TSEA56 - Kandidatprojekt i elektronik
LIPS Designspecifikation

Version 1.0

Grupp 4

Hynén Ulfsjöö, Olle	ollul666
Wasteson, Emil	emiwa068
Tronje, Elena	eletr654
Gustafsson, Lovisa	lovgu777
Inge, Zimon	zimin415
Strömberg, Isak	isast763

10 maj 2016

Status

Granskad	Olle Hynén Ulfsjöö	2016-03-21
Godkänd	Peter Johansson	2016-03-22

PROJEKTIDENTITET

2016/VT, Undsättningsrobot Gr. 4

Linköpings tekniska högskola, ISY

Namn	**Ansvar**	**Telefon**	**E-post**
Isak Strömberg (IS)	Projektledare	073-980 38 50	isast763@student.liu.se
Olle Hynén Ulfsjöö (OHU)	Dokumentansvarig	070-072 91 84	ollul666@student.liu.se
Emil Wasteson (EW)	Hårdvaruansvarig	076-836 61 66	emiwa068@student.liu.se
Elena Tronje (ET)	Mjukvaruansvarig	072-276 92 93	eletr654@student.liu.se
Zimon Inge (ZI)	Testansvarig	070-171 35 18	zimin415@student.liu.se
Lovisa Gustafsson (LG)	Leveransansvarig	070-210 32 53	lovgu777@student.liu.se

E-postlista för hela gruppen: isast763@student.liu.se

Kund: ISY, Linköpings universitet tel: 013-28 10 00, fax: 013-13 92 82
Kontaktperson hos kund: Mattias Krysander
tel: 013-28 21 98, e-post: matkr@isy.liu.se

Kursansvarig: Tomas Svensson
tel: 013-28 13 68, e-post: tomass@isy.liu.se
Handledare: Peter Johansson
tel: 013-28 13 45, e-post: peter.a.johansson@liu.se

Innehåll

Dokumenthistorik

Version	**Datum**	**Utförda förändringar**	**Utförda av**	**Granskad**
1.0	2016-03-23	Tredje utkastet	Grupp 4	OHU
0.2	2016-03-21	Andra utkastet	Grupp 4	OHU
0.1	2016-03-07	Första utkastet	Grupp 4	IS

1 Inledning

Det här dokumentet innehåller en beskrivning av en undsättningsrobot, dess komponenter och hur kommunikationen mellan de olika modulerna sker. Syftet är att ge en detaljerad beskrivning som underlag för konstruktionsarbetet.

1.1 Det totala systemet

Det totala system kommer att bestå av tre processormoduler, en kommunikationsenhet för Bluetooth®, ett chassi med inbyggda motorer, en gripklo, ett styrservo och ett antal olika sensorer. De olika modulerna följer:

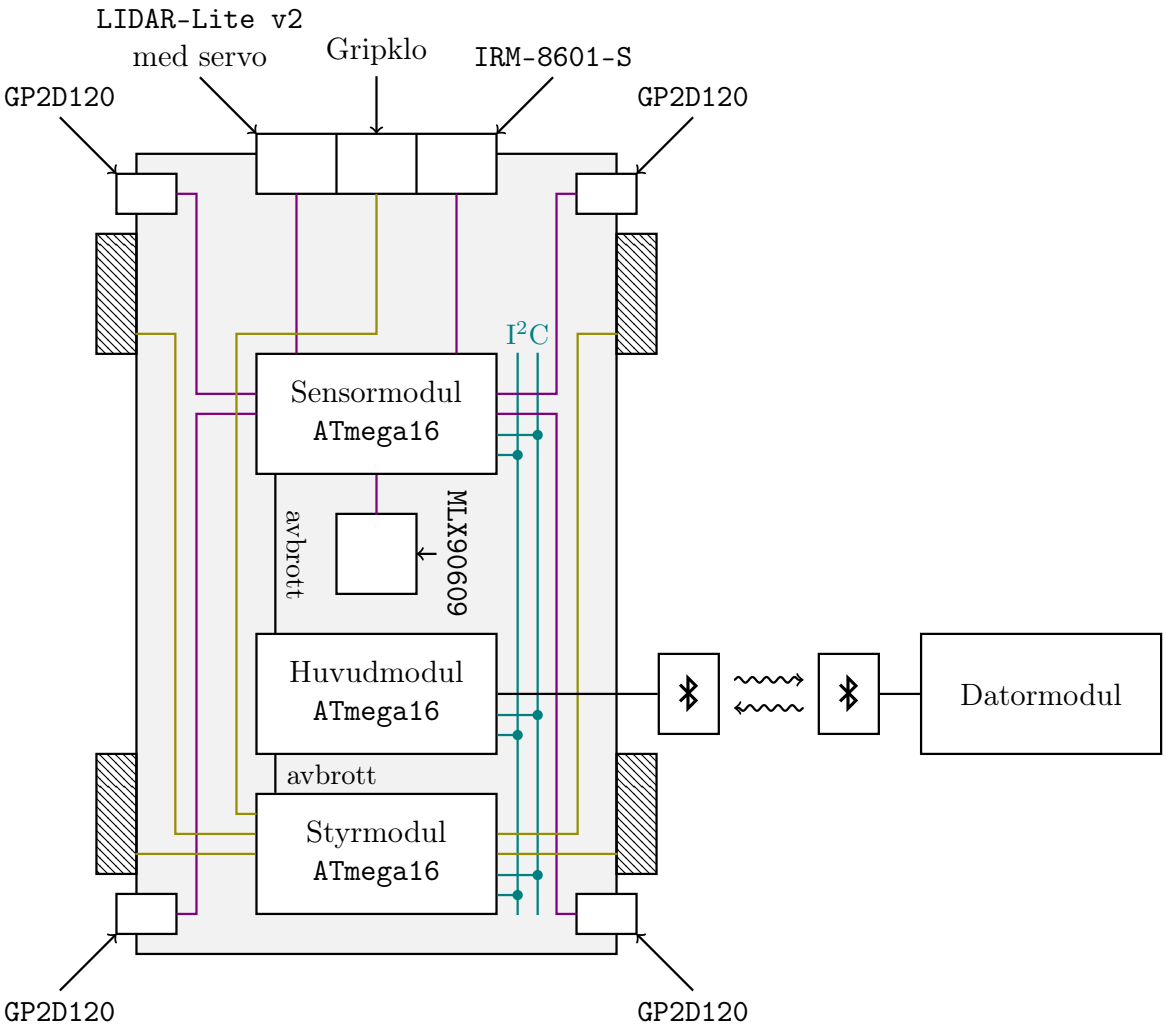
- Huvudmodul - sköter kommunikationen, både internt via I^2C-bussen och externt via en Bluetooth®-enhet.
- Sensormodul - är direkt kopplad till alla sensorer, tar emot och konverterar (om nödvändigt) all data från dessa. Den färdigkonverterade datan skickas sedan vidare till huvudmodulen.
- Styrmodul - är sammankopplad med gripklon, ett servo för rotation av en sensorer och chassits motorer. Den tar emot instruktioner och data för reglering från huvudmodulen.

Förutom förbindelserna med olika externa enheter är de tre modulerna sammankopplade med en I^2C-buss där huvudmodulen är master och kontrollerar kommunikationen.

Undsättningsroboten kommer att regleras med hjälp av PD-reglering. Nödvändiga konstanter för att kunna utföra denna typ reglering kommuniceras från datormodulen till huvudmodulen via Bluetooth®. Via I^2C-bussen vidarebefordrar huvudmodulen därefter konstanterna till styrmodulen där PD-reglering utförs.

Utöver detta så kommer även lysdioder att kopplas på kretskorten, detta för att underlätta felsökning. En utsignal kan då kopplas till en lysdiod för att säkerställa att ett värde generas till utporten.

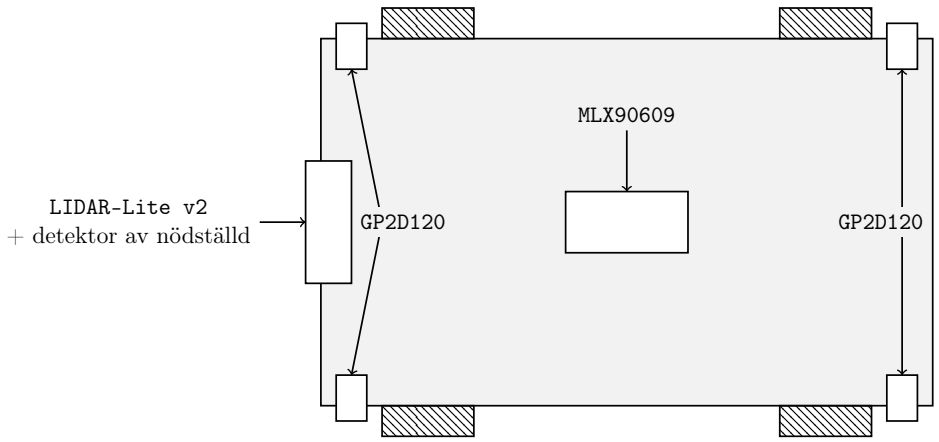
En övergripande bild över systemet återfinns i figur **??**.



Figur 1: Det totala systemet

1.2 Sensorer

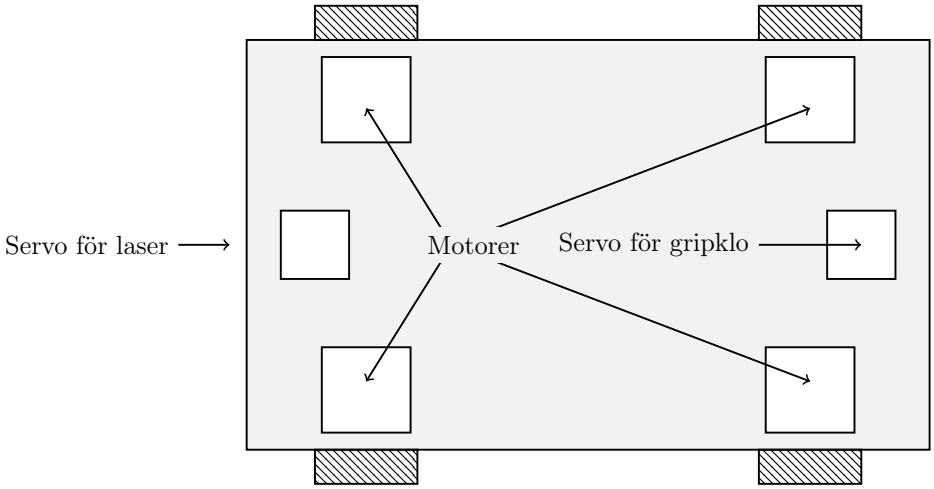
Roboten kommer vara utrustad med sex stycken sensorer. Fyra infraröda sensorer av typen `GP2D120` används för att bestämma avstånden till väggarna på robotens sidor och robotens vinkel i förhållande till dessa. Vidare finns en vinkelhastighetsgivare, `MLX90609` som används för att reglera rotation av roboten. Slutligen finns en lasersensor av typen `Lidar-Lite v2` för att bestämma avståndet till väggar framför roboten. Den sistnämnda är placerad på ett servo för att kunna roteras. Placeringen av sensorerna visas nedan i figur **??**.



Figur 2: Placering av sensorer

1.3 Ställdon

Roboten kommer använda sig av totalt sex olika ställdon. Fyra av dessa är motorerna som driver varsitt hjul. Motorerna kan dock inte styras individuellt utan enbart parvis, varför de egentligen kan ses som två enheter. Utöver dessa ska roboten ha ett servo i gripklon, för att kunna öppna och stänga densamma, och ett servo för rotation av lasersensorn. Ställdonens placering visas nedan i figur **??**.

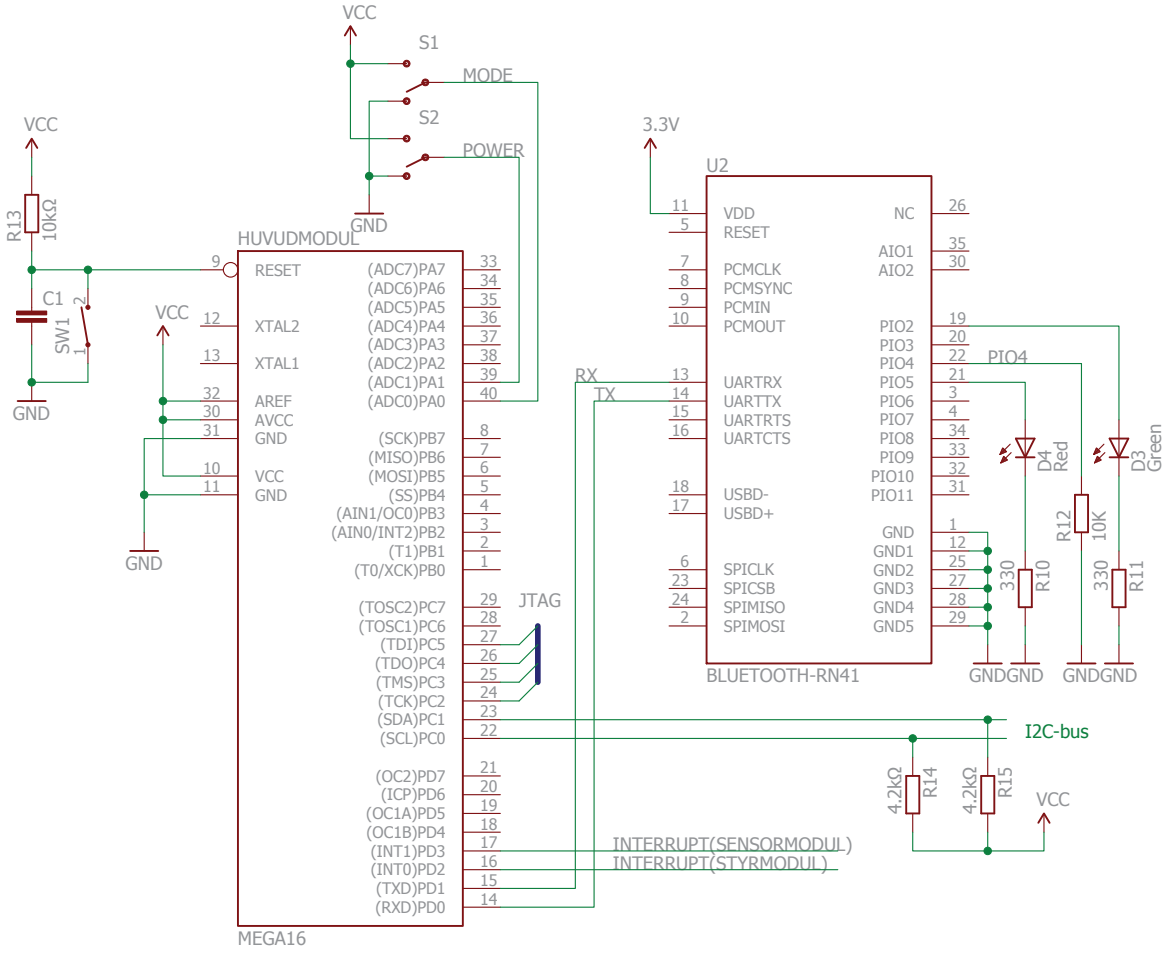


Figur 3: Placering av ställdon

2 Delmodul 1 - Huvudmodul

Robotens huvudmodul har som uppgift att kommunicera med dels en extern datormodul och dels med robotens övriga moduler. Kommunikationen med de övriga modulerna sker genom en tvåtrådsbuss, I^2C, och huvudmodulen fungerar där som överordnad de andra processorenheterna. Kommunikationen med datormodulen sker via Bluetooth®. Mer om intermodulär kommunikation går att läsa i avsnitt **??** Förutom detta ska huvudmodulen även sköta den interna kartläggningen.

Kopplingsschema för huvudmodulen ser ut enligt figur **??**. Till avbrottsingång `INT0` är styrmodulen kopplad och till `INT1` sensormodulen. Dessa avbrott används då styr- eller sensormodulen vill initiera kommunikation med huvudmodulen. Som det framgår av figuren räcker `ATmega16` för det modulen behöver. Det finns möjlighet att koppla in ytterligare funktion då flera utgångar ej används.



Figur 4: Kopplingsschema över huvudmodul

2.1 Detaljerad beskrivning

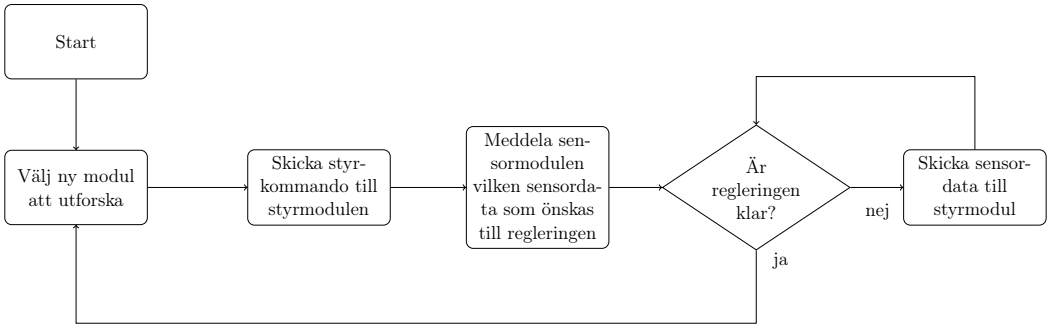
Nedan följer en mer ingående beskrivning av robotens olika körlägen, avbrottshantering och kartläggning.

2.1.1 Körläge

En brytare för val av körläge kopplas till huvudmodulen på en dataingång. Ingången läses en gång per varv i huvudloopen.

Autonomt läge

Kommunikationsflödet i autonomt läge finns beskrivet i figur **??**. Beslutet av vilken ny kartmodul, en ruta på ca 40x40 cm, som roboten ska avsöka tas utifrån den avsökningsalgoritm som implementeras. Oberoende av vilken kartmodul som väljs ser det efterföljande flödet likadant ut. I frågan om när regleringen är klar avvaktas ett avbrott från styrmodulen, vilket beskrivs närmare i avsnitt **??**.



Figur 5: Flödesschema som beskriver förloppet vid autonom styrning

Manuellt läge

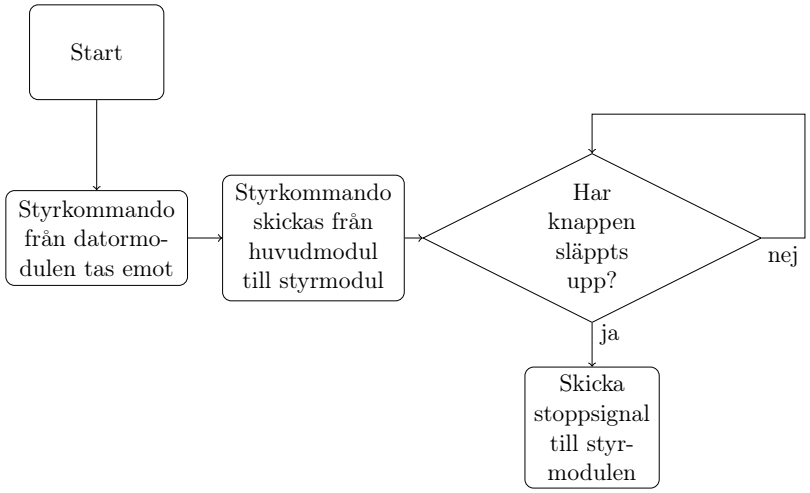
I figur **??** syns kommunikationsflödet när roboten är inställd på manuell styrning. Så fort datormodulen upptäcker att ett styrkommando matas in skickas det vidare till huvudmodulen. När det aktuella styrkommandot ändras meddelas huvudmodulen, som i sin tur meddelar styrmodulen.

2.1.2 Strömbrytare

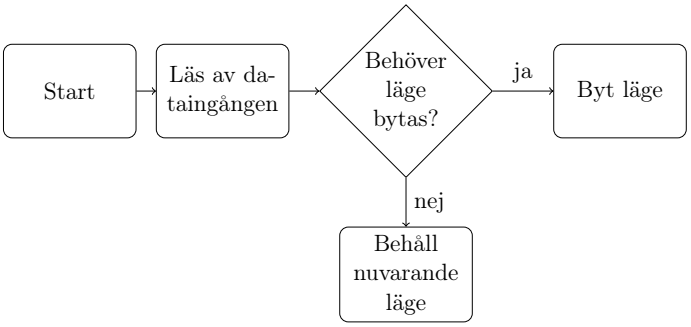
Strömbrytare kopplas till en dataingång. Detta medför att förändring inte nödvändigtvis sker på en gång. I figur **??** visas förloppet då huvudprogrammet är redo att läsa av ingången.

2.1.3 Sensormodul

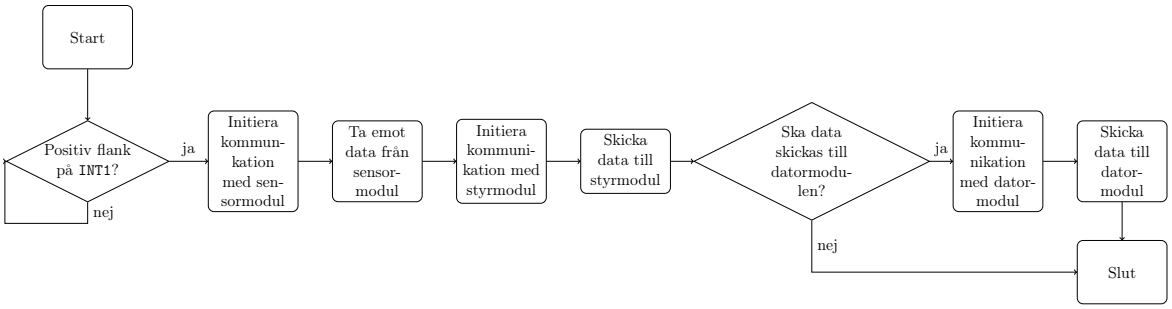
Sensormodulen är kopplad till huvudmodulens avbrottsingång `INT1`. Händelseförloppet vid aktiverat avbrott kan ses i flödesschemat i figur **??**.



Figur 6: Flödesschema som beskriver förloppet vid manuell styrning



Figur 7: Flödesschema som beskriver förloppet vid då strömbrytaren aktiveras



Figur 8: Flödesschema som beskriver förloppet vid avbrott på ingång `INT1`

2.1.4 Styrmodul

Styrmodulen är kopplad till huvudmodulens avbrottsingång `INT0`. Ett avbrott initieras vid positiv flank. Avbrottssignalen kommer skickas (från styrmodulen) när regleringen av nuvarande styrkommando är klart. Huvudmodulen kan då ta ett nytt beslut enligt flödesschemat i figur **??**.

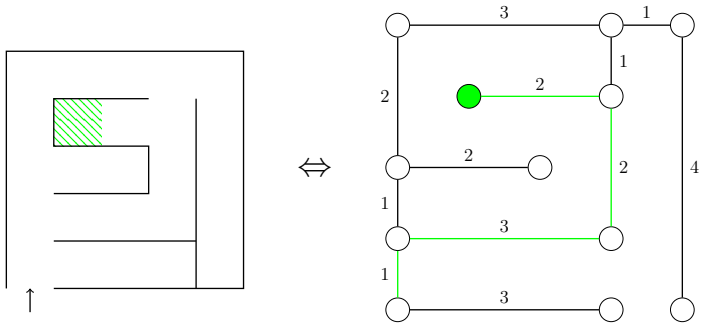
2.1.5 Kartläggning

Eftersom kartan enligt appendix C består av moduler på 40x40cm, räcker det att uppdatera kartläggningen en gång per modul. Genom mätvärden från lasersensor, då den riktas framåt, kan avståndet till den främre väggen beräknas. Denna signal samplas med sådan frekvens att roboten kan beräkna när den har färdats till nästa modul.

IR-sensorerna på robotens sida kan mäta i avståndsområdet 5-30 cm och bottnas med andra ord när en sväng eller korsning upptäcks. Utifrån denna information kopplar kartläggningsalgoritmen ihop föregående modul med den typ av korsning som roboten åkt in i.

Kartläggningsalgoritmen är av väggföljartyp. Eftersom banan är uppbyggd så att varken höger- eller vänsterföljning prioriteras, se appendix C, så faller valet på högerföljning. Med andra ord följer roboten höger vägg och försöker först att svänga höger, därefter köra rakt fram och svänger vänster allra sist. Eftersom roboten endast ska finna den kortaste vägen från ingången till nödställde behöver nödvändigtvis inte hela kartan utforskas. Exempelvis, ifall fågelvägen från roboten till ingången är längre än en redan utforskad rutt ska roboten inte fortsätta utforska den korridoren. Vidare behöver roboten inte utforska en återvändsgränd ifall den främre sensorn indikerar att det endast är en modul framöver, förutsatt att målet inte har upptäckts i modulen.

Mjukvarumässigt representeras kartan med hjälp av noder och bågar, se figur **??** för ett exempel av labyrintkonvertering. En ny nod bildas vid varje korsning eller där roboten tvingas svänga. Bågkostnaden utgör det relativa avståndet mellan två sammankopplade noderna och relativa avståndet mellan två moduler är en längdenhet.



Figur 9: En labyrint och dess motsvarande nätverk. Bågkostnaderna är det relativa avståndet och den optimala vägen är markerad grön.

När kartan är representerad som ett nätverk och roboten med säkerhet har utforskat den kortaste vägen mellan ingång och nödställd, påbörjas optimeringen. Optimeringen för att hitta den kortaste vägen sker med hjälp av optimerande metoder för att lösa minkostnadsnätverk.

Roboten tillämpar optimeringsmodellen A* vilken är en vidareutveckling av Dijkstras algoritm. A* är en bäst-först algoritm med målet att finna den billigaste vägen mellan två noder. I grund och botten finns samma maskineri som i Dijkstras algoritm med skillnaden att noder avsöks i stigande ordning av

$$f(n) = g(n) + h(n)$$

där $g(n)$ utgör kostnaden av rutten från startnod till nod n och $h(n)$ en heuristik som estimerar den billigaste vägen från nod n till slutnod. Med andra ord är Dijkstras algoritm ett speciallfall av A* där $h(n) = 0$. Heuristiken som används är det så kallade *Manhattan*-avståndet eftersom labyrintens korsningar är ortogonala.

Låt V vara mängden av samtliga noder, S mängden av avsökta noder och Q mängden av oavsökta noder. Då implementeras algoritmens utifrån pseudokoden i algoritm **??**.

Algorithm 1 A*

```
1:  function AStarAlgorithm(Graph, s)
2:      g(s) ← 0
3:      f(s) ← g(s) + h(s)
4:      for all v ∈ V − {s} do
5:          g(v) ← ∞
6:          f(v) ← ∞
7:      S ← ∅
8:      Q ← V
9:      while Q ≠ ∅ do
10:         u ← minFvalue(Q, f)                ▷ Välj nod med lägst f(n)
11:         S ← S + {u}
12:         Q ← Q − {u}
13:         for all v ∈ neighbours(u) do       ▷ Itererare över grannar till u
14:             if g(u) + w(u, v) < g(v) then
15:                 g(v) ← g(u) + w(u, v)
16:                 f(v) ← g(v) + h(v)
        return g
```

2.2 Hårdvara

Huvudmodulen kommer kräva följande hårdvara:

- `ATmega16`
- Bluetooth® (FireFly)

2.3 Mjukvara

Koden i huvudmodulen är skriven i C. Nedan beskrivs vad som händer i huvudprogrammet respektive i avbrott.

2.3.1 Huvudprogram

I huvudmodulens huvudloop ska följande saker genomföras:

- Läsa av körläge (autonomt eller manuellt)
- Läsa av strömbrytaren (aktiv eller i viloläge)
- I autonomnt körläge, hantera avsökningen och skicka tillhörande körkommando

2.3.2 Avbrott

De avbrott som är kopplade till huvudmodulen ska hantera följande:

- Hämta och skicka sensordata
- I autonomnt körläge trigga beslut kring vilken kartmodul som ska avsökas
- I manuellt körläge ta emot körkommando via Bluetooth®

3 Delmodul 2 - Sensormodul

Sensormodulens uppdrag är att kommunicera med både sensorerna och huvudmodulen. Mätdata samplas, omvandlas och konverteras (vid behov) med konstant frekvens. För avståndssensorerna innebär detta att den analoga inspänningen omvandlas till ett digitalt värde som sedan konverteras till enheten centimeter.

Den främre avståndssensorn (laser-sensorn) placeras på ett roterande servo så att den kan *scanna* korridoren roboten befinner sig i. Detta förutsätter att servot går att styra så att vinkelutslaget går att beräkna med godtycklig precision. Tillåter inte servot en sådan styrning placeras laser-sensorn så att den konstant pekar rakt framåt.

Vinkelhastighets-sensorerna används för att beräkna vinkelutslag då roboten tar en sväng. Mätdatan behöver därför integreras under ett tidsintervall och konverteras till en vinkel. Hur den nödställde ska representeras är ännu inte fastställt. Förslagsvis används en IR-fyr i kombination med en IR-sensor. Sensormodulens uppdrag blir då att ta beslutet om när den nödställde är funnen.

3.1 Hårdvara

Den hårdvara som sensormodulen kommer att kräva är:

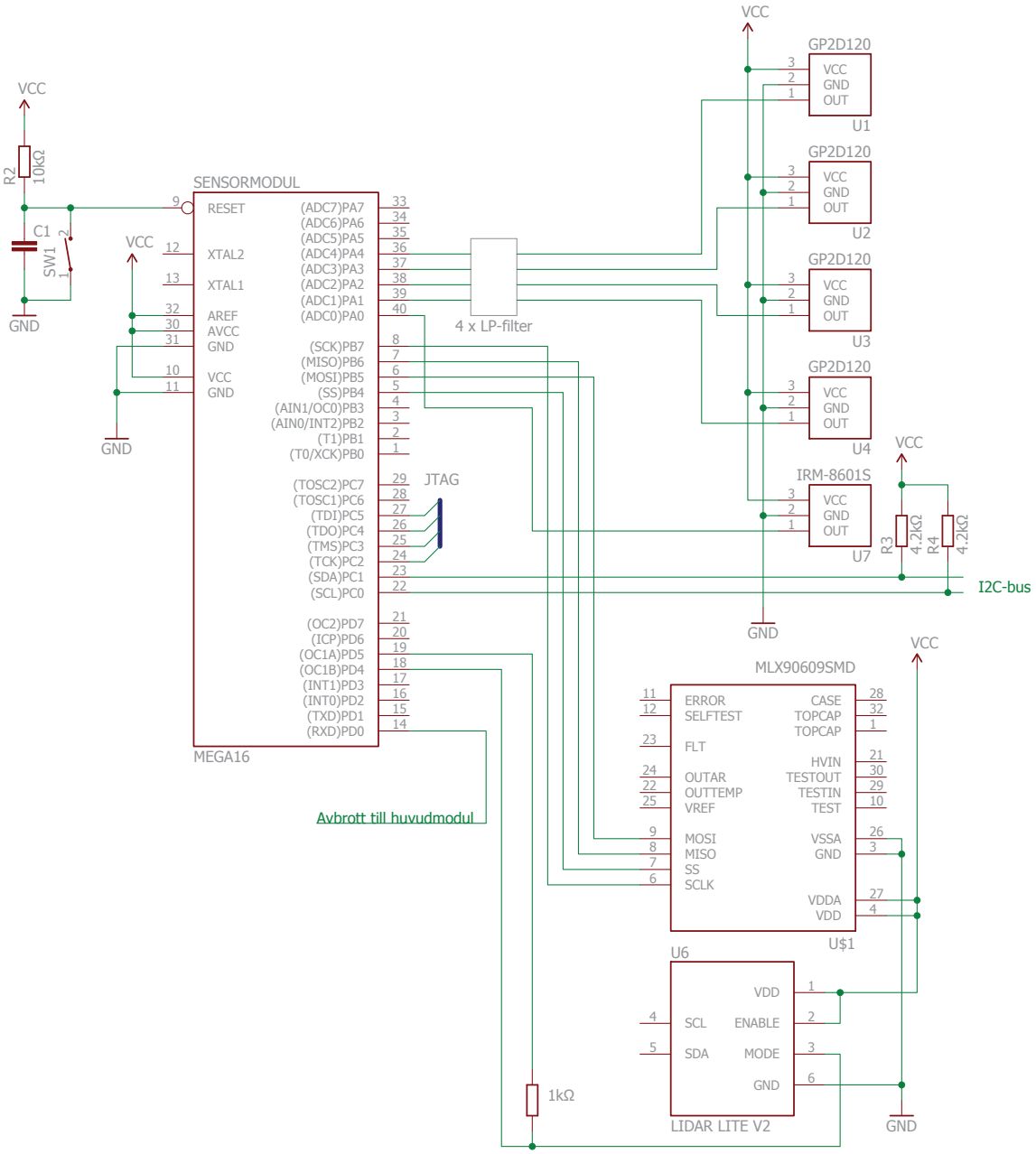
- Processorenhet, `ATmega16` x 1
- IR-sensor, `GP2D120` x 4
- Laser-sensor, `LIDAR-Lite v2` x1
- Vinkelhastighetssensor, `MLX90609` x 1
- IR-detektor, `IRM-8601-S` x 1

3.2 Detaljerad beskrivning

Sensormodulens processorenhet, `ATmega16`, är ansluten med en I^2C-buss. Den har som uppgift att kommunicera med huvudmodulen, där sensormodulens processorenheter agerar som slavenhet i förhållande till huvudmodulens dito. IR-sensorn `GP2D120` och IR-detektorn `IRM-8601-S` ansluts direkt till processorenhetens ingångar för AD-omvandling. Figur **??** visar en övergripande bild över sensormodulens implementation med dess respektive delkomponenter.

3.2.1 Processorenhet

Insamlat sensordata bearbetas i modulens processorenhet för att därefter kommuniceras till huvudmodulen. Figur **??** illustrerar `ATmega16` och dess ingångar, samt hur respektive sensorenhet ska kopplas samman med processorn.



Figur 10: Kopplingsschema över sensormodul

3.2.2 Laser-sensor

`Lidar-Lite v2` är en kraftfull lasersensor av begränsad storlek med ett flertal funktioner ändamålsenliga för exempelvis en undsättningsrobot. Utöver att kunna mäta avstånd, så kan den dessutom mäta hastighet och signalstyrka hos mål på upptill 40 meters avstånd. Detta med en felmarginal på 2,5 cm. Andra fördelar hos sensorn inkluderar låg strömkonsumtion och en upplösning 1 cm [**?**].

3.2.3 IR-sensor

För att kunna reglera på ett önskat sätt behöver roboten veta hur den förhåller sig i förhållande till sidoväggarna. Eftersom roboten ska kunna köra i mitten av korridorer med cirka 40 centimeters bredd och roboten själv är ca 20 cm bred, kommer ett avstånd mellan sidoväggarna och roboten på 10 cm vara önskvärt. För detta är IR-sensorn `GP2D120` bra, eftersom den kan mäta avstånd till objekt som befinner sig på 4-30 cm från sensorn.

Mätandet av avstånd sker genom att IR-ljus med en viss våglängd skickas ut från sensorn, reflekteras mot väggarna för att sedan detekteras av sensorn. Genom att mäta den vinkel reflektionen sker med kan avståndet till väggen bedömas. `GP2D120` är relativt enkel och har enbart två ingångar som ska kopplas till en 5 volts strömkälla, respektive en jord. Utsignalen är analog i form av spänning och på grund av mängden högfrekvent brus i signalen måste den lågpassfiltreras.

3.2.4 Gyro/accelerometer

I den labyrint där roboten ska hitta den nödställde kommer det finnas korsningar, vilket ställer krav på roboten att svänga. För att klara av detta används `MLX90609`. Genom att mäta hur mycket roboten roterar kan den avgöra när en 90-graderssväng är genomförd och det är dags att sluta rotera. Hastigheten på rotationen kan ställas in efter önskat behov.

3.2.5 IR-detektor

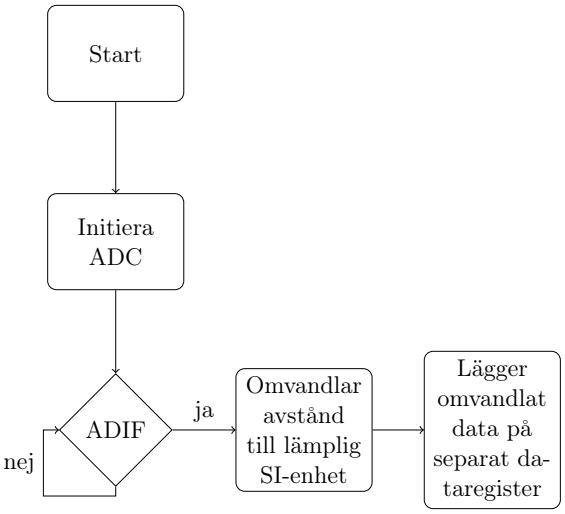
IR-detektorn `IRM-8601-S` används i syfte att identifiera den nödställde, vilken ska sända ut en IR-signal. Liksom figur **??** illustrerar så är `IRM-8601-S` förhållandevis trivial i sitt utförande, enheten har 2 ingångar vilka ska anslutas med 5 V samt jord.

3.3 Mjukvara

Sensormodulens kod är skriven i C och har som huvuduppgift att göra om de analoga signaler som uppmäts till digitala signaler som skickas visare till huvudmodulen. De olika sensorerna tar in olika typer av analoga signaler, vilket innebär att även fler typer av omvandlingar behöver vara möjliga. Alla omvandlingar sker i enhetens processorenhet, `ATmega16`.

3.3.1 IR-sensor

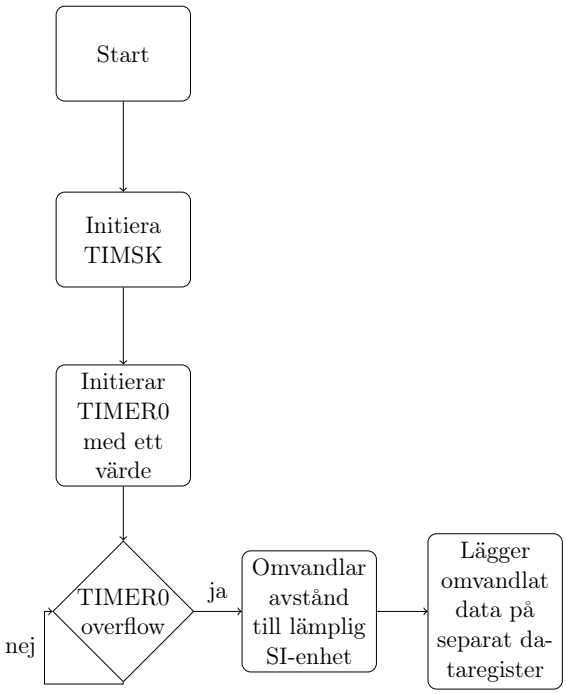
Initiering av AD-ingångarnas status- och kontrollregister (`ADCSRA`) sker i processorns startrutin. Processorenheten kommer därefter att fortsätta med dess respektive instruktioner tills dess att något sensorvärde är färdigomvandlat, varpå ett avbrott kommer att kallas. Figur **??** visar ett flödesschema vilket beskriver hur hanteringen av sensorvärden ska fungera.



Figur 11: Flödesschema som illustrerar hur AD-omvandlade sensorvärden ska behandlas

3.3.2 Laser-sensor

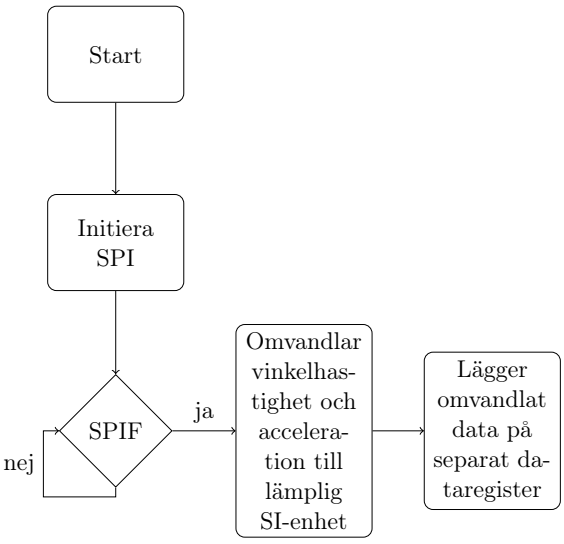
Då lasersensorn `LIDAR-lite v2` kommer att kopplas samman med processorenheten via ett PWM-interface så kommer avbrott, vilket kallas med hjälp av en timer (`TCNT0`), att behövas. För att göra detta så krävs det att `TIMSK`-registret (Timer/CounterInterrupt Mask Register) initeras. `TIMER0`-registret kommer att användas för att programmera timern, det är ett 8-bitars register vilket innebär att timern kan räkna upp 255 innan den nollställs och avbrott (`TIMER0` overflow) anropas. Figur **??** ger en översiktlig bild av hur PWM-interfacet är tänkt fungera.



Figur 12: Flödesschema som en översiktlig bild av hur sensorvärden från lasersensorn ska mottagas samt processeras

3.3.3 Vinkelhastighetssensor

Eftersom informationen från `MLX90609` kommer att överföras via ett SPI-interface så är initiering av `SPCR` (SPI Control Register) nödvändigt. Detta ska ske när processorn startas. Övriga instruktioner kommer därefter att utföras tills dess att avbrottsflaggan tillhörande SPI:ns statusregister 1-ställs, varpå stackpekaren kommer att gå in i avbrottet för att behandla sensorvärdena från sensorn. Figur **??** visar ett flödesschema vilket beskriver hur SPI-interfacet ska fungera samt hur sensorvärdena hanteras när dess avbrottsflagga ettställs.



Figur 13: Flödesschema som illustrerar hur sensorvärden från gyrot ska behandlas

3.3.4 Kodstruktur

Sensormodulen har som nämnt tidigare tre huvuduppgifter: ta emot rådata från sensorerna (både i form av analoga och digitala signaler), A/D-omvandla och konvertera dessa signaler (om det krävs) och sedan skicka paket av relevant, konverterad sensordata till huvudmodulen.

3.3.5 Mottagande av data och omvandling

Sensordatat kommer tas emot i tre olika format: analoga signaler från IR-sensorerna, PWM-signal från lasersensorn och SPI-signal från gyrot. De analoga signalerna A/D-omvandlas kontinuerligt. När A/D-omvandlingen är klar initieras en avbrottsrutin som sparar det icke bearbetade digitala värdet i en separat variabel för varje sensor.

3.3.6 Konvertering

Konvertering av de sensorvärden som inte är linjära sker kontinuerligt i huvudloopen. Programmet hämtar ett värde i taget från variablerna för okonverterade sensordata, konverterar sedan dessa med hjälp av en tabell eller funktion och sparar det nya värdet i en ny variabel.

3.3.7 Skicka sensordatapaket

De färdigkonverterade datapaketen från sensormodulen kommer skickas med jämna mellanrum, med en frekvens på ca 10-20 Hz. Det passar därför bra att initiera subrutinen, som har hand om överföringen, med ett avbrott kopplat till en klocka med lämplig frekvens. Detta avbrott har högst prioritet hos sensormodulen. Subrutinen skickar sedan en positiv

flank till huvudmodulen, varpå ett avbrott initieras hos huvudmodulen och den upprättar kontakt via I^2C. När hela datapaketet sedan överförts är denna avbrottsrutin avklarad.

4 Delmodul 3 - Styrmodul

Styrmodulens uppgift är att översätta styrkommandon till servostyrning av de fyra hjulen och gripklon. Styrkommandona kommer via I^2C-bussen från huvudmodulen och konverteras därefter till en PWM-sekvens. PWM-sekvensen skickas till gripklon respektive H-bryggorna som styr ett hjulpar var.

Kopplingsschemat ser ut enligt figur **??**, det framgår från figuren att utgångarna från `ATmega16` räcker för modulens ändamål. Både avbrottsingång `INT0` och `INT1` används inte men reservas ifall ytterligare funktionalitet önskas lägga till. Det finns även möjlighet att att styra en extra komponent via PWM eftersom utgången `OC2` är oanvänd.

Prestandan hos `ATmega`16 räcker för modulens syfte.

4.1 Detaljerad beskrivning

Nedan följer en beskrivning av vilka styrkommandon styrmodulen ska kunna ta emot, hur kommunikationsprotokollet ser ut, hur regleringen ska ske samt övrig fakta kring modulen.

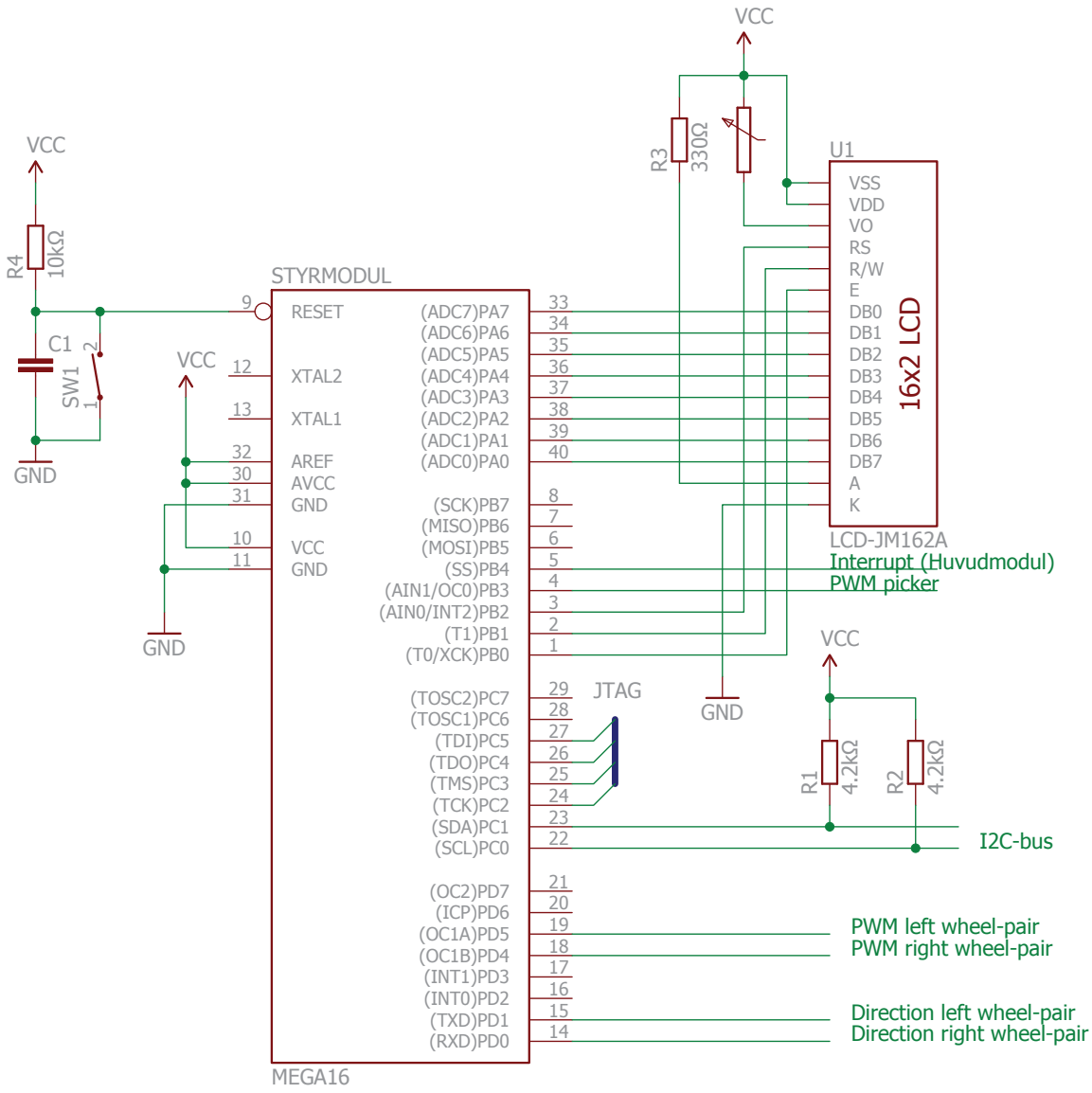
4.1.1 Styrkommandon

Styrmodulen tar emot kommandon från huvudmodulen och verkställer dessa. Följande kommandon kommer kunna skickas: *reglering av/på, gripklo öppna/stäng, fram/bak, sväng höger/vänster, rotera höger/vänster och stopp*. Kommandot *reglering av/på* ställer in styrmodulen för manuell eller autonom körning, vilket påverkar tolkningen av det data som skickas från huvudmodulen.

Tillsammans med kommandona för rörelse skickas även ett tal som, beroende på om reglering är aktiverad eller inte, anger en sträcka/vinkel för förflyttningen eller med vilken hastighet (i procent av maxhastighet) förflyttningen ska utföras. Ett kommando kan exempelvis se ut såhär: *fram 80*, varpå roboten antingen kör framåt 80 cm eller med 80% av maxhastighet, beroende på om regleringen är aktiverad eller inte. Modulen kommer vara designad så att ett körkommando gäller tills dess att ett nytt kommando skickas, varför även kommandot för *stopp* är nödvändigt.

Eftersom kommandona till styrmodulen skickas på den form som visats ovan kommer all reglering behöva ske i styrmodulen. Därför kommer även mätdata att tas emot från huvudmodulen. Beroende på vilken typ av förflyttning roboten utför så kommer följande sensorvärden att skickas från huvudmodulen:

- fram/bak - avstånd till vägg fram och på sidorna.
- sväng höger/vänster - inga (kommandot används enbart vid manuell styrning och behöver således inte regleras).
- rotera höger/vänster - vinkel och vinkelhastighet.



Figur 14: Kopplingsschema över styrmodul

Dessa sensorvärden kommer skickas med en frekvens av 10-20 Hz förutsatt att reglering är aktiverad, det vill säga vid autonom styrning av roboten.

När styrmodulen har genomfört sitt körkommando från huvudmodulen skickar den en positiv flank på `PB4`. Denna utgång är kopplad till en avbrottsingång på huvudmodulen och tolkas som att styrmodulen är redo för ett nytt kommando.

4.1.2 Reglering

Vid manuell styrning av roboten sker ingen reglering. Detta för att användaren ska få ohämmad kontroll över roboten och förväntas sköta "regleringen" manuellt. Vid autonom styrning kommer dock alla förflyttningar behöva regleras. Som nämnt i avsnitt **??** kommer huvudmodulen skicka relevanta sensorvärden vid autonom körning. Beroende på vilket styrkommando som skickats väljer styrmodulen sedan en styrmod och reglerar rörelsen. De olika styrmoderna och hur deras reglering sker finns beskrivet mer ingående i avsnitt **??**.

4.1.3 Övrigt

Plattformen som roboten byggs på har färdiga kretsar för att styra chassits hjul. Hjulen kan styras parvis (höger och vänster) genom att sätta en bit för att bestämma rotationsriktningen och sedan skicka en PWM-signal för att bestämma rotationshastigheten. Roboten styrs alltså differentiellt, det vill säga genom olika hastighet på höger och vänster sida.

Roboten ska även vara utrustad med en gripklo, för att kunna greppa de förnödenheter som ska transporteras. Klons aktiva del består av ett servo av typen `RG-180` som styrs med en PWM-signal.

Det ska även finnas en alphanumerisk LCD-display, av typen `LCD JM162A`, på roboten, som används för att visa relevanta sensorvärden vid reglering. Den har totalt 16 pinnar, varav åtta är databitar, tre används för konfiguration och resterande är referenssignaler och strömtillförsel.

4.2 Hårdvara

Styrmodulen kommer kräva följande hårdvara:

- `ATmega16`
- Robotplattform (Terminator)
- Gripklo

4.3 Mjukvara

Koden för styrmodulen är skriven i C och består främst av två delar, reglering och display. Gripklon styrs direkt av ett styrkommando och kan endast öppnas respektive stängas.

4.3.1 Display

Displayen är en alphanumerisk 16x2 LCD-display och ska visa utvalda sensorvärden i realtid. I koden används drivrutinen *lcd.h* för att initiera och skriva text till displayen. Genom att anropa initieringsmetoden rensas displayen och pekaren placeras på rätt position. Därefter skrivs typen av mätvärde och det faktiska värdet ut. Tack vare två rader kan flera mätvärden skrivas ut samtidigt, förslagsvis ett i respektive hörn av displayen.

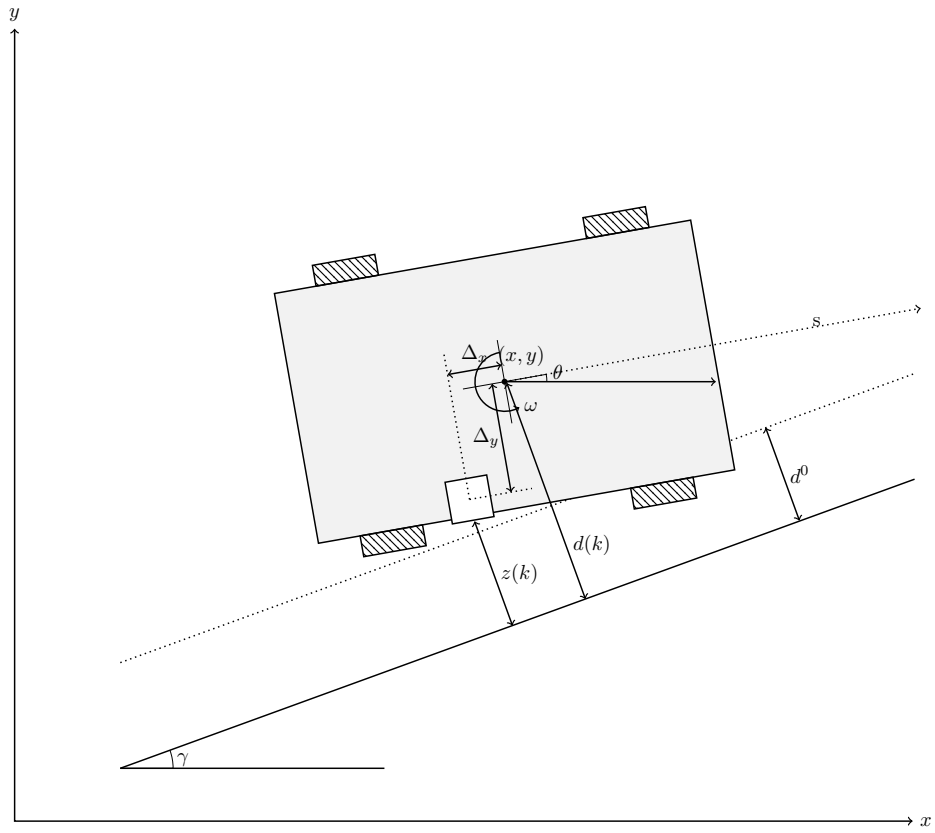
Utskriften till displayen sker i huvudloopen och i takt med att nya mätvärden tas emot.

4.3.2 Regleringsmodeller

Roboten kommer ha två olika styrmoder: *rak körning i korridor* och *rotation*. Av dessa två är det *rak körning i korridor* som presenterar den största utmaningarna. Reglerering av rotationen sker enkelt med en PD-reglering, varför den inte kommer beskrivas mer ingående här.

Rak körning i korridor Betrakta roboten och en vägg i ett kartesiskt koordinatsystem. Detta exempel kan jämföras med roboten i en korridor, där vi för enkelhetens skulle enbart betraktar ena väggen. Väggens vinkel i förhållande till koordinatsystemet är γ och robotens vinkel är θ. Robotens hastighet v fås genom derivering av avståndet till en godtycklig främre vägg s. Robotens uppmätta avstånd till väggen är z och det önskade avståndet är d^0. Robotens vinkelhastighet betecknas med ω och fås från gyrot. Detta illustreras i figur **??** nedan. Den faktiska roboten har två avståndssensorer per sida, varför robotens vinkel i förhållande till väggen $\theta - \gamma$ enkelt kan beräknas. Dessa har dock ersatts av en enskild sensor i figuren för att behålla överskådligheten.

Liksom vid rotation kommer en PD-reglering att användas vid rak körning i korridor. Modellen är dock något mer komplicerad. Eftersom roboten inte kan förflytta sig i sidled, och på så sätt enkelt anpassa sitt avstånd till väggen på sidan, måste sidoförflyttningen översättas till förändring av robotens vinkel i förhållande till väggen, vilket sedan kan översättas till en hastighetsskillnad i drivningen på robotens olika sidor. En mer detaljerad beskrivning av modellen finns i förstudien.



Figur 15: Väggföljning

4.3.3 Huvudloop och avbrott

Eftersom I^{2}C-kommunikationen är avbrottstyrd består styrmodulens uppdrag till största del av att besvara kommandon från huvudmodulen. När ett avbrott signaleras, betjänas det direkt och kallar på metoder utgående från typen av information som anländer. I fallet av manuell styrning mot styrkommandon sker ingen reglering gentemot väggar och kommandona översätts endast till en PWM-sekvens åt motorerna.

Då roboten navigerar i autonomnt läge består kommunikationen även av sensordata från huvudmodulen. Här kallas en metod som läser in sensordata och lagrar dessa i lokala variabler. Dessa variabler används i huvudloopens reglering och på så sätt sker alltid regleringen utifrån de allra senaste sensorvärdena.

Även LCD-displayen uppdateras i huvudloopen. Det finns inget behov av att uppdatera displayen i takt med att nya sensorvärden anländer, eftersom det skulle medföra en lång avbrottsrutin och lämna mindre tid åt regleringen. I stället uppdateras LCD-displayen endast vid var tionde uppdatering av sensordata, denna fördelning kan komma att ändras under projektets gång.

Gripklon styrs av en separat metod och aktiveras endast då huvudmodulen informerar

styrmodulen om att den ska användas. Här återfinns en enkel reglermodell som tillåter styrmodulen att öppna respektive stänga gripklon.

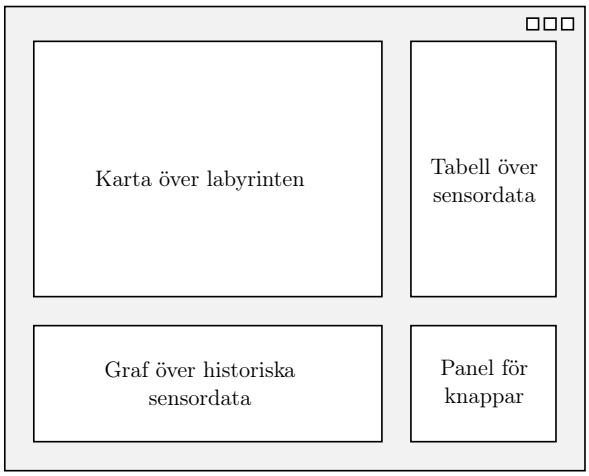
När styrmodulen har genomfört styrkommandot signaleras huvudmodulen genom ett avbrott. Då får huvudmodulen vetskap om att styrmodulen är redo för ytterligare ett styrkommando.

5 Delmodul 4 - Datormodul

Datormodulens syfte är att presentera ett gränssnitt för att styra, kartlägga och felsöka roboten i realtid. Kommunikationen sker via Bluetooth® och består av styrkommandon (datormodul → huvudmodul) samt mätvärden och karta (huvudmodul → datormodul).

5.1 Detaljerad beskrivning

Mjukvarans grafiska gränssnitt konstrueras enligt figur **??**.



Figur 16: Datormodulens GUI

Kartan över labyrinten ritas successivt upp, modulvis, under robotens färd, och uppdateras alltså efter varje förflyttning till en ny banmodul. I det fall att datorn kopplar upp till roboten efter påbörjad körning skickas all den hittills lagrade kartdatan direkt, varpå kartan sedan uppdateras som beskrivet ovan. Kartan presenteras sedan ur ett fågelperspektiv. I första hand ritas kartan i två dimensioner och i mån av tid utvecklas en tredimensionell karta där roboten följs ur ett tredjepersons-perspektiv.

Tabellen över sensordata skriver ut informationen från den senaste kommunikationen med huvudmodulen. Kommunikationen består av sensorvärden som huvudmodulen har tagit del av från sensormodulen, som skickas till datormodulen ett fåtal gånger per sekund. Eftersom sensormodulen konverterar de analoga sensor-spänningarna till SI-enheter presenterar tabell-rutan avstånd och vinklar istället för de faktiska spänningsnivåerna.

Grafen över historiska sensordata tar del av samma information som tabellen, men ritar även ut tidigare data i form av en graf. Denna ruta används främst för felsökning och ger ett tidsperspektiv till sensordatan.

Panelen för knappar visar vilken knappkombination från tangentbordet som är aktiv. Tabell **??** visar möjliga knappkombinationer och dess översättning till styrkommando. Panelen fyller i de knappar som hålls nere och varnar ifall en knappkombination är otillåten, exempelvis ifall ← och → trycks samtidigt.

Knapptryckning					Översättning
↑				=	Kör framåt
↑	←			=	Kör framåt vänster
↑		→		=	Kör framåt höger
	←			=	Rotera moturs
		→		=	Rotera medurs
	←		↓	=	Kör bakåt vänster
		→	↓	=	Kör bakåt höger
			↓	=	Kör bakåt

Tabell 2: Knapptryckningar och dess översättning

5.2 Hårdvara

Datormodulen kräver en dator kompatibel med Java och Bluetooth®.

5.3 Mjukvara

Gränssnittet programmeras i Java SE 7 och körs i Java Runtime Environment. Valet av programmeringsspråk faller på Java eftersom samma mjukvara ska köras på Windows, OSX och Linux. Kommunikationen via Bluetooth® sker med hjälp av ett existerande Bluetooth®-API utvecklat av Oracle för Java.

Swing och AWT används för att ge användaren ett grafiskt gränssnitt. Själva huvudklassen är centrerad kring Bluetooth®-kommunikationen och loopar i takt med att en ny sändning har skickats eller mottagits.

6 Intermodulär kommunikation

För kommunikation mellan de olika modulerna kommer en I^{2}C-buss och Bluetooth® användas.

6.1 I^{2}C-buss

För att kommunicera mellan de olika modulerna kommer den två-trådade I^{2}C-bussen att användas. Då modulerna i huvudsak kommer bestå av processorn `ATmega16` kommer allt vara kopplat till samma buss. Detta för att `ATmega16` enbart har en uppsättning av de ingångar som krävs, `SCL` och `SDA`. Eftersom mycket av tänkandet ligger i huvudmodulen får den master-status. Resterade blir slavar då informationen som ska mellan sensormodulen och styrmodulen även behövs i huvudmodulen.

6.1.1 Slavadresser

I tabell **??** listas den adress respektive modul ska konfigureras med.

Modul	**Adress**
Huvudmodul	1100100
Sensormodul	1100101
Styrmodul	1100110

Tabell 3: Modulernas adresser för kommunikation över I^{2}C-bussen

6.2 Bluetooth®-kommunikation

För dataöverföring via Bluetooth® kommer FireFly-modulen användas. Robotens enhet kommer vara kopplad till huvudmodulens processor. För att initiera kontakt används handskakningssignaler för att avgöra om vardera ände är redo att skicka respektive ta emot. Överföringen sker seriellt och när huvudmodulen mottagit informationen som datormodulen skickat genereras ett avbrott. Avbrott genereras genom att USART-gränssnittet tänder en flagga när det finns information att hämta i bufferten.

6.3 Informationsflöde

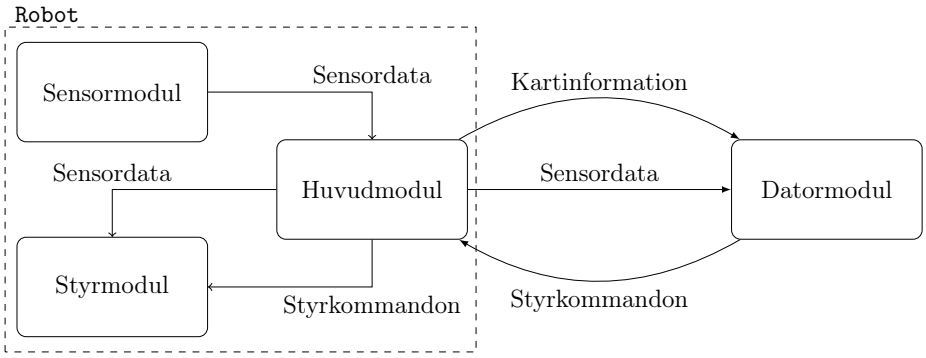
Eftersom flera olika typer av data kommer skickas mellan de olika enheterna inleds kommunikationen med ett kommunikations-ID som betecknar vilken typ av data som

kommer skickas. Varje datatyps respektive ID finns i tabellen nedan:

ID	**Datatyp**
0	Styrkommando
1	Kartinformation
2	Sensordata

Tabell 4: Tabell över kommunikations-ID

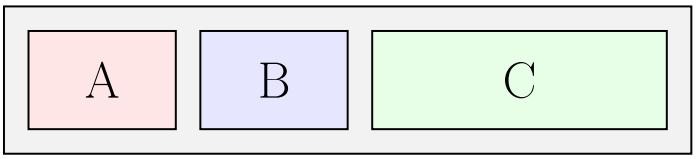
I figur **??** visualiseras informationsflödet. Där anges mottagare och sändare samt vilken typ av information som går mellan dem.



Figur 17: Schema över informationsflödet

6.3.1 Kommunikationsprotokoll - styrkommandon

Följande protokoll används för att skicka styrkommandon (gäller både för datormodul → huvudmodul och huvudmodul → styrmodul):



Figur 18: Protokoll för överföring av styrkommandon

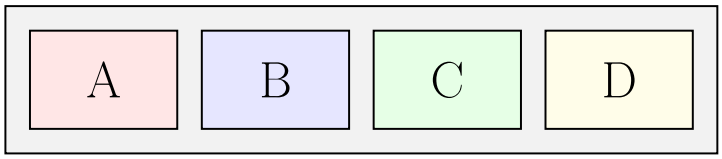
Respektive segment motsvarar följande:

- A (1 byte) - Kommunikations-ID
- B(1 byte) - Definierar vilket kommando eller vilken typ av värde som skickas, se tabell **??** och tabell **??** för specifika värden.
- C (2 byte) - Hastighet (om körkommando och inte reglering), sträcka/vinkel (om körkommando och reglering) eller värde (om sensorvärde).

6.3.2 Kommunikationsprotokoll - kartinformation

Huvudmodulen kommer för varje ny kartmodul skicka data till datormodulen som ritar upp den nya informationen grafiskt.

Varje korsning kommer att representeras som en nod och vägarna däremellan som bågar. För att så enkelt som möjligt överföra denna information från huvudmodulen till datormodulen kommer varje nod att representeras enligt protokollet nedan:



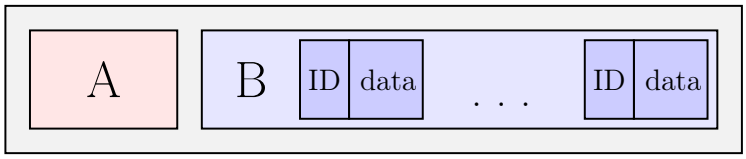
Figur 19: Protokoll för överföring av kartdata

Respektive segment motsvarar följande:

- A (1 byte) - Kommunikations-ID
- B (1 byte) - Motsvarar nodens x-koordinat (tillåtna värden: 0-15)
- C (1 byte) - Motsvarar nodens y-koordinat (tillåtna värden: 0-15)
- D (1 byte) - Representerar väggar kring noden (11 = vägg, 00 = ingen vägg) i nordlig, västlig, sydlig och östlig riktning (b0-1 = N, b2-3 = W, b4-5 = S, b6-7 = E).

6.3.3 Kommunikationsprotokoll - sensordata

Sensordata skickas enligt följande protokoll:



Figur 20: Protokoll för överföring av sensordata

Respektive segment motsvarar följande:

- A - Kommunikations-ID
- B - Sensordatapaket

Sensordatapaketet består av ett antal sensorvärden som skickas tillsammans med ett ID, för att identifiera vilken sensor värdet tillhör. Beroende på vilken styrmod roboten befinner sig i skickas olika sensorvärden. En tabell över vilka sensorvärden som skickas i respektive styrmod och en över de olika sensorernas ID finns nedan.

Styrmod	**Skickade sensordata**
Körning rakt fram i korridor	Avstånd till främre väggen s, avstånd till sidoväggen z, vinkelhastighet ω, vinkel i förhållande till väggen $(\theta - \gamma)$
Rotation	Vinkelhastighet ω

Tabell 5: Tabell över de olika styrmoderna. För förtydligande av variabler, se figur **??**

ID	**Sensor**	**Enhet**
0	IR-sensor höger-fram	cm
1	IR-sensor vänster-fram	cm
2	IR-sensor höger-bak	cm
3	IR-sensor vänster-bak	cm
4	Lasersensor	cm
5	Rörelse i x-led	m/s^2
6	Rörelse i y-led	m/s^2
7	Rörelse i z-led	m/s^2
8	Rotation kring x-axeln	rad/s
9	Rotation kring y-axeln	rad/s
10	Rotation kring z-axeln	rad/s
11	IR-sensor för mål	cm

Tabell 6: Tabell över sensorerna

ID	**Styrkommando**
0	Reglering av/på
1	Kör fram
2	Kör bak
3	Rotera åt höger
4	Rotera åt vänster
5	Gripklo öppna/stäng

Tabell 7: Tabell över styrkommandon

7 Implementering

Nedan beskrivs hur robotens delar ska implementeras.

7.1 Fysisk design

Roboten kommer bestå av 3 virkort där varje virkort motsvarar en av robotens tre moduler. Detta för att enkelt kunna se modulariteten och jobba med olika moduler samtidigt.

7.2 Testning och feedback

I början av utvecklingen av roboten färdigställs I^2C-bussen för att tidigt ha kommunikationen klar så att denna kan testas kontinuerligt under projektets gång. Hårdvara och mjukvara bör testas för sig först och sedan ihop för att lättare felsöka. Hjälp för testning är exempelvis Atmel Studio och JTAG ICE för on-chip debugging eller logikanalysator. Varje delmodul testas med simulerad data för att se att den fungerar som önskat. För att enklare kunna se vad som händer kan en extern klocka kopplas till processorn för att få tillräckligt låg frekvens. Optimalt är att varje delfunktion av roboten testas så fort den är klar för att kunna säkerställa att den fungerar innan utvecklingsarbetet fortgår.

Inom utvecklingsarbetet för varje aktivitet hanteras också eventuell felhantering. Denna felhantering görs på alla eventuella problem som kan tänkas uppstå. Till processorn kan exempelvis en LED-lampa kopplas för att få feedback att något gått fel. I programvara i datorn är det lämpligt att använda sig av en körlogg för att kunna gå tillbaka och se vilken data som skickades när för att smidigt kunna se var och när något gick fel om det gjorde det. Den LCD-display som är kopplad på styrmodulen ska förutom sensordata även kunna visa styrkommandon för att kunna se vad som händer vid körning.

7.3 Sampling

Det som begränsar tiden mellan samplingar av sensordata är hur lång tid A/D omvandlingen tar för den data som tar längst tid att omvandla. Sampling bör ske så ofta som möjligt för så bra värde som möjligt så att styrmodulen kan få uppdaterade värden skickade till sig. Sampling kommer ske med frekvens mellan 10-20 Hz för att hinna med omvandling av IR-sensorernas data.