A High-Speed 40-channel USB 2.0 DAC for Adaptive Optics

Y.T.B. Mulder
J. Romijn
D.S.M. Verhaert

Delft University of Technology

TU Delft

This document was typeset by LaTeX.

A High-Speed 40-channel USB 2.0 DAC for Adaptive Optics

by

Y.T.B. Mulder
J. Romijn
D.S.M. Verhaert

to obtain the degree of Bachelor of Science
at the Delft University of Technology,
to be defended on Tuesday June 30, 2015 at 1:30 PM.

Y.T.B. Mulder	4232755
J. Romijn	4223802
D.S.M. Verhaert	4234685

Date of submission:	June 22, 2015	
Project duration:	April 20, 2015 – July 3, 2015	
Supervisor:	Prof.dr. G. Vdovin	Director of Flexible Optical B.V.
Thesis committee:	Prof.dr. K.L.M. Bertels	TU Delft, Jury chairman
	Dr.ir. J.A. Martínez Castañeda	TU Delft, Jury member
	Dr. S. Patlan	Senior Researcher at Flexible Optical B.V.

ISBN 978-94-6186-493-2

An electronic version of this thesis is available at `http://repository.tudelft.nl/`.

TU Delft Delft University of Technology

Preface

This document is the final product of our Bachelor Graduation Project at the Delft University of Technology in the Netherlands. This project is part of the three yearlong Bachelor Electrical Engineering program and has to be completed to officially receive a Bachelors degree and to be able to carry the name BSc. In this project students will show their academic capability and should be able to design an electronic system in a structured manner. The project officially started on the 20th of April and ended on the 3rd of July for a total period of 11 weeks.

The commissioner of our Bachelor Graduation Project is Prof. dr. G. Vdovin, full professor at the Delft Center for Systems and Control. He is also director of Flexible Optical B.V., a company specialized in application-oriented development of laser and high resolution imaging adaptive optics, located in Rijswijk, the Netherlands.

The main goal of this thesis is to develop a 'next generation' *digital-to-analog converter* (DAC), which is sold by Flexible Optical B.V. as part of their measurement systems. Two different implementations have been researched by a total of six electrical engineering students. These six students are divided into two groups of three students. The authors of this thesis have researched an implementation using *Universal Serial Bus* (USB) 2.0 in combination with a single 40-channel DAC chip.

The design choices and decisions regarding this implementation are described in this thesis. All subsystems of the final implementation are elaborated and their respective performance and results are shown and discussed.

Y.T.B. Mulder
J. Romijn
D.S.M. Verhaert
Delft, June 2015

Acknowledgements

We would like to thank the supervisor of our project Prof. dr. G. Vdovin for commissioning the assignment. Next, we would greatly like to thank Dr. Seva Patlan, Senior Researcher at Flexible Optical B.V., for meeting with us on a weekly basis and supplying us with components and knowledge. We would like to thank TransIP for sponsoring and supplying us with a server and we would also like to thank our team members: Laurens van Dam, Sander van Dijk, and Michel van der Kaay. Their positive mindset, discussion skills, and support were very valuable. Lastly, we would like to thank A.C. van Dijk from ADAS Automatisering, dr. ir. A.J. van Genderen, and dr. E.J.H. Mulder for reviewing this thesis and the European Organisation for Astronomical Research in the Southern Hemisphere for supplying us with the wonderful cover photo [1].

Abstract

In this project, a USB DAC was designed. This was done according to a proposal from Flexible Optical B.V. which is located in Rijswijk. This device is used to correct for atmospheric interference in optical measurement systems. The existing DAC is ten years old and needs to be updated to requirements of today. The 'next generation' DAC has been researched in-depth to see if the demanded requirements could be met. The design is divided into four subsystems: USB, controller, DAC, and power regulation and components have been chosen accordingly.

Research showed that it was impossible to achieve the requested refresh rate with USB 2.0. The requirements were therefore altered to accommodate this shortcoming of USB 2.0, since a working prototype was more desirable than a faster product.

The decision was then made to use a combo-unit that handles both the USB and the controller subsystem. For the DAC subsystem a single 40-channel DAC chip with a resolution of 16 bits was chosen. According to the needs of the other subsystems, a power regulation subsystem was designed.

The entire system was first tested on a breadboard. This was used for prototyping and initial measurements. These measurement results did not fully comply with the requirements and adjustments had to be made. With these adjustments eventually all of the requirements have been fulfilled.

To finalize the design and create a more stable and consistent design, a printed circuit board was designed and manufactured by Eurocircuits. However, it was not manufactured in time, and therefore no results regarding the printed circuit board are presented in this thesis.

There are still some measurements that need to be done to ensure that all requirements are met. The printed circuit board design can still be improved upon to reduce distortion and increase its compactness. Furthermore, pick and place files should be made to make the manufacturing more efficient and other accommodating documents and files should still be finished for a final, market-ready product.

Contents

1

Introduction

This thesis is the result of a project conducted in the field of digital-to-analog converters, commissioned by Flexible Optical B.V. which is located in Rijswijk, the Netherlands [2]. Flexible Optical B.V. is a small company specialized in the field of application-oriented development of laser and high resolution imaging adaptive optics.

A *digital-to-analog converter* (DAC) is an electronic device which converts a digital signal into an analog signal, for example a current or a voltage. Digital signals can be easily manipulated, stored, and transmitted without degradation whereas analog signals can not and are used to interact with the real world; thus conversion between the two domains is necessary.

Adaptive optics is a technology to correct for optical imperfections in real-time [3]. An everyday example of adaptive optics is a continuous mode auto-focus in a digital camera. More serious applications can be found in the fields of astronomy and biological imaging, where continuous mode correction is applied using a large amount of control signals to correct for complicated aberrations. Aberrations are deviations from the ideal, which lead to reduced power efficiency or reduced resolution of the system.

Lithographic lenses used in adaptive optics provide almost aberration-free imaging and are constructed with immense precise components. However, even though this precision is so high, in the real world optical systems suffer from external parameters which can not be controlled, such as atmospheric turbulence or thermal effects. The aberrations that arise from these external parameters are called static aberrations. Correcting for these aberrations is needed and herefore a smart feedback system is used to correct the lenses. The high-speed 40-channel DAC discussed in this thesis plays a crucial role in this feedback system, since it converts the digital corrections calculated by a computer to analog signals to control the deformable mirrors. Dynamic aberrations also exist, of which defocus is one of the most simple examples. For example, the human eye needs to change focus when the distance to an object is changed. The same holds for optical systems and that is the reason for their adjustable design.

In short, adaptive optics is a rapidly growing branch of applied optics, in which the feedback system plays an ever increasing role in improving the overall performance of the system.

The ’next generation’ USB DAC, which will be designed during this project, will have improved specifications over the already existing DAC developed ten years ago by Flexible Optical B.V. The main improvements compared to the existing DAC are an increased refresh rate and a higher resolution, while maintaining the same form factor.

The focus of this thesis is the structural design process and implementation of the ’next generation’ DAC, in particular the USB 2.0 implementation in combination with a single 40-channel DAC chip. The problem of this thesis is defined in Chapter 2 and the conducted research is discussed in Chapter 3. The final implementation is discussed in Chapter 4, followed by the achieved results in Chapter 5. These results are discussed in Chapter 6 and Chapter 7 discusses the ethical aspects of the project. Finally, project-wide conclusions are drawn and future work is discussed in Chapter 8.

2

Problem Definition

This chapter defines the main problem of this bachelor thesis. The existing product manufactured by Flexible Optical B.V. will be discussed, after which the program of requirements for the 'next generation' version will be summarized.

2.1. Existing Product

Ten years ago, Flexible Optical B.V. developed a 40-channel USB DAC. The main purpose of this USB DAC is to control deformable mirrors used in adaptive optics. A block diagram of a basic measurement setup is shown in Figure 2.1. The source signal, for example light from a telescope, transmits a signal which is prone to disturbances and results in a distorted wavefront. The wavefront is reflected off and corrected by a wavefront sensor. This correction is applied by using a camera in a negative feedback loop, which detects the wavefront after it has been reflected and sends the data to a control system, which in turn calculates the correction for and adjusts the deformable mirrors.

The deformable mirror is controlled by many analog input signals, which control either actuators or an electromagnetic field to deform the mirror's surface. To transfer the calculated correction from the control system to the mirrors, a DAC is needed and upgrading the existing product is the main goal of this project. The specifications of the existing model are summarized in Table 2.1.

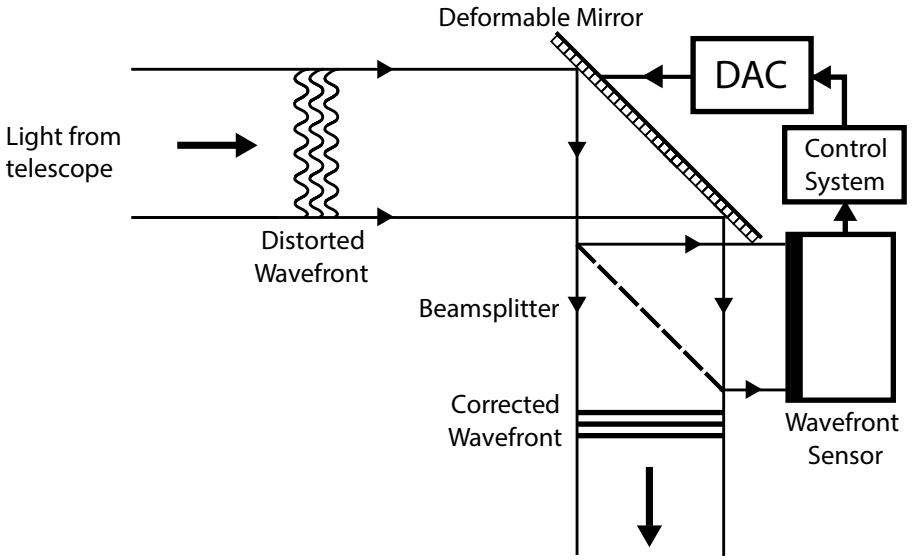


Figure 2.1: Basic adaptive optics measurement setup [4].

2.2. Program of Requirements

In this section, the various requirements demanded by Flexible Optical B.V. are summarized and elaborated on. The 'next generation' DAC is mainly used in the professional market for adaptive optics measurement setups. The applicable fields of research have grown in the past years and a faster and more precise DAC is demanded. After this project, Flexible Optical B.V. will manufacture and distribute the product.

2.2.1. Demanded Specifications

Table 2.1: Existing and 'next generation' DAC specifications [2].

Parameter	Existing DAC	'Next generation' DAC
Analog outputs	40 channels	40 channels
Refresh rate	1,000 frames per second	20,000 frames per second
Resolution	12 bits per channel	16 bits per channel
Output mode	Synchronous for all channels	Synchronous for all channels
Output range	0 V - 5 V with adjustable limits	0 V - 5 V or −5 V - 5 V with adjustable limits
Power	Supplied by USB	Supplied by USB
Load capacitance	> 500 pF	> 500 pF

Flexible Optical B.V. has stated certain technical requirements in their proposal. The existing and 'next generation' specifications can be found in Table 2.1. The biggest shortcoming of the existing DAC is the refresh rate of 1,000 frames per second and the resolution of 12 bits. An increase in refresh rate of the DAC is demanded because it should match the typical frequencies of the disturbances. For example, for atmospheric turbulence the typical frequency is in the range of 100 Hz to 10 kHz [3], depending on the conditions. Introducing a faster DAC device would allow for an expansion of applicability since applications of adaptive optics are nowadays not limited to astronomy, but also used in biological imaging and many other fields of research. Increasing the resolution results in a greater dynamic range and a higher level of precision to control the deformable mirrors.

Analog Outputs

A synchronous output for all 40 channels is necessary since otherwise the mirrors will not properly form. Deformable mirrors manufactured by Flexible Optical B.V. have between 17 and 109 control signals. Most of them have under 40 control signals and these could be controlled with the 'next generation' 40-channel DAC, developed in this project.

Output Range

Furthermore, the output voltage is requested to be either unipolar, with a range of 0 V to 5 V, or bipolar, with a range of −5 V to 5 V. The user should be able to adjust the outputs if necessary for a particular situation. An analog unipolar output voltage can only have one polarity, whereas an analog bipolar output voltage can have both positive and negative polarity. In the unipolar case, one bit equals

$$\frac{V_{span}}{2^{bits}} = \frac{5\,V}{2^{16}} = 76\,\mu V. \tag{2.1}$$

In the bipolar case, one bit equals

$$\frac{V_{span}}{2^{bits}} = \frac{10\,V}{2^{16}} = 152\,\mu V. \tag{2.2}$$

A bipolar output range is more desirable since this increases the voltage of the least significant bit and makes the DAC less prone to errors.

Power

Supplying the final product with power through the USB connection is purely out of convenience, due to the fact that only one cable is needed to connect the USB DAC to a computer. Also, USB is nowadays present on nearly every computer, which makes it widely employable.

Load Capacitance

The load capacitance parameter specifies the capacitive load to be driven. To guarantee that the voltage amplifier, which amplifies the analog signals from the DAC, can operate correctly, the final product should be able to drive capacitive loads larger than 500 pF.

Latency

A parameter that has not been addressed explicitly is the latency of the product. The deformable mirrors will be compensated in real time, since every moment that the mirror is not formed properly, data is wasted. In the ideal case, the latency is zero seconds, but this is of course never possible. There is always a trade-off between data throughput and latency. In this case, the focus lies on minimizing latency and Flexible Optical B.V. mentioned that it should lie in the order of one updated frame. An acceptable latency lies in the range between

$$\frac{1}{\text{Refresh rate}} = \frac{1}{20.000\,\text{kHz}} = 50\,\mu\text{s} \tag{2.3}$$

and

$$\frac{2}{\text{Refresh rate}} = \frac{2}{20.000\,\text{kHz}} = 100\,\mu\text{s}. \tag{2.4}$$

Flexible Optical B.V. requests that all these parameters are updated to standards of today. At the end of the project, a completely functional DAC prototype has to be presented. This prototype has to be conform to the technical specifications mentioned earlier, but also with the existing form factor. In other words, all of the components of the 'next generation' DAC have to fit inside the same box as the existing DAC. The in- and outputs have to be in the same location and the maximum dimensions of the *Printed Circuit Board* (PCB) have to be 77 mm by 135 mm.

2.2.2. Demanded Documents and Files

The following accompanying documents and files should be handed over to Flexible Optical B.V after the project for replication, fabrication and assembly.

- Schematic of the device.
- Bill of materials with catalog numbers from a major supplier.
- Results of computer simulations.
- PCB design in a popular CAD package accompanied by Gerber files.
- Pick-and-place file for the PCB.
- Any other files necessary for assembly and fabrication.
- Complete set of software, such as drivers and firmware.
- Set of example programs for both Linux and Windows.
- Wrappers for C, LabVIEW, and MATLAB.

3

Research

In this chapter, the system as a whole will be split up into various subsystems and each subsystem will be researched and evaluated in their corresponding section. At the end of each subsection, a conclusion will be drawn based on the conducted research in order to choose the best suitable component for the final version of the product.

3.1. System Overview

The very first step in structured system design is to divide the system into several subsystems. As specified by Flexible Optical B.V. in the proposal, the input of the system is either USB 2.0 or USB 3.0. The USB subsystem handles both data and power. The data packets received from the computer contain both digital signals and control signals. These digital signals should be converted to analog signals whereas the control signals are used to control the DAC chip. The job of the controller is to distinguish both types of data. It should present the signals for digital-to-analog conversion to the DAC through a protocol which the DAC chip understands and it should use the control data to control the DAC. The DAC chip is the final subsystem in the signal chain. Since all these subsystems require proper power, the last subsystem is the power regulation subsystem which regulates power and ensures that noise is suppressed as much as possible. A block diagram of the complete DAC system can be seen in Figure 3.1. Each subsystem will be researched and evaluated in the following sections.

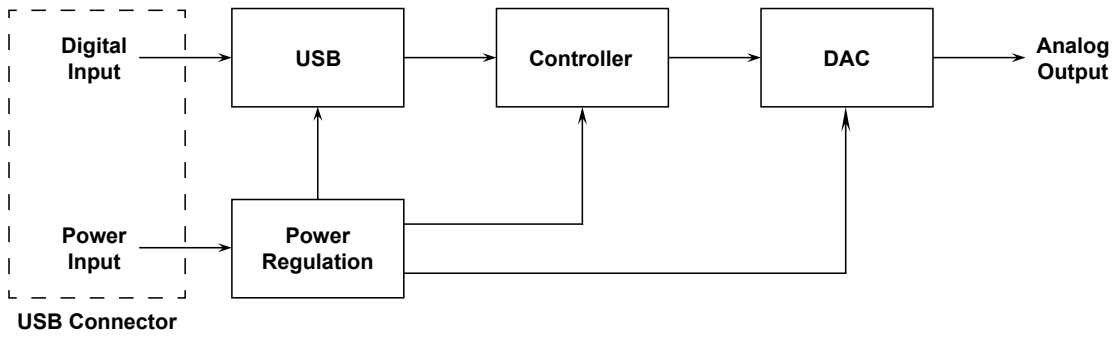


Figure 3.1: Block diagram of the USB DAC.

3.2. Universal Serial Bus Subsystem

Universal Serial Bus (USB) is a protocol designed to standardize the connection of computer peripherals to personal computers, both to communicate and supply electrical power. It is designed and maintained by leading companies in electronics such as Compaq, DEC, IBM, Intel, Microsoft, NEC, and Nortel. USB was mainly designed for its simplicity, especially plug-and-play and compatibility, rather than high performance [5]. Besides computers, USB has become commonplace on other devices too, such as smart-phones and tablets. USB has effectively replaced a variety of earlier interfaces, such as serial and parallel ports, as well as separate power chargers for portable devices [6].

USB has been used and improved a lot since its introduction in 1996, and several versions have been released since. These versions, their release date, and transfer speed can be found in Table 3.1.

Table 3.1: Version history overview of USB with their respective speeds [7] [8].

Release name	Release date	Speed and max. signaling rate
USB 1.0	January 1996	Low Speed (1.5 Mbit/s), Full Speed (12 Mbit/s)
USB 1.1	August 1998	Low Speed (1.5 Mbit/s), Full Speed (12 Mbit/s)
USB 2.0	April 2000	High Speed (480 Mbit/s)
USB 3.0	November 2008	SuperSpeed (5 Gbit/s)
USB 3.1	July 2013	SuperSpeed+ (10 Gbit/s)

USB 1.0 is the first official version of USB. It was released with two possible speed modes, low speed (1.5 Mbits/s) and full speed (12 Mbits/s). Two years later USB 1.0 was replaced by USB 1.1. This version fixed issues identified in USB 1.0 and was the first version that was widely adopted.

USB 2.0 was released in April 2000, adding a higher maximum signaling rate of 480 Mbits/s called high speed. This is faster than low speed and full speed, although USB 2.0 is also compatible with these two lower speeds. USB 2.0 is still the most commonly used protocol by far, since it is compatible with practically every device and there is much documentation available.

USB 3.0 and 3.1 are the two newest versions of USB. With these versions new speeds were introduced, namely SuperSpeed (5 Gbit/s) and SuperSpeed+ (10 Gbit/s) for USB 3.0 and USB 3.1 respectively.

Table 3.2: USB 3.0 and 3.1 connector pin-outs [9]. Pins 1 to 4 are referred to as the USB 2.0 pins, while pins 5 to 9 are referred to as the SuperSpeed pins and are exclusive to USB 3.0 and 3.1. The D in D- and D+ stands for Data, where STdA_SSR and STdA_SST stand for Standard-A connector for SuperSpeed receiver and transmitter respectively.

Pin	Signal name	Description
1	VBUS	USB Power
2	D-	USB differential pair
3	D+	
4	GND	USB Ground
5	StdA_SSRX-	USB 3.0 SuperSpeed receiver differential pair
6	StdA_SSRX+	
7	GND_DRAIN	USB 3.0 Ground
8	StdA_SSTX-	USB 3.0 SuperSpeed transmitter differential pair
9	StdA_SSTX+	

The upgrade from USB 1.1 to USB 2.0 was achieved by using the same amount of pins and optimizing the data throughput through these pins. Therefore all versions up to USB 2.0 consists of four pins, one for power, one for ground, and two for data [9]. USB 3.0 and 3.1 were introduced with five extra pins. One of these pins is ground and the other four pins form two differential pairs and are for SuperSpeed or SuperSpeed+ data transfer. An overview can be found in Table 3.2. Since the older USB connectors only contain four pins, it is impossible to use USB 3.0 or 3.1 at Superspeed or Superspeed+ in older connectors. When USB 3.0 or 3.1 are connected to older connectors, their five new pins will not be

connected and they will only utilise the standard four pins using the old speeds, making them backwards compatible.

The older versions, USB 1.0 and 1.1, are outdated and have very low speeds. USB 3.1 has increased speed and can definitely prove to be advantageous, but is still new and therefore not commonly supported and rarely found on personal computers. There is almost no literature available and it is more expensive. Therefore, realistic options for this project are USB 2.0 using high speed and USB 3.0. Research in these two types has been conducted, with a focus on the refresh rate, transfer types, and power capabilities. The results can be found in this section.

3.2.1. Speed

With USB 2.0, the host receives data at a certain frequency which is called the refresh rate. This data needs to be sent with a high signaling rate and a low latency. Commonly, these terms are often confused with each other, especially the refresh rate and the signaling rate. Therefore this subsection will focus on explaining these terms.

- **Signaling rate** is the rate at which the data is transferred from the host to the receiver and is defined as data transfers per second, mostly being in bits or bytes per second.
- **Refresh rate** is the rate the host can communicate with the device and determines the maximum time between data transfers. It is defined in Hertz, which is one cycle per second.
- **Latency** is the delay between the sent input signal and the corresponding received output signal defined in seconds.

Signaling Rate

The signaling rate does not have a big impact on the DAC, since the data throughput does not have to be that high, but the refresh rate does. Roughly estimated and not taking control or any other signals in consideration, the data throughput equals

$$16\,\text{bits} * 40\,\text{channels} * 20\,\text{kHz} = 12.8\,\text{Mbit/s}. \tag{3.1}$$

Compared to the earlier mentioned speeds, this is easily achieved with either USB 2.0 high speed of 480 Mbits/s or USB 3.0 SuperSpeed of 5 Gbits/s.

Refresh Rate

As discussed earlier, USB 2.0 has three modes: low speed, full speed, and high speed. The shortest time for a bus interval of USB 2.0 using low and full speed is defined as 1 ms [9]. This implies that a maximum refresh rate of

$$f = \frac{1}{T} = \frac{1}{1\,\text{ms}} = 1\,\text{kHz} \tag{3.2}$$

can be achieved, with f the frequency and T the period. This calculation also holds for high speed, of which the shortest time of an interval is 125 µs. This results in a maximum refresh rate of

$$f = \frac{1}{T} = \frac{1}{125\,\mu\text{s}} = 8\,\text{kHz}. \tag{3.3}$$

Data will be polled at 1 kHz for low and full speed, and at 8 kHz for high speed.

An overview of how polling works can be found in Figure 3.2. At every polling interval, represented by a black line, the USB protocol will check if a packet is ready to be sent and, if this is the case, it will process the packet. The upper part of the figure shows full/low-speed frame sizes whereas the lower part shows high-speed micro-frame sizes, which are eight times as short. The time that it takes to send the packet differs heavily per transfer, because the USB protocol handles aspects like queuing and errors differently depending on the transfer type used, which in this figure is isochronous. Transfer types will be discussed in Section 3.2.2.

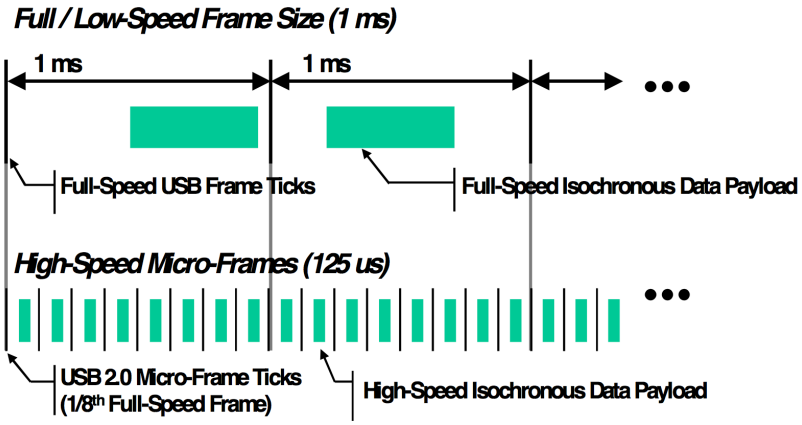


Figure 3.2: Overview of USB 2.0 polling [6].

Where USB 2.0 works with polling, USB 3.0 in SuperSpeed mode works with asynchronous notifications. These asynchronous notifications come from the device and use a point-to-point link between the host and each device to indicate a request to send data to the device. Therefore the polling interval no longer exists. Due to this difference in operation, a much higher refresh rate can be achieved. However this also makes USB 3.0 depend on many other factors like hardware, the cable used, using a laptop in power saver mode and even minor things like moving the mouse while measuring. Therefore, a definite value is hard to determine. A possibility is to test a USB 3.0 transmission with an evaluation board and measure the refresh rate. This is costly, time consuming, and outside of the scope of this project, since a working prototype has to be delivered.

Latency

The latency of USB 2.0 using high speed and USB 3.0 will now shortly be discussed in light of the just discussed USB polling. In the optimal case, a packet is ready to be sent in an infinitely small time before the bus interval. If the packet gets processed and sent immediately, depending on the package and the timings in the protocol, it is possible that it arrives at the host after only several microseconds and so the latency of this packet will only be this long. Therefore the minimum latency of USB can not be clearly defined since it is dependent on uncontrollable factors.

In the worst case scenario, a packet is ready to be sent in an infinitely small time after the bus interval. As soon as the next bus interval has passed it might take a while to process this package. Furthermore errors can occur which might result in USB resending the package or it never arriving at all depending on the transfer type used. Therefore the maximum latency of USB can also not be defined.

Because of all these factors it is incredibly hard to make an assumption about the latency of USB. For simplicity the average latency of USB 2.0 is in this thesis assumed to be 125 µs, the same length as the bus interval.

3.2.2. Transfer types

USB uses four types of transfers [10], each optimized for a different purpose. These transfer types are present since the beginning of USB and therefore are used on every version of USB. They are used to standardize the communication of computer peripherals and must be applicable to a variety of applications. The different types of transfers are the following.

- **Control** transfer is the only bidirectional transfer type. Control packets are typically bursty, random packets, which are initiated by the host and use best effort delivery. This kind of data transfer often occurs for identification and configuration of new devices.
- **Bulk** transfer is optimized for large, bursty data. It is an unidirectional transfer type and mostly used when a high data rate is required. A good example for bulk transfer is a transfer between a computer and a USB stick, or for example a print-job.

- **Interrupt** transfers are useful when data has to transfer within a specific amount of time, meaning with the shortest amount of latency. It is mostly used for devices which generate the signals themselves and send these to the host, such as keyboards or mice.
- **Isochronous** transfers are streaming, real-time transfers that are useful when data has to arrive at a constant rate. These transfers are optimized for a high refresh rate.

The control transfer type is used for bidirectional transfers and the other three are used to optimize the signaling rate, the latency, and the refresh rate. Characteristics per transmission mode are listed in Table 3.3. In this table the maximum signaling rate of SuperSpeed and SuperSpeed+ are not stated, since these rates are heavily dependent on external factors as stated earlier. Note that the bulk and isochronous transfer types do not support low speed.

Table 3.3: USB maximum signaling rate per transfer type [10]. Assumes transfers use maximum packet size. The signaling rate is in bytes per millisecond. An empty space means that speed is not allowed with the corresponding transfer type.

Maximum signaling rate	Control	Bulk	Interrupt	Isochronous
Low speed [B/ms]	24	-	0.8	-
Full speed [B/ms]	832	1,216	64	1,023
High speed [B/ms]	15,872	53,248	24,576	24,576

Since the data in this USB DAC has to be transferred as real-time as possible the initial idea was to use isochronous transfer, since this transfer type prioritizes refresh rate over actual reliable data throughput and a high signaling rate. Control was not chosen since this protocol is optimized for bidirectional transfers, mostly being identification and configuration, and not for its signaling rate and refresh rate. Bulk and interrupt were not chosen because their refresh rate was thought not to be able to achieve the maximum refresh rate of USB 2.0, being 8 kHz. However, research led to the following: bulk and interrupt are not optimized for transfer while the bus is busy and in that case these kinds of data may slowly trickle over the bus [11], but in the case of this USB DAC, the bus is exclusively used by one device and so the refresh rate of these modes will not be negatively affected. Therefore the refresh rate of bulk, interrupt, and isochronous are identical and equal to the maximum refresh rate of 8 kHz.

The disadvantage of isochronous mode is that it does not check if data has been received; it is not built in with retry or guarantee of delivery. Because bulk has a guarantee of delivery and interrupt has a next period retry built in, both make sure that the data is present in the next period and both are a solution for this project.

3.2.3. USB Power

The requirements dictate that the power supply should originate from the USB connection. The most interesting parameters are the current and voltage supplied by USB, and their stability.

To be able to work with multiple devices, two hub types are defined in USB, being low power and high power. A hub is a device that expands a single USB port to multiple ports. Low power can draw up to 100 mA and high power can draw up to 500 mA. The voltage properties of both hub types are listed in Table 3.4. It shows that the voltage swing is rather large and will have to be corrected in the implementation.

Table 3.4: Voltage properties of a USB 2.0 connection in non-transient conditions [10].

Hub type	Minimum value	Nominal value	Maximum value
Low Power	4.4 V	5 V	5.25 V
High Power	4.75 V	5 V	5.25 V

The previous table shows that there is a voltage swing of 0.85 V peak-to-peak in low power and 0.5 V in high power. High power has the lowest voltage swing, this corresponds to a possible deviation of ±5 % from the nominal value. In addition, the voltage can briefly drop to values as low as 4.07 V in transient conditions [10]. These transient conditions only occur for example when the cable is connected or disconnected. The voltage properties of USB 3.0 are the same as for USB 2.0. The current however can be as large as 900 mA and devices can deliver current to the host [9].

The final device will be USB powered and the power specifications for all subsystems need to be determined before the power regulation subsystem can be designed. During the selection of components for the various subsystems, the current and voltage specifications need to be taken into account, since a limited amount of power can be supplied by the USB connection. Also the voltage swing of USB needs to be corrected to guarantee correct operation. Section 3.5 discusses the power regulation subsystem more in-depth.

3.2.4. Concluding Remarks

Important parameters of the USB subsystem are the transfer type, power specifications and most importantly the refresh rate. Either bulk or interrupt transfer proves to be the most optimal transfer type since it can achieve the requested maximum refresh rate and has error correction. The power supplied by the USB connection is not stable enough and has to be regulated to guarantee correct operation of all subsystems, but in terms of maximum current, USB 3.0 can supply more than USB 2.0. The estimated refresh rate of USB 2.0 is 8 kHz whereas the refresh rate of USB 3.0 is more promising, but its actual refresh rate is hard to determine. If there was more time available, USB 3.0 could have been researched. However, considering all factors and the scope of this project, the choice has been made by both the students and Flexible Optical B.V. to use USB 2.0: a working product is preferred over a faster product. The requested refresh rate has therefore been modified to the maximum of USB 2.0, which is 8 kHz. This is still eight times higher compared to the existing model. The change of this refresh rate also implies a change in the acceptable latency, and using the same formula as in Chapter 2, it now lies in the range between 125 µs and 250 µs.

3.3. Controller Subsystem

In this section, the controller subsystem is discussed, which serves as an intermediary device between the USB controller and the DAC. It needs to read and decode data from the USB subsystem and make sure that the received packets are converted into a protocol that can be used by the DAC. The controller might also need extra pins to control certain functions of the DAC. These three aspects will now be discussed.

3.3.1. Controller Input

As discussed in Section 3.2, USB 2.0 will be used for this project and the controller will receive all its input signals from this subsystem. Recall that USB 2.0 has four pins: two are used for power and two for a differential data pair. The data received from the USB connection will be processed by a USB protocol engine and converted to either parallel or serial output signals. In our case, either a parallel or serial protocol has to be found to control and send data to the DAC.

3.3.2. Protocol

The data from the USB connection can be communicated over a channel or bus in two ways, being parallel or serial. In serial communication, one bit of data is sent at a time whereas with parallel communication several bits are sent at the same time [12]. Nowadays, serial communication is mostly used since synchronizing parallel communications gets more difficult as signal frequencies increase: when this happens signal transit time can not be guaranteed to be equal for all data lines. The higher the frequency, the more deviations will impact the signal frequency. The receiver of the data will have to wait untill all data is synchronized, which lowers the refresh rate. Therefore serial communication is a better and easier choice.

Several serial communication protocols exist, each optimized for a certain field. In embedded systems, widely adopted protocols are I^2C, SPI, and USB [13], where I^2C and SPI are commonly used between integrated circuits. Since these two protocols and its derivatives are so widely used and many information is available, only these two will be considered and briefly discussed, since the scope of this project does not allow for more exotic solutions.

I^2C Protocol

The *Inter-Integrated Circuit* (I^2C) protocol is designed for synchronous serial communication between micro-processors and other ICs. It allows multiple 'slaves' to communicate with one or more 'masters' [14] and requires a mere two pins. Therefore I^2C is incredibly pin efficient compared to other protocols. The *Serial Data* (SDA) line is responsible for all data communications, while the *Serial Clock* (SCL) bus line synchronizes the communications by means of a clock signal.

I^2C data transfers have five different transfer speeds developed over the years. Bidirectional data transfer speeds of 100 kbit/s, 400 kbit/s, 1 Mbit/s, and 3.4 Mbit/s can be achieved and the so-called Ultra Fast-mode, which is an unidirectional data transfer speed, can achieve up to 5 Mbit/s [15]. An unidirectional transfer is sufficient for communication between the controller and the DAC, but bidirectional transfer would allow read back from the DAC, to for example check certain settings [16].

SPI Protocol

The second option is the *Serial Peripheral Interface* (SPI), also designed for serial communication between master and slave [17]. Unlike I^2C, SPI can only support one master, is purely bidirectional, and uses a total of four pins if one slave is used. *Serial Clock* (SCLK) provides the clock rate from master to slave, while the *Master Output Slave Input* (MOSI) and *Master Input Slave Output* (MISO) transfer data between the two. Finally, *Slave Select* (SS) selects the desired slave to transfer data to or from. Every added slave requires a new SS pin, which makes the pin count increase rapidly if many slaves are used. SPI has a relatively high throughput and is quite flexible. A maximum bit rate of 100 Mbits/s can be achieved [18].

Both protocols are summarized in Table 3.5. Most important for this project are the transfer speed, the complexity, and the amount of pins needed. As shown in the table, SPI is faster than I^2C and the hardware is easier compared to I^2C, even though I^2C uses two pins. Due to the modified requirements, USB 2.0 will be used and the rough estimated data throughput, as calculated in Section 3.2.1, has changed to

$$16\,\text{bits} * 40\,\text{channels} * 8\,\text{kHz} = 5.12\,\text{Mbit/s}. \tag{3.4}$$

Even control signals have not yet been taken into account, and I^2C proves to be inadequate. Thus choosing SPI seems rather obvious, since it has a faster and bidirectional data transfer, which allows read back from the DAC. The extra two pins needed for SPI are a small price to pay compared to these advantages.

Table 3.5: Comparison between I^2C and SPI. The maximal transfer speed of I^2C is for the unidirectional case and n is the amount of slaves connected.

Parameter	**I^2C**	**SPI**
Maximal Transfer Speed	5 Mbits/s	100 Mbits/s
Amount of Connections	2	3 + n
Multiple Masters	Yes	No
Hardware Complexity	High	Low

3.3.3. Control Signals

Control signals are needed to control certain functions of the DAC, or even the entire DAC. Therefore the chosen controller has to have *General-purpose input/output* (GPIO) pins. At the simplest level, a GPIO pin is a switch that can be turned on or off at any time to, for example, reset the DAC at the

beginning of its runtime [19]. When for example two 20-channel DACs are controlled through SPI, five pins are needed for the SPI interface, but also control pins and pins to synchronize both DACs are needed. Multiple DACs result in a more complex control system which needs more pins and since all outputs need to be updated simultaneously, extra pins are needed to synchronize both DACs. A single 40-channel DAC can be controlled through SPI with four pins. The specific amount of GPIO pins needed depends heavily on the chosen DAC and an estimation can not be given without choosing one.

3.3.4. Concluding Remarks

The controller is an intermediary between the USB controller and DAC. The data received from the USB connection needs to be converted to a serial protocol, which was chosen to be SPI due to its fast transfer speed and ease of implementation. The controller also needs to have a yet to be determined amount of GPIO pins to control functions of the DAC or DACs, depending on the final implementation. From the point of view of the controller, a single 40-channel DAC would be easier to implement than multiple DACs, since a smaller control system and less pins are needed.

3.4. Digital-to-Analog Converter Subsystem

The DAC subsystem converts the digital data signals from the host to an analog signal. This is necessary because the deformable mirrors are controlled using analog signals. The mirrors deform due to electro-magnetic fields, which have to be controlled by analog voltages. The digital signals can easily be manipulated, stored, and transmitted without degradation.
There are several types of DACs available, but since the deformable mirrors are voltage controlled, only the most commonly used DACs with a voltage output are discussed. These are Delta-Sigma DACs, pulse-width modulated DACs, and resistive DACs.

3.4.1. Delta-Sigma DACs

Delta-Sigma DACs are commonly used in integrated circuit solutions. The name Delta-Sigma relates to the core working principles of such a DAC, where Sigma (Σ) represents the summing over the difference Delta (Δ). Delta-Sigma modulators are available for both analog-to-digital and digital-to-analog conversion. In this section, the heart of a Delta-Sigma DAC is discussed, which is the Delta-Sigma modulator [20].

Figure 3.3 shows a block diagram of a first-order Delta-Sigma modulator on a high level [21] [22]. Many Delta-Sigma DACs have signal shaping sections to guarantee correct operation. However, this signal shaping is outside of the scope of this research and will therefore not be discussed. The main goal of Delta-Sigma modulation is to increase transmission efficiency by only transmitting the change, called delta, of consecutive samples of the input signal, rather than the actual samples [23]. The input signal is fed into a subtractor and the difference is taken between the input and output of the comparator, which is fed through an ADC. When the output of the comparator is 0, the input signal is directly passed to the integrator, which sums the input until it triggers the comparator. At that point, the output changes from 0 to a reference voltage [24]. To smoothen the output of the comparator, a *low-pass filter* (LPF) is used.

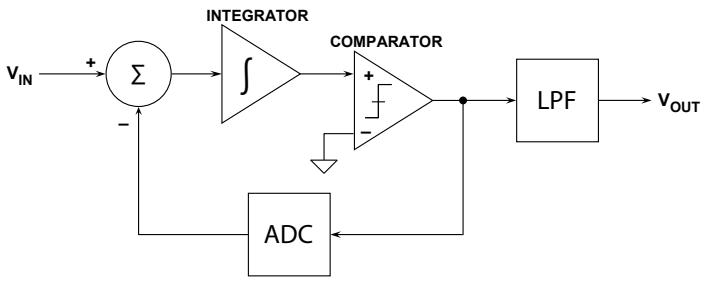


Figure 3.3: Block diagram of a Delta-Sigma modulator on a high level [25].

Advantages of a Delta-Sigma DAC are their low cost and high accuracy. Also noise can be filtered away for a large part because of the ability to either increase the clock rate, to oversample the input rate, or to use a higher order Delta-Sigma modulator. Though Delta-Sigma DACs have many advantages which seem very promising, their circuits are of high complexity, mainly higher order modulators. Most chips are tailored towards audio solutions and have a maximum of eight output channels, which is not enough for this project. Multiple chips could be used in parallel, but synchronization algorithms are needed to guarantee correct operation.

3.4.2. Pulse-Width Modulated DACs

Another implementation could be *Pulse-Width Modulation* (PWM), which is a form of signal modulation in which the data is represented as square waves with equal amplitudes but different periods.

When using PWM, the period of a signal is expressed as the ratio of the time of a high signal to the total time of that period, also known as the duty cycle. When the duty cycle is low, it corresponds to a low voltage. The duty cycle is expressed in a percentage, where 100 % means the signal is never low in a period.

In theory, a PWM DAC transforms the offered digital input signal to a PWM signal, after which it is sent to a low-pass filter which removes high frequency components and ideally leaving only the *Direct Current* (DC) component [26]. This implementation is widely used in low-cost applications [27]. For example, many micro-controllers already have a PWM output. Figure 3.4 shows a PWM DAC, where the input PWM signal enters a low-pass filter, after which only the DC component should be left.

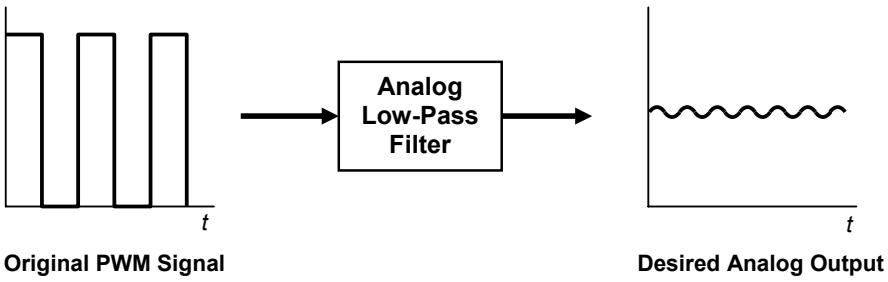


Figure 3.4: Block Diagram of a PWM DAC conversion [28].

Advantages of PWM DACs are their simplicity and low cost of the low-pass filter used. However, since the low-pass filter is a first order filter, the roll off of 20 dB/decade is rather sluggish which in some applications may not be sufficient. Switching signals are filtered by the first order filter, but the resulting signal has a ripple which bottlenecks the achievable resolution of the system. This ripple is clearly visible in Figure 3.4. Higher order or expensive filters could be used, but there is not enough space available. In short, PWM DACs are an easy and cheap way for low resolution DACs, but the required resolution of 16 bits would not be feasible [28].

3.4.3. Resistive DACs

Resistive DACs are a third feasible implementation. These DACs consist of resistor networks which means that they are completely different from the previous two implementations, which both use low-pass filters. The main advantage of resistive DACs is that they can be implemented with uni-polar or bi-polar output ranges by simply applying a negative or positive voltage to the input. In the following section, various kinds of resistive DACs are discussed.

Binary Weighted DAC

A binary weighted DAC uses the ability to add binary weighted currents, which are obtained by applying a voltage over the resistors which each represent a bit. The total current is then converted to an output voltage using an operational amplifier [29]. This type of DAC is one of the fastest implementations available, since the addition is done in the analog domain. The structure of a N-bit binary weighted DAC is shown in Figure 3.5. In this figure, LSB stands for *Least Significant Bit*, which determines whether the number is even or odd. MSB stands for *Most Significant Bit*, which is the bit position in a binary number having the greatest value.

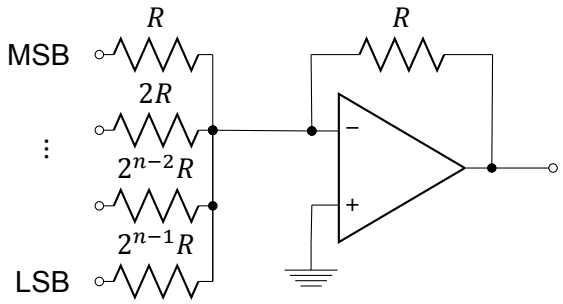


Figure 3.5: Architecture of a N-bit binary weighted DAC.

The binary weighted DAC requires less resistors than the R2R DAC, which will be discussed later, per bit. However the values of these resistors should be doubled per bit used, requiring a large variety of resistor values. Since the default resistor is increased with factors of two, the resulting resistor values can be hard to obtain without combining resistors in series or parallel. Doing this results in a decreased accuracy of the DAC. Also, since these values need to be very precise, these DACs are very expensive and have a limited resolution.

In addition to the already mentioned advantages, correction can be done by either switching a bit high, neutral, or even to a negative value, because a binary weighted DAC is bipolar per bit. Obtaining a bipolar output range is as easy as applying a negative voltage to the resistor bit.
The main disadvantage of the binary weighted DAC is the need for an operational amplifier. The network is then no longer fully static, which increases the settling time, lowering the maximum sample frequency. Another disadvantage is that the non-linearity, gain error, and offset error of the op-amp will disturb the output of the total system. Also high precision resistors are needed which are expensive. A solution is to manufacture precise resistors in integrated circuits, since this is relatively cheap. The problem of this architecture is the need for various resistor values, which again makes manufacturing difficult.

R2R DAC

The R2R DAC is the second variant of resistive DACs. It consists of two resistor values throughout the circuit as the name suggests: R and 2R. The absolute value of the resistors do not matter in theory, as long as the resistance is twice as high [30]. Figure 3.6 illustrates an R2R circuit with N input bits. Bit 1 is the MSB, bit N the LSB, and both are driven from logic gates. The bits are switched between 0 V and voltage reference Vref, which represent a digital zero and one respectively. The resistive network causes the bits to be weighted relative to their contribution to the output voltage Vout [31].

A negative voltage output can be achieved by inverting the reference voltage. As stated before, the main problem of resistive DACs is their accuracy at higher bit resolutions. For example, for a 16-bit R2R DAC, the resistors for the MSB should have a maximum error of

$$\frac{\Delta R}{R} < \frac{1}{2^{16\,\text{bits}}} \approx 0.0015\,\%. \tag{3.5}$$

Therefore, 0.0015 % resistors are required and such accurate resistors are more expensive than conventional resistors. Manufacturing two different resistor values is also more complex than a resistive DAC were only one value is needed. However, R2R DACs have low noise and high accuracy and this architecture can achieve high-voltage outputs [33].

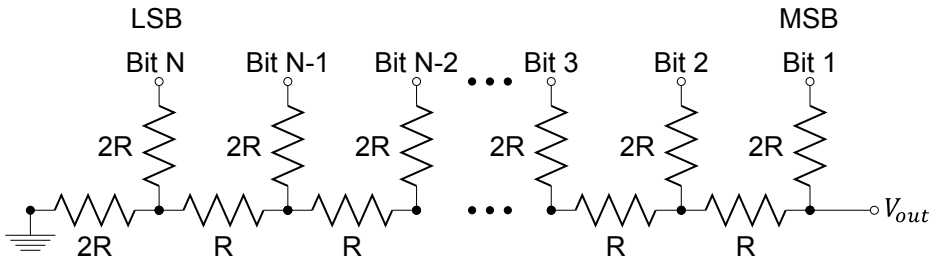


Figure 3.6: Architecture of a R2R DAC with N bits [32].

Resistive-String DAC

A third implementation of a resistive DAC is the Resistive-string (R-string) DAC. The R-string DAC requires a large amount of switches compared to the binary weighted DAC and R2R DAC. Figure 3.7 shows the architecture of a 3-bit R-string DAC to get an idea of the size of such an implementation. An N-bit version would theoretically consist of 2N resistors with equal values in series and 2N switches. As shown in the figure, each bit represents one of the vertical arrays of switches, were a switch is connected between each node of the chain between the reference voltage Vref and ground. The output is taken from the appropriate node by closing the correct switches. This type of DAC is therefore generally used in integrated circuits, because integrated circuits can easily contain large amounts of switches and resistors.

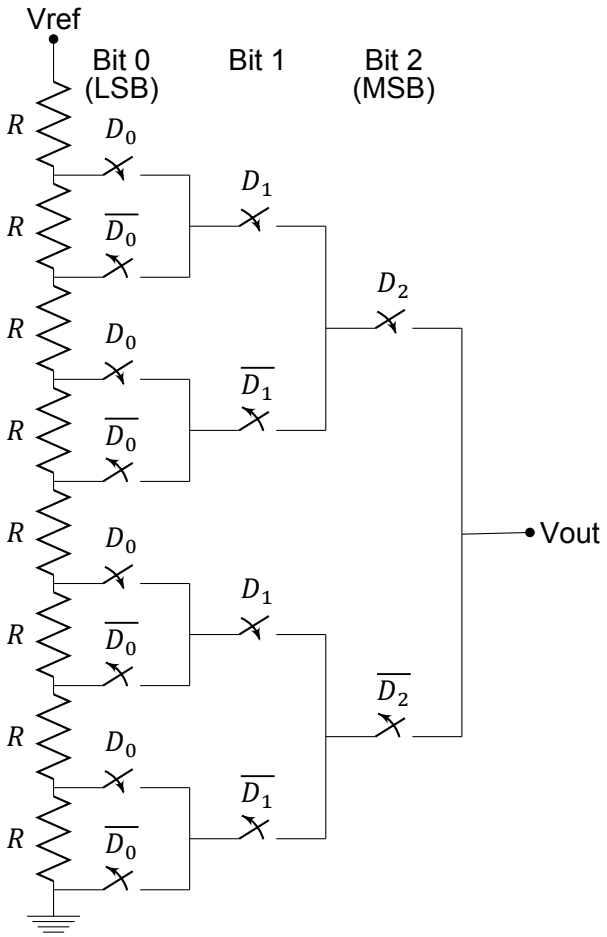


Figure 3.7: Architecture of a 3-bit R-string DAC.

One of the advantages of a R-string DAC is that due to the production process, the resistors can be trimmed to obtain near exact values, preventing the use of external accurate and expensive resistors. The low power draw and small die area make small packages possible and monotonicity is guaranteed [33]. However, these circuits are complex and the switches have to be controlled properly, which is prone to errors.

3.4.4. Concluding Remarks

This section discussed many implementations of DACs, each with their own advantages and disadvantages. One of the main requirements for the final product is that the components used should be widely available. Even though some of the discussed implementations have many advantages, they may for example not be easily manufactured. Keeping the requirements in mind, the best option is the R-String DAC. Although it has disadvantages such as needing high precision resistors and a complex control system, all resistor values are the same, which prevents the use of external accurate and expensive resistors, and the architecture guarantees monotonicity and allows for a small IC package.

3.5. Power Regulation Subsystem

As stated before, the device has to be powered by the USB connection, which specifications were described in Section 3.2.3. The nominal voltage equals 5 V and can have a maximum deviation of ±5 %. Also, the current that can be drawn depends on the hub type, with a maximum of 500 mA.

Since the power that the USB connection delivers is not stable, a power regulation subsystem is needed. This section elaborates on the research done regarding this power subsystem. Since the components have not yet been chosen in this chapter, the exact requirements of each component are not yet known and therefore this section only discusses general research and pitfalls.

To achieve this power regulation subsystem, there are two approaches. Either design a power regulator system using basic components or use integrated circuits that will do the regulation. Designing a regulator is outside of the scope of this project and therefore readily available solutions will be used. In the following chapter, each subsystem will list the required current and voltage levels for the to-be-designed power regulation subsystem. With those specifications in mind, the final subsystem can be designed.

3.5.1. Power Regulators

Linear Converter

The most basic way to change a voltage is by using a voltage divider, which is nothing more than two resistors in series. The input voltage will divide itself over these two resistors. For example, if the resistors are of equal value, the voltage at the node between the two resistors will be equal to half of the input voltage. The upside of this converter is that the architecture is very simple, since it consists of just two resistors. A downside is that only voltages lower than the supply voltage can be obtained and the voltage stability is not improved with this converter. Additionally, to make an accurate voltage value, accurate resistors are needed, which are expensive.

Operational Amplifier

Another suitable solution for power conversion is an operational amplifier, or op-amp. An op-amp can increase or decrease its input voltage level with a predetermined gain. The upside of an op-amp is that it is a simple implementation and that it can achieve higher voltage levels than its input voltage. However, an op-amp can only increase its input voltage level until it matches its supply voltage level. Since both the input voltage and the supply voltage will be that of the USB connection, only equal or lower output voltage levels can be achieved. Another downside of an op-amp is that it suffers from nonlinearities in its gain. This means that the gain can deviate of what it should be, causing the DC output level to be too high or too low.

Switched-mode Converter

Switched-mode converters can convert voltages or currents to a higher or lower value. Examples of this type of converter are buck, boost, and buck-boost converters. To do the conversion, they use an inductor, a diode and a switch. Each converter has a different configuration using these three components. To perform the required conversion, the switch closes and opens with a specific frequency. This switching causes a ripple on the output. This ripple originates from the fact that energy is stored and released from the inductor and causes an *Alternating Current* (AC) on the DC output. The upside of this type of converter is that it is the only one in this section that is capable to obtain higher voltage levels than its supply voltage. The downside is that it is much more difficult to implement: it requires more components, including a switch. This switch has to be controlled on a certain frequency.

3.5.2. Decoupling power signals

The power originates from the USB connection, from which it will be regulated by this subsystem. The proper current and voltage levels will subsequently supply the other subsystems with power. All this transporting of power means that there will be many power lines running through the design.

These power lines can couple to distortion caused by other components, which causes them to have an AC component or ripple on the signal. This ripple will then enter the components through the power lines and can affect their outputs or otherwise harm the IC. Especially the analog outputs are very sensitive to this.

To suppress distortion, decoupling capacitors will be used, which act as a low-pass filter to filter out AC components. They are placed close to the power input of a component. After the filter, the path length should be as short as possible to avoid additional error coupling.

Another reason to use decoupling capacitors is to avoid sudden drops or rises in the supply voltage. When a supplied IC switches some output from high to low or vice versa, the power supply quickly rises or drops. It takes time for the power supply to return to its nominal value. A capacitor stores charge and acts as a small supply source. If the supply voltage drops, the capacitor voltage will drop much slower, because of the capacitor discharging. The same goes for the supply voltage rising, the capacitor voltage will rise slower, because the capacitor is then charging. Both the frequency characteristic and the charging problem need to be taken care of. To do this, each will require its own capacitor. For the charging problem, a large electrolytic capacitor is used because it stores charge very well. However, these electrolytic capacitors have poor characteristics regarding AC filtering. Therefore, a smaller ceramic capacitor is used which has good AC filtering characteristics [34].

3.5.3. Concluding Remarks

To supply all subsystems of the final implementation with power, a power regulation subsystem is needed. This power regulation subsystem should deliver different current and voltage levels and increased stability. It will consist of readily available components, because designing a power regulator from the ground up is outside the scope of this project.

Three different kinds of power regulators have been discussed. If the design needs voltage levels that are higher than that of the USB connection, the switch-mode converter would be a good choice. The downsides of this converter are that it is more complex to implement and that its output voltage level is not stable because of a ripple caused by the switching. Therefore in the case of only lower voltage levels, the other two implementations can be more practical, being the linear converter and the operational amplifier. Their implementation is not a real problem, since already available components will be used.

Furthermore, decoupling capacitors will be used to ensure that there is a more stable voltage supply and that AC distortion components are suppressed.

4

Implementation

The previous chapter discussed various theoretical implementations and solutions to problems imposed by the requirements. At the end of each subsystem, a small conclusion was presented in which a certain direction or implementation was chosen. These implementations will be extensively elaborated in this chapter with the content narrowed down to this project's final selection of components. The subsystems of this chapter are discussed in the same order as in the previous chapter, except for the USB and the controller sections, which will be combined into one section.

4.1. USB Controller Subsystem

In the previous chapter the decision was made to use the USB 2.0 protocol for this project. Therefore it is necessary that the requirements of the final product are compared with the limitations of the USB 2.0 protocol and that this protocol is optimally set for our product. Next a controller has to be chosen which is compatible with these specifications and the type of data sent. For both subsystems, a component has been found which combines these functionalities into one unit, from now on referred to as the USB controller. The working of this USB controller will be explained in detail and finally its schematic will be discussed.

4.1.1. Requirements

The requirements as requested by Flexible Optical B.V. can be found in Table 2.1. The challenge in the USB controller lies in transferring the data for 40 channels simultaneously at a high refresh rate and moreover supplying power with a low ripple without degradating any of the other requirements. As discussed in Section 3.2 USB 2.0 will be used but therefore the USB subsystem proves to be a bottleneck for the refresh rate since the maximum refresh rate it can achieve is 8 kHz using high speed. Since the required refresh rate for this project has been modified to match this frequency of 8 kHz achieving this frequency should be possible. The preferred transfer mode was determined to be either bulk or interrupt. Furthermore the USB controller should convert the incoming USB data to the SPI protocol and have GPIO pins to be able to control functions of the DAC or DACs depending on the final implementation, which was concluded in Section 3.3.

4.1.2. Choice of Component

After an extensive search, a suitable USB controller was found: the FT232H [35] from FTDI. This controller complies with the requested demands, since it uses USB 2.0, bulk transfer, supports conversion to the SPI protocol and has GPIO pins. Another advantage of using this USB controller is that an evaluation board including this controller was also available, namely the UM232H [36]. This evaluation board contains the FT232H, the necessary decoupling capacitors, a crystal oscillator and an *Electrically Erasable Programmable Read-Only Memory* EEPROM. This EEPROM is required to facilitate slight adjustments to timings of several USB signals, because these timings are not only dependant

on the USB signal drives, but also by factors such as the PCB layout and external components. An overview of the module is shown in Appendix C.1. On this module, the FT232H is shown along with numerous capacitors and the EEPROM is soldered to the bottom of this module. The chosen EEPROM is the 93LC56B from Microchip [37]. In the following subsection the most important aspects of the USB controller will be discussed.

4.1.3. Architecture

Input power and data

The UM232H has been designed to be compatible with USB. It contains a USB connection to a host system via a mini-B USB connector. Please refer to Section 4.1.4 for a detailed overview of how these pins are connected to the controller.

The UM232H has two modes for power: self-powered and bus powered. In self-powered mode the device gets its power from its own supply which is then connected to an external power source, while in bus-powered mode the device gets its power from the USB bus. Since the USB DAC in this thesis needs to be powered by the USB port and does not have access to an external power source, the bus-powered mode is chosen. In this mode the the 5 V is directly connected to the controller. This 5 V however is not stable enough for connection with the core of the controller. Therefore in the controller this rather unstable power supply of USB is corrected for and converted to a 3.3 V and 1.8 V stable voltage by an internal voltage regulator which is suitable for connection with the core of FT232H. See Table 4.1 for all power related signals of the FT232H.

Table 4.1: USB controller power input characteristics.

Component	Parameter	Input/Output	Nominal voltage
93LC56B	V_{CC}	Input	1.8 V to 5.5 V
FT232H	V_{CCA}	Output	1.8 V
FT232H	V_{CCD}	Output	3.3 V
FT232H	V_{CCIO}	Input	3.3 V
FT232H	V_{CORE}	Output	1.8 V
FT232H	V_{PHY}	Input	3.3 V
FT232H	V_{PLL}	Input	3.3 V
FT232H	V_{REGIN}	Input	3.6 V to 5.5 V

V_{CCA} and V_{CORE} are outputs. The 3.3 V inputs can be supplied by V_{CCD}, the 3.3 V output. Also V_{CC}, the EEPROM supply voltage, is supplied by this 3.3 V voltage. This leaves only one voltage to be supplied by the power subsystem, which is shown in Table 4.2.

Table 4.2: USB controller power input characteristics.

Component	Parameter	Nominal voltage	Deviation	Deliverable Current
FT232H	V_{REGIN}	3.6 V to 5.5 V	±0.25 V	54 mA

On the evaluation board the data is received by the differential data pair from USB and directly connected to the controller which supports this format. The controller then handles the data and converts it to a protocol corresponding to its configuration mode, which will be discussed now.

Configuration Mode

The USB controller has two buses: one of eight pins and one of ten pins for a total of 18 pins. These buses are called the ADBUS and the ACBUS respectively. On each bus all pins are configured accordingly to which configuration mode is chosen. Choosing this configuration is important since that determines the operation of the entire controller. There are nine different configuration modes [36]. A configuration mode had to be selected which supports SPI, the chosen serial protocol for the output signals, and which is able to configure GPIO pins.

Accordingly the *Multi-Protocol Synchronous Serial Engine* (MPSSE) configuration mode has been chosen. MPSSE is a mode that provides a flexible means of interfacing synchronous serial devices to a USB port, in our case using SPI. Using MPSSE the eighteen output pins of the USB controller will then be configured as follows: four pins for the SPI output, twelve GPIO pins and two other pins. The four SPI pins are configured to be on the channels ADBUS0 to ADBUS3. The twelve GPIO pins on the other hand are configured to be on the channels ADBUS4 to ADBUS 7 and ACBUS0 to ACBUS7. These four SPI and twelve GPIO pins can be easily controlled by software which will be discussed in Section 4.5. The other two pins, being ACBUS8 and ACBUS9, are present and can be connected to LEDs. ACBUS8 will give a high signal when the USB controller is successfully receiving power where ACBUS9 will give a high signal if the controller is sending/receiving data.

Latency

The latency of the USB controller can be divided into three parts, being the latency of USB 2.0, the USB controller itself and the speed of a GPIO pin switching. It was estimated in section 3.2 that USB has a latency of 125 µs. The latency of the USB controller can be set to 0 ms and is basically very small compared to the latency of the other parameters [35]. The latency of changing a GPIO signal from low to high is also considered to be very small since the fastest frequency a GPIO can be updated with is 22 MHz [38]. This leads to a GPIO latency of

$$T = \frac{1}{f} = \frac{1}{22\,\text{MHz}} = 45\,\text{ns} \tag{4.1}$$

which can be considered to be zero when compared to the other latency parameters like USB. Herefore the latency of the entire USB controller subsystem can be considered to be equal to the latency of USB 2.0, being 125 µs.

Refresh Rate

The refresh rate of the USB controller can be measured by capturing the packets which the computer sends to the device. This can be done by using the *Universal Serial Bus Packet capture* (USBPcap) extension in Wireshark [39]. This measurement has been done by sending data continuously to the USB controller for almost a minute and the summary of this is shown in Figure 4.1.

Traffic	Captured	Displayed	Displayed %	Marked	Marked %
Packets	395013	395013	100.000%	0	0.000%
Between first and last packet	51.041 sec				
Avg. packets/sec	7739.079				
Avg. packet size	509 bytes				
Bytes	201097709	201097709	100.000%	0	0.000%
Avg. bytes/sec	3939898.082				
Avg. MBit/sec	31.519				

Figure 4.1: Summary of the measurement of USB packets using USBPcap in Wireshark.

The two most important aspects regarding the refresh rate are the amount of packets and the time between the first and the last packet. These can be used to calculate the time for each packet:

$$\frac{51.041\,\text{s}}{395\,013\,\text{Packets}} = 129.21 * 10^{-6}\,\text{s} = 129.21\,\mu\text{s} = 7.739\,\text{kHz} \tag{4.2}$$

This indicates a that the time between intervals using this USBS controller is 129.21 µs. When inverted this implies a refresh rate of 7.739 kHz. Compared to the polling interval of 125 µs and the refresh rate of 8 kHz of USB 2.0 using high speed, it can be concluded that this estimation is correct. This because packets have to be resent when errors occur and so the real estimation takes slightly longer than the optimal situation.

4.1.4. Schematic

The datasheet of the UM232H [36] shows how to connect the FT232H. This will now be discussed in four parts. The schematic of the entire design is shown in Appendix C.3.

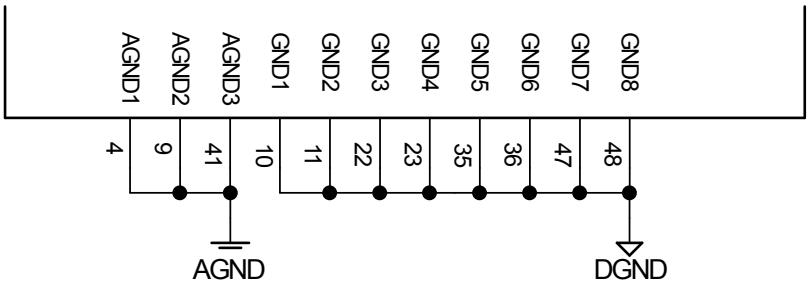


Figure 4.2: First part of the schematic of the FT232H showing the grounding.

The first part is shown in Figure 4.2 and shows all pins which are needed to be connected to ground. Most pins are connected to digital ground, but there are also some that require to be connected to analog ground. Grounding is extensively discussed in Section 4.6.7.

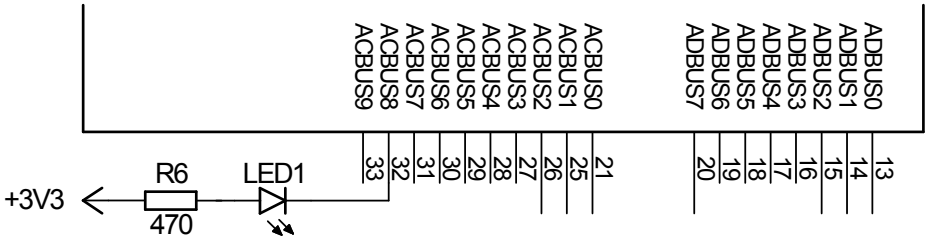


Figure 4.3: Second part of the schematic of the FT232H showing the output pins.

Figure 4.3 shows the second part of the connection schematic. It shows the output pins which are used to send data to the DAC and to control the DAC. Both the ACBUS and the ADBUS can be seen, some pins are extended to show that they are used. Also, the notification LED connected to ACBUS8 is shown. ACBUS9 is not connected since this was not required in the specified form factor.

The third part is shown in Figure 4.4. All grounding is done to digital ground, except for the TEST pin since this pin will not disturb the analog ground. Power lines are decoupled using capacitors as described in Section 3.5.2.

The V_{REGIN} is connected to the USB connection. Signals V_{CCORE} and V_{CCA} are signals that are used internally and are the result of the internal voltage regulator. They are decoupled externally.

The next two pins, D_P and D_M, are for the data of the USB connection. They are protected using *Electrostatic Discharge* (ESD) protection. The RESET pin is active low and is connected through a capacitor to the 3.3 V supply. Active low means that the device resets when the input voltage is low. The capacitor is a timing capacitor to form the internal reset pulse. This causes the device to reset on power up. After a short amount of time the reset input will become low, because the capacitor is then fully charged. REF provides a reference for the current and must be connected to ground through a 12 kΩ resistor.

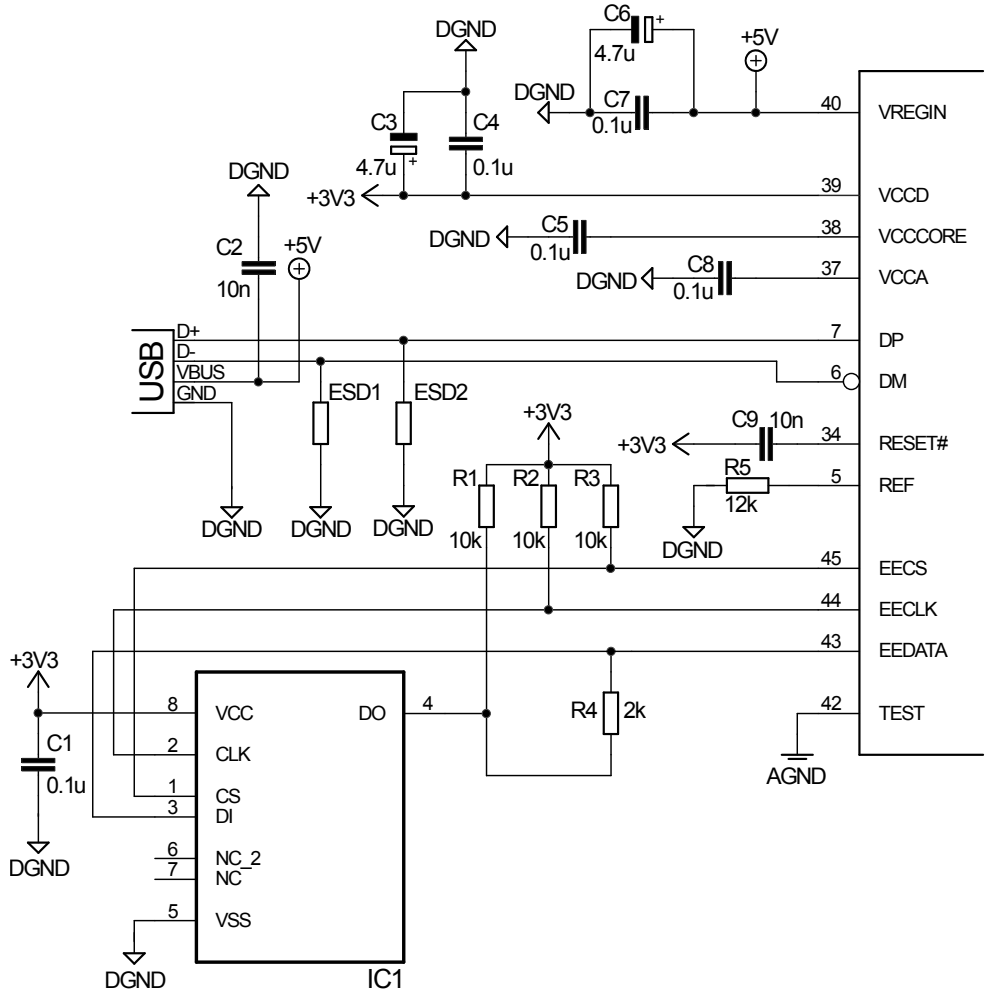


Figure 4.4: Third part of the schematic of the FT232H.

The next three pins, EECS, EECLK and EEDATA are for the connection to the EEPROM. These represent the chip select (CS), the clock (CLK) and the data (DO and DI) respectively. The power for the EEPROM is supplied by the 3.3 V output V_{CCD} of the FT232H. The CS, CLK and DO are connected to the 3.3 V supply through resistors. These are pull-down resistors which force the signal to become low when not driven. There is also a resistor between DO and DI which ensures that it is not possible to use them at the same time, because it forces a voltage difference between the two signals. Finally the TEST pin is connected to analog ground.

The fourth and final part is shown in Figure 4.5. V_{PHY} and V_{PLL} are precise analog power inputs, and V_{CCIO1}, V_{CCIO2}, and V_{CCIO3} are power inputs for internal use. These require a 3.3 V source, which can be delivered by the output V_{CCD} as discussed in the previous part. The datasheet recommends to place ferrite beads between the V_{PHY} and V_{PLL} power inputs and the power supply to prevent electromagnetic interference [36].

Pins XCSI and XCSO are connected to the crystal oscillator. This crystal oscillator is also connected to two capacitors, which are necessary for oscillation. The reason is that the crystal oscillator can be considered inductive, so together with the capacitances, these form a LC-network. Such a network has a resonating frequency, which is the frequency of the clock. However, true LC-networks do not exist in practice since there is always a resistance which dissipates energy making it an RLC-network. An internal amplifier in the FT232H is needed to correct for this dissipation.

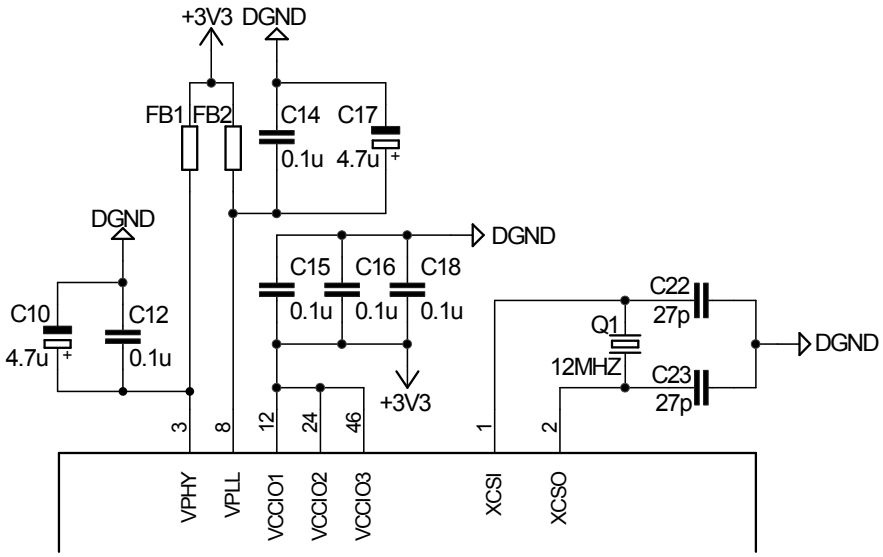


Figure 4.5: Fourth part of the schematic of the FT232H.

4.2. Digital-to-Analog Converter Subsystem

This section discusses the implementation of the DAC subsystem, as used in the final product. First of all, the choice of the component is discussed, after which the working principles will be elaborated, and the section will conclude with the way the DAC is controlled, its characteristics and a schematic.

4.2.1. Requirements

As concluded in Section 3.4.3, a R-string DAC seems to be the most feasible solution since integrated circuit solutions are easily available. Also, Flexible Optical B.V. demands a working prototype at the end of the project, which means that no exotic solutions can be experimented with or developed due to the length of the project, even though some solutions seem very promising. In Section 3.3.4, a single 40-channel DAC chip is preferred over multiple separate DACs, since a smaller control system, less pins, and a less powerful micro-controller are needed. The chosen chip will also need to comply to the stated requirements from Chapter 2. Possibly the most important question is if a single 40-channel DAC will be used, or multiple separate DACs with a total of 40-channels.

4.2.2. Choice of Component

While searching for R-string DACs, the company Analog Devices, Inc. came up as one of the leading companies in manufacturing DAC integrated circuits for instrumentation, control, and optical system applications. Our supervisor also spoke highly of this company, which narrowed down our search. After researching the product list of this company, the AD5370 seemed like the perfect solution with respect to our requirements. The main features of the AD5370 chip [40] are the following.

- 40-channel DAC.
- Simultaneous update of DAC outputs.
- Guaranteed monotonic to 16 bits.
- SPI, QSPI, MICROWIRE, and DSP interface standards.
- Maximum output voltage span of $4 \times V_{REF}$, with a maximum of 12 V.
- Nominal output voltage span of -4 V to 8 V.
- System calibration function allowing user-programmable offset and gain.

One of the main features of this chip is the 16-bit monotonic resolution, which operates linearly over its complete resolution. Another feature is the several on-board serial interfaces of the DAC, which include SPI. SPI was already chosen as interface in Section 3.3.4 due to its fast transfer speed and ease of implementation. The AD5370 matches very well to the given requirements. A comparison of the requirements versus the specifications of the 'next generation' DAC is shown in Table 4.3. The number of analog outputs, resolution and output mode all match one on one. Therefore the remaining four parameters will be examined.

Table 4.3: 'Next generation' requirements compared to the specifications of the AD5370.

Parameter	**'Next generation' DAC**	**AD5370**
Analog outputs	40 channels	40 channels
Refresh rate	8,000 frames per second	33,333 frames per second
Resolution	16 bits per channel	16 bits per channel
Output mode	Synchronous for all channels	Synchronous for all channels
Output range	0 V - 5 V or −5 V - 5 V with adjustable limits	$4xV_{REF}$ with programmable offset and gain
Power	Supplied by USB 2.0	Supplied by USB 2.0
Load capacitance	> 500 pF	≤ 2.200 pF

Refresh Rate

One of the most important parameters of the AD5370 is the refresh rate. The datasheet does not supply a value for this parameter. However, timing characteristics are available and using these an approximation was made. The refresh rate of the DAC is assumed to be the inverse of the output settling time, since this parameter holds for every update of the output signals and has the largest latency in the subsystem. Some of the other timing parameters are 1000 times smaller than the output rise time. In the worst case scenario, in which the maximum value of the output settling time equals 30 µs, the refresh rate equals

$$\frac{1}{30\,\mu s} \approx 33\,333\,\text{Hz} = 33\,333\,\text{frames per second}. \tag{4.3}$$

Since this approximation is 1.67 times larger than the requirement, if other timing characteristics are also taken into account the refresh rate would still be sufficient.

Output Range

The output range is dependent on the voltage V_{REF}, with a maximum range of 12 V. V_{REF} is the reference input and is referred to analog ground. Flexible Optical B.V. requires either an unipolar output range of 0 V - 5 V or bipolar range of −5 V - 5 V, but they prefer the second range, since it is less prone to errors. With the output range given, V_{REF} can be determined and, according to the datasheet, is equal to

$$4 * V_{REF} = V_{OUT}\ \text{range} \implies V_{REF} = \frac{10\,V}{4} = 2.5\,V. \tag{4.4}$$

The required output span of 10 V, centered around 0 V, allows for ample headroom for V_{DD} and V_{SS}. V_{DD} is the positive analog power supply and V_{SS} the negative analog power supply and both require a minimum headroom voltage of 1.4 V. This would result in the following requirements for the power regulation subsystem

$$V_{DD} \geq 5\,V + 1.4\,V = 6.4\,V \tag{4.5}$$

$$V_{SS} \geq -5\,V - 1.4\,V = -6.4\,V. \tag{4.6}$$

However, the datasheet specifies a range of 9 V to 16.5 V for V_{DD} and a range of −16.5 V to −8 V for V_{SS}. This means that the calculated values for V_{DD} and V_{SS} are both below the operating range of the AD5370. The refined power specifications for the power regulation subsystem are summarized in Table 4.4.

Table 4.4: Refined power specifications of the AD5370 [40].

Parameter	Voltage	Current
DV_{CC}	2.5 V to 5.5 V	2 mA
V_{REF}	2.5 V	60 nA
V_{DD}	9 V to 16.5 V	20 mA
V_{SS}	−16.5 V to −8 V	20 mA

Power

Since one of the requirements is that the complete system is powered by the USB connection, the power draw of the DAC is an important factor. Since USB 2.0 is used, the maximum power budget is

$$P = I * V = 500\,mA * 5\,V = 2500\,mW. \tag{4.7}$$

The typical unloaded power draw of the AD5370 is 280 mW [40] when V_{SS} = −8 V, V_{DD} = 9.5 V, and DV_{CC} = 2.5 V. Since higher supply voltages will be used, the losses will increase and so the power drawn. The drawn power of a loaded DAC is more interesting, but harder to determine since it depends on many factors. For simplicity, only the output load current will be taken into account, since this parameter is specified in the datasheet. The maximum output load current per channel equals 1 mA, which results in a total of 40 mA when all channels are updated simultaneously. Even if the power drawn by the DAC will increase drastically, the power that can be delivered by USB 2.0 should be adequate.

Load Capacitance

Before the signals will enter the deformable mirrors, the DAC will drive a high voltage amplifier, of which only the capacitive load is specified in the requirements, and should be larger than 500 pF. Due to the built-in output buffer, the AD5370 can drive capacitive loads up to 2.200 pF, which is far beyond the requirement. Depending on the headroom of the power regulation subsystem the drivable output capacitance may be lower if too much current is drawn by the load. Since the characteristics of the load are unknown, no estimation can be made regarding this requirement.

As shown in this section, the AD5370 is a good choice given the requirements and the scope of this project. For this design the choice was made to use SPI as the serial interface. Because of this, the following sections will only discuss SPI and not the other serial interfaces available on the AD5370.

4.2.3. Architecture

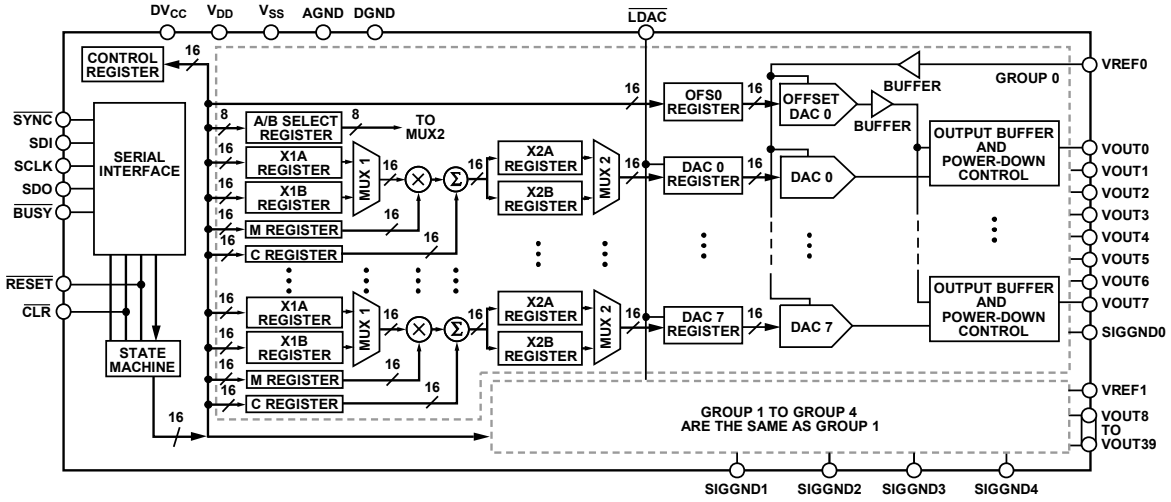


Figure 4.6: AD5370 block diagram [40].

The AD5370 has a total of 40 output channels, each with a resolution of 16 bits. A block diagram of the AD5370 chip is shown in Figure 4.6. As shown in the block diagram, the digital data to be converted is read by the SPI interface on the left side of the block diagram. It is then interpreted by the state machine, which determines the task to be carried out, for example: write an offset value, adjust the gain, or write data to a certain group and channel to be converted to an analog output signal. When data is received by the SPI interface, the state machine determines to which group and channel this data is sent. In total, the chip consists of five groups, each with eight buffered output channels. The many in- and output signals of the AD5370 will now be briefly explained.

- **$\overline{\text{RESET}}$** is an active low digital reset input as discussed in section 4.1.4, which properly initializes the registers. This should be done as soon as possible during startup.
- **$\overline{\text{BUSY}}$** is an active low output and shows if the DAC can be updated or not.
- **VOUTx** are the 40 DAC outputs, where x represents the corresponding channel number. These are individually buffered.
- **SIGGNDy** is the ground connection to which DAC group y is referenced.
- **VREFz** is the reference input which determines the output voltage range and is referred to as AGND. VREF0 determines group 0 and VREF1 the other four groups.
- **V_{DD}** is the positive analog power supply input.
- **V_{SS}** is the negative analog power supply input.
- **DGND** is the ground connection for all digital circuitry.
- **DV_{CC}** is the logic power supply input.
- **$\overline{\text{SYNC}}$** is an active low input and synchronizes the frame for the SPI interface.
- **SCLK** is the serial clock input for the SPI interface. Data is clocked on the falling edge of SCLK.
- **SDI** is the serial data input for the SPI interface. Data must be valid on the falling edge of SCLK.
- **SDO** is the serial data output for the SPI interface. SDO can for example be used for readback. Data is clocked on the rising edge of SCLK and is valid on the falling edge of SCLK.
- **AGND** is the ground connection for all analog circuitry.
- **$\overline{\text{LDAC}}$** is an active low input, which updates the DAC outputs when taken low.
- **$\overline{\text{CLR}}$** is an asynchronous active low clear input. When it is low, the input of each output buffer stage is switched to the potential on the corresponding SIGGND pin.

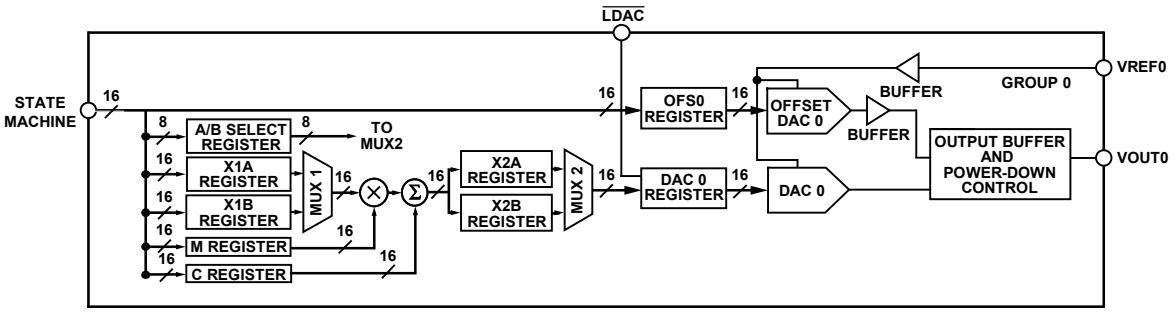


Figure 4.7: Block diagram of a single DAC channel of the AD5370 [40].

Each of the five DAC groups consist of the same set of sub-components. A block diagram of a single DAC channel is shown in Figure 4.7. Starting from the left hand side, the input of each channel is data from the state machine [40]. The state machine determines which register needs to be written. The actual DAC data-word can be written to either register X1A or X1B which depends on the control register. This control register controls MUX 1 and is shown in Figure 4.6. The contents of the control register also affects all channels. Each channel also has a gain and offset register, called the M and C register respectively. These registers allow for trimming gain and offset errors of the entire signal chain. Data from the X1A register is multiplied by the M register and summed by the C register. The calibrated data is then stored in the X2A register. The same holds for the X1B and X2B registers. Even though the figure shows a multiplier and adder for each channel, actually only one of both is present in the chip and is shared among all channels. This imposes restrictions on the update speed when several channels are updated at once. Every time new data is written to the X1A, X1B, M or C register, the data for either register X2A or X2B is recalculated and updated automatically. As shown in Figure 4.7, register X2A and X2B of all channels in a single group are controlled by the A/B select register. MUX 2 sends the data to the DAC register, which will send the data to the DAC by taking the previously discussed control signal $\overline{\text{LDAC}}$ low. Finally, the data is routed to a 16 bit resistive-string DAC, followed by an output buffer. As discussed in Section 3.4.3, an R-string DAC is a string of resistors with equal value, from V_{REF} to analog ground. This architecture guarantees DAC monotonicity, since the resistors can be trimmed to near exact values due to the manufacturing process, as mentioned in Section 3.4.3. In addition to the M and C registers to trim the gain and offset of each channel, there are also two 14 bit offset channels. One is for group 0 and the other is for group 1 to 4. These registers allow for an offset within a predefined range. It is for example possible to set the output range to be symmetrical or asymmetrical around 0 V. While the AD5370 is factory calibrated, since a bipolar output range centered around 0 V is required it will operate outside of its optimal calibrated state. Chapter 5 discusses the need, or not, for re-calibration of the AD5370.

4.2.4. Controlling the DAC

This section explains how the DAC is controlled, from a hardware point of view. First of all the write and read procedures of SPI are discussed after which adjustments to the output buffer are discussed to achieve a bipolar range of −5 V to 5 V. The setup procedure of the DAC is however not discussed here, but in Section 4.5.

SPI Write Procedure

The AD5370 contains a SPI interface which can handle clock frequencies up to 50 MHz for write operations and 20 MHz for read operations. To minimize power consumption and on-chip digital noise, the SPI interface only powers up fully on the falling edge of $\overline{\text{SYNC}}$, when data is written to the interface. Recall that SPI with a single slave uses four pins: SCLK, SDI, SDO, and SYNC. When data is written to the DAC using SPI, SDO is not used since this is for readback.
All registers except for the X2A, X2B, and DAC registers are directly accessible when using SPI. The X2A and X2B registeres are updated when new data is written to the X1A, X1B, M, or C register and the DAC registers are updated by $\overline{\text{LDAC}}$. The serial word which has to be sent to the DAC is 24 bits long, of which two bits are mode bits, six bits are address bits and the last 16 bits are data bits, which determine what is done with the data.
Figure 4.8 shows a timing diagram of the SPI write operation. The complete timing diagram can be found in Appendix C.2 and C.1. When data is applied to SDI, the write cycle will start on the falling edge of $\overline{\text{SYNC}}$. At least 24 falling clock edges must pass before all 24 bits of data are transfered and $\overline{\text{SYNC}}$ can be taken high. The addressed input register is updated on the rising edge of $\overline{\text{SYNC}}$. The next data transfer can take place when $\overline{\text{SYNC}}$ is low again. To update all output channels simultaneously, $\overline{\text{LDAC}}$ is taken low, after which VOUT on each channel updates.

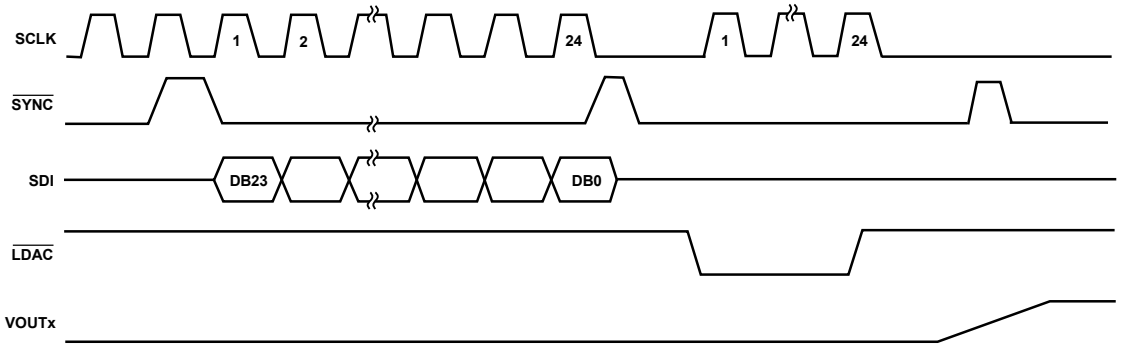


Figure 4.8: SPI write timing diagram of the AD5370 [40].

SPI Read Procedure

All registers, except the X2A, X2B, and DAC registers, can be read back directly from the SPI interface. Figure 4.9 shows a timing diagram of the SPI read operation and the complete timing diagram can be found in Appendix C.2 and C.1. When a readback command is received, data from the selected register is clocked out during the next SPI operation to the SDO pin.

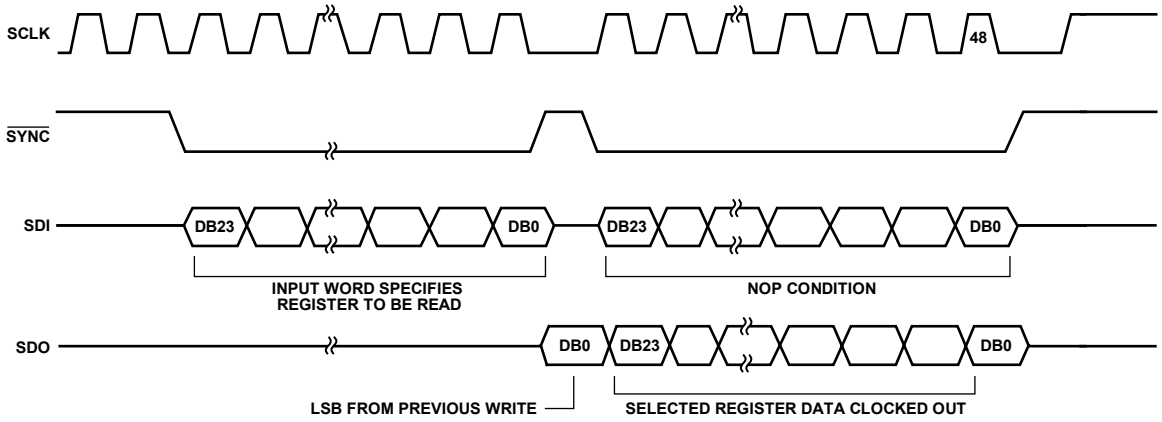


Figure 4.9: SPI read timing diagram of the AD5370 [40].

Adjusting Output Buffer

As mentioned earlier, there are also two 14 bit offset channels present. These registers allow for an offset within a predefined range, where register OFS0 determines the offset of group 0 and register OFS1 determines the offset of group 1 until 4. The required output voltage span ranges from −5 V to 5 V, while the nominal output voltage span ranges from −4 V to 8 V. Subject to the limitations of available headroom, it is possible to set the output symmetrical around 0 V by changing the values stored in both OFS0 and OFS1. Adjusting the output range results in an extra offset due to the gain error of the offset DAC. The amount of offset depends on the reference and how much the offset DAC deviates from its default value. The worst-case offset occurs at positive or negative full scale, but can be removed by using the C register.

4.2.5. Characteristics

Accuracy

The accuracy of the DAC is a very important aspect and is determined by the LSB. This DAC has a 16 bits resolution and an operating voltage range of 10 V. As calculated in Section 2.2.1, theoretically one LSB equals 152 µV. The datasheet of the AD5370 mentions that the integral nonlinearity equals ±4 LSB and the differential nonlinearity equals ±1 LSB.
The *integral nonlinearity* (INL) is the maximum deviation from the theoretical linear case over the entire range. In reality, the DAC is not linear and its output value can deviate at most 4 LSB from the theoretical value. The *differential nonlineairity* (DNL) is the maximum deviation caused by a single step size. The stepsize can deviate at most 1 LSB to the theoretical stepsize, which is of course 1 LSB. Figure 4.10 theoretically shows the DNL and INL. Δ is the ideal difference between two succesive steps, while Δact is the actual difference. DNL is equal to the difference between these two. INL is also shown and is the maximum deviation between the ideal and actual value.
The deviation at which an error occurs can be calculated. The theoretical linear case is assumed. An error means that the output value is ±1 LSB off. This error is a result of a voltage deviation coming from the power supply or voltage reference. It occurs when the deviation is one half of an LSB, which equals 76 µV. Since the maximum deviation from the theoretical linear case is 4 LSB, this reduces the effective resolution of the DAC to 14 bits.

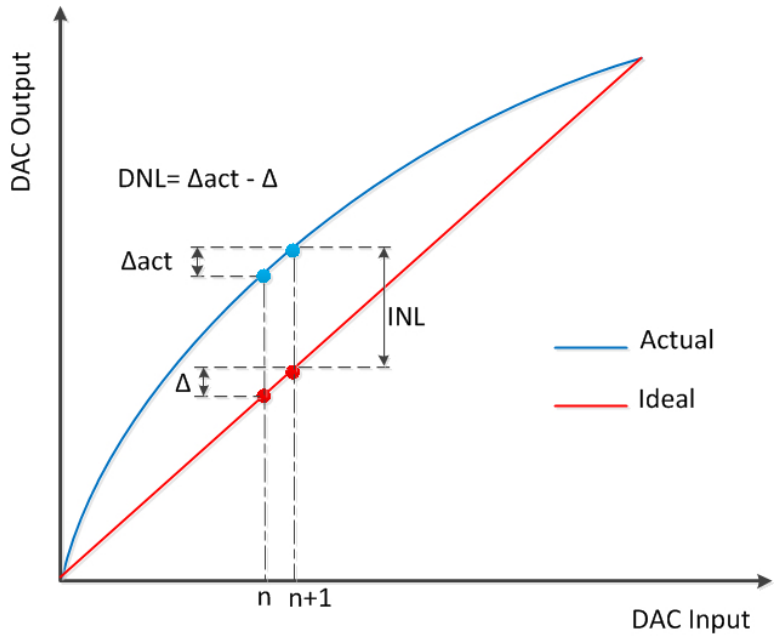


Figure 4.10: Figure illustrating the integral nonlinearity and differential nonlineairity [41].

The DAC can be calibrated to correct for induced erros. Two parameters are needed to do this, which are the zero-scale and full-scale error. A zero-scale error is the error in the output voltage when only zeros are loaded into the DAC register and is a measure between the actual VOUT and ideal VOUT. A full-scale error is the error in the output voltage when only ones are loaded into the DAC register and is also a measure between the actual VOUT and ideal VOUT. It does not include the zero-scale error. Then there is the gain error, which is expressed as the following.

$$\text{Gain error} = \text{Full-scale error} - \text{Zero-scale error} \tag{4.8}$$

The zero-scale error can be corrected by adding the amount of LSBs to the default value of the offset, or C register and the full-scale error can be corrected by subtracting it from the default value of the gain, or M register.

Latency

As shown in Figure 4.7, the longest data path is as follows: SPI interface, state machine, X1A or X1B register, X2A or X2B register, DAC register, DAC, and finally the output buffer. The actual latency of the DAC is hard to determine, but an approximation is made. No data can be found regarding the latency of the state machine or the output buffer, however data was available on the rest. The SPI interface runs at 30 MHz and this implies a clock period equals 0.033 µs. 24 pulses are needed to transmit all data, which results in a total tranmission time of 0.8 µs. Loading X1A, X1B, C, or M to all channels must be done sequentially, since only one multiplier is available and shared among all channels, and takes a maximum of 24.9 µs. The X2A and X2B registers are calculated each time new data is written to the X1A, X1B, C, or M registers. This calculation takes approximately 1.5 µs. The DAC register is neglected, since $\overline{\text{LDAC}}$ has to be taken low for a minimum of 10 ns. Finally, the output rise time is 30 µs, which result in a total latency of 57.2 µs.

4.2.6. Schematic

This section discusses the schematic of the DAC, which is shown in Figure 4.11. It follows from the DAC and the DAC evaluation board datasheets. The schematic of the entire design is shown in Appendix C.3.

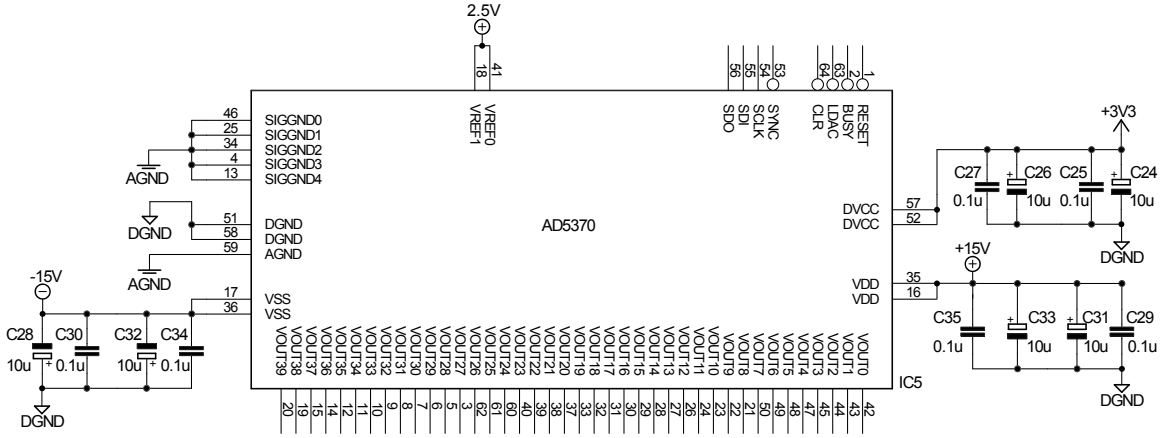


Figure 4.11: Schematic of the DAC connection diagram.

The bottom side of the schematic shows all forty output pins. These are connected to the two output connectors. On the upper left, the grounding is shown. As discussed before, each group has its own signal ground which is connected to analog ground. This provides the reference for the R-string DAC. Additionally, there are some digital grounding pins for the other subsystems, which were shown in Figure 4.6 and 4.7. More about grounding is discussed in Section 4.6.7.

The left and right side also show the three power input pairs. All power supply pins are decoupled as was discussed in Section 3.5.2. Finally, the top side shows the reference voltage input and the pins that are used to communicate with the USB controller. RESET, $\overline{\text{LDAC}}$, CLR and $\overline{\text{SYNC}}$ are connected to GPIO pins of the USB controller, while SCLK, SDI, and SDO are connected to the SPI pins of the USB controller.

4.3. Power Regulation Subsystem

The need for a power regulation subsystem was already addressed in Section 3.5. At this point all the components that will be used in the implementation are known, namely those that were discussed in the previous sections. Every component has its own current and voltage requirements and these will be discussed in this section.

4.3.1. Requirements

Table 4.5 shows all the requirements from the DAC and USB controller datasheets for the power signals that need to be supplied to the components of the system. While the required voltage follows directly from the datasheet the maximum deviation in the table does not and will be elaborated on in this subsection. The current drawn is also an important factor since the device will have to be powered using the USB connection and the current of this connection has to be enough for the entire device. The value for this current is given for a supply voltage of 9 V. The power regulation subsystem will be designed according to these requirements.

Table 4.5: Requirements from the datasheets [40] [35].

Component	Parameter	Required voltage	Maximum deviation	Current drawn
AD5370	DV_{CC}	2.5 V to 5.5 V	±194 mV	2 mA
AD5370	V_{DD}	9 V to 16.5 V	±194 mV	20 mA
AD5370	V_{REF}	2.45 V to 2.55 V	±19 µV	60 nA
AD5370	V_{SS}	−16.5 V to −8 V	±194 mV	20 mA
FT232H	V_{REGIN}	3.6 V to 5.5 V	±0.25 V	54 mA

DV_{CC}, V_{DD}, V_{REF}, and V_{SS} are voltages that will directly affect the analog outputs while V_{REGIN} will not. Therefore unlike the other signals V_{REGIN} can be connected directly to the USB supply voltage. Because of this it has the same voltage and deviation as the USB connection. Please refer to Table 4.5 to see that V_{REF} is the signal that has the most impact on the analog outputs since it has the lowest allowed deviation.

The reasoning behind the maximum deviation is as follows. One LSB equals 152 µV as was calculated in Section 2.2.1. An error can occur if the deviation in the analog outputs is greater than half of the value that one LSB represents. This means that the deviation caused by the power inputs should be at most 76 µV. For simplicity this value is used in the following two calculations.

The first calculation regards V_{REF}. The range of the analog outputs is determined to be four times V_{REF}. This means that a deviation in V_{REF} alters the range times four and consequently, the maximum deviation in V_{REF} should therefore be equal to 76 µV over 4 = 19 µV. This is also equal to one eighth of an LSB.

The second calculation regards the signals DV_{CC}, V_{DD}, and V_{SS}. They influence the analog outputs according to

$$\frac{\Delta\text{full scale}}{\Delta\text{input voltage}} = \text{gain}. \tag{4.9}$$

The gain is given in dB in the datasheet [40] and Δ full scale is one half of an LSB, which equals 76 µV. Taking all three signals into account gives

$$\Delta\text{full scale} = \Delta DV_{CC} * G_1 + \Delta V_{DD} * G_2 + \Delta V_{SS} * G_3 \tag{4.10}$$

where G is the gain. V_{DD} and V_{SS} will most likely be supplied by the same component. For simplicity we take the deviation of all three as equal. This gives

$$\Delta\text{full scale} = \Delta V * (G_1 + G_2 + G_3), \tag{4.11}$$

where ΔV is the deviation of either DV_{CC}, V_{DD}, or V_{SS}. Substituting for the gain values from the datasheet results in a maximum deviation of 194 mV for each signal.

In practice, the maximum deviations should be lower than the table shows. This is because both calculations use one half of an LSB, while actually the total error should be one half of an LSB.

The current drawn in Table 4.5 is for the unloaded system, meaning the connected system without any operations running and so for the minimum current drawn when connected. The total current drawn in the table adds up to 96 mA. This implies that the absolute minimum value that is required to be supplied by the USB connection is 96 mA, which can be easily supplied by it.

4.3.2. Choice of Components

With these requirements in mind, components had to be found. With the use of two ICs, almost all requirements can be met. One of them supplies the voltage for V_{DD} and V_{SS}. This is the Murata Manufacturing Co. NTA0515MC, which has an efficiency of 78 %. The other supplies the reference voltage V_{REF}. This is the Analog Devices Inc. ADR431B. These two components will be the main outline of this section.

That means that there are still two voltages that are unaccounted for. These are DV_{CC} and V_{REGIN}. These voltages do not need a special component to supply them, because they can be supplied by existing parts. As stated earlier V_{REGIN} can simply be connected to the USB supply voltage. DV_{CC} can be supplied by the 3.3 V output of the USB controller, since this is the required voltage and a sufficient current can also be delivered. Table 4.6 shows all information about the two ICs for the regulation system.

Table 4.6: Voltage specifications in the design [35] [42].

Component	Parameter	Nominal voltage	Deviation	Current
ADR431B	V_{REF}	2.5 V	±1.75 µV	30 mA
FT232H	DV_{CC}	3.3 V	±30 mV	16 mA
NTA0515MC	V_{DD}	15 V	±55 mV	33 mA
NTA0515MC	V_{SS}	−15 V	±55 mV	33 mA
USB connection	V_{REGIN}	5 V	±0.25 V	500 mA

Comparing Tables 4.5 and 4.6, all requirements are accounted for. Note that the ADR431B can have an offset of ±1 mV and that this can be corrected by a register in the DAC.

Not only the output power of the regulation ICs is important. The required maximum input power also has to be taken into account, since it is powered by the USB connection. The NTA0515MC requires at most 265 mA and 4.5 V to 5.5 V, according to the datasheet. Using the efficiency however, we find that the current will only be 127 mA.

The ADR431B requires at most 15 mA and 4.5 V to 18 V. The total maximum current that has to be delivered by the USB connection is the current that goes to the NTA0515MC, the ADR431B and the FT232H. This adds up to a maximum of 328 mA. Using the earlier mentioned current obtained using the efficiency, we find a total of 199 mA. This amount can still be delivered by the USB connection.

4.3.3. Architecture

The ADR431 is a DC-DC converter. It does not use switching because that would not allow for high precision. Instead it uses an *eXtra implanted junction Field-Effect Transistor* (XFET). This consists of two *Junction Field-Effect Transistors* (JFETs). These JFETs have a different pinch-off voltage. The difference in pinch-off voltage can be amplified to provide a stable reference voltage. The schematic is shown in Figure 4.12.

The NTA0515MC is also a DC-DC converter. Since its output voltage is higher than its input voltage, it uses a boost converter for the conversion. The schematic of such a converter is shown in Figure 4.13. A boost converter uses a switch to store energy in the magnetic field of an inductor. This switch causes a higher output voltage but also a ripple in the output voltage, which makes it unstable. Because of this, it is necessary to place a low-pass filter at the output of this converter which improves the stability. The datasheet recommends a LC-filter for this. Since limited space is available on the PCB, the choice was made to only use capacitors to suppress as much of the filter noise as possible.

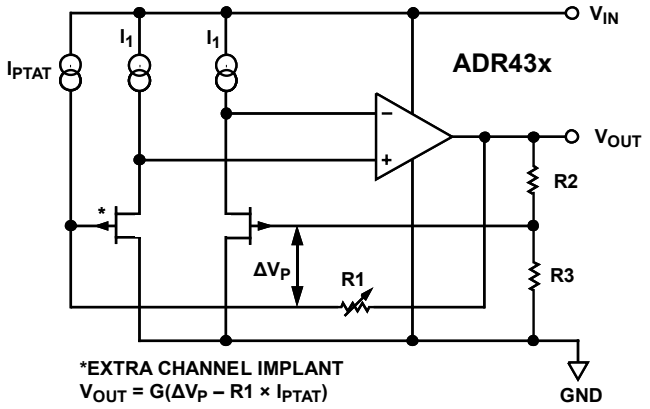


Figure 4.12: Schematic of the precision converter [43].

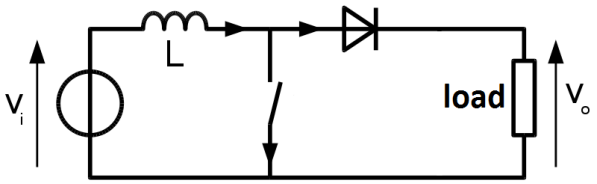


Figure 4.13: Schematic of a boost converter [44].

4.3.4. Schematics

Schematics of the actual implementation for both voltage regulators are shown in Figures 4.14 and 4.15, which are suggested by the datasheets [42] [43] of both components. The schematic of the entire design is shown in Appendix C.3.

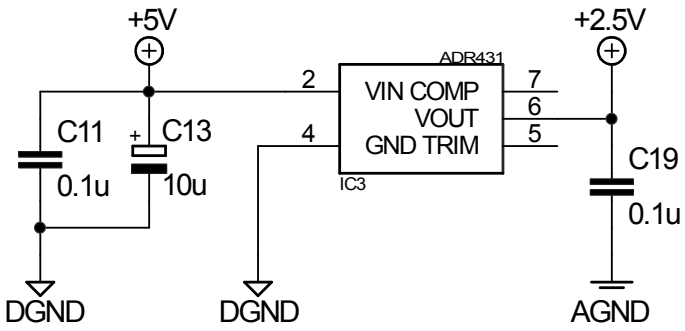


Figure 4.14: Schematic of the ADR431B.

The schematic of the ADR431B shows that it is supplied by the 5 V voltage source of the USB connection and that it has the requested 2.5 V output reference voltage. Both the voltage supply and the output voltage are decoupled. The input is grounded to digital ground to decrease the influence on the analog outputs. The output is grounded to analog ground, since it determines the range of the analog outputs and therefore has to be extremely accurate. If it was grounded to digital ground, it would be influenced by a lot of other components. Furthermore, COMP and TRIM are not connected. These two pins can be used to adjust the nominal value of the voltage. However, since the DAC can easily correct for an offset in the reference voltage using a register they are not used.

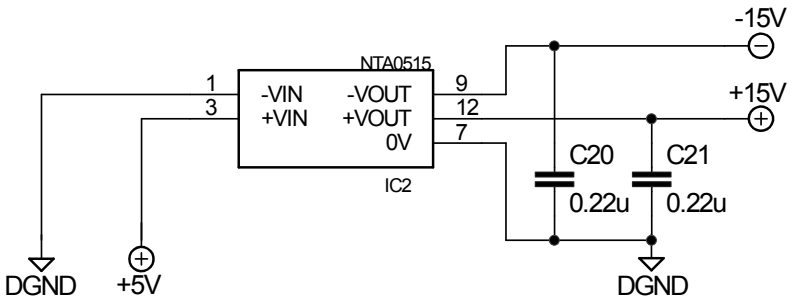


Figure 4.15: Schematic of the NTA0515MC.

The NTA0515MC is also grounded to digital ground, since it is a power component. The 5 V input voltage is supplied by the USB connection and the output voltages are 15 V and −15 V. As was mentioned before, the NTA0515MC works by switching which causes a ripple on the output voltage. Like the ADR431B therefore it also has decoupling capacitors.

4.4. Prototype

Before the design is turned into a more permanent version it needs to be tested as described in Section 4.6. This testing was done with the use of a breadboard. In the previous sections, all the components were chosen for the design. They are listed below.

1. USB controller and EEPROM: UM232H
2. Power Supply: NTA0515MC
3. Voltage reference: ADR431B
4. DAC: AD5370

The UM232H evaluation board used contains the FT232H and the 93LC56B. All these components were put on the breadboard and connected. Figure 4.17 shows the breadboard with all components connected.

First of all, verifications had to be made. Therefore all the voltage characteristics that were mentioned in the previous section, were measured. This was initially done using a multimeter, but later on an oscilloscope was used. The results of these measurements are shown and discussed in Chapter 5.

Furthermore, the communications between the USB controller and the DAC were monitored using a logic analyzer. This was needed to find possible errors while testing, see Figure 4.16. The SPI signals and GPIO signals were timed as described in the timing scheme in Section 4.2.4.

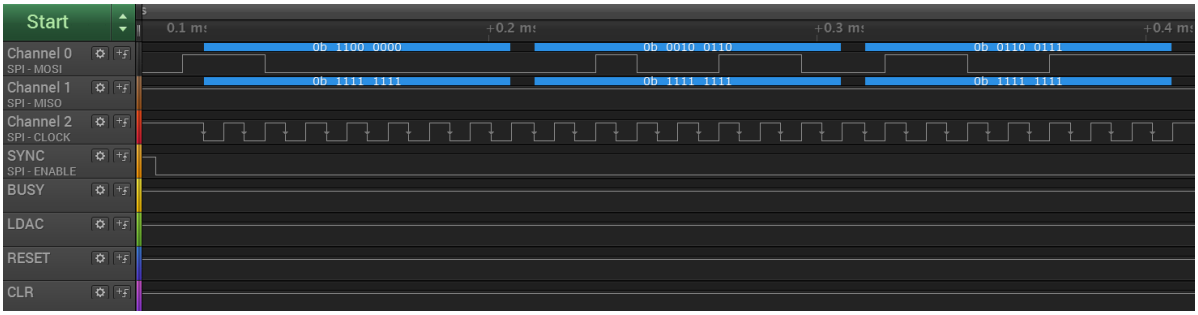


Figure 4.16: Example of SPI signals on logic analyzer.

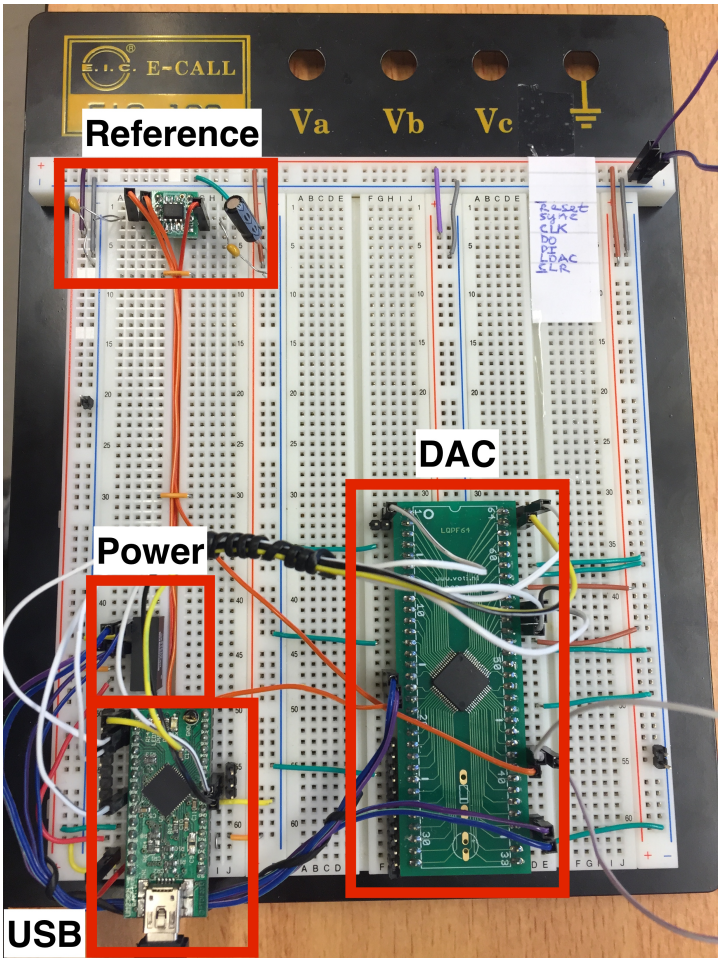


Figure 4.17: The prototype on a breadboard.

4.5. Software

To control the DAC suitable software is necessary. It runs on the computer and handles the USB controller via the USB port. That USB controller in turn controls the DAC. Therefore, to be able to control the entire system it is necessary to know how to communicate with this USB controller. A program has to be written for a computer to control the total system; this is the main outline of this chapter. It consists of a brief explanation of the MPSSE library and an overview of the program with detailed instructions which need to be sent to control the FT232H.

4.5.1. MPSSE

In Section 4.1 it was decided to use the MPSSE configuration mode. The MPSSE manual [45] contains details on how to send commands using the MPSSE library. The way these commands need to be sent is as follows. First an op-code in the form of a hexadecimal value of length two is sent, which in the binary domain corresponds to 8 bits or one byte. This op-code specifies the operation to be performed and the number of bytes that need to be sent until the function is finished and the program is ready to receive the next op-code. Next, if the op-code specifies that no more bytes need to be sent, the instruction is finished. If the op-code specifies that several bytes follow the controller will wait for these next data bytes. Once the instruction has been finished the controller is ready to receive a new op-code. A detailed description of how these op-codes are sent and which are sent in the software used in this design can be found in the next subsection.

4.5.2. Program overview

The functioning of the program can be represented in pseudocode. Refer to Algorithm 4.5.1 for this pseudocode.

Algorithm 4.5.1: DAC($value$)

comment: pseudocode for controlling the DAC

$connect$
$reset$
$set\_output\_offset(value)$
while 1
do $\begin{cases} \textbf{for } channel \leftarrow 0 \textbf{ to } 40 \\ \quad \textbf{do } set\_value(value) \\ cycleLDAC \end{cases}$

The first thing that happens is that the DAC connects and resets and that the offset is set to a certain value. These are functions that are necessary for initialization. Next an infinite while-loop is issued. In this while-loop every channel of the DAC is updated with the value that is passed to the algorithm. After every channel is assigned a value accordingly this is buffered to the output by the command *cycleLDAC*. How the various functions work and which commands need to be sent to execute these functions are discussed in the following subsections. Since the controller, which is controlled by the software, sends specific signals to the DAC the control signals that were discussed in Section 4.2 are used in the current section.

Connect

The program starts by letting the computer connect to the USB controller using several functions defined in the library of the FT232H. At the same time, to make sure the right settings are used, several commands are sent as soon as the device is connected. These are the following:

0x8A The controller has a mode to set a frequency value of 12 MHz to the master clock for backwards compatibility with older devices by dividing the clock by five. This is not used since a faster speed is preferable and therefore the op-code 0x8A is sent to the controller, which disables this clock division frequency. Consequently this results in a frequency of 60 MHz

0x97 Adaptive clocking is a feature which is required when using the JTAG interface on an ARM processor. This op-code is sent to disable this adaptive clocking.

0x8D Three phase clocking is a feature which is necessary when using protocols like I^2C. However in the case of using the SPI protocol it is not necessary and therefore this command disables three phase clocking.

0x86 **0xFF** **0xFF** The TCK/CK clock divisor is also set to a frequency of 60 MHz. This is done by sending three bytes: the first byte contains the op-code 0x86 that specifies that the TCK/CK clock divisor needs to be modified, and the following two bytes specify the value of this frequency. These two bits are both 0xFF since we want to set the TCK/CK clock divisor to its maximum possible value.

Reset

To reset the DAC two signals have to be reset: RESET and CLR. These signals are connected to the DAC on pins 1 and 64, and connected to the USB controller this corresponds to the GPIOs on ACBUS1 and ACBUS2. These values are changed with the following commands:

0x82 First the op-code 0x82 is sent which indicates that the ACBUS needs to be changed. The next
0xF9 command forces a value on each channel of this ACBUS. Since ACBUS1 and ACBUS2 need
0xFF to change, this corresponds to the sixth and the seventh bit of the byte. These two bits are set to zero and this results in the bitstream (a series of bits) of the ACBUS having the value 1111 1001; in hexadecimal representation this equals 0xF9. In the third byte the direction of each channel of the ACBUS is set, which can be either input with a zero or output with a one. Since the ACBUS is fully used as an output all these bits have to be equal to one, which in binary representation equals 1111 1111 and in hexadecimal representation 0xFF. Note that for this project the direction stream always has to be set to output mode and therefore it will not be mentioned in the future.

0x82 After the RESET and CLR have been set to zero, these must quickly revert back to their original
0xFF values of one or else the system will not function. The first command here is again to indicate
0xFF that the ACBUS needs to be changed, the second is to set the bitstream to 1111 1111 and the last command is again to set the direction of the ACBUS.

0x80 Lastly the ADBUS will be initialized. The command 0x80 is used to select the ADBUS, the next
0xFF command to set the bitstream to 1111 1111 and the final command to set the direction of all
0xFF these channels to output.

Set value

The next function that will be elaborated on is setting the values of the DAC itself using the SPI protocol, instead of just setting the GPIOs as in the previous cases. To do this the following commands are required.

0x10 First the op-code 0x10 is sent which indicates that data will be sent every clock cycle using SPI.
0x02 The next two commands then state the amount of bytes that need to be sent. Please refer to
0x00 Table 4.7 for an overview of several possible commands, their binary representation and the amount of bytes they indicate should be received. For example, when sending a bitstream full of zeros, the program will wait until it receives one byte before actually sending data to the DAC. When sending a bitstream full of ones, the program will wait for the maximum amount of bytes defined to be sent, which is 65536 bytes, before actually sending these bytes to the DAC. In this case the AD5370 is configured to receive a total of 24 bits at a time. This equals 3 bytes and therefore the two commands are which are used for the lower bitstream and the higher bitstream are 0x02 and 0x00 respectively. This means that the program will wait for three bytes (two plus one byte) to be sent to the USB controller after sending these to the DAC.

Table 4.7: Example of the working of the set value command.

Commands	Command in binary	Bytes to receive
0x10, 0x00, 0x00	0000 0000 0000 0000	1
0x10, 0xFF, 0xFF	1111 1111 1111 1111	65536
0x10, 0x02, 0x00	0000 0000 0000 0010	3

After these commands the DAC will wait for three bytes to be sent. These all have a specific purpose: the first byte contains the mode and the address and the second and the third byte contain the data. The mode consists of two bits, making it possible to send four possible combinations to the DAC which each correspond to their own mode. These are stated with their respective function in the following description.

Mode 11 Writes to the DAC input data (X) register, depending on the control register A/B bit. This register stores the actual values for the voltages at the output. The control register A/B bit will in the case of this implementation always be on register A. The reason for this being that the use of two registers adds functionality and buffering in other implementations but is not practical in this implementation.

Mode 10 Writes to the DAC offset (C) register. This register determines the offset of each channel and allows the output range of each DAC channel to be offset within a defined range.

Mode 01 Writes to the DAC gain (M) register. This register determines the gain of each channel and allows the output range of each DAC channel to have a certain gain within a defined range.

Mode 00 Special function, used in combination with other bits of the data-word. This mode is used for special utilities, for example to write to change the output offset, the control register or to read back from the DAC.

These two mode bits are the first two bits of the 24 bits that need to be sent to the AD5370. The next 6 bits specify the address where the data needs to be sent to. It can be configured to send data to all channels at the same time, to a group of channels or to each channel individually. Finally the last 16 bits are for the actual data which comes from the input. A function has been created for each mode. For mode 11, 10, 01 and 00 these functions are: *set_value*, *set_offset*, *set_gain* and *set_special* respectively. In the final code only the first and the last mode are necessary. Mode 11 is necessary since the value for every channel of course needs as a main aspect of the code. Mode 00 is necessary for special functions, where in the pseudocode shown in Algorithm 4.5.1 the special function *set_output_offset* is used. When using mode 00 instead of reading the next 6 bits as an address the DAC reads this as a special function code. In the case of the function set_output_offset these next 6 bits need to be the equal to the bitstreams 0000 10 and 0000 11 to select the registers OFS0 and OFS1 respectively. These functions are discussed in section 4.2.4 and, when both called, effectively write to the output registers of all channels and change the output range to the correct value being −5 V to 5 V as requested in Table 2.1.

Cycle LDAC

The last thing that needs to be done is to update all the outputs of the DAC simultaneously. This is done by setting the LDAC signal low. Therefore the following commands are sent.

0x82
0xFE
0xFF The first byte contains the command to make the controller write to the GPIOs on the ACBUS. The second byte forces the specific value on the ACBUS, in this case making the bitstream equal to 1111 1110. This last bit corresponds to the GPIO on ACBUS7 and makes the LDAC signal low. The third byte is again for the direction stream.

0x82
0xFF
0xFF This instruction is identical to the previous instruction except for one thing: it makes the bitstream equal to 1111 1111, making the LDAC signal high again and so making the DAC ready to be written again.

This concludes the discussion of each of the functions that are shown in Algorithm 4.5.1.

4.6. Printed Circuit Board

To finalize the implementation, a *Printed Circuit Board* (PCB) has been designed. It contains all components and connections integrated on one board. This section discusses how this was done and which challenges were faced.

4.6.1. Schematic

To design a PCB, a schematic which makes the connection of the components is required since it is impossible to create a PCB without knowing which components to use and how to interconnect them. This schematic was made with CadSoft EAGLE [46] software, which will be referred to as EAGLE from now on. The schematics in the previous sections were also made using EAGLE. The creation of the schematic itself in EAGLE is quite straight forward. With the use of libraries components can be selected, which are then represented with so-called symbols. When all the required components have been found, all that is left is to connect them in the correct way.

4.6.2. Components

Most of the earlier mentioned components can be found in libraries. Unfortunately, it can occur that the correct component can not be found in the default libraries of Eagle. In this case there are two approaches. The first approach is to search the internet for a custom library which includes the required component since most of the time the developer of the component also has an EAGLE library available. In the case that there is no such library available the other approach has to be taken: creating the component in EAGLE. This requires skill to do since no errors are allowed, otherwise the PCB will be worthless, where the biggest concern is the creation of the footprint.

4.6.3. Footprints

The footprint is the actual size of a component. This is what will be placed on the board. It is not shown in the schematic since the actual size of the component does not matter there. Before the footprint can be chosen or created, the component package has to be chosen since the package determines the size of the component. There are several different packages, they will be discussed in the next section.

The previous section mentioned that it can be necessary to create a component in EAGLE from scratch. The first thing to do is to create a symbol. This symbol will be shown in the schematic. It can have any size or shape, but will need all the connection pins that the component has. Next is the footprint which determines the size on the board. This has to be made using the package measurements described in the component's datasheet. It contains the size and placement of the soldering pads and also the size of the outline of the component [47]. To wrap it all up, the symbol and the footprint are connected. This is done by connecting the pins of the symbol to the corresponding pads in the footprint. The created component can then be used in the design.

An example of a footprint is shown in Figure 4.18. This is the footprint of the AD5370. The white rectangle represents the outline of the footprint, the red lines the soldering pads and the circle in the corner is used for tracking the correct placement of the unit.

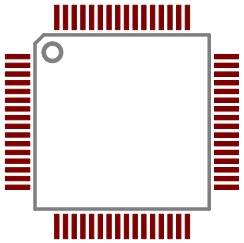


Figure 4.18: Footprint of the AD5370.

4.6.4. Component packages

As addressed before, the package of each component has to be chosen before the footprints can be selected. For ICs there is usually little choice in package since the datasheet gives only a few options. Components like capacitors and resistors have a larger variety available.

The first aspect that has to be considered is the attachment of the component to the board. Basically, there are two options. Either the pins of the component go through the board or they lie on the surface of the board. This last option is called a *surface mount device* (SMD). Advantages of using this method are a cheaper unit and easier adjustment of the components. Since this PCB will have many connections with relatively small components SMD packages have been chosen. An example of a SMD package can be seen in Figure 4.19, which is again the AD5370.

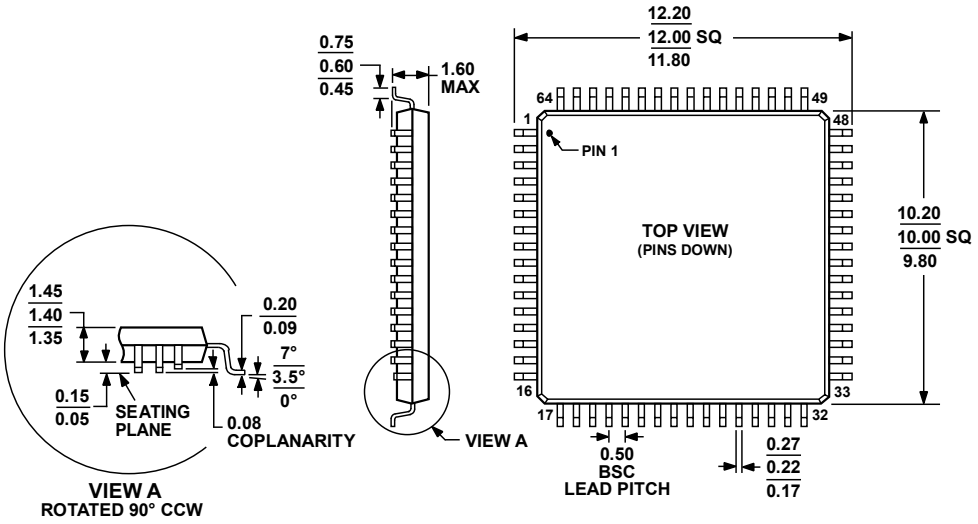


Figure 4.19: Package of the AD5370 [40].

The SMD packages for resistors and capacitors have a wide variety, which is shown in Table 4.8. Their size is noted using either imperial or metric codes. The difference between these two is simply the notation. The code is build up using the length as the first two numbers and the width as the last two numbers. Imperial uses inch and metric uses mm.

Table 4.8: The different imperial and metric codes [48] for SMD component packages.

Code		Length		Width	
Imperial	**Metric**	**inch**	**mm**	**inch**	**mm**
0201	0603	0.02	0.6	0.01	0.3
0402	1005	0.04	1.0	0.02	0.5
0603	1608	0.06	1.6	0.03	0.8
0805	2012	0.08	2.0	0.05	1.2
1206	3216	0.12	3.2	0.06	1.6
1210	3225	0.12	3.2	0.10	2.5
1218	3246	0.12	3.2	0.18	4.6
2010	5025	0.20	5.0	0.10	2.5
2512	6332	0.25	6.3	0.12	3.2

All the resistors and capacitors in the PCB design have imperial code 0805 or metric code 2012. This package is small, but not too small to solder them on the PCB by hand properly. Also, this is a very common size and all the required values are available.

4.6.5. Layout

At this point, the schematic is done and the footprints are chosen. Now is the time to determine the ideal location of the components on the board. To do this, there are several things to keep in mind.

First of all, the datasheets dictate that the decoupling capacitors for the different supply voltages of the DAC and the USB controller should be close to these ICs. Therefore these are placed in groups together.

Next the DAC analog outputs are very prone to errors. Some aspects to keep in mind to prevent this are as follows. They should be close to the output pins to minimize the path length. The use of vias

should be kept to a minimum to minimize distortion of the signal. Also the power signals should be isolated from the analog outputs to prevent distortion.

Finally the crystal oscillator should be close to the USB controller, because the clock is a signal that is very important and should not be distorted. Furthermore, the overall signal path should be kept as short as possible to prevent distorion and keep the latency to a minimum. Also, Flexible Optical B.V. has specifications regarding the size of the PCB and the location of input and output connectors and also demand that all components must be on the same side of the PCB.

4.6.6. Tracing

Tracing is the process of putting traces to electrically connect various components with each other. The design consists of different kinds of signals which will be discussed now. These are power lines, ground planes, analog output signals and digital signals. These all have to fit onto two layers whilst keeping the previously stated design choices into account.

The power lines supply the ICs with power. Their traces should be wider than normal traces to keep power losses to a minimum, though close to the ICs the traces will have to become less wide because of space issues. If traces have to cross each other this should be done in a perpendicular way, minimizing the distortion. It is a general rule, but for power lines it is more important [49]. The DAC has power pairs, which means that it has two pins for each power supply that need to be connected to the same power line. Furthermore, the power lines will need to connect to the decoupling capacitors before they connect to the power input pins.

The analog outputs were already addressed in the previous section. Their traces should be short. They should be isolated from power lines as much as possible. The use of vias needs to be limited. To do this, an optimal connection configuration needs to be found between the analog output pins of the DAC and the 20-pin output connectors. This also means that this optimal connection configuration needs to be implemented in the software using something like a lookup table.

In some cases tracing digital signals has a high priority, because when these have extremely high frequencies this will cause distortion for the rest of the PCB. However because these frequencies are not that high in this design the digital signals are less important and so they are considered last when tracing the design. Of course, it would still be better if these signals could also have short traces and minimal distortion.

In Figure 4.20 an example is given of some traces. The red lines are copper traces on the top layer of the PCB. The blue lines are copper traces on the bottom layer of the PCB. The green circles are vias which go through the board to connect the top and the bottom layer.

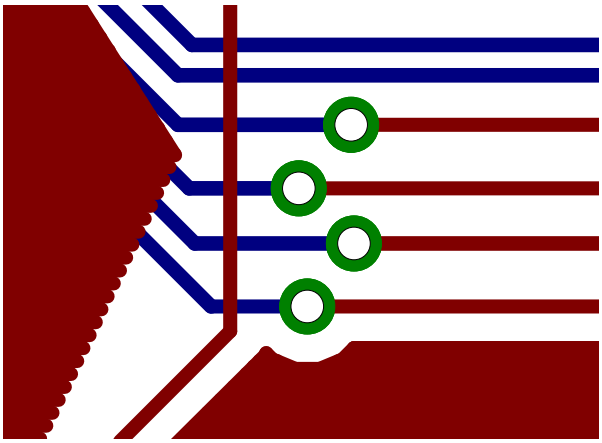


Figure 4.20: Example of some traces.

4.6.7. Grounding

In the design, there are two kinds of ground: digital and analog. The analog ground is used to act as a reference for the analog outputs, but the USB controller also uses it. Analog ground is for signals that need to be protected against disturbances. Components like the ones for power therefore use a different ground, namely the digital ground.

The digital plane is situated on the top layer and is located at the power components being the DAC, the USB controller and some other parts. It is situated around most components, isolating them from interference. The analog plane is situated on the bottom layer because of spatial issues. It is situated around the DAC and the analog outputs, isolating the analog outputs as much as possible.

The two planes have to be connected somehow. This is done only on a single spot using a trace, otherwise there could be a current that flows in circles causing induction effects. Additionally, it is preferred that the two ground planes do not overlap with each other. Although they are both ground, there could still be small voltage differences which can cause parasitic capacitances. Therefore this also will be held into account when finalizing the PCB.

4.6.8. Finalizing the PCB

To finalize the PCB, there still are some things to be done. First of all, the device has to be tested once it has been built. To do this it would be convenient to have some pins to measure. Therefore an 8-pin header to measure all the communication between the USB controller and the DAC is implemented on the PCB. The final PCB design can be viewed in C.3. Unfortunately, the PCB was not manufactured in time to have measuring results in Chapter 5.

Before the design can be produced, it has to be checked. This can be done using certain design rule check files. These files will check if there are sizes or distances in the design that are too small or too large. For instance, it can occur that two traces are too close to each other or that a trace is too close to a via.

When the design has been checked, Gerber files need to be produced. A Gerber file contains the information of a certain layer. It contains for example the bottom copper traces, the top copper traces, the drill holes and the component names. EAGLE can produce these files easily with the use of the so called 'job' files. Finally, these Gerber files can be sent to a manufacturer of PCBs, which in this case was Eurocircuits. All the components of the design and all the layers are listed in Appendix C.3.

Pick-and-Place files are files that allow for a machine to automatically solder the components to the PCB. The necessary files to allow this automatically soldering are: a bill of Materials, the Gerber files and the pick-and-place files themselves. At this time the first two are available, whereas the last one however is not. This last file will not be achieved because of limited time in the project, but it is mentioned in Section 6.5.

5

Test Process and Results

This chapter discusses the methods of testing and the results that were obtained. As was mentioned in Section 4.6 there are no results of the PCB measurements since it was not manufactured in time. Therefore, only the measurement results of the prototype on the breadboard from Section 4.4 are discussed.

5.1. Analog Outputs

There are several things that have to be tested regarding the analog outputs. First of all it must be checked if the correct DA conversion is done. Next the output refresh rate must be measured and the voltage characteristics from the power regulator subsystem and the voltage reference must be reviewed. The latency has to be estimated and measured, and finally, the resolution must be investigated.

5.1.1. Refresh Rate

To determine the output refresh rate, an oscilloscope was used with an accuracy of 5 mV. This allows for measuring the frequency of the output signal. The analog outputs will be altered between their maximum and minimum value. One period of the signal then consists of a maximum and a minimum. If the frequency is then measured with an oscilloscope this will measure the amount of wavelengths, or ups and downs taken together per second. However since in reality sampling occurs at every up flank and at every down flank this measured frequency will have to be multiplied by two to obtain the actual output refresh frequency.

The alternating analog output is shown in Figure 5.1, which is a capture of the oscilloscope. It shows the predicted signal with a 50% duty cycle on ups and downs. The frequency of this signal is 4.4 kHz. The output refresh rate is then twice as high as this frequency, meaning that it is 8.8 kHz.

5.1.2. Resolution

Next the resolution is tested by measuring fifteen subsequent values. The average difference between these values is then calculated and compared to the required resolution. Recall that one LSB equals 152 µV and that the DAC can have a maximum error of four LSB at a certain point and two consecutive steps can have a maximum error of one LSB. The measured analog outputs values which result from this are listed in Appendix C.3. The minimum step size is 0.1 mV, the maximum step size is 0.2 mV and the average step size is 0.15 mV.

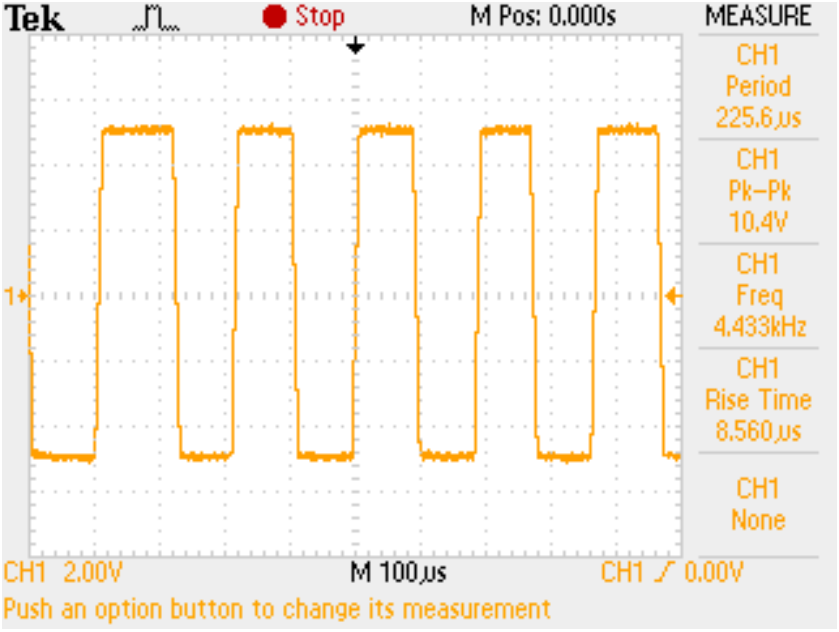


Figure 5.1: Analog output refresh rate measurement.

5.1.3. Output range

To measure the voltage value of the analog outputs, a multimeter is used. The accuracy of this multimeter is 0.1 mV and 1 mA. The minimal and maximal values are tested to check if the range is correct and if the digital value is indeed converted to the correct analog value.

The analog outputs are measured first. The minimum and maximum value are sent to the DAC by the USB controller by using the software. This causes the analog outputs to take the values −5.005 V and 5.005 V as measured by the multimeters respectively. In theory, these values should be −5.000 V and 5.000 V. The range is therefore 0.010 V bigger than it should be.

5.1.4. Latency

The latency is very difficult to measure. It is however possible to give an estimation. Each component in Chapter 4 that transfers data already has an estimation of its latency. Using this a rough estimation can be made about the latency.

The latency of the DAC is approximately equal to 57.2 µs where that of the USB controller is determined to be approximately equal to 125 µs. This results in a total latency of the system of

$$57.2\mu s + 125\mu s = 182.2\mu s. \tag{5.1}$$

5.2. Power

There are power signals and a reference voltage which need to be measured. They were already listed and discussed in Section 4.3. Now it has to be confirmed whether they meet up to the requirements. For this several aspects need to be measured: the nominal voltage, the voltage deviation and the current drawn.

The nominal voltage and the voltage deviation are measured using the oscilloscope which can measure peak-to-peak ripples. All measurements were done in idle mode without a load. The current drawn is measured using the multimeter, where the current in USB is by connecting both ends of this multimeter with a USB current measurer. These power characteristics were measured and displayed in Table 5.1. The captures of the oscilloscope are displayed in Appendix C.3.

V_{DD} and V_{SS} are the supply voltages that are produced by the NTA0515MC. However, in the early stage of the design, the NTA0512MC was used which had output voltages of −12 V and 12 V instead of the earlier discussed −15 V and 15 V. This was later changed to have a little more headroom on the supply voltage for the DAC. It was decided that the NTA0515MC will be used in the PCB and the NTA0512MC for the breadboard. Unfortunately, the NTA0512MC broke down at some point but no new component was placed on the breadboard. Therefore, Table 5.1 shows the power characteristics of the NTA0512MC. Unfortunately, the deviation was not accurately measured. It is however of the same series with the same accuracy, so the measurement is still relevant. Using a multimeter, the total current draw was measured and equaled to 172 mA.

Table 5.1: Measured power characteristics.

Parameter	Nominal voltage	Deviation	Deliverable Current
V_{REF}	2.57 V	±5.60 mV	0 mA
DV_{CC}	3.47 V	±10.4 mV	1 mA
V_{DD}	12.4 V	-	-
V_{SS}	−12.4 V	-	-
V_{REGIN}	5.17 V	±52.0 mV	58 mA

5.3. Calibration

In Section 4.2.5, the zero-scale and full-scale errors were discussed. At the time the breadboard was tested, no calibration was done, even though the nominal output voltage range was not used. The need for calibration will now be discussed.
Earlier in this chapter, the minimum and maximum output voltage were measured and equal to −5.005 V and 5.005 V respectively. The zero-scale error is equal to the difference between the actual and required voltage and expressed in LSB is equal to

$$\text{Zero-scale error} = 0.005\,\text{V} \approx 33\,\text{LSB}. \tag{5.2}$$

This error can be reduced by adding 33 LSB to the default value (0x2000) of the C register. The full-scale error is equal to the zero-scale error plus the difference between the actual and required voltage. The full-scale error is then equal to

$$\text{Full-scale error} = 0.005\,\text{V} + 0.005\,\text{V} = 0.010\,\text{V} \approx 66\,\text{LSB} \tag{5.3}$$

This error can be reduced by subtracting 66 LSB from the default value (0x3FFF) of the M register. This technique is usually enough to calibrate the DAC, but there are limitations. If it does not calibrate properly, increasing the reference value and subsequently recalculating the gain and offset register values can overcome this. The breadboard implementation has not been calibrated, since the DAC will be used in a feedback system, where errors already exist. However, if calibration is necessary or demanded, it can be easily implemented in the connection procedure of the driver.

5.4. Software

In Section 4.5 the software which resulted from the design was discussed. This software has successfully been implemented and is compatible with the USB DAC. It is capable of sending values to all the registers of each DAC with simple finetuning. It can be configured to send data to the input data (X) register but also to the offset (C), gain (M) and special function register using its GPIO pins and the MPSSE configuration mode. An overview of the working of this code can be found in Algorithm 4.5.1. With this software the final product can easily be used by Flexible Optical B.V. and implemented into their feedback system for adaptive optics.

5.5. Printed Circuit Board

The design was finalized by combining all subsystems on a single board. Unfortunately, the PCB was not manufactured in time. Even though the very first version of the design was immediately ordered, the parts arrived at the last day at which the design could be assembled. Parts were soldered to the board and measurements were attempted. However, the available time proved to be too short to have solid results. The result of the unsoldered design is shown in Figure 5.2. A picture of the real PCB manufactured by Eurocircuits can be viewed in Appendix C.10.

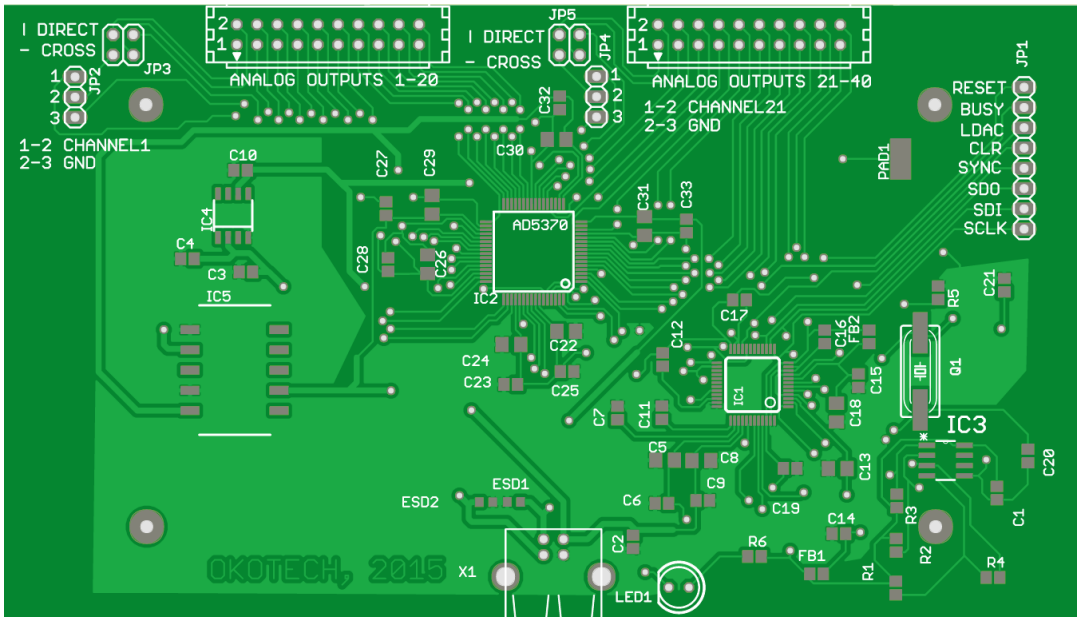


Figure 5.2: Final PCB design.

6

Discussion

In this chapter, the achieved results from Chapter 5 will be discussed per subsystem and explanations will be sought regarding the observed deviations. The measurement results will be compared to the requirements, as stated in Chapter 2.

6.1. USB Controller Subsystem

6.1.1. Refresh rate

The maximum refresh rate of USB 2.0 is limited to 8 kHz and therefore this is a huge bottleneck for the refresh rate. The bulk transfer type that is used in this implementation proved to be almost sufficient while using the USBPcap test, delivering a refresh rate of 7.7 kHz.

However, when measuring the total system the output refresh rate was measured at 8.8 kHz while in theory this should be limited to 8 kHz. It was concluded that this measurement was averaged over a too short amount of time, which gives an inaccurate average since in this case there were only a few transitions measured from minimum to maximum.

Measurements of the breadboard show that the output frequency is 8 kHz, which is also stated in the updated requirements. This means that both estimations made concerning the refresh rate of the USB controller and the DAC subsystem were correct. For a higher refresh rate, for example the initial required refresh rate of 20 kHz, USB 3.0 could be a possible solution.

6.1.2. Latency

The latency has been estimated to be equal to 125 µs with only the USB 2.0 high speed protocol, let alone the rest of the circuit. This puts a heavy constraint on the latency of the total device since already half of the maximum allowed latency is 'used up' by the USB 2.0 protocol. The total latency has been calculated to be 182.2 µs, which is below the maximum of 250 µs. A possible way to improve this latency is by using USB 3.0.

6.2. Digital-to-Analog Converter Subsystem

6.2.1. Resolution

The subsequent sample measurement showed that the average LSB equals 0.15 mV, which is nearly equal to the theoretically determined LSB of 152 µV or 0.152 mV. Furthermore, the minimum and maximum step size do not deviate more than one LSB from the theoretical step size, as guaranteed by the datasheet. The maximum deviation from the theoretical linear case is ±4 LSB, which could reduce the effective resolution of the DAC to 14 bits. During the tests, this maximum deviation was not observed.

6.2.2. Output Range

The minimum and maximum values of the analog outputs were converted nearly without any deviation from the digital values. The range of the analog outputs was 0.010 V bigger than expected and V_{REF} was measured at 2.57 V, while it should have been 2.50 V. This directly affects the range of the analog outputs. This deviation can be corrected for, by using the M and C registers of the AD5370. The final implementation has not been calibrated since the DAC will be used in a feedback system where errors already exist.

6.2.3. Load Capacitance

The drivable load capacitance of the final implementation was hard to determine. The high voltage amplifier, which the DAC will drive in a measurement setup, was not tested in conjunction with the final implementation and therefore, a conclusion can not be drawn about this requirement. In the future, the exact drivable load capacitance could be determined or even improved.

6.3. Power Regulation Subsystem

6.3.1. Reference Voltage

V_{REF} was measured at a nominal value of 2.57 V. That means that the reference voltage has an offset of 70 mV. This is strange since the datasheet guaranteed a nominal value of 2.500 V with a maximum offset of 1 mV. This was however measured on the breadboard prototype. It is possible that using this prototype, noise effects on V_{REF} introduce irregularities, which are the reason for the measured offset. Results will probably be better when using the PCB, since connections are much shorter and less noise or other disturbances can harm V_{REF}.

6.3.2. Supply Voltage

Unfortunately, the NTA0515MC could not be measured, because it was not available for the breadboard prototype. The NTA0512MC, which is of the same series, was initially used and measured in conjunction with the breadboard prototype. Only its nominal voltage was measured before it broke down, and performed as expected. To make sure that the specifications are met, the deviation should also have been measured.

6.3.3. Current Drawn

The current draw of each power signal was also measured. Each individual power signal drew less current than expected. The total current drawn equaled 172 mA, measured with the NTA0512MC. The maximum was determined to be 328 mA and when using the efficiency of the NTA0515MC, the current draw should be equal to 199 mA.

This last value is relatively close to the measured value. Both the measured value and the determined value are for the unloaded case, where the design does not have to drive anything. The drawn current will be higher when there is a load connected to it. There is however a lot of headroom, since the USB connection can deliver up to 500 mA.

6.4. Software

Creating the software to control the USB controller has proven to be a tiresome experience. It has taken a lot longer than was planned because the documentation of the USB controller was not very clear. This caused searching for the drivers and also initial communication and the steps required to connect to the USB controller to be a hard process. However in the end a solution was found and a working code has been delivered.

6.5. Printed Circuit Board

In the design the choice was made to not use the LC-filter at the output of the NTA0515MC, because the supplied power inputs already had their decoupling capacitors. However, the switching noise caused by the switch inside the NTA0515MC should be taken care of directly at the output of the NTA0515MC. Therefore, a future version should have this LC-filter.

There are traces in the PCB design that run on the top layer and then go to the bottom layer, because they need to cross a different signal. After this crossing, a via is used to return to the top layer. However, in most cases this is unnecessary since it concerns the analog outputs. This is because the output connector is of the through-hole type so the traces can also connect to it on the bottom layer.

Unfortunately, the PCB was not manufactured in time so there are no solid results to conclude anything about the quality of the PCB design. However, the manufactured design was the first version of the PCB design. After this first design was ordered, work continued on the design, which up until now led to a fourth design of the PCB. The first version has a lower chance of working correctly, but can certainly be used to verify if we are on the correct path.

When the PCB design is working correctly, the manufacturing process can be made more effective. To do this, pick-and-place files should be made, which take away the job of soldering by hand or with a reflow oven.

7

Ethical Aspects

When a design is technically correct, this does not automatically imply that it can be released to the public immediately. This is because a technical design does not only influence technical aspects but also many non-technical aspects, which also hold for this design. Therefore the ethics regarding the USB DAC will be discussed in this chapter.

7.1. Applications

Adaptive optics is commonly a harmless technology when used for measuring things like the smoothness of mirrors or aberrations in the atmosphere. However it can also be used for purposes which are not so harmless, for example air defense in military purposes. Here adaptive optics can be implemented when for example a laser is used to shoot a hostile rocket out of the air. In the feedback system which is used to keep tracking the hostile rocket the USB DAC of this thesis could be used, or since this thesis is publicly available a recreated DAC could be implemented using another input than USB but the same technique, resulting in a much faster refresh rate. When used in a military application like this, the design could save lives when in the right hands, but harm many if it falls into the wrong hands.

7.2. Production Circumstances

Another ethical aspect which needs to be considered is the production. When this design is taken into mass production, the production process will be optimized and made as cheap as possible. Therefore it is likely that all or most components will be ordered from low-income countries. Low-income countries are not well-known for their strict regulation on a fair production process and therefore it is the task of the management to take on this responsibility. The employees of the companies where these components are produced have a right to be treated fairly. Working conditions for them have to be good: they should earn a fair wage, have an acceptable work schedule, should work in a safe environment and have their own say in the company. Next to the production of these components the resources that are required for these components should also be delved fairly. Herefore the people that delve these resources should have the same good working conditions as the employees of the component manufacturing company.

7.3. Restriction of Hazardous Substances

The just discussed delved resources which are used in the design should also comply to the *Restriction of Hazardous Substances* (RoHS). This directive restricts (with exceptions) the use of six hazardous materials in the manufacture of various types of electronic and electrical equipment [50]. It restricts the following six substances with maximum permitted concentrations:

1. Lead (0,1 %)
2. Mercury (0,1 %)
3. Cadmium (0,1 %)
4. Hexavalent chromium (0,1 %)
5. Polybrominated biphenyls (0,1 %)
6. Polybrominated diphenyl ether (0,1 %)

The directive implies that every single substance that could be theoretically be separated mechanically has to follow these above maximum concentrations. Since the RoHS also applies to imported products it applies to the USB DAC created in this project. When this DAC has even slightly more of one of the above concentrations in one substance, the entire system would fail the requirements of the RoHS. Since this is only a directive, following it is not compulsory but certainly an ethical aspect that should be considered.

8

Conclusions

8.1. Conclusions

The implementation was split up into three subsystems: USB controller, DAC and power regulation. Each subsystem has its own requirements, which are summarized in Table 8.1.

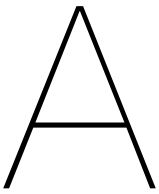
Table 8.1: Fulfilled 'next generation' DAC specifications.

Parameter	**'Next generation' DAC**	**Fulfilled?**
Analog outputs	40 channels	✓
Refresh rate	8,000 frames per second	✓
Resolution	16 bits per channel	✓
Output mode	Synchronous for all channels	✓
Output range	0 V - 5 V or −5 V - 5 V with adjustable limits	✓
Power	Supplied by USB	✓
Load capacitance	> 500 pF	Unknown

The breadboard prototype meets all requirements, except for the drivable load capacitance, since this was not measured and can therefore not be guaranteed. The prototype has 40-channels, each with a resolution of 16 bits, and all channels are updated synchronously. The requirement for the refresh rate has been adjusted during the project, but this increased our chances of having a working prototype at the end of the project. The reason for the adjustment was that the demanded refresh rate was not achievable with USB 2.0. The resolution of the analog outputs is almost met, since the accuracy is nearly correct. While the output range is bipolar, it does not have adjustable limits, and the device can be powered using the USB connection.

8.2. Future Work

Several aspects of the current prototype can be improved. The power subsystem meets the specifications, but the NTA0515MC still needs to be measured, since no unit was available for the prototype on the breadboard. Also the resolution can be improved by properly calibration of the prototype, which was not yet done at the time this thesis was published. Adjustable limits can be easily implemented in the current driver. The drivable capacitive load was also not measured, which still need to be done to guarantee correct operation. Besides this several documents and files have yet to be finished for replication, fabrication and assembly purposes. These are for example pick-and-place files, drivers, code examples and code wrappers.



List of Acronyms

Acronym	Meaning
AC	Alternating Current
DA	Digital-to-Analog
DAC	Digital-to-Analog Converter
DC	Direct Current
DNL	Differential Nonlineairity
IC	Integrated Circuit
INL	Iintegral Nonlinearity
LPF	Low-pass Filter
LSB	Least Significant Bit
MPSSE	Multi-Protocol Synchronous Serial Engine
MSB	Most Significant Bit
PCB	Printed Circuit Board
PWM	Pulse-Width Modulation
R-string	Resistive-string
RoHS	Restriction of Hazardous Substances
SMD	Surface Mount Device
USB	Universal Serial Bus
USBPcap	Universal Serial Bus Packet capture

B

Author per Section

Chapter	Section	Author
Preface	-	Y.T.B. Mulder
Abstract	-	J. Romijn
1. Introduction	-	Y.T.B. Mulder
2. Problem Definition	1. Existing Product	Y.T.B. Mulder
	2. Program of Requirements	Y.T.B. Mulder
3. Research	1. System Overview	Y.T.B. Mulder
	2. Universal Serial Bus Subsystem	D.S.M. Verhaert
	3. Controller Subsystem	D.S.M. Verhaert
	4. Digital-to-Analog Converter Subsystem	Y.T.B. Mulder
	5. Power Regulation Subsystem	J. Romijn
4. Implementation	1. USB Controller Subsystem	D.S.M. Verhaert
	2. Digital-to-Analog Converter Subsystem	Y.T.B. Mulder
	3. Power Regulation Subsystem	J. Romijn
	4. Prototype	J. Romijn
	5. Software	D.S.M. Verhaert
	6. Printed Circuit Board	J. Romijn
5. Test Process and Results	1. Method	J. Romijn
	2. Results	J. Romijn
	3. Printed Circuit Board	J. Romijn
6. Discussion	1. USB Controller Subsystem	D.S.M Verhaert
	2. Digital-to-Analog Converter Subsystem	Y.T.B. Mulder
	3. Power Regulation Subsystem	J. Romijn
	4. Software	D.S.M. Verhaert
	5. Printed Circuit Board	J. Romijn
7. Ethical aspects	1. Applications	D.S.M. Verhaert
	2. Production Circumstances	D.S.M. Verhaert
	3. Restriction of Hazardous Substances	D.S.M. Verhaert
8. Conclusions	1. Conclusions	All
	2. Future Work	All

C

Figures

C.1. USB Subsystem

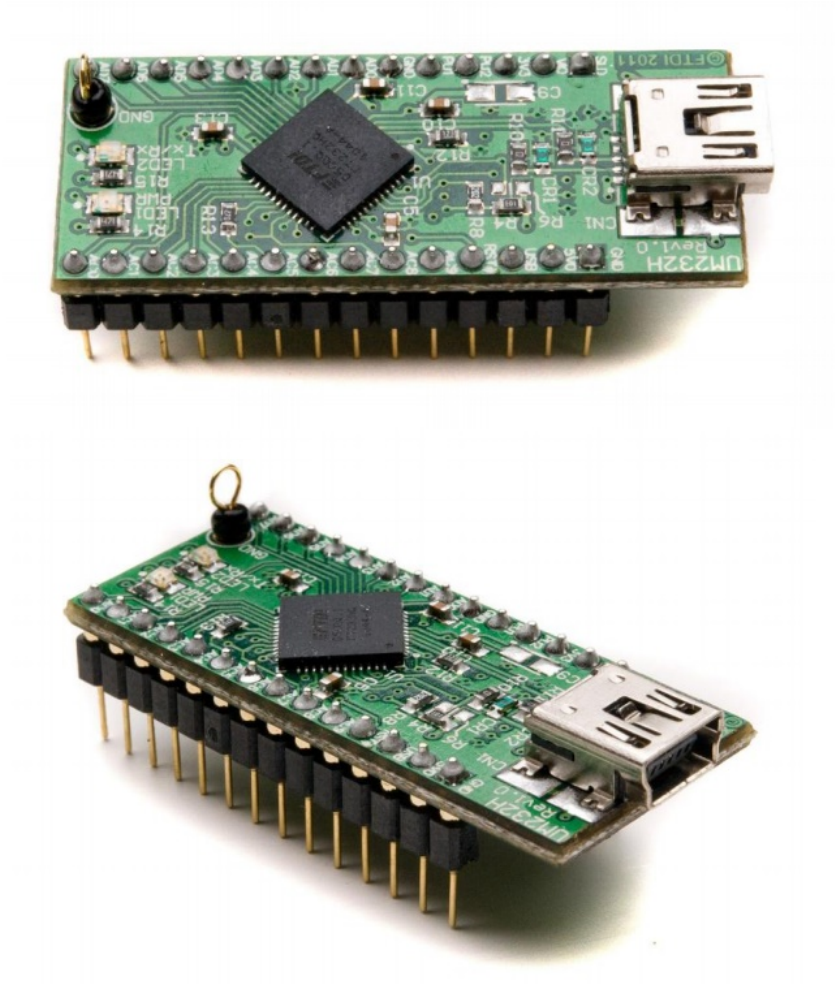


Figure C.1: UM232H USB to Serial/FIFO Development Module [36].

C.2. DAC Subsystem

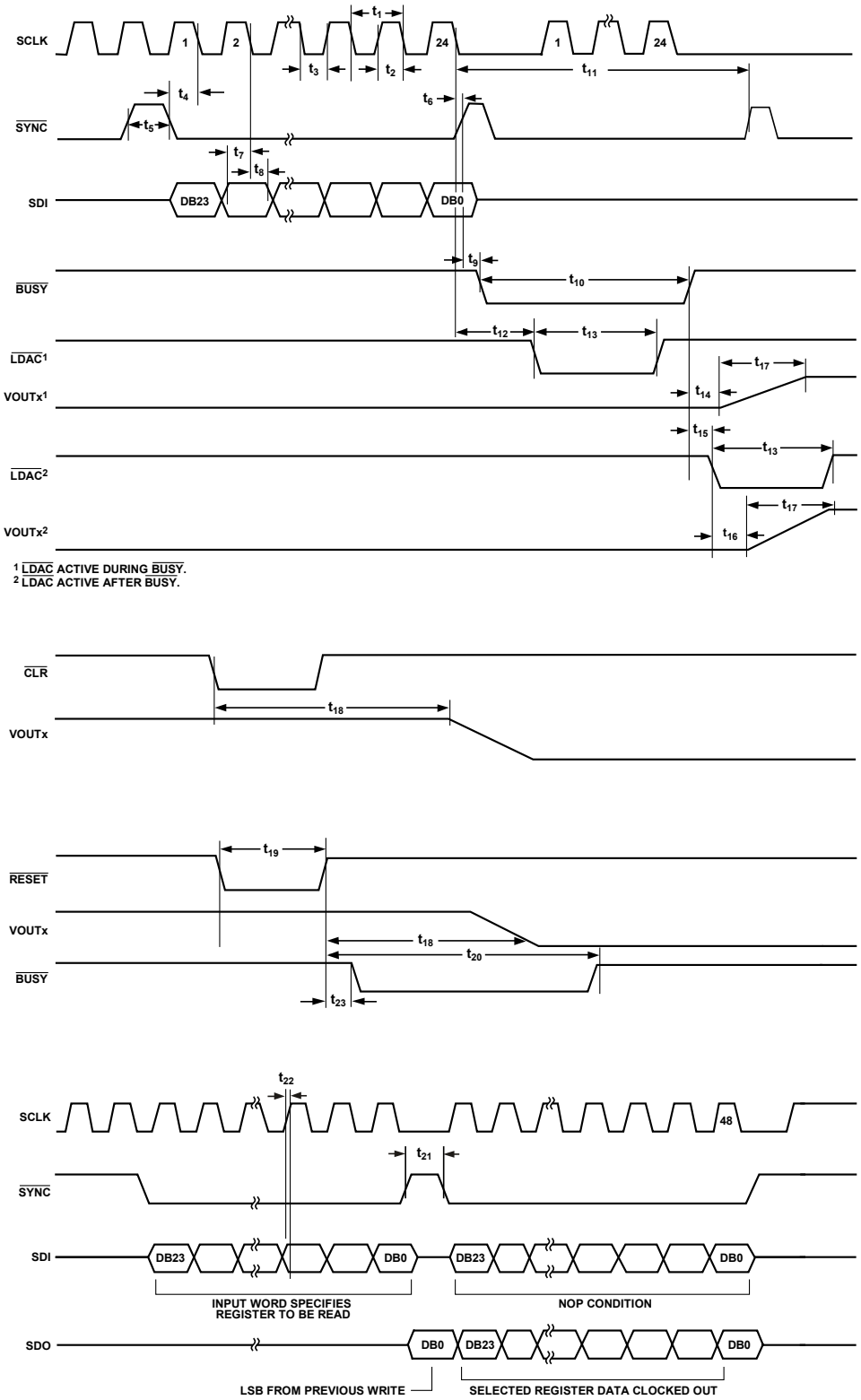


Figure C.2: AD5370 SPI interface write and read timing diagram [40].

Figure C.1 summarizes the timing characteristics of Figure C.2. These timings hold for the following parameters.

- DV_{CC} = 2.5 V to 5.5 V.
- V_{DD} = 9 V to 16.5 V.
- V_{SS} = -16.5 V to -4.5 V.
- V_{REF} = 3 V.
- AGND = DGND = SIGGND = 0 V.
- CL = 200 pF to GND.
- R_L = open circuit.
- Gain (M), offset (C), and DAC offset registers at default values.
- All specifications T_{MIN} to T_{MAX}, unless otherwise noted.

Table C.1: AD5370 SPI interface write and read timing characteristics [40].

	Limit at T_{MIN}, T_{MAX}				
Parameter [1, 2]	Min	Typ	Max	Unit	Description
t_1	20			ns	SCLK cycle time
t_2	8			ns	SCLK high time
t_3	8			ns	SCLK low time
t_4	11			ns	$\overline{SYNC}$ falling edge to SCLK falling edge setup time
t_5	20			ns	Minimum $\overline{SYNC}$ high time
t_6	10			ns	24th SCLK falling edge to $\overline{SYNC}$ rising edge
t_7	5			ns	Data setup time
t_8	5			ns	Data hold time
t_9			42	ns	$\overline{SYNC}$ rising edge to $\overline{BUSY}$ falling edge
t_{10}			1.5	µs	$\overline{BUSY}$ pulse width low (single-channel update)
t_{11}			600	ns	Single-channel update cycle time
t_{12}	20			ns	$\overline{SYNC}$ rising edge to $\overline{LDAC}$ falling edge
t_{13}	10			ns	$\overline{LDAC}$ pulse width low
t_{14}			3	µs	$\overline{BUSY}$ rising edge to DAC output response time
t_{15}	0			ns	$\overline{BUSY}$ rising edge to $\overline{LDAC}$ falling edge
t_{16}			3	µs	$\overline{LDAC}$ falling edge to DAC output response time
t_{17}		20	30	µs	DAC output settling time
t_{18}			140	ns	$\overline{CLR}$/$\overline{RESET}$ pulse activation time
t_{19}	30			ns	$\overline{RESET}$ pulse width low
t_{20}			400	µs	$\overline{RESET}$ time indicated by $\overline{BUSY}$ low
t_{21}	270			ns	Minimum $\overline{SYNC}$ high time in readback mode
t_{22}			25	ns	SCLK rising edge to SDO valid
t_{23}			80	ns	$\overline{RESET}$ rising edge to $\overline{BUSY}$ falling edge

[1] Guaranteed by design and characterization, not production tested.
[2] All input signals are specified with $t_R = t_F$ = 2 ns (10% to 90% of DV_{CC}) and timed from a voltage level of 1.2 V.

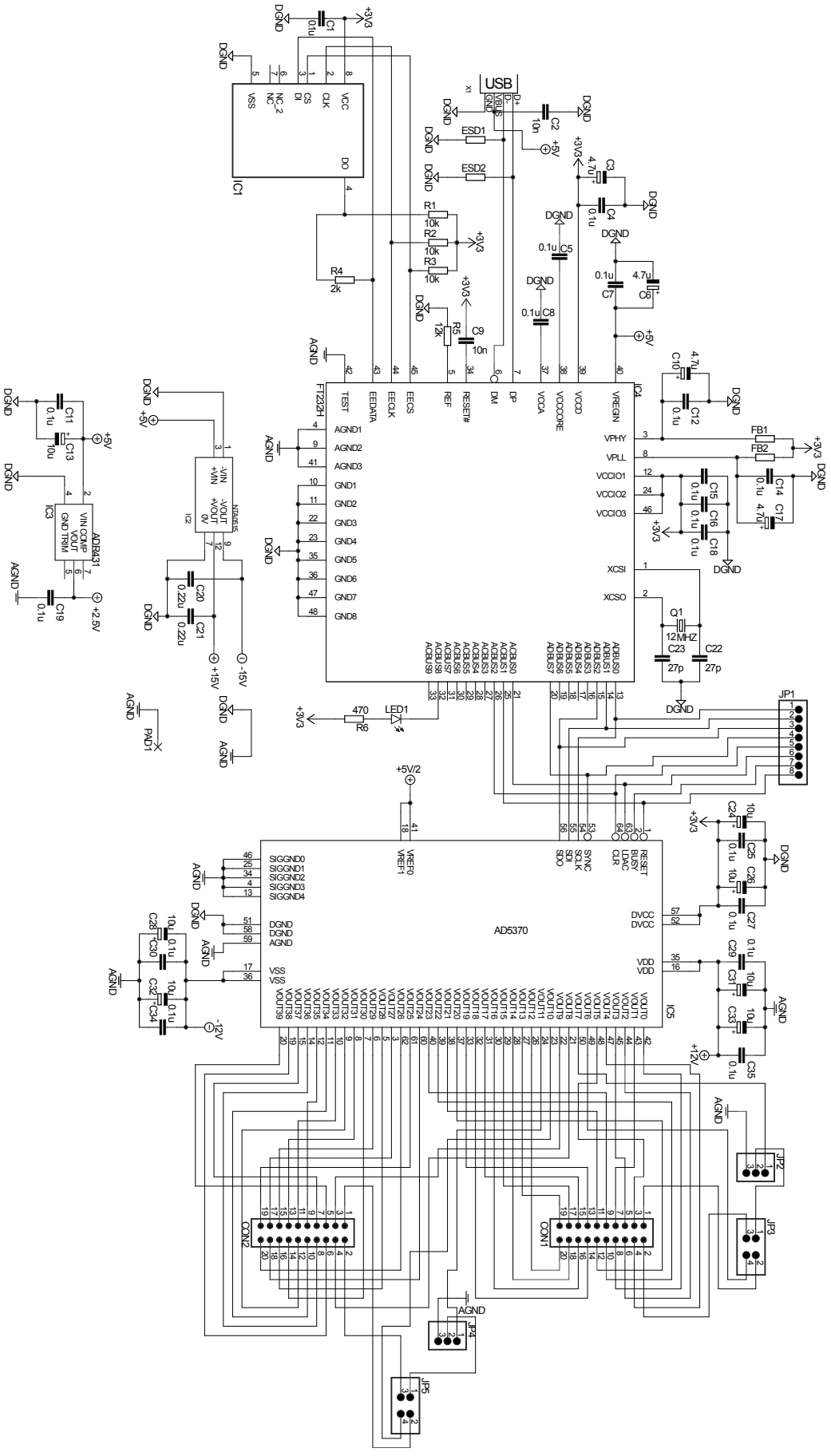


Figure C.3: Schematic of the final implementation.

C.3. PCB

Table C.2: PCB bill of materials. Some components are very general, so they don't have a Farnell code.

Component	Manufacturer	Part number	Value	Farnell[1]	Quantity
Capacitor	AVX	TAJR475K010RNJ	4.7 µF	1135076	4
Capacitor	AVX	TPSP106M010R2000	10 µF	1658716	6
Capacitor	Kemet	C0805C103J5RACTU	10 nF	1650861	2
Capacitor	Multicomp	MC0805B104J500CT	0.1 µF	2320839	18
Capacitor	Multicomp	MC0805N270J500CT	27 pF	1759196	2
Capacitor	Murata	GRM219R61E106KA12D	10 µF	2426961	1
Crystal	Abracon	ABLS-12.000MHZ-B2-T	12 MHz	1652551	1
DAC	Analog Devices	AD5370BSTZ	-	2461390	1
DC-DC	Murata	NTA0515MC	-	1021594	1
EEPROM	Microchip	93LC56B-I/SN	-	9758259	1
ESD	Littlefuse	PGB1010603NRHF	600 Ω/0.5 A	2383320	2
Ferrite bead	Littlefuse	PGB1010603NR	-	2383319	2
LED	-	-	5mm	-	1
Output pins	-	-	20 pins	-	2
Pinheader	-	-	1x03	-	2
Pinheader	-	-	1x08	-	1
Pinheader	-	-	2x02	-	2
V_{REF}	Analog Devices	ADR431BRZ	-	1498664	1
Resistor	Multicomp	MC0805S8F4700T5E	470 Ω	1632491	1
Resistor	Multicomp	MCWR08X1002FTL	10 kΩ	2447553	3
Resistor	Multicomp	MCWR08X1202FTL	12 kΩ	2447563	1
Resistor	Multicomp	MCWR08X2001FTL	2 kΩ	2447622	1
USB	Lumberg	USB-B 2411 03	-	1308876	1
USB	FTDI	FT232HL-REEL	-	1870924	1

[1]This column species the order code for `http://nl.farnell.com`.

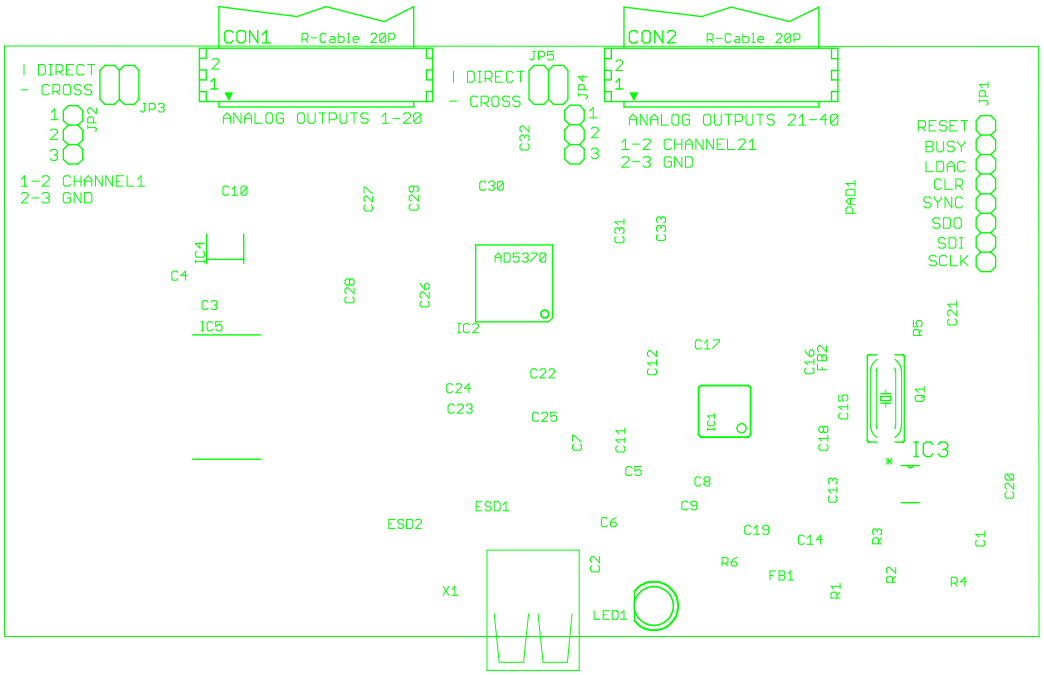


Figure C.4: The text layer of the PCB, visualized using the Gerber files.

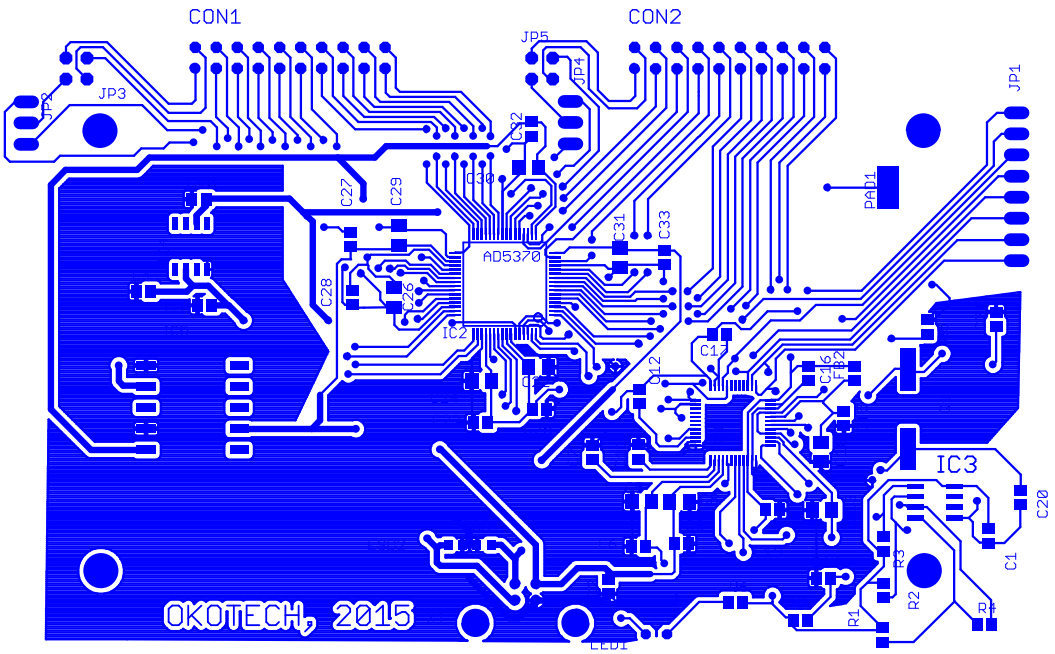


Figure C.5: The top copper layer of the PCB, visualized using the Gerber files.

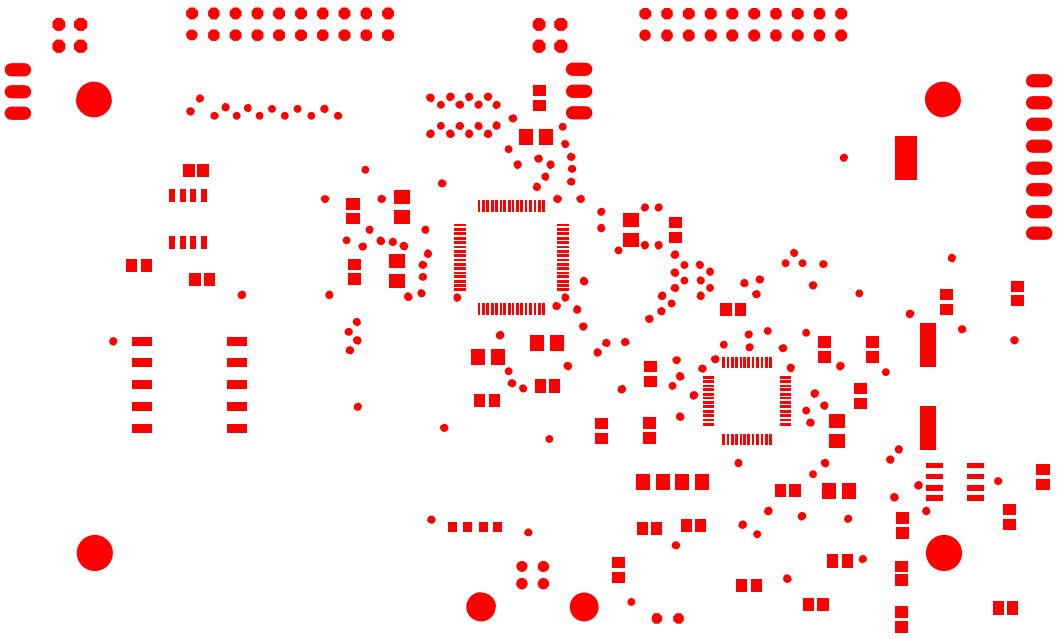


Figure C.6: The exposed parts of the top layer of the PCB, visualized using the Gerber files.

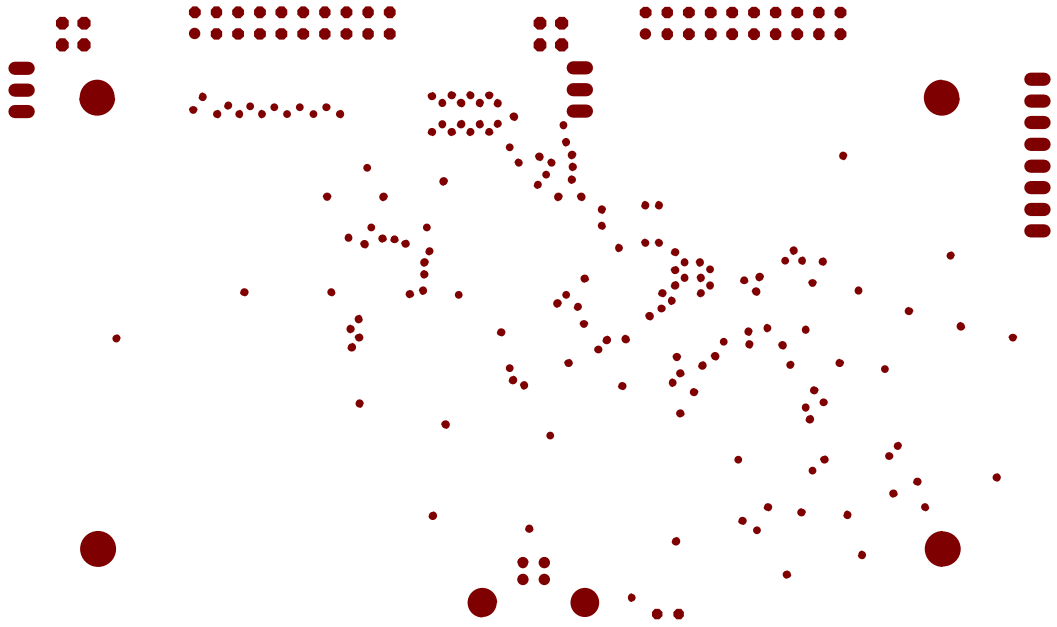


Figure C.7: The exposed parts of the bottom layer of the PCB, visualized using the Gerber files.

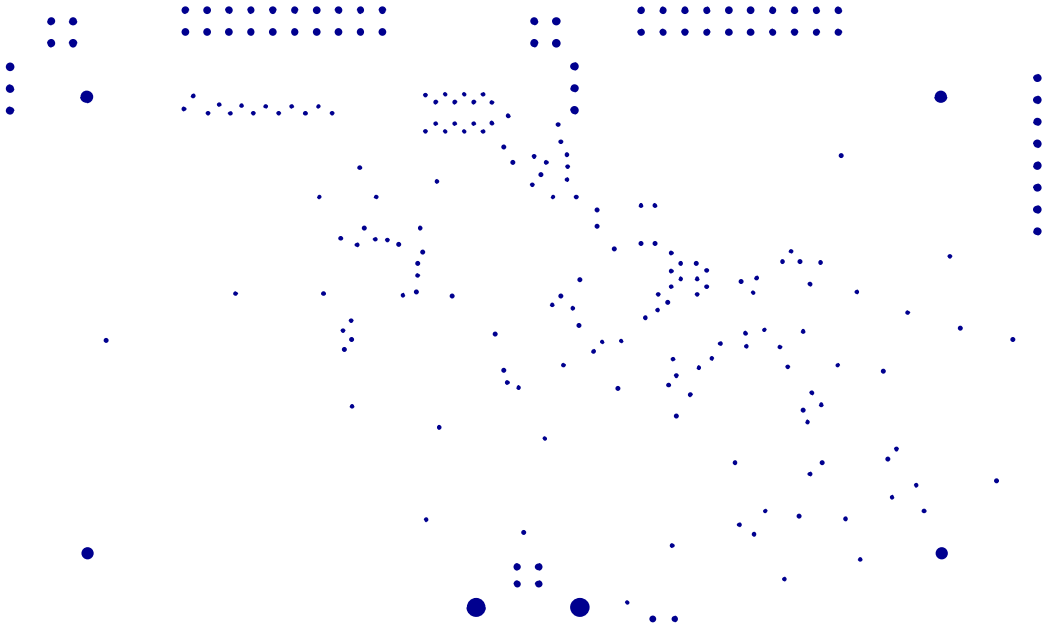


Figure C.8: The drill hole layer of the PCB, visualized using the Gerber files.

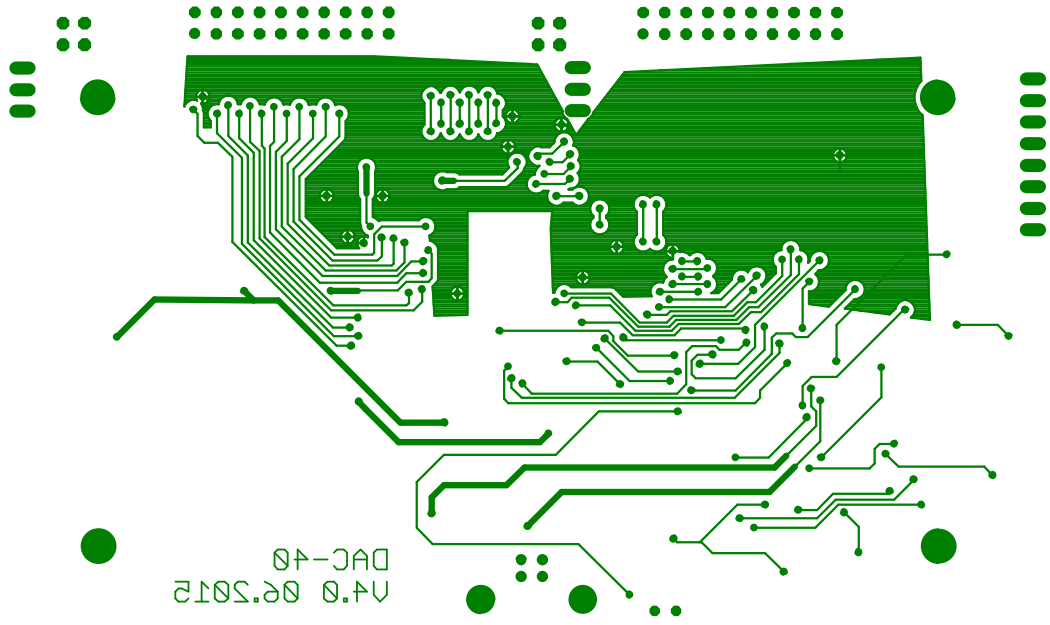


Figure C.9: The bottom copper layer of the PCB, visualized using the Gerber files.

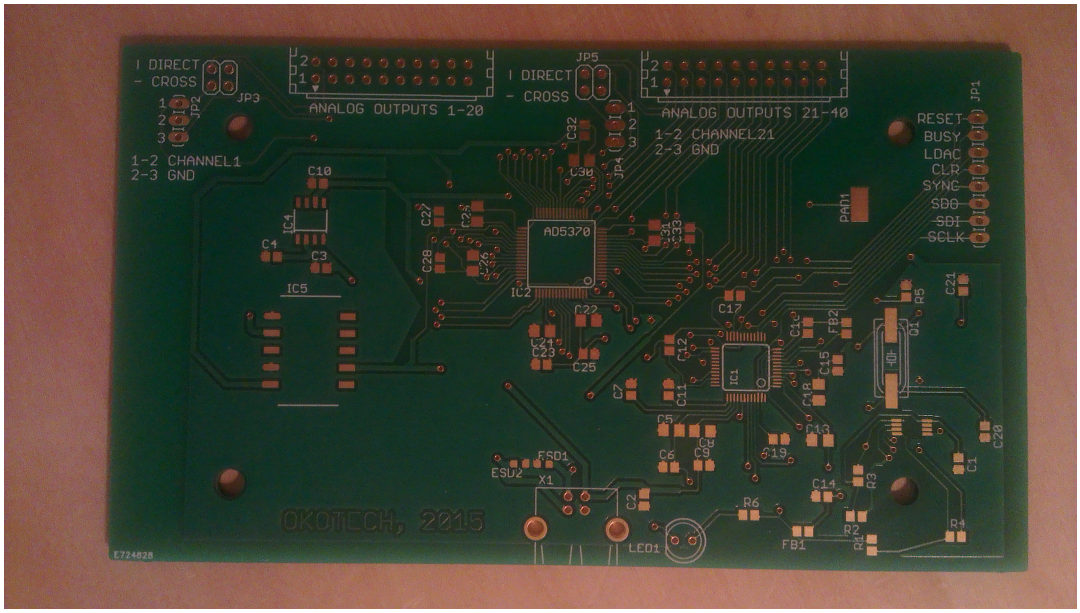


Figure C.10: PCB design, manufactured by Eurocircuits, without components soldered on it.

C.4. Results

Table C.3: Subsequent values measurement.

Step	Value [mV]
1	0.5
2	0.6
3	0.8
4	0.9
5	1.1
6	1.3
7	1.4
8	1.6
9	1.7
10	1.9
11	2.0
12	2.2
13	2.3
14	2.4
15	2.6

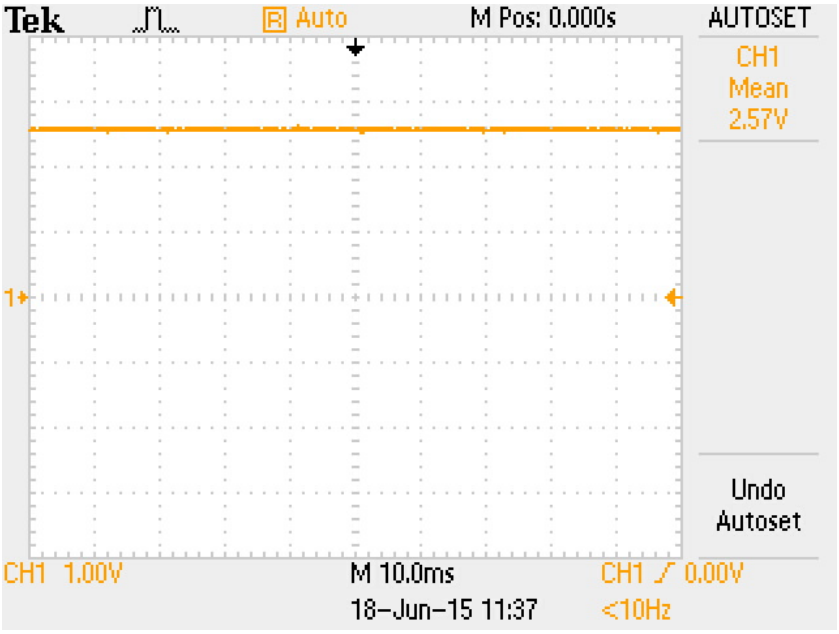


Figure C.11: Mean value of V_{REF}.

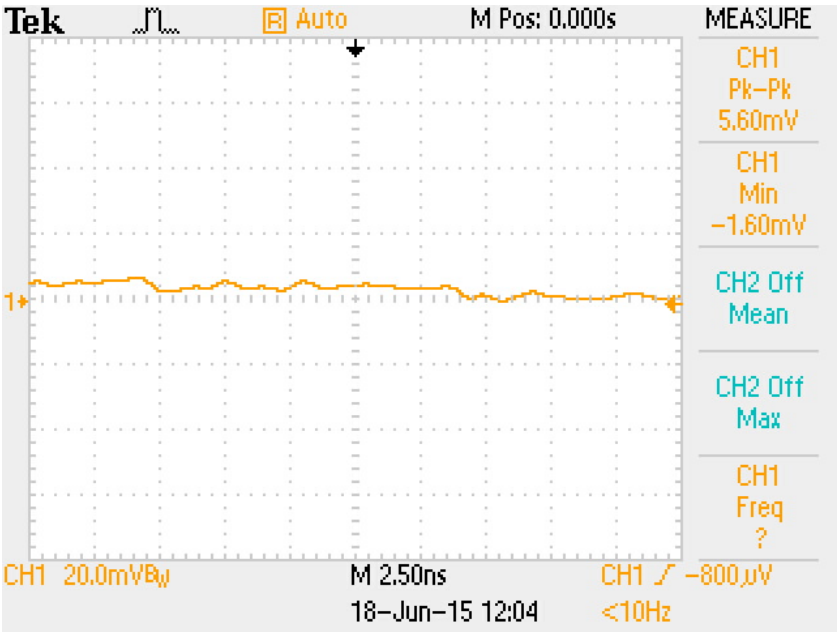


Figure C.12: Ripple of V_{REF}.

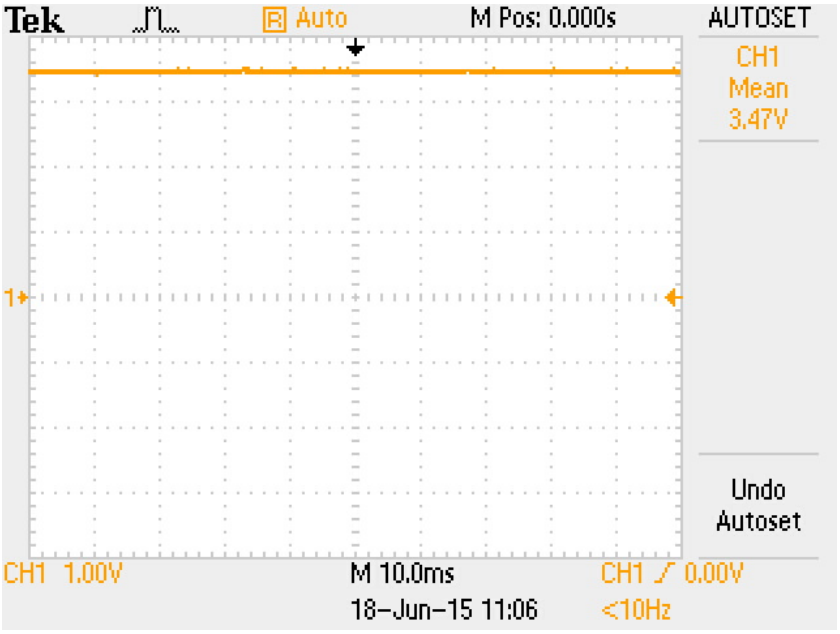


Figure C.13: Mean value of DV_{CC}.

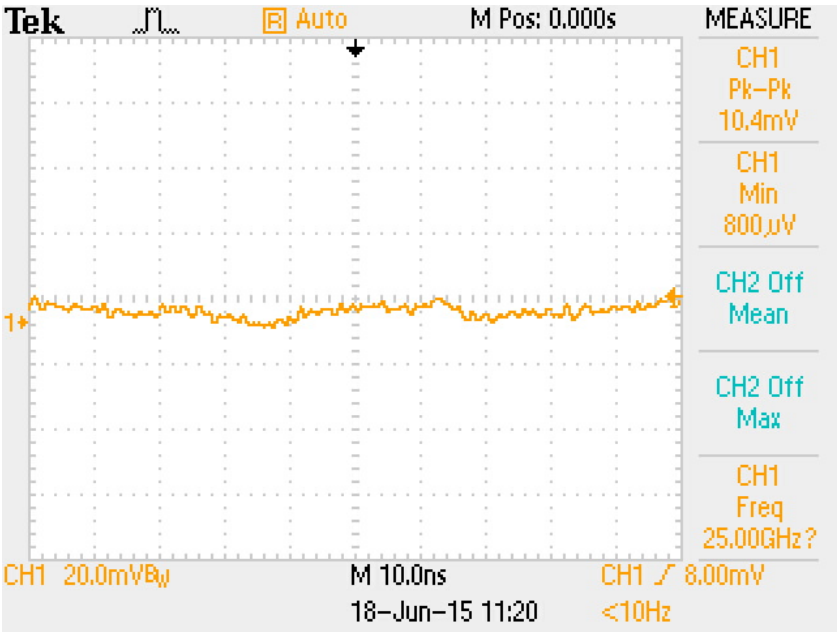


Figure C.14: Ripple of DV_{CC}.

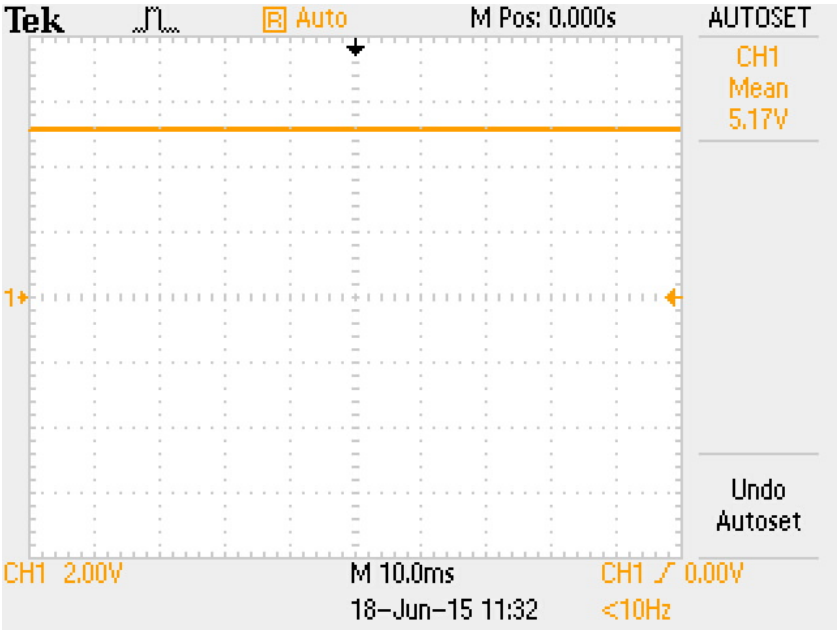


Figure C.15: Mean value of V_{REGIN}.

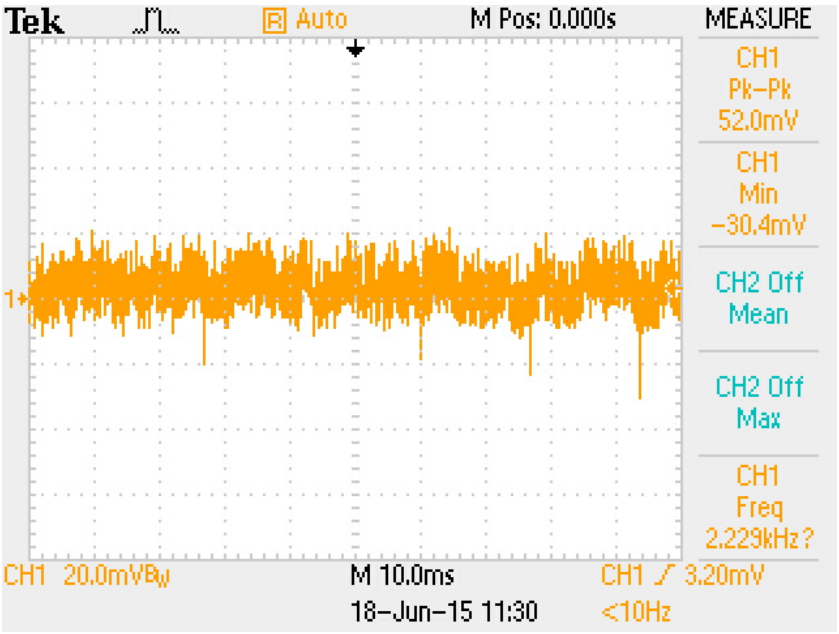


Figure C.16: Ripple of V_{REGIN}.

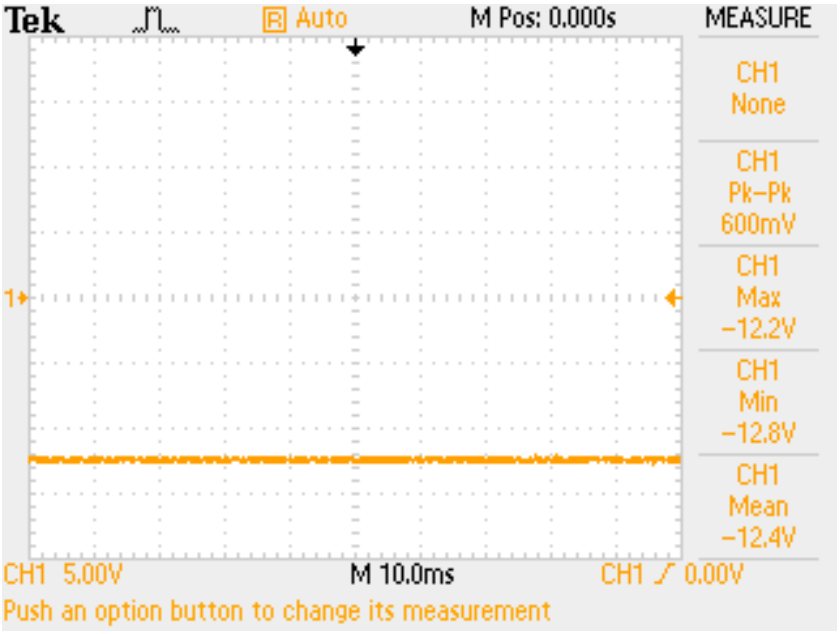


Figure C.17: Mean value of V_{SS}.

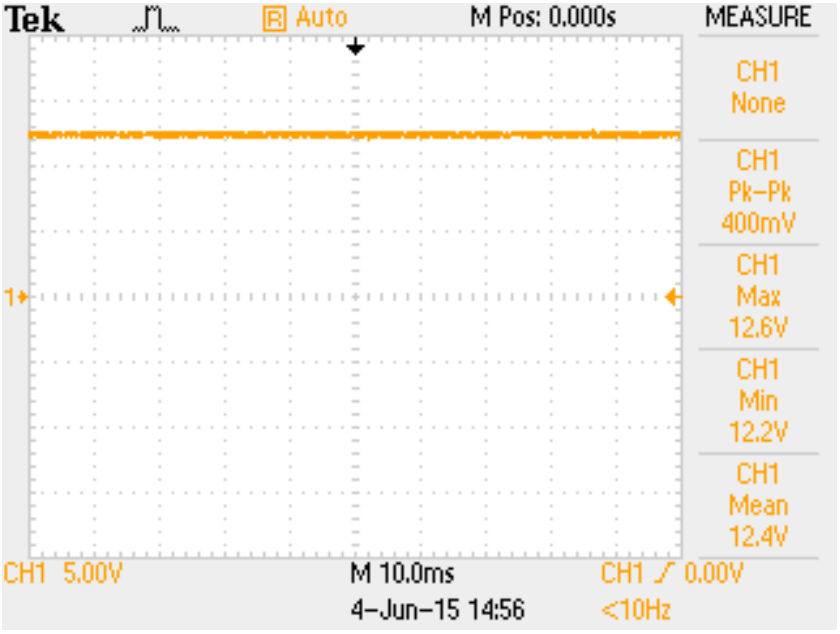


Figure C.18: Mean value of V_{DD}.

Bibliography

[1] ESO/B. Tafreshi (twanight.org). (2012, June). Laser Guide Star, Very Large Telescope [Online]. Available: http://www.eso.org/public/images/potw1225a/

[2] OKO Tech Flexible Optical BV. (2015, March). High-speed multichannel USB DAC proposal.

[3] OKO Tech Flexible Optical BV. (2013, May). OKO GUIDE TO ADAPTIVE OPTICS [Online]. Available: http://www.okotech.com/images/pdfs/catwww4.pdf

[4] L. van Dam, S.A. van Dijk, M.S. van der Kaay. (2015, June). A High-Speed 40-channel USB 3.0 DAC for Adaptive Optics. ISBN 978-94-6186-494-9

[5] M. Wright. (2013, Feb, 20). Implementing USB 3.0 in MCU- and Microprocessor-Based Systems [Online]. Available: http://www.digikey.com/en/articles/techzone/2013/feb/implementing-usb-30-in-mcu-and-microprocessorbased-systems

[6] Compaq, Hewlett-Packard, Intel, Lucent, Microsoft, NEC, Philips. (2000, April). Universal Serial Bus Specification [Online]. Available: http://www.usb.org/developers/docs/usb20_docs/usb_20_060115.zip

[7] IDS Imaging Development Systems GmbH. (2012). USB 3.0 – Background Factors, New Features, and Applicability as a Camera Interface [Online]. Available: http://www.xmultiple.com/xwebsite-forum88.htm

[8] USB press. (2013). SuperSpeed USB 10 Gbps – Ready for Development [Online] . Available: http://www.usb.org/press/USB-IF_Press_Releases/SuperSpeedUSB_10Gbps_Available_20130731.pdf

[9] Hewlett-Packard Company, Intel Corporation, Microsoft Corporation, Renesas Corporation ST-Ericsson, Texas Instruments. (2013). Universal Serial Bus 3.1 Specification, pp 5-19 [Online]. Available: http://www.usb.org/developers/docs/usb_31_060115.zip

[10] J. Axelson. (2005). USB Complete: Everything You Need to Develop Custom USB Peripherals.

[11] Peacock, Craig. (2011, June). USB in a NutShell. [Online]. Available: http://www.beyondlogic.org/usbnutshell/usb4.shtml#Control.

[12] Lindblom, Jim. (2012, December). Serial communication [Online]. Available: https://learn.sparkfun.com/tutorials/serial-communication?_ga=1.141534266.926496093.1430384539

[13] Leens, Frederic. (2013). Introduction to I^2C and SPI protocols [Online]. Available: http://www.byteparadigm.com/applications/introduction-to-i2c-and-spi-\protocols/?/article/AA-00255

[14] Hord, Michael. (2003, July). I2C [Online]. Available: https://learn.sparkfun.com/tutorials/i2c.

[15] NXP Semiconductors. (2014, Apr). I^2C-bus specification and user manual [Online]. Available: http://www.nxp.com/documents/user_manual/UM10204.pdf

[16] Using the I2C Bus [Online]. Available: http://www.robot-electronics.co.uk/acatalog/I2C_Tutorial.html

[17] Motorola, Inc. (2000, Jan.). SPI Block Guide V03.06 [Online]. Available: http://www.ee.nmt.edu/~teare/ee308l/datasheets/S12SPIV3.pdf.

[18] Grusin, Mike. (2013, Jan.) Serial Peripheral Interface (SPI) [Online]. Available: https://learn.sparkfun.com/tutorials/serial-peripheral-interface-spi

[19] Balachandran, Sasang. (2009, August). General Purpose Input/Output (GPIO) [Online]. Available: http://www.egr.msu.edu/classes/ece480/capstone/fall09/group03/AN_balachandran.pdf

[20] U. Beis. An Introduction to Delta Sigma Converters [Online]. Available: http://www.beis.de/Elektronik/DeltaSigma/DeltaSigma.html

[21] B. Baker. (2011). How delta-sigma ADCs work [Online]. Available: http://www.ti.com/general/docs/lit/getliterature.tsp?baseLiteratureNumber=slyt423

[22] S. Owen. (2012, August). Delta-Sigma ADC Basics [Online]. Available: http://stuartowen.com/home/2012/08/13/delta-sigma-adc-basics/#!/single_blog

[23] S. Smith. (1997). The Scientist and Engineer's Guide to Digital Signal Processing. 1st ed. San Diego: California Technical processing, pp. 35-66.

[24] E. Janssen, A. van Roermund. (2011). Look-Ahead Based Sigma-Delta Modulation. ISBN 978-94-007-1386-4

[25] W. Kester. (2008, October). ADC Architectures III: Sigma-Delta ADC Basics [Online]. Available: http://www.analog.com/media/en/training-seminars/tutorials/MT-022.pdf

[26] A. Oppenheim and R. Schafer. (1975). Digital Signal Processing, Englewood Cliffs, Prentice Hall.

[27] S. Tilden and Solomo Max. (2011). IEEE Standard for Terminology and Test Methods of Digital-to-Analog Converter Devices [Online]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=6152113

[28] Texas Instruments, Inc. (2008, September). Using PWM Output as a Digital-to-Analog Converter on a TMS320F280x Digital Signal Controller [Online]. Available: http://www.ti.com/lit/an/spraa88a/spraa88a.pdf.

[29] A. Gardner, M. Salman, D. Fernandes, J. Roberts. (2006, October). Digital to Analog Converters [Online]. Available: ume.gatech.edu/mechatronics_course/DAC_F06.ppt

[30] P. O'Reilly and C. El-Khoury. (2012, March). Modern DAC's and DAC Buffers Improve System Performance Simplify Design [Online]. Analog Dialogue. Available: http://www.analog.com/library/analogDialogue/archives/46-03/dacs_buffers.pdf.

[31] All About Circuits. Vol. IV - Digital [Online]. Available: http://www.allaboutcircuits.com/textbook/digital/chpt-13/r-2r-dac/

[32] BI Technologies. BI Technologies R2R Resistor Ladder Networks [Online]. Available: https://en.wikipedia.org/wiki/Resistor_ladder

[33] Planet Analog. (2007, October). Understand the differences between R2R and String DAC architectures [Online]. Available: http://www.planetanalog.com/document.asp?doc_id=527548

[34] Capacitorguide. Coupling and Decoupling [Online]. Available: http://www.capacitorguide.com/coupling-and-decoupling/

[35] FTDI. (2014). Single Channel HiSpeed USB to Multipurpose UART/FIFO IC [Online]. Available: http://www.ftdichip.com/Support/Documents/DataSheets/ICs/DS_FT232H.pdf.

[36] Future Technology Devices International Limited. (2011). UM232H Single Channel USB Hi-Speed FT232H Development Module Datasheet. [Online]. Available: http://www.ftdichip.com/Support/Documents/DataSheets/Modules/DS_UM232H.pdf.

[37] Microchip. 2K Microwire Compatible Serial EEPROM [Online]. Available: http://ww1.microchip.com/downloads/en/DeviceDoc/21794F.pdf.

[38] J. Pihlajamaa. (2012, July 3). Benchmarking Raspberry Pi GPIO Speed [Online]. Available: http://codeandlife.com/2012/07/03/benchmarking-raspberry-pi-gpio-speed/.

[39] T. Mon. (2013, Feb). USBPcap - USB Packet capture for Windows [Online]. Available: http://desowin.org/usbpcap/index.html.

[40] Analog Devices. (2013, Jul). 40-Channel, 16-bit, Serial Input, Voltage Output DAC [Online]. Available: http://www.analog.com/media/en/technical-documentation/data-sheets/AD5370.pdf.

[41] O. Ozbek. (2011, October). Improve DAC integral nonlinearity through gain correction [Online]. Analog EE Times Europe. Available: http://analog-eetimes.com/en/improve-dac-integral-nonlinearity-through-gain-correction.html?cmp_id=71&news_id=222902593

[42] Murata. Isolated 1W Dual Output SM DC/DC Converters [Online]. Available: http://power.murata.com/data/power/ncl/kdc_nta.pdf

[43] Analog Devices. Ultralow Noise XFET Voltage References with Current Sink and Source Capability [Online]. Available: http://www.analog.com/media/en/technical-documentation/data-sheets/ADR430_431_433_434_435.pdf

[44] Wikipedia. Figure of a boost converter [Online]. Available: https://en.wikipedia.org/wiki/Boost_converter

[45] Future Technology Devices International Ltd. Command Processor for MPSSE and MCU Host Bus Emulation Modes [Online]. Available: http://www.ftdichip.com/Support/Documents/AppNotes/AN_108_Command_Processor_for_MPSSE_and_MCU_Host_Bus_Emulation_Modes.pdf

[46] CadSoft EAGLE. PCB design software. http://www.cadsoftusa.com/

[47] Sparkfun. (2008, June). Lecture 10 - Eagle: Creating a new part [Online]. Available: https://www.sparkfun.com/tutorials/110

[48] Resistorguide. Resistor Sizes and Packages [Online]. Available: http://www.resistorguide.com/resistor-sizes-and-packages/

[49] Analog Devices. Printed Circuit Board (PCB) design issues [Online]. Available: http://www.analog.com/library/analogDialogue/archives/43-09/EDch%2012%20pc%20issues.pdf

[50] European parliamant and concil. (2011, June). DIRECTIVE 2011/65/EU OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL of 8 June 2011 [Online]. Available: http://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:02011L0065-20140129&from=EN