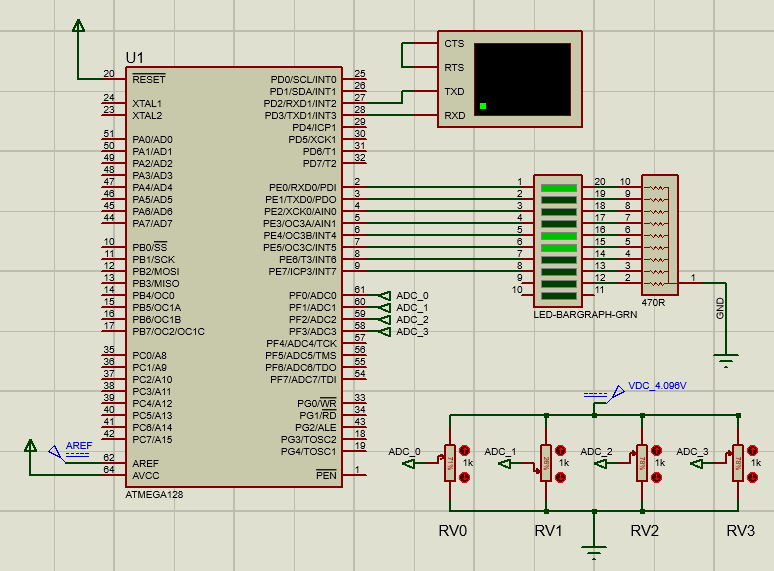
# Laboratorul nr. 6

## 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 E, F(ADC) a microcontroler-lui ATmega128
* configurarea / utilizarea modulului / interfeței seriale asincrone USART1 utilizând mecanismul de întrerupere atât pentru recepția cat si pentru transmisia pe portul serial.

### **Tema 1**

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



**Modul de lucru :**

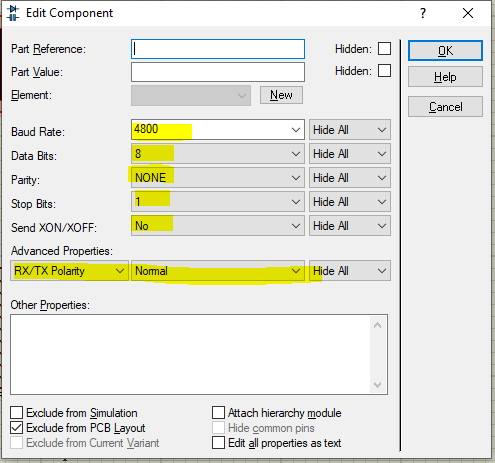
**Pașii pentru lansarea programului *Proteus, crearea proiectului, plasarea componentelor pe planșa de lucru au fost prezentați in cadrul laboratorului cu numărul 1.***

## Adăugarea terminalului virtual pe planșa de lucru:

## 

## Ajustarea parametrilor terminalului virtual:

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



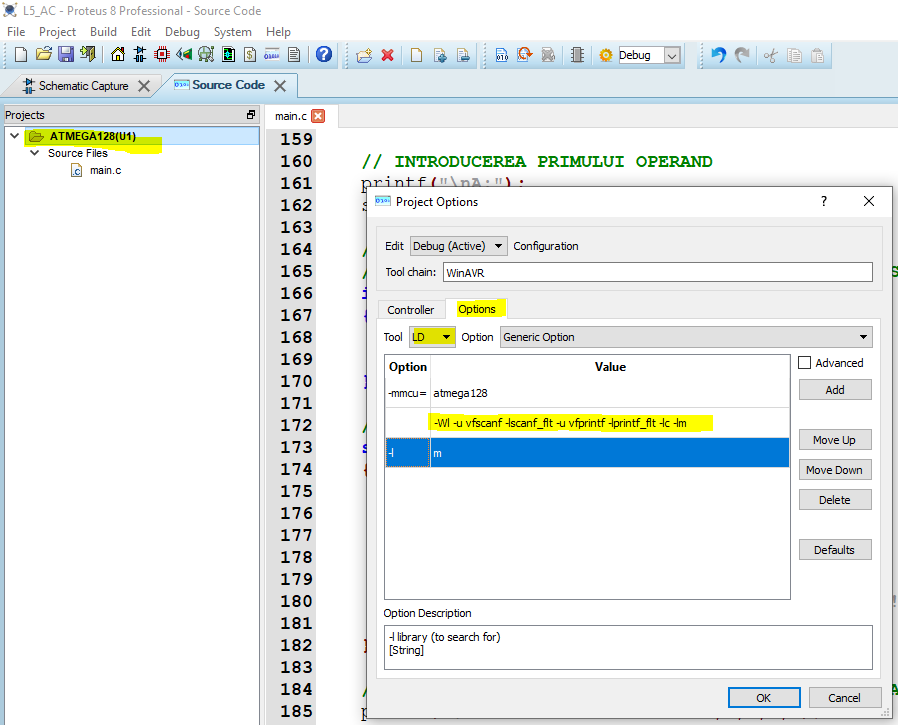
Următorii pașii necesari au fost descriși in cadrul laboratorului cu numărul 1

## Interconectarea componentelor.

## Rularea / oprirea aplicației de tip firmware:

## Modificarea codului sursa (Source Code)

Configurarea compilatorului (linker-lui) pentru a o feri suport pentru printf / scanf formatul standard de lucru.



Opțiunea pentru LD: **-Wl -u vfscanf -lscanf\_flt -u vfprintf -lprintf\_flt -lc -lm**

Pornind de la următorul cod sursa rezolvați următoarele doua teme (codul poate fi descărcat de pe platforma elearning):

#include <stdio.h>

#include <ctype.h>

#include <string.h>

#include <stddef.h>

#include <stdlib.h>

#include <avr/io.h>

#include <avr/interrupt.h>

#include <avr/sleep.h>

#include <math.h>

// VALOAREA REFERINTEI DE TENIUNE CONECTATA LA PINUL AREF

const float AREF=4.096;

// DECLARAREA BUFFER-ULUI DE TRANSMISIE PE SERIALA (DIMENSIUNEA ALOCATA 128 BYTES)

char TxBuff[128];

float ADC\_0=0.123;

float ADC\_1=1.234;

float ADC\_2=2.345;

float ADC\_3=3.456;

// == TEMA2 ==

// INITIALIZAREA MODULULUI UART1, CERINTE:

// IDENTIFICAREA REGISTRILOR RESPONSABILI CU, COMUNICATIA PE INTERFATA USART1

// ACTIVARE MODULUI DE LUCRU: RECEPTIE TRANSMISIE DATE IN INTRERUPERE (REG.: UCSR1B - pg. 189)

// ( MECANISMUL DE COMUNICATIE CU INTRERUPERE - 'INTERRUPT METHOD')

// BAUD RATE:4800 (BIT U2X=0, FOSC=1.00MHZ, REG.: UBRR1L, TABEL: 82)

// FORMULA DE CALCUL A BAUD RATE-LUI (TABEL: 74)

// DIMENSIUNE PACHET DATA: 8 BIT (TABEL: 80)

// COMUNICATIE SERIALA ASINCRONA: (REG.: UCSR1C  TABEL: 77)

// FARA PARITATE (REG.: UCSR1C  TABEL: 78)

// 1 BIT STOP: (REG.: UCSR1C  TABEL: 79)

// CONTROL TRASFER DATE: FARA CONTROL (SW/HW)

void uart1\_init(void)

{

 UCSR1B = 0x00; //disable while setting baud rate

 UCSR1A = 0x00;

 UCSR1C = 0x06;

 UBRR1L = 0x0C; //set baud rate lo

 UBRR1H = 0x00; //set baud rate hi

 UCSR1B = 0x18;

}

// INITIALIZAREA CONVERTORULUI ADC

void adc\_init(void)

{

 ADCSRA = 0x00; // DEZACTIVAREA MODULULUI ADC PENTRU CONFIGURARE

 ADMUX  = 0x00; // SELECTARE REFERINTA DE TENSIUNE EXTERNA (AREF) / CANALUL DE ACHIZITIE 0

 ACSR   = 0x80; // DEZACTIVARE COMPARATOR ANALOGIC

 ADCSRA = 0xC6; // CONFIGURAREA MODULUI DE FUNCTIONARE / ACTIVAREA ACESTUIA

}

// == TEMA3A ==

// IMPLEMENTAREA FUNCTIONALITATII RUTINEI DE RECEPTIE IN INTRERUPERE, CERINTE:

// IDENTIFICAREA REGISTRULUI RESPONSABIL CU RECEPTIA / TRANSMISIA PE INTERFATA USART1 (PG. 188)

// IDENTIFICAREA REGISTRULUI DE STAREA PENTRU INTERFATA USART1 (PG. 188)

// DACA VALOAREA BYTE-UL (CARACTERUL) RECEPTIONAT SE AFLA IN INTERVALUL [0..3]

// BUFFER-UL TxBuff VA CONTINE URMATORUL MESAJ:

// "\nADC\_{X}: %1.3f [V]", UNDE {X} REPREZINTA NUMARUL CANALULUI ANALOGIC SELECTAT

// PE BARGRAPF-UL CU LEDURI SE VA AFISA CODUL ASCII AL CARACTERULUI RECEPTIONAT

// TOT IN ACEASTA RUTINA SE VA INITIA INCEPEREA TRANSMISIEI IN INTRERUPERE A CONTINUTULUI BUFFER-LUI TxBuff

ISR(USART1\_RX\_vect)

{

/\*

 unsigned char Rcv=UDR1;

 PORTE=Rcv;

 switch(Rcv)

 {

   case '0': sprintf(TxBuff,"\nADC\_0: %1.3f [V]",ADC\_0); break;

   case '1': sprintf(TxBuff,"\nADC\_1: %1.3f [V]",ADC\_1); break;

   case '2': sprintf(TxBuff,"\nADC\_2: %1.3f [V]",ADC\_2); break;

   case '3': sprintf(TxBuff,"\nADC\_3: %1.3f [V]",ADC\_3); break;

   default : sprintf(TxBuff,"\nVA ROG IN INTERVALUL [0..3]");

 }

 UDR1='\r';

\*/

}

// == TEMA3B ==

// IMPLEMENTAREA FUNCTIONALITATII RUTINEI DE TRANSMISIE IN INTRERUPERE, CERINTE:

// CONTINUTUL BUFFER-LUI TxBuff[] SE VA TRANSMITE PE SERIALA CARACTER CU CARACTER

// PRIN INTERMEDIUL REGISTRULUI UDR1

// TRANSMISIA PE SERIALA SE VA OPRI LA INTALNIREA CARACTERULUI CARE MARCHEAZA

// SFARSIT DE STRING '\0'

// CARACTERUL CARE MARCHEAZA SFARSITUL DE STRING '\0' NU TREBUIE TRANSMIS!!

// LA INTALNIREA CARACTERULUI CARE MARCHEAZA SFARSIT DE STRING '\0' CONTORUL idx

// TREBUIE RESETAT LA 0 (0 -> PREGATIREA PENTRU O NOUA TRANSMISIE)

ISR(USART1\_TX\_vect)

{

  //TODO MODIFICATI CONTINUTUL RUTINEI

  static unsigned char idx=0;

  UDR1=TxBuff[idx];

}

// RUTINA RESPONSABILA CU ACHIZITIA SEMNALULUI ANALOGIC

int ReadAdc(char Ch)

{

    ADMUX = Ch & 0x0F;          // REFERINTA DE TESIUNE EXTERNA AREF, FILTRARE CANALE DE LUCRU

    ADCSRA |= (1<<ADSC);        // START CONVERSIE ADC

    while (ADCSRA & (1<<ADSC)); // ASTEAPTA FINALIZAREA PROCESULUI DE CONVERSIE ADC

    return ADC;                 // RETURNEAZA VALOAREA CONVERSIEI ADC (10BIT)

}

int main()

 {

  // DECLARAREA SI INITIALIZAREA STRINGULUI "msg"

  char \* msg="http://users.utcluj.ro/~baruch/sie/labor/Port-Serial.pdf";

  // 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();

  // APELAREA RUTINEI RESPONSABILE CU INITIALIZAREA INTERFETEI SERIALE ASINCRONE USAR1

  uart1\_init();

  // APELAREA RUTINEI RESPONSABILE CU INITIALIZAREA MODULULUI ADC

  adc\_init();

  // SETAREA PORTULUI 'E' CA PORT DE IESIRE ( COMANDA LEDURILOR BARGRAPHULUI ATASAT uCONTROLERULUI )

  // PE PORTUL 'E' SE VA AFISA CODUL ASCII AL ULTIMEI TASTE APASATE IN TERMINALUL VIRTUAL

  PORTE=0xAA;

  DDRE=0xFF;

  // ACTIVAREA TUTUROR INTRERUPERILOR (SETAREA IN 1 A BITULUI RESPONSABIL CU ACTIVAREA / DEZACTIVAREA GLOBALA

  // A INTRERUPERILOR, IN CADRUL REGISTRULUI SREG

  sei();

  // SETAREA BUFFERULUI TxBuff CU URMATORUL MESAJ

  sprintf(TxBuff,"\nCOMUNICATIA SERIALA ASINCRONA IN INTRERUPERE\r\n%s\r\n\r\nSISTEM DAQ(ADC), ALEGETI CANALUL DE LUCRU[0..3]:\r\n",msg);

  // DECLANSAREA MECANISMULUI DE TRANSMISIEI PE SERIALA

  UDR1='\r';

  // BUCLA INFINITA

  // ACHIZITIA MARIMILOR ANALOGICE PRIN INTERMEDIUL CONVERTORULUI ADC

  // PROGRAMUL DEMONSTRATIV ACHIZITIONEAZA MARIMILE ANALOGICE APLICATE CANALELOR 0 - 3

  // MARIMILE ANALOGICE ACHIZITIONATE SUNT CONVERTITE LA FORMATUL NATURAL DE REPREZENTARE

  // [0..1023] -> [0.000V - 4.095V ]

  while (1)

   {

     ADC\_0=(float)ReadAdc(0)\*AREF/1024.;

     ADC\_1=(float)ReadAdc(1)\*AREF/1024.;

     ADC\_2=(float)ReadAdc(2)\*AREF/1024.;

     ADC\_3=(float)ReadAdc(3)\*AREF/1024.;

   }

 }

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

**== TEMA2 ==**

Initializarea modulului UART1, cerinte:

## - identificarea registrelor responsabili cu, comunicația pe interfața USART1

## - activare modului de lucru: recepția / transmisia datelor in întrerupere (reg.: UCSR1B - pg. 189)

## ( mecanismul de comunicație cu întrerupere - 'interrupt method')

## - baud rate:4800 (bit u2x=0, fosc=1.00MHz, reg.: UBRR1L, tabel: 82)

## formula de calcul a baud rate-lui (tabel: 74)

## - dimensiune pachet data: 8 bit (tabel: 80)

## - comunicație seriala asincrona: (reg.: UCSR1C tabel: 77)

## - fără paritate (reg.: UCSR1C tabel: 78)

## - 1 bit stop: (reg.: UCSR1C tabel: 79)

## - control transfer date: fără control (sw/hw)

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

**== TEMA3A ==**

Implementarea funcționalității rutinei de recepție in întrerupere, cerințe:

- identificarea registrului responsabil cu recepția / transmisia pe interfața USART1 (pg. 188)

- identificarea registrului de starea pentru interfața USART1 (pg. 188)

- daca valoarea byte-ul (caracterul) recepționat se afla in intervalul [0..3]

- buffer-ul TxBuff va conține următorul mesaj:

***"\nadc\_{x}: %1.3f [v]",*** unde ***{x}*** reprezintă numărul canalului analogic selectat

- pe bargrapf-ul cu leduri se va afișa codul ASCII al caracterului recepționat

- tot in aceasta rutina se va iniția începerea transmisiei in întrerupere a conținutului buffer-lui TxBuff.

**== TEMA3B ==**

Implementarea funcționalității rutinei de transmisie in întrerupere, cerințe:

- conținutul buffer-lui TxBuff[] se va transmite pe seriala caracter cu caracter

- prin intermediul registrului udr1

- transmisia pe seriala se va opri la întâlnirea caracterului care marchează sfârșit de strung '\0'

- caracterul care marchează sfârșitul de strung '\0' nu trebuie transmis!!

- la întâlnirea caracterului care marchează sfârșit de strung '\0' contorul ***idx*** trebuie resetat la 0

( 0 -> pregătirea pentru o noua transmisie )

**IMPORTANT**

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

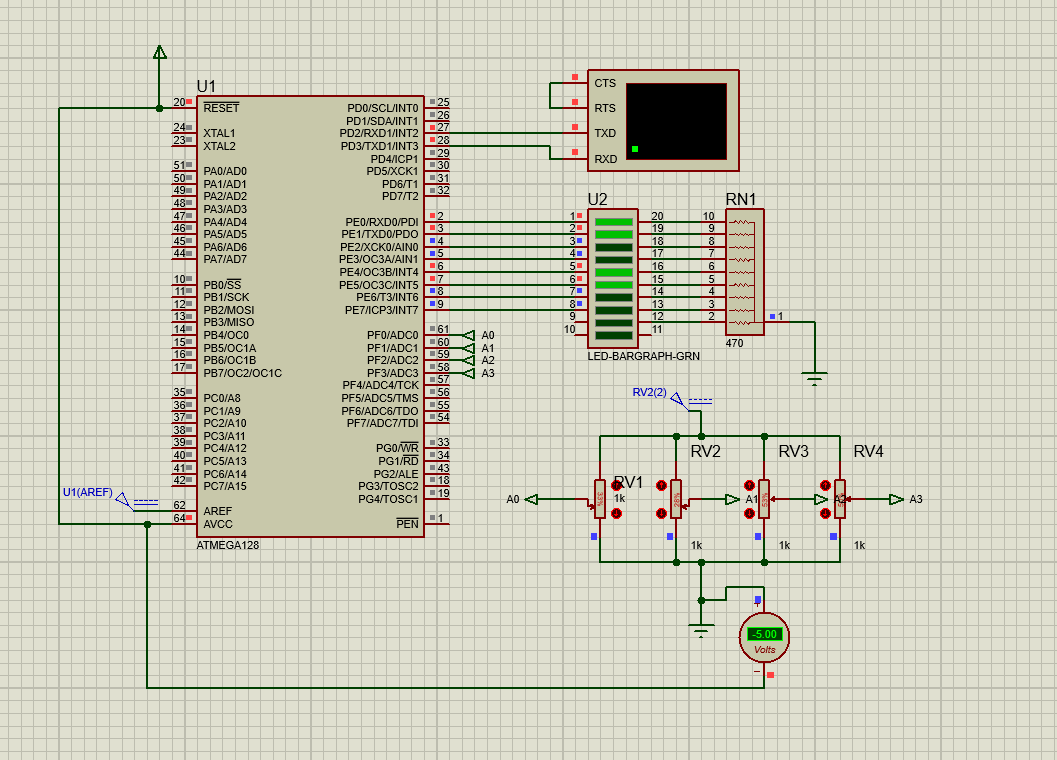
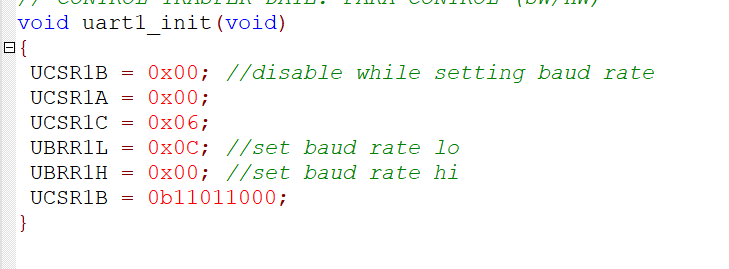
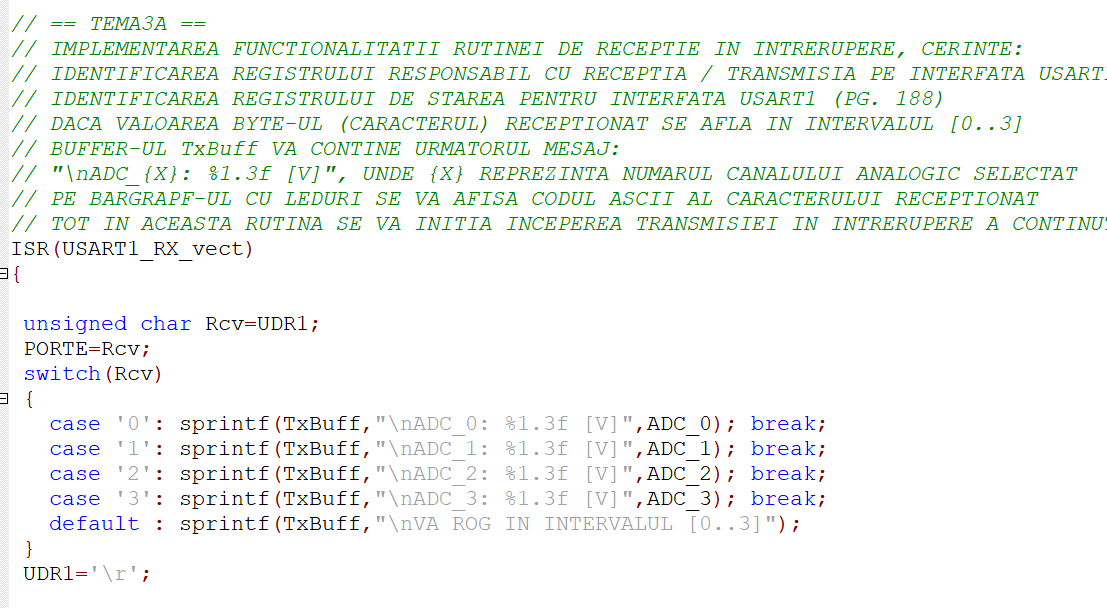
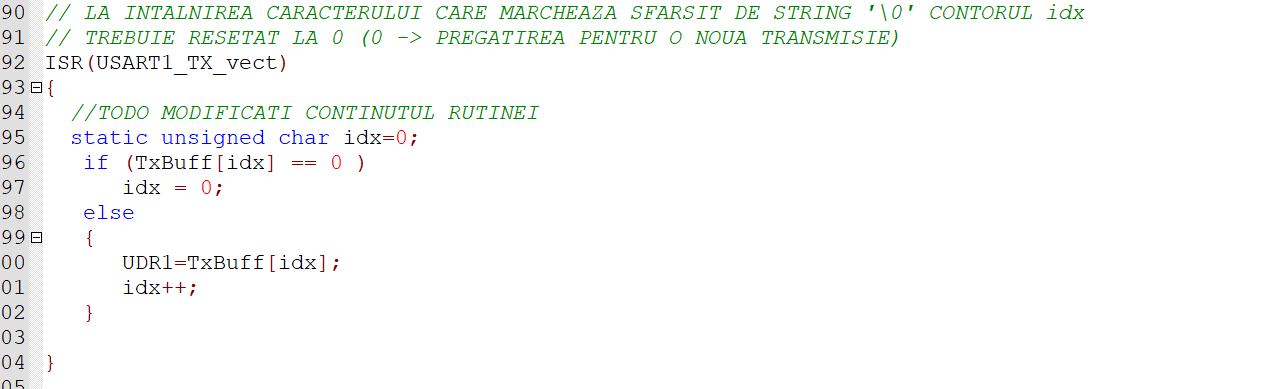
**să nu existe suprapuneri între înscrisuri şi elementele de circuit.**

**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** )

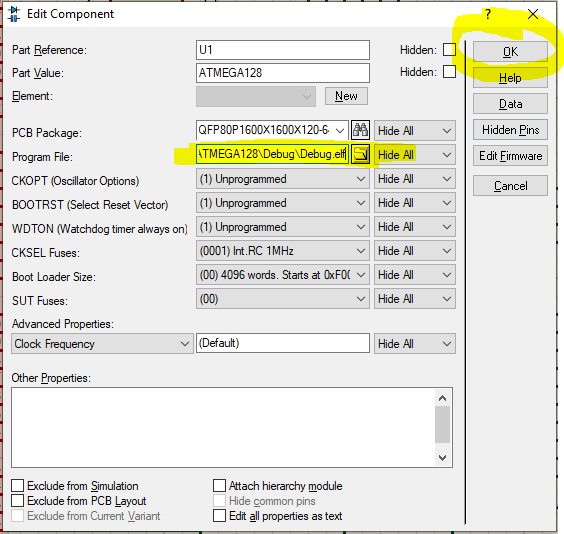
* Rezolvarea temei 1;
* 
* Rezolvarea temei 2;
* 
* Rezolvarea temei 3;
* 
* 

## 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 )



Codul HEX DEMO poate fi descarcat de pe platforma: https://elearning.unitbv.ro/