SISTEM DE CALCUL IN TIMP REAL

SCTR

SZOKE ENIKO

Curs 2

Cuprins

Sistem de calcul in timp real

- Sistem de supraveghere si comanda proceselor
- Sisteme tranzactionale, de informare sau conducere (luare de decizii)
- Procese rapide - din Actionari electrice
- Supraveghere umana: culegere selectiv informatii
  analiza situatii
  decizii logice

Sistem de calcul in timp real

- Sistem de calcul:

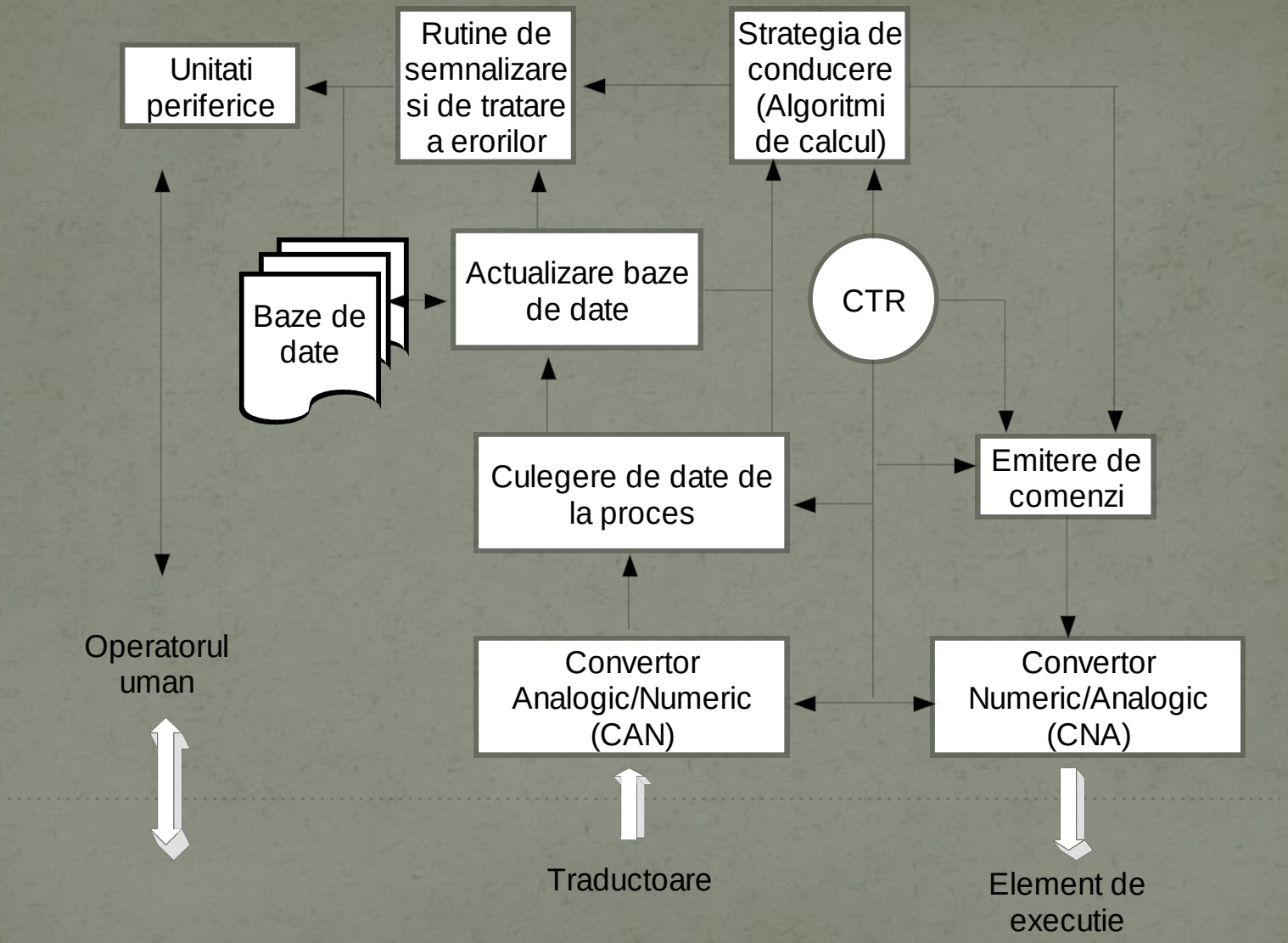
- Capacitate nelimitata de efectuare operatiunilor de supraveghere si comanda
- Volumul si viteza mare de culegere si stocare a informatiilor de la proces
- Capacitatea din ce in ce mai mare de prelucrare a informatiilor
- Obiectivitatea si precizia SC

in luarea deciziilor si

punerea in aplicare

(fara “pile sau obligatii”)

Structura unui sistem de calcul in timp real



Structura unui sistem de calcul in timp real

Traductoarele:

Poarta de intrare a informatiilor pentru SC.

Convertesc informatiile variabilelor ce descriu procesul.

Masoara marimi electrice/neelectrice si le transforma in marimi electrice (analogice sau numerice)

Calitati:

Redarea cat mai precis variatia marimilor masurate

Reactia cat mai rapida la modificarea marimilor masurate

Ieftin, robust

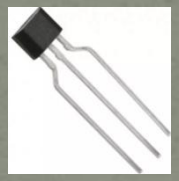
In actionari electrice: traductoare de curent

traductoare de tensiune

de flux (Hall)

de viteza, pozitie

(tahogeneratoare, encodere, resolvere)



Din pct.de vedere a fiabilitatii si robustetii cele mai problematice sunt tr. de viteza si de pozitie

actionari fara senzori mecanici (observere, estimatoare).

Constantele de timp de ordinul µs.

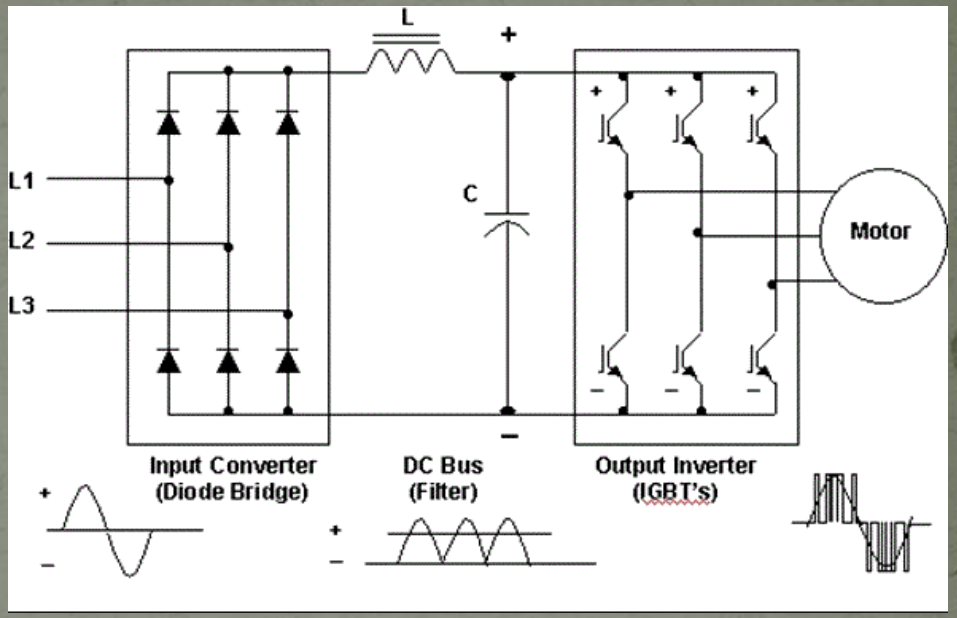
Structura unui sistem de calcul in timp real

Element de executie (EE):

Interfata dinspre SCTR inspre proces.

Intr-un sistem de actionare electrica comandat de SC:

EE = convertorul electronic de putere care alimenteaza motorul.



f cu care convertorul primeste noi comenzi (e capabil?) ~ timp de raspuns a SC.

$t_{convertor} = intreg * t_{raspuns\ SC}$

Probl: Fie un sistem de actionare cu un convertor cu frecventa maxima de comanda de 25kHz (40 µs). Timpul de raspuns al SC este de 200 µs.

Structura unui sistem de calcul in timp real

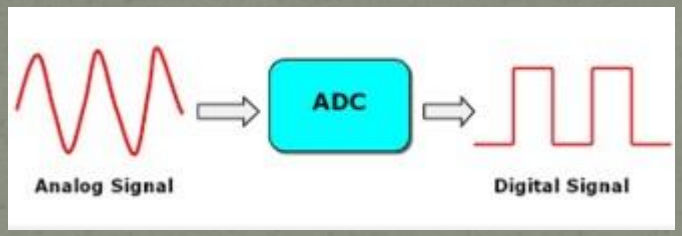
Convertoare Analog/Numerice (CAN), Numeric/Analogice (CNA):

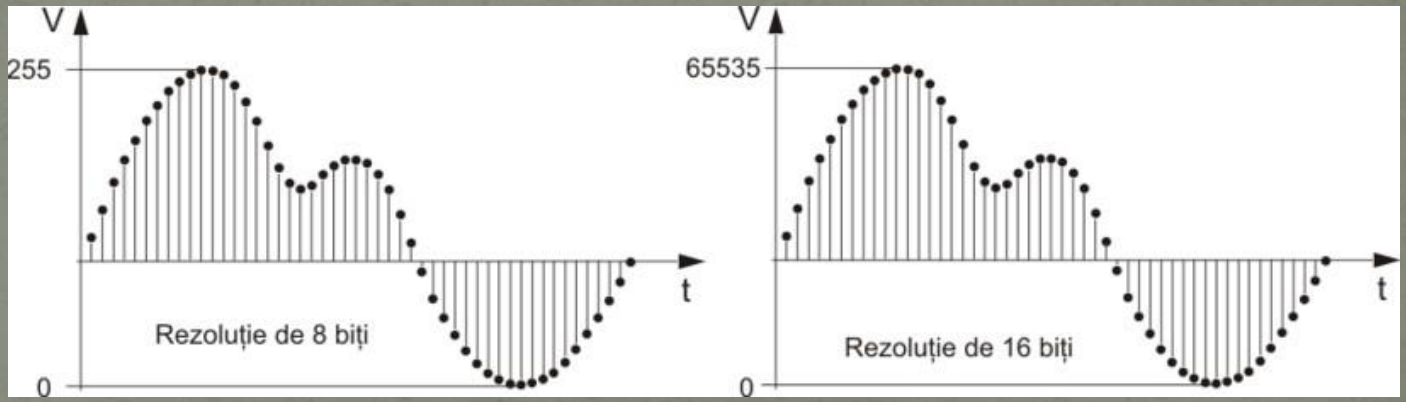
Transforma marimile existente intre proces si SC.

Calitati:

Timpul de conversie(µs)

Nr de biti pe care se face conversia





Structura unui sistem de calcul in timp real

Reprezentarea numerelor reale

Diferentiere intre numerele intregi si numerele reale.

Nr real - exista spatiu finit → reprezentare numai a unei submultimi **nr. rational.**

Orice nr real se reprezinta in SC aproximat printr-un nr. rational.

n - biti

n este multiplu de 8 (cuvant, dublucuvant)

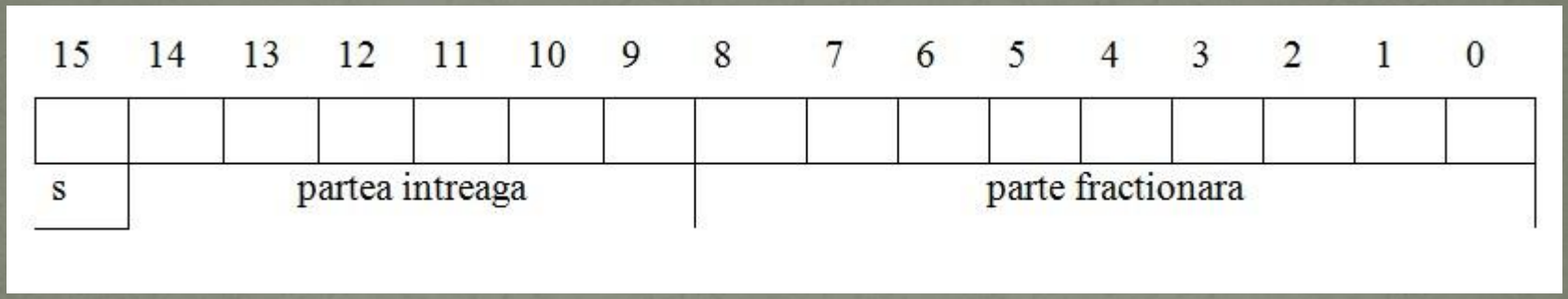
Reprezentarea semnului - folosind bitul de semn:
- 0 nr pozitiv
- 1 nr negativ

Reprezentarea in virgula fixa

Reprezentarea in virgula mobila

Reprezentarea in virgula fixa:

- Este o reprezentare naturala: semn + partea intreaga + partea fractionara
- Bitul cel mai semnificativ este rezervat semnului.
- Daca partea intreaga a nr. are prea putine cifre, se adauga la stanga zerouri suplimentare.
- Daca partea intreaga are prea multe cifre se pierd cifrele cele mai semnificative ce depasesc spatiul de reprezentare.



Exemplu: Sa se reprezinte (+10011,0101010101)2

(-10011,0101010101)2

(+11110011,010101010

Se limiteaza gama nr reprezentabile.

Reprezentarea in virgula mobila:

La depasire se pierd cifrele cele mai putin semnificativ

$$x = \pm\ o,m * b^e$$

m - mantisa numarului

b - o baza de numeratie

e - exponent

Standarde de reprezentare:

IEEE simpla precizie (4 octeti)

s	c=e+127		m	
31	30	23	22	0

IEEE dubla precizie (8 octeti)

Turbo Pascal Extended (10 octeti)

In aceste forme mantisa m = (1 si 2)

Exemplu:

$(-3572,54)_{10}$ → IEEE simpla precizie

- se trece nr in baza 16 retinand 14 cifre care sunt suficiente pt reprezentare
- $(-DF4,8A3D708A3D7\ldots\ldots)_{16}$
- $(-1101\ 1111\ 0100,1000\ 1010\ 0011\ 1101\ 0111\ 0000\ldots\ldots\ldots\ldots)_{2}$
- forma normalizata si se retine numai 52 de cifre dupa virgula
- $(-1,1011\ 1110\ 1001\ 0001\ldots\ldots..)_{2} * 2^{11}$
- $(-1,BE914\ldots.)_{16} * 2^{11}$
- semnul s = "- "
- $e = (11)_{10}$
- caracteristica $c = e + 127 = (138)_{10} = (8A)_{16} = (10001100)_{2}$

s	c=e+127	m
1	10001010	10111110100100010100011

$(C55F48A3)_{16}$

Conversii

- intregul 15, reprezentat in binar, pe un octet se reprezinta astfel: 0000 1111 sau 0fh

- Conversie hexa – zecimal:
  $2a75.bdb3_{16}=2*16^3+a*16^2+7*16^1+5*16^0+b*16^{-1}+d*16^{-2}+b*16^{-3}+3*16^{-4}=$
  $2*16^3+10*16^2+7*16^1+5*16^0+11*16^{-1}+13*16^{-2}+11*16^{-3}+3*16^{-4}=10869.7414703369141_{10}$
- Conversie binar – hexa: $101101100111011_2 = 5B3B_{16}$
- Conversie hexa – binar: $5B2C_{16} = 0101101100101100_2$

Structura unui sistem de calcul in timp real

Strategii de conversie si achizitie:

1) Principul DMA "Direct Memory Acces". Achizitia de date fara implicarea microprocesorului.

Creste viteza de acces a datelor

Timpul de raspuns al SC se micsoreaza

2) Utilizarea de memorii rapide RAM "Random Acces Memory" (Gb)

3) Utilizarea unui al doilea procesor specializat pentru achizitie si prelucrare primara a datelor (programare multiprocesor) MASTER-SLAVE

Structura unui sistem de calcul in timp real

Ceasul de timp real (CTR):

Inima SC.

Sincronizeaza :

- procesul de prelucrare numerica a informatiilor,
- procesul de culegere si reactualizare date
- emiterea comenzilor
- procese de conversie.

Structura unui sistem de calcul in timp real

- Orice SCTR utilizează CTR pentru:
  - a genera întreruperi la anumite intervale de timp
  - întreținere dată şi oră
  - stabilire intervale de eşantionare a procesului
  - stabilire intervale de comunicații
  - memorare date pe disc
  - elaborare de rapoarte la imprimantă
  - sistemul de operare timp real pentru planificarea şi dispecerizarea task-urilor

Esential in programarea concurenta.

Operatii consumatoare de timp:

Culegere si achizitie de date:

traductor - CAN - culegere de date - prelucrare

Procesarea numerica a datelor pe baza algoritmilor de conducere si control

Emiterea si aplicarea comenzii

emiterea comenzii - CAN - element de executie

Concluzie: Trebuie ales traductoarele cu const. de timp mica,
convertoare cu timp de conversie mic si
algoritmi de comanda optimali (texecutie)

Structura unui sistem de calcul in timp real

Tipuri de programe:

- Programe de culegere si prelucrare primara a datelor de la proces
- Algoritmi de comanda si control obtinuti pe baza unor strategii de conducere a procesului
- Programe de generare si emitere a comenzilor pentru executie (spre proces)
- Programe de actualizare de baze de date
- Programe de tratare a erorilor si generare de semnalizari

Structura unei interfete utilizator pentru sisteme de calcul in timp real

Interfata:

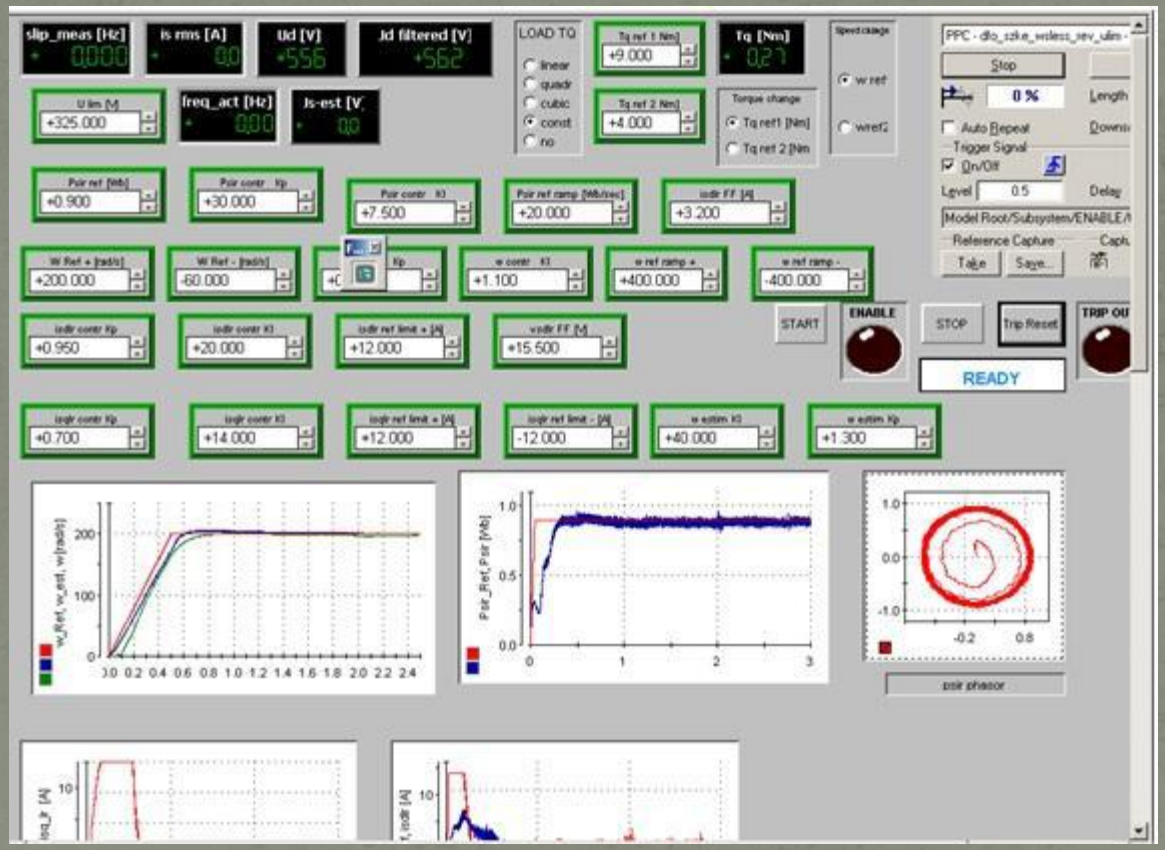
Totalitatea elementelor de hardware si de software care au ca scop sa faciliteze comunicarea intre SCTR si operatorul uman.

Calitati:

Utilizatorul poate lua decizii critice pe baza unor informatii provenite de la SC. Deci este important comunicarea foarte buna intre SC si operator uman.

Structura unei interfete utilizator pentru sisteme de calcul in timp real

Exemplu de interfață de comandă și control realizată în mediul Control Desk.



Probleme specifice la implementare:

Structura unui SCTR pt. c-da AE depinde atat de partea hardware cat si de partea software legate de programele de sistem si de aplicatie.

- Probleme specifice legate de echipamentul de calcul (hardware)
  - Interfete de cuplare la proces
  - Terminale specializate
  - Sisteme de intreruperi utilizate
  - Performantele ceasului in TR
  - Fiabilitate - cuplarea unui sistem de forta (energii) cu un sistem de curenti slabi (informatii)

- Probleme specifice legate de software
  - Alegerea sistemului de operare in TR optim
  - Utilizarea pt programele de aplicatie a unor limbaje de programare in TR

- Probleme specifice legate de ingineria proiectarii si implementarii in TR

Probleme specifice legate de interfete de cuplare la proces

- Adaptarea semnalelor, adica interfațarea perifericului cu calculatorul.

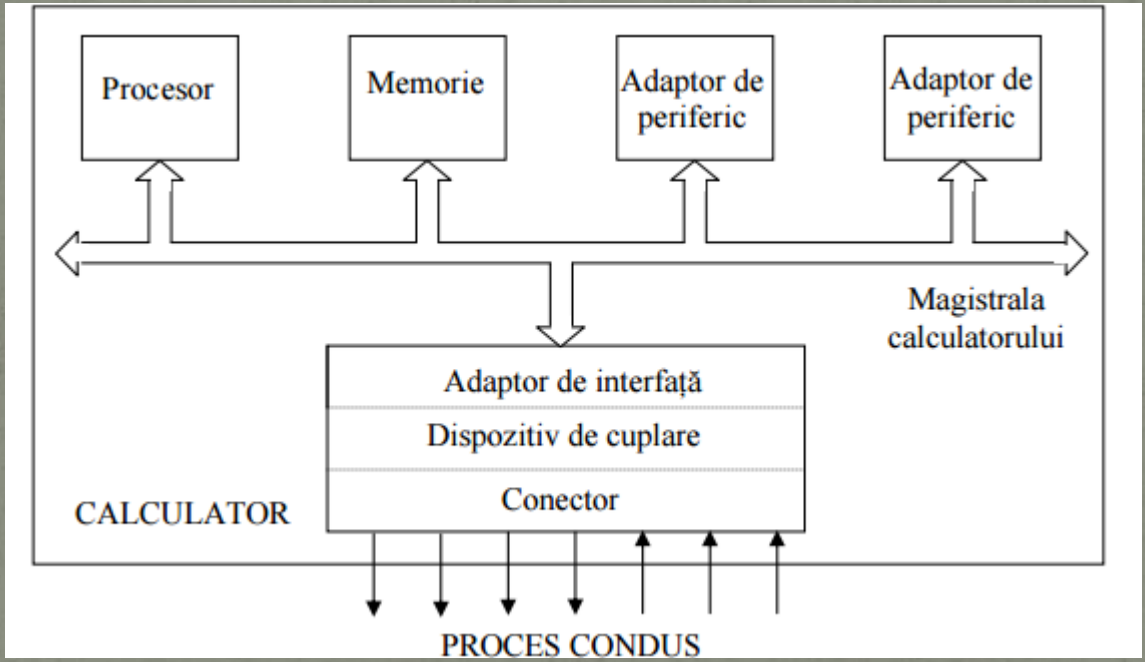
  Această funcție este asigurată de către un **adaptor de interfață**, denumit simplu și **interfață**, care convertește semnalele de la echipamentul periferic în semnale normalizate care pot fi transmise procesorului pe un canal intrare/ieșire. Deoarece traficul aferent fiecărui periferic este în general scăzut, canalele intrare/ieșire sunt de regulă multiplexate.

  Unitatea de interfață trebuie să fie modulară pentru a permite adaptarea ușoară a SC la caracteristicile particulare ale procesului condus. Modularitatea este asigurată prin plăci sau module conectate la o magistrală comună.

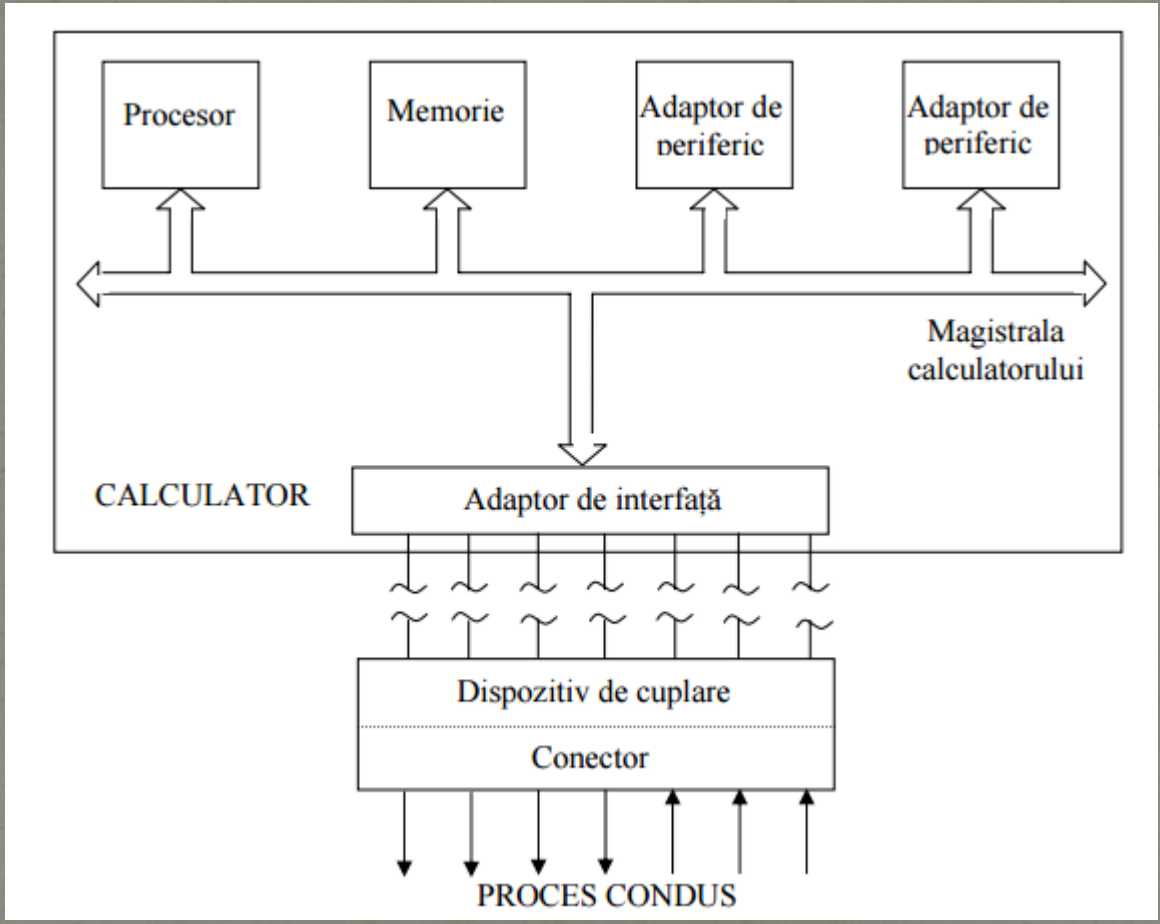
O interfață de proces este alcătuită din trei componente de bază:

- **conectorul**, care asigură legătura mecanică/electrică între cablajul calculatorului şi cablajul exterior
- **dispozitivul de cuplare** care joacă rolul de interfață electrică cu senzorii şi elementele de execuție
- **adaptorul de interfață** care asigură conectarea la magistrala calculatorului.

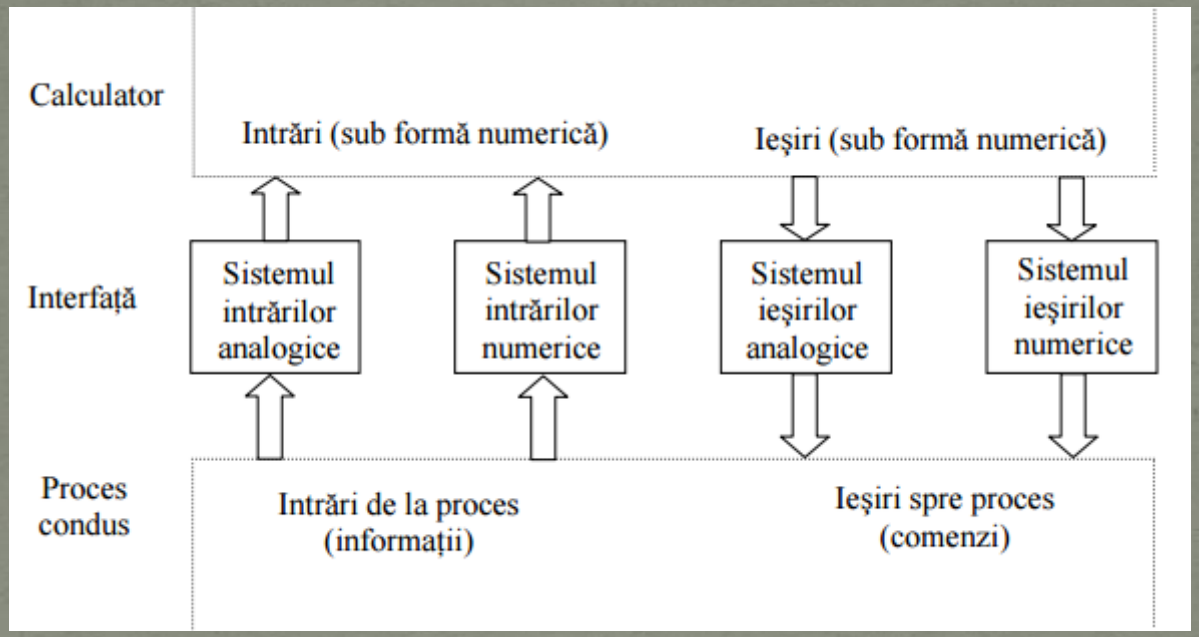
Unitatea de interfață de proces poate fi plasată în interiorul calculatorului : rezultând o structură integrată care are avantajul simplității, dar are și dezavantaje. Prezența în interiorul calculatorului a unor curenți de nivel mare poate genera perturbații și poate duce la apariția defectelor. Pe de altă parte, legăturile cu senzorii care furnizează semnale de nivel mic pot fi lungi (calculatorul poate fi plasat departe de procesul condus) și atunci trebuie să se adopte sisteme de transmisie analogice care pot fi scumpe.



Dispunerea conectorului în exteriorul calculatorului. Uneori se plasează în exteriorul calculatorului atât conectorul cât şi dispozitivul de cuplare la proces. Această arhitectură prezintă inconvenientul unor legături lungi între adaptorul de interfață şi dispozitivul de cuplare, ceea ce poate duce la limitarea performanțelor interfeței de proces.

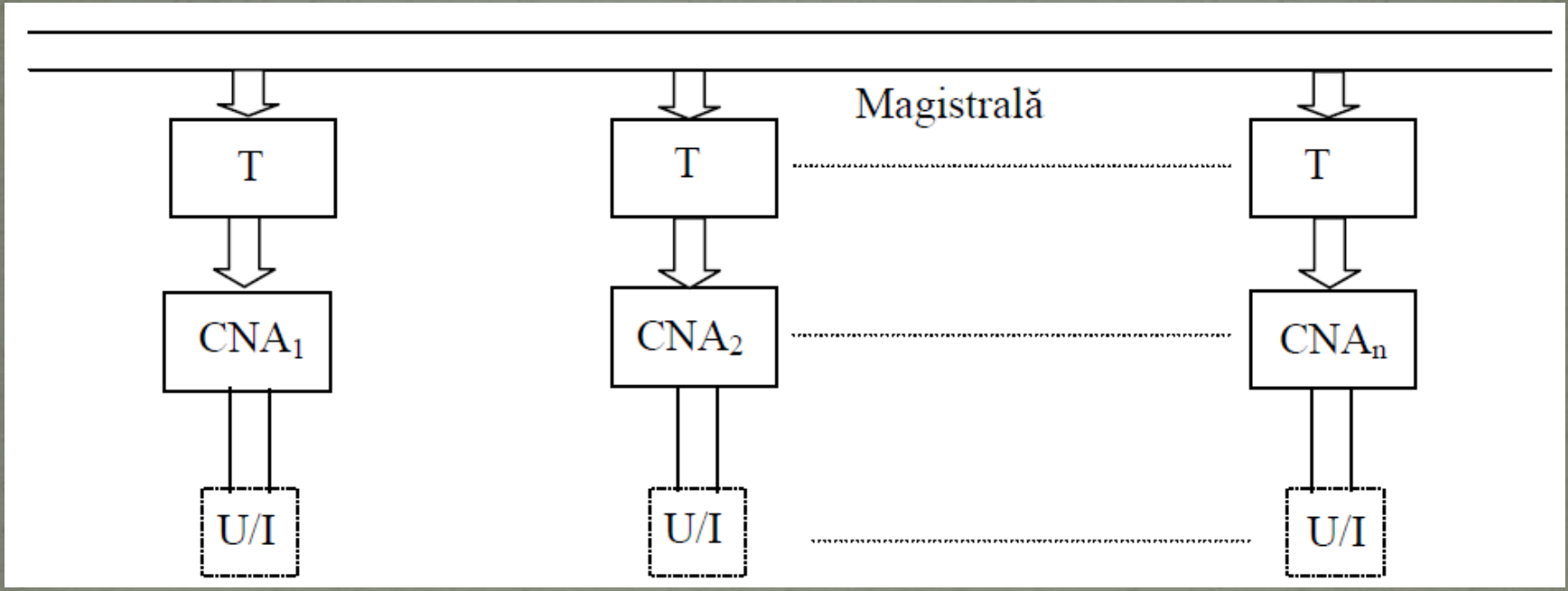


Indiferent de arhitectura interfeței de proces, structura generală este prezentata mai jos, care pune în evidență realizarea funcțiilor principale ale interfeței de proces prin subsistemele componente.



Sistemul de interfata iesiri analogice

Sistemul de ieşiri analogice este o componentă a interfețelor de proces destinată transmiterii spre proces a comenzilor în formă analogică, elaborate în urma rulării unor programe care implementează algoritmii de conducere. Sistemul de ieşiri analogice conține în mod necesar convertoare numeric-analogice (CNA)



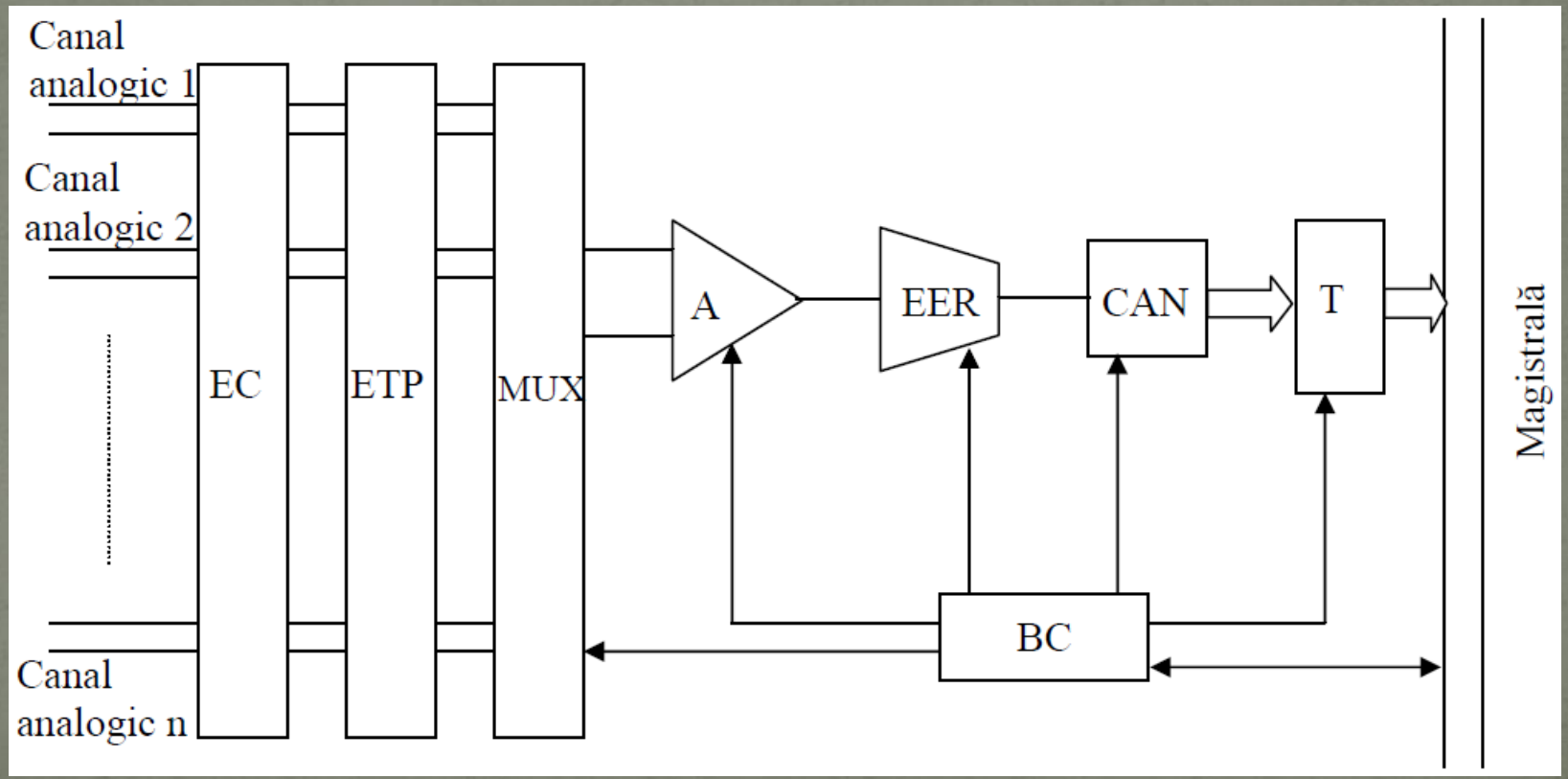
Tampoanele (buffere) T pastreaza valoarea numerica a comenzii. În unele cazuri ieşirea CNA (semnal de tip tensiune) este convertită în semnal de curent prin convertoare tensiune/curent U/I.

Sistemul de interfata iesiri numerice

Acest sistem este o componentă a interfeței de proces care este destinată transmiterii spre proces a comenzilor numerice. Comenzile sunt de obicei în formă binară (de regulă cu un singur rang), de tip tren de impulsuri, frecvență de impulsuri, durată de impuls. Sistemul ieşirilor numerice conține şi amplificatoare de putere şi circuite de separare galvanică.

Sistemul de interfata intrari analogice

Este o componentă a interfeței de proces destinată interfațării sistemului de conducere cu semnalele analogice din proces. Acestea se prezintă de regulă sub formă de semnale de curent continuu: intensitate (0-20 mA, 2-10 mA, 4-20 mA etc.) sau tensiune (de nivel mic: 10-20 mV, -10 - +10 mV, de nivel mediu: 0 - 10 V, 2 - 10 V, 0 - 20 V, de nivel mare 0-100 V).

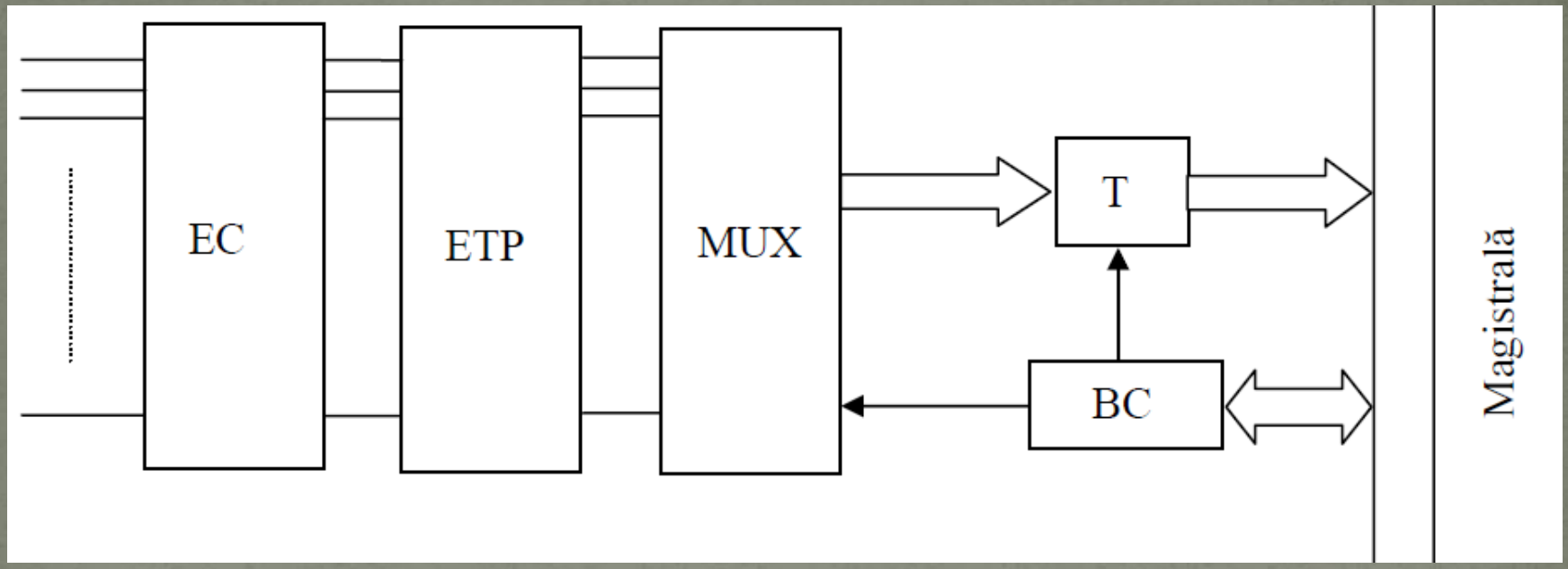


Componentele sistemului sunt:

- elementul de conectare (EC) care are rolul de a conecta la sistem conductoarele care transportă semnalul analogic de la proces;
- elementele de tratare primară a informației (ETP) destinate unor prelucrări care nu necesită amplificare (conversii curent/tensiune, filtrare);
- multiplexorul (MUX) care permite selectarea unuia din cele *n canale analogice;*
- amplificatorul (A) care adaptează nivelul semnalului de intrare selectat de multiplexor şi impedanța canalului la elementul următor;
- elementul de eşantionare şi reținere (EER) (sau circuit *Sample/Hold) care are rolul* de a păstra constantă valoarea eşantionului de tensiune pe durata conversiei analognumerice;
- convertorul analogic-numeric (CAN)
- bufferul T care este necesar conectării ieşirii CAN la magistrala sistemului de conducere;
- blocul de comandă BC care asigură coordonarea operațiilor care se desfăşoară în sistem.

Sistemul de interfata intrari numerice

Sistemul intrărilor numerice este un subsistem al interfețelor de proces care are rolul de a colecta din proces informațiile de formă numerică (semnale binare cu unul sau mai multe ranguri, tren de impulsuri) sau cvasinumerică (durată de impuls, frecvența impulsurilor).



Componentele sistemului sunt:

- elementul de conectare (EC);
- elementele de tratare primară a informației (ETP) - izolare galvanică, filtrări şi protecții
- multiplexorul (MUX) cu ajutorul căruia se selectează un semnal sau un grup de semnale
- bufferul de conectare la magistrală (T);
- blocul de comandă BC care asigură secvențializarea corectă a operațiilor din sistemul de intrări numerice

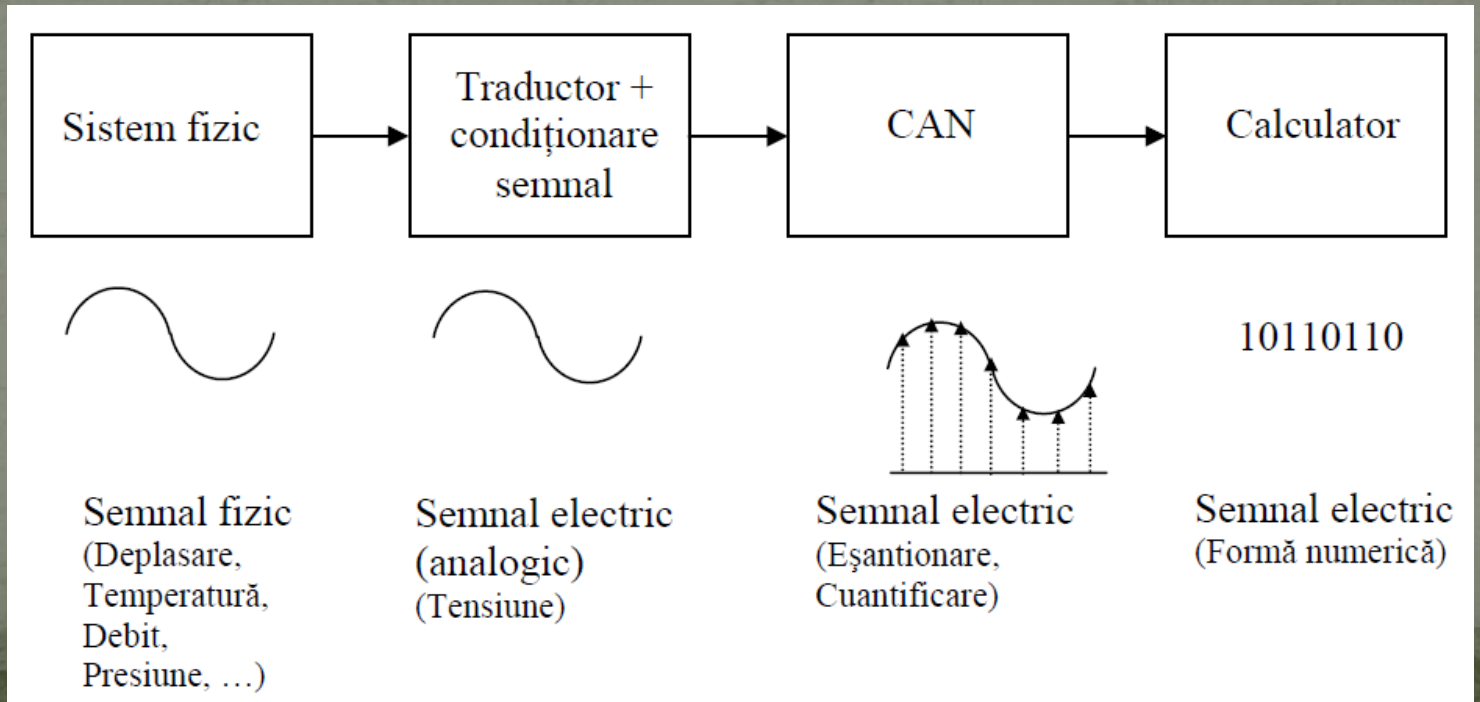
Interfețele de proces asigură:

- **funcții de conversie**
- **functii de protecție**
- functii de **adaptare**

Prin **condiționarea semnalelor** se înțelege în sens larg adaptarea dintre traductoare şi circuitele de conversie analog-numerică.

Tipul de condiționare depinde evident de senzorii care sunt utilizați.

De exemplu, un semnal poate avea nivel mic şi necesită o amplificare, sau poate conține componente parazite care cer realizarea unei filtrări.



Condiționarea semnalelor se realizează prin operații cum ar fi:

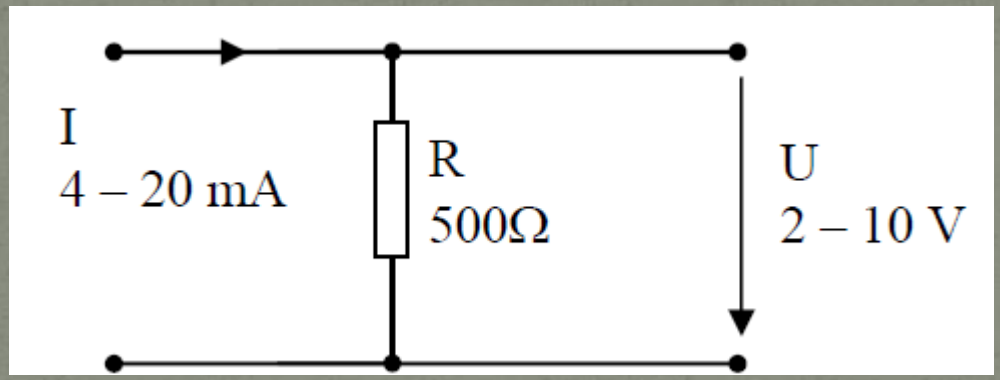
- conversii de semnal (cum ar fi conversia curent/tensiune)
- izolare galvanică
- amplificare
- filtrare
- liniarizare
- multiplexare
- alimentarea senzorilor pasivi

Conversii de semnal

Conversie curent/tensiune

În cazul în care traductorul furnizează la ieşire semnal unificat de curent continuu (2-10 mA, 4-20 mA), trebuie realizată o conversie în tensiune, care de altfel se face foarte simplu, cu ajutorul unei rezistenţe. Valoarea rezistenţei se determină în funcţie de curentul de ieşire al traductorului şi de domeniul de tensiune dorit. De exemplu:

I=4 ÷ 20 mA şi tensiune U = 2 ÷ 10V rezultă rezistenţa R = 500Ω



Dacă traductorul furnizează un curent de valoare mică este necesară folosirea unui convertor curent/tensiune cu amplificator operaţional.

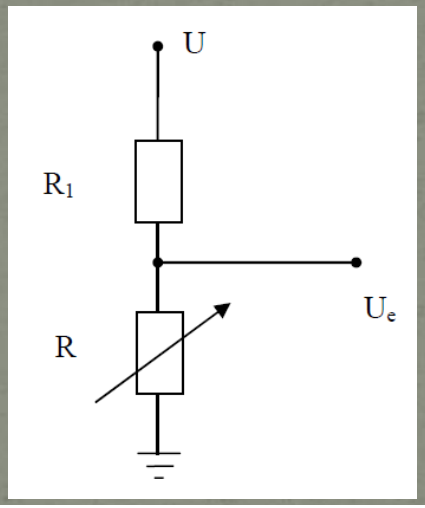
Conversie rezistenta/tensiune

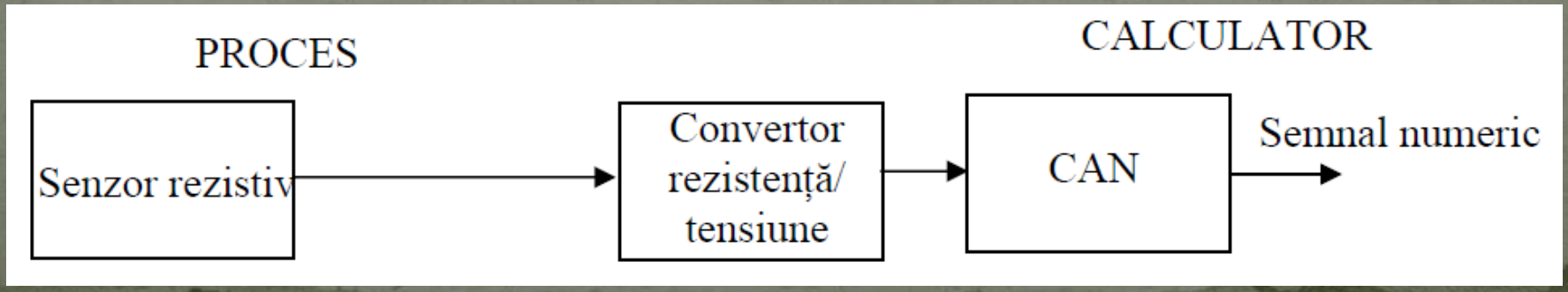
În cazul senzorilor a căror ieşire este de tip rezistență (termorezistențele) este necesară conversia variațiilor rezistenței în variații de tensiune.

Această conversie se poate realiza cu un simplu montaj potențiometric

Dacă R este rezistența senzorului, tensiunea de ieşire este:

$$U_e = \frac{RU}{R + R_1}$$

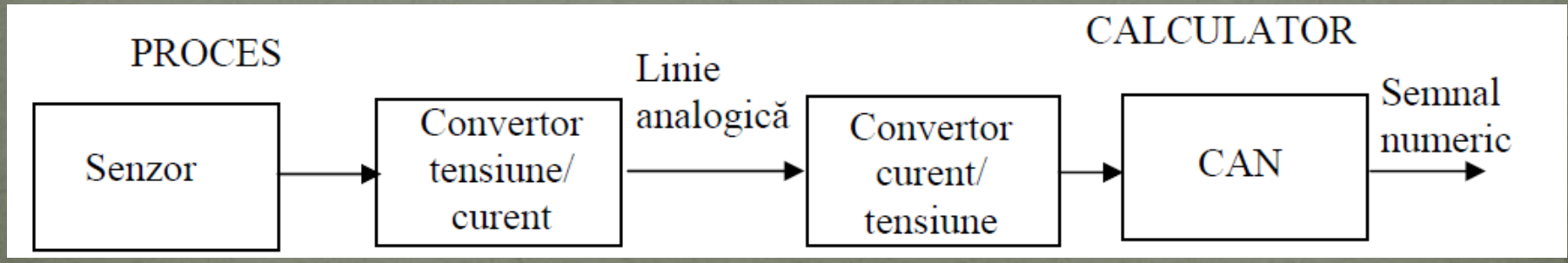




În sistemele industriale, senzorii sau elementele de execuție pot fi amplasate la distanțe relativ mari față de calculatoarele de proces. Prin urmare este necesară o linie de transmisie analogică capabilă să transmită semnalul analogic util fără ca acesta să fie alterat.

Pentru astfel de cazuri se evită folosirea semnalului de tensiune, care depinde puternic de rezistența liniei de transmisie, şi se preferă utilizarea surselor de curent, semnalul util fiind intensitatea curentului.

Deoarece CAN-urile utilizează semnal de tensiune, pot fi necesare două conversii: o conversie tensiune/curent dacă semnalul care provine de la senzor este de tip tensiune şi o conversie curent/tensiune la intrarea CAN



Izolare galvanica

Interfețele de proces asigură joncțiunea dintre SC şi procesul condus, şi prin urmare trebuie luate măsuri de siguranță astfel încât defectarea unei componente a procesului condus (inclusiv traductoare, elemente de execuție) să nu provoace defectarea sistemului de calcul sau viceversa, căderea sistemului de calcul să nu determine rezultate catastrofale pentru proces.

Printre tehnicile **de protecție** se pot enumera:

- izolarea galvanică între circuitele calculatorului şi cele ale procesului;
- protecții la supracurenți şi supratensiuni (circuite cu diode Zener, fuzibile etc.);
- implementarea unor unități de rezervă de interfață în cazul unor aplicații critice;
- detecția şi anticiparea defectelor: diagnoză, teste etc.;

Izolarea galvanică constă în eliminarea oricărei conexiuni electrice directe între circuitele calculatorului şi cele corespunzătoare procesului.

Izolarea galvanică se realizează de regulă cu transformatoare sau optocuploare şi oferă în primul rând posibilitatea de separare completă a alimentărilor celor două sisteme.

Separarea galvanică permite rezolvarea problemelor de împământare şi legare la nul precum şi problemele de alimentare a circuitelor de putere şi a celor de nivel mic.

Sistemul de întreruperi

Sistemul de intreruperi este acea parte a unui SC care permite detectia unor evenimente externe sau interne si declansarea unor actiuni pentru tratarea lor. Astfel de evenimente pot fi:

- receptia unui caracter pe un canal serial,
- golirea unui registru de transmisie,
- impuls generat de un contor de timp,
- tentativa de executie a unui cod de instructiune nepermis (inexistent sau protejat),
- terminarea unei anumite operatii de catre o interfata,
- eroare in timpul executiei unei operatii aritmetice (impartire ci zero) si multe altele.

Intreruperile permit calculatorului sa reactioneze rapid la aceste evenimente, sa se sincronizeze cu ele si sa le trateze in timp util.

Setul de intreruperi difera de la un procesor la alta.

Sistemul de întreruperi (Exemplu)

Metoda cea mai lentă dar şi cea mai uzuală de a transfera datele achiziționate în memoria sistemului este utilizarea **sistemului de întreruperi**.

Placa generează un semnal de tip **Interrupt Request** (IRQ) atunci când este achiziționat un eşantion sau mai multe eşantioane. Procesul de transfer al datelor prin sistemul de întreruperi se desfăşoară conform următoarelor etape generale:

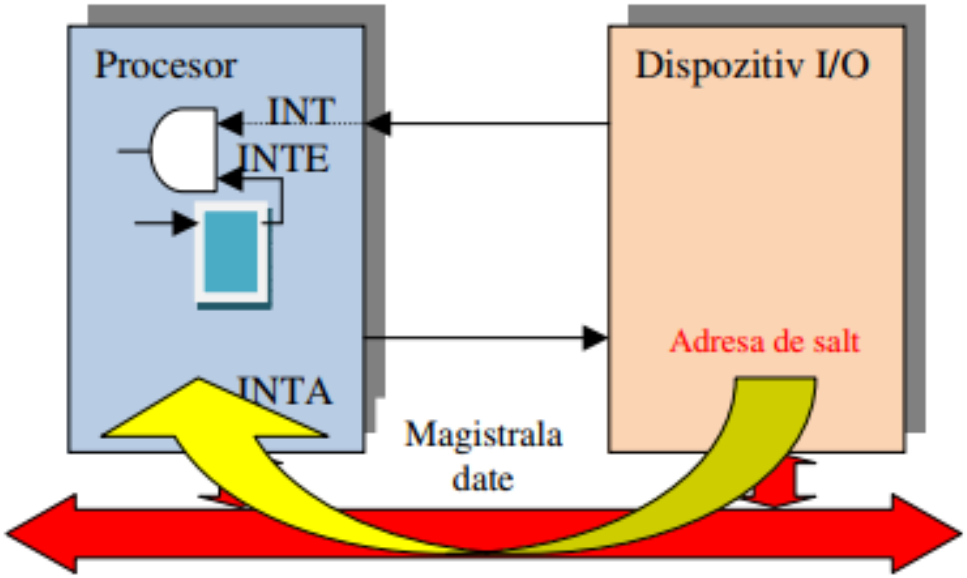
1. Placa utilizator instalată va lansa către microprocesor un semnal de tipul **Interrupt Request**, adică o cerere de întrerupere desemnată de un anumit cod; CERERE DE INTRERUPERE

2. Microprocesorul abandonează (în funcție de prioritatea întreruperii) temporar acțiunea în curs de desfăşurare şi transmite către placă un mesaj de recepție, numit **Interrupt Acknowledge**; ACCEPTARE INTRERUPERE

3. După aceasta, sistemul de operare va executa o rutină specială care salvează starea regiştrilor curenți ai microprocesorului şi care citeşte din tabela vectorilor de întrerupere adresa la care se află numărul canalului de întrerupere cerut.

4. În continuare se poate da controlul rutinei driver aflată la adresa respectivă, rutină care răspunde de activitatea dispozitivului care a emis semnalul IRQ. După rezolvarea acestei rutine, sistemul de operare va reface starea microprocesorului şi sistemul revine la starea şi procesul desfăşurat înainte de apariția semnalului IRQ.

Transferul de date



Limbajul ADA:

- Limbaj de programare standard pt. SCTR.
- Proiectat pornind de la Pascal în urma evaluării unui mare număr de limbaje de programare. Ada este limbajul obligatoriu impus de Pentagon pentru proiectele software ale Departamentului Apărării a SUA.
- Este denumit după Augusta Ada Byron, contesa de Lovelace, fiica lordului Byron și asistenta lui Charles Babbage, care este considerată primul programator din lume.

```
with Ada.Text_IO;
use Ada.Text_IO;

procedure MyProg is

 Name : String ( 1 .. 5);

begin

 Put_Line("What is your name?");
 Get(Name);

 if ( Name = "Apple") then

  Put_Line("Hallo " &  Name);
 else
  Put_Line("I dont know you!");
 end if;

end MyProg;
```

Limbajul ADA:

```
with Ada.Text_IO; use Ada.Text_IO;
procedure Hello is
begin
  Put_Line ("Hello, world!");
end Hello;
```

788 Kb

```
#include <iostream>

using namespace std;

int main()
{
    cout << "Hallo Welt!";
}
```

12.5 Kb

```
namespace bla
{
  void print()
  {
    // function bla::print()
  }
}

namespace blabla
{
  void print()
  {
    // function blabla::print()
  }
}
```

```
// namespaces
#include <iostream>
using namespace std;

namespace foo
{
  int value() { return 5; }
}

namespace bar
{
  const double pi = 3.1416;
  double value() { return 2*pi; }
}

int main () {
  cout << foo::value() << '\n';
  cout << bar::value() << '\n';
  cout << bar::pi << '\n';
  return 0;
}
```

```
5
6.2832
3.1416
```

Cele mai importante limbaje de programare:

- **1. ALGOL**

  Creat în 1958, Acesta a fost unul dintre primele încercări făcute pentru a crea un limbaj de programare, care ar putea fi utilizate pe mașini diferite.

  Astăzi, utilizarea Algol este minima, dar rădăcinile multor limbaje de programare majore sunt bazate pe el.

- **2. COBOL**

  COBOL-Common Business Oriented Language – a fost creat în 1959 și a găsit utilizarea în sistemele de afaceri, cum ar fi de asigurare, bancare etc. Chiar și astăzi computerele mainframe care rulează în departamentele guvernamentale, băncile, centralele nucleare folosesc COBOL și încă sunt produse și dezvoltate.

- **3. FORTRAN**

  Creat în 1957, Fortran - numit dupa Formula Traducator - ar putea fi considerat primul limbaj de programare de uz general, care a fost utilizat pentru calcule științifice grele. Astăzi, încă mai găsește utilizarea sa în rândul comunității științifice, fizicieni și ingineri.

- **4. ADA**
- **5. PL/I**
- Creat în 1964 și a introdus în 1969, PL / I reprezintă limbajul de programare number One.Inventat de către un comitet IBM, această limbă a pierdut in favoarea sa în 1970. Cu toate acestea, PL / I este încă folosit în IBM System / 360 mainframe-uri, datorită poziției dominante IBM în tehnologie.

Cele mai importante limbaje de programare:

- **6. PASCAL**

  Numit după bine cunoscutul matematician Blaise Pascal, acest limbaj a fost creat în 1968, care a atins punctul culminant în 1980. Una dintre cele mai populare descendenți ai Algol, Pascal este încă predată ca programarea orientată pe obiecte în multe locuri.

- **7. LISP**

  Lisp, creat pentru prima dată în 1958, standuri pentru lista de prelucrare. Este al doilea cel mai vechi limbaj de programare de nivel înalt, care își găsește utilizarea sa chiar și astăzi.

- **8. C**

  Proiectat de Dennis Ritchie, C este unul dintre limbile de programare cele mai influente din toate timpurile. Acest limbaj portabil este cunoscut pentru viteza care pot fi folosit pentru a accesa straturile de nivel inferior ale unui sistem. Mama a multe alte limbaje de programare Python -, PHP, Perl, MATLAB-C este de asemenea folosit pentru a scrie o bucată mare de sisteme de operare Unix, Windows și Linux.

- **9. C++**

  C ++ este un limbaj de programare orientat pe obiecte, care a fost creată între 1979 și 1983.

  Există tone de software variind de la jocuri, software birou, video playere care sunt scrise în C ++.

- **10. Java**

  Dezvoltat de James Gosling la Sun Microsystems, Java aparut pentru prima data acum 21 de ani în 1995. Prin urmare, este un limbaj orientat pe obiect și este mai simplu de utilizat decât una dintre influențatori sale C ++.

  Sistem mobil cel mai popular din lume de operare Android este scris în Java. De asemenea, Java continuă să fie baza a milioanelor de aplicatii web client-server si este cel mai popular limbaj de programare.

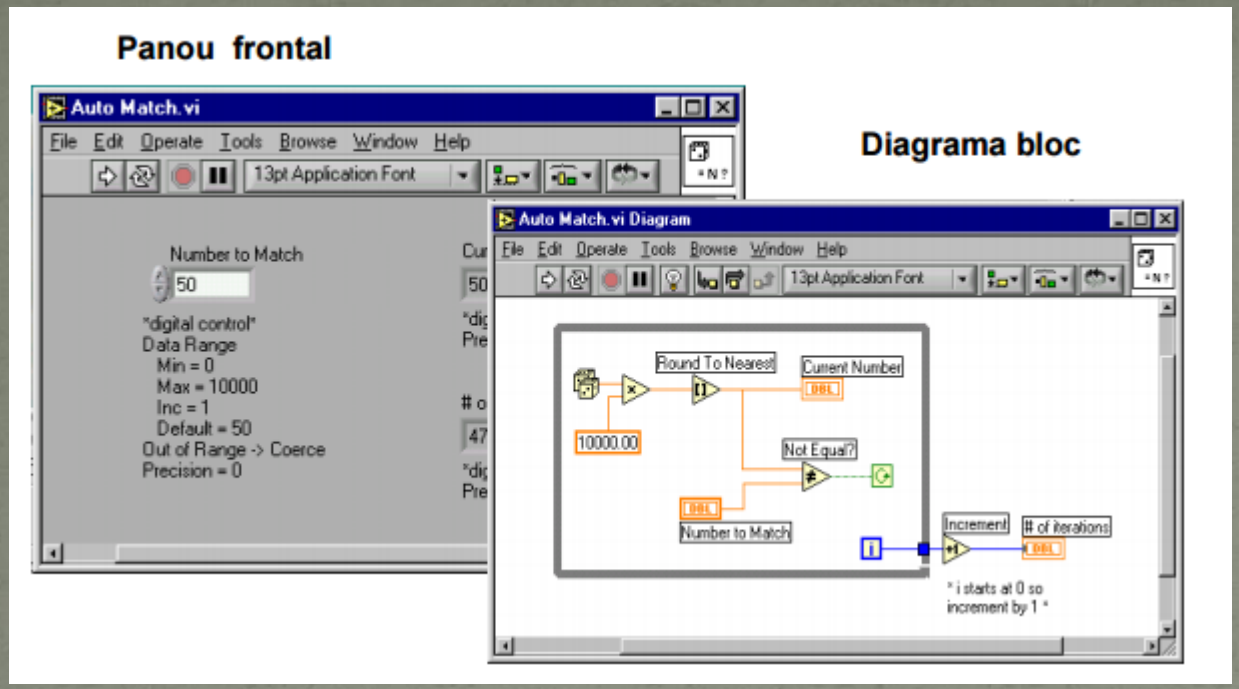
Cele mai importante limbaje de programare:

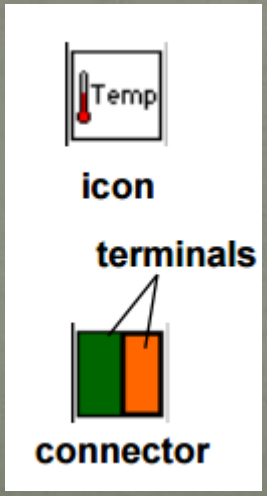
- **11. Javascript**
- Proiectat de Brendan Eich în 1995, JavaScript are un nivel înalt de programare dinamică. Această limbă este adesea menţionată ca „limbă de web”.
- JavaScript, HTML, CSS si sunt una dintre cele trei tehnologii de bază, care sunt folosite pentru a produce conţinut web. Este sprijinit şi folosit de către toate browserele web moderne.
- **12. PYTHON**
- Creat de programator olandez Guido van Rossum, Python a fost proiectat cu scopul de a scrie cod simplu şi uşor de citit.
  Python este una dintre cele mai populare limbaje de programare, care permite să se scrie cod în mai puţine linii. Python este utilizat de organizaţii, cum ar fi Google, NASA, Yahoo, şi CERN. Se găseşte de asemenea aplicaţii majore în comunitatea ştiinţifică, extragerea datelor şi câmpul AI.
- **13. SQL**
- SQL sau limbaj de programare structurată, a fost proiectat de Donald D. Chamberlin si Raymond F. Boyce.
  Iniţial bazată pe calculul şi algebra relaţională, acest limbaj de programare cu scop special este utilizat pe scară largă ca un standard pentru sistemele de management al bazelor de date relaţionale.
- **16. RUBY**
- Ruby – un scop dinamic, limbaj general foloseste programarea orientată pe obiect a fost dezvoltat în 1990 de către Yukihiro „Matz” Matsumoto. Influenţat de Perl, Ada, Eiffel, şi Lisp, Ruby vine cu un management automat al memoriei. Ruby împreună cu cadru web, este utilizat pentru a crea aplicaţii web cu uşurinţă.
- **17. PHP**
- PHP a fost proiectat de Rasmus Lerdorf în 1994, PHP iniţial a fost pagina pentru Personal Home. Hypertext Preprocessor. În acest scop limbaj de programare general, este una dintre cele mai frecvent utilizate tehnologii în dezvoltarea de server-side. Este folosit pentru a face site-uri web dinamice şi este coloana vertebrală a WordPress.

- LabVIEW este un mediu de programare utilizat mai ales pentru realizarea măsurătorilor şi monitorizarea unor procese automatizate. Pentru scrierea programelor în LabView, se utilizează limbajul grafic G, limbaj de programare de generatia a 5-a, mediul LabView continând mai multe biblioteci de functii predefinite pentru achizitia, prelucrarea, afişarea şi transmiterea datelor. Programele realizate în LabView se numesc instrumente virtuale (Visual Instruments - VIs), la baza acestora stând conceptele de modularizare şi ierarhie arborescentă. Când se proiectează şi se implementează un IV, trebuie să se tină cont de natura modulară a acestuia: să poată fi utilizat atât ca program principal cât şi ca subrutină în componenta unui alt IV.

Str**uctura unui program** IV:

- Panoul frontal;
- Diagrama bloc;
- Pictograma si conectorul





- Prin pictograma este reprezentat un VI in cadrul altei diagrame bloc

- Prin intermediul conectorilor se permite conectarea VI-ului initial ca "subVI" in cadrul unui nou VI

- Într-un sat fără electricitate – în care copiii nu au în case computere, tablete sau telefoane – locuiesc şapte fraţi.

  Primul citeşte o carte, al doilea a plecat să ducă vaca la câmp, al treilea joacă şah, al patrulea rezolvă un rebus dintr-un almanah vechi, al cincilea pune masa, iar al şaselea umflă o minge. Ce face al şaptelea dintre fraţi?