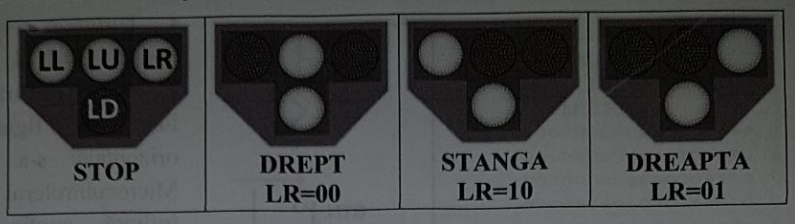
Semafor tramvai – examen 23 iunie 2019

Subiecte.

Să se proiecteze un microsistem bazat pe microcontrolerul ATmega16 care controlează un semafor pentru tramvai.



Semaforul este compus din 4 LED-uri notate **LD** (LED Down), **LU** (LED Up), **LR** (LED Rigth) şi **LL** (LED Left). Cele 4 LED-uri sunt aprinse sau stinse pentru a forma configuraţiile din figură.

Semaforul funcţionează ciclic. **Un ciclu** este alcătuit din două părţi: prima parte durează 30 de secunde iar a doua 45 de secunde.

1. În prima parte a ciclului se permite trecerea tramvaiului şi se indică direcţia pe care se face trecerea. Sunt posibile direcţiile DREPT, STANGA şi DREAPTA. Direcţiile vor fi afişate conform figurilor prezentate mai sus.

   Microcontrolerul determină configuraţia care se va afişa prin citirea a două semnale: L şi R. Aceste semnale sunt intrări pentru microcontroler. Semnificaţia semnalelor L şi R este prezentată în figurile de mai sus.

2. În a doua parte a ciclului se afişează configuraţia STOP indiferent de valorile lui L şi R.

3. Intensitatea luminoasă a LED-urilor va depinde de intensitatea luminii solare. Intensitatea luminii solare este furnizată de un senzor cu trei ieşiri notate s_2 s_1 s_0. Cele 3 ieşiri formează numărul S= s_2 s_1 s_0. Când intensitatea luminii solare este maximă S="111" iar când este minimă S="000".

   Intensitatea luminoasă a LED-urilor (**ILED**) variază în funcţie de S în intervalul [30% Lmax, Lmax], Lmax fiind luminozitatea maximă a unui LED. Luminozitatea maximă este aceeaşi pentru toate LED-urile.

   ILED depinde de S liniar: Pentru S="000" ILED=30%Lmax, pentru S="001" ILED=40%Lmax, ..., pentru S="111" ILED=Lmax.

Specificaţii şi cerinţe obligatorii:

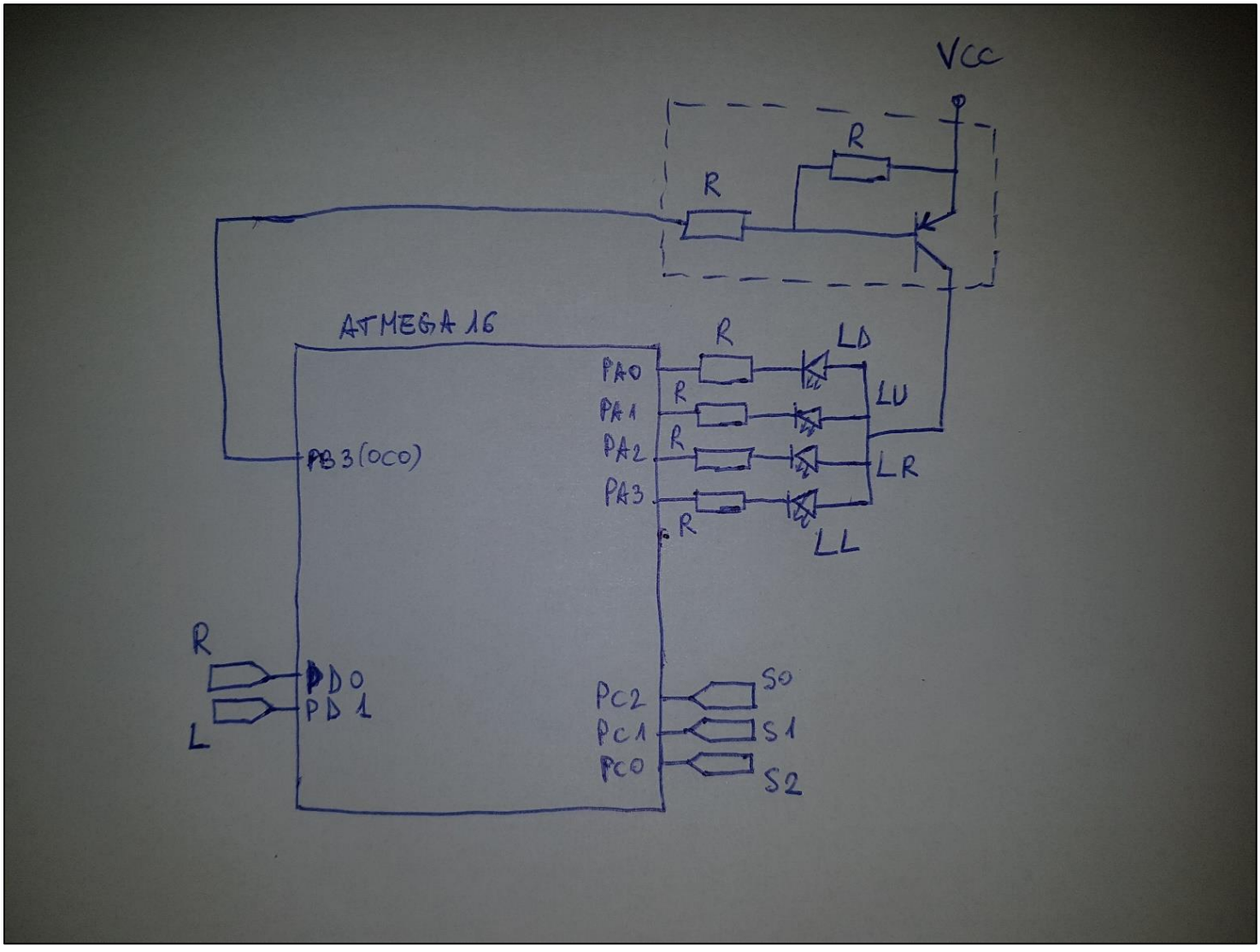
1. Ceasului microcontrolerului se generează cu un cristal de **8,192 MHz**.
2. **Nu se admit** întârzieri implementate cu bucle soft de tip for(i=0;i<DELAY;i++){}
3. Se va folosi **un singur timer de 8 biţi**. Acest timer va fi folosit atât pentru măsurarea timpului cât şi pentru intensitatea LED-urilor. Tcycle trebuie sa fie adecvat pentru PWM (5 – 15 ms);
4. Programul trebuie să aibă o singură funcţie main(), eventual alte funcţii, o singură bucla principală while() şi ISR-uri (dacă este cazul). Orice funcţie în afară de *main* trebuie sa fie apelată direct sau indirect din *main* sau din ISR.

Implementarea care nu respecta cerinţele de mai sus **nu se punctează**.

Se cere:

1. Schema de conectarea a semnalelor L, R, s_2, s_1, s_0 şi a LED-urilor LD, LU, LR şi LL la pinii ATMega16 şi configurarea porturilor. **1 punct**
2. Stabilirea modului de funcţionare a timerului şi iniţializarea timerului. Alegerea şi setările timerului (numărul timerului, modul, N, *p*, ieşirea OC, etc) se justifică. Trebuie să se facă o singură setare de mod. **3 puncte** (1,5 puncte dacă timerul se poate folosi numai pentru măsurarea timpului, 1,5 puncte dacă timerul se poate folosi numai pentru ILED, 3 puncte dacă se poate folosi pentru ambele)
3. Codul C pentru gestiunea ciclului şi pentru afişarea configuraţiei semaforului. **3 puncte** (se acordă dacă timerul se poate folosi pentru măsurarea timpului).
4. Codul C pentru controlul ILED în funcţie de S. **2 puncte** (se acordă dacă timerul se poate folosi pentru ILED)

Schema:


Vcc
R
R
ATMEGA 16
R
LD
PA0
R
PA1
LU
R
PA2
LR
PA3
R
LL
PB3(OC0)
R
PD0
PD1
L
PC2
PC1
PC0
S0
S1
S2

Codul:

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

#define clrbit(var,bit) var &= ~(1<<bit)
#define setbit(var,bit) var |= 1<<bit

int main(){

   unsigned char cycles = 0; //aici dupa 125 de cicluri, a trecut o secunda
   unsigned char inRL = 0; //pentru afisarea sensului de mers al tramvaiului
   unsigned char seconds = 0; //numarul de secunde
   unsigned char inSensor = 0; //senzorul de lumina

   //bit 7 - FOC0 nu se programeaza, ramane 0
   //bits 3,6 - WGM01 si WGM00 = 11, adica mod PWM
   //bits 5:4 - COM01 si COM00 = 11, set OC0 on compare match, clear OC0 at
BOTTOM
   //bits 2:0 - CS02, CS01 si CS00 = 100, adica clk/256 (de la prescaler)
   //8.192MHz / 256 / 256 = 125
   TCCR0 = 0b01111100;

   OCR0 = 0; //LED-urile sunt la inceput stinse

   DDRA = 0x0f; //PA3:0 sunt iesiri
   DDRB = 0x08; //PB3 (OC0) este iesire
   DDRC = 0; //portul C este configurat ca port de intrare
   DDRD = 0; //portul D este configurat ca port de intrare

   PORTA = 0x0f;
   PORTB = 0x08;

   while(1){

      if (TIFR & 1<<TOV0){
         TIFR |= 1<<TOV0; //clear TOV0

         cycles++;

         if (cycles == 125){
            cycles = 0;
            seconds++;
         }
      }

      if(seconds > 74){
         seconds = 0;
      }

      if (seconds < 30){

         inRL = PIND & 0b00000011;
```

```
         if(inRL == 0){
            clrbit(PORTA, 0);
            clrbit(PORTA, 1);
            setbit(PORTA, 2);
            setbit(PORTA, 3);
         }

         if(inRL == 1){
            clrbit(PORTA, 0);
            clrbit(PORTA, 2);
            setbit(PORTA, 1);
            setbit(PORTA, 3);
         }

         if(inRL == 2){
            clrbit(PORTA, 0);
            clrbit(PORTA, 3);
            setbit(PORTA, 1);
            setbit(PORTA, 2);
         }
      }//end if (seconds < 30)


      //75 sec = 30 sec (prima parte) + 45 sec (a doua parte)
      if (seconds > 29 && seconds < 75){
         clrbit(PORTA, 1);
         clrbit(PORTA, 2);
         clrbit(PORTA, 3);
         setbit(PORTA, 0);
      }//end if (seconds > 29 && seconds < 75)


      inSensor = 0b00000111 & PINC;
      OCR0 = (inSensor * 10 + 30) * 255 / 100; //line by Andrei Vinatoru

   }//end while
}//end main
```