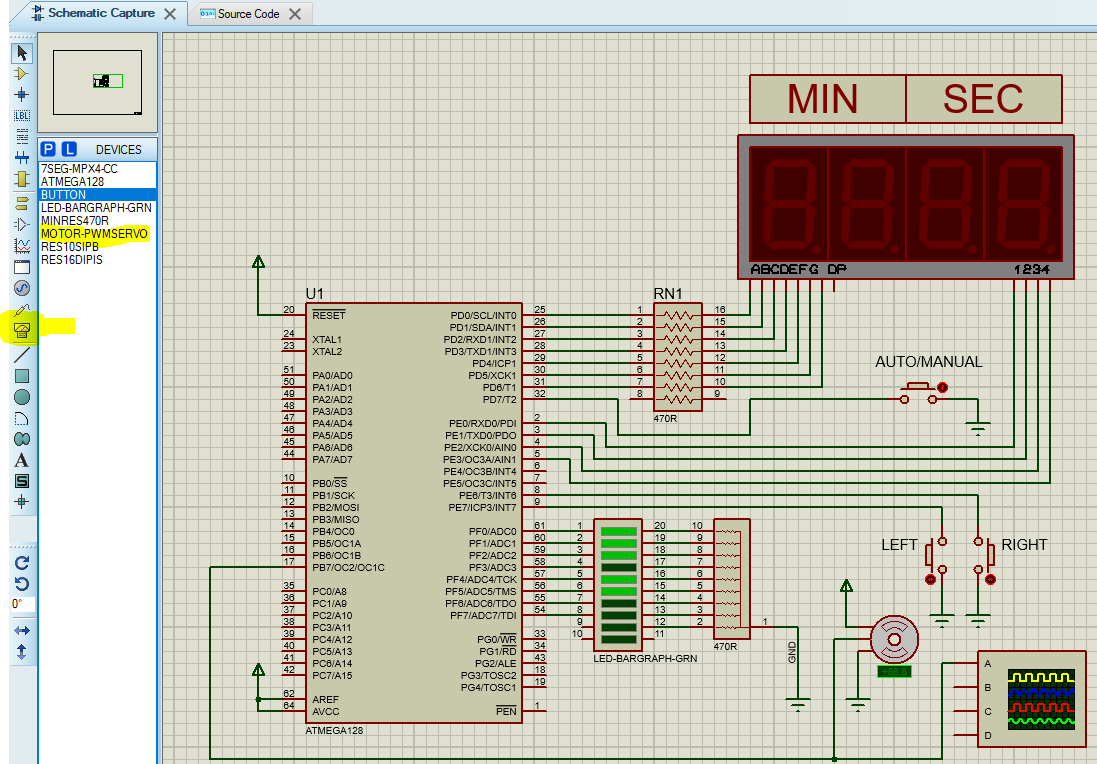
# Laboratorul nr. 4

## Obiective. În urma efectuării lucrării de laborator se învață:

* desenarea circuitelor utilizând programul Proteus;
* plasarea componentelor pe foaia de desenare;
* editarea componentelor (nume, valoare și simbol);
* trasarea liniilor de conexiuni;
* configurarea / accesarea portului D, E, F a microcontroler-lui ATmega128
* configurarea / utilizarea modulului numărător TCNT1 (modul normal de operare)
* configurarea / utilizarea modulului numărător TCNT2 (PWM – Fast / Phase correct)
* configurarea / utilizarea întreruperilor externe INT6 / INT7

### **Tema 1**

Să se deseneze cu ajutorul programului *Proteus* circuitul din fig. L1-1.

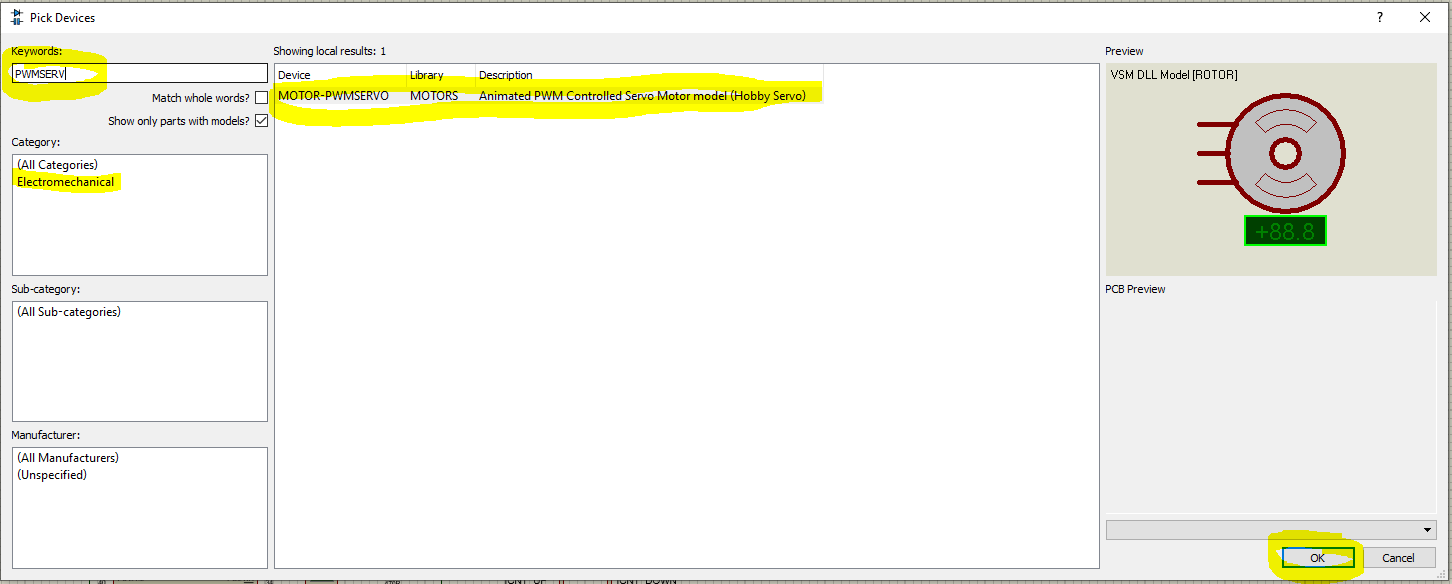


**Modul de lucru :**

**Pașii pentru lansarea programului *Proteus, crearea proiectului, plasarea componentelor pe planșa de lucru au fost prezentați in cadrul laboratorului cu numărul 1.***

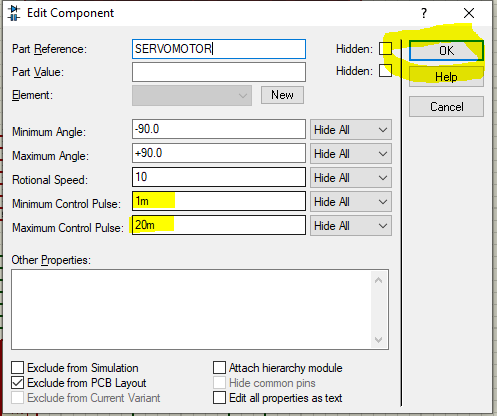
## Adăugarea servomotorului comandat PWM pe planșa de lucru:

ALT+L > Library > Pick Parts

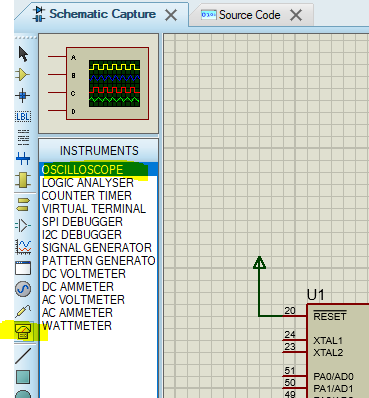


## Ajustarea parametrilor servomotorului:

Poziționam mouse-ul deasupra componentei pe care dorim sa o re-parametrizam, apăsam tastele CTRL+E.



Adăugarea osciloscopului pe planșa de lucru:



Următorii pașii necesari au fost descriși in cadrul laboratorului cu numărul 1

## Interconectarea componentelor.

## Rularea / oprirea aplicației de tip firmware:

## Modificarea codului sursa (Source Code)

Pornind de la următorul cod sursa rezolvați următoarele doua teme:

#include <inttypes.h>

#include <avr/io.h>

#include <avr/interrupt.h>

#include <avr/sleep.h>

#include <math.h>

// NECESARA PENTRU FUNCTIA \_delay\_ms(10);

#include <util/delay.h>

// TESTAREA VALORII BITULUI DE PE POZITIA [7..0]

// RETURNEAZA 0 DACA BITUL DE PE POZITIA INDICATA ESTE 0

// RETURNEAZA O VALOARE DIFERITA DE 0 DACA BITUL DE PE POZITIA INDICATA ESTE 1

#define TBIT(var,pos) ((var) &   (1<<(pos)))

// SETAREA IN 1 A BITULUI DE PE POZITIA [7..0]

#define SBIT(var,pos) ((var) |=  (1<<(pos)))

// SETAREA IN 0 A BITULUI DE PE POZITIA [7..0]

#define CBIT(var,pos) ((var) &= ~(1<<(pos)))

// NEGAREA BITULUI DE PE POZITIA [7..0]

#define NBIT(var,pos) ((var) ^=  (1<<(pos)))

// CORESPONDENTA SEGMENTELOR DE LED ACTIVE CORESPUNZATOARE NUMARULUI AFISAT

//                    SGFEDCBA

unsigned char hx[10]={0b10111111,  // 0

                      0b10000110,  // 1

                      0b11011011,  // 2

                      0b11001111,  // 3

                      0b11100110,  // 4

                      0b11101101,  // 5

                      0b11111101,  // 6

                      0b10000111,  // 7

                      0b11111111,  // 8

                      0b11101111}; // 9

// VARIABILA ALOCATA AFISORUL CU 4x7SEGMENTE PENTRU AFISAREA VALORII DORITE

// EX. VALOAREA INITIAL 0034

signed int val\_7seg=0034;

// DEFINIREA VARIABILEI GLOBALE TICK, VALOARE DE START 5

// VARIABILA TICK ESTE UTILIZATA PENTRU IMPLEMENTAREA NUMARATORULUI

// PRIN INTERMEDIUL EVENIMENTELOR GENERATE DE INTRERUPERILE EXTERNE INT6 / INT7

unsigned int tick=3590;

// RUTINA EXEMPLU PENTRU CREAREA UNEI INTARZIERI PROGRAMABILE

void DLY(volatile long D)

{

    while(D--);

}

// RUTINA RESPONSABILA CU AFISAREA VALORII STOCATE IN VAL\_7SEG PE AFISORUL

// 4x7SEGMENTE

void fUpdate7Seg()

{

  // VARIABIL DE TIP STATIC UTILIZATA PENTRU SELECTAREA DIGITULUI DE AFISAT 0..3

  static unsigned char ds=0;

  // SALVAREA VALORII DE AFISAT INTR-UN REGISTRU TEMPORAR (DUBLA BUFFERARE)

  // EVITAREA SCHIMBARII VALORII DE AFISAT IN TIMPUL PROCESULUI DE AFISARE

  unsigned int tmp=val\_7seg;

  // ALOCARE A 4 ZONE DE MEMORIE NECESARE CELOR PATRU DIGITI AI AFISORULUI 4x7SEG

  unsigned char dg[4]={5,6,7,8};

  // RESETAREA VALORII DIGITULUI CURENT (TOATE SEGMENTELE DE LEDURI SUNT STINSE)

  PORTD=0b00000000;

  // CONVERSIA INTEGER LA FORMATUL 4x7SEG

  dg[3]=tmp%10;

  tmp/=10;

  dg[2]=tmp%10;

  tmp/=10;

  dg[1]=tmp%10;

  tmp/=10;

  dg[0]=tmp%10;

  // SELECTAREA DIGIT-LUI ACTIV

  ds=(ds+1)%4;

  PORTE=~(1<<ds);

  // ACTIVAREA LED-URILOR AFERENTE VALORI AFISATE

  PORTD  =hx[dg[ds]];

  \_delay\_ms(10);

}

// RUTINA DE DESERVIRE A INTRERUPEREI GENERATE DE EVENIMENTUL DE DEPASIRE (TIMER OVERFLOW 1)

// TIMER1\_OVF\_vect - ADRESA ALOCATA VECTORULUI DE INTRERUPERE TIMER1\_OVF\_vect

ISR(TIMER1\_OVF\_vect)

{

 // = TEMA 2B =

 // AJUSTATI VALORILE REGISTRELOR TCNT1H / TCNT1L ASTFEL INCAT PERIOADA DE GENERARE

 // A INTRERUPERILOR SA FIE DE 1Hz (1sec.)

  TCNT1H = 0x00; // SETARE TIMER 1 ...

  TCNT1L = 0x00;

  // LIMITAREA SUPERIOARA A CONTORULUI TICK, PENTRU OBTINEREA CELOR 60MIN / 60SEC

  if(tick<3599)

     tick++;

  else

     tick=0;

}

// RUTINA DE DESERVIRE A INTRERUPEREI EXTERNE INT6

// APASAREA BUTONULUI RIGHT (PE6/T3/INT6, PINUL 8) VA DETERMINA INCREMENTAREA VARIABILEI TICK

// INT6\_vect - ADRESA ALOCATA VECTORULUI DE INTRERUPERE INT7

ISR(INT6\_vect)

{

 // LIMITAREA SUPERIOARA A FACTORULUI DE UPLERE A SEMNALULUI PWM

 // PASUL DE INCREMENTARE 5 LA FIECARE APASARE DE BUTON (RIGHT)

 if(OCR2<170)

    OCR2+=5;

}

// RUTINA DE DESERVIRE A INTRERUPEREI EXTERNE INT7

// APASAREA BUTONULUI LEFT (PE7/ICP3/INT7, PINUL 9) VA DETERMINA DECREMENTAREA VARIABILEI TICK

// INT6\_vect - ADRESA ALOCATA VECTORULUI DE INTRERUPERE INT7

ISR(INT7\_vect)

{

 // LIMITAREA INFERIOARA A FACTORULUI DE UPLERE A SEMNALULUI PWM

 // PASUL DE DECREMENTARE 5 LA FIECARE APASARE DE BUTON (LEFT)

  if(OCR2>9)

     OCR2-=5;

}

// == TEMA2 == CERINTE:

// INITIALIZAREA MODULULUI TIMER1

// MODUL DE FUNCTIONARE: NORMAL

// FRECVENTA CEASULUI INTERN 1MHz (CIRCUIT OSCILANT RC INTERN: 1MHz )

// FRECVENTA INTRERUPERILOR GENERATE DE MODULUL TIMER1: 1Hz (1sec)

// DIVIZORUL DE FRECVENTA(PRESCALER): 256

// MODUL DE FUNCTIONARE OC2: CLEAR (PINUL OC2 TRECE IN 0

// IN MOMENTUL IN CARE REGISTRELE TCNT2 = OCR2 )

void timer1\_init(void)

{

 TCCR1B = 0x00; // STOP TIMER 1

 TCNT1H = 0x00; // SETARE VALORII START A REGISTRULUI (TCNT1=[TCNT1H][TCNT1L]) ...

 TCNT1L = 0x00;

 TCCR1A = 0x00;

 TCCR1B = 0x00; // PRESCALER 256 & START TIMER 1

 TIMSK  = 0x00; // ACTIVAREA MODULUI DE LUCRU IN INTRERUPERE PENTRU TIMER 1

}

// == TEMA3 == CERINTE:

// INITIALIZAREA MODULULUI TIMER2

// FORMA DE UNDA GENERATA: PHASE CORRECT PWM

// FRECVENTA SEMNALULUI PWM: ~30.6Hz

// DIVIZORUL DE FRECVENTA(PRESCALER): 64

// MODUL DE FUNCTIONARE OC2: CLEAR (PINUL OC2 TRECE IN 0

// IN MOMENTUL IN CARE REGISTRELE TCNT2 = OCR2 )

void timer2\_init(void)

{

 TCCR2 = 0x00; //stop

 TCNT2 = 0x00; //setup

 OCR2  = 0x00;

 TCCR2 = 0x00; //start

}

int main()

 {

  // VARIABILE LOCALE UTILIZATE PENTRU CONVERSIE TICKs MINUTE/SECUNDE

  // IDX: ADRESAREA VALORILOR PWM PENTRU MODUL AUTO - SERVOMOTOR

  unsigned char min=0, sec=0, osec=0, idx=0;

  unsigned char pwm[]={  5, 20, 40, 60, 80,100,120,140,

                       160,150,120,100, 80, 60, 40, 20};

  // DEZACTIVAREA TUTUROR INTRERUPERILOR PENTRU PROGRAMAREA INTERFETELOR

  // ( ARE ROLUL DE A PREVINE PROGRAMAREA DEFECTUOASA SAU PARTIALA A MODULELOR / INTERFETELOR )

  // SE REALIZEAZA PRIN SETAREA IN 0 A BITULUI 7 IN CADRUL REGISTRULUI SREG

  // A BITUL 7 DIN SREG ESTE RESPONSABIL CU ACTIVAREA / DEZACTIVAREA GLOBALA A INTRERUPERILOR

  cli();

  // SETAREA PORTULUI "E" (PORTE ALOCAT SELECTARI DIGITILOR DG3..DG0)

  // PINII PE3..PE0 CONFIGURATI CA PINI DE IESIRE

  DDRE=0x0F;

  // PORTE PE3..PE0 PINI SETATI IN 0 LOGIC -> 0V

  PORTE=0xC0;

  // CONFIGURAREA PORTULUI "D",

  // PINII PORTD[6..0] CONFIGURATI CA PINI DE IESIRE

  //(PORTD[6..0] ALOCAT CONTROLULUI SEGMENTELOR DE LED A..G)

  // PINUL PD7/T2 PIN DE INTRARE

  DDRD=0x7F;

  // PINII PD6..PD0 SETATI IN 0 LOGIC -> 0V (TOATE SEGMENTELE DE LEDURI STINSE)

  // ACTIVAREA REZISTENTEI DE PULL-UP PENTRU PINUL PD7/T2

  PORTD=0x80;

  // SETAREA PORTULUI "F" (PORTF ALOCAT BARGRAPHULUI)

  // PINII PF7..PF0 CONFIGURATI CA PINI DE IESIRE

  DDRF=0xFF;

  PORTF=0xAA;

  // PINUL PB7/OC2 - UTILIZAT PENTRU GENERARE SEMNAL PWM \

  // PIN CONFIGURAT CA PIN DE IESIRE

  DDRB=0x80;

  PORTB=0x80;

  // = TEMA2 =

  // IDENTIFICATI SI REPROGRAMATI REGISTRII MODULULUI TIMER / COUNTER1 (VEZI TIMER1.PDF)

  // VEZI RUTINA DE MAI SUS timer1\_init():

  timer1\_init();

  // = TEMA3 =

  // IDENTIFICATI SI REPROGRAMATI REGISTRII MODULULUI TIMER / COUNTER2 (VEZI TIMER2.PDF)

  // VEZI RUTINA DE MAI SUS timer2\_init():

  timer2\_init();

  // CONFIGURAREA INTRERUPERILOR EXTERNE INT6/INT7

  EICRB = 0b11100000;

  EIMSK = 0xC0;

  // ACTIVAREA TUTUROR INTRERUPERILOR (SETAREA IN 1 IN CADRUL REGISTRULUI SREG

  // A BITULUI RESPONSABIL CU ACTIVAREA / DEZACTIVAREA GLOBALA A INTRERUPERILOR

  sei();

   // BUCLA INFINITA

   while (1)

    {

    // APELAREA RUTINEI PENTRU ACTUALIZAREA VALORII PE AFISORUL 4x7SEG

    // VALOAREA DE AFISAT SE AFLA IN STOCATA IN VARIABILA VAL\_7SEG

    fUpdate7Seg();

    osec=sec;

    // CONVERSIA NUMARATORULUI TICK LA MINUTE / SECUNDE

    // VARIABILA TICK ESTE INCREMENTATA IN RUTINA DE INTRERUPERE ISR(TIMER1\_OVF\_vect)

    sec=(unsigned char)(tick % 60);

    min=(unsigned char)(tick / 60);

    // IDENTIFICARE MOD FUNCTIONARE: AUTO/MANUAL

    if( !(PIND & 0x80) )

     {

       // DETECTARE EVENIMENT SCHIMBARE SECUNDA

       if(osec!=sec)

       {

         // SETARE VALOARE PWM

         OCR2=pwm[idx];

       idx=(idx+1)&0x0F;

       }

     }

    // AFISAREA PE BARGRAPH IN FORMAT BINAR A REGISTRULUI OCR2

    // VALOAREA CONTINUTA DE CATRE REGISTRL OCR2[0..255]

    // REGISTRUL OCR2 DETERMINA FACTORL DE UMPLERE AL SEMNALULUI PWM GENERAT

    PORTF=OCR2;

    // AFISAREA PE AFISORUL CU 7 SEGMENTE A MINUTELOR & SECUNDELOR

    // (PRIMI 2 DIGITI DIN PARTEA STANGA SUNT ALOCATI PENTRU MINUTE    )

    // (PRIMI 2 DIGITI DIN PARTEA DREAPTA SUNT REZERVATI PENTRU SECUNDE)

    val\_7seg = min\*100+sec;

    }

 }

## Tema 2 (vezi codul sursa de mai sus, secțiunea // = TEMA 2 =  )

Identificați si reprogramați regiștrii modulului TIMER/COUNTER1 (vezi TIMER1.pdf) astfel încât:

- numărătorul TIMER/COUNTER1 sa ruleze in modul întrerupere.

- întreruperea sa fie generata periodic, cu perioada de 1HZ (1sec.).

- semnalul aplicat pinului PD7/T2 va fi generat de apăsarea butonului TCNT2.

## Tema 3

TEMA 3 (vezi codul sursa de mai sus, secțiunea // = TEMA 3 =  )

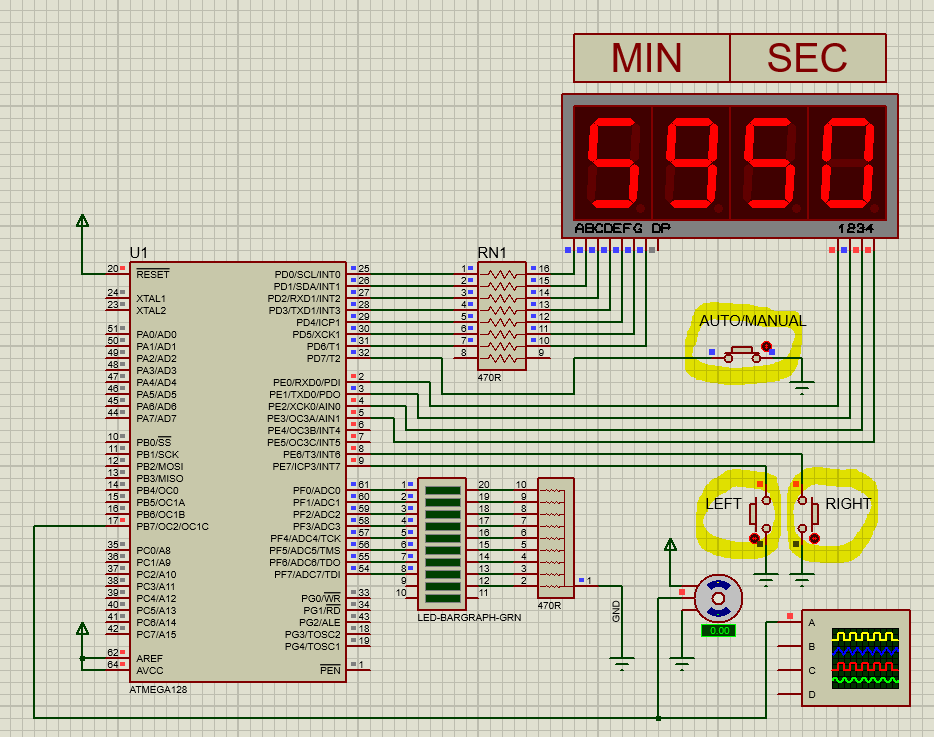
Identificați regiștrii aferenți modulului TIMER/COUNTER2 (vezi TIMER2*.pdf*)

Reprogramați regiștrii identificați astfel încât:

- pe pinul PB7/OC2/OC1C sa se genereze un semnal de tipul PWM cu frecventa ~ 30.6Hz.

- ajustarea factorul de umplere al semnalului PWM se va realiza fie automat prin intermediul butonului AUTO/MANUAL, fie prin intermediul butoanelor LEFT / RIGHT

- semnalul PWM generat are rolul de a controla poziția de funcționare a servomotorului [-90 .. +90 grade geometrice].



**IMPORTANT**

**Buna practică inginerească cere ca desenul să fie foarte clar,**

**să nu existe suprapuneri între înscrisuri şi elementele de circuit.**

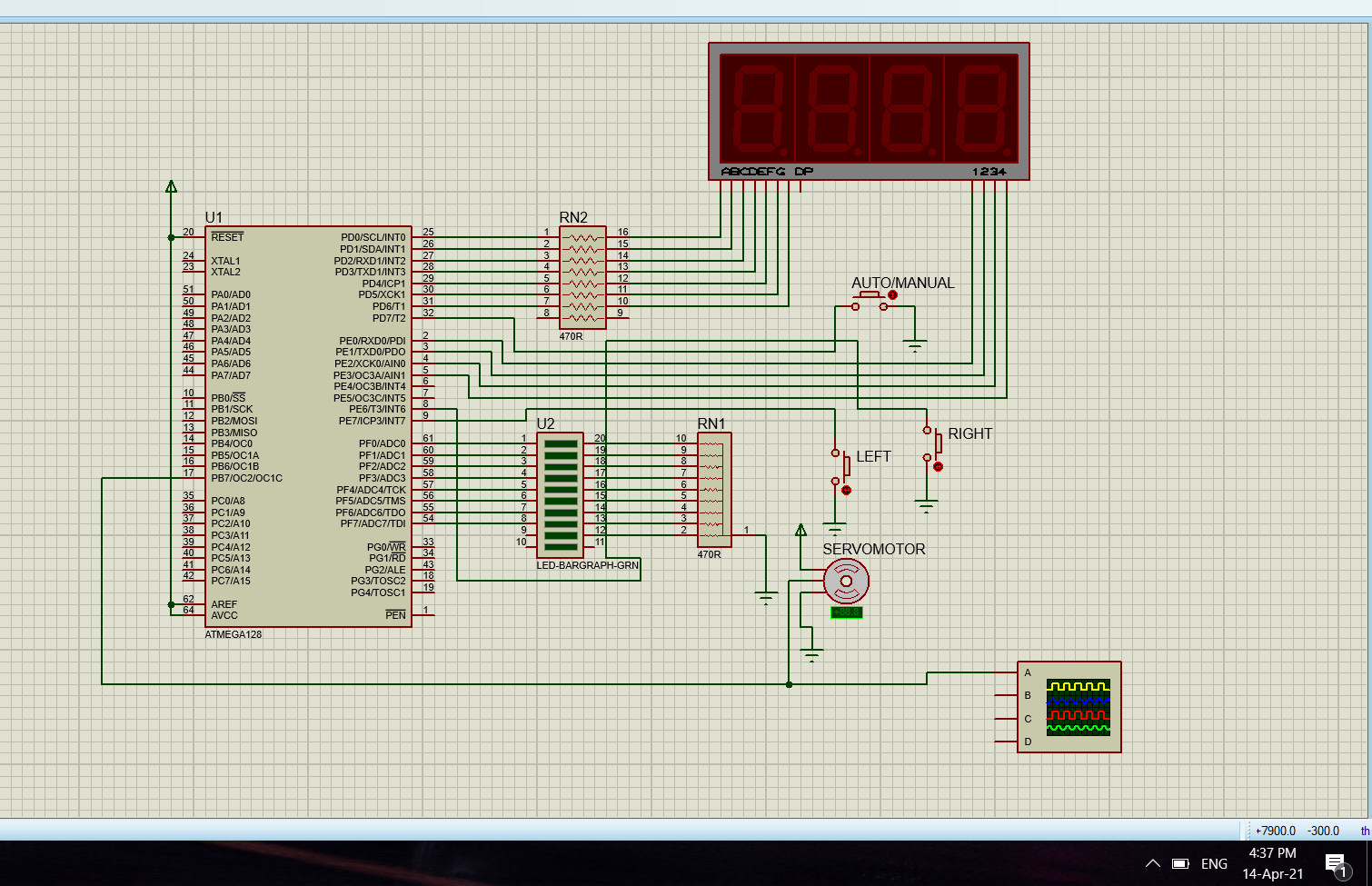
**Toate înscrisurile (nume, valori, parametri) se deplasează până când se văd clar atât componentele cât şi înscrisurile.**

## Cerințe

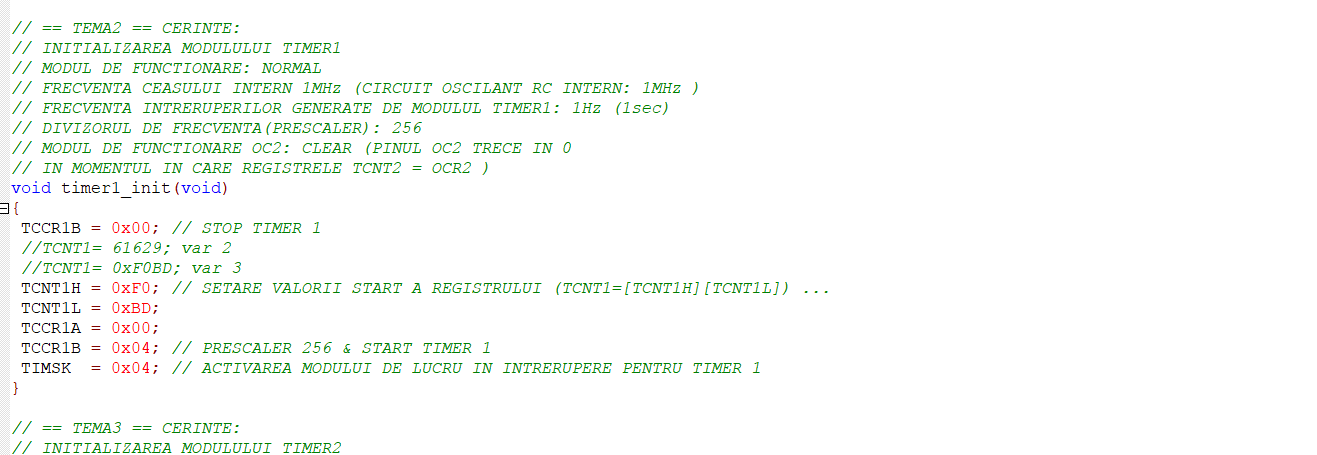
Lucrarea trebuie sa cuprindă rezolvarea temelor:

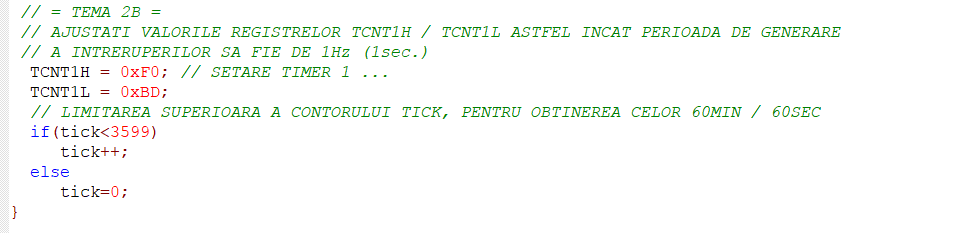
( **funcționalitatea descrisa mai sus poate fi vizualizata in videoclipul L4\_AC.mp4** )

* Rezolvarea temei 1;

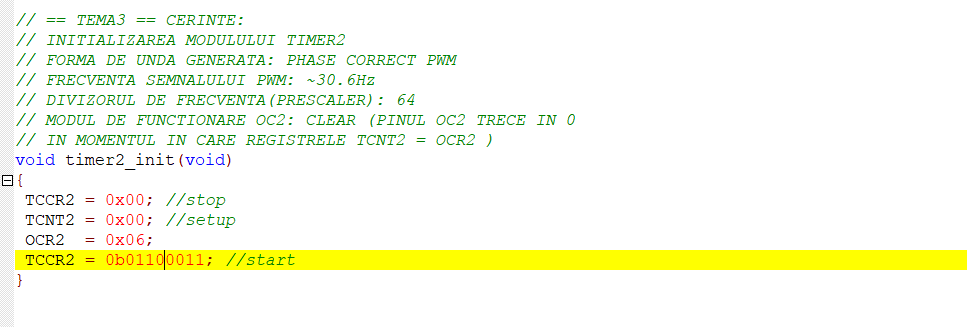


* Rezolvarea temei 2;





* Rezolvarea temei 3;

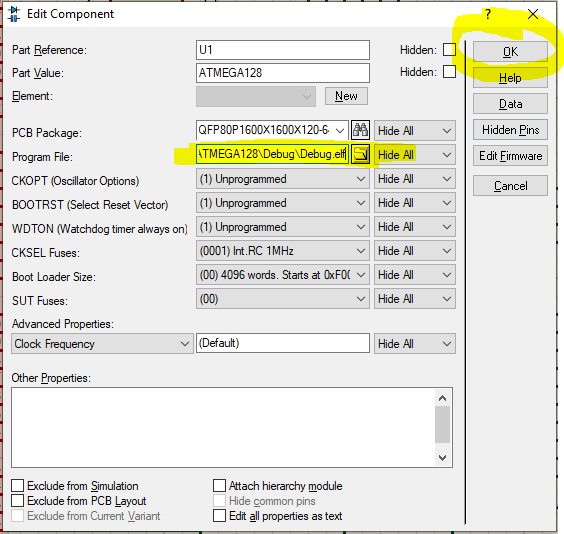


## CODUL HEX DEMO:

Încărcarea codului HEX DEMO presupune executarea următorilor pași:

* salvarea într-un fișier cu extensia hex a codului in format Intel HEX (<https://en.wikipedia.org/wiki/Intel_HEX>) de mai jos.
* încărcarea in microcontroler a codului hex ( poziționam mouse-ul deasupra microcontrolerului,

apăsam tastele CTRL+E, selectam fișierul cu extensia .hex salvat )



:100000000C9446000C9465000C9465000C946500FB

:100010000C9465000C9465000C9465000C9422010E

:100020000C943E010C9465000C9465000C946500E2

:100030000C9465000C9465000C94FC000C94650015

:100040000C9465000C9465000C9465000C9465009C

:100050000C9465000C9465000C9465000C9465008C

:100060000C9465000C9465000C9465000C9465007C

:100070000C9465000C9465000C9465000C9465006C

:100080000C9465000C9465000C94650011241FBE4F

:10009000CFEFD0E1DEBFCDBF11E0A0E0B1E0EAE3F9

:1000A000F4E000E00BBF02C007900D92AE31B10743

:1000B000D9F711E0AEE1B1E001C01D92AF31B10757

:1000C000E1F70E946F010C941B020C940000DF9377

:1000D000CF9300D000D0CDB7DEB769837A838B830E

:1000E0009C8329813A814B815C81DA01C9010197A6

:1000F000A109B10989839A83AB83BC83211531059A

:100100004105510571F70F900F900F900F90CF910F

:10011000DF910895FF920F931F93DF93CF9300D049

:1001200000D0CDB7DEB720910A0130910B0102E378

:1001300010E0F8011082C9016AE070E00E94F40149

:10014000F82EC9016AE070E00E94F401262F372FD3

:10015000862F972F6AE070E00E94F401F82F822F1B

:10016000932F6AE070E00E94F401262F372F862F2C

:10017000972F6AE070E00E94F401E82F822F932FFE

:100180006AE070E00E94F401CB016AE070E00E9436

:10019000F401482F80911E0190E0019664E070E028

:1001A0000E94080280931E0190E021E030E0082EBA

:1001B00002C0220F331F0A94E2F7209523B9498326

:1001C000EA83FB83FC82FE01E80FF91F8181E0E0F6

:1001D000F1E0E80FF11D8081F801808384EC99E063

:1001E0000197F1F70F900F900F900F90CF91DF9143

:1001F0001F910F91FF9008951F920F920FB60F92CB

:1002000011242F938F939F9380EF8DBD8EEB8CBD28

:1002100080910C0190910D012EE08F30920730F407

:10022000019690930D0180930C0104C010920D0172

:1002300010920C019F918F912F910F900FBE0F90F4

:100240001F9018951F920F920FB60F920BB60F9238

:1002500011248F93EF93FF9383B58A3A28F4E3E454

:10026000F0E080818B5F8083FF91EF918F910F9001

:100270000BBE0F900FBE0F901F9018951F920F92FC

:100280000FB60F920BB60F9211248F93EF93FF933B

:1002900083B58A3028F0E3E4F0E0808185508083E4

:1002A000FF91EF918F910F900BBE0F900FBE0F90AB

:1002B0001F901895EEE4F0E0108280EF8DBD8EEB7C

:1002C0008CBD1FBC84E0808387BF0895E5E4F0E027

:1002D000108214BC85E083BD83EE808308954F9225

:1002E0005F927F928F929F92AF92BF92CF92DF9256

:1002F000EF92FF920F931F93DF93CF93CDB7DEB7AB

:1003000060970FB6F894DEBF0FBECDBFDE01119629

:10031000EEE0F1E080E101900D928150E1F7F89478

:100320008FE082B910EC13B98FE781BB80E882BB04

:100330009FEF909361008AEA8093620097BB98BB1D

:100340000E945A010E94660180EE8ABF19BF78940C

:10035000FF24EE240F2EF0E38F2E9924F02D2E0192

:100360000894411C511C0F2EF3E4CF2EDD24F02DF8

:100370000F2EF2E6AF2EBB24F02D0F2EF4E67F2ECB

:10038000F02D0E948A0000910C0110910D01C8010E

:100390006CE370E00E94F401282FF401808188232F

:1003A0007CF0F21669F0F201EE0DF11D8081F6018C

:1003B00080838E2D8F5F0F2EFFE0EF2EF02DE82231

:1003C000F6018081F5018083C8016CE370E00E9432

:1003D000F401679DB0011124620F711D70930B0130

:1003E00060930A01F22ECDCFAA1BBB1B51E107C0BF

:1003F000AA1FBB1FA617B70710F0A61BB70B881FB5

:10040000991F5A95A9F780959095BC01CD01089543

:1004100097FB092E07260AD077FD04D0E5DF06D02A

:1004200000201AF4709561957F4F0895F6F7909526

:0A04300081959F4F0895F894FFCFC7

:10043A00BF86DBCFE6EDFD87FFEF1C00060E051435

:0E044A00283C5064788CA0967864503C2814AE

:00000001FF