Introductory Computing

Computer Fundamentals

Computer Fundamentals:

Sl. No	Topics	Hrs
1	History, Generations, Classification of Computers;	1
2	Organization of a Computer; (functional block diagram, ALU, CU, Memory, I/O), Instruction Execution Cycle.	3
3	Concept of Programming and Programming Languages; (M/C language, Assembly Language, HLL), OS, Software tools	3

What is a Computer?

- Earlier: A Calculating Device that can perform arithmetic operations at enormous speed.

  It was the main objective for inventing the computer. But more than 80% of the work done by computers today is of non-numeric/mathematical in nature.
- More Accurately: It may be defined as an Electronic Device that operates on input data according to a program stored in its memory to produce result(s).
- A Computer is capable of performing almost any task provided that the task can be reduced to a series of logical steps. (More on this later, when we will discuss the term Algorithm)

Characteristics of Computer

- Automatic
- Speed (most of the cases in MIPs, MFLOPs)
- Accuracy (Very accurate provided the program is written correctly, and Input are according to specification) – Garbage-In-Garbage-Out
- Diligence (Free from monotony, tiredness, lack of concentration)
- Versatility
- Power of Remembering
- No Feelings
- No IQ

Evolution of Computer

- Abacus (simple addition & Subtraction) , around 600 B.C.
- First Mechanical Adding Machine by Blaise Pascal in 1642.
- Later, Calculator for Multiplication by Baron Gottfried in 1671
- In 1822 Charles Babbage's Difference Engine, later Analytical Engine (completely automatic, basic arithmetic operations at speed 60 OPs) in 1842.
- In 1952, IBM introduced 700 series machine. In 1953 IBM produced IBM-650 and sold over 1000 no. of computers

Computer Generations

- To distinguish varying Hardware and Software Technologies.
- First Generation (1942-1955): Vacuum Tubes; EM relay memory; punched cards secondary mem; M/C and Assembly languages; stored program concept; Bulky in size; Unreliable; difficult to use; Scientific applns (e.g., ENIAC, UNIVAC 1, IBM 701)
- Second Generation (1955-1964): Transistors; Magnetic Disk/Tapes/cores memory;Batch OS; HLL; Scientific/Commercial applns; faster,smaller, more reliable and costly (e.g., Honeywell 400, IBM 7030 etc)

Computer Generations (contd)

- Third Generation (1964-1975): ICs (SSI and MSI); larger capacity in storage; Minicomputers; Timesharing OS; Standardization of HLL; Faster, Reliable, easier and cheaper to produce; Scientific, Commercial, Interactive on-line applications (e.g., IBM 360/370, PDP-11, CDC 6600 etc.)
- Fourth Generation (1975-1989): ICs (VLSI technology); Microprocessors; Semiconductor Memory; larger capacity disk (Hard, floppy); PC, Super Computer based on Vector/Symmetric multiprocessing; Computer networks; Multiprocessing OS; GUI; Concurrent Language, OOPD, n/w based applns; Small, reliable PCs, Mainframe; rapid s/w development (e.g., IBM PC, Apple II, VAX 9000, CRAY -1,2 etc.)

Computer Generations (contd)

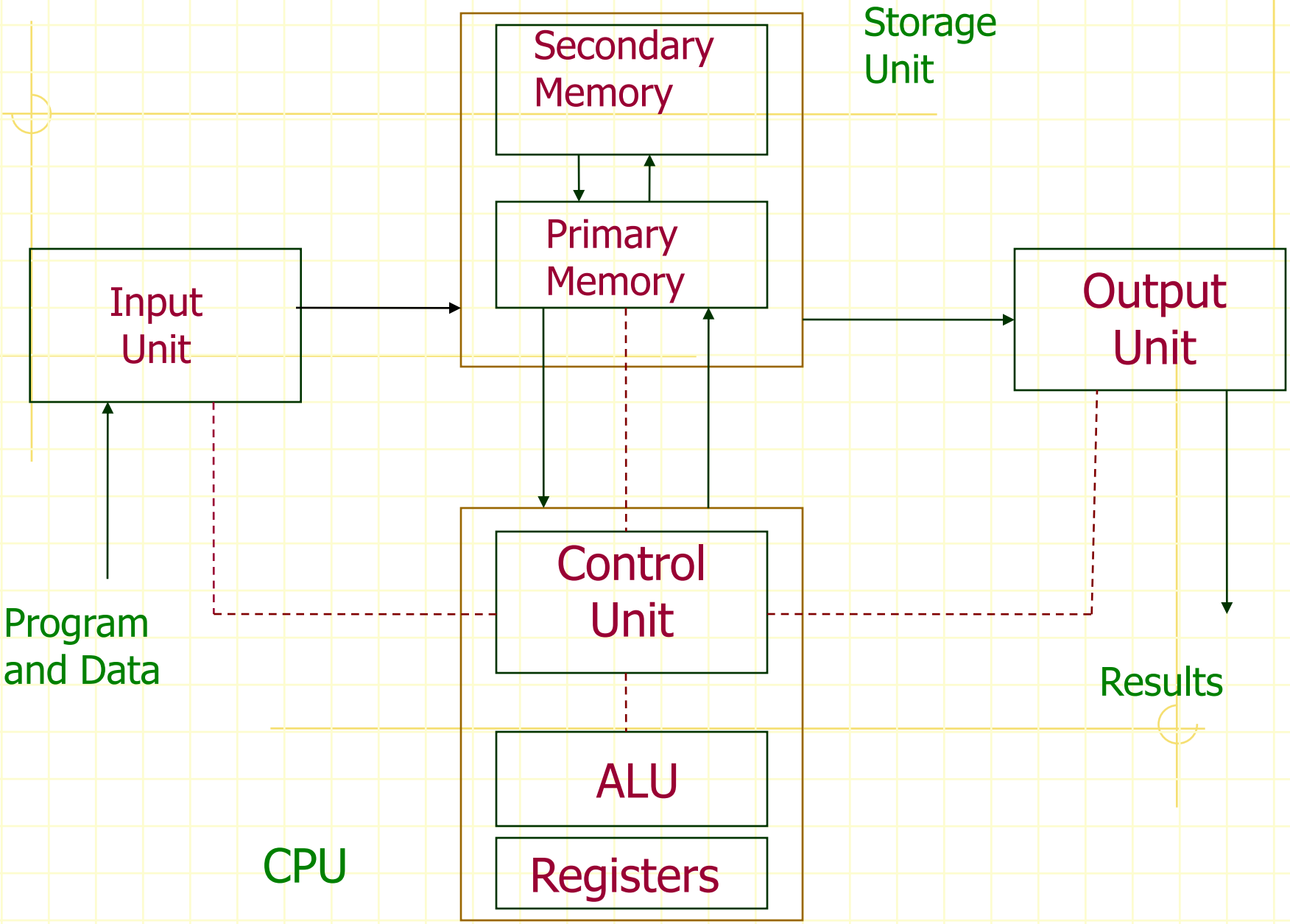
- Fifth Generation (1989-present): ICs (ULSI); larger capacity memory; RAID; Optical Disks; Notebook Computers; Powerful Desktop/Workstations; Internet; Cluster computing; Microkernal-based OS; Multithreading; Parallel programming library MPI, PVM; JAVA, WWW, Internet-based applications; AI; Portable/powerful/cheaper/reliable desktop computers; mobile computing; general purpose m/cs; easy to upgrade; rapid s/w development (e.g. IBM Notebook, P-IV, SUN Workstations, SGI, PARAM 10000)

Classification of Computers

- Traditionally: Computers were classified by their size, processing speed and cost (e.g. microcomputer, minicomputer, mainframe computer and supercomputer).

  With rapidly changing technology, this classification is no more relevant.
- Nowadays: Computers are classified based on their mode of use. These are – Notebook, personal computer, workstations, mainframe systems, supercomputers, and client and servers.

Functional Block Diagram



The CPU & Von-Neumann

- What's in the box? – reminder
- CPU Overview
  - Control Unit
  - ALU
  - Registers
- Catching a Bus
  - Tying it all together
- Mr Von Neumann
  - His legacy

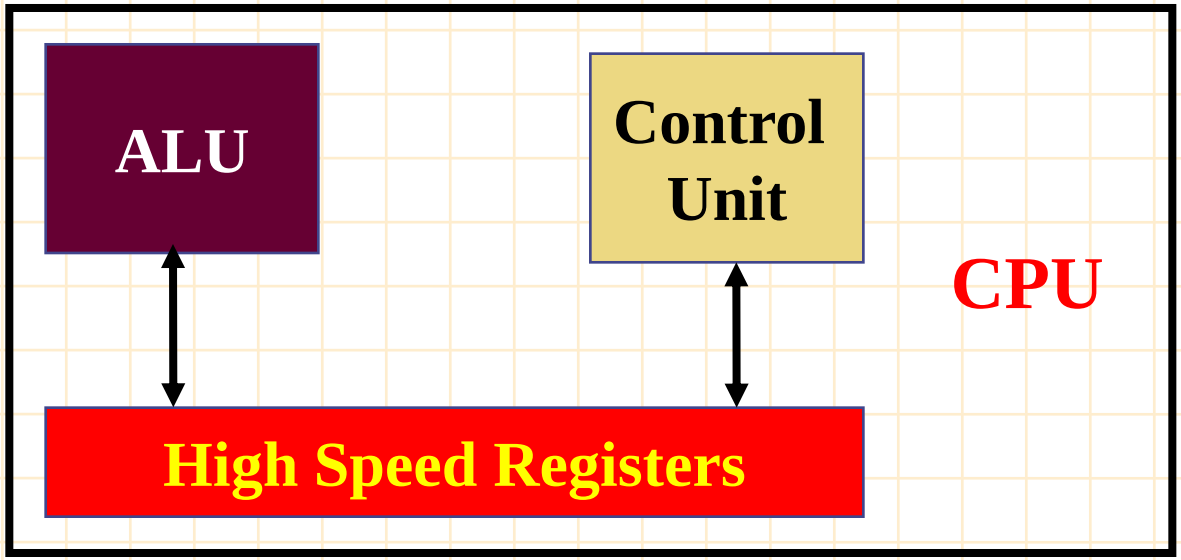
Computer Architecture

- CPU
  - Central processing Unit
  - Speed measure in clock cycles
    - Hertz (Hz) – usually MHz or GHz
    - How quickly the CPU can execute instructions
- CPU often measured in 'bits'
  - 32-bit Processor / 64-bit processor
  - Confusion!!
    - Is this CPU memory word length?
    - Is this data bus width?

usually the answer

Central Processing Unit (CPU)

- CPU has three important parts:
  - ALU (Arithmetic and Logic Unit)
  - Control Unit
  - Registers



Central Processing Unit (CPU)

- Arithmetic & Logic Unit
  - Handles mathematical and logical functions (numerical)
  - Deals with non-numerical logic operations
- Control Unit
  - Handles all low-level hardware operations
    - Input & Output Devices and CPU
  - Carries out instruction handling
    - Fetch Execute Cycle

CPU Registers

- Storage areas within the CPU
  - Used to temporarily store data read from memory
- Accessible at High Speed
- Anything for processing must be kept in a register
- Can also hold the address of a memory location
- Registers are used to process instructions and data during the Fetch Execute Cycle
  - Two main types of register:
    - Instruction Register (IR)
      - Commands to be performed
    - Data Registers
      - Data upon which operations will be performed

CPU Registers

- Common registers in the CPU:
  - Program Counter (PC)
    - Holds the memory address of the next instruction to be executed
  - Memory Buffer Register (MBR)
    - Briefly holds data and instructions that travel to and from memory
    - Sometimes called MDR (Memory Data Register)

CPU Registers

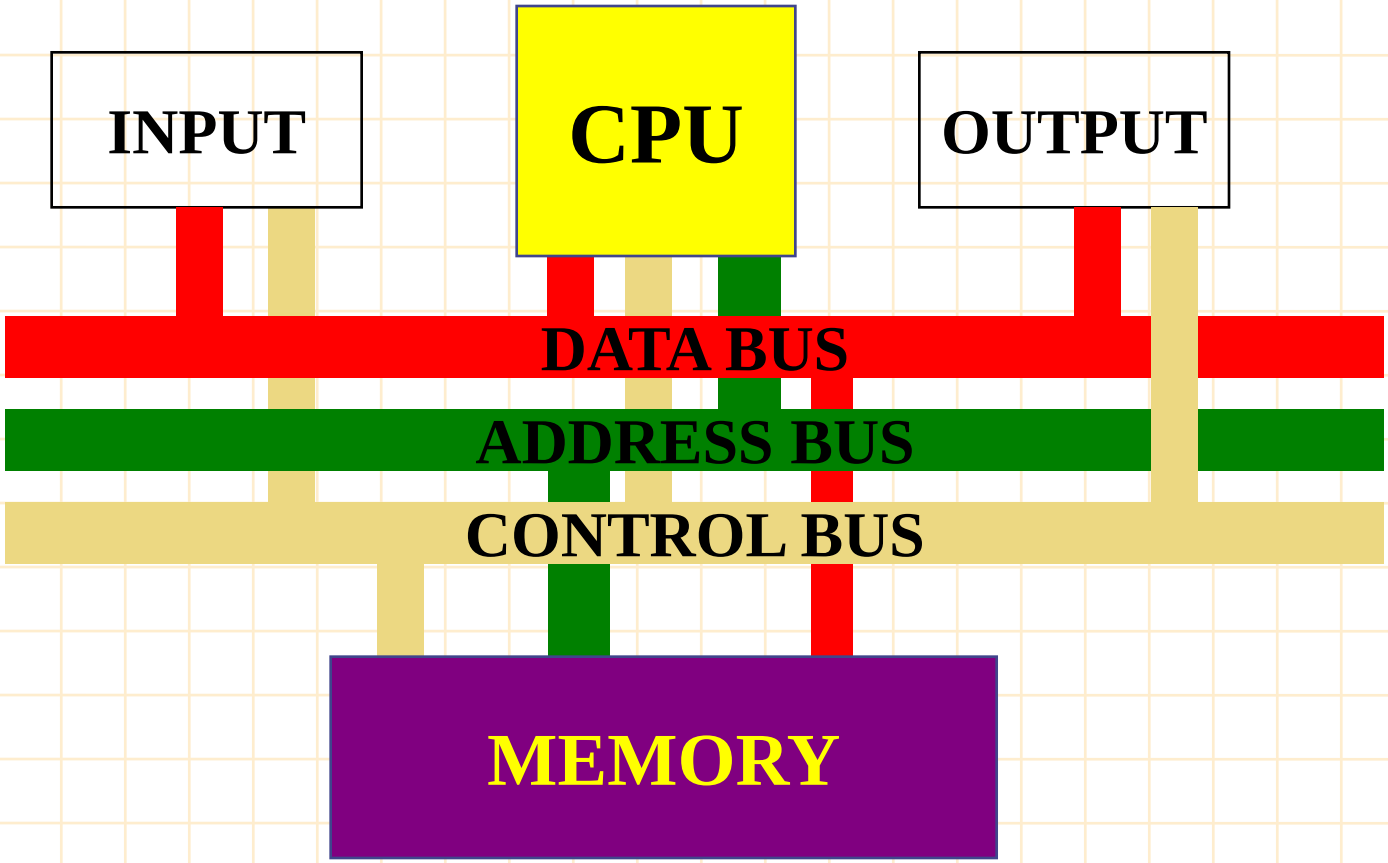
- Common registers in the CPU:
  - Memory Address Register (MAR)
    - Holds the memory address locations of data and/or instructions to be read / written to memory
  - Current Instruction Register (CIR)
    - Holds the instruction which is to be executed
  - General Purpose Registers
    - 'Working areas' for data processed by the ALU

Buses

- A collection of wires which connects together the internal components of the computer
  - Allows transfer of data
- Main types of bus:
  - Data bus
    - Carries actual data bits (information)
  - Address bus
    - Transfers locations where data should be sent
  - Control bus
    - Carries status information

Buses

- How buses fit into the computer system:

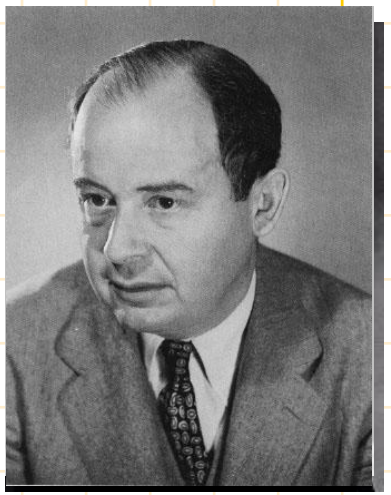


CPU Buses

- Parallel connections between low-level components of the computer
  - Size is measured by the number of parallel connections on the bus
    - E.g.- 32-bit wide bus = 32 individual wires
    - These bursts are called *words*
    - A word is a set measure of bits (in this case we have a 32-bit word)
- Signals on buses follow strict timing sequences
- Some buses are bi-directional
  - Allowing two way flow of information

Computer Architecture

- The Von Neumann Model
  - 1903 - 1957
  - Mathematician
  - Quantum physicist
  - Worked on ENIAC
    - **Electronic Numerical Integrator and Computer**
    - **Major development in computer technology**
  - Responsible for developing the Fetch Execute Cycle, and his namesake -
    - 'Von Neumann Model'

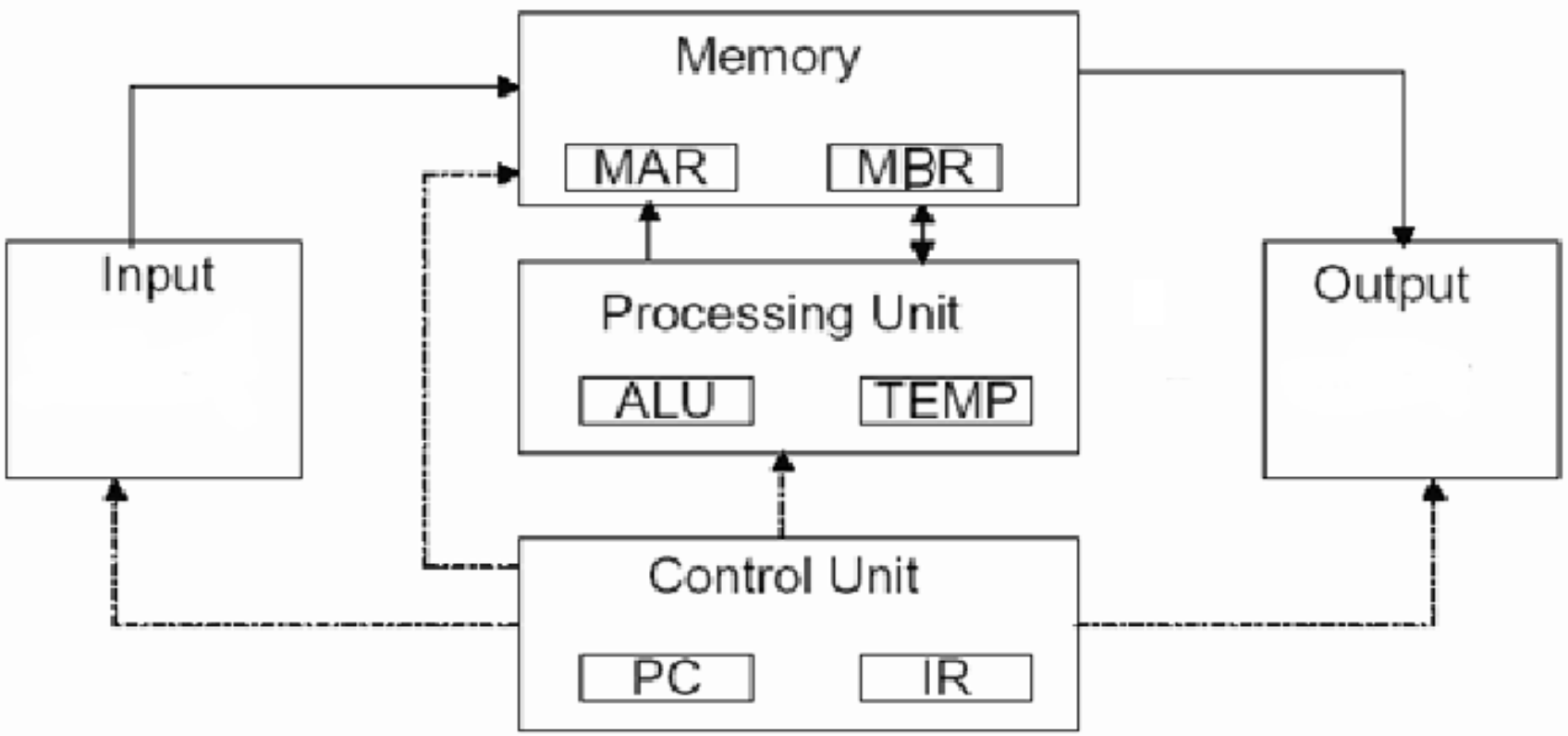


Von Neumann Model

- Logically defines a complete computer system
- Centralised control of all processes of the computer system
- Defines main parts of the machine:
  - Memory
    - Storage for instructions and data
  - Processing unit
    - ALU functions
  - Control unit
    - Interpreting instructions
    - Issuing commands
  - Input and Output
    - For entering and retrieving data

Von Neumann Model

- Logical Structure of the computer system
  - Routes of data transfer during processing

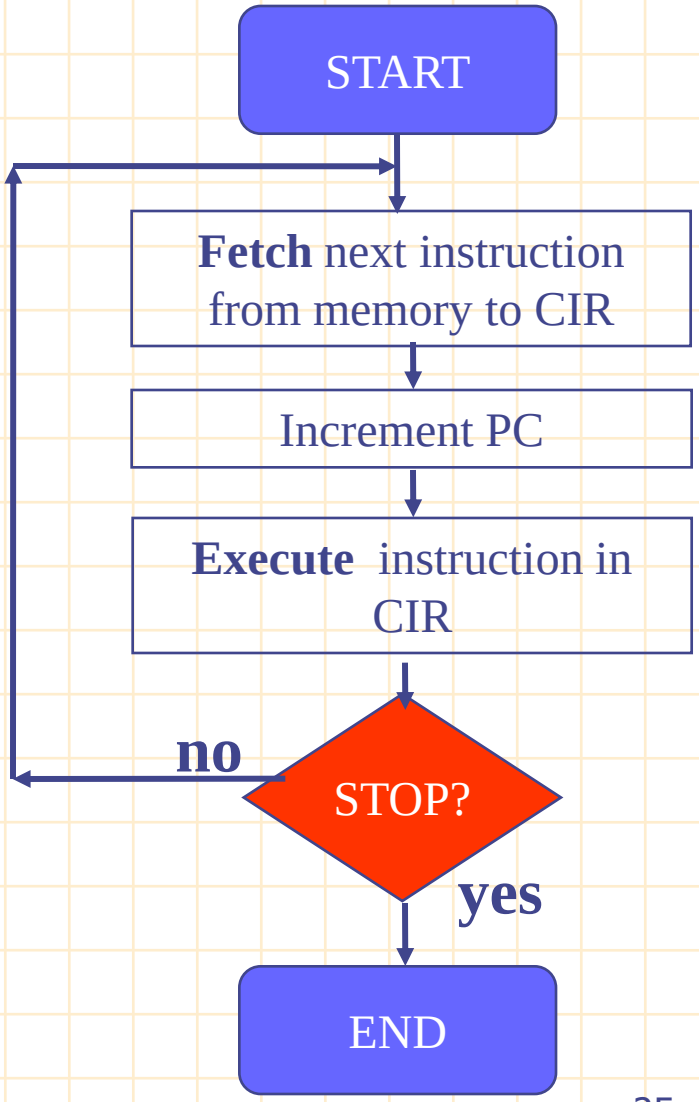


Von Neumann Model

- Problem?
  - Von-Neumann Bottleneck
  - All instructions must be retrieved from memory before they are processed
    - Memory (RAM) runs at slower speeds than the processor is capable of
    - The difference between the speed of the RAM and the speed of the processor is the 'bottleneck'
    - This is being remedied by faster RAM technologies
      - Such as SDRAM

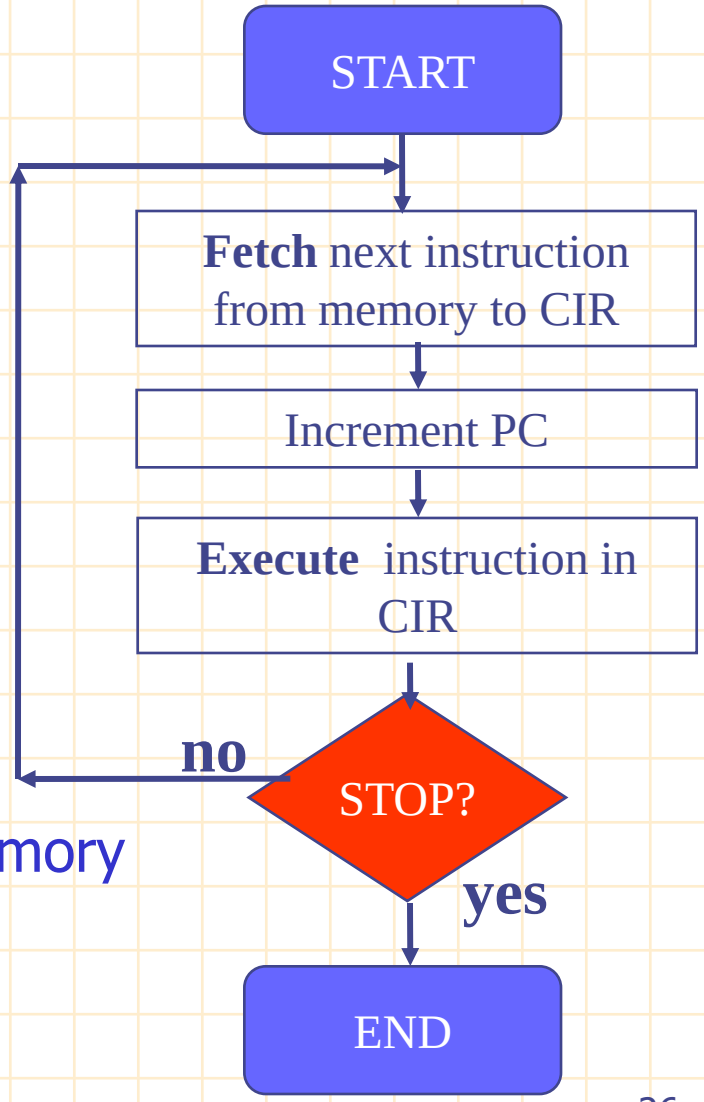
Fetch Execute Cycle

- Defines how instructions are retrieved and carried out inside the processor (CPU)
- Sometimes called the *Instruction Cycle* or *Automatic Sequence Control*

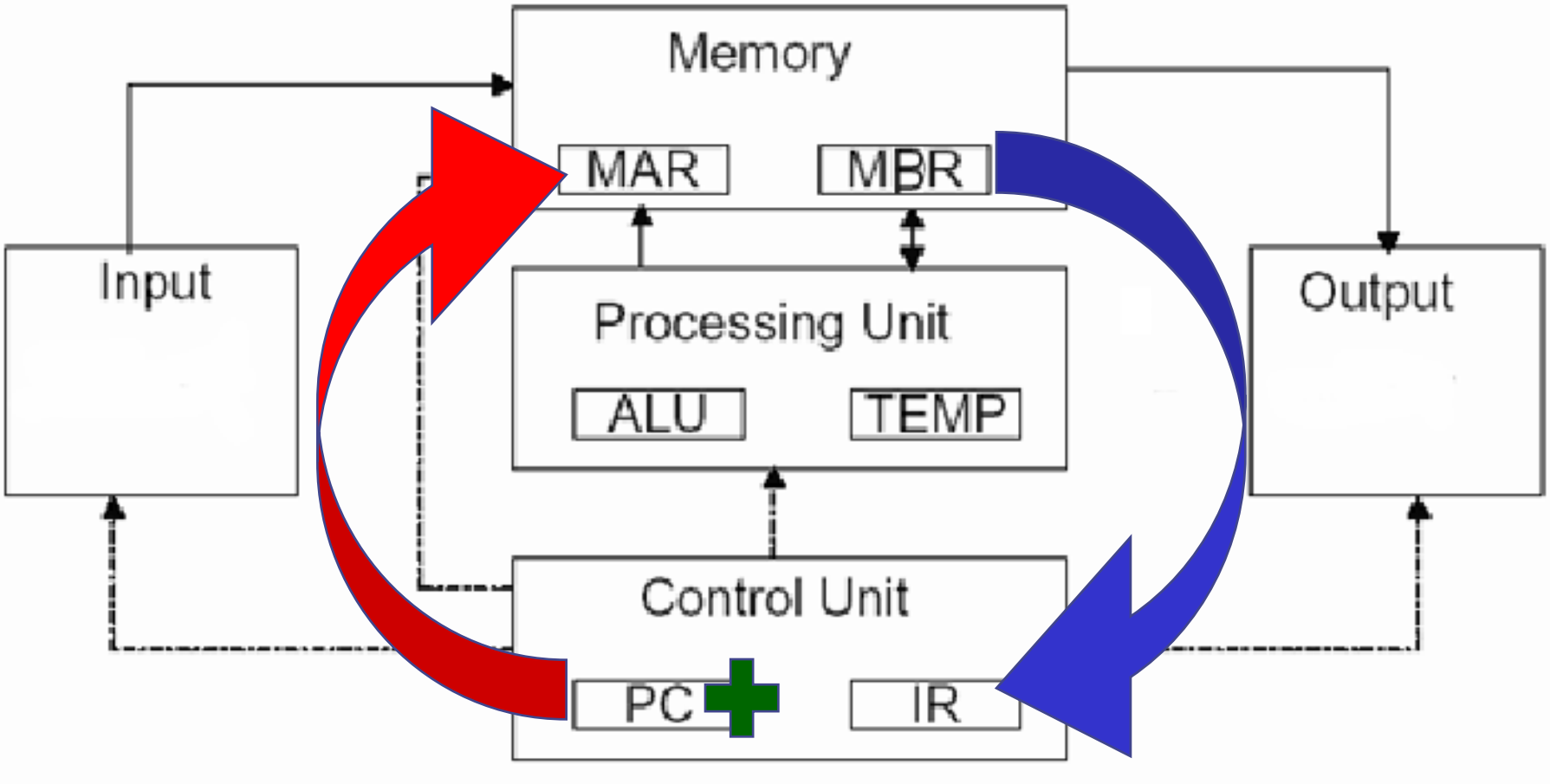


Fetch Execute Cycle

- **Fetch Stage**
  - Copy contents of PC into MAR
    - Value of PC presented via the address bus
  - Increment PC (point to next instruction)
  - Copy instruction from MBR into CIR via data bus
    - Instruction retrieved from memory
    - Placed in CIR

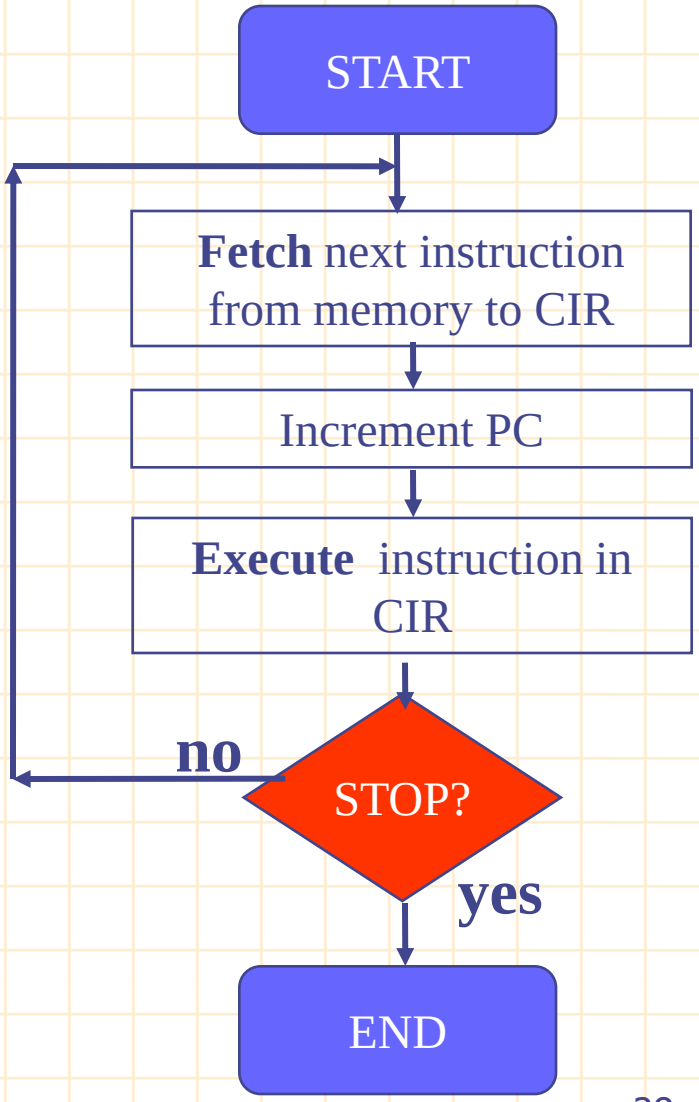


Fetch!

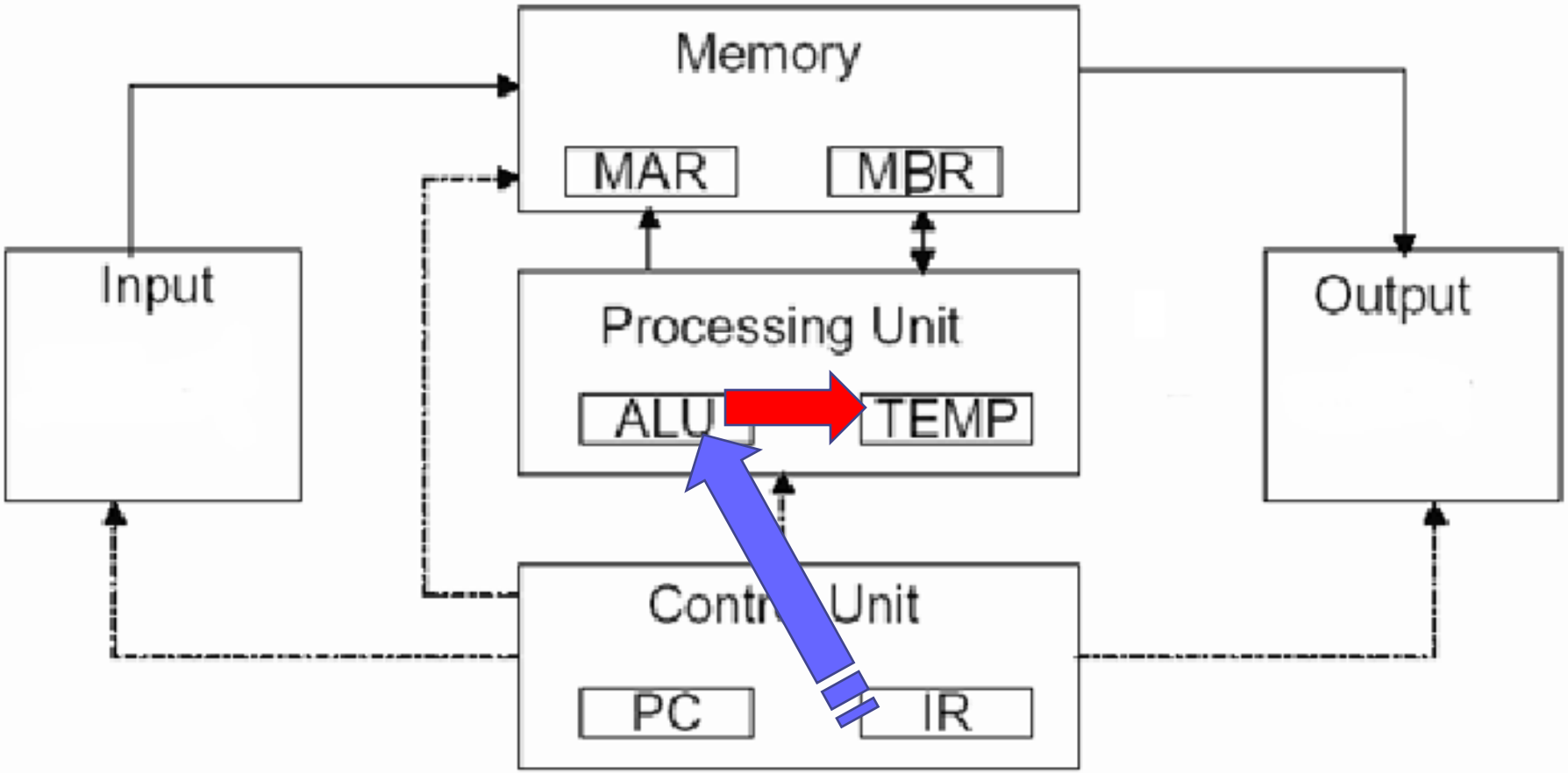

Memory
MAR
MBR
Input
Processing Unit
ALU
TEMP
Output
Control Unit
PC
IR

Fetch Execute Cycle

- Execute Stage
  - Decode instruction from CIR
  - Run instruction in CIR
    - May require getting data from memory
  - Unless current instruction is STOP, repeat cycle



Execute…


Memory
MAR
MBR
Input
Processing Unit
ALU
TEMP
Output
Contr Unit
PC
IR

MIPS & Hertz

- MIPS = Millions of Instructions per Second
  - How many instructions a processor can carry out each second
  - Old form of measurement
    - Inaccurate
    - Some instructions take longer than others
- Hz = number of complete cycles per second
- MHz = Millions of cycles per second
  - In a processor a cycle is when the state of the control lines are changed

CPU Instructions

- Instruction Set
  - The types of instruction that a particular machine can execute
  - The instructions that are carried out during the Fetch Execute Cycle
  - Types of instruction:
    - Arithmetic and logical calculations on data
    - Input and output of data
    - Changing the sequence of program execution
    - Transferring data between memory and CPU registers
    - Transferring data between CPU registers

CPU Instructions

- Instructions are split into two parts:
  - **Opcode** (Operation Code)- the operation to be carried out
  - **Operand**- The data upon which the operation should be performed
- Different manufacturers - different instruction sets
  - Can vary in architecture
  - Functions will often be the same or similar but may vary in name
  - More advanced functions may be present
    - Likely to vary between manufacturers
  - e.g.- Intel instruction set Vs. AMD instruction set

Assembly Language

- Is at a level below programming languages
  - Eg.- C++, Java, Pascal
- Assembly language is converted into machine code
  - Machine code is raw data that would take ages for a human to decipher
  - This is the data and instructions which is used by the Fetch Execute Cycle

Assembly Language

- Programs or sequences can be written in assembly language
  - Which is what is effectively done when we compile a C++ program
- Why write in assembly language?
  - Faster (direct) access to CPU
  - Some programs need to be written to operate at a lower level
    - E.g.- Device Drivers

A simple Assembly program (Honest!)

```
org 100h
    mov dx,msg
    mov ah,9
    int 21h
    mov ah,4Ch
    int 21h
    msg db 'Hello, World!',0Dh,0Ah,'$'
```

A simple Assembly program (Honest!)

```
org 100h
```

Tells the compiler (NASM) the program will be loaded at memory address 100h

```
mov dx,msg
```

Moves the address of our message (msg) into a register which is known as the DX Register (Data Register)

```
mov ah,9
```

Moves the value 9 into a register called the AH Register

```
int 21h
```

'int' calls an ISR (interrupt service routine) "DOS Services" this is combined with contents of AH (9) to determine that we want to output a message– contents of DX (msg)

A simple Assembly program (Honest!)

```
mov ah,4Ch
int 21h
```

Effectively tells the processor to stop (combines int 21h with contents of AH <now 4Ch>). Otherwise it will try to fetch and execute the next instructions it comes to

```
msg db 'Hello, World!',0Dh,0Ah,'$'
```

msg is a variable name (the name of out message string)

db is an instruction to the compiler to use the information the follows as data

Then out message ‘Hello, World!’ (note: ` ` marks)

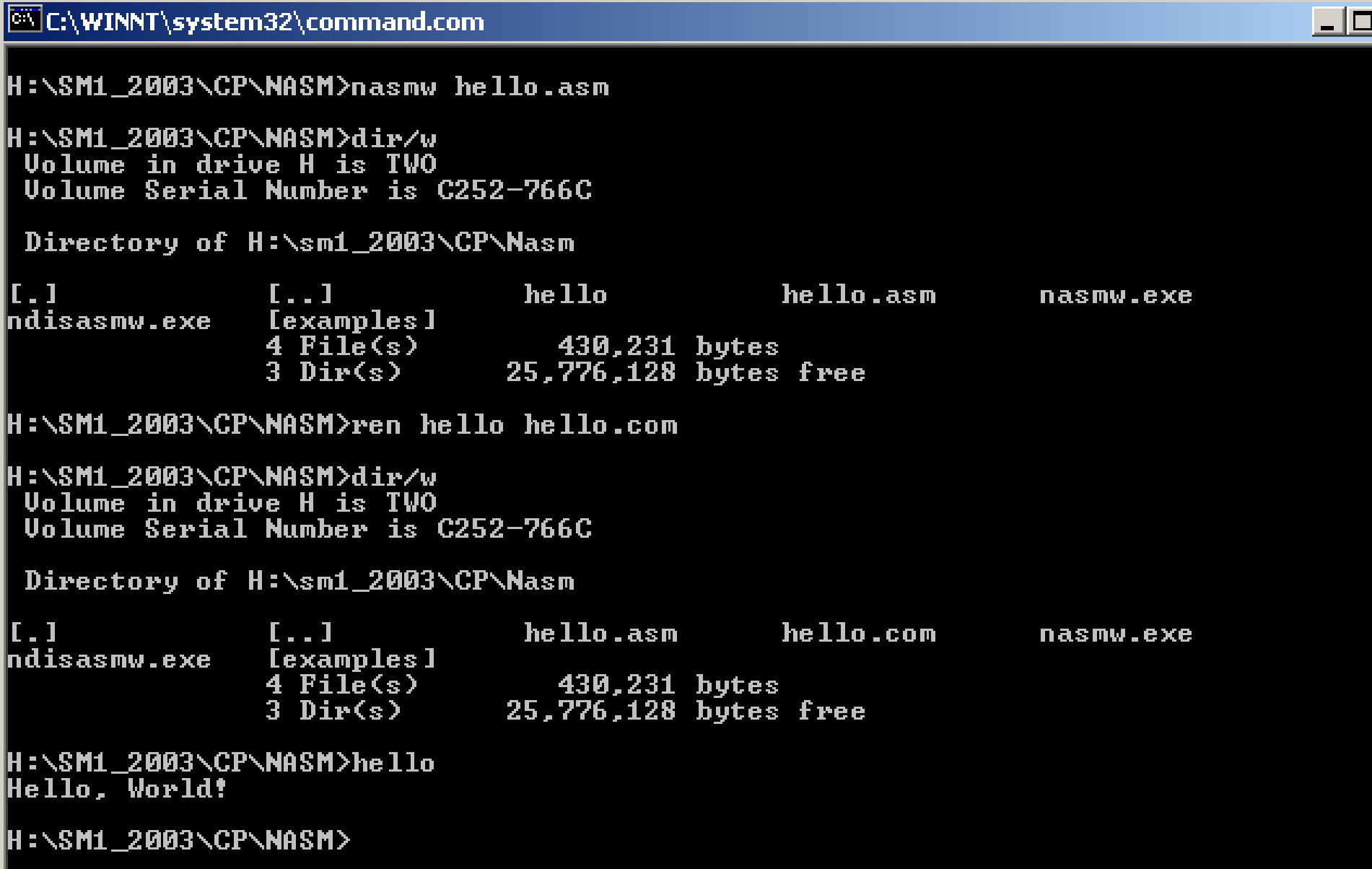
0Dh, 0Ah – performs carriage return and line feed

$ terminate string output – (int 21h & 9 in ah requirement)

A simple Assembly program (Honest!)

- OK, so what does it actually do?
  - Output "Hello, World!" to the screen
- How?
  - Type the program into a text document and call it '**`hello.asm`**'
  - Use the NASM program
    - This is used to compile assembly language programs
  - Rename the produced file as type COM
    - **`ren hello hello.com`**
  - Run the program
    - **`hello`**

A simple Assembly program (Honest!)



Von-Neumann Bottleneck

- Originally CPU & RAM ran at similar speeds
  - CPU development began to increase faster than RAM
- Fundamental problem:
  - CPU is faster than RAM

- Attempt to solve the problem:
  - Sophisticated RAM technologies

www.knozall.com/squeezingthroughthevonneuman.htm

Memory (RAM)

- Many different types of RAM
  - **SDRAM**
    - Synchronous Dynamic RAM
    - Synchronises with processor buses
    - Max. approx 133 MHz
    - Contemporary
  - **DDR SDRAM**
    - Double Data Rate SDRAM
    - Transfers data on both sides of the clock cycle
    - Effectively doubles transmission rate

Memory (RAM)

- **FPRAM**
  - Fast Page RAM
  - "Page Mode Memory"
  - Dynamic RAM
  - Allows faster access to adjacent memory locations
    - Does not always store complete addresses
- **NVRAM**
  - Non Volatile RAM
  - Retains it's contents when power is switched off
  - Powered by a battery
  - Or uses an EEPROM chip
    - Electrically Erasable Programmable ROM
    - Combination of SRAM and EEPROM chips
  - SRAM is an NVRAM derivative

Video Memory (VRAM)

- **WRAM**
  - Windows RAM
    - Windows are large blocks of memory
    - Supports two paths to transport data
      - Sends data for display as new information is being sent to the graphic adapter's memory
      - Same principle as standard VRAM
      - Faster than ordinary VRAM because of Windowing
- **SGRAM**
  - Synchronous Graphic RAM
    - Dynamic RAM
    - Synchronise with processor buses up to 100 MHz
    - Capable of opening two memory pages at once
      - Simulates dual data transmission of VRAM and WRAM
    - Better than standard VRAM