Lucrarea 10

Portul paralel la calculatoarele IBM-PC

1. Introducere

Iniţial imprimantele se conectau la calculatoare prin intermediul interfeţelor seriale, dar mai târziu datorită dezvoltării tehnologiei, imprimantele au devenit din ce în ce mai rapide şi transmisia serială a fost înlocuită de transmisia paralelă. Atunci când s-a proiectat acest mod de comunicare în paralel al datelor (denumit și "mod Centronics"), s-a considerat legătura specifică dintre PC și imprimantă, asigurând astfel un singur sens al datelor: dinspre PC înspre imprimantă. Prin acest port se pot transmite date doar într-o singură direcţie (de la calculator spre periferic) la o viteză tipică de 50 KB/s dar poate ajunge şi până la 150 KB/s. Ulterior însă, datorită avantajelor prezentate de acest mod de transfer, din ce în ce mai multe dispozitive au început să fie conectate pe portul parallel la PC, apărând astfel nevoia de comunicare și în sens invers a datelor. Prima posibilitate de a transfera date și în sens invers prin portul parallel a fost implementată la sistemele tip IBM PS2 (1987); acestea, permiteau selectarea sensului invers de transfer al datelor prin intermediul pinului C5 din RC al portului, denumit "Dir" (de la Direction). Deși la bootarea sistemului, în mod automat BIOS-ul configurează portul parallel de date doar de ieșire, din program se putea modifica ulterior valoarea acestui bit în 1 pentru a asigura sensul de intrare al datelor în PC. Ieșirile registrului de date erau trecute în starea de înaltă impedanță, permițând astfel citirea semnalelor aplicate pe liniile de date prin registrul de reacție de pe date. Această îmbunătățire a asigurat conectarea perifericelor (precum scanner, instrumente de achiziție de semnal, etc.) la PC prin conectorul de port parallel și nu printr-un slot de expansiune al plăcii de bază.

Mai târziu, au fost implementate și alte modalități de a asigura transferul bidirecțional al datelor prin portul paralel, dar în permanență s-a ținut cont de promisiunea celor de la IBM de "backward compatibility" (orice sistem nou sau componentă nouă lansată pe piață, să asigure compatibilitatea cu produsul anterior – din considerente de marketing); deși cele 2 variante de port paralel apărute ulterior, denumite ***EPP (Enhanced Parallel Port)*** și ***ECP (Extended Capability Port)*** diferă foarte mult d.p.d.v. al facilităților oferite de varianta de port inițială (denumită ***SPP (Standard Parallel Port)***), ca implementare, acestea au moștenit toți regiștrii SPP (**RD (registrul de date)** la AB[1]+0, **RS (registrul de stare)** la AB+1, **RC (registrul de control)** la AB+2).

Inițial, cei 3 regiștrii **SPP** dețineau următoarele semnale:

- 8 biți de date – pe aceștia se va transmite informația înspre imprimantă (sau periferic) – în **RD**, la AB+0;
- Un semnal Strobe (care să spună perifericului când există date valide pe cele 8 linii de date) și încă 3 semnale de control al perifericului (să treacă automat sau nu la o linie nouă, să se inițializeze, resp. să se activeze) – în **RC**, la AB+2;
- Un semnal Ack (de confirmare a faptului că perifericul a preluat datele de pe cele 8 linii de date) și încă 4 semnale de stare a perifericului (raportat la imprimantă: dacă e ocupată, dacă mai are hârtie, dacă e online și resp. dacă există vreo eroare de funcționare) – în **RS**, la AB+1.

Portul paralel de tip EPP folosește în plus încă 5 adrese de port (până la AB+7), iar portul paralel de tip ECP folosește în plus încă 3 adrese de port, începând de la o adresă de port cu offset 400h (al 4-lea registru se află la AB+400h, al 5-lea registru se află la AB+401h, iar al 6-lea registru se află la AB+402h).

Implementarea portului EPP a adus avantaje d.p.d.v. al vitezei de transfer a datelor, reușind aceasta în special din cauza faptului că semnalele de handshaking dintre PC și perifericul conectat pe port sunt implementate prin hardware și nu prin software cum erau inițial, la varianta SPP. Pe de altă parte, portul ECP deține toate avantajele portului de tip EPP, dar în plus, deține capabilități de transfer DMA și de compresie a datelor (RLE).

Din punct de vedere logic, interfaţa paralelă standard este formată din două registre şi un buffer de intrare, ocupând trei adrese consecutive din spaţiul adreselor de porturi al procesorului. Portul paralel standard conţine un registru de date pe 8 biţi unidirecţional (la portul paralel standard bidirecţional acest registru este bidirecţional), un registru de control bidirecţional pe 6 biţi dintre care doar 4 linii sunt legate la connector-ul interfeţei paralele şi un buffer de intrare pe 8 biţi din care sunt folosiţi doar 5 pentru liniile de stare de la periferic.

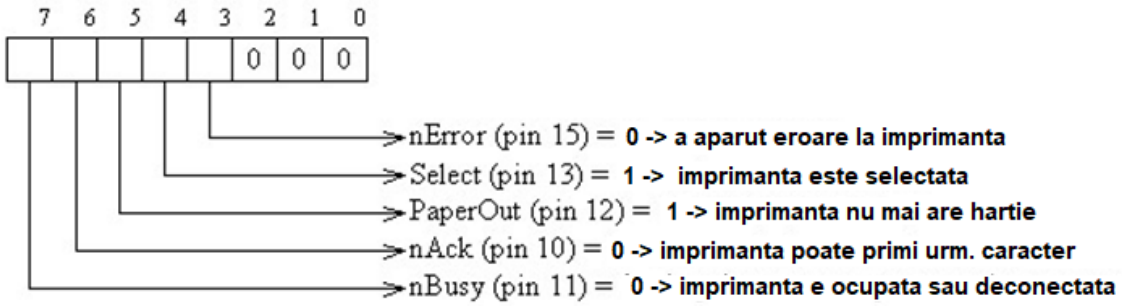
2. Portul paralel standard (SPP)

Conectorul standard al portului paralel este de tip DB 25S (mamă) cu 25 contacte, plasat pe panoul spate al PC. Conectarea la imprimantă se face printr-un cablu (L< 2m) cu un conector DB 25P (tată) spre PC şi un conector Centronics cu 36 de pini spre imprimantă. Semnalele care se utilizează se împart în 3 grupe:

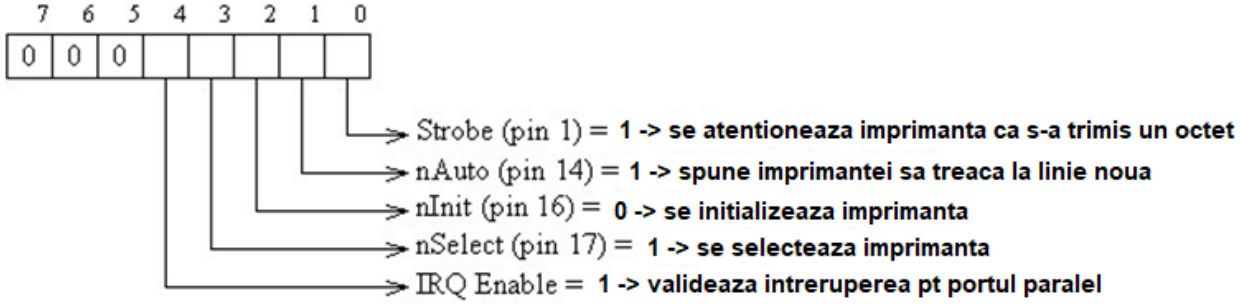
- 8 linii de date (D0-D7)
- 5 linii de stare (S3-S7)
- 4 linii de control (C0-C3)

[1] AB = adresă de bază

Registrul de stare : Se află la adresa AB+1 şi poate fi accesat doar în citire, scrierile fiind ignorate.



Registrul de control : Se află la adresa AB+2 şi poate fi accesat atât în scriere cât şi în citire.

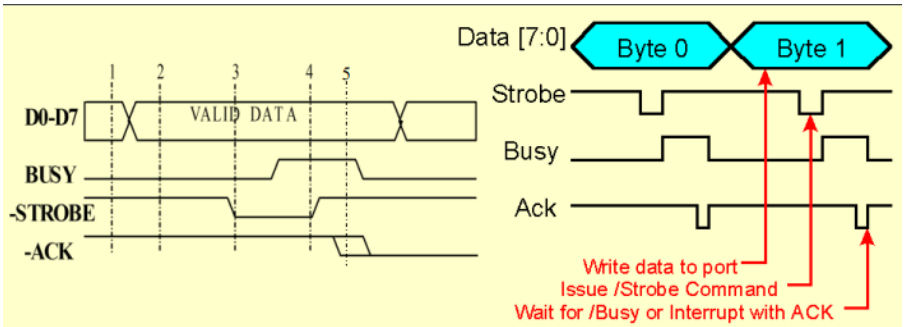


Liniile corespunzătoare registrului de control sunt bidirecţionale. Ieşirile registrului de control sunt legate la cuplă prin intermediul unor inversoare Open-Colector, pentru a asigura procesorului şi posibilitatea de a citi aceste linii.
Biţii 4 şi 5 sunt pentru controlul intern al interfeţei paralele. Bitul 4 este destinat validării întreruperilor iar prin bitul 5 se validează intrarea pentru registrul de date (la modul PS/2).

Programarea interfeţei paralele standard

Pentru trimiterea unui caracter la periferic se poate folosi un program care implementează următorii paşi:

1. **Verifică semnalul Busy (RS)**
2. Dacă BUSY=0, Scrie codul caracterului în registrul de date (RD).
   (Verifică dacă perifericul nu este ocupat, testând linia Busy din registrul de stare. Dacă perifericul este ocupat (Busy=1), aşteaptă.)
3. Dacă Busy nu este activ (=0), activează semnalul de **Strobe** din RC (în 0), specificându-i perifericului că are date disponibile pe liniile de date;
4. Dacă Busy devine activ (=1,) se dezactivează **Strobe** (în 1) din RC (după aproximativ 5microsec)
5. Când **ACk** devine activ, se confirmă PC-ului că s-a făcut preluarea octetului de către imprimantă



Adresarea interfeţei paralele

O intefaţă paralelă este identificată prin adresa ei de bază şi printr-un identificator atribuit de BIOS: LPT1, LPT2, LPT3, ...
Interfaţa paralelă are alocate trei adrese de bază:

Adresa	Descriere
3BCh – 3BFh	Folosite de interfeţele paralele incorporate în plăcile video. (Ex. PS/2)
378h – 37Fh	Adresa uzuală pentru LPT1
278h – 27Fh	Adresa uzuală pentru LPT2

La pornirea calculatorului, BIOS-ul (Basic Input/Output System) va determina numărul porturilor din sistem şi la va atribui denumirile LPT1, LPT2 şi LPT3. BIOS-ul verifică, prima dată, dacă există un port la adresa 3BCh, dacă există îi atribuie numele LPT1; apoi verifică adresa de port 378h, dacă găseşte un port aici îi atribuie următoarea denumire liberă; şi în final verifică adresa 278h şi dacă există un port la această adresă îi atribuie şi acestuia următoarea denumire nefolosită. Pentru a determina ulterior care sunt adresele alocate pentru LPT1, LPT2 şi LPT3 se pot consulta următoarele adrese din zona de date BIOS, unde există tabelul portrilor LPT:

Adresa de start	**Ce arată:**
0000:0408	Adresa de bază pentru LPT1
0000:040A	Adresa de bază pentru LPT2
0000:040C	Adresa de bază pentru LPT3
0000:040E	Adresa de bază pentru LPT4 (BIOS-urile mai noi)

Acest tabel conţine trei cuvinte pe 16 biţi (patru la unele BIOS-uri) fiecare intrare conţinând adresa I/O de bază a portului paralel, dacă acesta există, sau 0 dacă acesta nu există.

Adresele de bază (AB) uzuale ale ***porturilor paralele*** în sisteme PC sunt: **3BCh, 378h, 278h**[2] **(LPT1, LPT2, LPT3)**.

- ROM BIOS-ul rezervă în memoria RAM **4 cuvinte de câte 16 biţi fiecare**, începând cu **adresa 0000:408h**, unde salvează *adresa de bază a porturilor paralele* detectate după reset. Sistemul de operare DOS şi BIOS-ul foloseau aceste adrese asociindu-le dispozitivelor de tip LPT (Line Print Terminal) standard pt imprimantă LPT1 şi LPT2, resp. LPT3 şi chiar LPT4 (adresa celui de-al 4-lea dispozitiv LPT4 nu este standardizată).

Semnalele și regiștrii asociați

Convenţiile folosite la descrierea fizică a portului sunt următoarele:

‘**-**’ ***în faţa semnalelor*** indică faptul că semnalul respectiv este ***activ pe nivel “low”*** (în Tabelul 1, acest lucru a fost semnalat cu „n” în fața numelui semnalului: la Strobe (C_0), Autofeed (C_1), Init (C_2), Selin (C_3), Error (S_3), Ack (S_6));

- în Figurile 6 și 7, acest lucru e semnalat prin cerculeț;

‘-’ ca sufix arată faptul că ***valoarea bitului (din registrul portului paralel) corespunzător semnalului a fost inversat*** (la Strobe (C0), Autofeed (C_1), Selin(C_3), Busy (S_7));

- în Figura 6, acest lucru e semnalat printr-un inversor,
iar în Figura 7 apare un minus în fața numelui semnalului;

de exemplu, la S_7(“Busy”), imprimanta trimite semnal de valoare logică 1, iar în registrul portului paralel ajunge ca 0; d.p.d.v. al unui bit de control, de ex, dacă vrem să trimitem 0 prin Selin (C_3), va trebui să înscriem în registrul de control pe bitul 3 valoarea 1.

Tabel1: Semnificațiile pinilor conectorului de port paralel standard (SPP)

Pin DB 25	Pin Centronics	Semnificaţie Pin SPP	Notație	Bit Registru	In/Out
1	1	- Strobe	**nStrobe -**	**C0 -**	Out
2	2	Data 0	**D0**	**D0**	Out
3	3	Data 1	**D1**	**D1**	Out
4	4	Data 2	**D2**	**D2**	Out
5	5	Data 3	**D3**	**D3**	Out
6	6	Data 4	**D4**	**D4**	Out
7	7	Data 5	**D5**	**D5**	Out
8	8	Data 6	**D6**	**D6**	Out
9	9	Data 7	**D7**	**D7**	Out
10	10	- Ack	**nAck**	**S6**	In
11	11	Busy	**Busy -**	**S7 -**	In
12	12	PaperEnd	**PaperEnd**	**S5**	In
13	13	SelectOut	**SelOut**	**S4**	In
14	14	- Autofeed	**nAutofeed -**	**C1 -**	Out
15	32	- Error	**nError**	**S3**	In
16	31	- Init	**nInit**	**C2**	Out
17	36	- Selin	**nSelin -**	**C3 -**	Out
18-25 (12 pini)	16,17,19-30,33 (15 pini)	GND	**GND**	-	-

[2] Sau după cum reiese (o eventuală schemă a decodificatorului de adresă) din specificaţiile tehnice ale plăcii de bază a sistemului

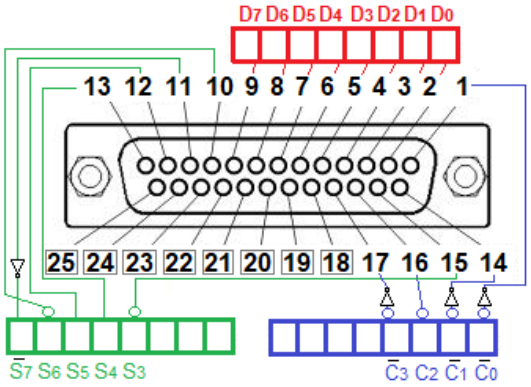


Figura 6. Corespondența pinilor cu biții din regiștrii portului

Registrul de DATE (Adr. Bază) - In acest registru se memorează datele ce urmează a fi transmise. Ieşirile portului paralel corespunzătoare acestui registru sunt ieşiri de nivele TTL. Portul dispune şi de un registru de reacţie pe date care permite citirea datelor înscrise.

D7	D6	D5	D4	D3	D2	D1	D0
Data 7	**Data 6**	**Data 5**	**Data 4**	**Data 3**	**Data 2**	**Data 1**	**Data 0**

Registrul de STARE (Adr. Bază +1) - Din acest registru se citesc de către sistem semnalele de stare ale perifericului.

S7	S6	S5	S4	S3	S2	S1	S0
Busy	**Ack**	**PaperEnd**	**SelOut**	**Error**			

Registrul de CONTROL (Adr. Bază +2) – cu ajutorul acestui registru, sistemul controlează diverse acțiuni asupra perifericului (se comandă perifericul)

C7	C6	C5	C4	C3	C2	C1	C0
		Dir	**IrqEn**	**Selin -**	**Init**	**Autofeed -**	**Strobe -**

Bulina din Figura 7 arată că semnalul are ”n ” ca prefix, deci că pinul e activ pe 0, nu pe 1;
semnul ”– ” ca sufix arată că valoarea în registrul corespunzător a fost inversată: ex: Strobe, Busy, Autofeed, Selin, dar nu și Init.

Semnalul **IrqEn** se folosește pentru a **valida apariţia întreruperii hard pe IRQ7, pe frontul crescător al semnalului Ack (atunci când de la imprimantă sau de la periferic vine semnal, se poate trimite o întrerupere prin linia IRQ7 a PIC-ului pe linia INTR de la CPU); pentru ca atunci când apare impuls pe Ack, să se trimită întrerupere pe linia IRQ7 a PIC-ului, e nevoie ca bitul b4 din registrul de control să fie programat în 1; în plus, nu trebuie uitat că IF (Interrupt Flag) trebuie validat, deci trebuie ca IF=1.**

Semnalul **Dir** s-a folosit începând de la sistemele tip PS2 pt a **selecta direcţia semnalelor de date, astfel:**
dacă Dir=0, datele vor fi „de ieşire”, iar
dacă Dir=1, datele vor fi „de intrare” (numai la porturile bidirecţionale - la PS2)
Inițial, acest pin al portului a fost conectat la masă: datele doar ieșeau din PC înspre imprimantă (sau dispoz. paralel conectat pe port);
Registrul de date (AB+0):
În cazul calculatoarelor de tip PS2, portul de date este bidirecţional; prin intermediul bitului C5 din registrul de control („Dir”) se poate inversa sensul datelor, astfel putându-se citi date introduse din exterior (de la un periferic conectat pe portul paralel).
Registrul de stare (AB+1):
Busy (-): este un semnal activ pe High, care ajunge inversat în registrul portului: imprimanta trimite Low pe acest pin atâta timp cât încă nu e gata să primească date noi (imprimanta este ocupată sau bufferul de date este plin); bitul din registrul de stare (**RS**) va ajunge în 0 atunci când imprimanta este „ready”(adică nu e “Busy”) și deci poate prelua date noi;
nAck (-): atunci când de la imprimantă vine semnal în Low (pt că pinul Ack e activ pe 0) pentru minim 10 ms, va confirma sistemului că perifericul a preluat ultimul caracter trimis prin **RD**, deci poate trimite următorul;
PaperEnd: imprimanta trimite High pe acest pin (ce ajunge ca bit de 1 în registrul **RS** pe bitul 5) pentru a semnala faptul că nu mai are hârtie;
SelOut (sau **Select**)**:** imprimanta trimite High pe acest pin pt a arăta că e selectată;
nError (-) (semnal activ pe Low) pus în Low de imprimantă pt a arăta că a apărut o eroare la tipărire, imprimanta este neoperaţională sau nu mai are hârtie (arată deci o posibilă eroare în funcționare);

Registrul de control (AB+2):

nStrobe (-,-): printr-un impuls pozitiv (în "1") de cel puţin 1μs se anunţă imprimanta că o nouă dată este validă pe liniile date; astfel, în **RC**, pentru a comanda acest lucru, se va înscrie 0 pe bitul 0;

nAutofeed (-,-): dacă este în "1", spune imprimantei să treacă la o nouă linie după recepţionarea unui caracter CR;

nInit (-): printr-un impuls mai mare de 50μs în "0", se iniţializează imprimanta; în registrul de control se programează 0 pe bitul 2;

nSelin (-,-): pus în Low de PC pt a permite printarea; în registrul **RC**, bitul 3 se programează în 1, fiind apoi trimis inversat;

Semnalul **IrqEn** se folosește pentru a **valida apariţia întreruperii hard pe IRQ7, pe frontul crescător al semnalului Ack (dacă IrqEn=1, se validează Irq7; dacă IrqEn=0, se invalidează Irq7); la poarta AND, deoarece semnalul Ack e activ pe Low, semnalul trebuie inversat;**

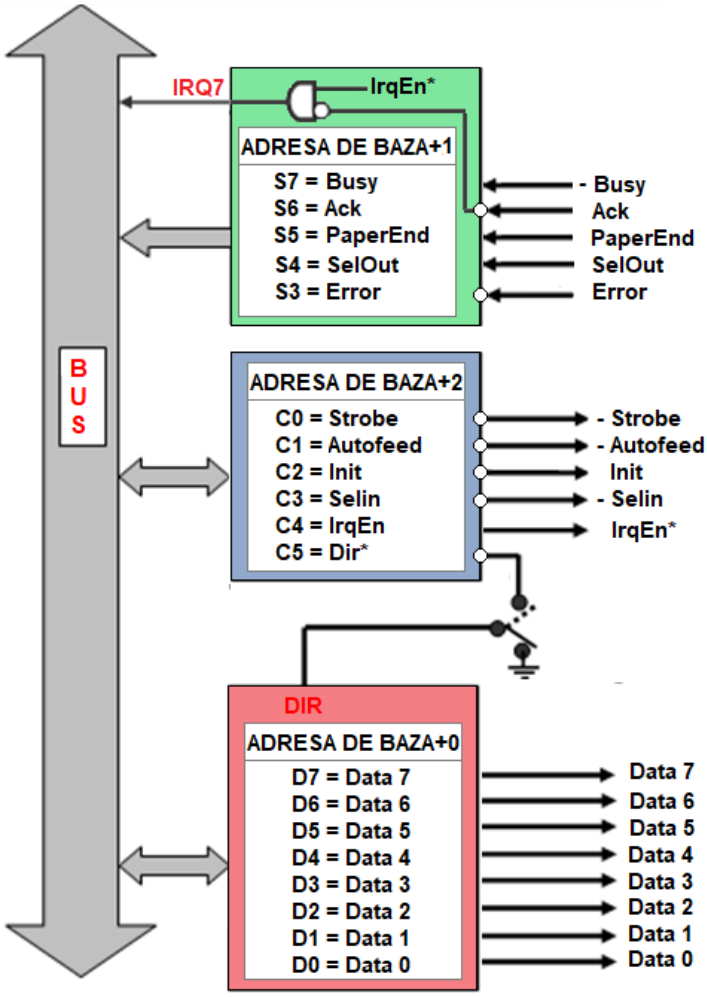
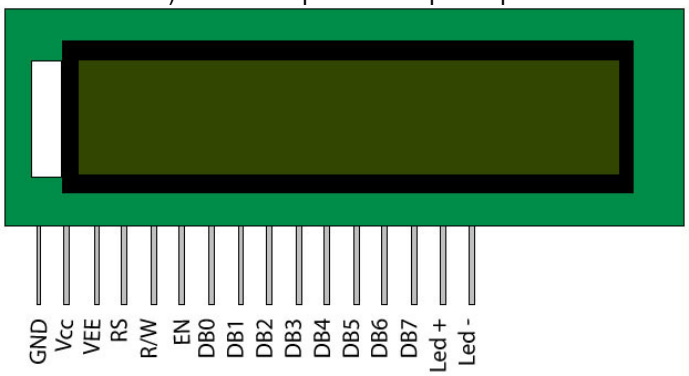


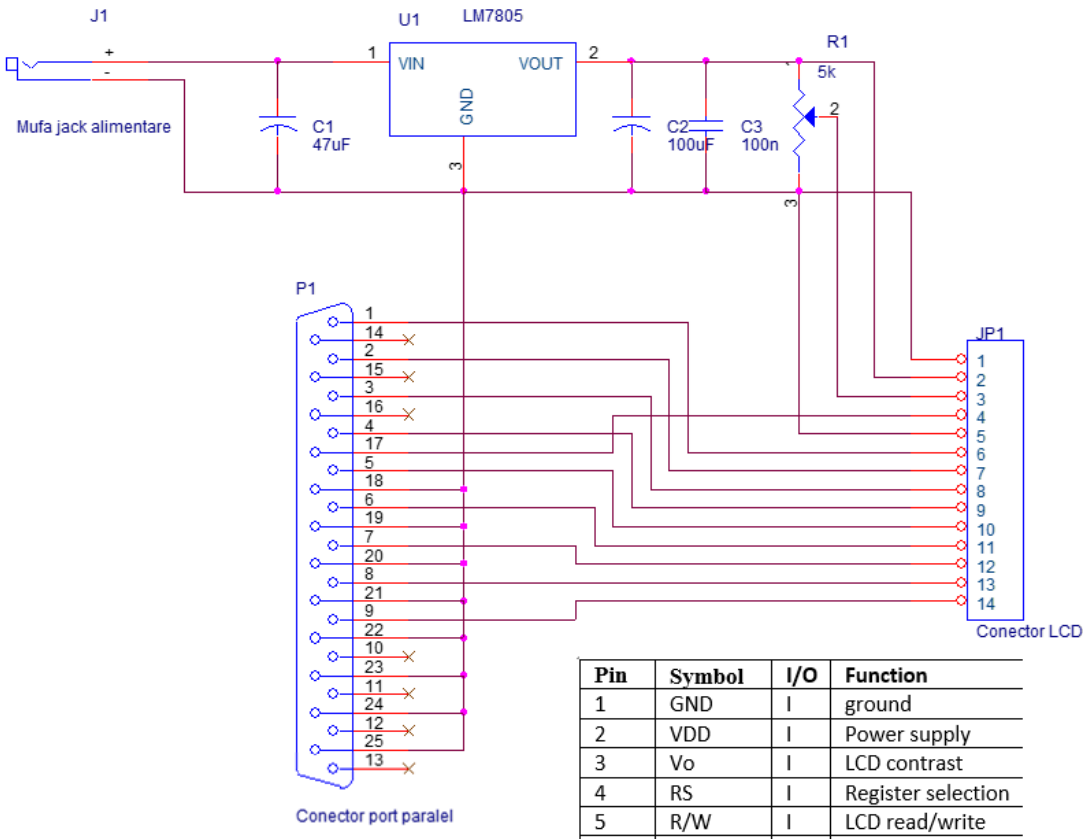
Fig.7 Schema bloc a portului paralel standard la IBM PC / PS2

Aplicatia 1: Conectarea unui LCD pe portul paralel:

Fie o plăcuță pe care se conectează un LCD tip 16x1, de exemplu PVC160101P(N), având pinul R/W tras la masă. LCD-ul e format din mai multe componente: afișajul propriu-zis, un microcontroller, o memorie DDRAM (Display Data RAM) și o memorie CGRAM (Character Generator RAM). LCD-ul dispune de 14 pini după cum urmează:



In gen. exista 2 abodari posibile: **Busy Flag** sau **Timed Delay** (**Synchronization between LCD commands and between data access operations is based on reading the LCD *busy flag* or on time delay loops**)



Pin	Symbol	I/O	Function
1	GND	I	ground
2	VDD	I	Power supply
3	Vo	I	LCD contrast
4	RS	I	Register selection
5	R/W	I	LCD read/write
6	E	I	Activation signal
7÷14	DB0÷DB7	I	Data bus

Conectorul afişajului (LCD) are 14 pini a căror semnificatie este prezentată în tabelul următor:

Tabel 1: Descrierea pinilor LCD-ului

Pin	Simbol	I/O	Funcție	Conectare în aplicație
1	Gnd	I	Masă	GND (pin3 pe schemă)
2	VDD	I	Alimentare	Vout (pin 2 pe schemă LM7805)
3	VEE	I	Contrast LCD	Potențiometru
4	RS (Register Selection)	I	Selecție registru intern	**Pin 17 PP: Selin** dacă **RS=0** -> se selectează **RI** (Registrul de instrucțiuni) al LCD-ului dacă **RS=1** -> se selectează **RD** (Registrul de date) al LCD-ului
5	R/W (Read/ Write)	I	Citire/ Scriere LCD	dacă R/W=0 -> se selectează operația de scriere în LCD dacă R/W=1 -> se selectează operația de citire din LCD **la noi: la Gnd -> folosim variant cu Delay, nu cu citire BusyFlag**
6	E (Enable)	I	Semnal de activare	**Pin 1 PP: Strobe** Semnal de ceas (Clock ⊓↓) pentru inițierea transferului de pe pinii de date: acest pin este folosit de LCD pentru a reține (latch) informația de pe pinii de date; e nevoie de un impuls de minim 450 ns ca LCD-ul să preia datele de pe pinii de date (pe *frontul descrescător* al semnalului); **E=0 -> Disable LCD,** **E=1 -> Enable LCD**
7-14	DB0-DB7	I/O	Bus de date	Cei 8 pini de date sunt folosiți pentru a citi sau înscrie „informație" în LCD, în unul din cei 2 regiștri interni ai LCD-ului (registrul de comenzi sau registrul de date)

Tabel 2: Comenzi tipice (instrucțiuni) pe care le poate primi LCD-ul:

Instrucțiune /comandă	COD										DESCRIERE	Timp tipic de execuție
	RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0		
Clear display	0	0	0	0	0	0	0	0	0	1	Clears Display & Returns to Address 0 – home position. Set I/D= 1 for the Entry Mode Set	1.64ms
Return home	0	0	0	0	0	0	0	0	1	*	Returns Cursor to Address 0 (set the DDRAM address to 0) – home position. Also returns the display being shifted to the original position. DDRAM contents remain unchanged.	1.64ms
Entry mode set	0	0	0	0	0	0	0	1	I/D	S	Set the shifting direction of the cursor and the possibility of shifting: *I/D: Set Cursor Moving Direction* I/D=1: Increment; I/D=0: Decrement *S: Specify Shift of Display* S=1: The display is shifted S=0: The display is not shifted	40µs
Display ON/OFF control	0	0	0	0	0	0	1	D	C	B	*Display* D=1: Display on; D=0: Display off *Cursor* C=1: Cursor on; C=0: Cursor off *Blink* B=1: Blink on; B=0: Blink off	40µs
Cursor/display shift	0	0	0	0	0	1	S/C	R/L	*	*	Moves cursor or shifts the display without changing DDRAM contents S/C=0: Cursor Shift (RAM unchanged) S/C=1: Display Shift (RAM unchanged) R/L=1: Shift to the Right R/L=0: Shift to the Left	40µs
Function set	0	0	0	0	1	DL	N	F	*	*	Sets data bus length (DL), number of display lines (N), and character fonts (F). DL=1 for 8 bits; DL=0 for 4 bits, F=0 for 5x7 dots; F=1 for 5x10 dots N=0 for 1 line display; N=1 for 2 lines display	40µs
Set CGRAM address	0	0	0	1	MSB ←----ACG---→					LSB	Sets CGRAM address. CGRAM data is sent or/and received after this instruction.	40µs
Set DDRAM address	0	0	1	MSB ←----ADD---→						LSB	Sets DDRAM address. DDRAM data is sent or/and received after this instruction	40µs
Read busy flag and address	0	1	BF	MSB ←----AC---→						LSB	Reads Busy Flag BF and address counter contents.	40µs
Write data in CGRAM or DDRAM	1	0	MSB							LSB	Write data in CGRAM or DDRAM	40µs
Read data from CGRAM or DDRAM	1	1	MSB ←---- ACG ---→							LSB	Read data from CGRAM or DDRAM	40µs

00h = 1000 0000b = **80h**
40h = 1100 0000b = **C0h**

Harta memoriei DDRAM pentru afişajul de 16×1 caractere:

Caracter:

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
00	01	02	03	04	05	06	07	40	41	42	43	44	45	46	47

Pentru setarea adresei DDRAM în care sunt stocate de către microcontrolerul intern al LCD-ului datele ce vor fi procesate sau afişate în funcţie de starea flagului RS se transmite pe busul de date D0-D7 :

o valoare de 1 (bitul DB7) urmata de alti 7 biti (bitii DB6-DB0)

Deci adresa memoriei DDRAM se scrie pe 7 biţi – (v. tabelul cu comenzi), iar bitul al 8-lea, c.m.s (DB7) este blocat in 1.

=> pornind de la ***harta memoriei (cu adresele pe 7 biti)*** putem crea adresele ce trebuie trimise la busul de date pentru a seta direct ***pe 8 biţi*** adresa necesară.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
80	81	82	83	84	85	86	87	C0	C1	C2	C3	C4	C5	C6	C7

Adresele fizice de la **07h** la **40h** sunt locaţii de memorie liberă (pentru a fi folosită în alte scopuri).

Constatăm că pentru a afişa pe toate cele 16 caractere,

trebuie afişate **primele 8 începând cu adresa 80h** şi **următoarele 8 începând cu adresa C0h** (datorita DB7 blocat in 1).

C Code of the program :

```
#include "stdlib.h"
#include "conio.h"
#include "time.h"
#include "stdio.h"
#include "windows.h"
#include "pt_ioctl.c" ;// driver necesar la deschiderea/ închiderea porturilor PC
int port1=0x0378; // RD aflat la adresa  de bază a portului parallel LPT1
int port2=0x037a; // RC aflat la adresa  de bază+2 a portului parallel LPT1
int val[]={0x38,0x0f,0x01}; // comenzi date LCD-ului
int prim[]={'*','M','e','s','a','j',' ','d'}; // 8 caractere
int sec[]={'e',' ','t','e','s','t','!','*'};     // 8 caractere
; pt LCD 16 x 2 :
; // prim[]={'*','M','e','s','a','j',' ','d','a','a','a','a','a','a','a','a'}; // 16 caractere
; // sec[]={'e',' ','t','e','s','t','!','*','b','b','b','b','b','b','b','b'}; // 16 caractere
int dly=1, i;
char text[]="Se va scrie pe LCD textul:";
void init_lcd(void);
void scrie_lcd(void);
int main(void)
{
unsigned char value;
OpenPortTalk();

system("cls");               // curăță ecran
printf("\n%s\n\n",text);   // afișează mesaj pe monitorul PC
; // pt preluarea mesajului de la tastatură
; for(i=0;i<8;i++) putch(prim[i]);
; for(i=0;i<8;i++) putch(sec[i]);
puts("\n...apasati orice tasta pt. stergerea LCD-ului\n\n");
init_lcd(); // 3 czi din tabel
scrie_lcd();
getch();
outportb(port2,254); // (RC)<-254=FEh=1111 1110b
Sleep(dly);
outportb(port1,1); // (RD)<-1=01h=0000 0001b
Sleep(dly);
outportb(port2,1); // (RC)<-1=01h=0000 0001b
Sleep(dly);
printf("...gata, s-a sters\n\n apasati o tasta pt. terminarea procesului.!"); // afișează mesaj pe monitorul PC
getch();
return 0;

ClosePortTalk();
}
```

```
void init_lcd()
{ for(int i=0;i<3;i++)
        { outportb(port2,254); // (RC)<-254=FEh=1111 1110b
          Sleep(dly);
          outportb(port1,val[i]);
                        // (RD)<-38h=0011 1000b
                        // (RD)<-0Fh=0000 1111b
                        // (RD)<-01h=0000 0001b
          Sleep(dly);
          outportb(port2,1); // (RC)<-1=01h=0000 0001b
          Sleep(dly);
        }
}

void scrie_lcd()
{       outportb(port2,254); // (RC)<-254=FEh=1111 1110b
        Sleep(dly);
        outportb(port1,0x80); // (RD)<-80h=1000 0000b
        Sleep(dly);
        outportb(port2,1); // (RC)<-1=01h=0000 0001b
        Sleep(dly);
        outportb(port2,0); // (RC)<-0=00h=0000 0000b
        Sleep(dly);
        for(int i=0;i<16;i++)
        {       outportb(port1,prim[i]); // (RD)<-prim[i]
                Sleep(dly);
                outportb(port2,1); // (RC)<-1=01h=0000 0001b
                Sleep(dly);
                outportb(port2,0); // (RC)<-0=00h=0000 0000b
                Sleep(dly);
        }
        outportb(port2,254); // (RC)<-254=FEh=1111 1110b
        Sleep(dly);
        outportb(port1,0xc0); // (RD)<-C0h=1100 0000b
        Sleep(dly);
        outportb(port2,1); // (RC)<-1=01h=0000 0001b
        Sleep(dly);
        outportb(port2,0); // (RC)<-0=00h=0000 0000b
        Sleep(dly);
        for(int j=0;j<16;j++)
        {       outportb(port1,sec[j]); // (RD)<-sec [j]
                Sleep(dly);
                outportb(port2,1); // (RC)<-1=01h=0000 0001b
                Sleep(dly);
                outportb(port2,0); // (RC)<-0=00h=0000 0000b
                Sleep(dly);
        }
}
```

DDRAM memory map: with blue color it is mentioned the LCD type

LCD 16x1 = LCD 16 characters x 1 Line

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
00	01	02	03	04	05	06	07	40	41	42	43	44	45	46	47

LCD 16x2 = LCD 16 characters x 2 Lines

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
00	01	02	03												0Fh
40	41	42	43												4Fh

RC1604A-YHY-ESX = LCD 16 characters x 4 Lines

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
00	01	02	03												0Fh
40	41	42	43												4Fh
10	11	12	13												1Fh
50	51	52	53												5Fh

DEM20485SYH-LY = LCD 20x4 characters, 5x8 dots

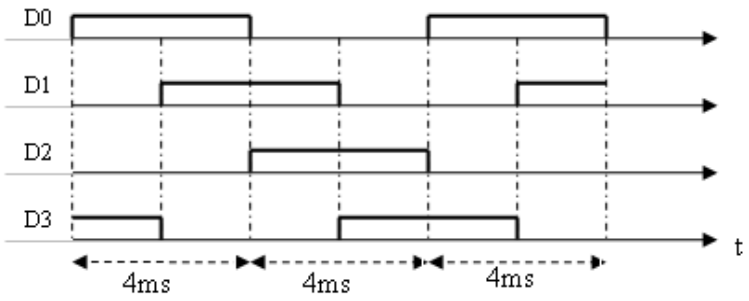
1	2	3	4	5	6	7	8	9	10	11	. . .	20
00	01	02	03									13h
40	41	42	43									53h
14	11	12	13									27h
54	51	52	53									67h

Aplicatia 2. Pe liniile de date (D0-D3) ale portului paralel standard (SPP) se doreste generarea urm. forme de semnal, folosind intreruperi.

(1.5p) a. Programati Ch0 al timerului (40h) a.i. sa furnizeze frecventa dorita generarii formei de semnal;

(1.5p) b. Scrieti secventa de redirectare a intreruperii.

(2p) c. Scrieti procedura NEW_INT8, folosita la generarea esantioanelor pe port. Scrieti secventa de cod din programul principal (sau adaptati si rutina de la b)) a.i. sa se geereze un nr de 200 periade ale semnalului. Definiti toate variabilele semnalului.



a. $n_0 = \frac{Fclk_0}{Fout_0} = \frac{Fclk}{F}$; ;

Din figura se observa ca un esantion dureaza jumatate din 4ms => Tesantion = 2 ms

F=Fesantion=1/Tesantion =1/2ms =0.5 kHz

n0=1,2MHz / (1/2ms) = 1,2 MHz / 0.5 KHz = 2,4 K = 2400

Programarea canalului 0 al circuitului timer:

```
mov al, 36h             ;cuvant de mod (selectare canal 0, citire/scriere LS apoi MS, mod 3)
out 43h, al
mov ax, 2400            ;constanta de divizare
out 40h,al
xor al,al
mov al, ah
out 40h,al
```

b. Redirectarea intreruperii IRQ0 (tipul 8) a timerului, pe rutina utilizator numita *new_int8:*

```
mov ax, 3508h           ;salvez adresa vechii intreruperi IRQ0 (tip 08) in ES:BX
int 21h                         ;se apeleaza intreruperea DOS
mov irq0_old, bx        ;salvez adr offset a vechii intr
mov irq0_old+2, es              ;si adresa de segment
push ds
mov dx, offset new_int8 ;se incarca in TVI noua adresa a rutinei de tratare a intr IRQ0
mov ax, seg new_int8
mov ds, ax
mov ax, 2508h
int 21h
pop ds
……  / program  principal
push ds                 ;reactivarea vechii intr IRQ0
mov ax, 2508h
lds dx, dword ptr irq0_old
int 21
pop ds
```

c. Pe portul paralel se transmite un semnal periodic format asa cum se vede pe figura din 4 esantioane. Fiecare esantion e scris pe 8 biti, din care 4 (D0-3) se vad pe figura. Astfel, ceilalti 4 biti pot fi considerati la masa D4,5,6,7=0.

Semnalul se va obtine:
primul esantion: D7 …0 = 0000 1001
al doilea esantion: D7 …0 = 0000 0011
al treilea esantion: D7 …0 = 0000 0110
al patrulea esantion: D7 …0 = 0000 1100
=> semnal db 09h, 03h, 06h, 0C h ; cele 4 esantioane ale semnalului

```
        .data
        semnal    db    X0, X1, …. X3
    .code
    ……..
        mov si, offset semnal
        mov dx, 378h            ; se selecteaza registru de date a Portului Paralel
    mov nrEs,0
new_int8   proc   near          ;noua rutina de intrerupere
        push ds
        push dx
        push bx
        mov al, [si]            ; selectare valori sir
        out dx,al               ;se trimit pe rand valorile corespunzatoare generarii formei de unda, pe liniile de date ale PP
        inc si
        inc nrEs
        cmp si, 4               ; se verifica daca s-a ajuns la sfarsitul sirului
        jl et1
        mov si, offset semnal   ;
et1:    mov al, 20h
        out 20h, al             ;EOI  -> PIC e anuntat ca s-a terminat RTI
        pop bx
        pop dx
```

```
            pop ds
            iret
new_int8  endp
et:
    cmp nrEs,800     ; 200 per?
     Jz quit
Jmp et

Quit: gata program
Call NEW_INT8
```