Vitaliy Prymak

Homework #3

If mux 0 does not 1 mean it off it just connection. For example

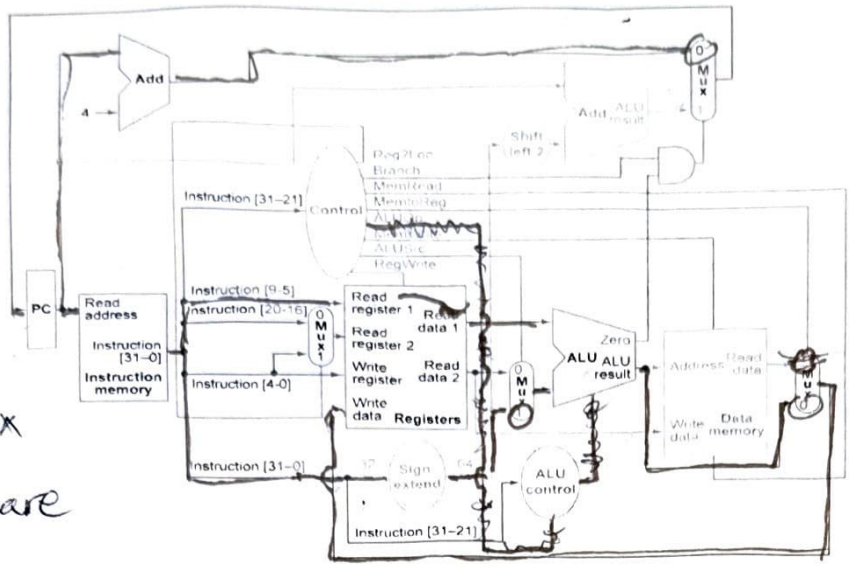
Read data → ALU: 1 / 0

Read data → Sign: 0 / 1

5 points each

1. We do not discuss the datapath for I-type instructions like ADDI or ANDI.
   (a) What additional logic blocks, if any, are needed to add I-type instructions to the CPU shown in the following figure (Figure 4.23 in the textbook)? Add any necessary logic blocks to it and explain their purpose.
   (b) Show the dataflow for this instruction using dash lines on the following figure

need PC as well in order to go to next instructions, if we not jumping then it 0.



if we do not use Mux then it X does not care

(c) List the values of the signals generated by the control unit for ADDI. Explain the reasoning for any "don't care" control signals.

if Reg2Loc = 0 it will turn on Reg 20-16

Reg2Loc	ALUSrc	MemtoReg	RegWrite	MemRead	MemWrite	Branch	ALUOp1	ALUOp0
X	1	0	1	0	0	0	1	0

taken from table for R-type

2. Problems in this exercise assume that the logic blocks used to implement a processor's datapath have the following latencies:

Shift left — PC — Register write

I-Mem / D-Mem	Register File	Mux	ALU	Adder	Single gate	Register Read	Register Setup	Sign extend	Control
250 ps	150 ps	25 ps	200 ps	150 ps	5 ps	30 ps	20 ps	50 ps	50 ps

10ps

"Register read" is the time needed after the rising clock edge for the new register value to appear on the output. This value applies to the PC only. "Shift left 2" takes 2 single gates time.

(a) What is the latency of an R-type instruction (i.e., how long must the clock period be to ensure that this instruction works correctly)? PC → IM → Mux → RF → MUX → ALU → MUX → RF= 20 (30, 250, 25, 150, 25, 200, 25)

(b) What is the latency of LDUR? (Check your answer carefully. Many students place extra muxes on the critical path.) PC → IM → RF → Sign → MUX → ALU → DMem → MUX → RFsetup (50, 250, 150, 50, 25, 200, 250, 25, 20)

(c) What is the latency of STUR? (Check your answer carefully. Many students place extra muxes on the critical path.) Register Read → I-Mem → RF → Sign → Mux → ALU → D-Mem

Longer 5X (d) What is the latency of B? Register Read → IM → Sign extend → 2 · Single Gates → ALU → MUX

(e) What is the minimum clock period for this CPU?

CBZ: Register Read | IM | RF | ALU | MUX

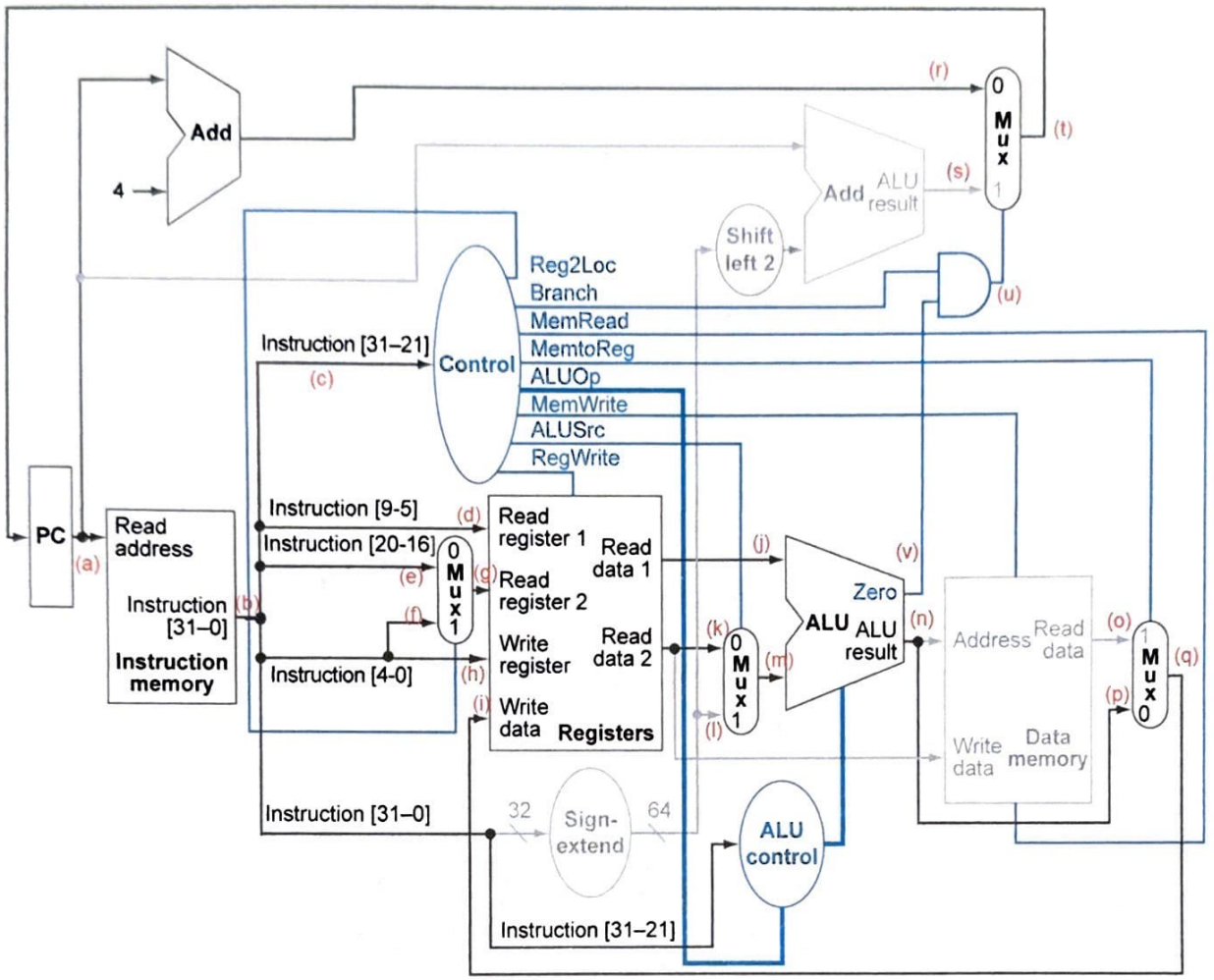
LDUR: Register Read → I-Mem → Sign Extend → 2 · Single gate → Adder → Mux → Reg

Sign and RF are at same time so we take only the longest

3. Consider the execution of this LEGV8 instruction: **SUB X3, X7, X9**. Below is its binary representation (the corresponding decimal is **1703182563**):

bits	31:21	20:16	15:10	9:5	4:0
	11001011000	01001	000000	00111	00011
	1624	**9**	**0**	**7**	**3**

Assume that this instruction is stored in address **20** (i.e. PC) in memory. **X7 = X9 = 5**. Write down the values (in decimal) of the inputs and outputs of the components labelled with (a) to (v) and their number of bits. If a value is unknow, put the symbol " ? ".



Value of (a):	# of bits:	Value of (b):	# of bits:
Value of (c):	# of bits:	Value of (d):	# of bits:
Value of (e):	# of bits:	Value of (f):	# of bits:
Value of (g):	# of bits:	Value of (h):	# of bits:
Value of (i):	# of bits:	Value of (j):	# of bits:
Value of (k):	# of bits:	Value of (l):	# of bits:

a: PC = 20; 64 bits — PC is special register (not in our instruction set)

b) 1703182563; 31 bits

c) 1624; 11 bits

d) 7; 5 bits

e) 9; 5 bits — based on diagram

f) 0; 2 bits — we can not choose because it's already destination register

g) 9; 5 bits — source register; it's not calculated. Calculation is ALU

h) 3; 5 bits — destination where it will be stored

i) 1703182563; 31 bits — [31-0] it's whole value but it won't be selected

j) 5, 64 bits — 64 bits because value representable as 64 bits. Value 000... 101

k) 5, 64 bits

l) 5, 64 bits

m) 0, 64 bits

n) 1, 1 bit — branch we do not use so it's 0 and 1 = 0

o) 0, 1 bit

s) 0, 1 bit

Sign Extend → Shift left 2 → ALU

1703182563 · 4 + 20 — 64 bits

r) 24, ~~5~~ 64 bits

h) 0, 64 bit

p) 0, 1 bit

q) 0, 64 bit

i) 0, 64 bits

o) X does not matter

blue lines is 1 bit only the other ones 64 mostly or 5 bits

4)

R-TYPE	I-TYPE	LDUR	STUR	CBZ	B
24%	28%	25%	10%	11%	2%

a) What fraction of all instructions use data memory:

LDUR + STUR = 25% + 10% = 35%

35%
100%

b) Instruction memory?

R-TYPE + I-TYPE + LDUR + STUR CBZ B

c) sign extend?

I-Type, CBZ, LDUR, LDUR

Value of (m):	# of bits:	Value of (n):	# of bits:
Value of (o):	# of bits:	Value of (p):	# of bits:
Value of (q):	# of bits:	Value of (r):	# of bits:
Value of (s):	# of bits:	Value of (t):	# of bits:
Value of (u):	# of bits:	Value of (v):	# of bits:

4. Consider the following instruction mix:

R-type	I-Type	LDUR	STUR	CBZ	B
24%	28%	25%	10%	11%	2%

(a) What fraction of all instructions use data memory?

(b) What fraction of all instructions use instruction memory?

(c) What fraction of all instructions use the sign extender?

5. When silicon chips are fabricated, defects in materials (e.g., silicon) and manufacturing errors can result in defective circuits. A very common defect is for one signal wire to get "broken" and always register a logical 1. This is often called a "stuck-at-1" fault. When you list instructions, assume instructions are only of the following types: loads, stores, arithmetic (R-type), arithmetic with immediate(I-type), and branches.

(a) Which instructions fail to operate correctly if the MemToReg wire is stuck at 1? ~~R-TYPE~~ not because ~~it~~ is not because it's only not going further on, LOAD, STUR

(b) Which instructions fail to operate correctly if the ALUSrc wire is stuck at 1? LOAD, STUR, I-TYPE

(c) Which instructions fail to operate correctly if the Reg2Loc wire is stuck at 1? R-TYPE, LDUR

6. In this exercise, we examine how pipelining affects the clock cycle time of the processor. Problems in this exercise assume that individual stages of the datapath have the following latencies:

IF	ID	EX	MEM	WB
250 ps	350 ps	150 ps	300 ps	200 ps

Also, assume that instructions executed by the processor are broken down as follows:

ALU	Branch/Jump	LDUR	STUR
45%	20%	20%	15%

(a) What is the clock cycle time in a pipelined and non-pipelined processor? Pipelined = 350, Non-pipelined = 1250

(b) What is the total latency of an LDUR instruction in a pipelined and non-pipelined processor? P = 350·5 = 1750; NP = 1250

(c) If we can split one stage of the pipelined datapath into two new stages, each with half the latency of the original stage, which stage would you split and what is the new clock cycle time of the processor? ID stage, reduced clock cycle time = 300

(d) Assuming there are no stalls or hazards, what is the utilization of the data memory? 20% + 15% = 35%

(e) Assuming there are no stalls or hazards, what is the utilization of the write-register port of the "Registers" unit? 45% + 20% = 65%

7. Consider the LEGv8 code below. Assume that X1 is initialized to 11 and X2 is initialized to 22.

```
ADDI X1, X2, #5
ADD X3, X1, X2
ADDI X4, X1, #15
ADD X5, X1, X1
```

(a) What would the final values of registers X3 and X4 be if the above instructions are executed in a pipeline processor that does not handle data hazards (i.e., does not stall the pipeline or use data forwarding for data hazards)? X3: 11+22=33 X4: 11+15=26

(b) What would the final values of register X5 be if the above instructions are executed in a pipeline processor that does not handle data hazards (i.e., does not stall the pipeline or use data forwarding for data hazards)? Assume the register file is written at the beginning of the cycle and read at the end of a cycle. Therefore, an ID stage will return the results of a WB state occurring during the same cycle. See Section 4.7 and Figure 4.51 for details. 22+5=27 27+27=54

(c) Suppose you executed the code on a version of the pipeline from Section 4.5 that handles data hazards by simply stalling the pipeline (i.e. inserting NOP instructions where necessary). Show the pipeline timing diagram below when the code is executed.

ADDI X1, X2, #5	IF	ID	EX	MEM	WB																
ADD X3, X1, X2		IF	ID	EX	ME	WB															
ADDI X4, X1, #15			IF	ID	EX	ME	WB														
ADD X5, X1, X1				IF	ID	EX	ME	WB													

(d) Suppose you executed the code below on a pipeline from Section 4.5 that uses data forwarding for handling data hazards. Show the pipeline timing diagram below:

ADDI X1, X2, #5	IF	ID	EX	MEM	WB																
ADD X3, X1, X2		IF	S	S	ID	EX	ME	WB													
ADDI X4, X1, #15					IF	ID	EX	ME	WB												
ADD X5, X3, X2						IF	ID	EX	ME	WB											

8. Consider the following loop.

```
LOOP: LDUR X10, [X1, #0]
      LDUR X11, [X1, #8]
      ADD X12, X10, X11
      STUR X12, [X1, #-8]
      SUBI X1, X1, #16
      CBNZ X12, LOOP
```

8.b. Cycle number:

```
Loop: LDUR X10, [X1, #0]
      LDUR X11, [X1, #8]
      SUBI X1, X1, #16
      ADD X12, X10, X11
      STUR X12, [X1, #8]
      CBNZ X12, LOOP
2nd iteration: LDUR X10, [X1, #0]
```

(a) Assume that data and control hazards are handled by simply stalling the pipeline (i.e. inserting NOP instructions where necessary). Show the pipeline timing diagram of the code execution.

Cycle number:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20

LOOP: LDUR X10, [X1, #0]	IF ID EX ME WB
LDUR X11, [X1, #8]	IF ID EX ME WB
ADD X12, X10, X11	IF S S ID EX ME WB
STUR X12, [X1, #-8]	IF S S ID EX ME WB
SUBI X1, X1, #16	IF ID EX ME WB
CBNZ X12, LOOP	IF ID EX ME WB
2nd iteration: LDUR X10, [X1, #0]	S S IF ID EX ME WB

(b) Can you reorder the code to reduce the number of stalls? If yes, show the reordered code.

(c) Show the pipeline timing diagram of the code execution with data forwarding and assume that the branch is handled by predicting it **as taken** and the branch target address is calculated at the ID stage.

Cycle number:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
LOOP: LDUR X10, [X1, #0]	IF	ID	EX	ME	WB															
LDUR X11, [X1, #8]		IF	ID	EX	ME	WB														
ADD X12, X10, X11			IF	ID	S	EX	ME	WB												
STUR X12, [X1, #-8]				IF	S	ID	EX	ME	WB											
SUBI X1, X1, #16					IF	ID	EX	ME	WB											
CBNZ X12, LOOP						IF	ID	EX	ME	WB										
2nd iteration: LDUR X10, [X1, #0]								S	IF	ID	EX	ME	WB							

(d) What is the speedup for the execution of (c) over (a)? 14/11 = 1,27 times faster

9. Compare the performance of a single-cycle datapath machine, a multi-cycle datapath machine and an ideal 5-stage pipeline machine. Assume that the single-cycle machine has a clock rate of 200MHz, and the multiple-cycle and pipelined machine have a clock rate of 1GHz. The CPI's of load, store, ALU, branch/jumps in the multi-cycle machine are 5, 4, 4, 3, respectively. The program has the following instruction mix:

- Load: 30 percent
- Store: 10 percent
- ALU/R-format instr.: 40 percent
- Conditional Branch: 10 percent
- Jump instr.: Remaining instructions

Calculate the performance (CPU time) of the 3 machines.

9. Multi-cycle:

$$CPI = 0,3 \cdot 5 + 0,1 \cdot 4 + 0,4 \cdot 4 + 0,1 \cdot 3 =$$

$$= 1,5 + 0,4 + 1,6 + 0,3 = 3,8$$

$$CPU\ time = IC \cdot CPI / CR =$$

$$= IC \cdot \frac{3,8}{1 \cdot 10^9} = 3,8 \cdot IC \cdot 10^{-9}$$

Single-Cycle

$$CPU\ time = IC \cdot CPI / CR =$$

$$= IC \cdot \frac{0,95}{200 \cdot 10^6} = ,00475 \cdot IC \cdot 10^{-6} =$$

$$= 4,75 \cdot IC \cdot 10^{-9}$$

Pipelined

$$CPU\ time = IC \cdot CPI / CR =$$

$$= \frac{0,95}{1 \cdot 10^9} = 0,95 \cdot IC \cdot 10^{-9}$$