Chapter 1
Contents

- Basic elements
- Processor registers
- Instruction execution
- Interrupts
- Memory hierarchy
- Cache memory
- I/O communication techniques
Operating Systems

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

- Processor
- Main Memory
  - referred to as real memory or primary memory
  - volatile
- I/O modules
  - handles external environment
    - secondary memory devices
    - communications equipment
    - terminals
- System interconnection
  - structure that provides for communication among processors, memory, and I/O modules
Top-Level Components

Figure 1.1 Computer Components: Top-Level View

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
Top-level Components (Cont’d)

- **MAR - Memory Address Register**
  - address for next read or write

- **MBR - Memory Buffer Register**
  - data to be written into memory
  - data read from memory

- **I/OAR - I/O Address**
  - specifies a particular I/O device

- **I/OBR - I/O Buffer**
  - used for the exchange of data between an I/O module and the processor
Processor Registers

- Memory that is faster and smaller than main memory
- Temporarily stores data during processing
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
Processor Registers

User-visible registers
- May be referenced by machine language
- Available to all programs - application programs and system programs

Types of registers
- Data
- Address
- Condition Code
User-Visible Registers

- **Data Registers**
  - can be assigned by the programmer

- **Address Registers**
  - contain main memory address of data and instructions
  - may contain a portion of an address that is used to calculate the complete address
    - index register
    - segment pointer
    - stack pointer
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 the top of the stack
User-Visible Registers

- **Condition Codes or Flags**
  - Bits set by the processor hardware as a result of operations
  - Can be accessed by a program but not be changed
- **Examples**
  - positive result
  - negative result
  - zero
  - overflow
  - carry
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
**Instruction Execution**

- Processor executes instructions in a program
- Instructions are fetched from memory one at a time
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
Instruction Register

- Fetched instruction is placed here
- 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
Example of Program Execution
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 from the I/O task
- Processor is free to do other things
Interrupts

- An interruption of the normal sequence of execution
  - 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

- Improves processing efficiency
  - Allows the processor to execute other instructions while an I/O operation is in progress
Figure 1.5 Program Flow of Control Without and With Interrupts

(a) No interrupts
(b) Interrupts; short I/O wait
(c) Interrupts; long I/O wait
Classes of Interrupts

- Program
  - arithmetic overflow
  - division by zero
  - execute illegal instruction
  - reference outside user’s memory space
- Timer
- I/O
- Hardware failure
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
Instruction Cycle with Interrupts
Interrupt Cycle

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

Device controller or other system hardware issues an interrupt

Processor finishes execution of current instruction

Processor signals acknowledgment of interrupt

Processor pushes PSW and PC onto control stack

Processor loads new PC value based on interrupt

Save remainder of process state information

Process interrupt

Restore process state information

Restore old PSW and PC
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
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
Multiple Interrupts

Figure 1.12 Transfer of Control with Multiple Interrupts
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
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
**Disk Cache**

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

- Invisible to operating system
- Increase the speed of accessing memory
- Processor speed is much faster than memory speed
Cache Memory

Figure 1.16 Cache and Main Memory
Cache Memory

- Contains a portion of main memory
- Processor first checks the cache
- If not found in cache, the block of memory containing the needed information is moved to the cache
Cache/Main-Memory Structure

(a) Main Memory

- Memory Address
- Block \((k\) words)\
- Word Length \(2^n - 1\)

(b) Cache

- Slot Number
- Tag
- Block Length \((k\) words)
Cache Design

- **Cache size**
  - reasonably 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
Cache Design

Mapping function
- determines which cache location the block will occupy

Replacement algorithm
- determines which block to replace
- Least-Recently-Used (LRU) algorithm
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
Programmed I/O

- I/O module performs the action
- Sets appropriate bits in the I/O status register
- No interrupts occur
- Processor checks status until operation is complete
Programmed I/O

1. Issue Read command to I/O Module
2. Read Status of I/O Module
3. Check Status
   - Not Ready: Error Condition
   - Ready: Read word from I/O Module
4. Write word into memory
5. CPU Memory
6. Done?
   - No: Next Instruction
   - Yes: Next Instruction
Interrupt-Driven I/O

- Processor is interrupted when I/O module is 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
Interrupt-Driven I/O

1. Issue Read command to I/O Module
2. Read Status of I/O Module
3. Check Status
4. If Ready, proceed to Read word from I/O Module
5. Read word from I/O Module
6. Write word into memory
7. Check if done?
   - Yes: Next Instruction
   - No: Repeat steps from Issue Read command to I/O Module
8. Error Condition: Interrupt
9. CPU I/O
10. Do something else
11. Ready

CPU I/O

I/O CPU

CPU Memory
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
Direct Memory Access

Issue Read block command to I/O module

Read status of DMA module

CPU → DMA
→ Do something else

Interrupt

DMA → CPU

Next Instruction