Group Management and Technology

Assembly and implementation of a 5-axis machine

UC Leuven Limburg
MOVING MINDS

Stagescriptie ingediend door

Jeroen Lekens

Tot het verwerven van het diploma van:

Electromechanics
Specialization Automation

Intern counsellor(s):
Prof. Uwe Heuert
Eng. Dietmar Glatz

Lect. Bart Smeets

Academic year: 2017-2018
Reference number: E18_S_EM_AUT_12

Group Management and Technology

Assembly and implementation of a 5-axis machine

Stagescriptie ingediend door

Jeroen Lekens

Tot het verwerven van het diploma van:

Electromechanics
Specialization Automation

Intern counsellor(s):
Prof. Uwe Heuert
Eng. Dietmar Glatz

Lect. Bart Smeets

Academic year: 2017-2018
Reference number: E18_S_EM_AUT_12

Assembly and implementation of a 5-axis machine

Foreword

First of all, I would like to introduce myself. My name is Jeroen and I study a Bachelor in Electromechanical engineering at the college UCLL, formerly known as the Khlim. The abbreviation UCLL stands for University College Leuven Limburg, because of the collaboration between the University of Leuven and the College of Diepenbeek. The campus itself is situated in the East of Belgium, in the province of Limburg.

To finish my studies, I have to write a thesis and do a presentation about the project that I've done during my internship. As for the internship, I chose to complete it in Germany as part of the Erasmus program. The reason was that one of my lecturers informed me about the program and on top of that a former classmate who went on Erasmus to Merseburg convinced me to live the same unique experience.

The audience my thesis is written for is mainly engineers, people with a technical background, students and everyone who is interested in technical innovations.

During the realisation of my project I've been assisted by Steven Hollmann but with the intention of me taking the majority of tasks upon myself. Both my supervisors and Steven gave me the opportunity to write my own piece of work. I hope to have succeeded in this matter.

Now I wish to thank everyone who supported me and or contributed to the realisation of my Bachelor thesis.

First of all, I would like to thank my internal promotor Prof. U. Heuert and my mentor D. Glatz of the University of Applied Sciences in Merseburg. They were always ready to help me forward whenever I had any questions concerning my project and their guidance throughout my thesis helped me going into the right directions. Apart from that, they were also easy to reach out to and very helpful giving me an insight into the subject.

Secondly, I'm also very grateful to my external mentor B. Smeets for the guidance throughout my internship and the quick replies whenever I had any questions. Furthermore, I would like to express my gratitude to my language teachers U. Schiffke and M. Kuhn for reviewing my thesis and suggesting improvements. Besides that, I would also like to thank L. Coenegracht, my lecturer in microcontrollers, for the quick and helpful response when I asked him about debouncing. A very special gratitude goes to S. Hollmann for assisting me throughout the project and making it possible for me to come up with the right solutions.

Furthermore, I want to thank the International Office and the buddy team for helping me get around Merseburg and giving me an unforgettable experience. I'm also thankful to my parents, siblings, family and friends for supporting me in my decision to go abroad and in the completion of my thesis. I would also like to mention my other colleagues, they contributed to a very pleasant work environment and were always happy to help wherever they could. Last but not least, a very sincere thank you to all the other people I met during my internship.

Table of contents

Abstract

Assembly and implementation of a 5-axis machine

by : Jeroen Lekens

promoters : U. Heuert - Prof. HoMe

D. Glatz - Eng. HoMe

S. Hollmann – Eng. HoMe

B. Smeets - Lect. UC Leuven-Limburg

Exceeding Solutions is a spin-off company founded by Professor Uwe Heuert, located in Merseburg, which is in the east of Germany in the federal state of Saxony-Anhalt. Exceeding Solutions works closely together with the University of Applied Sciences of Merseburg also called HoMe and mainly focuses on smart metering. Their major objective is to build a smart metering test system and provide the security standards for smart meter gateways in order to implement them in smart grids. Apart from their main objective they also have a few other projects, for example research on 3D printing by means of a 5-axis machine, under the supervision of the engineer Dietmar Glatz.

The principle of a 5-axis 3D printing system is relatively unutilized because of the uncertainty around the industrial uses of 3D printing. Therefore, research is currently being done into the applications of 3D printing and the principles which are suitable for industrial purposes. The major objective of this thesis is the creation of a prototype with a different approach to general 5-axis 3D printing principles. The aim is to be able to follow the same procedures as industrial robots but without restricting the total amount of workspace considerably and also to avoid the problems of singularities which occur in robotic arms. On top of that the system shouldn't be limited to 3D printing but also be able to do other CNC operations by making the tools interchangeable.

Furthermore, a test environment to repair or reconstruct certain machinery components will be created so that maintenance time and costs can be reduced since it isn't always necessary to manufacture new parts. The reason being able to reduce maintenance time and cost by not having to manufacture new parts.

This thesis will describe the procedures we used in order to design, assemble, implement software for the system and to solve the problems related to that.

List of figures

Glossary

Backlash

The room or space which remains when different gear or transmission elements with most often teeth mesh together.

CAD

Computer aided design

CNC

Computer numerical control

Dimensional stability

The degree to which a material is able to maintain its original dimensions when subjected to changes.

Effector

This is the tool which is used by the machine.

PA

Polyamide

Lead

The axial advance or linear displacement of a helix or a screw during one revolution.

Pitch

This is the distance between two teeth of a gear or two threads of a screw or bolt.

Pitch can also be the rotatory movement around the X-axis in a cartesian coordinate system.

SLA

Stereolithography apparatus or simplified photo-solidification or resin printing.

Slicing

The process of taking a 3D model and translating the model into individual layers which are then used for generating the machine code that 3D printers use for printing.

Specific strength

Known as the strength-to-weight ratio it is the material's strength divided by its density.

Yaw

The rotatory movement around the Z-axis in a cartesian coordinate system.

Chapter 1: Introduction

In this chapter, the problem Exceeding Solutions wants to solve will be described. First of all, there is following a description about the company Exceeding Solutions, the variety of subjects they deal with and the market position they have. After that, there is a background story and the importance of the application in the future. Next there is a description about the new topic which my thesis revolves around. To conclude, a summary will be given concerning the different aspects which will be addressed in this essay.

1.1 Exceeding Solutions

Exceeding Solutions is a spin-off company which provides hardware and software solutions throughout Germany. Currently solutions are already being offered and used by several renowned partners mostly for testing purposes. They range from solutions within the process and automation technology as well as smart metering. On top of that, IT services are increasingly in demand all over Germany due to the digitization of both work and society. This is why Exceeding Solutions excels at providing personalised software which meets the needs of their customers. In essence Exceeding Solutions is an innovative company which is always in search for new and better technological developments in a variety of sectors so that it has become a renowned company for its expertise in a wide range of subjects.

Exceeding Solutions also took an interest in building a system which is able to do a variety of things, for example 3D printing and 3D scanning. The reason is that they have become partnered with a company which wants to build a modular system for cleaning industrial installations. The goal is to do some research into the topic by creating a representation of the installation which is able to recognise objects and able to handle an ultra-high-pressure waterjet. Basically, the principle or procedures needed in order to do that are very similar to the 3D printing and 3D scanning applications. Therefore, the design of a prototype which is similar to the needed system will be the subject of my thesis.

1.2 Additive manufacturing

Additive manufacturing is nothing new, 3D printing, for example, was actually already present in the 1980's and the first technique to be patented was SLA. Apart from that, 3D printing did not have a big breakthrough until just a few years ago. This was because the only existing principles were patented and there was no need for additive manufacturing to be used in business. Thus, 3D printing was cast aside. Only a few main 3D printing manufacturers continued to develop 3D printers and research applications within several sectors and for certain target groups. To give some examples, industrial, medical, aeronautical, automotive and eventually also individual use.

1.3 3D printing is booming

Until recently, additive manufacturing was not well known by most people. Just a few years ago, people shed new light on additive manufacturing mainly because of the patents having expired and the increase in developed CAD software becoming available. This made 3D printing more accessible to a group of mainly individuals who formed communities. Because of the popularity which 3D printing gained within this group, the industry took interest again and has reconsidered the uses for additive manufacturing upon this day. The focus is now on finding new, promising methods for production and not just on prototyping. At this point there is already a wide variety of materials and principles for additive manufacturing being used while more innovations are still to come. Therefore, speculations are being made for additive manufacturing to bring about a revolution in generic consumption-driven production. This could possibly result in a cheaper, personalised and more durable way of manufacturing.

1.4 The importance of my project

The major objective of my thesis is to build a prototype in order to do further research into the applications of 3D printing, 3D scanning and other CNC operations. The addition of 3D scanning will make it possible to recreate and repair parts with minimum effort and on top of that, having interchangeable tools will result in flexible and faster production. Yet another advantage could be the reduction in maintenance time and cost because there is not always a need for manufacturing new parts.

We try to achieve this goal by approaching the mechanical design from a different perspective. A 5-axis machine is created by adding two extra axes to a conventional fused filament fabrication 3D printer with 3 cartesian axes. These additional rotatory axes make it possible to move the effector relative to the work plane.

However, the challenge is how to support the extruded material when the extruder is pointed within a certain angle, away from the vertical position. Another difficulty is the way software will need to be able to translate the information given to the system into machine language. Since a CNC-controller is used, machine language is required to coordinate the drivers of the actuators. The instructions that are needed to do that are created by slicing CAD models. This means that the model created with the help of a CAD program is made understandable for the 3D printer.

At the moment there is no such machine which is able to perform the given tasks available on the market. As mentioned earlier, there is also the future application for a modular cleaning installation which will use the same principles. Therefore, this is a good opportunity to do research into this topic and find suitable principles and get an insight into industrial and individual manufacturing.

1.4.1 Robotic manipulators

Concerning the mechanical design, it is also possible to use an industrial robot arm. However, the disadvantage is that the workspace available will be restricted considerably because an industrial robot's movement is limited. Besides that, kinematic singularities which occur in industrial manipulators like robot arms will prove to be the greatest difficulty for 3D printing purposes. Therefore, the mechanical construction will consist of a gantry system. The major advantage of using a gantry is, of course, that the workspace is not restricted.

1.5 Scope of my thesis

This bachelor's thesis will mainly focus on designing a system and overcoming problems in order to be able to work with a certain amount of specifications and become an adaptable asset for a wide range of applications. In chapter 2, we will discuss the procedures used and the requirements for the mechanical design. In chapter 3, we will address a feature required for the axis providing the yaw motion and the way we approached it. Following up with chapter 4 we describe the procedure of configuring the stepper motors for the axes of the system. After that, we will describe the connections, the wiring and the implementation of the electronics in chapter 5. Finally, we will present some results in chapter 6 and as well, conclusions which can be drawn from the work that we have accomplished.

Chapter 2: Mechanical design

One of the most important parts of the thesis is the mechanical design. As mentioned above, the system has to be able to do multiple CNC driven task. The most import examples would be 3D printing and 3D scanning. Therefore, we want to create a design which features great stiffness in every axis. We want to achieve this in order to obtain an accurate machine which is able to withstand, for example, the vibrations due the changes in direction and speed during 3D printing. This is why the machine should be robust and stiff. Besides that, we want to have a great repeatability and be able to move in a wide variety of directions at different velocities.

2.1 Design of the machine

By the time I arrived in Merseburg and got my project assigned, there were already concrete plans for the machine to be made. It was already decided, for example, that we would design the machine around an OpenBuilds kit, the ACROsystem [1]. On top of that, part of the extruder holder was already created using Rhinoceros CAD software.

As for the benefits of this OpenBuilds building kit, we can easily modify the design according to our preferences. That is because the company provides a wide variety of components and products, for example, aluminium profiles which can be used freely within a custom-made design. Secondly, the use of the ACROsystem will result in quite an accurate representation of what the machine should look like in its finalised stage. The reason for using the building kit is that we are able to make the prototype economical and within reasonable time.

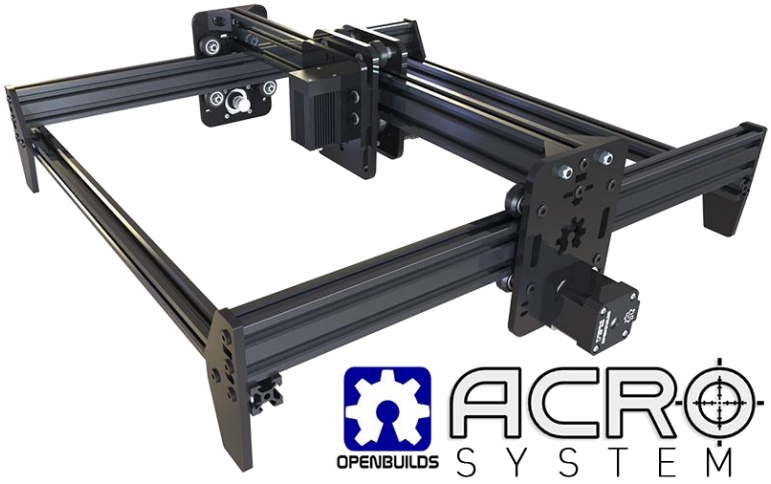

Figure 1 - OpenBuilds ACRO system from [2]

The idea is to place the gantry system on top of a structure made with profiles and add several other axes to it. The gantry will provide the XY-plane, a linear rail will become the Z-axis and the

remaining two rotatory joints are for the yaw and pitch motion of the effector. To visualise the machine, I've provided several images of the design in the attachment.

2.2 Limitations of designing around the building kit

What has to be considered are the limitations to the tolerances which can be achieved with this system. As for the disadvantages which could come with the system, we can foresee issues within the components that are used in the gantry system. For example, the ACROsystem is a gantry system which consists of aluminium profiles, polymer wheels, polymer mounting plates and a teethed timing belt transmission.

First of all, a belt transmission is not the greatest solution in terms of accuracy and repeatability of the system. That is because of the gaps and spaces between the teeth of the pulleys and the belts in order to prevent them from interlocking. Because of this clearance, also known as backlash, hysteresis is formed which in essence is a certain deviation in displacement.

Furthermore, the acrylic components do not offer enough strength and therefore we have to replace and redesign them wherever this is possible.

2.2.1 Why is backlash undesirable?

Firstly, backlash is used in terms of gears and means that there is some room for error, in other words, tolerance between the elements. In our case, we use a teethed timing belt and a pulley. Whenever then, the teeth mesh with each other, there is this little space which is not totally filled up by both transmission elements, sometimes referred to as mechanical play. The disadvantage is therefore that this play makes it difficult to achieve a great accuracy in positioning systems.

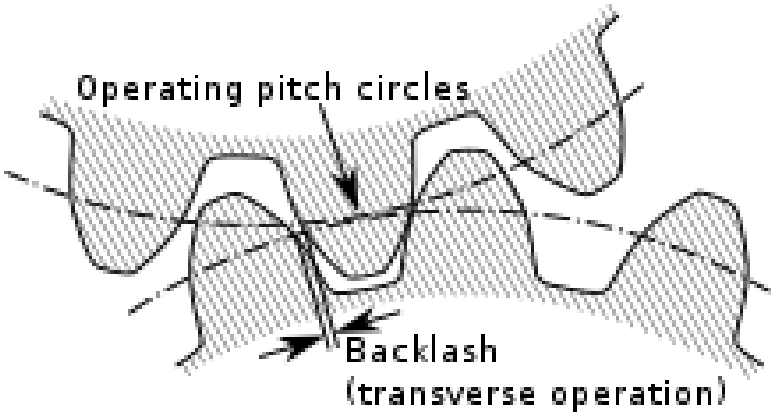

Figure 2 - Backlash illustration from quora.com [3]

On the other hand, backlash cannot be avoided for the majority of transmission elements because a certain amount of lash must be allowed to prevent the elements from jamming. This means for example, that the gears are getting locked up in their motion. Luckily the effects of

backlash can mostly be compensated or negated within software or by using specially designed gears. As a result, backlash might not be too critical for a 5-axis machine.

In addition to backlash, we also have a certain amount of elasticity because of the teethed timing belt which results in a certain tolerance within the displacement. This difference with each displacement will definitely cause the accuracy of the system to be limited and has to be considered. That is why in the end we might have to opt for a different transmission depending on the workload and the purpose of the machine.

2.2.1 The addition of the Z-axis to the gantry system

The ACROsystem is provided with another axis which resembles the Z-axis. The Z-axis is a beam which consists of a leadscrew transmission system, anti-backlash components and also guidance wheels made of polymer material. Because of the fact that the weight of the axis is substantial and there are components made of polymer, we had to improve the rigidness of the system. Therefore, we made custom mounting plates out of aluminium to compensate the momentum which the Z-axis creates around the X-axis. A sectioned image of the mounting plates can be found at page 49.

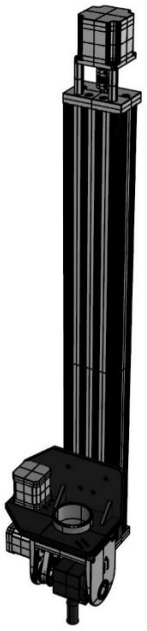

Figure 3 - Assembly of the Z-axis and extruder

2.2.3 The X and Y-axis

Furthermore, the X-axis of the system is similar to the Y-axis, apart from the driving principle. The X-axis uses two stepper motors which have to collaborate and which are placed opposite each other with the X-axis in between. For the X-axis, stiffness is even more important because we have little room for error considering the alignment of the system. Therefore, we experimented with a few setups of mounting plates and profiles before we found the better solution. In the end we found that a C-beam profile would meet the requirements in terms of rigidness. Using an additional double profile, we made sure the mounting plates were placed parallel. The combination of parallel plates and the rigidness of the profiles makes it possible to reduce the alignment error to a minimum.

2.2.4 The holder for the extruder with the pitch and yaw axis

Next up is the design of the holder for the extruder which was already partially designed. My role concerning this holder was to modify the design bearing the issue of placing the limit switches in mind. As for the holder of the extruder, we were quite flexible because we designed the parts to be 3D printed. Therefore, changes to the design are easily made and, on top of that, we can greatly reduce the amount of weight. The filament we used to print the parts of the holder is a PLA reinforced one with carbon fibres. These give the parts better properties regarding stiffness, dimensional stability and strength-to-weight ratio.

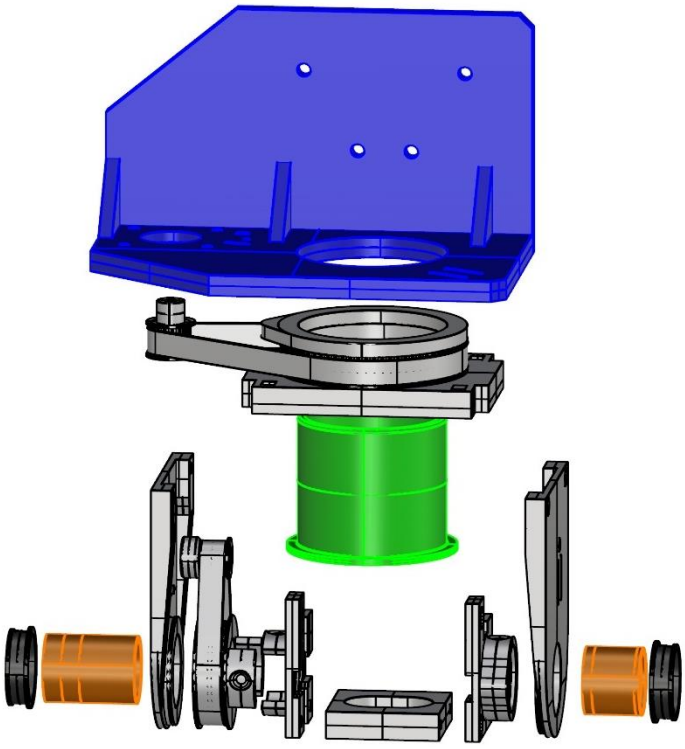

Figure 4 - Exploded view from the extruder holder

2.2.5 The frame of the machine

To become a sturdy framework for the system we used aluminium profiles, fixated with custom made corner plates. We used several types of profiles coming from the OpenBuilds part store, therefore you can find several sketches with the measurements in the attachment.

Because the frame is not exceedingly stiff with this setup we plan to reinforce the structure with side beams and plates mounted in between the currently placed profiles. The assembly will then become closed up from three sides and also receive a ground plate at the bottom of the frame. This should result in the construction becoming more robust and vibration resistant. For an illustration of the current framework, have a look in the attachment at page 45.

2.3 Materials and components

Aluminium is mainly used since it is lightweight and thus especially beneficial for us because of the specific strength it provides. Having lightweight components is necessary in order to reduce the load on the system. The Z-axis, however, is in itself - without the extruder - already a heavy load which we have to consider.

Another concern is the polymer wheels which are needed to guide the gantry system over the full extent of the axis. The polymer wheels consist of a thermoplastic called Delrin which is known for its high rigidness, low friction applications and the exceptional property to maintain its original dimensions. This property is often referred to as dimensional stability. Apart from its beneficial properties, the wear which will happen over time needs to be taken into consideration. Besides that, it is also not certain to which extent this material can be used for our purposes considering the strength of the material and environmental properties. Since we are building a prototype for a system which will be used for real applications in the near future, the results from this project will be of importance in many respects.

As for the future uses of this prototype, all the flaws in design, materials and or principles being used should be determined first in order to make improvements for the final version. For example, changing the belt transmission system into a rack and pinion gear mechanism could result in a stiffer and more robust solution. However, backlash would still remain.

In addition, you can find several images of parts which we designed for the system within the attachment, at the end of this thesis.

Chapter 3: Limit switches for the yaw axis

The purpose of the jaw axis is that we want to be able to move the yaw axis further than one full revolution. It should also be able, for example, to rotate around the circumference of a cylinder which is positioned stationary inside the work plane. One of the applications could be that we want to be able to print filament across the total extent of the circumference of a cylinder. However, the problem is that we want to ensure the axis is restricted in its movement.

In order to control the 5-axis machine we use a CNC USB controller from PlanetCNC. This CNC controller gives us several options for adding limit switches and work with range and motion limits. These options could make it very easy to move within a predefined range after the homing procedure is done.

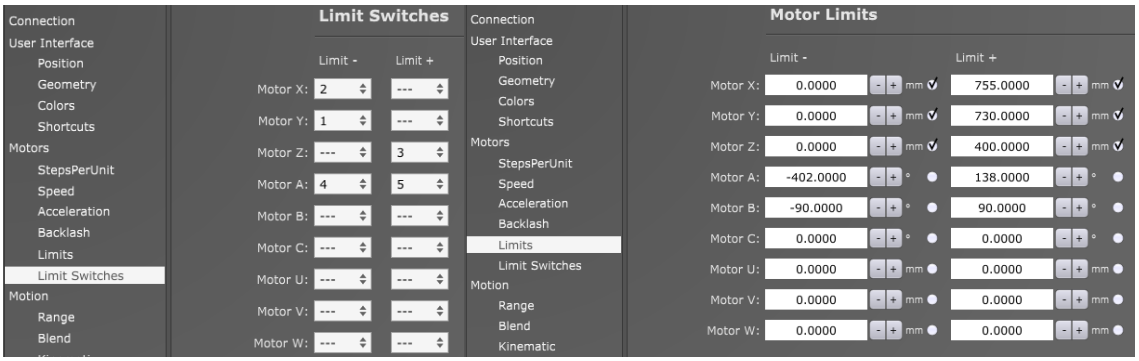

Figure 5 - Limit switch and motor limit configuration in PlanetCNC

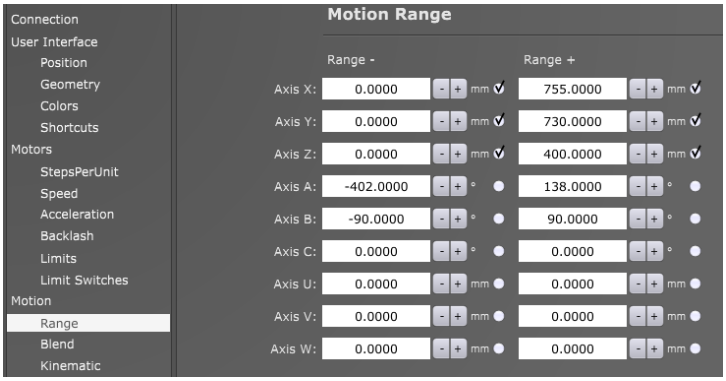

Figure 6 - Motion range configuration in PlanetCNC

3.1 Why go further than one revolution?

Using predefined ranges would mean we only need to have one limit switch and would simplify the task at hand considerably. However, in order to be certain not to exceed our limits, we want to implement hard limits on the yaw axis. If we used one limit switch for both hard limits, a seam would be formed during the extrusion of filament. This is caused by the consecutive triggering

of the switch when the trigger element is coming from the opposite direction which results in only partially completing one revolution.

To clarify this, I will give an example: whenever we trigger the switch from one side and then continue the movement until we reach the switch from the other side. A short distance will remain between the first and second position where the switch was triggered. This gap could result in a bad 3D printed model in 3D printing applications and is therefore the reason an overlap is required. This overlap can be achieved by being able to move further than one full revolution. Therefore, we also use two limit switches in our setup.

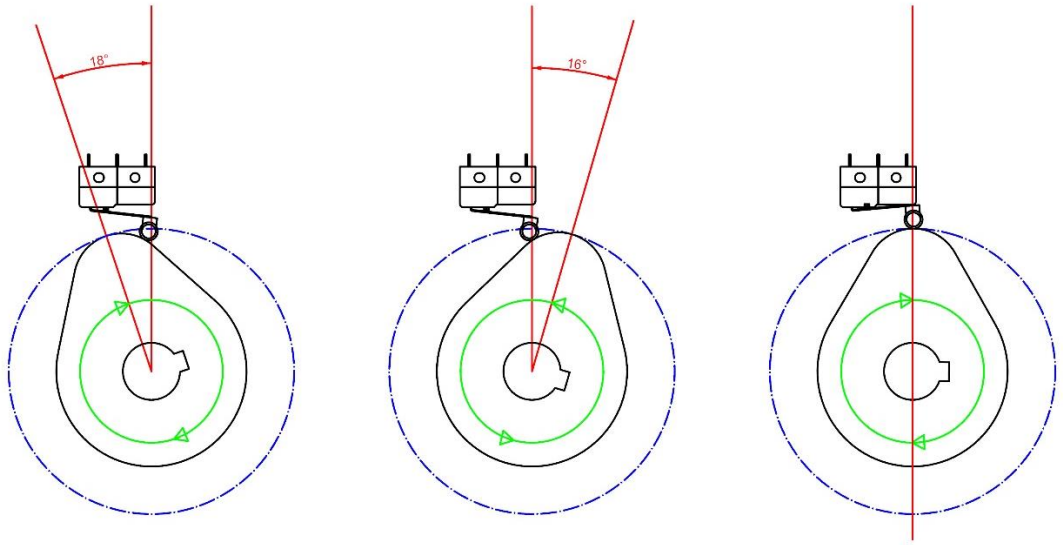

Figure 7 - Visualisation for the limit switch issue

3.2 Limitations of the CNC controller software

After experimenting with the PlanetCNC software, however, we came concluded that we were limited by the capabilities of the software. As a result, we needed to do a workaround to accomplish what was required. This is because the software does not provide any advanced options or possibilities to program custom features concerning the inputs for the limit switch. This is not satisfactory as we need to be able to determine the sequence in which the states of the switches change. A possible sequence would be to ignore one of the switches when it is triggered for the first time and activate the corresponding input of the CNC controller at the second time. Otherwise we would only do half a rotation for instance.

Depending on the degree of freedom which is desired for the yaw axis, we should be able to ignore multiple triggers of the switches. In order to achieve this, we came up with several ideas, for example, using two limit switches and a rotating cam which triggers the switches consecutively and use a microcontroller for programming the sequence. Something similar to this would be the use of optical switches and a disc with reference holes, instead of the regular switches and a rotating cylindrical cam. Another idea was to use a specially designed mechanical spiral which would make a linear motion and trigger, end switches on the upper and lower limit. However, this would have been too difficult to manufacture and therefore we chose the first

solution, since we could manufacture the cylindrical cam and use a microcontroller immediately. On top of that, I had worked with microcontrollers before.

3.4 Using a microcontroller to program the desired sequence

To express the program which is needed in a pseudocode we would have to increment or decrement a variable for each switch depending on which direction the axis rotates. Whenever the correct sequence is met, the inputs of the controller should receive a corresponding signal.

```
while (1)
    if (switch1 == true)
        if (Direction == true)
        Decrease counter1;
        if (Direction == false)
        Increase counter1;
    end
    if (switch2 == true)
        if (Direction == true)
        Decrease counter2;
        if (Direction == false)
        Increase counter2;
    end
    if (counter1 == -1 && counter2 == 0
        counter1 = 0;
        output1 = true;
        delay = 500ms;
        output1 = false;
    end
    if (counter1 == 1 && counter2 == 2)
        counter2 = 1;
        output2 = true;
        delay = 500ms;
        output2 = false;
    end
end
```

Figure 8 - Pseudocode for the sequence

Down below is an image which serves as a visualisation for the sequence which we tend to achieve. As for the code itself it also sets its value after reaching the limits to the value of the limit itself. Since we don't want to be able to count further than our limit. This value of course depends on the amount of rotations we want to be able to make.

Figure 9 - Sequence Visualisation

3.5 Multitasking capabilities of the microcontroller

In order to make it possible for the microcontroller to do other tasks a timer is used which starts an interrupt service routine every millisecond. This interrupt service routine is used to run a function which reads the input changes.

```
#include <avr/io.h>
#include "Timer0.h"

//global Variable
volatile unsigned int Timer0Counter = 0;

void InitTimer0()
{
    TCCR0 |= (1<<WGM01)|(0b100<<CS00);      // WGM01 is set to 1 so the timer is cleared on compare match mode
                                            // prescaler to 256 -> freq 31250
    OCR0=31;                                // freq 1000 -> tick 1 ms OCR0=31
    TIMSK = 1<<OCIE0;                       // Timer/Counter0 Output Compare Match Interrupt Enable
}
```

Figure 10 - Initialisation function for the timer

Therefore the advantage of working with interrupts is that we do not have to watch every input constantly but only on a certain interval, resulting in the microcontroller being able to do other things in the meantime. Since the microcontroller is to be used for managing the limit inputs of the CNC controller it is necessary to use its capabilities properly. Down below you can see the interrupt service routine with a timer variable which is incremented every millisecond and on top of that a function which performs the debounce of inputs every 10 ms.

```
// Called at about 1000Hz
ISR(TIMER0_COMP_vect)
{
    static unsigned int DebounceTimeMark;
    // Debounce buttons. debounce() is declared static inline
    // in debounce.h so we will not suffer from the added overhead
    // of a (external) function call
    Timer0Counter++;

    if (Timer0Counter - 10 > DebounceTimeMark)
    {
        DebounceTimeMark=Timer0Counter;
        debounce();
    }
}
```

Figure 11 - Interrupt service routine

In the main program, is another variable which is used as a separate counter for the stay alive function. Each time, this variable reaches its maximum value, the output for the indicator LED is toggled. This is done, to make sure the microprocessor is not being reset during operation.

```
//local variables
unsigned int StayAlive = 0;
unsigned int StayAlivePriode = 200;


// PORTC output
DDRC = 0xFF;

InitTimer0();
// Enable global interrupts
sei();

debounce_init();

while(1)
{
    if (Timer0Counter - StayAlivePriode > StayAlive)    //every 200 ms toggle stay alive output for a LED indicator
    {
        StayAlive = Timer0Counter;
        PORTC ^= (1<<PC2);
    }
```

Figure 12 - Function for the stay alive indicator

3.6 What is contact bounce and why is it a problem?

After some testing I realised there was a need to debounce the switches. Switch bouncing is a problem every switch faces no matter how well it is manufactured. Bouncing is caused by the mechanical parts in a switch. Whenever we activate a switch the metal contacts within the switch come together but only for a very short time. It then starts opening and closing several times and the time it is closed increases each time a contact is made until the switch is eventually fully closed. This bouncing we speak of happens in a very short time frame of just a few microseconds. It seems as if the activation was instant but a microcontroller is much faster and will be able to register multiple instances of the switch changing states. This means that the microcontroller receives multiple button or switch activations with a single button press or switch trigger.

If we look at an oscilloscope image of a switch being triggered a possible image could be the one below. Starting off in closed state, going through a period of bounce and in the end remain with a constant voltage over the switch.

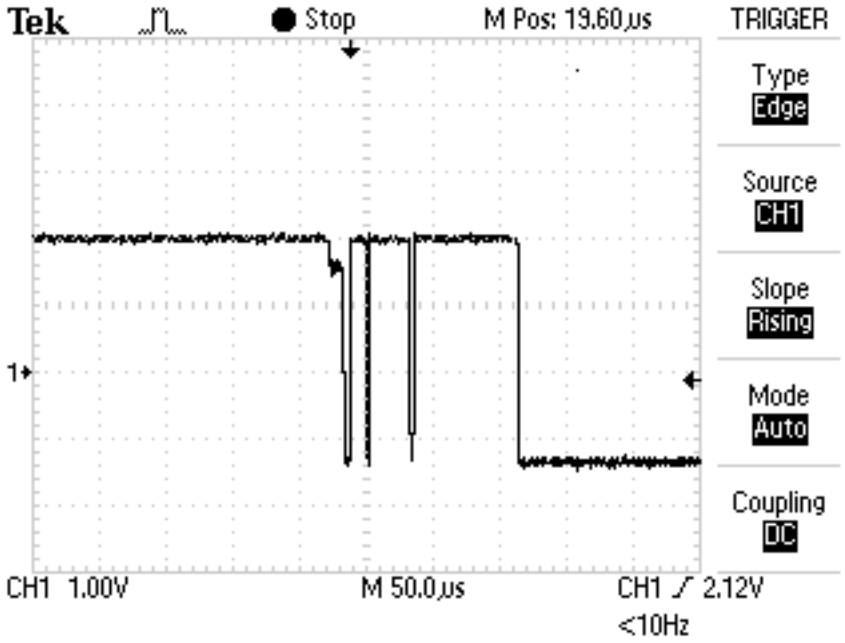

Figure 13 - Oscilloscope image of contact bounce from [4]

3.7 What is switch debouncing?

There are two common methods to resolve switch or contact bounce, for example, hardware debouncing with a capacitor or software debouncing.

Hardware debouncing is done by adding a capacitor to the circuit in order to achieve a passive low pass filter. A passive low pass filter is a circuit consisting of a resistor and a capacitor in series. Resistors, capacitors and inductors are considered passive components, therefore it is a passive low pass filter.

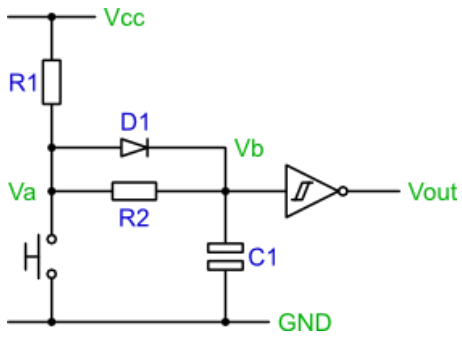

Figure 14 - Hardware debounce circuit from [4]

What the filter does is block out the high frequency signals produced by the bouncing of the switch. As for the frequency, for example, if the bouncing happens within 1ms the bouncing can be filtered out by filtering all the frequencies above 1khz. This will result in the switch giving us

a bounce free activation. However, for applications with a lot of switches it is not convenient and cost efficient to provide every switch with a condensator.

Thus a better solution would be a software debounce but it proves to be a slightly more difficult and time consuming than a hardware solution. Yet in the long run, it leads to a solution that is cheaper and saves space for hardware. To describe briefly what a software debounce does, it is using a principle where a certain delay is added to the input in order to overcome the contact bounce of the switch.

3.8 How to implement a software debounce?

The approach for software debounce can differ but the principle is quite similar. For example, we can read the input signals several times during a period and, depending on how many times we saw a change in the input, we can determine if the switch was triggered. The microcontroller that was chosen for the limit switch program would also be useful here.

For the debouncing of the inputs, I've followed a tutorial by Snigelen at an AVR forum and modified it to our needs [1]. This is because I didn't have any experience related to contact debouncing. Downbelow is the pseudocode for the debounce principle which is used.

```
if (time to check button)
    read current button state
    if (current button state != button state)
        increase counter
        if (counter >= 4)
            change state
            reset counter
        end
    else
        reset counter
    end
end
```

Figure 15 - Pseudocode debounce

3.9 Program description

First of all, I would like to refer to the tutorial of Snigelen [5], which I used for my program and he himself in his turn, based his code on the one from Peter Dannegger. I'll now give a short description about how the program functions.

To start off we have the debounce function in the interrupt service routine we discussed earlier. The debounce function polls each input and does this 4 times in a row. Furthermore, we use only three variables, one to keep track of the debounced button state, one for the current button state and one to determine if the button is pressed. This is because we only use one bit for each button in each variable which gives us the opportunity to give us 8 debounced inputs. In order

to check the state of all pins on a port and detect if there was a change we can simply find out by using the following line.

```
// Read buttons. Xor with button_state to see which ones are about to change state.
uint8_t state_changed = BUTTON_PIN ^ button_state;
```

By doing this, state_changed will contain zero for the bits that didn't change and a one for the bits that currently differ from the saved state.

The way the counting is done, is very interesting since we use vertical counting. By using only 2 bytes we have eight 2-bit variables which we can use for the counting. Therefore, we need to use some logical operations.

First of all, we want to be able to "decrease and set" or "increase and reset" each counter individually. Because we only want to change the value for the corresponding input. In the tutorial they choose to decrease the counter and set it back to binary 11 when reaching 0. The reason being we only need one and-operation to detect the set or beginning state. This operation is referred to as the "decrease or set" operation.

Next is a truth table where M stands for mask and indicates if the counter is being decreased or set to binary 11. Then, there is the most significant bit and the least significant bit for both the beginning situation, as the value after a reset or a decrease happened.

	Before		After	
M	H_b	L_b	H_a	L_a
0	0	0	1	1
0	0	1	1	1
0	1	0	1	1
0	1	1	1	1
1	0	0	1	1
1	0	1	0	0
1	1	0	0	1
1	1	1	1	0

Table 1 - Truth table "decrease or set"

What we can conclude out of the truth table is, that whenever M is zero, a reset to binary 11 is done. The counter is also set back to binary 11 whenever the counter reaches the binary value of zero.

```
#define VC_DEC_OR_SET(high, low, mask)          \
low = ~(low & mask);                            \
high = low ^ (high & mask)
```

Figure 16 - function definition for decrease and set

What is next is to update state_changed and make sure it only holds a 1 whenever the counter overflowed. We can tell this by looking at the conditions of state_changed because we have information about when a counter is being decreased and reset.

```
// Update state_changed to have a 1 only if the counter overflowed
state_changed &= vcount_low & vcount_high;
```

Figure 17 - Determine if overflowed

If we now have any ones left over then we know the counter overflowed and we then want to update button_state by using a XOR operation.

```
// Change button_state for the buttons who's counters rolled over
button_state ^= state_changed;
```

Figure 18 - Update button state

Lastly the buttons_down variable is updated with the changed states of the inputs where a button press is detected.

```
// Update button_down with buttons who's counters rolled over and who's state is 1 (pressed)
buttons_down |= button_state & state_changed;
```

Figure 19 - Update buttons pressed

By calling the function buttons_down we can check the input bits and perform actions with it. For further information or a more in dept clarification I would like to recommend you have a look into the debounce tutorial [5].

To continue with the main program, we start with initialising several functions, for example, the timer and the debounce function. As for the infinite while loop we have a function for the stay alive indicator and several if statements like in the pseudocode for the limit switch program. The complete programme code can be found within the attachment.

3.10 Problems concerning the microcontroller

During the process of implementing the microcontroller into the system, several problems occurred. But considering my time of stay which has almost come to an end, I'll not be able to solve every problem concerning the program and the microcontroller. However, we'll provide a proof of concept and suggest possible future solutions to resolve the issues at hand.

First of all, we have an issue with the microcontroller behaving in a weird way whenever we switch on the power supply for the stepper drivers. Therefore, we implemented a function in the program in order to indicate if the microcontroller was being reset in a way. It was clear to us that the power supply for the stepper drivers was causing interference in the peripherals. Since we noticed the interference on the monitor of the computer which we were using. This is why it might be necessary to place a filter between the net and the power supply for the stepper drivers.

By using the stay alive function, we were able to determine that the microcontroller was in fact being reset. On second thoughts, the reason might be that the supply voltage for the microcontroller needs to be filtered and stabilised and therefore causes the resets. The reason for thinking this, was because I inspected several connection diagrams which indicate a capacitor being used in the circuit supplying the microcontroller with power. On top of that one of my colleagues confirmed that this might be the case, since the capacitor in the circuit is being used to stabilise and buffer the supply voltage.

Secondly, we increased the frequency of the timer which is used for the interrupt service routine. We did this since the polling of the inputs may be done quicker and also for the

implementation of the stay alive function. However, because of an error in the main program, the program stopped working. This is because a part of the main program responsible for the outputs of the microcontroller is not being executed anymore. Therefore, we have to modify the program.

On top of that, we realised that, there is also an issue with the sequence of the program being lost. This happens whenever we're still on top of one of the switches and then reverse the direction of the yaw-axis. To resolve this, we have to make it so that the switches are activated on release with for example a falling edge detection. What needs to be considered however is the starting position and the way the homing procedure is done. That is because we can already foresee another problem concerning the hard limits of the program.

To go further into detail, whenever we would do the homing procedure the limit switch used as a reference will be approached and activate once it is being released. After this the axis will reverse in direction and continues to position itself in its starting position. Which would mean that it would run into the switch again whereupon the switch would interrupt the homing procedure when it is released for the second time.

Lastly, we're also not able to detect a break or fault in the cable coming from the switches to the microcontroller which would make the use of NC-switches meaningless. The reason being, that at this moment the microcontroller is using internal pull-up resistors with NC-switches. For that reason, we have to implement external pull-up resistors instead, so we're able to detect if the cable is faulty.

3.11 Future modifications to the program and microcontroller

What needs to be done is rethink the program once more for the reasons given above and to make sure the microcontroller is behaving correctly. To quickly summarize the tasks at hand, we first need to modify the program so that the main program is executed properly. Next, we have to resolve the issue around losing the sequence of the limit switch program. Following up with adding a capacitor to the circuit powering the microcontroller and making sure it resolved the issue at hand. To finish it off, we have to be able to detect cable breaks through the microcontroller.

Chapter 4: Configuring the axes in the PlanetCNC software

The configuration of the axis is relatively easy and the procedure on how to do it can be found within the tutorials provided by PlanetCNC. As for us we'll be describing the procedures which are important for our system and how we made the calculations.

4.1 Configuring the step per unit values for our system

Step per unit (SPU) values would translate to the amount of step pulses that are needed by the controller in order to move the corresponding axis of the machine over the distance of one unit. Which unit system is used is free to choose so for our system we'll work with millimetres.

To start off there are a few parameters which are of importance. The SPU value depends on the stepper motors themselves, the configuration of the stepper motor drivers and the displacement which is delivered by a pulley or leadscrew. This is sometimes referred to as the pitch or lead depending on which element is used.

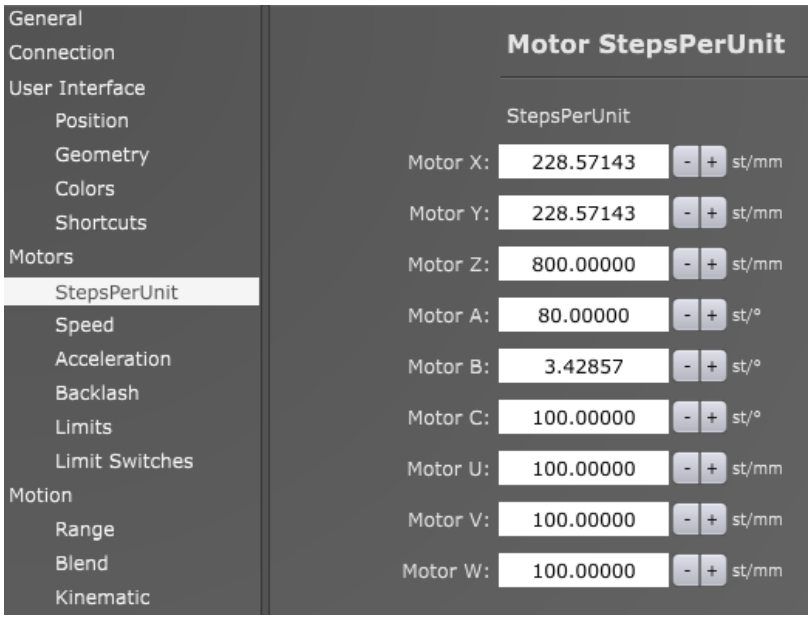

Figure 20 - SPU configuration in PlanetCNC

4.1.1 The parameter for the motor

As for the motors, steppers often have 200 or 400 full steps per revolution of the rotor. This would mean that one step results in a displacement of 1.8° or 0.9° depending on the number of steps per revolution as given above. The motor parameter is referred to as M in the documentation of PlanetCNC.

4.1.2 The parameter for the step size

With stepper motors it is possible to increase the number of steps the motor does per revolution. This can be achieved by several excitation methods for example normal mode also called full step, after that comes half step and last but not least micro stepping. The importance of step size is that the resolution can be increased and therefore the motors will run smoother. This does however also mean that the amount of torque which will remain will decrease substantially. This is because the torque falls of dramatically for each increase in number of micro-steps per full step. Stepping size is referred to as S within the formula provided by PlanetCNC.

4.1.3 The displacement depending on the transmission principle

To know the SPU value, it is important to know the displacement which the transmission system produces with one revolution. Whenever we use pulleys, leadscrews and or other gears we want to make sure we have the specifications of the elements. The parameter which stands for the displacement, pitch or lead depending on the transmission is referred to the letter P in the formula.

4.1.4 Calculating the SPU value

When all the parameters are known than we just have to fill in the formula given below.

$$\mathrm{SPU} = \frac{(\mathrm{M} * \mathrm{S})}{P}$$

If we however don't know the value for the displacement we could also configure the software with a standard value. Which would be 200 SPU depending on the motor for instance. After that we'll be able to translate the corresponding axis over a certain distance in metric units for instance. By measuring the resulting distance with a measuring tool. We then can continue to calculate the value with the following formula.

$$\text{Correct SPU value} = \frac{(\text{Current SPU value} * \text{Entered distance value})}{\text{Measured distance value}}$$

If the SPU value is still not accurate you have to repeat the procedure several times.

4.1.4.1 Calculation based on the parameters we have

For the X- and Y-axis we're using the same set of transmission elements. A toothed belt and pulley with 14 teeth and a pitch of 2 mm in between the teeth. When we fill in the formula accordingly, it results in SPU value of approximately 228.57 steps per mm.

$$SPU = \frac{200 * 32}{14 * 2} = 228.571\ st/mm$$

The Z-axis uses a lead screw and has a displacement of 8mm for one revolution therefore the SPU value is as follows.

$$SPU = \frac{200 * 32}{8} = 800\ st/mm$$

As for the SPU value for the rotary joints the transmission ratio of the gear setup had to be considered. Since it uses cog wheels and cogged belts. Calculating the SPU values goes as follows, first of all our steppers have a step size of 1.8 degrees per step, this is important in order to calculate how many steps is equal to one degree. Further division of 1.8° by 32 is needed because of the micro stepping factor, this results in 0.05625 degrees per micro step.

$$degrees\ per\ microstep = \frac{1.8°}{32\mu st} = 0.05625°$$

If we then take one over this number, we receive a value which represents the number of micro steps per degree.

$$microsteps\ per\ degree = \frac{1}{0.05625°} \cong 17.77\ \ldots$$

Transmission ratio is calculated as follows.

$$gear\ ratio = \frac{rotations\ of\ the\ driven\ gear}{rotations\ of\ the\ driver\ gear} = \frac{Z^2}{Z_1}$$

$$i_1 = \frac{40}{21} \approx 1.91$$

$$i_2 = \frac{72}{16} = 4.5$$

Afterwards we just have to factor the transmission ratio and then we'll have the right SPU value for the rotary joints.

$$SPU = 17.78 * 1.91 \cong 33.862\ \mu st/°$$

$$SPU = 17.78 * \frac{72}{16} \cong 80\ \mu st/°$$

4.2 Configure the limit switch inputs in the CNC software

It is clear that we intend to use limit switches as a safety precaution but for the majority of the axis we only use one switch which serves as a reference for the homing procedure. Therefore, the programmed ranges will be used to prevent the system from crashing.

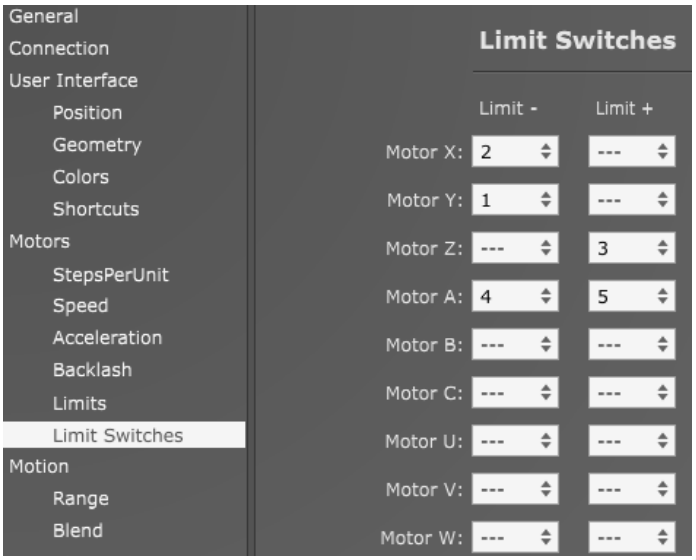

Figure 21 - Single or double input configuration in PlanetCNC

4.3 Set the motor limits in the software

The motor can be restricted in its movement by configuring the motor limits in the controller software. A motor limit can be assigned by filling in the desired value and tick the radio button next to it. Because there is only one limit switch per axis for the majority of the axes we have to slightly adapt the procedure given at the tutorial of PlanetCNC.

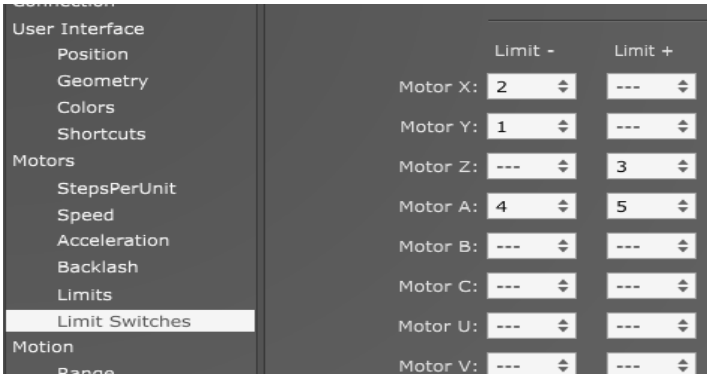

Figure 22 - Motion limit configurating in PlanetCNC

We start the procedure in a similar way by approaching and activating a switch with the corresponding axis. If the switch is then triggered, we can write down the coordinates from the software and fill in the value in either the negative or positive motor limit according to the implementation of the switch. Following up by moving in the opposite direction. However, this step is different because we do not have a switch at the opposite side, therefore we carefully move until the desired position is reached. We then write this position down and apply the value to the corresponding limit.

4.4 Set the motion limits in the software

By configuring the motion ranges we can restrict the movement of the effector. Therefore, the difference between motor and motion limits is, that we can either limit the actual movement of the motors or limit the movement of the tool that is used. However, since we have a multi-axis machine we cannot simply use the same values as in the motor limits. This is because we are not following the same kinematic rules as a XYZ-axis machine. Configuring the motion ranges is similar to the procedure used for the motor limits.

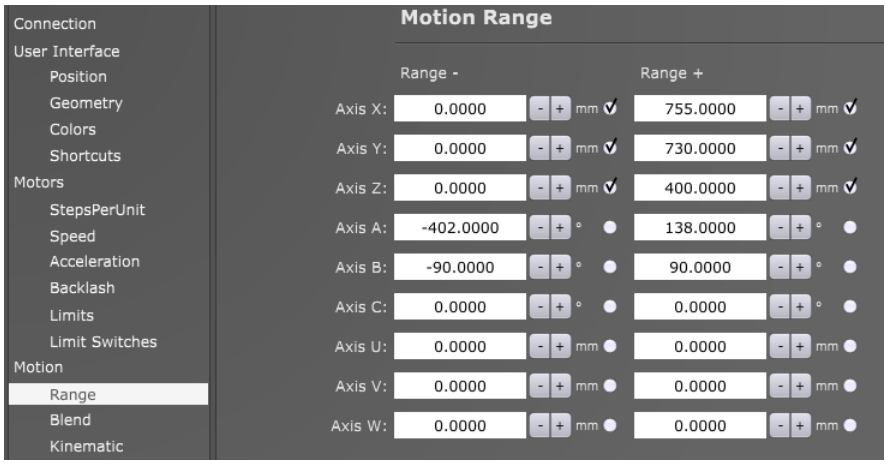

Figure 23 - Motion range configuration in PlanetCNC

4.5 The homing procedure

After doing all the previous procedures we continue with the homing procedure. First, we configure the homing sequence by simply selecting a number for each axis according to the desired sequence. Secondly, we have to determine in which direction the homing is done for each axis independently. Followed by the positions which the axes of the machine should return to after reaching the corresponding switch. These are assigned by inserting the values which the axes should return to. Lastly, the switch hysteresis is the displacement the machine has to make in order to release the switch after activation.

Figure 24 - Homing configuration in PlanetCNC

For further information about setting up a CNC machine in the PlanetCNC TNG software, I would like to recommend you the tutorial provided by PlanetCNC.

Chapter 5: Implementation of the electronics

Stepper motors have been chosen for the driving elements of the system. Therefore, the whole system is designed around this type of motor. Stepper motors are a great solution for this project because they are easy to use and are a brushless type of DC motor. Since stepper motors are used we also chose for a CNC USB controller from PlanetCNC which will control the drivers for the stepper motors. In addition to these components, external power supplies, one for the drivers and one for the controller, custom made adapters for managing the connections and a microcontroller will be used.

The layout for the location of the electrical components and a wiring example can be found in the attachments of this thesis. See figure 53 and 54.

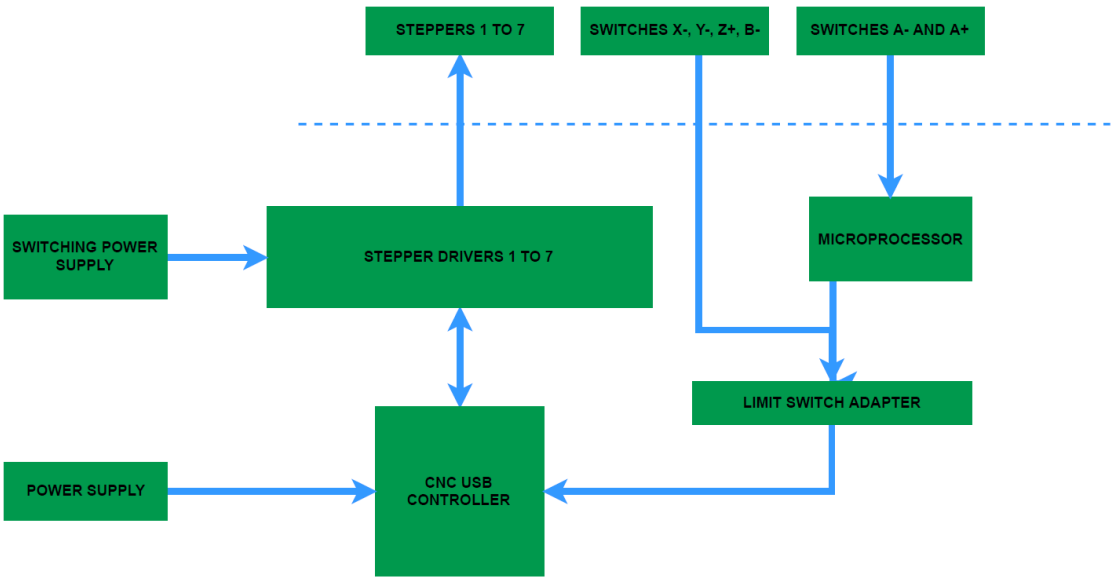

Figure 25 - Connection diagram

5.1 The stepper motors and drivers

In total we have 6 Nema stepper motors, 5 of the frame size 17 and only one of size 23. Each one powered by their corresponding driver which demands a sourcing voltage of 9 up to 40V DC. Therefore, we chose an external power supply able to deliver 12.5 amperes with a voltage of 24 Volts. With this power supply we'll be capable of powering 7 stepper drivers. For further specifications for the power supply, have a glimpse in the attachment.

Figure 26 - Nema 17 stepper motor image from RepRap [6]

Specifications for the stepper motors							
Model	**Holding Torque**	**Rated voltage**	**Shaft**	**Step angle**	**Motor length**	**Rated current**	**Inductance**
17HS4417	40.0 Ncm	2.6 V	Ø 5 mm single	1.8°	40 mm	1.7 A	2.8 mH
SH05601020	100 Ncm	3.6 V	Ø 6.35 mm single	1.8°	47,14 mm	2.0 A	n/a

Table 2 - Stepper motor specifications

With these specifications we estimated the power supply which is necessary to supply the drivers of current.

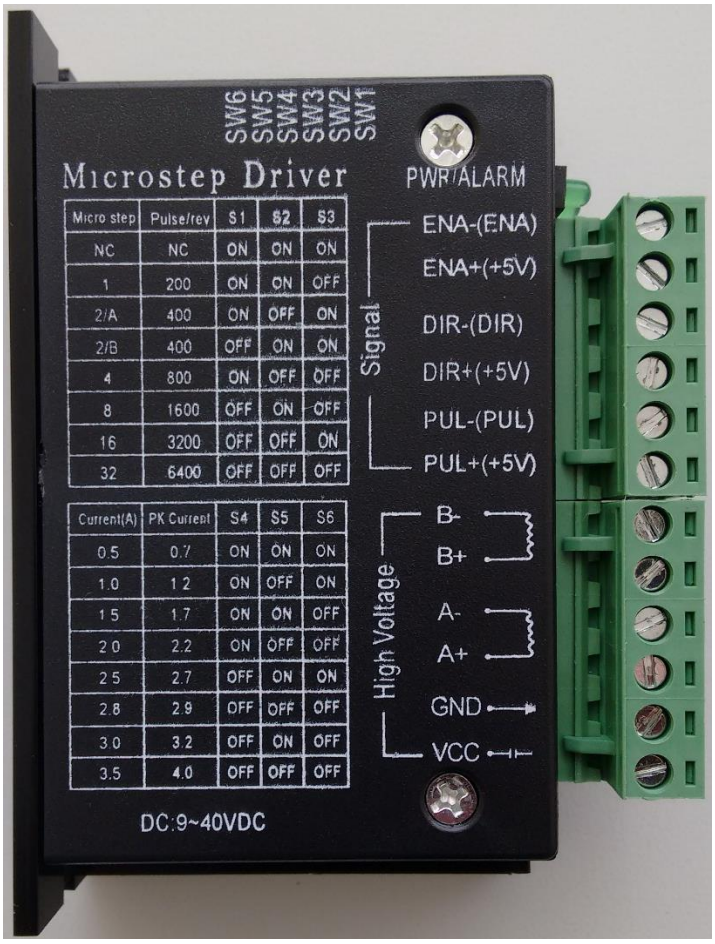

Figure 27 - Picture from one of our stepper drivers

5.2 CNC USB controller

The CNC USB controller of PlanetCNC is a link between a personal computer and the motor drivers supporting step and direction control. Since it provides us with a fully integrated hard- and software solution, additional software is not required.

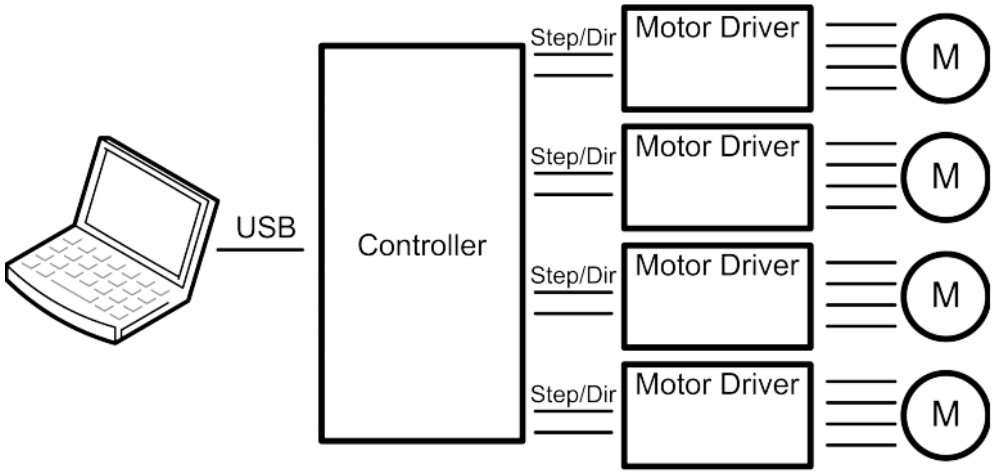

Figure 28 - Link between PC and controller by PlanetCNC [7]

A connection between personal computer and controller can be established, simply by using a USB-cable and opening the connection settings in the software. Next, we have to make sure the controller is selected from the drop list and a valid license for the controller at hand must be assigned to the software.

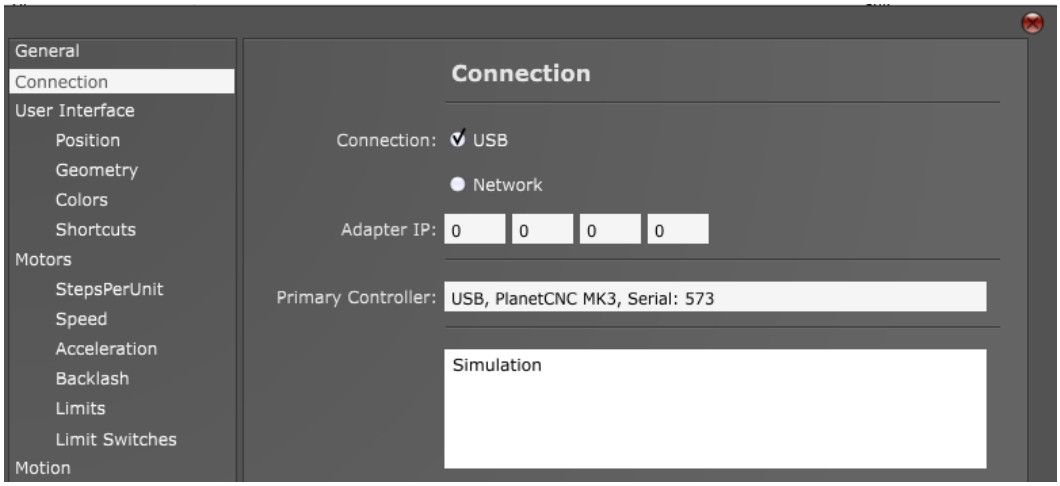

Figure 29 - Connection settings in PlanetCNC

Furthermore, a network connection is also available but will not be addressed in this thesis. However, if necessary please have a visit at the PlanetCNC blog.

The important connections we use on the PCB are of course the sockets for the drivers, the limits, the auxiliaries and the supply power input. As shown in figure 54 which shows us a wiring example, we can see that the controller uses a different external power supply from the drivers which should be able to provide 8 – 24V DC and at least 200mA. Secondly, the auxiliary socket is used to supply the microcontroller with a supply voltage of 5V DC.

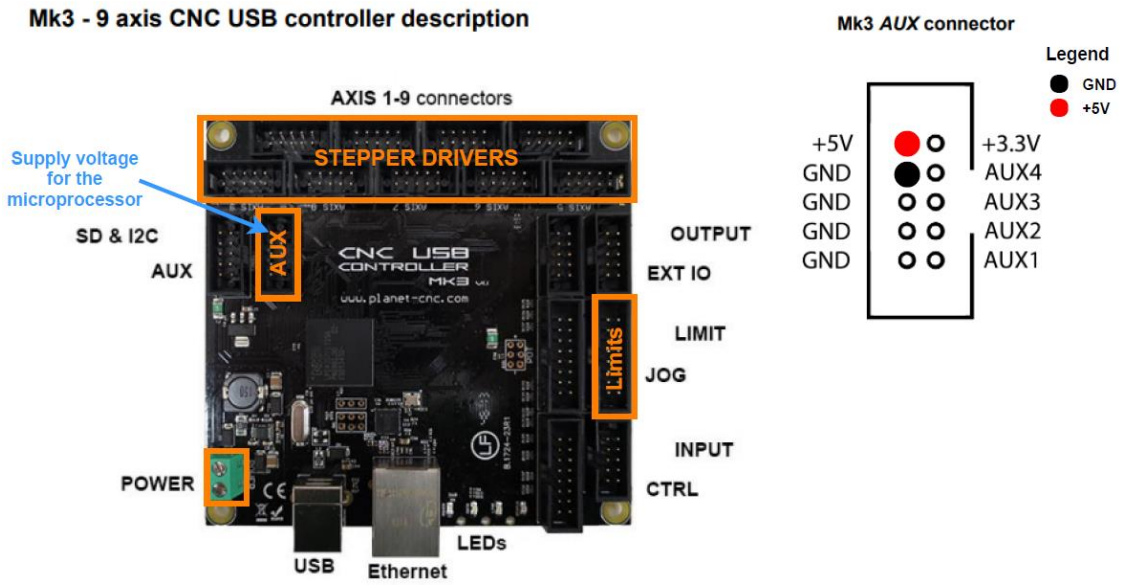

Figure 30 - CNC USB controller from PlanetCNC

5.3 The microcontroller

For the moment we're using a barebone Atmega32 microcontroller for the limit switch program intended for the yaw-axis. Therefore, we use a temporary breadboard to make some quick and easy connection but, in the end, the best solution would be a dedicated manufactured circuit board. But before we're able to create one we have to make sure we have implemented all the peripherals we need.

Furthermore, the inputs and outputs are freely programmable, the reason they are on different ports is because a previous PCB we used restricted the setup. Future inputs and outputs are best to be set on one port, for example on PORTA. The reason for not doing this already is that we consider using a different controller for example the Arduino Nano.

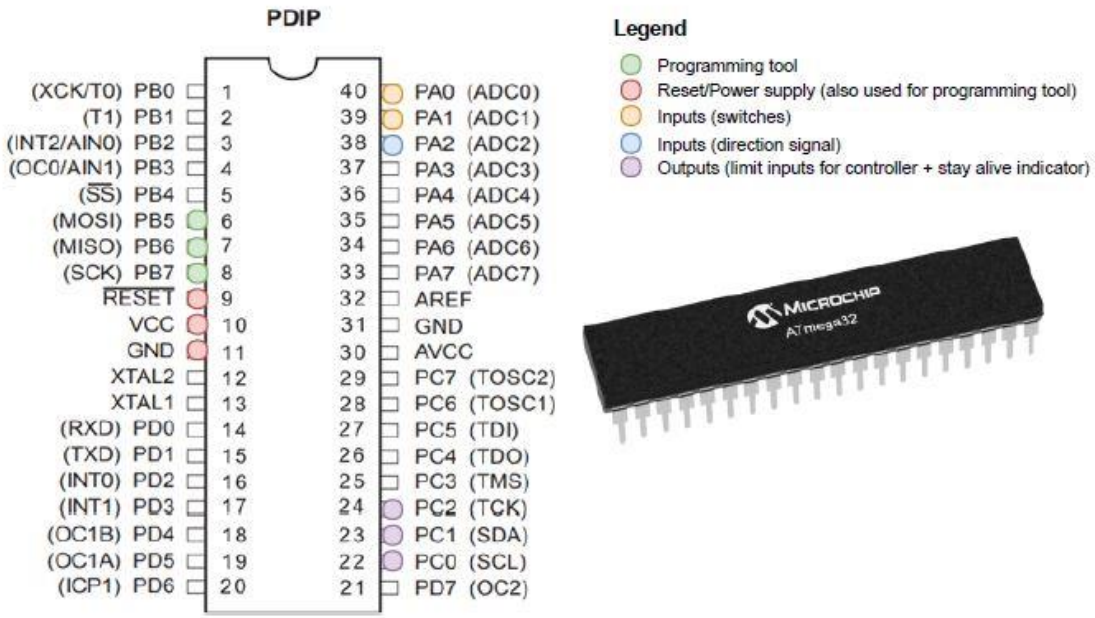

Figure 31 - Microcontroller configuration

The following image is to represent the wiring and the connections going into or coming from the microcontroller. For instance, we have the signals from the limit switch setup of the yaw-axis, together with the direction signal of the corresponding driver coming into the microcontroller. Next up, are the outputs which we send to the CNC USB controller and in the end, we'll also use another output for a stay alive indicator. To make the microcontroller accessible for a serial programmer, we also use the serial interface of the microcontroller. Lastly, we have to provide the microcontroller with some other components to secure a stable power supply and other functions for instance to reset the device. Figure in the attachment represents a connection diagram for the limit switches and the microcontroller.

5.4 Adapters, connectors and cable management

To make the implementation of the electronics easier and more convenient, we designed several adapters. First of all, we have made a circuit board for connecting the stepper drivers to the CNC USB controller. Something to note is that depending on the type of driver and the wiring configuration we don't need the +5V and Error connection. For more info about these driver configurations I would like to refer to the PlanetCNC blog.

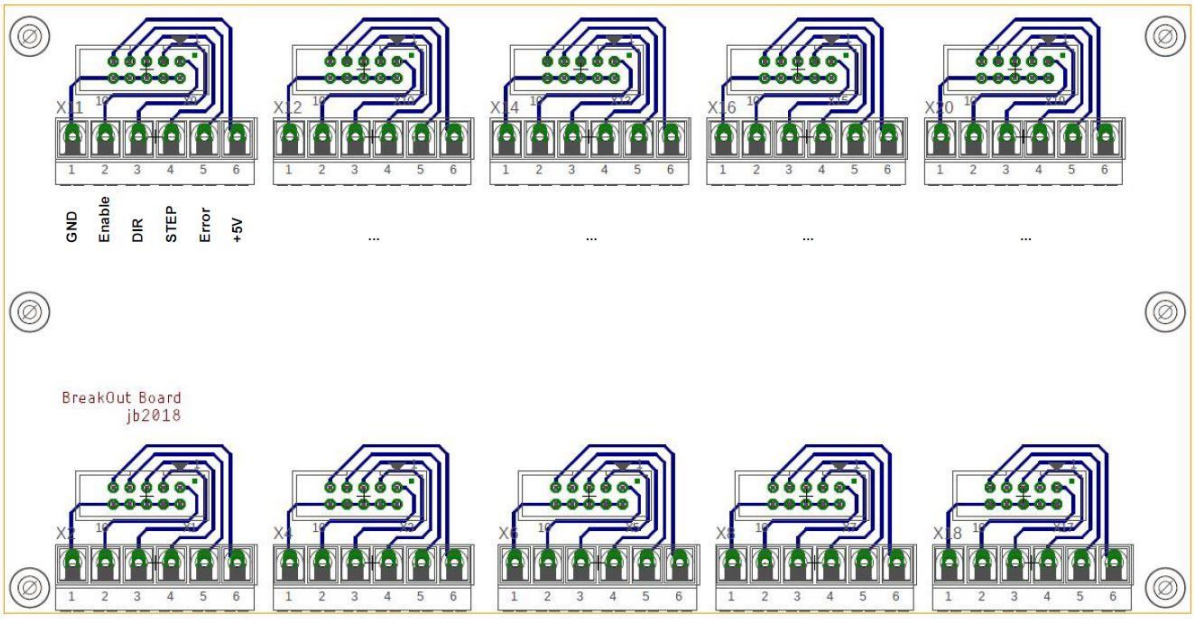

Figure 32 - Breakout for the stepper driver adapter

The most important connections for the drivers which we use are enable, step, direction and ground. Therefore, we designed the layout to have these four connections coming first in the terminal. With the 8-pin PCB socket, we can easily make a connection from controller to adapter.

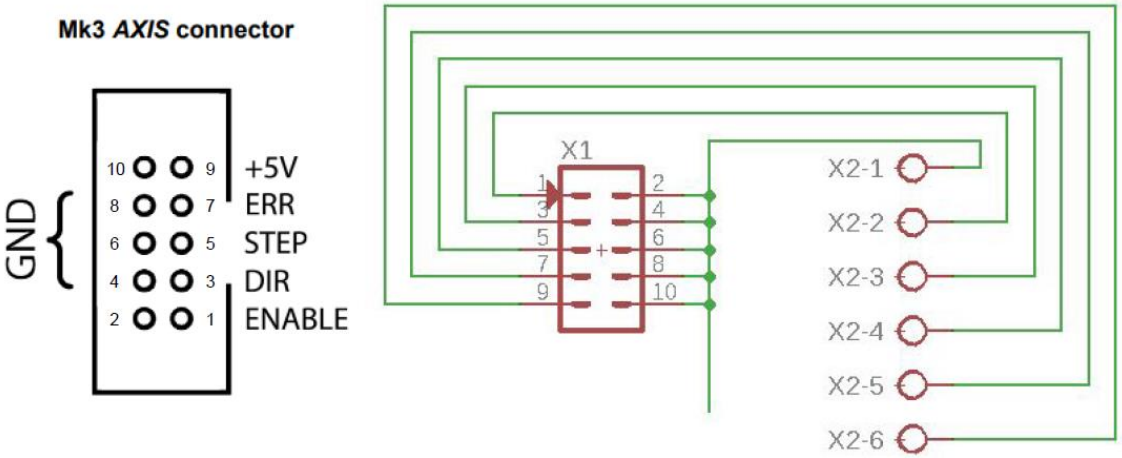

Figure 33 - Socket diagram for the driver adapter

Secondly, we created a circuit board for connecting the limit inputs in an easy manner. Therefore, it contains a 16-pin socket similar to the one from the CNC controller and screw terminals.

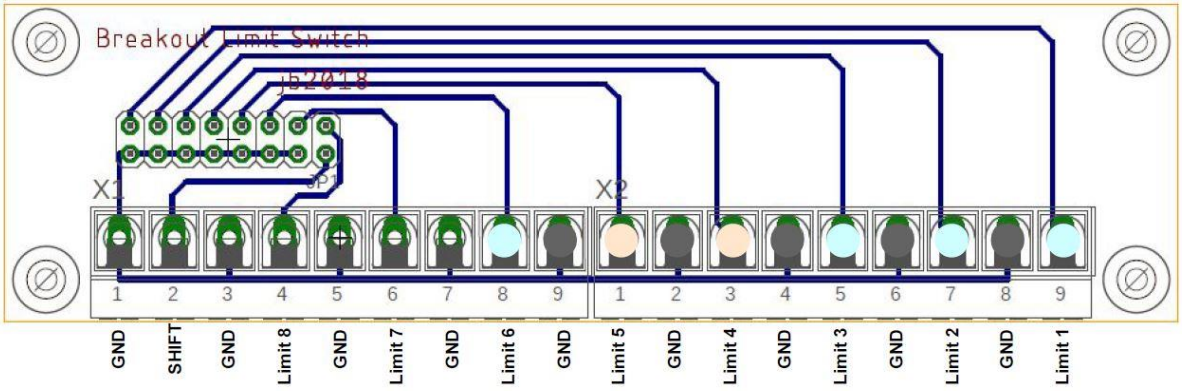

Figure 34 - Breakout for the limit switch adapter

In our configuration we only use up to 6 limit switches, the ones marked by a pink dot are coming from the microcontroller.

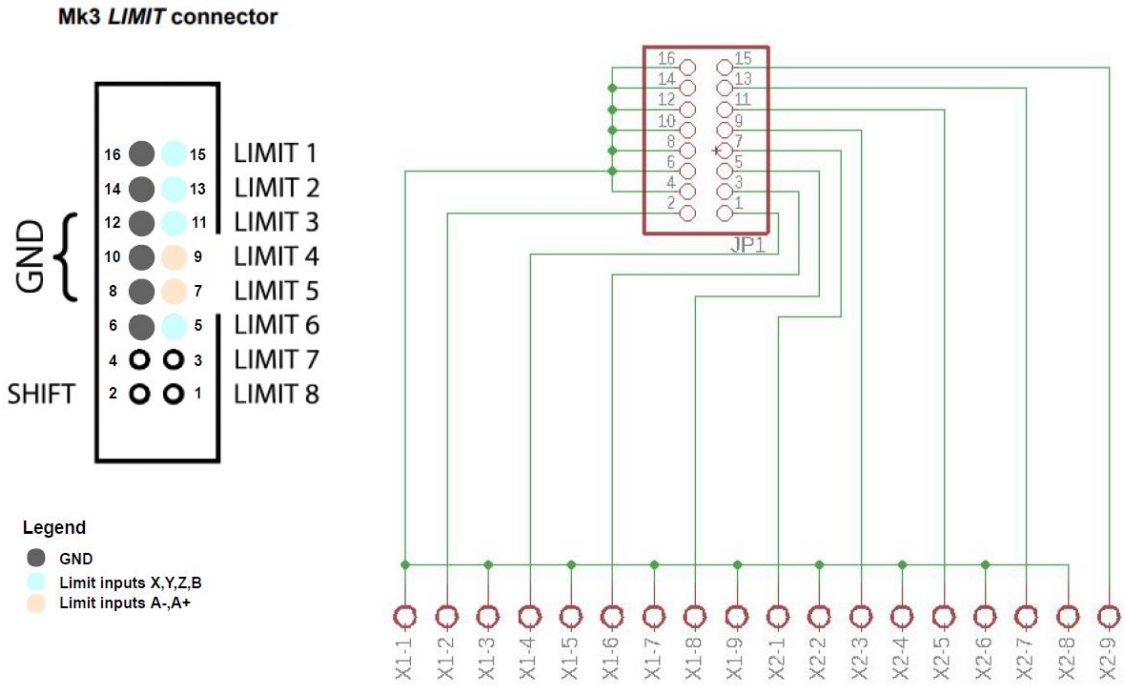

Figure 35 - Socket diagram for the limit switch adapter

Something to consider is, that we do not want to create daisy-chains in the power distribution to the stepper drivers. This would mean that we supply the next driver coming from the previous drive, instead of connecting each drive directly to one common point. Therefore, we still need to create a rail terminal for distributing the power evenly over the drivers in a star configuration.

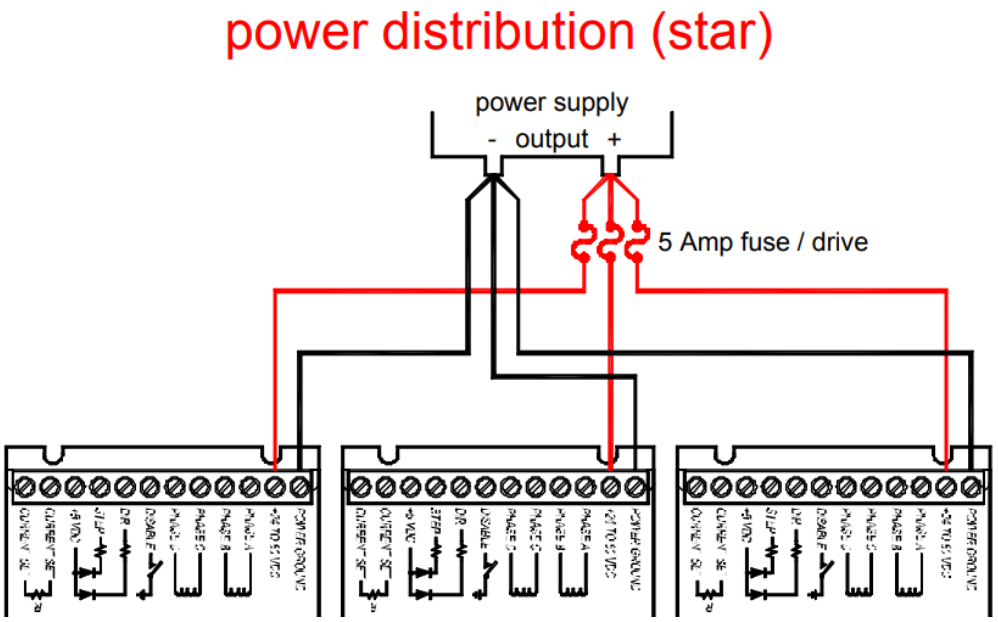

Figure 36 - Power distribution (star), image from PlanetCNC [7]

Because the power supply doesn't have enough contacts we have to provide a custom-made rail terminal to make the star configuration possible.

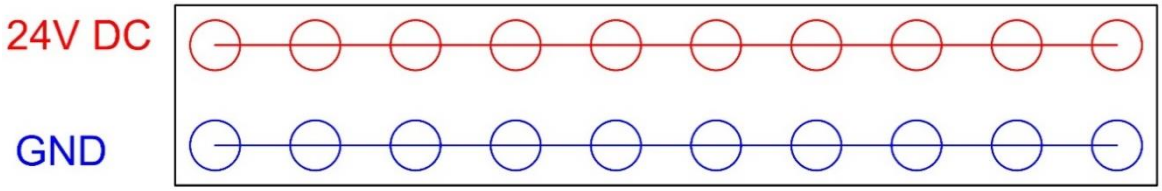

Figure 37 - Rail terminal sketch for wiring the drivers in a star configuration

Secondly, we have to make an extra adapter for conveniently connecting the microcontroller to the controller and the limit switches. Since we possibly want to use an Arduino instead of a barebone microcontroller, we do not know for certain which connections are needed. Therefore, we couldn't design this adapter yet.

Chapter 6: conceptualize 3D scanning

3D scanning is a major part of this thesis since we want to be able to repair or reconstruct parts and adapt to situations by recognising objects. The reason for this, is of course to achieve a system which is flexible and innovative in manufacturing processes. Besides that, in the case of the modular cleaning installation, object recognition will be the key in order to automate the process.

Since my colleague was already working on the scanning software we first implemented an XBOX Kinect camera to test the concept. Therefore, we made a small program with G-code instructions to move the effector around in the workspace, in order to perform a scan of an object.

The following image is from the 3D scan which we performed. The result of the scanning procedure was quite good considering the use of a low-end camera which was not designed for this purpose. In the attachment at page 53, you can find an image with the reconstruction of the object.

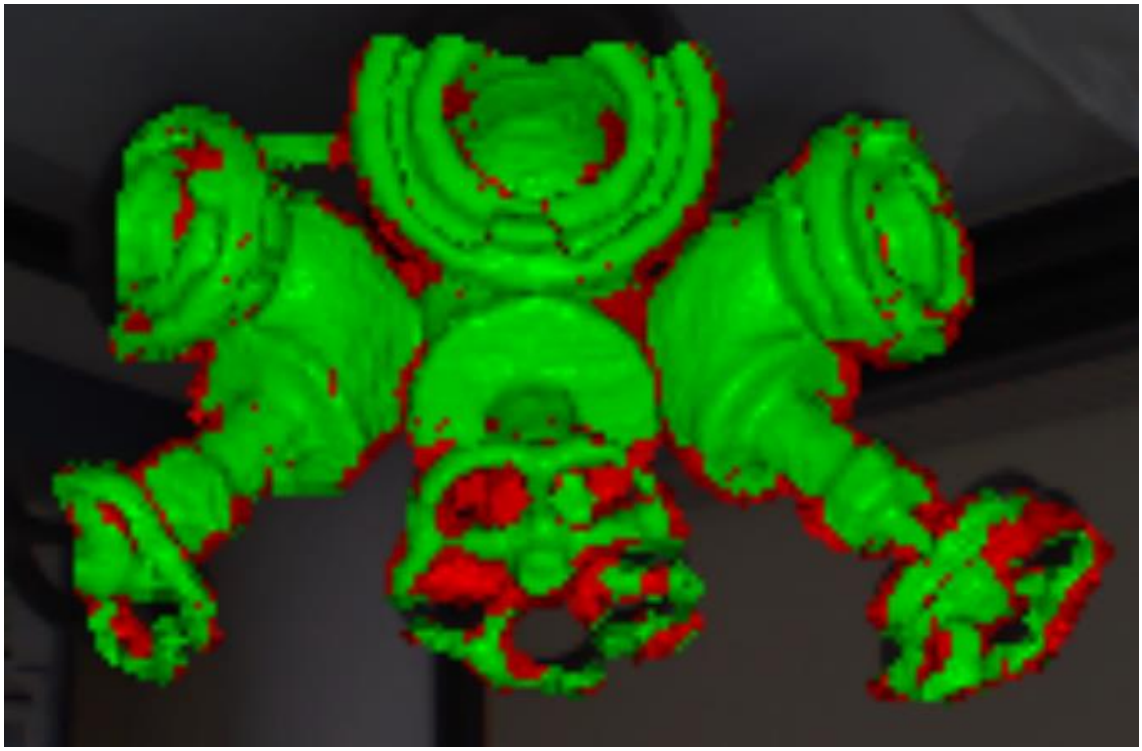

Figure 38 - 3D scan of an object

Conclusion

The objective for my thesis as mentioned before is the creation of a prototype which proves to be meaningful by doing research in a wide range of applications. Besides that, we have this collaboration with another company in order to build a modular industrial cleaning installation. Therefore, we mainly focussed on additive manufacturing by means of 3D printing and also 3D scanning. Since the procedures which are needed would be similar.

During the assembly of the system, we ran into several problems and therefore the mechanical design proved to be time consuming and one of the more challenging tasks throughout the project. Because we were limited in the manufacturing we could do ourselves we had to consider manufactured building parts which we could modify to our needs. This is why we designed the system around the OpenBuilds kit. However, this also happened to limit the accuracy, stiffness and the model of the system in general. Nevertheless, it is a prototype and a learning experience which we can build on for future designs in order to finalise the machine.

As for the implementation of the limit switch program we were not able to succeed. Since my internship was coming to an end and the delay we built up during the assembly, there was too little time left over to experiment with the microcontroller. However, we provided a description for the future modifications which have to be done to the program and the microcontroller setup. Besides that, we can foresee another problem being created considering the loss of the sequence, after the changes are made. This is why we have to rethink the main program and modify it to our needs.

Next, we experienced our equipment behaving in an unusual way and tried to locate the problem. We were able to point out that several components were susceptible to interference from either the steppers, the power supply for the drivers, noise or a combination of these. Therefore, we consider improving the system by first making sure the microcontroller has an adequate circuitry which provides a stable power supply. Following up, we could add a filter to the power supply to minimize the harmonic distortion and on top of that make sure every cable is shielded properly.

For the same reasons for not being able to finish the implementation of the microcontroller, we did not succeed in proving the concept of 3D printing. However, after putting the system together, we prioritised the application of 3D scanning in order to proof the concept of our machine. By attaching a 3D scanner as an effector and demonstrate the procedure for scanning an object with a programme in G-code.

To conclude this thesis, we were able to create a representation of a 5-axis machine and perform a demonstration of a 3D scan. The project is however not yet finished and will need to be continued in order to resolve the remaining issues and to be able to research into the topic of additive manufacturing. Therefore, I described the problems at hand and procedures we thought about in order to solve them.

Bibliography

[1] “OpenBuilds ACRO 55 20" x 20",” [Online]. Available: https://openbuildspartstore.com/openbuilds-acro-55-20-x-20/. [Accessed 2018].

[2] “Build List | OpenBuilds,” [Online]. Available: https://openbuilds.com/. [Accessed 2018].

[3] R. Sanmugan, “What is the backlash in gears? How is it advantageous or disadvantageous?,” Amrita University, 27 May 2016. [Online]. Available: https://www.quora.com/What-is-the-backlash-in-gears-How-is-it-advantageous-or-disadvantageous. [Accessed 2018].

[4] A. Greensted, “Switch Debouncing - The Lab Book Pages,” 17 June 2010. [Online]. Available: http://www.labbookpages.co.uk/electronics/debounce.html. [Accessed 2018].

[5] Snigelen, “[TUT][SOFT][HARD] Button debouncing in software,” 5 February 2015. [Online]. Available: https://www.avrfreaks.net/sites/default/files/forum_attachments/debounce.pdf. [Accessed 2018].

[6] “NEMA 17 Stepper motor - RepRap,” RepRap, [Online]. Available: https://reprap.org/wiki/NEMA_17_Stepper_motor. [Accessed 2018].

[7] “Planet CNC - CNC USB Controllers,” [Online]. Available: https://planet-cnc.com/. [Accessed 2018].

[8] “ATmega32 - 8-bit AVR Microcontrollers - Microcontrollers and Processors,” 2018. [Online]. Available: https://www.microchip.com/wwwproducts/en/ATmega32.

[9] M. Kuhn and P. Van Gansen, “Leidraad bij het schrijven van een scriptie,” 2017.

[10] “Master thesis acknowledgement template | Acknowledgment sample,” [Online]. Available: https://acknowledgementsample.com/master-thesis-acknowledgement-template/. [Accessed 2018].

[11] Ganssle, Jack;, “Debouncing Contacts and Switches in Embedded Systems,” The ganssle group, March 2014. [Online]. Available: http://www.ganssle.com/debouncing.htm. [Accessed 2018].

[12] Christoffersen, Jens;, “Switch Bounce and How to Deal with It,” 3 September 2015. [Online]. Available: https://www.allaboutcircuits.com/technical-articles/switch-bounce-how-to-deal-with-it/. [Accessed 2018].

[13] “What is 3D printing? How does a 3D printer work? Learn 3D printing,” [Online]. Available: https://3dprinting.com/what-is-3d-printing/.

[14] H. Bensoussan, “The History of 3D Printing: From the 80s to Today,” 14 December 2016. [Online]. Available: https://www.sculpteo.com/blog/2016/12/14/the-history-of-3d-printing-3d-printing-technologies-from-the-80s-to-today/. [Accessed 2018].

[15] “IEEE Xplore Digital Library,” Institude of Electrical and Electronics Engineerings, [Online]. Available: https://ieeexplore.ieee.org/Xplore/home.jsp. [Accessed June 2018].

[16] “Passive Low Pass Filter,” Aspencore LLC, [Online]. Available: https://www.electronics-tutorials.ws/. [Accessed 2018].

[17] “Switch Bounce and Other Dirty Little Secrets,” Maxim Integrated Products, 1 September 2000. [Online]. Available: https://www.maximintegrated.com/en/app-notes/index.mvp/id/287. [Accessed 2018].

Attachments

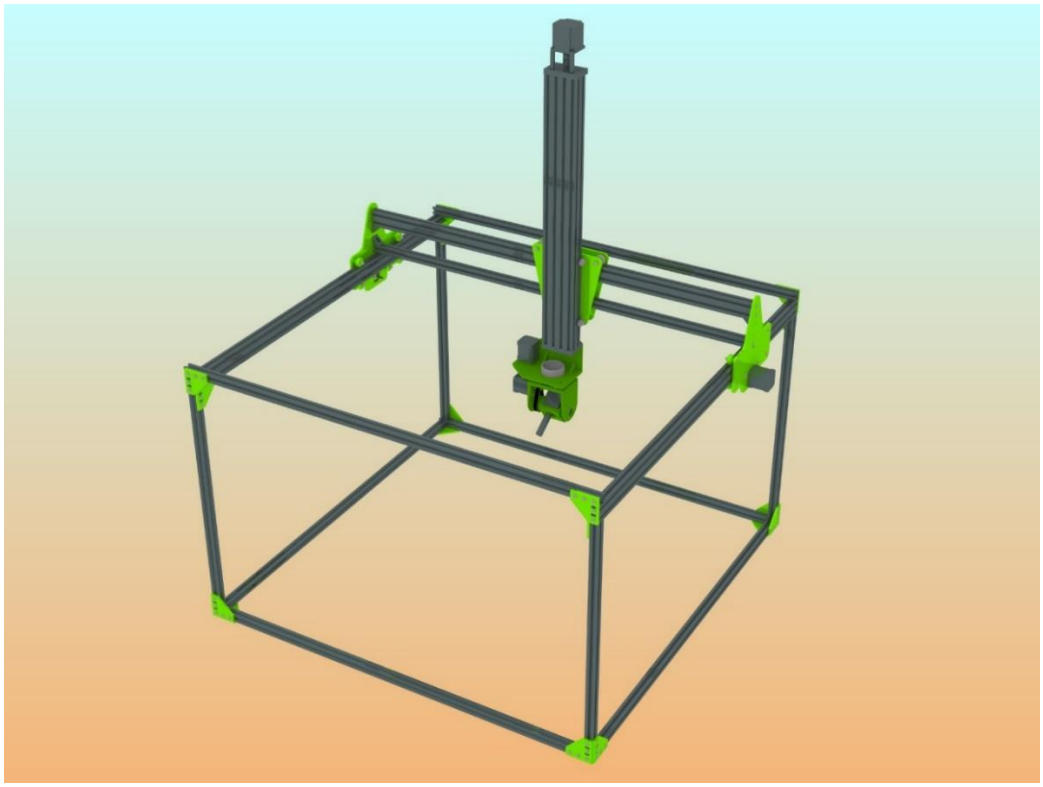

Figure 39 - Render of assembly 1

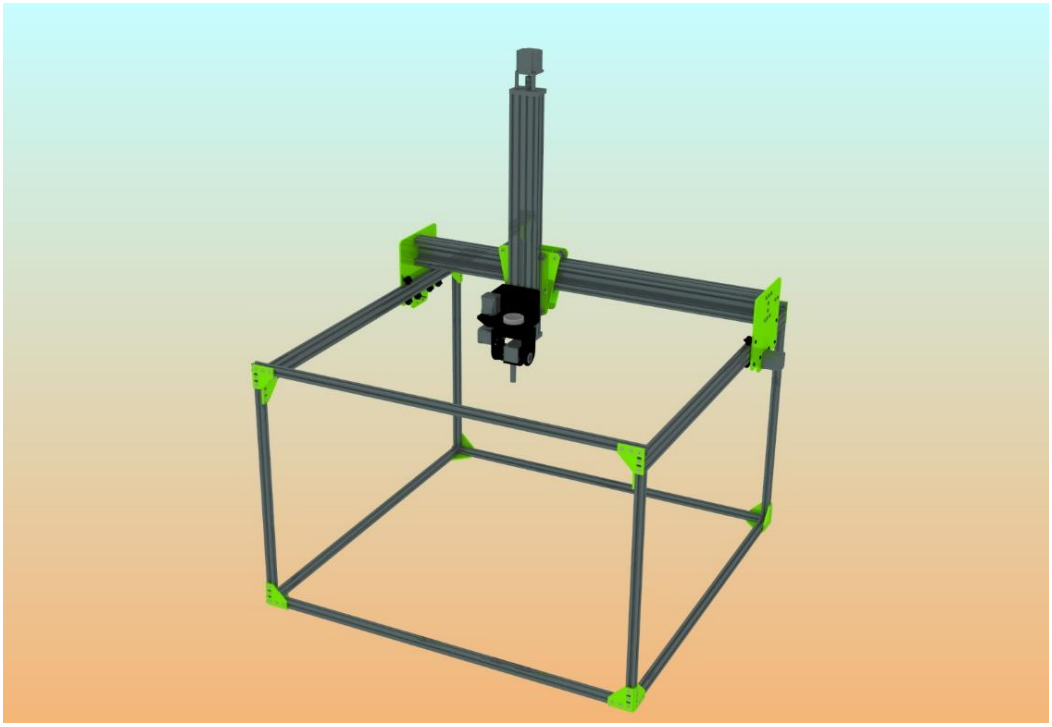

Figure 40 - Render of assembly 2

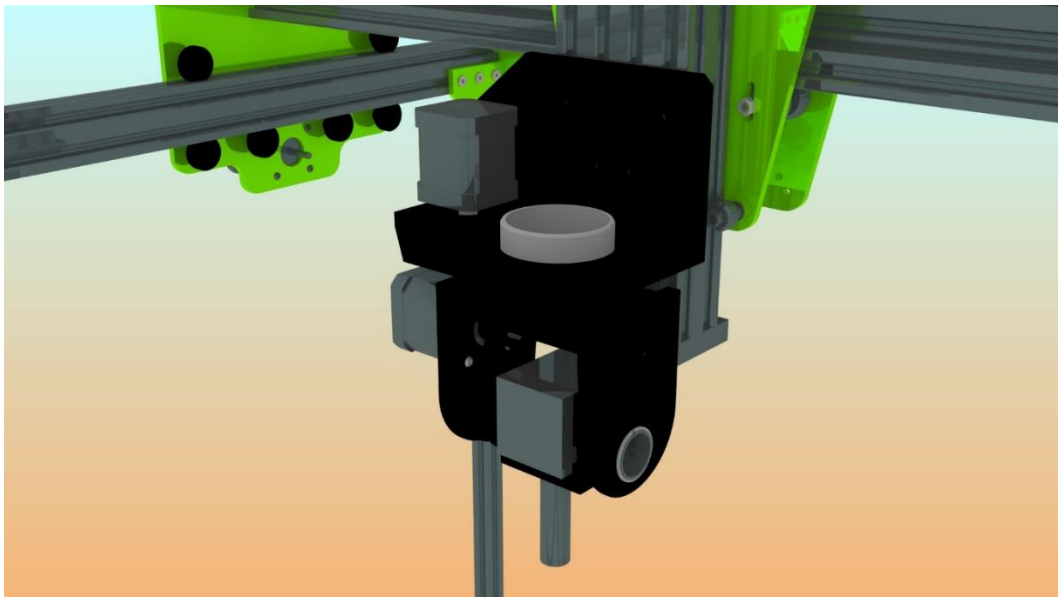

Figure 41 - Render of the extruder holder

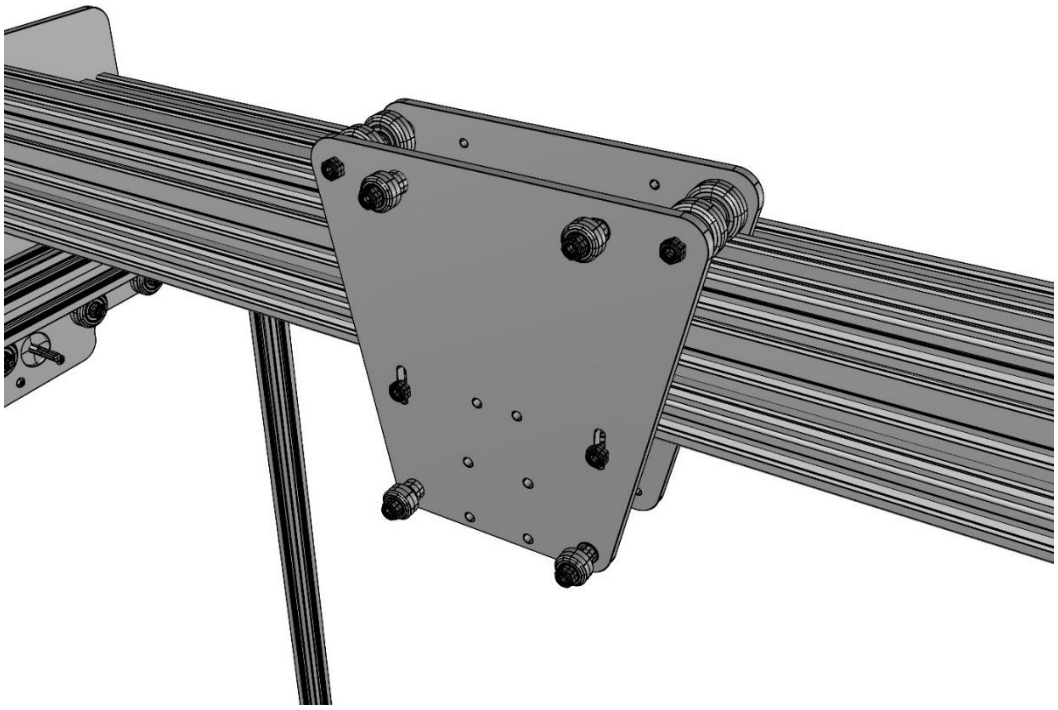

Figure 42 - Carriage for the Y-axis

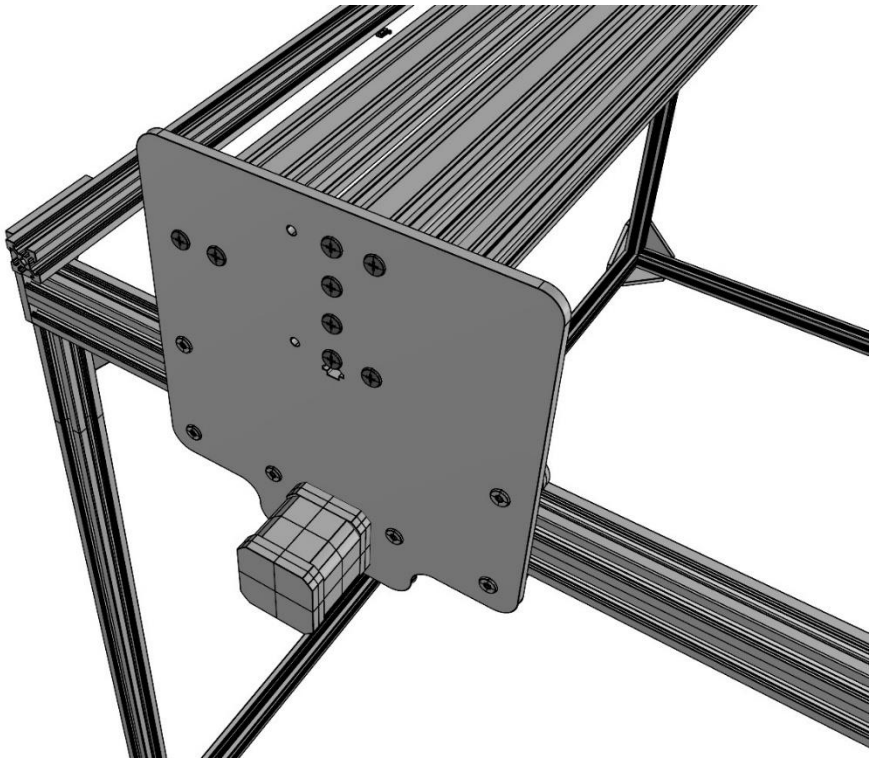

Figure 43 - Carriage for the X-axis

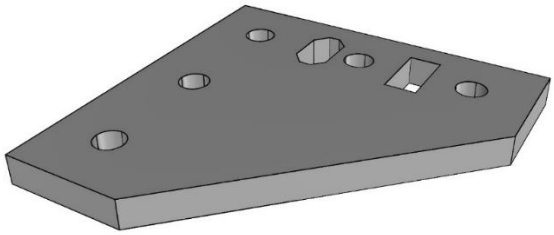

Figure 44 - Corner connection plate

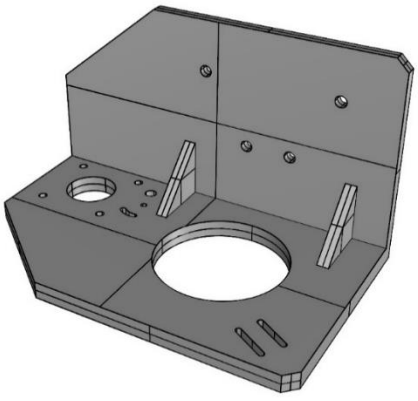

Figure 45 - Extruder holder part 1

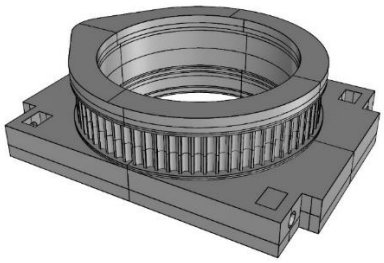

Figure 46 - Extruder holder part 2

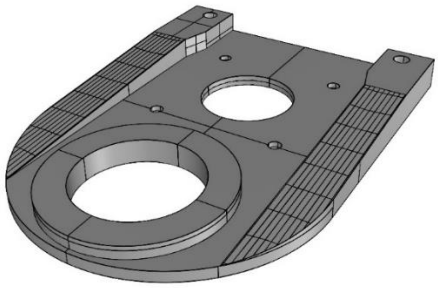

Figure 47 - Extruder holder part 3

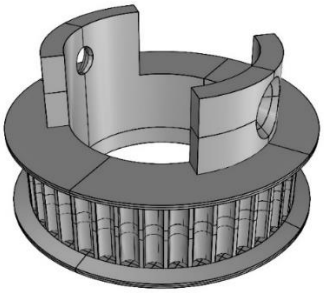

Figure 48 - Extruder holder part 4

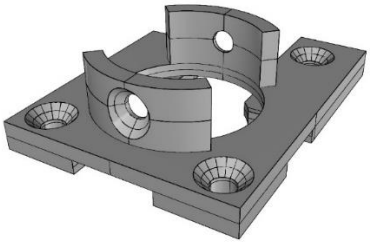

Figure 49 - Extruder holder part 5

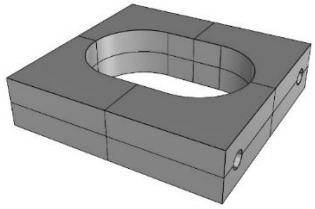

Figure 50 - Extruder holder part 6

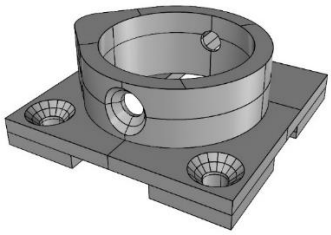

Figure 51 - Extruder holder part 7

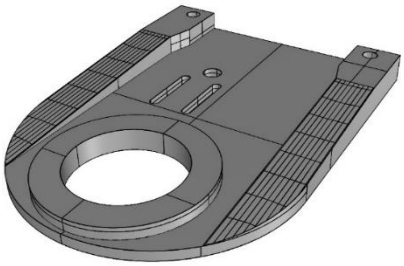

Figure 52 - Extruder holder part 8

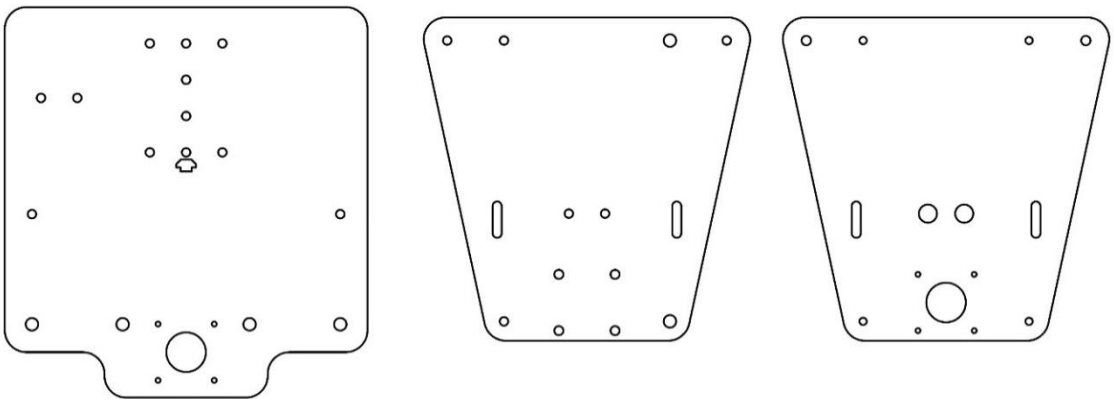

Figure 53 - Design of the mounting plates

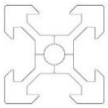

Figure 54 - 20 x 20 profile

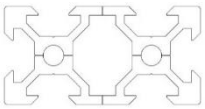

Figure 55 - 20 x 40 profile

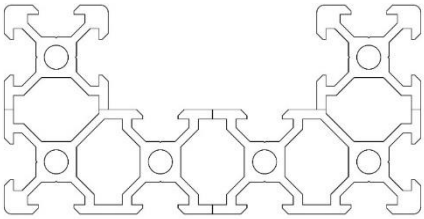

Figure 56 - C-beam profile

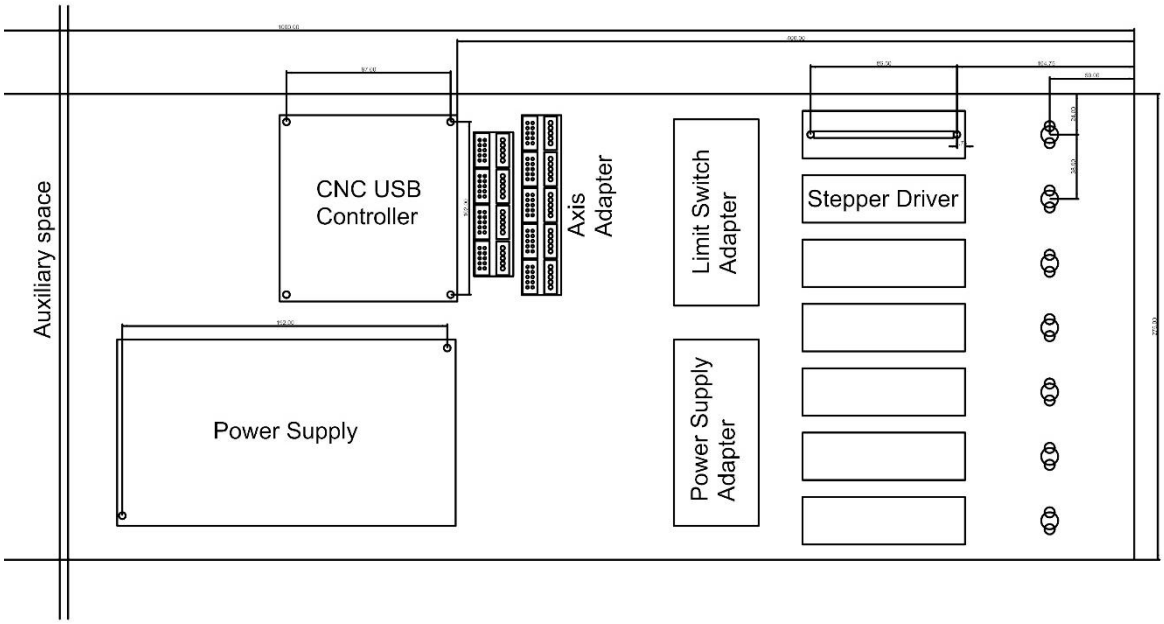

Figure 57 - Layout for the electronics

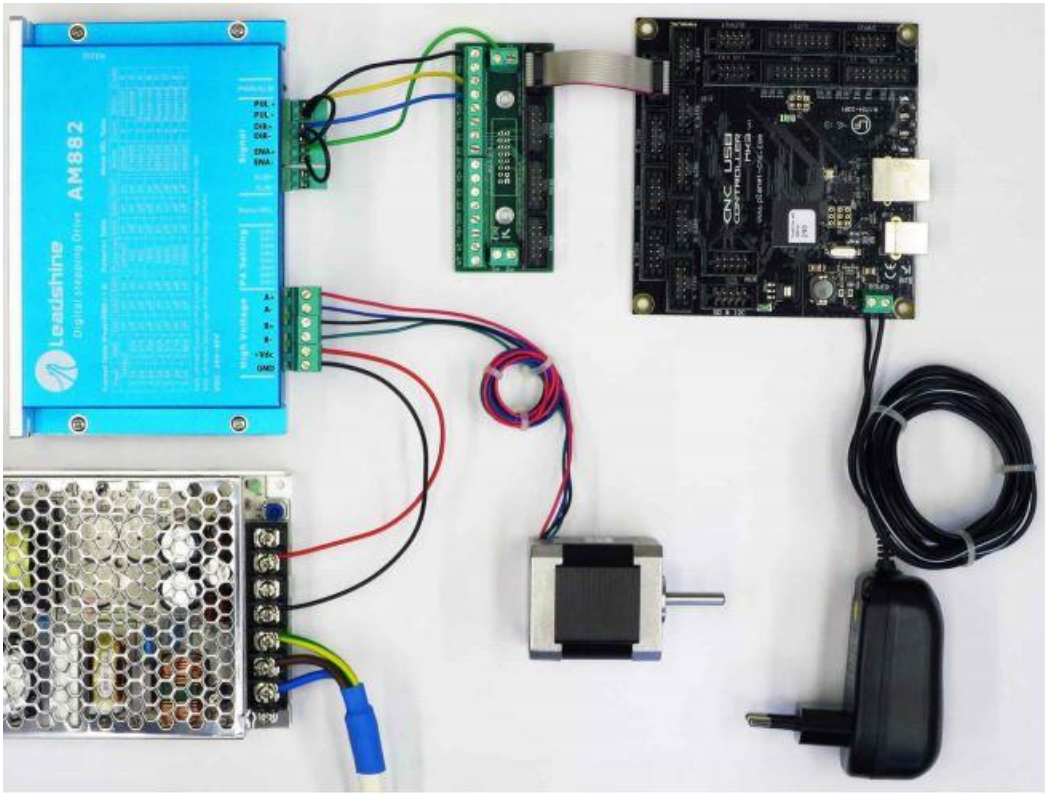

Figure 58 - Wiring example from PlanetCNC [7]

Figure 59 - 24V 12.5A switching power supply

Specifications:	
AC Input	110V-220V
Output / Current	DC 24V / 12.5A
Power	300W
Dimensions:	20 x 11 x 5 (cm)
Weight	800g

Table 3 - Specification table for the power supply

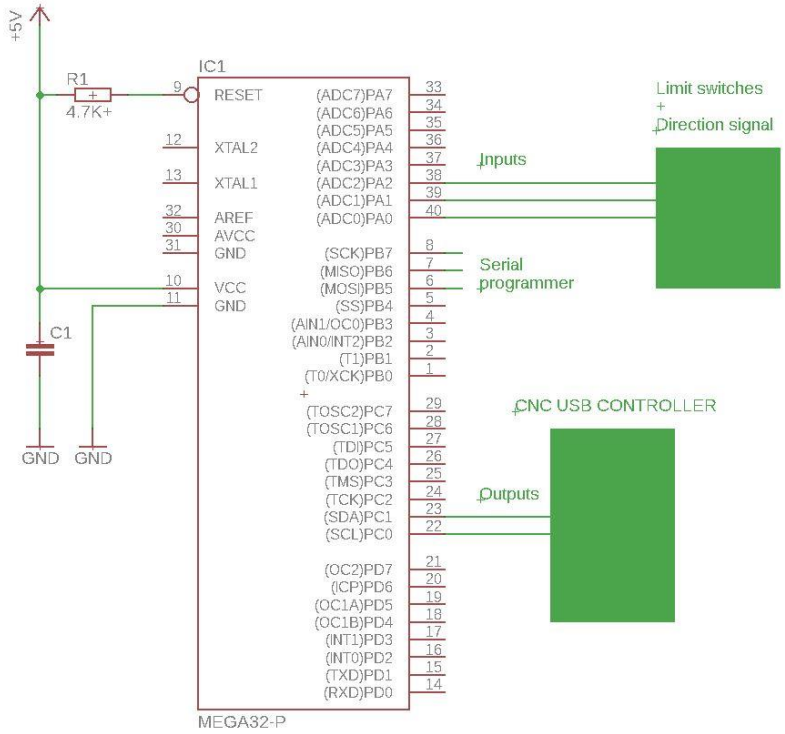

Figure 60 - Connection diagram for the µP

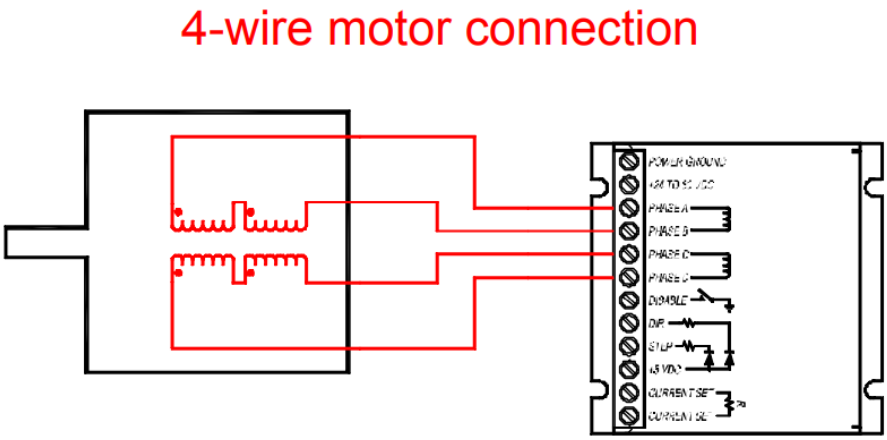

Figure 61 - Stepper motor coil configuration

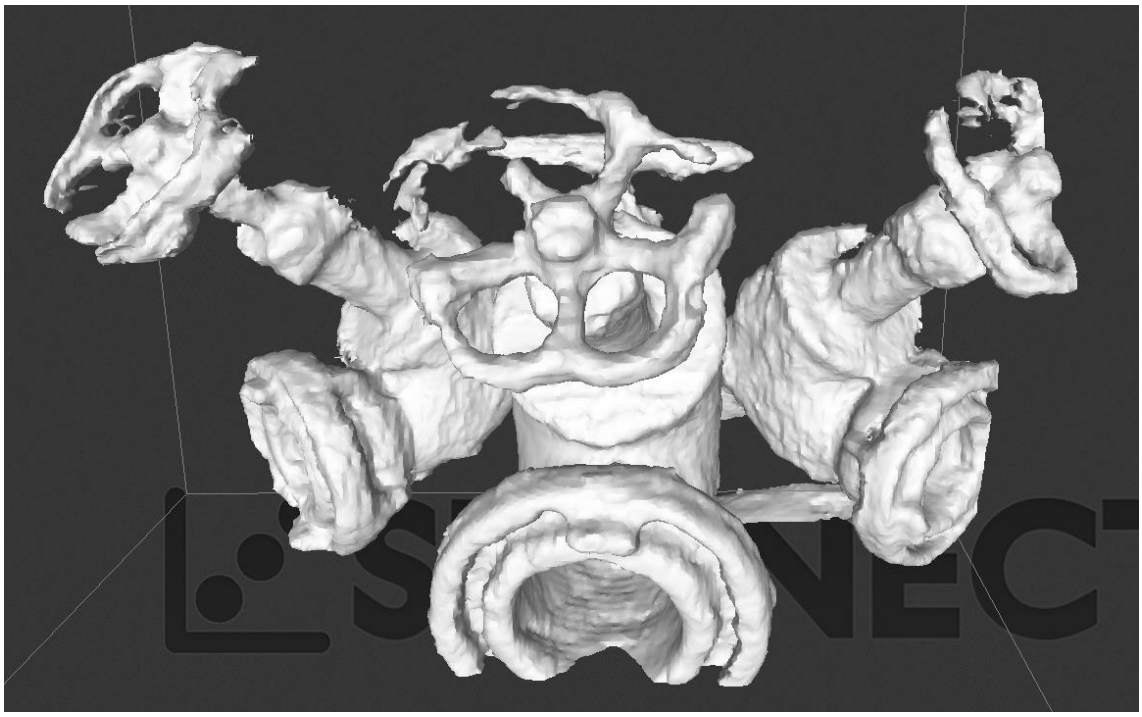

Figure 62 - Reconstruction of a 3D scan

```c
/*
* EndSwitches.c
*
* Created: 02/03/2018 12:56:57
* Author : WorkPC
*/
#define F_CPU 8000000UL

#include <avr/io.h>
#include <avr/interrupt.h>
#include "avr/delay.h"

#include "debounce.h"
#include "Timer0.h"

// global Variables form other files
extern volatile unsigned int Timer0Counter;

int i = 0;
int j = 0;
int DIR;

// Called at about 1000Hz   the ISR is executed every ms
ISR(TIMER0_COMP_vect)
{
    static unsigned int DebounceTimeMark;
    // Debounce buttons. debounce() is declared static inline
    // in debounce.h so we will not suffer from the added overhead
    // of a (external) function call
    Timer0Counter++;

    if (Timer0Counter - 10 > DebounceTimeMark)  //every 10 ms call debounce
    {
        DebounceTimeMark=Timer0Counter;
        debounce();
    }
}

int main(void)
{

    //local variables
    unsigned int StayAlive = 0;
    unsigned int StayAlivePriode = 200;

    // PORTC output
    DDRC = 0xFF;

    InitTimer0();
    // Enable global interrupts
    sei();

    debounce_init();

    while(1)
    {
```

```
        if (Timer0Counter - StayAlivePriode > StayAlive)    //every 200 ms toggle
          stay alive output for a LED indicator
        {
            StayAlive = Timer0Counter;
            PORTC ^= (1<<PC2);
        }

        if (button_down(BUTTON1_MASK))      //Read input 1 for button press
        {
            if (PINA & DIRECTIONPIN)        //Direction signal is +5V
            i--;
            if ((PINA & DIRECTIONPIN)==0)   //Direction signal is 0V
            i++;
        }

        if (button_down(BUTTON2_MASK))      //Read input 2 for button press
        {
            if (PINA & DIRECTIONPIN)        //Direction signal is +5V
            j--;
            if ((PINA & DIRECTIONPIN)==0)   //Direction signal is 0V
            j++;
        }


        if (i==-1 && j==0)
        {
            i = 0;
            PORTC |= (1<<PC0);      //ouput 1 for limit input 1 becomes high
            _delay_ms(500);
            PORTC &= ~(1<<PC0);     //ouput 1 for limit input 1 becomes low
        }


        if (i==1 && j==2)
        {
            j = 1;
            PORTC |= (1<<PC1);      //ouput 2 for limit input 2 becomes high
            _delay_ms(500);
            PORTC &= ~(1<<PC1);     //ouput 2 for limit input 2 becomes low
        }

    }
}
```

```c
/*
 * Timer0.c
 *
 * Created: 08/03/2018 09:25:49
 *  Author: WorkPC
 */

#include <avr/io.h>
#include "Timer0.h"

//global Variable
volatile unsigned int Timer0Counter = 0;

void InitTimer0()
{
    TCCR0 |= (1<<WGM01)|(0b100<<CS00);      // WGM01 is set to 1 so the timer is
      cleared on compare match mode
                                            // prescaler to 256 -> freq 31250
    OCR0=31;                                // freq 1000 -> tick 1 ms OCR0=31
    TIMSK = 1<<OCIE0;                       // Timer/Counter0 Output Compare Match
      Interrupt Enable
}
```

```
/*
 * Timer0.h
 *
 * Created: 08/03/2018 09:24:29
 *  Author: WorkPC
 */


#ifndef TIMER0_H_
#define TIMER0_H_

void InitTimer0(void);

#endif /* TIMER0_H_ */
```

```c
/*
* Debounce.c
*
* Created: 08/03/2018 11:05:15
*  Author: WorkPC
*/

#include "debounce.h"

// Bits is set to one if a debounced press is detected.
volatile uint8_t buttons_down;

// Return non-zero if a button matching mask is pressed.
uint8_t button_down(uint8_t button_mask)
{
    // ATOMIC_BLOCK is needed if debounce() is called from within an ISR
    ATOMIC_BLOCK(ATOMIC_RESTORESTATE){
        // And with debounced state for a one if they match
        button_mask &= buttons_down;
        // Clear if there was a match
        buttons_down ^= button_mask;
    }
    // Return non-zero if there was a match
    return button_mask;
}
void debounce_init(void)
{
    // Button pins as input
    BUTTON_DDR &= ~(BUTTON_MASK);
    // Enable pullup on buttons
    BUTTON_PORT |= BUTTON_MASK;
}
```

```
/*
* Debounce.h
*
* Created: 08/03/2018 11:02:09
*  Author: WorkPC
*/


#ifndef DEBOUNCE_H_
#define DEBOUNCE_H_

#include <avr/io.h>
#include <avr/interrupt.h>
#include <util/atomic.h>

// Buttons connected to PA0 and PA1
#define BUTTON_PORT  PORTA
#define BUTTON_PIN   (PINA0|PINA1)//PINA
#define BUTTON_DDR   DDRA
#define BUTTON1_MASK (1<<PA0)
#define BUTTON2_MASK (1<<PA1)
#define DIRECTIONPIN (1<<PA2)
#define BUTTON_MASK  (BUTTON1_MASK | BUTTON2_MASK)

// Variable to tell that the button is pressed (and debounced).
// Can be read with button_down() which will clear it.
extern volatile uint8_t buttons_down;

// Return non-zero if a button matching mask is pressed.
uint8_t button_down(uint8_t button_mask);

// Make button pins inputs and activate internal pullups.
void debounce_init(void);

// Decrease 2 bit vertical counter where mask = 1.
// Set counters to binary 11 where mask = 0.
#define VC_DEC_OR_SET(high, low, mask)          \
low = ~(low & mask);                            \
high = low ^ (high & mask)

// Check button state and set bits in the button_down variable if a
// debounced button down press is detected.
// Call this function every 10 ms or so.
static inline void debounce(void)
{
    // Eight vertical two bit counters for number of equal states
    static uint8_t vcount_low  = 0xFF, vcount_high = 0xFF;

    // Keeps track of current (debounced) state
    static uint8_t button_state = 0;

    // Read buttons (active low so invert with ~). Xor with
    // button_state to see which ones are about to change state
    uint8_t state_changed = BUTTON_PIN ^ button_state;              // orginally   ↵
      inverted for positive logic: uint8_t state_changed = ~BUTTON_PIN ^           ↵
      button_state;
```

```
    // Decrease counters where state_changed = 1, set the others to 0b11.
    VC_DEC_OR_SET(vcount_high, vcount_low, state_changed);

    // Update state_changed to have a 1 only if the counter overflowed
    state_changed &= vcount_low & vcount_high;

    // Change button_state for the buttons who's counters rolled over
    button_state ^= state_changed;

    // Update button_down with buttons who's counters rolled over
    // and who's state is 1 (pressed)
    buttons_down |= button_state & state_changed;
}
#endif /* DEBOUNCE_H_ */
```