Problem 1 – MIPS Instruction Set Architecture (22 pts, 10 mins)

Extending the Immediate Field in MIPS (6 pts)
Mark the following statements true or false about executing the MIPS Core instructions from column 1 of the Green Card. If it's false, give a counterexample.

		Assertion	Counterexample
T	F	The immediate field of branches is sign extended	
T	F	The immediate field of and immediate (andi) and or immediate (ori) is zero extended.	
T	F	The immediate field of add immediate (addi) and set less than immediate (slti) is sign extended but the immediate field of add immediate unsigned (addiu) and set less than immediate unsigned (sltiu) is zero extended.	

Pipelines and MIPS (12 pts)
Assume a 5-stage pipeline as in Ch. 6 executing the MIPS Core instructions from column 1 of the Green Card. The following statements could affect the control lines if they were true. Mark them true or false. If it's false, give a counterexample.

		Assertion	Counterexample
T	F	All R-format instructions write one of the 32 general purpose registers in the WB stage	
T	F	All I-format instructions except branches write one of the 32 general purpose registers in the WB stage	
T	F	No J-format instructions write one of the 32 general purpose registers in the WB stage	
T	F	All R-format instructions use two of the 32 general purpose registers in the ID stage.	
T	F	No I-format instructions read two of the 32 general purpose registers in the ID stage.	
T	F	No J-format instructions read any of the 32 general purpose registers in the ID stage.	

Pipelines and MIPS Redux

Part A: (2 points)
Assume that Mem[100] == 200, $15 == 100

```
lw      $0, 0($15)
addu    $0, $0, $0
sw      $0, 100($15)
```

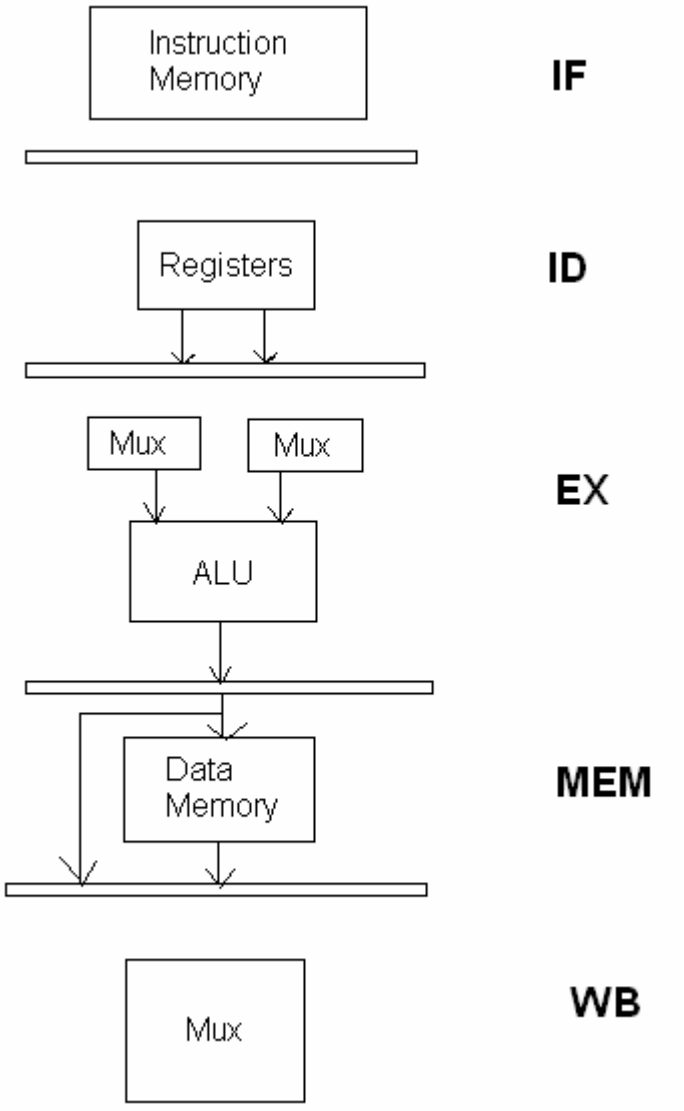
What is the value stored into location 200 as the result of the code above? _____

Part B: (2 points)
Assume a 5-stage pipeline with forwarding. How does pipeline control ensure proper execution of the code above?

Problem 2 – Pipeline Datapaths (14 points, 10 minutes)
The figure below shows part of the datapath of the MIPS pipelined processor shown in the book and in class. The partial datapath includes muxes for forwarding data from the WB stage to the EX stage. Note that this partial datapath does not include the ALU mux for immediate operands: that mux does NOT play a role in this problem. Also note that no wires are shown.

On this figure, draw in ALL the wires that connect the data inputs of the two EX-stage ALU muxes to the other boxes shown in the diagram. Also in this figure, draw ALL the inputs AND the output of the mux in the WB stage.

- Do not add unneeded inputs to the muxes.
- Do not add new boxes to the schematic.
- Do not add mux control lines.



Problem 3 – Performance (19 points, 10-15 minutes)
In the following two questions, we will be comparing the performance of two MIPS processors on an application with the following instruction mix.

Cycle Latency Lable

Operation	Frequency	Processor A	Processor B
Integer arithmetic / logical	30%	1	2
Load / Store	30%	2	4
Branch (predicted)	20%	1	1
Branch (mispredicted)	10%	19	10
Floating Point	10%	40	30

Part A: Calculate the CPI of each processor (6 points)

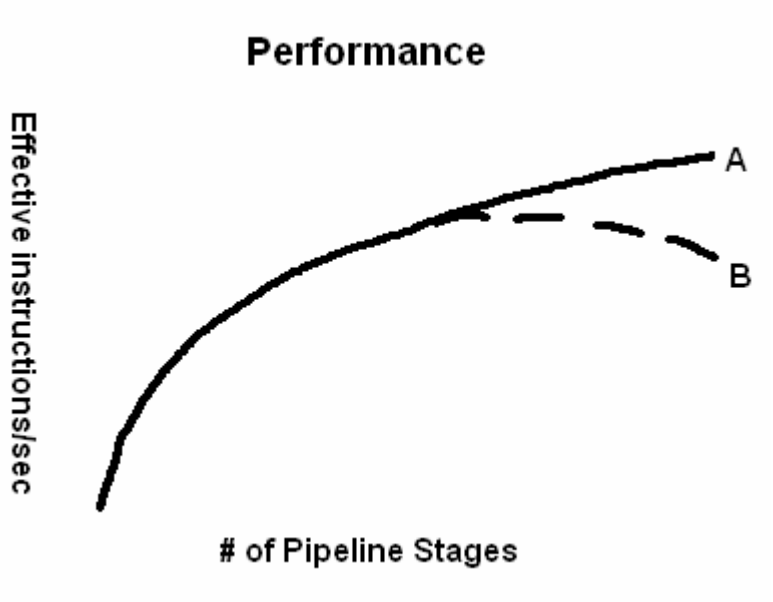
Price/Performance Table

Processor	Clock Speed	Price
Processor A	2.8 GHz	$133
Processor B	2.0 GHz	$100

Part B: Calculate the MIPS of each processor. (6 points)

Part C: Which of the two processors has the best cost/performance ratio? How could one get a different result without modifying the processors? (7 points)

Problem 4 – Pipeline Performance (15 points, 10 minutes)
Your final project is to implement a deep pipelined processor. However, there may be diminishing returns or detrimental effects that come with very long pipelines. Below is a graph of two possible performance curves.



Assume you have the following:

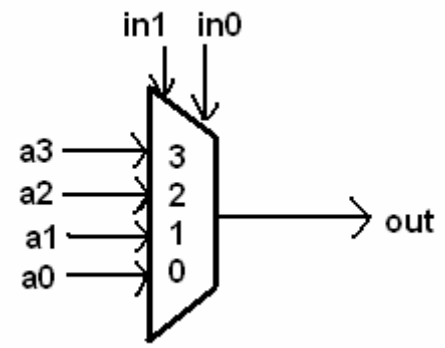
$t_{clk\text{-}to\text{-}q}$ t_{setup} t_{cl}

Part A: The main purpose of pipelining is to increase throughput by minimizing t_{cl} per clock cycle. Consider curve A in the graph above. Ignoring clock skew, give an expression for the theoretical maximum effective instructions/sec we will se with infinite pipeline stages. (10 points)

Part B: Now consider the more realistic curve B for processors with long pipelines. Ignoring clock skew and taking into consideration the hardware/software interface, for what reason we might see curve B instead of curve A? (5 points)

Problem 5 – Verilog and Logic Design (10 points, 5 minutes)

Consider the four-input multiplexer shown below.



Write a behavioral Verilog module for the mux.

- Use an always statement that contains a single case statement with 4 cases.
- Use blocking assignments exclusively.
- Use only the variables declared below.

```
module mux_case(in0, in1, a0, a1, a2, a3, out);

input     a0, a1, a2, a3, in0, in1;
output    out;
reg       out;
```

Problem 6 – FPGA vs. ASIC (18 points, 15 minutes)

This question gives a set of circumstances. Match the circumstances with the corresponding design aspect. Note that there are some circumstances which can affect multiple aspects of the design process. Select the BEST one. Each design process should only have one circumstance and all circumstances should be assigned.

For each design aspect, check either FPGA or ASIC depending on what the circumstance makes a better case for.

Part A – Circumstances (14 points)

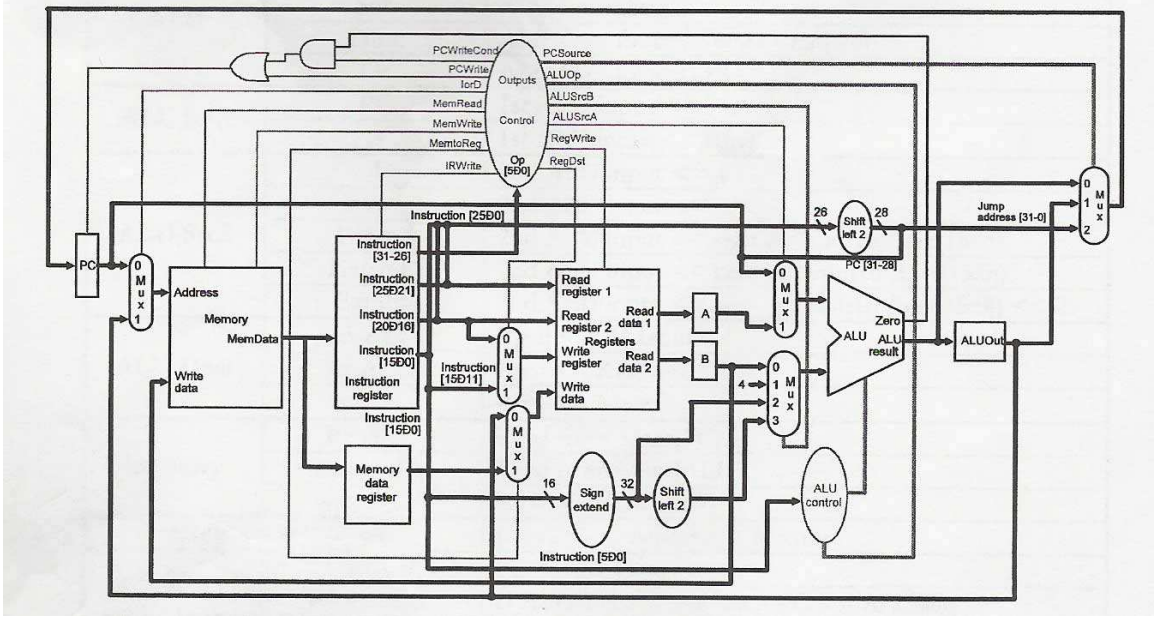
a. The device being designed is going to be used in a mixed signal application requiring analog and digital signal processing.
b. Custom mask sets cost upward of 1 million dollars.
c. Ford motor company needs devices for all air bag deployment systems in car models for years 2005-2007.
d. Design requires the expertise of 4 different groups of engineers in your company.
e. Analysts expect shift in the marketplace within 6 months.
f. Device developed for McDonald's new "Toy Story 3" happy meal toy.
g. Device customer wants parts now but is still waiting for final system specifications.

Design Aspect	**Circumstance**	**FPGA**	**ASIC**
Time to Market Pressures	_____	_____	_____
Heterogeneous System Components	_____	_____	_____
Personnel Costs ($/engineer/year/project)	_____	_____	_____
Manufacturing Costs	_____	_____	_____
Production Volume	_____	_____	_____
Specification Flexibility	_____	_____	_____
Final Sales Price	_____	_____	_____
_____	_____	_____	_____

Part B: Come up with an additional circumstance and design aspect and put them in the final line of the table. Describe the circumstance below. This aspect should NOT reasonably fall under any of the other aspects already mentioned. (4 points)

Problem 7 – Multi-cycle Design (22 points, 30 minutes)

Figure 1: Original Multicycle Datapath



Most modern computers use semaphores for software synchronization. These semaphores are basically variables that must be read, modified, and written back in one atomic operation. To assure that this command is atomic, these semaphores are usually implemented partially in hardware as atomic test-and-set or atomic swap. In this problem, you will be adding the `swap` instruction.

```
swap $r1, $r2, $r3
```

The swap instruction uses three registers, $r1 and $r2 as offsets to a base address is $r3. Here is the RTL description of `swap`.

```
Mem[$r1 + $r3] <-> Mem[$r2 + $r3]
```

Part A: Explain how the R-type instruction format can be used to support the `swap` instruction. In the figure below, specify the value of each field. (3 points)

R-type Instruction:

Op	**Rs**	**Rt**	**Rd**	**Shamt**	**Func**
_____	_____	_____	_____	_____	_____

Field Name	Values for Field	Function of Field
ALU	Add	ALU adds
	Sub	ALU subtracts
	Func	ALU does function code (Inst[5:0])
	Or	ALU does logical OR
ALU Src1	PC	1st ALU input <= PC
	rs	1st ALU input <= R[rs]
ALU Src2	4	2nd ALU input <= 4
	rt	2nd ALU input <= R[rt]
	Extend	2nd ALU input <= sign ext imm16 (Inst[15:0])
	Extend0	2nd ALU input <= zero ext imm16 (Inst[15:0])
	ExtShft	2nd ALU input <= sign ext imm16 (Inst[15:0]) << 2
ALU Dest	rd-ALU	R[rd] <= ALUout
	rt-ALU	R[rt] <= ALUout
	rt-Mem	R[rt] <= Mem
Memory	Read-PC	Read from Mem[PC]
	Read-ALU	Read from Mem[ALU]
	Write-ALU	Write to Mem[ALU]
MemReg	IR	Instruction register <= Memory
PC Write	ALU	PC <= ALUout
	ALUoutCond	If ALU Zero is true, then PC <= ALUout
Sequence	Seq	Go to the next sequential microinstruction
	Fetch	Go to the first microinstruction
	Dispatch	Dispatch using the ROM
Memory Src	RegB	Write data from RegB
	MDR Delay	Write data from the Swap Register
RegB Src	rt	Read register 2 <= rt (Inst[20:16])
	rd	Read register 2 <= rd (Inst[15:11])

Figure 2: Microcode assembly-language that supports the `swap` instruction.

Part B – Datapath Modifications (3 points)
Explain briefly why each of the following changes has been made to the datapath.

- Addition of the swap register

- The mux between the swap register and memory.

- The mux between the instruction and Read Register 2 input of the register file.

Part C – Microcode (16 points)
Below is partially completed microcode for the `swap` instruction. Fill in the missing code.

Label	**ALU**	**SRC1**	**SRC2**	**ALUDest**	**Memory**	**MemReg**	**PCWrite**	**Sequencer**	**MemSrc**	**RegRb**
Fetch	Add	PC	4		ReadPC	IR	ALU	Seq		
DPATCH	Add	PC	ExtShift					Dispatch		rt
Swap	Add	Rs	Rt					Seq		_____
Read	Add	Rs	Rt		Read-ALU			_____		_____
Read	Add	_____	_____		_____			_____		_____
Write	Add	_____	_____		_____			_____	_____	
Write					_____			_____	_____	