Chapter 4

Programming and the Program Execution Process

Outline

- **Computer Programs**
  - **Program execution and associated hardware**
- **The Programming Process**
- **The Operating System**
- **Programming languages**

Program

- **The behaviour of the computer is controlled by a set of step-by-step instructions called program.**
- **Every thing interesting or useful about a computer behaviour results from its program rather than the hardware it carries it.**
- **The computer processes information by executing a program stored in memory.**
- **The execution takes place with in the CPU and is controlled by the CU.**

Program Execution

- Executing a program requires the CPU to examine each program instruction in memory and send out the command signals required to perform each instruction.
- Although instructions are normally performed consecutively (sequencing), they can be skipped (branching) or repeated (looping) under program control. (Structured Programming)
- During execution, data can be entered by the operator (user), or from a saved file.
- After processing, the program output can be displayed or printed as a result.

Example…

- The sequence of instructions could be as follows :- (Sequencing)
    - Get out of bed
    - Have breakfast
    - Get dressed
    - Get into car
    - Drive to work
    - End of program

Example: Finding the square root of a number

(May use Branching and Looping)

1. Start with a guess, *g*
2. If *g*g* is close enough to *x*, then g is a good approximation of the square root of *x*, *jump to step 5*
3. Otherwise, create a new guess by averaging *g* and *x/g*.

   i.e., $g_{new} = (g_{old} + x/g_{old})/2$
4. Using this new guess, go back to step 2
5. Stop

You can easily code this algorithm, compile the source, and run the program

When you double click on an icon to run a program, here is what happens:

1. The program, which is stored inside the hard disk drive, is transferred to the RAM memory.
2. The CPU, using a circuit called memory controller, loads the program data from the RAM memory. A program is a series of instructions to the CPU.
3. The data, now inside the CPU, is processed.
4. What happens next will depend on the program. The CPU could continue to load and execute the program or could do something with the processed data, like displaying something on the screen.

The sequence of CPU steps can be expressed in pseudocode:

loop

fetch the instruction pointed by (the value in) IP

advance the instruction pointer (IP)

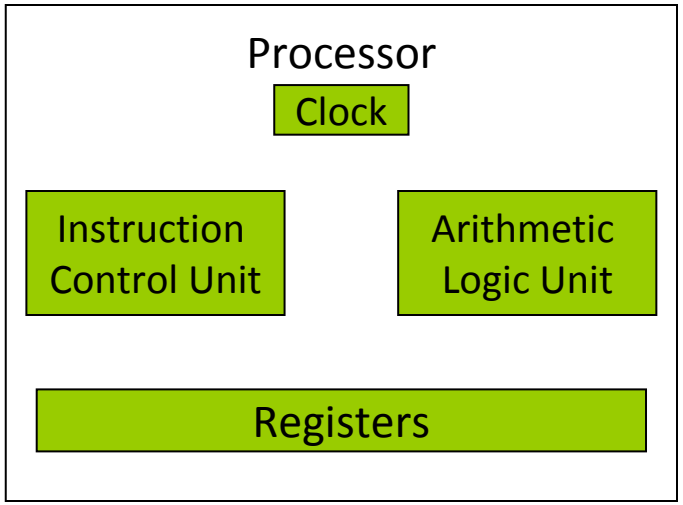
decode the instruction

if memory operand needed, read value from memory

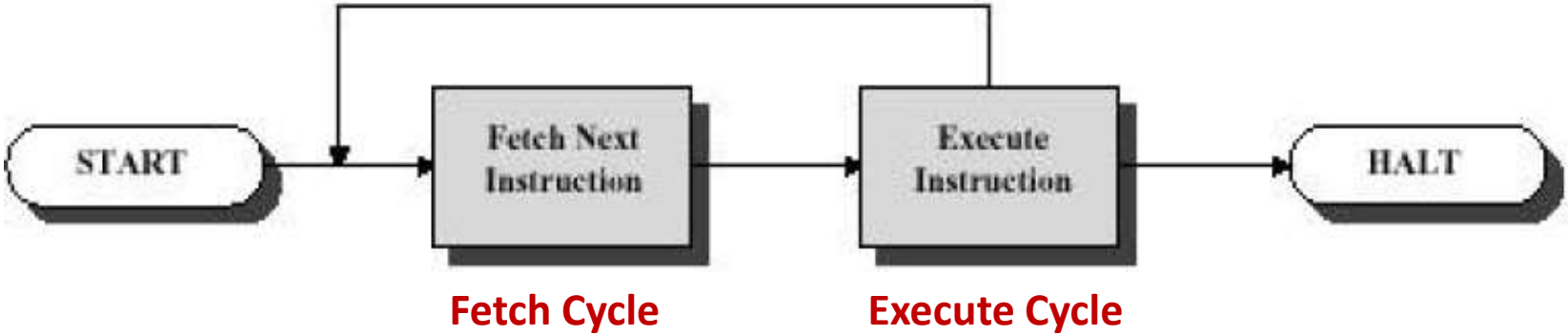
execute the instruction

if result is memory operand, write result to memory

continue loop



- Processing required for a single instruction is called an instruction cycle (Fetch-Execute Cycle), and can be viewed as shown below: **2 Steps**

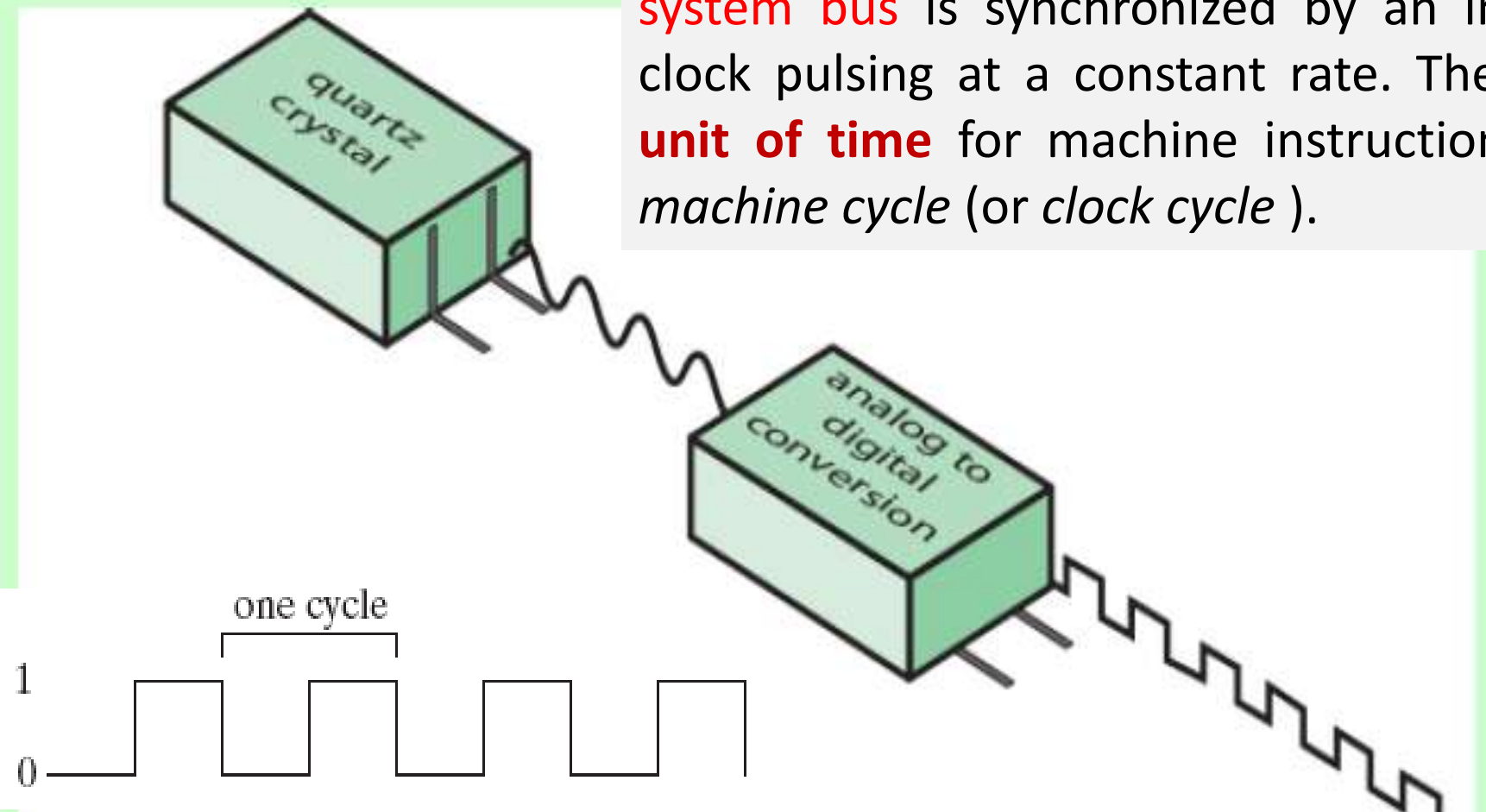


- Fetch – CPU(CU) reads an instruction from a location in memory and decodes the instruction (determine what it means)
  - Program counter (PC/Instruction Pointer) register keeps track of which instruction executes next
  - Normally, CPU increments PC after each fetch
  - Fetched instruction is loaded into the instruction register (IR)

- Execute - CPU executes the instruction
  - May involve several operations
  - May utilize previously changed state of CPU
  - General categories:
    - CPU-Memory: Data may be transferred from CPU to memory or vice-versa
    - CPU-IO: Data may be transferred between CPU and an I/O module
    - Data Processing: CPU (ALU) may perform some arithmetic or logic operation on the data
    - Control: An instruction may specify that the sequence of execution be altered

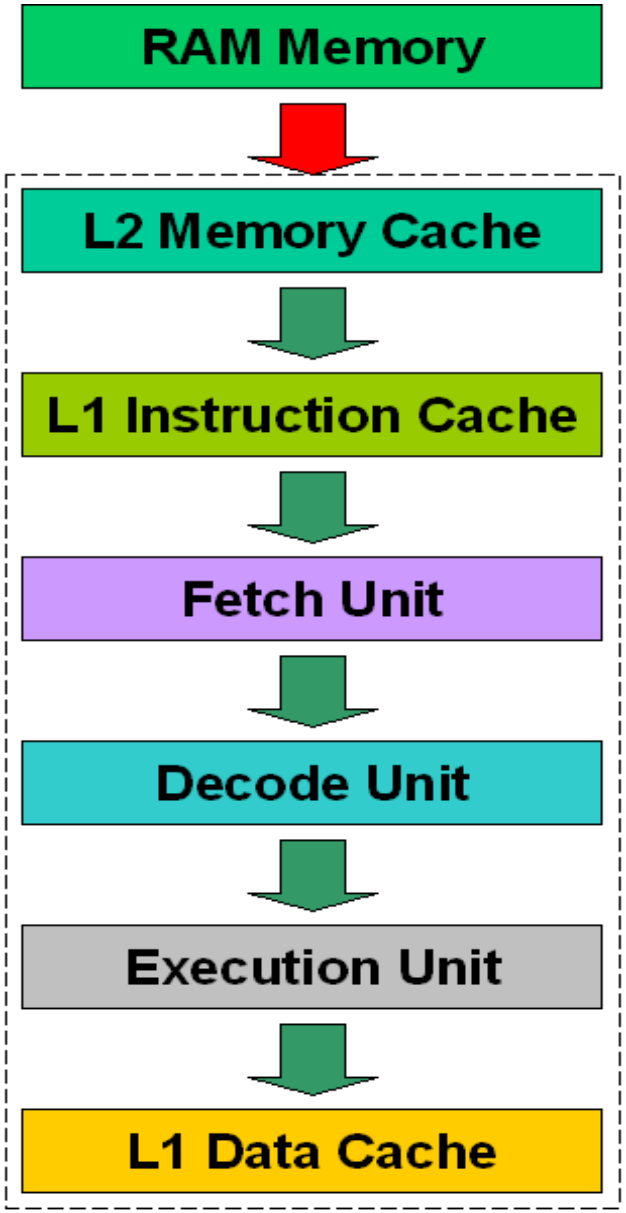
System Clock

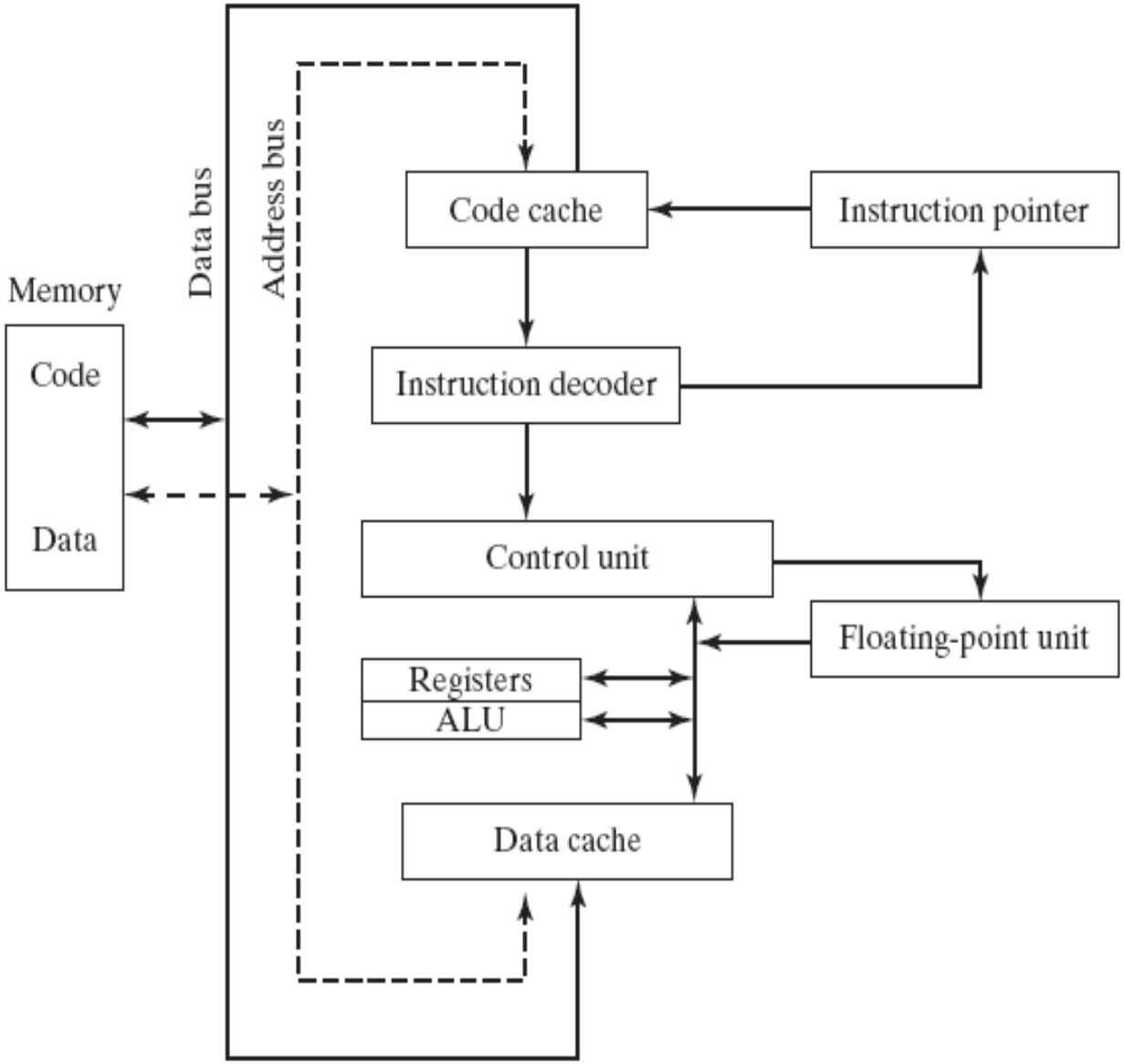
Each operation involving the CPU and the system bus is synchronized by an internal clock pulsing at a constant rate. The **basic unit of time** for machine instructions is a *machine cycle* (or *clock cycle*).



How much time it takes to execute an instruction that takes ten cpu cycles (10 clock ticks) in a 4 ghz processor?

Simplified block diagrams of a modern CPU





CPU Cont...

- Instruction execution takes place in discrete steps
  - —Fetch, decode, load and store, arithmetic or logical
  - —Usually require multiple clock cycles per instruction
- Pipelining → simultaneous execution of instructions

CU starts execution of next instruction while other instructions are still being processed in other parts of the cpu.

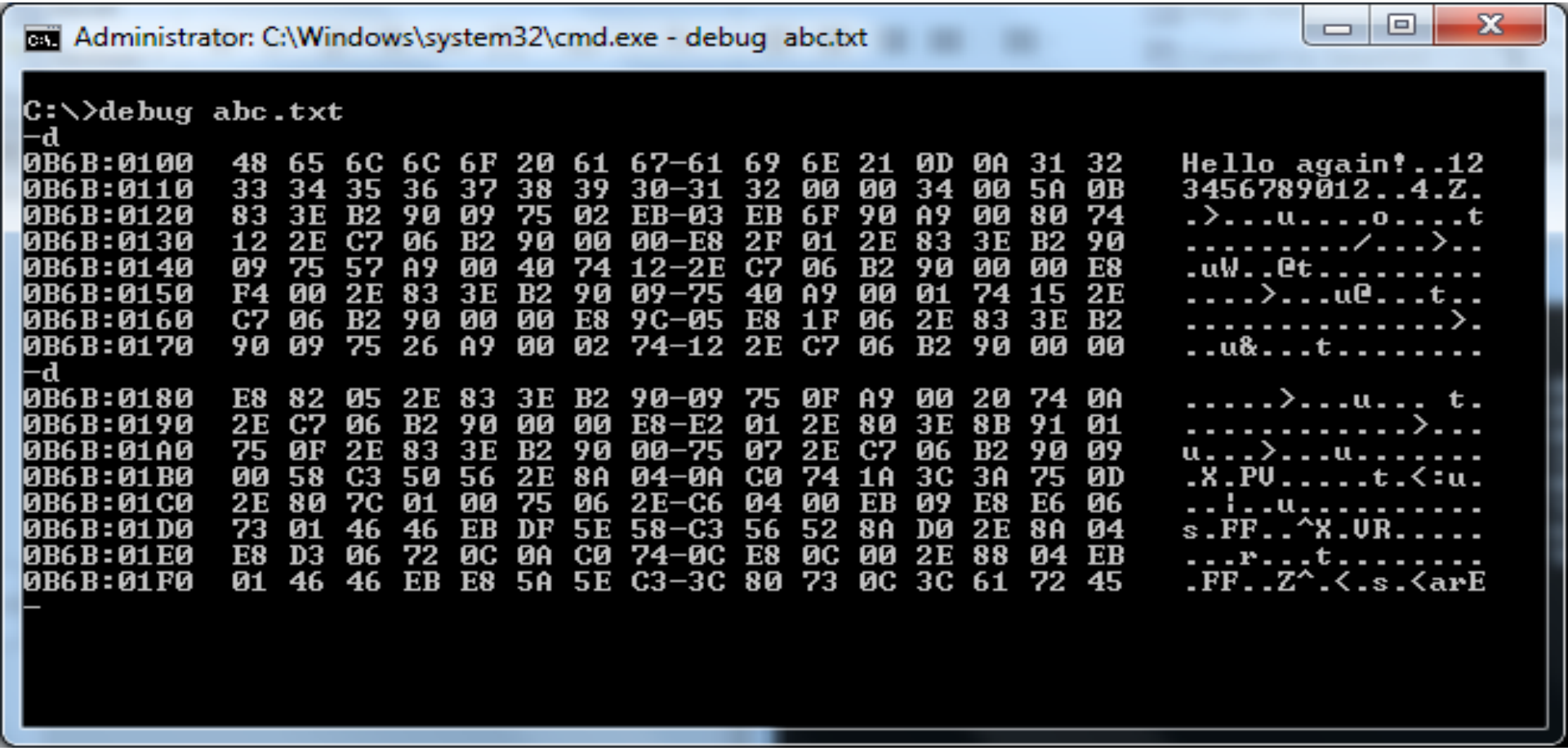
CPU Cont…

Processor speed depends on:

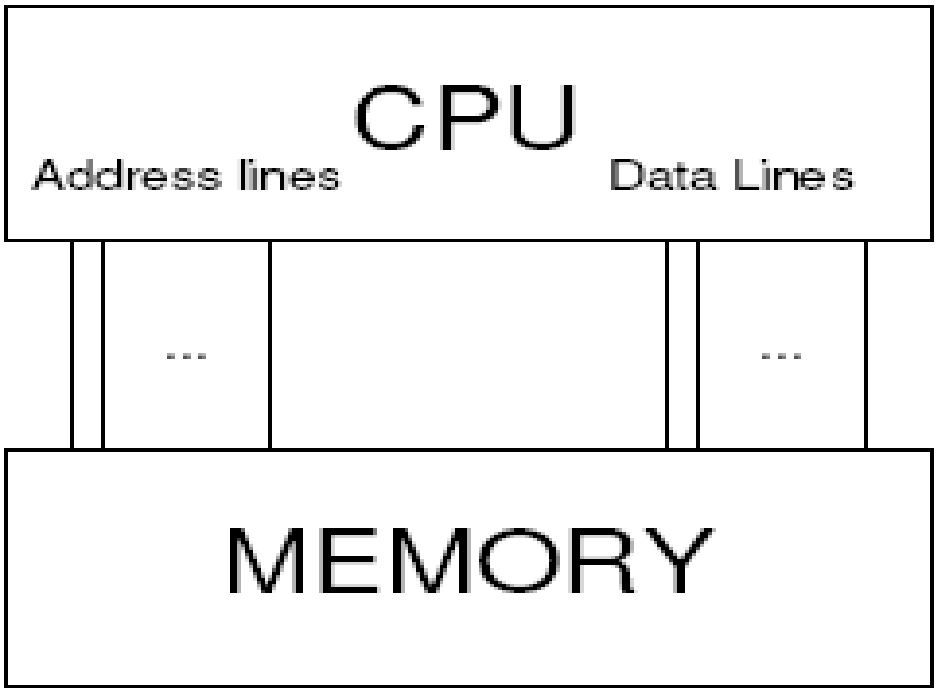
- Internal Clock Speed
- Type of Instruction Set
- Processor Implementation
- Compiler Design (efficient binary executable)
- Cache and Memory Hierarchy
- etc…

Reading Assignment: CISC & RISC, MIPS & MFLOPS

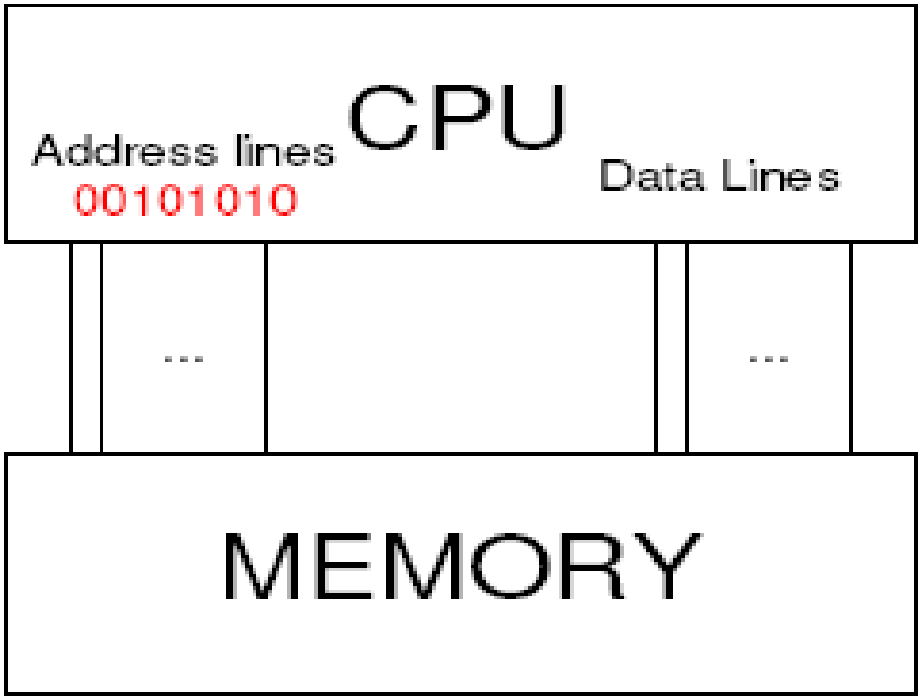
A snapshot of memory address and content (data)



CPU and Memory Interaction

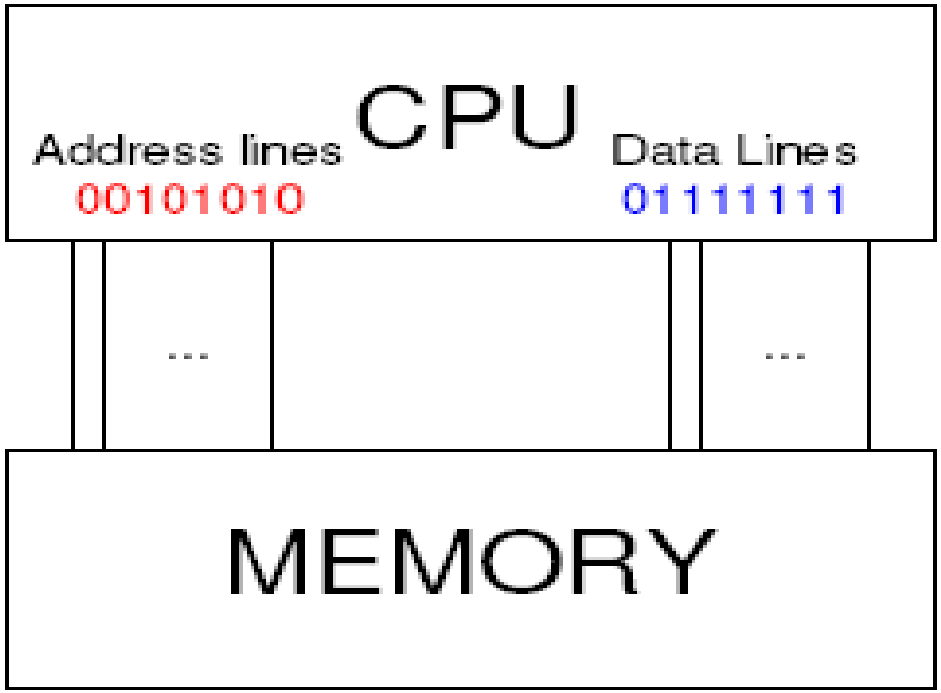


CPU Issues an Address



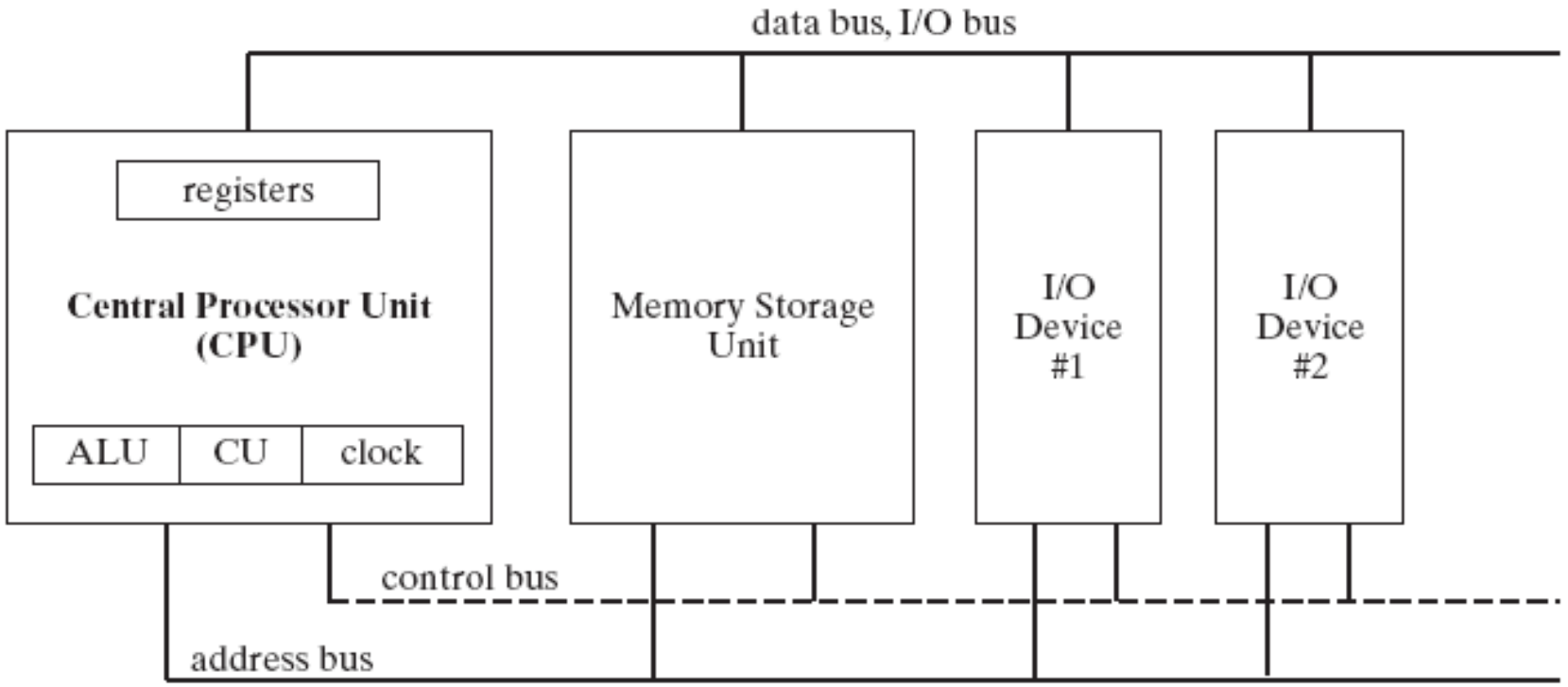
CPU: I need the contents of memory location 42

Memory makes the data available



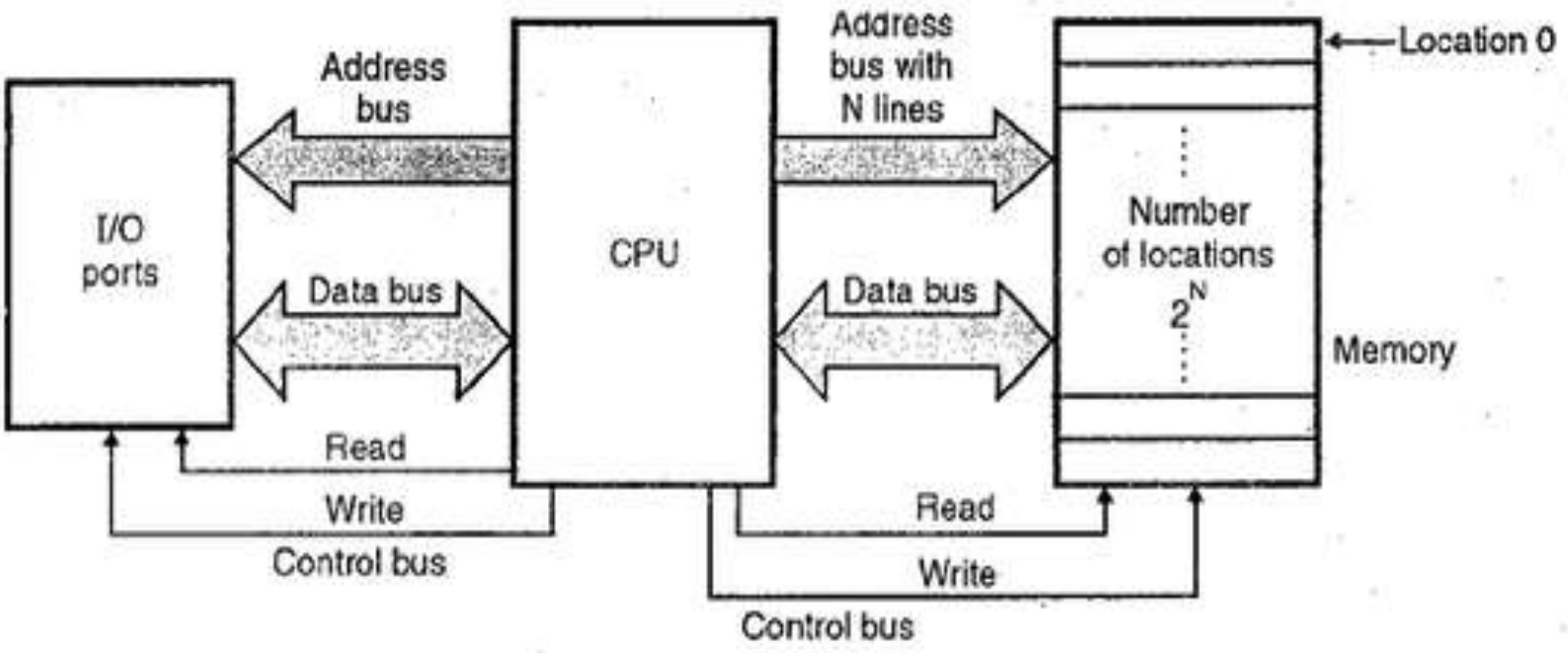
Memory: Location 42 contains the number 127

Block Diagram of a Microcomputer.



- Address bus width limits the amount of memory that can be installed in the computer

- Address bus width limits the amount of memory that can be installed in the computer



The three types of buses and their utility

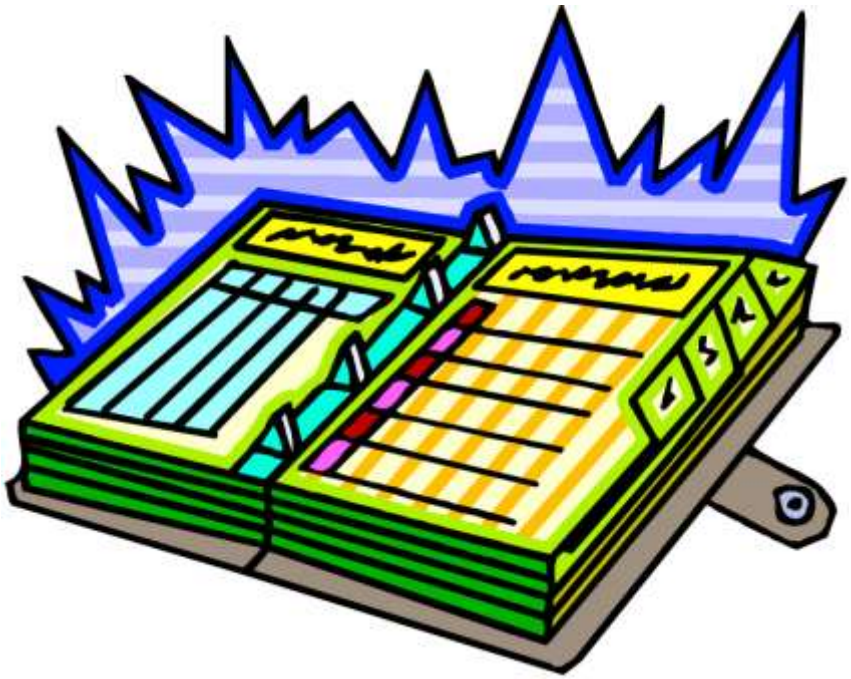
A single 1- 0 transmission is referred to as a clock cycle or bus cycle

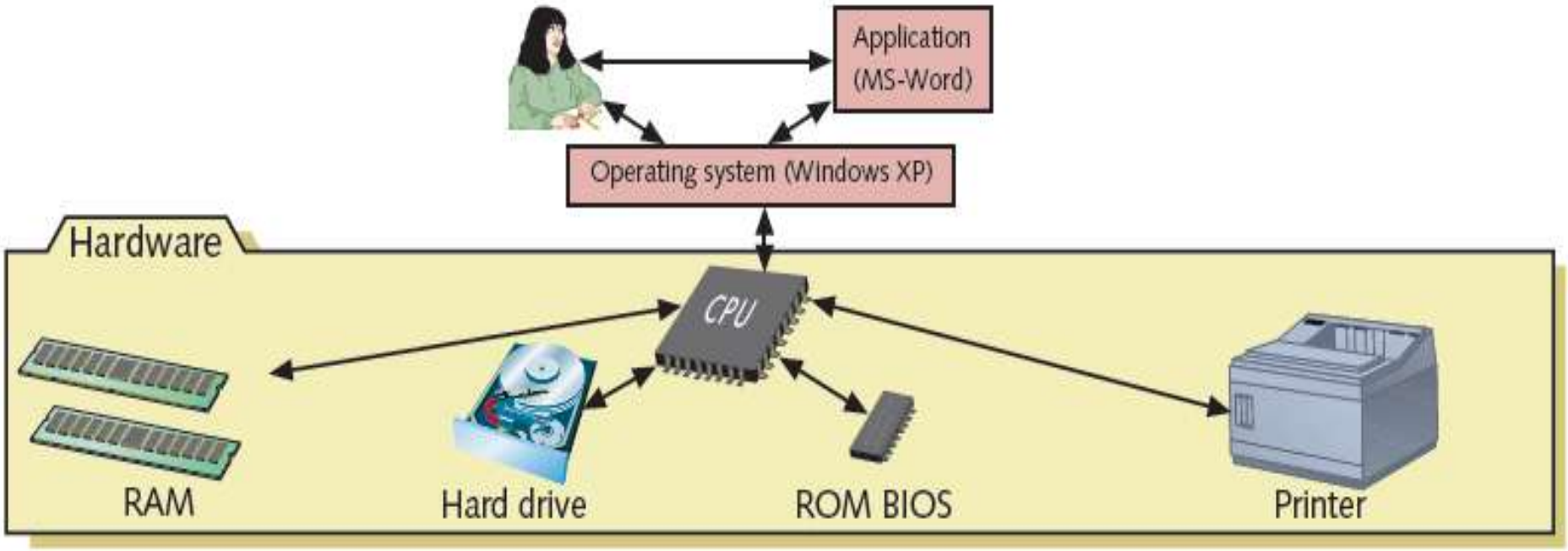
Computer Startup

- The power is turned on
  - The Basic Input/Output System (BIOS)
    - Loads from a memory chip (ROM) and executes
    - Initializes the hardware (keyboard, disk drive, mouse, etc)
  - Then loads the operating system into memory and executes it
- The Operating System waits for user input
- The user starts a program
  - By double clicking on an icon or file
  - Or by click on Start->Program->Some Program

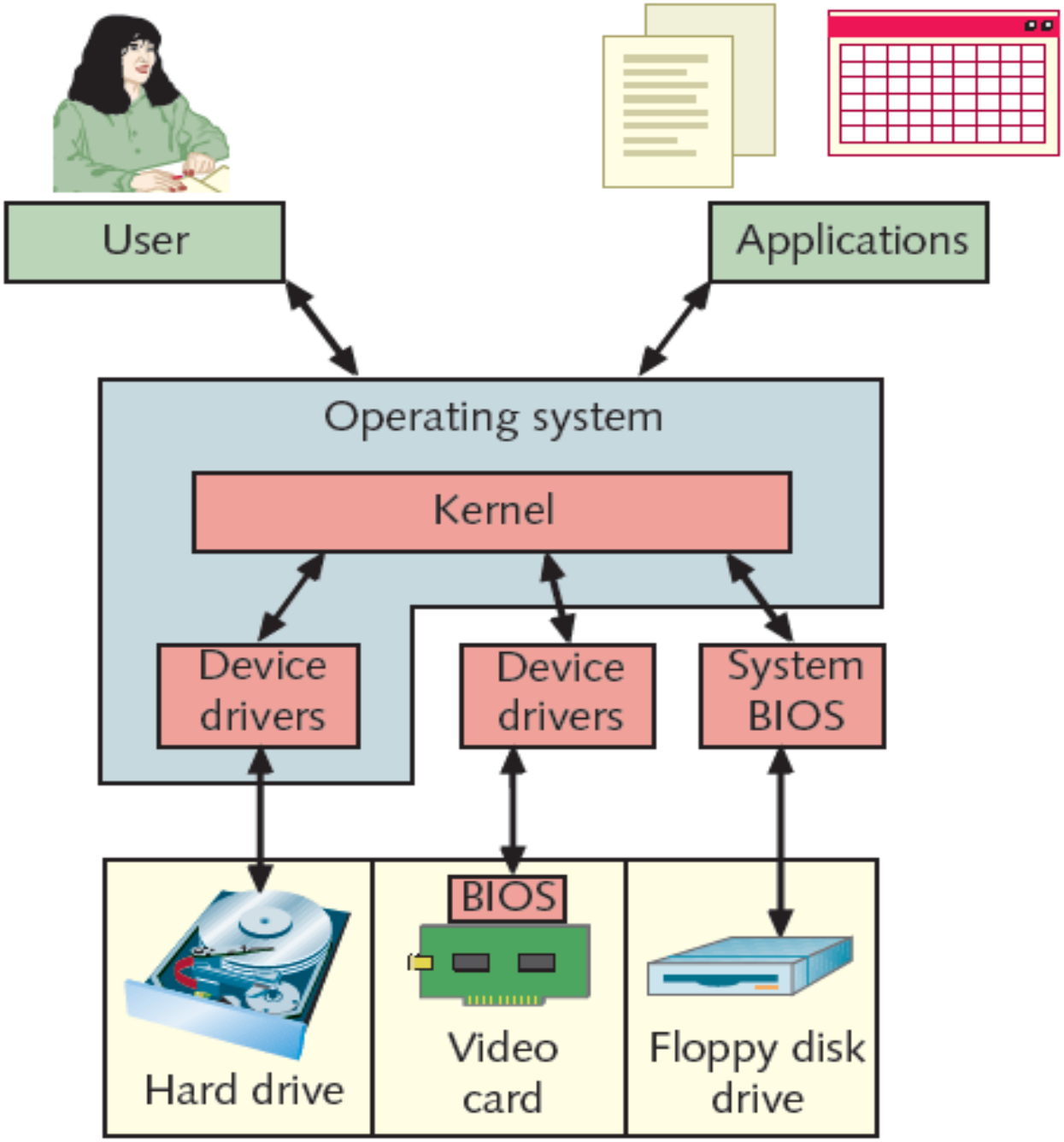
Operating System - Organizer

- Keep track of executing programs
  - Give them time with the CPU
  - A program gets a slice of time with the CPU
- Keep track of memory
  - Decide when to move some data to disk (virtual memory)
- Keep track of disk space
  - Decide where to store stuff
- Interface with the user
  - Accept input via keyboard and mouse
- Keep track of devices
  - USB drives, cameras, etc





Users and applications depend on the OS to relate to all hardware components



An OS relates to hardware by way of BIOS and device drivers

The Operating System and the Kernel

kernel: The operating system kernel is the part of the operating system that responds to system calls, interrupts and exceptions.

Ex. system("cls"); from C++ source, in <stdlib.h>

operating system: The operating system as a whole includes the kernel, and may include other related programs that provide services for applications.

This may include things like:

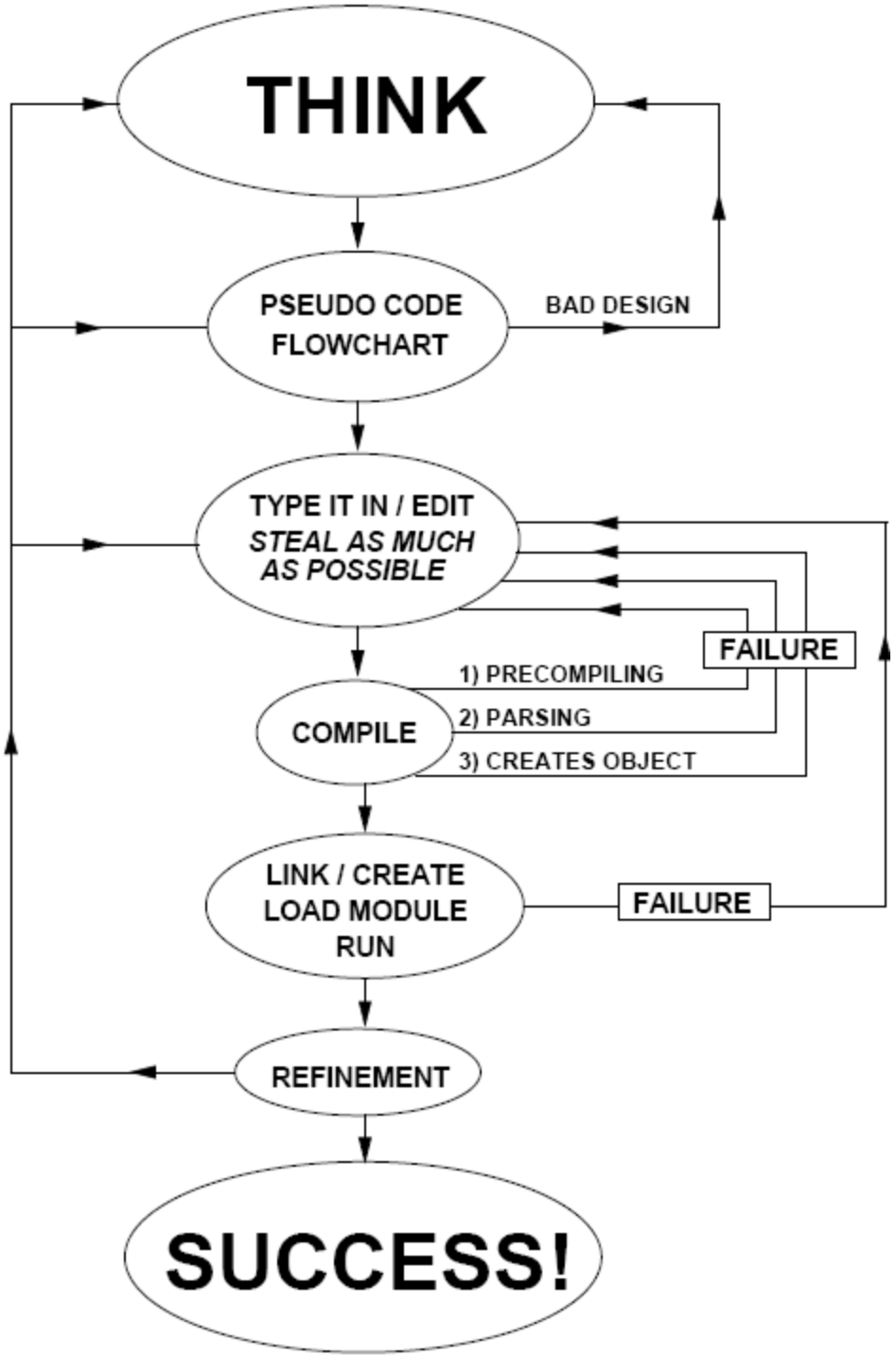
- utility programs
- command interpreters
- programming libraries

Program Execution

- The operating system reads the program into memory
  - adds it to a list of programs that want to execute
- It gives it a time slice of the CPU
  - adds it to a list of programs that are executing
- It saves the current state of the CPU when it gives another program a time slice of the CPU
  - Context switch
- It can restore the state of the CPU

Computer program development Process

This model of editing, preprocessing, parsing, object-code generation, linking, loading and running is followed by all computer code-development environments.



Compiler

- Software that translates a high-level language program into machine language.
- Input to the compiler is a **source file** (created by word processor or editor) containing the text of a high-level language program.
- If it is syntactically correct, compiler will save in an **object file** which is a machine language instructions for the same job.

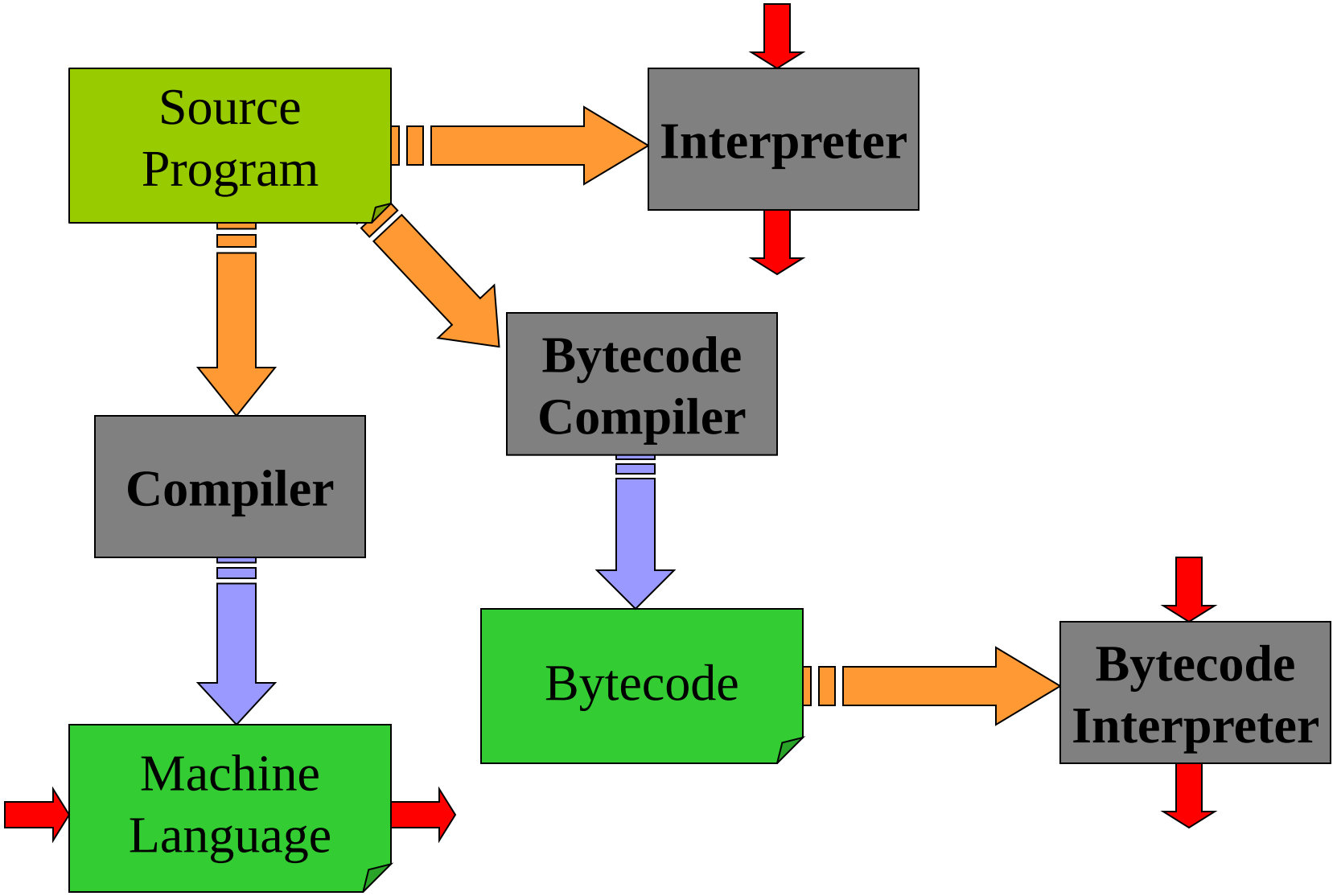
Linker

- All machine instructions are not complete.
- High-level language programs use at least one of the function that reside in other object files available to the system.
- Linker combines several object files, resolving cross references between the files, into one executable file (machine language program).

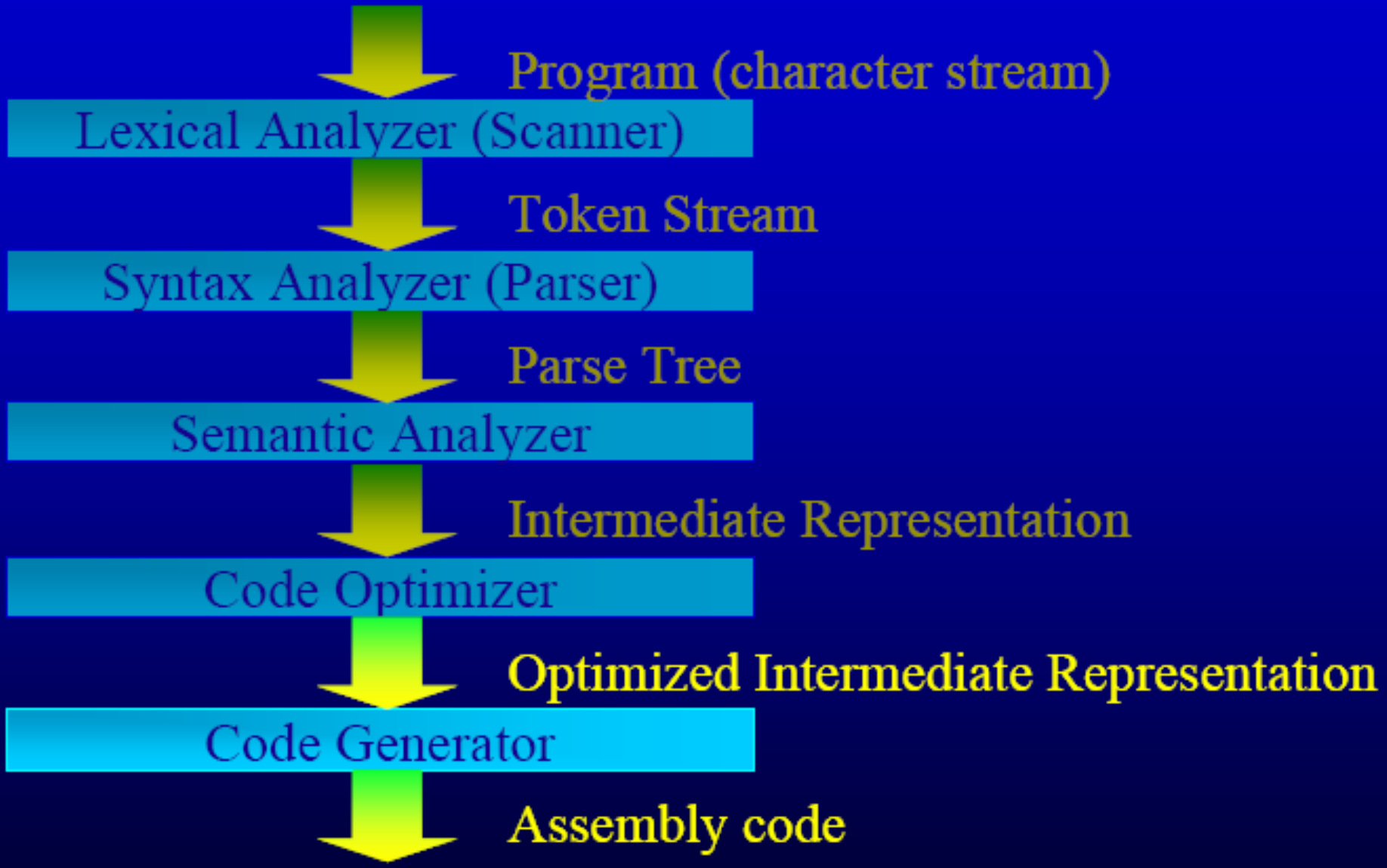
Loader

- To run an executable file, the loader must copy all the instructions into memory and direct the CPU to begin execution with the first instruction.
- As the program executes, it takes input data from source(s) and sends results to output devices.

Programs can be executed in different ways.



Anatomy of a Compiler



Classification of programming languages

Imperative

- Procedural: C, Ada, **Pascal**, **Algol**, **FORTRAN**, . . .
- Object oriented: **Scala**, C#, Java, **Smalltalk**, **SIMULA**, . . .
- Scripting: Perl, Python, PHP, . . .

Declarative

- Functional: Haskell, SML, Lisp, Scheme, . . .
- Logic: Prolog
- Dataflow: Id, Val
- Constraint-based: spreadsheets
- Template-based: XSLT

Why are there so many languages?

- Evolution.
- Special purposes.
- Personal preference.