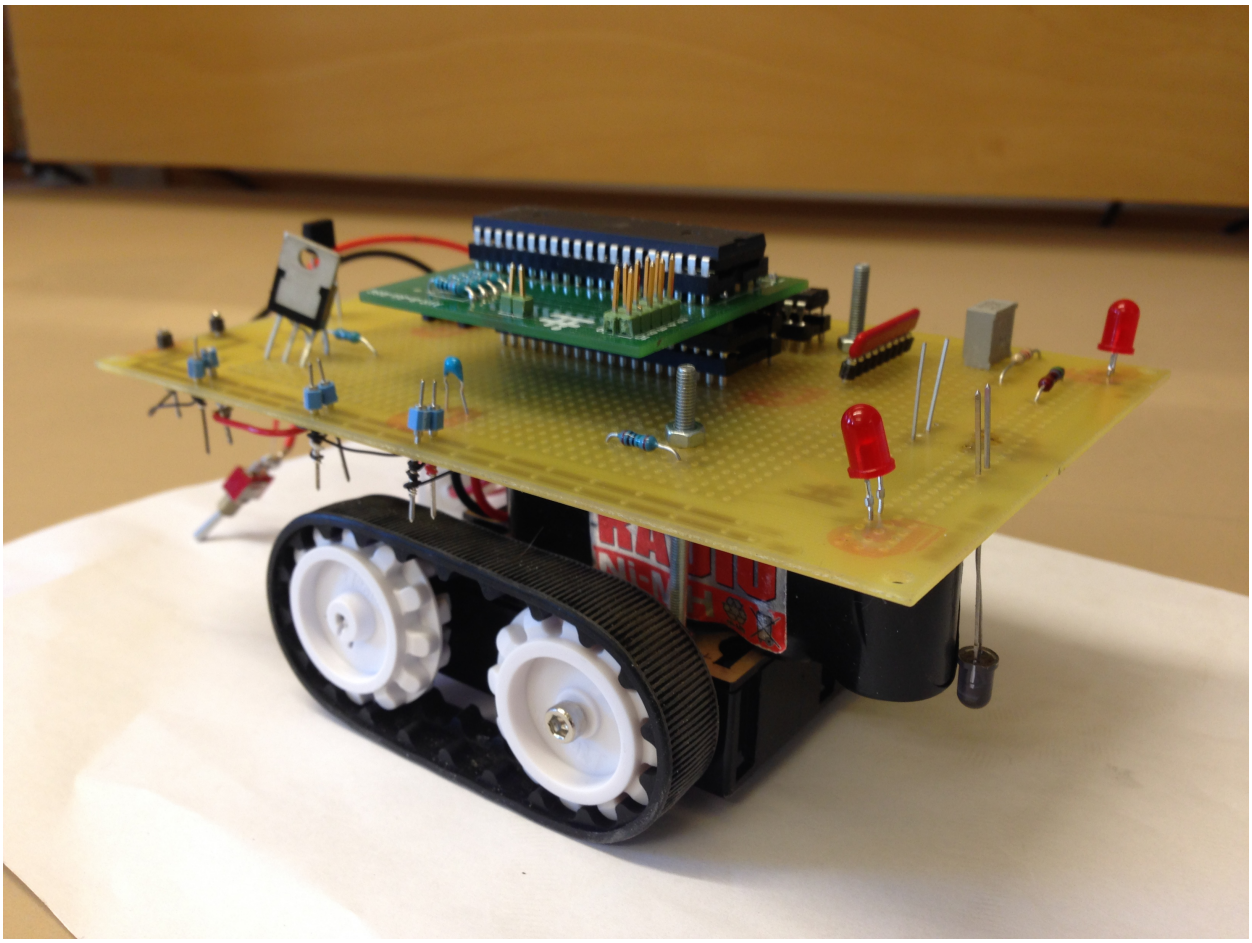
# Minesweeper



Grupp 12
Johan Barkfors & Karl Hedlund
Handledare: Bertil Lindvall
Projektarbete inom kursen EITF11 Digitala projekt
Vårterminen 2014
Lunds tekniska högskola
Elektro- och informationsteknik

## Innehållsförteckning

# 1. Inledning

## 1.1 Bakgrund

I denna rapport följer en beskrivning av projektet *Minesweeper* som utförts i kursen Digitala projekt på Lunds tekniska högskola. Rapporten innefattar kravspecifikationen och konstruktionen av hård- och mjukvara.

## 1.2 Syfte

Syftet med Minesweeper har varit att gruppen ska ta sig an grundläggande kunskaper inom embedded systems.

## 1.3 Disposition

Rapporten är indelad i kapitel. Först demostreras kravspecifikationen för projektet och följs sedan av hårdvaru- och mjukvarubeskrivning, reflektioner, resultat och, slutsatser. Längst bak i arbetet bifogas referenser och bilagor. Rapporten är skriven på ett sådant sett att den förutsätter grundkunskaper inom ämnet *Embedded systems*.

## 1.3 Avgränsningar

Rapporten beskriver planeringen, arbetet med, och resultatet av projektet Minesweeper.

## 2. Kravspecifikation

I detta avsnitt följer en kravspecifikation för *Minröjaren*, syftet med denna är att klargöra *primära* respektive *sekundära* krav för projektet. De primära kraven avser till de krav som projektet *måste* uppfylla, de sekundära kraven avser till de som implementeras *i mån av tid*.

Minröjaren är ett fordon som styrs från distans och som kan identifiera abnormiteter (minor) på en markyta i form av reflektiva ytor. Vid identifiering av en mina ska Minröjaren skicka en varningssignal till operatören.

### 2.1 Primära krav

Fordonet ska:

- Vara batteridrivet.
- Kunna styras av en operatör med en IR-kontroll.
- Ha förmågan att köra fram- och baklänges, samt svänga höger respektive vänster.
- Ha förmågan att identifiera vita objekt (minor) på vissa ytor.

När Minröjaren identifierar en mina ska denna sätta igång ett varningsprotokoll med följande utformning:

- Stanna, samt ignorera operatörens kommandon temporärt.
- Signalera fara genom att lysa med en eller flera dioder.
- Efter en förutbestämd tid återfår operatören kontrollen och varningssignalerna ska avbrytas.

### 2.2 Sekundära krav

Fordonet ska i mån av tid:

- I varningsprotokollet inkludera en ljudsignal för att indikera fara.
- Kunna verka självdrivande på ett område genom att identifiera hinder och hitta lämpliga vägar automatiskt.

## 3. Hårdvara

I följande avsnitt beskrivs den hårdvara som används för Minesweeper.

### 3.1 Processor

Minesweeper har en microprocessor av typen Atmega16 som tillhandahåller totalt 40 pinnar, varav 32 är tillgängliga för I/O. Det finns totalt fyra portar A, B, C och D som har olika funktioner tillgängliga. Till Minesweeper har totalt 21 pinnar använts, var och varför beskrivs i kommande avsnitt.

#### 3.2.1 Transformator

Processorn kan maximalt hantera 5V och eftersom batteriet ger 6V har en transformator kopplats in. Transformatorn är av typen LP38855 (Figur 1).

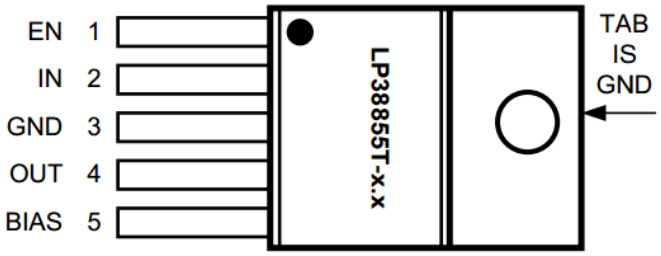



**Figure 1 - LP38855**

### 3.2 Motor

Minesweeper använder sig en motorsats från tillverkaren Polou som har fyra färdigmonterade hjul. Satsen har ingen inre logik, utan kör med en hastighet proportionell med den inmatade spänningen. Motorerna är oberoende av varandra och kan röra sig fram- och tillbaka, vilket tillåter olika körlägen.

#### 3.2.1 H-brygga

För att sända signaler till motorn används en H-brygga av typen DRV8833 (Figur 2). H-bryggan får instruktioner från processorn genom de fyra IN-pinnarna och styr motorerna genom OUT-pinnarna.

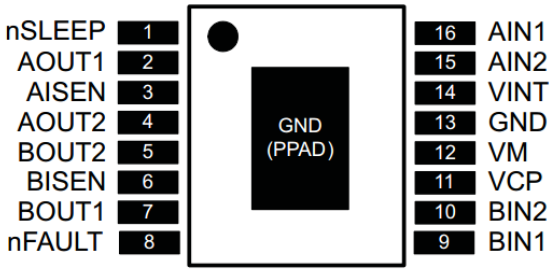



**Figure 2 - DRV8833**

### 3.3 Styrning
Minesweeper fjärrstyrs med hjälp av en fjärrkontroll. Nedan följer en beskrivning av de komponenter som hanterar fjärrstyrningen.

#### 3.3.1 Fjärrkontroll
Fjärrkontrollen som användes är färdigbyggd, troligtvis av fabrikatet Philips, och använder sig av RC-5 protokollet.

#### 3.3.2 IR-Receiver
IR-signalerna från fjärrkontrollen registreras av en receiver av typen IRM8608S. Receivern skickar digitala spänningspulser till IR-decodern.

#### 3.3.3 IR-Decoder
Spänningspulserna från IR-receivern omvandlas i decodern av typen SAA3049A (Figur 3) till ett tal mellan 0 och 255 (8 bitar). Hur IR-decodern är kopplad finns beskrivet i kopplingschemat (Bilaga 2). Talen blir inversen av den knapp som trycktes ner, 1 blir t.ex. 11111110.

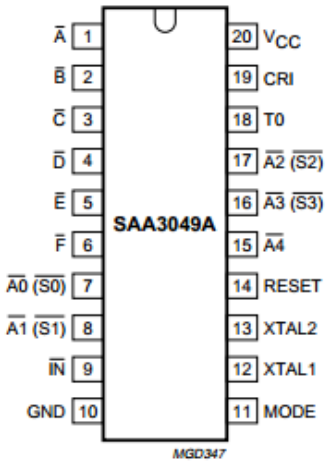



**Figure 3 - SAA3049A**

### 3.4 Ljussensor
Tanken är att Minesweeper ska kunna upptäcka skillnader på markytan, därför en ljussensor byggts. Principen är att en IR-emitter skickar en signal och att en fotodiod fångar signalen. Nedan följer en beskrivning av de ingående komponenterna.

#### 3.4.1 IR-Emitter
En IR-emitter av typen LD271 används för att kunna skicka ut en IR-signal. Emittern är kopplad direkt till batteriet och skickar signaler sålänge som spänningen är påslagen.

### 3.4.2 Fotodiod
En fotodiod av typen SFH203FA används för att ta emot IR-signaler. Fotodioden tar lätt åt sig brus från omgivningen och för att minimera detta har den delvis täckts av en skärm.

## 3.5 Kondensator
För att minska brus i AREF används en kondensator, hur den är kopplad finns beskrivet i kopplingsschemat (Bilaga 2).

## 3.6 Motstånd
Minesweeper har flera motstånd, dessa används för att garantera att rätt strömmängd når komponenterna. Vilka motstånd som använts och hur finns beskrivet i kopplingschemat (Bilaga 2).

# 4. Mjukvara

I följande avsnitt beskrivs den mjukvara som utvecklats åt Minesweeper.

## 4.1 Språk

Utvecklingsmiljön AtmelStudio 6 har använts och all programvara har skrivits i C. Projektmedlemarna har haft små eller inga förkunskaper i C och en stor del av projektet har handlat om att lära sig programmera i detta språk.

## 4.2 Kod

Koden finns i sin helhet i Bilaga 1, men består av metoderna:

initPorts – initierar alla portar som ska användas.

initADC – aktiverar den inbyggda AD-omvandlarfunktionen på processorns port A.

initInterrupts – startar en klocka som orsakar konstanta interrupts.

read_ADC – sätter igång en omvandling av AD-omvandlaren och returnerar 8 bitar av svaret (0-255)

check_light – kontrollerar om bilen står vid en mina och initierar rätt protokoll (varning eller avbryt varning).

init_warning – avbryter bilens aktivitet och tänder varningslamporna.

abort_warning – släcker varningslamporna.

ISR – den metod som körs när ett interrupt sker. Metoden kontrollerar om bilen får köra, dvs inte står vid en mina. Om så är fallet utförs det kommando som senast gavs av fjärrkontrollen.

Forward, back, left, right, stop – kör bilen i motsvarande riktning.

## 5. Arbetet

I följande avsnitt beskrivs den arbetsprocess som utförs.

### 5.1 Förkunskaper

Ingen av projektmedlemmarna hade några förkunskaper eller någon tidigare erfarenhet i ämnet embedded systems samtidigt som kunskaperna inom C var mycket begränsade.

### 5.2 Utmaningar

Under projektets gång stötte projektgruppen på flertalet trösklar. Trösklarna var av varierande former och nedan finns ett antal av de större i en sammanfattad form.

- Kopplingsschemat

Den första tröskeln som gruppen stötte på var kopplingschemat. Anledningen till varför det var en utmaning var eftersom gruppen inte hade någon förkunskap i att bygga ett kopplingschema. Vad som var svårast var att föreställa sig vad som skulle behövas för att uppfylla kravbilden. Med att erfarenheten blev större blev detta desto simplare.

- Hårdvarumontering

Även om monteringen av komponeterna kan te sig simpelt var de kanske den största utmaningen under projektet. De problem som var störst var att veta vilken pinne som skulle användas till vad och vid vilket tillfälle som det krävdes ett motstånd. Även detta blev allt enklare under projektets gång.

Ett annat stort problem med monteringen av hårdvaran var att lödningen i vissa fall visade sig vara bristfällig. Felsökningen av denna bristfälliga lödning var svår eftersom felen framstod som mycket slumpmässiga, lösningen på problemet blev att gruppen i större utsträckning tvinnade sladdarna.

- Mjukvatruutveckling

Eftersom gruppen inte hade några förkunskaper av liknande utveckling var programmeringen mycket utmanande. Det som var svårast var att samla rätt information från databladen och ta reda på vilka register som lämpade sig bäst. Att lära sig hur man hanterar ett interupt var svårt och gruppen behövde mycket tid för att lära sig hantera specifika interupts.

### 5.3 Lärdommar

Minesweeper har visat sig vara mycket lärorikt för projektgruppen och har inneburit ökad kunskap inom flera områden. Lärdommarna har varit så stora att de kommer att kunna vara till nytta i framtiden, antingen för genomförandet av- eller bedömningen av- liknande projekt.

# 6. Resultat och slutsats

I följande avsnitt beskrivs resultatet av Minesweeper.

## 6.1 Resultat

Resultatet av Minesweeper är en bil som uppfyller kravspecifikationens primära mål. Nedan följer några egenskaper som bör belysas ytterligare.

- Minesweeper är svårstyrd och svarar dåligt på nya kommandon, anledningen till varför den svarar dåligt antas vara därför att IR-recivern (IRM8608S) inte fångar upp de kommandon som skickas från fjärrkontrollen.
- Minesweeper har en hög hastighet och hinner därför förflytta sig en bit efter att den detekterat en ljusannomalitet. I fallet där Minesweeper hinner passera ljusannomaliteten efter att den stannat kommer den att att börja åka igen automatiskt.
- Minesweeper kan enbart göra en svängning i taget, därefter måste operatören trycka på stoppknappen för att kunna svänga igen.
- Under projektet beslutades det att implementera en on/off knapp till ljussensorn skulle byggas. För att kunna avgöra om ljussensorn är avstängd eller inte har en lampa installerats. Detta var inte angivet i kravspecifikationen.

## 6.2 Förslag på förbättringar

I en vidareutveckling av produkten skulle följande förbättringar kunna göras:

- Växla ner motorerna för att minska hastigheten. Skulle ge bättre kontroll och precision.
- Använd en kraftfullare IR-mottagare samt sändare (fjärrkontroll) för att öka räckviden.
- Använd fler IR- och fotodioder för att öka känsligheten och precisionen vid detektion av minor.
- I nuläget kan inte bilen svänga och köra framåt samtidigt. Detta skulle kunna åtgärdas med modifierad kod och användning av externa interrupts.
- För att göra varningen mer effektfull kan bilen även utsända en ljudsignal.

## 6.3 Slutsats

Minesweeper har varit lärorikt och utmanande. Det har gett projektmedlemarna övning i planering och utförande att projekt samtidigt som den tekniska kunskapen inom elektroteknik och datateknik har förbättrats avsevärt. Antagandet är att lärdommarna från Minesweeper kommer att hjälpa gruppmedlemmarna till att enklare se behoven och utmaningarna i andra teknikprojekt, både i egenskapen som projektledare och utvecklare.

## Bilaga 1 – Kod

```
#include <avr/io.h>
#include <util/delay.h>
#include <avr/interrupt.h>

#define bforward 0b11111111                    //Knapp 2
#define bback        0b11111101                //Knapp 0
#define bleft 0b11111010                       //Knapp 5
#define bright 0b11111000                      //Knapp 7
#define photodiod_on 0b11111100                //Knapp 3
#define photodiod_off 0b11111011               //Knapp 4
#define bstop 0b11111001                       //Knapp 6

int right_done = 0;                            //Om bilen precis har kört till höger
int left_done = 0;                             //Om bilen precis har kört till vänster

void initPorts()
{
       DDRA = 0b11011000;          //IR MODE OUT|IR RESET OUT|IR DIOD IN|RÖD DIOD OUT x2
       DDRB = 0b00000000;          //IN från IR RECEIVER
       DDRC = 0b00000000;
       DDRD = 0b11111111;          //OUT till motorerna
       PORTA = 0b00000000;
       PORTB = 0b11110000;
       PINB = 0b00000000;
       PORTD = 0b00000000;
}

void initADC(){
       ADMUX=(1<<REFS0) |(1<<ADLAR);      //Sätt referensspänning till Vcc (+5V) och
                                          //vänsterjustera ADC-registret
       ADCSRA |= (1<<ADEN) |(1<<ADPS0) |(1<<ADPS1);     //Aktivera AD-konvertering, sätt
                                                        //konverteringsklockan till 1/8 av
                                                        //processorns
}

void initInterrupts()
{
       //Aktivera en timer
       TCNT0 = 0x00;
       TCCR0 = 0x01;
       TIMSK = 0x01;
       sei();                                           //Aktivera interupts
}

uint8_t read_ADC(){
       ADMUX |= 0b00000101;                             //Läs från PA5
       ADCSRA |= (1<<ADSC);                             //Starta konvertering A->D
       while(!(ADCSRA &(1<<ADIF)));                     //Gör ingenting så länge
       konverteringen inte är klar
       return ADCH;                                     //Returnera 8 bitar (0-255)
}

void init_warning()
{
       PORTA |= (1<<PA4);                               //Tänd lampan
       stop();                                          //Stoppa motorerna
```

```
	_delay_ms(20);                                  //Stå still i X sekunder
}

void abort_warning()
{
	PORTA &= ~(1 << PA4);                                  //Släck lampan
}

check_light(){
	uint8_t light_level = read_ADC();        //Uppmät ljusnivå
	if(light_level > 250)                    // && is_checking_for_light == 1)
                        //Om nivån är tillräckligt hög : Initiera varningsprotokoll
	{
		init_warning();
	}
	else                                  //Annars: Avbryt varningsprotokoll
	{
		abort_warning();
	}
}


ISR(TIMER0_OVF_vect)
{
	if(PORTA &= (1<<PA3)){
		check_light();              //Kontrollera om bilen är vid en mina, annars...
	}
	else                                  //Annars: Avbryt varningsprotokoll
	{
		abort_warning();
	}

	if(PINB == bforward){             //Kör framåt (knapp 2)
		left_done = 0;
		right_done = 0;
		forward();
	}
	else if(PINB == bleft){           //Kör vänster (knapp 5)
		right_done = 0;
		if(left_done==0)
			left();
		else stop();
	}
	else if(PINB == bright){          //Kör höger (knapp 7)
		left_done = 0;
		if(right_done==0)
			right();
		else stop();
	}
	else if(PINB == bback){           //Backa (knapp 0)
		left_done = 0;
		right_done = 0;
		back();
	}
	else if(PINB == bstop){           //Stanna (knapp 6)
		left_done = 0;
		right_done = 0;
		stop();
```

```
        }
        else if(PINB == photodiod_on){                  //Sätt på diod (knapp 3)
                PORTA |= (1<<PA3);
        }
        else if(PINB == photodiod_off){                 //Stäng av diod (knapp 4)
                PORTA &= ~(1 << PA3);
}
}

void forward()
{
        PORTD = 0b10100000;
}

void back()
{
        PORTD = 0b01010000;
}

void stop()
{
        PORTD = 0b00000000;
        _delay_ms(500);
}

void right()
{
        PORTD = 0b00110000;
        _delay_ms(500);
        stop();
        right_done=1;
}

void left()
{
        PORTD = 0b11000000;
        _delay_ms(500);
        stop();
        left_done=1;
}

void main(void)
{
        initPorts();
        initInterrupts();
        initADC();
        PORTD = 0;

    while(1)
    {
    }
}
```

## Bilaga 2 – Kretsschema