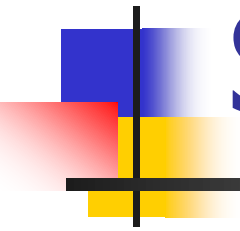
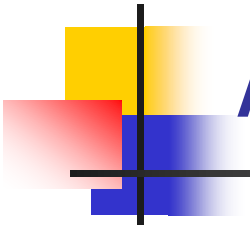


Evolution of Computer Systems

BIT211: Computer Systems Organization



Agenda

- Evolution of computer system
- Turing Machine: How computers function
- The Von-Neumann Architecture
- The fetch-execute cycle

EVOLUTION OF COMPUTER SYSTEMS

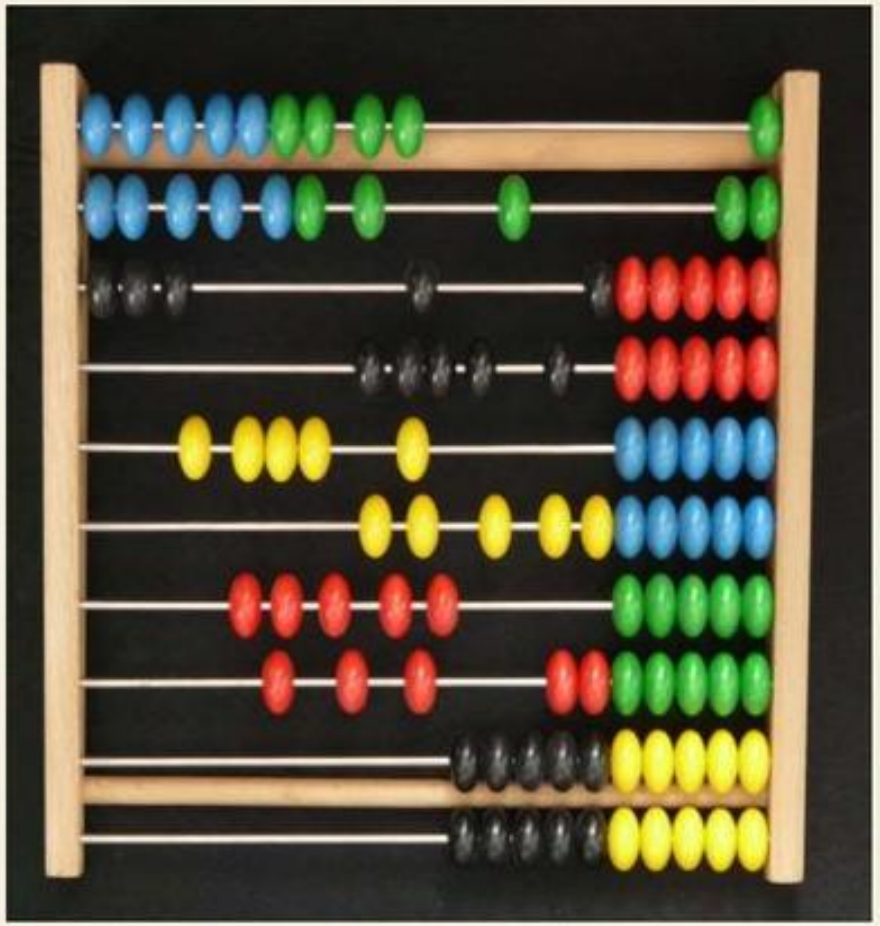
A computer system is a basic, complete and functional computer, including all the hardware and software required to make it functional for a user.

It should have the ability to receive user input, process data, and with the processed data, create information for storage and/or output.

A computer system allows users to input, manipulate and store data. Computer systems typically include a computer, monitor, keyboard, mouse and other optional components. All of these components also can be integrated into all-in-one units, such as laptop computers.

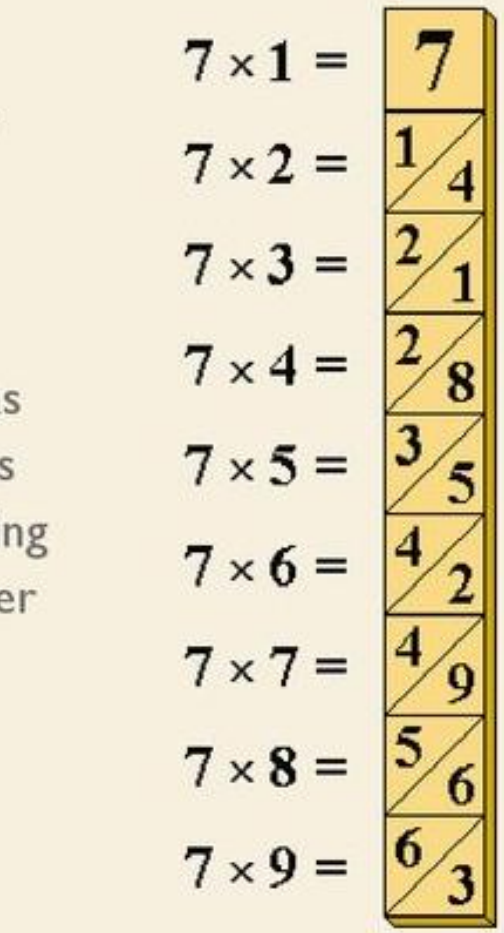
EVOLUTION OF COMPUTER SYSTEMS

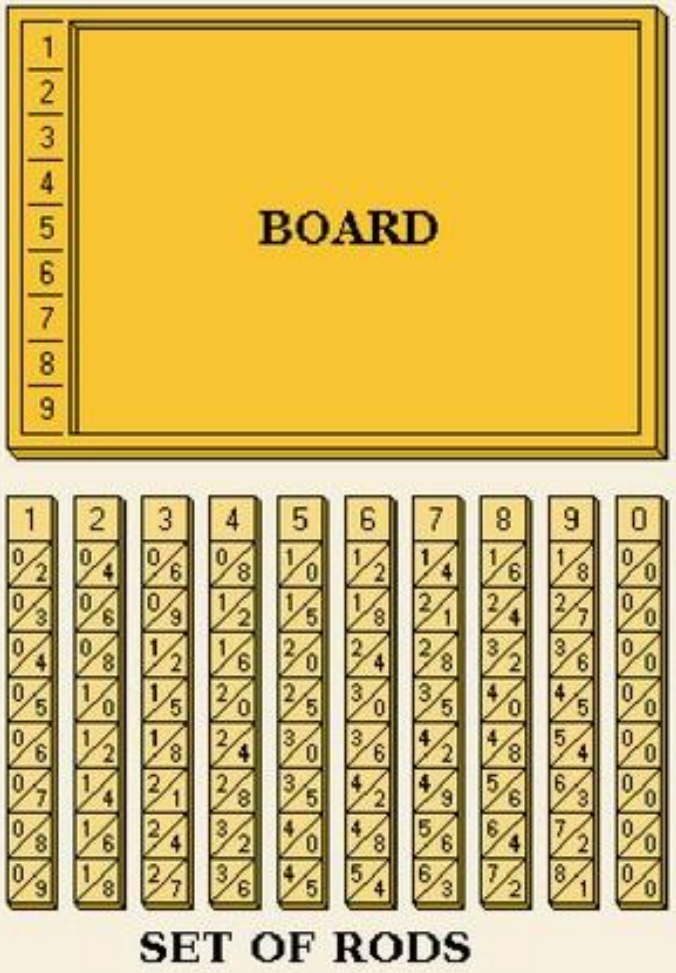
- **ABACUS**- Many centuries ago when man started to count the numbers, he thought of a device which can trace the numbers and thus came the existence of ABACUS. It was the first counting device which was developed in China more than 3000 years ago. The name Abacus was obtained from Greek word Abax which means slab. This device basically consists of a rectangular wooden frame and beads. The frame contains horizontal rods and the beads which have holes are passed through the rods. Counting was done by moving the beads from one end of the frame to the other.



EVOLUTION OF COMPUTER SYSTEMS

- **Napier's Bones-** It is a device which contains a set of rods made of bones. It was developed by John Napier, a Scottish Mathematician and hence the device was named as Napier's Bones. The device was mainly developed for performing multiplication and division. Later in 1614 he also introduced logarithms.





EVOLUTION OF COMPUTER SYSTEMS

- **Pascaline-**Pascaline is a calculating machine developed by Blaise Pascal, a French Mathematician. It was the first device with an ability to perform additions and subtractions on whole numbers. The device is made up of interlocked cog wheels which contains numbers 0 to 9 on its circumference. When one wheel completes its rotation the other wheel moves by one segment. Pascal patented this device in 1647 and produced it on mass scale and earned a handful of money.

FIRST GENERATION (1940-1956)

The first generation of computer were huge, slow, expensive and often unreliable. In 1946, two Americans, Presper Eckert and Willian Mauchly build the ENIAC (Electronic Numerical Integrator and Computer). It use vacuum tube instead of mechanical switches of the MARK

MARK I Features:

- It could perform five basic arithmetic operations: addition, subtraction, multiplication, division and table reference
- It took approximately 0.3 seconds to add two numbers and 4.5 seconds for multiplication of two numbers

MARK I Disadvantages :

- It was huge in size
- Complex in design.
- Very slow

FIRST GENERATION (1940-1956)

- **UNIVAC-** The UNIVAC (universal automatic Computer) was the first digital computer invented by Mauchly and Ekert

In 1951, Eckert and Mauchly build the UNIVAC, which could calculate at the rate of 10,000 addition per seconds.

SECOND GENERATION (1956-1963)

Features :

Vacuum tubes were replaced by *transistors*.

Transistor is a small device that transfers electronic signals through resistors

The creation of transistor spark the production of a wave of second generation computer. Transistor was small devices use to transfer electronic signals across a resister. Transistors had many advantages compared to other hardware technology.

transistors were smaller than vacuum tubes

- they needed no warm up time
- consumed less energy
- generated much less heat
- faster and more reliable
-

THIRD GENERATION (1964-1971)

Features :

In this generation microelectronics technology was introduced that made it possible to integrate large number of circuit elements into very small surface of silicon known as chips. This new technology was called *INTEGRATED CIRCUIT INTEGRATED CIRCUIT*

Advantages A new concept in this generation was that of a family of computer which allowed computer to be upgraded and expanded as necessary. ·

- Silicone chips were reliable, compact and cheaper.
- · Sold hardware and software separately which created the software industry.
- · customer service industry flourished (reservation and credit checks)

FOURTH GENERATION (1971-PRESENT)

It took only 55 years for the 4 generations to evolve. The growth of the computer industry developed technologies of computer inventions. There are many types of computer models such as:

- ·Apple Macintosh 
- ·IBM 
- ·DELL 
- ·ACER 

FOURTH GENERATION (1971-PRESENT)

In 1971 Intel created the first microprocessor. In 1976, Steve Jobs built the first Apple computer. Then, in 1981, IBM introduced its first personal computer.

During the fourth generation, hardware technology such as silicone chips, microprocessor and storage devices were invented. A microprocessor is a specialized chip which is developed for computer memory and logic.

FOURTH GENERATION (1971-PRESENT)

The microprocessor is a large-scale integrated circuit which contained thousands of transistors. The transistors on this one chip are capable of performing all of the functions of a computer's central processing unit.

Advantages

- · Computers became 100 times smaller than ENIAC (Electronic Numerical Integrator and Computer) the first computer
- · Gain in speed, reliability and storage capacity
- · Personal and software industry boomed

FIFTH GENERATION (PRESENT & BEYOND)

The fifth generation computers are technologically advance and are still being development to become more efficient. The inventions of new hardware technology in the fifth generation have grown rapidly including many other modern computer devices such as :

- · silicone chips
- · processor
- · robotics
- · virtual reality
- · intelligent systems
- · programs which translate languages

Historical perspectives – Turing's machine

- Alan Turing – developed the abstract notion of a "Turing machine":
  - NOT a physical machine, but a mathematical model of one type of **very simple** abstract calculating device
  - He proved that there were Turing machines (Universal machines) that were capable of calculating anything and everything that any Turing machine could calculate (given enough time and enough memory)

The upshot of this result of Turing's:

- Very simple devices can be designed that are as powerful computers as can be (if we don't have to worry about how much time it will take or how much memory is involved)

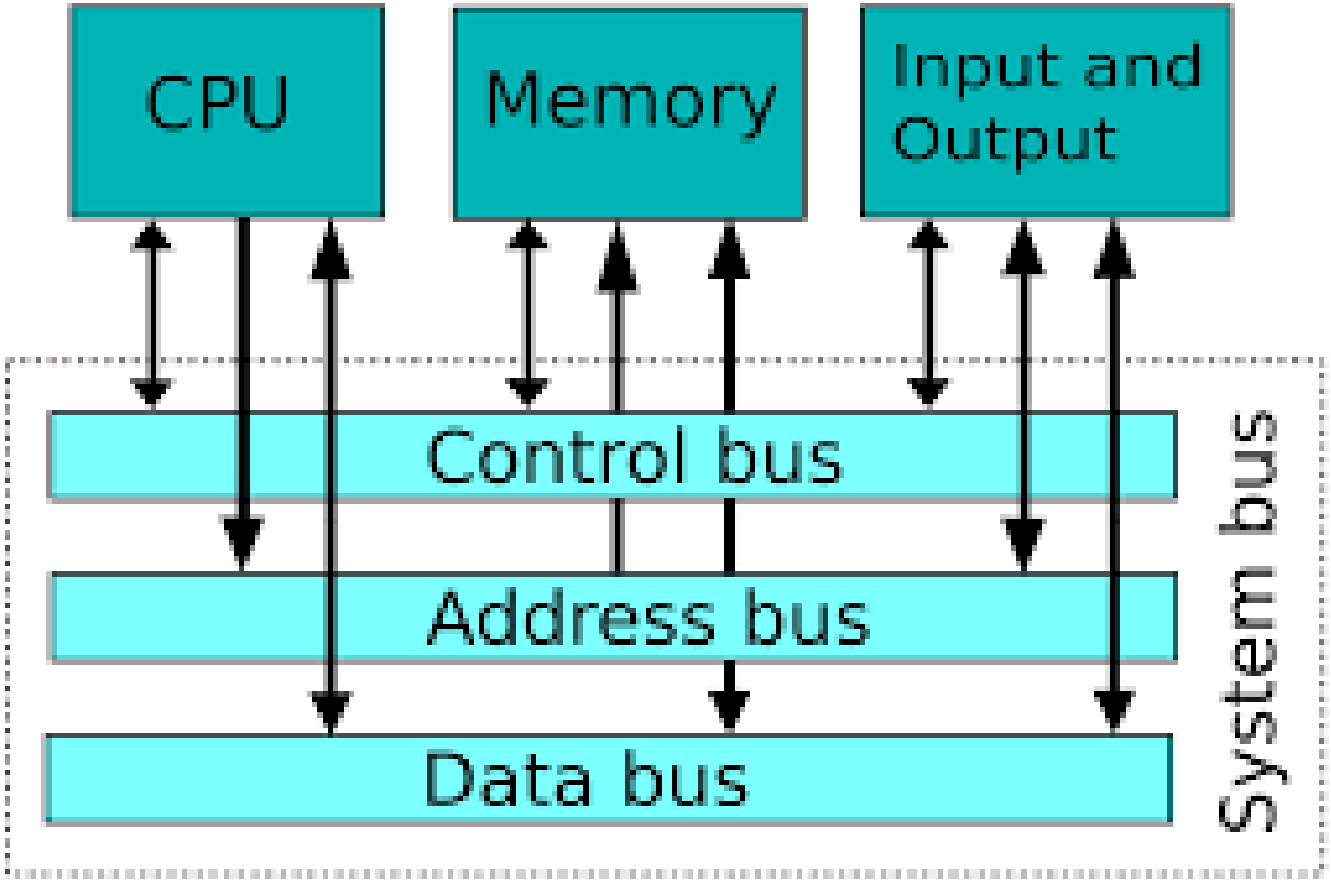
- The practical lesson – make things that have larger and more efficient memory and faster speeds and we can do anything that a computer can do

Historical perspectives: von Neumann's architecture

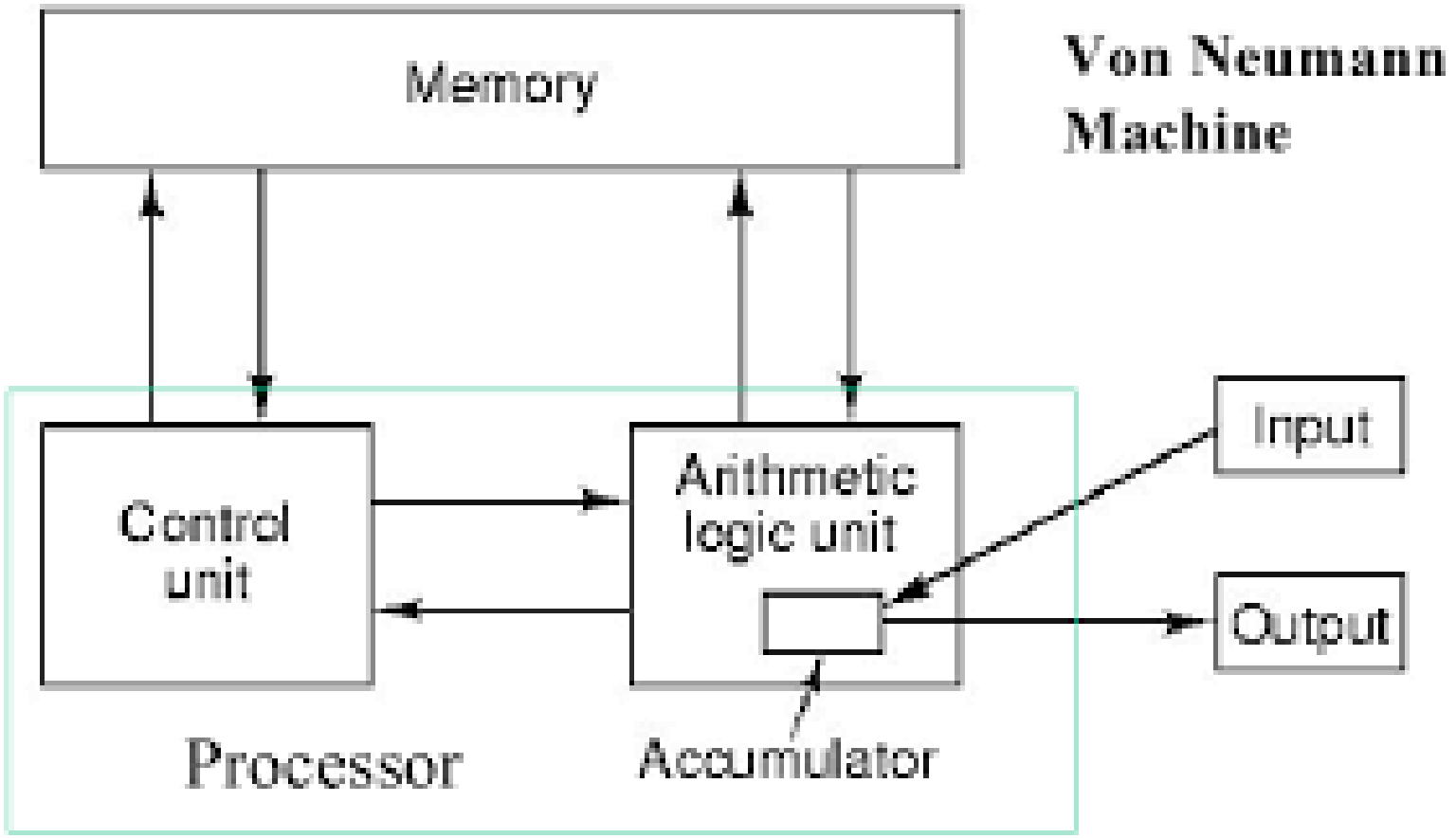
- Von Neumann added in a number of ideas that made digital computing practical and his ideas remain those of most digital programming to this day

- Key among those ideas:
  - Programs themselves are really just like any other piece of data (0s and 1s) at least from the standpoint of the computer

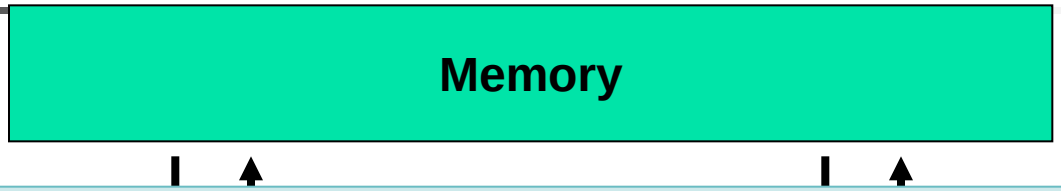
Von Neumann architecture (simplified)



Von Neumann architecture



Von Neumann architecture



Stores the program and the data

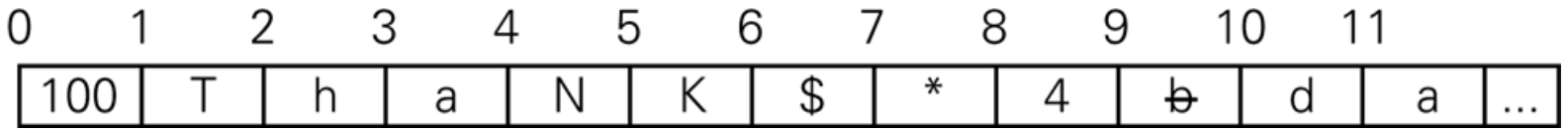
It has specific and discrete locations – each (generally) 1 byte

Each location has a specific address

These addresses store values

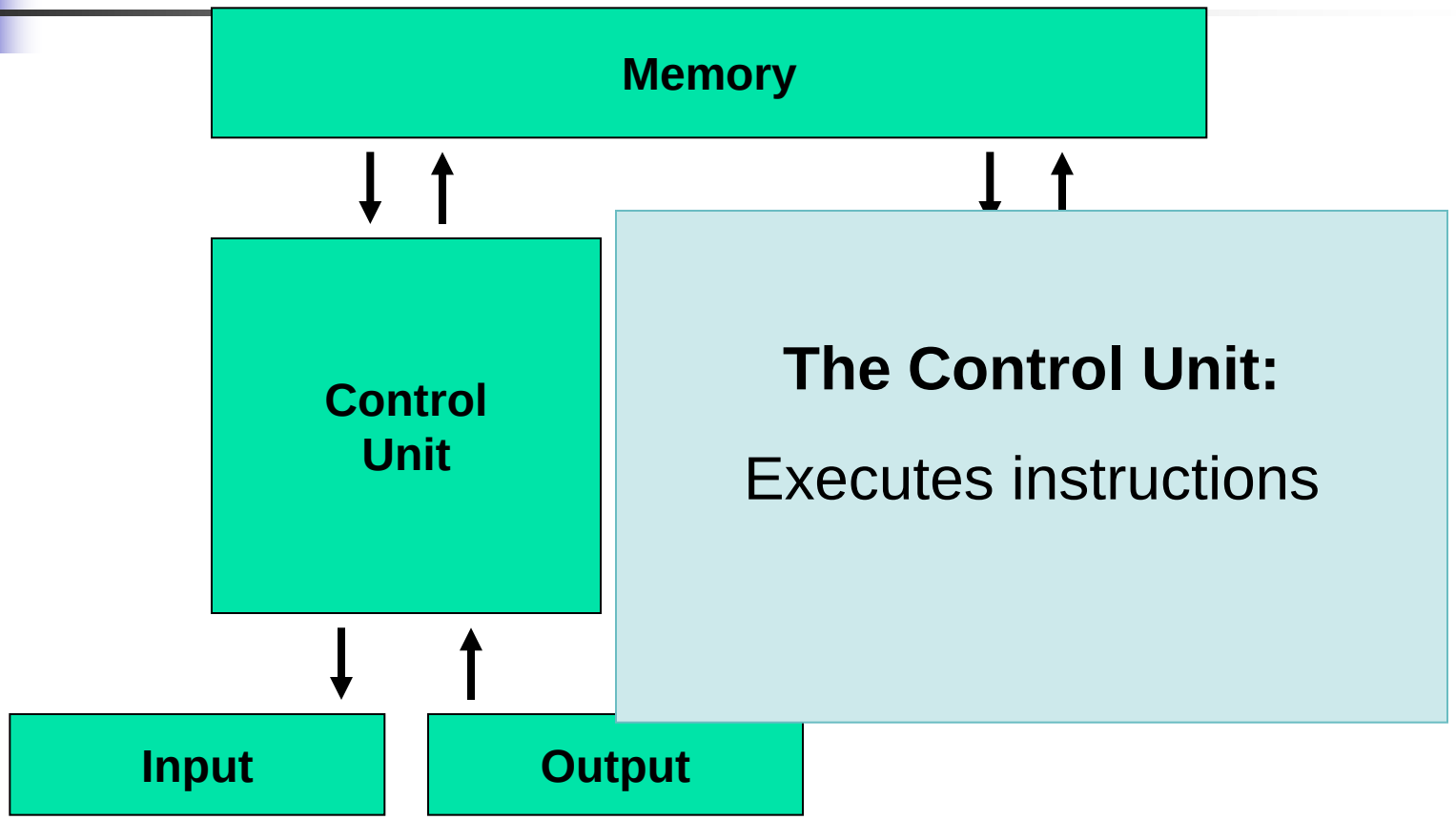
The total size of memory locations is limited

Diagram of computer memory

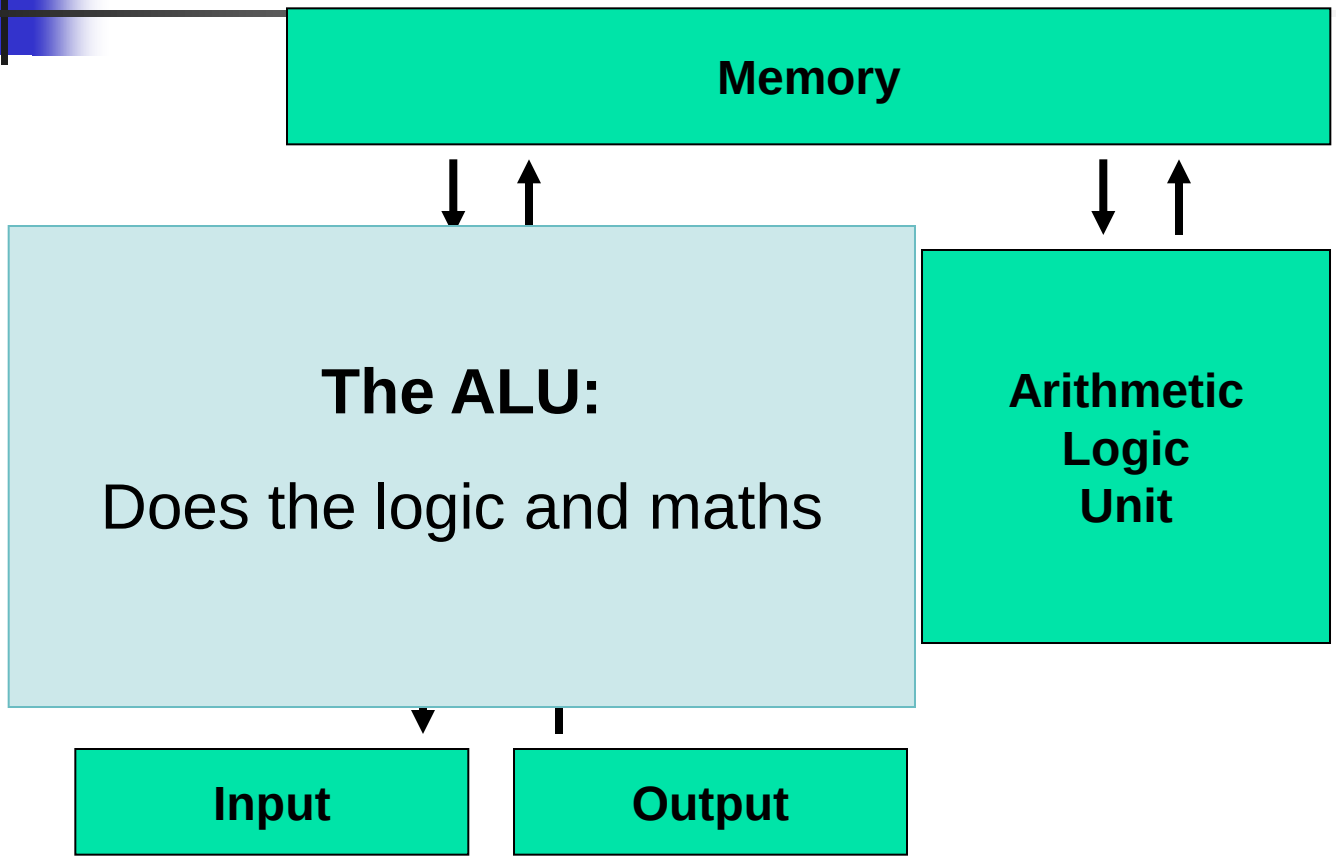


01010101	00101010	00111010	11101001

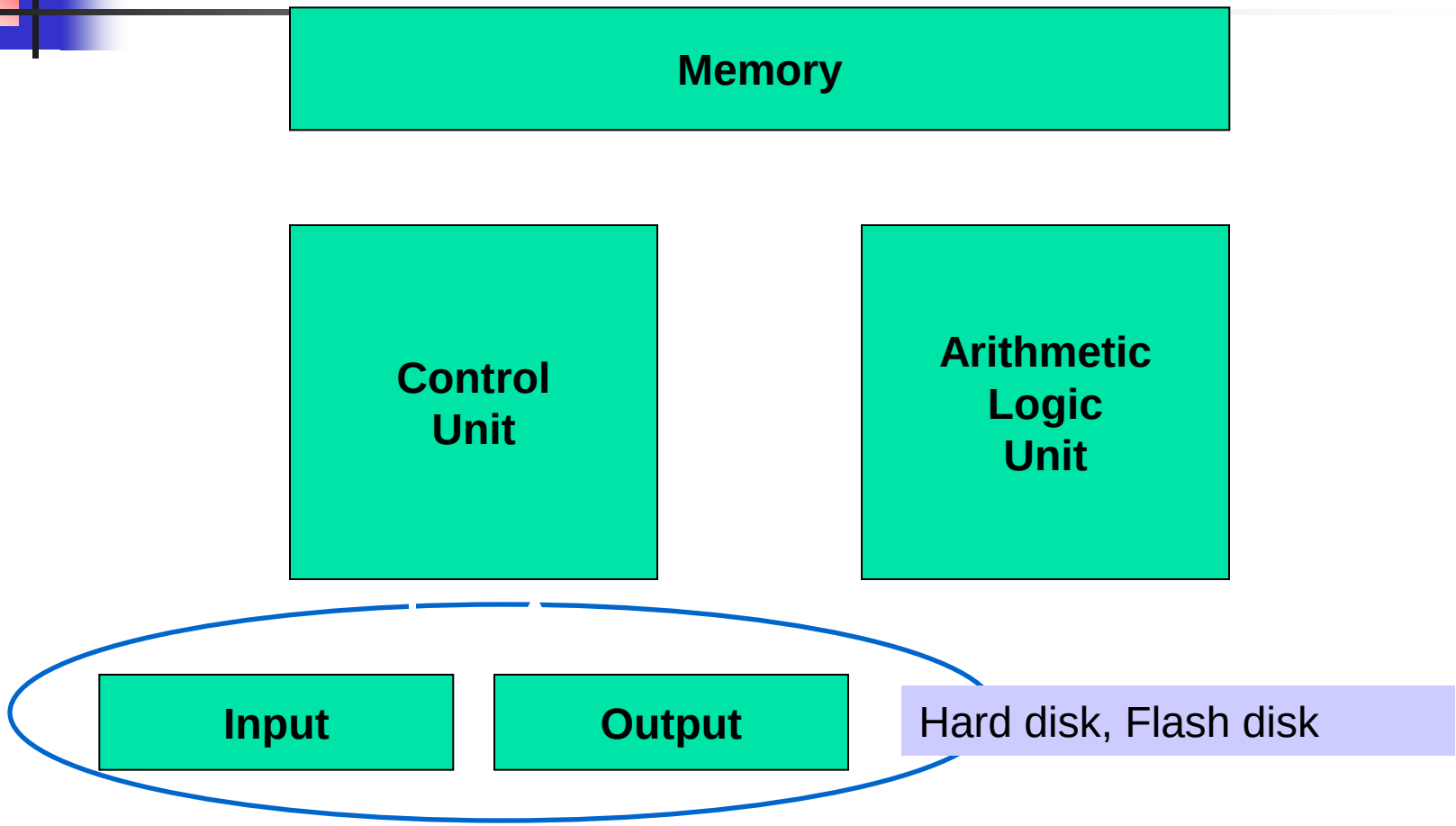
Von Neumann architecture (simplified)



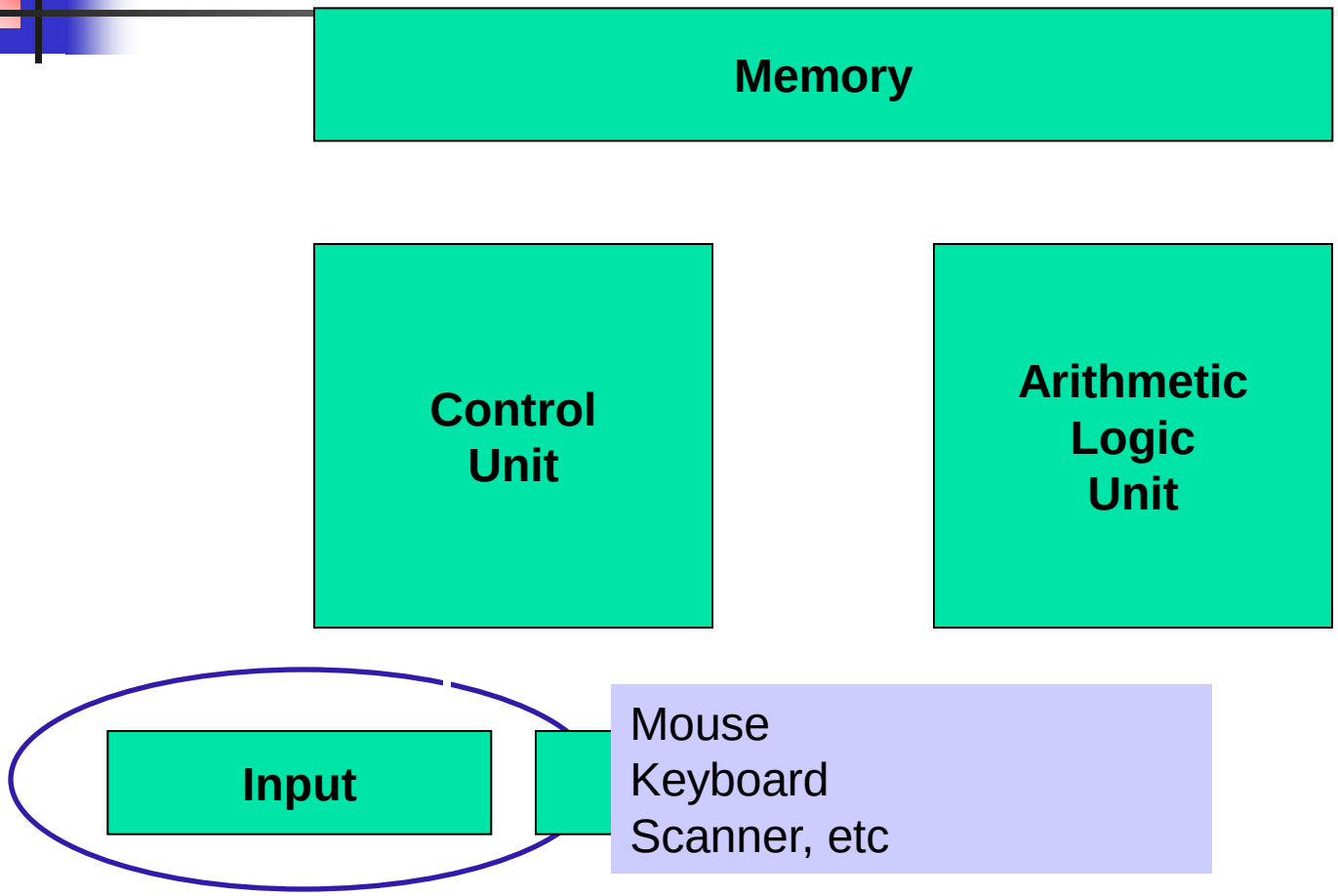
Von Neumann architecture (simplified)



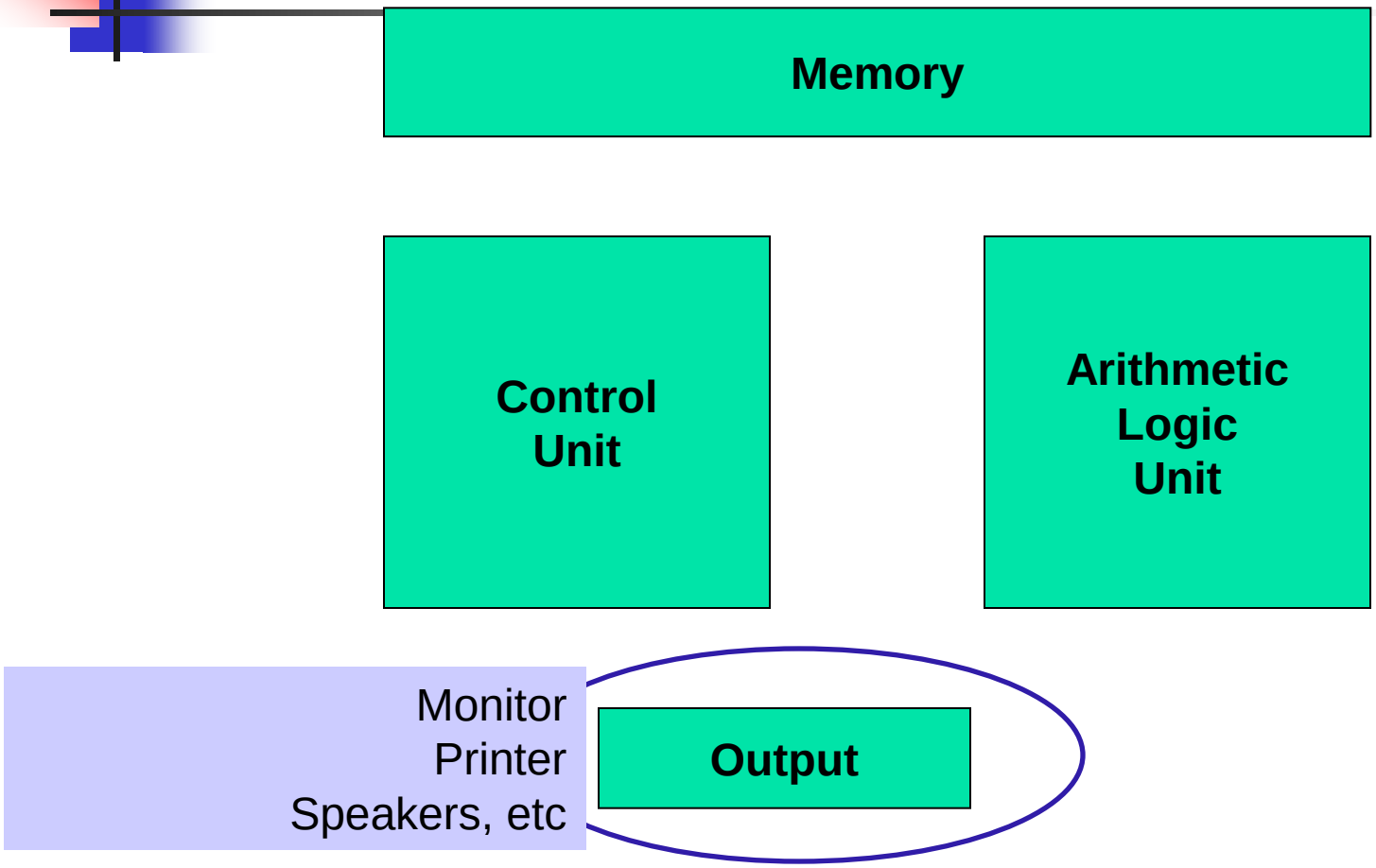
Devices for both input and output



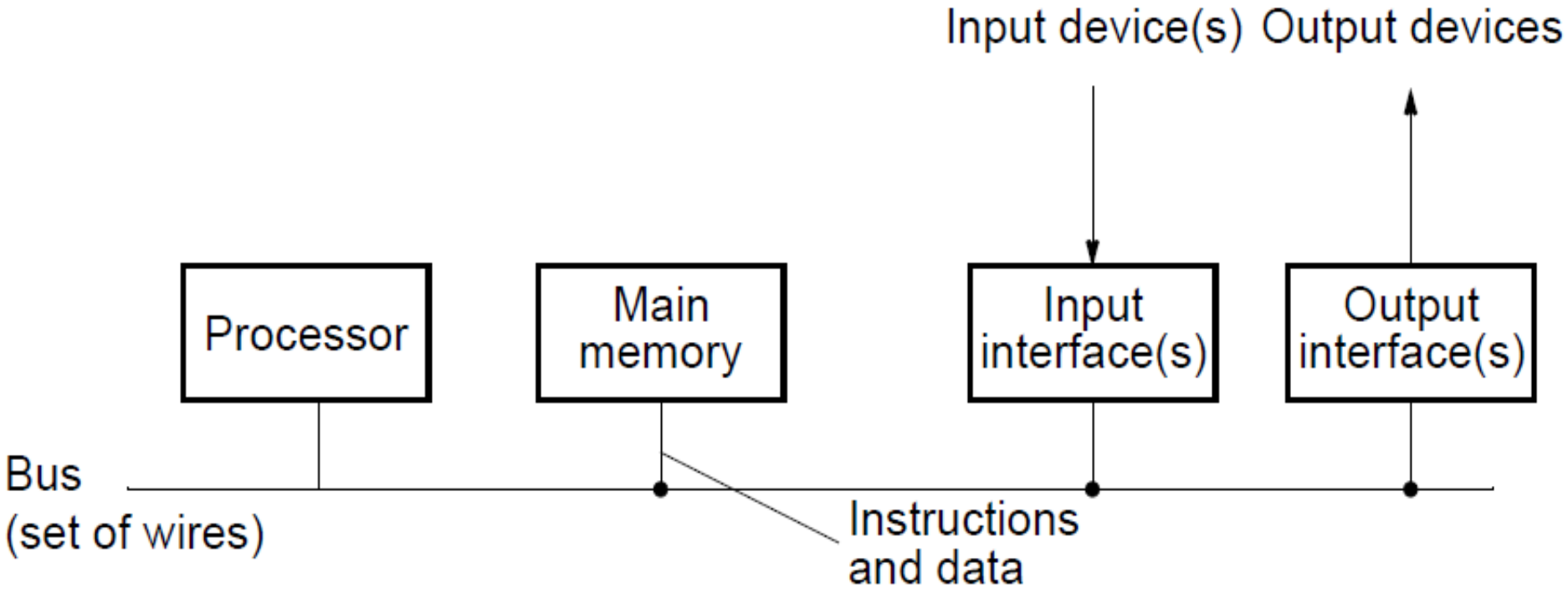
Input devices



Output devices



Internal Structure of computer



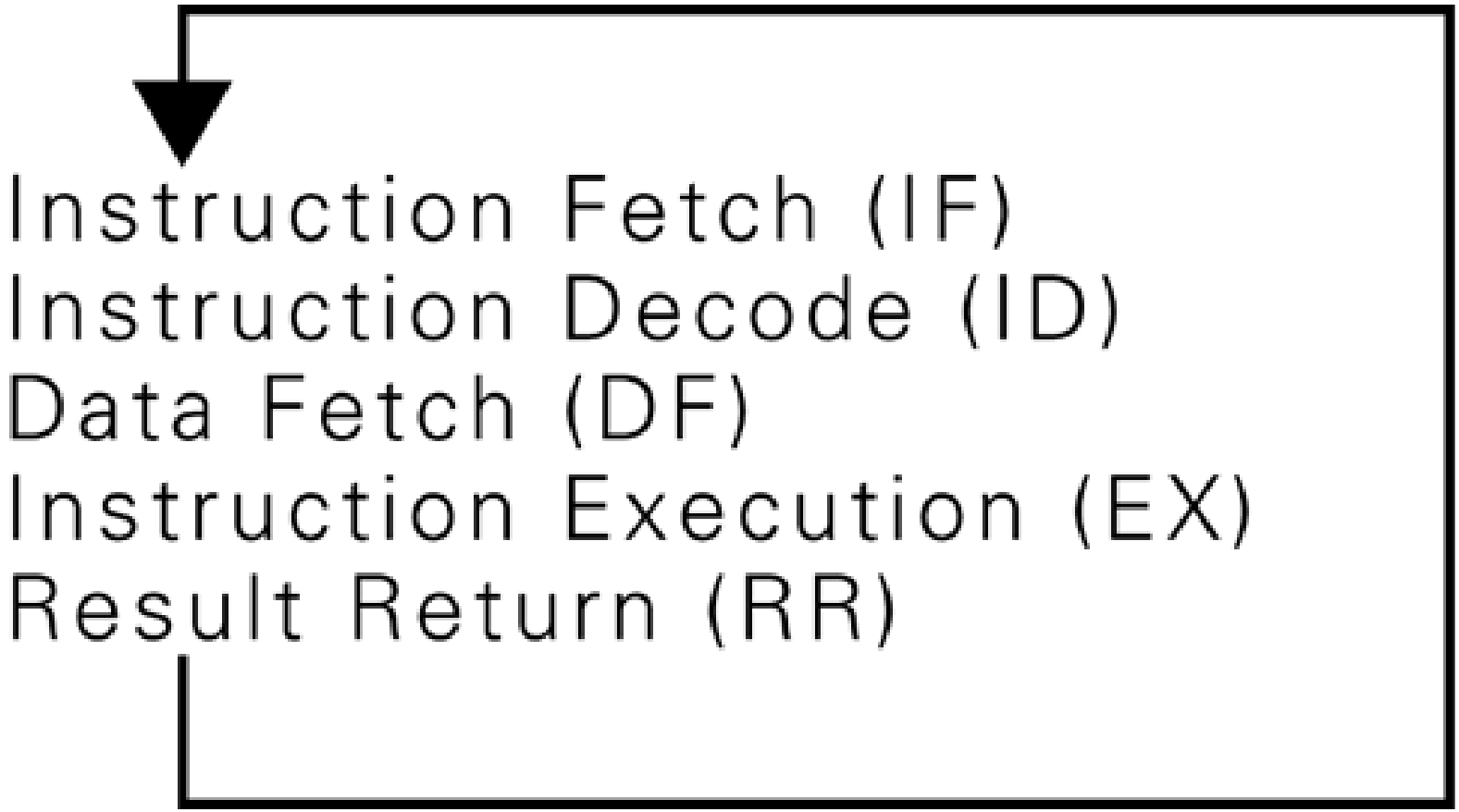
Computer buses

- A bus is a set of parallel wires connecting two or more components of the computer.

- **Data bus** – carries data from a given location to the CPU
- **Address bus** – carries the address of a particular memory location during transfer of data to memory
- **Control bus** – control signals are sent along this bus.

- The data, address and control buses connect the processor, memory and I/O controllers. These are all system buses.

The Fetch-Execute cycle.



Fetch-Execute Cycle

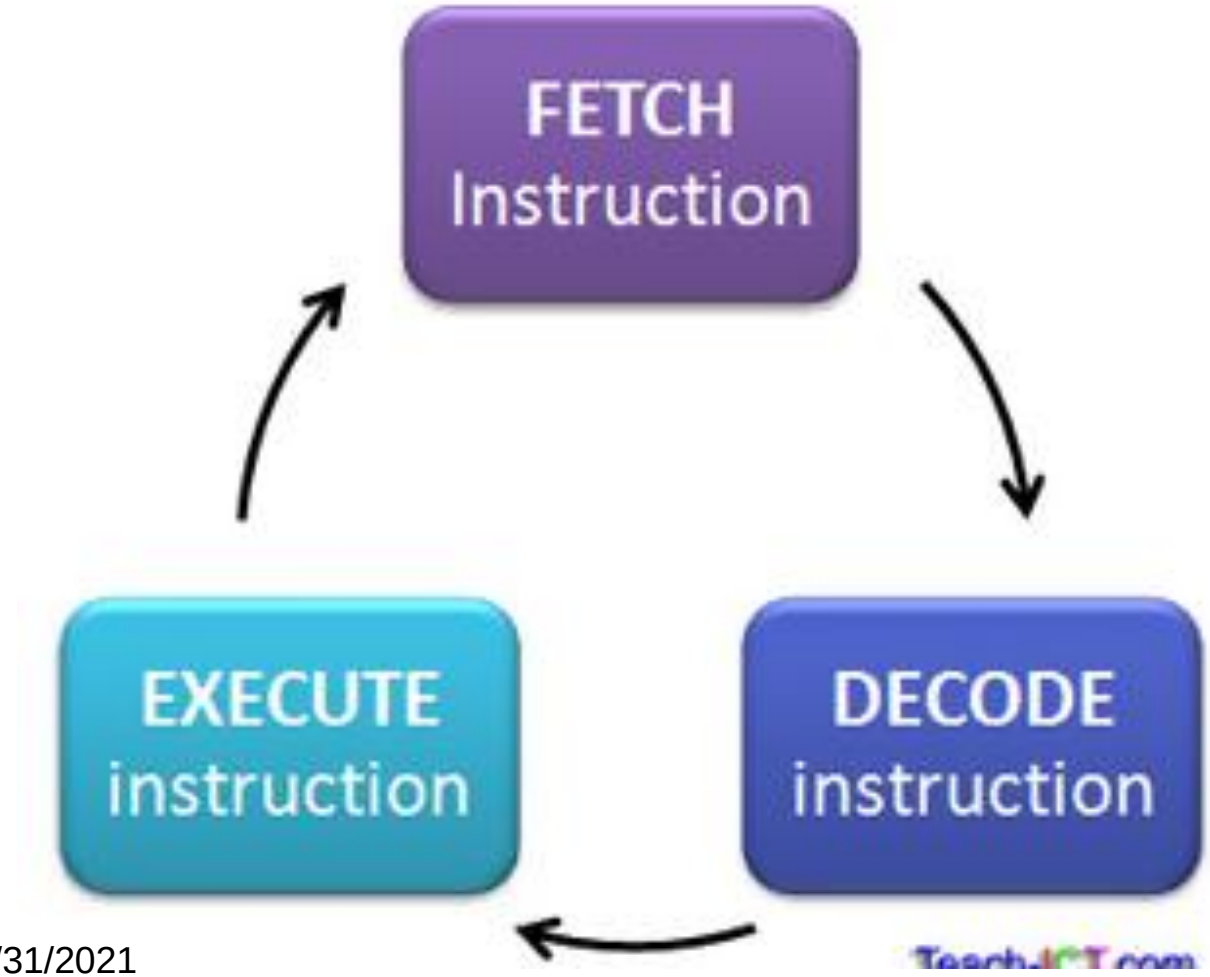
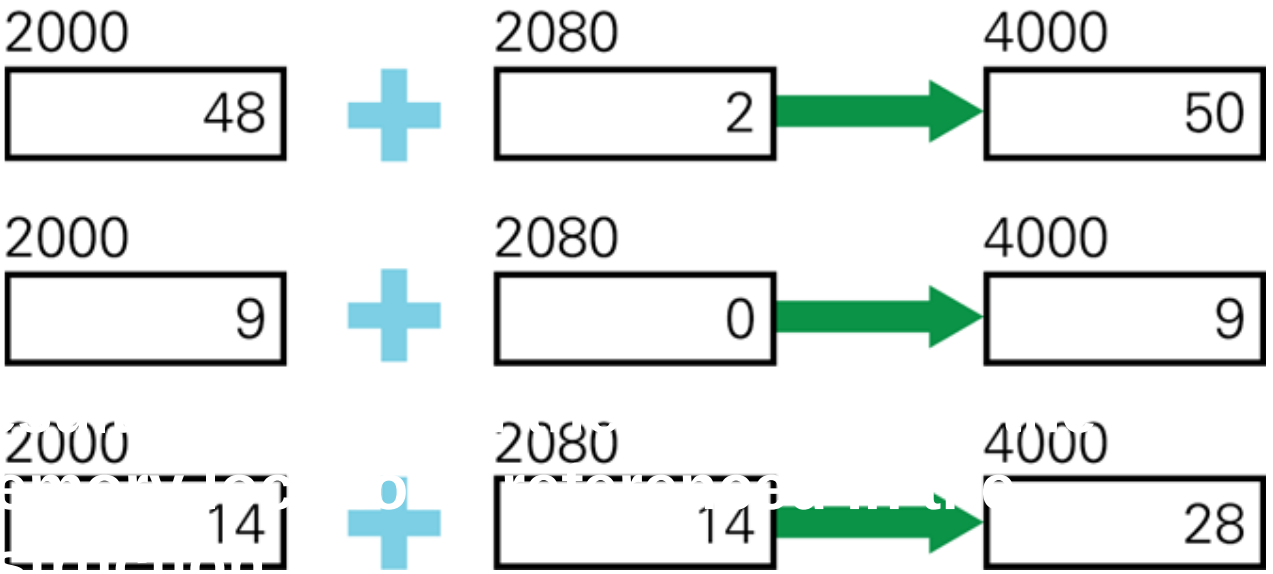
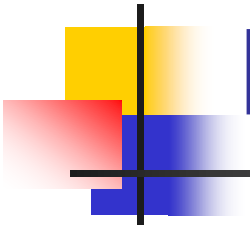


Illustration of a single instruction

ADD 2000 2080 4000



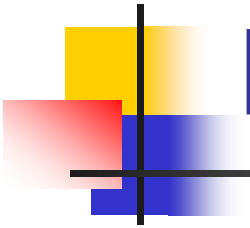


Fetch-Execute Cycle

1) Load the address that is in the program counter (PC) into the memory address register (MAR).

2) Increment the PC by 1.

3) Load the instruction that is in the memory address given by the MAR into the memory data register (MDR).

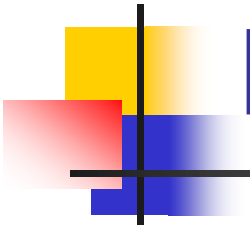


Fetch-Execute Cycle

4) Load the instruction that is now in the MDR into the current instruction register (CIR).

5) Decode the instruction that is in the CIR.

6) If the instruction is a jump instruction then
   a) Load the address part of the instruction into the PC
   b) Reset by going to step 1.

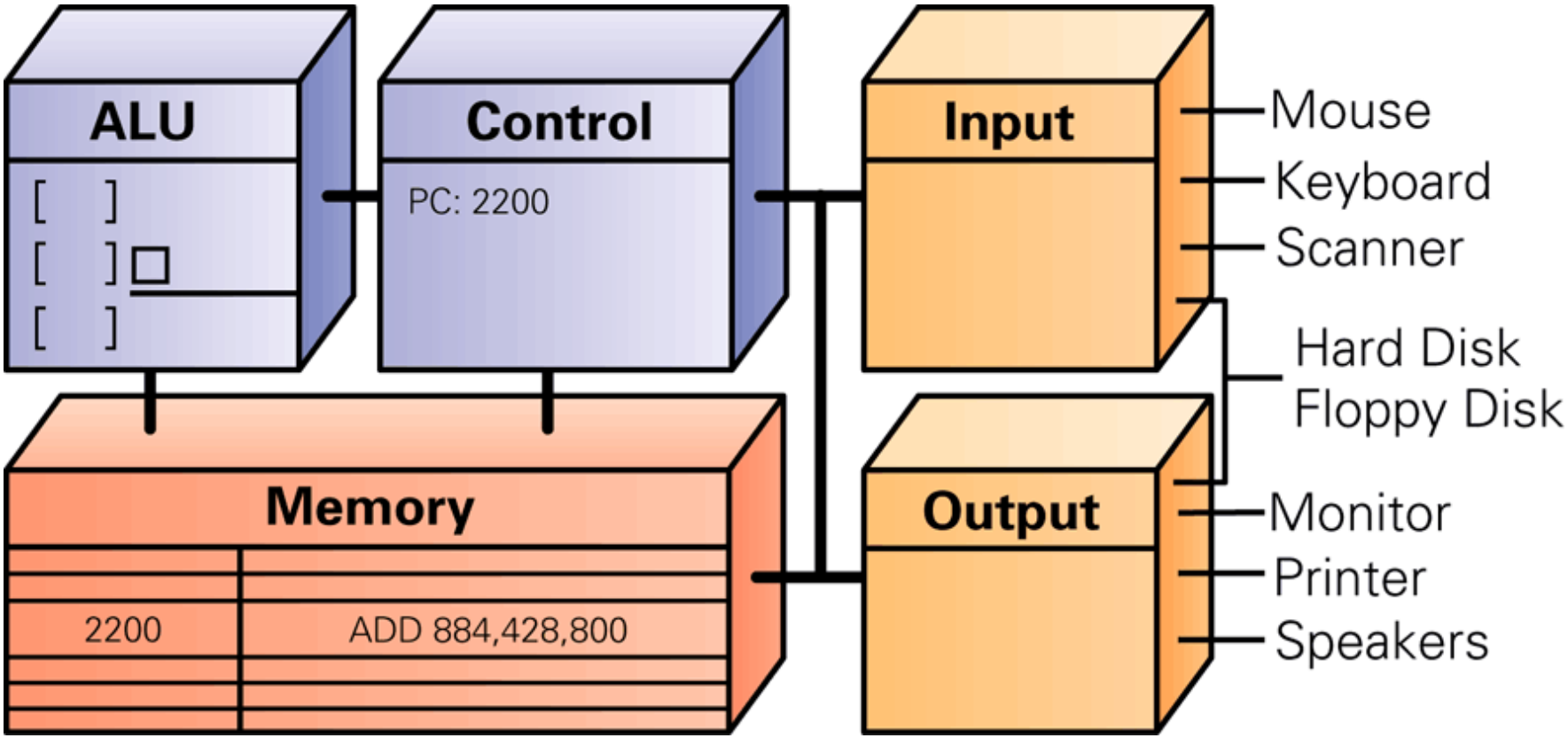


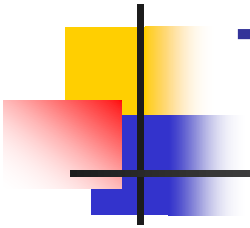
Fetch-Execute Cycle

7) Execute the instruction.

8) Reset by going to step 1.

Computer before executing an ADD instruction.

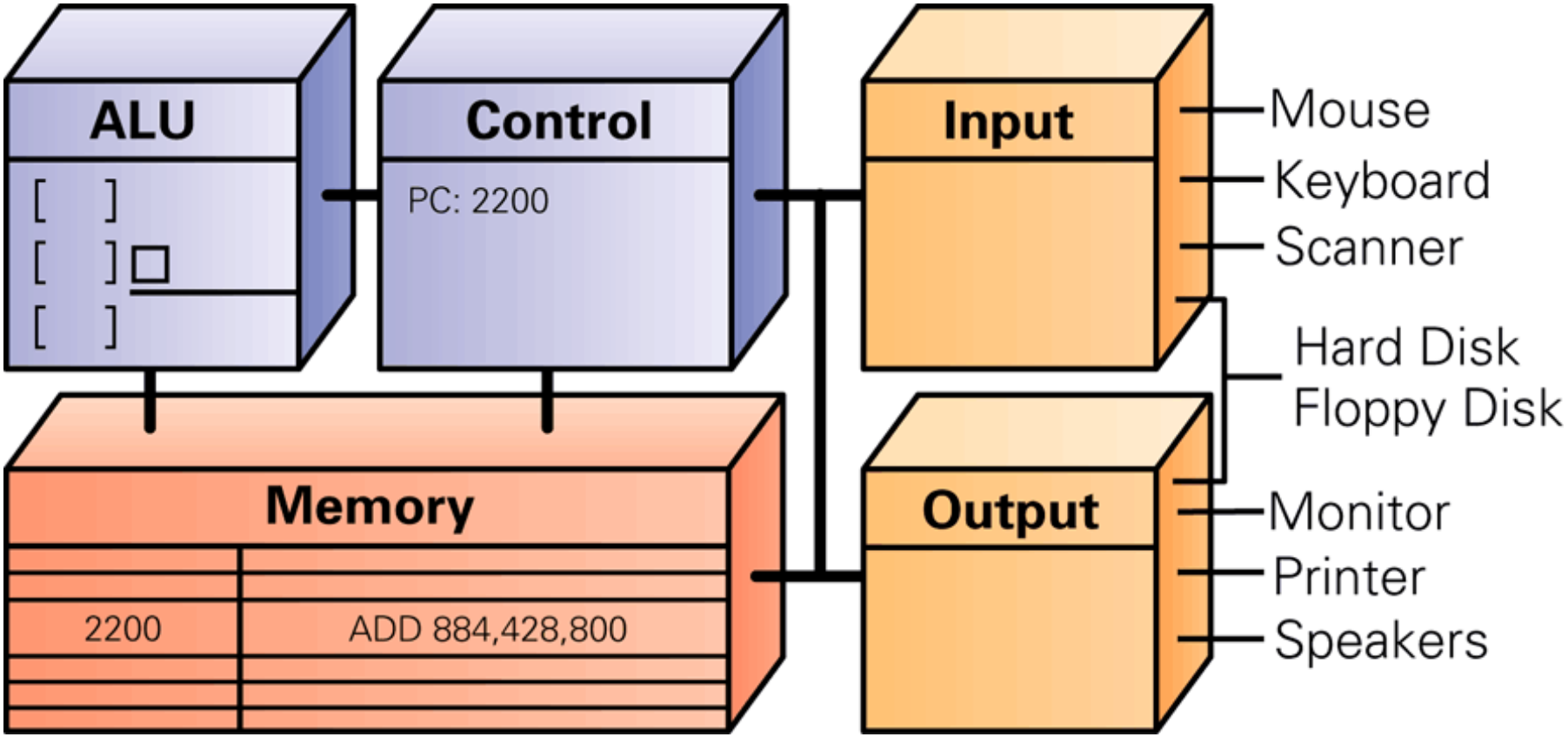




The Program counter

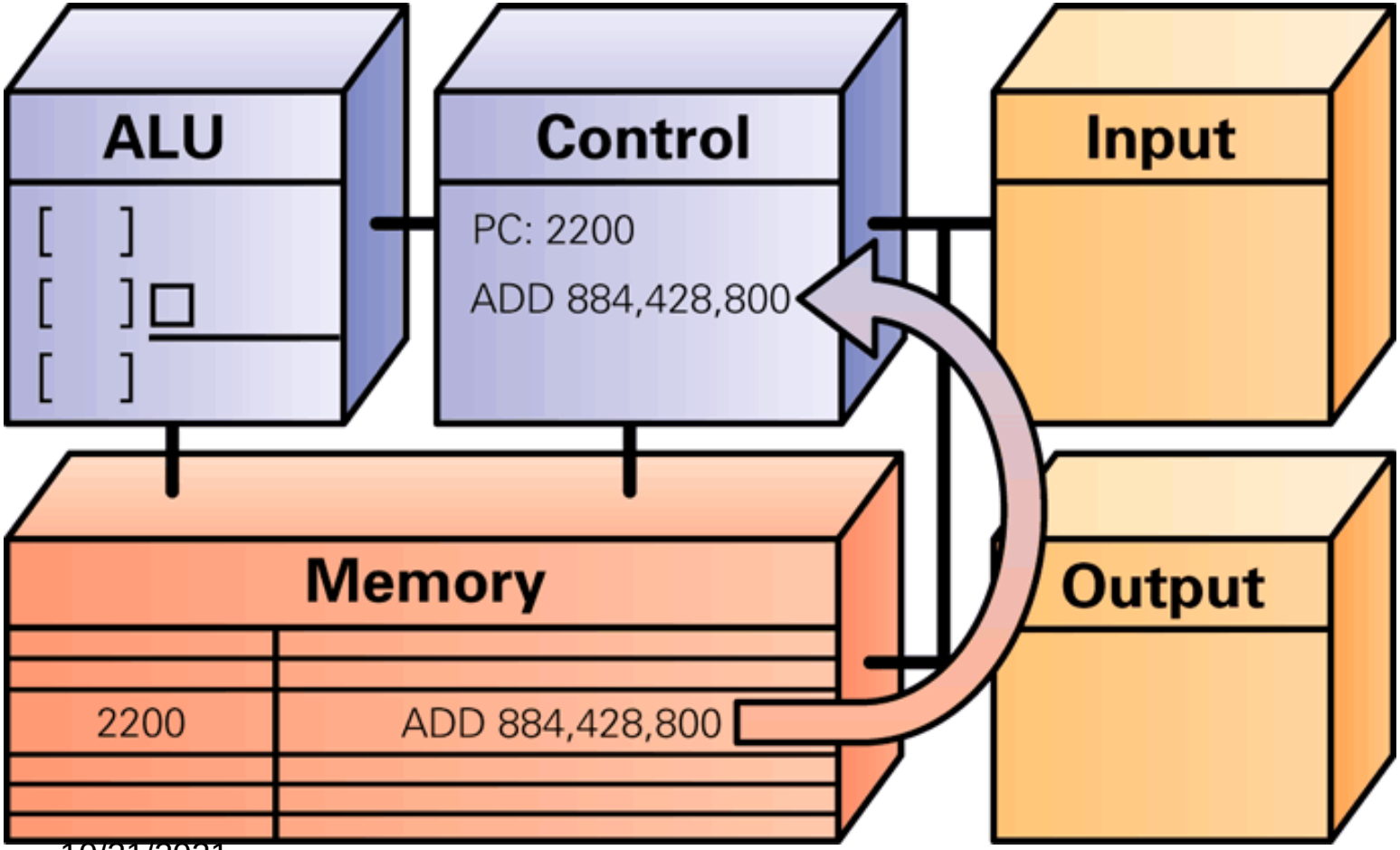
- This keeps track of what step needs to be executed next.
- It uses the address of the next instruction as its way of keeping track. This is called the program counter. (Here 'PC')
- Some instructions change the Program counter (branch and jump instructions)

Computer before executing an ADD instruction - again



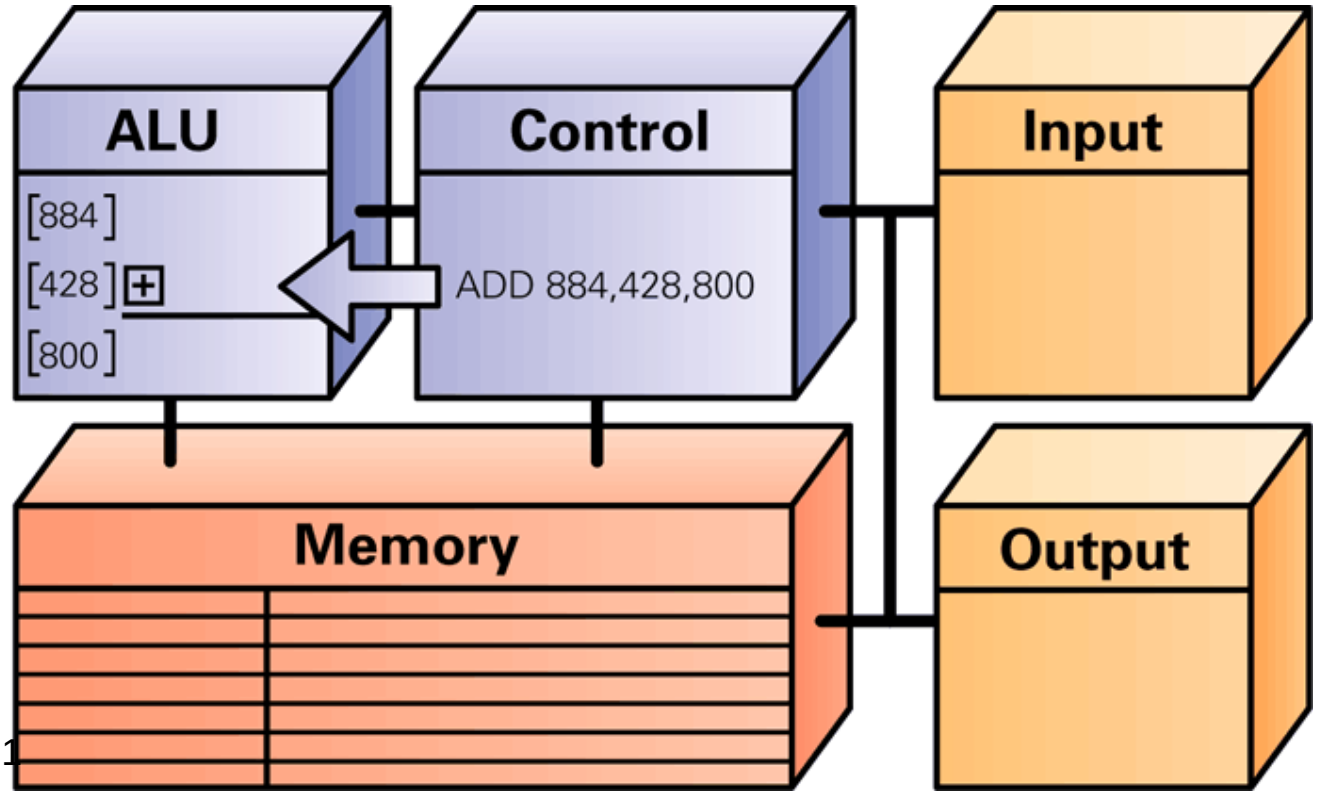
Instruction Fetch:

Move instruction from memory to the control unit.



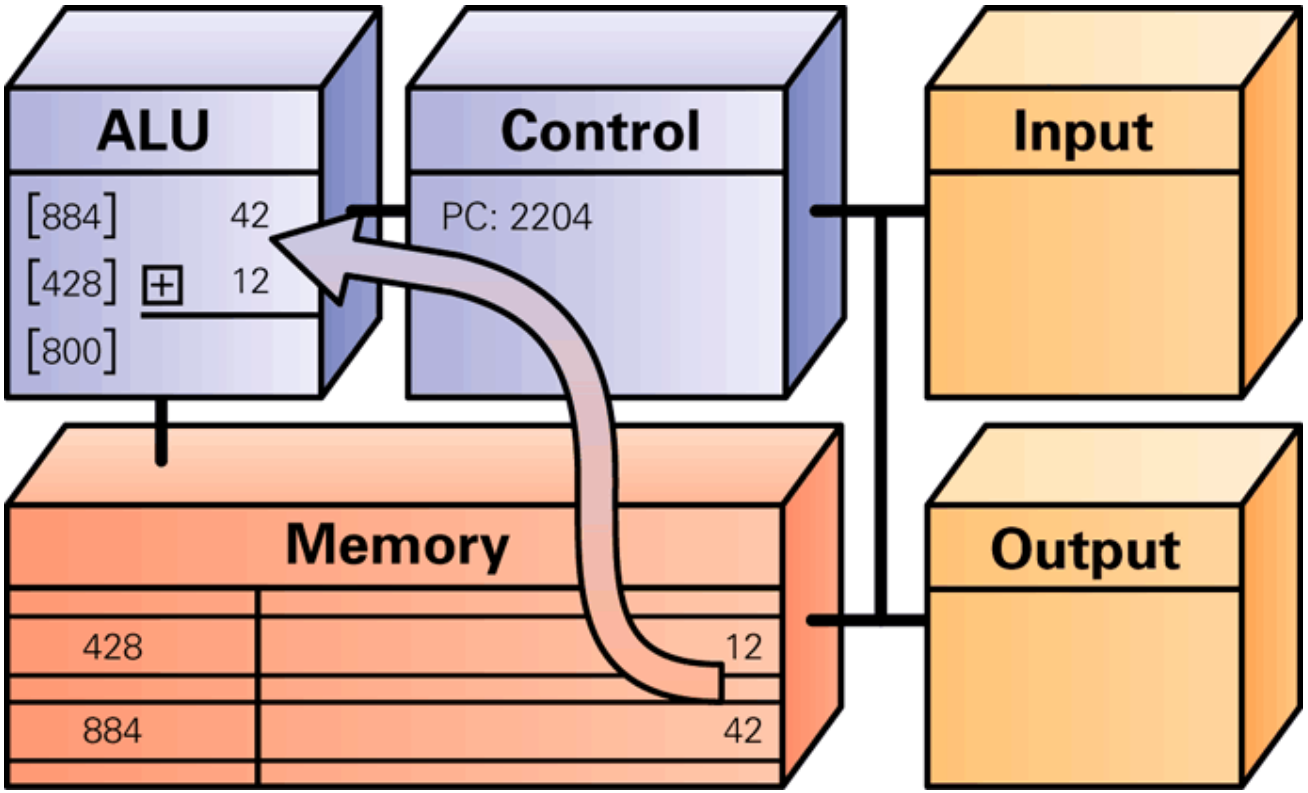
Instruction Decode:

Pull apart the instruction, set up the operation in the ALU, and compute the source and destination operand addresses.



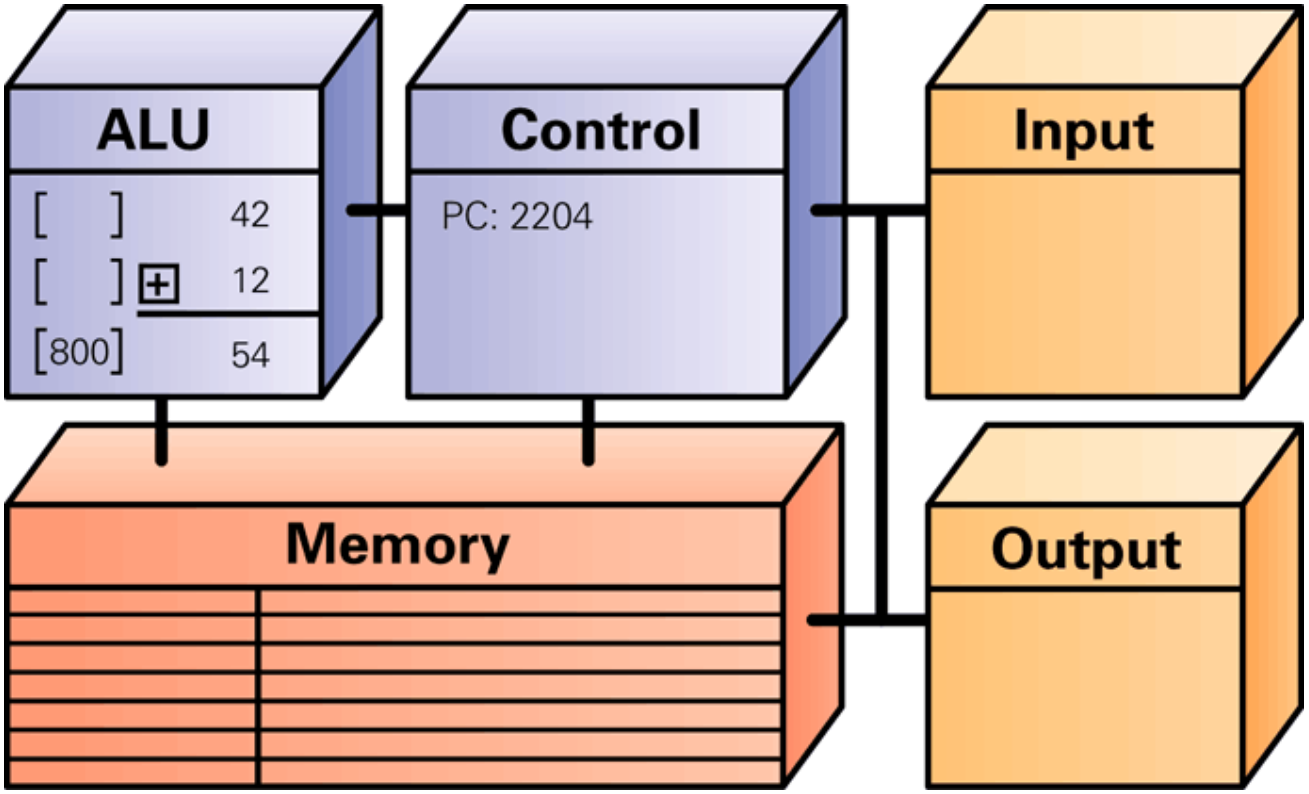
Data Fetch:

Move the operands from memory to the ALU.



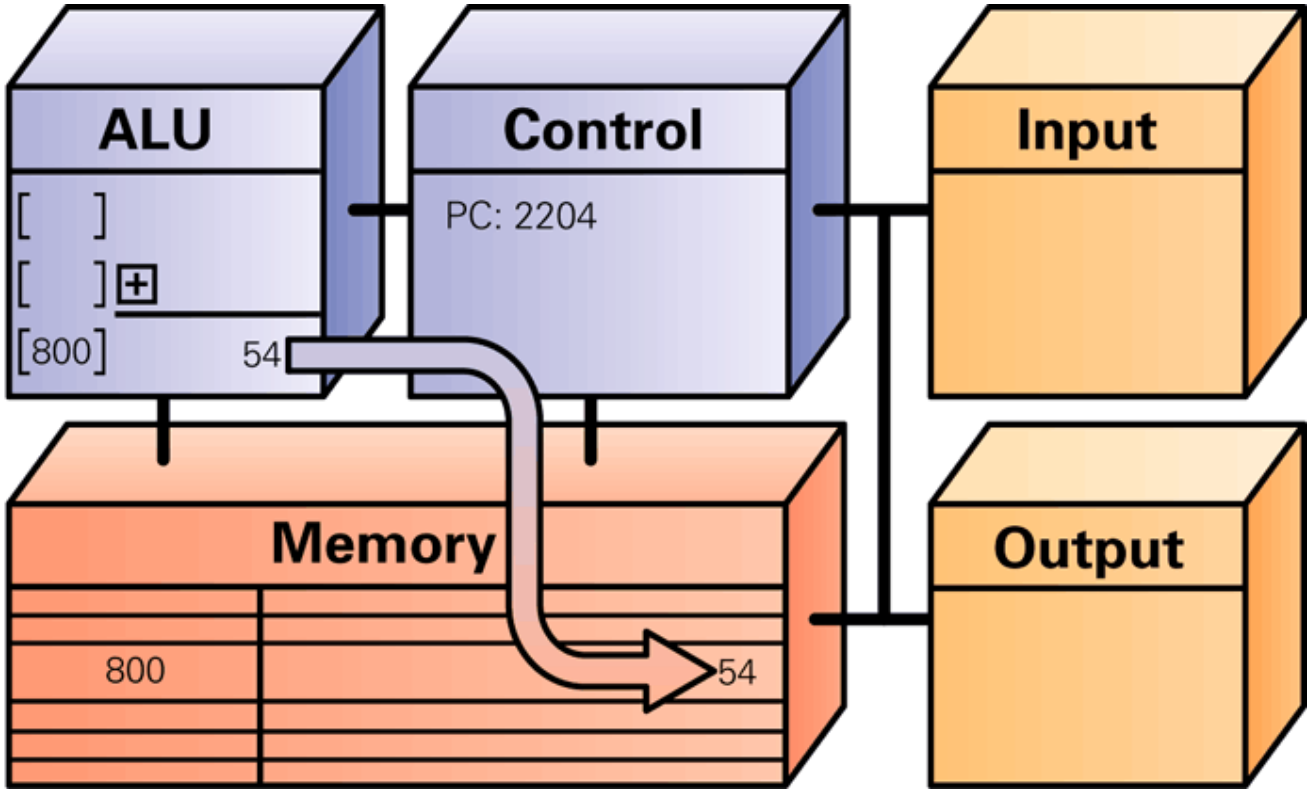
Execute:

Compute the result of the operation in the ALU.



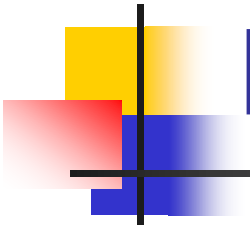
Result Return:

Store the result from the ALU into the memory at the destination address.



Registers

Register	Meaning	Purpose
PC	Program Counter	keeps track of where to find the next instruction so that a copy of the instruction can be placed in the CIR
CIR	Current Instruction Register	holds the instruction that is to be executed
MAR	Memory Address Register	hold the memory address that contains either the next piece of data or an instruction that is to be used
MDR	Memory Data Register	acts like a buffer and holds anything that is copied from the memory ready for the processor to use it.
IR	Index Register	Used for vector/array operations, and for modifying operand addresses during the running of a program.
ALU	Arithmetic Logic Unit	This is where data is processed. This involves arithmetic and logical operations.
CU	Control Unit	fetches instructions from memory, decodes them and synchronises the operations before sending signals to other parts of the computer
Accumulator	Holds results	



Drawbacks of Von Neumann

1. Both data and programs share the same memory space.

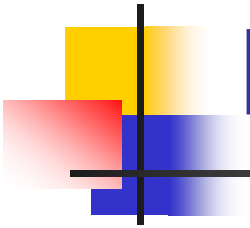
   This is a problem because it is quite easy for a poorly written or faulty piece of code to write data into an area holding other instructions, so trashing that program.

Drawbacks of Von Neumann

2. Every piece of data and instruction has to pass across the data bus in order to move from main memory into the CPU (and back again).
   - This is a problem because the data bus is a lot slower than the rate at which the CPU can carry out instructions.
   - This is called the 'Von Neumann bottleneck'.
   - If nothing were done, the CPU would spend most of its time waiting around for instructions.
   - A special kind of memory called a 'Cache' (pronounced 'cash') is used to tackle this problem

Drawbacks of Von Neumann

3. The rate at which data needs to be fetched and the rate at which instructions need to be fetched are often very different. And yet they share the same bottlenecked data bus.
   - To solve the problem idea of the Harvard Architecture is considered that to split the memory into two parts.
   - One part for data and another part for programs. Each part is accessed with a different bus.
   - This means the CPU can be fetching both data and instructions at the same time.

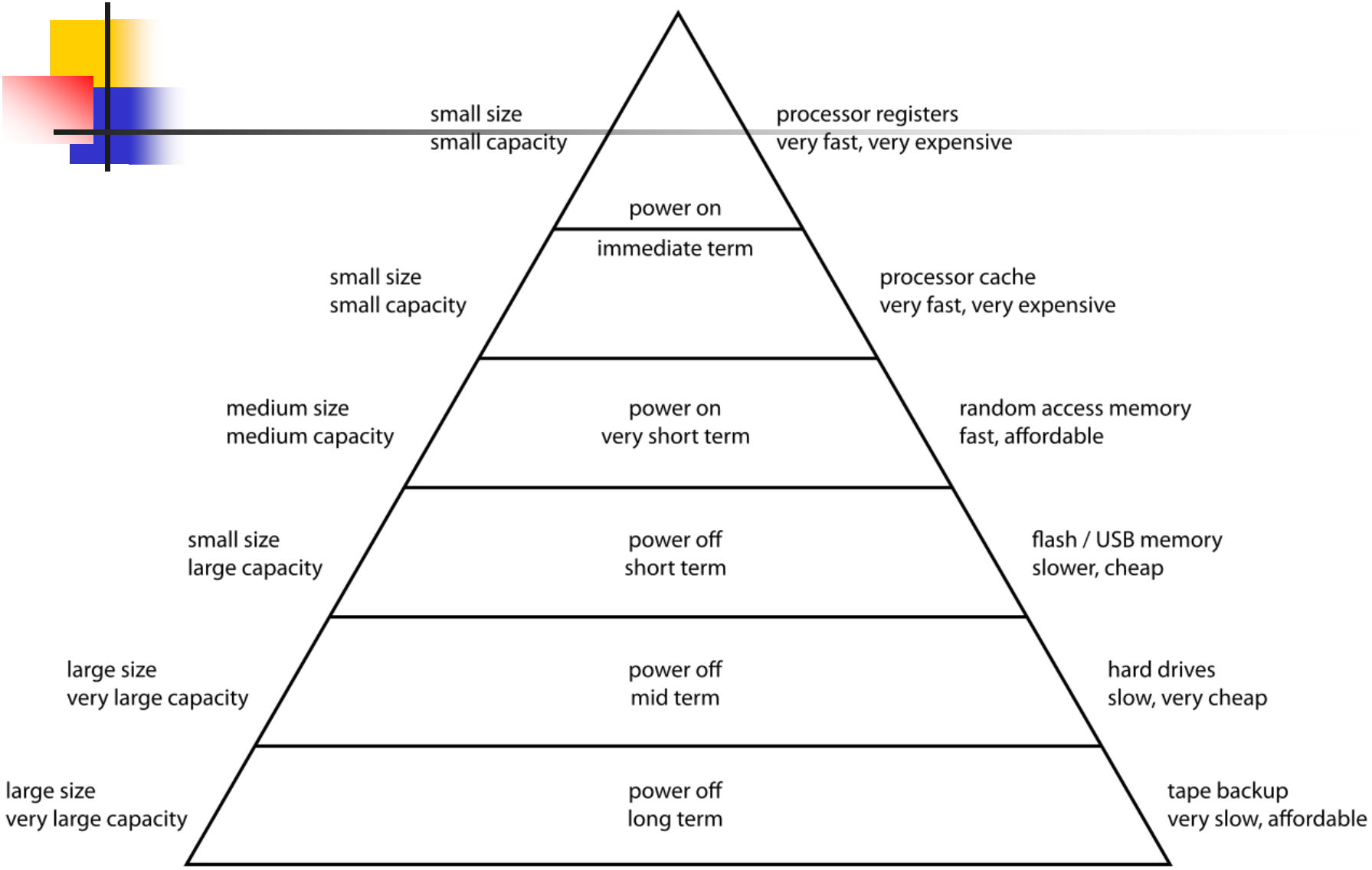


Exam questions

Describe the fetch execute cycle. [4 marks]

Draw a flow diagram or write in pseudo-code an algorithm for alphabetising a collection of CDs. [4 marks]

Computer Memory Hierarchy



Memory: ROM & RAM

- Main store (or computer memory) is divided into **Read Only Memory** (ROM) and **Random Access Memory** (RAM).

- **ROM** is memory that cannot be changed by a program or user. ROM retains its memory even after the computer is turned off. For example, ROM stores the instructions for the computer to start up when it is turned on again.

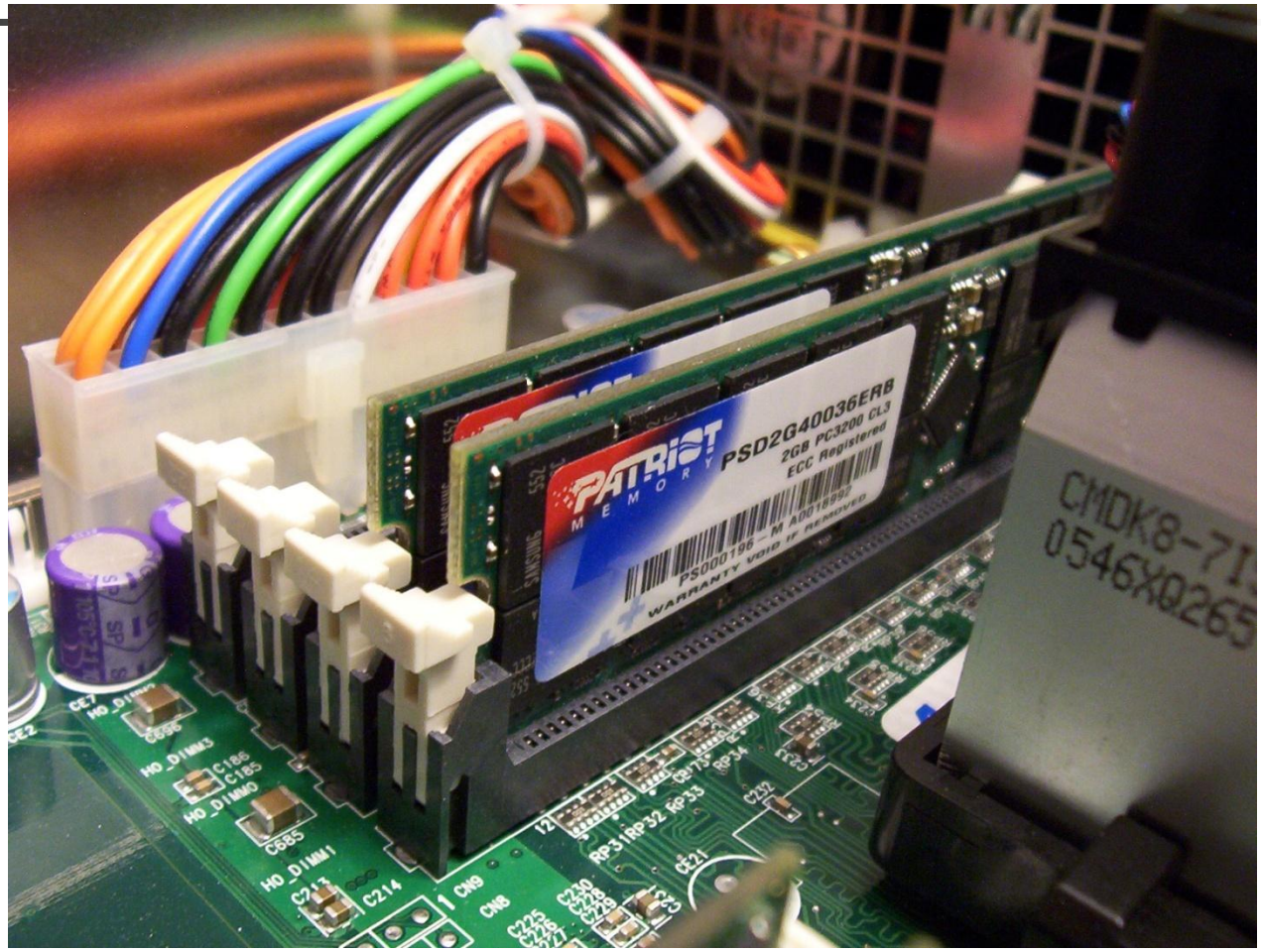
RAM

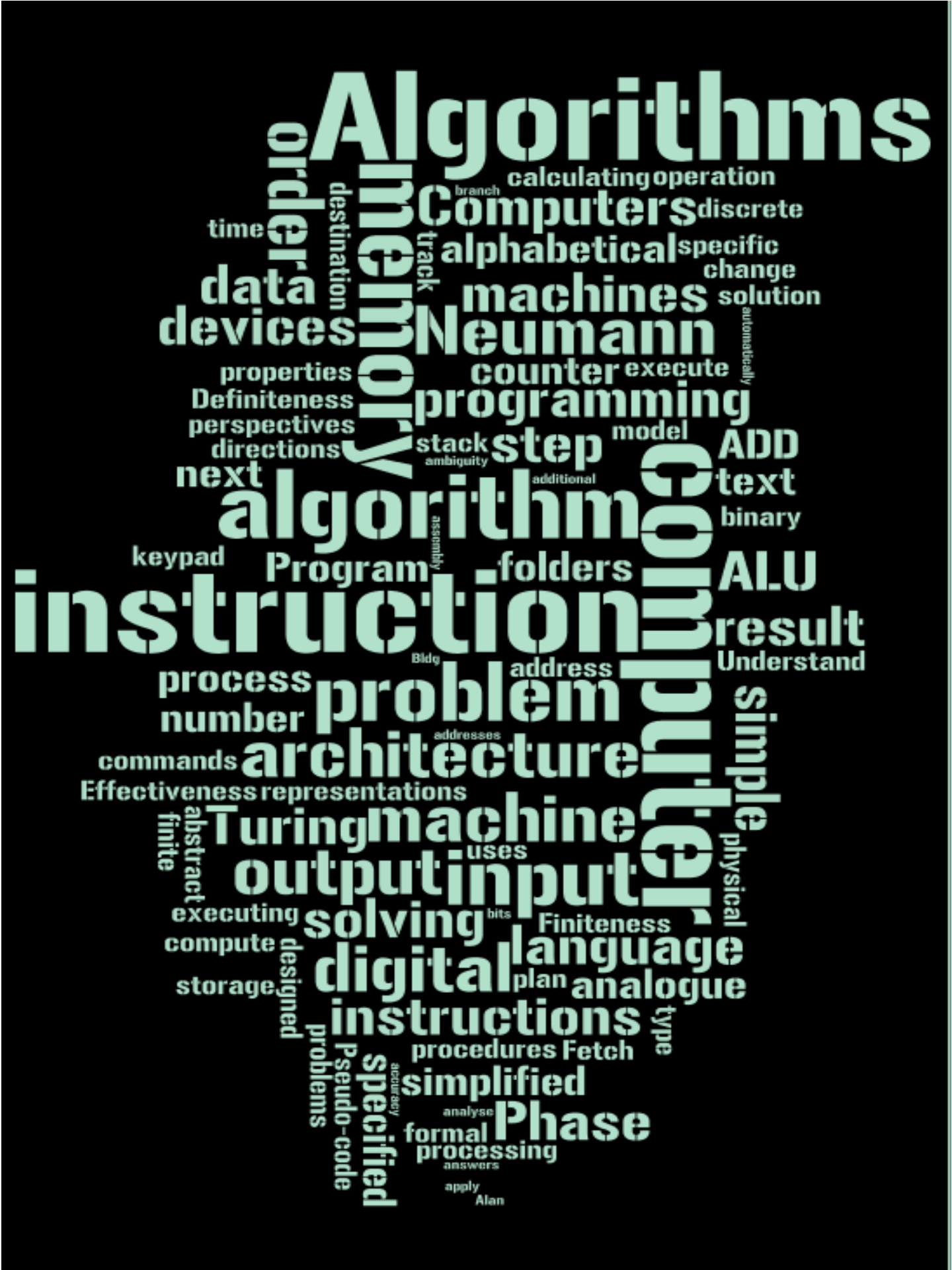
- **RAM** is a fast **temporary** type of memory in which programs, applications and data are stored. Examples of what's stored in RAM:
  - The operating system
  - Applications
  - The graphical user interface (GUI)

- If a computer loses power, all data stored in its RAM is lost.

RAM

RAM


PATRIOT
MEMORY
PSD2G40036ERB
2GB PC-3200 CL3
ECC Registered
WARRANTY VOID IF REMOVED
HD_DIMM3
HD_DIMM2
HD_DIMM1
HD_DIMM0
C685
C214
CN9
CN8


Algorithms
memory
order
Computers
calculating
operation
branch
discrete
time
destination
track
alphabetical
specific
change
data
machines
solution
devices
Neumann
automatically
properties
counter
execute
Definiteness
programming
perspectives
directions
stack
step
model
ADD
ambiguity
next
algorithm
additional
text
binary
keypad
Program
assembly
folders
ALU
instruction
computer
result
Bldg
Understand
process
problem
address
simple
number
addresses
architecture
commands
Effectiveness
representations
Turing
machine
abstract
finite
uses
physical
output
input
executing
solving
bits
Finiteness
compute
designed
digital
language
storage
plan
analogue
instructions
type
procedures
Fetch
problems
Pseudo-code
specified
simplified
accuracy
analyse
Phase
formal
processing
answers
apply
Alan