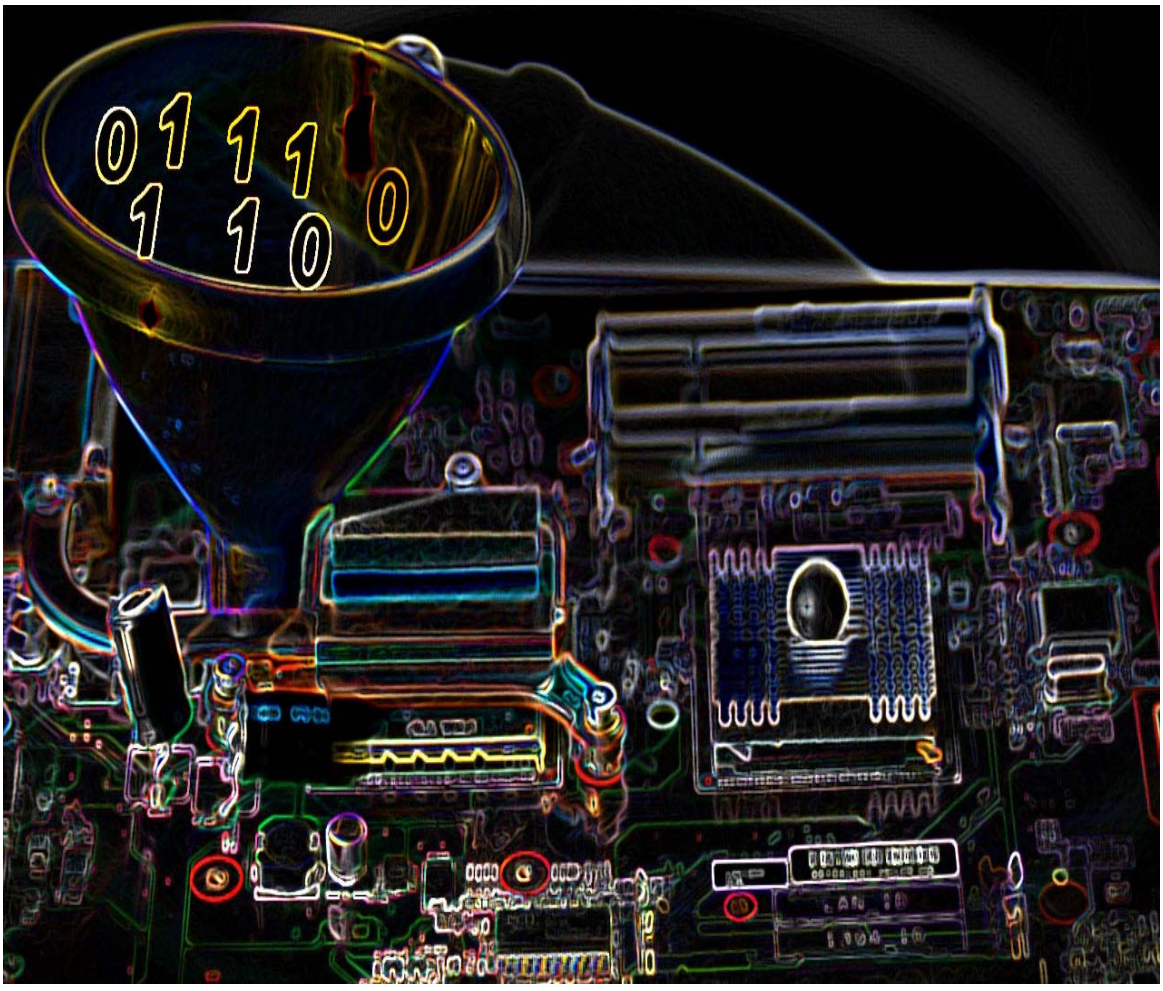
Petre OGRUȚAN

Interfațare şi protocoale la nivelul fizic şi nivelul legăturii de date



Editura Universității Transilvania din Braşov
2015

Coperta: fotografia unei plăci video, stilizată, autor Petre Ogruțan

Cuprins

Despre carte

Preocupările mele în activitatea ştiințifică şi cercetare s-au îndreptat către interfațare, un domeniu care se ocupă cu legăturile între diferite subsisteme ale unui sistem de calcul. Se pot remarca două caracteristici speciale ale acestui domeniu, varietatea şi dinamismul. Dacă în categoria sistemelor de calcul includem calculatoarele PC existente acum pe scară largă dar şi categoria sistemelor pe bază de calculator (embedded systems) obținem o varietate enormă a tipurilor de interfețe. Orice persoană are în jurul ei acum cel puțin trei sisteme de calcul, un telefon, un card bancar şi un notebook sau o tabletă. Dinamismul poate fi remarcat şi mai uşor, este suficient să ne gândim cum arătau acum 5 ani cele trei lucruri din jurul nostru enumerate mai înainte

Privesc cu nostalgie la prima carte pe care am scris-o în acest domeniu, cea de Interfețe şi periferice îm 1994. Vorbeam acolo de câteva magistrale care în acel moment aveau tendința de răspândire şi de care astăzi nu a mai auzit nimeni. A doua carte, Tehnici de interfațare a apărut în anul 2000 şi am alocat acolo un spațiu consistent interfeței paralele care acum nu mai este inclusă în niciun calculator PC. În 2003 am abordat problema interfețelor specializate în cartea cu acelaşi nume, iar ultima carte, apărută în 2011 este una de aplicații cu microcontrollere în care am abordat interfețele din microcontrollere şi cele care se pot conecta la microcontrollere, alături de descrierea unor aplicații.

Am învățat în decursul timpului că există câteva principii fundamentale de interfațare care nu se schimbă. O carte care să reziste în timp dar să nu plictisească trebuie să conțină atât părțile fundamentale cât şi tipurile semnificative de interfețe moderne. Această carte este o carte de sinteză a cercetărilor mele în domeniu şi este împărțită in două părți. Prima parte cuprinde principiile de interfațare, descrise într-un mod simplificat şi cu exemplificate. A doua parte cuprinde câteva exemple din cele mai cunoscute interfețe moderne. Descrierea interfețelor este legată de protocoalele de comunicații de date. Cartea se focalizează asupra nivelului fizic şi asupra protocoalelor care gestionează transferul de date la nivel fizic.

Cartea este destinată în primul rând cercetătorilor în domeniul sistemelor înglobate care pot găsi aici idei pentru a asigura conectivitatea cea mai potrivită unei aplicații. Cartea poate fi utilizată de specialişti în domenii conexe pentru a-şi face o imagine globală asupra interfațării. Şi nu în ultimul rând cartea poate fi folosită de studenții de la programele de studiu în domeniul electric pentru a face cunoştință cu un domeniu extrem de vast şi dinamic. Pentru aceştia cartea a fost rescrisă într-un format special. Materialul a fost împărțit în 14 module de învățare şi a fost completat cu 12 lucrări de laborator. Câteva mici filme ataşate fac acest material mai atractiv.

Petre Ogruțan, Braşov, decembrie 2014

1.Noțiuni introductive

1.1.Definiții. Istorie: prima comunicație cu protocol.

În domeniul calculatoarelor **interfața** este punctul de întâlnire a unității centrale cu dispozitivele periferice cu scopul transferului de date. Interfețele pot fi hardware sau software, iar în cazul interfețelor software situarea punctului de întâlnire este între două programe.

Interfețele pot fi bidirecționale în care datele circulă în ambele direcții (ex. interfața cu hard discul) şi unidirecționale în care datele circulă într-o singură direcție (ex. interfața cu mouse-ul). Interfețele pot fi punct la punct ceea ce înseamnă că pot fi conectate pe o linie doar un sistem gazdă şi un periferic (ex. interfața PS2 cu tastatura) sau multipunct în care la un sistem gazdă se pot lega mai multe periferice (ex. USB).

Pentru ca două sisteme de calcul să comunice între ele este nevoie ca formatul mesajului să fie bine stabilit ca să fie recunoscut de ambele sisteme. **Protocolul** conține un set de reguli care stabilesc structura mesajului şi asigură sincronizarea comunicației. De regulă protocoalele sunt standardizate de autorități în domeniul respectiv.

Echipamentul periferic este un dispozitiv conectat la un sistem de calcul gazdă care extinde funcționalitatea sistemului gazdă dar nu face parte din nucleul de calcul. Echipamentul periferic este deseori, dar nu întotdeauna dependent de sistemul gazdă.

Primul sistem de comunicații cu protocol a fost telegraful hidraulic al lui Aeneas, construit în secolul 4 BC, figura 1.1:

Fig. 1.1. Telegraful hidraulic al lui Aeneas (sursa http://en.wikipedia.org/wiki/Hydraulic_telegraph)

Recipientul se umple cu apă, peste care se aşează un plutitor care are fixată o tijă verticală. Pe tijă există gradaţii cu însemnări relevante pentru purtarea războiului (vreau cavalerie, artilerie, etc.). Dacă s-a dorit începerea unei transmisiuni, soldatul operator ridica torţa. Operatorul de le recepţie, observând cererea, indica că este pregătit pentru recepţie ridicând şi el torţa şi rămânea foarte atent la punctul de transmisie. Operatorul de la transmisie cobora torţa de semnalizare şi deschidea dopul de pe recipient, ceea ce făcea (sincron) şi cel de la recepţie. Apa începea să coboare identic în ambele recipiente cu viteză egală. La nivelul superior al vasului apăreau rând pe rând inscripţiile mesaj de pe tijă. În momentul în care apărea cea dorită a fi transmisă, operatorul ridica torţa, şi punea dopul. La fel făcea şi cel de la recepţie. În acest moment putea citi şi el mesajul. Pentru alt mesaj, recipientele trebuiau reumplute. Se pare că în istorie acesta a fost primul sistem sincron.

Într-o schemă bloc în care este figurat un sistem de calcul, figura 1.2. se pot observa interfeţele, ca puncte de întâlnire a datelor de intrare şi ieşire, care pot fi analogice sau digitale.

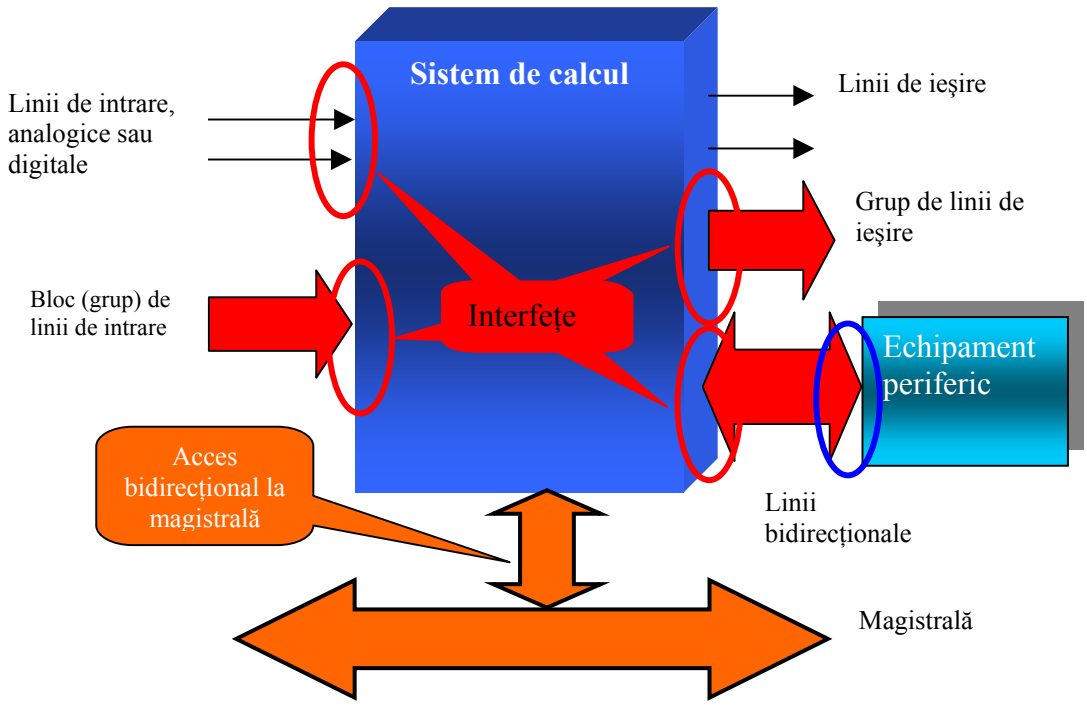


Fig. 1.2. Interfeţe marcate pe o schemă bloc

Mărimile de intrare şi de ieşire pot fi analogice sau digitale. Un semnal analogic are o variaţie continuă în timp. Exemple: temperatura, umiditatea, iluminarea etc. Un semnal

digital are o variație discretă în timp, el poate lua doar 2 valori. Exemplu: comanda dată de un întrerupător către un bec.

Sistemul de calcul procesează digital, prin urmare există interfețe de conversie a semnalelor analogice în semnale digitale numite convertoare analog digitale şi circuite de conversie inversă numite convertoare digital analogice. Exemplu: placa de sunet a unui calculator conține un convertor analog digital la intrarea de la microfon şi un convertor digital analogic la ieşirea de boxe. Aceste convertoare sunt importante pentru că în natură predomină mărimile analogice şi pentru ca acestea să fie măsurate şi prelucrate cu calculatorul digital este nevoie de conversia analog digitală şi digital analogică.

Dacă la portul cu linii bidirecționale a sistemului de calcul din figura 1.2 se conectează un echipament periferic EP acesta conține o interfață de conectare care trebuie să fie compatibilă cu cea a sistemului gazdă.

1.2.Interfețe paralele şi seriale

Un proces de comunicații de date necesită cel puțin 5 elemente, figura 1.3:

- Transmițător;
- Mesaj;
- Interfață binară (digitală);
- Canal de comunicație;
- Receptor.

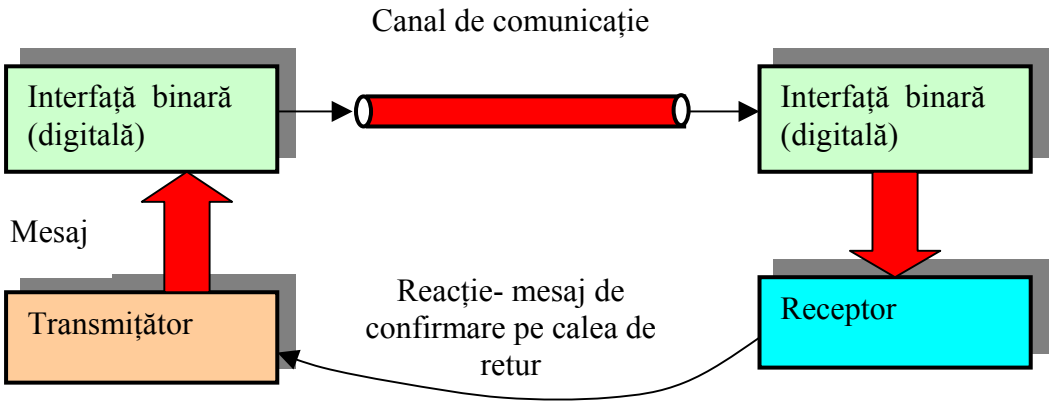


Figura 1.3. Procesul de comunicații de date

Pentru transmisia unui mesaj, fiecărui grup de biți trebuie să îi corespundă un caracter (literă, cifră, semn special). Cel mai cunoscut este codul ASCII (American Standard Code for Information Interchange).

La transferul paralel informaţia este transmisă pe mai multe linii (8, 16, 32, 64, 128 ...), cu un număr de biţi transmis la un impuls de tact egal cu numărul de linii, cuvintele fiind transmise succesiv, figura 1.4.

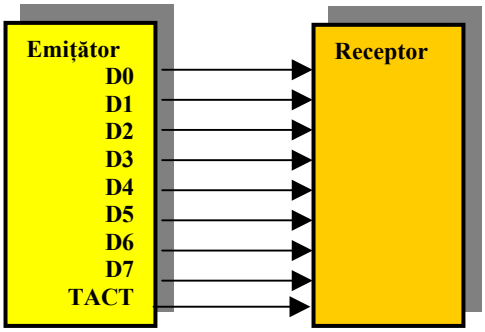


Figura 1.4. Schema bloc generală a interfeţei paralele pe 8 biţi

Diagrama de timp comprimată a transferului este dată în figura 1.5. Semnalul de tact are rolul de stabili momentul exact al citirii datelor, pentru a evita situaţiile în care datele sunt citite când încă nu sunt stabile pe linie. În anumite interfeţe acest semnal poate lipsi dar viteza de transfer este mai mică în acest caz.

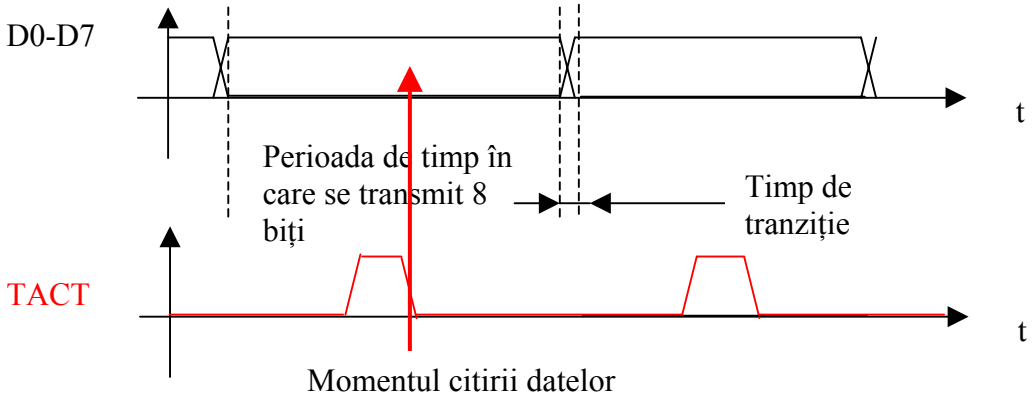


Figura 1.5. Diagrama de timp a transferului

În această carte vor fi prezentate transferurile paralele şi exemple de protocoale prin porturile paralele de I/O de uz general, dar şi alte câteva aplicaţii cum ar fi de exemplu circuitul de conversie USB- paralel FTDI. Magistralele şi interfeţele paralele sunt în scădere de piaţă, fiind înlocuite de cele seriale, de aceea şi ponderea lor în această carte este mai scăzută.

La transferul serial informația este transmisă bit după bit, pe mai puține fire (minimum 2 fire, dintre care unul de referință, masa electrică), figura 1.6. Pentru transmisia datelor acestea trebuie codificate. În carte vor vor fi descrise codificările NRZ, Manchester, 8B10B etc. Prețul mic al interfețelor seriale au dus la răspândirea acestor interfețe în majoritatea aplicațiilor.

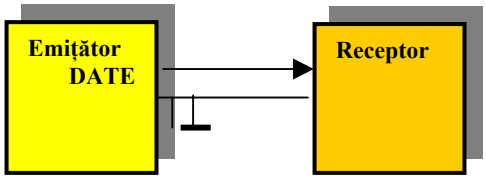


Figura 1.6. Schema bloc generală a interfeței seriale pe un fir
(linia de masă nu este considerată)

În perioada actuală numărul, viteza şi calitatea interfețelor seriale cresc, cucerind din ce în ce mai multe segmente de piață.

Tactul în transmisiile seriale are o importanță majoră. Dacă datele seriale vin la receptor pe un singur fir receptorul le poate citi / eşantiona cu o anumită frecvență şi să obțină datele recepționate ca în figura 1.7:

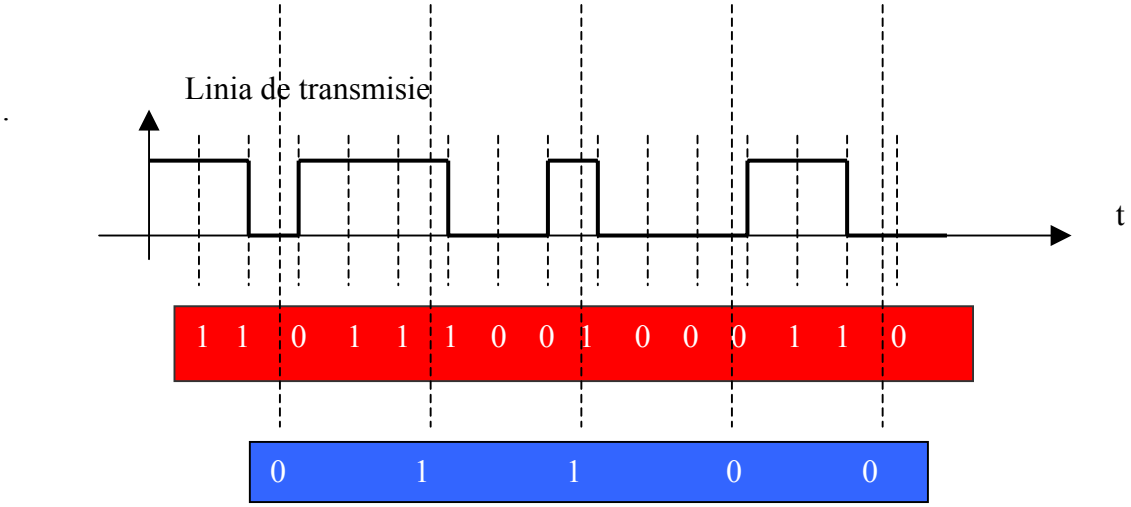


Figura 1.7. importanța eşantionării corecte

Dacă datele sunt eşantionate cu o anumită frecvență (pe fond roşu) se obține un anumit şir de date iar dacă datele sunt eşantionate cu altă frecvență (pe fond albastru) se obține un şir de date diferit. Acest lucru arată importanța ca datele să fie recepționate cu acelaşi

tact cu care au fost trimise. Este nevoie **ca tactul de transmisie să fie cunoscut de receptor.**

1.3. Verificarea corectitudinii datelor transmise cu bit de paritate

Teorema a 2-a a lui Shannon din teoria transmiterii informaţiei pentru canale afectate de perturbaţii afirmă că: ***Pentru o sursă de informaţie cu debitul de R bps (biţi pe secundă) şi un canal de capacitatea C bps, dacă R<C există un cod având cuvintele de lungime n astfel încât probabilitatea de eroare la decodare este oricât de mică.*** La un cuvânt binar de forma $A_2=a_7a_6a_5a_4a_3a_2a_1a_0$ i se ataşează la emiţătorul de informaţie un bit numit de paritate a_p care poate fi egal cu 1 (paritate pară) dacă numărul de biţi din cuvânt este par şi 0 dacă numărul de biţi din cuvânt este impar (sau invers, cu condiţia ca la emiţător şi la receptor convenţia să fie aceeaşi). Controlul de paritate poate detecta o eroare dar nu o poate corecta. Se poate solicita o repetare a mesajului prin linia de retur. Controlul de paritate se foloseşte la transmisii fără perturbaţii majore, de exemplu la debite mari de informaţii pe distanţe mici (lucrul procesorului cu memoria de exemplu) sau distanţe mari şi debite mici (transmisia serială RS232).

La receptor se generează un bit de paritate după acelaşi algoritm şi se compară cu bitul de paritate transmis. Dacă aceşti biţi sunt egali transmisia a fost corectă, dacă nu transmisia a fost eronată. Schema bloc a transferului cu verificarea bitului de paritate este dată în figura 1.8.

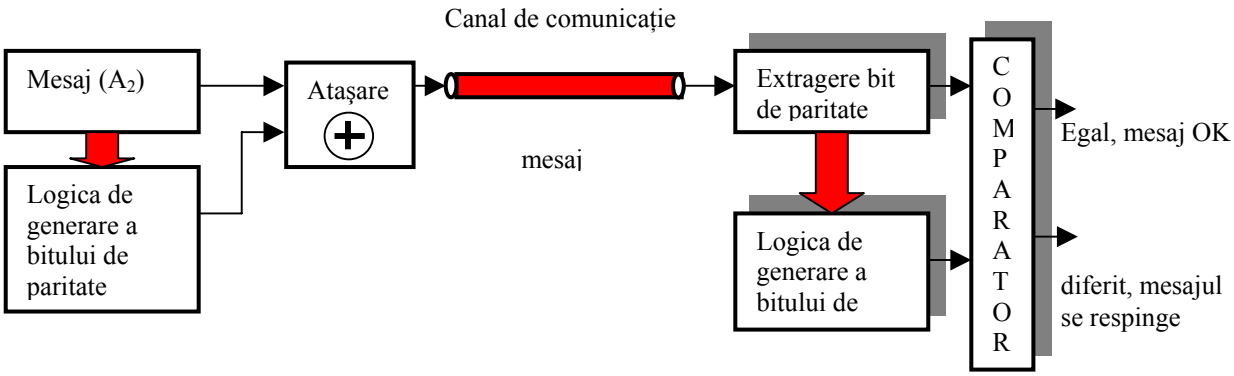


Figura 1.8. Schema bloc a transferului cu verificarea bitului de paritate

1.4.Rolul unui buffer în transferul de date

Viteza unui sistem de calcul depinde de 2 componente:

1. viteza cu care procesorul execută operaţiile;
2. viteza cu care circulă datele în sistem (viteza de I/O, viteza de acces la memorie).

Dacă procesorul este forţat să ramână inactiv perioade lungi de timp deoarece sistemul de I/O nu poate transfera datele nu îi sunt furnizate, sistemul este limitat I/O. Ideal este ca cele două viteze sa fie comparabile. În figura 1.9 se poate vedea o diagramă a fluxului de date.

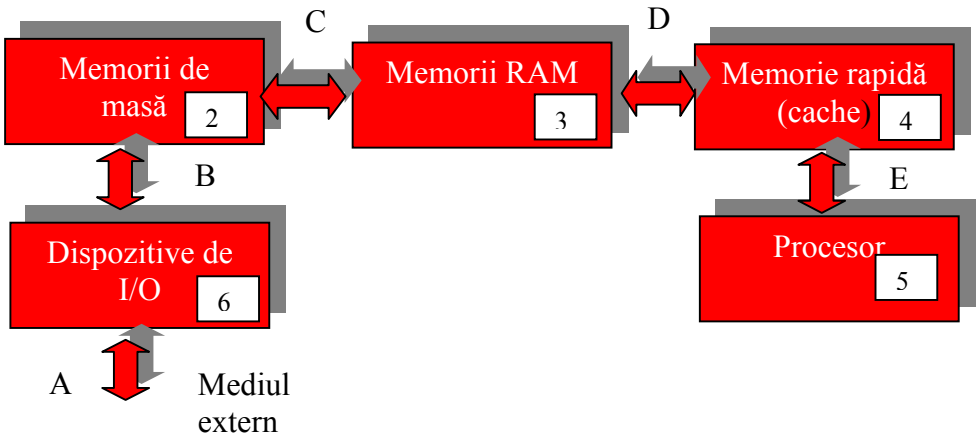


Figura 1.9. Fluxul de date într-un sistem de calcul

Pentru mărirea eficienţei se iau anumite măsuri, ca de exemplu:

1. Pentru a elibera procesorul de sarcina supravegherii transferului, pe calea C se poate face transfer DMA;
2. Transferul pe calea A este lent când schimbul de date cade în sarcina operatorului. Lucrul în întreruperi (cu tastatura, cu imprimanta, cu reţeaua) duce la optimizarea încărcării sistemului;
4. Eficienţa este maximă când capacitatea blocurilor 1,2,3 de a furniza date este egală cu capacitatea blocului 5 de a le prelucra.
5. Memoria rapidă conţine date şi secvenţe de program. Lucrul între blocurile 4 si 5 este foarte rapid. Dar există momente în care este nevoie de informaţia din memoria de bază (mai ales când apar instrucţiuni de salt). În aceste momente este nevoie să se schimbe întregul conţinut al memoriei rapide.

Un buffer de date situat între procesor şi echipamentul periferic are un rol important în mărirea eficienţei transferului de date, pentru că permite procesorului să fie liber de

sarcina de a transfera date perioade mai lungi de timp. Principiul transferului este dat în figura 1.10:

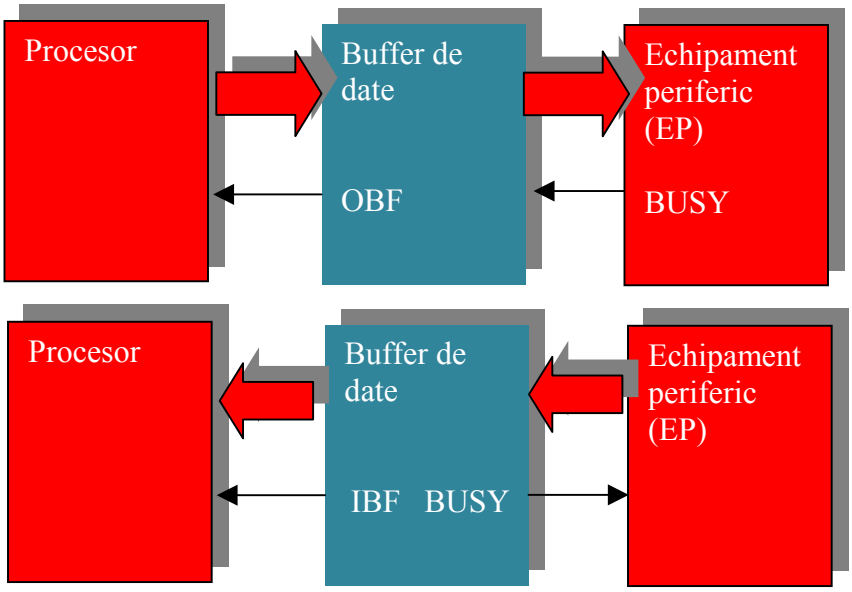


Figura 1.10. Rolul unui buffer de date în eficiența transferului

În cazul unui transfer de date de la procesor la EP (sus) datele se încarcă rapid în buffer (la viteza procesorului) până la umplerea bufferului, semnalizată cu semnalul OBF (Output Buffer Full). Datele sunt transferate lent la EP, primirea lor fiind acceptată dacă semnalul BUSY este inactiv. Invers, datele sunt încărcate în buffer până când acesta se umple, semnalizând cu BUSY. După umplerea lentă a bufferului procesorul este anunțat cu semnalul IBF (Input Buffer Full) şi preia rapid datele. În anumite aplicații bufferul trebuie să asigure un transfer de date cu viteza constantă cu EP, cum este de exemplu la scrierea pe medii optice.

Diversitatea de interfețe şi protocoale este foarte mare. Apar multe tipuri de interfețe noi dintre care unele se răspândesc iar altele dispar în timp. Cartea de Interfațare şi Protocoale se focalizează pe interfețe şi protocoale la nivel fizic, apropiat de microprocesorul sau microcontrollerul gazdă. Cartea conține două părți, prima conține noțiunile de bază ale conectării pe magistrală, pe un port paralel şi la o interfață asincronă simplă. A doua parte prezintă sumar câteva tipuri de interfețe şi protocoalele specifice.

2.Transferul de date

2.1. Clasificare

Transferul de date studiat în acest capitol este cel care are loc între un dispozitiv de I/O şi unitatea centrală. Transferul poate fi:

1. Programat;
2. Prin întreruperi;
3. Prin acces direct la memorie, DMA (Direct Memory Access).

Software-ul care controlează transferul de date se numeşte driver soft. Funcţiile acestui software sunt:

1. Iniţializarea dispozitivului de I/O;
2. Iniţierea transferului de date;
3. Transferul de date;
4. Terminarea transferului şi raportarea rezultatelor transferului.

Iniţializarea dispozitivului de I/O se face la punerea sub tensiune a dispozitivului sau la un RESET generat de unitatea central. Să ne aducem aminte cum face o imprimantă cu jet la pornire- un set de operaţii de iniţializare şi calibrare, un hard disc- testul de integritate a suportului, care se manifestă vizibil sau prin sunete caracteristice.

Comanda software a unui dispozitiv de intrare ieşire este realizată de:

1. Driver-ul software din calculatorul gazdă;
2. Programul executat de microprocesorul sau microcontrollerul dispozitivului de I/O, dacă acesta dispune de o unitate centrală.

Împărţirea sarcinilor între driver şi programul din dispozitivul de I/O depinde de fiecare aplicaţie, tendinţa actual fiind de preluare a cât mai multe sarcini de către dispozitivul de I/O pentru eliberarea timpului consumat de calculatorul gazdă.

În cazul unui sistem de calcul din familia x86 transferul poate fi înţeles pornind de la schema bloc din figura 2.1. Transferurile efectuate de procesor prin program pot fi:

A.Citire din memorie, MOV AL, [BX]

Pe magistrala de adrese este pusă adresa de memorie din registrul BX, care identifică o locaţie de memorie. Pe magistrala de comenzi este activat semnalul MEMR. Conţinutul locaţiei de memorie adresată este pus pe magistrala de date şi intră în registrul AL al microprocesorului (linii cu roşu).

B.Scriere în memorie, MOV [BX], AL

Pe magistrala de adrese este pusă adresa de memorie din registrul BX, care identifică o locație de memorie. Pe magistrala de comenzi este activat semnalul MEMW. Conținutul registrului AL este pus pe magistrala de date şi este salvat în locația adresată.

C.Citire de la un dispozitiv de I/O, IN AL, DX

Pe magistrala de adrese este pusă adresa dispozitivului din registrul DX, care identifică dispozitivul. Pe magistrala de comenzi este activat semnalul IOR. Conținutul portului de intrare este pus pe magistrala de date şi intră în registrul AL al microprocesorului (linii cu albastru).

D.Scriere la un dispozitiv de I/O, OUT DX, AL

Pe magistrala de adrese este pusă adresa dispozitivului din registrul DX, care identifică dispozitivul. Pe magistrala de comenzi este activat semnalul IOW. Conținutul registrului AL al microprocesorului este pus pe magistrala de date şi este trimis la portul de ieşire.

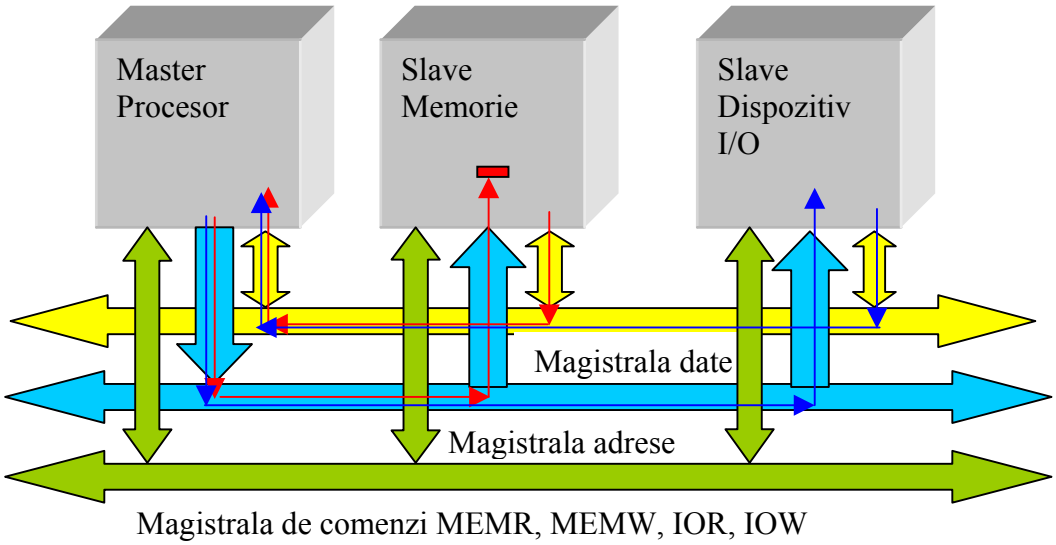


Figura 2.1. Schema bloc principială a unui sistem x86

2.2.Transferul programat

În acest mod de transfer toate operațiile de I/O sunt sub controlul programului. Transferul programat are două variante, direct şi prin interogare (polling).

a.Transferul programat direct

În transferul programat direct calculatorul gazdă citeşte sau scrie un port de intrare / ieşire în mod direct. De exemplu secvența de program scrisă în limbajul de asamblare x86:

```
MOV DX, adresa port          se încarcă în DX adresa portului de I/O
```

IN AL, DX	se citesc date de la portul adresat
MOV [BX], AL	se salvează datele în memorie, la adresa specificată de BX
MOV AL, date	se încarcă date în acumulator
OUT DX,AL	datele se trimit la portul cu adresa specificată în DX

Secvența anterioară citeşte date de la un port cu adresa specificată în registrul DX, le salvează în memorie şi scrie apoi date în acelaşi port.

b.Transferul programat prin interogare

Fiecărui dispozitiv de I/O i se atribuie un bit (fanion) care indică faptul că acesta este gata pentru transfer. Presupunem 8 dispozitive de I/O care au ataşate 8 porturi de date şi un port pentru citirea fanioanelor, structura fiind dată în figura 2.2.

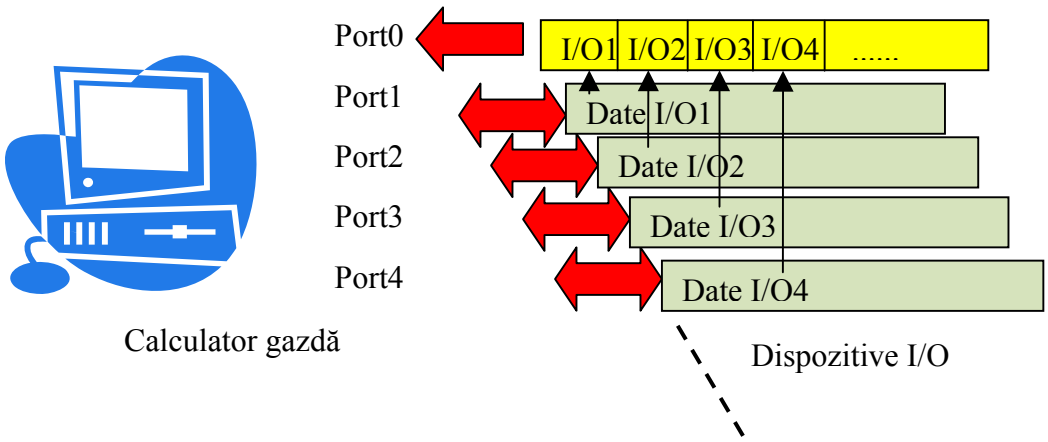


Figura 2.2. Transferul prin interogare

Fanioanele sunt interogate ciclic şi se serveşte acel periferic care solicită un transfer de date prin schimbarea stării fanionului. Dacă de exemplu cererea se face cu nivel logic 1 al fanionului, un exemplu de program este:

Start:	MOV DX, adresa Port0	
	IN AL,DX	se citesc fanioanele
	JN adresa1	la adresa1 este secvența de program de transfer cu I/O1
	RCL AL,01	deplasare AL la stânga
	JN adesa2	la adresa2 este secvența de program de transfer cu I/O2
	……	
	JMP start	

Se citeşte portul care conține fanioanele, apoi se verifică primul bit. Dacă cel mai semnificativ bit este 1 numărul binar din acumulator este negativ şi dispozitivul solicită un transfer. După testarea tuturor celor 8 fanioane bucla de citire şi testare este reluată. Secvențele de program de transfer pentru fiecare dispozitiv sunt secvențe de transfer programat direct, aşa cum au fost descrise anterior. Desigur că acest program este unul principial, care trebuie completat şi extins pentru a deveni un program funcțional.

Prioritatea dispozitivelor de I/O care sunt servite poate fi stabilită prin poziția fanionului corespunzător în octetul de fanioane. Dacă mai multe dispozitive solicită un transfer în acelaşi timp, servirea se face în ordinea poziției fanioanelor. Această ordine se poate modifica software.

Dacă un dispozitiv de I/O este mai rapid decât celelalte fanionul lui poate fi testat mai des în bucla de citire şi testare, caz în care citirea portului cu fanioane trebuie realizată înainte de testarea fiecărui fanion.

Prin software se poate **masca** un dispozitiv de I/O prin excluderea din bucla de testare a fanionului.

Pentru a evita o testare continuă, printr-o logică combinațională se poate semnaliza când un dispozitiv de I/O are nevoie de un transfer de date, printr-un SAU logic realizat hard între cele 8 fanioane şi care să fie citit de procesor, figura 2.3.

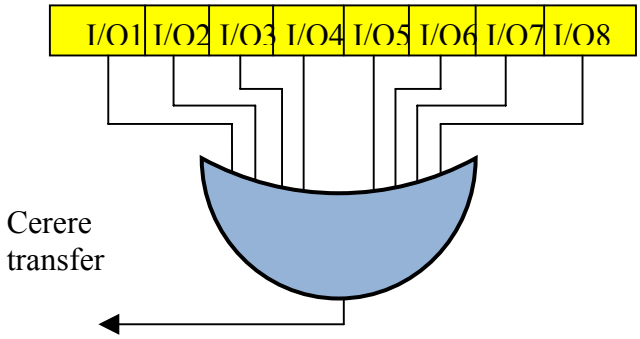


Figura 2.2. Semnalizarea unei cereri de transfer

Acest mod de lucru este foarte des folosit atunci când se combină transferul programat cu cel prin întreruperi. În acest caz semnalizarea unei cereri de transfer se face printr-o întrerupere, după care unitatea centrală identifică dispozitivul care are nevoie de transfer prin interogare.

Dacă există mai mult de 8 dispozitive de I/O care pot solicita un transfer sau sunt mai puține linii în portul de citire a fanioanelor, fanioanle pot fi codificate, procesorul citind astfel codul binar al dispozitivului care solicită un transfer.

Avantajele transferului programat sunt simplitatea hardware şi software şi faptul că există controlul strict al timpului în care se face un transfer de date.

Dezavantajele transferului programat sunt:

1. Sistemul se ocupă în cea mai mare parte a timpului cu dispozitivele de I/O, devenind dedicat acestei sarcini;

2.3.Transferul prin întreruperi

Transferul prin întreruperi are **avantajul** că gestionează mai bine timpul procesorului sau microcontrollerului. Procesorul rulează un program care este întrerupt la apariția unei cereri de transfer şi este reluat după ce transferul s-a terminat. Un alt **avantaj** este servirea mai rapidă pentru că programul principal se poate întrerupe în orice moment. Transferul efectiv durează la fel de mult ca şi prin transfer programat.

Familiile de procesoare au structuri diferite ale sistemului de întreruperi. Familia de procesoare x86 dispune de protocolul de dialog din figura 2.4.

Dispozitivul care cere o întrerupere va lansa un semnal INT. Procesorul verifică dacă întreruperile sunt validate cu semnalul INTE. Dacă întreruperile nu sunt validate cererea de întrerupere se ignoră. Dacă sunt validate atunci termină instrucțiunea în curs şi salvează în stivă conținutul registrului PC (Program Counter) şi a registrului de stare. Procesorul generează semnalul de acceptare a întreruperii INTA şi aşteaptă ca dispozitivul de I/O să pună pe magistrala de date adresa de salt la care se află rutina de servire a întreruperii.

Activarea şi dezactivarea sistemului de întreruperi se face prin program cu instrucțiunile STI şi CLI. Dezactivarea sistemului de întreruperi se face automat la RESET-ul sistemului sau după semnalul de acceptare a unei cereri de întrerupere INTA.

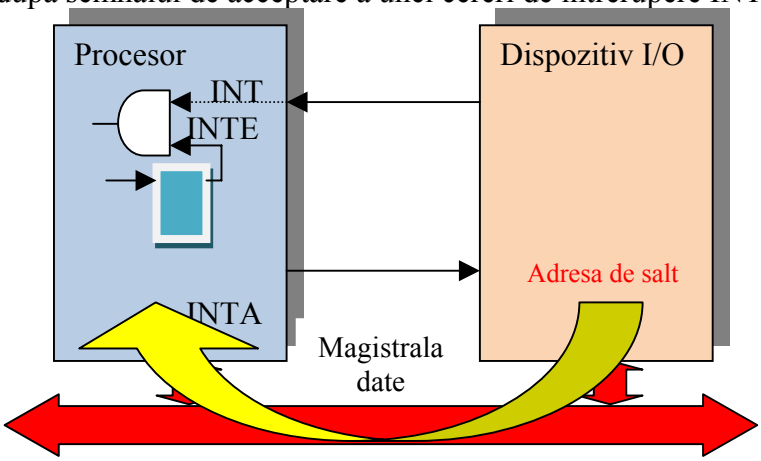


Figura 2.4. Protocolul de cerere şi acceptare a întreruperii

Pentru gestionarea mai multor întreruperi se poate folosi un controller specializat, aşa cum este de exemplu Intel 8259, care poate gestiona 8 nivele de întrerupere dar care permite şi cascadarea pentru mai mult de 8 nivele, figura 2.5.

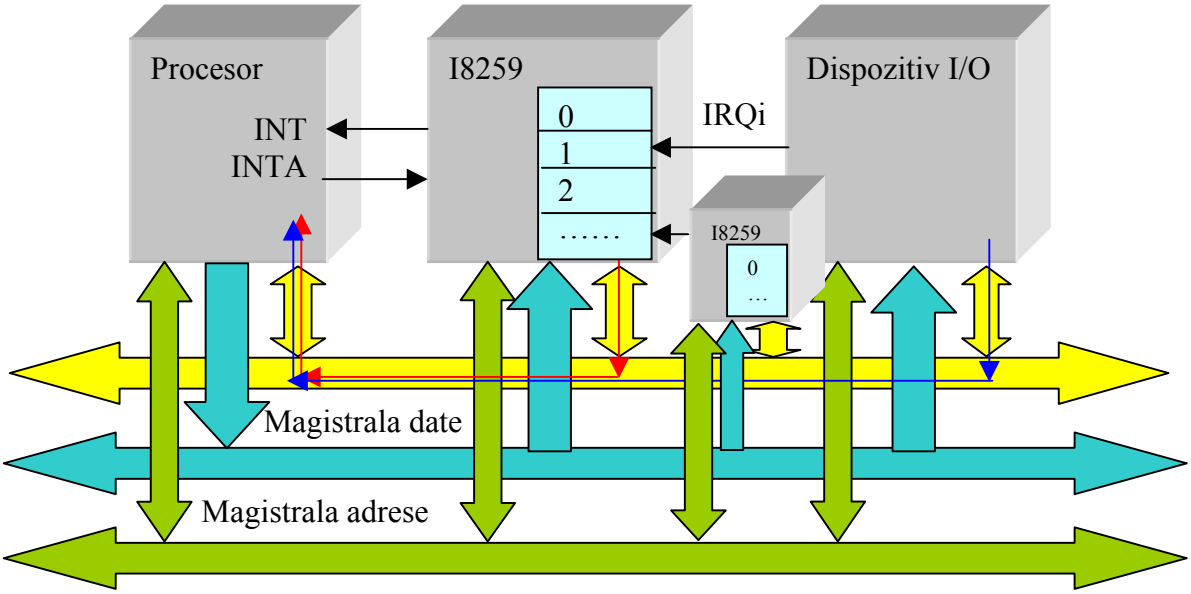


Figura 2.5. Sistemul de întreruperi gestionat de controllerul I8259

Dispozitivul de I/O cere o întrerupere IRQi. I8259 analizează cererea şi dacă nu este mascată şi nu este în curs de servire o întrerupere mai prioritară cere o întrerupere INT către procesor. Procesorul analizează cererea şi dacă o acceptă răspunde cu INTA către I8259. Controllerul I8259 pune pe magistrala de date adresa de salt la rutina de servire a întreruperii (traseul cu săgeți roşii). În rutina de servire a întreruperii datele se transferă între dispozitivul de I/O şi procesor (traseul cu săgeți albastre).

Punerea adresei de salt de către dispozitivul de I/O avea dezavantajul că limita portabilitatea dispozitivelor de I/O, acestea putând fi funcționale pe anumite sisteme şi nefuncționale pe altele. Utilizarea controllerului I8259 care are sarcina de a pune adresele de salt rezolvă această problemă.

Prioritatea cererilor de întrerupere se poate programa. Implicit IRQ0 este cel mai prioritar. Se poate programa **mascarea** unor cereri de întrerupere cu un registru de măşti. Controllerele de întrerupere pot fi cascadate, în acest caz numărul de linii de întrerupere poate crește. Dacă se cuplează două controllere cascadate, numărul liniilor de întrerupere disponibile devine 15.

Din punct de vedere software diferența între numărul de linii de adresă şi de date la un procesor implică dificultăți la transmisia unei adrese de salt pe magistrala de date. În general salturile se fac într-o zonă redusă de memorie, numită tabel al vectorilor de

întrerupere. În acest tabel nu se află câte o rutină de servire pentru fiecare nivel de întrerupere ci câte o instrucțiune de salt la o astfel de rutină, din cauza lipsei de spațiu.

Sarcinile controllerului I8259 şi datele cu care se programează sunt date în următorul tabel:

Sarcini I8259	Programare
Generare protocol de cerere şi acceptare întrerupere	-
Gestionare priorități	registru de definire a priorităților
Mascare selectivă	registru de măşti
Punere adrese de salt pe magistrala de date	adrese de salt

2.4.Transferul prin DMA

La transferul programat şi la cel prin întreruperi datele circulă între dispozitivul de I/O, acumulatorul procesorului şi memorie. Dacă dispozitivele de I/O sunt rapide, cu o viteză comparabilă cu cea a procesorului transferul se poate face prin DMA (Direct Memory Access). Acesta este cel mai rapid mod de transfer de date şi se poate face doar prin intermediul unui controller specializat, figura 2.6.

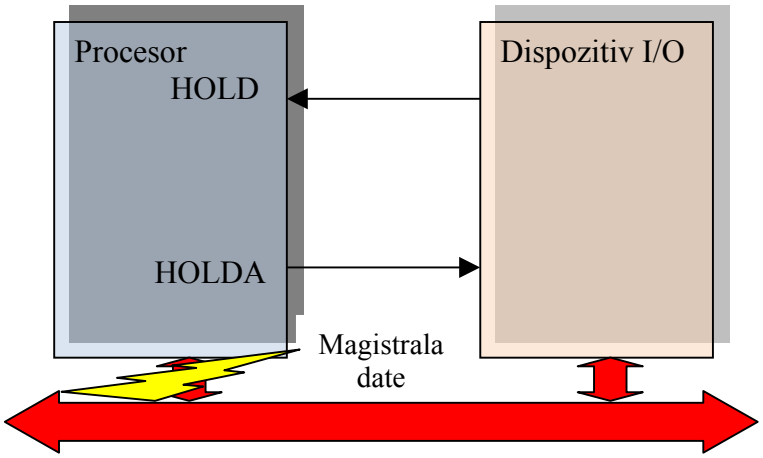


Figura 2.6. Protocolul de cerere şi acceptare de transfer DMA

Dispozitivul de intrare ieşire solicită un transfer DMA prin semnalul HOLD solicitând ca procesorul să-şi suspende activitatea prin trecerea magistralelor de date în înaltă impedanță. Când procesorul acceptă această cerere generează un semnal de HOLDA (HOLD Acknowledgement) şi trece magistralele în înaltă impedanță. Deoarece în acest caz nu mai există un dispozitiv master pe care să pună adrese şi semnale de comandă pe magistrală, acest mod de transfer are nevoie de un controller specializat care să preia rolul de master.

O schemă bloc a transferului DMA în familia x86 gestionat de controllerul Intel 8257 este dată în figura 2.7.

La primirea unei cereri de transfer DMA (numită DRQ) de la dispozitivul de I/O pe unul dintre cele 4 canale ale I8257, controllerul analizează cererea. Dacă cererea nu este mascată şi dacă nu este în curs de de execuție un transfer DMA, controllerul cere procesorului suspendarea activității cu semnalul HOLD. În momentul acceptării cererii procesorul activează semnalul HOLDA şi controllerul răspunde dispozitivului I/O că cererea a fost acceptată cu semnalul HOLDA.

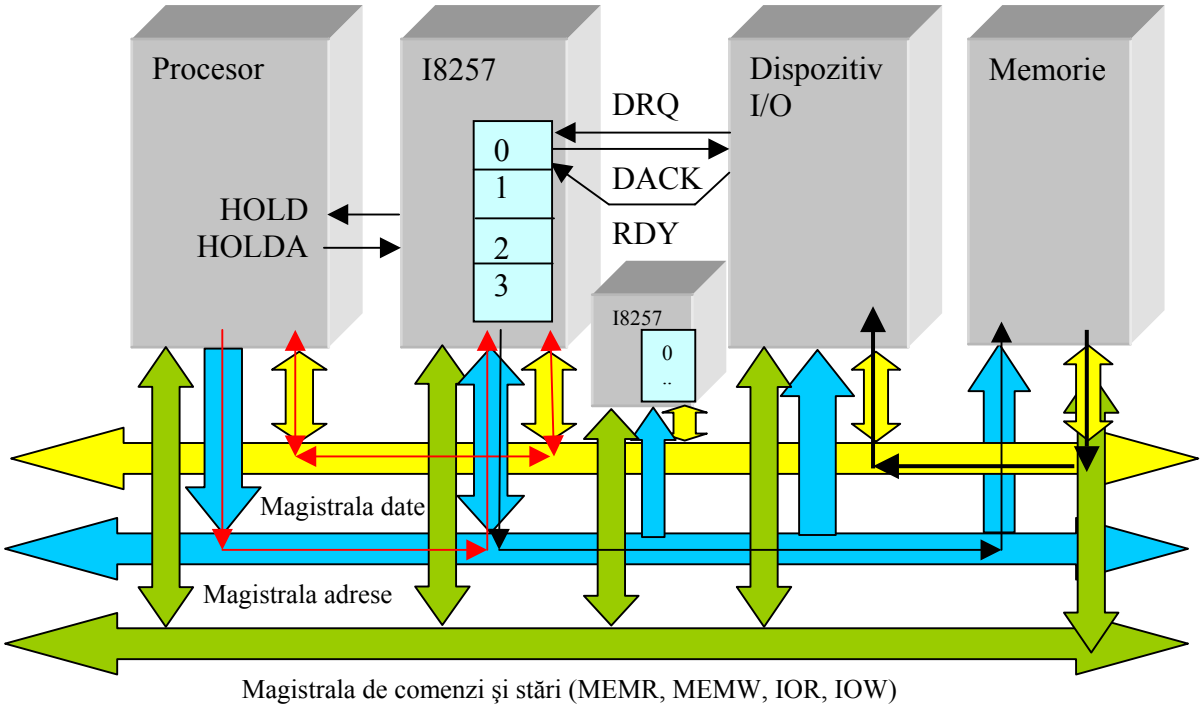


Figura 2.7. Sistemul DMA gestionat de controllerul I8257

După acceptarea cererii datele se transferă între dispozitivul de I/O şi memorie (linia neagră), fără a mai trece prin procesor. Controllerul DMA devine master pe magistrală şi pune adresele de memorie, precum şi semnalele de comandă MEMR, MEMW, IOR, IOW) (linii negre). Atunci când procesorul este master pe magistrală, controllerul DMA

este slave şi poate fi adresat în vederea programării lui. În acest caz procesorul pune adrese şi comenzi (linii roşii). Viteza dispozitivului de I/O este diferită de a memoriei (mai mică) şi sincronizarea datelor transferate se face cu semnalul RDY. Arbitrarea magistralelor se face doar la primul cuvânt, transferul având loc în continuare în salvă.

Prioritatea cererilor de DMA se poate programa. Implicit DRQ0 este cel mai prioritar. Se poate programa **mascarea** unor cereri de DMA cu un registru de măşti. Controllerele de DMA pot fi cascadate, în acest caz numărul de linii poate creşte. Dacă se cuplează două controllere cascadate, numărul liniilor disponibile devine 7.

Sarcinile controllerului I8257 şi datele cu care se programează sunt date în următorul tabel:

Sarcini I8257	Programare
Generare protocol de cerere şi acceptare DMA	-
Gestionare priorităţi	registru de definire a priorităţilor
Mascare selectivă	registru de măşti
Punere adrese pe magistrala de adrese şi comenzi pe magistrala de comenzi	Adresa de început a zonei de memorie şi numărul de cuvinte de transferat

2.5.Programe de comandă a transferului

În principiu, oricare ar fi metoda de transfer, programat, prin întreruperi sau prin DMA, programul care comandă transferul trebuie să aibă trei părţi:

1. Iniţializare transfer date
2. Transfer date
3. Terminarea transferului de date

Transferul de date programat

1.În segmental de iniţializare se scrie într-un contor lungimea zonei de memorie din care se citesc, sau în care se scriu date, fixează adresa de început şi porneşte lucrul cu dispozitivul de I/O.

2.În segmental de transfer de date începe scrierea în memorie sau citirea din memorie şi trimiterea la dispozitivul de I/O. Se decrementează contorul de cuvinte transferate, se incrementează adresa de memorie şi se verifică dacă citirea / scrierea s-a făcut corect (cu bit de paritate sau CRC). Dacă s-a detectat o eroare se reia de la ultimul cuvânt corect sau se abandonează transferul. Se verifică dacă s-au scris /citit toate datele printr-o metodă care ţine cont de natura datelor (un caracter special la codificarea ASCII etc.)

3.Dacă s-a întâlnit caracterul de terminare se comunică raportul transferului- număr de cuvinte transferate, existenţa unor erori. La sfârşit se comandă oprirea transferului.

Transferul prin întreruperi

1.În plus faţă de operaţiile de la transferul programat se iniţializează controllerul de întreruperi şi se validează întreruperile. La sfărşitul etapei porneşte transferul.

2.Începe când perifericul generează o cerere de întrerupere. Se salvează în stivă datele programului principal şi se trece la servirea întreruperii. Transferul are loc ca şi la transferul programat.

3.După comunicarea raportului se reiau din stivă datele programului întrerupt.

Transferul DMA

1.În plus faţă de operaţiile de la transferul programat se iniţializează controllerul DMA. La sfărşitul etapei porneşte transferul.

2.De transfer se ocupă controllerul DMA.

3.Printr-o cerere de întrerupere controllerul DMA anunţă că a terminat transferul. Se citeşte starea dispozitivului de I/O, dacă transferul a avut erori şi numărul de cuvinte transferate.

Un program de inventariere a resurselor unui PC (http://www.sysinfolab.com/) arată zona de memorie alocată unui dispozitiv de I/O, în figura 2.8. în cazul interfeţei SATA. Alte informaţii arată numărul liniei de întrerupere şi modul DMA de lucru cu hard discul prin interfaţa SATA.

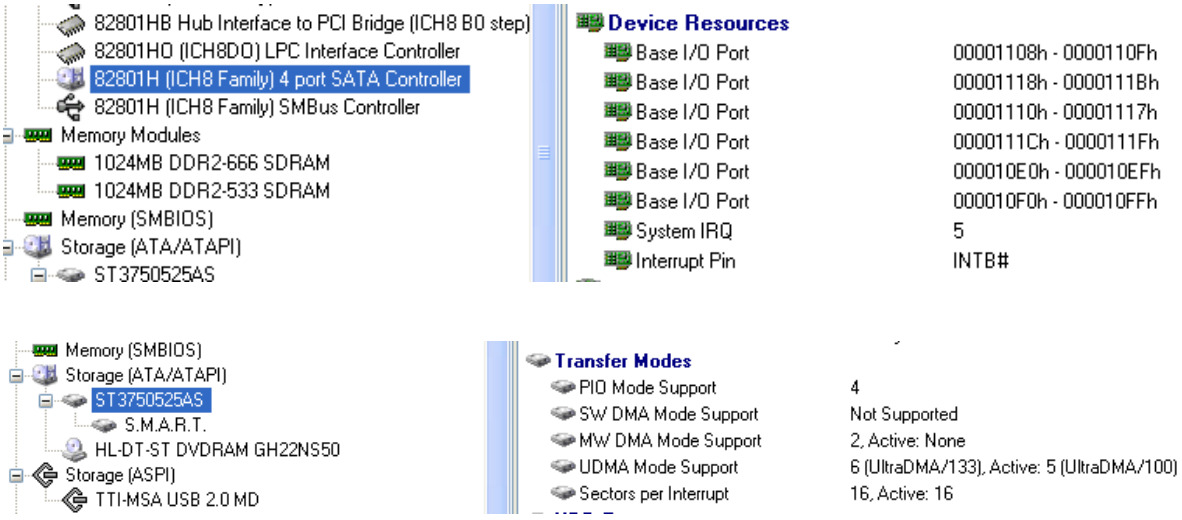


Figura 2.8.Configurarea unui hard disc cu interfaţă SATA şi alocarea zonei de memorie, nivele de întrerupere şi DMA

2.6.Sistemele de întreruperi şi DMA în microcontrollere

La un microcontroller sursele de întrerupere pot fi externe (semnale cuplate la pini), figura 2.9. sau por fi interne, de la interfețele integrate în microcontroller, cum sunt convertorul analog digital, timerul sau interfața serială. Cererile de întrerupere pot fi mascate cu un registru de măşti programat anterior în microcontroller de programul utilizatorului. O cerere nemascată va fi transmisă unității centrale care întrerupe programul curent şi face un salt la o adresă dintr-un tabel de adrese, modul particular de salt fiind specific diverselor familii de microcontrollere.

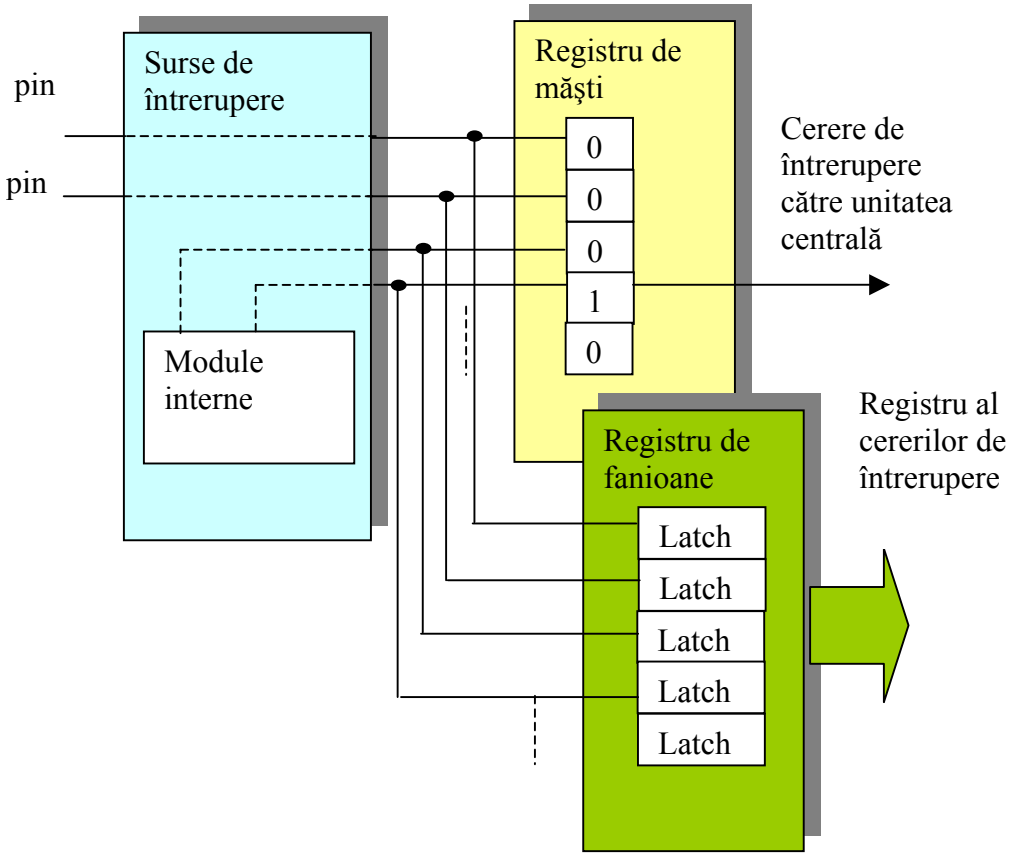


Figura 2.9. Sistemul de întreruperi la microcontrollere

Activarea unei întreruperi are ca efect şi poziționarea unui fanion în registrul de fanioane, permițând microcontrollerului să identifice sursa întreruperii. Registrul de fanioane este citit de programul utilizator prin transfer programat şi prin interogare se determină sursa întreruperii. Acesta este modul combinat între transferul programat prin interogare şi transferul prin întreruperi.

Este posibilă utilizarea exclusiv a transferului prin interogare prin interogarea în buclă a surselor de întrerupere în programul principal. În cazul unei aplicații de termometru

electronic citirea senzorului de temperatură se poate face prin interogare, periodic. Dacă cumva programul principal durează mai mult, de exemplu pentru că s-a detectat apăsarea unui buton prin care se modifică programarea termometrului, temperatura este citită mai târziu, ceea ce nu afectează utilizatorul. Dacă într-un alt exemplu se citeşte starea unui buton prin interogare, trebuie ca programul principal în nicio situație să nu depăşească timpul de apăsare scurtă a butonului de către utilizator, deci trebuie făcute unele calcule. Dacă se utilizează sistemul de întreruperi nu trebuie făcut niciun calcul.

Transferul DMA este implementat în microcontrollerele mai complexe, care trebuie să asigure viteze mari de transfer şi să prelucreze cantități mari de date. O descriere sumară a avantajelor transferului DMA la microcontrollerele din familia STM32, cu schema bloc din figura 2.10.

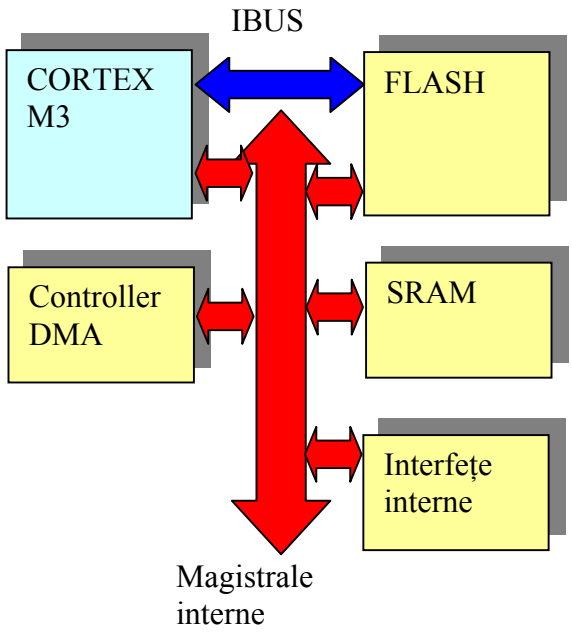


Figura 2.10. Schema bloc a familiei de microcontrollere STM32

Controllerul DMA are 7 canale, iar la modelele mai complexe există 2 controllere DMA cu 12 canale independente. Transferul DMA se poate executa între memorie şi dispozitivele de I/O dar şi între memorie şi memorie sau între două dispozitive de I/O.

O particularitate a transferului DMA este că dialogul de acces la magistrală (cerere şi acceptare de DMA, programarea controllerului DMA cu adresele de transfer) se face în acelaşi timp cu transferul propriu zis, deci la terminarea unui transfer DMA poate începe imediat altul.

Programarea unui canal DMA este simplă şi se face cu 4 regiştri, unul conține adrese de memorie, unul adrese pentru dispozitivul de I/O, unul numărul de cuvinte de transferat şi unul date de configurare.

O altă particularitate este faptul că în timpul transferului unitatea centrală poate lucra. Un ciclu de magistrală are 8 tacte, din care un transfer DMA ocupă doar 5. În timpul rămas liber unitatea centrală poate prelua instrucțiuni din memoria FLASH (transferul se face pe o magistrală specială IBUS) şi le poate executa.

În figura 2.11. se arată că un transfer de 32 de biți prin DMA (stânga) durează 214µs şi un transfer programat (dreapta) durează 544µs.

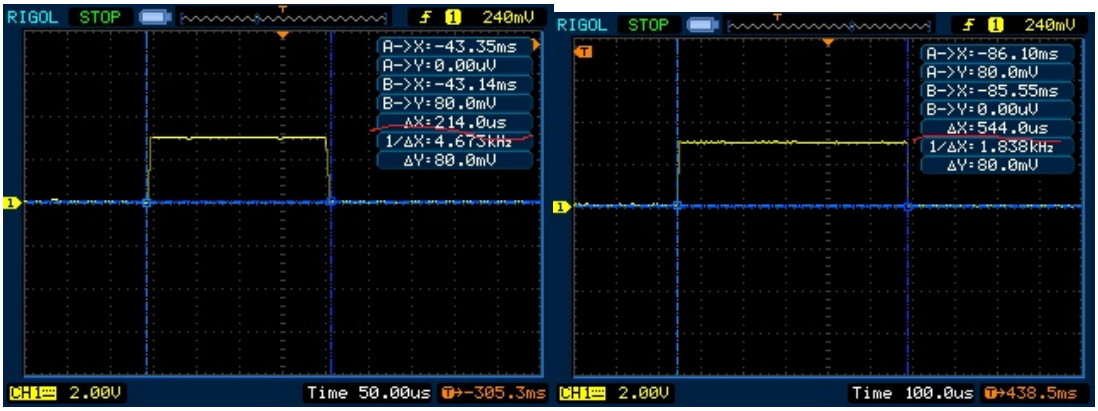


Figura 2.11. Comparație între transferul DMA şi programat (sursa: http://www.embedds.com/using-direct-memory-access-dma-in-stm23-projects/)

3.Magistrale

3.1. Introducere

Legătura între procesor şi EP (Echipamente Periferice) se realizează prin canale I/O (de intrare/ieşire) prin intermediul magistralei. Evoluția în timp a canalelor I/O este în acelaşi timp o evoluție a creşterii complexității şi performanțelor. Pot fi enumerate următoarele etape:

1. CPU controlează direct EP;
2. Este adăugat un modul I/O (o interfață serială sau paralelă, programabile). CPU comandă EP prin transfer programat (direct sau prin interogare);
3. Aceeaşi configurație ca la 2, dar transferul are loc prin întreruperi;
4. Modulul I/O are acces direct la memorie prin DMA. Modulul poate muta informația direct în memorie, accesul CPU fiind necesar doar la începutul şi sfârşitul transferului;
5. Modulul I/O foloseşte un microcalculator sau un microcontroller, cu instrucțiuni proprii. CPU programează procesorul I/O pentru un transfer, pe care acesta îl execută folosind instrucțiunile proprii. Când transferul se termină, procesorul I/O întrerupe CPU pentru a comunica rezultatele transferului;
6. Microcontrollerul are memorie locală. El poate controla astfel mai multe EP cu o intervenție minimă din partea CPU. Memoria locală poate fi folosită şi ca buffer de date, realizând astfel o rată de transfer mare.

Definiție: o magistrală este un subsistem cu funcția de comutator universal bidirecțional prin care se transferă date în interiorul unui sistem de calcul sau între sisteme de calcul. Schema bloc a unui sistem bazat pe magistrale este dată în figura 3.1:

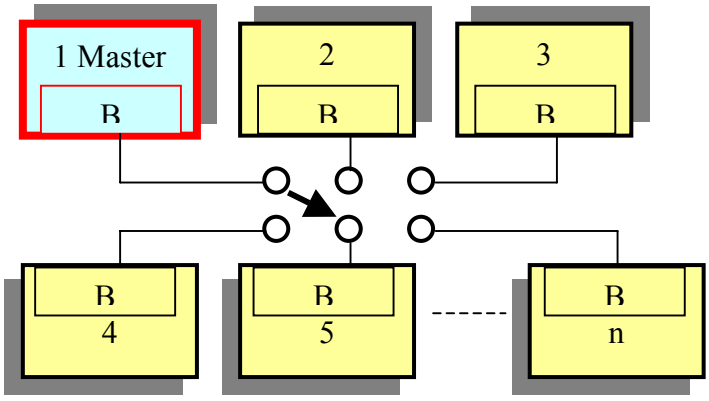


Figura 3.1. Structura unui sistem pe bază de magistrale

Sistemul prezentat în figura 3.1 este format din *n* subsisteme dintre care unul este Master la un moment dat iar celelalte Slave. Comutatorul este comandat de Master care stabileşte subsistemul Slave cu care face transfer de date şi stabileşte sensul transferului. Subsistemele pot fi explorate ciclic sau, mai eficient transferul poate fi realizat printr-o logică de priorităţi. Bufferele B optimizează traficul de date, aceste buffere fiind introduse în structura magistralei începând cu magistrala PCI.. Dacă fiecare subsistem poate transfera date cu viteza v_i şi viteza cea mai mare este:

v_{max}=max (v_i), atunci viteza magistralei V_M va fi:

V_M=k.v_{max} , adică mai mare (cu un coeficient k) decât viteza celui mai rapid subsistem

Magistralele pot fi de două tipuri:

1. Sincrone, la care există o linie de tact şi toate transferurile se fac într-un număr întreg de tacte, ciclul fiind numit ciclu de magistrală. Acest tip de magistrală este cel mai simplu şi ca urmare s-a răspândit şi la microcontrollere;
2. Asicrone, transferul poate dura oricât, este nevoie însă de un protocol de dialog. Pentru ca un transfer nereuşit să nu blocheze sistemul este nevoie de un mecanism de supraveghere a magistralei.

Prima magistrală a apărut în 1984 în structura calculatorului IBM PC şi s-a numit **ISA** (Industry Standard Architecture). Magistrala este sincronă, are 16 biţi de date, 24 de biţi de adresă, subansamblele conectate se configurează manual, viteza maximă este de 16Mbps. Specificaţiile tehnice ale magistralei au fost în domeniul public, ceea ce a determinat ca magistrala să fie un succes şi ca urmare compatibilele IBM PC să se răspândească în toată lumea.

Specificaţiile fiind libere, IBM nu a beneficiat financiar prea mult de pe urma acestei magistrale, aşa încât a lansat în 1987 o nouă magistrală, superioară, pentru calculatoarele PS2 numită **MCA** (Microchannel). Caracteristicile ei: 32 de biţi de adresă, 32 de biţi de date, configurare automată, viteza maximă 32Mbps, cu posibilitatea de a lucra multiprocesor. Pentru a construi dispozitive MCA era nevoie de cumpărarea licenţei de la IBM şi au fost puţine firme care să facă acest lucru, ca urmare calculatoarele cu MCA nu s-au răspândit.

Ca reacţie la MCA, un consorţiu de firme (Compaq, Epson, Hewlett Packard, NEC, Olivetti şi Zenith) a lansat magistrala **EISA** (Extended ISA) în 1988. Magistrala EISA are performanţe cel puţin atât de bune ca şi MCA: 32 de biţi de adresă, 32 de biţi de date, configurare automată sau manuală, viteza maximă 120Mbps şi asigura compatibilitate cu plăcile ISA. Specificaţiile au fost în domeniul public şi probabil magistrala ar fi avut succes.

EISA nu s-a răspândit pentru că INTEL a lansat în 1990 magistrala **PCI** (Peripheral Component Interconnect) cu specificaţii în domeniul public, cu un concept nou al arhitecturii. PCI poate lucra cu 32 sau 64 de biţi de date la o viteză de maximum 2,112Gbps. Conceptul de ierarhizare a magistralei după viteză permite compatibilitatea cu magistrala ISA. În ultimul timp plachetele ISA au dispărut dar PCI a păstrat intern o magistrală de viteză mică numită LPC (Low Pin Count), de fapt o magistrală ISA cu semnale multiplexate pe aceleaşi linii pentru economia de pini.

3.2.Magistrale ierarhizate

Magistrala PCI are o arhitectură care permite existenţa a două magistrale pentru I/O, una de viteză mare şi una de viteză mică, figura 3.2.. La magistrala de viteză mare se conectează dispozitivele rapide iar la cea lentă se cuplează canalele care necesită o viteză mică. Culorile sunt sugestive, cele mai calde sugerând o viteză mai mare.

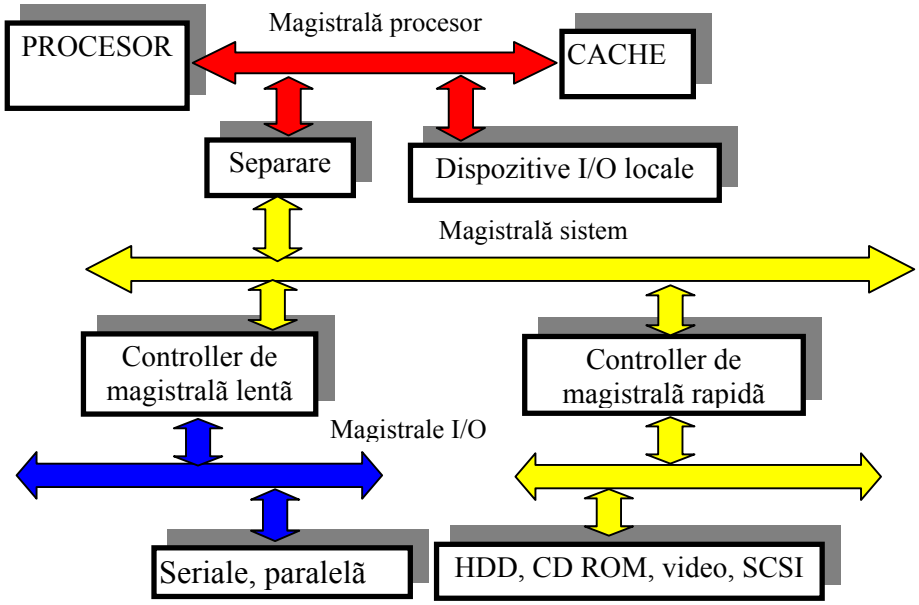


Figura 3.2. Principiul magistralei ierarhizate

Dacă se analizează cazul concret al sistemelor PC lucrurile devin mai clare. Structura ierarhizată a magistralelor este dată în figura 3.3. Se poate vedea structura pe 3 nivele, cel mai de sus fiind constituit pe lângă magistrala procesorului care are viteza cea mai mare, tactul fiind FSB (Front Side Bus). Pe această magistrală sunt conectate controllerul video integrat, controllerul de memorie DRAM şi controllerul magistralei PCI. Pe al doilea nivel este situată magistrala PCI cu un tact de 33MHz, la care sunt conectate dispozitivele de I/O rapide, adică canale USB, interfaţa SATA, interfaţa Ethernet

10/100Mbps. AL treilea nivel este reprezentat de magistrala LPC (standard ISA) la care sunt conectate cele mai lente dispozitive de I/O, adică canalele PS2 pentru tastatură şi mouse. Probabil că în timp se va renunța la nivelul magistralei LPC, existând tendința ca dispozitivele de I/O să devină mai rapide şi migrarea către USB. În general creşterea vitezei interfețelor face ca acestea să urce pe diagrama magistralelor. Un exemplu clar este interfața Ethernet care la viteza de 10/100Mbps este conectată la magistrala PCI dar la viteza de 1Gbps este conectată la magistrala procesor.

Funcționarea sistemului ierarhizat de magistrale poate fi exemplificat prin modul în care se execută instrucțiunea MOV DX,AL la portul de ieşire 0378H, portul paralel. Instrucțiunea este analizată de controllerele de magistrală ierarhizate. Controllerul PCI are o tabelă de adrese de I/O şi verifică dacă adresa din instrucțiune este alocată unui periferic rapid. În acest caz nu este alocată, aşa că instrucțiunea este executată pe magistrala LPC. O instrucțiune de citire/ scriere cu hard discul ar fi fost executată pe magistrala PCI.

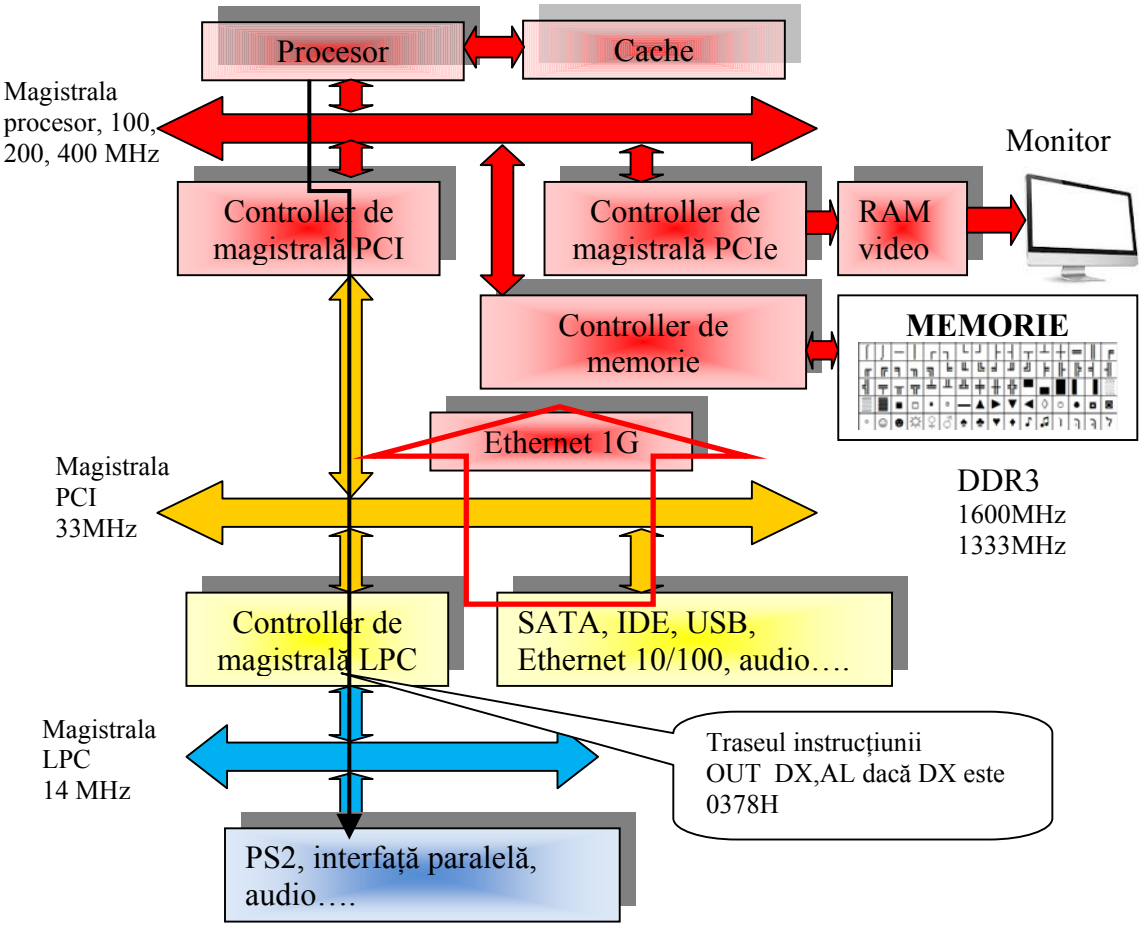


Figura 3.3. Principiul magistralei ierarhizate la PC

3.3. Diagrame de semnal la acces

Diagramele de timp ale accesului pe magistrală arată modul cum se desfăşoară în timp transferul de date şi rolul semnalelor importante de comandă.

Pe o magistrală sincronă transferul de date durează un anumit număr de tacte de ceas, perioada unui transfer fiind numit ciclu. Ciclurile pot fi de citire sau scriere (din perspectiva procesorului), pot fi cicluri de transfer cu memoria sau cu dispozitivele de intrare ieşire, pot fi cicluri de transfer gestionate de procesor sau cicluri de acces DMA. Un ciclu deosebit care poate exista la anumite sisteme este cel de întrerupere. Sunt exemplificate ciclurile de magistrală în cazul unei magistrale generice simple, asemănătoare cu cea mai simplă magistrală (ISA / LPC) la sistemele x86 şi cu magistralele microcontrollerelor.

Ciclul de citire din memorie este reprezentat în figura 3.4. Un astfel de ciclu este declanşat de o instrucţiune de citire din memorie, la x86 este MOV AL, [BX].

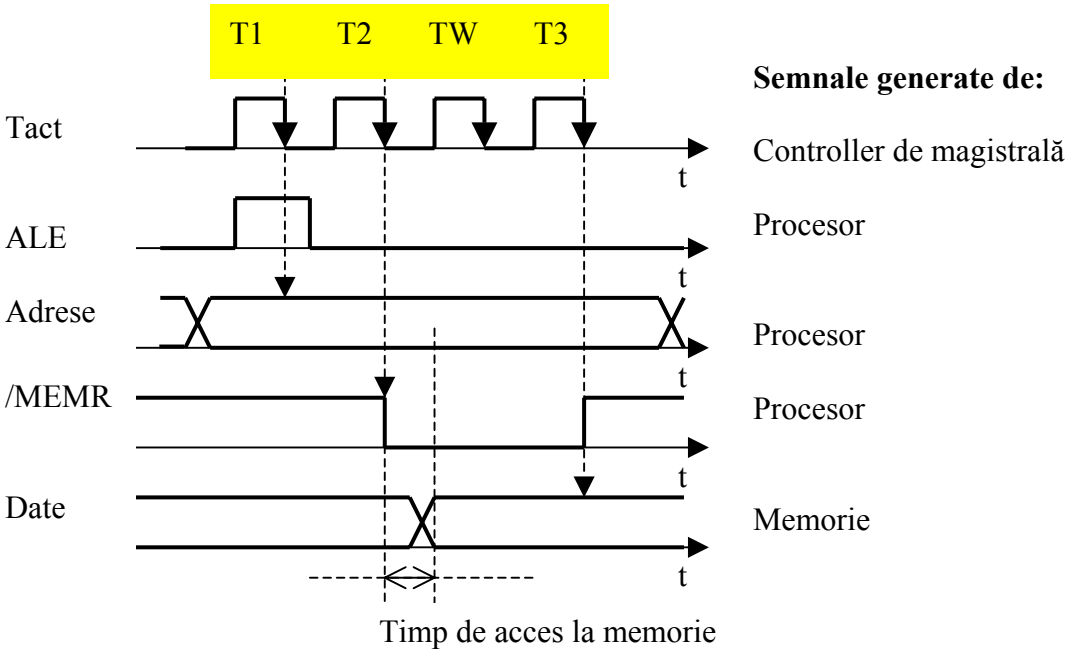


Figura 3.4. Ciclul de citire din memorie

Semnalul ALE (Address Latch Enable) este utilizat pentru memorarea adreselor într-un latch. Memorarea se face pe frontul primului impuls de tact. La executarea instrucţiunii de citire din memorie MOV AL, [BX] procesorul activează semnalul MEMR. După trecerea unui timp de întârziere datele sunt citite din memorie şi apar pe magistrala de

date. Ele sunt eşantionate de frontul ultimului impuls de tact. De regulă memoria este mai lentă decât procesorul, de aceea este posibil ca la citire şi scriere să se introducă un impuls de tact suplimentar TW numit impuls de *wait* care generează o stare de *wait*. La acest ciclu de citire celelalte semnale de comandă MEMW, IOR şi IOW sunt inactive.

Ciclul de scriere în memorie este reprezentat în figura 3.5 şi este asemănător cu ciclul de citire. Un astfel de ciclu este declanşat de o instrucțiune de scriere în memorie, la x86 este MOV [BX], AL.

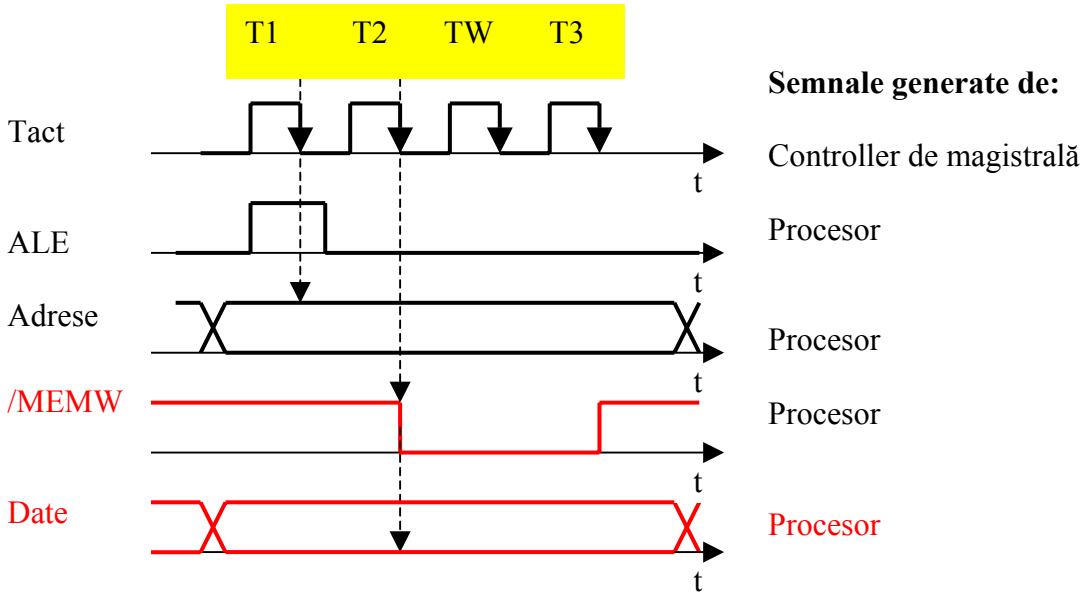


Figura 3.5. Ciclul de scriere în memorie

Instrucțiunea de scriere în memorie MOV [BX], AL activează semnalul MEMW, celelalte semnale de comandă fiind inactive. Modificările față de ciclul de citire au fost reprezentate cu roşu pentru a fi mai vizibile.

Ciclul de citire de la un dispozitiv de I/O este reprezentat în figura 3.6. Un astfel de ciclu este declanşat de o instrucțiune de citire de la un port, la x86 este IN AL, DX. Execuția acestei instrucțiuni activează semnalul IOR. După trecerea unui interval de timp de acces la dispozitivul de I/O datele sunt disponibile pe magistrala de date. Datele sunt eşantionate pe frontul crescător al semnalului /IOR. Dispozitivele de I/O sunt mai lente decât memoria aşa încât poate fi nevoie de inserarea mai multor impulsuri de tact TW de aşteptare (WAIT)

Ciclul de scriere la un dispozitiv de I/O este reprezentat în figura 3.7. Un astfel de ciclu este declanşat de o instrucțiune de scriere la un port, la x86 este OUT DX, AL.

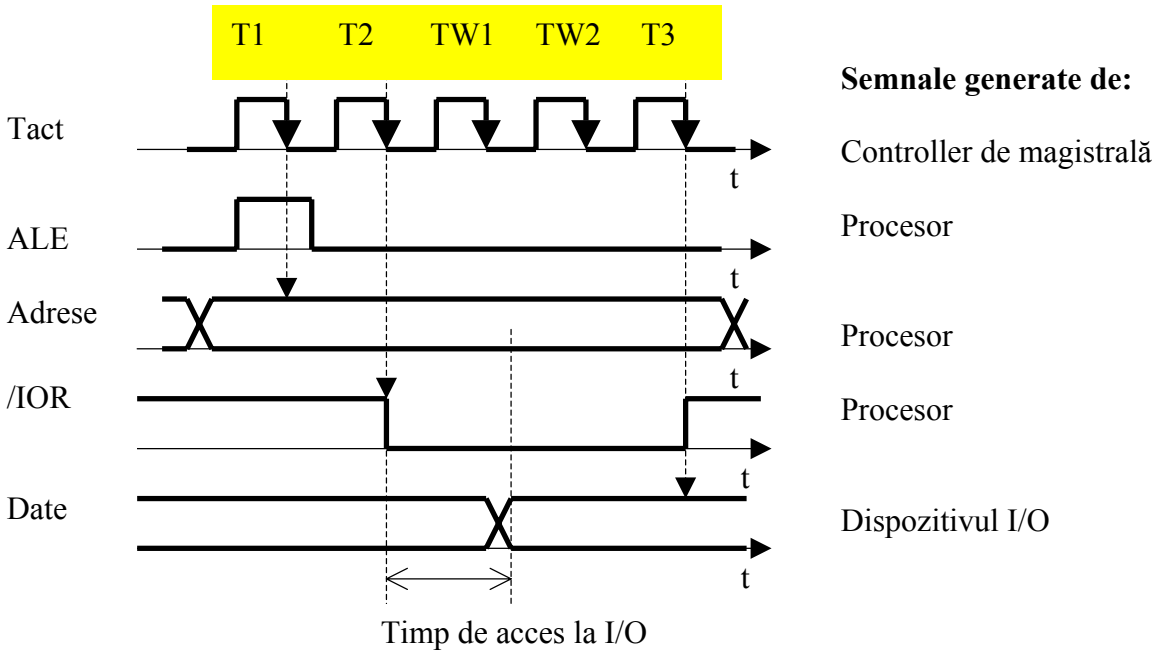


Figura 3.6. Ciclul de citire de la I/O

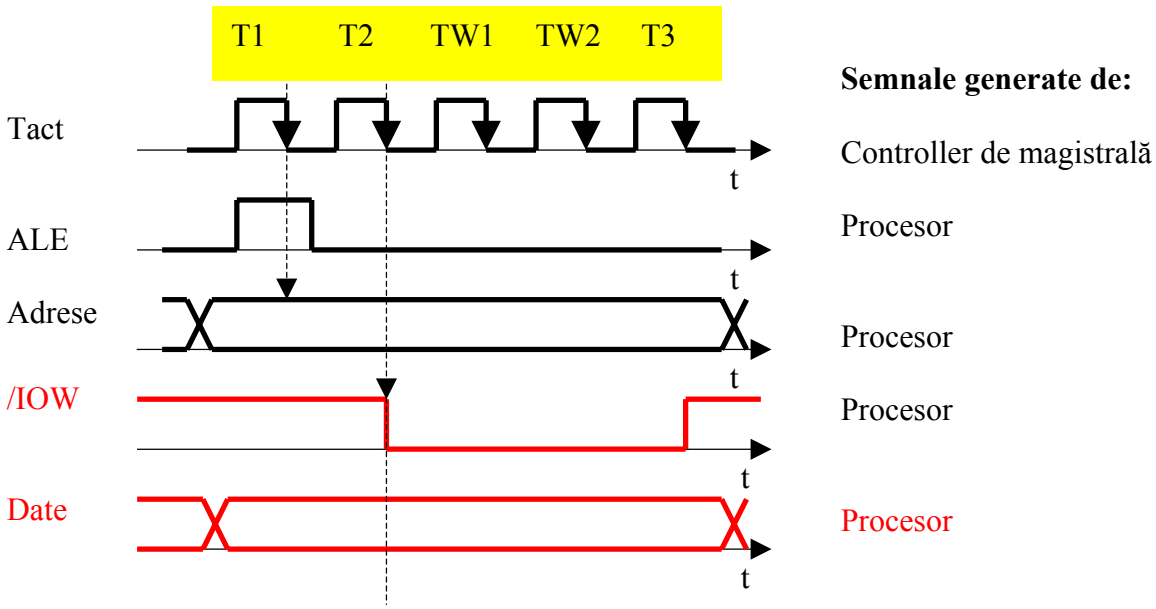


Figura 3.7. Ciclul de scriere la I/O

Modificările la ciclul de scriere față de ciclul de citire sunt marcate cu roşu pentru a uşura înțelegerea.

În ciclurile DMA procesorul nu are controlul magistralelor, controlul fiind preluat de controllerul de magistrală. Ciclul de scriere DMA reprezentat în figura 3.8. înseamnă citirea de la un dispozitiv de I/O şi scrierea în memorie.

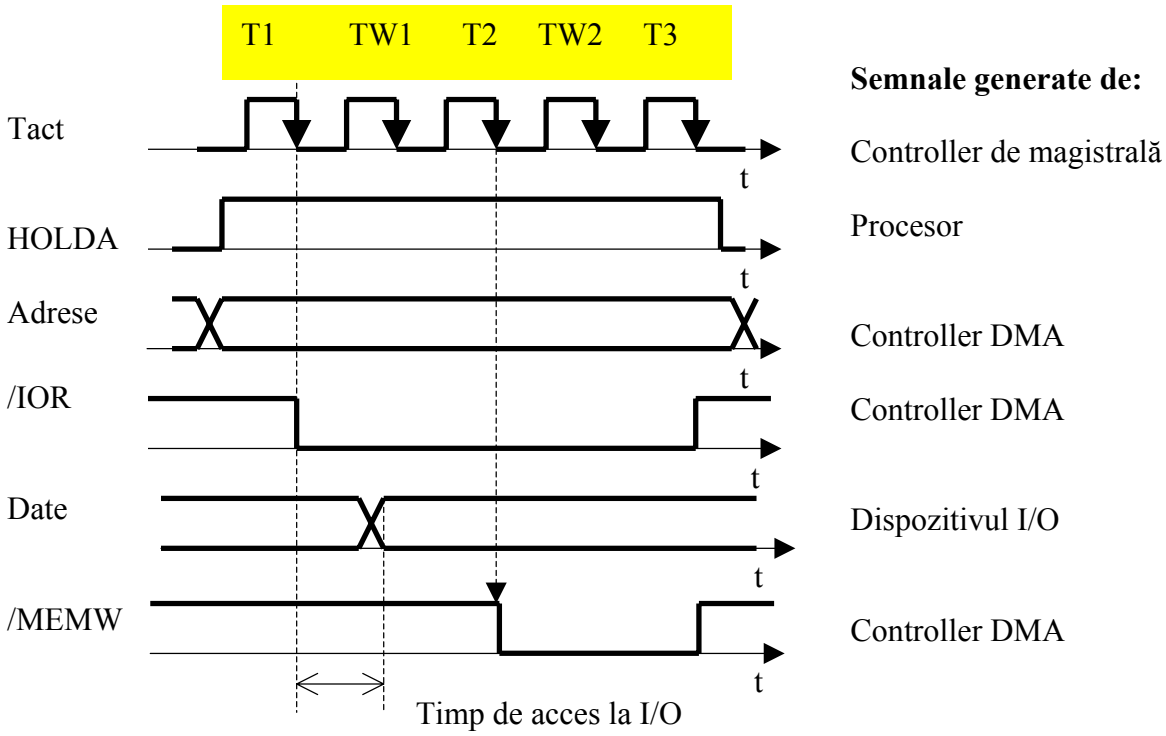


Figura 3.8. Ciclul de scriere DMA

Cedarea magistralelor de către procesor se face în urma unui dialog, după care procesorul confirmă cedarea prin activarea semnalului HOLDA. Controllerul DMA pune adresa de memorie pe magistrală şi activează semnalul /IOR. Datele sunt puse de dispozitivul de I/O pe magistrală, după care controllerul DMA activează semnalul /MEMW şi datele se înscriu în memorie.

Ciclul de citire DMA reprezentat în figura 3.9. înseamnă citirea din memorie şi scrierea la un dispozitiv de I/O. Cu roşu au fost marcate modificările.

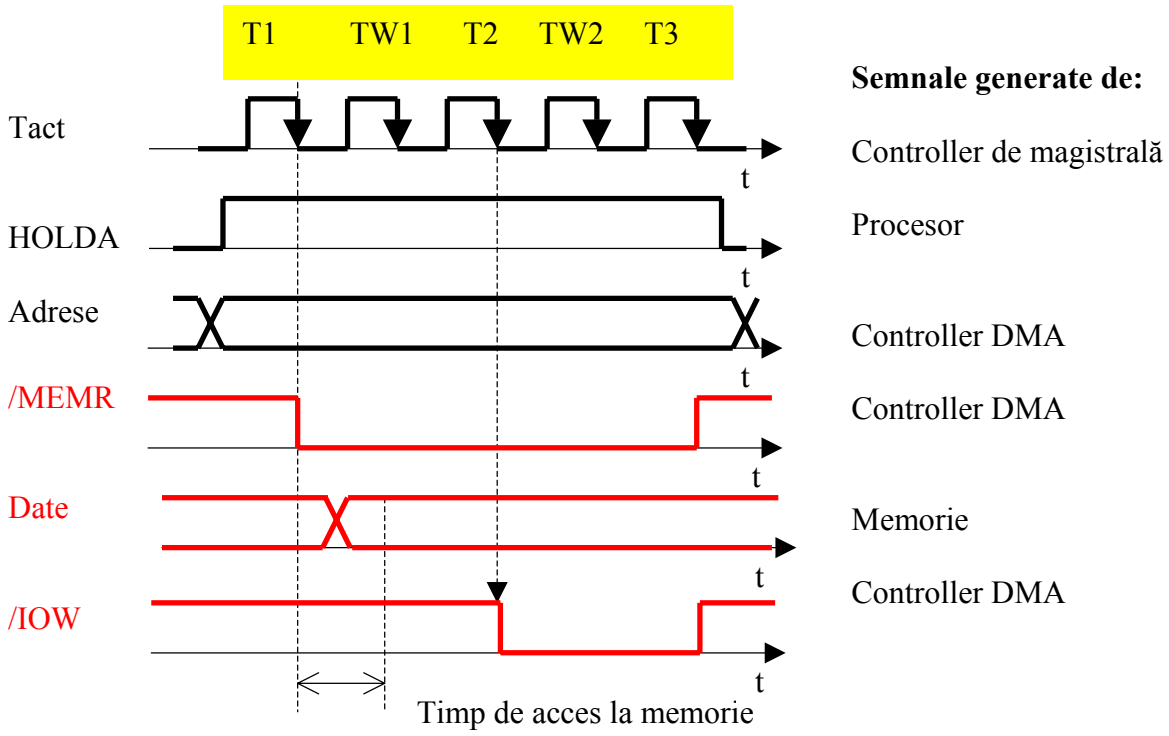


Figura 3.9. Ciclul de citire DMA

Din diagramele de semnal se poate determina aproximativ creşterea vitezei de transfer prin DMA. Un transfer programat înseamnă o instrucţiune de acces la memorie (4 tacte) şi una de acces la un dispozitivde I/O (5 tacte), în total 9 tacte. Un ciclu DMA care face acelaşi lucru durează 5 tacte, deci creşterea de viteză este de 9/5 ori (de 1,8 ori). Desigur că mărirea vitezei de transfer depinde de numărul de tacte de *wait* introduse.

3.4. Magistrale multiplexate

Microcontrollerele au beneficiat la apariţie de experienţa dezvoltării magistralelor de PC şi au împrumutat variantele cele mai simple de magistrale sincrone. Transferul de date pe o magistrală sincronă apare la execuţia unei instrucţiuni a microcontrollerului. La execuţia instrucţiunii, în funcţie de instrucţiunea executată, se generează automat un semnal care stabileşte spaţiul de adresare şi sensul transferului. În cazul în care spaţiul de adresare este comun pentru zona de memorie şi zona de I/O atunci există două semnale de comandă - RD şi WR care stabilesc sensul transferului. Dacă spaţiul de adresare este diferit pentru memorie şi IO atunci există patru semnale- IOR, IOW, MEMR, MEMW.

Diagrame de timp pentru transferuri tipice pe o magistrală sincronă simplă cu spațiu comun pentru memorie şi I/O au fost prezentate anterior.

La un PC la magistralele vechi liniile de adresă şi de date sunt diferite pentru că nu s-a pus problema economiei de spațiu. De regulă, în microcontrollere şi la magistralele noi de PC nu se poate accepta un număr atât de mare de linii din cauza costurilor de realizare a circuitului, de aceea liniile de adrese şi date sunt multiplexate. Aceasta înseamnă cuplarea unui latch de adrese în exteriorul microcontrollerului care să fie încărcat cu adrese, comanda latch-ului fiind realizată cu semnalul ALE (Address Latch Enable). O diagramă de timp în acest caz este dată în figura 3.10:

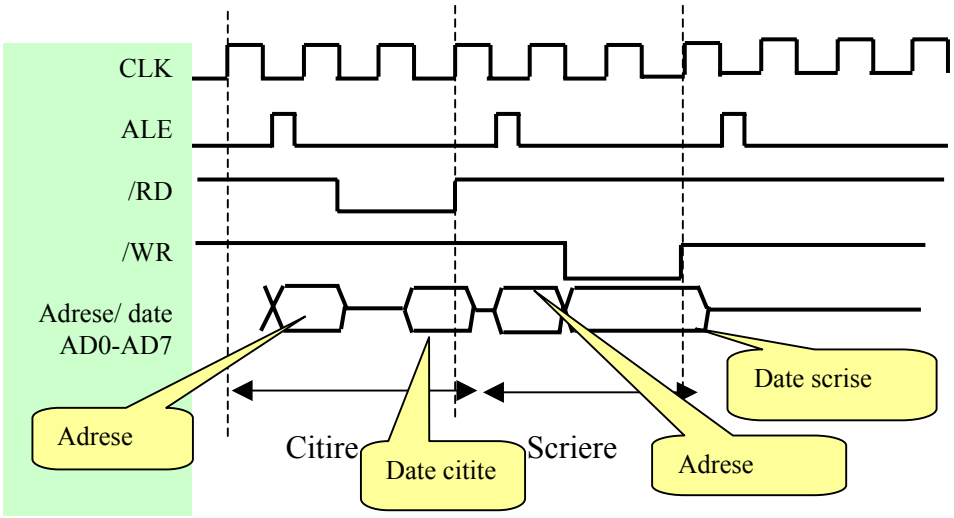


Figura 3.10. Diagrama de timp pentru un ciclu de citire urmat de un ciclu de scriere în cazul unei magistrale de adrese şi date multiplexate

3.5.Magistralele PCI şi PCI Express

Asociația PCI-SIG (www.pcisig.com) grupează un număr mare de producători (circa 900) interesați în armonizarea structurilor şi caracteristicilor magistralelor PCI. Tabelul 3.1 arată evoluția în timp a caracteristicilor magistralelor PCI:

Tabel 3.1.

Magistrala	Anul apariției	Debit maxim
PCI 33MHz	1993	133Mo/s
PCI 66MHz	1995	266Mo/s
PCI-X 133MHz	1999	533Mo/s

PCI-X 266MHz	2002	1066Mo/s
PCI-X 533MHz	2002	2131Mo/s
PCI Express	2002	16Go/s (x32)

Specificaţiile PCI permit cuplarea a maximum 32 de dispozitive pe magistrală. Cu toate acestea, din motive de încărcare a magistralei, practic se pot conecta doar 5-10 dispozitive. O diagramă de semnale tipică pentru accesul la magistrala PCI este dată în figura 3.11.

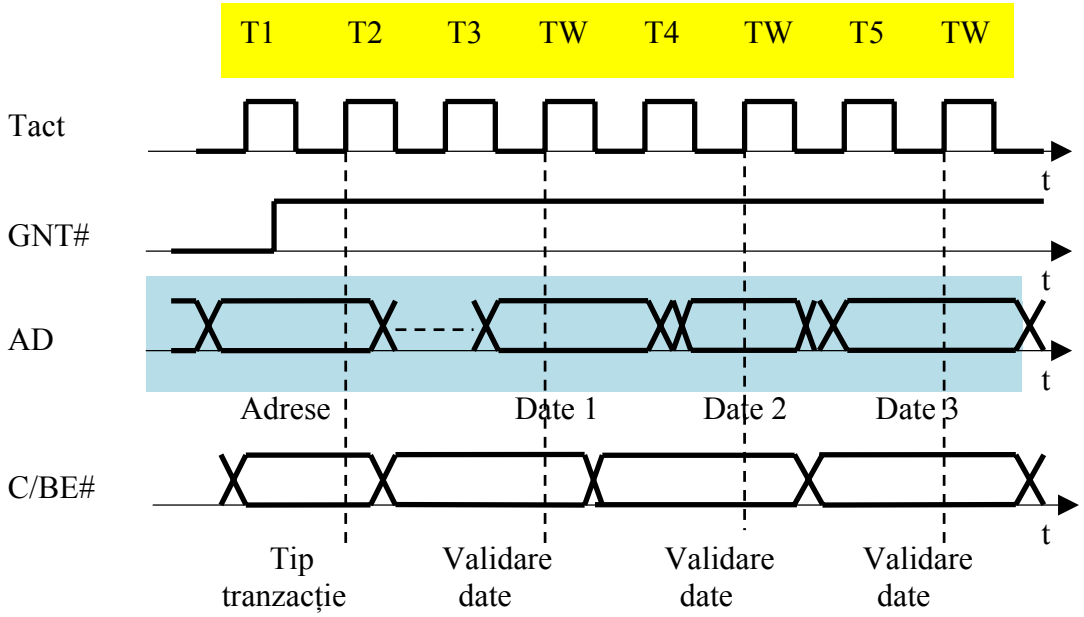


Figura 3.11. O diagramă de semnale pentru accesul la magistrala PCI

Este foarte importantă caracteristica PCI de a permite transferuri în salvă (burst). Magistrala PCI admite 12 tipuri de tranzacţii (cicluri) printre care tranzacţii cu memoria, cu dispozitive de I/O, de configurare, etc.

Liniile de date şi adrese sunt multiplexate (AD) pentru a reduce numărul de linii. Un dispozitiv master cuplat pe magistrală poate solicita magistrala cu un semnal REQ# şi când aceasta este liberă controllerul de magistrală va răspunde cu GNT#. La primul tact pe liniile AD masterul pune adresa, iar pe liniile C/BE# se pune tipul tranzacţiei. După adresă, pe liniile AD urmează datele, cuvânt după cuvânt. Se poate vedea că după fiecare tact urmează un tact de *wait* pentru a permite sincronizarea cu date puse pe magistrală cu diferite întârzieri. Pe liniile C/BE# se pun cuvinte de validare a datelor.

Una dintre problemele majore ale magistralelor paralele este transmisia semnalelor pe mai multe linii (16, 32 sau 64), apropiate între ele, la viteze de transfer din ce în ce mai mari. Efectele perturbatoare care se manifestă la aceste transmisii, în special reflexiile şi diafonia nu mai pot fi neglijate şi limitează creşterea în continuare a vitezei de transfer. Soluții care micşorează diafonia şi reflexiile cum ar fi de exemplu trasee mai late (cu rezistență mai mică), introducerea traseelor de masă între traseele de semnal etc. nu pot fi practic aplicate pentru că dimensiunile noilor echipamente trebuie să fie din ce în ce mai mici. Sunt foarte interesante soluții de codificare a datelor transmise pe magistrala paralelă astfel încât combinațiile de nivele logice pe linii adiacente care produc diafonii să fie eliminate. În figura 3.12. se poate vedea un asemenea sistem care verifică combinațiile care produc diafonii şi le înlocuieşte, semnalizând acest lucru cu o linie specială care se adaugă la liniile magistralei.

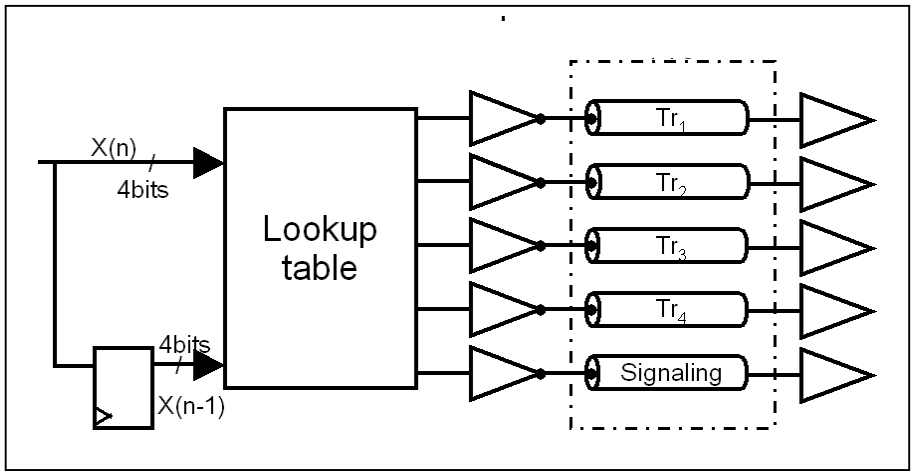


Figura 3.12. Eliminarea combinațiilor care produc diafonii (Sursa: M. Dărăban, Teză de doctorat, Cluj Napoca, 2012, online: http://www.etti.utcluj.ro/download/988_PhD_Thesis.pdf)

Un răspuns posibil la aceste probleme este apariția unui nou tip de magistrală. PCI Express are o altă filozofie a magistralelor, de la magistrale paralele la care fiecare linie este de semnal este un traseu clasic, la o magistrală paralelă care conține mai multe grupuri de linii de comunicații seriale numite canale. Un canal conține 2 linii de transmisie seriale unidirecționale (simplex), fiecare linie fiind compusă din 2 fire cu transmisie diferențială (High- speed LVDS, Low Voltage Differential Signaling), figura 3.13.

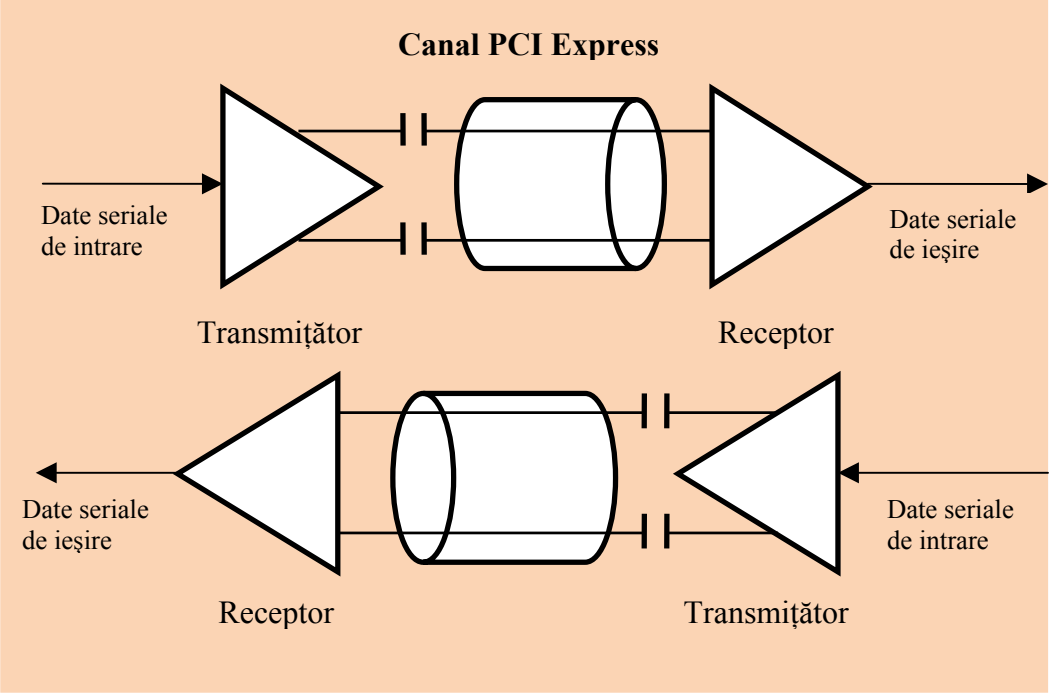


Figura 3.13. Canal PCI Express

Transmisia serială la PCI Express este o transmisie punct la punct, sincronă, cu cadre de date, cu refacerea tactului din datele citite, codarea fiind o codare de grup numită 8b/10b, care va fi tratată într-un capitol ulterior, dedicat interfeței IEEE1394.

Magistrala PCI Express poate conține un canal (PCIe x2), 2 canale (PCIe x4), ş.a.m.d. până la 16 canale (PCIe x32).

Controllerul de magistrală conține un comutator (Switch) care comută un dispozitiv master la unul slave, ca în diagrama generală din figura 3.1. O schemă generală a magistralei PCI Express este dată în figura 3.14.

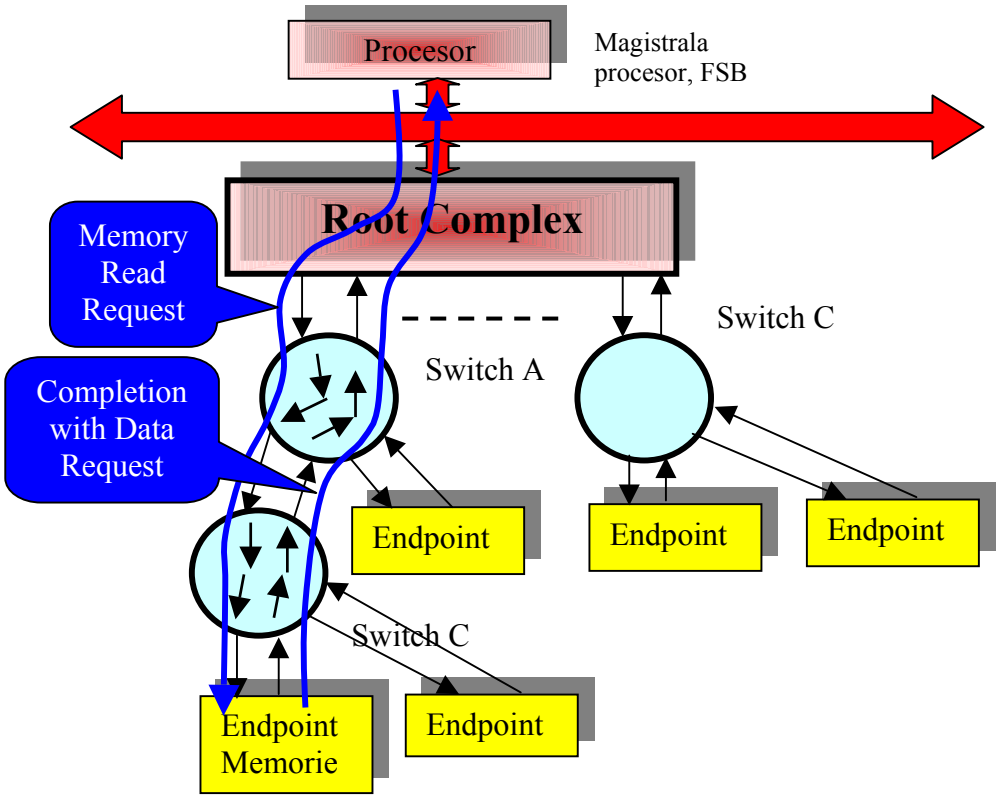


Figura 3.14. Structura magistralei PCI Express

Arhitectura magistralei seamănă cu arhitectura USB, viteza superioară fiind asigurată de un număr mai mare de canale seriale prin care se transferă date simultan. Dacă procesorul are nevoie de un transfer de date de citire de la memorie atunci trimite un cadru de comandă Memory Read Request. Aceasta ajunge la Endpoint-ul memorie prin trecerea prin Switch-ul A şi C, comandate pentru a asigura traseul către Endpoint-ul dorit de către controllerul de magistrală. Memoria răspunde acestei cereri cu cadrul Completion with Data Request. Transferul de date este bidirecţional full duplex prin fiecare canal serial cu câte două linii de transmisie. Toate dispozitivele cuplate la magistrală conţin buffere de transmisie şi recepţie.

4.Interfețe paralele

4.1. Interfețe paralele neprogramabile

Interfețele paralele pot fi programabile sau neprogramabile. O altă clasificare împarte interfețele paralele în interfețe unidirecționale şi bidirecționale. Rolul interfețelor paralele este ca să extindă numărul de linii de transfer paralel de date sau să introducă un protocol pentru gestionarea unui transfer de date.

A.Interfața neprogramabilă unidirecțională pe 8 biți

Schema bloc a unei astfel de interfețe (Intel 8282) este dată în figura 4.1:

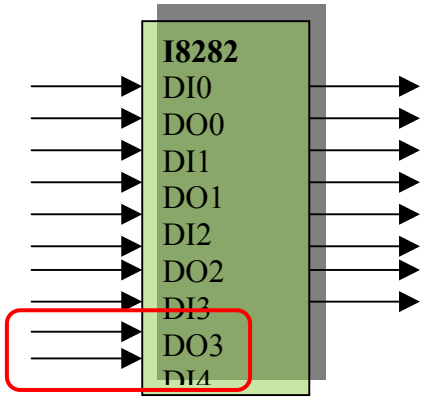


Figura 4.1. Interfața paralelă neprogramabilă unidirecțională

Circuitul este un buffer pe 8 biți cu ieşiri cu trei stări. Dacă STB (Strobe) este logic 1 ieşirile urmăresc nivelul logic al intrărilor. Pe frontul negativ al semnalului STB datele de intrare sunt memorate în circuit. Semnalul /OE (Output Enable) /OE logic 1 face ca ieşirile să treacă în înaltă impedanță. La trecerea lui /OE din unu în zero datele se pierd din buffer. Există astfel de circuite care au un inversor inserat pe fiecare linie. Principala utilizare a acestor circuite este pentru liniile de adresă, înscrierea în buffer fiind asigurată de semnalul ALE. Sunt marcate într-un chenar roşu semnalele care adaugă un protocol simplu de transfer.

B.Interfața neprogamabilă bidirecțională pe 8 biți

Schema bloc a unei astfel de interfețe (Intel 8286) este dată în figura 4.2.

Circuitul are ieşiri cu trei stări şi se utilizează mai ales la transferul pe magistrala de date, fiind bidirecțional. Intrarea T stabileşte sensul de transfer al datelor, pe nivel sensul este de la A la B iar pe nivel zero de la B la A. Dacă /OE este 1 liniile din A şi B trec în înaltă

impedanță. Sunt marcate într-un chenar roşu semnalele care adaugă un protocol simplu de transfer.

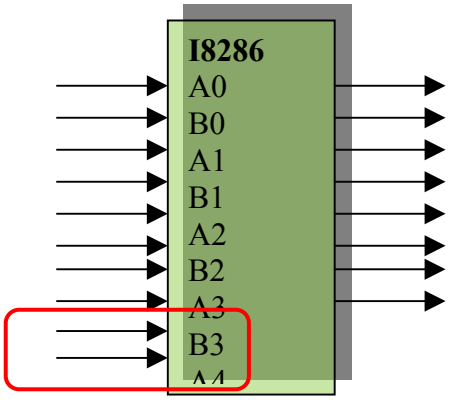


Figura 4.2. Interfața paralelă neprogramabilă bidirecțională

4.2. Interfața paralelă programabilă

Circuitul tipic pentru această categorie este circuitul Intel 8255 care a fost realizat pentru prima oară pentru microprocesorul Intel 8085, dar a fost preluat la microprocesoarele pe 16 biți (Intel 80286) şi utilizat la primele PC-uri. Circuitul a fost foarte reuşit şi ca urmare a fost preluat şi de alți producători pentru microprocesoarele proprii, cum a fost de exemplu Motorola pentru familia 6800, numind interfața Motorola 6820 PIA (Peripheral Interface Adapter).

Schema bloc a circuitului este dată în figura 4.3:

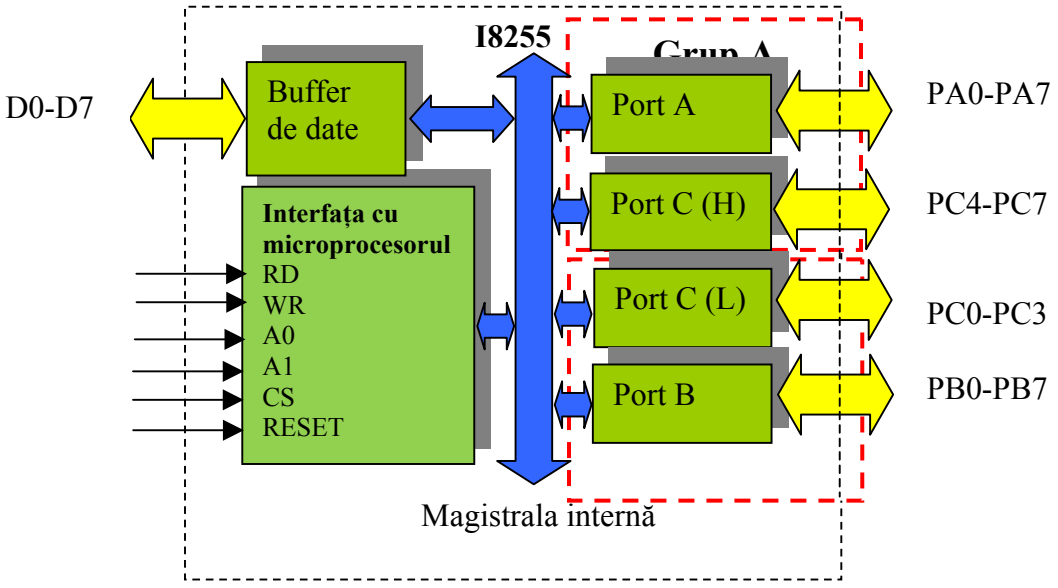


Figura 4.3. Schema bloc a circuitului interfață paralelă programabilă INTEL 8255

Semnalele de interfață cu microprocesorul sau microcontrollerul gazdă au următoarea semnificație:

- RD se execută cu ciclu de citire de la un port sau de la memorie. Circuitul se poate mapa în zona de memorie sau I/O, dar este firesc ca el să fie mapat în zona de I/O. Astfel, la acest pin se conectează semnalul de magistrală IOR;
- WR se execută cu ciclu de scriere la un port sau în memorie. La acest pin se conectează semnalul de magistrală IOW;
- A0 este o linie care împreună cu A1 selectează registrele interne ale interfeței paralele. Se conectează de regulă la linia cel mai puțin semnificativă de adresă;
- A1 este o linie care împreună cu A0 selectează registrele interne ale interfeței paralele. Se conectează de regulă la linia de adresă A1 (A0 este cel mai puțin semnificativ);
- RESET este o linie care comandă inițializarea circuitului prin ştergerea informației din toate registrele;
- CS este o linie care selectează circuitul. Formării acestui semnal I se dedică un capitol ulterior;
- D0-D7 magistrala de date a gazdei, 8 linii bidirecționale;
- PA0-PA7 8 linii bidirecționale care formează portul A;
- PC4-PC7 4 linii bidirecționale, partea mai semnificativă (H) a portului C care pot fi folosite independent sau ca şi semnale de protocol pentru portul A. Aceste linii formează împreună cu portul A grupul A;
- PC0-PC3 4 linii bidirecționale, partea mai puțin semnificativă (L) a portului C care pot fi folosite independent sau ca şi semnale de protocol pentru portul B. Aceste linii formează împreună cu portul B grupul B;
- PB0-PB7 8 linii bidirecționale care formează portul B.

În funcție de liniile de adresă A0 şi A1, dacă CS este activ, se selectează porturile conform tabelului următor:

A1	**A0**	**Registru**
0	0	portA
0	1	port B
1	0	port C
1	1	registru de comandă

Modurile de lucru ale interfeței sunt:

1. Modul 0 este modul de bază de intrare / ieşire, asigură funcționarea porturilor A şi B ca porturi de 8 biți. Portul C cu cele două părți ale lui de 4 biți are posibilitatea de poziționare individuală a fiecărei linii pe zero sau unu. Porturile pot fi programate ca ieşiri sau intrări şi mențin sensul programat până la o nouă programare.
2. Modul 1 este un mod de lucru cu posibilitatea de implementare a unui protocol de transfer. Se pot folosi două grupuri, grupul A şi grupul B formate din porturile A şi B ca porturi de date, asistate de semnale de comandă din portul C. Porturile de date A şi B pot fi programate ca ieşiri sau intrări şi mențin sensul programat până la o nouă programare. Se poate folosi unul dintre porturile A sau B în mod 0 şi celălalt în mod 1.
3. Modul 2 este un mod în care portul A este folosit ca port de date bidirecțional, asistat de toate liniile portului C. Portul B poate fi folosit în acest caz doar în modul 0.

4.3.Protocoale de transfer

A.Protocolul de transfer în mod 1

Pentru a ilustra protocoalele de transfer paralel sunt prezentate trei variante, una pentru transferul unidirecțional de intrare, una pentru transferul unidirecțional de ieşire şi una pentru transferul bidirecțional. Modurile de transfer prezentate sunt cele de la circuitul interfață programabilă I8255 descris anterior şi sunt tipice pentru transferul paralel al datelor.

A1. Transferul unidirecțional de intrare

Schema bloc de conectare a unui Echipament Periferic (EP) la portul A este dată în figura 4.4.

Semnalele de protocol din portul C sunt generate automat de circuitul I8255 iar semnalele de stare pot fi citite de procesorul gazdă. Semnalele de protocol au următoarea semnificație:

STBA (Strobe A) este un strob de intrare pe poziția liniei PC4. Cu acest semnal EP încarcă datele puse pe liniile de date în portul de date A.

IBFA (Input Buffer Full A) este o linie de stare pe poziția liniei PC5 care arată că datele au fost încărcate în bufferul portului A. Acest semnal este activat de semnalul de STBA de la EP şi este dezactivat când procesorul citeşte datele (când apare un RD active). Semnalul IBFA poate fi folosit de procesor care este informat că datele sunt în portul A sau de EP ca o confirmare a primirii datelor.

INTRA (Interrupt Request A) este o cerere de întrerupere către microprocesor pe poziția liniei PC3, activată când datele sunt în portul A. Cererea este activată de semnalul STBA şi dezactivată de semnalul RD. Cererea este validată de un bistabil intern comandat de **INTEA** (Interrupt Enable A), semnal de validare întreruperi pe poziția PC4.

Liniile PC6 şi PC7 pot fi utilizate de către proiectant pentru a implementa operații suplimentare de protocol. Dacă aceste operații nu sunt necesare, se pot folosi liniile ca linii de semnalizare a operațiilor în curs (cum este de exemplu LED-ul care indică transferul cu hard discul pe panoul frontal al unui PC).

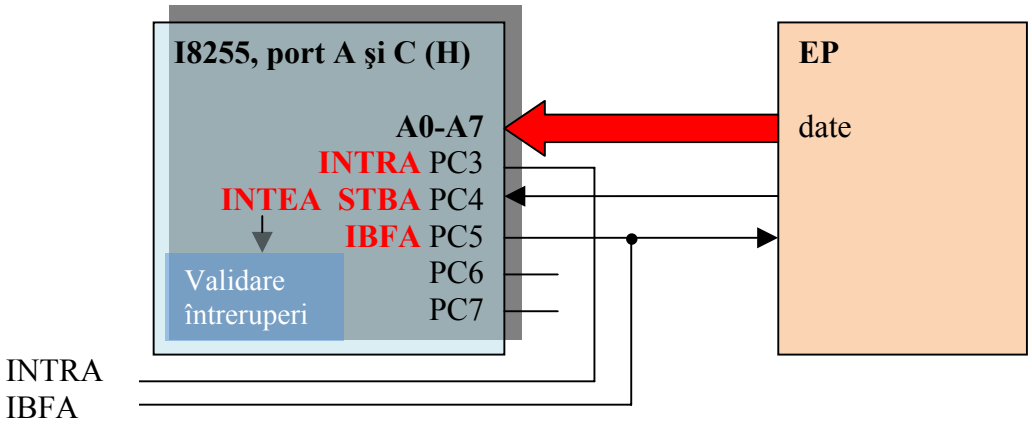


Figura 4.4. Protocolul de citire date

Corespunzător portului B semnalele sunt: INTEB (PC2), STBB (PC2), IBFB (PC1), INTRB (PC0).

A2. Transferul unidirecțional de ieşire

Protocolul de scriere date este ilustrat în figura 4.5. Semnalele de protocol sunt:

OBFA (Output Buffer Full A) pe poziția liniei PC7 semnalează că registrul de ieşire este plin şi datele pot fi preluate de EP. Semnalul este activat de scrierea datelor în I8255 de către procesor, lucru indicat de semnalul de magistrală WR şi este dezactivat de preluarea datelor de către EP, semnalizată cu semnalul ACKA. Semnalul OBFA poate fi folosit de procesor care este informat că datele au fost preluate de EP sau de EP care este informat că datele sunt în portul A şi pot fi citite.

ACKA (Acknowledge A) este un semnal pe poziția liniei PC6 care confirmă că datele au fost acceptate de EP.

INTRA (Interrupt Request A) este o cerere de întrerupere către microprocesor pe poziția liniei PC3, activată când datele au fost preluate de EP. Cererea de întrerupere este activată de semnalul ACKA şi dezactivată de WR. Cererea este validată de un bistabil intern comandat de **INTEA** (Interrupt Enable A), semnal de validare întreruperi pe poziția PC6.

Liniile PC4 şi PC5 pot fi utilizate de către proiectant pentru a implementa operații suplimentare de protocol.

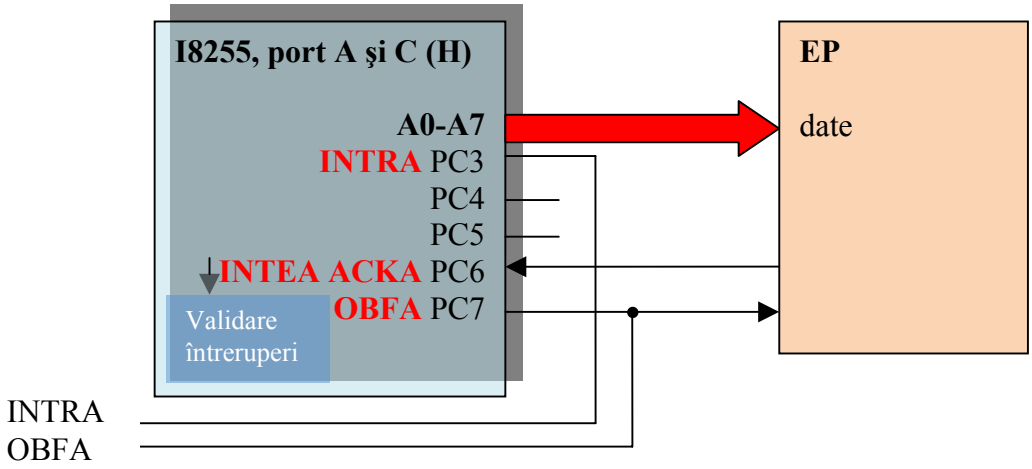


Figura 4.5. Protocolul de scriere date

Corespunzător portului B semnalele sunt: INTEB (PC2), OBFB (PC1), ACKB (PC2), INTRB (PC0).

A3.Transferul bidirecțional în modul 2 de funcționare

Acest mod de transfer poate fi realizat în modul 2 de funcționare al interfeței în care transferul are loc cu portul A asistat de semnale de protocol din portul C. Semnalele de protocol sunt date în figura 4.6. şi sunt următoarele:

INTRA (PC3) cerere de întrerupere, generată la operațiile de intrare şi ieşire. Întreruperile generate de o operație de ieşire sunt validate de **INTE1** (PC6) iar cele generate de o operație de intrare de **INTE2** (PC4)

OBFA (PC7) generată în cadrul unei operații de ieşire

ACKA (PC6) generată în cadrul unei operații de ieşire

STBA (PC4) generată în cadrul unei operații de intrare

IBFA (PC5) generată în cadrul unei operații de intrare

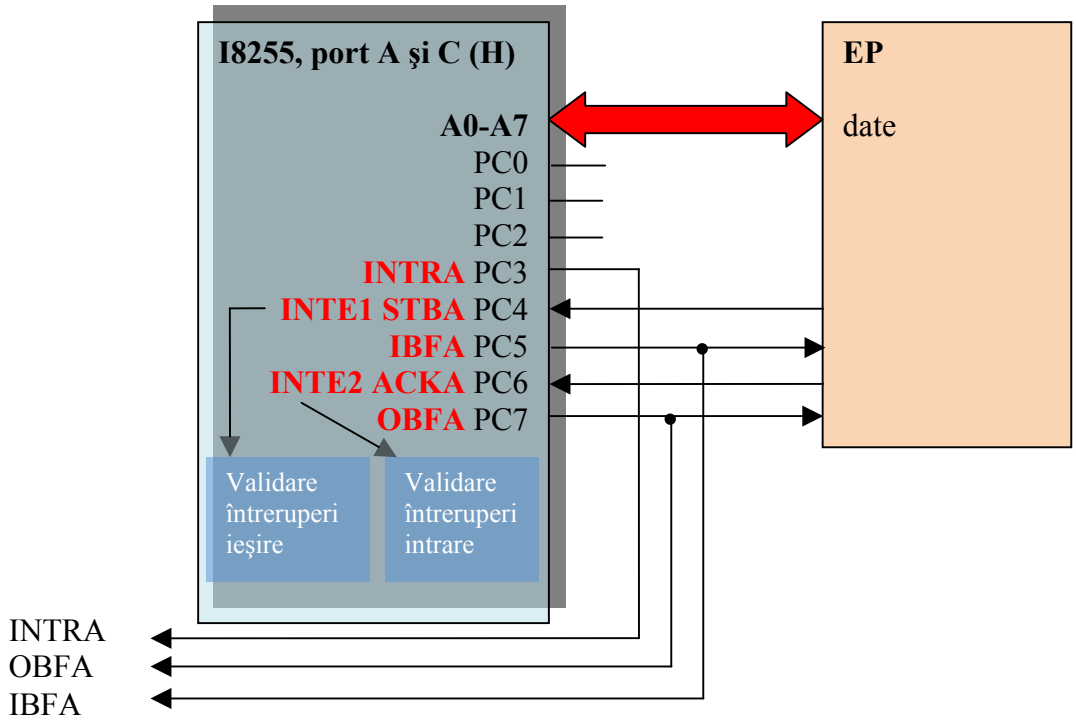


Figura 4.6. Protocolul bidirecțional

4.4.Programarea circuitului de interfață paralelă

Înainte de începerea unui transfer de date circuitul de interfață paralelă trebuie programat. Pentru aceasta, în primul rând se trimite un cuvânt care comandă modul de lucru în registrul de comandă. Structura acestui cuvânt este următoarea:

D7 (MSBit)=1

D6	**D5**	**mod grup A**
0	0	grup A mod 0
0	1	grup A mod 1
1	x	grup A mod 2

D4=0 A port de ieşire, D4=1 A port de intrare
D3=0 PC4-PC7 ieşiri, D3=1PC4-PC7 intrări
D2=0 mod 0 pentru grup B, D2=1 mod 1 pentru grup B
D1=0 B port de ieşire, D1=1 B port de intrare
D0=0 PC0-PC3 ieşiri, D0=1PC0-PC3 intrări

Un al doilea cuvânt de comandă se poate trimite în registrul de comandă. Dacă s-au selectat modurile 1 sau 2 de lucru este necesar să fie accesibil fiecare bit al portului C pentru poziționare individuală. Structura cuvântului de poziționare a unui bit în portul C este următoarea:

D7=0, D6=x, D5=x, D4=x
D3, D2, D1 reprezintă prin decodificare adresa bitului în octet
D0 reprezintă valoarea cu care se înscrie bitul selectat.

Starea circuitului interfață paralelă în modurile 1 şi 2 de lucru poate fi determinată prin citirea portului C, astfel:

Bit	**Mod 1 starea port A intrare**	**Mod 1 starea port A ieşire**
D7	I/O	OBFA
D6	I/O	ACKA
D5	IBFA	I/O
D4	STBA	I/O
D3	INTRA	INTRA

Bit	**Mod 1 starea port B intrare**	**Mod 1 starea port B ieşire**
D2	STBB	ACKB
D1	IBFB	OBFB
D0	INTRB	INTRB

Bit	**Mod 2 starea port A**
D7	OBFA
D6	INTE1
D5	IBFA
D4	INTE2
D3	INTRA

4.5.Exemplu de implementare

Se propune realizarea unei interfețe paralele cuplată pe o magistrală de microprocesor compatibil x86 de 8 biți de date şi 16 biți de adresă. În portul A se conectează 8 LED-uri iar în portul B 4 întrerupătoare. Se cere realizarea schemei bloc şi a unui program în limbaj de asamblare care să afişeze pe LED-uri starea întrerupătoarelor.

Schema bloc a interfeței este dată în figura 4.7.

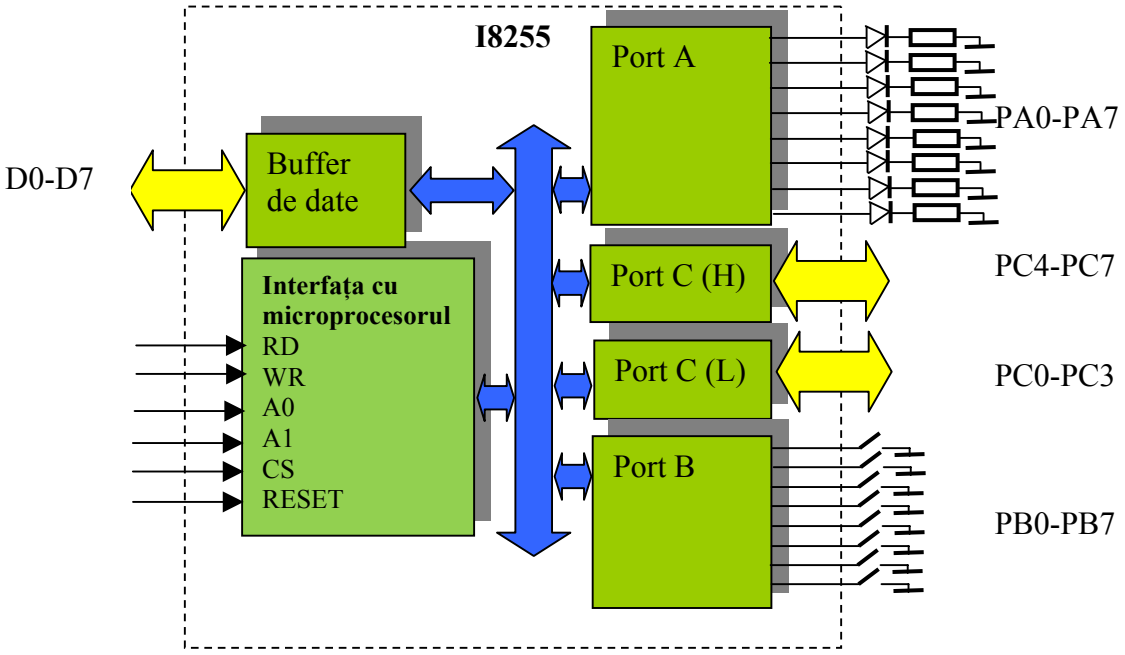


Figura 4.7. Schema electrică a interfeței cu întrerupătoare şi LED-uri

Se alege modul 0 de lucru a interfeței, acesta fiind cel mai simplu mod în care se poate rezolva problema. Se programează interfața paralelă apoi se citeşte portul B şi ceea ce s-a citit se trimite la portul A pentru a aprinde LED-urile. În această variantă închiderea unui întrerupător determină ca nivelul citit să fie zero, prin urmare se va afişa pe linia corespunzătoare a portului A cu un LED stins. Un întrerupător neapăsat va însemna un LED aprins.

Pentru programare portul A trebuie să fie de ieşire, portul B de intrare, iar portul C indiferent. Secvența de programare va fi:

```
MOV DX,0003
MOV AL, 82H
OUT DX,AL
```

Secvența de citire întrerupătoare şi afişare pe LED-uri va fi:

```
Start   MOV DX,0001
        IN AL,DX
        MOV DX,0000
        OUT DX,AL
        JMP start
```

5.Interfețe seriale

5.1.Tactul în transmisiile seriale

În primul capitol a fost demonstrată importanța cunoaşterii tactului cu care s-au emis date la receptor. Pentru a se asigura la receptor tactul corect de recepție există mai multe variante, prezentate în figura 5.1.

1. Tactul de transmisie se transmite de la emițător la receptor pentru citire, figura 5.1 a. Se asigură viteză mare, dar distanțele sunt mici. Costurile unui fir suplimentar sunt mari. Exemple sunt interfața SPI (Serial Peripheral Interface), IEEE1394a (1995), JTAG.

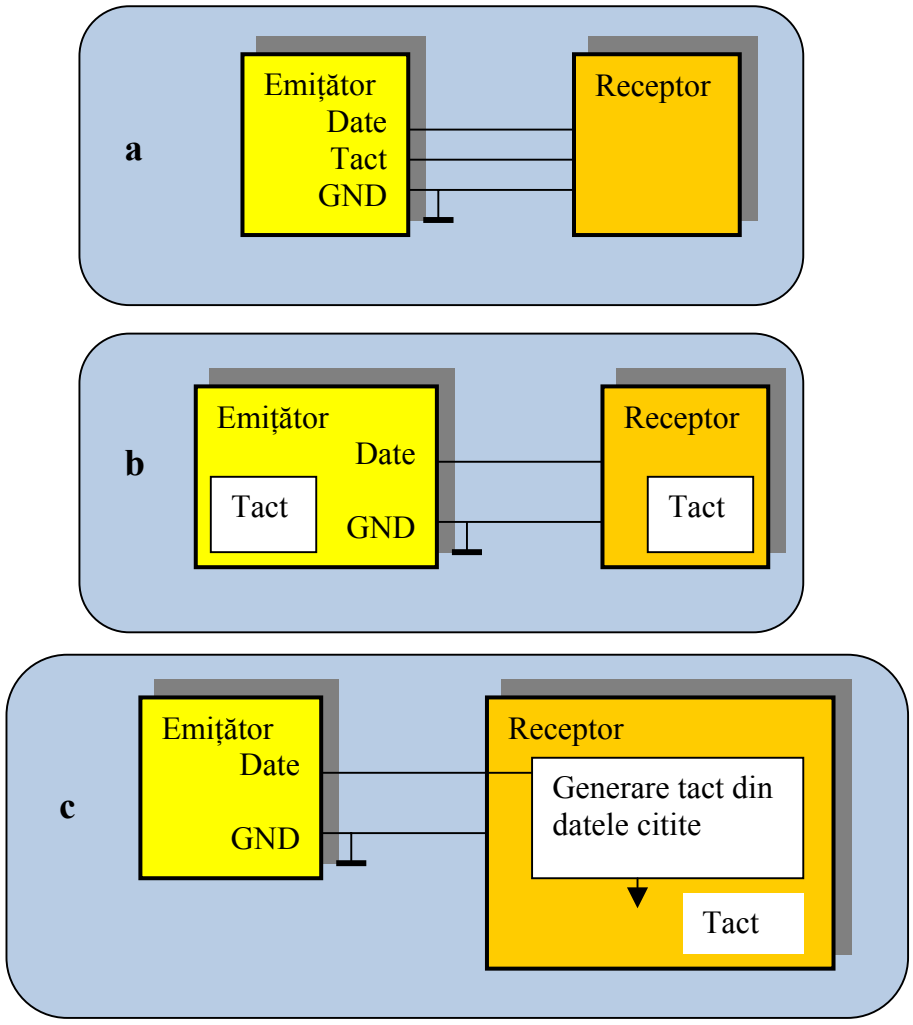


Figura 5.1. Variante de asigurare a tactului la recepție

1. Emisia şi recepția se fac cu acelaşi tact, standard, cunoscut şi de emițător şi de receptor. Distanțele pot fi mari, nu apar costurile unui fir suplimentar, figura 5.1 b. Viteza nu poate fi mare datorită faptului că între tactele standard există diferențe. Câteva exemple sunt interfața serială asincronă RS232 şi LIN.
2. Refacerea tactului din datele emise este cea mai modernă metodă folosită la cele mai multe aplicații actuale. Se pot asigura distanțe mari, costuri mici, viteze mari. Circuitul care reface tactul din date se numeşte buclă PLL şi nu poate reface datele decât dacă există variații permanente de nivel, figura 5.1 c. Exemple sunt USB, CAN, FlexRay, IEEE1394b (2002), dar şi Ethernet, SATA etc. Transmisiile wireless, de exemplu Bluetooth şi ZigBee pot fi asociate cu transmisiile seriale pe un singur fir cu refacerea tactului.

5.2. Codarea datelor

Primul pas în transmiterea datelor digitale pe o linie de transmisie este asocierea fiecărui bit cu variația unui semnal electric printr-un proces de codare. În continuare vor fi prezentate câteva tipuri de codări şi exemple de codare.

A.Codarea NRZ (Non Return to Zero)

Cea mai simplă şi naturală codare este cea în care fiecare valoare logică se codează cu un nivel de tensiune.

Regula de codare este:

1. Zero se codifică cu 0V;
2. Unu se codifică cu +5V.

În figura 5.2. 16 biți într-o succesiune aleasă la întâmplare în care datele au multe schimbări de nivel logic au fost codificați cu 7 tranziții (2,28 biți/tranziție). Dacă apare un şir de 0-uri sau 1-uri semnalul electric nu are variații.

Intervalul de timp în care se codează un bit se numeşte celulă bit (interval cu roşu)

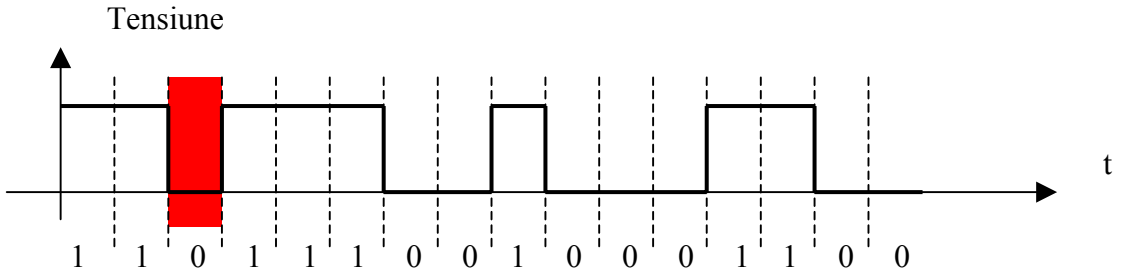


Figura 5.2. Codarea NRZ

B.Codarea NRZI (Non Return to Zero Inverted)

Este o codare asemănătoare cu NRZ, cu regula de codare:

1. Zero se codifică cu lipsa unui front.
2. Unu se codifică cu un front.

În figura 5.3. 16 biți într-o succesiune aleasă la întâmplare au fost codificați cu 7 tranziții (2,28 biți/tranziție). Dacă apare un şir de 0-uri sau 1-uri semnalul electric nu are variații.

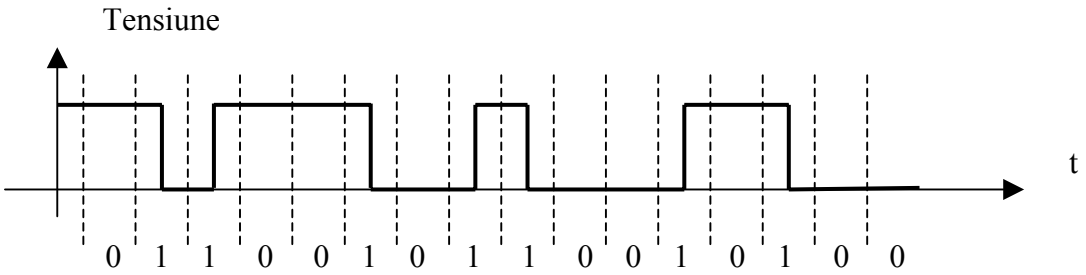


Figura 5.3. Codarea NRZI

C.Codarea Manchester

Codarea Manchester a fost realizată prima oară la Universitatea din Manchester cu ocazia construirii calculatorului Mark 1 (numit şi Baby) în 1948.

Regula de codare este:

1. Zero se codifică cu un front descrescător;
2. Unu se codifică cu un front crescător.

În figura 5.4. 8 biți într-o succesiune aleasă la întâmplare au fost codificați 8 biți cu 12 tranziții (1,5 tranziții pe bit). Dacă apare un şir de 0-uri sau 1-uri semnalul electric are variații. Codarea Manchester se foloseşte la rețeaua Ethernet.

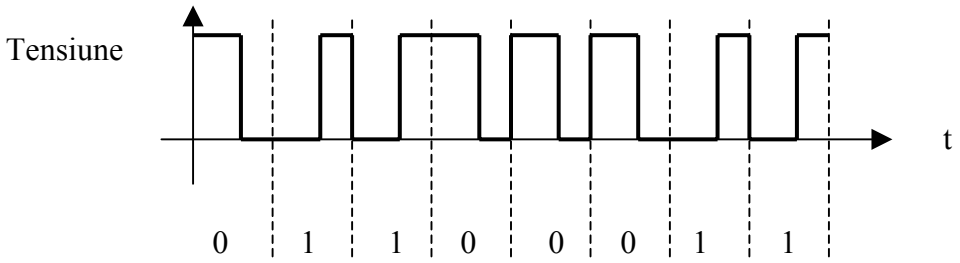


Figura 5.4. Codarea Manchester

D. Codarea FM (Frequency Modulation)

Codarea FM şi MFM (Modified FM) au fost introduse de IBM în anii 1970 pentru codificarea datelor la scrierea pe hard discuri. La aceste codări fiecare celulă bit începe cu un impuls de tact, iar regula de codare este:

1. Zero se codifică cu lipsa unui impuls.
2. Unu se codifică cu un impuls.

În figura 5.5. 8 biți într-o succesiune aleasă la întâmplare au fost codificați cu 24 tranziții (3 tranziții pe bit). Codarea FM s-a folosit la înscrierea datelor pe suporturi magnetice, în prezent fiind înlocuită cu codări mai eficiente.

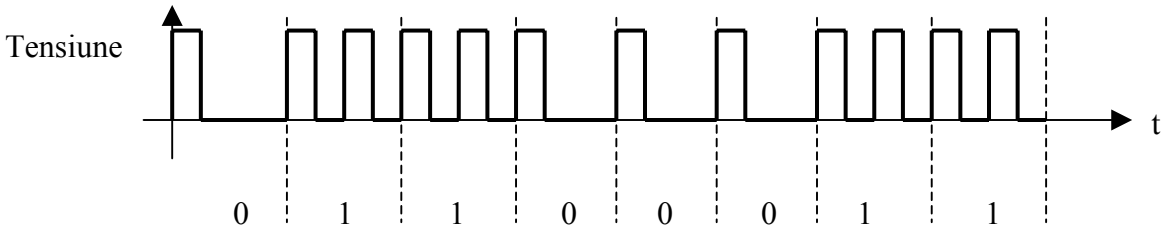


Figura 5.5. Codarea FM

Un tabel centralizator care cuprinde exemplele prezentate este dat în continuare:

Tabelul 5.1.

Codare	**Eficiență (biți/tranziție)**
NRZ	2.28
NRZI	2.28
Manchester	0.66
FM	0.33

Concluzie în această etapă:

Limita unui canal de transmisii de date seriale este dată de numărul de tranziții pe secundă. Interesul este să se trimită cât mai mulți biți cu cât mai puține tranziții. Deci la codurile NRZ şi NRZI în exemplele alese au fost codificați în medie 2,28 biți cu o tranziție, iar la codurile FM şi Manchester eficiența a fost mult redusă, doar 0,33 respectiv 0,66 biți cu o tranziție. Concluzia este că NRZ şi NRZI sunt cele mai eficiente din acest punct de vedere.

În figura 5.1 c se poate vedea că tactul de citire este refăcut din datele citite cu o buclă PLL. Dacă o codare permite un număr mare de valori logice de acelaşi fel succesive care nu produc nicio tranziție, bucla PLL pierde sincronizarea şi transmisia nu poate fi efectuată. Prin urmare, la codurile NRZ şi NRZI tactul nu se poate reface din datele citite şi codurile nu pot fi utilizate în modul de transfer cu refacerea tactului din date. Aceste coduri se numesc din acest motiv coduri neautosincronizabile. Codurile FM şi Manchester, chiar dacă asigură o codare mai puțin eficientă sunt coduri autosincronizabile. Tabelul 5.1 poate fi completat astfel:

Tabelul 5.2

Codare	Eficiență (biți/tranziție)	Autosincronizabilitate
NRZ	2.28	NU
NRZI	2.28	NU
Manchester	0.66	DA
FM	0.33	DA

Cercetările orientate spre găsirea unor noi metode de codificare eficientă şi autosincronizabile au fost îndreptate către modificarea NRZ sau NRZI care au o eficiență ridicată pentru a deveni autosincronizabile. Astfel au apărut codificările numite de grup, sau cu adăugare de biți (engl. bit stuffing). Aceste codificări impun ca la transmisia codată NRZ sau NRZI dacă apare un semnal codat cu mai multe celule bit fără tranziții decât o anumită limită, emițătorul forțează o tranziție, deci se adaugă un bit suplimentar.

În figura 5.6. se poate observa sus un semnal codat NRZ cu o succesiune de 6 valori de 1 logic succesive. Dacă codarea de grup admite doar 4 valori succesive de 1 logic, atunci emițătorul inserează un bit (cu roşu) suplimentar care generează 2 tranziții.

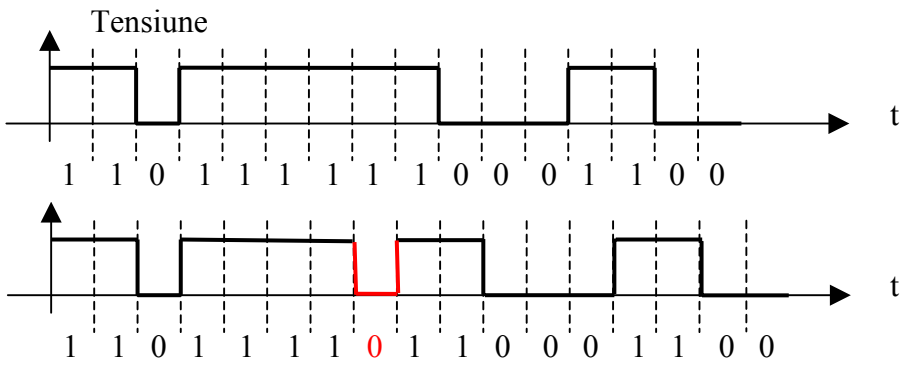


Figura 5.6. Codarea de grup

În această situație 16 biți de date au fost codificați NRZ cu 5 tranziții (3,2 biți cu o tranziție) iar după inserarea bitului suplimentar cei 16 biți sunt codificați cu 7 tranziții, adică 2,28 biți codificați cu o tranziție. Astfel se păstrează aproape la fel de mare eficiența codului NRZ şi codarea a devenit autosincronizabilă.

Vor fi descries în detaliu exemple de astfel de codări în capitolele următoare. Codarea 8B10B va fi descrisă la interfața serial IEEE1394 (Capitolul 11), codarea EFM la înregistrarea pe suporturi optice şi codarea RLL la scrierea pe suporturi magnetice (Capitolul 12).

5.3.Transmisii seriale asincrone şi sincrone

Al doilea pas în transmisia informației este asocierea biunivocă a unui caracter cu o configurație binară. Cel mai utilizat este codul ASCII. În codul ASCII (American Standard Code for Information Interchange) fiecare literă este reprezentată de un număr. De exemplu, litera **A** este reprezentată prin numărul **65**, în timp ce pentru litera **z** este alocat numărul **122** în zecimal. În codul ASCII fiecare caracter este codificat prin 7 biți. Semne speciale sunt de exemplu caracterul de comandă de revenire la începutul liniei CR (Carriage Return) şi avans la o nouă linie (Line Feed) care împreună au efectul apăsării tastei Return.

Formatul, integritatea şi dispunerea în timp a schimbului de date între procese care comunică între ele, fiecare desfăşurându-se independent în sisteme de calcul interconectate, se stabilesc printr-un set de reguli şi convenții specificate de un **protocol de comunicație**.

Un grup de cuvinte binare formează un cadru (bloc) de date. Protocoalele seriale pot fi orientate pe cuvânt sau pe cadru. Un cuvânt respectiv un bloc de date reprezintă entități de informație care se tratează unitar la receptor, adică sunt acceptate sau respinse în

întregime. În transferul asincron (RS232, LIN) se transmit cuvinte şi informația de sincronizare este ataşată cuvântului dar în cele mai multe tipuri de transfer (USB, CAN, IEEE1394b, Ethernet, Bluetooth, ZigBee) informația de sincronizare este ataşată cadrului, transmisiile find numite sincrone. La un cuvânt respectiv la un bloc de date se adaugă şi informație de verificare a corectitudinii transferului (bit de paritate sau cuvinte de control).

În figura 5.7. este prezentată o diagramă simplificată a transmiterii cuvântului CURS.

Se poate observa că informația de sincronizare (marcată cu roşu) este ataşată blocului de date la transmisia sincronă şi fiecărui cuvânt la transmisia asincronă.

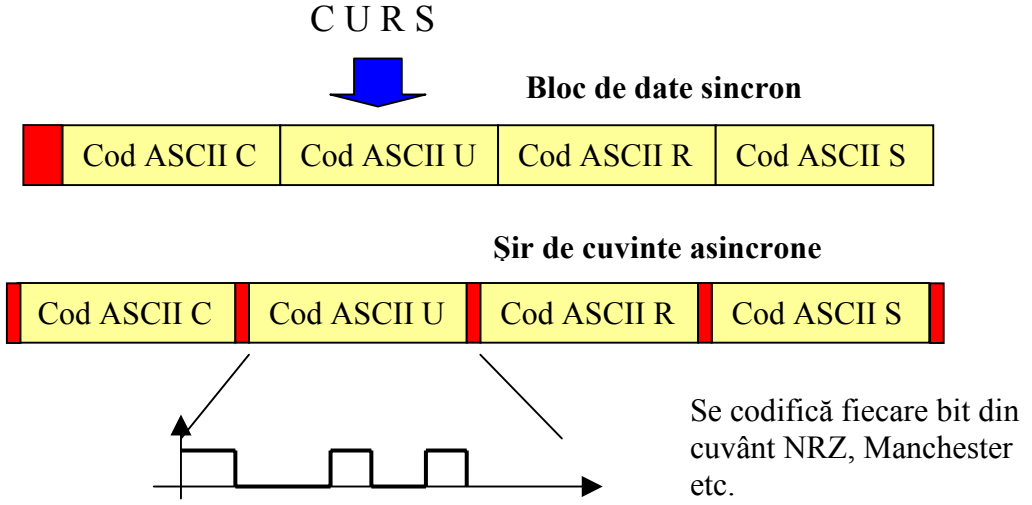


Figura 5.7. Cuvântul CURS transmis ca bloc de date sincron (sus) şi asincron (jos)

A.Structura unui cuvânt serial asincron

Linia de transmisie este în repaus în starea MARK (1 logic). Emisia unui caracter este precedată de trecerea liniei în starea SPACE (0 logic) pe durata unui bit numit de START, figura 5.8.

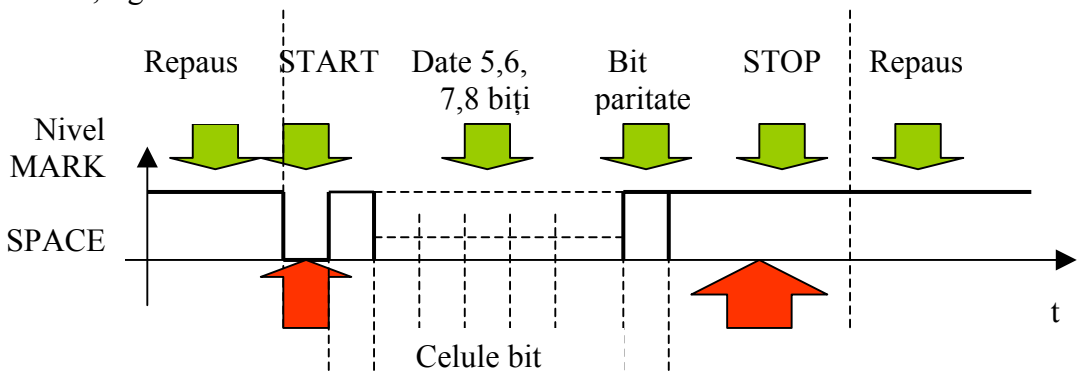


Figura 5.8. Structura unui cuvânt serial asincron

După bitul de START urmează 5, 6, 7 sau 8 biţi de date cu codare NRZ. Codarea NRZ nu este autosincronizabilă, prin urmare transmisia serială asincronă poate fi cu tact standard sau cu transmisia tactului. Transmisia se numeşte asincronă pentru că emisia unui caracter poate avea loc în orice moment, dacă linia este în repaus, în stare MARK. Biţii în cuvânt sunt transmişi sincron, cu un tact stadard sau cu un tact transmis, fiecare bit fiind codificat cu un nivel logic într-o celulă bit. După biţii de date urmează un bit de paritate pentru verificarea corectitudinii mesajului la receptor. După bitul de paritate urmează un bit, un bit şi jumătate sau doi biţi de STOP, în stare MARK, după care linia reintră în repaus.

În figura 5.8. a fost marcată cu săgeţi roşii informaţia de sincronizare, biţii de START şi STOP. Această informaţie este în cel mai favorabil caz 2 biţi, care la 8 biţi transmişi în cuvânt înseamnă 25%. Informaţia suplimentară transmisă în cazul acesta este 25%.

B. Structura unui cadru de date sincron

Un cadru de date este format din mai multe cuvinte transmise serial. Informaţia de sincronizare este transmisă la începutul cadrului (blocului) de date, figura 5.9.

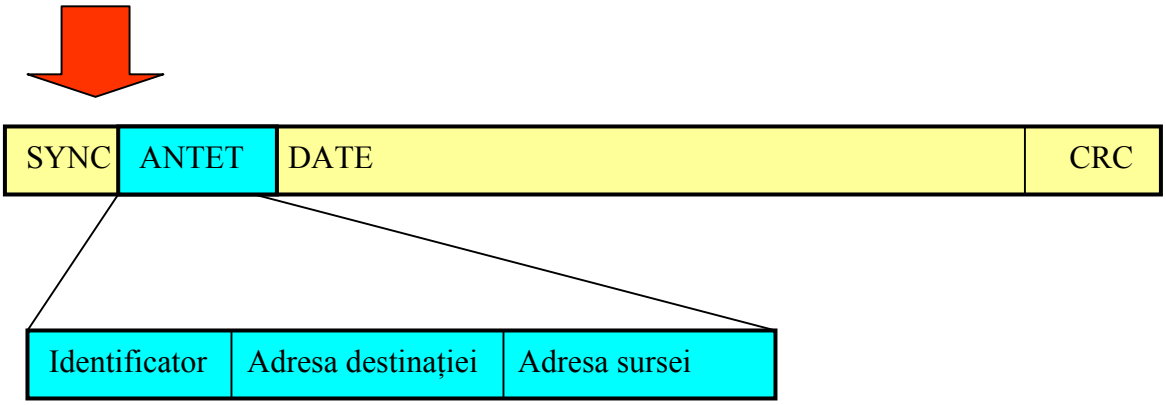


Figura 5.9. Structura unui cadru de date tipic

La transmisia sincronă este posibilă transmisia datelor cu refacerea tactului la receptor din datele transmise. De aceea, un cadru de date începe cu o zonă de sincronizare care conţine tranziţii dese, pentru a permite buclei PLL din receptor să se sincronizeze. Urmează apoi de regulă un antet format dintr-un identificator care depinde de tipul transmisiei, adresa destinaţiei şi adresa sursei. Zona de date are cea mai mare dimensiune şi este urmată de cuvinte CRC (Cyclic Redundancy Code) pentru verificarea corectitudinii transmisiei. Presupunând că zona de date este formată din 100 de cuvinte

iar zona de sincronizare şi antetul au 10 cuvinte se poate vedea că surplusul de informație (overhead bits) este doar 10%, mai mic decât în transmisia asincronă.

Structuri particulare de cadre de date vor fi prezentate la transmisia Ethernet (în capitolul 8), la USB (Capitolul 9), la IEEE 1394 şi SATA (Capitolul 11) etc.

5.4. Standardul RS232

EIA (*Electronics Industries Association*) împreună cu TIA (*Telecommunication Industry Association*) au realizat standarde pentru interfețe seriale. Standardul utilizat în prezent pentru conectarea microcontrollerelor la un calculator gazdă este EIA/TIA RS232-E. Valorile tensiunilor admisibile pentru fiecare nivel logic sunt date în figura 5.10. Valorile maxime admisibile sunt +25V şi –25V.

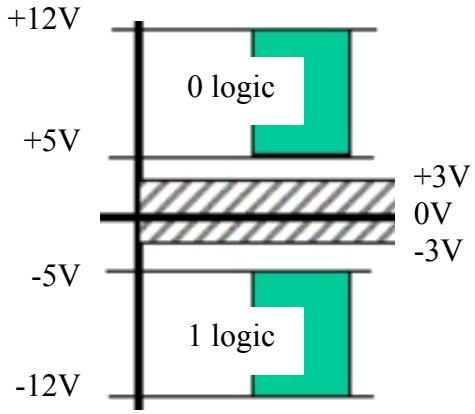


Figura 5.10. Nivelele de tensiune ale valorilor logice în RS232

Nivelele mai mari de tensiune asigură o margine de zgomot mai mare şi prin urmare o imunitate mai mare la perturbații. Distanța maximă de transfer este de 15m la un debit de informație de 115Kbps.

Cele mai importante semnale de interfață în standardul RS232 sunt:

Tabelul 1

Semnal	**Sens**	**Semnificație**
RxD	←	recepție date
TxD	→	emisie date
RxC	←	ceas de recepție
TxC	←	ceas de emisie

RTS	→	Request to Send (cerere de emisie)
CTS	←	Clear to Send (gata de emisie)
DTR	→	Data Terminal Ready (terminal de date gata?)
DSR	←	Data Set Ready (terminal de date gata)

Termenii Half Duplex (HDX) şi Full Duplex (FDX) se referă la simultaneitatea transferului de date în ambele sensuri între două echipamente. Dacă există flux de date atât într-un sens cât şi în celălalt, dar nu simultan, atunci transferul este HDX. Dacă există flux de date într-un sens şi în celălalt în acelaşi timp, atunci transferul este FDX. În industrie se foloseşte pentru HDX termenul TWA (Two Way Alternate) şi pentru FDX termenul TWS (Two Way Simultaneous).

O legătură punct la punct conectează două dispozitive, iar o legătură multipunct conectează mai mult de două dispozitive. Aceste legături sunt ilustrate în frigura 5.11. În legătura punct la punct, unul din dispozitive este emiţător şi celălalt este receptor (stânga). În legătura multipunct un singur dispozitiv este emiţător şi dintre celelalte, la un moment dat, unul sau mai multe dispozitive sunt receptoare (dreapta). Receptorul este activat printr-un mecanism de adresare specific interfeţei.

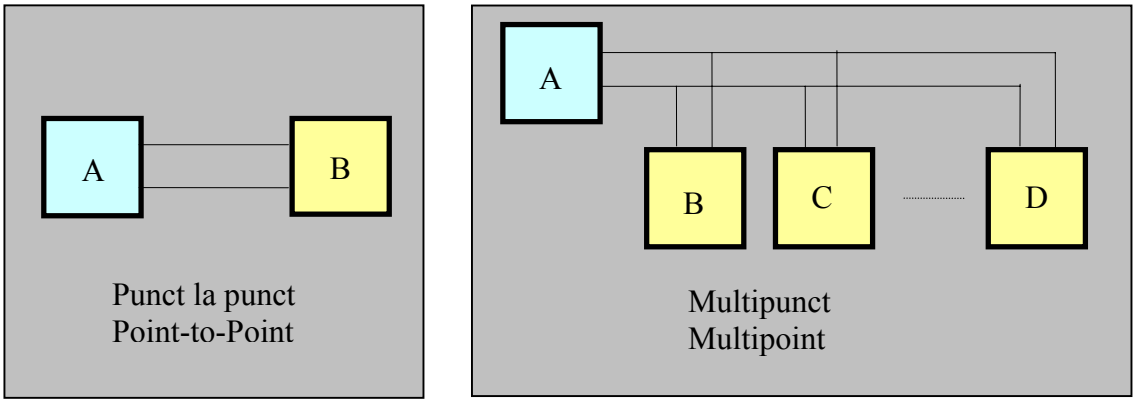


Figura 5.11.Legătură punct la punct şi multipunct

Din punctul de vedere al acestor precizări, interfaţa serială RS232 este o interfaţă punct la punct, Full Duplex.

O legătură completă RS232 între două sisteme de calcul utilizează toate semnalele principale din Tabelul 1 şi este dată în figura 5.12.

Datele circulă prin liniile TxD şi RxD în mod FDX sau HDX, depinde de capacitatea circuitelor seriale de interfaţă. Perechile de semnale RTS /CTS şi DTR /DSR au rolul de a

implementa un protocol hardware de comunicaţie. Unul dintre sistemele conectate solicită un transfer prin semnalul RTS sau DTR, iar celălalt prin CTS sau DSR confirmă disponibilitatea receptorului de a primi date. Tactul de recepţie şi cel de emisie pot fi diferite dare de cele mai multe ori ele sunt egale şi provin de la un genrator extern. Transmisia serială RS232 poate fi cu tact standard şi în acest caz cele două generatoare de tact generează un tact precizat în standard şi în foile de catalog a circuitelor. Dacă tactul este generat la un singur sistem şi este transmis prin linia de transmisie se obţine o transmisie serială cu transmiterea tactului, care poate asigura un debit mai mare de informaţie. În transmisia RS232 cele două sisteme trebuie să aibă masă comună, deci conexiunea trebuie completată cu o linie de masă (GND).

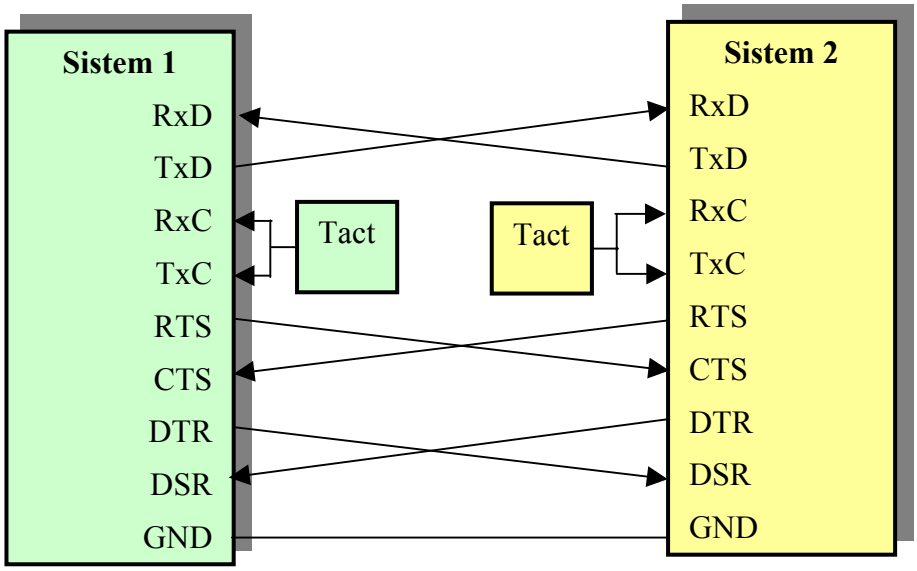


Figura 5.12. Legătură RS232 completă

Dintre cele două perechi de linii de protocol RTS /CTS şi DTR /DSR se poate folosi doar una, scăzând astfel numărul de linii de conexiune de la 7 la 5. Protocolul hardware de comunicaţie se numeşte DTR sau RTS în funcţie de perechea utilizată. Pentru transfer poate fi implementat şi un protocol software care se numeşte XonXoff. Acest protocol înseamnă transmisia pe linia TxD şi recepţia pe RxD a unor coduri, unul care semnifică "liber" (Xon) şi unul care semnifică "ocupat" pentru Xoff. În această situaţie numărul de linii de conexiune scade la 3, figura 5.13.

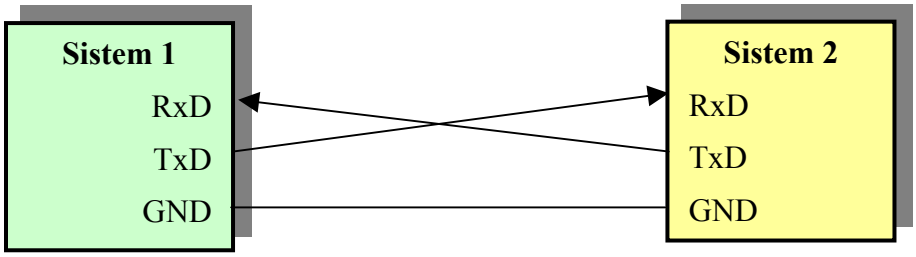


Figura 5.13. Transmisia RS232 pe 3 fire cu protocol software

Tactul de transmisie este standard dar evoluția echipamentelor a necesitat mărirea vitezei de comunicație. De aceea s-au definit viteze standard de comunicație iar circuitele de interfață serială prelucrează prin divizare tactul standard pentru a obține mai multe viteze de comunicație standard, selectabile software. Vitezele de comunicație standard sunt (în Baud): 300, 600, 1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200.

5.5.Circuit de interfață programabil

A.Descrierea circuitului

Ca exemplu de circuit programabil de interfață serială RS232 a fost ales circuitul Intel 8251, care a fost primul circuit conceput în acest scop de Intel. Circuitul se numeşte circuit USART (Universal Syncronous Asyncronous Receiver Transmitter) pentru că poate lucra atât în mod asincron cât şi sincron. Dacă la interfețele paralele primul circuit programabil realizat de Intel a fost utilizat o lungă perioadă de timp, circuitul de interfață serială I8251 a fost înlocuit după scurt timp de interfețe mai evoluate. Alegerea acestui circuit s-a datorat valorii educative, pentru că realizează atât mod asincron cât şi sincron spre deosebire de circuitele mai noi care realizează doar modul asincron al interfeței RS232. Un alt motiv este simplitatea circuitului I8251 şi modul simplu de programare.

Schema bloc a circuitului este dată în figura 5.14.

Semnalele de interfață cu microprocesorul gazdă au aceeaşi semnificație ca la circuitul de interfață paralelă:

- RD se execută cu ciclu de citire de la un port sau de la memorie. Acest pin se conectează la semnalul de magistrală IOR;
- WR se execută cu ciclu de scriere la un port sau în memorie. Acest pin se conectează la semnalul de magistrală IOW;

- C/D, Comenzi / Date, cu 0 arată transfer de date, cu 1 transfer de comenzi. Se conectează de regulă la linia cel mai puțin semnificativă de adresă;
- CLK tact de magistrală pentru comanda operațiilor interne în interfață;
- RESET este o linie care comandă inițializarea circuitului prin ştergerea informației din toate registrele;
- CS este o linie care selectează circuitul. Formării acestui semnal i se dedică un capitol ulterior;
- D0-D7 magistrala de date a gazdei, 8 linii bidirecționale;

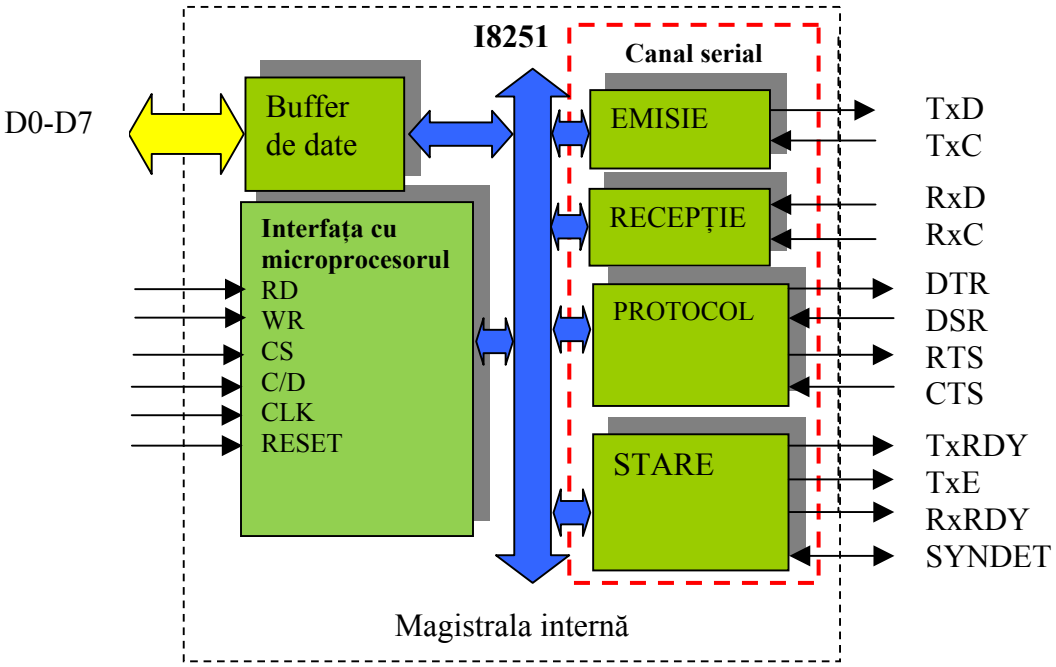


Figura 5.14. Schema bloc a circuitului programabil de interfață serială I8251

Semnalele de conectare cu linia RS232 sunt cele standard, descrise la 5b.1. În plus față de acestea există câteva semnale care au rolul de a informa procesorul gazdă de starea circuitului. Aceste semnale sunt:

- TxRDY transmițător pregătit pentru a primi date de pe magistrala D0-D7;
- RxRDY receptorul a recepționat un caracter care este gata de a fi trimis pe magistrala D0-D7;
- TxE transmițător gol;
- SYNDET un semnal cu sens dublu, interfața anunță că a detectat sincronizarea sau este anunțată că un circuit extern a detectat sincronizarea.

Aceste semnale pot fi utilizate ca cereri de întrerupere pentru procesor.

B.Funcționarea circuitului

Circuitul are ca şi funcție principală conversia paralel serie şi serie paralel. Un cuvânt care trebuie transmis paralel este scris în circuit în format paralel de către procesor pe liniile D0-D7, este serializat şi în mod asincron i se adaugă biți de START, STOP şi de paritate. Serializarea se face cu tactul TxC. Este transmis apoi serial pe linia TxD, cu un protocol hardware activând şi citind perechile RTS/CTS, DTR/DSR sau un protocol software XonXoff. După transmisia cuvântului se activează semnalul TxRDY pentru ca procesorul să poată trimite alt cuvânt.

La recepția în mod asincron informația serială care vine pe linia RxD este citită serial cu tactul RxC începând cu primul front care este recunoscut ca bit de START. Se verifică paritatea, apoi sunt eliminați biții de START şi STOP şi se face conversia serie paralel. Circuitul activează semnalul RxRDY pentru ca procesorul să ştie că poate citi caracterul în format paralel pe liniile D0-D7.

În mod sincron la recepție circuitul poate lucra în două moduri, cu sincronizare internă sau cu sincronizare externă. Cu sincronizare internă la recepție citeşte continuu datele de pe linia RxD şi le compară cu un cuvânt de 8 biți de sincronizare. Acest cuvânt este standardizat, este folosit la cadre de date cu codare ASCII şi marchează începutul unui cadru de date. Acest mod de funcționare în care circuitul urmăreşte sincronizarea se numeşte mod *Hunt*. În acest mod circuitul nu transmite date spre procesor. În momentul recunoaşterii cuvântului de sincronizare, datele sunt citite serial de pe linia RxD, sunt transformate în format paralel, se verifică bitul de paritate şi sunt transmise spre procesor pe liniile D0-D7. La detectarea sincronizării se activează semnalul SYNDET. Semnalul RxRDY se activează la fiecare cuvânt recepționat.

În mod sincron la recepția cu sincronizare externă, un circuit extern se ocupă cu recunoaşterea cuvântului de sincronizare. Când este recunoscut un astfel de cuvânt se activează către circuitul de interfață serială semnalul SYNDET şi circuitul începe să citească serial date de pe linia RxD, sunt transformate în format paralel, se verifică bitul de paritate şi sunt transmise spre procesor pe liniile D0-D7.

În mod sincron la transmisie un cadru de date este transferat octet după octet în format paralel pe liniile D0-D7 către circuit, circuitul inserează cuvântul de sincronizare, calculează şi inserează bitul de paritate, serializează cuvintele şi le transmite pe linia TxD. Când a terminat de trimis un cuvânt semnalizează procesorului prin activarea semnalului TxRDY.

În mod sincron cuvântul de sincronizare poate avea un octet sau doi octeți.

C.Programarea circuitului

Primul cuvânt de comandă trimis în circuit cu linia C/D=1 comandă modul de lucru şi principalii parametri de funcționare, astfel:

Dacă D1=0 şi D0=0 modul de lucru este sincron, la orice altă combinație modul este asincron.

Semnificația biților din primul cuvânt de comandă în mod asincron

D7	D6	biți de STOP
0	0	invalid
0	1	1 bit
1	0	1 ½ bit
1	1	2 biți

D5	D4	paritate
X	0	dezactivat
0	1	paritate pară
1	1	paritate impară

D3	D2	lungime caracter
0	0	5 biți
0	1	6 biți
1	0	7 biți
1	1	8 biți

D1	D0	Tact de transmisie
0	0	invalid
0	1	tact/64
1	0	tact/16
1	1	tact

Semnificația biților din primul cuvânt de comandă în mod sincron

D7=0 cuvântul de sincronizare are doi octeți, D7=1 cuvântul de sincronizare are 1 octet

D6=0 sincronizare internă, D6=1 sincronizare externă

D5	D4	paritate
X	0	dezactivat
0	1	paritate pară
1	1	paritate impară

D3	D2	lungime caracter
0	0	5 biți
0	1	6 biți
1	0	7 biți
1	1	8 biți

D1=0, D0=0 pentru selectarea modului sincron

Dacă s-a selectat modul sincron cu sincronizare externă atunci după primul cuvânt de comandă nu mai urmează alt cuvânt de programare specific modului sincron. Dacă s-a selectat modul sincron cu sincronizare internă cu un octet de sincronizare atunci urmează un al doilea cuvânt de programare care conține octetul de sincronizare. Dacă s-a selectat modul sincron cu sincronizare internă cu doi octeți de sincronizare atunci urmează al doilea şi al treilea cuvânt care conțin aceşti doi octeți.

Al doilea cuvânt de comandă şi cuvântul de stare

Structura cuvântului al doilea de comandă este dată în stânga şi a cuvântului de stare (citit cu semnalul C/D=1) este dată în dreapta.

Al doilea cuvânt de comandă		**Cuvânt de stare**	
Bit	**Semnificație**	**Bit**	**Semnificație**
D7	Intrare în mod *Hunt*	D7	DSR
D6	Reset intern	D6	SYNDET
D5	RTS	D5	FE (Frame Error)
D4	Anularea tuturor erorilor din registrul de stare	D4	OE (Overrun Error)
D3	Transmisie caracter BREAK	D3	PE (Parity Error)
D2	Activare recepție	D2	TxE
D1	DTR	D1	RxRDY
D0	Activare emisie	D0	TxRDY

Primul cuvânt de comandă programează parametric comunicației, iar al doilea cuvânt de comandă declanşează execuția unei operații: transmisia, recepție, modul de detectare a sincronizării etc. Transmisia unui caracter BREAK înseamnă că linia de transmisie este trecută în stare SPACE şi nu mai este posibilă transmisia informațiilor pe linie. O utilizare posibilă a acestui caracter BREAK este la detectarea automată a vitezei de transmisie. Lungimea caracterului, prin urmare durata menținerii liniei în stare SPACE poate da informații receptorului despre viteza de transmisie, pentru ca acesta să-şi poată adapta tactul de recepție. Cu al doilea cuvânt de comandă se pot genera semnalele de protocol DTR şi RTS dacă protocolul folosit este cel hardware. Dacă se foloseşte protocolul software aceste linii nefolosite se pot utiliza de exemplu pentru semnalizarea stării circuitului.

Cuvântul de stare conține starea liniilor TxE, RxRDY,TxRDY şi SYNDET pentru a putea fi citite de procesor în cazul în care transferul este programat. În cuvântul de stare se poate citi semnalul de protocol DSR şi se pot identifica erorile care au apărut la transmisie: eroarea de paritate PE când bitul de paritate transmis nu corespunde cu cel generat la receptor, eroarea de sincronizare FE când nu s-a recepționat un bit de STOP corespunzător şi eroarea de suprascriere (OE) când s-a recepționat un cuvânt înainte ca cel anterior să fie citit.

5.6. Modificarea nivelului de tensiune

În figura 5.10 se arată că nivelele de tensiune RS232 sunt între 3 şi 15V, iar din figura 5.14 se vede că circuitul interfață serială nu are circuite de modificare de nivel şi fiind în tehnologie TTL are nivele între 0 şi 5V. Prin urmare, semnalele generate şi recepționate de circuitul interfață serială trebuie convertite ca nivel în semnale RS232. Pentru aceasta, cel mai cunoscut şi utilizat circuit este MAX232. De cele mai multe ori, modulele echipate cu interfață serială sunt alimentate doar de la 5V, aşa încât circuitul MAX232 conține în interior două convertoare DC/DC pentru a forma +12V şi -12V din 5V. Schema bloc internă a circuitului MAX232 este dată în figura 5.15 (stânga):

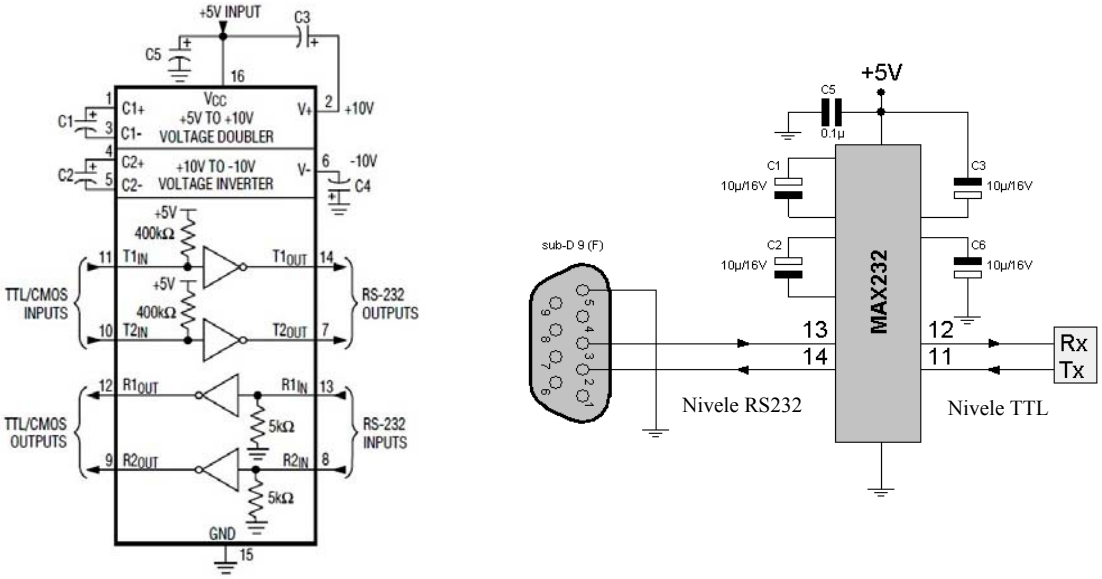


Figura 5.15. Schema bloc internă a circuitului MAX232

În figura 5.15 în dreapta se poate vedea o schemă electric tipică de utilizare a unui circuit MAX232 în cazul unei legături seriale RS232 cu protocol software, la care comunicația se face pe două fire TxD şi RxD, şi GND.

În figura 5.16. se poate vedea o diagramă de timp vizualizată cu un osciloscop pentru transmiterea serială a unui caracter. Jos este reprezentat semnalul la nivele TTL iar sus se observă semnalul cu amplitudine mai mare (RS232). Se poate observa că polaritatae la RS232 este inversată de MAX232.

Figura 5.16. Diagrama transferului serial RS232 a unui caracter (jos la nivel TTL, sus la nivel RS232)

O conexiune serială între un microcontroller şi un PC prin interfața serială este simplă, figura 5.17.

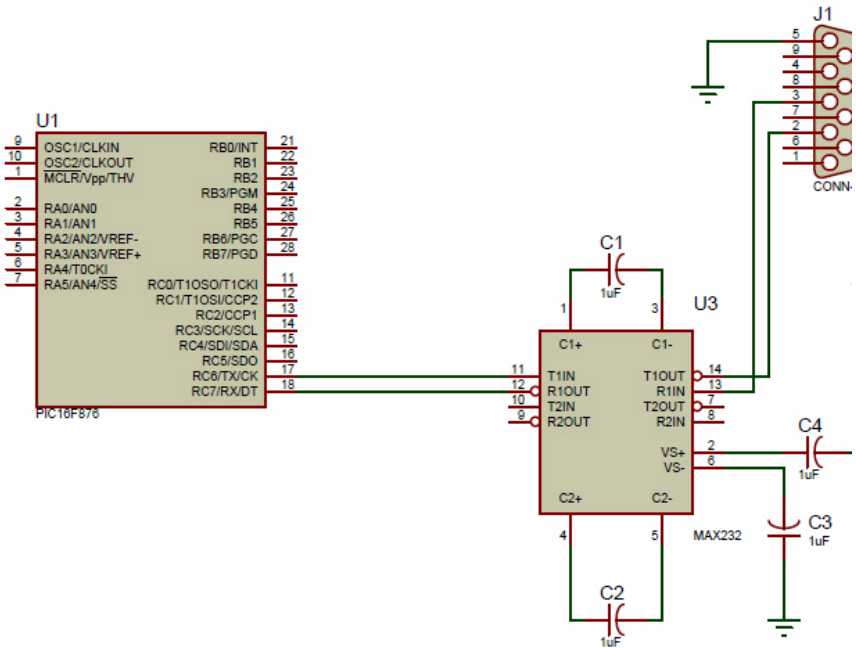


Figura 5.17. Conectarea unui microcontroller la un PC prin interfața serială cu MAX232, sursa http://creativeelectron.net/blog/2009/10/pic-serial-communition-in-pic-microcontroller-1/

6. Conectarea la un calculator pe magistrală şi la un port paralel

6.1. Selectarea unui dispozitiv pe magistrală

La un moment dat pe o magistrală sunt active 2 subsisteme, dintre care unul este Masterul. Iar celălalt este subsistemul Slave selectat de Master, figura 6.1:

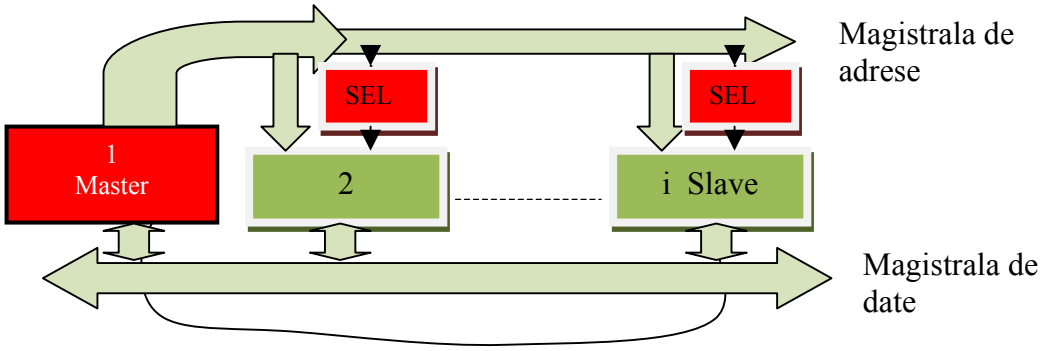


Figura 6.1. Selecția prin adrese

Transferul de date are loc între Master şi subsistemul i pe magistrala de date. Liniile de adresă necesare pentru adresarea locațiilor sau registrelor interne ale subsistemelor sunt conectate direct la acestea. Cu liniile rămase libere se selectează subsistemul cu care Masterul transferă date. Subsistemul selectat devine activ, celelalte subsisteme intră în înaltă impedanță la interfața cu magistrala. Selectarea poate fi:

1. Selectarea liniară se poate realiza dacă numărul liniilor de adresă rămase disponibile este mai mare sau egal cu numărul circuitelor care trebuie selectate. Fiecare circuit este selectat cu o linie de adresă, avantajul soluției fiind simplitatea iar dezavantajul fiind faptul că se pierde din spațiul de adresare;
2. Selectarea decodificată, în care liniile de adresă rămase libere sunt introduse într-un decodificator DCD, iar ieşirile DCD selectează fiecare un circuit. Se poate astfel folosi întregul spațiu de adresare;
3. Selectarea mixtă, în care unele linii de adresă libere selectează direct câte un circuit în timp ce alte linii selectează prin intermediul unui DCD.

În primul exemplu se propune conectarea pe o magistrală de adrese de 16 biți a unor circuite de memorie de 16k. Selectarea liniară şi harta memoriei sunt date în figura 6.2.

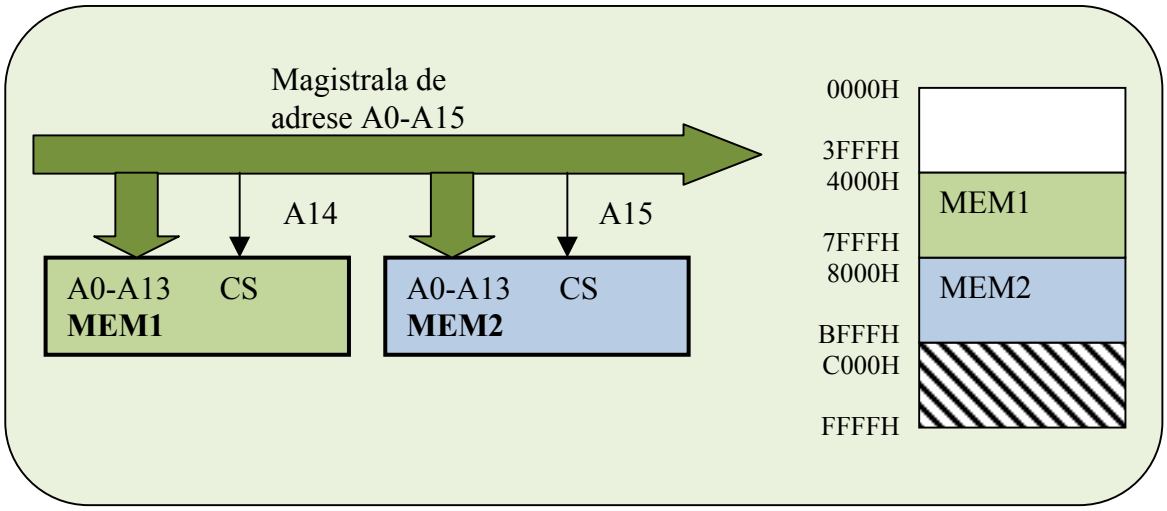


Figura 6.2. Selectarea liniară în cazul memoriilor (CS activ în 1)

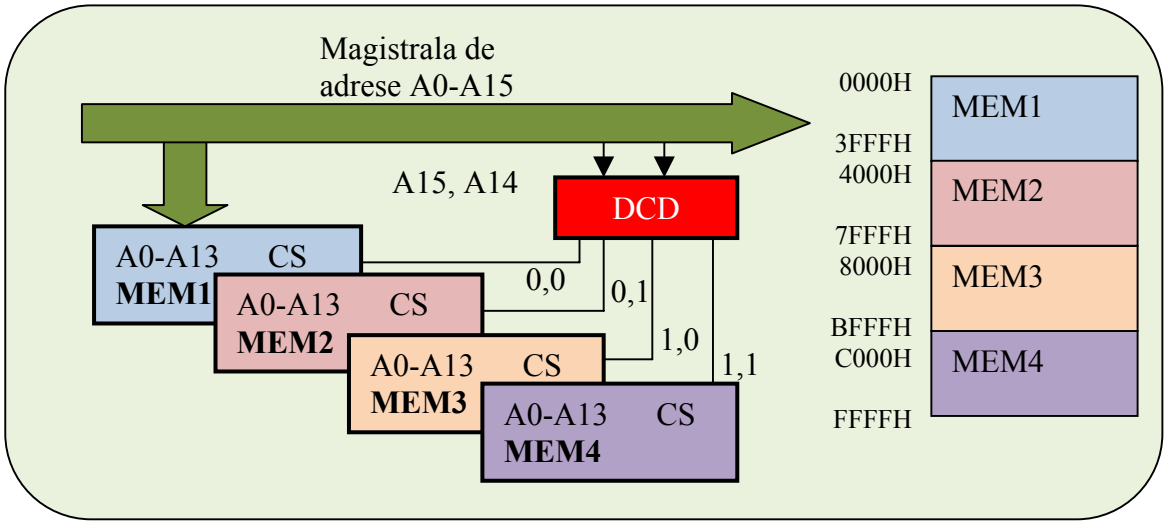


Figura 6.3. Selectarea decodificată în cazul memoriilor

Se poate observa că în cazul selectării decodificate (figura 6.3.) se foloseşte întreaga zonă de adresare. În cazul conectării unui dispozitiv de IO se aplică aceleaşi reguli, diferența fiind dată de faptul că dispozitivele de IO au mai puține linii de adresare pentru regiştrii interni, ca urmare rămân mai multe linii de adresă libere. În al doilea exemplu se conectează o interfață paralelă programabilă pe o magistrală de adrese de 16 biți. Interfața utilizează 2 linii de adresă pentru selectarea celor 3 regiştrii interni. Să presupunem că

adresa de bază la care dorim conectarea circuitului este 0180H. În figura 6.4. este dată schema bloc de conectare şi harta memoriei.

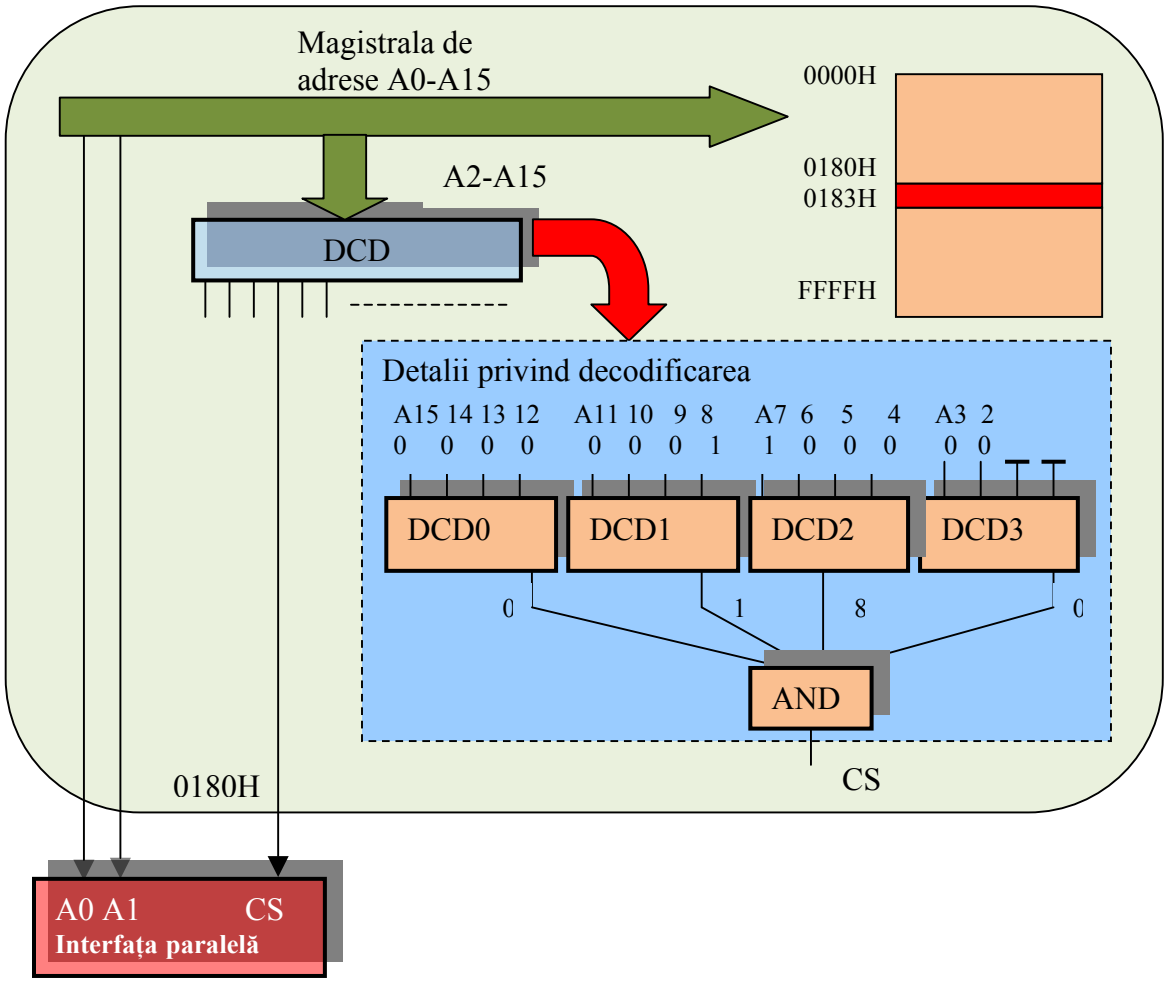


Figura 6.4. Conectarea unui circuit interfață paralelă pe magistrală

Selecția prin decodificare a fost realizată cu 4 DCD cu 4 intrări. Schema bloc este detaliată în partea de selecție. După programarea circuitului de interfață paralelă se pot trimite /recepționa date de un octet cu o singură instrucțiune sau două, dacă datele sunt extrase din memorie, ceea ce înseamnă, de exemplu la un microcontroller RISC cu instrucțiuni executate într-un singur tact o rată de transfer de ordinul de mărime a tactului. La MC complexe, aşa cum sunt de exemplu MC pe 32 de biți de la Fujitsu se pun la dispoziția utilizatorului semnale de CS pentru diferite zone de memorie liberă pentru a uşura implementarea aplicațiilor care necesită lucrul pe o magistrală externă.

De exemplu, considerând circuitul de interfaţă paralelă conectat printr-o magistrală simplă ca şi cea descrisă în capitolul de magistrale, programul de citire din memorie şi trimitere la un port de ieşire va fi:

```
        MOV BX, [adresa inițială de memorie]
        MOV DX, adresa portului de ieşire
Start:  MOV AL, [BX]
        OUT AL,DX
        INC BX
        JMP start
```

În buclă sunt o instrucţiune de citire din memorie (4 tacte), una de scriere la un port (5 tacte), una de salt şi una de incrementare a unui registru. Dacă presupunem că programul din buclă durează 20 de tacte şi frecvenţa magistralei este de 10MHz atunci viteza maximă de transfer poate fi 10MHz/20=500kHz. Dacă se transferă 8 biţi la o parcurgere a buclei, atunci debitul maxim de informaţie va fi de 500Ko/s.

6.2.Exemple de conectare pe magistrală la microcontrollere

Familia MCS 51 prevede magistrala externă chiar şi la modelele mai ieftine, aşa cum este de exemplu MC Atmel AT89LS51. Pe magistrala externă poate fi conectată memoria de date şi memoria de program, spaţiile de adresare fiind separate. Portul 0 poate fi configurat ca port de adrese şi date multiplexate pe 8 biţi astfel: P0.0 – AD0... P0.7 – AD7. Portul 2 generează adresele superioare pentru adresarea pe 16 biţi: P2.0 – A8.... P2.7 - A15. Semnalele de comandă generate sunt ALE, /RD, /WR şi /PSEN (Program Store Enable) care este un strob pentru memoria externă de program. O diagramă de timp de acces la magistrală în cazul unui transfer de citire cu memoria de date (/PSEN inactiv), figura 6.5.

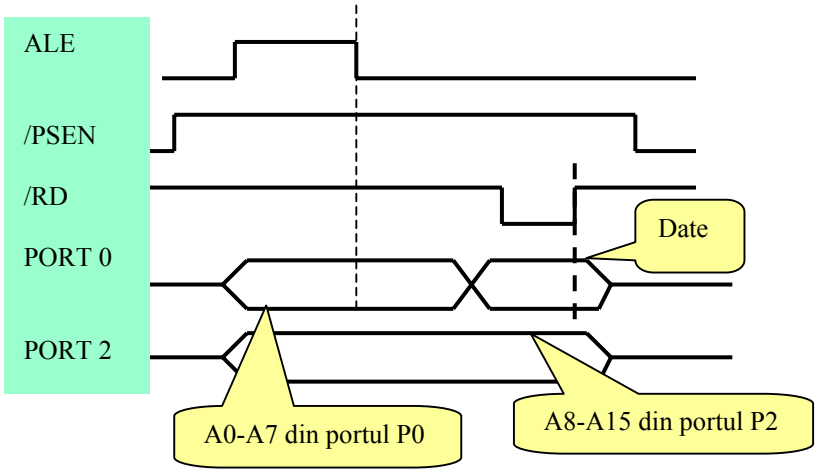


Figura 6.5. Diagrama de timp la un ciclu de citire dintr-o memorie externă de date

O aplicaţie de conectare la un MC din familia MCS 51 a unei memorii externe EPROM de 64kocteţi pe magistrală şi a unui afişaj LCD pe un port paralel este dată în figura 6.6:

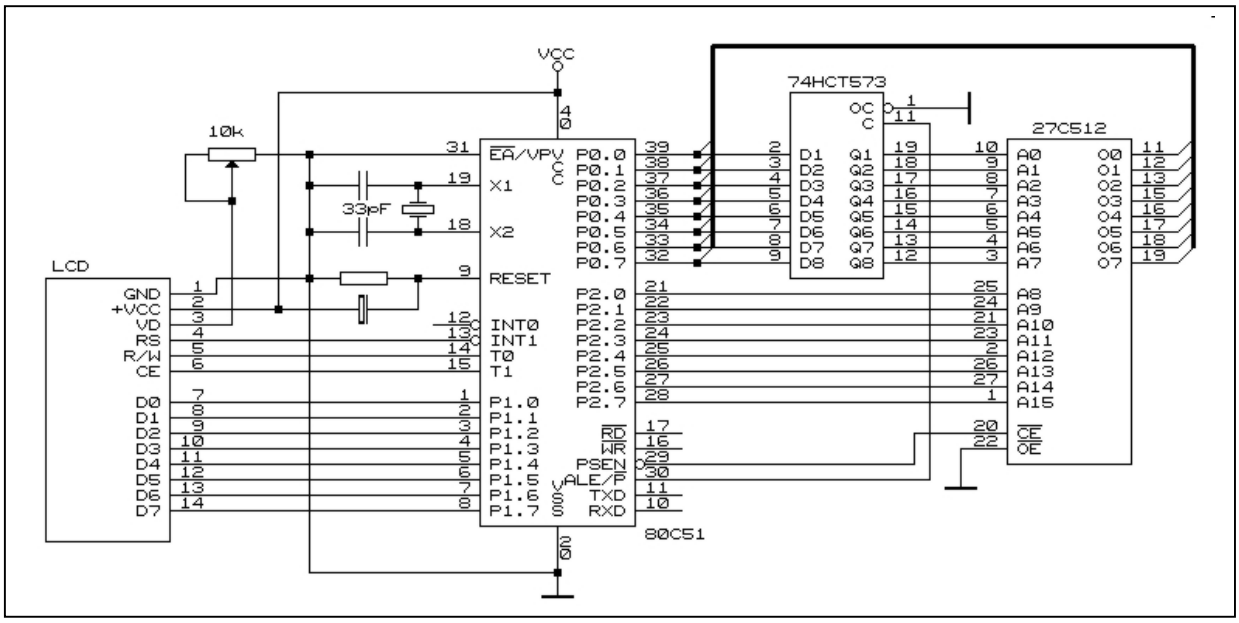


Figura 6.6. Conectarea unei memorii externe şi a unui LCD la un MC 80C51 (sursa www.ustr.net)

Pentru a adresa memoria EPROM se folosesc 16 biţi de adresă, cei mai puţin semnificativi la portul P0, cei mai semnificativi la portul P2. Un impuls pe linia ALE încarcă biţii de adresă din portul P0 în latch-ul 74HC573. Se activează PSEN pentru că în EPROM sunt stocate instrucţiuni (memorie externă de program), datele fiind pe liniile de date din portul P0 putând fi citite (/RD activ) sau scrise (/WR activ). Afişajul LCD este conectat cu liniile de date la portul P1 şi semnalele de comandă astfel: RS – datele transmise pot fi un caracter de afişat sau o comandă, RW – sensul de transfer al datelor, CE – comanda de afişare. Un exemplu de programare:

```
MOV P1,A        datele de afişat sunt trimise la portul P1
SETB RS         datele reprezintă un caracter de afişat
CLR RW          sensul este de scriere în afişaj
NOP             întârziere
SETB CE         comanda de afişare
NOP             întârziere
NOP
CLR CE          linia de comanda de afişare este readusă la 0
```

RET întoarcere în programul principal

Înainte de a se lucra cu afişajul LCD este nevoie de trimiterea unor date pentru inițializare, funcție de tipul afişajului.

MC AVR pe 8 biți Atmega64 admite lucrul cu magistrală externă pentru cuplarea unei memorii externe printr-o interfață internă specială numită XMEM. Magistrala externă este pe 8 biți de date şi 16 biți de adresă astfel: PA0- AD0... PA7-AD7, PC0-A8...PC7-A15, PG0-/WR, PG1-/RD, PG2-ALE. Datele sunt multiplexate cu adresele pe portul PA. O schemă bloc de conectare a unei memorii externe este dată în figura 6.7.

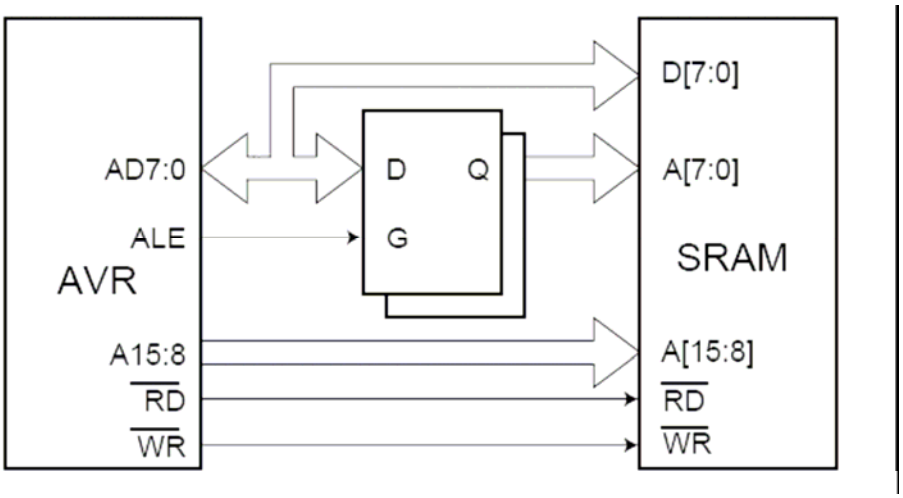


Figura 6.7. Schema bloc de conectare a unei memorii externe la un Atmega64

Interfața de magistrală XMEM va detecta un acces la memoria externă şi nu va ține cont de setările de direcție ale porturilor PA şi PC. Viteza de transfer a datelor este mare, de aceea se impun condiții de viteză pentru latch-ul de adrese. Pentru că memoriile au timpi diferiți de acces se pot defini 4 variante de lucru cu memoria externă prin inserarea a 0, 1 sau 2 stări de WAIT.

O diagramă de timp pentru accesul la memoria externă pe 16 biți, la citire este dată în figura 6.8.

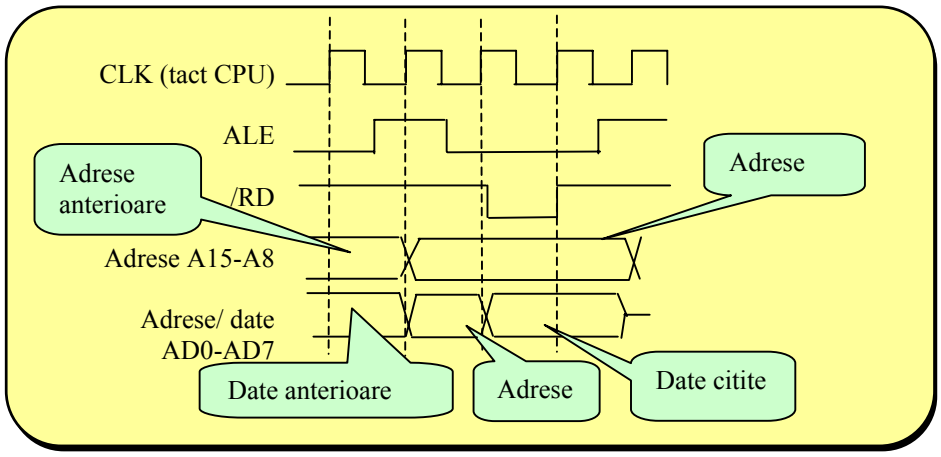


Figura 6.8. Diagrama de timp pentru un acces la citire

În cazul microcontrollerelor cu putere de calcul mai mare, cum este de exemplu la microcontrollerele Fujitsu de 32 de biți magistrala externă are mai multe linii. Lucrul cu magistrala externă este mai complicat, transferurile de date pot fi pe 8, 16 sau 32 de biți. În cazul MC din familia Fujitsu MB91F care sunt MC RISC, structura este complexă şi magistralele sunt de mai multe tipuri. Între CPU şi memorie magistrala este Harvard (magistrale diferite pentru date şi instrucțiuni) pe 32 de biți, legătura cu dispozitivele de I/O fiind realizată cu o magistrală pe 16 biți, iar în exterior MC este prevăzut cu o magistrală externă Von Neuman (magistrală comună pentru date şi instrucțiuni, aspecte care se vor detalia în Capitolul 7).

Caracteristicile magistralei externe sunt:

- Spațiul extern este adresabil cu 32 de biți fiind de 4GB.
- Se pot defini până la 8 bank-uri independente cu semnale de CS generate de MC. Bank-urile pot fi de minimum 64K şi pot ocupa orice poziție logică în aria externă.
- Magistrala de date poate fi pe 32/16/8 biți, lărgime care poate fi diferită pentru bank-uri diferite.
- Se pot transfera date prin DMA.

O schemă bloc a interfeței cu magistrala externă este dată în figura 6.9.

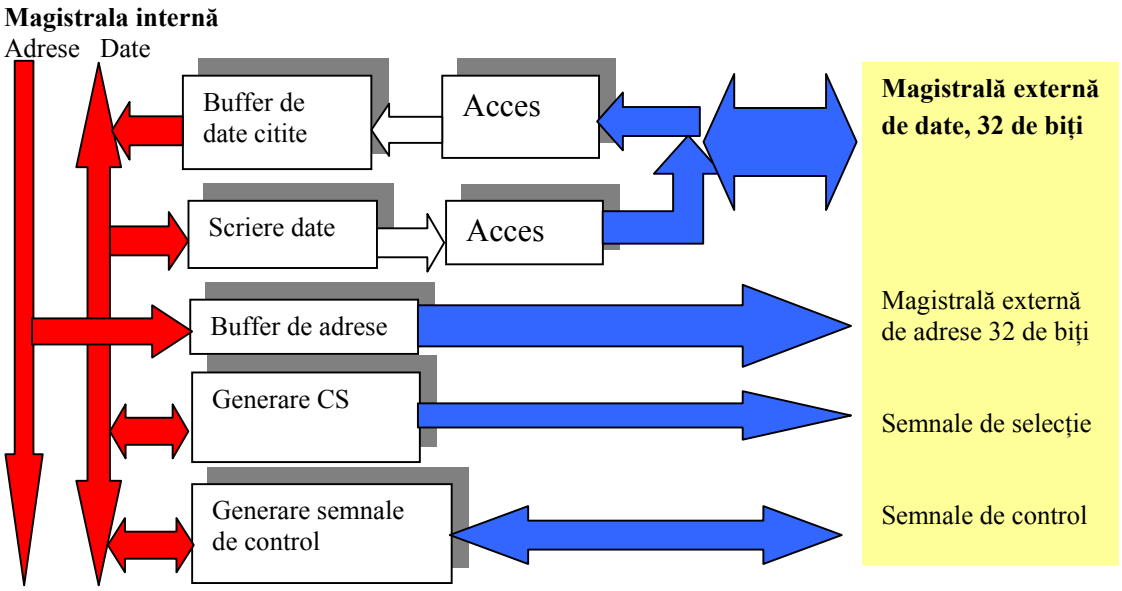


Figura 6.9. Interfața cu magistrala externă la microcontrollerele Fujitsu MB91F pe 32 biți

6.3. Conectarea pe porturi paralele

Conectarea unui dispozitiv (Echipament periferic EP) la un port paralel este cea mai simplă soluție, mai ales la microcontrollere. De multe ori o aplicație simplă implementată cu microcontroller foloseşte microcontrollere ieftine care nu au magistrală externă. În cazul în care transferul de date este pe 8 biți şi nu este un transfer cu protocol, atunci conectarea este banală, figura 6.10. stânga. Un exemplu este aplicația cu o felicitare care atunci când este deschisă cântă o melodie. Dacă este nevoie de un transfer cu protocol atunci se pot folosi linii dintr-un al doilea port al microcontrollerului, figura 6.10 dreapta. În cazul reprezentat transferul este bidirecțional şi pe liniile P1.0 şi P1.1 se generează semnalele de RD şi WR.

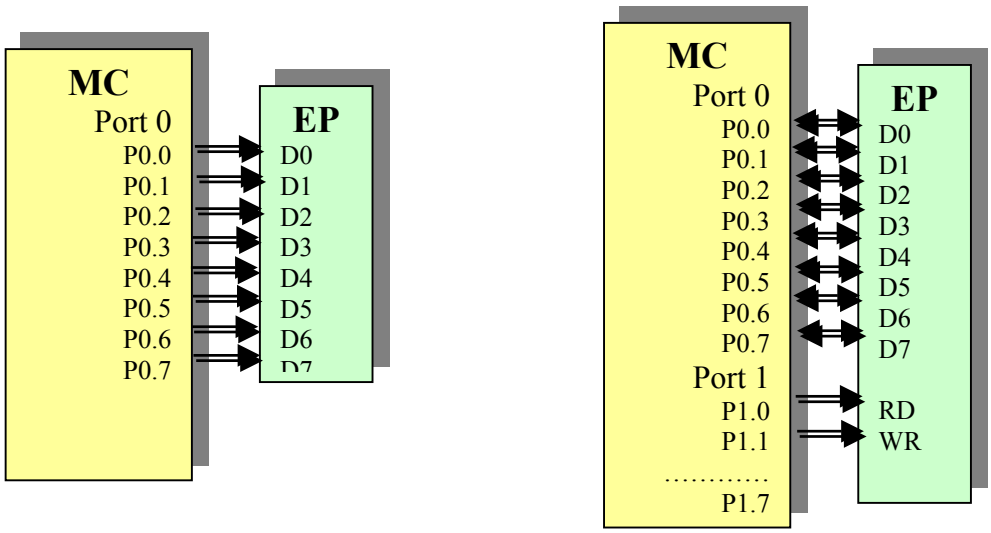


Figura 6.10. Conectarea la porturi paralele

La o conectare cu protocol se poate evalua viteza maximă de transfer prin scrierea unei secvenţe de program (pentru a putea compara cu viteza de transfer pe magistrală secvenţa este scrisă în limbajul x86).

```
        MOV BX, [adresa iniţială de memorie]
Start:  MOV DX, adresa portului de ieşire P0
        MOV AL, [BX]
        OUT AL,DX                    ↕  Se trimit date în portul P0
        INC BX
        MOV DX, adresa portului de ieşire P1
        MOV AL, 02                   ↕  WR este trecut în High
        OUT AL,DX
        MOV AL,00                    ↕  WR este trecut în Low
        OUT AL,DX
        JMP start
```

Se poate vedea că secvenţa de program din buclă este de 2,5 ori mai lungă decât la trimiterea datelor pe magistrală deoarece semnalul WR trebuie activat soft. Acest lucru înseamnă că viteza maximă de transfer în acest caz este de 2,5 ori mai mică, deci 200Ko/s.

Dacă interfața echipamentului periferic este pe 16 biți şi microcontrollerul are disponibile doar două porturi, dacă transferul are nevoie de linii de procol şi viteza de transfer necesară nu este mare se pot folosi metode de conectare care includ circuite suplimentare.

Astfel, dacă este nevoie de un transfer pe 16 biți de ieşire din microcontroller se pot folosi 2 circuite buffer (figura 6.11) iar dacă transferul este de intrare se pot folosi multiplexoare (figura 6.12.

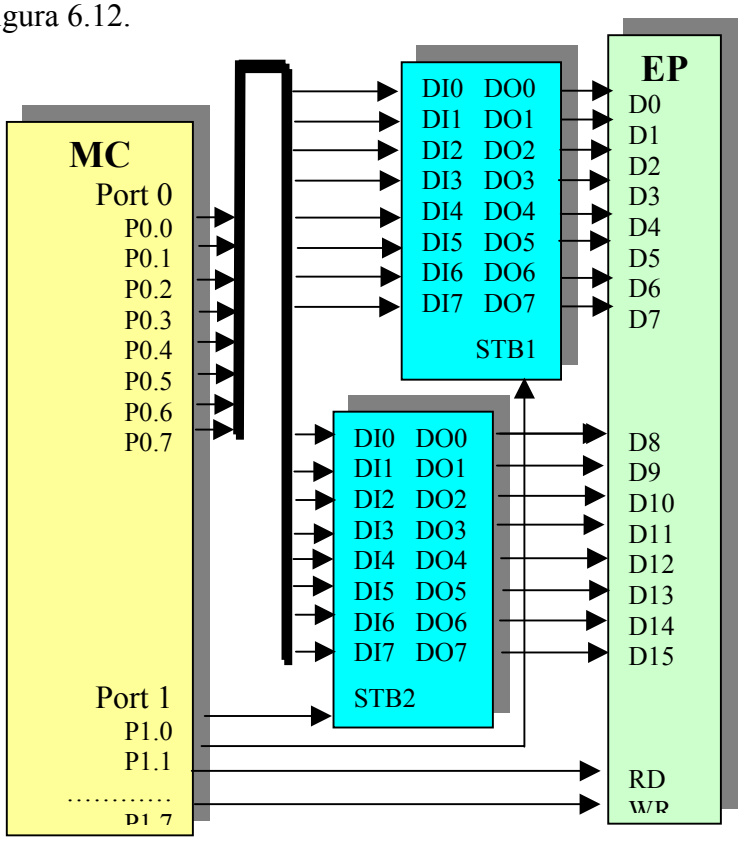


Figura 6.11. Conectarea unui EP de ieşire pe 16 biți la un port de date

În figura 6.11 se poate vedea conectarea a două circuite circuite buffer de 8 biți care au la intrare aceleaşi linii de date de la portul P0 al microcontrollerului. În primul pas se încarcă primul buffer comandat cu un strob (STB1), la liniile de date fiind puşi 8 biți din cei 16 ai cuvântului. În al doilea pas se încarcă al doilea buffer comandat cu STB2 şi având la liniile de date ceilalți 8 biți. Cu un semnal de WR cei 16 biți sunt aplicați la EP.

În figura 6.12. câte 2 linii de date de la EP sunt aplicate unor multiplexoare 2 la 1. Cele 2 căi ale multiplexoarele sunt comandate de unul sau două linii auxiliare din portul P1. În primul pas se citesc 8 biți selectând una dintre căi şi activând semnalul RD, apoi se citesc următorii 8 biți activând cealaltă cale şi semnalul RD.

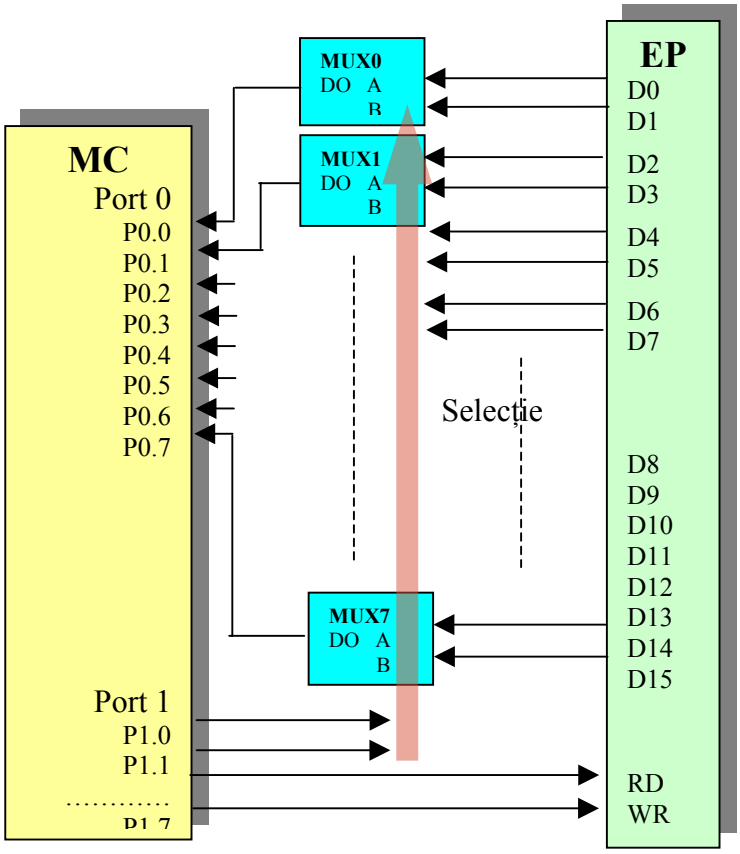


Figura 6.12. Conectarea unui EP de intrare pe 16 biți la un port de date

A treia extindere analizată este în cazul în care se doreşte conectarea mai multor EP la acelaşi port de date. Această situație este analizată într-un exemplu de conectare a unui circuit de interfață paralelă la porturi paralele, schema bloc din figura 6.13:

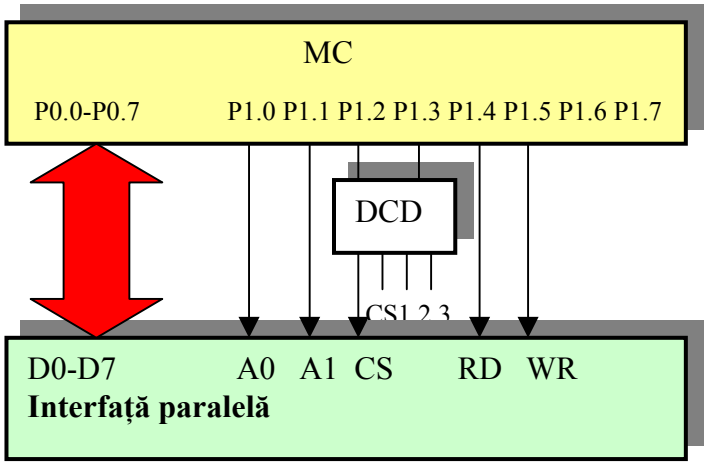


Figura 6.13. Conectarea unui circuit interfață paralelă la porturi paralele

Portul 0 al MC este folosit pentru transferul de date pe 8 biți, bidirecțional. Liniile 0 şi 1 din portul P1 sunt folosite pentru selectarea registrelor interne ale interfeței paralele. Liniile 2 şi 3 prin decodificare pot selecta unul din patru circuite conectate cu liniile de date la portul P0 al MC. Liniile 3 şi 4 ale portului P1 stabilesc sensul transferului. Se poate observa din schema bloc faptul că o conectare la porturi paralele este mai simplă decât una pe magistrală, mai ales dac numărul de circuite conectate este mic.

Pentru un transfer de un octet se pune octetul pe magistrala de date apoi se selectează portul intern al interfeței paralele prin punerea unui cuvânt pe portul 1, apoi se trimite un RD sau WR, ceea ce înseamnă un cuvânt pe portul P1 care face RD sau WR 1, urmat de un cuvânt care face RD sau WR 0. Din această succesiune de cuvinte de comandă se poate deduce că transferul este de cel puțin 4 ori mai lent decât la conectarea pe magistrală.

6.4. Concluzii

La alegerea modului de conectare pe magistrală sau la un port paralel există cazuri simple, în care decizia este uşor de luat. De exemplu dacă EP este pe 8 biți, unidirecțional şi transferul este fără protocol se va opta rapid pentru cuplarea la un port. Dacă aplicația solicită o viteză mare de transfer, cum este de exemplu un transfer de date video se va alege cuplarea pe magistrală. Sunt cazuri însă la care alegerea este mai dificilă, în cazul în care de exemplu EP este de 16 biți, bidirecțional şi transferul cu protocol. În acest caz este nevoie de circuite suplimentare atât la cuplarea la magistrală cât şi la un port paralel. Poate fi mai favorabilă găsirea unui microcontroller cu mai multe porturi paralele pentru a nu mai fi nevoie de multiplicarea liniilor unui port. Alegerea trebuie să aibă ca scop principal asigurarea funcționalității cu un număr de circuite suplimentare cât mai mic, ceea ce asigură simplitatea maximă cu consecințe favorabila la mărirea fiabilității şi scăderea costurilor.

7.Interfețe integrate în microcontrollere

7.1. Microcontrollere

Microcontrollerul (MC) are schema bloc generală simplificată din figura 7.1. În această schemă sunt puse în evidență interfețele, numite şi dispozitive de intrare ieşire I/O.

Unitatea centrală execută instrucțiunile din memoria program, pe care le primeşte prin magistrala de date. Structura Harvard este posibilă şi răspândită la MC pentru că de regulă instrucțiunile sunt stocate în memoria ROM iar datele în cea RAM. Fiecare MC are un controller de întreruperi şi unele au controller de DMA care admit atât intrări din exterior cât şi de la modulele interne. Modulele de I/O pot fi seriale sau paralele. Fiecare modul transferă date cu exteriorul prin intermediul registrului de date (RD). Modulul este comandat (configurat) de unitatea centrală prin intermediul unui registru de comenzi (RC) şi se poate citi starea modulului prin registrul de stare (RS), prin care se pot şi cere întreruperi. Registrele modulelor de I/O pot fi văzute de UC ca locații de memorie (la familia Motorola) sau ca dispozitive de I/O într-un spațiu de adresare separat (MCS 51). De regulă structura de bază a familiei conține anumite interfețe considerate foarte importante (timer, canal serial UART) şi linii de I/O grupate în porturi paralele de uz general. Pe structura de bază se adaugă diferite tipuri de interfețe care împart liniile de I/O cu porturile paralele de uz general.

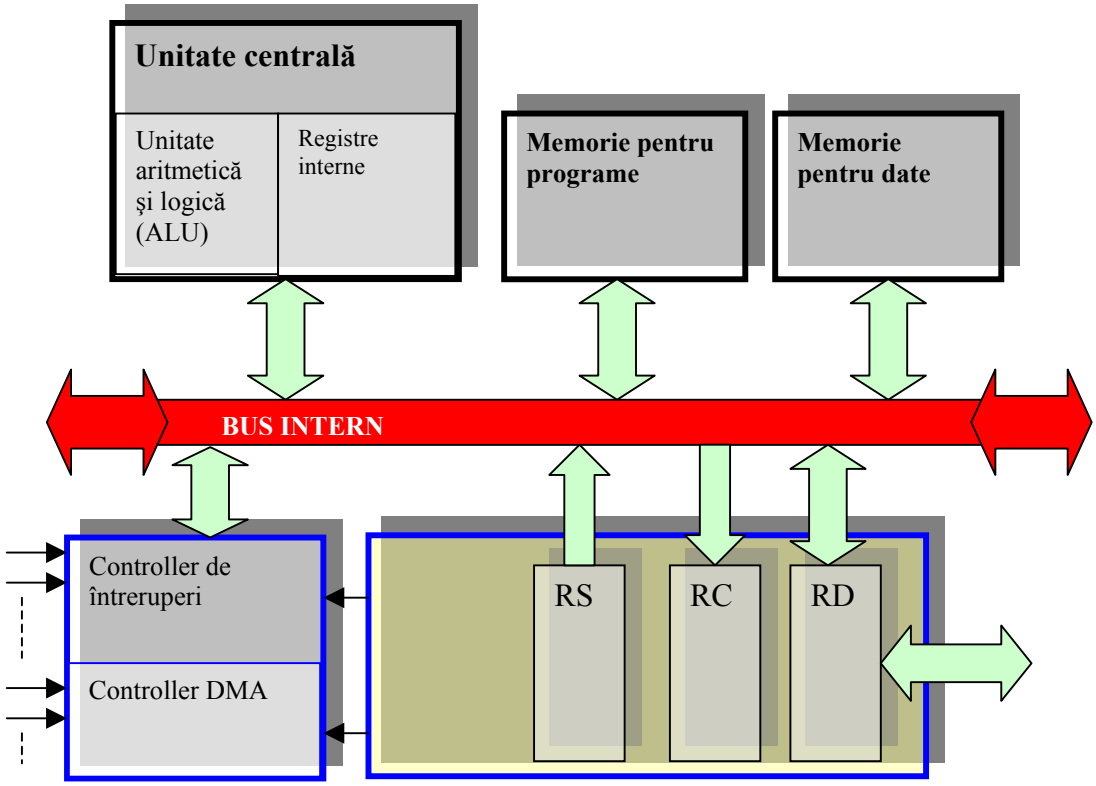


Figura 7.1. Schema bloc generală simplificată a unui microcontroller

Blocurile interne ale MC sunt legate între ele printr-o magistrală (bus) de date şi una de adrese. Mărimea acestor magistrale constituie una dintre caracteristicile cele mai importante ale unui MC.

Arhitectura Von Neuman prevede existenţa unui bus unic folosit pentru circulaţia datelor şi a instrucţiunilor. Când un controller cu o astfel de arhitectură adresează memoria pe linia de date se pune întâi codul instrucţiunii apoi conţinutul memoriei, accesul fiind realizat în 2 paşi, deci destul de lent. Arhitectura Harvard prevede un bus separat pentru date şi instrucţiuni. Când instrucţiunea este pusă pe busul de instrucţiuni, datele de la instrucţiunea anterioară sunt pe busul de date. Structura MC este mai complexă, dar performanţele de viteză sunt mai bune. Magistrala de date şi cea de adrese pot fi separate sau multiplexate. Magistralele pot să nu fie scoase în exterior (Motorola 6805) sau pot fi scoase în exterior direct (MCS 51) sau multiplexate (MC pe 16 sau 32 de biţi).

7.2. Unitatea centrală şi memoria

Unitatea centrală este formată din ALU şi un set de regiştri interni, figura 7.2., similari unor locaţii de memorie, folosiţi pentru memorarea unor date des folosite sau pentru programarea unor anumite funcţii. Diferitele familii de MC folosesc seturi diferite de regiştri. Există însă câţiva regiştri comuni:

1. **A** (Accumulator) registrul acumulator care este folosit deseori pentru a stoca un operand şi rezultatul unei operaţii aritmetice.
2. **I** registru de index, folosit la adresări indirecte.
3. **S** registru de stare, care conţine indicatorii de stare: Carry, Zero etc.
4. **PC** (Program Counter) este stocată adresa următoarei instrucţiuni de executat. După un RESET (iniţializarea MC), registrul PC se încarcă dintr-o locaţie de memorie numită vector de reset. Această locaţie conţine adresa primei instrucţiuni de executat. După execuţia acestei prime instrucţiuni, PC se incrementează.
5. **SP** (Stack Pointer) conţine indicatorul de stivă. Stiva este o memorie de tip LIFO, în care ultimul octet stocat este primul scos din memorie. Conţinutul acestui registru stabileşte adresa din memorie unde este definită stiva.

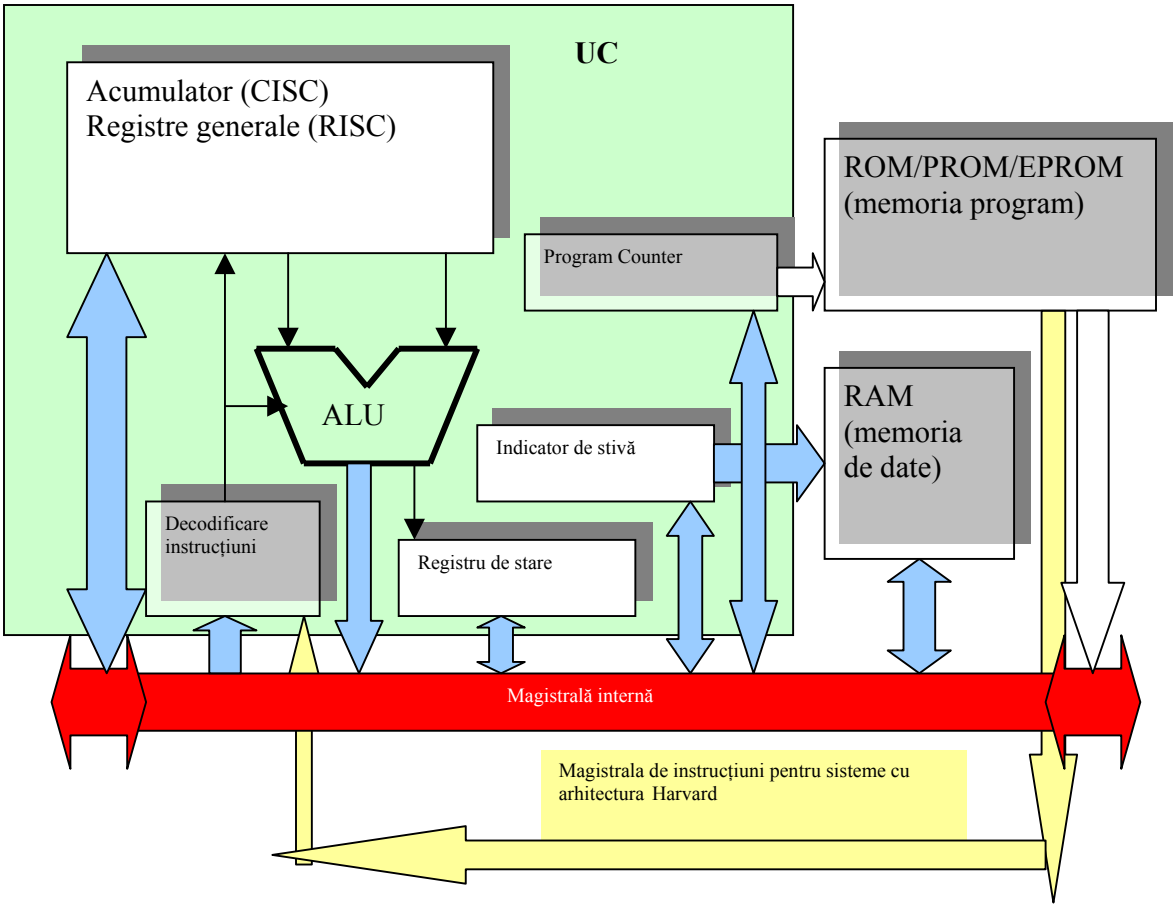


Figura 7.2. Schema bloc a unei unităţi centrale tipice

Tipurile de memorii utilizate în prezent la microcontrollere sunt:

1. Memoria ROM (Read Only Memory) este cea mai ieftină şi simplă memorie şi se foloseşte la stocarea programelor în faza de fabricaţie. Unitatea centrală poate citi informaţiile, dar nu le poate modifica.
2. Memoria PROM (Programmable Read Only Memory) este similară cu memoria ROM, dar ea se poate programa de către utilizator. După posibilităţile de ştergere, această memorie poate fi de mai multe feluri:

- OTP (One Time Programmable, PROM) este de fapt o memorie EPROM, dar *chipul* a fost capsulat într-o capsulă de material plastic fără fereastră, care este mult mai ieftină. Memoria nu se poate şterge sau reprograma. Preţul unui MC cu OTP este mic, viteza este bună, dar aplicaţiile sunt lipsite de flexibilitate.
- Memorii care pot fi şterse electric de către unitatea centrală, în timpul funcţionării. Ştergerea este selectivă, iar pentru reînscriere trebuie făcuţi mai mulţi paşi. Astfel de memorii sunt cele EEPROM (Electrically Erasable Programmable Read Only Memory) şi memoria FLASH EPROM.

3.Memoria RAM (Random Access Memory) este o memorie volatilă care poate fi citită sau scrisă de unitatea centrală. Locațiile din RAM pot fi accesibile în orice ordine. Pe chip, memoria RAM ocupă mult loc şi implicit costurile de implementare sunt mari. De aceea un MC include de obicei puțin RAM.

Observații:

1. Stocarea programelor în memorii nevolatile permite ca MC să fie programat fără a fi scos din circuitul în care funcționează (Field Programming/ Reprogramming).
2. Producătorii recomandă ca la producții de volum mare să se folosească memoria ROM, care se înscrie la fabricant, cu mască, la producții de volum mic să se folosească memoria OTP (PROM) iar pentru prototipuri să se folosească memoria EEPROM sau FLASH.
3. Unele familii de MC tratează spațiul de intrare ieşire ca şi memoria, iar altele au spații diferite de adresare pentru memorie şi spațiul I/O. Tratarea unitară a acestor spații are avantajul simplității dar limitează numărul de locații de memorie adresate. Tratarea unitară este justificată de asemănările existente între stocarea unui bit în memorie sau într-un latch de I/O.

7.3. Timerul

Timerul este modulul intern care realizează funcțiile de timp, fiind unul dintre modulele cele mai importante şi des folosite. Modulele timer sunt foarte diferite ca şi funcționalități şi complexitate, aspect de care trebuie ținut cont la alegerea microcontroller-ului pentru aplicațiile sensibile la timp. Schema bloc a modulului timer este dată în figura 7.3.

Tactul pentru timer poate fi ales tactul sistem sau un tact de la un pin extern. Tactul poate fi divizat de un număr programabil de ori cu un numărător de prescalare (timer de prescalare). Comanda funcționării timerului se face cu un registru de control şi stare.

Timerul este format din 3 module, a căror caracteristică este numărul de biți:

- numărător (timer)
- registru de încărcare
- registru de captură

Modurile de lucru sunt următoarele:

1. mod numărător, în care Timerul numără tactul de la intrare şi la depăşire înscrie un bit în registrul de stare sau se poate cere o întrerupere. Dacă tactul este extern înseamnă că se face o numărare a evenimentelor externe. De regulă frontul activ se poate programa, adică să se facă o incrementare a numărătorului la front pozitiv sau negativ.
2. mod captură, în care un pin exterior poate comanda oprirea timerului şi încărcarea valorii la care a ajuns în registrul de încărcare. Acest mod poate fi folosit pentru măsurarea unei perioade de timp.

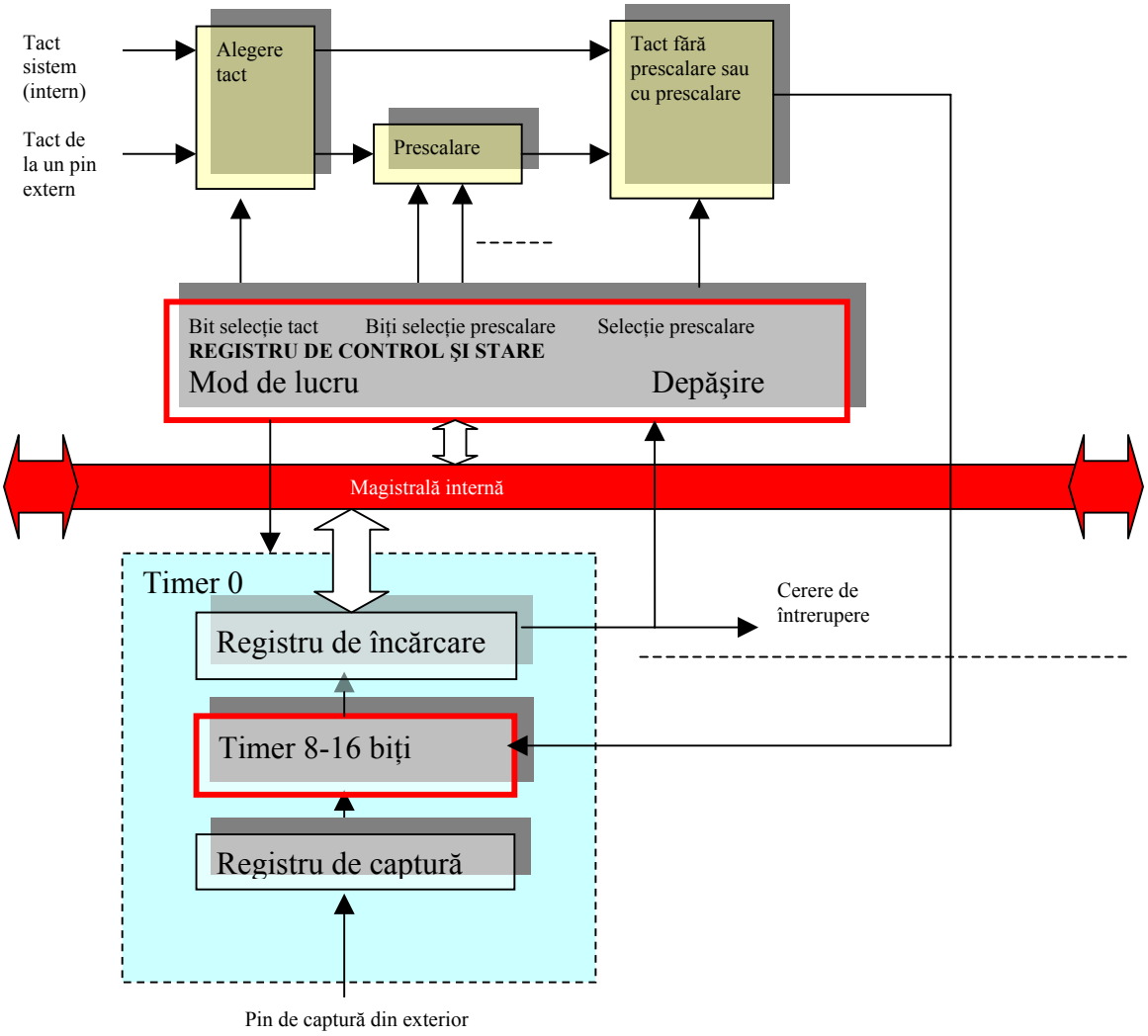


Figura 7.3. Schema boc a modulului timer

7.4. Interfață de comunicații seriale asincrone

Portul de comunicații seriale asincrone este numit UART (Universal Asyncronous Receiver Transmitter), iar Motorola îl numeşte pentru microcontrollerele proprii port SCI (Serial Communications Interface). Schema bloc a interfeței este dată în figura 7.4.

Caracterele seriale sunt transmise sau recepționate serial în registrele de transmisie sau recepție. La recepția unui caracter, acesta se încarcă în bufferul de recepție şi se cere o întrerupere. La emisie, un caracter se introduce în bufferul de transmisie de unde este trecut în registrul de deplasare şi se transmite serial, cerându-se şi o întrerupere. Ceasul poate fi selectat intern sau extern. Dacă este selectat intern, el se formează din tactul sistemului cu o divizare printr-un numărător de 16 biți (prescalare) şi apoi un numărător de 11 biți. Comanda USART se realizează cu un registru de stare şi control

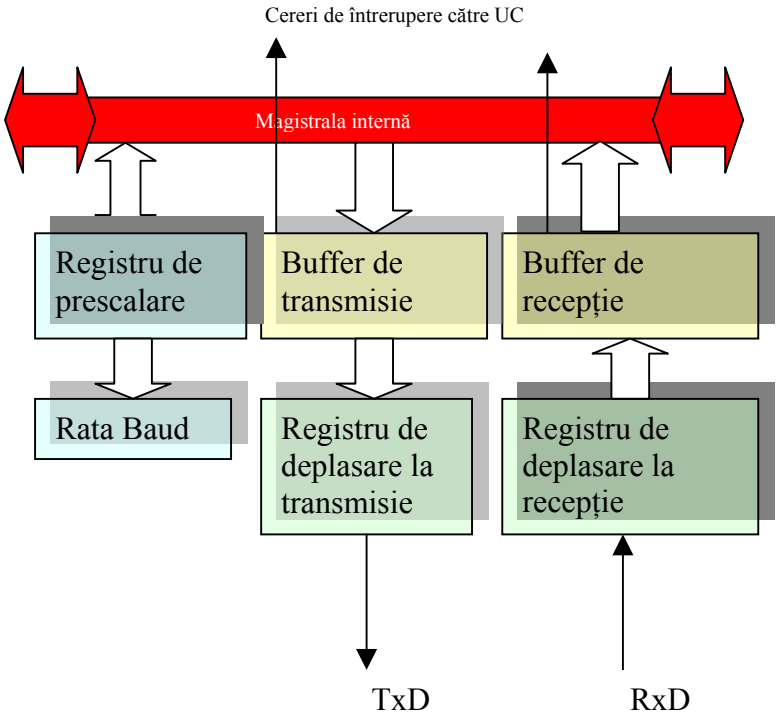


Figura 7.4. Schema bloc a interfeței de comunicații seriale asincrone

7.5. Interfață de comunicații seriale sincrone

Cu portul SPI (**Serial Peripheral Interface)** se poate realiza o comunicație sincronă simplă, folosită de regulă pentru a transfera date între circuite pe aceeaşi placă cu MC. Un transfer bidirecțional necesită 3 pini, unul dintre ei fiind alocat ceasului de transmisie generat de masterul SPI. Cu SPI se pot realiza transferuri şi între MC. Transferurile pot fi full duplex. Schema bloc a conectării unor dispozitive prin interfeța serială sincronă este dată în figura 7.5.

Numai un master SPI poate iniția un transfer. Masterul scrie un octet în registrul de transmisie SPI de undele datele merg într-un registru de deplasare care le serializează şi le transmite cu ceasul de transmisie. Transmisia se termină după 8 tacte. În slave datele intră în registrul de deplasare cu tactul de recepție, acelaşi cu cel de transmisie. Când au intrat 8 biți, caracterul intră în registrul de date. Pentru a se evita erorile de viteză (sau de suprascriere)(Overrun) trebuie ca octetul din registrul de date să fie citit înainte ca un alt octet să fie transmis din registrul de deplasare.

Pinii au următoarea semnificație:

- SCK (Serial Clock) este ieşire de tact pentru sincronizare;
- MOSI (Master Output Slave Input) este ieşirea serială pentru MASTER;

- MISO (Master Input Slave Output) este intrarea serială pentru MASTER;
- /SS (Slave Select) selectează circuitul SLAVE şi protejează MC dacă două circuite sunt master. Acest semnal activ dezactivează la celălalt port SPI modul master.

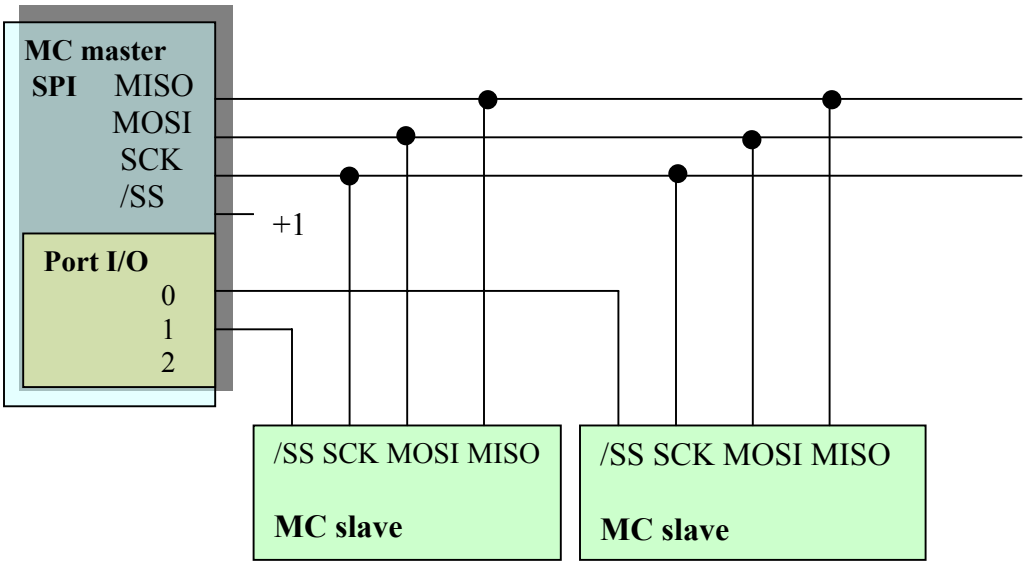


Figura 7.5. Schema bloc a conectării unor dispozitive prin interfeţa serială sincronă

Dacă dispozitivul master lucrează cu mai multe dispozitive slave atunci semnalul /SS nu este suficient şi selecţia dispozitivului slave se face cu linii de ieşire dintr-un port auxiliar, ca în figură.

7.6. Interfaţa CAN

Protocolul CAN (**Controller Area Network)** a fost definit de BOSCH în 1991 pentru utilizarea pe o magistrală la autoturisme, unde să îndeplinească condiţii specifice: procesare în timp real, fiabilitate într-un mediu perturbat şi preţ mic.

La transmisia CAN datele sunt codificate pentru a fi trimise pe linie în cod NRZ, iar la fiecare grup consecutiv de 5 biţi cu aceeaşi valoare logică se introduce un bit (deci o tranziţie) care se extrage la decodificare. Nivelele pe linie sunt numite dominant (0) şi recesiv (1).

Transferul de date prin CAN se face cu cadre (blocuri de date) care sunt citite de toate dispozitivele cuplate la CAN dar sunt reţinute de acestea doar acelea care conţin adresa dispozitivului.

Fiecare dispozitiv CAN recepţionează toate cadrele şi dispune de un filtru de acceptanţă care selectează cadrul cu adresa proprie a dispozitivului. în configuraţie se pot adăuga noi dispozitive, cu adresă proprie, fără nici un efort. Dacă cadrul recepţionat este eronat, toate

dispozitivele CAN trimit un cadru de control care indică o eroare. Fiecare cadru eronat incrementează în dispozitiv un numărător de erori (care este decrementat de cadrele valide). Un număr de erori mai mare de o anumită limită produce decuplarea dispozitivului de la magistrala CAN. Schema bloc a interfeței CAN este dată în figura 7.6.

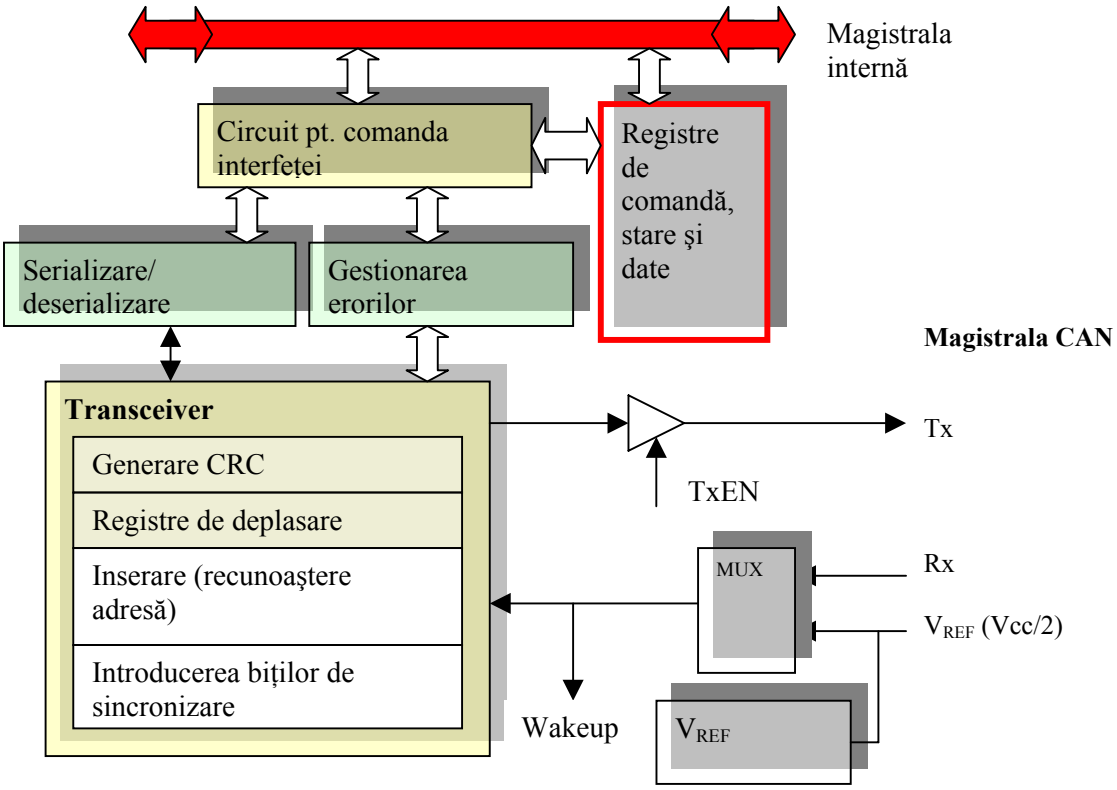


Figura 7.6. Schema bloc a interfeței CAN

7.7. Ceasul de gardă

Ceasul de gardă este un timer care poate fi programat să numere un tact care provine de la un registru de prescalare, figura 7.7. Dacă numărătorul ajunge la capăt, semnalul de depăşire declanşează un RESET al circuitului. Este sarcina programatorului să scrie în registrul de control un cuvânt care va reiniţializa numărătorul. În cazul în care MC nu mai este sub controlul programului, el va fi resetat de către ceasul de gardă

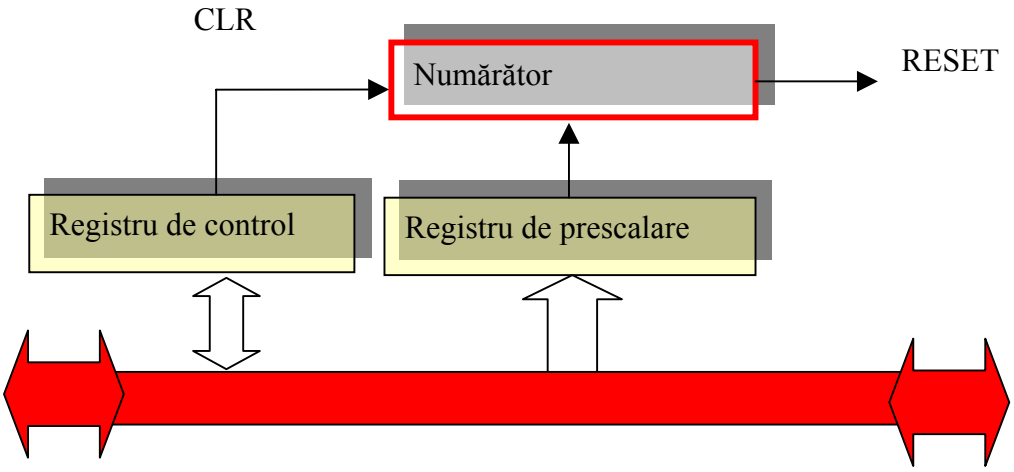


Figura 7.7. Ceasul de gardă

7.8. Generator PWM

Modulația impulsurilor în lățime (Pulse Width Modulation) are multe aplicații, mai ales în comanda motoarelor de curent continuu sau a surselor de alimentare. Din acest motiv, unele MC includ în structura lor un modulator PWM ca interfață distinctă. Un semnal PWM arată ca în figura 7.8.

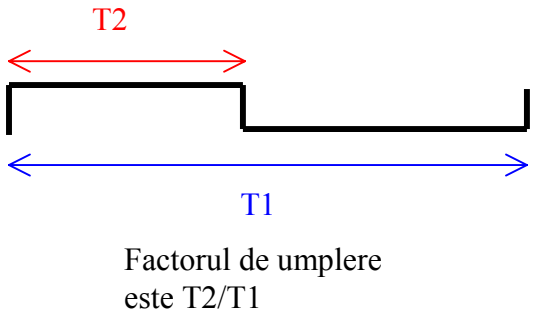


Figura 7.8. Semnal cu punerea în evidență a factorului de umplere

Schema bloc a unui generator PWM este reprezentat în figura 7.9. Frecvența de repetiție este programată cu un registru de prescalare care generează un ceas pentru un numărător de 8 biți. Conținutul numărătorului este comparat cu cel al registrului PWM, dacă este mai mare ieşirea PWM este LOW, dacă este mai mic sau egal PWM este HIGH. Factorul de umplere poate fi astfel modificat între 0 şi 254/255.

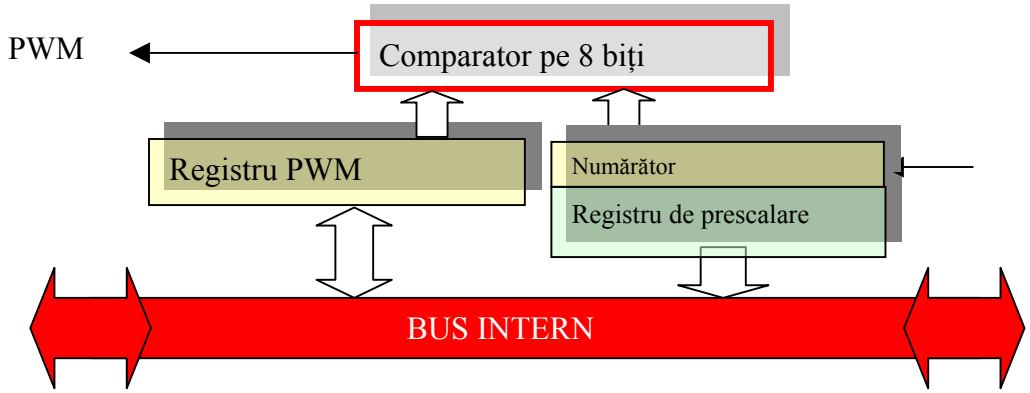


Figura 7.9. Schema bloc a unui generator PWM

7.9. Convertor analog digital

Schema bloc a unei interfețe tipice de conversie a semnalelor analogice în semnale digitale este dată în figura 7.10. Circuitul analogic de intrare constă într-un multiplexor analogic şi un convertor A/D de 8-10-12 biți cu aproximații succesive (sunt câteva MC cu convertor cu integrare, COP8). Tensiunea de referință pentru convertor şi masa analogică sunt conectate prin pini speciali. Convertorul este controlat de registrul de control, care selectează şi canalul de conversie. Terminarea conversiei este semnalizată cu un bit ACK tot în registrul de control, iar rezultatul conversiei este stocat în registrul de date.

O conversie poate fi declanşată în 3 feluri:
-start în operare normală şi reintrare în operare normală;
-start în operare normală apoi intrare în mod inactiv (Idle);
-intrare în mod inactiv şi declanşarea unei conversii din exterior printr-un pin exterior.

De regulă modul de conversie poate fi:
-conversie singulară
-conversie continuă

Cu registrul de control se poate programa:
-selecția canalului analogic dorit de la intrare la convertorul AD;
-se poate programa ca o conversie să fie declanşată de pinul extern ;
-se poate declanşa o conversie;
-conține un bit care semnalează că s-a terminat conversia, care poate solicita o cerere de întrerupere;

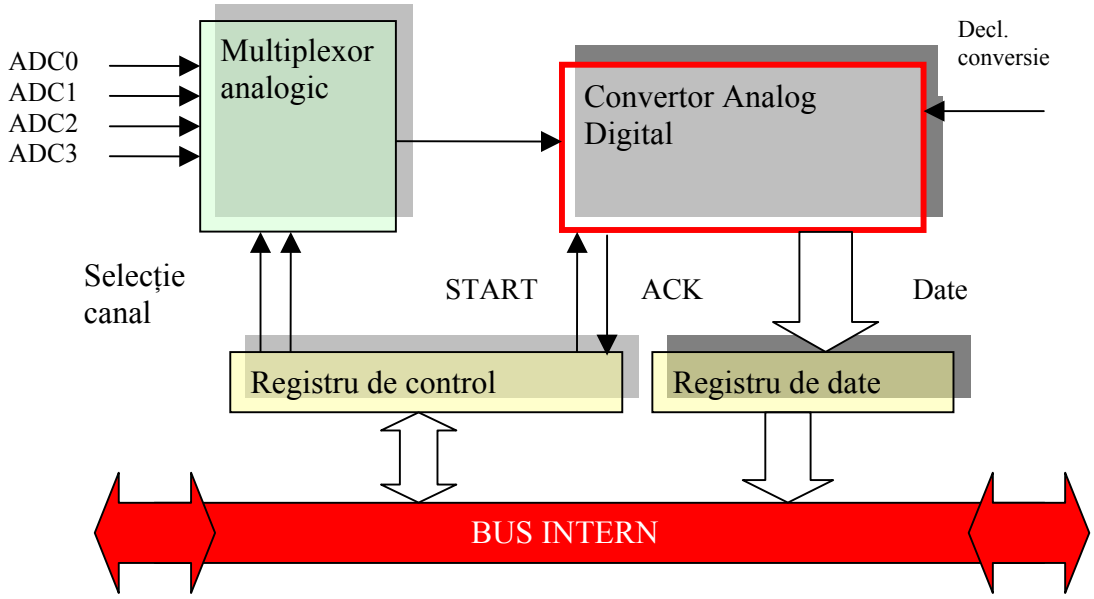


Figura 7.10. Schema bloc a unui convertor analog digital

7.10. Proiectarea sistemelor cu MC în vederea siguranței în exploatare

Pot apare 2 categorii de probleme: aplicația poate genera perturbații (conduse sau radiate) sau poate fi susceptibilă la perturbații (conduse sau radiate). Descoperirea unor **probleme de EMI** în timpul producției aplicației poate fi costisitor deoarece s-ar putea să fie necesară reproiectarea aplicației, de aceea este necesar să se **proiecteze în vederea EMC**.

Perturbațiile sunt generate de armonicile semnalelor digitale din circuit. Ele pot fi radiate de buclele de cablaj care se comportă ca şi antene sau sunt conduse spre sursa de alimentare. Orice cale inductivă sau capacitivă pe traseul acestor armonici poate provoca vârfuri de tensiune sau căderi de tensiune. Pentru un sistem cu un MC, perturbațiile sunt generate de regulă de cablaj, deoarece circuitele integrate au dimensiuni prea mici pentru a putea emite. Semnalul cu frecvența cea mai mare este tactul sistemului, generat cu un circuit oscilant cu cuarț. Datorită faptului că forma semnalului este apropiată de forma sinusoidală, conținutul de armonici este mic. Dacă tactul este adus din exterior, este nevoie de mare grijă pentru a reduce buclele de circuit emisive.

Pentru un sistem care are memorii externe cuplate la MC, liniile de transfer pot fi emisive, deoarece frecvențele de tranziție sunt mari.

Prin metodele de **programare defensivă** se poate îmbunătăţi mult siguranţa în funcţionare, fără nici un hardware suplimentar. Câteva din cele mai eficiente metode sunt:

-reîncărcarea periodică a registrelor care comandă pinii de I/O şi a celor mai importante registre. Pinii de I/O sunt legătura MC cu exteriorul, de aceea ei sunt supuşi perturbaţiilor. Readucerea lor la nivele corecte micşorează probabilitatea ca o perturbaţie să se propage în circuit.

-citirea repetată a semnalelor de intrare micşorează riscul unei citiri greşite. De exemplu citirea unui pin care este legat la o tastă de 3 ori la rând la intervalul de timp normal pentru 3 citiri succesive, dacă s-a citit aceeaşi valoare, elimină posibilitatea unei perturbaţii.

-dacă există locaţii în RAM nefolosite, după fiecare etapă de rulare a programului se scrie un bit în RAM. Înainte de rularea unei rutine critice se verifică valoarea stocată în RAM şi rutina se execută doar în cazul în care valoarea din RAM este corectă.

-dacă într-o aplicaţie există memorie nefolosită, aceasta se umple cu instrucţiuni de salt într-un loc cunoscut pentru ca un salt neprevăzut în memorie datorat unei perturbaţii să fie anulat de saltul în locul cunoscut, cu o anumită probabilitate.

7.11. Medii de programare şi exemple

Pentru a realiza programe, pentru fiecare familie de microcontrollere există medii software care uşurează munca de programare, testare şi înscriere a codului în MC. Astfel câteva exemple de capturi de ecrane ale unor medii de dezvoltare sunt date în figura 7.11.

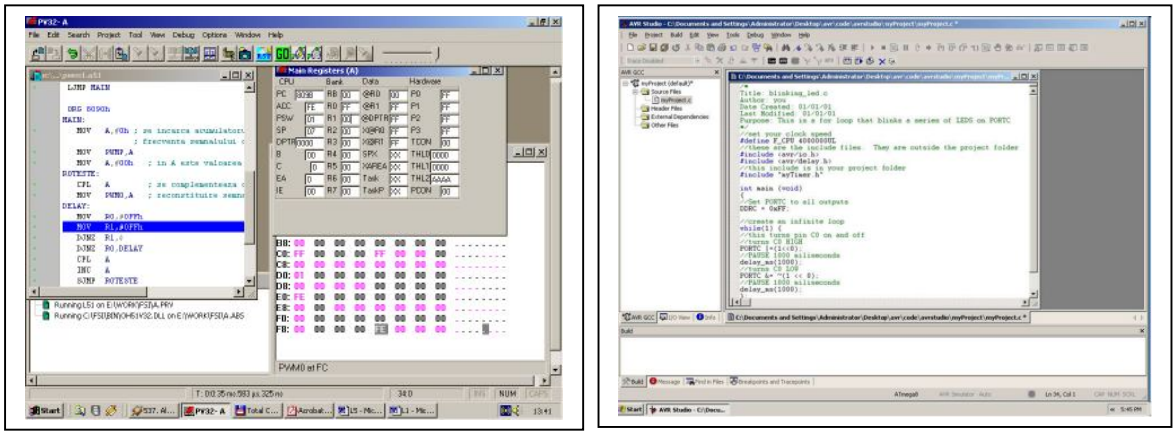


Figura 7.11. Mediu de dezvoltare pentru MCS 51 Franklin software (stânga) şi mediu de dezvoltare pentru ATMEL RISC, AVR Studio, (dreapta)

Mediul de dezvoltare MPLAB pentru PIC este arătat în figura 7.12. stânga şi o aplicație cu microcontroller realizată de un student la proiect este dată în dreapta.

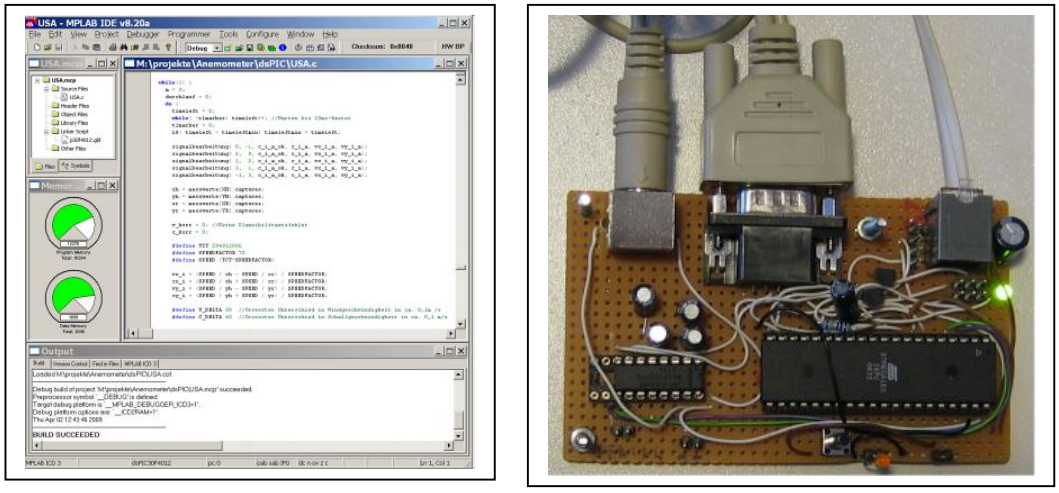


Figura 7.12. Mediul de dezvoltare MPLAB pentru PIC (stânga) şi o aplicație cu microcontroller (dreapta)

Se poate observa în aplicația cu microcontroller faptul că realizarea a fost la nivel de amator, fără construirea unui cablaj, totuşi aplicația a funcționat, a fost simplu de realizat şi studentul a avut ocazia să învețe multe lucruri. Pe placă, al doilea circuit este un MAX232 pentru adaptarea de nivel de tensiune de la interfața serială asincronă UART la RS232.

O aplicație realizată pentru proiectul de diplomă este un sistem mobil echipat cu un senzor de gaz, un modul Arduino cu microcontroller, drivere pentru motoare de curent continuu şi modul Bluetooth, figura 7.13 stânga. Comanda de la distanță se face cu un telefon mobil, interfața grafică a programului fiind arătată în dreapta.

Figura 7.13. Aplicație cu microcontroller

8.Interfețe pentru rețeaua Ethernet

8.1. Introducere

Ethernet este denumirea unei familii de tehnologii de rețele de calculatoare bazată pe transmisia cadrelor şi utilizată la implementarea rețelelor locale LAN. Numele provine de la cuvântul englez *ether* (tradus: eter), despre care multă vreme s-a crezut că este mediul în care acționau şi comunicau zeitățile. Ethernet-ul se defineşte printr-un set de standarde pentru cablare şi semnalizare electrică aparținând primelor două niveluri din modelul OSI - nivelul fizic şi legătură de date.

Ethernet-ul este standardizat de IEEE în seria de standarde IEEE 802.3. Aceste standarde permit transmisia datelor prin mai multe medii fizice, cum ar fi:

- cabluri coaxiale, folosite în primele rețele Ethernet, în topologie bus;
- cabluri torsadate, pentru conectarea sistemelor individuale la rețea, în topologie stea;
- cabluri de fibră optică, pentru viteză şi debit mare de date.

Istoricul Ethernet începe în 1973 la Centrul de Cercetări de la Palo Alto al corporației Xerox PARC, când Robert Metcalfe a proiectat şi testat prima rețea. El a dezvoltat metode fizice de cablare ce conectau dispozitive pe Ethernet şi standardele de comunicație pe cablu. Inițial, comunicația se desfăşura la viteza de cca. 3 Mbps, pe un singur cablu, partajat de toate dispozitivele din rețea. Acest lucru a permis extinderea rețelei fără a necesita modificări asupra dispozitivelor existente în rețea.

În 1979 Digital Equipment Corporation (DEC) şi Intel s-au asociat cu Xerox pentru standardizarea sistemului. Prima specificație a celor trei companii, denumită *Ethernet Blue Book*, a fost lansată în 1980, cunoscută sub denumirea *DIX standard*. Era un sistem pe 10 Mbit/s ce utiliza cablu coaxial gros ca backbone în interiorul unei clădiri, cu cabluri coaxiale subțiri legate la intervale de 2.5 m pentru a conecta stațiile de lucru.

IEEE (*Institute of Electrical and Electronic Engineers*) a lansat în 1983 standardul oficial *Ethernet* denumit IEEE 802.3 după numele grupului de lucru care a răspuns de dezvoltarea sa. În 1985 a lansat versiunea 2 (IEEE 802.3a) cunoscută sub denumirea *Thin Ethernet* sau *10Base2*, în acest caz lungimea maximă a cablului este 185 m.

Structura unei rețele locale mici este dată în figura 8.1. Topologia (structura) este în stea, ceea ce prezintă avantajul că întreruperea unui fir nu întrerupe rețeaua, dar şi dezavantajul unui consum mai mare de cablu față de topologia bus. Tot în aceeaşi figură sunt prezentate în partea de jos un conector RJ45 (stânga), cablu UTP (Unshielded Twisted

Pair) (mijloc) şi STP (Shielded Twisted Pair) (dreapta). În figura 8.2. este reprezentat un cablu optic, conectori şi structura internă.

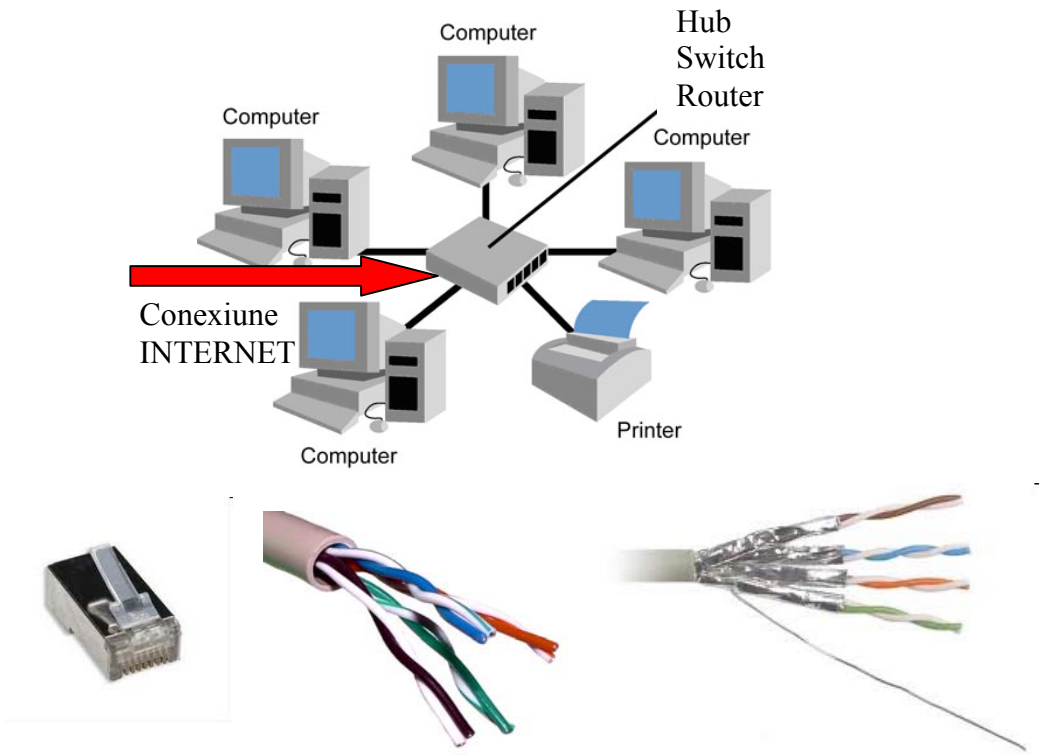


Figura 8.1. Structura unei rețele locale Ethernet, conector şi cabluri pentru transmisie

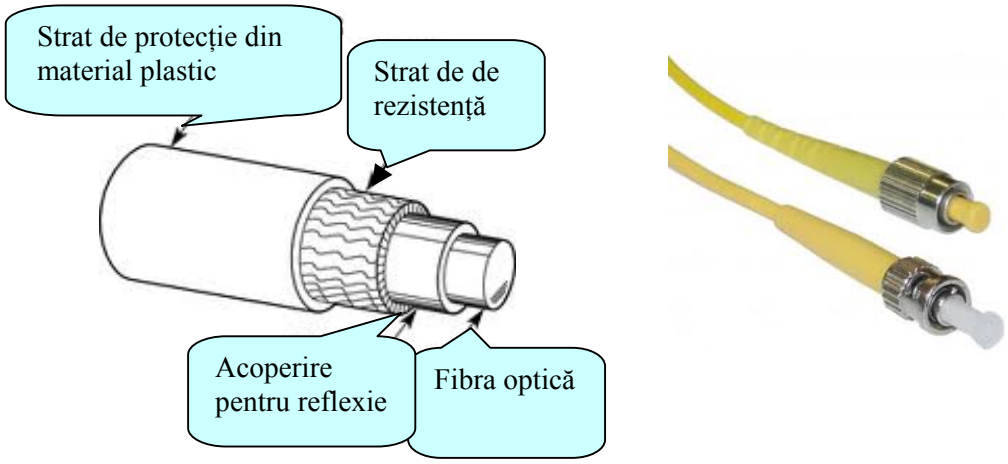


Figura 8.2. Cablu optic

Adresa IP (Internet Protocol) este o adresă numerică alocată fiecărui calculator conectat în Internet. Adresa IP permite identificarea expeditorului şi destinaţiei unui mesaj. Prima versiune apărută care este folosită şi astăzi este **IPv4** în care adresa este pe 32 de biţi. Reprezentarea canonică a IP-ului IPv4 este pe grupe de 8 biţi, în zecimal, separate de punct, de exemplu: 192.168.0.1. Creşterea numărului de calculatoare cuplate în Internet a făcut ca IP-urile în această versiune să fie insuficiente şi astfel a apărut **IPv6**, pe 128 biţi. Autoritatea internaţională **Internet Assigned Numbers Authority** (IANA) distribuie adresele IP la 5 autorităţi regionale care apoi le distribuie la ISP (Internet Service Provider)

O schemă bloc a conexiunii prin reţea Ethernet locală a calculatoarelor este dată în figura 8.3. Fiecare calculator trimite date serial pe Tx şi recepţionează date pe Rx. Două perechi de fire sunt libere pentru conectarea unei linii telefonice. Cadrul (şirul) de date seriale conţine adresa IP a sursei şi a destinaţiei. În centrul reţelei se poate afla un Switch care analizează adresa destinatarului şi trimite datele doar la destinatar sau un Router care, în plus faţă de switch, stabileşte automat adrese IP pentru fiecare calculator cuplat şi permite conexiunea la Internet.

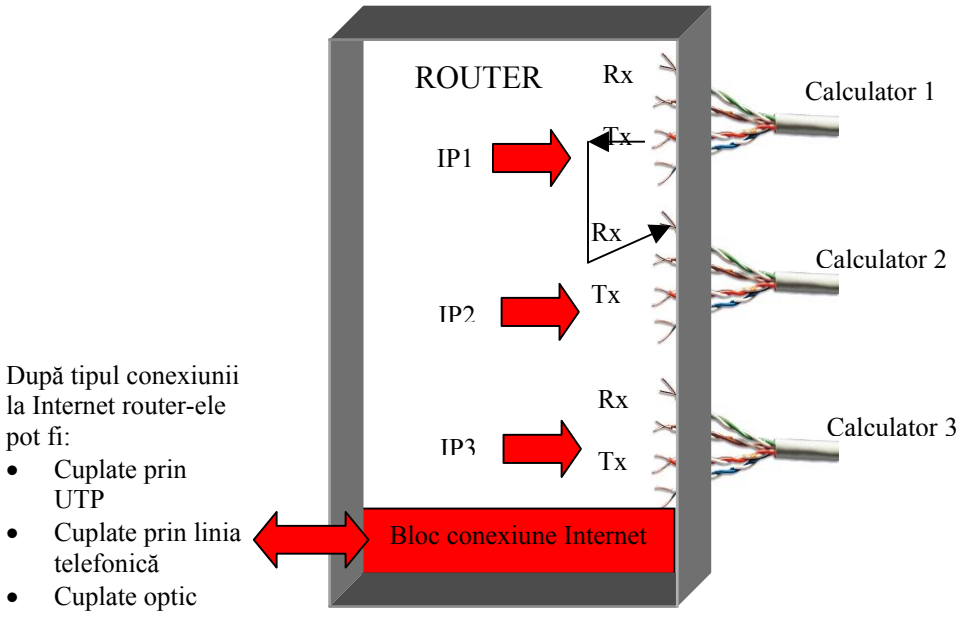


Figura 8.3. Schema bloc a conexiunii prin reţea Ethernet locală a calculatoarelor

Dacă 2 sau mai multe calculatoare transmit date în exact acelaşi moment se produce o coliziune, figura 8.4. Coliziunea este detectată de ambele calculatoare care au produs-o pentru că ambele urmăresc linia Tx. Linia Tx este comună, deci dacă un calculator pune

logic 1 şi unul logic 0 linia va fi în 0. Când s-a detectat o coliziune ambele calculatoare încetează transmisia şi o reîncep după trecerea unui interval de timp generat aleator (la întâmplare).

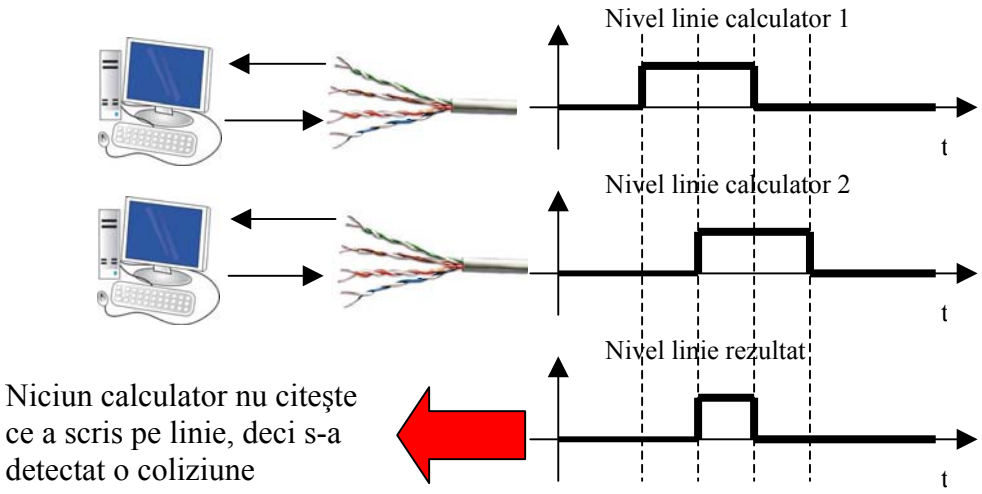


Figura 8.4. Coliziunea

8.2.Circuitul interfață de rețea RTL 8019

RTL 8019 este un controller de rețea Ethernet produs de Realtek care oferă o soluție simplă şi performantă aplicațiilor cu transfer de date prin rețea. Circuitul permite transfer full duplex pe UTP (şi dacă plăcile de rețea sunt cuplate între ele printr-un switch adecvat) mărind rata de transfer de la 10Mbps la 20Mbps. Circuitul suportă 3 nivele de economie de energie: mod adormit, mod oprit (Power Down) dar cu tactul în funcțiune şi mod oprit cu tactul oprit.

La RTL 8019 poate fi conectată o memorie ROM numită BROM (Boot ROM) din care se poate încărca un set de date care vor aduce sistemul de operare de pe server, caz în care stația conectată nu are nevoie de hard disc (aplicație importantă la sistemele care comandă automatizări). Memoria ROM poate fi de 16K, 32K sau 64K şi poate fi citită de sistem prin interfața cu magistrala, începând de la o adresă configurabilă. După incărcarea sistemului de operare memoria BROM poate fi invalidată pentru a elibera zona de adrese ocupată.

Pentru a mări viteza de transfer, cadrele recepționate sau cele de emis pot fi stocate într-o memorie locală SRAM cuplată la RTL 8019. Se pot conecta până la 32Kocteți SRAM. Există variante de RTL 8019 cu memorie SRAM integrată.

Configurația şi parametrii de inițializare pot fi stocați într-o memorie serială EEPROM cuplată la circuit. Această memorie (de tip 9346) poate fi programată cu circuitul cu RTL 8019. Schema bloc simplificată a circuitului RTL 8019 este dată în figura 8.5:

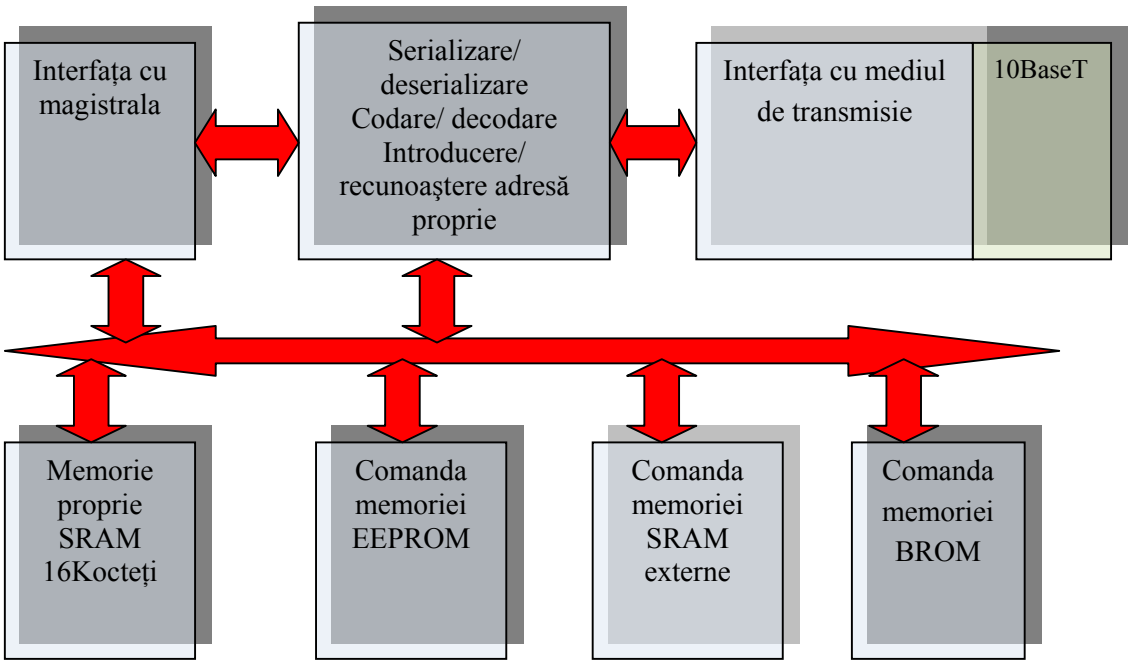


Figura 8.5. Schema bloc a circuitului RTL 8019

Programarea circuitului se realizează cu un set de registre citite /scrise de procesor la adrese de I/O. Aceste adrese sunt relative la o adresă de bază I/O care poate fi selectată la inițializare (una din 16 adrese posibile). Linia de întrerupere cu care lucrează circuitul poate fi programată ca una din 8 linii posibile.

Programarea adresei de bază de I/O, a dimensiunii memoriei BROM, linia de întrerupere se fac la inițializare prin jumperi. Desigur că unii parametrii pot fi modificați prin scrierea registrelor de configurare.

Cele mai importante semnalele la pini (din punct de vedere al aplicațiilor vizate) sunt semnalele de interfață cu magistrala:

- INT7-0 cereri de întrerupere, dintre care numai una este selectată la un moment dat;

- IORB indică un ciclu de citire iar IOWB indică un ciclu de scriere la un dispozitiv de I/O (din perspectiva procesorului);
- SA19-SA0 magistrala de adrese;
- SD15-SD0 magistrala de date;
- SMEMRB indică un ciclu de citire din memorie (din perspectiva procesorului).

Pentru transmisia unui cadru, informația de transmis se înscrie în RTL 8019 în memoria proprie prin transfer de memorie la adresa la care RTL 8019 a fost configurat. Transmisia se face prin DMA sau prin adresare şi înscrierea locației pe magistrală sau prin înscrierea datelor în zona de I/O a RTL 8019 la un port I/O. Recepția datelor se face prin transfer programat de memorie pe magistrală, prin DMA sau la un port I/O prin citirea registrului FIFO care este vârful unei stive în care se află cadrul recepționat.

Se poate observa că aceste semnale de interfață cu magistrala sunt cele studiate în capitolele anterioare *Magistrale* şi *Conectarea pe magistrală*, ceea ce subliniază importanța aspectelor fundamentale.

8.3. Cadru de date la transmisia Ethernet

Protocolul MAC (Media Access Control) este folosit pentru implementarea nivelului Legătură de Date în tehnologia Ethernet. Protocolul MAC încapsulează datele adăugând un antet de 14 bytes în fața datelor şi 4 bytes de informație de control (CRC) la sfârşitul datelor. Întregul cadru de date este precedat de o perioadă de timp şi un preambul de 8 bytes.

Formatul unui cadru de date Ethernet este format din următoarele câmpuri:

- Preambulul- Orice transmisie de cadru Ethernet începe cu o secvență de 8 bytes. Această secvență constă din 62 de biți alternanți de 1 şi 0 care sunt urmați de doi biți de 1. Scopul acestui câmp este de a permite receptorului sa-şi sincronizeze ceasul de recepție a datelor cu ceasul de transmisie. Orice transmisie serială sincronă cu refacerea tactului din datele citite şi care este organizată în cadre de date conține un preambul de sincronizare.

- Delimitatorul de început de cadru- Ultimul byte din preambul care se termină cu doi biți de 1 anunță începutul unui cadru de date. La primirea codului special "11" interfața Ethernet de recepție tratează următorii biți ca date.

- Adresa sursă şi adresa destinație- Aceste adrese sunt de 6 bytes fiecare şi reprezintă adresele fizice (adrese MAC) ale transmițătorului şi receptorului.

- Tip- Prin acest câmp este specificat protocolul superior care va prelua datele după procesarea Ethernet (valoarea 0x0800 identifică protocolul IP , acesta fiind şi cel mai folosit).

- Data- După procesarea Ethernet aceste date sunt trimise protocolului superior specificat în câmpul Tip. Lungimea minimă a acestui câmp este de 46 bytes şi cea maximă este de 1500 bytes. Orice depăşire a acestor limite este considerată eroare.

- CRC- acest câmp poartă şi denumirea de FCS – Frame Check Sequence şi este calculat de către sursă şi recalculat la destinaţie pentru detectarea eventualelor erori survenite în timpul transmisiei. Dacă se detectează vreo eroare cadrul de date este abandonat. Protocolul MAC nu oferă nici un mecanism prin care să se indice sursei că un anumit cadru de date a fost abandonat din cauza apariţiei unei erori.

Trebuie menţionat faptul că, fiind vorba de o topologie de tip multipunct (broadcast), cadrele Ethernet din reţea sunt preluate de către toate calculatoarele conectate la reţea. După efectuarea verificărilor privind corectitudinea cadrului, se verifică dacă acel cadru este destinat calculatorului respectiv. Verificarea se face pe baza adresei MAC de destinaţie. Această adresă poate fi chiar adresa MAC a calculatorului respectiv, şi atunci cadrul este procesat, adresa MAC de broadcast, şi în acest caz cadrul fiind procesat, sau o altă adresă MAC, şi atunci cadrul este abandonat.

Deoarece tehnologia Ethernet implementează funcţiile celor două nivele OSI, Legătură de Date şi Fizic, tot aici trebuie implementată şi metoda de adresare fizică. Protocolul MAC conţine şi această schemă de adresare. Această adresă fizică, ca şi întreaga tehnologie Ethernet, este implementată în interiorul interfeţei de reţea.

Adresa fizică constă dintr-un număr de 48 de biţi scris în hexazecimal care identifică în mod unic în lume o interfaţă de reţea. Această unicitate este dată de primii 24 de biţi care reprezintă un număr ce identifică producătorul interfeţei. Acest număr este denumit OUI – Organizational Unique Identifier şi este administrat şi distribuit producătorilor de către IEEE. Ceilalţi 24 de biţi reprezintă numărul de serie al interfeţei şi este administrat de către producător. Aceste adrese MAC mai poartă şi denumirea de adrese BIAs – Burned-in Addresses deoarece ele sunt înscrise într-o memorie ROM de pe interfaţa de reţea.

8.4. Circuitul interfaţă de reţea CS8900A

Circuitul CS8900 (producător CIRRUS LOGIC) este o interfaţă pentru reţea Ethernet care încorporează toate circuitele digitale şi analogice necesare conectării la reţea pe de o parte şi la magistrală sau la un port paralel pe de altă parte.

Schema bloc a circuitului CS8900 este dată în figura 8.6.:

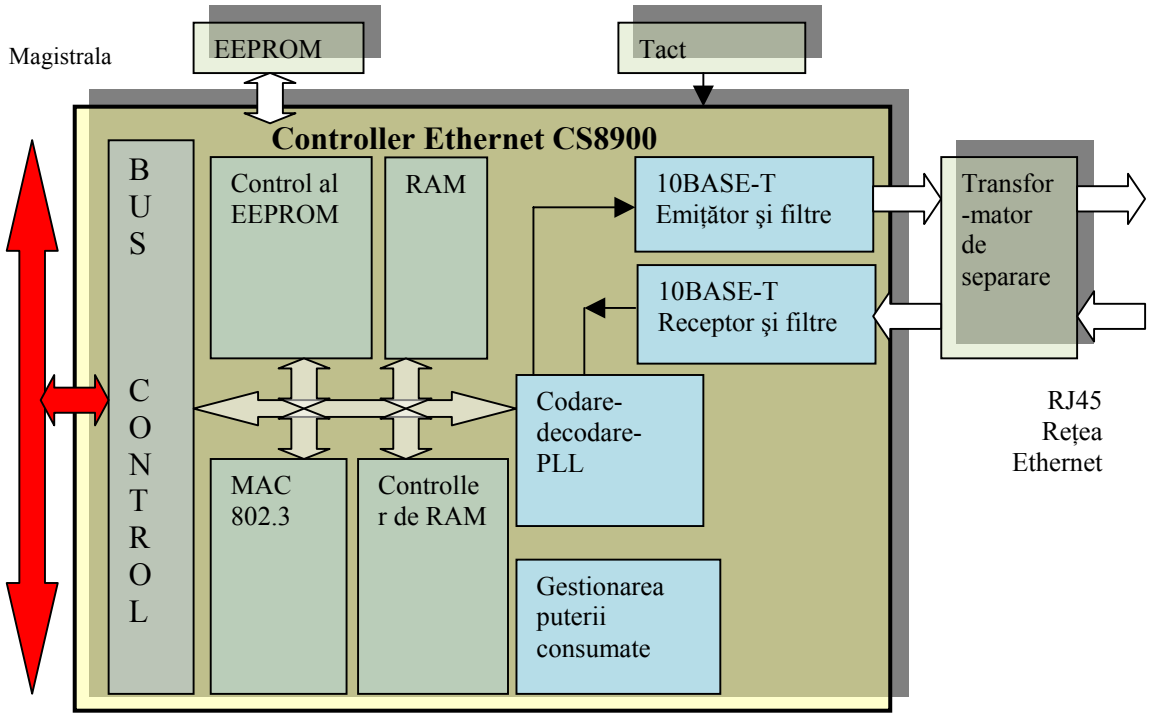


Figura 8.6. Schema bloc a circuitului CS8900 (sursa www.crystal.com)

Blocurile principale sunt:

1.Interfaţa cu magistrala, toate liniile de interconectare având capabilităţi de încărcare standard. Circuitul poate cere o întrerupere pe una din patru linii şi poate cere un transfer DMA pe una din trei linii. Liniile se selectează la iniţializarea circuitului.

2.Memoria internă (4K octeţi) face ca o memorie externă să nu mai fie necesară. Circuitul introduce în această memorie un întreg cadru de trimis sau recepţionat.

3.Modulul MAC (Media Access Control) de acces la reţea asigură accesul conform standardului IEEE 802.3 în mod full duplex. MAC se ocupă de toate aspectele legate de transmisia cadrului cum ar fi detecţia de coliziuni, generarea şi detectarea preambulului, generarea şi verificarea CRC. Modulul MAC retransmite automat cadrul după detectarea unei coliziuni.

4.Interfaţa cu EEPROM este necesară pentru a citi un EEPROM serial opţional care conţine datele de configurare ale circuitului.

5.Interfaţa analogică cu reţeaua conţine codorul şi decodorul Manchester, circuitul de refacere a tactului din semnalul recepţionat (cu PLL), transceiver pentru 10BASE-T. Trei

LED-uri arată starea circuitului: starea ON sau OFF a legăturii, activitatea Ethernet şi starea magistralei. Transceiverul 10BASE-T conţine emiţătoare şi receptoare de linie şi filtre analogice, în exterior fiind necesar doar un transformator de separare. Sunt suportate cabluri cu impedanţa caracteristică de 100, 120 şi 150Ω, ecranate sau neecranate.

Cele mai importante semnalele la pini sunt:

Semnale de interfaţă cu magistrala: SA0-19 (adrese), SD0-15 (date), RESET, /MEMR, /MEMW, /IOR, INTRQ0-3, DMARQ0-2, DMACK0-2, ALE. Semnale de interfaţă cu mediul de transmisie: TXD+, TXD-, RXD+, RXD- pentru 10BaseT.

Funcţionarea circuitului are la bază 2 funcţii: să trimită un cadru Ethernet şi să recepţioneze un cadru. Înainte de emisie sau recepţie, circuitul trebuie configurat. Spre deosebire de RTL 8019, CS 8900 nu gestionează liniile de date şi adrese ale memoriei externe, fiind de aceea un circuit cu mai puţine terminale şi fiind astfel mai uşor de folosit.

Transmisia unui cadru începe cu o comandă (Transmit Command) prin care se precizează când să înceapă transmisia (după ce toţi octeţii au fost transmişi în CS8900 de exemplu), existenţa CRC etc. Se trimite apoi lungimea cadrului, (Transmit Length), după care se trimit datele în CS prin transfer de memorie în memoria proprie sau prin transfer I/O. Circuitul transmite cadrul în reţea începând cu un preambul urmat de adresa destinaţiei, adresa sursei, date, octeţi de CRC (se pot citi amănunte în paginile anterioare, la structura cadrului Ethernet).

Recepţia unui cadru este realizată de CS şi cadrul este stocat în memorie. Recepţia se face prin decodare Manchester, apoi sunt eliminate preambulul, adresa destinatarului este verificată şi dacă corespunde cu adresa programată în CS cadrul este memorat şi se anunţă procesorul printr-o întrerupere. În a doua fază datele din memorie se transferă în memoria sistemului pe magistrală, prin transfer de memorie, transfer I/O sau prin DMA.

Transferul de memorie se face prin accesul direct al procesorului la memoria internă a CS, adresată pe liniile de adresă, sensul transferului fiind dat de MEMR sau MEMW. Registrele de configurare pot fi accesate şi ele în acest mod. La transferul I/O, CS este accesat prin 8 registre de 16 biţi, văzute de procesor în spaţiul de I/O. Sensul transferului este dat de IOR şi IOW. Acest al doilea mod (I/O) este ales implicit la pornirea circuitului (sau la RESET).

Programarea circuitului CS8900 se face printr-un concept original de programare, prin intermediul memoriei interne numită PacketPage. Accesul la această memorie poate fi realizat atât prin transfer cu memoria cât şi I/O. Transferul cu memoria este preferat deoarece ciclul de memorie este de regulă mai scurt decât cel de I/O. Conectarea lui

CS8900 la un microcontroller face imposibilă tratarea PacketPage prin transfer de memorie.

8.5. Interfațarea circuitelor CS8900 şi RTL 8019 cu microcontrollere

Circuitul CS8900A poate fi folosit cu multă uşurință prin cuplarea cu un microcontroller. În cazul folosirii unui microcontroller AT89S53 (pe 8 biți) transferul se face pe 8 biți. CS8900 poate lucra în acest mod cu unele restricții. Drivere pentru lucrul pe 16 şi pe 8 biți sunt disponibile pe pagina de la Cirrus Logic. Avantajele circuitului CS8900 sunt un număr mai mic de pini, documentație mai clară, modul de programare mai simplu datorită conceptului de PacketPage. Dezavantajul principal este că se procură greu.

Conectarea circuitului CS8900 la microcontrollerul AT89S53 este prezentată în figura 8.7. şi figura 8.8. Legătura de date între circuite este realizată printr-o magistrală de date de 8 biți formată de portul P0 al microcontrollerului. Magistrala de date permite cuplarea şi a altor circuite, care astfel pot comunica cu rețeaua, transferul fiind comandat de microcontroller.

Sensul transferului este comandat de semnalele /RD şi /WR (P3.6 şi P3.7). Cu un latch de adrese se memorează liniile de adresă, care sunt generate multiplexat cu cele de date la portul P0 al microcontrollerului. Latchul este comandat de semnalul ALE. Se obțin liniile de adrese A0, A1, A2, A3 necesare pentru selecția registrelor interne ale CS8900 şi rămân libere liniile de adrese A4-A7 cu care se pot selecta diferite circuite cuplate pe magistrala de date. Circuitul CS8900 este selectat cu linia A7. Un semnal de RESET format în Schema 2 inițializează atât microcontrollerul cât şi controllerul de rețea la punerea sub tensiune sau la apăsarea unui buton.

Microcontrollerul are prevăzute 2 interfețe pentru comunicația cu sistemul gazdă:

- Interfața serială RS232 pentru transferul de date. De la microcontroller spre RS232 este conectat un circuit pentru modificarea de nivel MAX232;
- Interfața serială sincronă SPI.

Două generatoare de tact furnizează tactul pentru microcontroller (10MHz) şi pentru controllerul de rețea (20MHz).

La controllerul de rețea se adaugă un minim de componente externe:

- Transformator de izolare pentru 10BASE-T (cablu UTP);
- 2 LED-uri de semnalizare.

O altă variantă de cuplare (arătată în figura 8.9.) este cu microcontroller RISC AT90S şi controller de rețea RTL8019. Diferența este că nu se creează o magistrală de date şi adrese, deci nu se pot cupla şi alte circuite. Altfel schema este asemănătoare. La microcontroller apar cele 2 interfețe de transfer de date RS232 cu adaptare de nivel (MAX232) şi cea serială sincronă SPI. La controllerul de rețea apare transformatorul de izolare şi LED-urilor de semnalizare. Transferul de date are loc prin portul PC al microcontrollerului iar selecția registrelor interne pentru RTL8019 precum şi precizarea sensului transferului se realizează cu semnale din portul PA.

Un circuit RTL8019 se poate conecta cu un microcontroller din familia 8051, soluția fiind menționată în cornelius@ethernet.isdn-development.de.

Câteva caracteristici principale ale folosirii circuitului RTL sunt:

- Memoria RAM internă de 16 Kocteți asigură un număr mic de componente externe circuitului (ca şi la CS8900);
- Pentru aplicații TCP-IP este necesară memoria EEPROM externă pentru a stoca codul IP;

Ca dezavantaj se menționează capsula circuitului (100 de pini) destul de dificil de lipit în condiții ne-industriale.

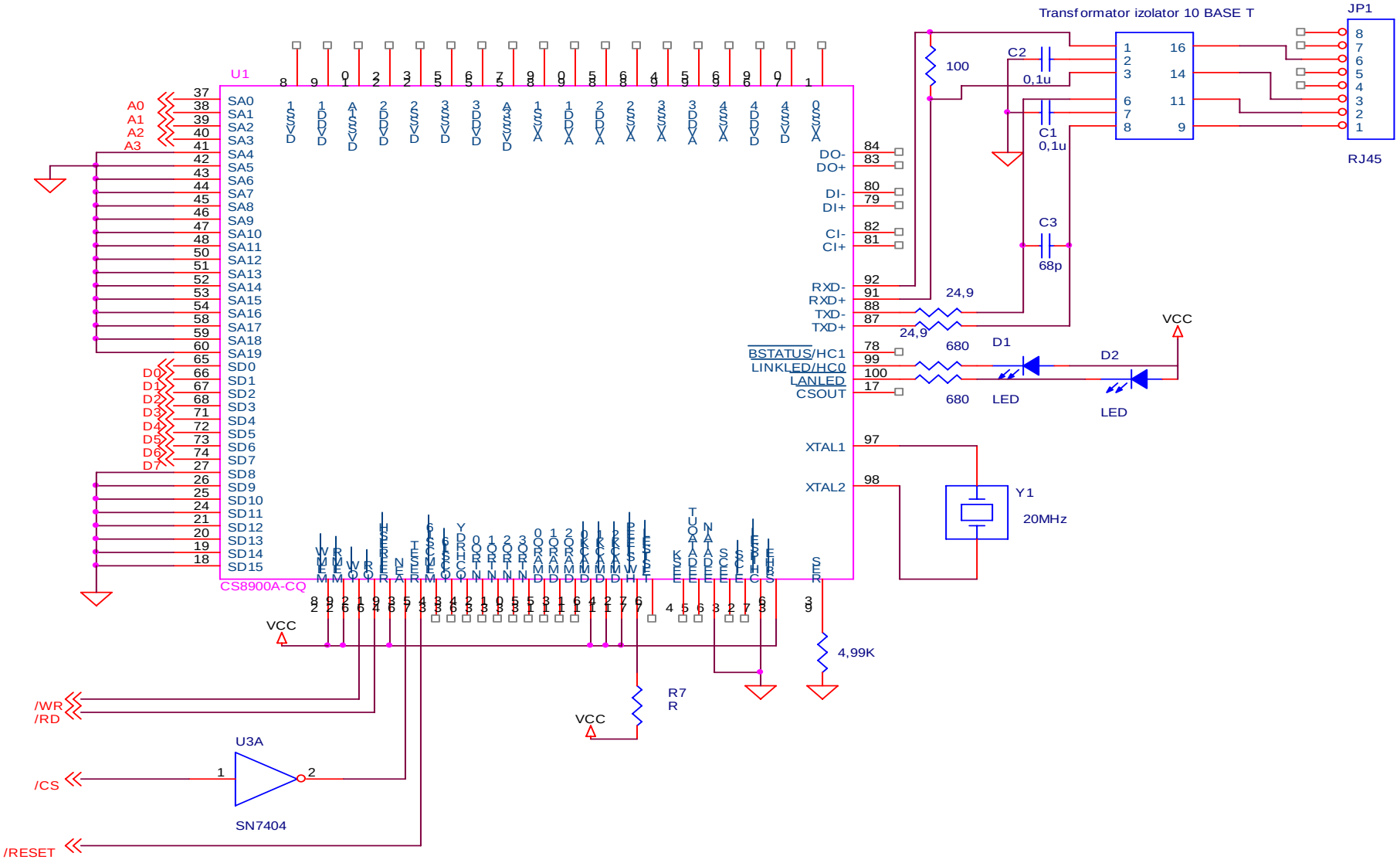


Figura 8.7. Circuitul CS8900 şi componentele externe pentru cuplarea la AT89S53

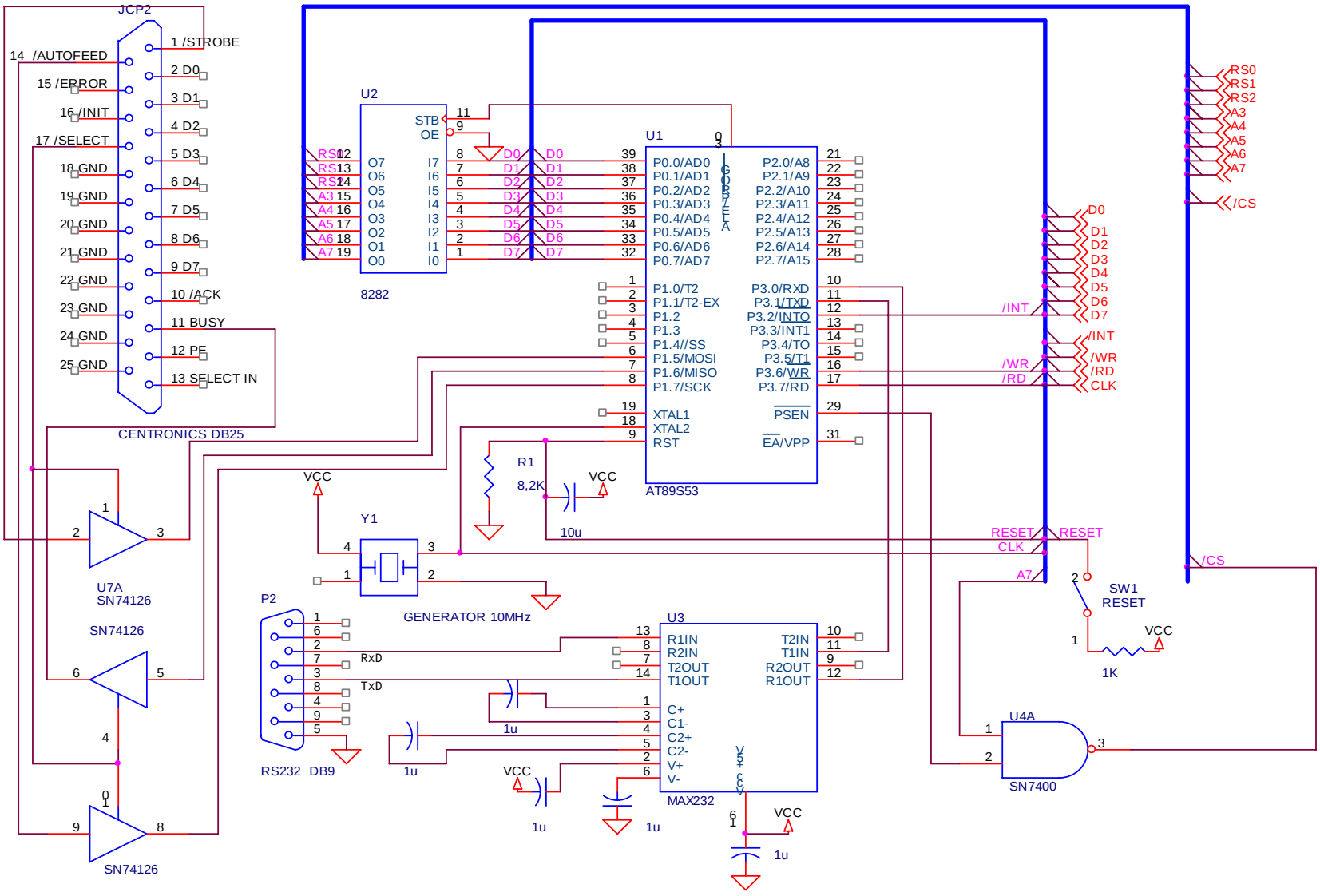


Figura 8.8. Schema de conectare a circuitului AT89S53 cu CS8900

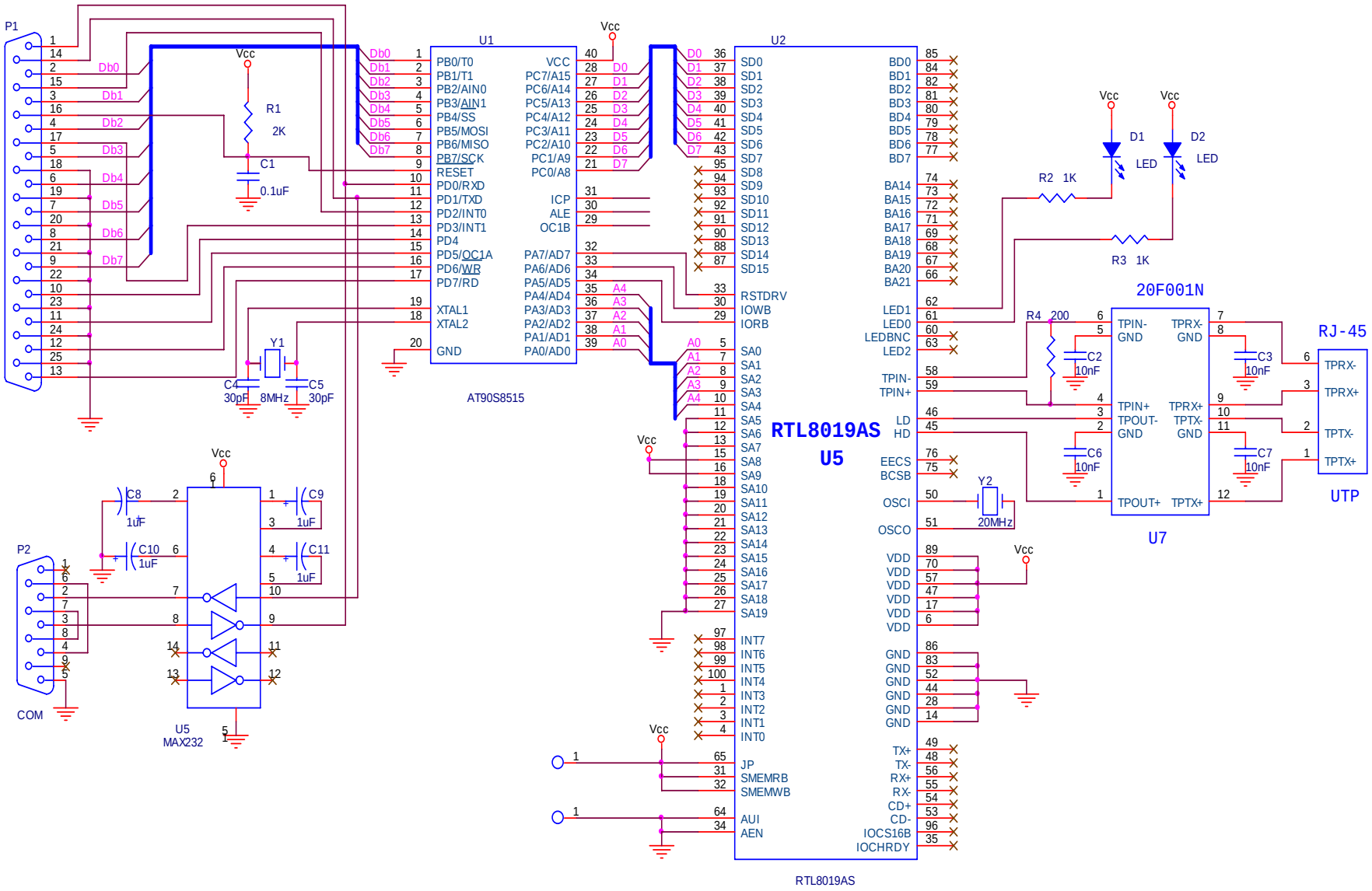


Figura 8.9. Conectarea unui circuit RTL8019 la un microcontroller RISC (AT90S8515)

8.6. Web server Site Player

În aproximativ 2 cm pătrați, SitePlayer (figura 8.10 stânga) include un server web, controller Ethernet 10baseT, memorie pentru stocarea paginilor web, coprocesor pentru obiectele grafice şi o interfață serială. Structura SitePayer conține:

- Controllerul de rețea RTL8019AS Realtek se ocupă de semnalele Ethernet şi transferă pachetele IP către microcontroller
- Microcontrollerul Philips 89C51 se ocupă de protocolul TCP/IP, suportă 8 porturi I/O şi un port serial.

SitePlayer se poate conecta prin interfața serială RS232 la un microcontroller şi prin conexiunea Ethernet se poate cupla la Internet. Pagina pusă de SitePlayer pe Internet poate conține date măsurate de microcontroller dintr-un proces şi, invers, pot fi generate comenzi pentru proces.

La pornire, SitePlayer afişează o pagină de test (figura 8.10 dreapta) prin intermediul căreia se poate aprinde ledul roşu şi ledul verde prezent pe placa de dezvoltare.

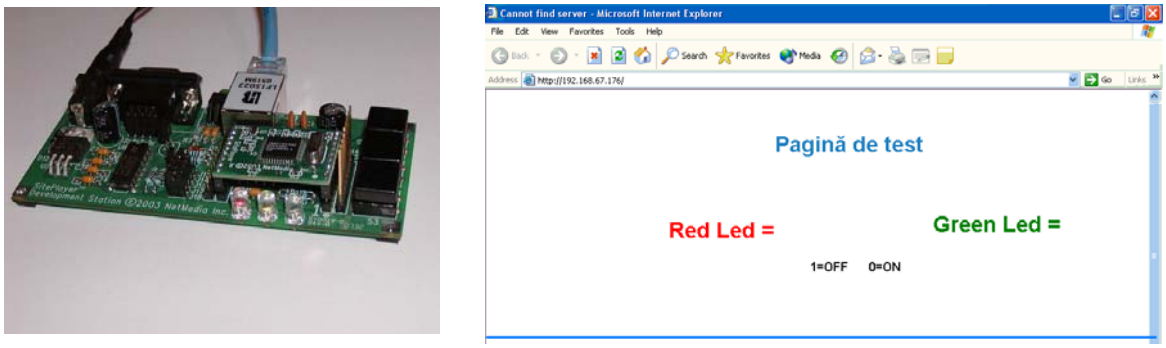


Figura 8.10. SitePlayer şi pagina inițială de test

Codul care indică SitePlayer-ului cum să funcționeze şi ce pagini web trebuie să servească trebuie definit de SitePlayer Definition file şi apoi asamblat într-o imagine binară (SitePlayer Binary image) folosind utilitarul SiteLinker. Imaginea binară este scrisă în memoria flash prin conexiunea Ethernet. Interacțiunea cu dispozitivul serial şi servirea paginilor web se poate face atât folosind placa de dezvoltare cât şi emulatorul SitePlayePC.

Paşii necesari creării unui proiect sunt:

1. Definirea şi crearea obiectelor (folosind un editor de texte) în fişierul SitePlayer Definition File(.SPD)
2. Creare paginilor web folosind un editor HTML.

3. Asamblarea şi download-area fişierului binar SitePlayer Binary file (.SPB) folosind programul SiteLinker.
4. Simularea folosind programul SitePlayerPC.

Două exemple de pagini create pe SitePlayer este cea din figura 8.11. Cu ajutorul unor imagini *.gif care reprezintă cifrele de la 0 la 9 ale unui display cu leduri se poate realiza un afişaj simplu şi elegant, imaginea fiind dată în stânga. O pagină realizată la un proiect de diplomă este cea din dreapta. Un microcontroller supraveghează rețeaua de alimentare de 230V A.C. şi are o interfață cu un SitePlayer. Fiecare eveniment din rețea (de exemplu o supratensiune) este detectat şi microcontrollerul trimite un mesaj care va avea ca efect completarea în tabel a descrierii evenimentului împreună cu ora la care a apărut. Lista de evenimente pot fi văzute pe Internet de la distanță de firma distribuitoare de energie sau de proprietarul locației respective.

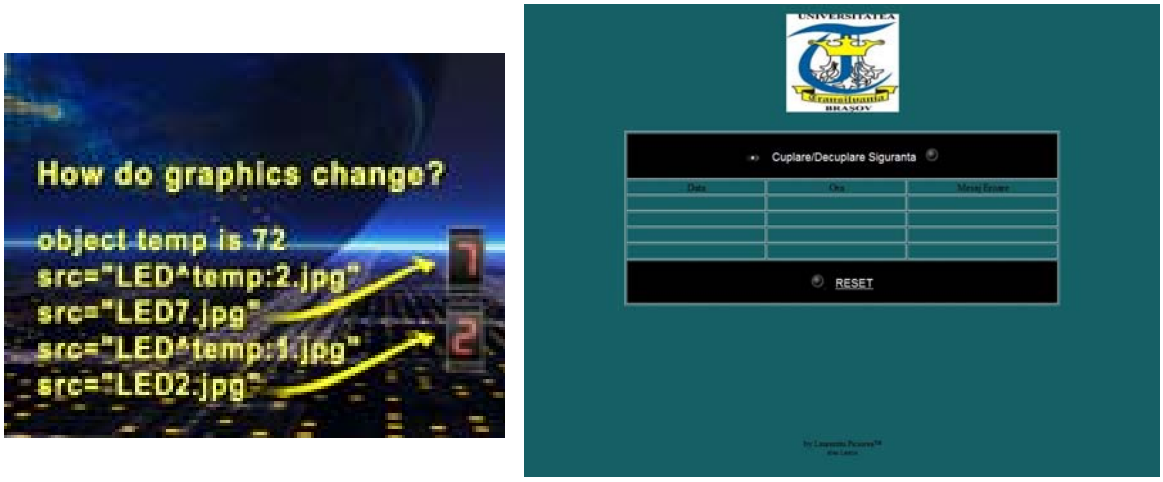


Figura 8.11. Exemple de pagini

9.Magistrala USB (Universal Serial Bus)

9.1.Descriere şi caracteristici

În prezent conectarea pe USB este cea mai răspândită variantă de conectare a dispozitivelor periferice la calculatoare PC. Printre perifericele cu USB se pot menționa imprimantele, stick-urile de memorie flash, aparatele foto, telefoanele mobile dar şi tensiometre, jucării sau fierbătoare de cafea, figura 9.1.

Figura 9.1. Exemple de dispozitive USB

Magistrala USB este bazată pe o transmisie serială, sincronă, entitatea este cadrul (blocul) de date, codificare de grup cu adăugare de biți, cu refacerea tactului din datele citite, verificarea corectitudinii transferului cu CRC.

Magistrala USB a fost introdusă cu dorința de a oferi utilizatorilor o interfață universală, cu viteză mare şi uşor de folosit, mai ieftină pentru că, fiind serială, cablurile şi conectorii costă mai puțin. Aceste considerente au impus magistrala USB pe piață, în prezent aceasta ocupând o cotă mare de piață în domeniul interfațărilor. Complexitatea USB este mai mare decât a interfețelor înlocuite, adică a interfeței RS232 şi a interfeței paralele, prin urmare implementarea ei în microcontrollere a fost mai dificilă. În 2008 au fost vândute peste 3 miliarde de dispozitive USB, iar intrarea pe piață a USB 3.0 (SuperSpeed USB) în anul 2009, cu o estimare de vânzări până în 2015 de 25% din totalul dispozitivelor USB asigură condițiile supraviețuirii îndelungate. Caracteristicile principale ale magistralei USB:

- rata de transfer este de 1,5 Mbps la USB 1.0 (Low Speed), 12Mbps la USB 1.1 (Full Speed), 480Mbps la USB 2.0 (Hi Speed) şi 4,8Gbps la USB 3.0 (Super Speed);

- conectează până la 127 de dispozitive la un calculator gazdă, într-o topologie de tip stea multiplă. Nu se pot conecta dispozitive USB fără gazdă (ceea ce este posibil la interfața IEEE 1394);
- configurarea este automată, adică se poate conecta un dispozitiv USB fizic în mers (Hot Plug In). Se remarcă creşterea complexității software față de partea hardware;
- cablul conține linii de alimentare, aşa că dispozitivele USB pot fi alimentate de la gazdă (bus powered device) sau pot avea alimentare proprie (self powered device). Din acest motiv cablurile au conectori diferiți pentru conectarea spre gazdă (upstream) şi spre dispozitiv (downstream);
- distanța de conectare este de maximum 5m, distanța se poate mări prin inserarea de hub-uri.

Specificațiile acestei magistrale descriu atributele de magistralã, definesc protocolul, tipurile de tranzacții, administrarea magistralei (bus management) şi totodată furnizează informații necesare pentru construirea unui sistem în acest standard.

Magistrala USB defineşte trei categorii de dispozitive fizice:

- gazda USB (USB Host)
- funcții USB (USB function)
- distribuitoare USB (USB Hub)

9.2.Arhitectura magistralei

Dispozitivele USB sunt conectate într-o topologie de tip stea multiplă. Topologia USB este reprezentată în figura 9.2. În nodul fiecărei stele se găseşte un hub.

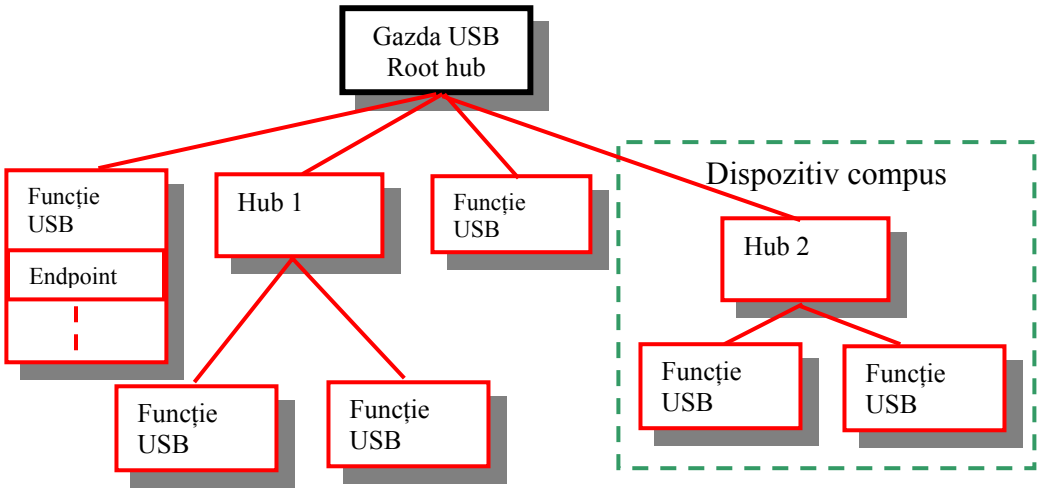


Figura 9.2. Arhitectura USB

Legătura este multipunct pe magistrală dar punct la punct între hub şi dispozitive. Este posibil ca un dispozitiv fizic să conţină mai multe funcţii şi un hub, acest dispozitiv numindu-se compus. Un exemplu este o multifuncţională care conţine imprimantă, scanner şi fax, toate acestea fiind funcţii USB.

Fiecare dispozitiv USB poate dispune de una sau mai multe endpoint-uri prin care gazda comunicã cu dispozitivul. Un endpoint este un registru intern, adresabil de gazdă în care se pot trimite sau din care se pot citi informaţii specifice. Toate dispozitivele posedă un endpoint special, *endpoint zero*, care este privit ca pipe de control. Pipe-ului endpoint zero îi este asociată informaţia ce descrie complet dispozitivul USB: clasa de dispozitiv, informaţii de power management, producător etc.

Iniţiatorul transferurilor de date pe magistralã este gazda USB. Protocolul folosit este protocol prin interogare (de tip polled). Datele vehiculate pe magistralã sunt grupate în pachete iar o tranzacţie de magistralã implicã transmiterea a cel mult trei pachete. Fiecare tranzacţie începe prin trimiterea de către gazdă a unui pachet de semnalizare -*token packet*- care descrie tipul şi sensul tranzacţiei, adresa dispozitivului USB şi numărul nodului destinaţie (endpoint). Dispozitivul adresat se selectează prin decodificarea adresei ce-i corespunde. Urmează transferul de date de la gazdă spre dispozitivul adresat sau invers, după cum este specificat în pachetul de semnalizare. Receptorul răspunde în această tranzacţie printr-un pachet de dialog -*handshake packet*- prin care se confirmã (sau nu) încheierea cu succes a transferului de date.

9.3.Nivelul fizic

Aspectele electrice şi mecanice ale interfeţei sunt reglementate foarte precis în specificaţiile de magistralã. Semnalele electrice sunt transmise diferenţial (D+ şi D-). Codificarea utilizatã este NRZI cu împănare de biţi (*bit-stuffing*) şi tactul de recepţie este generat din datele transmise, codul fiind autosincronizabil.

Cablul USB are patru fire, semnalul util este transportat pe două conductoare torsadate iar pe celelalte două conductoare cablul transportă tensiunea de alimentare nominală de +5V (V_{BUS}) şi potenţialul de referinţă (GND), figura 9.3.

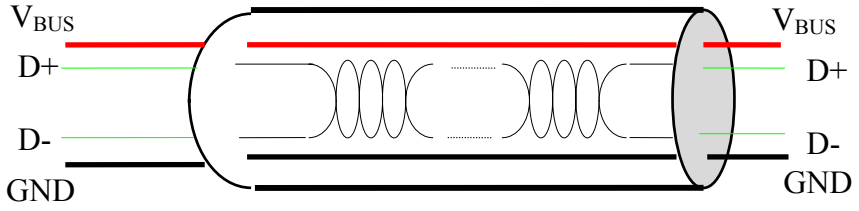


Figura 9.3. cablu USB

Tensiunea transmisă pe linie nu este tensiunea de alimentare a calculatorului gazdă ci este gestionată de controllerul USB, aşa încât o suprasarcină este detectată şi un mesaj de eroare este afişat de sistemul de operare. Câteva variante de cabluri USB sunt date în figura 9.4.

Figura 9.4. Cabluri USB, de la stânga spre dreapta: prelungitor, downstream-upstream, mini USB, USB 3.0

Uşurința cu care este utilizatã USB rezultã din atributul special de tip plug-and-play al acestei magistrale. USB acceptã cuplarea şi decuplarea de dispozitive în orice moment; sistemul software se adaptează dinamic la modificările fizice de topologie. Un dispozitiv USB este plasat fizic în structurã prin ataşarea la portul unui hub. Hub-ul dispune de indicatori de stare la fiecare port pentru a semnaliza cuplarea sau decuplarea unui dispozitiv. Gazda sesizează semnalizarea de la hub şi atribuie o adresã unicã dispozitivului. La decuplare hub-ul dezactivează portul şi indicã gazdei acest eveniment. Pentru a se adapta dinamic, sistemul software USB este permanent într-un proces de inventariere a magistralei (bus counting).

9.4.Transferul de date prin cadre

Arhitectura USB distinge patru tipuri de bază de transferuri de date:

- transferuri de control (*Control Transfers*) - sunt folosite pentru configurare şi comandã şi obligatoriu trebuie să fie suportate de toate perifericele;
- transferuri cu volum mare de date (*Bulk Data Transfers*) permit dispozitivelor să schimbe cantități mari de informație cu gazda pe măsură ce magistrala devine disponibilă, (ex.: camere digitale, scannere sau imprimante);
- transferuri prin întreruperi (*Interrupt Data Transfers*) a fost proiectat ca suport pentru periferice de intrare controlate de om, (tastaturã, mouse, joystick), care au nevoie să comunice rar, cantități mici de date. Datele transferate în acest mod sunt

caractere, coordonate sau semnalizări de evenimente organizate în unul sau mai mulţi octeţi;

- transferuri izocrone (*Isochronous Transfers*) - asigurã un acces garantat la magistralã, flux de date constant şi tolerează erorile de transmisie. Datele izocrone sunt continue şi în timp real la toate nivelele: generare, emisie, recepţie şi utilizare la receptor. Acest tip de transfer este folosit pentru fluxuri de transfer în timp real cum ar fi sistemele audio.

USB foloseşte un protocol bazat pe pachete de date (Data Packet). Un pachet de date este o colecţie de cadre de date (Data Frame). Numãrul de biţi dintr-un cadru nu are o valoare fixã. Biţii sunt trimişi spre magistralã astfel: primul bit este cel mai puţin semnificativ bit (LSB) din cadru, urmat de bitul mai semnificativ pânã la trimiterea celui mai semnificativ (MSB) bit din cadrul respectiv.

Protocolul USB defineşte patru tipuri de pachete de date:

- pachet de semnalizare (Token Packet)
- pachet de date (Data Packet)
- pachet de dialog (Handshake Packet)
- pachet special (Special Packet)

a. Pachetul de semnalizare (Token Packet)

Orice transfer începe prin trimiterea de cãtre gazdã a unui pachet de semnalizare. Un pachet are 32 de biţi împãrţiţi în cinci câmpuri, figura 9.5.

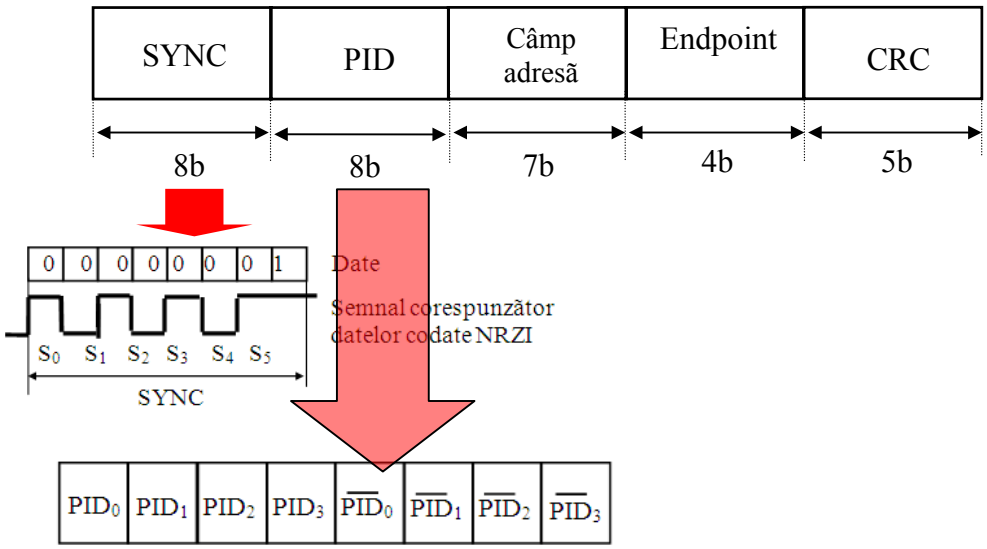


Figura 9.5. Pachetul de semnalizare (sus), câmpul de sincronizare (mijloc) şi câmpul PID (jos)

Toate pachetele conțin la începutul lor un câmp de sincronizare, numit SYNC care permite buclei PLL pentru refacerea tactului din receptor să se sincronizeze, şi un câmp identificator de pachet, numit PID (Packet Identifier).

SYNC este primul câmp din orice pachet USB. Câmpul de sincronizare este constituit dintr-o serie de biți care produc un şir dens de tranziții utilizând schema de codificare NRZI cerută de standardul USB. Câmpul apare ca o serie de trei tranziții 1/0 urmată de o marcă cu lățimea a două impulsuri. Datele din câmp au succesiunea de valori 0000 0001. Câmpul PID urmează câmpului SYNC într-un pachet USB şi are lungimea de 8 biți. Primii patru biți indică tipul pachetului, iar următorii patru sunt în ordine primii patru complementați (complement față de 1) şi sunt folosiți ca biți de verificare pentru a confirma acuratețea primilor patru. Câmpul PID defineşte trei categorii de pachete handshake:

- Pachetul handshake ACK indică emițătorului că pachetul de date a fost recepționat fără erori;
- Pachetul handshake NAK indică faptul că o funcție nu a fost capabilă să recepționeze date de la gazdă (într-o tranzacție OUT) sau că o funcție nu are date de transmis gazdei (într-o tranzacție IN). O gazdă nu poate trimite niciodată NAK;
- Pachetul STALL este emis de o funcție ca răspuns la un pachet de semnalizare IN sau după o tranzacție de date OUT, indicând că funcția nu este capabilă să emită sau să recepționeze date. Gazda nu poate răspunde cu pachet STALL.

b.Pachetul de date şi pachetul handshake

Informația propriu-zisă este transferată în sistemele USB sub forma unor pachete de date, figura 9.6. După câmpurile SYNC şi PID urmează câmpul de date care este compus dintr-un număr întreg de octeți, de la 0B la 1023B. Corectitudinea câmpului de date este asigurată prin câmpul de verificare ciclică CRC de 16b aflat la sfârşitul pachetului.

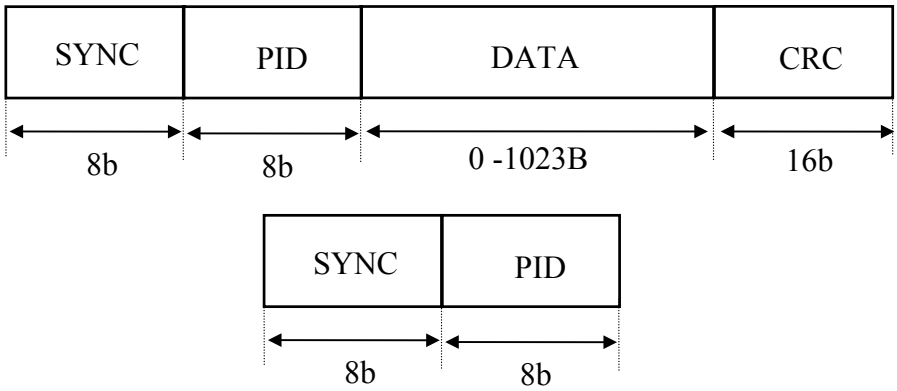


Figura 9.6. Structura pachetelor de date (sus) şi handshake (jos)

Pachetele handshake sau de dialog sunt folosite pentru a raporta starea unui transfer de date, pentru a indica recepţia cu succes a datelor sau pentru a întoarce valori care indicã acceptarea/ respingerea unei comenzi sau o stare de HALT la dispozitiv. Acest tip de pachet este compus doar din douã câmpuri; SYNC şi PID. Structura este reprezentatã în figura 9.6 jos.

Un dialog simplificat este dat în figura 9.7. Cu un pachet de semnalizare se indică adresa dispozitivului şi tipul tranzacţiei care presupunem că este un transfer de date spre gazda USB. Dispozitivul USB răspunde cu pachetul de date. Gazda confirmă primirea cu un pachet handshake.

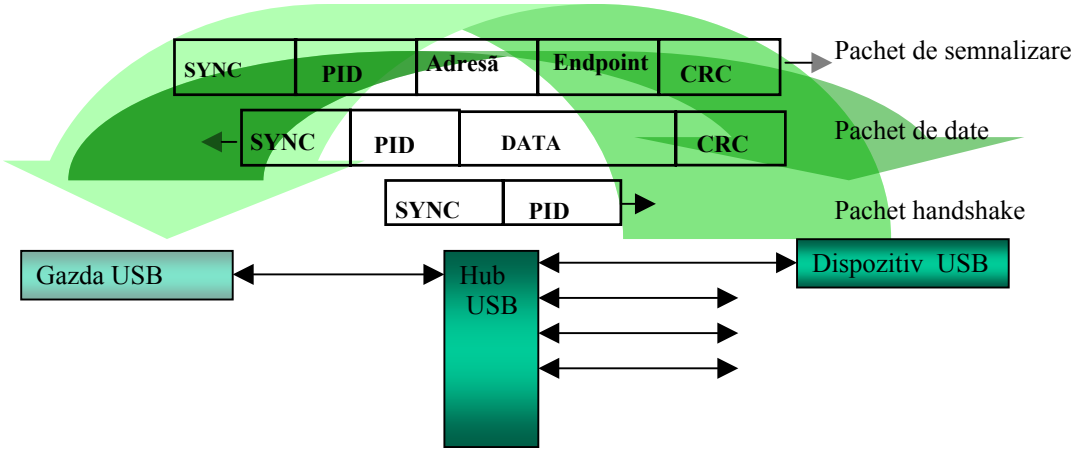


Figura 9.7. Dialog USB simplificat

9.5. Cuplarea unui microcontroller (MC) la USB printr-o interfaţă specializată

Dacă aplicaţia necesită cuplarea unui MC la USB atunci există două variante, utilizarea unei interfeţe specializate aşa cum sunt cele din familia FTDI sau alegerea unui MC care are interfaţă USB integrată.

Circuitele FTDI [2] cel mai cunoscute sunt cele de conversie USB-RS232 FT8U232AM (USB 1.1) şi FT8U232BM (USB 2.0) şi cele de conversie USB-paralel FT8U245AM (USB 1.1) şi FT8U245BM (USB 2.0). Protocolul USB este încorporat total în circuit şi nu este nevoie de programarea formării sau gestionării cadrelor USB.

Emiţătorul / receptorul USB transmit /recepţionează datele USB. Motorul serial codifică / decodifică datele, asamblează cadrul USB, inserează sau verifică CRC. Datele sunt convertite în format paralel şi sunt transferate printr-un protocol paralel simplu.

Un generator de tact de 6MHz cu un cristal în exterior generează semnalul de tact, care este multiplicat de 8 ori şi constituie tactul intern al circuitului. Un generator de 3,3V alimentează blocurile interne dar tensiunea generată poate fi folosită şi în exterior. EEPROM-ul serial memorează date privitoare la configurația circuitului.

Datele în format paralel pot fi citite sau scrise printr-un protocol controlat de semnalele RD, WR, TxE şi RxF dar pot fi transferate automat cu o periodicitate dată de un timer intern, ceea ce face posibile aplicații în care FTDI nu este cuplat în partea paralelă la un microcontroller ci la un simplu element de execuție sau traductor. Acest mod de lucru se numeşte Bit Bang.

Schema bloc a circuitului 245BM este dată în figura 9.8.

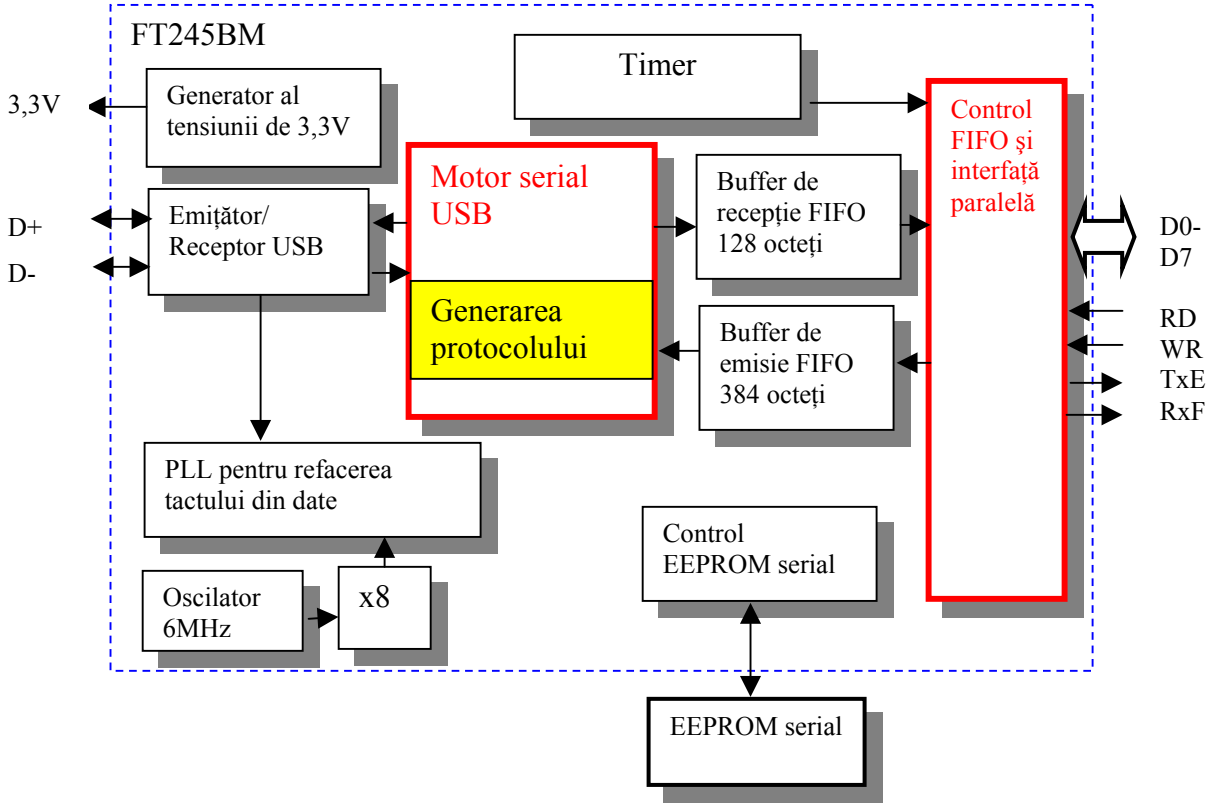


Figura 9.8. Schema bloc a unui circuit FT245BM

Scrierea datelor se face când TxE este în stare 0 logic. După memorarea octetului în bufferul de transmisie TxE devine din nou 0 logic. La recepția datelor se foloseşte RxF care în stare 0 logic anunță că s-a recepționat un caracter, figura 9.9.

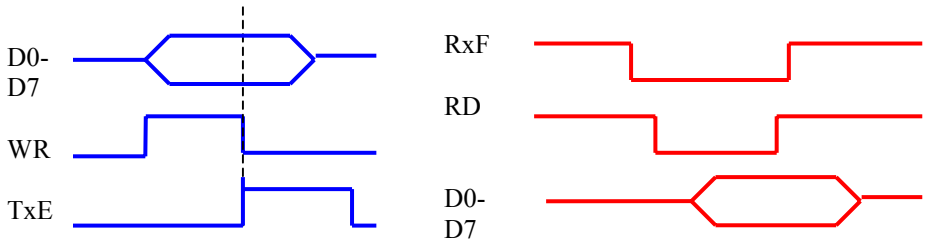


Figura 9.9. Scrierea / citirea datelor în mod paralel

Circuitul FT232BM are o schemă bloc asemănătoare, diferența fiind blocul de interfață care este în acest caz serială. Semnalele sunt cele de la RS232: TxD, RxD, RTS, CTS, DTR, DSR, DCD, RI şi în plus TxDEN un semnal de validare transmisie necesar la standardul RS485.

Două semnale care arată că se transmit sau se recepționează date TxLED şi RxLED pot să fie folosite la comanda unor indicatoare luminoase de activitate. Protocoalele permise sunt cele hard DTR sau CTS şi soft Xon-Xoff. Un circuit generator de rată de Baud asigură tactul standard necesar transmisiei.

Interfața cu microcontrollerul este simplă şi uşor de implementat, constă ca şi hardware în conectarea câtorva semnale, TxD cu RxD la FT232BM şi cele 8 linii de date şi 4 de protocol la FT245BM.

Interfața cu calculatorul gazdă, de regulă un PC înseamnă în primul rând descărcarea driverelor de pe site-ul FTDI pentru sistemul de operare instalat. Cuplarea circuitului FTDI la USB (în cazul sistemului de operare WINDOWS) va avea ca efect dialogul "Found new hardware…). După instalarea driverelor circuitul FTDI va apărea ca în figura 9.10:

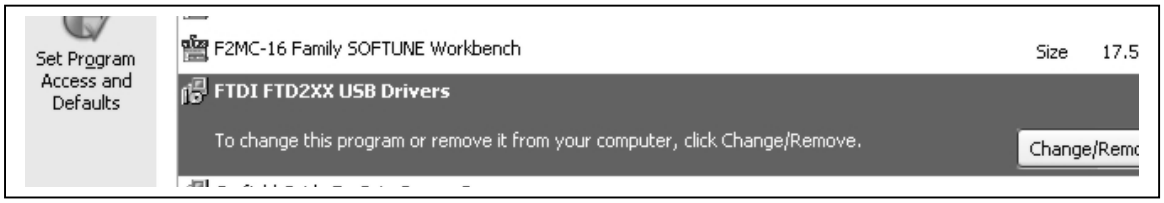


Figura 9.10. Driverul software instalat sub WINDOWS

Pentru transferul datelor FTDI pune la dispoziția utilizatorului o bibliotecă DLL şi documentație pentru programare (API). Astfel, o scriere / citire de date prin USB în FTDI se poate face cu o comandă FT_write / FT_read.

9.6. Microcontrollere cu USB integrat

Un model de microcontroller cu USB integrat este ATMEL AT90USB care este disponibil în diverse combinații de memorie. Interfața USB are următoarele caracteristici:

- Viteza este de 1,5 Mbps la USB 1.0 (Low Speed), 12Mbps la USB 1.1 (Full Speed);
- Conține 7 endpoint-uri cu dimensiunile de 64 octeți (endpoint 0, de control), 256 octeți (endpoint 1) şi câte 64 octeți celelalte;
- Conține o memorie dual port DPRAM de 832 de octeți pentru endpoint-uri.

Schema bloc a interfeței USB integrate este dată în figura 9.11:

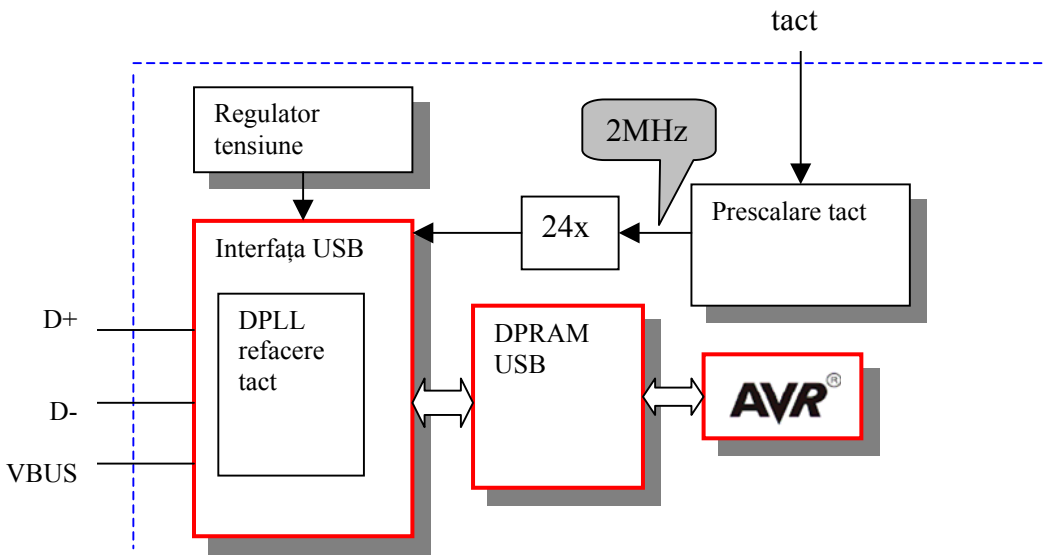


Figura 9.11: schema bloc a interfeței USB

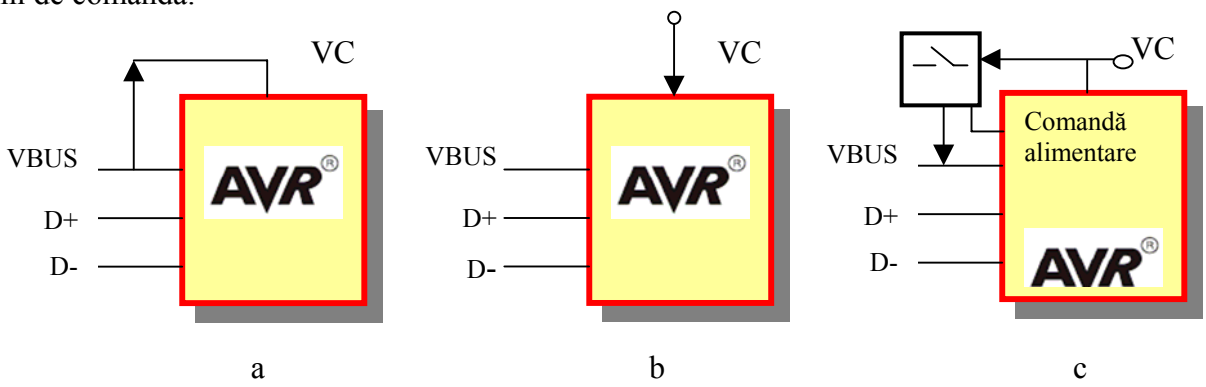
Tactul necesar pentru interfața USB este de 48MHz cu toleranța de 0,25%, deci trebuie acordată atenție tactului extern şi prescalării. Circuitul PLL este digital (DPLL) pentru a asigura performanțele necesare refacerii tactului. Regulatorul de tensiune asigură tensiunea necesară pentru D+ şi D- (3V...3,6V) din tensiunea de alimentare a circuitului

(5V). MC admite lucrul OTG (On-The-Go) conform cu suplimentul specificațiilor USB, care permite transmisia de semnalizări pe lina VBUS.

Modurile în care lucrează MC pot fi:

- Funcție (ATMEL denumeşte dispozitiv USB) cu alimentare de la USB (Figura 9.12 a);
- Funcție cu alimentare proprie (Figura 9.12 b);
- Gazdă USB (Figura 9.12 c), mod care nu este implementat în toate MC din familie.

Modul de lucru gazdă sau funcție şi vitezele de transfer se selectează cu valori logice la pini de comandă.



a b c

Figura 9.12: modurile de lucru ale MC

Pe liniile de date D+ şi D- se inserează rezistoare serie de 22Ω.

Transmiterea datelor pe aceeaşi linie dar din surse diferite prevăzute cu buffer-e şi spre destinații diferite fluidizează traficul. Mărimea endpoint-urilor poate fi programată în MC, cu anumite condiționări. Fiecare endpoint poate cere o întrerupere atunci când este plin (la recepție) sau gol (la emisie). Registrele de programare a USB au semnificații diferite pentru modul de lucru gazdă sau funcție.

MC are posibilitatea de lucru cu economie de energie în următoarele moduri:

- Mod *Idle*, în care CPU este oprit, repornirea fiind posibilă la orice întrerupere USB;
- Mod *Power Down*, în care CPU şi periferia sunt oprite, repornirea fiind posibilă doar la anumite întreruperi USB;
- Mod *Freeze Clock*, în care tactul pentru USB este oprit. Intrarea în acest mod se poate comanda printr-un bit într-un registru de comandă USB. Repornirea este posibilă doar la anumite întreruperi USB.

Transferul de date bazat pe endpoint-uri şi pipe-uri este reprezentat în figura 9.13:

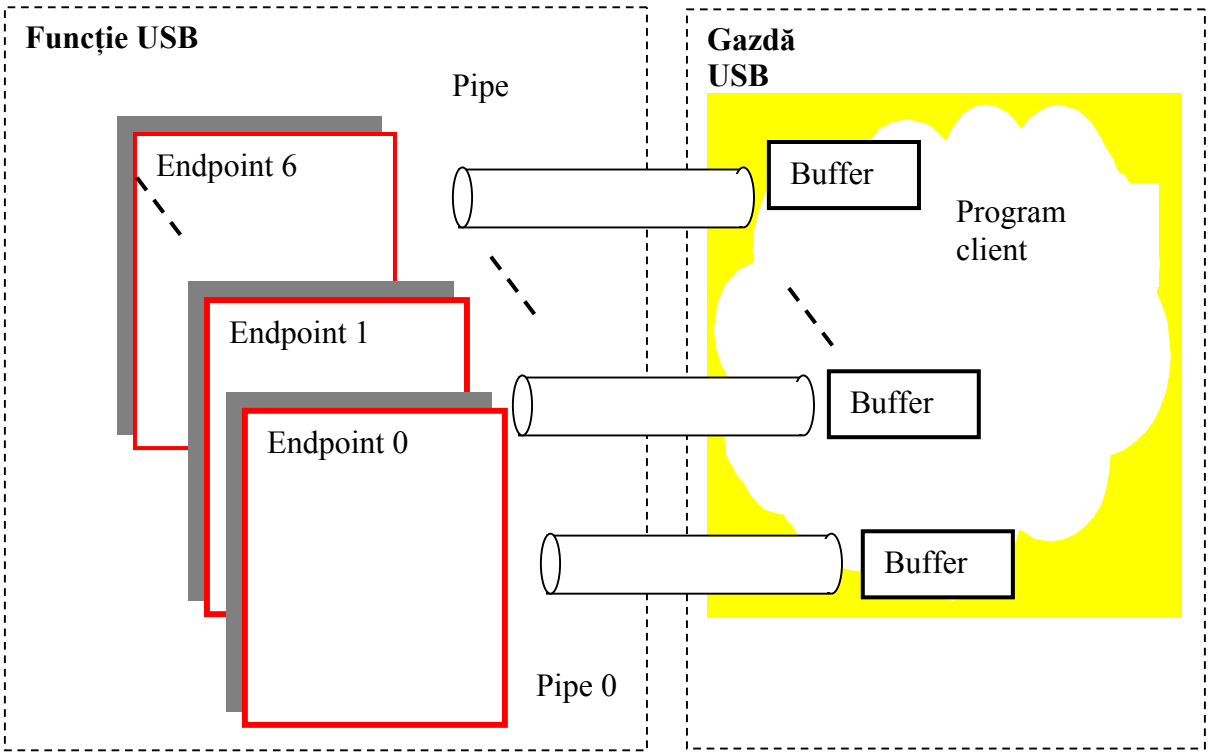


Figura 9.13. Transferul de date bazat pe endpoint-uri

Modul de lucru ca gazdă USB sau dispozitiv USB este determinat de valoarea logică a unui pin (UID) sau software prin poziționarea unui pin într-un registru USB. Modul de lucru Low Speed sau Full Speed se poate selecta prin valoarea logică a doi pini externi. Valorile logice de comandă pot fi stabilite local cu rezistoare sau de la distanță.

În funcție de modul de lucru ales pentru interfața USB, software-ul trebuie să comande următoarele acțiuni:

1.Pornirea interfeței USB

- pornirea regulatorului de tensiune;
- configurarea PLL, validarea PLL şi comanda unui întârzieri pentru ca PLL să se sincronizeze;
- validarea şi configurarea interfeței USB prin alegerea vitezei, configurarea endpoint-urilor etc.;
- ataşarea unui dispozitiv USB.

2.Oprirea interfeței USB:

- detaşarea dispozitivelor USB;
- invalidarea interfeței USB, a circuitului PLL şi a regulatorului de tensiune.

Intrarea în modul de lucru cu economie de energie se face astfel: se opreşte tactul, se opreşte bucla PLL, se validează întreruperile care scot interfața din acest mod de lucru, se opreşte CPU. Reintrarea în modul de lucru se face cu aceleaşi operații în ordine inversă. Ca exemplu de transfer de date prin USB, în figura 9.14 se arată diagrama de timp pentru un transfer de date de scriere:

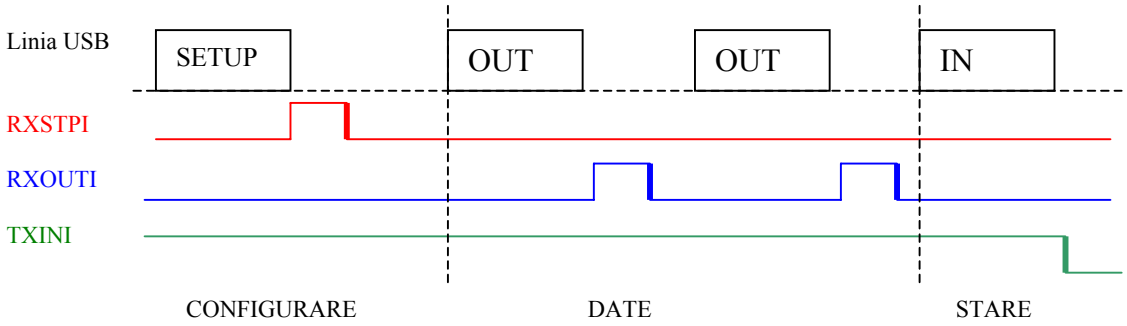


Figura 9.14. Scriere USB

Un pachet de semnalizare este generat pentru stabilirea destinației transferului. Se generează o întrerupere RXSTPI (generată la transmisia pachetului de semnalizare). Prin software se şterge informația din endpoint-ul transmis şi se transmit pachete de date de ieşire. După fiecare pachet de date transmis se generează o întrerupere RXOUTI pentru a informa MC de transmisia pachetului şi a putea şterge conținutul endpoint-ului folosit. După recepționarea pachetelor de date dispozitivul destinație răspunde cu un pachet de dialog pentru a confirma primirea datelor. Se generează un semnal TXINI în zero pentru a permite recepționarea unui alt pachet de dialog. Fronturile îngroşate sunt cele generate prin comenzi software iar cele neîngroşate sunt generate hardware ca întreruperi.

10.Interfețe pentru comunicații wireless

10.1.Introducere

Multe dintre aplicațiile cu microcontroller necesită o transmisie de date fără fir. Comunicațiile fără fir înseamnă transferul informației prin intermediul câmpului electromagnetic în gama de frecvență 9kHz-300GHz. Spectrul electromagnetic este o resursă publică şi alocarea gamelor de frecvențe pentru diferite transmisii se realizează de către organisme naționale şi internaționale. În spectru există benzi de frecvență pentru care trebuie licență şi benzi libere, aşa cum este banda ISM (Industrial, Scientific and Medical - 2,4GHz).

Istoricul transmisiilor fără fir începe cu David E. Hughes care a transmis codul Morse in 1878 cu o bobină parcursă de curent electric şi a patentat invenția. În urma acestui patent s-a născut compania Western Union Telegraph. Thomas Alva Edison (1847 – 1931), un prolific inventator, care a inventat becul cu incandescență, microfonul etc. a inventat în 1888 o metodă de transmisie fără fir cu ajutorul unui magnet vibrator, invenție patentată şi aplicată la căile ferate. Heinrich Rudolf Hertz (1857 –1894) a avut realizări importante în domeniul teoriei câmpului electromagnetic. A demonstrat teoretic posibilitatea transmisiei radio dar nu a realizat experimente. Nikola Tesla (1856 – 1943) a avut realizări experimentale deosebite, cum ar fi de exemplu realizarea unui model de vapor telecomandat de la distanța de câțiva kilometri, a construit antene, sisteme de transport al energiei fără fir etc.

Actrița Hedy Lamarr, o vestită frumusețe de la Hollywood, numită de contemporani cea mai frumoasă femeie a lumii a devenit vestită în urma filmului Ecstasy, unde apărea nud, prima apariție nud în cinematografie. În cel de-al doilea război mondial rachetele şi torpilele începuseră să fie ghidate prin radio, dar bruierea semnalului făcea ca ghidarea să nu reuşească. Hedy Lamarr care a privit multă vreme activitatea navelor în porturi a inventat în 1942 un mod de a schimba frecvențele de transmisie foarte repede în timpul transmisiei pentru ca transmisia să nu poată fi bruiată. Schimbarea frecvenței se făcea prin programul înscris pe un tub, ca şi cel al flaşnetei. Invenția a fost făcută cu 20 de ani prea devreme. Prima aplicație a fost realizată de armata SUA în timpul crizei rachetelor din Cuba, metoda de salt de frecvență fiind folosită la ghidarea rachetelor. Astăzi, metoda este folosită la WLAN, Bluetooth, ghidarea rachetelor, comunicații prin satelit etc. Această invenție istorică, cunoscută de puțină lume dovedeşte că pasiunea pentru noutate şi patriotismul nu sunt doar vorbe.

Există în prezent o mare varietate de transmisii fără fir. Alegerea uneia sau alteia se face în funcție de aplicație. Dacă este nevoie ca transferul de date să fie între două module cu

microcontroller la distanță mică şi aplicația trebuie să fie ieftină atunci se pot folosi protocoale proprietare implementate în module ieftine. Dacă este vorba de un aparat care transmite date direct în Internet şi este situat mai departe de oraş atunci se poate folosi o transmisie GPRS. Dacă receptorul de date este un PDA sau un telefon mobil şi distanța de transmisie este mică atunci o transmisie Bluetooth este cea mai potrivită. Necesitatea integrării într-o rețea de măsură existentă poate obliga proiectantul să aleagă metoda folosită în rețea, de exemplu ZigBee. O transmisie cu debit mare de informație poate determina alegerea unei transmisii WLAN, costurile fiind însă mai mari ca la variantele anterioare. În consecință, principalele criterii de alegere a metodei de transmisie sunt:

- Distanța de transmisie;
- Debitul de informație;
- Restricții determinate de conectarea la o rețea existență;
- Poziția geografică;
- Costurile admisibile.

Partea de programare a modulelor wireless este uşurată dacă se folosesc comenzi AT. Comenzile AT sunt şiruri de date care încep cu prefixul AT şi pot fi trimise modulului cu un program cum este de exemplu Hyper-Terminal din Windows dacă modulul este conectat la un PC prin interfața RS232 sau direct de la microcontroller. Conectarea la un PC este utilă în faza inițială de punere la punct a părții software. După ce programul a fost pus la punct se programează microcontrollerul care comunică cu modulul GPRS tot prin interfața serială RS232, cu aceleaşi comenzi AT.

10.2.Transmisii cu protocoale proprietare

Pentru a realiza o transmisie simplă de date la distanțe de câteva zeci de metri, cu debit mic de informație şi care nu trebuie să fie interconectată printr-un anumit standard într-o rețea de date se pot folosi module sau interfețe cu protocol proprietar. Consultând pagina web a unei firme de componente (www.adelaida.com) se pot găsi multe asemenea module, cu prețuri variind între 6 şi 50 de dolari la cumpărarea unei bucăți.

Perechea de circuite RFM01 (receptor) şi RFM02 (emițător), figura 10.1 sunt echipate cu interfață SPI, lucrează în banda 433MHz iar debitul de informație maxim este de 115,2Kbps la o distanță de maximum 300m. Modulația datelor este FSK, receptorul conține o buclă PLL şi poate fi alimentat între 2,4V şi 5,4V, făcând posibilă realizarea de module portabile, alimentate de la două baterii. Dimensiunile mici de 18mm x 14mm x 9mm asigură posibilitatea miniaturizării aplicațiilor. Receptorul are protecție la subtensiune şi poate asigura reglajul automat al unor parametri ai antenei.

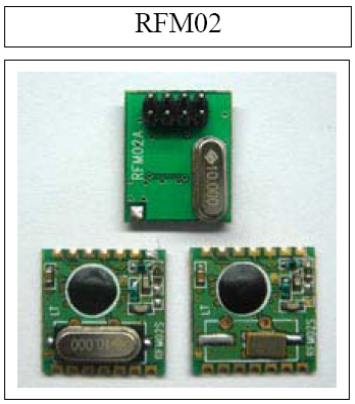


Figura 10.1. Circuite RFM (sursa http://www.adelaida.ro/module-comunicatii-radio-rf/)

Schemele simple a receptorului din figura 10.2 şi a transmiţătorului în figura 10.3 arată simplitatea interconectării cu un microcontroller din familia ATmega.

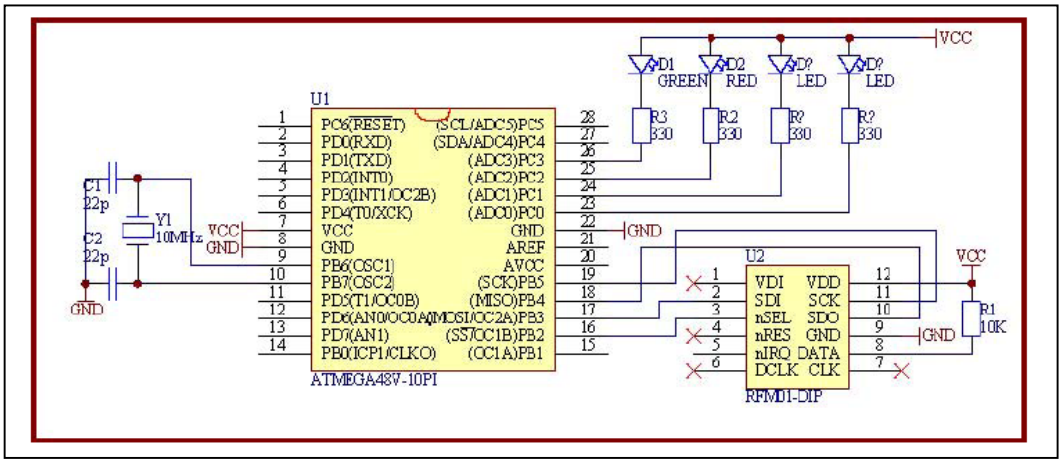


Figura 10.2. Receptor cu RFM01 (sursa www.mikrocontroller.net)

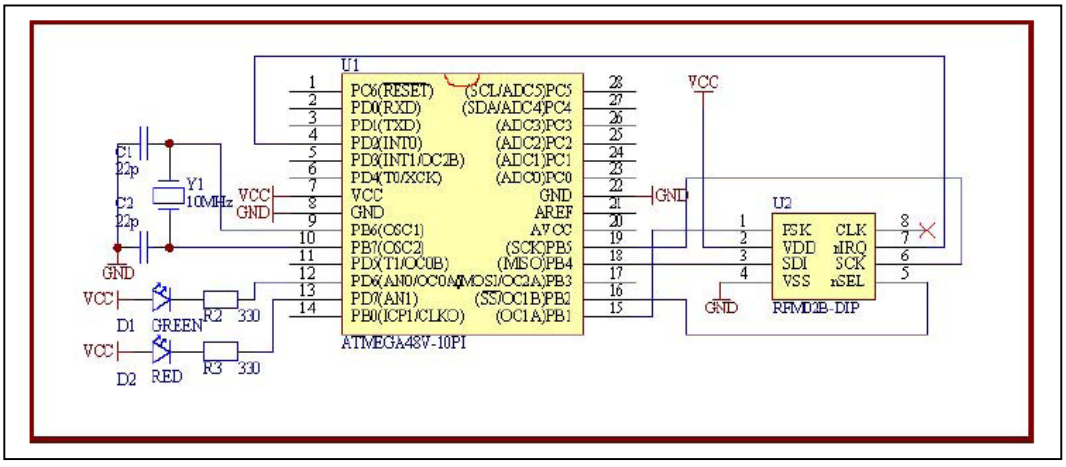


Figura 10.3. Transmiţător cu RFM02 (sursa www.mikrocontroller.net)

Poate cea mai simplă soluție este folosirea perechii de circuite hibride TLP434A/ RLP434, figura 10.4:

Figura 10.4. Circuite TLP434A (stânga) şi RLP434 (dreapta), sursa http://www.coolcircuit.com/project/rf_remote/

Frecvența de lucru este 433,92MHz, debitul maxim este de 4,8Kbps la o distanță de maximum 200m cu o antenă adaptată. Tensiunea de alimentare este de 3V-12V iar interfața cu microcontrollerul este serială. Modulația datelor este ASK iar prețul unui circuit este sub 6 dolari. Un proiect realizat cu această pereche de circuite este dat în http://www.coolcircuit.com/project/rf_remote/ , datele seriale transmise fiind codate Manchester de un microcontroller PIC12F509. Schema este extrem de simplă.

O altă variantă este circuitul HM-TR (figura 10.5) construit de Hope Microelectronics Co. Ltd. (http://www.hoperf.com/) care conține un receptor şi un transmițător pe aceeaşi placă care pot asigura comunicația half duplex între două puncte, interfața fiind RS232.

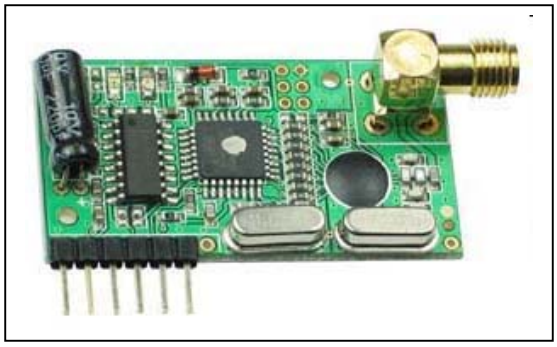


Figura 10.5: Circuitul HM-TR, RS232, half duplex, sursa http://www.hoperf.com/

Circuitul lucrează în banda de frecvenţe între 310,24Mhz şi 929,27MHz, deci se poate lucra la 433MHz sau 868MHz., modulaţia fiind FSK. Debitul maxim este de 19,2Kbps, dimensiunile modulului fiind 24x43mm. Distanţa maximă determinată într-un spaţiu fără obstacole este de 330m. Denumirea circuitului HM-TR 433/RS232 sau HM-TR 866/TTL sugerează frecvenţa de lucru şi nivelele de tensiune de interfaţă (RS232 sau TTL).

10.3.Transmisia datelor prin GPRS

Datorită necesităţii de a putea avea acces la informaţie şi de a fi totodată mobil, s-a recurs la folosirea terminalelor mobile GSM pentru transmisii de date. Comunicaţiile de date prin intermediul reţelelor de telefonie mobilă au devenit mult mai eficiente în momentul în care a fost împrumutată o idee de la reţelele de calculatoare, cea a comutaţiei de pachete. Informaţia este încapsulată în pachete care circulă prin intermediul unor echipamente de reţea până la destinaţie. Adresele sursei şi destinaţiei sunt conţinute în pachet. Astfel a luat naştere GPRS (General Packet Radio Services).

Rata de transfer maximă care se poate obţine prin GPRS este de 171,2 kbps adică 21.4kBps. Transferul datelor poate fi efectuat prin UDP (User Datagram Protocol), sau prin TCP/IP. Avantajul transferului prin TCP/IP constă în faptul ca pachetele ajung la destinaţie în ordinea în care au fost transmise şi există o garanţie a transmisiei corecte a pachetelor.

Sistemul de transmisie GPRS este pus la dispoziţie de operatorii de telefonie mobilă şi datele achiziţionate sunt trimise la un server al utilizatorului. Fiecare modul GPRS trebuie să aibă un card SIM furnizat de operatorul de telefonie mobilă cu un tip de abonament sau în sistem pre-plătit pentru transferul de date. Tarifele sunt de regulă funcţie de traficul realizat.

În aplicaţiile realizate la proiecte de diplomă au fost folosite două tipuri de module GPRS, produse de TELIT, modelul GM862-GPRS şi modelul EZ10. Ambele modele sunt echipate cu interfeţe RS232, figura 10.6:

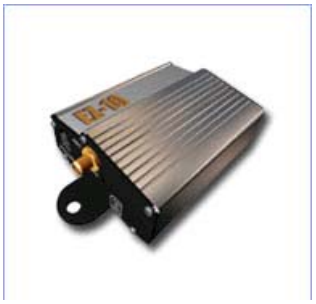


Figura 10.6. Modulul GPRS GM862 (stânga) şi EZ10 (dreapta) (sursa **www.telit.com**)

EZ10 este un modul GPRS/GPS construit ca ansamblu separat de placa cu microcontroller la care se conectează prin o conexiune serială RS232. Modulul este construit pe baza circuitului GM862, având suplimentar şi funcţia de GPS. EZ10 administrează intern stiva TCP-IP şi uşurează astfel implementarea aplicaţiei. Alimentarea modulului se face de la un alimentator extern. Caracteristicile principale ale modulelor GPRS sunt: Dual Band 900-1800MHz, EASY GPRS (comenzi AT incluse), RS232 UART, nivele CMOS la GM862 şi RS232 la EZ10, auto-bauding de la 2.4 până la 57.6 Kbps, interfaţă card SIM, 3V şi 1.8V, Maxim 13 x GPIO porturi, 2 convertoare A/D, agendă numere de telefon, este suportat codul PUK2 pentru condiţia de blocare, audio integrat, posibilitatea de lucru cu SMS, GPS integrat (în EZ10).

Conexiunea modulului TELIT cu microcontrollerul se face prin interfaţa serială RS232, comunicaţia fiind bazată pe comenzi AT. La activarea conexiunii GPRS trebuie specificaţi parametrii reţelei şi numărul de telefon apelat şi se stabileşte o conexiune între modem şi un server de date (nu se poate stabili o conexiune între două modemuri GPRS). Conectarea cu o aplicaţie aflată pe un server se realizează astfel:

- Cu o comandă AT se setează proprietăţile GPRS pentru a permite modemului GPRS să activeze conexiunea GPRS ori de câte ori este nevoie de un transfer de date;
- Cu o comandă AT se setează parametrii de autentificare, nume utilizator şi parolă care vor fi folosiţi pentru validarea conectării;
- Cu o comandă AT se definesc portul de conectare la server şi protocolul UDP sau TCP-IP;
- Se porneşte conexiunea cu o comandă de formare a numărului şi conectare.

Comenzile utilizate la conectare sunt:

1. **AT-** Comandă vidă, va întoarce întotdeauna răspunsul OK
2. **AT#USERID[=<user>]** – se trimite numele reţelei pentru autentificare, folosită sub forma **AT#USERID="net.vodafone.ro“** (autentificare)
3. **AT#PASSW= <pwd>** - se trimite parola pentru autentificare, folosită sub forma A**T#PASSW="vodafone"** (parolă)
5. **AT+CPIN[=<pin> [,<newpin>]] –** se trimite codul PIN**,** folosită sub forma **AT+CPIN=2649** (cod PIN)
6. **AT+CREG=?** - Dupa ce se introduce codul PIN al cartelei SIM se asteapta până când se efectuează conectarea. Comanda a fost folosita în forma: **AT+CREG?**
7. **AT+CGDCONT=1 -** Se utilizează o conexiune IP prin serverul GGSN cu numele „net.vodafone.ro” fără o compresie a datelor sau a header-ului pachetelor. Comanda a fost folosită în forma: **AT+CGDCONT=1,"ip","net.vodafone.ro","0.0.0.0",0,0**

9. **AT#GPRS[=[<mode>]]** – activare GPRS cu 1, dezactivare cu 0, transmisia a fost activată cu **AT#GPRS=1** . AT#GPRS? interoghează starea modemului
10. **AT#SKTD=0 -** Tipul socketului folosit este TCP, numărul portului pe care ascultă serverul este 2222, adresa de IP a serverului este „86.125.93.184" iar conexiunea se închide când serverul închide portul. Comanda a fost folosită în forma: **AT#SKTD=0,2222,"86.125.93.184",0**

După această secvență urmează transmiterea datelor. Practic tot ce primeşte modulul prin USART va fi transmis către server urmând ca după închiderea conexiunii modulul să fie trecut în stare oprită. O listă completă a comenzilor AT se poate găsi în documentațiile TELIT, de exemplu în (www.semiconductorstore.com/pdf/newsite/Telit/GM862-GPS/GM862-GPS_Software_User_Guide_r4.pdf). Oprirea sa se va face hardware. Transferul de date între modulul cu microcontroller şi modemul GPRS are loc în aplicațiile realizate astfel:

- Unitatea de procesare selectează prin intermediul circuitului de selecţie ieşirea USART-ului către modemul GPRS.
- Unitatea de procesare activează modemul.
- Unitatea de procesare inițializează înregistrarea în rețeaua GSM a modemului şi conectarea la GPRS.
- Unitatea de procesare transmite efectiv pe USART datele de transmis. Aceste date vor fi transmise prin intermediul modemului la server-ul cu o anumită adresă IP. Pe acest server este rulat un program ce ascultă şi primeşte pe portul 2222 pachete prin TCP/IP. Aceste pachete conțin ca identificator codul IMEI (International Mobile Equipment Identity – Identitatea Internațională a Echipamentului Mobil) al modemului, cod ce este unic. Ca măsură de siguranță, acest cod este căutat într-o listă în care se află toate codurile IMEI ale echipamentelor ce vor fi utilizate, iar dacă codul IMEI nu se află în lista echipamentelor va închide conexiunea.
- Unitatea de procesare dezactivează modemul;

Aplicațiile GPRS sunt mai simplu de implementat decât pare la prima vedere, aceasta datorită în primul rând programabilității medemurilor GPRS cu comenzi AT (Easy GPRS). Nu este necesară cunoaşterea traseului datelor până la serverul utilizatorului, singura condiție pusă serverului fiind să aibă atribuit un IP fix. Transmisia GPRS se pretează la aplicații în care nu există un receptor în apropierea punctului de culegere de date dar există acoperire de telefonie mobilă.

10.4. Bluetooth

Prin Bluetooth se realizează transferuri de date pe distanțe scurte între un calculator şi diverse echipamente periferice, de exemplu căşti, telefoane mobile, playere, imprimante,

camere video, GPS etc. Numele provine de la numele unui rege danez, Blatand din secolul 10 care a unit triburile scandinave. Modulaţia datelor este asemănătoare cu cea de la WLAN, adică împărţirea spectrului alocat în mai multe canale şi o modulaţie GFSK (Gaussian Frequency-Shift Keying) a datelor pe fiecare canal. În modul de transmisie de date salturile de frecvenţă sunt 1600/s, iar în modul de descoperire poate fi de 3200/s pentru a micşora timpul de conectare. Spectrul alocat este situat în banda de 2,4GHz, între 2,402GHz şi 2,480GHz. Distanţa de transmisie este de uzual de 1m, viteza fiind de 1Mbps, dar există unele dispozitive cu putere mai mare de emisie care asigură distanţe până la 100m. Conectarea unui dispozitiv Bluetooth la un calculator gazdă se realizează printr-un software de descoperire.

Există două variante de implementare a unui sistem înglobat cu transmisie Bluetooth, prin utilizarea unei interfeţe Bluetooth conectate la un microcontroller existent sau utilizarea unui microcontroller cu Bluetooth integrat. Momentan prima soluţie este preferată asigurând o viteză de implementare mai mare şi costuri mai mici.

Interfaţa Bluetooth de la Rayson BTM222. figura 10.7. conţine un nucleu BlueCore4 cu interfeţe de conectare SPI, UART, USB şi o interfaţă PCM (Pulse Code Modulation) pentru conectarea unui modul audio. De la nucleu datele sunt emise printr-un amplificator de putere prin antenă, iar datele recepţionate sunt amplificate cu un LNA. Alimentarea de putere şi cea a nucleului sunt diferite.

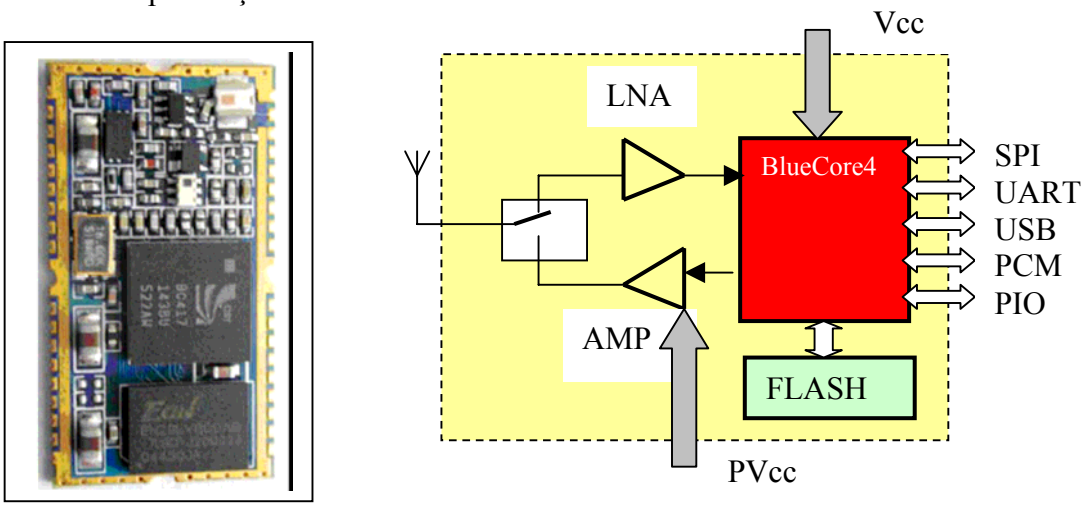


Figura 10.7. Interfaţa Bluetooth, aspect (stânga) şi schema bloc (dreapta) (sursa http://www.tme.eu/ro/details/btm-222/module-bluetooth/rayson/)

Interfaţa asigură transfer Bluetooth versiunea 2 cu EDR (Enhanced Data Rate) de până la 3Mbps. Sunt posibile moduri de lucru cu economie de energie. Alimentarea este între 3V şi 3,6V, puterea de emisie fiind de 18dBm.

Un alt modul Bluetooth care poate fi utilizat în aplicații cu microcontrollere este adaptorul serial LM058, figura 10.8. Acest adaptor este conform cu specificațiile v2.0+EDR, şi asigură o distanță de transmisie de 100m, viteza maximă fiind de 115,2kbps dar şi 230,4kbps cu tact transmis. Alimentarea poate fi realizată cu un alimentator de 5V, prin cupla USB sau de la un semnal serial nefolosit.

Figura 10.8. Adaptorul Bluetooth serial LM058 (sursa www.farnell.ro)

Modulul poate fi programat cu comenzi AT. Lista completă a comenzilor AT este dată în foile de catalog.Câteva comenzi AT sunt:

AT – verificare
AT+ENQ – afişează toate setările, cele de Bluetooth şi de RS232
AT+ACON – validează conectarea automată
AT+CONN =xxxxxxxxxxxx– stabileşte o conexiune cu dispozitivul a cărui adresă este xxxxxxxxxxxx
AT+FIND – caută un dispozitiv Bluetooth timp de un minut
AT+NAME – stabileşte un nume pentru un dispozitiv Bluetooth
AT+PIN – trimite codul PIN
AT+RESET –inițializează dispozitivul
AT+BAUD – stabileşte viteza de comunicație prin RS232

Un microcontroller complex Bluetooth este Atmel AT76C551 bazat pe un nucleu ARM7. Microcontrollerul prototip este echipat cu interfețe USB, UART şi PCMCIA şi se foloseşte la punerea la punct a aplicațiilor. Pentru producția în serie se fabrică microcontrollere cu una dintre interfețe. Microcontrollerul poate fi folosit la realizarea adaptoarelor USB Bluetooth pentru calculatoare desktop, adaptoare PCMCIA pentru notebook-uri, adaptoare USB pentru imprimante, adaptoare pentru camere digitale, telefoane mobile, PDA etc.

10.5. Zigbee

Transmisia ZigBee este o transmisie wireless mai ieftină decât Bluetooth, asigură un consum mai redus de energie şi dimensiuni mici dar asigură şi un debit mai mic de date. Numele se pare că provine de la zborul în zig zag al albinelor care îşi transmit date referitoare la poziția sursei de hrană. Acest tip de transmisie se pretează la aplicații de tip rețea de senzori (rețele *mesh*). Prima apariție a ZigBee a fost în 1998, ca urmare a nevoii de o interfață mai ieftină decât Bluetooth pentru aplicații cu mulți senzori în care rețeaua se autoconfigurează la intrarea sau ieşirea unor senzori din activitate. Ca aplicații se pot menționa sisteme de senzori în domeniul casnic (incendiu, fum etc.), industrial, medical pentru urmărirea datelor provenite de la un pacient, în telecomunicații etc. Banda alocată este 2,4GHz, dar şi alte 2 benzi folosite în diferite țări. Modulația este în cuadratură, QPSK şi asigură 250kbps la o distanță uzual până la 70m. Intervalul de bandă folosit este între 2,405GHz şi 2,480GHz, împărțit în canale de 5MHz. Modulele ZigBee pot lucra în modul punct la punct sau punct la multipunct şi o rețea de astfel de dispozitive necesită un dispozitiv cu funcția de coordonator. Rețeaua *mesh* permite conexiuni radio de date între dispozitive mai îndepărtate decât raza de acțiune radio prin interpunerea unor noduri ZigBee intermediare iar defectarea unui nod poate fi transparentă prin preluarea sarcinilor de alt nod. Zigbee a fost standardizat de IEEE cu numele IEEE 802.15.4.

XBee asigură o rată de transfer de 250kbps la distanţe de maximum 100m în spații închise şi 1,6km în spații fără obstacole, iar datele sunt furnizate printr-o interfață serială care admite şi comenzi AT, viteze posibile fiind între 1200bps şi 1Mbps. Comunicarea radio poate fi criptată (AES) iar corectitudinea transmisiei este asigurată de un mecanism de confirmare (ACK) şi reîncercare. Puterea de emisie este de maximum 50mW (17dBm) la 2,4GHz. Pentru legătura cu senzorii modulul are 10 pini de I/O şi un canal de conversie A/D pe 10 biți. Tensiunea de alimentare poate fi între 2,1V şi 3,3V, curentul maxim (în cazul transmisiei) fiind de 295mA. Există şi variante de module cu consum mic şi rază de transmisie mai mică.

Aceste module pot realiza o rețea *mesh* şi astfel se pot implementa rețele cu proprietăți de descoperire a noilor dispozitive, eliminarea dispozitivelor defecte etc. Astfel modelul XBee-PRO asigură doar o comunicație punct la punct iar XBee Znet poate fi interconectat într-o rețea *mesh*.

Modulele XBee realizează o interfață ZigBee conform standardului IEEE 802.15.4, lucrează la frecvența de 2,4GHz şi sunt destinate pentru funcția de senzor în rețele wireless, având ca şi caracteristici preț mic, putere consumată mică (63mW) şi fiabilitate ridicată, figura 2.3.4. stânga. Distanța maximă de transmisie este 30m în interior şi 90m în exterior dar există module de putere mai mare care asigură transmisia pe distanţe mai mari de până la 1600m, la un debit al informației de 250kbps.

Modulul Xbee conţine un convertor AD integrat şi are un set de 8 linii care pot fi configurate ca linii digitale de I /O sau intrări pentru conversie AD. Circuitul dispune şi de un generator PWM integrat. Semnificaţia pinilor este dată în figura 10.9. dreapta.

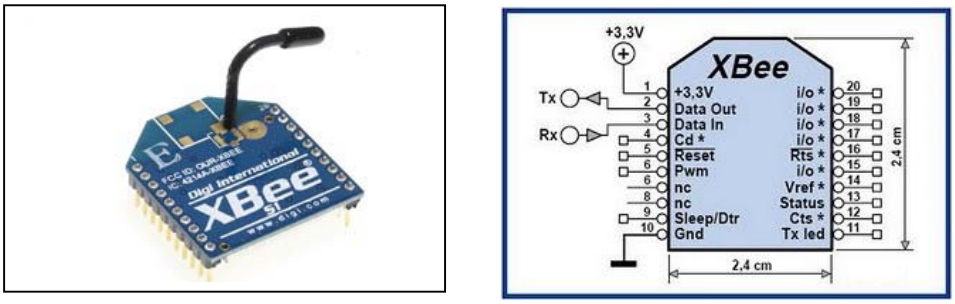


Figura 10.9. Xbee (sursa http://www.digi.com/controls/products/wireless-wired-embedded-solutions/)

Implicit modulul funcţionează în modul transparent în care tot ce se trimite/recepţionează pe interfaţa serială este emis/ recepţionat RF. Pentru a trece în modul de comenzi AT se trimit trei caractere + succesiv. Reluarea modului transparent se face cu comanda ATCN. Un exemplu de programare a adresei modulului este:

Comanda AT	**Răspuns**	**Rezultat**
+++	OK	intrare în mod comenzi AT
ATDL <Enter>	adresa curentă	se afişează valoarea curentă a adresei
ATDL1A0D <Enter>	OK	se trimite adresa dorită
ATWR <Enter>	OK	se scrie informaţia în memorie
ATCN <Enter>	OK	ieşire din modul de comenzi AT

Un alt exemplu este comanda ATD6 P, unde P este un parametru care poate fi 0-5 prin care se stabileşte dacă pinul 5 al circuitului este cu funcţia de RTS, intrare analogică, pin de I/0 etc.

10.6. RFID

Identificare prin frecvenţă radio (*Radio-Frequency Identification* sau RFID) este o metodă de identificare automată care se bazează pe stocarea şi regăsirea datelor la distanţă, folosind dispozitive numite etichete RFID (*tag RFID*) şi transmiţătoare RFID. Tehnologia necesită o cooperare a unui aparat cititor de RFID cu eticheta RFID. O etichetă RFID este un obiect mic sau foarte mic (sub 1 mm x 1 mm) care poate fi aplicat sau încorporat într-un produs, animal, sau chiar persoană, cu scopul de identificare şi

urmărire, folosind undele radio. Unele etichete pot fi citite de la mulți metri depărtare, chiar mult peste 50 m, iar eticheta se poate afla şi în afara razei de vedere a cititorului.

O istorie a RFID scrisă de unul dintre cei care au dezvoltat această tehnologie este dată în lucrarea J. Landt, *The history of RFID*, online la: http://www.transcore.com/pdf/). Stockman a scris în 1948 prima lucrare care prevede posibilitatea RFID *Communication by Means of Reflected Power*. În 1960 R. F. Harrington studiază teoretic RFID dar primul patent a fost acordat lui H.W. Cardullo în 1973 pentru o etichetă RFID cu memorie (http://www.rfidjournal.com/article/view/1338/1/129) . În 1970 guvernul SUA a început cercetările la Los Alamos pentru un sistem de urmărire a materialelor nucleare, avându-l ca membru pe J. Landt. Preluarea ideii de RFID în comerțul retail a constat în eticheta de un bit- articol plătit sau nu, în 1960. După Los Alamos cercetătorii au fondat propria companie pentru dezvoltarea de aplicații comerciale. Primele etichete au fost realizate cu funcționare la 125kHz, dar frecvența a crescut ulterior pentru a extinde raza de acțiune. În 1999 a fost fondat la MIT Auto-ID Center de câteva companii importante pentru a realiza tag-uri ieftine care se pot ataşa pe orice produs. La Auto-ID Center s-au alipit peste 100 de companii până în 2003 şi au dezvoltat EPC (Electronic Product Code) şi o arhitectură de rețea cu acces Internet pentru gestionarea datelor.

Cele mai multe etichete (tag-uri) RFID conțin cel puțin două părți, figura 10.10:

- un circuit integrat pentru stocarea şi prelucrarea de informații, modulare şi demodulare a unui semnal de radio-frecvență (RF), şi alte funcții speciale (*transponder RFID*);
- o antenă pentru recepționarea şi transmiterea de semnale radio.

Figura 10.10. Eticheta RFID

Astăzi tehnologia RFID este deja folosită în domenii foarte numeroase. De exemplu în lanțul de aprovizionare al întreprinderilor, pentru a îmbunătăți eficiența inventarelor, pentru urmărirea produselor în cursul fabricației şi pentru managementul produselor. Alte exemple de utilizare a RFID sunt:

- măsurarea timpului de la cursele atletice;
- controlul paşapoartelor;
- aplicarea taxelor rutiere pe anumite autostrăzi etc.;

- urmărirea produselor (vacile unei cirezi, cărţile unei biblioteci, transcontainerele pe un vapor);
- urmărirea locomotivelor şi vagoanelor la căile ferate;
- autentificarea persoanelor care doresc să intre în zone speciale;
- paza şi inventarierea în muzee.

Din cauza miniaturizării permanente a tag-urilor, ajunsă până acolo încât ele sunt din ce în ce mai greu de văzut şi recunoscut, a apărut şi o problematică gravă, cea a potenţialului pentru spionaj în cele mai diverse domenii.

În figura 10.11 se observă cititorul (stânga) care generează un câmp electromagnetic pentru alimentarea transponderului din eticheta RFID. Aceeaşi bobină este folosită ca antenă pentru transferul de date.

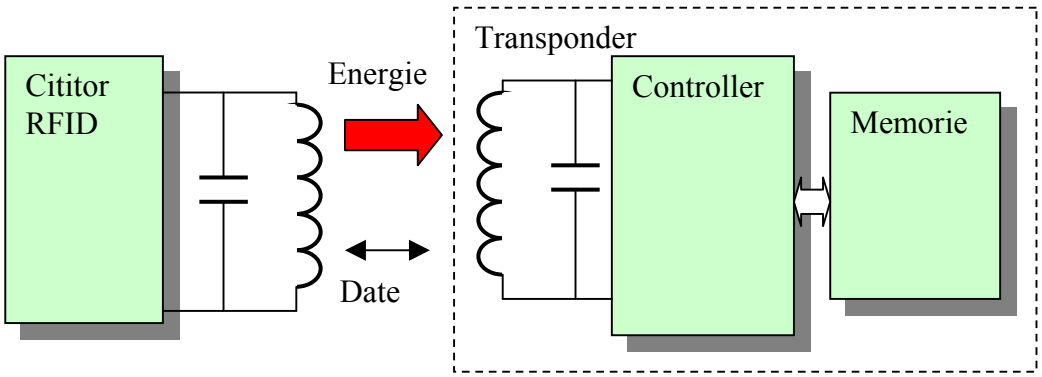


Figura 10.11. Structura sistemului RFID

Etichetele pot fi pasive (ca în figură, cazul cel mai frecvent), ele folosind energia furnizată de cititor, fiind astfel mai mici, mai ieftine şi având o viaţă mai lungă. Pentru extinderea razei de funcţionare etichetele pot fi active, având o sursă de alimentare proprie.

Frecvenţa de lucru poate fi 125kHz care asigură un preţ scăzut dar o rată mică de transfer şi o etichetă de dimensiuni mai mari. O altă frecvenţă este de 13,56MHz care elimină dezavantajele de la 125kHz dar costurile sunt mai mari. Alte frecvenţe utilizate în prezent sunt cea de 869MHz – 950MHz (funcţie de zona geografică) şi 2,4GHz, banda ISM în care se aglomerează aproape toate tipurile de comunicaţii de rază scurtă.

Din punctul de vedere al iniţierii transferului există două situaţii, când eticheta iniţiază transferul când intră în zona de acţiune a cititorului (TTF, Tag Talks First) cu dezavantajul că intrarea simultană a mai multe etichete duce la apariţia unor întârzieri şi a doua situaţie când cititorul interoghează continuu zona de acţiune şi identifică eticheta care intră în zonă (RTF, Reader Talks First).

Cel mai simplu mod de implementare este utilizarea unei interfaţe specializate de citire /scriere a etichetelor, cum este cea de la Netronix, H1M-005. Interfaţa lucrează la

frecvenţa de 125kHz, poate fi alimentată între 4,1V şi 5,5V, asigură un debit de 4kbps la maximum 20cm. Legătura cu un calculator gazdă sau microcontroller se face prin RS232 la viteza de 9600Bd. Conectarea este simplă, ca în figura 10.12:

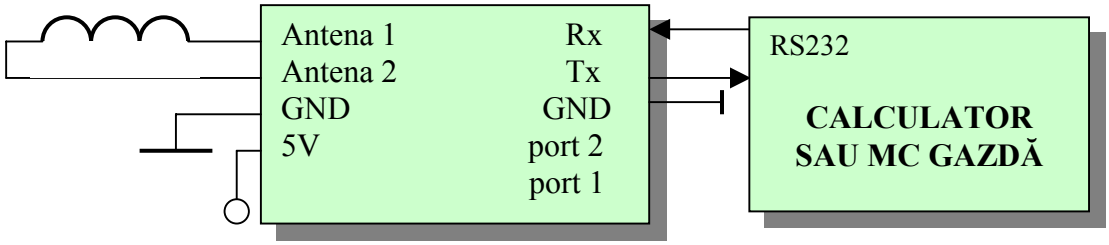


Figura 10.12. Conectarea modulului Netronix H1M-005 (sursa www.netronix.pl)

Formatul comenzii către etichetă şi al răspunsului primit sunt date în Tabelul 1 respectiv Tabelul 2:

Tabelul 1

Adresa modulului	**Lungime cadru**	**Comandă**	**Parametri (1..n)**	**CRC**
1 octet	1 octet	1 octet	n octeţi	2 octeţi

Tabelul 2

Adresa modulului	**Lungime cadru**	**Răspuns**	**Parametri (1..n)**	**Confirmare operaţie**	**CRC**
1 octet	1 octet	1 octet	n octeţi	1 octet	2 octeţi

Fiecare etichetă (de tip HITAG1) are o adresă proprie. Dacă adresa pusă de cititor este 00H nu va răspunde nici un modul, dacă este FFH vor răspunde toate modulele aflate în raza de acţiune. Parametri există sau nu în funcţie de tipul comenzii. Confirmarea operaţiei specifică în răspuns corectitudinea execuţiei.

Eticheta este văzută de cititor ca o memorie organizată în 16 blocuri, fiecare bloc are 4 pagini, fiecare pagină are 4 octeţi, deci un total de 256 octeţi în 64 de pagini. Blocurile 0 şi 1 sunt rezervate pentru configurare. Unele blocuri sunt publice, iar altele sunt protejate la citire.

Pentru exemplificare se descriu câteva comenzi şi răspunsuri care pot fi trimise de calculatorul gazdă spre HIM-005 prin interfaţa serială în formatul din Tabelul 1 respectiv Tabelul 2:

- Scriere în etichetă- codul instrucţiunii este A0H, parametri sunt 4 octeţi de scris şi adresa paginii (sunt 64 de pagini, adresa este între 00H şi 3FH). Codul

răspunsului este A1H, ca parametri este dată adresa etichetei şi confirmarea operaţiei care este FFH dacă s-a scris cu succes;

- Citirea din etichetă- codul instrucţiunii este A2H, parametru este adresa paginii. Codul răspunsului este A3H, ca parametri este dată adresa etichetei, cei 4 octeţi de date citiţi şi confirmarea operaţiei care este FFH dacă s-a citit cu succes;
- Comanda de cuplare / decuplare a câmpului electromagnetic generat de antenă- codul comenzii este 10H /12H, fără parametri, răspunsul are codul 11H /13H şi confirmarea operaţiei FFH;
- Scrierea / citirea unui bit din liniile de I/O locale ale HIM-005- codul comenzii este E0H / E2H, parametru este numărul portului şi bitul de scris, codul răspunsului este E1H / E3H, parametru bitul citit şi confirmarea operaţiei întotdeauna FFH;
- Setarea amplificării receptorului- codul comenzii este F0H, parametru este amplificarea (0,1,2 sau 3), codul răspunsului este F1H şi confirmarea operaţiei întotdeauna FFH.

În figura 10.13 este arătat un model experimental de sistem RFID cu Netronix HIM-005 realizat ca şi proiect de licenţă. În stânga este modulul electronic cu HIM-005 şi un microcontroller ca şi sistem gazdă, cu antena cuplată şi o etichetă HITAG1 în partea de jos a fotografiei. În dreapta este o captură de ecran a softului pus la dispoziţie de Netronix pentru citirea / scrierea etichetei.

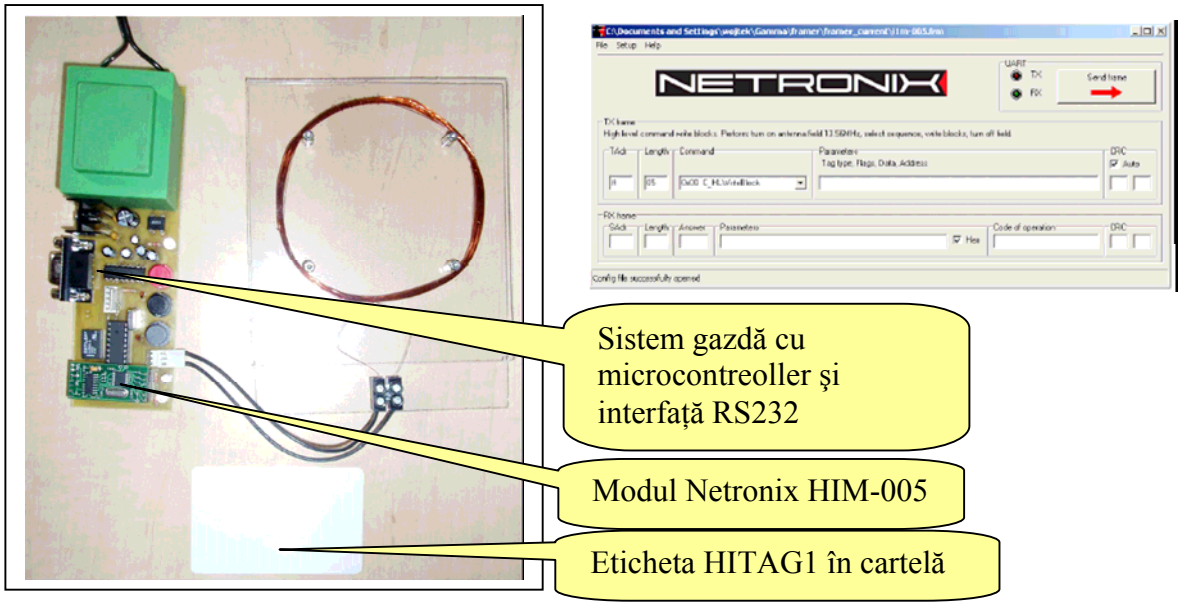


Figura 10.13. Model experimental de sistem RFID cu Netronix HIM-005 (lucrare de diplomă)

10.7.Concluzii

1. Concepția, proiectare şi realizarea practică a unei comunicații wireless este mai simplă decât pare, aceasta datorită circuitelor de interfaţare specializate. Utilizarea circuitelor specializate micşorează timpul de realizare a unei aplicații – *Time to market.*

2. Concepția unui sistem electronic de comunicații începe cu un studiu pe net în ceea ce priveşte existența circuitelor specializate (pe paginile constructorilor de circuite- ATMEL, TI, Microchip etc.) apoi disponibilitatea comercială (pe paginile furnizorilor din România- Vitacom, ECAS, Adelaida, Farnell etc.);

3. Interfaţarea unui circuit specializat cu un microcontroller se reduce de cele mai multe ori la conectarea printr-o interfață standard serială sau paralelă, de aceea este importantă studierea detaliată a acestor interfețe.

Un tabel comparativ al standardelor parcurse în acest capitol cu avantaje şi dezavantaje este dat în tabelul următor:

	ZigBee	**GPRS**	**802.11**	**Bluetooth**	**Protocoale proprietare**
Aplicații	Monitorizare şi control	Rețele internaționale de date şi voce	Internet	Conectivitate între dispozitive	Conectivitate între dispozitive
Durata bateriei	Ani	Săptămâni	Săptămâni	Săptămâni	Luni
Viteza	250kps	2Mbps	54Mbps (300Mbps la n)	720kbps	115kbps
Raza	100m	km	100m	100m	200m
Avantaje	Putere şi cost mic	Rază mare de acțiune	Viteza	Comoditate	Preț şi consum mic, simplitate
Rețea	Posibilă	Acces Internet	Posibilă	Posibilă	Greu de implememtat

Prin analiza acestui tabel se pot deduce variantele optime de transmisie radio pentru fiecare aplicație. După alegerea principiului de transmisie se trece la alegerea variantei constructive. În acest capitol sunt prezentate de regulă câte trei variante constructive. Prima, cea mai simplă, care optimizează timpul de realizare a unei aplicații este folosirea unui modul specializat. A doua, un compromis între timpul necesar dezvoltării şi costului este folosirea unui circuit de interfață conectat la microcontrollerul aplicației. A treia, care asigură costul cel mai redus al aplicației este utilizarea unui microcontroller care conține integrată interfața de comunicații aleasă. Totuşi, în această variantă mai trebuie adăugat de regulă un circuit transceiver (partea de RF).

11. Alte interfețe: IEEE1394, IrDA, SATA

11.1. IEEE 1394

IEEE 1394 este o interfață serială cunoscută sub numele de FireWire (Apple), i.LINK (Sony) şi LYNX (TI). Interfața a fost adoptată de HANA (High Definition Audio-Video Alliance) ca interfață standard disponibilă şi wireless, pe fibră optică sau pe cablu coaxial. Dezvoltarea interfeței a început în 1980 şi a fost încheiată în 1995. IEEE 1394 a fost aplicată şi în aviația militară ca magistrală pentru F-22 Raptor şi F-35. Navetele spațiale NASA au folosit IEEE 1394 pentru anumiți senzori. În industria auto a fost implementată o versiune numită IDB 1394. Cu toate că IEEE 1394 nu are răspândirea pe care o are USB, majoritatea camerelor digitale sunt echipate cu o astfel de interfață.

Ca şi în majoritatea comunicațiilor seriale transferul de date este bazat pe pachete. Canalul comun de date este conceput să poată fi folosit pe rând de fiecare dispozitiv care îl solicită. Există un interval de timp specificat (numit *fairness* interval) în cadrul căruia un dispozitiv are accesul la canalul de date comun. După ce dispozitivul a trimis un pachet de date se aşteaptă scurgerea unui timp de separare (numit *sub-action gap*) după care un alt dispozitiv poate trimite un pachet. Dacă după scurgerea timpului de separare nici un dispozitiv nu are de transmis vreun pachet, urmează o secvență de reset.

Pentru a face posibilă funcționarea dispozitivelor care necesită flux de date în timp real, IEEE-1394 foloseşte un mod special de transfer, modul izocron, ca şi USB. Un dispozitiv ce necesită date izocrone emite la fiecare 125μs un pachet special de temporizare prin care asigură prioritatea transferului. Această schemă de arbitrare garantează un minim de buffer-e pentru date audio sau video (1 byte la dispozitive audio, până la 6 bytes la dispozitive video). Perioada de 125μs coincide cu perioada de eşantionare din sistemul de telefonie digitală, astfel interfața IEEE-1394 poate fi plasată în sistemul ISDN (Integrated Service Digital Network).

IEEE 1394 este asemănătoare cu USB, aşa încât este utilă o comparație:

- La IEEE 1394 nu este nevoie de un calculator gazdă;
- IEEE 1394 asigură o viteză efectivă de transfer mai mare decât USB (dovedit pe sistemul de operare MAC OS X dar cu rezultate contradictorii sub Windows);
- Implementarea IEEE 1394 are costuri mai mari: licența Apple (0.25$/sistem) şi hardware mai scump cu 1-2$;
- Ambele standarde pun la dispoziție prin cablul de transmisie de date o tensiune de alimentare, sunt plug and play şi admit hot swapping. IEEE 1394 admite tronsoane de cablu de maximum 4.5m şi poate alimenta o sarcină cu consum de până la 45W.

- Fiecare dispozitiv IEEE 1394 are un identificator propriu unic, (IEEE EUI-64) care este o adresă asemănătoare cu adresa MAC de 48 de biţi.

În decursul timpului au fost realizate mai multe variante constructive:
1.FireWire 400 (IEEE 1394/1995). Versiunea originală poate transfera date cu viteze de 100, 200 sau 400 Mbps (S100, S200, S400) în mod half duplex. Modul de codificare al datelor este data strobe D/S;
2.FireWire 800 (IEEE 1394b/2002). Versiunea a doua asigură o viteză de 800Mbps în mod full duplex. Conectica este diferită faţă de varianta anterioară. Modul de codificare al datelor este 8B10B.
3.FireWire S800T (IEEE 1394c/2006). Versiunea a treia utilizează cablu Ethernet categoria 5e. Nu există încă implementări în sisteme disponibile pe piaţă datorită confuziei posibile la o placă de bază echipată cu 2 conectori RJ45, unul cu interfaţă Ethernet şi unul IEEE 1394.
4.**FireWire S1600 şi S3200**Se lucrează la versiunile de 1.6Gbps şi 3.2Gbps, care vor fi concurenţi pentru USB 3.0. Conectorii sunt cei de la versiunea FireWire 800.

Se poate implementa o reţea de calculatoare prin legături IEEE 1394 în mod IPv4 sau IPv6. Sistemele de operare care include suport pentru acest tip de reţea sunt MAC OS X, Windows ME, 2000, XP şi Server 2003. Windows Vista şi Server 2008 nu mai conţin acest suport.

În figura 11.1 este dat un tabel cu conexiunile la conectorii IEEE 1394 cu 4, 6 şi 9 pini şi structura unui cablu IEEE 1394.

4 pini	6 pini	9 pini	Funcţie	Descriere
	1	8	Vcc	30V nestabilizat
	2	6	Masă	Masă pentru tensiune
1	3	1	TPB-	semnal diferenţial B
2	4	2	TPB+	semnal diferenţial B
3	5	3	TPA-	semnal diferenţial A
4	6	4	TPA+	semnal diferenţial A
		5	ecran A	
		9	ecran B	

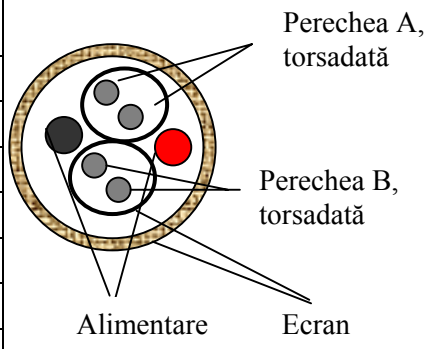


Figura 11.1. Tabel de conexiuni şi structura cablului IEEE1394

Codificarea datelor D/S este o codificare NRZ cu transmiterea tactului şi necesită 2 linii de semnal, una de date şi una de strob. Un SAU Exclusiv între cele 2 semnale reconstituie

tactul, figura 11.2. Petru transmisia datelor este nevoie de ambele perechi FireWire, deci este posibil doar un transfer half duplex. Codificarea este aplicată la FireWire 400.

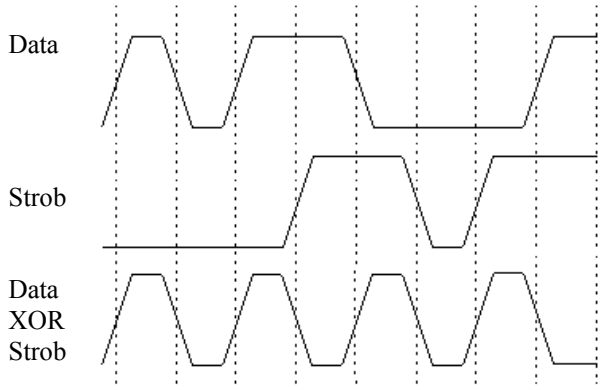


Figura 11.2. Codificarea Data /Strobe

Codificarea 8B10B a fost imaginată de Al. Widmer şi P. Franaszek de la IBM în 1983 şi IBM a obținut un patent. Răspândirea codificării a luat avânt după expirarea patentului. Aplicațiile dovedesc eficiența codificării: PCI Express, SATA, SAS, Fibre Channel, IEEE 1394b, Gigabit Ethernet (mai puțin la 1000BaseT), DVI, HDMI, USB 3.0 şi seamănă cu codificarea folosită la CD (Eight to Fourteen Modulation). În principiu codificarea asigură o componentă DC mică pentru ca şirul de date să poată trece prin transformatorul de separare Ethernet, adică numărul de 0-uri este aproape egal cu numărul de 1-uri. Într-un şir de 20 biți diferența între numărul de 0 şi de 1 poate fi maxim 2. Codul este autosincronizabil şi se admit maxim 5 valori de 0 sau de 1 succesive. Codificarea atribuie la 8b o entitate de 10b numită simbol sau caracter. La 5b mai puțin semnificativi se atribuie 6b (porțiunea 5b/6b) iar la 3b mai semnificativi se atribuie 4b (porțiunea 3b/4b). Se definesc 12 simboluri speciale de control care marchează începutul cadrului, sfârşitul cadrului, skip, etc. Datorită codării cuvintelor de 8b cu simboluri de 10b anumite valori din cele 1024 pot fi excluse pentru a realiza condiția de a nu exista 5 valori de 0 sau de 1 consecutive. Pe linie se transmite întâi porțiunea 5b/6b apoi 3b/4b. Datele pot fi notate ca D.x.y unde x este porțiunea 5b/6b şi poate fi 0-31 iar y este porțiunea 3b/4b şi poate fi 0-7 ca valori necodate. Se defineşte RD (Running Disparity ca diferența între numărul de biți de 1 şi numărul de biți de 0. Se urmăreşte obținerea RD cât mai mic. În acest scop grupurile 5b/6b şi 3b/4b se stabilesc în funcție de RD anterior ca valori negate sau nenegate. De exemplu D.00 se codifică ca 100111 (RD inițial este –1 şi rezultă RD=+1) sau 011000 (RD inițial este +1 şi rezultă RD=–1) La fel, în funcție de RD inițial se codifică şi grupul 3b/4b D.x.0 se codifică ca 1011 (RD inițial este –1 şi rezultă RD=+1) sau 0100 RD inițial este +1 şi rezultă RD=-1). Astfel în ipoteza RD inițial –1, D.00.0 se codifică ca 1001110100 şi rezultă RD=0

Topologia unei arhitecturi IEEE 1394 este de tip stea multiplă (arbore) cu posibilitatea de înlănțuire (daisy-chain). În figura 11.3 sunt prezentate două spații de lucru unite cu un bridge. Cele 2 spații sunt izolate din punctul de vedere al traficului de date. Spațiul 1 de lucru ocupă mare parte a benzii din cauza traficului video, dar în spațiul de lucru 2 calculatorul are întregul control al traficului. Este posibil ca şi calculatorul 2 să solicite date video, chiar dacă calculatorul 1 este oprit. Este figurat un repetor care măreşte distanța de conectare şi un splitter care adaugă 2 porturi unui port IEEE 1394.

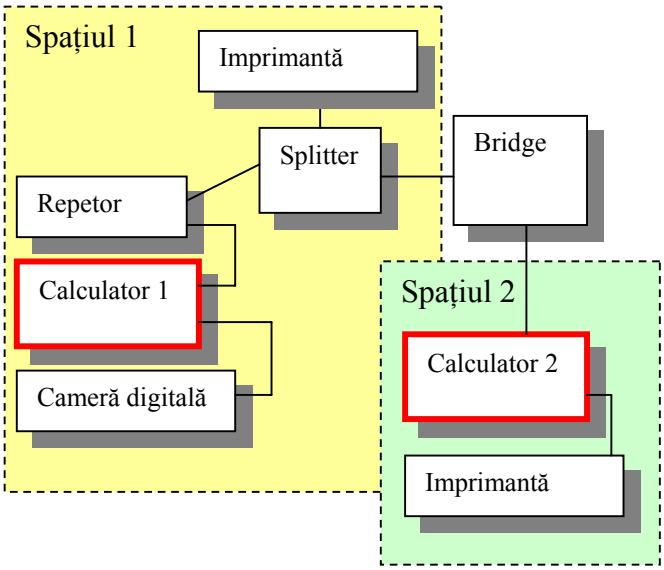


Figura 11.3. Topologia unei arhitecturi IEEE 1394

Pentru a transmite date în mod asincron dispozitivul IEEE 1394 compune un cadru care conține adresele sursei şi destinației, apoi date şi CRC. Când receptorul acceptă datele un cadru de confirmare este trimis la transmițător. Transmițătorul are posibilitatea să trimită încă 63 de cadre continuu pentru a mări viteza de transfer. Dacă cadrul de confirmare returnează o eroare se aplică o metodă de reacție la eroare.

În mod izocron emițătorul solicită un canal izocron iar dacă receptorul îl acceptă i se asigură un interval de timp de transfer pentru a asigura banda necesară transferului. Se pot defini până la 64 de canale izocrone. În exemplul din figura 11.4. în pachetul de date de 125μs sunt definite 2 intervale de timp pentru 2 transferuri izocrone. Timpul rămas liber se poate folosi la transferuri asincrone.

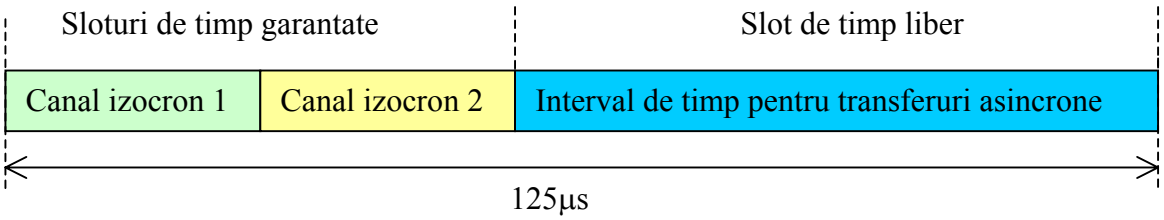


Figura 11.4. Transferuri izocrone pentru a asigura un flux de date în timp real

Circuite IEEE 1394

Nivelele ISO OSI (Open Systems Interconnection) în cazul IEEE 1394 sunt date simplificat în figura 11.5.

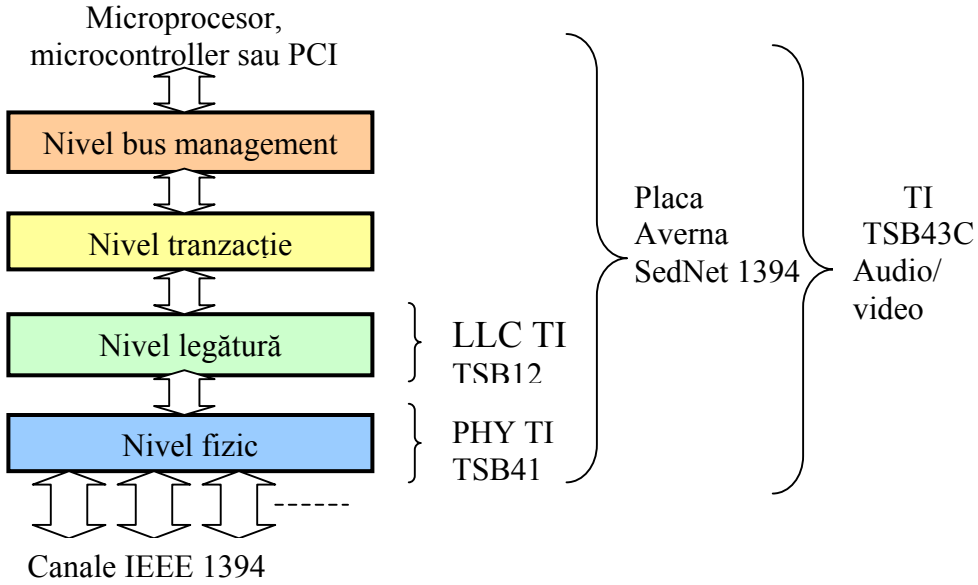


Figura 11.5. structura stivei OSI la IEEE 1394

Nivelul bus-management defineşte funcţiile de bazã de control precum şi registrele de control şi de stare necesare dispozitivelor conectate pentru a-şi face porturile operaţionale. Acest nivel se ocupã şi de asigurarea canalelor, de arbitrare, mastering şi de erori.

Nivelul de tranzacţie mediazã operaţiile de scriere şi citire. Standardul permite la acest nivel operaţii cu cuvinte de lungime variabilã.

Nivelul de legãturã realizeazã controlul logic în legãtura IEEE 1394. Acest nivel realizeazã formarea cadrele la transmisie şi extragerea informaţiei din cadrele recepţionate.

Nivelul fizic presupune atât protocolul transferului cât şi mediul efectiv de transfer. Partea de protocol controlează accesul la legătură, iar partea de mediu este constituită din cabluri şi conectori. La acest nivel se realizează codarea şi decodarea datelor, se asigură nivelele de tensiune necesare şi se face arbitrarea magistralei.

Circuitele IEEE 1394 au un grad de complexitate mai mare decât cele USB, unul dintre motive fiind acela că pot lucra şi independent de calculatorul gazdă. Multe circuite sunt realizate cu magistrală PCI pentru a fi utilizate în calculatoare PC. În figura 11.5. sunt date câteva exemple de circuite şi plăci şi nivelele OSI pe care le acoperă. A fost figurat un circuit care acoperă nivelul fizic (PHY Pysical Layer Controller) şi unul care acoperă nivelul de legătură (LLC Link Layer Controller). Unele circuite, aşa cum este TSB43C de la Texas Instruments acoperă mai multe nivele şi încorporează blocuri de prelucrare audio video, un alt motiv pentru complexitatea mai mare a circuitelor.Placa SedNet de la Averna este un sistem de dezvoltare IEEE 1394 cu microcontroller Motorola şi arată ca în figura 11.6:

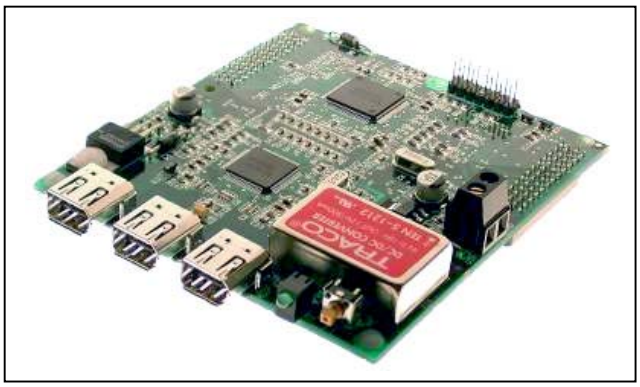


Figura 11.6. placa IEEE 1394 SedNet, sursa
ftp://ftp.adi.com/pub/adiuk/DDS%20V2.0/Averna_1394-OHCI_Boards.pdf

Acest sistem de dezvoltare este o soluție hardware şi software completă pentru gestionarea unei comunicații IEEE 1394 între aplicația unui client care rulează pe un microcontroller care se conectează cu această placă prin intermediul unor linii de I/O sau o aplicație client care rulează pe microcontrollerul plăcii SedNet. Schema bloc a plăcii SedNet este dată în figura 11.7:

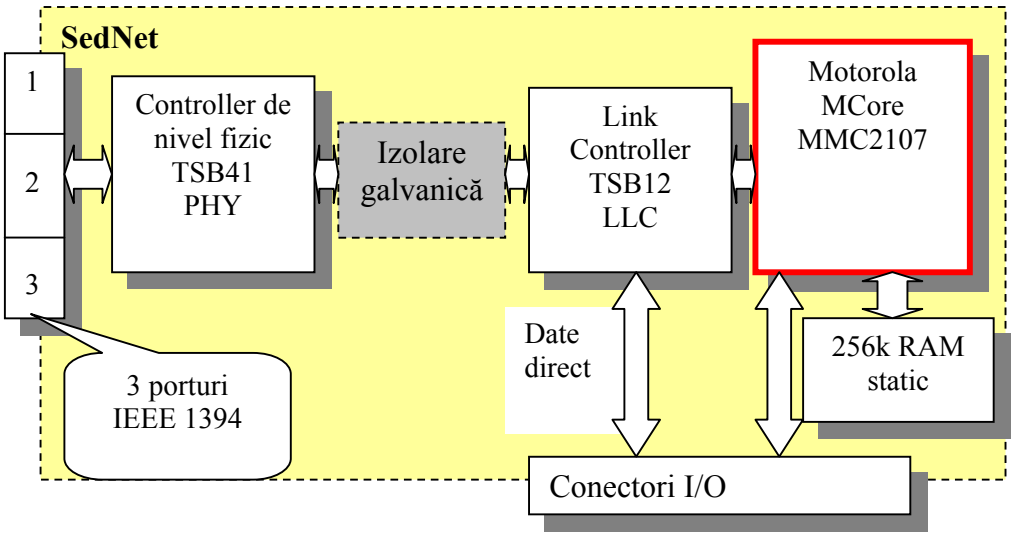


Figura 11.7. schema bloc a plăcii SedNet

Partea software se numeşte Micro-Stack, rulează pe microcontrollerul plăcii şi realizează nivelele bus-management şi tranzacţie din comunicaţia IEEE 1394. Există şi varianta de a cumpăra sursa programului Micro-Stack şi de a o porta pe un alt microcontroller pentru a dezvolta o soluţie hardware proprie. Alimentarea plăcii este realizată cu alimentator propriu sau prin cablul IEEE 1394, dar în al doilea caz acest caz nu este posibilă izolarea galvanică între controllerul de nivel fizic şi restul plăcii.Microcontrollerul MMC2107 are încorporat 128k FLASH iar pentru testare şi debugging portul JTAG este scos la pinii de I/O. Accesul este posibil de la distanţă prin IEEE 1394 la toate resursele plăcii.

Se pot conecta maximum 62 dispozitive IEEE 1394, cu viteze de transfer posibile 100, 200 şi 400Mbps. Placa suportă atât transferuri izocrone cât şi asincrone, datele putând fi preluate direct, fără intermediul microcontrollerului MMC2107 de la controllerul de nivel legătură. Legătura directă este recomandată pentru transferul datelor cu volum mare, cum ar fi cele de la dispozitivele video. Semnalele de legătură cu microcontrollerul sunt cele obişnuite - un port RS232, SPI, JTAG, semnale de întrerupere, reset, tact, linii de I/O etc. Placa a fost special concepută pentru aplicaţii înglobate care nu conţin PC pentru că nu are interfaţă PCI.

11.2.Transferul de date în infraroşu IrDA

În 28 iunie 1993, un grup de 120 de reprezentanţi din 50 de companii de calculatoare au creat o asociaţie numită Infrared Developers Association (IrDA) cu scopul de a standardiza comunicaţiile în infraroşu. Tehnologia era deja pusă la punct în telecomenzile TV şi la transferul de date între Notebook-uri, dar se simţea nevoia unui standard. Primul

standard, bazat pe portul serial RS232 a fost aprobat în 1994. Acest standard foloseşte specificaţiile portului serial, aceeaşi structurã de date dar din pãcate şi limitele vitezei. În 1995 a fost aprobat un nou standard de mare vitezã care împinge limita de vitezã la 1Mbps.

În cadrul comunicaţiilor necablate (wireless), standardul IrDA face parte din categoria transmisiei infraroşu directe, o comunicaţie punct la punct. Între echipamente trebuie sã existe vizibilitate directã. În afara acestui tip de comunicaţie mai existã comunicaţia infraroşu difuzã, o comunicaţie ce permite legãturi multiple şi care nu necesitã vizibilitate directã între echipamente, dar care necesitã materiale speciale de construcţie a clãdirilor.

Avantajele comunicaţiei infraroşu sunt evidente: uşurarea portabilitãţii aparatelor, eliminarea cuplajelor cablate (nu se mai întrerup firele şi nu se mai întâmplã ca un conector sã nu facã contact), eliminarea perturbaţiilor electromagnetice radiate, deci implicit eliminarea interferenţelor electromagnetice. Aceste avantaje sunt majore în zonele de lucru cu regim special, cum ar fi centralele nucleare, laboratoarele de cercetare şi mãsurãri de precizie, acceleratoarele de particule etc. Aceste avantaje au dus la rãspândirea echipamentelor cu interfaţã IrDA. Existã Notebook-uri care au o astfel de interfaţã, imprimante pentru Notebook-uri, dar şi mouse şi tastaturi infraroşu pentru calculatoarele desktop uzuale, sau unele telefoane mobile. Faptul cã perifericele în infraroşu se fabricã în multe exemplare a dus firesc şi la scãderea preţurilor, care le face accesibile.

Codificarea datelor

La viteze între 2.4 kbps şi 1.152 Mbps datele se codificã RZI (Return to Zero Invert). Prin aceastã codificare unui 0 logic îi corespunde un impuls, iar la un 1 logic nu apare nici un impuls, figura 11.8, sus. Impulsul are o duratã fixã, mai micã decât durata celulei bit.

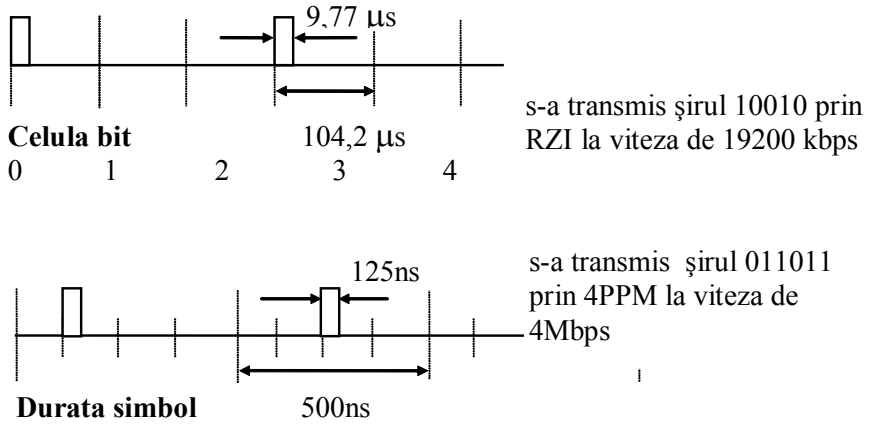


Figura 11.8. Codificarea datelor la IrDA

Un impuls durează 3/16 din lungimea unei celule bit. Un impuls cu 10% mai lung este acceptabil. De exemplu, celula bit la viteza de 9600 bps durează 104.2μs , deci durata impulsului este 18.53 μs. La viteze mai mari durata impulsului este 1/4 din durata celulei bit. Durata impulsului în celula bit este astfel aleasă ca receptorul optic să poată distinge impulsurile.

La viteza de 4 Mbps codificarea se face prin modularea impulsurilor în poziție. IrDA implică 4 poziții pentru impuls, de aceea codificarea se numeşte 4PPM (4 Pulse Position Modulation). Această codificare foloseşte poziția unui impuls în celula bit pentru a indica o valoare logică. Lungimea celulei bit se numeşte durata unui simbol (symbol duration) şi este împărțită în 4 segmente egale numite *chips*. Un impuls poate apare în unul şi numai în unul din aceste segmente. Fiecare impuls în una din 4 poziții poate codifica 2 valori binare. Pentru transmisia unui octet sunt suficiente 4 celule bit. Datele sunt transmise în format pe 8 biți de date, un bit de start, un bit de stop, fără bit de paritate. Se transmit astfel 10 biți/caracter. Cel mai puțin semnificativ bit se transmite primul. Durata impulsului este de 125ns pentru un impuls singular. Pentru mărirea siguranței transferului se pot folosi impulsuri duble, a căror durată totală este de 250ns.Codificarea 4PPM este autosincronizabilă, deoarece în fiecare celulă bit există un impuls. Codificarea RZI însă nu este autosincronizabilă, deoarece poate apare un şir lung de valori logice 1 care înseamnă lipsa impulsurilor transmise o perioadă lungă de timp, perioadă în care receptorul poate pierde sincronizarea.

Sistemele IrDA de viteze mici lucrează în mod asincron şi la aceste sisteme transmisia se face cu tact standard (cu acest cod neautosincronizabil), pentru că nu pot apare erori prea mari la transmisia a doar 10 biți. La viteze medii însă, în cazul transmisiei sincrone, este nevoie de autosincronizare.

Există multe firme care produc cicuite pentru transferul de date IrDA, aşa cum sunt: Texas Instruments, MAXIM, Sharp, Novalog, Agilent Technologies, California Eastern Laboratories, EXAR, Linear Technology etc. Ca variante constructive se poate opta pentru un transceiver IrDA care să se conecteze la un circuit UART existent, se poate alege un circuit UART cu port IrDA sau se poate realiza o interfață IrDA cu microcontroller.

Circuite IrDA

Un transceiver TI de tip TIR1000 poate lucra atât IrDA cât şi în standardul de transfer infraroşu al Hewlett Packard HPSIR. Viteza poate fi între 1200 şi 115200 bps, iar tensiunea de alimentare între 2,7 şi 5V. Este disponibil în capsulă PSOP (Plastic Small

Outline Package) cu 8 terminale. Circuitul codează şi decodează semnalele IrDA, aşa încât el se poate conecta la un UART, figura 11.9.

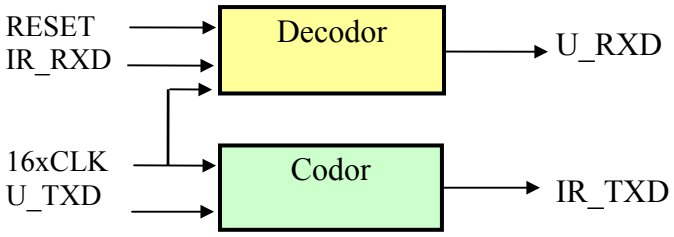


Figura 11.9. Transceiver TIR1000

1. 16xCLK este un semnal de ceas care trebuie să fie de 16 ori mai mare decât rata de transmisie. Tactul maxim este de 16x115200=1.843MHz.
2. IR_RXD intrarea pentru date recepționate IrDA, cu factorul de umplere de 3/16, de la un dispozitiv optoelectronic.
3. IR_TXD ieşire pentru datele emise IrDA către o diodă în infraroşu.
4. RESET inițializare circuit, legat la linia de RESET a circuitului UART.
5. U_RXD date decodate, spre intrarea de date a UART.
6. U_TXD date de transmis de la ieşirea de date a UART.

Un modul care conține un emițător şi un receptor IR (ZHX1010) este prezentat în figura 11.10 stânga iar în mijloc este schema de conectare care arată simplitatea soluției. Pentru conectarea la un PC se poate folosi un adaptor USB IrDA (dreapta).

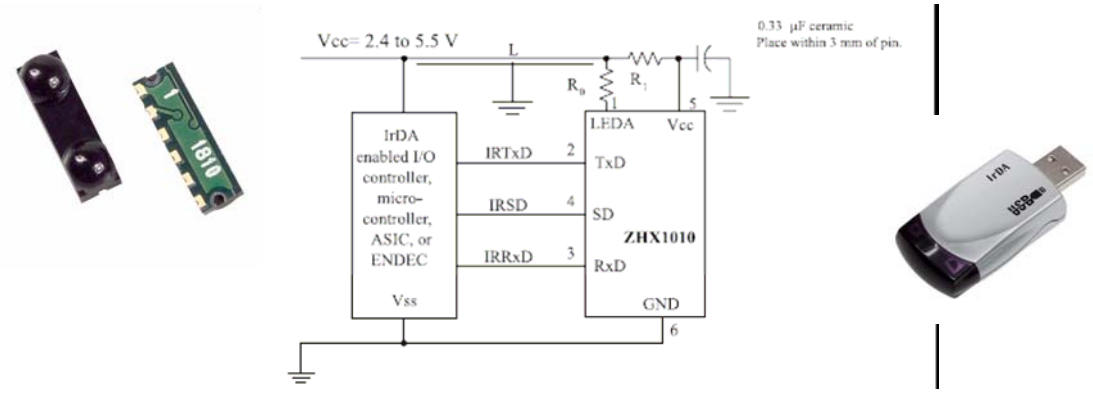


Figura 11.10. Modul IR (sursa http://www.alldatasheet.com/datasheet-pdf/pdf/113918/ZILOG/ZHX1010.html) şi adaptor USB IrDA (sursa http://www.i-tec-europe.eu/?t=3&v=20)

11.3.Interfaţa SATA

Interfaţa serială SATA este cea mai răspândită interfaţă în prezent pentru cuplarea hard discurilor. Interfaţa este formată din 2 perechi de fire cu transmisie LVDS (Low Voltage Differential Signaling, 250mV), o pereche pentru date emise, una pentru date recepţionate, transmisia fiind diferenţială. Datele sunt codificate 8B/10B ca şi Ethernet Gigabit, PCIe sau Fibre Channel. Legătura SATA este o legătură punct la punct, fiecare drive este conectat printr-un cablu serial la placa de bază. SATA a înlocuit interfaţa paralelă ATA (numită şi PATA) (Advanced Technology Attachment).

Prima generaţie SATA (SATA 150) comunică cu viteza de 1,5Gbps, ceea ce înseamnă circa 1,2Gbps informaţie utilă, adică 150Mocteţi/s, ceva mai mult decât ATA133. SATA 300 (SATA 2) asigură o viteză de 3Gbps şi are compatibilitate în jos la SATA 150 în urma unei negocieri la iniţializare. Acest protocol nu este implementat în toate dispozitivele gazdă, aşa încât pe unele hard discuri există un jumper pentru selecţia SATA 150 sau SATA 300.

Avantajele SATA sunt:

- •Viteza de transfer ridicată
- •Posibilitatea implementării Hot Plug In
- •Posibilitatea unor porturi externe SATA (eSATA)
- •Cablul de date este mai mic deci mai ieftin şi asigură o circulaţie mai bună a aerului în carcasă.

Cablul de alimentare SATA trasmite tensiuni de 3,3V, 5V şi 12V. Adaptorul de la sursa ATX nu are 3,3V ceea ce împiedică hard discurile astfel alimentate să aibă capacitatea de Hot Plug In.

Protocolul în interfața SATA

Interfața SATA este o interfață serială sincronă cu refacerea tactului din datele citite, codificarea fiind cu adăugare de biți, transmisia fiind realizată cu cadre (blocuri) de date. Sunt definite secvențe de date numite primitive SATA care sunt de fapt cadre de date utilizate pentru comenzi / stări.

- SYNC – SincronizareX_RDY – Interfața este gata să transmită un cadru
- R_RDY – Interfața este gata să recepționeze un cadru
- R_IP – Interfața este în curs de recepție a unui cadru
- WTRM – Interfața a terminat de trimis un cadru şi aşteaptă terminarea conexiunii
- R_OK – Interfața a terminat de recepționat un cadru şi CRC este corect
- R_ERR – Interfața a terminat de recepționat un cadru şi CRC nu este corect
- SOF – Început de cadru
- EOF – Terminarea unui cadru

Înainte de transferul de date interfața SATA- negociază viteze de transfer. Pentru aceasta se folosesc cadrele sau primitivele de negociere, figura 11:

- COMINIT – Folosit de dispozitivul SATA pentru a solicita o inițializare
- COMRESET – Folosit de gazdă pentru a forța un Reset
- COMWAKE – Folosit de dispozitiv sau gazdă pentru a scoate interfața din modul adormit.

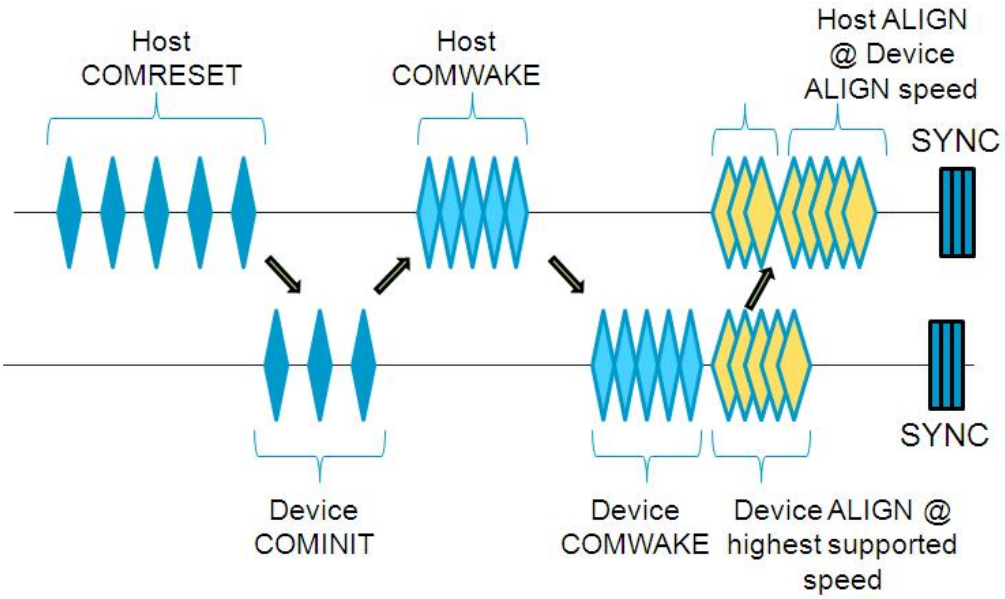


Figura 11.11. Negocierea vitezei (sursa http://www.serialtek.com/sata_protocol_overview.asp)

Gazda trimite un COMRESET care resetează drive-ul. Drive-ul solicită cu COMINIT iniţializarea comunicaţiei. Gazda trimite COMWAKE care scoate drive-ul din modul adormit. Drive-ul trimite ca răspuns COMWAKE şi ALIGN la frecvenţa cea mai mare de comunicaţie.

Secvenţele SYNC se trimit pentru a permite sincronizarea buclei PLL din receptor atât de la gazdă la drive cât şi invers.

După negocierea vitezei se transmit primitive de sincronizare pentru a permite sincronizarea buclei PLL din receptor atât de la gazdă la drive cât şi invers, figura 11.12.

Cu primitiva X_RDY gazda comunică că este gata să transmită date.

Cu primitiva R_RDY drive-ul comunică că este gata să recepționeze date.

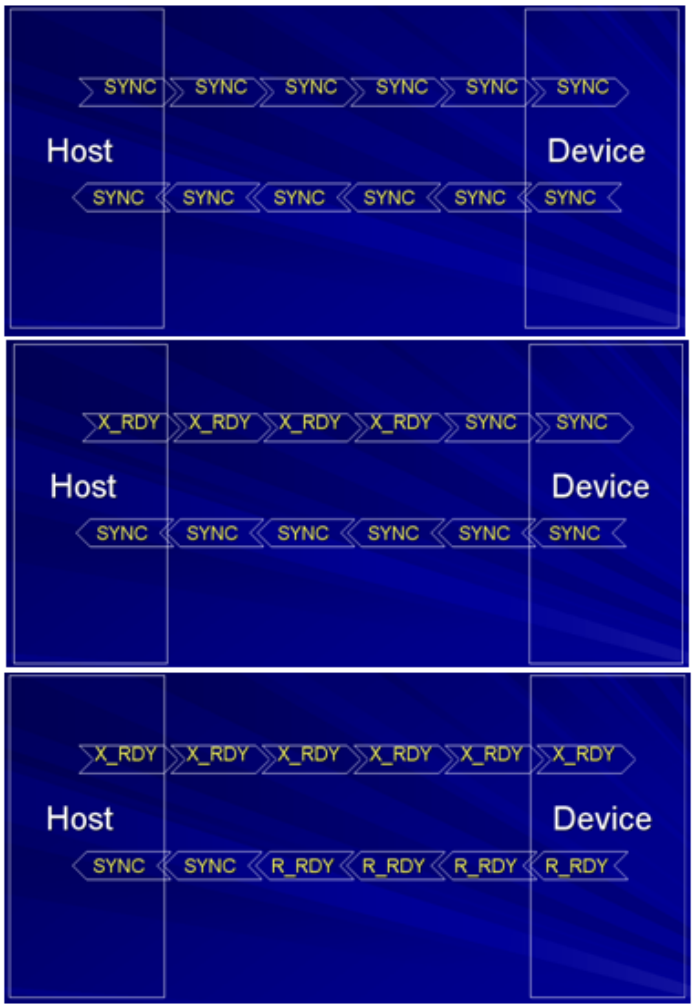
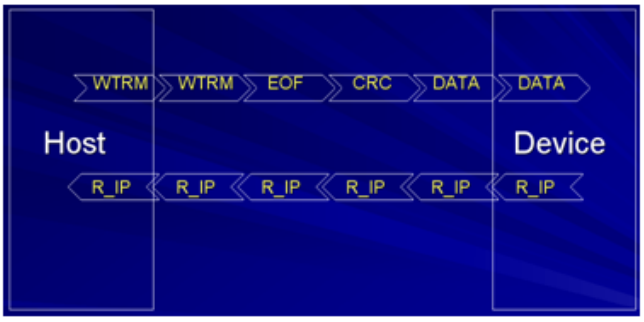


Figura 11.12. Protocolul de transfer de date SATA (1) (sursa www.eng.auburn.edu)

Gazda începe să transmită date, precedate de primitiva SOF, figura 13.

Cu primitiva R_IP drive-ul anunţă că primeşte date

După terminarea cadrului de date gazda inserează CRC şi primitiva EOF. Transmite apoi primitiva WTRM aşteptând închiderea conexiunii.



Drive-ul anunţă recepţia completă a datelor şi cu primitiva R_OK comunică un CRC corect.

După terminarea transmisiei de date atât gazda cât şi drive-ul transmit primitiva SYNC.

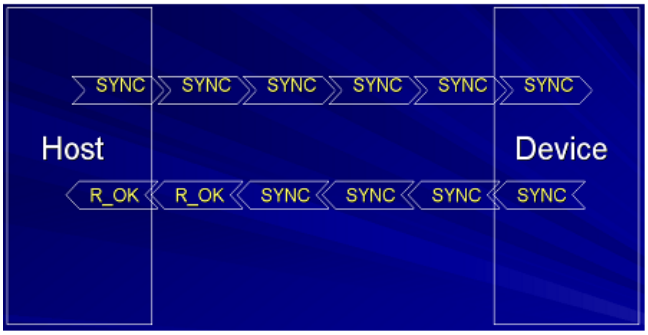


Figura 11.12. Protocolul de transfer de date SATA (2) (sursa www.eng.auburn.edu)

12. Paralelă între stocarea datelor pe suporturi magnetice şi optice şi transmisia serială

12.1.Introducere

Codarea datelor în vederea stocării lor pe suporturi magnetice şi optice are multe puncte comune cu transmisia serială. Acest lucru se datorează faptului că datele memorate pe suport magnetic sau optic sunt aranjate bit după bit pe o pistă a platanului hard discului sau pe pista spirală a unui DVD. Scopul urmărit în dezvoltarea acestor suporturi de informație este mărirea capacității de stocare, dată de micşorarea zonei pe suport ocupată de un bit. Acest scop a generat cercetări ți îmbunătățiri continue în creşterea eficienței codării. Tendința este clară la unitățile de hard disc (HDD) care, datorită faptului că au apărut înaintea unităților optice au trecut prin toate fazele de dezvoltare a codării. La primele hard discuri cele mai simple metode de codare asigurau 20Mocteți la o unitate de 5 inch, în timp ce acum se ating uzual capacități de 1T la unități de 2,5 inch.

Înainte de apariția hard discurilor primele înregistrări magnetice pe bandă au fost realizate prin înregistrarea a două piste alăturate, una cu date şi alta cu tact. Acest mod de înregistrare este echivalent cu transmisia serială cu transmiterea tactului. Ulterior, toate codările au urmărit să asigura refacerea tactului din datele transmise, deci au fost folosite exclusiv codări autosincronizabile.

12.2. Codarea pe suporturile magnetice

Unitatea de hard disc a fost realizată pentru prima dată în 1956 de IBM. Unitatea IBM Ramac 350 avea dimensiunea a 2 combine frigorifice.În 1973 IBM introduce hard discul Winchester la care discurile puteau fi scoase din unitate şi înlocuite cu alt set.În 1980 apar hard discurile de 5,25” iar în 1983 hard discul de 10M intră în componența calculatorului IBM PC XT. Un hard disc Seagate de 5,25” de 20M arată ca în fotografia din figura 12.1.

Figura 12.1. Hard disc Seagate de 5,25” de 20M (sursa: colecția personală P. Ogruțan)

Componentele importante din structura unui hard disc sunt reprezentate în figura 12.2.

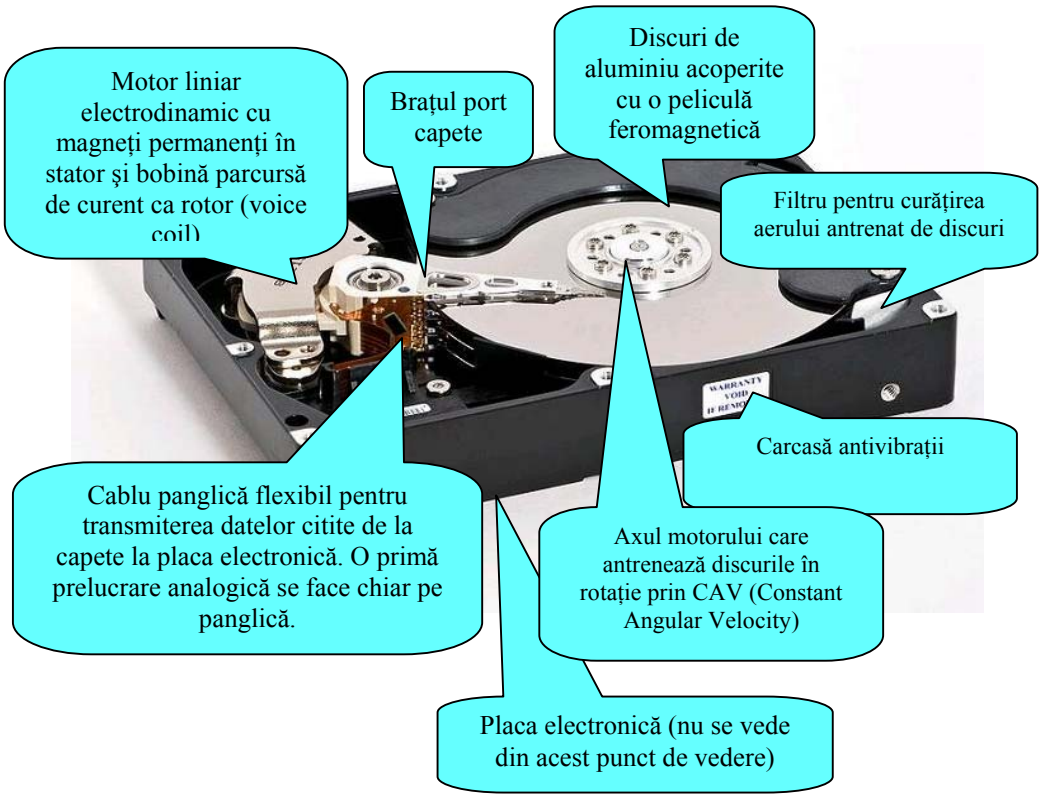


Figura 12.2. Componentele importante din structura unui hard disc

Câteva dintre caracteristicile cele mai importante ale unităților de hard disc sunt:

1. **Timpul de acces** este timpul care trece de la o comandă de acces a sistemului electronic până datele solicitate sunt accesibile. Timpul de acces apare ca urmare a naturii mecanice a brațului care poartă capetele şi a sistemului de rotație a discurilor.
2. **Timpul de poziționare** este timpul necesar poziționării capetelor pe cilindrul dorit. La primele hard discuri la care poziționarea se făcea cu motoare pas cu pas timpul era de ordinul 500ms. Timpul mediu de poziționare la acționările actuale cu motor de curent continuu electrodinamic este de 3-20ms.
3. **Timpul (viteza) de transfer** poate fi dată pentru viteza datelor seriale preluate de pe disc odată ce capetele au fost poziționate şi evident depinde de viteza de rotație. La un HDD de 7200rpm viteza tipică este 1Gbps. Viteza de transfer poate fi dată şi pentru transferul datelor din buffer-ul HDD în calculatorul gazdă,

interfața SATA permite 3Gbps. Acest timp depinde de viteza de rotație a discurilor **dar depinde în cea mai mare măsură de metoda de codare utilizată**.

4. **Timpul de latență** este timpul necesar ca sistemul de rotație să aducă sectorul solicitat în dreptul capetelor. Înregistrarea unui bit de informație pe suport magnetic se face prin înregistrarea a două zone magnetizate cu sensuri contrare alăturate. La trecerea suportului magnetic prin fața capului de citire, tranziția dintre zonele magnetizate va genera o tensiune în bobina capului magnetic, figura 12.3.

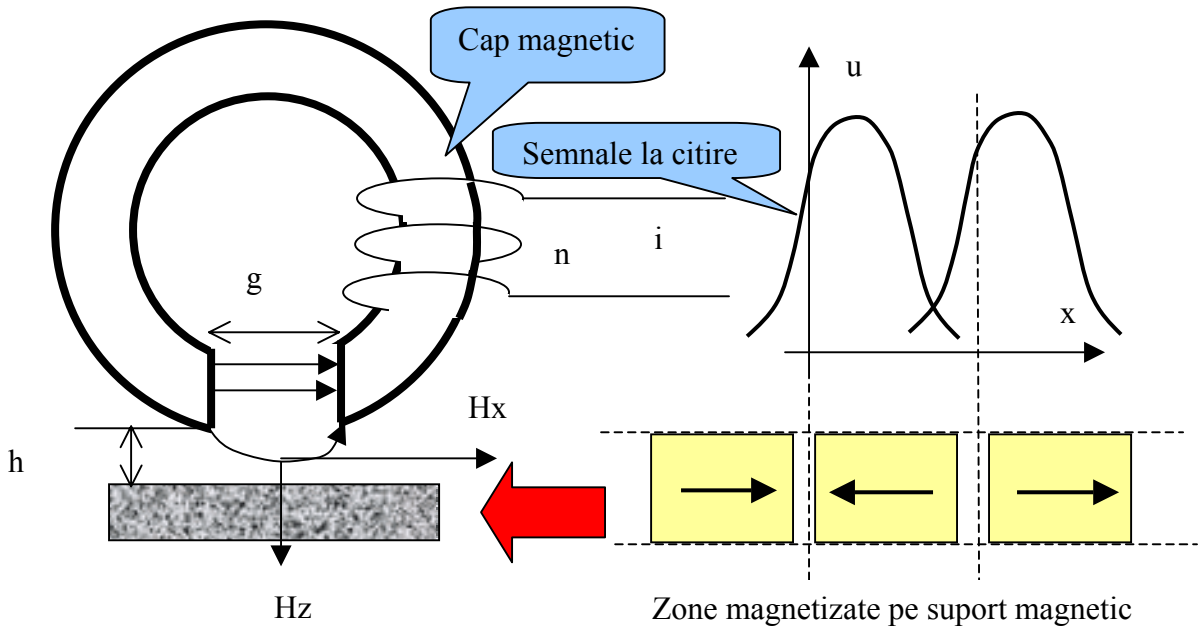


Figura 12.3. Principiul scrierii şi citirii magnetice

Capul de scriere citire este un inel cu un întrefier şi o înfăşurare cu *n* spire străbătută de curentul *i* care crează un flux magnetic în vecinătatea spațiului interpolar şi care determină o magnetizare a stratului magnetic al mediului de stocare. Scrierea este realizată prin fluxul de dispersie şi nu de fluxul prin întrefier. Cu cât întrefierul g este mai mic cu atât densitatea de scriere poate fi mai mare. Componenta Hx a câmpului magnetic este responsabilă de scrierea zonei magnetizatela primele generații de hard discuri, iar componenta Hz este utilizată la noile generații la care zonele magnetizate sunt verticale.

Codarea datelor pentru înregistrarea zonelor magnetizate s-a dezvoltat în timp şi a avut următoarea evoluţie:

1.Codare FM (Modulaţie în Frecvenţă), fiecare bit este precedat de un impuls de tact. În exemplul din figura 12.4. (stânga) se obţine un număr de 3 tranziţii pentru 2 biţi, deci 0,66 biţi pe tranziţie.

2.Codare MFM (Modified FM). Impulsurile de tact se elimină dacă în celula precedentă sau în cea curentă există un bit de date 1. În exemplul din figura 12.4. (dreapta) sunt 4 biţi şi 3 tranziţii, deci 1,33 biţi pe tranziţie.

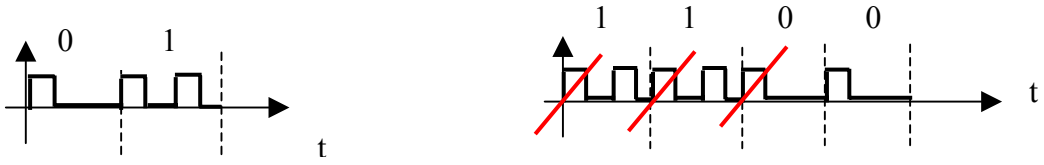


Figura 12.4. Codare FM (stânga) şi MFM (dreapta)

3.Codarea RLL (Run-length limited). Codarea iniţială RLL realizată de IBM este o codare de grup care admite maxim două valori de zero consecutive conform tabelului alăturat. Se codifică astfel 4 biţi cu 5 biţi, aproximativ 3,5 tranziţii în fiecare grupă de 5 biţi, deci 1,43 biţi pe tranziţie, figura 12.5. stânga.

4.Codarea RLL 1,7 codează 2 biţi în grupe de 3 biţi. Anumite combinaţii ale celor 2 biţi se codează ţinând cont de combinaţia anterioară, conform tabelului din figura 12.5. (mijloc). 2 biţi se codifică în medie cu 1,5 tranziţii deci 0,75 tranziţii pe bit. În exemplul din figura 12.5 (dreapta) sunt 7 tranziţii şi 14 biţi, ceea ce înseamnă 2 biţi pe tranziţie.

Data	Encoded
0000	11001
0001	11011
0010	10010
0011	10011
0100	11101
0101	10101
0110	10110
0111	10111

Data	Encoded
1000	11010
1001	01001
1010	01010
1011	01011
1100	11110
1101	01101
1110	01110
1111	01111

Data	Encoded
00 00	101 000
00 01	100 000
10 00	001 000
10 01	010 000
00	101
01	100
10	001
11	010

Date: 0 0 1 0 1 1 0 1 0 0 0 1 1 0

Cod: 101 001 010 100 100 000 001

Figura 12.5. Tabele de codare RLL (stânga), RLL 1,7 (mijloc) şi un exemplu de codare

Datele sunt aranjate în cilindri, fețe (capete de citire) şi sectoare, aranjarea fiind numită CHS (Cylinder-Head-Sector), figura 12.6. Cilindrul este ansamblul pistelor cu acelaşi număr de pe toate fețele discurilor. Structura unui sector depinde de tipul şi fabricantul hard discului.

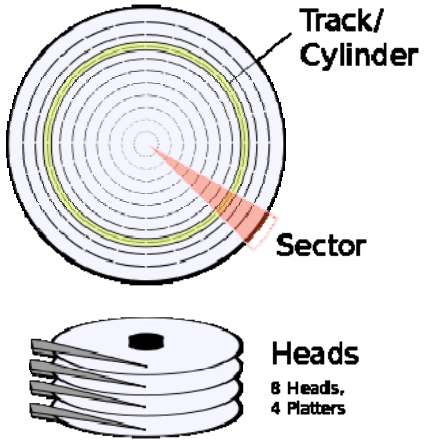


Figura 12.6. Aranjarea datelor pe hard disc în cilindri, fețe şi sectoare (CHS, Cylinder-Head-Sector)

În fiecare sector datele sunt aranjate sub formă de cadre de date, ceea ce plasează codificarea datelor alături de interfețele seriale cu informația de sincronizare ataşată unui cadru (USB, Ethernet, SATA, etc.).

Cadrul de date, figura 12.7. conține un câmp de identificare sector şi un câmp de date, separate de un interval.

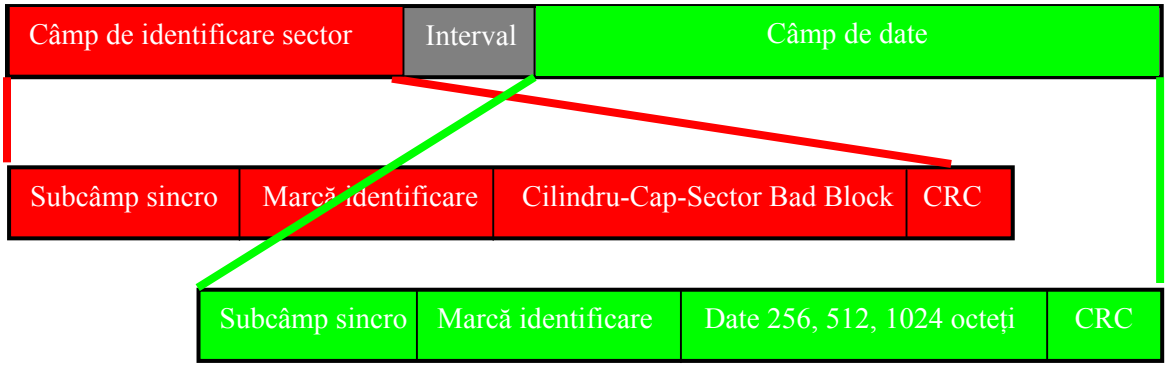


Figura 12.7. Cadrul de date

Subcâmpul de sincronizare este necesar pentru ca bucla PLL de refacere a tactului din datele citite la receptor să se poată sincroniza. Marca de identificare are o structură specială care nu respectă regula de codare, de exemplu are o succesiune de nivele de zero

sau de unu mai lungă decât cea maxim admisă de codarea de grup. Informația utilă este formată din numărul cilindrului, al capului şi a sectorului curent, precum şi marcări speciale cum ar fi Bad Block. Un sector marcat cu Bad Block de o acțiune de formatare nu este citit sau scris.

12.3. Codarea pe suporturile optice

Unitatea optică citeşte sau scrie date codificate binar pe un suport circular, numit disc optic (CD, DVD sau BlueRay).

În 1961 David Paul Gregg a înregistrat în SUA patentul unității optice. Music Corporation of America (MCA) a cumpărat patentul lui Gregg împreună cu toată compania lui. Acest reper istoric constituie începutul primei etape, **etapa CD**. În această etapă un CD poate înmagazina 700MB pe o față.

În 1989 Pioneer a înregistrat un patent pentru unități optice care a adus beneficii financiare substanțiale. După acest an a fost dezvoltată a doua etapă istorică a unităților optice, **etapa DVD**. Un DVD poate înmagazina circa 4,7GB sau 8GB (dual layer) pe o singură față.

După anul 2000 un grup de firme (Apple, Dell, Hitachi, HP, JVC, LG, Mitsubishi, Panasonic, Pioneer, Philips, Samsung, Sharp, Sony, TDK şi Thomson, grupate în asociația BDA Blu-ray Disc Association) au dezvoltat tehnologia **Blu-ray** care constituie **etapa a treia** în dezvoltarea unităților optice. Capacitatea este de 25GB sau 50GB (dual layer) pe o singură față.

În etapa a patra se prevede atingerea unor capacități de ordinul a un terabyte în tehnologii holografice sau cu discuri din materiale speciale.

Suportul este format dintr-un substrat transparent care protejează stratul care conține informația, figura 12.8. Informația este codificată prin adâncituri (ridicături) aranjate sub forma unei piste spirale. Ridicăturile au dimensiunea de l/4 şi o rază laser incidentă pe o ridicătură este reflectată 30% iar incidentă pe o adâncitură este reflectată 100%. Un strat reflectorizant asigură reflexia razei laser şi este protejat de un strat protector pe care se poate aplica eticheta comercială. Datele numerice din figură se referă la discul CD. Există suporturi de informație care pot fi scrise pe ambele părți, acestea având o structură simetrică. Principalele componente ale unei unități optice sunt arătate în figura 12.9.

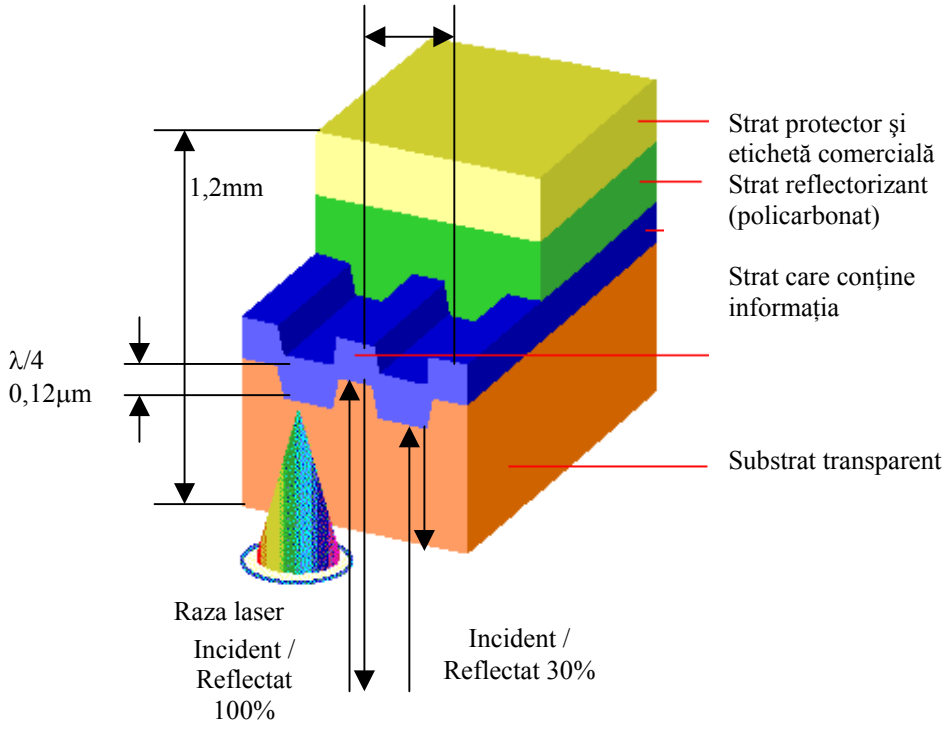


Figura 12.8. Principiul citirii optice

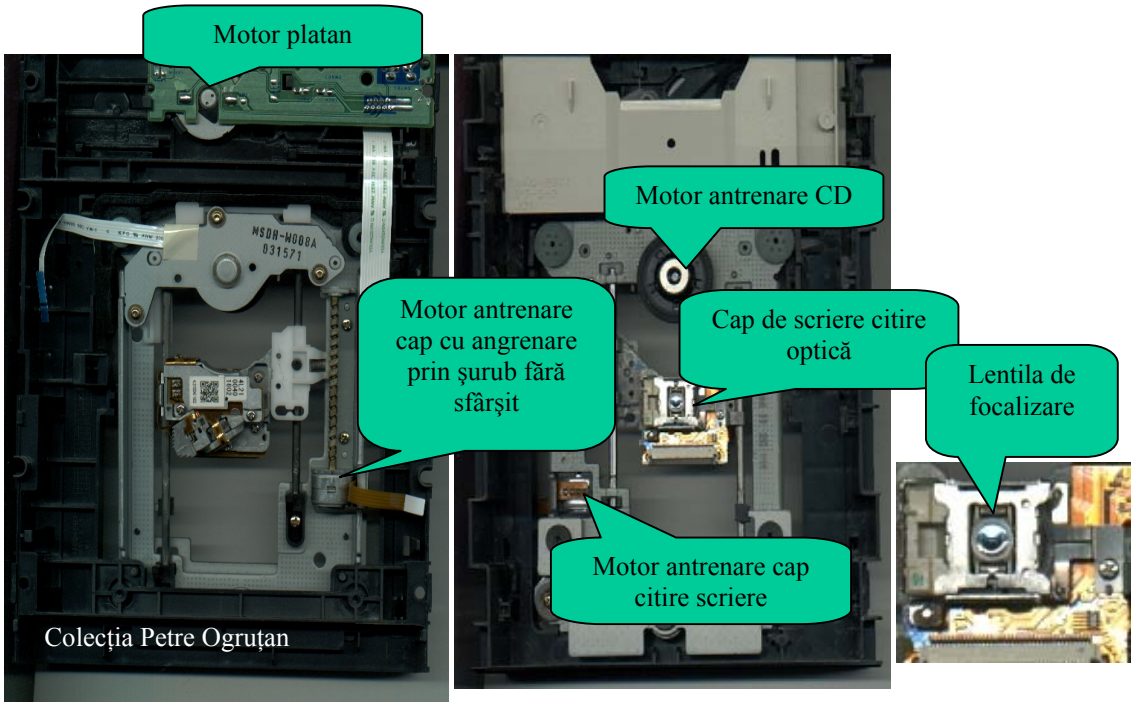


Figura 12.9. Principalele componente ale unităţii optice

Codificarea datelor pe suporturile optice este realizată prin EFM la CD şi EFM PLUS la DVD. Codarea EFM (**Eight-to-fourteen modulation)** se afirmă că a fost inventată de Kees A. Schouhamer Immink (sursa http://en.wikipedia.org/wiki/Eight-to-fourteen_modulation). Acest inventator şi om de ştiinţă danez a avut contribuţii însemnate la dezvoltarea tehnologiei optice.

Regula de codare EFM este ca între două valori de unu logic să existe intercalate minimum două şi maximum 10 valori de zero logic, codate NRZI (Non Return to Zero Inverted). La un cod de 14 biţi la care există 2^{14} = 16384 combinaţii, 267 dintre acestea îndeplinesc condiţia EFM, deci acoperitor pentru a putea codifica cele 256 de combinaţii ale cuvintelor pe 8 biţi. Astfel prin codarea EFM fiecărui cuvânt de 8 biţi îi corespunde un cod de 14 biţi. Corespondenţa cuvintelor de 8 biţi cu cele de 14 biţi este realizată practic prin tabele de conversie hardware, lista corespondenţelor între cuvintele de 8 biţi şi codurile EFM corespunzătoare fiind dată de exemplu în http://www.laesieworks.com/digicom/Storage_CD_8to14.html .

Codificarea unui cuvânt de 8 biţi şi înscrierea lui pe un suport optic poate fi exemplificată în figura 12.10.

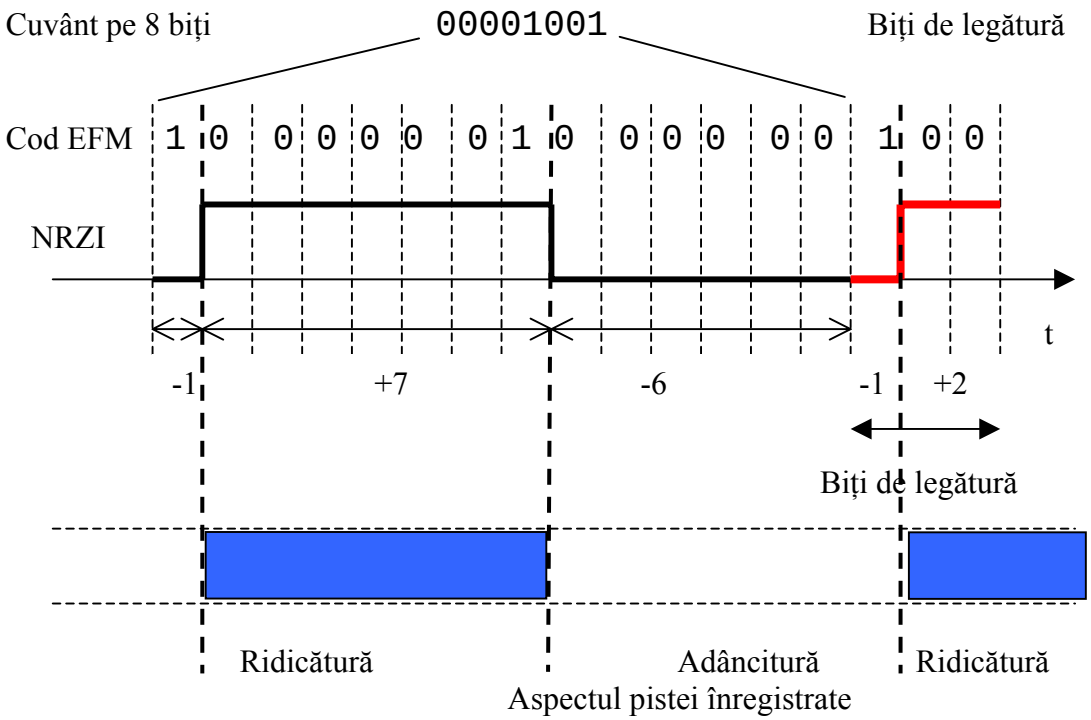


Figura 12.10. Codarea EFM

Pentru ca semnalul codat să conțină o componentă continuă cât mai mică, după cuvântul codat EFM se adaugă 3 biți, numiți biți de legătură (merging bits). Se calculează pentru fiecare cuvânt codat EFM suma digitală (DSV Digital Sum Value) prin adăugarea unui +1 pentru fiecare celulă bit în care semnalul este în 1 logic şi –1 pentru fiecare celulă bit în care semnalul este zero. Se adaugă grupul de 3 biți astfel încât DSV să fie cât mai aproape de zero. Dacă suma digitală este aproape de zero înseamnă că semnalul stă intervale de timp aproape egale în zero logic şi unu logic. Semnalul astfel codat se înscrie pe suportul optic sub forma unor ridicături şi adâncituri ca în imaginea de jos din figura 12.10.

Se poate vedea că această codificare este autosincronizabilă şi intră în categoria codificărilor de grup. Eficiența codificării este mare, în acest exemplu au fost codificați 8 biți cu 3 tranziții, adică 2,66 biți pe tranziție.

Codificarea EFM PLUS utilizată la DVD are aceleaşi reguli de codificare dar lipsesc biții de legătură. Codul EFM PLUS este generat de un automat finit.

12.4. Concluzii

Din acest capitol se poate vedea că la transmisiile seriale şi la stocarea pe suporturi magnetice şi optice codările sunt asemănătoare, ceea ce dovedeşte o frumoasă unitate a teoriei aplicate. Imagini sugestive ale zonelor înregistrate sunt date în figura 12.11. la un hard disc şi în figura 12.12. la un CD.

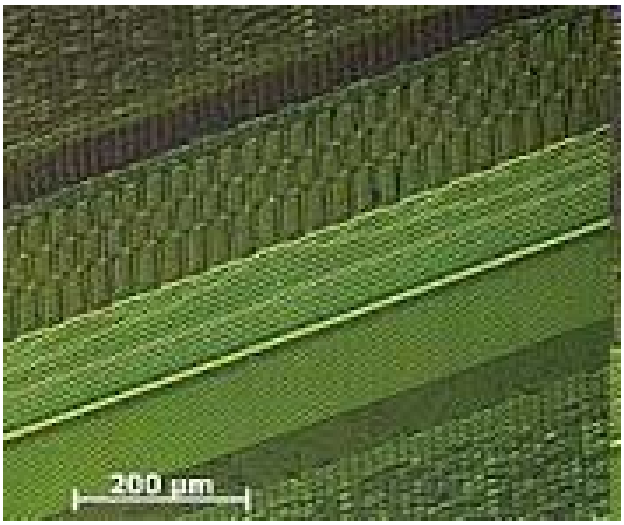


Figura 12.11. Imagine a datelor scrise pe un hard disc, sursa:
http://en.wikipedia.org/wiki/Hard_disk_drive

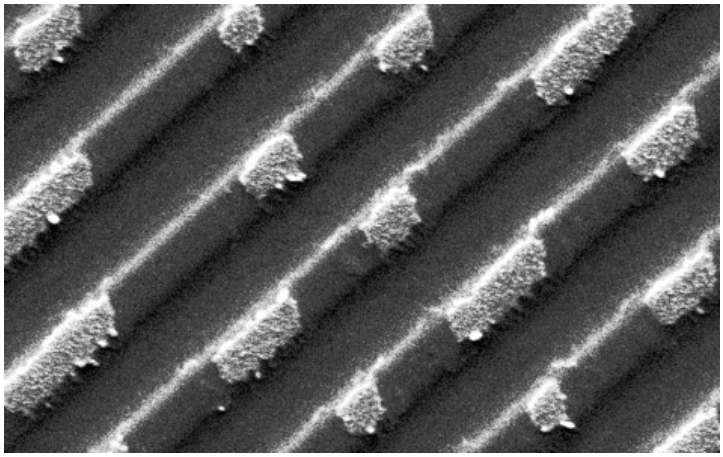


Figura 12.12. Imagine a datelor scrise pe un CD cu polimeri care reflectă lumina, imagine mărită de 10000 de ori, sursa: http://www.polymersolutions.com/psi-newsletter-archive/november-psi

Un alt exemplu de viziune unitară aplicată la diversele metode de stocare apare în tabelul din figura 12.11.:

Device	Critical feature-size F	Area (F^2)	Density (Gbit/sq. in)
Hard Disk	50 nm (MR width)	1.0	250
DRAM	45 nm (half pitch)	6.0	50
NAND (2 bit)	43 nm (half pitch)	2.0	175
NAND (1 bit)	43 nm (half pitch)	4.0	87
Blue Ray	210 nm (λ/2)	1.5	10

Figura 12.11. Tabel comparativ al limitelor principiilor de stocare

Analiza comparativă arată unitatea modurilor de stocare a informației. Principiile de stocare cu semiconductori, optice şi magnetice pot fi comparate prin densitate şi preț, (sursa https://www.usenix.org/legacy/event/fast10/tutorials/T2.pdf).

Bibliografie

Capitolul 1

1. M. Romanca, P. Ogrutan, *Sisteme cu calculator incorporat. Aplicatii cu microcontrollere*, Editura Universitatii Transilvania Brasov, 2011, pag. 1-4 online la: http://vega.unitbv.ro/~ogrutan/Microcontrollere2011/1-introducere-interfete%20paralele%20si%20seriale.pdf
2. C. Gerigan, P. Ogruţan, *Tehnici de interfaţare*, Ed. Transilvania Braşov, 2000, 315p., ISBN 973-9474-94-2, pag. 2-11, online la: http://vega.unitbv.ro/~ogrutan/ti/cap1.pdf
3. http://en.wikipedia.org/wiki/Interface_%28computing%29
4. http://en.wikipedia.org/wiki/Communications_protocol
5. https://en.wikipedia.org/wiki/Peripheral
6. http://en.wikipedia.org/wiki/Hydraulic_telegraph

Capitolul 2

7. P. Ogrutan, *Interfeţe şi echipamente periferice*, Editura Universitatii Transilvania Brasov, 1994, pag. 7-12
8. *A hardware interrupt tutorial*, http://www.best-microcontroller-projects.com/index.html
9. *Using Direct Memory Access (DMA) in STM32 projects*, http://www.embedds.com/using-direct-memory-access-dma-in-stm23-projects/
10. P. Borza, C. Gerigan, P. Ogrutan, Gh.Toacse, Microcontrollere. Aplicatii, Editura Tehnica, Bucuresti 2000, http://vega.unitbv.ro/~ogrutan/Microcontrollere/curs.pdf

Capitolul 3

11. M. Romanca, P. Ogrutan, *Sisteme cu calculator incorporat. Aplicatii cu microcontrollere*, Editura Universitatii Transilvania Brasov, 2011, pag. 5-18 online la: http://vega.unitbv.ro/~ogrutan/Microcontrollere2011/2-magistrale.pdf
12. P. Ogruţan, C. Gerigan, N. Banciu *Memorii, interfeţe şi periferice. Interfeţe specializate*, Ed. Transilvania Braşov, 2003, 190 pagini, ISBN 973-635-118-1, pag. 1-30, online la: http://vega.unitbv.ro/~ogrutan/ii/chipset.pdf
13. C. Gerigan, P. Ogrutan, Tehnici de interfatare, Editura Transilvania, Brasov 2000, pag 24-45, online: http://vega.unitbv.ro/~ogrutan/ti/cap3.pdf
14. R. Budruk, D. Anderson, T. Shanley, PCI Express System Architecture, Addison-Wesley, 2003, online: http://www.mindshare.com/files/ebooks/pci%20express%20system%20architecture.pdf

Capitolul 4

15. P. Ogrutan, C. Gerigan, Memorii, interfete si periferice, Indrumar de laborator, Reprografia Universitatii 1998, online la: http://vega.unitbv.ro/~ogrutan/lab/LAB31.pdf, pag 5-8
16. Intel 8255, online la: http://en.wikipedia.org/wiki/Intel_8255
17. K. Ray, K.M. Bhurkhandi, Advanced Microprocessors And Peripherals, McGraw-Hill, 2006

18. http://books.google.ro/books?id=KJNpD2KimEsC&pg=PA184&lpg=PA184&dq=intel+8255+16+bit&source=bl&ots=eLAuXGt1Xd&sig=CngNOTU-CB_y_Je6pZIrog4RmRI&hl=ro&sa=X&ei=mVRyUvXTI5Kk4ATKtYHgDA&ved=0CHQQ6AEwCQ#v=onepage&q=intel%208255%2016%20bit&f=false, pag 184-211

Capitolul 5

19. M. Romanca, P. Ogrutan, *Sisteme cu calculator incorporat. Aplicatii cu microcontrollere*, Editura Universitatii Transilvania Brasov, 2011, pag. 1-4 online la: http://vega.unitbv.ro/~ogrutan/Microcontrollere2011/1-introducere-interfete%20paralele%20si%20seriale.pdf
20. Communication Systems, Line Codes, online la: http://en.wikibooks.org/wiki/Communication_Systems/Line_Codes
21. Sebastian Andrews, Communication Networks, online la: http://services.eng.uts.edu.au/~kumbes/ra/Transmission/Digital%20Wave%20Formatting/index.htm
22. Bit stuffing, online la: http://en.wikipedia.org/wiki/Bit_stuffing
23. C. Gerigan, P. Ogruţan, *Tehnici de interfaţare*, Ed. Transilvania Braşov, 2000, 315p., ISBN 973-9474-94-2, pag. 69-93, online la: http://vega.unitbv.ro/~ogrutan/ti/cap5.pdf
24. RS-232 online la: http://en.wikipedia.org/wiki/RS-232
25. RS232 detalii, online la: http://academic.evergreen.edu/projects/biophysics/technotes/electron/serial.htm
26. I8251 online la: http://www.electronics.dit.ie/staff/tscarff/8251usart/8251.htm
27. http://creativeelectron.net/blog/2009/10/pic-serial-communition-in-pic-microcontroller-1/

Capitolul 6

28. M. Romanca, P. Ogrutan, *Sisteme cu calculator incorporat. Aplicatii cu microcontrollere*, Editura Universitatii Transilvania Brasov, 2011, pag. 1-4 online la: http://vega.unitbv.ro/~ogrutan/Microcontrollere2011/
29. Wagner Lipnharski , 8051+LCD+EPROM, online: www.ustr.net
30. P.Ogrutan, *Microcontrollere si controllere grafice Fujitsu*, Ed. Universitatii Transilvania Brasov, 2006, 182 pag, ISBN 973-635-621-3
31. P. Borza, C. Gerigan, P. Ogruţan, Gh. Toacşe, *Microcontrollere. Aplicaţii*, Editura Tehnică Bucureşti, 2001, ISBN973-31-1577-6

Capitolul 7

32. M. Romanca, P. Ogrutan, *Sisteme cu calculator incorporat. Aplicatii cu microcontrollere*, Editura Universitatii Transilvania Brasov, 2011, pag. 1-4 online la: http://vega.unitbv.ro/~ogrutan/Microcontrollere2011/
33. C. Gerigan, P. Ogruţan, *Tehnici de interfaţare*, Ed. Transilvania Braşov, 2000, 315p., ISBN 973-9474-94-2, pag. 154-174, online la: http://vega.unitbv.ro/~ogrutan/ti/index.html
34. P.Ogrutan, *Microcontrollere si controllere grafice Fujitsu,* Ed. Universitatii Transilvania Brasov, 2006, 182 pag, ISBN 973-635-621-3, pag. 125-134, online: http://vega.unitbv.ro/~ogrutan/Microcontrollere%20Fujitsu/interfete3-112-136.pdf ,

Capitolul 8
35. P. Ogruțan, C. Gerigan, N. Banciu *Memorii, interfețe şi periferice. Interfețe specializate*, Ed. Transilvania Braşov, 2003, 190 pagini, ISBN 973-635-118-1, pag. 31-56, online la: http://vega.unitbv.ro/~ogrutan/ii/retea.pdf
36. Ayres, J. *Using the CRYSTAL CS8900A in 8 Bit Mode*, AN181, online: www.crystal.com
37. Werner C. *Implementation notes for an Ethernet solution usable for embedded systems using the RTL8019*, 2002, online: cornelius@ethernet.isdn-development.de
38. http://netmedia.com/siteplayer/

Capitolul 9
39. M. Romanca, P. Ogrutan, *Sisteme cu calculator incorporat. Aplicatii cu microcontrollere*, Editura Universitatii Transilvania Brasov, 2011, pag. 19-31 online la: http://vega.unitbv.ro/~ogrutan/Microcontrollere2011/3-usb-ieee1394.pdf
40. C. Gerigan, P. Ogruțan, *Tehnici de interfațare*, Ed. Transilvania Braşov, 2000, 315p., ISBN 973-9474-94-2, pag. 114-130, online la: http://vega.unitbv.ro/~ogrutan/ti/index.html
41. http://www.usb.org/home
42. http://www.ftdichips.com/

Capitolul 10
43. M. Romanca, P. Ogrutan, *Sisteme cu calculator incorporat. Aplicatii cu microcontrollere*, Editura Universitatii Transilvania Brasov, 2011, pag. 1-4 online la: http://vega.unitbv.ro/~ogrutan/Microcontrollere2011/7-aplicatii%20mobile.pdf
44. P. Ogruțan, C. Gerigan, N. Banciu "Memorii, interfețe şi periferice. Interfețe specializate", Ed. Transilvania Braşov, 2003, 190 pagini, ISBN 973-635-118-1
45. C. Gerigan, P. Ogruțan, "Tehnici de interfațare", Ed. Transilvania Braşov, 2000, 315p., ISBN 973-9474-94-2
46. P. Ogruțan, C. Gerigan, N. Banciu "Memorii, interfețe şi periferice. Interfețe specializate", Ed. Transilvania Braşov, 2003, 190 pagini, ISBN 973-635-118-1
47. C. Gerigan, P. Ogruțan, "Tehnici de interfațare", Ed. Transilvania Braşov, 2000, 315p., ISBN 973-9474-94-2

Capitolul 11
48. M. Romanca, P. Ogrutan, *Sisteme cu calculator incorporat. Aplicatii cu microcontrollere*, Editura Universitatii Transilvania Brasov, 2011, pag. 31-42, online la: http://vega.unitbv.ro/~ogrutan/Microcontrollere2011/
49. C. Gerigan, P. Ogruțan, *Tehnici de interfațare*, Ed. Transilvania Braşov, 2000, ISBN 973-9474-94-2, pag 137-145, online la: http://vega.unitbv.ro/~ogrutan/
50. C. Erickson, *Department of Electrical and Computer Engineering*, Auburn University, Auburn, AL 36849 Chris.Erickson@auburn.edu, www.eng.auburn.edu

Capitolul 12

51. M. Romanca, P. Ogrutan, *Sisteme cu calculator incorporat. Aplicatii cu microcontrollere*, Editura Universitatii Transilvania Brasov, 2011, pag. 1-4 online la: http://vega.unitbv.ro/~ogrutan/Microcontrollere2011/
52. R. Freitas, L. Chiu, *Solid-State Storage: Technology, Design and Applications*, IBM Almaden Research Center, 2010, online: https://www.usenix.org/legacy/event/fast10/tutorials/T2.pdf

Cuprins