UNIVERSITY OF CALIFORNIA, LOS ANGELES

UCLA

BERKELEY • DAVIS • IRVINE • LOS ANGELES • RIVERSIDE • SAN DIEGO • SAN FRANCISCO

SANTA BARBARA • SANTA CRUZ

CS M151B / EE M116C

Midterm Exam #1

All work and answers should be written directly on these pages, use the backs of pages if needed.

This is an open book, open notes quiz – but you cannot share books or notes.

We will follow the departmental guidelines on reporting incidents of academic dishonesty – do not make us enforce the rules. Keep your eyes on your own exam!

NAME: Srikanth Ashtalakshmi V Rajendran

ID: 804 478 782

Do not write anything in the area below on this page:

Problem 1: 12 (18) +2

Problem 2: 19 (20)

Problem 3: 29 (30)

Problem 4: 32 (32)

Total: 92 (out of 100)

= 94

1. ***Hardware Design ISA-bout Tradeoffs (18 points):*** Consider the three properties of execution time. For the following design decisions, indicate how these three properties could impact a MIPS processor running an application *A* with the following instruction breakdown and latencies:

Instruction	% of Instructions in *A*	Instruction Latency (cycles)
Load	25%	5
Store	5%	4
Simple R-type (i.e. adds, ands, shifts)	45%	4
Multiply	10%	5
BEQ/BNE	10%	3
Jump	5%	3

Note that this architecture is not the single cycle datapath and is not pipelined. Your answer for each should be that it either ***could increase, could decrease, or will stay the same.*** You may only circle one answer per property. The first one is done for you as an example.

Example: ***Design Decision***: Use a carry lookahead adder instead of a ripple carry adder.

- *Instruction Count* could increase could decrease will stay the same
- *Cycle Time* could increase could decrease will stay the same
- *CPI* could increase could decrease will stay the same

a. ***Design Decision***: Increase the size of the immediate field for I-type instructions by reducing the number of registers in the ISA.

2

- *Instruction Count* could increase could decrease will stay the same
- *Cycle Time* could increase could decrease will stay the same
- *CPI* could increase could decrease will stay the same

b. ***Design Decision***: A new physical design technology is used which can reduce wire latency.

4

- *Instruction Count* could increase could decrease will stay the same
- *Cycle Time* could increase could decrease will stay the same
- *CPI* could increase could decrease will stay the same

c. ***Design Decision***: A new compiler is used to compile application *A*. The compiler uses shifts in place of multiply instruction – it takes multiple shift instructions to replace a single multiply.

6

- *Instruction Count* could increase could decrease will stay the same
- *Cycle Time* could increase could decrease will stay the same
- *CPI* could increase could decrease will stay the same

2. ***Performance Anxiety (20 points)***: Consider the same application ***A*** and baseline MIPS processor from the previous problem. Application ***A*** executes three billion instructions. Answer the questions below – show your work.

a. What is the CPI for application ***A***?

CPI: 4.2

5

$CPI = 0.25(5) + 0.05(4) + 0.45(4) + 0.10(5) + 0.10(3) + 0.05(3)$

$CPI = 4.2$

b. The hardware running the application from part (a) has a cycle time of 300 ps (that is picoseconds). What is the execution time for that hardware to run the application?

5

ET: 3.78 s

$ET = IC \times CPI \times CT$

$ET = (3\times10^9)\times(4.2)\times(300\times10^{-12}) = 3.78s$

c. Now suppose we add an instruction that does a multiply/accumulate operation. The multiply/accumulate operation replaces one multiply instruction and one add instruction. This optimization allows the compiler to replace 50% of multiplies. What is the new CPI?

4

mul/accu CPI = 7

$\left(\frac{50}{100}\right)\times 10\% = 5\%$

CPI: 4.1

Out of 95% (100% − 5%)

% of multiplies = 10% − 5% = 5%

% of simple R-types (add) = 45% − 5% = 40%

% of mul/accu = 5%

New CPI = [0.25(5) + 0.05(4) + 0.40(4) + 0.05(5) + 0.10(3) + 0.05(3) + 0.05(7)] ÷ 0.95

New CPI = 4.1

d. What is the execution time for the same hardware from part (b) to run the new application from part (c)?

$IC = \frac{95}{100} \times (3\times10^9) = 2.85\times10^9$

ET: 3.51 s

5

$ET = (2.85\times10^9)\times(4.1)\times(300\times10^{-12})$

$ET = 3.5055\,s \simeq 3.51\,s$

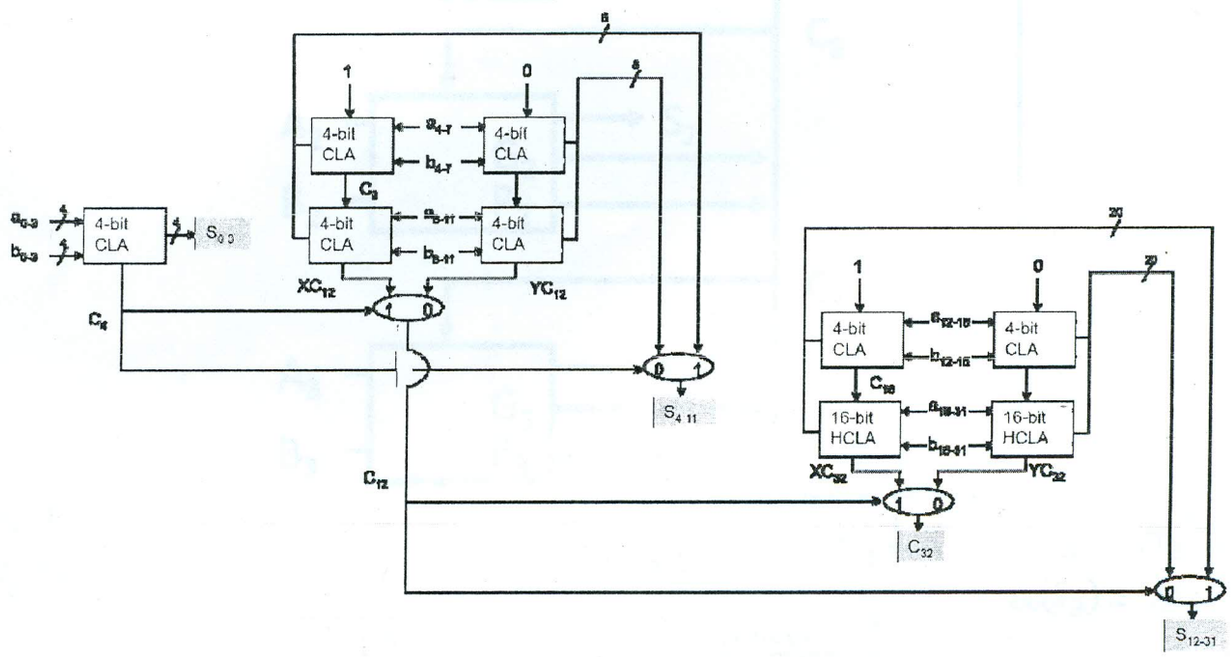
19

3. ***Putting the CLA into UCLA (30 points)***: Assume for the rest of this problem that all logic gates have the following delays:

Fan In	Delay
1	6T
2	7T
3	8T
4	9T
5	10T
6 or more	2T x fan-in

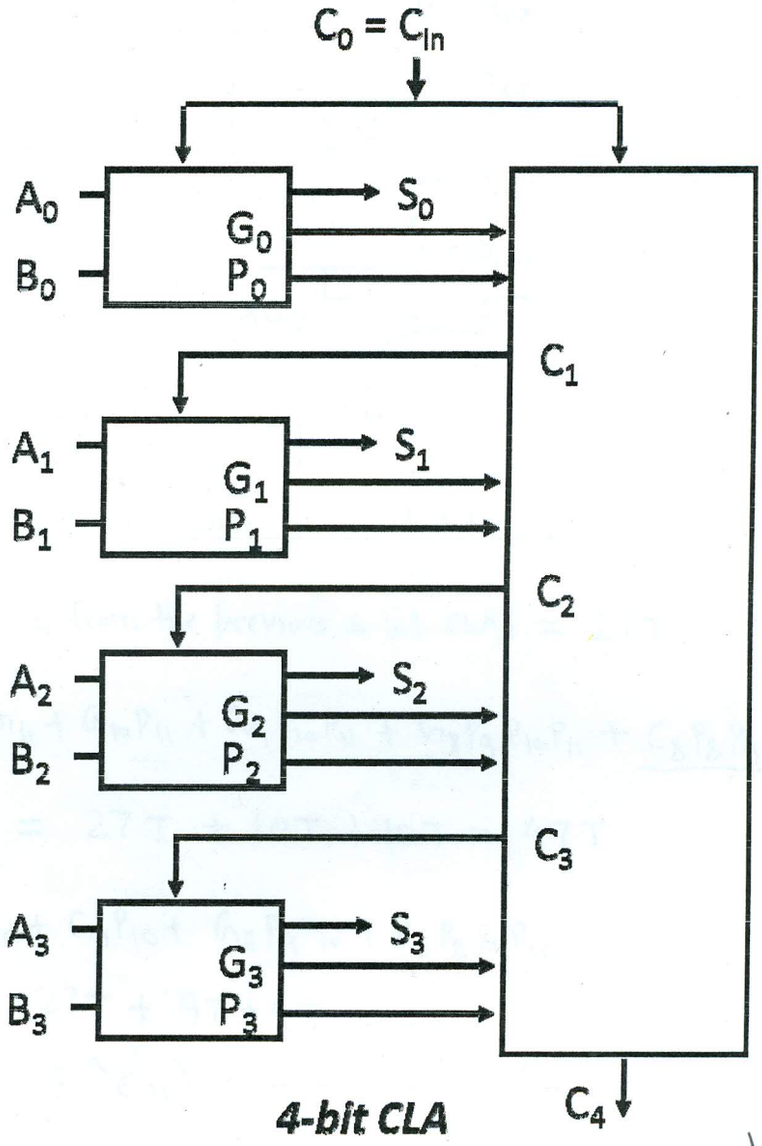
So a 2-input AND gate would have delay 7T and a 4-input OR gate would have delay 9T. For simplicity, assume that mux's have delay 7T regardless of fan-in.

We will create a 32-bit adder out of some building blocks we've covered in class. We will use the 4-bit carry lookahead (4-bit CLA) that we covered in class as one basic building block of this design. And we will use it (as we did in class) to make a 16-bit hierarchical CLA (16-bit HCLA). But instead of connecting these in series to make a 32-bit adder, we will use carry select to speed up the 32-bit adder. The design will look as follows (be sure to note where we are using the 4-bit CLA and where we are using the 16-bit HCLA):


4-bit CLA
4-bit CLA
4-bit CLA
4-bit CLA
4-bit CLA
4-bit CLA
4-bit CLA
4-bit CLA
16-bit HCLA
16-bit HCLA
1
0
1
0
1
0

Your task is to find the maximal delay of this design – i.e. determine the delays of $S_{0\text{-}31}$ and C_{32} – the maximal delay of these outputs will be the maximal delay of the entire design. To do this (and to help with possible partial credit) please use the diagrams on the following pages and fill in the tables in every page.

Single 4-bit CLA:



4-bit CLA

I'm using
$S_n = (A_n \wedge B_n \wedge C_n)$

$S_3 = (A_3 \wedge B_3 \wedge C_3)$
$d(S_3) = d(C_3) + 8T$
$= 25T + 8T$
$= 33T$

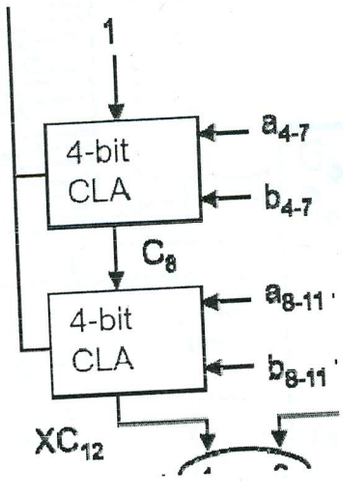
$G_i = A_i \cdot B_i$
$P_i = A_i \wedge B_i$

Output	Delay (in terms of T)	
G0	7T	*(1 points)*
P0	7T	*(1 points)*
G3	7T	*(1 points)*
P3	7T	*(1 points)*
C3	25T	*(2 points)*
C4	27T	*(2 points)*
S3	33T	*(1 points)*

$C_3 = G_2 + G_1P_2 + G_0P_1P_2 + \underline{C_0P_0P_1P_2}$
$d(C_3) = 7T + 9T + 9T = 25T$

$C_4 = G_3 + G_2P_3 + G_1P_2P_3 + G_0P_1P_2P_3 + \underline{C_0P_0P_1P_2P_3}$
$d(C_4) = 7T + 10T + 10T = 27T$

Ripple Carry 4-bit CLAs:



Output	Delay (in terms of T)	
C8	27T	(1 points)
XC12 (before any muxing)	47T	(2 points)
S11 (before any muxing)	53T	(2 points)

$d(C_8) = d(C_4 \text{ from the previous 4-bit CLA}) = 27T$

$XC_{12} = G_{11} + G_{10}P_{11} + G_9 P_{10} P_{11} + G_8 P_9 P_{10} P_{11} + \underline{C_8 P_8 P_9 P_{10} P_{11}}$

$d(XC_{12}) = 27T + 10T + 10T = 47T$

$C_{11} = G_{10} + G_9 P_{10} + G_8 P_9 P_{10} + \underline{C_8 P_8 P_9 P_{10}}$

$d(C_{11}) = 27T + 9T + 9T = 45T$

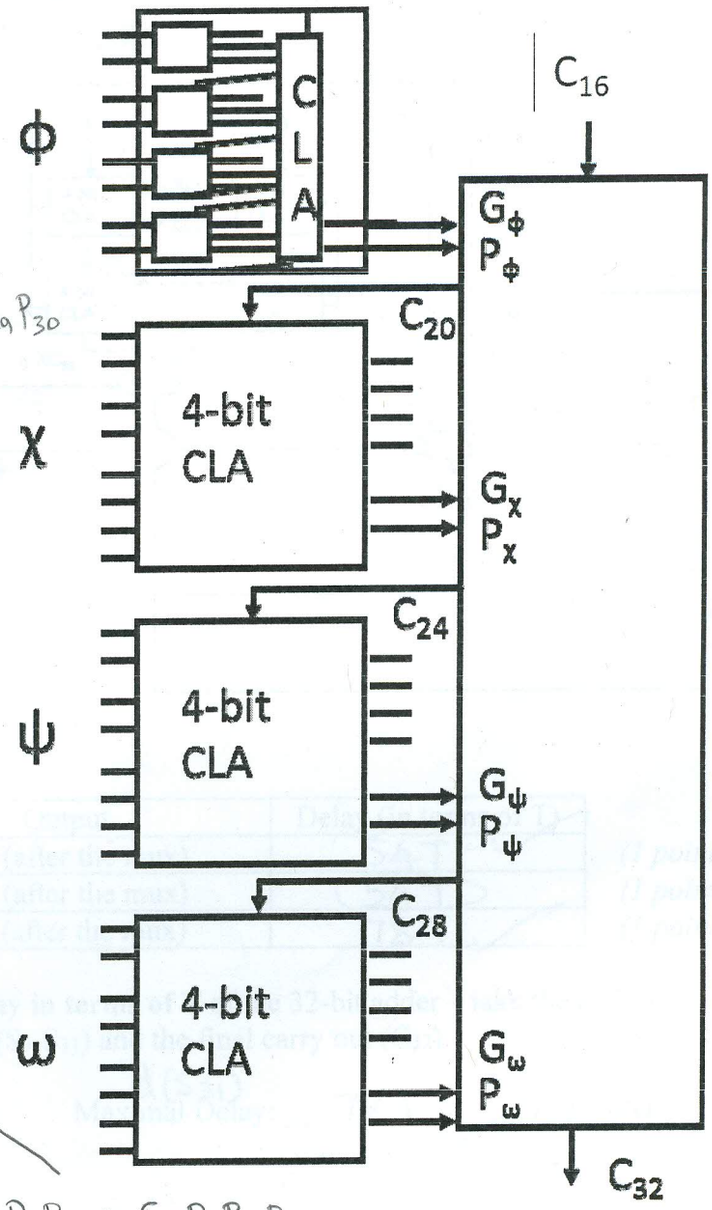
$S_{11} = (A_{11} \hat{} B_{11} \hat{} C_{11})$

$d(S_{11}) = 45T + 8T = 53T$

$C_{32} = G_\omega + G_\psi P_\omega + G_\chi P_\psi P_\omega + G_\phi P_\chi P_\psi P_\omega + \underline{C_{16} P_\phi P_\chi P_\psi P_\omega}$

$d(C_{32}) = 27T + 10T + 10T = 47T$

16-bit HCLA:



$C_{31} = G_{30} + G_{29}P_{30} + G_{28}P_{29}P_{30} + \underline{C_{28}P_{28}P_{29}P_{30}}$

$d(C_{31}) = 45T + 9T + 9T = 63T$

$S_{31} = A_{31} \wedge B_{31} \wedge C_{31}$

$d(S_{31}) = 63T + 8T = 71T$

$\overset{27T}{d(C_{16})} > \overset{25T}{d(G_\phi)}$

$C_{28} = G_\psi + G_\chi P_\psi + G_\phi P_\chi P_\psi + \underline{C_{16} P_\phi P_\chi P_\psi}$

$d(C_{28}) = 27T + 9T + 9T = 45T$

16-bit HCLA

$d(C_{16}) = d(C_8) = d(C_4) = 27T$

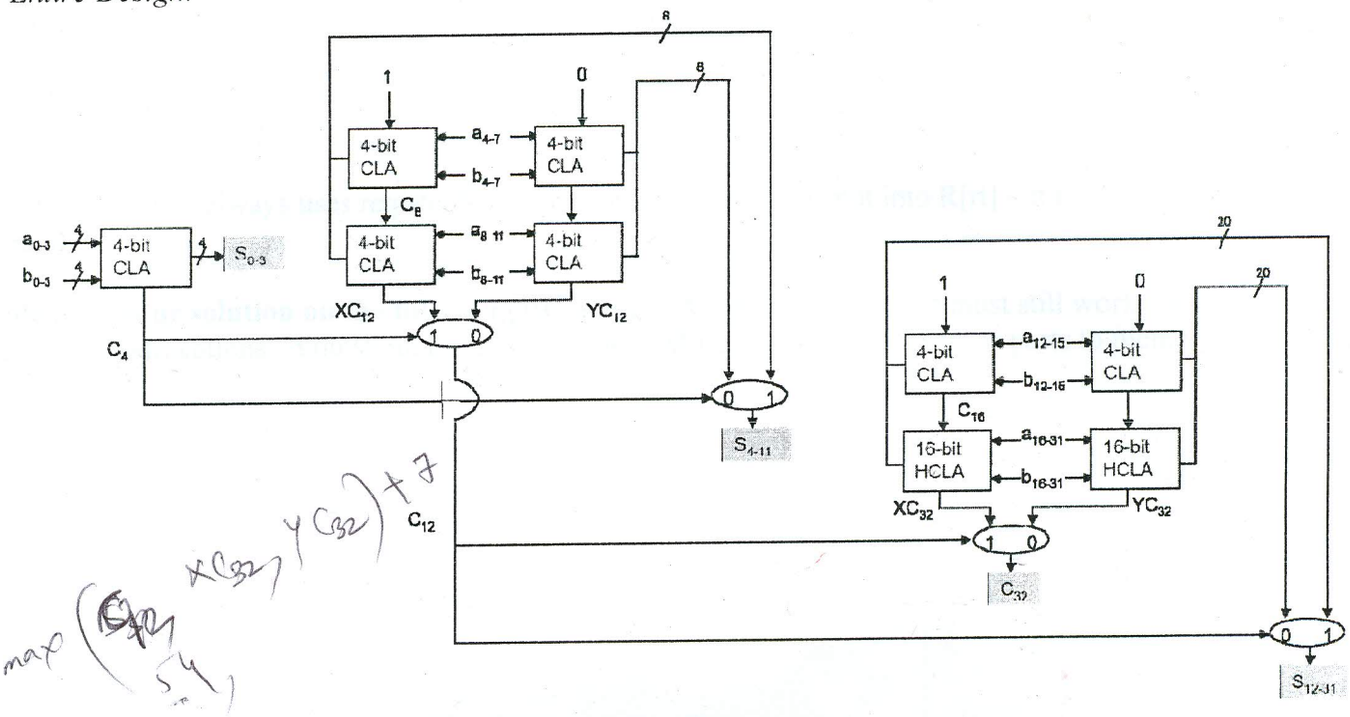
Output	Delay (in terms of T)	
Gω	25T	*(2 points)*
Pω	16T	*(2 points)*
C16	27T	*(2 points)*
C28	45T	*(2 points)*
C32 (before any muxing)	47T	*(2 points)*
S31 (before any muxing)	71T	*(2 points)*

$G_\omega = G_{31} + G_{30}P_{31} + G_{29}P_{30}P_{31} + \underline{G_{28}P_{29}P_{30}P_{31}}$

$d(G_\omega) = 7T + 9T + 9T = 25T$

$P_\omega = P_{31}P_{30}P_{29}P_{28}$ $\quad d(P_\omega) = 7T + 9T = 16T$

Entire Design:



Output	Delay (in terms of T)	
C12 (after the mux)	54T	*(1 points)*
C32 (after the mux)	54T	*(1 points)*
S31 (after the mux)	78T	*(1 points)*

Find the maximum delay **in terms of T** of the 32-bit adder – take the maximum of all output bits – including the sum bits (S_0-S_{31}) and the final carry out (C_{32}).

Maximal Delay: d(S31) 78T *(1 points)*

$d(C_{12})$ after mux $= d(XC_{12}) + 7T = 47T + 7T = 54T$

$d(C_{32})$ after mux $= 47T + 7T = 54T$

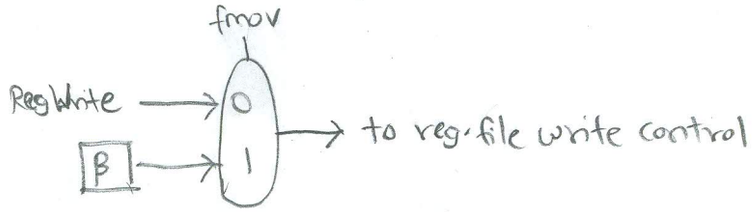
$d(S_{31})$ after mux $= 71T + 7T = 78T$

4. ***FMOV (32 points)***: Consider the single-cycle processor implementation. Your task will be to augment this datapath with a new instruction: the *fmov* instruction. This instruction will be an I-type instruction, and will have the following effect:

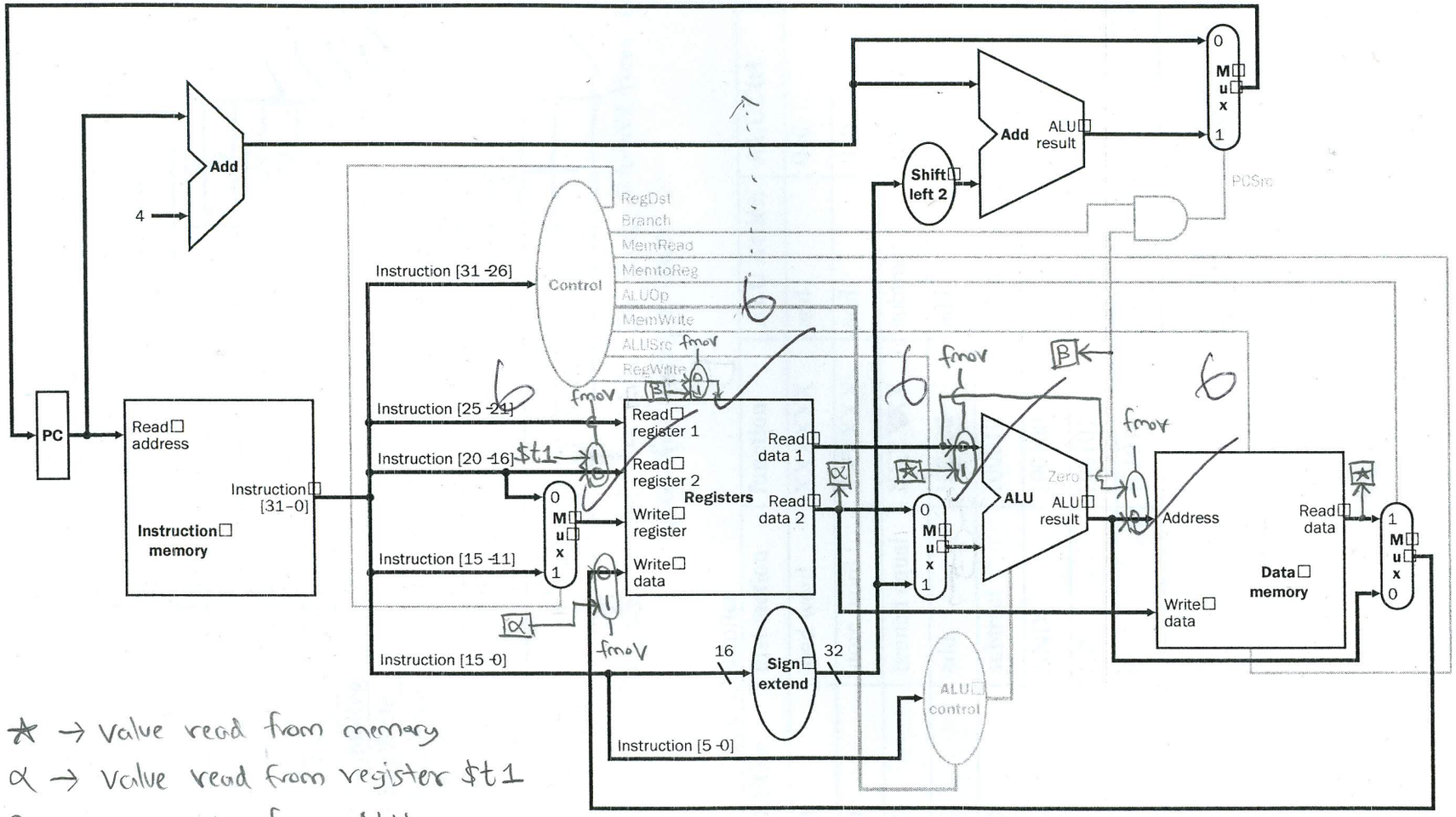
```
if (M[R[rs]]==SE(I))
        R[rt]=R[$t1]
```

Note that the *fmov* always uses register $t1 as the source value that is put into R[rt] – it is an implicit operand.

Implement your solution on the following two pages. All other instructions must still work correctly after your modifications. You should not add any new ALUs, register file ports, or ports to memory.



Write only occurs if β == 1



★ → value read from memory

α → value read from register $t1

β → zero value from ALU

Main Controller

Input or Output	Signal Name	R-format	lw	sw	Beq	fmov
Inputs	Op5	0	1	1	0	1
	Op4	0	0	0	0	1
	Op3	0	0	1	0	0
	Op2	0	0	0	1	0
	Op1	0	1	1	0	0
	Op0	0	1	1	0	0
Outputs	RegDst	1	0	X	X	0
	ALUSrc	0	1	1	0	1
	MemtoReg	0	1	X	X	X
	RegWrite	1	1	0	0	X
	MemRead	0	1	0	0	1
	MemWrite	0	0	1	0	0
	Branch	0	0	0	1	0
	ALUOp1	1	0	0	0	0
	ALUOp0	0	0	0	1	1
	fmov	0	0	0	0	1

ALU op is set to 01 so that subtraction can be used for comparison

ALU Controller

Opcode	**ALUOp**	**instruction**	**function**	**ALU Action**	**ALUCtrl**
Lw	00	load word	XXXXXX	add	010
Sw	00	store word	XXXXXX	add	010
Beq	01	branch equal	XXXXXX	subtract	110
R-type	10	add	100000	add	010
R-type	10	subtract	100010	subtract	110
R-type	10	AND	100100	AND	000
R-type	10	OR	100101	OR	001
R-type	10	SLT	101010	SLT	111