**Main Goal Of This Project:**

Our main goal in this Project is design a processor for (ADD, AND ,LD, ST ,ADDI, ANDI, CMP, JUMP, JE, JA , JBE, JAE) instruction sets. Processor will have 12 bit address and 16 bits data width. Processor will have 5 parts.Register file holds register values and signal to write into any register. There will be 16 registers in processor. . Instruction Memory will be a read-only memory and instructions will be stored in this component.If the current instruction is not oneof the JUMP instructions(JUMP, JE, JA etc.); the next instruction will be fetched and executed consecutively from this memory. Data Memory will be read-write memory which will store data. Program will be able to read data from data memory, and also store data to this memory. Data Memory will have 12 bits address width, and 16 bits data width. Control Unit will produce proper signals to all datapath components. For example if the instruction is ST, control unit should produce memWrite signal which will allow Data Memory component to write data value on its data input to the address on its address input. Arithmetic Logic Unit (ALU) will compute arithmetic operations ADD,AND,ADDI,ANDI. Operands will be fetched from register+register or register+ immediate value. Result will be stored to the Register File. Control unit should produce proper signals to ALU according to instruction opcode .

Here is our instruction sets:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | OPCODE | | | |  |  |  |  |  |  |  |  |  |  |  |  |
|  | *0* | *1* | *2* | *3* | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| ADD | *0* | *0* | *0* | *0* | **DEST** | | | | **SRC1** | | | | **SRC2** | | | |
| ADDI | *0* | *0* | *0* | *1* | **DEST** | | | | **SRC1** | | | | **IMM** | | | |
| AND | *0* | *0* | *1* | *0* | **DEST** | | | | **SRC1** | | | | **SRC2** | | | |
| ANDI | *0* | *0* | *1* | *1* | **DEST** | | | | **SRC1** | | | | **IMM** | | | |
| LD | *0* | *1* | *0* | *0* | **DEST** | | | | **ADDR** | | | | | | | |
| ST | *0* | *1* | *0* | *1* | **SRC** | | | | **ADDR** | | | | | | | |
| CMP | *0* | *1* | *1* | *0* | 0 | 0 | 0 | 0 | **OP1** | | | | **OP2** | | | |
| JUMP | *0* | *1* | *1* | *1* | **OFFSET** | | | | | | | | | | | |
| JE | *1* | *0* | *0* | *0* | **ADDR** | | | | | | | | | | | |
| JA | *1* | *0* | *0* | *1* | **ADDR** | | | | | | | | | | | |
| JB | *1* | *0* | *1* | *0* | **ADDR** | | | | | | | | | | | |
| JBE | *1* | *0* | *1* | *1* | **ADDR** | | | | | | | | | | | |
| JAE | *1* | *1* | *0* | *0* | **ADDR** | | | | | | | | | | | |

In this iteration we write a java code .Code reads instruction inputs from input.txt and turns them into 16 bit width hexadecimal value.

**Here is our sample inputs and outputs:**

ADD R3,R7,R4 0000001101110100 0374

ADDI R1,R0,7 0001000100000111 1107

And R2,R0,-3 0010001000000011 2203

ANDI R5,R0,-4 0011010100001100 350C

ST R2,3 0101001000000011 5203

LD R4,4 0100010000000100 4404

ST R3,11 0101001100001011 530B

LD R7,3 0100011100000011 4703

JUMP 5 0111000000000101 7005

JUMP -6 0111000011111010 70FA

ADD R0,R0,R0 0000000000000000 0000

CMP R5,R7 0110000001010111 6057

JA -41 1001111111010111 9FD7

JE 65 1000000001000001 8041

JB -3 1010111111111101 AFFD

JBE 7 1011000000000111 B007

JAE -7 1100111111111001 CFF9