CMSC 132: Computer Architecture

Asst. Prof. Reginald Neil C. Recario
rncrecario@gmail.com
Institute of Computer Science
University of the Philippines Los Baños

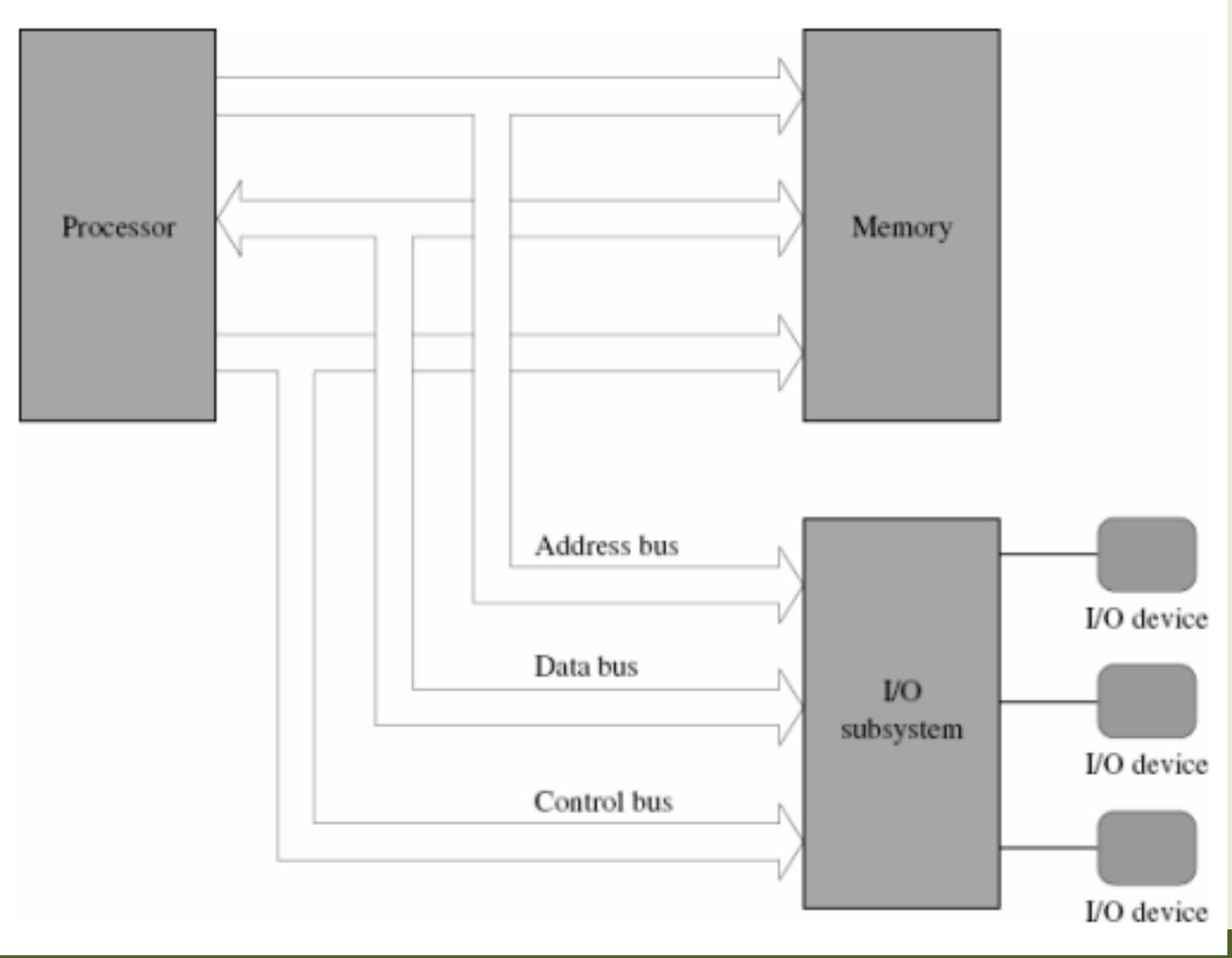
The Von Neumann Computer Model

- This slide set is from http://www.c-jump.com/CIS77/CPU/VonNeumann/lecture.html with some minor changes done.

The Von Neumann Computer Model

- John von Neumann described the model in 1945
- Three main blocks:
  - Central Processing Unit (CPU)
  - Memory
  - I/O
- Blocks are connected by a system bus
- Registers are the most prominent items of the CPU

The Von Neumann Computer Model



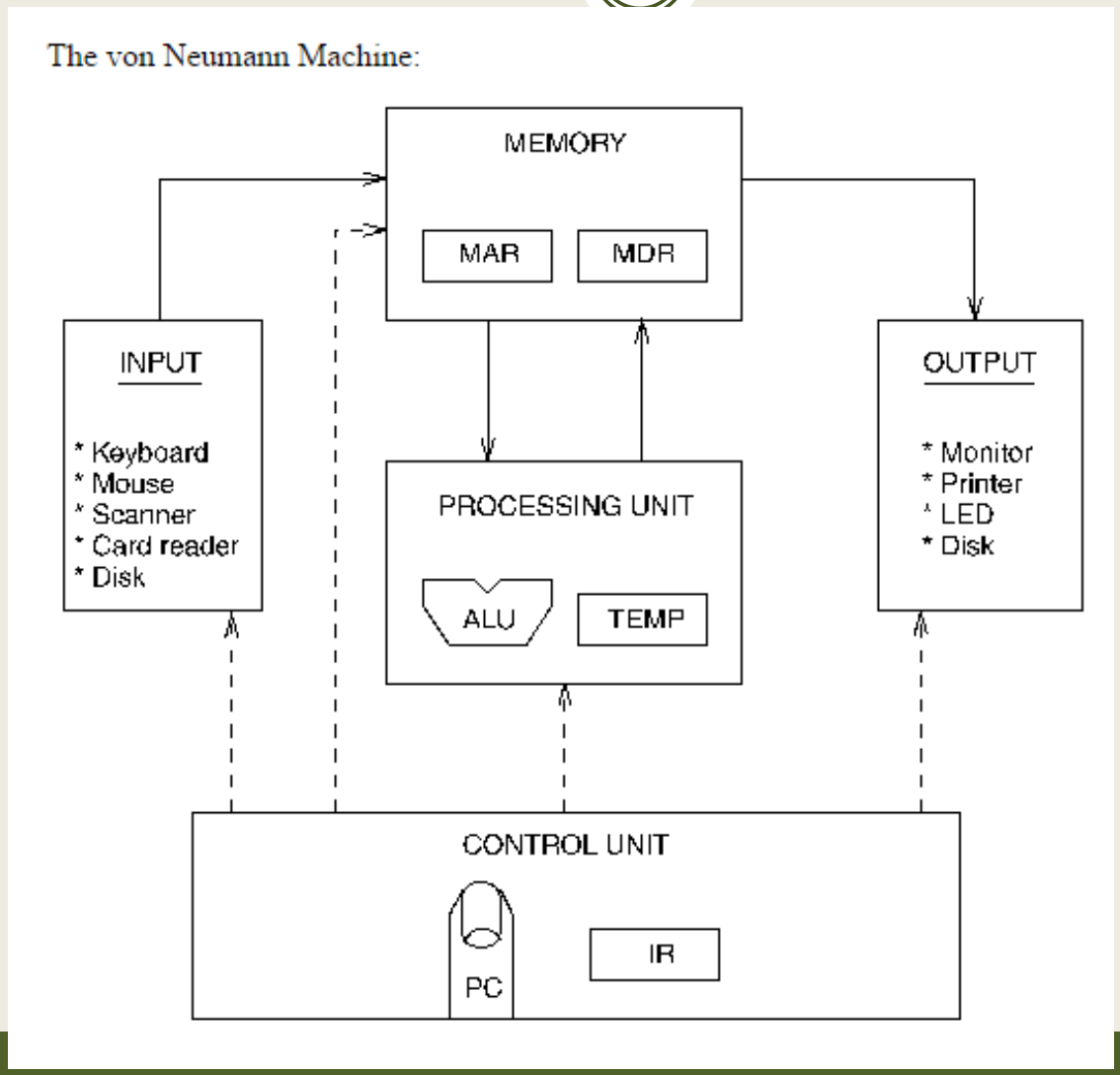
The Von Neumann Computer Model

- Types of Buses
  - Address Bus – bus specifying physical memory address
  - Data Bus – bus for transferring data
  - Control Bus – bus used in device communication

The Von Neumann Computer Model

- Components of the VNCM:
- Memory – storage
- Processing Unit – information processing
- Input – information from an interface going to the computer
- Output – information from the computer to an interface
- Control Unit – regulates task performance making sure that instructions are executed properly

The Von Neumann Computer Model



The Von Neumann Computer Model

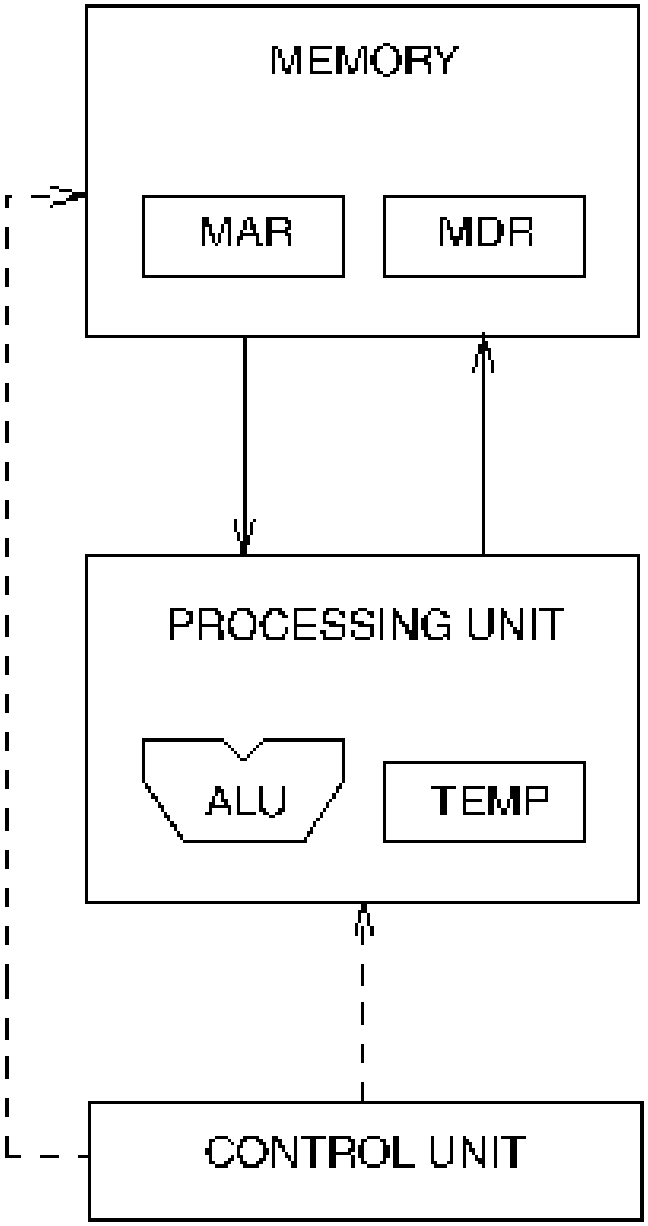
- Memory:
- Memory Address Register
  - Register storing memory address pointing to where data will be fetched or stored
- Memory Data Register
  - Register containing the data to fetched or stored from the computer storage

The Von Neumann Computer Model

- Memory Communication (Reading):
  - The address of the location is put in MAR.
  - The memory is enabled for a read.
  - The value is put in MDR by the memory.

The Von Neumann Computer Model

- Memory Communication (Writing):
  - The address of the location is put in MAR.
  - The write enable signal is asserted.
  - The value in MDR is written to the location specified.


MEMORY
MAR
MDR
PROCESSING UNIT
ALU
TEMP
CONTROL UNIT

The Von Neumann Computer Model

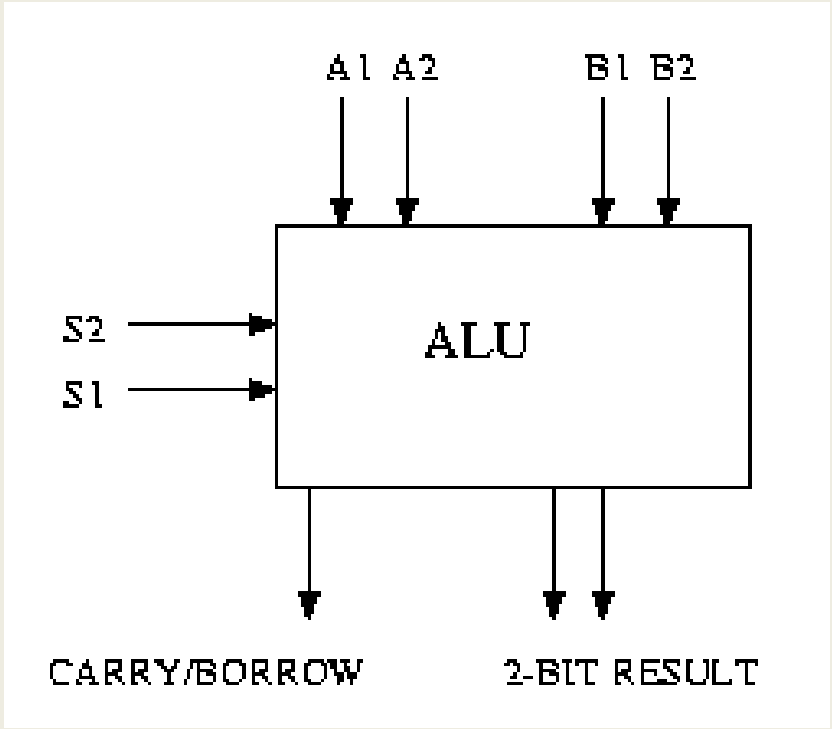
- ALU – Arithmetic Logic Unit
  - Component that performs the arithmetic and logical operations
  - Input: Integer Operand(s), Status, Opcode
  - Output: Status, Result

The Von Neumann Computer Model

- ALU
- Status:
  - Carry Out – result of excess in additional operation, borrowing in subtraction or overflow
  - Overflow – exceeding allowed bus range
  - Parity – indicator of whether the number of bits is odd or even (even parity, odd parity)
  - Negative – indicating the operation result is negative
  - Zero – indicating a zero value input

The Von Neumann Computer Model

- ALU Operations:
  - Arithmetic Operations
  - Bitwise Logical Operations
  - Shift Operations

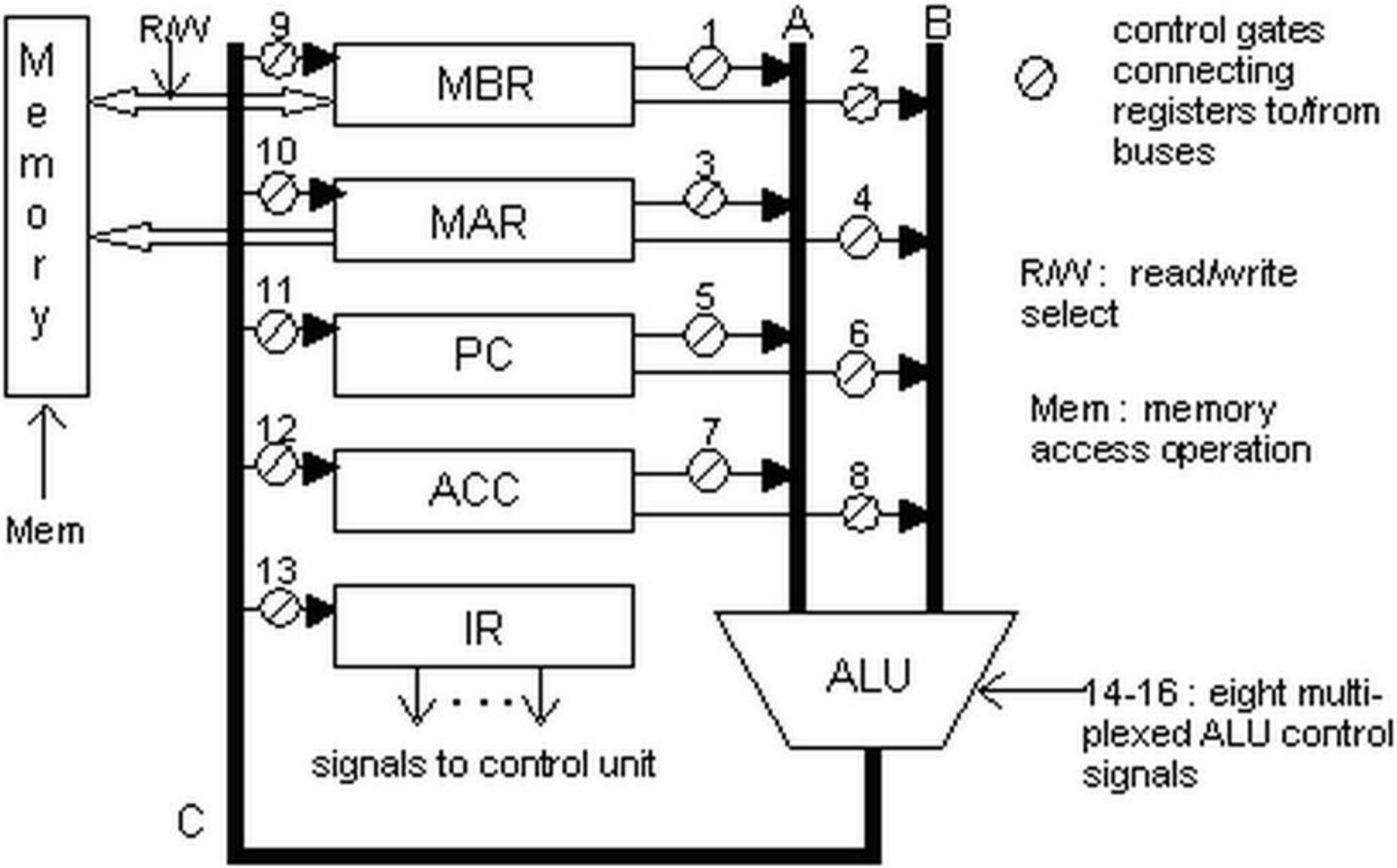


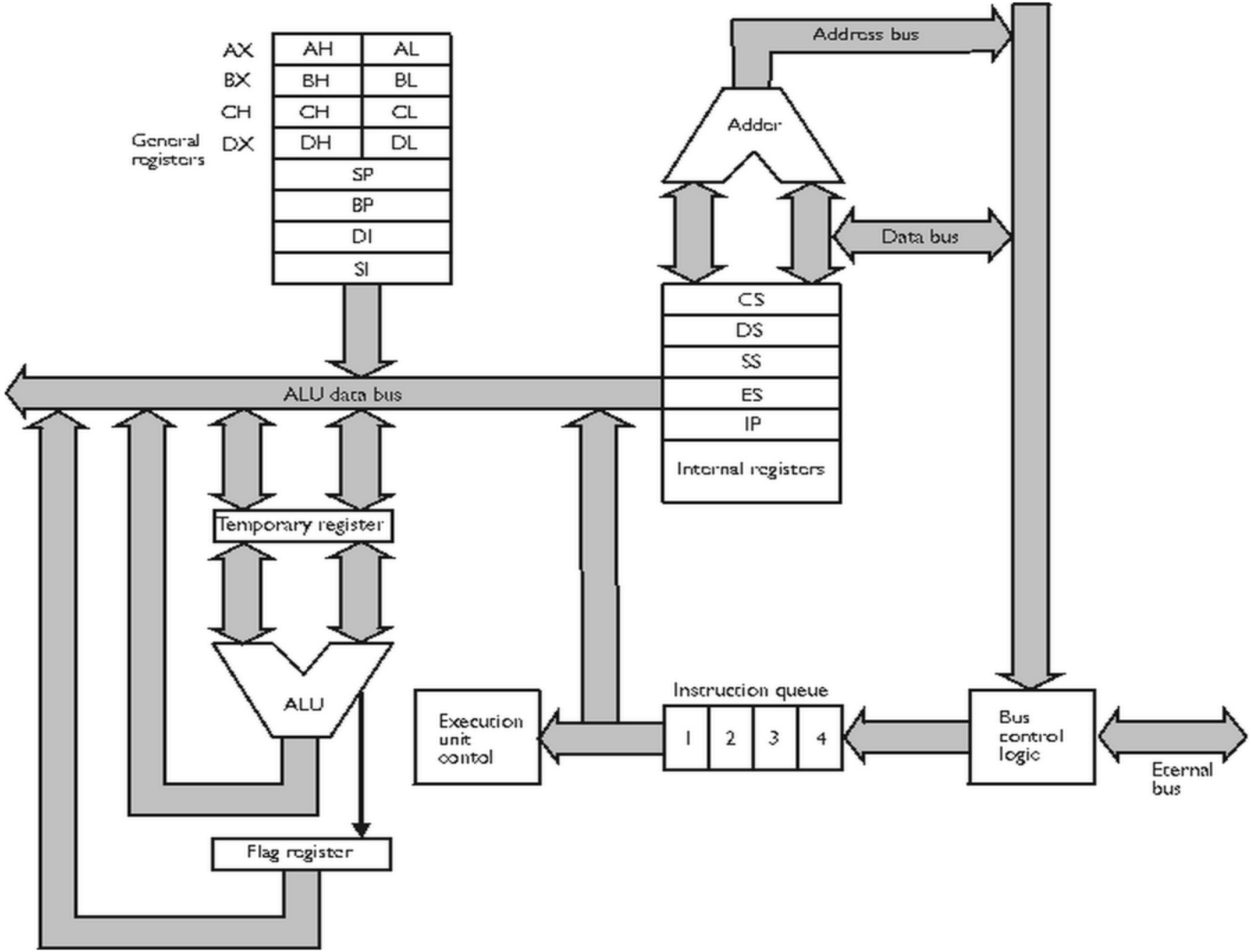
http://3.bp.blogspot.com/-rXBoGqB19-U/UzUENbd6ndI/AAAAAAAAAiQ/4fIAF4VZE6M/s1600/alu_sch.gif

The Von Neumann Computer Model

- Data-path
  - Linking of hardware units such as ALU, registers and memory
  - Bit flow is controlled by gates (on/off) through microinstructions

A Simplifed Datapath




AX
BX
CH
DX
General registers
AH
AL
BH
BL
CH
CL
DH
DL
SP
BP
DI
SI
Address bus
Adder
Data bus
CS
DS
SS
ES
IP
Internal registers
ALU data bus
Temporary register
ALU
Flag register
Execution unit control
Instruction queue
1
2
3
4
Bus control logic
External bus

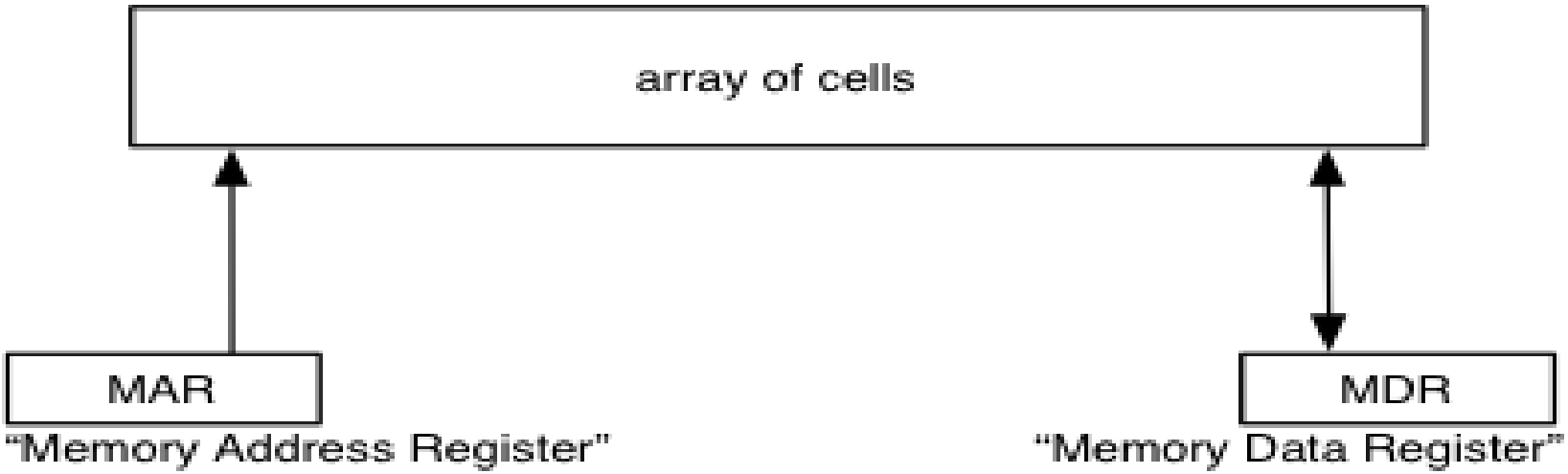
The Von Neumann Computer Model

- Memory Operations
  - There are two key operations
    - Fetch (address)
      - Returns the value
      - Value at the given address left unchanged
    - Store(address, value)
      - Writes a (new) value on the specified address

The Von Neumann Computer Model

- Memory
  - Such cases of random-access is called RAM
  - Those that non-volatile or read-only is called ROM

Memory Operations:



fetch (addr):

1. Put *addr* into MAR
2. Tell memory unit to "load"
3. Memory copies data into MDR

store (addr, new-value):

1. Put *addr* into MAR
2. Put *new-value* into MDR
3. Tell memory unit to "store"
4. Memory stores data from MDR into memory cell.

The Von Neumann Computer Model

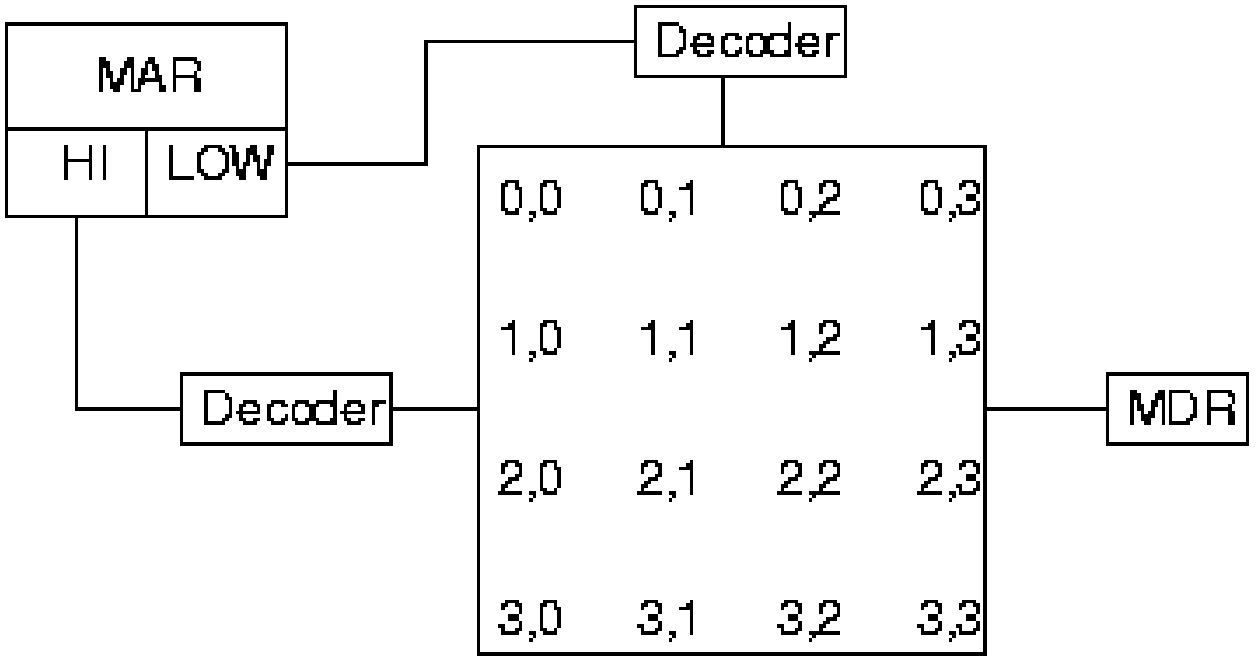
- MAR and MDR
  - Memory Address Register
  - Memory Data Register

The Von Neumann Computer Model

- MAR is connected to the address bus.
- MAR is "the only way" for the CPU to communicate with address bus.
- Tri-state buffer between MAR and the address bus prevents MAR from continuously dumping its output to the address bus.
- MAR can hold either an instruction address or a data address.

The Von Neumann Computer Model

Memory Circuitry:



The Von Neumann Computer Model

- **MDR** Stands for *memory data register*.
- **MDR** is connected to the data bus.
- Data can go in both directions: to and from memory,
- Therefore, **MDR** can load its data from
  - the data bus (for reading data)
  - one of the CPU registers (for storing data.)
- A **2-1 MUX** circuit selects input from one of the two.

The Von Neumann Computer Model

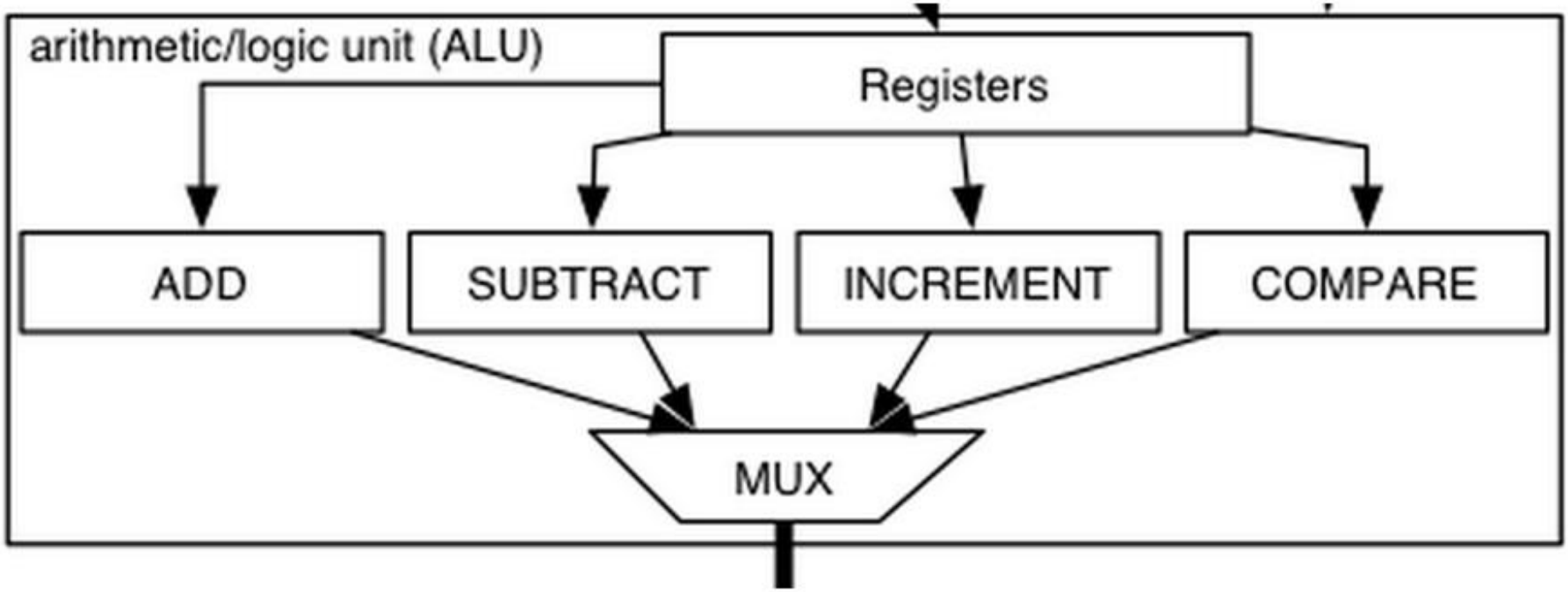
- **ALU**
- Processing unit is hardware that implements Arithmetic and Logical Operations
- **ALU** stands for *Arithmetic and Logic Unit*, capable of performing
- ADD, SUBTRACT, AND, OR, and NOT operations.

The Von Neumann Computer Model

- **ALU**
- The size of input quantities of ALU is often referred to as *word length* of the computer.
- Many processors today have word length of 32 and 64 bit.
- Processing unit also includes a set of **registers** for *temporary storage* of data and memory addressing.

The Von Neumann Computer Model

- **ALU**



The Von Neumann Computer Model

- **ALU and the Word Length**
- The size of quantities processed by ALU is the *word length* of the computer.
- The word length does not affect what a computer can compute.

The Von Neumann Computer Model

- **ALU and the Word Length**
- A computer with a smaller word length can do *the same* computation as one with a larger word length…
  - …but it will take more time.
- For example, to add two 64 bit numbers,
  - word length = 16 takes 4 adds.
  - word length = 32 takes 2 adds.
  - word length = 64 takes 1 add.

The Von Neumann Computer Model

- **CU (Control Unit)**
- Manages the Processing Unit.
- Implemented as FSM.
- FSM directs all activity.
- Clock-based step-by-step precessing, cycle-by-cycle.
- FSM is controlled by the
  - Clock signal
  - Instruction Register
  - Reset signal

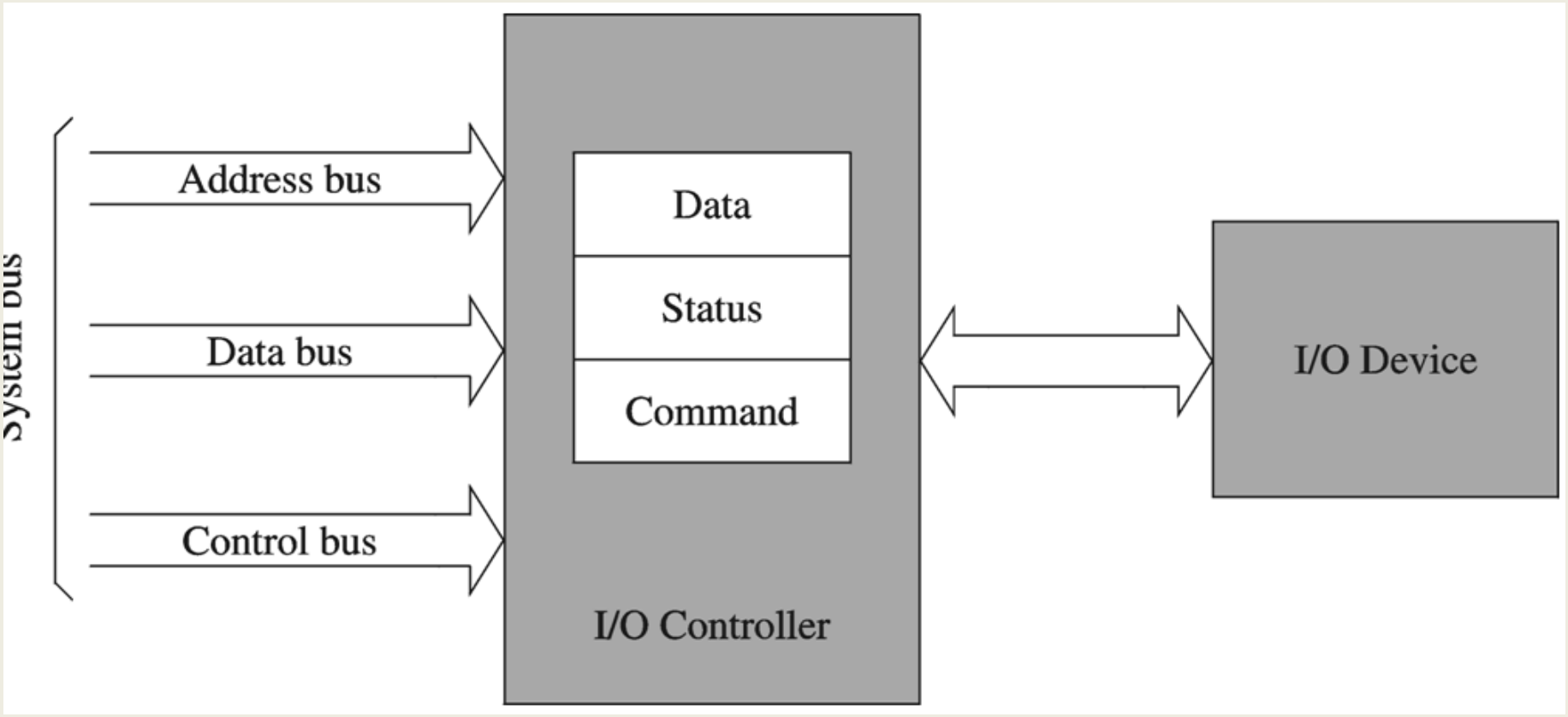
The Von Neumann Computer Model

- Control unit includes
  - Instruction Register IR
  - Instruction Pointier IP (aka Program Counter PC)
- FSM *outputs* of the control unit,
- shown by dash lines, have two purposes:
  - Control processing that takes place inside the ALU.
  - Authorize read/write gate control of the CPU data-path.

The Von Neumann Computer Model

- **Input/Output**
- I/O controller provides the necessary interface to I/O devices.
- Takes care of low-level, device-dependent details.
- Provides necessary electrical signal interface.

The Von Neumann Computer Model



The Von Neumann Computer Model

- **Input/Output Ports**
- Processor and I/O interface points for exchanging data are called I/O ports.

The Von Neumann Computer Model

- **Input/Output Ports**
- Two ways of *mapping* I/O ports:
  - **Memory-mapped I/O**:
    - I/O ports are mapped to the memory address space.
    - Reading/writing I/O is similar to reading/writing memory.
    - Can use memory read/write instructions.
    - Motorola 68000 uses memory-mapped I/O.

The Von Neumann Computer Model

- **Input/Output Ports**
- Two ways of *mapping* I/O ports:
  - **Isolated I/O**:
    - Separate *I/O address space*.
    - Intel 80x86 processors support isolated I/O.
    - Requires special I/O instructions, like IN and OUT on x86.

The Von Neumann Computer Model

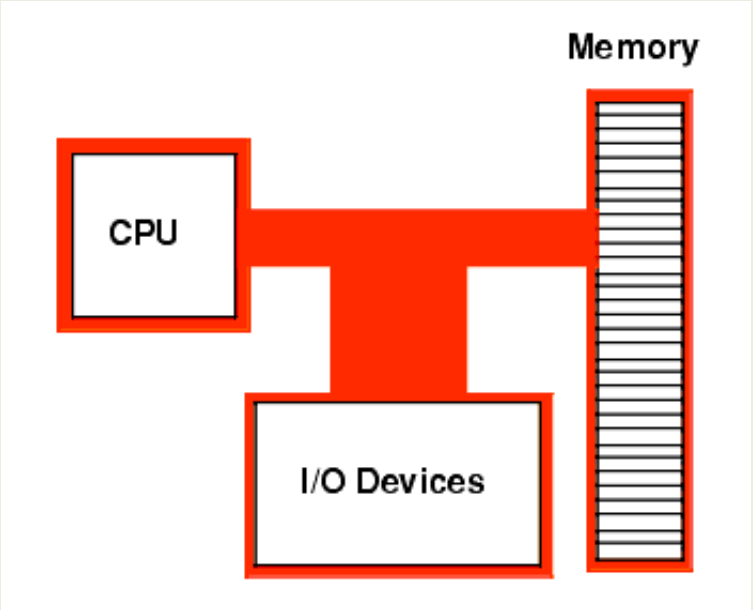
- **Input/Output Address Space**
- Pentium x86 provides 64 KB of I/O address space.
- Can be used for 8-, 16-, and 32-bit I/O ports.
- Combination cannot exceed the total I/O address space, can have either
  - 64 K 8-bit ports
  - 32 K 16-bit ports
  - 16 K 32-bit ports
  - A combination of these for a total of 64 KB.

The Von Neumann Computer Model

- **Input/Output Address Space**
- I/O instructions do not go through memory *segmentation* or *paging*.
- I/O address refers to the physical I/O address.

The Von Neumann Computer Model

- **Console Input/Output in Protected Memory Mode**
- Using console in Protected Memory Mode:
  - Keyboard input
  - Console display output
  - Ascii character encoding



The Von Neumann Computer Model

- **Instruction Processing**
- Central idea of von Neumann model is that both program and data stored in computer memory:
- Program is a sequence of instructions
- Instruction is a binary encoding of operations and operands

The Von Neumann Computer Model

- **Instruction Processing**
- :For example, an arithmetic expression
  - **-a + b * c**
- could be computed by a program with three machine instructions

The Von Neumann Computer Model

- **Instruction Processing**

```
neg ax ; negate (multiply by -1)
imul bx, cx ; multiply and store result
  in bx
add ax, bx ; add and store result in ax
```

where NEG, IMUL, ADD are arithemtic instructions, **AX**, **BX**, and **CX** are operands.

The Von Neumann Computer Model

- **Instruction Processing**
- Instructions are most basic units of processing.
- Instructions are executed under control of the control unit.

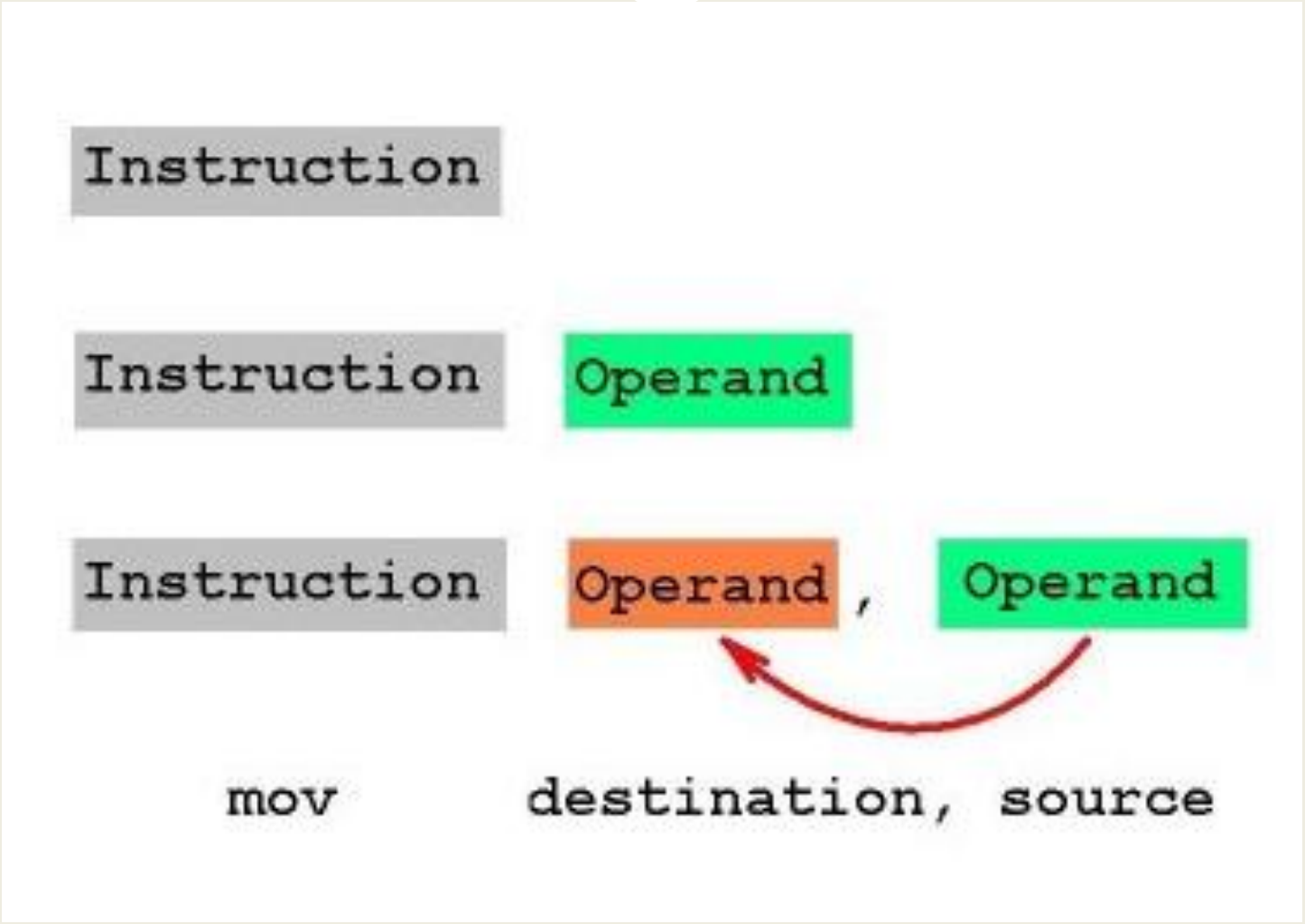
The Von Neumann Computer Model

- **Instruction Components**
- Instruction in memory has two parts: opcode and operands.
- The opcode identifies the operation that instruction does

The Von Neumann Computer Model

- **Instruction Components**
- The operands are subjects of the operation, such as data values, registers, or memory addresses.
- Due to variety of opcodes and operands, instructions may occupy different sizes of bytes in memory.
- Instructions with no operands can have *implied operands*, those that are not explicitly shown.

The Von Neumann Computer Model



The Von Neumann Computer Model

- **Instruction Components**
- The operands are subjects of the operation, such as data values, registers, or memory addresses.
- Due to variety of opcodes and operands, instructions may occupy different sizes of bytes in memory.
- Instructions with no operands can have *implied operands*, those that are not explicitly shown.

Reference(s):

- _____. http://www.c-jump.com/CIS77/CPU/VonNeumann/lecture.html
- **Hennessy, J.L., Patterson, D.A**. Computer Architecture: A Quantitative Approach (4th Ed)