Design of a controller for an adjustable frictionless resistance fitness device, with wireless user feedback

June 2015

Tong Dong Qiu	4226690
Niels van der Veen	4220560

Supervisors:

Henk Polinder
Dong Liu

Delft University Of Technology
Faculty of EEMCS

This report is written as a thesis for the bachelor graduation project of Electrical Engineering of the faculty Electrical Engineering, Mathematics and Computer Science of Delft University of Technology. It proposes a design for a control system for a variable frictionless resistance training device, that is able to be set and give feedback to the user via a wireless connection. Within a period of nine weeks, the design was made, built and tested and is discussed in this thesis.

Contents

Chapter 1

Introduction

1.1 Problem definition

The problem to be solved is to design an adjustable frictionless resistance alternative to a friction based resistance of a portable fitness device called the DISQ. The DISQ is a fitness device that involves pulling a cord. The cord is attached to the hip, and a pulley is attached to the ankle of the user. There are two cords and two pulleys, one on each side. An impression of the DISQ is shown in figure 1.1. The resistance that is experienced when pulling the cord is variable, and established by using a braking disk. This method relies on friction and is therefore subject to wear. The objective is to design a frictionless resistance to eliminate that wear.

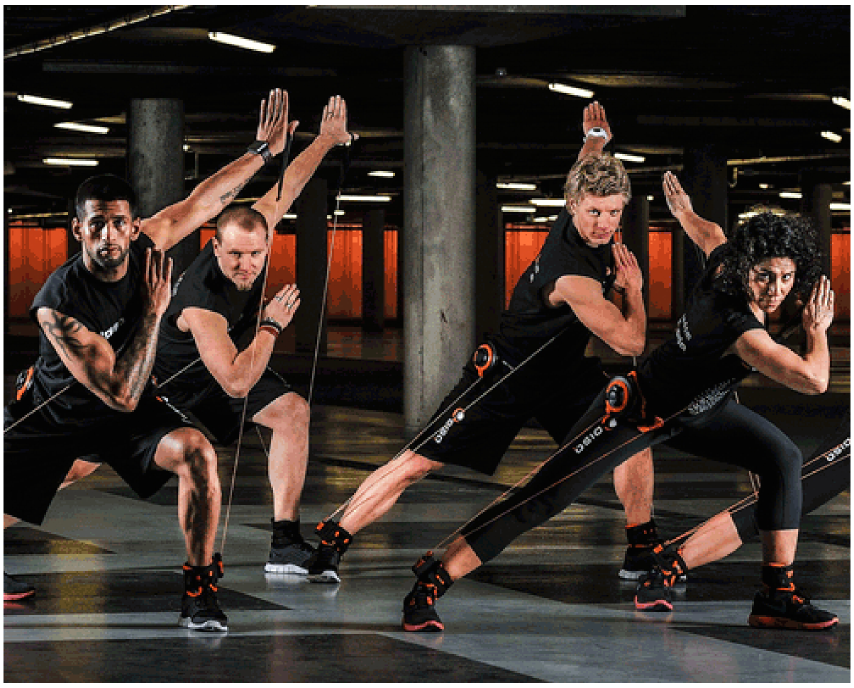


Figure 1.1: The DISQ in use

1.2 Requirements

In this section the requirements for the product will be set. The product is the improved version of the portable fitness device "The DISQ".

1.2.1 Requirements for the intended use

- A physical resistance has to be applied when the cord is extended

- The physical resistance must be adjustable by the user
- The physical resistance should preferably be independent of the speed of the extension
- The system should function without a connection to an external power supply during operation
- The system should preferably never require connection to an external power supply
- The system should preferably use the energy the user generates
- The system should provide the user with feedback on his performance
- The system should preferably provide its feedback to the user via a wireless connection to a smartphone
- The system should be able to start providing feedback to the user within the first 10 seconds of an exercise
- The system should be able to maintain the Bluetooth connection to the smartphone for at least 5 minutes after the end of an exercise due to resting periods between exercises
- The system should not reach harmful temperatures
- The system should not weigh more than 1 kg
- The system should not emit any noise which a user would find annoying
- The system should not expose the user to sensible voltage levels

1.2.2 Requirements for the designed system

1.2.2.1 Usage characteristics

- The system has to be at most twice as large as the current DISQ
- The system should function at cord extension speeds up to 2.5 m/s
- The system should be able to provide a resistance of up to 100 N at a speed of 1.5 m/s
- The system should remain functioning (pass all the requirements set in this programme of requirements) up to at least two years after purchase.
- The time it takes to start a workout should not be increased by more than 10 seconds compared to that needed for the current versions of the DISQ
- The system should remain functioning (pass all the requirements set in this programme of requirements) for the following usage scenarios:
  - A one hour exercise at 45 W power input to the system
  - A ten minute work out at 90 W power input to the system

1.2.2.2 Ecological requirements

- Any components containing dangerous substances should be easily separable from the rest of the device
- Any components containing dangerous substances should not be easily accessible for users

1.2.3 Production requirements

- The manufacturing costs of the device should not be higher than €250 when producing more than 1000

1.3 Thesis setup

In this thesis a hierarchical structure is maintained via chapters, sections, subsections, subsubsections and paragraphs. The order in which different elements will be discussed, is as follows. First, the design of the system will be discussed, beginning with a brief introduction of the system overviews. Then, the different components of the developed hardware are discussed starting with the schematics and ending with the final PCB-designs. After the hardware is discussed, the software is covered starting with MCU firmware, the GUI and ending with the wireless communication protocol. After the system design, the system testing is discussed, concerning test setups, test results and test discussions of various parts of the system. The thesis ends with a note on future improvements, a conclusion and finally ethical considerations regarding the final product.

Chapter 2

System design

2.1 Introduction

2.1.1 Global system overview

As written earlier, the problem to be solved is to design a variable frictionless resistance braking system for an existing fitness device. Different solutions to this problem were to be found. Reaching from eddy current braking to the use of electrorheological(ER) fluids. Eddy current braking was soon rejected due to the fact that the resistive torque it provides, strongly depends on the (angular) speed of a to be braked object and is negligible at low speeds. Also, a mechanical instead of an electrical controlling system is needed or the use of very large electromagnets with driving currents of approximately 10 amperes is inevitable [1]. This high current would need to come from an external power supply, this is not a practical choice for a portable device. Moreover, the use of high currents result in heat generation and a lot of losses. Very large electromagnets are not a realistic solution for a portable fitness device and the design of a mechanical control system was found to be beyond the scope of the Electrical Engineering programme. ER fluids was a very interesting solution which is based on one of the properties of ER fluids, namely the ability to change the viscosity of the fluid when a (variable) electric field is applied. However, the implementation of this solution was found to be unfeasable in the available time slot. This was mainly due to the need of designing a, as thin as possible, water-tight enclosure on which an electric field of several kilovolts has to be applied. On top of that, rotating parts had to be positioned inside this enclosure. The design of such a system is also beyond the scope of the bachelor Electrical Engineering programme.

The proposed solution for this problem is based on the conversion of energy from the mechanical to the electrical domain and the dissipation of this energy in a variable resistive load. The user of the fitness device can be given feedback in the form of information about his physical performance and the device itself operates without the need of an external power source. The system can be divided into three parts and is shown in figure 2.1:

- The conversion from the mechanical to the electrical domain
- A subsystem for storing and dissipation of energy or Energy Management System (EMS)
- A control system for controlling various signals as well as to provide an interface to the user and provide information about the generated and dissipated energy

The conversion subsystem contains a three-phase motor, that will act as a generator. The three phases are rectified and their energy is stored in a capacitor. This subsystem also contains a current sensor, that will be read and interpreted by the control system.

The Energy Management System (EMS) will store generated electrical energy in a lithium-ion battery, and dissipate the rest into resistors. When the battery is charged, the EMS uses a DC-DC converter to provide the control system with a suitable voltage.

The latter subsystem is the system which is discussed in this thesis.

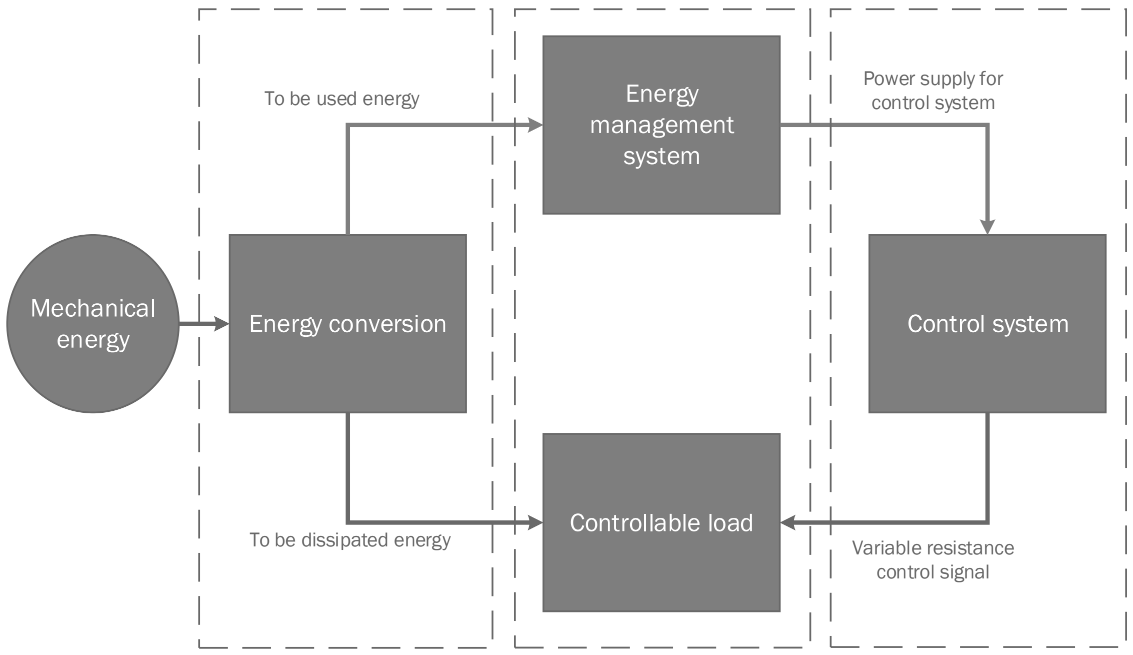


Figure 2.1: Global system block diagram

2.1.1.1 Control system problem definition

As stated before, a control system has to be designed to solve the problem discussed earlier in this thesis. The problem definition for this subsystem can be formulated as follows:
How to design a control system that is able to control the resistance a user experiences when exercising with a portable fitness device and that is able give feedback to that user, efficiently and user friendly.

2.1.2 Control system overview

The control system controls the resistance the user experiences. The user should be able to choose a certain resistance setting, so the control system must be able to vary the resistance. Controlling the resistance is done by outputting a Pulse Width Modulation (PWM) signal, varying its duty cycle allows a variable resistance.

The control system also measures the electrical power generated, and sends this information via a wireless connection to the user. The user is provided with an app on his or her smartphone and receives the information there. To prevent the generator from overheating, the temperature is monitored. When the temperature gets too high, the controller must be able to decrease the power generated by limiting the current or totally break the current flown into the load. This decreases the torque felt by the user and limits the power. The measured temperature can also be sent to the app.

The wireless connection can be used to give feedback, such as power or temperature measurements. But it can also be used to control the DISQ. The user should be able to control the resistance via the app, and view the resistance setting when the resistance is set on the device itself. Also, the user should be able to create its own training scheme on the app. An overview of the control system is shown in figure 2.2.

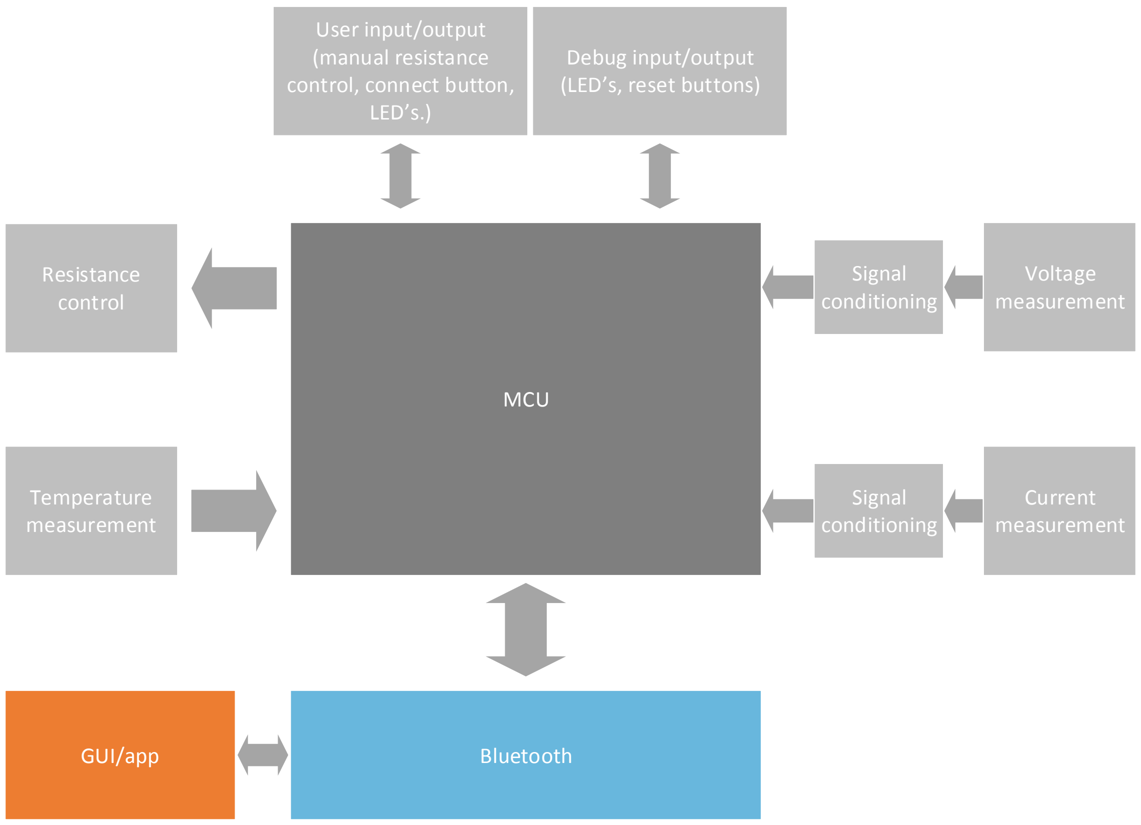


Figure 2.2: Block diagram of the controller

2.1.3 Requirements

As can be read from the previous sections, various requirements are applicable to the control system:

- **Main function:** The user should be able to control the resistance he or she experiences both via a wireless connection with the help of an app and with buttons on the device itself.
- **Human interface:** The user should be provided with feedback about his performance in the form of information about the generated electric power.
- **Protection:** The control system monitors the temperature of the generator and is able to respond to this temperature to prevent the generator from overheating.
- **Power saving:** The control system should be able to operate from a battery power source for a reasonable time.

2.2 Hardware

2.2.1 Introduction

In this section, the hardware of the designed control system is discussed. The hardware is decomposed into smaller components and the design choices for each component are explained.

The hardware will be placed on a printed circuit board (PCB), because

- a PCB prototype will generally be smaller than a breadboard. The advantage of this is that the prototype will be more portable and can be efficiently taken to other places when needed.

- a PCB with soldered components and embedded "wires"(traces) is a lot more reliable than a breadboard with "snapped in place" components and physical wires put in connector holes.
- a PCB can be designed with the help of computer aided tools (CAT). This helps the designer get an overview of the, otherwise, rather complicated system. When breadboard based design is used, the clutter, among other things, makes everything very confusing.

2.2.2 Schematic

The controller is designed and the schematic diagram is shown in figure 2.3.

The letters in the schematic correspond with the following parts of the control system, which will be made clear in the following of this thesis:

- A: Power supply
- B: Temperature measurement
- C: Power measurement
- D: Programming (ICSP) + reset
- E: Status leds
- F: Microcontroller
- G: External IO's
- H: User buttons
- I: Bluetooth module
- J: Bluetooth reset buttons

2.2.3 Microcontroller

The heart of the control system is a microcontroller or MCU. This programmable part is responsible for all actions of the control system such as processing user input, processing information about the generated power, sending feedback to the user and provide the required control signal for the adjustable resistance. In this subsection, the MCU of the control system is discussed.

2.2.3.1 MCU requirements

To do measurements, the controller must have an analog-to-digital converter (ADC). An external ADC could be used, but this would take up space. The MCU should have one internally. The resolution the power can be measured in depends on the resolution of the ADC. When a supply voltage VDD of 3.3 V is used, the total input range for the ADC is 3.3 V. When errors of 0.1% due to decimation of the input signal are tolerated (that is $3.3\text{ V} \cdot 0.001 = 0.0033\text{ V}$), the ADC needs to be able to distinguish $\frac{100\%}{0.1\%} = 1000$ steps. A 10-bit ADC (1024 levels) is then sufficient. A decimation error of 0.1% is found to be acceptable since modern measurement equipment show similar inaccuracies [2]. To control the resistance, a PWM signal is needed. This can also be implemented in the software. A PWM module is an asset, but not required. To communicate wirelessly with the user, a transceiver is needed. What type of transceiver depends on the form of wireless communication that is going to be used. Several options are: Zigbee, Bluetooth, WiFi, ultrasonic and infrared (section 2.2.8). An MCU with a built-in transceiver is probably easier to implement, since there is no extra communication between the MCU and transceiver needed. The DISQ will feature buttons with which the resistance can be set. The MCU will need to have IO pins to accommodate that. Most MCU's have general purpose input/output (GPIO) pins.

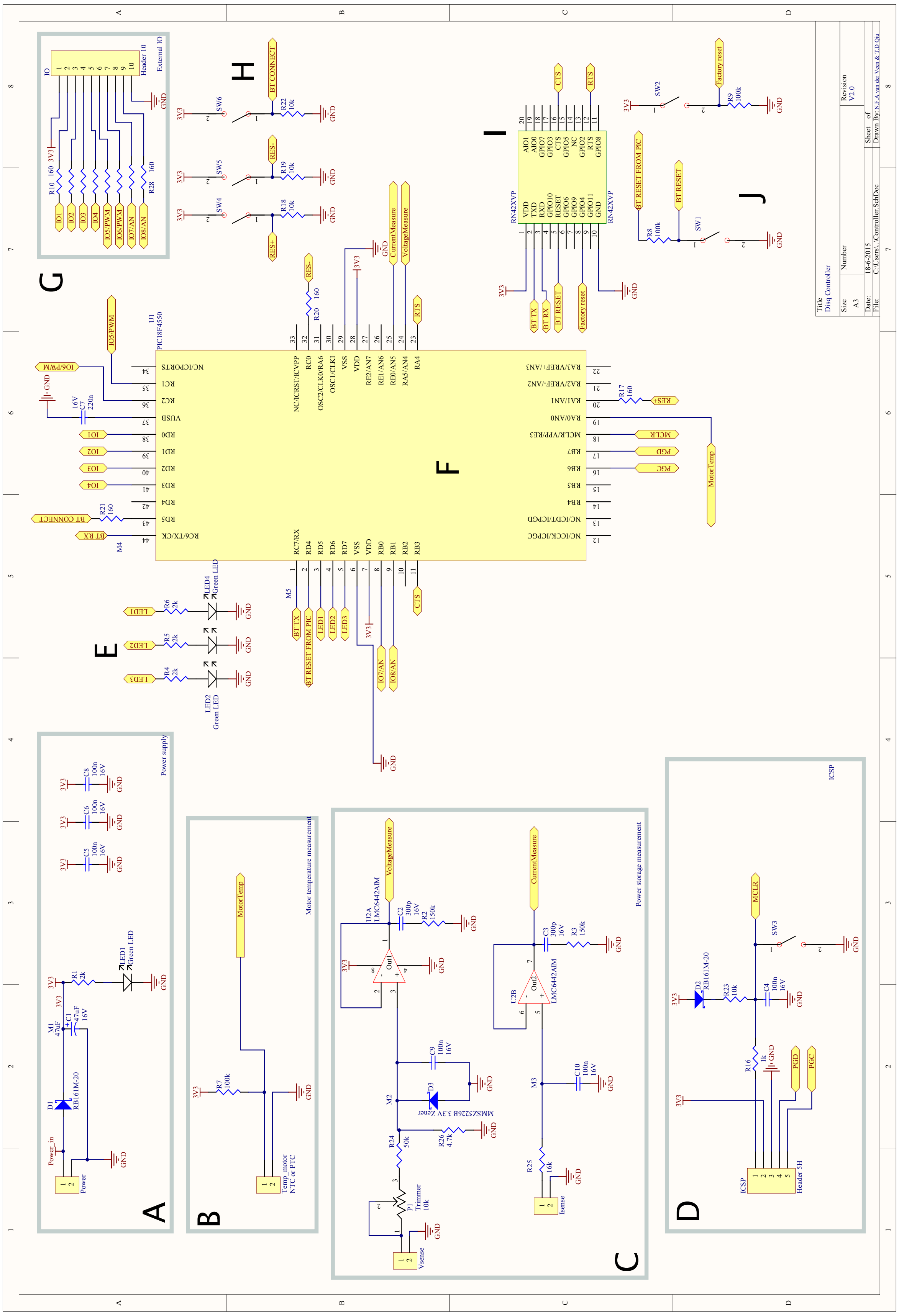

A
B
C
D
E
F
G
H
I
J
Power supply
Motor temperature measurement
Power storage measurement
ICSP
External IO
U1
PIC18F4550
RN42XVP
LMC6442AIM
Title Disq Controller
Size A3
Revision V2.0
Date: 18-6-2015
Drawn By: N.F.A van der Veen & T.D.Qiu
File: C:\Users\...\Controller.SchDoc

Figure 2.3: Schematic diagram of the controller
C7 is a design error and can be removed
Pins 42 to 44 are called RD4, RD5, RD6, but should be named RC4, RC5, RC6

2.2.3.2 Arduino vs custom

An MCU can be of different forms. It is possible to use a off the shelf "computing platform" like the well known Arduino, or to design a custom control system in which every component can be designed to comply with the requirements. The use of an off the shelf platform leads to some limitations and reduces the design freedom of the control system. As written in section 2.2.3.1, the control system needs an ADC, PWM output capabilities, communication options and buttons. When using an Arduino for example, all external circuitry for measurement and communication has to be designed and implemented seperately from the Arduino. Apart from decreasing the flexibility of such a system this way, it also is very likely that the final prototype (and product) will physically be much larger compared to a custom designed control system. Mainly for these reasons the choice was made for a custom designed control system.

2.2.3.3 Choice of MCU

Now that the design choice of using a custom control system is justified, the choice of the used MCU has to be explained. Since there are countless different MCU's on the market today for almost every application possible, a comparison between different MCU's would be of no use. The MCU used in the control system is a PIC18F4550 from Microchip^{TM}. The main reason for using this MCU is the fact that previous experience was present with this device and that a programmer for this device was available at hand. Also, this type of MCU complies to the requirements set in 2.2.3.1. In table 2.1, a summary of some of the PIC18F4550 specifications is shown[3].

Number of GPIO pins	35
Clock source	Internal or external
Supply voltage (V)	2.0 to 5.5
Available memory (bytes)	Flash: 32K, SRAM: 2048, EEPROM: 256
Communication protocols	USB, UART[1], SPI[2], I2C[3]
Available internal modules	10-bit ADC, CCP, PWM, Timers
Package	44-pin TQFP[4]
Price (€)[5]	1+ 5.89, 100+ 3.95

Table 2.1: Specification summary of PIC18F4550. [1]Universal asynchronous receiver/transmitter, [2]Serial Peripheral Interface, [3]Inter-IC, [4]Thin Quad Flat Package, [5]As found on the 7th of June 2015 on http://nl.farnell.com for 1+ and 100+ device count.

2.2.3.4 Connections

The used PIC18F4550 MCU comes in a 44-pins TQFP package as shown in figure 2.4. These pins have to be connected to several subparts of the control system in order for the MCU to communicate with these subparts. In table 2.2, these connections are tabulated.

2.2.4 Power supply

The Energy Management System(EMS) is responsible for supplying a steady 3.6 V to our PCB. The typical supply voltage is 3.3 V, but a Schottky diode (type RB161M-20) is used to prevent possible damage to the PCB when the polarity of the supply voltage is accidentally reversed. The use of a Schottky diode here is beneficial, because Schottky diodes generally have a lower forward voltage than "regular" semiconductor diodes. The RB161M-20 has a forward voltage of 240 mV when the current drawn is 200 mA, which is the absolute maximum for the PCB. The RB161M-20 is chosen for its small forward voltage and low costs compared to other possible diodes, as can be seen in table 2.3. The forward voltage drops to 85 mV when no current is drawn[4]. The absolute maximum supply voltage for the Bluetooth module of 3.6 V is never reached, ensuring safe operation. The op-amps and MCU have higher maximum supply voltages.

Pin number	Short description	Function	Input/Output	Analog channel (if applicable)
1	RC7	BT TX	I	
2	RD4	BT reset from PIC	O	
3	RD5	LED1	O	
4	RD6	LED2	O	
5	RD7	LED3	O	
6	VSS	Ground		
7	VDD	+3.3 V supply		
8	RB0	IO7/AN	I/O	
9	RB1	IO8/AN	I/O	
10	NC[1]			
11	RB3	CTS	O	
12	NC			
13	NC			
14	NC			
15	NC			
16	RB6	PGC	I	
17	RB7	PGD	I	
18	MCLR	MCLR	I	
19	RA0	Motor temperature	I	0
20	RA1	RES+	I	
21	NC			
22	NC			
23	RA4	RTS	I	
24	RA5	V_{sense}	I	4
25	RE0	I_{sense}	I	5
26	NC			
27	NC			
28	VDD	+3.3 V supply		
29	VSS	Ground		
30	NC			
31	NC			
32	RC0	RES-	I	
33	NC			
34	NC			
35	RC1	IO5/PWM	I/O	
36	RC2	IO6/PWM	I/O	
37	NC			
38	RD0	IO1	I/O	
39	RD1	IO2	I/O	
40	RD2	IO3	I/O	
41	RD3	IO4	I/O	
42	NC			
43	RC5	BT connect	I	
44	RC6	BT RX	O	

Table 2.2: Pin connections of the PIC18F4550 in designed control system

[1] Not Connected

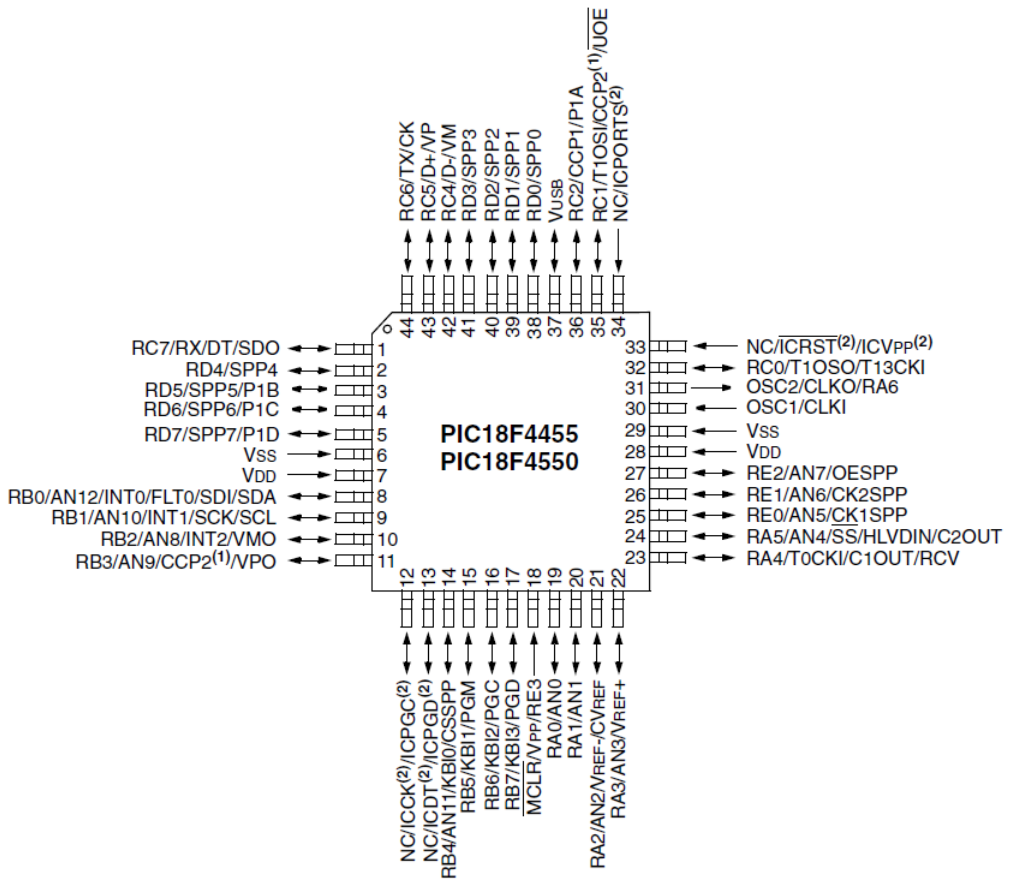


Figure 2.4: Pinmap of the PIC18F4550

The power supply block is responsible for keeping a reliable 3.3 V supply voltage for all components on the PCB. The capacitor C1 keeps the voltage constant by filtering possible unwanted ripple from the power supply. Also, capacitive decoupling (C5,C6,C8) is present near integrated circuits to decrease high frequency noise and improve performance[5]. The green LED indicates that the PCB is powered. A 2 kΩ resistance in series with the LED provides a current of $\frac{V_{supply}-V_f}{R_{series}} = \frac{3.3-2.1}{2k} = 0.6$ mA. In which V_f is the voltage across the LED. This current results in a luminous intensity of 3% relative to the intensity for a current of 20 mA [6]. The circuit is shown in figure 2.5.

	BAT42W	BAT54T1G	RB161M-20
Max current (I_f)(mA)	200	200	1000
Forward voltage (V_f)(mV)	650	350	220
Costs (€)	0.05	0.55	0.105

Table 2.3: Different possible polarityprotection diodes [7][8][4]
These prices were found on May 1st 2015 on www.farnell.com for 1000 or more units.

2.2.5 Temperature measurement

To measure the temperature of the generator, a thermistor is attached to it. The thermistor is placed in a voltage divider as shown in figure 2.6, and the resulting voltage is read by the ADC module of the MCU.

Placing the thermistor in the upper or lower part of the divider makes no difference in sensitivity, it is placed in the lower leg.

Since the 10 bit ADC module is fed by 3.3 V. It can detect voltage differences of $\frac{3.3}{2^{10}} = 3.22$ mV. Since the power consumption needs to be low, 100 kΩ is chosen as a base for both the resistor

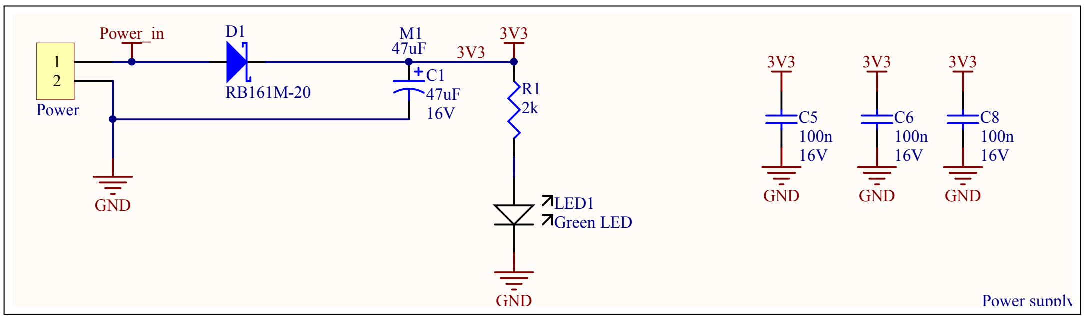


Figure 2.5: The power block

and thermistor. This gives a leakage current of 16.5 μA at room temperature ($T_0 = 25°C = 298.15$ K). Since there are no suitable 100 kΩ Positive Temperature Coefficient (PTC) thermistors for temperature measuring (most PTCs have a temperature curve which is not very useful in temperature measurements [9]), an Negative Temperature Coefficient (NTC) type is chosen. The next step is calculating how big the change in thermistor value must be to be detectable. The voltage at T_0 is $\frac{3.3}{2} = 1.65$ V. The next detectable voltage is $1.65 + 0.00322 = 1.65322$. The necessary change in thermistor value (δ) is calculated as follows:

$$\frac{100000 + \delta}{200000 + \delta} = \frac{1.65322}{3.3} \tag{2.1}$$

$$\delta = 391.1\ \Omega$$

The Steinhart-Hart equation is an empirically derived relation between the resistance and temperature of an NTC. Using this equation should give an accuracy of ±0.001°C over 100°C[10]. However, the temperature range of this application is about 20-60°C, in which case the simpler Beta model can be used, since that is ±0.5°C accurate over a 50°C span [10]. This is enough for this application.

The equation for the Beta model looks like this:

$$\frac{1}{T} = \frac{1}{T_0} + \frac{1}{\beta}\ln\left(\frac{R}{R_0}\right) \tag{2.2}$$

Where R_0 is 100 kΩ and the temperatures are in K. Since changes in R larger than 391 Ω can be detected. R must be 99609 Ω at a temperature $T > T_0$. By choosing the value of T that must be measured, a value for β can be calculated.

$$\beta = \frac{-0.00391766}{\left(\frac{1}{T} - \frac{1}{T_0}\right)} \tag{2.3}$$

T(°C)	T(K)	ΔT	β
25.1	298.15	0.1	3484
25.5	298.65	0.5	698
26	299.15	1	349
26.5	299.65	1.5	233
27	300.15	2	175
27.5	300.65	2.5	140
30	303.15	5	71

Table 2.4: Resolution vs β value

Table 2.4 shows that to measure temperature with a resolution of 0.1 K, a thermistor with a β of at least 3484 is needed. We chose a suitable thermistor that has a β value of 4190 [11].

The maximum temperature to be measured is 60°C, the resistance that our thermistor will have at that temperature is 22846 Ω, which corresponds to a voltage of 0.6138 V and an ADC level of 190. At this point, the MCU should disconnect the resistances for safety reasons.

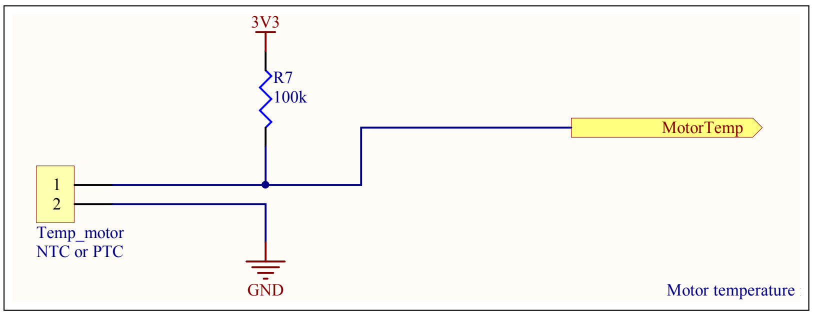


Figure 2.6: The temperature sensing circuit

2.2.6 Power measurement

The power measurement block measures the voltage and current at the output of the generator. The measured voltage and current are both fed as voltages into the MCU. The MCU converts them to digital signals. To prevent loading the generator and influencing its resistance, two buffers are used. These buffers consist of an op-amp in voltage follower configuration, ensuring a high input resistance. This is shown in figures 2.7 and 2.10. The voltage and current to be measured should not exceed 40 V and 10 A respectively.

Voltage measurement Because the generator can generate voltages up to 40 V, and the input of the op-amp may not exceed 3.6 V(V_{DD}+0.3) [12], a signal conditioning circuit is used. The input voltage is first divided, and if it is still higher than 3.3 V, the Zener diode will breakdown and limit the input of the op-amp. For this Zener diode, a MMSZ5226B from Fairchild^{TM} is used for its wide availability at the time of use. The division ratio of the voltage divider lies between $\frac{4.7}{54.7} \approx 0.0859$ and $\frac{4.7}{64.7} \approx 0.0726$ depending on the setting of the trimmer. This tunes the maximum unclipped generator voltage from 38.4 V to 45.4 V.

The values of the resistors are chosen to get a high enough input impedance, such that the generator is not affected too much. The 100 nF capacitance together with R24 and P1 (figure 2.7) acts as a low pass filter, of which the -3dB frequency (f_{-3dB}) lies between 192 Hz and 252 Hz, depending on the value of the trimmer (from high to low respectively). This f_{-3dB} is chosen low enough to filter out noise (e.g. from the Zener diode), and high enough to be able to measure the user's movement frequencies.

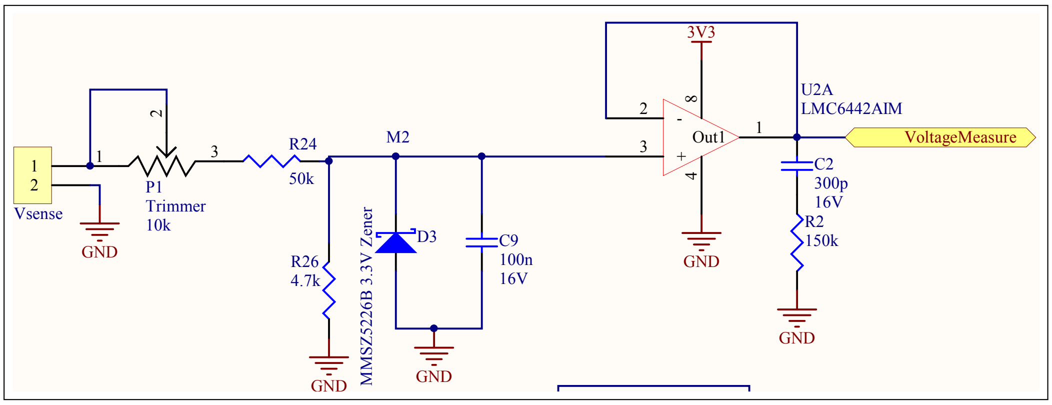


Figure 2.7: Voltage measurement circuit

The circuit in figure 2.7 is simulated and the simulation results are shown in figure 2.8.

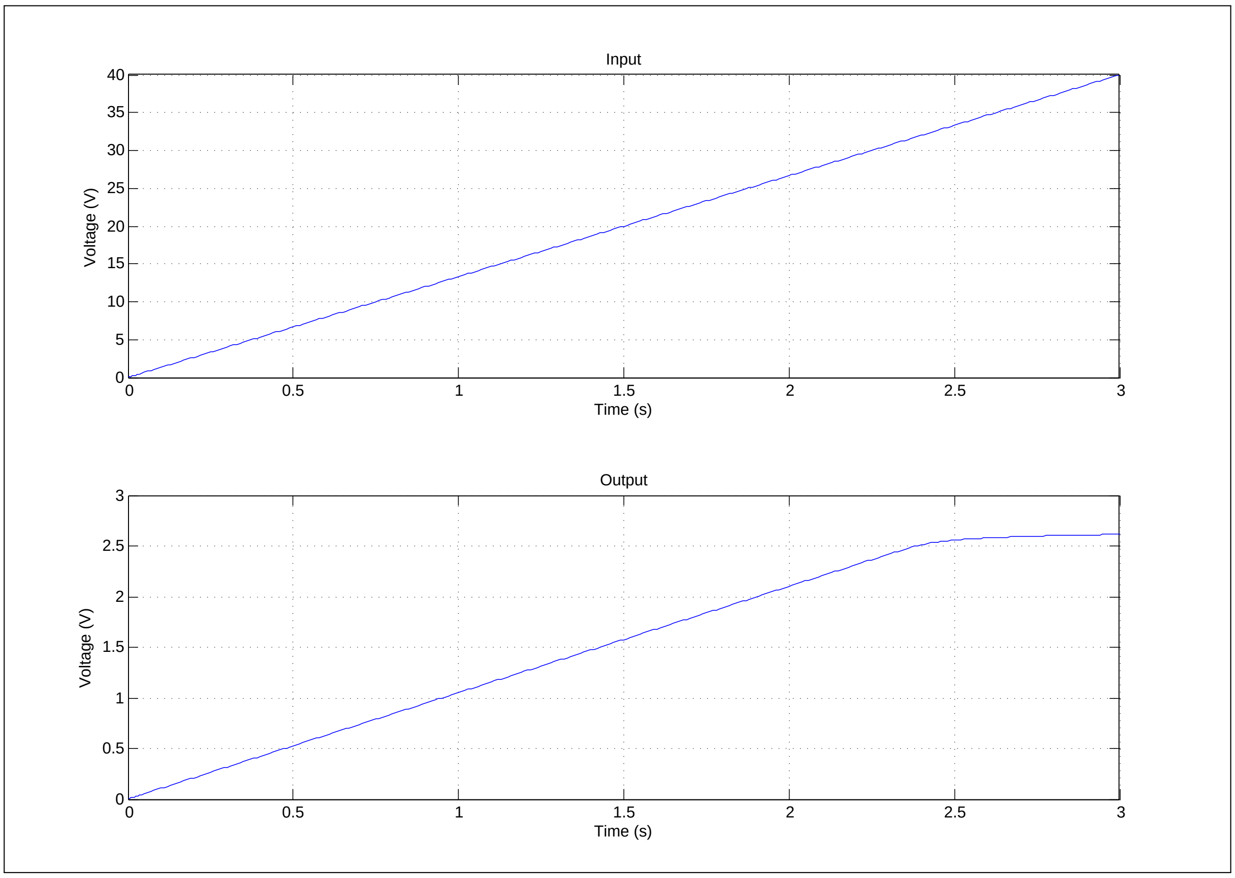


Figure 2.8: The simulated input and output of the voltage measurement circuit. The input voltage is the voltage at pin 1 at trimmer P1 in figure 2.7, the output voltage is the voltage at the output of the op-amp. The trimmer is set to 5 kΩ.

From this simulation it can be seen that the signal conditioning circuit for the voltage measurement indeed limits the input voltage of the op-amp. Although, this voltage threshold is not at the predicted 3.3 V. The reason for this is the low current that is allowed to flow through the Zener at normal conditions ($I_{Zener,max} = \frac{V_{in}}{50\text{k}\Omega} \approx \frac{40\text{ V}}{50\text{k}\Omega} = 0.8$). The value of the property V_{Zener} of the Zener diode decreases when decreasing the Zener current [13]. The consequence of this problem is that the resolution of the voltage measurement is somewhat smaller if the input voltage is high. However, this is not found to be a problem, since the input voltage is not that high very often.

Current measurement The current is measured similarly to the measurement of the voltage. Except, this time the high side current shunt monitor, INA169 from Texas Instruments$^{\text{TM}}$ is used. This component measures the current from the rectifier and converts it into a voltage which is fed into an RC filter followed by the op-amp buffer. The output voltage of this component will never be higher than $V^{+} - 0.9 \approx 3.6 - 0.9 = 2.7$[14]. Therefore, no overvoltage protection is applied in the current measurement circuit. The relation between measured current and output voltage of the INA169 is: $V_{\text{out}} = 0.28 \cdot I_{\text{shunt}}$. The INA169 current sensor is not part of the control system and will not be discussed in this thesis.

2.2.7 IO to Energy Management System (EMS)

The effective electrical resistance that the generator sees is controlled by a Pulse-width Modulation (PWM) signal and a MOSFET and resistor in series, as shown in figure 2.11. The PWM signal turns the MOSFET on or off, the duty cycle(D) determines the exact effective resistance, and is controlled by the MCU. The duty cycle is directly related to the torque of the generator, this is how the torque is controlled. The effective resistance seen by the generator is $R_{eff} = \frac{R_1}{D}$

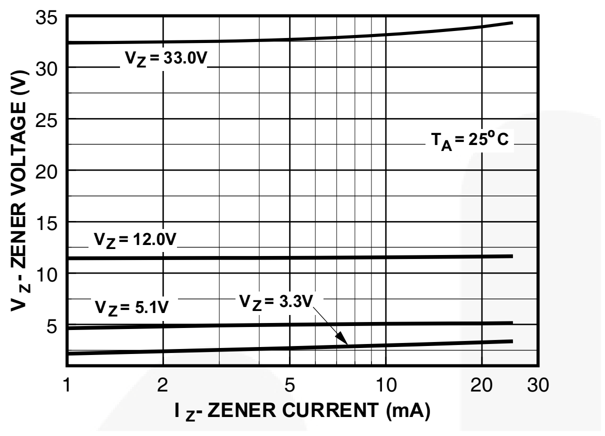


Figure 2.9: V_{Zener} vs I_{Zener}

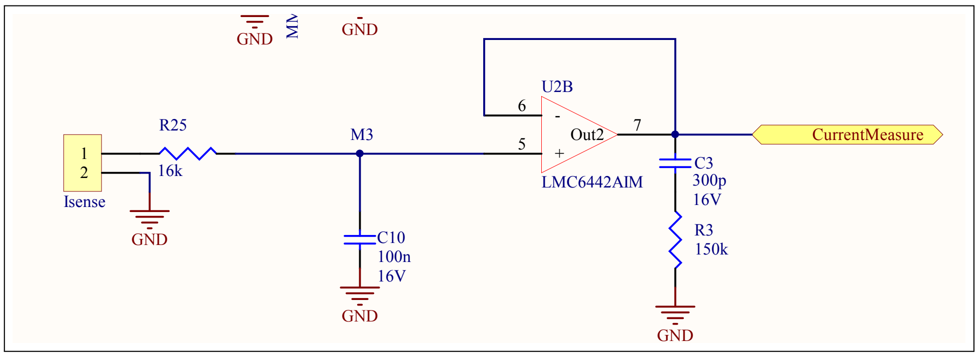


Figure 2.10: Current measurement circuit

and the current is related to the resistance by $I = \frac{V_{in}}{R_{eff}}$, in which V_{in} is the rectified generator voltage. Subsequently, the torque is related to the current by $\tau = KI$. Then,

$$\tau = K \cdot \frac{D \cdot V_{in}}{R_1} \tag{2.4}$$

As can be seen in equation 2.4, a larger duty cycle means the effective resistance is smaller, which produces more current and more torque, and vice versa.

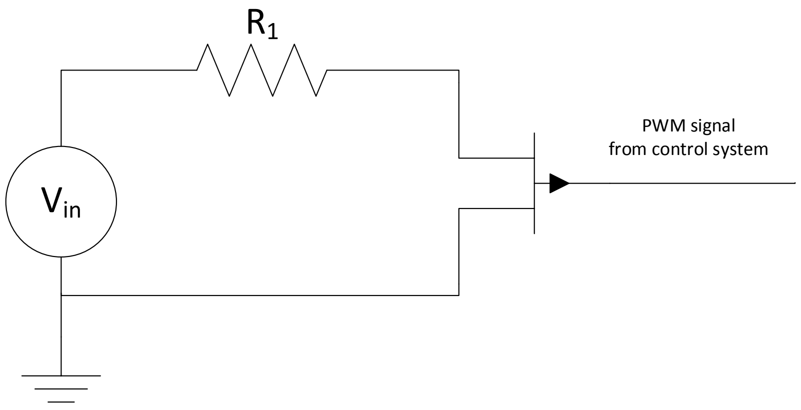


Figure 2.11: Circuit for controlling the generator resistance

To turn the MOSFET on, its gate capacitance must be charged, this takes time. This is modeled by a basic RC circuit as shown in figure 2.12. The gate capacitance is 1600 pF[15], and the resistance is 160 Ω. The formula for charging a capacitor is:

$$V_C = V_S(1 - e^{\frac{-t}{\tau}}) \tag{2.5}$$

Where $\tau = RC = 160 \cdot 1600 \cdot 10^{-12} = 256$ ns. After one τ, the V_C will be $0.63V_S$. After 4τ, V_C is $0.98V_S$. This takes 1024 ns. If for example, the PWM switching frequency is 100 kHz, every 5 μs the MCU output will change. Switching takes $\frac{1024 \text{ ns}}{5\ \mu\text{s}} = 20.5\%$ of the time. This can make the PWM signal look distorted.

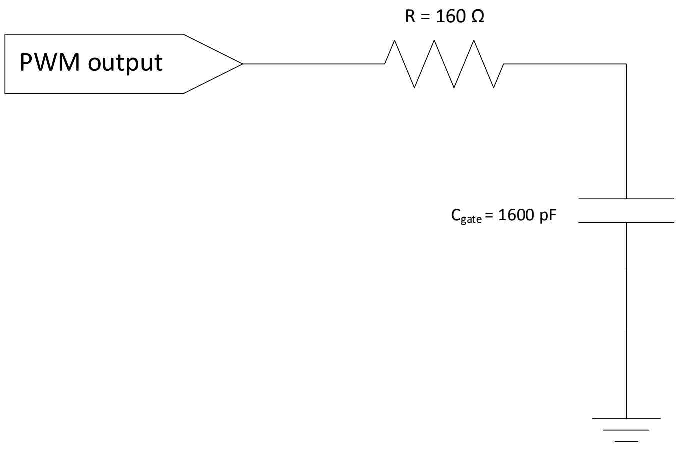


Figure 2.12: Model for turning the MOSFET on and off

2.2.8 Wireless communication

Feedback needs to be send to the user, e.g. to an app on their smartphone. The app could also be used to control the resistance. The feedback consists of power measurements, either as voltage and current separately, or as actual power. This does not have to be updated constantly. A refresh rate of 1 Hz is planned. For debugging purposes, the measured motor temperature will also be sent. If the user selects a different resistance in the app or with the buttons on the DISQ, this information must also be sent.

Several wireless communication options are considered in table 2.5 together with some options that are not considered quite useful for the system as they are very sensitive to disturbances. The communication technique should be widely supported, so that the system can be used in a commercial product. Zigbee® is therefore not a viable option. Other important aspects are costs, range and power consumption. The estimated minimum range is 10 meters. The ranges for both Bluetooth and Wifi are suitable, although Class 3 Bluetooth modules have too little range. Since Bluetooth modules are cheaper and use less power than Wifi modules[16][17], Bluetooth is the chosen communication technique. Several Bluetooth modules are considered in table 2.6.

	Zigbee	Bluetooth	Wifi	Ultrasonic	Infrared
Range	+	+	+	-	-
Power consumption	++	+	-	+	+
Smartphone supported	- -	++	++	-	- -
Costs	+	++	+	++	++

Table 2.5: Wireless communication options

There are a few options concerning the communication between the MCU and the BT module. First of all, a choice between parallel and serial communication can be made. A parallel connection would be established by 8 separate data wires. Additional wires can be used for a clock, to signal the direction of data flow or handshaking signals. Parallel communication can be faster than serial, but takes up much more space. Since the required data rate is very low, and a PCB will be designed for the system, serial communication is the better option for this application. Serial Peripheral Interface (SPI) needs 4 lines[18], and I2C is only half-duplex, which means that data cannot travel in two directions at the same time[19]. USB has a relatively complex protocol[20]. UART uses the least amount of lines, can operate in full duplex and is easier to implement, due to the available library from the MCU manufacturer Microchip[21]. UART is chosen as communication protocol, the Bluetooth modules listed in table 2.6 all have a UART connection.

Name	TX/RX current (mA)	Idle current (μA)	Price per 1 (€)	Price per 1 1000+ (€)
nRF8001	14/14	-	3,77	2,46
nRF51822	10/13	-	4,09	2,67
nRF51422	9,7/8	2,6	4,31	2,81
RN-42	30/45	500	17,25	17,25
RN-42 XVP	**30/45**	**3000**	**17,27**	**17,10**
RN4020	16/16	1500	8,03	6,90
BL620	10,5/-	3,5	13,62	12,28
BR-C46AR	50/40	1400	43,46	21,29
RFD22301	12/12	-	15,35	13,09
BLE113	14,3/14,3	-	13	11,7
PAN1721	14/14,7	-	13	10,64
AMS001	10,8/12,8	12	7,74	11,25

Table 2.6: Different Bluetooth modules

The chosen Bluetooth module is the RN-42 XVP, which may not have the best performance or price, but is due to its footprint and shipping time the best option for the prototype.

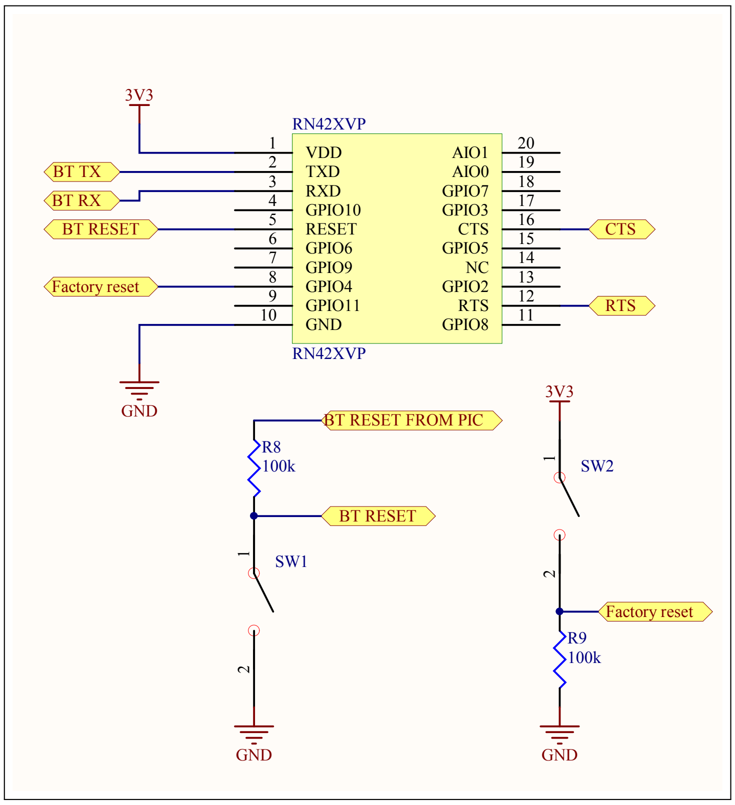


Figure 2.13: The connection between the MCU and the Bluetooth module

2.2.9 Onboard IO

The system provides access to 6 buttons, of which three are for the user and the other three are for debugging purposes. Two buttons are used to control the resistance. When 'RES+' or 'RES-' is high, the software will change the resistance if the button is still high after 1 ms. The software waits for the button to stop bouncing [22].

The third button is called 'BT CONNECT', which will allow the BT module to be powered for 10 seconds. During this time, a connection will have to be made, and the app will send an '@' character. Otherwise, the MCU will turn the BT module off. Turning the BT module off has two advantages: it consumes less power, and the BT module will not be always discoverable, which would make finding your own DISQ difficult if there are many DISQ's active in the neighborhood.

There are three debug LEDs which can be used for user feedback on the PCB after the product is finished.

2.2.10 Debug IO

Apart from the previously discussed user IO, also access to debugging IO is available on the control system. Debugging IO is the on-board IO that is not provided to the user. Its existence is purely intended for the developers and should not be needed for the user under normal conditions.

The input consists of three buttons. System reset, Bluetooth reset and factory reset for the Bluetooth module. In case of a problem, these buttons can be used. A system reset and Bluetooth reset can be imposed by pressing the corresponding buttons for at least 2 μs [3] and 160 μs [23], respectively. A factory reset is triggered by pressing the button on powerup and then release the button and press it two more times with a delay of 1 second between the switching [23].

The debug output consists of the three status LEDs.

2.2.11 Programming interface

The programming of the MCU is done with a PICkit^{TM} 2 device. This device is a programmer for PIC microcontrollers and is made by Microchip®. Apart from that, ICSP (In Circuit Serial Programming) is used, which enables programming without removing the MCU from the PCB. Removing the MCU is, in this case, almost impossible. Figure 2.14 shows the ICSP-circuit.

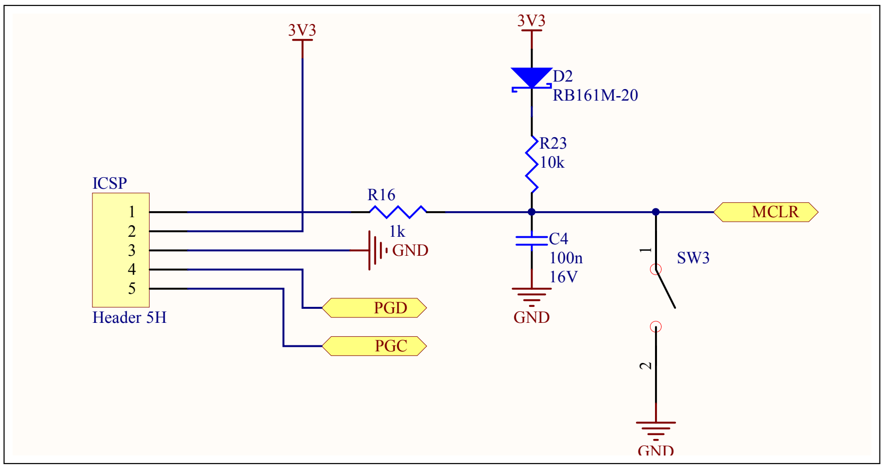


Figure 2.14: ICSP circuit

The PICkit^{TM} 2 is connected to the header ICSP. The output of the programmer is shown in figure 2.15.

PGD and PGC are programming data and clock respectively. V_{PP} is the programming voltage needed for the PIC on its MCLR pin to enter programming mode. V_{PP} is around 12 V

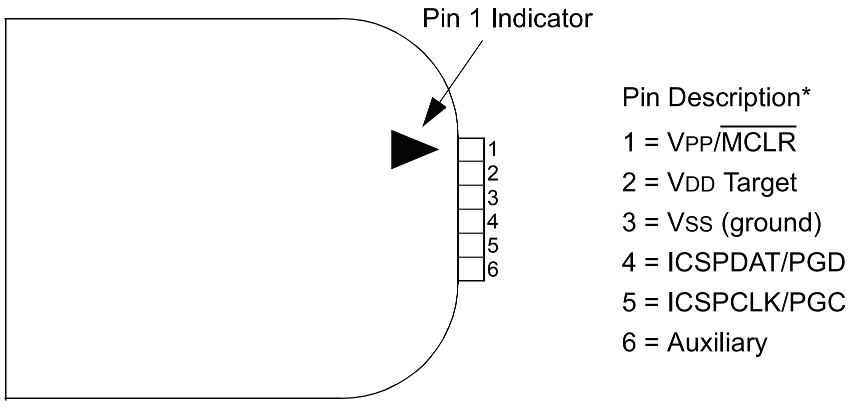


Figure 2.15: Output of the PICkit^{TM} 2 programmer

[24]. MCLR is also the Master Clear pin used for resetting the MCU when pulled to ground. This is done with SW3. R16 and R23 are current limiting resistors and C4 is used for debouncing SW3 and filtering high frequency noise. D3 is a Schottky diode which prevents current flowing from the PICKIT to the power supply voltage of the controller PCB.

2.2.12 PCB design

In figure 2.16 the designed PCB is shown. From upper left to lower right, the following components can be distinguished:

- A: Power input connector + power supply
- B: ICSP programming header + circuit
- C: Bluetooth^{TM} module
- D: V_{sense}, I_{sense} and T_{sense} input connectors with signal conditioning circuit and op-amps.
- E: Status LEDs
- F: MCU
- G: User and debug buttons
- H: I/O header (with PWM)

As can be seen, all different parts are connected via traces on the PCB. The designed PCB is a dual-layer PCB which means that it consists only of a bottom and a top layer. Both the top and bottom layers of this PCB contain a big ground plane: a plane of copper to make sure that a well defined ground is available anywhere in the circuit [5]. Vias (interconnections between layers on a PCB) are placed for four reasons:

- To provide a connection between a trace on one plane and the other plane
- To make a solid connection between the two ground planes (via stitching)
- To shield certain traces from the rest of the circuit (via shielding)
- To provide the developers with test points (numerated as M1, M2, etc.)

Via stitching should provide a low impedance connection between the ground planes and is used to help maintain a well defined ground in the entire PCB[25]. Via shielding is in this design only applied to the PWM-signals going out of the MCU. Via shielding provides a "wall" that helps reducing EM radiation from that particular place on the PCB[26]. The reason for shielding the PWM-signals is that they are high frequency signals which can lead to disturbances (e.g.

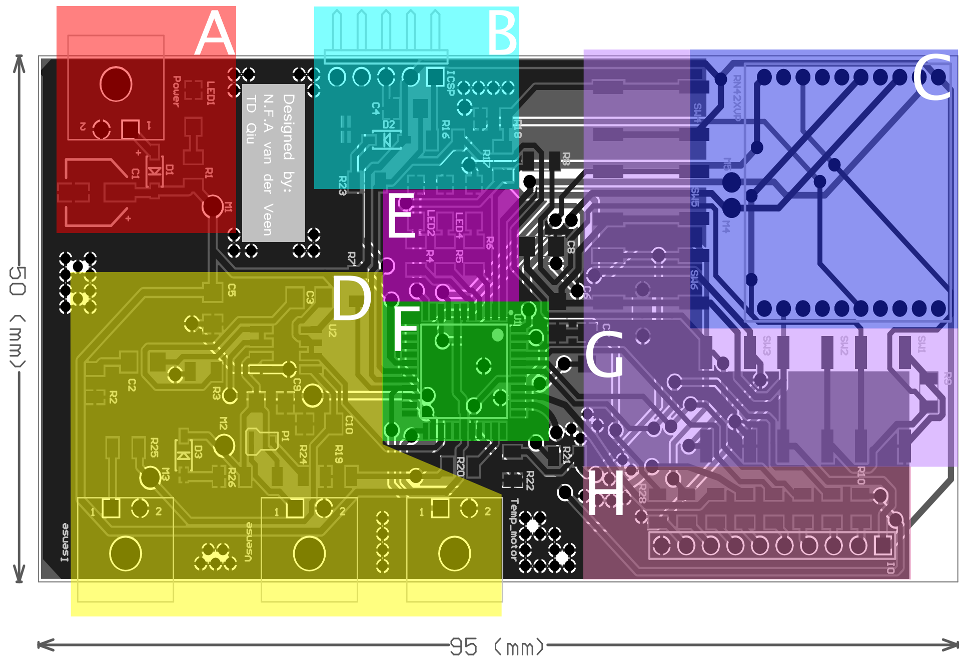


Figure 2.16: The designed PCB with color-marked areas

crosstalk) to other signals nearby. A closeup of both via stitching and shielding can be seen in figure 2.17. The perpendicular intersection of the two signal traces is also helps to reduce crosstalk.

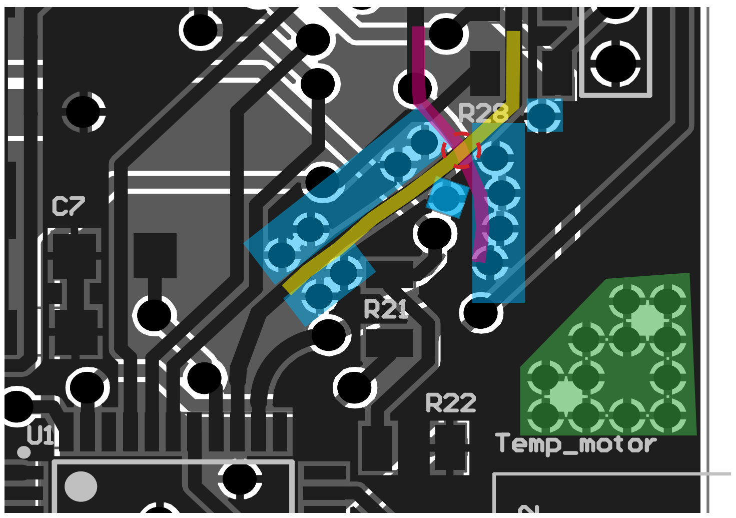


Figure 2.17: Via shielding and via stitching. The blue areas indicate shielding vias whereas the green area shows stitching vias. The yellow and magenta lines are PWM-signal traces. The red marker shows the perpendicular intersection of the two channels.

According to the data sheet of the Bluetooth^{TM} module, the antenna should be positioned away from ground planes to prevent disturbances [27]. This is indicated in figure 2.18.

2.3 Software

2.3.1 Introduction

The software of the control system interconnects basically everything. Most importantly, it connects the user to the physical hardware and controls that hardware accordingly. In the section

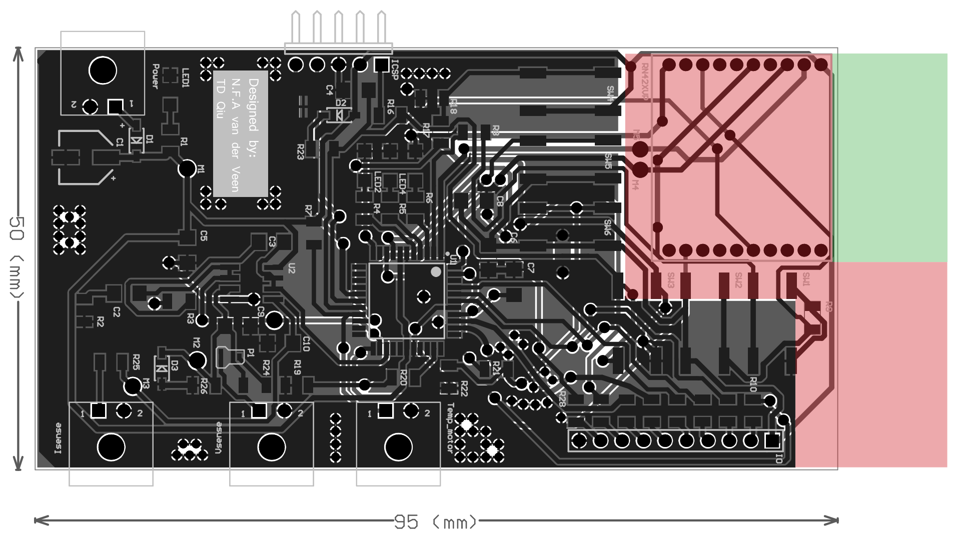


Figure 2.18: Antenna placement. The green area represents the antenna and the red area shows the places where a ground plane is absent to prevent disturbances.

”Software” of this thesis, three parts are discussed. First, the software that is programmed into the MCU is discussed, following by the software for the app (or GUI) and the data transfer algorithms.

2.3.2 MCU firmware

2.3.2.1 Firmware introduction

Firmware is the software that programmed as read-only into a programmable device. In this case this is the software programmed into the MCU. Without the firmware, the control system will not perform any actions. Therefore, this crucial element is discussed thoroughly in the following sections.

For writing the firmware, the MPLAB^{TM}-IDE from Microchip$^{®}$ is used and the firmware itself is written in C. Some functions in the firmware are implemented with the help of libraries[21]. When this is applicable, it will be discussed in the corresponding section.

Limitations The only limitations, apart from the limitations originating from the programming language and IDE itself, are the limited size of the RAM and ROM of the MCU. This has to be taken into account when developing the firmware.

2.3.2.2 Configuration

The MCU has to be configured in a way it performs optimally in the current application. There are important peripheral related configuration values, such as PWM frequency and ADC read frequency. Table 2.7 lists the initial and final values, the initial values may have changed after testing, which will be explained in chapter 3. The initial values will be explained in the next sections. In table 2.8 other configurations are shown and they are from now on referred to as ”miscellaneous configurations”.

	Oscillator	Prescaler Timer 2	PR2	Postscaler Timer 2	Interrupt frequency Timer 2	PWM frequency	ADC read frequency
Initial value	8 MHz	1:1	20	1:16	6250 Hz	95.24 kHz	195 Hz
Final value	8 MHz	1:4	20	1:10	2500 Hz	23.81 kHz	625 Hz

Table 2.7: Initial and final values for different parameters

Config	Function	Value	Description
FOSC	Oscillator selection	INTOSC_HS	Use internal oscillator, high speed used for USB (not used)
PWRT	Power up timer	ON	After a reset or power-up, the PWRT holds the device in reset so that the clock can stabilize
BOR	Brown out reset	OFF	The device will reset when the supply voltage is lower than BORV
BORV	Brown out voltage	3	BOR threshold, arbitrary value since it is not used
WDT	Watchdog timer	OFF	When the watchdog timer is not cleared for a certain period, it resets the device
WDTPS	Watchdog postscaler	32768	Determines the period after which the WDT resets the device
MCLRE	MCLR enable	ON	Used by programmer to put the device in programming mode
STVREN	Stack full/underflow reset	ON	Resets the device when the program stack is full/underflowed
LVP	Low voltage ICSP	ON	Use low programming voltage on MCLR pin
ICPRT	In-circuit debug/programming port	OFF	Alternative to the used in-circuit serial programming
XINST	Extended instruction set	OFF	Adds 8 instructions
CPx	Code protection bit x	OFF	Prevents reading the memory block from outside
WRTx	Write protection bit x	OFF	Prevents the MCU from writing its own program memory
EBRTx	Table read protection bit x	OFF	Prevents the MCU from reading other program memory blocks

Table 2.8: Miscellaneous configurations of the MCU

The Watchdog Timer(WDT) is not used, because we had difficulties with implementing it. The WDT is usually cleared at the beginning of each main loop iteration. This lets the WDT know the program is still running, and prevents it from resetting the device. However, the WDT did not seem to reset the device without the 'clear WDT' command. Due to time limitations, and the fact that the program has never frozen, we decided to not use the WDT.

The Extended Instruction Set (XINST) is mostly used for recursive code or code that uses a lot of subsystems or software components to run. This does not apply to our code, so XINST is disabled.

The Stack Full/Underflow Reset (STVREN) resets the device when the program stack is full or underflowed. Our program is quite simple, so we do not expect any exceptions of this sort, but STVREN is enabled for extra safety.

2.3.2.3 Oscillator

The primary clock source for the MCU comes from an internal oscillator (INTOSC). The frequency of this oscillator is 8 MHz, but can be lowered with a postscaler to 31.25 kHz, 125 kHz, 250 kHz, 500 kHz, 1 MHz, 2 MHz or 4 MHz. These frequencies can be tuned with 4 tuning bits to calibrate the clock frequency. The MCU also contains an internal RC oscillator (INTRC), which provides a nominal 31 kHz output. On device resets, the default output frequency of INTOSC is 1 MHz. If another clock frequency is required, this must be set in the initialization part of the firmware. INTRC will always be the clock source for the Watchdog Timer.

An external oscillator can also be used. The MCU has a phase-locked loop (PLL) block[28] that produces a fixed 96 MHz output from a fixed 4 MHz input. The input must be 4 MHz, which can be established by prescaling the external oscillator. The output can be postscaled to obtain frequencies of 48 MHz, 32 MHz, 24 MHz or 16 MHz.

The Timer1 module has its own oscillator, which can be used as a clock source as well. If this mode is used, both the CPU and the peripherals will be clocked from the Timer1 oscillator.

For now 8 MHz is chosen as oscillator frequency (F_{OSC}). This gives an oscillator period (T_{OSC}) of 125 ns.

2.3.2.4 Timers

The PIC18F4550 has 4 timer modules. Timer0 can be used in 8 or 16 bit modes, and has an 8 bit prescaler. Timer1 is a 16 bit timer and its internal oscillator can be used to run the device. Timer2 is an 8 bit timer and has a prescaler (1:1, 1:4 and 1:16) and a postscaler (1:1 through 1:16). Timer2 also generates a PWM output. Timer3 is a 16 bit timer.

Timers 0, 1 and 3 have selectable clock sources (internal or external) and can generate interrupts on overflow. Timer2 (figure 2.19) can set the PWM output pin high when the PR2 (writeable by software) value is matched, the timer can also generate an interrupt at this time. The frequency of the interrupts can be lowered by using a postscaler, but the PWM signal is not affected by the postscaler. A high prescale value lower the frequency of interrupts, but also lowers the PWM frequency, which may increase audible noise. So a low prescale value is better, but only if the interrupts can be handled sufficiently fast. The Timer2 interrupts are used to obtain a 1 Hz signal, therefore the interrupt frequency must be a whole number. The values chosen are listed in table 2.7, the final values represent the final values chosen after testing. The calculation of the PWM frequency will be explained later.

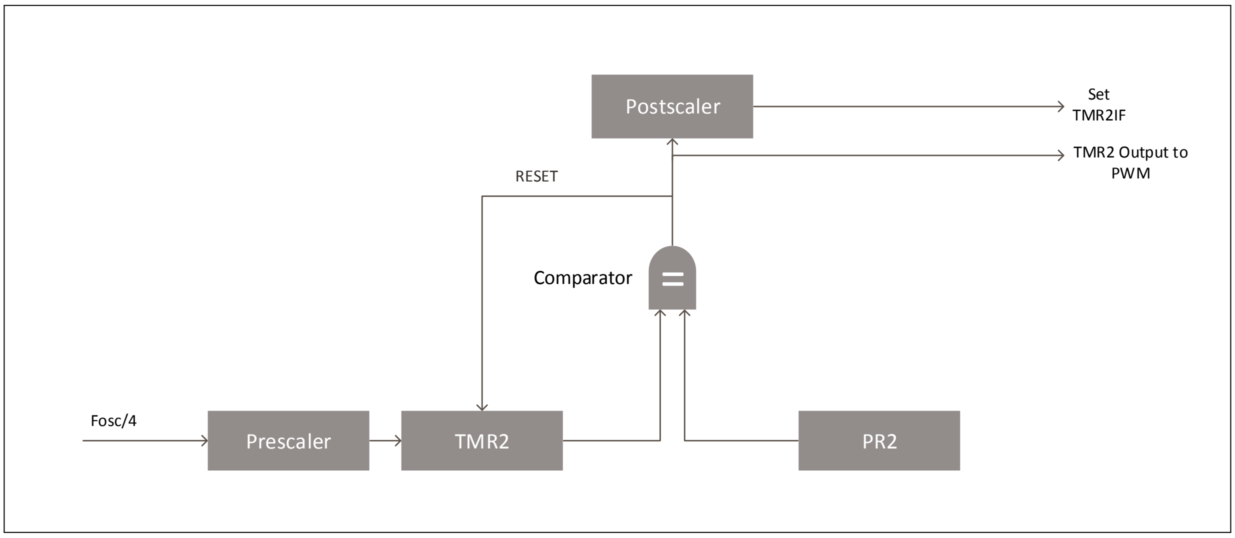


Figure 2.19: The simplified Timer2 module, TMR2IF is the Timer2 interrupt flag

2.3.2.5 ADC

The ADC module has 13 input channels and 10 bit resolution. The ADC can generate an interrupt when the conversion is done. The ADC operates on a different frequency (F_{AD}) then the rest of the MCU. The minimum ADC clock period (T_{AD}) is 0.7 μs. To calculate the minimum acquisition time, the equations below must be used. The input circuit of the ADC is shown in figure 2.20.
For the calculations, some assumptions are made: $C_{HOLD} = 25$ pF, $R_s = 2.5$ kΩ, Temp= 85°C (MCU max), $R_{IC} = 1$ kΩ, $V_{DD} = 3.3$ V$\rightarrow R_{SS} = 4$ kΩ. [3]

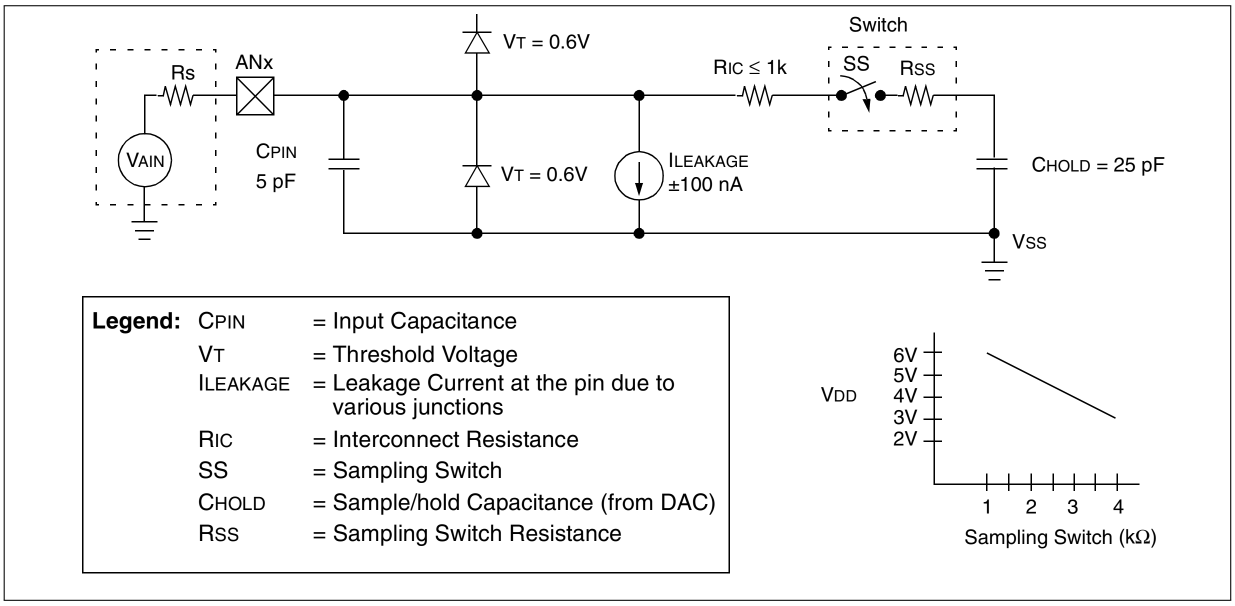

VT = 0.6V
Switch
Rs
ANx
RIC ≤ 1k
SS
RSS
VAIN
CPIN
5 pF
VT = 0.6V
ILEAKAGE
±100 nA
CHOLD = 25 pF
VSS
Legend: CPIN = Input Capacitance
VT = Threshold Voltage
ILEAKAGE = Leakage Current at the pin due to various junctions
RIC = Interconnect Resistance
SS = Sampling Switch
CHOLD = Sample/hold Capacitance (from DAC)
RSS = Sampling Switch Resistance
VDD
6V
5V
4V
3V
2V
1
2
3
4
Sampling Switch (kΩ)

Figure 2.20: The analog input model [3]

$$\begin{aligned} T_{ACQ} &= \text{Amplifier Settling Time} + \text{Holding Capacitor Charging Time} + \\ &\quad \text{Temperature Coefficient} \\ &= T_{AMP} + T_C + T_{COFF} \end{aligned} \tag{2.6}$$

$$T_{AMP} = 0.2\ \mu\text{s} \tag{2.7}$$

$$\begin{aligned} T_C &= -(C_{HOLD})(R_{IC} + R_{SS} + R_S)\ln(\frac{1}{2048}) \\ &= -(25 \cdot 10^{-12})(1000 + 4000 + 2500)\ln(0.0004883) \\ &= 1.43\ \mu\text{s} \end{aligned} \tag{2.8}$$

$$\begin{aligned} T_{COFF} &= (\text{Temp} - 25°\text{C})(0.02\ \mu\text{s}/°\text{C}) \\ &= (85°\text{C} - 25°\text{C})(0.02\ \mu\text{s}/°\text{C}) \\ &= 1.2\ \mu\text{s} \end{aligned} \tag{2.9}$$

The minimum acquisition time (T_{ACQ}) is 0.2 μs + 1.43 μs + 1.2 μs = 2.83 μs.

The T_{AD} can be chosen using a scaler: $T_{AD} = \{2, 4, 8, 16, 32, 64\} \cdot T_{OSC} = \{0.25, 0.5, 1, 2, 4, 8\} \cdot \mu s$. The T_{AD} is chosen to be 1 μs, which is higher than the minimum of 0.7 μs. This means F_{OSC} must be divided by 8.

The T_{ACQ} can be chosen as well: $T_{ACQ} = \{2, 4, 6, 8, 12, 16, 20\} \cdot T_{AD} = \{2, 4, 6, 8, 12, 16, 20\} \mu s$. The T_{ACQ} is chosen to be 4 μs, which also meets the requirement. This means that $T_{ACQ} = 4 \cdot T_{AD}$.

Filtering Since the temperature needs to be measured less often than V and I, it is measured only once in 17 times, during these 17 measurements, the voltage and current are each measured 8 times. This value can be easily configured using the 'ADC_SEQUENCE_SIZE' value: of a number N reads, the first read is a temperature measurement, and the rest is filled with V and I reads. Apart from the RC low-pass filtering in the V and I measurement channels, digital filtering is performed on all three measurement channels (V, I and T). This digital filter is in the form of a moving average filter. The filter size for each channel is variable, but in the final prototype the programmed sizes are shown in table 2.9. For the moving average filter, the filter period is given by $T_f = L_{filter} \cdot T_{measurement}$. Since the ADC reads with a frequency of 625 Hz, the time of one measurement is $\frac{1}{625} = 0.0016$ s. The filter periods are also given in 2.9.

Measurement channel	Filter size	Filter period (s)
V	16	0.055
I	16	0.055
T	64	1.6

Table 2.9: Filter sizes and periods for different measurement channels

2.3.2.6 PWM

The PWM module can be configured to have a single, half-bridge or full-bridge output. For this application, only a single output is used. The timing of the PWM signal is configured via Timer2. The formula for the period is as follows:

$$\text{PWM Period} = [(\text{PR2}) + 1] \cdot 4 \cdot T_{OSC} \cdot (\text{TMR2 Prescale Value}) \tag{2.10}$$

Table 2.10 lists several possible PWM frequencies. A low PWM frequency may increase audible noise. The audible frequency band is 20 Hz to 20 kHz. A frequency of 100 kHz is aimed for.

When the value of Timer2 is equal to PR2, Timer2 is cleared and the PWM output pin is set high (unless the PWM output is configured to be active-low).

The duty cycle can be specified in 10 bits. When Timer2 is equal to the specified duty cycle, the output pin is set low (unless the PWM output is active-low). The PWM duty cycle can be calculated using the following equation:

$$\text{PWM Duty Cycle} = [\text{PR2+1}] \cdot T_{OSC} \cdot (\text{TMR2 Prescale Value}) \tag{2.11}$$

The PWM duty cycle can range from 0 to $1023 \cdot 125$ ns $\cdot\, 1 = 127.9\mu s$, with a resolution of 125 ns.
The resolution of the PWM is given by the following formula:

$$\text{PWM Resolution (max)} = \frac{\log(\frac{F_{OSC}}{F_{PWM}})}{log(2)} \text{ bits} \tag{2.12}$$

Timer 2 prescaler	PR2	PWM period	PWM frequency	PWM resolution (bits)
1:1	0	500 ns	2 MHz	2
1:1	4	2.5 μs	400 kHz	4
1:1	9	5 μs	200 kHz	5
1:1	19	10 μs	100 kHz	6
1:1	20	10.5 μs	95.24 kHz	6
1:1	255	2048 μs	488 Hz	14
1:4	20	42 μs	23.81 kHz	8

Table 2.10: Different possible PWM frequencies, 95.24 kHz is chosen, as listed in table 2.7

2.3.2.7 UART

The UART module features basic TX/RX IO, which are used to send and receive data. Arbitrary IO ports can be used for CTS/RTS if hardware flow control is needed. Since the chosen Bluetooth module has hardware flow connections, two MCU ports are chosen to serve as hardware flow control. However, since the communication will be asynchronous, full duplex and quite sparse, hardware flow control is actually not used.

The UART module can generate interrupts when a byte is transmitted, and/or when a byte is received. For this application, transmission interrupts are turned off, since it is not important to know when a byte is transmitted. Receive interrupts are turned on, so that an interrupt handler can read the data immediately.

Baud rate is the amount of bits that can be transmitted per second. The two modules must agree on a baud rate, since they do not share a clock for timing.

2.3.2.8 Interrupts vs polling

The program needs to know if something is done, so that the next step can be taken. An example is the ADC conversion, the firmware can start a conversion by setting a flag. The ADC module will also set a flag when the conversion is done. But how does the firmware know the conversion is done? There are two ways: polling and interrupts. When polling, the firmware looks as much as possible at the flag. This is not very efficient since the flag will be low for most of the time. The other way is via interrupts. When enabled, the ADC module will trigger an interrupt, which interrupts the regular program loop. The executed code is called an 'interrupt handler' or 'interrupt service routine (ISR). When the ISR is finished, the main loop will continue execution. The ISR must be short, since it prevents the regular program from running. Usually, ISRs just indicate an interrupt by setting a flag, instead of handling the interrupt itself. The program loop will then take care of the rest. An example is given in code snippet 2.1.

Listing 2.1: How interrupts are used

```
int g_ADCDone;  // global flag to indicate an interrupt has been generated

// regular program loop
while(1){
    if(g_ADCDone){
        ReadADCResult();
    }
}

void InterruptHandler(void){
    if(Interrupts.ADC == 1){
        Interrupts.ADC = 0;       // clear interrupt flag
        g_ADCDone = 1;            // signal main loop
    }
}
```

The flowcharts for the main loop and interrupt handler are shown in figures 2.21 and 2.22.

2.3.2.9 Smart Resistance

An extra feature of the "new" DISQ is the ability to provide constant torque to the user. In this thesis, this feature is called "Smart Resistance" (SR). This means that the user will experience a constant "resistance" when he or she is exercising, no matter how fast the cord is being pulled. The way to implement this feature is to use a PID (Proportional-Integral-Derivative) controller to control the current flowing from the generator, and therefore the torque. The proportional part works intuitive: if there is an error, try to adjust the output by sending or changing a control signal, in this case the PWM duty cycle. The integral term accumulates the past errors, to detect whether the error actually becomes small enough. The derivative term considers the rate of change in the error and uses this to predict the future error. The derivative term is not used very often, since it is quite sensitive to noise[29]. Since the measured output is averaged, some noise has already been filtered, this makes using the derivative term a possibility.

Recall that the relation between torque and current is given by:

$$\tau = K_1 \cdot I \tag{2.13}$$

And the relation between voltage and angular velocity is given by:

$$V = K_2 \cdot \omega \tag{2.14}$$

In which K_1 and K_2 are constants.

A PID controller can, as stated before, control the current in such a way that it is constant over the entire voltage swing. In other words: the torque is constant over the entire angular velocity swing. To control the current over a variable voltage, the resistance has to be controlled.

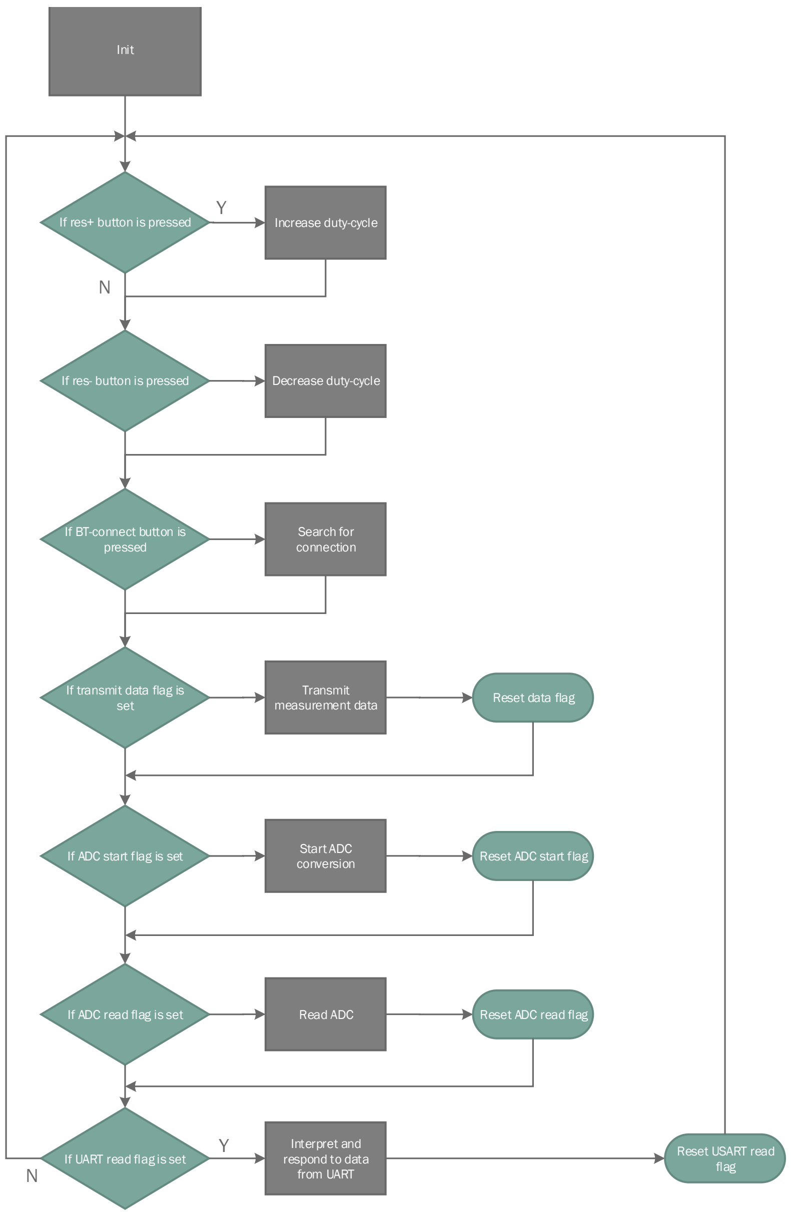


Figure 2.21: The flowchart for the main loop

An this is exactly what the control system described in this thesis is capable of. The MCU measures the voltage and current continuously and, to provide a constant current, it calculates what the resistance should be. After that, it changes the duty cycle of the PWM signal accordingly.

Figure 2.23 shows the block diagram of the PID controller. The reference is determined by the user, and the measured output is the output of the averaged measurement. Both V- and I_{sense} can be used. A higher voltage means a higher resistance to keep the current constant. Since the MCU is able to directly control the effective resistance, it should be possible to base the PID controller on V_{sense}. However, since the current should be constant, we chose to measure the current.

If the measured current is lower than the reference, the duty cycle should increase. This lowers the effective electrical resistance, and increases the current.

A problem arises when the user is momentarily not exercising. For example when the cord of the DISQ retracts and the user prepares for a new cord pull. In the "pause" in movement between these events, no current flows and the integral part of the PID controller integrates an error of $I_{ref} - 0 = I_{ref}$. This will set the physical resistance to the highest possible value because the PID controller "wants" to increase the current drastically. In practice, this results in a maximum physical resistance setting, every time the user begins pulling the cord.

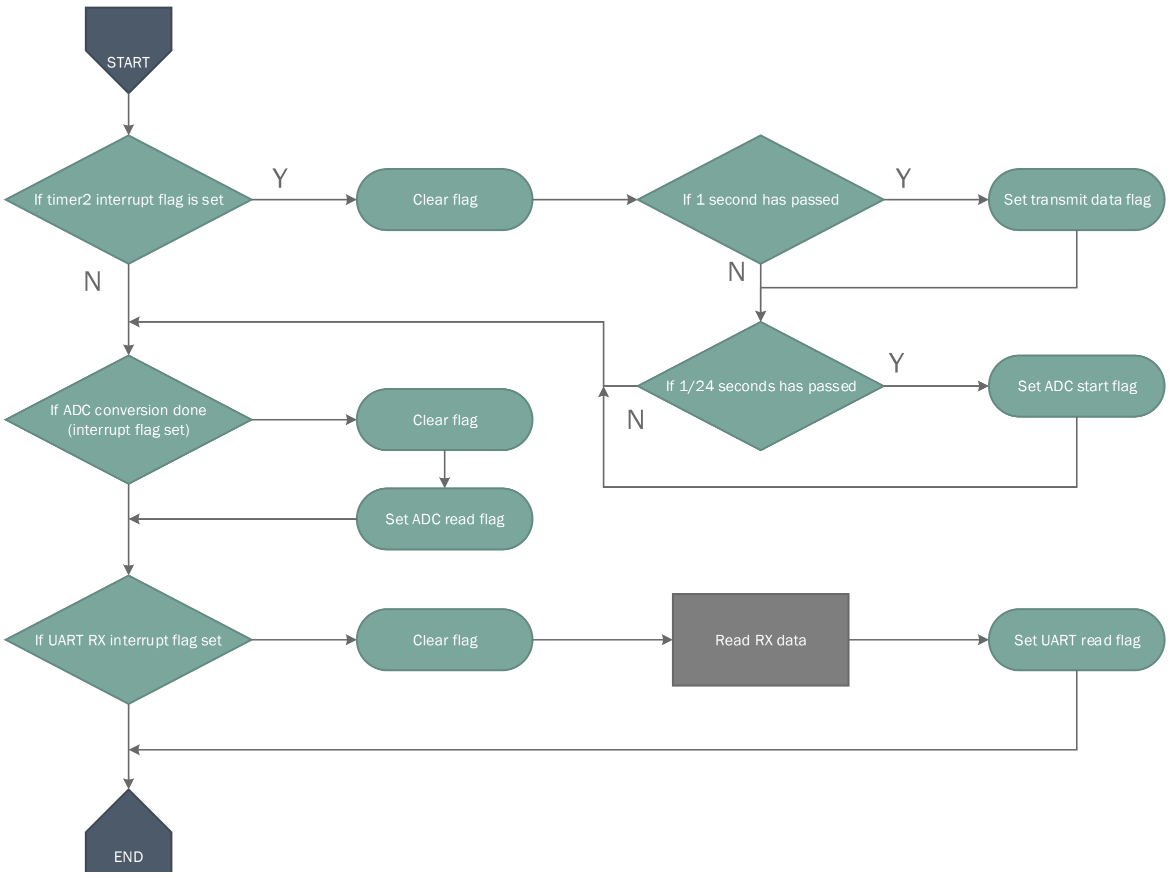


Figure 2.22: The flowchart for the interrupt handler

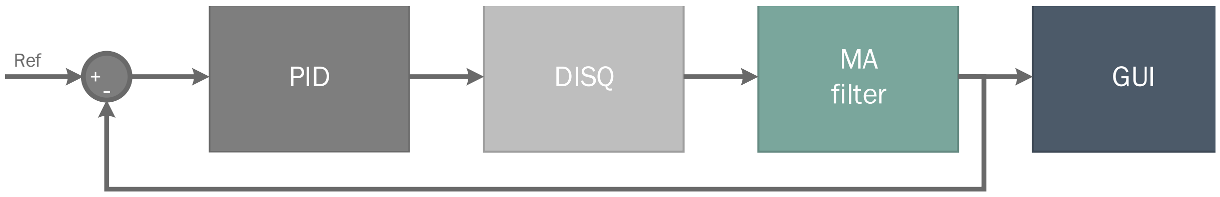


Figure 2.23: The block diagram of the PID controller

To solve this problem, the MCU can set the resistance to the reference value (the resistance value initially set by the user) every time the current is zero.

2.3.2.10 GUI

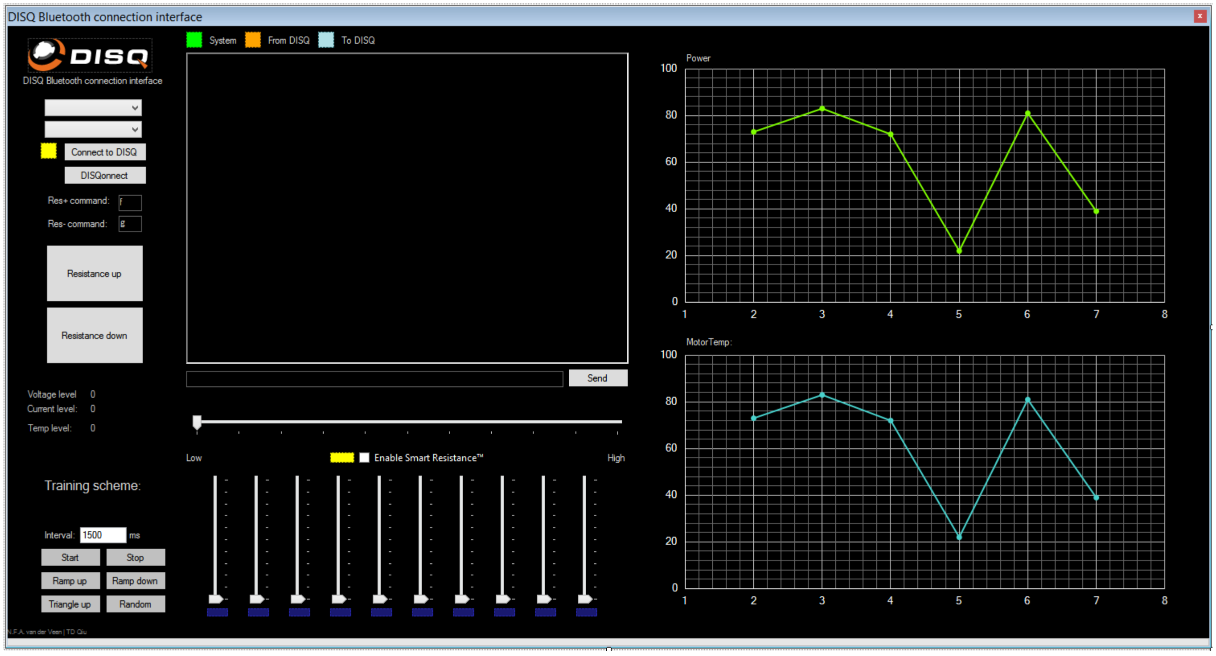
Introduction The GUI, or Graphical User Interface, is the software running on a computer to provide a connection to the control system via Bluetooth^{TM}and send and receive messages via this connection. It can be seen as the ”prototyping version” of the app that will run on smartphones. The software is entirely written in Visual C++ and will likely run on every Windows PC with access to Bluetooth^{TM}.

Features The GUI has many features as it provides the developer with:

- Options for establishing the connection to the control system (select COM port and define baud rate)
- A console in which system messages, messages to the control system and messages from the control system are displayed in different colors
- The realtime voltage, current and temperature values

- A slider for adjusting the resistance of the DISQ
- A sequencer like control block for creating custom resistance patterns for the DISQ (i.e. a training scheme creator)
- A checkbox for enabling and disabling "Smart Resistance"
- A graph showing the generated power over time
- A graph showing the motor temperature over time

The GUI interface is shown in figure 2.24.



Res+ command: f
Res- command: g
Voltage level 0
Current level: 0
Temp level: 0
Interval: 1500 ms

Figure 2.24: The GUI interface

The more in-depth information about how data is transferred is discussed in section 2.3.3.

2.3.3 Bluetooth

2.3.3.1 Connection configuration

Different baud rates are compatible with the used Bluetooth$^{\mathrm{TM}}$ module from a minimum of 1200 bps up to 921 kbps [27]. For the control system 115.2 kbps is chosen since this was the default for the module and worked without any problems.

The MCU has to be configured in order to communicate at the specified baud rate. This is done by setting the value of the SPBRGH:SPBRG registers. According to the datasheet[3], a baud rate of 115.2 kbps in combination with an 8 MHz clock frequency can only be established using a high baud rate (BRGH = 1) and the 16 mode of the baud rate generator (BRG16 = 1). The calculation for the SPBRGH:SPBRG registers is as follows:

$$Br = \frac{F_{osc}}{4(n+1)} \tag{2.15}$$

in which Br is the baud rate, F_{osc} is the clock frequency of the MCU and n is the value of the previously mentioned registers. Solving for n results in:

$$n = \frac{8000000}{4 \cdot 115200} - 1 = 16.36 \tag{2.16}$$

The actual baud rate is $Br = \frac{8000000}{4(16+1)} = 117647$ which is 2.08% higher than the required baud rate. Testing will show later that this is not a problem.

2.3.3.2 Sending and receiving data

Since the Bluetooth^{TM} module only accepts bytes on its UART RX pin, the data send must be encoded. The V-, I-, T-measurements are all 10 bits. The 10 bit measurement data from the V-, I- and T-measurements is split into two bytes. In this case, this means that 5 bits are in one byte and the other 5 bits are in another byte. What is left are three bits in each byte. These three bits are used as a tag to help the app/GUI identify which data is sent. The app/GUI reads the two bytes and extracts the tag bits. Finally, the data is stored in an array in which the index of the data is the decimal value of the tag bits. An overview of this process is given in figure 2.25.

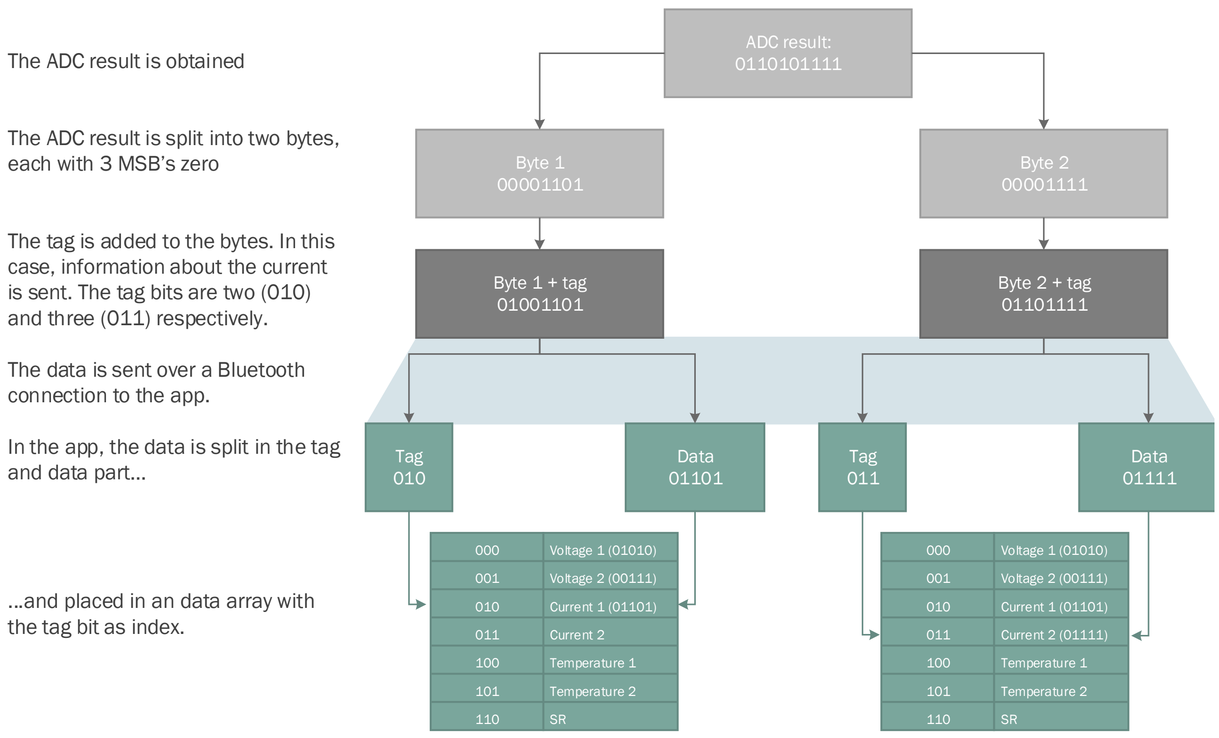

The ADC result is obtained
The ADC result is split into two bytes, each with 3 MSB's zero
The tag is added to the bytes. In this case, information about the current is sent. The tag bits are two (010) and three (011) respectively.
The data is sent over a Bluetooth connection to the app.
In the app, the data is split in the tag and data part...
...and placed in an data array with the tag bit as index.
ADC result: 0110101111
Byte 1 00001101
Byte 2 00001111
Byte 1 + tag 01001101
Byte 2 + tag 01101111
Tag 010
Data 01101
Tag 011
Data 01111
000 Voltage 1 (01010)
001 Voltage 2 (00111)
010 Current 1 (01101)
011 Current 2
100 Temperature 1
101 Temperature 2
110 SR
000 Voltage 1 (01010)
001 Voltage 2 (00111)
010 Current 1 (01101)
011 Current 2 (01111)
100 Temperature 1
101 Temperature 2
110 SR

Figure 2.25: The encoding and decoding schemes

The data received from the app consists mostly of resistance commands. The app will tell the MCU which resistance is needed. The app will also send a special character when a connection is established for the first time. This will tell the MCU to keep the BT module powered. All information from the app to the DISQ is encoded into characters as seen in table 2.11.

Character	Decimal ASCII code	Bitstring	Meaning
@	64	01000000	BT connection established
0	48	00110000	Resistance level 0
1	49	00110001	Resistance level 1
2	50	00110010	Resistance level 2
3	51	00110011	Resistance level 3
4	52	00110100	Resistance level 4
5	53	00110101	Resistance level 5
6	54	00110110	Resistance level 6
7	55	00110111	Resistance level 7
8	56	00111000	Resistance level 8
9	57	00111001	Resistance level 9
a	97	01100001	Resistance level 10
+	43	00101011	Turn Smart Resistance[1] on
-	45	00101101	Turn Smart Resistance off
!	33	00100001	BT connection terminated

Table 2.11: The different commands[30] and their meaning
[1] as discussed in section 2.3

Chapter 3

System testing

3.1 Introduction

The designed hardware and software discussed in the previous parts of this thesis are tested and important signals are measured. Mainly, these measurements were done with the use of a PicoScope® 2204A from PicoTechnology® together with the PicoScope® 6 software. The PicoScope® is a two channel, 10 MHz PC USB-oscilloscope with an integrated AWG or Arbitrary Waveform Generator. The software provides an interface to the oscilloscope and allows to export data, among other things. In table 3.1 some specifications of the device are given.

Property	Value	Unit
Bandwidth (-3 dB)	10	MHz
Max sampling rate	100	MS/s
Buffer memory	8	kS
AWG max frequency	100	kHz
Timebase accuracy	± 100	ppm
Noise	< 150	μV RMS
DC accuracy	±3	% of full scale

Table 3.1: Some properties of the PicoScope® 2204A [31]

3.2 Buttons

There are 6 buttons on the PCB. Three user buttons and three debug buttons. Apart from the BT factory reset, every button was tested.

Test method and setup The three user buttons are connected to IO pins of the MCU. The IO pin can be read in the software, after which a status LED is turned on. The LED will stay on as long as the button is pressed.

The two reset buttons can be tested by pushing them, and checking if the corresponding module is active. The BT module will have a LED blinking at 1 Hz if it is powered, and the MCU will turn the status LEDS on.

Test results The RES- and BT CONNECT buttons worked as expected, but the RES+ didn't work. It turned out that the RES+ pin was configured to be an analogue input pin.

The reset buttons worked as expected. The respective module respond seemingly instantaneous.

Test discussion Due to limited flexibility in the MCU, the RES+ pin can't be a digital pin if V_{sense} and I_{sense} aren't digital too. The choice here is either power measurements, or using the RES+ button. We temporarily made the input pins digital, and tested RES+ again. The button itself works, we only have a wrong configuration. It can be solved by treating the RES+ IO pin as analog, and connecting it to the ADC. Reading the digitized value will tell the MCU if the button is pressed. This process will be quite fast, so the apparent functionality might not have been affected. However, this is not implemented or tested.

3.3 Measurement channels

3.3.1 Max ADC reads per second

The internal analog to digital converter of the PIC18F4550 is tested on maximum read frequency.

Test method and setup Program code is written to measure how much time it takes to perform a certain number of ADC conversions and is shown in listing 3.1. A stopwatch is used to determine the time it takes to complete 50000 ADC conversions.

Listing 3.1: Code to test the maximum ADC reads per second

```
while(1){
    if(ADCStartFlag){
        StartADCConversion();
    }

    if(ADCReadFlag){
        Isense = ReadADC();
        ADCStartFlag = 1;
        ++ADCCounter;
        if(ADCCounter > 39999){
            STATUS_LED1 = 1;
        }
    }
}

void InterruptHandler(void){
    if(ADCConversionDone){
        ADCReadFlag = 1;
    }
}
```

Test results The average time over three measurements is 63 seconds, assuming a human reaction time of 0.3 seconds, this corresponds to an ADC max read frequency of $\frac{50000}{62.7} = 797 \approx$ 800 Hz.

Test discussion The ADC can perform about 800 reads per second. Assuming the temperature does not need to be measured frequently, the voltage and current can be measured about 400 times per second. The ADC actually reads 625 times per second. This corresponds to 293 V and I reads, and 39 T reads per second. The length of their moving average filters is 16, 16 and 64 for V, I and T respectively. The V and I filters average over 55 ms, and the T filter over 1.6 seconds.

Initially, the global system interrupt frequency was 6250 Hz. However, at that rate, communication to the GUI was problematic. Inconsistent data values are received and a lot of noise appeared to be present on the data. It was tried to lower the interrupt frequency from 6250 to 2500, in order to provide more time for the MCU to process all actions. This solved the issue. Due to the lower interrupt frequency, another ADC frequency had to be chosen. Since the max ADC frequency is 800 Hz, a frequency of $\frac{2500}{4} = 625$ Hz is chosen.

3.3.2 Temperature measurement

The NTC accuracy is tested and calibrated.

Test method and setup The NTC is connected to the control system such that the output values of the ADC are displayed in the GUI. Then, the NTC is exposed to different temperatures. These temperatures were measured with a Basetech^{TM} MINI 1 Infrared Thermometer. The according ADC values are recorded.

Test results The test results are shown in figure 3.1.

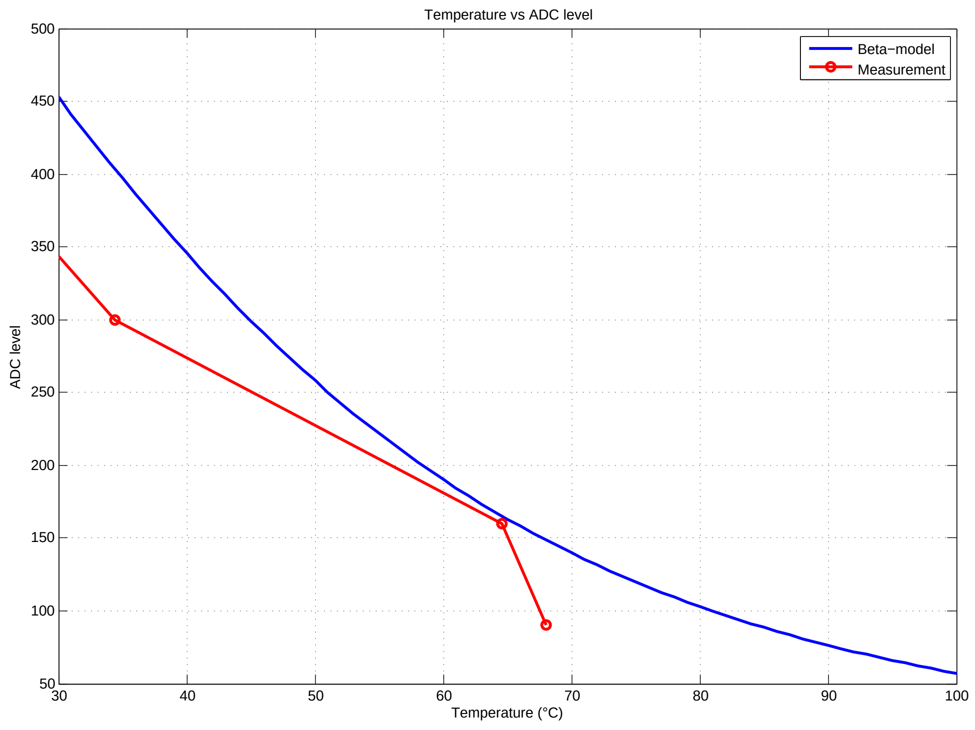


Figure 3.1: NTC temperature and ADC read value. Both the measured level and the level according to the Beta-model are shown.

Test discussion As can be seen from figure 3.1, the calculated ADC value and the measured ADC value differ significantly. This can be explained by multiple factors:

- The measured temperature is not the actual temperature inside the NTC. Because an IR thermometer is used, an area is measured instead of a (contact) point
- The accuracy of the used thermometer is not high enough. It provides an accuracy of ± 2 °C or ± 2 % [32]
- A time delay between the temperature reading and the ADC value recording is present which enables the NTC to cooldown between the different readings

3.3.3 Op-amp

The op-amps as voltage follower in the V_{sense} and I_{sense} circuits are tested.

Test setup The PicoScope® is connected to the output of both op-amps. After that, both measurement channels are connected to V_{DD} and the scope is set to AC coupling.

Test results Measuring as described above, the following results are obtained (figure 3.2).

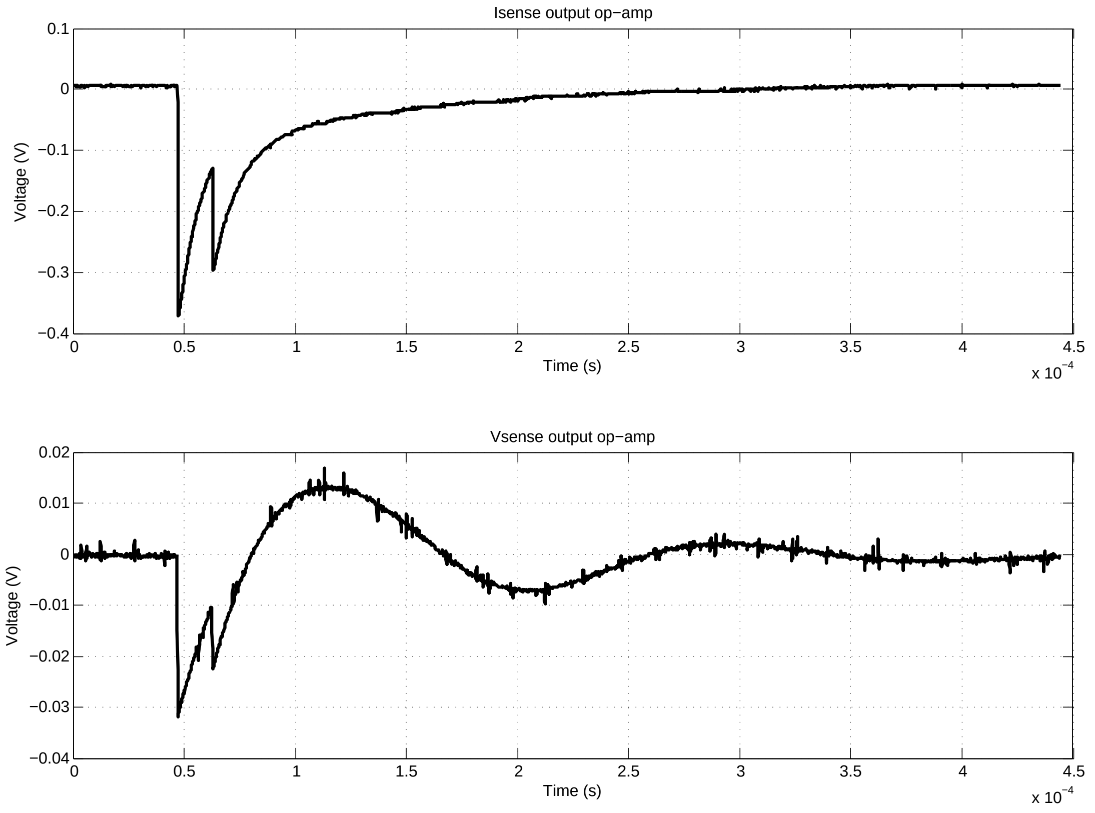


Figure 3.2: Measurements on the output of both op-amps

Test discussion Figure 3.2 clearly shows that on both op-amp outputs a voltage spike is present. However, the spikes differ in magnitude. This might be explained by the fact that the input voltages are different. The input voltage of I_{sense} is 3.3 V whereas the input voltage of V_{sense} is approximately $\frac{3.3}{12} = 0.275V$ where the value 12 resembles the ratio of the voltage divider. The ratio of the magnitude of the spikes is also approximately 12. The shape of the waveforms after the spike is also different; the V_{sense} output oscillates where the I_{sense} does not show this behavior. The shape of the spikes themselves are suspiciously similar. The reason for these spikes could lie in the non-linearity of the transistors inside the op-amp causing crossover distortion [33]. This is, however, not very likely, since this type of distortion generally shows another waveform shape [34].

3.3.4 Power measurement input circuitry transfer

To verify the bandpass filters' -3 dB frequencies, some measurements have been done.

Test setup The PicoScope® function generator was attached to the V_{sense} or I_{sense} terminals as input. The input was set at $0.5 + 0.5\sin(2\pi f t)$. The output was measured at the label Voltage- or CurrentMeasure. These values were converted to dBV and subtracted to get the gain. To get a transfer function, several input frequencies were used.

Test results The results are shown in 3.4.

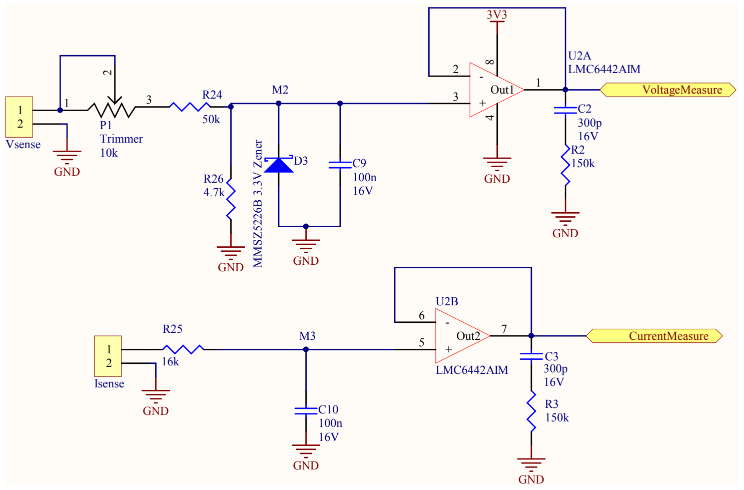


Figure 3.3: The V- and I_{sense} input circuitry

Test discussion The transfer of I_{sense} starts above 0 dB, this means the output voltage is higher than the input. The op-amp must have caused this, since there are no other active elements in the circuit. Figure 3.5 shows the step response of the op-amp in voltage follower mode, with different load capacities. The op-amp is connected to the output, which according to figure 2.20 is about 30 pF. The oscillation in the step response is in that case quite small, it doesn't explain the gain. We assume it's a measurement error.

The transfer of V_{sense} starts at -10.2 dB which is a gain of 0.0955, this could caused by the voltage divider. The theoretical minimum division ratio is 0.0726 and the maximum division ration is 0.0859. The slightly higher gain can be caused by measurement errors or inaccuracies of the resistance values.

3.4 PWM output

The PWM signal outputted by the control system is tested.

Test method and setup The setup for the PWM testing is as follows. The controller is connected to the PICkit^{TM} and is supplied by 3.3 V. The PicoScope® is connected to two different outputs and ground. The two measured outputs are:

- The PWM output (IO5, header 10, pin 6 2.3)
- A non-PWM output next to the PWM output (IO6, header 10, pin 7 2.3)

Also, the spectrum of the PWM signal is obtained.

Test results In figure 3.6, the results of the described test is shown.

The obtained spectrum of the PWM signal is shown in figure 3.7 together with an indication of the audible frequency domain.

Test discussion It can be seen in figure 3.6 that crosstalk is noticeable on outputs close to the PWM signal. The magnitude of the "ghosting" signal is around $\frac{0.008}{3.3} * 100\% = 0.24\%$ of the actual signal. This is found to be negligible, since this value is very small and, in addition, the outputs close to the PWM signal are not used in this design.

The spectrum shown in figure 3.7 clearly shows the PWM frequency as the biggest peak at approximately 95 MHz, also it is clear that a lot of harmonics are present in the PWM signal. Also in the audible domain. The peaks in the audible domain are at approximately 260 Hz, 6.2 kHz, 12 kHz and 18 kHz respectively.

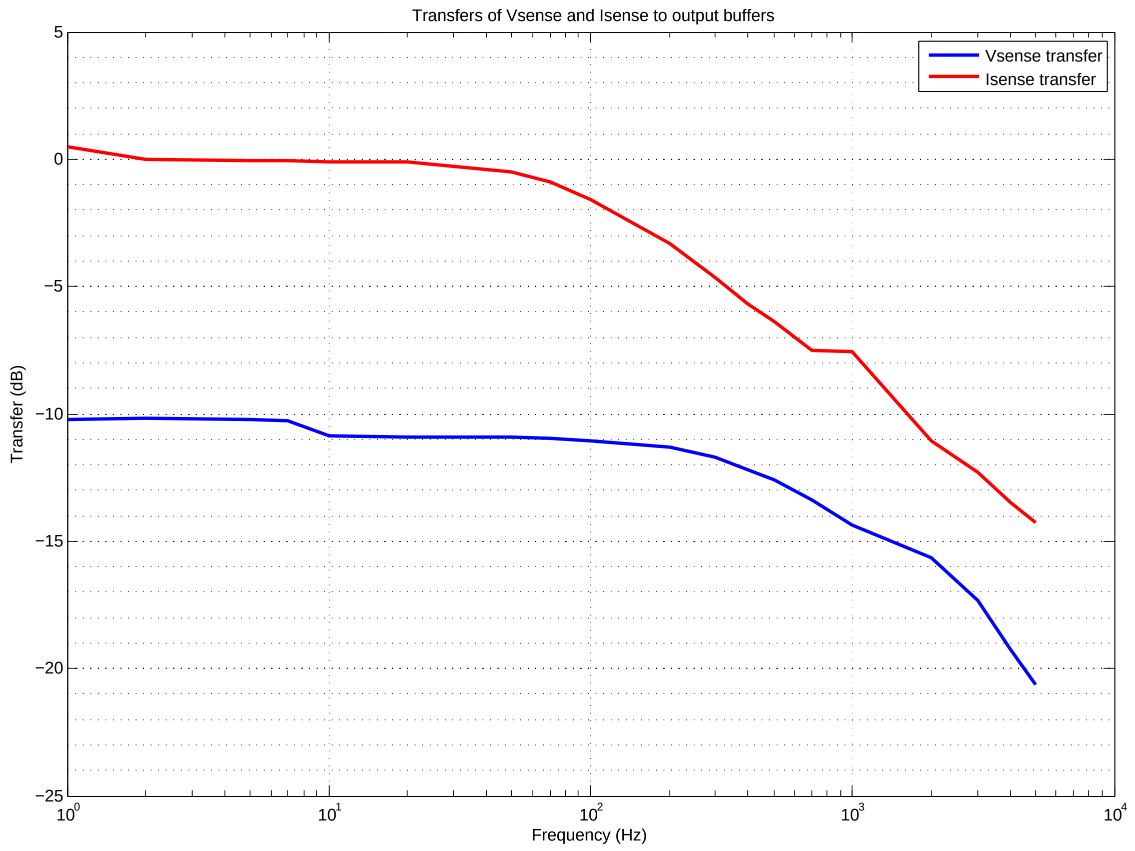


Figure 3.4: Transfer functions of V_{sense} and I_{sense} input circuits

Testing the PWM signal in combination with the generator and energy management system shows that with a PWM frequency of 95.24 kHz, there is no audible noise. However, the PWM signal looks quite distorted. Further testing shows that at 6 kHz, there is audible noise. A PWM frequency of 23.8 kHz was found to be a good frequency, as there was no noise and little distortion.

3.5 Power consumption

The power consumption of the control system as a whole is measured, to get insight in the power consumption of different modes of operation of this system. The different modes of the Bluetooth$^{\text{TM}}$ module are tested.

Test method and setup The consumed power is represented by the current flowing into the controller and this is measured with the use of a low side shunt resistance of 0.33 Ω. A bench top power supply unit (PSU) supplies power to the control system. The voltage of the PSU was set to approximately 3.3 V. The voltage across this resistance is measured with the PicoScope$^{®}$ and filtered in MATLAB$^{\text{TM}}$ with the function *smooth*. The current can then be obtained by following Ohm's law ($I = \frac{V}{R}$).

The test setup is shown in figure 3.8 is which PSU is the power supply feeding the control system.

Test results In figure 3.9 the obtained measurement results are shown.

Test discussion There is a noticeable difference in the amount of drawn current by the control system for different modes of operation. On top of that, the shape of the different waveforms

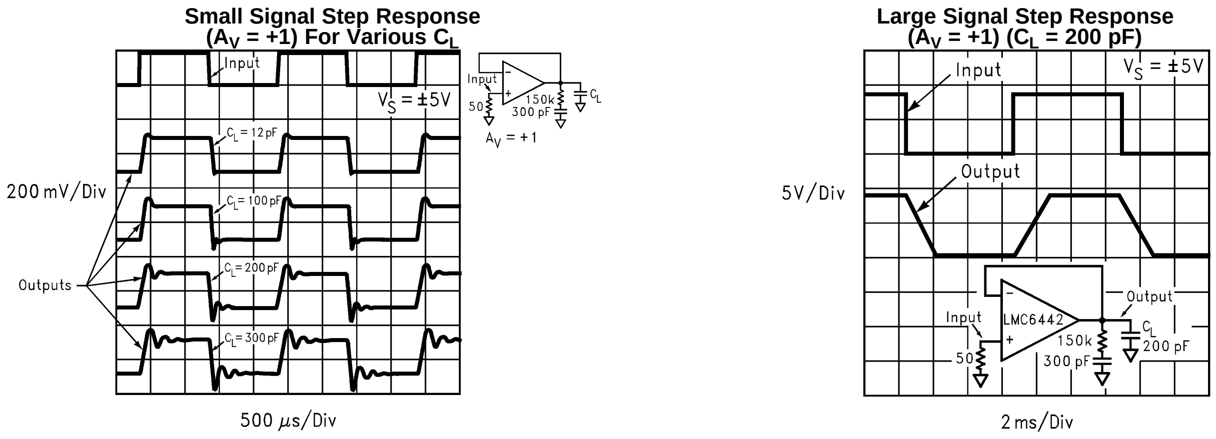


Figure 3.5: Step response of the op-amp in voltage follower mode

are not the same. Both are explained by the difference in Bluetooth activity of the various modes of operation. When a Bluetooth connection is established and data transfer is active, it is obvious that more current is drawn then when the Bluetooth module is completely disabled. The difference between data transfer and no data transfer is probably due to the possibility that, when no data is transferred, the Bluetooth module regularly checks if the connection is still live. The shape of the "Bluetooth searching for connection" waveform is explained by the fact that the RN42XVP module uses a "sniffing" protocol to check for potential connections [23].

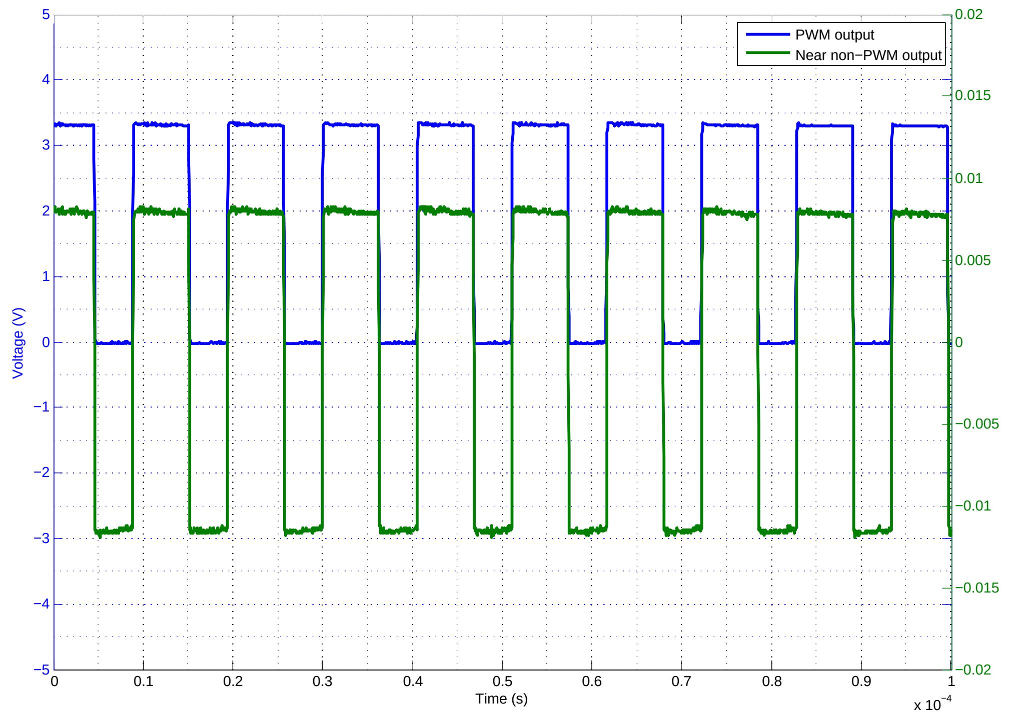


Figure 3.6: Measured PWM and crosstalk signals

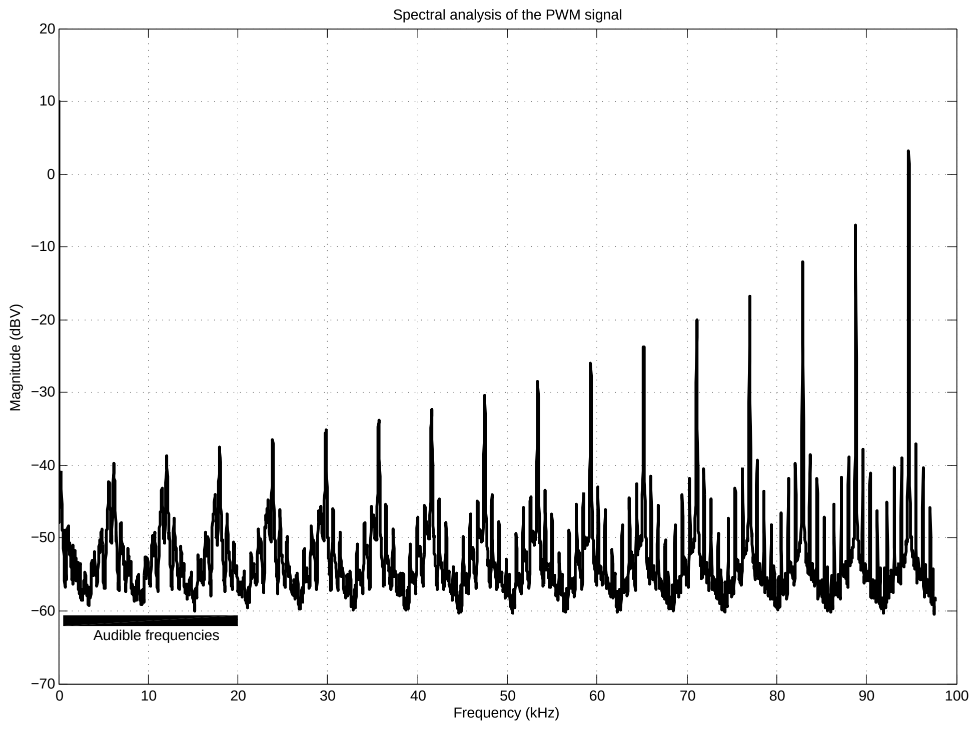


Figure 3.7: Spectrum of PWM signal

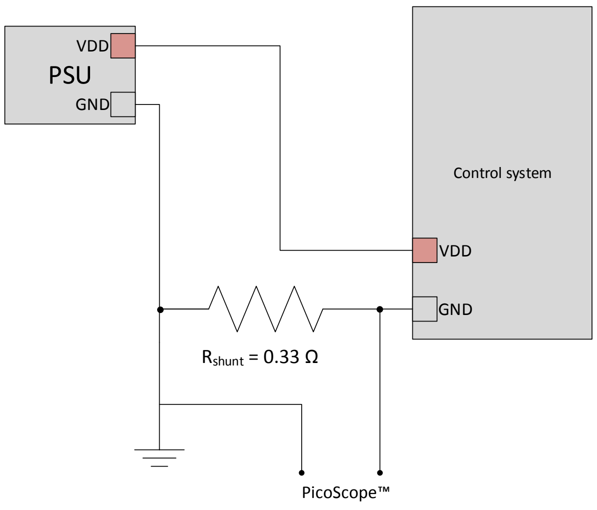


Figure 3.8: The test setup of the power measurement

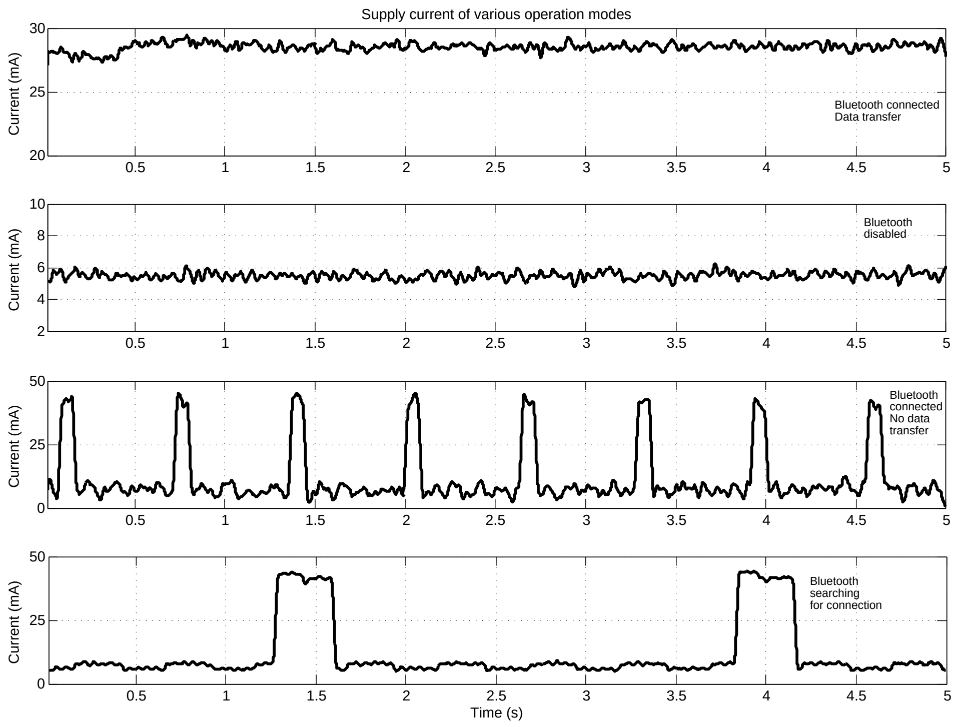


Figure 3.9: The measured current consumption for various modes of operation. From top to bottom: 1.Bluetooth connected and data transfer active, 2. Bluetooth disabled, 3.Bluetooth connected with no data transfer active, 4.Bluetooth searching for connection.

Chapter 4

Future improvements

The prototype that has been built will probably not be suitable as a commercial product. To make it more suitable, several improvements can be made concerning cost price, size, stability and performance.

4.1 Remove components

First of all, a lot of components can be removed, since their purpose is debugging or protection. For example, the polarity protection diode in the power block can be removed, as it may be assumed that in a final product, the polarity of the supply voltage will not be reversed. The components that can be removed without decreasing functionality are listed in tabel 4.1.

Identifier	Function	Needed in final product
C1	Power supply, decoupling	[1]
C4	Programming	✗
C5,C6	Power supply, decoupling	[1]
C7	Design error, V_{USB}	✗
C8	Power supply, decoupling	[1]
D1	Power supply, polarity protection	✗
D2	Programming	✗
LED1	Power supply, indicator	✗
LED2,LED3,LED4	Status indicator	✗
P1	Power measurement, calibrate	✗[2]
R1	Power supply, feed LED1	✗
R4,R5,R6	Status, feed status LEDs	✗
R8	Bluetooth, current limit	✓
R9	Bluetooth, pull down/current limit	[3]
R16	Programming	✗
R23	Programming	✗
SW1,SW2	Bluetooth, reset	[3]
SW3	MCU, reset	[3]

Table 4.1: [1]Only if decoupling is desired
[2]Can be used for calibration voltage measurement
[3]Only if manual resets are desired

Removing these components allows for a smaller PCB or even an IC. The decoupling capacitors serve as high frequency noise filters for V_{DD}. These should not be removed.

The 160 Ω IO resistors limit the current output of the MCU IO port, which may not exceed 25 mA [3]. The maximum output current is limited to $\frac{3.3V}{160\Omega} = 20.6\text{mA}$.

Name	Processor	Communication protocols	ADC resolution	PWM	TX/RX current	Price 1000+ (€)
AMS001	ARM Cortex-M3	UART,SPI,I2C	13	✓	10.8/12.8	6.25
nRF51422	ARM Cortex-M0	UART,SPI,I2C	10	✗	8/9.7	2.81
nRF51822	ARM Cortex-M0	UART,SPI,I2C	10	✗	8/9.7	2.67
RFD22301	ARM Cortex-M0	UART,SPI,I2C	10	✗	12/12	13.09
TiWi-uB1[1]	Intel MCS-51	SPI,I2C	12	✗	12.6/15.7	7.21
CC2541	Intel MCS-51	UART,SPI,I2C	12	✓	18.2/17.9	4.53

Table 4.2: A comparison between combined CPU/BT ICs
[1]The TiWi-uB1 contains a CC2541 chip

Since the INA169 current sensor already contains a buffer, the I_{sense} voltage follower isn't needed anymore.

The trimmer in the V_{sense} voltage divider isn't needed, as the exact maximum generator voltage will be known when further testing is done.

If a more accurate temperature measurement is needed, different thermistors can be used, or a different configuration. A bridge can be used to measure more accurately, or a current source can be used to decrease the amount of noise and current. The current source has a variable current output, when put in series with a small thermistor, the resistor noise will be relatively small.

4.2 Different components

The MCU was mainly chosen because it is familiar, the Bluetooth module because it is easily assembled to the PCB. There are cheaper and more efficient Bluetooth modules. For example, the nRF8001 and nRF8002 are much cheaper than the RN42XVP and use less power. They also have more supported communication protocols.

A different MCU can also be chosen. The MCU needs an ADC, IO pins and a communication protocol (UART, SPI, I2C, USB). The MCU also needs a PWM module, or an efficient way to create a PWM signal. The PWM signal must be able to drive the MOSFET. This means it must allow enough current to charge the gate capacitance, e.g. the Precision32 can sink 300 mA and source 150 mA[35], this is 6 times higher than the PIC18F2550, which can only source 25 mA[3]. Further measurements have to be done to find out what the losses in the MOSFET are, and if higher source current is the solution to lowering these losses. There are many microcontrollers that meet the requirements. Choosing a cheap and small one seems to be the way to go.

The MCU and BT module can also be combined into one IC. There are many Bluetooth modules that feature a CPU. Some are listed in table 4.2.

4.3 Stability GUI

When the Bluetooth module disconnects and the GUI is still expecting a connection, the GUI crashes. Changes to the GUI have to be made to improve the stability.

4.4 Code protection

The PIC18F4550 features code protection. This can be used to prevent other people from reading the firmware on the ROM and preserve the owners property.

4.5 Power consumption

Improvements concerning power consumption can be made. Larger resistor values can be used in e.g. the temperature voltage divider.

The power is proportional to the clock frequency: $P = \alpha \cdot C \cdot V^2 \cdot f$, where α is the activity factor, C the capacity of the MOSFETs, V the supply voltage and f the clock frequency[36]. The activity factor is smaller than or equal to one, because not all MOSFETs switch each clock cycle. Lowering the clock frequency should decrease the power consumption.

To lower the power consumption, both the MCU and BT module could go into sleep/idle mode. If the BT module sleeps, the MCU can wake it up by sending a character via UART. The MCU features both idle and sleep modes, and can be woken from both modes by triggering an interrupt. In idle mode, the CPU is off, but the peripherals are still on. Timer2 will still trigger many interrupts per seconds, causing the MCU to be woken all the time. In sleep mode, both the CPU and peripherals are off. The MCU can only be woken via an external interrupt, two external interrupt pins are connected (IO7 and IO8). The control system would have to wake up when someone is using the DISQ. This can be done by measuring the V_{sense} before the voltage divider. A proposed wake circuit is shown in figure 4.1.

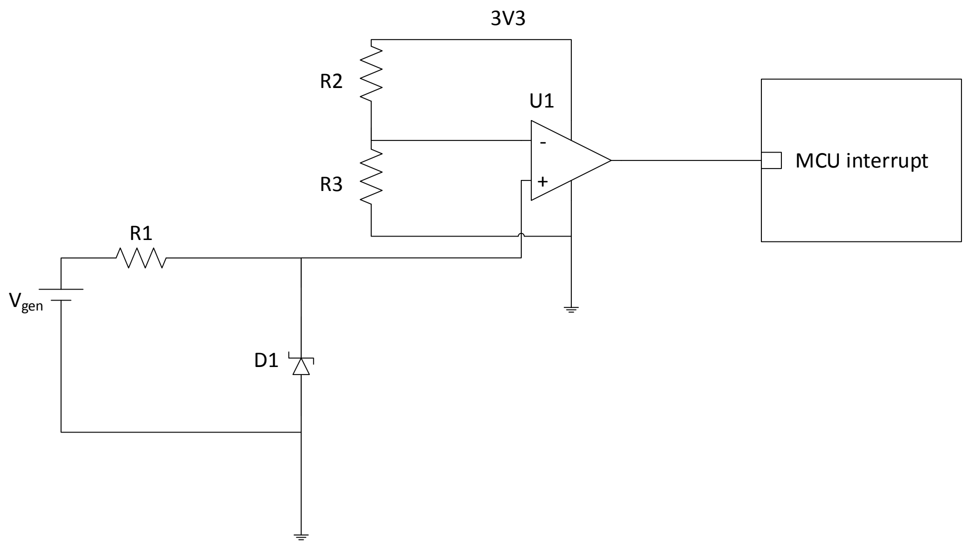


Figure 4.1: Proposed wake circuit

Chapter 5

Conclusion

The problem to be encountered was to design a variable resistance frictionless resistance braking system for an existing fitness device called "The DISQ". In this thesis, every aspect of the design of the control system for this application is discussed. Testing and measurement shows among others that it can be said that the previously set requirements are met:

- Feedback can be provided to the user about the generated electric power
- The temperature of the generator is measured and the control system can respond to this temperature to prevent overheating
- Since the entire control system draws little current (< 30 mA in normal operation), it is likely to be able to operate from a battery power source for a reasonable time

Another important requirement is that the user must be able to control the resistance he or she experiences. This should be done both via a wireless connection and with the buttons on the device itself. Controlling the resistance via a wireless connection works without problems, but due to a design error, the 'RES+' button doesn't work. This can be solved, by using the ADC, in which case the requirement would be fully met.

As always, there are things to improve when the design will ever get anything further than a prototype. This is discussed in the corresponding chapter.

Chapter 6

Ethical considerations

This chapter focuses on the impact the new DISQ can have on the world. Also a number of ethical problems will be analyzed and discussed.

- The new, electrical DISQ will lead to more people sporting, because it's easier than going to the gym now. People can use the DISQ at home. With the app, a network can be build, allowing competitions and appealing to the competitive side of people. The social interaction can prevent people from losing motivation and stopping. More sporting also means their health increases, this could decrease the total costs for healthcare.

- The new DISQ will use a wireless Bluetooth$^{\text{TM}}$ connection to communicate with the user. With many people sporting, this can create a big network. When this network is big enough, it is possible that the amount of networks in the ether can lead to problems. However, this problem is not very realistic since a huge number of Bluetooth$^{\text{TM}}$ are needed in the same area to cause, probably only local, problems.

- For the same reason as the point mentioned above, it is possible that the new DISQ is targeted by hackers when it gains in popularity. Hackers that try to send commands to other peoples DISQ. This can lead to dangerous situations. Consider for example the following scenario: Person A is sporting in the park. He has his DISQ set to a very low resistance, because that is what he finds comfortable at that moment. He runs trough the park with the DISQ and while he is doing that, suddenly person B managed to increase the resistance to the maximum. Due to the almost instantaneous torque increment person A experienced, he trips and is injured.

- The designed control system is designed with RoHS components[37]. This has to be done in order to bring a final product to the market. Problems can arise when the RoHS regulations are not met.

- The Energy Management System (EMS) uses a lithium-ion battery (li-ion). Lithium is extracted in mines all over the world. These mines may cause damage to local flora and fauna. When considering Kantian ethics, we should question whether trading flora and fauna for lithium mines can be a universal law.

- The DISQ can be assembled in different locations. The cost of assembly in low-wage countries is much lower than in western countries. However, the working conditions are often poor. Utilitarianism considers only the consequences of an action. The consequences of choosing a low-wage country to assemble in are: possible poor working conditions for workers, and a lower priced DISQ for consumers. A simple approach is counting the people: there are probably more consumers than workers, so the poor working conditions are outweighed. However, the working conditions do not have to be poor, the client can monitor the assembly process. This would mean that production in low-wage countries is justified.

- Performance data could be sold to (health) insurance companies in order to provide more insight in the health of citizens to those insurance companies. An ethical dilemma is present here between a possibly great increase in profit for the DISQ company and the personal privacy of citizens. Considering virtue ethics, selling information affects privacy. Virtue ethics not only considers virtues but also values and privacy is an important value nowadays.

Bibliography

[1] M. O. Gulbahce, D. A. Kocabas, and I. Habir, “Finite elements analysis of a small power eddy current brake,” in *MECHATRONIKA, 2012 15th International Symposium.* IEEE, 2012.

[2] *Fluke 170 Series True-rms Digital Multimeter Extended Specifications*, Fluke, Mar. 2001.

[3] *PIC18F2455/2550/4455/4550 Data Sheet*, Microchip, Feb. 2006, preliminary.

[4] *Schottky Barrier Diode RB161M-20*, ROHM Semiconductor, Apr. 2011, rev. B.

[5] P. Horowitz and W. Hill, *The Art of Electronics*, 3rd ed. Cambridge University Press, 2015, p. 856.

[6] *SMD CHIP LED LAMP Super Bright Green*, Kingbright, Mar. 2012.

[7] *Schottky Diode SOD-123*, Multicomp, Mar. 2013.

[8] *BAT54T1G, SBAT54T1G Schottky Barrier Diodes*, ON Semiconductor, Apr. 2014, rev. 10.

[9] *PTC Thermistors Engineering Notes*, Digi-key, Jun. 2014.

[10] *NTC Thermistors Engineering Notes*, Spectrum Inc., Jun. 2014.

[11] *NTC Thermistors, Radial Leaded, Standerd Precision*, Vishay, Dec. 2013.

[12] *LMC6442 Duel Micropower Rail-to-Rail Output Single Supply Operational Amplifier*, Texas Instruments, Mar. 2015.

[13] *MMSZ5226B, Zener*, Fairchild, Sep. 2014.

[14] *INA169, High-Side Measurement CURRENT SHUNT MONITOR*, Texas Instruments, Jan. 2015.

[15] *STD30NF06L*, STMicroelektronics, Apr. 2010.

[16] “Bluetooth vs. wi-fi,” http://www.diffen.com/difference/Bluetooth_vs_Wifi, accessed: 15-6-2015.

[17] “Difference between bluetooth and wifi,” http://www.engineersgarage.com/contribution/difference-between-bluetooth-and-wifi, accessed: 15-6-2015.

[18] “Serial peripheral interface (spi),” https://learn.sparkfun.com/tutorials/serial-peripheral-interface-spi, accessed: 10-6-2015.

[19] “I2c,” https://learn.sparkfun.com/tutorials/i2c, accessed: 10-6-2015.

[20] “Usb in a nutshell,” http://www.beyondlogic.org/usbnutshell/usb3.shtml#USBProtocols, accessed: 10-6-2015.

[21] *MPLAB C18 C COMPILER LIBRARIES*, Microchip, Nov. 2004.

[22] *MMSZ5226B, Zener*, Fairchild, Sep. 2014.

[23] *RN41XV & RN42XV Bluetooth Module*, Roving Networks, Mar. 2012, version 1.0.

[24] *PICkit^{TM} 2 Programmer/Debugger User's Guide*, Microchip, Feb. 2008.

[25] "Via stitching," http://techdocs.altium.com/display/ADOH/Via+Stitching, accessed: 17-6-2015.

[26] "Via shielding," http://techdocs.altium.com/display/ADOH/Via+Shielding, accessed: 17-6-2015.

[27] *RN42/RN42N Class 2 Bluetooth Module*, Roving Networks, Nov. 2013, version 2.32r.

[28] P. Horowitz and W. Hill, *The Art of Electronics*, 3rd ed. Cambridge University Press, 2015, p. 955,956.

[29] "Pid theory explained," http://www.ni.com/white-paper/3782/en/, accessed: 19-6-2015.

[30] "Ascii - binary character table," http://sticksandstones.kstrom.com/appen.html, accessed: 8-6-2015.

[31] *PicoScope® 2200A Series, PC Oscilloscopes with arbitrary waveform generator*, Pico Technology, Mar. 2014.

[32] "Basetech mini 1 infrarood-thermometer optiek (thermometer) 1:1 -33 tot 220 c," https://www.conrad.nl/nl/basetech-mini-1-infrarood-thermometer-optiek-thermometer-11-33-tot-220-c-122301.html?WT.mc_id=gshop&gclid=CIW8jqixmcYCFeLItAodCxIABw&WT.srch=1, accessed: 18-6-2015.

[33] "Lm324 glitching in a basic circuit," http://forum.allaboutcircuits.com/threads/lm324-glitching-in-a-basic-circuit.76083/, accessed: 20-6-2015.

[34] P. Horowitz and W. Hill, *The Art of Electronics*, 3rd ed. Cambridge University Press, 2015, p. 309-311.

[35] *The Precision32 Family of Mixed-Signal MCUs*, The Linley Group, Feb. 2012.

[36] N. H. Weste and K. Eshraghian, "Principles of cmos vlsi design: a systems perspective," *NASA STI/Recon Technical Report A*, vol. 85, p. 47028, 1985.

[37] "Rohs regulations government guidance notes," https://www.gov.uk/government/uploads/system/uploads/attachment_data/file/31803/11-526-rohs-regulations-government-guidance-notes.pdf, accessed: 10-6-2015.