**Ramaiah Institute of Technology**
**(Autonomous Institute, Affiliated to VTU)**

**Department of CSE**

**Programme: B.E**
**Course: Computer Organization**

**Term: Jan to May 2019**
**Course Code: CS45**

Activity V: Designing an ALU to perform arithmetic and logical functions using Logisim simulator.

| | | |
|---|---|---|
| **Name:** MUSKAN GUPTA | **Marks:** /10 | **Date:** 21/5/20 |
| **USN:** 1MS18CS078 | **Signature of the Faculty:** | |

**Objective:** To simulate the working of Arithmetic and Logical Unit using simulator.

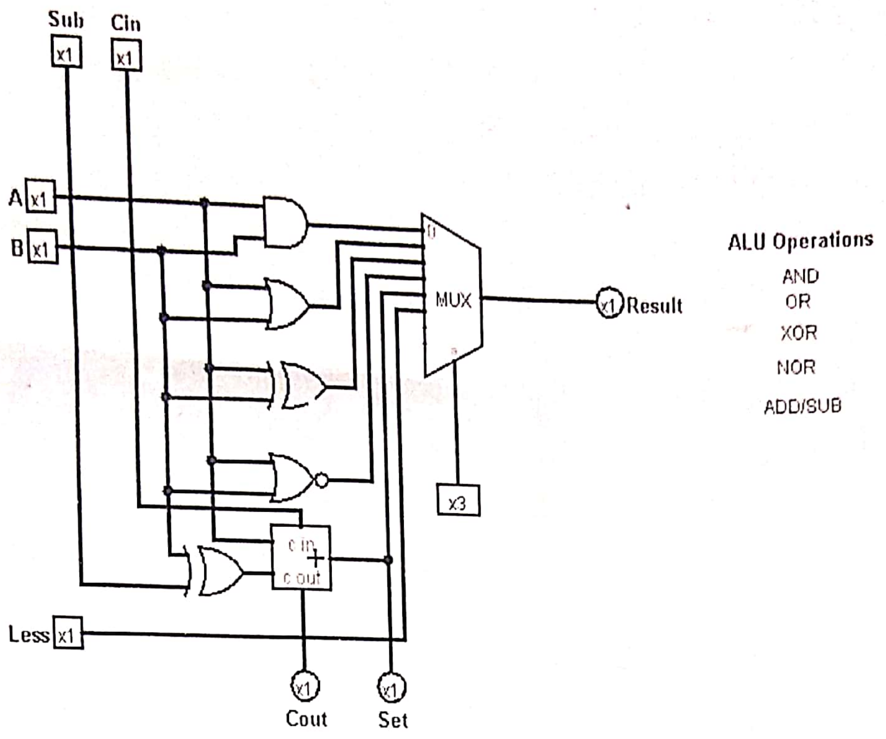
**Simulator Description:** Logisim is an educational tool for designing and simulating digital logic circuits. With its simple toolbar interface and simulation of circuits as you build them, it is simple enough to facilitate learning the most basic concepts related to logic circuits. With the capacity to build larger circuits from smaller sub circuits, and to draw bundles of wires with a single mouse drag, Logisim can be used (and is used) to design and simulate entire CPUs for educational purposes.

**Activity to be performed by students:**

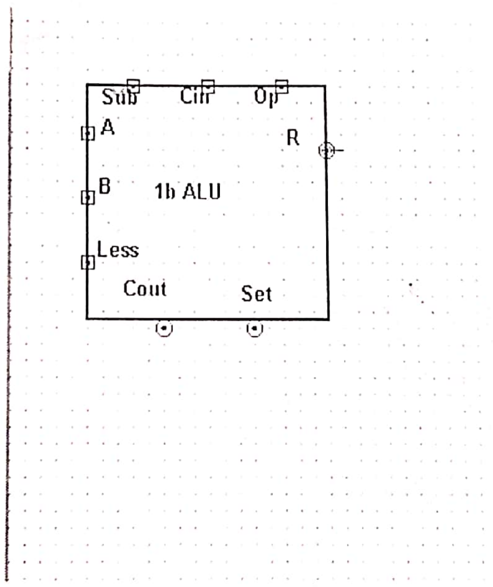
List out the steps in designing ALU

1. Add the two i/p pins, Name them A and B.
2. Add OR, AND, EX-OR, NOR gates and a 1-bit adder.
3. Connect the A's and B's of all the gates to their respective pins.
4. Add an output pin and name it Result.
5. Add a 1-bit multiplexer with 3 select bits.
6. ~~Connect~~ Connect the outputs of all gates to the mux.
7. Connect 3-bit input pin to mux.
8. Add i/p pin to Cin and output pin to Cout.
9. Add an EX-OR gate. Connect its o/p to Cout. The first i/p must be connected to B and the second to another i/p pin SUB.
10. Add another i/p and name it Less. Connect it to mux.
11. Add an output pin and name it Set. Connect it to the multiplexer o/p of adder unit.

## Snapshots

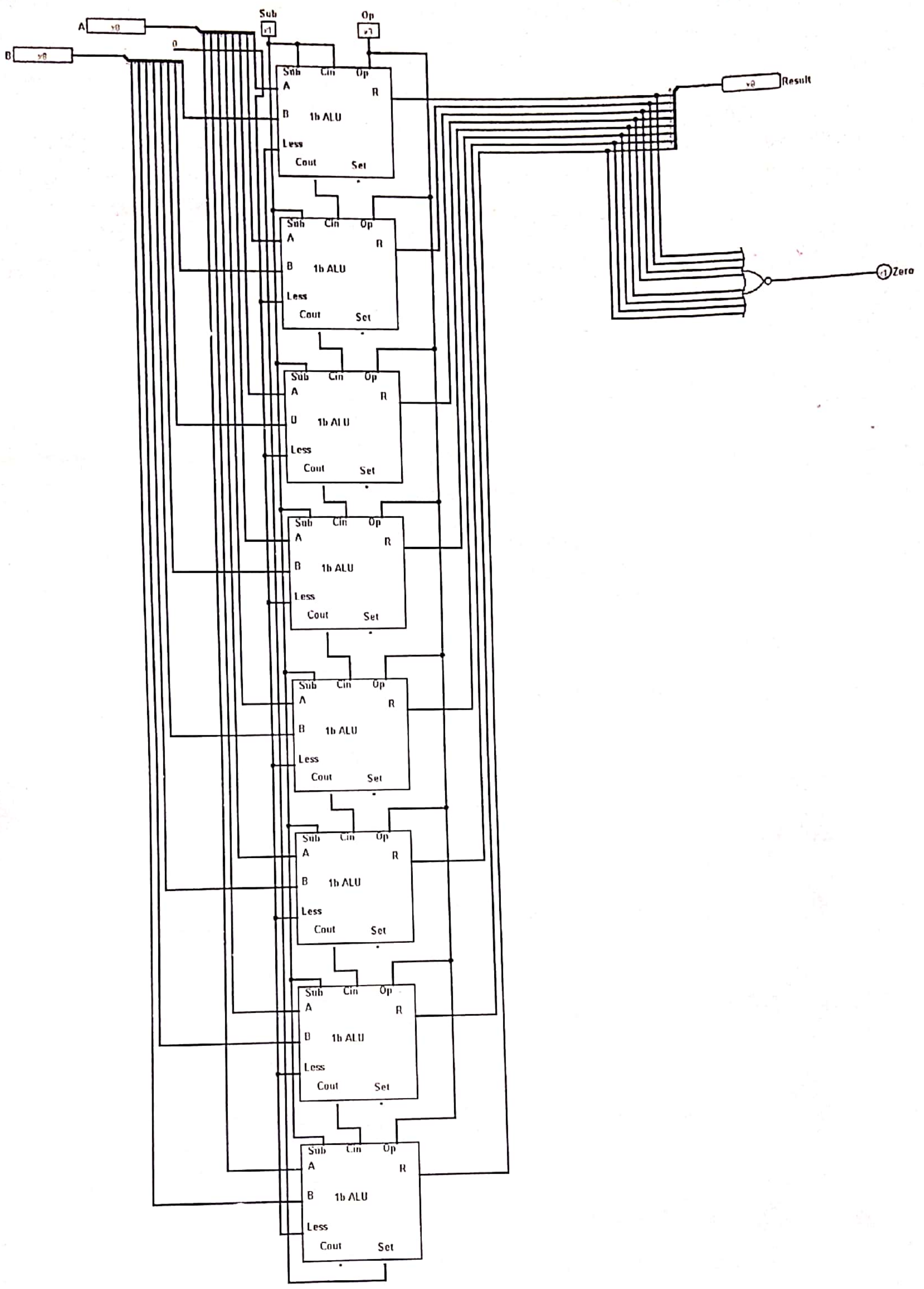


1-bit ALU



ALU object

Snapshots


A
x0
B
x0
0
Sub
x1
Op
x1
Result
x0
Zero
x1
Sub
Cin
Op
A
R
B
1b ALU
Less
Cout
Set

Ramaiah Institute of Technology
(Autonomous Institute, Affiliated to VTU)

Department of CSE

| | |
|---|---|
| Programme: B.E | Term: Jan to May 2019 |
| Course: Computer Organization | Course Code: CS45 |

**Activity VI:** Designing memory system using Logisim simulator.

| | | | |
|---|---|---|---|
| Name: Muskan Gupta | | Marks: /10 | Date: 20/5/20 |
| USN: 1MS18CS078 | | Signature of the Faculty: | |

**Objective:** To simulate the writing operation on memory.

**Simulator Description:** Logisim is an educational tool for designing and simulating digital logic circuits. With its simple toolbar interface and simulation of circuits as you build them, it is simple enough to facilitate learning the most basic concepts related to logic circuits. With the capacity to build larger circuits from smaller sub circuits, and to draw bundles of wires with a single mouse drag, Logisim can be used (and is used) to design and simulate entire CPUs for educational purposes.

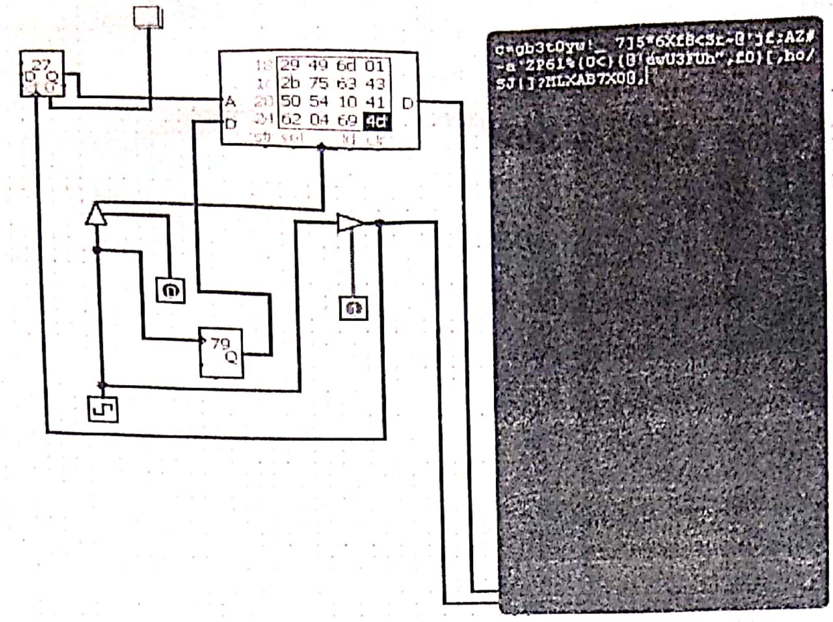
**Activity to be performed by students:**

**List out the steps in designing memory system**

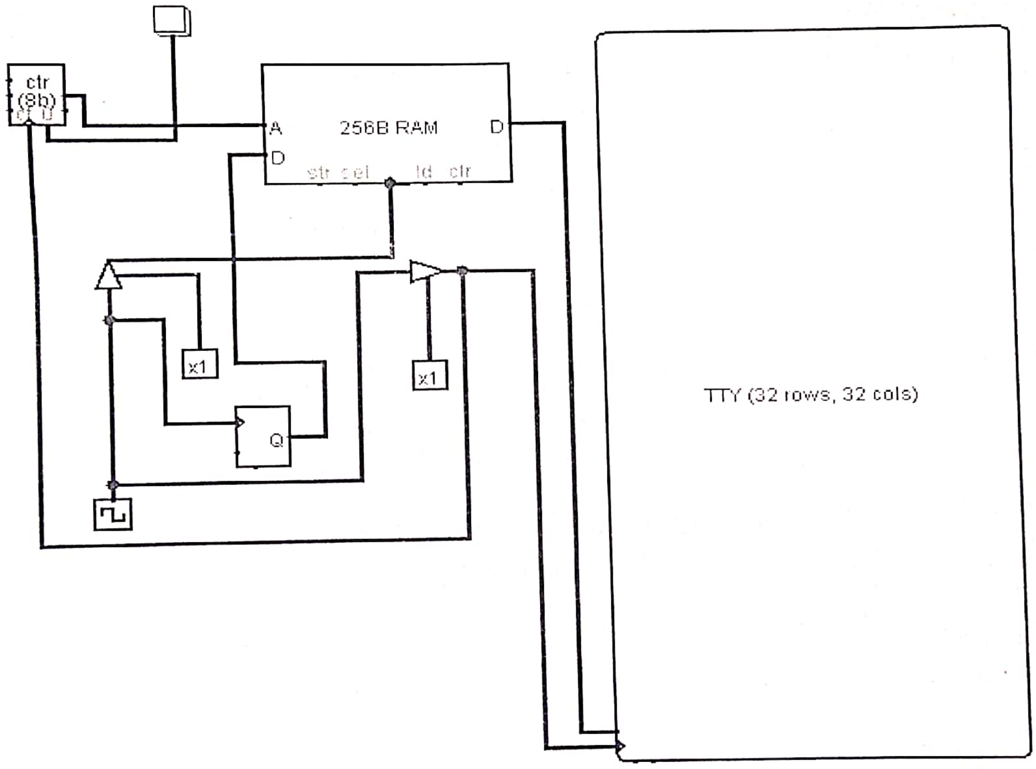
1. Add a RAM with separate load and store selected.
2. Add a counter and convert Q to A of the RAM.
3. Add a controller buffer and connect its o/p to RAM.
4. Add a clock and connect to the i/p of the buffer.
5. Add a TTY unit with 32 rows and columns.
Make the connections with RAM.
6. Add a 7-bit random number generator, connect Q to D.
7. Add another controlled buffer connect it to TTY. Also add an i/p pin to the buffer.
8. Connect the o/p of the second buffer to the counter.
9 Connect a button to the counter.

Observations and Snapshots:

Snapshots -



```
c*gb3t0yw!_ 7]5*6X£B<3r~@'}£;AZ#
~a'ZP61%(0<)(B'dwU3FUh",£0)[,ho/
SJ|]?MLXAB7X00,|
```



**Ramaiah Institute of Technology**
**(Autonomous Institute, Affiliated to VTU)**

**Department of CSE**

**Term: Jan to May 2019**
**Course Code: CS45**

**Programme: B.E**
**Course: Computer Organization**

Activity VII: To simulate advantages of using pipeline technique in executing a program.

| | | |
|---|---|---|
| | | **Date:** 22/5/20 |
| **Name:** Muskan Gupta | **Marks:** /10 | |
| **USN:** 1MS18CS078 | **Signature of the Faculty:** | |

**Objective:** To learn and analyze the performance of the CPU by overlapping of instructions using CPUOS-SIM simulator.

**Simulator Used:** CPUOS-SIM is a software development environment for the simulation of simple computers. It was developed by Dale Skrien to help users to understand computer architectures.

Modern CPU`s contain several semi-independent circuits involved in decoding and executing each machine instruction. Separate circuit elements perform each of these typical steps:

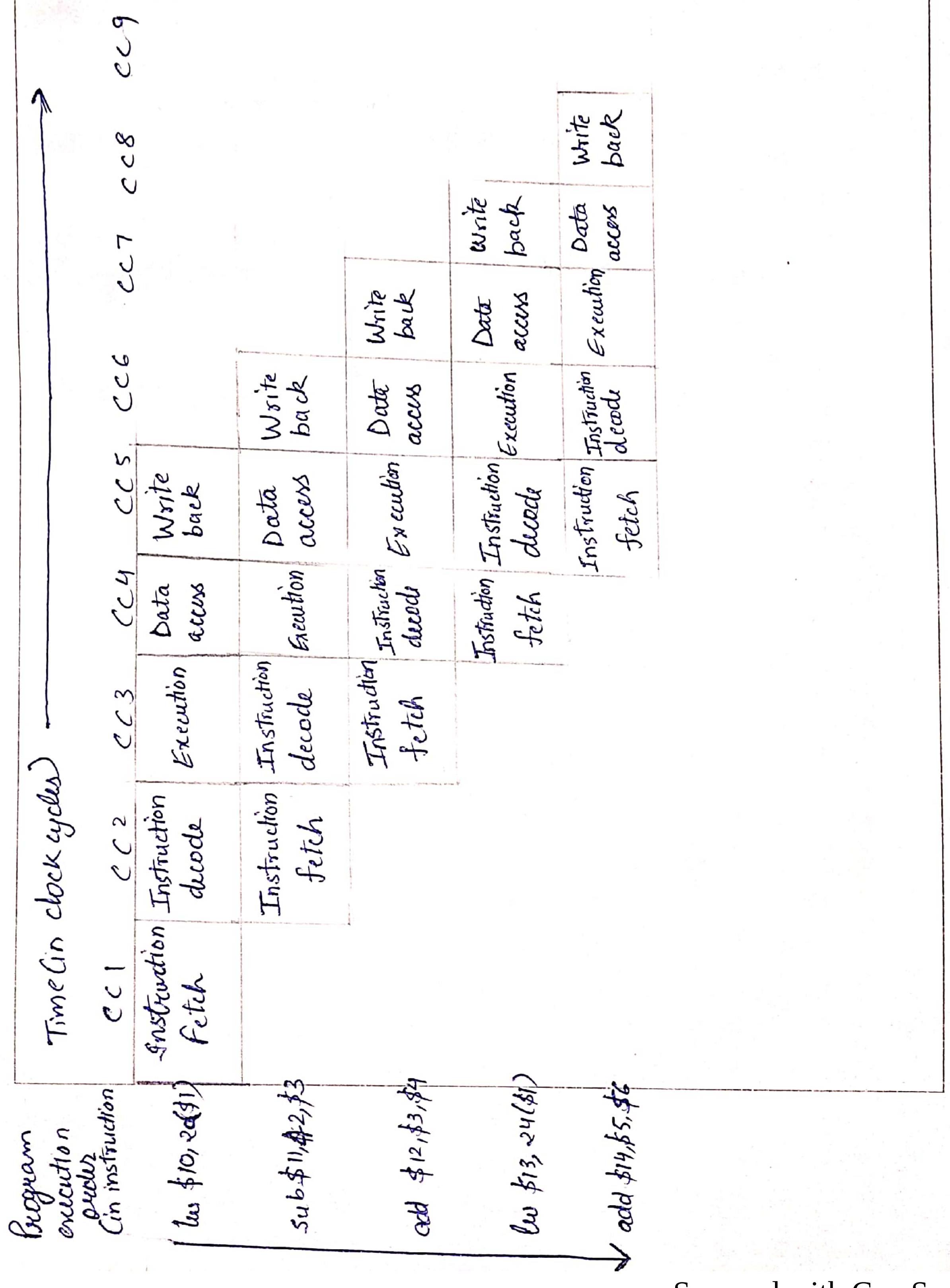
- Fetch the next instruction from memory into an internal CPU register.
- Decode the instruction to determine which function sub-circuits it requires.
- Read any input operands required from high-speed registers or directly from memory.
- Execute the operation using the selected sub-circuits.
- Write any output results to high-speed registers or directly to memory.

Separate sections of the CPU circuitry are used for each of these steps. This allows these circuit sections to be arranged into a sequential pipeline, with the output of one step feeding into the next step.

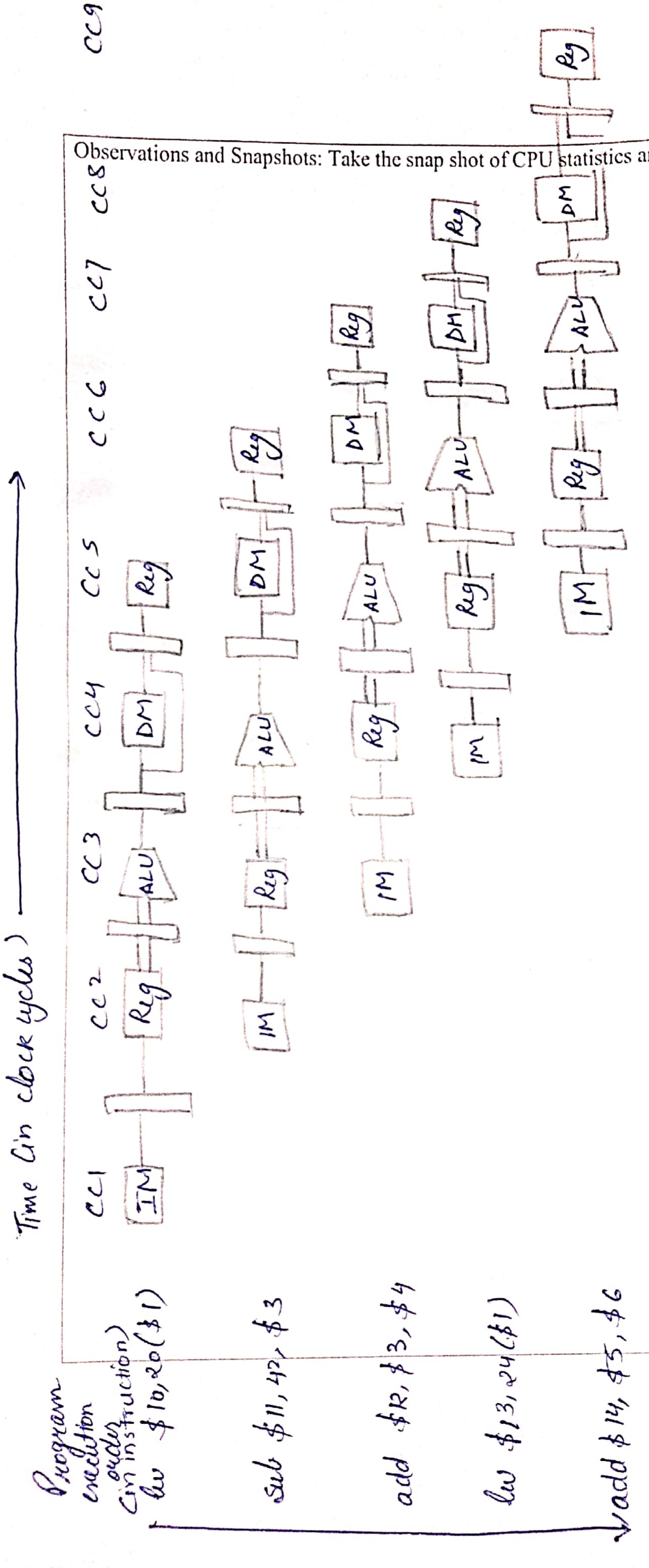
**Activity to be performed by students:**

With diagram demonstrate the execution of the following instructions using pipelining technique.

lw $10,20($1)
sub $11, 42, $3
add $12, $3, $4
lw $13, 24($1)
add $14, $5, $6

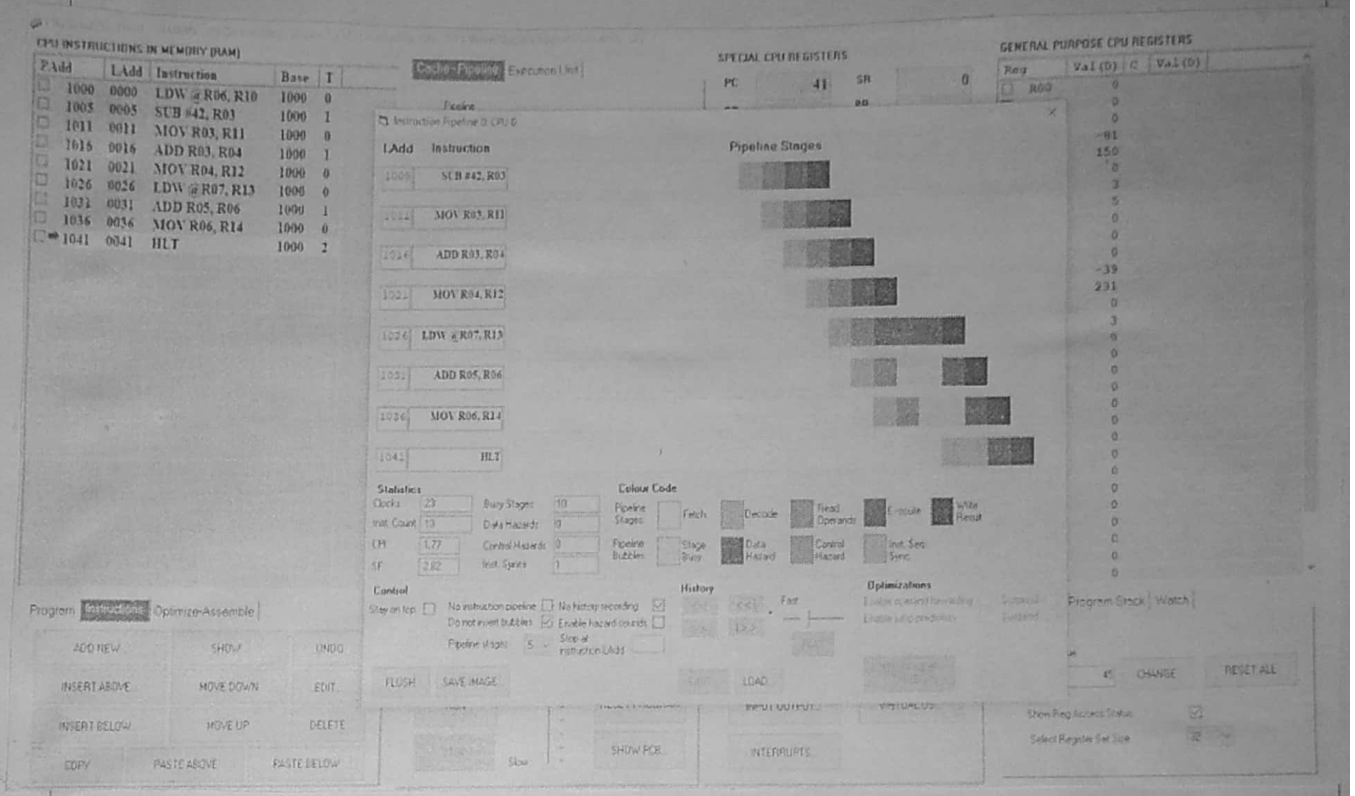


Observations and Snapshots: Take the snap shot of CPU statistics and pipeline design.



Scanned with CamScanner

Snapshots ·



CPU INSTRUCTIONS IN MEMORY (RAM)

| PAdd | LAdd | Instruction | Base | T |
|---|---|---|---|---|
| 1000 | 0000 | LDW @R06, R10 | 1000 | 0 |
| 1005 | 0005 | SUB #42, R03 | 1000 | 1 |
| 1011 | 0011 | MOV R03, R11 | 1000 | 0 |
| 1016 | 0016 | ADD R03, R04 | 1000 | 1 |
| 1021 | 0021 | MOV R04, R12 | 1000 | 0 |
| 1026 | 0026 | LDW @R07, R13 | 1000 | 0 |
| 1031 | 0031 | ADD R05, R06 | 1000 | 1 |
| 1036 | 0036 | MOV R06, R14 | 1000 | 0 |
| 1041 | 0041 | HLT | 1000 | 2 |

Instruction Pipeline 0: CPU 0

| LAdd | Instruction |
|---|---|
| 1005 | SUB #42, R03 |
| 1011 | MOV R03, R11 |
| 1016 | ADD R03, R04 |
| 1021 | MOV R04, R12 |
| 1026 | LDW @R07, R13 |
| 1031 | ADD R05, R06 |
| 1036 | MOV R06, R14 |
| 1041 | HLT |

Statistics

| | | | |
|---|---|---|---|
| Clocks | 23 | Busy Stages | 10 |
| Inst. Count | 13 | Data Hazards | 0 |
| CPI | 1.77 | Control Hazards | 0 |
| SF | 2.82 | Inst. Syncs | 1 |

Control

Stay on top ☐ No instruction pipeline ☐ No history recording ☑

Do not insert bubbles ☑ Enable hazard counts ☐

Pipeline stages 5 Stop at instruction LAdd

SPECIAL CPU REGISTERS

PC 41 SR 0

GENERAL PURPOSE CPU REGISTERS

| Reg | Val (D) |
|---|---|
| R00 | 0 |
| | 0 |
| | 0 |
| | -81 |
| | 150 |
| | 0 |
| | 3 |
| | 5 |
| | 0 |
| | 0 |
| | 0 |
| | -39 |
| | 231 |
| | 0 |
| | 3 |
| | 0 |
| | 0 |
| | 0 |
| | 0 |
| | 0 |
| | 0 |
| | 0 |
| | 0 |
| | 0 |
| | 0 |
| | 0 |
| | 0 |
| | 0 |
| | 0 |
| | 0 |

Snapshots: