

Organizacija i arhitektura računala

Poglavlje 6

Input/Output sustavi

I/O problemi...

- ⌘ Velika raznovrsnost periferije
 - ☒ Raznovrsne brzine prijenosa podataka
 - ☒ Na raznim komunikacijskim brzinama
 - ☒ U različitim formatima
- ⌘ Sve su sporije od CPU-a i RAM-a
- ⌘ Potrebne su I/O jedinice

I/O jedinice/moduli

- ⌘ to su sučelja prema CPU i memoriji
- ⌘ isto tako to su sučelja i prema više od jedne IO jedinice

Vanjski uređaji

⌘ orijentirane prema čovjeku

☒ Monitor, pisač, tipkovnica

⌘ orijentirane prema stroju

☒ nadzor i kontrola/upravljanje

⌘ komunikacija

☒ modem

☒ Network Interface Card (NIC)

I/O funkcije modula

- ⌘ Control & Timing
- ⌘ CPU komunikacija
- ⌘ komunikacija sa jedinicama
- ⌘ Data buffering
- ⌘ detekcija pogreške

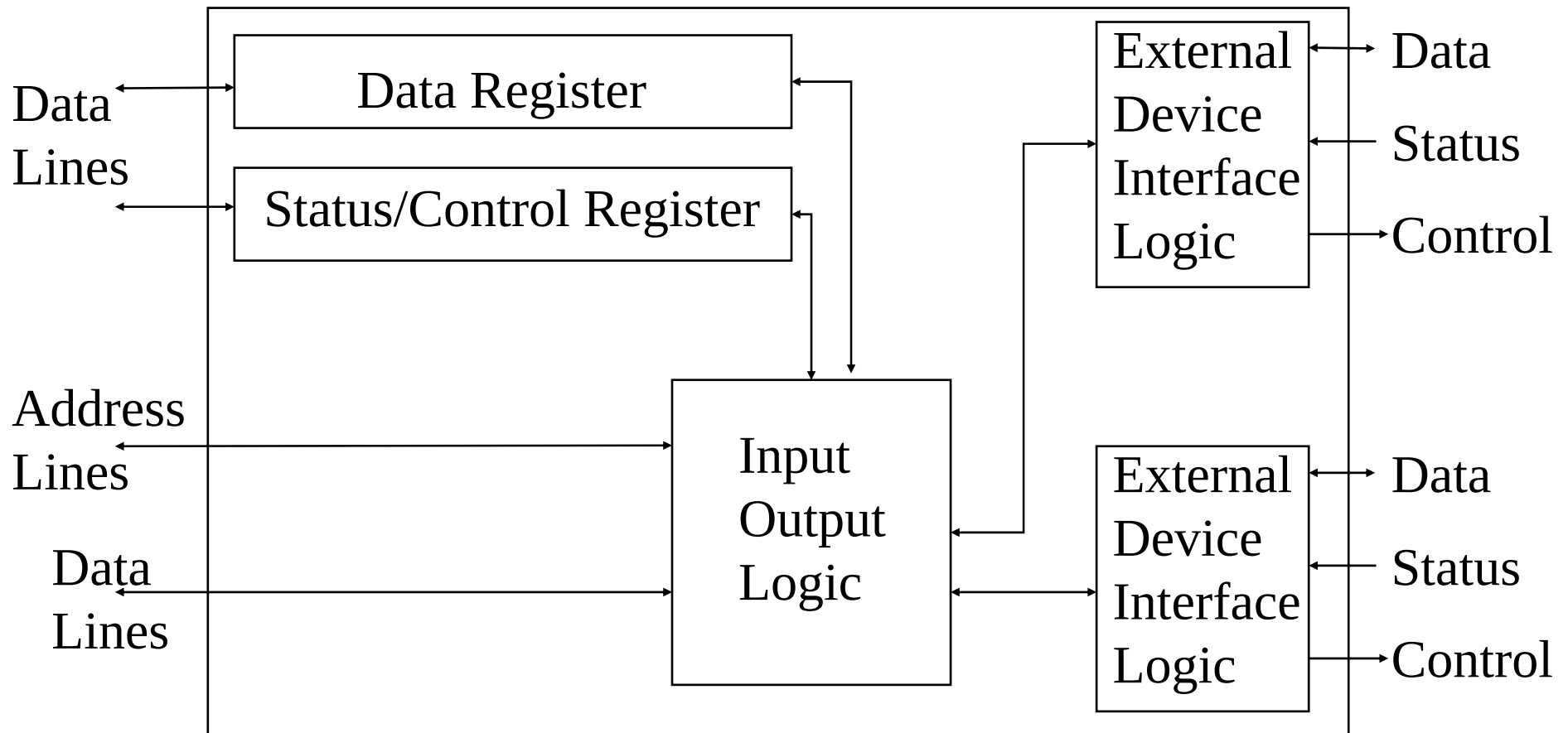
I/O koraci...

- ⌘ CPU provjerava status modula I/O jedinice
- ⌘ I/O modul odgovara svojim statusom
- ⌘ Ako je modul spreman tada CPU zahtijeva razmjenu podataka
- ⌘ I/O modul dobiva informacije od jedinice
- ⌘ I/O modul proslijeđuje dobivene podatke ka CPU-u
- ⌘ mogu postojati varijacije u koracima, npr. DMA prijenos podataka

Dijagram I/O modula

Systems Bus Interface

External Device Interface



Odluke I/O modula

- ⌘ Skriti ili otkriti sva svoja svojstva CPU-u?
- ⌘ odluka da li podržava samo jedan ili više I/O uređaja
- ⌘ upravlja funkcijama jedinice ili to ostavlja procesoru da obavi
- ⌘ isto tako to može prepustiti i operativnom sustavu
 - ☑ npr. Unix sve I/O jedinice tretira kao običan file/datoteku

I/O tehnike

- ⌘ Programirljive
- ⌘ pogonjene programskim prekidima
- ⌘ Direct Memory Access (DMA)

Programirani I/O

⌘ CPU ima izravnu kontrolu nad I/O

- ☐ provjerava status

- ☐ Read/write komande

- ☐ prijenos podataka

⌘ CPU čeka I/O modul da završi operaciju

⌘ ovo bespotrebno troši dragocjeno vrijeme CPU-a

Programirani I/O – detaljnije...

- ⌘ CPU zahtijeva I/O operaciju
- ⌘ I/O modul izvršava operaciju
- ⌘ I/O modul postavlja status bit
- ⌘ CPU periodički provjerava statusni bit
- ⌘ I/O modul ne izvještava direktno CPU o svom stanju
- ⌘ I/O modul ne djeluje preko interrupta
- ⌘ CPU može čekati ili provjeriti kasnije da li je modul završio sa svojim poslom

I/O komande

⌘ CPU adresira modul

- ☒ identifikira modul

⌘ CPU postavlja komandne zahtjeve

- ☒ kontrola – kaže modulu što da učini

 - ☒ npr. zavrti disk

- ☒ Testira – provjerava status

 - ☒ npr. napajanje? greška?

- ☒ Čita/piše

 - ☒ Modul prebacuje podatke putem posebne buffer memorije u I/O jedinicu

Adresiranje I/O uređaja

- ⌘ Gledano od strane CPU-a to je isto kao i adresiranje memorije
- ⌘ Svaki uređaj ima jedinstveni identifikator
- ⌘ CPU komande sadrže i te identifikatore

I/O mapiranje

⌘ Memorijski mapirani I/O

- ☒ Uređaji i memorija dijele isti adresni prostor
- ☒ I/O se pomaša kao obična memorijska lokacija
- ☒ nema posebnih komandi za I/O operacije

⌘ Izolirani I/O

- ☒ Odvojeni adresni prostor
- ☒ Potrebna dodatna I/O ili memory select linija
- ☒ Specijalne komande za I/O operacije
 - ☒ Ograničeni skup takvih komandi

Interruptom pogonjeni I/O

- ⌘ rješava probleme čekanja CPU-a
- ⌘ Nema provjere stanja uređaja od strane CPU-a
- ⌘ I/O modul izaziva interrupt kada je on spreman

Interruptom pogonjeni I/O

Osnovne operacije

- ⌘ CPU zahtijeva komandu čitanja
- ⌘ I/O modul uzima podatak od svoje periferije dok procesor radi nešto drugo
- ⌘ I/O modul prekida CPU interruptom
- ⌘ CPU traži prijenos podataka
- ⌘ I/O modul prenosi podatke

Gledano od strane CPU-a...

- ⌘ zahtijeva komandu čitanja podataka
- ⌘ Radi nešto drugo
- ⌘ Provjerava pojavu interrupta
- ⌘ Ako je interrupt:
 - ☒ Snima trenutno stanje (registre)
 - ☒ obrađuje interrupt
 - ☒ zahvaća i sprema podatke

Problemi kod izgradnje takvog sustava...

- ⌘ Kako otkriti koji je modul proizveo interrupt ako ima više takvih modula?
- ⌘ Kako obraditi višestruke interrupte kada se pojave?
 - ☐ npr. jedan je interrupt prekinut drugim, itd...

Modul za identifikaciju interrupta (1)

⌘ Druga interrupt linija za svaki modul

☑ PC

☑ Limitiran broj uređaja

⌘ Softversko prozivanje i ispitivanje

☑ CPU pita svaki modul za interrupt

☑ Sporoooo....

Modul za identifikaciju interrupta (2)

⌘ Daisy Chain ili hardversko prozivanje

- ☑ Interrupt Acknowledge šalje se putem interrupt lanca
- ☑ Modul postavlja adresu svog interrupt vektora na sabirnicu
- ☑ CPU analizira interrupt vektor i identificira izvor interrupta

⌘ Bus Master

- ☑ Modul mora prvo preuzeti kontrolu nad sabirnicom pa tek onda izazvati interrupt
- ☑ npr. PCI ili SCSI

Višestruki interrupti

- ⌘ Svaka interrupt linija ima svoj prioritet
- ⌘ Interrupti višeg prioriteta mogu prekinuti interrupte nižeg prioriteta
- ⌘ Ako se koristi tehnika bus masteringa/preuzimanje kontrole sabirnice, tada onaj tko je preuzeo sabirnicu može izazvati interrupt

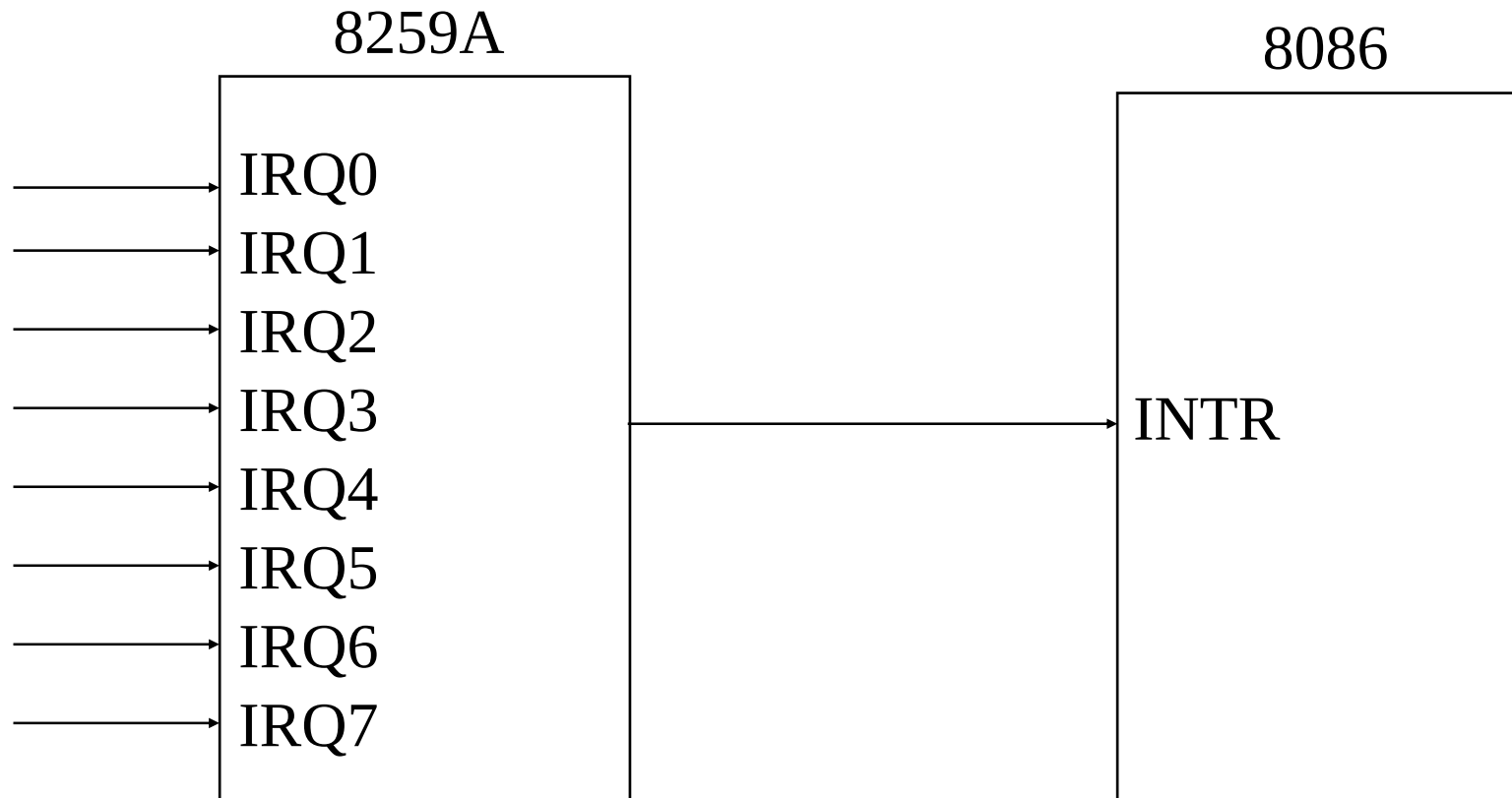
Primjer - PC Sabirnica

- ⌘ 80x86 ima jednu interrupt liniju
- ⌘ 8086 sustavi rabe 8259A interrupt kontroler
- ⌘ 8259A ima 8 interrupt linija

Sekvenca događaja

- ⌘ 8259A registrira i prihvata interrupt
- ⌘ 8259A određuje prioritet interrupta
- ⌘ 8259A signalizira 8086 (podiže INTR liniju)
- ⌘ CPU potvrđuje prijem interrupta
- ⌘ 8259A postavlja odgovarajuću adresu interrupt vektora na sabirnicu
- ⌘ CPU procesira interrupt

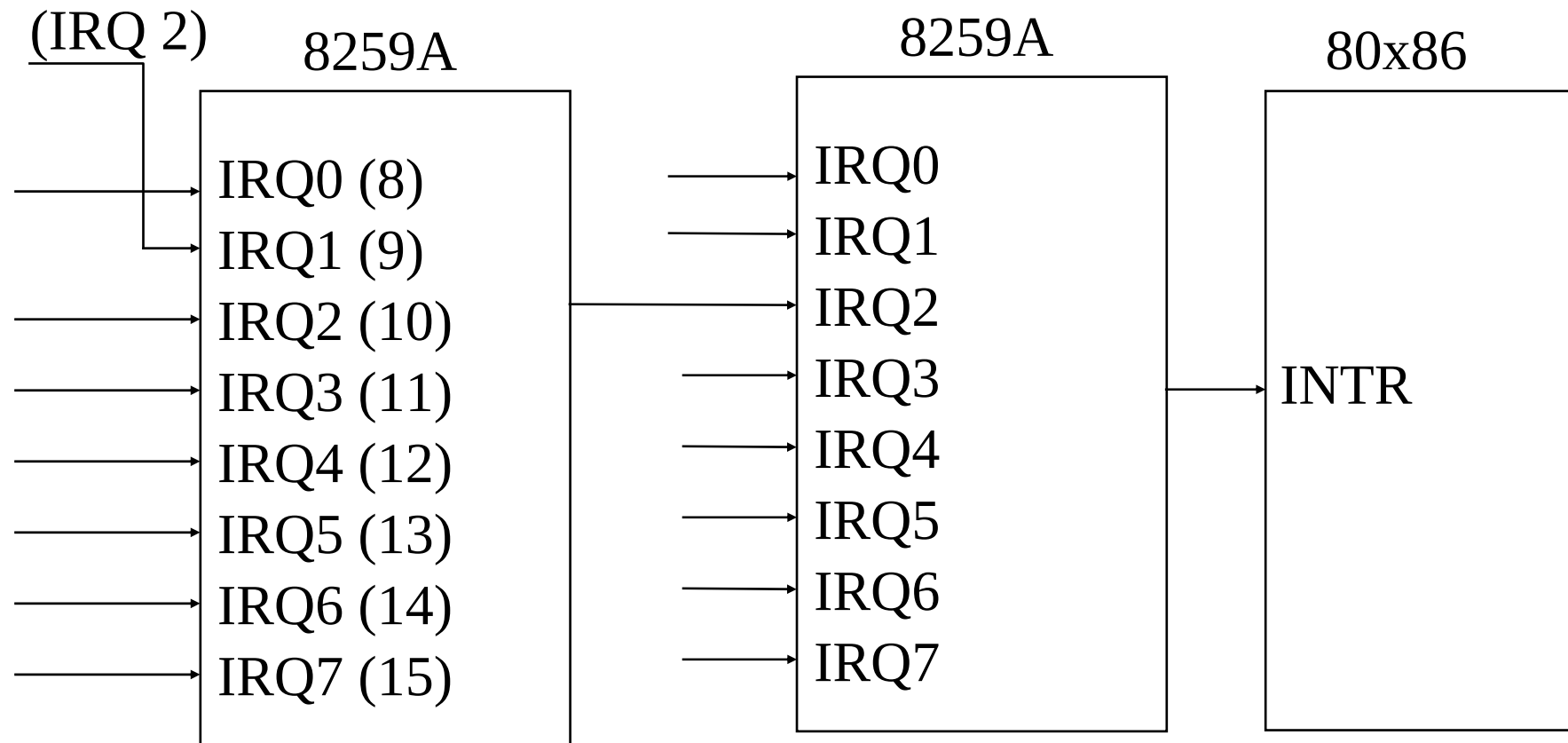
PC Interrupt Layout



ISA Bus interrupt sistem

- ⌘ ISA ulančuje dva 8259As kontrolera zajedno
- ⌘ povezuje ih putem interrupta IRQ2
- ⌘ Daje ukupno 15 linija interrupata
 - ☒ 16 linija minus jedna za uspostavljanje veze među njima
- ⌘ IRQ 9 se koristi za preusmjeravanje korištenja IRQ2
 - ☒ da bi se postigla unazadna kompatibilnost
- ⌘ Danas je to inkorporirano u sam čipset

ISA interrupt system



DMA

Direct Memory Access

- ⌘ Interruptom pogonjen i programiran I/O prijenos podataka inače zahtijeva aktivnost CPU-a
 - ☒ Brzine prijenosa su limitirane
 - ☒ CPU je tada zauzet
- ⌘ DMA predstavlja rješenje ovih problema

DMA funkcija

- ⌘ dodatan modul/hardware na sistemskoj sabirnici
- ⌘ DMA preuzima kontrolu nad I/O od CPU-a

DMA operacije

⌘ CPU “kaže” DMA kontroleru:

- ☑ Čitaj/piši

- ☑ adresu uređaja

- ☑ Početnu adresu memorijskog bloka podataka

- ☑ koju količinu podataka treba prenijeti

⌘ CPU nastavlja sa drugim poslom

⌘ DMA kontroler obavlja prijenos podataka

⌘ DMA kontroler šalje interrupt kada je gotov sa prijenosom

DMA prijenos

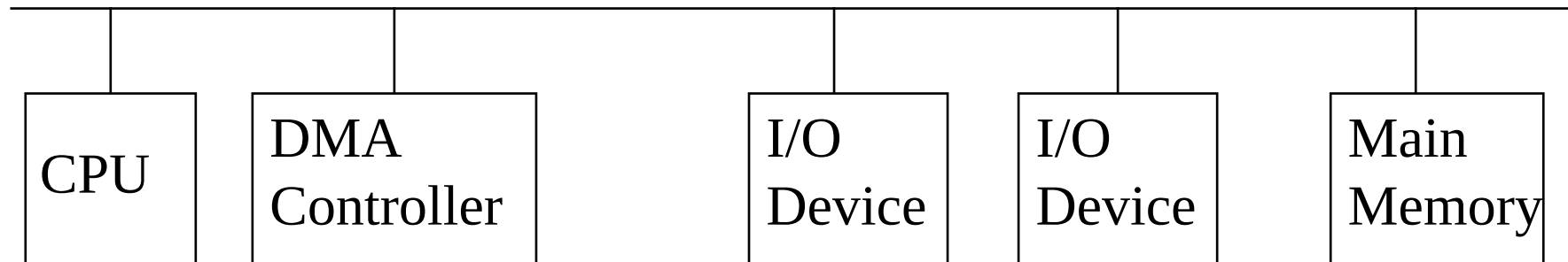
Krađa ciklusa...?!?

- ⌘ DMA preuzima ciklus prijenosa podataka od sabirnice
- ⌘ Prijenos jedne riječi podatka
- ⌘ Ovo ne koristi interrupt
 - ☒ CPU program se ne prekida
- ⌘ CPU se onemogućuje prije nego pokuša pristupiti sabirnici
 - ☒ npr. prije nego li se operand ili podatak zahvate ili zapiše
- ⌘ To ponešto usporava izvršavanje i rad CPU-a ali ne puno u odnosu kada CPU ima kontrolu nad sabirnicom

Dodatna pitanja...

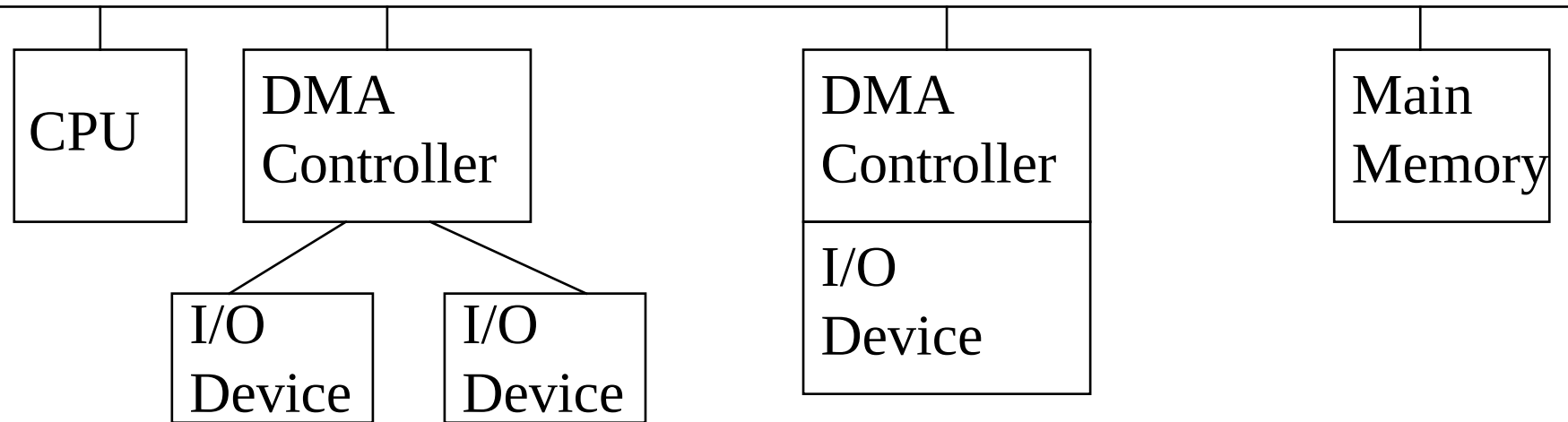
- ⌘ Kakav efekt ima cachiranje memorije na DMA transfer?
- ⌘ Pomoć: koliko je sistemska sabirnica slobodna/raspoloživa za transfer?

DMA ustroj (1)



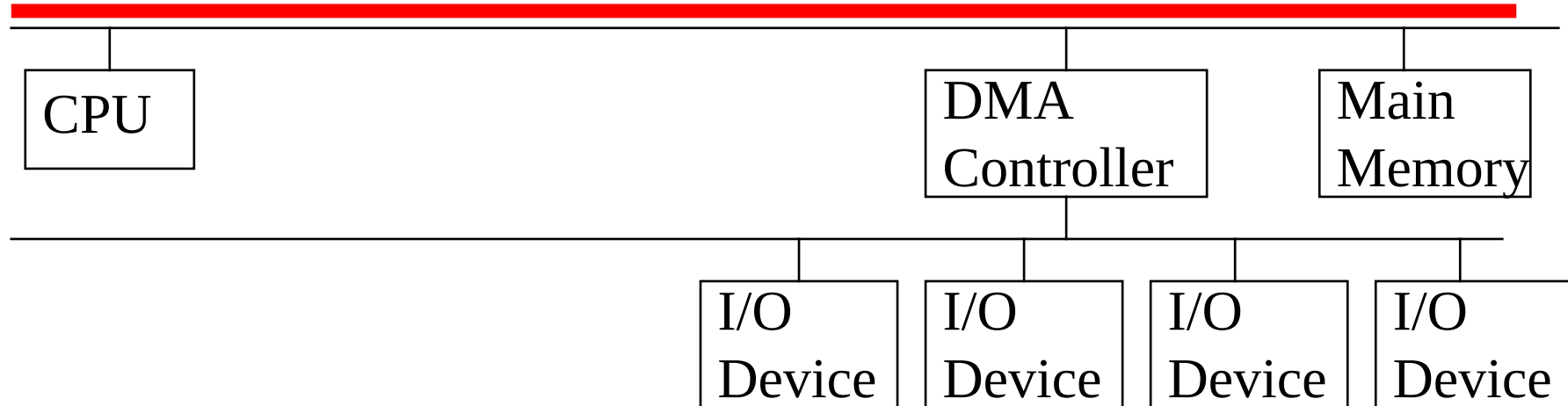
- ⌘ jedna sabirnica, odspojeni DMA kontroler
- ⌘ Svaki transfer koristi sabirnicu dva puta
 - ☒ I/O u DMA a onda DMA u memoriju
- ⌘ CPU je tada u jednom ciklusu dva puta onemogućen u korištenju sabirnice

DMA ustroj (2)



- ⌘ Jedna sabirnica, integrirani DMA kontroler
- ⌘ kontroler može podržavati više od >1 uređaja
- ⌘ Svaki transfer koristi sabirnicu samo jednom
 - ☑ DMA u memoriju
- ⌘ CPU je onemogućen samo jednom

DMA ustroj (3)



- ⌘ Odvojena I/O sabirnica
- ⌘ Sabirnica podržava sve DMA sposobne uređaje
- ⌘ Svaki transfer koristi sabirnicu jednom
 - ⌘ DMA u memoriju
- ⌘ CPU je onemogućen samo jednom

I/O kanali

- ⌘ I/O postaju sve sofisticiranije
- ⌘ npr. 3D grafičke kartice
- ⌘ CPU nalaže I/O kontroleru da obavi prijenos podataka
- ⌘ I/O kontroler obavlja cijeli transfer
- ⌘ Povećanje brzine
 - ☑ Smanjuje opterećenje CPU-a
 - ☑ Posebno građeni procesori za ovu namjenu su daleko najbrži u obavljanju ove funkcije

Povezivanje putem sučelja

- ⌘ Povezivanje uređaja međusobno
- ⌘ Koliko žica?
- ⌘ Procesori/memorije/sučelja za posebnu namjenu?
- ⌘ npr. SCSI, FireWire, USB, ...

Small Computer Systems Interface (SCSI)

- ⌘ Paralelno sučelje
- ⌘ 8, 16, 32 bit podatkovne linije
- ⌘ Daisy chained
- ⌘ Uređaji su neovisni jedna o drugom u lancu
- ⌘ Uređaji u lancu mogu komunicirati jedan sa drugim kao i u slučaju host-slave komunikacije

SCSI - 1

⌘ Rane 1980s

⌘ 8 bit

⌘ 5MHz

⌘ Data rate 5MBytes.s⁻¹

⌘ 7 uređaja u lancu

☐ 8 ako uključimo i Host-kontroler

SCSI - 2

- ⌘ 1991
- ⌘ 16 i 32 bit
- ⌘ 10MHz
- ⌘ Data rate 20 ili 40 Mbytes.s⁻¹

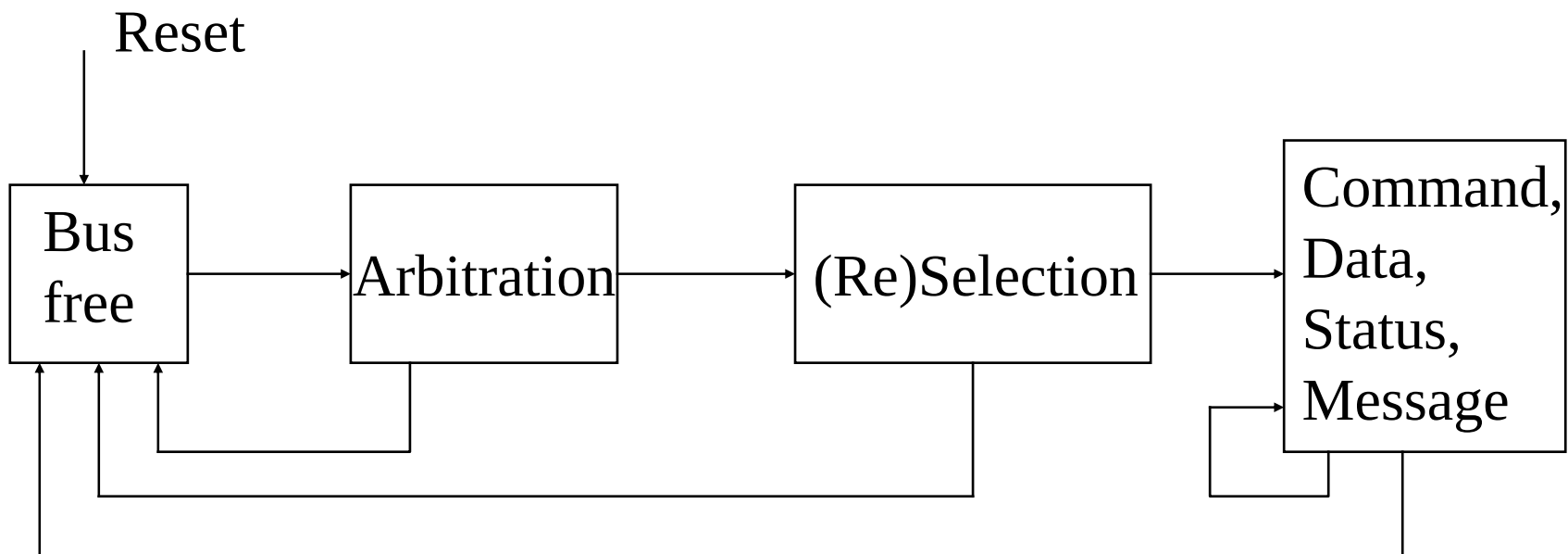
SCSI Signalizacija (1)

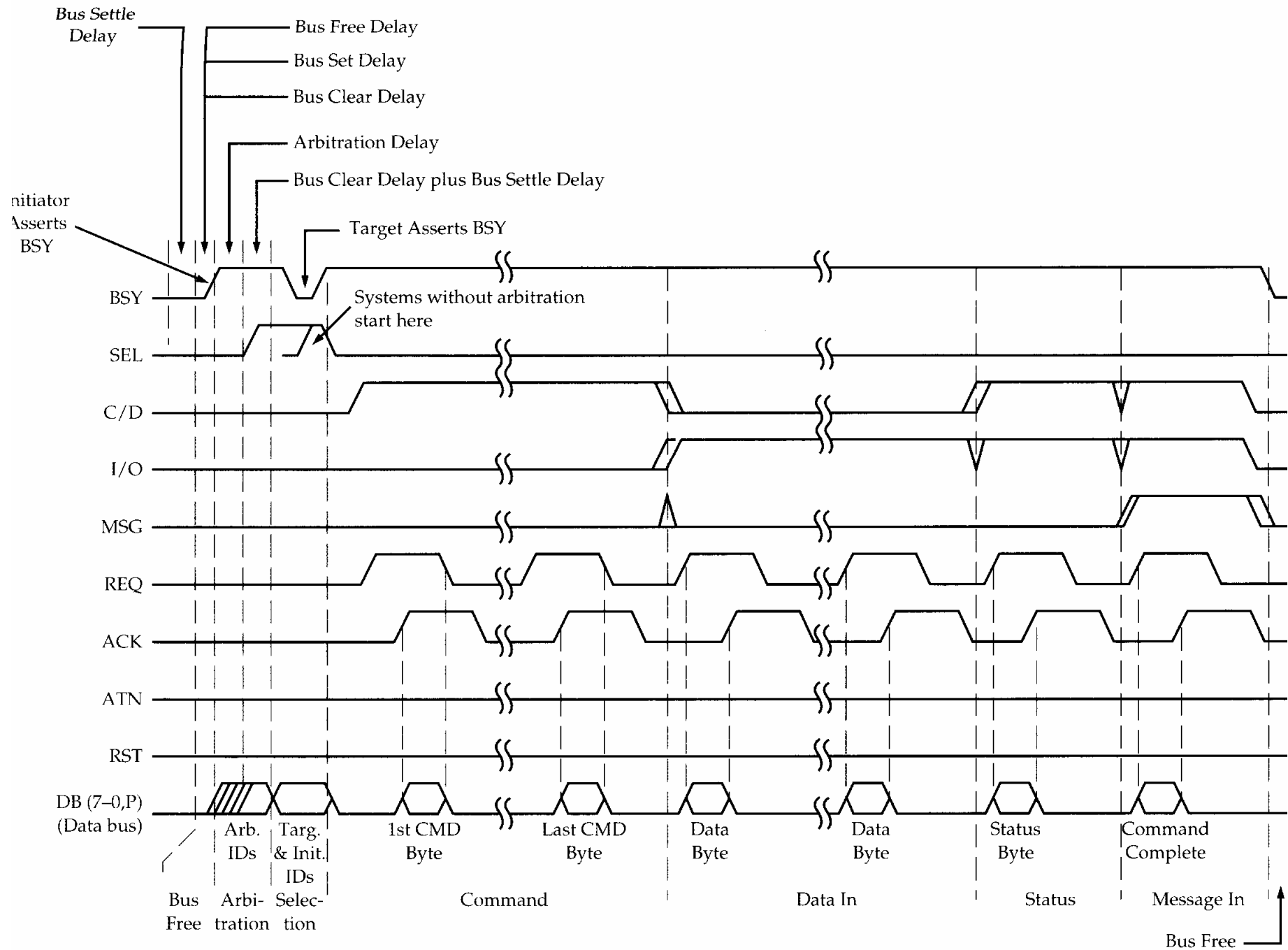
- ⌘ Između inicijatora i odredišta
 - ☒ obično je to host i device/uređaj
- ⌘ Da li je sabirnica slobodna? (npr. Ethernet)
- ⌘ Arbitracija – preuzimanje kontrole nad sabirnicom (kao kod PCI)
- ⌘ Odabir odredišnog uređaja
- ⌘ Reselection/ponovno odabiranje istog uređaja
 - ☒ omogućava rekonekciju nakon npr. suspend moda
 - ☒ npr. ako zahtjevana operacija traje određeno vrijeme tada se sabirnica može osloboditi za komunikaciju drugih uređaja

SCSI signalizacija (2)

- ⌘ Komanda – inicijator zahtijeva komunikaciju sa odredištem
- ⌘ Zahtijevaju se podaci
- ⌘ Zahtijeva se status
- ⌘ Zahtijeva se poruka (obostrano)

Faze rada SCSI sabirnice





Podešavanje/konfiguracija SCSI sabirnice

⌘ Sabirnica mora biti električki terminirana na oba kraja

- ☒ uobičajeno je da je na jednom kraju host adapter

- ☒ na drugom kraju su terminatorski priključci

⌘ SCSI identifikator se mora postaviti

- ☒ jumperi ili prekidači

- ☒ moraju biti jedinstveni u cijelom lancu

- ☒ 0 (zero) identifikator za boot jedinicu

- ☒ veći broj znači veći prioritet u arbitraciji

IEEE 1394 FireWire

- ⌘ Serijska sabirnica visokih performansi
- ⌘ Brza
- ⌘ mali troškovi, jeftina
- ⌘ jednostavna za implementaciju
- ⌘ Koristi se kod digitalnih kamera, VCR-a, TV-a, fotoaparata, ...

FireWire ustroj

- ⌘ Daisy chain
- ⌘ Do 63 devices na jednom portu
 - ☐ ustvari 64 od kojih je jedna i samo sučelje
- ⌘ Do 1022 busa se mogu povezati jednim bridge čipsetom
- ⌘ Automatska konfiguracija
- ⌘ Nema sabirničkih terminatora
- ⌘ Može se oblikovati u stablastu strukturu

Organizacija FireWire-a u 3 nivoa – 3 Layer Stack

⌘ Fizička

- ☐ Prijenosni medij, električne i signalizacijske karakteristike

⌘ Veza/Link

- ☐ Prijenos podataka u paketima

⌘ Transakcije

- ☐ Zahtjev-odgovor protokol

FireWire – Fizikalni nivo

⌘ Brzine prijenosa od 25 do 400Mbps

⌘ Dvije forme/načina arbitracije

- ☑ Temeljeno na stablastoj strukturi

- ☑ Korjenski uređaj djeluje kao arbitar/pregovarač

- ☑ "Prvi dođe-prvi bude poslužen"

- ☑ Tko je najbliži korjenskom uređaju ima najveći prioritet

- ☑ "fer" arbitracija

- ☑ i žurbena/urgent arbitracija

FireWire - Link nivo

⌘ Dvije vrste prijenosa

☐ Asinkorni prijenos

- ☒ Varijabilna količina podataka sudjeluje u procesu prijenosa u obliku paketa
- ☒ Explicitno adresiranje
- ☒ prisutna potvrda prijema i/ili predaje podataka jednog paketa

☐ Izokrona/sinkrona

- ☒ Podaci se prenose u paketima fiksne veličine u pravilnim vremenskim intervalima
- ☒ Vrlo jednostavno/pojednostavljeno adresiranje
- ☒ Nema potvrde prijema/slanja paketa sa podacima