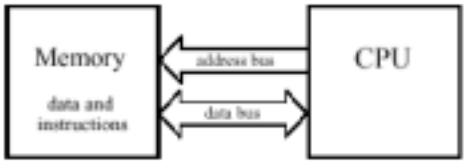
3. STRUCTURA ŞI FUNCŢIONAREA UNUI SISTEM CU MICROPROCESOR

Principiile generale prezentate în continuare se regăsesc în structura şi funcţionarea unui sistem de prelucrare numerică indiferent de tipul circuitului integrat pe scară largă utilizat şi care poate fi microprocesor, microcontroler sau procesor numeric de semnal (DSP).

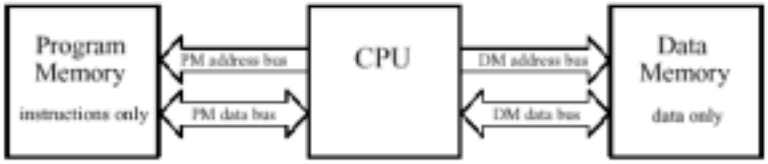
Unitatea de informaţie numerică sau logică utilizată în prelucrarea numerică este variabila binară (bitul) cu valori numerice 0 şi 1 sau logice DA şi NU. Pentru reprezentarea informaţiei numerice sau logice se utilizează secvenţe de 8, 16, 24, sau 32 de biţi, numite cuvinte. Cuvintele de 8 biţi se numesc octeţi.

Figura 3.1a reprezintă arhitectura clasică a unui microcalculator cunoscută sub numele de **arhitectura Von Neumann**, după matematicianul John Von Neumann (1903 - 1957). După cum se vede în figura 3.1.a arhitectura Von Neumann conţine o singură memorie şi o singură magistrală pentru transferul datelor din şi dinspre unitatea centrală de procesare (CPU). Pentru a analiza modul de fucţionare a acestuia şi a arhitecturilor ulterioare să

a. Arhitectura Von Neumann



b. Arhitectura Harvard



c. Arhitectura Super Harvard

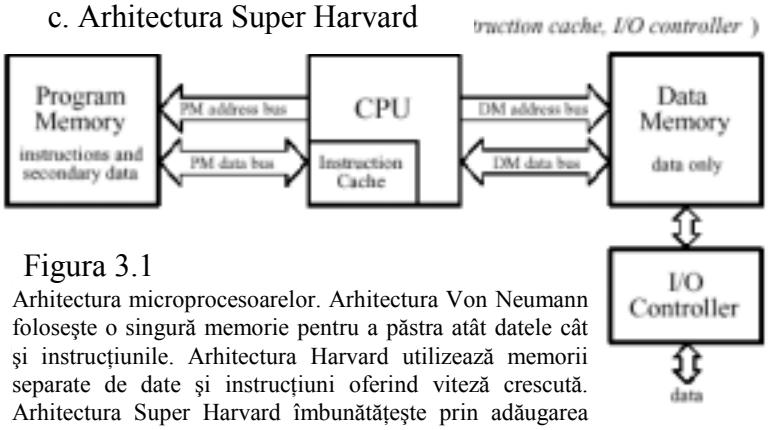


Figura 3.1
Arhitectura microprocesoarelor. Arhitectura Von Neumann foloseşte o singură memorie pentru a păstra atât datele cât şi instrucţiunile. Arhitectura Harvard utilizează memorii separate de date şi instrucţiuni oferind viteză crescută. Arhitectura Super Harvard îmbunătăţeşte prin adăugarea unui cache de instrucţiuni şi a unui controler de I/E.

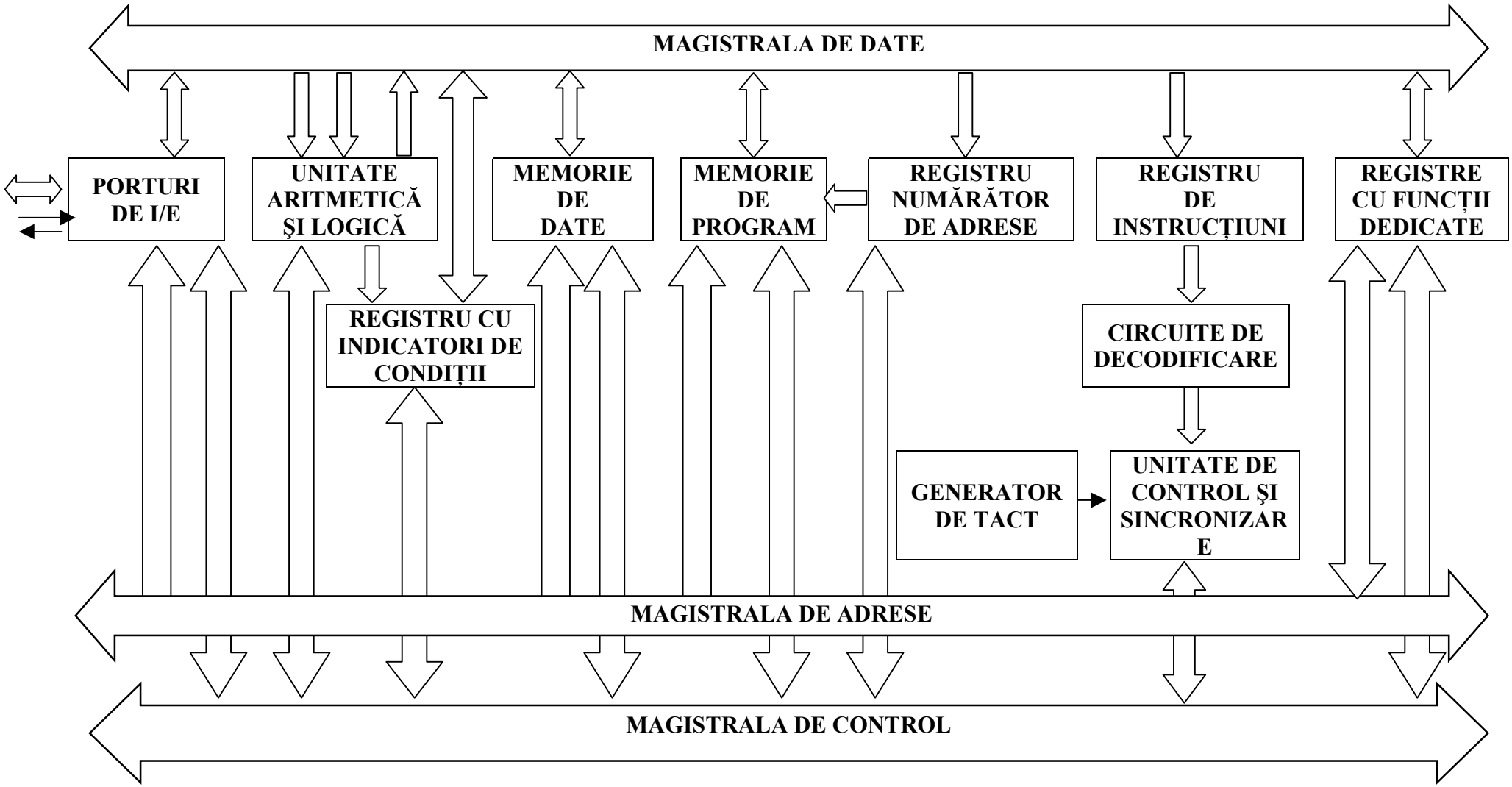
presupunem că trebuie să multiplicăm două numere care se găsesc undeva în memorie. Pentru aceasta trebuie să aducem trei valori binare din memorie: numerele care trebuiesc multiplicate şi instrucţiunea care descrie ce trebuie făcut. Multiplicarea a două numere necesită cel puţin trei cicluri de clock, câte unul pentru transferul fiecărui număr prin magistrală din memorie la CPU. Nu am socotit timpul necesar pentru transferul rezultatului înapoi în memorie pentru că am presupus ca acesta rămâne în CPU pentru manipulări ulterioare. Arhitectura Von

Neumann este destul de satisfăcătoare când este suficientă executarea tuturor taskurilor în serie. De fapt majoritatea calculatoarelor de astăzi au arhitectură Von Neumann. Avem nevoie de alte arhitecturi numai dacă este nevoie de o procesare foarte rapidă şi suntem dispuşi să plătim preţul pentru o arhitectură mai complexă. Aceasta necesitate a condus la **arhitectura Harvard** prezentată în figura 3.1.b Cum se poate vedea aceasta are memorii separate de date şi program cu magistrale separate pentru fiecare. Deoarece aceste magistrale lucrează independent, instrucţiunile şi datele pot fi aduse în acelaşi timp îmbunătăţind viteza faţă de arhitectura cu o singură magistrală. Figura 1.c prezintă următoarea treaptă de complexitate, **arhitectura Super Harvard** sau **arhitectura Harvard modificată.** Aceasta este construită după arhitectura Harvard căreia i s-au adus câteva îmbunătăţiri, relevate în figura 3.1.c: un *cache pentru instrucţiuni* şi un *controler de I/E*.

Dezavantajul structurii Harvard este că magistrala memoriei de date este mai ocupată decât magistrala memoriei de program. Când două numere sunt multiplicate două valori binare (care reprezintă numerele) trebuiesc transferate prin magistrala memoriei de date, în timp ce numai o valoare binară (instrucţiunea de program) este transferată prin magistrala memoriei de program. Pentru a îmbunătăţii această situaţie o parte din date se pot reloca în memoria program. Spre exemplu coeficienţii de înmulţire se pot plasa în memoria de program în timp ce semnalul de intrare se poate păstra in memoria de date. La prima vedere asta pare să nu ajute deoarece în acest caz trebuie să transferăm o valoare prin magistrala memoriei de date (eşantionul din semnalul de intrare) dar două valori prin magistrala memoriei de program (instrucţiunea de program şi coeficientul). De fapt dacă executăm instrucţiuni aleatoare situaţia nu va fi cu nimic mai bună. Totuşi de obicei algoritmii de procesare digitală a semnalelor îşi desfăşoară o mare parte din timpul de execuţie în bucle. Aceasta înseamnă că acelaşi set de instrucţiuni se vor transfera din memoria de program în CPU. Arhitectura Super Harvard se foloseşte de acest lucru prin includerea unui cache (memorie imediată) de instrucţiuni in CPU. Aceasta este o mica memorie de aproximativ 32 din cele mai recente instrucţiuni. La prima trecere prin buclă instrucţiunea este transferată prin magistrala memoriei de program. Aceasta duce la o execuţie mai lentă deoarece trebuie transferat prin aceeaşi magistrala şi coeficientul aflat tot în memoria de program. Totuşi la o nouă execuţie a buclei instrucţiunea poate fi extrasă din memoria cache. Aceasta înseamnă că toate transferurile de la memorie la CPU pot fi realizate într-un singur ciclu: eşantionul din semnalul de intrare este transferat prin magistrala memoriei de date, coeficientul prin magistrala memoriei de program, şi instrucţiunea din cache-ul de instrucţiuni. Acest transfer eficient de date este numită *lăţime de mare bandă, de acces la memorie*.

Structura unui sistem cu microprocesor este prezentată în detaliu în figura 3.2. Componenta esenţială este Unitatea Aritmetică şi Logică (ALU) care poate efectua operaţii de adunare, scădere, înmulţire, incrementare, decrementare, ŞI, SAU, SAU-exclusiv etc. ALU este un circuit combinaţional deci pentru memorarea temporară a operanzilor şi a rezultatului unei operaţii apare necesitatea unor registre de memorare. ALU generează rezultatul unei operaţii, şi în funcţie de acest rezultat, poziţionează la nivel logic 0 sau 1 biţii indicatori de condiţii. Aceştia pot indica de exemplu una dintre condiţiile: depăşirea domeniului de valori corespunzător reprezentării operanzilor ca urmare a unei operaţii de adunare sau scădere, obţinerea unui rezultat zero ca urmare a unei operaţii aritmetice sau logice, obţinerea unui rezultat pozitiv sau negativ, obţinerea unui rezultat cu un număr par sau impar de biţi cu valori de 1 etc.

Operanzii şi rezultatele operaţiilor se numesc date. Datele se memorează în registre şi în locaţii ale memoriei sistemului. Un registru sau o locaţie care conţine un operand pentru o operaţie se numeşte sursă pentru operaţia corespunzătoare. Un registru sau o locaţie în care se încarcă rezultatul unei operaţii se numeşte destinaţie pentru operaţia corespunzătoare.


MAGISTRALA DE DATE
PORTURI DE I/E
UNITATE ARITMETICĂ ŞI LOGICĂ
MEMORIE DE DATE
MEMORIE DE PROGRAM
REGISTRU NUMĂRĂTOR DE ADRESE
REGISTRU DE INSTRUCȚIUNI
REGISTRE CU FUNCȚII DEDICATE
REGISTRU CU INDICATORI DE CONDIȚII
CIRCUITE DE DECODIFICARE
GENERATOR DE TACT
UNITATE DE CONTROL ŞI SINCRONIZARE
MAGISTRALA DE ADRESE
MAGISTRALA DE CONTROL

Transferul datelor între componentele sistemului se realizează prin magistrala de date care interconectează aceste componente. Din punct de vedere fizic magistrala constă în conexiuni de 8, 16, sau 32 fire de legătură paralele. În funcție de numărul de biți ai magistralei de date se spune că sistemul este de 8, 16, sau 32 biți.

Memoria de date din structura unui sistem cu microprocesor (SMPU) se utilizează pentru memorarea datelor şi este de tip RAM sau ROM. Memoria de date de tip ROM se utilizează pentru memorarea unor constante ce intervin în algoritmii de prelucrare numerică. Adresarea locațiilor memoriei de date se face prin magistrala de adrese a SMPU de 16, 20, 24, sau 32 de biți. Fiecărei adrese referitoare la memoria de date îi corespunde o locație de memorie la care se poate efectua o operație de scriere a cuvântului încărcat pe magistrala de date de o altă componentă a SMPU, sau o operație de citire de către o altă componentă a SMPU a conținutului memoriei adresate, conținut încărcat de memorie pe magistrala de date. Magistralei de adrese de 16, 20, 24, sau 32 de biți îi corespunde un spațiu de adresare de 64K, 1M, 16M şi respectiv 4G cuvinte.

SMPU cu prinde un grup de registre de 8, 16 sau 32 biți, numite interne, având funcții speciale (dedicate). Aceste registre conțin date, adrese, şi informații de control. Astfel există registre dedicate care se utilizează ca surse cu operanzi sau ca destinații ale rezultatelor pentru anumite operații. Un registru cu funcții dedicate este registrul acumulator, notat A sau ACC, utilizat ca sursă şi destinație în multe operații aritmetice, logice şi de transfer.

Transferul de date între SMPU şi alte echipamente de intrare/ieşire (I/E) se realizează prin porturi de I/E care pot fi de tip paralel sau serie. În cazul unui port paralel, transferul unui cuvânt între SMPU şi un echipament de I/E se realizează printr-o magistrală de I/E de 8, 16 sau 32 de linii, funcție de lungimea cuvântului. transferul unui cuvânt de la un echipament la SMPU se numeşte operație de intrare, iar transferul invers se numeşte operație de ieşire. În cazul unui port serial, transferul de date se realizează prin două linii de comunicație, de transmisie şi respectiv de recepție. Biții corespunzători unui cuvânt se încarcă în ordine succesivă, cu frecvența de comunicație, pe linia de recepție sau de transmisie, funcție de sensul transferului. Adresarea porturilor de I/E se realizează prin magistrala de adrese a SMPU.

Din cele prezentate rezultă că magistrala de adrese conține cuvinte de adresare a datelor din memoria de date, registrele cu funcții speciale şi porturile de intrare. Aceste componente ale SMPU pot încărca magistrala de date. Deoarece la un moment dat o singură componentă a SMPU poate încărca magistrala de date, rezultă necesitatea selecției componentelor sistemului funcție de operațiile executate de acesta. Această selecție se realizează prin magistrala de control a SMPU de către unitatea de control şi sincronizare.

Funcția de prelucrare numerică este realizată de către sistem prin execuția secvențială a unor operații aritmetice, logice şi de transfer. Operațiile de transfer se realizează între componentele SMPU sau între SMPU şi echipamentele de I/E. O operație se realizează prin execuția de către SMPU a unei instrucțiuni. Rezultă că o succesiune de operații corespunde unei succesiuni de instrucțiuni, care formează un program. O instrucțiune este definită prin 1-4 cuvinte de 8 sau 16 biți, care conțin codul operației de executat, operanzii sau adresele operanzilor şi adresa destinației. Cuvintele care definesc o instrucțiune reprezintă codul maşină al instrucțiunii. Elaborarea unui program prin scrierea codurilor maşină ale instrucțiunilor corespunzătoare se numeşte programare în limbaj maşină.

Fiecărei instrucțiuni îi corespunde o scriere simbolică (cu caractere alfanumerice) care trebuie să precizeze aceleaşi informații ca şi codul maşină, informații care constau în codul operației, operanzii sau adresele operanzilor şi adresa destinației. Simbolul corespunzător codului operației se numeşte mnemonică. Elaborarea unui program prin scrierea simbolică a instrucțiunilor se numeşte programare în limbaj de asamblare.

Codurile maşină ale instrucţiunilor unui program sunt plasate la adrese succesive în memoria de program a SMPU. Memoria de program de tip ROM sau RAM, este conectată ca şi memoria de date la magistrala de date şi de adrese ale SMPU. Rezultă că magistrala de date se încarcă şi cu cuvintele reprezentând codurile instrucţiunilor.

Execuţia unei instrucţiuni începe cu extragerea din memoria de program a primului cuvânt din codul maşină, cuvânt care precizează codul operaţiei corespunzătoare instrucţiunii. Sub comanda unităţii de control şi sincronizare, acest cuvânt este transferat prin magistrala de date în registrul de instrucţiuni. Registrul de instrucţiuni realizează memorarea temporară a cuvântului cod operaţie în scopul decodificării. Rezultatul decodificării este transmis la unitatea de control şi sincronizare care comandă funcţionarea componentelor SMPU pentru execuţia instrucţiunii identificată prin decodificare. Această comandă se realizează prin magistrala de control. În cazul în care codul maşină al instrucţiunii conţine mai mult de un cuvânt execuţia instrucţiunii cuprinde şi extragerea din memoria de program a celorlalte cuvinte conţinând date şi/sau adrese. Extragerea în ordine succesivă a cuvintelor reprezentând codurile maşină ale instrucţiunilor unui program se realizează prin adresarea memoriei de program cu registrul numărător de adrese ale programului (PC). Registrul PC (de 16 biţi) se incrementează cu o unitate după fiecare extragere de cuvânt cod de instrucţiune. Unitatea de control şi sincronizare poate comanda încărcarea în registrul PC şi a altor valori decât cele rezultate din numărare în ordine naturală, rezultând salturi în citirea memoriei de program. Efectuarea unui astfel de salt se numeşte transfer al controlului şi poate rezulta ca urmare a execuţiei unei instrucţiuni de transfer al controlului (salt, apel de subrutină, revenire din subrutină) sau ca urmare a unei cereri de întrerupere.

Din cele prezentate rezultă că execuţia unei instrucţiuni de către SMPU cuprinde următoarele operaţii de bază:

- *extragere cod operaţie* – transferul din memoria de program în registrul de instrucţiuni al primului cuvânt din codul maşină al instrucţiunii, cuvânt care conţine codul operaţiei de executat prin instrucţiune;
- *decodificare* – analiza cuvântului cod operaţie cu circuitele pentru decodificarea instrucţiunilor şi transferul rezultatului decodificării la unităţii de control şi sincronizare;
- *transfer operanzi* - transferul operanzilor între componentele SMPU (memorie de program, memorie de date, registre, porturi de I/E) în scopul execuţiei instrucţiunii;
- *execuţie* – execuţia operaţiei aritmetice, logice sau de transfer precizată de codul operaţie al instrucţiunii.

Execuţia unei instrucţiuni începe cu extragere cod operaţie şi decodificare, continuând cu o secvenţă specifică de operaţii de bază de transfer operanzi execuţie. Astfel este necesară funcţionarea secvenţială şi sincronizată a SMPU, care se obţine prin comanda componentelor SMPU de către unitatea de control şi sincronizare. Viteza de execuţie a instrucţiunilor este funcţie de frecvenţa semnalului de tact al SMPU. În general o operaţie de bază se efectuează în 3 – 12 perioade ale semnalului de tact. Intervalul corespunzător efectuării unei operaţii de bază se numeşte **ciclu maşină** al sistemului. Ciclurile maşină corespunzătoare unei instrucţiuni definesc un **ciclu de instrucţiune**. Execuţia unei instrucţiuni durează câteva cicluri maşină incluzând : ciclu de extragere cod operaţie, ciclu de decodificare şi apoi funcţie de tipul instrucţiunii, cicluri de citire/scriere, cicluri de intrare/ieşire şi cicluri de execuţie. Efectuarea acestor cicluri poate implica componente diferite ale SMPU. Rezultă posibilitatea ca SMPU să efectueze simultan cicluri diferite din execuţia unei instrucţiuni sau din execuţia unor instrucţiuni succesive. Această tehnică de suprapunere în timp a execuţiei ciclurilor maşină (operaţie paralelă) se numeşte tehnică pipeline şi utilizarea ei în funcţionarea unui SMPU conduce la mărirea vitezei de lucru a acesteia.

În general, realizarea unui SMPU se bazează pe utilizarea unui circuit integrat pe scară largă de tip microprocesor, microcontroler sau procesor numeric de semnal, care conține, pentru orice tip, următoarele componente din structura generală a unui SMPU: unitate aritmetică şi logică, registru cu indicatori de condiții, registru numărător de adrese, registru de instrucțiuni, circuite de decodificare a instrucțiunilor, unitate de control şi sincronizare şi registre cu funcții speciale. Toate aceste componente se numesc interne, relativ la circuitul integrat utilizat ca bază pentru realizarea SMPU. În acest sens registrele din structura SMPU se numesc registre interne. Un circuit de tip microcontroler conține toate componentele din structura generală a unui SMPU deci conține şi memorie internă şi porturi de I/E.

3.1 Microprocesorul. Definitie

Microprocesorul este o unitate centrala de prelucrare realizata intr-un singur circuit integrat. In unele lucrari de specialitate, in categoria microprocesoarelor sunt incluse si procesoarele realizate cu un numar redus de circuite VLSI (de exemplu procesoarele obtinute prin alipirea mai multor unitati elementare de prelucrare pe un bit - arhitecturi "bit-slice").

Datorita costului redus si fiabilitatii ridicate microprocesoarele au o larga aplicabilitate fiind folosite pentru realizarea de microcalculatoare, echipamente periferice, sisteme de conducere a proceselor industriale, aparate medicale, etc. Aparitia acestora a revolutionat modul de proiectare al sistemelor de calcul.

3.2 Structura interna a unui microprocesor

In principiu un microprocesor inglobeaza toate blocurile functionale ale unei unitati centrale si anume:

-unitatea aritmetica si logica

-unitatea de comanda

-registre generale si speciale.

Un microprocesor dispune de un set de instructiuni adaptat structurii sale interne. In cazul microprocesoarelor actuale (ex.: 8088, 80486, Pentium) intr-un singur cip sunt inglobate mai multe procesoare care pot efectua o serie de operatii in paralel. Pentru un observator extern insa aceste procesoare se comporta asemeni unei singure unitati centrale.

3.2.1 Semnalele unui microprocesor

Semnalele unui microprocesor se pot imparti functie de rolul acestora in trei categorii: adrese, date si comenzi. Prin aceste semnale microprocesorul controleaza fluxul de date si secventa de evenimente dintr-un sistem de calcul. Aceste semnale sunt conectate la modulele de memorie si de intrare/iesire ale sistemului de calcul prin intermediul unui set de fire paralele care alcatuiesc o **magistrala**.

Numarul de linii de adresa determina spatiul de adresare al microprocesorului sau altfel spus numarul maxim de locatii de memorie sau de locatii dde intrare/iesire care pot fi adresate de microprocesor. Daca n este numarul acestor linii atunci spatiul maxim de adresare este 2^n . In mod obisnuit se folosesc 16, 20, 24 si 32 de linii.

Numarul de semnale de date de pe magistrala determina latimea maxima a cuvintului de date (sau cod) care se poate transfera . Numarul de semnale de date este in strinsa corelatie cu structura interna a microprocesorului. De obicei numarul semnalelor de date este multiplu de 8 (8, 16, 32, 64).

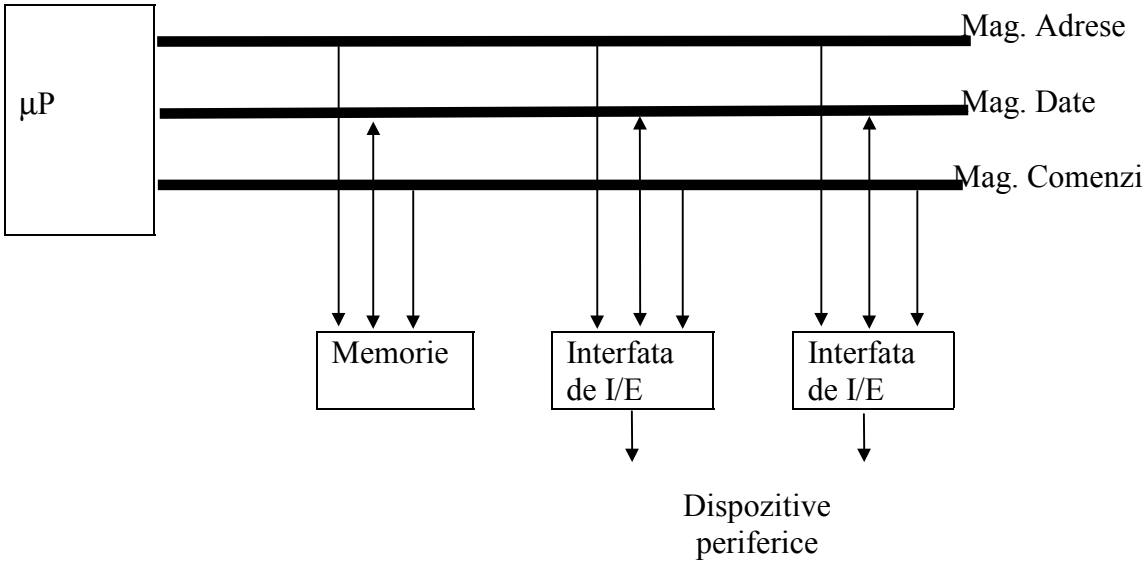


Figura 3.3. Configuratia de principiu a unui sistem cu microprocesor

Pe linga semnale de adrese si date un procesor utilizeaza si o serie de semnale de control. Aceste semnale au rolul de a controla fluxul de date si de a sincroniza microprocesorul cu anumite evenimente externe. Aceste semnale pot fi clasificate dupa natura lor in urmatoarele categorii:

-semnale de control
-semnale de intrerupere
-semnale pentru arbitrarea magistralei
-semnale de stare
-semnale diverse.

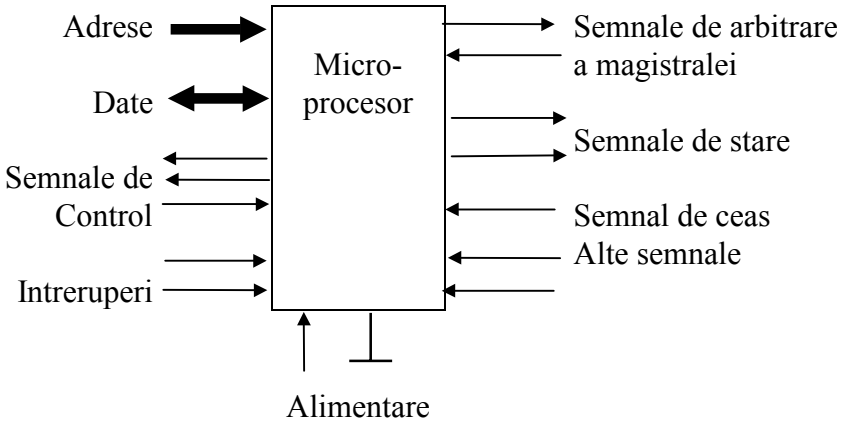


Figura 3.4 Semnalele unui microprocesor

Semnalele de control (figura 3.4) indica sensul transferului de date (citire/scriere) precum si sursa/destinatia transferului (memorie sau port de intrare/iesire).

Semnalele de intrerupere sunt intrari pentru microprocesor si au rolul de a indica aparitia unor evenimente externe. Microprocesorul anunta acceptarea unei intreruperi prin activarea unui semnal specific.

Semnalele de arbitrare a magistralei sunt necesare pentru reglarea traficului pe magistrala; permit accesul mai multor unitati master la resursele conectate pe magistrala. Unitatile master sunt cele care pot controla fluxul de date pe magistrala (ex: microprocesoare,

unitati de acces direct la memorie). Unitatile slave sunt cele care pot fi accesate (citite/scrise) de catre unitatile master.

Semnalele de stare permit monitorizarea functionarii microprocesorului de catre un depanator sau de catre alte module intim legate de microprocesor (coprocesor, controloare de magistrala etc.).

Intr-un sistem cu microprocesor se pot utiliza mai multe magistrale. Avantajul utilizarii mai multor magistrale consta in posibilitatea efectuarii unor transferuri in paralel. Utilizarea mai multor magistrale este indicata mai ales in cazul in care intr-un sistem sunt mai multe unitati master.