CURS NR. 9, 10

PROCESOARE INTEL

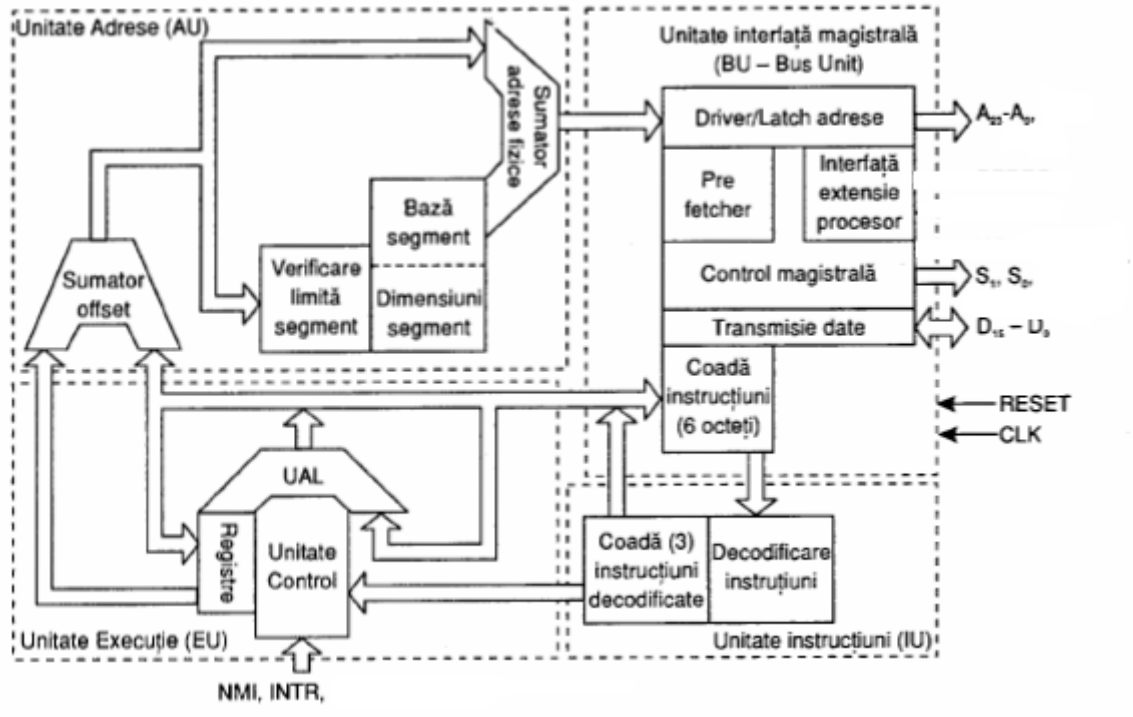
Structura procesorului 286

Arhitectura internă a microprocesorului 286 constă din patru unităţi funcţionale, care lucrează în paralel şi realizează execuţia instrucţiunilor ca pe o bandă de asamblare.

Funcţionarea procesorului se poate rezuma, în principiu, la o succesiune de operaţii care se execută în mod repetat :

1. Citeşte instrucţiunea - pe durata acestui ciclu se transmite adresa instrucţiunii de executat şi se aduce, din memorie, instrucţiunea în CPU (ciclul ***Fetch***).
2. Decodifică instrucţiunea.
3. Transmite adresa şi citeşte un operand din memorie, dacă se specifică în instrucţiune (ciclul ***Read***).
4. Execută instrucţiunea (ciclul ***Execution***).
5. Transmite adresa şi scrie rezultatul în memorie, dacă instrucţiunea o cere (ciclul ***Write***);

Cele patru unităţi funcţionale ale procesorului 286 sunt: *unitatea de adresare* (AU), *unitatea interfaţă cu magistrala* (BU), *unitatea de decodificare instrucţiuni* (IU) şi *unitatea de execuţie* (EU).



Unitatea de adresare (AU-Address Unit) determină adresa fizică şi, conţine un sumator de deplasament (offset), care determină deplasamentul în funcţie de modul de adresare şi un

sumator de adrese fizice, acesta determină adresa fizică, ca sumă între adresa de segment şi offset. De asemenea, unitatea mai realizează diferite verificări; verifică limita şi dimensiunea unui segment şi furnizează adresa de bază a segmentului.

Unitatea de interfaţă cu magistrala (BU – Bus Unit) conţine circuite driver şi latch pentru adrese, o unitate de citire anticipată a instrucţiunilor (PreFetcher), interfaţa cu extensia de procesor (extensia de procesor este un alt procesor specializat, ca de ex. procesorul matematic), logica pentru controlul magistralei, o coadă de instrucţiuni (6 octeţi) şi realizează transmisia/recepţia datelor. Lungimea cozii a fost aleasă astfel încât BU să ţină ocupată EU cât mai mult timp posibil. Această unitate realizează comunicaţia cu exteriorul.

Unitatea de citire anticipată (prefetcher) a instrucţiunilor realizează funcţia de anticipare a programului. Atunci când BU nu efectuează cicluri de magistrală (citire/scriere operand) pentru execuţia unei instrucţiuni, această unitate utilizează BU pentru citirea secvenţială, avans, a fluxului de instrucţiuni. BU lansează o cerere de citire (din memorie) de instrucţiune, imediat ce există 2 octeţi liberi în coada de instrucţiuni; citirea instrucţiunilor se face numai pe 16 biţi, într-un ciclu de acces la memorie. Dacă, prin program, se citeşte la o adresă impară, BU citeşte octetul de la adresa impară şi apoi reia citirea a câte doi octeţi de la adresele pare următoare.

În general, BU conţine cel puţin un octet în coada de instrucţiuni, deci EU nu trebuie să aştepte citirea instrucţiunilor. În coadă sunt introduse instrucţiunile de la adresele imediat următoare instrucţiunii curente, reprezentând instrucţiunile ce urmează să se execute, dacă nu se întâlneşte o instrucţiune de transfer al controlului de program (salt). Dacă, la un moment dat, se transferă controlul la altă locaţie de memorie, coada este iniţializată, BU citeşte instrucţiunea de la noua adresă, o transferă către EU şi începe imediat citirea instrucţiunilor următoare.

Unitatea de decodificare a instrucţiunilor (IU - Instruction Unit) preia octeţii din coada de instrucţiuni şi îi translează în microcod. Instrucţiunile decodificate (în număr de trei) sunt puse într-o coadă, unde aşteaptă să fie prelucrate de către unitatea de execuţie (EU). Unitatea lucrează în paralel cu celelalte unităţi şi începe o nouă decodificare în momentul în care o locaţie din coadă devine liberă.

Unitatea de execuţie (EU - Execution Unit) execută instrucţiunile din coada de instrucţiuni decodificate şi comunică cu celelalte unităţi. Această unitate implementează funcţiile de execuţie ale tuturor instrucţiunilor, furnizează adrese şi date spre AU, respectiv BU, manipulează registrele generale şi indicatorii de stare. Cu excepţia unor pini de control, EU este complet izolată de exterior. Când EU este gata pentru execuţia unei instrucţiuni, citeşte codul acesteia din coada de instrucţiuni decodificate gestionată de IU şi apoi execută instrucţiunea. Cele două unităţi funcţionează asincron şi se sincronizează în caz de coadă goală sau coadă plină. Dacă în timpul execuţiei unei instrucţiuni EU necesită un acces la memorie sau la

circuitele de I/O, se lansează o cerere de acces către BU, care controlează si efectuează accesul la adresa furnizată către AU. Dacă o astfel de cerere . apare în timp ce BU este într-un ciclu magistrală de citire a unei instrucţiuni, BU termină ciclul curent după care răspunde cererii de acces lansată de EU.

Semnificaţia semnalelor din schema bloc a procesorului:

· D15 ÷ Do= magistrala bidirecţională de date; ea este folosită ca intrare pentru date de la memorie, porturi de I/O, cicluri de recunoaştere întrerupere, respectiv ca ieşire pentru datele transmise la memorie sau porturi de I/O;

· A23 ÷ Ao = magistrala de adrese (de ieşire), care furnizează adresa fizică de memorie sau adresa unui port de I/O;

· INTR, NMI = cereri de întreruperi externe.

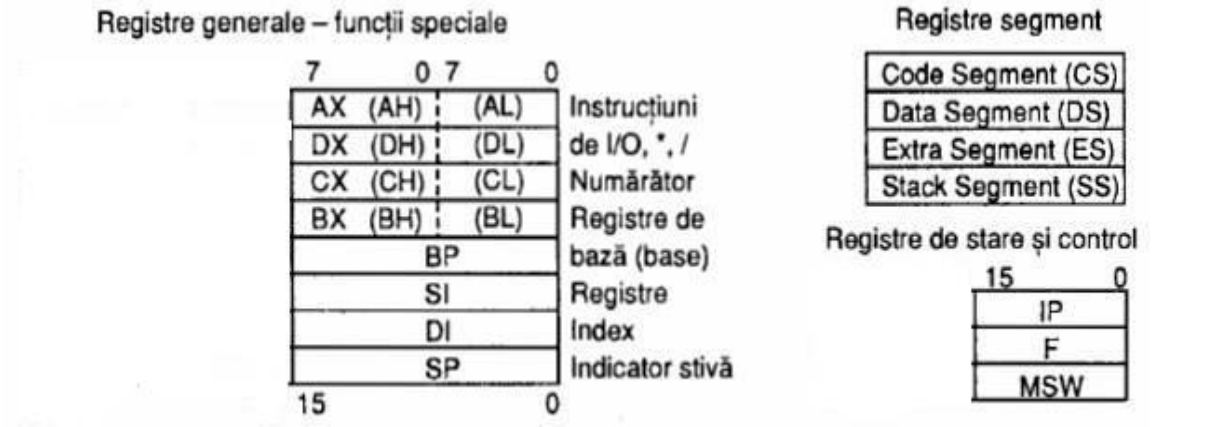
ARHITECTURA DE BAZĂ. SETUL DE REGISTRE

Setul de registre

Un registru este în esenţă o mică memorie, cu destinaţie specială. Spre deosebire de o locaţie de memorie, un registru de memorie oferă un acces rapid, fiind uşor de adresat. Ca zone de memorie, registrele sunt utilizate pentru memorarea de informaţie asupra căreia se operează într-un anumit mod. Aceştia sunt registrele specializate. Tot ca zone de memorie, registrele sunt folosite drept zone de prelucrare a datelor de către unitatea aritmetică-logică şi în transferul datelor cu memoria internă.

Procesorul 286 are 15 registre, grupate în patru categorii:

Cele patru categorii de registre sunt: generale, de segment, index şi de bază, de stare şi control.



Registrele generale

Sunt opt registre cu scop general: AX, BX, CX, DX, BP, SP, SI, DI, utilizate pentru a păstra operanzi logici şi aritmetici. Patru dintre acestea pot fi utilizate fie ca registre de 16 biţi referite ca AX, BX, CX, DX, fie pot fi împărţite în perechi de registre separate de câte opt biţi,

referite astfel: AH, AL, BH, BL, CH, CL, DH, DL (H=high, L~low, respectiv primii 8 biţi şi ultimii 8 biţi din registrele respective).

Majoritatea instrucţiunilor utilizează în acelaşi mod toate registrele; există însă instrucţiuni pentru care anumite registre generale au o anumită semnificaţie:

- AX şi DX sunt utilizate în instrucţiunile de înmulţire, împărţire şi cele de I/O cu o semnificaţie prestabilită; de asemenea, registrul AL pentru operaţii de translaţie de cod sau în operaţiile cu numere reprezentate în BCD (binary coded decimal);
- CX este utilizat drept numărător pentru operaţii de deplasare, rotire, bucle software, repetări hardware ale unor instrucţiuni.
- Registrele BX si BP, numite si registre bază, păstrează adresa de bază a structurilor de date, în timp ce registrele index SI si DI stochează offsetul în cadrul unei structuri de date;
- SP este registrul și mai specializat, deoarece el contine offsetul vârfului stivei în cadrul segmentului stivei.

Registrele segment

Sunt patru registre speciale ce permit selectarea, în orice moment, a segmentelor de memorie care sunt adresabile imediat:

- registrul segment de cod - CS
- registrul segment de date - DS
- registrul extrasegment, folosit pentru suplimentarea dimensiunii segmentului de date - ES
- registrul segment de stivă, folosit pentru localizarea stivei în memoria internă - SS

Pe lângă registrele de segment, vizibile programatorului, mai există pentru fiecare registru segment şi registre cache de descriptori de segment, care nu sunt vizibile programatorului, dar este util să cunoaştem conţinutul lor. Aceste registre, asociate cu fiecare registru segment, conţin: adresa de bază a segmentului (24 biţi), limita segmentului (16 biţi) si alte atribute de segment (8 biţi), deci în total 48 biţi. Când valoarea unui selector este încărcată într-un registru segment, registrul cache de descriptor asociat este în mod automat actualizat.

Orice referire la memorie va implica în mod automat registrul cache de descriptor de segment, asociat cu registrul segment utilizat pentru referire. Adresa de bază a segmentului devine o componentă în calculul adresei fizice, limita este utilizată pentru operaţii de verificare a limitei segmentului, iar atributele sunt utilizate pentru verificarea tipului (drepturilor) de referire la memorie.

Registre index şi de bază

Patru dintre registrele generale (BP, BX - registre de bază, SI, DI - registre index) pot fi utilizate, de asemenea, pentru a determina adresele de offset (deplasament) ale operanzilor din

memorie. Aceste registre pot conţine adresele de bază sau indexul pentru anumite locaţii de memorie dintr-un segment. Modul de adresare determină registrul specific utilizat pentru calcularea adresei efective a operandului. Registrele index sunt utilizate în mod implicit pentru a referi şirurile sursă, respectiv destinaţie pentru operaţiile pe şiruri. Registrul BP se referă implicit (dacă instrucţiunea nu este precedată de un prefix segment) la segmentul de stivă curent (SS). Tot la segmentul de stivă face referire, implicit pentru operaţiile cu stiva şi registrul SP (Stack Pointer), dar operaţiile cu stiva (PUSH, POP, CALL, RET, INT, IRET) actualizează în mod automat registrul SP, afectând în mod corespunzător stiva (extragerea sau introducerea informaţiei în stivă). În schimb, registrul BP poate fi folosit pentru acces la date din stivă (parametrii sau variabilele locale procedurii), fără însă a extrage datele din stivă (adică fără a modifica stiva).

Registre de stare şi control

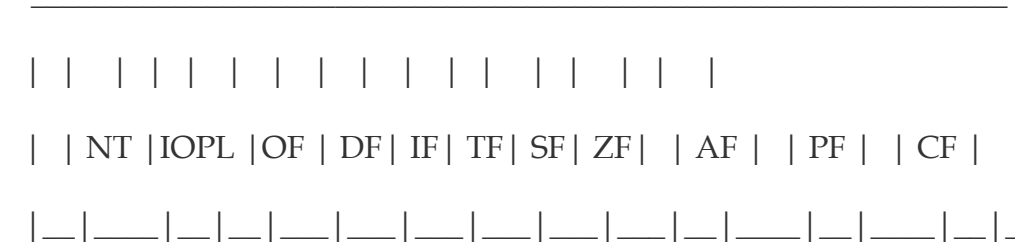
Sunt trei registre cu scop special, care memorează şi controlează anumite stări ale procesorului: IP, MSW şi F.

Registrul IP (Instruction Pointer) conţine adresa de offset pentru următoarea instrucţiune secvenţială ce va fi executată.

Registrul MSW (Machine Status Word) memorează starea procesorului, adică memorează dacă a avut loc o comutare de task şi controlează modul de operare a procesorului. Dintre cei 16 biţi sunt folosiţi doar ultimii patru - unul trece procesorul în modul protejat, iar ceilalţi trei controlează interfaţa cu extensia de procesor - şi au următoarea semnificaţie:

- PE (Protected mod Enable) este poziţionat pe 1 când se trece în modul protejat de lucru;
- MP (Monitor Processor extension/Math Present) specifică dacă este prezentă în sistem extensia de procesor;
- EM (Emulate processor extension) specifică dacă funcţiile extensiei de procesor sunt emulate prin software (1) sau nu (0);
- TS (Task Switched) memorează dacă a avut loc o comutare de task.

Registrul de indicatori F (Flags Register) conţine trei categorii de indicatori: de stare, de control şi specializaţi.



Registrul de indicatori (Flags Register)

Indicatorii de stare:

- *OF (Overflow Flag)* – indică (este poziţionat pe 1) dacă rezultatul unei operaţii aritmetice depăşeşte domeniul de reprezentare (limita superioară/inferioară a acestuia) pentru numerele cu care se lucrează, adică rezultatul nu poate fi memorat în destinaţia stabilită de instrucţiune (de exemplu, împărţirea prin zero sau adunarea/înmulţirea a două numere mari la limita superioară a domeniului).
- *SF (Sign Flag)* - este indicatorul de semn al rezultatului unei operaţii şi, de fapt, coincide cu primul bit al rezultatului, fiind 1 dacă numărul este negativ şi 0 pentru pozitiv.
- *ZF (Zero Flag)* - este poziţionat pe 1 dacă rezultatul unei operaţii aritmetice sau logice este zero, altfel bitul este poziţionat pe 0.
- *AF (Auxiliary carry Flag)* - este 1 dacă în urma execuţiei unei instrucţiuni a apărut un transport din rangul 3 în rangul 4 sau un împrumut dinspre rangul 4 spre rangul 3. Acest indicator este utilizat pentru implementarea aritmeticii pentru numere zecimale codificate binar (BCD - Binary Coded Decimal).
- *PF (Parity Flag)* - este poziţionat pe 1 când numărul de unităţi din rezultat este par (paritate pară); a nu se confunda cu paritatea unui număr (de exemplu numărul 2, care este un număr par, are reprezentarea binară 00000010, iar acest octet are paritatea impară - o singură unitate).
- *CF (Carry Flag)* - se poziţionează pe 1 dacă a apărut un transport sau un împrumut în/din rangul cel mai semnificativ al rezultatului, în urma execuţiei unei instrucţiuni aritmetice.

Indicatorii de control

- *DF (Direction Flag)* - este utilizat de instrucţiunile pe şiruri şi specifică direcţia de parcurgere a acestora:
  - 0, adică şirurile se parcurg de la adrese mici spre adrese mari;
  - 1 , adică şirurile sunt parcurse invers.
- *IF (Interrupt Flag)* - acest indicator controlează acceptarea semnalelor de întrerupere externă. Dacă IF = 1 , este activat sistemul de întreruperi, adică sunt acceptate semnale de întrerupere externă (mascabile, pe linia INTR), altfel acestea sunt ignorate. Indicatorul nu are influenţă asupra semnalului de întrerupere nemascabilă - NMI.
- *TF (Trace Flag)* - este utilizat pentru controlul execuţiei instrucţiunilor în regim pas cu pas (instrucţiune cu instrucţiune), în scopul depanării programelor.

Indicatori speciali:

- *IOPL (Input/Output Privilege Level)* - acest indicator ocupă doi biţi şi defineşte dreptul de a utiliza instrucţiuni de intrare/ieşire (I/O).
- *NT (Nested Task)* - este automat poziţionat pe 1 sau 0 de operaţiile de comutare de task; execuţia unei instrucţiuni IRET, cu NT=1, realizează o comutare de task.

CURS NR. 11

SUBSISTEMUL DE INTRARE-IESIRE

1. Magistrale – conecteaza unitatile functionale interne ale sistemului de calcul
2. Sistemul de I/E – conecteaza dispozitivele de I/E

MAGISTRALE

Magistrala este o cale electrica intre mai multe unitati functionale. Magistralelel se clasifica în:

- magistrale interne – ce se afla în interiorul microprocesorului;
- magistrale externe – în exteriorul microprocesorului.

Pentru orice magistrală se definieste un protocol de magistrală (bus protocol) care precizează regulile de functionare pe magistrală, și anume:

- semnificaâia semnalelor (date, adrese, control și stare);
- nivelele electrice ale semnalelor;
- specificatii mecanice și electrice.

Primele calculatoare aveau o singură magistrală care trecea pe la toate componentele sistemului de calcul numita magistrala sistem (system bus).

Calculatoarele noi au:

- o magistrala specializată între microprocesor și memorie numita magistrală de memorie;
- una sau mai multe magistrale ce leaga microprocesorul cu dispozitivele de I/E, numite magistrale de I/E.

Functionarea magistralelor

Gestiunea magistralei este realizata de un controler de magistrală (bus controller). În raport cu inițierea unei operații de transfer pe magistrala, dispozitivele conectate la aceasta pot fi:

- active (master);
- pasive (slave).

Memoria are rol de slave. Celelalte dispozitive pot fi master sau slave.

Exemple de relatii master-slave:

Master	Slave	Operatie
UCP	Memoria	Prelucrarea instructiunilor si datelor
UCP	Dispozitiv de I/E	Initializarea transferurilor de I/E
UCP	Coprocesor	UCP trimite instructiuni coprocesorului
Coprocesor	UCP	Coprocesorul prelucreaza operanzii de la UCP
I/E	Memorie	Acces direct la memorie (DMA)

Performantele unei magistrale sunt legate de ***viteza de transfer*** si de ***lărgimea de bandă***. Acestea sunt determinate de principalii parametri de proiectare, anume:

- numarul de linii de adresa, de date si de control;
- frecventa ceasului;
- mecanismul de arbitraj;
- gestionarea întreruperilor;
- gestionarea erorilor.

În functie de sincronizarea operațiilor de transfer magistralele se împart în:

- magistrale sincrone;
- magistrale asincrone/

Magistrale sincrone

Transferul datelor se face sincronizat dupa un semnal de ceas magistrală. Frecventa lui determină lungimea unui ciclu magistrala. Fiecare activitate se executa intr-un numar intreg de cicluri magistrala.

De exemplu o operatie de R/W se executa in cel putin trei cicli magistrala (timpi pentru stabilire adrese, semnale de comanda si date propriu zise).

O varianta de accelerare utilizeaza transferurile sincrone in blocuri de date (burst),indicandu-se adresa initiala si numarul de octeti de transferat.

Magistrale asincrone

Nu există un ceas master, iar ciclurile de magistrală pot avea orice lungime. Protocolul este format dintr-un set de semnale care se interblocheaza (handshaking). Se utilizeaza semnale suplimentare pentru realizarea unui dialog cu interblocare între cele 2 dispozitive, master si slave, între care se face transferul. Dialogul este independent de timp (semnalele se comanda reciproc), viteza de transfer adaptandu-se astfel automat la performantele dispozitivelor aflate in dialog.

Arbitrajul magistralei

Deoarece atât procesorul, cât și adaptoarele dispozitivelor de I/E pot dori simultan sa devina master pe magistrală, este necesară arbitrarea cererilor acestor dispozitive active.

Arbitrarea magistralei se poate realiza:

- centralizat (cu dispozitiv specializat ca arbitru de magistrala - unele microprocesoare pot avea incorporat un astfel de dispozitiv);
- descentralizat, cu linii suplimentare pe magistrală.

Ambele variante asigura accesul la magistrală, în functie de prioritătile alocate initial dispozitivelor aflate în conflict.

SISTEMUL DE INTRARE/IESIRE

Sistemul de I/E este componenta sistemului de calcul ce realizeaza comunicarea acestuia cu mediul extern.

Fiecare echipament de I/E este însoțit atât de:

1. interfața hardware (controller sau adaptor) - formată dintr-o serie de componente electronice ce asigură compatibilizarea nivelelor de semnal, cu standardele de magistrală utilizate, precum și o prima prelucrare logică a acestora.
2. interfața software (driver) - ce asigură o prelucrare logică și compatibilizarea semnificației semnalelor de la ieșirea interfeței hard cu standardele soft ale sistemului de operare.

Fiecare adaptor al unui dispozitiv de I/E contine (cel putin) 2 registrii. Unul va fi utilizat pentru transferul datelor, iar celalalt va contine informatii de comanda si stare. O informatie de stare esentiala este conditia ready, care arata daca adaptorul a incheiat operatia comandata si este pregatit sa preia o noua comanda. O operatie de I/E se executa prin transferul informatiilor de control si stare precum si a datelor propriu-zise intre procesor si registrele adaptorului. Transferul datelor intre procesor si adaptor are loc daca adaptorul este ready, adica a incheiat o operatie de intrare si are date pregatite pentru transfer in registrul de date, sau a incheiat o operatie de iesire si poate accepta alte date in registrul de date.

Exista doua categorii de arhitecturi ale sistemelor de I/E utilizate la sistemele de calcul moderne:

- canale de date;
- DMA.

Canale de date

Canalele de date se utilizeaza în special la calculatoarele de tip mainframe, cu arhitecturi complexe, conectate în retele cu utilizatori multiplii, frecventa operatiilor de I/E este foarte mare, iar magistrala este foarte solicitată. La aceste sisteme se utilizeaza calculatoare specializate pentru realizarea transferurilor de I/E, numite canale de I/E.

Arhitectura unui sistem cu canale de I/E presupune existenta unor magistrale specializate, independente:

- o magistrala pentru accesul CPU la memorie,
- magistrala de I/E prin care CPU comunica cu canalele,
- magistrala memoriei prin care canalele comunica cu memoria.

Un canal de date este de fapt un procesor de I/E la care sunt atasate toate dispozitivele de I/E. Acesta va executa un program special, comunicat de CPU și destinat executarii unui transfer complex. Se va executa urmatoarea succesiune de operatii:

- CPU transmite canalului programul de I/E .
- Canalul executa operatiile, realizand transferuri directe intre memoria principala si sistemul de I/E.
- Canalul semnalizeaza, printr-o intrerupere la procesor, incheierea transferului.

Functie de viteza de transfer a dispozitivelor atasate, exista doua tipuri de canale de I/E:

- canal multiplexor, la care se atasaza dispozitive lente (imprimante, terminale), canal care realizeaza, in paralel, operatii din programe de transfer pentru mai multe din dispozitivele conectate la el.
- canal selector, la care se atasaza dispozitivele rapide (discuri) si care realizeaza un singur transfer la un moment dat.

Avantajul acestei organizari consta în transferarea de la CPU catre procesorul de I/E a responsabilitatii majoritatii operatiilor legate de transferul datelor intre periferice si memoria principala.

DMA (acces direct la memorie)

Calculatoarele personale (PC-uri) au o structura mai simpla a sistemului de I/E. Acestea sunt organizate in jurul unor magistrale la care se conecteaza principalele module ale sistemului: placa de baza (procesor, memorie, controller-e pentru dispozitivele standard mai putin placa video) si controller-ele dispozitivelor de I/E.

Adaptoarele dispozitivelor ce transfera blocuri de date (discuri) realizeaza acces direct la memorie (DMA), adica transfera blocuri de caractere direct între disc și memoria principala, fără a ocupa timpul procesorului central.

Aceasta schema este utilizata pentru perifericele rapide și presupune eliberarea CPU de detaliile de realizare a transferului propriu-zis a unui grup de informatii între adaptorul dispozitivului de I/E și memoria principală. Aceasta functie a CPU este preluata de DMA, dispozitiv conectat la magistrala sistemului.

Chip-ul DMA va contine cel putin 4 registrii care vor fi incarcati, de catre programul ce se executa pe procesor, cu urmatoarele informatii ce definesc transferul:

- adresa de memorie la care se face transferul,
- numarul de octeti sau cuvinte ce trebuie transferat (contor),
- identificarea, în spatiul adreselor de I/E, a dispozitivului de I/E cu care se face transferul,

- sensul transferului (citire = intrare, scriere = iesire).

Utilizand aceste informatii, dispozitivul DMA va realiza transferul comandat, prin magistrala sistemului. DMA, devine astfel dispozitiv activ pe magistrala, intrând în competiție cu procesorul. Pentru transferul fiecarui cuvânt va emite semnalul bus request către dispozitivul de arbitraj a accesului pe magistrala și doar la încheierea transferului comandat va lansa o întrerupere către procesor.

Referitor la protocolul de acces la magistrala, DMA este prioritar fata de CPU, deoarece transferurile cu dispozitivele de I/E sunt, în general, condiționate de timp.

Problema ce apare în acest caz este legată de faptul ca aceste transferuri directe se fac pe acceasi magistrala cu restul transferurilor din sistem. Soluția problemei constă în utilizarea unui mecanism pentru arbitrarea accesului la magistrală.