NTNU
Kunnskap for en bedre verden

Department of Engineering Cybernetics

TTK4550 - Specialisation project

From hardware to control algorithms: Retrofitting a legacy robot using open source solutions

Author:
Kristian Blom

18.12.23

Table of Contents

List of Figures vii

List of Tables ix

1 List of abbreviations 1

2 Introduction 2

2.1 Purpose . . . 2

2.2 Background and motivation . . . 2

2.3 Scope . . . 3

3 System description 4

3.1 External control system . . . 4

3.2 ORCA Arm description . . . 4

3.2.1 Mechanical layout . . . 6

3.2.2 Electrical layout . . . 6

3.2.3 Functional description . . . 7

3.2.4 Breakdown of a control unit . . . 7

3.3 Remaining parts . . . 8

4 System analysis 9

4.1 Functional requirements . . . 9

4.2 Mechanical requirements . . . 9

4.2.1 RM1: PCB form factors . . . 9

4.2.2 RM1.1: PCB outlines . . . 9

4.2.3 RM1.2: PCB mount points . . . 10

4.2.4 RM2: Wiring . . . 10

4.2.5 RM3: Encoder connectors . . . 10

4.3 Electrical requirements . . . 10

4.3.1 RE1: Power supply . . . 10

4.4 Control requirements . . . 11

4.4.1 RC1: Centralization . . . 11

4.4.2 RC2: Measurement . . . 11

4.4.3 RC3: Communication . . . 11

4.4.4 RC3.1: Internal communication . . . 11

4.4.5 RC3.2: External communication . . . 11
4.5 Administrative requirements . . . 12
4.5.1 RA1: Price . . . 12
4.5.2 RA2: Complexity . . . 12

5 Theory **13**
5.1 The V-model . . . 13
5.2 Pulse Width Modulation (PWM) and brushed DC motors . . . 13
5.3 Voltage regulation . . . 14
5.4 Motor driver current sense . . . 14

6 System architecture **15**
6.1 Summary of constraints . . . 15
6.2 Design concepts . . . 15
6.2.1 Concept 1: One primary MCU . . . 15
6.2.2 Concept 2: Distributed control . . . 16
6.2.3 Concept 3: Free wiring . . . 16
6.3 Final design . . . 20
6.3.1 Overview . . . 20
6.3.2 Internal communication: CAN bus and I2C . . . 20
6.3.3 External communication: Universal Serial Bus (USB) . . . 20
6.3.4 Control units . . . 20

7 Component selection **22**
7.1 Integrated Circuits . . . 22
7.1.1 Microcontroller unit: STM32F303RE . . . 22
7.1.2 Motor driver: DRV8251A . . . 23
7.1.3 Intertial measurement unit: LSM6DSMTR . . . 23
7.1.4 CAN transceiver: TJA1057BT . . . 23
7.1.5 Voltage regulator: LM317HV TO220 . . . 23
7.1.6 TVS array: CDSC706-0504C . . . 24
7.1.7 Optocoupler: OPB971N51 . . . 24
7.1.8 Motor relay: JV3SKT . . . 24
7.2 Connectors . . . 24
7.2.1 VBUS connectors: 3.5 mm 4P Phoenix screw terminal . . . 24
7.2.2 20 pin connector socket: TE 1761681-7 . . . 24

7.2.3 16 pin connector socket: TE 2-1761603-6 . . . 24
7.2.4 Motor connector socket: Molex 70543-0001 . . . 25
7.2.5 Encoder connector pin header: 2x5 2.54 mm generic . . . 25
7.3 Generic circuitry components . . . 25
7.3.1 Bulk capacitors . . . 25
7.3.2 Oscillator: ABL-1774766 . . . 25

8 Circuit design **26**
8.1 Tools . . . 26
8.2 Assemblies . . . 26
8.2.1 Low voltage assembly . . . 26
8.2.2 Motor driver assembly . . . 28
8.2.3 CAN assembly . . . 29
8.2.4 IMU assembly . . . 29
8.2.5 USB assembly . . . 31
8.2.6 MCU pinout design . . . 32
8.2.7 MCU power supply assembly . . . 34
8.2.8 External oscillator assembly . . . 34
8.2.9 Reset and bootmode assemblies . . . 34
8.2.10 Flash header pin assembly . . . 35
8.2.11 Encoder connector assembly . . . 36
8.3 Torso . . . 36
8.3.1 1A: Power and CAN connector . . . 36
8.3.2 3A: Low voltage jumpers and debug . . . 36
8.3.3 1B: 20 pin ribbon cable connector . . . 36
8.3.4 4B: Pins pulled low . . . 36
8.3.5 1C: UART . . . 37
8.3.6 5C: Mounting holes . . . 37
8.3.7 3D: Motor assembly debug header . . . 37
8.4 Shoulder . . . 38
8.4.1 1A: Power and CAN connectors . . . 38
8.4.2 3A: Low voltage debug and jumpers . . . 38
8.4.3 4A: USB assembly . . . 38
8.4.4 5A: IMU interrupt jumper . . . 39
8.4.5 5A: IMU connector . . . 39

8.4.6 4B: Pins pulled low . . . 39
8.4.7 1C: UART . . . 39
8.4.8 5C: Mounting holes . . . 39
8.4.9 3D: Motor assembly debug header . . . 40
8.5 Hand . . . 40
8.5.1 1A: Power, CAN and IMU connector . . . 40
8.5.2 3A: Low voltage jumpers . . . 40
8.5.3 3A: Optocoupler and A/B connectors . . . 40
8.5.4 4A: UART and flash connector . . . 40
8.5.5 5A: IMU interrupt and wrist optocoupler jumper . . . 41
8.5.6 1B: Encoder connectors . . . 41
8.5.7 4B: Pins pulled low . . . 41
8.5.8 2C: Reset . . . 41
8.5.9 5C: Mounting holes . . . 41
8.5.10 1D: Motor assembly . . . 41
8.6 IMU boards . . . 43
8.7 Auxiliary boards . . . 44
8.7.1 Bogie board . . . 44
8.7.2 Rail board . . . 44

9 PCB design **46**
9.1 Tools . . . 46
9.1.1 Schematic and layout: KiCad . . . 46
9.2 Design process and rules . . . 46
9.2.1 Layering . . . 46
9.2.2 Via stitching . . . 46
9.2.3 Constraints . . . 47
9.2.4 THT vs SMD, footprint sizes . . . 47
9.3 About the presentation of PCB layouts . . . 47
9.4 About mount point presentation . . . 47
9.5 Torso . . . 48
9.5.1 Outline and mount points . . . 48
9.5.2 Front copper and silk . . . 49
9.5.3 Back copper and silk . . . 53
9.6 Shoulder . . . 54

9.6.1 Outline and mount points . . . 54
9.6.2 Front copper and silk . . . 55
9.6.3 Back copper and silk . . . 56
9.7 Hand . . . 57
9.7.1 Outline and mount points . . . 57
9.7.2 Front copper and silk . . . 58
9.7.3 Back copper and silk . . . 60
9.8 IMU boards . . . 61
9.8.1 Outline and mount points . . . 61
9.8.2 Front copper and silk . . . 61
9.8.3 Back copper and silk . . . 62
9.9 Auxiliary boards . . . 63
9.9.1 Outline and mount points . . . 63
9.9.2 Front copper and silk . . . 64
9.9.3 Back copper and silk . . . 66

10 PCB production **67**

11 Software design **71**
11.1 Tools . . . 71
11.1.1 CubeMX . . . 71
11.1.2 ST-LINK programmer . . . 71
11.1.3 VSCode . . . 71
11.2 Driver implementation . . . 71
11.3 Test unit implementation . . . 71

12 Testing and validation **72**
12.1 Phase 1: Test design . . . 72
12.2 Phase 2: Circuit design validation . . . 72
12.3 Phase 3: Validation of produced units . . . 73

13 Results **76**
13.1 Test results . . . 76
13.1.1 Design tests . . . 76
13.1.2 Production tests . . . 76
13.2 Qualitative evaluation of produced PCBs, aspects not covered by tests . . . 77
13.3 Mechanical integration . . . 78

13.4 Drivers 81

14 Discussion **82**

14.1 Test results 82

14.2 General 82

15 Conclusion **85**

16 Further work: Improvements for Mk1.1 **86**

16.1 High priority 86

16.1.1 USB assembly 86

16.1.2 IMU assemblies 86

16.1.3 CAN bus assemblies 86

16.1.4 Mount holes 86

16.1.5 20 pin connector 87

16.1.6 Motor driver assembly 87

16.1.7 Optocoupler circuit 87

16.2 Low priority 87

16.2.1 Hand bulk capacitors and interface pins 87

16.2.2 Voltage regulators 87

16.2.3 Encoders 87

16.2.4 TVS array 88

16.2.5 Motor driver relay control circuit 88

16.2.6 Motor characterisation 88

16.2.7 PCB production 88

16.3 Mk1.1 matrix 88

Bibliography **90**

Appendix **92**

A Bill of materials 92

B STM32F303 configuration report 96

C TTK8 Report: Testing and validation 123

List of Figures

1 Arm description . . . 5
2 Old control unit . . . 7
3 Schematic of control unit . . . 8
4 The V-model applied to a software project . . . 13
5 Architectural concept 1 . . . 17
6 Architectural concept 2 . . . 18
7 Architectural concept 3 . . . 19
8 System architecture . . . 21
9 Reference: Low voltage assembly . . . 26
10 Schematic: Low voltage assembly . . . 27
11 Reference: Motor driver assembly . . . 28
12 Schematic: Motor driver assembly . . . 28
13 Reference: CAN assembly . . . 29
14 Reference: IMU assembly . . . 30
15 Schematic: IMU assembly . . . 31
16 Reference: USB assembly . . . 32
17 MCU pinout . . . 32
18 Reference: MCU power supply assembly . . . 34
19 Reference: Oscillator, reset and bootmode . . . 35
20 Reference: Flash header assembly . . . 35
21 Schematic: Torso control unit . . . 38
22 Schematic: Shoulder control unit . . . 39
23 Schematic: Hand motor assembly . . . 42
24 Schematic: Hand control unit . . . 42
25 Schematic: IMU board . . . 43
26 Schematic: Bogie and rail boards . . . 45
27 Layout: Torso measurements . . . 48
28 USB Impedance matching . . . 49
29 Layout: Torso front copper . . . 52
30 Layout: Torso back copper . . . 53
31 Original shoulder unit . . . 54
32 Layout: Shoulder measurements . . . 55
33 Layout: Shoulder front copper . . . 56

34 Layout: Shoulder back copper 56
35 Original hand unit 57
36 Layout: Hand measurements 58
37 Layout: Hand front copper 59
38 Layout: Hand back copper 60
39 Layout: IMU board measurements and front copper 61
40 Layout: IMU board back copper 62
41 Original auxiliary units 63
42 Layout: Auxiliary boards measurements 64
43 Layout: Auxiliaries front copper 65
44 Layout: Auxiliaries back copper 66
45 Production: Reflow oven illustration 67
46 Production: Hand headers 68
47 Comparison: Torso 69
48 Comparison: Shoulder 69
49 Comparison: Hand 69
50 Comparison: IMU and auxiliaries 70
51 TTK8 test units 72
52 Integration: Control units 79
53 Integration: Auxiliary units 80

List of Tables

1 List of abbreviations 1
2 Joints and elements of the arm 4
3 Origina parts kept 8
4 STM32F303RE pinout 33
5 Torso 20 pin connector pinout 37
6 Pin description of the hand programming header 40
7 Bogie 16 pin connector pinout 44
8 TVS diode pinout 45
9 Pinout of the 14 pin connector 45
10 Table of torso hardpoint mount points 48
11 USB impedance matching parameters 49
12 Explanation of motor drivers and relay control debug headers 50
13 Explanation of low voltage debug and CAN/power bus connector 51
14 Explanation of flash header pins 51
15 Table of shoulder hardpoint mount points 54
16 IMU silk description 55
17 Table of hand hardpoint mount points 57
18 Hand flash/UART and IMU interrupt headers 59
19 Table of IMU harpoint mount points 61
20 IMU board pin header silk description 62
21 Table of auxiliary hardpoint mount points 63
22 Auxiliaries front silk description 65
23 A summary of all design tests and their results 73
24 A summary of all production tests and their results (cont. in table 25) 74
25 Cont.: A summary of all production tests and their results 75
26 A summary of further work 89
27 Auxiliaries BOM (Bogie and rail boards) 92
28 IMU BOM (single board) 92
29 Torso BOM 93
30 Shoulder BOM 94
31 Hand BOM 95

1 List of abbreviations

Table 1: List of abbreviations

Abbreviation	**Meaning**
ADC	Analog to digital converter
BOM	Bill of materials
CAN	Controller Area Network
CPR	Counts per cycle (encoder)
DAC	Digital to analog converter
DOF	Degrees of freedom
ESE	Embedded systems engineering
HAL	Hardware abstraction layer
HiZ	High impedance
I2C	Inter-integrated circuit
IC	Integrated circuit
IMU	Inertial measurement unit
JLCPCB	Company name, PCB manufacturer
LED	Light emitting diode
MCU	Microcontroller unit
MP	Mount point
PCB	Printed circuit board
PWM	Pulse width modulation
RA	Requirement, administrative
RC	Requirement, control
RE	Requirement, electrical
RM	Requirement, mechanical
SMD	Surface mounted device
ST	Company name, MCU manufacturer
THT	Through hole technology
UART	Universal Asynchronous Reciever/Transmitter
USB	Universal Serial Bus
VBUS	Voltage bus

2 Introduction

2.1 Purpose

This report describes the specialisation project of Kristian Blom during the autumn semester of 2023.

2.2 Background and motivation

From the initial project description: *This project is defined and commissioned by a student in cooperation with Omega Verksted (OV). OV is a student-operated workshop and makerspace with solid roots in the Department of Engineering Cybernetics through more than 50 years of out-of-class electronics and other engineering projects. For some time, an ORCA 6DOF robotic arm for industrial chemical analysis has resided on OV[...].The arm was manufactured in the mid-90s, and previous attempts to contact it using publicly available resources have been unsuccessful.*

Embedded systems engineering (ESE) is a field on the edge between ready availability for hobbyists and the cutting edge of technology. Like many computing-adjacent engineering fields, it has one foot in the open-source culture propagated by the Internet and its ideals of information freedom, and one in the world of secrecy and NDAs which represents much of the world's industrial interests. ESE can be low cost home automation with a Raspberry Pi and user friendly software developed by individuals in their spare time, or it can be highly optimised industrial solutions developed by teams of specialists. This project tilts, at least in spirit, towards the former. The main motivation behind the project is not to find groundbreaking solutions to robotics control, but to combine elements from the Master programme in a wholesome embedded engineering project.

From breaking down the refurbishment problem in the system analysis to writing handover documentation for the final product, the goal is as much to see how far one person can get with one year of development using all available resources as it is to actually produce a deliverable. The "specialisation project" part of the project which this report concerns, focuses on hardware development – specifically printed circuit board (PCB) design. The goal is to restore the original function of the arm. Few NTNU courses, if any, offer experience with this aspect of embedded development, and so this has to a large extent been a journey of trial, failure, and trusting the advice of Internet fora and component datasheets.

The main reason for this project's focus on using open source solutions is the intention of handing over a maintainable project to Omega Verksted. Closed-source solutions often require expensive licenses, and few students have the skills to use them for this reason. Additionally, closed source often comes with a lessened sense of ownership as the interpretation and use of intellectual property will ultimately be at the mercy of a third party. All of these points are a poor match with the student driven workshop that is Omega Verksted. A dominant aspect of the early phase of the project was therefore to find a set of software solutions that would meet the requirements for PCB design and microcontroller programming. These solutions do exist, and using software like KiCad and ST's CubeMX, a novice engineer on a shoestring budget may develop, produce and program embedded systems quite efficiently.

Lastly, there is a recycling and reuse angle to the project. The arm is old, but mechanically intact. It may have been decomissioned for any number of reasons: the control software may not have been compatible with modern operating systems; the electronics may have worn out over time; the former users may simply have wanted a more modern or versatile arm for their use case. However, the arm being mechanically defunct was likely not the reason for its decommissioning. Reusing hardware without access to spare parts may not be a particularly viable solution in industrial applications, but for some select use cases, such as developing a robotic platform for use in a student workshop, it may be just as good as investing in a newer, proprietary robot. On a more subjective plane, breathing new life in old hardware may also be deeply satisfactory[1].

[1] Additionally, it falls vaguely in line with UN sustainability goal 12[13]. Re: department-wide email sent 21.11.23

2.3 Scope

The main goal of this project has been to design and manufacture replacement electronics for the arm. With a few notable exceptions, all electronic components inside the arm were to be replaced. Additionally, low level hardware drivers and communication protocols were to be written in preparation of the arm's use in more advanced robotics projects.

Scope

1. Brief description of the old system
2. Finding a new microcontroller unit (MCU)
3. Designing the new circuits
4. Producing the new circuits
5. Verifying the new circuits
6. Write low-level drivers
7. Write a simple world interface

3 System description

This section describes the ORCA robotic arm in its original state, and is illustrated by figure 1. As the project aims to restore the arm's functionality, this section also lays the foundation for establishing design constraints.

3.1 External control system

In addition to the arm itself, the orginial system consisted of a power supply box and a non-specific computer with high-level control software and user interface. The power supply box interfaces with the arm by DSUB15 and the computer by HPIB (IEEE-488) or RS232. The exact function of the power supply box, beyond converting wall power to 48V DC and relaying the information on the HPIB bus, was never ascertained. The only requirement of the computer was that it must run MS-DOS or Windows 3[20].

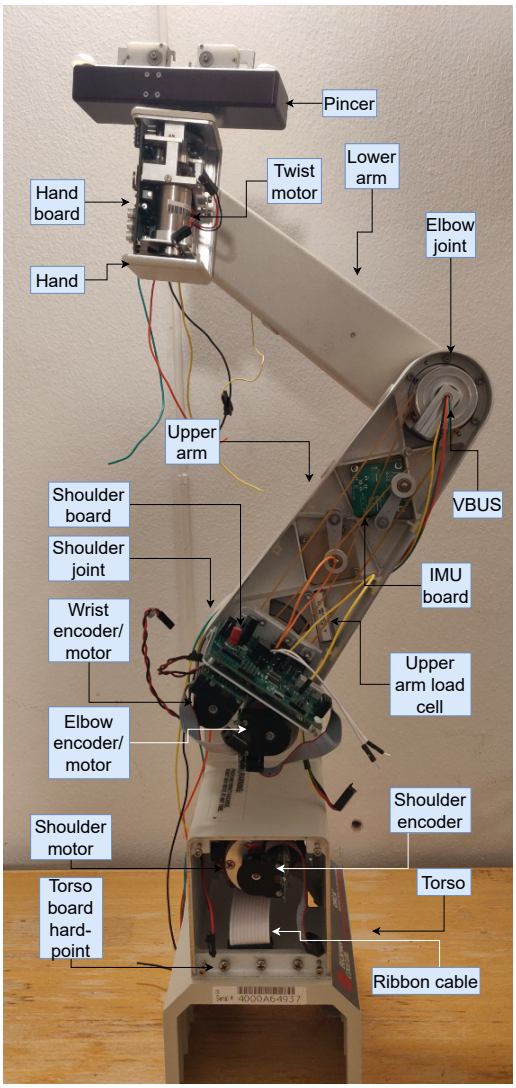
3.2 ORCA Arm description

The ORCA arm is a 5DOF robotic arm mounted on a linear rail by bogie, for a full system of 6DOF. Joints and elements of the arm are summarised in table 2.

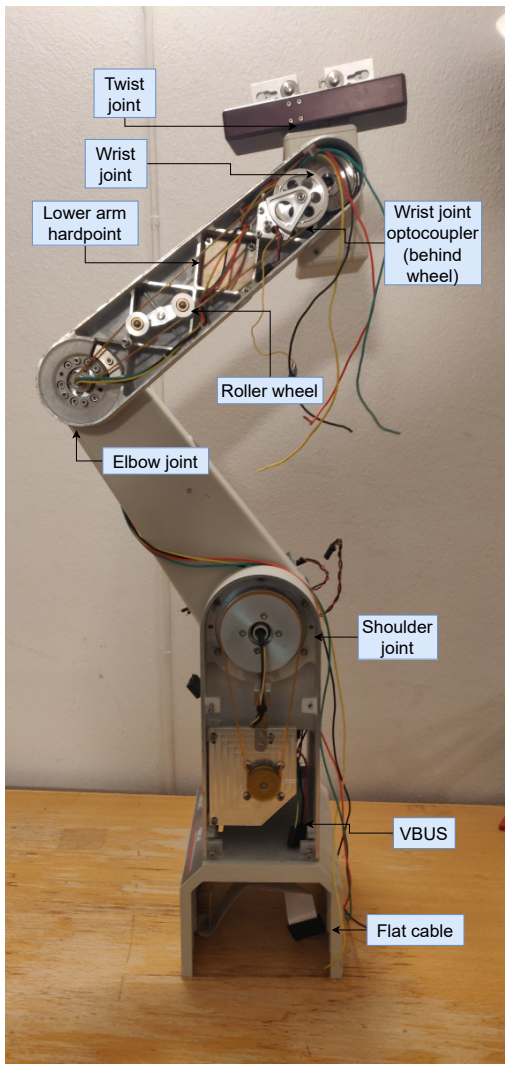
The joints	...are connected by the elements
linear actuator,	linear rail,
shoulder,	torso,
elbow,	upper arm,
wrist,	lower arm,
twist and	hand and
pinch	gripper.

Table 2: Joints and elements of the arm

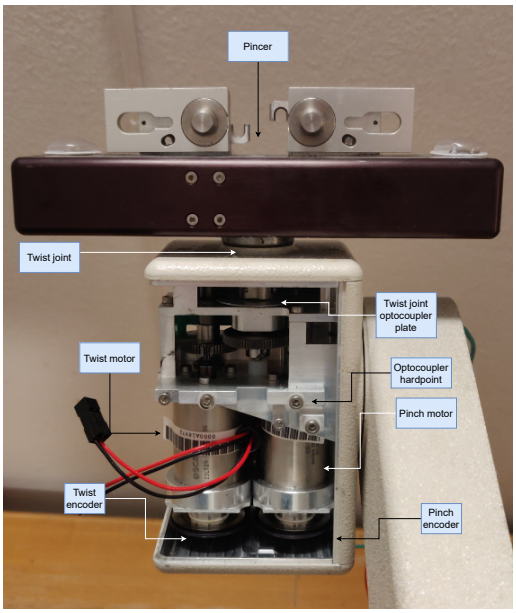
The shoulder, elbow and wrist joints rotate on a plane orthogonal to the ground, and the rail moves the entire arm such that the hand section may reach any point in a half cylinder of a radius of approximately 50cm, and a length equal to that of the linear rail. The twist joint rotates the pincer along an axis parallel to the arm plane, its direction dictated by the wrist joint, while the pincer is a linear joint at most 8cm wide[21].



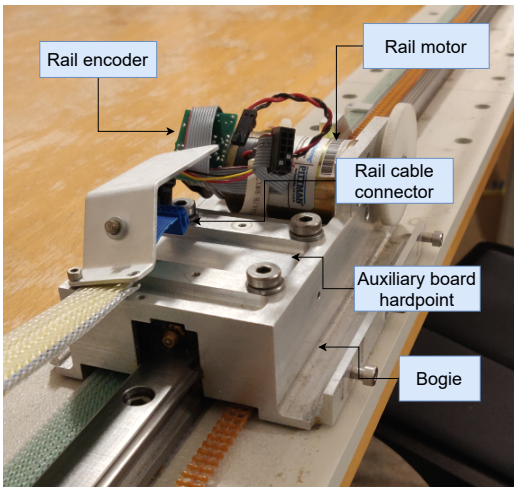
Arm viewed from the front.



Arm from the back.



A more detailed image of the hand and pincer.



The bogie on which the torso is mounted.

Figure 1: The arm with annotations.

3.2.1 Mechanical layout

Joint movement is driven by motors located in the linear rail, torso, shoulder and hand, and all are geared at various ratios. In the case of the shoulder, elbow and wrist joints, there are belts connecting the motors to their respective joints kept taut by spring loaded roller wheels. The same joints are also designed such that electrical wiring may pass through them undisturbed by the arm's movement.

Limitations on movement for each joint:

- Linear rail: Until the arm bogie crashes in either end of the rail, about 1.5 meters.
- Shoulder: Until the upper arm crashes in either side of the torso, about 270 degrees.
- Elbow: Until the hand crashes in the upper arm, about 300 degrees.
- Wrist: Locks mechanically after 3 rotations.
- Twist: Unlimited.
- Pinch: until the pincer claws reach either the middle or outer ends of the rail.

Printed circuit boards (PCBs) may be installed in 9 different locations on the arm, referred to as hardpoints. Each hardpoint has a number of bolt holes, threaded rods etc., referred to as mount points.

3.2.2 Electrical layout

A DSUB15 connector mounted in one end of the arm's linear rail serves at the system's electrical entry point. Immediately inside the arm, an array of TVS diodes protect the circuit. 14 of the connector's 15 leads continue to the arm bogie and into the torso.

In the lower half of the torso, mounted on the bogie, is the linear actuator and an end switch. The information passed from the DSUB is combined with the actuator and end switch interfaces in a 20 pin ribbon cable.

The upper half of the torso contains one of three control units, likely responsible for driving the linear and shoulder actuators, as well as the torso actuator. The unit receives the 20 pin ribbon cable from below and passes on a four wire interface to the shoulder.

The shoulder contains the second of three control units, likely responsible for driving the elbow and wrist actuators, as well as the elbow and wrist actuators. It receives a four wire interface and passes it on towards the hand.

The upper arm has one load cell and an inertial measurement unit (IMU), data processed by a microprocessor interfaced with the four wire bus.

The lower arm has one load cell and an IMU, data processed by a microprocessor interfaced with the four wire bus. Additionally, one sensor of undetermined type, likely an optocoupler, triggers on a number of specific locations along the wrist rotation. The sensor is connected to the microprocessor.

The hand contains the third of three control units, likely responsible for driving the twist and pinch actuators, as well as an optocoupler mounted such that it can establish the twist joint's absolute position.

3.2.3 Functional description

Based on the components identified, the system is presumed to have implemented the following functions:

- Load measurement, i.e. weight carried by the arm;
- Motor power draw, inferred from current sense from the motor drivers;
- Angular rate or acceleration measurement in the arm sections;
- Joint position, inferred from encoder, optocoupler and IMU measurements;
- Joint movement;
- Internal/external communication; and
- Power supply for all units.

3.2.4 Breakdown of a control unit

Figure 3 shows a simplified schematic of the original control unit mounted in the torso, traces mapped using a multimeter in continuity mode. Figure 2 shows a picture of the same unit. While identifying the components was sufficient to guess most of the arm's original functionality, mapping the traces was gave valuable confirmation. We see that the Motorola MC68HC11 microprocessor interfaces with two Agilent HCTL-2000 decoders, two Texas Instruments LMD18200 H-bridges/motor drivers, one Texas Instruments CD74HC multivibrator, an Intersil CP82C54 interval timer and a Texas Instruments SN75176AP differential bus tranciever. Additionally, there is an ST L4960 switching regulator, the 20-pin ribbon cable connector and one 10-pin encoder connector.

In the absence of a detailed data sheet or user manual for the arm, knowing the specifications of each part, and particularly the motor drivers, is essential to setting design parameters for the refurbished system.

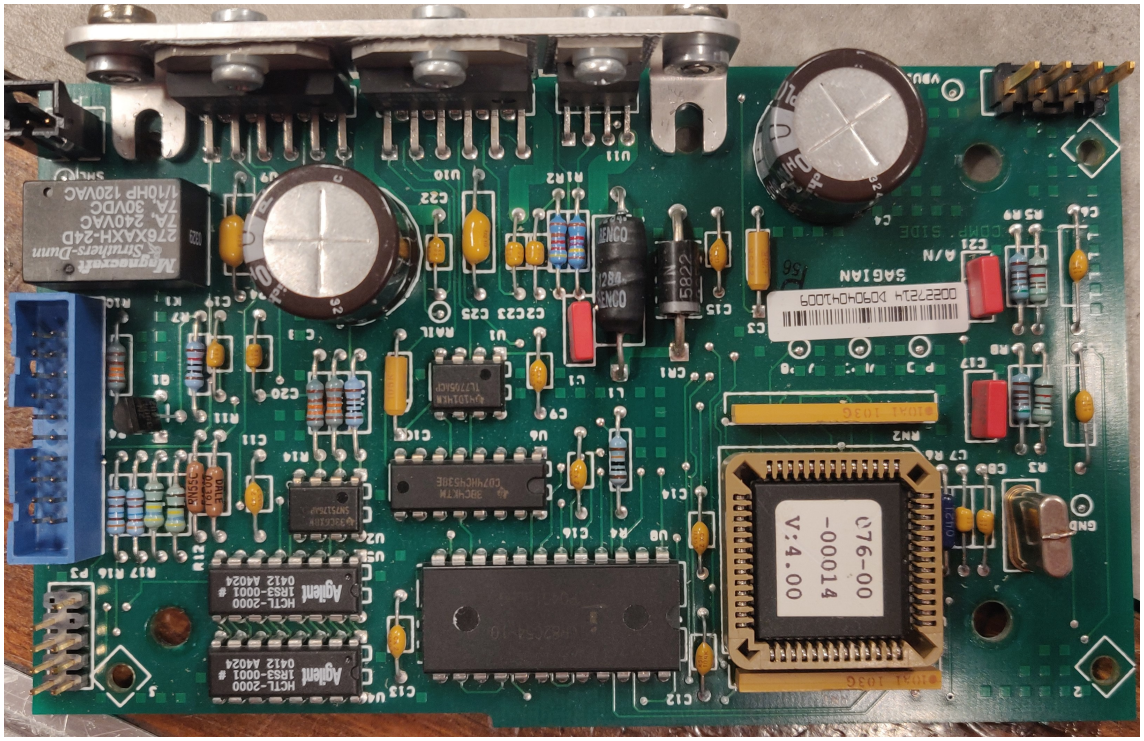


Figure 2: Old control unit installed in the torso

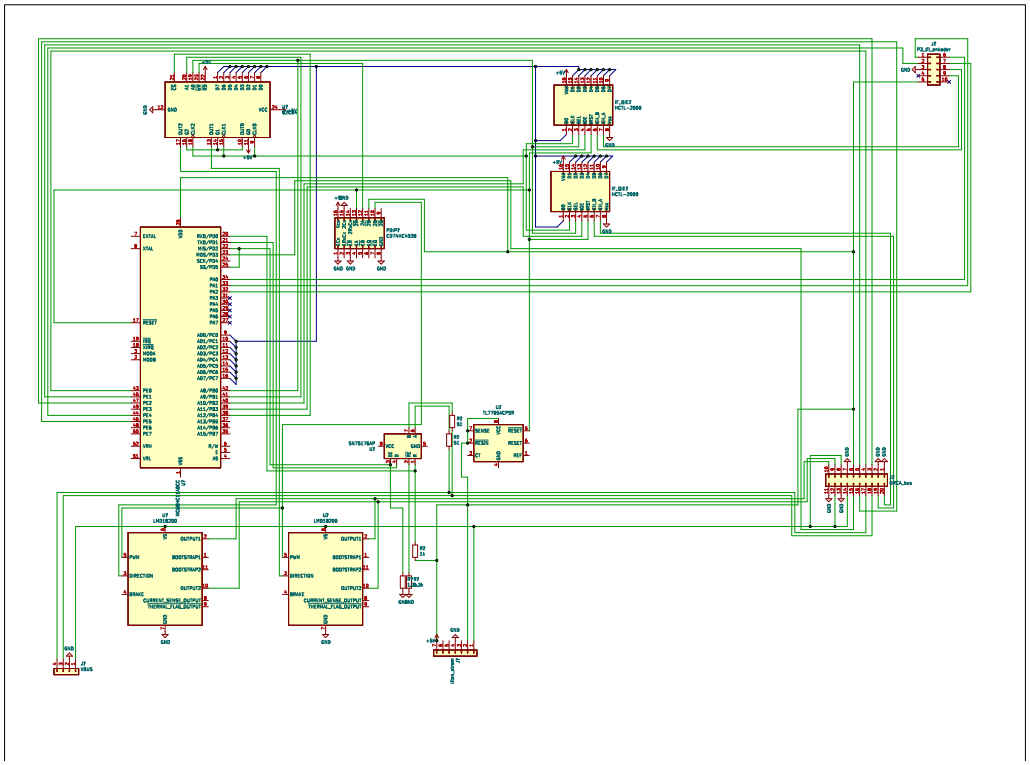


Figure 3: A simplified schematic of the old torso control unit

3.3 Remaining parts

The motors, encoders, optical and end switch sensors were kept throughout the refurbishment project. Replacing these parts would have been expensive and/or mechanically invasive. Table 3 summarises them.

Table 3: Original parts kept in the project

Part name	**Description**	**Part no.**
Rail motor	Pittman 40mm 30.3V BDC motor	9233C59-R1
Shoulder motor	Pittman 54mm 30.3VDC BDC motor	14205C389-R1
Elbow motor	Pittman 40mm 24V BDC motor	9234C59-R1
Wrist motor	Pittman 30mm 30.3V BDC motor	8224C143-R2
Twist motor	Escap 23mm 23LT2R, 12V BDC motor	23LT2R
Pinch motor	Escap 23mm 23LT2R, 12V BDC motor	23LT2R
Twist rotational sensor	TT Electronics Photologic Slotted Optical Switch	OPB960_990-3241431
Wrist rotational sensor	N/A	N/A
End switch	N/A	N/A
Motor encoder	Optical quadrature encoder 500CPR	HEDS-9100

4 System analysis

This chapter describes design constraints and requirements of the new system. They are based on the old hardware. Generally, the keywords **must** and **shall** denote requirements which the failure to heed would significantly compromise the system's ability to fulfill its intended function of restoring the arm's previous capabilities. Similarly, the keyword **should** denotes requirements which the failure to heed may only reduce the system's ability to the same.

4.1 Functional requirements

Aiming to restore or improve upon the original capabilities of the arm, the system should implement the following functions.

- **F1: Operate the 6DOF arm**
- **F1.1: Control the position of 6 joints concurrently**
- **F2: Ensure limits of safe operations are not exceeded**
- **F2.1: Measure operating voltages and currents at key positions**
- **F2.2: Measure torque of 6 motors concurrently**
- **F3: Supply sufficient power to support F1 and F2**
- **F4: Implement a sufficient communication infrastructure to support F1 and F2**

4.2 Mechanical requirements

4.2.1 RM1: PCB form factors

The arm has 9 hardpoints for PCBs in various sizes. On the assumption that modern technology is generally smaller and more versatile than that of the mid 90s, one goal of this project is to reduce the number and variety of PCB designs, centralizing functionality and therey making future development and maintainance simpler. Nevertheless, the design must conform to the form factors enabled by the existing hardpoints and their respective mount points.

4.2.2 RM1.1: PCB outlines

Measurements of existing boards, height measured as the tallest component (WxLxH). This requirement may be understood as "must not exceed".

Linear board: 56x77x15mm

Torso board: 140x84x23mm

Shoulder board: 89x89x23mm

Arm boards (x3): 40x40x10mm

Hand boards (x2): 86x56x8mm

Having the most space, using the hardpoint for the torso or shoulder board seems like the most reasonable basis for a centralized design.

4.2.3 RM1.2: PCB mount points

Mount points must be respected by one of two modes:

Ignored: PCB outline is drawn such that it does not overlap with the mount point at all.

Utilized: PCB is designed with the mount point in mind, such that the center of the PCB's hole aligns with the center of the mount point, and the diameter is at least large enough to accommodate the mount point's intended bolt size.

4.2.4 RM2: Wiring

The arm has limited space for internal wiring. The four wire bus running the length of the arm is threaded through a hole in the elbow and wrist joints with a diameter of 4mm. Altering the mechanical components of the arm is outside the scope of this project, and the motor controller design must account for this.

4.2.5 RM3: Encoder connectors

Of the 6 HEDS encoders in the arm, 4 are soldered to assembly PCBs with 2x5 pin connectors wired to their respective controller boards. The remaining 2 (hand) are plugged directly into their controller boards, their 5 interface pins exposed when the boards are removed. It is probably not feasible to remove the assembly PCBs without destroying the HEDS' pins, so the new design must account for 2 different mechanical interfaces with the encoders.

4.3 Electrical requirements

4.3.1 RE1: Power supply

Complete data sheets for the motors could not be found, but cable dimensions indicate approximately 70W for the four upper motors and control units. Further testing of power consumption from the lower motors, linear rail and shoulder, is needed, but the original power supply was rated for 350VA[21].

- **RE1.1:** The motor drivers should be rated for 3A continuous supply at 55V[2].
- **RE1.2:** The power supply of a given unit must be sufficient to supply the unit at peak power draw.
- **RE1.3:** In the interest of standardisation, all units should be compatible with a 3.3V power supply.

[2]From the datasheet of the original LMD18200 H-bridges

4.4 Control requirements

4.4.1 RC1: Centralization

Control should be centralized as much as possible within the constraints of RM2, minimizing the amount of microprocessors and other integrated circuits in the system and thereby its complexity.

4.4.2 RC2: Measurement

The system must measure

- **RC2.1:** 6x HEDS encoder pulse train signals
- **RC2.2:** 6x motor voltage outputs
- **RC2.3:** Grip strength
- **RC2.4:** 6x motor currents

The system should measure

- **RC2.5:** 6x motor controller temperature
- **RC2.6:** 2x 2-axis IMU output signals
- **RC2.7:** 6x motor torque (inferred from current)
- **RC2.8:** 3x switch signals

RC2.1-3 are necessary to fulfill F1.1.

RC2.4 is necessary to fulfill the safety functions F2.1.

RC2.5 may provide useful data in the implementation of F2.

RC2.6 aids pose estimation.

RC2.7 may provide useful data in the implementation of F2.

RC2.8 may be usefor for pose estimation

4.4.3 RC3: Communication

The original MCU was capable of supporting bus speeds of up to 5MHz[17]. On the assumption that this was necessary and sufficient for the orignial system to function, the refurbished system should not implement communication interfaces with a bit rate lower than 5Mb/s.

4.4.4 RC3.1: Internal communication

From RM2, the system will likely need to be distributed among multiple ($\geq$2) nodes communicating via a serial interface. The system must implement such an interface.

4.4.5 RC3.2: External communication

The system must be able to communicate with the outside world, i.e. receive movement orders from a high-level ($\geq$HAL) control system. In keeping with the project's goal of user friendliness, the chosen protocol should not require uncommon hardware.

4.5 Administrative requirements

4.5.1 RA1: Price

Components should generally be as cheap as possible while still fulfilling their purpose, as the project's budget must be assumed to be 0.

4.5.2 RA2: Complexity

In order to reduce the project's overall complexity, parts should as far as reasonably practiable be chosen such that the final bill of materials (BOM) has as few unique part numbers as possible. Fewer unique parts imply fewer unique interfaces between parts, which would reduce the chances of making mistakes when designing and/or using said interfaces.

5 Theory

In decreasing order of relevance, this project builds on the curricula of *TTK4155 – Embedded and industrial computer systems design*, *TFE4101 – Electrical circuits and digital design*, *TTK4147 – Real-time systems* and *TDT4160 – Computers and digital design*. This section discusses some concepts important to the project, but much has been left out. This is further discussed in section 14.

5.1 The V-model

The design process of this project is based on the V-model for systems engineering[18]. The V-model breaks a project down in tiers of decreasing abstraction levels, with requirement specification at the top and module implementation at the bottom. A key point of the V-model is the feeding back of preliminary results from the validation of one module/tier into its design before proceeding to the next, repeatedly evaluating the project on all abstraction levels. In this report, section 4 roughly represents the top level, while the test units presented in section 12 represents the bottom level. Figure 4 illustrates the V-model as applied to a software project, but is largely applicable to a hardware project as well.

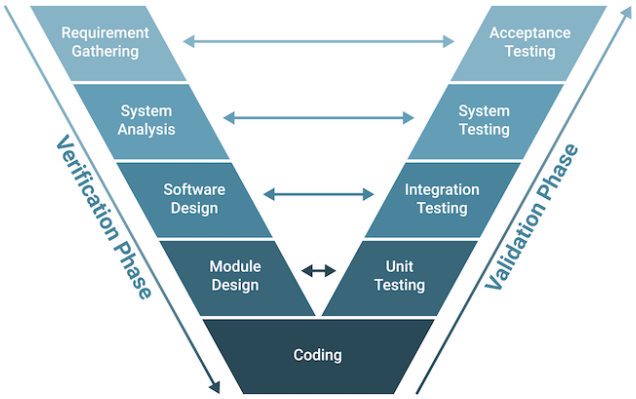


Figure 4: The V-model applied to a software project

5.2 Pulse Width Modulation (PWM) and brushed DC motors

The robotic arm uses 6 brushed DC (BDC) motors for joint actuation, assumed to be shunt motors. One defining property of BDC motors is that they develop a torque T_m proportional to their armature current I_a[19]. With a constant armature resistance R_a, Ohm's law gives that the torque is proportional to the applied voltage V_a:

$$T_m \propto V_a (= R_a I_a) \tag{1}$$

The motor drivers supplying the motors act as voltage level shifters in the sense that they send the full motor supply voltage V_m to the motors when they are activated by the MCU's logic signals. In order to send a regulated voltage signal to a motor, a driver must be activated at such a frequency that the motor "sees" the resultant average voltage. This is the task of the PWM signal controlling the motor driver. The PWM signal is defined by a frequency f_{PWM} and a duty cycle DT. The PWM signal will be high for a portion of the PWM period p_{PWM} defined by the duty cycle, and low for the remainder. The PWM frequency f_{PWM} is the inverse of p_{PWM}. Thus the average voltage V_{avg} applied to a motor over the course of one PWM period is

$$V_{avg} = V_m \frac{DT}{f_{PWM}}, \tag{2}$$

and the torque developed by the motor is

$$T_a \propto V_m \frac{DT}{f_{PWM}}, \tag{3}$$

provided that f_{PWM} is sufficiently high for the motor coils to act as a low pass filter for the PWM signal.

5.3 Voltage regulation

This project uses linear voltge regulators. These are less efficient than switching regulators, but simpler and less expensive. The particular model used, LM317HV, is adjustable, meaning that the input/output voltage ratio may be adjusted by resistors placed externally to the unit. The conversion ratio is given by the unit's dropout or "headroom" voltage V_{REF} and the voltage divider ratio between the resistors R_1 and R_2. Additionally, the regulator consumes an adjustment current I_{ADJ} which has a minor impact on the output voltage.

$$V_{OUT} = V_{REF}(1 + \frac{R_2}{R_1}) + I_{ADJ}R_2 \tag{4}$$

5.4 Motor driver current sense

The motor driver current sense resistor R_{IROPI} should be chosen such that the maximum current deliverable by the motor driver I_{MAX} (4.1A) equates to the highest voltage measurable by the MCU's analog to digital converter (ADC) $V_{MAX,m}$. As the ADC is configured as single ended differential (see section 8.2.6) and the analog supply voltage V_{VDDA} is 3.3V, $V_{MAX,m}$ is also 3.3V.

Thus R_{IPROPI} is given as such:

$$R_{IPROPI} = \frac{V_{MAX,m}}{A_{IPROPI} I_{MAX}} \tag{5}$$

The conversion from ADC raw data value to measured voltage is given by

$$V_m = V_{MAX,m} \frac{n_m}{N_{ADC}}, \tag{6}$$

where n_m is the measured ADC value and N_{ADC} is the maximum ADC value as determined by its resolution. The conversion to motor current draw I_m is made by inserting for Ohm's law on the left hand side of equation 6:

$$I_m = V_{MAX,m} \frac{n_m}{R_{IPROPI} N_{ADC}} \tag{7}$$

6 System architecture

With the constraints of the refurbished system mapped out in the previous chapters, this chapter describes the process and result of the system's architectural design. The system components outside the arm are unchanged, that is, there will still be a need for an external power supply and a computer through which a user interface is offered. Thus, this chapter primarily concerns the inside of the arm.

6.1 Summary of constraints

- Number and shape of the PCB hardpoints.
- Wiring space limitations.
- Information transfer.
- Standardisation of parts

6.2 Design concepts

The following three concepts are outlines of the PCB design, oriented around the existing parts and a minimum of required/wanted parts from the system description. The primary purpose of the concepts is to explore solutions to the project's stated goal of reducing the system's overall complexity (i.e. number of PCBs and MCUs) in opposition to the limited space for wiring between the arm's internal sections (i.e. through the joint holes) – not provide detailed insight into what each PCB will actually look like or contain. Assumption 1 applies to all sections except the junction between the PCB mounting points in the linear bogie and the torso. M0-5 are motors, E0-5 are encoders.

It should be noted that these concepts were made before load cells were excluded from the refurbished design, before the optocouplers were discovered, and after the STM32 family of microprocessors had been suggested as a replacement MCU. Additionally, it was assumed that the motor drivers would be driven by digital to analog converter (DAC) output rather than by PWM due to a lack of prior research. The same holds for reading the encoder signals from ADCs. The reasoning for the choice of MCU is elaborated in section 7.1.1.

Assumption 1: It is effectively not possible to run more than four wires through the joints.

Consequence: The hand, arm lengths, torso and linear rail must to some extent be considered separate entities.

Unknown: A Controller Area Network (CAN) bus is sufficiently fast (1Mb/s) to support transfer of data to/from all six motors in addition to intermittent polling of load cells, IMUs etc to the MCUs.

6.2.1 Concept 1: One primary MCU

In this concept, illustrated in figure 5, one STM32Gx controls the whole system on orders received via USB from an outside master. The "simple" MCUs are a stand-in for whatever minimal solution is necessary to transfer data from the units onto the bus, ideally not even something programmable. Depending on the complexity of the simple MCUs, there may be little to no data buffering, possibly leading to data loss and/or bus contention. The bus splits on the lower part of the arm (shoulder/torso) do reduce traffic per bus. While centralised, this design makes poor use of the G series' several ADCs and DACs, and requires multiple unique PCB designs. Also depending on the choice of simple MCU, this design may severely violate **RA2: Complexity** (4.5.2) by requiring the design of two pinouts, maintain two sets of hardware drivers, et cetera. This may be a good

design if the math accelerators[24] of the G series were properly utilised and the simple MCUs required minimal programming.

6.2.2 Concept 2: Distributed control

This concept, illustrated in figure 6, distributes control to 6 identical STM32F3x series MCUs, with one master unit responsible for communication with the outside controller via USB. Lower traffic over the CAN bus as processing is done directly on the relevant MCU, compared to the previous design. Violates **RC1: Centralisation** (4.4.1) to some extent, but makes it easier to standardise the PCB design. PCB schematics and layout will also be significantly simpler as more functions are handled on board the MCUs. Guaranteed to only make use of one MCU model, which should be easier to program than the previous design. Of the three concepts, this is likely the closest to the original design.

6.2.3 Concept 3: Free wiring

An optimal scenario where Assumption 1 fails, and there is space for several extra signal wires in the joint holes. Two STM32Gx's handle roughly half the processing each, where one is also responsible for communicating with the outside world via its USB interface. This solution would make good use of the G series' peripherals, possibly even needing an additional external ADC to handle a load cell or IMU. CAN traffic becomes a non-issue as the only messages passed are for task coordination. The design has two virtually identical "main" PCB designs, with the remaining only having the required minimum to attach the relevant ICs to the arm chassis.

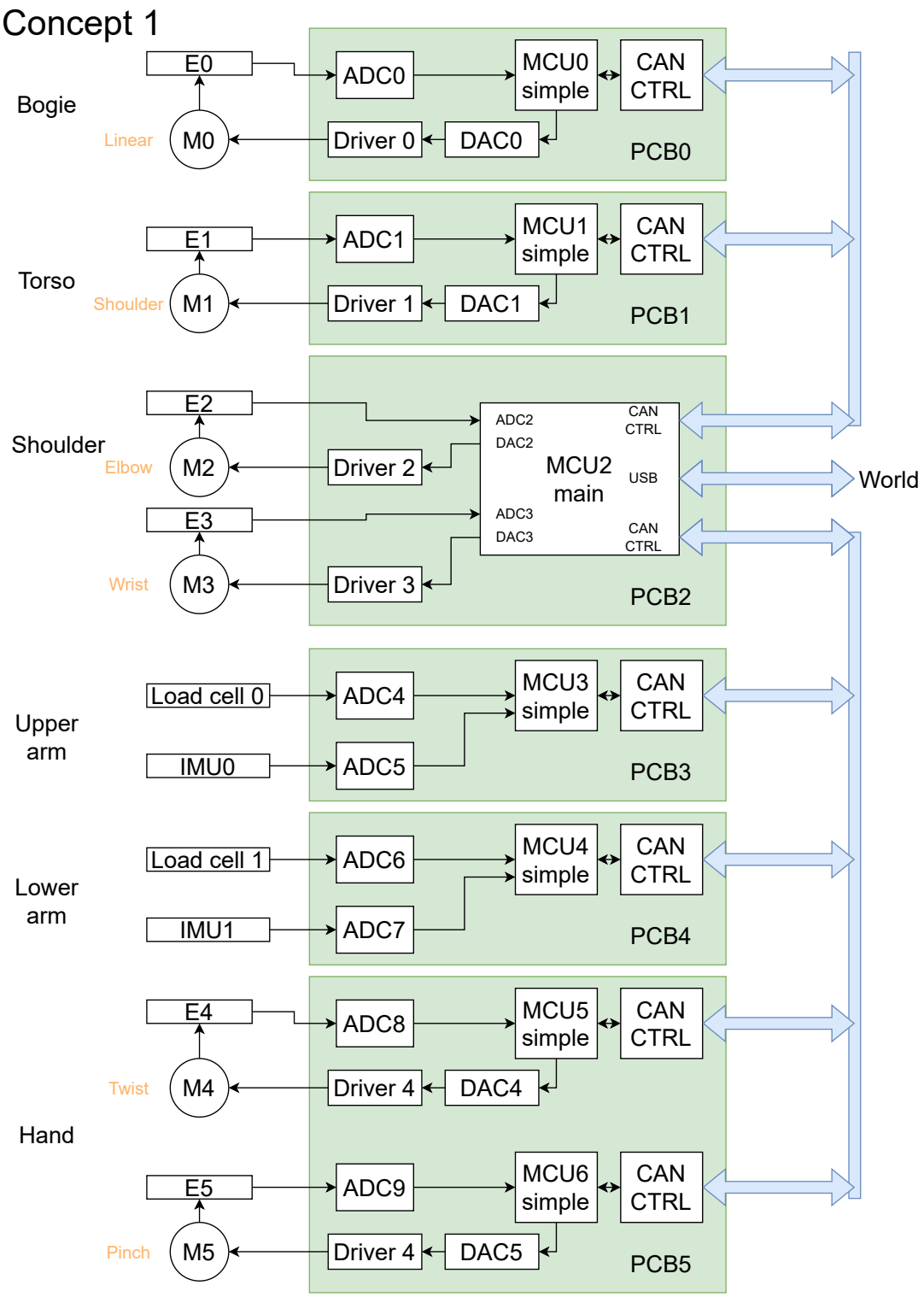


Figure 5: Architectural concept 1: One MCU

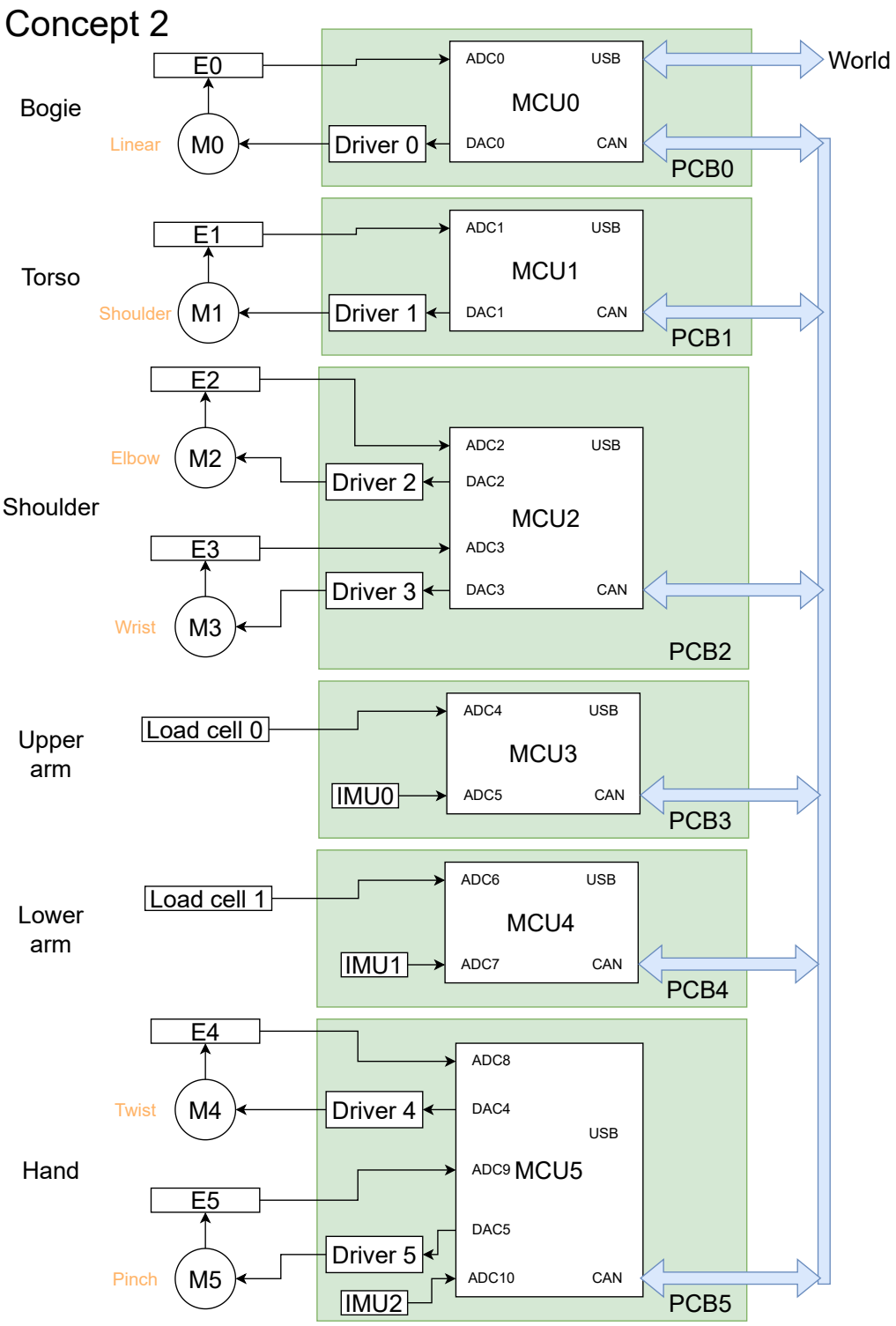


Figure 6: Architectural concept 2: Distributed control

Concept 3

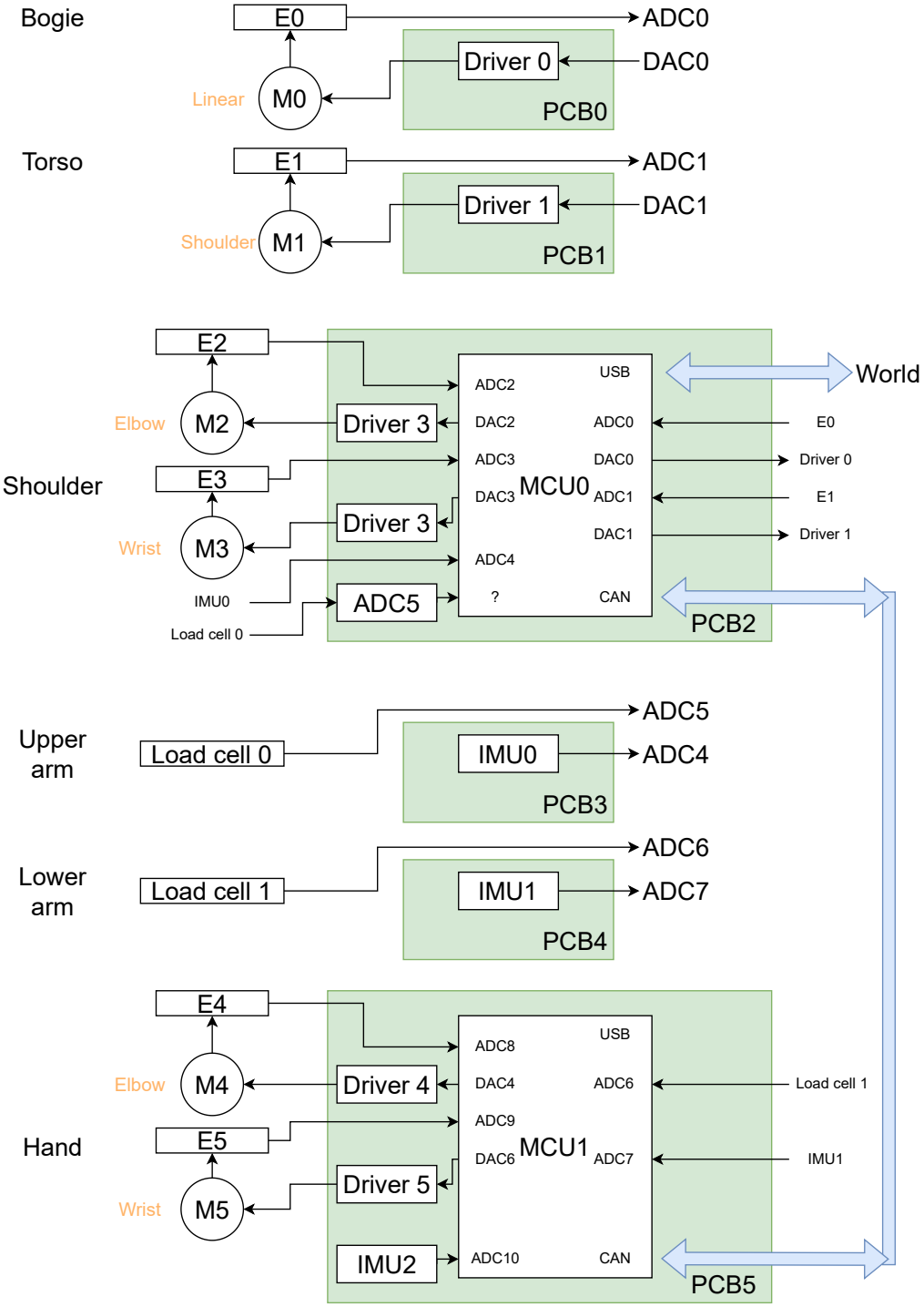


Figure 7: Architectural concept 3: Free wiring

6.3 Final design

6.3.1 Overview

The system consists of 7 PCB designs, of which 3 are control units, 2 are IMU mounts, and 2 are auxiliary units with little function other than maintaining compatibility with the existing parts. The first auxiliary (AUX0) is mounted in the rail and interfaces with the DSUB connector through which all information passes into and out from the arm. The second auxiliary (AUX1) connects the torso control unit with the bogie, where the linear motor and end switch are mounted. Relevant information is also passed on to AUX0. The control units CTRLn recreate the original control units, and are identical except for the MCU peripherals in use. The IMU boards primarily function as mounts for the IMUs.

The load cells were struck from the design in part because no data sheet could be found for the ones originally installed in the arm, and in part because finding new ones seemed like a significant increase in scope of an already ambitious project. As figure 8 was produced post hoc, unit names were changed to reflect the system implementation.

6.3.2 Internal communication: CAN bus and I2C

The CAN standard nominally supports transmission speeds of up to 1Mb/s, with some implementations supporting 5Mb/s. It is commonly used in industrial applications and its associated hardware is readily available from vendors, and is therefore chosen to implement **RC3.1: Internal communication**(4.4.4).

Additionally, as it was discovered that most IMUs use the Inter-Integrated Circuit (I2C) bus for communication, this replaces some of the ADC functions in the concepts.

6.3.3 External communication: Universal Serial Bus (USB)

Virtually all computers, and many modern MCUs, are equipped with USB2.0 interfaces. USB is therefore chosen as the primary implementation of **RC3.2: External communication**(4.4.5). As a backup solution, and solution for use during early testing, the system will implement Universal Asynchronous Reciever/Transmitter (UART).

6.3.4 Control units

The arm was inspected in more detail after the concepts were drawn, and concept 2 was chosen for further elaboration. Concept 1 was discarded because even an MCU just advanced enough to handle CAN and ADC/DAC conversion is still an MCU, and would warrant its own environment for programming, pinout design et cetera. This would result in having to handle at least two different MCUs, which would greatly increase the project's complexity compared to only having one. Concept 3 was discarded because even though **RM2: Wiring** (4.2.4) was somewhat more lenient than initially assumed, the junction between the shoulder and torso hardpoints would not have fit the necessary number of wires.

The final design therefore is quite similar to the original one, but reduces the number of processors from 6 to 3. Three control units located in the torso, shoulder and hand, respectively, are responsible for driving two motors each. Additionally, the hand control unit collects data from two IMUs located in the hand and lower arm as well as the wrist and twist optocouplers, the shoulder unit collect data from the upper arm IMU, and the torso unit registers the linear rail end switch. This is illustrated in 8. Lastly the torso unit is responsible for communication with an external computer by USB. Having three almost identical units was thought to streamline the circuit design, as only minor changes would be required between them. This followed from the assumption that it would

be possible to find an MCU with enough peripherals to support every function at once, i.e. only one MCU pinout would need to be designed for the entire system.

As this project mainly focuses on hardware design, little thought has been given to precisely what the MCUs will be communicating between themselves or to the external computer. The assumption is that the CAN and USB standards will be more than sufficient in 2023, to do what could be done in 1993.

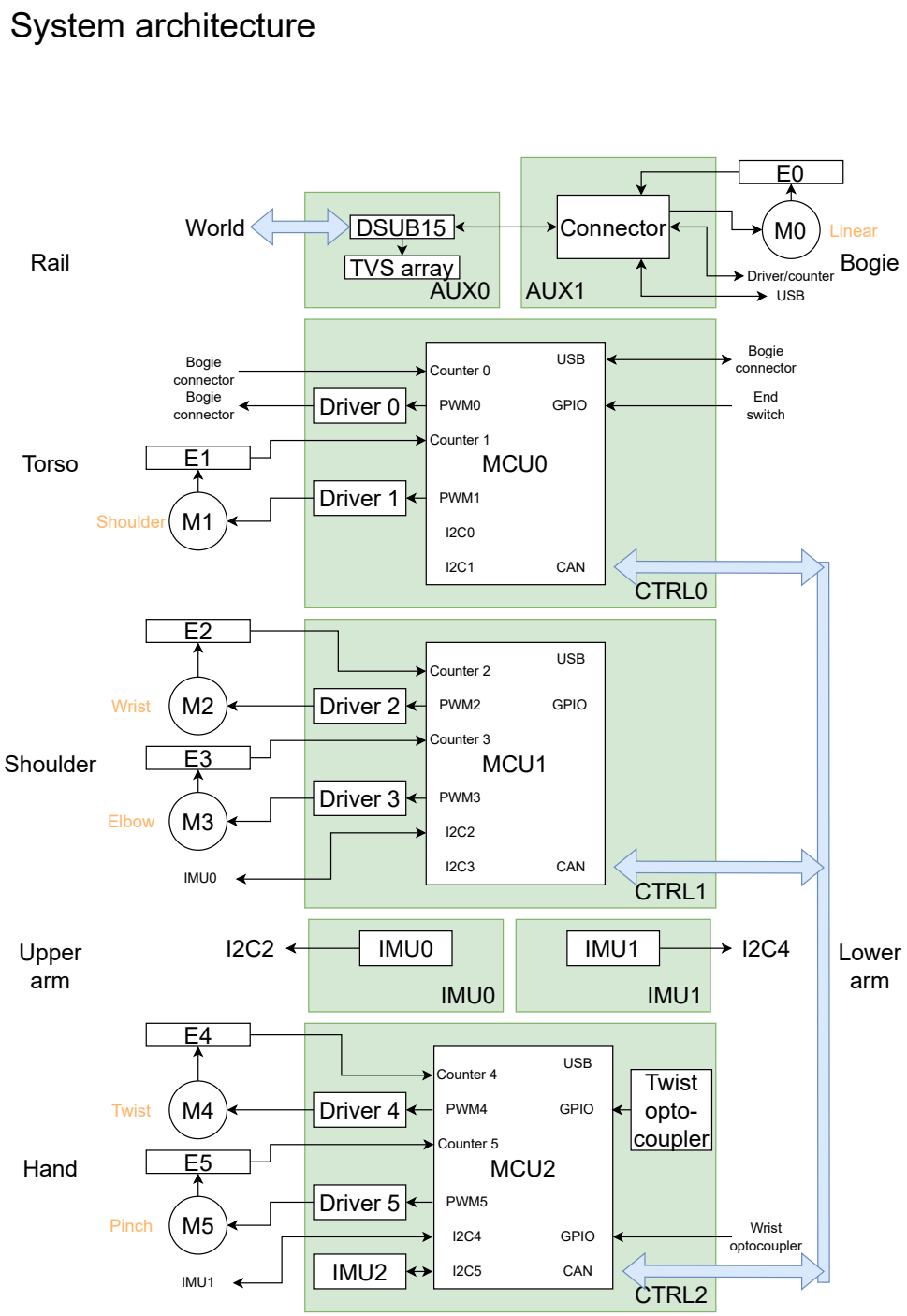


Figure 8: Architectural overview of the refurbished system. The naming scheme has been changed from the concepts to better reflect the implementation.

7 Component selection

Components were chosen with the constraints presented in section 4 in mind, as well as requirements implied by the design presented in section 6.3. The integrated circuits (ICs) are presented in the order in which they were selected, while the remaining components are ordered roughly by how much thought was given to their selection.

7.1 Integrated Circuits

7.1.1 Microcontroller unit: STM32F303RE

The MCU is at the heart of the project, but choosing one is a chicken-and-egg problem. On the one hand, an MCU must be chosen which can support the project's required functions, and on the other, the choice of MCU will dictate much of the project's further implementation. This illustrates the importance of adequately mapping out the requirements of a project before making design desicions. The choice of MCU is constrained primarily by the control requirements listed in section 4.4 and **RA2: Complexity**(4.5.2). In order to fulfill **RA2** in particular, the MCU should be able to support the sum of all unique functions, as illustrated by the architecture, on a single pin configuration. This way, only one pin configuration will be needed in the project. Those functions are:

1. 2 PWM signals for motor control (F1.1)
2. 2 quadrature encoder counters for motor control (F1.1, RC2.1)
3. 1 CAN interface (RC3.1)
4. 1 USB interface (RC3.2)
5. 2 I2C interfaces (RC2.6)
6. 2 ADCs for motor current measurement (RC2.4)
7. 2 GPIO for switching signals (RC2.8)

As described on ST's webiste about their "mainstream MCUs"[24], initially suggested by the project's supervisor, the STM32 G4 and F3 series are intended for motor control and instrumentation. This seemed like a reasonable entry point for selection. The G4 series is advertised to have math accelerators in addition to motor control and USB interfaces, and can be clocked up to 170 MHz[26]. This seems like an attractive choice, and considering that the exact computational load the MCUs would be under has not been clearly defined, the math accelerators may prove valuable. However, putting this functionality to use may prove difficult to someone with limited experience with embedded programming. The F3 series was eventually selected because it seemed like a reasonable compromise between complexity and capability. The F3 can be clocked up to 72 Mhz, and appears to have the same functionality as the G4 except for the math accelerators.

Narrowing down the selection, STM32F303 was chosen over the other F3 units because it had the largest memory and advertised "up to" 3 DAC peripherals[25], which at the time were thought to be necessary for motor control. In hindsight, the number of PWM compatible timing and encoder reading peripherals would have been a better criterion. It was later discovered that the F303 can support up to four PWM outputs and/or encoder inputs, which is also sufficient. Additionally, it satisfies all the items from the above list as well as having a peripheral interconnect matrix, interrupt mappable GPIO and several 5V tolerant pins. The RE variety was finally chosen for its LQFP64 package and large memory. It was thought that a lower pin number may have been insufficient, while the 0.5mm pitch of the 100+ pin packages may be impractical to solder.

7.1.2 Motor driver: DRV8251A

The choice of motor driver is dictated by the requirements of the motors, as specified by **RE1.1** (4.3.1). An attempt at finding original data sheets for all 6 motors was unsuccessful, so the datasheet of the H-bridges were used as a guideline. Respecting **RA2: Complexity** (4.5.2), the same model of motor driver was chosen for all 6 motors: Using a smaller or less capable driver for the smaller motors may have required additional circuitry for stepping down the bus voltage before feeding it to the drivers. As they are still in stock at the preferred vendor, using the original model, LMD18200T, would have been possible. However, they are prohibitively expensive at 285NOK/pc[3].

The Texas Instruments DRV family of brushed DC motor drivers was recommended[4] as a cheap and relatively simple driver. DRV8251A was chosen for its peak output current of 4.1A, maximum motor supply voltage of 48V, unit supply voltage of 3.3V, and current sense output (**F2, RC2.4, RC2.7**(4.4.2))[34]. This is a violation of **RE1.1**, but considered acceptable as the system's intended power supply is rated to 48V. A major advantage of the DRV8251A over LMD18200T is that it costs 10NOK/pc at the preferred vendor.

7.1.3 Intertial measurement unit: LSM6DSMTR

It was originally thought that the IMU would interface with the MCU by analog input. However, a cursory search at the preferred vendor's website[5] revealed that most units use I2C or SPI, doing the ADC conversion internally, and that the units commonly support multiple modes of measurement: not only acceleration, but also turn rate. The LSM6DSMTR was chosen because it was in stock, can be limited to 2g acceleration in three axes, and is compatible with a 3.3V supply voltage[22]. Additionally it can be programmed to trigger interrupts under various conditions, and supports turn rate measurements. The unit implements **RC2.6**(4.4.2).

7.1.4 CAN transceiver: TJA1057BT

The CAN tranciever implements **RC3.1: Internal communication** (4.4.4). No particular requirements were imposed on this selection other than that the component should be advertised as a CAN transciever. The TJA1057BT was chosen because it was in stock at the preferred vendor, and is able to interface with a microcontroller operating on 3.3V without additional hardware[16]. It is compatible with the high-speed CAN implementation of 5Mb/s, and requires a 5V supply.

7.1.5 Voltage regulator: LM317HV TO220

The voltage regulators must adhere to **RE1.2**(4.3.1). The system's intended power supply outputs 48V, and the units which the regulators are to supply require either 5V or 3.3V. Respecting **RA2: Complexity** (4.5.2), the same model of regulator should be able to supply all units and it should be able to step down a voltage differential of 44.7V. This limits the selection to adjustable regulators. Switching regulators are known to be more energy efficient than linear voltage regulators, but typically require a more complex control circuit. For that reason the linear voltage regulator category was chosen. Limiting the search to adjustable regulators with an input voltage between 50V and 65V yielded only the LM317HV. At a voltage differential of more than 15V, it may supply up to 0.3A[35]. A cursory investigation of the datasheets of the parts supplied by 3.3V and 5V, respectively, resulted in an estimated 320mW and 800mW for the two voltage levels. This is well within the 0.3A limit in both cases.

A TO-220 package was chosen over an NDS package because it was the most similar to the L4960

[3]https://no.mouser.com/ProductDetail/Texas-Instruments/LMD18200T?qs=7lkVKPoqpbafhq6dHaRGuw%3D%3D, Dec 10 2023

[4]By Omega Verksted

[5]no.mouser.com, "inertial measurment unit", sort price low-high

used in the original system, which was assumed to make it easier to integrate with respect to **RM1.2: PCB mount points**(4.2.3).

7.1.6 TVS array: CDSC706-0504C

The Transient Voltage Suppression (TVS) array is part of the implementation of **F2: Ensure limits of safe operations are not exceeded** (4.1), and protects the data lines of the USB and UART buses as they enter the arm in the rail. The array consists of 4 diodes with a breakdown voltage of 6V[6].

7.1.7 Optocoupler: OPB971N51

One could argue that this fits better in the generic category, but it is presented here because it implements one of the system's functions, **F1.1: Control the position of 6 joints concurrently**, aiding in pose estimation. This model is the same as the original part, and was selected for that reason: respecting **RM1: PCB Form factors**(4.2.1), it must fit with a mechanical plate rotating with the twist joint. It requires a 5V supply[36].

7.1.8 Motor relay: JV3SKT

The original control units use a relay in conjunction with the motor drivers. Their exact function was not established, but it is assumed that they were part of a safety function. The refurbished system uses the same model of relay, a normally open circuit with a 3.3V activated coil[9]. It is incorporated into the motor driver circuit such that the coil must be active in order for current to reach the motors, i.e. a "dead man's switch". It is a key element in the implementation of **F2: Ensure limits of safe operations are not exceeded**.

7.2 Connectors

7.2.1 VBUS connectors: 3.5 mm 4P Phoenix screw terminal

The voltage bus (VBUS) connectors ensure that the CAN bus and system supply voltage (48V) reaches the entire arm, and thus are part of the implementation of **RE1.2** (4.3.1) and **RC3.1: Internal communication** (4.4.4). The original system used a plug connector, but an attempt at finding the exact model was not successful.

7.2.2 20 pin connector socket: TE 1761681-7

This connector was chosen to be as physically similar to the original 20 pin ribbon cable connector as possible[33]. The space connecting the torso to the bogie is limited, and replacing the original ribbon cable did not seem worthwile as it is optimised for its purpose. The connector was found by searching for 20 pin board-to-board ribbon connectors with a 2.54 pitch on TE Connectivity's website, and then cross-checking availability at the preferred vendor. TE was chosen for the initial search because they are known to offer a wide selection of connectors.

7.2.3 16 pin connector socket: TE 2-1761603-6

This connector couples the bogie to the rail entry point via a 16 wire ribbon cable installed in the rail[32]. Similarly to the torso-bogie ribbon cable, this cable is intact and well optimised for its purpose, so no replacement was considered. The connector was found in a similar manner to the 20 pin, but sorting for 16 pins.

7.2.4 Motor connector socket: Molex 70543-0001

The motor connectors were chosen to be as similar to the original motor connectors as possible. While replacing them or using a generic 2.54 mm pitch pin header may have been feasible, it was thought that the motor connectors should be properly attached to their control units. The process of finding it was similar as for the other connectors, but this time searching Molex' website.

7.2.5 Encoder connector pin header: 2x5 2.54 mm generic

Encoder connectors are generic 2.54 mm pitch male pin headers found at Omega Verksted, arranged in a 2x5 grid. Of the 10 sockets on the female 2x5 connector on the encoders' ribbon cables, 2 are sealed so as to provide a non-ambiguous attachment to the header. The headers were modified to accomodate this. The headers are part of the implementation of **RC2.1: 6x HEDS encoder pulse train signals** (4.4.2).

7.3 Generic circuitry components

7.3.1 Bulk capacitors

Each BDC motor has one bulk capacitor associated with it, its capacitance related to the motor's specifications. The capacitors were chosen such that they would have an identical capacitance and voltage rating equal to or greater than those of the original parts.

7.3.2 Oscillator: ABL-1774766

The oscillator crystal frequency of 16 MHz was chosen automatically by CubeMX (introduced in 8.1), ST's tool for STM32 hardware initialisation. After the peripherals relevant to the project were selected in CubeMX, the software ran a clock solver and suggested a 16 MHz external oscillator be used. This particular model was chosen because it was affordable and physically similar to what is suggested by the STM's reference schematic[27].

8 Circuit design

For circuit design, the architecture was segmented into "assemblies" consisting of one or more IC and their/its remaining circuit components. These assemblies could then be reused across boards with little modification. Section 8.2 describes each assembly, and the following subsections describe how they were used or modified in each instance. The circuit designs were generally based on application references in the relevant ICs' datasheets.

Language note: The assemblies are defined with distinct inputs and outputs. This is a conceptual interface definition, and does not necessarily represent the physical layout of a unit. Typically the phrase "outputs the same" refers to input and output being physically the same pins/wires.

8.1 Tools

The software package KiCad:EESchema was used for the circuit design. ST's CubeMX software was used for the MCU pinout design. Both of these are available for free, although CubeMX requires email registration at the ST website.

8.2 Assemblies

8.2.1 Low voltage assembly

The low voltage assembly, presented in figure 10, consists of two LM317HV voltage regulators, their adjustment and protection circuits, and two indication LEDs. The assembly takes in the 48V system supply voltage and ground reference, and outputs 5V, 3.3V and a ground reference. The indicator LEDs are connected between V_{out} and GND in series with 240Ω resistors, and thus indicate whether there is an output voltage from the regulators or not.

The remainder of the circuit is based on figures 15 and 16 from the LM317's datasheet[35], see figure 9. The protective diodes D[3..6] were introduced out of an abundance of caution, even though the datasheet specifies that these are unnecessary for output capacitances of less than $25\mu F$. The adjustment resistors' "R1" values were set to 240Ω per the datasheet, while the "R2" values were set to $0.386k\Omega$ and $0.706k\Omega$ for 3.3V and 5V output, respectively. In the design, these are represented by three-terminal potentiometers. The values were calculated from equation 4, where $I_{ADJ} = 50\mu A$ and $V_{REF} = 1.25V$ can be found in section 7.5 of the data sheet.

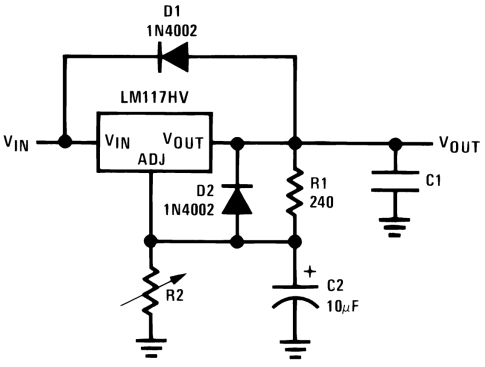


Figure 15

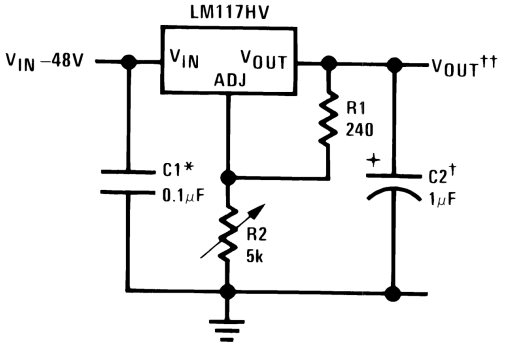


Figure 16

Figure 9: Low voltage assembly reference schematic

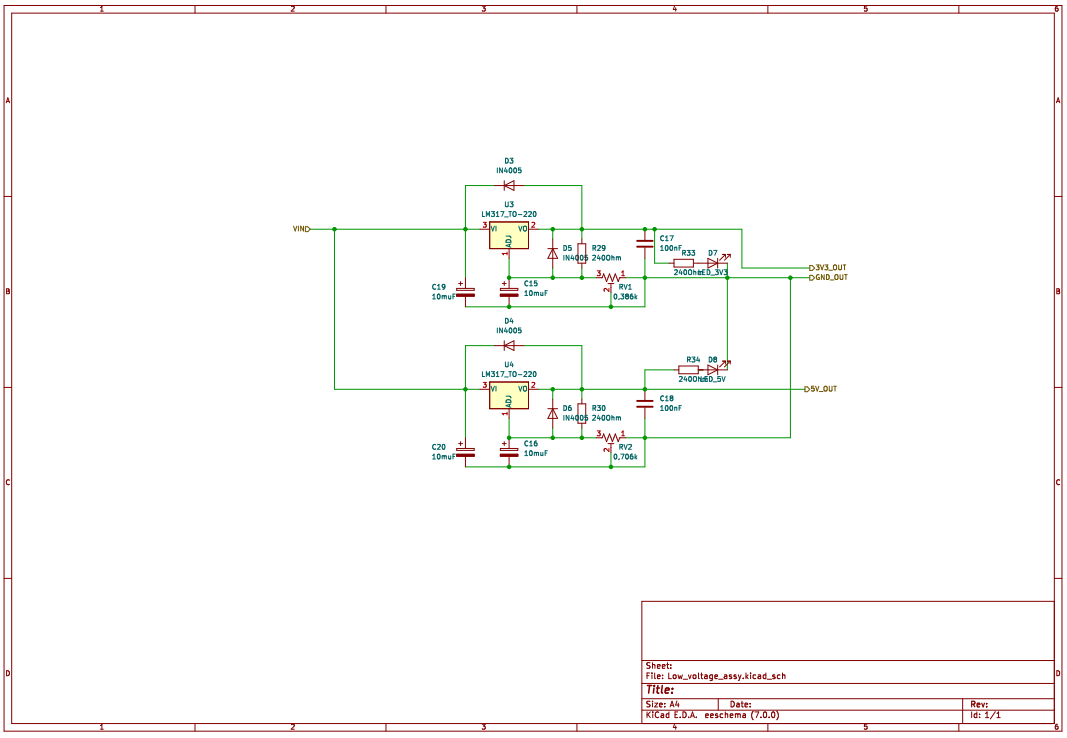


Figure 10: 3.3V and 5V regulator circuit

8.2.2 Motor driver assembly

The motor driver assembly, presented in figure 12, consists of two DRV8251A H-bridge motor drivers, the motor connectors and the motor relay with a control MOSFET. The driver circuit is based on figure 9-1 in the datasheet[34], see figure 11. The assembly takes in 3.3V for supply of all units, 2xPWM for each driver, a relay enable signal, and a ground reference. It outputs a motor current proportional voltage, $DRVn_{IPROPI}$, for interpretation by the MCU, and a PWM modulated voltage signal to each motor.

The values for the filter capacitors C3 and C4 were chosen per the datasheet, while the bulk capacitors for the motors were chosen based on the old hardware. The R_{IPROPI} value of 510Ω was chosen based on equation 5.

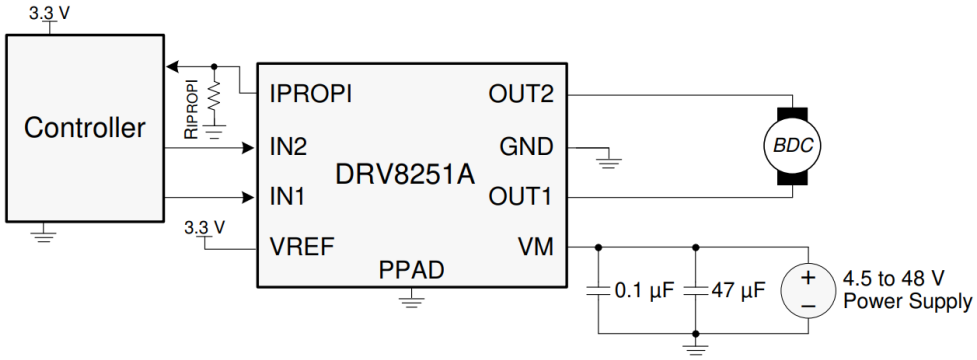


Figure 9-1. Typical Connections

Figure 11: Motor driver reference diagram

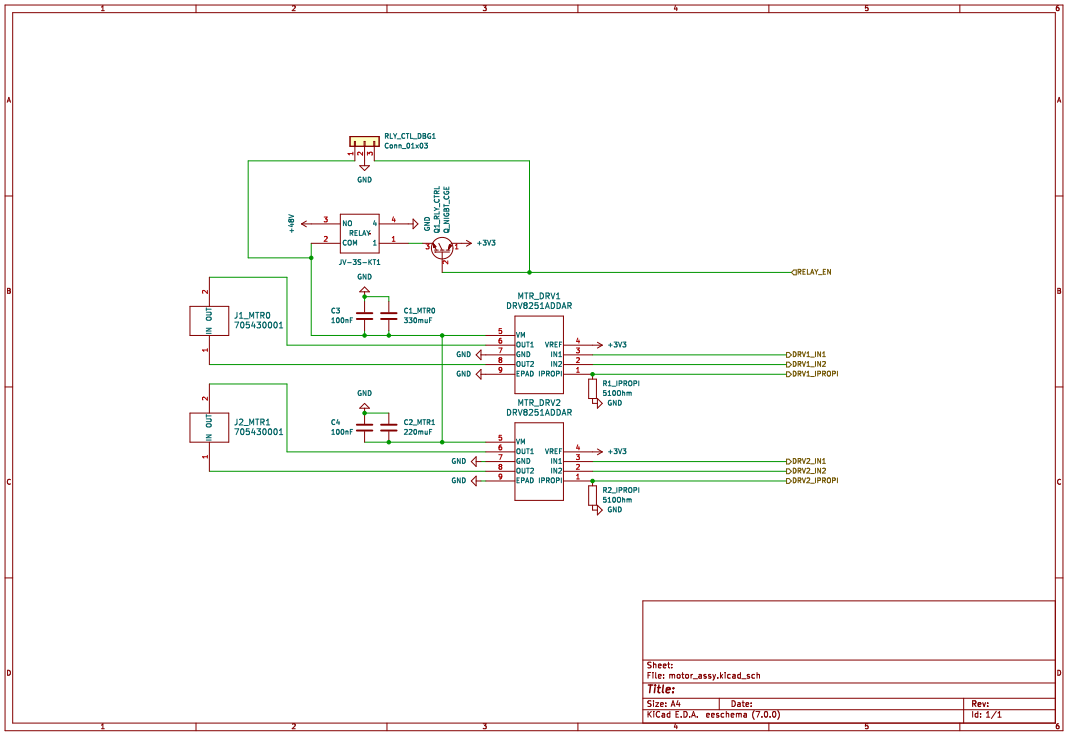


Figure 12: Motor driver circuit

8.2.3 CAN assembly

The CAN assembly, seen in quadrant 2A of figure 21, consists of the TJA1057BT CAN transciever unit and its power circuit. It takes in 5V power supply, 3.3V logic reference, ground reference, chip select, data received from its associated controller (CAN_RX), and the two-way CAN high/low signal bus between the other control units. It outputs the same bus, and data transmitted to its associated controller (CAN_TX). Note that the schematic uses the GT transceiver model, but this has no impact on the circuit design. The circuit is based on Figure 6 in the unit's datasheet[16], see figure 13.

The CAN bus requires a 120Ω termination resistor in each end. The datasheet recommends two 60Ω resistors in series, with a decoupling capacitance in parallel, but this was omitted from the design for the sake of simplicity. Otherwise, no particular design consideration apply to this assembly.

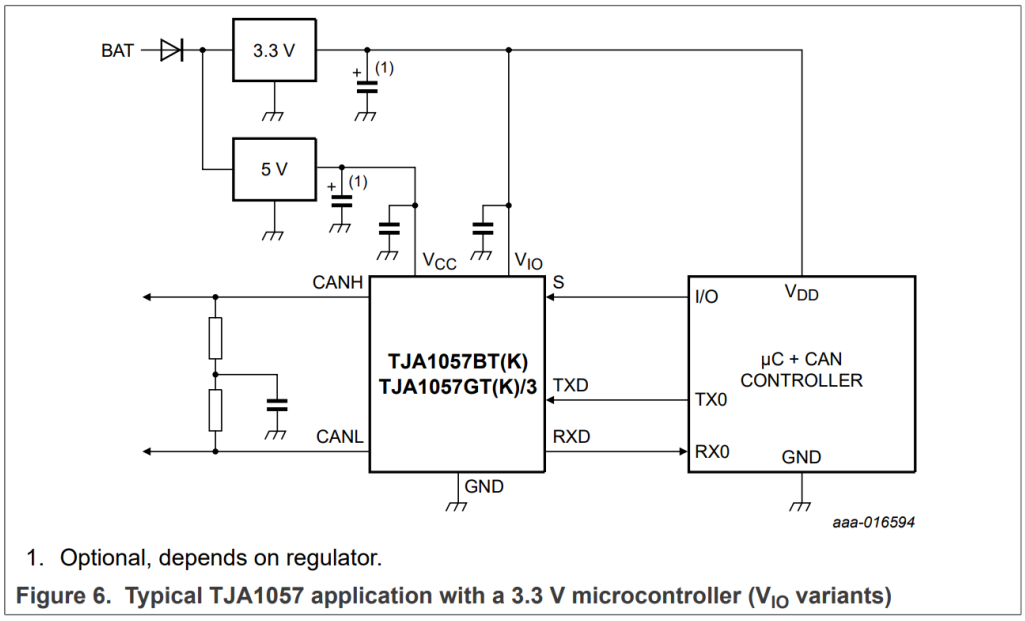


Figure 13: CAN assembly reference schematic

8.2.4 IMU assembly

The IMU assembly, presented in figure 15, consists of the LSM6DSMTR IMU unit, its power circuit and I2C bus. The unit takes in 3.3V, a ground reference and the I2C bus from its associated control unit. It outputs the same I2C bus and a programmable interrupt. The circuit is based on figure 17 in the unit's datasheet, which illustrates the unit as an I2C slave[29]. See figure 14. Design choices:

- Pin 1: In I2C mode, this pin decides the last bit of the unit's I2C ID number(table 2, LSM6DSM datasheet[22]). It may be pulled low or high, which allows two units to be connected to the same bus. As the MCU has enough I2C ports to allow one IMU per port, all IMUs have Pin 1 pulled low.
- Pin 2,3: In I2C mode, these pins have no function and may be pulled high or low, but cannot be left floating. They were pulled high because it is recommended they be pulled high for optimal power consumption during start-up (AN4987 chapter 3.6[29]).
- Pin 4: One of two programmable interrupt output pins. It was included in the output because it gives some freedom for software design, but is not intended to implement any particular function.

- Pin 9: One of two programmable interrupt output pins. It was pulled to ground to simplify layout and/or save wiring space in the arm depending on the use case, its pull-down resistor value of $50k\Omega$ based on best practice for unused pins[27][7].
- Pin 10, 11: Not connected, as recommended by the datasheet.
- Pin 12: The chip select pin is pulled high, leaving the unit always on. This would simplify layout and/or save wiring space in the arm depending on the use case. As this project is not particularly power sensitive, it was not deemed necessary to be able to turn the unit off.
- Pin 13, 14: The pullup resistor values of $10k\Omega$ were recommended in the datasheet as a typical value. While I2C bus bitrate is affected by the pullup resistor value, it was not deemed necessary to choose a non-standard value for this project.

Figure 17. LSM6DSM electrical connections in Mode 1

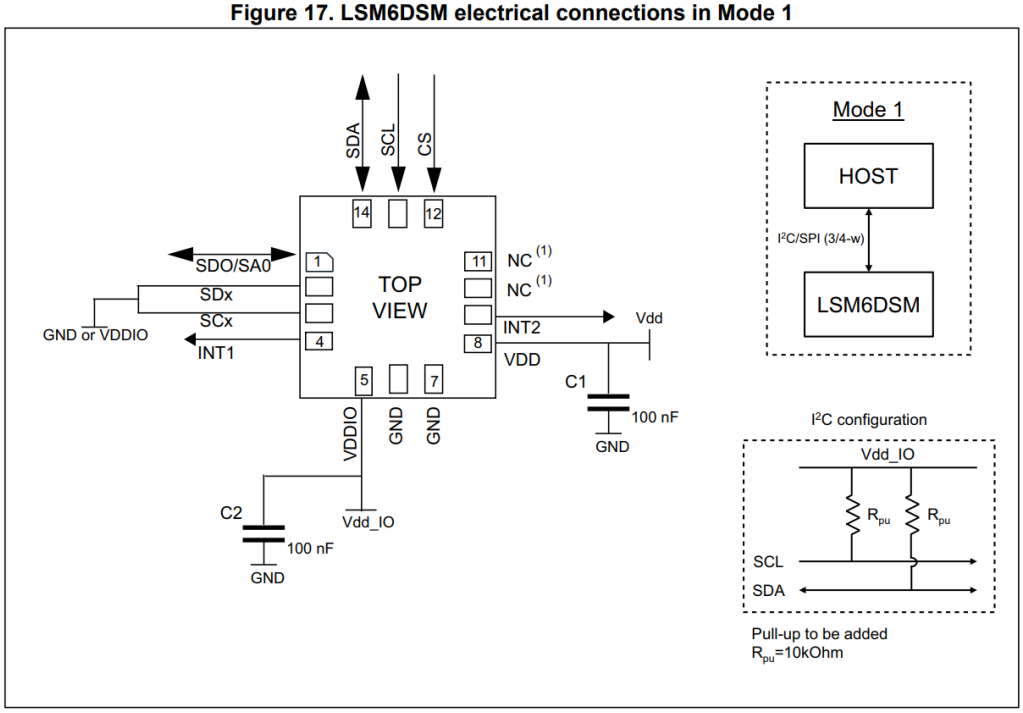


Figure 14: IMU assembly reference diagram

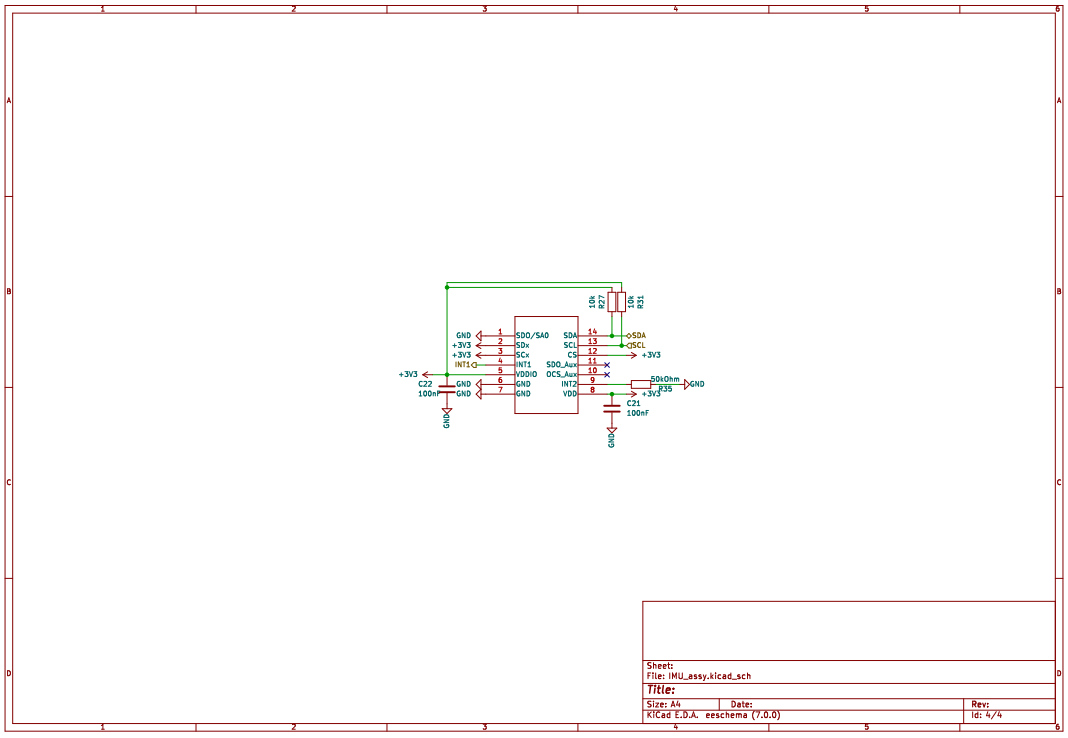


Figure 15: IMU assembly circuit

8.2.5 USB assembly

The USB assembly, seen in quadrant 4A of figure 21, consists of the 4 USB lines 5V (VBUS), GND, DP and DM, a USB connector and the MCU's USB interface. While the data lines and VBUS are eventually pulled to the DSUB connector in the rail, the connector was included for ease of access during testing. Connector takes in a 5V logic reference, ground reference, and the USB's data lines. It outputs the same data bus to the MCU's USB interface, which is 5V tolerant (table 13, MCU datasheet[23]), and a stepped-down VBUS signal for bus detection. The connector is in parallel with the lines reaching the rail, so only one of the two access points may be used at any given time. The circuit is based on figure 5 of ST's Application Note (AN) 4879 for USB hardware[28], see figure 16.

Table 3 of AN4879 specifies that STM32F303 does not have an embedded pullup for the DP line, and the control units are "self powered". Figure 5 was selected as design reference for those reasons. The values of the voltage divider resistors R3 and R8 for VBUS detection, $30k\Omega$ and $20k\Omega$ respectively, were chosen as typical values for stepping down from 5V to 3V. 3V is within the minimum logic high voltage of the MCU's 5V tolerant pins, specified to 1.85V for 3.3V Vdd in table 66 of the datasheet[23].

Note: AN4879 states that the pullup must be on the DP line, while figure 21 shows that it was in fact connected to the DM line. This error is further discussed in chapter 13.

Figure 5. **USB FS upstream port without embedded pull-up resistor in self-powered applications**

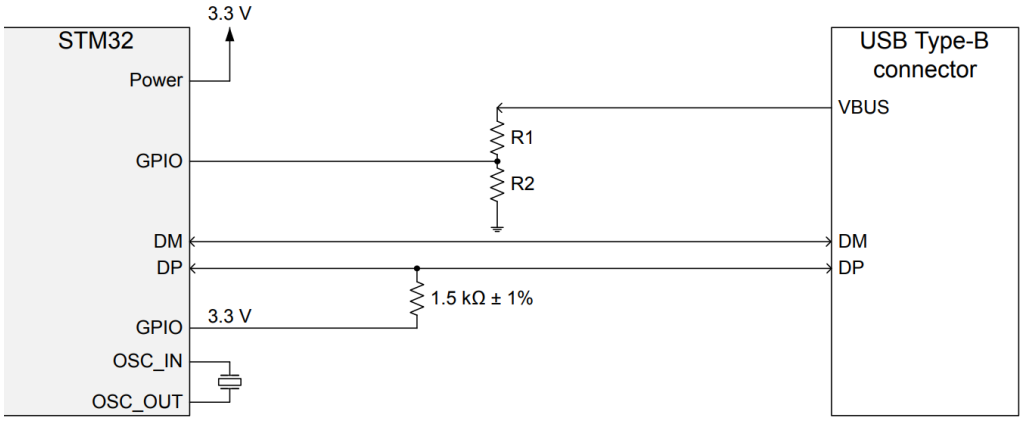


Figure 16: USB assembly reference schematic

8.2.6 MCU pinout design

The design of the MCU's pin configuration was largely constrained by the availability of peripherals on a given pin. Some effort was made to avoid trace crossing for the PCB design. The pinout is summmarised in table 4, and illustrated in figure 17. An automatically generated report further detailing the initialisation of peripherals, etc., can be found in the appendix.

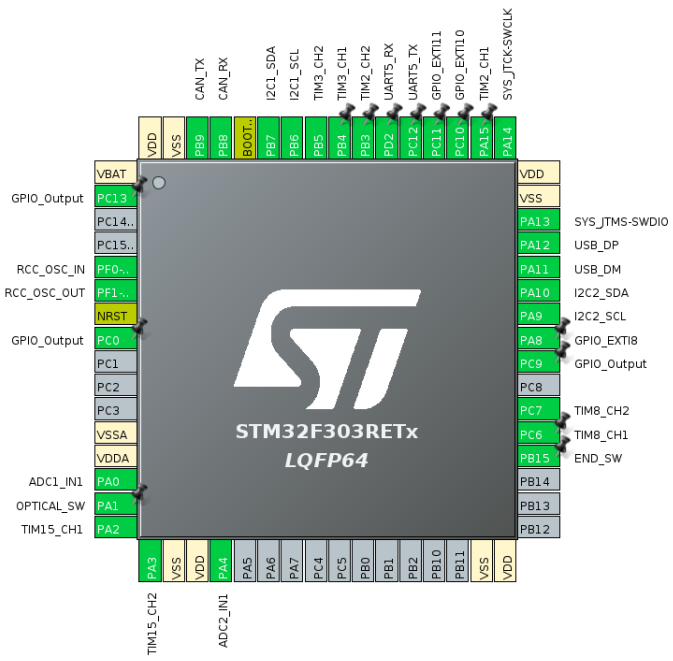


Figure 17: Configuration of pins in CubeMX

Table 4: Pin configuration of the MCU. Unmentioned pins are in RESET state (HiZ)

Pin nb	**PIN**	**FUNCTION**	**LABEL**
1	VBAT	Battery power in	
2	PC13	GPIO_Output	
5	PF0-OSC_IN	RCC_OSC_IN	
6	PF1-OSC_OUT	RCC_OSC_OUT	
7	NRST	NRST	Reset
8	PC0	GPIO_Output	
12	VSSA	Analog ground ref	
13	VDDA	Analog voltage ref	3.3V
14	PA0	ADC1_IN1	DRV2_VIPROPI
15	PA1	GPIO_EXTI1	OPTICAL_SW
16	PA2	TIM15_CH1	DRV2_IN2
17	PA3	TIM15_CH2	DRV2_IN1
18	VSS	GND	
19	VDD	3.3V	
20	PA4	ADC2_IN1	DRV1_VIPROPI
31	VSS	GND	
32	VDD	3.3V	
36	PB15	GPIO_EXTI15	END_SW
37	PC6	TIM8_CH1	ENC0_CHA
38	PC7	TIM8_CH2	ENC0_CHB
40	PC9	GPIO_Output	
41	PA8	GPIO_EXTI8	
42	PA9	I2C2_SCL	
43	PA10	I2C2_SDA	
44	PA11	USB_DM	
45	PA12	USB_DP	
46	PA13	SYS_JTMS-SWDIO	
47	VSS	GND	
48	VDD	3.3V	
49	PA14	SYS_JTCK-SWCLK	
50	PA15	TIM2_CH1	DRV1_IN2
51	PC10	GPIO_EXTI10	
52	PC11	GPIO_EXTI11	
53	PC12	UART5_TX	
54	PD2	UART5_RX	
55	PB3	TIM2_CH2	DRV1_IN1
56	PB4	TIM3_CH1	ENC1_CHA
57	PB5	TIM3_CH2	ENC1_CHB
58	PB6	I2C1_SCL	
59	PB7	I2C1_SDA	
61	PB8	CAN_RX	
62	PB9	CAN_TX	
63	VSS	GND	
64	VDD	3.3V	

8.2.7 MCU power supply assembly

The MCU power supply assembly can be seen in quadrants 3B and 3C of figure 21, and is based on the reference design presented in figure 13 of ST's AN4206[27], see figure 18. It consists of 7 decoupling capacitors, a zener diode with a reverse voltage of 3.3V and an indicator LED. The zener was incorporated as a surge or overvoltage protection measure, with the indicator LED in series to alert the user of such an incident. Table 6 of AN4206 recommend connecting V_{BAT}, which may be used in external battery applications, to V_{DD} when no external battery is used. The analog peripheral may be given a separate voltage reference through V_{DDA}, but this was selected against for the sake of simplicity.

- C4: $4.7\mu F$ decouples the power circuit as a whole,
- C5: $1\mu F$ decouples V_{DDA},
- C6: $10nF$ decouples V_{DDA}, and
- C[7..10]: $100nF$ decouple one V_{DD} each.

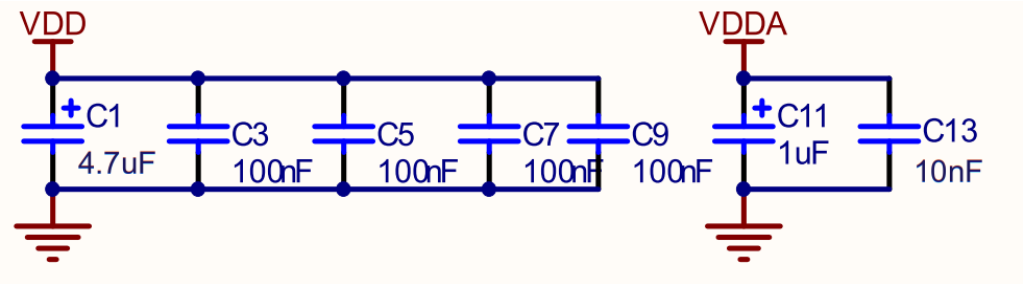


Figure 18: MCU power supply reference schematic

8.2.8 External oscillator assembly

The external oscillator assembly can be seen in quadrant 4D of figure 21, and is based on the reference design in figure 13 of ST's AN4206[27], see figure 19. It consists of the 16MHz oscillator crystal, two filtering capacitors and an external resistor. The assembly interfaces with the two external oscillator pins of the MCU.

The filtering capacitors C11 and C12 were set to 20pF, per the reference design. The external resistor R4 was set to 340Ω as Omega Verksted had none available at 390Ω, the reference value, for the preferred pad size of 0805. ST's AN2867 chapter 3.5.3 was interpreted to recommend a lower, rather than higher, value than the reference for the external resistor.

8.2.9 Reset and bootmode assemblies

The reset and bootmode assemblies can be seen in quadrants 2C and 3D of figure 21, respectively, and are based on the reference design in figure 13 of ST's AN4206[27], see figure 19. The reset assembly consists of a pushbutton in parallel with a filtering capacitor, and a line to the MCU's reset pin. The MCU is reset when the pushbutton is pressed pulling the reset pin low. This function is intended for development use, as the button is never accessible while a board is installed in the arm. The bootmode assembly consists of a jumper, a resistor, and a line to the MCU's BOOT0 pin. According to table 1 of AN4206: When the pin is pulled up, the MCU may boot from either system memory or SRAM. When the pin is pulled down, the MCU will boot from its main flash memory. The pin is pulled low when the jumper is not installed.

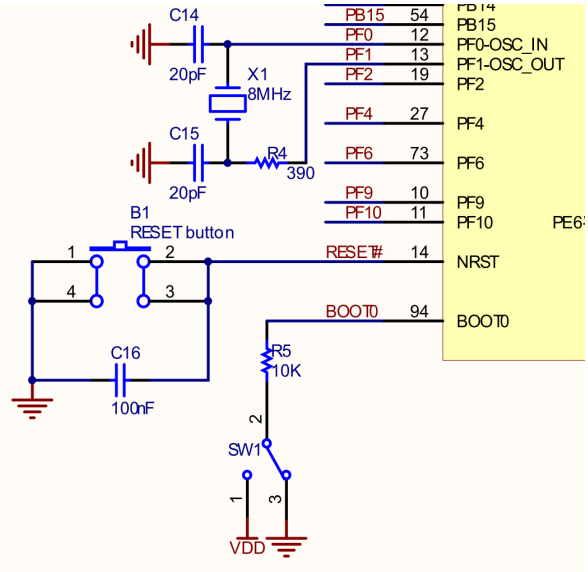


Figure 19: Oscillator, reset and bootmode reference schematic

8.2.10 Flash header pin assembly

The flash header pin assembly is used for programming/flashing the MCU, and can be seen in quadrant 4A of figure 21. It is based on table 5 of the STM32F303-NUCLEO development kit datasheet[30], as the same ST-LINK programmer would be used to flash the MCUs of the refurbished control units, see figure 20. It consists of a 1x5 2.54mm header. The header takes in 3.3V and ground references, the data and clock signal bus of the MCU's flash pins, and the MCU's reset pin. It sends all 5 signals to the ST-LINK programmer.

Table 5. Debug connector CN4 (SWD)

Pin	CN4	Designation
1	VDD_TARGET	VDD from application
2	SWCLK	SWD clock
3	GND	ground
4	SWDIO	SWD data input/output
5	NRST	RESET of target STM32
6	SWO	Reserved

Figure 9. Using ST-LINK/V2-1 to program the STM32 on an external application

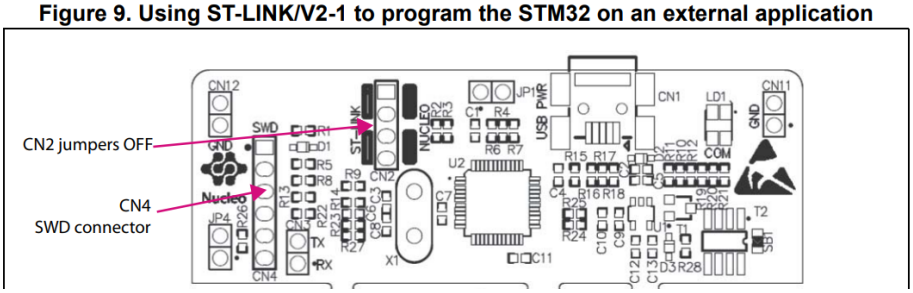


Figure 20: Flash header reference table and ST-LINK illustration

8.2.11 Encoder connector assembly

The encoder connector assembly can be seen in quadrant 1C of figure 21. It consists of one 2x5 pin 2.5 mm connector header. It takes in 5V power supply, ground reference and encoder signals from the encoder, and outputs the same to the encoder and MCU respectively. After it was determined that the encoders would work using only 4 of the 10 pins of the encoder connector, a decision was made to leave all other pins unconnected. Their function was undetermined, and pulling them high or low may have caused damage to the system.

In order to find which pin on the connector served what purpose, the datasheet of the HEDS-9100 encoder[2] was inspected to find its pin configuration. Each of its 5 pins on the wrist motor (arbitrarily chosen) were tested for continuity with each of the 10 connector pins, and a 1:1 match with 4 of the 5 encoder pins was found. The remaining pin is not connected, so this result was expected. A 5V supply was connected to the encoder, the two output channels connected to an oscilloscope, fingers were crossed, and the wrist joint was moved manually in the hopes of observing a pulse train on the oscilloscope. This was successful, and the remaining pins of the connector were consequently not given more attention.

8.3 Torso

The torso control unit, circuit schematic presented in figure 21, makes use of all the assemblies previously mentioned except for the IMU. Additionally, instead of using two encoder connectors, one set of signals is sent to the bogie. The following subsections discuss design features unique to the torso design, sorted by quadrant of the schematic.

8.3.1 1A: Power and CAN connector

The connector PWR_CAN_OUT1 is the exit point of the board, from which 48V and CAN bus are sent upwards the arm towards the shoulder and hand control units. As the torso unit represents one of the end points of the CAN bus, the terminator resistor is connected.

8.3.2 3A: Low voltage jumpers and debug

There are two jumpers connecting the 3.3V and 5V outputs to the rest of the board. This allows for the voltage to be tuned via the potentiometers and inspected via the debug header LOW_VLT_DGB1 before other units are connected to the low voltage supplies.

8.3.3 1B: 20 pin ribbon cable connector

The 20 pin connector for the ribbon cable between the torso and bogie. The pinout is summarised in table 5.

8.3.4 4B: Pins pulled low

All unused pins on the MCU are pulled to ground by a $40k\Omega$ resistor. Opposed to leaving them floating, pulling them low may prevent interference to other pins. The resistor restricts current flow in the event of a surge, which may help prevent damage to the MCU. ST's AN4206 chapter 5.5[27] recommends pulling up or down unused pins, and the value is based on best practice considerations[7].

Pin number	Signal
1	MTR1_OUT1: One of two PWM signals to the rail motor.
2	MTR1_OUT2: One of two PWM signals to the rail motor.
3..7	+48V: System power supply, in.
8..10	GND: System power supply, out.
11	GND: Signal ground reference.
12	UART_RX: UART receive, for external communication.
13	UART_TX: UART transmit, for external communication.
14	ENC0_A: One of two encoder signals from the rail motor.
15	ENC0_B: One of two encoder signals from the rail motor.
16	+5V: 5V power supply to end switch sensor in bogie.
17	DP: USB DP line.
18	DM: USB DM line.
19	VBUS: USB VBUS line.
20	END_SWITCH: End switch signal from bogie.

Table 5: Torso 20 pin connector pinout

8.3.5 1C: UART

UART was included in the external communication as a safe and simple alternative to USB. It would be useful before USB was implemented, and as a backup implementation of **RC3.2: External communication** (4.4.5) should the USB implementation have failed. The header is in parallel with the 20 pin connector, so only one of the two access points may be used at any given time.

8.3.6 5C: Mounting holes

Copying the design of the original unit, none of the mounting holes of the refurbished torso control unit are grounded to the arm chassis. This was somewhat unexpected as grounding the chassis seems to be common practice in all electrical systems design. See section 8.7 for a possible explanation.

8.3.7 3D: Motor assembly debug header

This header is a 1x6 2.54mm pitch header intended for measuring the PWM signals from the MCU to the drivers, and the DRVn_IPROPI voltage from the drivers to the MCU's ADC.

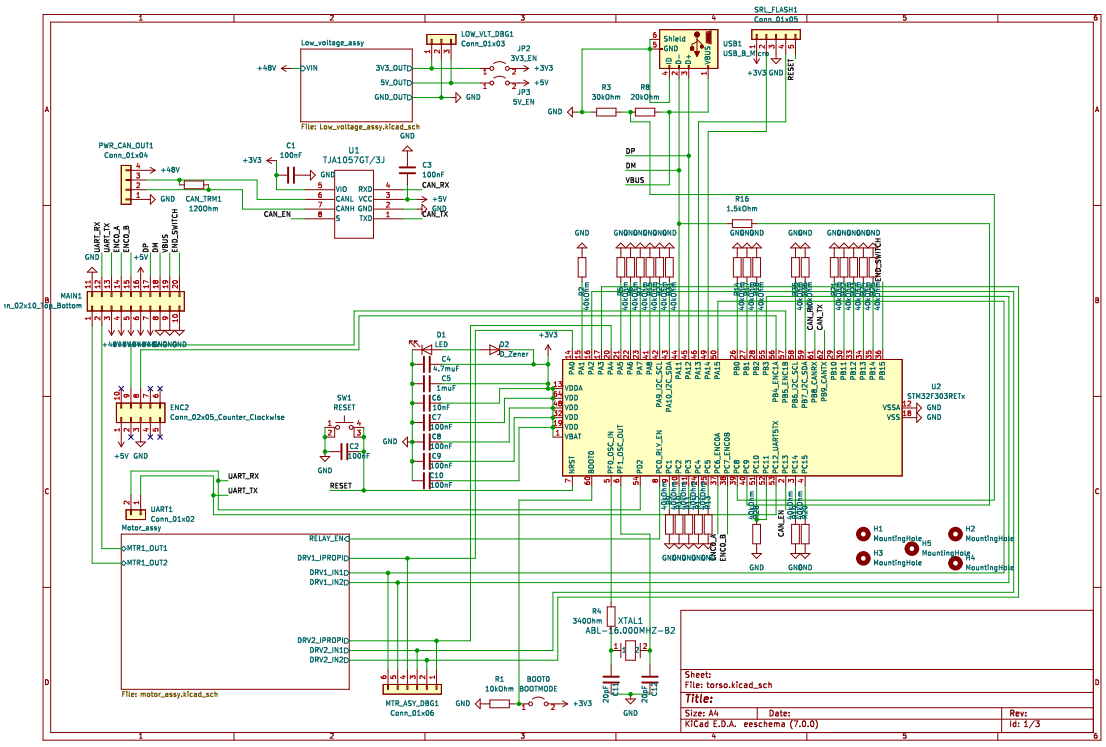


Figure 21: The full circuit design of the torso control unit

8.4 Shoulder

The torso control unit, circuit schematic in figure 22, makes use of all the assemblies previously mentioned, except for the IMU. The following subsections discuss design features unique to the torso design, sorted by quadrant of the schematic.

8.4.1 1A: Power and CAN connectors

The connector PWR_CAN_IN1 is the entry point of the board, from which 48V and CAN bus are received from the torso control unit. The CAN terminator has been included in the schematic for testing purposes, but as the shoulder does not represent an end point of the CAN bus, the corresponding PCB footprint would be left empty before installation in the arm.

The connector PWR_CAN_OUT1 is the exit point of the board, to which 48V and CAN bus are sent upwards the arm towards the hand control unit. It is in parallel with the IN connector.

8.4.2 3A: Low voltage debug and jumpers

See section 8.3.2.

8.4.3 4A: USB assembly

The USB assembly was included in the shoulder control unit for testing and development purposes, and has no association with **RC3.2: External communication** (4.4.5). It is only connected to the shoulder MCU, and will not be available when the board is installed in the arm.

8.4.4 5A: IMU interrupt jumper

The shoulder unit interfaces with the upper arm IMU. When connected, this jumper grounds the MCU interrupt input pin PC10 and the interrupt output from the upper arm IMU board. When unconnected, the interrupt signal is passed to PC10. Its intended use is for it to be connected only when the IMU interrupt is not in use.

8.4.5 5A: IMU connector

This connector takes in 5V and ground reference, and the I2C bus between the upper arm IMU and shoulder MCU, and sends the same to the upper arm IMU.

8.4.6 4B: Pins pulled low

See section 8.3.4.

8.4.7 1C: UART

See section 8.3.5. It was included for testing and development purposes, and has no association with **RC3.2: External communication** (4.4.5). It is only connected to the shoulder MCU, and will not be available when the board is installed in the arm.

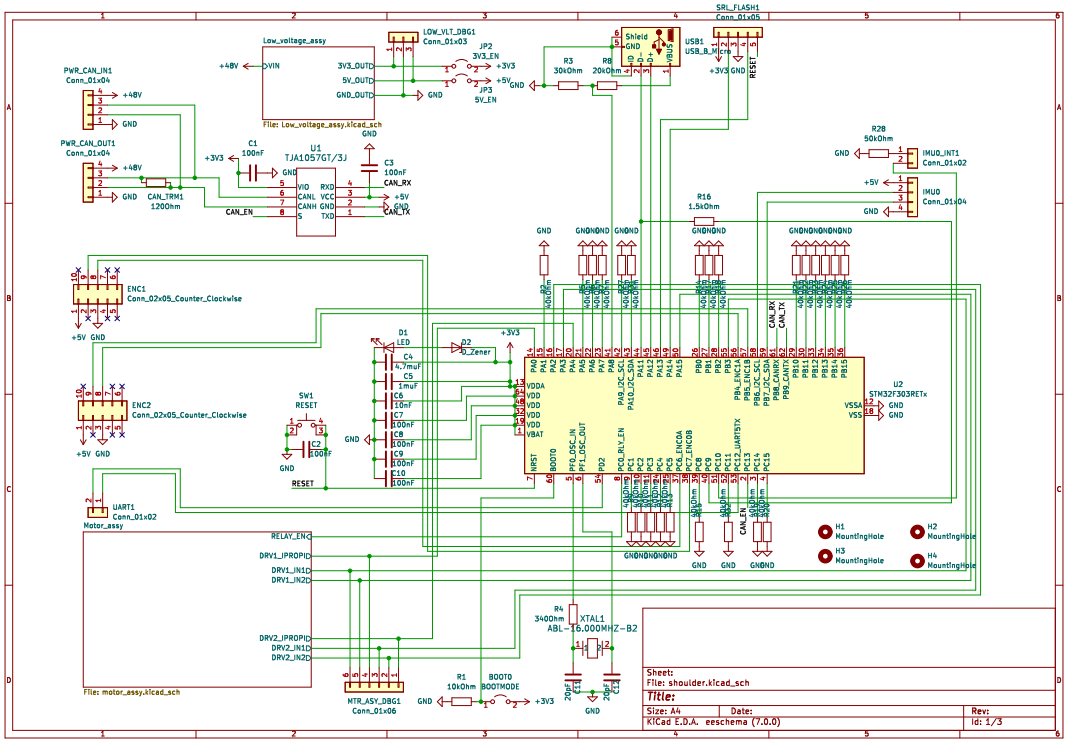


Figure 22: The full circuit design of the shoulder control unit

8.4.8 5C: Mounting holes

See section 8.3.6.

8.4.9 3D: Motor assembly debug header

See section 8.3.7.

8.5 Hand

There is significantly less space in the hand compared to the other two control unit hardpoints. To overcome this, the original control unit makes use of two physical boards (see 35) connected by a ribbon cable. While most of the components of the refurbished design could fit on one board ("hand A"), the twist optocoupler had to remain on the other ("hand B") for mechanical reasons detailed in section 9.7. However, all components in the hand are represented in figure 24. The hand makes use of all assemblies previously mentioned except for the USB and encoder connector assemblies. The following subsections discuss design features unique to the hand design, sorted by quadrant of the schematic.

8.5.1 1A: Power, CAN and IMU connector

The connector named PWR_CAN_IMU1 serves as the board's entry point for 48V and CAN bus from the shoulder and torso, as well as the connector for the lower arm IMU board with its I2C bus, 5V supply and ground reference. As the hand is one of the end points of the CAN bus, the termination resistor is connected.

8.5.2 3A: Low voltage jumpers

See section 8.3.2. Note that the debug header has been removed to save space.

8.5.3 3A: Optocoupler and A/B connectors

The optocoupler is an OPB971N51, and the circuit is based on the block diagram in its datasheet[36]. As it is the same unit as in the original design, the resistor values of R2 and R7, $4.7k\Omega$ and $40k\Omega$ respectively, were selected based on measurements on the original unit. When the input diode is triggered, the voltage over R7 should rise to 3.3V. The signal is sent to the CON_B1 connector, which is connected to CON_A1 and then to the MCU's PA1 pin.

8.5.4 4A: UART and flash connector

As with the shoulder, UART was included for testing and development purposes, and has no association with **RC3.2: External communication**(4.4.5). To save space, the UART was combined with the programming header in SRL_FLS_URT1.

The programming pins are described in table 6.

Pin number	Signal
1	3.3V reference
3	Clock
5	Ground reference
7	Data
9	Reset

Table 6: Pin description of the hand programming header

8.5.5 5A: IMU interrupt and wrist optocoupler jumper

See section 8.4.4. As the hand control unit interfaces with two IMUs, this jumper accomodates the interrupts from both. The hand control unit is also responsible for registering the wrist optocoupler signal. This signal may be connected to pin 3 instead of the lower arm IMU interrupt, see section 8.6.

8.5.6 1B: Encoder connectors

The hand encoders do not use the same 2x5 ribbon wire connector as the other units, but instead use the male pins on the enocders slotted directly into the board. To accomodate this, the standard encoder connector was modified to a 1x5 row. The electrical interface remains the same: The encoder is supplied with 5V, and the two encoder signals are passed to the MCU's encoder counter peripheral.

8.5.7 4B: Pins pulled low

See section 8.3.4. To reduce board space usage, unconnected pins next to one another on the MCU were all connected to the same resistor.

8.5.8 2C: Reset

In the hand's reset circuit, the pushbutton has been replaced with a 1x2 1.27 mm header to save space.

8.5.9 5C: Mounting holes

See section 8.3.6.

8.5.10 1D: Motor assembly

Some adjustsments had to be made to the motor assembly in order for it to fit on the hand. The two motor connectors were merged to a single 2x4 2.54mm pin header, the relay and control MOSFET were removed, and the bulk capacitors were split in three units in parallel for each driver. The schematic is presented in figure 23.

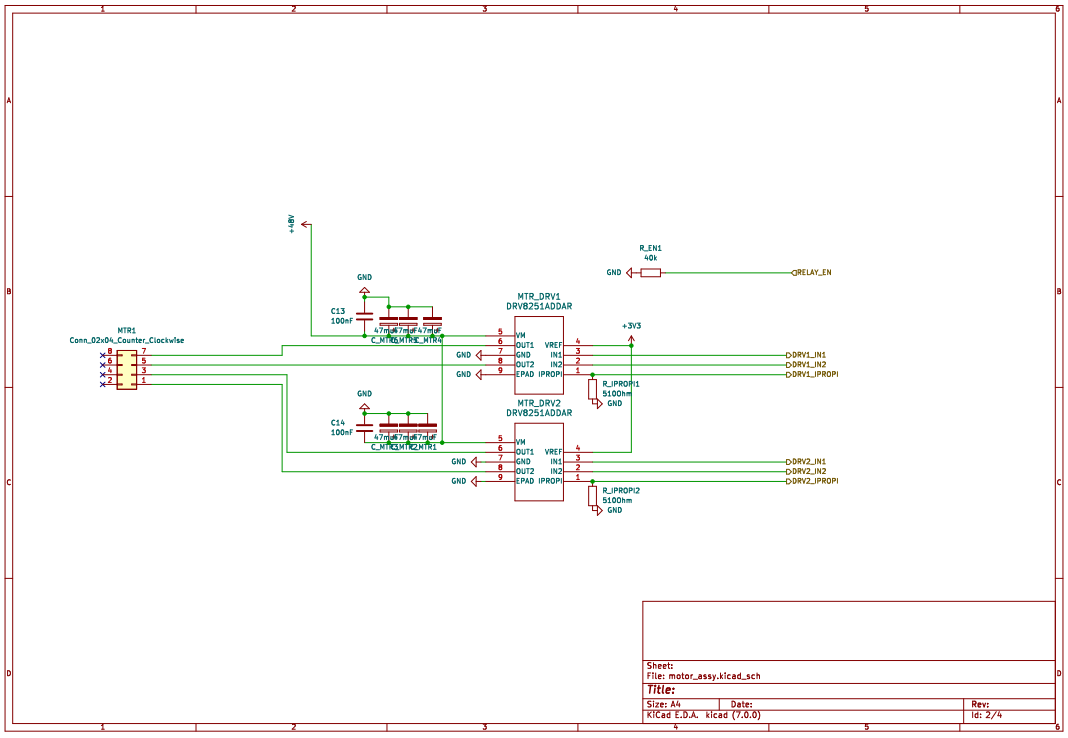


Figure 23: The modified schematic for the hand motor assembly

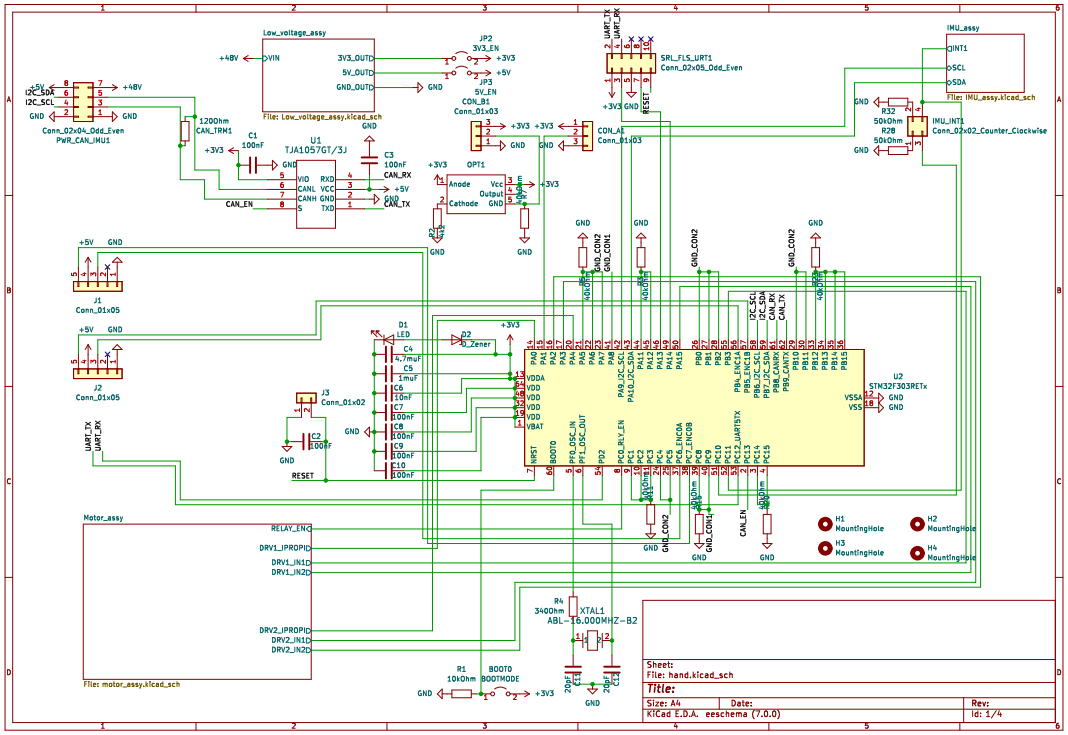


Figure 24: The full circuit design of the hand control unit

8.6 IMU boards

The IMU circuit uses only the IMU assembly, illustrated in figure 25. The same circuit is used in both the upper and lower arm IMU PCBs. The 1x5 2.54mm pin header takes in 5V, ground and the I2C bus from the associated control unit, and outputs I2C to the IMU, the interrupt signal INT1 to the associated MCU, and 5V to the voltage divider supplying the IMU and, optionally, the 1x3 2.54mm pin header. In the lower arm, the 1x3 2.54mm pin header interfaces with the wrist optocoupler, assumed to use a 5V supply. In the upper arm, this header has no function.

The jumper OPT_EN1 may be connected in order to supply the wrist optocoupler. The jumper INT_EN1 may be connected in one of two positions in order to send either the wrist optocoupler signal or the IMU interrupt signal to the associated MCU. For the lower arm, this means that only one of the IMU interrupt or wrist optocoupler may be in use at a given time.

The voltage divider steps the 5V supply voltage down to 3.13V to supply the IMU assembly.

Copying the design of the original boards mounted in the arm sections, the mounting holes are grounded.

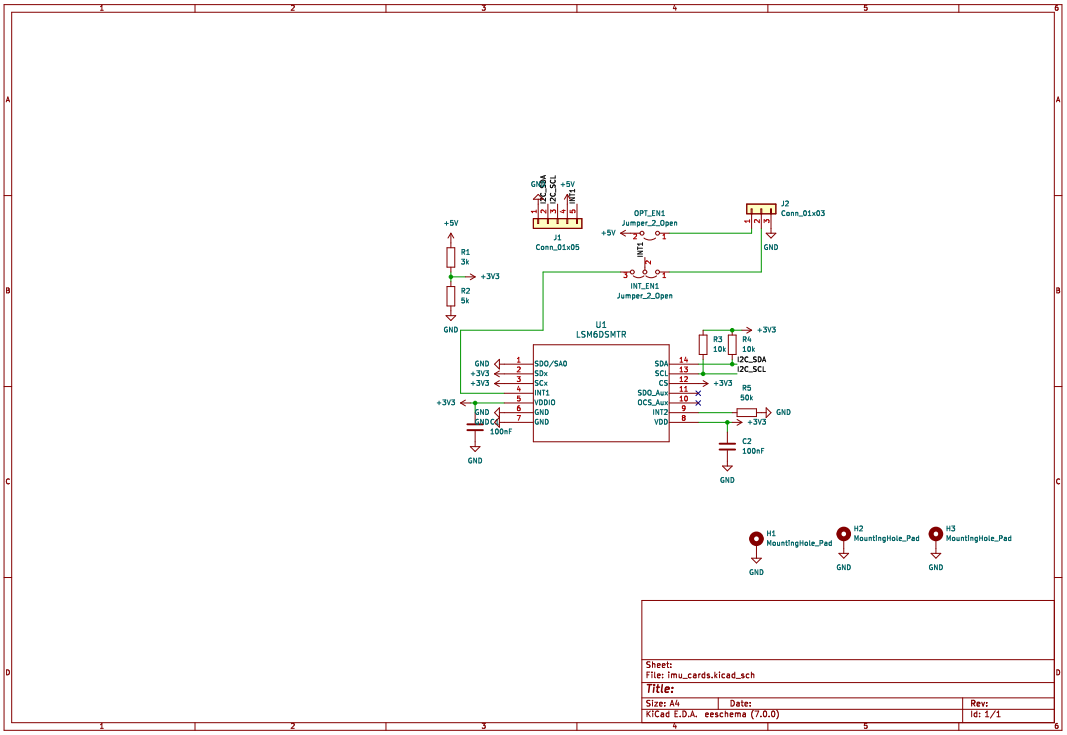


Figure 25: The circuit design of an IMU board

8.7 Auxiliary boards

The bogie and rail connector boards use one encoder connector assembly and were drawn in the same schematic, see figure 26. Together, they are referred to as auxiliary boards.

8.7.1 Bogie board

The bogie board consists of the 20 pin ribbon cable connector interfacing with the torso control unit MAIN1, the rail motor connector MTR1, the rail motor encoder connector ENC0, the end switch connector END_SWITCH1, and the 16 pin connector interfacing the bogie board with the rail board OUT1.

The pinout of the 20 pin connector is identical to that of the torso, see section 8.3.3, except that the order of the signals on the second row (pin 11-20) has been reversed. This error is further discussed in section 13.

The end switch connector takes in 5V and ground for supply, and sends the end switch signal back to the 20 pin connector.

The mounting holes are not grounded.

The 16 pin connector is the entry point to the bogie. It takes in 48V and system ground, as well as the UART and USB signal lines from the rail board. It outputs a 5V line to supply the TVS array on the rail board. The pinout is summarised in table 7.

Pin number	**Signal**
1	Ground signal reference
2	USB DP
3	USB DM
4	USB VBUS
5	UART RX
6	UART TX
7	Ground signal reference
8	5V supply
9..11	Voltage supply ground
12..16	48V

Table 7: Bogie 16 pin connector pinout

8.7.2 Rail board

The rail board consists of a 2x7 2.54mm pin header interfacing with the rail's DSUB15 connector DSUB1, the 16 pin connector interfacing the rail board with the bogie OUT2, the TVS array protecting the UART and USB lines from surges, and an indicator LED.

The pinout of OUT2 is identical to that of the bogie, except that ground has been renamed from GND to EARTH, and 48V to HI_V to prevent KiCad from insisting that there be a track between those nodes in the PCB layout.

The circuit for the TVS arrays is based on the the schematic on the first page of the unit's datasheet[6]. It is in parallel with the data lines of UART and USB between the two connectors OUT2 and DSUB1, and has an indicator LED in series with its ground port. The pinout is summarised in table 8.

The 2x7 2.54mm pin header DSUB1 connects to a ribbon cable which is soldered to the original DSUB15 connector on the rail, and its pinout is summarised in table 9.

The mounting holes for the rail board are grounded. This board is connected to every other board,

Pin number	Signal
1	USB DP
2	Ground
3	USB DM
4	UART RX
5	5V
6	UART TX

Table 8: TVS diode pinout

Pin number	Signal
1	Signal ground
3	USB VBUS
5	UART RX
7	UART TX
9,11,13	Voltage supply ground
2	USB DP
4	USB DM
6,8	Voltage supply ground
10,12,14	48V

Table 9: Pinout of the 14 pin connector

uninterrupted, by the main bus ground line. The control boards are therefore grounded to the arm chassis via the rail board. It is assumed that this is done to prevent ground loops between the arm chassis and the main bus ground line.

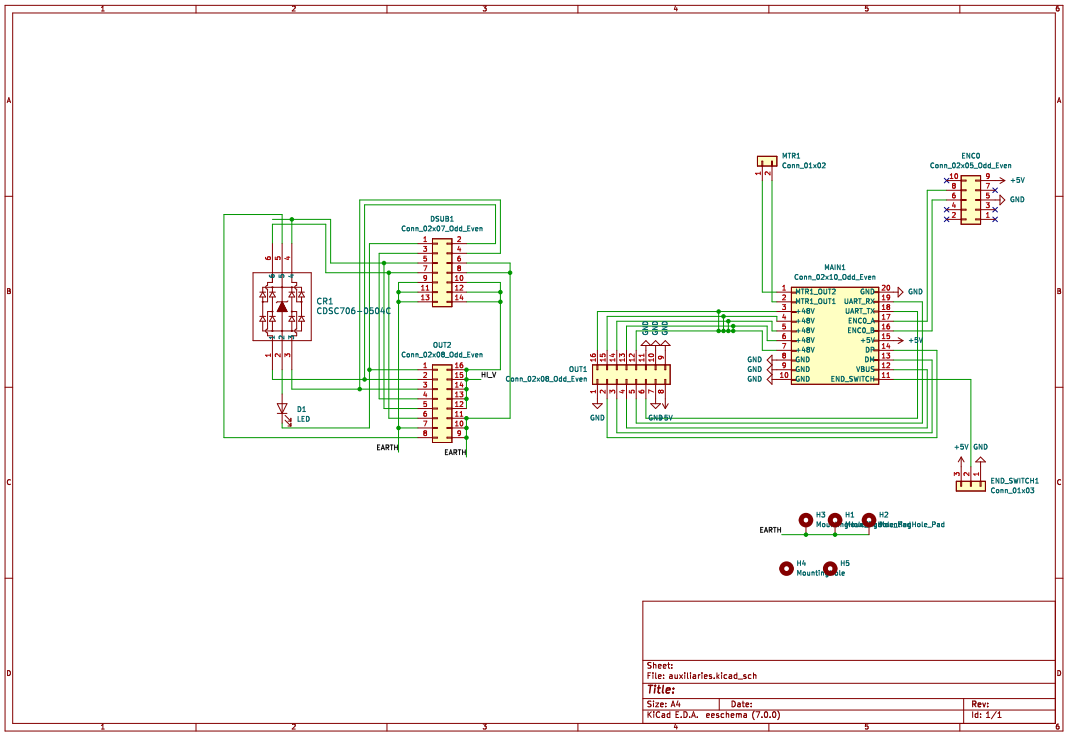


Figure 26: The circuit design of the bogie and rail boards, also known as auxiliary boards

9 PCB design

9.1 Tools

9.1.1 Schematic and layout: KiCad

The software package KiCad:PCB Editor was used for the PCB layout design. The calculator tool KiCad:Calculator Tools was used for USB impedance matching.

9.2 Design process and rules

All PCBs were designed from the corresponding circuits. While KiCad has a wide selection of footprints available for generic components, some of the ICs used in the project were not immediately available. In those instances, the footprints were downloaded from the UltraLibrarian website[12]. The relevant footprints were found by searching for the IC's model number, and downloading a version compatible with KiCad6. Were the footprints not available from UltraLibrarian, they were drawn manually based on the component's datasheet. As for the presentation in the following sections, it was challenging to export the layouts in a readable format, particularly the measurements for mount points. In the relevant figures, non-plated (i.e. not grounded) mount points are barely visible as a black circle intersecting the measurement lines.

The components for which footprints had to be drawn manually were:

- Motor control relay JV-3S-KT, and
- Optocoupler OPB971N51

9.2.1 Layering

All boards except for the IMU boards were designed with 4 layers of copper. This relaxed constraints on trace layout compared to using 2 layers. The layers were used as follows:

1. Front copper: Primary signal layer, filled with ground otherwise.
2. Internal copper 1: Ground layer.
3. Internal copper 2: 3.3V layer.
4. Back copper: Secondary signal layer, filled with 5V otherwise.

Unless otherwise noted, the thickness and material type of the PCB were the defaults of JLCPCB[11]. Trace to fill clearance was set to 0.3 mm unless the board had a USB interface. In that case, the trace to fill clearance was 0.6 mm to reduce the potential for noise on the USB data lines.

9.2.2 Via stitching

The boards effectively have two ground planes in the top two layers. As the layouts of the refurbished control boards are more compact than the original, but the boards are of the same size, this leaves much empty space. In order to ensure an even potential of 0V between the layers all over the boards, vias have been placed in matrix patterns in sections with few components.

9.2.3 Constraints

Constraints for trace width, via sizes, fill zones etc were largely dictated by JLCPCB's manufacturing capabilities[11]. Most notably, they do not offer blind or buried vias. Component placement were in some instances constrained by **RM1: PCB Form factors** (4.2.1), and all boards were constrained by **RM1.1: PCB outlines** (4.2.2) and **RM1.2: Mount points** (4.2.3). Mount point dimensions were measured with a caliper on the original board before design of the electrical layout was started.

Generally, components interfacing with parts of the orginial system were placed approximately where they used to be. These parts are shaped, oriented or otherwise optimised for the original system in such a way that placing the components differently would be impractical. This often constrained the placement of the remaining components as well, and was a primary challenge when designing the layouts.

9.2.4 THT vs SMD, footprint sizes

The primary deciding factor for using either a through-hole (THT) or surface mounted (SMD) version of a given component, when all other constraining factors were accounted for, was its availability at Omega Verksted or the preferred online vendor. Preference was given to SMD on the assumption that this would save production time, if either was an option. The practicality of mounting very small footprints, such as 0402, by hand was of some concern, and larger footprint sizes were preferred. Note that where stated, footprint sizes are given in imperial units. Most footprints are not discussed in this section, and **a full bill of materials can be found in the appendix**.

9.3 About the presentation of PCB layouts

As with the circuit designs, the PCB layouts are similar enough that treating them all at the same level of detail would result in much double work. On the other hand, presenting each subassembly before discussing the unique features of each board would also be impractical as it is much harder to tell which parts belong where in a PCB layout than on a schematic. As a compromise, the torso unit is treated in detail, and the other control units less so. Generally applicable considerations like trace widths, noise reduction measures and thermal dissipation can be assumed to be the same across all boards unless otherwise noted.

9.4 About mount point presentation

Each PCB layout has a subsection dedicated to board outline and mount points. The sections consist of a table and a measurements figure, and these should be read in conjunction. The mount point designators follow two patterns, Hn and MPn. H designators are bolt holes drilled in the PCB, assigned by KiCad and visible in the "front copper" figure of the corresponding PCB. MP designators are mount points accessed via the LM317's mounting hole. As there are no absolute coordinates for the mount points, the "reference" column of the table indicate which measurement line to look for in the measurements figure when identifying the mount point.

9.5 Torso

9.5.1 Outline and mount points

The torso has 8 mount points. Three connect the PCB to a metal plate inside the torso, two connect a metal bar heat sink to the PCB, and three connect the motor drivers and 5V power supply to the heat sink. See figure 2 for a visual reference. Per **RM1.2: PCB mount points** (4.2.3), 7 mount points were utilised, and 1 ignored. The ignored mount point is the leftmost of the three on the heat sink, and corresponds to the measurement line of 32.5mm in the upper left of figure 27. The two remaining mount points on the heat sink were used for the voltage regulators, and correspond to the measurement lines of 55mm and 71.5mm. The **board outline is 140x84mm**.

Mount point	Mode	Footprint	Reference
H1: Lower left	Utilised	2.7 mm	17 mm
H2: Lower right	Utilised	2.7 mm	17.45 mm
H3: Heat sink right	Utilised	2.7 mm	84.25 mm
H4: Heat sink left	Utilised	2.7 mm	14.75 mm
H5: Upper right	Utilised	2.7 mm	17.45 mm
MP6	Ignored	N/A	32.5 mm
MP7: Step down	Utilised	LM317	71.5 mm
MP8: Step down	Utilised	LM317	84.25 mm

Table 10: Table of torso hardpoint mount points

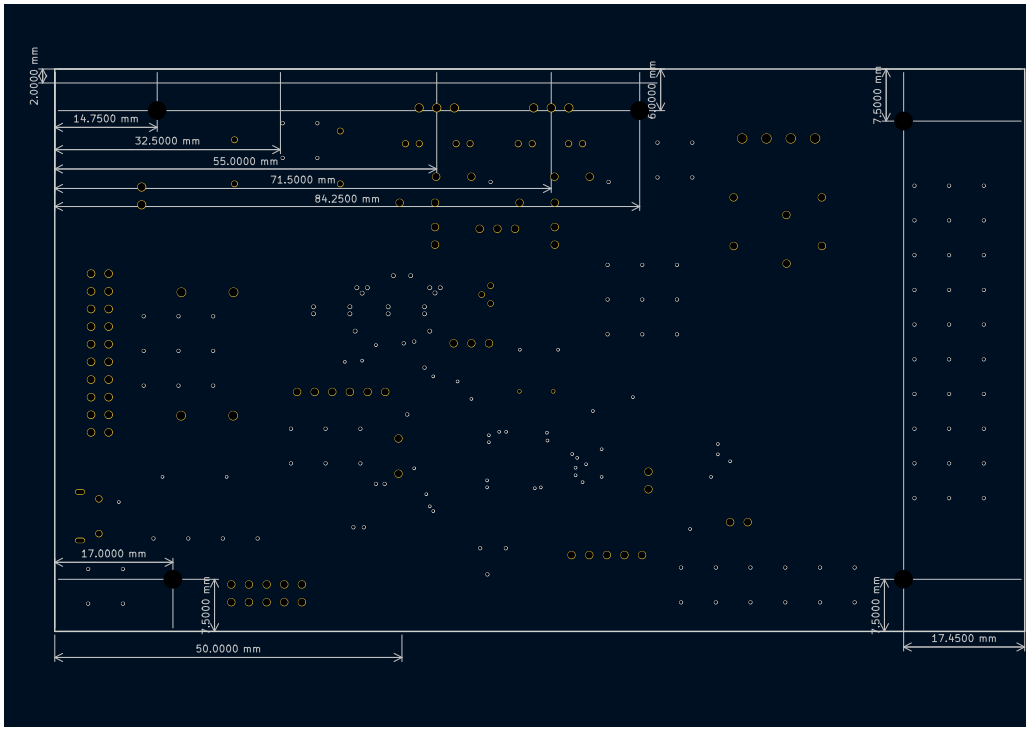


Figure 27: Measurements of outline and mount points of the torso

9.5.2 Front copper and silk

The torso front copper and silk are presented in figure 29.

20 pin ribbon cable connector MAIN1: Placed to the left on the board to match the orginal placement, as the ribbon cable is oriented in such a way that placing it anywhere else is impractical. The 48V trace supplying the rest of the arm can be seen along the upper edge of the board. The trace width of 3mm was chosen to be similar to that of the original control unit, and because this trace will carry a significantly higher current than any other in the system.

USB connector USB1 and interface, impedance matching: The connector was placed below the 20 pin connector in order to be available even while the board is installed in the arm (but before the torso lid is installed). The VBUS, DP and DM traces can be seen going up towards the 20 pin connector and right towards the MCU. The DP and DM traces were kept on the front copper, and with an even distance of 0.3 mm in an attempt at impedance matching and noise reduction. Additionally, the recommended maximum trace length for USB data traces is 8 inches (≈200 mm)[10]. The total length of the data traces is approximately 95 mm between the MCU and 20 pin connector. The three resistors associated with the USB assembly can be seen as a cluster below the MCU. The relatively large footprint sizes of 0805 and 1206 were chosen for easy manual installation.

Some consideration was given to impedance matching of the USB traces. This affected the choice of materials and layer thicknesses for PCB production. Chapter 7 of the USB2.0 specification defines a characteristic impedance of 90Ω[37]. Figure 28 illustrates the calculations, and table 11 summarise the final values. H was arbitrarily set to 0.1 mm, and parameters W and S were tuned until a differential impedance of 90.2Ω was reached.

Parameter	Value	Reason
εr	4.5	FR4 property
$tan\delta$	0.02	FR4 property
ρ	1.72e-08	Copper property
H	0.1 mm	Experimentally set
T	0.035 mm	JLCPCB standard
W	0.2 mm	Experimentally set
S	0.25 mm	Experimentally set
L	100 mm	Approximate trace length
Differential Impedance	90.1579 Ω	Result

Table 11: USB impedance matching parameters

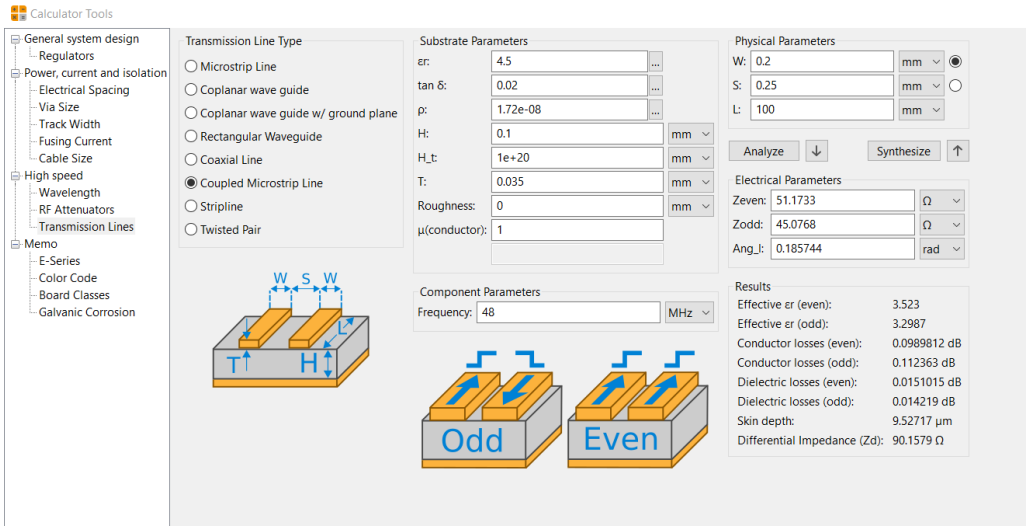


Figure 28: The calculator used to find suitable parameters for USB data traces

Encoder connector header ENC2: Placed next to H1 to match the original placement, as the ribbon cable is oriented in such a way that placing it anywhere else on the board is impractical.

Motor connector socket MTR2: Placed above MAIN1 to match the original placement, as the length and placement of the motor cable is such that placing it anywhere else is impractical.

Motor bulk capacitors and relay: Capacitors were placed next to MAIN1 in part because they are physically large at approximately 20mm height and may not have fit under the shoulder motor if they were placed further in on the board, and in part because it is considered best practice to place capacitors close to their users. In this case, the capacitors are about 20mm in trace length away from the motor drivers (back copper). A trace width of 2 mm was chosen to supply the two motor drivers, sent through the relay visible above the motor capacitors. This width was somewhat arbitrarily chosen: in part because it is similar to the original design, and in part because 2 mm was assumed to be sufficient to supply the rail and shoulder motors at peak capacity.

Motor drivers, relay control and debug headers: The motor drivers and relay control transistor are placed to the right of the bulk capacitors, marked MTR_DRV1, MTR_DRV2 and RLY_CTRL1 respectively. Their debug headers MTR_ASY_DBG1 and RLY_CTL_DBG1 are just below them. The placement of the drivers were opposingly constrained by MTR2 and MAIN1, and the MCU. The drivers should be as close as possible to the connectors to reduce power dissipation, and because shorter traces would reduce the potential for the $\leq$48V 25kHz PWM signal to induce noise in other components close to the trace. Opposing this, the drivers should be close to the MCU and connected by straight traces that pass few vias in order to reduce PWM signal attenuation and noise induction along the traces[1]. As the two PWM signal ports sit on opposing sides of the MCU, some of these constraints must be violated. The PWM traces between MTR_DRV1 and the MCU are mostly on the back copper, but bend around the MCU directly underneath the crystal. This may be suboptimal with regards to noise, but is probably not critical as they are separated by two layers of copper. The corresponding traces for MTR_DRV2 can be seen going diagonally from the driver to the MCU on the front copper.

The underside of the motor drivers have a thermal dissipation pad which must be connected to ground. In addition to being connected to the top ground layer, 4 vias with a diameter of 0.5 mm connect each thermal pad to the middle ground layer to dissipate heat effectively.

The trace width from the drivers to the motor connectors is set to 1 mm, the assumption being that they will need about half the capacity of the supply trace each. As with the PWM signal traces, DRV2 is on the front copper while DRV1 is on the back.

The silk screen notations of the debug headers are summarised in table 12. **Note** that the ground pin of the relay debug header may safely be used as a reference for measurements on the motor assembly debug header.

Motor debug	Meaning	Relay debug	Meaning
2I	V_{IPROPI}	E	Enable
22	DRV2 PWM2	G	Ground
21	DRV2 PWM1	+	3.3V
1I	V_{IPROPI}		
12	DRV1 PWM2		
11	DRV1 PWM1		

Table 12: Explanation of motor drivers and relay control debug headers

Voltage regulators and CAN bus connector: The low voltage assembly and CAN bus connector fill most of the upper right quadrant of the board. The placement of these assemblies were largely dictated by the mount points MP7 and MP8 for the voltage regulators, seen as U4 and U3 on the silk, respectively. The trace width of 1 mm used on the main parts of the circuit was chosen out of an abundance of caution, and because these assemblies were not particularly constrained for space. The indicator LEDs D8 and D7 are visible on the middle right of each unit. To separate them, red was chosen for 5V and green for 3.3V. The jumpers JP3 and JP2 connect 5V and 3.3V to the respective layers, respectively, and **must not be connected** before RV1 and RV2 have

been tuned and the correct voltage has been read out on the LOW_VLT_DBG1 header.

The placement of the voltage adjustment potentiometers RV1 and RV2 was constrained by the practicalities of using a flat iron in close proximity to sensitive electronics. When facing outwards with no other components between them and the board's edge, they may be tuned efficiently and with minimal risk to nearby components. Having them close to the edge makes it easier to find a compatible flat iron. RV1 adjusts the 3.3V regulator, RV2 5V. The placement of the potmeters over the CAN bus traces was deemed to have no significant consequence with regards to noise, as the regulators' adjustment current I_{ADJ} is at most $100\mu A$[35] at an approximately constant voltage.

The connector PWR_CAN_OUT1 which sends power and the CAN bus to the other parts of the arm was placed to match the placement on the original unit, and supplementally to reduce the length of the 48V trace compared to having it further towards the vertical middle of the board. Placing it further left is not an option, as the bus cables may not fit under the shoulder motor. The connector silk is explained in table 13.

Low voltage debug	Meaning	CAN, PWR connector	Meaning
Left	5V	G	GND
Middle	GND	H	CAN High
Right	3.3V	L	CAN Low
		+	48V

Table 13: Explanation of low voltage debug and CAN/power bus connector

CAN transciever: The can transciever is placed to the right of the MCU, designated U1, its position decided by the position of the MCU's CAN port. The tranciever power pins were connected to the power planes by vias placed under the tranciever IC. The CAN terminator resistor CAN_TRM1 was given a 0603 footprint to fit between the signal pads on the transciever while also be large enough to be easily placed by hand.

Flash, UART and BOOT0 headers: Placement was decided by the position of the relevant ports on the MCU, and there were few conflicts. All three units can be seen below the CAN tranciever, to the right of the MCU. The programming header SRL_FLASH1 was placed close to the MCU to minimise risk of signal attenuation in long traces during programming. As a compromise, the reset trace was drawn under the MCU. While it may have been preferable to draw the trace on the back copper from a noise induction perspective, the space for vias on the top side of the MCU, where the reset signal enters, is limited. It is assumed that the reset signal is not toggled often enough during programming for this to be an issue. The silk symbols are explained in table 14.

Flash header	Meaning
R	Reset
D	Data
G	Ground
C	Clock
+	3.3V

Table 14: Explanation of flash header pins

MCU and crystal: Placement of the MCU was dictated by, and the dictator of, the placement of all other components. It is designated U2, and is placed in the middle towards the bottom of the board. After the components which would interface with the original system part were placed, the MCU was rotated such that as few traces as possible crossed. The remaining components were then placed in a loose ring around the MCU to further minimise trace crossings, thus reducing the need for vias and traces on the back copper. The trace width of most signals entering the MCU is 0.25 mm, chosen as the widest possible traces compatible with the MCU's pads. Where the space between the pads is particularly narrow, such as the bottom pad row, the width was reduced to 0.20 mm. The crystal was placed as close as possible to the MCU without blocking trace paths for the top pad row.

Zener, indicator LED and pull-downs The zener diode D2, large decoupling capacitors C4 and C5, and the surge indicator LED can be seen left of the MCU. As there are power pins all around the MCU, there was no single natural "entry point" on which to place these components, but they were placed as close as possible to the MCU without blocking other signal traces. The pulldown resistors can be seen all around the MCU, footprint size of 0402 chosen to save space. The only constraint on the placement of the pulldowns were that they should not block the traces of other signals while remaining in the vicinity of the MCU.

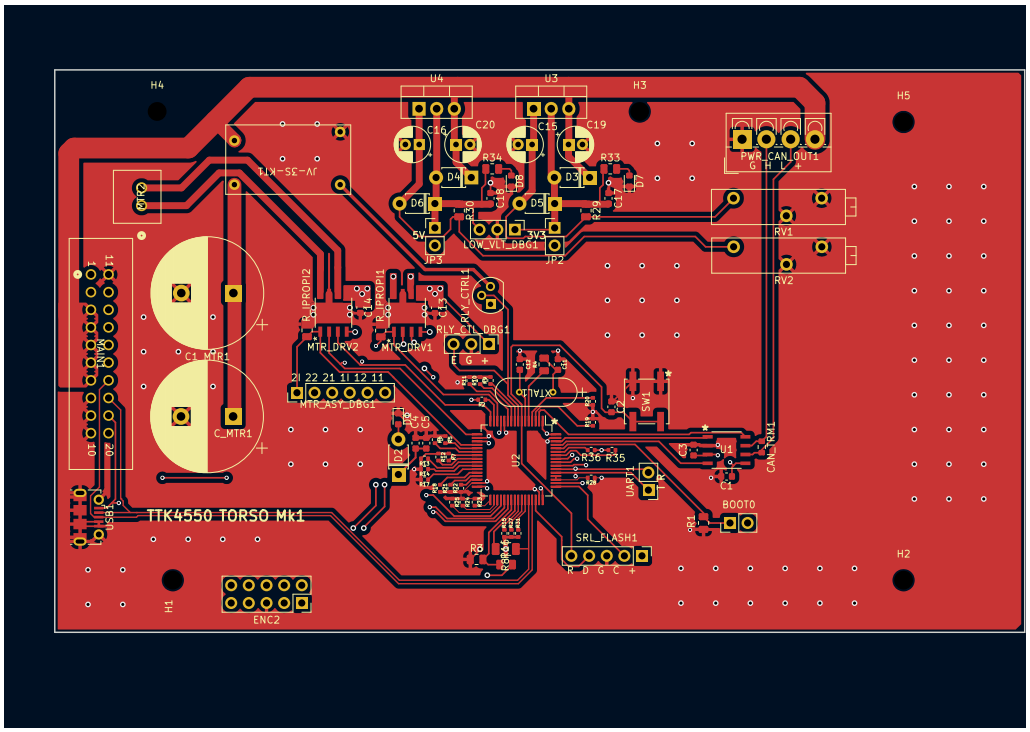


Figure 29: Front copper and silk layers of the torso PCB

9.5.3 Back copper and silk

The torso back copper can be seen in figure 30.

MCU decoupling capacitors: The decoupling capacitors for each of the MCU's power pin pairs were placed directly underneath the pins, as recommended by chapter 5.4 of AN4206[27], vias under the MCU to avoid conflicts with signals on the front copper. Their 0603 footprint was chosen as a compromise between fitting under the power pin pair and being practical to solder by hand.

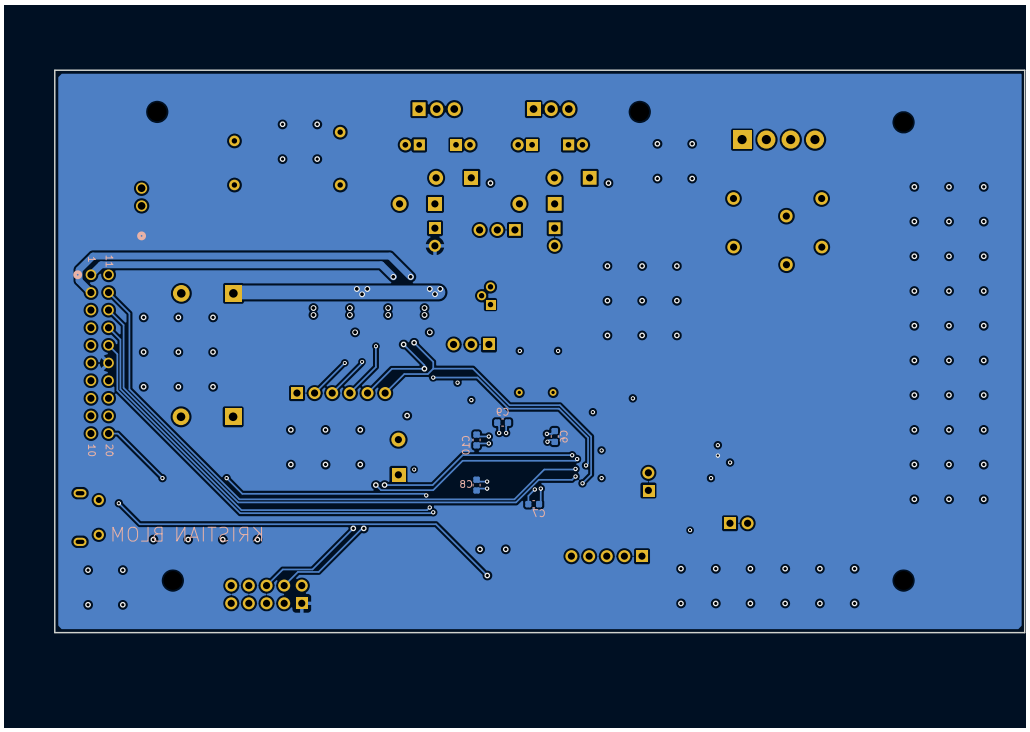


Figure 30: Back copper and silk layers of the torso PCB

9.6 Shoulder

Design considerations discussed for the torso generally apply to the shoulder as well. Header silk symbols not unique to the shoulder can be found in the torso subsections. The board was designed before filling of the front copper and via stiching was introduced as a standard.

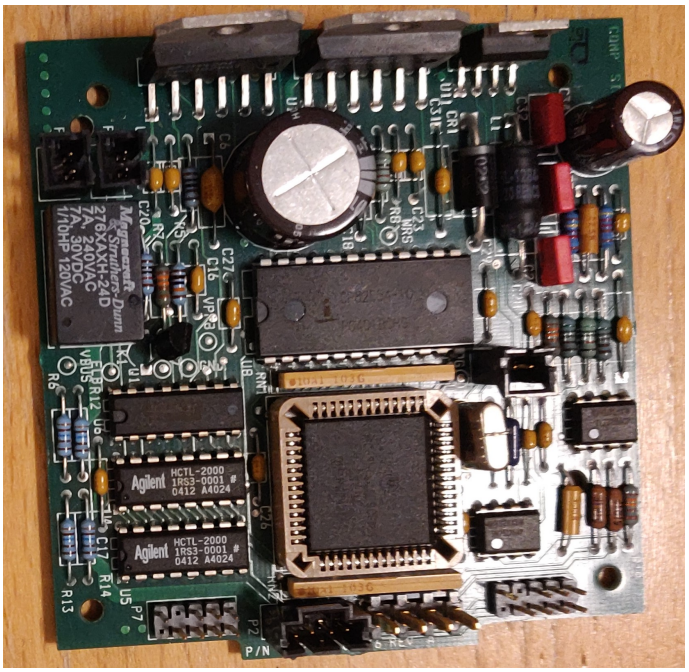


Figure 31: The original shoulder unit, used as a layout reference. Note that the heat sink for the voltage regulators and H-bridges has been removed

9.6.1 Outline and mount points

The shoulder has 7 mount points. Two connect the PCB to the shoulder, two connect a metal bar heat sink to the PCB, and three connect the motor drivers and 5V power supply to the heat sink. See figure 2 for a visual reference of the heat sink, and figure 31 for the remainder of the board. Per **RM1.2: PCB mount points** (4.2.3), 6 mount points were utilised, and 1 ignored. The ignored mount point is the leftmost of the three on the heat sink, and corresponds to the measurement line of 62.5mm in figure 32. The two remaining mount points on the heat sink were used for the voltage regulators, and correspond to the measurement lines of 21mm and 40mm. The **board outline is 88x88mm**.

Mount point	Mode	Footprint	Reference
H1: Lower left	Utilised	2.7 mm	4.15 mm
H2: Lower right	Utilised	2.7 mm	5.0 mm
H3: Heat sink right	Utilised	2.7 mm	10 mm
H4: Heat sink left	Utilised	2.7 mm	10 mm
MP5: Step down	Utilised	LM317	21 mm
MP6: Step down	Utilised	LM317	40 mm
MP7	Ignored	N/A	62.5 mm

Table 15: Table of shoulder hardpoint mount points

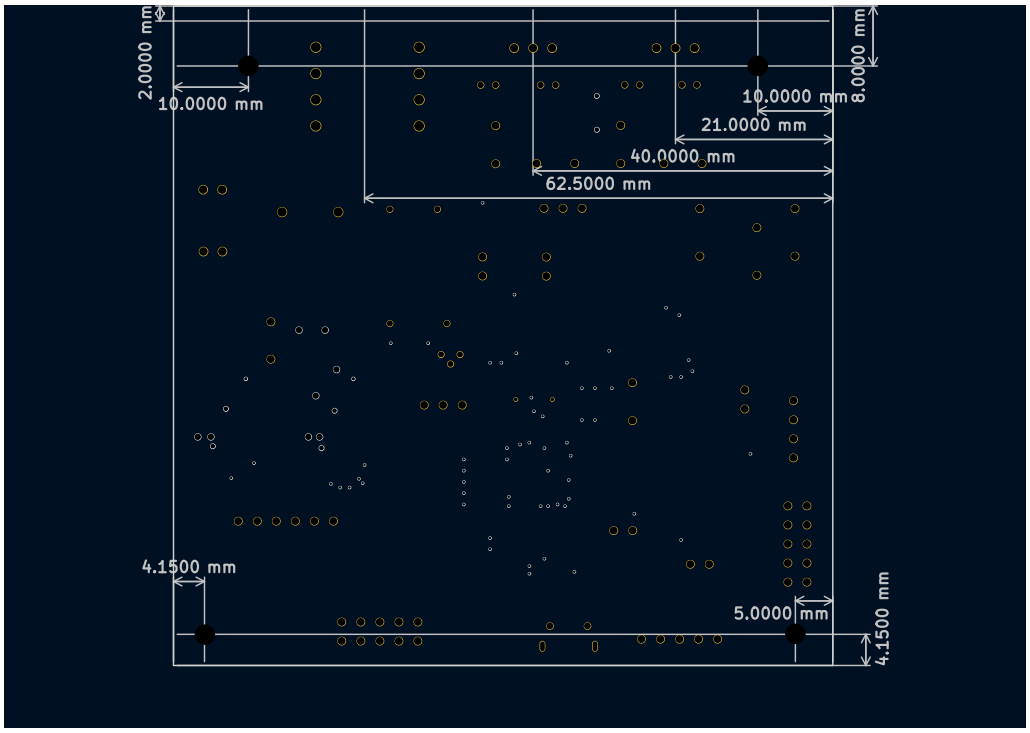


Figure 32: Measurements of outline and mount points of the shoulder

9.6.2 Front copper and silk

The shoulder front copper and silk are presented in figure 33

Motor connector sockets (MTR1, MTR2), encoder connector header 1 (ENC1): Positions were kept from the old design because moving them would have been impractical. Motor connector sockets are on the upper left of the board near the left edge, while ENC1 is on the lower left.

CAN bus and power connectors: Placed on the upper left near the top edge of the board in part to keep the 48V trace short, and in part to make space for the USB and programming headers. They were originally on the bottom edge of the board, as seen in figre 31, but moving them up did not pose an issue.

USB connector, programming header and encoder connector header 2 (ENC2): USB connector USB1 and programming header SRL_FLASH1 were placed near the bottom edge of the board in part to make the interface easily available during testing, and in part because of the position of the relevant pins of the MCU. Moving ENC2 away from its original position near the bottom edge to the lower right edge was not a problem with regards to the encoder connector.

IMU connector header IMU0 and interrupt jumper IMU0_INT1: The IMU header IMU0 was placed above ENC2, primarily because of the position of the relevant pins of the MCU. An explanation of the header silk symbols can be found in table 16. IMU0_INT1 can be found left of ENC2, below the MCU.

IMU header	Meaning
G	Ground
D	Data
C	Clock
+	5V

Table 16: IMU silk description

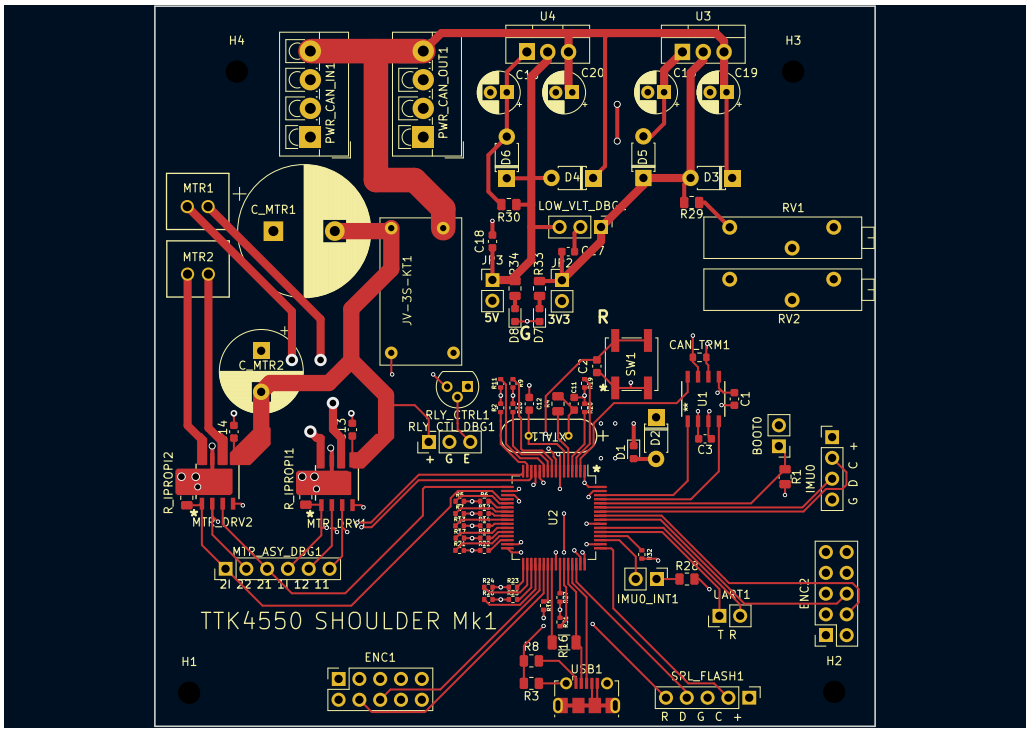


Figure 33: Front copper and silk layers of the shoulder PCB

9.6.3 Back copper and silk

CAN data lines: The CAN data line traces were drawn under the voltage regulator assemblies, an can be seen in the upper right quadrant of the board. The 3V layer is immideately above the lines, which may make them susceptible to noise from that layer. CAN being a differential pair bus, it is assumed that they are sufficiently robust against noise that this will not be an issue.

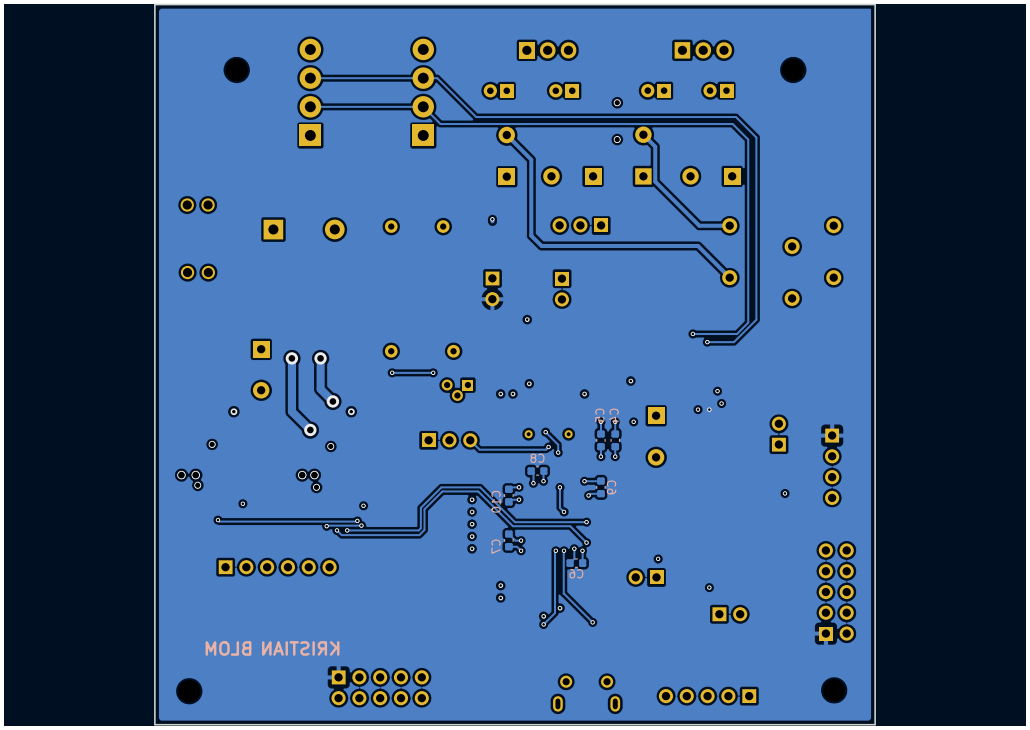


Figure 34: Back copper and silk layers of the shoulder PCB

9.7 Hand

Design considerations discussed for the torso generally apply to the hand as well. Header silk symbols not unique to the hand can be found in the shoulder and torso subsections.

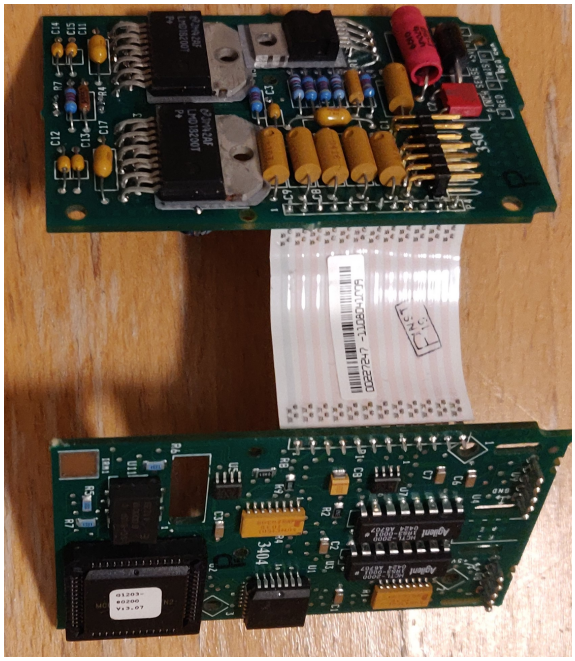


Figure 35: The original hand unit, used as a layout reference.

9.7.1 Outline and mount points

The hand has 5 mount points, 2 on the PCB containing the MCU and other signal processing units (dubbed A), and 3 on the PCB containing the voltage regulator and H-bridges (dubbed B). See figure 35 for a visual reference. Per **RM1.2: PCB mount points** (4.2.3), 4 mount points were utilised, and 1 ignored. The ignored mount point is on board B, coinciding with the board's voltage regulator top hole, and corresponds to the lower blue rectangle in the middle right of board B in figure 36. The **board outline for board A is 81x54mm, board B is 37x54mm**.

Creating the outline for hand board A was significantly more challenging than the other boards, but the process was the same. A caliper was used to take measurements of the holes and lower right mouse bite, and the rectangular shapes were rounded in accordance with the capabilities of JLCPCB[11]. The upper right cutout accomodates a gear which is part of the arm's twist joint assembly (a mechanical system not further discussed in this report), and the four smaller holes near the bottom edge accomodate metal bars of unknown purpose on the encoders, see datasheet for more details[2]. The bottom right mouse bite is necessary to fit the board inside the hand.

Mount point	Mode	Footprint	Reference
H1: Step down	Utilised	2.2 mm	30.475 mm
H2: Step down	Utilised	2.2 mm	2.475 mm
H3: Middle right	Utilised	2.7 mm	9.75 mm
H4: Right	Utilised	2.7 mm	9.75 mm
MP5: Right	Ignored	N/A	Lower blue rectangle

Table 17: Table of hand hardpoint mount points

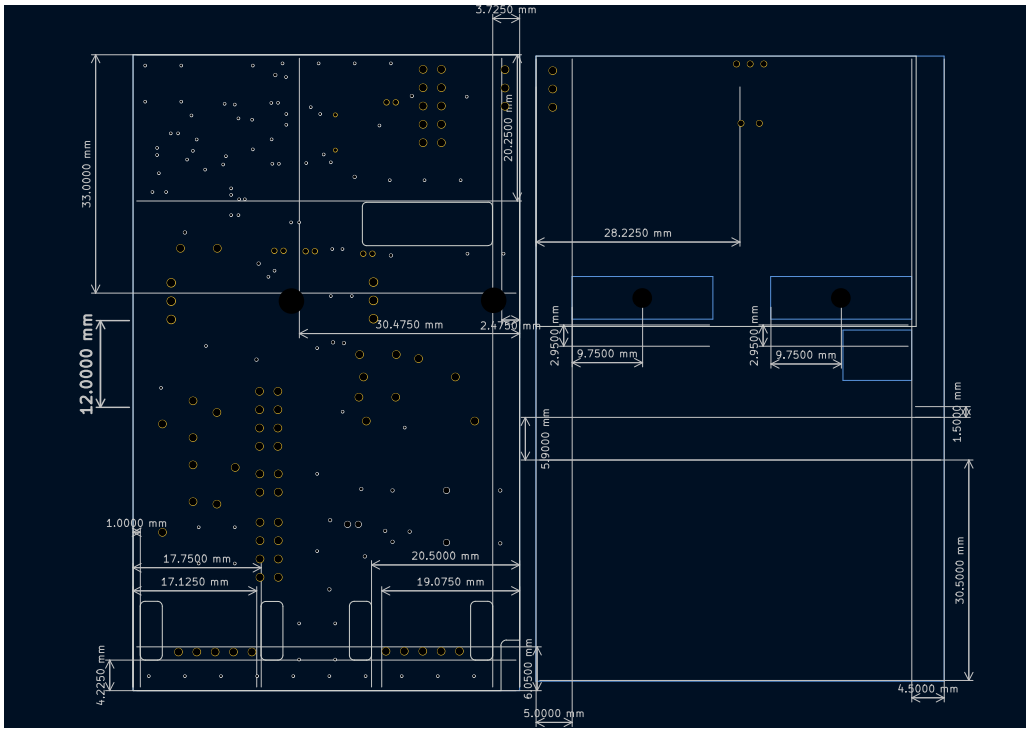


Figure 36: Measurements of outline and mount points of the hand

9.7.2 Front copper and silk

The decision to place nearly all components on board A was made after it became apparent that it would be possible to do so. This nearly eliminated the need for the ribbon cable connecting board A and B in the original design, making manufacture and installation easier. The constrained space meant that fewer considerations could be taken with regards to signal noise. The front copper is presented in figure 37. This is further discussed in section 14.

MCU, hand IMU and crystal: Mirroring the original design, the MCU was placed in the upper left corner. This was done in part to keep the MCU away from the noise it may have been subjected to were it placed closer to the motor driver an low voltage assemblies, and in part because it would have been difficult to use this space for any other components. The placement of the hand IMU to the left of the MCU was dictated by the position of the relevant pins on the MCU. As with the other control units, the crystal was placed as close to the MCU as possible while making space for other traces.

Flash and UART header SRL_FLS_URT1: The merged programming and UART pin header was placed in the upper right in order to be available while the board is installed in the arm during testing. The header is horisontal, and the pin descriptions on the silk match the header footprint in the board plane. They are aligned such that each symbol corresponds to the pin relatively above them on the board plane. E.g., the symbol "D" corresponds to the pin installed on the left column, second row from the bottom. The full silk description can be found in table 18.

Reset, boot, 5V and 3.3V jumpers: The jumpers are below and to the right of the MCU, their footprints reduced from 2.54 mm to 1.27 mm in order to save space.

Voltage regulators: Inspired by the original design of hand B, the voltage regulators have been placed such that their top holes coincide with mount points H1 and H2. The vertical footprints used in the other boards were swapped for horisontal ones.

CAN transceiver: The CAN transciever (U1) is placed between the voltage regulators, constrained by the position of the MCU's CAN port and the gear hole to its upper right. This may be a suboptimal position with regards to noise, see sections 16 and 14 for more information.

Flash/UART header	Meaning	IMU interrupt	Meaning
R	Reset	0	Hand IMU interrupt
D	Data	1	Lower arm IMU interrupt
G	Ground	G	Ground
C	Clock		
+	3.3V		
R (bold)	UART Receive		
T	UART Transmit		

Table 18: Hand flash/UART and IMU interrupt headers

CAN, power, IMU header PWR_CAN_IMU1: This header is placed below the left voltage regulator and contains the IMU header (see 9.6.2) in the left column and CAN/power header (see 9.5.2) in the right column. Its placement was dictated by the lack of space for wires in the hand.

IMU interrupt jumper IMU_INT1: This jumper is placed below the CAN and IMU header, the placement dictated by the lack of space for wires in the hand. Both IMU interrupt pins may be grounded by this jumper, specified in table 18.

Motor pin header MTR1: This header is in the bottom left corner of hand A. The original unit places this header on the bottom edge of hand B, but keeping this layout was elected against in the interest of minimising space dedicated to the A/B ribbon cable. The header was given its new position after it was determined that the motor connectors would fit in that space. The upper 4 positions correspond to MTR_DRV1, and the lower 4 to MTR_DRV2.

Motor encoder slots: The motor encoder pins slot into plated holes along the bottom edge of hand A, their positions entirely dictated by the encoders themselves. In an effort to reduce noise potential in what is clearly a suboptimal situation, the signal traces were kept straight and along the left edge of the board. Minimum trace to outline capabilites of JLCPCB were respected[11].

Motor driver bulk capacitors: The bulk capacitors are grouped in the lower right quadrant of hand A. Where the other bulk capacitors are THT, these were changed to SMD in order to save footprint space. Additionally, this opened up space on the back copper.

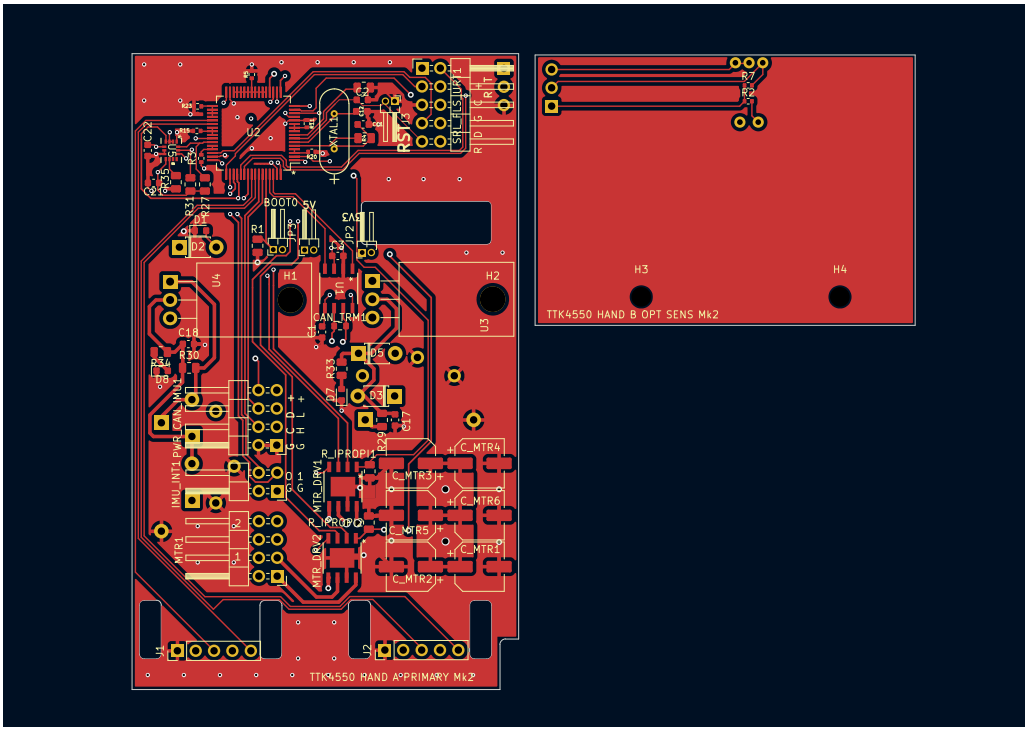


Figure 37: Front copper and silk layers of the hand PCBs

9.7.3 Back copper and silk

Adjustment potentiometers: RV1 is in the middle right of hand A, facing outwards. This makes adjustment practical, as previously discussed. RV2 had to be placed in the middle left, facing downwards, due to conflicts with other units and signal traces. Due to it being a THT component, placing it underneath the motor bulk connectors would have been impractical. Its placement makes it unreachable after the board is installed, and **it must be tuned before installation**. RV2 tunes the 5V assembly.

Ribbon cable: The connection point for the 3-wire ribbon cable connecting the twist optocoupler on hand B is in the upper right corner. Due to mechanical constraints inside the hand, the next available position is below the H2 mount point. Using that position would require hand B to be approximately 10 mm longer, and the ribbon cable would block access to RV1 mounted on the back copper.

Optocoupler: Placement was entirely dictated by the position of the metal plate which triggers the optocoupler, footprint can be seen in the upper middle of hand B.

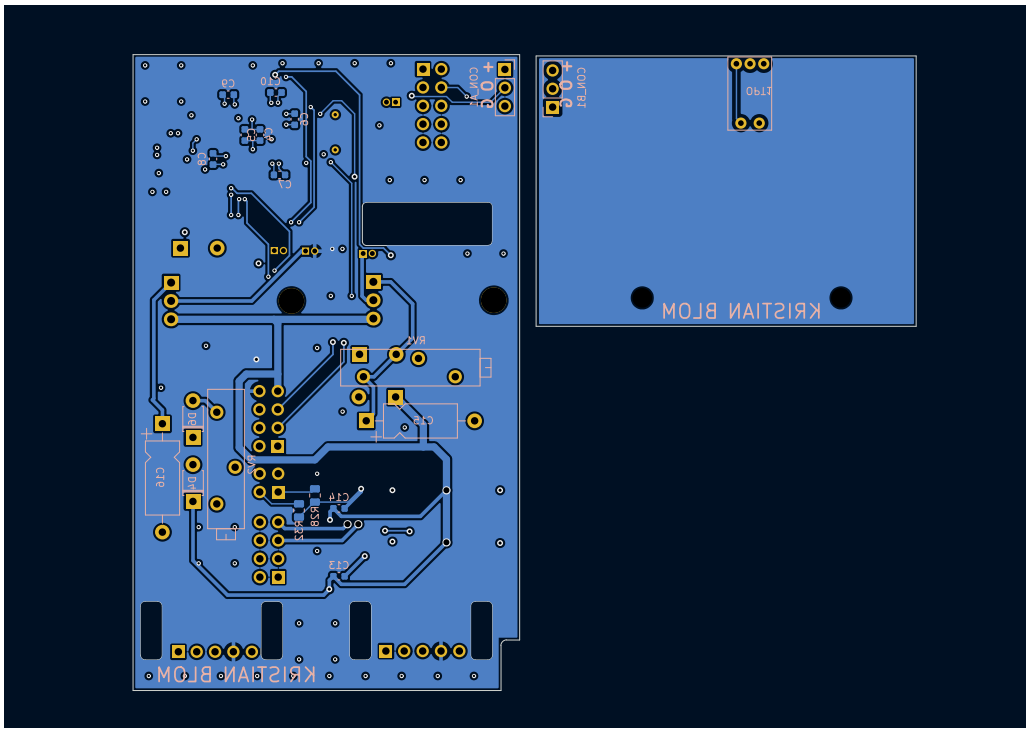


Figure 38: Back copper and silk layers of the hand PCBs

9.8 IMU boards

9.8.1 Outline and mount points

The lower and upper arm board mounts have 3 mount points each, all of which were utilised for the IMU boards. See figure 39. **The board outline is 40x40mm**.

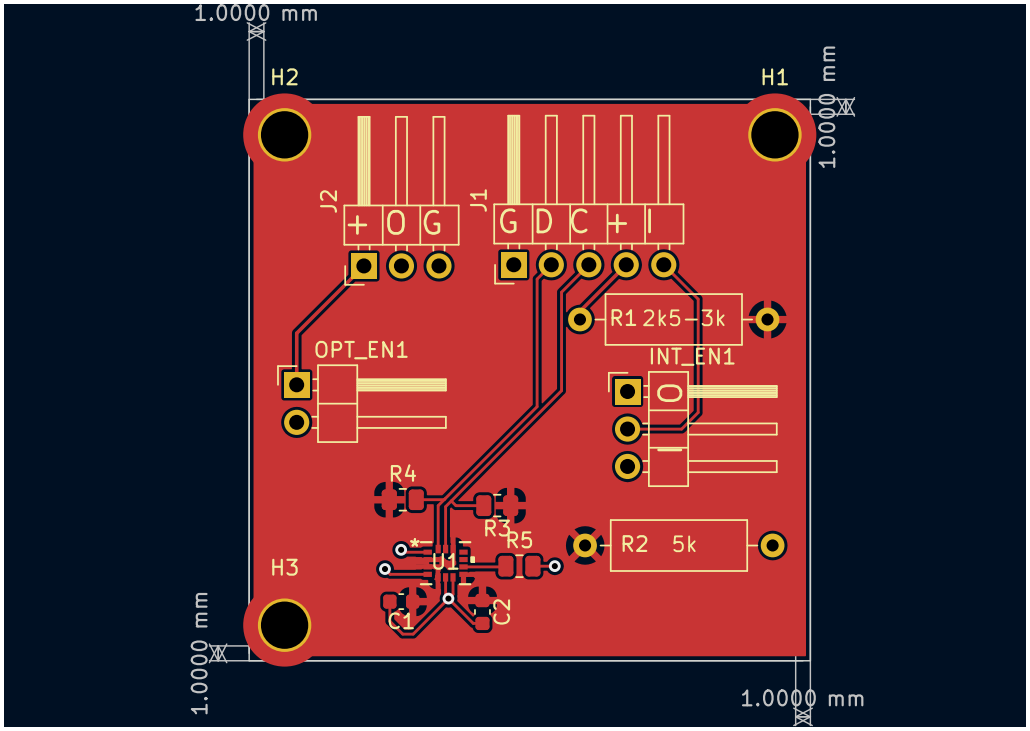


Figure 39: Measurements of outline and mount points of the IMU boards, and front copper design

Mount point	Mode	Footprint	Reference
H1: Upper right	Utilised	3.2 mm	1 mm
H2: Upper left	Utilised	3.2 mm	1 mm
H3: Lower left	Utilised	3.2 mm	1 mm

Table 19: Table of IMU harpoint mount points

9.8.2 Front copper and silk

As mentioned, the IMU boards only use two layers of copper. The front copper is filled with 3.3V, while the back copper is filled with ground. Via stitching has therefore been omitted in this board[6]. The front copper is presented in figure 39. The design had to accomodate the hardpoints in both the upper and lower arm sections. They are different in that the upper arm allows for vertically mounted headers, while the lower arm may only fit horisontally mounted headers. The pin header orientations were chosen due to constraints on wire space in the lower arm. The placements were chosen to minimise trace crossings, oriented around the pinout of the IMU. A summary table of the silk symbols can be found in table 20.

Wrist optocoupler header: J2, upper left corner.

I2C, power and interrupt header: J1, upper right corner. The ”interrupt” pin may forward either the wrist optocoupler signal or the IMU interrupt, depending on the setting of the INT_EN1

[6]Obviously

jumper.

Optocoupler enable jumper: OPT_EN1, middle left.

Interrupt selection jumper: INT_EN1, middle right. Connecting the upper and middle pins sends the optocoupler signal to the interrupt pin of J1. Connecting the lower and middle pins sends the IMU interrupt.

Optocoupler header	**Meaning**	**I2C header**	**Meaning**
+	5V out	G	Ground
O	Optocoupler signal	D	Data
G	Ground	C	Clock
Interrupt jumper	**Meaning**	+	5V in
O	Optocoupler signal	I	Interrupt/Optocoupler out
I	IMU interrupt		

Table 20: IMU board pin header silk description

9.8.3 Back copper and silk

The IMU board back copper is presented in figure 40

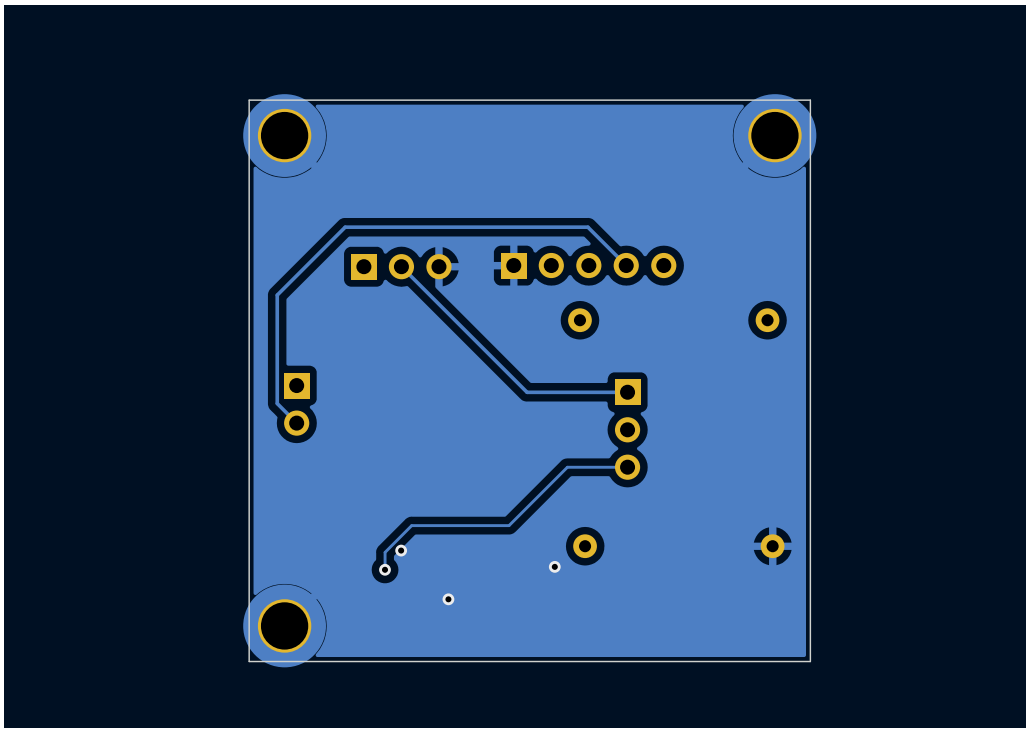


Figure 40: IMU board back copper design

9.9 Auxiliary boards

As a visual reference, the original auxiliary boards are presented in figure 41

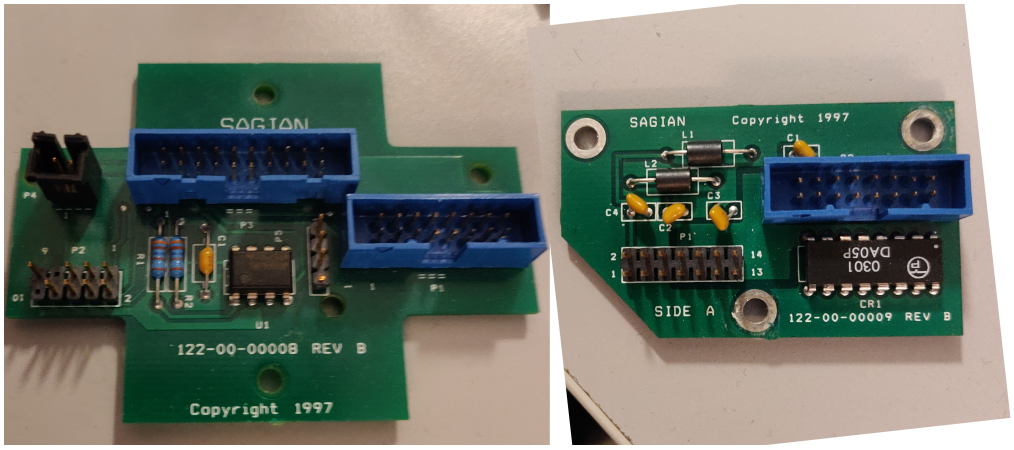


Figure 41: The original bogie and rail boards, used as a reference for the refurbished design

9.9.1 Outline and mount points

The auxiliaries have 5 mount points. Two connect the bogie (dubbed A) board to the bogie mount points, and three connect the rail board (dubbed B) to the rail entry point. See figure 41 for a visual reference. Per **RM1.2: PCB mount points** (4.2.3), all mount points were utilised. The board outlines are irregular, but **their rectangular outlines are 78x56.5mm (bogie) and 58.5x34.5mm (rail)**.

Mount point	Mode	Footprint	Reference
H1: Upper right	Utilised	3.6 mm	2.05 mm
H2: Middle right	Utilised	3.6 mm	28.05 mm
H3: Lower right	Utilised	3.6 mm	4.05 mm
H4: Left	Utilised	3.2 mm	4.2 mm
H5: Middle left	Utilised	3.2 mm	19.75 mm

Table 21: Table of auxiliary hardpoint mount points

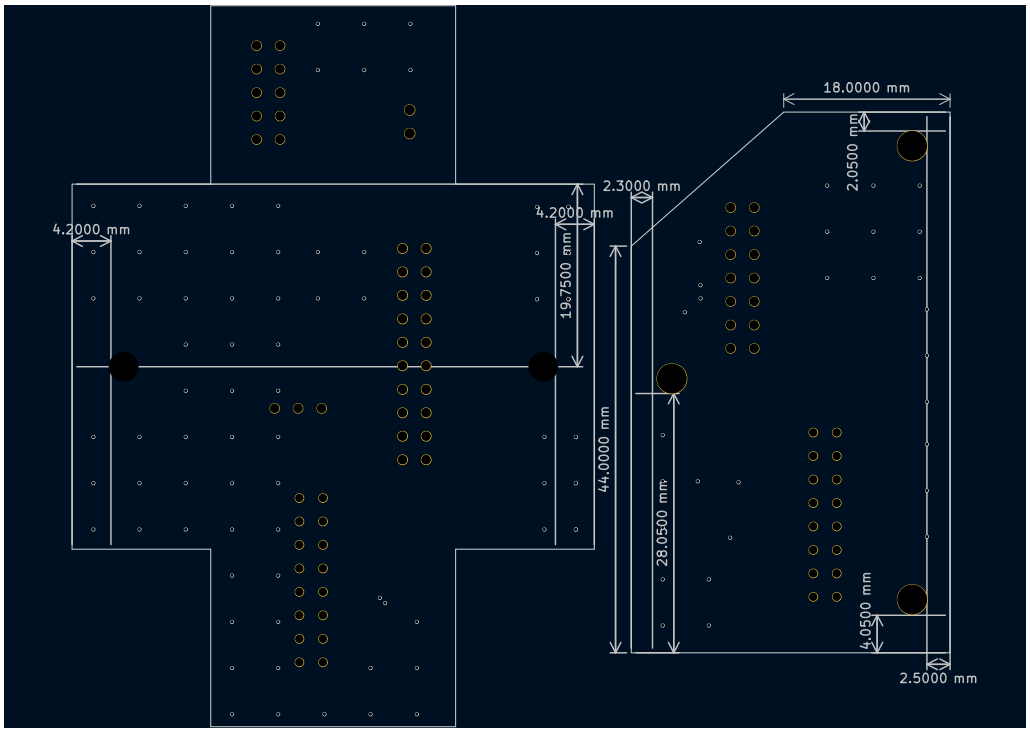


Figure 42: Measurements of outline and mount points of the auxiliary boards

9.9.2 Front copper and silk

The placement of components on the auxiliary boards are virtually identical to those of the original. The primary reason these boards were included in the refurbishment was that it was deemed easier to make new boards than mapping the originals completely enough to reuse them safely – a process which also could have resulted in a decision to make new versions. The silk refer to the boards as "Mk2", as a production error resulted in a remake. The correction was insignificant, and a better name would have been "Mk1.1". The names "bogie board" and "rail board" were introduced in this report, and the boards were previously referred to as "Auxiliary A" and "Auxiliary B" respectively. The placement and rotation of all components were chosen to ensure compatibility with the original parts. Some effort was made during the circuit design to minimise trace crossings in the layout. A summary table of the silk descriptors can be found in 22.

ENC0: Upper left of A, rail encoder connector pin header.

MTR1: Upper right of A, rail motor connector socket.

MAIN1: Middle right of A, 20 pin ribbon connector socket.

END_SWITCH1: Middle right of A, end switch connector header.

OUT1: Lower edge of A, 16 pin rail cable connector socket.

DSUB1: Upper left of B, 14 pin DSUB15 connector pin header.

OUT2: Lower edge of B, 16 pin rail cable connector socket.

MAIN1 connector	Meaning	OUT1 connector	Meaning
ES	End switch	G	Power ground
VB	USB VBUS	+	48V
DM	USB DM	-	Signal ground
DP	USB DP	P	USB DP
5	5V	M	USB DM
B	Rail encoder ch B	V	USB VBUS
A	Rail encoder ch A	R	UART Receive
T	UART Transmit	T	UART Transmit
R	UART Receive	G	Power ground
G	Ground	5	5V
M1	Rail motor ch 1		
M2	Rail motor ch 2	**End switch**	**Meaning**
+	48V	G	Ground
		O	Output
		+	5V

Table 22: Auxiliaries front silk description

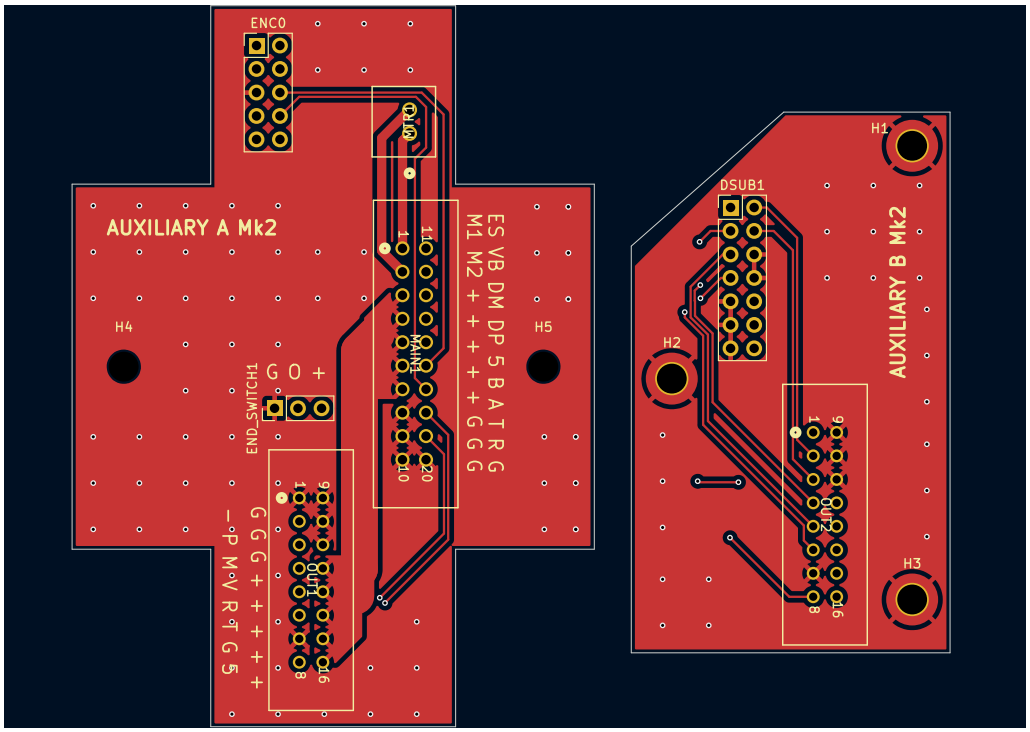


Figure 43: Front copper and silk layers of the auxiliary PCBs

9.9.3 Back copper and silk

The auxiliary back copper is presented in figure 44.

CR1: Lower left quadrant of B, TVS array protecting UART and USB data lines from surges.

D1: Below H2, indicator diode for TVS array, only visible during testing. Placed on the back copper due to the orientation of the rail board after installation.

Silk symbols are as described for the front silk. Placed on the back copper due to the orientation of the rail board after installation.

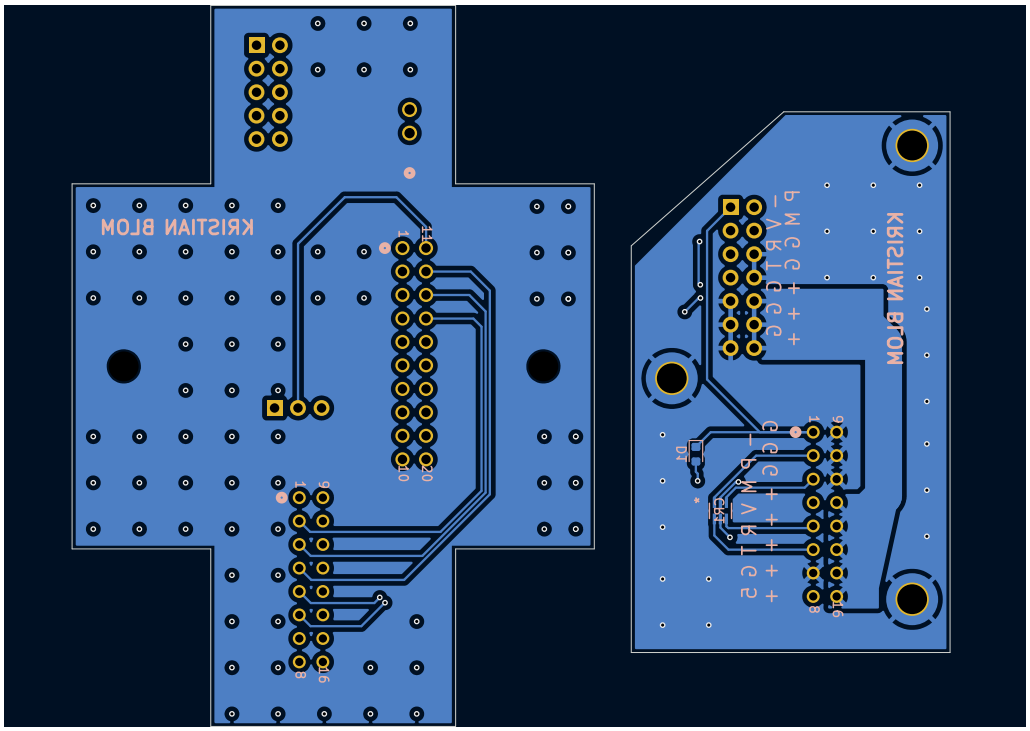


Figure 44: Back copper and silk layers of the auxiliary PCBs

10 PCB production

All boards produced for installation in the arm were manufacured by JLCPCB, a company specialising in low-volume PCB prototype production. Their default manufacture setting was chosen in all instances, with two notable exceptions: A lead free surface treatment was chosen in all instances. For the torso and shoulder boards, the impedance matching layer stackup JLC04161H-3313 was chosen to accomodate the USB assemblies.

Front copper SMD components were soldered in a reflow oven at Omega Verksted, while THT and back copper SMD components were soldered by hand. A stereoscopic microscope was used to aid the placement of the smaller components, such as the pulldown resistors with an 0402 footprint, and to inspect the results of the reflow oven solders. Use of the reflow oven is illustrated in figure 45.

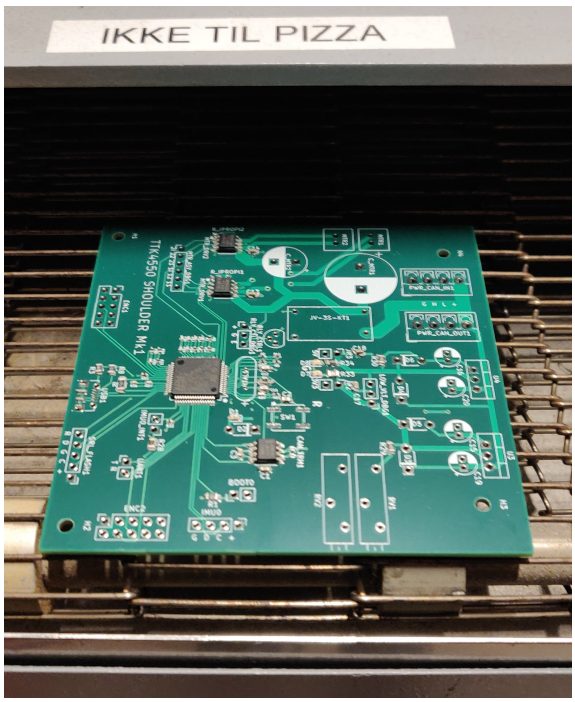


Figure 45: The refurbished shoulder board entering the reflow oven at Omega Verksted. From bottom to top, visible ICs are CAN transciever, MCU and the two motor drivers

Hand production note: There are several THT footprints overlapping with the positions of horisontal pin headers on hand A. In an effort to prevent solder tin mounds from blocking access to the pins, the headers were installed at the maximum angle the mechanical slack between the pin headers and their footprints would permit. See figure 46.

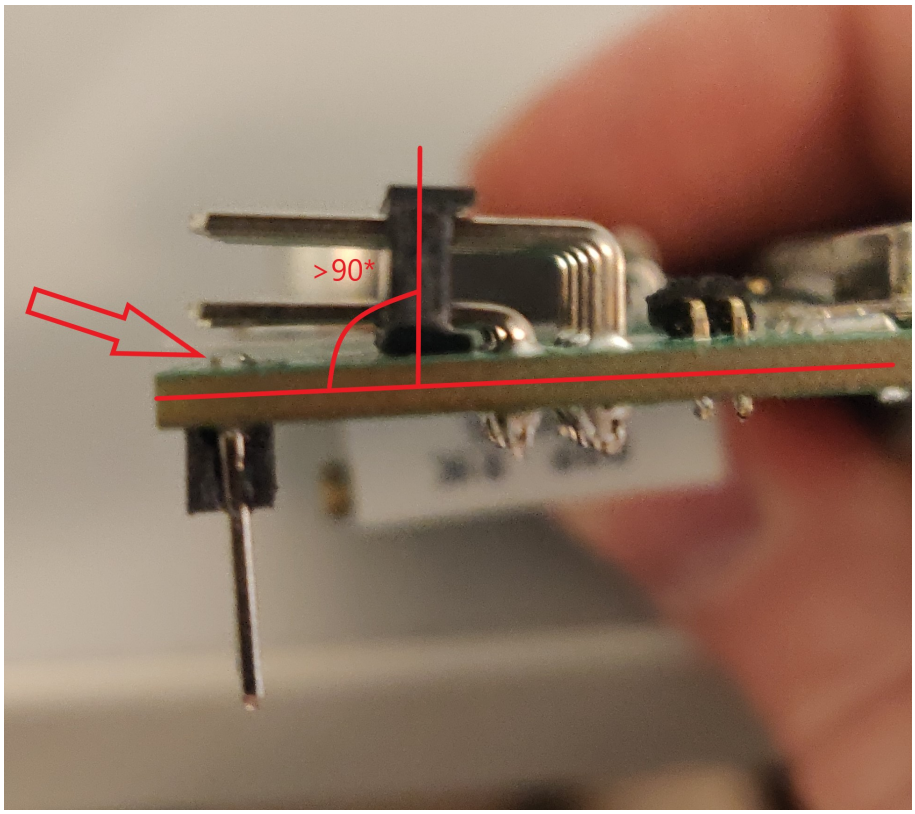


Figure 46: Illustration of the hand pin header angled installation. Arrow points at a solder tin mound

Figures 47, 48 and 49 present the produced torso, shoulder and hand control units, respectively. Bogie and rail boards are presented in figure 50. Refurbished units are on the right.

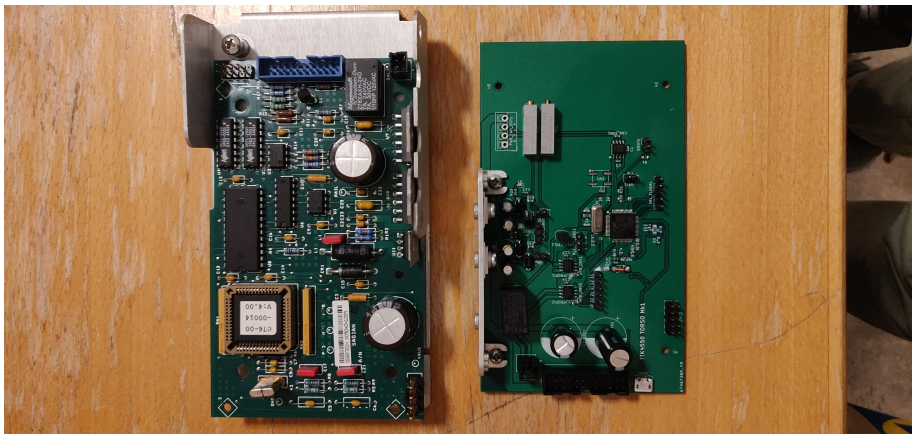


Figure 47: Comparison of the old and new torso control unit

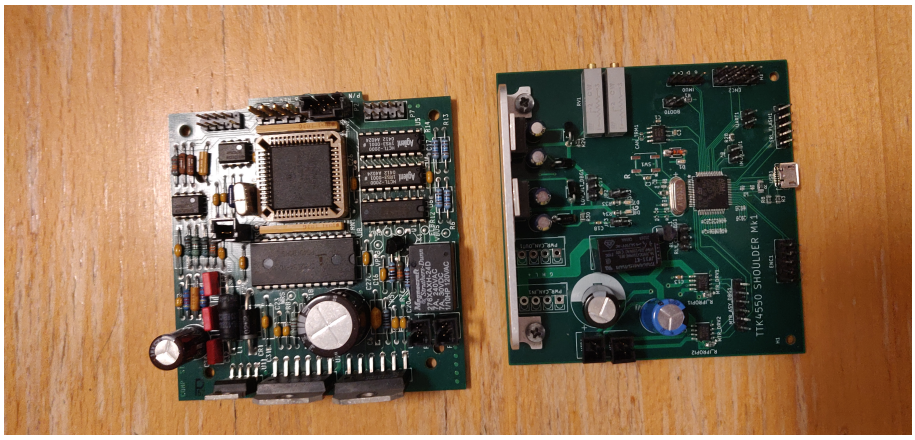


Figure 48: Comparison of the old and new shoulder control unit

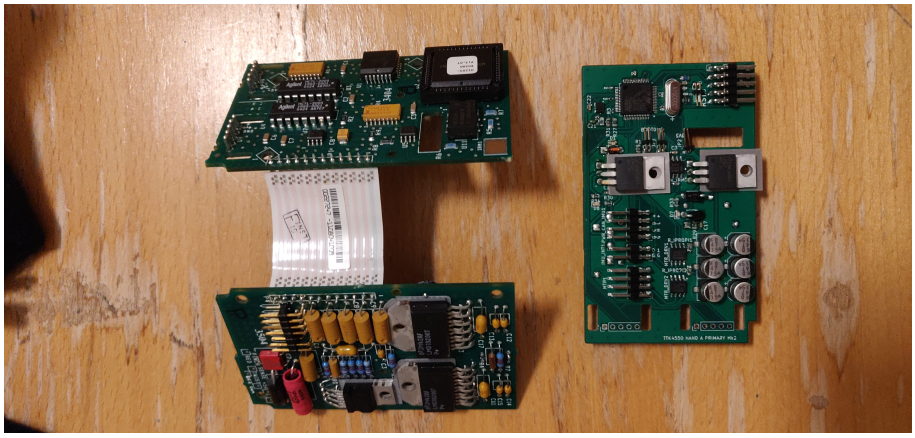
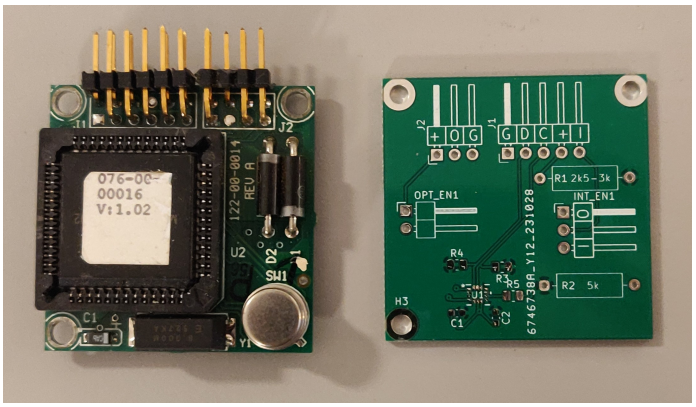
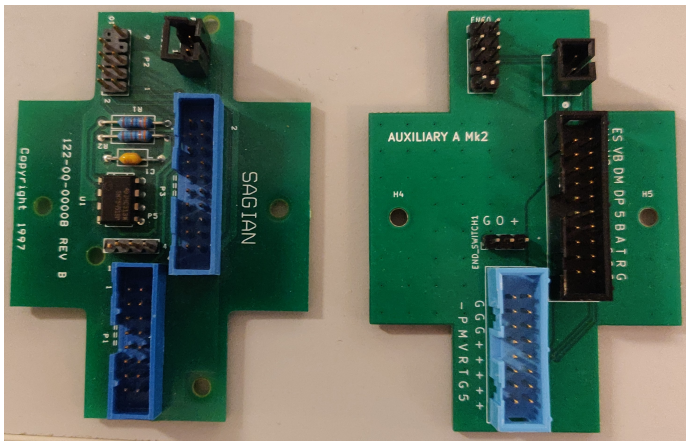


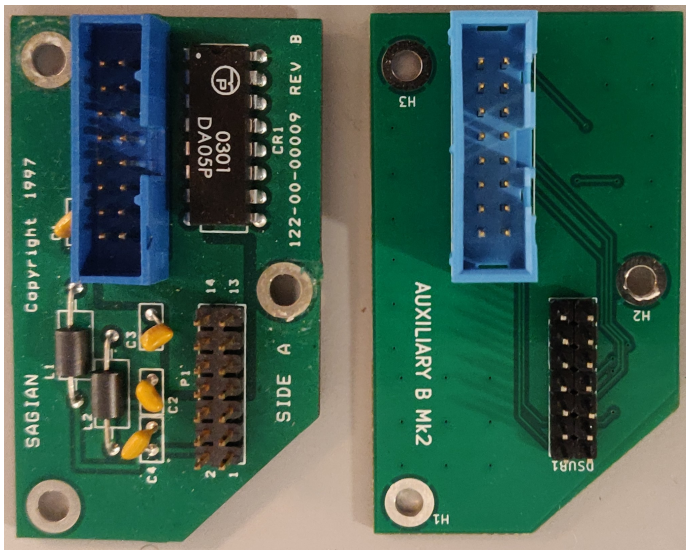
Figure 49: Comparison of the old and new hand control unit (hand B not pictured)



IMU. Not soldered due to design error, see section 13



Bogie



Rail

Figure 50: Comparison of the old and new IMU and auxiliary units

11 Software design

11.1 Tools

11.1.1 CubeMX

As discussed in section 8.2.6, the ST software suite CubeMX was used for MCU configuration. Additonally, it was used to generate source files and a Makefile for use with VSCode.

11.1.2 ST-LINK programmer

An ST-LINK debugger was used for programming the MCUs in the project. The debugger was integrated in the NUCLEO-STM32F303 development kit, which imposed some limitations on its use. For instance, the embedded UART interface could not be used for communication with MCUs external to the development kit.

11.1.3 VSCode

VSCode with the extension "STM32 for VSCode"[8] was used for software development, compilation and programming (flashing) of the MCUs in this project.

11.2 Driver implementation

As mentioned in section 2.3, two of the goals of the project was to write hardware (low level) drivers and a "simple world interface". These were implemented as part of the test and verification regime described in section 12. The intention for the "world interface" was to send and receive data via USB or UART. UART was successfully implemented for the test units, but hardware and time constraints prevented this from being implemented on the produced control units. Items 6 and 7 of the scope description can therefore only be said to have been partially fulfilled. This is further discussed in section 13.

11.3 Test unit implementation

As a part of the test regime implemented for item 5 of the scope description in section 2.3, basic hardware drivers were implemented for use on the produced test units described in that section. The drivers were written with further development on the produced control units in mind, for instance by avoiding hard coding peripheral addresses unique to the NUCLEO development kit into the tests or by configuring the NUCLEO development kit MCU similarly to those of the produced control units. The source code is available from the github repo[5], and a more comprehensive discussion of the drivers can be found in the appendix.

For each set of tests operating on the same systems, one configuration was made in CubeMX and peripheral initialisation code was generated for use in VSCode. Test code was written by referring to the STM32 HAL user manual[31], and making minimal abstractions for ease of use.

12 Testing and validation

Testing and validation is an integral part of any development project. In this project, the work was "outsourced" to another subject, *TTK8: Construction of embedded systems*. This was done in order to increase the overall time budget of the project, and thereby ensure that testing was given due attention. An in-depth report was written for that subject, and this section summarises its contents. This is mentioned in the interest of transparency with regards to rules concerning self-plagiarism at NTNU[14][15]. The tests and their results are presented in tables 23, 24 and 25. The results are discussed in section 13. The tables were orgininally presented in the TTK8 report[4]. The report is included in the appendix.

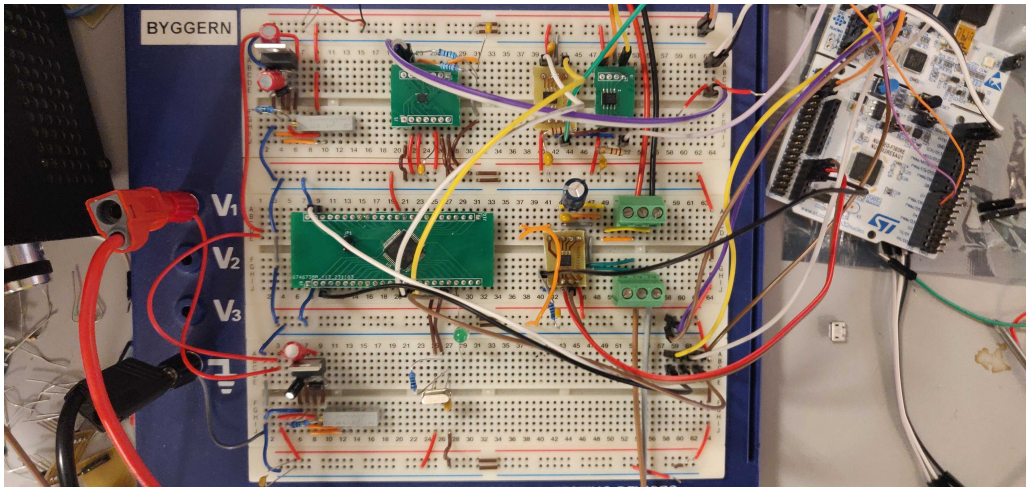


Figure 51: Visual reference of the breadboard used for circuit design validation

Validation was divided in three phases:

1. Test design based on requirement specification and circuit design,
2. Circuit design validation, and
3. Validation of produced units

12.1 Phase 1: Test design

After each assembly discussed in section 8.2 was designed, a number of tests were specified with success criteria such that the appropriate function of the assemblies would be ascertained ("design tests"). A similar process was undertaken for the produced PCBs ("production tests"). The design and production tests largely mirror one another, as the PCBs were designed from the evaluated circuits.

12.2 Phase 2: Circuit design validation

The voltage regulator, motor driver, MCU power and programming, IMU, and CAN assemblies were prototyped on a breadboard. This included designing and producing breakout boards for the ICs. Tests requiring software, such as those pertaining to the motor drivers and IMU, were run from a NUCLEO-STM32F303RE development kit. Negative tests results were noted. Some led to immediate changes in the PCB design, but most were relegated to the "further work" section of this report, see 16, due to a lack of time. See also section 13.

12.3 Phase 3: Validation of produced units

Finally, the produced units were subjected to the production tests. Not all tests were completed due to a lack of time, or only partially completed. This is further discussed in section 13.

Test no	Test name	Component	Req.	Description	Criterion	Result
TE1.1D	DC-buck 48-3V3	LM317	RE1	Verify design of low-voltage power supply circuit	Measure 3.3VDC with voltmeter	OK
TE2.1D	DC-buck 48-5V	LM317	RE1	Verify design of low-voltage power supply circuit	Measure 5VDC with voltmeter	OK
TE3.1D	AC/DC 240/48	TBD	RE1	Verify design of high-voltage power supply circuit	Measure 48VDC with voltmeter	N/A
TC1.1D	Motor driver PWM-signal	DRV8251A	RC2.2	Verify design of motor driver circuit	Observe PWM pulse train on oscilloscope	OK
TC1.2D	Motor PWM control	DRV8251A	RC2.2	Verify design of motor driver circuit	Observe motion in connected motor	OK
TC1.3D	Current draw motor	DRV8251A, STM32F303	RC2.4	Verify measurement of motor current draw	Read current measurement from motor driver to ADC on MCU	OK, raw data only
TC2.1D	I2C-bus to IMU	LSM6DSM	RC2.6	Verify connection of I2C to IMU	Read ACK from IMU with oscilloscope	OK
TC2.2D	Contact IMU	LSM6DSM	RC2.6	Verify design of IMU circuit	Read identity from IMU's WHOAMI register	OK, read 6A
TC3.1D	CAN circuit	TJA1057BT	RC3.1	Verify design of CAN transceiver circuit	Read CAN message on oscilloscope on CANH/L lines	OK
TC3.2D	CAN bus between MCU	STM32F303	RC3.1	Verify that CAN messages are sent and received	Observe correct decoding of CAN message in receiver	Negative.
TC4.1D	USB circuit	STM32F303	RC3.2	Verify USB connection between MCU and external computer	TBD (To Be Determined)	N/A
TC5.1D	MCU circuit	STM32F303	RE1.2	Verify design of power and data circuit for MCU	Successful flashing of software to MCU	OK, LED blinking

Table 23: A summary of all design tests and their results

Test no	Test name	Component	Req.	Description	Criterion	Torso	Hand	Shoulder
TE1.1P	DC-buck 48-3V3	LM317	RE1	Verify produced low-voltage power supply circuit on all three boards	Measure 3.3VDC with voltmeter	OK	Short circuit in GND/ 48V	OK
TE2.1P	DC-buck 48-5V	LM317	RE1	Verify produced low-voltage power supply circuit on all three boards	Measure 5VDC with voltmeter	OK	Fail	OK
TE3.1P	MCU power supply	STM32F303	RE1	Verify power supply to MCU	Measure stable 3.3V at MCU for all three boards	OK	Fail	OK
TE3.2P	MCU clock circuit	STM32F303	N/A	Verify clock circuit for MCU	Measure stable 16MHz at MCU for all three boards	OK	Fail	OK after resistor swap.
TE4.1P	Motor driver power supply	DRV8215A	RE1	Verify power supply to motor drivers	Measure stable 3.3V at motor drivers for all three boards	OK	Fail	OK
TE5.1P	CAN transceiver power supply	TJA1057BT	RE1	Verify power supply to CAN transceivers	Measure stable 3.3V at CAN transceivers	OK	Fail	OK
TC1.1P	Encoder reading	HEDS-9100	RC2.1	Verify that all 6 existing encoders work	Observe quadrature pulse train on oscilloscope	OK	OK	OK
TC1.2P	Motor driver PWM signal	DRV8215A, STM32F303	RC2.2	Verify generated PWM signals for motor control for all 6 drivers	Observe PWM signal 0-100% on oscilloscope	OK	Fail	OK
TC1.3P	Motor current draw	DRV8215A, STM32F303	RC2.4	Verify measurement of current draw from motor for all 6 drivers	Read measurement via STM's ADC to terminal 0-100%	Not completed due to missing UART to PC	Fail	Fail

Table 24: A summary of all production tests and their results (cont. in table 25)

Test no	Test name	Component	Req.	Description	Criterion	Torso	Hand	Shoulder
TC2.1P	Angle measurement from IMU	LSM6DSM, STM32F303	RC2.6	Verify angle measurement from IMU to MCU for all three IMUs	Read measurement of inclination angle over I2C to terminal	N/A	Fail	Not completed due to lack of time, successful in test bench
TC3.1P	CAN bus between MCU	STM32F303	RC3.1	Verify sending of CAN messages between master and slave MCUs for both slaves	Read a CAN message sent master-slave-master to terminal	Not completed due to lack of time	Fail	N/A
TC4.1P	USB connection	STM32F303	RC3.2	Verify that messages sent over USB from external computer to master MCU are received	Read a USB message sent external-master-external to terminal	Not completed due to lack of time	N/A	N/A
TC5.1P	Optocoupler	OPT971N51	RC2.8	Verify that optocoupler for wrist position is correctly connected	Read correct high/low from optocoupler in wrist to terminal	N/A	JLCPCB sent the wrong board	N/A
TC6.1P	End switch	TBD	RC2.8	Verify that end stops in linear track are correctly connected	Read correct high/low from end stops in linear track to terminal	Not completed due to lack of time	N/A	N/A
TC7.1P	MCU circuit	STM32F303, ST-LINK	RE1.1	Verification of produced MCU board	Successful flashing of software to MCU	OK	Fail	OK
TC8.1P	Communication	STM32F303, ST-LINK	RC3.2	Test of simple communication between external PC and MCU	Read UART message over oscilloscope	OK	Fail	OK

Table 25: Cont.: A summary of all production tests and their results

13 Results

13.1 Test results

Only non-positive results are discussed here, and the reader is referred to the appendix for an in-depth presentation of the results.

13.1.1 Design tests

The design tests, presented in table 23 were for the most part successful. The most critical parts of the system, such as power supply, motor drivers, encoder readings and MCU flashing were successful without notes.

TC1.3D: Raw data from the ADC was read successfully. The shoulder control unit was tasked with moving the wrist joint back and forth in a waving motion, and a pattern in the ADC data was observed which was concurrent with joint movement. However, it could not be established wether or not the measured current draw, i.e. the conversion from raw ADC data to current, was likely to be correct.

TC2.nD: The IMU board circuit was redesigned after an initial test. It had been implemented on the bread board according to figure 25, which features a voltage divider from 5V to approximately 3.3V. When it had been assumed that this would be sufficient to power the IMU, the I2C bus pullups had not been accounted for. Acceleration in the Z axis was successfully read after the voltage divider had been switched to the regulated 3.3V line.

TC3.nD: The CAN circuit was tested between the development kit and the MCU on the breadboard. While a correctly configured CAN message could be observed by oscilloscope on any point of the data lines, no indication was observed that the recipient MCU (breadboard) had registered the message.

13.1.2 Production tests

The production tests were not as successful as the design test, revealing multiple major and minor errors – including several which were not intentionally covered by the tests. The production tests are presented in tables 24 and 25.

TE1.1P Hand: The hand control unit was found to be short circuited between the 48V entry point and ground.

TE3.2P Shoulder: A constant voltage of 2V was initally measured between the shoulder oscillator legs. The error was found to be the oscillator resistor, which had an installed value two orders of magnitude greater than the design value. Replacing the resistor with the correct value cleared the issue.

TC1.3P Torso: As the UART connection integrated on the development kit could not be configured to communicate with MCUs other than the one on the development kit, no tests requiring UART/communication with external computer could be completed in time. **Note** that "master" and "slave" in the description of TC1.3P refer to the intended roles of the torso and shoulder/hand units in the system respectively. CAN itself peer-to-peer.

TC2.1P Shoulder: Not completed on the shoulder due to missing UART and short time. However, a value of approximately 16400, the expected value for 1G of acceleration, was read repeatedly from the IMU z axis using the breadboard and development kit. The value decreased as the board was rotated. While not a positive result outright, this serves to indicate that the IMU assembly is correctly designed and programmed.

TC5.1P Hand: The manufacturer shipped the wrong board, and one completely unrelated to

the project was received instead. Attempts at having them remade and shipped correctly for free within the required timeframe were unsuccessful.

TC8.1P: This test was implemented in an attempt at verifying the torso and shoulder's ability to communicate by UART. A single ASCII table symbol was sent repeatedly on the UART pin. The bit sequence was read by oscilloscope and confirmed to match that of the ASCII symbol.

13.2 Qualitative evaluation of produced PCBs, aspects not covered by tests

As mentioned, many noteworthy discoveries were made, whose contents were not explicitly covered by the tests. They are presented here in no particular order.

USB circuit design: During the review of circuit design for the production of this report, it was discovered that the 1.5kΩ pullup resistor was placed on the wrong data line of the USB bus according to AN4879[28], which specifies that it must be placed on the DP line. However, the USB specification chapter 7.1.5.1[37] states that the resistor may be placed on the DM line. Furthermore, AN4879 states that "A DP pull-up must be connected only when VBUS is plugged" (Fig. 5 description), and it is not known whether the current design, with the pullup always connected and the line driven directly by the MCU, as opposed to via an external transistor, is viable. Finally, the ST USB middleware expects VBUS detection to happen on pin PA9, currently occupied by one of the I2C bus lines.

Shoulder short circuit: A short ciruit of the shoulder control unit occured during preparations for a test. The working hypothesis is that a voltage of approximately 20V was applied across the motor relay debug header, causing damage to the circuit downstream.

MAIN1 socket reversal: During the integration of the torso control unit with the bogie board, a mismatch between the pin configurations of the 20 pin connector sockets of the two was discovered. An investigation determined that this was due to the schematic symbols being different. In both cases, a generic 2x10 pin header symbol was used, but the pin numbering schemes were different: The torso symbol uses a "top to bottom" scheme, while the bogie symbol uses "odd/even" . This led to a reversal of the signal order on one of the two rows on the bogie board.

Shoulder bulk capacitor: The footprint of the $220\mu F$ bulk capacitor of the shoulder was reversed, leading to it being installed with reversed polarity. The capacitor was damaged[7] and had to be replaced, but no further damage to the circuit was registered. **Note** that the replacement is rated for 16V only.

Wrist optocoupler 3.3V: The wrist optocoupler on hand B was supplied with 3.3V. Its datasheet[36] states that the minimum supply voltage is 4.5V.

Hand A solder points: During the integration of the hand control unit, it was discovered that some of the THT components on the front copper may come into contact with the hand itself, possibly causing shorts. Further investigation is required before a final conclusion may be reached.

ADC/PWM interference: During testing of the motor current draw, TC1.3P, it was discovered that the ADC input pin PA4 produced a steady-state voltage output proportional to the duty cycle of one of the PWM signals produced by the MCU's TIM2 timer peripherals as if the pin had been misconfigured as a DAC. The PWM signals drive motor driver 1. Pin PA4 may be configured for PWM output from TIM2, and it is believed that the observered voltage is the result of interference from the timer module. This failure mode does not seem to be mentioned in any relevant document[3]. Configuring PA4 to the reset state removes the output signal, but also disables the ADC functionality. The measurements were carried out by oscilloscope on the motor driver debug header on the torso and shoulder.

[7] In a non-violent manner

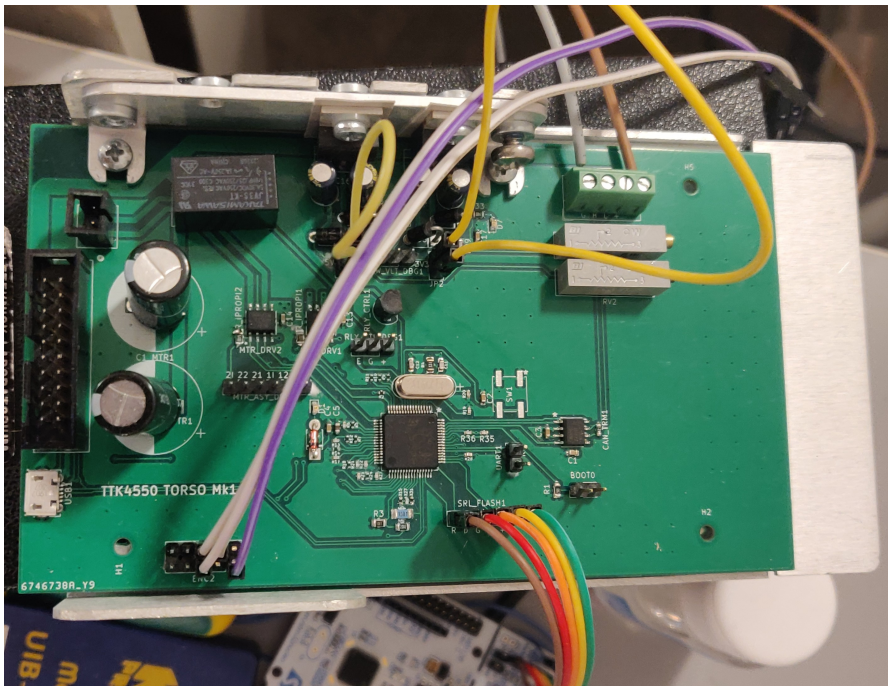
13.3 Mechanical integration

Torso: The positioning and dimensions of the mount points H1, H2 and H5 were wrong, as can be seen in figure 52, and must be entirely redesigned. Additionally the mount points of the heat sink and voltage regulators should be shifted at least 5mm left, such that the board fits entirely within its frame.

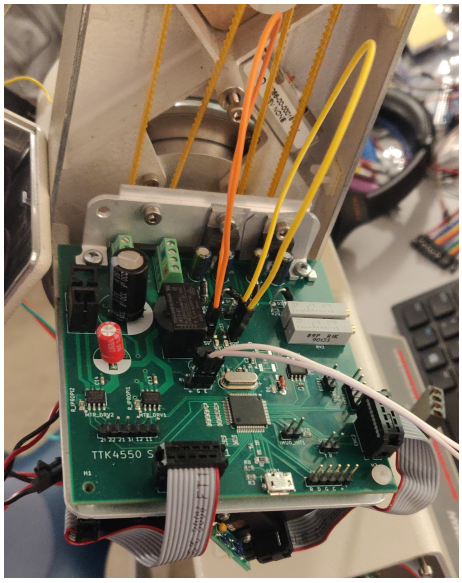
Shoulder: Mount point H1 was approximately 1mm too far left. Remaining mount points fit adequately, but should probably have been 0.1-0.2mm larger.

Hand A: Mount points were correctly dimensioned, but the heads of the currently installed fastening bolts have a diameter such that they go through the voltage regulators. As the top metal of the regulators are connected to their output pins, they are effectively shorted to the hand chassis.

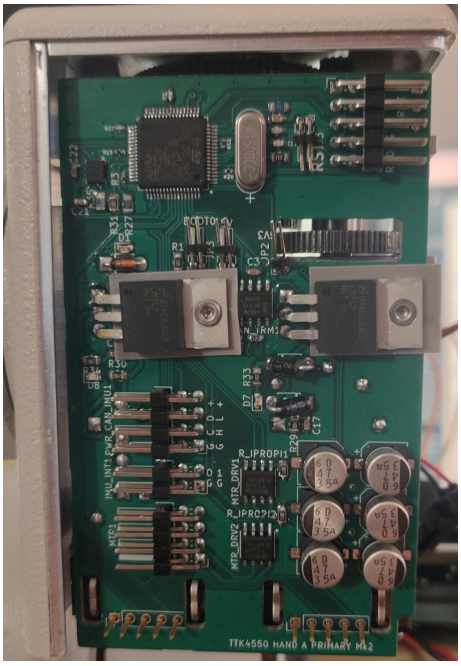
Bogie: Encoder connector header was rotated 180 degrees relative to its original orientation. This makes the connector marginally harder to install.



Torso control unit readied for testing. The frame it is resting in is installed in the torso hardpoint

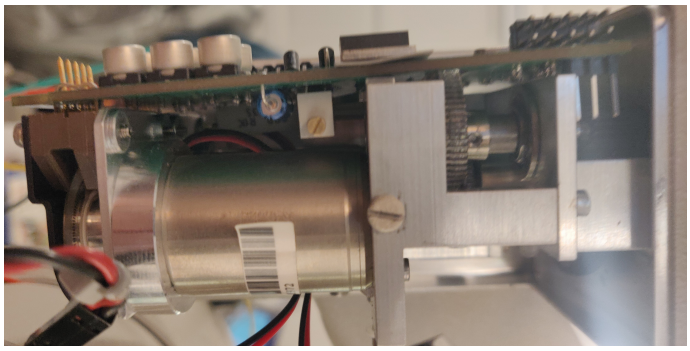


Shoulder control unit. The orange and yellow wires jump 5V and 3.3V, respectively

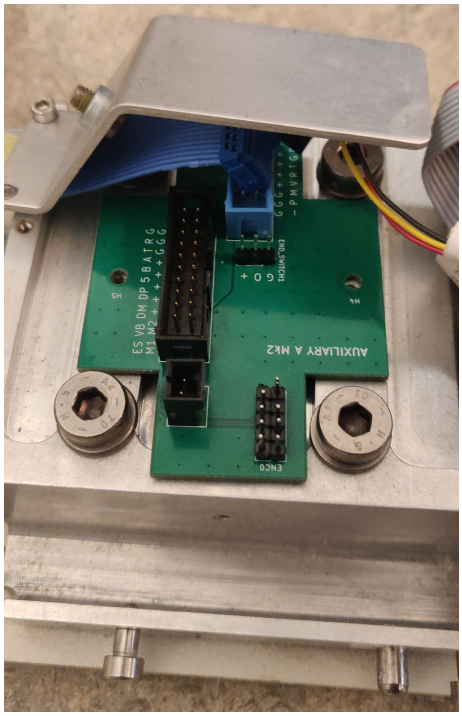


Hand control unit. The bolts touch the metal part of the voltage regulators, erroneously grounding them.

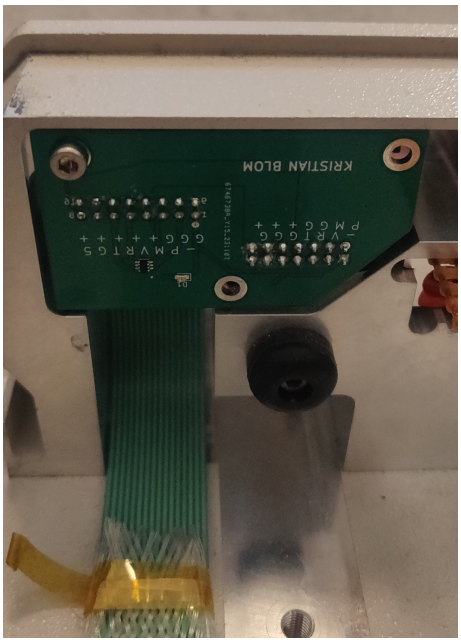
Figure 52: Control units installed in their hardpoints



Underside of the hand board, and a visual reference for the space constraits in the hand



Bogie auxiliary unit installed on its mount point



Rail auxiliary unit installed on its mount point

Figure 53: Auxiliary units installed in their hardpoints, and the underside of the hand

13.4 Drivers

In accordance with items 6 and 7 from the list in section 2.3, simple hardware drivers were written. Most were tested on the development kit. Current capabilities are:

- Encoder signal reading: Implemented. Encoder movement is reliably measured.
- Motor voltage regulation: Implemented. PWM duty cycle is reliably controlled.
- Joint position regulation: Partially implemented. Combining the two previous capabilites lets one joint be controlled to follow a position reference on the form of an encoder counter value. This has not been tested for negative count values, and relevant concepts like "degrees of joint movement per encoder count" are yet to be established.
- CAN message transmission: Partially implemented, see section 13.1.1.
- I2C message transmission: Implemented, see section 13.1.2.
- IMU acceleration measurement: Implemented, see section 13.1.2.
- Current draw measurement: Partially implemented, see sections 13.1.1 and 13.2.
- UART message transmission: Implemented.

The source code will not be presented here, but may be accessed from the github repo[5].

14 Discussion

14.1 Test results

TC1.3D: This uncertainty could likely have been resolved by, for instance, placing an amperemeter in series with the ADC. Unfortunately, this equipment was not readily available.

TC2.nD: The IMU assembly voltage was measured to drop to approximately 1.7V during bus operation, which is the minimum supply voltage for the IMU. The logic high threshold voltage for the MCU's 5V tolerant pins is 1.35V. The assumption is that while the average voltage during bus operation was technically sufficient for the IMU assembly to work as intended, the IMU suffered brownouts approximately once per clock cycle causing the I2C transmission to fail. Using a voltage divider instead of a step-down regulator to supply the IMU assembly was an unnessecary mistake stemming from a lack of experience with pullup resistors in general and I2C specifically. The mindset that "the bus is yet another high-impedance unit with near-negligible current draw compared to the divider" took precedence over the basic understanding of how voltage falls over resistors in parallel.

TC3.nD: Successful registration would have resulted in the recipient MCU blinking its LED, see figure 51 for a visual reference. It is assumed that the fault lies in either CAN message masking or message receive interrupt handling.

TE1.1P Hand: Further investigation is required to find whether the fault is in the design or manufacture process. As all copies of the hand board (1 was soldered, 4 were not) had the same error, and KiCad gave no indication that a short might have occured, the preliminary conclusion is that the fault lies with the manufacturer. No further tests were attempted on the hand unit after this error was discovered. The fault likely could have been isolated by, for instance, attempting to feed 3.3V and 5V directly into their respective jumpers. However, the footprint downsizing from 2.54mm to 1.27mm meant that no compatible jumper wires were availably to try this safely.

TE3.2P Shoulder: This particular result had a trivial solution, but serves as a reminder to visually inspect all components before installation.

TC1.3P Torso: This is a good example of how if the project's PCB production goal had been smaller, the design could have been verified to a larger degree.

TC2.1P Shoulder: The more accurate number is 16384, or half the maximum value of a 16 bit signed integer (INT16). The IMU was configured to a maximum of $\pm 2G$, the ADC converted value stored as INT16. The IMU circuit design error (TC2.nD) indicates that TC2.1P would not have had a positive result with the produced IMU board in any case, but that a redesign using a dedicated voltage regulator would.

TC5.1P Hand: No further elaboration necessary.

TC8.1P: This is a clear indication that UART works, and that using it for external communication is only a matter of aquiring the appropriate adapter.

14.2 General

Overall, the project's results were mixed. While a majority of test results were positive, the end product is one where only one of two motor drivers on two of three boards work as intended, and no communication either between the boards or externally has been completely confirmed outside of the test units. Three primary causes can be pointed at for this somewhat bleak verdict: lack of time to properly implement USB and/or UART, a short circuit in the hand of unknown origin, and interference observed between the ADC and TIM2 peripherals. The latter may be the most interesting "finding" of the project, though wholly incidental and outside of its scope. If the results can be reproduced and confirmed as a hardware error in this particular microprocessor, the manufacturer should probably be notified. As for the short circuit, more investigation is needed

before it can be established whether this was a design or production error.

Resource management also deserves some discussion. It seems likely that if the scope of this project had been reduced to the production of only one control unit, the overall progression could have been better. While some time was saved on circuit design by reusing assemblies, much time was spent on the unique layouts of each control unit. Production time and money spent on components comes on top of this. This resulted in much time being spent on duplicate work which may not have contributed much to the project's progression. If, for instance, the shoulder unit had been the sole focus of the project, almost all functionality could have been tested with a single unit. The remaining control units could then have been designed from a template which had been tested "in vivo". As it stands, all control units will need several minor changes to their design. The same lessons likely could have been learned while developing only a single control unit, saving time and money.

Such an approach could also have led to a more solid theoretical foundation for the project. The Theory section (5) of this report is quite thin, reflecting the project's theoretical underpinnings. The assumption at the beginning of the project was that most of the design elements had already been created and that finding design references would be a minor challenge. While this was to some extent true, a better theoretical understanding would likely have made the evaluation of design choices easier. A good example of this is the USB circuit and the placement of the DP/DM line resistor: if the USB specification had been read before the application note, it would have been clear that the placement is a design choice and not a requirement. Having had that understanding, rather than basing the design on the reference schematic alone, may have prevented that mistake.

Nevertheless, a majority of the defined tests were successful. While several design changes will be necessary for the system to be hardware complete, they are for the most part not complex. Section 16 treats this in more detail, but the most complex change needed is likely the reconfiguring of the MCU pins concering the ADC/timer error. This seems like a surmountable problem, considering the large amount of available pins on the MCU.

One key aspect of the project which was not addressed by the tests was the choice of MCU. The STM32F303 was chosen to a large extent due to its numerous ADC/DAC peripherals. It was thought that DACs would be used for motor driver control, while ADCs would measure IMUs and load cells. The F303 was determined to be sufficient even after it was established that these elements would require PWM and I2C, but a more thorough reevaluation of the MCU selection may have resulted in a more optimal choice. This illustrates the interconnectedness of design parameters affecting the outcome of this project and the relevance of systems design models such as the V-model. Strict adherance to the V-model was not an objective in this project, but taking more time to evaluate preliminary results throughout the design process may have increased the quality of the final product.

The goal of reducing hardware complexity of the system compared to the original should be considered to have been fulfilled. 6 MCUs, 3 of which were complemented by external floating point and encoder counter units, were identified in the original design. They were replaced by 3 MCUs embedded in significantly less complex circuits, judging by the number of components in the original control units. Hand A is a particularly good example of this: Were it not for the optocoupler, the footprint of the entire control unit would have been halved. As for design evaluations of the hand unit, it may have been preferable to swap the signals and the 48V traces under the voltage regulators layerwise, as the signal lines may currently be subject to noise from the regulators. However, this cannot be tested until the cause of the 48V/GND short circuit has been established.

Finally, the project specification and requirements could have been more specific, particularly with regards to power delivery and data transfer rates. The general assumption was that modern equivalents to the originial hardware would be more than sufficient to control the robotic arm considering the increase in computational power per unit volume over the last 30 years. Regarding power delivery specifically, the compontents were chosen to be as similar to the original system as reasonably practical.

There are several aspects which could have been addressed in further detail if time had permitted: The initialisation of MCU peripherals has been relegated to an automatically generated report in

the appendix, but was in reality a non-trivial part of the hardware driver implementation; One hypothesis for the hand short circuit concerns the merging of multiple PCB designs into a single file before sending them to production at JLCPCB; The various interdependent design constraints could have made for an interesting diagram and more comprehensive discussion of systems design theory; The characterisation of the motor torque constants of the motors where no datasheet could be found; And the mechanical integration of the control units almost certainly deserved more attention than the two figures it was given. This report opted to focus on the circuit and PCB design as those represent the bulk of the work in the project.

15 Conclusion

This project set out to design and produce replacement control circuits for an ORCA robotic arm produced in the mid/late 1990s, and write software sufficient to verify their functionality. While the circuits were produced, several shortcomings were found in the verification phase. The project was a moderate success: correcting the mistakes should not be particularly challenging, but will likely be time consuming and expensive. If the hardware production goal had been reduced, more time could have been spent building a theoretical foundation for design choices, increasing the overall quality of the produced units. The initial assumption that most or all design elements could be found from online sources and utilised without a deeper understanding of the underlying electric/electronic design theory was an underestimation of the complexity of both the elements and their interactions.

The project was highly valuable as a learning experience, however. There are few opportunities to gain practical experience with PCB/electronic systems design in an academic setting, which is arguably a relevant skillset for someone specialising in embedded systems design. Considering a starting point of almost no previous experience with KiCad or similar systems, this aspect of the project was a success.

16 Further work: Improvements for Mk1.1

The specialisation project was supposed to lay the foundation for a master thesis project focusing on the implementation of high-level control of the robotic arm. Some improvements to the hardware must be implemented before that can happen, however. The redesign is dubbed Mk1.1.

16.1 High priority

16.1.1 USB assembly

- VBUS voltage divider values: Implement $82k\Omega$ and $33k\Omega$, AN4879 chapter 2.6.
- VBUS pin: PA9 is the default VBUS detection pin, AN4879 chapter 2.6. PA9 is currently occupied by I2C. Find out how to change VBUS detection pin, or move the I2C bus.
- Pullup resistor: Move to DP line.
- Pullup transistor: Confirm that the GPIO pin may drive the DP line directly through the pullup resistor. If not, find a suitable transistor to connect to 3.3V.

Additionally, AN4879 and the USB2.0 should be more closely examined before the changes are implemented.

16.1.2 IMU assemblies

- Consider pulling both 3.3V and 5V lines from the control units to the IMU boards. If not, redesign with a dedicated 3.3V step-down converter.
- Redesign the headers and jumpers on both the control units and IMU boards such that both the IMU interrupt and optocoupler signals may be active. The hand interrupt jumper should be 3-place. *This implies the activation of an additional GPIO pin on the MCU.*
- Move the IMU header on the shoulder unit closer to the voltage regulators in order to shorten wire length. Do not prioritise.

16.1.3 CAN bus assemblies

- Verify that the circuit works using the test units/breadboard. Implement changes in the control units' design, if necessary.
- Move CAN bus connector on torso further out, and also towards the middle such that the CAN bus does not go under the potmeters. Do not prioritise
- Consider moving the Hand A CAN tranciever above the gear hole to reduce noise pullution from voltage regulators. Consider also moving connection headers to hand B and/or bottom copper. Do not prioritise.

16.1.4 Mount holes

- Hand: Consider using a bolt with a larger head than the one currently in use, and/or one made of nylon, to reduce the danger of shorting the voltage regulators on installation.
- Shoulder: Increase the size of the mount holes, shift lower left by 0.5-1mm left.
- Torso: All mount hole sizes and positions must be revisited.

- Rail: Consider revisiting the placement of all mount holes for a better fit.
- Use millimeter paper for the measurements instead of/in addition to a caliper.

16.1.5 20 pin connector

- Change the connector socket symbol on either the torso or bogie, and ensure that the pin positions as well as signal names match.

16.1.6 Motor driver assembly

- Switch which driver is connected to which pwm output on torso to reduce number of vias, and try to move the ports such that they aren't on opposing sides of the MCU. Consider the implication on the motor driver assemblies of the other control units.
- Fix the ADC/PWM output error by not using TIM2 or moving the ADC pin.
- Hand DRV1 is connected to the output pin with silk symbol 2. Switch silk symbols.

16.1.7 Optocoupler circuit

- Hand optocoupler circuit must be redesigned for 5V input. Note that the hand optocoupler pin PA1 is not 5V tolerant. Change input pin, or divide output voltage.
- Test the function of the wrist optocoupler. Do not prioritise fixing if result is negative. Ensure that MCU input pin is 5V tolerant.

16.2 Low priority

16.2.1 Hand bulk capacitors and interface pins

- The placement of the motor bulk capacitors and power/CAN/motor pin headers of hand A should be swapped for eaiser mechanical integration with the hand.
- Replace the 1.72mm jumpers with 2.54mm for easier debugging.

16.2.2 Voltage regulators

- Consider replacing LM317 with a switching regulators to increase efficiency.
- Consider redesigning the torso power assembly such that the regulators face downwards. This would reduce the sensitivity to mount point design.
- Replace the indicator LEDs' resistor with a significantly higher value to reduce current draw and eye strain when working on the units.

16.2.3 Encoders

- Consider investigating the original encoder circuit boards to establish the purpose of the remaining pins, and using the result in Mk1.1.
- Consider rotating the bogie encoder connector header 180 degrees for easier mechanical integration.

16.2.4 TVS array

- Find a TVS array which can protect all lines entering the system. Consider using the original model.

16.2.5 Motor driver relay control circuit

- Verify that the circuit works by testing on breadboard.

16.2.6 Motor characterisation

Datasheets for 4 out of 6 motors are missing. Their torque/current draw conversion ratios must be found. Look for datasheets of similar motors, and/or find a way to establish this experimentally.

16.2.7 PCB production

Avoid multiboard production as far as possible when ordering Mk1.1 to reduce the possibility of production errors[8].

16.3 Mk1.1 matrix

Table 26 summarises how the items for further work affect each board design.

[8]This concerns JLCPCB and KiCad details that haven't been discussed

Table 26: A summary of further work

Item/affects unit	**Hand**	**Shoulder**	**Torso**	**Bogie**	**Rail**	**IMU (board)**	**MCU**
USB VBUS voltage divider		X	X				
USB VBUS pin		X	X				X
USB pullup on DM		X	X				
USB pullup transistor		X	X				
IMU 5V and 3.3V lines	X	X				X	
IMU/OPT header design	X	X				X	X
IMU header placement		X					
CAN verification	X	X	X				
CAN/48V connector placement			X				
CAN transciever placement	X						
Nylon bolts	X						
Mount point placement		X	X		X		
Mount point size		X	X				
20 pin connector			X	X			
Motor driver to PWM output	X	X	X				X
Motor driver ADC/PWM							X
Motor driver header silk	X						
Optocoupler 5V	X					X	X
Wrist optocoupler function	X					X	X
Bulk cap/pin header swap	X						
1.27mm jumpers	X						
Switching regulators	X	X	X				
Horisontal voltage regulators			X				
Voltage regulator LED resistor	X	X	X				
Encoder circuits							
Encoder header rotation				X			
TVS array					X		
Motor driver relay control		X	X				
Motor characterisation							
PCB production	X			X	X		

Bibliography

[1] Cadence PCB Design & Analysis. *PCB Design for Pulse Width Modulation Applications*. URL: https://resources.pcb.cadence.com/blog/2019-pcb-design-for-pulse-width-modulation-applications (visited on 8th Dec. 2023).

[2] *HEDS-9000/9100 Two Channel Optical Incremental Encoder Modules*. 2016.

[3] Kristian Blom. *ADC input disturbed by PWM*. URL: https://community.st.com/t5/stm32-mcus-products/adc-input-disturbed-by-pwm/m-p/613657 (visited on 10th Dec. 2023).

[4] Kristian Blom. 'Design og implementering av tester for produserte kretskort'. In: *Unpublished* (2023).

[5] Kristian Blom. *TTK4550 Prosjektoppgave*. URL: https://github.com/kristblo/TTK4550_Prosjektoppgave (visited on 11th Dec. 2023).

[6] *CDSC706-0504C - Surface Mount TVS Diode Array*. 2015.

[7] DIGILENT. *Pull-Down and Pull-Up Resistors*. URL: https://web.archive.org/web/20230601090648/https://learn.digilentinc.com/Documents/176 (visited on 8th Dec. 2023).

[8] Bureau Moeilijke Dingen. *stm32-for-vscode*. URL: https://marketplace.visualstudio.com/items?itemName=bmd.stm32-for-vscode (visited on 9th Dec. 2023).

[9] *Power relay 1 Pole - 5A Medium Load Control JV series*. 2022.

[10] Texas Instruments. *Layout basics for USB designs*. URL: https://www.ti.com/video/6087491555001#transcript-tab (visited on 8th Dec. 2023).

[11] JLCPCB. *PCB Manufacturing & Assembly Capabilities*. URL: https://jlcpcb.com/capabilities/pcb-capabilities (visited on 8th Dec. 2023).

[12] Ultra Librarian. *Ultra Librarian home page*. URL: https://www.ultralibrarian.com/ (visited on 8th Dec. 2023).

[13] United Nations. *Ensure sustainable consumption and production patterns*. URL: https://sdgs.un.org/goals/goal12 (visited on 17th Dec. 2023).

[14] NTNU. *Hva er plagiat?* URL: https://i.ntnu.no/oppgaveskriving/plagiering (visited on 9th Dec. 2023).

[15] NTNU. *Retningslinjer ved behandling av fusk eller forsøk på fusk til eksamen ved Norges teknisk-naturvitenskapelige universitet (NTNU)*. URL: https://i.ntnu.no/c/wiki/get_page_attachment?p_l_id=1307200529&nodeId=1306956301&title=Generelle+lover+og+regler+-+studier&fileName=Fusk%20retningslinjer%20at%20NTNU%20vedtatt%20250619.pdf (visited on 9th Dec. 2023).

[16] *TJA1057 High-speed CAN transceiver*. 2023.

[17] NXP. *MC68HC11F1 Technical summary 8-bit microcontroller*. 1997.

[18] Artem Oppermann. *What is the V-model in software development*. URL: https://builtin.com/software-engineering-perspectives/v-model (visited on 16th Dec. 2023).

[19] ScienceReady HSC Resources. *Operation of a Simple DC Motor*. URL: https://scienceready.com.au/pages/operation-of-a-simple-dc-motor (visited on 16th Dec. 2023).

[20] Sagian. *ORCA manual chapter 2: Installation and setup*. 1995.

[21] Sagian. *ORCA manual part 1: Introduction*. 1995.

[22] *LSM6DSM iNEMO interial module: always-on 3D accelerometer and 3D gyroscope*. 2017.

[23] *STM32F303xD STM32F303xE*. 2016.

[24] ST. *STM product page*. URL: https://www.st.com/en/microcontrollers-microprocessors/stm32-mainstream-mcus.html (visited on 1st Dec. 2023).

[25] ST. *STM32F3 Series*. URL: https://www.st.com/en/microcontrollers-microprocessors/stm32f3-series.html (visited on 14th Dec. 2023).

[26] ST. *STM32G4 Series*. URL: https://www.st.com/en/microcontrollers-microprocessors/stm32g4-series.html (visited on 14th Dec. 2023).

[27] *AN4206: Getting started with STM32F303 series hardware development.* 2015.

[28] *AN4879: Introduction to USB hardware and PCB guidelines using STM32 MCUs.* 2023.

[29] *AN4987: LSM6DSM: always-on 3D accelerometer and 3D gyroscope.* 2018.

[30] *UM1724 User Manual STM32 Nucleo-64 boards (MB1136).* 2020.

[31] *UM1786: Description of STM32F3 HAL and low-layer drivers.* 2018.

[32] *TE Connectivity: 2-1761603-6.* 2023.

[33] *TE Connectivity: 1761681-7.* 2015.

[34] *DRV8251A 4.1-A Brushed DC Motor Driver with Integrated Current Sense and Regulation.* 2022.

[35] *LMx17HV High Voltage Three-Terminal Adjustable Regulator With Overload Protection.* 2015.

[36] *Photologic® Slotted Optical Switch OPB960, OPB970, OPB980, OPB990 Series.* 2019.

[37] *Universal Serial Bus Specification Revision 2.0.* 2000.

Appendix

A Bill of materials

This bill of materials is based on automatically generated reports from KiCad

Table 27: Auxiliaries BOM (Bogie and rail boards)

Id	Designator	Footprint	Quantity	Designation
1	OUT2, OUT1	2-1761603-6_TYC	2	Conn_02x08_Odd_Even
2	H3, H1, H2	MountingHole_3.2mm_M3_DIN965_ Pad_TopBottom	3	MountingHole_Pad
3	ENC0	PinHeader_2x05_P2.54mm_Vertical	1	Conn_02x05_Odd_Even
4	DSUB1	PinHeader_2x07_P2.54mm_Vertical	1	Conn_02x07_Odd_Even
5	H5, H4	MountingHole_2.7mm_M2.5	2	MountingHole
6	END_SWITCH1	PinHeader_1x03_P2.54mm_Vertical	1	Conn_01x03
7	MTR1	70543-0001_MOL	1	Conn_01x02
8	MAIN1	1761681-7_TYC	1	Conn_02x10_Odd_Even
9	D1	LED_0603_1608Metric	1	LED
10	CR1	CDSC706-0504C_BRN	1	CDSC706-0504C

Table 28: IMU BOM (single board)

Id	Designator	Footprint	Quantity	Designation
1	H1,H2,H3	MountingHole_2.7mm_M2.5_DIN965_ Pad_TopBottom	3	MountingHole_Pad
2	R5	R_0805_2012Metric	1	50k
3	C1,C2	C_0603_1608Metric	2	100nF
4	U1	LGA-14L_2P5X3X0P83_STM	1	LSM6DSMTR
5	J2	PinHeader_1x03_P2.54mm_Horizontal	1	Conn_01x03
6	R3,R4	R_0805_2012Metric	2	10k
7	R2	R_Axial_DIN0309_L9.0mm_D3.2mm_P12.70mm_Horizontal	1	5k
8	J1	PinHeader_1x05_P2.54mm_Horizontal	1	Conn_01x05
9	OPT_EN1	PinHeader_1x02_P2.54mm_Horizontal	1	Jumper_2_Open
10	R1	R_Axial_DIN0309_L9.0mm_D3.2mm_P12.70mm_Horizontal	1	3k
11	INT_EN1	PinHeader_1x03_P2.54mm_Horizontal	1	Jumper_2_Open

Table 29: Torso BOM

Id	Designator	Footprint	Quantity	Designation
1	D6, D5, D4, D3	D_T-1_P5.08mm_Horizontal	4	IN4005
2	MAIN1	1761681-7_TYC	1	Conn_02x10_Top_Bottom
3	C4	C_0603_1608Metric	1	4.7muF
4	D2	D_T-1_P5.08mm_Horizontal	1	D_Zener
5	C1, C13, C14, C17, C3, C2, C18, C7, C10, C8, C9	C_0603_1608Metric	11	100nF
6	H5, H3, H4, H1, H2	MountingHole_2.7mm_M2.5	5	MountingHole
7	U1	SOT96-1_SO8_NXP	1	TJA1057GT/3J
8	U3, U4	TO-220-3_Vertical	2	LM317_TO-220
9	RV2	Potentiometer_Bourns_3006P_Horizontal	1	0, 706k
10	RLY_CTL_DBG1, LOW_VLT_DBG1	PinHeader_1x03_P2.54mm_Vertical	2	Conn_01x03
11	R29, R33, R30, R34	R_0805_2012Metric	4	240Ohm
12	USB1	USB_Micro-B_Molex-105017-0001	1	USB_B_Micro
13	C20, C19, C16, C15	CP_Radial_D5.0mm_P2.00mm	4	10muF
14	R2, R13, R27, R19, R15, R6, R5, R36, R18, R22, R7, R35, R31, R28, R11, R24, R20, R23, R12, R14, R17, R10, R9, R21, R25	R_0402_1005Metric	25	40kOhm
15	UART1	PinHeader_1x02_P2.54mm_Vertical	1	Conn_01x02
16	R_IPROPI2, R_IPROPI1	R_0805_2012Metric	2	510Ohm
17	MTR_ASY_DBG1	PinHeader_1x06_P2.54mm_Vertical	1	Conn_01x06
18	C5	C_0603_1608Metric	1	1muF
19	SRL_FLASH1	PinHeader_1x05_P2.54mm_Vertical	1	Conn_01x05
20	D8	LED_0603_1608Metric	1	LED_5V
21	MTR2	70543-0001_MOL	1	705430001
22	XTAL1	XTAL_ABL-16_ABR	1	ABL-16.000MHZ-B2
23	MTR_DRV2, MTR_DRV1	SOIC8_DDA_TEX	2	DRV8251ADDAR
24	JP3	PinHeader_1x02_P2.54mm_Vertical	1	5V_EN
25	D7	LED_0603_1608Metric	1	LED_3V3
26	R4	R_0805_2012Metric	1	340Ohm
27	SW1	SW_Push_1P1T_NO_CK_KSC6xxJ	1	RESET
28	RV1	Potentiometer_Bourns_3006P_Horizontal	1	0, 386k
29	C11, C12	C_0603_1608Metric	2	20pF
30	D1	LED_0603_1608Metric	1	LED
31	RLY_CTRL1	TO-92	1	Q_NIGBT_CGE
32	R3	R_0805_2012Metric	1	30kOhm
33	PWR_CAN_OUT1	TerminalBlock_4Ucon_1x04_P3.50mm_Vertical	1	Conn_01x04
34	JP2	PinHeader_1x02_P2.54mm_Vertical	1	3V3_EN
35	ENC2	PinHeader_2x05_P2.54mm_Vertical	1	Conn_02x05_Counter_Clockwise
36	JV-3S-KT1	JV-3S-KT	1	RELAY
37	R16	R_1206_3216Metric	1	1.5kOhm
38	U2	LQFP-64_10x10mm_P0.5mm	1	STM32F303RETx
39	BOOT0	PinHeader_1x02_P2.54mm_Vertical	1	BOOTMODE
40	R8	R_0805_2012Metric	1	20kOhm
41	R1	R_0805_2012Metric	1	10kOhm
42	CAN_TRM1	R_0603_1608Metric	1	120Ohm
43	C_MTR1, C1_MTR1	CP_Radial_D16.0mm_P7.50mm	2	330muF
44	C6	C_0603_1608Metric	1	10nF

Table 30: Shoulder BOM

Id	Designator	Footprint	Quantity	Designation
1	R32, R2, R13, R26, R27, R19, R15, R6, R5, R18, R22, R7, R31, R11, R24, R20, R23, R12, R14, R17, R10, R9, R21, R25	R_0402_1005Metric	24	40kOhm
2	D6, D5, D4, D3	D_T-1_P5.08mm_Horizontal	4	IN4005
3	D2	D_T-1_P5.08mm_Horizontal	1	D_Zener
4	C1, C13, C14, C17, C3, C2, C18, C7, C10, C8, C9	C_0603_1608Metric	11	100nF
5	U1	SOT96-1_SO8_NXP	1	TJA1057GT/3J
6	U3, U4	TO-220-3_Vertical	2	LM317_TO-220
7	RV2	Potentiometer_Bourns_3006P_Horizontal	1	0, 706k
8	ENC1, ENC2	PinHeader_2x05_P2.54mm_Vertical	2	Conn_02x05_ Counter_Clockwise
9	IMU0_INT1, UART1	PinHeader_1x02_P2.54mm_Vertical	2	Conn_01x02
10	R29, R33, R30, R34	R_0805_2012Metric	4	240Ohm
11	USB1	USB_Micro-B_Molex-105017-0001	1	USB_B_Micro
12	MTR1, MTR2	70543-0001_MOL	2	705430001
13	MTR_ASY_DBG1	PinHeader_1x06_P2.54mm_Vertical	1	Conn_01x06
14	C20, C19, C16, C15	CP_Radial_D5.0mm_P2.00mm	4	10muF
15	R_IPROPI2, R_IPROPI1	R_0805_2012Metric	2	510Ohm
16	SRL_FLASH1	PinHeader_1x05_P2.54mm_Vertical	1	Conn_01x05
17	D8	LED_0603_1608Metric	1	LED_5V
18	H3, H4, H1, H2	MountingHole_2.7mm_M2.5	4	MountingHole
19	C_MTR2	CP_Radial_D10.0mm_P5.00mm	1	220muF
20	XTAL1	XTAL_ABL-16_ABR	1	ABL-16.000MHZ-B2
21	MTR_DRV2, MTR_DRV1	SOIC8_DDA_TEX	2	DRV8251ADDAR
22	R28	R_0805_2012Metric	1	50kOhm
23	JP3	PinHeader_1x02_P2.54mm_Vertical	1	5V_EN
24	D7	LED_0603_1608Metric	1	LED_3V3
25	R4	R_0805_2012Metric	1	340Ohm
26	SW1	SW_Push_1P1T_NO_CK_KSC6xxJ	1	RESET
27	RV1	Potentiometer_Bourns_3006P_Horizontal	1	0, 386k
28	C11, C12	C_0603_1608Metric	2	20pF
29	LOW_VLT_DBG1, RLY_CTL_DBG1	PinHeader_1x03_P2.54mm_Vertical	2	Conn_01x03
30	D1	LED_0603_1608Metric	1	LED
31	RLY_CTRL1	TO-92	1	Q_NIGBT_CGE
32	R3	R_0805_2012Metric	1	30kOhm
33	PWR_CAN_OUT1, PWR_CAN_IN1	TerminalBlock_4Ucon_1x04_ P3.50mm_Vertical	2	Conn_01x04
34	JP2	PinHeader_1x02_P2.54mm_Vertical	1	3V3_EN
35	JV-3S-KT1	JV-3S-KT	1	RELAY
36	R16	R_1206_3216Metric	1	1.5kOhm
37	U2	LQFP-64_10x10mm_P0.5mm	1	STM32F303RETx
38	BOOT0	PinHeader_1x02_P2.54mm_Vertical	1	BOOTMODE
39	R8	R_0805_2012Metric	1	20kOhm
40	R1	R_0805_2012Metric	1	10kOhm
41	CAN_TRM1	R_0603_1608Metric	1	120Ohm
42	C_MTR1	CP_Radial_D16.0mm_P7.50mm	1	330muF
43	IMU0	PinHeader_1x04_P2.54mm_Vertical	1	Conn_01x04
44	C4	C_0603_1608Metric	1	4.7muF
45	C5	C_0603_1608Metric	1	1muF
46	C6	C_0603_1608Metric	1	10nF

Table 31: Hand BOM

Id	Designator	Footprint	Quantity	Designation
1	D2	D_T-1_P5.08mm_Horizontal	1	D_Zener
2	C1, C21, C17, C3, C2, C22, C18, C7, C13, C14, C10, C8, C9	C_0603_1608Metric	13	100nF
3	U1	SOT96-1_SO8_NXP	1	TJA1057GT/3J
4	U3, U4	TO-220-3_Horizontal_TabDown	2	LM317_TO-220
5	J1, J2	PinHeader_1x05_P2.54mm_Vertical	2	Conn_01x05
6	R29, R33, R30, R34	R_0805_2012Metric	4	240Ohm
7	D5, D3, D6, D4	D_T-1_P5.08mm_Horizontal	4	IN4005
8	IMU_INT1	PinHeader_2x02_P2.54mm_Horizontal	1	Conn_02x02_ Counter_Clockwise
9	C_MTR6, C_MTR3, C_MTR2, C_MTR5, C_MTR1, C_MTR4	CP_Elec_6.3x7.7	6	47muF
10	PWR_CAN_IMU1	PinHeader_2x04_P2.54mm_Horizontal	1	Conn_02x04_Odd_Even
11	R2	R_0402_1005Metric	1	4k2
12	R_IPROPI2, R_IPROPI1	R_0805_2012Metric	2	510Ohm
13	MTR1	PinHeader_2x04_P2.54mm_Horizontal	1	Conn_02x04_ Counter_Clockwise
14	R27, R31	R_0805_2012Metric	2	10k
15	R_EN1	R_0805_2012Metric	1	40k
16	R15, R5, R7, R11, R20, R23, R3	R_0402_1005Metric	7	40kOhm
17	D8	LED_0603_1608Metric	1	LED_5V
18	H3, H4	MountingHole_3.2mm_M3	2	MountingHole
19	XTAL1	XTAL_ABL-16_ABR	1	ABL-16.000MHZ-B2
20	MTR_DRV2, MTR_DRV1	SOIC8_DDA_TEX	2	DRV8251ADDAR
21	JP3	PinHeader_1x02_P1.27mm_Horizontal	1	5V_EN
22	U5	LGA-14L_2P5X3X0P83_STM	1	LSM6DSMTR
23	D7	LED_0603_1608Metric	1	LED_3V3
24	R4	R_0805_2012Metric	1	340Ohm
25	C11, C12	C_0603_1608Metric	2	20pF
26	D1	LED_0603_1608Metric	1	LED
27	SRL_FLS_URT1	PinHeader_2x05_P2.54mm_Horizontal	1	Conn_02x05_Odd_Even
28	JP2	PinHeader_1x02_P1.27mm_Horizontal	1	3V3_EN
29	U2	LQFP-64_10x10mm_P0.5mm	1	STM32F303RETx
30	BOOT0	PinHeader_1x02_P1.27mm_Horizontal	1	BOOTMODE
31	R1	R_0805_2012Metric	1	10kOhm
32	H1, H2	MountingHole_2.2mm_M2	2	MountingHole
33	CAN_TRM1	R_0603_1608Metric	1	120Ohm
34	R35, R28, R32	R_0805_2012Metric	3	50kOhm
35	J3	PinHeader_1x02_P1.27mm_Horizontal	1	Conn_01x02
36	C4	C_0603_1608Metric	1	4.7muF
37	RV2	Potentiometer_Bourns_3006P_Horizontal	1	0, 706k
38	CON_B1, CON_A1	PinHeader_1x03_P2.54mm_Vertical	2	Conn_01x03
39	OPT1	OPB971N51	1	
40	C5	C_0603_1608Metric	1	1muF
41	C16, C15	CP_Axial_L10.0mm_D4.5mm_ P15.00mm_Horizontal	2	10muF
42	RV1	Potentiometer_Bourns_3006P_Horizontal	1	0, 386k
43	C6	C_0603_1608Metric	1	10nF

B STM32F303 configuration report

Starts next page


STM32
CubeMX

1. Description

1.1. Project

Project Name	STM_config
Board Name	custom
Generated with:	STM32CubeMX 6.9.1
Date	09/15/2023

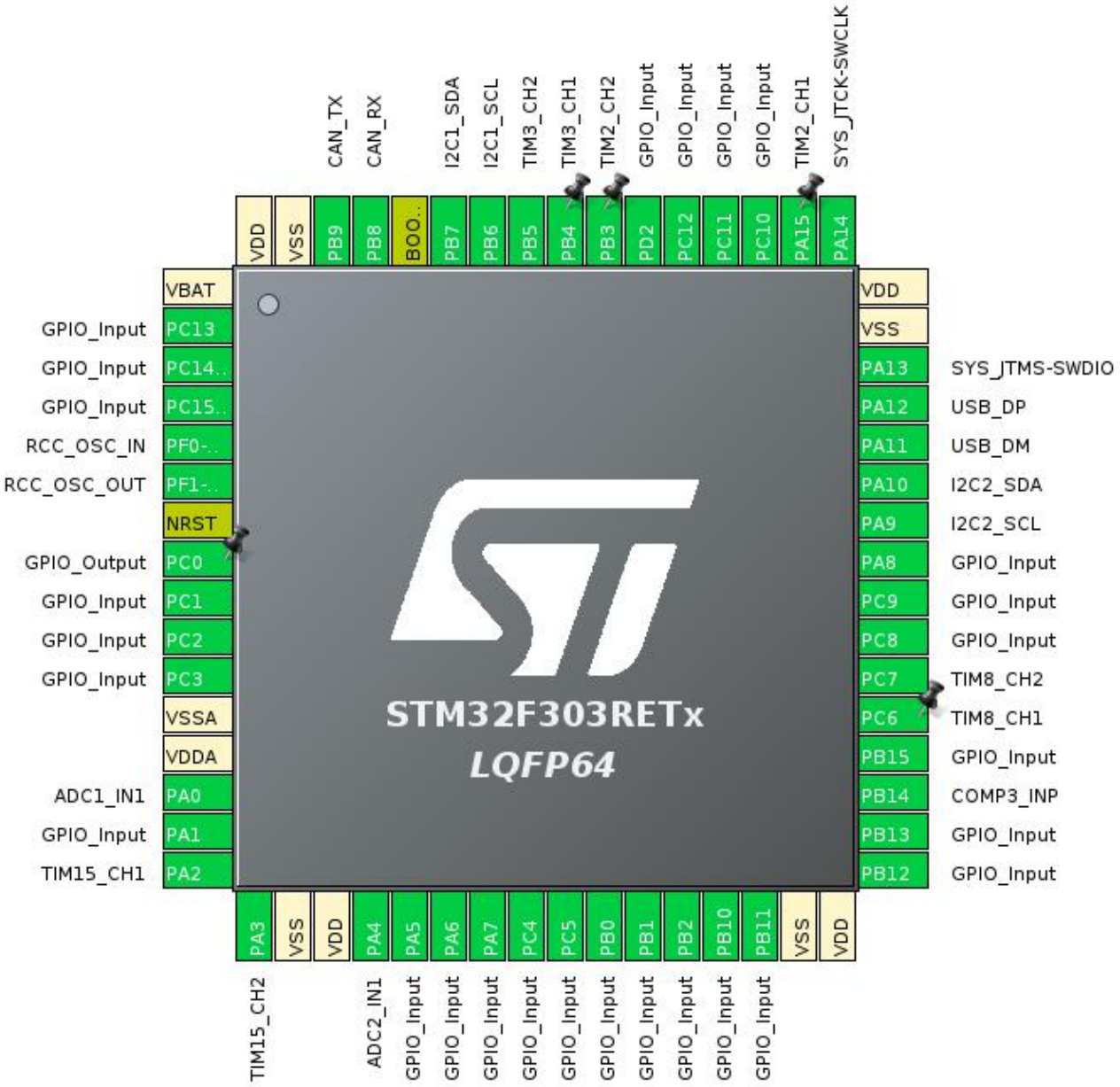
1.2. MCU

MCU Series	STM32F3
MCU Line	STM32F303
MCU name	STM32F303RETx
MCU Package	LQFP64
MCU Pin number	64

1.3. Core(s) information

Core(s)	Arm Cortex-M4

2. Pinout Configuration



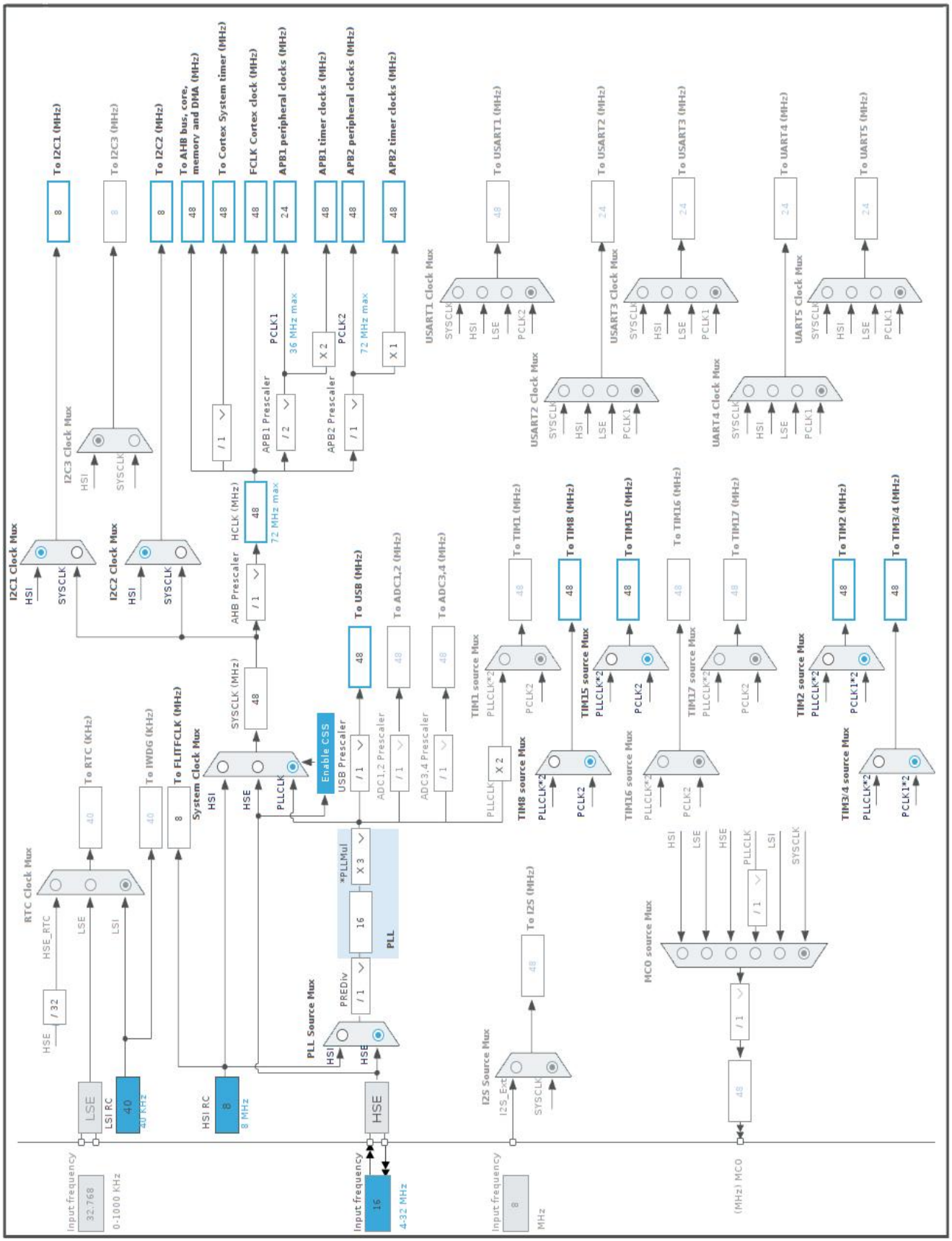
3. Pins Configuration

Pin Number LQFP64	Pin Name (function after reset)	Pin Type	Alternate Function(s)	Label
1	VBAT	Power		
2	PC13 *	I/O	GPIO_Input	
3	PC14-OSC32_IN *	I/O	GPIO_Input	
4	PC15-OSC32_OUT *	I/O	GPIO_Input	
5	PF0-OSC_IN	I/O	RCC_OSC_IN	
6	PF1-OSC_OUT	I/O	RCC_OSC_OUT	
7	NRST	Reset		
8	PC0 *	I/O	GPIO_Output	
9	PC1 *	I/O	GPIO_Input	
10	PC2 *	I/O	GPIO_Input	
11	PC3 *	I/O	GPIO_Input	
12	VSSA	Power		
13	VDDA	Power		
14	PA0	I/O	ADC1_IN1	
15	PA1 *	I/O	GPIO_Input	
16	PA2	I/O	TIM15_CH1	
17	PA3	I/O	TIM15_CH2	
18	VSS	Power		
19	VDD	Power		
20	PA4	I/O	ADC2_IN1	
21	PA5 *	I/O	GPIO_Input	
22	PA6 *	I/O	GPIO_Input	
23	PA7 *	I/O	GPIO_Input	
24	PC4 *	I/O	GPIO_Input	
25	PC5 *	I/O	GPIO_Input	
26	PB0 *	I/O	GPIO_Input	
27	PB1 *	I/O	GPIO_Input	
28	PB2 *	I/O	GPIO_Input	
29	PB10 *	I/O	GPIO_Input	
30	PB11 *	I/O	GPIO_Input	
31	VSS	Power		
32	VDD	Power		
33	PB12 *	I/O	GPIO_Input	
34	PB13 *	I/O	GPIO_Input	
35	PB14	I/O	COMP3_INP	
36	PB15 *	I/O	GPIO_Input	

Pin Number LQFP64	Pin Name (function after reset)	Pin Type	Alternate Function(s)	Label
37	PC6	I/O	TIM8_CH1	
38	PC7	I/O	TIM8_CH2	
39	PC8 *	I/O	GPIO_Input	
40	PC9 *	I/O	GPIO_Input	
41	PA8 *	I/O	GPIO_Input	
42	PA9	I/O	I2C2_SCL	
43	PA10	I/O	I2C2_SDA	
44	PA11	I/O	USB_DM	
45	PA12	I/O	USB_DP	
46	PA13	I/O	SYS_JTMS-SWDIO	
47	VSS	Power		
48	VDD	Power		
49	PA14	I/O	SYS_JTCK-SWCLK	
50	PA15	I/O	TIM2_CH1	
51	PC10 *	I/O	GPIO_Input	
52	PC11 *	I/O	GPIO_Input	
53	PC12 *	I/O	GPIO_Input	
54	PD2 *	I/O	GPIO_Input	
55	PB3	I/O	TIM2_CH2	
56	PB4	I/O	TIM3_CH1	
57	PB5	I/O	TIM3_CH2	
58	PB6	I/O	I2C1_SCL	
59	PB7	I/O	I2C1_SDA	
60	BOOT0	Boot		
61	PB8	I/O	CAN_RX	
62	PB9	I/O	CAN_TX	
63	VSS	Power		
64	VDD	Power		

* The pin is affected with an I/O function

4. Clock Tree Configuration



5. Software Project

5.1. Project Settings

Name	Value
Project Name	STM_config
Project Folder	/home/ov_robotarm/TTK4550_Prosjektoppgave/documentation/STM_config
Toolchain / IDE	EWARM V8.32
Firmware Package Name and Version	STM32Cube FW_F3 V1.11.4
Application Structure	Advanced
Generate Under Root	No
Do not generate the main()	No
Minimum Heap Size	0x200
Minimum Stack Size	0x400

5.2. Code Generation Settings

Name	Value
STM32Cube MCU packages and embedded software	Copy all used libraries into the project folder
Generate peripheral initialization as a pair of '.c/.h' files	No
Backup previously generated files when re-generating	No
Keep User Code when re-generating	Yes
Delete previously generated files when not re-generated	Yes
Set all free pins as analog (to optimize the power consumption)	No
Enable Full Assert	No

5.3. Advanced Settings - Generated Function Calls

Rank	Function Name	Peripheral Instance Name
1	SystemClock_Config	RCC
2	MX_GPIO_Init	GPIO
3	MX_DMA_Init	DMA
4	MX_ADC1_Init	ADC1
5	MX_ADC2_Init	ADC2
6	MX_CAN_Init	CAN
7	MX_USB_PCD_Init	USB
8	MX_COMP3_Init	COMP3
9	MX_TIM2_Init	TIM2
10	MX_TIM3_Init	TIM3
11	MX_TIM8_Init	TIM8

Rank	Function Name	Peripheral Instance Name
12	MX_TIM15_Init	TIM15
13	MX_I2C1_Init	I2C1
14	MX_I2C2_Init	I2C2

1. Power Consumption Calculator report

1.1. Microcontroller Selection

Series	STM32F3
Line	STM32F303
MCU	STM32F303RETx
Datasheet	DS10362_Rev5

1.2. Parameter Selection

Temperature	25
Vdd	3.6

1.3. Battery Selection

Battery	Li-SOCL2(A3400)
Capacity	3400.0 mAh
Self Discharge	0.08 %/month
Nominal Voltage	3.6 V
Max Cont Current	100.0 mA
Max Pulse Current	200.0 mA
Cells in series	1
Cells in parallel	1

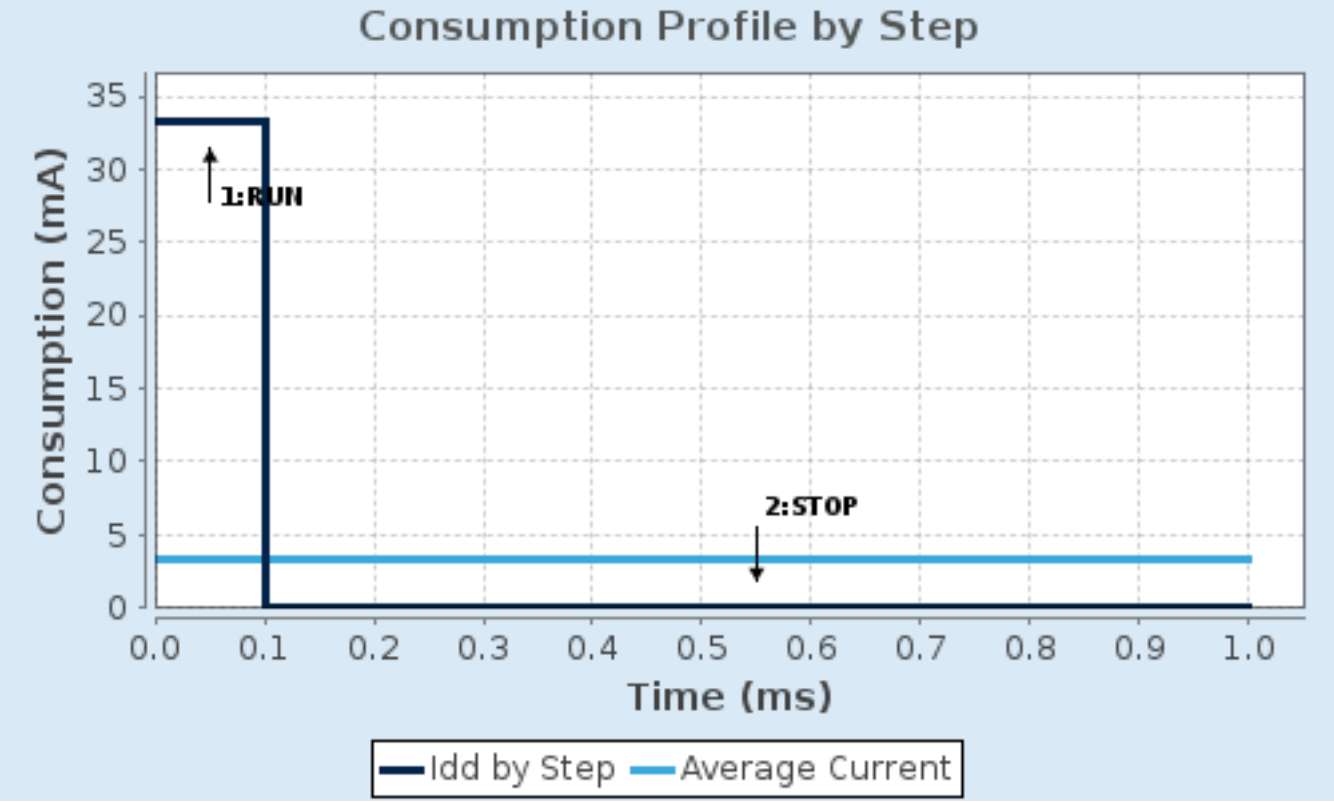
1.4. Sequence

Step	Step1	Step2
Mode	RUN	STOP
Vdd	3.6	3.6
Voltage Source	Battery	Battery
Range	No Scale	No Scale
Fetch Type	FLASH	n/a
CPU Frequency	72 MHz	0 Hz
Clock Configuration	HSEBYP PLL	Regulator_LP
Clock Source Frequency	8 MHz	0 Hz
Peripherals		
Additional Cons.	0 mA	0 mA
Average Current	33.24 mA	9.8 µA
Duration	0.1 ms	0.9 ms
DMIPS	63.0	0.0
Ta Max	99.5	105
Category	In DS Table	In DS Table

1.5. Results

Sequence Time	1 ms	Average Current	3.33 mA
Battery Life	1 month, 12 days, 1 hour	Average DMIPS	63.0 DMIPS

1.6. Chart


Consumption Profile by Step
Consumption (mA)
35
30
25
20
15
10
5
0
1:RUN
2:STOP
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1.0
Time (ms)
Idd by Step
Average Current

2. Peripherals and Middlewares Configuration

2.1. ADC1

IN1: IN1 Single-ended

2.1.1. Parameter Settings:

ADCs_Common_Settings:

Mode	Independent mode

ADC_Settings:

Clock Prescaler	**Synchronous clock mode divided by 2 ***
Resolution	ADC 12-bit resolution
Data Alignment	Right alignment
Scan Conversion Mode	Disabled
Continuous Conversion Mode	Disabled
Discontinuous Conversion Mode	Disabled
DMA Continuous Requests	Disabled
End Of Conversion Selection	End of single conversion
Overrun behaviour	Overrun data overwritten
Low Power Auto Wait	Disabled

ADC_Regular_ConversionMode:

Enable Regular Conversions	Enable
Number Of Conversion	1
External Trigger Conversion Source	Regular Conversion launched by software
External Trigger Conversion Edge	None
SequencerNbRanks	1
Rank	1
Channel	Channel 1
Sampling Time	1.5 Cycles
Offset Number	No offset
Offset	0

ADC_Injected_ConversionMode:

Enable Injected Conversions	Enable
Number Of Conversions	0

Analog Watchdog 1:

Enable Analog WatchDog1 Mode	false

Analog Watchdog 2:

Enable Analog WatchDog2 Mode	false

Analog Watchdog 3:

Enable Analog WatchDog3 Mode	false

2.2. ADC2

IN1: IN1 Single-ended

2.2.1. Parameter Settings:

ADCs_Common_Settings:

Mode	Independent mode

ADC_Settings:

Clock Prescaler	**Synchronous clock mode divided by 2 ***
Resolution	ADC 12-bit resolution
Data Alignment	Right alignment
Scan Conversion Mode	Disabled
Continuous Conversion Mode	Disabled
Discontinuous Conversion Mode	Disabled
DMA Continuous Requests	Disabled
End Of Conversion Selection	End of single conversion
Overrun behaviour	Overrun data overwritten
Low Power Auto Wait	Disabled

ADC_Regular_ConversionMode:

Enable Regular Conversions	Enable
Number Of Conversion	1
External Trigger Conversion Source	Regular Conversion launched by software
External Trigger Conversion Edge	None
SequencerNbRanks	1
Rank	1
Channel	Channel 1
Sampling Time	1.5 Cycles
Offset Number	No offset
Offset	0

ADC_Injected_ConversionMode:

Enable Injected Conversions	Enable
Number Of Conversions	0

Analog Watchdog 1:

Enable Analog WatchDog1 Mode	false

Analog Watchdog 2:

Enable Analog WatchDog2 Mode	false

Analog Watchdog 3:

Enable Analog WatchDog3 Mode	false

2.3. CAN

mode: Activated

2.3.1. Parameter Settings:

Bit Timings Parameters:

Prescaler (for Time Quantum)	16
Time Quantum	**666.6666666666666 ***
Time Quanta in Bit Segment 1	1 Time
Time Quanta in Bit Segment 2	1 Time
Time for one Bit	**2000 ***
Baud Rate	**500000 ***
ReSynchronization Jump Width	1 Time

Basic Parameters:

Time Triggered Communication Mode	Disable
Automatic Bus-Off Management	Disable
Automatic Wake-Up Mode	Disable
Automatic Retransmission	Disable
Receive Fifo Locked Mode	Disable
Transmit Fifo Priority	Disable

Advanced Parameters:

Operating Mode	Normal

2.4. COMP3

mode: Input [+]

Input [-]: Internal VRef

2.4.1. Parameter Settings:

Basic Parameters:

Interrupt Trigger Mode	None
Blanking Source	None

Output Parameters:

Output Polarity	Not Inverted
Output Internal Selection	None

2.5. I2C1

I2C: I2C

2.5.1. Parameter Settings:

Timing configuration:

I2C Speed Mode	Standard Mode
I2C Speed Frequency (KHz)	100
Rise Time (ns)	0
Fall Time (ns)	0
Coefficient of Digital Filter	0
Analog Filter	Enabled
Timing	0x2000090E

Slave Features:

Clock No Stretch Mode	Disabled
General Call Address Detection	Disabled
Primary Address Length selection	7-bit
Dual Address Acknowledged	Disabled
Primary slave address	0

2.6. I2C2

I2C: I2C

2.6.1. Parameter Settings:

Timing configuration:

I2C Speed Mode	Standard Mode
I2C Speed Frequency (KHz)	100
Rise Time (ns)	0
Fall Time (ns)	0
Coefficient of Digital Filter	0
Analog Filter	Enabled
Timing	0x2000090E

Slave Features:

Clock No Stretch Mode	Disabled
General Call Address Detection	Disabled
Primary Address Length selection	7-bit
Dual Address Acknowledged	Disabled
Primary slave address	0

2.7. RCC

High Speed Clock (HSE): Crystal/Ceramic Resonator

2.7.1. Parameter Settings:

System Parameters:

VDD voltage (V)	3.3
Prefetch Buffer	Enabled
Flash Latency(WS)	1 WS (2 CPU cycle)

RCC Parameters:

HSI Calibration Value	16
HSE Startup Timout Value (ms)	100
LSE Startup Timout Value (ms)	5000

2.8. SYS

Debug: Serial Wire

Timebase Source: SysTick

2.9. TIM2

Clock Source : Internal Clock

Channel1: PWM Generation CH1

Channel2: PWM Generation CH2

2.9.1. Parameter Settings:

Counter Settings:

Prescaler (PSC - 16 bits value)	0
Counter Mode	Up
Counter Period (AutoReload Register - 32 bits value)	4294967295
Internal Clock Division (CKD)	No Division
auto-reload preload	Disable

Trigger Output (TRGO) Parameters:

Master/Slave Mode (MSM bit)	Disable (Trigger input effect not delayed)
Trigger Event Selection TRGO	Reset (UG bit from TIMx_EGR)

Clear Input:

Clear Input Source	Disable

PWM Generation Channel 1:

Mode	PWM mode 1
Pulse (32 bits value)	0
Output compare preload	Enable
Fast Mode	Disable
CH Polarity	High

PWM Generation Channel 2:

Mode	PWM mode 1
Pulse (32 bits value)	0
Output compare preload	Enable
Fast Mode	Disable
CH Polarity	High

2.10. TIM3

Combined Channels: Encoder Mode

2.10.1. Parameter Settings:

Counter Settings:

Prescaler (PSC - 16 bits value)	0
Counter Mode	Up
Counter Period (AutoReload Register - 16 bits value)	65535
Internal Clock Division (CKD)	No Division
auto-reload preload	Disable

Trigger Output (TRGO) Parameters:

Master/Slave Mode (MSM bit)	Disable (Trigger input effect not delayed)
Trigger Event Selection TRGO	Reset (UG bit from TIMx_EGR)

Encoder:

Encoder Mode	Encoder Mode TI1
____ Parameters for Channel 1 ____	
Polarity	Rising Edge
IC Selection	Direct
Prescaler Division Ratio	No division
Input Filter	0
____ Parameters for Channel 2 ____	
Polarity	Rising Edge
IC Selection	Direct
Prescaler Division Ratio	No division
Input Filter	0

2.11. TIM8

Combined Channels: Encoder Mode

2.11.1. Parameter Settings:

Counter Settings:

Prescaler (PSC - 16 bits value)	0
Counter Mode	Up
Counter Period (AutoReload Register - 16 bits value)	65535
Internal Clock Division (CKD)	No Division
Repetition Counter (RCR - 16 bits value)	0
auto-reload preload	Disable

Trigger Output (TRGO) Parameters:

Master/Slave Mode (MSM bit)	Disable (Trigger input effect not delayed)
Trigger Event Selection TRGO	Reset (UG bit from TIMx_EGR)
Trigger Event Selection TRGO2	Reset (UG bit from TIMx_EGR)

Encoder:

Encoder Mode	Encoder Mode TI1
____ Parameters for Channel 1 ____	
Polarity	Rising Edge
IC Selection	Direct
Prescaler Division Ratio	No division
Input Filter	0
____ Parameters for Channel 2 ____	
Polarity	Rising Edge
IC Selection	Direct
Prescaler Division Ratio	No division
Input Filter	0

2.12. TIM15

Channel1: PWM Generation CH1

Channel2: PWM Generation CH2

2.12.1. Parameter Settings:

Counter Settings:

Prescaler (PSC - 16 bits value)	0
Counter Mode	Up
Counter Period (AutoReload Register - 16 bits value)	65535
Internal Clock Division (CKD)	No Division

Repetition Counter (RCR - 8 bits value)	0
auto-reload preload	Disable

Trigger Output (TRGO) Parameters:

Master/Slave Mode (MSM bit)	Disable (Trigger input effect not delayed)
Trigger Event Selection	Reset (UG bit from TIMx_EGR)

Break And Dead Time management - BRK Configuration:

BRK State	Disable
BRK Polarity	High
BRK Filter (4 bits value)	0

Break And Dead Time management - Output Configuration:

Automatic Output State	Disable
Off State Selection for Run Mode (OSSR)	Disable
Off State Selection for Idle Mode (OSSI)	Disable
Lock Configuration	Off

PWM Generation Channel 1:

Mode	PWM mode 1
Pulse (16 bits value)	0
Output compare preload	Enable
Fast Mode	Disable
CH Polarity	High
CH Idle State	Reset

PWM Generation Channel 2:

Mode	PWM mode 1
Pulse (16 bits value)	0
Output compare preload	Enable
Fast Mode	Disable
CH Polarity	High
CH Idle State	Reset

2.13. USB

mode: Device (FS)

2.13.1. Parameter Settings:

Basic Parameters:

Speed	Full Speed 12MBit/s
Physical interface	Internal Phy

Power Parameters:

Low Power	Disabled
Link Power Management	Disabled

*** User modified value**

3. System Configuration

3.1. GPIO configuration

IP	Pin	Signal	GPIO mode	GPIO pull/up pull down	Max Speed	User Label
ADC1	PA0	ADC1_IN1	Analog mode	No pull-up and no pull-down	n/a	
ADC2	PA4	ADC2_IN1	Analog mode	No pull-up and no pull-down	n/a	
CAN	PB8	CAN_RX	Alternate Function Push Pull	No pull-up and no pull-down	**High ***	
	PB9	CAN_TX	Alternate Function Push Pull	No pull-up and no pull-down	**High ***	
COMP3	PB14	COMP3_INP	Analog mode	No pull-up and no pull-down	n/a	
I2C1	PB6	I2C1_SCL	Alternate Function Open Drain	No pull-up and no pull-down	**High ***	
	PB7	I2C1_SDA	Alternate Function Open Drain	No pull-up and no pull-down	**High ***	
I2C2	PA9	I2C2_SCL	Alternate Function Open Drain	No pull-up and no pull-down	**High ***	
	PA10	I2C2_SDA	Alternate Function Open Drain	No pull-up and no pull-down	**High ***	
RCC	PF0-OSC_IN	RCC_OSC_IN	n/a	n/a	n/a	
	PF1-OSC_OUT	RCC_OSC_OUT	n/a	n/a	n/a	
SYS	PA13	SYS_JTMS-SWDIO	n/a	n/a	n/a	
	PA14	SYS_JTCK-SWCLK	n/a	n/a	n/a	
TIM2	PA15	TIM2_CH1	Alternate Function Push Pull	No pull-up and no pull-down	Low	
	PB3	TIM2_CH2	Alternate Function Push Pull	No pull-up and no pull-down	Low	
TIM3	PB4	TIM3_CH1	Alternate Function Push Pull	No pull-up and no pull-down	Low	
	PB5	TIM3_CH2	Alternate Function Push Pull	No pull-up and no pull-down	Low	
TIM8	PC6	TIM8_CH1	Alternate Function Push Pull	No pull-up and no pull-down	Low	
	PC7	TIM8_CH2	Alternate Function Push Pull	No pull-up and no pull-down	Low	
TIM15	PA2	TIM15_CH1	Alternate Function Push Pull	No pull-up and no pull-down	Low	
	PA3	TIM15_CH2	Alternate Function Push Pull	No pull-up and no pull-down	Low	
USB	PA11	USB_DM	n/a	n/a	n/a	
	PA12	USB_DP	n/a	n/a	n/a	
GPIO	PC13	GPIO_Input	Input mode	No pull-up and no pull-down	n/a	
	PC14-OSC32_IN	GPIO_Input	Input mode	No pull-up and no pull-down	n/a	
	PC15-OSC32_OUT	GPIO_Input	Input mode	No pull-up and no pull-down	n/a	
	PC0	GPIO_Output	Output Push Pull	No pull-up and no pull-down	Low	
	PC1	GPIO_Input	Input mode	No pull-up and no pull-down	n/a	

IP	Pin	Signal	GPIO mode	GPIO pull/up pull down	Max Speed	User Label
	PC2	GPIO_Input	Input mode	No pull-up and no pull-down	n/a	
	PC3	GPIO_Input	Input mode	No pull-up and no pull-down	n/a	
	PA1	GPIO_Input	Input mode	No pull-up and no pull-down	n/a	
	PA5	GPIO_Input	Input mode	No pull-up and no pull-down	n/a	
	PA6	GPIO_Input	Input mode	No pull-up and no pull-down	n/a	
	PA7	GPIO_Input	Input mode	No pull-up and no pull-down	n/a	
	PC4	GPIO_Input	Input mode	No pull-up and no pull-down	n/a	
	PC5	GPIO_Input	Input mode	No pull-up and no pull-down	n/a	
	PB0	GPIO_Input	Input mode	No pull-up and no pull-down	n/a	
	PB1	GPIO_Input	Input mode	No pull-up and no pull-down	n/a	
	PB2	GPIO_Input	Input mode	No pull-up and no pull-down	n/a	
	PB10	GPIO_Input	Input mode	No pull-up and no pull-down	n/a	
	PB11	GPIO_Input	Input mode	No pull-up and no pull-down	n/a	
	PB12	GPIO_Input	Input mode	No pull-up and no pull-down	n/a	
	PB13	GPIO_Input	Input mode	No pull-up and no pull-down	n/a	
	PB15	GPIO_Input	Input mode	No pull-up and no pull-down	n/a	
	PC8	GPIO_Input	Input mode	No pull-up and no pull-down	n/a	
	PC9	GPIO_Input	Input mode	No pull-up and no pull-down	n/a	
	PA8	GPIO_Input	Input mode	No pull-up and no pull-down	n/a	
	PC10	GPIO_Input	Input mode	No pull-up and no pull-down	n/a	
	PC11	GPIO_Input	Input mode	No pull-up and no pull-down	n/a	
	PC12	GPIO_Input	Input mode	No pull-up and no pull-down	n/a	
	PD2	GPIO_Input	Input mode	No pull-up and no pull-down	n/a	

3.2. DMA configuration

DMA request	Stream	Direction	Priority
ADC1	DMA1_Channel1	Peripheral To Memory	Low

ADC1: DMA1_Channel1 DMA request Settings:

Mode:	Normal
Peripheral Increment:	Disable
Memory Increment:	**Enable ***
Peripheral Data Width:	Half Word
Memory Data Width:	Half Word

3.3. NVIC configuration

3.3.1. NVIC

Interrupt Table	Enable	Preenmption Priority	SubPriority
Non maskable interrupt	true	0	0
Hard fault interrupt	true	0	0
Memory management fault	true	0	0
Pre-fetch fault, memory access fault	true	0	0
Undefined instruction or illegal state	true	0	0
System service call via SWI instruction	true	0	0
Debug monitor	true	0	0
Pendable request for system service	true	0	0
System tick timer	true	15	0
DMA1 channel1 global interrupt	true	0	0
PVD interrupt through EXTI line 16	unused		
Flash global interrupt	unused		
RCC global interrupt	unused		
ADC1 and ADC2 interrupts	unused		
USB high priority or CAN_TX interrupts	unused		
USB low priority or CAN_RX0 interrupts	unused		
CAN_RX1 interrupt	unused		
CAN_SCE interrupt	unused		
TIM1 break and TIM15 interrupts	unused		
TIM2 global interrupt	unused		
TIM3 global interrupt	unused		
I2C1 event global interrupt / I2C1 wake-up interrupt through EXTI line 23	unused		
I2C1 error interrupt	unused		
I2C2 event global interrupt / I2C2 wake-up interrupt through EXTI line 24	unused		
I2C2 error interrupt	unused		
TIM8 break global interrupt	unused		
TIM8 update interrupt	unused		
TIM8 trigger and commutation interrupt	unused		
TIM8 capture compare interrupt	unused		
COMP1, COMP2 and COMP3 interrupts through EXTI lines 21, 22 and 29	unused		
USB high priority interrupt remap	unused		
USB low priority interrupt remap	unused		
Floating point unit interrupt	unused		

3.3.2. NVIC Code generation

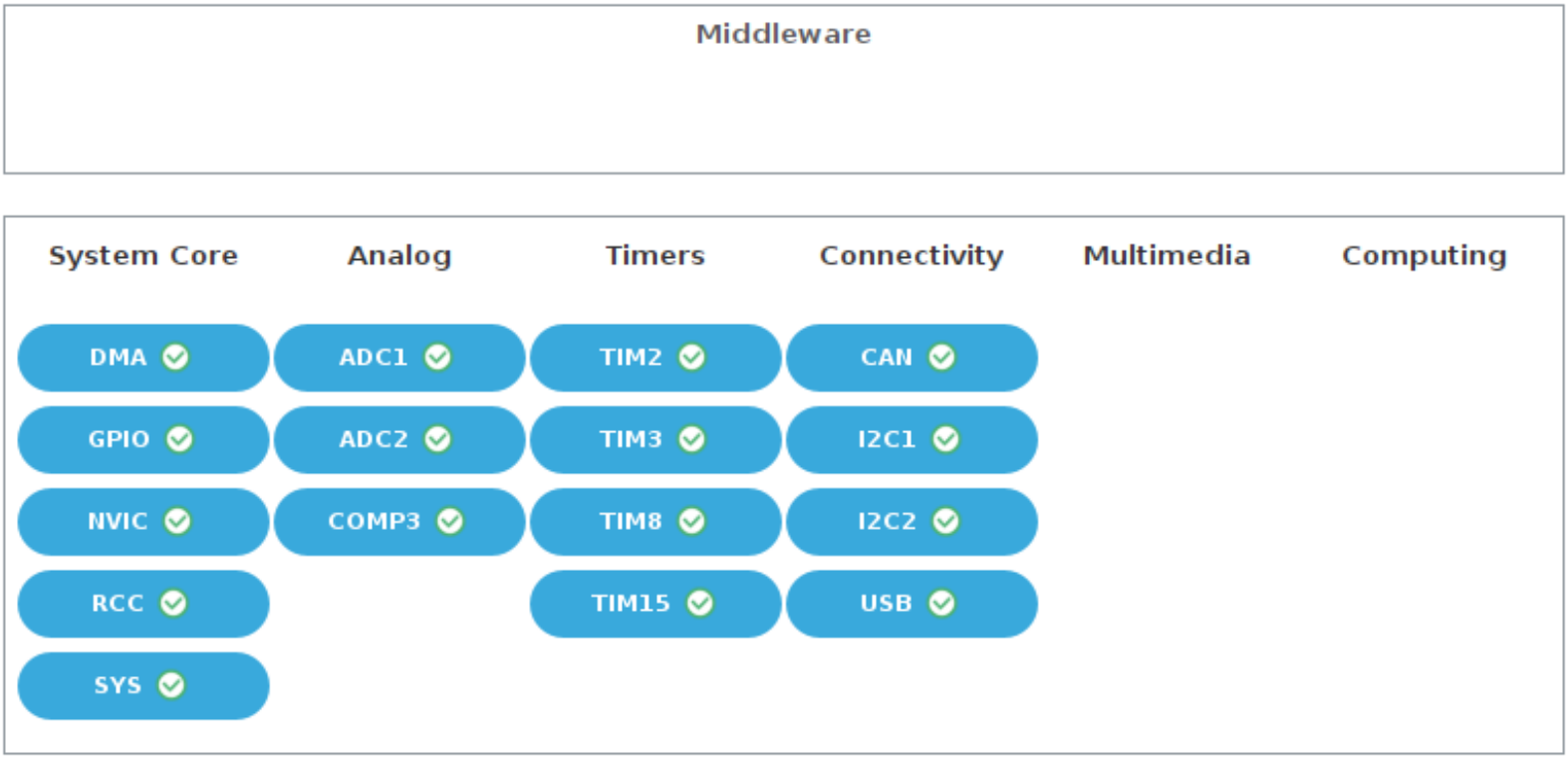
Enabled interrupt Table	Select for init sequence ordering	Generate IRQ handler	Call HAL handler
Non maskable interrupt	false	true	false
Hard fault interrupt	false	true	false
Memory management fault	false	true	false
Pre-fetch fault, memory access fault	false	true	false
Undefined instruction or illegal state	false	true	false
System service call via SWI instruction	false	true	false
Debug monitor	false	true	false
Pendable request for system service	false	true	false
System tick timer	false	true	true
DMA1 channel1 global interrupt	false	true	true

*** User modified value**

4. System Views

4.1. Category view

4.1.1. Current



C TTK8 Report: Testing and validation

Starts next page

NTNU
Kunnskap for en bedre verden

Institutt for Teknisk Kybernetikk

TTK8 - Konstruksjon av innebygde systemer

Design og implementering av tester for produserte kretskort

Forfatter:
Kristian Blom

22.11.2023

Innhold

Figurer **iii**

Tabeller **iv**

1 Forkortelser og oversettelser **1**

2 Introduksjon **2**

2.1 Bakgrunn og motivasjon 2

2.2 Omfang 2

3 Systembeskrivelse **3**

3.1 Forklaring i kontekst av robotarmen 4

3.1.1 Nærmere beskrivelse av armen 6

4 Design av testenheter **7**

4.1 Komponentvalg 7

4.2 Kretsdesign 8

4.3 Produksjon av utbrytningskort 10

4.3.1 Design av utbrytningskort 10

4.3.2 Fremkalling i etsebad 11

4.3.3 Lodding og boring av hull 12

4.4 Sammenstilling av testmoduler, resultater 13

4.4.1 Lavspent spenningsregulering 14

4.4.2 Motordriver 15

4.4.3 Treghetssensor 16

4.4.4 Mikroprosessor 17

5 Tester **18**

5.1 Beskrivelse av testnumre 18

5.2 Beskrivelse av programvare 18

5.3 Designtester 18

5.3.1 TE1.1D: DC-buck 48-3V3 18

5.3.2 TE2.1D: DC-buck 48-5V 18

5.3.3 TC1.1D: Motordriver PWM-signal 19

5.3.4 TC1.2D: Motordriver PWM control 19

5.3.5 TC1.3D: Strømtrekk motor 20

5.3.6 TC2.1D: I2C-buss til IMU . . . 20
5.3.7 TC2.2D: Kontakt med IMU . . . 21
5.3.8 TC3.1D: CAN-oppkobling . . . 21
5.3.9 TC3.2D: CAN-buss mellom MCU . . . 21
5.3.10 TC4.1D: USB-krets . . . 21
5.3.11 TC5.1D: MCU oppkobling . . . 22
5.4 Produksjonstester . . . 22
5.4.1 TE1.1P: DC-buck 48-3V3 . . . 22
5.4.2 TE2.1P: DC-buck 48-5V . . . 22
5.4.3 TE3.1P: Strømforsyning MCU . . . 22
5.4.4 TE3.2P: Klokkekrets MCU . . . 22
5.4.5 TE4.1P: Strømforsyning motordrivere . . . 22
5.4.6 TE5.1P: Strømforsyning CAN-transcievere . . . 22
5.4.7 TC1.1P: Enkoderavlesing . . . 23
5.4.8 TC1.2P: Motordriver PWM-signal . . . 23
5.4.9 TC1.3P: Strømtrekk motor . . . 23
5.4.10 TC2.1P: Vinkelmåling fra IMU . . . 23
5.4.11 TC3.1P: CAN-buss mellom MCU . . . 23
5.4.12 TC4.1P: USB-kobling . . . 24
5.4.13 TC5.1P: Optocoupler . . . 24
5.4.14 TC6.1P: Endestopper . . . 24
5.4.15 TC7.1P: MCU oppkobling . . . 24
5.4.16 TC8.1P: UART-kommunikasjon . . . 24

6 Testresultater, diskusjon **25**
6.1 Designtester . . . 25
6.1.1 TC1.1D: Motordriver PWM-signal . . . 26
6.1.2 TC1.2D: Motor PWM-regulering . . . 26
6.1.3 TC1.3D: Strømtrekk motor . . . 26
6.1.4 TC2.1D: I2C-buss til IMU . . . 27
6.1.5 TC2.2D: Kontakt med IMU . . . 27
6.1.6 TC5.1D: MCU oppkobling . . . 28
6.2 Testing av produserte kort . . . 28
6.2.1 TC1.1P: Enkoderavlesing . . . 29
6.2.2 TC1.2P: Motordriver PWM-signal . . . 30

6.2.3 TC1.3P: Strømtrekk motor . . . 30
6.2.4 TC2.1P: Vinkelmåling fra IMU . . . 31
6.2.5 TC7.1P: MCU oppkobling . . . 31
6.2.6 TC8.1P: UART-kommunikasjon . . . 31
6.3 Øvrige funn . . . 31

7 Avsluttende refleksjoner **32**
7.1 Oppsummering . . . 32
7.2 Lærdom . . . 32
7.3 Nytteverdi . . . 33

Bibliografi **34**

Vedlegg **35**
A Kravspesifikasjon TTK4550 . . . 35

Figurer

1 Kontekstdiagram . . . 3
2 Strukturert analyse . . . 4
3 Armen . . . 5
4 Produserte kort . . . 6
5 Kretsskjema treghetssensor . . . 8
6 Kretsskjema motordriver . . . 8
7 Kretsskjema lavspent . . . 9
8 Kretsskjema mikroprosessor . . . 9
9 Testbenker i KiCad . . . 10
10 Etsing . . . 11
11 Boring . . . 12
12 Kobling av spenningsregulatorene for 3.3V og 5V . . . 13
13 Lavspent oppkobling . . . 14
14 Motordriver oppkobling . . . 15
15 Treghetssensor oppkobling . . . 16
16 Oppkobling mikroprosessor . . . 17
17 Resultat TC1.1D . . . 26
18 Resultat TC1.3D . . . 27
19 Resultat TC2.1D . . . 27

20 Resultat TC5.1D 28
21 Resultat TC1.1P 30
22 Resultat TC1.2P 30
23 Resultat TC8.1P 31

Tabeller

1 Oversettelser og forkortelser 1
2 Resultater designtester 25
3 Produkttestresultater 29

1 Forkortelser og oversettelser

Sortert alfabetisk på norsk. En del faguttrykk i denne oppgaven ble oversatt mer eller mindre ad-hoc. Lista er antakelig ikke uttømmende, men er basert på hvilke uttrykk forfatteren var mest usikker på.

Forkortelse	**Engelsk**	**Norsk**
ACK	Acknowledge	–
CAN	Controller Area Network	–
I2C	InterIntegrated Circuit	–
STM	STMicroelectronics	–
USB	Universal Serial Bus	–
ADC	Analog Digital Converter	Analog til digital-omformer
–	Debugging	Avlusing
–	Track width	Banebredde
–	Pin	Bein
BDC	Brushed Direct Current motor	Børstet likestrømsmotor
DOF	Degrees Of Freedom	Frihetsgrader
–	Switching regulator	Ikke-lineær regulator
IC	Integrated Circuit	Integrert krets
–	Pinch	Klyping
–	Breadboard	Koblingsbrett
PCB	Printed Circuit Board	Kretskort
–	Quadrature Encoder	Kvadraturpulsenkoder
–	Target unit	Målenhet
–	Hardware	Maskinvare
MCU	Micro Controller Unit	Mikroprosessor
–	Bulk capacitor	Motorkondensator
–	Optocoupler	Optisk kobler
–	Software	Programvare
PWM	Pulse Width Modulation	Pulsbreddemodulering
–	(software) flashing	Skriving (av programvare)
–	Voltage regulator	Spenningsregulator
IMU	Intertial Measurement Unit	Treghetssensor
–	Breakout board	Utbrytningskort
–	Noise reduction capacitor	Utjevningskondensator
DevKit	Development Kit	Utviklingskort
–	Twist	Vridning

Tabell 1: Oversettelser og forkortelser

2 Introduksjon

Dette dokumentet beskriver semesteroppgaven gjennomført av meg, Kristian Blom, i emnet *TTK8 – Konstruksjon av innebygde systemer* høsten 2023. Oppgaven var å designe og produsere testmoduler, noe misvisende kalt testbenker i tidligere innleveringer, for prosjektoppgaven i emnet *TTK4550 – Fordypningsprosjekt*, samt utarbeide og gjennomføre et valideringsregime for kretskort produsert for denne. Arbeidet gjort i TTK8 må derfor forstås i sammenheng med prosjektbeskrivelsen for TTK4550, og noen elementer i denne rapporten vil derfor være duplisert i rapporten til TTK4550. Disse vil være tydelig markert.

2.1 Bakgrunn og motivasjon

Omega Verksted er en av mange studentdrevne interesseorganisasjoner på NTNU, og spesialiserer seg på å legge til rette for at studenter skal kunne gjennomføre elektronikkprosjekter på hobbybasis. På et ukjent tidspunkt i løpet av 2021 ble det donert en Beckman Coulter ORCA robotarm til verkstedet i håp om at noen kunne få glede av den. Dette er en 6DOF arm med skulder-, albue- og håndledd montert på en lineæraktuator i tillegg til vridning og klyping av fingre, trolig produsert på midten av 1990-tallet. Virkemåten er nokså enkel: 6 børstede likestrømsmotorer drives av 3 mikroprosessorer i tilbakekopling med én kvadraturpulstogsenkoder per motor. I tillegg er det én helningssensor i hver av de to armseksjonene for å hjelpe til med tilstandsdefinering. Etter gjentatte, mislykkede forsøk på å få kontakt med den ble det bestemt at det skulle designes et helt nytt sett med kretskort og tilhørende programvaresystemer for å drive den.

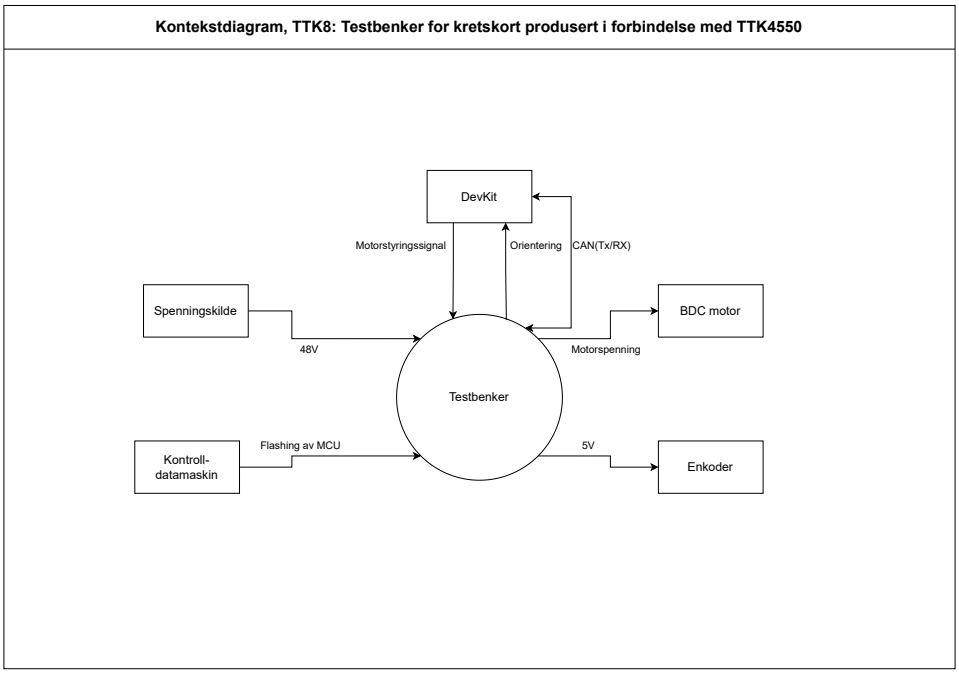
Dette er et omfattende prosjekt, og en vesentlig del av det vil være å verifisere at de produserte enhetene, i form av både maskinvare og programvare, virker som tiltenkt. Et nyttig tiltak i dette henseende er å sette opp de designede kretsene på koblingsbrett, slik at en får muligheten til å avdekke og rette feil i et miljø der det er enkelt å gjøre endringer før kretsene sendes til produksjon. Dette gir så et verdifullt datagrunnlag for oppsett og gjennomføring av valideringstester på de endelige produktene: å gjøre flere iterasjoner med testdesign direkte på produserte kretskort ville være både kostbart og tidkrevende.

2.2 Omfang

Denne oppgaven omfatter

1. produksjon av modulære maskinvaretester for kretsene designet for robotarmen,
2. produksjon av modulære programvaretester for kretsene designet for robotarmen, og
3. utarbeiding og gjennomføring av valideringstester for både testenheter og produserte kretskort, fortrinnsvis med fokus på grensesnitt mellom maskinvaremodulene.

3 Systembeskrivelse

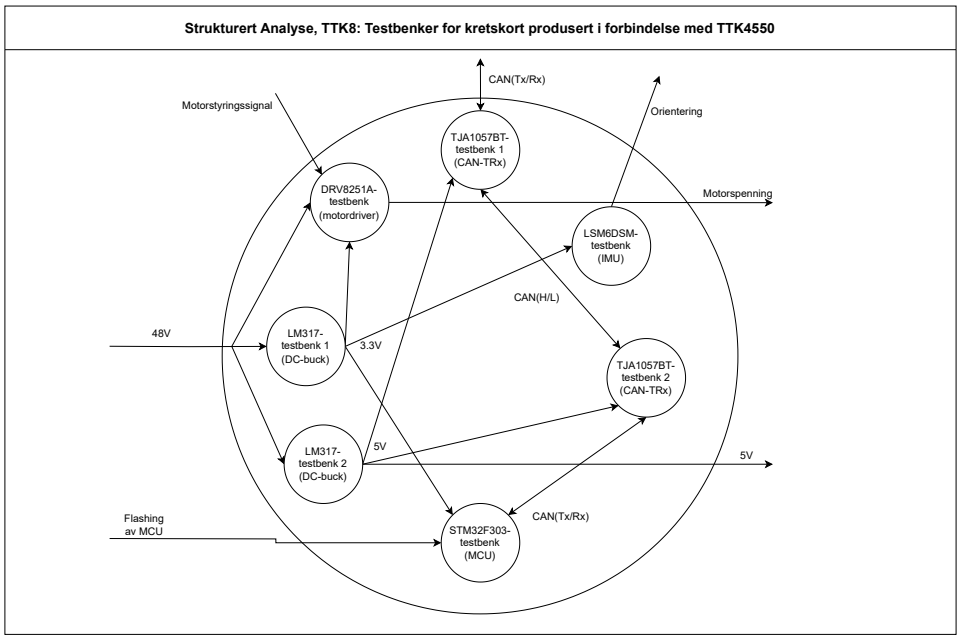


Figur 1: Kontekstdiagram

Dette kapittelet beskriver i hovedsak maskinvaretestene, altså punkt 1 i lista i kapittel 2.2.

Som vi ser av Figur 1 består systemet av 'testbenker' som grenser mot eksterne enheter som representerer elementene som vil inngå i å drive robotarmen.

1. DevKit: Utviklingskortet NUCLEO-F303RE er designet rundt mikroprosessoren STM32F303RE, som er hovedprosessoren for robotarmen.
2. Spenningskilde: Mascot Type 719 representerer en hittil ubestemt hovedforsyning for robotarmen. Denne leverer maksimalt 30V, det lyktes ikke å finne en labforsyning med 48V.
3. BDC motor: Børsteløs likestrømsmotor av ukjent modell, en av 6 motorer montert i armen.
4. Enkoder: HEDS9100, en av 6 kvadraturpulsenkodere montert i armen.
5. Kontrolldatamaskin: Dell All-In-One Intel i7 med Ubuntu 22.04 representerer en hittil ubestemt datamaskin som vil ha hovedansvaret for å sende settpunkt m.v. til prosessorene i armen. Brukes her hovedsakelig til skriving av mikroprosessorene og avlusing av programvare.



Figur 2: Strukturert analyse

Testbenkene som er hovedsubjekt for denne oppgaven ser vi i Figur 2. Disse er 7 moduler som skal teste og validere de viktigste funksjonene for armen. De består i hovedsak av en integrert krets med tilhørende støttekomponenter.

1. DRV8251A: Motordriver for børsteløse likestrømsmotorer. Disse tar inn to pulsbreddemodulerte signaler fra utviklingskortet og forsterker dem til input-spenningen før de sendes videre til motoren.
2. LM317: Justerbar, lineær speningsregulator. Tar inn input-spenning, justerer ned til henholdsvis 3.3V og 5V, og sender til de relevante enhetene. Det er altså to av disse enhetene.
3. TJA1057B: CAN-transciever, tester CAN-bussen som skal gå mellom prosessorene i armen. Det er to av disse enhetene, én koblet til utviklingskortet og én til mikroprosessoren.
4. STM32F303: Mikroprosessor. Testbenken tester strømforsyning, oppkobling av CAN-buss og flashing.
5. LSM6DSM: treakse treghetssensor og gyroskop. Kommuniserer med utviklingskortet over I2C.

3.1 Forklaring i kontekst av robotarmen

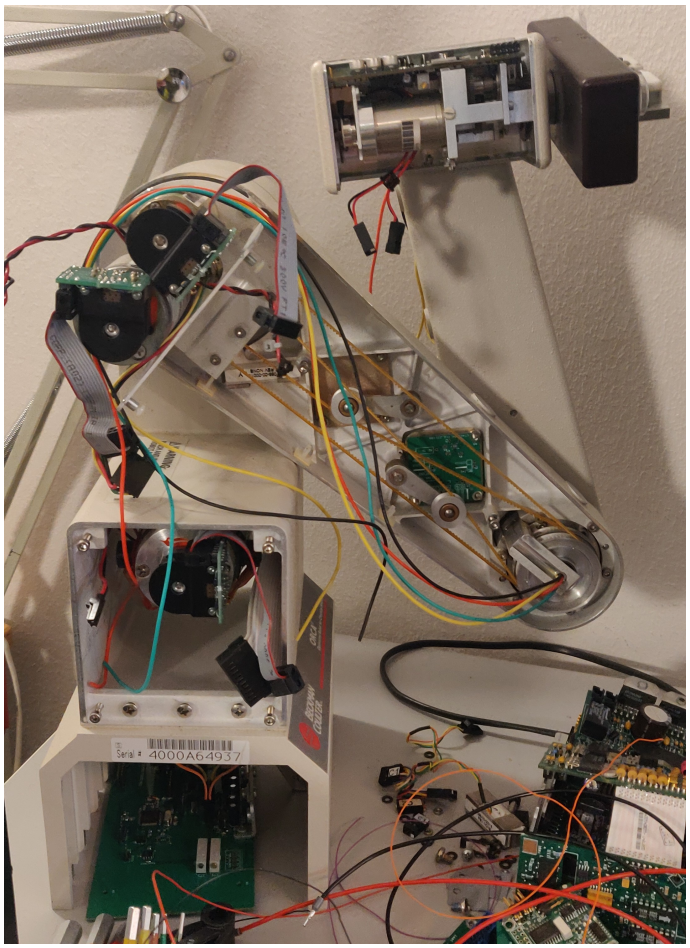
Dette underkapittelet omhandler i stor grad arbeid gjort for TTK4550.

Noe videre forklaring behøves for å forklare hensikten til systemet. Kretskortene som vil stå i robotarmen består i hovedsak av tre kort, hver med én prosessor som har ansvar for å drive to motorer med tilbakekopling fra enkodere. I tillegg vil én prosessor kommunisere med to treghetssensor, og én vil kommunisere med én treghetssensor, over I2C. Dette er en funksjonsmessig gjenskaping av det originale systemet som styrte armen. Dernest vil de tre prosessorene kommunisere med hverandre over CAN-buss, og dette nødvendiggjør CAN-transcieverne. Til slutt vil det naturligvis også være nødvendig å kunne skrive programvare til mikroprosessorene.

Funksjonene som skal testes er altså:

1. Strømforsyning på 3.3V og 5V spenning.
2. Dataoverføring via CAN-buss.
3. Dataoverføring via I2C.
4. Skriving av programvare til mikroprosessor.
5. Lesing av kvadraturpulsenkodere.
6. Regulering av spenning til motorer.

Ettersom det til slutt er en mikroprosessor montert på et produsert kretskort som vil være ansvarlig for alle disse funksjonene kunne en argumentert for å sløyfe utviklingskortet og gjøre alt via mikroprosessortestbenken. Utviklingskortet er i midlertid en kjent størrelse som det i utgangspunktet skal være trygt å anta at virker i henhold til databladet. Ved å modularisere designet får en vanntette skott mellom funksjonene som testes, og kan lettere validere at designet fungerer som tiltenkt. Så hvorfor lage en egen testenhet for mikroprosessoren i det hele tatt – en 'tester' jo skriving til mikroprosessoren hver gang en laster inn et nytt program på utviklingskortet? Det er delvis fordi det uansett trengs to enheter for å teste en CAN-buss, og delvis fordi skriving til mikroprosessor er innebygget i utviklingskortet og dermed ikke kan anses som en tilstrekkelig test av designet som skal inn i robotarmen. De andre funksjonene er derimot koblet helt eller tilnærmet likt slik de vil være i armen, som på utviklingskortet.



Figur 3: Armen oppgaven dreier seg om

3.1.1 Nærmere beskrivelse av armen

Her følger en mer detaljert beskrivelse av armen og maskinvaren som inngår i den.

Armen har altså 6 ledd, hvorav 5 er synlige i figur 3. I 'hånda' er det montert ett kretskort med ansvar for å styre de to motorene som er ansvarlige for vridning og klyping av 'fingrene' (svart boks øverst til høyre i bildet), måling av helningsvinkel til hånda, og måling av helningsvikel i 'underarmen'. I tillegg leser den av måling fra en optisk kobler som reagerer på hvorvidt fingrene er rotert 180 grader eller ikke. Den optiske kobleren er montert på et annet kort enn mikroprosessoren, også montert i hånda. Treghetssensoren for underarmen er montert på et eget kort i underarmen, ikke synlig på bildet. Kvadraturpulsenkoderne som måler rotasjonen til motorene er festet i bakenden av hver motor, særlig synlig i skulderleddet oppe til venstre i bildet.

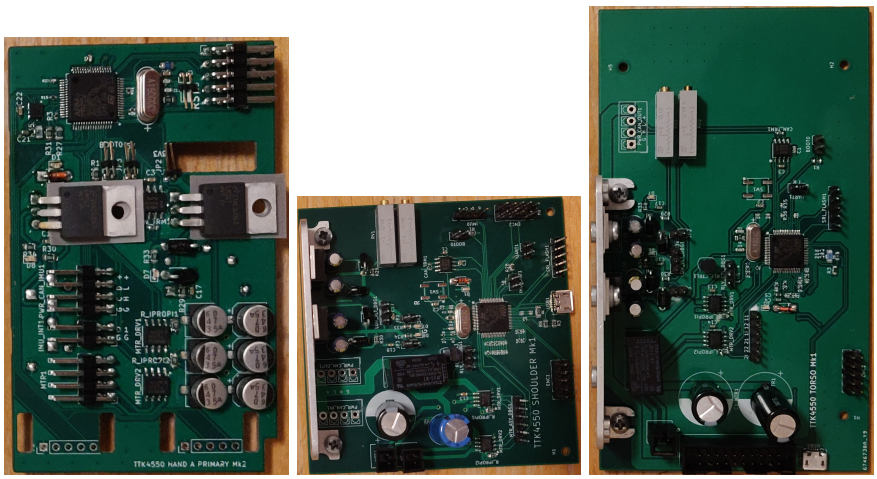
I 'skulderen' er det montert ett kretskort med ansvar for å styre de to motorene som aktuerer 'albuen' og 'håndleddet' via belter og tannhjul knapt synlige på bildet, og måling av helningsvinkel i 'overarmen'. Kortet er altså montert rett under de to motorene synlige øverst til venstre i bildet, men var tilfeldigvis demontert da bildet ble tatt. Treghetssensoren for overarmen er montert på et eget kort i overarmen, synlig omtrent midt i bildet.

I 'torso' er det montert ett kretskort med ansvar for å styre de to motorene som aktuerer 'skulderen' og lineæraktuatoren som armen normalt er montert på, lese av en endestoppbryter på lineæraktuatoren, og kommunisere med en ekstern datamaskin over USB. Lineæraktuatoren er omtrent 1.5 meter lang og fikk ikke plass i bildet – eller på labplassen min, for den saks skyld. Kortet ligger for anledningen på pulten, gjemt under armen i hulrommet for lineæraktuatoren, men monteres i det kvadratiske hulrommet rett ovenfor. Der er også skulder-motoren knapt synlig.

Kommunikasjon mellom mikrokontrollerne og treghetssensorene foregår altså over I2C, og kommunikasjon mellom mikrokontrollerne foregår over CAN-buss.

Felles for alle kortene er at de har to spenningsregulatorer, én for 3.3V og én for 5V, i begge tilfeller nedregulert fra en 48V ekstern forsyning. De fleste enhetene på kortene forsynes med 3.3V, men enkoderne, CAN-buss, USB og endestoppbryteren behøver 5V.

Det er disse tre kortene denne oppgaven har som mål å verifisere, og i testresultatene, kapittel 6, er de omtalt som henholdsvis 'torso', 'shoulder' og 'hand' (TTK4550 gjennomføres på engelsk). Det bør bemerkes at dette i utgangspunktet er tre forskjellige utlegg av den samme kretsen, så i prinsippet skal det holde å teste hvilket som helst av de tre for å verifisere designet av et gitt delsystem. Kosmetiske forskjeller som kondensatorstørrelser og liknende finnes, men alle grensesnitt mellom delsystemene er like – altså er eksempelvis motordriverne koblet til de samme beina på alle tre mikroprosessorene. Figur 4 viser de tre produserte kortene. Det understrekes igjen at dette ble produsert for TTK4550.



Hand Shoulder Torso

Figur 4: Kortene produsert for å erstatte maskinvaren fra midten av 90-tallet

4 Design av testenheter

Dette kapittelet beskriver i hovedsak implementasjonen av punkt 1 i lista 2.2.

4.1 Komponentvalg

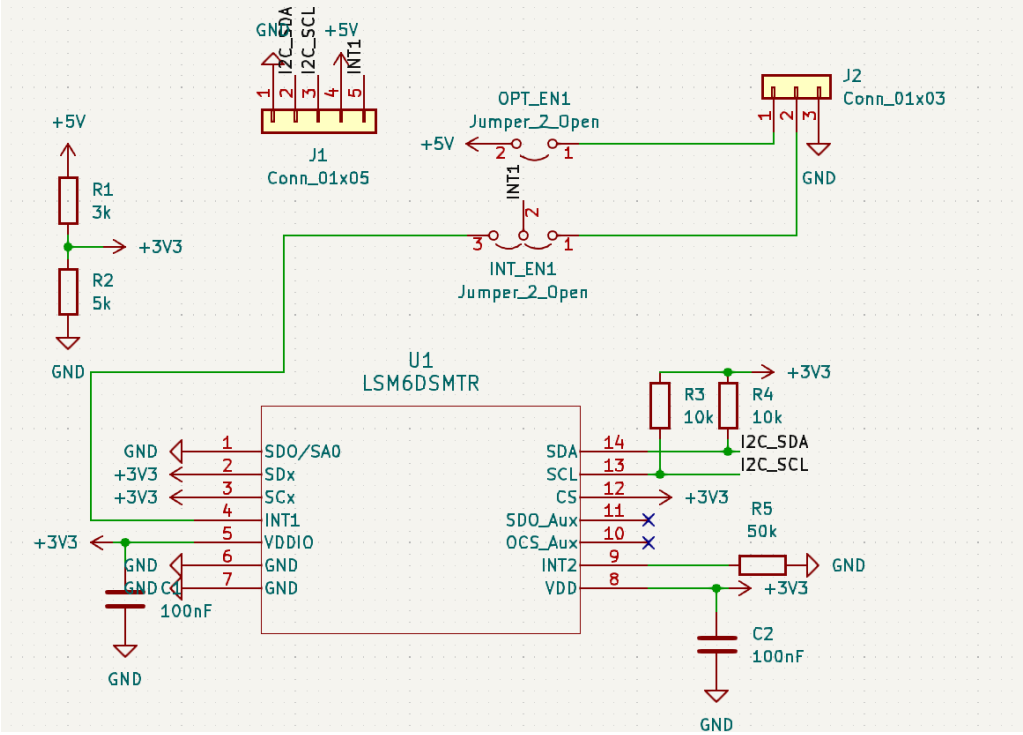
De integrerte kretsene brukt i testbenkene ble valgt fordi de ble funnet å oppfylle kravene satt av TTK4550-oppgaven, gjengitt vedlegg A. En oppsummerende liste følger. Øvrige komponenter, slik som mostander og kondensatorer, ble valgt på grunnlag av referanse- og anbefalte kretsskjema fra de integrerte kretsenes datablader.

- Lineær spenningsregulator LM317HV: Justerbar, enklere justeringskrets enn en ikke-lineær regulator[1], tåler opp mot 57V differensiale mellom spenning ut og inn.
- Motordriver DRV8251A: Kan drive 48V ved 4A, dette er en god match mot de originale motordriverene. PWM-grensesnitt mot mikroprosessor er forholdsvis enkelt å sette opp. Støtter strømmåling fra motorene, som er vesentlig både i estimeringen av ytelse og som sikkerhetsfunksjon.
- CAN-transciever TJA1057BT: Ingen særlige krav ut over å oppfylle kravene til en CAN-transciever.
- Treghetssensor LSM6DSM: Måling av minst 2g i opp mot 3 akser, både akselerasjon og vinkelhastighet.
- Mikroprosessor STM32F303: Kan håndtere grensesnitt og signalbehandling fra alle ovennevnte uten bruk av eksterne enheter ut over et klokkekrystall. Altså individuelt drevne PWM-kanaler, enkoderavlesing, CAN-/I2C-/USB-grensesnitt m.v.

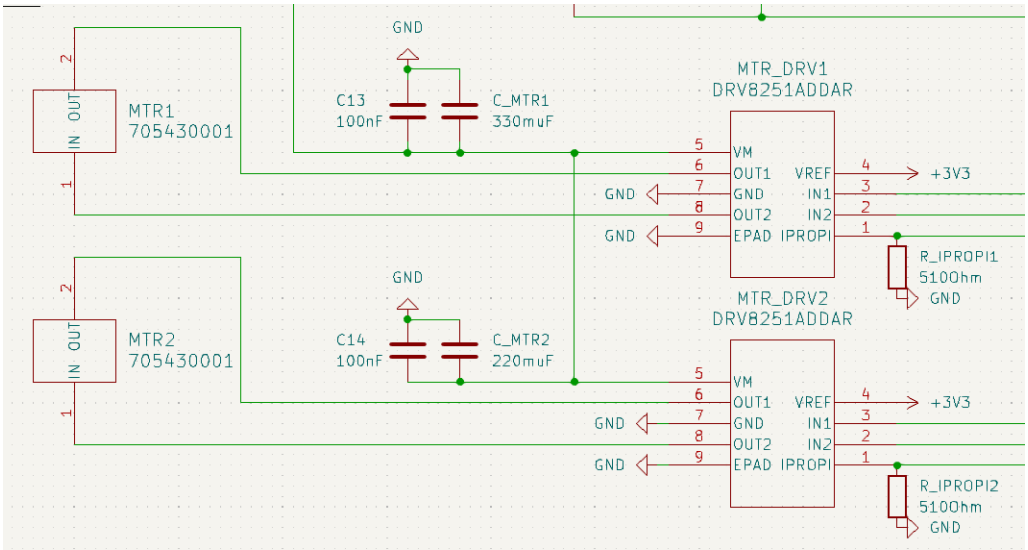
[1] Det foreslås å innføre 'tasteregulator' som en mer konsis oversettelse av switching regulator

4.2 Kretsdesign

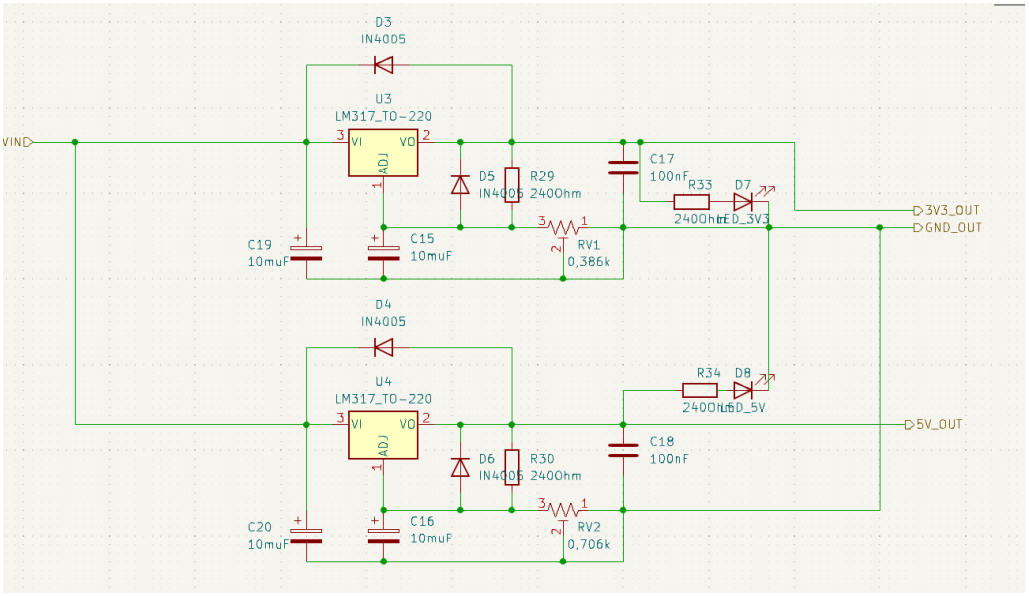
Det understrekes at kretsskjemaene presentert i dette underkapittelet ble produsert i forbindelse med emnet TTK4550, og inkluderes her for fullstendighetens skyld.



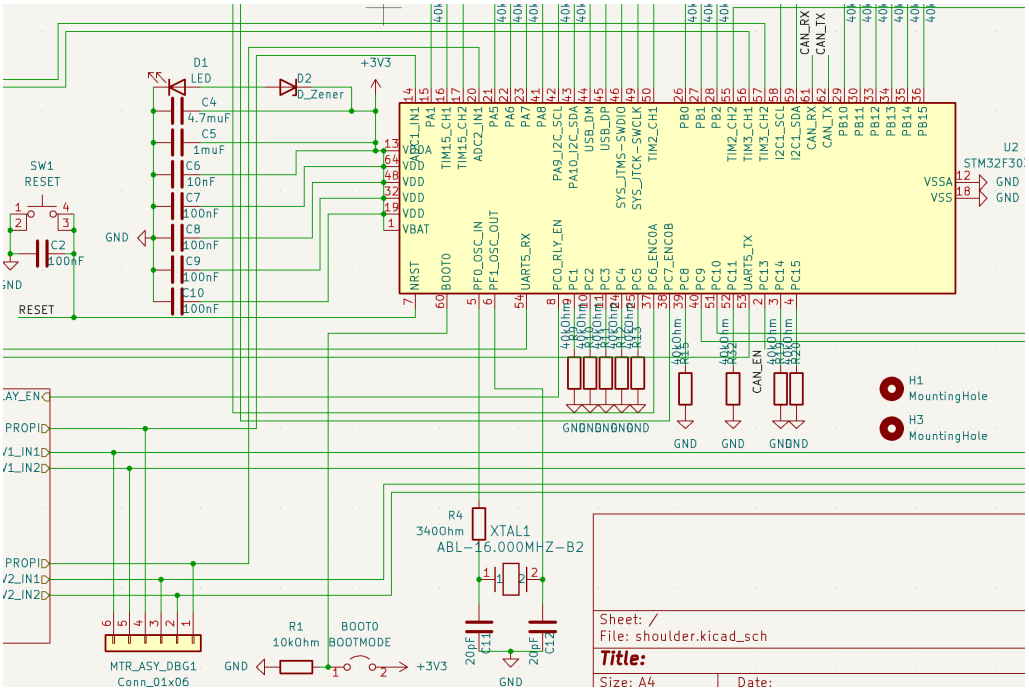
Figur 5: Kretsskjema for treghetssensoren LSM6DSM. Koblingene J2, OPT_EN1 og INT_EN1 er ikke relevante for TTK8. Ellers sees I2C og strømforsyning på J1, og en enkel spenningsdeler som regulerer fra 5V til 3.3V i øvre venstre hjørne. Basert på figur 17 i enhetens datablad [2].



Figur 6: Kretsskjemaet viser to DRV8251A-enheter, men det er altså kun én som er koblet opp for TTK8. Beina IN1 og IN2 tar imot PWM-signal fra mikrokontrolleren, mens VM tar inn motorspenningen. De to kretsene er identiske utenom verdien av motorkondensatoren C_MTRn. Basert på Figure 9-1 i enhetens datablad [3].



Figur 7: Kretsskjemaet viser spenningsregulatoren LM317HV, justert for henholdsvis 3.3V og 5V i de to enhetene. Trimmepotmeterne RVn gjør det mulig å justere spenningen ut. I testbenkene er funksjonstest-LED'ene D7 og D8 sløyfet. Designet er basert på Figure 15 og Figure 16 i enhetens datablad [4].



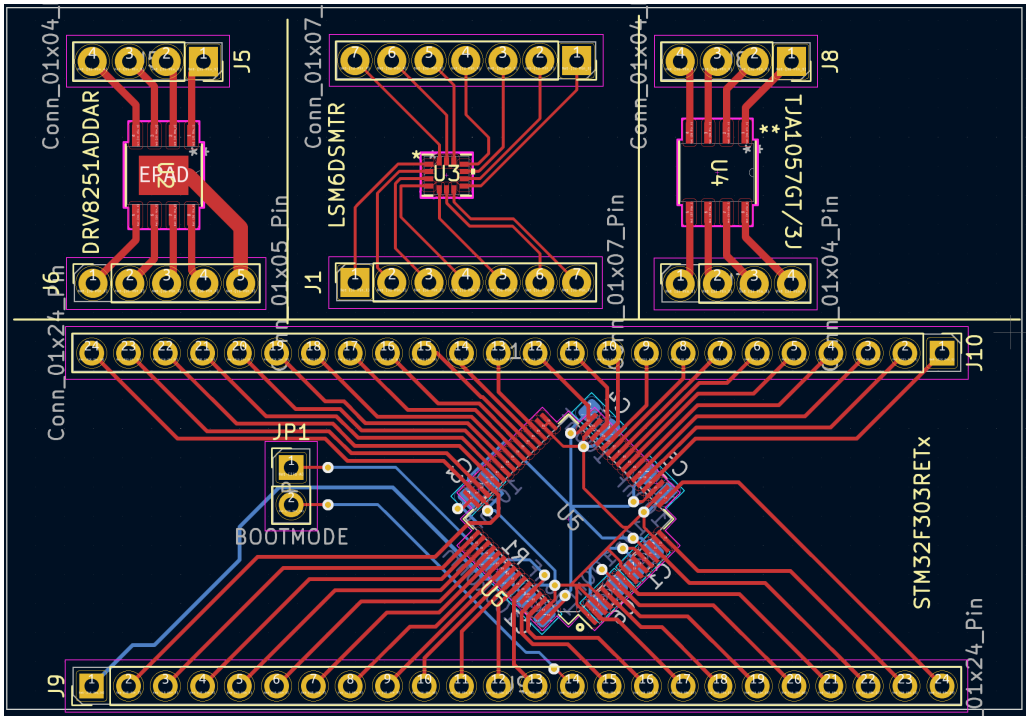
Figur 8: Kretsskjemaet viser mikroprosessoren STM32F303RET6 slik den er koblet på et av kortene. Bildet er noe avkortet, ettersom en fullstendig oversikt i praskis vil vise hele skjemaet for kortet. De relevante elementene for TTK8 er utjevningskondensatorene mellom 3.3V forsyning og jord til venstre i bildet, klokkekretsen nederst i midten, beina [7,46,49] som brukes til skriving, og beina [61,62] som brukes til CAN-buss. Basert på Figure 14 i STM32F303 applikasjonsnotat [1]

.

4.3 Produksjon av utbrytningskort

For å gjøre de integrerte kretsene, altså alle untatt spenningsregulatorene, kompatible med koblingsbrett, ble det designet og produsert utbrytningskort til disse. Med unntak av kortet til mikroprosessoren hadde de ingen funksjon ut over mekanisk tilpasning til koblingsbrettet. Ettersom mikroprosessoren har 64 bein, hvorav kun et fåtall ville være relevant for testbenken, ble flere bein kortsluttet mot hverandre og ledet til jord. I tillegg ble det montert utjevningskondensatorer for kraftforsyningen på kortet, og kortet har kun ett bein for henoldsvis 3.3V og jord. Alt dette for å spare plass på koblingsbrettet.

4.3.1 Design av utbrytningskort



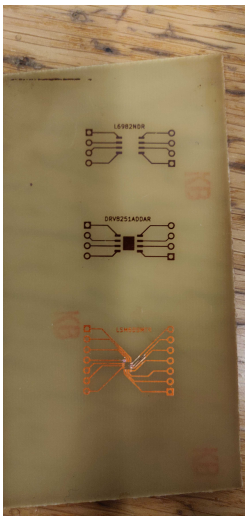
Figur 9: Øverst fra venstre: Motordriver, treghetssensor, CAN-transciever, mikroprosessor

Testbenkene i Figur 9 ble designet i KiCad 7. For at testenhetene skulle passe i koblingsbrettet måtte avstanden mellom beina være en multippel av 2.54 mm i begge retninger. I tillegg måtte hullene være minst 1 mm i diameter, da dette er det minste tilgjengelige boret på Omega Verksted. Banebredden ble tilpasset bredden av de individuelle loddepunktene på hver enhet.

De blå banene i utbrytningskortet til mikroprosessoren representerer baner på undersiden av kortet. Disse er primært strømforsyning, og man kan skimte fotavtrykkene til 6 utjevningskondensatorer på 100 nF og en motstand på 10 kOhm i tilknytning til JP1.

4.3.2 Fremkalling i etsebad

Etsen på Omega Verksted bruker natriumpersulfat og lut



Tidlig utgave av testbenkene for strømforsyning, motordriver og treghetssensor

Figur 10: Etsing

Etter at designet i KiCad var ferdig ble kortene etset frem av kobberplater ved bruk av etsebadet på Omega Verksted. Det bruker lut til fremkalling, og natriumpersulfat til etsing. Hver test med badet kunne ta 20-40 minutter avhengig av forholdene, bl.a. temperaturen i badet, mengden kobber som skulle etses vekk fra plata, og hvor mye kobber løsningen hadde absorbert siden sist den ble byttet. Normal pH ligger rundt 2 for etsen, og rundt 14 for fremkalleren. I håndteringen av disse ble det derfor ikke prioritert å ta illustrerende bilder underveis, selv om akkurat denne kombinasjonen av kjemikalier angivelig er kjent som 'forholdsvis snill' hva HMS-risiko angår.

Med riktig kombinasjon av forhold i etsen kan en oppnå svært gode resultater for enkeltsidige kretskort. Det er for eksempel ikke spesielt vanskelig å etse baner ned til 0.2 mm bredde, slik som banene i den endelige testenheten til treghetsmåleren, så lenge en sørger for god sirkulasjon i badet og inspiserer ofte mot slutten av etsingen. Det er derimot forholdsvis lett å gjøre feil som å speilvende designet på kobberplata – noe en uerfaren etser gjerne ikke innser at har skjedd før han eller hun setter seg ned for å montere komponenter. I tillegg var det vanskelig å sørge for lik plassering av fremre og bakre lag ved fremkalling av tosidige kort, særlig når feiltoleransen er på om lag 0.1 mm. Dette var noe av grunnen til at mikroprosessorkortet til slutt ble produsert

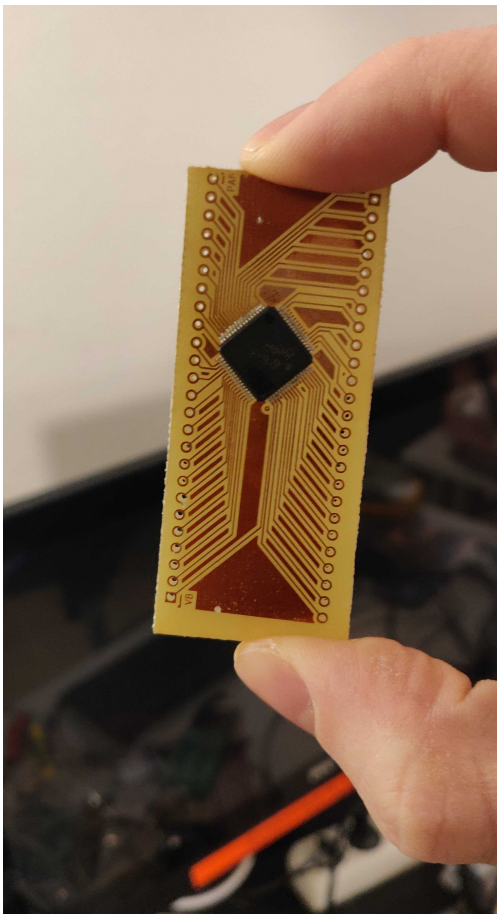
profesjonelt.

4.3.3 Lodding og boring av hull

Neste steg etter etsing er boring av hull til utbrytningskortets bein, se Figur 11. Pinnene som typisk brukes på et koblingsbrett ble funnet å passe fint i et hull på 1 mm i diameter, så denne bordiameteren ble brukt. For produksjon av viaer til tosidige kort kreves det en egen viapresse som ikke ble funnet intakt, og dette ble siste spikeren i kista for produksjonen av utbrytningskort for mikroprosessoren. I tillegg har boremaskina ingen mulighet for måling før boret treffer kortplata. Ettersom alle hull bores på øyemål blir rekka med hull til beina noe skeve. For kort med få bein er dette nokså uproblematisk og gjør loddeprosessen i neste steg enklere, men for mikroprosessoren ble det ikke mulig å få på plass flere enn et par bein av gangen.

Siste steg er lodding. For disse kortene ble det i all hovedsak benyttet loddepasta og varmluftspistol for å feste komponentene til kortene. For mikroprosessoren ble varmluftspistolen i midlertid byttet ut med loddeovn, da jevn oppvarming av alle prosessorens 64 bein er vesentlig for at kapillærkreftene i det smeltede tinnet skal kunne trekke komponenten nøyaktig på plass oppå loddepunktene. De større komponentene var mindre ømfintlige for dette. Resultatet er illustrert i Figur 11, og inspeksjon med mikroskop viste at få bein hadde blitt kortsluttet. En kombinasjon av flussmiddel, loddebolt og skalpell under stereomikroskop ble brukt for å reparere kortslutninger.

Kretskortboret på Omega Verksted kan i prinsippet bore helt ned til 0.6 mm, men ingen av disse borene ble funnet intakte.



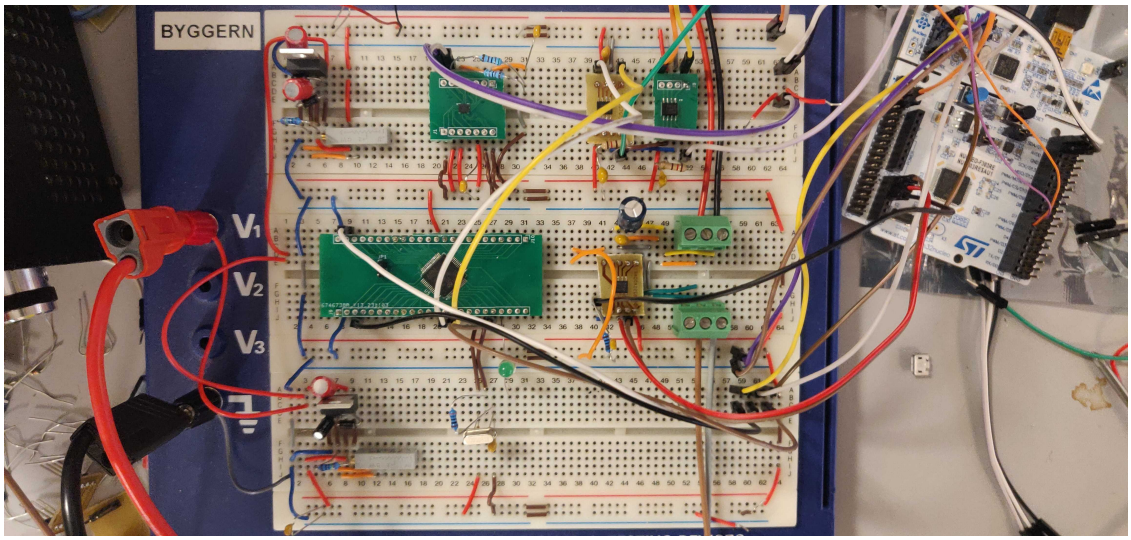
Forsøk på boring av kort til mikroprosessoren. Det synes ikke så godt på bildet, men rekkene med hull er nokså skeve.

Figur 11: Boremaskin og resultat

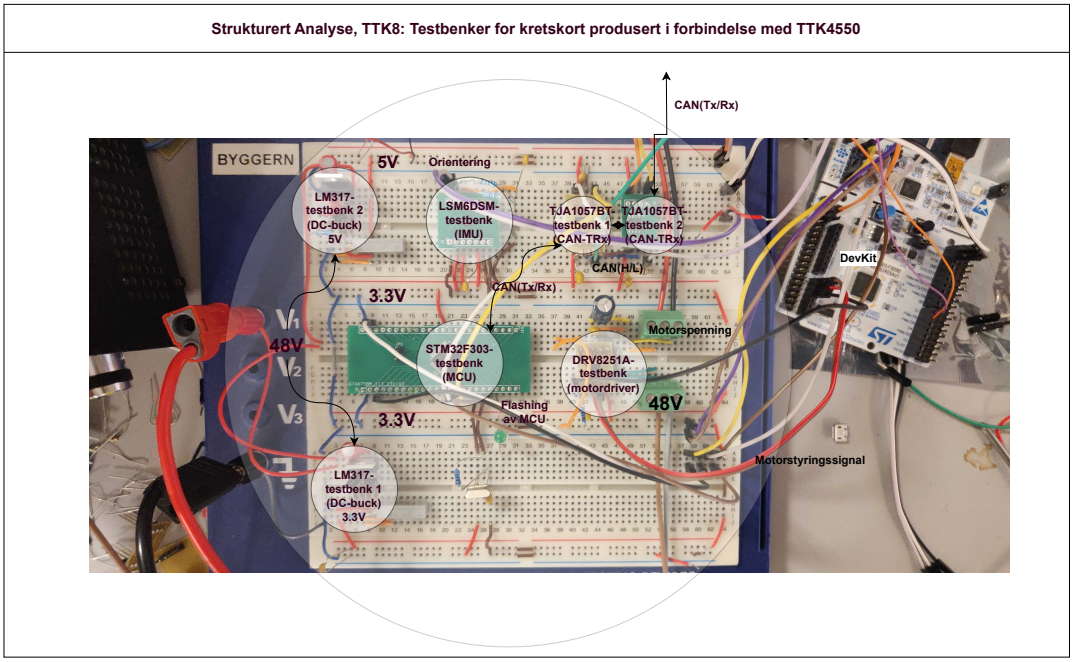
4.4 Sammenstilling av testmoduler, resultater

Testmodulene ble satt sammen på bakgrunn av kretsskjemaene i kapittel 4.2. Dersom disse ikke virket, ble en fungerende løsning eksperimentert frem og kretsskjemaet oppdatert. Denne typen iterativ utvikling tar tid, men gir ofte gode resultater.

For oversiktens skyld er koblingsbrettet med alle testenhetene koblet opp presentert i Figur 12. Et forsøk på å knytte koblingsbrettet til den strukturelle analysen ble gjort, leseren får bedømme hvorvidt det var vellykket.



Hele koblingsbrettet med alle testenhetene koblet opp.

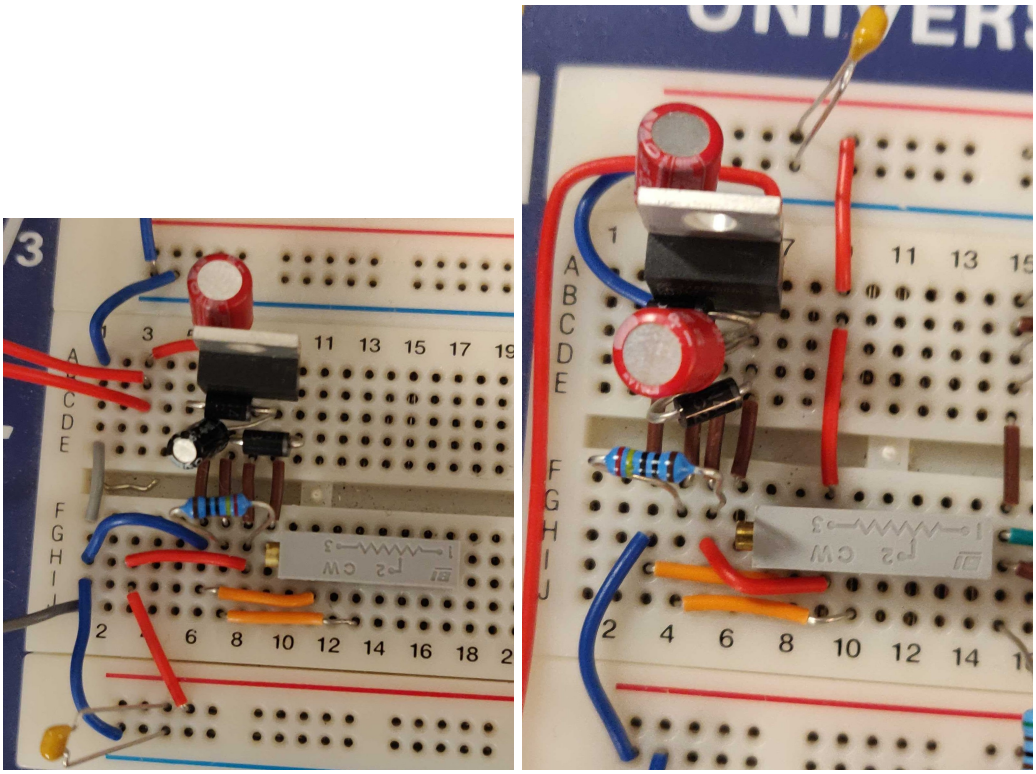


Koblingsbrettet med noen av elementene fra den strukturelle analysen lagt oppå.

Figur 12: Kobling av spenningsregulatorene for 3.3V og 5V

4.4.1 Lavspent spenningsregulering

Kretsene i Figur 13 tester funksjonen til spenningsregulatoren L317HV, og er satt opp identisk med kretsen i figur 7. Høyspenttilkoblingen (48V i designet, ¡=30V i praksis) mates inn på baksiden av regulatorene, og den lavere spenningen føres ut til skinnene på koblingsbrettet. Felles jord for begge kretsene kan sees som en grå ledning på vei ut av bildet nede til venstre på figuren for 3.3V. Her var kretsen designet korrekt på første forsøk, så ingen iterasjon var nødvendig. Det grå potmeteret justeres individuelt for de to regulatorene, og spenningen leses av med multimeter mellom jord og regulatorens midterste ben.



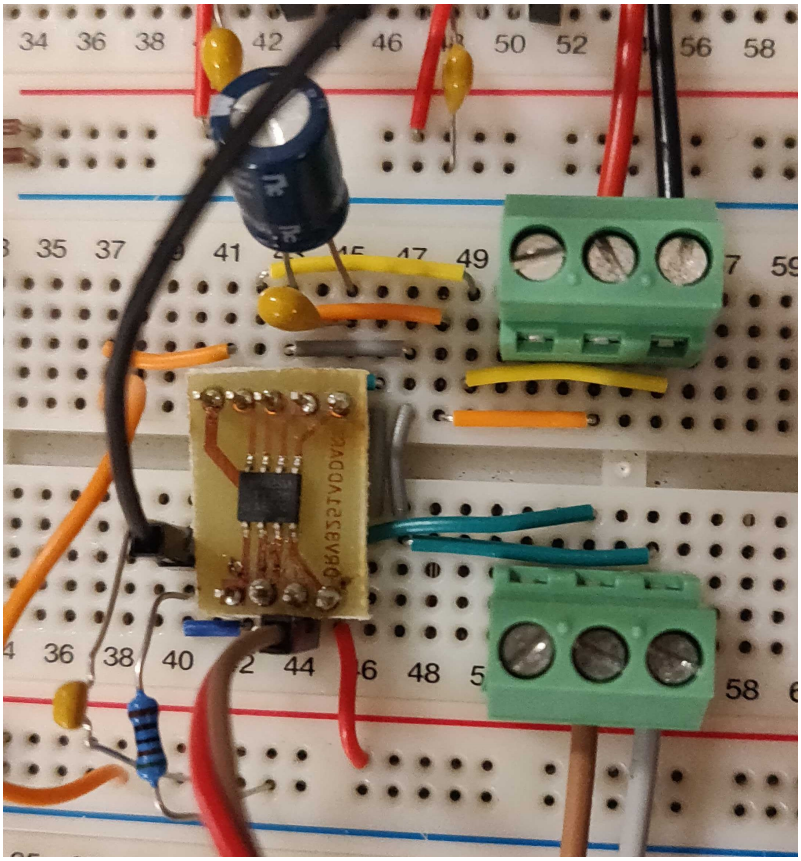
3.3V

5V. Ledningene helt til høyre i bildet inngår **ikke** i kretsen.

Figur 13: Kobling av spenningsregulatorene for 3.3V og 5V

4.4.2 Motordriver

Kretsen i Figur 14 tester funksjonen til motordriveren DRV8215A, og er satt opp identisk med kretsen i figur 6. Nede til høyre kommer 48V inn i brun og grå ledning og føres inn i driverens høyspentinngang på bein 5. Det røde og brune lederparet på vei ut av bildet nede til venstre tar inn motorstyringssignalet fra utviklingskortet. Oppe til høyre føres den justerte motorspenningen ut av kretsen og inn i motoren i det røde og svarte ledningsparet. En tidligere utgave av utbrytningskortet inkluderte ikke den brede banen øverst til venstre på kortet. Den jorder varmeledningspunktet på undersiden av motordriveren, noe som viste seg å være nødvendig for at driveren skulle fungere. Den svarte ledningen på vei ut i venstre halvdel av bildet er koblet til en ADC på utviklingskortet, og måler spenningen mellom strømmålingsbeinet til driveren og jord. Strømmen blir ledet til jord gjennom motstanden nederst til venstre i bildet. Strømmåling er for ordens skyld utenfor omfanget av oppgaven som beskrevet i den strukturelle analysen, men viste seg å være en relevant funksjon å teste. Mer om dette i kapittelet om testing.

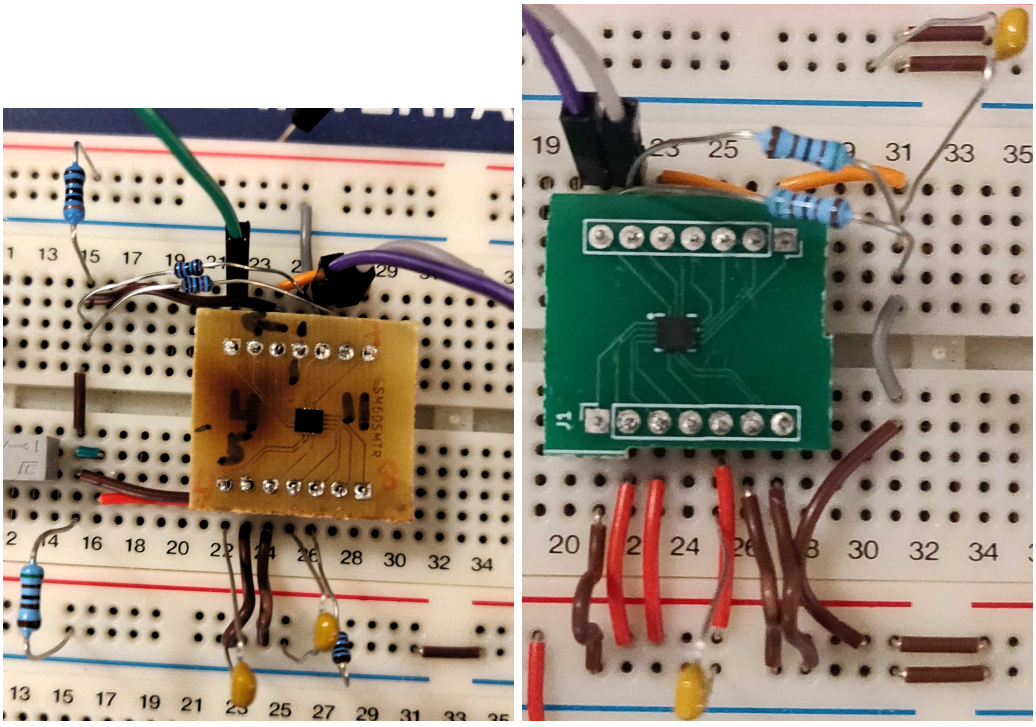


Figur 14: Motordriverkretsen

4.4.3 Treghetssensor

Kretsen i figur 15 tester funksjonen til treghetssensoren LSM6DSM. Versjon 1 er satt opp identisk med kretsen i Figur 5, og fungerte ikke. I tillegg til forvirring rundt hvilket av sensorens bein som havnet hvor på utbrytningskortet, kan sensoren ha blitt kortsluttet på noe tidspunkt. Den viktigste endringen er imidlertid frafallet av spenningsdeleren til venstre i bildet av versjon 1. I kretsskjemaet er to ordinære motstander ansvarlige for å justere ned spenningen fra 5V til 3.3V. Kretsen ble designet slik for å kunne støtte 5V forsyning til en annen sensor som er relevant for robotarmen (men faller utenfor omfanget av denne oppgaven) uten å introdusere en dedikert spenningsregulator. 5V forsyning kommer inn via motstanden øverst til venstre i bildet, og ledes mot jord via motstanden nederst til venstre. Akselerometermålingen fra treghetssensoren over I2C går til utviklingskortet via det lilla og grå ledningsparet øverst til høyre, med pull-up-motstander koblet på den produserte 3.3V-forsyningen fra spenningsdeleren.

I2C-bussen krever altså pull-up-mostander mellom bussen og logisk høy. Det antas at dette i praksis ble en parallellkobling mellom spenningsdeleren og I2C-bussen, slik at forsyningen og logisk høy ble 1.7V heller enn 3.3V. I versjon 2 ble derfor sensoren og I2C-bussen koblet på 3.3V-forsyningen fra spenningsregulator heller enn via 5V-forsyningen med spenningsdeler. Dette fungerte, men noe videre arbeid behøves for å implementere det i kretsskjemaet. I tillegg bruker versjon 2 en profesjonelt produsert versjon av kortet, som fjerner usikkerheten rundt hvor bein 1 er: nederst til venstre på utbrytningskortet. Utenom spenningsdeleren er oppkoblingen lik som i versjon 1, rent bortsett fra at sensoren er rotert og/eller speilvendt i forhold til versjon 1. 3.3V forsynes fra skinna nederst i bildet, og akselerometersignlet over I2C går til utviklingskortet via det lilla og grå ledningsparet øverst til venstre.



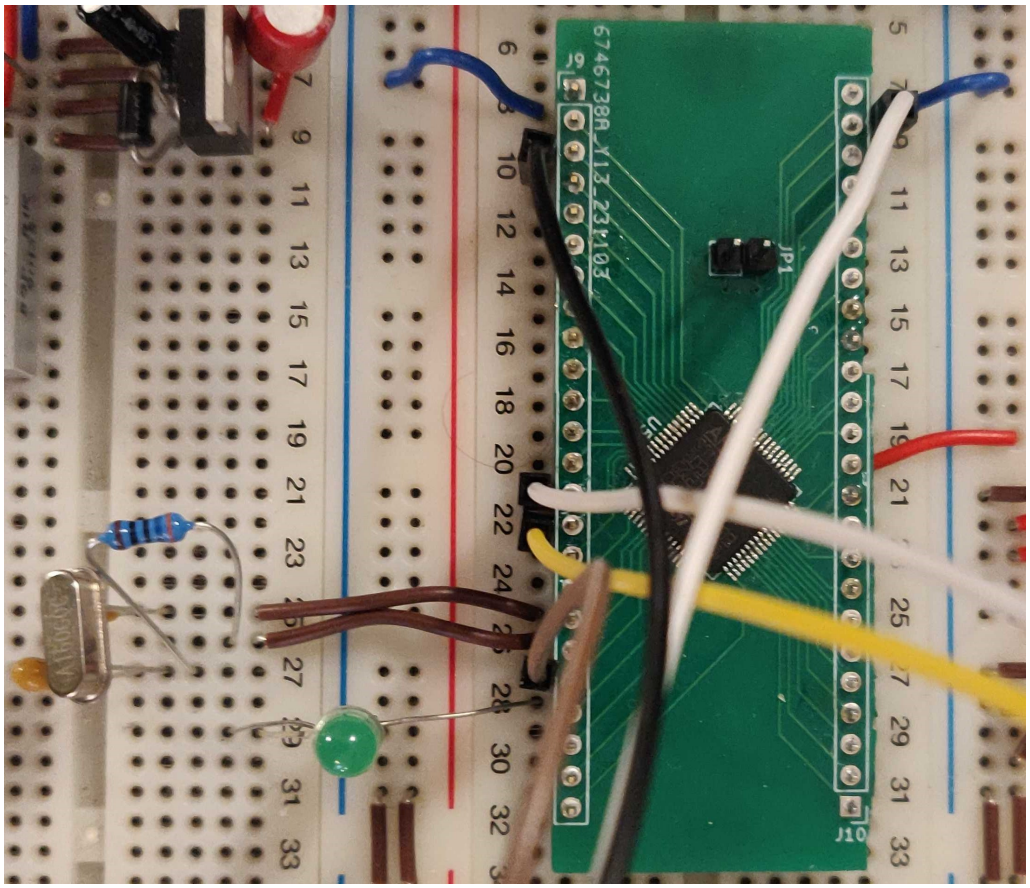
Versjon 1

Versjon 2 bruker den profesjonelt produserte versjonen av kortet

Figur 15: Kobling av treghetssensoren

4.4.4 Mikroprosessor

Oppkoblingen av mikroprosessoren på koblingsbrettet, figur 16, reflekterer at den kun skal brukes til å teste strømforsyning, CAN-bussen, og skriving av programvare som nevnt i figur 8. Utjevningskondensatorene er loddet på undersiden av utbrytningskortet. De to blå ledningene øverst på hver side er koblet til jord-skinna på koblingsbrettet, og den røde ledningen på midten til høyre er koblet til 3.3V-skinna. Nede til venstre er klokkekretsen, koblet til utbrytningskortet via brunt ledningspar. En grønn LED er koblet til mikroprosessorens bein 8 nede til venstre (ved en feiltakelse mangler det en 240Ω motstand mellom LED'en og jord på bildet) for å teste skriving av programvare. Hvit, svart og brun ledningtrio på vei ut av bildet nederst i midten er koblet på henholdsvis bein 7, 46 og 49 til utviklingskortet for skriving av programvare. Hvitt og gult ledningpar på vei ut av bildet nede til høyre er koblet på bein 61 og 62 til CAN-transciever. JP1 på utbrytningskortet er åpen, som vil si at bein 60 er koblet til jord via en $10k\Omega$ motstand heller enn 3.3V. Øvrige bein flyter, i motsetning til i kretsskjemaet hvor de er koblet til jord via en $40k\Omega$ motstand. Dette for å spare plass og oversikt på koblingsbrettet.



Figur 16: Oppkobling av mikroprosessoren på koblingsbrett

5 Tester

Per punkt 3 i kapittel 2.2 ble det utarbeidet valideringstester for testenhetene og kretskortene produsert for TTK4550. Disse er knyttet til kravspesifikasjonen utarbeidet for TTK4550. Dette kapittelet beskriver testene og deres gjennomførelse. Resultatene er beskrevet i kapittel 6. Kravspesifikasjonen utarbeidet for TTK4550 finnes i vedlegg A.

5.1 Beskrivelse av testnumre

Testene er ID-numre som knytter dem mot kravspesifikasjonen. Syntaksen er som følger: T-E/C-N.ID-D/P, der T står for Test, E electric, C control, N number, ID number, D design, P production. 'Electric' refererer til at testen er knyttet til et maskinvarekrav om infrastruktur for strøm/spenning, og 'Control' et programvarekrav om regulering. 'Design' refererer til testenhetene produsert for TTK8 og har som formål å verifisere nettopp designet av det relevante delsystemet, mens 'Production' omhandler kretkortene produsert for TTK4550. Det første tallet, 'N', representerer maskinvareenheten/delsystemet under test, mens det andre tallet, 'ID', representerer funksjonen testen skal verifisere på den enheten.

Test nummer **TE1.2D** leses eksempelvis som 'test nr 1 på delsystem knyttet til krav om elektrisk infrastruktur, funksjon nr 2, på designtestenhet'. Tilsvarende kan **TC3.1P** leses som 'test nr 3 på delsystem knyttet til krav om regulering, funksjon nr 2, på produsert enhet'.

Da testene ble utarbeidet ble det **ikke** lagt særlig vekt på å sørge for at P- og D-tester samsvarer, altså at eksempelvis **TC2.1D** og **TC2.1P** tester det samme for testenhet og produsert enhet. I ettertid blir det tydelig at dette hadde vært nyttig.

5.2 Beskrivelse av programvare

Flere av testene innebærer å skrive enkle testprogrammer til utviklingskortet for å kunne gjennomføres. Disse ble skrevet ved hjelp av CubeMX, et programvareverktøy for konfigurasjon av STM mikroprosessorer. CubeMX lar en pare maskinvarefunksjoner, slik som generereing av PWM-signal eller ADC-input, mot beina på mikroprosessorens pakning, for så å generere de relevante kildekodefilene i C. Dette resulterer i at selv nokså enkle tester innebærer mange filer og ofte flere hundre linjer med initialisering av registre m.v. Tester som inkluderer programvare har derfor kun relevante utdrag fra kildekoden inkludert i beskrivelsen, og kildekoden er ikke gjengitt i sin helhet noe sted i denne rapporten. Testprogrammene utgjør altså punkt 2 fra lista i kapittel 2.2.

5.3 Designtester

Designtestene utføres, som nevnt, på testenhetene produsert for TTK8. Programvare, der det er relevant, skrives til utviklingskortet STM32F303RETx-NULCEO. Testene er oppsummert i tabell 2.

5.3.1 TE1.1D: DC-buck 48-3V3

Verifisere design av lavspent kraftforsyningskrets, altså korrekt nedregulering av 48V til 3.3V.

Krav for å bestå: Måle 3.3V med voltmeter mellom jord og regulatorens 'ut'-bein.

5.3.2 TE2.1D: DC-buck 48-5V

Verifisere design av lavspent kraftforsyningskrets, altså korrekt nedregulering av 48V til 5V med L317HV.

Krav for å bestå: Måle 5V med voltmeter mellom jord og regulatorens 'ut'-bein.

5.3.3 TC1.1D: Motordriver PWM-signal

Verifisere design av motordriverkrets og programvare ved å måle PWM-regulert spenning ut fra motordriveren. En funksjon ble skrevet som lar brukeren angi hvilket PWM-register som skal skrives til, og prosentandel pulsbredde.

```
#define CLK_FQC 72000000
#define CTR_PRD 2880 //PWM-frekvens = CLK_FQC / CLK_PRD, her 25kHz

//Funksjonen
void SetPWD_DT(uint32_t* timer_counter, double pct)
{
  int ticks = (pct / 100) * CTR_PRD;

  *timer_counter = ticks;
}

//Eksempel på funksjonskall. Kanal 1 på PWM-timer 2 settes til 70%
SetPWD_DT(&(TIM2->CCR1), 70);
```

Krav for å bestå: Observere PWM-pulstog i oscilloskop.

5.3.4 TC1.2D: Motordriver PWM control

Verifisere design av motordriverkrets og programvare ved å observere bevegelse i motoren. Det kritiske momentet i denne testen er at PWM-signalet har høy nok frekvens til at motoren 'opplever' det som et konstant, jevnt likestrømssignal. Ved behov kan frekvensen justeres som beskrevet i koden i 5.3.3. PWM-signalene er satt opp i henhold til anbefalingen i motordriveres datablad kapittel 8.4.1 Bridge Control. Kodesnutten regulerer PWM-signalene slik at motoren roterer frem og tilbake og varierer hastigheten mellom ±50%. Den bygger på koden i 5.3.3

```
//Oppsett

void GoFWD(double pct){
  SetPWD_DT(&(TIM2->CCR1), 100); //Pol 1 er konstant høy
  SetPWD_DT(&(TIM2->CCR2), 100-pct); //Pol 2 er aktiv invers av hva brukeren ba om
}

void GoBWD(double pct){
  SetPWD_DT(&(TIM2->CCR2), 100);
  SetPWD_DT(&(TIM2->CCR1), 100-pct);
}

//Hovedløkke
while(1){
    double pct = 0;
    while(pct < 50){
      GoFWD(pct);
      ++pct;
      HAL_Delay(10);
    }
    while(pct > 0){
      GoFWD(pct);
```

```
        --pct;
        HAL_Delay(10);
      }
      while(pct < 50){
        GoBWD(pct);
        ++pct;
        HAL_Delay(10);
      }
      while(pct > 0){
        GoBWD(pct);
        --pct;
        HAL_Delay(10);
      }
  }
```

Krav for å bestå: Observere myk bevegelse i tilkoblet motor.

5.3.5 TC1.3D: Strømtrekk motor

Verifisere måling av motorstrømtrekk via motordriverens målebein. Motordriveren sender et strømsignal proporsjonalt med strømtrekket til motoren på et dedikert bein. Dette må leses av en ADC integrert i mikrokontrolleren og konverteres til en digital verdi. Å lage en konverteringsfunksjon fra digitalt avlest verdi til strømtrekk faller utenom omfanget av oppgaven. Koden i denne testen bruker kun funksjoner generert av CubeMX, såkalte HAL-funksjoner. Disse ble lagt inn i testen fra kapittel 5.3.4.

```
HAL_ADC_Start(&hadc2); //start konvertering fra analog til digital verdi
HAL_ADC_PollForConversion(&hadc2, HAL_MAX_DELAY); //vent på konvertering
int16_t adcval = HAL_ADC_GetValue(&hadc2); //les av resultatet fra konverteringen
UART_msg_num(adcval); //Send resultatet over UART til ekstern PC
```

Krav for å bestå: Lese strømmåling fra motordriver til ADC på mikrokontroller via UART.

5.3.6 TC2.1D: I2C-buss til IMU

Verifisere kobling av I2C til IMU ved å lese en melding over bussen. Per I2C-protokollen skal en slave-enhet sende et ACK-signal når den har mottatt en melding. Å lese ACK via oscilloskop vil derfor indikere at koblingen er gjort rett selv om dataen som sendes skulle være formatert feil i henhold til sensorens datablad. Meldingen kan altså være *mottatt* uten å være *forstått*, og det er tilstrekkelig for denne testen.

```
uin8_t SLAVE_READ = 0xD5; //Sensorens datablad spesifiserer denne
//kommandoen for å lese registre

HAL_StatusTypeDef ret; //Hjelpevariabel for avlusing
ret = HAL_I2C_Master_Transmit(&hi2c1,(uint16_t)SLAVE_READ,i2cbuf_tx,1,100); //i2cbuf_tx
//vil inneholde den spesifikke adressen det skal leses fra. Denne er her irrelevant.

if(ret != HAL_OK)
{
    UART_msg_txt("transmit failed "); //Si ifra over UART dersom mikroprosessoren ikke
    //opplever at sendingen var vellykket.
}
```

Krav for å bestå: Lese ACK fra IMU med oscilloskop

5.3.7 TC2.2D: Kontakt med IMU

Verifisere formatering av data sendt over I2C til treghetssensoren. Her utvides testen fra kapittel 5.3.6 til å inkludere formatering av data. Per sensorens datablad skal det være mulig å lese av et enkelt register kalt 'WHO_AM_I' for å bekrefte korrekt oppkobling og dataformatering. Dette registeret skal inneholde verdien **0x6A**, altså desimal 106.

```
uint8_t WHO_AM_I = 0x0F; //Registeraddresse
HAL_StatusTypeDef ret;
uint8_t i2cbuf_rx[1];

//Her spesifiseres det at det er ID-registeret som skal leses
ret = HAL_I2C_Master_Transmit(&hi2c1,(uint16_t)SLAVE_READ,&(WHO_AM_I),1,100);
if(ret != HAL_OK)
{
  UART_msg_txt("transmit failed ");
}

//Den returnerte dataen vil havne i i2cbuf_rx
ret = HAL_I2C_Master_Receive(&hi2c1, (uint16_t)SLAVE_READ, i2cbuf_rx,1,100);
if(ret != HAL_OK)
{
  UART_msg_txt("receive failed\n\r");
}
//Send innholdet av den avleste adressen til ekstrern PC over UART.
UART_msg_txt(i2cbuf_rx[0]);
```

Krav for å bestå: Lese korrekt identitet fra IMUens WHOAMI-register.

5.3.8 TC3.1D: CAN-oppkobling

Verifisere design av CAN-transcieverkrets. CAN-kommunikasjon mellom mikroprosessorene i robotarmen vil være avgjørende for at systemet virker. Dessverre ble det ikke tid til å utarbeide en test for dette delsystemet.

5.3.9 TC3.2D: CAN-buss mellom MCU

Verifisere at CAN-meldinger sendes og mottas.

Krav for å bestå: Observere korrekt dekoding av CAN-melding i mottaker.

5.3.10 TC4.1D: USB-krets

Verifisere USB-tilkobling mellom MCU og ekstern datamaskin. USB-kommunikasjon mellom den ene av robotarmens tre mikroprosessorer og ekstern datamaskin vil være svært nyttig for å kunne kommunisere med systemet. Dessverre ble det ikke tid til å utarbeide en test for dette delsystemet.

5.3.11 TC5.1D: MCU oppkobling

Verifisere design av strøm- og datakrets til mikroprosessor. Mikroprosessoren har 6 par med strømforsyningsbein (3.3V/jord). Skriving gjøres via avluseren/programmereren[2] ST-LINK som kobles til prosessoren med 5 bein, og ekstern datamaskin via USB. ST-LINK måler spenning i målenheten og gir tilbakemelding på hvorvidt skrivingen var vellykket. Det antas at dersom strømforsyningen til mikroprosessoren og/eller koblingen til ST-LINK er feil, vil ST-LINK gi en tilbakemelding som reflekterer dette.

ST-LINK-enheten er fysisk plassert på utviklingskortet, men kan kobles for å skrive til en ekstern mikroprosessor – altså en annen enn den på utviklingskortet. Det er mikroprosessoren på koblingsbrettet som brukes i denne testen.

Krav for å bestå: Vellykket skriving av programvare til mikroprosessor.

5.4 Produksjonstester

Disse testene utføres på kretskortene produsert for TTK4550. Testene er oppsummert i tabell 3.

5.4.1 TE1.1P: DC-buck 48-3V3

Verifisere produsert lavspent kraftforsyningskrets i alle tre kort, se kapittel 5.3.1

5.4.2 TE2.1P: DC-buck 48-5V

Verifisere produsert lavspent kraftforsyningskrets i alle tre kort, se kapittel 5.3.2

5.4.3 TE3.1P: Strømforsyning MCU

Verifisere strømforsyning til MCU.

Krav for å bestå: Måle 3.3V mellom henholdvis et av strømforsyningsbeina og jordbeina til mikrokontrolleren.

5.4.4 TE3.2P: Klokkekrets MCU

Verifisere klokkekrets til MCU med oscilloskop.

Krav for å bestå: Måle 16Mhz mellom klokkekrystall og jord.

5.4.5 TE4.1P: Strømforsyning motordrivere

Verifisere strømforsyning til motordrivere med voltmeter.

Krav for å bestå: Måle 3.3V mellom motordrivernes strømforsyninger og jord.

5.4.6 TE5.1P: Strømforsyning CAN-transcievere

Verifisere strømforsyning til CAN-transcievere med voltmeter.

Krav for å bestå: Måle 3.3V mellom CAN-transcieverens strømforsyning og jord.

[2]'Debugger' på engelsk, var usikker på hvilken oversettelse som er riktig

5.4.7 TC1.1P: Enkoderavlesing

Verifisere at alle 6 eksisterende enkodere virker med oscilloskop.

Krav for å bestå: Observere kvadraturpulstog ved rotasjon av motor.

5.4.8 TC1.2P: Motordriver PWM-signal

Verifisere genererte PWM-signal for motorstyring for alle 6 drivere, se kapittel 5.3.3.

5.4.9 TC1.3P: Strømtrekk motor

Verifisere måling av strømtrekk fra motor for alle 6 drivere, se kapittel 5.3.5.

5.4.10 TC2.1P: Vinkelmåling fra IMU

Verifisere vinkelmåling fra treghetssensor til MCU for alle tre treghetssensorer. Denne testen er en utvidelse av TC2.2D, se kapittel 5.3.6. Denne gangen skal treghetssensorens akselerometer initieres, og data fra den vertikale aksen (Z) leses. Dataen lagres som et 16-bits heltall på toerkomplement-format i 2 8-bits registere. Disse må leses av, og dataen konverteres tilbake til 16-bits heltall. Akselerometeret initieres til en rekkevidde på $\pm 2g$ akselerasjon, og en lesing av 1g akselerasjon vil normalt gi en rådataverdi på omkring 16384.

```
int16_t get_acc_z(void){
  uint8_t OUTZ_L_XL = 0x2C; //Nederste byte fra sensorens ADC
  uint8_t OUTZ_H_XL = 0x2D; //Øverste byte fra sensorens ADC
  uint8_t READ = 0xD5;
  uint8_t WRITE = 0xD4;

  uint8_t txbufh[1] = {OUTZ_H_XL};
  uint8_t txbufl[1] = {OUTZ_L_XL};
  uint8_t rxbufh[1];
  uint8_t rxbufl[1];

  HAL_StatusTypeDef i2cret; //HAL_OK-sjekkene sløyfet i dette utdraget.
  //Les av øverste byte fra sensoren
  i2cret = HAL_I2C_Master_Transmit(&hi2c1,(uint16_t)READ, txbufh,1,100);
  i2cret = HAL_I2C_Master_Receive(&hi2c1, (uint16_t)READ, rxbufh,1,100);

  //Les av nederste byte fra sensoren
  i2cret = HAL_I2C_Master_Transmit(&hi2c1,(uint16_t)READ, txbufl,1,100);
  i2cret = HAL_I2C_Master_Receive(&hi2c1, (uint16_t)READ, rxbufl,1,100);

  int16_t accval = (int16_t)(rxbufh[0]<<8|rxbufl[1]); //Konverter til int166

  return accval;
}
```

Krav for å bestå: Lese en rådataverdi på omkring 16384 fra sensoren.

5.4.11 TC3.1P: CAN-buss mellom MCU

Verifisere sending av CAN-meldinger mellom master og slave MCU for begge slaver. Det ble dessverre ikke tid til å implementere denne testen.

5.4.12 TC4.1P: USB-kobling

Verifisere at meldinger sendt over USB fra ekstern datamaskin til master MCU blir mottatt. Det ble dessverre ikke tid til å implementere denne testen.

5.4.13 TC5.1P: Optocoupler

Verifisere at optocoupler for håndleddsposisjon er riktig koblet. Det ble dessverre ikke tid til å implementere denne testen.

5.4.14 TC6.1P: Endestopper

Verifisere at endestopper i lineærbane er riktig koblet. Det ble dessvere ikke tid til å implementere denne testen.

5.4.15 TC7.1P: MCU oppkobling

Verifikasjon av produsert MCU-krets. Se kapittel 5.3.11.

5.4.16 TC8.1P: UART-kommunikasjon

Test av enkel komunikasjon mellom ekstern datamaskin og MCU. Det lyktes ikke å oppdrive en UART-adapter kompatibel med datamaskinen brukt i prosjektet i tide, så for å verifisere at UART er implementert korrekt leses en enkel melding mellom transmit-beinet og jord med oscilloskop. UART-leseren brukt med utviklingskortet kunne ikke uten videre modifiseres til å kommunisere med en ekstern enhet.

Krav for å bestå: Lese en UART-melding med oscilloskop.

6 Testresultater, diskusjon

Ettersom hver test er en tilnærmet lukket enhet tillates det å blande presentasjon av resultatene med en kort diskusjon der det er relevant.

6.1 Designtester

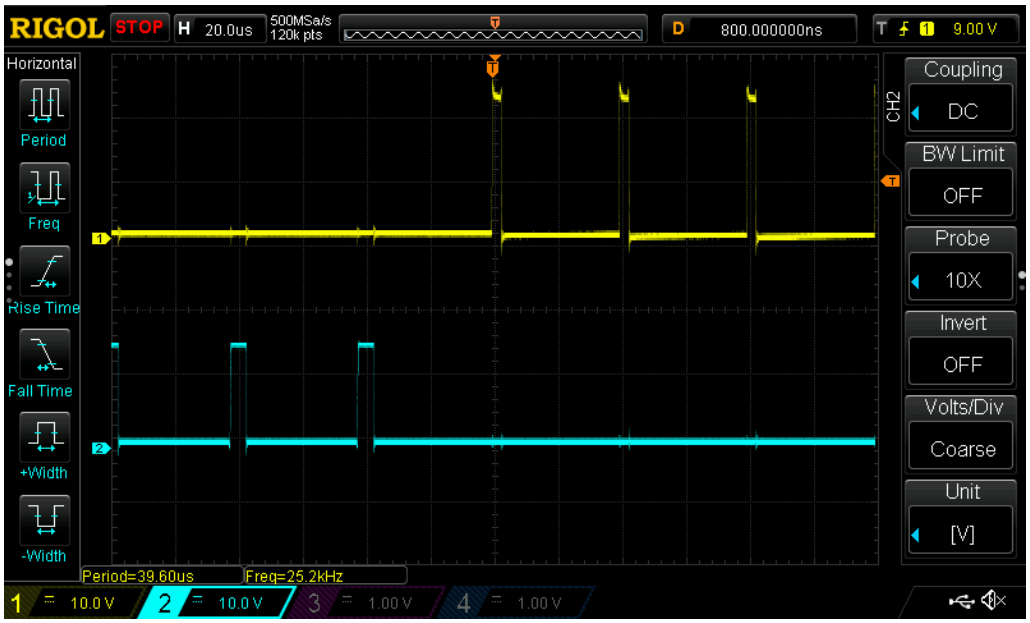
Alle tester unntatt dem som omhandler CAN og USB var vellykede. Ettersom målingene i TE-testene ble gjort med alminnelig volmeter ble det ikke tatt bilde av dem. Tabell 2 oppsummerer dette.

Testnr	**Navn**	**IC/komponent**	**Krav**	**Resultat**
TE1.1D	DC-buck 48-3V3	LM317	Måle 3.3VDC med voltmeter	OK
TE2.1D	DC-buck 48-5V	LM317	Måle 5VDC med voltmeter	OK
TC1.1D	Motordriver PWM-sig	DRV8251A	Observere PWM-pulstog i skop	OK
TC1.2D	Motor PWM control	DRV8251A	Observere bevegelse i tilkoblet motor	OK
TC1.3D	Strømtrekk motor	DRV8251A, STM32F303	Lese strømmåling fra motordriver til ADC på MCU	OK, kun rådata
TC2.1D	I2C-buss til IMU	LSM6DSM	Lese ACK fra IMU med oscilloskop	OK
TC2.2D	Kontakt med IMU	LSM6DSM	Lese identitet fra IMUens WHOAMI-register	OK, leser 6A
TC3.1D	CAN-oppkobling	TJA1057BT	TBD	
TC3.2D	CAN-buss mellom MCU	STM32F303	Observere korrekt dekoding av CAN-melding i mottaker	
TC4.1D	USB-krets	STM32F303	TBD	
TC5.1D	MCU oppkobling	STM32F303	Vellykket skriving av programvare til MCU	OK, LED blinker

Tabell 2: Oppsummering av testresultater av designtester

6.1.1 TC1.1D: Motordriver PWM-signal

Figur 17 viser PWM-signal på hver av de to inngangene til motordriveren, målt mellom utviklingskortet og motordriveren. De to kanalene er noe ute av fase, men funksjonen er korrekt. Frekvensen er målt til 25.2kHz, som også er forventet.



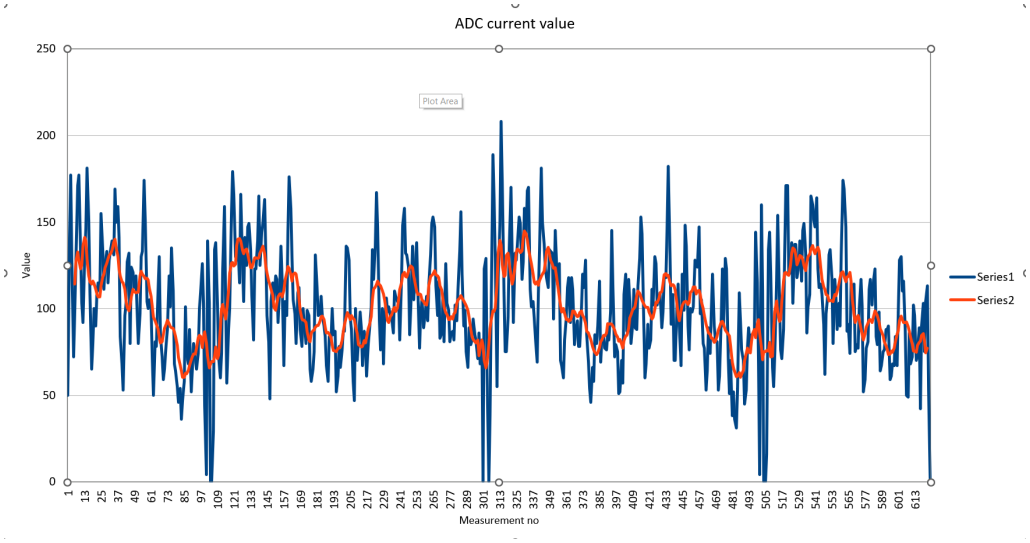
Figur 17: TC1.1D: PWM-signal målt mellom utviklingskort og motordriver

6.1.2 TC1.2D: Motor PWM-regulering

Det lyktes ikke å importere video i dette dokumentet, men håndleddet beveget seg mykt frem og tilbake under testen, utført samtidig som TC1.1D.

6.1.3 TC1.3D: Strømtrekk motor

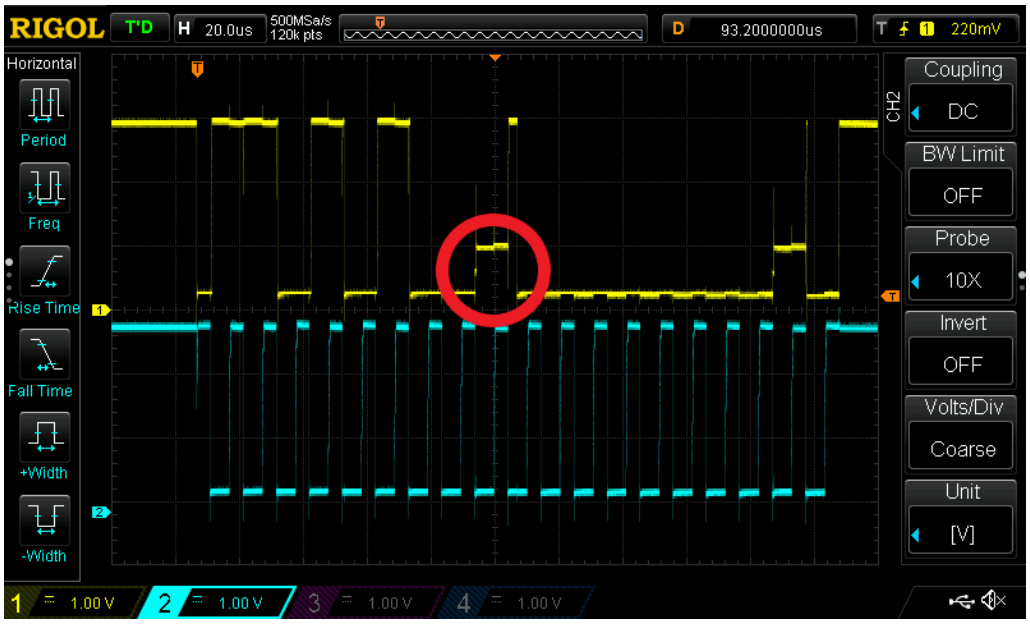
Figur 18 er basert på opptak av måling fra testen, logget gjennom UART samtidig som gjennomføringen av TC1.1D. X-aksen teller antallet målinger, omtrent 150 per sekund, mens Y-aksen er et forsøk på å vise strømtrekket i milliampere. Hvorvidt konverteringen fra råverdi til strømtrekk er korrekt vites ikke, men det kommer nokså tydelig frem av den oransje løpende gjennomsnittsmålingen at strømtrekket har det samme trekant-/sagtannmønsteret som bevegelsen til motoren skulle tilsi. Høydepunktene rundt måling 121, 325 og 529 sammenfaller med at motoren endrer retning. Testen ansees som vellykket.



Figur 18: TC1.3D: Måling av strømtrekk i motordriver

6.1.4 TC2.1D: I2C-buss til IMU

Figur 19 viser opptak fra I2C-bussen mellom treghetssensor og utviklingskort idet meldingen sendes. Den røde sirkelen antas å være sensorens ACK-signal ettersom utviklingskortet ikke avbrøt sendingen, men eksplisitt bekreftelse mangler.



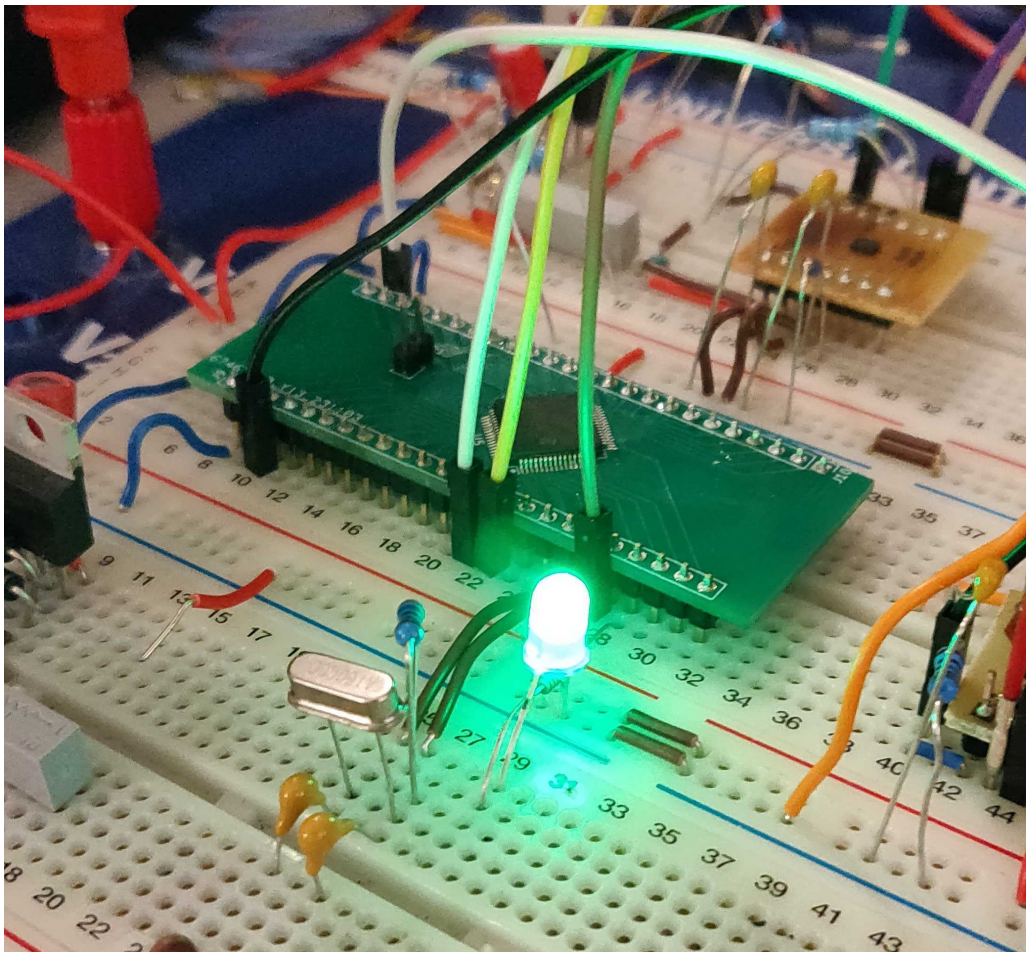
Figur 19: TC2.1D: ACK fra treghetssensor

6.1.5 TC2.2D: Kontakt med IMU

Verdien **0x6A** ble lest fra sensorens WHO_AM_I-register på addresse **0x0F**, via utviklingskortet.

6.1.6 TC5.1D: MCU oppkobling

Figur 20 viser den grønne LED'en som lyser etter at mikrokontrolleren er blitt skrevet til. I tillegg til at ST-LINK ga tilbakemelding om spenning på 3.26V over prosessoren og ingen feilmeldinger, som altså ville vært tilstrekkelig for å bestå testen, ble et enkelt program lastet opp som fikk LED'en til å blinke i intervaller på 1 sekund. Dette fungerte, og testen var vellykket.



Figur 20: TC5.1D: Oppkobling av mikrokontroller

6.2 Testing av produserte kort

Dette delkapittelet presenterer testene utført på de produserte kretskortene, som avbildet i figur 4. Dessverre var 'hand' kortsluttet mellom 48V og jord, så ingen av testene kunne utføres på dette kortet. Jevnt over var resultatene positive, men også for disse kortene falt testing av CAN-bus og USB bort på grunn av mangel på tid. I tillegg var det ikke tid til å lodde opp kortene for treghetssensorene, så test TC2.1P falt delvis bort. 'Hand' har som nevnt en treghetssensor loddet direkte på seg som kunne testet om dette delsystemet fungerer som tiltenkt, men kortet var altså kortsluttet. Videre falt TC5.1P bort ettersom kretskortprodusenten JLCPCB leverte feil kort.

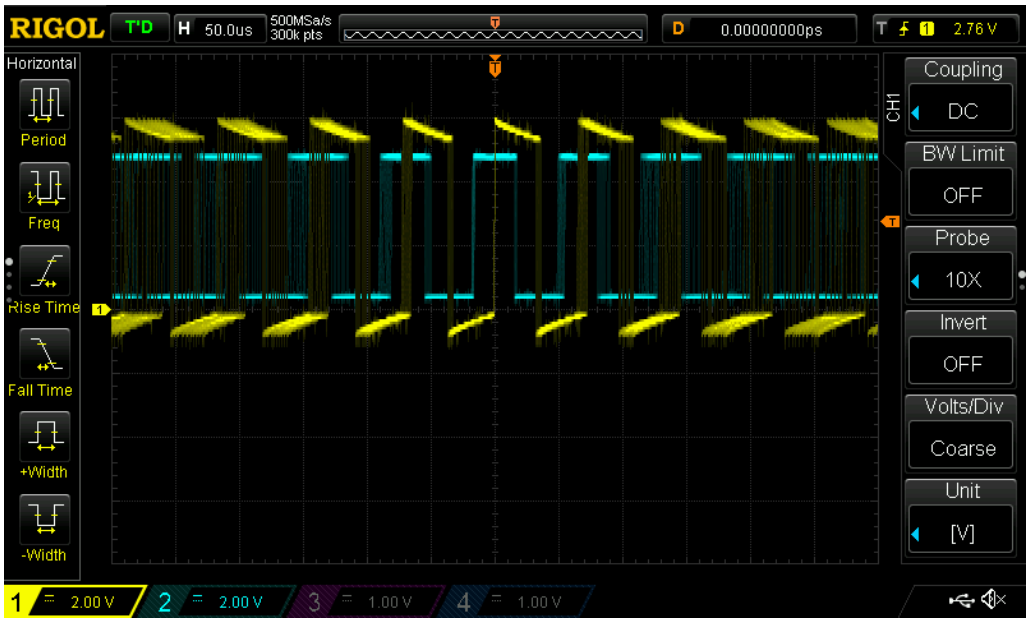
Som med designtestene ble TE-testene utført med voltmeter målt på de relevante beina uten at det ble tatt bilder. Resultatene er oppsummert i tabell 3.

Testnr	Navn	IC/komponent	Krav	Resultat
TE1.1P	DC-buck 48-3V3	LM317	Måle 3.3VDC med voltmeter	Torso, shoulder OK
TE2.1P	DC-buck 48-5V	LM317	Måle 5VDC med voltmeter	Torso, shoulder OK
TE3.1P	Strømforsyning MCU	STM32F303	Måle stabil 3.3V ved MCU for alle tre kort	Torso, shoulder OK
TE3.2P	Klokkekrets MCU	STM32F303	Måle stabil 16MHz ved MCU for alle tre kort	Torso, shoulder OK
TE4.1P	Strømforsyning motordrivere	DRV8215A	Måle stabil 3.3V ved motordrivere for alle tre kort	Torso, shoulder OK
TE5.1P	Strømforsyning CAN-transcievere	TJA1057BT	Måle stabil 3.3V ved CAN-transcievere	Torso, shoulder OK
TC1.1P	Enkoderavlesing	HEDS-9100	Observere kvadraturpulstog på oscilloskop	OK
TC1.2P	Motordriver PWM-signal	DRV8215A, STM32F303	Observere PWM-signal 0-100% på oscilloskop	Torso, shoulder OK
TC1.3P	Strømtrekk motor	DRV8215A, STM32F303	Lese måling via STM'ens ADC til terminal 0-100%	Ikke gjennomført pga manglende UART til PC
TC2.1P	Vinkelmåling fra IMU	LSM6DSM, STM32F303	Lese måling av helningsvinkel over I2C til terminal	Ikke gjennomført pga manglende tid, men fikk til i testbenk
TC3.1P	CAN-buss mellom MCU	STM32F303	Lese en CAN-melding sendt master-slave-master til terminal	Ikke gjennomgørt pga manglende tid
TC4.1P	USB-kobling	STM32F303	Lese en USB-melding sendt ekstern-master-ekstern til terminal	Ikke gjennomført pga manglende tid
TC5.1P	Optocoupler	TBD	Lese korrekt høy/lav fra optocoupler i håndledd til terminal	JLCPCB sendte feil kort
TC6.1P	Endestopper	TBD	Lese korrekt høy/lav fra endestopper i lineærbane til terminal	Ikke gjennomført pga manglende tid
TC7.1P	MCU oppkobling	STM32F303, ST-LINK	Vellykket flashing av programvare til MCU	Torso Ok
TC8.1P	Kommunikasjon	STM32F303, ST-LINK	Lese UART-melding over oscilloskop	Torso Ok

Tabell 3: Oppsummering av resultater av produkttester

6.2.1 TC1.1P: Enkoderavlesing

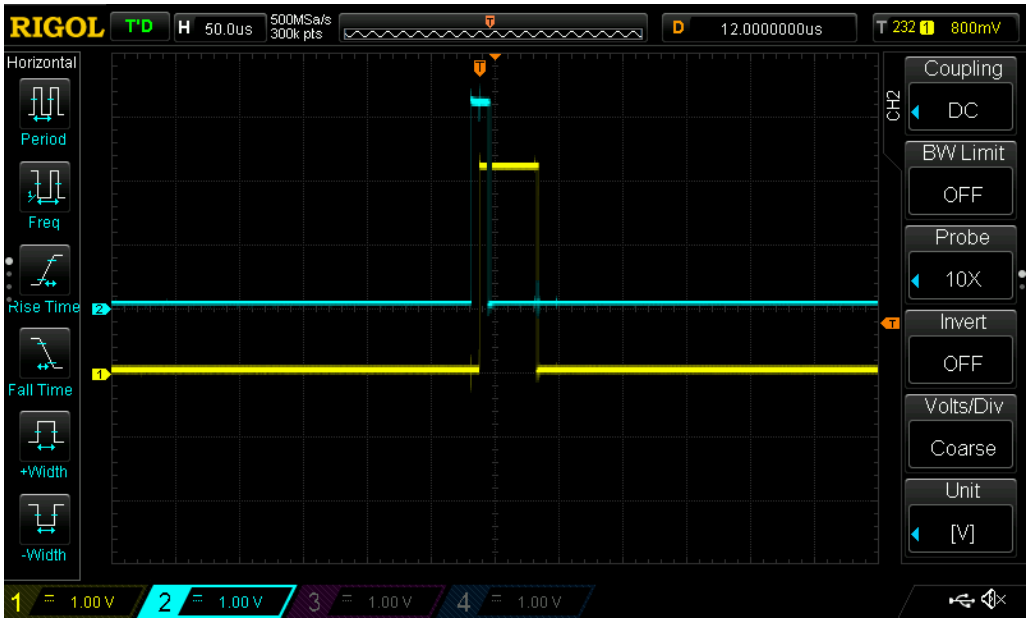
Figur 21 viser målingen fra en av enkoderne. Signalet er noe støyete, men viser tydelig 90 grader fase mellom de to kanalene.



Figur 21: TC1.1P: Enkoderavlesing

6.2.2 TC1.2P: Motordriver PWM-signal

Figur 22 viser målingen av PWM-signal mellom mikroprosessoren på 'torso' og ett av utgangssignalene på hver av motordriverne. Blå kanal viser 30% aktivering, gul 70%. Tilsvarende resultat ble registrert for 'shoulder'.



Figur 22: TC1.2P: Motodriver PWM-signal

6.2.3 TC1.3P: Strømtrekk motor

Det lyktes ikke å koble kortenes UART-bein til den eksterne datamaskinen, så denne testen falt bort.

6.2.4 TC2.1P: Vinkelmåling fra IMU

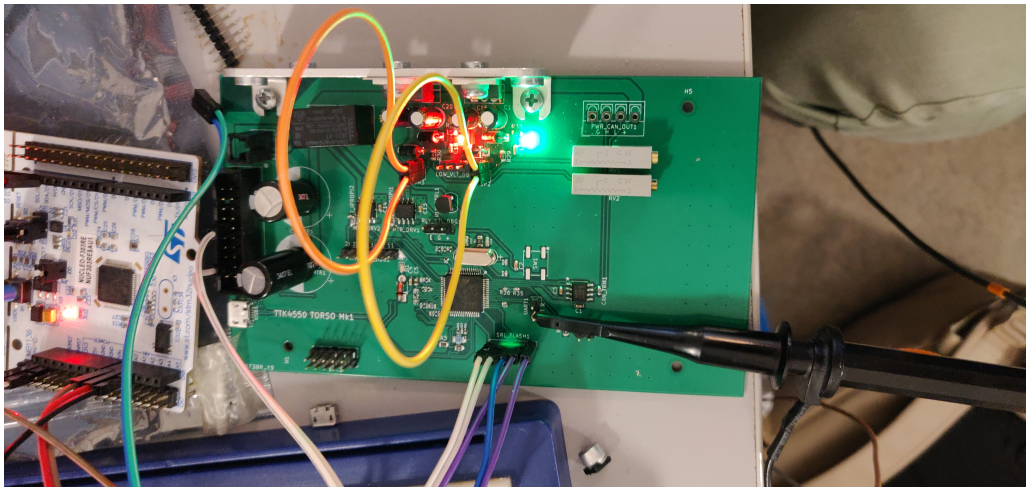
Det lyktes ikke å koble opp treghetssensoren til bruk i armen, og med feilen som ble forklart i kapittel 4.4.3 er det lite trolig at de ville fungert. Derimot ble testen gjennomført på koblingsbrettet via utviklingskortet, og dette var vellykket. Akselerasjonen når koblingsbrettet lå i ro ble målt til omkring 16400, akkurat som forventet, og ble gradvis lavere når brettet ble rotert. Det tyder på at prosedyrene for å lese av sensoren fungerer, og at denne kretsen var koblet korrekt.

6.2.5 TC7.1P: MCU oppkobling

Det ble ikke skrevet en dedikert test til dette, men de foregående testene tyder på at dette fungerer.

6.2.6 TC8.1P: UART-kommunikasjon

Selv om det ikke lyktes å koble kretskortene til den eksterne datamaskinen, kan UART leses av som bits med oscilloskop. Bokstaven 'U' har ASCII-verdi **0b01010101**, og dette ble lest av mellom transmit-beinet på torso og jord. Her burde det naturligvis vært et skjermbilde fra oscilloskopet, men figur 23 illustrerer testoppsettet. Oscilloskopproben kan sees på vei ut av bildet nede til høyre, festet ved UART-portens transmit-bein.



Figur 23: TC8.1P: UART-kommunikasjon

6.3 Øvrige funn

Det er ikke alt en kan tenke ut i forkant, særlig når en ikke har gjort noe liknende før. Under gjennomføringen av disse testene ble det oppdaget noen ting som kan være verdifulle å ta med videre, men som ikke eksplisitt ble dekket av testregimet.

- Spenningsregulatorene blir svært varme etter kort tid, selv om disse testene ble utført med en forsyningsspenning på maksimalt 30V. Differansen på omlag 25V er godt innenfor hva de er oppgitt å tåle, men det understreker viktigheten av et godt utformet varmedissipasjonssystem.
- Under test TE3.2P (Klokkekrets) av 'shoulder' viste først signalet kun støy rundt 2V. Det viste seg at motstanden i kretsen, R4 i figur 8, var $34k\Omega$ heller enn 340Ω. Testenheten på koblingsbrett ble brukt for å teste en hypotese om at dette var årsaken til feilen, og det viste seg å stemme. Kretsen fungerte som tiltenkt etter at motstanden var byttet til korrekt verdi. Denne feilen kunne ikke blitt oppdaget av TC5.1D eller TC7.1P, ettersom ST-LINK

tar kontroll over klokka til prosessoren under skriving og dermed ikke reagerer på feil i klokkekretsen.

- De to indikator-LED'ene i figur 23 er koblet i serie med en motstand på 240Ω hver seg, som vist i figur 7. De er ubehagelig lyse, og har til sammen en dissipert effekt på omlag 150mW. Til sammenlikning er mikroprosessoren estimert til å trekke rundt 50mW. Dette kan muligens forklare noe av varmeutviklingen i spenningsregulatorene. Motstandene kunne antakelig vært flere ganger høyere uten å forringe funksjonen til LED'ene.

7 Avsluttende refleksjoner

7.1 Oppsummering

Dette prosjektet har gapt over mye, og det meste er gjort for første gang (for min del, altså – jeg har ingen illusjoner rundt nyskapingsverdien av arbeidet her).

- Produksjon av utbrytningskort.
- Implementering av kretsdesign på koblingsbrett basert på anbefalinger i datablader.
- Utarbeiding av relevante tester for å verifisere feil i både test- og produserte enheter.
- Utføring av de samme testene.
- Dokumentasjon av arbeidet på en pedagogisk (eller i det minste forståelig) måte.

Jevnt over bør prosjektet kunne sies å være vellykket. Testenhetene ga en mulighet til å utbedre feil i designet der det var nødvendig, men med unntak av treghetssensoren ser designet ut til å fungere som tiltenkt. Det er uheldig at det ikke ble tid til å teste CAN-buss ettersom dette er en kritisk funksjon for robotarmen, men det er også en buss med forholdsvis enkel maskinvare i forhold til f.eks. USB. Anbefalte implementeringer presentert i komponentenes datablad er blitt fulgt etter beste evne, også for CAN-buss, og dette har fungert – det er ingen særlig grunn til å tro at det ikke også vil fungere for CAN-buss.

7.2 Lærdom

Dette prosjektet har som nevnt gått parallelt med prosjektoppgaven i TTK4550. Selv om jeg har et nokså godt kretsteknisk grunnlag fra tidligere gjennom emner på NTNU og prosjektadministrativ erfaring fra andre arenaer er robotarmprosjektet med god margin det mest komplekse jeg har gjort alene. I en ideell verden ville arbeidet med TTK8 vært nesten ferdig i god tid før kretskortene til TTK4550 ble sendt til produksjon, men realiteten er at de var ferdig loddet opp før mikroprosessoren sto i klar i koblingsbrettet. Jeg har sjeldent vært så lettet som da test *TC5.1D: MCU oppkobling* fungerte og LED'en blinket – for det betød jo at jeg etter all sannsynlighet hadde gjort det riktig på kretskortene også.

Fra før har jeg heller ikke hatt noe bevisst forhold til pull-up-mostander, og feilen med koblingen av treghetssensorene var en til dels ydmykende påminnelse om hvordan parallellkobling av motstander, pensum i første semester, fungerer. Min tilnærming til design kan oppsummeres som *modularisering og abstraksjon*: Så snart grensesnittene til en modul er definert går virkemåten ut av arbeidsminnet. I mitt hode var det derfor ikke noen forskjell på en spenningsregulator og en spenningsdeler – begge vil jo levere, i dette tilfellet, 3.3V til hva enn jeg kobler på den. Vanligvis er denne tilnærmingen en styrke, ettersom det lar meg skifte fokus mellom enheter i designet som helhet uten at detaljer fra forrige enhet blir liggende igjen i bevisstheten som et forstyrrende element. I dette tilfellet var det nettopp detaljene, altså at en spenningsdeler og spenningsregulator slett ikke er det samme, som gjorde at designet mislyktes.

Ettersom kretsene til testenhetene er basert på designene i TTK4550 kunne ikke alt arbeidet vært gjort sekvensielt, men jeg vet jo godt hvordan for eksempel V-modellen er ment å fungere. Skulle jeg gjort dette på nytt ville jeg ha brukt mer tid på verifikasjon av design før jeg gikk videre med produksjon. På den annen side, som en kollega fra Omega Verksted sa i august, tre måneder er *kort* tid til å utvikle et kretskort. I tillegg kommer momentet med blanding av arbeid mellom de to emnene. En mer iterativ prosess rundt TTK8 og TTK4550 ville kanskje gjort det vanskeligere å skille hvilket arbeid som tilhører hvilket emne.

Produksjonen av utbrytningskort tok mesteparten av tiden i dette prosjektet. Tiden fra ferdig design i KiCad til ferdig produsert utbrytningskort var omkring 2-4 timer per kort. Det var særlig etseprosessen beskrevet i kapittel 4.3.2 som tok tid, da det gjerne tok flere forsøk å oppnå et godt nok resultat. Til slutt ble det brukt profesjonelt produserte kort fra produsenten JLCPCB for mikroprosessor, treghetssensor og én av to CAN-transcievere. Tanken var at manuell etsing skulle spare tid totalt sett, men med profesjonell produksjon trengs det bare en halvtimes tid på å legge inn en bestilling. Produksjonstiden på en uke kan alltids fylles med noe annet nyttig. Dette er kanskje det enkleste punktet å være etterpåklok på. Det er spennende å lære nye (gamle) teknikker, men det tok svært mye tid i forhold til hva jeg fikk ut av det.

7.3 Nytteverdi

TTK8-prosjektet har vært temmelig nyttig. Det ga meg muligheten til å bruke nødvendig tid på å verifisere designet av kretskortene. Det ikke er sikkert at jeg hadde hatt kapasitet til det dersom jeg skulle brukt tiden på en annen fordypningsmodul, og det har nesten helt sikkert hevet kvaliteten på robotarmprosjektet som helhet. Arbeidet med kontekstdiagram og strukturert analyse av prosjektet ville til en viss grad ha foregått mentalt uavhengig av TTK8, men disse kravene til dokumentasjon har uansett gitt god anledning til å konkretisere tankene. Tiden brukt på etsing har vært et frustrasjonsmoment underveis, men at jeg ikke tenkte på å sende utbrytningskortene til produksjon tidligere får jeg nesten ta på egen kappe. Helt til slutt vil jeg også nevne at det å skrive denne rapporten på norsk har vært en interessant øvelse i å lete opp eller finne på norske oversettelser av faguttrykk jeg ellers alltid bruker engelske ord for. De har kanskje ikke fått så mye oppmerksomhet som de fortjener, men det har uansett vært en gledelig utfordring.

Bibliografi

[1] *AN4206: Getting started with STM32F303 series hardware development.* 2015.

[2] *LSM6DSM iNEMO interial module: always-on 3D accelerometer and 3D gyroscope.* 2017.

[3] *DRV8251A 4.1-A Brushed DC Motor Driver with Integrated Current Sense and Regulation.* 2022.

[4] *LMx17HV High Voltage Three-Terminal Adjustable Regulator With Overload Protection.* 2015.