Lab 4 – R-type Datapath

CECS 341 – Computer Architecture & Organization

Kenry Yu, 028210726

Olenka Bilinska, 028897191

Garret Towner, 028303091

Professor: Mandy He

California State University, Long Beach

College of Engineering

1250 Bellflower Blvd, Long Beach, CA 90840

March 18, 2022

Goal/Objective:

The objective of this lab is to design a MIPS Datapath for R-type instructions.

Technical Description/Steps:

To accomplish the objective, there are four design modules to design (program counter, program counter adder, control, and Datapath), three design modules to utilize (instruction memory, register file, and ALU), two data files to import ("imem" and "regfile32"), and a testbench to simulate the process.

Method for designing the Datapath:

To design the Datapath, all other design modules should be implemented first. After designing other modules, we initialized wires to connect the modules in the Datapath.

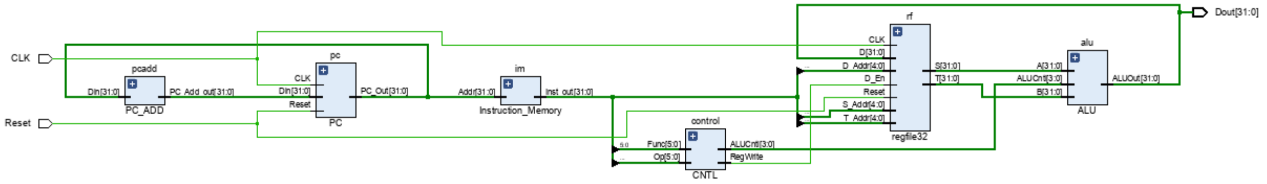
Modification of the ALU:

The lab required us to add two new operators to our ALU module, the SLT signed and SLT unsigned. To accomplish this feature, we used if else statement of Verilog to determine the output value. For SLT signed operator, we also added the 2's complement to compute the correct output.

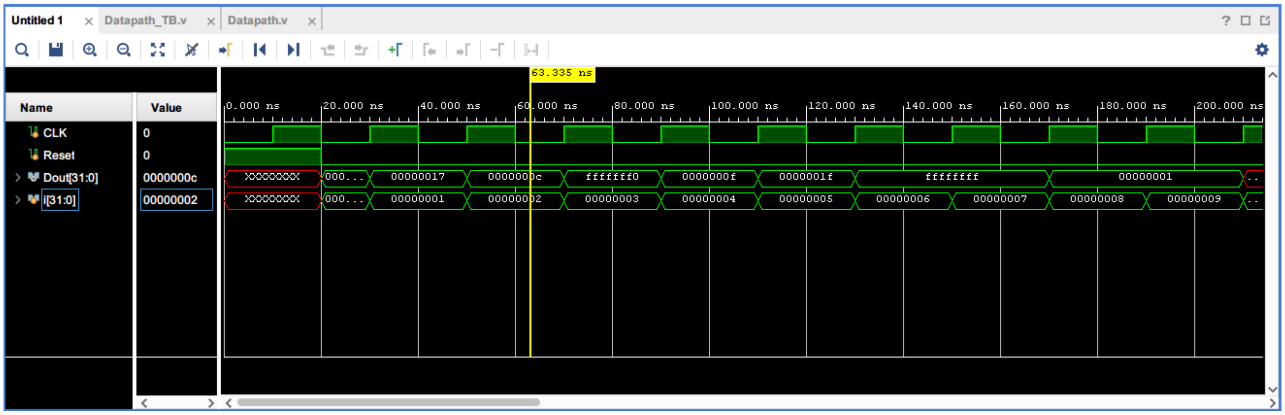
Testbench:

In the testbench, we import the instructions and register file from the data files. Then, we use a task that would utilize the data from the data files to output the results of the instruction into console log.

Results:

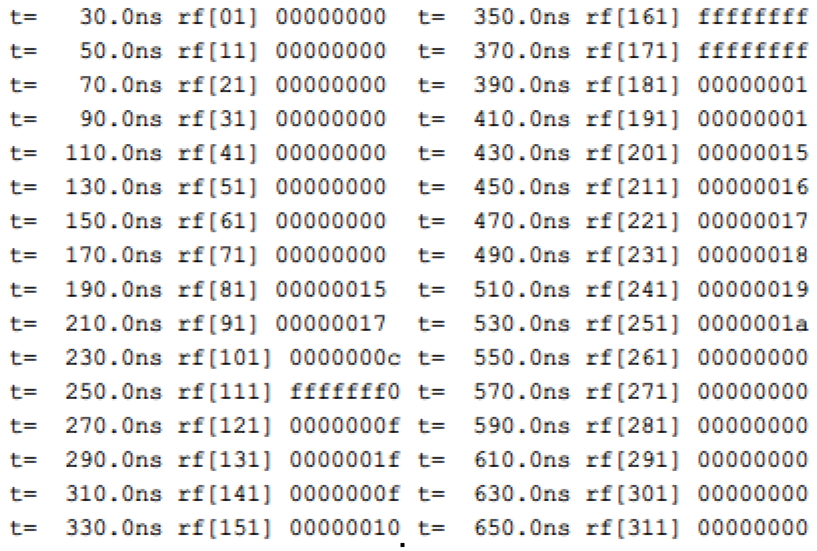


This is the schematic of the Datapath module. The schematic shows that wires have connected all the modules to work together.



Name	Value
CLK	0
Reset	0
Dout[31:0]	0000000c
i[31:0]	00000002

The waveform shows the clock is pulsing as desired. At the rising edge of the clock, the output changes according to different instructions from the data files.

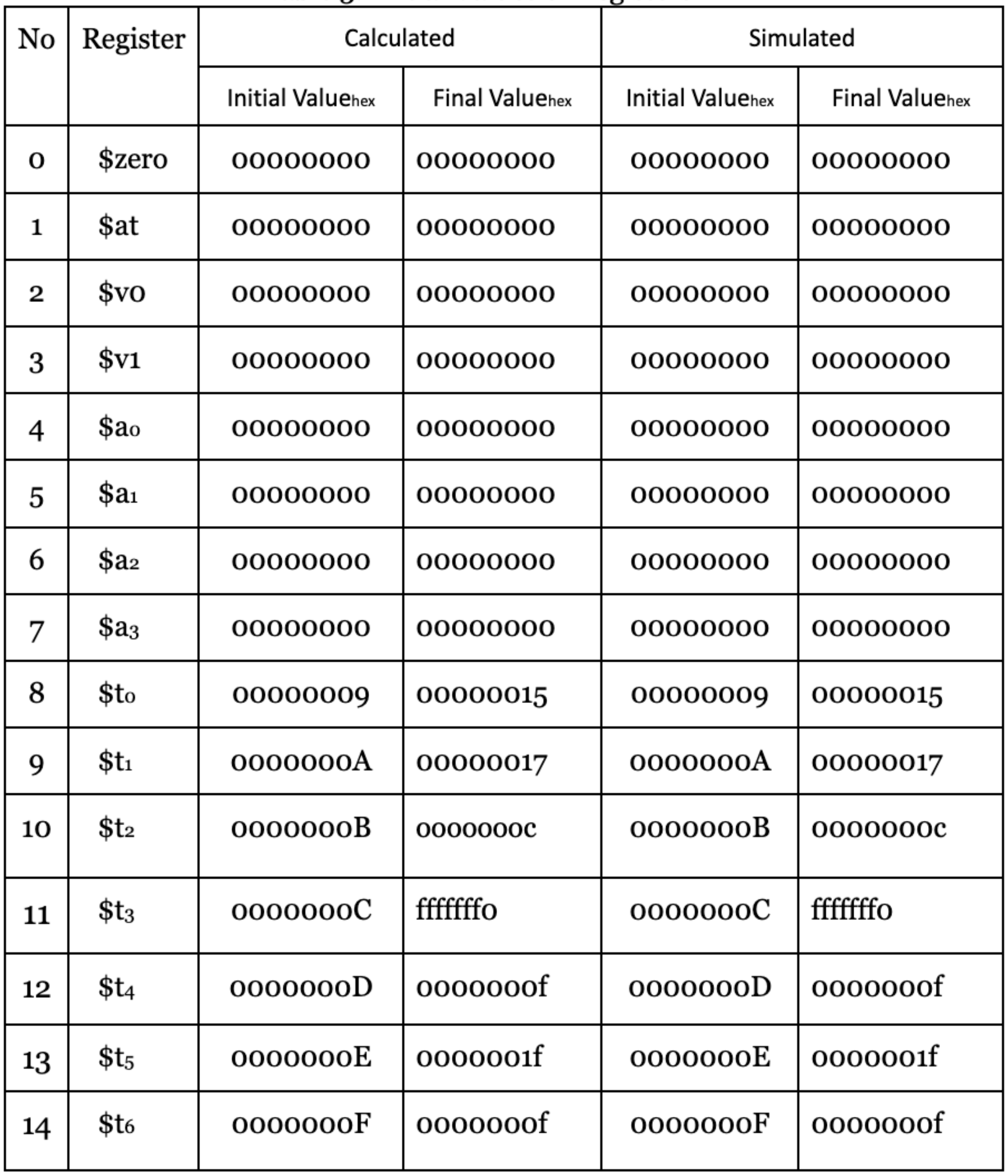

```
t=   30.0ns rf[01] 00000000  t=  350.0ns rf[161] ffffffff
t=   50.0ns rf[11] 00000000  t=  370.0ns rf[171] ffffffff
t=   70.0ns rf[21] 00000000  t=  390.0ns rf[181] 00000001
t=   90.0ns rf[31] 00000000  t=  410.0ns rf[191] 00000001
t=  110.0ns rf[41] 00000000  t=  430.0ns rf[201] 00000015
t=  130.0ns rf[51] 00000000  t=  450.0ns rf[211] 00000016
t=  150.0ns rf[61] 00000000  t=  470.0ns rf[221] 00000017
t=  170.0ns rf[71] 00000000  t=  490.0ns rf[231] 00000018
t=  190.0ns rf[81] 00000015  t=  510.0ns rf[241] 00000019
t=  210.0ns rf[91] 00000017  t=  530.0ns rf[251] 0000001a
t=  230.0ns rf[101] 0000000c t=  550.0ns rf[261] 00000000
t=  250.0ns rf[111] fffffff0 t=  570.0ns rf[271] 00000000
t=  270.0ns rf[121] 0000000f t=  590.0ns rf[281] 00000000
t=  290.0ns rf[131] 0000001f t=  610.0ns rf[291] 00000000
t=  310.0ns rf[141] 0000000f t=  630.0ns rf[301] 00000000
t=  330.0ns rf[151] 00000010 t=  650.0ns rf[311] 00000000
```

In the console log, the saved registers are displayed and proof that the Datapath program is working and generating correct output as intended.

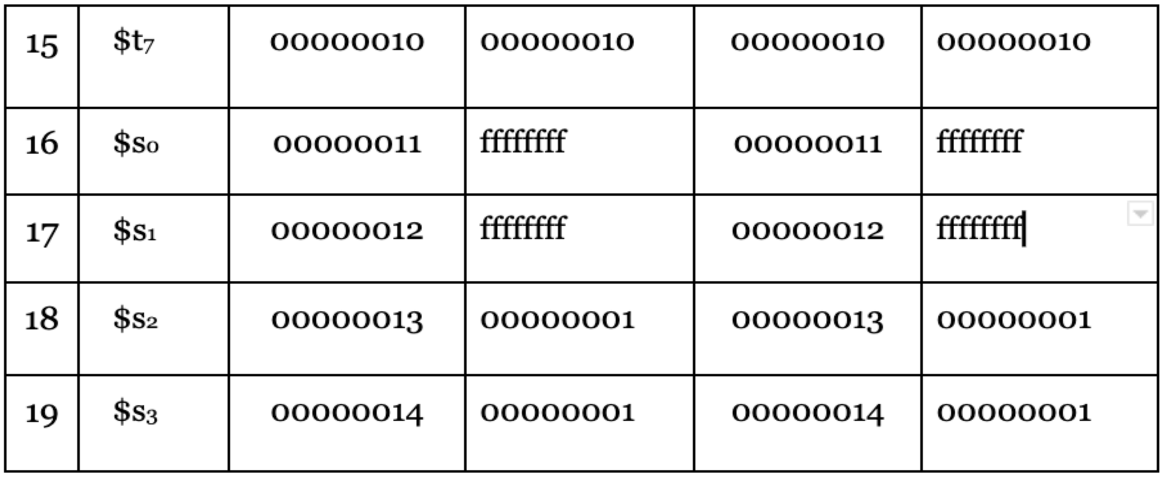
Conclusion:

In this lab, we learned how to wire multiple modules together, read data from data files, and create tasks in the module. In addition, we also strengthened our knowledge in MIPS because we learned how each module worked together to compute instructions. We encounter some errors in setting the clock to pulse as intended, but we quickly notice we have the wrong symbol in the code and fix it. After the changes, the outputs are successfully displayed.

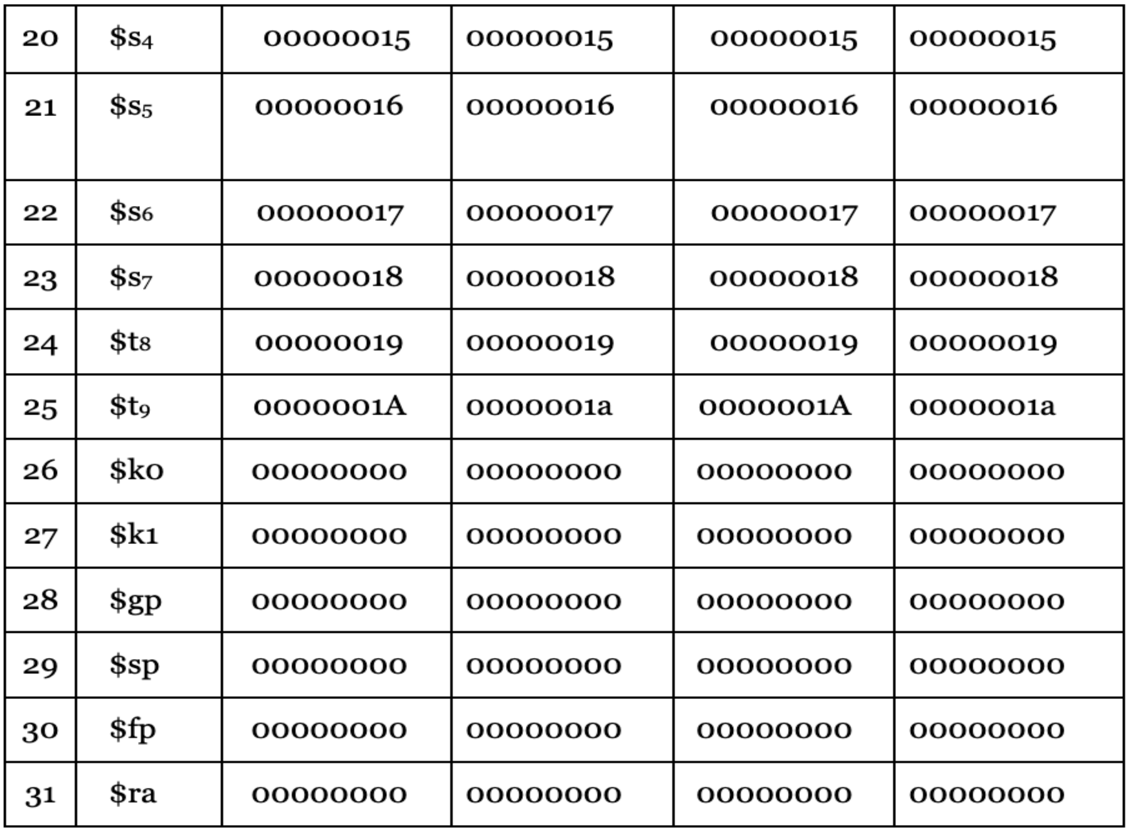
Table 3. Initial Values of Registers



No	Register	Calculated		Simulated	
		Initial Value$_{hex}$	Final Value$_{hex}$	Initial Value$_{hex}$	Final Value$_{hex}$
0	$zero	00000000	00000000	00000000	00000000
1	$at	00000000	00000000	00000000	00000000
2	$v0	00000000	00000000	00000000	00000000
3	$v1	00000000	00000000	00000000	00000000
4	$\$a_0$	00000000	00000000	00000000	00000000
5	$\$a_1$	00000000	00000000	00000000	00000000
6	$\$a_2$	00000000	00000000	00000000	00000000
7	$\$a_3$	00000000	00000000	00000000	00000000
8	$\$t_0$	00000009	00000015	00000009	00000015
9	$\$t_1$	0000000A	00000017	0000000A	00000017
10	$\$t_2$	0000000B	0000000c	0000000B	0000000c
11	$\$t_3$	0000000C	fffffff0	0000000C	fffffff0
12	$\$t_4$	0000000D	0000000f	0000000D	0000000f
13	$\$t_5$	0000000E	0000001f	0000000E	0000001f
14	$\$t_6$	0000000F	0000000f	0000000F	0000000f

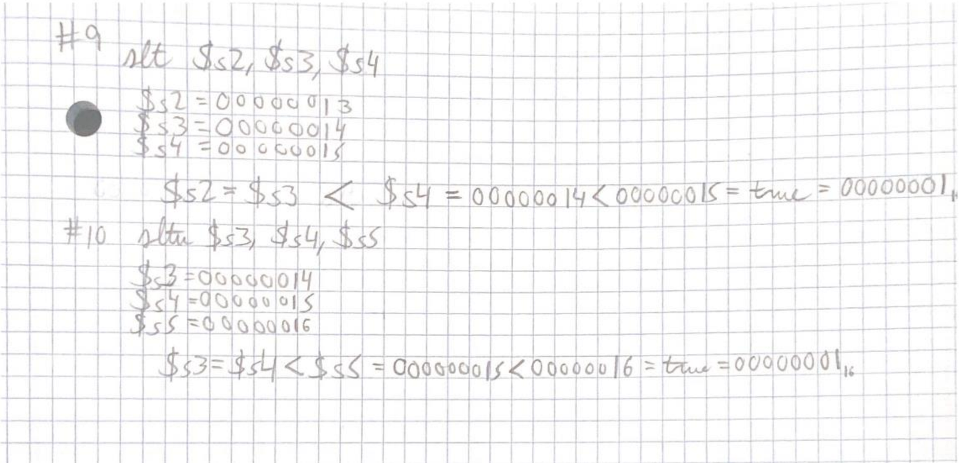


15	$t7	00000010	00000010	00000010	00000010
16	$s0	00000011	ffffffff	00000011	ffffffff
17	$s1	00000012	ffffffff	00000012	ffffffff
18	$s2	00000013	00000001	00000013	00000001
19	$s3	00000014	00000001	00000014	00000001



20	$s4	00000015	00000015	00000015	00000015
21	$s5	00000016	00000016	00000016	00000016
22	$s6	00000017	00000017	00000017	00000017
23	$s7	00000018	00000018	00000018	00000018
24	$t8	00000019	00000019	00000019	00000019
25	$t9	0000001A	0000001a	0000001A	0000001a
26	$k0	00000000	00000000	00000000	00000000
27	$k1	00000000	00000000	00000000	00000000
28	$gp	00000000	00000000	00000000	00000000
29	$sp	00000000	00000000	00000000	00000000
30	$fp	00000000	00000000	00000000	00000000
31	$ra	00000000	00000000	00000000	00000000

Hand solutions:



#9 slt $\$s2, \$s3, \$s4$

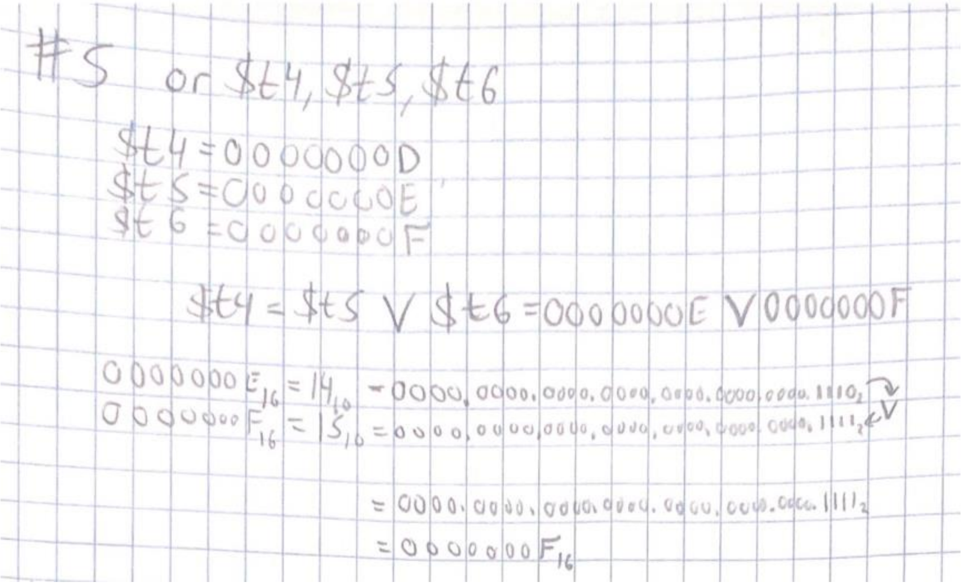
$\$s2 = 00000013$
$\$s3 = 00000014$
$\$s4 = 00000015$

$\$s2 = \$s3 < \$s4 = 00000014 < 00000015 = \text{tme} = 00000001$

#10 sltu $\$s3, \$s4, \$s5$

$\$s3 = 00000014$
$\$s4 = 00000015$
$\$s5 = 00000016$

$\$s3 = \$s4 < \$s5 = 000000015 < 00000016 = \text{tme} = 00000001_{16}$



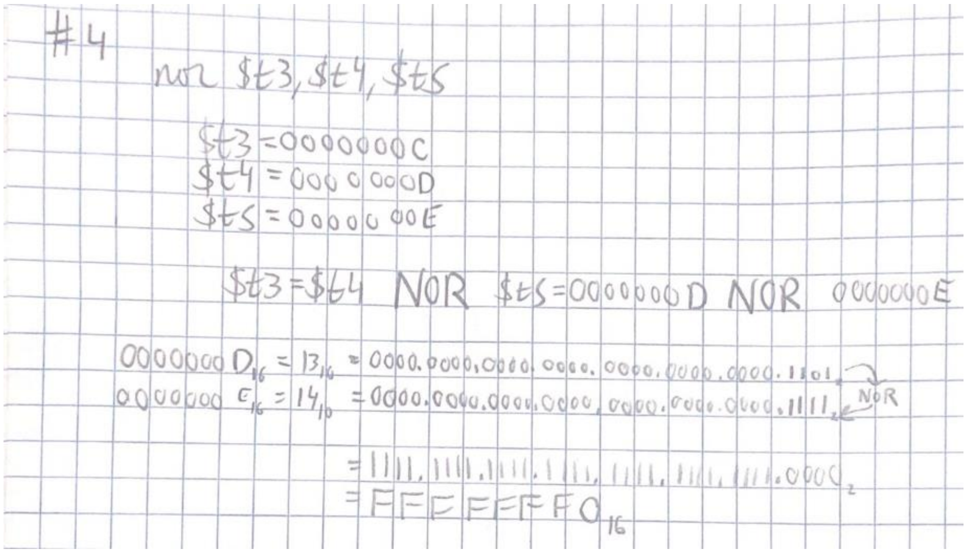
#5 or $\$t4, \$t5, \$t6$

$\$t4 = 0000000D$
$\$t5 = 0000000E$
$\$t6 = 0000000F$

$\$t4 = \$t5 \vee \$t6 = 0000000E \vee 0000000F$

$0000000E_{16} = 14_{10} = 0000,0000.0000.0000,0000.0000.0000.1110_2$
$0000000F_{16} = 15_{10} = 0000,0000,0000,0000,0000,0000.0000.1111_2 \leftarrow \vee$

$= 0000.0000.0000.0000.0000,0000.0000.1111_2$
$= 0000000F_{16}$



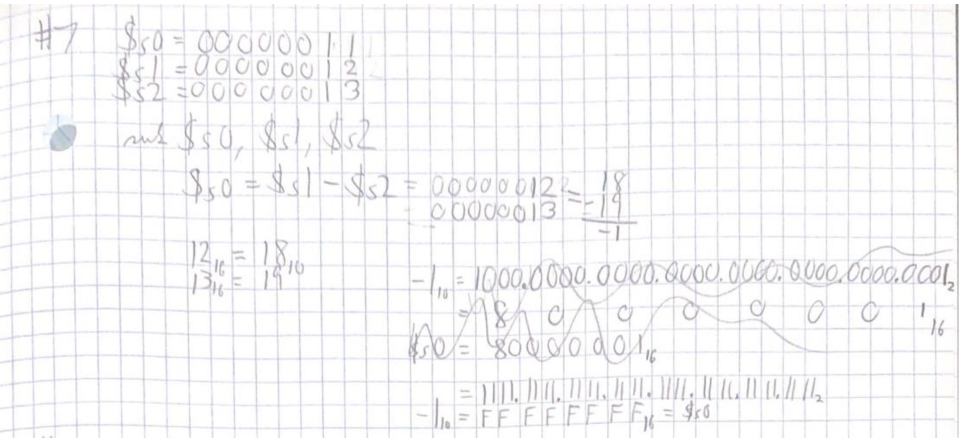
#4 nor $\$t3, \$t4, \$t5$

$\$t3 = 0000000C$
$\$t4 = 0000000D$
$\$t5 = 0000000E$

$\$t3 = \$t4 \text{ NOR } \$t5 = 0000000D \text{ NOR } 0000000E$

$0000000D_{16} = 13_{16} = 0000.0000.0000.0000.0000.0000.0000.1101_2$
$0000000E_{16} = 14_{10} = 0000.0000.0000.0000,0000.0000.0000.1111_2 \leftarrow \text{NOR}$

$= 1111.1111.1111.1111,1111.1111.1111.0000_2$
$= FFFFFFF0_{16}$



#7 $\$s0 = 00000011$

$\$s1 = 00000012$

$\$s2 = 00000013$

sub $s0, $s1, $s2

$\$s0 = \$s1 - \$s2 = 00000012 - 00000013 = 18 - 19 = -1$

$12_{16} = 18_{10}$

$13_{16} = 19_{10}$

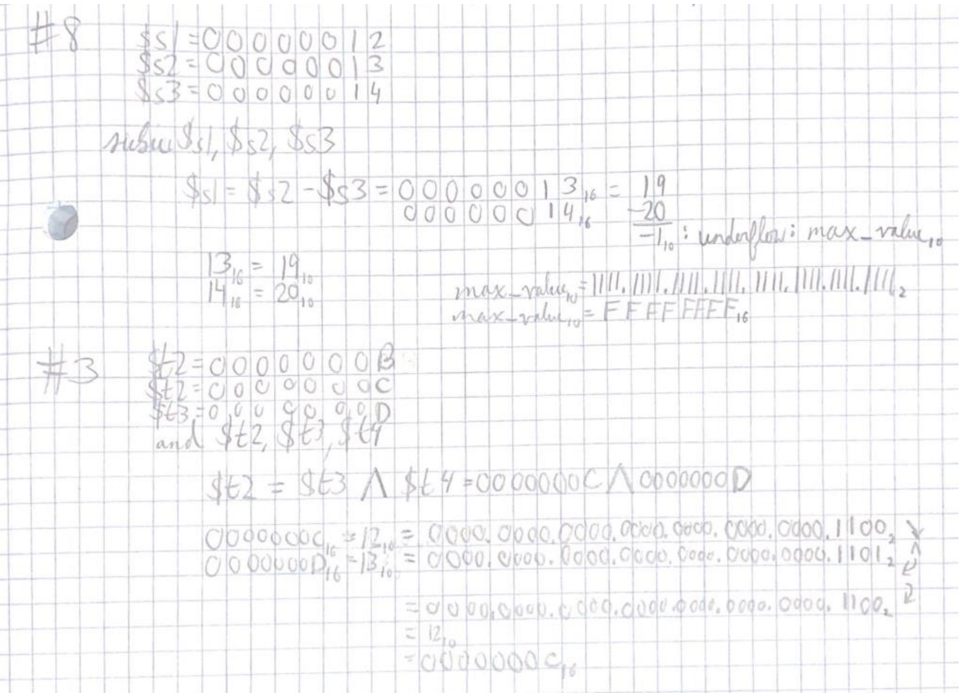
$-1_{10} = 1000.0000.0000.0000.0000.0000.0000.0001_2$

$= 8\ 0\ 0\ 0\ 0\ 0\ 0\ 1_{16}$

$\$s0 = 80000001_{16}$

$= 1111.1111.1111.1111.1111.1111.1111.1111_2$

$-1_{10} = FFFFFFFF_{16} = \$s0$



#8 $\$s1 = 00000012$

$\$s2 = 00000013$

$\$s3 = 00000014$

subu $s1, $s2, $s3

$\$s1 = \$s2 - \$s3 = 00000013_{16} - 00000014_{16} = 19 - 20 = -1_{10}$: underflow : max_value_{10}

$13_{16} = 19_{10}$

$14_{16} = 20_{10}$

$max\_value_{10} = 1111.1111.1111.1111.1111.1111.1111.1111_2$

$max\_value_{10} = FFFFFFFF_{16}$

#3 $\$t2 = 0000000B$

$\$t2 = 0000000C$

$\$t3 = 0000000D$

and $t2, $t3, $t4

$\$t2 = \$t3 \wedge \$t4 = 0000000C \wedge 0000000D$

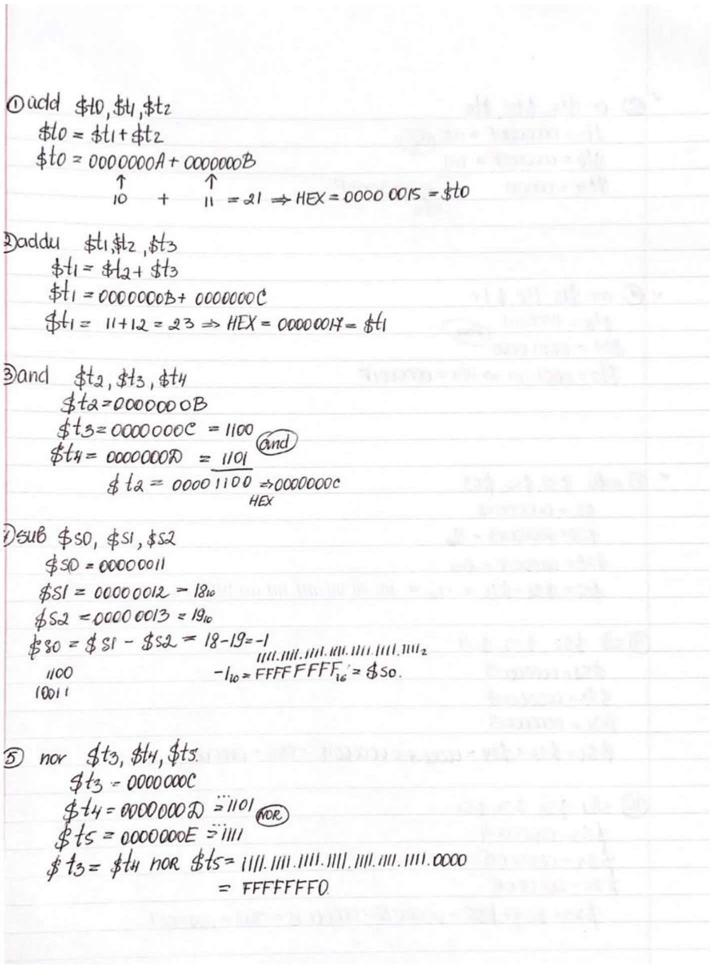
$0000000C_{16} = 12_{10} = 0000.0000.0000.0000.0000.0000.0000.1100_2$

$0000000D_{16} = 13_{10} = 0000.0000.0000.0000.0000.0000.0000.1101_2$

$= 0000.0000.0000.0000.0000.0000.0000.1100_2$

$= 12_{10}$

$= 0000000C_{16}$



① add \$t0, \$t1, \$t2

\$t0 = \$t1 + \$t2

\$t0 = 0000000A + 0000000B

$\overset{\uparrow}{10} + \overset{\uparrow}{11}$ = 21 ⇒ HEX = 0000 0015 = \$t0

② addu \$t1, \$t2, \$t3

\$t1 = \$t2 + \$t3

\$t1 = 0000000B + 0000000C

\$t1 = 11+12 = 23 ⇒ HEX = 00000017 = \$t1

③ and \$t2, \$t3, \$t4

\$t2 = 0000000B

\$t3 = 0000000C = 1100

\$t4 = 0000000D = 1101 and

\$t2 = 0000 1100 ⇒ 0000000C

HEX

④ sub \$s0, \$s1, \$s2

\$s0 = 00000011

\$s1 = 00000012 = 18_{10}

\$s2 = 0000 0013 = 19_{10}

\$s0 = \$s1 − \$s2 = 18−19 = −1

$1111.1111.1111.1111.1111.1111.1111_2$

1100

10011

-1_{10} = $FFFFFFFF_{16}$ = \$s0.

⑤ nor \$t3, \$t4, \$t5.

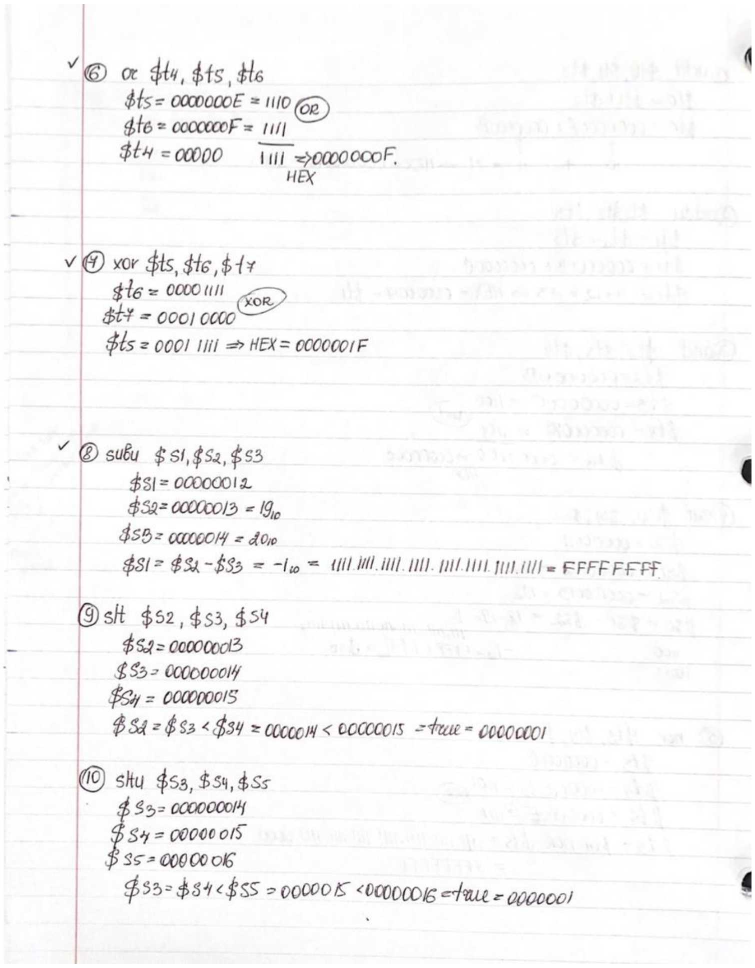
\$t3 = 0000000C

\$t4 = 0000000D = 1101 NOR

\$t5 = 0000000E = 1111

\$t3 = \$t4 nor \$t5 = 1111.1111.1111.1111.1111.1111.1111.0000

= FFFFFFF0



✓ ⑥ or \$t4, \$t5, \$t6

\$t5 = 0000000E = 1110 OR

\$t6 = 0000000F = 1111

\$t4 = 00000 $\overline{1111}$ ⇒ 0000000F.

HEX

✓ ⑦ xor \$t5, \$t6, \$t7

\$t6 = 0000 1111 XOR

\$t7 = 0001 0000

\$t5 = 0001 1111 ⇒ HEX = 0000001F

✓ ⑧ subu \$s1, \$s2, \$s3

\$s1 = 00000012

\$s2 = 00000013 = 19_{10}

\$s3 = 00000014 = 20_{10}

\$s1 = \$s2 − \$s3 = -1_{10} = 1111.1111.1111.1111.1111.1111.1111.1111 = FFFFFFFF.

⑨ slt \$s2, \$s3, \$s4

\$s2 = 000000013

\$s3 = 000000014

\$s4 = 000000015

\$s2 = \$s3 < \$s4 = 0000014 < 00000015 = true = 00000001

⑩ sltu \$s3, \$s4, \$s5

\$s3 = 000000014

\$s4 = 00000015

\$s5 = 00000016

\$s3 = \$s4 < \$s5 = 000015 < 00000016 = true = 0000001