Computer Organization & Software

(COEN 311)

Anjali Agarwal
Concordia University
Office: EV05.157

Acknowledgements

Material used in this course based on:

- C. Hamacher, Z. Vranesic and S. Zaky, “Computer Organization”,
- P. E. Livadas and Ch. Ward, “Computer organization and the MC68000
- D. A. Patterson and J. L. Hennessy, “Computer Organization & Design”
- William Stallings, “Computer Architecture and Design”

Course Contents

- Basic foundation of computer system material
- Principal components of a computer system
- Introduction to microprocessors - core of computer computation power
  - » *machine language and assembly language programming*
- Addressing modes and those for MC68000
- More detailed instructions for MC68000
- Subroutines
- Exception processing including internal and external interrupts
- Introduction to operating systems

Basic concepts - Algorithms

- ***Algorithm*** is an ordered sequence of steps terminating in finite time. Each step satisfies properties:
  - » *Runs only for* finite *time*
  - » *Is* computable, *i.e., it has well defined answer*
- Algorithms not necessary associated only with scientific (engineering) work
  - » *Every day tasks also presented in some form of algorithms, even if we are not aware of that*
    - Instructions of installation and operation of DVD players
    - Starting and driving a car (according to driving rules)
- Algorithms for computer execution must be formulated precisely, and in systematic, ordered way
  - » *Algorithms' presentation must be clear, and easy to verify for computer programmers even if programmers are not familiar with engineering/scientific area concerning algorithms*
    - *Example*: Programmers do not have to be experts in DSP in order to encode echo canceling algorithm

Basic concepts – Flow Charts

- Algorithms are often presented visually in form of flowcharts
- Flowcharts are types of *oriented graphs*, where each node represents a single instruction of algorithm
- Graphically, flowcharts are represented by boxes
  - *Three types of instruction boxes:*
    - ***State box***: *Rectangular*
    - ***Decision box***: *Diamond*
    - ***Conditional output box***: *Oval*

Basic concepts – Pseudocode

- Another way of presenting algorithms is in terms of pseudo code
- pseudocodes are commonly used to represent any arithmetic algorithms
- Example:

If student's grade is greater than or equal to 50
    Print "Passed"
Else
    Print "Failed"

Basic concepts – Programming Language

- Algorithms can be written in specific form, called programming language

  Like any spoken language, programming languages follow set of rules (grammar) and use specific symbols to denote data (words)

  Software languages like C/C++ widely used to describe all sort of algorithms from engineering and computer science applications

- High level programming languages
  - » *Speeds up program development time*
  - » *Provides more readable and maintainable programs*
  - » *Relieves user of system-dependent details*

- High level programming languages too abstract for machines to understand, and execute
  - » *Sophisticated translators needed that convert it into machine language that computer understands*

Basic concepts – Machine Language

- Less sophisticated, but more machine readable for computers are machine codes
  - » *Consists of sequences of 0's and 1's*
  - » *Specifies the operation, its size, and its access to data*

Sum = Sum + 5

For High-level language, it means *"add the 32-bit value* 5 *to the variable* SUM *and store the result in* SUM*"*

For Motorola 68000, it means *"add the integer value* 5 *to the memory address* SUM*"*

Machine Code:

0000 0110 1011 1001	Machine instruction: add a number to contents in memory
0000 0000 0000 0000 0000 0000 0000 0101	The number to add is 5
0000 0000 0000 0000 0000 0000 0000 1000	The address in memory, which corresponds to variable SUM is 8

Basic concepts – Assembly Language

- Machine language allows direct access to internal registers of the central processing unit and memory locations
  - » *Too tedious and inefficient to be practical*
- High level language programs are easier to write but the compiled code is larger and less efficient
- In Assembly language programming, machine operations are represented by mnemonic codes (such as ADD and MOVE) and symbolic names that specify memory addresses

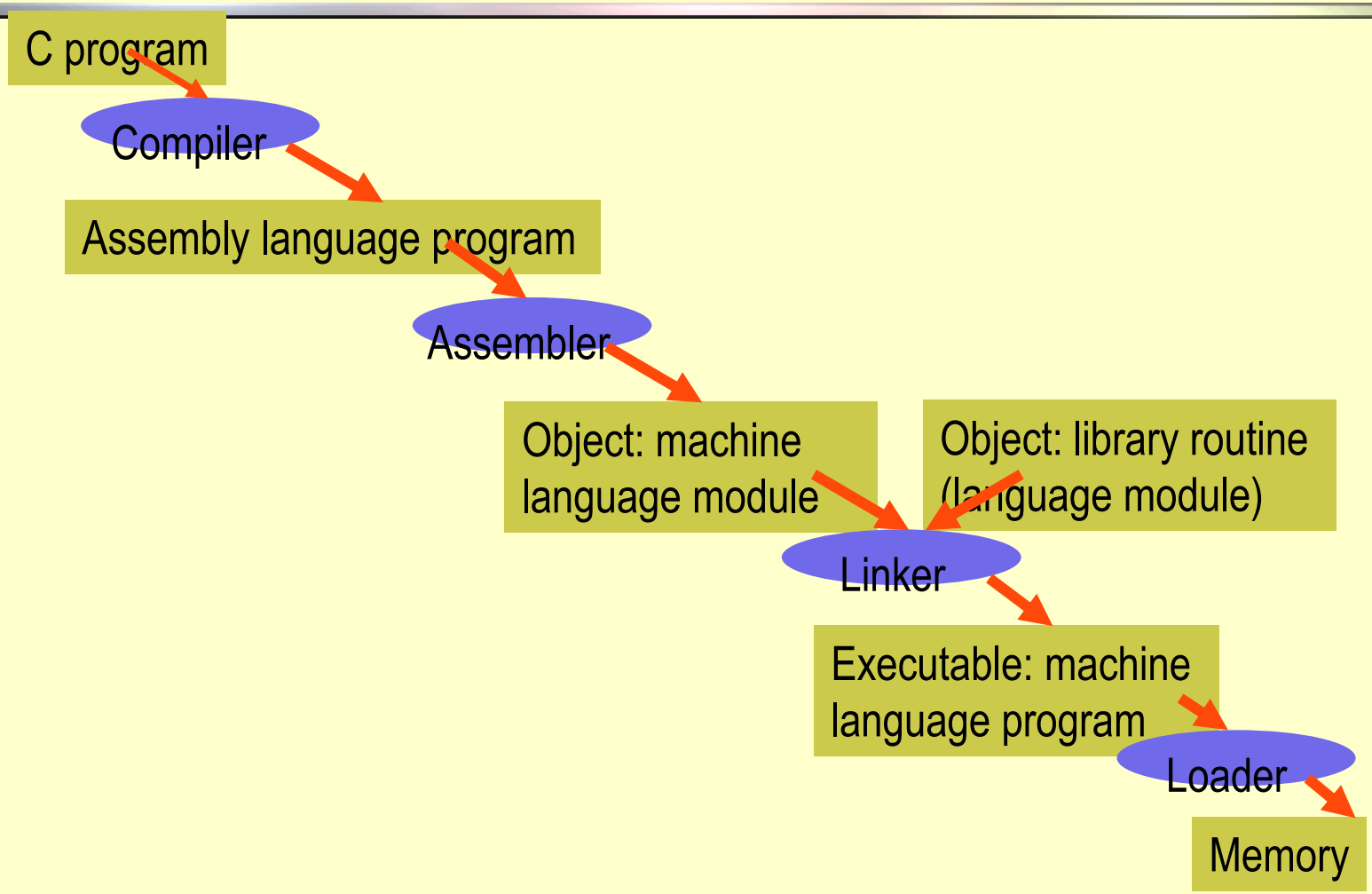
```
ADDI.L  #5, SUM
```

Basic concepts – Assembly Language

- Allows programmer to control precisely what the processor does,
- Offers a great deal of power to use all of the features of the processor,
- Resulting program is normally very fast and very compact,
- Timings can be calculated very precisely and program flow is easily controlled
- Code can be optimized for speed and storage size
- More efficient to use assembly language to communicate with peripheral devices such as printers and terminals

Learning assembly language teaches you how a computer works

Steps in Translation of C Program into Machine Instructions



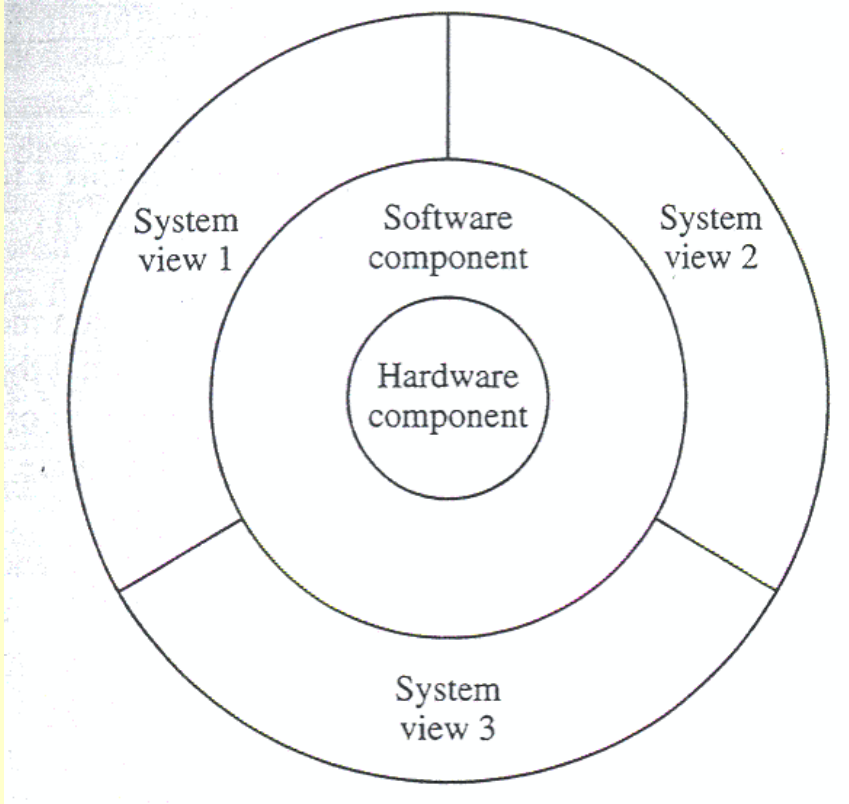
Steps in Translation of C Program into Machine Instructions (cont.)

- Compiler
  - *Transforms C program into assembly language program*
    - High level programming languages more compact and readable for users but not for machines
- Assembler
  - *Converts assembly instructions into machine language*
    - It also accepts numbers in different bases (binary, decimal, hexadecimal)
- Linker
  - *In order to save compiling time each procedure compiled individually*
    - In case of small changes to a given procedure, only it will be recompiled instead of the whole program
  - *Linker needed to combine (link) all individually compiled procedures into one program*
    - Linker produces executable file ready to run on computer
- Loader
  - *Copies code file into memory and launches the program*

Structured layers of computer system

Three principal components – hardware, software, user

Computers defined as *hardware* systems powered by *software*



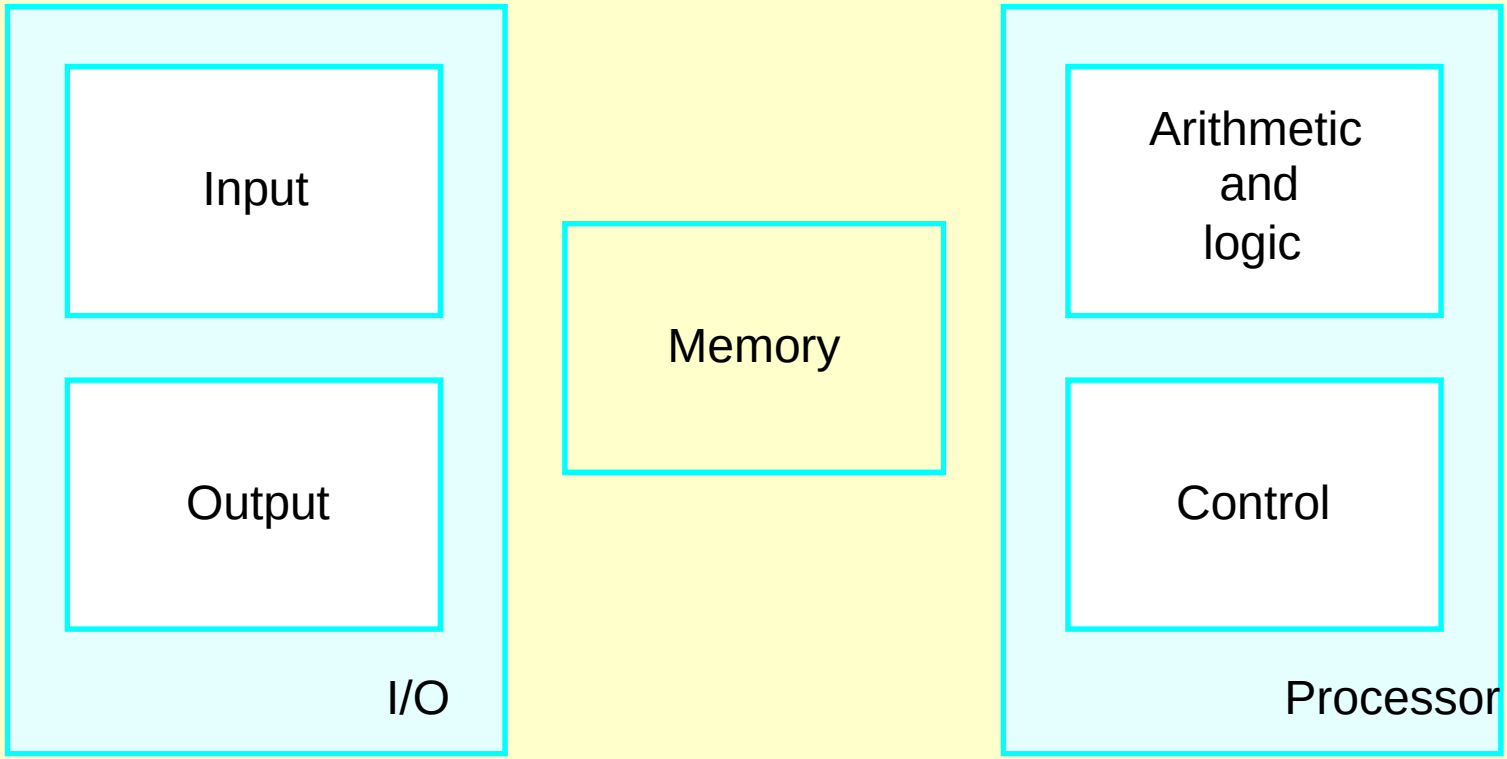
Computer System Onion

Computer Systems

– Personal Computers (PC): self contained processor, memory and storage and input-output (I/O) hardware. Networking common.
– Workstations: Systems with higher computational power (processor, memory and storage augmented when compared to PC). Networking very common.
– Mainframes: Systems designed for large data management and high power computing. Networked almost always.

• All of these systems have similar functional units which allow them to perform their respective tasks.

Hardware



Basic functional units of a computer.

Functional Units

Primary components - Central Processor (ALU and Control), Memory (Primary and Secondary), and Input/Output

Memory:

- Primary storage – all programs (instructions) are loaded before they are executed
  - » Instructions and data stored as sequence of bits
  - » Main memories are usually volatile but accessed at high speeds (ns)
- Secondary storage – nonvolatile where programs and data can be stored when not in use (optical discs, magnetic discs and tapes), these are slower access, less expensive, larger size.

I/O devices (keyboard, screens)– to load program into main memory

- *Enable communication between computer and outside word*

Controller – an I/O program that interprets and executes I/O commands

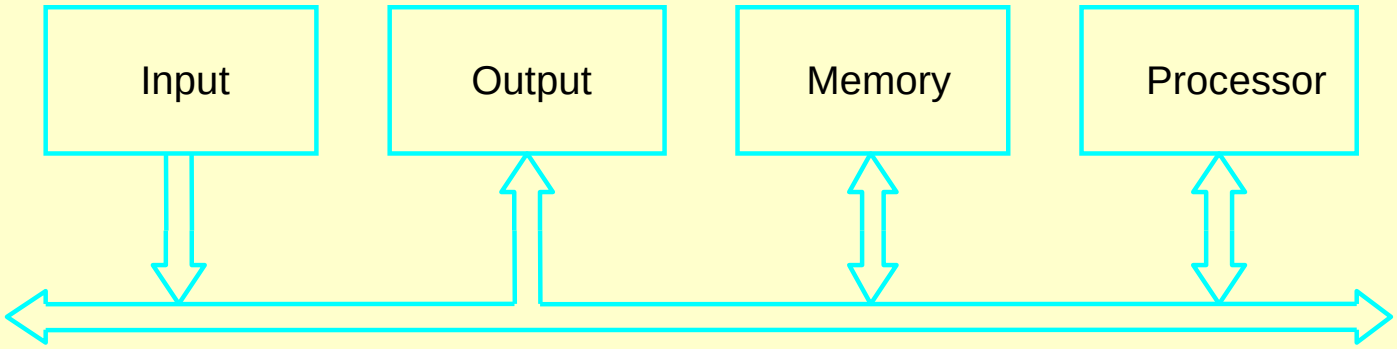
Central Processing Unit – interprets and executes the machine language instructions and supervises the transfer of data between primary and secondary storage

ALU – for execution of computer operations such as addition, comparison, etc.

Control unit – to coordinate the operations of other units, timings of these operations are governed by signals from the control unit

Bus Structures

Computer Architecture describes the way these components are connected and the manner in which they communicate



Single-bus structure.

n - bits (a word) of data are transferred in parallel by the bus

In addition the bus have lines for address and control purposes.

- *Address lines determine which two devices can use the bus*
- *Control lines determine the type of operation*

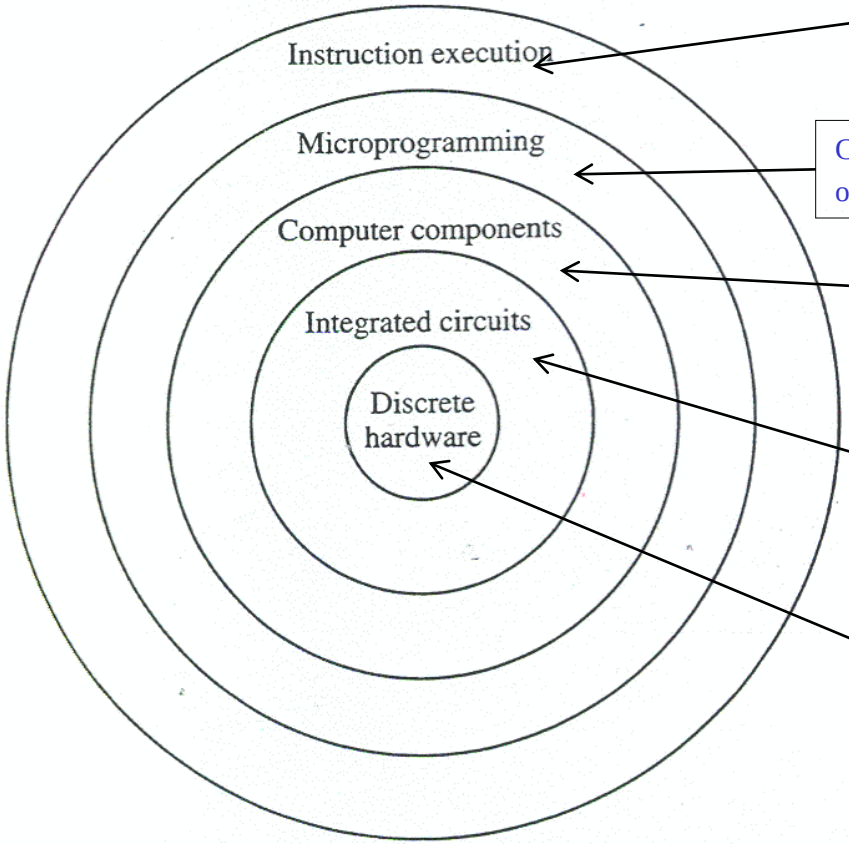
Basic Operational Concepts

- Program consists of list of instructions stored in memory
- Data to be used also stored in memory
- Individual instructions brought from memory into processor to be executed

Add LOCA, R0

  - *Fetch the instruction from memory*
    - Address of the memory location for the instruction
  - *The operand at LOCA is fetched and added to contents of R0*
  - *The resulting sum is stored in R0*

Inner layers of hardware



User sees the instruction set
(machine language instructions)
- highest layer of hardware view

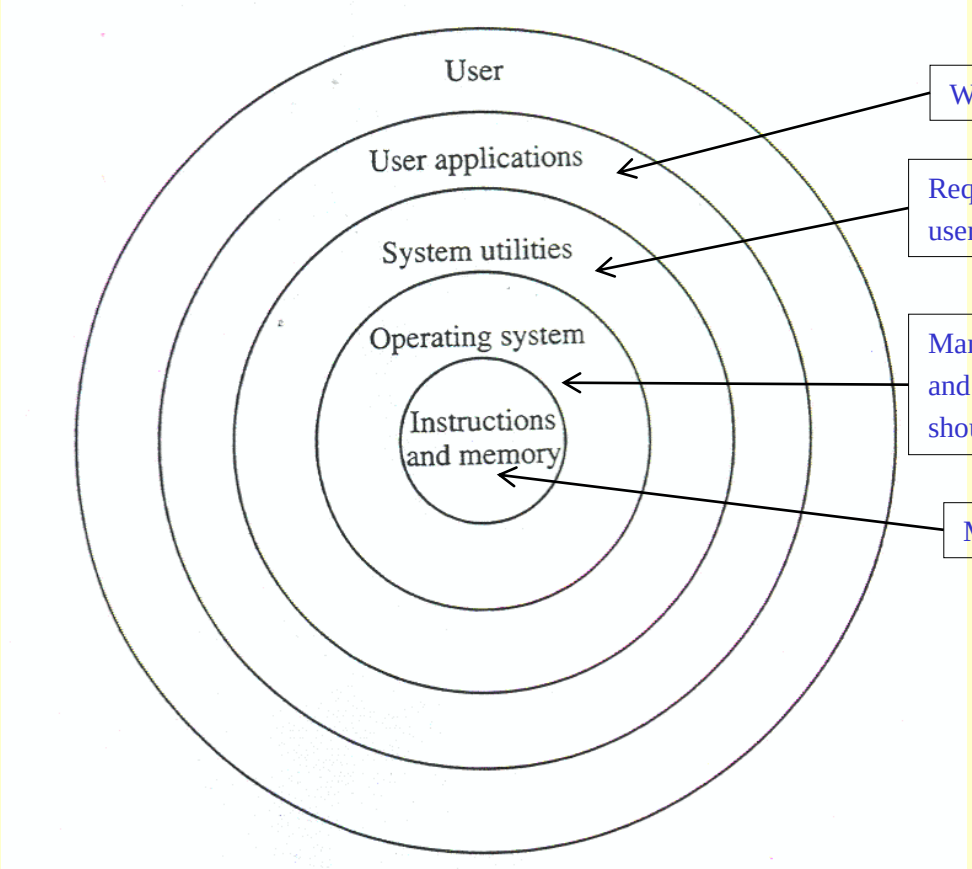
Central processor may be composed
of several micro-programmable components

CPU, Memory Devices

Gates (OR, AND, etc.)
Groups of gates (ICs)
- adders, shift registers
- ALU
- microprocessor

Transistor as primary component

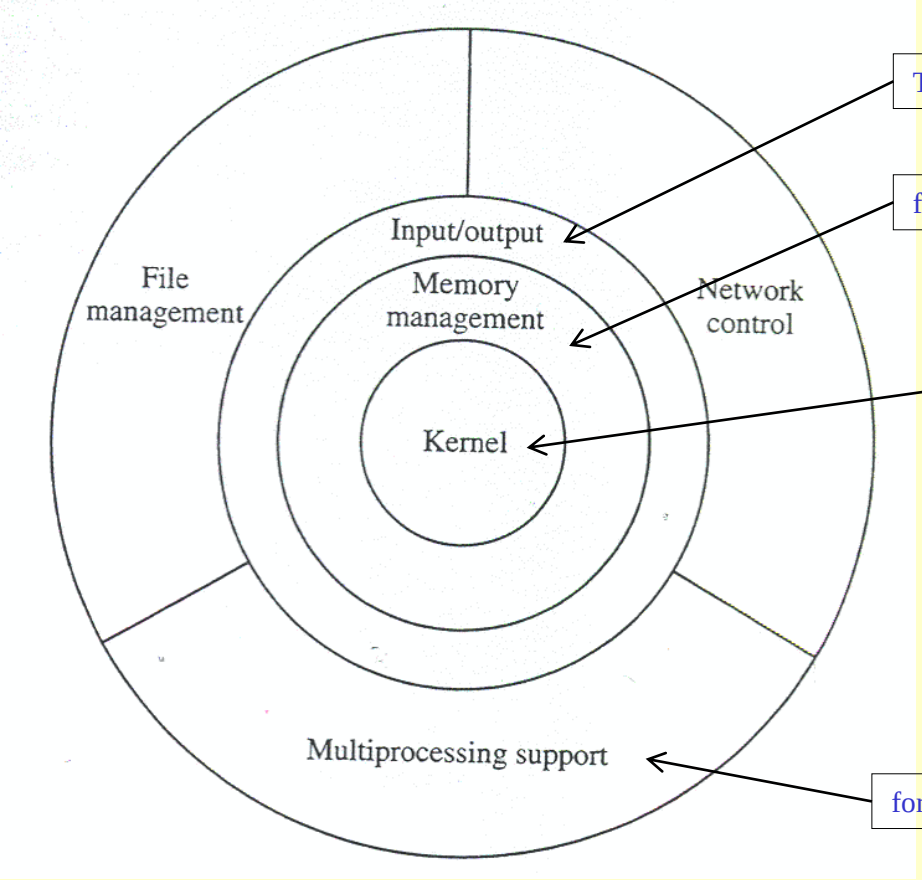
Software Layer



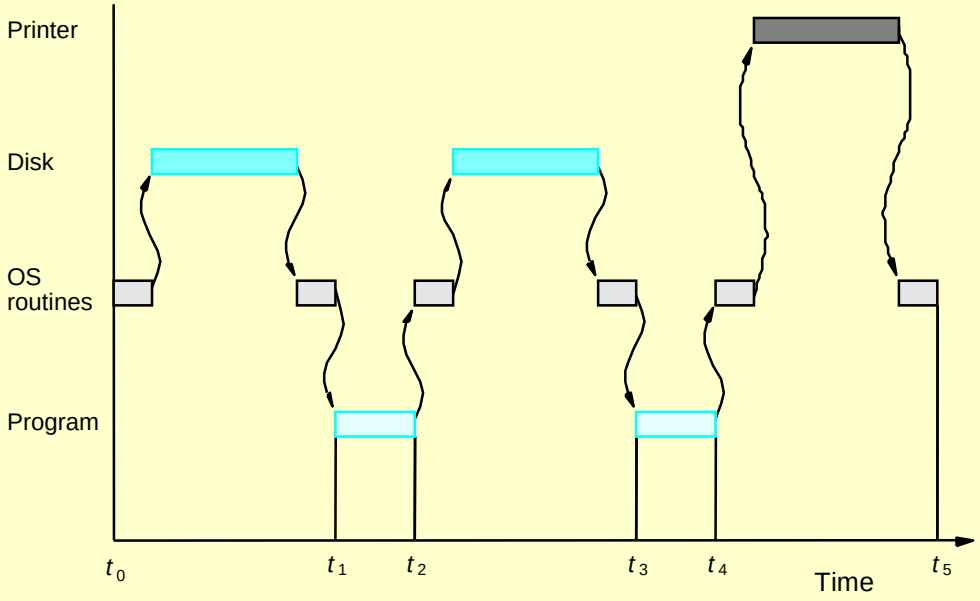
Software

- System Software – collection of programs needed to perform
  - » *Receiving and interpreting user commands*
  - » *Entering and editing application programs – text editor*
  - » *Running word processors, spreadsheets, etc.*
  - » *Controlling I/O*
  - » *Compilers/assemblers to translate programs from source code to object form as machine instructions*
  - » *Linking and running user-written application programs with existing standard library routines*
- System Software is responsible for coordination of all activities in a computer system
- Operating System (*OS*) - a key system software component
  - » *Assigns computer resources to individual application programs*
    - – Assigns memory and disk space to program and data files
    - – Move data between memory and disk units
    - – Handle I/O operations

Layers of Operating System



Basics of OS



User program and OS routine sharing of the processor.

Elapsed time: $t_5 - t_0$

Performance Criteria

- While there may be some debate about the best measure of system performance, there is general agreement that the faster a computer system can provide correct results, the better the performance.
- Elapsed time is one of the measure of the performance – affected by speed of processor (processor time), disk and printer.
- Processor time depends on the hardware (processor and memory connected by bus) involved in execution of individual machine instructions
- Use of cache memory minimizes movement of data between main memory and processor

Basic Performance Equation

- The performance parameter T is proportional to the product of N machine language instructions with an average number of steps per instruction of S and is inversely proportional to the clock rate R in cycles per second.

$$T = \frac{(N \; x \; S)}{R}$$

- This parameter is tested by running a program with N instructions with an average number of instructions per step of S. Each step is executed in one clock cycle.

Basic Performance Equation

- A low value of T corresponds to high performance. Performance can be improved by:
  - Having a compiler that will make the source code use fewer instructions. (Reduce N)
  - Having each instruction use a small number of basic steps. (Reduce S)
  - Having a high clock speed processor (Increase R)
- The above parameters are not independent and a change in one may affect the others.

Multiprocessors and Multicomputers

- Figure 12.2 (shared-memory multiprocessor system)
- Figure 12.4 (message-passing multicomputers)

Multiprocessor Multicomputer