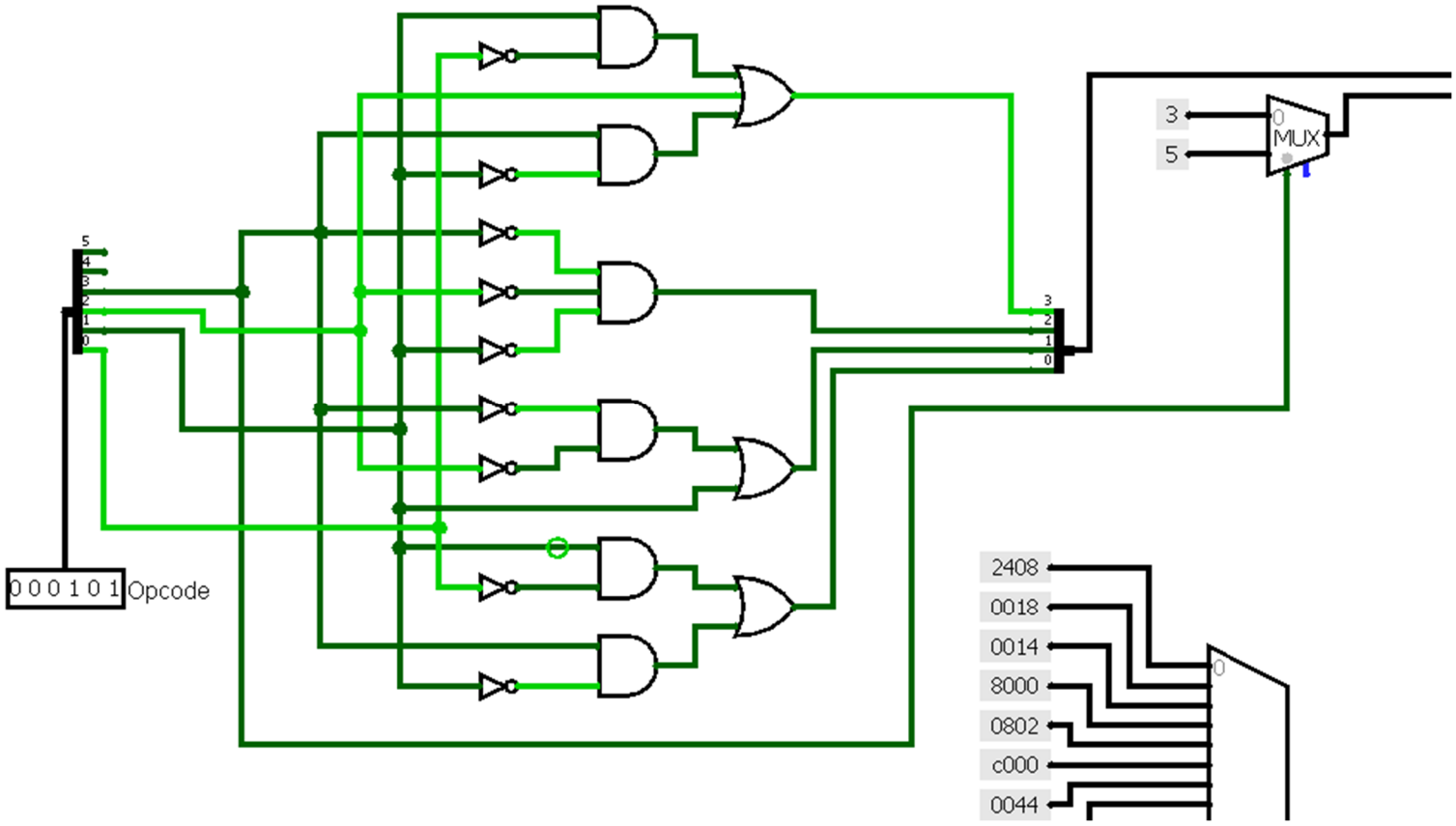
Computer Architecture Lab – CS322

Name: M. Maheeth Reddy	**Roll No.**: 1801CS31	**Date**: 13 November 2020

Lab 9 – Implement a Multi-Cycle Processor

Task 1:

In the given multi-cycle processor, there is an error in the **Control** block. The Control block uses the opcode in the instruction to identify the next state of the Control unit through Boolean algebra implemented in the form of gates. The error exists in this Boolean logic which causes invalid state transitions of the Control Unit. So, I have corrected it as shown.

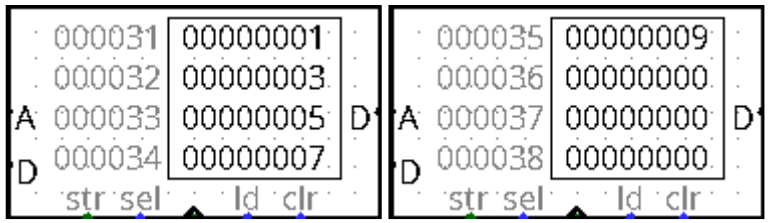


Task 2:

Program to store sum of numbers present in memory locations 49-53 at location 80

Instruction	Machine Code	Comments
lw $t1, 49($0)	0x8c090031	Load into t1 from loc 49
lw $t2, 50($0)	0x8c0a0032	Load into t2 from loc 50
lw $t3, 51($0)	0x8c0b0033	Load into t3 from loc 51
lw $t4, 52($0)	0x8c0c0034	Load into t4 from loc 52
lw $t5, 53($0)	0x8c0d0035	Load into t5 from loc 53
addi $t6, $0, 0	0x200e0000	Clear t6
add $t6, $t1, $t2	0x012a7020	Store sum of t1,t2 in t6
add $t6, $t6, $t3	0x01cb7020	Store sum of t6,t3 in t6
add $t6, $t6, $t4	0x01cc7020	Store sum of t6,t4 in t6
add $t6, $t6, $t5	0x01cd7020	Store sum of t6,t5 in t6
sw $t6, 80($0)	0xac0e0050	Store t6 value at loc 80

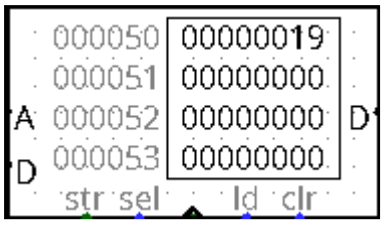
Values in Memory from addresses 49 to 53 (0x31 to 0x35):



Numbers are 1,3,5,7,9

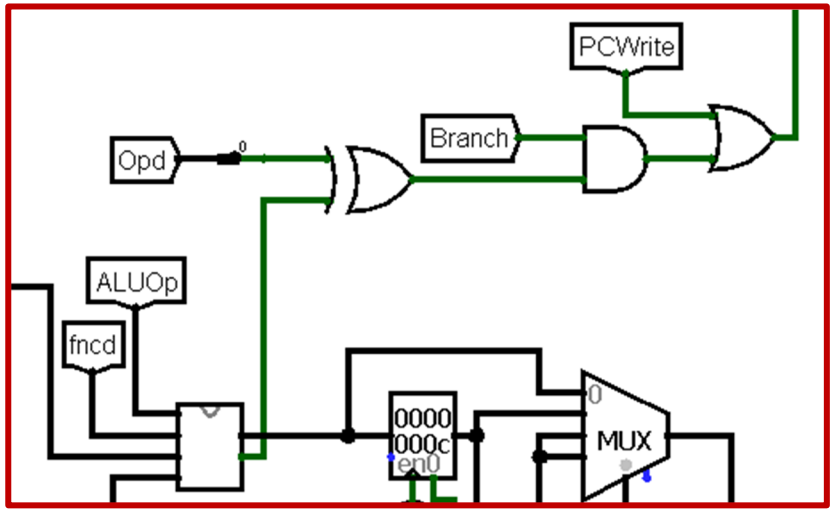
Sum is 25 which is 0x19 in hexadecimal

Sum stored in address 80 (0x50):



Task 3:

I have added the ***Branch on Not Equal*** instruction **(BNE)** to the given architecture. Instruction **Branch on Equal (BEQ)** is already implemented. It uses the Equal Flag from the ALU to check for equality between the values in the source registers. So, to incorporate BNE instruction into the given architecture, I did the following modification:



Program for demonstration:

Labels	Instruction	Machine Code	Comment
	addi $4, $0, 3	0x20040003	
	bne $4, $0, **here**	0x14800002	Jump to label here if value in $4 is not equal to $0. Offset is 2
	addi $1, $0, 1	0x20010001	
	addi $2, $0, 2	0x20020002	
here	addi $3, $0, 3	0x20030003	
	addi $5, $0, 5	0x20050005	
	bne $4, $3, **there**	0x14830002	Jump to label here if value in $4 is not equal to $3. Offset is 2
	addi $6, $0, 6	0x20060006	
	addi $7, $0, 7	0x20070007	
there	addi $8, $0, 8	0x20080008	