CUPRINS

A – Activități de predare

A1 - Sisteme mecatronice
Conceptul de mecatronică
Structura hard a unui sistem mecatronic
Microcontrolere

A2 - Interfațarea sistemelor periferice
Porturi, modalități de scriere şi citire
Afişajul cu 7 segmente
Butoane

A3 - Interfațatea senzorilor şi comunicații de date
Comunicația serială. Codificarea datelor şi controlul erorilor
Controlerul de comunicație USART
Afişajul alfanumeric cu cristale lichide LCD
Convertorul analog – digital (ADC)
Interfațarea senzorilor şi traductoarelor. Clasificare
Traductoare şi senzori de poziție. Potențiometrul
Senzori incrementali
Traductoare de lumină. Fotorezistența.
Traductoare de temperatură. Termistorul

A4 - Actuatori
Noțiuni generale.Sistematizarea actuatorilor utilizați in mecatronică
Motorul de curent continuu.
Interfațarea motoarelor de curent continuu la microcontroler.
Întreruperi. Tratarea intreruperilor.
Principiul de funcționare al unui Timer.
Semnale PWM.
Motorul pas cu pas
Servomecanismul

A5 - Algoritmi de control
Generalități.
Controlul în buclă deschisă.
Controlul în buclă închisă. Clasificarea regulatoarelor.
Analiza sistemelor de control în buclă închisă
Controlerul bipozițional
Controlerul proporțional (P)
Controlerele PI şi PD
Controlerul PID. Acordarea controlerului PID

B - Aplicații

B2 - Interfațarea sistemelor periferice

Comanda LED-urilor. Astabili cu două sau mai multe LED-uri.
Semafor cu LED-uri
Afisarea unei cifre. Astabil cu două cifre. Afişarea multiplexată
Buton cu LED. Numărător 0...9 cu buton

B3 - Interfațatea senzorilor şi comunicații de date

Trimiterea şi recepționarea unui caracter cu ajutorul interfeței seriale.
Afişarea textelor pe LCD.
Afişarea pe LCD a textelor receptionate pe interfața serială
Citirea ADC-ului şi afişarea valorii pe 8 LED-uri
Citirea ADC-ului şi afişarea valorii pe LCD
Interfațatea traductorului potențiometric
Interfațarea encoderului incremental
Citirea valorii unei fotorezistențe
Citirea valorii unui termistor NTC

B4 - Actuatori

Comanda bidirecțională a motorului de curent continuu
Variator de turație pentru motorul de curent continuu
Comanda motorului pas cu pas cu turație variabilă
Comanda servomecanismelor
Comanda unui braț robotic cu trei grade de mobilitate

B5 – Algoritmi de control

Studiul controlerelor P şi PID
Controlul poziției unui motor de curent continuu
Comanda deplasării in linie dreaptă a unei platformei mobile

Organizarea activităților

	A - Activitati teoretice	B - Activitati aplicative
Ziua 1	**A1, A2**	**B1, B2**
Ziua 2	**A2 , A3**	**B2, B3**
Ziua 3	**A3**	**B3**
Ziua 4	**A4**	**B4**
Ziua 5	**A5**	**B5**
Ziua 6	**Evaluare**	

UNIUNEA EUROPEANĂ
GUVERNUL ROMÂNIEI MINISTERUL MUNCII, FAMILIEI ŞI PROTECŢIEI SOCIALE AMPOSDRU
Fondul Social European POS DRU 2007-2013
Instrumente Structurale 2007-2013
MINISTERUL EDUCAŢIEI CERCETĂRII TINERETULUI ŞI SPORTULUI
OIPOSDRU
UNIVERSITATEA TEHNICĂ DIN CLUJ-NAPOCA

FlexFORM **Program de formare profesională flexibilă pe platforme mecatronice**

Modulul 2. Platforme mecatronice pentru educaţie şi formare flexibilă

Tehnologie şi educație mecatronică. Dezvoltare durabilă


OBIECTIV

❑ Exploatarea platformelor educaționale în scopul stimulării transferului de cunoaștere, a creativitatii, a gândirii integratoare, a flexibilității și capacității de adaptare a cursantului, respectiv elevului pentru a răspunde cu operativitate nevoilor în continuă schimbare ale pieței muncii (vizează exploatarea/programarea unor sisteme mecatronice modulare utile în sustinerea activităților de curs și laborator, pentru diverse discipline: fizică, mecatronică, electronică, informatică, tehnologii etc.)

COMPETENȚE GENERALE

❑ Cunoașterea și înțelegerea conceptelor fundamentale impuse de tehnologia mecatronică.
❑ Exploatarea platformelor educaționale în scopul stimulării transferului de cunoaștere a capacității de adaptare a cursantului la dinamica pieței muncii (programarea funcționalității sistemelor mecatronice, cunoașterea, dezvoltarea și implementarea tehnicilor de control)
❑ Gândirea sistemică, integratoare și deprinderea de a lucra în echipă

FlexFORM - Program de formare profesionala flexibila pe platforme mecatronice

OBIECTIV

❑ Exploatarea platformelor educaţionale în scopul stimulării transferului de cunoaştere, a creativitatii, a gândirii integratoare, a flexibilităţii şi capacităţii de adaptare a cursantului, respectiv elevului pentru a răspunde cu operativitate nevoilor în continuă schimbare ale pieţei muncii (vizează exploatarea/programarea unor sisteme mecatronice modulare utile în sustinerea activităţilor de curs şi laborator, pentru diverse discipline: fizică, mecatronică, electronică, informatică, tehnologii etc.)

COMPETENŢE GENERALE

❑ Cunoaşterea şi înţelegerea conceptelor fundamentale impuse de tehnologia mecatronică.

❑ Exploatarea platformelor educaţionale în scopul stimulării transferului de cunoaştere a capacităţii de adaptare a cursantului la dinamica pieţei muncii (programarea funcţionalităţii sistemelor mecatronice, cunoaşterea, dezvoltarea şi implementarea tehnicilor de control)

❑ Gândirea sistemică, integratoare şi deprinderea de a lucra în echipă

Conceptul de mecatronică

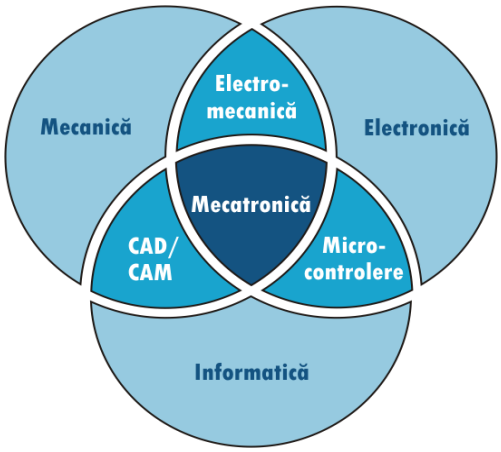


Conceptul de mecatronică s-a născut în Japonia la începutul deceniului al optulea al secolului trecut. Termenul în sine a fost brevetat de către concernul Yaskawa Electric Co. şi a fost utilizat pentru a descrie fuziunea tehnologică: mecanică - electronică – informatică.

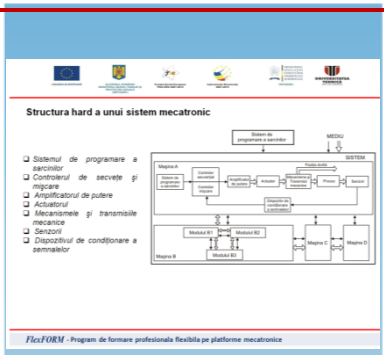
Coloana vertebrală a mecatronicii o constituie tehnologia mecanică, care s-a dezvoltat către mecanizare. Progresele în domeniul tehnologiei electronice, apariţia circuitelor integrate, mici ca dimensiuni, ieftine şi fiabile, au permis includerea electronicii în structurile mecanice. Se realizează astfel integrarea electromecanică.

Următorul pas în integrare a fost determinat de apariţia microprocesoarelor. Cu aceleaşi caracteristici constructive, ca şi circuitele integrate, microprocesoarele au putut fi integrate în structurile electromecanice realizate anterior.

Mecatronica este tehnologia mecanică cerută de societatea informaţională. În tehnologia tradiţională, elementele de bază sunt materialul şi energia. În mecatronică, acestor două elemente li se adaugă informaţia. Practic, problemele privind informaţia vizează: culegerea, prelucrarea, stocarea sau transmiterea acesteia, şi folosirea ei pentru a controla produsele, procesele şi sistemele.



Structura hard a unui sistem mecatronic



Sistemul de programare a sarcinilor, generează mişcările dorite şi secvenţele acestora în concordanţă cu cerinţele sau comenzile transmise. Acest modul este materializat de microcontrolere

Controlerul de secveţe şi mişcare compară parametrii curenţi ai mişcării cu cei impuşi şi realizează corecturile necesare. Cel mai des întâlnit algoritm de control este cel **P**roporţional-**I**ntegral-**D**erivativ (PID).

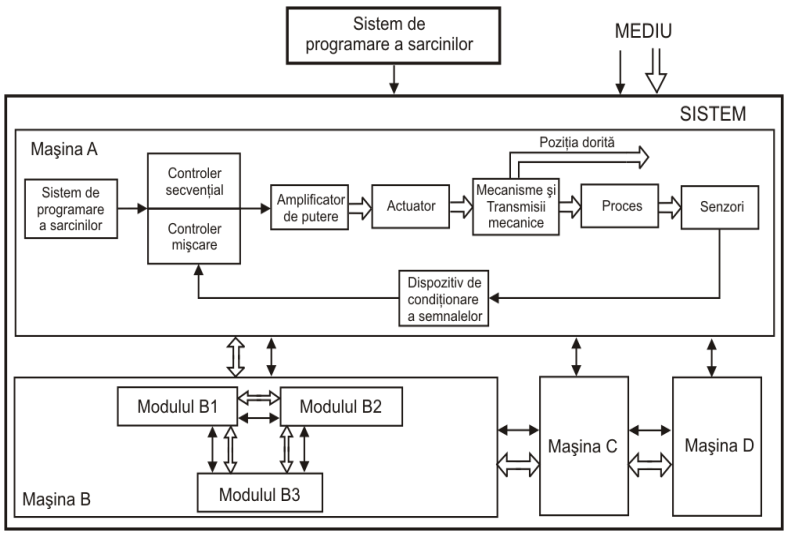
Amplificatorul de putere amplifică semnalul în concordanţă cu cerinţele actuatorului;

Actuatorul transformă semnalul corectat în semnal de intrare (moment, forţă, viteză) în acord cu cerinţele procesului;

Mecanismele şi transmisiile mecanice realizează adaptarea parametrilor actuatorului la cerinţele impuse de procesul tehnologic

Senzorii prelucrează informaţii privind parametrii procesului şi transmit semnale corespunzătoare spre controlerul mişcării;

Dispozitivul de condiţionare a semnalelor cuprinde filtre, amplificatoare, etc. care, prelucrează semnalele în concordanţă cu cerinţele impuse de intrarea în controlerul mişcării.



Microcontrolere

Un microcontroler este în esenţă o configuraţie minimală de sistem de calcul, capabil să execute la o viteză foarte mare instrucţiunile unui program stocat în memorie; acest program este o secvenţă logică de operaţii ce poate implementa algoritmii necesari pentru controlul proceselor.

Microcontrolerul are integrat pe acelaşi chip: oscilatorul, memoria (RAM, ROM, EEPROM), numărătoare, blocuri analogice, interfeţe de comunicaţie şi porturi de intrare-ieşire.

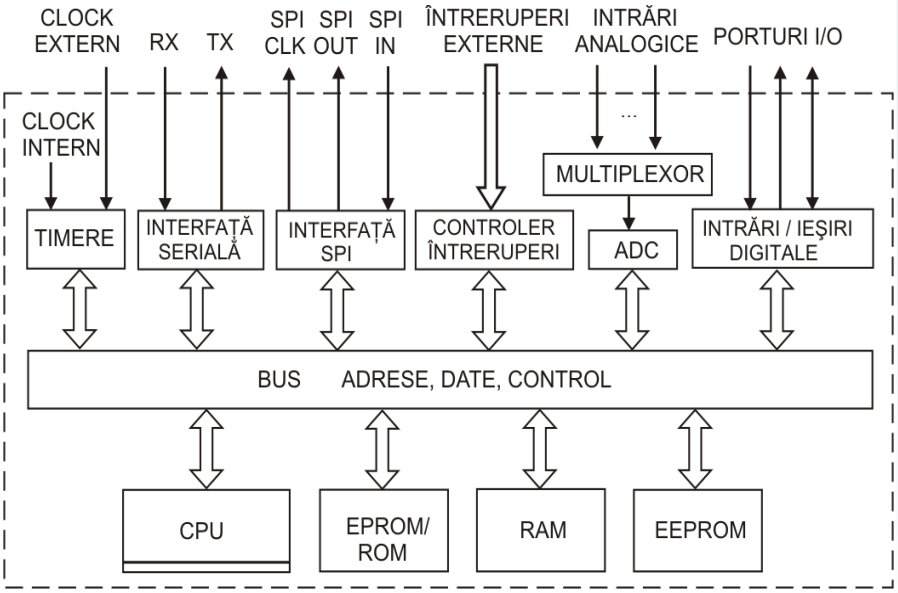
Microcontrolerele sunt caracterizate prin:

- dimensiune redusă a memoriei program şi a memoriei de date;

- conţin module pentru interfaţarea atât digitală cât şi analogică cu senzori şi actuatori;

- răspund rapid la evenimente externe;

- sunt de o mare varietate pentru a putea fi satisfăcute cerinţele diverselor aplicaţii la un raport preţ / performanţă corespunzător.



Ceea ce deosebeşte fundamental un microcontroler de un circuit integrat analogic sau digital, este faptul că el nu poate face nimic dacă nu este programat. Programul software conferă microcontrolerului, abilitatea de a realiza funcţii diferite cu aceeaşi configuraţie hardware .

Scrierea programului se realizează de obicei intr-un editor ce permite salvarea liniilor de comandă introduse. Există mai multe opţiuni pentru scrierea programului de control al aplicatiei şi anume:

- cod masină (cod hexadecimal)

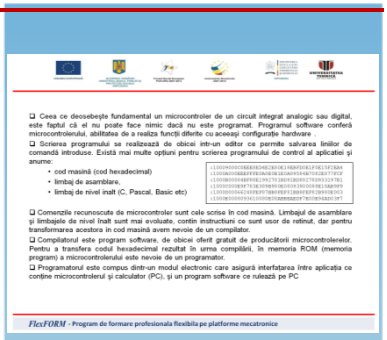
```
:10009000C0EEE8ED4E2E80E18EBFD0E1F0E15F2EA4
:1000A000EEEFFFE0A0E0B1E0A89584B7082E877FCF
:1000B00084BF88E1992781BD91BD88278D933197B1
:1000C000E9F783E389B980E08093900088E18AB9F9
:1000D00066248FEF87BB8FEF81BB8FEF82B980E003
:1000E00080936100080E08ABBEAEDF7E00E94AD03F7
```

- limbaj de asamblare,
- limbaj de nivel inalt (C, Pascal, Basic etc)

Comenzile recunoscute de microcontroler sunt cele scrise în cod masină. Limbajul de asamblare şi limbajele de nivel înalt sunt mai evoluate, contin instructiuni ce sunt usor de retinut, dar pentru transformarea acestora in cod masină avem nevoie de un compilator.

Compilatorul este program software, de obicei oferit gratuit de producătorii microcontrolerelor. Pentru a transfera codul hexadecimal rezultat în urma compilării, în memoria ROM (memoria program) a microcontrolerului este nevoie de un programator.

Programatorul este compus dintr-un modul electronic care asigură interfaţarea între aplicaţia ce conţine microcontrolerul şi calculator (PC), şi un program software ce rulează pe PC



Microcontrolerul ATMega16

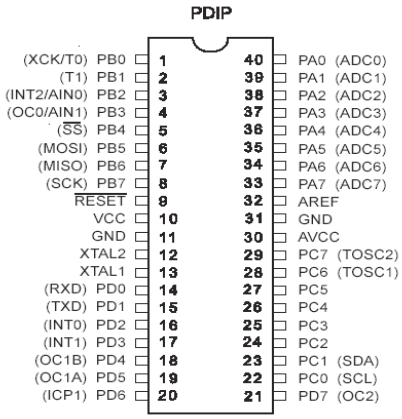


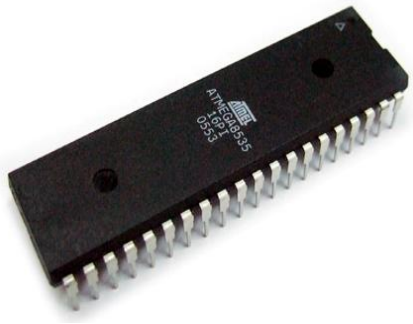
Famila AVR de la Atmel este formată din microcontrolere cu arhitectura pe 8 biti şi set redus de instructiuni (RISC).

Memoriile ROM, EEPROM şi SRAM sunt integrate în acelaşi chip, înlăturând nevoia de memorie externă.

Cifra din numele microcontrolerului indică mărimea memoriei de program (ROM); de exemplu ATMega**16** are **16**kB de memorie ROM

Majoritatea instructiunilor se execută într-un singur ciclu de ceas



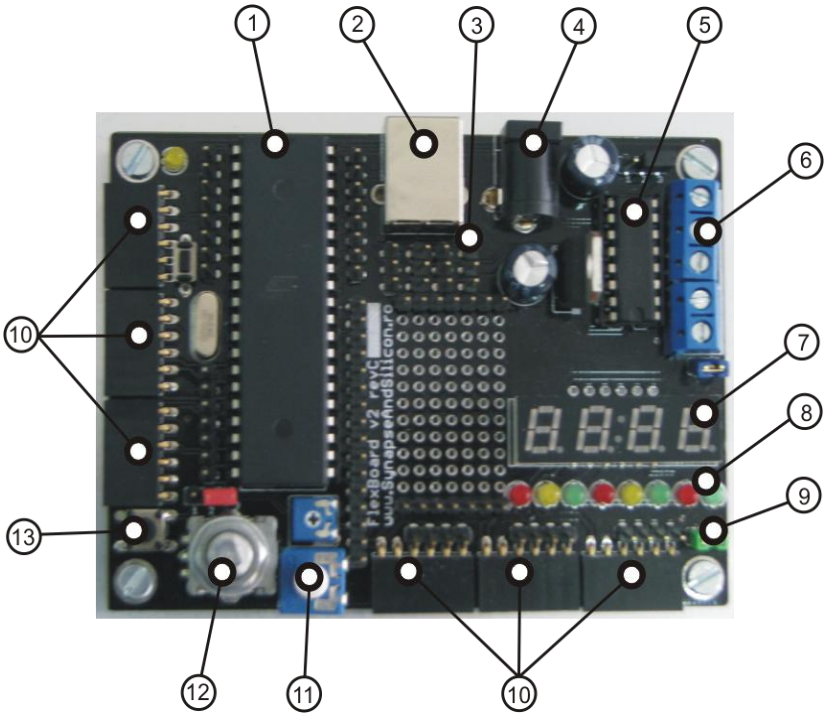


Caracteristici:

- tensiune alimentare 2,7...5 Vcc
- frecvenţa max. 16Mhz
- 512 bytes SRAM
- 16K bytes ROM
- 512 bytes EEPROM
- ADC 10biţi cu 8 intrări
- 4 porturi I/O pe 8 biţi
- 3 numărătoare: două pe 8 biţi şi unul pe 16 biţi
- 3 întreruperi externe
- 1 interfaţă comunicare serială
- 1 interfaţă SPI
- 1 interfaţa I^2C

Placa de dezvoltare





1 – Microcontroler ATMega16

2 – Conector USB pt comunicaţie şi/sau programare

3 – Conector servomotoare

4 – Mufă alimentare

5 – Circuit integrat L293

6 – Conector motoare

7 – Afişaj 7 segmente

8 – LED-uri

9 – Jumper activare afişaj 7 segmente

10 – Conectori module de expansiune

11 – Potenţiometru semireglabil

12 – Encoder

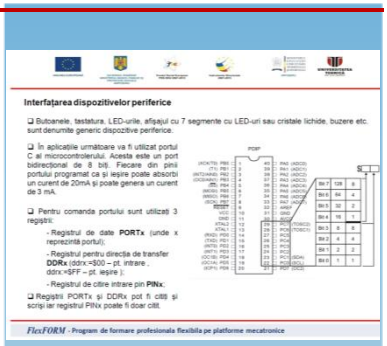
13 – Push button

INTERFAȚAREA DISPOZITIVELOR PERIFERICE

OBIECTIVE

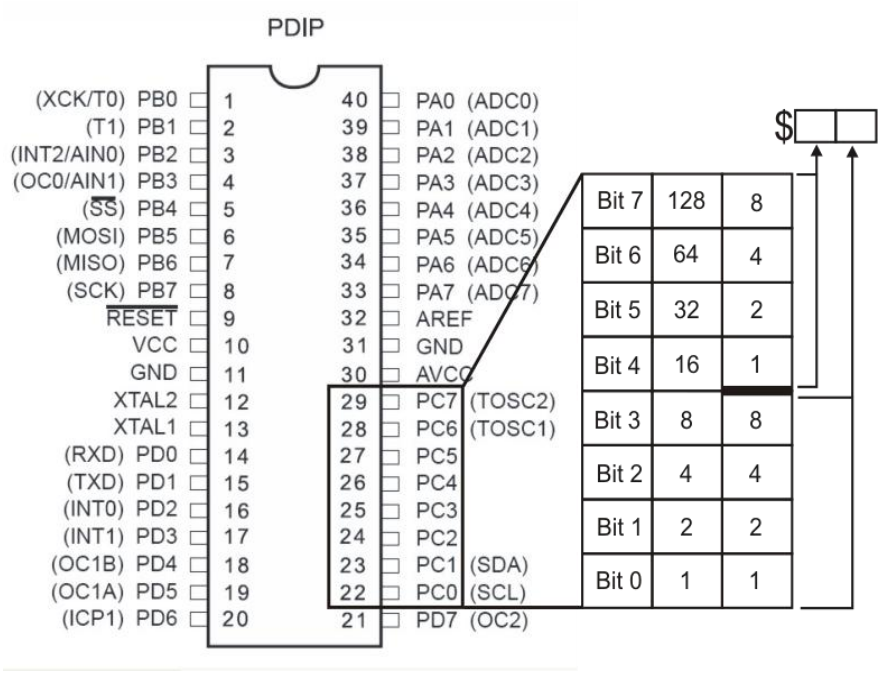
- Familiarizarea cu mediul de programare MikroPascal ®;
- Programarea porturilor de intrare/ieşire a microcontrolerului ;
- Utilizarea instrucțiunilor ***if , while*** şi ***for*** ;
- Afişarea unor stări utilizând LED-uri şi afişaje cu 7 segmente;
- Utilizarea butoanelor pentru introducerea comenzilor.

Interfaţarea dispozitivelor periferice



Butoanele, tastatura, LED-urile, afişajul cu 7 segmente cu LED-uri sau cristale lichide, buzere etc. sunt denumite generic dispozitive periferice.

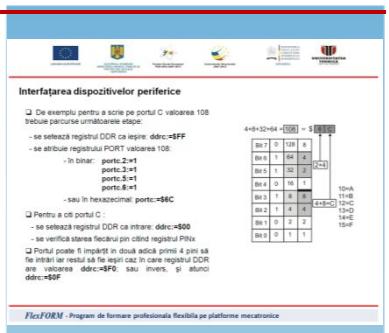
În aplicaţiile următoare, va fi utilizat portul C al microcontrolerului. Acesta este un port bidirecţional de 8 biţi. Fiecare din pinii portului programat ca şi ieşire poate absorbi un curent de 20mA şi poate genera un curent de 3 mA.



Pentru comanda portului sunt utilizaţi 3 regiştrii:

- Registrul de date **PORTx** (unde x reprezintă portul);
- Registrul pentru direcţia de transfer **DDRx** (ddrx:=$00 – pt. intrare , ddrx:=$FF – pt. ieşire);
- Registrul de citire intrare pin **PINx**;

Regiştrii PORTx şi DDRx pot fi citiţi şi scrişi iar registrul PINx poate fi doar citit.

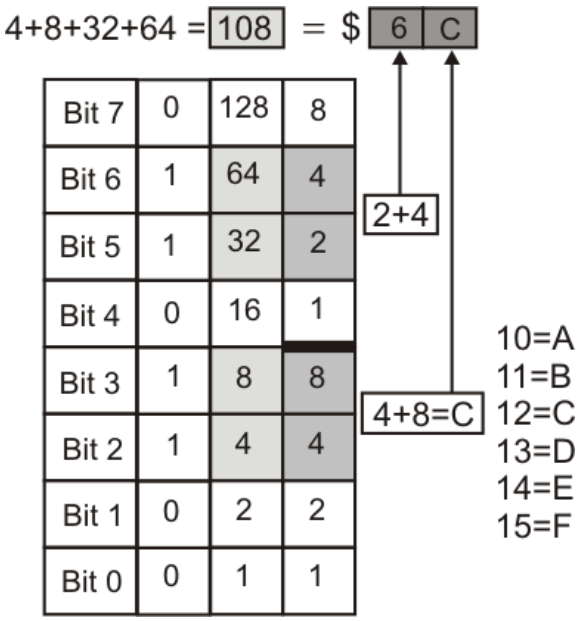


De exemplu pentru a scrie pe portul C valoarea 108 trebuie parcurse următoarele etape:

- se setează registrul DDR ca ieşire: **ddrc:=$FF**
- se atribuie registrului PORT valoarea 108:
  - în binar: **portc.2:=1**
    portc.3:=1
    portc.5:=1
    portc.6:=1
  - sau în hexazecimal: **portc:=$6C**

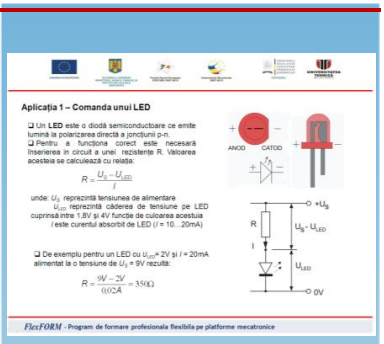
Pentru a citi portul C :

- se setează registrul DDR ca intrare: **ddrc:=$00**
- se verifică starea fiecărui pin citind registrul PINx



Portul poate fi impărţit in două adică primii 4 pini să fie intrări iar restul să fie ieşiri caz în care registrul DDR are valoarea **ddrc:=$F0**; sau invers şi atunci **ddrc:=$0F**

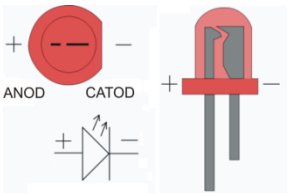
Aplicaţie - Comanda unui LED



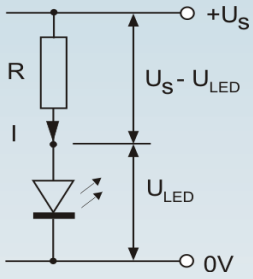
Cel mai simplu mod de a urmări starea logică a unui port este legarea unor diode LED pe pinii corespunzători portului respectiv. LED-ul se va aprinde dacă bitul are valoarea logică “1”, corespunzatoare unei tensiuni de 5V şi va fi stins pentru starea logica “0”.

Generalităţi

Diodele LED sunt dispozitive realizate prin doparea unui element semiconductor pentru a realiza o jonctiune p-n. La trecerea unui curent electric prin această jonctiune se eliberează energie sub formă de fotoni. LED-urile emit într-o bandă foarte restransă de lungimi de undă, facând ca lumina generată sa aibă o singură culoare. Lungimea de undă a luminii emise poate sa varieze în funcţie de tipul semiconductorului şi de modul în care se realizează doparea. Astfel se pot fabrica LED-uri care să emită în orice lungime de undă a spectrului, de la infraroşu la ultraviolet.



Cel mai simplu mod de polarizare este inserierea LED- ului cu o rezistenta. Valoarea acesteia se calculează cu relaţia:



$$R = \frac{U_S - U_{LED}}{I}$$

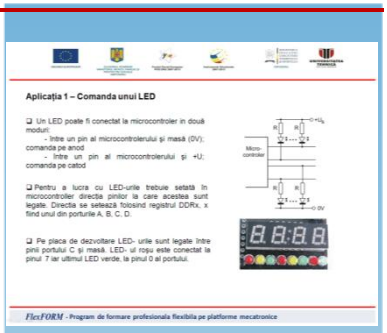
unde: U_S reprezintă tensiunea de alimentare

U_{LED} reprezintă căderea de tensiune pe LED cuprinsă între 1,8V şi 4V funcţie de culoarea acestuia

I este curentul absorbit de LED (I = 10…20mA)

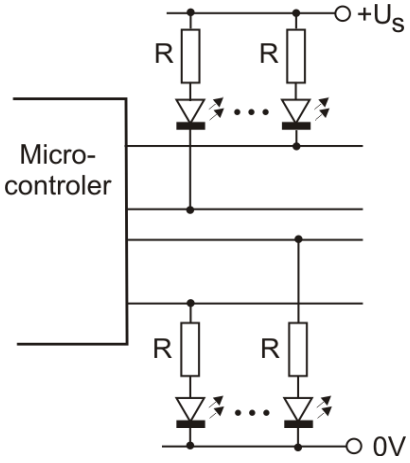
Exemplu: pentru un LED cu U_{LED}= 2V şi I = 20mA alimentat la o tensiune de U_S = 9V rezultă:

$$R = \frac{9V - 2V}{0{,}02A} = 350\Omega$$



Un LED poate fi conectat la microcontroler in două moduri:

- între un pin al microcontrolerului şi masă (0V); comanda pe anod
- între un pin al microcontrolerului şi +U; comanda pe catod

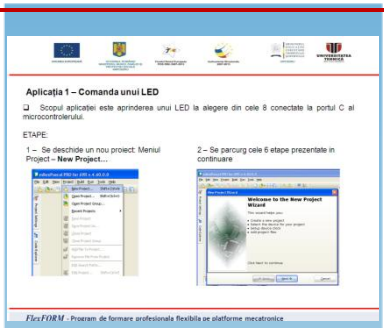


Pentru a putea comanda aprinderea LED-urilor, trebuie mai întâi setată în microcontroler direcţia pinilor la care acestea sunt legate. Directia se setează folosind registrul DDRx, x fiind unul din porturile A, B, C, D

Pe placa de dezvoltare LED- urile sunt legate între pinii portului C şi masă. LED- ul roşu este conectat la pinul 7 iar ultimul LED verde, la pinul 0 al portului.



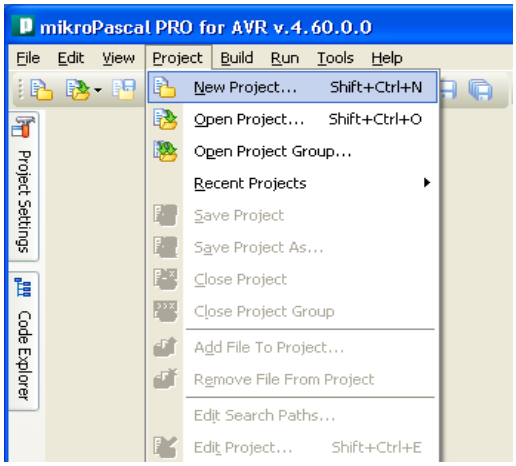
Aplicaţie - Comanda unui LED



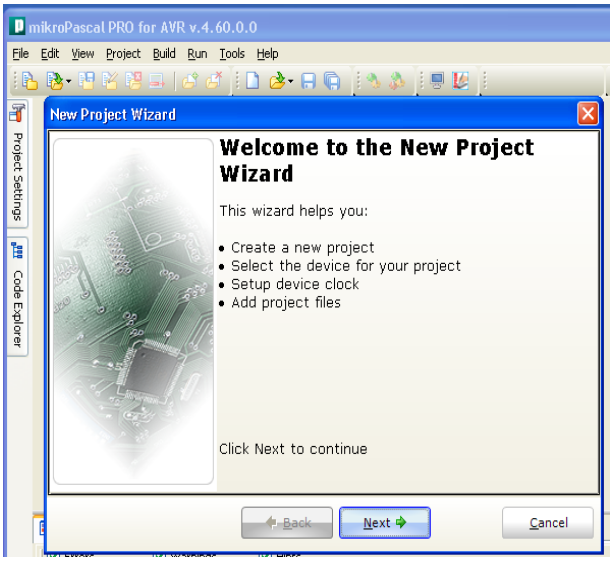
Scopul aplicaţiei este aprinderea unui LED la alegere din cele 8 conectate la portul C al microcontrolerului.

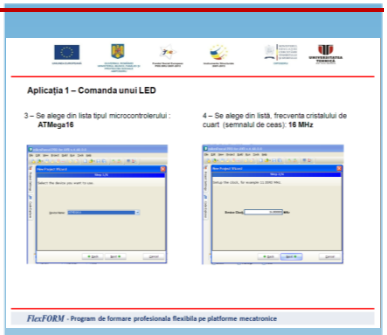
ETAPE:

1 – Se deschide un nou proiect: Meniul Project **– New Project…**



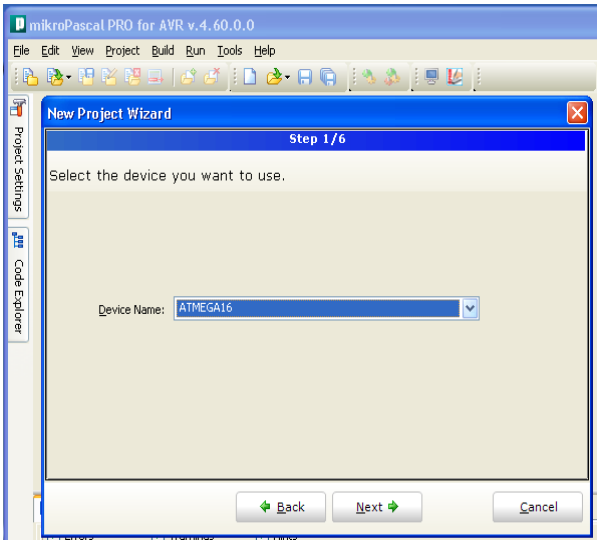
2 – Se parcurg cele 6 etape prezentate in continuare



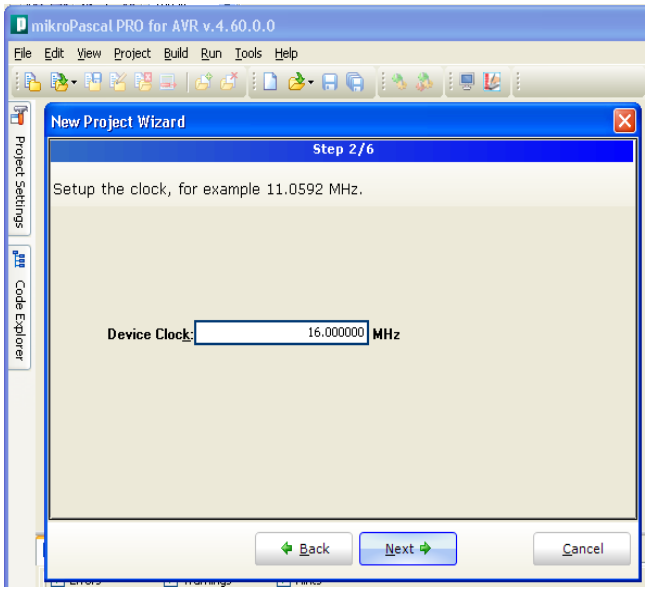


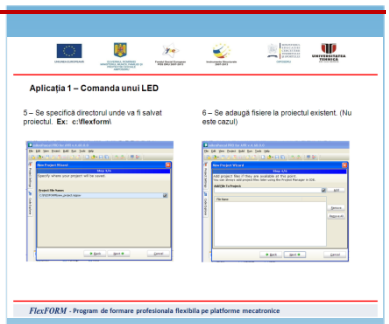
ETAPE (continuare):

3 – Se alege din listă tipul microcontrolerului : **ATMega16**



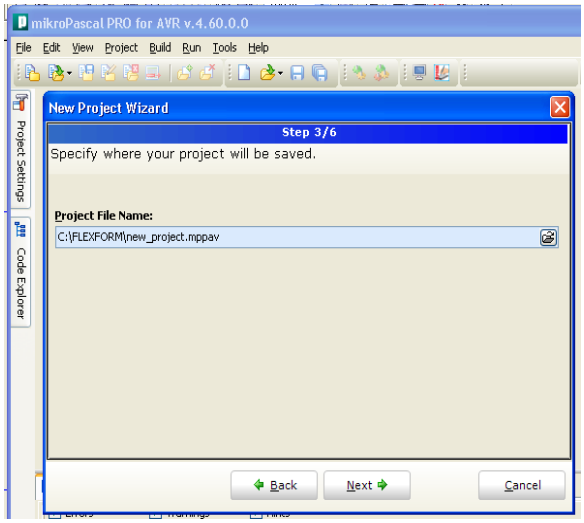
4 – Se specifică, frecventa cristalului de cuart (semnalul de ceas): **16 MHz**



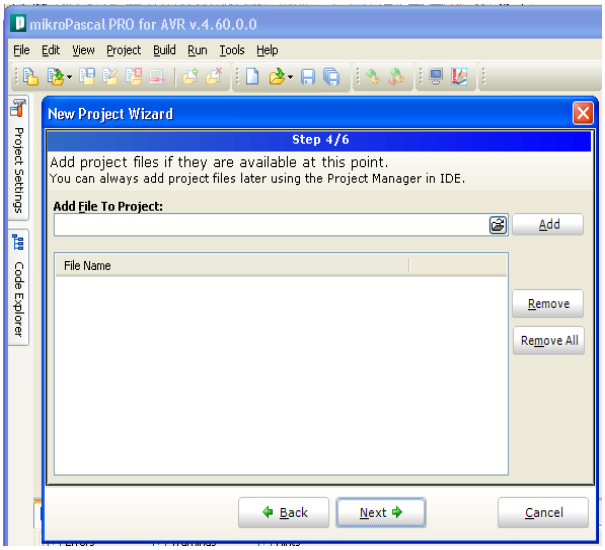


ETAPE (continuare):

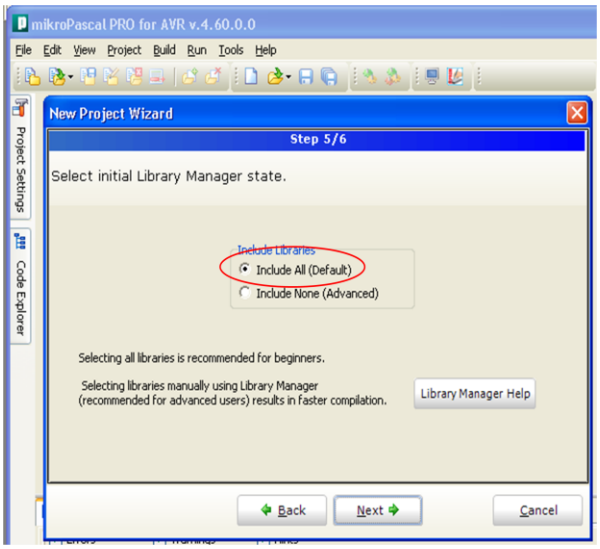
5 – Se specifică directorul unde va fi salvat proiectul. De exemplu: **c:\flexform**



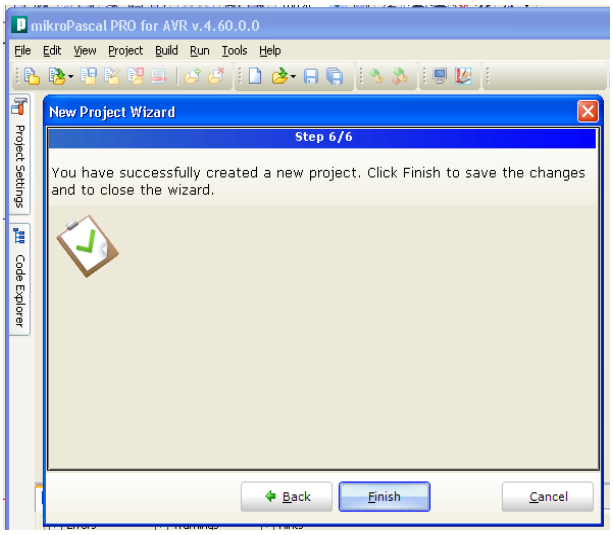
6 – Se adaugă fisiere noi la proiectul existent. (Nu este cazul)

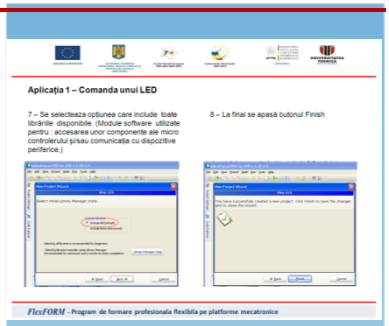


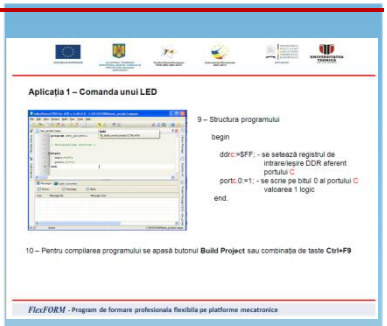
7 – Se selectează opţiunea de a include toate librăriile disponibile. (Module software utilizate pentru: accesarea unor componente ale microcontrolerului şi/sau comunicaţia cu dispozitive periferice, memorii, dispozitive de afişare etc)



8 – La final se apasă butonul Finish

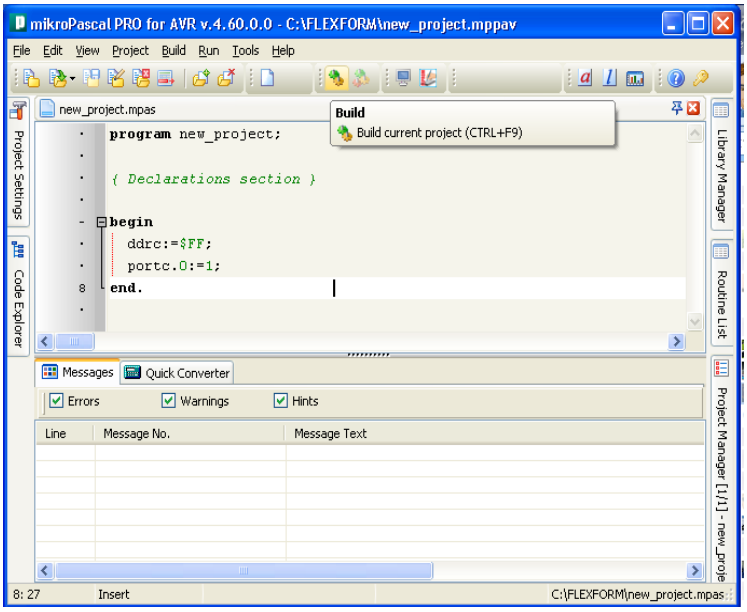






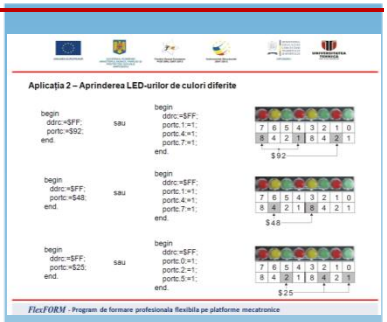
9 – Structura programului

```
begin
  ddrc:=$FF;     - se setează registrul de
                   intrare/ieşire DDR aferent
                   portului C
  portc.0:=1;    - se scrie pe bitul 0 al portului C
                   valoarea 1 logic
end.
```



10 – Pentru compilarea programului se apasă butonul **Build Project** sau combinaţia de taste **Ctrl+F9**

Aplicaţia 2 – Aprinderea LED-urilor de culori diferite

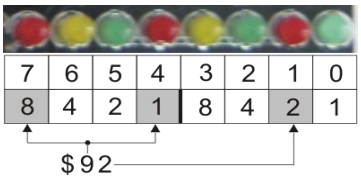


Structura programului este următoarea:

1. Pentru LED-urile roşii

```
begin
   ddrc:=$FF;
   portc:=$92;
end.
```

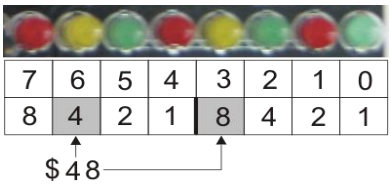
```
begin
   ddrc:=$FF;
   portc.1:=1;
   portc.4:=1;
   portc.7:=1;
end.
```



2. Pentru LED-urile galbene

```
begin
   ddrc:=$FF;
   portc:=$48;
end.
```

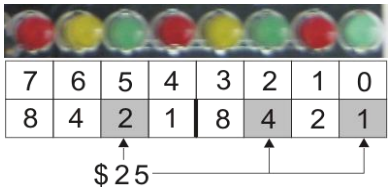
```
begin
   ddrc:=$FF;
   portc.3:=1;
   portc.6:=1;
end.
```



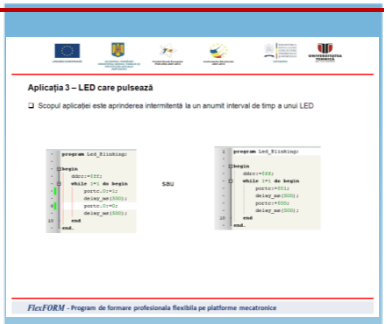
2. Pentru LED-urile verzi

```
begin
   ddrc:=$FF;
   portc:=$25;
end.
```

```
begin
   ddrc:=$FF;
   portc.0:=1;
   portc.2:=1;
   portc.5:=1;
end.
```



Aplicaţia 3 – LED care pulsează



Scopul aplicaţiei este aprinderea intermitentă la un anumit interval de timp a unui LED .

Structura programului este următoarea

- se setează portul C ca ieşire cu instrucţiunea

  ddrc := $FF

- cât timp condiţia „1=1” este îndeplinită, se execută umătorul ciclu: pinul 0 al portului C rămâne în 1 logic o perioadă de timp de 500ms după care trece în 0 logic, tot 500ms.
- efectul este aprinderea intermitentă la un interval de 500ms a LED-ului conectat la pinul 0 al portului C.
- Starea pinului 0 poate fi schimbată utilizând instrucţiunea:

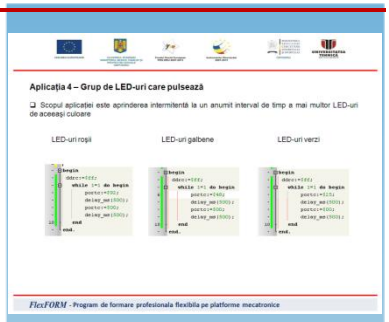
  portc.0 = 1 respectiv **portc.0 = 0**

```
begin
    ddrc:=$ff;
    while 1=1 do begin
        portc.0:=1;
        delay_ms(500);
        portc.0:=0;
        delay_ms(500);
    end
end.
```

sau: **portc = $01** respectiv **portc = $00**

```
begin
    ddrc:=$ff;
    while 1=1 do begin
        portc:=$01;
        delay_ms(500);
        portc:=$00;
        delay_ms(500);
    end
end.
```

Aplicaţia 4 – Grup de LED-uri care pulsează



Scopul aplicaţiei este aprinderea intermitentă la un anumit interval de timp a mai multor LED-uri de aceeaşi culoare.

Structura programului este aceeaşi ca şi în exemplul precedent.

1. LED-uri roşii

```
begin
  ddrc:=$ff;
    while 1=1 do begin
        portc:=$92;
        delay_ms(500);
        portc:=$00;
        delay_ms(500);
    end
end.
```

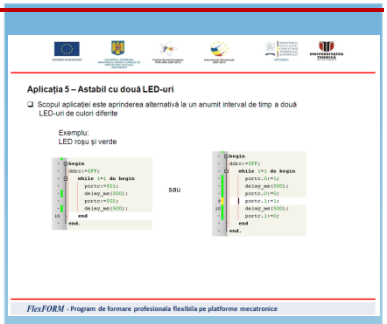
2. LED-uri galbene

```
begin
  ddrc:=$ff;
    while 1=1 do begin
        portc:=$48;
        delay_ms(500);
        portc:=$00;
        delay_ms(500);
    end
end.
```

3. LED-uri verzi

```
begin
  ddrc:=$ff;
    while 1=1 do begin
        portc:=$25;
        delay_ms(500);
        portc:=$00;
        delay_ms(500);
    end
end.
```

Aplicaţia 5 – Astabil cu două LED-uri



Scopul aplicaţiei este aprinderea alternativă la un anumit interval de timp a două LED-uri de culori diferite.

De exemplu, pentru a aprinde alternativ un LED roşu şi unul verde programul poate fi scris astfel:

```
begin
ddrc:=$FF;
   while 1=1 do begin
     portc:=$01;
     delay_ms(500);
     portc:=$02;
     delay_ms(500);
   end
end.
```

sau

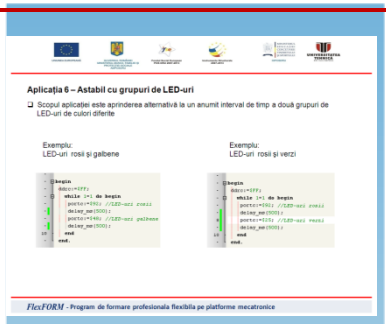
```
begin
ddrc:=$FF;
   while 1=1 do begin
     portc.0:=1;
     delay_ms(500);
     portc.0:=0;
     portc.1:=1;
     delay_ms(500);
     portc.1:=0;
   end
end.
```

Temă:

1. Scrieţi programul pentru a aprinde alternativ la un interval de 800 ms un LED roşu şi unul galben

2. Scrieţi programul pentru a aprinde alternativ la un interval de 1 s un LED verde şi unul galben

Aplicaţia 6 – Astabil cu grupuri de LED-uri



Scopul aplicaţiei este aprinderea alternativă la un anumit interval de timp a două grupuri de LED-uri de culori diferite

Pentru a aprinde alternativ LED-uri roşii şi galbene programul poate fi scris astfel:

```
begin
  ddrc:=$FF;
    while 1=1 do begin
      portc:=$92; //LED-uri rosii
      delay_ms(500);
      portc:=$48; //LED-uri galbene
      delay_ms(500);
    end
end.
```

Pentru LED-uri roşii şi verzi:

```
begin
  ddrc:=$FF;
    while 1=1 do begin
      portc:=$92; //LED-uri rosii
      delay_ms(500);
      portc:=$25; //LED-uri verzi
      delay_ms(500);
    end
end.
```

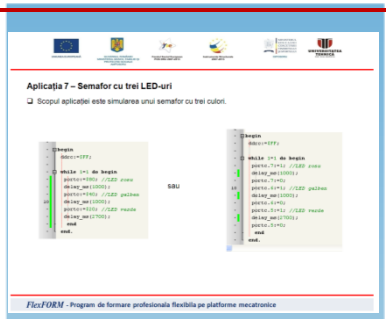
Temă:

1. Scrieţi programul pentru a aprinde alternativ la un interval de 800 ms un LED-uri galbene şi verzi.

2. Scrieţi programul pentru a aprinde alternativ la un interval de 600 ms un LED-uri roşii şi un LED galben

3. Scrieţi programul pentru a aprinde alternativ la un interval de 1 s un LED roşu şi două verzi

Aplicaţia 7 – Semafor cu trei LED-uri



Scopul aplicaţiei este simularea unui semafor cu trei culori. Culorile roşu şi galben rămân aprinse 1 secundă iar culoarea verde 2,7 secunde, după care ciclul se repetă.

Programul poate fi scris după cum urmează:

```
begin
  ddrc:=$FF;

 while 1=1 do begin
  portc:=$80; //LED rosu
  delay_ms(1000);
  portc:=$40; //LED galben
  delay_ms(1000);
  portc:=$20; //LED verde
  delay_ms(2700);
   end
 end.
```

sau:

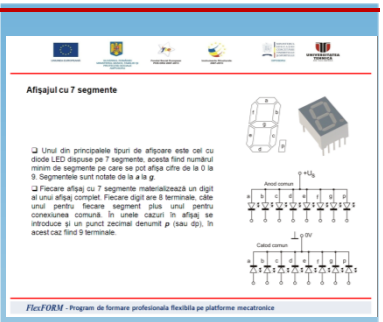
```
begin
  ddrc:=$FF;

 while 1=1 do begin
  portc.7:=1; //LED rosu
  delay_ms(1000);
  portc.7:=0;
  portc.6:=1; //LED galben
  delay_ms(1000);
  portc.6:=0;
  portc.5:=1; //LED verde
  delay_ms(2700);
  portc.5:=0;
   end
 end.
```

Temă:

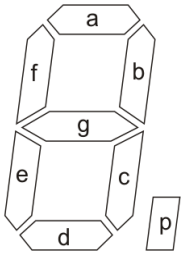
Scrieţi programul pentru simularea unui semafor cu trei culori la care culoarea roşie rămâne aprinsă 2 secunde, culoarea galbenă se aprinde intermitent 1,5 secunde, iar culoarea verde rămâne aprinsă 4 secunde.

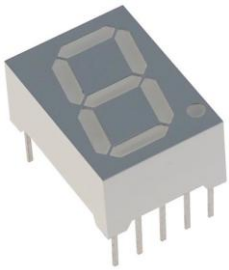
Afişajul cu 7 segmente



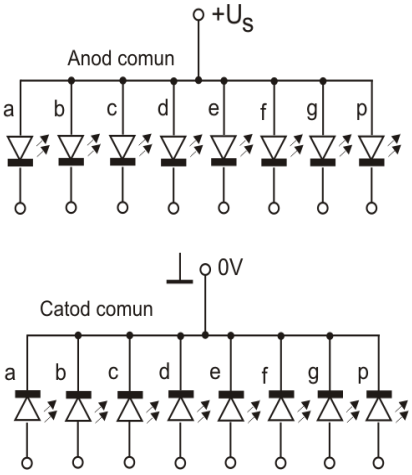
Unul din principalele tipuri de afişoare este cel cu diode LED dispuse pe 7 segmente, acesta fiind numarul minim de segmente pe care se pot afisa cifre de la 0 la 9. Segmentele sunt notate cu litere de la ***a*** la ***g***.

Fiecare afişaj cu 7 segmente materializează un digit al unui afişaj complet. Fiecare digit are 8 terminale, câte unul pentru fiecare segment plus unul pentru conexiunea comună. În unele cazuri în afişaj se introduce si un punct zecimal denumit ***p*** (sau dp), in acest caz fiind 9 terminale.





Din punct de vedere al conexiunii LED–urilor ce formează segmentele afişajului, se întâlnesc afişoare cu anod comun şi cu catod comun, denumirea provenind de la tipul terminalului diodelor care se conectează împreună.

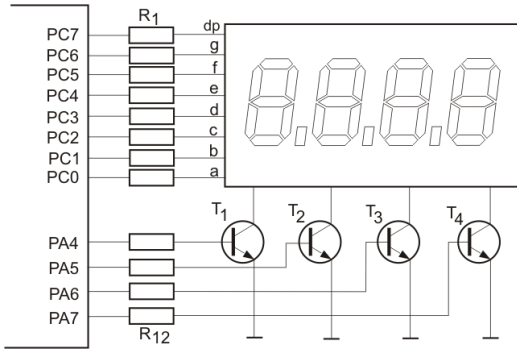


Mai multe afişoare (de ex. 4) se pot conecta la un microcontroler prin “multiplexare”. Numerele se afişează prin aprinderea succesivă a câte unui afişor din cele 4 existente, la o viteză atât de mare încât lasă impresia că toate cele 4 afişaje sunt aprinse simultan

Afişajul cu 7 segmente

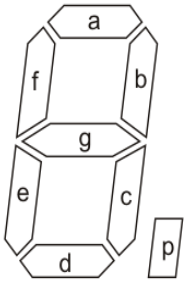


Cele 7+1 segmente ale afişajelor sunt legate în paralel si conectate la portul C al microcontrolerului. Pinii 4...7 ai portului A comandă, prin intermediul unor tranzistori, câte un afişaj.



Pentru a afişa o cifră pe unul din afişoare, acesta se activează prin intermediul pinului de comandă corespunzător, pe portul C fiind scris apoi numărul hexazecimal corespunzător cifrei dorite

Port C.	7	6	5	4	3	2	1	0	
	p	g	f	e	d	c	b	a	
$3F	0	0	1	1	1	1	1	1	0
$06	0	0	0	0	0	1	1	0	1
									2
									3
									4
									5
									6
									7
									8
									9
									,



Aplicaţia 8 – Afişarea unei cifre



Scopul aplicaţiei este afişarea unei cifre între 0 ... 9 pe unul din cele 4 afişaje

De exemplu pentru a scrie cifra 1 pe primul afişaj programul are următoarea formă:

- Se setează portul C si jumătate din portul A ca ieşiri cu instrucţiunile:

ddrc := $FF

ddra := $F0

- Se scrie pe portul C valoarea corespunzătoare cifrei 1:

portc := $06

- Se activează primul afişaj conectat la pinul 4 al portului A cu instrucţiunea:

porta.4 := 1

```
2
begin
    ddrc:=$ff;
    ddra:=$f0;
    portc:= $06;
    porta.4:=1;
end.
```

Cifra 2 pe al doilea afişaj

```
2
begin
    ddrc:=$ff;
    ddra:=$f0;
    portc:= $5B;
    porta.5:=1;
end.
```

Cifra 3 pe al treilea afişaj

```
2
begin
    ddrc:=$ff;
    ddra:=$f0;
    portc:= $4F;
    porta.6:=1;
end.
```

Cifra 4 pe ultimul afişaj

```
begin
    ddrc:=$ff;
    ddra:=$f0;
    portc:= $66;
    porta.7:=1;
end.
```

Aplicaţia 9 – Astabil cu două cifre (I)



Scopul aplicaţiei este afişarea alternativă a două cifre pe un afişaj.

De exemplu pentru a afişa alternativ cifrele 0 şi 1 pe primul afişaj, programul are următoarea formă:

```
begin
  ddrc:=$ff;
  ddra:=$f0;
  while 1=1 do begin
     portc:= $3F;
     porta.4:=1;
     delay_ms (500);
     portc:= $06;
     porta.4:=1;
     delay_ms (500);
  end;
  end.
```

Cifrele 2 şi 3 pe primul afişaj

```
begin
  ddrc:=$ff;
  ddra:=$f0;
  while 1=1 do begin
     portc:= $5B;
     porta.4:=1;
     delay_ms (500);
     portc:= $4F;
     porta.4:=1;
     delay_ms (500);
  end;
  end.
```

Cifrele 0 şi 9 pe primul afişaj

```
begin
  ddrc:=$ff;
  ddra:=$f0;
  while 1=1 do begin
     portc:= $3F;
     porta.4:=1;
     delay_ms (500);
     portc:= $6F;
     porta.4:=1;
     delay_ms (500);
  end;
  end.
```

Temă:

1. Scrieţi programul pentru afişarea alternativă a două cifre pe ultimul afişaj

Aplicaţia 10 – Astabil cu două cifre II



Scopul aplicaţiei este afişarea alternativă a două cifre pe afişaje diferite.

Ex: Cifrele 0 şi 1 pe afişajele 1 şi 2

```
begin
  ddrc:=$ff;
  ddra:=$f0;
while 1=1 do begin
    portc:= $3F;
    porta.4:=1;
    delay_ms (500);
    porta.4:=0;
    portc:= $06;
    porta.5:=1;
    delay_ms (500);
    porta.5:=0;
end;
end.
```

Cifrele 0 şi 1 pe afişajele 1 şi 4

```
begin
  ddrc:=$ff;
  ddra:=$f0;
while 1=1 do begin
    portc:= $3F;
    porta.4:=1;
    delay_ms (500);
    porta.4:=0;
    portc:= $06;
    porta.7:=1;
    delay_ms (500);
    porta.7:=0;
end;
end.
```

Cifrele 0 şi 1 pe afişajele 2 şi 3

```
begin
  ddrc:=$ff;
  ddra:=$f0;
while 1=1 do begin
    portc:= $3F;
    porta.5:=1;
    delay_ms (500);
    porta.5:=0;
    portc:= $06;
    porta.6:=1;
    delay_ms (500);
    porta.6:=0;
end;
end.
```

Aplicaţia 10 – Afişarea multiplexată a unui număr din 4 cifre



Pentru a uşura scrierea programului se poate defini iniţial o constantă de tip şir care să conţină codurile în hexazecimal corespunzătoare celor 10 cifre de la 0...9.

const cifra: array [0…9] of byte = ($3F, $06, $5B, $4F, $66, $6D, $7D, $07, $7F, $6F)

Pentru a afişa de exemplu cifra 9, se va scrie:

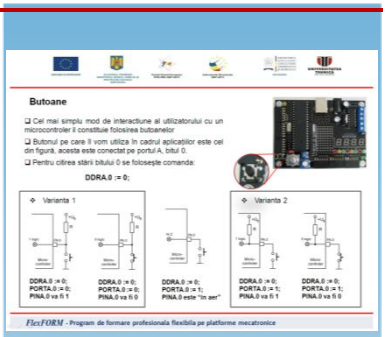
portc: = cifra[9]

unde 9 este ultimul termen din şirul definit iniţial adică $6F

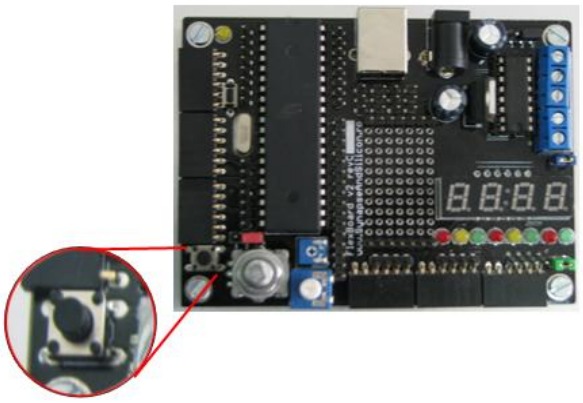
Numerele se afişează succesiv pe câte un afişaj, la o viteză suficient de mare, încât lasă impresia că toate cele 4 afişaje sunt aprinse simultan. Pentru aceasta fiecate afişaj va ramâne aprins un timp definit prin constanta **pauza.** In exemplul prezentat fiecare afişaj rămâne aprins 2 ms

```
const cifra:array [0..9] of byte=
($3f,$06,$5b,$4f,$66,$6D,$7D,$07,$7F,$6F);
const pauza=2;
begin
  ddrc:=$ff;
  ddra:=$f0;
  while 1=1 do begin
    portc:=cifra[1];
    porta.7:=1;
    delay_ms(pauza);
    porta.7:=0;
    portc:=cifra[9];
    porta.6:=1;
    delay_ms(pauza);
    porta.6:=0;
    portc:=cifra[3];
    porta.5:=1;
    delay_ms(pauza);
    porta.5:=0;
    portc:=cifra[2];
    porta.4:=1;
    delay_ms(pauza);
    porta.4:=0;
  end;
end.
```

Butoane



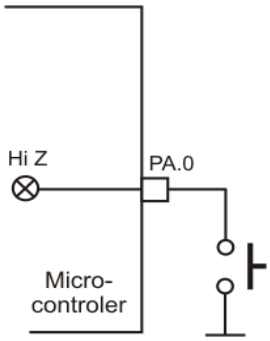
Cel mai simplu mod de interactiune al utilizatorului cu un microcontroler îl constituie folosirea butoanelor. Butonul pe care îl vom utiliza în cadrul aplicațiilor este cel din figură, acesta este conectat pe portul A, bitul 0.



Pentru citirea stării bitului 0 se foloseşte comanda:

DDRA.0 := 0;

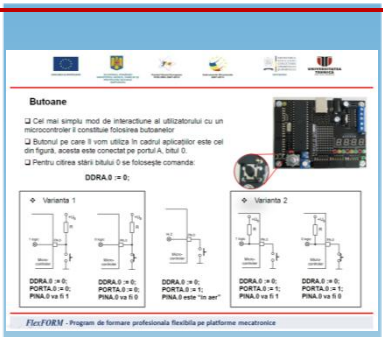
În figura de mai jos este reprezentat un buton conectat la pinul PA.0 al µC. La apăsarea butonului, intrarea PA.0 va fi legată la masă, deci va fi în starea logică “0”.



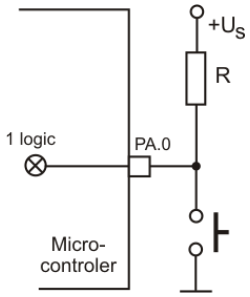
DDRA.0 := 0;
PORTA.0 := 1;
PINA.0 este “în aer”

Atunci când butonul nu este apăsat intrarea e în stare de impedanță mărită, ca și cum ar fi lăsată în aer, ea nefiind conectată nici la masă (0 logic), nici la U_S (1 logic). Această stare nu poate fi citită de către circuitele interne ale µC, pentru că un bit dintr-un registru poate să ia doar valorile 0 sau 1.

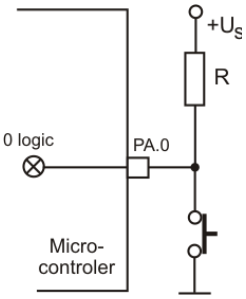
Butoane (continuare)



Corect butonul se conectează folosind o rezistenţă de pull-up între pinul de intrare și U_S. Aceasta rezistență are rolul de a **aduce intrarea în starea "1" logic atunci când butonul este liber** prin "ridicarea" potențialului liniei la U_S

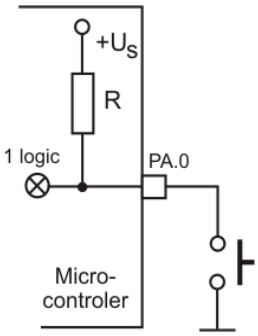


DDRA.0 := 0;
PORTA.0 := 0;
PINA.0 va fi 1

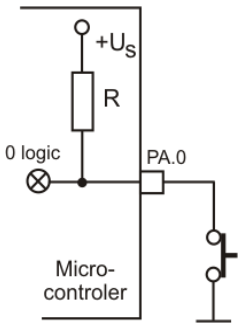


DDRA.0 := 0;
PORTA.0 := 0;
PINA.0 va fi 0

Pentru a economisi spațiu exterior, aceste rezistențe au fost incluse în interiorul circuitului integrat. Inițial ele sunt dezactivate iar activarea acestora se poate face prin software scriind o valoare în registrul de ieșire (PORTn) al unui port care a fost configurat drept intrare

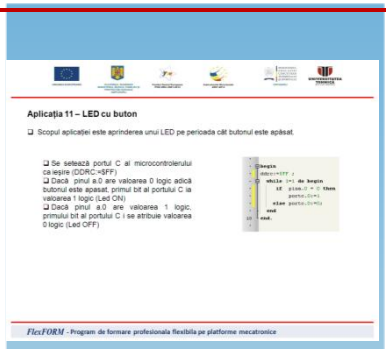


DDRA.0 := 0;
PORTA.0 := 0;
PINA.0 va fi 1



DDRA.0 := 0;
PORTA.0 := 0;
PINA.0 va fi 0

Aplicaţia 11 – LED cu buton



Scopul aplicaţiei este aprinderea unui LED pe perioada cât butonul este apăsat.

Programul are următoarea structură :

- se setează portul C al microcontrolerului ca ieşire:

ddrc:=$FF

- dacă pinul a.0 are valoarea 0 logic adică butonul este apasat, primul bit al portului C ia valoarea 1 logic (Led ON)
- dacă pinul a.0 are valoarea 1 logic, primului bit al portului C i se atribuie valoarea 0 logic (Led OFF)

```
begin
  ddrc:=$FF ;
    while 1=1 do begin
       if  pina.0 = 0 then
           portc.0:=1
      else portc.0:=0;
    end
end.
```

Aplicaţia 12 – LED cu buton ON/OFF



Scopul aplicaţiei este aprinderea unui LED la prima apăsare a butonului şi stingerea la următoarea apăsare

Stuctura programului este următoarea:

- se setează portul C al microcontrolerului ca ieşire cu instrucţiunea **ddrc:=$FF**
- primului bit al portului C i se atribuie valoarea 0 logic **portc.0:=0** (LED stins)
- In primul ciclu while programul aşteaptă apăsarea butonului (**pina.0 = 0**)
- În al doilea ciclu se aşteaptă eliberarea butonului (**pina.0 = 1**)
- După eliberarea butonului, primul bit al portului C ia valoarea 1 logic **portc.0:=1** (LED aprins) după care ciclul se repetă

```
begin
ddrc:=$FF ;
while 1=1 do begin
   portc.0:=0;
     while pina.0 = 0 do begin end;
     while pina.0 = 1 do begin end;
    portc.0:=1;
     while pina.0 = 0 do begin end;
     while pina.0 = 1 do begin end;

  end
end.
```

Aplicaţia 13– Numărător 0...9 cu buton



Scopul aplicaţiei este realizarea unui numărător de la 0 la 9 care se incrementează la fiecare apăsare a butonului

- se defineşte iniţial o constantă de tip şir care să conţină codurile în hexazecimal corespunzătoare celor 10 cifre de la 0...9:

const cifra: array [0…9] of byte = ($3F, $06, $5B, $4F, $66, $6D, $7D, $07, $7F, $6F)

- se activează primul afişaj conectat la pinul 4 al portului A cu instrucţiunea:

porta.4 := 1

- se scrie pe portul C valoarea corespunzătoare cifrei 0 adică :

portc := cifra [i]

unde ***i*** este primul termen din sirul definit iniţial

- la fiecare apăsare a butonului valoarea lui ***i*** creşte cu o unitate de la 0 până la 9 după care ciclul se reia.

```
const cifra:array [0..9] of byte=
($3f,$06,$5b,$4f,$66,$6D,$7D,$07,$7F,$6F);
var i:byte;
begin
  ddrc:=$FF;
  ddra:=$f0;
   while 1=1 do begin
    for i:=0 to 9 do begin
     porta.4:=1;
     portc:=cifra[i];
     while pina.0 = 0 do begin end;
     while pina.0 = 1 do begin end;
    end
  end
end.
```

INTERFAȚAREA SENZORILOR ŞI COMUNICAȚII DE DATE

OBIECTIVE

- Utilizarea protocolului de comunicație serială RS232 în aplicații de interfațare cu PC-ul;
- Achiziția de date utilizând convertorul analog-digital intern;
- Aplicații pentru afişarea datelor de PC şi/sau LCD;
- Interfațarea senzorilor analogici;
- Aplicație pentru citirea senzorului digital incremental în cuadratură

Comunicaţia serială



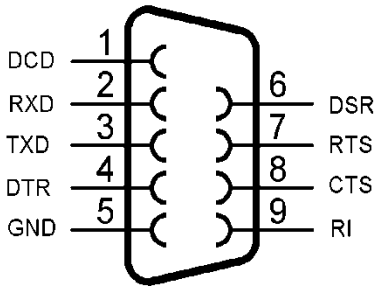
Comunicaţia serială este cea mai întâlnită metodă de interfaţare între sistemele ce conţin microcontrolere. Principalele avantaje sunt: numărul mic de linii necesare (minim una, de obicei două sau trei), distanţa relativ mare, viteza de comunicaţie suficient de ridicată pentru cele mai multe aplicaţii.

Pentru a permite compatibilitatea echipamentelor realizate de diferiţi producători, în 1960 s-a adoptat standardul numit RS232 realizat de EIA (Electronics Industries Association). Interfaţarea microcontrolelor cu calculatorul personal (PC) poate fi realizată utilizând comunicaţia serială conform standadrului RS232.

In general comunicaţia serială poate fi realizată în trei variante:

- *Simplex*, în care numai un echipament emite iar celălalt recepţionează;
- *Half-duplex* în care pe rând fiecare echipament transmite în timp ce echipamentul opus recepţionează;
- *Full-duplex* unde simultan fiecare echipament transmite şi recepţionează date.

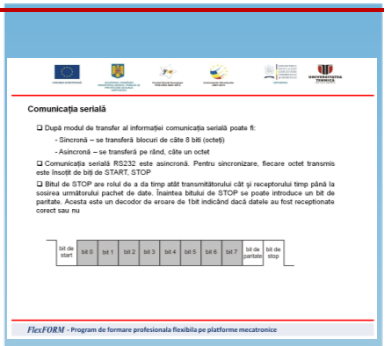
Viteza de comunicaţie se măsoară în **bps (bit per second)** şi reprezintă numărul de biţi ce poate fi transmis într-o secundă. Ratele de transfer standardizate pentru RS232 sunt: 110, 300, 1200, 2400, 4800, 9600, 19200, 38400, 57600, 76800,115200, 230400, 460800 912600 bps.





În comunicaţiile actuale cel mai frecvent sunt utilizate semnalele: RxD-intrare recepţie date, TxD-ieşire transmisie date şi GND-masă de semnal.

Deoarece standardul RS232 a fost adoptat înainte de apariţia familiei logice TTL, standardul nu este compatibil cu acestea. Nivelul 1 logic este reprezentat de o tensiune electrică cuprinsă între -3V şi -25V iar nivelul logic 0 este reprezentat de o tensiune electrică cuprinsă între +3V şi +25V. Zona situată între -3V şi 3V fiind nedefinită. Pentru conversia de nivele TTL <-> RS232 se utilizează circuite specializate


Comunicaţia serială

- După modul de transfer al informaţiei comunicaţia serială poate fi:
  - Sincronă – se transferă blocuri de câte 8 biţi (octeţi)
  - Asincronă – se transferă pe rând, câte un octet
- Comunicaţia serială RS232 este asincronă. Pentru sincronizare, fiecare octet transmis este însoţit de biţi de START, STOP
- Bitul de STOP are rolul de a da timp atât transmiţătorului cât şi receptorului timp până la sosirea următorului pachet de date. Înaintea bitului de STOP se poate introduce un bit de paritate. Acesta este un decodor de eroare de 1bit indicând dacă datele au fost receptionate corect sau nu

După modul de transfer al informaţiei comunicaţia serială poate fi:

Asincronă – emiţătorul inserează un impuls de tact, cunoscut ca bit de *Start*, în faţa fiecarui octet transmis. Astfel, pentru fiecare caracter ASCII avem o transmisie independentă, cu adăugarea biţilor de *Start*, *Stop* şi *Paritate*. Viteza de lucru se stabileste manual la inceputul transmisiei. Pentru ca această metodă să functioneze, trebuie să existe, o perioada de "pauză" între caractere, realizată cu bitul de *Stop.*

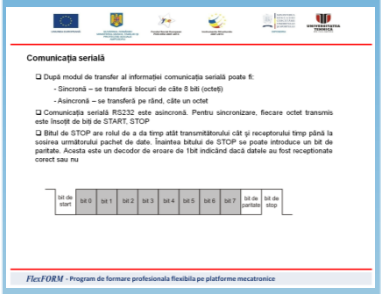
Sincronă – caracterele sunt transmise rapid, unul după altul, fără biţi de *Start* şi de *Stop*. Pentru sincronizare, mesajul transmis este precedat de caractere speciale de sincronizare, care pot fi detectate de receptor. Acestea sunt transmise încontinuu chiar şi când nu sunt date de transmis. Transmisiile în mod sincron folosesc aceeaşi pereche de fire pentru semnalele de date şi semnale de ceas. Acest tip de transmisie, este folosită în reţelele Ethernet.

Comunicaţia serială RS232 este asincronă. Pentru sincronizare, fiecare octet transmis este însoţit de biţi de START, STOP.



Bitul de STOP are rolul de a da timp atât transmitătorului cât şi receptorului timp până la sosirea următorului pachet de date. Uneori înaintea bitului de STOP se introduce un bit de paritate. Acesta este un decodor de eroare de 1bit indicând dacă datele au fost receptionate corect sau nu.

Codificarea datelor şi controlul erorilor



Erorile de comunicaţie pot să apară atunci când circuitele şi cablurile folosite pentru conexiune sunt afectate de interferenţe electrice. Aceste interferenţe sunt induse în firele de comunicaţie care se comportă ca nişte “antene”. Deoarece tensiunile cu care se lucreaza sunt mici, efectul interferenţelor este important.

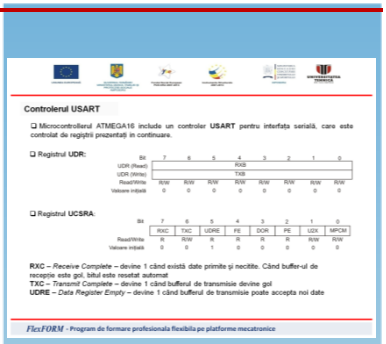
Metodele de detectie şi corecţie a erorilor implică introducerea de informaţie neesentială, pe lângă datele utile, în transmisia datelor. La oricare din metode, emiţătorul prelucrează o parte din date şi generează un fel de semnatură pe care apoi o transmite impreună cu datele utile. Când mesajul ajunge la receptor, acesta prelucrează datele primite şi generează o semnatură pe care o compară cu cea primită. Dacă cele două semnături nu coincid, atunci datele au fost alterate.

Metoda **bitului de paritate** se poate aplica pentru date binare de orice lungime. Pentru fiecare cuvânt este adaugat un bit de paritate (semnatură). Paritatea poate fi pară sau impară. Calcularea parităţii se poate face cu operatorul XOR (SAU exclusiv) între biţii cuvântului. Prin această metodă este posibilă doar **detecţia erorii singulare**, cand sunt afectaţi un numar impar de biţi.

Suma de control la nivel de bloc este alt mecanism de detecţie a erorilor de transmisie. Datele sunt împărţite în blocuri, care apoi se însumează şi se obţine o sumă care va fi trunchiată, inversată şi adaugată la sfârşit. La receptie, blocurile primite, care includ şi suma, se adună pe masură ce sosesc, şi dacă suma obţinută nu este 0 atunci înseamnă că datele sunt eronate şi secvenţa trebuie retransmisă.

Altă metodă de detecţie a erorilor este **CRC (Cyclic Redundant Check)**. În acest caz secventa de biţi este impărţită cu un număr special ales. Impărtirea se face în modulo 2, adică folosind operatorul XOR. Restul impărţirii reprezintaă semnătura care va fi adaugată la sfârsit, după biţii utili. La receptie, se recalculează restul impărtirii şi dacă nu coincide cu cel primit, atunci secventa este eronată. CRC-ul se poate calcula şi prin metode hardware, folosind registre cu deplasare şi porti logice XOR.

Controlerul de comunicaţie USART


Controlerul USART

❑ Microcontrollerul ATMEGA16 include un controler **USART** pentru interfaţa serială, care este controlat de regiştrii prezentaţi in continuare.

❑ Registrul **UDR:**

❑ Registrul **UCSRA:**

RXC – *Receive Complete* – devine 1 când există date primite şi necitite. Când buffer-ul de recepţie este gol, bitul este resetat automat

TXC – *Transmit Complete* – devine 1 când bufferul de transmisie devine gol

UDRE – *Data Register Empty* – devine 1 când bufferul de transmisie poate accepta noi date

FlexFORM - Program de formare profesionala flexibila pe platforme mecatronice

Din perspectiva microcontrolerului, comunicaţia serială se bazează pe: linia pentru transmisie, notată cu **Tx** şi linia pentru receptie, notată cu **Rx**. Microcontrolerul ATMEGA16 include un controler **USART** (**U**niversal **S**ynchronous-**A**synchronous **R**eceiver/**T**ransmitter) pentru interfaţa serială, care este controlat de regiştrii prezentaţi in continuare.

Registrul **UDR:**

Bit	7	6	5	4	3	2	1	0
UDR (Read)	RXB							
UDR (Write)	TXB							
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Valoare inițială	0	0	0	0	0	0	0	0

RXB şi TXB sunt bufferele de recepţie, respectiv transmisie. Ele folosesc aceeaşi adresă de I/O. Deci RXB este accesat citind din registrul UDR iar TXB scriind în UDR. Bufferul de transmisie poate fi scris numai atunci când bitul UDRE din portul UCSRA este 1. În caz contrar, scrierea nu este posibilă

Registrul **UCSRA:**

Bit	7	6	5	4	3	2	1	0
	RXC	TXC	UDRE	FE	DOR	PE	U2X	MPCM
Read/Write	R	R/W	R	R	R	R	R/W	R/W
Valoare inițială	0	0	1	0	0	0	0	0

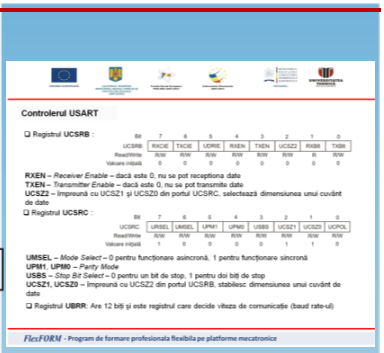
UCSRA este primul registru de stare al controlerului de comunicaţie. Biţii cei mai importanţi sunt:

RXC – *Receive Complete:* devine 1 când există date primite şi necitite. Când buffer-ul de recepţie este gol, bitul este resetat automat,

TXC – *Transmit Complete:* devine 1 când bufferul de transmisie devine gol,

UDRE – *Data Register Empty:* devine 1 când bufferul de transmisie poate accepta noi date .

Controlerul de comunicaţie USART



Registrul **UCSRB:**

Bit	7	6	5	4	3	2	1	0
UCSRB	RXCIE	TXCIE	UDRIE	RXEN	TXEN	UCSZ2	RXB8	TXB8
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R	R/W
Valoare inițială	0	0	0	0	0	0	0	0

UCSRA este al doilea registru de stare al controlerului de comunicaţie. Biţii cei mai importanţi sunt:

RXEN – *Receiver Enable:* dacă este 0, nu se pot recepţiona date,
TXEN – *Transmitter Enable:* dacă este 0, nu se pot transmite date,
UCSZ2 – împreună cu UCSZ1 şi UCSZ0 din portul UCSRC, selectează dimensiunea unui cuvânt de date.

Registrul **UCSRC:**

Bit	7	6	5	4	3	2	1	0
UCSRC	URSEL	UMSEL	UPM1	UPM0	USBS	UCSZ1	UCSZ0	UCPOL
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Valoare inițială	1	0	0	0	0	1	1	0

UMSEL – *Mode Select:* 0 pentru funcţionare asincronă, 1 pentru funcţionare sincronă
UPM1, UPM0 – *Parity Mode:*

PM1	UPM0	
1	0	Even Parity
1	1	Odd Parity

USBS – *Stop Bit Select:* 0 pentru un bit de stop, 1 pentru doi biţi de stop
UCSZ1, UCSZ0 – împreună cu **UCSZ2** din portul UCSRB, stabilesc dimensiunea unui cuvânt de date

UCSZ2	UCSZ1	UCSZ0	
0	0	0	5 bit
0	0	1	6 bit
0	1	0	7 bit
0	1	1	8 bit

Registrul **UBRR**: Are 12 biţi şi este registrul care stabileşte viteza de comunicaţie (baud rate-ul).

Aplicaţia 1 – Trimiterea unui caracter cu ajutorul interfeţei seriale



Aplicaţia are drept scop realizarea comunicaţiei seriale între PC şi microcontroler. La apăsarea unui buton microcontrolerul va trimite pe serial cifra 1. La eliberarea butonului va fi trimisă cifra 0.

Pentru iniţializarea controlerului USART se utilizează următoarea instrucţiune:

UARTx_Init(baud_rate: longint);

unde: *baud_rate* reprezintă viteza de comunicaţie (de ex: 9600 bps).

- Pentru a trimite şi/sau recepţiona caractere pe 8 biţi sau texte, se vor utiliza instrucţiunile:

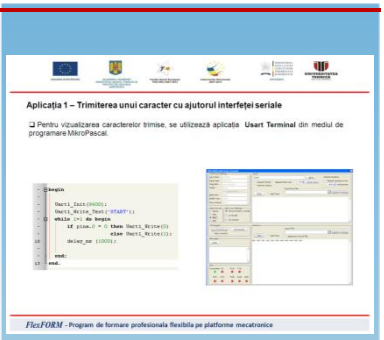
UARTx_Read(): byte;

UARTx_Read_Text(Output, Delimiter, Attempts);

UARTx_Write(data_:byte);

UARTx_Write_Text(uart_text);

Aplicaţia 1 – Trimiterea unui caracter cu ajutorul interfeţei seriale



Stuctura programului este următoarea:

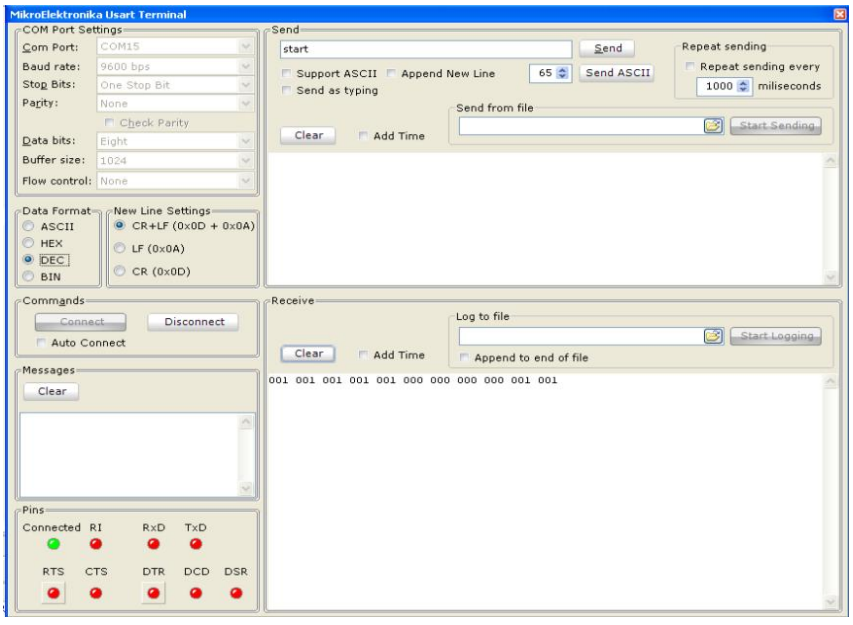
- la inceputul programului se iniţializează controlerul USART şi se stabileşte viteza de comunicaţie la 9600bps.
- Dacă este apăsat butonul conectat la pinul 0 al portului A se trimite caracterul 0
- Dacă butonul nu este apăsat atunci se trimite caracterul 1

```
begin

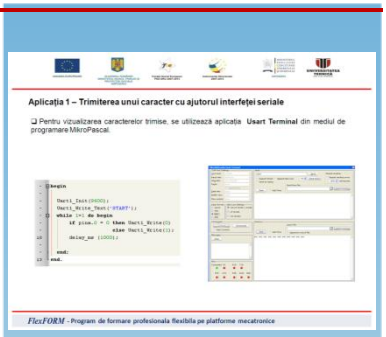
  Uart1_Init(9600);
  Uart1_Write_Text('START');
  while 1=1 do begin
      if pina.0 = 0 then Uart1_Write(0)
                    else Uart1_Write(1);
      delay_ms (1000);

  end;
end.
```

Pentru vizualizarea caracterelor trimise, se utilizează aplicaţia **Usart Terminal** din mediul de programare MikroPascal



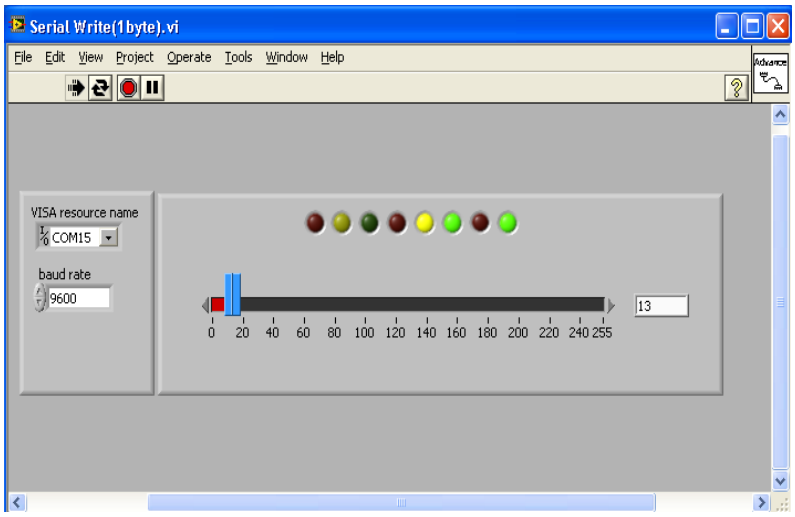
Aplicaţia 2 – Recepţionarea unui octet cu ajutorul interfeţei seriale



Aplicaţia are drept scop realizarea comunicaţiei seriale intre PC şi microcontroler şi afişarea octetului recepţionat pe 8 LED-uri.

Pentru generarea numerelor intre 0 şi 255 se va utiliza aplicaţia creată în acest scop în mediul LabVIEW.

```
var c:byte;
begin
ddrc:=$FF;
   Uart1_Init(9600);
   Uart1_Write_Text('START');
   while 1=1 do begin
     if ucsra.rxc = 1
     then begin
      c:= Uart1_Read;
      Uart1_Write(c);
      portc:=c;
     end;
   end;
end.
```



Afişajul alfanumeric cu cristale lichide LCD



Afişoarele alfanumerice cu cristale lichide pot afişa simboluri şi caractere formate din matrice de puncte (dot matrix).

După vizibilitatea caracterelor se deosebesc afişaje cu vizibilitate standard (unghiul de vizare 30...40 grd), respectiv afişoare cu unghi de vizare dublu.

După modul de formare a imaginii există module transreflective care nu necesită iluminare şi reflective cu backlight care dispun de o sursă proprie de iluminare cu LED sau folie electroluminiscentă.

După culoarea imaginii sunt afişoare normale cu matricea de puncte întunecate pe fond luminos sau inverse.

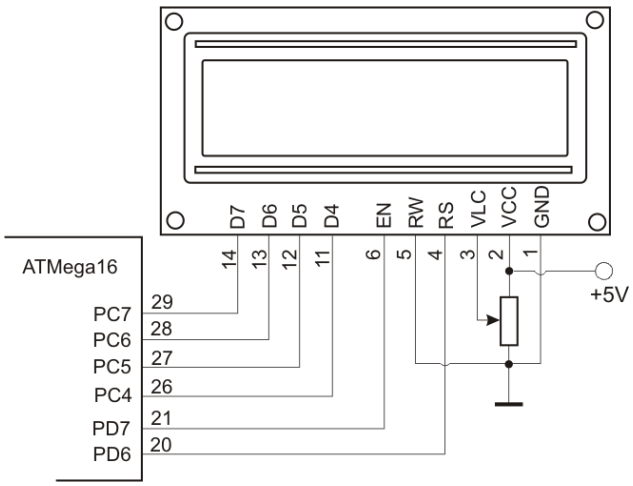
Majoritatea afişoarelor de acest tip folosesc ca driver circuitul integrat Hitachi HD44780. Interfața de comunicație este una paralelă pe 8 biţi, permițând astfel să se efectueaze scrieri sau citiri de date într-un mod simplu și rapid.



Pe cele 8 fire de date se transmit octeți (grupuri de 8 biti), acești octeți reprezentând coduri interne ale LCD-ului (în modul de instrucțiuni) sau coduri ASCII ale caracterelor ce se doresc a fi afișate (în modul de date). Din punct de vedere hardware interfațarea se efectuează pin la pin cu microcontrolerul.

În cazul în care se dorește interfațare pe 8 fire este suficient să se conecteze toți pinii D0-D7 la același port și cei de control RS, RW și EN pe un alt port și să se scrie programul ținând cont de această structură. Pinii VSS și GND se conectează la alimentare și masă Pinul 3, VLC, se conectează direct la masa, pentru contrast maxim, sau într-un potențiometru pentru a putea regla contrastul.

Există și o variantă de interfațare economică pe 4 biți de date +2 biți pentru comenzi. Daca nu intenționăm să citim informația provenită de la LCD ci doar să scriem date pe el, atunci pinul RW se leagă la masă.



Din punct de vedere al modului de comunicație se parcurg următorii pași:

- Se setează dacă se dorește a se trimite o instrucțiune sau un caracter de date (pinul R/S)
- Se trimit cei mai importanți 4 biti din octet,
- Se activează o perioadă de timp pinul EN
- Se trimit cei mai puțin semnificativi 4 biți din octet
- Se activează din nou o perioadă de timp pinul EN

In varianta de interfațare pe 4 biți, timpul necesar scrierii in modulul LCD crește semnificativ față de varianta de interfațare standard.

Aplicaţia 3 – Afișarea unui text pe LCD


Aplicația 3 – Afișarea unui text pe LCD

❑ Pentru a fi mai ușoară interfațarea unui modul LCD a fost concepută o librarie numită **flexunit** care include toate instrucțiunile necesare pentru afișarea caracterelor, numerelor etc.

❑Pentru a scrie un text pe LCD sunt suficiente două instrucțiuni:

- *Lcd_Init* care inițializează comunicația cu LCD-ul
- *Lcd_text (x,y, 'TEXT')* unde :

text indică faptul ca pe LCD va fi afișat un text

x reprezintă rândul

y reprezintă coloana

'TEXT' este textul care se doreste a fi afișat (max. 16 caractere)

```
program LCD;
uses flexunit;
begin
  Lcd_Init;
  Lcd_text(1,1,'MECATRONICA');
end.
```

FlexFORM - Program de formare profesionala flexibila pe platforme mecatronice

Pentru a fi mai uşoară interfaţarea unui modul LCD a fost concepută o librarie numită **flexunit** care include toate instrucțiunile necesare pentru afişarea caracterelor, numerelor etc.

Pentru a scrie un text pe LCD sunt suficiente două instrucţiuni:

- ***Lcd_Init*** care inițializează comunicația cu LCD-ul

- ***Lcd_text (x,y, 'TEXT')*** unde ***:***

text indică faptul ca pe LCD va fi afişat un text

x reprezintă rândul

y reprezintă coloana

'TEXT' este textul care se doreste a fi afişat (max. 16 caractere)

```
program LCD;
uses flexunit;
begin
  Lcd_Init;
  Lcd_text(1,1,'MECATRONICA');
end.
```

Aplicaţia 4 – Afişarea pe LCD a unui text recepţionat pe serial



Aplicaţia are drept scop realizarea comunicaţiei seriale intre PC şi microcontroler şi afişarea datelor receptionate pe LCD.

```
program pc_LCD;
uses flexunit;
    var k:word;
        s:string[5];
begin
    Uart1_Init(9600);
    Uart1_Write_Text('255');
    Lcd_Init;
   while 1=1 do begin
      if uart1_data_ready then begin
      k:=Uart1_Read;
       Lcd_word(1,1,k);
      delay_ms(30);
     end;
  end;
 end.
```

Convertorul analog – digital (ADC)


Convertorul analog – digital (ADC)

❑ Pentru a putea fi utilizate într-un sistem de calcul digital, semnale analogice trebuiesc convertite în valori numerice discrete.

❑ Un convertor analog – digital (ADC) este un circuit care convertește o tensiune analogică de la intrare într-o valoare digitală.

❑ Conversia semnalelor analogice în semnale digitale poate fi realizată prin:

- Metode hardware:
  - convertoare tensiune – frecvenţă sau tensiune – timp
  - convertoare cu pantă simplă dublă sau multiplă
  - convertoare cu eşantionare (sample & hold)
  - convertoare sigma-delta (convertoare pe 1bit)
- Metode software:
  - aproximaţii succesive
  - măsurarea timpului de încărcare al unui condensator

❑ O caracteristică importantă a unui ADC o constituie rezoluția acestuia.

FlexFORM - Program de formare profesionala flexibila pe platforme mecatronice

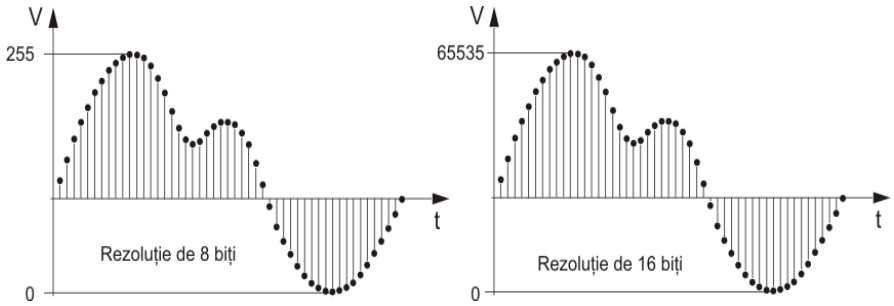
Pentru a putea fi utilizate într-un sistem de calcul digital, semnale analogice trebuiesc convertite în valori numerice discrete.

Un convertor analog – digital (ADC) este un circuit care convertește o tensiune analogică de la intrare într-o valoare digitală. Aceasta poate fi reprezentată în mai multe feluri în funcție de codificarea datelor: în binar, cod Gray etc

Conversia semnalelor analogice în semnale digitale poate fi realizată prin:

- Metode hardware:
  - convertoare tensiune – frecvenţă sau tensiune – timp
  - convertoare cu pantă simplă dublă sau multiplă
  - convertoare cu eşantionare (sample & hold)
  - convertoare sigma-delta (convertoare pe 1bit)
- Metode software:
  - aproximaţii succesive
  - măsurarea timpului de încărcare al unui condensator

O caracteristică importantă a unui ADC o constituie rezoluția acestuia. Rezoluția indică numărul de valori discrete pe care convertorul poate să le furnizeze la ieșirea sa în intervalul de măsură. Deoarece rezultatele conversiei sunt de obicei stocate intern sub formă binară, rezoluția unui convertor analog-digital este exprimată in biți



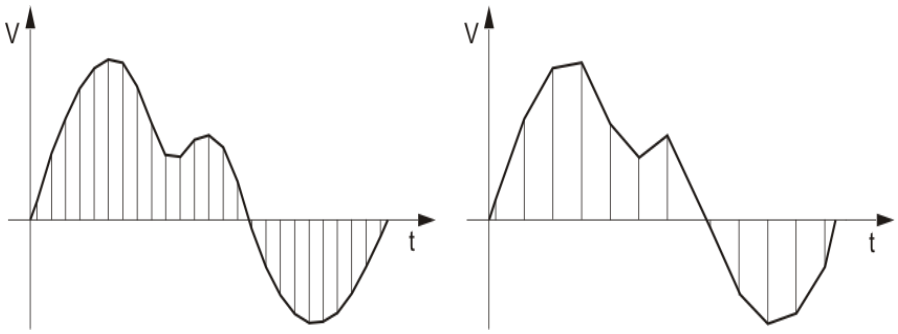
De exemplu, dacă rezoluția unui convertor este de 10 biți atunci el poate furniza 2^{10} = 1024 valori diferite la ieșire. Dacă gama de măsurare este de 0-5V, rezoluția de măsurare va fi:

$$\frac{5V - 0V}{1024} = 0{,}0048V = 4{,}8mV$$

O altă caracteristică importantă a unui convertor analog-digital o constituie rata de eșantionare. Aceasta depinde de timpul dintre două conversii succesive și afectează modul în care forma de undă originală va fi redată după procesarea digitală.

Care este rata minimă de eșantionare pentru a reproduce fără pierderi un semnal de o frecvența data? Teorema lui Nyquist spune că o rată de eșantionare de minim două ori mai mare decât frecvența semnalului măsurat este necesară pentru acest lucru, teorema aplicându-se și pentru un semnal compus dintr-un intreg spectru de frecvențe, cum ar fi vocea umană.

Limitele maxime ale auzului uman sunt 20Hz – 20kHz dar frecvențele vocii sunt în gama 80 - 4000 Hz, de aceea, centralele telefonice folosesc o rată de eşantionare a semnalului de 8000Hz. Rezultatul este o reproducere inteligibilă a vocii umane, suficientă pentru transmiterea de informații într-o convorbire obişnuită. Înregistrarea pe un CD audio are o rată de eșantionare de 44100Hz ceea ce este mai mult decât suficient pentru reproducerea fidelă a tuturor frecvențelor audibile.



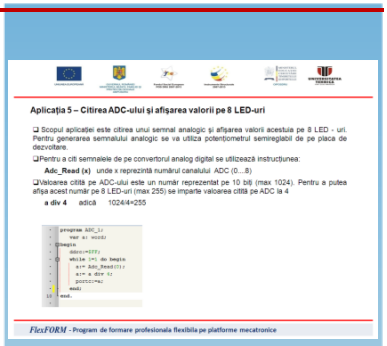
Convertorul analog-digital inclus în microcontrolerul Atmega16 este un ADC cu aproximări succesive. Are o rezoluție de 10 biţi şi poate măsura orice tensiune din gama 0-5V de pe opt intrări analogice multiplexate. Dacă semnalul de la intrare este prea mic în amplitudine, convertorul are facilitatea de preamplificare a acestuia in două setări, de 10x sau de 200x

Relația dintre valoarea pusă în registrul ADC şi tensiunea măsurată este următoarea:

$$ADC = \frac{V_{in} * 1024}{V_{ref}}$$

unde V_{in} este tensiunea măsurată iar V_{ref} este tensiunea aleasă ca referință

Aplicaţia 5 – Citirea ADC-ului şi afişarea valorii pe 8 LED-uri



Scopul aplicaţiei este citirea unui semnal analogic şi afişarea valorii acestuia pe 8 LED - uri. Pentru generarea semnalului analogic se va utiliza potenţiometrul semireglabil de pe placa de dezvoltare.

Pentru a citi semnalele de pe convertorul analog digital se utilizează instrucţiunea:

Adc_Read (x)

unde: x reprezintă numărul canalului ADC (0....8)

Valoarea citită pe ADC-ului este un număr reprezentat pe 10 biţi (max 1024). Pentru a putea afişa acest număr pe 8 LED-uri (max 255) se imparte valoarea citită pe ADC la 4

a div 4 adică 1024/4=255

```
program ADC_1;
   var a: word;
begin
   ddrc:=$FF;
   while 1=1 do begin
     a:= Adc_Read(0);
     a:= a div 4;
     portc:=a;
   end;
end.
```

Aplicaţia 6 – Citirea ADC-ului şi afişarea valorii pe LCD



Scopul aplicaţiei este citirea unui semnal analogic şi afişarea valorii acestuia pe LCD. Pentru generarea semnalului analogic se va utiliza potenţiometrul semireglabil de pe placa de dezvoltare.

Pe primul rând al LCD-ului va fi afişată valoarea citita de pa canalul 0 al convertorului analog –digital iar pe al doilea rând va fi afişat un text. (ex: FLEXFORM).

Pentru afişarea valorii ADC-ului care este un număr pe 10 biţi folosim instrucţiunea:

Lcd_word (x,y,k) , unde

word reprezintă formatul numărului care va fi afişat

x reprezintă rândul

y reprezintă coloana

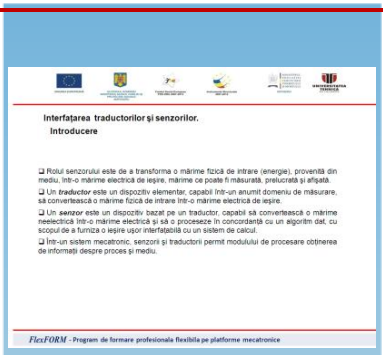
k este valoarea numărul care este afişat, în acest caz va fi valoarea citită pe ADC

Pentru afişarea textului vom folosi instrucţiunea :

Lcd_text (x,y, 'TEXT')

```
program LCD_2;
uses flexunit;
var k:word;
     s:string[5];
begin
 Lcd_Init;
 while 1=1 do begin
  k:=Adc_Read(0);
  Lcd_word(1,1,k);
  Lcd_text(2,1,'FLEXFORM');
   delay_ms(30);
  end;
end.
```

Interfaţarea traductorilor şi senzorilor.



Introducere

Rolul senzorului este de a transforma o mărime fizică de intrare (energie), provenită din mediu, într-o mărime electrică de ieşire, mărime ce poate fi măsurată, prelucrată şi afişată. În diferite discipline, pe lângă noţiunea de “senzor”, se utilizează noţiunea de “traductor”.

Un ***traductor*** este un dispozitiv elementar, capabil într-un anumit domeniu de măsurare, să convertească o mărime fizică de intrare într-o mărime electrică de ieşire. Traductorul în sine nu conţine elemente de procesare, scopul lui este doar realizarea conversiei.

Un ***senzor*** este un dispozitiv bazat pe un traductor, capabil să convertească o mărime neelectrică într-o mărime electrică şi să o proceseze în concordanţă cu un algoritm dat, cu scopul de a furniza o ieşire uşor interfaţabilă cu un sistem de calcul.

Principala diferenţă faţă de noţiunea de traductor constă în natura neelementară a senzorului prin faptul că el este capabil să posede şi alte funcţii pe lângă cea de conversie de energie

Într-un sistem mecatronic, senzorii şi traductorii permit modulului de procesare obţinerea de informaţii despre proces şi mediu. Fără aceste dispozitive, sistemul nu poate funcţiona. De multe ori, calitatea sistemului mecatronic este în cea mai mare parte dependentă de calitatea sistemului de senzori şi traductoare.

Clasificare


Clasificare

❑ Senzorii şi traductorii pot fi clasificaţi în raport cu caracteristicile lor de intrare/ieşire. Corespunzător mărimii fizice de intrare, aceste dispozitive pot fi:
- absolute
-incrementale

❑ Natura semnalului de ieşire determină dacă dispozitivul este:
- analog
- digital

❑ După natura mărimii fizice de intrare, putem deosebi senzori şi traductori pentru:
- mărimi geometrice (lungime, arie, volum, nivel, rugozitate etc.);
- mărimi fizice (masă, forţă, presiune, viteză, debit, temperatură etc.);
- mărimi chimice (gaze, ioni, pH, umiditate, concentraţii etc.);
-mărimi fotometrice.

❑ Principalele caracteristici ale senzorilor şi traductoarelor sunt:
- liniaritatea
- precizia
- rezoluţia

FlexFORM - Program de formare profesionala flexibila pe platforme mecatronice

Senzorii şi traductorii pot fi clasificaţi în raport cu caracteristicile lor de intrare/ieşire. Corespunzător mărimii fizice de intrare, aceste dispozitive pot fi :

- *absolute*: când pentru o origine fixată, semnalul electric de ieşire poate reprezenta toate valorile posibile ale semnalului de intrare;
- *incrementale*: când o origine nu poate fi fixată, oricare ar fi valoarea semnalului fizic de intrare, fiecare punct de măsură fiind luat drept origine pentru următorul punct de măsură.

Natura semnalului de ieşire determină dacă dispozitivul este:

- *analog*: când semnalul de ieşire este continuu şi depinde liniar sau neliniar de mărimea fizică de intrare;
- *digital*: când, pentru o mărime fizică de intrare continuă semnalul de ieşire poate lua un număr discret de valori.

Din punct de vedere energetic putem avea dispozitive:

- *active*: realizarea conversiei mărimii fizice de intrare în semnal electric se face fără aport de energie auxiliară (termocupluri, dispozitive fotoelectrice etc.);
- *pasive*: realizarea conversiei mărimii fizice de intrare în semnal electric se face cu aport de energie auxiliară (termorezistenţe, traductoare tensometrice etc.).

Principalele caracteristici statice sunt:

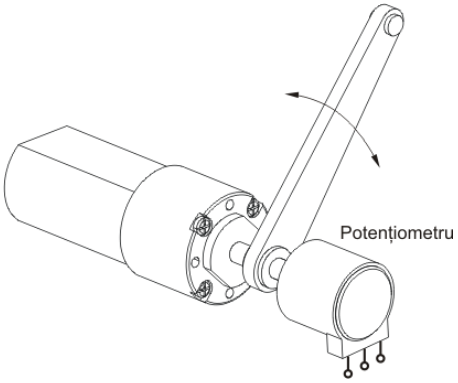
- *liniaritatea*: se defineşte ca “măsura” în care legătura între mărimea fizică de intrare şi semnalul electric de ieşire poate fi exprimată printr-o funcţie liniară;
- *precizia*: se defineşte ca toleranţa cu care o mărime poate fi repetată (adică “abilitatea” dispozitivului de a da aceeaşi valoare de ieşire pentru aceeaşi valoare de intrare);
- *rezoluţia*: se defineşte ca minimul variaţiei semnalului de ieşire, produs ca urmare a variaţiei continue a unui semnal fizic de intrare.

Traductoare şi senzori de poziţie

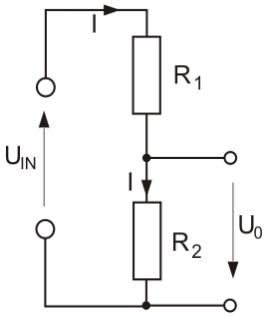


Măsurarea cu precizie a deplasărilor (implicit a poziţiei) este necesară în multe aplicaţii de mecanică fină, robotică etc. De exemplu, în cazul roboţilor, sistemul de control trebuie să primească date despre poziţia fiecărei articulaţii în scopul calculării poziţiei finale

Cel mai simplu traductor de poziţie este potenţiometrul. Acesta converteşte o deplasare mecanică într-un semnal electric, pe baza principiului divizorului de potenţial.



Regula divizorului de tensiune se poate aplica pentru a determina tensiunea la iesirea unui circuit, fără sarcină la iesire. În acest caz ambele rezistoare sunt parcurse de acelaşi curent I, determinat cu legea lui Ohm:



$$\left.\begin{aligned} I &= \frac{U_{in}}{R_1 + R_2} \\ U_0 &= IR_2 \end{aligned}\right\} => U_0 = \frac{R_2}{R_1 + R_2} U_{in}$$

Traductoare şi senzori de poziţie


Traductoare şi senzori de poziţie

❑ Un potenţiometru este alcătuit in principal dintr-un material rezistiv de forma unui arc de cerc şi un cursor care glisează pe materialul rezistiv. Ca şi material rezistiv se folosesc de obicei pelicule de carbon, oxizi metalici sau metalo-ceramice (cermet). Cursorul se realizează din bronz fosforos sau aliaje rezistente la uzură.

❑ Legea de variaţie a rezistenţei, care indică variaţia valorii rezistenţei electrice în funcţie de poziţia unghiulară a cursorului poate fi:

- liniară,
- logaritmică ,
- exponenţială ,
- invers exponenţială etc.

❑ Din punct de vedere constructiv potenţiometrele pot fi:

- ***simple***, echipate cu un singur element rezistiv şi care la randul lor pot fi circulare, rectilinii, multitură, cu intrerupător etc
- ***multiple***, două sau mai multe secţiuni comandate de un singur ax pe care sunt fixate cursoarele

FlexFORM - Program de formare profesionala flexibila pe platforme mecatronice

Un potenţiometru este alcătuit in principal dintr-un material rezistiv de forma unui arc de cerc şi un cursor care glisează pe materialul rezistiv. Ca şi material rezistiv se folosesc de obicei pelicule de carbon, oxizi metalici sau metalo-ceramice (cermet). Cursorul se realizează din bronz fosforos sau aliaje rezistente la uzură

Legea de variaţie a rezistenţei, care indică variaţia valorii rezistenţei electrice în funcţie de poziţia unghiulară a cursorului poate fi:

- liniară,

- logaritmică,

- exponenţială,

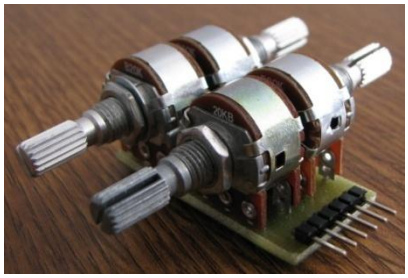
- invers exponenţială etc.

Din punct de vedere constructiv potenţiometrele pot fi:

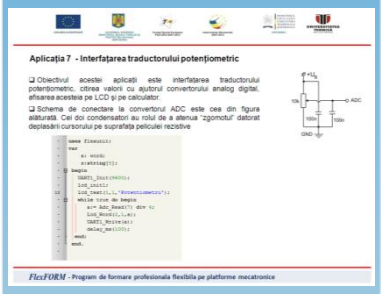
- ***simple***, echipate cu un singur element rezistiv şi care la randul lor pot fi circulare, rectilinii, multitură, cu intrerupător etc



- ***multiple***, două sau mai multe secţiuni comandate de un singur ax pe care sunt fixate cursoarele

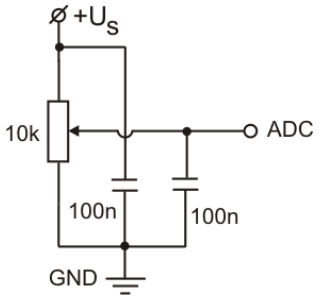


Aplicaţia 7 - Interfaţarea traductorului potenţiometric



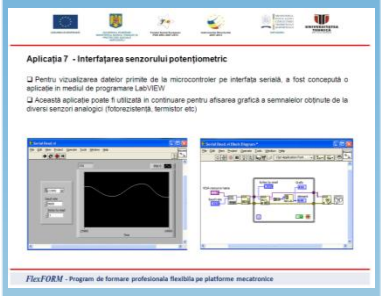
Obiectivul acestei aplicaţii este interfaţarea traductorului potenţiometric, citirea valorii cu ajutorul convertorului analog digital, afisarea acesteia pe LCD şi pe calculator.

Schema de conectare la convertorul ADC este cea din figura alăturată. Cei doi condensatori au rolul de a atenua “zgomotul” datorat deplasării cursorului pe suprafaţa peliculei rezistive.



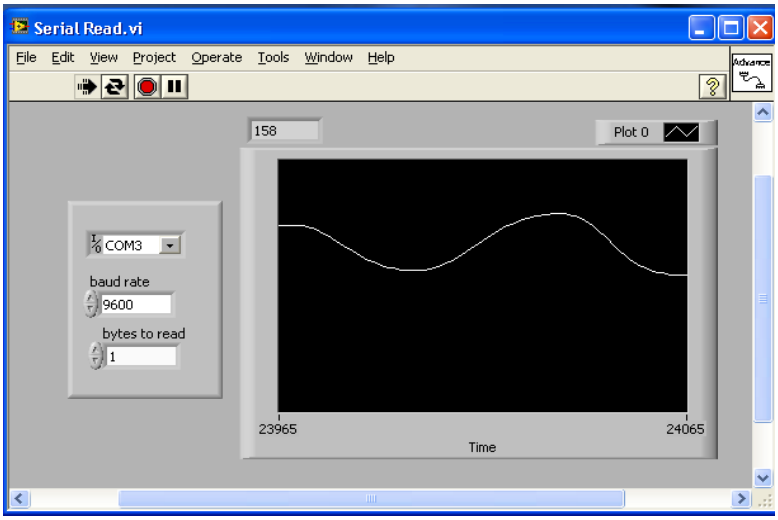

```
uses flexunit;
var
    a: word;
    s:string[5];
begin
  UART1_Init(9600);
  lcd_init1;
  lcd_text(1,1,'Potentiometru');
  while true do begin
     a:= Adc_Read(7) div 4;
     Lcd_Word(2,1,a);
     UART1_Write(a);
     delay_ms(100);
  end;
end.
```

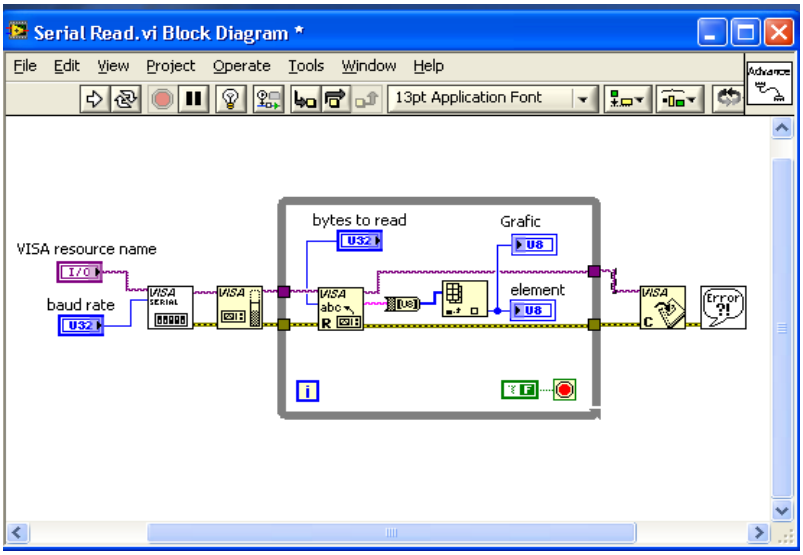
Aplicaţia 7 - Interfaţarea traductorului potenţiometric



Pentru vizualizarea datelor primite de la microcontroler pe interfaţa serială, a fost concepută o aplicaţie in mediul de programare LabVIEW

Această aplicaţie poate fi utilizată in continuare pentru afisarea grafică a semnalelor obţinute de la diversi senzori analogici (fotorezistenţă, termistor etc)






Traductoare şi senzori de poziţie

Senzori incrementali

❑ Senzorii digitali de poziţie oferă la ieşire unul sau mai multe trenuri de impulsuri dreptunghiulare de tensiune. De obicei senzorii digitali sunt preferaţi celor analogici datorită preciziei mai mari şi a insensibilităţii la zgomote şi perturbaţii.

❑ Senzorii digitali sunt incrementali sau absoluţi. Fiecare existând în două variante constructive: liniară, respectiv rotativă

❑ Procedeul de sesizare al incrementelor poate fi magnetic sau optic. Procedeul optic este cel mai răspândit datorită relativei simplităţi constructive şi a unor facilităţi de prelucrare a semnalelor.

FlexFORM - Program de formare profesionala flexibila pe platforme mecatronice

Traductoare şi senzori de poziţie

Senzori incrementali

Senzorii digitali de poziţie oferă la ieşire unul sau mai multe trenuri de impulsuri dreptunghiulare de tensiune. De obicei senzorii digitali sunt preferaţi celor analogici datorită preciziei mai mari şi a insensibilităţii la zgomote şi perturbaţii.

Senzorii digitali sunt *incrementali* sau ***absoluţi,*** fiecare existând în două variante constructive: liniară, respectiv rotativă.

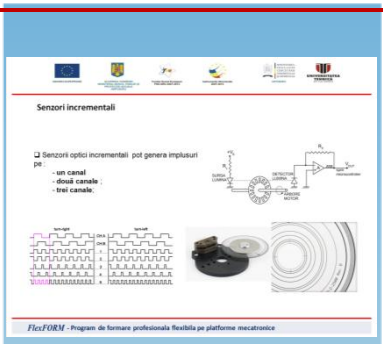
În principiu, un senzor digital oferă la ieşire un număr de impulsuri dreptunghiulare pentru fiecare unitate de deplasare parcursă de elementul mobil. Mărimea măsurată nu poate fi determinată la un moment dat (nu este o poziţie). Contorizând însă impulsurile de ieşire ale senzorului, într-un anumit interval de timp, se obţine un număr pe baza căruia se calculează deplasarea efectuată de elementul mobil în intervalul de timp respectiv. În concluzie, senzorii digitali incrementali permit măsurarea unor deplasări relative.

Procedeul de sesizare al incrementelor poate fi magnetic sau optic. Procedeul optic este cel mai răspândit datorită relativei simplităţi constructive şi a unor facilităţi de prelucrare a semnalelor.

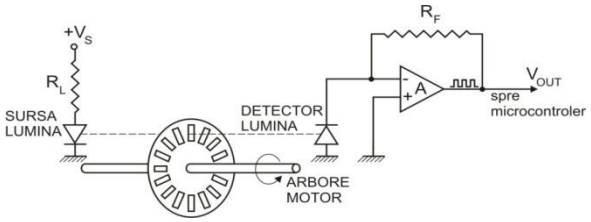
Discul sau rigla contine o retea optică de zone active, alternate cu interstiţii de aceeasi lăţime. Zonele active se disting de interstiţii, fie prin transparenţă (procedeu de măsurare de tip diascopic), fie prin puterea de reflecţie (procedeu de masurare episcopic).

Citirea este realizată de un cap (palpator) cu o grilă de urmărire (scanare) prevazută cu ferestre în dreptul fiecărei piste.

Senzori incrementali



Discul incremental conţine o reţea de zone active intercalate cu interstiţii, toate de aceeaşi lăţime. Lumina emisă de o sursă de lumină (LED) poate să treacă sau nu spre detectorul de lumină (o fotodiodă), funcţie de poziţia unghiulară a discului. Semnalul generat este amplificat şi transmis sub formă de impulsuri spre dispozitivul de calcul (microcontroler).

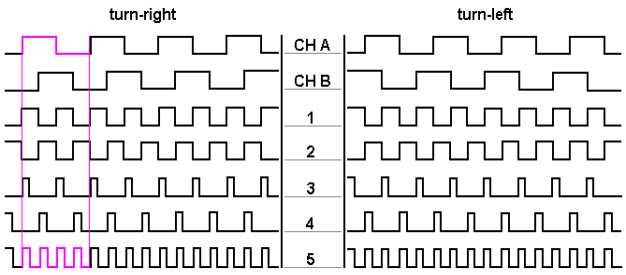






Senzorii optici incrementali pot genera implusuri pe:

- **un canal** ; in acest caz nu este posibilă determinarea sensului de rotaţie
- **două canale** ; semnalele sunt decalate cu p/2 unul faţă de celălalt fiind posibilă astfel determinarea sensului de rotaţie şi in plus multiplicarea cu 2 sau 4 a numărului de implusuri obţinute
- **trei canale**; permite generarea unui impuls la o rotaţie completă. Deci microcontrolerul poate face corecţia necesară dacă se pierd impulsuri şi ţine evidenţa numărului de rotaţii complete.



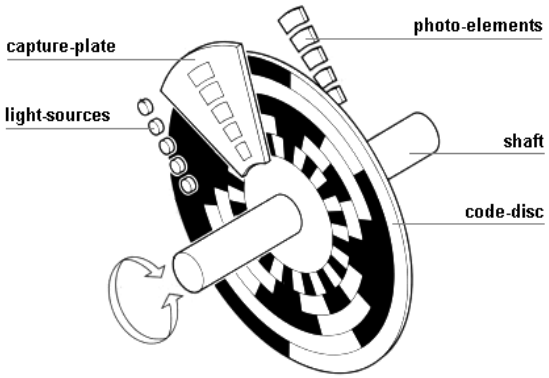
Senzori digitali absoluţi

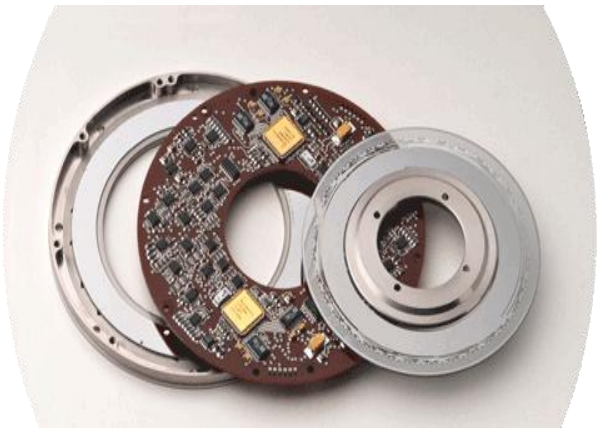


Un senzor digital absolut, alimentat cu tensiune, oferă la ieşire mai multe semnale numerice; nivelele logice ale acestora, citite la un moment dat, codifică poziţia elementului mobil al traductorului, fată de cel fix, la momentul respectiv.

Aceste traductoare se realizează în două variante constructive: liniară şi rotativă. Senzorul este compus dintr-un *cap de citire şi o riglă codificată (care* este elementul fix la traductorul liniar) sau un *disc codificat (elementul mobil la* traductorul rotativ). Rigla şi discul sunt realizate din sticlă transparentă, folosindu-se procedeul optic diascopic pentru citire *(*zonele active se deosebesc de interstiţii prin transparenţă).

Codificarea în cod binar (natural sau zecimal) sau în cod Gray se obţine prin trasarea a *m piste paralele (pe riglă) sau* concentrice (pe disc). Pistele conţin zone opace şi transparente, cu lărgimile şi modul de dispunere dependente de codul utilizat; numărul pistelor dictează rezoluţia măsurării.

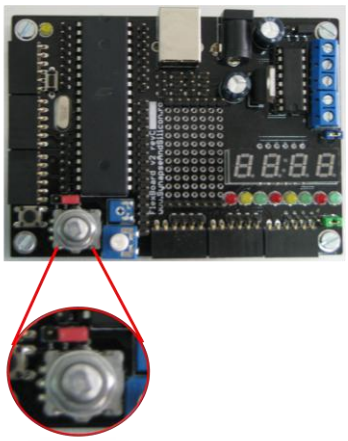


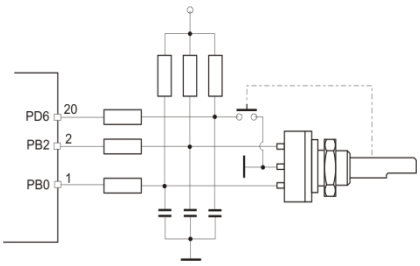


Aplicaţia 8 – Interfaţarea encoderului incremental (I)



Obiectivul acestei aplicaţii este interfaţarea senzorului incremental, determinarea sensului de rotaţie şi afişarea valori pe 8 LED-uri.






```
var a,b,av:byte;
    contor:byte;
begin
  ddrc:=$ff;
  while 1=1 do begin
    a:=pinb.0;
    b:=pinb.2;
    if (a<>av) and (a=1) then
      if b=0 then
        contor:=contor+1
        else
        contor:=contor-1;
    av:=a;
    portc:=contor;
  end;
end.
```

Aplicaţia 9 – Interfaţarea encoderului incremental (II)

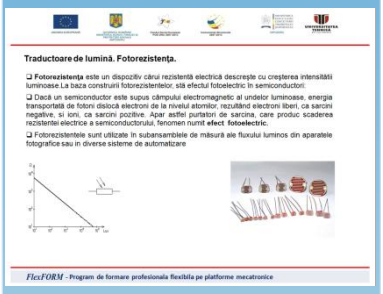

Aplicaţia 9 – Interfaţarea encoderului incremental

Obiectivul acestei aplicaţii este interfaţarea senzorului incremental, determinarea sensului de rotaţie şi afişarea numărului de impulsuri pe afişajul cu LCD.

Pe primul rând al LCD-ului va fi afişat numărul de impulsuri iar pe al doilea rând va fi afişat un text. (ex: TEST ENCODER)

```
uses flexunit;
var a,b,av:byte;
    contor:word;
    s:string[5];
begin
  Lcd_Init;
  Lcd_text(2,1,'Test encoder');
  contor:=0;
  while 1=1 do begin
    a:=pinb.0;
    b:=pinb.2;
    if (a<>av) and (a=1) then begin
      if b=0 then
        contor:=contor+1
      else
        contor:=contor-1;
      Lcd_word(1,1,contor);
    end;
    av:=a;
  end;
end.
```

Traductoare de lumină. Fotorezistenţa


Traductoare de lumină. Fotorezistența.

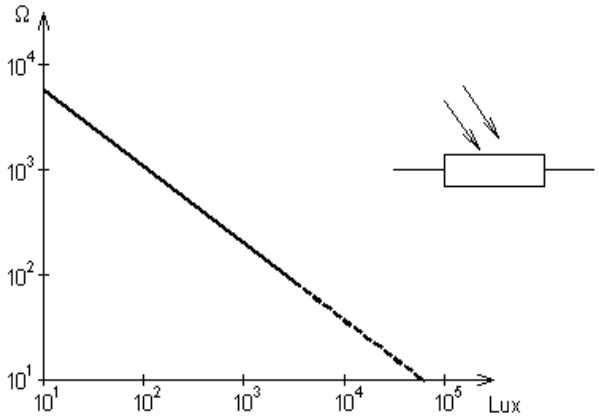
❑ Fotorezistența este un dispozitiv cărui rezistență electrică descreşte cu creşterea intensității luminoase.La baza construirii fotorezistentelor, stă efectul fotoelectric în semiconductori:

❑ Dacă un semiconductor este supus câmpului electromagnetic al undelor luminoase, energia transportată de fotoni dislocă electroni de la nivelul atomilor, rezultând electroni liberi, ca sarcini negative, si ioni, ca sarcini pozitive. Apar astfel purtători de sarcina, care produc scăderea rezistenţei electrice a semiconductorului, fenomen numit **efect fotoelectric**.

❑ Fotorezistentele sunt utilizate în subansamblele de măsură ale fluxului luminos din aparatele fotografice sau in diverse sisteme de automatizare

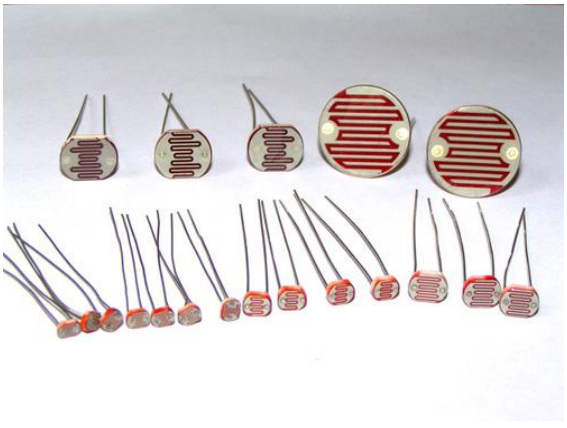
FlexFORM - Program de formare profesionala flexibila pe platforme mecatronice

Fotorezistenţa este un dispozitiv cărui rezistentă electrică descreşte cu creşterea intensitătii luminoase.La baza construirii fotorezistentelor, stă efectul fotoelectric în semiconductori:



Dacă un semiconductor este supus câmpului electromagnetic al undelor luminoase, energia transportată de fotoni dislocă electroni de la nivelul atomilor, rezultând electroni liberi, ca sarcini negative, si ioni, ca sarcini pozitive. Apar astfel purtători de sarcina, care produc scăderea rezistenţei electrice a semiconductorului, fenomen numit **efect fotoelectric**.

Fotorezistentele sunt utilizate în subansamblele de măsură ale fluxului luminos din aparatele fotografice sau in diverse sisteme de automatizare

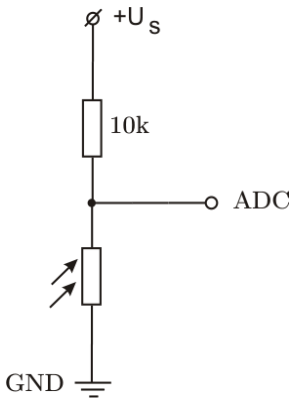


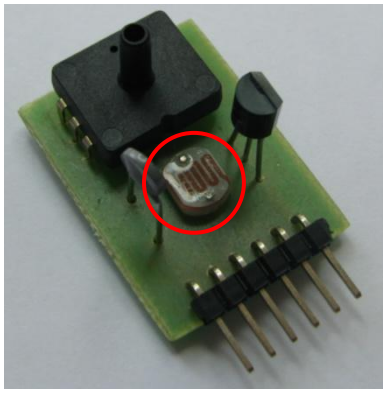
Aplicaţia 10 - Citirea valorii unei fotorezistenţe



Obiectivul acestei aplicaţii este citirea valorii unei fotorezistenţe, afisarea acesteia pe LCD şi pe calculator.

Schema de conectare la convertorul ADC este cea din figura alăturată.





Afişarea grafică,pe calculator, a valorilor citite se poate face utilizând programul in LabView prezentat in Aplicaţia 7

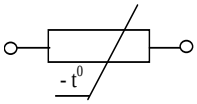
```
uses flexunit;
var
    a: word;
    s:string[5];
begin
UART1_Init(9600);
 lcd_init1;
 lcd_text(1,1,'Fotorezistenta');
while true do begin
     a:= Adc_Read(7) div 4;
     lcd_word(2,1,a);
     UART1_Write(a);
     delay_ms(100);
 end;
end.
```

Traductoare de temperatură. Termistorul

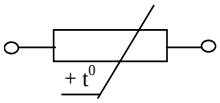


Termistorul este un dispozitiv semiconductor, realizat din amestecuri sinterizate din oxizi de mangan, cobalt, nichelul, fier, cupru, a cărui rezistentă variază semnificativ în funcţie de temperatură. Micşorarea sau creşterea rezistenţei este în strânsă corelaţie cu tipul termistorului, care poate fi:

- cu coeficientul de temperatură negativ, NTC la care rezistenţa scade cu creşterea temperaturii



- cu coeficientul de temperatură pozitiv, PTC la care rezistenţa creşte odată cu creşterea temperaturii



Legea de variatie a rezistentei in funcţie de temperatură pentru un termistor cu coeficientul de temperatură negativ, NTC este dată de relaţia:

$$R_T = A \cdot e^{\frac{B}{T}}$$

unde:

- R_T[K] este rezistenţa termistorului la temperatura T ;

- B[K] este o constantă de material; uzual B = (2000-5000)K;

- A(R_A)[Ω] constantă care depinde de tipul termistorului şi semnifică rezistenţa termistorului când temperatura tinde (ipotetic) spre infinit.



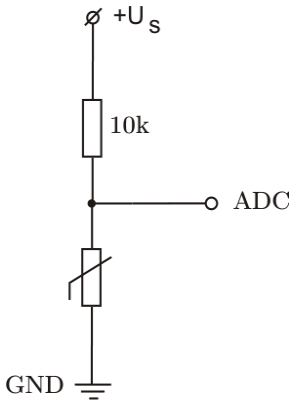


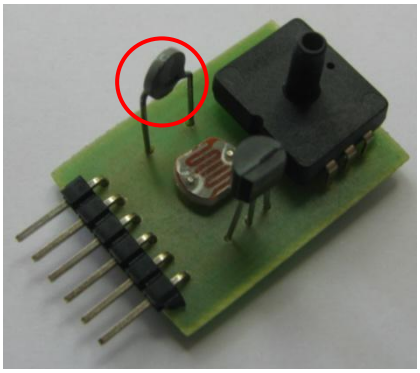
Aplicaţia 11 - Citirea valorii unui termistor NTC



Obiectivul acestei aplicaţii este citirea valorii unui termistor, afisarea acesteia pe LCD şi pe calculator.

Schema de conectare la convertorul ADC este cea din figura alăturată.





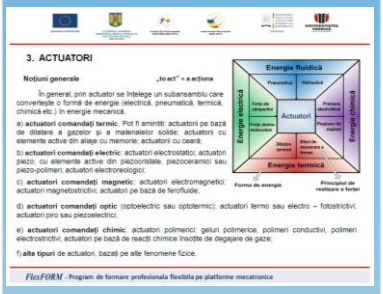
```
uses flexunit;
var
    a: word;
    s:string[5];
begin
  UART1_Init(9600);
  lcd_init1;
  lcd_text(1,1,'Termistor');
while true do begin
    a:= Adc_Read(7) div 2;
    Lcd_Word(2,1,a);
    UART1_Write(a);
    delay_ms(100);
  end;
end.
```

ACTUATORI

OBIECTIVE

- Prezentarea principalelor tipuri de actuatori utilizați în sistemele mecatronice
- Interfațarea principalelor tipuri de actuatori.
- Comanda motoarelor de curent continuu.
- Comanda motoarelor pas cu pas.
- Comanda servomecanismelor

Actuatori


3. ACTUATORI

Noţiuni generale „to act" = a acţiona

Energie fluidică

Pneumatică Hidraulică

Energie electrică

Actuatori

Energie chimică

Energie termică

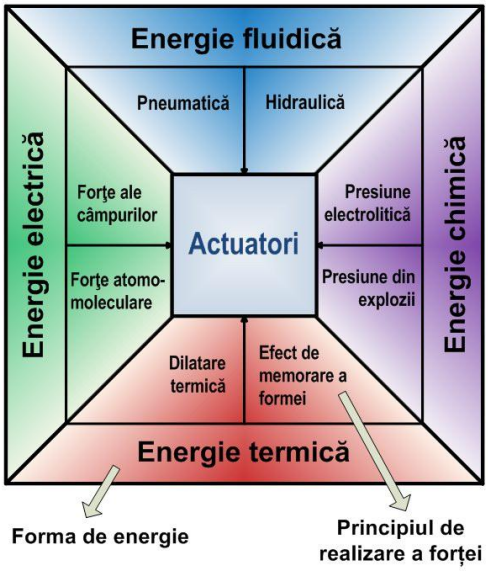
Forma de energie

Principiul de realizare a forţei

FlexFORM - Program de formare profesională flexibilă pe platforme mecatronice

Noţiuni generale

Pentru dezvoltarea forţelor şi momentelor şi realizarea mişcărilor în sistemele mecatronice se utilizează sisteme de acţionare de cele mai diferite tipuri şi forme de energie. Pentru cuprinderea tuturor acestor dispozitive tehnice într-o singură noţiune, se utilizează, termenul de actuator (de la verbul englez „to act" = a acţiona)

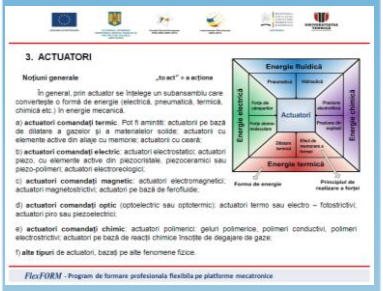


În general, prin actuator se înţelege un subansamblu care converteşte o formă de energie (electrică, pneumatică, termică, chimică etc.) în energie mecanică. Structura acestuia nu mai poate fi descompusă în sub-structuri decât cu riscul de a pierde capacitatea de generare a mişcării. Mişcarea simplă, elementară generată de un actuator poate fi utilizată direct pentru un anumit scop sau poate fi transmisă, transformată şi amplificată, sau corelată cu mişcarea obţinută de la alţi actuatori.

Acţionarea propriu-zisă este obţinută pe trei căi distincte, interacţiunea câmpurilor, interacţiune mecanică şi deformaţii limitate.

Interacţiunea câmpurilor magnetice, a curentului electric cu câmpuri magnetice precum şi interacţiunea sarcinilor electrice permit materializarea unor actuatori care au cursă teoretic nelimitată sau limitată, putând fi concepuţi ca actuatori liniari sau rotativi.

Acţionarea bazată pe interacţiunea mecanică presupune asigurarea fluxului de energie prin intermediul unui agent fizic, de regulă un lichid sau un gaz, a cărui presiune sau debit determină deplasarea sau deformarea unor elemente active. Astfel, actuatorii cu elemente deplasabile rotative sunt motoarele cu palete cu rotaţie parţială sau totală, turbinele iar actuatorii liniari de acest tip sunt reprezentaţi de cilindri. Actuatorii liniari cu elemente deformabile sunt cei cu tub flexibil şi cei cu tub Bourdon iar cei de rotaţie au în structură elemente active sub formă de tub răsucit sau tub anizotropic, curbat.


3. ACTUATORI

Noţiuni generale

Actuatori

Energie fluidică

Energie termică

Energie electrică

Energie chimică

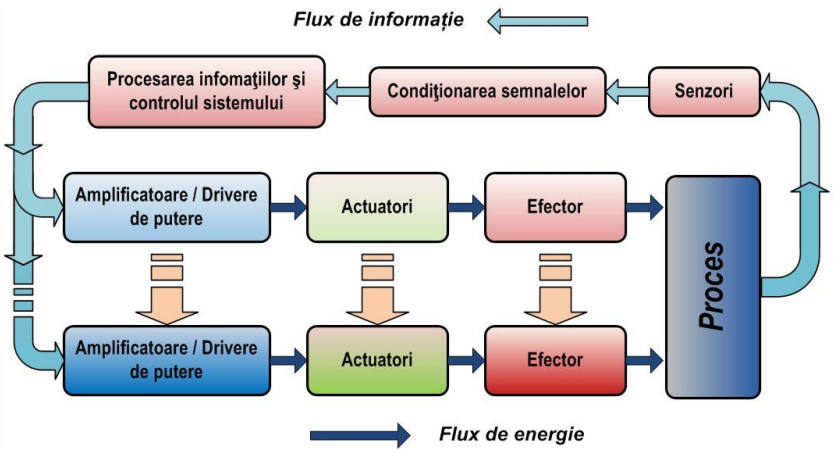
FlexFORM - Program de formare profesională flexibilă pe platforme mecatronice

Actuatorii a căror funcţionare se bazează pe deformaţiile liniare şi unghiulare limitate au în structură unul sau mai multe elemente din materiale "inteligente" - materiale care au capacitatea de a se deforma controlat, confecţionate sub formă lamelară dreaptă sau curbată preformată, cilindrică, formă de disc, bară, bară de torsiune, membrană, arc spiral sau elicoidal, astfel încât se obţin atât deplasări liniare cât şi deplasări unghiulare.

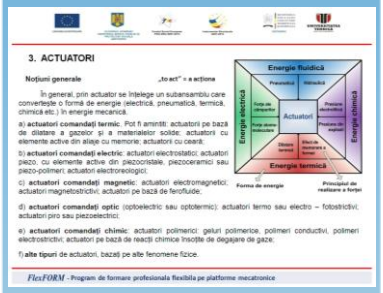
O caracteristică esenţială a elementelor din structura actuatorilor este faptul că au un triplu rol:

- rol structural (de a prelua încărcările şi sarcinile transmise);
- rol senzorial (oferind suplimentar posibilitatea integrării în structură a senzorilor şi traductorilor, pentru a obţine un control în buclă închisă);
- rol de acţionare.

În privinţa integrării într-un sistem, un actuator conectează partea de procesare a informaţiei din unitatea de control a sistemului cu procesul care trebuie investigat



Sistematizarea actuatorilor utilizaţi în mecatronică


3. ACTUATORI

Notiuni generale „to act" = a actiona

În general, prin actuator se înţelege un subansamblu care converteşte o formă de energie (electrică, pneumatică, termică, chimică etc.) în energie mecanică.

a) **actuatori comandaţi termic**. Pot fi amintiti: actuatori pe bază de dilatare a gazelor şi a materialelor solide; actuatori cu elemente active din aliaje cu memorie; actuatori cu ceară;

b) **actuatori comandaţi electric**: actuatori electrostatici; actuatori piezo, cu elemente active din piezocristale, piezoceramici sau piezo-polimeri; actuatori electroreologici;

c) **actuatori comandaţi magnetic**: actuatori electromagnetici; actuatori magnetostrictivi; actuatori pe bază de ferofluide;

d) **actuatori comandaţi optic** (optoelectric sau optotermic): actuatori termo sau electro – fotostrictivi; actuatori piro sau piezoelectrici;

e) **actuatori comandaţi chimic**: actuatori polimerici geluri polimerice, polimeri conductivi, polimeri electrostrictivi; actuatori pe bază de reacţii chimice însoţite de degajare de gaze;

f) **alte tipuri** de actuatori, bazaţi pe alte fenomene fizice.

Energie fluidică
Pneumatică
Hidraulică
Actuatori
Energie electrică
Energie chimică
Energie termică
Forma de energie
Principiul de realizare a forţei

FlexFORM - Program de formare profesională flexibilă pe platforme mecatronice

În funcţie de semnalul de intrare folosit pentru comanda actuatorilor, aceştia pot fi clasificaţi astfel:

a) ***actuatori comandaţi termic*** (prin intermediul unui flux de căldură):

- actuatori pe bază de dilatare a gazelor şi a materialelor solide: actuatori cu elemente active bimetalice sau actuatori pe bază de transformare de fază;

- actuatori cu elemente active din aliaje cu memorie;

- actuatori cu ceară;

b) ***actuatori comandaţi electric*** (prin intermediul intensităţii câmpului electric):

- actuatori electrostatici;

- actuatori piezo, cu elemente active din piezocristale, piezoceramici sau piezo-polimeri;

- actuatori electroreologici;

c) ***actuatori comandaţi magnetic*** (prin intermediul inducţiei câmpului magnetic):

- actuatori electromagnetici;

- actuatori magnetostrictivi;

- actuatori pe bază de ferofluide;

d) ***actuatori comandaţi optic*** (optoelectric sau optotermic):

- actuatori termo sau electro - fotostrictivi;

- actuatori piro sau piezoelectrici;

e) ***actuatori comandaţi chimic***:

- actuatori polimerici: geluri polimerice, polimeri conductivi, polimeri electrostrictivi;

- actuatori pe bază de reacţii chimice însoţite de degajare de gaze;

f) ***alte tipuri de actuatori***, bazaţi pe alte fenomene fizice.



Exemple :

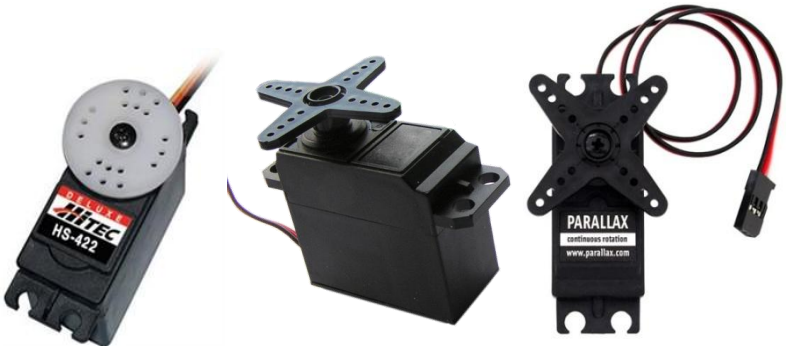
Motoare de curent continuu



Motoare pas cu pas



Servomecanisme



Motorul de curent continuu


Motorul de curent continuu

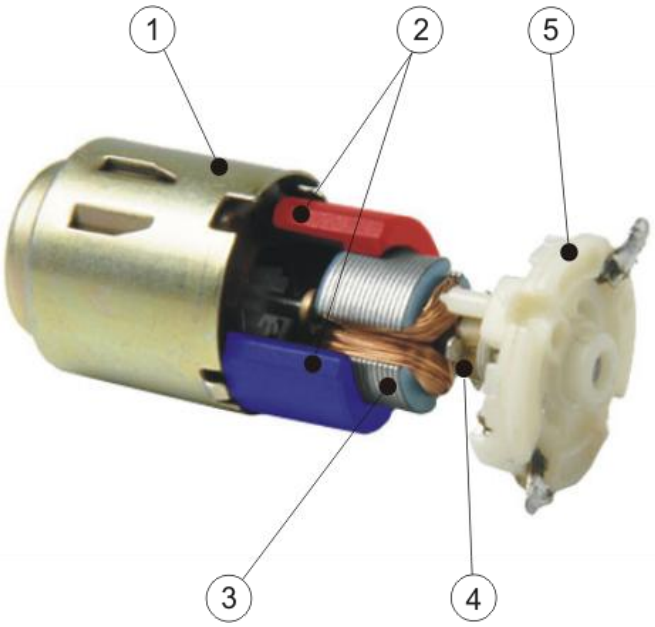
❑ Motorul de curent continuu este construit din două părţi componente: **stator** şi **rotor**.

❑ *Statorul* este partea fixă a motorului, în general exterioară, ce include carcasa 1 şi magneţii permanenţi 2.

❑ *Rotorul* este partea mobilă a motorului, plasată de obicei în interior. Este format dintr-un ax şi o armătură ce susţine înfăşurarea rotorică 3. Pe axul motorului este situat un colector 4 ce schimbă sensul curentului prin înfăşurarea rotorică. Tensiunea de alimentare este aplicată înfăşurărilor rotorului prin intermediul unui sistem de perii fixate pe capacul 5. Între stator şi rotor există o distanţă numită întrefier

FlexFORM - Program de formare profesionala flexibila pe platforme mecatronice

Motorul de curent continuu este construit în principal din două părţi componente: **stator** şi **rotor**

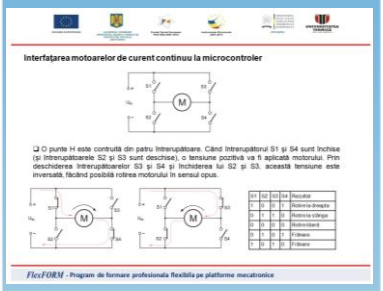


Statorul este partea fixă a motorului, în general exterioară, ce include carcasa 1 şi magneţii permanenţi 2.

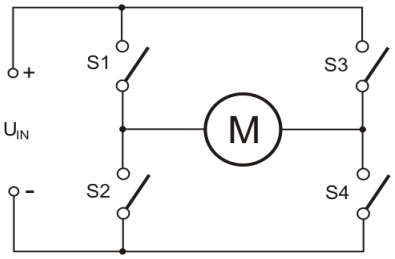
Rotorul este partea mobilă a motorului, plasată de obicei în interior. Este format dintr-un ax şi o armătură ce susţine înfăşurarea rotorică 3. Pe axul motorului este situat un *colector* 4 ce schimbă sensul curentului prin înfăşurarea rotorică. Tensiunea de alimentare este aplicată înfăşurărilor rotorului prin intermediul unui sistem de perii fixate pe capacul *5.* Între stator şi rotor există o distanţă numită *întrefier* .

Turaţia motorului este proporţională cu tensiunea aplicată înfăşurării rotorice şi invers proporţională cu câmpul magnetic de excitaţie. Turaţia se poate regla prin variaţia tensiunii aplicate motorului de la o valoare minimă până la valoarea nominală. Cuplul dezvoltat de motor poate fi reglat prin variaţia curentului aplicat infăşurării rotorice. Schimbarea sensului de rotaţie se face prin schimbarea polarităţii tensiunii de alimentare.

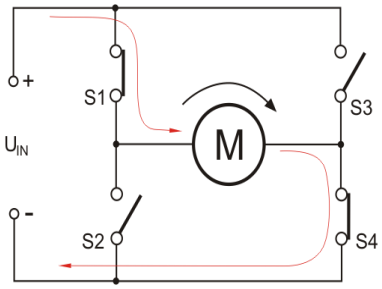
Interfaţarea motoarelor de curent continuu la microcontroler

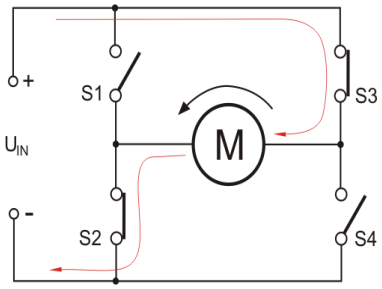


Datorită tensiunii şi curentului de valori foarte mici furnizate la ieşirea microcontrolerelor, este nevoie de circuite de amplificare pentru acţionarea motoarelor de curent continuu. O schemă simplă şi eficientă pentru comanda acestor motoare este reprezentată de “puntea H” (în engleză **H-bridge**).



O punte H este construită din patru întrerupătoare. Când întrerupătorul S1 şi S4 sunt închise (şi întrerupătoarele S2 şi S3 sunt deschise), o tensiune pozitivă va fi aplicată motorului. Prin deschiderea întrerupătoarelor S3 şi S4 şi închiderea lui S2 şi S3, această tensiune este inversată, făcând posibilă rotirea motorului în sensul opus.

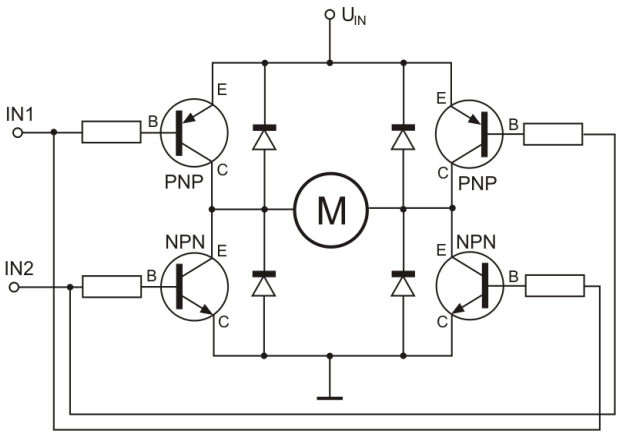




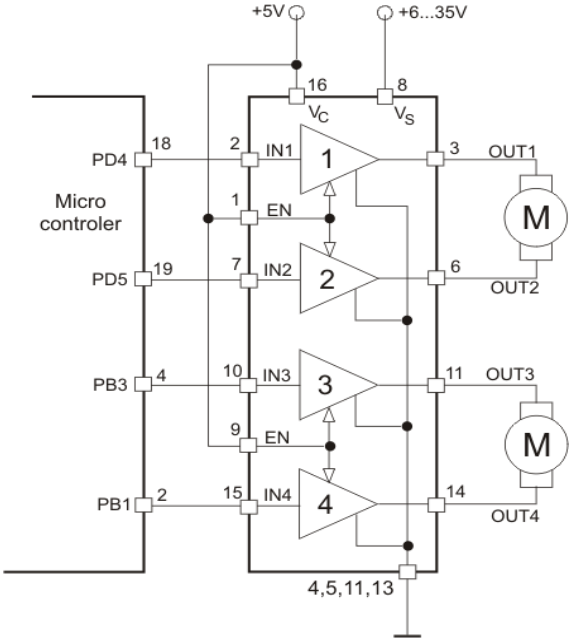
Aranjarea de tip punte H este în general folosită pentru a inversa polaritatea motorului, însă poate fi folosită şi pentru frânarea motorului (motorul se opreşte brusc datorită scurtcircuitării terminalelor sale) sau pentru a lăsa motorul să se rotească liber până la oprire

S1	S2	S3	S4	Rezultat
1	0	0	1	Rotire la dreapta
0	1	1	0	Rotire la stânga
0	0	0	0	Rotire liberă
0	1	0	1	Frânare
1	0	1	0	Frânare

Punţile H pot fi contruite din componente electronice sau se găsesc într-un singur circuit integrat.



Circuitul integrat **L293D** conţine 2 astfel de punţi care pot comanda motoare alimentate cu max 35V şi 600 mA. Fiecare din cele două punţi H are două terminale de intrare (**IN**put) şi un terminal de activare (**EN**able). Când terminalul EN este legat la 5V puntea H este activă. Dacă intrarea IN1 este in 1 logic (+5V) iar intrarea IN2 este in 0 logic (0V) motorul se roteşte; dacă stările celor două intrări sunt inversate , motorul se va roti în sens contrar. Când ambele intrări sunt în 0 logic motorul se opreşte iar dacă ambele sunt în 1 logic atunci axul motorului este frânat. Un exemplu de conectare a circuitului L293 la microcontroler este prezentat în figura de mai jos



Aplicaţia 1 – Acţionarea unui motor de curent continuu în sens orar



Scopul aplicaţiei este acţionarea unui motor de curent continuu, la apăsarea butonului, în sensul acelor de ceasornic. La eliberarea butonului, motorul se va opri.

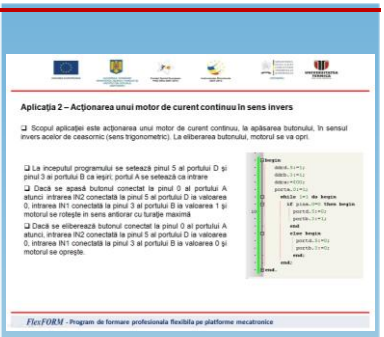
La inceputul programului se setează pinul 5 al portului D şi pinul 3 ai portului B ca ieşiri; portul A se setează ca intrare

Dacă se apasă butonul conectat la pinul 0 al portului A atunci intrarea IN2 conectată la pinul 5 al portului D ia valoarea 1, intrarea IN1 conectată la pinul 3 al portului B ia valoarea 0 şi motorul se roteşte in sens orar cu turaţie maximă

Dacă se eliberează butonul conectat la pinul 0 al portului A atunci, intrarea IN2 conectată la pinul 5 al portului D ia valoarea 0, intrarea IN1 conectată la pinul 3 al portului B ia valoarea 0 şi motorul se opreşte

```
begin
   ddrd.5:=1;
   ddrb.3:=1;
   ddra:=$00;
   porta.0:=1;
     while 1=1 do begin
       if pina.0=0 then begin
         portd.5:=1;
         portb.3:=0;
       end
       else begin
         portd.5:=0;
         portb.3:=0;
       end;
     end;
end.
```

Aplicaţia 2 – Acţionarea unui motor de curent continuu în sens trigonometric



Scopul aplicaţiei este acţionarea unui motor de curent continuu, la apăsarea butonului, în sensul invers acelor de ceasornic (sens trigonometric). La eliberarea butonului, motorul se va opri.

La inceputul programului se setează pinul 5 al portului D şi pinul 3 ai portului B ca ieşiri; portul A se setează ca intrare.

Dacă se apasă butonul conectat la pinul 0 al portului A atunci intrarea IN2 conectată la pinul 5 al portului D ia valoarea 0, intrarea IN1 conectată la pinul 3 al portului B ia valoarea 1 şi motorul se roteşte in sens antiorar cu turaţie maximă.

Dacă se eliberează butonul conectat la pinul 0 al portului A atunci, intrarea IN2 conectată la pinul 5 al portului D ia valoarea 0, intrarea IN1 conectată la pinul 3 al portului B ia valoarea 0 şi motorul se opreşte.

```
begin
   ddrd.5:=1;
   ddrb.3:=1;
   ddra:=$00;
   porta.0:=1;
     while 1=1 do begin
       if pina.0=0 then begin
         portd.5:=0;
         portb.3:=1;
       end
       else begin
         portd.5:=0;
         portb.3:=0;
       end;
     end;
end.
```

Intreruperi. Tratarea întreruperilor


Intreruperi. Tratarea intreruperilor

O întrerupere reprezintă un semnal sincron sau asincron de la un periferic ce semnalizează apariţia unui eveniment care trebuie tratat de către procesor. Tratarea intreruperii are ca efect suspendarea firului normal de execuţie al unui program şi lansarea în execuţie a unei rutine de tratare a intreruperii (RTI) .

Întreruperile hardware au fost introduse pentru a se elimina buclele pe care un procesor ar trebui să le facă în aşteptarea unui eveniment de la un periferic. Folosind un sistem de intreruperi, perifericele pot atenţiona procesorul în momentul producerii unei întreruperi (IRQ), acesta din urmă fiind liber să-şi ruleze programul normal în restul timpului şi să işi întrerupă execuţia doar atunci când este necesar

Înainte de a lansa în execuţie o rutină de tratare a intreruperii (RTI), procesorul trebuie să aibă la dispoziţie un mecanism prin care să salveze starea în care se află în momentul apariţiei întreruperii. Aceasta se face prin salvarea într-o memorie, a registrului contor de program (Program Counter), a registrelor de stare precum şi a tuturor variabilelor din program care sunt afectate de execuţia RTI. La sfârşitul execuţiei RTI starea anterioară a registrelor este refacută şi programul principal este reluat din punctul de unde a fost intrerupt.

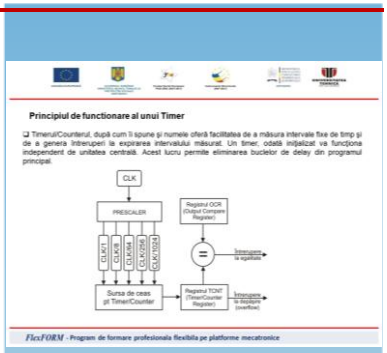
Perifericele care pot genera întreruperi la ATMega16 sunt timerele, interfaţa serială (USART), convertorul analog-digital (ADC), controlerul de memorie EPROM, comparatorul analog şi interfaţa serială I^2C. Deasemenea, procesorul poate să primească cereri de întreruperi externe din trei surse (INT0, 1 şi 2) ce corespund unui număr egal de pini exteriori:

INT0 pinul 16 (PD2)

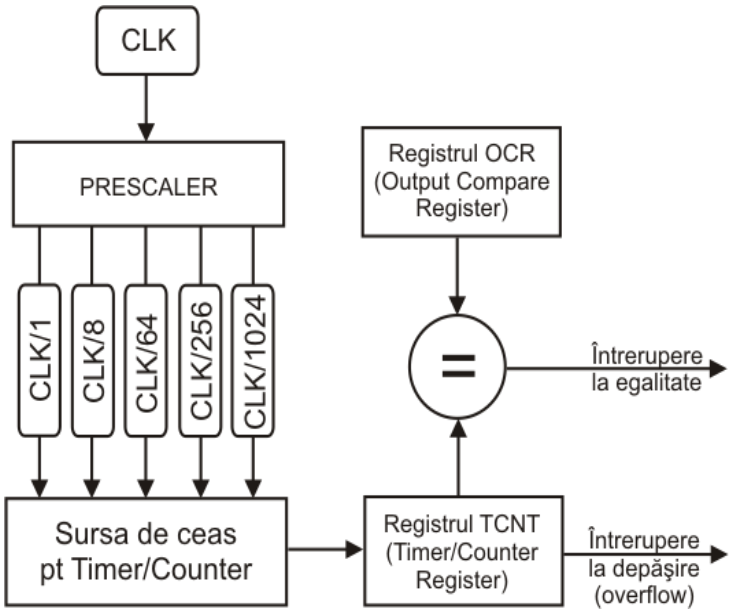
INT1 pinul 17 (PD3)

INT2 pinul 3 (PB2)

Principiul de functionare al unui Timer



Timerul/Counterul, după cum îi spune şi numele oferă facilitatea de a măsura intervale fixe de timp şi de a genera întreruperi la expirarea intervalului măsurat. Un timer, odată iniţializat va funcţiona independent de unitatea centrală. Acest lucru permite eliminarea buclelor de delay din programul principal.



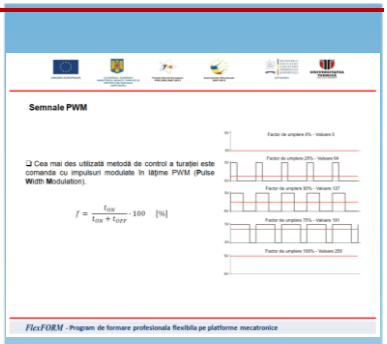
Principiul de funcţionare a unui Timer poate fi descris astfel :

- Prescaler-ul divizează în funcţie de necesităţile aplicaţiei frecvenţa de ceas şi odată cu divizarea incrementează registrul numărător TCNT .

- La fiecare incrementare a registrului numărător are loc o comparaţie între acest registru şi o valoare stocată în registrul OCR. Această valoare poate fi incărcată prin software de utilizator. Dacă are loc egalitatea se generează o intrerupere, în caz contrar incrementarea continuă

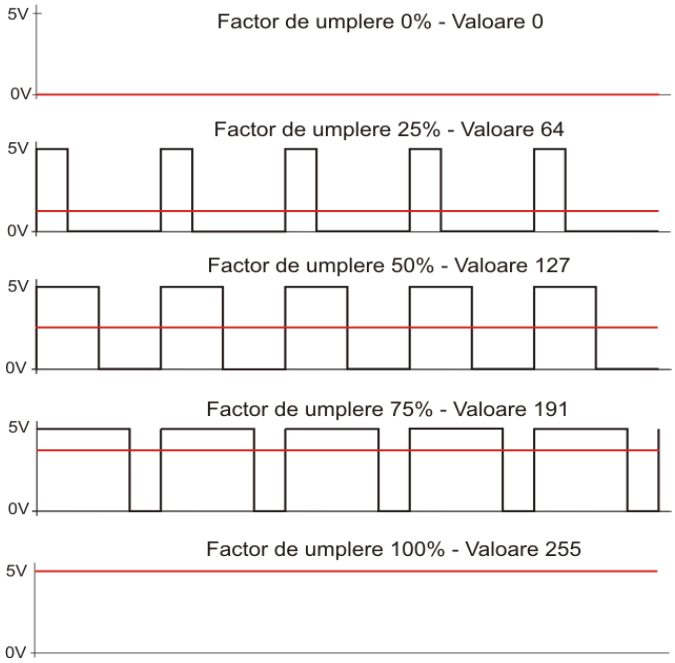
Timerele sunt prevăzute cu mai multe canale astfel că se pot desfăşura în paralel mai multe numărători. ATmega16 este prevăzut cu 3 unităţi de timer: două pe 8 biţi şi una pe 16 biţi.

Semnale PWM



Pentru a controla turaţia motoarelor de curent continuu există mai multe metode. Cea mai simplă ar fi înserierea unui rezistor variabil in circuitul de alimentare al motorului. Prin modificarea valorii rezistenţei se modifică tensiunea aplicată motorului. Variaţia tensiunii nu este proporţională cu variaţia rezistenţei ci depinde şi de curentul absorbit de motor. Această metodă nu este eficientă, având in vedere faptul că se pierde foarte multă energie prin disipare de căldură.

Cea mai des utilizată metodă de control a turației este comanda cu impulsuri modulate în lăţime PWM (**P**ulse **W**idth **M**odulation).



Variaţia turaţiei motorului se face prin creşterea sau micşorarea timpului în care semnalul de comandă are valoarea 1 logic (t_{ON}). Astfel la un factor de umplere de 100%, motorul are turaţia maximă, deci t_{ON} este maxim, iar t_{OFF} este zero. La un factor de umplere de 50 %, motorul are jumatate din turaţia nominală deci $t_{ON} = t_{OFF.}$ Iar la un factor de umplere de 0 % motorul este oprit deci t_{OFF} este maxim, iar t_{ON} este zero.

$$f = \frac{t_{ON}}{t_{ON} + t_{OFF}} \cdot 100 \quad [\%]$$

Majoritatea microcontrolerelor pot genera semnale PWM. Pentru măsurarea timpului cât semnalul este în 1 logic şi al timpului cât stă în 0 logic, se foloseşte un Timer.

Pentru a genera un astfel de semnal numărătorul este incrementat periodic şi este resetat la sfârşitul fiecărei perioade a PWM-ului. Când valoarea numărătorului este mai mare decat valoarea de referinţă, ieşirea PWM trece din starea 1 logic in starea 0 logic (sau invers)

In cazul microcontrolerului ATmega16 pentru a genera semnale PWM, este nevoie de programarea a doi regiştrii TCCR1A şi TCCR1B. Regiştrii de control (TCCR1A/B) sunt regiştrii de 8 biti, cu rol de a configura Timerul pentru un anumit mod de functionare.

7	6	5	4	3	2	1	0	
COM1A1	COM1A0	COM1B1	COM1B0	FOC1A	FOC1B	WGM11	WGM10	TCCR1A

7	6	5	4	3	2	1	0	
ICNC1	ICES1	–	WGM13	WGM12	CS12	CS11	CS10	TCCR1B

Pentru a genera un semnal PWM standard, trebuie setaţi biţii COM1A1 şi COM1B1 din registrul TCCR1A. Aceasta înseamnă că li se va atribui valoarea 1 logic. Pentru stabili numărul de biti corespunzător factorului de umplere al PWM-ul trebuie setaţi biţii WGM10 şi WGM12. Dacă mărimea factorului de umplere trebuie exprimată printr-un număr intreg pe 10 biti atunci celor doi biţi li se atribuie valoarea1. Pentru a stabili frecvenţa semnalului se poate folosi un divizor de frecvenţă (prescaler) Dacă se utilizează un prescaler de 8, atunci bitul CS11 are valoarea 1.

Astfel, registrul TCCR1A are in final valoarea

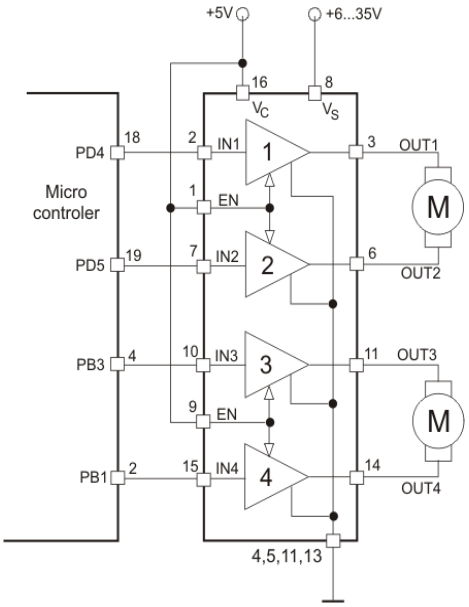
TCCR1A := 10100001 ; sau **TCCR1A := $A1 ;**

iar TCCR1B:

TCCR1B := 00001010 ; sau **TCCR1B := $0A.**

Pentru a uşura generarea semnalelor PWM şi a elimina necesitatea programării celor doi regiştrii de control, in librăria ***flexunit*** a fost inclusă o procedură care poate fi apelată cu instrucţiunea *Init_pwm.*

Circuitul L293 de pe placa de dezvoltare permite comanda a două motoare de curent continuu.





Pentru a comanda cele două motoare se utilizează instrucţiunile:

- ***Init_pwm*** care initializează cei doi regiştii de control TCCR1A şi TCCR1B
- ***motor0(x)***
- ***motor1(y)***

unde:

- *motor0* reprezintă motorul conectat la pinii 3 şi 6
- *motor1* este motorul conectat la pinii 11 şi 14
- ***x*** şi ***y*** sunt numere cuprinse intre -127 şi 127 şi semnifică valoarea factorului de umplere al semnalului PWM.

Dacă ***x*** şi/sau ***y*** au valoarea – 127 atunci motorul se va invârti cu turaţie maximă de exemplu în sens orar; dacă au valoarea 127 se va invârti cu turaţie maximă in celălalt sens iar dacă au valoarea 0 motorul nu se roteşte.

Aplicaţia 3 – Comanda bidirecţională a motorului de curent continuu



Scopul aplicaţiei este comanda bidirecţională a unui motor de curent continuu cu posibilitatea reglării vitezei.

La inceputul programului iniţializează comunicaţia cu LCD-ul şi cei doi regiştii PWM

Dacă se apasă butonul conectat la pinul 6 al portului D atunci motorul se roteşte in sens antiorar cu o turaţie corespunzătoare unui factor de umplere de 48, timp de 1,5 secunde, după care işi va schimba sensul şi se va roti cu aceeasi viteză tot 1,5 secunde. Apoi ciclul se repetă

Dacă se eliberează butonul conectat la pinul 6 al portului D atunci, motorul se opreşte.

Sensul se rotaţie al motorului va fi afişat pe LCD

```
program mcc_3;
uses flexunit;

begin
     init_pwm ;
     lcd_init;
  while 1=1 do begin
        if pind.6 =0 then begin
               motor0(48);
               lcd_text (1,1, 'MOTOR   CW');
               delay_ms(1500);

               motor0(-50);
               lcd_text (1,1, 'MOTOR  CCW');
               delay_ms(1500);
        end
             else begin
               motor0(0);
               lcd_text (1,1, 'MOTOR STOP');
             end
    end;
      delay_ms(10);
end.
```

Aplicaţia 4 – Variator de turaţie (I)



Scopul aplicaţiei este modificarea turaţiei unui motor de curent continuu.

La inceputul programului iniţializează comunicaţia cu LCD-ul şi cei doi regiştii PWM

Pe canalul 1 al ADC-ului (pinul 39) este conectat un potenţiometru. La mijlocul cursei potenţiometrului turaşia motorului este zero. La rotirea potenţiometrului într-un sens turaţia motorului va creşte proporţional cu unghiul de rotaţie. La schimbarea sensului de rotaţie al potenţiometrului se schimbă şi turaţia motorului.

Pe primul rând al afişajului LCD va fi afişat textul “Variator turaţie” iar pe al doilea valoarea factorului de umplere al semnalului de comandă PWM

```
program mcc_4;
uses flexunit;
var w:word;
pwm:integer ;
begin
   lcd_init;
   init_pwm;
   lcd_text(1,1,'Variator turatie');
   while true do begin
      w:=adc_read(1);
      delay_ms(10);
      pwm :=(w div 4)-128 ;
      motor0(pwm) ;
      lcd_int(2,1,pwm);
   end;
end.
```

Aplicaţia 5 – Variator de turaţie (II)



Scopul aplicaţiei este modificarea turaţiei unui motor de curent continuu cu ajutorul encoderului de pe placa de dezvoltare.

Unul din terminalele encoderului este legat la pinul PB2 adică la pinul care corespunde intreruperii externe ***int_2.*** Din acest motiv la inceputul programului este o procedură ce tratează intreruperea externă.

Dacă pinul 2 al portului B este 1 logic atunci variabila **in2** este decrementată cu 5 unităţi; dacă este in 0 logic atunci variabila va fi incrementată cu 5 unităţi. Practic se realizează un numărător care numără din 5 in 5 de la -128 pâna la 127

La rotirea encoderului într-un sens turaţia motorului va creşte proporţional cu unghiul de rotaţie. La schimbarea sensului de rotaţie al encoderului se schimbă şi turaţia motorului.

Pe primul rând al afişajului LCD va fi afişat textul “PWM test” iar pe al doilea valoarea factorului de umplere al semnalului de comandă PWM adică valoare obţinută prin rotirea encoderului.

```
program var_pro;
uses flexunit;
var in2:integer;

//Intreruperi ***********************
procedure int_2; org IVT_ADDR_INT2;
begin
  if pinb.0=1  then in2:=in2-5
               else in2:=in2+5;
  if in2>128   then in2:=127;
  if in2<-128  then in2:=-128;
end;

//Program principal *****************
begin
  init_pwm;
  init_int2;
  lcd_init;
  Lcd_text(1, 1, 'PWM Test');
  in2:=0;
  while true do begin
    lcd_int(2,1,in2);
    motor0(in2);
  end;
end.
```

Motorul pas cu pas

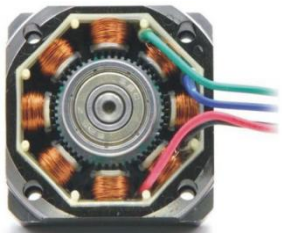


Motorul pas cu pas este un convertor electromecanic ce funcţionează pe principiul reluctanţei minime şi care realizează transformarea unui şir de impulsuri digitale într-o mişcare de rotaţie. Mişcarea rotorului motorului pas cu pas constă din deplasări unghiulare discrete, succesive, de mărimi egale, care reprezintă paşii motorului.

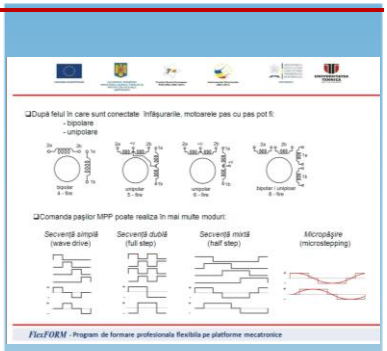
Numărul paşilor efectuaţi corespunde cu numărul impulsurilor de comandă aplicate fazelor motorului. Deplasarea unghiulară totală, constituită dintr-un număr de paşi egal cu numărul de impulsuri de comandă aplicat pe fazele motorului, determină poziţia finală a rotorului. Această poziţie este memorată până la aplicarea unui nou impuls de comandă. Univocitatea conversiei impulsuri - deplasare, asociată cu memorarea poziţiei, fac din motorul pas cu pas un bun element de execuţie, integrat în sistemele de reglare a poziţiei în circuit deschis.

Viteza unui motor pas cu pas poate fi reglată în limite largi prin modificarea frecvenţei impulsurilor de intrare. De exemplu, dacă pasul unghiular al motorului este 1,8° numărul de impulsuri necesare efectuării unei rotaţii complete este 200, iar pentru un semnal de intrare cu frecvenţa de 400 impulsuri pe secundă turaţia motorului este de 120 rotaţii pe minut.

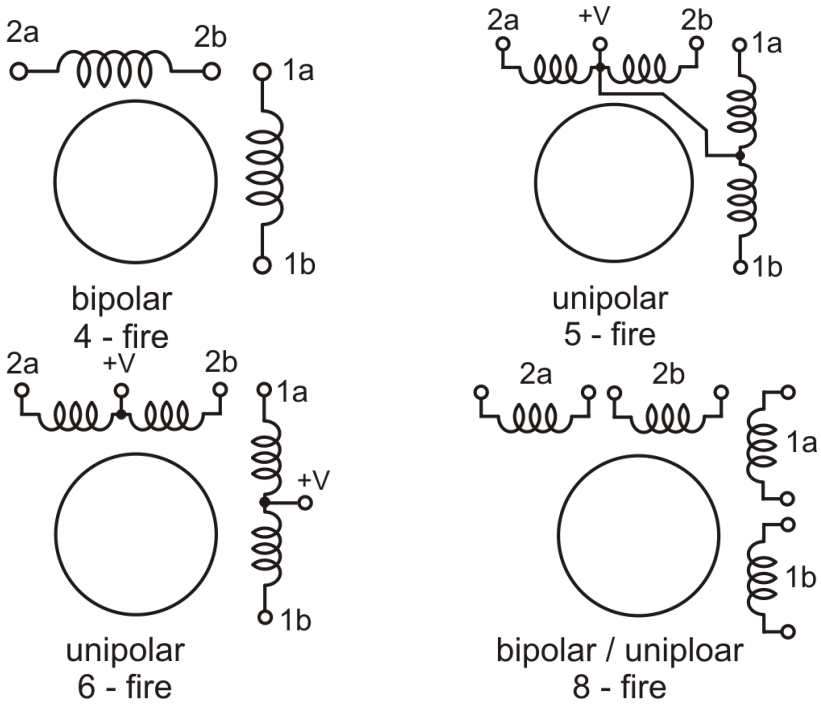




Motorul pas cu pas hibrid este o combinaţie a primelor două tipuri, fiind varianta de motor utilizată în marea majoritate a aplicaţiilor. Constructiv, rotorul motorului este constituit dintr-un magnet permanent cu mai multi poli şi un stator din material feromagnetic pe care sunt realizate infăşurările. Unghiul de rotaţie (pasul motorului) este determinat de relaţia existentă între numărul de poli ai statorului şi numărul de poli ai rotorului. La apariția unui semnal de comandă pe unul din polii statorici rotorul se va deplasa până când polii săi se vor alinia în dreptul polilor opuși statorici. Rotirea se va face practic din pol în pol



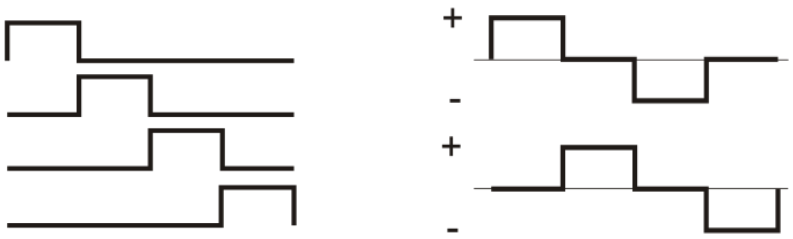
După felul în care sunt conectate înfăşurarile, motoarele pas cu pas pot fi: bipolare sau unipolare

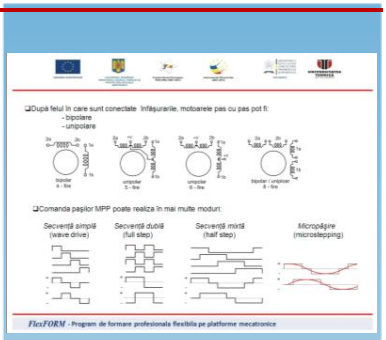


În cazul motoarelor pas cu pas unipolare fiecare dintre cele două bobine are o priză mediană care se conectează la borna pozitivă a sursei de alimentare. Mişcarea de rotaţie este generată prin legarea succesivă la masă a câte unei înfăşurări statorice. În cazul motorului bipolar mişcarea are loc prin schimbarea sensului curentului in înfăşurările statorice.

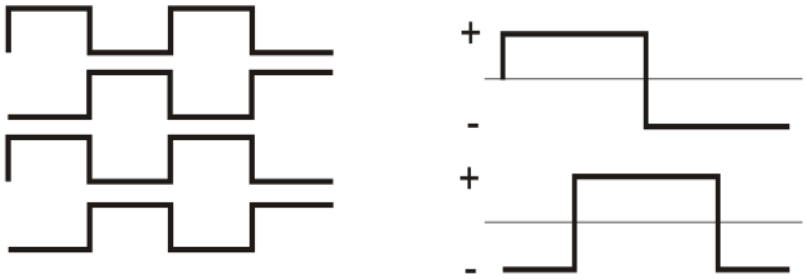
Comanda paşilor MPP poate realiza în mai multe moduri:

- *Secvenţă simplă* (wave drive), în care este alimentată pe rând câte o singură fază statorică . În acest caz cuplul motorului este aprox .30% de valoarea nominală Este utilizată în cazurile unde se impune un consum redus de energie

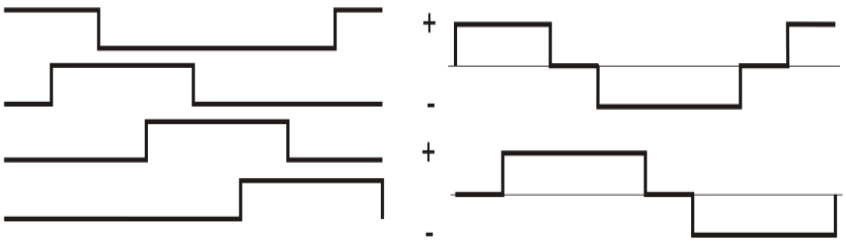




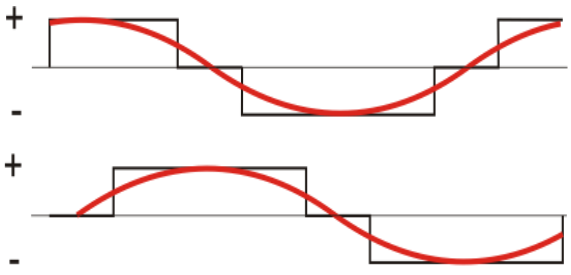
- *Secvenţă dublă* (full step) în care sunt alimentate simultan câte 2 faze. Este cea mai des utilizată metodă. Cuplul dezvoltat de motor şi frecvenţa de păşire au valori maxime.



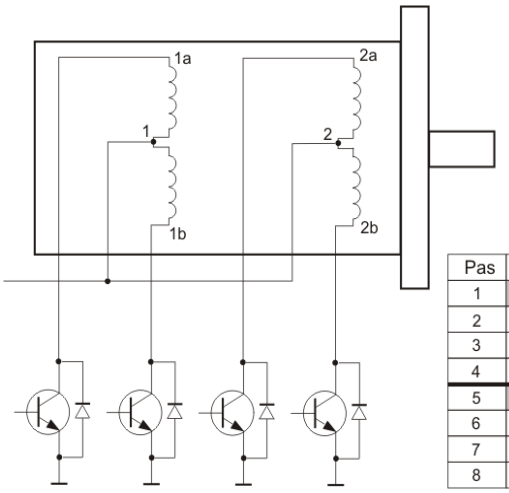
- *Secvenţă mixtă* (half step) presupune alimentarea, succesivă a unei faze, urmată de alimentare a 2 faze, şamd. În acest caz unghiul de rotaţie este jumătate din pasul nominal al motorului.



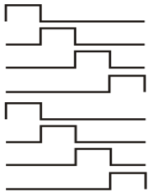
- *Micropăşire* (microstepping) este o metodă specială de control în poziţii intermediare celor obţinute prin primele trei metode. De exemplu, pot fi realizate poziţionări la 1/10, 1/16, 1/32, 1/125 din pasul motorului, prin utilizarea unor curenţi de comandă a fazelor cu valori diferite de cea nominală, astfel încât suma curenţilor de comandă prin cele două faze alăturate, comandate simultan să fie constantă, egală cu valoarea nominală. Cu ajutorul acestei metode sunt asigurate atât poziţionări fine, cât si operări line, fără şocuri, însă cuplul dezvoltat este mai mic decât în primele trei cazuri



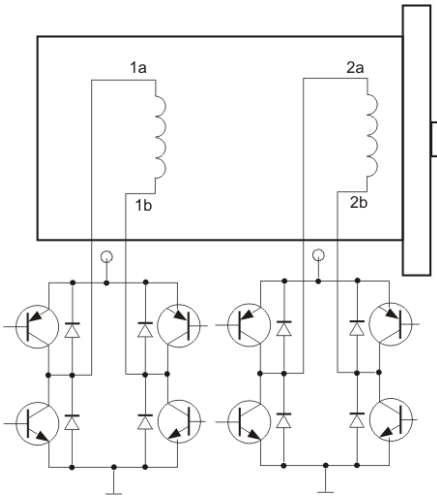
În cazul motoarelor pas cu pas unipolare fiecare dintre cele două bobine are o priză mediană care se conectează la borna pozitivă a sursei de alimentare Prin comanda succesivă a fiecarui tranzistor se pune la masă cate o bobină a motorului şi se generează mişcarea de rotaţie.



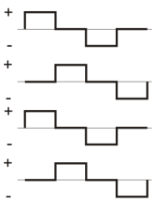
Pas	1a	1b	2a	2b
1	1	0	0	0
2	0	1	0	0
3	0	0	1	0
4	0	0	0	1
5	1	0	0	0
6	0	1	0	0
7	0	0	1	0
8	0	0	0	1



Comanda digitală a motorului bipolar se face similar cu cea unipolară; diferenta apare la partea de putere unde bobinele motorului sunt alimentate consecutiv şi alternativ cu ajutorul unor punţi H.



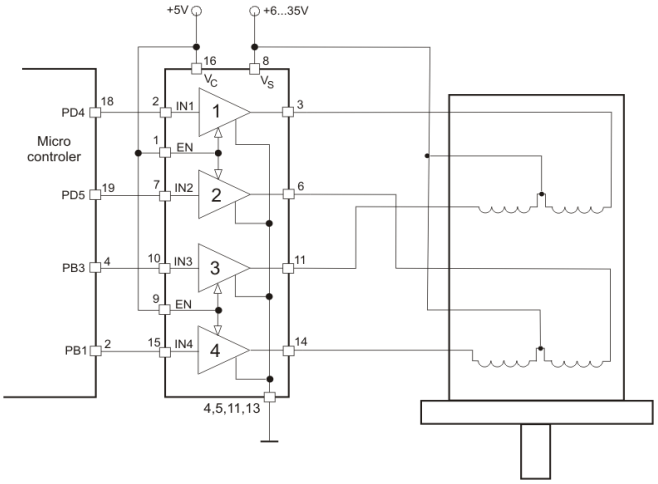
Pas	1a	1b	2a	2b
1	+	-	-	-
2	-	+	-	-
3	-	-	+	-
4	-	-	-	+
5	+	-	-	-
6	-	+	-	-
7	-	-	+	-
8	-	-	-	+



Aplicaţia 6 – Comanda motorului pas cu pas cu turaţie variabilă

Scopul aplicaţiei este comanda unui motor pas cu pas unipolar cu o turaţie variabilă, valoarea acesteia fiind stabilită cu ajutorul unui potenţiometru.

La apăsarea butonului conectat la pinul 6 al portului D sensul de rotaţie al motorului se schimbă.Comada se face in secvenţă simplă de paşire (wave drive)



	7	6	5	4	3	2	1	0	
PortB	0	0	0	0	0	0	1	0	=$02
PortD	0	0	0	1	0	0	0	0	=$10
PortB	0	0	0	0	1	0	0	0	=$08
PortD	0	0	1	0	0	0	0	0	=$20

Pas	PB1	PD4	PB3	PD5
1	+	-	-	-
2	-	+	-	-
3	-	-	+	-
4	-	-	-	+

```
program MPP_potentiometru_schimbare_sens;
const i : integer = 20;
      pas:array [0..3]of byte = ($02,$10,$08,$20);
var   t: integer;

procedure delay(var x : integer);
var i: integer;
begin
for i:=0 to x do delay_ms(1);
end;
//******* Program principal*********
begin
 ddrb:=$0a;
 ddrd:=$30;
   while true do begin
    t := Adc_Read(1);
    t:=t/32;
      if t<2 then t:=2;
      if t<127 then begin
      if  pind.6 = 1 then begin
       portb:=pas[0]; delay(t);
       portd:=pas[1]; delay(t);
       portb:=pas[2]; delay(t);
       portd:=pas[3]; delay(t);
      end;
      if pind.6 = 0 then begin
       portd:=pas[3]; delay(t);
       portb:=pas[2]; delay(t);
       portd:=pas[1]; delay(t);
       portb:=pas[0]; delay(t);
      end;
    end;
  end;
end.
```

Aplicaţia 7 – Comanda motorului pas cu pas cu turaţie variabilă (II)



Scopul aplicaţiei este comanda unui motor pas cu pas unipolar cu o turaţie variabilă, valoarea acesteia fiind stabilită cu ajutorul unui potenţiometru şi afişarea pe LCD a numărului de paşi efectuat.

La apăsarea butonului conectat la pinul 6 al portului D sensul de rotaţie al motorului se schimbă.

Comada se face in secvenţă simplă de paşire (wave drive) pentru fiecare pas al motorului se va aprinde câte un LED.

	7	6	5	4	3	2	1	0	
PortB	0	0	0	0	0	0	1	0	=$02
PortD	0	0	0	1	0	0	0	0	=$10
PortB	0	0	0	0	1	0	0	0	=$08
PortD	0	0	1	0	0	0	0	0	=$20

Pas	PB1	PD4	PB3	PD5
1	+	-	-	-
2	-	+	-	-
3	-	-	+	-
4	-	-	-	+

```
program MPP_potentiometru;
uses flexunit;
const i : integer = 20;
      pas:array [0..3]of byte = ($02,$10,$08,$20);
var   t,nrpasi:byte;
      s:string[5];
procedure delay(var x : integer);
var i: integer;
begin
for i:=0 to x do delay_ms(1);
end;
begin
 ddrb:=$0a;
 ddrd:=$30;
 ddrc:=$FF;
 Lcd_Init1;
 nrpasi:=0;
   while true do
   begin
    t:=Adc_Read(1);
    t:=t/16;
    lcd_text(1,1,'Motor PAS cu PAS');
    lcd_byte(2,1,nrpasi);
      if t<5 then t:=5;
      if t<63 then begin
      if  pind.6 = 1 then begin
        portb:=pas[0]; portc:=$01; delay(t);
        portd:=pas[1]; portc:=$02; delay(t);
        portb:=pas[2]; portc:=$04; delay(t);
        portd:=pas[3]; portc:=$08; delay(t);
        inc(nrpasi);
      end;
      if pind.6 = 0 then begin
        portd:=pas[3]; portc:=$08; delay(t);
        portb:=pas[2]; portc:=$04; delay(t);
        portd:=pas[1]; portc:=$02; delay(t);
        portb:=pas[0]; portc:=$01;delay(t);
        nrpasi:=nrpasi-1;
      end;
    end;
  end;
end.
```

Servomecanismul

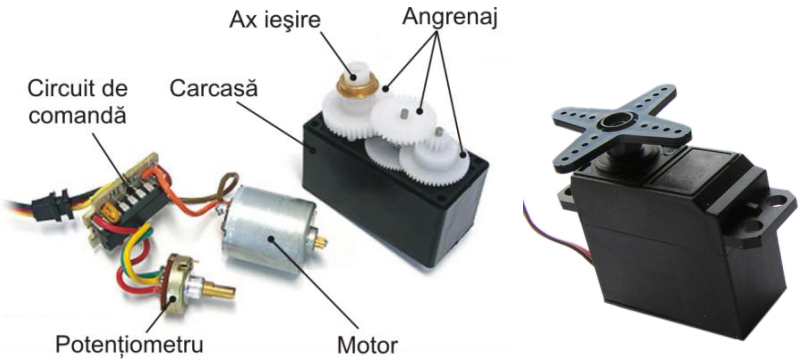

Servomecanismul

❑ Un servomecanism este compus dintr-un motor de curent continuu care prin intermediul unui angrenaj de roţi dinţate antrenează un ax de ieşire. Pe axul de ieşire este montat un potenţiometru cu rol de senzor de poziţie.

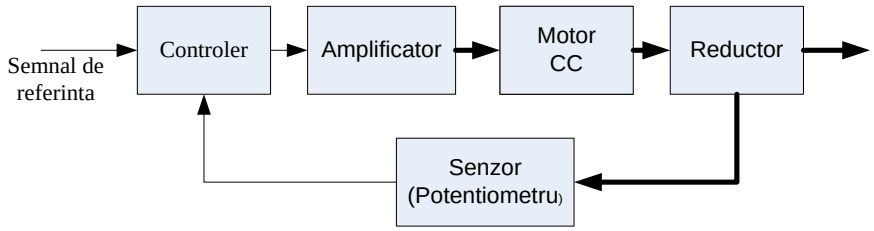
❑ Motorul servomecanismului nu este alimentat continuu, ci este controlat printr-o serie de impulsuri de tip ON/OFF. Durata unui ciclu este de 20 ms, din care valoarea standard pentru obţinerea poziţiei de centru (90°) a servomecanismului, este de 1,5 ms în valoarea 1 iar 18,5 ms în valoarea 0. Poziţia de minim sau de maxim se obţine la valori diferite pentru fiecare tip de servomecanism în parte.

FlexFORM - Program de formare profesională flexibilă pe platforme mecatronice

Un servomecanism este compus dintr-un motor de curent continuu care prin intermediul unui angrenaj de roţi dinţate antrenează un ax de ieşire. Pe axul de ieşire este montat un potenţiometru cu rol de senzor de poziţie. Arborele de ieşire al servomotoarelor nu face o rotaţie completă, ci doar 180 de grade



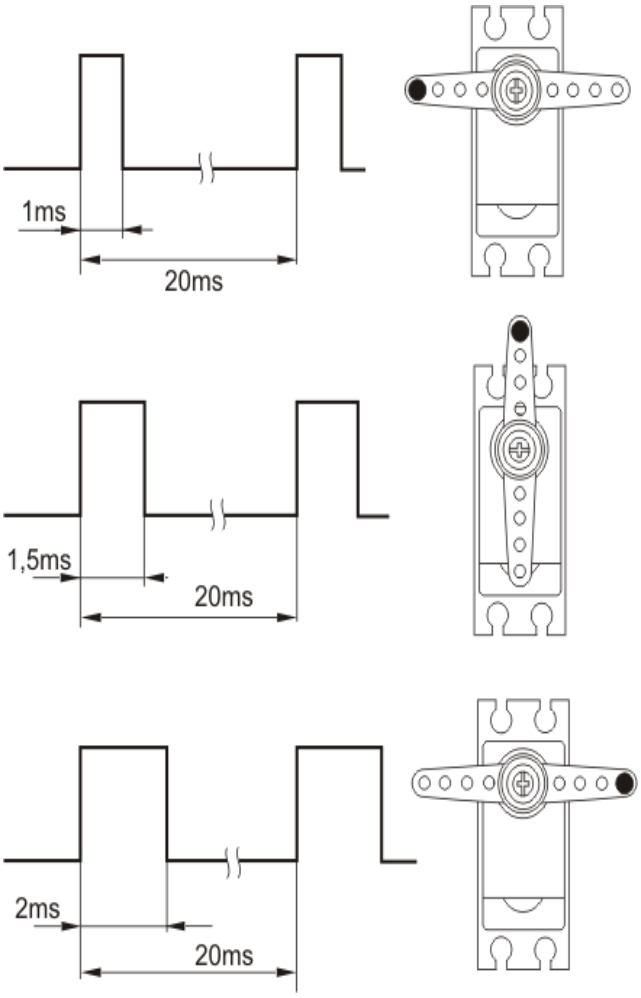
Circuitul de control are rolul de a modifica parametrii funcţionali (poziţie, viteză etc.) ai servomotorului în funcţie de semnalul de intrare (referință). Determinarea poziției la care se află arborele servomotorului se face prin intermediul unui senzor de tip rezistiv (potențiometru). Modificarea unghiului de rotație al arborelui va determina modificarea poziției cursorului potențiometrului şi implicit tensiunea de ieșire a acestuia. Valoarea tensiunii este utilizată de sistemul de control pentru a determina poziția/turația reală a servomecanismului, și comparând-o cu referința va rezulta semnalul de comandă pentru motorul de curent continuu.



Motorul servomecanismului nu este alimentat continuu, ci este controlat printr-o serie de impulsuri de tip ON/OFF. Durata unui ciclu este de 20 ms, din care valoarea standard pentru obţinerea poziţiei de centru (90°) a servomecanismului, este de 1,5 ms în valoarea 1 şi 18,5 ms în valoarea 0.

In acest interval circuitul de control compară, poziţia unghiulară a axului servomotorului cu unghiul dorit, unghi dat de factorul de umplere al semnalului PWM. În funcţie de valoarea obţinută, circuitul de control va alimenta motorul de curent continuu cu o tensiune care să determine rotirea axului servomotorului în vederea anulării diferenţei dintre cele două valori. Reductorul montat pe axul servomotorului amplifică cuplul dezvoltat de acesta.

Poziţia de minim sau de maxim se obţine la valori diferite pentru fiecare tip de servomecanism în parte. Cele mai uzuale valori sunt 1 ms pentru 180° şi 2 ms pentru 0°.




Servomecanismul

Aplicaţia 8 – Comanda servomecanismului



Scopul aplicaţiei este poziţionarea axului de ieşire al unui servomecanism in două poziţii distincte, cu ajutorul unui buton.

La inceputul programului iniţializează porturile A şi D ale microcontrolerului. Sevomecanismul este conectat la pinul 3 al portului A prin intermediul unui conector.

Dacă se apasă butonul conectat la pinul 6 al portului D atunci axul servomecanismului va fi poziţionat la mijlocul cursei, adică la 90 grade. Când se ajunge în poziţia dorită servomecanismul se opreşte.

Dacă se eliberează butonul conectat la pinul 6 al portului D atunci, axul servomecanismului va fi poziţionat la inceputul cursei adică la 0 grade.

```
program servo1;
const a = 20000;
begin
ddra:=$0F;
ddrd:=$00;
portd.6:=1;
  while true do begin
     if pind.6 = 0 then begin
        porta.3:=1;
        delay_us (1500);
        porta.3:=0;
        delay_us (a-1500);
      end
     else begin
        porta.3:=1;
        delay_us (2000);
        porta.3:=0;
        delay_us(a-2000);
      end;
  end;
end.
```

Aplicaţia 9 – Comanda poziţiei unui servomecanism



Scopul aplicaţiei este controlul poziţiei axului servomecanismului, cu ajutorul unui potenţiometru.

La inceputul programului iniţializează comunicaţia cu LCD-ul Pe primul rând al LCD-ului va fi afişat textul “Test servo” iar pe al doilea valoarea citită pe ADC

Axul servomecanismului va avea o deplasare unghiulară proporţională cu unghiul de rotaţie al potenţiometrului conectat la canalul 1 al ADC-ului. Sevomecanismul este conectat la pinul 3 al portului A

```
program servo_adc;
uses flexunit;
var a:word;

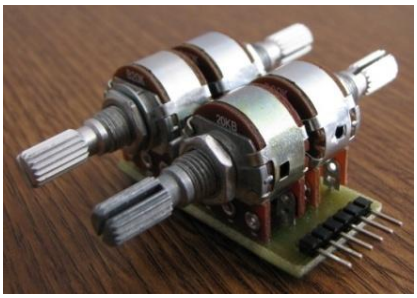
begin
  lcd_init;
  lcd_text(1,1, 'TEST SERVO');
  ddra.3:=1;
   while 1=1 do begin
   a:=Adc_Read(1);
   a:=a+1000;
   porta.3:=1;
   stai_us(a);
   porta.3:=0;
   stai_us(20000-a);
   lcd_word(2,1, a);
 end;
end.
```

Aplicaţia 10 – Comanda poziţiei pentru trei servomecanisme



Scopul aplicaţiei este controlul poziţiei a trei servomecanisme, cu ajutorul a trei potenţiometre.

Poziţia fiecărui servomecanism este controlată cu ajutorul a câte unui potenţiometru. Acestea sunt conectate la canalele 0, 5 şi 6 ale ADC+ului



Cele trei servomecanisme sunt conectate la pinii 1, 2 şi 3 ai portului A.

```
program servo3;
uses flexunit;
var i,p1,p2,p3,p4,p5:word;

begin
   ddra:=$0e;
   while true do
   begin
   //*******Servo1*********
   porta.1:=1;
   p1:=1000+adc_read(5);
   stai_us(p1);
   porta.1:=0;
   //*******Servo2*********
   porta.2:=1;
   p2:=1000+adc_read(6);
   stai_us(p2);
   porta.2:=0;
   //*******Servo3*********
   porta.3:=1;
   p3:=1000+adc_read(0);
   stai_us(p3);
   porta.3:=0;
   stai_us(20000-(p1+p2+p3));
  end;
end.
```

Aplicaţia 11 – Comanda unui braţ robotic cu trei grade de mobilitate


Aplicaţia 11 – Comanda unui braţ robotic cu trei grade de mobilitate

Introducere

❑ Braţul robotic cu trei grade de mobilitate este montat pe o platformă mobilă antrenată prin intermediul a două motoare de curent continuu.

❑ Principalele elemente componente sunt :

1. Platformă
2. Suport
3. Roţi
4. Motoare de curent continuu cu reductor
5,6,8. Servomecanisme
7,9. Braţele robotului

❑ Cele trei grade de mobilitate ale braţului robotic sunt asigurate de cele trei servomecanisme din structură.

❑ Pentru asamblarea braţului robotic si a platformei mobile se parcurg succesiv etapele prezentate in continuare:

FlexFORM - Program de formare profesionala flexibila pe platforme mecatronice

Introducere

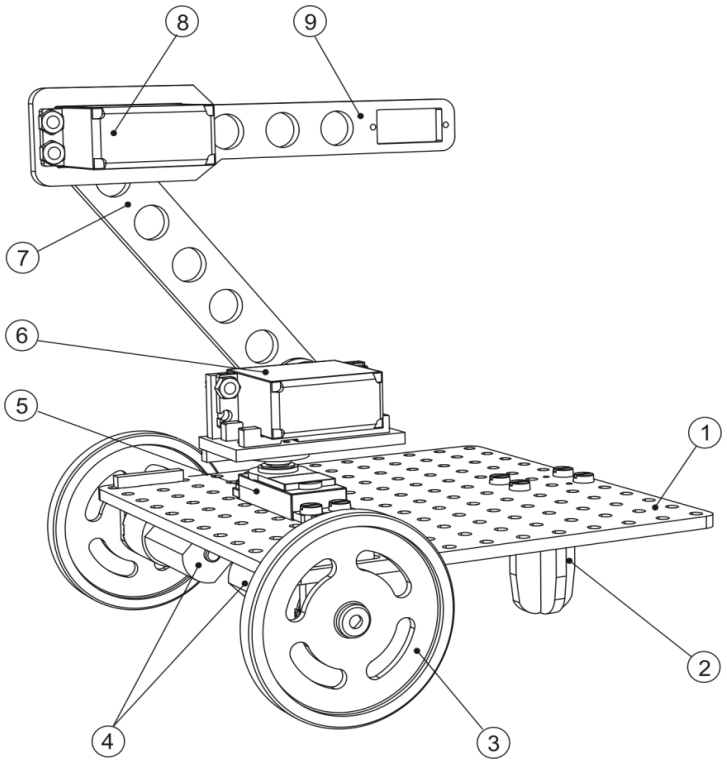
Braţul robotic cu trei grade de mobilitate este montat pe o platformă mobilă antrenată prin intermediul a două motoare de curent continuu.

Principalele elemente componente sunt :

1. Platformă
2. Suport
3. Roţi
4. Motoare de curent continuu cu reductor

5,6,8. Servomecanisme

7,9. Braţele robotului

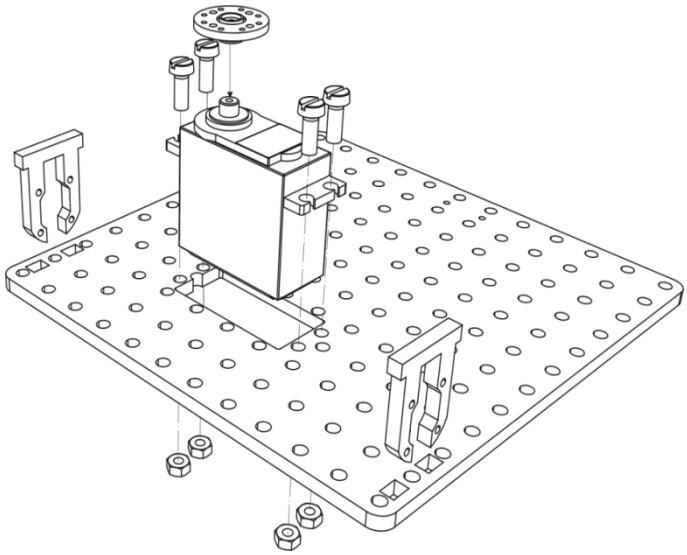


Cele trei grade de mobilitate ale braţului robotic sunt asigurate de cele trei servomecanisme din structură.

Pentru asamblarea braţului robotic si a platformei mobile se parcurg succesiv etapele prezentate in continuare:

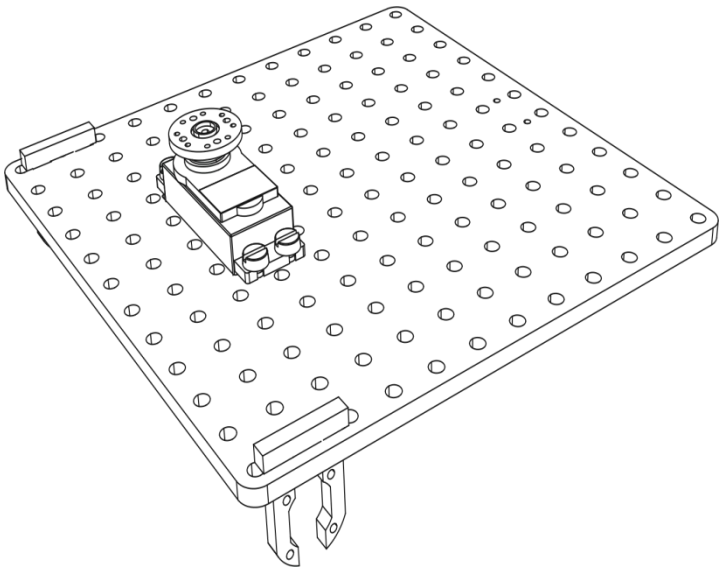
Asamblarea braţului robotic





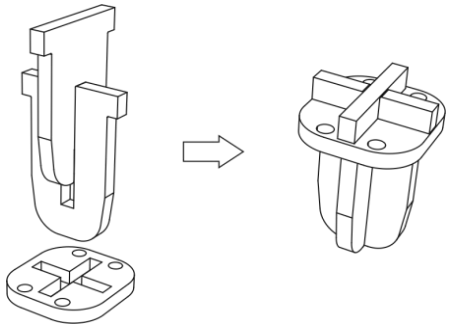
Se introduce servomotorul *5* in deschizătura prevazută în platforma *1* şi se fixează cu ajutorul a patru suruburi M4 cu piuliţă. Se montează rozeta pe axul servomecanismului.

La extremităţile platformei sunt prevăzute patru orificii dreptunghiulare pentru fixarea suporturilor celor două motoare de curent continuu. Aceste motoare servesc la antrenarea platformei mobile.

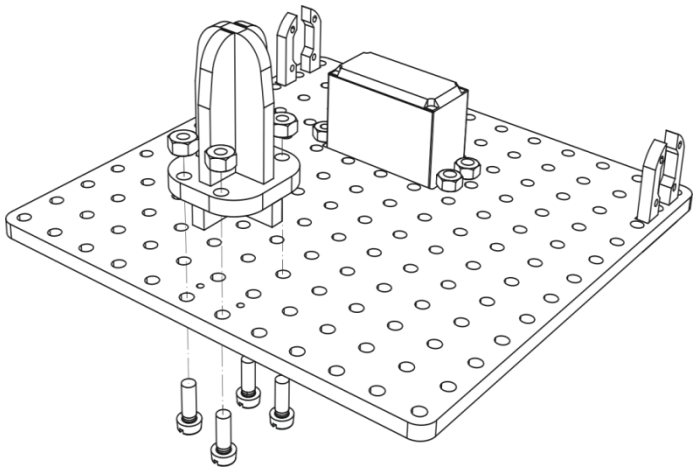


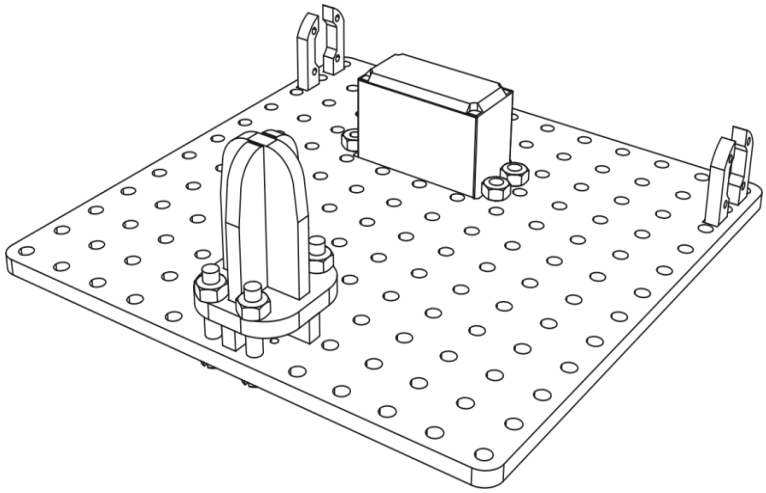


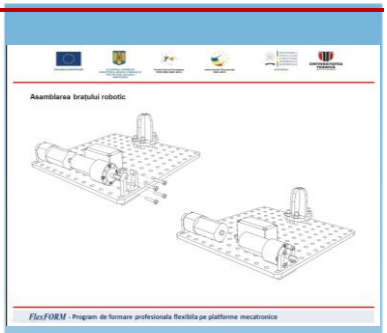
Se asamblează cele trei elemente componente ale suportului platformei mobile

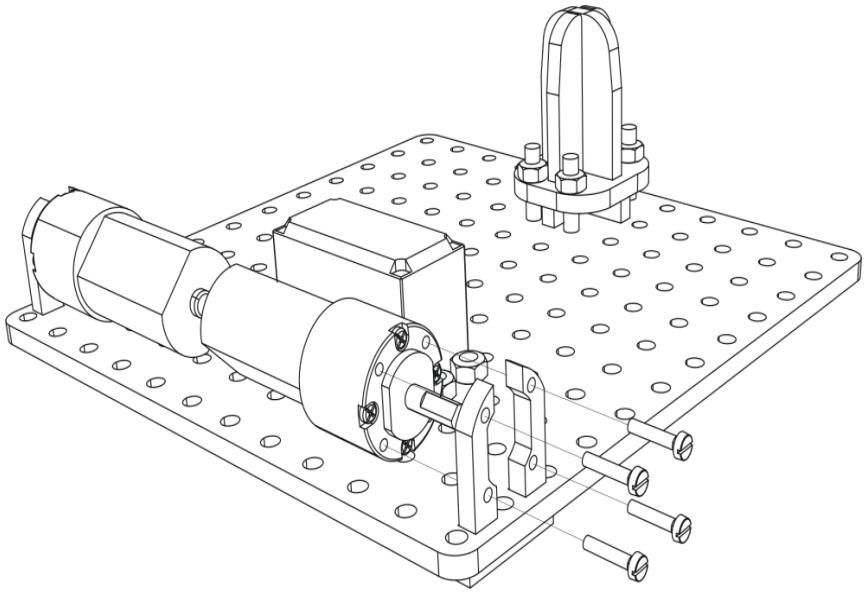


Se intoarce platforma *1*, având deja montate servomecanismul *5* şi cele două suporturi pentru motoare, şi se fixează cu ajutorul a patru şuruburi M4, suportul *2*.

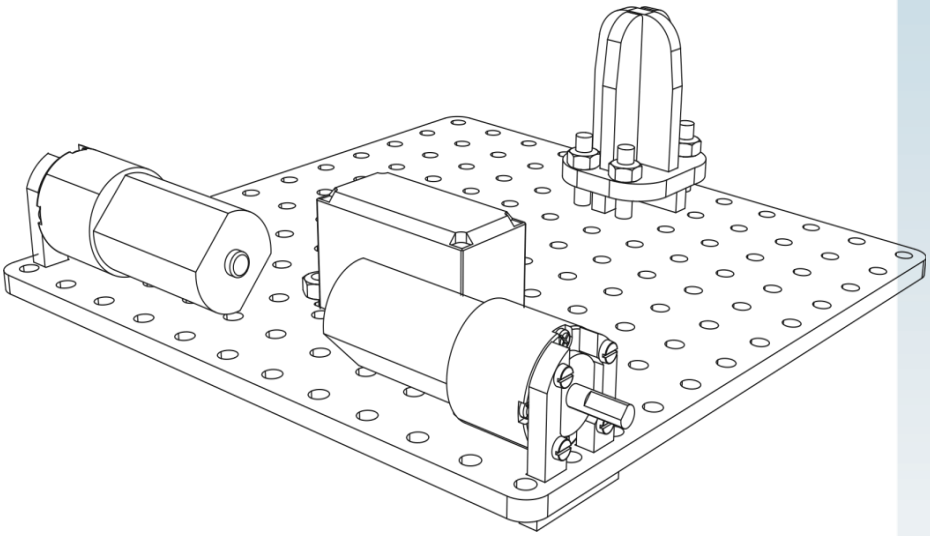






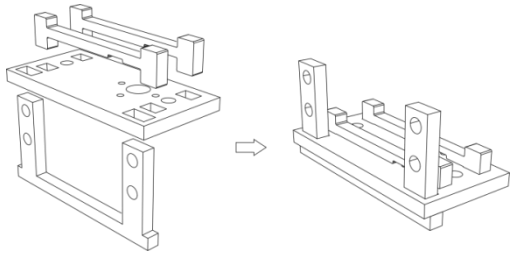


Se montează cele două motoare de curent continuu cu reductor *4* folosite pentru antrenarea platformei mobile.

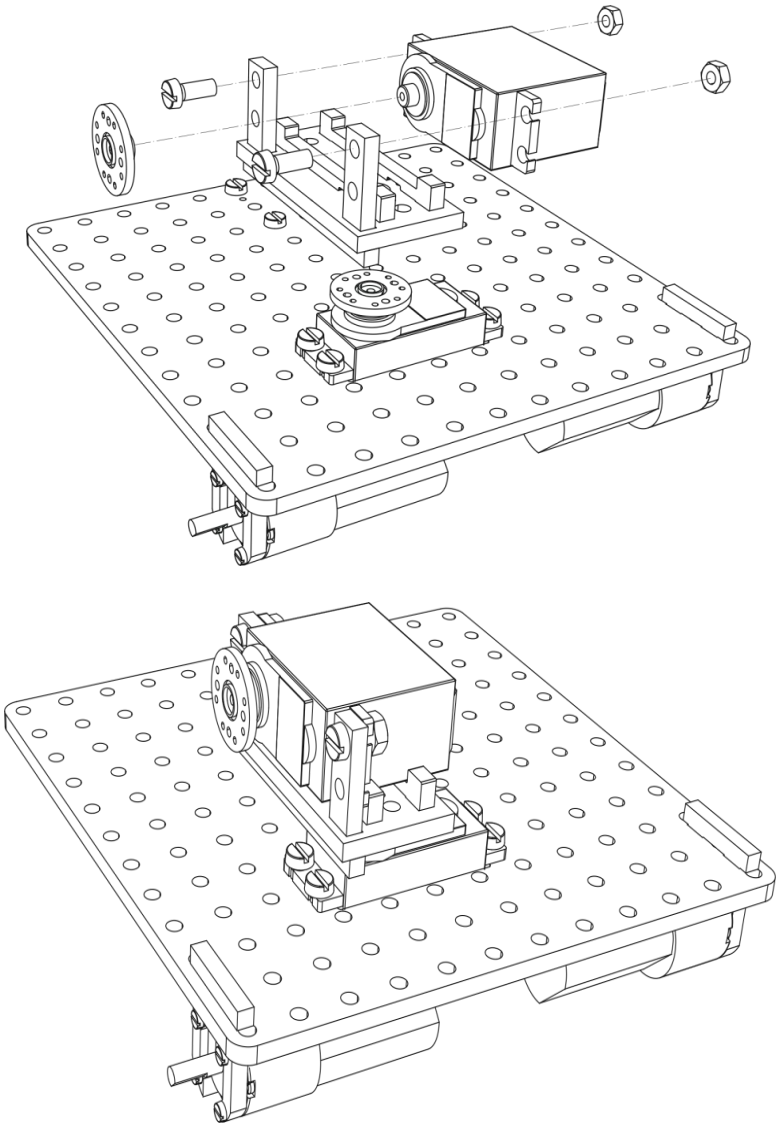




Se asamblează cele patru elemente componente ale suportului pentru servomecanismul *6.* Acest suport permite conectarea braţului mobil la servomecanismul fixat pe platforma mobilă

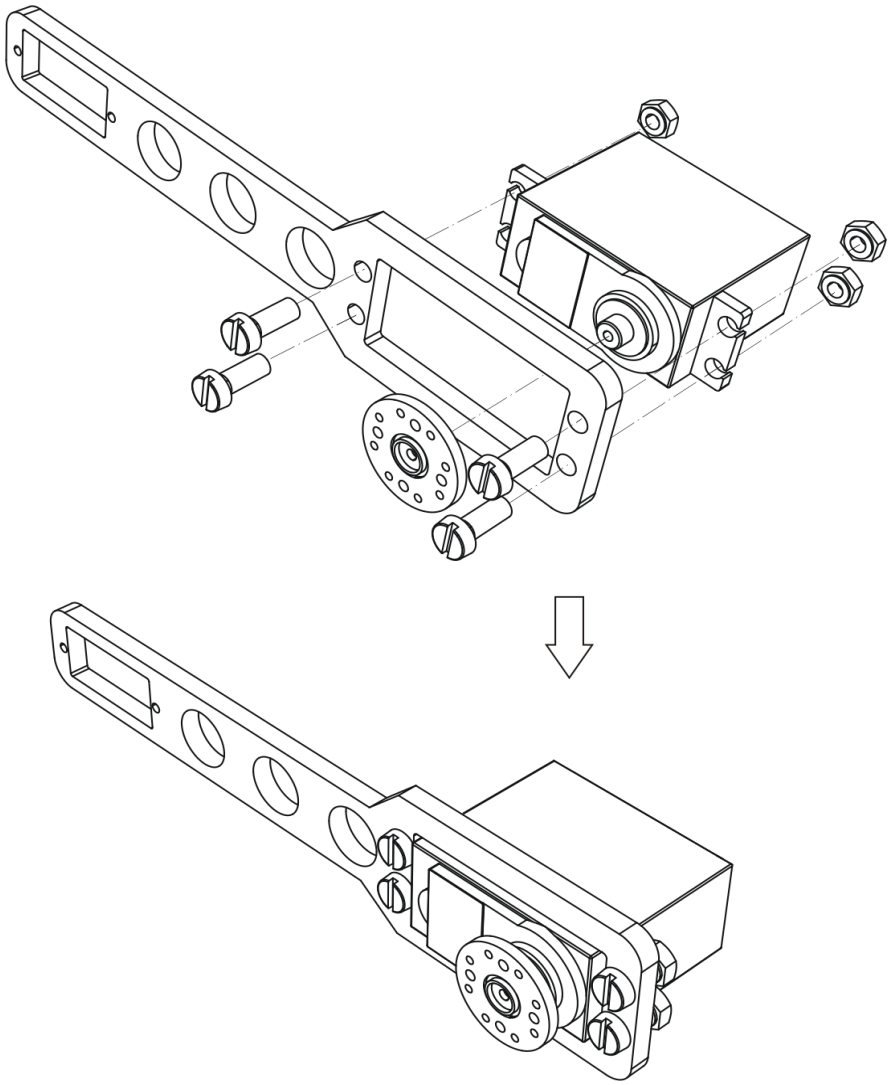


Se fixează servomecanismul *6* de suport du ajutorul a două suruburi M4 cu piuliţă. Se montează rozeta pe axul servomecanismului.

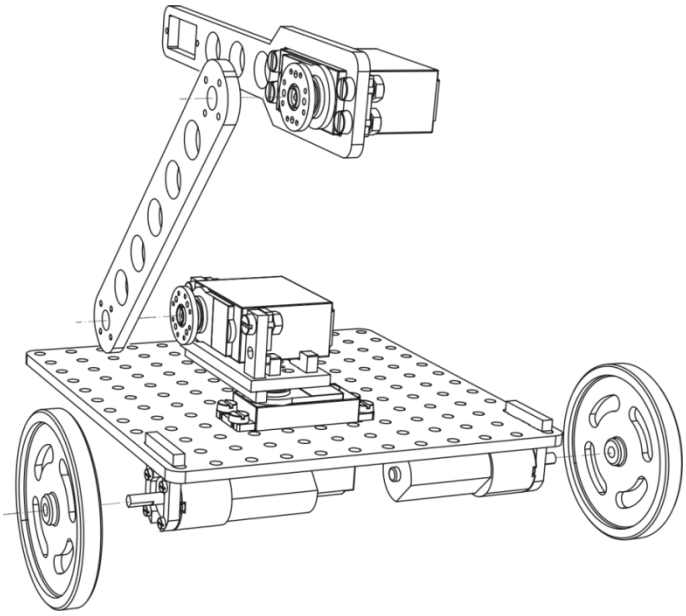


Se fixează, cu ajutorul şuruburilor, braţul *9* de servomecanismul *8*.

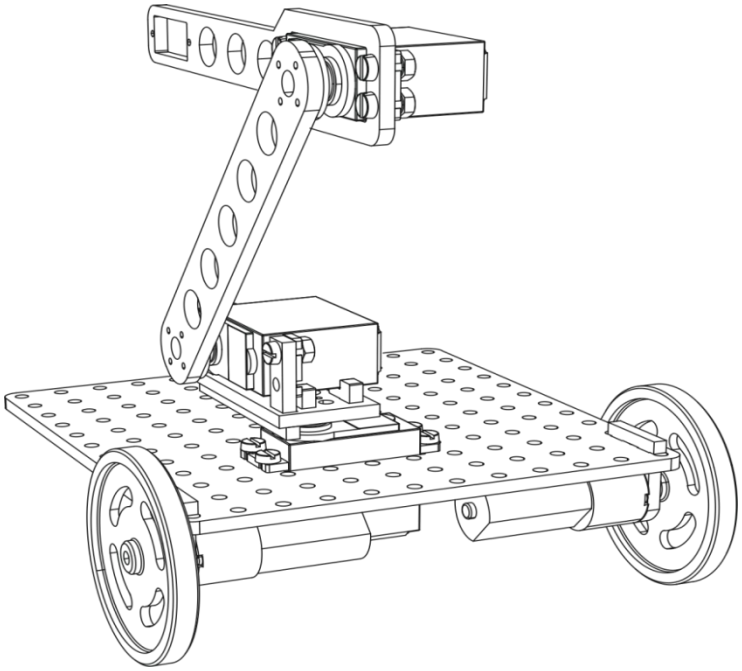








Se conectează braţul 9 şi servomecanismul 8 de restul platformei cu ajutorul elementului 7. Se montează roţile 3 pe cele două motoare de antrenare



Programul de comandă

Programul pentru comanda celor trei servomecanisme poate fi cel scris in cadrul Aplicaţiei 10 sau unul asemănător cu acesta. La scrierea programului se ţine cont de faptul că sevomecanismele sunt conectate la pinii 1, 2 şi 3 ai portului A, iar cei trei potenţiometrii la canalele 0, 5 şi 6 ale ADC-ului


Programul de comandă

❑ Programul pentru comanda celor trei servomecanisme poate fi cel scris in cadrul Aplicaţiei 10

❑ Sevomecanismele pot fi conectate la pinii 1, 2 şi 3 ai portului A, iar cei trei potenţiometrii la canalele 0, 5 şi 6 ale ADC-ului

FlexFORM - Program de formare profesionala flexibila pe platforme mecatronice

```
program servo3;
uses flexunit;
var i,p1,p2,p3,p4,p5:word;

begin
   ddra:=$0e;
   while true do
   begin
   //*******Servo1*********
   porta.1:=1;
   p1:=1000+adc_read(5);
   stai_us(p1);
   porta.1:=0;
   //*******Servo2*********
   porta.2:=1;
   p2:=1000+adc_read(6);
   stai_us(p2);
   porta.2:=0;
   //*******Servo3*********
   porta.3:=1;
   p3:=1000+adc_read(0);
   stai_us(p3);
   porta.3:=0;
   stai_us(20000-(p1+p2+p3));
  end;
end.
```

```
uses flexunit;
var i,j:byte;
    timp,a:word;

begin
  ddra:=$0f;
  while 1=1 do begin
    timp:=20000;
    for i:=0 to 3 do begin
      a:=adc_read(i+4);
      porta.i:=1;
      a:=500+2*a;
      timp:=timp-a;
      stai_us(a);
      porta.i:=0;
    end;
    stai_us(timp);
  end;
end.
```

ALGORITMI DE CONTROL

OBIECTIVE

- Cunoaşterea principalilor algoritmi de control.
- Prezentarea controlerului ON/OFF şi a controlerului proporțional.
- Prezentarea algoritmului PID. Acordarea controlerului.
- Controlul unei platforme mobile acționată cu motoare de curent continuu.

Generalităţi


Generalităţi

❑ **Sistemul** reprezintă o mulţime de elemente aflate în interacţiune, care se comportă unitar în relaţiile sale cu mediul, adică înseamnă o unitate complexă. Orice obiect sau fenomen poate fi considerat ca un sistem, cu condiţia de a-l putea separa de celelalte obiecte şi fenomene şi de a-l defini riguros şi univoc

❑ **Procesul** reprezintă o transformare a unui sistem, indicată prin modificarea unor mărimi de proces (ex: modificarea temperaturii, a presiunii, a poziţiei, etc.) Parametrii unui proces sunt reprezentaţi de mărimile fizice care caracterizează procesul. Pot fi :

- **Parametri de intrare** - mărimi fizice măsurabile, exterioare procesului, care influenţează comportarea acestuia;
- **Parametri de stare** - înglobează informaţia referitoare la evoluţia anterioară a procesului ;
- **Parametri de ieşire** - mărimi a căror evoluţie dorim să o controlăm.

❑ **Funcţia de transfer** a unui proces este expresia matematică a dependenţei dintre parametrii de ieşire, parametrii de stare şi parametrii de intrare ai unui proces.

FlexFORM - Program de formare profesionala flexibila pe platforme mecatronice

Sistemul reprezintă o mulţime de elemente aflate în interacţiune, care se comportă unitar în relaţiile sale cu mediul, adică înseamnă o unitate complexă. Orice obiect sau fenomen poate fi considerat ca un sistem, cu condiţia de a-l putea separa de celelalte obiecte şi fenomene şi de a-l defini riguros şi univoc. (Ex: motorul electric, schimbatorul de caldură etc)

Procesul reprezintă o transformare a unui sistem, indicată prin modificarea unor mărimi de proces. De exemplu deplasarea mecanică a unor obiecte, transformarile fizico-chimice ale unor fluide, modificările de formă ale unor componente, variaţiile unor mărimi fizice precum temperatură, nivel, concentraţie, etc. sunt exemple de procese. Parametrii unui proces sunt reprezentaţi de mărimile fizice care caracterizează procesul. Pot fi :

- **Parametri de intrare** - mărimi fizice măsurabile, exterioare procesului, care influenţează comportarea acestuia;
- **Parametri de stare** - înglobează informaţia referitoare la evoluţia anterioară a procesului ;
- **Parametri de ieşire** - mărimi a căror evoluţie dorim să o controlăm.

Studiul analitic al proceselor reprezintă stabilirea structurii acestora şi a interdependenţei dintre mărimile de stare pe baza unor legi cunoscute (modelare)

Studiul empiric al unui proces se realizează prin stabilirea structurii procesului şi a interdependenţelor dintre mărimile sale de stare pe baza experienţei obţinute prin observaţii (identificare)

Modelul matematic al procesului este o reprezentare a sistemului real prin relaţii matematice, ce înglobează anumite proprietăţi ale sistemului considerat. Fiecare model reflectă doar anumite proprietăţi ale "originalului", celelalte fiind neglijate datorită faptului că ele nu pot fi descrise sau nu interesează într-un anume caz concret

Funcţia de transfer a unui proces este expresia matematică a dependenţei dintre parametrii de ieşire, parametrii de stare şi parametrii de intrare ai unui proces.

Inevitabil prin modelarea matematică procesului se omit o serie de detalii considerate mai puţin importante pentru descrierea şi mai ales pentru evoluţia acestuia. Simplificările sunt necesare pentru reducerea complexităţii sistemului. In mod frecvent se omit condiţiile de mediu care influenţează un anumit proces (frecare, temperatură, umiditate, vibraţii, etc.), datorită dificultătii de cuantizare a acestora. Influenţa cumulativă a factorilor de mediu se modelează prin conceptul de "*zgomot*"


Generalităţi

❑ **Perturbaţiile sau zgomotele** sunt mărimile fizice care influenţează evoluţia unui proces, dar a căror mărime şi evoluţie în timp este necunoscută şi/sau necontrolată. Exemple: zgomote electromagnetice, variaţii ale parametrilor de mediu (temperatură, presiune, umiditate, etc.), variaţii ale tensiunii de alimentare, forţe şi/sau momente exterioare etc.

❑ **Semnalul** reprezintă o mărime purtătoare a unei informaţii. Semnalele asigură legătura între sistemul controlat şi cel de control.

Pot fi:

analogice – care au un domeniu continuu de valori (ex. variaţia tensiunii)

digitale – care au un număr finit de valori discrete; de cele mai multe ori se folosesc semnale care au două valori distincte (codificate cu 0 şi 1) şi care reprezintă două stări diferite ale unui element de proces (ex.: închis/deschis, valid/invalid, pornit/oprit, etc.)

continue – au variaţie continuă în timp

discrete (eşantionate) – au variaţie discretă în timp, adică funcţiile ce le reprezintă au valori definite doar la anumite momente de timp

❑ **Controlul** este un proces complex care presupune monitorizarea parametrilor unui sistem şi intervenţia pentru reglarea şi menţinerea acestora la valori impuse de buna funcţionare a sistemului.

FlexFORM - Program de formare profesională flexibilă pe platforme mecatronice

Perturbaţiile sau zgomotele sunt mărimile fizice care influenţează evoluţia unui proces, dar a căror mărime şi evoluţie în timp este necunoscută şi/sau necontrolată

Informaţia cu privire la parametrii de proces se transmite prin intermediul unor **semnale.** Un semnal este o marime fizică ce este capabilă să transmită o informaţie. Semnalele asigură legătura între sistemul controlat şi cel de control. Pot fi:

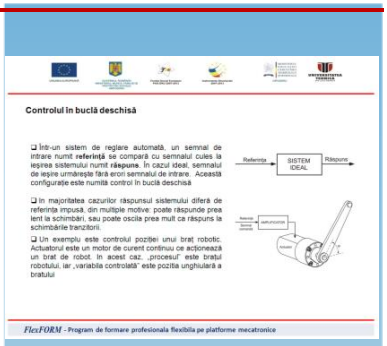
analogice – care au un domeniu continuu de valori (ex. variaţia tensiunii)

digitale – care au un număr finit de valori discrete; de cele mai multe ori se folosesc semnale care au două valori distincte (codificate cu 0 şi 1) şi care reprezintă două stări diferite ale unui element de proces (ex.: închis/deschis, valid/invalid, pornit/oprit, etc.)

Caracterul continuu sau discret al semnalelor se referă atât la evoluţia în timp a semnalelor (semnale continue sau eşantionate) cât şi la modul de variaţie a acestora (semnale analogice sau digitale).

Controlul este un proces complex care presupune monitorizarea parametrilor unui sistem şi intervenţia pentru reglarea şi menţinerea acestora la valori impuse de buna funcţionare a sistemului. De exemplu menţinerea temperaturii unei încăperi la o valoare stabilită sau deplasarea unui braţ robotic într-un interval prestabilit reprezintă posibile obiective ale unui sistem de control. Dacă controlul se realizează prin tehnologii digitale, în particular prin folosirea unui microsistem de calcul, atunci vorbim despre un **sistem digital de control**.

Controlul în buclă deschisă

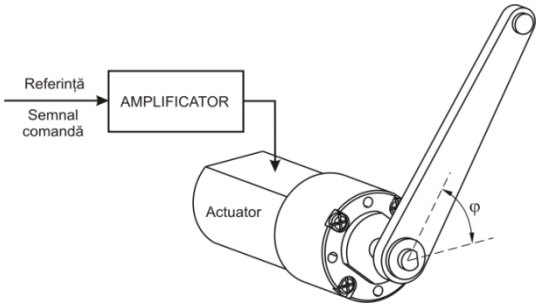


Într-un sistem de reglare automată, un semnal de intrare numit **referinţă** se compară cu semnalul cules la ieşirea sistemului numit **răspuns**. În cazul ideal, semnalul de ieşire urmăreşte fără erori semnalul de intrare. Această configuraţie este numită control în buclă deschisă.



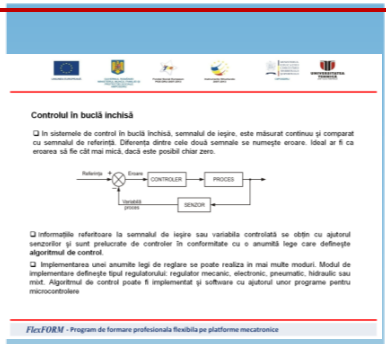
Intr-un sistem în buclă deschisă, controlerul calculează independent mărimea semnalului de comandă necesar actuatorului pentru a indeplini sarcina propusa. In acest mod, controlerul nu primeste nici o informatie din partea actuatorului. În majoritatea cazurilor răspunsul sistemului diferă de referinţa impusă, din multiple motive: poate răspunde prea lent la schimbări, sau poate oscila prea mult ca răspuns la schimbările tranzitorii.

Un exemplu îl poate constitui controlul poziţiei unui braţ robotic acţionat prin intermediul unui actuator (motor de curent continuu). In acest caz, braţul robotului reprezintă „procesul” iar pozitia unghiulară a bratului este „variabila controlată”.

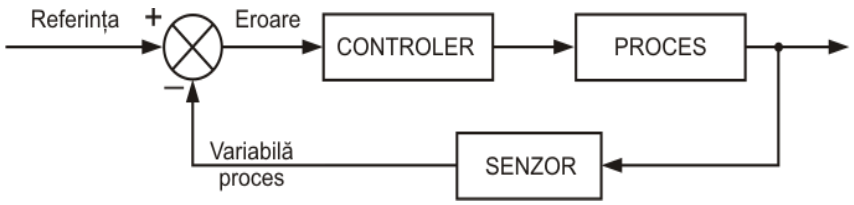


Braţul de robot trebuie rotit cu 40 de grade in sens trigonometric, dintr-o poziţie considerată iniţială. Dacă viteza unghiulară a motorului este constantă şi are valoarea de 10 grade pe secundă înseamnă că durata semnalului de comandă pentru actuator va fi de 4 secunde. Datorită perturbaţiilor (ex: forta/moment rezistent) bratul de robot se deplasează doar 35 de grade. Apare astfel o eroare de 5 grade. În lipsa unui semnal referitor la poziţia curentă a braţului robotic, controlerul nu va actiona in vreun fel pentru reducerea acestei erori.

Controlul în buclă închisă



În sistemele de control în buclă închisă, semnalul de ieşire, este măsurat continuu şi comparat cu semnalul de referinţă. Diferenţa dintre cele două semnale se numeşte eroare. Ideal ar fi ca eroarea să fie cât mai mică, dacă este posibil chiar zero.



Informaţiile referitoare la semnalul de ieşire sau variabila controlată se obţin cu ajutorul senzorilor şi sunt prelucrate de controler în conformitate cu o anumită lege care defineşte **algoritmul de control**.

Implementarea unei anumite legi de reglare se poate realiza in mai multe moduri. Modul de implementare defineşte tipul regulatorului. Acestea pot fi:

- **mecanice, hidraulice, pneumatice** – se folosesc componente mecanice, hidraulice sau pneumatice mai mult sau mai puţin standardizate; este dificil de implementat o funcţie de reglaj, optimă din punct de vedere teoretic.
- **regulatoare electronice sau analogice** – se folosesc componente electronice active (tranzistor, amplificator operaţional) şi pasive (rezistenţă, condensator, bobină); precizia de implementare a funcţiei de reglaj depinde de precizia componentelor
- **regulatoare digitale sau numerice** – utilizează componente digitale (porţi logice, bistabile, etc.), inclusiv microprocesoare; funcţia de reglaj se implementează printr-o schemă logică sau prin program; în ultimul caz pot fi implementate funcţii complexe de reglaj, iar precizia de reglaj nu depinde de precizia componentelor

Clasificarea regulatoarelor (controlerelor)



După natura semnalului de comandă generat:

- **regulatoare continue** – semnalul de comandă este o funcţie continuă în raport cu abaterea şi cu timpul (ex.: regulatoare P, PI, PID, etc.)
- **regulatoare discontinue** – semnalul de comandă este o funcţie care are discontinuităţi în raport cu abaterea (ex.: regulatoare bipoziţionale, tripoziţionale şi în mai multe trepte)
- **discrete** – semnalul de comandă este o funcţie discretă în timp, adică se generează impulsuri modulate în amplitudine, frecvenţă, factor de umplere sau se generează informaţii codificate binar

După mărimea constantelor de timp implicate

- **regulatoare pentru procese lente** – constantele de timp ale procesului sunt foarte mari (peste zeci de secunde); exemple: reglare temperatură, nivel, concentraţii de gaze
- **regulatoare pentru procese rapide** – constantele de timp sunt relativ mici (sub câteva secunde); exemplu: reglarea turaţiei motoarelor, poziţionare, reglare presiune

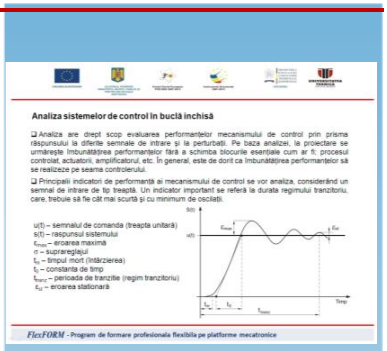
După legea de reglare

- **regulatoare de prag** – bipoziţionale sau tripoziţionale
- **regulatoare continue,** proporţionale (P), proporţional-derivative (PD), proporţional-integrative (PI) şi proporţional-integrativ-derivative (PID)

După gradul de adaptabilitate

- **regulatoare clasice** (neadaptive) – coeficienţii de reglaj se acordează manual
- **regulatoare autoadaptive** – coeficienţii de reglaj se acordează automat

Analiza sistemelor de control în buclă inchisă


Analiza sistemelor de control în buclă închisă

❑ Analiza are drept scop evaluarea performanţelor mecanismului de control prin prisma răspunsului la diferite semnale de intrare şi la perturbaţii. Pe baza analizei, la proiectare se urmăreşte îmbunătăţirea performanţelor fără a schimba blocurile esenţiale cum ar fi: procesul controlat, actuatorii, amplificatorul, etc. În general, este de dorit ca îmbunătăţirea performanţelor să se realizeze pe seama controlerului.

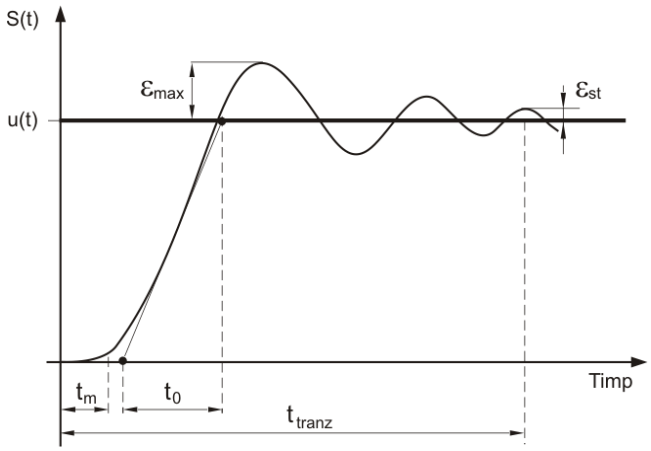
❑ Principalii indicatori de performanţă ai mecanismului de control se vor analiza, considerând un semnal de intrare de tip treaptă. Un indicator important se referă la durata regimului tranzitoriu, care, trebuie să fie cât mai scurtă şi cu minimum de oscilaţii.

u(t) – semnalul de comanda (treapta unitară)
s(t) – raspunsul sistemului
ε_{max} – eroarea maximă
σ – suprareglajul
t_m – timpul mort (întârzierea)
t_0 – constanta de timp
t_{tranz} – perioada de tranziţie (regim tranzitoriu)
ε_{st} – eroarea stationară

FlexFORM - Program de formare profesionala flexibila pe platforme mecatronice

Analiza are drept scop evaluarea performanţelor mecanismului de control prin prisma răspunsului la diferite semnale de intrare şi la perturbaţii. Pe baza analizei, la proiectare se urmăreşte îmbunătăţirea performanţelor fără a schimba blocurile esenţiale cum ar fi: procesul controlat, actuatorii, amplificatorul, etc. În general, este de dorit ca îmbunătăţirea performanţelor să se realizeze pe seama controlerului.

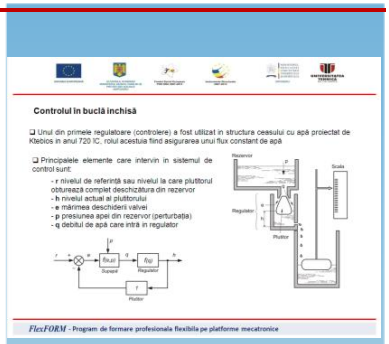
.



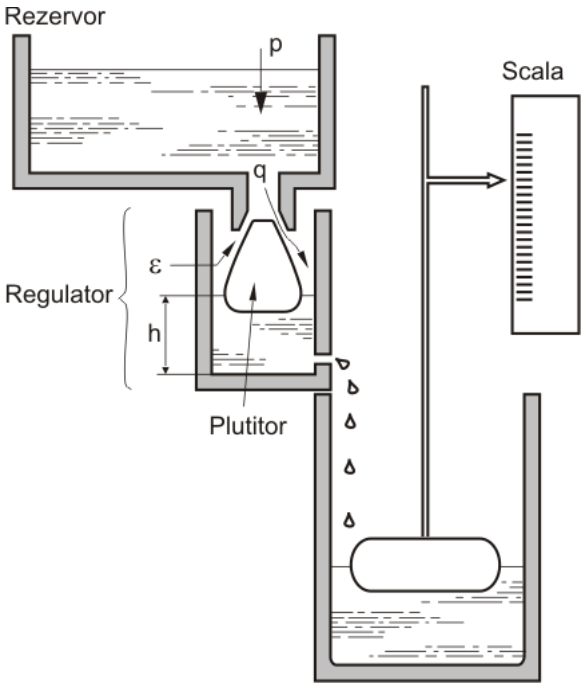
Parametrii definiţi în raport cu răspunsul tranzitoriu al sistemului atunci când la intrarea acestuia se aplică o mărime de intrare treaptă unitară, sunt:

- u(t) – semnalul de comandă (treapta unitară)
- s(t) – răspunsul sistemului
- ε_{max} – eroarea maximă
- σ – suprareglajul este raportul dintre eroarea maximă şi valoarea de regim staţionar ale mărimii de ieşire exprimată în procente
- t_m – timpul mort (întârzierea)
- t_0 – constanta de timp
- t_{tranz} – regimul tranzitoriu este intervalul de timp care începe odată cu semnalul de intrare şi durează până când răspunsul sistemului intră în intervalul de valori ale erorii staţionare
- ε_{st} – eroarea stationară care reprezintă diferenţa dintre valoarea referinţei şi valoarea răspunsului sistemului regim staţionar

Exemple de sisteme de control în buclă inchisă

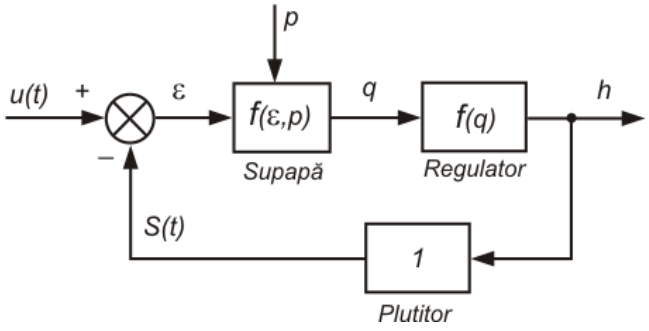


Unul din primele regulatoare (controlere) a fost utilizat in structura ceasului cu apă proiectat de Ktebios in anul 720 îHr, rolul acestuia fiind asigurarea unui flux constant de apă.

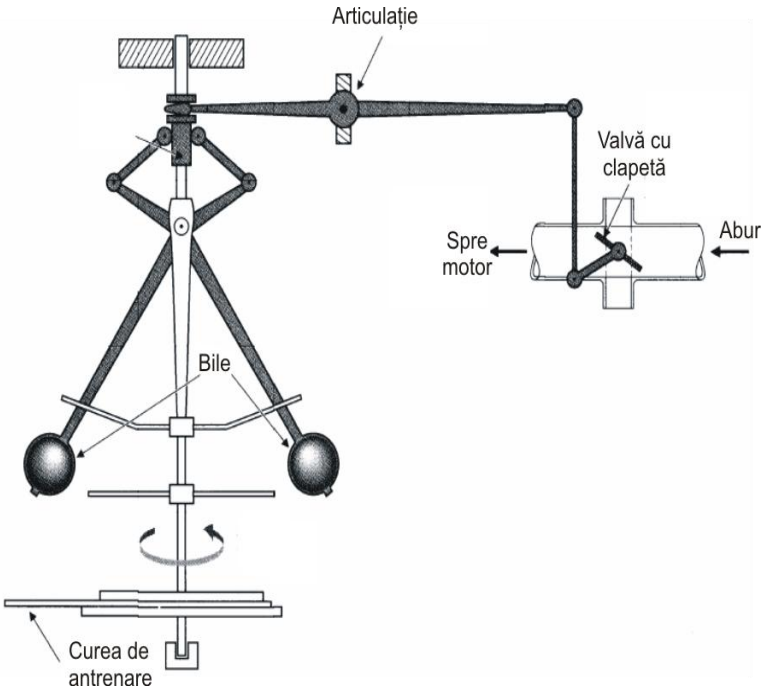


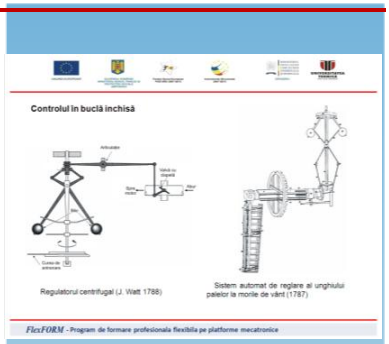
Principalele elemente care intervin in sistemul de control sunt:

- **u** nivelul de referinţă sau nivelul la care plutitorul obturează complet deschizătura din rezervor
- **h** nivelul actual al plutitorului
- **ε** mărimea deschiderii valvei
- **p** presiunea apei din rezervor (perturbaţia)
- **q** debitul de apă care intră in regulator

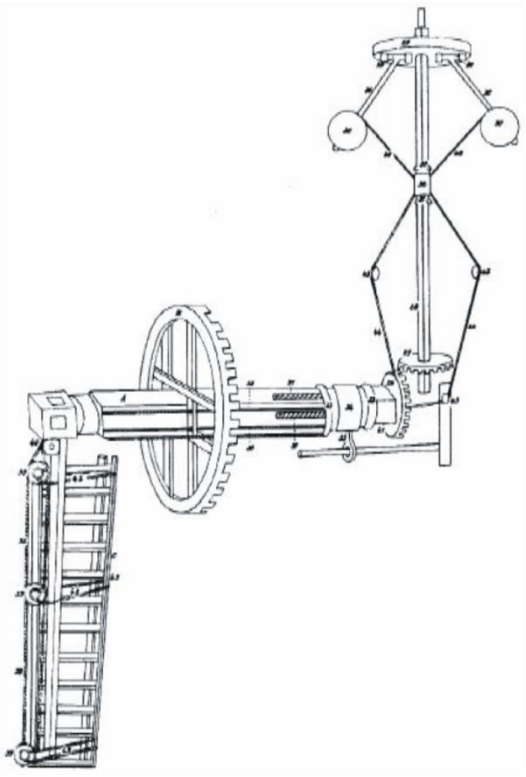


Regulatorul centrifugal (J. Watt 1788)



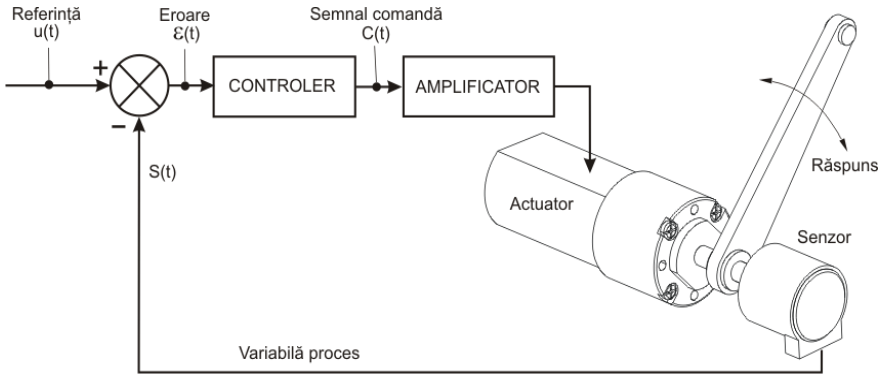


Sistem automat de reglare al unghiului palelor la morile de vânt (1787)





Considerăm un braţ de robot care trebuie să execute o mişcare de rotaţie în sens trigonometric în intervalul 0° - 30°. Valoarea unghiului la care trebuie poziţionat braţul robotic va reprezenta pentru controler, "*referinţa*". Aceasta mărime pentru a putea fi analizată de sistemul de control trebuie în prealabil convertită într-un semnal (tensiune, curent etc). Informaţia provenită de la senzorul de poziţie, montat pe axul motorului, reprezintă "*variabila*" controlată.



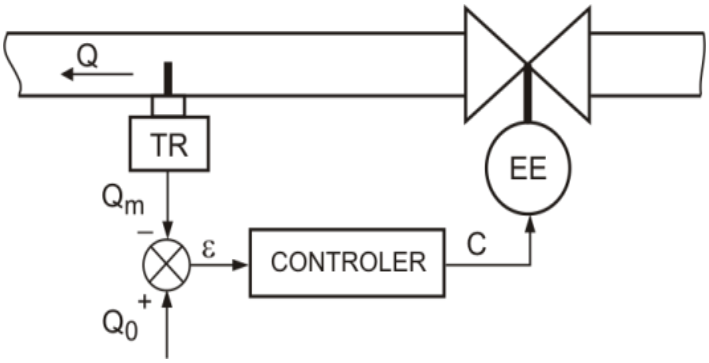
Înaintea controlerului se găseşte un comparator. Acesta efectuează operaţia de scădere între semnalul de referinţă şi semnalul primit de la senzor. Valoarea acesteia reprezintă "*eroarea*" sau abaterea. Controlerul prelucrează, conform algoritmului de control, eroarea şi furnizează la ieşire un semnal de comandă. Pentru a putea fi utilizat semnalul de comandă este in prealabil amplificat. La începutul intervalului de poziţionare, diferenţa între valoarea curentă şi referinţă, este mare, semnalul de comandă va avea valoarea maximă iar motorul va incepe să se rotească.

„Răspunsul sistemului" este reprezentat de mişcarea de rotaţie a braţului şi este cuantificat cu ajutorul senzorului de poziţie. Pe măsură ce motorul roteşte braţul, semnalul de eroare se va micşora iar la un moment dat va fi zero. În acest moment semnalul de comandă este nul deci, motorul se opreşte.

În cazul in care datorită perturbaţiilor externe (forte de frecare, moment rezistent la axul motorului etc) braţul robotic nu ajunge în poziţia dorită, controlerul sesizează eroarea şi efectuează corecţiile necesare



Pentru a putea controla automat debitul de fluid într-o conductă este necesar montarea unui traductor de debit. Prin intermediul traductorului TR se obţin informaţii privind debitul real Q_m în conductă. Aceste informaţii, pentru a putea fi analizate, sunt convertite într-un semnal (tensiune, curent etc) $u_m(t)$. În cazul senzorilor digitali informaţia privind debitul va fi reprezentată printr-un număr de biţi de ex. 10 sau 12



Valoarea obţinută de la senzor se compară cu valoarea impusă pentru debitul Q_0, exprimată prin semnalul u(t). Rezultatul operaţiei este reprezentat de semnalul de eroare:

$$e = u(t) - u_m(t)$$

Controlerul are rolul de a prelucra după o anumită lege, eroarea rezultată din comparaţia celor două mărimi şi de a furniza la ieşire un semnal de comandă C(t) pentru elementul de execuţie EE (actuator). Şi in acest caz semnalul de comandă este amplificat. Elementul de execuţie (actuatorul) deschide sau închide robinetul montat pe conductă.

În cazul în care debitul prin conductă este mai mic decât valoarea impusă controlerul va comanda deschiderea robinetului şi invers.

În funcţie de tipul algoritmului de control şi a robinetului montat pe conductă, debitul poate fi reglat secvenţial sau continuu după o anumită lege de variaţie.

Avantaje şi dezavantaje


Controlul în buclă închisă

❑ Principalele avantaje ale unui sistem in buclă închisă faţă de unul în buclă deschisă sunt:
- posibilitatea de a controla un sistem căruia nu i se cunoaşte exact modelul matematic
- reducerea sensibilităţii la variaţia parametrilor
- reducerea efectului factorilor perturbatori
- adaptabilitatea sistemului la condiţii diferite
- posibilitatea de a stabiliza un proces instabil

❑ Exemple de algoritmi de control in buclă închisă :
- ON-OFF
- P (proporţional)
- PID
- Fuzzy logic
- Reţele neuronale
- Control predictiv
- Control adaptiv
- Control optimal
- Control robust

FlexFORM - Program de formare profesionala flexibila pe platforme mecatronice

Principalele avantaje ale unui sistem in buclă închisă faţă de unul în buclă deschisă sunt:

- posibilitatea de a controla un sistem căruia nu i se cunoaşte exact modelul matematic

- reducerea sensibilităţii la variaţia parametrilor

- reducerea efectului factorilor perturbatori

- adaptabilitatea sistemului

- posibilitatea de a stabiliza un proces instabil

Exemple de algoritmi de control in buclă închisă :

- ON-OFF

- P (proporţional)

- PID

- Fuzzy logic

- Reţele neuronale

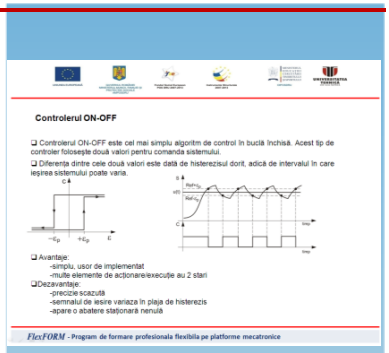
- Control predictiv

- Control adaptiv

- Control optimal

- Control robust

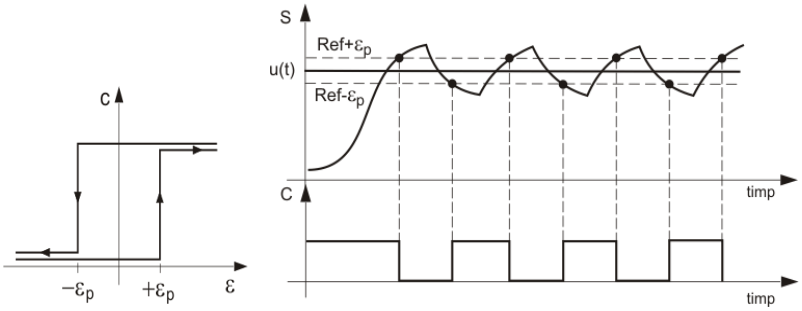
Controlerul bipoziţional (ON-OFF)



Controlul bipoziţional este cea mai simplă formă de control în buclă închisă. Acest tip de control foloseşte două valori pentru comanda sistemului:

- o valoare (ON) pentru pornirea sistemului atunci când valoarea citită de senzor este mai mică decât referinţa;
- o altă valoare (OFF) pentru oprirea sistemului atunci când valoarea citită de senzor depăşeşte referinţa.

Diferența dintre cele două valori este dată de histerezisul dorit, adică de intervalul în care ieşirea sistemului poate varia.



Acest tip de control se foloseşte în domeniile unde sistemul are inerţii mari şi precizia nu are mare importanţă.

Ex: Termostatul pentru centrale termice de apartament.

Rolul termostatului este de a menține temperatura la o anumită valoare de ex. 21^o cu o variație admisă de plus minus un grad. Asta înseamnă că el comandă pornirea centralei atunci când temperatura măsurată de senzor coboară sub 20^o şi opreşte centrala atunci când temperatura trece de 22^o. Deci putem spune că sistemul are un histerezis de 2^o

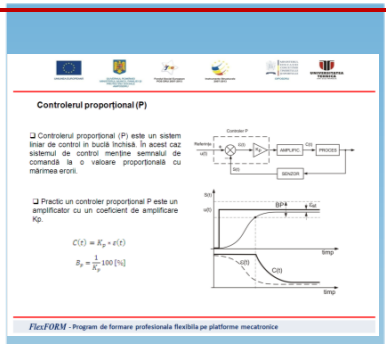
Avantaje:

- simplu, usor de implementat;
- multe elemente de acţionare/execuţie au 2 stari.

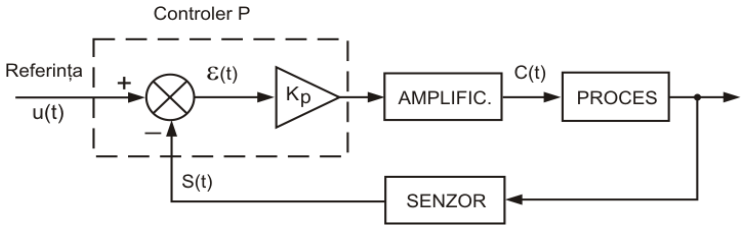
Dezavantaje:

- precizie scazută;
- semnalul de ieşire variază în plaja de histerezis;
- apare o abatere staţionară nenulă.

Controlerul proporţional (P)



În cazul algoritmului de control proporţional, sistemul de control menţine semnalul de comandă la o valoare proporţională cu mărimea erorii.



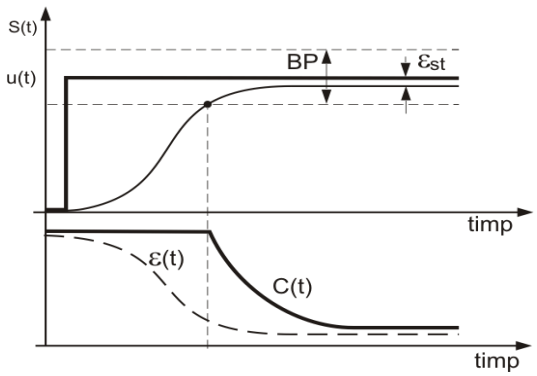
Practic un controler proporțional P este un amplificator cu un coeficient de amplificare *Kp*.

$$C(t) = K_P * \varepsilon(t)$$

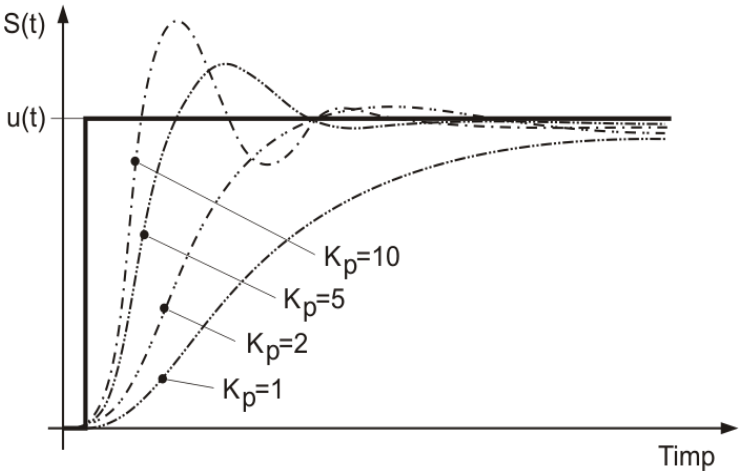
Uneori în loc de K_P se utilizează termenul ***bandă de proporţionalitate*** B_P. Acesta este definit ca fiind procentul din domeniul mărimii de intrare în regulator ε(t) pentru care controlerul proporțional determină o valoare C(t) egală cu 100% din domeniul posibil pentru mărimea de ieşire. Când domeniul de variaţie al erorii ε(t) este egal cu domeniul de variaţie al mărimii de comandă C(t), banda de proporţionalitate se determină cu relaţia:

$$B_P = \frac{1}{K_P} 100\ [\%]$$

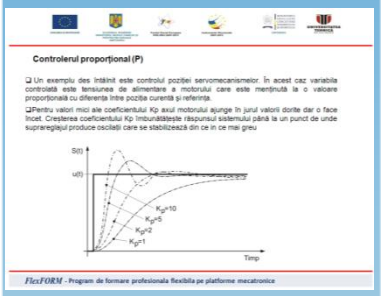
În figura de mai jos este reprezentat răspunsul la semnal de intrare treaptă al unui regulator proporţional. Eroarea staționară ε_{st} reprezintă diferenţa dintre valoarea referinţei şi valoarea răspunsului sistemului regim staţionar.



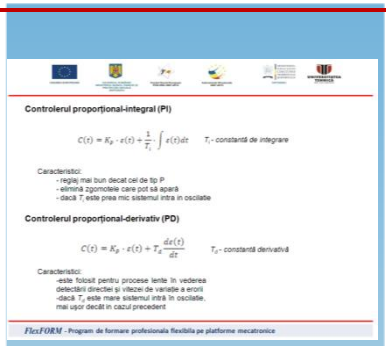
Un exemplu de utilizare a controlerului proporţional controlul poziției servomecanismelor. Circuitul de control are rolul de a modifica parametrii funcţionali ai servomotorului în funcţie de semnalul de referință. Determinarea poziției la care se află arborele servomotorului se face prin intermediul unui senzor de tip rezistiv (potențiometru). Modificarea unghiului de rotație al arborelui va determina modificarea poziției cursorului potențiometrului şi implicit tensiunea de ieșire a acestuia. Valoarea tensiunii este utilizată de sistemul de control pentru a determina poziția reală a servomecanismului și comparând-o cu referința va rezulta semnalul de comandă pentru motorul de curent continuu



Pentru valori mici ale coeficientului *Kp* axul motorului ajunge în jurul valorii dorite dar o face încet. Creșterea coeficientului *Kp* îmbunătățește răspunsul sistemului până la un punct de unde suprareglajul produce oscilații care se stabilizează din ce in ce mai greu.



Controlerul proporţional-integral (PI)



Acest controler stabileşte între mărimea de ieşire C(t) şi mărimea de intrare ε(t) o relaţie de dependenţă descrisă de relaţia :

$$C(t) = K_P \cdot \varepsilon(t) + \frac{1}{T_i} \cdot \int \varepsilon(t)dt$$

Constanta T_i se numeşte *constantă de integrare* şi se măsoară în unităţi de timp. Datorită posibilităţii de combinare a celor două acţiuni, proporţională şi integrală, prin modificarea simultană a celor două constante, controlerele *PI* permit obţinerea de performanţe superioare în realizarea legilor de reglare.

Caracteristici:

- reglaj mai bun decat cel de tip *P*
- elimină zgomotele care pot să apară
- dacă T_i este prea mic sistemul intra in oscilatie

Controlerul proporţional-derivativ (PD)

Aceste controlere, stabilesc între mărimea de ieşire C(t) şi mărimea de intrare ε(t) o relaţie de dependenţă descrisă de:

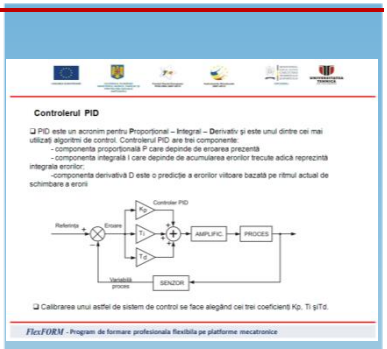
$$C(t) = K_P \cdot \varepsilon(t) + T_d \frac{d\varepsilon(t)}{dt}$$

unde factorul T_d se numeşte *constantă derivativă* şi se măsoară în unităţi de timp. Prezenţa componentei derivative care apare la momentul iniţial şi este de scurtă durată, are ca efect o accelerare a regimului tranzitoriu şi deci o reducere a acestuia

Caracteristici:

- folosit pentru procese lente în vederea detectării directiei şi vitezei de variaţie a erorii
- dacă T_d este mare sistemul intră în oscilatie, mai uşor decât in cazul precedent

Controlerul PID


Controlerul PID

❑ PID este un acronim pentru Proporțional – Integral – Derivativ și este unul dintre cei mai utilizați algoritmi de control. Controlerul PID are trei componente:
- componenta proporțională P care depinde de eroarea prezentă
- componenta integrală I care depinde de acumularea erorilor trecute adică reprezintă integrala erorilor;
- componenta derivativă D este o predicție a erorilor viitoare bazată pe ritmul actual de schimbare a eroriiControler PID
Referința
Eroare
Kp
Ti
Td
AMPLIFIC.
PROCES
Variabilă proces
SENZOR
❑ Calibrarea unui astfel de sistem de control se face alegând cei trei coeficienți Kp, Ti și Td.
FlexFORM - Program de formare profesională flexibilă pe platforme mecatronice

Aceste controlere sunt cele mai complexe regulatoare cu acţiune continuă, care asigură performanţe de reglare superioare, atât în regim staţionar cât şi în regim tranzitoriu. Ele înglobează efectele proportional *P*, integral *I* şi derivativ *D*, conform legii de reglare:

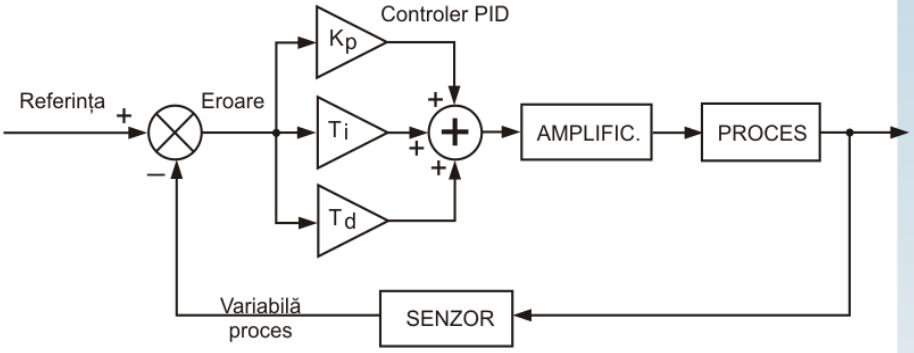
$$C(t) = K_P \cdot \varepsilon(t) + \frac{1}{T_i} \cdot \int \varepsilon(t)dt + T_d \frac{d\varepsilon(t)}{dt}$$

PID este un acronim pentru **P**roporțional – **I**ntegral – **D**erivativ şi este unul dintre cei mai utilizaţi algoritmi de control. Controlerul PID are trei componente:

- componenta proporţională *P* care depinde de eroarea prezentă

- componenta integrală *I* care depinde de acumularea erorilor trecute adică reprezintă integrala erorilor;

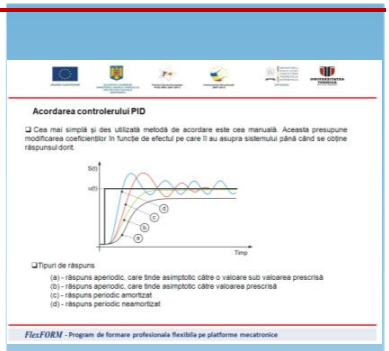
- componenta derivativă *D* este o predicție a erorilor viitoare bazată pe ritmul actual de schimbare a erorii



Caracteristici:

- performanţele cele mai bune în categoria regulatoarelor continue
- coeficientii, K_p, T_i, T_d trebuie “acordaţi” în conformitate cu comportamentul sistemului controlat (pe baza raspunsului la treaptă unitară)
- dacă reglajul nu este adecvat sistemul poate să intre în oscilatie
- acordarea se face pe baza unor criterii cum ar fi: eroare minimă, timp minim de atingere a valorii prescrise etc.

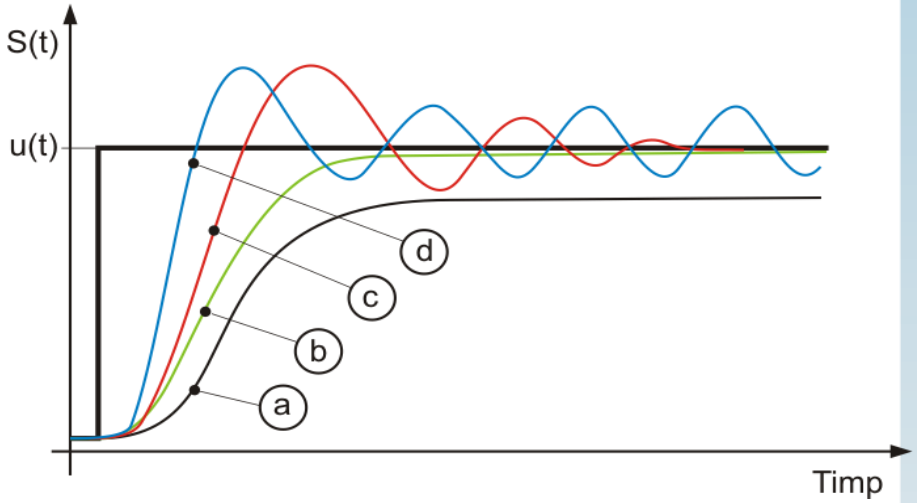
Acordarea controlerului PID



Nu există o solutie unică de acordare a controlerelor, aceasta se face în funcţie de obiectivul urmărit.

Metodele de acordare a regulatoarelor se bazează pe anumite criterii de performanţă, cum ar fi: criteriul minimizării abaterii maxime sau criteriul timpului minim de stabilizare

Cea mai simplă şi des utilizată metodă de acordare este cea manuală. Aceasta presupune modificarea coeficienților în funcție de efectul pe care îl au asupra sistemului până când se obţine răspunsul dorit.



Tipuri de răspuns

(a) - răspuns aperiodic, care tinde asimptotic către o valoare sub valoarea prescrisă. În acest caz există eroare in regin staţionar

(b) - răspuns aperiodic, care tinde asimptotic către valoarea prescrisă

(c) - răspuns periodic amortizat

(d) - răspuns periodic neamortizat

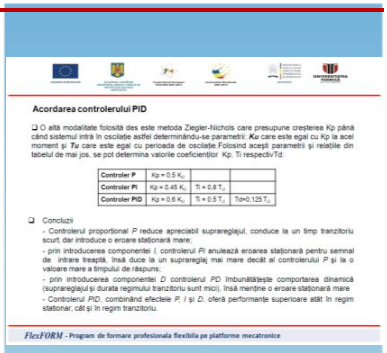
O altă modalitate folosită des este metoda Ziegler-Nichols. Această metodă de acordare se bazează pe criteriul suprafeţei minime pentru graficul abaterii. Pentru stabilirea valorilor parametrilor de acord ai controlerlui se parcurg următoarele etape:

- Se anulează efectul integrativ (T_i = ∞) şi derivativ (T_d = 0), iar constanta de proporţionalitate (K_P) se fixează la valoarea minimă.
- Se măreşte treptat valoarea constantei de proporţionalitate K_P până când sistemul intră în oscilaţie.
- Se măsoară perioada oscilaţiei (T_U) şi se notează valoarea constantei de proporţionalitate pentru care sistemul a intrat în oscilaţie (K_U).
- Conform criteriului suprafeţei minime se aleg următoarele valori pentru constantele regulatorului:

Controler P	Kp = 0,5 K_U		
Controler PI	Kp = 0,45 K_U	Ti = 0,8 T_U	
Controler PID	Kp = 0,6 K_U	Ti = 0,5 T_U	Td=0,125 T_U

Analizând comparativ controlere prezentate anterior se desprind următoarele concluzii

- Controlerul proporţional *P* reduce apreciabil suprareglajul, conduce la un timp tranzitoriu scurt, dar introduce o eroare staţionară mare;
- Prin introducerea componentei *I,* controlerul *PI* anulează eroarea staţionară pentru semnal de intrare treaptă, însă duce la un suprareglaj mai mare decât al controlerului *P* şi la o valoare mare a timpului de răspuns;
- Prin introducerea componentei *D* controlerul *PD* îmbunătăţeşte comportarea dinamică (suprareglajul şi durata regimului tranzitoriu sunt mici), însă menţine o eroare staţionară mare
- Controlerul *PID*, combinând efectele *P, I* şi *D*, oferă performanţe superioare atât în regim stationar, cât şi în regim tranzitoriu.


Acordarea controlerului PID

❑ O altă modalitate folosită des este metoda Ziegler-Nichols care presupune creşterea Kp până când sistemul intră în oscilaţie astfel determinându-se parametrii **Ku** care este egal cu Kp la acel moment şi **Tu** care este egal cu perioada de oscilaţie.Folosind aceşti parametrii şi relaţiile din tabelul de mai jos, se pot determina valorile coeficienţilor Kp, Ti respectivTd:

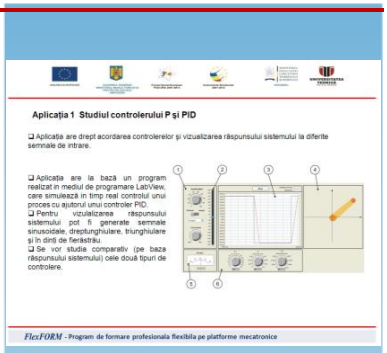
Controler P	$Kp = 0,5\ K_U$		
Controler PI	$Kp = 0,45\ K_U$	$Ti = 0,8\ T_U$	
Controler PID	$Kp = 0,6\ K_U$	$Ti = 0,5\ T_U$	$Td=0,125\ T_U$

❑ Concluzii

- Controlerul proporţional *P* reduce apreciabil suprareglajul, conduce la un timp tranzitoriu scurt, dar introduce o eroare staţionară mare;
- prin introducerea componentei *I*, controlerul *PI* anulează eroarea staţionară pentru semnal de intrare treaptă, însă duce la un suprareglaj mai mare decât al controlerului *P* şi la o valoare mare a timpului de răspuns;
- prin introducerea componentei *D* controlerul *PD* îmbunătăţeşte comportarea dinamică (suprareglajul şi durata regimului tranzitoriu sunt mici), însă menţine o eroare staţionară mare
- Controlerul *PID*, combinând efectele *P, I* şi *D*, oferă performanţe superioare atât în regim stationar, cât şi în regim tranzitoriu.

FlexFORM - Program de formare profesionala flexibila pe platforme mecatronice

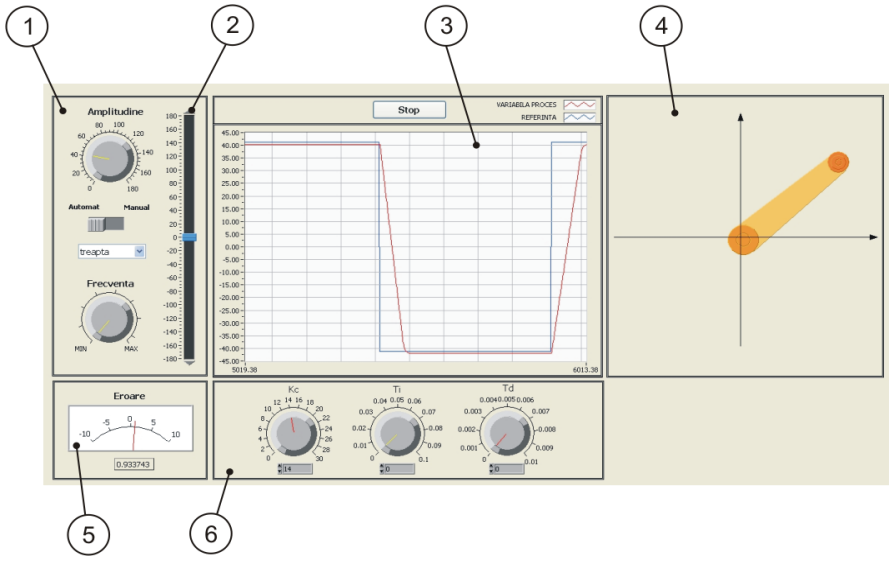
Aplicaţia 1 Studiul controlerului P şi PID



Scopul aplicaţiei este acordarea controlerelor P şi PID şi vizualizarea răspunsului sistemului la diferite semnale de intrare.

Aplicaţia are la bază un program realizat in mediul de programare LabVIEW, care simulează in timp real controlul unui proces cu ajutorul unui controler PID. Pentru a rezulta un controler proporţional, se anulează efectul integrativ şi derivativ.

Pentru vizulalizarea răspunsului sistemului pot fi generate semnale sinusoidale, dreptunghiulare, triunghiulare şi în dinţi de fierăstrău



Amplitudinea şi frecvenţa semnalelor generate se pot modifica cu ajutorul butoanelor *1* . Cu ajutorul cursorului *2* se poate stabili manual valoarea semnalului de referinţă. Forma semnalului de referinţă şi răspunsul sistemuului sunt afişate în timp real pe graficul *3.* Eroarea staţionară este afişată pe indicatorul *5*. Valoarile parametrilor de acord ai controlerului pot fi modificate cu ajutorul butoanelor *6.*

În fereastra 4 este simulată funcţionarea unui braţ robotic a cărui poziţie unghiulară este controlată cu ajutorul acestui controler. Valoarea semnalului generat reprezintă poziţia unghiulară a braţului (referinţa) Se vor studia comparativ (pe baza răspunsului sistemului) cele două tipuri de controlere.

Aplicaţia 2 Controlul poziţiei unui servomotor de curent continuu


Aplicaţia 2 Controlul poziţiei unui servomotor de curent continuu

Aplicaţia vizează controlul poziţiei axului unuia din motoarele utilizate la antrenarea platformei mobile. În acest scop in microcontroler va fi scris un program care să citească senzorul in cuadratură montat pe axul motorului şi să transmită datele la calculator prin intermediul interfeţei seriale.

Programul va primi de la calculator semnalele de comandă pentru motorul de curent continuu tot prin intermediul interfeţei seriale. Algoritmul de control va fi implementat pe calculator cu ajutorul unui program scris in LabVIEW.

Structura programului pentru microcontroler este următoarea:

```
program flex_motor_pro_unit_pid_labview;
uses flexunit;
var  p:byte;
   in0:integer;
procedure int_0; org IVT_ADDR_INT0;
begin
if pina.6=1 then in0:=in0-1
            else in0:=in0+1;
end;

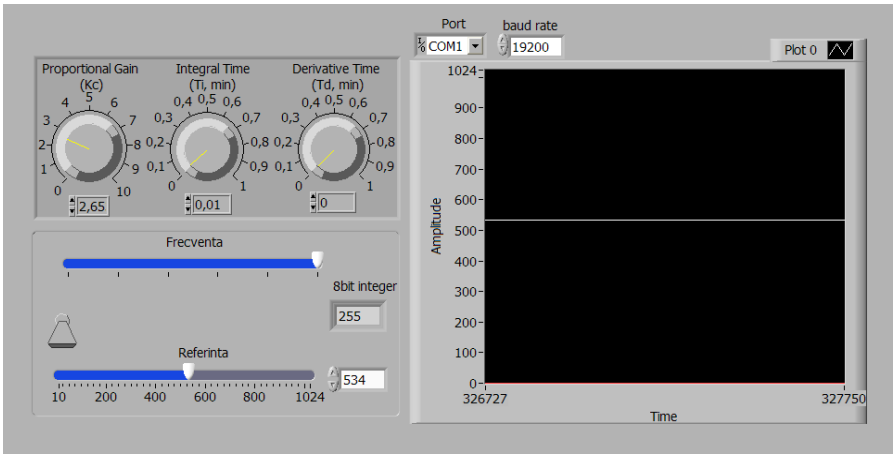
 begin
    init_pwm;
    init_int0;
    serial_init;
    porta.6:=1;
    in0:=0;
    motor0(0);
   repeat
   if (UART1_Data_Ready() = 1) then
      begin
           p:= UART1_Read();
           motor0(p-127);
      end;
    serial_send1(in0);
    until 1<>1;
end.
```

Aplicaţia 2 Controlul poziţiei unui servomotor de curent continuu (continuare)



Programul scris in LabView va citi semnalele trimise de microcontroler le va compara cu referinţa şi apoi va genera semnalele de comandă în vederea minimizării erorii de poziţionare. Practic semnalul de comandă este un semnal PWM al cărui factor de umplere stabileşte viteza de rotaţie a motorului.

Pentru acordarea controlerului PID, programul permite generarea unui semnal sinusoidal cu frecvenţă variabilă. Forma semnalului de referinţă şi răspunsul sistemului (motor de curent continuu + reductor + roată) sunt afişate grafic.



Se vor studia comparativ răspunsurile sistemului pentru diferite valori ale celor trei parametrii de acord ai regulatorului.

Aplicaţia 3 (demo) Sincronizarea turaţiei motoarelor de antrenare a platformei mobile

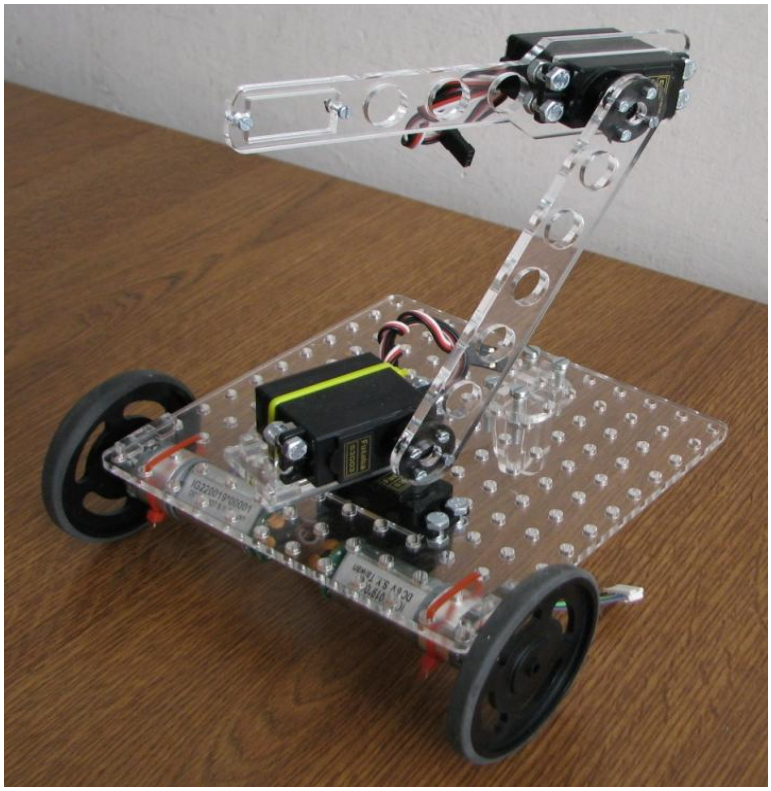

Aplicaţia 3 (demo) Sincronizarea turaţiei motoarelor de antrenare a platformei mobile

❑ Pentru deplasarea platformei în linie dreaptă este necesară sincronizarea turaţiei celor două motoare de antrenare.

❑ În acest scop au fost elaborate două programe : primul care controlează separat turaţia fiecărui motor pe baza unui semnal de referinţă comun; al doilea care sincronizează motoarele ţinând cont de diferenţa de turaţie dintre cele două motoare diferenţă datorată perturbaţiilor (obstacole, coeficienţi de frecare diferiţi etc). Semnalul de referinţă pentru turaţia motoarelor se obţine cu ajutorul unui potenţiometru conectat la placa de dezvoltare

❑Se vor studia comparativ cele două soluţii.

Scopul aplicaţiei este comanda deplasării in linie dreaptă a platformei mobile. Pentru aceasta este necesară sincronizarea turaţiei celor două motoare de antrenare.



În acest scop au fost elaborate două programe :

- primul care controlează separat turaţia fiecărui motor pe baza unui semnal de referinţă comun;

- al doilea care sincronizează motoarele ţinând cont de diferenţa de turaţie dintre cele două motoare diferenţă datorată perturbaţiilor (obstacole, coeficienţi de frecare diferiţi etc).

Semnalul de referinţă pentru turaţia motoarelor se obţine cu ajutorul unui potenţiometru conectat la placa de dezvoltare.

Se vor studia comparativ cele două solutii.