**Wenzhou-Kean University Group Assignment-4 Fall 2021**

**CPS 2390 W\_\_ Computer Organization & Architecture   
Name: \_\_\_\_Wang Xinyu\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ID Number: \_\_\_\_\_1098648\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ID Number: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ID Number: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

# Problem 1 (1 Mark)

Explain the relationship, and their operation, between the PC and MAR registers as well

as the IR and MDR registers during the FETCH cycle. What are their respective sizes in a system with 32 address bits with 64 bit addressability?

MAR use the value in PC to access memory

IR copy the value from MDR

Size of PC: 32 bits

Size of IR: 64 bits

Size of MAR: 64 bits

Size of MDR: 32 bits

# Problem 2 (1 Mark)

What mathematical relationship do the registers R1 and R2 have in relationship to each other when looking at the result in R3 given the following sequence? Assume all register values are positive. (Please show your steps for solving this problem to receive full credit)

1. Convert the binary pattern to its Assembly mnemonics
2. Update/show the contents of the registry file and CC register for each step
3. List the operation being performed

|  |  |
| --- | --- |
| Address | Instruction |
| x4400 | 0101011011100000 |
| x4401 | 0001011011000001 |
| x4402 | 0001010010111111 |
| x4403 | 0000010111111101 |
|  |  |

# 1)

# AND R3, R3, #0

# ADD R3, R3, R1

# ADD R2, R2, #-1

# BRp 0x4401 #-3

2)

x4400: R1=1, R2=1, R3=0

x4401: R1=1, R2=1, R3=1

x4402: R1=1, R2=1, R3=1

X4401: R1=1, R2=1, R3=1

X4402: R1=1, R2=0, R3=1

3)

R3 = R1 \* R2

**Problem 3 (1 Mark)**

The program shown below is loaded into memory starting at address 0x30FF. After the program is run what is the value in register R2? Convert the binary into Assembly.

|  |  |  |
| --- | --- | --- |
| 0x30FF | 1110 0010 0000 0001 | LEA R1 0x1 |
| 0x3100 | 0110 0100 0100 0010 | LDR R2 R1 0x2 |
| 0x3101 | 1111 0000 0010 0101 | TRAP 0x25 |
| 0x3102 | 0001 0100 0100 0001 | ADD R2 R1 R1 |
| 0x3103 | 0001 0100 1000 0010 | ADD R2 R2 R2 |
| R2 contains 0001 0100 1000 0010, | | |

# Problem 4 (1 Mark)

State the contents of the registers R1, R2, R3, and R4 after the program shown below halts. The program starts at 0x3100:

|  |  |
| --- | --- |
| Address | Data |
| 0011 0001 0000 0000 | 1110 001 000100000 |
| 0011 0001 0000 0001 | 0010 010 000100000 |
| 0011 0001 0000 0010 | 1010 011 000100000 |
| 0011 0001 0000 0011 | 0110 100 010 000001 |
| 0011 0001 0000 0100 | 1111 0000 0010 0101 |
| … | … |
| 0011 0001 0010 0010 | 0100 0101 0110 0110 |
| 0011 0001 0010 0011 | 0100 0101 0110 0111 |
| … | … |
| 0100 0101 0110 0111 | 1010 1011 1100 1101 |
| 0100 0101 0110 1000 | 1111 1110 1101 0011 |

R1 = 0x3121

R2 = 0x4566

R3 = 0xABCD

R4 = 0xABCD

# Problem 5 (1 Mark)

Using only the instructions provided for LC-3 create the following assembly operations using only ONE LC-3 instruction:

* 1. MOV: This instruction moves the contents of one register to another without changing the source register.

ADD R1 R2 0x0

* 1. CLR: This instruction clears the contents of a register setting its value to zero.

AND R1 R1 0x0

* 1. LSL: This instruction shifts each of the bits in a register to the left by one position.

ADD R1 R1 R1

* 1. NOP: This instruction does nothing and also does not change the contents of any register.

AND R1 R1 R1