No partial marks

1. State whether each of the following are [T]rue or [F]alse next to the statement in the space provided OR fill in the blanks appropriately.

   (a) (1 point) T or F: A compiler can influence both the number of instructions that are executed as well as the average CPI of the program. [T]

   (b) (1 point) A SLT instruction will need the 3 bit ALU Op Code to be set to 111

   (c) (1 point) T or F: For all I type instructions the ALU is used to compute a memory address by addition. [F]

   (d) (2 points) Suppose in the datapath shown on the last page (MIPS Single Cycle Datapath Reference), the Control unit fails such that the RegWrite signal is always stuck at 0. An example of an instruction that will still work correctly on this processor is SW, Jump etc.

   (e) (2 points) For the single-cycle implementation shown on the last page, suppose we were to probe a few of the control signals and found that for a particular clock cycle, ALUSrc=0, MemtoReg=0, RegWrite=0, and MemRead=0. Which of the following MIPS instructions might have been executing? Check ALL that apply. Incorrect answers will result in -0.5 points with an incorrect answer being that ANY part of the solution is incorrect. In other words, I will accept only perfect answers:-).

   a. lw ✗ ∵ RegWrite = 0
   b. sw ✗ ∵ ALUSrc = 0
   c. beq ✓
   d. add ✗
   e. addi ✗

   only BEQ is possible.

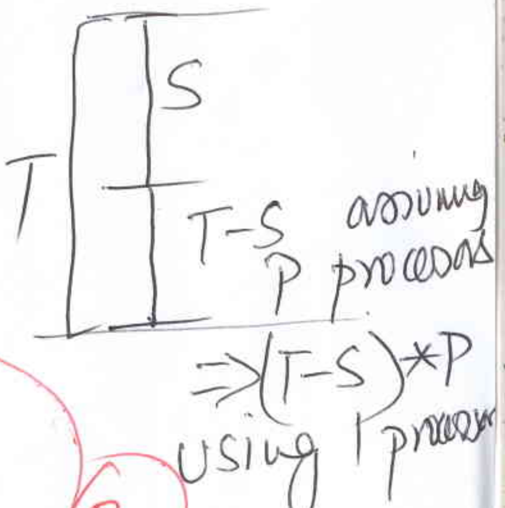
2. (3 points) A program is known to have a serial component that takes S seconds to execute, but the rest of the computation is arbitrarily parallelizable. If the program runs in T seconds using P processors, how long would it take to run using X processors? Assume $T > S$ and $P > 1$.

Application of Amdahl's Law.

Parallelizable portion = T-S.

1

$\Rightarrow$ Serial time of program with 1 processor $= S + (T-S)*P$

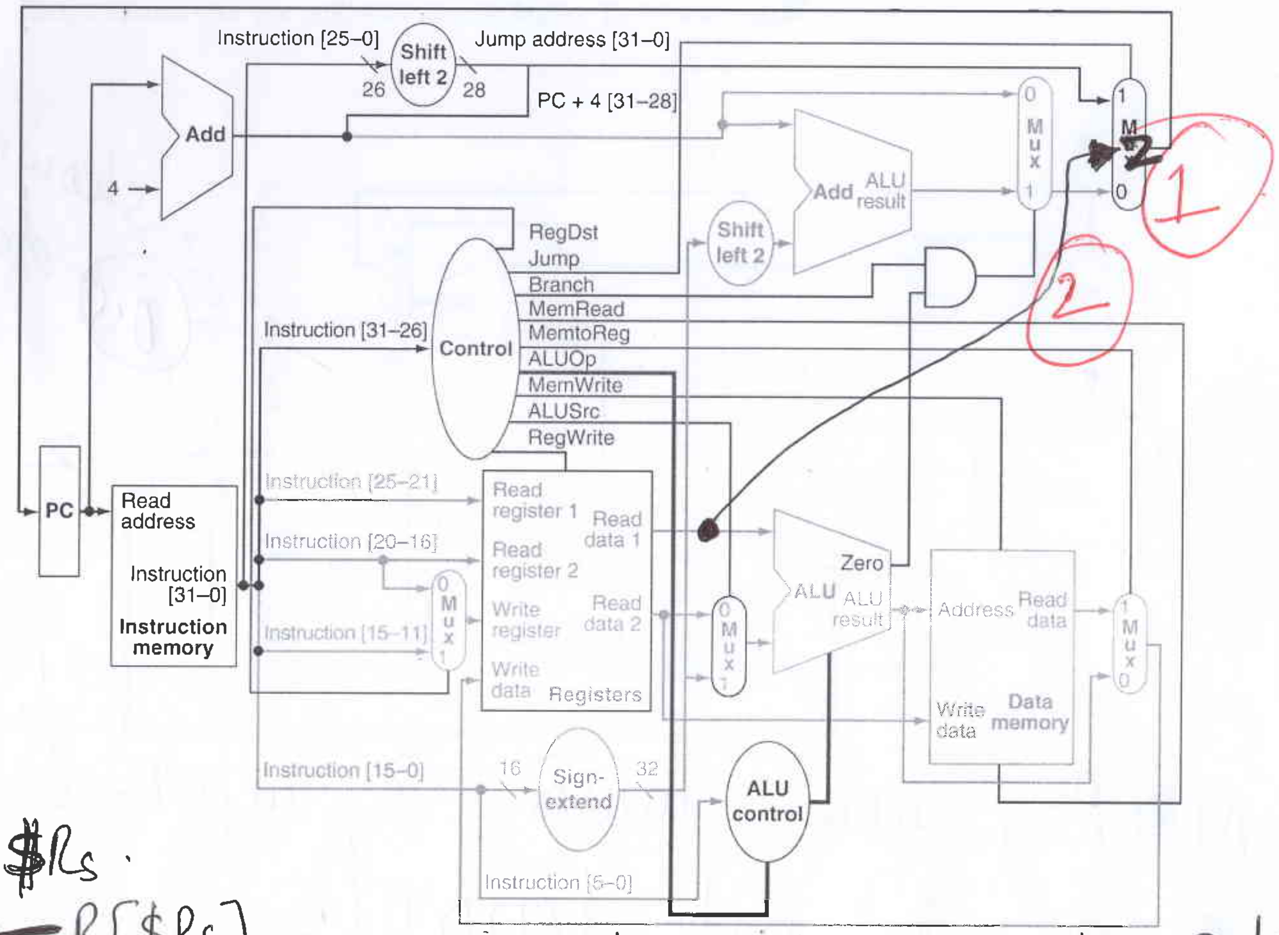


So for X processors

total time $= S + \frac{(T-S)*P}{X}$

2

3. (a) (5 points) Modify the datapath shown below to support the JR instruction in addition to the instructions it already supports. Show your modifications clearly ON the diagram. A spare diagram is included at the end of paper for you to practice on but only this page will be evaluated for points. Write down here any change to the existing control lines and what they impact.



JR $Rs.
PC ← R[$Rs].
The jump control will now be 2 bits
2 = 2 for JR.

2

(b) (3 points) For the datapath shown above (before your modifications) write down the control signal values for a *BEQ* instruction.

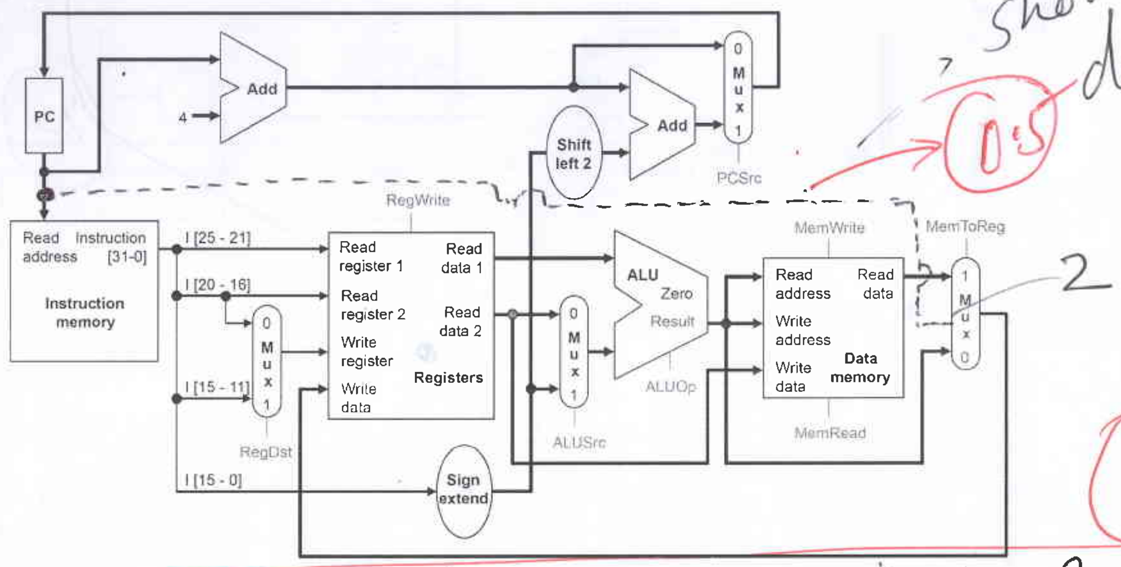
Jump	ALU Src	ALU Op	RegWrite	Branch	RegDst
0	0	SUB	0	1	X

0·5 x 6

4. (5 points) We want to implement a new I-type MIPS instruction `getpc $rt` which sets register \$rt to the PC value of this instruction. Make changes to the given datapath to implement the `getpc` instruction. Indicate the value of **all** control signals, including any new control signals.

shown by dashed lines

0.5



1

M to Reg now takes 3 inputs, & so need a 2 bit control.

Control Signals needed are:

RegDst = 0

PCSrc = 0

MemRead = MemWrite = 0

RegWrite = 1

ALU Src = X

ALU OP = X

Mem to Reg = 2

$0.5 \times \cancel{7}$

$= 3.5$

5. Consider the reference single cycle datapath shown at the end of the paper. In this implementation the clock cycle is determined by the longest possible path in the machine. The critical paths for the different instruction types that need to be considered are: R-type, Load-word, and store-word. All instructions have the same instruction fetch and decode steps. The basic register transfer of the instructions are:

```
Fetch/Decode: Instruction <- IMEM[PC];
R-type: R[rd] <- R[rs] op R[rt]; PC <- PC + 4;
load: R[rt] <- DMEM[ R[rs] + signext(offset)]; PC <- PC +4;
store: DMEM[ R[rs] + signext(offset)] <- R[Rt]; PC <- PC +4;
```

The timing for the various components/actions are:

ALU 10 ns
Adder 8 ns
ALU Control Unit 2 ns
Bit Shifter 3 ns
Control Unit 4 ns
Sign/zero extender 3 ns
2-1 MUX 2 ns
Memory (read/write) (instruction or data) 15 ns
PC Register (read action) 1 ns
PC Register (write action) 1 ns
Register file (read action) 7 ns
Register file (write action) 5 ns
Logic (1 or more levels of gates) 1 ns

(a) (5 points) In the table below, indicate the components that determine the critical path for the respective instruction, in the order that the critical path occurs. If a component is used, but not part of the critical path of the instruction (i.e., happens in parallel with another component), it should not be in the table. The register file is used for reading and for writing; it will appear twice for some instructions. *All instructions begin by reading the PC register with a latency of 2ns.*

2
1.5
1.5

Instruction Type			HW	Elements		used			
R Type	P.C	IM	Control	Read Reg	ALU Con	ALU	Control	RegWrite	
LW	"	"	"	"	"	"	DM	Control	RegWrite
SW	"	"	"	"	"	"	Read Reg	"	DM

(b) (3 points) Place the latencies of the components that you have decided for the critical path of each instruction in the table below. Compute the sum of each of the component latencies for each instruction.

Instruction Type										
R Type	2ns	15	4	7	2	10	4	5		
LW	2ns	15	4	7	2	10	15	4	5	0
SW	2ns	15	4	7	2	10	7	4	15	

(c) (4 points) Use the total latency column to derive the following critical path information:

- Given the data path latencies above, which instruction determines the overall machine critical path (latency)?

1.5

SW

- What will be the resultant clock cycle time of the machine based on the critical path instruction?

1.5

66 ns.

- What frequency will the machine run?

1

$1/66\text{ns} = 15.1$ MHz.

6. (5 points) Suppose we have a processor that has the following CPI figures for different kinds of instructions:

Kind	CPI
Memory	4
ALU	1
Cond. Branch	2
Jump	1

Our customers mainly execute two types of operating systems that have the following mix of instructions:

OS	% memory	% ALU	% Cond. Branch	% jump
Linoox	30	25	15	30
Windough	20	30	25	25

Given the available budget we can afford to make one of these two possible changes in the next generation of the processor:

- Reduce the number of cycles needed by memory instructions from 4 to 2, or
- ~~Reduce the number of cycles needed by~~ conditional branches from 2 to 1.

Which change should we make? Give a quantitative argument to support your decision.

2

4

Option 1

Lets look at the contribution of each type of instruction to the CPI before & after the change.

OS	CPI Contrib	New CPI Contrib	Improvement
Loonix	4 X 0.3 = 1.2	2 x 0.3 = 0.6	0.6
Windough	4 X 0.2 = 0.8	2 x 0.2 = 0.4	0.4

2

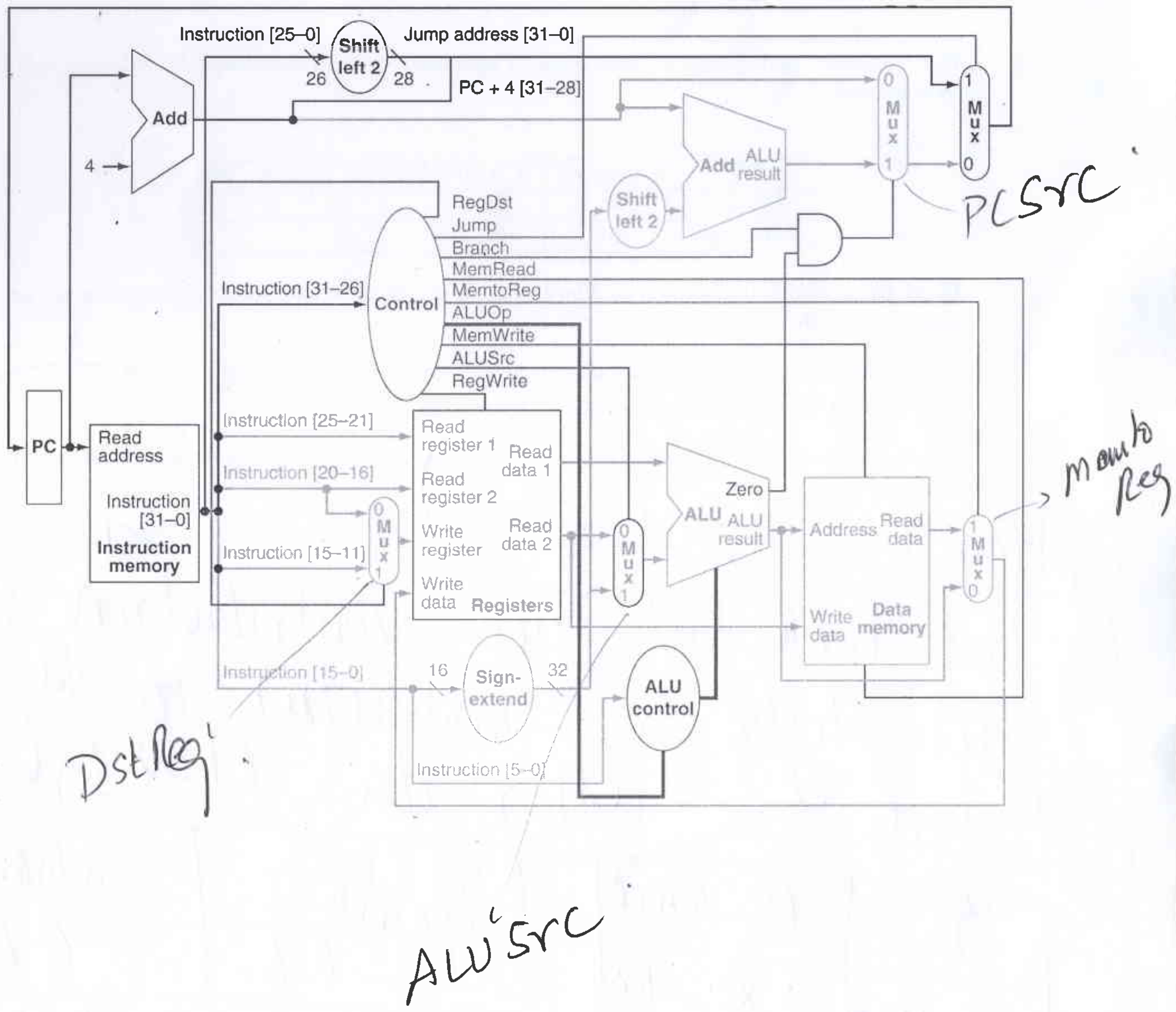
Option 2

OS	Old CPI contrib	New CPI contrib	Improvement
Loonix	2x0.15	1 x 0.15	0.15
Windough	2x0.25	1 x 0.25	0.25

Option 1 is better!

1

MIPS Single Cycle Datapath Reference


Instruction [25–0]
Shift left 2
Jump address [31–0]
26
28
PC + 4 [31–28]
Add
4
Add ALU result
Shift left 2
Mux
RegDst
Jump
Branch
MemRead
MemtoReg
ALUOp
MemWrite
ALUSrc
RegWrite
Instruction [31–26]
Control
PC
Read address
Instruction [31–0]
Instruction memory
Instruction [25–21]
Instruction [20–16]
Instruction [15–11]
Read register 1
Read register 2
Write register
Write data
Read data 1
Read data 2
Registers
Zero
ALU
ALU result
Address
Read data
Write data
Data memory
Instruction [15–0]
16
Sign-extend
32
ALU control
Instruction [5–0]
PCSrc
memto Reg
DstReg
ALUSrc