Jawahar Education Societys Annasaheb Chudaman Patil College of Engineering, Kharghar, Navi Mumbai

NAME: PRIYUSH BHIMRAO KHOBRAGADE

PRN NO: 211112018

SUBJECT: DIGITAL LOGIC & COMPUTER ORGANIZATION AND ARCHITECTURE LAB

06

AIM- To Study on CPU.

Practical No.6

● **Aim**: - To Study on CPU

● **Objectives:** Our main objective for this experiment is to show the basic top level functionality, organization and architecture of a computer.

a) Design a single instruction CPU
b) Design a CPU with more instructions

● **Outcomes**: Leaning activities designed in two stages, a basic stage and an advanced stage. In the basic stage, it is recommended to perform the experiment firstly, on the given encapsulated working module, secondly, on the module designed by the student, having gone through the gone through the procedure By performing the experiment on the working module, students can only observe the input-output behaviour. Whereas, performing experiments on the designed module, students can do circuit analysis, error analysis in addition with the input-output behaviour.

● **Hardware / Software Required:** Virtual Lab simulator for Computer Organization and Architecture developed by the Department of CSE, IIT Kharagpur.

● **Theory**:

CPU Design:

At the top level a computer consists of a CPU (central processing unit), memory, I/O components, with one or more modules of each type. These modules are interconnected in a specific manner to achieve the basic functionality of a computer i.e. executing programs. At the top level a computer system can be described as follows:

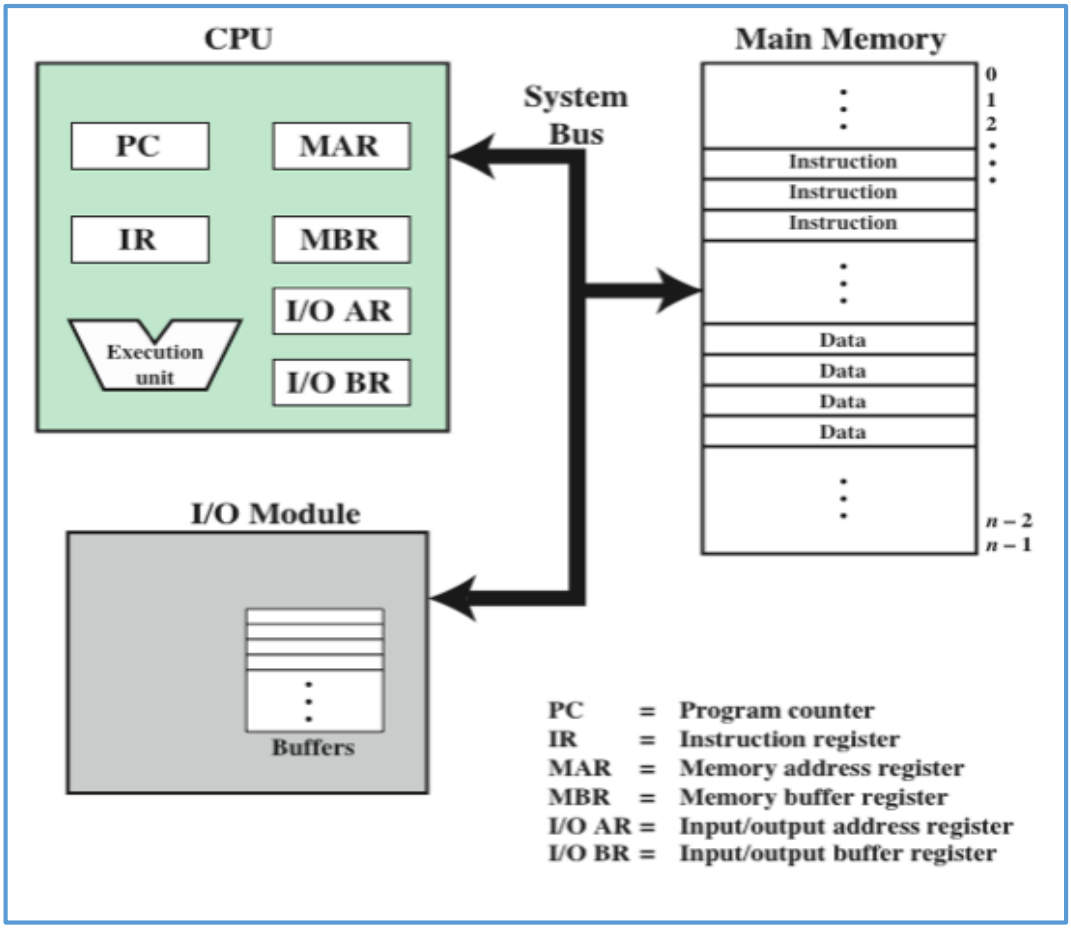
- Describing the external behaviour of each component i.e. the data and the control signals that it exchanges with other components.
- Describing the interconnection structure and the controls required.

We are considering the Von Neumann architecture. Some of the basic features of this architecture are as follows:

- Data and instructions are stored in a single read -write memory.
- the contents of the memory are addressable by location
- execution occurs in a sequential manner (unless explicitly specified) from instruction to the next

AIM: - To Study On CPU.

- **Top level components and interactions among them:**



CPU exchanges data with memory. For this CPU uses two internal resisters. Following is a block diagram of a basic computer system:

- Memory address register (MAR): which specifies the address for the next read or write
- Memory buffer register (MBR): which contains the data to be written into memory or receives the data read from memory.
- I/O buffer (I/O BR): register is used for the exchange of data between an I/O module and the CPU.

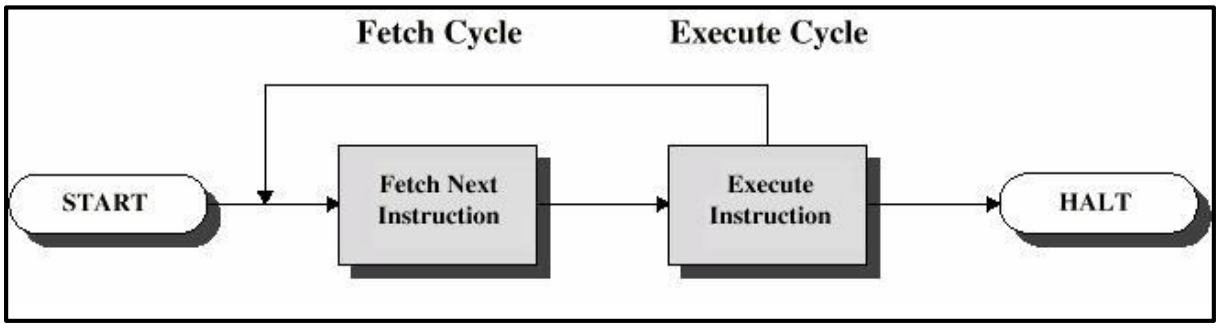
• A memory module consists of a set of a locations defined by sequentially numbered addresses. Each location c interpreted as either an instruction or data.

• An I/O module transfers data from external devices to CPU and memory versa

• The basic function of a computer is to execute a program which consists of a set of instructions stored in the memory. Processing required for a single instruction is called an instruction cycle which consists instruction fetch and instruct execute. A register called program counter (PC) holds the address of the next instruction. Unless told otherwise the processor always increments PC after each instruction fetch so that the next instruction is

fetched in sequence. The fetched instruction is fetched into a register called instruction register (IR).

The basic instruction cycle is shown in the following figure:



- After fetching an instruction, processor executes the instruction by doing some processing on the data which may involve arithmetic and logic unit (ALU), specified by the instruction, then processor writes back the result (if any) to the memory.

This experiment provides a single instruction CPU with built--in controller. A working memory has been provided to check the working principle of the CPU. The single instruction which this CPU supports is SBN (subtract and branch if negative). The format of this instruction is:

- SBN a,b,c Mem[a] = Mem[a] - Mem[b] if(Mem[a] < 0) goto c
- a ,b, c are 4 bit addresses
- Mem[a] denotes the contents of memory address a

In Experiment, no I/O module has been included for the purpose of simplicity. The working memory should contain the program and data in binary format. The CPU executes the program. For halting, it uses self-loop no other halt instruction is provided.

● **Procedure/ Program**:

1. Start the simulator as directed. This simulator supports 5-value logic

2. To perform the experiment on the given modules, we need the CPU, the working memory with a program and data loaded, a clock input, Bit display (to give input, which will toggle its value with a double click), Bit displays(for seeing output), wires.

3. Load memory: click on the memory button in the left pane. You can either load by filling the form or you can directly load the memory provides 4-bit space and 12 bit data word, thus providing 16 memory addresses starting from 0000 to 1111. For loading from file, the

file should contain only binary values; it must contain 16 lines, each line c to be stored in the corresponding memory address. For example, content of first line will be loaded to the 0000 address of the memory, similarly, the second line will correspond to the 0001 address and so on, and finally the content of fed to the 1111 address. The program should use self-loop for halting, for example, the instruction stored at address 1010 will cause self-loop execution, if it content of has -1 in binary format (in 2's complement), the content of b once the execution reaches to this 1010 address, it will finally point to itself.

4. Instantiating the memory: after loading the memory, click on the memory component from the computer design drawer in the palette of the simulator then click on the position of the design editor where you want to put the component (no drag and drop, simple click will serve the purpose).

5. The pin configuration of the component is shown whenever the mouse is hovered on any canned component of the palette or the toolbar will *show it constantly* button on the toolbar will show it constantly in the left pane. Pin numbering starts from 1 and from the bottom left corner (indicating with the circle) and increases anticlockwise.

6. Pin configuration of the memory module:

a. Input pins (upper terminals): memory enable: 30, R/W': 29, address: 25-28 data: 13-24 (13 is LSB)
b. Output pins (lower terminals): data output: 1 -12(1 is MSB).

7. Instantiate the CPU from the computer design drawer in the palette of the simulator then click on the position of the design editor where you want to put the component.

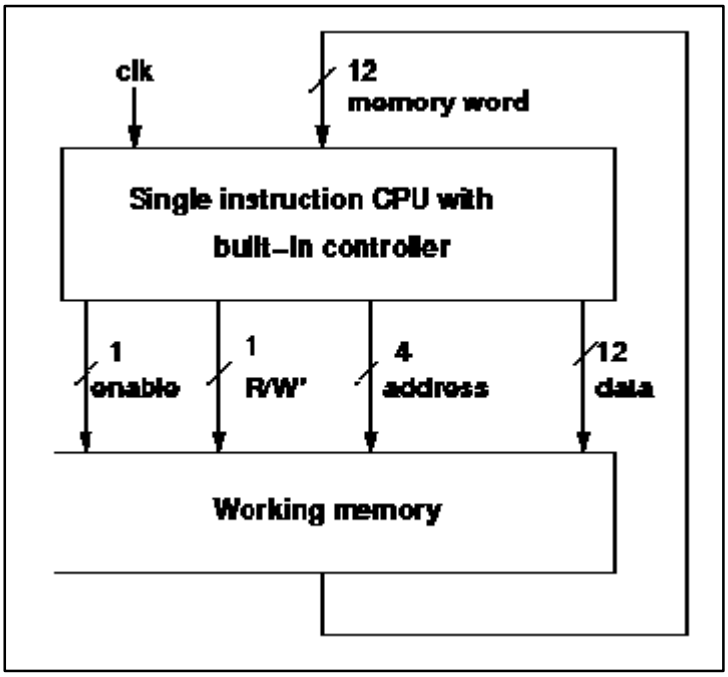
8. Pin configuration of the CPU:

a) Input pins (upper terminals): data input: 20-31(20 is MSB), clock input:19
b) Output pins (lower terminals): memory enable: 1, R/W': 2, address: 3-6(3 is MSB), data output: 7-18(7 is MSB).

9. To connect any two components select the Connection menu of Palette, and then click on the Source terminal and click on the target terminal. According to the following diagram connect all the components. Conn terminals of the CPU, specified data path outputs to the inputs of the controller, the clock input, Bit switches with the inputs and Bit displays component with the outputs (from Display and Input drawer of the pallet, i drawer). After the connection is over click the selection tool in the palette.

10. Start clock and observe the behaviour of the CPU. See the content of memory by clicking show memory button in the left pane. Observe how the program is executing sequentially and modify the data content as per the program.

AIM: - To Study On CPU.



- **Conclusion:** Hence, we **understanding** CPU design and its different modules.