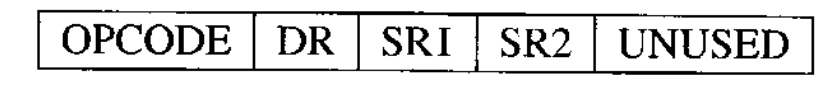
Assignment 3

Computer Organization

Deadline: 11:55pm Tuesday, Dec 3, 2024

Student ID:\_\_\_\_2330016056\_\_\_\_\_\_\_\_\_\_ Name:\_\_\_\_\_\_\_\_Bohan YANG\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. Suppose a 32-bit instruction takes the following format: (10 points)



DR: Destination Register; SR1 & SR2: Source Register 1&2

If there are 225 opcodes and 120 general purpose registers,

1. What is the minimum number of bits required to represent the OPCODE? Explain why. (2 points)
2. What is the minimum number of bits required to represent the Destination Register (DR)? Explain why. (2 points)
3. What is maximum number of UNUSED bits in the instruction encoding? Explain why. (2 points)
4. Suppose the address space contains 216 memory locations and the addressability is 32 bits. How many bytes can the memory store in total? (2 points)
5. How many address lines are necessary? How many data lines are necessary (2 points)
6. 8, because
7. 7, because
8. 3, because
9. 262144 Bytes, because
   1. Address lines:
   2. Data lines: 32
10. The LC-3 does not have an opcode for the logical function OR. That is, there is no instruction in the LC-3 ISA that performs the OR operation. However, we can write a sequence of instructions to implement the OR operation. The four instruction sequence below performs the OR of the contents of register 1 and register 2 and puts the result in register 3. Fill in the two missing instructions so that the four instruction sequence will do the job. (8 points)

(1): 1001 100 001 111111 // R4 = NOT R1

(2): 1001 101 010 111111 // R5 = NOT R2

(3): 0101 110 100 000 101 // R6 = R4 AND R5

(4): 1001 110 110 111111 // R6 = NOT R6

1. The PC contains x3010. The following memory locations contain values as shown: (10 points)



The following three LC-3 instructions are then executed, causing a value to be loaded into R6.

Address X3010:  1110 0110 0011 1111

Address X3011:  0110 1000 1100 0000

Address X3012:  0110 1101 0000 0000

1. Explain what each instruction does. (6 points)
2. What is that value to be loaded into R6? (2 points)
3. We could replace the three-instruction sequence with a single instruction. What is it? Write the instruction in binary. (2 points)
4. **x3010: 1110 0110 0011 1111 // LEA R3 PC+0x3F**

PC + 0x3F = x3011 + x3F = x3050

LEA loads the effective address of a memory location into a register.

R3 now contains the value x3050.

**x3011: 0110 1000 1100 0000 // LDR R4 R3+0**

base address = value in R3 = x3050

Offset = 0

Address accessed = x3050 + 0 = x3050.

The value at memory address x3050 is x70A4.

R4 now contains x70A4.

**x3012: 0110 1101 0000 0000 // LDR R6 R4+0**

Base address = value in R4 = x70A4.

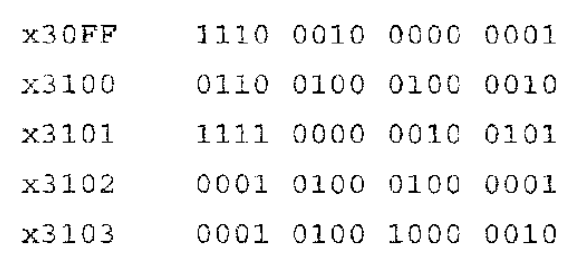
Offset = 0

Address accessed = x70A4 + 0 = x70A4.

The value at memory address x70A4 is x123B.

R6 now contains x123B.

1. The final value loaded into R6 is x123B.
2. **1010 110 00011 1111 // LDI R6 PC+0x3F**
3. Suppose the following LC-3 program is loaded into memory starting at location x30FF: (12 points)



* 1. Explain what each instruction does. (10 points)
  2. If the program is executed, what is the value in R2 at the end of execution? (2 points)
     + 1. x30FF: 1110 0010 0000 0001 // LEA, R1, #1

set R1 to the address PC + 1

R1 = x3101

* + - 1. x3100: 0110 0100 0100 0010 // LDR, R2, R1, #2

loads the value from value in address R1 + 2 to R2

R1 + 2 = x3103

R2 = 0001 0100 0100 0001 = x1482

* + - 1. x3101: 1111 0000 0010 0101 // TRAP x25

Halt (stops execution)

* + - 1. x3102: 0001 0100 0100 0001 // ADD, R2, R1, R1

R2 = R1 + R1 = x6202

* + - 1. x3103: 0001 0100 1000 0010 // ADD, R2, R2, R2

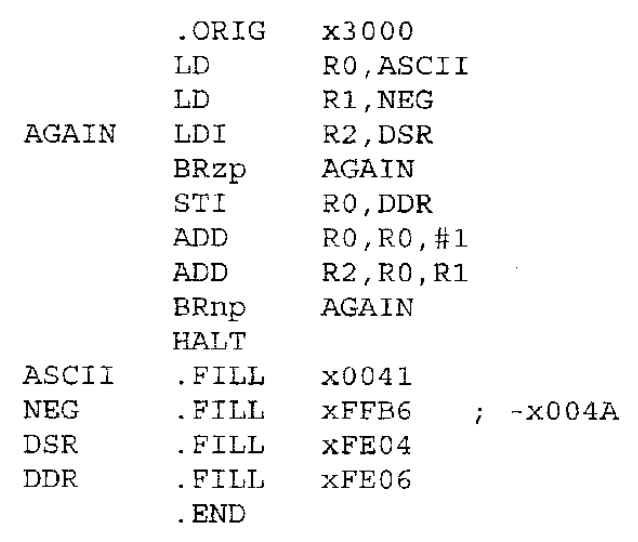
R2 = R2 + R2 = xC404

* 1. The program halts after TRAP x25, so the instructions at x3102 and x3103 are not executed. The value in R2 is x1482.

1. 1) Construct the symbol table for the following LC-3 assembly language program. (10 points)

2) Assemble it into binary machine code line by line by hand. (30 points)

3) What does the program do? (10 points)



* + - * 1. Symbol Table:

|  |  |
| --- | --- |
| Symbol | Address |
| AGAIN | x3002 |
| ASCII | x3009 |
| NEG | x300A |
| DSR | x300B |
| DDR | x300C |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Address | Label | Hex | Binary | Instruction |
| x3000 |  | x2008 | 0010 0000 0000 1000 | LD R0, ASCII |
| x3001 |  | x2208 | 0010 0010 0000 1000 | LD R1, NEG |
| x3002 | AGAIN | xA408 | 1010 0100 0000 1000 | LDI R2, DSR |
| x3003 |  | x07FE | 0000 0111 1111 1110 | BRzp AGAIN |
| x3004 |  | xB007 | 1011 0000 0000 0111 | STI R0, DDR |
| x3005 |  | x1021 | 0001 0000 0010 0001 | ADD R0, R0, #1 |
| x3006 |  | x1401 | 0001 0100 0000 0001 | ADD R2, R0, R1 |
| x3007 |  | x0BFA | 0000 1011 1111 1010 | BRnp AGAIN |
| x3008 |  | xF025 | 1111 0000 0010 0101 | HALT |
| x3009 | ASCII | x0041 | 0000 0000 0100 0001 | NOP |
| x300A | NEG | xFFB6 | 1111 1111 1011 0110 | .FILL xFFB6 |
| x300B | DSR | xFE04 | 1111 1110 0000 0100 | .FILL xFE04 |
| x300C | DDR | xFE06 | 1111 1110 0000 0110 | .FILL xFE06 |

* + - * 1. Print “ABCDEFGHI”