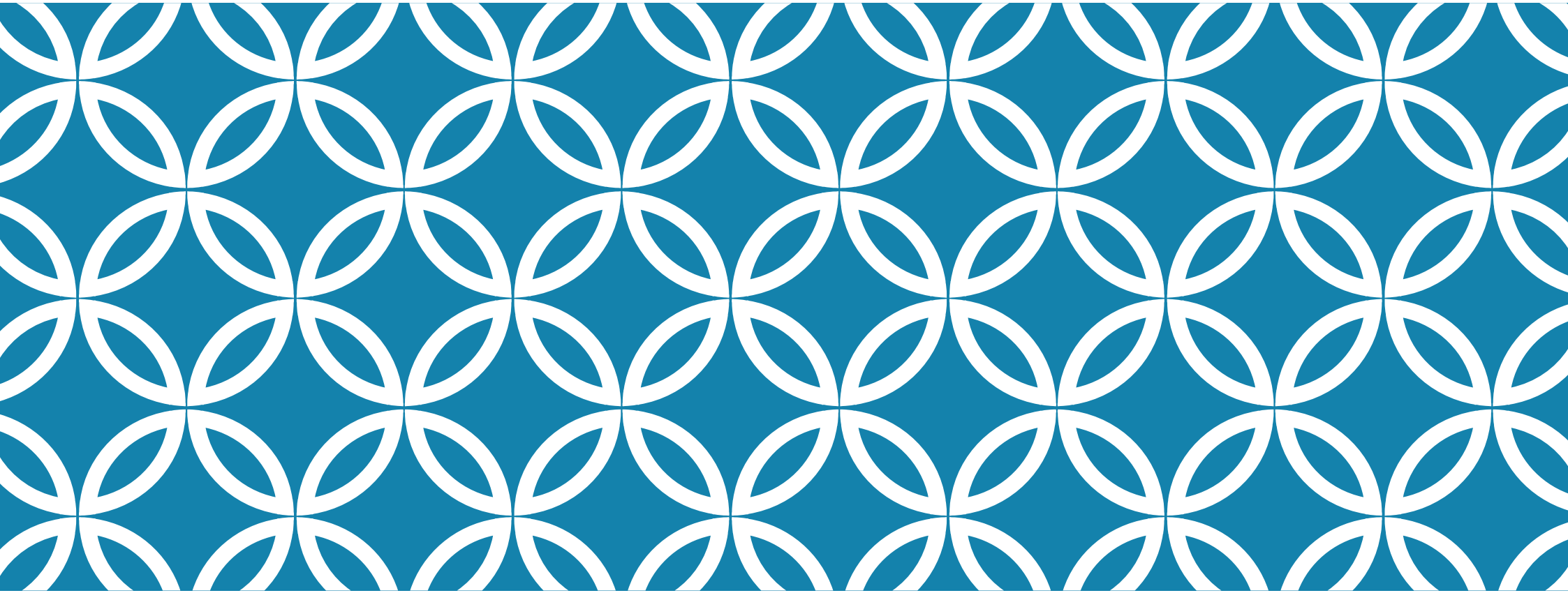
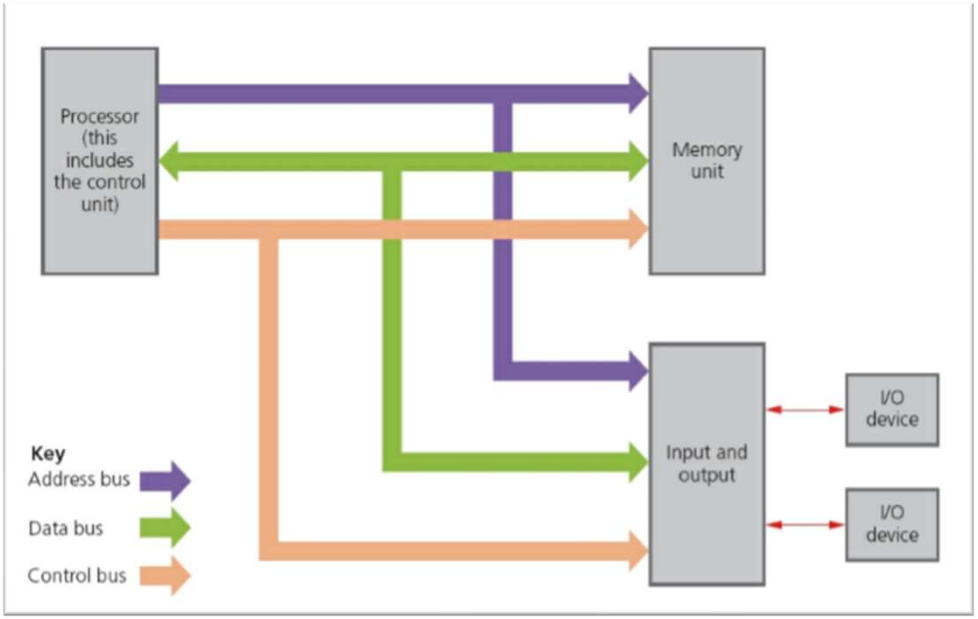


COMPUTER ARCHITECTURE

Chapter 3.1

3.1 COMPUTER ARCHITECTURE

- Von Neumann architecture
  - He developed the concept of the “stored program computer”
- Concepts of Von Neumann architecture
  - the concept of a central processing unit (CPU or processor)
  - the CPU was able to access the memory directly
  - computer memories could store programs as well as data
  - stored programs were made up of instructions which could be executed in sequential order



3.1 COMPUTER ARCHITECTURE

- Memory Unit
  - It is RAM
  - It is also known as “Immediate Access Store (IAS)”
  - It is used to stored the programs that is running
  - It contains instructions and data
  - *** Address is not information that is stored in RAM. It is location where instructions and data are stored

Address	Contents
1111 0000	0111 0010
1111 0001	0101 1011
1111 0010	1101 1101
1111 0011	0111 1011
1111 1100	1110 1010
1111 1101	1001 0101
1111 1110	1000 0010
1111 1111	0101 0101

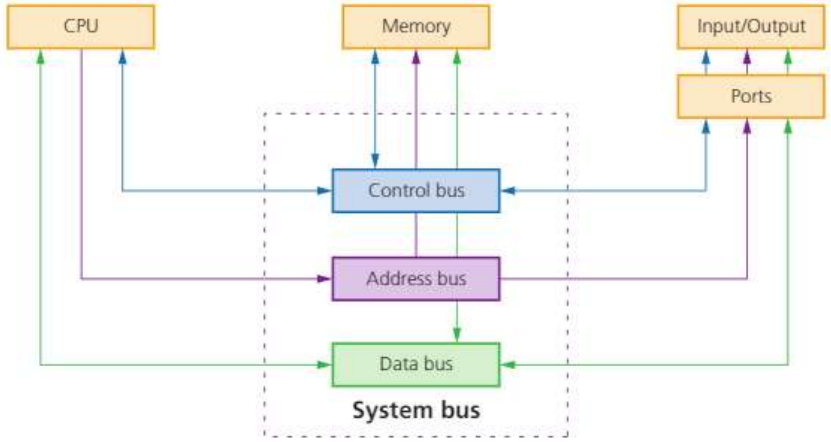
3.1 COMPUTER ARCHITECTURE

- Instructions and data in RAM are opcode and operand
- Opcode and Operand in this table are Assembly language which will be studied in chapter software
- Opcode is an instruction
- Operand can be either address or data

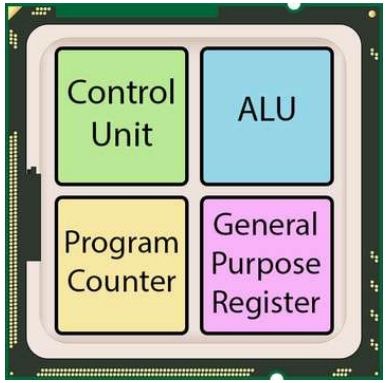
Instruction			Explanation
Label	Op code	Operand	
Data movement instructions			
	LDM	#n	Immediate addressing. Load the number *n* to ACC
	LDD	<address>	Direct addressing. Load the contents of the location at the given address to ACC
	LDI	<address>	Indirect addressing. The address to be used is at the given address. Load the contents of this second address to ACC
	LDX	<address>	Indexed addressing. Form the address from <address> + the contents of the index register. Copy the contents of this calculated address to ACC
	LDR	#n	Immediate addressing. Load the number *n* to IX
	STO	<address>	Store the contents of ACC at the given address
	STX	<address>	Indexed addressing. Form the address from <address> + the contents of the index register. Copy the contents from ACC to this calculated address
	STI	<address>	Indirect addressing. The address to be used is at the given address. Store the contents of ACC at this second address

3.1 COMPUTER ARCHITECTURE

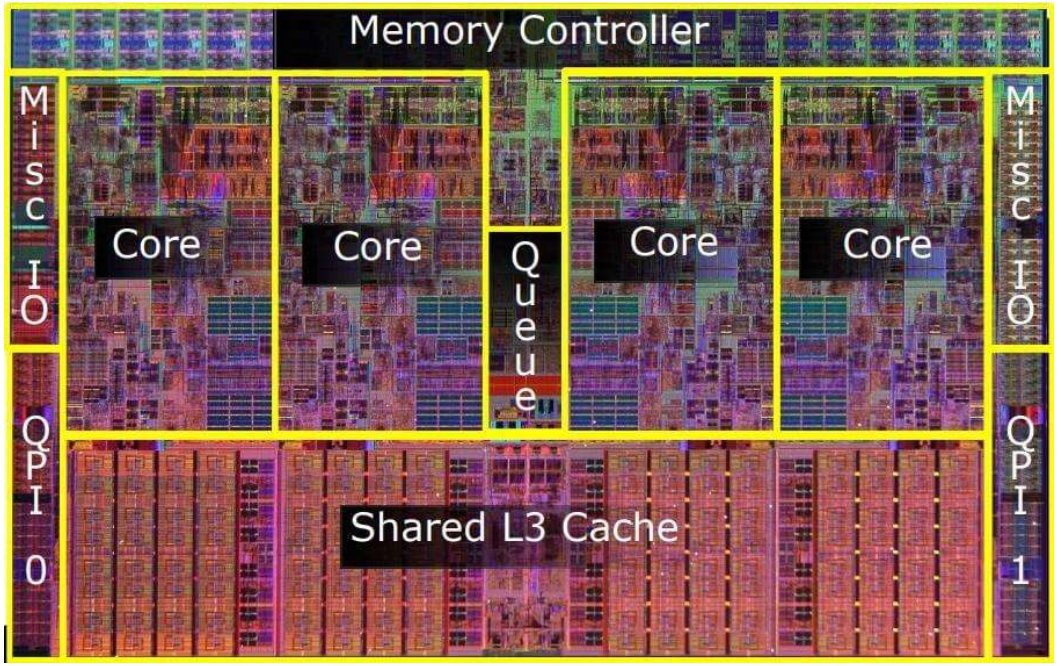
- Buses
  - They are used in computers as parallel transmission components
  - Bus width
    - Number of bits that are sent at a time
  - Address bus
    - It transfers address of memory between components
    - It is unidirectional bus
    - The wider the bus , the more memory locations that can be directly access
    - 16 bits can address 2^{16} (65 536) memory
    - 32 bits allows 4 294 967 296
  - Data bus
    - It transfer data memory between unit
    - It is bi-directional bus
    - The wider bus width, the larger data can be transferred
  - Control bus
    - It transfers signals from CU to tell the other components what to do
    - Bi-directional bus
    - The wider bus width, more instructions can be performed



3.1 COMPUTER ARCHITECTURE

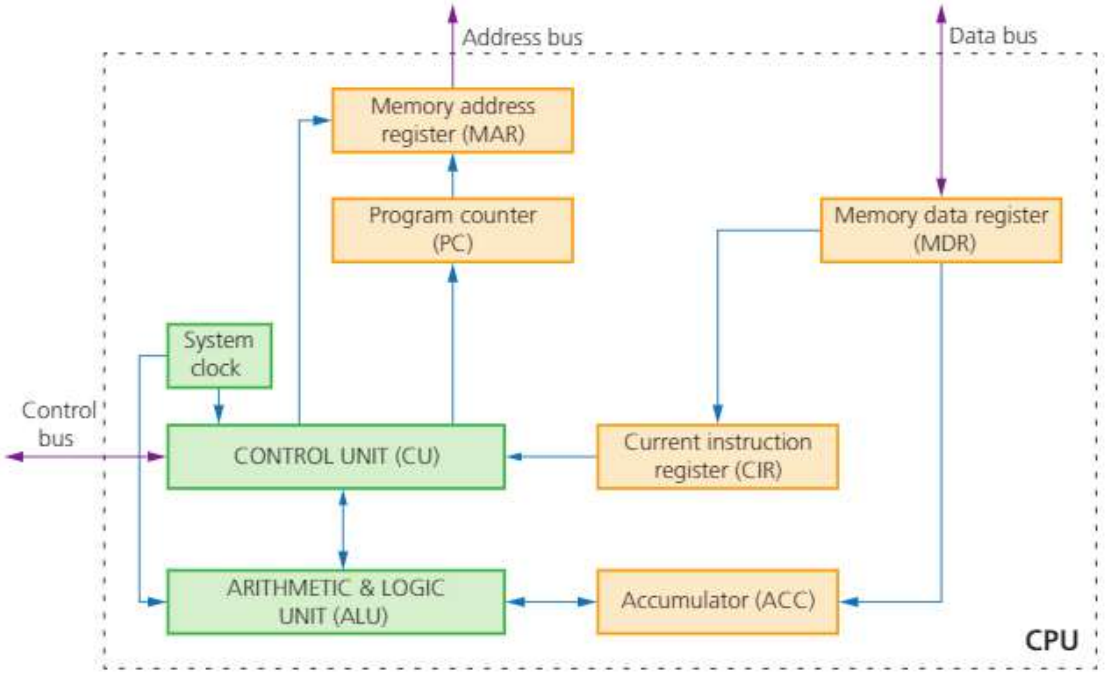


- Central Processing Unit (CPU)
  - Aka microprocessor or processor
  - It executes or process all the instructions and data
  - It consists of
    - Control Unit (CU)
    - Arithmetic and Logic Unit (ALU)
    - Registers and buses
    - These components are very tiny, nanometer size (10^{-9})



3.1 COMPUTER ARCHITECTURE

- Components of the central processing unit (CPU)
  - Arithmetic and Logic Unit(ALU)
    - It performs arithmetic calculation (+, -, shifting)
    - Logical operation (AND, OR, NOT)
  - Control Unit (CU)
    - It sends signals to tell the other component what to do (via control bus)
    - It synchronise of data and instructions with the use of system clock

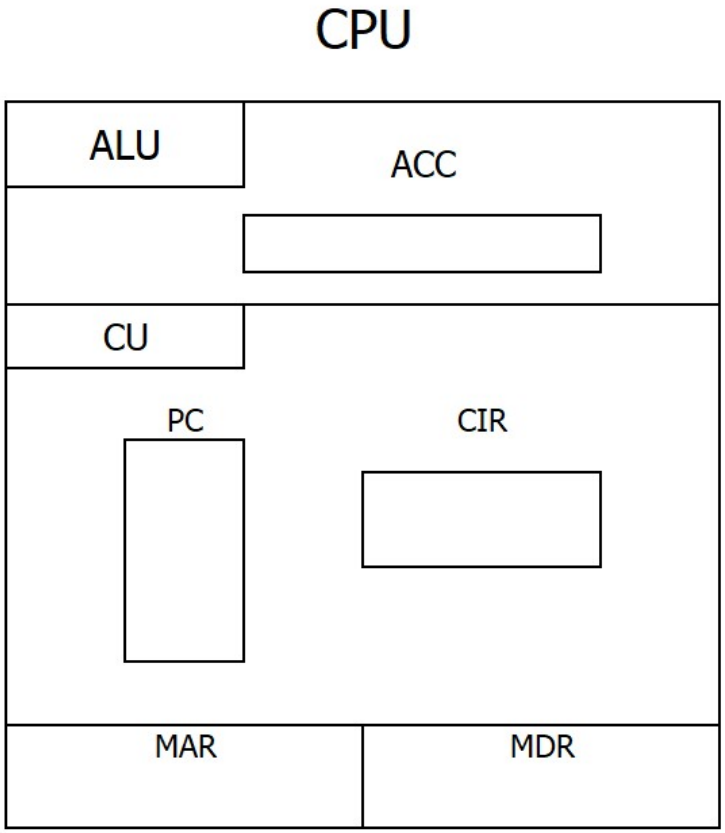


3.1 COMPUTER ARCHITECTURE

- Components of the central processing unit (CPU)
  - Registers

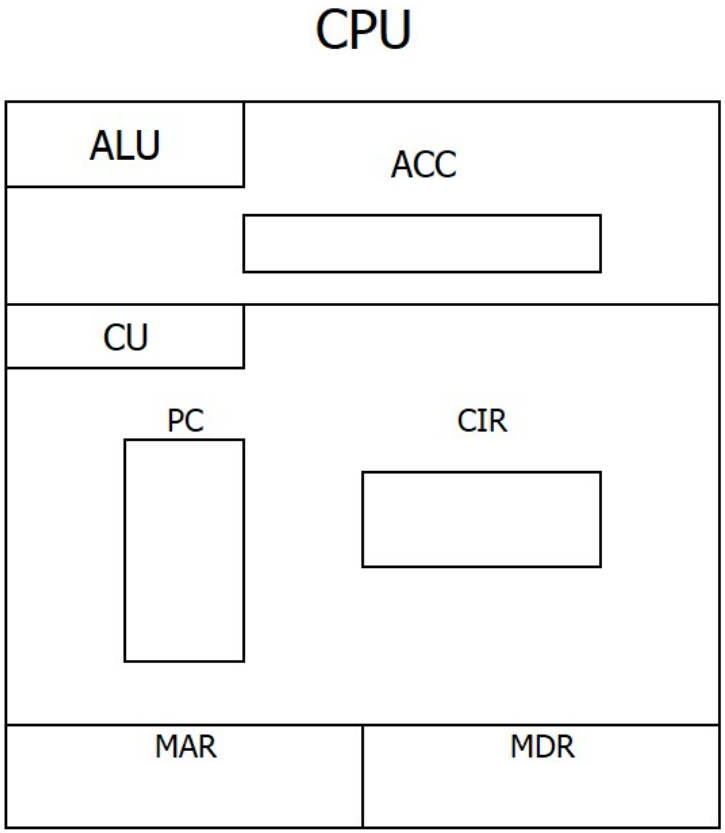
Register	Abbreviation used	Function/purpose of register
current instruction register	CIR	this register stores the current instruction being decoded and executed
accumulator	ACC	this register is used when carrying out ALU calculations; it stores data temporarily during the calculations
memory address register	MAR	this register stores the address of the memory location currently being read from or written to
memory data/ buffer register	MDR	this register stores data which has just been read from memory or data which is about to be written to memory
program counter	PC	this register stores the address where the next instruction to be read can be found

3.1 READ PROCESS



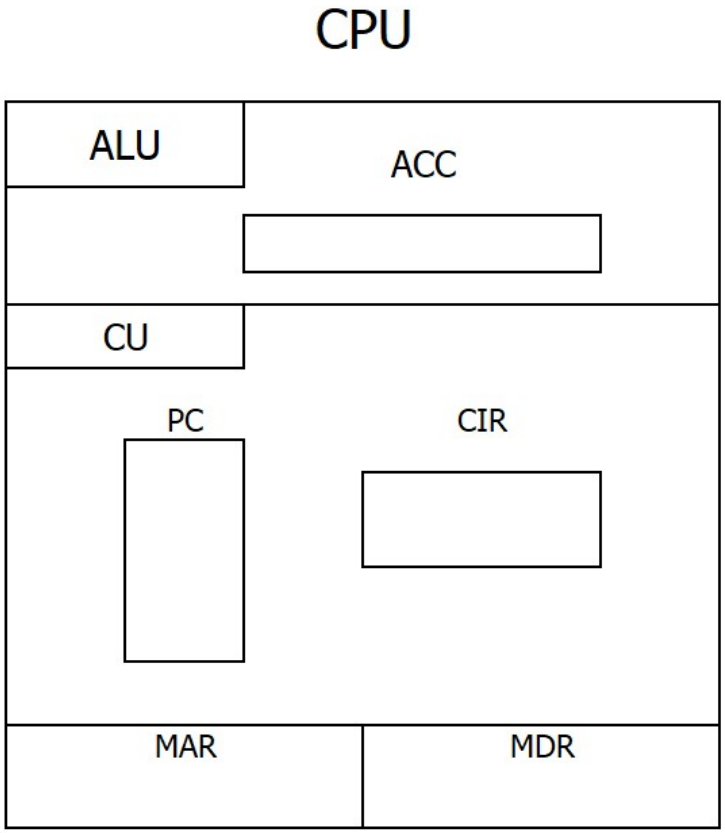
Address	Instruction and data

3.1 READ PROCESS



Address	Instruction and data

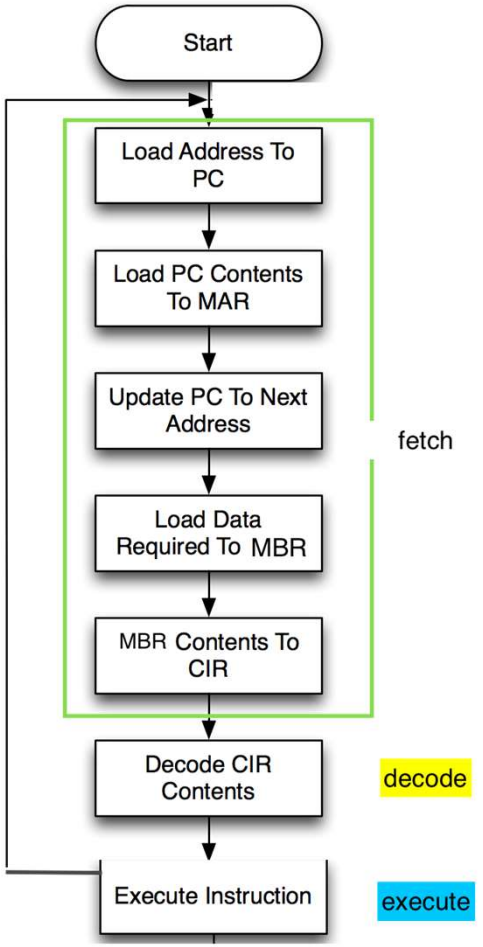
3.1 WRITE PROCESS



Address	Instruction and data

3.1 COMPUTER ARCHITECTURE

- Fetch-Decode-Execute cycle (FE cycle)
  - To carry out a set of instructions the CPU fetches data or instructions then decode the instruction and executes the instruction
- Fetch
  - Address of instruction/data are loaded into PC
  - The address is loaded into MAR and is sent to memory unit with address bus
  - CU sends signal (READ) to the memory
  - Instruction or data at the address is then sent back to MDR
  - The pointer in PC is then moved to the next address
- Decode
  - Instruction from MDR is loaded into CIR to be decoded
- Execute
  - The CPU do the operation as the instruction was decoded



QUESTION

Data can be read but not altered

Carries out operations such as addition and multiplication

Stores bootstrap loader and BIOS

Fetches each instruction in turn

Hold an address of instruction that is about to be read from/written to

Hold instruction being decoded

Carries out operations such as AND, OR, NOT

Stores part of the operating system currently in use

Stores data currently in use

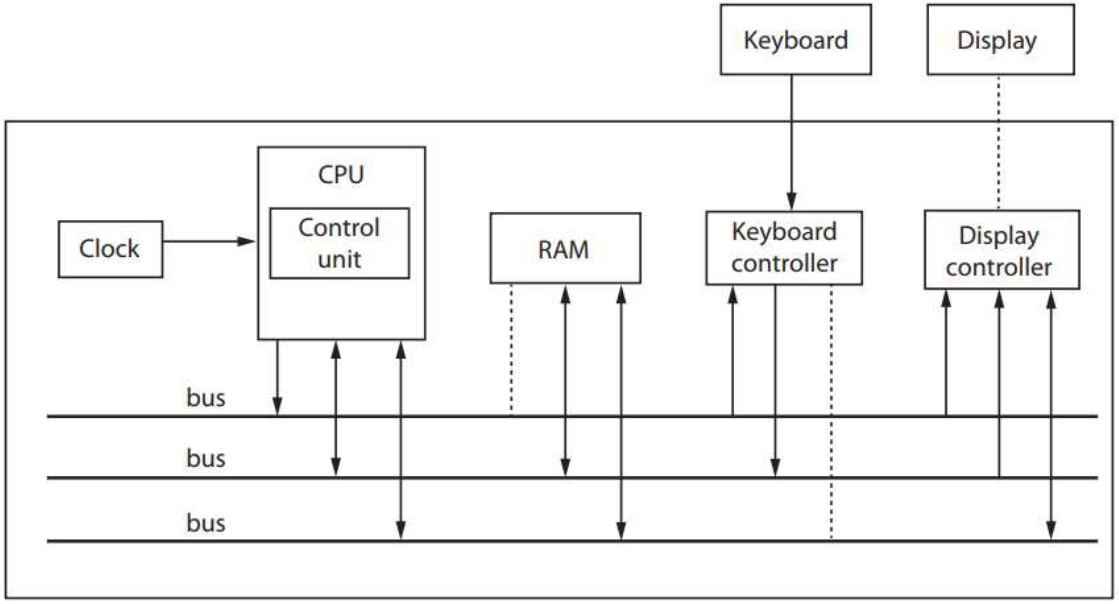
Manages execution of each instruction

Hold data during calculation

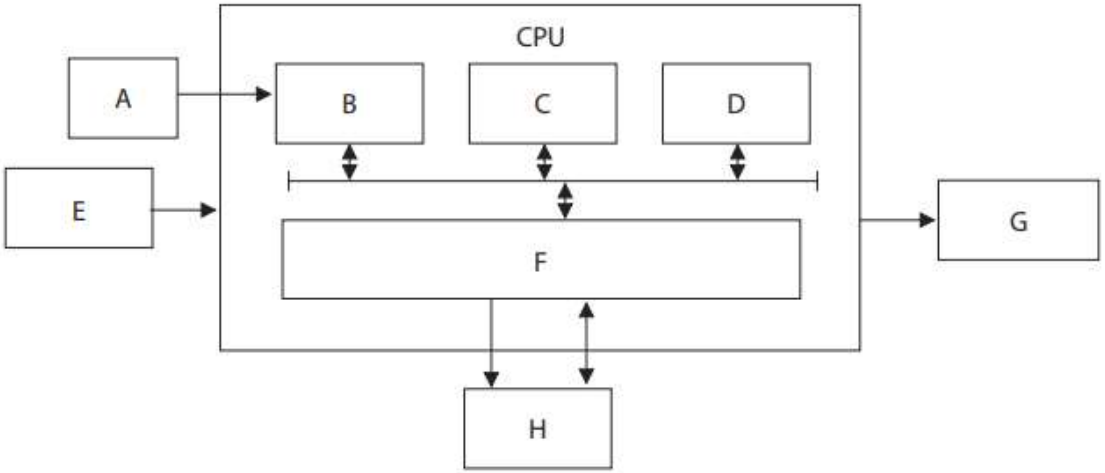
Hold instruction or data that has been read from RAM or about to be written to RAM

QUESTION

- Complete the diagram by labelling the buses and adding directional arrows to the dotted lines to show the flow of communication.



QUESTION



Component	Letter
CU	
Outputs	
Registers	C
RAM	
Cache	F
Inputs	
ALU	
Clock	A