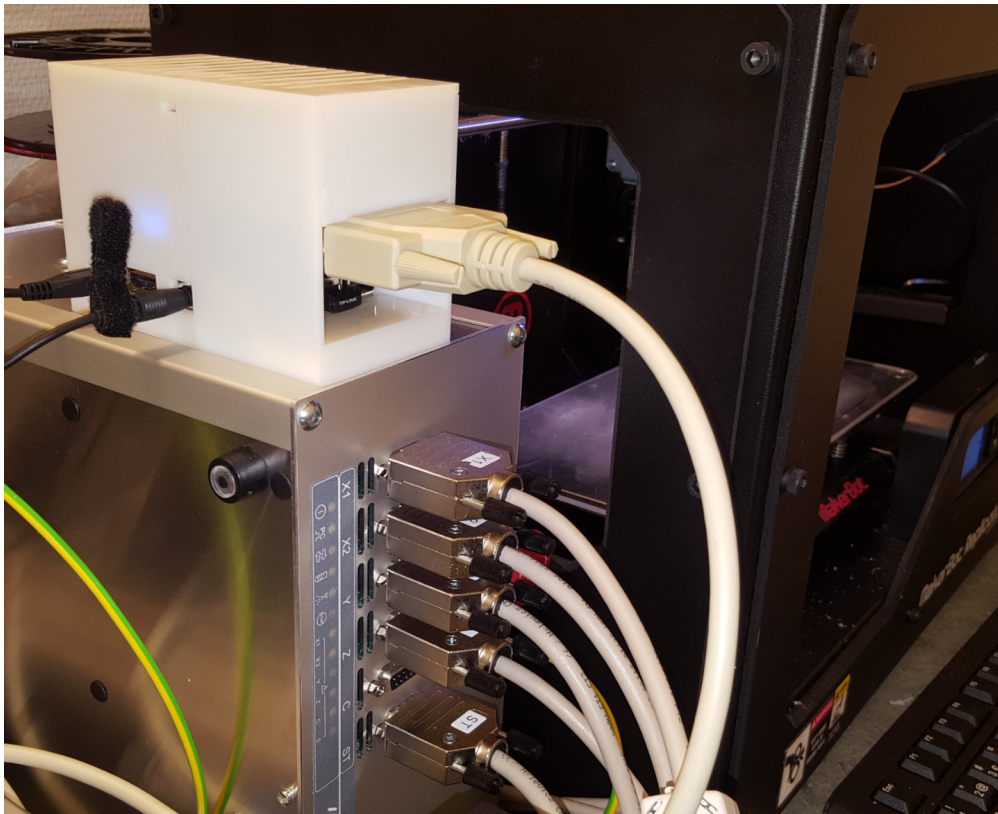




CHALMERS



Självständig G-kodsändare för styrkontroll av en CNC-maskin

Independent G-code Sender for Controlling a CNC-router

Examensarbete inom Data- och informationsteknik

FREDRIK ELFVERSSON

EXAMENSARBETE 2017

Självständig G-kodsändare för styrkontroll av en CNC-maskin

Independent G-code Sender for Controlling a CNC-router

FREDRIK ELFVERSSON



CHALMERS
UNIVERSITY OF TECHNOLOGY

Data- och informationsteknik
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2017

Självständig G-kodsändare för styrkontroll av en CNC-maskin
Independent G-code Sender for Controlling a CNC-router
FREDRIK ELFVERSSON

© FREDRIK ELFVERSSON, 2017.

Supervisor: Sakib SisteK, Data- och informationsteknik
Examiner: Peter Lundin

Institutionen för Data- och informationsteknik
Fredrik Elfversson
Chalmers University of Technology / University of Gothenburg
SE-412 96 Göteborg
Telephone +46 31 772 1000

The Author grants to Chalmers University of Technology and University of Gothenburg the non-exclusive right to publish the Work electronically and in a non-commercial purpose make it accessible on the Internet.

The Author warrants that he/she is the author to the Work, and warrants that the Work does not contain text, pictures or other material that violates copyright law. The Author shall, when transferring the rights of the Work to a third party (for example a publisher or a company), acknowledge the third party about this agreement. If the Author has signed a copyright agreement with a third party regarding the Work, the Author warrants hereby that he/she has obtained any necessary permission from this third party to let Chalmers University of Technology and University of Gothenburg store the Work electronically and make it accessible on the Internet.

Cover: CNC-controller from Heiz with the independent G-code sender on top of it, connected with a LPT connection.

Satt i text med L^AT_EX
Institutionen för Data- och informationsteknik
Gothenburg, Sweden 2017

Sammanfattning

CNC, eller *Computer Numerical Control* är ett styrsystem utvecklat för att kontrollera verkstadsmaskiner och tillåter dessa att utföra avancerade arbeten näst intill automatiskt. Följande rapport beskriver ett tillvägagångssätt för att ta fram mjukvara och hårdvara, inklusive kapsling, för tvådimensionell styrning av en datorstyrd CNC-fräs på ett så enkelt sätt som möjligt för användaren. Projektet beskriver hur styrhårdvaran blir trådlös, och hur denna sedan kommunicerar med fräsmaskinens styrenhet för att utföra snabba fräsarbeten som kräver minimal handpåläggning av användaren. Sändaren är ansluten via en parallellport med 25 ingångar till en färdig, kommersiell fräs. Den utvecklade sändarens system baserar på ett Raspberry Pi och är helt självständigt i det att användaren inte behöver några slags förinstallationer annat än en webbläsare för att ansluta till gränssnittet med hjälp av en dator, tablet, telefon eller liknande. Beroende på typ av CNC-kontroller och dess ingångar på LPT-porten kan enkla ändringar göras för att anpassa projektet till en mängd olika CNC-maskiner. Projektet är helt baserat på gratis Open-source-mjukvara och eliminerar behovet av en stor datorterminal och är istället liten, billig och kompakt. Vidare presenterar även projektet ett förslag på hur standardiserade filer i ett typ av CAD-format, DXF, kan köras direkt med hjälp av sändaren.

Abstract

CNC, or *Computer Numerical Control* is a control system designed to control workshop machines, allowing them to perform advanced tasks almost automatically. The following report describes one approach to developing software and hardware, including an enclosure, for two-dimensional control of a computer-controlled router in a way that requires as little as possible of the user. The project describes how the control hardware is made to work wirelessly, and how it then communicates with the milling router control unit to perform quick milling work that requires minimal assistance from the user. The sender is connected via a parallel port with 25 inputs to an already complete, commercial milling machine. The developed sender's system is based on a Raspberry Pi and is completely independent in a way that the user does not need any kind of preinstallations other than a web browser to connect to the sender's interface using a computer, tablet, phone, or similar. Depending on the type of CNC controller and its inputs on the parallel port, simple changes can be made to customize the project for a variety of CNC machines. The project is based entirely on free open source software and eliminates the need of a large computer terminal and instead, the solution is small, cheap and compact. In addition, the project also presents a proposal on how standardized files in one kind of CAD format, DXF, can be run directly using the sender.

Enheter och förkortningar

Symbol		Beskrivning
<i>GPIO</i>	—	General Purpose Input/Output
<i>Rx</i>	—	Serial Reciever. Mottagare för seriell kommunikation
<i>Tx</i>	—	Serial Transmitter. Sändare för seriell kommunikation
<i>G – kod/RS – 274</i>	—	Programmeringsspråk för numerisk kontroll
<i>DB25</i>	—	Parallelport med 25 ingångar
<i>CNC</i>	—	Computer Numerical Control
<i>PCB</i>	—	Mönsterkort/Tryckt krets
<i>SoC</i>	—	System on Chip. Chipbaserat system inklusive processor, GPU, lagringsminne och så vidare.
<i>Git</i>	—	Versionshanteringsprogram över internet för att hantera källkod.
<i>NPM</i>	—	Node Process Manager. Versionshanteringsprogram för javascript-applikationer.
<i>CAD</i>	—	Computer Aided Design. Mjukvara som utvecklar designfiler med exakta mått för industriell framställning.
<i>CAM</i>	—	Computer Aided Manufacturing. Mjukvara som förenklar industriella processer genom att till exempel omvandla en CAD-fil till utförbar kod för en fräsmaskin.

Innehåll

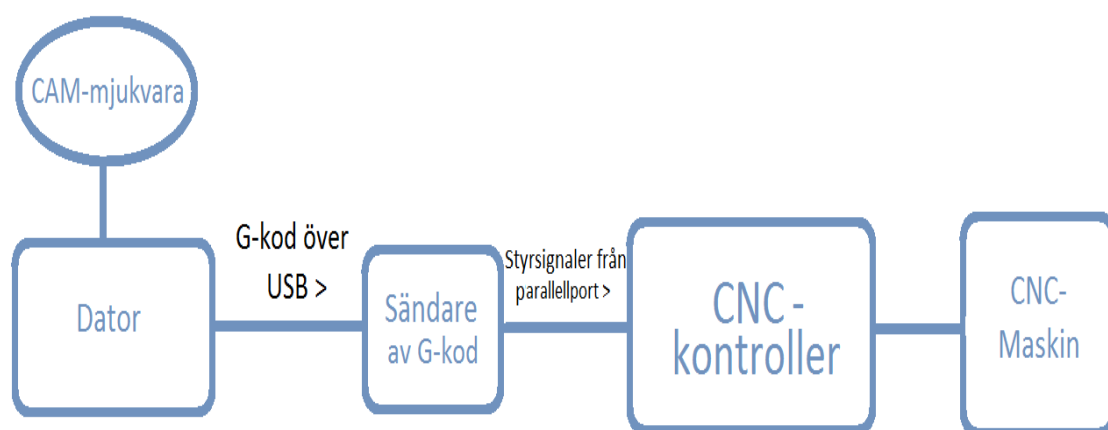
1	Introduktion	1
1.1	Bakgrund	1
1.2	Syfte	2
1.3	Mål	2
1.4	Avgränsningar	2
2	Teknisk bakgrund	4
2.1	CNC-kontroller och -maskin samt den tidigare sändaren	4
2.2	G-kod	4
2.3	Raspberry Pi	4
2.4	Gertduino	5
2.5	G25-Sköld	7
2.6	NE555 Pulsgenerator	7
2.6.1	Utträkning av kretskomponenter	8
2.7	Mjukvara	9
2.7.1	GRBL	9
2.7.2	CNC.js	9
3	Metod	10
3.1	Versionshantering	10
3.2	Hårdvaruhantering	10
3.3	Testkörning och utvecklingsverktyg	10
4	Utförande, Mjukvara	12
4.1	Grundmjukvara	12
4.1.1	Arduino IDE och initiering av Gertduino	12
4.1.2	Nätverksanslutning för Raspberry Pi	14
4.2	Initiering av GRBL	15
4.2.1	Parameterinställningar i GRBL för High-Z 720-s	16
4.3	Initiering av CNC.js	18
4.3.1	Initiering av Production Process Manager PM2	19
5	Utförande, Anslutningar	20
5.1	Systemets fullständiga uppbyggnad	20
5.2	Seriell kommunikation mellan Raspberry Pi och Gertduino	20
5.3	Gertduino till G25-sköld	21
5.3.1	Digitala portar	22
5.3.2	Analog portar	22
5.4	G25-sköld till CNC-kontroller	23
5.5	NE555 Pulsgenerator till CNC-kontroller	24

6	Kapsling	25
6.1	3D-printning av Kapsling	25
6.1.1	Dimensionering	25
7	Resultat	27
8	Diskussion och slutsatser	28
8.1	Trovärdighetsanalys	28
8.2	Förbättringar	28
8.2.1	Hårdvara och miljöaspekter	28
8.2.2	Mjukvara	29
8.2.3	Fördelar och nackdelar med bCNC kontra CNC.js	31
8.3	Slutsats	32
9	Bilaga A: Företagets typiska CAD-fil i DXF-format	35

1 | Introduktion

1.1 Bakgrund

Företaget Tele-Radio AB i Askim har nyligen inhandlat en slitstark CNC-fräsmaskin (Computer Numerical Control) för industriellt bruk: en typ av maskin som fräser ut ett objekt ur olika hårda material så som stål, trä och komposit. CNC är ett begrepp som beskriver en typ av automatisering där maskiner styrs av en mängd kommandon från, i projektets fall, kodspråket *g-kod* som skickas från någon typ av sändare, oftast en dator med USB-port. Kommandon i *g-kod*, eller RS-274 som det även benämns, består av olika X, Y och Z-koordinater och kortkommandon som kompileras i sändarens processor till styrsignaler som tilldelas olika ingångar anslutna till maskinens axlar genom spänningspulser från CNC-maskinens kontrollenhet. Kontrollenheten styrs från början av en datorterminal med tillhörande program via en logisk parallell-port med 25 ingångar. Med hjälp av kodkommandon kan fräsmaskinen fräsa ut olika former med hög noggrannhet på exempelvis en hård plastpanel, i företagets fall själva konsolen till produkten som säljs. Det nuvarande datorprogrammet som används för att styra fräsmaskinen via dator inte särskilt användarvänligt och dyrt med stort utrymme för förbättring inom exempelvis automation.



Figur 1.1: Förklaring över hur systemet som ersätts är uppbyggt.

1.2 Syfte

Syftet är att på företagets begäran förenkla maskinens styrning och utesluta behovet av en dator-terminal med tillhörande, utdaterade program och portar. Mjukvaran som skall ersättas saknar en användarvänlig, modern utformning och kräver kalibrering av hårdvara och CAD-fil innan körning, samt en dator inom kabelavstånd till fräsmaskinen. Systemet kräver dessutom egen typ av sändare som en brygga mellan dator och fräsmaskin. Denna sändare är dyr och specialanpassad för mjukvaran som den levereras med. Detta innebär ett problem eftersom bolaget behöver en lösning som tillåter att dotterbolagen så småningom även de skall kunna inhandla sina egna fräsmaskiner och kunna köra dessa direkt, snabbt och med enkelhet utan någon större utbildning om hur en CNC-maskin fungerar utifrån en CAD-fil. Projektet ämnar alltså att förkorta vägen mellan CAD-fil till färdigfräst, fysiskt objekt så att beställningar kan uppfyllas lokalt och med enkelhet av icke-kunniga inom ämnet. Den behöver ta liten plats och vara självständig på det viset att vilket operativsystem som helst kan ansluta till den utan någon installation. Den önskade funktionen ska vara en självständig, små-skalig dator som tar emot en CAD-fil trådlöst och sedan styr fräsmaskinen självständigt utan konstant övervakning eller understöd av användaren.

1.3 Mål

Målet är att skapa en komplett hård- och mjukvarulösning som:

1. Tar emot körfiler och kommunicerar trådlöst med en dator, tablet eller liknande och är självständig.
2. Omvandlar icke-kompatibla filer till ett kodformat som fräsmaskinen kan läsa.
3. Skickar filen i form av styrsignaler till maskinen.
4. Exekverar nedladdad fil så att riktiga fräsarbeten kan utföras för företaget med enkelhet och med minimal handpåläggning.

För att beskriva hur detta går till innefattar rapporten:

1. En hårdvarudel, som beskriver systemet mjukvaran kräver och är uppbyggt på.
2. En mjukvarudel, som beskriver hur filformatet omvandlas och tas emot, för att sedan skickas i realtid till maskinen.
3. Kalibrering av fräsmaskinen och mjukvaran som skickar styrsignaler i realtid för att utföra ett bra fräsjobb vid behov och inom korta tidsintervall.

1.4 Avgränsningar

Projektet kommer att använda sig av det Arduino-inspirerade Gertduino-kortet och en Raspberry Pi, samt en skräddarsydd sköld till Gertduinon. Hårdvaran skyddas av en 3d-utskrivna kapsling, som skapas som en del av projektet i 3d-manipulerande mjukvara. För mjukvarumässig konfigurerings av Raspberry PI använder sig projektet av öppen source-mjukvara som går att hämta från internet utan kostnad. För att Arduinons processor

skall styra fräsmaskinen används en klassisk och mycket populär mjukvara: *GRBL* som är open source och därför utan kostnad via github. Rapportens innehåll är avgränsat till detta innehåll.

Rapporten inkluderar dessutom inte fullständig kalibrering av hårdvara och av GRBL eftersom detta kräver många testkörningar och därmed mycket tid och tillgång till fräsmaskinen. Rapporten inkluderar endast en lösning av problemet. Problemspecifikationen är öppen och har därför fler än en lösning som presenteras i rapporten.

Fräsmaskinen saknar en höjdsensor på sin Z-axel, därmed inkluderar projektet inte en lösning för 3-dimensionella objekt med varierande yta utan endast 2-dimensionella ytor på förspecifierad höjd. Höjden för den yta som blir fräst måste justeras manuellt innan första körning genom att stega höjd-axeln fram till att den har fysisk kontakt med ytan som skall fräsas. På så vis vet maskinen vilken höjd-koordinat den skall utgå från.

Endast en av fräsmaskinens kontrollerportar används i projektet eftersom denna port innehåller de essentiella funktionerna.

2 | Teknisk bakgrund

2.1 CNC-kontroller och -maskin samt den tidigare sändaren

CNC-maskinen som används i projektet är en High-Z 720-s av det tyska företaget *CNC-Maschinenbau* som säljs tillsammans med en kompletterande CNC-kontroller av typen *Zero-3*[5]. Maskinen har fyra stycken axlar, två för x-dimensionen för förbättrad stabilitet och en axel var för y- och z-dimensionen. Maskinens stegmotorer tar emot styrsignaler via seriella utportar på CNC-kontrollern som läser insignaler i TTL-nivå från två LPT-ingångar på kontrollerns baksida. Endast en av portarna behöver användas för att CNC-kontrollern skall fungera. Den andra porten tillför ett antal extra funktioner som ej används i projektet. Maskinen styrs från början av företagets egenutvecklade mjukvara, via en sändardosa innehållande styrhårdvara, liknande den som utvecklas i projektet. Denna sändare kräver dock en seriell anslutning till en dator via en USB-kabel som också förser den med ström och är i sig därför inte självständig. Vidare kräver hårdvaran en bestämd typ av mjukvara, exempelvis den som medföljer maskinen för att kunna utföra arbeten. Denna mjukvara kräver en god förståelse av användare över hur fräsmaskinen fungerar och har dessutom ett föråldrat gränssnitt.

2.2 G-kod

G-kod, eller RS274, är ett kodspråk framtaget för att framförallt automatisera industriella verktyg. Koden består av rader i följd där varje ny rad innehåller ett stycke koordinater, ett kommando, och ytterligare parametrar så som till exempel hur hög hastighet verktyget skall stegvis förflyttas i. Vanligtvis används en CAM-mjukvara för att framställa en bana som verktyget skall förflytta sig i och utföra operationer längst med. G-koden utläses i många fall från en CAD-fil i CAM-mjukvaran eftersom CAD-ritningen innehåller de exakta mått som krävs för att utföra ett korrekt arbete för verktyget. Kodspråket används för exempelvis fräsning och hantering av laser.

2.3 Raspberry Pi

Raspberry Pi är en modulär mikrodator av *Raspberry Pi Foundation* i passande storlek för projektet. Modellen som används är modell b+ som är en revision av originalraspberrien

och ersatte denna i juli, 2014. Modell b+ processorhastighet, se 2.1, är inte hög men tillräcklig för att köra ett program som sänder g-kod och samtidigt kommunicera trådlöst via datorterminal.

Specifikationer	
SoC	Broadcom BCM2835
CPU	700 MHz Low Power ARM1176JZFS Applications Processor
GPU	Dual Core VideoCore IV® Multimedia Co-Processor
Minne (SDRAM)	512 MB (Delas med GPU)
Operativsystem	Version av Linux, körs från SD-kort
Dimensioner	85 x 56 x 17 mm
Eltillförsel	Micro USB socket 5V, 2A

Tabell 2.1: Specifikationer för Raspberry Pi modell b+ [16]

Raspberry Pi är dessutom utrustad med ett antal ingångar: fyra stycken USB-ingångar, en HDMI-ingång, en nätverks-ingång och en SD-kortläsare. För att lagra debian-operativsystemet och programvara används ett 16 GB SD-minneskort.

2.4 Gertduino

Gertduino är en typ expansionskort som monteras ovanpå en raspberry. Kortet är utvecklat av *Gert Van Loo*, en av ingenjörerna bakom raspberriens alfabårdvara. Detta kort delar många av sina egenskaper med Arduino Uno - en klassisk bräda för hobbyelektronik - men kommer med ett antal fördelar för projektet. Nedan följer brädornas specifikationer:

Specifikationer för Arduino	
Mikrokontroller	Atmega328p
Driftspänning	5 Volt
Digitala I/O pins	14 stycken, varav 6 stycker erbjuder PWM-utmatning
Analoga pins	6 stycken
Operativsystem	Version av Linux, körs från SD-kort
Dimensioner	85 x 56 x 17 mm
Eltilförsel	Micro USB socket 5V, 2A
DC Current per I/O Pin	20 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB (ATmega328P) varav 0.5 KB bootloader
SRAM	2 KB (ATmega328P)
EEPROM	1 KB (ATmega328P)
Clock Speed	16 MHz

Tabell 2.2: Specifikationer för Arduino Uno [1]

Mikrokontrollern *Atmega328p* används av både Gertduino och Arduino Uno. För att skriva till mikrokontrollerns flashminne på en Arduino används en USB-port. Gertduinon saknar en USB-port men har en GPIO-hane på kortets undersida som tillåter Gertduinon att monteras ovanpå en Raspberry Pi. Därefter går det att skriva till Gertduinons mikrokontroller via GPIO-porten. För att ansluta en Arduino till en Raspberry Pi [3] krävs ofta två olika spänningskällor på grund av de skiljande driftspänningarna, alternativt en spänningskonverterare i form av ett kort om Raspberrien skall förse Arduinon med spänning. Därför underlättar Gertduinon kraftigt genom att enkelt ansluta ovanpå raspberriens GPIO-port.

Skillnader mellan Arduino och Gertduino		
USB	Slave interface	-
Reset button	Yes	Yes
Power supply	7..12V, ~250 mA	5V Raspberry-Pi
3V3 supply	~50 mA	~150 mA.
LED's	One Not-buffered	Six Buffered
User pushbuttons	-	Två
RS232 buffer	-	Ja
Real-Time-Clock	-	Ja
Infra-red interface	-	Ja

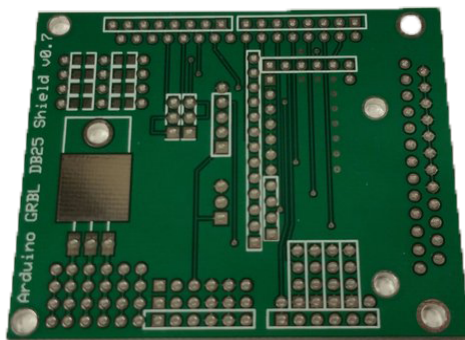
Tabell 2.3: Skillnader mellan Arduino och Gertduino [11]

Utöver *Atmega328p* är Gertduinon även utrustad med en extra mikrokontroller, en *Atmega48*. Denna kan förse Gertduinon med funktioner som en realtidsklocka med reservbatteri och en IR-mottagare samt ett antal extra I/O-portar. Dessa funktioner används ej i projektet men möjligheten att använda den extra mikrokontrollern har lämnats öppen.

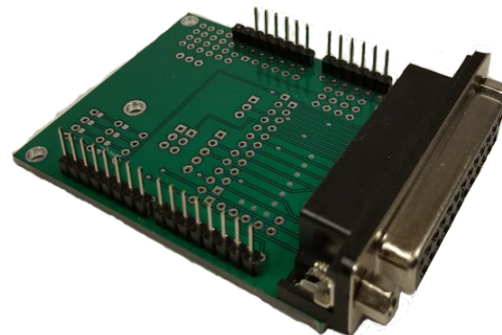
2.5 G25-Sköld

G25-skölden är ett elektroniskt kit specialutvecklat av *KC-Linear*, ett företag baserat i USA[14]. Produkten används specifikt för att styra en CNC-kontroller som får in-sig-naler via en LPT-anslutning, med en Arduino. Paketet inkluderar en DB25-hane: en 25-pin parallell-port, ett PCB (Printed Circuit Board) och kopplingslister. Kopplingslisterna löds fast för de kontaktpunkter i brädan som skall användas och som korresponderar med db25-utgångarna, se 2.1b .

Figur 2.1: G25-sköld



(a) G-25 PCB. Vita inramade områden längst med kanterna ansluts till Arduinon, medan de 25 hålen på höger sida ansluts till DB25-hanen. Bild tagen av Ron.[17]



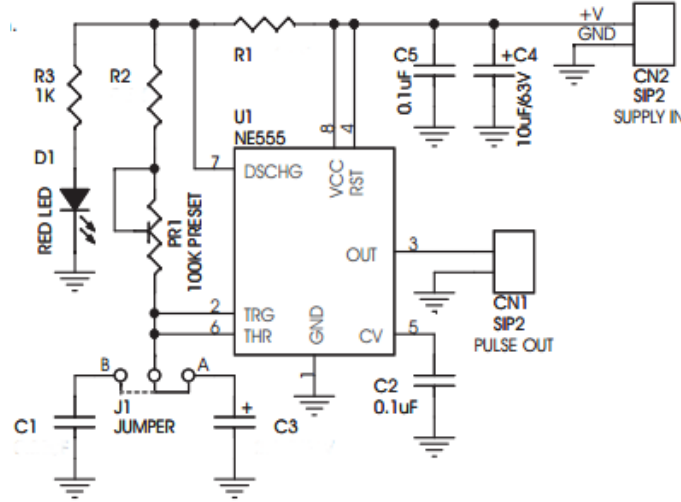
(b) Undersidan av PCB med lödad db25-kontakt och kontaktlister för att ansluta till Gertduinon. Bild tagen av Ron.[17]

2.6 NE555 Pulsgenerator

Pulsgeneratoren är en krets uppbyggd kring den klassiska 555-timern. Originalkretsen är utformad av Electrokit Eonkits [9] och används i projektet för att generera en kontinuerlig, fyrkantsvåg som slår av och på, 5/0 V, med frekvensen 12,5 kHz och en ungefärlig 50% pulskvot. CNC-kontrollern kräver en sådan signal på en av pinnarna av LPT-porten för att sättas i funktion som en säkerhetsåtgärd för att verifiera att styrmjukvaran är i gång. Kretsen innehåller en spänningsdelare med en potentiometer i serie med en av resistorerna i spänningsdelaren som kan varieras för att ändra frekvensen inom ett intervall. Detta tillåter viss justering av utsignalens frekvens för att skapa en acceptabel signal för CNC-kontrollern.

2.6.1 Utträkning av kretskomponenter

Kretsen är en enkel, astabil 555 oscillatorkrets. C5, C4 filtrerar matningsspänningen från oönskade frekvenser eftersom matningsspänningen i vissa fall riskerar vara otillräckligt reglerad. *RST* nollställer 555-timern när matningsspänningen är låg. LED-dioden används för att visa att kretsen har strömtillförsel.



Figur 2.2: Pulsgenerators krets från Eonkits manual[9].

För uträkning av pulskvoten och frekvens används t_1 för tiden då pulsen är hög och t_2 för tiden då pulsen är låg:

$$t_1 = 0.693(R_1 + R_2)C \quad (2.1)$$

$$t_2 = 0.693 * R_2 * C \quad (2.2)$$

$$T = t_1 + t_2 \quad (2.3)$$

Frekvensen för porten skall vara 12.5 kHz.

$$f = 12.5 * 10^3 = \frac{1}{T}$$

vilket leder till att:

$$T = 8 * 10^{-5} s$$

Genom att kombinera ovan formler går det att få ett uttryck för T beroende av R_1 , R_2 och C .

$$T = t_1 + t_2 = 0.693(R_1 + R_2)C + 0.693 * R_2 * C = 0.693 * C(R_1 + 2R_2) \quad (2.4)$$

och pulskvoten blir

$$\frac{R_1 + R_2}{R_1 + 2R_2} \% \quad (2.5)$$

Från 2.5 kan slutsatsen dras att pulskvoten kan ej bli lägre än 50% eftersom R_2 existerar. Vid högre resistansvärden för R_2 , blir R_1 allt mindre relevant i divisionen, därav närmar sig pulskvoten det önskade 50%-värdet. Eftersom R_2 skall vara mycket större än R_1 används därför en 100 k Ω -potentiometer, och R_1 och R_2 antar rimligt valfria grundresistansvärden 2.2 k Ω och 5.6 k Ω eftersom potentiometern vid användning ej kommer att vara nollställd. Med denna information i åtanke kan C dimensioneras. För att tillåta viss justering runtom 12.5kHz-värdet går det att vid uträkningen anta att potentiometern är inställd på 70k Ω .

$$C = \frac{T}{0.693 * (R_1 + 2R_2)} = \frac{8 * 10^{-5}}{0.693 * (2200 + 151200)} \approx 0.753nF$$

Från standardiserade kondensatorvärden approximeras detta värde till att $C = 1nF$, alltså en kondensator med koden 102. Kretsen tillåter två olika kondensatorer för att variera frekvensintervallet med hjälp av en kontaktanslutning $J1$. I projektet används bara ett frekvensintervall och kontaktanslutningen är därför satt i endast ett läge.

2.7 Mjukvara

2.7.1 GRBL

GRBL är en gratis, open-source lättvikt-mjukvara för Atmega AVR chip skapad för att kontrollera rörliga maskiner. GRBL har varit i utveckling sedan innan 2009 och har nått en nivå av utveckling som gör GRBL i många fall kraftfullare än annan mjukvara för rörelsekontroll[18]. GRBL kan användas för flera olika axelmaskiner, så som 3d-skrivare, CNC-fräsar och lasermaskiner och har därför blivit en typ av standard bland såväl hobbyister och industriella applikationer. Som indata kräver GRBL att ren g-kod skickas med exempelvis ett enkelt skript eller ett mer utvecklat GUI som i projektets fall är CNC.js. Dessa GUI kan exempelvis vara ett CAM-program eller en g-kod-sändare. GRBL kommer med ett antal inställningar som går att konfigurera med seriell kommunikation i terminalen. Se bifogad lista för inställningar. Dessa inställningar sparas mellan körningar i AVR-chippets EEPROM.

2.7.2 CNC.js

CNC.js är ett gratis, webbläsarbaserat, open-source javascript av Cheton[4], som tillåter sändning av G-kod trådlöst via en modern webbläsare. CNC.js tillåter en Raspberry att vara en lokal värd för ett GUI automatiskt från boot. Tillträde till GUI fås genom att mata in Raspberriens lokala ip följt av en valfri port, i projektets fall, port 8000. Från gränssnittet laddas en g-kodsfil upp och en visualisering av CNC-maskinens bana kan bli sedd i tre dimensioner. Med ett enkelt knapptryck påbörjas sedan fräs-operationen.

3 | Metod

Arbetet består till stor del av informationssamling från en mängd källor på internet och med testkalibrering och -körning. För att finna de komponenter som krävs har flera olika lösningar undersöks och inspiration har sökts från forum för hemmagjorda CNC-maskiner och liknande källor. Lösningen att använda GRBL valdes för att denna mjukvara är under konstant uppdatering till skillnad från andra, äldre program för gkods-sändning, samt att GRBL inte kräver ett skräddarsytt *System on Chip* utan kan istället använda sig av ett hobbykort så som en Gertduino. Detta kräver en god förståelse för hur GRBL fungerar och därför behöver en del tid spenderas för just detta i avsikt att kunna justera och kalibrera CNC-maskinen för körning med hjälp av GRBL. Raspberry Pi valdes som lösning för ett bassystem eftersom kortets utvecklingsmiljö tillåter stor flexibilitet samt om eventuella fel uppstår finns en ansevärd mängd information för felsökning tillgänglig på grund av kortets popularitet. Utöver detta har kortet trådlös nätverksanslutning och en mängd andra funktioner som gör projektet möjligt att utveckla.

3.1 Versionshantering

Versionshantering sker via Git och NPM genom Raspbian-miljön. Erfarenhet för detta saknas, så en del tid avsätts för att detta skall fungera. Vid större ändringar i mjukvara i Raspbian sparas en säkerhetskopia av SD-kortet lokalt.

3.2 Hårdvaruhantering

För G25-skölden och kretskortet för pulsgeneratoren krävs ett antal lödningar. Dessa görs med lödpenna hos Tele-radio AB. utöver dessa kräver inget av elektroniken större ingrepp. Ledningstrådar är väl isolerade och utrustade med anslutningsportar som kan anslutas och kopplas ur vid behag.

3.3 Testkörning och utvecklingsverktyg

Vid testkörning av hårdvaran för kalibrering, ansluter CNC.js till Gertduinon och CNC-maskinens axlar kan sedan stegas med hjälp av menyn i CNC.js. I tidigt utvecklingsstadium används ett externt nätverksmodem från NETGEAR för att ansluta till Raspbian-miljön utan att använda sig av en HDMI-kabel och extern mus samt tangentbord med hjälp av

ett fjärranslutet skrivbord. Med denna installation kan justeringar utföras i Raspbian med exempelvis en laptop. Ändringar i Raspbian-miljön för Raspberry-kortet görs med stöd av bash i kommandotolken. För att eventuellt justera GRBL, används CNC.js terminal för seriell kommunikation.

4 | Utförande, Mjukvara

4.1 Grundmjukvara

Rasberrien som används i projektet använder ett antal program för att skapa en god miljö för resterande mjukvara. Operativsystemet som används är *Raspbian Jessie*, en icke-officiell port av *Debian* vars kompileringsinställningar har anpassats för att skapa optimerad hard-float kod för Raspberry Pi skapad av *Raspberry Pi Foundation*. Denna optimerade användning av Rasberriens hårdvara tillåter ökad prestanda av processorn i samtliga program. För installation i Debian-miljö används bash som standard shell-språk.

4.1.1 Arduino IDE och initiering av Gertduino

För att Initiera Gertduinon för användning med Raspberry Pi krävs en initiering av Gertduinons hårdvara via Raspberry Pi samt mjukvara i Raspberry Pi som tillåter programmering av Gertduinons mikrokontroller via Raspbian-miljön. Hemsidor så som *Gordons Projects*[13] och *Friends of the Unicorn*[10] understödjer med de resurser som krävs för intiering. Mjukvaran är *Arduino Integrated Development Environment (IDE)* och används för att skriva över GRBL till Gertduinons atmega328p flash-minne. För initering av Gertduino görs följande programmering i Rasberriens shell-terminal:

```
1 sudo apt-get update
2 sudo apt-get upgrade
```

Dessa två rader av kod uppdaterar Rasberriens mjukvara. Därefter kan exempelmjukvara för Gertduino hämtas från *element14*[8], den officiella hemsidan för Gertduino, i form av en .zip-fil som senare kan användas för att verifiera att initieringen är fullgjord.

```
1 cd
2 unzip gertduino.zip
3 mv gertduino Gertduino
```

Detta byter register till Rasberriens grundregister, därefter packas .zip-filen upp och *gertduino* flyttas in i ett nytt filregister *Gertduino*. Detta görs eftersom i senare steg kommer ännu ett paket att hämtas hem från *Gordons Projects*[12] med samma namn. Eftersom Gertduinon ansluter via Rasberriens GPIO som redan är förprogrammerad med vissa funktioner, när pin 5 och 6 kortsluts av Gertduinon, aktiveras Rasberriens säkerhetsläge vid uppstart som standard. Denna funktion måste alltså avaktiveras i rasberriens konfigurationsfil. Detta görs med kodraden:

```
1 avoid_safe_mode=1
```

inuti

```
1 sudo nano /boot/config.txt
```

Därefter installeras ett antal nödvändiga paket så som en C-kompilator och Arduino IDE på systemet med följande kommando:

```
1 sudo apt-get install arduino
```

Vid nästkommande steg för installation av Arduino IDE, är programvaran *AVRdude* nödvändig för programmering av mikrokontrollern. *AVRdude* är ett verktyg som gör det möjligt att manipulera och ladda upp information i Atmel AVR-chip. En version av *AVRdude* anpassad för Gertduino samt en mjukvara för installation med *AVRdude* hämtas från *Gordons Projects*[12].

```
1 cd
2 wget -O- http://project-downloads.drogon.net/gertduino/gertduino.tgz |
   tar xzf -
3 cd gertduino
```

Ovan rader hämtar hem och extraherar den nedladdade .tgz-filen i ett nytt register med namnet *gertduino* och byter till registret. Med nödvändiga resurser extraherade krävs nu ett antal utbyten av konfigurationsfiler i systemet och *AVRdude*. Detta görs med följande shell-skript:

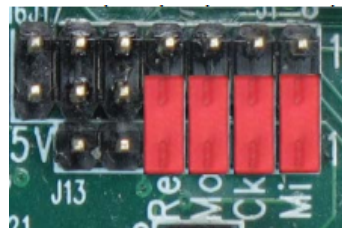
```
1 cd avrdude
2
3 # konfig. filer
4
5 sudo cp /etc/avrdude.conf /var/tmp
6 sudo cp avrdude.conf /etc
7
8 sudo cp /usr/share/arduino/hardware/arduino/boards.txt /var/tmp
9 sudo cp boards.txt /usr/share/arduino/hardware/arduino/boards.txt
10
11 sudo cp /usr/share/arduino/hardware/arduino/programmers.txt /var/tmp
12 sudo cp programmers.txt /usr/share/arduino/hardware/arduino/programmers
   .txt
13
14 # Binary
15
16 sudo cp /usr/bin/avrdude /var/tmp
17 sudo cp avrdude /usr/bin/avrdude
18
19 # Make avrdude set uid root
20
21 sudo chmod 4755 /usr/bin/avrdude
```

Skriptet kopierar och förflyttar ett flertal filer från de nedladdade resurserna in i arduino-registren och AVRdude-registren, dessa filer tillåter Arduino IDE att användas för gertduino utöver standard Arduino-chip. Därefter körs det ett medkommande skript från resurserna, *avrsetup*, som med hjälp av *AVRdude* tillåter en enkel initiering av atmega328p- och atmega48-chippen på Gertduino-brädan.

Atmega-48

Innan *avrsetup* körs krävs en ändring i hårdvaran. För att konfigurera Atmega-48. Detta görs genom att koppla ett antal pinnkontakter på gertduinons kontaktbräda enligt instruktioner från Gertduinons manual[11], se 4.1. Därefter kan *avrsetup* köras för atmega 48 med följande kommando.

```
1 ./avrsetup
```

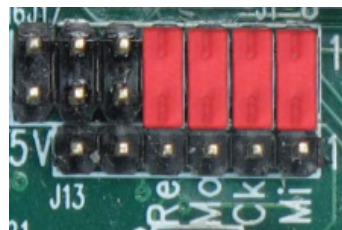


Figur 4.1: Kontaktanslutningar för konfigurering av Atmega48. Bild från Gertduino-manualen. [11]

Atmega-328p

För konfigurering av Atmega328p-chippet kopplas istället Gertduinons kontaktbräda enligt bild 4.2. Därefter kan avrsetup köras för atmega328p med följande kommando. Med dessa konfigurationer utförda är Gertduinon och Arduino IDE konfigurerade och redo för programmering.

```
1 ./avrsetup
```



Figur 4.2: Kontaktanslutningar för konfigurering av Atmega-328p. Bild från Gertduino-manualen. [11]

4.1.2 Nätverksanslutning för Raspberry Pi

För att kommunicera trådlöst med CNC.js krävs en anslutning till ett nätverk som delas med styrenheten. Styrenheten kan exempelvis vara en smarttelefon, surfplatta eller en liknande datorenhet med tillräcklig prestanda för att öppna en webbläsare. CNC-fräsen är lokaliserad i ett rum som är utan nätverksuttag, vilket gör en trådlös, WLAN-anslutning till router nödvändig. *Raspberry modell b+* saknar till skillnad från de nyutkomna versionerna av raspberry, inbyggd trådlös anslutning, således är en nätverksadapter nödvändig. Projektet använder en *TL-WN725N v.2* från företaget *TP-link* som är en liten USB-adapter som kan monteras i av de fyra USB-uttagen på Raspberrien. Adaptern har en bandbredd

på 150 Mbps, vilket är tillräckligt för ändamålet, och har låg effekt som bidrar till en lägre temperatur i Raspberriens skal. I Raspbian aktiveras WLAN i en meny från skrivbordet, dock saknas de drivrutiner som krävs för *TL-WN725N v.2* drivmodul 8188eu i operativsystemet eftersom adaptern inte är kompatibel. Detta åtgärdas med skraddarsydda drivrutiner från MrEngman på Raspberry-forumet[15]. I terminalen:

```
1 sudo wget https://dl.dropboxusercontent.com/u/80256631/install-wifi -O
  /usr/bin/install-wifi
2 sudo chmod +x /usr/bin/install-wifi
```

Ovan rader installerar ett shell-script som lagrar vilken version av operativsystemet som används, version 4.4.31 930 och därefter laddar ner och installerar motsvarande drivrutiner. Med dessa drivrutiner är nu USB-adaptren kompatibel med Raspberrien och en anslutning kan enkelt aktiveras via Raspbians rullgardinmeny för WLAN. Ett trådlöst nätverk använder sig oftast av DHCP IP-tilldelningar vilket innebär att om till exempel en omstart av nätverket sker kan raspberriens ip-adress ändras. Detta innebär att användaren måste veta om den nya IP-adressen vid varje anslutning i en webbläsare vilket är problematiskt. För att lösa detta problem kan följande kommando skrivas i terminalen:

```
1 sudo apt-get install avahi-daemon
```

Kommandot installerar mjukvaran *Avahi-daemon* som tillåter anslutning till IP-adressen med en .local adress. Som standard är denna adress *raspberrypi.local*. Projektet använder inte detta värddamn utan istället namnet *trcnc*. Värddnamnet kan enkelt justeras i raspberriens host-fil. Med *Avahi-daemon* är nu anslutningsadressen för GUI:

```
1 trcnc.local:8000
```

Denna adress går att ansluta till med samtliga webbläsarkompatibla system.

4.2 Initering av GRBL

För installation av GRBL på chippet krävs Arduino IDE. Mjukvaran införskaffas från GRBLs github[18], i projektets fall används den PWM-aktiverade versionen 1.1e som tillåter PWM-modulerad hastighet av spindeln. Den tidigare initeringen av Arduino IDE för gertduino tillåter nu direkt inskrivning av data på gertduinons atmega-chip via en rullgardinsmeny. Väl inskrivet konfirmeras GRBLs installation med hjälp av en mjukvara för seriell kommunikation, exempelvis Arduino IDE eller i projektets fall Minicom. Minicom är en terminalbaserad applikation som inkluderas i Raspbian.

```
1 sudo apt-get install minicom
```

Minicom installeras med ovan kommando. Därefter används applikationen med ett enkelt kommando:

```
1 minicom -b 115200 -o -D /dev/ttyAMA0
```

B-kommandot följs av 115200, vilket är den baudrate GRBL kommunicerar med seriellt mellan Raspberry och Gertduino. Se kapitel 5.2 för ytterligare information. Därefter beskrivs den port som skall testas. I projektets fall är porten /dev/ttyAMA0, vilket syftar

på Raspberries seriella utport på GPIO-pinnarna. Andra typer seriella utportar kan även de användas för kommunikation mellan en Raspberry och en dator, exempelvis /dev/ttyUSB0, som syftar på en av Raspberriens usb-portar. Med GRBL installerat är systemet redo för ett GUI för tydlig sändning av g-code.

4.2.1 Parameterinställningar i GRBL för High-Z 720-s

För att GRBL skall fungera optimalt med CNC-maskinen krävs ett antal inställningar av parametrar så som hastighet, acceleration och antal steg per millimeter. GRBL behöver även information om hur stor arbetsyta CNC-maskinen tillåter. I avsnittet anslutningar, se kapitel 4, går det att notera att maskinen är utrustad med så kallade limit-switches. Dessa sensorer är anslutna vid slutvärden i avståndintervallen av varje axel och ger av en signal om någon av axlarna har nått sin maximala sträcka i intervallet. GRBL kan konfigureras för att implementera dessa för en mer säker och hållbar operation men är ej nödvändiga. Projektet tillåter dessa att användas och portarna är anslutna men konfigurering av parametrarna exkluderas på grund av projektets avgränsningar. Parametervärden matas in i terminalen i CNC.js över seriell kommunikation och sparas i gertduinons EEPROM-minne. I terminalen kan följande kommando skrivas in:

```
1 $$
```

GRBL svarar då med en komplett lista över kommandon och standardinställningar för en typisk CNC-hårdvara:

```
1 $0=10 (step pulse , usec)
2 $1=25 (step idle delay , msec)
3 $2=0 (step port invert mask:00000000)
4 $3=6 (dir port invert mask:00000110)
5 $4=0 (step enable invert , bool)
6 $5=0 (limit pins invert , bool)
7 $6=0 (probe pin invert , bool)
8 $10=3 (status report mask:00000011)
9 $11=0.020 (junction deviation , mm)
10 $12=0.002 (arc tolerance , mm)
11 $13=0 (report inches , bool)
12 $20=0 (soft limits , bool)
13 $21=0 (hard limits , bool)
14 $22=0 (homing cycle , bool)
15 $23=1 (homing dir invert mask:00000001)
16 $24=50.000 (homing feed , mm/min)
17 $25=635.000 (homing seek , mm/min)
18 $26=250 (homing debounce , msec)
19 $27=1.000 (homing pull-off , mm)
20 $100=314.961 (x , step/mm)
21 $101=314.961 (y , step/mm)
22 $102=314.961 (z , step/mm)
23 $110=635.000 (x max rate , mm/min)
24 $111=635.000 (y max rate , mm/min)
25 $112=635.000 (z max rate , mm/min)
26 $120=50.000 (x accel , mm/sec^2)
27 $121=50.000 (y accel , mm/sec^2)
28 $122=50.000 (z accel , mm/sec^2)
29 $130=225.000 (x max travel , mm)
30 $131=125.000 (y max travel , mm)
```

```
31 $132=170.000 (z max travel , mm)
```

Från manualen för *CNC High-Z 720-s*[7] går det att notera att stegpulslängden och stegpulsfördröjningen är korrekt inställda för maskinen. Det går dessutom att se från CNC-kontrollerns manual[6] och från testkörning att inga av portarna behöver inverteras och justeras för NC till NO eller vice versa. Inställningar 100-132 justeras då arbetsytan är större och hastigheter och acceleration skiljer sig. Enligt manualen för *CNC High-Z 720-s*[7] sker 1600 pulser per revolution av axlarna. Detta stämmer för samtliga axlar. Manualen konstaterar även att avståndet mellan gängorna på axlarna är 6 mm. Därav är antal steg per mm:

$$1600/6 = 266.66 \text{ steg/mm}$$

Detta gäller för samtliga axlar. Därav skrivs följande kommandon in i GRBL:

```
1 $100=266.66 (x , step /mm)
2 $101=266.66 (y , step /mm)
3 $102=266.66 (z , step /mm)
```

Max rate syftar på hur hög hastighet varje axel kan färdas i. Standardvärdet är mycket lägre än CNC-maskinens hastighet. Enligt manualen är detta värde 66 mm per sekund, eller 3960 mm per minut för X- och Y-axeln och 1800 mm per minut för Z-axeln. För att inte slita på axlarna och för att ta hänsyn till bormunstyckets vikt och friktionen vid fräsning bör detta värde sättas till ett lägre värde. Efter ett antal operationer kan detta värde höjas om risken anses vara liten. Därav skrivs följande kommandon in i GRBL:

```
1 $110=2000 (x max rate , mm/min)
2 $111=2000 (y max rate , mm/min)
3 $112=1500 (z max rate , mm/min)
```

Accelerationen kan konfigureras till ett högre värde om så önskas men fungerar väl vid testkörning. Därav exkluderas konfigurering av dessa parametrar. För att GRBL skall få en uppfattning om hur stor den egentliga arbetsytan är måste parametrarna 130-132 justeras enligt de egentliga värdena. Dessa parametrar beskriver färdavstånden för varje axel och tillåter ett antal säkerhetsfunktioner i GRBL att fungera så som *Soft limits* som avbryter en operation om någon av axlarna rör sig utanför avstånden. Enligt manualen är dessa värden för X-axlarna 720 mm, Y-axeln 420 mm och Z-axeln 110 mm. Därav skrivs följande kommandon in i GRBL:

```
1 $130=720 (x max travel , mm)
2 $131=420 (y max travel , mm)
3 $132=110 (z max travel , mm)
```

Homing Cycle och Hard Limits

När en CNC-fräs används för första gången eller om ett fatalt fel inträffar och maskinen tappar steg mellan GRBL och den fysiska hårdvaran krävs att en så kallad *Homing Cycle* genomförs. Detta tillåter GRBL att veta precis var bormstycket är positionerat relativt till maskinens nollkoordinater. Cykeln matar med bestämd hastighet ut pulser på först X- samt Y-axeln framtill det att gränssensorerna drar de aktivt höga signalerna på gertduinons Limit-pinnar till jord och notifierar GRBL att respektive axel har nått sin nollpunkt.

Dessa pinnar ligger normalt höga runt 5 V med hjälp av gertduinons interna pull up resistorer. För att konfigurera denna cykel bör först en långsam sökningshastighet ges så att användaren kan försäkra sig om att sensorerna kommer att ge i från sig en signal i tid, och förhindra axlarna från att röra sig innan dessa når sin ändpunkt vilket kan innebära allvarliga skador på hårdvaran. Utöver detta justeras *homing dir invert mask* med en bitmask som förklarar vilken av axlarnas nollpunkter är positionerade i en negativ riktning. Detta förhindrar en homing cycle från att köra axlarna åt fel håll, bort från sensorerna. *Homing debounce* ges i millisekunder och behöver justeras vid testkörning så att när sensorerna aktiveras flyttas axlarna bort från nollpunkten, utom sensorernas räckvidd, innan dessa ger en ytterligare signal och oupphörligt låser axeln nära nollpunkten.

```
1 $21=1 (hard limits , bool)
2 $22=1 (homing cycle , bool)
3 $23=3 (homing dir invert mask:00000001)
4 $24=100 (homing feed , mm/min)
5 $25=200 (homing seek , mm/min)
6 $26=250 (homing debounce , msec)
7 $27=10 (homing pull-off , mm)
```

Ovan värden skickas till GRBL och aktiverar både Hard limits och Homing-funktionen. Dessa värden gav vid testkörning ett välfungerande resultat. Siffran 3 motsvarar binärt en etta på den minst signifikanta biten och den näst minst signifikanta biten vilket indikerar att x- och y-axelns nollpunkter är positionerade i motsatt riktning från det GRBL är inställd på som standard.

4.3 Initiering av CNC.js

För användning av programvaran krävs en förberedelse av systemet som skall vara uppdaterat och uppgraderat. Raspberrien kräver ett antal mjukvaror för att kunna kommunicera direkt till Git-API:et. Detta görs med kommandot:

```
1 sudo apt-get install -y build-essential git
```

Kommandot installerar alla de debian-baserade, essentiella mjukvaror som krävs för att installera git-paketet. Git-paketet har ett beroendeförhållande som build-essential registrerar och anpassar sin installation för. Därefter krävs en installation av en JavaScript-miljö med namnet Node.js som CNC.js körs i. Nedhämtning och installation görs manuellt:

```
1 wget https://nodejs.org/dist/v4.5.0/node-v4.5.0-linux-armv61.tar.xz
2 tar -xvf node-v4.5.0-linux-armv61.tar.xz
3 cd node-v4.5.0-linux-armv61
4 sudo cp -R * /usr/local/
```

Node.js är ett processorkrävande skript och körs bäst av moderna Raspberries. Projektet använder en äldre modell med den tidigare ARM-arkitekturen av AVR-chip, nämligen ARMv6 istället för ARMv7[16] och kräver därför en tidigare version av node.js, anpassad för arkitekturen. Första raden hämtar versionen och resterande tre rader av kod öppnar och extraherar tar.xz-filen till ett nytt register med ett likadant namn, går in i registret och kopierar samt installerar innehållet i */usr/local/*. Med node-miljön går det nu att installera node-paket, det vill säga komplett mjukvara i miljön, i projektets fall CNC.js, direkt

till systemet med hjälp av det inkluderade *Node Package Manager (NPM)*. Med NPM uppdaterat går det nu att med endast ett kommando installera sagd programvara.

```
1 sudo npm install -g cncjs --unsafe-perm
```

Installationen är en global sådan och behöver därför tillgång till register som kräver root-tillstånd, därför används ändelsen *--unsafe-perm* i kommandot som ger installationen fullständigt tillstånd.

4.3.1 Initiering av Production Process Manager PM2

Production Process Manager tillåter CNC.js att automatiskt starta på en valfri port då raspberrien bootar, så att denna kan anslutas till vid användning genom webbläsarens adressfält.

```
1 sudo npm install -g pm2
```

Ovan kodrad installerar pm2 med hjälp av NPM globalt med root-tillstånd.

```
1 pm2 startup # To start PM2 as pi / current user
```

Production Process Manager startas efter installation.

```
1 sudo su -c "env PATH=$PATH:/usr/bin pm2 startup linux -u pi --hp /  
home/pi"
```

Med ovankod aktiveras med su-kommandot en root-session genom att byta till rootanvändaren och kommandot körs.

```
1 pm2 start $(which cnc) -- --port 8000
```

Production Process Manager startar CNC.js på port 8000.

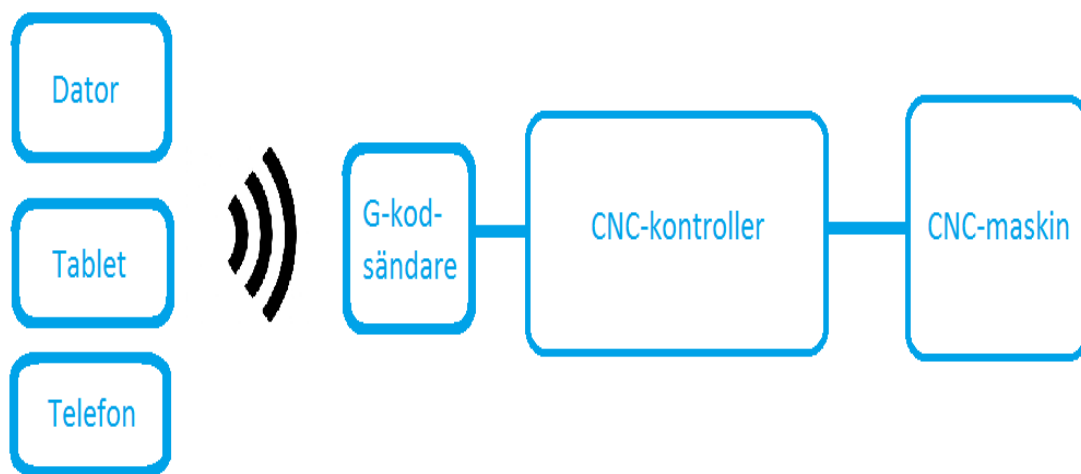
```
1 # Set current running apps to startup  
2 pm2 save
```

Applikationer som nuvarande körs via *Production Process Manager* sparas så att dessa initieras varje gång Raspbian startas.

5 | Utförande, Anslutningar

5.1 Systemets fullständiga uppbyggnad

Det nya systemet är trådlöst och G-kodsändaren kommer därför kunna kommunicera med den enhet som används för att kontrollera maskinen och ladda upp filen via WLAN. CNC.js tillåter flera användare att ansluta till gränssnittet på samma gång. Den nya sändarens hårdvara är liten och kompakt och samtliga delar lagras ovanpå varandra i en stack inuti den kapsling som beskrivs i sektion 6.1.



Figur 5.1: Förklaring över hur det nya systemet är uppbyggt.

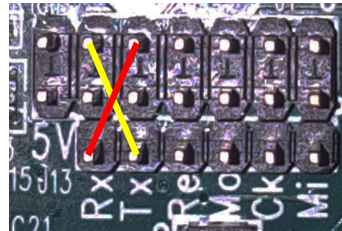
5.2 Seriell kommunikation mellan Raspberry Pi och Gertduino

För att raspberrien skall kommunicera med gertduinons atmega328p krävs en seriell kontakt mellan de bägge korten. Seriell kommunikation är en typ av kommunikation där sändaren endast skickar en bit av information åt gången till mottagaren, istället för exempelvis parallell kommunikation där sändaren kan skicka flera bitar via flera portar till mottagarens flera in-portar. Hastigheten, eller så kallad *baud rate* som bitarna skickas i, beror av raspberriens processorhastighet och hur hög hastighet GRBL kan läsa. Enligt GRBL wiki-hemsida är baud-hastigheten 115200 bitar per sekund för effektiv kommuni-

kation[18]. Baud-hastigheten justeras i CNC.js. Två av GPIO-portens 26 pinnar används för seriell kommunikation. I mjukvaran representeras dessa med följande:

```
1 /dev/ttyAMA0
```

En av pinnarna är Rx som tar emot seriell data från en extern källa, exempelvis ett expansionskort, medan den andra är Tx som skickar seriell data från rasberrien. Gertduino-kortet är även det utrustad med dessa portar. Således ansluts rasberriens Tx till gertduinons Rx och gertduinons Tx till rasberriens Rx . Detta görs på gertduinokortet med två ledningskablar enligt figur 5.2.

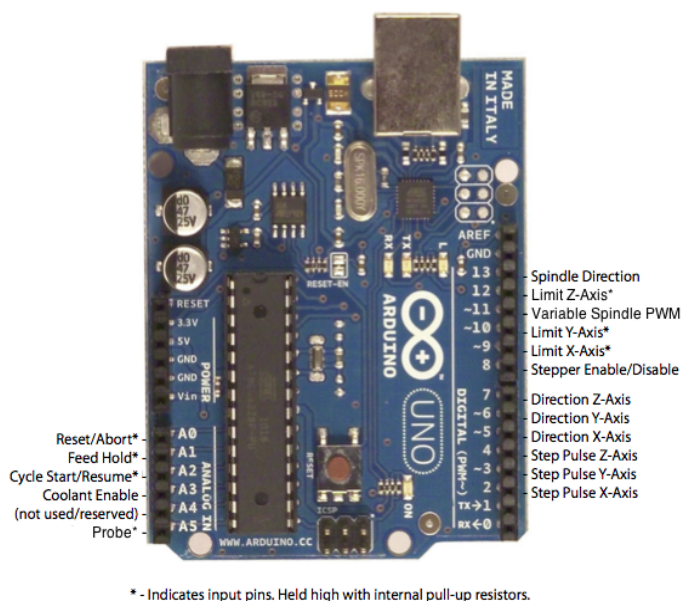


Figur 5.2: Seriella ingångar och utgångar, Rx och Tx på Gertduino-brädan.

5.3 Gertduino till G25-sköld

Gertduinons atmega AVR-chip, skriven med GRBL v1.1e, har en standardutformning som används i projektet, se figur 5.3a som är direkt ansluten till G25-skölden, se tidigare figur 2.1b.

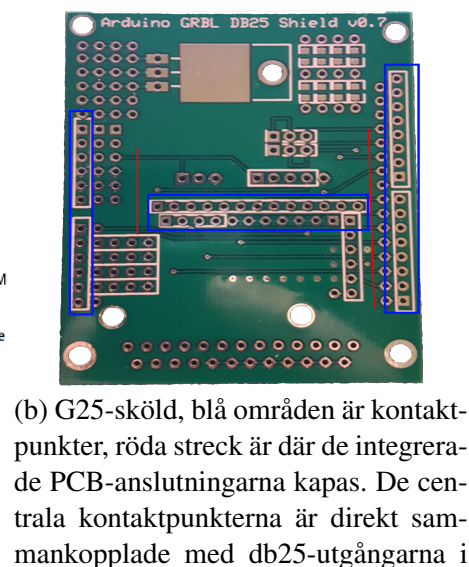
Figur 5.3: G25- och GRBL-anslutningar



* - Indicates input pins. Held high with internal pull-up resistors.

(a) GRBLs PWM-layout i version v1.1e, bild från GRBL PCB.

Wiki [18]



(b) G25-sköld, blå områden är kontaktpunkter, röda streck är där de integrerade PCB-anslutningarna kapas. De centrala kontaktpunkterna är direkt sammankopplade med db25-utgångarna i

5.3.1 Digitala portar

D2- till och med D4-utgångarna skickar ut cirka 1300 pulser per sekund som standard, vilket är snabbare än den tidigare G-kodsändaren som användes av cnc-kontrollern normalt. Pulsbredden är dessutom kortare, vilket går att justera i GRBL. Den kortare pulsbredden och en högre mängd pulser leder till ett mer responsivt system som reagerar med mindre tidsförskjutning vid plötsliga ändringar i G-kod, exempelvis ett nödstopp eller en sträcka som ändrar riktning ofta under en kort tid. Motorn stegar sin axel vid varje stegpuls, vid pulsens stigning.

D5-D7 skickar en något längre varande puls som förklarar för cnc-kontrollern att riktningen skall ändras. Längden bestäms av hur responsiva axlarna är. Vid varje puls ändrar axelns stegmotor sin rotationsriktning. D8, Stepper Enable/Disable används som ett säkerhetsverktyg vid projekt som använder egna stegdrivers, för att informera när stegmotorerna skall vara aktiverade. Denna port används ej i projektet, eftersom den redan färdigbyggda cnc-kontrollern hanterar denna typ av hårdvarukontroll.

Limit-portarna D9, D10 och D12 används för att informera CNC-kontrollern om när en axel har nått slutet av ett fast avståndsintervall, det vill säga axlarnas gränsavstånd. Genom att konfigurera dessa hard-limits i GRBL, kan gkod-sändaren anpassa hur borret skall röra sig relativt till axlarnas gränser och avbryta operation om limit-signalen aktiveras. Dessa har ett konstant högt spänningsvärde. När en av sensorerna vid axlarnas slut aktiveras, kopplas CNC-kontrollerns normalt öppna limit-port till jord, vilket ger limit-portarna på Gertduinon ett lågt spänningsvärde och informerar GRBL om att en sensor aktiverats vilket sätter programmet i nödstoppsläge.

Port D11 hanterar hastigheten på spindeln, exempelvis i ett intervall mellan 1000-3000 RPM. Den gör detta genom så kallad *Pulse Width Modulation (PWM)*. Beroende på hur stor duty cycle signalen som skickas till denna port har, desto snabbare roterar spindeln. En hög duty cycle ger en hög spindelhastighet och vice versa. Företaget använder sig ej av en spindel som tillåter sagd funktion, men projektet har funktionen aktiverad om detta eventuellt önskas ändras i framtiden.

5.3.2 Analoga portar

A0-A5 är analoga signaler. Olika CNC-installeringar har beroende på typ av operation olika krav. En installation som fräser i mycket hårda material under en längre tid kan exempelvis behöva en kylningsfunktion som till exempel sprutar ut en konstant vattendimma vid munstycket. G-kod har kommandon för att aktivera dessa under längre och kortare tid och för detta används *Coolant Enable* porten. Projektet använder ej denna funktion eftersom materialen som används är relativt mjuka och operationerna pågår i kort tid med lång viloperiod mellan varje ny operation. *Probe*-funktionen kan användas för automatiskt lokalisera vilket avstånd på Z-axeln objektet är från bormunstycket. ett exempel med g-kod:

```
1 G38.2 X20 Y15 Z-100
```

Där siffrorna är i enheten millimeter. Z-100-kommandot sänker munstycket med 100 mm. Om munstycket kommer i kontakt med en yta innan gränsvillkoret har uppfyllts sätts Probe-porten till jord och Z-avståndet sparas i EEPROM-minnet. För att denna funktion skall vara i funktion, krävs att en probe-grind är ansluten till porten för att leda om till jord. Projektet använder ej denna funktion och porten lämnas därför öppen. *Feed Hold* används för att informera CNC-kontrollern att införa ett tillfälligt uppehåll av inmatning av G-kod och g-kodsströmmen avbryts tillfälligt men behåller sin position i koden. *Reset/Abort* nollställer CNC-kontrollen och bryter strömmen av g-kod framtill att en ny ström påbörjas.

5.4 G25-sköld till CNC-kontroller

G25-skölden har med tryckta kretsar en standardutformning som leder dessa till särskilda portar på db25-hanen via den centrala kontaktpunkten. Denna utformning är anpassad för hur den typiska CNC-kontrollern med LPT-port ser ut. I projektets fall är portarna för db25-grinden annorlunda, se tabell 5.1 för deras respektive användning. Därav kapas de integrerade kretsarna från originalpunkten, se 5.3b till den centrala kontaktpunkten som leder direkt till DB-25 och speglar utportarnas utformning. Vid denna centrala punkt löds istället kontakter fast och anslutningarna till de digitala och analoga utportarna på gertduino görs därefter med extern ledningstråd.

Pin CNC	Pin Gertduino	Beskrivning
1	D13	Spindelrelä
2	D5	Dir. X-axel
3	D2	Step X-axel
4	D6	Dir. Y-axel
5	D3	Step Y-axel
6	D7	Dir. Z-axel
7	D4	Step. Z-axel
8	-	-
9	-	-
10	D12	Limit Z
11	A0	E-stopp
12	D10	Limit Y
13	D9	Limit X

Pin CNC	Pin Gertduino	Beskrivning
14	A3	Kylning
15	A5	Probe
16	-	Extern 12.5 kHz-puls
17	D11	Spindel-PWM
18 - 25	GND	Jord

Tabell 5.1: Tabell över port/pin-anslutningar från Gertduino till CNC-kontrollerns LPT-port.

5.5 NE555 Pulsgenerator till CNC-kontroller

Pulsgeneratorns utport ansluts till pinne 16 på LPT-porten via g25-skölden. De två jordutgångarna i kretsen ansluts till raspberriens jord via g25-skölden för att skapa en gemensam sådan. Spänningen tillförs till kretsen med raspberriens 5 V utport, se figur 5.3a. Således ges en konstant 12.5 kHz signal på pinne 16 av LPT-porten från det att systemet är i gång vilket aktiverar CNC-kontrollern. Utan denna signal är kontrollerns huvudkrets spänningslös och signalen agerar alltså som en verifikation att programmet och portarna är redo för att ett arbete skall utföras.

6 | Kapsling

6.1 3D-printning av Kapsling

För att skapa en säker och robust kapsling för systemet används applikationen *123D Design* från Autodesk [2]. Programmet erbjuder en intuitiv 3d-utvecklingsmiljö med grundläggande verktyg och kan exportera i .stl, ett filformat som används av 3d-printerprogram. Kapslingens grundläggande formgivning är inspirerad av Wandrsons design från webbsidan Thingiverse [19].

6.1.1 Dimensionering

Vid utveckling och dimensionering av en kapsling måste ett flertal faktorer tas hänsyn till så som rum- och systemtemperatur, risk för partiklar eller vatten, samt storlek. Om raspberriens system på kortet når en ungefärlig temperatur av 80 grader celsius, riskerar systemet att strypas. Vid normal rumtemperatur och operation når systemet inte denna temperatur och därav krävs ingen fläkt. En fläkt innebär dessutom komplikationer eftersom en sådan komponent även den har värmeförluster och kräver ytterligare strömtillförsel. Om systemets strömtillförsel tillgodoses av kortets system kan detta vara kontraproduktivt och leda till förhöjda temperaturer. Således kan en fläkt anses vara överflödig och kapslingen behöver inte dimensioneras därefter. Istället används öppningar i locket och raspberrien är upphöjd en 1 cm ovan botten för att skapa ett hållbart luftflöde. CNC-maskinen använder sig inte av vattenkylning vid arbeten och kapslingen behöver därav inte vara vattentät utan endast fungera som ett skydd från kontakt med spänningsförda delar och större partiklar. Kapslingen dimensioneras i två stycken, ett lock och huvudstycket som Raspberrien skruvas fast i. Huvudstycket behöver sju öppningar: en för strömtillförsel, HDMI-, USB- och SD-kortingång, två knäpphål för locket, samt en öppning för LPT-porten. Se följande tabell 6.1 för dimensioner för dessa öppningar.

Typ av öppning	Dimensioner i millimeter
HDMI	18x10
USB	12x8
SD-kort	18x6
Knäpphål för lock	2x6x3
LPT, ethernet och USB-portar	59x31

Tabell 6.1: Dimensioner för öppningar i kapsling

För att bestämma skalets höjd och bredd användes ungefärliga modeller av en modell b+, gertduino och en g25-sköld med dimensioner från de tekniska specifikationerna och skalets dimensioner anpassades därefter. det krävs dessutom ett utrymme mellan lock och g25-sköld för att ta hänsyn till de ledningstrådar som är anslutna på g25-sköldens ovansida. Skalets höjd med lock på är därför 84 mm, och bredden är 120 mm med samtliga komponenter i åtanke. För att skalet skall vara robust och inte flyttas av LPT-kabeln är tjockleken 3 mm vilket ger en bra tyngd och en låg tyngdpunkt.

7 | Resultat

Resultatet av arbetet uppfyller till stor del de utsatta målen. G-kodsändaren är liten, kompakt och har spänningsmatning via en 5V USB-kabel, samt kan anslutas till ett nätverk med dess Wifi-adapter eller via ethernet-kabel om det förstnämnda ej är tillgängligt. Därefter kan användaren med en smarttelefon, dator, tablet eller liknande ansluta till sändarens användargränssnitt genom att ansluta lokalt till <http://trcnc.local:8000/>, utan några krav på installationer eller förinstallerad mjukvara. Sändaren är alltså självständig och inte beroende av annan mjukvara.

I CNC.js gränssnitt är det enkelt att ladda upp en körfil och sedan följa processen i en WebGL-baserad 3d-miljö vilket ger en intuitiv och tillfredställande inblick i fräsprocessen. Innan en första fräsning utförs med den nya sändaren sker en kalibrering av fräs-maskinen enligt den beskriven tidigare i rapporten som sedan följs av en hemcykel. Efter kalibrering är samtliga värden för CNC-maskinen sparad i gertduinons processorminne så att kalibrering inte behöver återupprepas. Hemcykeln utförs genom att användaren ansluten till maskinen och gränssnittet trycker på *homing cycle* i gränssnittet varefter en cykel utförs. Därefter kan maskinen med ett knapptryck flyttas till arbetsnollpunkten och sedan utföra arbetet. Så länge stegningar av axlarna inte sker utan att GRBL är igång behövs en hemcykel inte köras inför nästa fräsning. Sändaren kan lämnas i gång mellan arbeten eftersom dess komponenter har så pass liten värmeutveckling.

Sändaren tar alltså mot filer och kommunicerar trådlöst med användaren och kan sedan utföra fräsarbeten med en enkel knapptryckning. Sändaren kan endast läsa filformat innehållande direkt G-kod (.gcode, .ngc och liknande) och kan alltså inte omvandla CAD-filer så som .dxf-filer. För detta krävs ytterligare CAM-mjukvara. I diskussionsdelen presenteras ett exempel på hur en sådan kan se ut om sändaren samtidigt skall behålla sin självständighet.

8 | Diskussion och slutsatser

8.1 Trovärdighetsanalys

Problemdefinitionen tillät relativt fria tyglar men var i kort mening: Utforma en lösning som minimerar handpåläggning vid fräsning av en plastkonsol. Systemet skulle dessutom vara anpassad för CNC-kontrollern Zero-3 och arbeta i symbios med denna. Ytterligare plus var om systemet kunde arbeta direkt från en CAD-fil i exempelvis .dxf-format från ett CAD-program och sköta hela processen, hela vägen fram till färdigt objekt.

Vid projektets start saknades kunskap om vad som faktiskt krävs av en mjukvara för att genomföra en sådan kompilering som normalt sköts av så kallade CAM-program. Dessa program behöver en mängd funktioner för att skapa lämplig g-kod och därav en djup förståelse av hur G-kod fungerar. Utöver detta tog hårdvarusteget av projektet långt längre tid än väntat, då till exempel funktioner så som säkerhetssignalen på 12.5 kHz till CNC-kontrollern så att denna fungerar, inte var känd från början och upptäcktes inte förrän när projektet skulle testköras. Därav valdes istället att fokusera på att göra en välfungerande hårdvarulösning som tillåter CNC-kontrollern att styras med GRBL och sedan föreslå en lösning på problemet i projektets diskussion.

8.2 Förbättringar

Projektet kan förbättras om tid finns på ett flertal olika vis. I följande sektion följer ett antal förslag på sådana förbättringar.

8.2.1 Hårdvara och miljöaspekter

Raspberry Pi

Hårdvarumässigt använder projektet en äldre modell av raspberry pi, modell b+. På senare tid har en mer modern version släppts: Raspberry Pi 3, som är utrustad med inbyggd WLAN-kompatibilitet. Därav krävs ingen wlan-adapter vilket innebär färre komponenter och dessutom ett mer effektivt, hållbart och miljövänligt system då wlan-adaptern är mer strömkrävande än den inbyggda WLAN-funktionen i Raspberry Pi 3. Utöver detta tillåter den kraftfullare processorn att javascript-applikationer körs bättre och således kan CNC.js drivas med mindre eftersläpning i mjukvaran och med en mindre risk för mjukvarukrasch.

Förbättrad anslutning mellan Arduino och CNC-kontroller

Vidare kan det tilläggas att G25-skölden är något av ett provisorium, då kortets printade krets till stor del inte används i projektet utan kringgås med externa ledningstrådar. Istället för att ta hjälp av av G25-skölden bör projektet utvecklas till att använda ett specialanpassat kretskort för ändamålet. Kortet kan göras i en designmjukvara för PCB så som *orCAD* eller Autocads *Eagle*. Detta tillåter projektets sändare att konstrueras i större mängd utan att vara beroende av att importera en produkt från ett utomstående företag för att sedan behöva omforma denna. Med andra ord blir projektet billigare, mer kompakt och dessutom mer hållbart ur en miljösynvinkel om ett färdigt PCB-kort som integrerar pulsgeneratorkretsen och CNC-skölden med dess kablar till ett tryckt kort från en PCB-leverantör.

Övriga hårdvaruförbättringar

I projektets fall är ledningstrådarna överdrivet långa vilket leder till större värmeförluster. Dessa kan göras kortare eller bortses från helt med tidigare nämnda förslag. En annan förbättring är att introducera en strömbrytare mellan raspberriens 5 V-anslutning och pulsgeneratoren så att denna inte konstant förses med spänning och således pulsar CNC-kontrollern även när denna inte är i bruk men raspberrien är i gång.

8.2.2 Mjukvara

Mjukvarumässigt kan systemet effektiviseras och utvecklas så länge tid finns. Just nu använder sig systemet av CNC.js som uppdateras relativt sällan. Programmet har ett tillfredsställande gränssnitt och många funktioner men är javascript-baserat och således krävande för ett svagt system som en raspberry. Att utforma ett eget program, exakt anpassad för projektets ändamål i en mer optimerad kodmiljö kan öka prestandan och förbättra användarvänligheten. Eftersom raspberrien kör ett eget operativsystem kan användaren självklart installera den mjukvara användaren anser vara tillfredsställande men i projektets fall och med den tid som tilläts för undersökning efter mjukvara ansågs CNC.js uppfylla villkoren bäst i enlighet med projektets syfte. Exempel på annan förbättrad mjukvara hade kunnat vara en sådan som tillåter flera fräsprojekt att fräsas ut under en och samma operation i följd genom att knyta samman de olika projektens block av g-kod då arbetsytan har utrymme för flera plastkonsoler. Alternativt finns även programmet bCNC.

Exempel på fortsatt förbättring av projektets mjukvara med bCNC

Från GitHub går Öpen Source-mjukvaran *bCNC*[20] av dess skapare *Vasilis Vlachoudis* att hämta hem. Mjukvaran vars kod helt är uppbyggd på Python, tillåter redigering av g-kod i en CAD-liknande miljö och kan öppna CAD-filer av bland annat filformatet *DXF*. Mjukvaran tillåter liksom CNC.js att skicka G-kod till GRBL, samtidigt som den kan redigera i samma gränssnitt. Vidare har Vasilis utvecklat en tilläggsfunktion som tillåter egenskriven Pythonkod att köras i programmets gränssnitt och ta vara på programmets funktioner utan att inverka på resten av programmets funktionalitet. Teleradio:s CAD-filer är standardiserade och en tilläggsfunktion kan således skapas, specialanpassad för denna

typ av CAD-fil så att användaren kan, bara med ett knapptryck utifrån CAD-filen, skapa en fil av kommandon redo att köras direkt från fräsmaskinens arbetsnollpunkt. Följande kodstycke är helt baserat på bCNC:s källfunktioner och körs på exempelvis den CAD-fil bifogad i bilagan:

```

1
2  def execute(self , app): #The execute function executes the code
3                             #within as a plugin.
4                             #The user can run the plugin by pressing a
5                             #button on the interface.
6
7  # Get selected blocks from editor
8  app.loadConfig()
9  blocks = app.editor.getSelectedBlocks()
10
11  if not blocks:
12      app.editor.selectAll()
13  app.insertCommand("MOVE CENTER", True)
14  app.insertCommand("ROTATE CW", True)
15  del app.gcode.blocks[0:2]
16  app.refresh()
17
18  app.insertCommand("OPTIMIZE", True)
19
20  app.profile(INSIDE)
21
22  app.selectInvert()
23  app.editor.deleteBlock()
24  app.editor.selectAll()
25
26  app.gcode.addBlockFromString("Header", app.gcode.header)
27  app.gcode.addBlockFromString("Footer", app.gcode.footer)
28
29  app.insertCommand("DOWN", True)
30  app.insertCommand("OPTIMIZE", True)
31
32  app.editor.selectAll()
33      app.refresh()
34  app.setStatus(_("Adjusted .dxf-file"))

```

Tilläggs-koden tar vara på de funktioner som finns att finna i det visuella gränssnittet av bCNC. Koden gör i följande order:

1. Markerar samtliga block av G-kod inladdad i bCNCs redigeringsvy.
2. Roterar samtliga block 90 grader åt klockans håll så att dessa passar överens med arbetsytan.
3. Centraliserar markerade block runt blockens gemensamma mittpunkt, alltså utgångspunkten för arbetet eller arbetsytans nollpunkt.
4. Markerar bara och sedan tar bort det andra blocket i listan av block, vilket är det block av G-kod som markerar fräsyntans gräns. Denna bana skall alltså inte fräsas.
5. PROFILE-funktionen skapar en inre offset-bana till de markerade blocken av G-kod. Banan har en offset på bormunstyckets diameter i millimeter dividerat med

- två. Detta är alltså så att hålen som fräses inte skall bli för stora, utan programmet kompenserar för de värden inmatade vid kalibrering som definierar bormunstyckets form.
6. Invertar valda objekt så att de yttre banorna markeras. Tar sedan bort sagda banor och markerar sedan resterande block.
 7. OPTIMIZE-funktionen räknar sedan ut den kortaste banan för CNC-maskinen att färdas mellan block.
 8. Koden bifogar sedan ett "header- och ett footer-block i toppen och botten av listan av block. Dessa block innehåller den kod som initialiserar bormaskinen innan fräsning, genom att exempelvis slå på borrets rotation. footer-blocket stänger av sagd rotation och kör tillbaka bormaskinen till arbetsnollpunkten.

Därefter är koden redo att köras och arbetet kan utifrån tidigare kalibrering köras. Koden kan utföras från bCNC:s HTML-gränssnitt med ett knapptryck, utan att behöva använda bCNC:s redigerare. Därefter kan koden utföras från samma gränssnitt vilket tillåter användaren att utföra arbeten med enkelhet.

8.2.3 Fördelar och nackdelar med bCNC kontra CNC.js

För nuvarande anses CNC.js vara den mjukvara som är mest lämpad för själva sändningen av kod i enlighet med projektets syfte, att vara användarvänlig och enkel för en oerfaren användare, samtidigt som den är stabil och tål en hög arbetslast. CNC.js tillåter dessutom en bättre trådlös funktion genom dess webbläsarbaserade gränssnitt och kräver inte att användaren är inloggad till raspberriens operativsystem via exempelvis fjärrskrivbordet. bCNC är en lättare mjukvara som tillåter viss hantering via en webbläsare, men HTML-gränssnittet är bristfälligt i funktionsmängd, och saknar den intuitiva 3d-miljö som CNC.js erbjuder. Trots det att programmet är pythonbaserat och kräver mindre prestanda i sig, visar sig CNC.js vara den mjukvara som kräver minst. För att bCNCs fullständiga funktioner så som kodredigeraren skall tillämpas måste användaren få tillgång till operativsystemets skrivbordsmiljö i raspberrien. Att ansluta till raspberrien med fjärrskrivbord innebär mycket låg prestanda i bCNCs gränssnitt och dessutom en tidsfördröjning tillförd av den trådlösa anslutningen. Ansluter användaren istället via en browser, behöver inte systemet förspilla en bit av prestandan på att rendera ett grafiskt skrivbord, samtidigt som den enhet som webbläsaren körs genom utför det tunga grafikarbetet i webbläsaren. En möjlig lösning är att köra bCNC:s HTML-gränssnitt i en flik i webbläsaren och från denna utföra det ovannämnda programtillägget och sedan spara koden i en mapp som är läsbar av CNC.js. På så vis behöver användaren inte ha tillgång till raspberriens skrivbordsmiljö för att utföra de funktioner som krävs själv och kan istället endast använda sig av webbläsaren.

Framtiden för projektet kan dock komma att ändra på hur mjukvaran är uppbyggd, vilket leder rapporten vidare till sin slutsats.

8.3 Slutsats

Framtiden för projektet är att utveckla en stabilare version, i en större mängd, närmare bestämt runt fem stycken. Företaget har bestämt sig för att inhandla ett flertal nya fräsmaskiner för dotterbolagen, och projektets sändare är både billigare att sätta ihop för företaget, och mer intuitiv i enlighet med projektets syfte. För att detta skall vara möjligt kommer en ny sköld för Gertduinon att utvecklas, som implementerar pulsgeneratorkretsen och de kablar monterade på skölden beskriven i rapporten in i PCB-form. Detta kommer att innebära bättre isolation och mindre plats, samt en mer färdig applikation. Vidare kommer operativsystemet och mjukvaran använda en senare version av Raspberry Pi, troligtvis en modell 3. Projektet kan i nuvarande tillstånd inte utföra hela den väg av en fräsning som projektets ambition var att fullfölja, dock har en god hårdvaruplattform tagits fram, som är mycket modulär och baserar på GRBL, som är en populär sändarmjukvara under ständig utveckling. Detta innebär att nya mjukvaror utvecklas kontinuerligt för sagd mjukvara, ofta via populära plattformar som GitHub vilket i sig innebär att projektets plattform kan uppdateras med tiden och så småningom vara så pass komplett att användaren med bara ett knapptryck kan gå från CAD-fil till färdigfräst konsol.

Referenser

- [1] Arduino LLC. *Arduino/Genuino UNO*. 2017. URL: <https://www.arduino.cc/en/Main/arduinoBoardUno> (hämtad 2017-01-24).
- [2] Autodesk. *Autodesk 123D Design*. URL: <http://www.123dapp.com/design> (hämtad 2017-01-30).
- [3] CharlesGantt. *Getting up and running with the Gertduino | element14 | GertDuino*. URL: <https://www.element14.com/community/community/raspberry-pi/raspberry-pi-accessories/gertduino/blog/2014/01/14/getting-up-and-running-with-the-gertduino> (hämtad 2017-01-23).
- [4] Cheton Wu. *cncjs by cheton*. 2017. URL: <https://cheton.github.io/cnc.js/> (hämtad 2017-01-23).
- [5] CNC-step. *CNC Router Machine | For 2D and 3D working on Wood Aluminium Stone*. URL: <https://www.cnc-step.de/en/cnc-machines/cnc-router-machine/> (hämtad 2017-01-23).
- [6] CNC-step. "CNC-STEP Zero 3 5-channel stepper motor controller User Manual". I: (2012). URL: www.cnc-step.de.
- [7] CNC-step. "High-Z Standard-Series Short description". I: (2014).
- [8] Element14. *GertDuino | element14*. URL: <https://www.element14.com/community/community/raspberry-pi/raspberry-pi-accessories/gertduino> (hämtad 2017-01-23).
- [9] Eonkits. *Pulsgenerator (555) @ Electrokit*. 2005. URL: <https://www.electrokit.com/pulsgenerator-555.46152> (hämtad 2017-01-24).
- [10] Friends of The Unicorn. *Friends of The Unicorn*. URL: <http://friendsoftheunicorn.net/> (hämtad 2017-02-01).
- [11] G.J van Loo. "GertDuino Board User Manual". I: (2013), s. 1–26. URL: <https://www.element14.com/community/servlet/JiveServlet/downloadBody/64534-102-2-287165/User%20manual%20Gerduino%205.6.pdf>.
- [12] Gordon @ Drogon. *ATmega Setup | Gordons Projects*. URL: <https://projects.drogon.net/raspberry-pi/gertduino/atmega-setup/> (hämtad 2017-02-01).
- [13] Gordon @ Drogon. *Gordon Hendersons Projects pages*. URL: <https://projects.drogon.net/> (hämtad 2017-02-01).

- [14] KC Linear. *KC Linear - Linear Ball Bearings, Mounts, Shafts for CNC Routers*. URL: <http://kcllinear.com/> (hämtad 2017-01-24).
- [15] MrEngman. (UPDATE) *Drivers for TL-WN725N V2 - 3.6.11+ -> 4.xx.xx+*. URL: <https://www.raspberrypi.org/forums/viewtopic.php?t=62371> (hämtad 2017-01-23).
- [16] Raspberry Pi Foundation. *Raspberry Pi Hardware - Raspberry Pi Documentation*. URL: <https://www.raspberrypi.org/documentation/hardware/raspberrypi/> (hämtad 2017-01-23).
- [17] Ron Light. *Arduino GRBL to DB25 CNC Shield Kit from Ron on Tindie*. 2017. URL: <https://www.tindie.com/products/Ron/arduino-grbl-to-db25-cnc-shield-kit/> (hämtad 2017-01-24).
- [18] Sungeun K. Jeon. *GRBL(tm)*. URL: <https://github.com/gnea/grbl>.
- [19] Wandrson. *A small enclosure for a Raspberry P B+/2 and the Pi TNC board by wandrson - Thingiverse*. URL: <https://www.thingiverse.com/thing:763710> (hämtad 2017-01-31).
- [20] Vlachoudis Vasilis. *Vlachoudis bCNC*. URL: <https://github.com/vlachoudis/bCNC> (hämtad 2017-02-01).

9 | Bilaga A: Företagets typiska CAD-fil i DXF-format

