1. Definiţii ale S.I.

1) Un SI este un sistem ce integreaza HW şi SW şi proiectat pt o anumită funcţionalitate.
2) Un SI este un sistem de calcul cu scop predefinit inclus într – un dispozitiv pe care îl conduce.
3) Un SI este un sistem de calcul cu cerinţe specifice. SI execută sarcini predefinite.
4) Un SI este un sistem de procesare a informaţiei, parte a unui sistem mai mare sau a unui dispozitiv.
5) Un SI este o combinaţie de HW şi SW cu programare şi facilităţi fixe, proiectat pt un tip de aplicaţii.
6) Un SI este o combinaţie de HW, SW si, posibil, elemente mecanice sau alte elemente, proiectat pt a realiza o funcţie dedicată.
7) Def actualizată: SI constituie un subdomeniu ingineriei calculatoarelor, bazat pe circuite logice programabile de utilizator şi orientat pe aplicaţii de timp real

2. **Prezentaţi diferenţele faţă de calculatoarele de uz general**.

1) Interfaţa cu omul: led – uri, LCD – uri, minitastaturi;
2) Sisteme de intrare/ ieşire simple, fără periferie;
3) Pot include porturi de diagnosticare;
4) Pot include FPGA – uri, circuite analogice;

3. Domenii de aplicabilitate ale S.I.

- Ind automobilelor, Transporturi,Ind aeronautică, aerospaţială, Telecomunicaţii, Medicină, Automatizări ,Robotică,

4. Caracteristici şi cerinţe la S.I.

1) ***Conectare la mediul exterior***, - monitorizare (prin intermediul senzorilor)
   - comanda(prin intermediul actuatorilor=disp care converteşte val num în efecte fizice)
2) ***Funcţionare reactivă***:- un sistem reactiv este în continuă interacţiune cu mediul înconjurător
   - un sistem reactiv poate fi gândit ca fiind într – o anumită stare, aşteptând o intrare;
   - pt fiecare intrare execută una sau mai multe operaţii şi generează o ieşire
3) ***Funcţionare în timp real***: timpul devine un parametru al execuţiei operaţiilor;
   -există constrângeri de timp =>hard (produc efecte grave, uneori dezastruoase, la nerespectare) =>soft (produc efecte negative la nerespectare);
4) ***Eficienţa***: un SI trebuie să fie eficient; aceasta poate fi evaluată cu următoarele metrici:
   i. Consumul de energie: trebuie minimizat;
   ii. Dimensiunea codului: cod mare → memorie de program mare;
   iii. Execuţie implicând minim de circuite;
   iv. Greutate şi dimensiune mici;
   v. Cost redus.
5) ***Funcţionare în medii grele***: căldură excesivă, vibraţii, coroziune, fluctuaţii ale tensiunii de alimentare;
6) ***Dependabilitate***: foarte importantă datorită conexiunii cu mediul exterior;
   i. Fiabilitate: probabilitatea ca un sistem să nu se defecteze;
   ii. Mentenabilitate: probabilitatea ca o defecţiune să poată fi reparată într – un timp anumit;
   iii. Siguranţă: probabilitatea ca o defecţiune să nu cauzeze efecte catastrofale;
   iv. Disponibilitate: probabilitatea ca un sistem să fie disponibil.

5. Direcţii în studiul S.I.

- HW şi circuite de bază , LP, SO, Reţele de SI; Modelare, simulare şi validare; Aplicaţii.

6. **Legătura dintre S.I. şi "ubiquitous and pervasive computing"** *:*

- Formulate în perioada 2001 - 2003; SE BAZEAZA PE SI!!!
- Spre deosebire de modelul desktop, în UPC omul determină procesarea informaţiei de către mai multe sisteme, fără a fi necesar să fie conştient de acest lucru;
- Se bazează pe sist de calcul de dim. mici,(uneori nesesizate), cu anumite sarcini,care comunică între ele, cu un sist central.

7. Structura unui S.I.

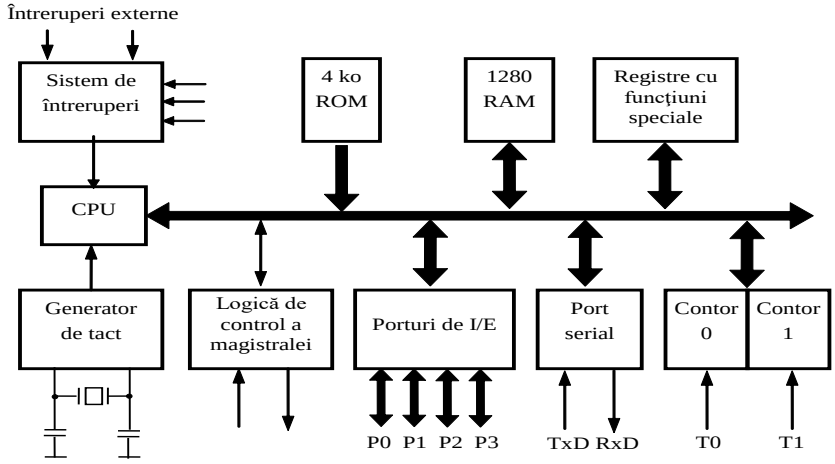
- ***Unitatea centrală***: pt a decide dacă un procesor este potrivit pt un SI trebuiesc luate în considerare câteva trăsături:
  - Numărul pinilor de I/ E
  - Numărul interfeţelor;
  - Cerinţele de memorie;

- Numărul liniilor de întrerupere;
- Facilităţi de timp – real
- Viteza
- Setul de instrucţiuni: RISC sau CISC;
- Instrumente de dezvoltare: decisive în testare şi depanare; costul lor trebuie luat în considerare.

- ***Memoria***
  - Situaţia ideală:când mem. internă,de date şi program, este suficientă, În caz contrar, este necesară mem externă;
  - Capacitatea de mem. gestionată de un microcontroler este mai mică decât cea gestionată de un microprocesor.
- ***Intrări/ ieşiri specifice:***
  - Intrările/ ieşirile SI sunt specifice:
    - Citesc informaţia de la senzori analogici sau digitali,
    - Primesc comenzi din exterior, fie pe linii digitale fie de la comutatoare, minitastaturi,
    - Afişează informaţia pe led – uri, LCD – uri, afişaje cu 7 segmente,
    - Comandă actuatori.
  - SI pot comunica pe linii seriale, cu sau fără fir, cu alte SI sau calculatoare de uz general;
  - Pot dispune de port serial pt programare în sistem;
  - Pot dispune de port pt depanare în sistem.
- ***Software de bază:***
  - Constă în sisteme de operare în timp real (RTOS), necesar pt SI complexe, de exemplu cele distribuite;
  - Exemple de RTOS: NetBSD, eCOS, Windows CE, OSEK etc.
  - SI simple, de exemplu majoritatea din aplicaţiile domestice, nu necesită software de bază.
- ***Software aplicativ:***
  - Implementează funcţionalitatea cerută;
  - Necesită limbajul de programare şi mediul de programare; mediul de programare rulează pe un PC;
  - Asigură operaţii ca: monitorizare, procesare, comandă şi control.

1) registre de uz general,
2) registre care corespund porturilor,
3) registre pt comanda modulelor periferice şi

8. **<u>Explicaţi noţiunea de RTOS (Real-Operating System)</u>**



1) Sunt proiectate pt sisteme care sunt caracterizate printr-un control foarte precis al timpului de răsp şi al performanţei;
2) În cadrul unui RTOS operaţiile pot fi prioritizate a.i. părţile cele mai critice ale aplicaţiei să poată lua controlul asupra procesorului exact în momentul în care este nevoie de acest lucru;TOATE celelalte operaţii sunt oprite temporar;
3) Comportament de tip determinist.
4) Sunt prevăzute cu mecanisme de prioritizare a taskurilor în funcţie de importanţa acestora ;
5) Sisteme de operare care nu sunt în timp real - pot fi injectate întârzieri aleatoare într-o aplicaţie;

9. **<u>Structura internă a 80C51.</u>**

10. **<u>Ce se intelege prin "Registre cu functiuni speciale"? 2 exemple la 8051.</u>**

-Microcontrolerul 80C51 conţine un grup de registre interne, cu funcţiuni speciale, SFR ("Special Function Registers");

-Registrele cu funcţiuni speciale sunt adresabile în mod direct, adresele lor se află în zona 80H – FFH;

-Există câteva tipuri de registre şi anume:

4) registre pt transferul datelor cu modulele periferic

11. Modul 0 al contorului 80C51.

1) TIMER 0 şi 1 sunt configurate ca numărătoare pe 13 biţi;
2) TL0 şi TL1 sunt numărătoare alcătuite din 5 ranguri(cele mai puţin semnificative)
3) TH0 şi TH1 sunt numărătoare pe 8 biţi;
4) indicatorii de depăşire TF0 şi TF;
5) Semnalul care apare la ieşirea de depăşire a lui TIMER 1 este sursă pt tactul serial.

12. Modurile de lucru ale interfetei seriale 8051

Modul 0…3

Modul 0 :

- Este modul numit şi I/ E extins în care se transferă date pe 8 biţi, sincronizate cu tactul microcontrolerului;
- Terminalul TxD este folosit doar pentru a genera tactul iar terminalul RxD este folosit pentru a transfera date în ambele sensuri;

Modul 1:

- Este un mod UART în care se transferă caractere pe 10 biţi: 1 bit de START, 8 biţi de date, primul fiind cel mai puţin semnificativ şi 1 bit de STOP;
- Rata de transfer este determinată de frecvenţa semnalului de la ieşirea de depăşire a lui TIMER 1, fTIM;

Modul 2:

- Mod UART cu 11 biţi/ caracter: 1 bit de START, 8 biţi de date, 1 bit programabil şi 1 bit de STOP;
- La transmisie, bitul al 9-lea este TB8 şi poate fi programat iar la recepţie, bitul al 9-lea este încărcat în RB8;

Modul 3:

- Mod UART cu 11 biţi/ caracter care diferă de modul 2 doar prin rata de transfer care aici depinde de f_{TIM}; astfel:
  - $R = 2^{SMOD} x\ f_{TIM}/ 32$ dacă TIMER 1 lucrează în modul 0 sau 1 şi
  - $R = 2^{SMOD} x\ f_{osc}/ 32 x 12\ x (256 - (TH1))$ dacă TIMER 1 lucrează în modul 2.

13. **Descrieţi funcţionarea sistemului multimicrocontroler bazat pe interfaţa serială a uc 8051.**

Cuprinde:

1) logica de control pt transmisie şi recepţie,
2) registrul de control SCON ("Serial Control Register")- contine biti de control pt a specifica:
3) registrele tampon SBUF ("Serial Buffer Register")- alc din 2 reg:
4) rangul SMOD;

14. Sistemul de întreruperi al 8051.

- 5 surse pt întreruperi:
  1) 2 întreruperi externe generate la intrările /INT0 şi /INT1
  2) 3 întreruperi interne: - 2 de la circuitele contoare/ temporizatoare
     - 1 una de la întreruperea serială.
- Cererile de întrerupere setează indicatori care sunt ranguri
  din registrele TCON şi SCON şi care generează întreruperile;

15. Alocarea priorităţilor la 8051.

Rolul registrului IP este acela de a aloca priorităţi;

i. fiecărei surse i se poate aloca ,independent, oricare nivel de prioritate (scazut sau ridicat)
ii. Fiecărei întreruperi îi corespunde un rang:
1- prioritate ridicată
0- prioritate scăzută;
iii. Dacă apar simultan 2 cereri de întrerupere
- de priorităţi diferite, va fi tratată cererea de prioritate mare;
- de aceeaşi prioritate, uc le va lua în considerare într-o ordine prestabilită (IE0, TF0, IE1, TF1, RI + TI);
iv. Rutina de tratare a unei întreruperi de prioritate scăzută poate fi întreruptă de o cerere de prioritate ridicată dar nu şi invers; rutina de tratare a unei întreruperi nu poate fi întreruptă de o cerere de aceeaşi prioritate.

16. Modurile de lucru cu consum redus ale 8051.

- 2 moduri: Idle şi Power – Down;

IDLE :Oscilatorul, interfaţa serială, contoarele/ temporizatoarele şi sistemul de întreruperi continuă să funcţioneze dar tactul nu mai ajunge la CPU. Întregul CPU îşi păstrează starea;

- Ieşirea: prin întrerupere validată sau RST;

Power-Down :

- Oscilatorul se opreşte, ca urmare starea întregului microcontroler rămâne

nemodificată (microcontrolerul "îngheaţă");
- Singura ieşire din acest mod este prin activarea intrării RST;
  În acest mod Vcc poate fi redus la 2 V

- Utile în aplicaţii în care consumul este un factor critic;
- Instalarea lor se face prin program, acţionând asupra a câte unui rang din registrul PCON ("Power Control Register")

17. Porturile 8051.

- uc 8051 dispune de 4 porturi bidirectionale, pe 8 biţi, notate cu P0, P1, P2 şi P3; Fiecare port are
-Porturile sunt de uz general, fiecare rang poate fi programat independent ca intrare sau ieşire

18. Ce se intelege prin PWM? Principii de funcţionare şi exemple de aplicaţii unde poate fi utilizat.

- Generează ieşiri modulate în durată;
- Facilitate utilă pt comanda motoarelor de curent continuu;
- Implementat cu un contor/ temporizator pe 32 biţi cu un divizor intern de 32 biţi;
- Facilitatea PWM poate fi dezactivată;
- Perioada şi durata pot fi programate;
- Facilitate de control individual al fronturilor crescator si scazator:utilă pt comanda motoarelor de curent continuu multi -fază
- Posibilitate de a genera: - Ieşire activă la 1 la fiecare început de ciclu;
  - Ieşire activă la 1 sau la 0 în timpul ciclului;
- Ex: telecomunicatii,controlul turatiei motoarelor de curent continuu, surse de tensiune in comutatie ,Controlul“sliding mode”

19. Prezentaţi 3 moduri de adresare în cadrul limbajului de asamblare a 80C51.

1) ***Adresarea directă*:** operandul este specificat printr-o adresă pe 8 biţi în cadrul instrucţiunii; exemplu: MOV A,09H;

2) ***Adresarea indirectă*:** în cadrul instrucţiunii se specifică un registru care conţine adresa operandului;

3) ***Adresarea de registru*:** este folosită pt adresarea unui operand aflat în unul din registrele R0 - R7; cod eficient;

4) ***Adresarea imediată*:** în cadrul instrucţiunii, după cod urmează o constantă; exemplu: ADD A,#64 sau ADD A,#64H;

20. Prezentati 3 exemple de instructiuni ale uc 8051 de tipuri diferite si descrieti semnificatiile lor.

ADD A,10H – *o instructiune aritmetica* cu adresare directa (operandul este specificat printr-o adresă pe 8 biţi)

SWAP A – *instr logica* de interschimbare a celor doua jumatati ale acumulatorului (ex: conversia unui nr din binar in BCD)

MOV B,#80 – *instr de transfer* cu memoria RAM interna, adresarea este implicita (operandul este in zecimal)

21. Caracteristici ale unui limba[illegible] putea fi folosit în S.I.

-un circuit de intrare
-un registru intern
-un etaj de iesire

- Eficienta; - posi[illegible] low-level la hardwa[illegible]
- Bine definit; - de uz general
- sa poata citi si scrie in locatii particulare de memorie(ex: pointeri)
- refolosirea codului (care a fost deja testat)

22. Precizaţi diferenţele de alocare în memorie a variabilelor comparativ cu constantele.

1) Identificatorii ce reprezinta valori de date variabile si necesita portiuni de memorie ce pot modificate in timpul executiei programului se numesc *variabile*
2) Valorile constante nu necesita spatiu de memorie ce poate fi modificat: o data ce valoarea unei constante a fost scrisa in memorie, aceasta nu se va modifica niciodata => compilatorul aloca un block din spatiul de memorie alocat programului, de obicei in ROM, pt fiecare dintre acesti identificatori

23. Avantaje si dezavantaje referitoare la folosirea definiţiilor simbolice (#define) şi a co[illegible] (const) în programarea S.I.

-Avantaj: sunt vizi
-Deiav: stocate in

1) Constantele declarate cu **const**
2) Constantele simbolice (#define)

-Av: nu ocupa spa
-recomandate in

24. Explicaţi impactul folosirii încapsulării de date utilizând *struct* şi *union* asupra performanţei SI.

- ***Incapsularea datelor - struct***
  1) Incapsuleaza date de tipuri diferite ce apartin aceluiasi obiect
  2) Sprijina gruparea datelor din program
  3) Accesul la un membru al structurii se face cu ajutorul operatorului “.” , si operatorului pointer “->”
- ***Incapsularea datelor – union***
  a. O uniune reprezinta suprapunerea elementelor de tipuri de date diferite in aceeasi locatie de

b. Dimensiunea uniunii este data de dimensiunea celui mai mare element
c. De obicei, tipul union este folosit in sisteme incorporate pt a crea un tampon pt retinerea diferitelor tipuri de date. Acest lucru poate economisi mem. refolosind acelasi block de 16 biti in fiecare functie ce necesita variabile temporare

25. Descrieţi folosirea facilităţii Bit Fields in limbajul C. Exemplu

1) Ofera posibilitatea accesarii unui singur bit sau a unui grup de biti din memoria neadresabila pe biti
2) Ordinea bitilor poate fi definita cu ajutorul cuvantului cheie *struct* .

Exemplu: struct TxIC
{ unsigned int glvl: 2;
unsigned int ilvl: 4;
unsigned int ie: 1;
unsigned int ir: 1;
unsigned int : 0;
} t7ic;
t7ic.ilvl = 12;

26. Programarea întreruperilor 80C51 (def întreruperi, etape de activare a unei întreruperi, rutina de tratare).

- O întrerupere este un semnal asincron care necesită o anumită atenţie; Opreşte CPU-ul microcontrolerului, acesta oprind execuţia taskului curent pt a trata întreruperea; după ce a fost tratată CPU-ul microcontrolerul reia execuţia taskului oprit

Activarea unei întreruperi:

1. Iniţializarea surselor de întreruperi, cum ar fi temporizatoarele (Timers), întreruperile externe sau UART;
2. Setarea biţilor din registrul IE care corespund surselor de întreruperi care doresc a fi activate

- Rutina de tratare a întreruperii (ISR) este acea rutină pe care microcontrolerul o apelează de fiecare dată când apare o întrerupere; poate apărea sub forma unei metode în C;

27. Programarea întreruperilor externe.

- Microcontrolerul 80C51 are două intrări pt întreruperi externe, fiind localizate la pinii P3.2 şi P3.3(INT0şi INT1)
- Întreruperile externe sunt activate fie pe front, fie pe nivel;
- dc o întrerupere este activată pe nivel,at. întreruperea apare când un semnal"low"este aplicat la intrarea întreruperii externe
- dc o întrerupere este activată pe front, atunci întreruperea apare când o tranziţie "high-to-low" este aplicată la intrarea întreruperii externe

- ***Paşii care trebuie realizaţi:***

1. Iniţializare a întreruperilor externe care vor fi folosite; selectarea dacă întreruperea externă este activată pe front sau pe nivel se realizează prin setarea la 1 sau 0 a IT0 pt întreruperea externă 0 sau IT1 pt întreruperea externă 1;
2. Activarea unei anumite întreruperi externe care va fi folosita prin setarea EX0 sau EX1 a registrului IE;

Activarea întreruperilor globale prin setarea EA din registrul IE.

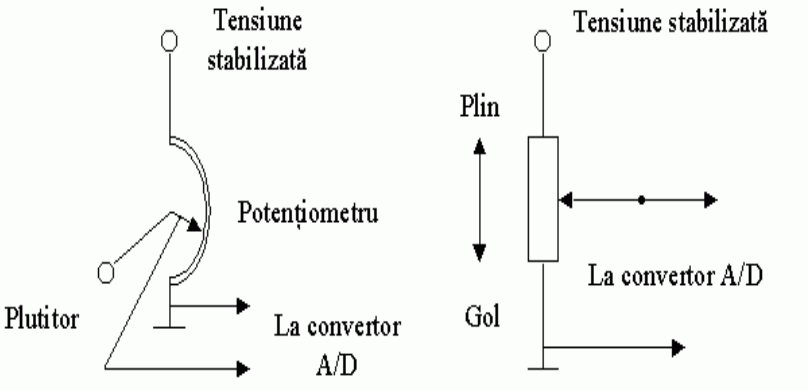
28. Sa se prezinte o metoda de generare a intarzierilor in domeniul microsecundelor, milisecundelor, secundelor. Explicatii si exemplu.

Folosirea instrucţiunii NOP (No OPeration)
NOP durează 6 cicluri de clock ale microcontrolerului

```
SETB P1.7        ; setează pin 7 port 1 la 1 logic]
ACALL DELAY
  NOP            ; menţine 1 logic pe pinul 7,
port 1
  NOP
  NOP
  NOP
CLR P1.7         ; pune pin 7 port 1 la 0 logic
```

DELAY: MOV R0, #88 <- valoarea in mili,microscunde

29. Exemple de senzori în automobile. Soluţii pt măsurarea cantităţii de combustibil.



- Senzori:
  - Cantitatea de combustibil;

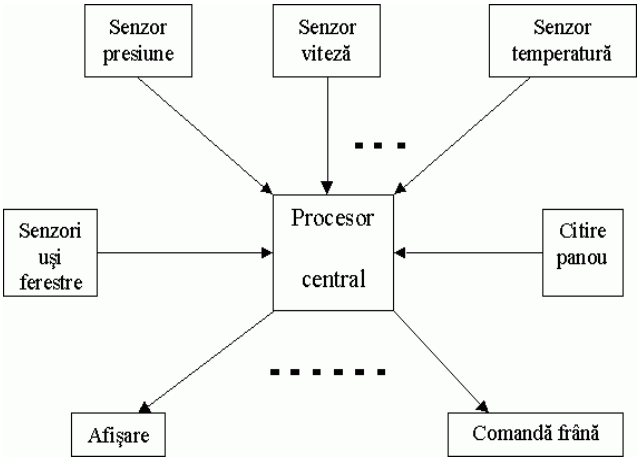
- Viteza de curgere a combustibilului;
- Presiunea uleiului;
- Temperatura lichidului de răcire;
- Temperatura de afară şi din habitaclu;
- De distanţă;
- Greutatea pasagerilor;
- Presiunea pneurilor;
- Umiditatea şi lumina externe;

• Exemple de senzori: - Pt cantitatea de combustibil:

30. **<u>Soluţie centralizată şi descentralizată de comunicare în automobile.</u>**

Centralizata:
Descentralizata

- Prioritizare a mesajelor; Garantare a timpilor de latenţă;



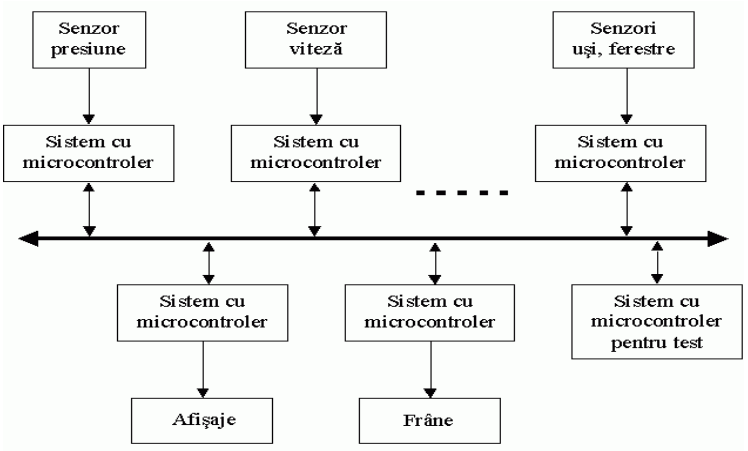
- Flexibilitate în configurare;
- Recepţie multiplă cu sincronizare de timp;
- Multimaster; Detectare de erori şi semnalizare a acestora;

2) ***Rata de transfer maximă***: 1 Mbit/sec la o distanţă max. de 40 m;

3) ***Datele sunt transmise în cadre***; un cadru include şi un identificator a cărui valoare numerică determină prioritatea cadrului; lungimea identificatorului stabileşte versiunea:

31. <u>Caracteristici generale ale magistralei CAN.</u>

1) Caracteristici:



- Retransmisie automată a mesajelor corupte atunci cînd magistrala devine liberă;
- Distincţie între erori temporare şi defecţiuni permanente ale nodurilor şi oprire autonomă a nodurilor defecte;
  - CAN 2.0A (CAN standard) cu identificator de 11 biţi;
  - CAN 2.0B (CAN extins) cu identificator de 29 biţi; nefolosit în automobile;

32. Arbitrarea pe magistrala CAN.

1) Nodul care câştigă arbitrarea continuă transmisia;
2) Nodul care pierde arbitrarea aşteaptă pînă la eliberarea magistralei;
3) Arbitrarea are loc la nivel de bit prin operaţia AND realizată de mediul fizic; există biţi dominanţi şi biţi recesivi; atunci cînd un nod care transmite un bit recesiv observă un bit dominant, va opri transmisia;
4) Fig. următoare arată arbitrarea:
5) Nodul transmiţător monitorizează linia;
6) Semnalul trebuie să se propage pînă la cel mai depărtat nod şi să revină la nodul transmiţător înainte deciziei ca urmare durata unui bit trebuie să fie cel puţin dublă faţă de întârziere; rezultă limitări: 1 Mbit/sec. la max. 40 m, 250 Kbit/sec. la max. 250 m etc.

33. Reprezentarea biţilor pe linie la magistrala CAN.

-Reprezentarea biţilor pe linie: prin metoda NRZ (Non Return to Zero):

1) 1 logic înseamnă o tranziţie a liniei;
2) 0 logic înseamnă lipsa tranziţiei;
3) După 5 biţi de valoare 0 este inserat un bit complementar, pt resincronizare;

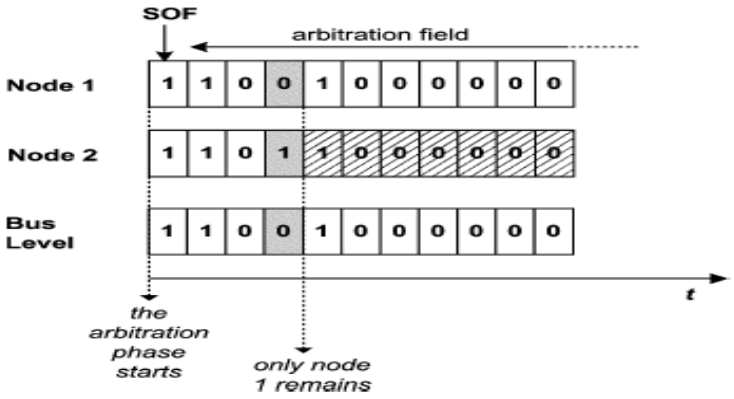
34. Tipuri de erori şi tratarea lor la magistrala CAN.

Tratarea erorilor: există 5 tipuri de erori:

1) ***Bit error***: orice transmiţător CAN asigură şi monitorizarea liniei; eroarea este detectată dacă bitul primit diferă de cel emis; excepţii sunt în timpul arbitrării şi al acceptării;
2) ***Stuff error***: atunci când sunt detectaţi 6 biţi cu acelaşi nivel;
3) ***CRC error***: atunci când câmpul CRC transmis diferă de cel calculat de receptor;
4) ***Form error***: atunci când un câmp cu structură fixă conţine biţi ilegali;
5) ***Acknowledgment error***: atunci când câmpul de acceptare nu are forma cerută;

35. Diferenţa între activarea în timp şi în funcţie de evenimente la nivelul S.I.

• ***Event-Driven***
– Activation of system services based on (external) events (i.e.,change of the state of an (external) entity)
• ***Time-Driven***



– Activation of system services based on the progression of (real-) time (i.e., at pre-defined points in time)

36. Avantaje şi dezavantaje la activarea în timp şi în funcţie de evenimente la nivelul S.I.

Event-Driven Systems

•Avantaje – Adaptiv
•Dezavantaje
– Lipsa predictibilitatii
– Testabilitate limitata

Time-Driven Systems

• Avantaje – Predictiv
• Dezajantaje
– Lipsa adaptabilitatii
– Mai greu de specificat si implementat tool-uri de asistenta

37. Caracteristici generale ale protocolului FlexRay.

1) determinist
2) rata de transfer de date 10 MBaud
3) time triggered bus access
4) free standard
5) protocol de toleranta la erori

38. Exemple de sisteme unde protocolul FlexRay poate fi folosit.

- Power train: sistemul de management al motorului, sistemul de transmisie
- Chassis area: damper control, driving dynamics, EWB (electronic wedge brake)
- Central gateway
- Future trends (application almost in entire vehicle)
- Serial production (extra equipment) since 2006

39. Motivaţii pt folosirea protocolului FlexRay. Comparaţie cu magistrala CAN.

- Multe functii si unitati de control
- close control circuits
- latime de banda mare

- comportament determinist
- magistrale diferite