**KWAME NKRUMAH UNIVERSITY OF SCIENCE AND TECHNOLOGY**

**COLLEGE OF ENGINEERING**

**COMPUTER ENGINEERING**



**MEMBERS**

Ofori Emmanuel 9344917

Tweneboah Koduah Emmanuel 9347017

Akunor Sheikh Abdul Basit 9338717

Ofori Evans Gyan 9345017

Razak Tackie 9347917

Ewudzie Eunice Kakra Ama 9342417

**THE 8-BIT CPU**

All computers have central processing unit (CPU). It is responsible for executing and processing data correctly. It performs a fetch-decode-execute cycle on the right data. We are going to explore the inner workings of the 8-bit CPU we built and the various components that it is made up of. This CPU was built using the Von Neumann architecture.

The CPU has three main components; the arithmetic logic unit (ALU), the control unit (CU) and the memory unit (registers). There are other different pieces which come together to make the processor functional. And the roles they play are very important and will be discussed later.

**The ALU**

The arithmetic logic unit does the arithmetic operations and logic operations that are needed when a program is executing. The ALU of this CPU has inputs for the values used for the operations, a select lines input and an output for the result of the operation. The select lines specify what operation the ALU should carry. The result of an operation is put back into the accumulator

**The Control Unit**

The CU can be seen as a coordinator of the CPU. It is responsible for decoding the instruction it receives to assert the correct lines for all the operations that occur in the CPU. Without it the opcode would be useless and instructions wouldn’t be carried out.

**The Registers**

These are the storage units that are directly available to the processor. They make information access very quick. They are used to store all kinds of data including addresses, instructions or the data to be used during program execution. A typical register is made up of D flip-flops and each flip-flop is responsible for storing a bit. So depending on the type of register you want to build say 8-bit register, we would need 8 D flip-flops to build it.

**The Bus**

All the other parts of the CPU communicate with each other through the bus. The bus makes it possible for data and addresses to be moved from one memory location to the other. Since the main bus is shared only selected components can use it at any point in time and this may slow down some program executions. For this CPU we made use of control lines, data bus and address lines.

**The Clock**

For all the components in the CPU to be synchronized the clock is needed. It’s used to regulate the CPU and determines how fast instructions should be executed. The work of the clock is indispensable and without it there would collisions in the bus and data loss which we don’t want to occur.

**A Deeper Insight**

We will look at the high-level view of the CPU and how the pieces come together to make it functional.

OI MI RI RO II IO AI BI AO OI J CE EO S1 S2 S3

BI

II

IO

MI

RI

RO

OI

EO

S

AI

AO

CO

J

CE

4LSB

4MSB

**MAR**

**RAM**

**PC**

**AC**

**IR**

**ALU**

**B Reg**

**OUT Reg**

**DISPLAY**

**CU**

**Status Reg**

4LSB

4LSB

Figure 1

You can clearly see how the different components of the CPU have been placed carefully at their various convenient positions. You can also notice that they are sharing the main bus. This view assumes that the clock has been connected to the pieces that need it. The more detailed schematic of our CPU can be seen on the next page (figure 2). Just as the high-level depicts you can see the different components there too except complete and in details. In the high-level view you can see the control lines coming from the CU are the same ones pointing to the various CPU components. To know which right lines to assert, the CU decodes the instruction coming in from the IR.

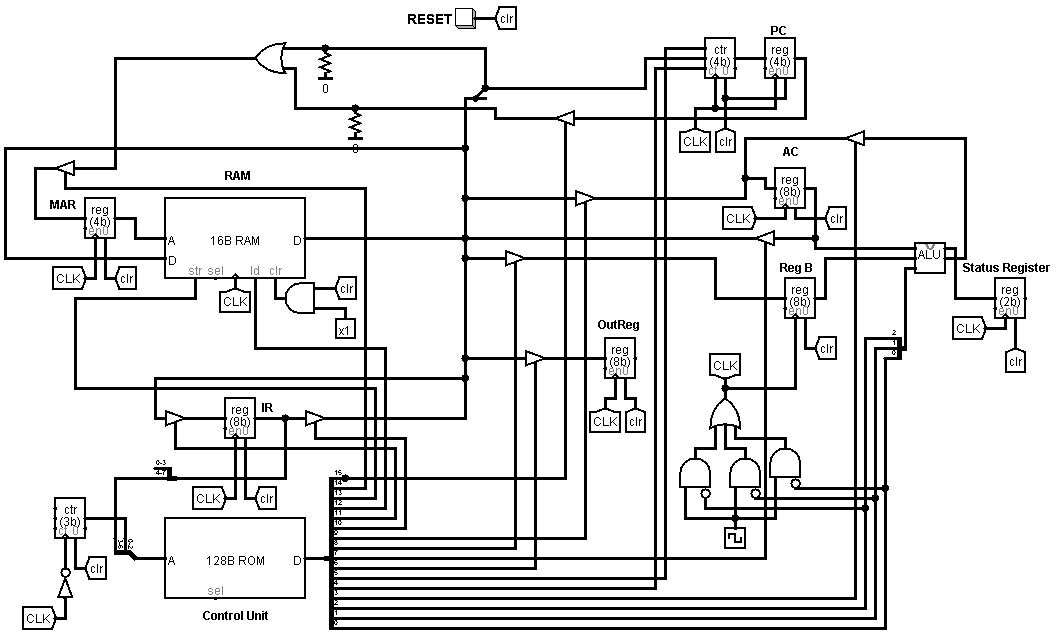


Figure 2

**The ISA**

The instruction set architecture defines a set of instructions that a particular CPU can understand and formatting to use. The ISA sits between the software and the hardware and provides a way for the software to interact with the components of the CPU. For our 8-bit CPU we’ve defined a set of instructions that are understood by this particular CPU. Every instruction contains 8 bits. The 4 MSB of the group of bits (instruction) is the opcode and the 4 LSB is the operand. The opcode simply specifies the actual instruction to be executed and the operand is the address of the value that will be used during the execution of the instruction if it’s needed. The table below shows the specific instruction and the opcode associated with it.

|  |  |  |
| --- | --- | --- |
| **Instruction** | **Mnemonic** | **Opcode** |
| Load | LD | 0001 |
| Store | ST | 0010 |
| Add | ADD | 0011 |
| Subtract | SUB | 0100 |
| Multiply | MUL | 0101 |
| Divide | DIV | 0110 |
| Jump | JMP | 0111 |
| Halt | HLT | 1000 |
| Out | OUT | 1001 |
| OR | OR | 1010 |
| AND | AND | 1011 |
| NOT | NOT | 1100 |

This CPU implements microprogramming which is basically a program which asserts control lines so that a particular task will be carried out. So the instructions are input into a microprogram which converts the instructions into control lines. For the purpose of this CPU the microcode for the program (interpreter) is placed in a ROM.