Prajwal
1MS18CS091

List out the steps in designing ALU

Step1: Add the two input pins, Drop two East facing input pins on the canvas 4 bits each label A and B ensure that input is 4 bits

Step2: Add the Adder/subtractor and gates Now we add the sub circuits created earlier Select circuits under main project handler folder

Step3: Add the multipliers
Take on of more inputs and generate a single output in logisim multiplexers and under plexer folder click multiplexer icon and drop two of them into canvas

Step4: Add controls
Drop two pins on the canvas north facing with 1 data bit. Label them 0 & 1 respectively

Step5: Add a splitter
Next we add a splitter into our circuit that takes one line from the second multiplexer and split to 4 inputs to an OR gate for a 4 bit ALU

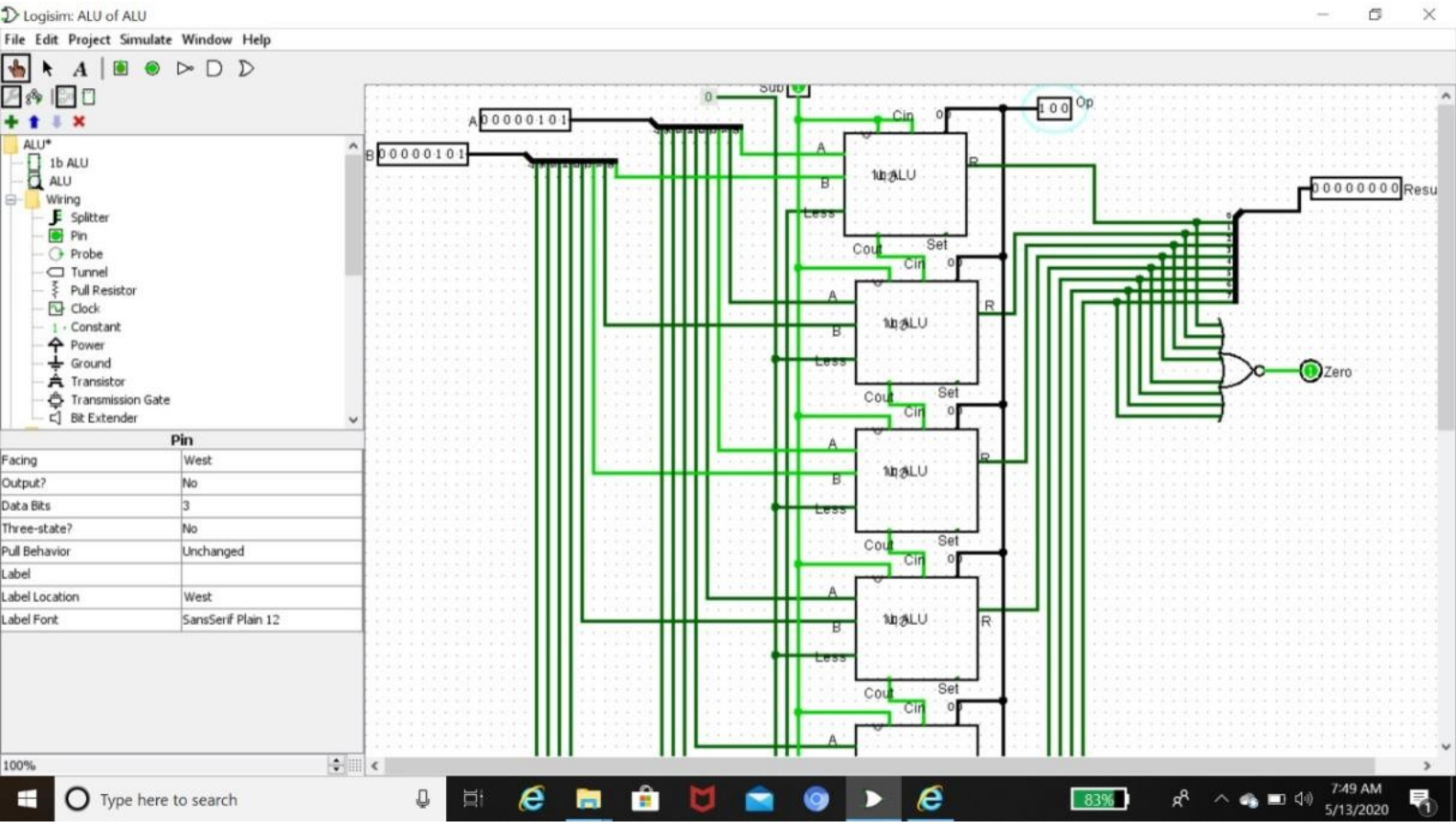
Step6: Add another OR gate and NOT gate Now we add an OR gate after the splitter which has 4 inputs. To right of the OR gate and a NOT gate.

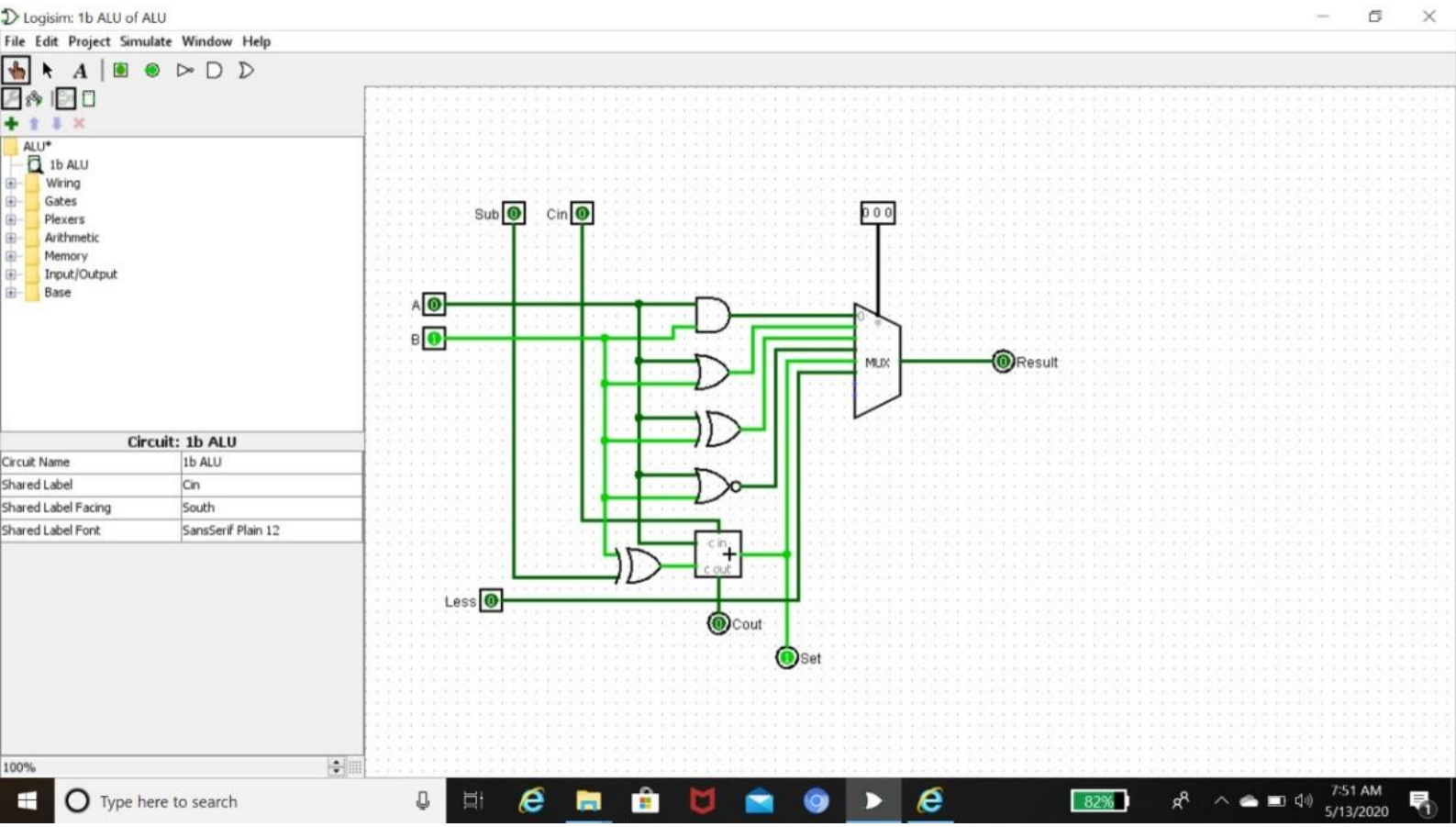
This Arrangement accounts for zero output when all the bits result in zero.

Prajwal
1MS18CS091

The Not gate following The OR gate achieves this
Finally add a single bit pin after the NOT gate to
store the result, label it zero

Step 7: Add a result pin for the Mux
we handled the zeroes carring from the Mux
but we also need to account for valid combinations
inputs from A, B and the control inputs
in


Logisim: ALU of ALU
File Edit Project Simulate Window Help
ALU*
1b ALU
ALU
Wiring
Splitter
Pin
Probe
Tunnel
Pull Resistor
Clock
Constant
Power
Ground
Transistor
Transmission Gate
Bit Extender
Pin
Facing West
Output? No
Data Bits 3
Three-state? No
Pull Behavior Unchanged
Label
Label Location West
Label Font SansSerif Plain 12
100%
A 00000101
B 00000101
Sub
0
Cin
Op
100
1bALU
A
B
Less
Cout
Set
R
00000000 Resu
Zero
Type here to search
83%
7:49 AM
5/13/2020


Logisim: 1b ALU of ALU
File Edit Project Simulate Window Help
ALU*
1b ALU
Wiring
Gates
Plexers
Arithmetic
Memory
Input/Output
Base
Circuit: 1b ALU
Circuit Name 1b ALU
Shared Label Cin
Shared Label Facing South
Shared Label Font SansSerif Plain 12
100%
Sub
Cin
000
A
B
MUX
Result
c in
c out
Less
Cout
Set
Type here to search
82%
7:51 AM
5/13/2020

Praiwd
1MS18CS091

List out the steps in designing memory system

Step1 : Add ram
Select a separate Load and Store operation for RAM

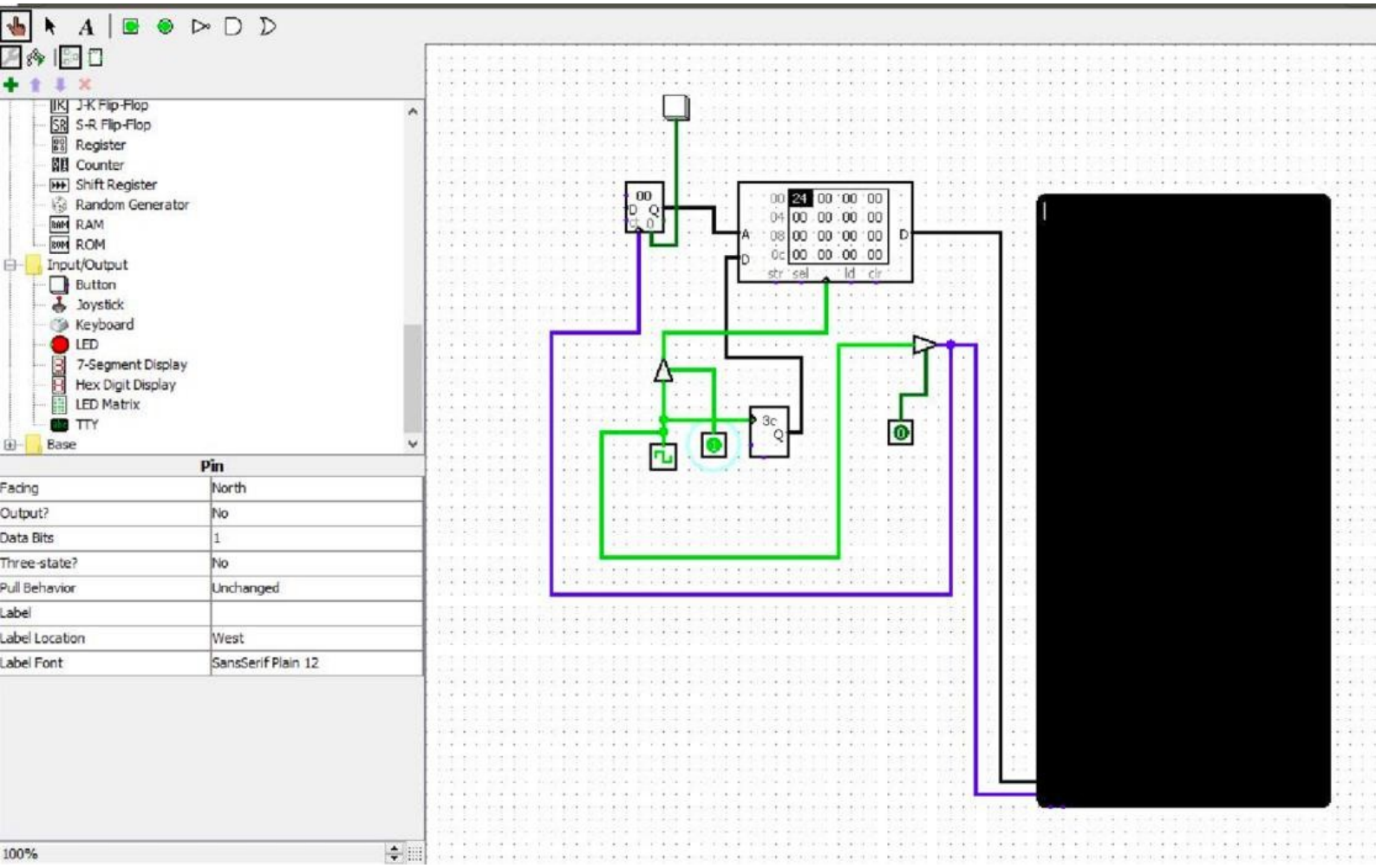
Step2 : Add Counter

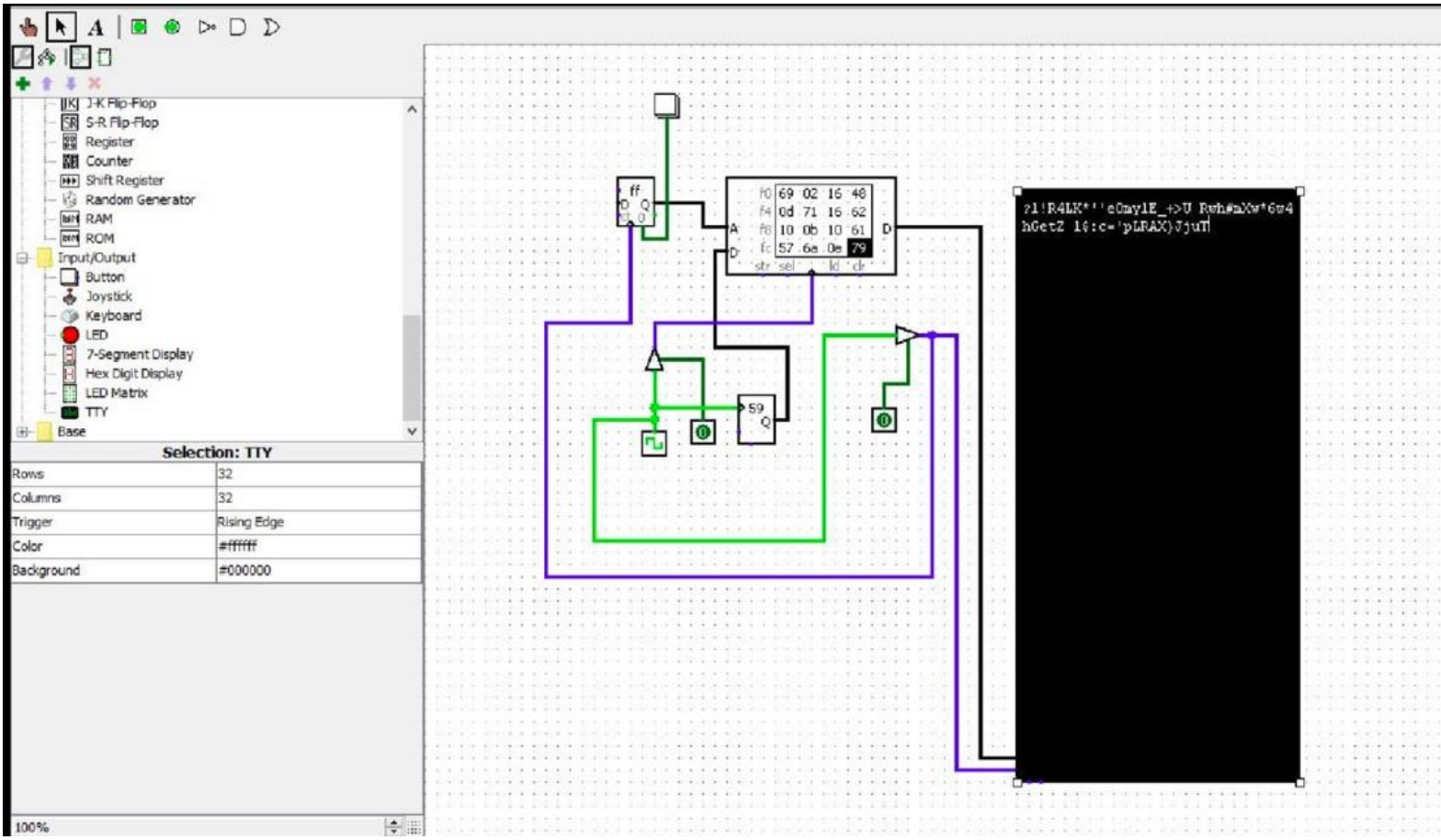
Step3 : Connect Counter Clock, and Controlled Buffer to the RAM

Step 3 : Add TTY
To display Data Read on Memory

step 4 : Add Random Generator
To Generate different address location
Add input and another Controlled Buffer to the Random Generator

Step5 : Add Button
Connect Button to Counter.

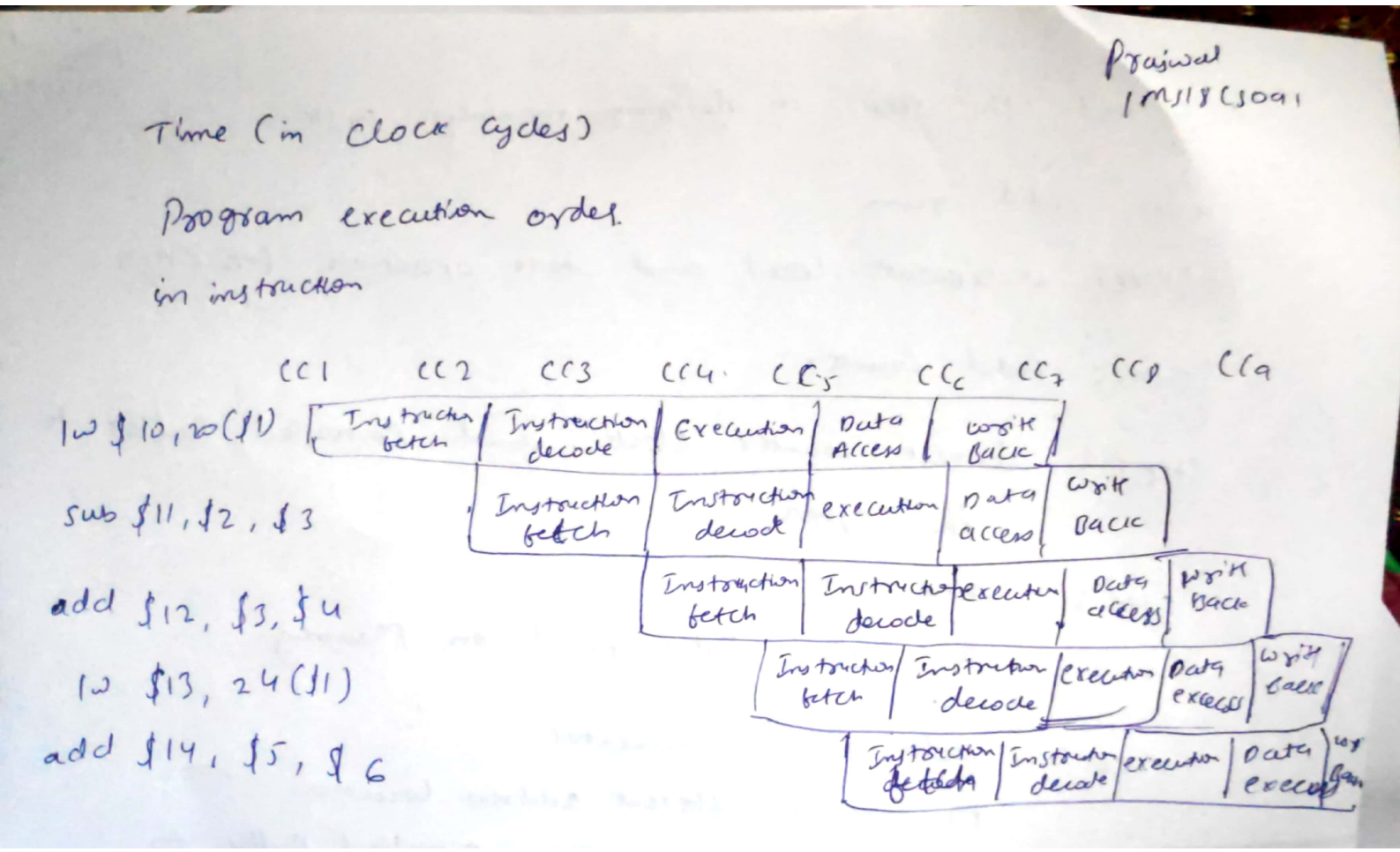

J-K Flip-Flop
S-R Flip-Flop
Register
Counter
Shift Register
Random Generator
RAM
ROM
Input/Output
Button
Joystick
Keyboard
LED
7-Segment Display
Hex Digit Display
LED Matrix
TTY
Base
Pin
Facing North
Output? No
Data Bits 1
Three-state? No
Pull Behavior Unchanged
Label
Label Location West
Label Font SansSerif Plain 12
100%


J-K Flip-Flop
S-R Flip-Flop
Register
Counter
Shift Register
Random Generator
RAM
ROM
Input/Output
Button
Joystick
Keyboard
LED
7-Segment Display
Hex Digit Display
LED Matrix
TTY
Base
Selection: TTY
Rows 32
Columns 32
Trigger Rising Edge
Color #ffffff
Background #000000
100%
f0 69 02 16 48
f4 0d 71 16 62
f8 10 0b 10 61
fc 57 6a 0e 79
A
D
59
Q
?l!R4LK*''eOmylE_+>U Rvh#mXw*6w4
hGetZ 1(:c='pLRAX)JjuT

Prajwal
1MS18CS091

Time (in clock cycles)

Program execution order
in instruction



Prasad
1MS18CS091

Time in clock cycles

CC1 CC2 CC3 CC4 CC5 CC6 CC7

Program execution order

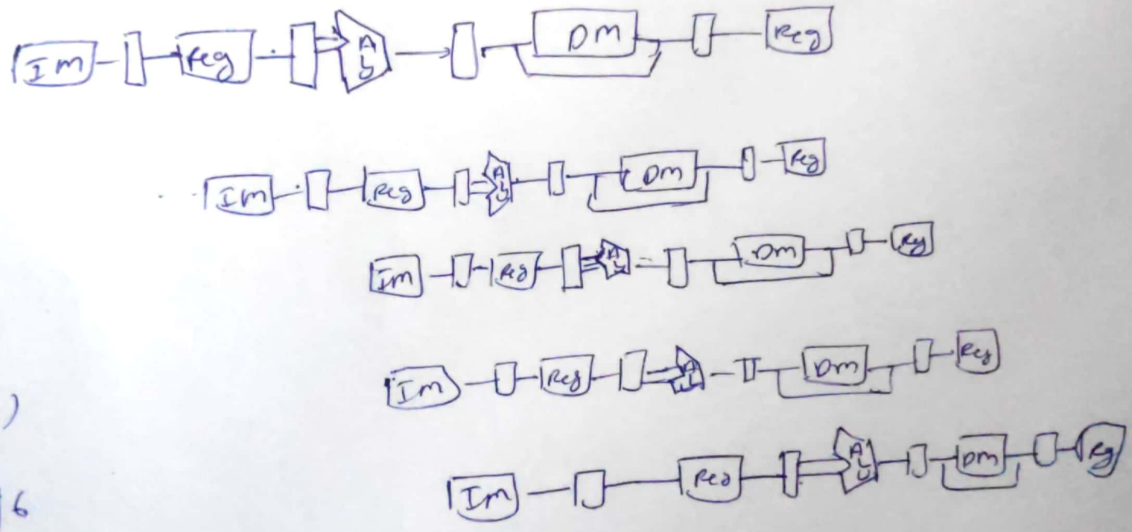
lw $10, 20($1)

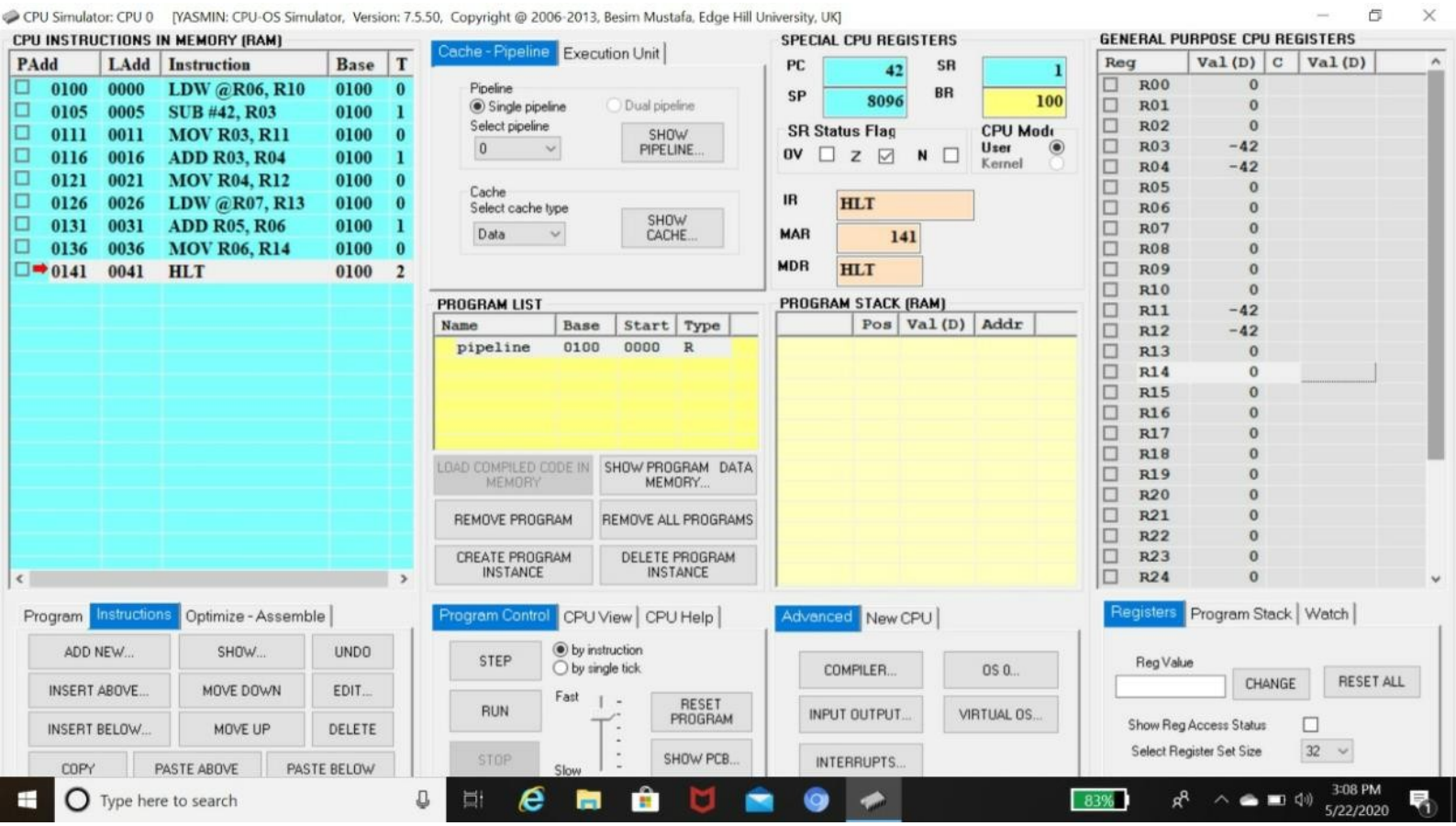
Sub $11, $2, $3

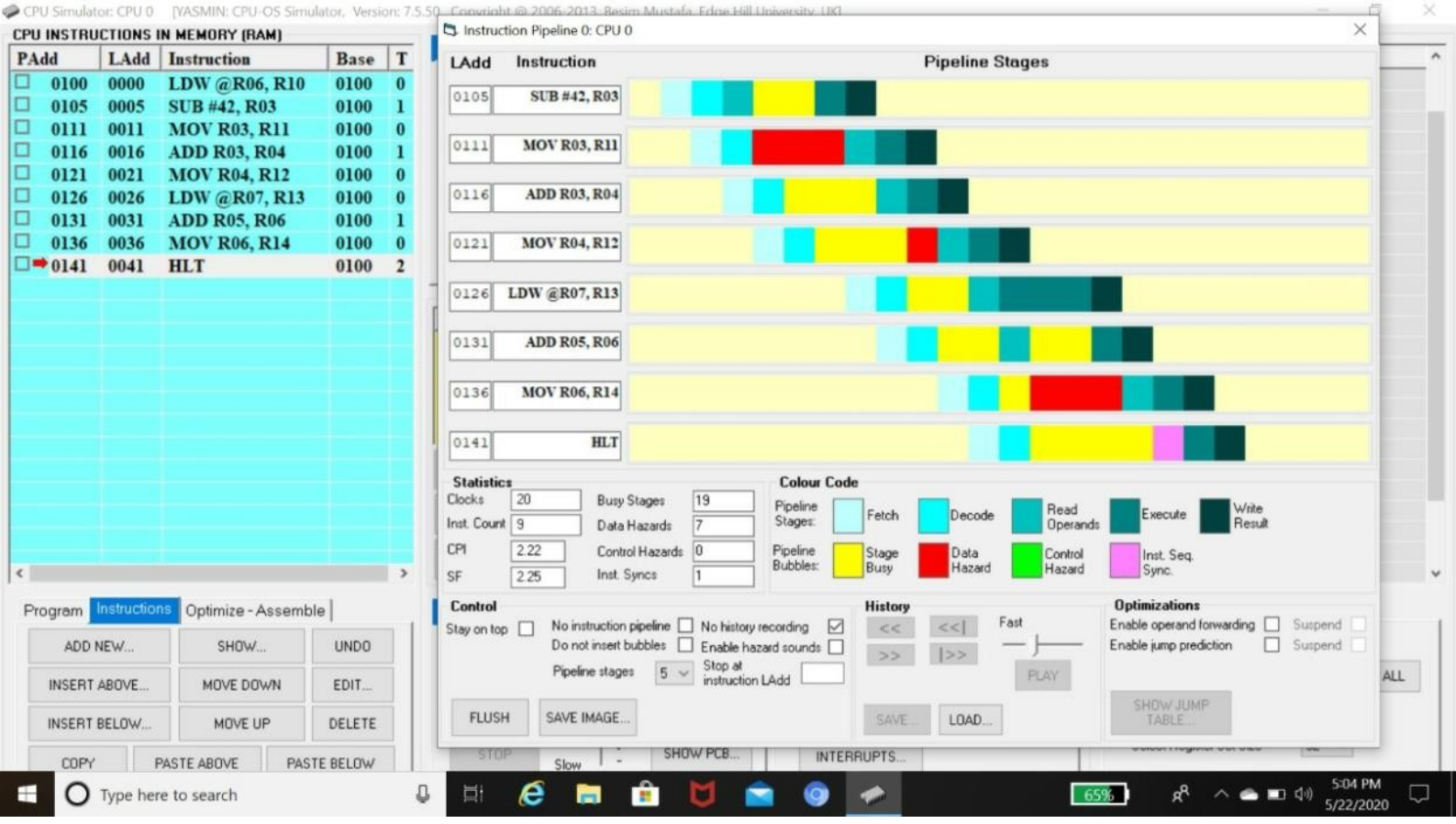
add $12, $3, $4

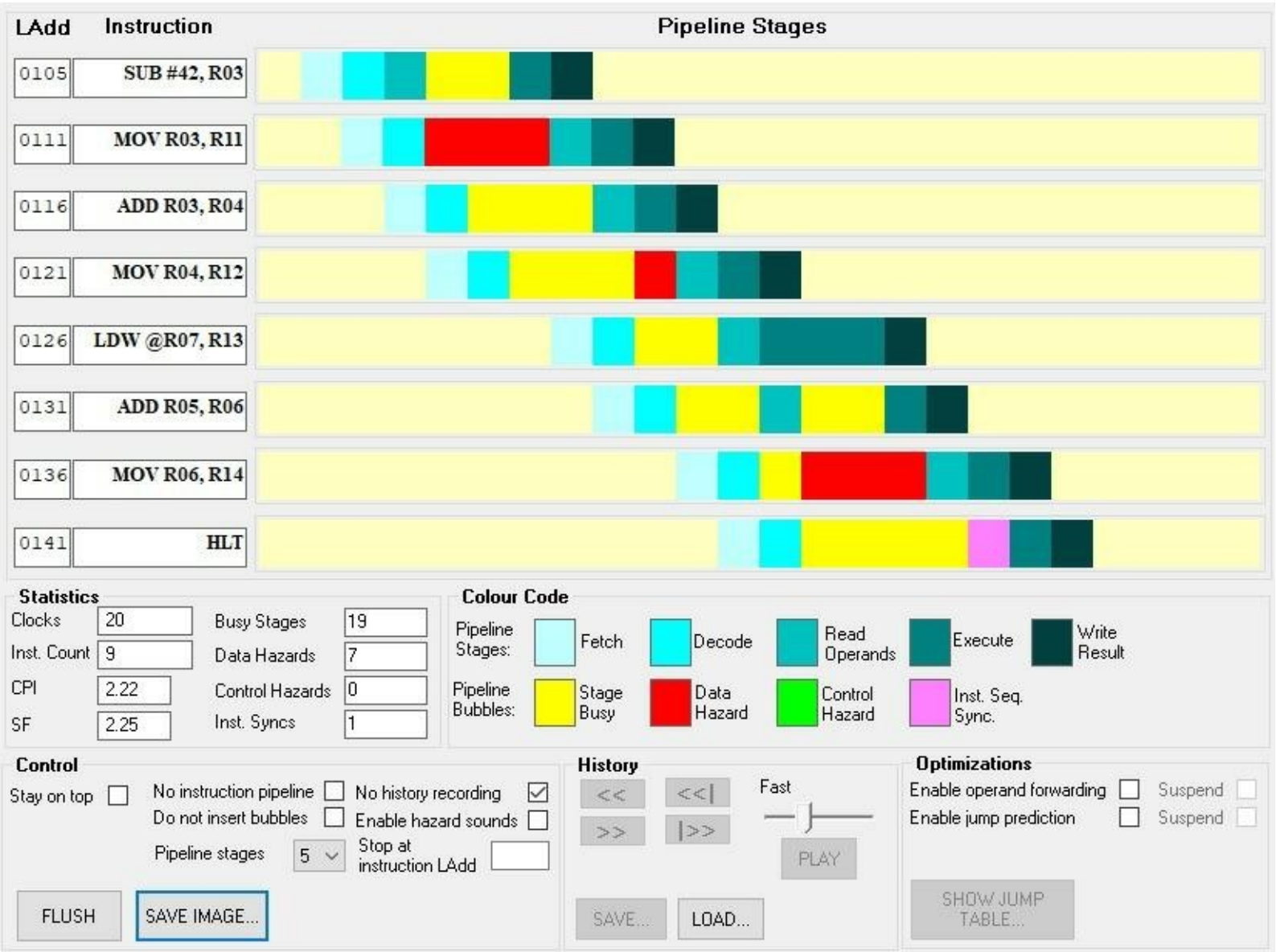
lw $13, 24($1)

add $14, $5, $6




CPU Simulator: CPU 0 [YASMIN: CPU-OS Simulator, Version: 7.5.50, Copyright @ 2006-2013, Besim Mustafa, Edge Hill University, UK]
CPU INSTRUCTIONS IN MEMORY (RAM)
PAdd LAdd Instruction Base T
0100 0000 LDW @R06, R10 0100 0
0105 0005 SUB #42, R03 0100 1
0111 0011 MOV R03, R11 0100 0
0116 0016 ADD R03, R04 0100 1
0121 0021 MOV R04, R12 0100 0
0126 0026 LDW @R07, R13 0100 0
0131 0031 ADD R05, R06 0100 1
0136 0036 MOV R06, R14 0100 0
0141 0041 HLT 0100 2
Program Instructions Optimize - Assemble
ADD NEW... SHOW... UNDO
INSERT ABOVE... MOVE DOWN EDIT...
INSERT BELOW... MOVE UP DELETE
COPY PASTE ABOVE PASTE BELOW
Cache - Pipeline Execution Unit
Pipeline
Single pipeline Dual pipeline
Select pipeline
0
SHOW PIPELINE...
Cache
Select cache type
Data
SHOW CACHE...
PROGRAM LIST
Name Base Start Type
pipeline 0100 0000 R
LOAD COMPILED CODE IN MEMORY
SHOW PROGRAM DATA MEMORY...
REMOVE PROGRAM
REMOVE ALL PROGRAMS
CREATE PROGRAM INSTANCE
DELETE PROGRAM INSTANCE
Program Control CPU View CPU Help
STEP
by instruction
by single tick
RUN
Fast
Slow
RESET PROGRAM
STOP
SHOW PCB...
SPECIAL CPU REGISTERS
PC 42
SR 1
SP 8096
BR 100
SR Status Flag
OV Z N
CPU Mode
User
Kernel
IR HLT
MAR 141
MDR HLT
PROGRAM STACK (RAM)
Pos Val (D) Addr
Advanced New CPU
COMPILER...
OS 0...
INPUT OUTPUT...
VIRTUAL OS...
INTERRUPTS...
GENERAL PURPOSE CPU REGISTERS
Reg Val (D) C Val (D)
R00 0
R01 0
R02 0
R03 -42
R04 -42
R05 0
R06 0
R07 0
R08 0
R09 0
R10 0
R11 -42
R12 -42
R13 0
R14 0
R15 0
R16 0
R17 0
R18 0
R19 0
R20 0
R21 0
R22 0
R23 0
R24 0
Registers Program Stack Watch
Reg Value
CHANGE
RESET ALL
Show Reg Access Status
Select Register Set Size
32
Type here to search
83%
3:08 PM
5/22/2020


CPU Simulator: CPU 0 [YASMIN: CPU-OS Simulator, Version: 7.5.50, Copyright @ 2006-2013, Besim Mustafa, Edge Hill University, UK]
CPU INSTRUCTIONS IN MEMORY (RAM)
PAdd LAdd Instruction Base T
0100 0000 LDW @R06, R10 0100 0
0105 0005 SUB #42, R03 0100 1
0111 0011 MOV R03, R11 0100 0
0116 0016 ADD R03, R04 0100 1
0121 0021 MOV R04, R12 0100 0
0126 0026 LDW @R07, R13 0100 0
0131 0031 ADD R05, R06 0100 1
0136 0036 MOV R06, R14 0100 0
0141 0041 HLT 0100 2
Program Instructions Optimize - Assemble
ADD NEW... SHOW... UNDO
INSERT ABOVE... MOVE DOWN EDIT...
INSERT BELOW... MOVE UP DELETE
COPY PASTE ABOVE PASTE BELOW
Instruction Pipeline 0: CPU 0
LAdd Instruction Pipeline Stages
0105 SUB #42, R03
0111 MOV R03, R11
0116 ADD R03, R04
0121 MOV R04, R12
0126 LDW @R07, R13
0131 ADD R05, R06
0136 MOV R06, R14
0141 HLT
Statistics
Clocks 20
Busy Stages 19
Inst. Count 9
Data Hazards 7
CPI 2.22
Control Hazards 0
SF 2.25
Inst. Syncs 1
Colour Code
Pipeline Stages: Fetch Decode Read Operands Execute Write Result
Pipeline Bubbles: Stage Busy Data Hazard Control Hazard Inst. Seq. Sync.
Control
Stay on top
No instruction pipeline
No history recording
Do not insert bubbles
Enable hazard sounds
Pipeline stages 5
Stop at instruction LAdd
FLUSH
SAVE IMAGE...
History
<< <<| >> |>>
Fast
PLAY
SAVE...
LOAD...
Optimizations
Enable operand forwarding Suspend
Enable jump prediction Suspend
SHOW JUMP TABLE...
Type here to search
65%
5:04 PM
5/22/2020



Statistics

Clocks	20	Busy Stages	19
Inst. Count	9	Data Hazards	7
CPI	2.22	Control Hazards	0
SF	2.25	Inst. Syncs	1

Colour Code

Pipeline Stages: Fetch, Decode, Read Operands, Execute, Write Result

Pipeline Bubbles: Stage Busy, Data Hazard, Control Hazard, Inst. Seq. Sync.

Control

Stay on top ☐

No instruction pipeline ☐ No history recording ☑

Do not insert bubbles ☐ Enable hazard sounds ☐

Pipeline stages 5 Stop at instruction LAdd

FLUSH | SAVE IMAGE...

History

<< <<| >> |>>

Fast

PLAY

SAVE... | LOAD...

Optimizations

Enable operand forwarding ☐ Suspend ☐

Enable jump prediction ☐ Suspend ☐

SHOW JUMP TABLE...