Operativsystem översikt

Mittuniversitetet
MID SWEDEN UNIVERSITY

Operativsystem

- Ger en eller flera processer tillgång till de samlade resurserna
- Erbjuder ett antal tjänster till användaren
- Hanterar sekundärt minne och I/O-enheter

Mittuniversitetet
MID SWEDEN UNIVERSITY

Datorns grundelement

- Processor
- Minne
  - kallas primärminne
  - flyktigt
- I/O-enheter
  - Sekundära minnesenheter (diskar, band…)
  - kommunikationsenheter
  - terminaler
- Systembuss
  - kommunikation mellan processorer, minne och I/O-enheter

Mittuniversitetet
MID SWEDEN UNIVERSITY

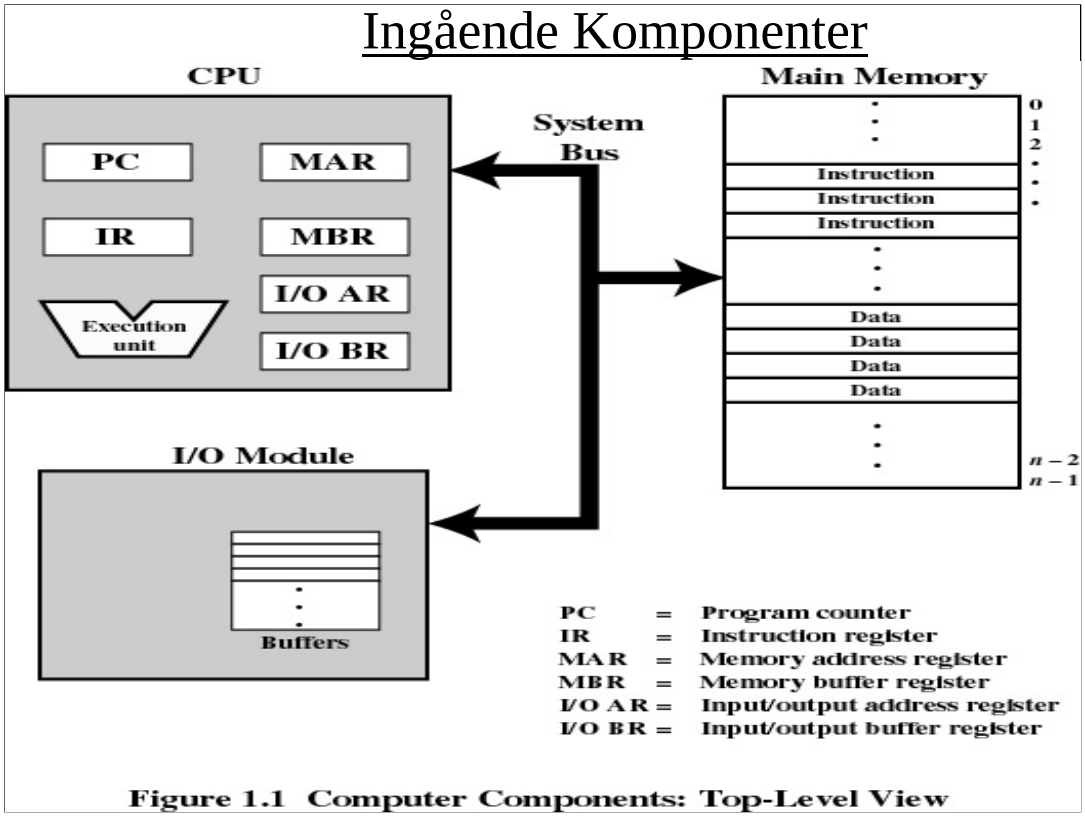


Figure 1.1 Computer Components: Top-Level View

Bild 4 visar en del av (en mycket förenklad) processorns interna strukturer och dess förbindelse med yttre
enheter. Notera *Execution unit* som motsvarar ALU (Arithmetic and Logic Unit) där allt egentligt arbete utförs dvs. aritmetiska funktioner som +, - osv. och logiska funktioner AND, OR osv. (Här ingår också flyttalsenhet etc.) Endast viss flyttning av data kan ske utan att ALU är inblandad.

Kring ALU finns ett antal register. En del för lagring av temporära data, andra för att hålla adresser. PC är det register som innehåller adressen till (pekar på) nästa instruktion som ska läsas in.

Systembussen består av flera bussar: databuss, adressbuss, kontrollbuss. Dessa samverkar för att flytta data mellan processor och t ex minnet.

Skillnad mellan register som jag som programmerare kan använda och de som endast processorn kan skriva till. De kan dock vara läsbara. Statusflaggor (som är en bit i ett register) kan indikera ex vis negativt tal, overflow, resultat av logisk operation och är naturligtvis läs men inte skrivbara.

Processorns Register

- Användarregister
  - Används av programmeraren för att minska accessen till det långsammare primärminnet
- Kontroll och statusregister
  - Används av processorn för övervakning av arbetet
  - Endast läsbara för användare
  - Används av operativsystemet för att kontrollera exekveringen av programmet

Mittuniversitetet
MID SWEDEN UNIVERSITY

Register ligger i CPU och att snabbare både att accessa och adressera

Skrivbara Register

- Kan nås via maskinspråk
- Tillgängliga för alla program – applikation eller systemprogram
- Typer av register
  - Data
  - Adress
    - Index
    - Segment pekare
    - Stack pekare

Mittuniversitetet
MID SWEDEN UNIVERSITY

User-visible

Skrivbara Register

- Adressregister
  - Index
    - adderas till en basadress för att få rätt adress
  - Segmentpekare
    - används vid segmenterat minne, minnesadressen består av segmentadress plus offset
  - Stackpekare
    - pekar till toppen på stacken

Mittuniversitetet
MID SWEDEN UNIVERSITY

Stacken

Stacken är en viktig funktion. Den används också för temporär lagring, men ligger i primärminnet. Det är egentligen inget speciellt minne, utan en del av minnet som används.

Data jag vill lagra, läggs där stackpekaren pekar och stackpekarens värde ändras till nästa adress.

Om jag lagrar mer data hamnar det ”ovanför” det förra och måste plocka bort innan jag når det jag la dit tidigare.

Stacken har en LIFO-struktur (Last-In-First-Out) dvs. det som läggs in sist måste hämtas ut först

Kontroll och statusregister

- *Program Counter* (PC)
  - Innehåller adressen till nästa instruktion
- Instruktionsregister (IR)
  - Innehåller senast hämtade instruktion
- *Program Status Word* (PSW)
  - statusregister
  - Interrupt enable/disable
  - Supervisor/user mode

Mittuniversitetet
MID SWEDEN UNIVERSITY

Innehåller = pekar på

Neg, Eq, Spill, /0

Kontroll och statusregister

- Statusregistret innehåller flaggor
  - Bitar som sätts av processorn/ALU som resultat av operationer
  - Flaggorna kan läsas av programmet, men inte ändras
  - Exempel
    - Resultatet är positivt
    - Resultatet är negativt
    - Noll
    - Overflow/spill

Mittuniversitetet
MID SWEDEN UNIVERSITY

Exekvering av program

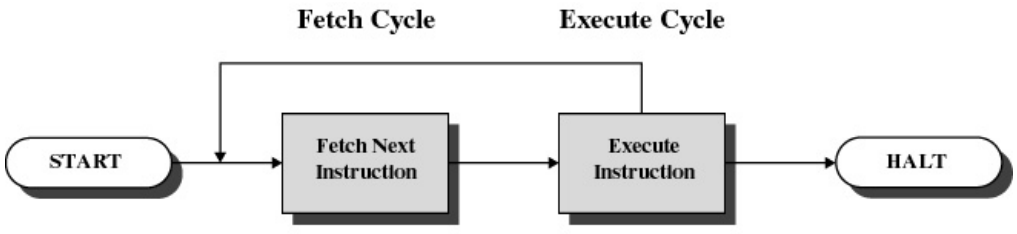


Figure 1.2 Basic Instruction Cycle

Mittuniversitetet
MID SWEDEN UNIVERSITY

Exekvering av program

Bild 10 visar maskincykeln, men en del saknas. Mellan *Fetch* och *Execute* ska *Decode* finnas.

Den instruktion som PC pekar på läses in (fetch) och läggs i IR (instruktionsregistret).

Instruktionen avkodas (decode) och exekveras sedan av processorn (execute). Sedan börjar en ny cykel dvs. nästa instruktion läses osv.

Hämta och exekvera

- *Program counter* (PC) pekar på adressen för den instruktion som ska hämtas nästa gång
- Processorn hämtar instruktion från minnet
- *Program counter* ökas efter varje hämtning

Mittuniversitetet
MID SWEDEN UNIVERSITY

Instruktionsregister

- Senast hämtade instruktion placeras i instruktionsregistret
- Typer av instruktioner
  - Processor-minne
    - Flytta data mellan processor och minne
  - Processor-I/O
    - data flyttas till eller från periferienhet
  - Databehandling
    - aritmetisk eller logisk operation på data
  - Kontroll
    - förändrar sekvensen av instruktioner

Mittuniversitetet
MID SWEDEN UNIVERSITY

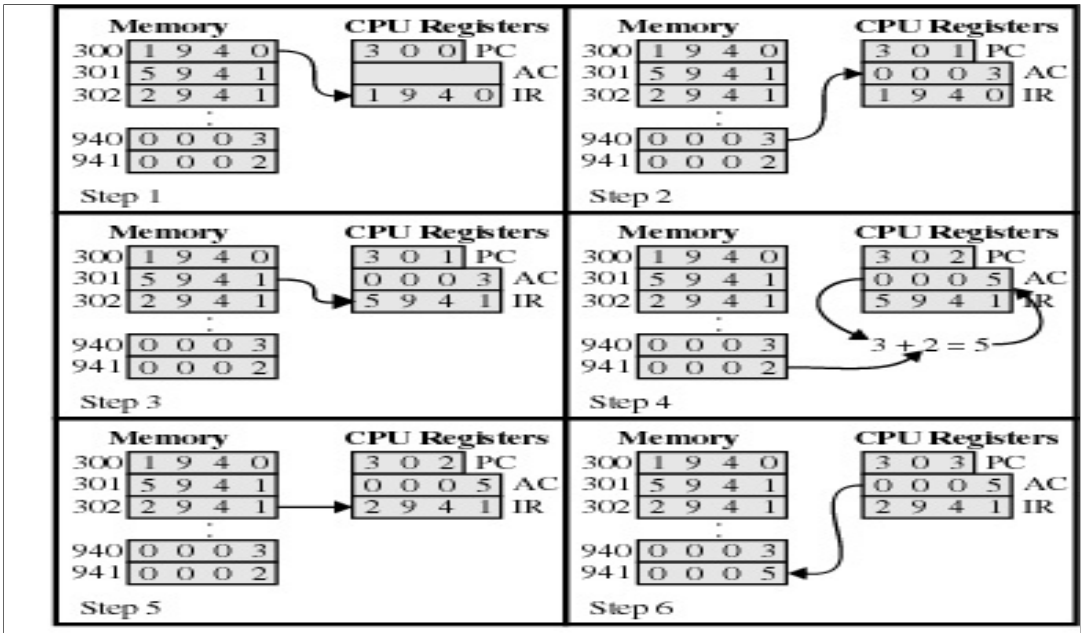


Figure 1.4 Example of Program Execution (contents of memory and registers in hexadecimal)

Exekvering av Program

Addition av två tal (3 och 2) som redan ligger i minnet (940, 941)

Resultatet (5) skriver över adress 941

Direct Memory Access (DMA)

- I/O-enheter utbyter data direkt med minnet
- Processorn ger I/O-enheten rätt att läsa och skriva direkt till minnet
- Avlastar processorn från datatransporten
- Processorn kan utföra annat arbete under tiden

Mittuniversitetet
MID SWEDEN UNIVERSITY

Interrupts (avbrott)

- Avbryter den normala exekveringssekvensen
- Ökar processeffektiviteten
- Ger processorn möjlighet att exekvera andra instruktioner medan en I/O-operation pågår
- Avbrottet i en exekvering av en process, pga. ett externt interrupt, sker på ett sådant sätt att den normala exekveringen kan fortsätta när interrupt är hanterat

Mittuniversitetet
MID SWEDEN UNIVERSITY

Interrupt

Interrupt är en viktig funktion för att öka effektiviteten. Ett program exekveras tills något händer. I/O-anrop sker oftast via interrupt. Ex vis ner man trycker på en tangent på tangentbordet så skickas ett interrupt som talar om för processorn att det finns tecken att hämta i tangentbordsbufferten. Vid ett interrupt avbryter processorn det den håller på med för att hantera interruptet. För att kunna återuppta arbetet efter interruptet så sparar innehållet i samtliga processorns register på stacken. När arbetet ska återupptas hämtas allt tillbaka och processorn fortsätter där den blev avbruten.

Vid speciella tillfällen, när ett program inte får avbrytas, kan interrupten tillfälligt stängas av (disable). De interrupt som kommer in kommer då att läggas i kö.

Olika Typer Av Interrupt

- Program
  - aritmetiskt overflow (spill)
  - division med noll
  - exekvering av olaglig instruktion
  - Referens utanför tillåtet minnesområde
- Timer
- I/O
- Hårdvarufel

Mittuniversitetet
MID SWEDEN UNIVERSITY

Interrupthanterare

- Ett program som undersöker aktuellt interrupt och utför lämplig åtgärd
- Kontrollen överförs vid ett interrupt till interrupthanteraren
- Interrupthanteraren är en bit av operativsystemet

Mittuniversitetet
MID SWEDEN UNIVERSITY

Interruptsekvens

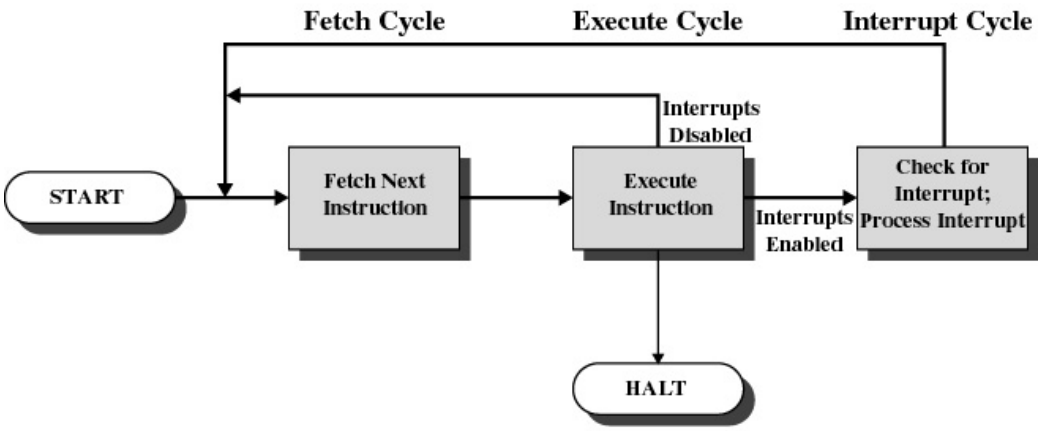


Figure 1.7 Instruction Cycle with Interrupts

Mittuniversitetet
MID SWEDEN UNIVERSITY

Interruptsekvens

- Processorn kontrollerar om det finns väntande interrupt
- Nej - hämta nästa instruktion i det aktuella programmet
- Ja det finns väntande interrupt - avbryt exekveringen av pågående program och kör interrupthanteraren

Mittuniversitetet
MID SWEDEN UNIVERSITY

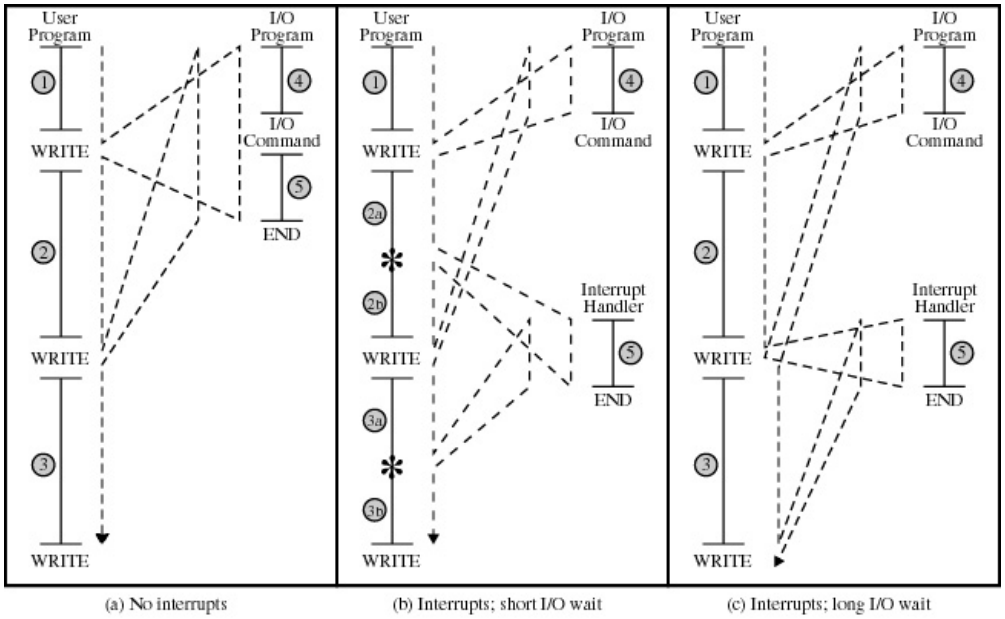


Figure 1.5 Program Flow of Control Without and With Interrupts

Bild 20: visar skillnaden i exekveringen av ett program med två utskrifter, utan och med interrupthantering (utskriftsrutinen innehåller i sig ett I/O-anrop). Utan interrupt så lämnas kontrollen helt över till utskriften varje gång och huvudprogrammet blockeras tills allt är klart. Med interrupt så kan programmet fortsätta att exekvera medan utskriften väntar på sitt I/O-anrop. Effektiviteten ökar med interrupt.

Multipla Interrupt

- Stäng av andra interrupt under tiden ett interrupt processas
  - Processorn ignorerar ny interruptbegäran
- Eller hantera alla nya när de kommer

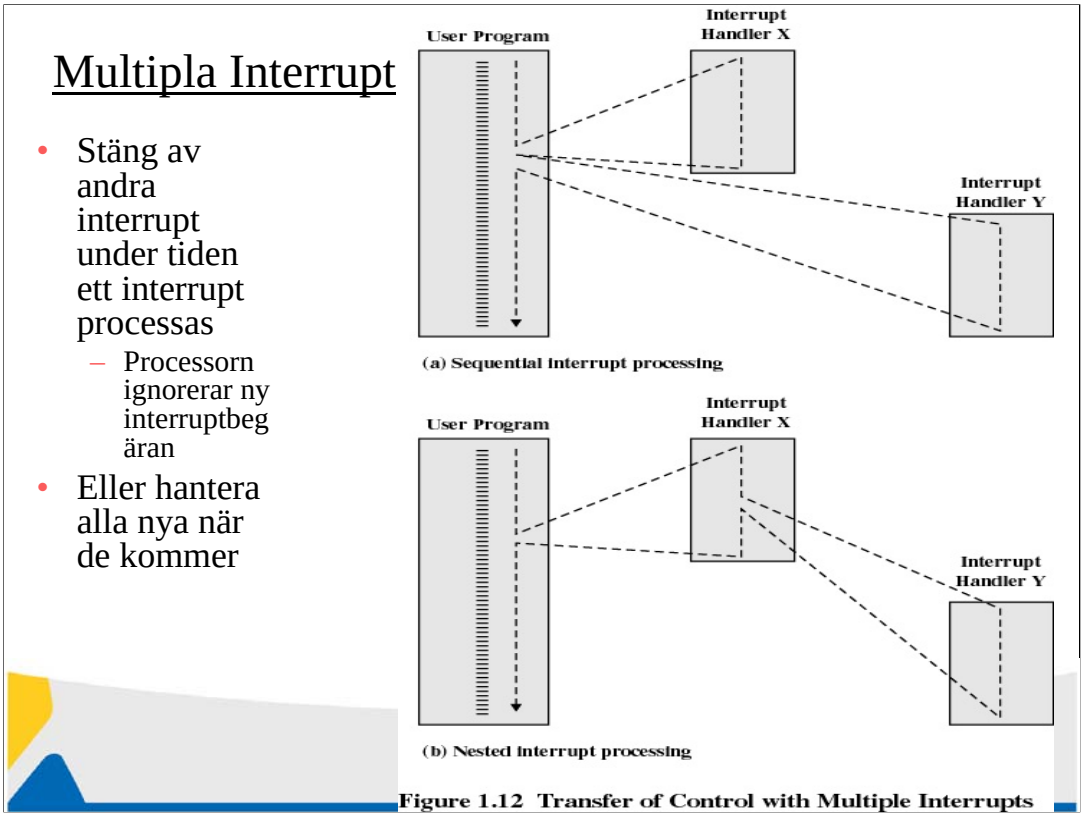


Figure 1.12 Transfer of Control with Multiple Interrupts

Multipla Interrupt Sekventiell Hantering

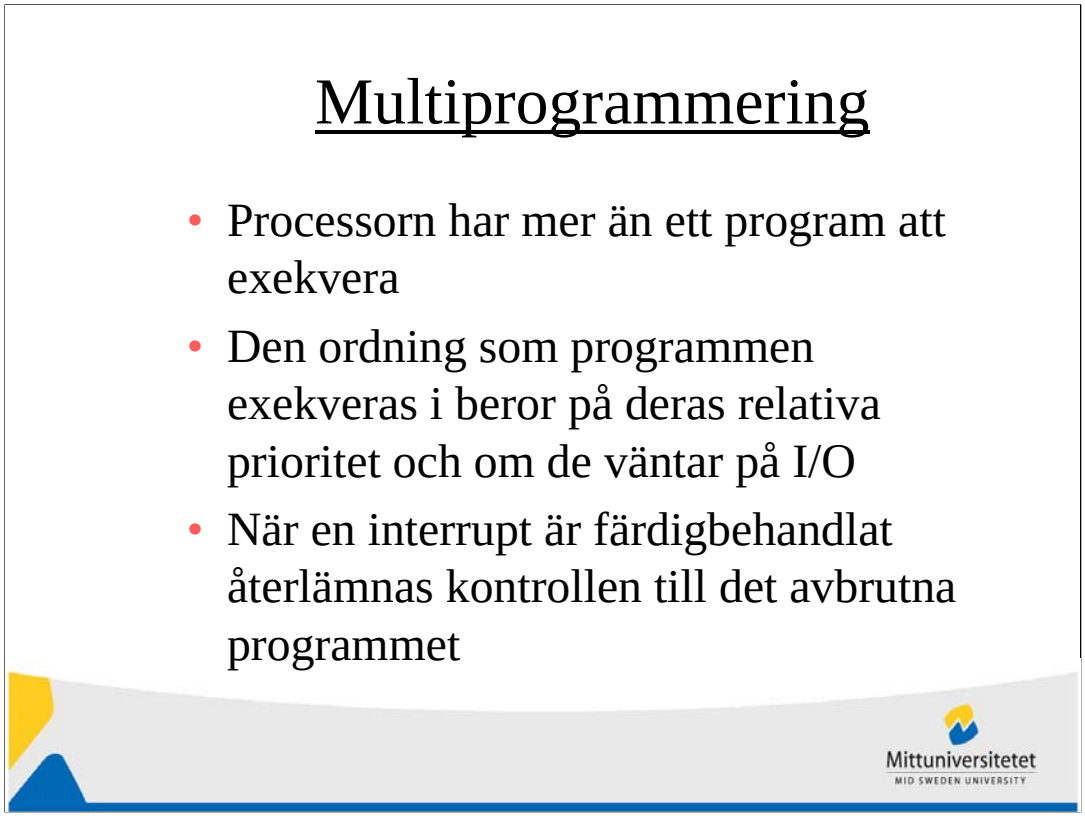
- Stäng av (disable) interrupt så att processorn kan slutföra sin uppgift
- Nya interrupts köas tills processorn är klar att ta emot nya (enable)
- Först när interruptrutinen är färdig letar processorn efter nya

Mittuniversitetet
MID SWEDEN UNIVERSITY

Multipla Interrupt Prioritering

- Ett interrupt med högre prioritet avbryter ett med lägre (interrupt i interrupt)
- Data från snabbare överföringar/mindre buffertar ges högre prioritet

Mittuniversitetet
MID SWEDEN UNIVERSITY



Multiprogrammering

Bild 24: är en introduktion till shedulering dvs. hur OS väljer vilken process som ska exekveras.

Minneshierarki

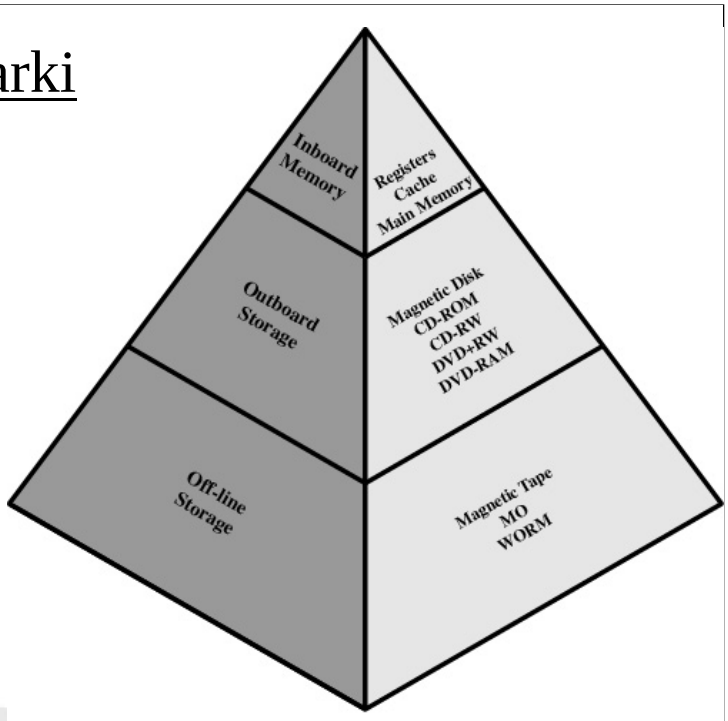


Figure 1.14 The Memory Hierarchy

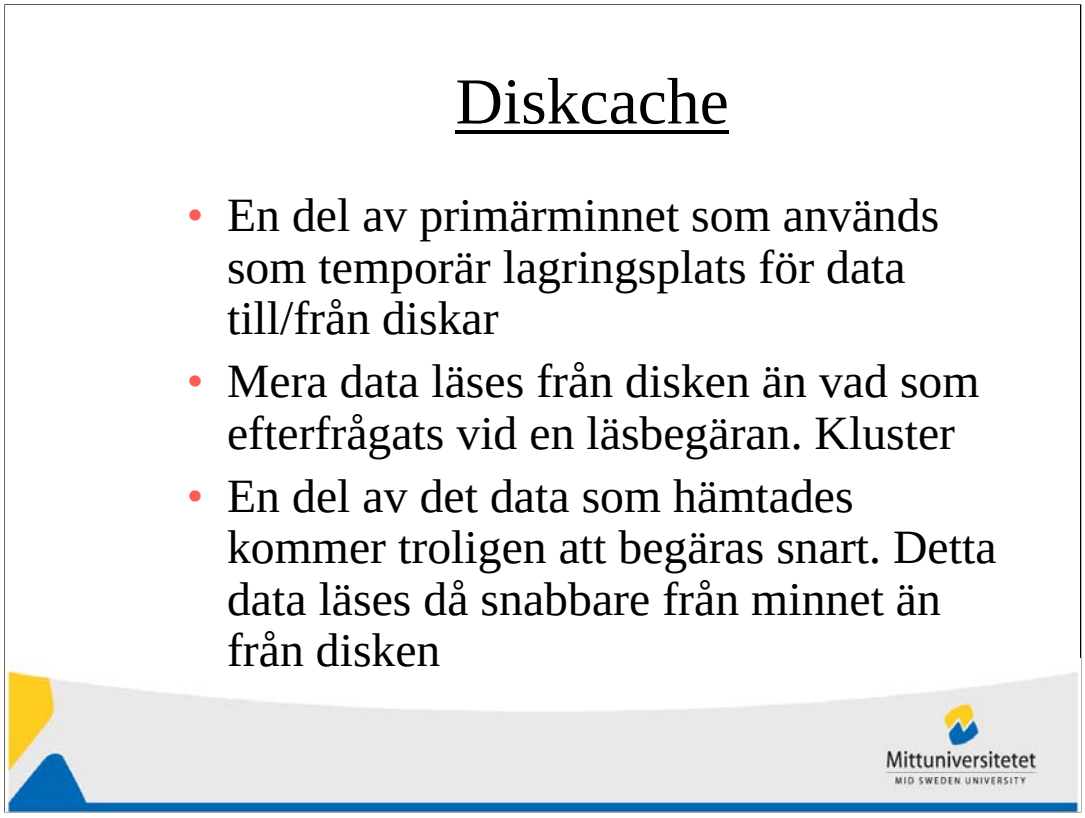
Ned i hierarkin innebär:

Lägre kostnad/bit

Ökad kapacitet

Ökad accesstid

Processorn accessar minnet mera sällan



Cache

Bild 27: beroende på det sätt som program är skrivna (ofta loopar) så är sannolikheten stor att det data som behövs nästa gång ligger i närheten av det som användes senast. Detta kallas **lokalitetsprincipen**. Man strävar efter att cachen alltid ska innehålla det som behövs (hög hit-rate). I cachen bör alltså ligga en större bit runt omkring aktuell position för PC.

Cacheminne

- Osynligt för operativsystemet
- Minskar accesstid till data
- Primärminnet är långsammare än processorns cache

Mittuniversitetet
MID SWEDEN UNIVERSITY

Cacheminne

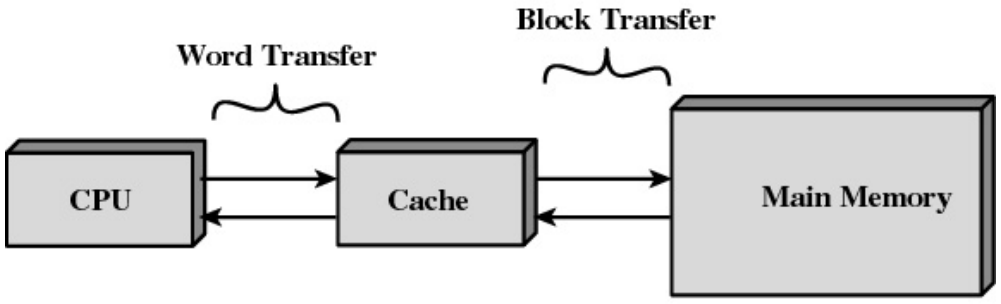


Figure 1.16 Cache and Main Memory

Mittuniversitetet
MID SWEDEN UNIVERSITY

Cacheminne

Innehåller en del inläst primärminne

Processorn letar först i cachen

Om sökt data inte finns där så läses ett block som innehåller detta in till cachen

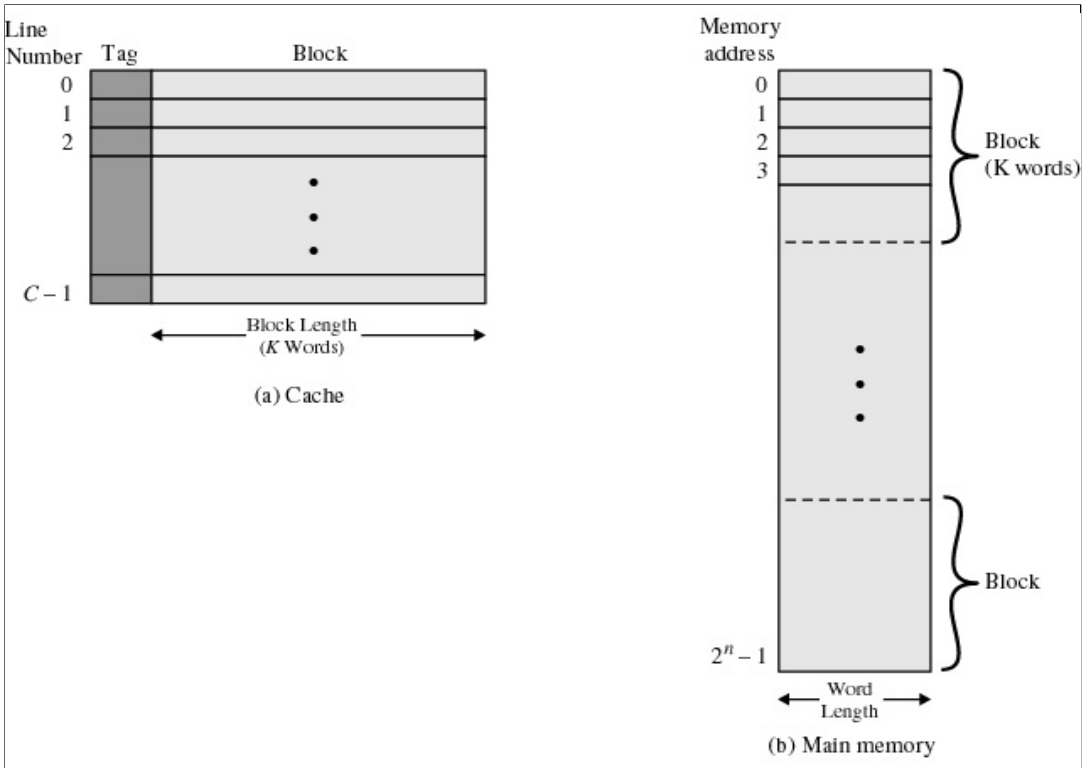


Figure 1.17 Cache/Main-Memory Structure

Cachedesign

- Cachens storlek
  - redan liten cache påverkar prestanda positivt
- Blockstorlek
  - Hur mycket som hämtas från primärminnet
  - Träff (hit) betyder att sökt data hittades i cacheminnet
  - Större hämtad blockstorlek ger flera träffar, tills sannolikheten för att sedan önskat data finns i det nya är mindre än att det fanns i det som har kastats ur cachen

Mittuniversitetet
MID SWEDEN UNIVERSITY

Cachedesign

- Mappning av cache
  - Bestämmer vilken plats i cacheminnet som inläst block får
- *Replacement algorithm*
  - Väljer vilket block som ska bytas ut
  - *Least-Recently-Used* (LRU) algoritm

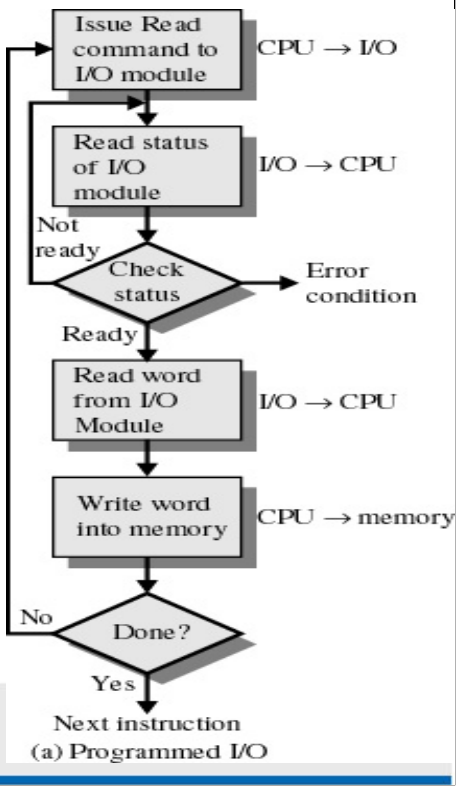
Mittuniversitetet
MID SWEDEN UNIVERSITY

Cachedesign

- *Write policy.* När ska ändrat data skrivas tillbaka till primärminnet
  - Varje gång som ett block uppdateras
  - Varje gång som ett block är utbytt
    - Minimerar minnesaccess
    - Primärminnet innehåller inte alltid aktuell data

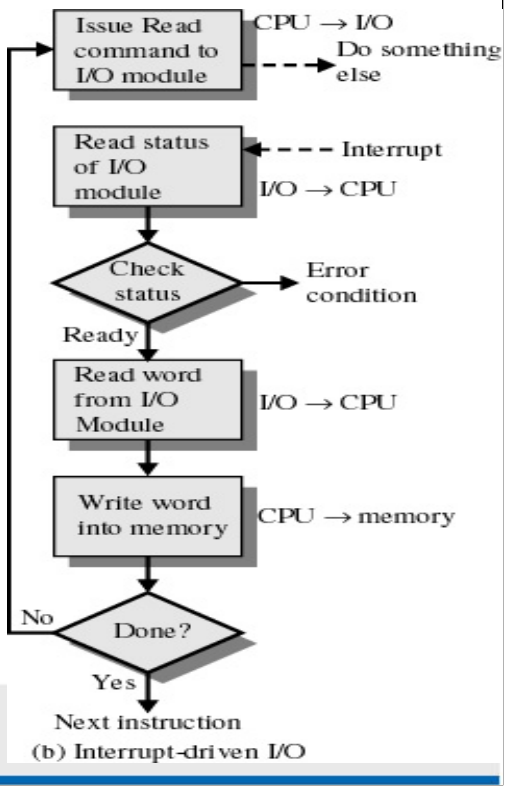
Programmerad I/O

- I/O-enhet utför arbetet, inte processorn
- Sätter lämpliga bitar i I/O-statusregister
- Ingen interrupt sker
- Processorn undersöker status tills operationen är färdig



Interruptstyrd I/O

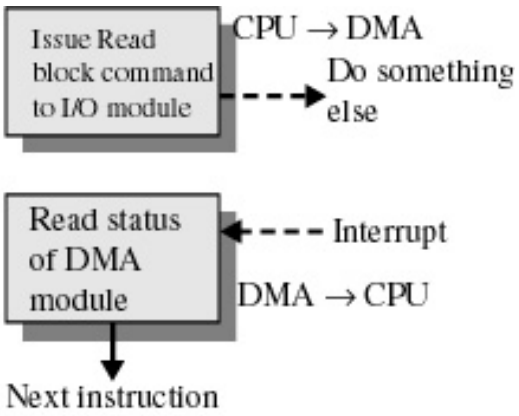
- Processorn får ett interrupt när I/O-enheten är klar att utbyta data
- Processorn kan göra annat under tiden
- Ingen onödig väntan
- Förbrukar mycket processortid eftersom allt data måste passera genom processorn



(b) Interrupt-driven I/O

Direct Memory Access

- Flyttar ett block med data direkt till/från minnet
- Interrupt när det är klart
- Processorn är bara inblandad i början och i slutet av överföringen



(c) Direct memory access

Mittuniversitetet
MID SWEDEN UNIVERSITY