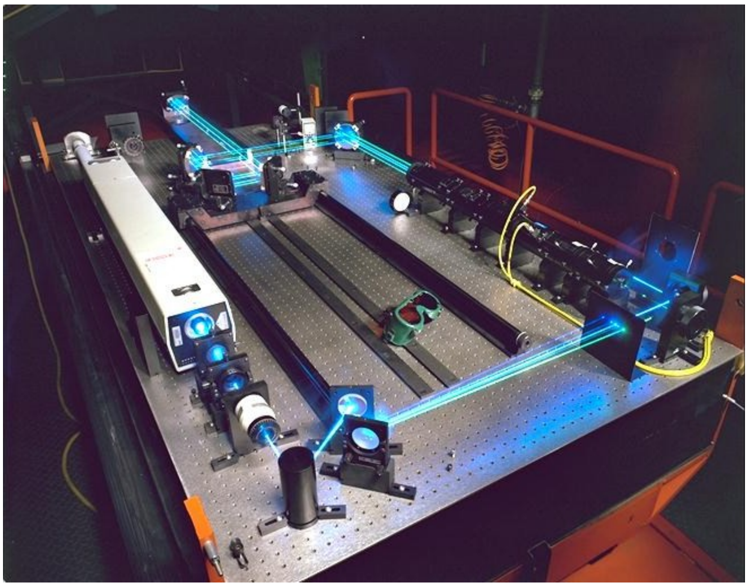


Figure 3.18.: Optical Laser Table(from Wikipedia)

4. Results and Outlook

The device offers all functionality described in the thesis. It is now the third iteration of this kind of device at the Lund University. A lot of the existing experience was used to design this device. A few small issues still remain and could be fixed within the current implementation due to its modularity.

4.1. Amplifier - Reference Measurement

The amplification module containing the sensor on it was not dimensioned for the f2f experiment, rather its reference measurements were taken in a scattered light environment on the laser table for a 1Khz laser. With the f-2f experiment the light impact on the sensor is too weak to get a decent noise free FFT. The 200 kHz laser in Lund was not ready as writing this, but in general its estimated per-pulse energy is much lower on a faster laser and therefor the FFT signal will be even worse in that scenario.

With the modular design it still poses no problem, since just the amplification module would need a redesign. For the measurement probably a two stage amplification is necessary since it need to be one or even two decades stronger then the current implementation.

4.2. Communication Problems TCP

While measurements where running in the laboratory a steady increase of the latency between the graphical user interface and the sensor in front of the light source was observed. That was pretty obvious when an obstacle was moved in front of the sensor. This might be a known problem with the windowing algorithm in TCP/IP and might be corrected by changing TCP/IP parameters or switching the protocol to UDP/IP.

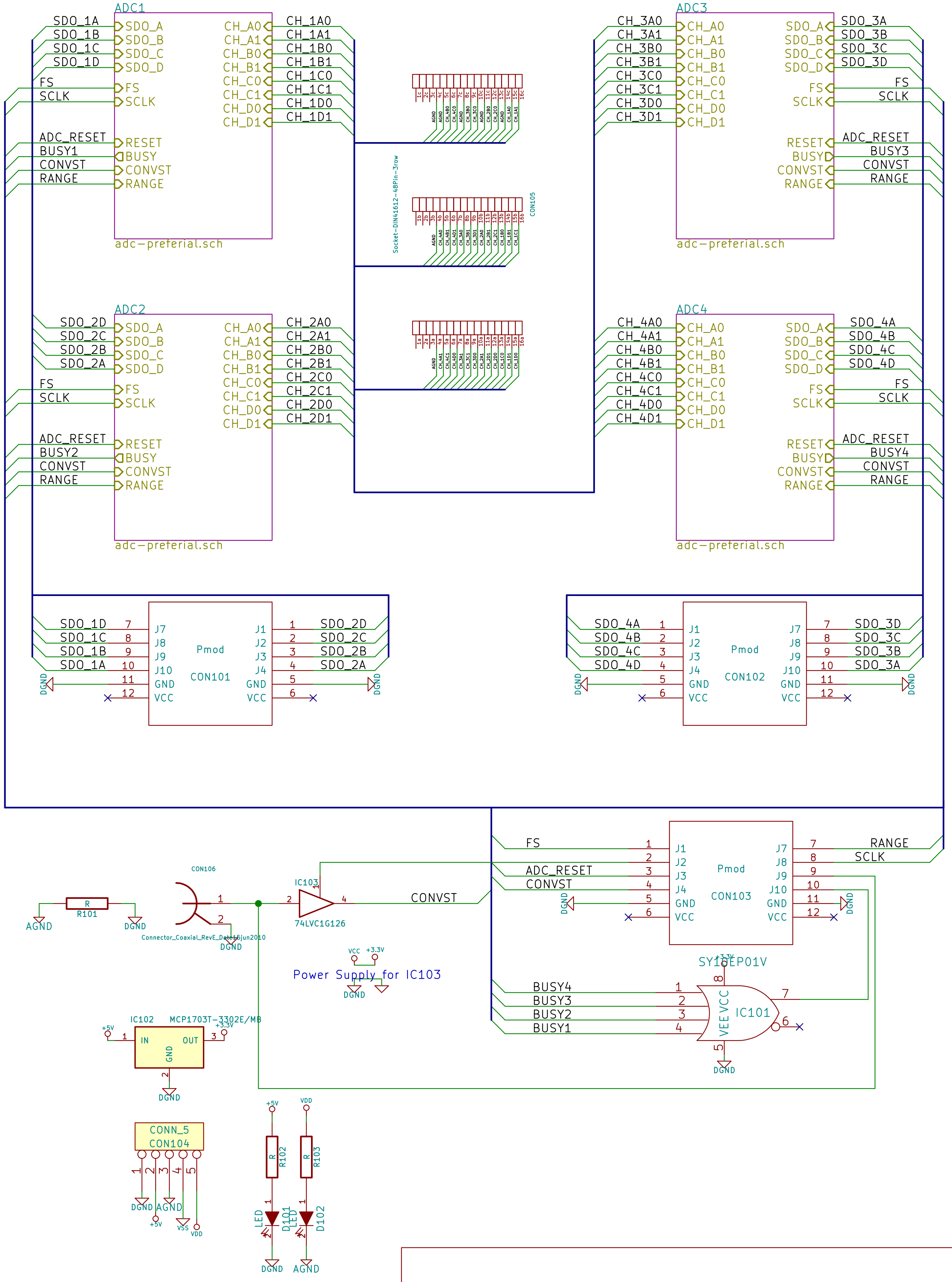
4.3. Evolving to a Product

Most of the steps while engineering have been done while keeping in mind that this should be the basis of a product. This can be seen in the design of the PCBs the housing and the selection of the components. More than 1000 hours have been invested to realize this device. At the end unclarity about the value of this work and its intellectual ownership, the developments into the direction have been stopped.

Appendix

Appendix A.

Schematics

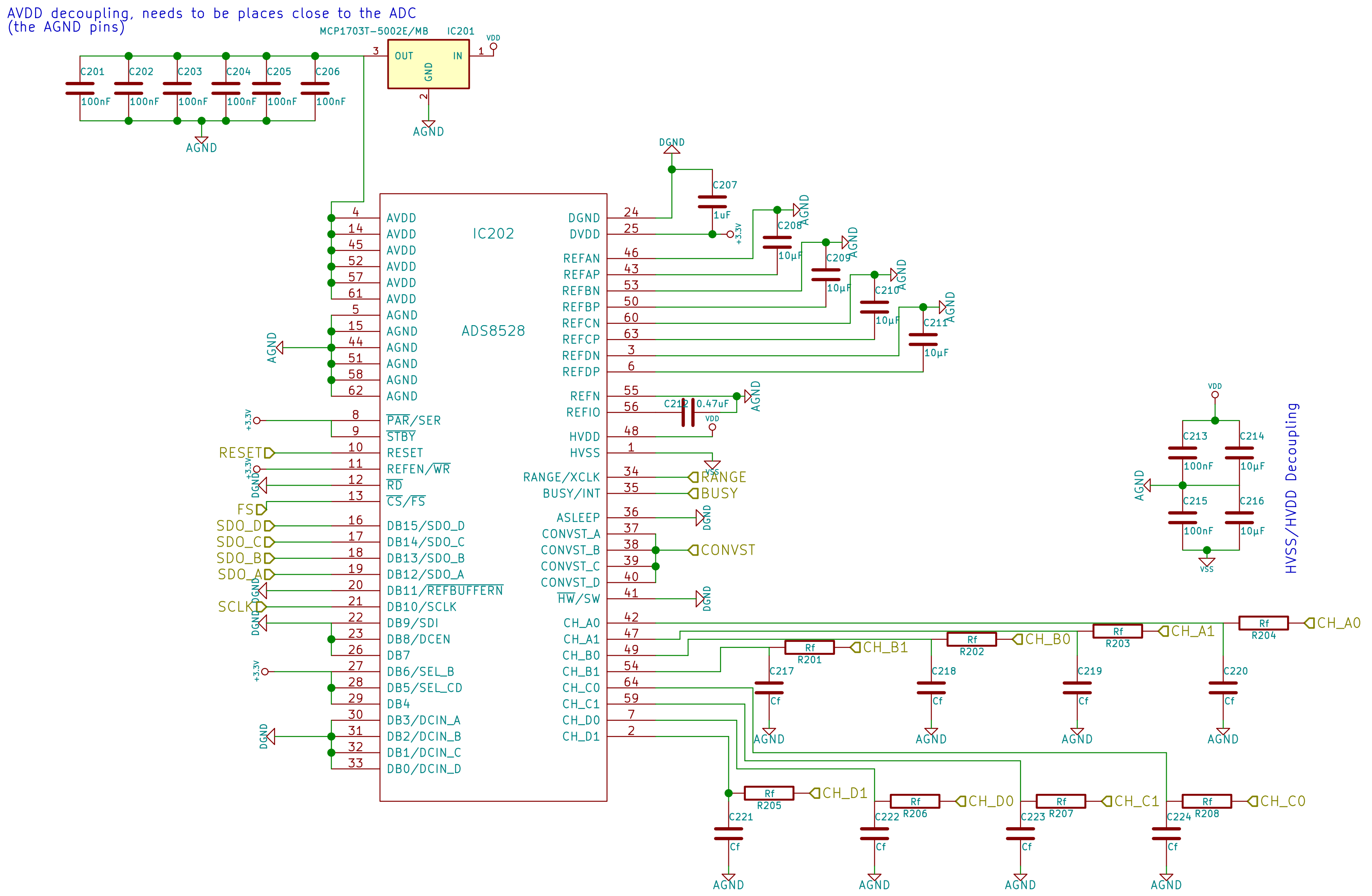


Sheet: /
File: adc.sch

Title: ADC Board

Size: A4 Date: Rev:

KiCad E.D.A. kicad (2015−05−10 BZR 5649)−product Id: 1/5

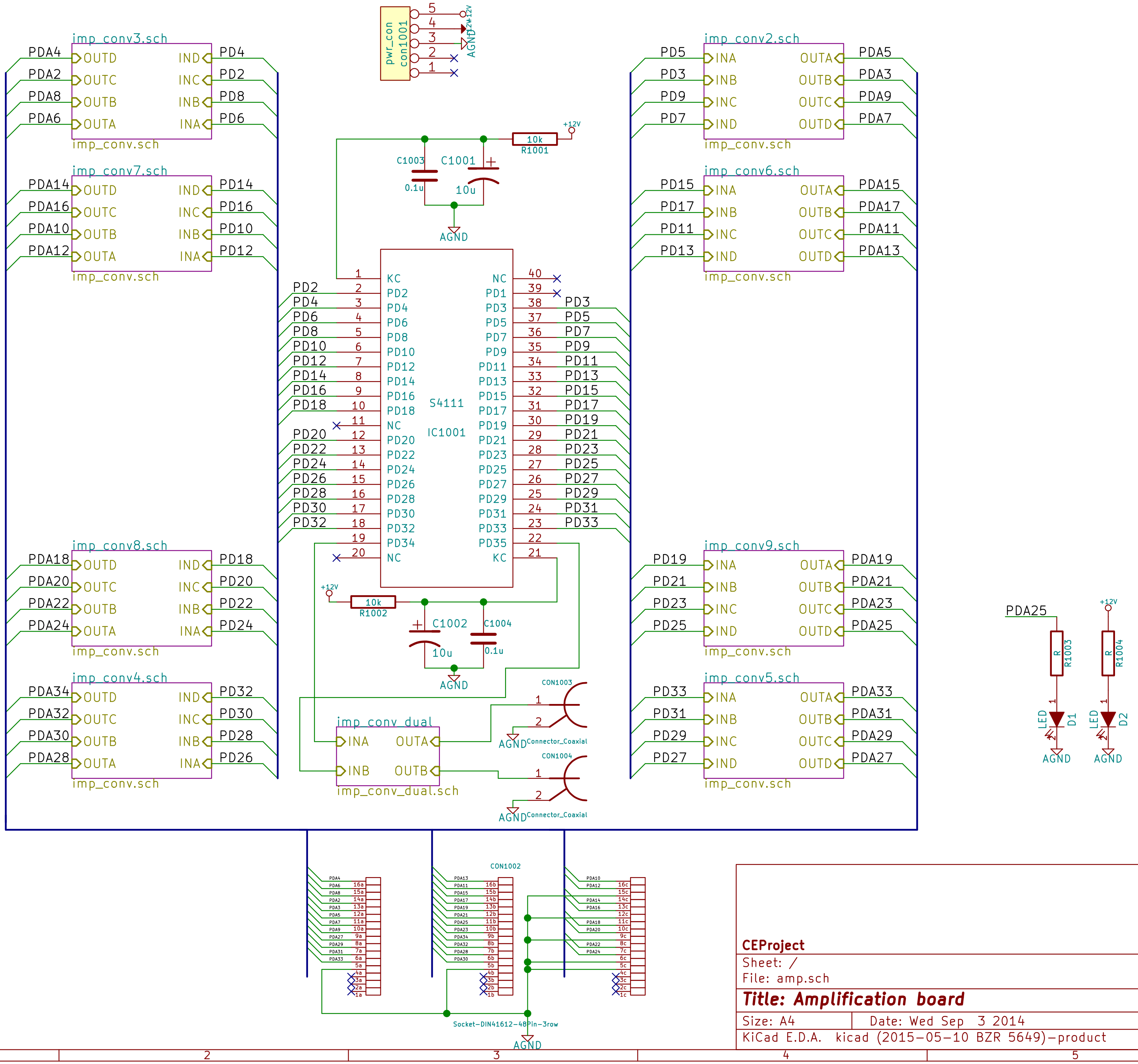


Sheet: /ADC1/
File: adc-preferial.sch

Title:

Size: A4 Date: Rev:

KiCad E.D.A. kicad (2015-05-10 BZR 5649)-product Id: 2/5



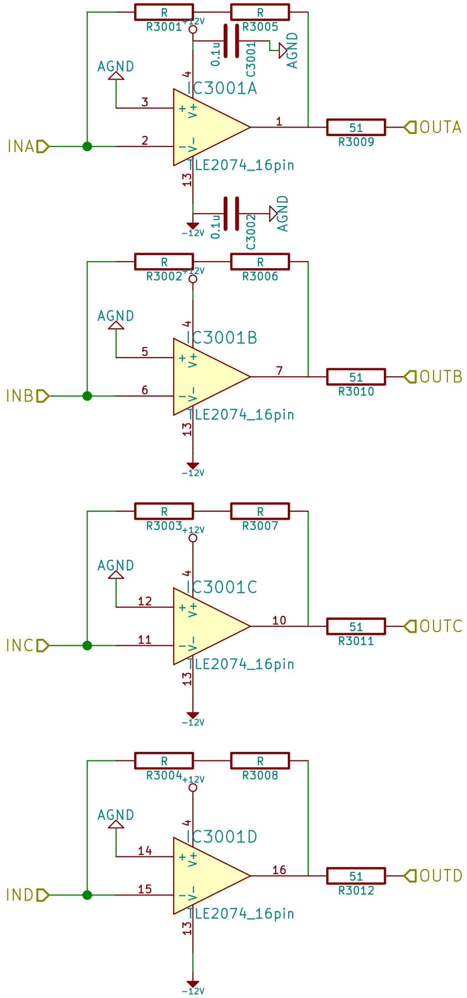
CEProject

Sheet: /

File: amp.sch

Title: Amplification board

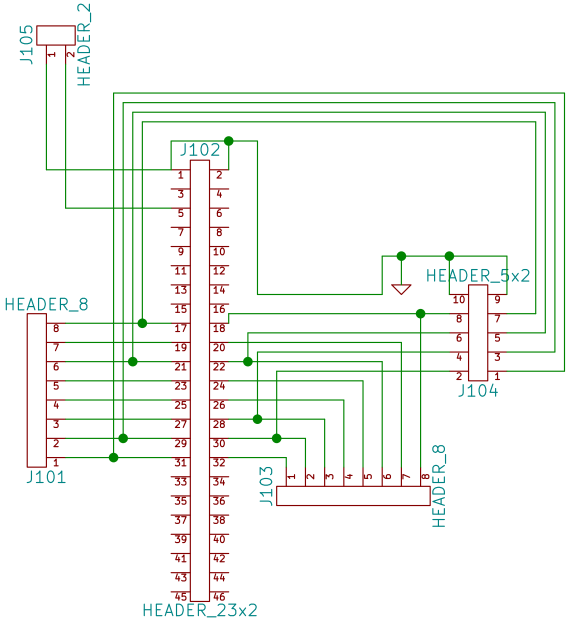
Size: A4	Date: Wed Sep 3 2014	Rev: v1
KiCad E.D.A. kicad (2015-05-10 BZR 5649)-product		Id: 1/10



Sheet: /imp_conv3.sch/
File: imp_conv.sch

Title:

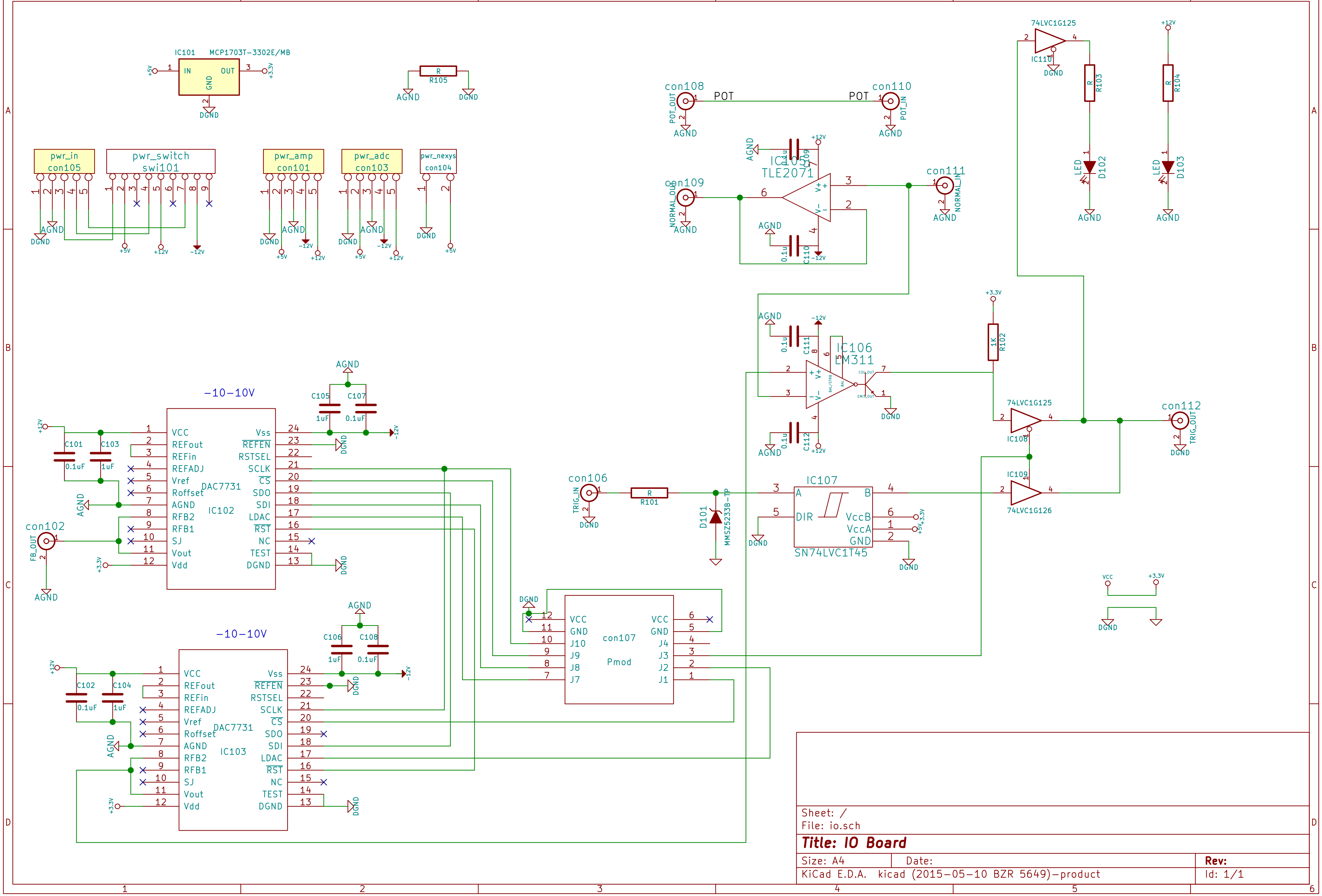
Size: A4	Date:	**Rev:**
KiCad E.D.A. kicad (2015−05−10 BZR 5649)−product		Id: 2/10



Sheet: /
File: bbb-pmod-adapter.sch

Title:

Size: A4	Date:	**Rev:**
KiCad E.D.A. kicad (2015-05-10 BZR 5649)-product		Id: 1/1



Sheet: /

File: io.sch

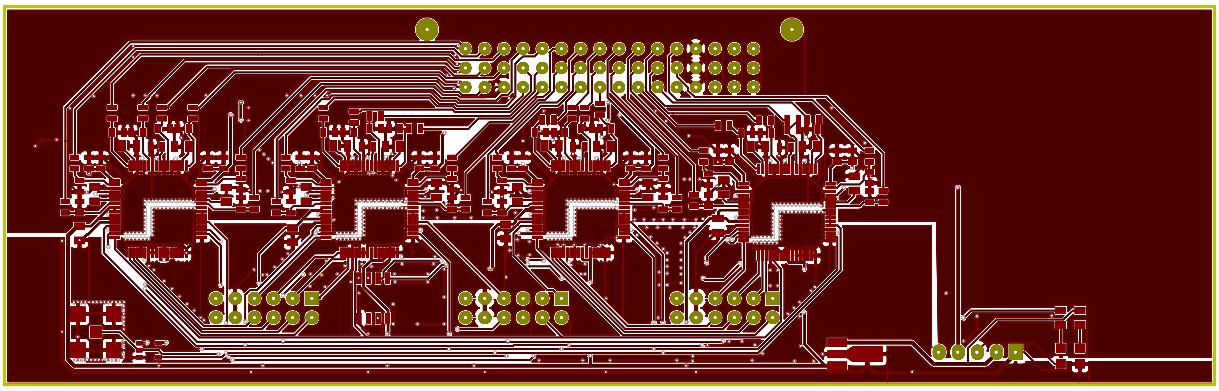
Title: IO Board

Size: A4 Date: Rev:

KiCad E.D.A. kicad (2015-05-10 BZR 5649)-product Id: 1/1

Appendix B.

PCB Layouts

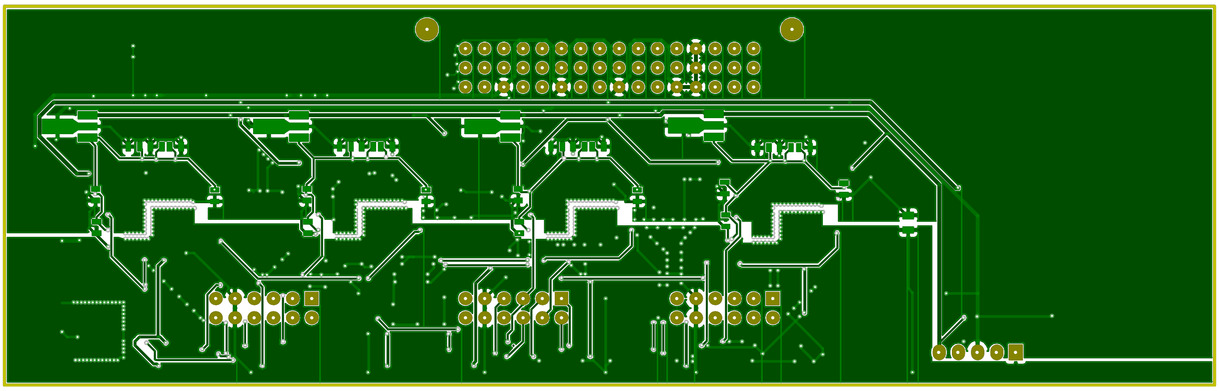


Sheet:

File: adc.kicad_pcb

Title: ADC CEP Project

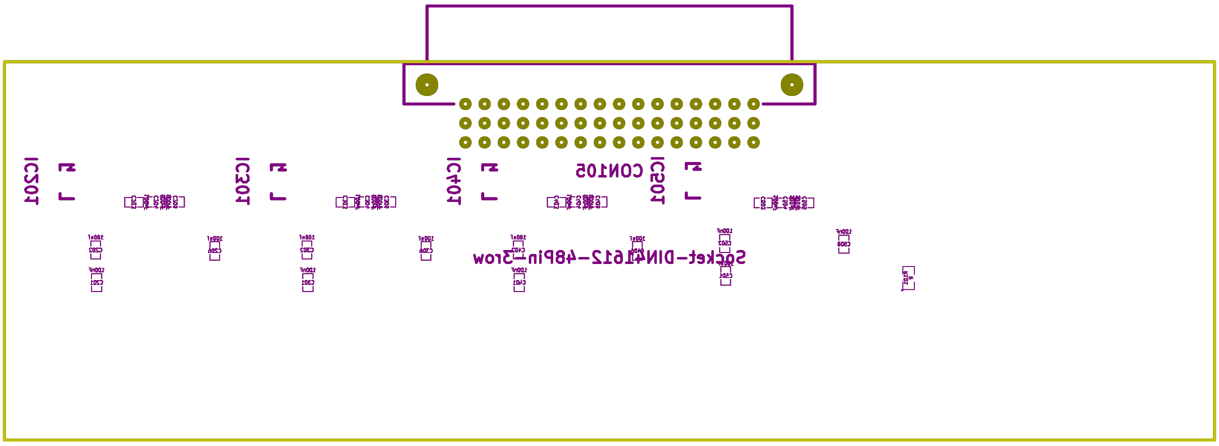
Size: A4	Date:	Rev:
KiCad E.D.A.	kicad (2015-05-10 BZR 5649)-product	Id: 1/1



Sheet:
File: adc.kicad_pcb

Title: ADC CEP Project

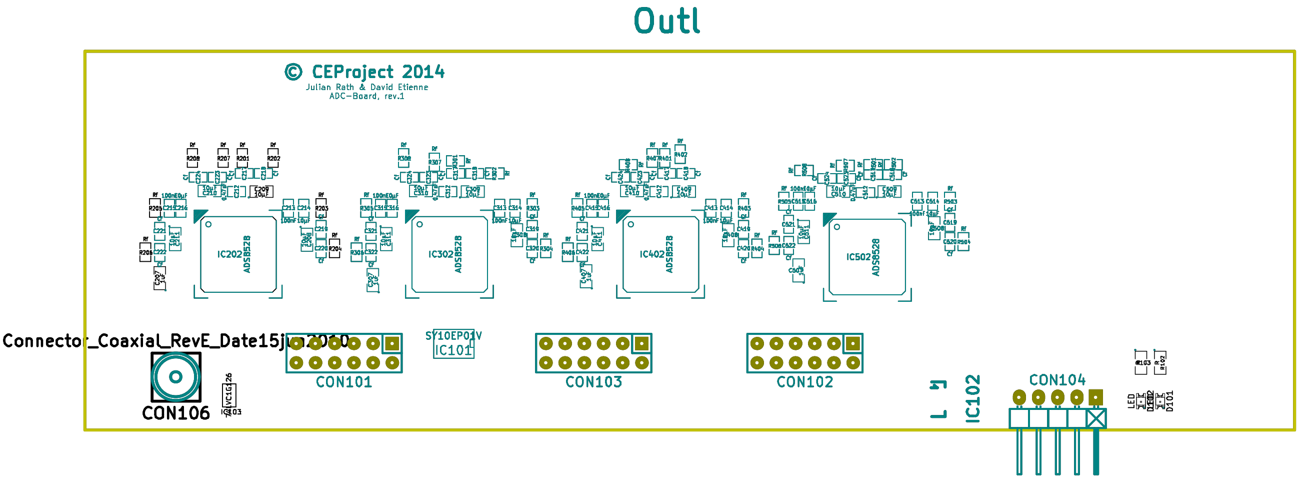
Size: A4	Date:	**Rev:**
KiCad E.D.A. kicad (2015-05-10 BZR 5649)-product		Id: 1/1



Sheet:
File: adc.kicad_pcb

Title: ADC CEP Project

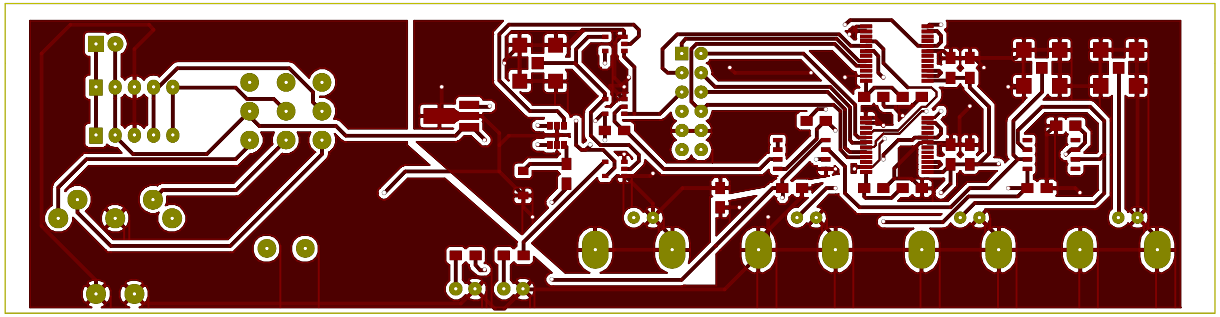
Size: A4 Date: **Rev:**
KiCad E.D.A. kicad (2015-05-10 BZR 5649)-product Id: 1/1



Sheet:
File: adc.kicad_pcb

Title: ADC CEP Project

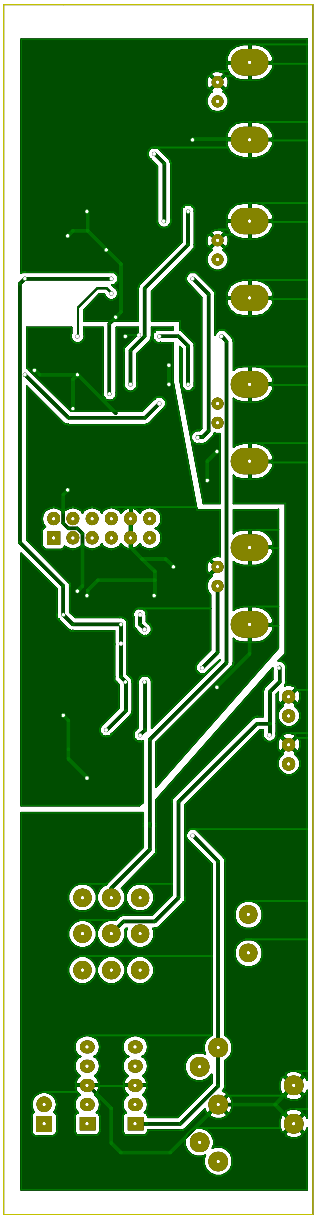
Size: A4	Date:	**Rev:**
KiCad E.D.A.	kicad (2015-05-10 BZR 5649)-product	Id: 1/1



Sheet:
File: io.kicad_pcb

Title:

Size: A4	Date:	**Rev:**
KiCad E.D.A. kicad (2015-05-10 BZR 5649)-product		Id: 1/1

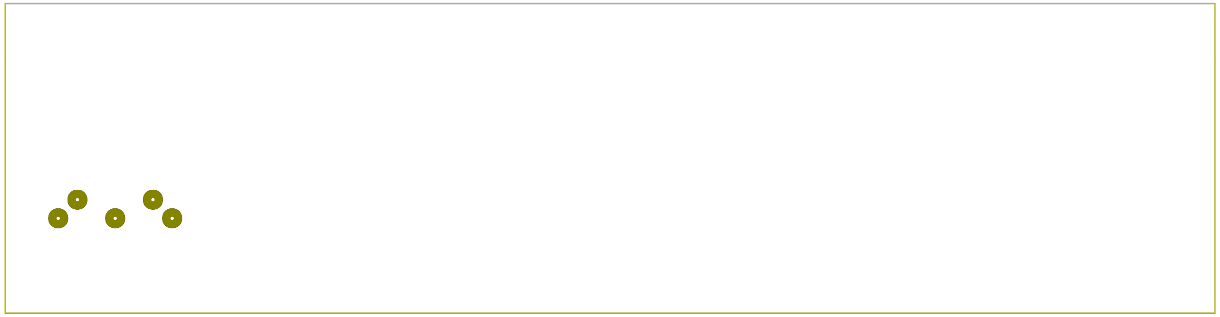


Sheet:
File: io.kicad_pcb

Title:

Size: A4 Date:
KiCad E.D.A. kicad (2015-05-10 BZR 5649)-product

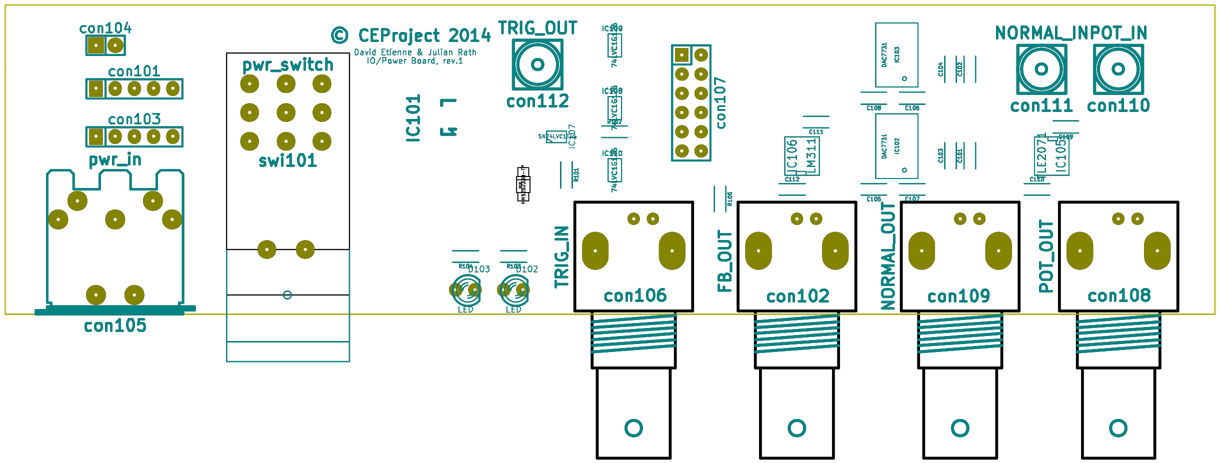
Rev:
Id: 1/1



Sheet:
File: io.kicad_pcb

Title:

Size: A4 Date: **Rev:**
KiCad E.D.A. kicad (2015-05-10 BZR 5649)-product Id: 1/1

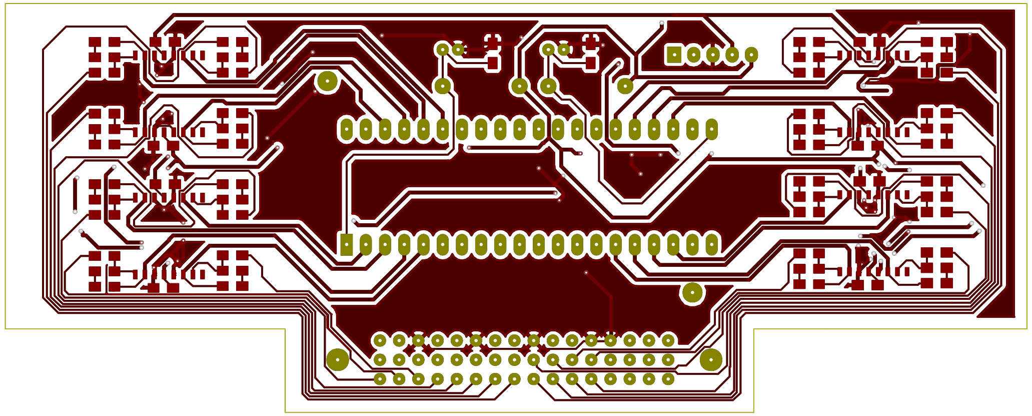


Sheet:
File: io.kicad_pcb

Title:

Size: A4 | Date: | **Rev:**

KiCad E.D.A. kicad (2015-05-10 BZR 5649)-product | Id: 1/1

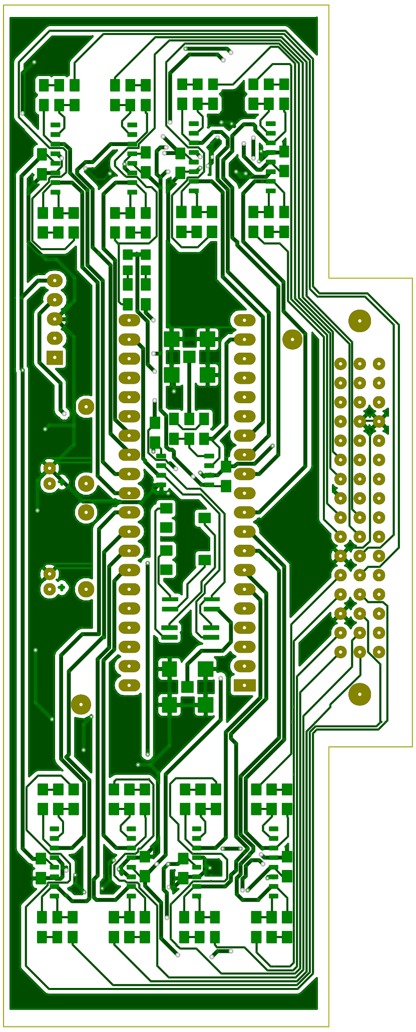


Sheet:
File: amp.kicad_pcb

Title:

Size: A4 | Date: | **Rev:**

KiCad E.D.A. kicad (2015-05-10 BZR 5649)-product | Id: 1/1

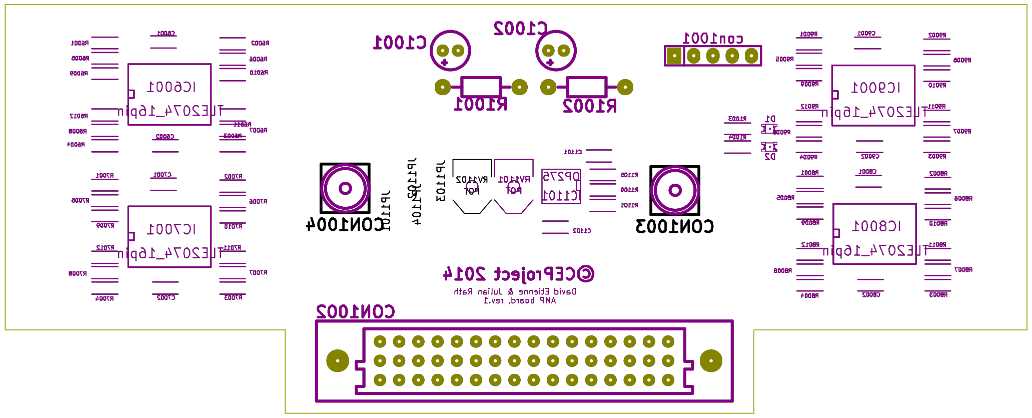


Sheet:
File: amp.kicad_pcb

Title:

Size: A4 Date:
KiCad E.D.A. kicad (2015-05-10 BZR 5649)-product

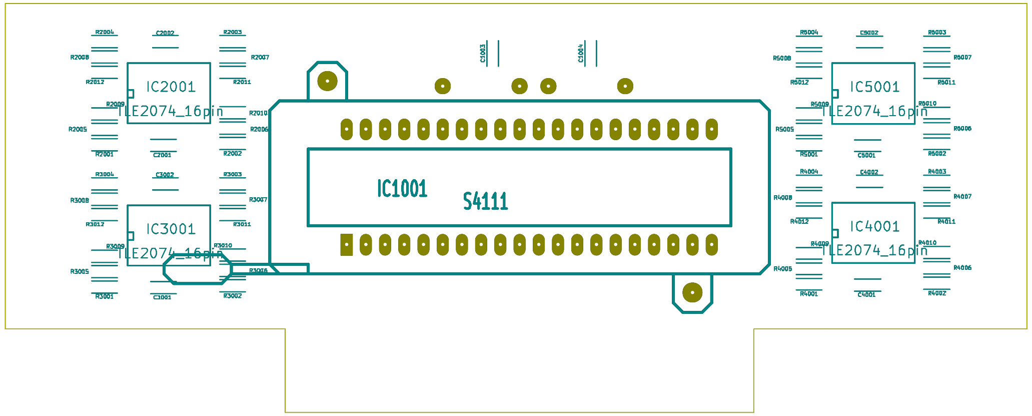
Rev:
Id: 1/1



Sheet:
File: amp.kicad_pcb

Title:

Size: A4	Date:	Rev:
KiCad E.D.A.	kicad (2015-05-10 BZR 5649)-product	Id: 1/1



Sheet:
File: amp.kicad_pcb

Title:

Size: A4	Date:	**Rev:**
KiCad E.D.A. kicad (2015-05-10 BZR 5649)-product		Id: 1/1

Acronyms

ADC Analog Digital Converter. xi, xii, 12–14, 17–21, 32, 33, 36–40, 42, 43, 48

CEO Carrier Envelope Offset. xi, 5

CEP Carrier Envelope Phase. v, vi, ix, xi, 2, 5–7, 31, 32, 34, 36, 38, 40, 42, 44, 46, 48, 50, 52

CORDIC COordinate Rotation DIgital Computer. 28, 40, 41, 43

DAC Digital Analog Converter. 19, 35, 40–43

FFT Fast Fourier Transform. 19, 27, 28, 32, 41, 43, 45, 47, 55

FPGA Field Programmable Gate Array. ix, xii, 14, 17, 20–23, 32, 35, 36, 38, 40–46, 49

GUI Graphical User Interface. 46

PCB Printed Circuit Board. ix, xii, 32–34, 39, 48

PRU Programmable Realtime Unit. 27, 44

RPC Remote Procedure Call. 24, 25, 45–47

SPI Serial Peripheral Interface. xi, 15, 18, 21, 22, 38, 40–45

VHDL Very High Speed Integrated Circuit Hardware Description Language. 22

Bibliography

[1] *12-, 14-, 16-Bit, Eight-Channel, Simultaneous Sampling ANALOG-TO-DIGITAL CONVERTERS*. ADS8528. Revised October 2011. Texas Instruments Incorporated. Apr. 2011 (cit. on pp. 37, 39, 48).

[2] *16-Bit, Voltage Output, Serial Input DIGITAL-TO-ANALOG CONVERTER*. DAC7731. SBAS249B. Texas Instruments Incorporated. Dec. 2001 (cit. on p. 41).

[3] Bonnie Baker. "A glossary of analog-to-digital specifications and performance characteristics." In: *Application Report sbaa147* (2006) (cit. on pp. 13, 20).

[4] Luke Campangiola. *http://www.pyqtgraph.org/*. URL: http://www.pyqtgraph.org/ (cit. on p. 46).

[5] Wikimedia Commons. *A Serial Peripheral Interface (SPI) bus with a single master and three slaves daisy chained together.* 2007. URL: https://commons.wikimedia.org/wiki/File:SPI_three_slaves_daisy_chained.svg (cit. on p. 21).

[6] Wikimedia Commons. *A siA single master and three slaves on a Serial Peripheral Interface (SPI) bus.ngle master and three slaves on a Serial Peripheral Interface (SPI) bus.* 2007. URL: https://en.wikipedia.org/wiki/File:SPI_three_slaves.svg (cit. on p. 22).

[7] Wikimedia Commons. *A simplified diagram of a sample and hold circuit.* 2009. URL: https://commons.wikimedia.org/wiki/File:Sample-hold-circuit.svg (cit. on p. 19).

[8] Wikimedia Commons. *An 8-bit Serial Peripheral Interface Bus transfer using two shift-registers: one in the master and one in the slave. After 8 single-bit transfers the master and slave will have transfered each other's register value.* 2007. URL: https://commons.wikimedia.org/wiki/File:SPI_8-bit_circular_transfer.svg (cit. on p. 20).

[9] SciPy developers. *http://scipy.org/*. URL: http://scipy.org/ (cit. on pp. 22, 25, 46).

[10] D Etienne et al. “Single Shot Detection of the Carrier-to-Envelope Phase of an Ultrashort Pulse Laser at high Repetition Rate.” In: Sept. 2014 (cit. on p. 2).

[11] Thomas Fordell et al. “High-speed carrier-envelope phase drift detection of amplified laser pulses.” In: *OPTICS EXPRESS, 23645* 19.24 (Nov. 2011) (cit. on pp. 2–4, 7, 8, 11, 14).

[12] Python Software Foundation. *http://python.org/*. URL: http://python.org (cit. on pp. 22, 46).

[13] Python Software Foundation. *https://docs.python.org/3/library/multiprocessing.html*. URL: https://docs.python.org/3/library/multiprocessing.html (cit. on p. 46).

[14] *Getting Started with TINA-TI^{TM}*. Texas Instruments. Aug. 2008. URL: https://www.ti.com/lit/ug/sbou052a/sbou052a.pdf (cit. on p. 33).

[15] Jerald Graeme. *Photodiode Amplifiers: OP AMP Solutions*. 1st ed. New York, NY, USA: McGraw-Hill, Inc., 1996. ISBN: 007024247X, 9780070242470 (cit. on p. 13).

[16] Harald Hartl. *Elektronische Schaltungstechnik: mit Beispielen in PSpice*. Vol. 7321. Pearson Deutschland GmbH, 2008 (cit. on pp. 11, 16).

[17] Xilinx Inc. *7 Series DSP48E1 Slice*. Nov. 2015. URL: http://www.xilinx.com/support/documentation/user_guides/ug479_7Series_DSP48E1.pdf (cit. on p. 21).

[18] Texas Instruments. *AM335x sitara processors*. May 2015. URL: http://www.ti.com/lit/ds/symlink/am3358.pdf (cit. on p. 26).

[19] U Keller. “Ultrafast solid-state laser oscillators: a success story for the last 20 years with no end in sight.” In: *Applied Physics B* 100.1 (2010), pp. 15–28 (cit. on pp. 5, 6).

[20] Simon St Laurent et al. *Programming web services with XML-RPC*. ” O’Reilly Media, Inc.”, 2001 (cit. on p. 25).

[21] Chris Liechti. *http://pyserial.sourceforge.net/*. URL: http://pyserial.sourceforge.net/ (cit. on p. 46).

[22] Miguel Miranda. “Sources and Diagnostics for Attosecond Science.” eng. PhD thesis. Lund University, 2012, p. 218. ISBN: 978-91-7473-392-1 (cit. on p. 1).

[23] Alan V Oppenheim, Ronald W Schafer, John R Buck, et al. *Discrete-time signal processing*. Vol. 2. Prentice-hall Englewood Cliffs, 1989 (cit. on p. 28).

[24] *Opto-Semiconductor Handbook*. Hamamatsu Photonics K.K. Solid State Division, 2014 (cit. on pp. 10, 14–16, 33).

[25] *Overview of Linux kernel SPI support*. Linux Kernel. Feb. 2012. URL: https://www.kernel.org/doc/Documentation/spi/spi-summary (cit. on p. 44).

[26] *Products for MATLAB & Simulink Version R2015a*. URL: https://de.mathworks.com/store/link/products/academic/new?s_iid=htb_buy_gtwy_cta2 (cit. on p. 25).

[27] *PyQt*. Riverbank Computing Limited. URL: http://www.riverbankcomputing.co.uk/software/pyqt/intro (cit. on p. 46).

[28] *Si Photodiode Array S4111/S4114 Series*. Hamamatsu Photonics K.K. Oct. 2011. URL: http://www.hamamatsu.com/resources/pdf/ssd/s4111-16r_etc_kmpd1002e.pdf (cit. on pp. 8, 9, 11, 12).

[29] D.J. Smith. “VHDL and Verilog compared and contrasted-plus modeled example written in VHDL, Verilog and C.” In: *Design Automation Conference Proceedings 1996, 33rd*. June 1996, pp. 771–776. DOI: 10.1109/DAC.1996.545676 (cit. on p. 22).

[30] JohnA. Stankovic and Krithi Ramamritham. “What is predictability for real-time systems?” English. In: *Real-Time Systems* 2.4 (1990), pp. 247–254. ISSN: 0922-6443. DOI: 10.1007/BF01995673. URL: http://dx.doi.org/10.1007/BF01995673 (cit. on p. 18).

[31] D. Stranneby. *Digital Signal Processing and Applications*. Elsevier Science, 2004. ISBN: 9780080472522. URL: https://books.google.at/books?id=NKK1DdqcDVUC (cit. on p. 17).

[32] *TLE207x, TLE207xA EXCALIBUR LOW-NOISE HIGH-SPEED JFET-INPUT OPERATIONAL AMPLIFIERS*. Texas Instruments. Dec. 2009. URL: http://www.ti.com/lit/gpn/tle2074 (cit. on p. 33).

[33] Jack E Volder. "The CORDIC trigonometric computing technique." In: *Electronic Computers, IRE Transactions on* 3 (1959), pp. 330–334 (cit. on p. 28).

[34] Dave Winer et al. *Xml-rpc specification*. 1999 (cit. on p. 25).

[35] John G Ziegler and Nathaniel B Nichols. "Optimum settings for automatic controllers." In: *trans. ASME* 64.11 (1942) (cit. on p. 29).