

L01-CS8421-08-23-08

Architecture Overview P1

CS8421

Computing Systems

Dr. Ken Hoganson

Class

Will

Start

Momentarily...



CS 8421

- Look at course web page and online syllabus

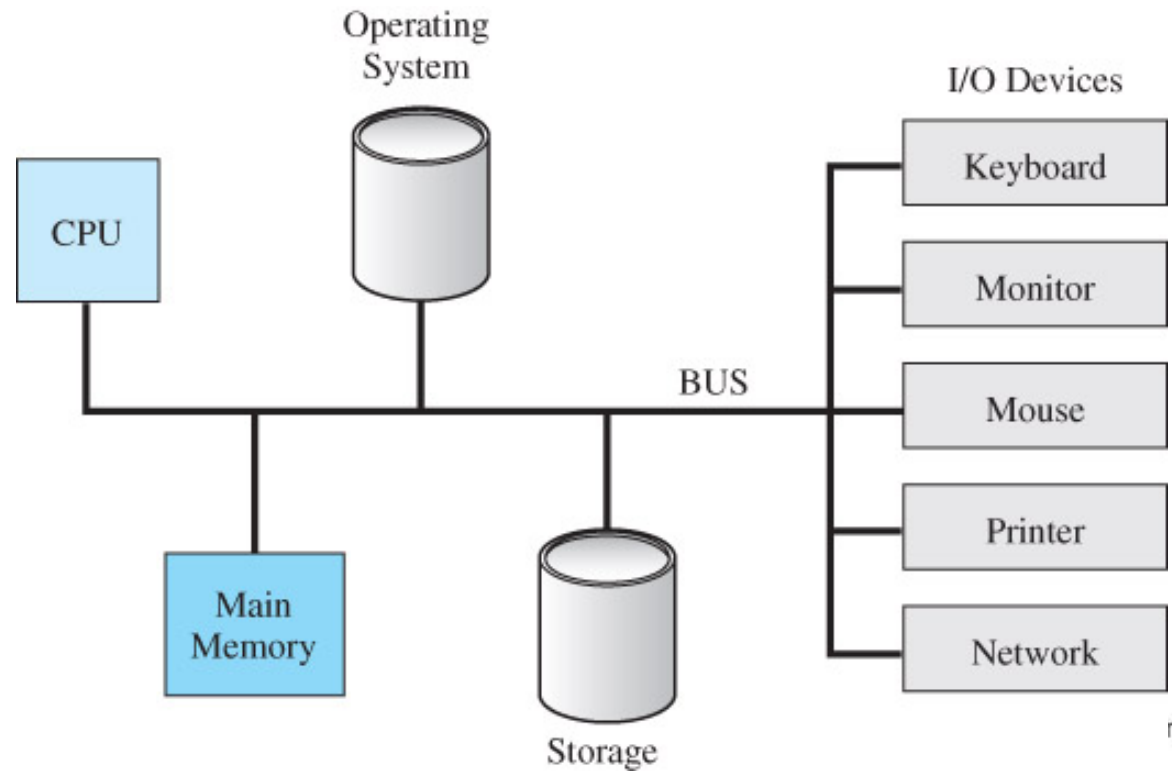


Computing System

Consists of Six Components:

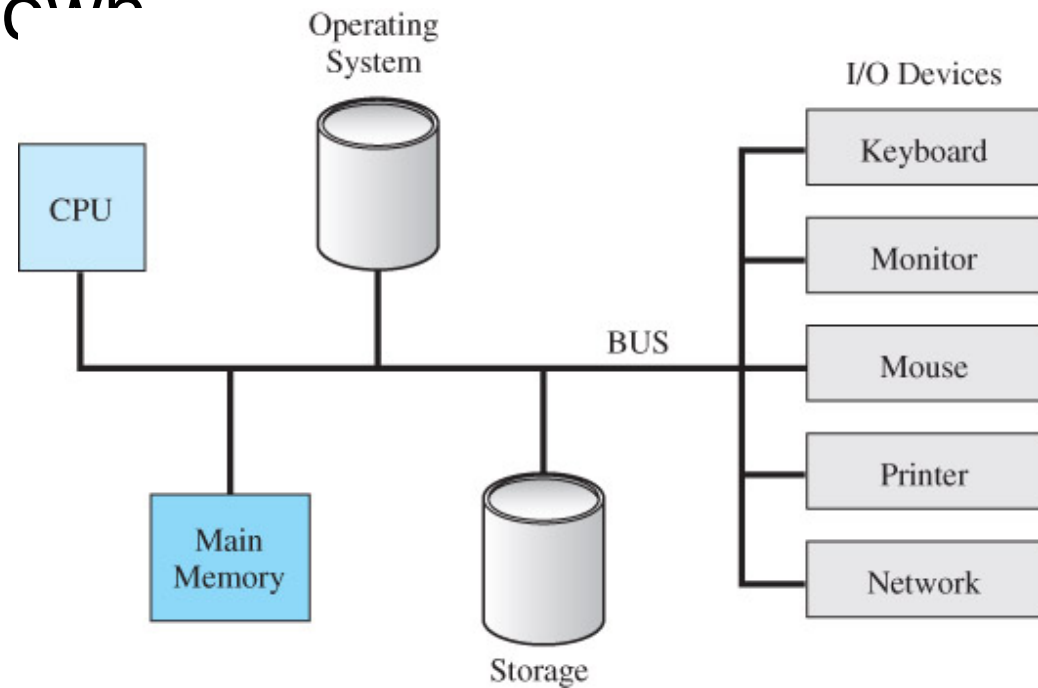
➤ CPU: The Central Processing Unit.

- The processor chip that can interpret and execute the instructions in a computer program (software).



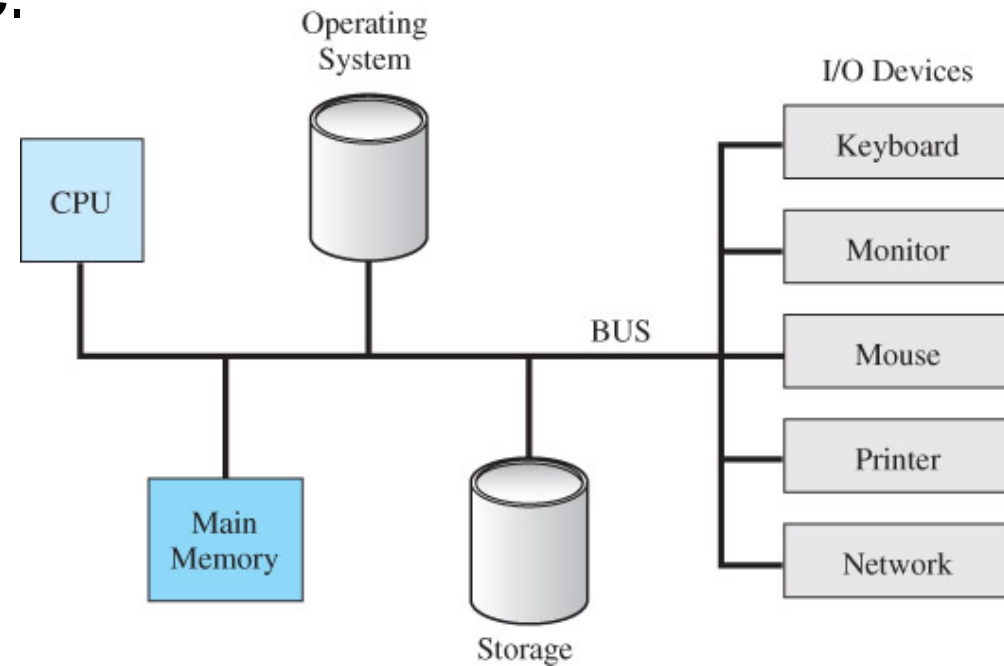
Computing System

Main Memory: This is working storage for programs and information, which is used while the computer is turned on and running. Main memory is generally not permanent or fixed storage, its contents are wiped clean when the machine is power-down.



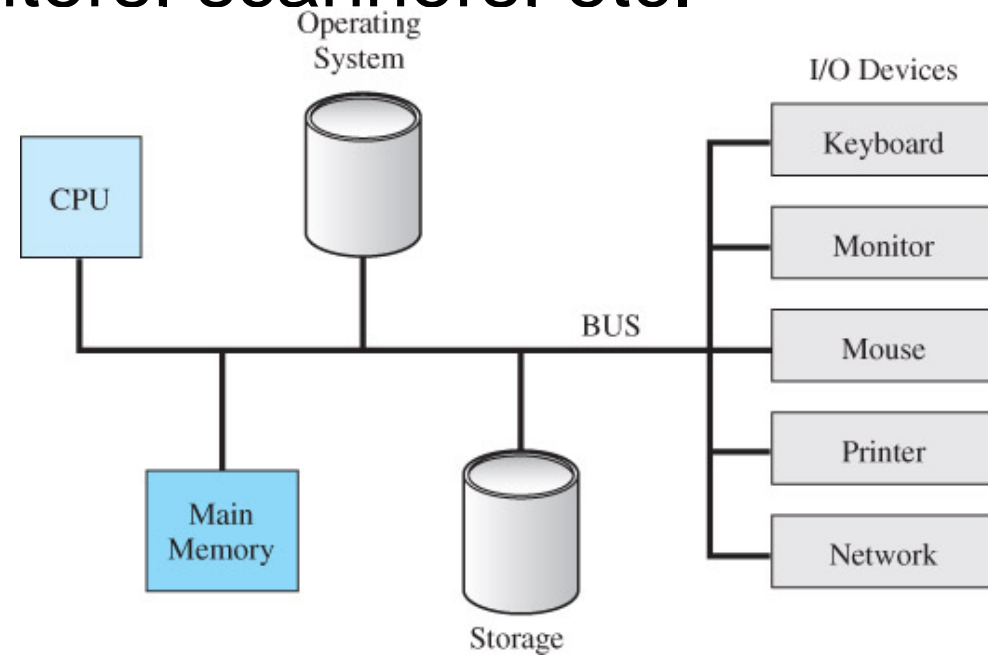
Computing System

- **Secondary Storage:** This term covers a variety of types of devices to store and retrieve data, information, and software programs. Devices range in speed, amount of storage, and cost. These can include hard-drives, floppy drives, ram-drives, CDs, etc.



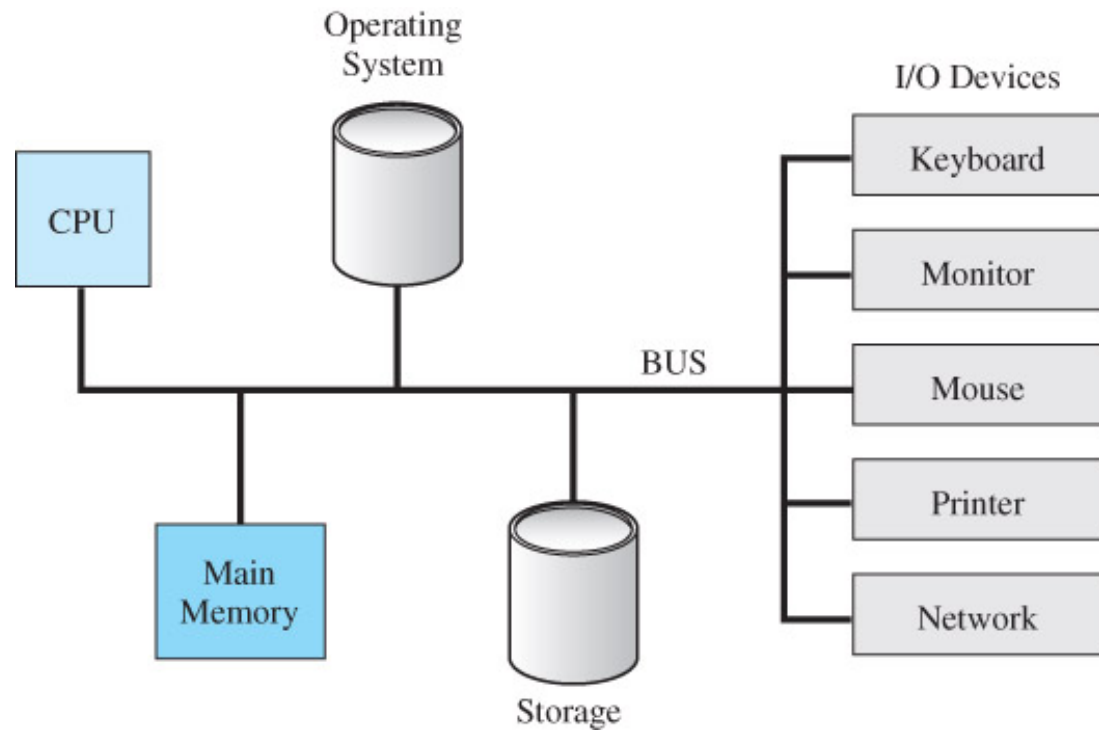
Computing System

- **I/O Devices:** This is a category of devices used to provide input to the machine, or display output for the user or to communicate with other computers. Devices in this category include: monitor, keyboard, mouse, network card, modem, camera, printers. scanners. etc.



Computing System

- **Bus:** an electrical highway that is used to connect the components. The bus is multiple wires, so that many bits can be communicated between devices at the same time. It is typical for a computer system to have two or more buses.



6th Component: Operating System

The operating system includes low level software for controlling the hardware devices, as well as software for managing programs and the resources in the computer system.

- User interface – typically a Graphical User Interface (GUI) though other types of interfaces are possible.
- Programming interface: a way for programmers to utilized portions of the operating system routines in developing software.
- Most of the operating system is software, which resides in secondary storage and is loaded into the computer's memory when the computer boots-up.
- Usually, small portion of system code is stored in hardware in a chip, and may be called a Basic Input-Output System (BIOS), which contains instructions for loading and starting the operating system.

The Central Processing Unit (CPU)

- contains the logic used to execute or process instructions
- a single chip that is the master of all the other devices in the system and any secondary processors.
- The CPU chip itself is quite small, the size of a fingernail or smaller.
- It is enveloped in a plastic or ceramic package
- The ceramic enclosure protects the fragile CPU,
 - connects input and output wires to pins on the chip for easy connection to the rest of the computer system
 - and is involved in transferring waste heat away from the chip.
- Depending on how fast the CPU operates, many chips can generate sufficient heat to cause internal failures unless the excess heat is dealt-with in some way.

Numbers

Inside the CPU everything is stored as numbers

- Represented inside the computer with binary digits.
- The binary number system has only two digits: zero and one.
- This two-digit system turns out to be convenient to build and manufacture using modern digital electronics technologies.
- A binary representation of the number eighteen for instance, looks like this in binary:

10010

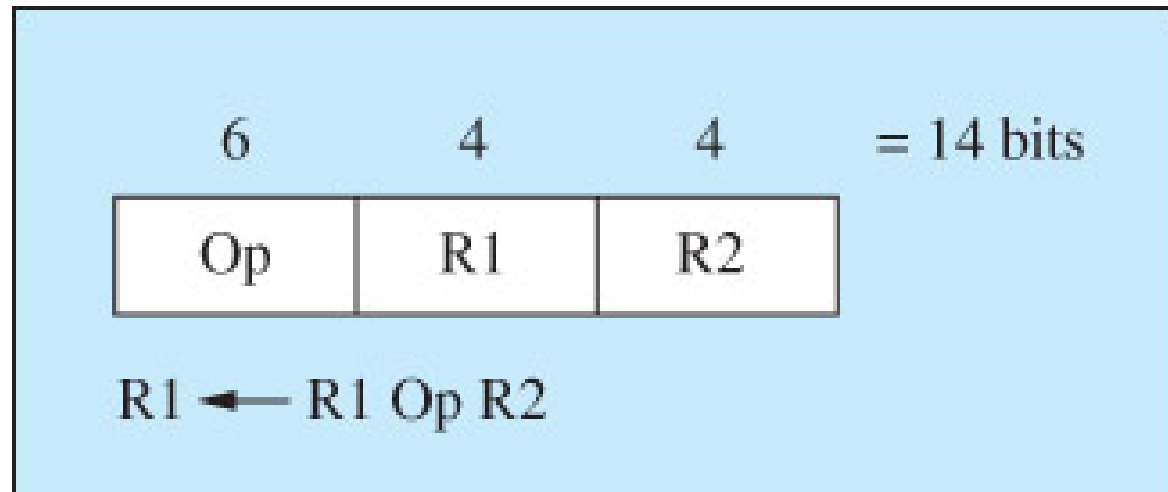


Computer

- All information that we manipulate must be translated at some point by the computer and computer software, into simple binary representations.
- The CPU operates on values represented as binary digits, and in fact, has no “knowledge” about the meaning of the binary numbers or what they represent in our world.
- It simple-mindedly manipulates the data represented by binary numbers as it is told to do so by its programs, which are also translated or converted into binary representations before the computer can work with them.
- The computer itself has no intelligence of its own, and all of its abilities are simply the result of capturing the intelligence and logic of its makers (both hardware designers and software programmers).

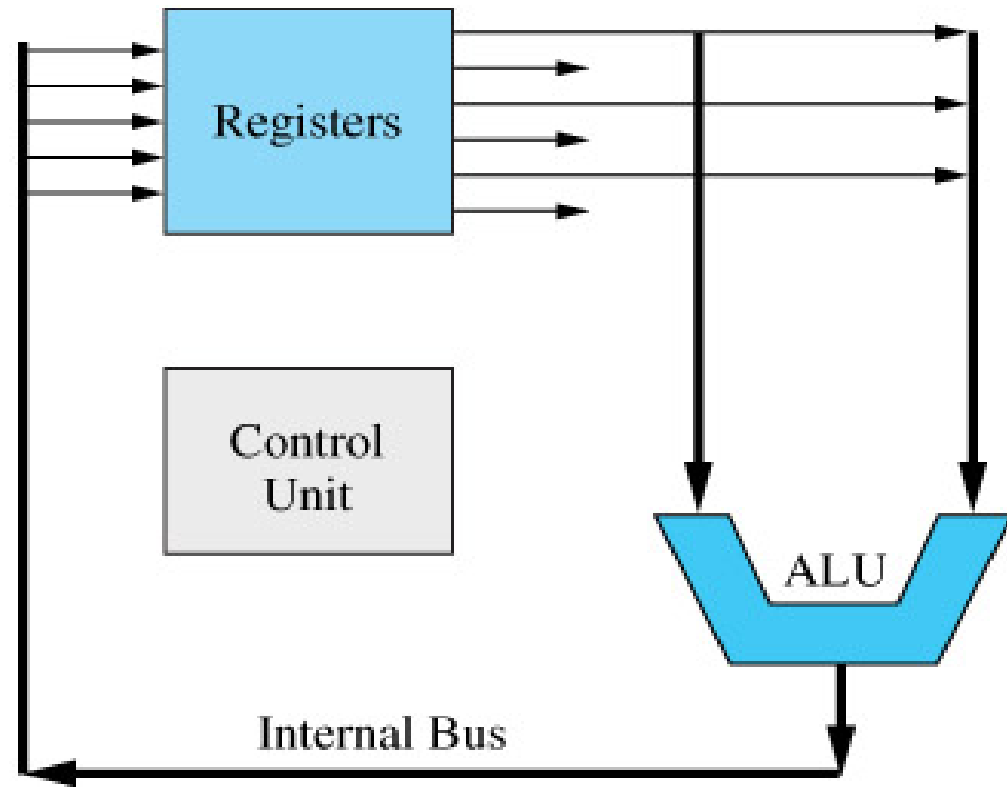
CPU Components

1. **Registers:** a set of temporary storage locations
 - add two numbers together, each number would first be loaded from memory into a register, prior to adding process.
 - Result goes in a register



CPU Components

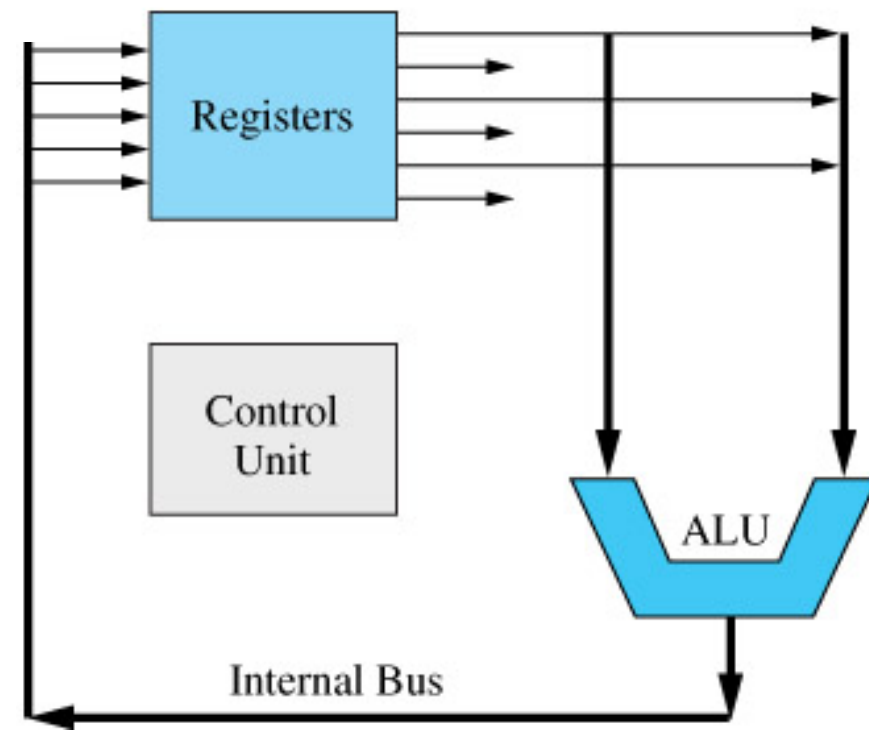
- 1. Registers:** a set of temporary storage locations
 - add two numbers together, each number would first be loaded from memory into a register, prior to adding process.
 - Result goes in a register



CPU Components

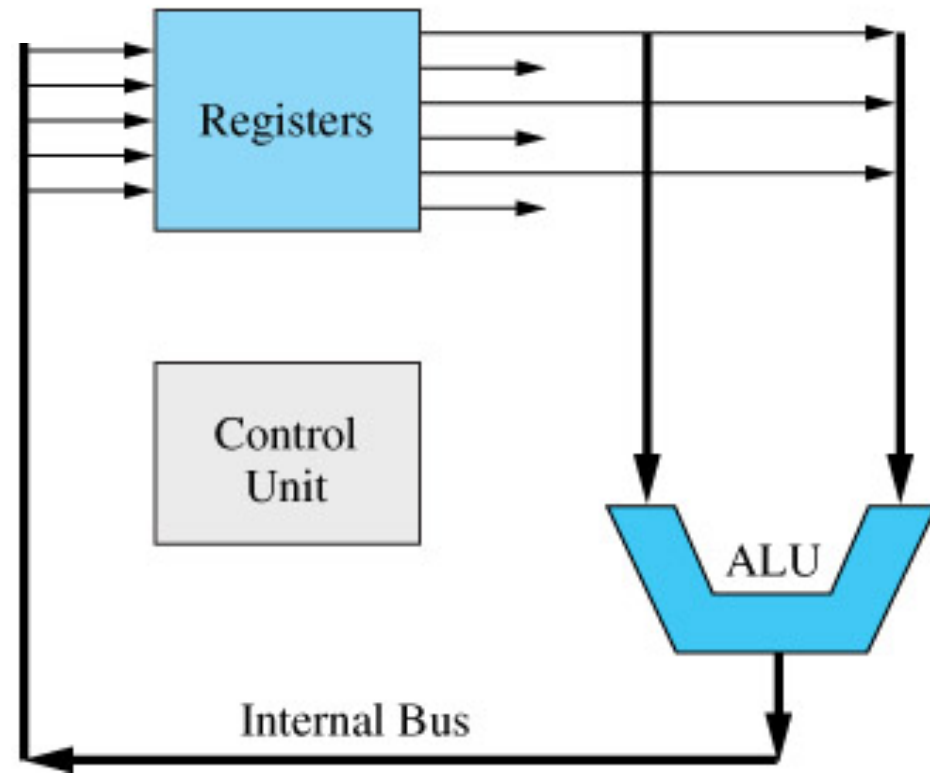
2. Arithmetic Logic Unit (ALU):

- this is the logic that can do operations with binary numbers.
- math functions like addition, subtraction, multiplication and division,
- the ALU can also manipulate numbers in other ways and compare numbers together for equality and inequalities (greater than, less than).



CPU Components

- 3. Control Unit:** logic that is written into the hardware chip that determines the sequence of events
- Things like: how to decode an instruction
 - how to move data from one register to the ALU
 - where to put results from the ALU
 - which instruction should be processed next



Assembly Language

- The following is an instruction that adds the contents of two registers together.

ADR R1 R2

- This representation is an example of what an assembly language instruction looks like.
- Assembly language is a low level programming language – we prefer easier to use languages.

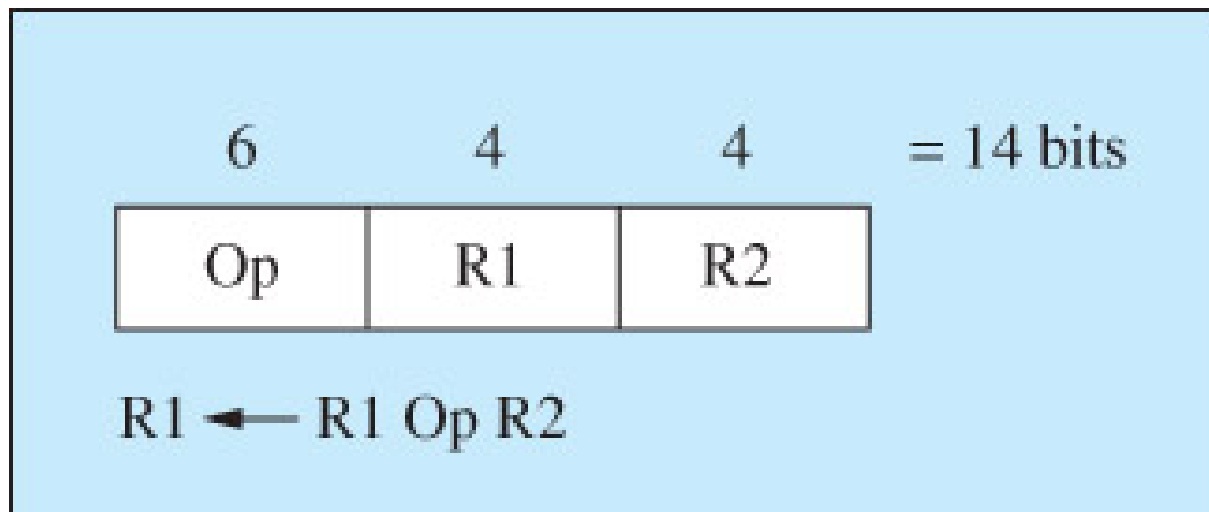
- The assembly language code can be directly translated (by an assembler program) into machine code which the computer can process:

ADR R1 R2

00 010000 0001 0010

- The computer can process the machine code
- hard for humans to work with
- Hence, the creation of assembly language and other higher level programming languages.

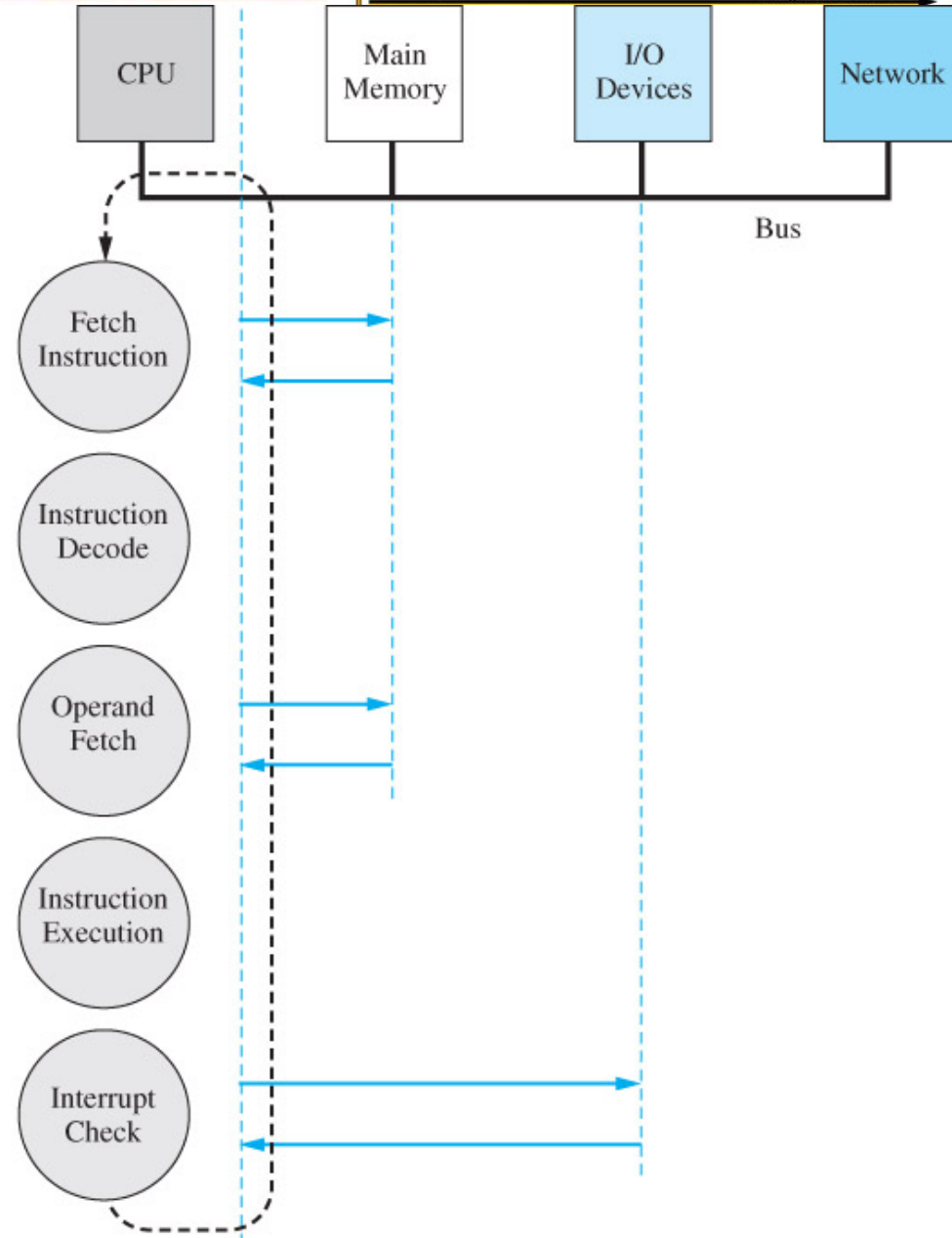
- Instruction Format: machine code and assembly language is organized in an instruction format
- Operation to be performed (OpCode or Op)
- Operands that the instruction will operate on (R1 and R2 are registers).



Fetch Decode Execute Cycle

- CPU repeats the cycle for each instruction
 1. Fetch the Instruction
 2. Decode the instruction (opcode and operands)
 3. Fetch and needed operands
 4. Execute the instruction
 5. Check for Interrupt

Processing Cycle



End of Lecture

**End
Of
Today's
Lecture.**

This slide intentionally left blank

