UNIVERSITATEA NAȚIONALĂ DE ȘTIINȚĂ ȘI TEHNOLOGIE POLITEHNICA BUCUREȘTI

Facultatea de Electronică, Telecomunicații și Tehnologia Informație

PROIECT MICROCONTROLERE

Semafor Inteligent cu Prioritizare a Vehiculelor de Urgență

Student: Tabacu Dimitrie

Grupa: 422E

București 2025

Cuprins

Lista figurilor

Introducere

Obiectivul general al proiectului este implementarea unui **sistem de semafor inteligent** prin intermediul unui **proces secvențial**, capabil să regleze traficul rutier în funcție de fluxul de mașini și să acorde prioritate vehiculelor de urgență (ex: ambulanță, pompieri, poliție).

Un proces secvențial poate fi reprezentat printr-un automat finit de stări (FSM), unde tranzițiile între stări sunt controlate de evenimente externe (senzori de trafic, semnal de urgență) sau de un cronometru intern.

Pentru implementarea semaforului cu FSM, sunt necesare următoarele elemente:

- un set finit de stări, inclusiv o stare inițială (ex: stare normală – verde)

- un set finit de intrări, provenite de la senzori sau temporizator

- un set finit de tranziții, care definesc modul în care semaforul trece de la o stare la alta în funcție de condițiile detectate.

Funcționalitatea semaforului:

- Comutarea automată a culorilor semaforului între verde, galben și roșu pe baza unui cronometru intern.

- Detectarea traficului cu senzori (presiune sau infraroșu), pentru a prelungi verdele în caz de aglomerație sau a menține roșu în lipsa vehiculelor.

- Prioritizarea vehiculelor de urgență, prin activarea unei logici combinaționale care comută instantaneu semaforul pe verde pentru a permite trecerea rapidă a vehiculului.

- Revenirea automată la funcționarea standard după ce vehiculul de urgență a trecut.

Proiectul are la bază programarea unui microcontroler de tip ATmega164P, prin implementarea codului în limbaj C cu ajutorul aplicației CodeVisionAVR. Procesul secvențial va fi observat în AVR Studio, unde folosesc porturi de intrare (switch-uri) și de ieșire (LED-uri).

Capitolul 1
Date inițiale de proiectare

Proiectul simuleaza comportamentul real într-o intersecție cu un singur flux de trafic. Sistemul va răspunde la un set fix de condiții de intrare, simulate prin butoane (switch-uri), și va afișa rezultatele vizual cu ajutorul LED-urilor.

Exista la baza 4 scenarii de functionare:

- Idle – starea de asteptare(sistem inactiv), unde semaforul sta pe rosu

- Modul de funcționare standard – este activat după inițializarea sistemului prin apăsarea butonului START. În acest mod, semaforul comută în mod ciclic între stările verde (10s), galben (3s) și roșu (10s), indiferent de prezența sau absența traficului.

- Modul de funcționare adaptivă la trafic – este activat atunci când, în timpul funcționării normale, sistemul detectează trafic (prin apăsarea butonului TRAFIC). În acest caz, durata semnalului verde este prelungită, iar comutarea pe roșu este amânată pentru a permite fluidizarea traficului.

- Modul de prioritate pentru urgență – este activat în orice moment prin apăsarea butonului URGENȚĂ. Semaforul comută imediat pe verde pentru a permite trecerea vehiculului de urgență, având prioritate față de orice alt scenariu activ. După finalizarea urgenței, sistemul revine la modul de funcționare anterior.

Prin aceste patru moduri de funcționare, semaforul inteligent oferă o soluție pentru gestionarea traficului într-o intersecție cu o singură stradă, adaptându-se automat la prezența vehiculelor și la intervențiile urgente. Implementarea logicii în stil FSM, combinată cu utilizarea întreruperilor (Timer0 Overflow), permite controlul eficient al sistemului în timp real.

Capitolul 2
Descrierea funcționalității semaforului

Pentru implementarea semaforului inteligent, am utilizat microcontrolerul ATmega164P, care permite atât codului în limbaj C și simularea funcționării în AVR Studio, utilizând componente virtuale precum LED-uri și butoane.

Microcontrolerul este conectat la o sursă de alimentare, esențială pentru funcționarea în regim real. De asemenea, LED-urile și butoanele (switch-urile) utilizate în proiect sunt alimentate din aceeași sursă, conform conexiunilor standard pentru aplicații embedded.

Descriere funcțională a componentelor

Sursa de alimentare furnizează tensiunea necesară funcționării întregului sistem, asigurând alimentarea atât a microcontrolerului, cât și a butoanelor și LED-urilor.

Senzorul de trafic are rolul de a simula prezența unui flux de vehicule. Atunci când acesta este activat, sistemul detectează existența traficului și prelungește automat durata semnalului verde pentru a permite trecerea mai multor autovehicule.

Senzorul de urgență simulează semnalul emis de un vehicul prioritar, cum ar fi o ambulanță, o mașină de pompieri sau una de poliție. La activarea acestui senzor, sistemul comută imediat semaforul pe verde, oferind astfel prioritate vehiculului de urgență.

Blocul de condiționare a semnalului are funcția de a prelua semnalele de la senzori și butoane, de a le filtra și de a le transmite sub formă de impulsuri clare către microcontroler.

Butoanele START și RESET permit interacțiunea directă a utilizatorului cu sistemul.

Microcontrolerul ATmega164P reprezintă componenta centrală a sistemului. Acesta primește semnalele de intrare de la senzori și butoane, procesează aceste informații și controlează comportamentul semaforului conform unui algoritm determinat de un automat finit de stări (FSM).

Ieșirile sistemului sunt reprezentate de LED-uri, care oferă o reprezentare vizuală clară a stării curente a semaforului și a contextului de funcționare. Astfel:

LED7 indică activarea culorii roșu,

LED6 indică galben,

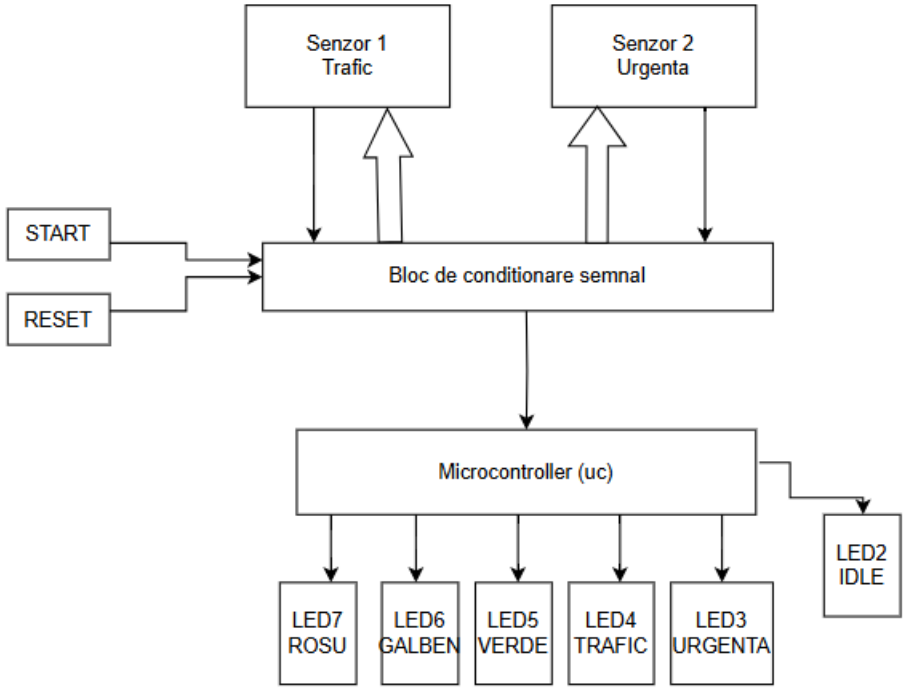
LED5 indică verde,

LED4 se aprinde când este detectat trafic,

LED3 semnalizează o urgență activă,

LED2 indică faptul că sistemul se află în starea de așteptare (IDLE).

Aceste ieșiri vizuale permit monitorizarea în timp real a comportamentului sistemului și oferă un mod intuitiv de validare a funcționării corecte.

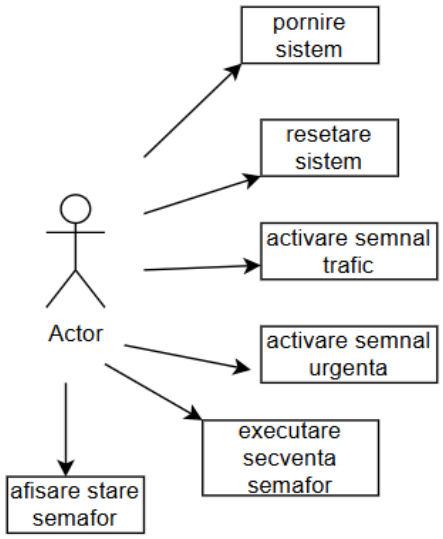


2.1Schema bloc a sistemului

Pentru a implementa graful procesului secvențial și ulterior codul în limbajul C, am realizat o diagramă care sintetizează funcționalitatea completă a sistemului de semafor inteligent. Această diagramă reprezintă cazurile de utilizare relevante pentru utilizator(actor), în funcție de acțiunile declanșate prin apăsarea butoanelor sau activarea senzorilor.

În această schemă, microcontrolerul ATmega164P are un rol central, fiind responsabil de citirea și interpretarea semnalelor de intrare provenite de la butoane și senzori. Acesta procesează

intrările, actualizează stările interne ale FSM și decide comportamentul semaforului. Microcontrolerul controlează LED-urile de ieșire, oferind feedback vizual în timp real asupra stării sistemului: culoarea semaforului activă, semnalizarea traficului detectat, urgența sau starea IDLE.



2.2. Diagrama cazurilor de utilizare a sistemului

Capitolul 3
Descrierea funcționalității porturilor de intrare și ieșire

Portul principal de intrare utilizat în proiect este PORTD, configurat pentru a prelua semnalele de la butonul START, butonul de RESET, precum și de la simularile de trafic și urgență. În total, sunt utilizate 4 dintre cei 8 pini disponibili, fiecare conectat la un switch cu revenire (activ pe 0). Aceste butoane sunt esențiale pentru controlul funcționării semaforului în cadrul automatului finit de stări (FSM).

Configurația folosită este următoarea:

Intrari

SW7	SW6	SW5	SW4	SW3	SW2	SW1	SW0
START	TRAFIC	URGENTA	RESET/IDLE	-	-	-	-

3.1. Port intrare D

Aceste intrări sunt citite în cadrul întreruperii Timer0 și determină tranzițiile între stările semaforului. Restul pinilor de pe PORTD nu sunt utilizați în această implementare.

SW7-SW3 reprezintă butoanele de comandă active în 0 cu revenire (apăsat şi eliberat):

Start- Semaforul este pornit

Trafic- atunci cand este trafic, butonul se apasa pentru a se mentine culoarea verde , mai mult timp (evitare aglomeratiei)

Urgenta- atunci cand se apropie un vehicul de urgenta, semaforul este automat comutat pe verde

Reset/idle- initializare- semaforul revine la culoarea rosie atunci cand nu sunt vehicule

Modul de lucru:

-mentinerea culorii verzi in scopul evitarii ambuteiajelor

-schimbarea de urgenta in culoarea verde pentru a permite trecerea vehiculelor in misiune

Modul de operare:

Se apasa Start- semaforul va sta 10 secunde pe culoare rosie, 3 secunde pe galben, 10 secunde pe verde.

Se apasa trafic- s a detectat faptul ca fluxul de trafic a deposit un prag critic, se prelungeste culoarea verde cu 5 secunde

Se apasa urgenta- Un semnal radio emis de un vehicul de urgență activează un circuit combinational care schimbă semaforul pe verde instantaneu.

Se apasa Idle- semaforul ramane pe culoarea rosie pentru ca senzorii nu detecteaza prezenta vehiculelor

Din modul Idle se poate trece in modul urgenta prin apasarea butonului Urgenta.

Tabela de adevăr pentru CLC şi determinarea măştilor necesare

Circuitul logic combinational (CLC) va fi folosit pentru semnalizarea traficului intens si trecerea vehiculelor de urgenta

Intrare	Start	Urgenta	Trafic	Reset
Masca	0	1	1	0
Index	0	Urgenta	Trafic	0

MASCA = 0x60 (binar: 0110 0000)

Intrare	Start	Urgenta	Trafic	Reset
Masca	0	1	0	0
Index	0	Urgenta	0	0

&

Intrare	Start	Urgenta	Trafic	Reset
Masca	0	0	1	0
Index	0	0	Trafic	0

Deplasările nu sunt necesare, astfel că se poate calcula direct indexul pentru tabela de adevăr:
INDEX = TMP_Urgenta | TMP_Trafic

TMP_Urgenta	0	Urgenta	0	0
TMP_Trafic	0	0	Trafic	0
Index	0	Urgenta	Trafic	0

Urgenta	Trafic
0	0
0	1
1	0
1	1

şi ţinând cont că senzorul este activ în 1 =>

Urgenta	Trafic	Semnificatie
0	0	----
0	1	Trafic crescut
1	0	Urgenta active
1	1	Trafic+Urgenta

Pe baza semnificatiei celor 4 combinatii de valori ale biitlor de intrare, se pot determina iesirile astfel

TRA (SW6)	**URG (SW5)**	**Semnificație**	**LED4 (TRAFIC)**	**LED3 (URGENȚĂ)**	**Valoare Hex**
0	0	Trafic **și** urgență active	1	1	0x03
0	1	Doar trafic activ	1	0	0x02
1	0	Doar urgență activă	0	1	0x01
1	1	Nici trafic, nici urgență (idle)	0	0	0x00

Tabela de adevăr a CLC-ului va fi definită ca vector TAB cu 4 elemente, după cum urmează:

```
char in;
char iesire;
char TAB[] = {0x03, 0x02, 0x01, 0x00}; // LED4 | LED3

void CLC(void) {
   char TRA, URG, index;

   TRA = in & 0x40; // bit 6 – TRAFIC
   URG = in & 0x20; // bit 5 – URGENȚĂ

   index = (TRA >> 5) | (URG >> 5);
   iesire = TAB[index];
}
```

Portul de iesire B

Pentru semnalizarea stării curente a semaforului și a condițiilor speciale, proiectul utilizează PORTB ca port de ieșire. Acesta controlează un total de 6 LED-uri conectate la ieșirile microcontrolerului. LED-urile sunt active în 0 (se aprind când ieșirea este 0 logic) și oferă o reprezentare vizuală în timp real a comportamentului sistemului.

Iesiri

Led7	Led6	Led5	Led4	Led3	Led2	Led1
Rosu	Verde	Galben	Trafic	Urgenta	Idle	**---**

3.2.Port de ieșire B

LED-urile folosite:

- LED7 = ROȘU
- LED6 = VERDE
- LED5 = GALBEN
- LED4 = TRAFIC
- LED3 = URGENȚĂ
- LED2 = IDLE

Funcționarea sistemului poate fi descrisă printr-o succesiune de stări și tranziții, în care portul de intrare D este folosit pentru comenzi (buton START, TRAFIC, URGENȚĂ, RESET), iar portul de ieșire B este utilizat pentru semnalizarea stării curente prin LED-uri. Iată fluxul logic de funcționare:

1. Stare inițială – IDLE
   La pornirea sistemului, automatul finit de stări se află în starea IDLE, indicată vizual prin aprinderea LED2 și LED7. În această stare, sistemul este inactiv și așteaptă activarea prin apăsarea butonului START (SW7).

2. Activare sistem – tranziție din IDLE
   Apăsarea butonului SW7 declanșează tranziția din starea IDLE către starea de funcționare normală. Sistemul începe secvența standard a semaforului:

   LED2 se stinge (IDLE dezactivat)

   LED5 (verde) se aprinde

3. Funcționare standard
   Sistemul intră într-un ciclu automat de comutare a semaforului, controlat de Timer0:

   Verde – 10 secunde (LED5)

   Galben – 3 secunde (LED6)

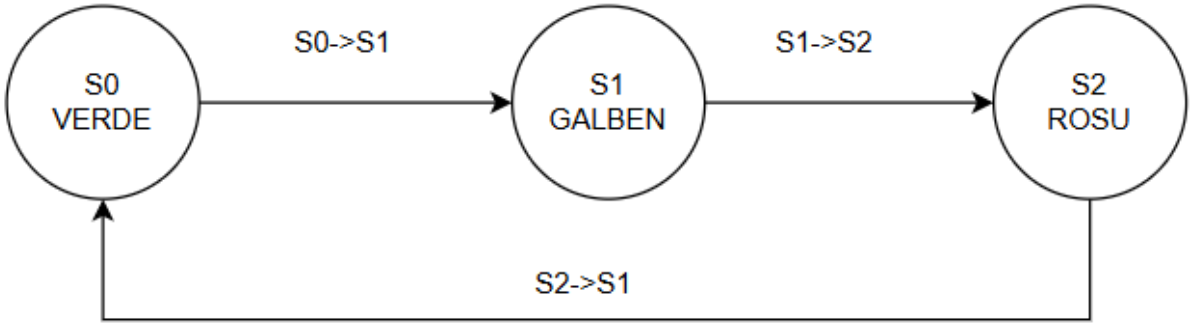
   Roșu – 10 secunde (LED7)
   Acest ciclu se repetă continuu, în absența unor condiții externe.

4. Detectarea traficului
   În orice moment al funcționării standard, apăsarea butonului TRAFIC (SW6) semnalează prezența unui vehicul. În această situație, sistemul prelungește durata semnalului verde, întârzie tranziția către galben și aprinde LED4 .

5. Activarea urgenței
   Dacă se apasă butonul URGENȚĂ (SW5), semaforul comută imediat pe verde (LED5), indiferent de starea curentă. LED3 se aprinde pentru a indica faptul că o urgență este activă. După finalizarea urgenței (eliberarea butonului), sistemul revine la starea de funcționare anterioară.

6. Resetare sistem – revenire în IDLE
   În orice moment, apăsarea butonului RESET (SW4) oprește funcționarea sistemului și îl readuce în starea IDLE. În acest caz, toate LED-urile se sting, cu excepția LED2 și LED7.

Graf CLS



3.3. graf CLS

Acest grafic descrie ciclul standard al semaforului atunci când funcționează fără intervenții externe. Este un sistem ciclic sincronizat prin temporizator (Timer0). Toate tranzițiile se bazează pe durate fixe de timp.

Stare	LED activ	Descriere
S0	LED5 = VERDE	Permite trecerea vehiculelor pe Strada A
S1	LED6 = GALBEN	Semnal de avertizare — urmează roșu
S2	LED7 = ROȘU	Stop complet. Rămâne activ dacă sistemul revine în IDLE

Graful CLS descrie ciclul automat de funcționare al semaforului, care alternează între verde, galben și roșu în funcție de temporizator. Sistemul pornește cu verde (LED5) unde sta 10 secunde, trece în galben (LED6) unde sta 3 secunde, apoi în roșu (LED7), unde sta 10 secunde, pentru oprire. După o perioadă fixă, ciclul se repetă.

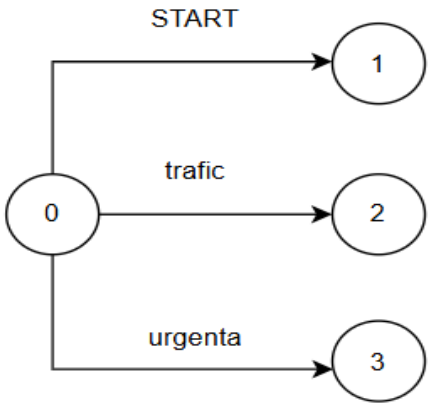
Capitolul 4. Codul sursă

4.1 Proiectarea codului sursă corespunzător sistemului

Funcționarea semaforului inteligent este modelată ca un proces secvențial (PS), implementat sub forma unui automat finit de stări (FSM). Un proces secvențial este caracterizat de faptul că:

- sistemul are memorie internă (o stare curentă),
- ieșirile și tranzițiile depind de această stare și de intrările primite,
- evoluția în timp se face în pași bine definiți

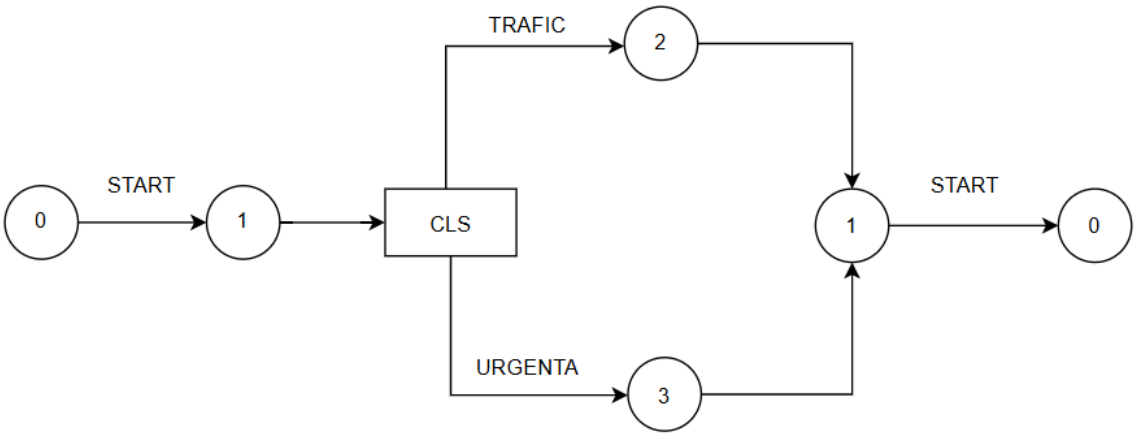
Consider o stare inițială (IDLE). Din această stare, este posibilă trecerea în orice altă stare (TRAFIC, URGENȚĂ) prin apăsarea switch-ului START care porneste sistemul.



4.1.Mod programare semafor

Conform cerinţei, programarea semaforului se face astfel:
– Trecerea în modul de programare prin apăsarea butonului START (SW7)
– Comutarea în secvenţa pentru regim de flux de trafic ridicat prin apăsarea butonului TRAFIC (SW5)
– Comutarea în secvenţa pentru regim de urgenţă prin apăsarea butonului URGENŢĂ (SW6)
– Ieşirea din modul de programare (revenirea în starea IDLE) prin apăsarea butonului START (SW7)

În plus, ţinând cont de faptul că modificarea culorilor se realizează prin intermediul circuitului logic secvenţial (CLS), graful asociat programării semaforului devine:



4.2.Graf varianta 1

Stările au următoarele semnificaţii:

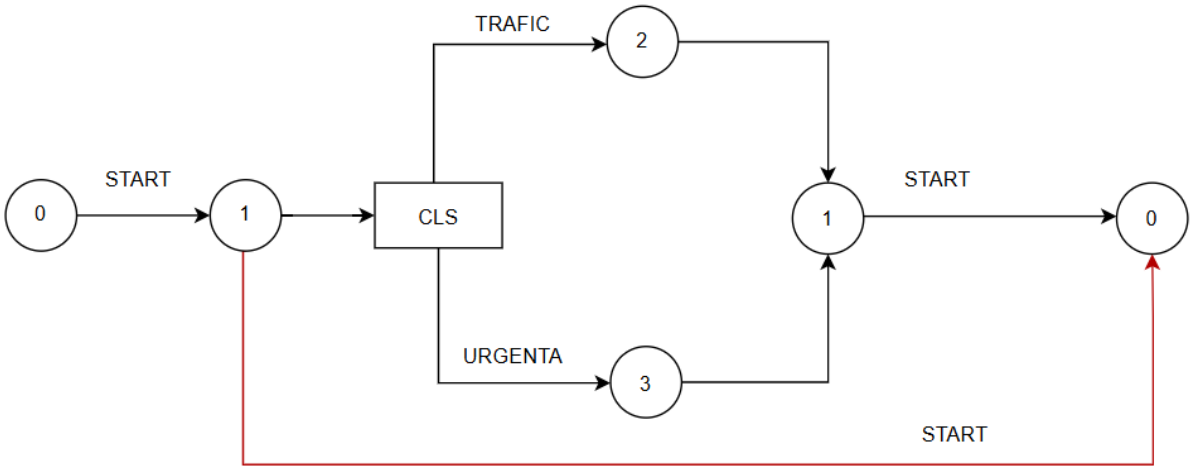
0 – Starea iniţială(Idle)

1 – Modul de programare(start)

2 – regim de trafc

3 – regim de urgenta

Există posibilitatea ca, în urma apăsării butonului START, utilizatorul să renunţe la programarea semaforului, întrucât nu este nevoie să comute în unul din cele două regimuri disponibile, iar graful asociat programării devine:



4.3.Graf varianta 2

În momentul de faţă, graful ilustrează modalitatea de tranziţie dintr-o stare în alta, dar nu şi ieşirile corespunzătoare fiecărei stări. Astfel, se pot considera următoarele efecte:

Selecția modului de programare (prin apăsarea butonului START și trecerea din starea 0 în 1) are ca efect aprinderea LED5 (verde), indicând începutul secvenței normale a semaforului (verde-galben-roșu) stingerea LED2 și LED7, asociate stării IDLE

Sistemul este pregătit pentru a primi comenzi legate de TRAFIC sau URGENȚĂ

Activarea stării TRAFIC (starea 2) prin apăsarea butonului TRAFIC (SW5) are ca

efect: aprinderea LED4 (semnal trafic detectat), prelungirea duratei semnalului verde în FSM.

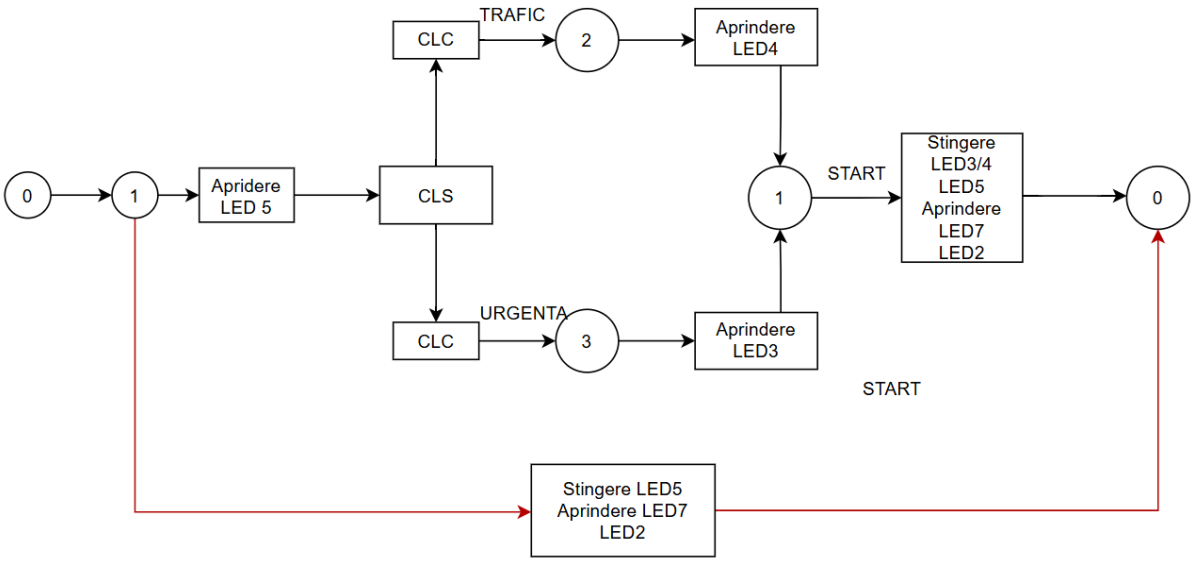
Activarea stării URGENȚĂ (starea 3) prin apăsarea butonului URGENȚĂ (SW6) are ca efect: aprinderea LED3 (prioritate vehicul de urgență), comutarea forțată a semaforului pe verde.

Revenirea în starea 0 (IDLE), fie direct din starea 1, fie după ieșirea din 2 sau 3, prin apăsarea din nou a butonului START (SW7), are ca efect stingerea tuturor LED-urilor, cu excepția:

LED7 (semafor roșu) – activ pentru siguranță

LED2 – care semnalizează revenirea în IDLE

Ţinând cont de aceste aspecte, graful devine:



4.4.Graf final

4.2. Implementarea codului sursă

```
#include <mega164a.h>
#include <delay.h>

#define LED_ROSU     PORTB.7
#define LED_GALBEN   PORTB.6
#define LED_VERDE    PORTB.5
#define LED_TRAFFIC  PORTB.4
#define LED_URGENTA  PORTB.3
#define LED_IDLE     PORTB.2

#define SW_START    ((PIND & (1 << 7)) == 0)
#define SW_TRAFFIC  ((PIND & (1 << 6)) == 0)
#define SW_URGENTA  ((PIND & (1 << 5)) == 0)
#define SW_IDLE     ((PIND & (1 << 4)) == 0)

#define T_VERDE       500
#define T_GALBEN      150
#define T_PRELUNGIT   250

#define IDLE          0
#define NORMAL        1
#define ASTEPT_VERDE  2
```

```
#define ASTEPT_GALBEN 3
#define URGENTA       4
#define TRAFIC        5
#define ASTEPT_TRAFIC_G 6

volatile unsigned char stare = IDLE;
volatile unsigned int timp = 0;

interrupt [TIM0_OVF] void timer0_ovf_isr(void)
{
   TCNT0 = 0x4E;
   timp++;
}

void semafor_idle() {
   LED_ROSU = 1;
   LED_GALBEN = 0;
   LED_VERDE = 0;
   LED_IDLE = 1;
   LED_TRAFFIC = 0;
   LED_URGENTA = 0;
}

void semafor_verde() {
   LED_ROSU = 0;
   LED_GALBEN = 0;
   LED_VERDE = 1;
```

```
    LED_IDLE = 0;
}

void semafor_galben() {
    LED_ROSU = 0;
    LED_GALBEN = 1;
    LED_VERDE = 0;
    LED_IDLE = 0;
}

void afiseaza_leduri() {
    LED_TRAFFIC = 0;
    LED_URGENTA = 0;
    LED_IDLE = 0;

    switch (stare) {
        case IDLE:
            semafor_idle();
            break;
        case NORMAL:
        case ASTEPT_VERDE:
            semafor_verde();
            break;
        case ASTEPT_GALBEN:
        case ASTEPT_TRAFIC_G:
            semafor_galben();
            break;
```

```
        case URGENTA:
            semafor_verde();
            LED_URGENTA = 1;
            break;
        case TRAFIC:
            semafor_verde();
            LED_TRAFFIC = 1;
            break;
        default:
            semafor_idle();
            break;
    }
}

void main(void)
{
    DDRB = 0xFF;
    PORTB = 0x00;

    DDRD = 0x00;
    PORTD = 0xFF;

    TCCR0A = 0x00;
    TCCR0B = (1 << CS02) | (1 << CS00);
    TCNT0 = 0x4E;
    TIMSK0 = (1 << TOIE0);
```

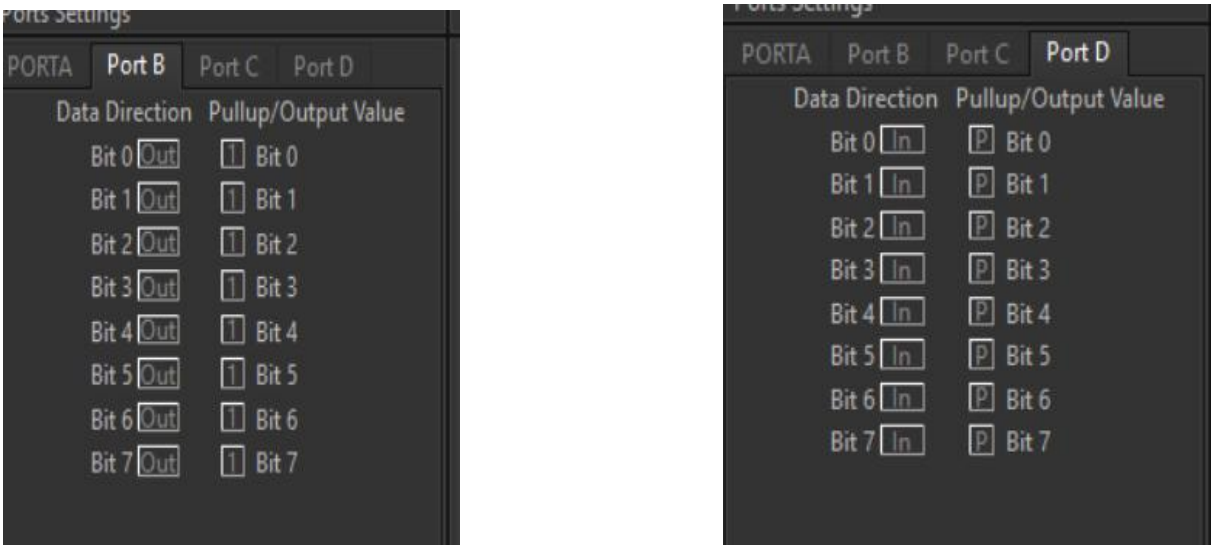
```
#asm("sei")

while(1)
{
   if(SW_START) {
      stare = IDLE;
      timp = 0;
   }

   if(stare == IDLE && SW_IDLE) {
      stare = NORMAL;
      timp = 0;
   }

   switch(stare) {
      case IDLE:
         break;

      case NORMAL:
         timp = 0;
         stare = ASTEPT_VERDE;
         break;

      case ASTEPT_VERDE:
         if(SW_URGENTA) {
            stare = URGENTA;
            timp = 0;
```

```
    } else if(SW_TRAFFIC) {
        stare = TRAFIC;
        timp = 0;
    } else if(timp >= T_VERDE) {
        stare = ASTEPT_GALBEN;
        timp = 0;
    }
    break;

case ASTEPT_GALBEN:
    if(timp >= T_GALBEN) {
        stare = IDLE;
        timp = 0;
    }
    break;

case URGENTA:
    if(timp >= T_VERDE) {
        stare = IDLE;
        timp = 0;
    }
    break;

case TRAFIC:
    if(timp >= (T_VERDE + T_PRELUNGIT)) {
        stare = ASTEPT_TRAFIC_G;
        timp = 0;
```

```
            }
            break;

        case ASTEPT_TRAFIC_G:
            if(timp >= T_GALBEN) {
                stare = IDLE;
                timp = 0;
            }
            break;

        default:
            stare = IDLE;
            timp = 0;
            break;
    }

    afiseaza_leduri();
  }
}
```

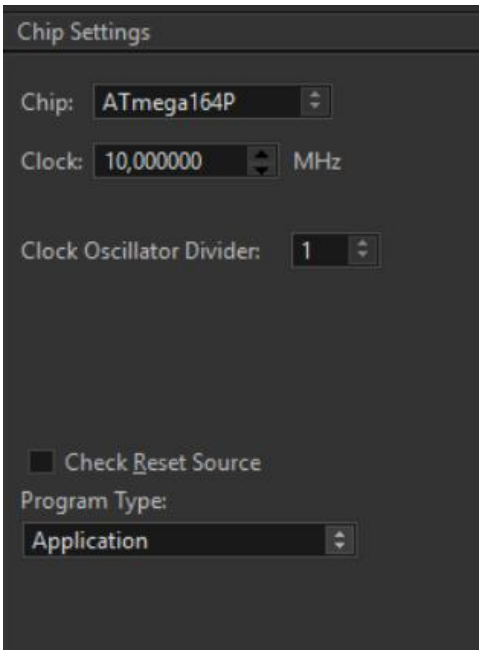
În implementarea codului am realizat setările specifice:

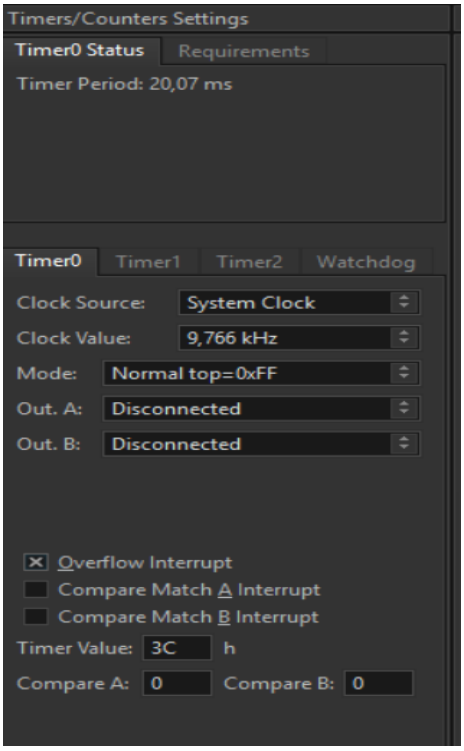
- Am configurat porturile de intrare și de iesire (PORTD-intrare PORTB-iesire)



Configurarea porturilor

setările pentru chip și pentru timer (pt a seta întreruperea la 20ms)



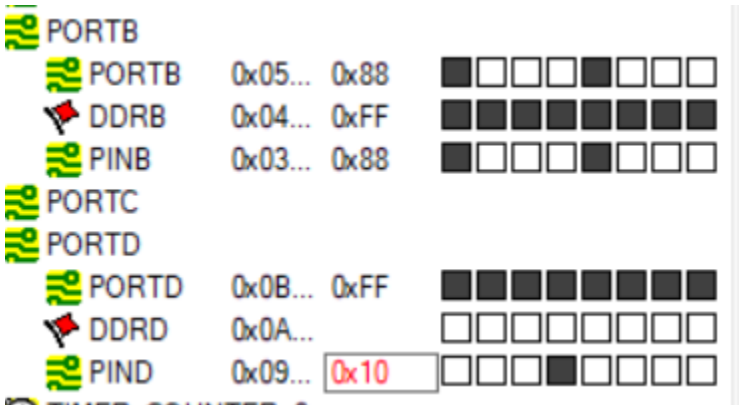


Configurarea chip și timer

Testare

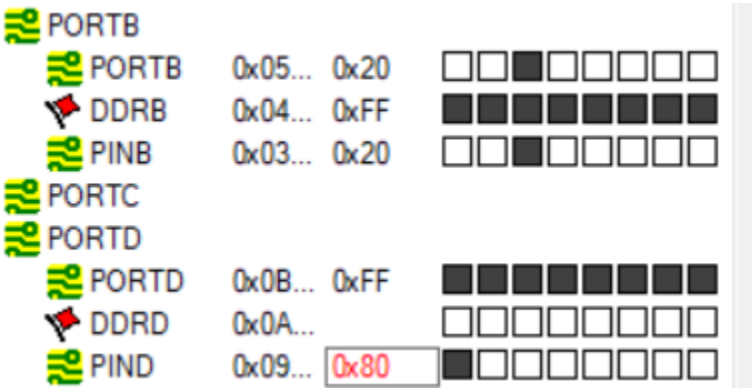
Verificarea funcționării corecte a sistemului semaforului inteligent s-a realizat utilizând interfața de butoane și LED-uri disponibilă în AVR Studio 4. Testarea a fost efectuată folosind butoanele SW7–SW4, corespunzătoare semnalelor de urgență și trafic. Funcționalitatea este vizibilă prin intermediul LED-urilor 7–2. Timpul este monitorizat cu ajutorul timer-ului intern (Timer0).

Configuratia pentru IDLE. Se apasa SW4 si se vor aprinde LED7 si LED2



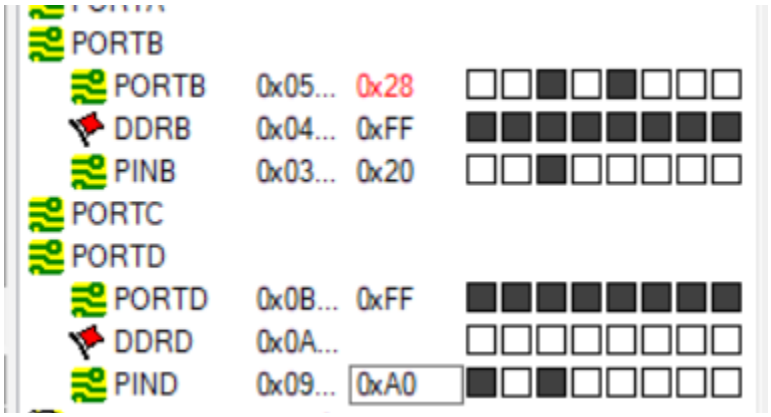
Configurația porturilor PORTB și PIND – starea idle

Configuratia pentru START. Se apasa SW7 si se va aprinde LED5



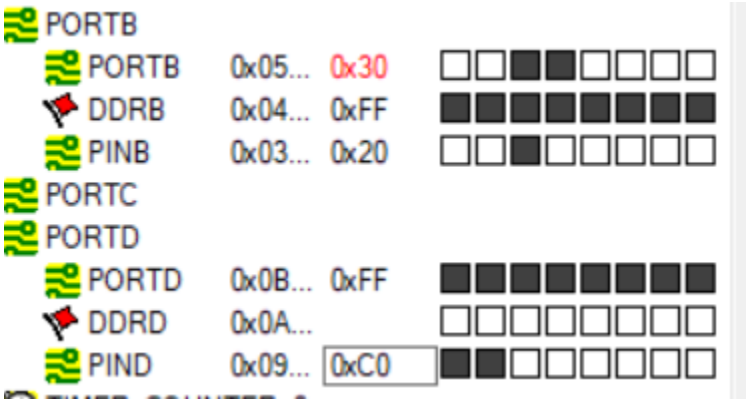
Configurația porturilor PORTB și PIND – starea START

Configuratia pentru URGENTA. Se apasa SW5 si se va aprinde LED4 si LED3



Configurația porturilor PORTB și PIND – starea URGENTA

Configuratia pentru TRAFIC. Se apasa SW6 si se va aprinde LED5 si LED4



Configurația porturilor PORTB și PIND – starea TRAFIC

Concluzii

Lucrul la acest proiect mi-a oferit o perspectivă clară asupra întregului proces, de la înțelegerea cerințelor până la testarea lor în simulator. Am construit sistemul pe două părți: un automat de stare care controlează ciclul normal al semaforului și o logică separată pentru situațiile de urgență si trafic.

Testele făcute în simulator au confirmat că semaforul funcționează cum trebuie: atunci când fluxul de trafic e crescut, verdele se prelungește, iar dacă nu e trafic, rămâne pe roșu.

În plus, sistemul răspunde rapid la semnalele de urgență, trecând imediat pe verde, apoi revine la modul normal.

Proiectul, finalizat și verificat în mediu de simulare, prezintă un potențial de aplicabilitate în sisteme reale. Posibile extensii includ implementarea hardware pentru gestionarea intersecțiilor, integrarea cu sisteme de monitorizare și control la distanță, precum și integrarea cu rețele de senzori pentru o semaforizare dinamică, bazată pe condițiile reale de trafic.