| $l$ |  |
| :---: | :--- |
| 1 | Roll XX1 |
| 2 | ADD A,B |
| 3 | XCHG B,A |
| 4 | MOV <br> A,[ADDRESS] |
| 5 | OUT A |
| 6 | INC A |
| 7 | RCR A |
| 8 | MOV A,BYTE |
| 9 | JNZ ADDRESS |
| 10 | PUSH B |
| 11 | POP B |
| 12 | CALL ADDRESS |
| 13 | RET |
| 14 | OR <br> A,[ADDRESS] |
| 15 | XOR <br> A,[ADDRESS] |
| 16 | HLT |


| Roll XX2 |  |
| :--- | :--- |
| 1 | ADD A,B |
| 2 | SUB A,B |
| 3 | XCHG B,A |
| 4 | MOV B,BYTE |
| 5 | RCR B |
| 6 | JMP ADDRESS |
| 7 | JNZ ADDRESS |
| 8 | PUSHF |
| 9 | OR A,BYTE |
| 10 | PUSH B |
| 11 | POP B |
| 12 | OUT A |
| 13 | CALL ADDRESS |
| 14 | RET |
| 15 | AND <br> 16 |
|  | A,[ADDRESS] |

Roll XX3

| 1 | ADD A,B |
| :---: | :--- |
| 2 | SUB A,B |
| 3 | XCHG B,A |
| 4 | IN A |
| 5 | OUT A |
| 6 | INC A |
| 7 | MOV <br> A,[ADDRESS] |
| 8 | MOV A,BYTE |
| 9 | JZ ADDRESS |
| 10 | PUSH B |
| 11 | POP B |
| 12 | RCL B |
| 13 | CALL ADDRESS |
| 14 | RET |
| 15 | AND <br> A,[ADDRESS] |
| 16 | HLT |

Roll XX4

| 1 | ADD A,B |
| :---: | :--- |
| 2 | SUB A,B |
| 3 | XCHG B,A |
| 4 | MOV <br> B,[ADDRESS] |
| 5 | OUT B |
| 6 | JNZ ADDRESS |
| 7 | RCR A |
| 8 | MOV B,BYTE |
| 9 | JMP ADDRESS |
| 10 | PUSH A |
| 11 | POP A |
| 12 | CALL ADDRESS |
| 13 | RET |
| 14 | XOR <br> A,[ADDRESS] |
| 15 | TEST B,BYTE |
| 16 | HLT |


|  | Roll XX5 |
| :--- | :--- |
| 1 | ADD A,B |
| 2 | SUB A,B |
| 3 | XCHG B,A |
| 4 | IN A |
| 5 | RCR B |
| 6 | DEC B |
| 7 | JZ ADDRESS |
| 8 | JMP ADDRESS |
| 9 | OR B,BYTE |
| 10 | PUSH B |
| 11 | POP B |
| 12 | OUT A |
| 13 | CALL ADDRESS |
| 14 | RET |
| 15 | AND <br> A,[ADDRESS] |
| 16 | HLT |

Roll XX6

| 1 | ADD A,B |
| :---: | :--- |
| 2 | SUB A,B |
| 3 | XCHG B,A |
| 4 | RCL A |
| 5 | OUT A |
| 6 | INC A |
| 7 | MOV <br> B,[ADDRESS] |
| 8 | MOV B,BYTE |
| 9 | JMP ADDRESS |
| 10 | PUSH B |
| 11 | POP B |
| 12 | NOT A |
| 13 | CALL ADDRESS |
| 14 | RET |
| 15 | TEST A,B |
| 16 | HLT |

Roll XX7

| 1 | ADD A,B |
| :---: | :--- |
| 2 | SUB A,B |
| 3 | XCHG B,A |
| 4 | MOV <br> A,[ADDRESS] |
| 5 | MOV <br> [ADDRESS],B |
| 6 | OUT A |
| 7 | TEST B,A |
| 8 | OR <br> B,[ADDRESS] |
| 9 | JNZ ADDRESS |
| 10 | JMP ADDRESS |
| 11 | PUSHF |
| 12 | PUSH A |
| 13 | POP A |
| 14 | CALL ADDRESS |
| 15 | RET |
| 16 | HLT |

Roll XX0

| 1 | ADD A,B |
| :---: | :--- |
| 2 | SUB A,B |
| 3 | XCHG B,A |
| 4 | MOV A,[ADDRESS] |
| 5 | RCR B |
| 6 | IN A |
| 7 | OUT A |
| 8 | AND A,B |
| 9 | TEST B,BYTE |
| 10 | OR B,BYTE |
| 11 | XOR A,[ADDRESS] |
| 12 | PUSH B |
| 13 | POP B |
| 14 | CALL ADDRESS |
| 15 | RET |
| 16 | HLT |


| 1 | ADD A,B |
| :---: | :--- |
| 2 | SUB A,B |
| 3 | XCHG B,A |
| 4 | RCL B |
| 5 | SHR A |
| 6 | MOV <br> [ADDRESS],A |
| 7 | XOR <br> A,[ADDRESS] |
| 8 | AND A,B |
| 9 | OR <br> B,[ADDRESS] |
| 10 | OUT A |
| 11 | JZ ADDRESS |
| 12 | PUSH B |
| 13 | POP B |
| 14 | CALL ADDRESS |
| 15 | RET |
| 16 | HLT |

Roll XX8

Roll XX9

| 1 | ADD A,B |
| :---: | :--- |
| 2 | SUB A,B |
| 3 | XCHG B,A |
| 4 | MOV <br> A,[ADDRESS] |
| 5 | MOV <br> [ADDRESS],B |
| 6 | JNZ ADDRESS |
| 7 | XOR <br> A,[ADDRESS] |
| 8 | PUSHF |
| 9 | IN B |
| 10 | OUT A |
| 11 | JMP ADDRESS |
| 12 | PUSH A |
| 13 | POP A |
| 14 | CALL ADDRESS |
| 15 | RET |
| 16 | HLT |

## Note:

Code RAM size: 1024x8
Data RAM size: 1024x8
Input port consists of 8 switches.
Output port consists of 8 LEDs.
Code segment and Data Segment should be separate.

To do:

1. Create an assembler to convert assembly code into machine code (.hex or .bin format)
2. Using simulation software, load the machine code file in a ROM model.
3. Design circuit to copy the content of ROM into appropriate RAM model upon power up
4. Design the 4 bit PC using the RAM models as code and data memory

Bonus points:
Implementation of Pipelining would be favorable for bonus.
Any special/innovative feature that you can implement.

Assignment Grading: You will need to explain the architecture of your design, explain if you have implemented any special / innovative features, and also run a given program.

Individual Submission of Assignment is required. Any sort of Plagiarism is strictly prohibited, and any detected plagiarism would be severely dealt with.

