5.Teste de calitate

a.Aparatura necesară

-2 calculatoare PC AT diferite

b.Elemente teoretice

CHECKIT

CHECKIT este un program de testare și diagnoză pentru calculatoare compatibile IBM-PC. Acesta poate să genereze rapoarte privind configurația calculatorulu, testează unitățile hardware ale calculatorului și afișează informații privind puterea de procesare a unității centrale.

Checkit este organizat în următoarele opțiuni:

SYSINFO

-afișează un raport ce descrie configurația hard și soft a sistemului.

-afișează un raport privind folosirea întreruperilor hard (IRQ) precum și o listă cu canalele DMA. Checkit poate să identifice atribuirea întreruperilor numai după ce driverul soft a fost instalat și activat.

-afișează setările curente din CMOS. Acestea pot fi tipărite sau salvate. Programul nu permite modificarea lor.

-afișează lista driverelor soft instalate împreună cu adresele din memorie, atributele și versiunea DOS

TESTS

-test de memorie RAM

-teste de hard disk; testarea se face în trei moduri

-citire liniară (începând cu cilindrul 0 se continuă secvențial pâna la ultimul cilindru)

-citire în fluture (se citește câte un cilindru din exterior apoi un cilindru din interior în ordine crescătoare respectiv descrescătoare)

-citire aleatoare

-testare unitate floppy disk. În cadrul acestui test se testează atât unitatea cât și discul

-testare CPU,NPU,DMA, întreruperi

-testarea ceasului de timp real

-testare porturi seriale

-testare porturi paralele

-testare imprimantă

-testarea subsistemului video

-RAM video

-testare în mod text

-testare în mod grafic

-testarea echipamentelor de introducere

-tastatură

-mouse

-joystick

-ș.a.

Programul permite rularea implicită a unor teste desemnate într-un Batch.

BENCHMARKS

Se măsoară performanțele sistemului pentru:

-sistemul de bază

-viteza CPU (DHRYSTONES)

-viteza video

-viteza matematică (WHETSTONES)

-hard disk -timpul de căutare mediu

-timpul de pășire (deplasare pistă cu pistă)

-viteza de transfer

TOOLS

-identificare de CHIP de RAM defect

-setare ceas de timp real

-formatare "low level "a hard disk-ului

SETUP

-permite selectarea modului de afișare color sau B/W
-permite direcționarea rapoartelor către ecran, fișier pe disc sau imprimantă
-descrierea memoriei pentru a fi folosită la identificarea CHIP-ului de RAM defect

QAPLUS

Acest program de testare și diagnoză dă posibilitatea utilizatorului să obțină informații despre configurația hard a sistemului, drivere soft instalate, performanțele sistemului și ale hard disk-ului precum și alocarea întreruperilor și a canalelor DMA. Aceste informații se pot obține selectând una din opțiunile meniului orizontal SYSINFO.

Din meniul orizontal TESTING se pot rula teste pentru toate unitățile calculatorului. În meniul vertical RUN TESTS din testări, se pot selecta testele dorite a fi rulate. Acestea sunt organizate în grupe de test astfel: componente de bază, adaptor video, unitate floppy disk, unitate hard disk, porturi seriale, porturi paralele, imprimantă și RAM. În mod implicit toate testele sunt selectate. Modul de selectare/deselectare este indicat de program.În cadrul grupelor se pot testa următoarele:

Componente pe placa de bază:

-CPU - unitatea centrală
-NPU - coprocesorul matematic
-INT - întreruperi
-TIMER- circuit de temporizare
-DMA - accesul direct la memorie
-CCLK- ceasul CMOS

Adaptor video:

-MDA - monochrome display adapter
-CGA - color graphics adapter
-EGA - extended graphics adapter
-VGA - video graphics array

Unitate floppy disc: testul este nedistructiv și se lucrează cu un disc flexibil gol formatată la capacitatea unității.

Unitatea hard disk: testul este nedistructiv; se testeaza capacitatea de a citi date de pe hard disk utilizând diferite metode de căutare.

Porturi seriale: Pentru ca testul să fie edificator este indicat să se ruleze cu semnale rebuclate în conector.

Porturi paralele: se recomandă aceleași condiții ca și la porturile seriale.

Imprimante: în cadrul acestui test se trimite la imprimantă un mesaj de test care este afișat și pe ecran, urmând să se facă comparație.

RAM

Testează memoria RAM (atât cea de bază cât și cea extinsă și expandată) folosind diferite metode de scrire citire. Se testează de asemenea sensibilitatea la zgomot.

Înainte de a proceda la rularea testelor se fac diferite setări care se referă la :

-numărul de cicluri în cadrul unui test (1,10,100,1000 sau continuu)
-pauză (da/nu) la sfârșitul unui test sau la înregistrarea unei erori.
-modul de comunicare a erorilor (ecran, fisier, imprimantă)

Selectarea perifericelor. În cadrul acestei opțiuni se comunică configurația porturilor seriale si a porturilor paralele. Se specifică dispozitivul atașat fiecărui port. Acestea pot fi buclă, imprimantă, modem, mouse, alt dipozitiv sau nici un dispozitiv. Pentru porturile seriale se stabilesc parametrii de comunicație.

În cadrul opțiunii INTERACT se pot testa :

-tastatura
-mouse-ul
-joystick-ul
-difuzorul

Depanare COM-uri: în cadrul acestei opțiuni se testează portul serial sau modemul prin vizualizarea la nivel de registru a efectului unor anume comenzi (comenzi trimise de programe de aplicații către COM)

Depanare RAM ; în cadrul acestei opțiuni se localizează CHIP-ul de RAM defect în cazul în care testul de RAM a comunicat eroare și după ce a fost definită amplasarea memoriei.

În meniul UTILITY există doua programe utilitare furnizate de firmă și există posibilitatea ca utilizatorul să poată rula programe proprii sub numele program1.exe și program2.exe din meniul principal. Utilitarele de firmă sunt pentru hard disk; unul pentru formatare de nivel low celălalt pentru parcarea capetelor de scriere citire de la hard disk.

Opțiunea SETUP permite:

-setarea calendarului și ceasului CMOS.

-modificarea informației existente în CMOS, informație care folosește la determinarea configurației hard la boot-are.

-setarea sunetului în mod on/off la eroare.

-setarea de culoare în mod on/off.

Utilizarea programului SUPERPCK

SUPERPCK este un program de definire a unei zone de memorie RAM a sistemului ca zonă de memorie CACHE pentru hard disk.

Instalarea programului în memoria extinsă se face cu SUPERPCK/A* . Programul devine rezident și interceptează orice acces la hard disk. Programul poate fi dezactivat cu SUPERPCK/U . După un anumit număr de operații cu discul, cu SUPERPCK/M se poate solicita un raport al acceselor fizice în raport cu cele logice. Fără CACHE numărul de accese fizice este egal cu numărul de accese logice. Cu CACHE numărul de accese fizice va fi mai mic decât numărul de accese logice.

Verificarea integrității datelor pe hard disc

Este nevoie periodic să se verifice datele pe hard disk pentru a se descoperi sectoarele defecte sau cele pentru care este nevoie de mai multe citiri pentru recuperarea informației și care pot deveni defecte.

COMPRESS

Este un program care face parte din PC SHELL. COMPRESS are următoarele opțiuni :

F4 - se face organizarea informației pe disc astfel încât fisierul să ocupe zone continue. Dacă un fișier este fragmentat, timpul necesitat la citirea lui este mai mare.

F7 - se face analiza discului și analiza suprafeței. Se citesc sectoarelepe rând, iar dacă unul dintre ele prezintă erori , conținutul lui se mută într-un sector liber. Sectorul defect se marchează pentru a nu mai fi folosit.

CTRL (nume drive) - se modifică unitatea care se verifică. Este posibilă și verificarea discurilor flexibile.

ESC - ieșire din program.

Uneori acest program solicită rularea utilitarului CHKDSK.

Utilitarul CHKDSK

Este utilizat pentru verificarea stării discului. Poate lucra și cu hard disk cu capacitate mărită prin SUPERSTOR. CHKDSK dă un raport :

-spațiul total

-spațiul ocupat de fișiere

-spațiul aflat în sectoare defecte

CHKDSK/F verifică lanțurile de alocare greșite (sectoarele nealocate sau alocate la două fișiere) pe care S.O. nu le poate utiliza. Aceste sectoare sunt alocate unor fișiere cu extensia .CHK pentru ca ele să poată fi șterse ulterior. Sub MSDOS este mai eficientă rularea programului SCANDISK.

Testul de disc TD

Realizat de Gh. Nicodin elimină două dezavantaje ale programului COMPRESS. TD salvează informația din fiecare sector, scrie un șir de octeți în sector și verifică dacă scrierea a fost corectă. Dacă da, reface informația inițială. Testarea prin scrire și citire este mai edificatoare decît cea exclusiv prin citire cum era la COMPRESS. Fată de COMPRESS care repetă citirile TD face o singură citire, deci acest test va găsi mai multe sectoare aflate la limita funcționării corecte. Dezavantajul major este durata mult mai mare a testului.

Teste sub WINDOWS

WINSPEED (Windows Performance Analysis)

Măsoară performanțele calculatorului din punct de vedere WINDOWS. Aceste performanțe nu depind numai de performanțele plăcii de bază, a hard discului și a plăcii VGA ci și de modul de configurare al WINDOWS. Testul are 3 părți:

CPU WINDMARK, ca termen de comparație fiind trecute: 286/12, 386/20, 386/25, 386/33, 486/25, 486/33.

VIDEO PERFORMANCE, care împarte plăcile VGA în POOR, OK, GOOD, GREAT

DISK PERFORMANCE, care împarte hard discurile + cuplor în aceleași categorii.

WINTACH , creat în 1992 de Texas Instruments

În acest test este posibilă definirea mai multor rezoluții:

A 640x480

B 800x600

C 1024x768

D 1280x1024

Testul dă indici de viteză relativ la sistemul 386DX/20 cu VGA standard și cu plăci de tip TIGA VGA, ATI ULTRA, ET 4000. Viteza este funcție și de numărul de culori cu care este instalat WINDOWS. Indicele rezultat este urmat de o literă (rezoluția aleasă) și o cifră care reprezintă numărul de culori:

1- 2 culori

4- 16 culori

8- 256 culori

16- direct color

32- true color

Rezultatele testului (indicii de viteză) sunt dați pentru următoarele tipuri de aplicații:

1.Procesare de texte

2.Proiectare, desenare (vectorial)

3.Calcul tabelar

4.Desenare (bitmap)

5.Indice global.

Rezultatele testelor pentru câteva calculatoare tipice sunt date în tabelul următor:

TEST	386DX/33 TRIDENT SVGA 1M HDD WD	486SLC/33 TRIDENT SVGA 1M HDD WD	486DX/33 TSENG LAB SVGA OPTI 1M HDD WD	586/133 TRIO SVGA 1M PCI HDD WD
WINDMARK	132	241	250	999
VIDEO	046	085	241	999
DISK	119	109	434	832
WORD PROCESSING	2.35C8-1.62A8-2.15B8	1.83A4	8.75B16	31.9A8-40.8B8
CAD	4.19C8-3.12A8-3.06B8	1.69A4	11.37B16	101.26A8-103.4B8
SPREADSHEET	2.35C8-1.82A8-2.06B8	1.41A4	7.81B16	44.8A8-50.6B8
PAINT	2.35C8-1.88A8-2.09B8	1.74A4	11.18B16	47.3A8-57.2B8
GLOBAL	2.81C8-2.11A8-2.38B8	1.67A4	9.78B16	56.31A8-63B8

c.Mersul lucrării

-se rulează programele de test menționate și se notează rezultatele lor

-se realizează o comparație între cele 2 calculatoare d.p.d.v. al performanțelor

6. Memoria

a. Aparatura necesară

-calculator PC AT 386-586

-calculator ROBOTRON A5120 cu sistemul de operare CPM
-osciloscop

b. Elemente teoretice

1.Fluxul de date în sistemele de calcul

În schema bloc a unui sistem de calcul se poate considera că există o ierarhie de memorii (ierarhizare în funcție de timpul de răspuns), figura 6.1.

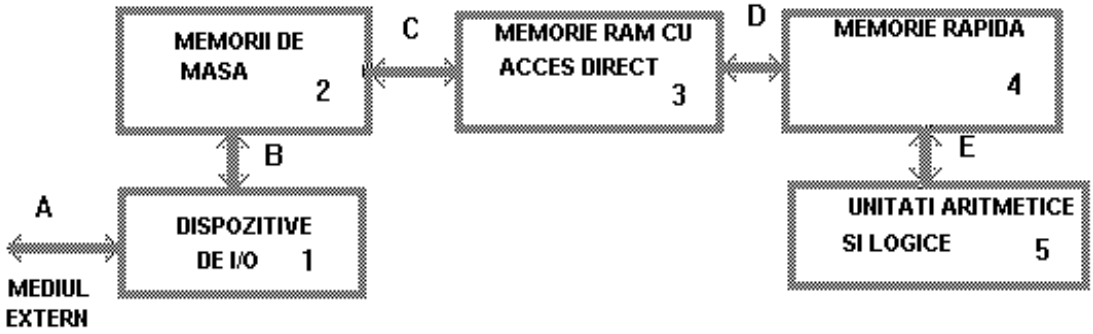

Figura 6.1.

Memoriile au viteza, capacitatea și prețul diferit Aceste caracteristici sunt trecute în tabelul următor:

Tip	Memorie[1]	Timp de acces	Capacitate	Preț (USD/ Mocte t)
Memorii de masă	hard disc (neamovibil)	10ms	1-10GBy	0.2
	disc flexibil (amovibil)	25ms	1.44MBy/disc	1
	CD (amovibil)	15ms	500MBy/disc (RO)	0.02
	unitate de bandă (amovibil)	s	350MBy-1GBy/casetă	0.01
Memorie	Memorie RAM cu acces direct	60ns	1-64MBy	5
	SRAM cache, memorie tampon rapidă	10ns	128-512KBy	30

Viteza unui sistem de calcul depinde de 2 componente:
-viteza cu care procrsorul execută operațiile;
-viteza cu care circulă datele în sistem (viteza de I/O, viteza de acces la memorie)
Dacă procesorul este forțat să rămână inactiv perioade lungi de timp deoarece sistemul de I/O nu poate transfera datele sistemul este *limitat I/O*. Ideal este ca cele două viteze să fie comparabile.
În figură se poate vedea o diagramă a fluxului de date. Pentru mărirea eficienței 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 rețeaua) duce la optimizarea încărcării sistemului.
3.Eficiența este maximă când capacitatea blocurilor 1,2,3 de a furniza date este egală cu capacitatea blocului 5 de a le prelucra.

[1] Caracteristicile și prețurile sunt la nivelul anului 1996 și în viitor trebuie privite cu rezervă, domeniul fiind foarte dinamic. Este posibil ca proporțiile să se păstreze mai mult timp.

4.Memoria rapidă conține date și secvențe de program. Lucrul între blocurile 4 și 5 este foarte rapid. Dar există momente în care este nevoie de informația din memoria de bază (mai ales când apar instrucțiuni de salt). În aceste momente este nevoie să se schimbe întregul conținut al memoriei rapide.
5.Dacă dimensiunea programelor depășește capacitatea de memorare a blocului 3 (uzual la aplicații sub WINDOWS) se pot transfera porțiuni între memoria RAM și memoria de masă. Iluzia unei memorii RAM de dimensiuni mai mari (cu viteză de acces mai mică însă) pe baza informației aflate într-un echipament periferic de stocare se numește *memorie virtuală.*

Atât memoria rapidă cât și memoria virtuală se bazează pe utilizarea unei memorii *asociative*. La această memorie căutarea informației se face după conținut și nu prin adresare.

Observație: diferența între memorie și un EP de stocare din punct de vedere funcțional poate fi foarte mică.

Bibliografie selectivă

(1) P. Ogruțan, "Interfețe și echipamente periferice", curs, Universitatea Transilvania Brașov, 1994

(2) Axel Kloth, "Bussysteme des PC", FRANZIS Verlag, 1993

(3)L.C. Eggebrecht,"Interfacing to the IBM Personal Computer", Macmillan Computer Publishing, Carmel Indiana, 1991

(4) P. Ogrutan, C. Gerigan, "Some aspects regarding serial asynchronous transmission", Proccedings of the 4 International Conference On Optimization of Electric and Electronic Equipments, Brasov, 1994

(5) **********, "INTEL 80286 Family", "INTEL Interfaces", Data Book, November 1985

(6) ********** , "8080 Microcomputer Peripherals User's Manual",

ANEXE

1. Circuitul interfață paralelă programabilă INTEL 8255, figura A1:

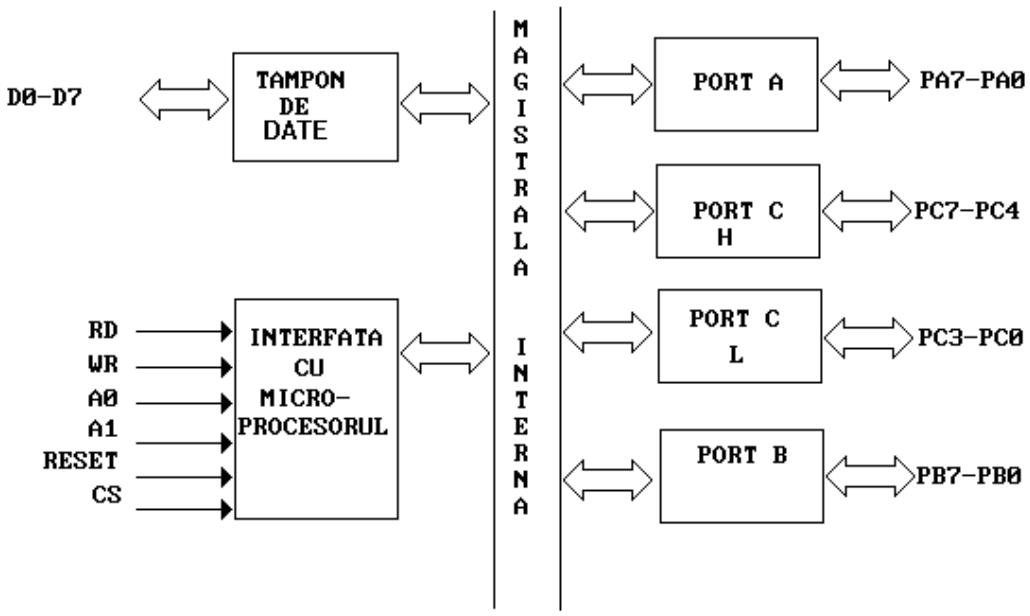

Figura A1

Semnificația pinilor:
RD -Se execută un ciclu de citire de la port sau de la memorie
WR -Se execută un ciclu de scriere la port sau la memorie
A0 -linie de adresă ,cel mai puțin semnificativ bit
A1 -linie de adresă
RESET -comanda RESET către 8255 anulează conținutul tuturor registrelor
CS -selecție circuit /CS=0 (activ pe 0) 8255 selectat
D0...D7 -bus de date , 8 linii bidirecționale ,3-state
PA7...PA0 -port A , 8 linii bidirecționale
PB7...PB0 -port B , 8 linii bidirecționale
PC7...PC4, PC3...PC0 -port C , 8 linii bidirecționale grupate în 2. În funcție de A0 și A1, dacă CS este activ se selectează porturile sau registrul de comandă.

A0	**A1**	
0	0	port A selectat
0	1	port B selectat
1	0	port C selectat
1	1	registrul cuvîntului de comandă

Modul de lucru al 8255:
MODUL 0 -mod de bază de intrare/ieșire, asigură:
A și B porturi de 8 biți
C, 2 porturi de 4 biți ,cu posibilitate de poziționare individuală în 1 sau 0.
MODUL 1 -mod de lucru cu posibilități de strobare pentru intrare/ieșire. Se pot folosi 2 porturi formate din portul A respectiv B de date, cu semnale de comandă de 3 biți din portul C.
Semnalele de comandă în acest mod vor fi:
a).pentru intrare
-INTEA comandat prin PC4, INTEB comandat prin PC2 (validare întreruperi)
-STBA strob de intrare (PC4), încarcă data în registrul de intrare
-IBFA Input Buffer Full (PC5), registrul de intrare încărcat. Este activat cĂnd perifericul generează STBA și este dezactivat cĂnd procesorul citește date (RD).
-INTRA cerere de întrerupere către microprocesor (PC3), cĂnd datele sunt în registrul de intrare (cĂnd bufferul este plin). Este dezactivat de semnalul RD . Cererea de întrerupere este validată de un bistabil intern INTE de validare întreruperi.
-PC6, PC7 pot fi programate ca intrări/ieșiri de un bit.
b).petru ieșire
-INTEA comandat prin PC6, INTEB comandat prin PC2
-OBFA Output Buffer Full (PC7) , registru de ieșire plin, activat de semnalul WR și dezactivat de periferic prin semnalul ACK.
-ACK acceptare date de către periferic (PC6), este activat cĂnd datele au fost preluate de periferic .
-INTRA cerere de întrerupere către microprocesor (PC 3), anunță că data a fost preluată. Validat de bistabilul intern INTE. Este activată de ACK și dezactivată de WR.
-PC4, PC5 pot fi programate ca intrări/ieșiri de un bit.
Corespunzător portului B semnalele sunt următoarele:
STBB PC2 OBFB PC1
IBFB PC1 ACKB PC2 INTRB PC0 INTRB PC0
Observație: Se poate folosi un port în modul 0 și unul în modul 1.

MODUL 2 -portul A este folosit ca port bidirecțional, asistat de liniile PC3...PC7 din portul C. Semnalele de comandă sunt următoarele:
-INTRA (PC3) cerere de întrerupere, emisă către microprocesor pentru intrări și ieșiri
-OBFA (PC7) în cadrul unei operații de ieșire
-ACKA (PC8) în cadrul unei operații de ieșire
-STBA (PC4) în cadrul unei operații de intrare
-IBFA (PC5) în cadrul unei operații de intrare
țntreruperile generate de o operație de ieșire sunt validate de INTE1, iar cele generate de o operație de intrare de INTE2.

INTE1 -controlat de PC6
INTE2 -controlat de PC4

Programarea circuitului INTEL 8255

Comanda modului de lucru:
Grup A: PA și PC4...PC7
Grup B: PB și PC0...PC3
B7=1

B6	B5	
0	0	grup A mod 0
0	1	grup A mod 1
1	x	grup A mod 2

B4 B4=0 PA ieșiri, B4=1 PA intrări
B3 B3=0 PC4...PC7 ieșiri, B3=1 PC4...PC7 intrări
B2 B2=0 mod 0 pentru grup B, B2=1 mod 1 pentru grup B
B1 B1=0 PB ieșiri, B1=1 PB intrări
B0 B0=0 PC0...PC3 ieșiri, B0=1 PC0...PC3 intrări.
Dacă s-a selectat modul 1 de lucru , este necesar să fie accesibil fiecare bit al PC pentru poziționare.
D7=0,D6=X,D5=X,D4=X
D3 D2 D1 reprezintă prin decodificare adresa bitului în octet.
D0=1 se înscrie 1, D0=0 se înscrie 0
țn acest fel se pot poziționa bistabilii de validare întreruperi INTE
Citirea stării 8255:
Citirea stării în mod 1 (prin citirea portului C)
D7,D6,D5,D4,D3 stare grup A
D2,D1,D0 stare grup B

Grup A (port intrare)	Grup A (port ieșire)
D7 I/E, D6 I/E	D7/OBFA
D5 IBFA	D6 INTEA
D4 INTEA	D5 I/E,D4 I/E
D3 INTRA	D3 INTRA

Grup B (port intrare)	Grup B (port ieșire)
D2 INTEB	D2 INTEB
D1 IBFB	D1 /OBFB
D0 INTRB	D0 INTRB

Citirea stării în mod 2 (prin citirea portului C)
D7 /OBFA
D6 INTE1
D5 IBFA
D4 INTE2 D3 INTRA
D2=X,D1=X,D0=X

2. Interfața serială programabilă INTEL 8251

(Receptor/transmițător Universal Sincron Asincron USART), figura A2:

.

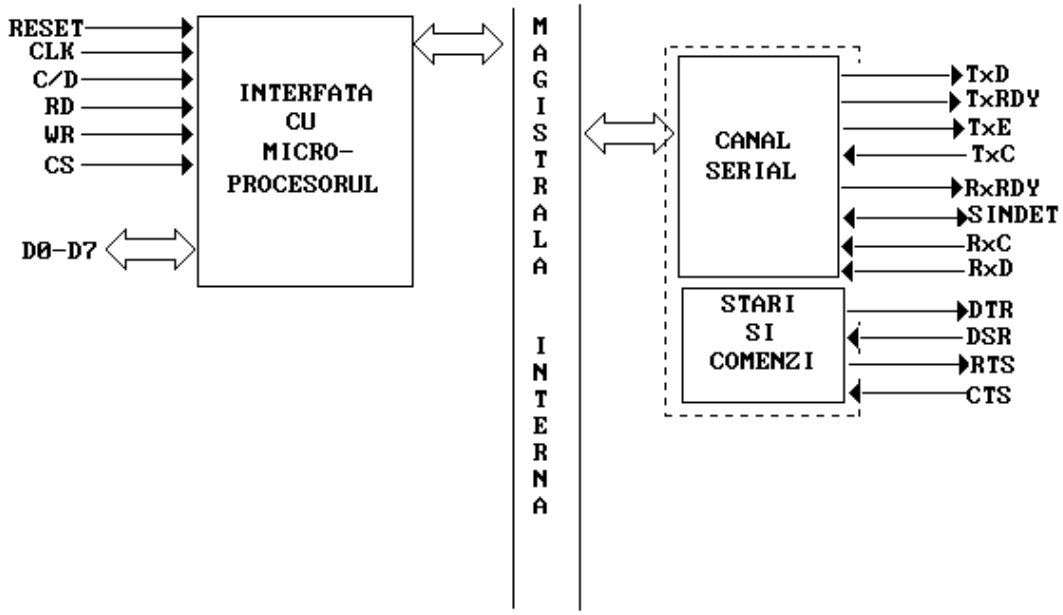

Figura A2

Semnificația semnalelor:

D0...D7 8 linii de date bidirecționale
RESET,CLK,C/D,RD,WR,CS ca și la I8255
TxRDY transmițător pregătit pentru a primii date pe magistrală
TxE transmițător gol, nu are date de transmis
RxRDY un caracter este pregătit pentru a fi transmis pe magistrala D0...D7.
SYNDET forțare sau detecție sincronă de date
DSR,DTR,CTS,RTS,TxC,RxC cu semnificația dată la curs

Programarea circuitului 8251

Primul cuvînt de comandă pe poziția B1 B0 comandă modul de lucru astfel: B0=0 mod sincron, altă combinație mod asincron.

Mod asincron

B7 B6 biții de stop
0 0 invalid
0 1 1 bit STOP
1 0 1 1/2 bit STOP
1 1 2 bit STOP

B5 B4 paritate
x 0 dezactivat
0 1 paritate pară
1 1 paritate impară

B3 B2 lungime caracter
0 0 5 biți
0 1 6 biți
1 0 7 biți
1 1 8 biți

B1 B0
0 0 invalid
0 1 rata transmisie=ceas/64
1 0 rata transmisie=ceas/16
1 1 rata transmisie=ceas

Modul sincron

După acest prim cuvânt de comandă trebuie să urmeze 2 octeți care reprezintă cuvântul de sincronizare (al doilea octet pentru BISYNC numai)

B1 B0
0 0 mod sincron
B7 B6 mod de sincronizare
x 0 SYN intern
x 1 SYN extern
0 x două caractere SYN
1 x un singur caracter SYN
B5 B4 control de paritate
x 0 dezactivat
0 1 paritate pară
1 0 paritate impară B3 B2 lungime caracter ca la mod asincron
Al doilea cuvânt de comandă
B7 intrarea în modul HUNT
B6 reset intern
B5 RTS se comandă semnalul corespunzător
B4 anularea tuturor erorilor din registrul de stare
B3 transmisie caracter BREAK dacă B3=1 -TxD este forțat în SPACE
B2 recepție activată RxE
B1 DTR se comandă semnalul corespunzător
B0 transmisie activată TxEN

Cuvântul de stare
B0 TxRDY starea semnalului TxRDY
B1 RxRDY
B2 TxE
B3 PE eroare de paritate
B4 OE eroare de sistem
B5 FE eroare de cadrare (asincron) când nu s-a detectat un bit de STOP
B6 SYNDET
B7 DSR

3.Circuitul timer I8253

Circuitul INTEL 8253 este un circuit timer specializat construit pentru a fi utilizat ca generator de bază de timp programabil în sistemele PC.

Caracteristici principale :
- trei numărătoare independente de 16 biti
- frecvența maximă de lucru 2 Mhz
- moduri programabile de lucru ale numărătoarelor
- moduri de numărare binar sau BCD
- un singur pin de alimentare +5V
- capsulă DIL 24 pini

Circuitul este versatil putând fi folosit într-o gamă foarte largă de aplicatii, cum ar fi :
- numărător programabil
- generator de impulsuri
- ceas de timp real
- controler pentru motoare

Schema bloc a circuitului este dată în figura A3:

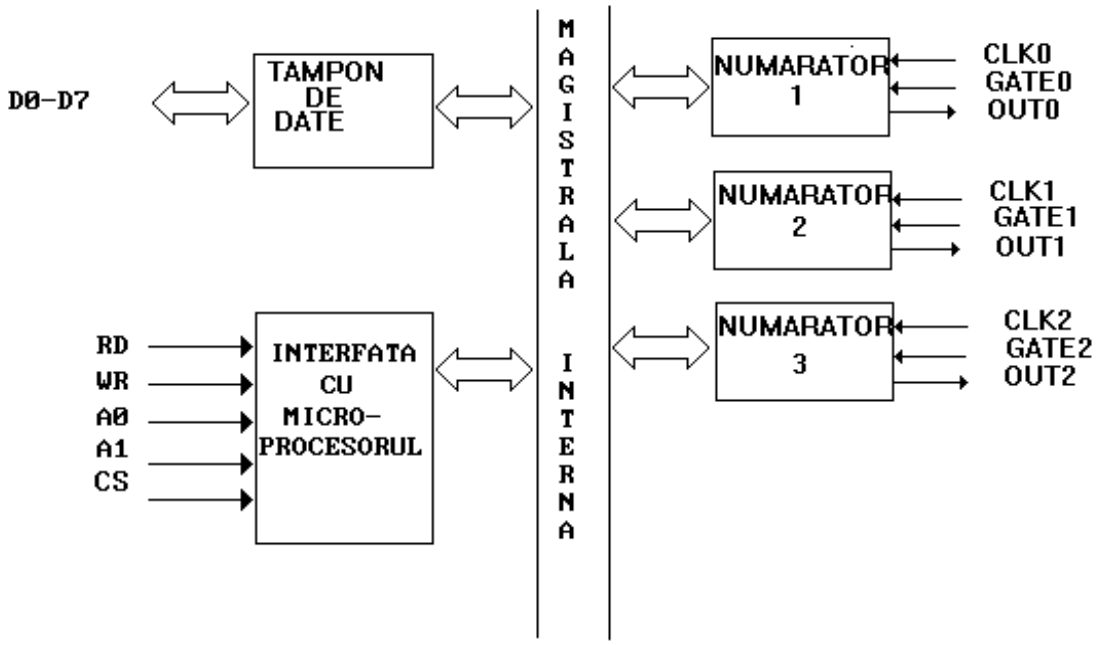

Figura A3

Semnificația semnalelor este ca la I8255 și I8251. În funcție de semnalele la intrare se pot face următoarele operații:

CS	RD	WR	A1	A0	
0	1	0	0	0	Load counter 0
0	1	0	0	1	Load counter 1
0	1	0	1	0	Load counter 2
0	1	0	1	1	Scriere cuvânt mod
0	0	1	0	0	Read counter 0
0	0	1	0	1	Read counter 1
0	0	1	1	0	Read counter 2
0	0	1	1	1	nimic 3 state
1	x	x	x	x	invalidare 3 state
0	1	1	1	x	nimic 3 state

I8253 conține trei canale, independent programabile în șase moduri:

Mod 0 - întrerupere la sfârșitul numărării
Mod 1 - impuls unic programabil
Mod 2 - generator de impulsuri
Mod 3 - generator de impulsuri dreptunghiulare
Mod 4 - strob declanșat software
Mod 5 - strob declanșat hardware

Semnificațiile pinului GATE

Mod / **Signal status**	'0' sau front descrescător	front crescător	'1'
0	invalidează numărarea	_____	validează numărarea
1	_____	1) inițializează numărătorul 2) resetează ieșirile după următorul clock	_____

2	1) invalidează numărarea 2) setează ieșirile în stare '1'	inițializează numărătorul	validează numărarea
3	1) invalidează numărarea 2) setează ieșirile în stare '1'	inițializează numărătorul	validează numărarea
4	invalidează numărarea	_____	validează numărarea
5	_____	inițializează numărătorul	_____

Pentru PC timer-ului I8253 i se alocă spațiul de adrese 040H - 05FH în felul următor:

040H - numărător canal 0 de 16 biți (tact pentru ceas sistem)
041H - numărător canal 1 de 16 biți (DRAM-refresh prin canal 0 DMA)
042H - numărător canal 2 de 16 biți (semnal pentru difuzor)

Programarea circuitului I8253

Programarea circuitului I8253 se face prin intermediul registrului de comandă, astfel:

D7	**D6**	**D5**	**D4**	**D3**	**D2**	**D1**	**D0**
SC1	SC0	RL1	RL0	M2	M1	M0	BCD

SC - Select counter

SC1	**SC0**	
0	0	Selecție counter 0
0	1	Selecție counter 1
1	0	Selecție counter 2
1	1	Ilegal

RL - Read / Load

RL1	**RL0**	
0	0	operație de memorare
0	1	Read/Load MSByte
1	0	Read/Load LSByte
1	1	Read/Load MSByte urmat de LSByte

M - Mod

M2	**M1**	**M0**	
0	0	0	mod 0
0	0	1	mod 1
x	1	0	mod 2
x	1	1	mod 3
1	0	0	mod 4
1	0	1	mod 5

BCD

0	numărător binar pe 16 biți
1	numărător BCD (4 decade)

4.Convertorul analog digital MMC 757

Convertorul analog digital este un convertor monolitic realizat în tehnologie CMOS, având rezoluția de 12 biți și ieșirile multiplexate pe 8 lini, în doi octeți.

Valorile limită absolute ale acestui circuit sunt date în tabelul următor:

Parametrul	Valoare min.	Valoare max.	Unitate de mãsurã
Tensiuni de alimentare VCC VDD	 -0.3 -0.3	 +7 +16.5	 V V
Tensiunea la orice terminal de ieºire	-0.3	+7	V
Tensiunea la orice terminal de intrare	-5	+18	V
Tensiunea la intrerea AIN	-5	+16.5	V
Tensiunea la intrările VHR,VHL	-0.3	+7	V
Gama temperaturilor de funcþionare	-25	+85	'C

Circuitul integrat MMC757 este un convertor analog digital de 12 biți cu ieșirile multiplexate pe 8 linii pentru a fi ușor interfațat cu magistrale de 8 biți specifice microprocesoarelor. Circuitul încorporează o referință internă compensată termic având valoarea tipică de 1.8 V. Poate fi utilizată și o referință externă, conectarea acesteia făcându-se la terminalele VRH și VRL ale convertorului D-A încorporat. Valoarea referinței externe poate fi între 0 și 6.5 V. Convertorul D-A este realizat cu o rețea rezistivă R-2R și comutatoare analogice MOS.

Principiul utilizat pentru conversia analog-digitală este cel al aproximației succesive. Comparatorul utilizat este alimentat la tensiunea de alimentare de 5 V prin intermediul unui terminal separat, VCCK pentru a se putea realiza o bună decuplare a zgomotelor. Consumul comparatorului este de circa 0.6 mA.

Partea digitală a circuitului cuprinde un generator de ceas intern, o logică de control, registrul de aproximație succesivă și blocul registrului și multiplexorului de ieșire cu logică tri-state. Schema bloc a circuitului integrat MMC 757 este dată în figura A4:

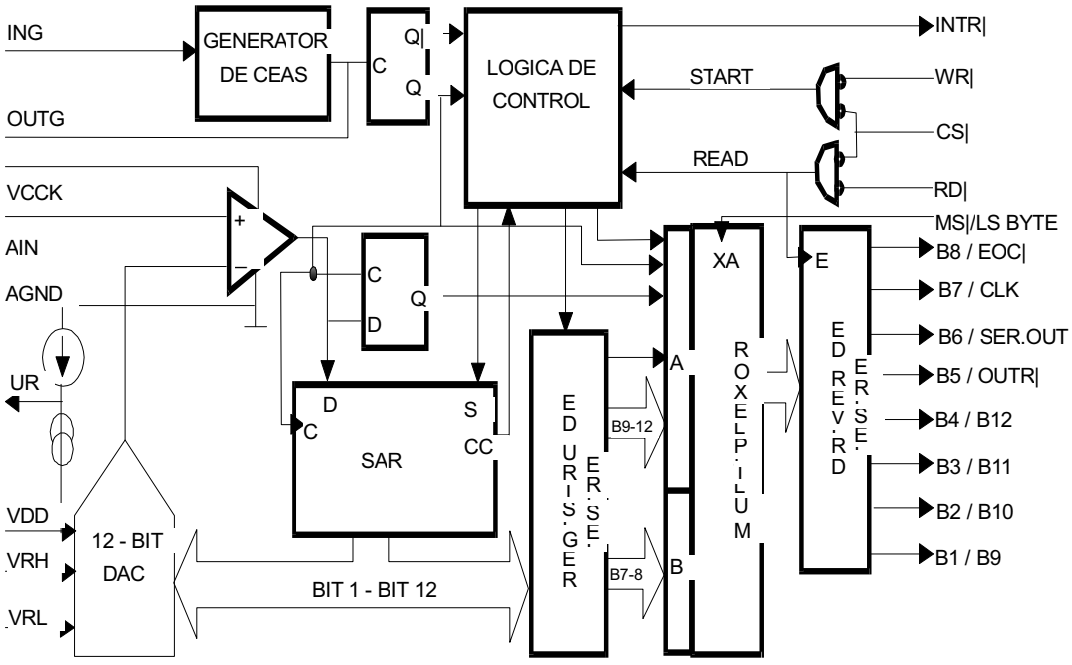

Figura A4

Circuitul utilizează două tensiuni de alimentare, VCC de 5 V +/-5%, și VDD, cu o gamă de variație foarte largă, între 4.75 V și 15.75 V. În lucrare tensiune VDD este de 5V de la pinul de 5V al magistralei ISA. Compatibilitatea intrărilor cu logica TTL este asigurată în toată gama tensiunilor de alimentare.

Pe durata impulsului de start (WR=CS=0) registrul de aproximație succesivă (SAR) este resetat. Conversia începe la tranziția cel puțin unuia dintre semnalele WR și CS în "1" logic. Comparatorul analogic compară ieșirea convertorului D/A cu tensiunea de intrare de la terminalul AIN și controlează registrul SAR. Testul biților începe cu MSB și ezultatul apare și la ieșirea serială. După 12 comparații, în SAR se obține codul binar de 12 biți. Acest cod este transferat în latch-ul de ieșire și convertorul este pregătit pentru o nouă conversie. Graficul conversiei e dat în fig. A5.

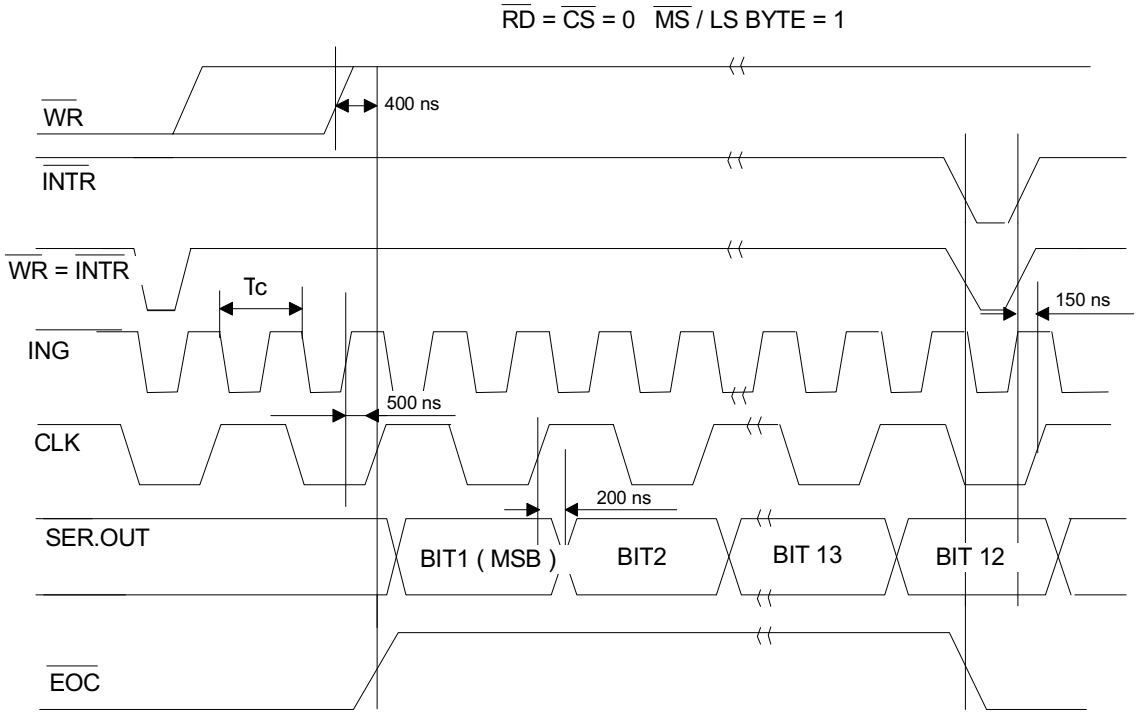

Figura A5

După o perioadă de ceas (o perioadă Tc a frecvenței de ceas fc) ieșirea INTR comută în starea "0" logic, această tranziție indicând că a fost finalizată conversia și rezultatul poate fi citit la ieșirile de date. Tranziția semnalului INTR are loc după 23..25 perioade de la frontul crescător (al doilea) al unui impuls de start având lățimea tw(WR)=Tc sau după 24..26 perioade de la frontul căzător (primul) al unui impuls de start cu lățimea tw(WR)<Tc. Procesul de conversie poate fi întrerupt de un al doilea impuls de start.

În lucrare MMC 757 este utilizat în modul de conversie continuă ("free-runing") prin conectarea semnalului INTR la intrarea WR și a intrării CS la masă. În acest caz timpul de conversie este de 25 perioade de ceas și durata întregului ciclu, incluzând semnalul INTR, este exact de 26 perioade de ceas.

Datele de la ieșire sunt citite prin aplicarea unui nivel logic "0" pe intrările RD și CS, fapt ce conduce la validarea driverelor de ieșire. Citirea octetului cel mai semnificativ este selectată prin nivelul logic "0" pe intrarea MS/LS BYTE. Dacă nivelul este “1” logic , atunci sunt citiți cei 4 biți LSB și semnalele OUTR , SER.OUT, CLK , și EOC. Rezultatul conversiei se menține în registrul de ieșire până la sfârșitul noii conversii. Alinierea la stânga a rezultatului conversiei permite ca în cazul preciziei de 8 biți , rezultatul conversiei să fie obținut printr-o singură citire.

La sfârșitul conversiei, semnalul EOC, exprimând starea convertorului A/D comută din “1” în “0” simultan cu INTR și revine în starea “1” numai când bitul cel mai semnificativ al noii

conversii apare la ieșirea serială SER.OUT. Astfel , EOC =1 pe durata conversiei și la intervale de două perioade de ceas, toți cei 12 biți apar unul după altul, începând cu MSB, la ieșirea SER.OUT.

Spre deosebire de EOC , ieșirea INTR trece în “1” logic la circa 300 ns după ce un nou impuls de start (WR=CS=0) este aplicat sau după ce este aplicată o comandă de citire (RD=CS=0). Frecvența de ceas poate fi selectată într-o gamă largă de la 25KHz pâna la peste 1MHz, dar precizia maximă de conversie se obține la frecvența fc cuprinsă între 100 și 250 Khz.

În lucrare, circuitul lucrează în modul *free-runing* prin conectarea semnalului INTR la intrarea WR și a intrării CS la masă, în acest caz avem un timp de conversie de 25 de perioade de clock. Frecvența de clock este reglată la 250khz cu rezistența R_G=22kOhm și C_G=330pF. Condensatorul C_G se leagă între intrarea ING și masă, iar rezistența R_G se leagă între ING și OUTG. ING este intrarea generatorului de ceas, OUTG ieșirea generatorului de ceas.

Conectarea circuitului MMC757 este prezentată în schema A6:

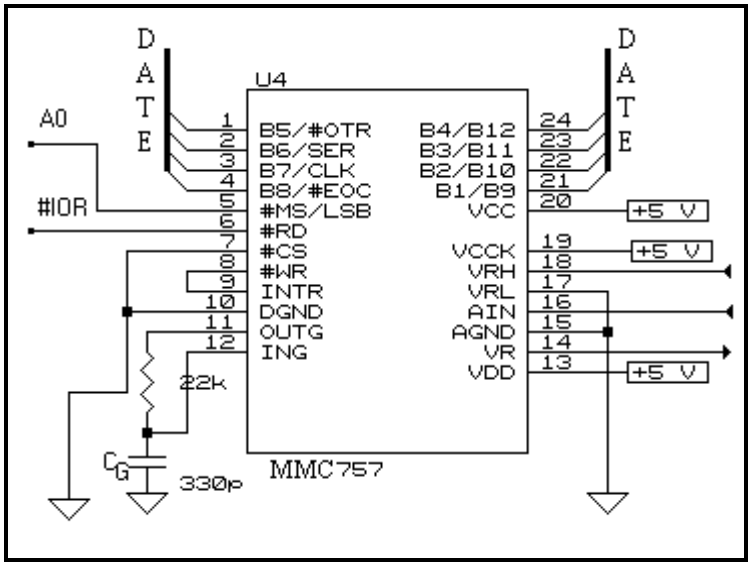

Figura A6

Linia de adresă A0 a fost legată la pinul #MS/LSB de selecție a octetului de ieșire. Astfel, la adresa 300H vom avea octetul cel mai semnificativ și la adresa 301H vom avea octetul mai puțin semnificativ. Pinul #RD este legat la linia #IOR a magistralei ISA. Ieșirea INTR este legată la intrarea #RD pentru că circuitul lucrează în modul de conversie continuă. VR reprezintă ieșirea tensiunii de referință incorporate în circuit, ea este folosită pentru a obține tensiunea VRH. În intarea analogică AIN intră semnalul analogic prelucrat de blocul de adaptare.

Semnificația terminalelor

În figura A6 este reprezentată și configurația terminalelor circuitului MMC757:

BIT1 - BIT12 - ieșirile de date ale convertorului

OUTR - “Vin este OUT OF RANGE”. Acest semnal este logic “0” când rezultatul conversiei este 000 sau FFF.

SER.OUT - ieșirea serială a rezultatului conversiei.

CLK - semnalul de ceas pentru sincronizarea citirii ieșirii seriale.

EOC - semnalizează încheierea conversiei. În afara ciclului de conversie e “0” .

WR - Intrare de scriere. Declanșează conversia. Activă în “0” .

RD - Comandă de citire. La aplicarea unui nivel logic “0” sunt validate ieșirile.

MS / LS BYTE - Comandă de selecție a octetului de date la ieșire : “0”-octet MSB “1”- octet LSB (și ieșire serială).

CS - Comanda de selecție a circuitului . În starea “0” , validează WR și RD

INTR - Sfârșit de conversie . Activ în “0”.

ING - Intrarea generatorului de ceas.

OUTG - Ieșirea generatorului de ceas.

AIN - Intrare analogică .

VRL - Intrarea LOW a tensiunii de referință.

VRH - Intrare HIGH a tensiunii de referință.

VR - Ieșirea tensiunii de referință încorporate în circuit.
VCC - Tensiunea de alimentare Vcc pentru partea digitală a circuitului.
VCCK - Tensiunea de intrare Vcc pentru comparatorul analogic.
VDD - Terminalul de intrare pentru aplicarea tensiunii de alimentare Vdd.
DGND - Masă digitală (terminalul comun al surselor de alimentare).
AGND - Masă analogică.

Instrucțiunile microprocesoarelor și coprocesoarelor din familia INTEL 8086, 80286, 80386, 80486

Arithmetic Instructions

ADD dest,src add two operands, placing result in dest
ADC dest,src add two operands, using carry from previous ADD
INC dest add 1 to dest (reg or r/m)
SUB dest,src subtract src from dest, leaving result in dest
SBB dest,src subtract using borrow (carry flag) from previous SUB
DEC dest subtract 1 from dest (reg or r/m)
CMP dest,src compare (non-destructive subtract) flags are set to indicate carry, overflow, etc.
NEG dest change sign of an operand
AAA adjust after ASCII addition (AL corrected for ASCII addition)
AAS adjust for ASCII subtraction (AL corrected for ASCII subtraction)
AAM adjust after ASCII multiply (AH:AL corrected for ASCII multiplication)
AAD adjust before ASCII division
DAA adjust after BCD addition (AL corrected for BCD addition)
DAS adjust for BCD subtraction (AL corrected for BCD subtraction)
MUL src multiply AL(AX) by unsigned value src (reg/mem)
IMUL src multiply AL(AX) by signed integer src (reg/mem)
IMUL reg16,r/m,im16 multiply reg/mem by 16-bit immediate signed value (286)
DIV src divide accumulator by unsigned value src (reg/mem). Division by 0 or result too large causes INT 0.
IDIV src divide accumulator by unsigned value src (reg/mem)
CBW convert byte to word (sign-extend AL into AH)
CWD convert word to dword (sign-extend AX into DX)
AND dest,src logical AND (mask; reset dest bits which are 0s in src)
TEST dest,src non-destructive AND (bit test; JZ will jump if no match)
OR dest,src inclusive OR (set dest bits which are 1s in src)
XOR dest,src exclusive OR (toggle dest bits which are 1s in src)
NOT dest form 1's complement (toggle every bit in dest)

Data Transfer Instructions

MOV dest,src transfer (copy) data to/from register, to/from memory
XCHG dest,src exchange values of two registers or register and r/m
IN port8 (or DX) input to AL(or AX) from I/O port
OUT port8 (or DX) output from AL(or AX) to I/O port
XLAT translate AL into a value in a translation table at BX
LEA reg16,addr load address into a register
LDS reg16,mem load DS and reg16 from memory variable
LES reg16,mem load ES and reg16 from memory variable
LAHF load/convert flags into AH
SAHF store/convert AH into flags
PUSH src transfer reg16 or r/m16 to stack
PUSH immed transfer immed16 (or sign-extended immed8) to stack.
PUSHA PUSH ALL: copy general registers to stack. (286 only)
SP-=10H; AX,BX,CX,DX,SI,DI,BP,SP saved on stack
PUSHF PUSH Flags: transfer flags register to stack
POP dest transfer from stack to dest (reg16, r/m16, or segreg)
POPA POP All: copy general registers from stack. 286 only.
SP+=10H; AX,BX,CX,DX,SI,DI,BP restored from stack

POPF POP Flags: transfer from stack to flags register

Execution Control Instructions

JMP target unconditionally transfer control to target
JCXZ short_label jump if CX==0
LOOP short_label CX=(CX-1) jump if CX!=0
LOOPE/ short_label CX=(CX-1) jump if CX!=0 && ZF==ZR==1
LOOPZ
LOOPNE/ short_label CX=(CX-1) jump if CX!=0 && ZF==NZ==0
LOOPNZ

Jccc short_label jump when condition ccc is met

JA/JNBE short_label jump if Above ((CF & ZF)==0 after unsigned math)
JAE/JNB short_label jump if Above or Equal (CF==NC==0 after unsigned math)
JB/JC short_label jump if Below/Jump if Carry set (CF==CY==1)
JE/JZ short_label jump if Equal (ZF==ZR==1)
JG/JNGE short_label jump if Greater (SF==(OF & ZF) after signed math)
JGE/JNL short_label jump if Greater or Equal (SF==OF after signed math)
JL/JNGE short_label jump if Less (ZF != OF after signed math)
JLE/JNG short_label jump if Less or Equal (SF!=OF || ZF==0 after signed math)
JNC short_label jump if carry not set (CF==NC==0) (same as JAE/JNB)
JNE/JNZ short_label jump if Not Equal (ZF==NZ==0)
JNO short_label jump if Not Overflow (OF==NO==0)
JNP/JPO short_label jump if Parity Odd (PF==PO==0: count of 1-bits is ODD)
JNS short_label jump if Not Sign (SF==PL==0: same as high-bit of dest)
JO short_label jump if Overflow (OF==OV==1)
JP/JPE short_label jump if Parity Even (PF==PE==1 count of 1-bits is EVEN)
JS short_label jump if Sign (SF==NG==1: same as high-bit of dest)

BOUND reg16,lmts perform limit-check on reg16. lmts is the address of a
2-word table with desired min/max limits. 286 only.
if (reg16<DS:[lmts]) or (reg16>DS:[lmts+2]) then INT 5

ENTER frmsiz,frms set high-level language stack frame. Use as the first
operation in a CALLed procedure. 286 only. Same as:
PUSH BP; (repeated frms times)
MOV BP,SP;
PUSH SP;
SUB SP,frmsiz; (allocate dynamic space on stack)
LEAVE undo the effect of ENTER. Use just before RET.
Restores SP and BP to values at time of ENTER.
INT type perform a software interrupt (call a system function)
INTO type if OF==OV==1, then perform INT type
IRET return from interrupt. Effectively the same as:
POP IP; POP CS; POPF

Processor Control Instructions

CLC clear the carry flag to NC
CMC complement (reverse the value of) the carry flag
STC set the carry flag to CY
CLD clear direction flag to UP (string ops auto-increment)
STD set direction flag to DN (string ops auto-decrement)
CLI disable maskable hardware interrupts
STI enable maskable hardware interrupts
CTS/CLTS clear task switch flag. 286
HLT halt processing (perform NOPs until an interrupt occurs)
WAIT/FWAIT wait for TEST line active (synchronize with coprocessor)
LOCK (prefix) prevent coprocessor bus access for the next instruction □386□ □486□ prevents access only to
mem affected by next opcode
SEG segreg (prefix) override default segreg for next EA calculation

Protection Control Instructions

All instructions on this page require 286 or later CPU

LGDT src load Global Descriptor Table from 6-byte table at src
SGDT dest store 6-byte Global Descriptor Table to memory at dest
LIDT src load Interrupt Descriptor Table from 6-byte table at src
SIDT dest store 6-byte Interrupt Descriptor Table to memory at dest
LLDT src load Local Descriptor Table (GDT selector) from reg/mem16
SLDT dest store Local Descriptor Table register into dest (r/m16)
LMSW src load Machine Status Word (use to enter protected mode)
SMSW src store Machine Status Word to reg/mem16
LTR src load Task Register (GDT selector) from reg/mem16
STR dest store Task Register to reg/mem16
LAR dest,src load high-byte of dest with Access Rights of src descriptor
LSL dest,src load dest with Segment Limit of descriptor named by src
ARPL lvl Adjust Requested Privilege Level to higher of current or lvl
VERR seg Sets ZF to ZR if task has read privileges for segment seg
VERW seg Sets ZF to ZR if task has write privileges for segment seg

String Operation Instructions

CLD clear direction flag to UP
sets - (Delta) to positive. String ops auto-increment
STD set direction flag to DN
sets - (Delta) to negative. String ops auto-decrement
REP/REPE/REPZ (prefix) repeat: perform string operation repeatedly CX=(CX-1); string op repeats until CX==0
REPNE/REPNZ (prefix) repeat: useful for string ops CMPS and SCAS
MOVSB copy byte(word) string (byte:-=1, word:-=2)
MOVSW
MOVSXB copy byte(word) with SF (sign) extension □386□
MOVSXW copy byte(word) with ZF (zero) extension □386□
LODSB copy string byte(word) into AL(AX)
LODSW
STOSB store bytes(words) into string
STOSW
CMPSB compare byte(word) strings (byte:-=1, word:-=2)
CMPSW
SCASB find byte(word) in string
SCASW
INSB port input byte(word) into string (byte:-=1, word:-=2)
INSW 286 only.
OUTSB port output byte(word) from string (byte:-=1, word:-=2)
OUTSW 286 only.

386/486-specific Opcodes

Opcodes on this page are supported by □386□. Opcodes for □486□ are flagged.

BSF reg,src Bit Scan Forward. Put index of next 1-bit of src into reg
BSR reg,src Bit Scan Reverse. Put index of prev 1-bit of src into reg
BSWAP reg32 Swap low-to-high ordering of 4-byte reg32 to high-to-low ordering. □486□ only
BT r/m,im8 Test bit. Put bit im8 (or reg) of reg/mem into CF.
r/m,reg (im8 or reg is used modulo 16 or modulo 32)
BTC r/m,im8 Test bit im8 (or reg) of r/m & complement that bit.
r/m,reg
BTR r/m,im8 Test bit im8 (or reg) of reg/mem and reset that bit.
r/m,reg
BTS r/m,im8 Test bit im8 (or reg) of reg/mem and set that bit.
r/m,reg
CMPXCHG r/m,reg Compare and Exchange. □486□ only
INVD Invalidate Data Cache. □486□ only
INVLPG mem Invalidate Translation Lookaside Buffer (TLB) entry □486□ only
JECXZ target Jump to target if ECX is 0.
LFS reg,mem Load FS and reg from memory.

LGS reg,mem Load GS and reg from memory.
LSS reg,mem Load SS and reg from memory.
MOV CRn,src32 Load src32 data into Control Register n (0,2,or 3)
MOV dest,CRn Load Control Register n into dest (reg/mem32)
MOV DRn,src32 Load src32 data into Debug Register n (0-3,6,or 7)
MOV dest,DRn Load Debug Register n into dest (reg/mem32)
MOV TRn,src32 Load src32 data into Test Register n (3-7)
MOV dest,TRn Load Test Register n into dest (reg/mem32)
SETccc dest If condition ccc is true, set the byte at dest to 1; else, set it to 0. (ccc is the same as options of conditional jump ops. See Execution Control Instructions)
SHLD r/m,reg,im8 Shift left r/m (16- or 32-bit) by im8 (or CL) bits,
SHLD r/m,reg,CL inserting data from reg into the vacated positions
SHRD r/m,reg,im8 Shift right r/m (16- or 32-bit) by im8 (or CL) bits,
SHRD r/m,reg,CL inserting data from reg into the vacated positions
WBINVD Write-Back and Invalidate Data Cache. □486□ only
XADD r/m,reg Exchange and Add. □486□ only

80x87 Floating Point Opcodes

These instructions can be executed when a 8087/80287/80387 coprocessor is available or when using an 80486 CPU.

Data Transfer and Constants
FLD src Load real: st(0) - src (mem32/mem64/mem80)
FILD src Load integer: st(0) - src (mem16/mem32/mem64)
FBLD src Load BCD: st(0) - src (mem80)
FLDZ Load zero: st(0) - 0.0
FLD1 Load 1: st(0) - 1.0
FLDPI Load pi: st(0) - Ñ (ie, pi)
FLDL2T Load log2(10): st(0) - log2(10)
FLDL2E Load log2(e): st(0) - log2(e)
FLDLG2 Load log10(2): st(0) - log10(2)
FLDLN2 Load loge(2): st(0) - loge(2)
FST dest Store real: dest - st(0) (mem32/mem64)
FSTP dest dest - st(0) (mem32/mem64/mem80); pop stack
FIST dest Store integer: dest - st(0) (mem32/mem64)
FISTP dest dest - st(0) (mem16/mem32/mem64); pop stack
FBST dest Store BCD: dest - st(0) (mem80)
FBSTP dest dest - st(0) (mem80); pop stack

Compare
FCOM Compare real: Set flags as for st(0) - st(1)
FCOM op Set flags as for st(0) - op (mem32/mem64)
FCOMP op Compare st(0) to op (reg/mem); pop stack
FCOMPP Compare st(0) to st(1); pop stack twice
FICOM op Compare integer: Set flags as for st(0) - op (mem16/mem32)
FICOMP op Compare st(0) to op (mem16/mem32); pop stack
FTST Test for zero: Compare st(0) to 0.0
FUCOM st(i) Unordered Compare: st(0) to st(i) □486□
FUCOMP st(i) Compare st(0) to st(i) and pop stack
FUCOMPP st(i) Compare st(0) to st(i) and pop stack twice
FXAM Examine: Eyeball st(0) (set condition codes)

Arithmetic
FADD Add real: st(0) = st(0) + st(1)
FADD src st(0) = st(0) + src (mem32/mem64)
FADD st(i),st st(i) = st(i) + st(0)
FADDP st(i),st st(i) = st(i) + st(0); pop stack
FIADD src Add integer: st(0) = st(0) + src (mem16/mem32)
FSUB Subtract real: st(0) = st(0) - st(1)
FSUB src st(0) = st(0) - src (reg/mem)
FSUB st(i),st st(i) = st(i) - st(0)
FSUBP st(i),st st(i) = st(i) - st(0); pop stack
FSUBR st(i),st Subtract Reversed: st(0) □ st(i) - st(0)
FSUBRP st(i),st st(0) = st(i) - st(0); pop stack

FISUB src Subtract integer: st(0) = st(0) - src (mem16/mem32)
FISUBR src Subtract Rvrsd int: st(0) = src - st(0) (mem16/mem32)
FMUL Multiply real: st(0) = st(0) * st(1)
FMUL st(i) st(0) = st(0) * st(i)
FMUL st(i) st(0) = st(0) * st(i)
FMUL st(i),st st(i) = st(0) * st(i)
FMULP st(i),st st(i) = st(0) * st(i); pop stack
FIMUL src Multiply integer: st(0) = st(0) * src (mem16/mem32)
FDIV Divide real: st(0) = st(0) ÷ st(1)
FDIV st(i) st(0) = st(0) ÷ t(i)
FDIV st(i),st st(i) = st(0) ÷ st(i)
FDIVP st(i),st st(i) = st(0) ÷ st(i); pop stack
FIDIV src Divide integer: st(0) = st(0) ÷ src (mem16/mem32)
FDIVR st(i),st Divide Rvrsd real: st(0) =st(i) ÷ st(0)
FDIVRP st(i),st st(0) = st(i) ÷ st(0); pop stack
FIDIVR src Divide Rvrsd int: st(0) = src ÷ st(0) (mem16/mem32)
FSQRT Square Root: st(0) = û st(0)
FSCALE Scale by power of 2
FXTRACT Extract exponent: st(0) = exponent of st(0); and gets pushed
FPREM Partial remainder: st(0) = st(0) MOD st(1)
FPREM1 Partial Remainder (IEEE): same as FPREM, but in IEEE standard □486□
FRNDINT Round to nearest int: st(0) = INT(st(0)); depends on RC flag
FABS Get absolute value: st(0) = ABS(st(0)); removes sign
FCHS Change sign: st(0) = -st(0)

Transcendental
FCOS Cosine: st(0) = COS(st(0))
FPTAN Partial tangent: st(0) = TAN(st(0))
FPATAN Partial Arctangent: st(0) = ATAN(st(0))
FSIN Sine: st(0) = SIN(st(0))
FSINCOS Sine and Cosine: st(0) = SIN(st(0)) and is pushed to st(1) st(0) = COS(st(0)
F2XM1 Calculate
FYL2X Calculate Y * log2(X): st(0) is Y; st(1) is X; this replaces st(0) and st(1) with: st(0) * log2(st(1))
FYL2XP1 Calculate Y * log2(X+1): st(0) is Y; st(1) is X; this replaces st(0) and st(1) with: st(0) * log2(st(1)+1)

Processor Control
FINIT Initilaize FPU
FSTSW AX store Status word
FSTSW dest dest = MSW (mem16)
FLDCW src Load control word
FSTCW dest Store control word
FCLEX Clear exceptions
FSTENV dest Store environment: store status, control and tag words and exception pointers into memory at dest
FLDENV src Load environment: load environment from memory at src
FSAVE dest Store FPU state: store FPU state into 94-bytes at dest
FRSTOR src Load FPU state: restore FPU state as saved by FSAVE
FINCSTP Increment FPU stack
FDECSTP Decrement FPU stack
FFREE st(i) Mark reg st(i) as unused
FNOP No operation: st(0) = st(0)
WAIT/FWAIT Synchronize FPU & CPU: Halt CPU until FPU finishes current opcode.