

# Computer System Overview

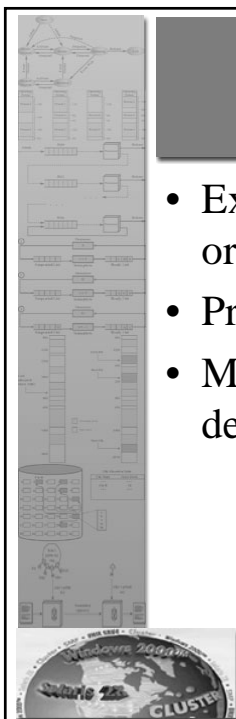
## Chapter 1

Muhammad Adri, MT

1

## Operating System

- Exploits the hardware resources of one or more processors
- Provides a set of services to system users
- Manages secondary memory and I/O devices




Muhammad Adri, MT

2

## Basic Elements

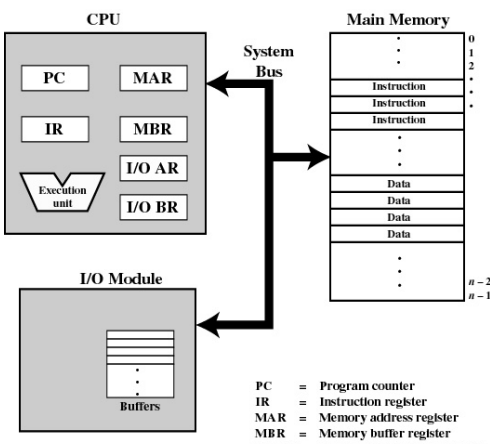
- Processor
- Main Memory
  - referred to as real memory or primary memory
  - volatile
- I/O modules
  - secondary memory devices
  - communications equipment
  - terminals
- System bus
  - communication among processors, memory, and I/O modules



Muhammad Adri, MT

3

## Top-Level Components



PC = Program counter  
 IR = Instruction register  
 MAR = Memory address register  
 MBR = Memory buffer register  
 I/O AR = Input/output address register  
 I/O BR = Input/output buffer register





Figure 1.1 Computer Components: Top-Level View

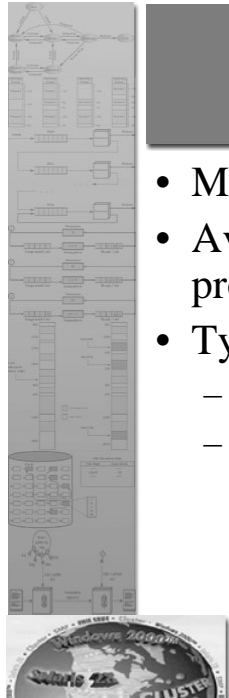
4



## Processor Registers

- User-visible registers
  - Enable programmer to minimize main-memory references by optimizing register use
- Control and status registers
  - Used by processor to control operating of the processor
  - Used by operating-system routines to control the execution of programs

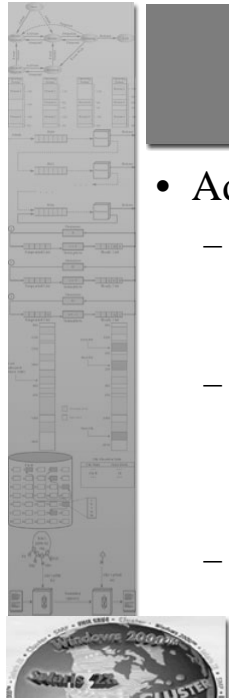
Muhammad Adri, MT 5



## User-Visible Registers

- May be referenced by machine language
- Available to all programs - application programs and system programs
- Types of registers
  - Data
  - Address
    - Index
    - Segment pointer
    - Stack pointer

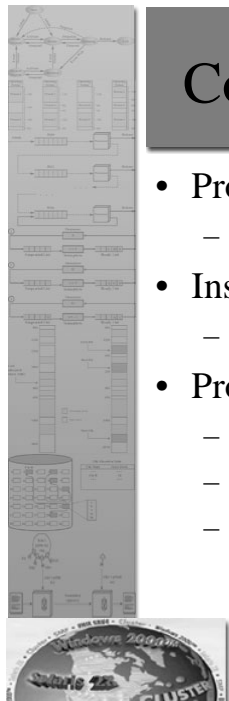
Muhammad Adri, MT 6



## User-Visible Registers

- Address Registers
  - Index
    - involves adding an index to a base value to get an address
  - Segment pointer
    - when memory is divided into segments, memory is referenced by a segment and an offset
  - Stack pointer
    - points to top of stack

Muhammad Adri, MT 7



## Control and Status Registers

- Program Counter (PC)
  - Contains the address of an instruction to be fetched
- Instruction Register (IR)
  - Contains the instruction most recently fetched
- Program Status Word (PSW)
  - condition codes
  - Interrupt enable/disable
  - Supervisor/user mode

Muhammad Adri, MT 8

## Control and Status Registers

- Condition Codes or Flags
  - Bits set by the processor hardware as a result of operations
  - Can be accessed by a program but not altered
  - Examples
    - positive result
    - negative result
    - zero
    - Overflow

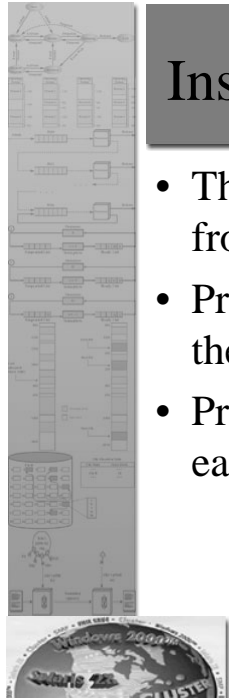
Muhammad Adri, MT 9

## Instruction Cycle

```
graph LR; START([START]) --> Fetch[Fetch Next Instruction]; Fetch --> Execute[Execute Instruction]; Execute --> HALT([HALT]);
```

Figure 1.2 Basic Instruction Cycle

Muhammad Adri, MT 10

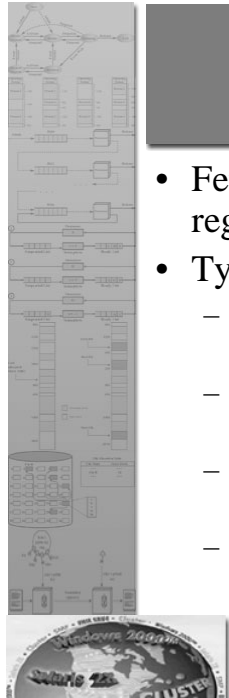


## Instruction Fetch and Execute

- The processor fetches the instruction from memory
- Program counter (PC) holds address of the instruction to be fetched next
- Program counter is incremented after each fetch

Muhammad Adri, MT

11

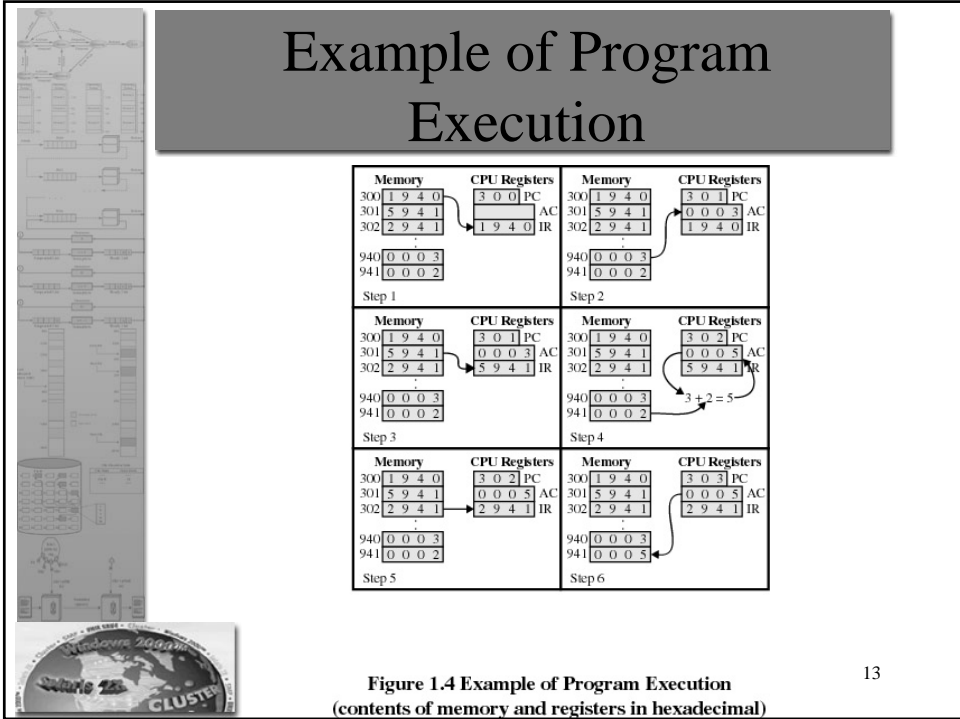


## Instruction Register

- Fetched instruction is placed in the instruction register
- Types of instructions
  - Processor-memory
    - transfer data between processor and memory
  - Processor-I/O
    - data transferred to or from a peripheral device
  - Data processing
    - arithmetic or logic operation on data
  - Control
    - alter sequence of execution

Muhammad Adri, MT

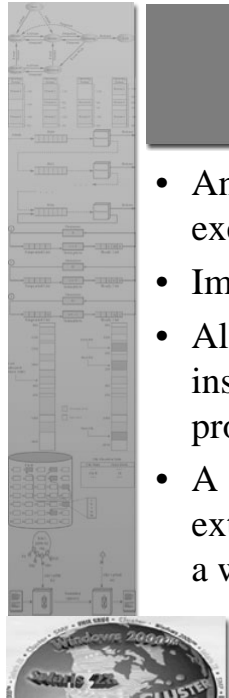
12



## Direct Memory Access (DMA)

- I/O exchanges occur directly with memory
- Processor grants I/O module authority to read from or write to memory
- Relieves the processor responsibility for the exchange
- Processor is free to do other things

Muhammad Adri, MT

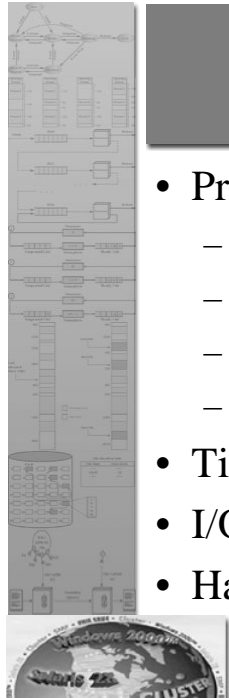


## Interrupts

- An interruption of the normal sequence of execution
- Improves processing efficiency
- Allows the processor to execute other instructions while an I/O operation is in progress
- A suspension of a process caused by an event external to that process and performed in such a way that the process can be resumed

Muhammad Adri, MT

15



## Classes of Interrupts

- Program
  - arithmetic overflow
  - division by zero
  - execute illegal instruction
  - reference outside user's memory space
- Timer
- I/O
- Hardware failure

Muhammad Adri, MT

16

## Interrupt Handler

- A program that determines nature of the interrupt and performs whatever actions are needed
- Control is transferred to this program
- Generally part of the operating system

Muhammad Adri, MT 17

## Interrupt Cycle

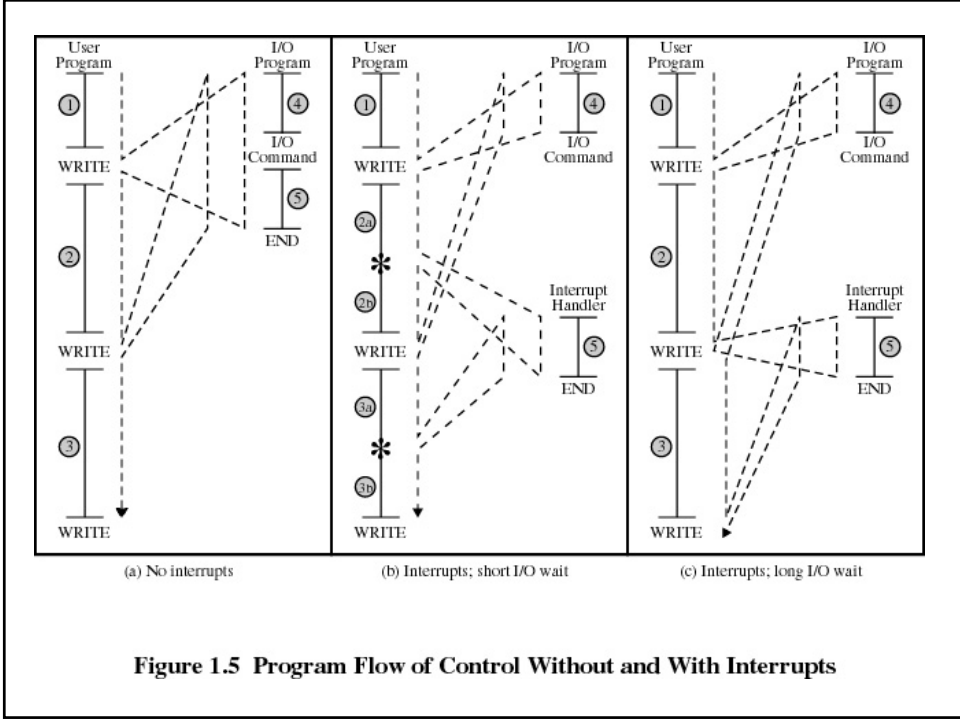
```
graph LR; START([START]) --> Fetch[Fetch Next Instruction]; Fetch --> Execute[Execute Instruction]; Execute -- "Interrupts Disabled" --> Fetch; Execute -- "Interrupts Enabled" --> Check[Check for Interrupt; Process Interrupt]; Check --> Fetch; Check --> HALT([HALT]);
```

Muhammad Adri, MT 18

## Interrupt Cycle

- Processor checks for interrupts
- If no interrupts fetch the next instruction for the current program
- If an interrupt is pending, suspend execution of the current program, and execute the interrupt handler

Muhammad Adri, MT 19



## Multiple Interrupts

- Disable interrupts while an interrupt is being processed
  - Processor ignores any new interrupt request signals

(a) Sequential interrupt processing

(b) Nested interrupt processing

Muhammad Adri, MT

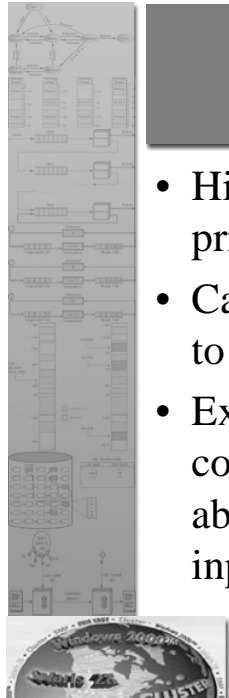
Figure 1.12 Transfer of Control with Multiple Interrupts

## Multiple Interrupts Sequential Order

- Disable interrupts so processor can complete task
- Interrupts remain pending until the processor enables interrupts
- After interrupt handler routine completes, the processor checks for additional interrupts

Muhammad Adri, MT

22

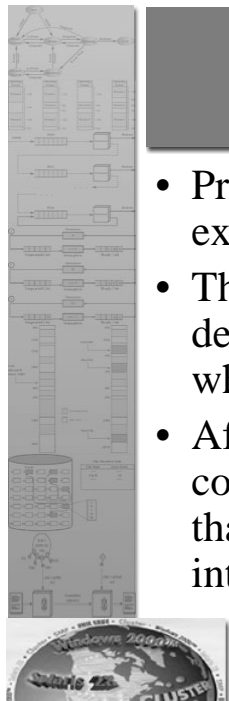


## Multiple Interrupts Priorities

- Higher priority interrupts cause lower-priority interrupts to wait
- Causes a lower-priority interrupt handler to be interrupted
- Example when input arrives from communication line, it needs to be absorbed quickly to make room for more input

Muhammad Adri, MT

23

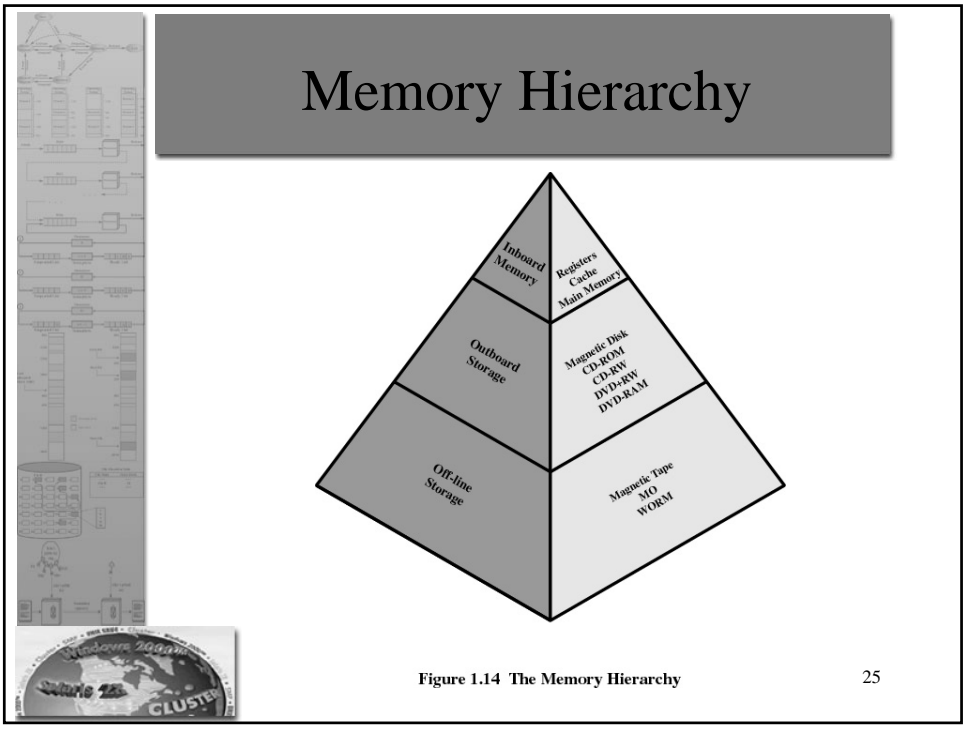


## Multiprogramming

- Processor has more than one program to execute
- The sequence the programs are executed depend on their relative priority and whether they are waiting for I/O
- After an interrupt handler completes, control may not return to the program that was executing at the time of the interrupt

Muhammad Adri, MT

24

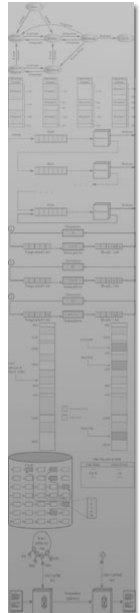


## Going Down the Hierarchy

- Decreasing cost per bit
- Increasing capacity
- Increasing access time
- Decreasing frequency of access of the memory by the processor
  - locality of reference


Muhammad Adri, MT

26

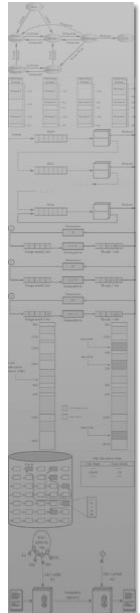


## Disk Cache

- A portion of main memory used as a buffer to temporarily to hold data for the disk
- Disk writes are clustered
- Some data written out may be referenced again. The data are retrieved rapidly from the software cache instead of slowly from disk




Muhammad Adri, MT 27

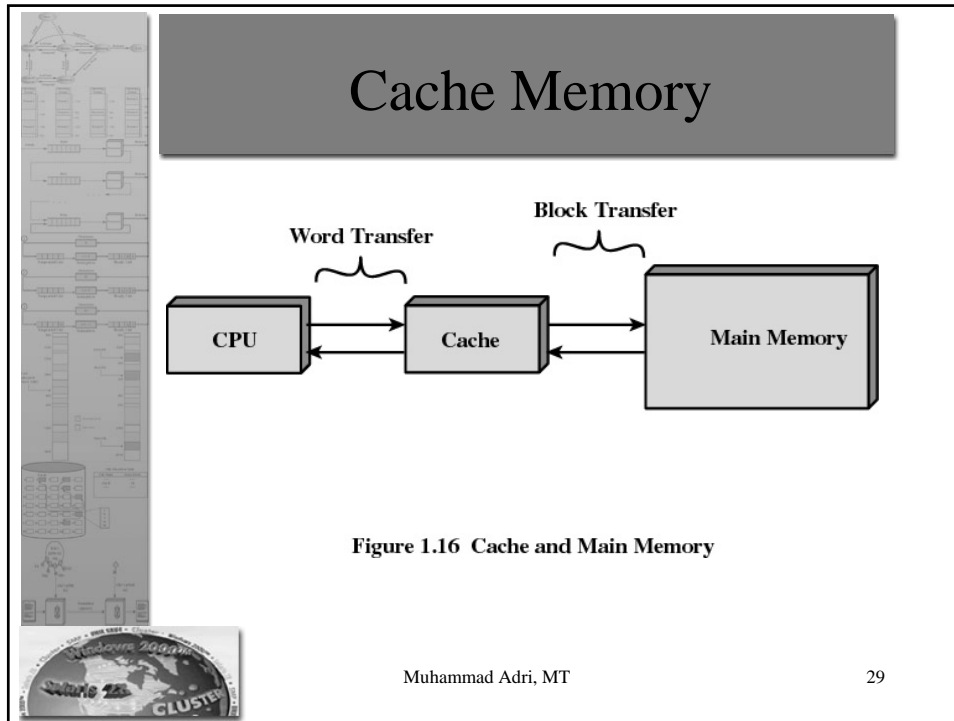


## Cache Memory

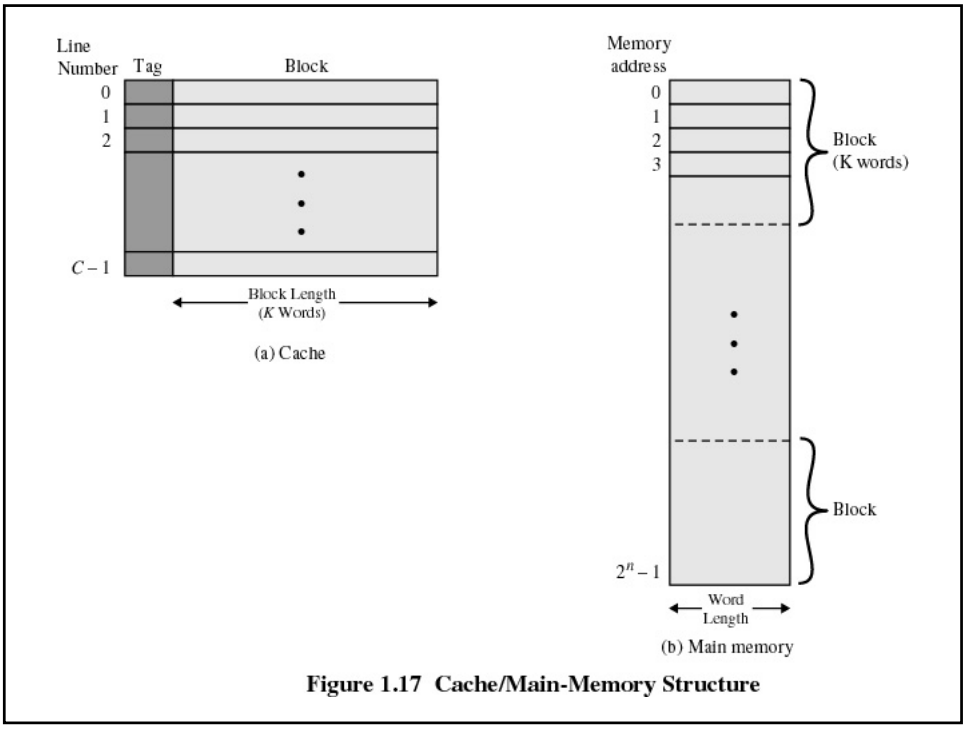
- Invisible to operating system
- Increase the speed of memory
- Processor speed is faster than memory speed



Muhammad Adri, MT 28



- ## Cache Memory
- Contains a portion of main memory
  - Processor first checks cache
  - If not found in cache, the block of memory containing the needed information is moved to the cache
- Muhammad Adri, MT
- 30




## Cache Design

- **Cache size**
  - small caches have a significant impact on performance
- **Block size**
  - the unit of data exchanged between cache and main memory
  - hit means the information was found in the cache
  - larger block size more hits until probability of using newly fetched data becomes less than the probability of reusing data that has been moved out of cache

Muhammad Adri, MT

32

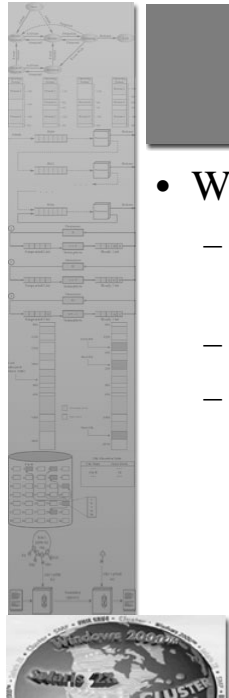


## Cache Design

- Mapping function
  - determines which cache location the block will occupy
- Replacement algorithm
  - determines which block to replace
  - Least-Recently-Used (LRU) algorithm

Muhammad Adri, MT

33



## Cache Design

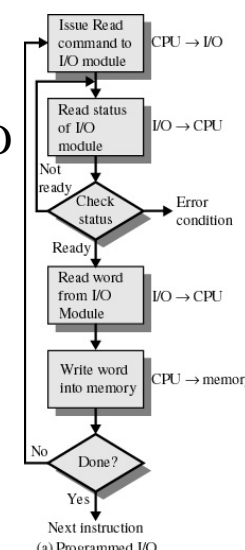
- Write policy
  - When the memory write operation takes place
  - Can occur every time block is updated
  - Can occur only when block is replaced
    - Minimizes memory operations
    - Leaves memory in an obsolete state

Muhammad Adri, MT

34

## Programmed I/O

- I/O module performs the action, not the processor
- Sets appropriate bits in the I/O status register
- No interrupts occur
- Processor checks status until operation is complete

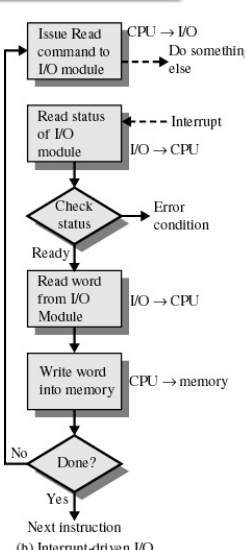


Next instruction  
(a) Programmed I/O

Muhammad Adri, MT

## Interrupt-Driven I/O

- Processor is interrupted when I/O module ready to exchange data
- Processor is free to do other work
- No needless waiting
- Consumes a lot of processor time because every word read or written passes through the processor

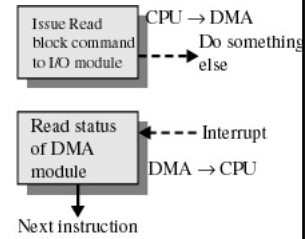


Next instruction  
(b) Interrupt-driven I/O

Muhammad Adri, MT

## Direct Memory Access

- Transfers a block of data directly to or from memory
- An interrupt is sent when the task is complete
- The processor is only involved at the beginning and end of the transfer



(c) Direct memory access

