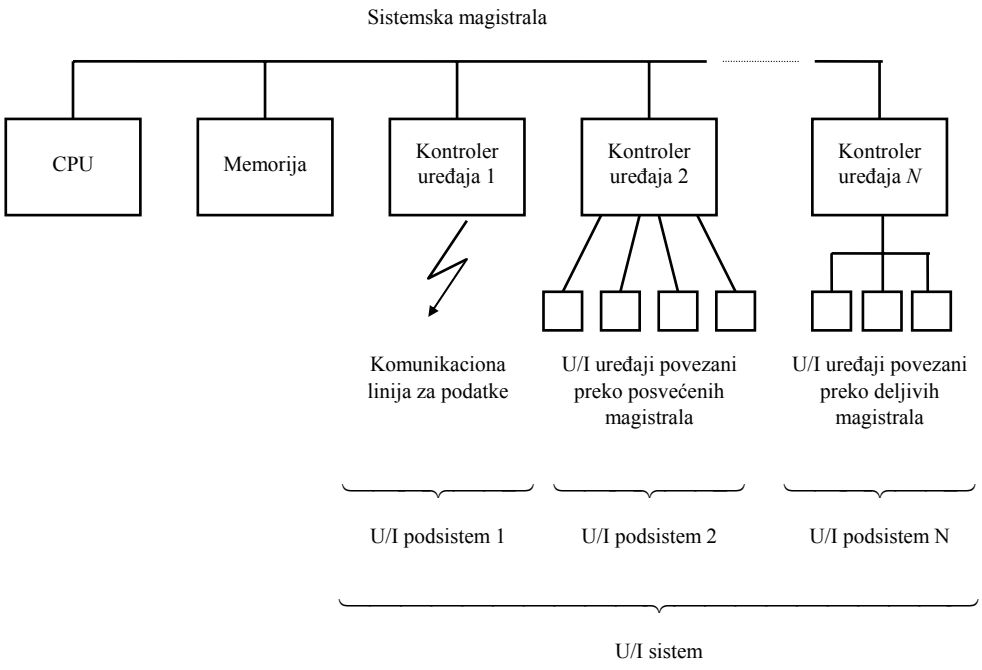
7. STRUKTURA U/I PODSISTEMA

7.1. Uvod

Komunikacija između računara i njegovog okruženja ostvaruje se preko U/I uređaja. U/I uređaji su delovi mašine namenjeni za čuvanje informacija, obavljanje konverzije podataka koji se prihvataju iz okruženja u oblik pogodan za čitanje od strane računara, i konverziju podataka na svojim izlazima u formi pogodnoj i razumljivoj za okruženje.

U/I uređaje koji se povezuju na CPU zovemo zajedničkim imenom *U/I sistem*. Jedan kontroler uređaja (*device controller*) i prateće uređaje koji se povezuju na kontroler zovemo *U/I podsistem* (I/O subsystem) (slika 7.1).



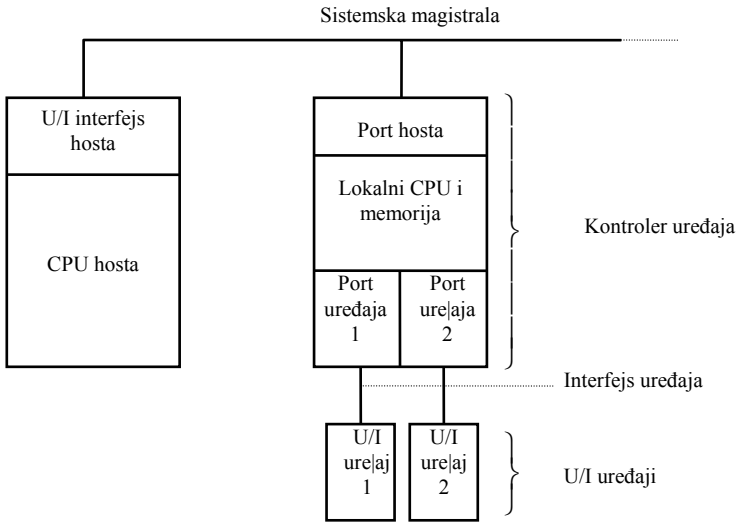
Sl. 7.1. Struktura U/I sistema i veza sa ostalim delovima računara.

Kontroler uređaja može da predstavlja jednostavan interfejs i nalazi se između CPU-a i U/I uređaja. U nešto složenijem izvođenju može da upravlja radom nekoliko uređaja koji mogu biti nezavisni od rada CPU-a. U svim slučajevima kontroleri uređaja omogućavaju realizaciju funkcija upravljanja i baferovanja koje su neophodne za korektan rad povezanih U/I uređaja. Kontroler uređaja može biti fizički u U/I uređaju, CPU-u, ili pakovan kao posebna jedinica. Neki od kontrolera uređaja mogu upravljati radom samo jednog uređaja, jer, ili je obično samo jedan uređaj takvog tipa potreban za rad sistema, ili zbog performansnih razloga (npr. veća brzina u radu kada se samo po jedan uređaj vezuje na jedan kontroler). Često, ipak, kontroler može biti zajednički za veći broj uređaja istog tipa, jer najveći broj uređaja nije istovremeno aktivan, tako da kontroler može da radi u multiprogramskom režimu rada.

Opšta struktura U/I podsistema prikazana je na slici 7.2.

U/I uređaji se povezuju na kontroler preko *interfejsa uređaja* (device interface). Kontroleri uređaja povezuju se na CPU preko *sistemske magistrale*.

Najjednostavniji kontroler uređaja obično čine sledeće dve celine: *port hosta* i *port uređaja*. Uobičajeno je da se po jedan port uređaja pridružuje uređaju. Host-port čuva statusnu informaciju koju CPU testira, kao i upravljačku informaciju (komandu) koju CPU upisuje u kontroler uređaja. Svaki port, posmatrano sa programske tačke gledišta definiše se svojom jedinstvenom adresom. Inteligentniji kontroleri uređaja karakterišu se većom lokalnom obradom i ugrađenom memorijom čime je omogućeno da se direktni pristup memoriji (DMA- direct memory access) i lokalna U/I obrada odvijaju konkurentno sa CPU-ovom aktivnošću. Port uređaja upravlja i obavlja prenos podataka ka/iz U/I uređaja. Veza između ova dva porta ostvarena je internom magistralom.



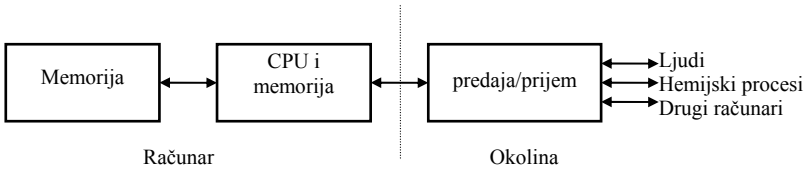
Sl. 7.2. Opšta struktura U/I podsistema.

7.2. U/I uređaji

U/I uređaji poseduju veliki broj osobina koje ih čine različitim od CPU-a ili memorije. Ova razlika je pre svega rezultat mehaničke prirode U/I uređaja (što čini da oni budu manje pouzdani i znatno sporiji u odnosu na elektronske uređaje) i raznovrsnosti u načinima njihove realizacije.

7.2.1. Klasifikacija

Skoro svaki računar opšte namene se može konfigurisati za specifičnu aplikaciju dodavanjem U/I uređaja koji su prilagođeni za tu aplikaciju. Ove U/I uređaje, koje često zovemo uređaji, možemo klasifikovati na sledeći način: *uređaji za memorisanje* - često se koriste od strane sistema za memorisanje podataka i njihovo povezivanje (čitanje), i *predajno/prijemni* (source/sink - pravi prevod je izvor/ponor) - koji se prvenstveno koriste od strane sistema za komuniciranje sa njegovim okruženjem (slika 7.3.).



Sl. 7.3. Veza računara i okruženja.

Uređaji za memorisanje

Ovi uređaji se mogu posmatrati kao memorijsko proširenje. Imajući u vidu da je memorija sistema realizovana elektronskim komponentama relativno skupa, i da je njen sadržaj nestalan nakon nestanka napajanja (misli se na RAM-ove), skoro kod svih sistema se danas ugrađuje mehanički uređaj za memorisanje. Svaki uređaj za memorisanje se karakteriše kapacitetom memorisanja koji je reda 1MB za flopi-disk, 1GB za "hard" disk, i do 50 GB za optički disk.

Primopredajni uređaji

Svrha primopredajnih uređaja je da obezbede komuniciranje između računara i njegovog okruženja. Prijem se ostvaruje preko uređaja tipa tastatura, analizator govora, digitajzer, miš i dr., ili preko uređaja tipa modem, senzor ili prekidač. Predajni uređaji, a to mogu biti štampači, grafički displeji, modemi, aktuatori i dr., generišu izlaznu informaciju. Komunikacione linije koje se koriste za spregu sa drugim računarima ili računarskim mrežama mogu se takođe smatrati primopredajnim uređajima.

7.2.2. Osobine U/I uređaja

Osobine uređaja koje imaju direktan uticaj na rad računarskog sistema su vreme pristupa, broj podataka koji se prenosi u jedinici vremena (propusnost) i procenat grešaka koje se javljaju u toku rada.

Vreme pristupa

Medijumi za memorisanje karakterišu se sekvencijalnim a ne proizvoljnim pristupom. Podatak kome pristupamo ne može se trenutno pročitati (ili se trenutno upisati) nego je najpre potrebno neko vreme da se obavi određeno mehaničko kretanje. Ovo vreme se zove *vreme pristupa*. Na primer, kod pomeranja glave diska, vreme pristupa čini vreme potrebno da se pozicionira glava (seek time) i vreme dok željeni sektor ne prođe izpod glave (rotational delay)

Vreme pristupa U/I uređajima je značajno duže od vremena pristupa glavnoj memoriji (oko 25ms za pomeranje glave na disku prema 0,5 μs za pristup glavnoj memoriji, a to znači da je taj odnos reda 5*10). Sa druge strane, vreme pristupa može takođe da varira značajno, ne samo zbog različitih uređaja (pristup informaciji na disku je brži od onog na magnetnoj traci), nego i zbog različitih tipova podataka koji se čuvaju na jednom istom uređaju. Pronalaženje proizvoljne reči na magnetnoj traci može da traje dugo, ali nalaženje sledeće reči može biti znatno kraće. Zbog toga, prirodno je da uređaji za memorisanje čitaju i upisuju podatke u blokovima, tako da pristup jednom podatku povlači sa sobom i pristup većem broju podataka (celom bloku podataka).

Vreme za prenos podataka

Vreme za prenos podataka U/I uređaja zavisi od iznosa podataka koji se prenose (obim bloka), i brzine prenosa podataka (vreme između sukcesivnih elemenata podataka). Obim bloka je određen tipom uređaja. Znakovno orijentisani uređaji (terminali) predaju svoje podatke bajt po bajt, dok blok orijentisani uređaji (diskovi ili trake) vrše prenos blokova podataka. Obim bloka je određen faktorima kao što su: vreme pristupa, vreme prenosa, fragmentacije, i zahtevom za baferovanje i dr.

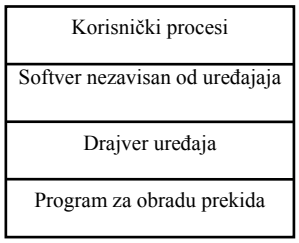
Brzina prenosa podataka je često mala u odnosu na prenos tipa memorija-memorija. Na primer, brzina prenosa ka/iz diska može biti 1MB/s. Ako usvojimo da je sektor na disku obima 1kB, za prenos jednog sektora biće potrebno 1ms.

Procenat greške

Zbog mehaničke prirode najvećeg broja uređaja za memorisanje, javlja se veći procenat greške u toku prenosa podataka u odnosu na onaj koji važi za prenos CPU↔memorija. Zbog fizičkog kretanja medijuma na kome je upisana informacija, velike gustine pakovanja (bitovi/inču), čestica u vazduhu, amortizacije i tolerancije javljaju se greške. Komunikacioni uređaji su često podložni greškama, zbog dugih neoklopljenih linija, koje su podložne radijacijama i drugim tipovima smetnji. Detekcija i korekcija ovih grešaka zahteva neki oblik hardverske redundancije kao i obezbeđenje neke mogućnosti da se pomoću softverskih sredstava ove greške isprave. Ovo se izvodi korišćenjem dodatnih bitova za proveru (korišćenjem kodova za korekciju greške), povratnom predajom primljene poruke (eho), ili ponovnim prenosom poruke.

7.3. U/I softver

Glavna uloga operativnih sistema je da upravlja svim U/I uređajima. Upravljanje se sastoji u korišćšenju U/I komandi i obradi prekida i grešaka. Pored toga, operativni sistem treba da obezbedi visok nivo intrefejsa prema uređajima. Interfejs treba da bude konzistentan za sve tipove uređaja kao i da bude jednostavan za korišćenje. Da bi ispunio ovo, U/I softver se organizuje na način kao što je prikazano na slici 7.4, tj. organizuje se u nivoe.



Sl. 7.4. Organizacija U/I softvera.

Na najnižem nivou se nalazi program za obradu prekida (interrupt handler). Program za pokretanje uređaja (device driver) sadrži sve kodove koji su karakteristični za taj tip uređaja. U suštini, to je jedini deo operativnog sistema koji prepoznaje prikačeni tip uređaja, kao što je tip korišćene U/I komande, fizička organizacija uređaja, mehaničke osobine uređaja i način koji ukazuje koje akcije treba preuzeti kada se jave greške u uređaju. Operativni sistem zahteva za svaki tip uređaja po jedan drajver uređaja, u kome se može specificirati upravljanje za veći broj uređaja (drajvova). Uloga drajvera uređaja je da prevede logičke komande visokog nivoa u fizičke komande niskog nivoa (na primer, čitaj disk *x*, pistu *y*, sektor *z*). Na primer, drajver uređaja obavlja sledeće akcije:

- obezbeđuje formiranje rada čekanja logičkih komandi,
- prevodi logički broj bloka u fizičku adresu na disku,
- preuređuje sekvencu zahteva za rad sa diskom sa ciljem da poboljaša propusnost,
- komunicira sa kontrolerom uređaja,
- vrši obradu greške (U/I uređaji imaju visok procenat greške).

Softver nezavisan od uređaja prevodi korisnički interfejs visokog nivoa (koristeći se pri tome simboličkim komandama) u logičke komande namenjene drajveruuređaja (uočimo da su granice sa drajverom uređaja ponekad proizvoljne, jer se funkcije mogu realizovati na oba nivoa, ili se realizuju na jednom od nivoa zbog performansnih razloga). Ovaj postupak čine sledeće aktivnosti:

1. Prevođenje simboličkog imena uređaja sa ciljem da se identifikuje odgovarajući drajver. Na primer, Unix simboličko ime /dev/tty0 se prevodi u ime koje pripada upravljačkom bloku, a specificira terminalski drajver uređaja.
2. Prevođenje obima (veličina) sistemskih logičkih blokova u fizičke blokove koji su po obimu prilagođeni uređaju. Ova aktivnost uključuje baferovanje, pakovanje ili raspakovanje podataka na sistemskom nivou u fizičke blokove na način kako se to zahteva od strane uređaja.
3. Alokacija i dealokacija prostora za memorisanje. Upravljanje prostorom za memorisanje, kod svih uređaja, izvodi se na način koji je nezavisan od tipa uređaja.

7.4. U/I procesi

Sve funkcije operativnog sistema se izvršavaju pomoću procesa. Za U/I proces možemo smatrati da se satoji iz dva dela (slika 7.2): deo koji se izvršava od strane CPU-a i deo koji se izvršava od strane kontrolera uređaja. Kontroler uređaja može sa CPU-ovim delom formirati jedinstveni proces (*direktni* U/I), ili formirati nezavisan proces (veći broj kontrolera uređaja je istovremeno aktivno i izvršava se paralelno sa CPU-ovim procesom). U zavisnosti od toga kakav se stepen sinhronizacije zahteva sa CPU-ovim procesom, moguće je razlikovati *preklopljeni* (overlapped) U/I, koji zahteva visok stepen sinhronizacije ili *autonomni* U/I, koji zahteva nizak stepen sinhronizacije.

Sinhronizacija između CPU-a i dela kontrolera uređaja koji se odnosi na U/I proces, ostvaruje se u obliku komunikacije, pri čemu se vrši izmena dva tipa informacije:

1. **Informacija kojom upravljamo uređajem** - čine je upravljačka i statusna informacija. Upravljačka informacija se predaje od strane CPU-a kontroleru uređaja sa ciljem da se obavi upravljanje, tj. ona postavlja U/I podsistem u određeno stanje. Deo upravljačke informacije, koja određuje stanje U/I podsistema, smešta se u jedan ili veći broj upravljačkih registara. Statusna informacija predaje se CPU-u od strane kontrolera uređaja

(nakon što je komanda prihvaćena), sa ciljem da se CPU informiše o stanju kontrolera uređaja i/ili uređaja. Ova informacija se smešta u jedan ili veći broj statusnih registara.

2. **Podaci uređaja** - ovo su podaci koji se čitaju ili upisuju iz/ka uređaju. Obično se podaci prenose preko specijalnog registra (registra za podatke) u portu "hosta". Operacija predaja podataka (upis) se koristi za prenos podataka CPU→kontroler uređaja→uređaj. Prijem podataka (čitanje) se ostvaruje u obrnutom smeru uređaj→kontroler uređaja→CPU.

Operacije koje se odnose na upravljanje i testiranje spremnosti sklopova koji učestvuju u prenosu podataka, ostvaruju se upisom informacije u upravljački registar, a nakon toga čitanjem stanja statusnog registra koji ukazuje na stanje u kome se nalazi kontroler uređaja i uređaj. U zavisnosti od upravljačke i statusne informacije, moguće je razlikovati (slika 7.2):

- **Upravljačko/statusnu informaciju porta "hosta"** - odnosi se na komunikaciju sa portom hosta. Kako po grupi povezanih uređaja postoji samo jedan port hosta, dovoljno je ugraditi jedan (skup) upravljačko/statusni registar (registara) porta hosta.
- **Upravljačko/statusnu informaciju porta uređaja** - ova informacija se direktno prenosi na priključeni uređaj. Kada je na jedan kontroler uređaja povezan veći broj uređaja, a svi ovi uređaji treba da su simultano aktivni, neophodno je da postoji veći broj upravljačko/statusnih registara portova uređaja.

Kod najvećeg broja računarskih arhitektura unapred je određen format jednog upravljčko/statusnog registra, koji predstavlja zbir upravljačko/statusne informacije kontrolera uređaja i svih priključenih uređaja. Na ovaj način, rutine za obradu prekida i izuzetaka se mogu pisati nezavisno od tipa uređaja. Treba napomenuti da se proste uređaje, kao što je štampač, zajednički upravljačko/statusni registar može biti dovoljan da čuva svu upravljčko/statusnu informaciju za port hosta i port uređaja.

7.4.1. Direktni (programirani) U/I

To je najprostiji oblik U/I-a, gde CPU i kontroler uređaja formiraju jedinstveni proces, čime se omogućava bilo kakav oblik konkurentnog rada. Sledeći program prikazuje sekvencu koraka koja je potrebna da se ostvari U/I operacija tipa upis. Koraci 1-3 formiraju petlju u kojoj CPU ispituje U/I uređaj. Ovo je neophodno zbog sinhronizacije CPU-ovog rada sa radom kontrolera uređaja koji se odnosi na taj proces.

CPU	Uređaj
(1) Čitanje statusnog registra uređaja	
	(2) Predaja statusa
(3) Ispitivanje statusa uređaja; ako uređaj nije spreman, idi na korak 1	
(4) Upiši podatak u registar podatakakoji pripada uređaju	
	(5) Prihvati podatak i postavi status na "Not Ready" sve dok se ne upišu u uređaj

Svaki korak se programira (direktni U/I se zbog toga takođe zove i *programirani* U/I) pomoću jedne ili većeg broja instrukcija kao što je prikazano na sledećem primeru.

Primer 7.1:

Sledeća programska sekvenca, napisana na asemblerskom jeziku za mikroprocesor MC68020, prikazuje generisanje niza znakova čiji je obim definisan simboličkim imenom "Broj", lociranom na adresi Pod_adr. Nakon toga se vrši štampanje. Interfejs štampača je 8-bitni. Statusni registar štampača lociran je na memorijskoj lokaciji Stampac_stat, a registar štampača za prihvatanje podataka na lokaciji Stampac_pod.

```
        MOVE.L    #Pod_adr,A0              ; A0 ukazuje na niz znakova
        MOVE.B    #Broj,D0                 ; u D0 čuva dužina niza znakova
Cekaj:  BTST.B    #Ready_bit,Stampac_stat  ; kružno ispitivanje statusa štampača i čekanje sve dok
                                           ; štampač ne bude spreman (ready) za prihvatanje podataka
        BPL       Cekaj
        MOVE.B    (A0)+,Stampac_pod        ; kopiraj podatak u registar
                                           ; podataka štampača i štampaj
        DBNE      D0,Cekaj                 ; dekrementiraj D0, ako je (D0)≠$0
                                           ; vrti se u petlji sve dok se ne odštampa ceo niz
```

Kod direktnog U/I-a potrebno je da postoje U/I instrukcije pomoću kojih se vrši pristup upravljačko/statusnom registru i registru za podatke. Ove instrukcije omogućavaju da CPU komunicira sa

kontrolerom uređaja. Pomoću ovih instrukcija specificira se operacija koja treba da se obavi, port hosta i port uređaja (često zajednički nazvani adresa uređaja) i obično CPU-ova adresa (koja je izvorište/odredište za U/I podatke). Za specifikaciju svake od ove tri komponente, postoji nekoliko alternativa. U daljem tekstu ukazaćemo na ove alternative.

7.4.2. Specifikacija U/I operacija

U/I operacija se može specificirati pomoću opkoda ili prostora za operand. Kada se za specifikaciju U/I operacije koristi opkod prostor (u zavisnosti od toga koliki je opkod prostor dostupan i koliko U/I operacija želimo da uvedemo u specifikaciju skupa naredbi) razlikujemo sledeće alternative:

- opkod prostorom se specificira da instrukcija bude U/I instrukcija i da se tačno, ona operacija koja se obavlja (obično zavisna od tipa uređaja) specificira u polju opkod-proširenje.
- opkod prostor specificira U/I operaciju. Ovo znači da su U/I operacije deo CPU-ove arhitekture i da svi uređaji treba da manipulišu svim operacijama na uniforman način. IBM/370, koja je kanalski orijentisana arhitektura, koristi ovaj metod. Za operacije koje zahtevaju realizaciju prenosa podataka (kod arhitekture IBM/370) uvedeno je polje za opkod proširenje, kojim se specificira U/I instrukcija. U/I instrukcije treba da su privilegovanog tipa.

Za specifikaciju U/I operacije može se koristiti i prostor operanda na taj način što se rezerviše deo ovog prostora za U/I. Za pristup upravljačko/statusnom ili registrima za podatke koriste se memorijske adrese. Naime, dolazi do preklapanja memorijskih lokacija (adresa) sa pomenutim registrima. Ova šema se zove *memorijsko preslikani* U/I (memory mapped I/O). Prednost memorijsko preslikane U/I tehnike je u tome što se sve instrukcije iz skupa naredbi CPU-a mogu koristiti za pristup U/I-u, a to se plaća malim gubitkom raspoloživog adresnog prostora glavne memorije. U/I operacije se mogu učiniti da budu privilegovanog tipa korišćenjem mehanizma zaštite memorijske oblasti koja se dodeljuje U/I-u. Skoro sve nove arhitekture, koje obično imaju 32-bitne adrese, koriste memorijsko preslikani U/I.

7.4.3. Specifikacija porta hosta i porta uređaja

Za specifikaciju porta hosta drajver uređaja ne postavlja nikakva ograničenja. Specifikacija se ostvaruje korišćenjem polja za definiciju neposrednog podatka, ili se (specifikacija) određuje na osnovu adrese operanda.

Kada je port hosta u stanju da upravlja većim brojem uređaja, U/I instrukcije moraju imati mogućnost specifikacije određenog porta uređaja. Na ovaj način se može proizvoljno pristupati upravljačkim/statusnim registrima pojedinih uređaja. Drajveri uređaja (device driver) (slika 7.4) projektovani su tako da omoguće rad sa svim uređajima određenog tipa koji su pridruženi jednom host portu. Da bi bili u stanju da koristimo isti programski kod kojim se rukuje bilo kojim pridruženim uređajem, specifikacija porta uređaja treba da se obavlja u toku izvršenja programa, tj. ako se ne sme ugraditi u programski kod (u toku kompilacije). Specifikacija porta uređaja može se izvesti na sledeće načine:

- dostupni su svi portovi- dostupni su upravljačko/statusni registri portova uređaja od svih uređaja (usvojimo da je na kontroler uređaja priključeno nekoliko uređaja), na primer, svi su memorijsko preslikani. Drajver uređaja može postaviti baznu adresu koja će pokazivati na blok upravljačko/statusnih registara pojedinog uređaja i koristiti način adresiranja sa baznim razmeštajem kada treba da pristupi određenom upravljačkom/statusnom registru.
- dostupan je samo jedan port - upravljačko/statusni registri porta uređaja, koji pripadaju jednom portu dostupni su u bilo koje vreme. Pojedini port uređaja se može birati preko upravljačkog registra u portu hosta. Izbor upravljačko/statusnog registra pojedinog porta uređaja ostvaruje se preko polja za opkod proširenje, ili preko memorijske adrese kada se koristi memorijsko preslikavanje.

7.4.4. Specifikacija CPU-ove adrese

Kada se obavlja prenos između CPU-a i U/I sistema, neophodno je specificirati internu CPU-ovu lokaciju koja može biti izvorište/odredište informacije. Kod ovakvog tipa prenosa javlja se jedan ozbiljan problem koji se ogleda u sledećem: Brzina rad CPU-a je obično mnogo veća od brzine rada U/I uređaja, tako da će CPU gubiti mnogo vremena ako stalno u petlji ispituje status uređaja (status ukazuje na spremnost za prenos), čime se ostvaruje sinhornizovani prenos informacije. Ako je sistem jednokorisnički (kao što je personalni računar koji radi pod MSDOS-om) ovakav način prenosa može biti prihvatljiv za korisnika, ali verovatno ne može biti prihvatljiv za korisnike multiprogramskih ili "time sharing" sistema, kod kojih se zahteva da se CPU-ova obrada preklapa (bude paralelna) sa U/I-om.

7.4.5. Preklapanje aktivnosti U/I-a i CPU-a (prekidna U/I tehnika)

Kod ove tehnike mehanizam rada sa direktnom sinhronizacijom (tzv. programirana U/I tehnika) koji se standardno realizuje izpitivanjem statusa uređaj u programskoj petlji (tj. predstavlja oblik "busy waiting") se zamenjuje mehanizmom prekida. Umesto da se u petlji stalno vrši kružno ispitivanje statusa spremnosti uređaja, CPU se oslobađa da može izvršavati neki drugi proces, dok proces dodeljen uređaju može takođe konkurentno da obavlja svoju aktivnost. Kada proces dodeljen uređaju treba, radi opsluživanja, da se sinhronizira sa odgovarajućim CPU-ovim procesom (obično je to prenos podataka), on signalizira CPU-u da zahteva opsluživanje preko prekida. Na ovaj način, pored mogućnosti za programiranim U/I, zahteva se i mehanizam prekida. Port hosta može da prekine rad CPU-a iz sledećih razloga:

- U/I uređaj je spreman za prenos narednog podatka,
- operacija je završena pa se nakon toga generiše prekid,
- signalizira se izuzetak, koji može ukazivati na: nevažeću komandu ili parametar, grešku u podatku ili grešku uređaja.

Prednost U/I tehnike sa preklapanjem (obično se u literaturi zove još i *prekidna* U/I tehnika; tj. overlapped I/O ili interrupt I/O) se satoji u tome što CPU nije kontinualno zauzet izvršenjem jednog U/I procesa određenog uređaja. CPU komutira na U/I proces samo u sinhronizacijućim periodima tj. u trenucima kada se obavlja prenos podataka. Ovi trenuci predstavljaju relativno kratak vremenski period, tako da je i drugim procesima dozvoljeno da (uključujući i U/I proces) se mogu izvršavati od strane CPU-a. Veliki broj U/I uređaja mogu na ovaj način da rade konkurentno. Nedostatak prekidne U/I tehnike ogleda se u tome što se u toku svakog prenosa podataka zahteva sinhronizacija procesa, a to može dovesti do toga da CPU u značajnoj meri izgubi svoje dragoceno vreme, jer se javlja potreba za komutacijom procesa, koja obično obuhvata izvršenje nekoliko stotina instrukcija.

7.4.6. Autonomni prenos podataka (Direktni memorijski pristup)

Potrebe za učestalom sinhronizacijom rada procesa, a shodno tome i izgubljeno CPU-ovo vreme zbog komutacije, može se u značajnoj meri smanjiti ako se sinhronizacija kod prenosa podataka ostvaruje na nivou blokova podataka, a ne na nivou jednog podatka. Veličina blokova obično varira od 80 znakova za matrični štampač do 512 znakova ili nekoliko kilobajtova za disk. Ovakav prenos rezultira da se vremensko prekoračenje zbog sinhronizacije reducira za dva do tri reda veličine. Mehanizam rada koga je potrebno ostvariti da se od strane U/I podsistema podržava autonomni prenos podataka zovemo *direktni memorijski pristup* (DMA).

Primer 7.2:

ledeća programska sekvenca koristi DMA za štampanje linije teksta. Ova aktivnost se obavlja iniciranjem DMA kontrolera u host portu sa neophodnim parametrima. Ovi parametri predstavljaju vrednosti koji se pune u Pod_adr (adresa registra za podatke koji se koristi za čuvanje CPU-ove adrese o podatku koji se prenosi) i Broj_reg (brojački registar koji specificira obim bloka).

```
MOVE.L      #Vred_adr,Pod_adr       ; Pod_adr ukazuje na znak u nizu
MOVE.B      #Broj,Broj_reg          ; Broj_reg sadrži dužinu niza
IOR.B       #Start,Stampac_start    ; start štampača i generisanje prekida
                                    ; kada se niz odštampa
```

Direktni pristup memoriji

DMA rad je analogan izvršenju "Multiple Data Move" instrukcije. Glavna razlika je u tome što se DMA izvršava od strane porta hosta a ne od strane CPU-a. Operandi koji su neophodni za DMA se zovu DMA parametri. Ovi parametri se smeštaju u portu hosta a to su oni isti koji su potrebni kao i kod kopiranja bloka (vidi pvi primer u ovoj glavi):

- CPU-ova memorijska adresa je adresa glavne memorije ka ili iz koje se podatak može preneti (upisivati/čitati).
- dužina bloka je broj, koji predstavlja veličinu oblasti glavne memorije u/iz koje treba preneti (upisivati/čitati) podatak.
- adresa uređaja je adresa U/I uređaja za memorisanje. Za slučaj disk drajva, ovu adresu čIne disk, pista i broj sektora. Za slučaj da je to primopredajni uređaj (kao što je štampač) adresu uređaja nije potrebno specificirati jer se podatak ne memoriše; to je niz koji se unosi ili generiše na izlazu kontrolera uređaja preko registra za podatke.

Izvršenje DMA operacije za štampač (vidi prethodni primer) se odvija na sledeći način:

1. vrši se prenos DMA parametara ka portu hosta i startuje se štampač.
2. kako je registar podatak štampača prazan, port uređaja zahteva da se u registar podataka upiše jedan znak, čime su ostvareni uslovi da štampanje može početi; ovo se izvodi preko DMA kanala.

3. DMA kanal u portu hosta zahteva upravljanje nad sistemskom magistralom, i kada se magistrala od strane CPU-a dodeli DMA operacija može da počne.
4. DMA operaciju kojom se vrši punjenje registra za podatke čine sledeći koraci:
   a) postavlja se memorijska adresa na sistemsku magistralu,
   b) obavlja se operacija čitanje memorije,
   c) upisuje se pročitani podatak u registar podataka,
   d) oslobađa se sistemska magistrala,
   e) povećava se memorijska adresa kako bi moglo da se ukaže na naredni znak u nizu,
   f) smanjuje se brojač.
5. Kod narednog zahteva za prenos podataka, akcije koje se preduzimaju zavise od vrednosti brojača:
   a) ako je Brojač≠$0, sledi korak 3
   b) ako je Brojač=0, zaustavlja se štampač i generiše prekid pomoću koga se signalizira da je operacija štampanja završena.

U zavisnosti od toga koliki je broj prenetih podataka po DMA operacji, razlikujemo dva tipa DMA:

1. **DMA sa jednostrukim ciklusom** - obavlja se samo jedan prenos u toku jedne DMA operacije (korak 3). Prednost ovog tipa DMA se sastoji u tome što se sistemska magistrala prepušta od strane CPU-a drugom gospodaru magistrale za vrlo kratak vremenski period. Nedostatak ovog pristupa je taj što se kod svakog prenosa javlja potreba za arbitriranjem, tj. odlučivanjem ko će u narednom trenutku da koristi magistralu (mehanizam arbitraže nalaže ugrađivanje složene logike, tj. hardvera).
2. **Paketni (burst) DMA** - u toku jedne DMA operacije vrši se prenos većeg broja podataka. Da bi se ostvario ovakav prenos, kontroler uređaja mora biti u stanju da baferuje (privremeno čuva) podatke. Prednost paketnog DMA ogleda se u većoj brzini i manjoj ceni arbitražne logike. Nedostatak se satoji u tome što se za duži period zauzima sistemska magistrala (CPU i drugi DMA kanali su tada pasivni), i većoj ceni koja se mora platiti zbog toga što kontroler uređaja mora da ima ugrađen hardver za baferovanje.

Povezivanje podataka

Najveći broj sistema za upravljanje memorijom alociraju/dealociraju memoriju u blokovima određene veličine (na primer, 512 bajtova ili više) koji se zovu stranice. Veličina stranice ne mora biti istog obima kao blokovi podataka uređaja za memorisanje. Kod jeedinice magnetne trake, zbog relativno dugačkih vremena potrebnih da se traka pokrene i zaustavi, težnja je da se formiraju duži blokovi (nazvani zapisi). U toku vremena pokretanja, na traci koja prolazi pored glave za upis/čitanje nije moguće vršiti upis/čitanje. Ovaj deo trake je poznat kao međuzapisni gap (praznina). Ako su zapisi duži iskorišćavanje trake je bolje.

Veličina sektora na disku takođe ne mora biti ista kao i veličina stranice. Na izbor veličine stranice utiče veliki broj faktora. U daljem tekstu analizirajmo slučaj kada je upis sektora na disku potrebno upisati P stranica. Jedan od načina da se organizuje upis P stranica je da se vrši upis stranice pod DMA kontrolom pa da se nakon toga generiše prekid i postave DMA parametri koji se odnose na narednu stranicu. Ako ne postoji baferovanje podatka u kontroleru uređaja, vreme potrebno da se opsluži prekid i postave naredni DMA parametri jednako je vremenu prenosa samo jednog elementa podatka, koje je reda 1μs, tj. veoma je kritično. Jedno od rešenja ovog problema ogleda se u ugradnji bafera veličine sektora (baferski stepen se ugrađuje u drajver uređaja) tako da veličina bloka U/I operacije odgovara veličini bloka uređaja. Drajver uređaja kopira P stranica ka/iz ovog bafera, dok se u bafer može upisivati/čitati ka/iz diska. Operacija kopiranja dovodi da CPU gubi dosta vremena. Varijanta ovog rešenja se ogleda u instalaciji bafera u kontroleru uređaja. Port hosta može nakon toga puniti ili prazniti bafer pod DMA kontrolom, i to jednu stranicu po DMA ciklusu. Na ovaj način CPU se rasterećuje od kopiranja. Nedostatak obe metode ja taj što za određene uređaje (na primer trake), veličina zapisa, koja određuje veličinu bafera, može varirati u velikoj meri tako da je često potrebno ugraditi dva bafera (dok se jedan bafer puni od strane uređaja, drugi se prazni od strane CPU-a).

Elegantnije rešenje, kod kojeg problemi koji su vezi sa kritičnim vremenom obrade prekida, veličine bafera i CPU-ovog gubitka vremena nisu tako izraziti, ogleda se u *povezivanju podataka* (data chaining). Osnovna ideja se satoji u tome da DMA kanal sam sebi postavlja DMA parametre, pa se na taj način uspešno rešava problem koji je u vezi sa veličinom sektora ili stranica. Da bi ovakav rad bio moguć, u port uređaja je potrebno ugraditi: Data Chain Register (DCR) i registar u kome se čuvaju DMA parametri. DCR ukazuje na lanac P blokova DMA parametara, lociranih u glavnoj memoriji. Obično, marker, nazvan DCF (Data Chain Flag), u DMA parametarskom bloku ukazuje kada povezivanje (lančanje) može da produži. Najveći broj savremenih arhitektura podržava rad povezivanja podataka, čime se ostvaruje nezavisnost između veličine stranice sa jedne, i veličine sektora na disku ili zapisa na traci sa druge strane. DMA operacija, koja koristi povezivanje obavlja se na sledeći način:

1. Za prenos DMA parametara koristi se DMA mehanizam. Za pristup DMA parametarskom bloku lociranom u glavnoj memoriji koristi se DCR.
2. Za prenos bloka podataka ka/iz U/I uređaja koristi se DMA mehanizam. Pod_adr (vidi prvi primer) se koristi za adresiranje podataka u glavnoj memoriji.

3. Nakon što se obavi prenos jednog bloka podataka, izvršava se jedna od sledećih akcija:
   a) postavlja se DCF u bloku partametara DMA, tako da se obezbedi prenos još jednog bloka podataka, prelazi se na korak (1);
   b) briše se DCF, DMA operacija je završena, i generiše se prekid.

Kada se povezivanje podataka koristi za operaciju čitanje, jedan blok memorisanih podataka (na disku ili traci) se razbija na nekoliko memorijskih stranica, pa se često ova operacija zove čitanje sa rasipanjem (scatter read operation). Kod operacije upis, vrši se prikupljanje nekoliko stranica sa ciljem da se izvrši njihov upis kao jedinstveni blok na uređaj za memorisanje (disk ili traka), pa je često ova operacija poznata kao prikupljanje radi upisa (gather write operation).

Autonomno upravljanje prenosom

Učestanost procesne sinhronizacije se može smanjiti, ali ono što je još važnije, propusnost na nivou autonomnosti U/I podsistema se može povećati ako se omogući da port hosta pribavi svoju upravljačku informaciju u obliku upravljačkog bloka. Pribavljanje upravljačkog bloka može se ostvariti pod kontrolom DMA, ako se koristi mehanizam iskorišćen za pribavljanje DMA parametara za povezivanje podataka. Ovo se zove *povezivanje komandi* (command chaining). Da bi ovaj mehanizam rada mogao da se realizuje neophodno je da se DMA mehanizam proširi na taj način što će se koristiti CCR (Command Chain Register), koji pokazuje na lanac komandi, i CCF-a (Command Chain Flag), koji ukazuje kada povezivanje komandi treba da se nastavi.

Kod korišćenja povezanih komandi, drajver uređaja je taj koji treba da pridruži svaku novu U/I komandu na kraju komandnog lanca. Lanac mogu da čine povezane liste upravljačkih blokova koji sadrže zahtevane U/I komande. Umesto da se čeka sve dok se sve U/I komande ne izvrše, određeni proces može da nastavi sa radom kada se njegova U/I komanda završi. U/I podsistemi koji podržavaju povezivanje (lančanje) komandi u svojim upravljačkim blokovima, poseduju markere koji ukazuju na to da će se generisati prekid nakon završetka te komande.

7.4.7. Struktura kontrolera uređaja

Ukažimo sada na neke aspekte kontrolera uređaja kroz rad host porta i nivoa njegove intelegencije, a nakon toga na rad porta uređaja.

Port hosta

Port hosta čine: hardver za dekodiranje U/I komandi, sklop za selekciju porta, upravljačko/statusni registri i hardver za prekid. Pored ovoga, u port hosta može biti ugrađena logika za podršku autokonfiguracije sistema, a to znači da se konfiguracija sistema može programski odrediti ako se ispitaju svi portovi hosta. Svaki port hosta treba, zbog toga, da sadrži identifikaciju tipa U/I podsistema, verziju, konfiguraciju (broj povezanih uređaja) i ažurirani status. Ovo stanje se može pročitati iz EPROM-a ili iz nekog porta na koga su povezani prekidači. Stanje prekidača korisnik postavlja u određeni položaj. Fleksibilnost dodele prioriteta obezbeđuje se programski.

Baferovanje

Baferovanje je funkcija koja se obavlja na implementacionom nivou, pa se zbog toga u kontroler uređaja ugrađuje baferski mehanizam. Ovaj mehanizam se koristi sa ciljem da se učini U/I proces manje kritičan sa stanovišta vremenskog izvršenja, a takođe i da se poboljšaju performanse sistema, koje se odnose na prenos podataka. Uređaji za memorisanje, kao što su jedinice diska, tipično, u toku prenosa podataka rade sa fiksnom brzinom. Kada se izvršava operacija upis, kontroler uređaja mora da predaje te podatke sa zahtevanom brzinom. Ako zbog sistemskog opterećenja, ova brzina ne može da se ostvari, kažemo da se javlja "under-run" greška (na disk treba da se upiše podatak koji još nije dostupan). Suprotan tip greške je "over-run error" (u registru za podatke pre nego što se prethodni podatak memoriše upisuje se novi).

Uvođenjem bafera, koji se sa programske tačke gledišta vidi kao deo memorije ugrađen u kontroler uređaja, postignuto je da prenos informacije bude manje kritičan ili ne bude vremenski kritičan, zbog toga što se sektori prvo upisuju u lokalni bafer. Na ovaj način se reduciraju ili eliminišu vremensko kritični zahtevi za sve oblike U/I-a (direktni U/I, prekidni U/I i autonomni prenos podataka).

Neinteligentni kontroleri uređaja

Kontroleri uređaja koji se koriste za direktni ili prekidni U/I obično ne zahtevaju lokalnu intelegenciju. Kontroler može biti veoma jednostavan a obično ga čine: interfejs sistemske magistrale, zajednički upravljačko/statusni registar i registar za podatke. Sva inteligencija je locirana u U/I softveru.

Inteligentni kontroleri uređaja

Inteligentni kontroleri uređaja obično se karakterišu moćnijom obradom i većim kapacitetom baferovanja (lokalna memorija) u odnosu na neinteligente kontrolere uređaja. Veća moć obrade ukazuje na to da se mogu izvršiti neke softverske funkcije, pa se na taj način rasterećuje CPU i postiže konkurentni rad.

Najjednostavniji oblik konkurentne operacije je mogućnost autonomnog prenosa podataka, korišćenjem DMA. Ovakav pristup zahteva da se u kontroleru uređaja ugradi malo inteligencije. Najsavremeniji kontroleri uređaja su kanali i U/I procesori.

Kanali - koriste se od strane arhitekture IBM/370. Kanali izvršavaju kanalske programe koji se sastoje od sekvence kanalskih komandi čime se ostvaruje povezivanje podataka i komandi. Kanalski programi i njihova mogućnost baferovanja imaju glavni doprinos smanjenju suvišnog utroška vremena i resursa prilikom U/I operacija i igraju važnu ulogu da se U/I aktivnost učini manje kritičnom sa stanovišta vremenskog izvršenja. Zbog svog uniformnog i konzistentnog interfejsa kanali olakšavaju implementaciju drajvera uređaja. U zavisnosti od mogućnosti multiprogramiranja porta hosta(nazvan kanal od strane IBM-a) razlikujemo dva tipa kanala:

1. **Selektorski kanali** - ovi kanali u toku trajanja operacije ostaju logički povezani na port uređaja (nazvan od strane IBM-a kao podkanal). Obično se koriste za povezivanje uređaja koje karakteriše velika brzina prenosa (na primer, jedinice diska ili magnetne trake) i koriste paketni DMA. Kada se selektorski kanal logički poveže na port uređaja, on je zauzet i nije sposoban prihvati komande za bilo koji drugi port uređaja koji je povezan na taj port hosta.
2. **Multiplekserski kanali** - ovi kanali dozvoljavaju da se istovremeno obavlja nekoliko U/I operacija u kojima učestvuje i port hosta. U ovom slučaju port hosta mora za svaki port uređaja da poseduje skup upravljačko/statusnih registara i registre za podatke. Kada više portova uređaja i uređaja istovremeno obavlja U/I operacije, drugi portovi uređaja i uređaji se mogu selektovati radi startovanja nove operacije. Kada nekoliko uređaja vrši prenos podataka, DMA hardver host porta se multipleksira po ceni redukovanja brzine prenosa podataka po uređaju. Zbog toga na multiplekser se obično povezuju uređaji koji prenose podatke sa manjim brzinama.

7.5. U/I procesori

Niska cena mikroprocesora otvorila je put ka povećanju inteligencije (funkcionalnosti) U/I podsistema. Umesto da se koriste jednostavne U/I komande (odgovaraju fizičkom nivou) koriste se logičke i/ili simboličke U/I komande visokog nivoa. veliki broj funkcija drajvera uređaja (obrada grešaka i optimizacija vremena pristupa) može da izvrši mikroprocesor U/I podsistema, nazvan U/I procesor (IOP).

Kada se obrada greške vrši od strane IOP-a, postiže se značajno poboljšanje rada sistema u odnosu na to kada bi tu grešku obrađivao drajver uređaja. U IOP je ugrađen znatno veći iznos specifičnog interfejsa, pa se zbog toga uređaj može nadgledati i upravljati na pouzdaniji, tačniji i direktniji način.

Funkcije kao što su softver koji je nezavisan od tipa uređaja, na primer alokacija/dealokacija memorisane informacije, i preslikavanje logičkih blokova u fizičke blokove i obratno, mogu se takođe ostvariti pomoću IOP-a.

Tehnologija je danas toliko napredovala da se DMA, povezivanje komandi i podataka, zajedno sa velikim brojem funkcija koje se odnose na port uređaja, integrišu u jedinstveni čip.

7.6. Port uređaja

Port uređaja upravlja uređajem i vrši prenos podataka ka/iz uređaja. Portovi uređaja se ne mogu deliti ili multipleksirati, pa se zbog toga uvek samo po jedan uređaj povezuje na port uređaja (slika 7.2). Način spajanja ovog porta i interfejsa uređaja ostvaruje se preko namenskih linija. Neke od upravljačkih funkcija uređaja, koje se obavljaju od strane porta uređaja, su:

- startovanje i zaustavljanje uređaja,
- inicijalizacija uređaja,
- upravljanje okruženjem
- konverzija električnih parametara,
- prepoznavanje adrese kod komunikacionih linija.

Neke od funkcija koje se odnose na prenos podataka a obavljaju se od strane porta uređaja su:

- konverzija podataka iz paralelnog u serijski oblik i obratno,
- sastavljanje/rastavljanje reči na bajtove,
- protokol za prenos podataka,
- provera parnosti ili CRC (*Cyclic Redundancy Check*).