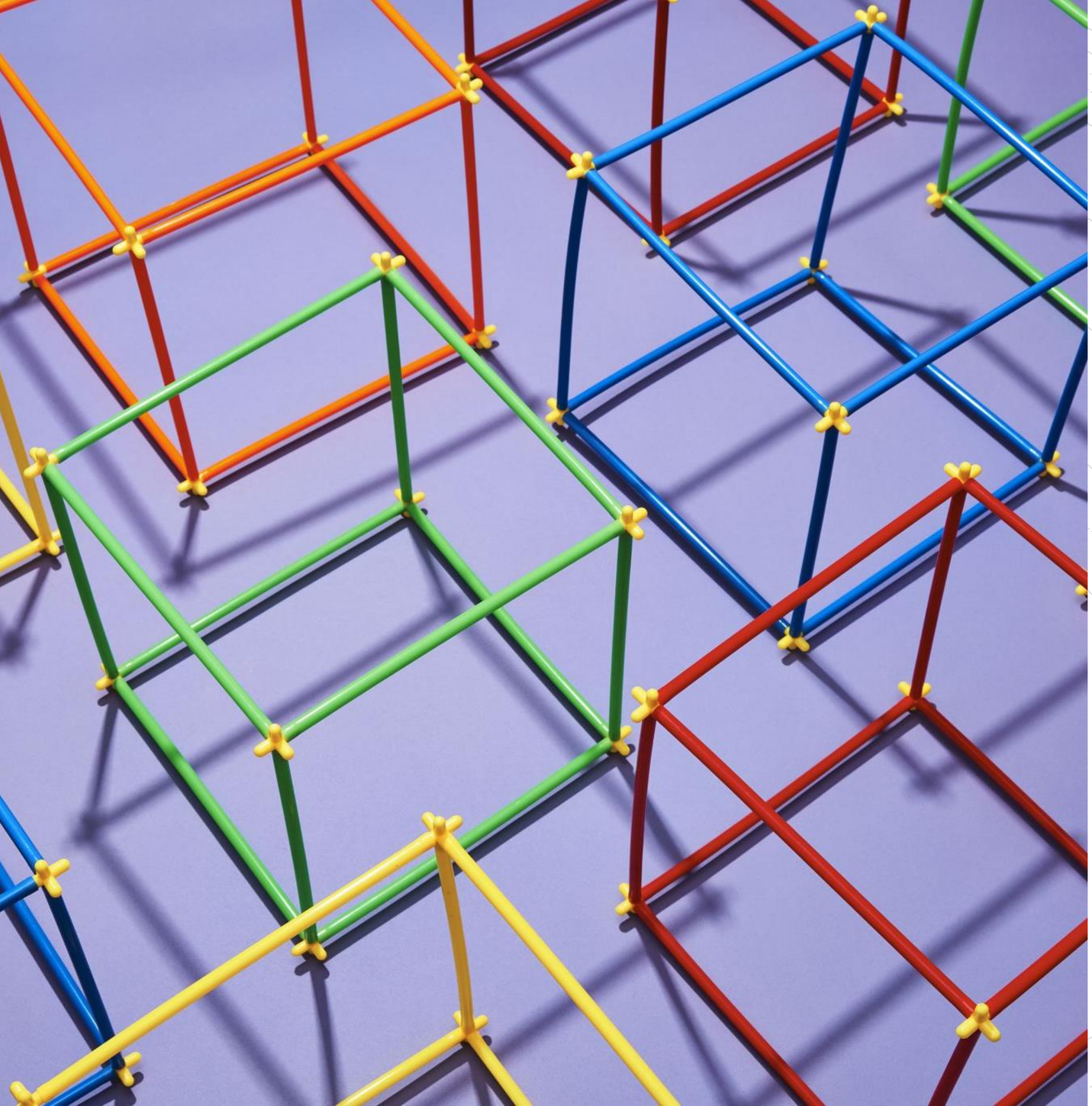
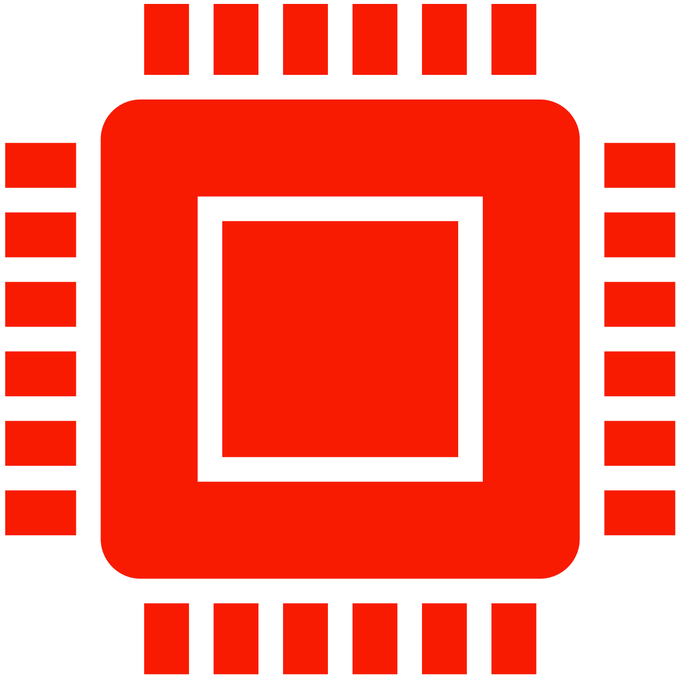
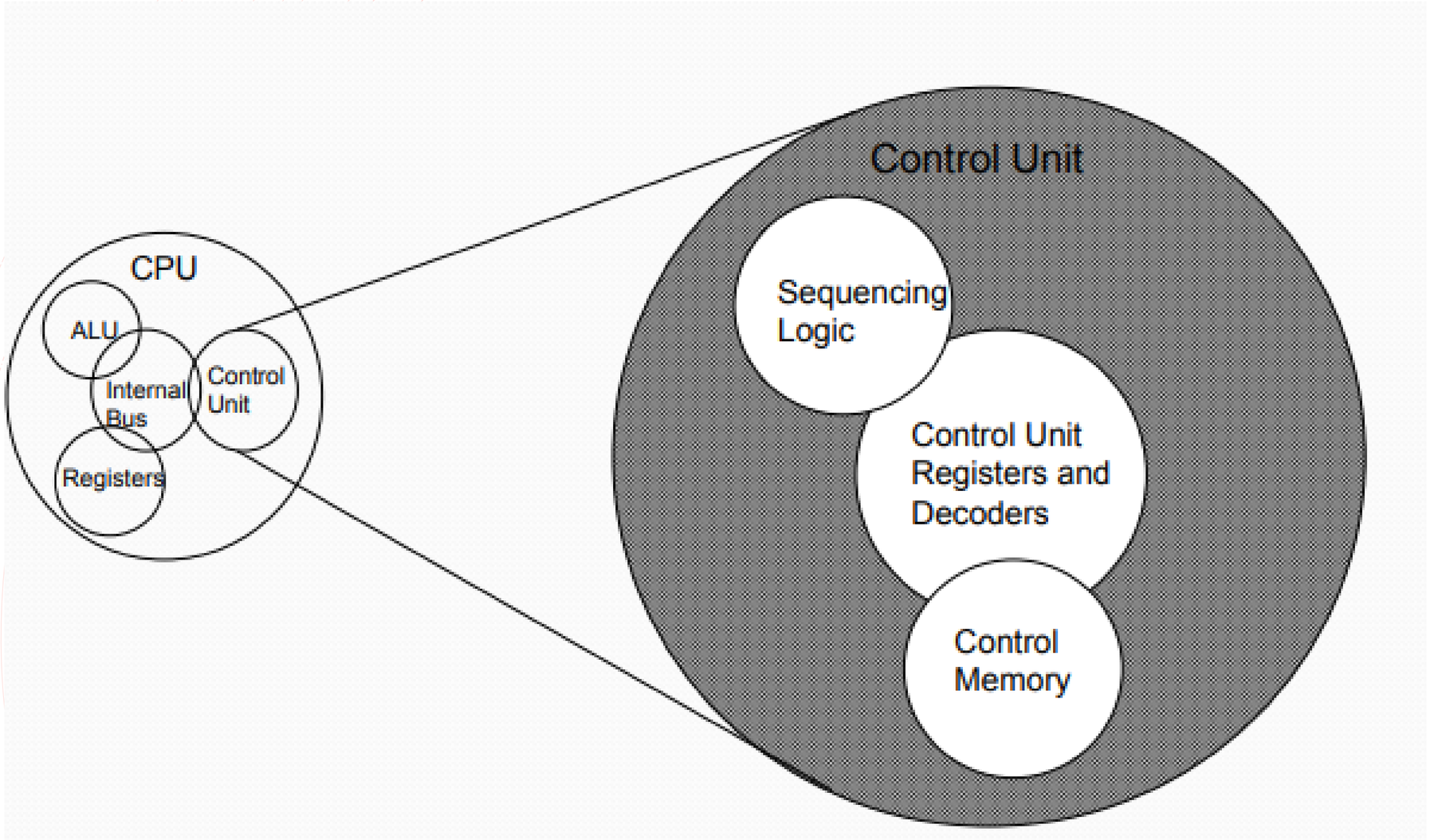


Essentials of Computer Organization and Architecture IN 2300

Chamalka Rajapaksha

CPU Design




CPU
ALU
Internal Bus
Control Unit
Registers
Control Unit
Sequencing Logic
Control Unit Registers and Decoders
Control Memory

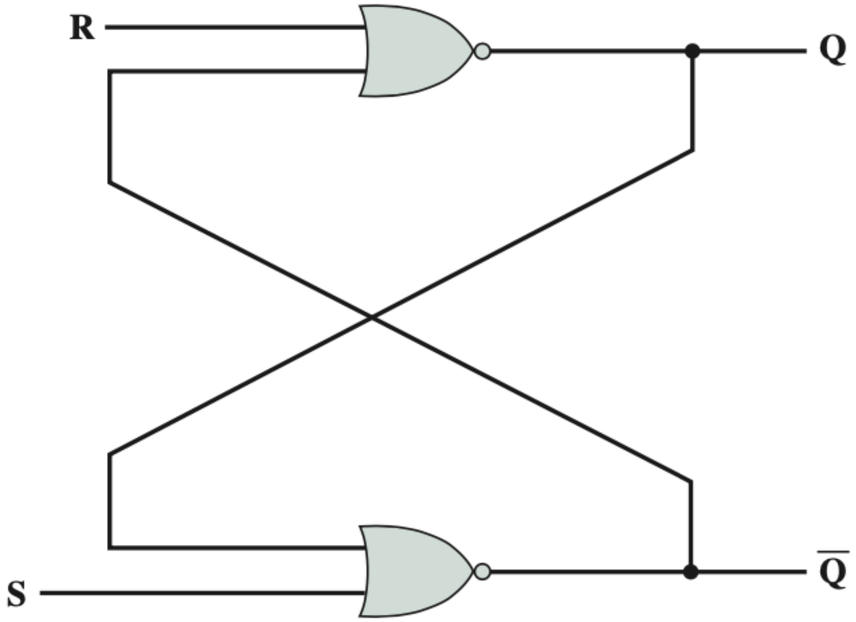
Sequential Logic

- Combinational circuits have only inputs and outputs. The output is a function of the input only.
- Sequential circuits have a notion of state. The output is a function of the input and the previous state.
- Commonly referred to as Finite state machine (FSM)
- An FSM distinguished from a combinational logic in that the past history inputs to the FSM influences its state and output.
- This is important for implementing memory circuits as well as control unit in a computer

Flip flop

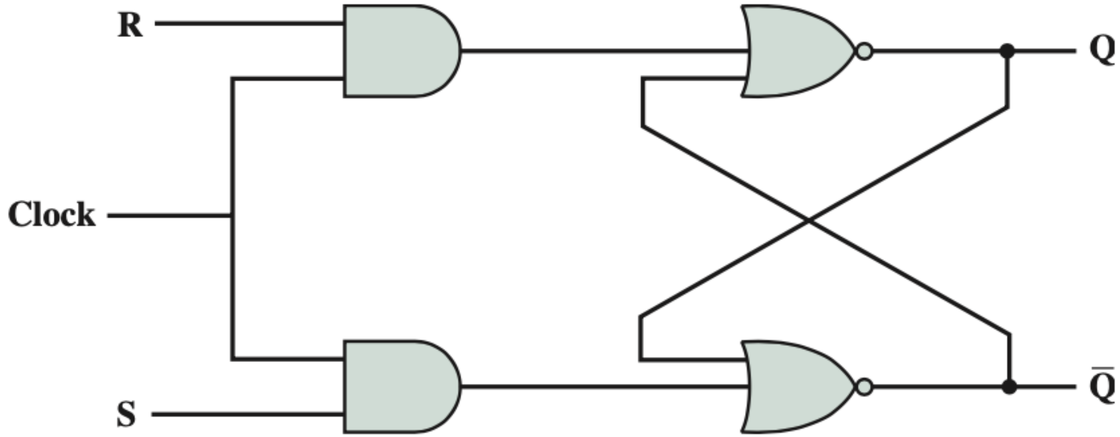
- Flip flop is the simplest form of sequential circuit.
- The basic circuit for storing data in digital machines.
- Eg: S-R latch,
- Two common characteristics
  - The flip- flop is a bistable device.
  - The flip- flop has two outputs (states).

S-R Latch



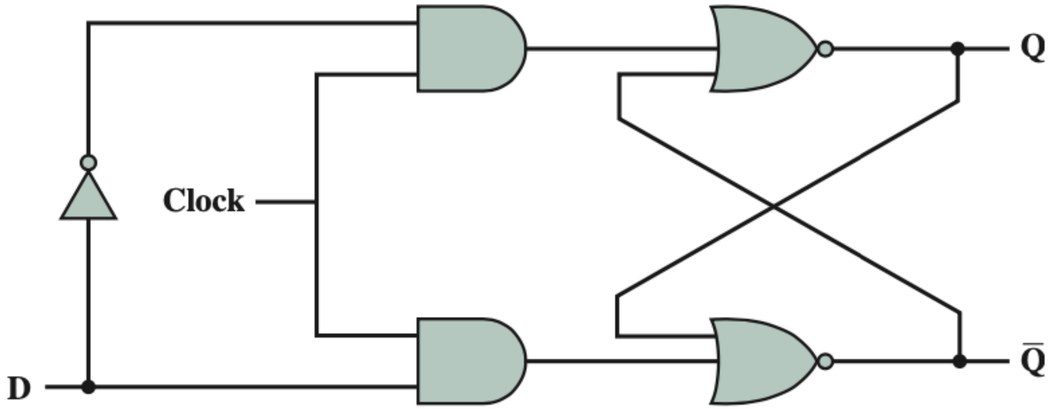
(b) Simplified Characteristic Table		
S	**R**	$\mathbf{Q}_{n+1}$
0	0	Q_n
0	1	0
1	0	1
1	1	—

Clocked S-R Flip Flop



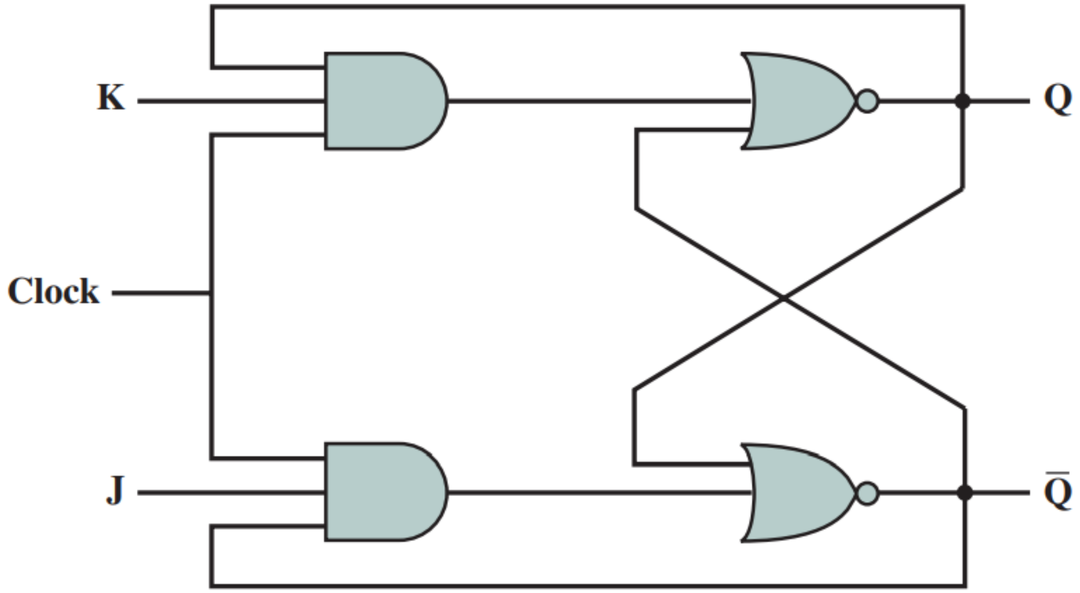
D Flip flop

- Sometimes referred to as data flip-flop – Store 1 bit of data
- Also known as delay flip-flop - because it delays a 0 or 1 applied to its input for a single clock pulse.
- The problem in the S-R latch has been addressed.



D	Q_{n+1}
0	0
1	1

J-K Flip flop



J	K	Q_{n+1}
0	0	Q_n
0	1	0
1	0	1
1	1	$\overline{Q_n}$

Registers

- One of the essential elements of the CPU
- Registers are formed from a group of flip flops arranged to hold and manipulate a data word using some common circuitry.

1. Parallel register - consists of a set of 1-bit memories that can be read or written simultaneously.

2. Shift register - accepts and/or transfers information serially. Used as interface to serial I/O devices, in ALU to perform logical shift and rotate function

Computer Word Structure

- The data and instruction words in computers tend to be organized systematically.
- Instruction word = data word = memory word (in most early cases)
- A special term used for an 8-bit word is called Byte.
  - 2^{10} Bytes = 1 KB, 2^{10} KB = 1 MB, 2^{10} MB = 1GB
- A number that identifies the location of a word in a memory is called address
  - Always used binary number, although hexadecimal used
  - Refers to a word location in memory

Von Neumann/Turing

- Stored Program concept
- Main memory storing programs and data
- ALU operating on binary data
- Control unit interpreting instructions from memory and executing
- Input and output equipment operated by control unit
- Princeton Institute for Advanced Studies (IAS)
- Completed 1952

The Stored Programme Concept

- Von Neumann's proposal was to store the program instructions right along with the data
- This may sound trivial, but it represented a profound paradigm shift
- The stored program concept was proposed about fifty years ago; to this day, it is the fundamental architecture that fuels computers.
- Think about how amazing that is, given the short shelf life of computer products and technologies…

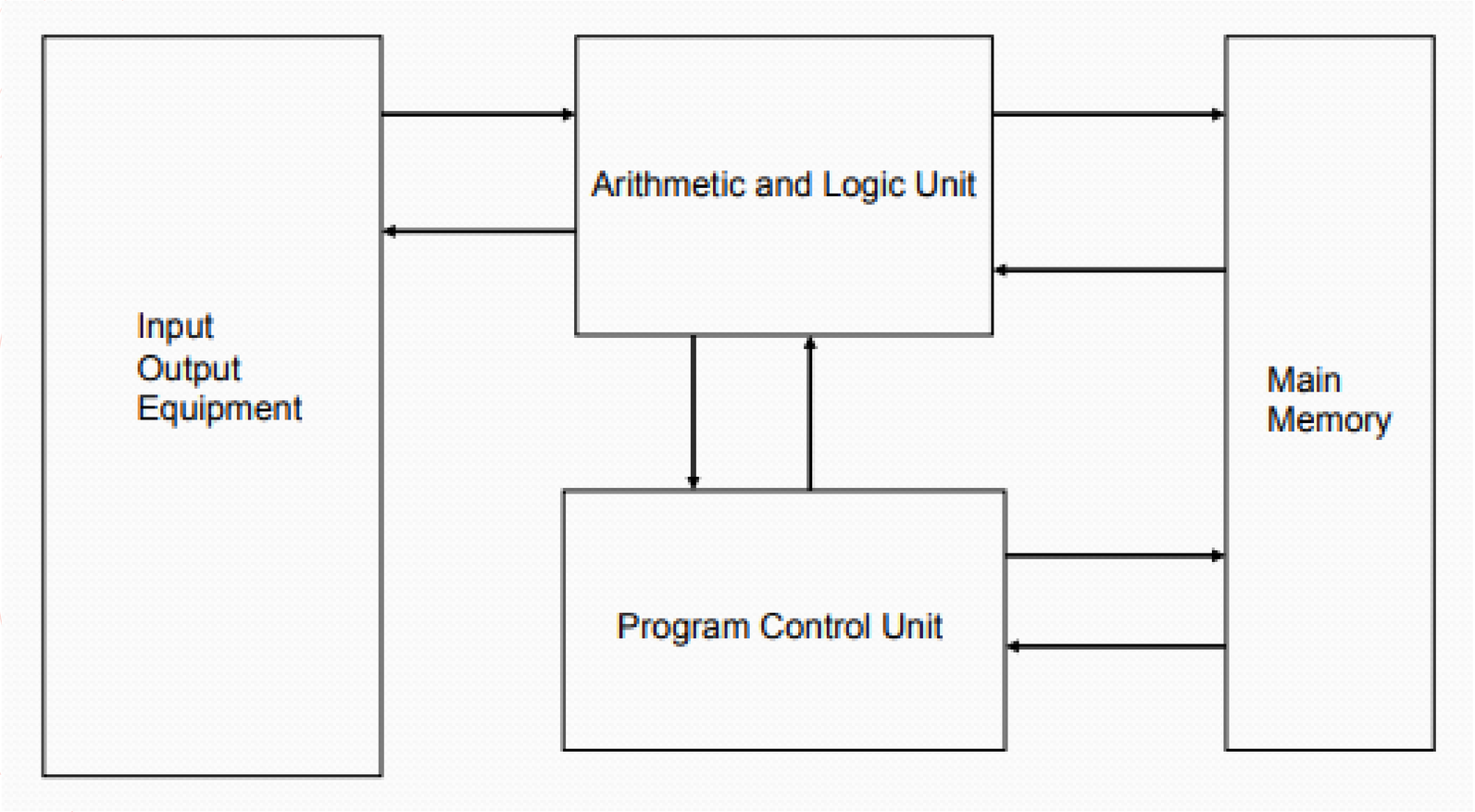
The Stored Program Concept and its Implications

- The Stored Program concept had several technical formations:
  - Four key sub-components operate together to make the stored program concept work
  - The process that moves information through the sub-components is called the “fetch execute” cycle
  - Unless otherwise indicated, program instructions are executed in sequential order
  - Easy to access and modify data and programs

Four Subcomponents

- There are four sub-components in von Neumann architecture:
  - Memory
  - Input/Output (called “IO”)
  - Arithmetic-Logic Unit
  - Control Unit
- While only 4 sub-components are called out, there is a 5th, key player in this operation: a bus, or wire, that connects the components together and over which data flows from one sub-component to another

Structure of Von Nuemann Machine



Von Neumann Bottleneck

- Shared bus between the program memory and data memory leads to the Von Neumann bottleneck
- Limited throughput (data transfer rate) between the CPU and memory compared to the amount of memory
- Limits the effective processing speed when the CPU is required to perform minimal processing on large amounts of data

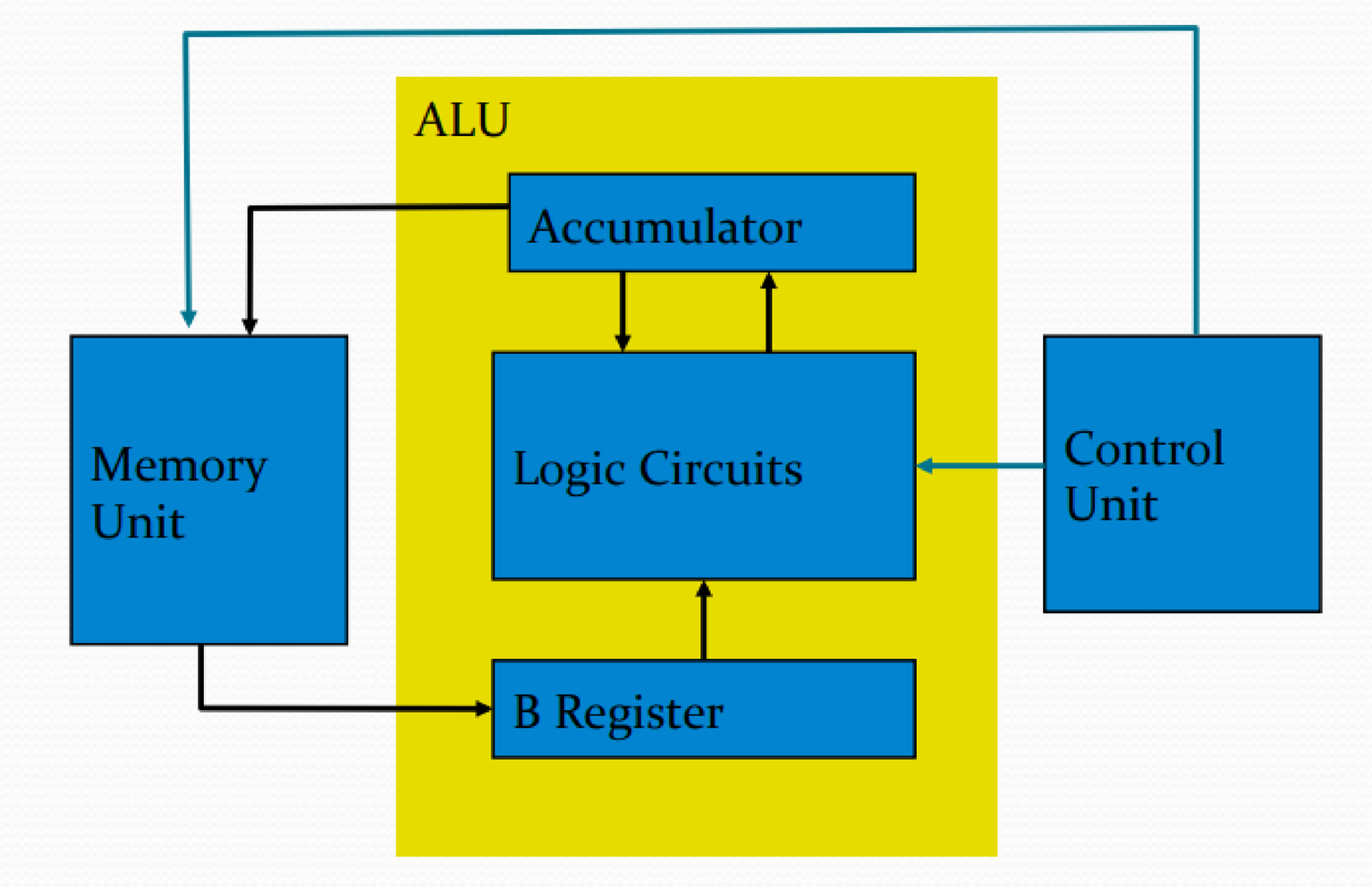
Harvard Architecture

- The name Harvard Architecture comes from the Harvard Mark I relay-based computer
- Harvard Architecture has physically separated signals and storage for program and data memory
- Possible to access program memory and data memory simultaneously
- Program memory is read-only and data memory is read-write
- Impossible for program contents to be modified by the program itself

ALU

- Perform arithmetic and logic operations on input data.
- Set of registers with accumulator register
- Registers store the results of all the arithmetic and logic operations

Functional Parts of ALU

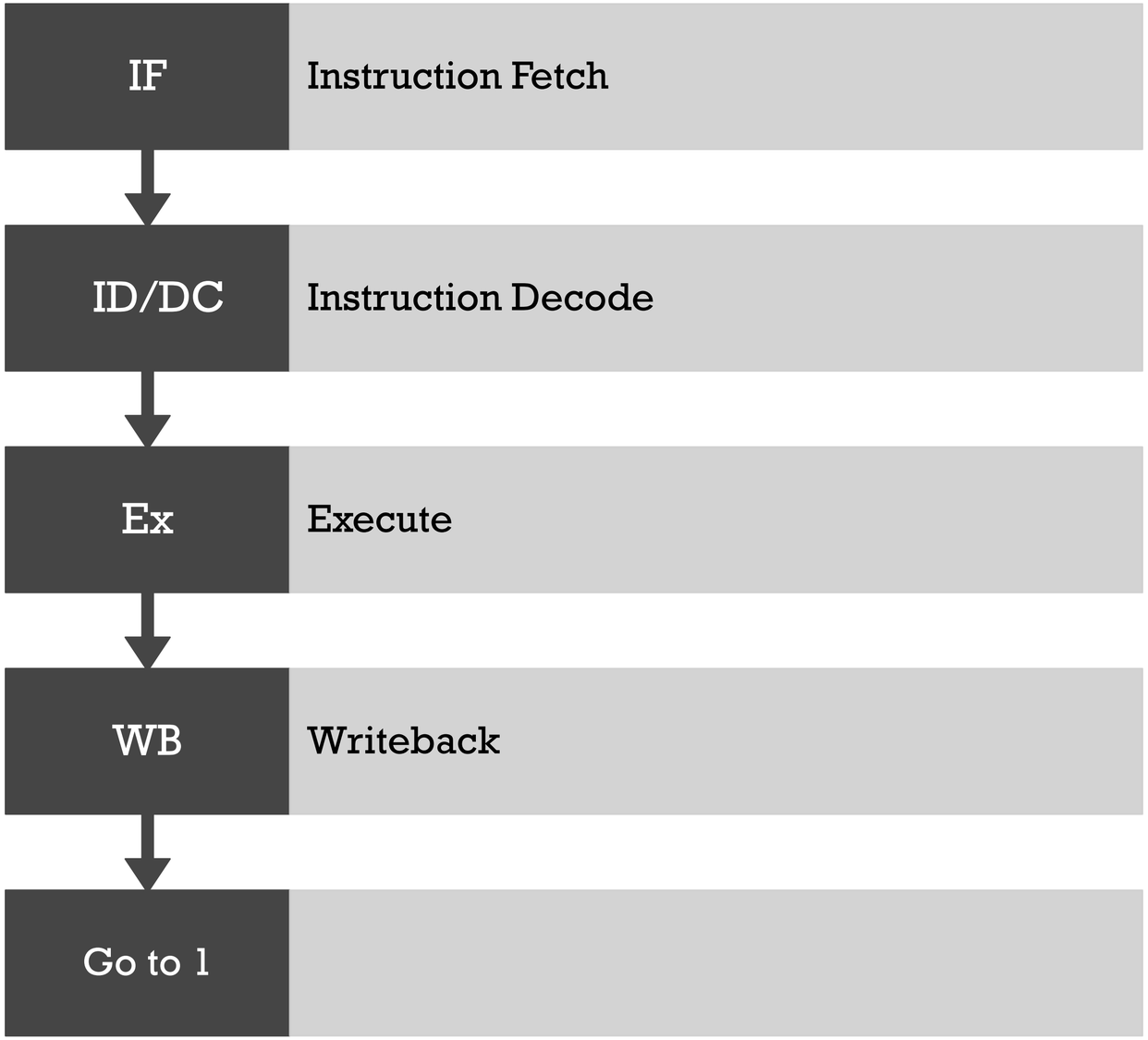


ALU Operation

- The control unit receive an instruction (from memory unit) specifying that a number stored in a particular memory location (address) is to be added to the number presently stored in the accumulator register
- The number to be added is transferred from memory to the B Register
- The number in the B register and the number in the accumulator register are added together in the logic circuits (upon command from the control unit)
- The result sum is then sent to the accumulator to be stored

Fetch/ Execute Cycle

- All computers can be summarized with just two basic components:
  - primary storage or memory
  - central processing unit or CPU
- CPU is the "brains" of the computer.
- Its function is to execute programs which are stored in memory
- This procedure is accomplished by the CPU fetching an instruction stored in memory, then executing the retrieved (fetched) instruction within the CPU before proceeding to fetch the next instruction from memory
- This process continues until they are told to stop


IF
Instruction Fetch
ID/DC
Instruction Decode
Ex
Execute
WB
Writeback
Go to 1

Instruction Cycle

- Extract the instruction from memory
- Calculate the address of the next instruction by advancing the PC
- Decode the opcode
- Calculate the address of operands (if any)
- Extract the operand from the memory
- Execute
- Calculate address of the result
- Store result in memory

MAR <- [PC]

PC 201

MAR 201

MBR

CIR

ACC

201	LOAD 206
202	ADD #205
203	STORE 205
204	HALT
205	203
206	1

MBR <- [Memory]$_{\text{MAR address}}$; PC <- [PC]+1

+1

PC 202

MAR 201

MBR LOAD 206

CIR

Address Bus

Data Bus

201	LOAD 206
202	ADD #205
203	STORE 205
204	HALT
205	203
206	1

ACC

CIR <- [MBR]

PC 202

MAR 201

MBR LOAD 206

CIR LOAD 206

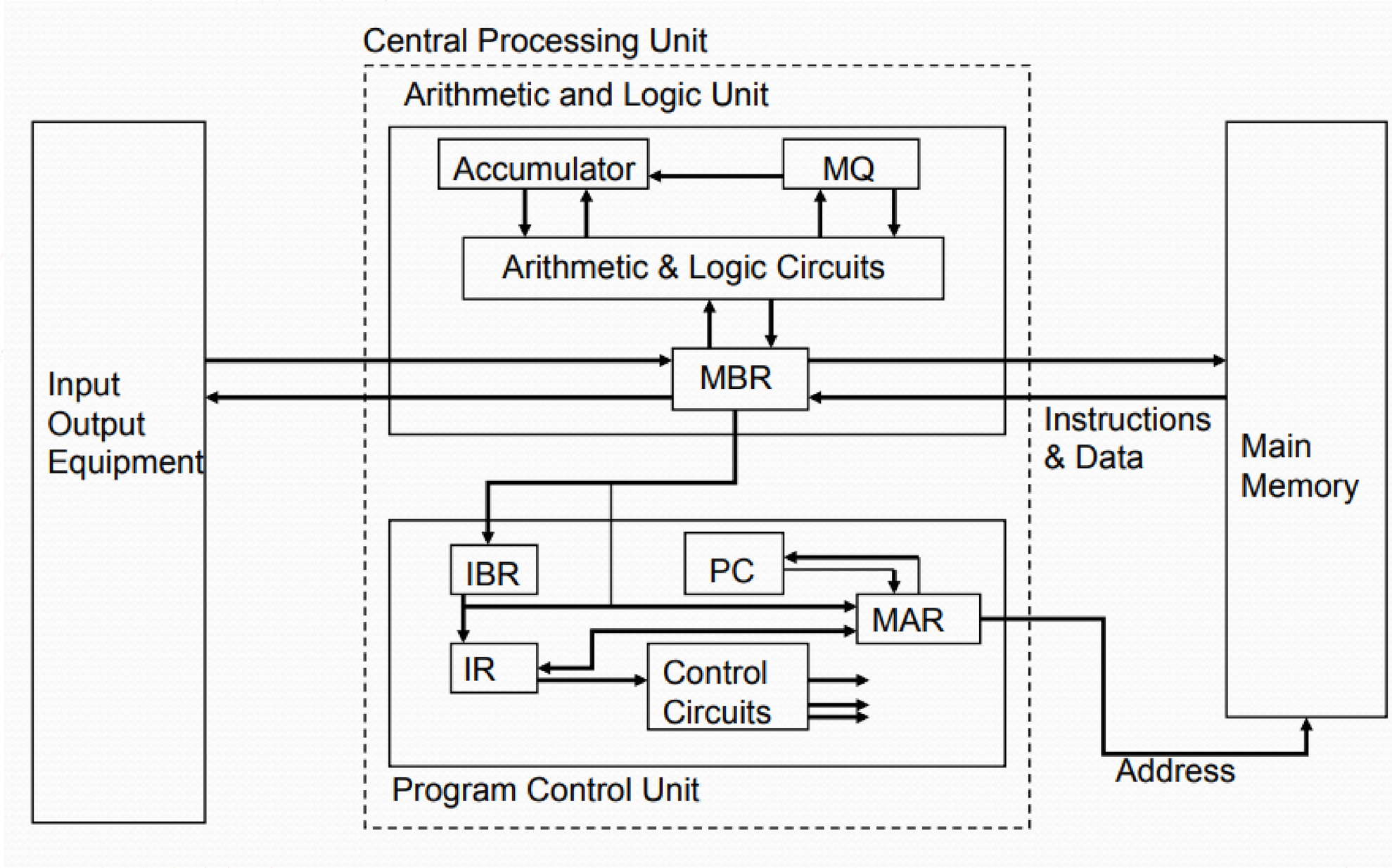
201	LOAD 206
202	ADD #205
203	STORE 205
204	HALT
205	203
206	1

ACC

IAS

- 1000 x 40 bit words
  - Binary number
  - 2 x 20 bit instructions
- Set of registers (storage in CPU)
  - Memory Buffer Register (MBR)
  - Memory Address Register (MAR)
  - Instruction Register (IR)
  - Instruction Buffer Register (IBR)
  - Program Counter (PC)
  - Accumulator (ACC)
  - Multiplier Quotient (MQ)

Structure of IAS



How IAS Operates?

Computer Organization – Motherboard/ chipset/ Busses

- The motherboard is the main circuit board inside a computer, containing the central processing unit (CPU), the bus, memory sockets, expansion slots, and more.
- The associated chips which help in overall system integration
- And which are fabricated on to the motherboard are collectively called the Chipset.
- Buses are the medium through which all the devices communicate.
  - Eg. PCI, Microchannel, ISA, EISA etc.

Computer Organization - CPU

- The part of a computer which controls all the other parts.
- Generally called the “processor”
- Composed of
  - Control Unit (CU)
  - Arithmetic and Logic Unit (ALU)
  - Registers and associated memory.

CPU (Cont)

- Control Unit (CU)
  - The control unit fetches instructions from memory and decodes them to produce signals which control the other parts of the computer.
  - This may cause it to transfer data between memory and ALU or to activate peripherals to perform input or output

CPU (Cont)

- Arithmetic and Logic Unit (ALU)
  - The combinatorial part of the central processing unit which performs operations such as addition, subtraction and multiplication of integers and bitwise AND, OR, NOT and other Boolean operations.
  - The width in bits of the data which the ALU handles is usually the same as that quoted for the processor as a whole whereas its external buses may be narrower.

CPU (Cont)

- Registers and Associated Memory
  - Registers are like internal buffers for the CPU. The CPU uses the registers to decode instructions and for using them for computing the final output (ALU).
  - Additionally the CPU may make use of internal memory (mostly SRAM as they are the fastest) for caching code and data for frequently accessed operations.

CPU - Instructions

- These are basically control commands for the control unit of the CPU which dictate the task to be performed (E.g. add, subtract, move, shift, interrupt etc.).
- The instructions can be considered as an interface to the features offered by the CPU.
- Instructions are composed primarily of "opcode" which specifies what the instruction is and the "operands" which specify the data on which that instruction needs to operate on.

CPU – Machine Programmes

- These are full length programs composed of several instructions which perform the real world task at hand.
- Every CPU has a Program Counter (PC) register which is used to keep track of the current instruction being executed.

CPU (Cont)

- Computers are classified into two broad categories depending upon the complexity of the instructions supported by the CPUs.
- RISC (Reduced Instruction Set Computer)
  - They have a very small subset of instructions to offer.
  - The format of their instructions are uniform.
  - Control Unit design is very simple.
  - They do not have flexibility.

CPU (Cont)

- CISC (Complex Instruction Set Computers)
  - They have a huge set of instructions at offer.
  - The format of their instructions are not uniform.
  - The control unit design of CISC CPUs are overly complex.
  - Since they can support any arbitrary instruction, they are flexible

CPU – Control Unit

- Control Unit (CU) is the heart of the CPU.
- Every instruction that the CPU supports has to be decoded by the CU and executed appropriately.
- Every instruction has a sequence of microinstructions behind it that carries out the operation of that instruction.

Control Unit - Designs

- Hardwired (Conventional)
  - used in high-performance and RISC computers
  - As the name implies, the microinstructions are hard-wired into the computer.
  - This is much faster but much less flexible.
  - Typically a grid/array is used to establish a sequence of connections to construct the appropriate circuitry.

Control Unit – Designs (Cont)

- Microprogrammed
  - The instructions are programmable.
  - i.e. the computer architect is able to program a particular instruction using micro-instructions and download the entire sequence of micro-instructions into the CU.
  - Slow, Additional time is spent decoding and executing microinstructions.
  - Very flexible.

Control Unit – Hardwired vs Microprogrammed

- Hardwired
  - Composed of combinatorial and sequential circuits that generates complete timing that corresponds with execution of each instruction.
  - Time consuming and expensive to design.
  - Difficult to modify.
  - but fast !

Control Unit – Hardwired vs Microprogrammed

- Microprogrammed
  - Design is simpler.
  - Problem of timing each microinstruction is broken down. Microinstruction cycle handles timing in a simple and systematic way
  - Easier to modify, meaning really flexible.
  - Slower than hardwired control since you have additional time spent in decoding microinstructions and searching them.

CPU – Instruction Cycle

- Operation of a computer consists of a sequence of instruction cycles.
- An instruction cycle is the time needed to execute one complete machine instruction.
- Instruction cycle can be subdivided into –
  - Fetch - retrieving the instruction
  - Indirect – retrieving indirect operands (if any)
  - Execute – executing the instruction.
  - Interrupt – Handle any exception (if any)

Instruction Cycle (Cont)

- Most important is the Fetch-Execute Cycle.
- It consists of
  - Extract the instruction from memory
  - Calculate the address of the next instruction by advancing the PC (Program Counter)
  - Decode the opcode
  - Calculate the address of the operands (if any)
  - Extract the operand from memory
  - Execute
  - Calculate address of the result
  - Store result in memory

Control Unit - Microprogramming

- Each step of an instruction can be further broken down into micro operations as discussed.
- These micro operations are accomplished using what are called micro instructions.
- Therefore, every instruction can be said to have a corresponding micro program which accomplishes the task of that instruction.

Microprogramming (Cont)

- Basic components of Microprogramming are
- Instruction Register (IR) - stores the current instruction.
- Control Store - repository for microprograms. A form of memory.
- Sequencer – a microprogram control unit. A control unit within another.
- Microprogram Program Counter - similar to a Program Counter

Microprogramming (Cont)

- Instruction Register stores current machine language instruction from program in main memory.
- Each machine instruction is represented as a series of microinstructions, in the Control Store.
- Micro Program Counter points to the appropriate location of the current microinstruction in the Control Store.
- Executing a machine instruction requires finding the microinstructions and executing them until the microinstruction has completed.

Microprogramming (Ordinary Operation)

- Sequencer causes address of current machine instruction to be placed in micro Program Counter
- It may also clear the microinstruction buffer.
- Sequencer initiates control-store read and transfers microinstruction to microinstruction buffer
- Micro instruction decoder decodes microinstruction into micro-orders and issues those orders over control lines
- If a non-branching microinstruction, then micro Program Counter is incremented, otherwise new address in control store is calculated and placed in micro Program Counter

Microprogramming (Organization of Microprograms)

- Each Machine Language operation is actually a sequence of microinstructions
- Each of these microprograms is placed in the control store.
- Included are the microprograms for instruction fetch, interrupt initiation and other routines separates from machine language instructions.

Microprogramming (Branching / Non-Branching)

- Branching microinstructions hold the branch address inside of the microinstruction itself.
- Branching occurs only within the Control Store (as opposed to a machine language branch to another location in main memory)
- Special bit designates whether a microinstruction is a branch or non-branch.
- Internal Address Bus is used to transfer new control store location
- Nonbranching instructions require that the micro Program Counter be incremented

Micro Instruction Formats

- Microinstructions are simply a list of control orders for the various buses and gates in the computer
- Horizontal Control
  - each microinstruction is a series of bits, each of which represents a single control line
- Vertical Control
  - groups of bits in the microinstruction represent commands. This requires decoding or demultiplexing before the control commands can be issued
- Horizontal is faster but requires greater lengthen microinstructions.

Computer Organization - Memory

- memory are internal storage areas in a system.
- Memory operations rely on these –
  - MAR (Memory Address Register) – Stores the address of the memory location that is to be accessed
  - MDR (Memory Data Register) – Stores the data that is either written to or read from memory.
  - MBR (Memory Buffer Register) – Buffer register for temporary storage
  - MCRs (Memory Control Registers) – To setup access to the memory.

Input/ Output System

- There is a need to communicate with components and with the external world. Each of the communication endpoints is called a "port".
- I/O System comprises of the set of all I/O devices in the computer system including physical devices and I/ O interface devices.
- In the early days of computers, I/O devices were limited to line printer, punch card reader.
- Today there are numerous types of I/O devices (terminal, mouse, scanner, etc…)
- CPU-controlled I/O
  - the CPU would interrupt its current process to directly handle

Questions?