Sisteme cu MicroProcesoare

Cursul 3

Programarea unui microcontroler

Conf. Gigel Măceșanu

Universitatea
Transilvania
din Brașov
FACULTATEA DE INGINERIE ELECTRICĂ
ȘI ȘTIINȚA CALCULATOARELOR

Cuprins

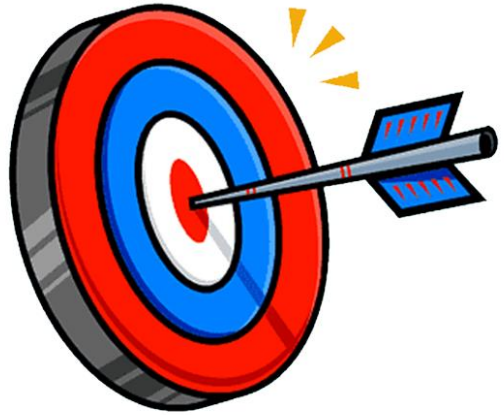
Universitatea
Transilvania
din Brașov
FACULTATEA DE INGINERIE ELECTRICĂ
ȘI ȘTIINȚA CALCULATOARELOR

- Programarea unui MC – prezentare generală
- Etapele programării
- Structura memoriei flash: bootloader și program aplicații
- Programarea modulelor Arduino
- Limbaje de programare
- Formatul unui fișier hex
- Programatoare și depanatoare

Obiectivul cursului

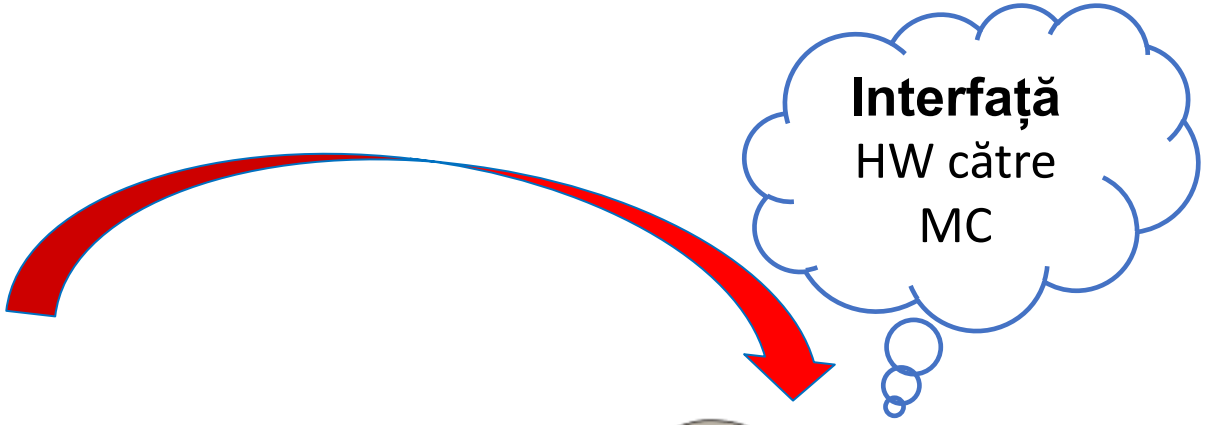
Universitatea Transilvania din Brașov
FACULTATEA DE INGINERIE ELECTRICĂ ȘI ȘTIINȚA CALCULATOARELOR

- Introducerea principalelor concepte și a etapelor necesare transferului unui program către memoria internă a unui microcontroler. Pentru aceasta se au în vedere:
  - Etapele prin care trece un program până este transformat într-un format specific MC
  - Structura memoriei, pentru un microcontroler AVR
  - Programarea efectivă a sistemelor Arduino
  - Limbaje de programare
  - Module HW necesare procesului de programare sau verificare



Programarea unui MC

Universitatea Transilvania din Brașov
FACULTATEA DE INGINERIE ELECTRICĂ ȘI ȘTIINȚA CALCULATOARELOR



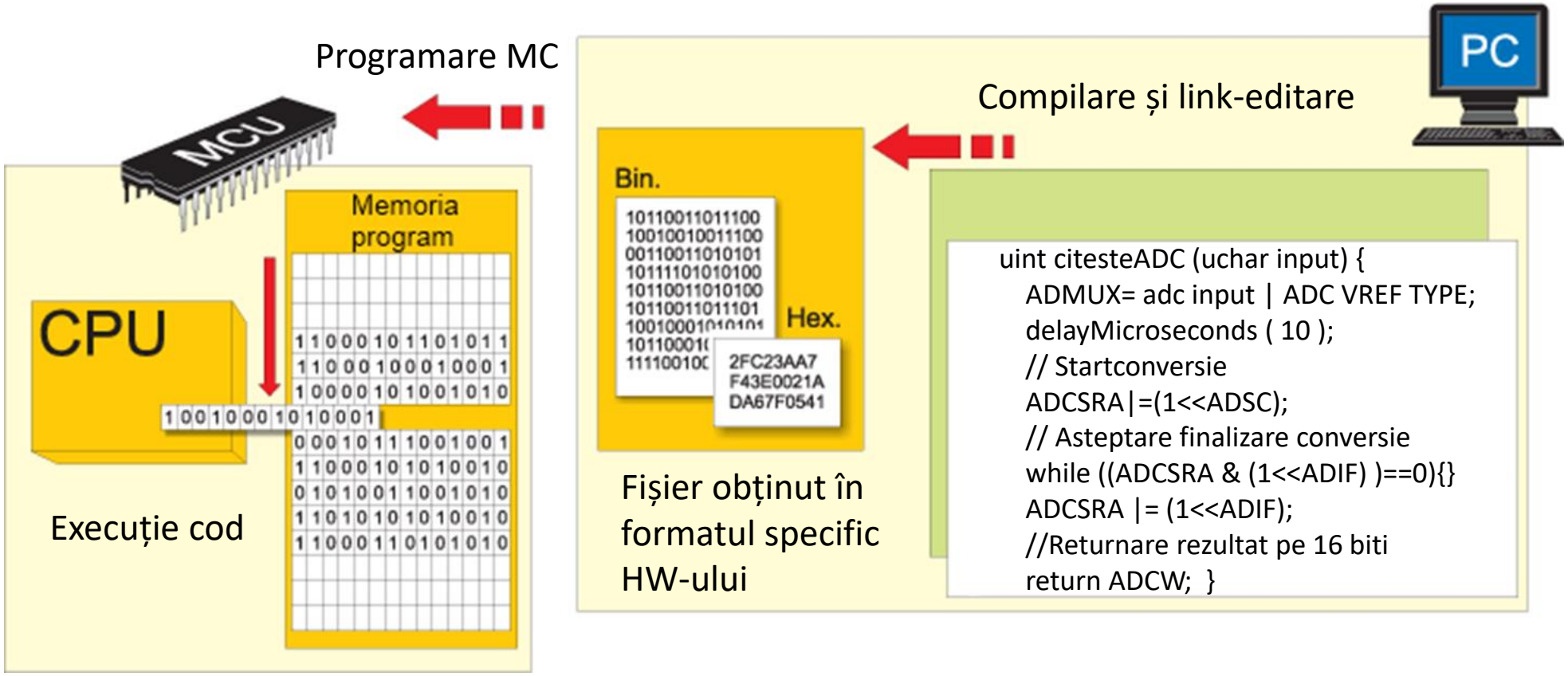
- Etapele programării unui MC

Etapele programării

Universitatea Transilvania din Brașov
FACULTATEA DE INGINERIE ELECTRICĂ ȘI ȘTIINȚA CALCULATOARELOR

- Etapele parcurse pentru programare unui MC când este utilizat limbajul C:

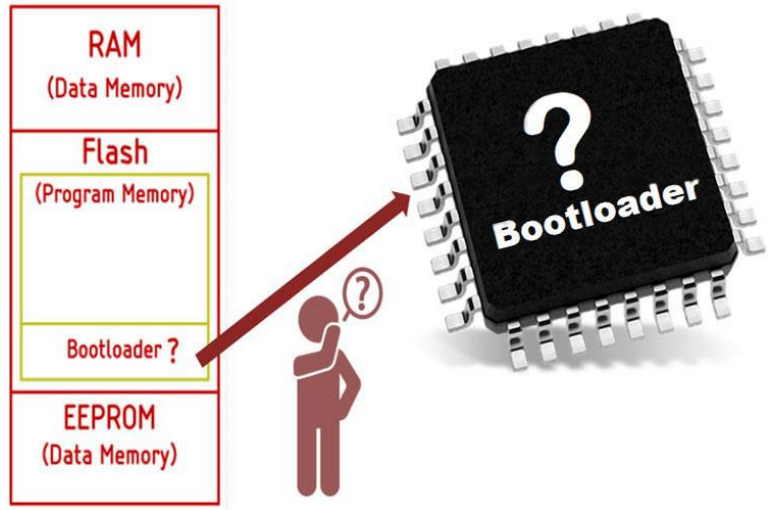
Scrierea codului sursă → Compilarea codului sursă în fișier obiect → Legarea fișierelor obiect și generare fișier binar → Programarea MC-ului cu fișierul binar → Execuția codului pe microcontroler



Bootloader

Universitatea Transilvania din Brașov
FACULTATEA DE INGINERIE ELECTRICĂ ȘI ȘTIINȚA CALCULATOARELOR

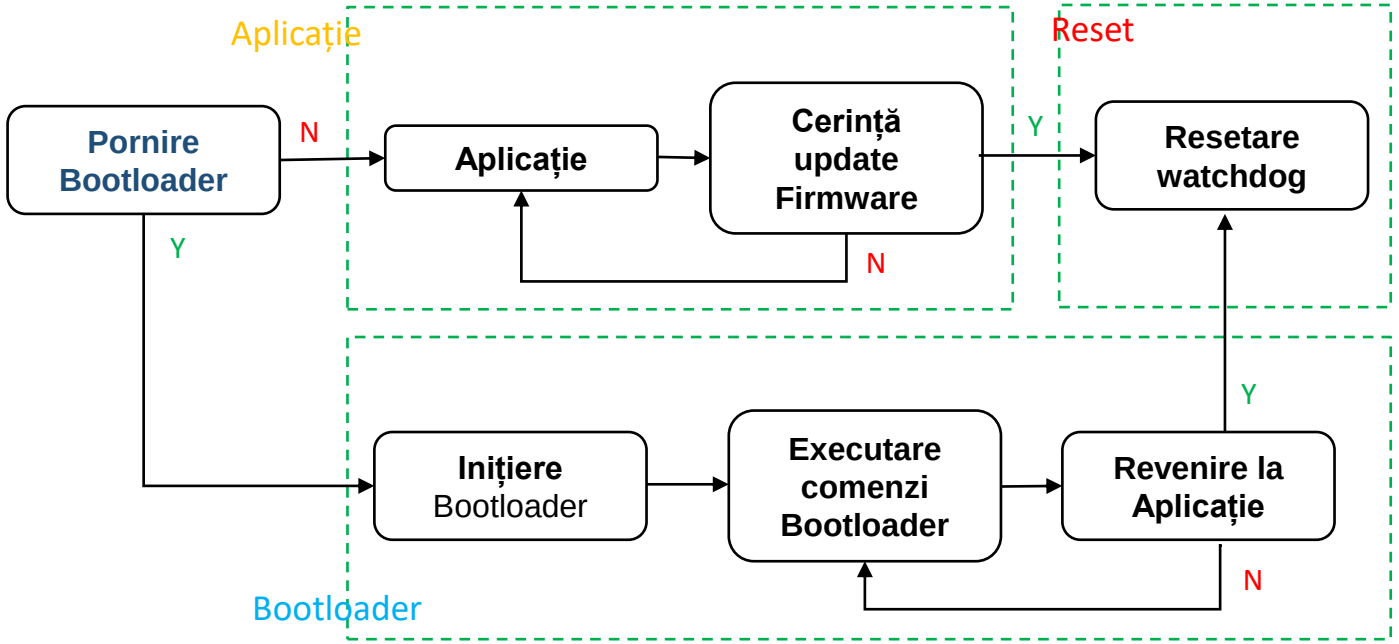
- Memoria flash a MC ATmega328P este împărțită în două secțiuni:
  - Bootloader
    - Un program care rulează când alimentăm microcontrolerul
    - Așteaptă trimiterea de pe un dispozitiv a programului utilizator, pe care îl scrie în memoria flash
    - Permite modificarea sau actualizarea software-lui de sistem
    - Poate utiliza diferite protocoale pentru transferul datelor: UART, I2C, USB
    - Facilitează programarea fără dispozitive externe



Bootloader

Universitatea Transilvania din Brașov
FACULTATEA DE INGINERIE ELECTRICĂ ȘI ȘTIINȚA CALCULATOARELOR

- Memoria flash a MC ATmega328P este împărțită în două secțiuni:
  - Bootloader
    - La pornire se decide dacă se execută cod aplicație sau bootloader code
    - Reset se poate da după un update de firmware



Memoria pentru aplicații

Universitatea Transilvania din Brașov
FACULTATEA DE INGINERIE ELECTRICĂ ȘI ȘTIINȚA CALCULATOARELOR

- Memoria flash a MC ATmega328P este împărțită în două secțiuni:
  - Program de aplicații
    - Conține programul dezvoltat de utilizator
    - Este salvat cu ajutorul bootload-erului
- Programarea în mod serial se realizează utilizând:

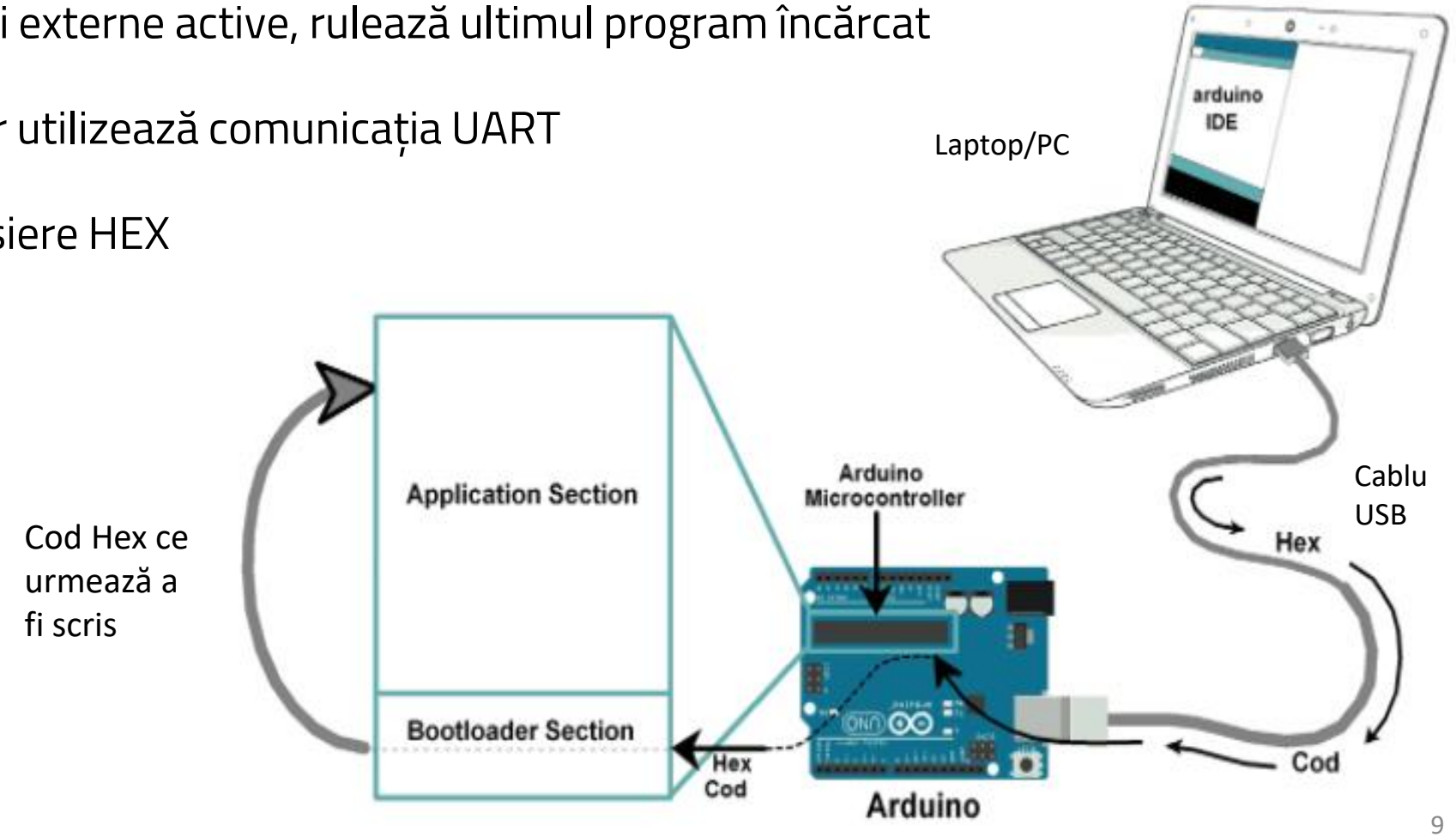
Simbol	Pin	I/O	Descriere
MOSI	PB3	I	Serial data in
MISO	PB4	O	Serial Data out
SCK	PB5	I	Serial Clock

- Microcontrolerele de pe placa Arduino sunt flash-uite cu un bootloader Arduino, astfel încât să fie compatibile cu Arduino IDE

Programarea modulelor Arduino

Universitatea Transilvania din Brașov
FACULTATEA DE INGINERIE ELECTRICĂ ȘI ȘTIINȚA CALCULATOARELOR

- Când un modul Arduino este alimentat sau Resetat, bootloader-ul Arduino rulează
- Dacă nu sunt intrări externe active, rulează ultimul program încărcat
- Arduino Bootloader utilizează comunicația UART
- Sunt transferate fișiere HEX



Programarea modulelor Arduino

Universitatea Transilvania din Brașov
FACULTATEA DE INGINERIE ELECTRICĂ ȘI ȘTIINȚA CALCULATOARELOR

- Pentru transferul către/de la Mc AVR se utilizează AVRDUDE (AVR Downloader Uploader) – pentru memorie Flash și EEPROM
- Protocolul de comunicație este STK500 (pentru a transfera fișierele hex compilate)
- Comunicația STK500 este utilizată între Avrdude (rulează pe PC/laptop) și bootloader (ruleaza pe mc avr)
- Un bootloader de Arduino poate fi compilat și încărcat în memoria flash utilizându-se Atmel Studio si USBasp (programator circuite)
- O placă Arduino poate fi folosită și ca modul programare a unui nou bootloader (pentru alte întrebuințări) – programator ISP (In-system programming)

Limbaje de programare

Universitatea Transilvania din Brașov
FACULTATEA DE INGINERIE ELECTRICĂ ȘI ȘTIINȚA CALCULATOARELOR

- Limbaje utilizate: Asamblare, C, C++, Python, Rust
- Limbajul de asamblare:
  - Implementare ușoară pentru programe de mici dimensiuni
  - Cea mai rapidă execuție a codului
  - Cel mai compact cod
  - Implementare complicată pentru programe de mari dimensiuni
  - Analiza completă a resurselor utilizate
  - Probleme la portarea codului, chiar și pe aceleași familii de MC

```
'Chip model
#chip 16F886, 4
#include <chipino.h>

'Main routine
Start:

 'Turn D13 LED on
 SET D13 ON
 Wait 1 sec

 'Now toggle the LEDs
 SET D13 OFF
 Wait 1 sec

 'Jump back to the start
Goto Start
```

Limbaje de programare

Universitatea Transilvania din Brașov
FACULTATEA DE INGINERIE ELECTRICĂ ȘI ȘTIINȚA CALCULATOARELOR

- Limbajul C:
  - Execuție rapidă a codului
  - Ușor de portat pe alte compilatoare sau familii de MC
  - Multe compilatoare disponibile
  - Multe funcții predefinite
  - Foarte răspândit ca și limbaj
  - Se pot utiliza unelte de analiza a codului (Polyspace verifica regulile MISRA)
  - Dificil de utilizat la început
- Limbajul C++:
  - Cod mai organizat și modular
  - Posibilitate de reutilizare a codului
  - Dimensiunea codului și cerințele de resurse pot fi mai mari decât în C

```
unsigned int ADCRead(unsigned char ch)
{
   int conv=0;
   if(ch>7) return 0;  //Invalid Channel

   ADCON0=(ch<<2);   //Select ADC Channel

   ADCON0.ADON=1;  //switch on the adc module

   ADCON0.GO_DONE=1;//Start conversion

   while(ADCON0.GO_DONE){} //wait for the conv

   ADCON0.ADON=0;  //switch off adc
    conv |= ADRESH << 2 | ADRESL >> 6   ;
   return conv;
}
```

Limbaje de programare

Universitatea Transilvania din Brașov
FACULTATEA DE INGINERIE ELECTRICĂ ȘI ȘTIINȚA CALCULATOARELOR

Limbajul C

```
void main() {
TRISB = 0;
PORTB = 0b00000011;
}
```

Limbajul de asamblare

```
_main:
        CLRF   TRISB+0
        MOVLW  3
        MOVWF  PORTB+0
        GOTO   $+0
```

Fișierul *.hex

```
:0A0000001828000000000000288E
:0E000A008312031321088A002008820008000D8
:1000180005208A110A128000840AA00A0319A10A7D
:08002800F003031D0C28080081
:0E00300083160313860103308312860001E28F8
:02400E00F21F9F
:00000001FF
```

Formatul unui fișier hex

Universitatea Transilvania din Brașov
FACULTATEA DE INGINERIE ELECTRICĂ ȘI ȘTIINȚA CALCULATOARELOR

- Fișierul .hex generat are o structură de forma: "BBaaAATTDDCC"
  - BB – reprezintă numărul de octeți de pe linia curentă
  - aaAA – adresa unde octeții se vor salva în memorie (aa – LSB, iar AA – MSB pentru adresă)
  - TT – reprezintă tipul de date, astfel:
    - 00 – date de tip program
    - 01 – EOF (End Of File)
    - 04 – adresă extinsă.
  - DD – octeții de date care conțin codul mașină generat din codul scris de programator (câți octeți sunt pe linie)
  - CC – Suma de control

Unitatea Centrală de Procesare - UCP

Universitatea Transilvania din Brașov
FACULTATEA DE INGINERIE ELECTRICĂ ȘI ȘTIINȚA CALCULATOARELOR

- Suma de control se calculează astfel:
  - 0A + 00 + 00 + 00 + 18 + 28 + 00 + 00 + 00 + 00 + 00 + 00 + 00 + 28 = 72 = 111 0010
  - ~ (01110010) = 10001101 + 1 = 10001110 = 8E

:0A	0000	00	18 28 00 00 00 00 00 00 00 28	8E
:0E	000A	00	8312031321088A00200882000800	D8
:10	0018	00	05208A110A128000840AA00A0319A10A	7D
:08	0028	00	F003031D0C280800	81
:0E	0030	00	8316031386010330831286001E28	F8
:02	400E	00	F21F	9F
:00	0000	01		FF //EOF

Formatul unui fișier hex

Universitatea Transilvania din Brașov
FACULTATEA DE INGINERIE ELECTRICĂ ȘI ȘTIINȚA CALCULATOARELOR

- Limbajul C vs Limbajul de asamblare:

Limbajul C

```
int nVar1 = 10;
int nVar2 = 20;
int nResult;

void main() {
    nResult = nVar1 *
nVar2;
}
```

Limbajul de asamblare

```
_main:
  MOVF        _nVar1+0, 0
  MOVWF       R0+0
  MOVF        _nVar1+1, 0
  MOVWF       R0+1
  MOVF        _nVar2+0, 0
  MOVWF       R4+0
  MOVF        _nVar2+1, 0
  MOVWF       R4+1
  CALL        _Mul_16x16_U+0
  MOVF        R0+0, 0
  MOVWF       _nResult+0
  MOVF        R0+1, 0
  MOVWF       _nResult+1
  GOTO        $+0
```

Formatul unui fișier hex

Universitatea Transilvania din Brașov
FACULTATEA DE INGINERIE ELECTRICĂ ȘI ȘTIINȚA CALCULATOARELOR

- Limbajul C vs Limbajul de asamblare:

Limbajul C

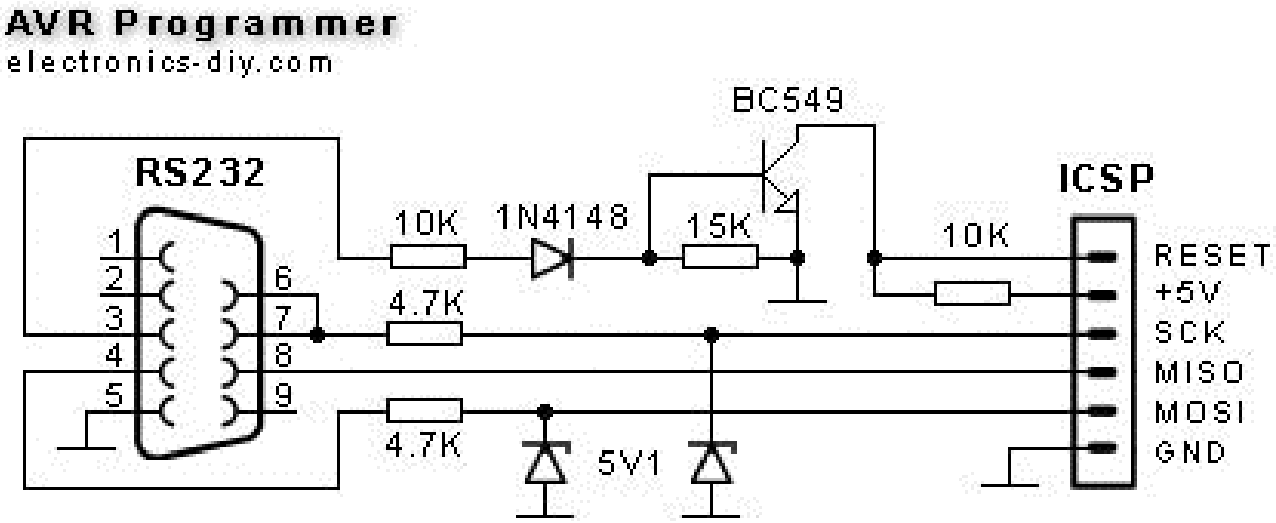
```
int nVar1 = 10;
int nVar2 = 20;
int nResult;

void main() {
    nResult = nVar1 *
nVar2;
}
```

Limbajul de asamblare

```
_main:
  MOVF        _nVar1+0, 0
  MOVWF       R0+0
  MOVF        _nVar1+1, 0
  MOVWF       R0+1
  MOVF        _nVar2+0, 0
  MOVWF       R4+0
  MOVF        _nVar2+1, 0
  MOVWF       R4+1
  CALL        _Mul_16x16_U+0
  MOVF        R0+0, 0
  MOVWF       _nResult+0
  MOVF        R0+1, 0
  MOVWF       _nResult+1
  GOTO        $+0
```

Programator și Depanatoare

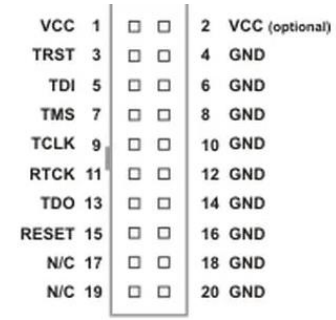
- Programatorul și depanatorul (debugger-ul) sunt instrumente esențiale pentru transferul codului compilat în microcontroler și pentru depanarea acestuia.
- Softul utilizat pentru programare este PonyProg (https://www.lancos.com/prog.html)
- Comunicația serială este utilizată pentru transferul datelor
- ICSP = In-Circuit Serial Programming
- Schema electrică a unui programator (MC din familia AVR):



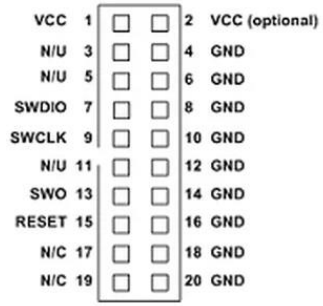
Programator și Debugger

Universitatea Transilvania din Brașov
FACULTATEA DE INGINERIE ELECTRICĂ ȘI ȘTIINȚA CALCULATOARELOR

- Din categoria programatoare și depanatoare amintim:
  - JTAG (Joint Test Action Group):
    - Un protocol standard pentru testarea și programarea circuitelor integrate
    - Utilizat pe scară largă pentru microcontrolere ARM și alte arhitecturi
    - Permite programarea și depanarea codului la nivel hardware, inclusiv setarea de breakpoints și urmărirea execuției
  - ISP (In-System Programming):
    - Permite programarea microcontrolerelor direct în sistemul lor final, fără a le scoate din circuit
    - Larg utilizat pentru microcontrolere AVR, PIC, și altele
    - Convenabil pentru actualizări de firmware în sisteme deja montate și operaționale
  - SWD (Serial Wire Debug):
    - O variantă a JTAG, specifică pentru microcontrolere ARM, care utilizează mai puține pini



JTAG



SWD

Programator și Debugger

Universitatea Transilvania din Brașov
FACULTATEA DE INGINERIE ELECTRICĂ ȘI ȘTIINȚA CALCULATOARELOR

- Un dispozitiv depanator facilitează examinarea pas cu pas a execuției unui program, identificarea erorilor (bugs) și înțelegerea comportamentului său
- Funcționalități de bază (depanator încorporat sau embedded):
  - Punct de întrerupere (breakpoint): permite programului să se opri la o anumită linie de cod pentru a inspecta variabile și registri
  - Execuție pas cu pas: permite executarea programului linie cu linie, oferind control exact asupra fluxului de execuție
  - Inspecția variabilelor: permite examinarea valorilor variabilelor în timpul execuției
  - Vizualizarea memoriei: permite examinarea conținutului memoriei, inclusiv a stivei și a heap-ului
  - Urmărirea apelurilor de funcție: permite urmărirea apelurilor de funcție și revenirea din ele

Concluzii

Universitatea Transilvania din Brașov
FACULTATEA DE INGINERIE ELECTRICĂ ȘI ȘTIINȚA CALCULATOARELOR

- Etapele necesare obținerii unui fișier binar și rulării acestuia pe un MC
  - Compilare – linkeditare – programare – execuție
- Structura memoriei unui MC AVR
  - Bootloader și memorie pentru aplicații
- Programarea unui modul Arduino
- Limbaje de programare
  - Avantaje și dezavantaje ale principalelor limbaje
- Verificarea funcționarii unui MC
  - Utilizarea unui depanator pentru evaluare

Contact:
Email: gigel.macesanu@unitbv.ro
elearning.unitbv.ro - Sisteme cu Microprocesoare