Jaypee Institute of Information Technology, Noida

T2 Solution

ODD Sem 2024

B.Tech. V Semester

Course Title: Computer Organization and Architecture

Course Code: 15B11CI313

Q.1 Write the program code for the following expression with three address and zero address instructions

$$X= A-B+C*(D*E-F)/ G$$

(a) Using a general register computer with three address instructions
(b) Using a stack organized computer with zero-address operation instructions

In the problem, assume that A, B, C, D, E, … reside in memory. Also assume that opcodes are 8 bits, memory addresses are 64 bits, and register addresses are 8 bits. find the total size of code in bytes in each case.

Solution :

To solve this, we will again write the code for both a general register computer (three-address instructions) and a stack-organized computer (zero-address instructions) for the expression:

$X=A-B+C\times(D\times E-F)/G$

We are also given that:

- Opcodes are 8 bits.
- Memory addresses are 64 bits.
- Register addresses are 8 bits.

Now, let's solve this step by step.

(a) Using a General Register Computer with Three-Address Instructions

In a three-address instruction format, an instruction can reference three operands: two sources and one destination. For example: ADD R1, A, B (R1 = A + B).

1. **Step-by-Step Calculation**:
    - T1 = D * E (Temporary value)
    - T2 = T1 - F

- T3 = C * T2
- T4 = T3 / G
- T5 = A - B
- X = T5 + T4

2. **Assembly Code** (three-address instructions):

```
MUL T1, D, E   ; T1 = D * E
SUB T2, T1, F  ; T2 = T1 - F
MUL T3, C, T2  ; T3 = C * T2
DIV T4, T3, G  ; T4 = T3 / G
SUB T5, A, B   ; T5 = A - B
ADD X, T5, T4  ; X = T5 + T4
```

3. **Instruction Size Calculation**:
   - **Opcode size**: 8 bits (1 byte).
   - **Address size**: 64 bits per memory operand.
   - Each instruction has three addresses (source1, source2, destination).
   - **Number of instructions** = 6.
   - **Total size of code** = 18+11+11+11+18+11= **80 bytes** **[1.5 Marks]**

(b) Using a Stack Organized Computer with Zero-Address Instructions

In a stack-organized computer, instructions operate on the stack with implicit operands, meaning the operations use the top elements of the stack.

1. **Step-by-Step Calculation**:
   - Push values onto the stack and perform operations in the correct sequence.
   - T1 = D * E
   - T2 = T1 - F
   - T3 = C * T2
   - T4 = T3 / G
   - T5 = A - B
   - X = T5 + T4
2. **Assembly Code** (zero-address instructions):

```
PUSH D
PUSH E
MUL       ; D * E
PUSH F
SUB       ; (D * E) - F
PUSH G
DIV       ; C * ((D * E) - F)
PUSH C
MUL       ; C * ((D * E) - F) / G
```

```
PUSH A
PUSH B
SUB       ; A - B
ADD       ; (A - B) + (C * ((D * E) - F) / G)
STORE X   ; Store result in X
```

3. **Instruction Size Calculation**:
   - **Opcode size**: 8 bits (1 byte).
   - **Operand size**: Zero-address machines don't include operands in instructions, except for memory access instructions like PUSH and STORE.
   - For PUSH and STORE, a memory address is 64 bits (8 bytes).
   - **Size of PUSH or STORE** = 8 bits (opcode) + 64 bits (memory address) = 9 bytes.
   - **Size of arithmetic instructions (ADD, SUB, MUL, DIV)** = 8 bits (1 byte) each (since no addresses are specified).
4. **Number of instructions**:
   - 7 PUSH instructions.
   - 1 STORE instruction.
   - 5 arithmetic instructions (MUL, SUB, MUL, DIV, ADD).
   - Total instructions = 7 (PUSH) + 1 (STORE) + 5 (arithmetic) = 13 instructions.
5. **Total size of code**:
   - **PUSH/STORE instructions**: 8 × 9 bytes = 72 bytes.
   - **Arithmetic instructions**: 6 × 1 byte = 6 bytes.
   - **Total size of code** = 72 +6 = **78 bytes**. **[1.5 Marks]**

Final Answer:

- (a) Total size for three-address instructions: **80 bytes**.
- (b) Total size for zero-address instructions: **78 bytes**.

Q 2. Assume an instruction set that uses a fixed 16-bit instruction length. Each Operand specifiers requires 6 bits in length. So Two Operand will require 12 bits and One operand 6 bits and so on. There are K two-operand instructions and L zero-operand instructions. What is the maximum number of one-operand instructions that can be supported? Suppose K is 15 and L is 1024 Give possible Opcode Range of Two Operand, One Operand and Zero Operand Instructions in bits.

Solutions:

We have 16 bits. So, total number of possible encodings = 2^16

But all these encodings are not distinct instructions as even if operands are different, we consider them as same instruction. So, we have to find the number of each types of instructions.

We have L Zero operand instructions. Number of encodings taken by these = L as they don't have any operand part.

We have K Two operand instructions. Number of encodings taken by these
= K×2^6×2^6=K×2^12 as there are 2 operands and each being 6 bits.

All the remaining encodings can be used for 1 operand instructions, which will be equal to 2^16−L−2^12×K

and M number of One operand (of 6 bits) instructions possible will be

2^16−L−2^12×K= 2^6x M

M= (2^16−L−2^12×K)/2^6

Putting Values K=15, L=1024 and M is calculated as 48 **[1.5 Marks]**

Range of Instructions.

15 Two Operand Instructions: 0000-1110

48 One Operand Instructions: 1111 0000 00 – 1111 1011 11

1024 Zero Operand Instructions: 1111 1100 0000 0000 – 1111 1111 1111 1111 **[1.5 Marks]**

Q3. The instruction **LHLD 3148H** is stored at memory location **2500H** in an 8085 microprocessor. This instruction is used to load the contents of two consecutive memory locations into the **H** and **L** registers. Assume that the memory locations **3148H** and **3149H** hold the values **34H** and **12H**, respectively. The machine code for LHLD is **2AH**.

(a) Find the number of machine cycles and T-states required for the execution of this instruction**.**

(b) After the execution of the LHLD instruction, what will be the values of the **L** and **H** registers?

(c) Draw the timing diagram for the execution of the LHLD 3148H instruction, including all necessary address, data, ALE, IO/M, S0, S1.

Solution- (a)

Opcode Fetch Cycle (4T):The opcode for LHLD is fetched from memory location 2500H.

Memory Read Cycle 1 (3T):Read the contents from memory location 3148H into the L register.

Memory Read Cycle 2 (3T):Read the contents from memory location 3149H into the H register.

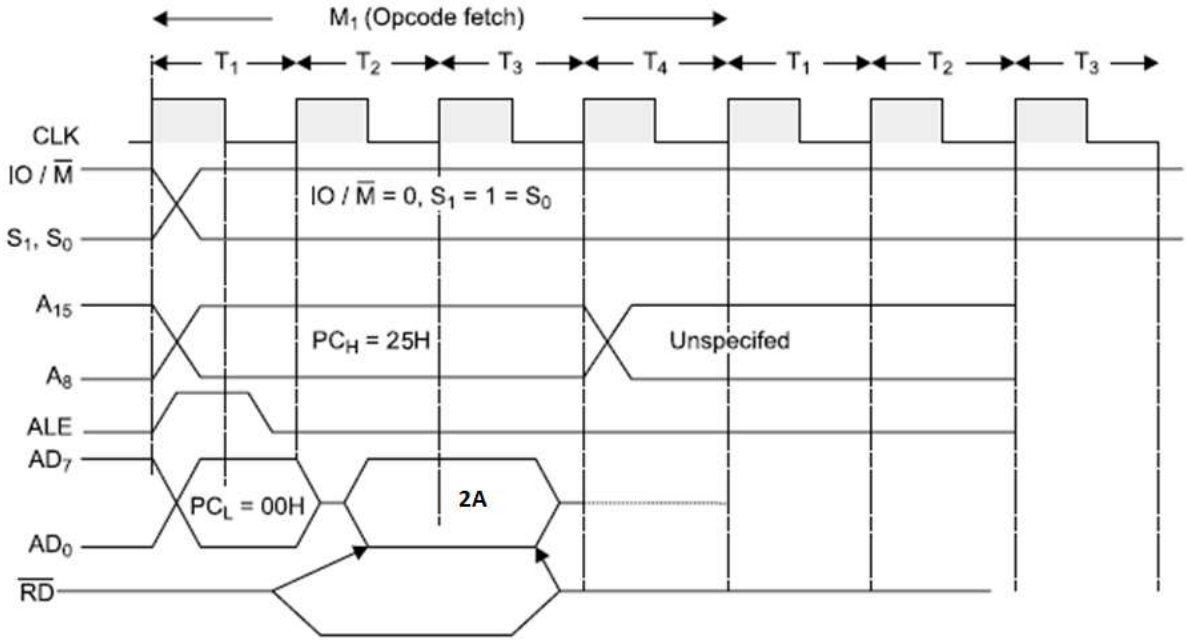
Total Machine Cycles and T-States:

- Total Machine Cycles: 3
- Total T-States = 4T (Opcode Fetch) + 3T + 3T +3T+3T= 16 T-states **[1Marks]**

(b) After executing the LHLD instruction:

- The contents of 3148H (34H) will be loaded into the L register (L Register = 34H).
- The contents of 3149H (12H) will be loaded into the H register (H Register = 12H). **[1Marks]**

(c) Timing Diagram for LHLD 3148H **[2 Marks]** **1 mark for timing diagram 1 marks for values of address and opcode**



Q4. [C213.5, Analyze Level, 4 Marks] Suppose 8085 microprocessor uses RAM chips of 1024 × 4 bit capacity. How many chips will be required to obtain a memory of capacity of 64 K bytes? Also explain how the chips are to be connected to address bus. Draw diagram for selection/ decoder logic and give address range for each byte read.

Solution:

Solution:

As given available chips = 1024 x 4 capacity (2^{10})

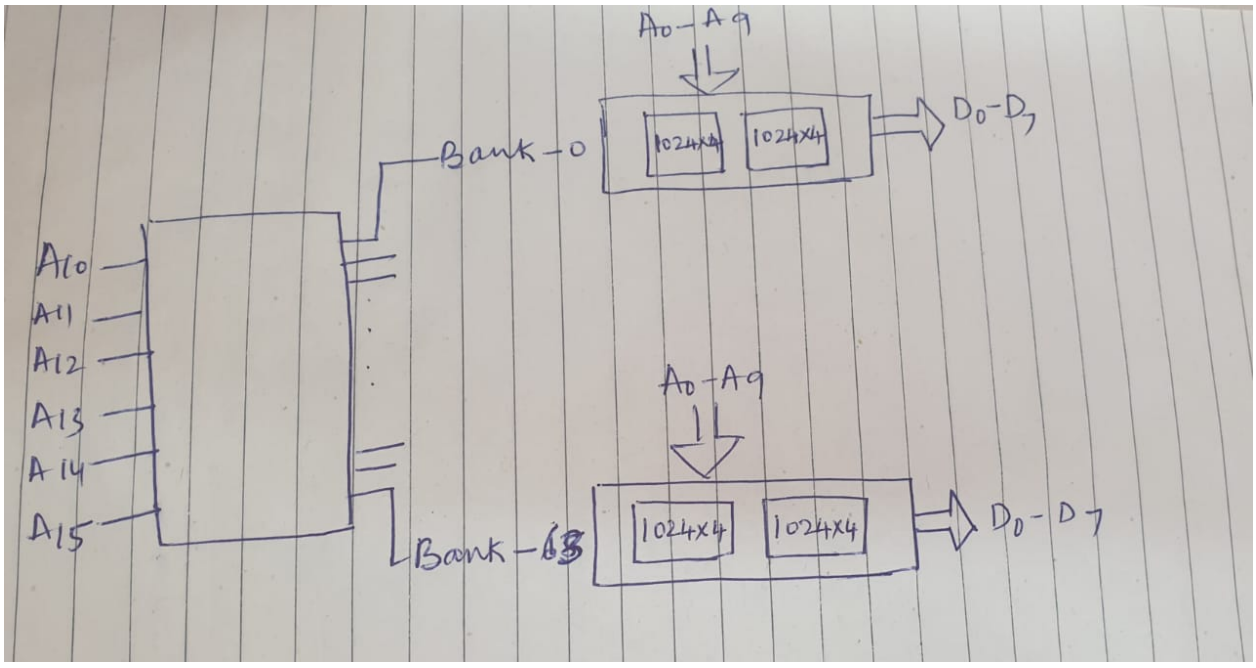
Required capacity = 64 x 1024 x 8 capacity
Number of Chips= (64 x 1024 x 8)/(1024X4) =64 x 2= 128 **[1Marks]**

Number of address lines are needed =64K= 64 x 1024 (that is 64 x 1024 $= 2^{16}$)

2 chips connected to parallel to form 8-bit word. So address lines will be common (A_9-A_0) to these chips. If we say it is a one bank, there will be total of 64 bank for 64KB memory interface.

Remaining address lines A_{15}-A_{10} select one bank from them.

- The first 10 address lines (A0 to A9) will be used to address the locations inside the RAM chip.
- The remaining 6 address lines (A10 to A15) will be used to select one of the 64 sets of two RAM chips, because each set of two chips stores 1 byte.
- A decoder is required to generate chip select signals for each pair of chips. A 6-to-64 line decoder (because there are 6 address lines) can be used to select the appropriate chip pair



Decoder Logic Diagram: [2Marks]

Bank-0 Address range is 0000h to 03FFh

Bank-63 Address Range is FC00h to FFFFh

[1Marks]

Q5. Calculate the time delay for this program. Assume the clock frequency of the system is **5 MHz**. Calculate the total time delay for this program. What will be the final value in the accumulator after the execution of the loop? **[3 Marks]**

MVI A, 25H

MVI B, 02H

MVI C, 07H

LOOP: ADD B

DCR C

JNZ LOOP

Solution

MVI A, 25H----------7T

MVI B, 02H---------- 7T

MVI C, 0H---------- 7T

LOOP: ADD B---------- 4T

DCR C----------4T

JNZ LOOP----------10/7

Total T state for initialization = 7+7+7=21 T-states

Total T state for Loop Execution:

Total for 7 iterations:

- First to Sixth Iteration: 4+4+10=18 T-states each
- Seventh Iteration: 4+4+7=15 T-states
- Total for the loop = 6×18+15=108+15=123 T states

Total T-States: Initialization + Loop=21+123=144 T states **[1 Marks]**

Total Time Delay=Total T-States × Clock period=144×0.2µs=28.8µs [1 Marks]

Final Value in Accumulator (A): 33H [1 Marks]

Q6. Given MIPS Assembly Code

```
lw $t1, 8($t2)       # Line 1
beq $t0, $t1, 8      # Line 2
```

Initial Conditions:

- Instruction memory starts at address 0x2000.
- $t2 = 0x1000
- Memory[0x1008] = 5
- $t0 = 5

What **Instruction format** and **addressing mode** is used for the lw and beq instruction in Line 1 and 2, and what is the **effective/target address** for the memory access?

Solution: **Addressing Mode and Effective Address for** lw $t1, 8($t2)

- **Instruction:** lw $t1, 8($t2)
- **Addressing Mode: Base Addressing Mode** (or Displacement Addressing Mode) **[0.5 Marks]**
- **Effective Address Calculation**:
  - Base Address = $t2 = 0x1000
  - Offset = 8
  - **Effective Address**: =0x1000+8=0x1008 **[1Marks]**
- **Value Loaded**:
  - The value at Memory[0x1008] = 5, so: $t1=5

2. Addressing Mode and Effective Address for ***beq $t0, $t1, 8***

- **Instruction:** beq $t0, $t1, 8
- **Addressing Mode: PC-relative Addressing Mode [0.5 Marks]**
- **Current PC Calculation**:
  - The address of the lw instruction is 0x2000, so the address of beq is: PC Current PC=0x2000+4=0x2004
- **Offset Calculation**:
  - The given offset in the instruction is 8 (in words).
  - **Effective Offset in bytes**: Offset bytes Effective Offset=8×4=32 bytes
- **Label (Target Address) Calculation**:
  - **Target Address**: Address PC Offset Target Address = Current PC+ Effective Offset =0x2004+4+8x4 = 0x2028 **[1 Marks]**
- **Branch Condition**:
  - $t0 = 5 and $t1 = 5.
  - Since 5 == 5, the branch **will be taken**, and the program will jump to the **target address** 0x2024.