Projekt Car-0-line

Av Grupp 11

Gabriel Niedziolka,

Simon Antonsson

& Wesley Carter

Sammanfattning

Inom ramen för delkursen Digitala system fick vi alla i uppdrag att gruppvis bygga en prototyp med en ATmega-1284 mikroprocessor. Gruppen valde att konstruera och programmera en radiostyrd bil. Målet var att bilen skulle kunna göra larvfotssvängningar, köra framåt och bakåt, genomföra svängningar när den skulle köra i respektive färdriktning, tuta och tända och släcka ljusen med hjälp av en trådlöst sammankopplad kontroll. Till en början planerades ett kretsschema och en kravsspeckstabell för de komponeneter som skulle behövas. Efter att ha blivit godkänd fick gruppen efterfrågat material och byggde samt programmerade prototypen. Resultatet var en prototyp som uppfyllde målet dock med en konstant istället för modulär hastighet.

Nyckelord

Projekt Car-0-line, Radiostyrd bil, Radiokontrol, Atmega 1284, kretsschema

Projekt Car-0-line, RC-carr, Radiocontroller, Atmega 1284, circuit diagram

Abstract

In the confines of the class Digital systems we received the task to construct a prototype, in groups, with the main component being the microprocessor ATmega-1284. The group responsible for this project decided to construct and program a radio-controlled car also known as a rc-car. The goal of the project was to build a rc-car that could, with a wireless connected controller, realize caterpillar tread turns, drive forwards and backwards, realize turns when driving in either direction, honk a horn, and ignite and kill its own lights. At first, the group

planned what components were necessary and compiled a circuit diagram as toconstruct the controller and the rc-car. After receiving approval to proceed with the requested components, we constructed and programmed the prototype and our result was one who achieved the goals, although the car only has a constant speed instead of a modular one.

1 Inledning

Denna rapport är kopplat till avslutningsprojektet i delkursen Digitala system. Projektet gick ut på att studenterna gruppvis skulle bygga en valfri prototyp med mikroprocessorn ATmega-1284. Prototypen skulle uppfylla en valfri funktion och antigen fungera fullständigt eller delvis i slutet av projekttiden. Under arbetets gång skulle gruppmedlemmarna samarbeta för att bygga, löda och programmera prototypen, skapa en hemsida som handlar om prototypen,

skriva en skriftlig rapport och hålla en muntlig kort redovisning. Inom ramen för projektet valde gruppen att konstruera en fullständig radiostyrd bil med lyktor och tuta.

1.1 Syfte

Syftet med projektet var att konstruera en prototyp med minst en ATmega-1284 mikroprocessor. Prototypen skulle dessutom uppfylla utvald funktion och gruppmedlemmarna skulle programmera och löda kretskort för att uppnå målet. Syftet var även att studenterna i mindre grupp skulle kunna genomföra projektet och redovisa detta till handledare och medstuderande i form av en skriftlig rapport, hemsida och en muntlig redovisning.

1.2 Målformulering

Gruppen valde att bygga en fullt fungerande radiostyrd bil som skulle kunna göra larvfotssvängningar, köra framåt och bakåt, genomföra svängningar när den skulle köra i respektive färdriktning, tuta, tända och släcka ljusen med hjälp av en trådlöst sammankopplad kontroll. För den sakens skull krävdes det att ett kretsschema upprättades, enligt vilket kontrollen och bilen byggdes ihop och samtidigt programmerades.

1.3 Problemformulering

Den radiostyrda bilen skall kunna:

1. Göra larvfotssvängningar
2. Köra framåt och bakåt
3. Genomföra svängningar när den skulle köra i respektive färdriktning
4. Tuta
5. Tända och släcka ljusen
6. Allt med hjälp av en trådlöst sammankopplad kontroll

1.4 Motivering av projektsarbetet

Vi valde att konstruera en radiostyrd bil då vi alla som barn ofta lekt med en och ansåg att det skulle vara intressant, lärorikt och framför allt häftigt att ha genomfört ett sådant arbete.

2 Teknisk bakgrund

För att skapa en radiostyrd bil och motsvarande kontroll behövs det många olika delar, en av de viktigaste delarna är processorn där vi använde två Atmega-1284 som vi programmerade i C. En annan viktig del är sändaren och mottagaren där vi använde två Parallax-433 med hjälp av UART för att sända och ta emot data.

2.1 Hårdvara

För att bygga bilen och tillhörande kontroll så användes en färdig bas med hjul och motorer. Det användes två stycken 10k-resistorer till knapparna för att vända värdena som kommer in till ettor istället för nollor, så när man trycker på knappen så får man en etta. För att styra bilen och kontrollen utnyttjades två Atmega-1284, en på varje kretskort. Parallax-433 användes för att kunna skicka kommandon från kontrollen till bilen. Två DC-motorer användes och drevs med PWM (pulsbreddsmodulering).

Komponent	Antal	Information
Atmega1284	2	Processor som programmerades för att styra alla delar.
DRV8833 Dual H-Bridge Drive	1	En H-Brygga som används för att styra DC-motorer.
Parallax 433	2	En sändare och mottagare beroende på vad man ger den för värde. Som mottagare så tar den upp brus när det inte finns någon sändare kopplad till den.
AA-Baterier	4	Strömkälla till bilen. 1,3V per batteri.
DC-motorer	2	Driver hjulen och får värden från H-Bryggan.

2.2 Mjukvara

För att bilen skulle kunna använda hårdvaran måste processorn vara programmerad med kod så att den skulle veta vad den skulle göra. All kod programmerades i Atmel Studio 7 i språket C. För att bilen skulle kunna veta vad den skulle göra vid olika knapptryckningar så skapades ett protokoll där en int skickades till bilen, värdet maskades sedan i bilens kod och beroende på vilken bit i värdet som var en etta så gjordes olika kommandon. Hastigheten på bilen ställdes in genom att ändra på OCRxx vilket är längden på pulsen, då längre puls ger mer spänning, vilket ger högre hastighet.

3 Metod

Arbetet började med att vi bestämde vad vi skulle bygga. Sedan började vi planera vilka komponenter som skulle behövas för att kunna göra en radiostyrd bil. Vi valde ut våra komponenter och började direkt efter att rita ett kopplingsschema för båda kretsarna.

Efter att handledaren godkänt arbetet fick vi våra komponenter, därefter började vi konstruera och löda kretsarna på mönsterkortet. Komponenterna kopplades samman med koppartrådar och kablar beroende på vilken kret och tolerans behov. När kontrollens krets var så gott som klar började vi även programmera koden till båda kretsarna, och när kretsarna väl var färdiga testade vi programmeringen och finjusterade koden.

Därefter felsökte vi koden, provkörde bilen och korrigerade delar i kretsen. När vi väl var färdiga med prototypen satte vi igång med att skapa hemsida och skriva rapporten

4 Resultat

Den radiostyrda bilen kan genomföra allt det gruppen planerade att den skulle kunna genomföra. Målet var att bygga en radiostyrd bil som kunde genom en trådlöst kopplad kontroll köra framåt och bakåt, genomföra svängningar när den körde i respektive färdriktning, larvfotssvängningar, tuta och tända och släcka ljusen. Dessvärre tog det lite mer tid att löda än förväntat, men eftersom vi även programmerade och gjorde det näst intill felfritt hade vi lagom tid till att korrigera.

5 Slutsats

Med detta projekt har vi lyckats att konstruera en radiostyrd bil som kan:

1. Genomföra larvfotssvängningar
2. Köra framåt och bakåt
3. Genomföra svängningar när den skulle köra i respektive färdriktning
4. Tuta
5. Tända och släcka ljusen
6. Allt med hjälp av en trådlöst sammankopplad kontroll

5.1 Framtida Utveckling

I framtiden kan denna tekniken användas för att få transport och logistik att vara komplett trådlös. Denna tekniken utvecklas redan och företag utövar tester för att utveckla tekniken till exempelvis lastbilschaufförer som i så fall inte skulle behöva resa lika mycket och skulle istället kunna stanna kvar vid sina hem.

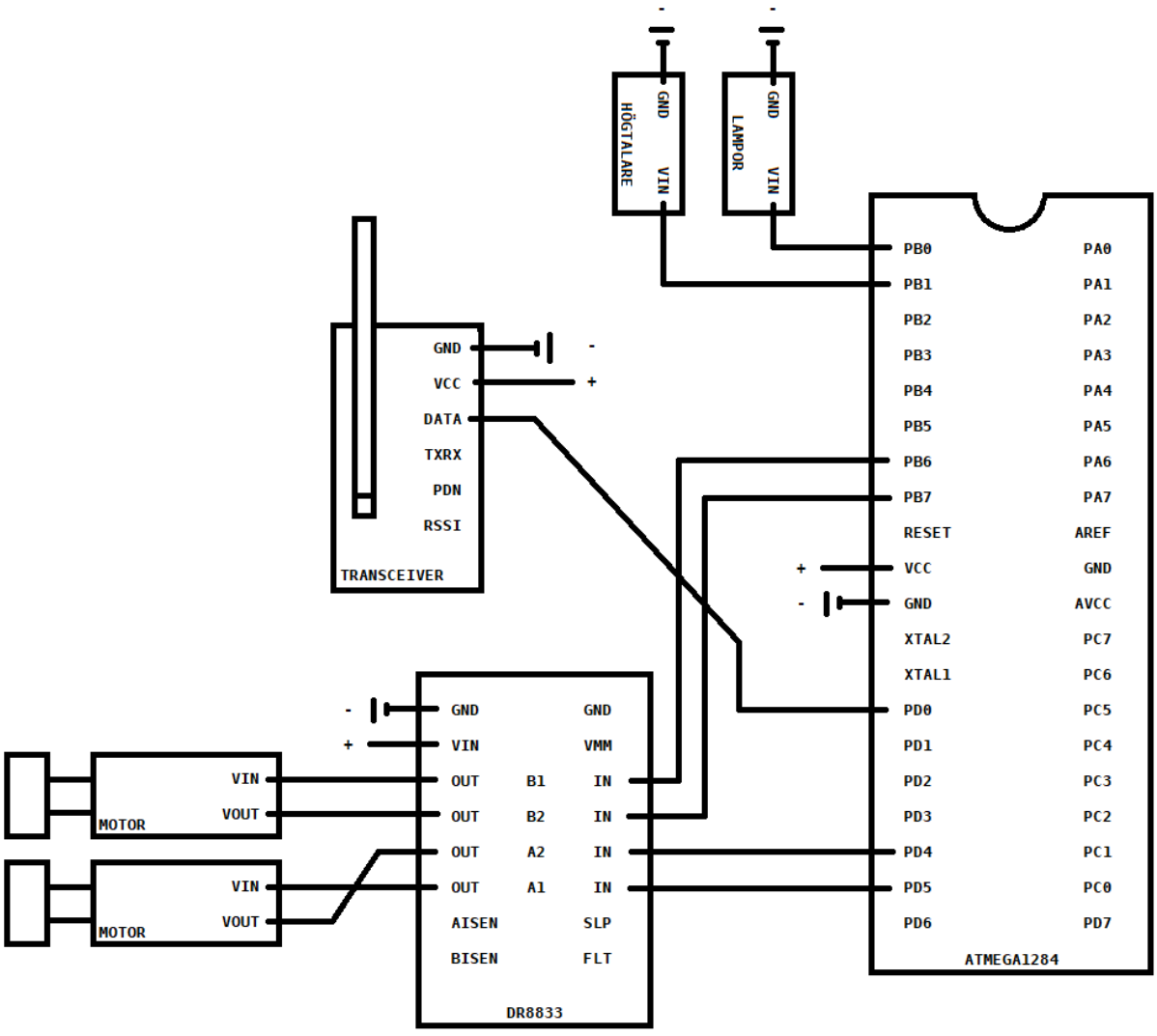
6 Källförteckning

H-brygga:
https://www.eit.lth.se/fileadmin/eit/courses/datablad/Lawicell/drv8833_Dual_motor_drive.pdf
ATmega-1284:
https://www.eit.lth.se/fileadmin/eit/courses/datablad/Processors/ATmega1284.pdf

Parallax 433 transceiver:
https://www.mouser.com/datasheet/2/321/parallax_27982-433-mhz-rf-transceiver-documentatio-1197467.pdf

7 Appendix

Kopplingsschema för bilen:



Kod för bilen:

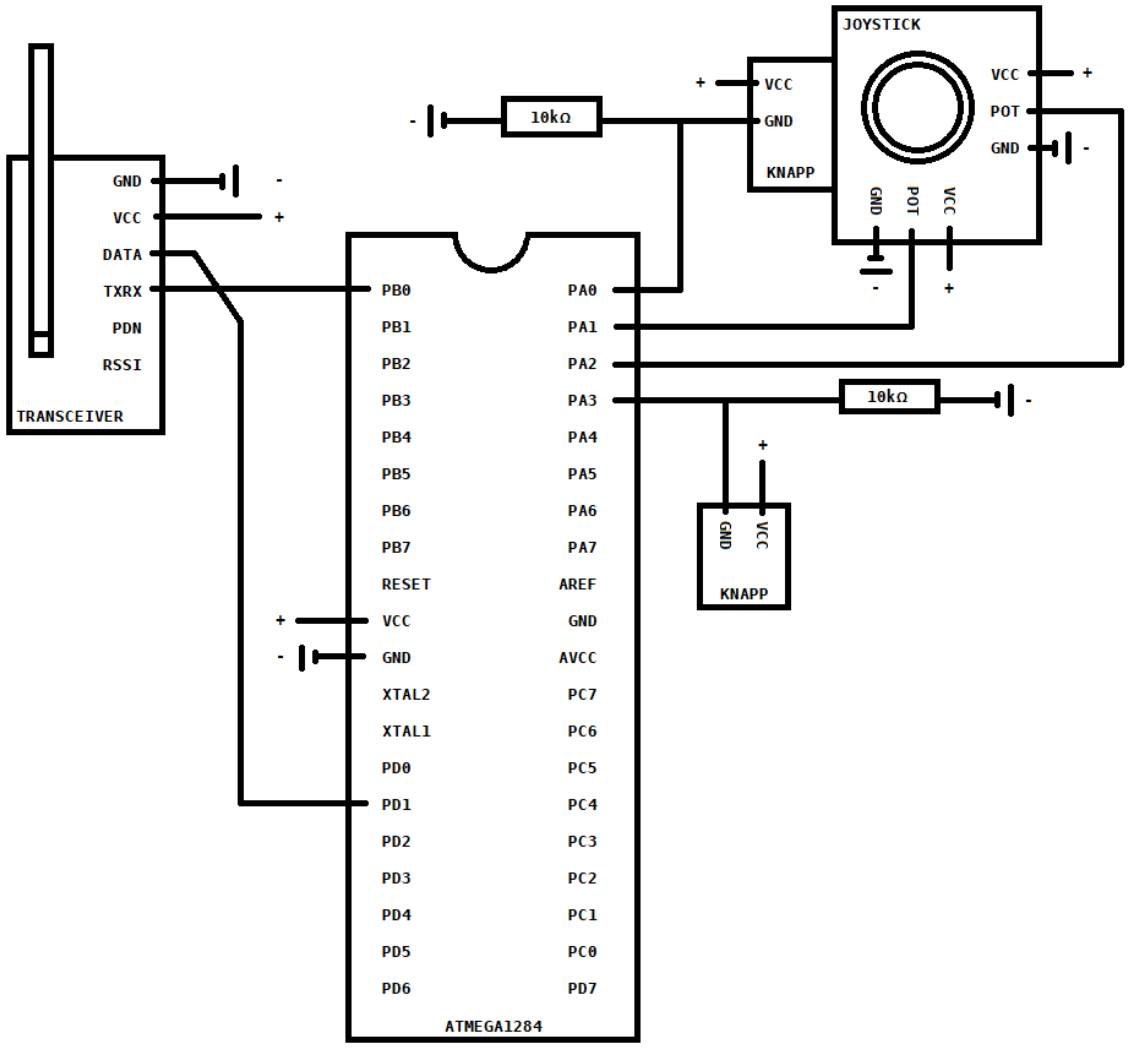
```
#include <avr/io.h>

void uart_init();
```

```
uint8_t uart_recieve();
void timer_init();


uint8_t data;
uint8_t lampswitch = 0;
uint8_t lamp = 0;

int main(void)
{
          uart_init();
          timer_init();
          DDRB = 0xff;       //sätter B och D  till utportar
          DDRD = 0xff;



    while (1)
    {

                    data = uart_recieve();      //får in datan från RX

                    //maskar för att få ut värden från protocol
                    uint8_t framMask = data & (0b00100000);
                    uint8_t bakMask = data & (0b00010000);
                    uint8_t hogerMask = data & (0b00001000);
                    uint8_t vansterMask = data & (0b00000100);
                    uint8_t tutaMask = data & (0b00000010);
                    uint8_t lamporMask = data & (0b00000001);

                    if(framMask == (0b00100000)){ //Fram
                              OCR3A = 25;                 //ändrar hastighet på hjulen
                              OCR3B = 25;
                              OCR1A = 25;
                              OCR1B = 25;

                              if (hogerMask == (0b00001000)){ //svänger höger medans man kör, höger Avstängd, Vänster fram

                                        TCCR3A = (0b00100010);      //PB7 på, Avstängd PB6      (Höger fram)
                                        TCCR1A = (0b00000010);      //PB5 på, Avstängd PD4      (vänster Bak)

                              }else if (vansterMask == (0b00000100)){ // svänger vänster medans man kör, Höger fram,
vänster Avstängd

                                        TCCR1A = (0b00100010);      //PB4 på, Avstängd PB5      (vänster Fram)
                                        TCCR3A = (0b00000010);      //PB6 på, Avstängd PB7  (Höger bak)


                              }else{              //om den bara ska åka fram

                                        TCCR3A = (0b00100010);      //PB7 på, Avstängd PB6      (Höger fram)
                                        TCCR1A = (0b00100010);      //PB4 på, Avstängd PB5      (vänster Fram)
                              }
```

```
}else if (bakMask == (0b00010000)){ //Bak

          OCR3A = 12;                //ändrar hastighet på hjulen
          OCR3B = 12;
          OCR1A = 12;
          OCR1B = 12;

          TCCR3A = (0b10000010);      //PB6 på, Avstängd PB7  (Höger bak)
          TCCR1A = (0b10000010);      //PB5 på, Avstängd PD4      (vänster Bak)



}else if (hogerMask == (0b00001000)){ //höger Bak, Vänster fram

          OCR3A = 12;                //ändrar hastighet på hjulen
          OCR3B = 12;
          OCR1A = 12;
          OCR1B = 12;

          TCCR3A = (0b00100010);      //PB7 på, Avstängd PB6      (Höger fram)
          TCCR1A = (0b10000010);      //PB5 på, Avstängd PD4      (vänster Bak)



}else if (vansterMask == (0b00000100)){ // Höger fram, vänster bak

          OCR3A = 12;                //ändrar hastighet på hjulen
          OCR3B = 12;
          OCR1A = 12;
          OCR1B = 12;

          TCCR1A = (0b00100010);      //PB4 på, Avstängd PB5      (vänster Fram)
          TCCR3A = (0b10000010);      //PB6 på, Avstängd PB7  (Höger bak)



}else{  //standby, allt Avstängt

          TCCR3A = (0b00000010);
          TCCR1A = (0b00000010);

}

if(tutaMask == (0b00000010)){         // sätter igång PB1 om insignalen är på och lampan är av
          PORTB |= 0b00000010;


}else { //stänger av PB1 om insignalen är på och lampan är på
          PORTB &= 0b11111101;
}
```

```
                if(lamporMask == (0b00000001)  && lampswitch == 0 && lamp == 0){ // sätter igång PB0 om insignalen är
på annars stänger av
                        PORTB |= 0b00000001;
                        lampswitch = 1;
                        lamp = 1;

                }else if(lamporMask == (0b00000001) && lampswitch == 0 && lamp == 1){
                        PORTB &= 0b11111110;
                        lampswitch = 1;
                        lamp = 0;
                }
                if(lamporMask == (0b00000000)){ //väntar tills knappen är av
                        lampswitch = 0;
                }
    }
}

void timer_init(){
        TCCR3A = (0b00000010);              //PB6 och PB7 avstängd
        TCCR3B = (0b00011101);
        TCCR1A = (0b00000010);              //PB4 och PB5 avstängd
        TCCR1B = (0b00011101);

        OCR3A = 12;                 //Puls  between 1ms and 2ms
        OCR3B = 12;                 //Puls
        OCR1A = 12;                 //Puls
        OCR1B = 12;                 //Puls
        ICR3 = 40;                  //Period 20ms
        ICR1 = 35;                  //Period

}

void uart_init(){

        UCSR0B |= (0b00010000);
        UBRR0 = 207;

}

uint8_t uart_recieve(){
        while (!(UCSR0A & (1 << (RXC0)))){ }

        return UDR0;
}
```

Kopplingsschema för kontrollen:



Kod för kontrollen:

```c
#include <avr/io.h>

void adc_init();
uint16_t adc_read();
void uart_init();
void uart_transmit(uint8_t);

uint16_t tempDataF;
uint16_t tempDataH;
uint8_t packet;

int main(void)
{
    DDRA = 0x00;    //sätter A till inport
```

```
DDRB = 0x01;    //sätter första på B till utport
PORTB = 0x01;   //sätter en 1 till TX

uart_init();
adc_init();

while (1)
{

    uart_transmit(packet);

    packet = 0;  // data att sända, plusas up och skickas sedan i början av while loopen

    /* Protocol
    0010 0000 == Fram
    0001 0000 == Bak
    0000 1000 == Höger
    0000 0100 == Vänster
    0000 0010 == Tuta
    0000 0001 == Lampor
    */

    ADMUX &= (0b11111110);            //port 0 fram & bak
    tempDataF = adc_read();
    if(tempDataF > 900){    //ger värde 32 om joystick är fram
        packet += 32;

        }else if(tempDataF < 100){ //ger värde 16 om joystick är bak
        packet += 16;

    }

    ADMUX |= (0b01000001);            //port 1 höger & vänster
    tempDataH = adc_read();
    if(tempDataH > 900){    //ger värde 4 om joystick är vänster
        packet += 4;

    }else if(tempDataH < 100){    //ger värde 8 om joystick är höger
        packet += 8;

    }

    uint8_t joyMask = (0b00000100);   // mask för port PA2
    uint8_t buttonMask = (0b00001000);  // mask för port PA3

    uint8_t joyButton = PINA & joyMask;  // gör om input för portA med hjälp av mask
    uint8_t buttonButton = PINA & buttonMask;  // gör om input för portA med hjälp av mask

    if( joyButton == (0b00000100)){            //port 2 tuta, ger värde 2
        packet += 2;
    }
    if( buttonButton == (0b00001000)){            //port 3 lampor, ger värde 1
        packet += 1;
```

```
        }
    }
}


void adc_init(){

        ADMUX |= (0b01000000);            //(page 331)
        ADCSRA |= (0b10000111);            //(page 334)
}

uint16_t adc_read(){

        ADCSRA |= (1 << ADSC);            // stänger av om den inte är använd

        while (ADCSRA & (1 << (ADSC))){        // sätt igång när den byter värde
        }

        return ADC;
}

void uart_init(){

    UCSR0B |= (0b00001000);
    UBRR0 = 207;

}

void uart_transmit(uint8_t packet){

    while (!(UCSR0A & (1 << UDRE0))){    }   // om den har något i sig vänta till man kan sända

    UDR0 = packet;
}
```