Name:

SSN(6):

CMSC 411 Computer Architecture
Midterm
Fall, 2001
November 1, 2001

For Two (2) points, guess the Theme of the exam:

Problem 1: Defining Properties (do 6 of 8 for 36 pts)

*****Answer this problem in the exam booklet*****

For full credit, your answer must include the item number, the term that you are defining, and a complete definition, in the exam booklet.

Too often we confuse "how something works" with "what this is". So, you must tell me what these things are or what the terms mean, for full credit. Partial credit will be given for examples and for partial answers. However, our intent is for you to capture *a complete* definition of (6) six of the following terms in the context of our our course. If you do more, we will only give credit for the first six (6) we see.

1.1 clock rate

1.2 big endian

1.3 register-memory architecture

1.4 register addressing mode

1.5 immediate operand

1.6 data hazard

1.7 branch penalty

1.8 exception handling

Problem 2: Duplicated Processors (3 of 4 for 24 pts)

*****Answer this problem in the exam booklet****

A new multi-processor system is to be constructed using a gross (144) of microprocessor-based machines and lots of ingenuity, some of it yours.

The 144 processors can be used in one of three modes at any given time: (1) all processors in use, (2) 72 processors in use, (3) serial mode (that's one at a time for you CS majors).

Please answer the following questions regarding this potential system. Be sure to include any assumptions necessary to justify your answer.

2.1 Suppose that the original compiler is replaced by one that eliminates some of the longer instructions, thus decreasing the CPI by 40%. However, it increases the instruction count by 60%. Write an expression for CR_{new}, the clock rate necessary to achieve the same performance as that prior to the modification.

2.2 A certain process must use serial mode 30% of the time. Give an expression for the effective speedup over serial mode if half the processors are used for 50% of the original execution time, and all processors are used for the remaining 20% of the original execution time.

2.3 Suppose that we wanted to permit an arbitrary number of processors to be used, from 1 (serial) up to n processors. Let p_i denote the fraction of time spent in a mode having i processors in use. Express Amdahl's law as a function of p_i, i, and n.

2.4 Is there some process for which a speedup of 144 feasible? Justify your answer for full credit.

*****Answer to problem 2 in the exam booklet*****

Problem 3: Doctoring Pipelines (up to 8 for 48)

*****Write the rest of your answers ON the exam*****

The **MMM-DLX**, Meesh's Maddeningly Magnificent-DLX architecture, has an unquenchable thirst for DLX, the ISA of choice for CMSC 411. The designer had an equally obsessive focus on exploiting ILP (instruction level parallelism). The next page and the back of the DLX code sheet contain all the information we have regarding this new (1,3) GPR architecture, and a sample code fragment is given below.

THE REMAINDER OF THE TEST DEALS WITH THE FOLLOWING MODIFICATION OF THE DLX ARCHITECTURE. SO DON'T SKIP THIS PART.

MMM-DLX Specifications

Warning: You MUST assume that the implementation includes separate memories; register writes and reads split across a clock cycle; and any forwarding, bypassing, or load interlocks required to resolve RAW hazards. Be sure to write down any *additional* assumptions you make.

The most significant modification in the **MMM-DLX** from the DLX architectures in H&P was moving the MEM stage from after the EX stage(s) to before the execution pipelines. Since ALU instructions can now have a memory access, we restrict the register addressing modes to register indirect. That is, loads and stores no longer require an effective address computation.

Instead, you have to have the exact address in the register. Thus, instruction (`LF F2, (R2)`) will put the value stored at the address in R2 into register F2. The instruction (`ADD R2,R3,(R1)`) will add the value stored at the address in R1 to the value in R3 and put the result in register R2. We call these new instructions ALUM (for ALU memory) to distinguish them from the standard ALU instructions. The symbol ALU/M refers to both ALU and ALUM instructions collectively.

A sample code fragment appears below:

```
LW R2, (R1)
LF F0, (R3)
LF F2, (R2)
MULTF F1, F0, F2
MULTF F1, (R4), F1
SF (R3),F1
ADDI R2, R2, #-16
LD F4, (R2)
ADDI R2, R2, #-16
ADDD F6,F4, (R2)
ADDI R2, R2, #-16
SD (R2),F6
SW (R4), R2
LW R3, (R10)
XORI R5, (R3), #-1
SW (R3),R5
```

*****The details of the MMM-DLX appear on the next page.*****

PS: Pipeline Stages

F	Instruction Fetch
ID	Decode instruction, fetch register values and test branch condition
MEM	Memory access for loads and stores and branch target computed
LWB/EX	Load write back (1 cycle) and INT and FP ALU Units execute (see FU chart below)
ALWB	ALU write back

FU: EX Stage Functional Units and Instruction Mix

The term **full pipe** (fully pipelined) means that each pipeline stage is one clock cycle.

Unit		#Stages or CC's	Ins Freq
INT ALU		2 stage	0.60
FP ADD		8 full pipe stages A1–A8	0.15
FP MULT		18 full pipe stages M1–M18	0.20
FP DIV		60 stages	0.05

CPU: CPU Execution-Time Specific Parameters

Ins Type	Ins Freq	# of CPU CC
ALU	0.40	β
ALUM	0.20	β
Load	0.20	1
Store	0.10	1
Cond Branch	0.09	3
Jump	0.01	1

Note: Problem 3.1 asks you to compute the value of β.

MEM: Memory Stall Parameters

Mem Access	Miss Rate	Miss Penalty
Ins Fetch	0.05	**XXX**
Data Fetch	0.10	**XXX**
Reads	**XXX**	100 CC
Writes	**XXX**	250 CC

TAKE A DEEP BREATH. NOW, DO ANY 8 (EIGHT). THERE ARE 12 SIX-POINT PROBLEMS THERE. FIND SOMETHING YOU CAN DO. AND LABEL YOUR WORK CLEARLY FOR PARTIAL CREDIT CONSIDERATION.

3.1 Write an expression for β, the average number of clock cycles for an ALU or ALUM instruction, assuming the ALU instruction mix shown in the FU Table.

3.2 Write an expression for the AVG CPI on the **MMM-DLX** assuming a perfect cache. That is, assume that the *miss rate* or **MR**, is zero. Your answer should be in terms of β, the average number of clock cycles for an ALU instruction, and assuming the instruction mix given in the CPU table.

3.3 Write an expression for the AVG CPI on the **MMM-DLX** taking the cache misses into account. Again, your answer should be in terms of β. Be sure to consider instruction and data fetches for each instruction type listed in the CPU Table.

3.4 Identify any potential structural hazards in the **MMM-DLX**. If none exist, explain why there are none.

3.5 Assess the potential for WAR, WAW, and RAR hazards in the **MMM-DLX**. That is, if they occur, give an example. If they don't, explain why they can't occur.

3.6 What type of strategy do you think is used in the **MMM-DLX** to deal with control hazards? Give and example of a control hazard, and explain your answer for full credit.

3.7 Write an expression for the speedup due to the **MMM-DLX** control hazard handling method over the method of stalling until the control hazard is cleared.

For the next set of problems, credit is 2 points per correct latency per row in each latency table. Partial credit will be given if work is organized and labelled clearly.

3.8, 3.9 Using the information in the FU table and the pipeline structure, determine the latencies in the **MMM-DLX** for the dependent pairs of ALU instructions in the table below. When the symbol

ALU/M is used, you may assume that the ALU and ALUM instructions have the same latencies associated with them.

Hint: for each dependent instruction pair, identify which stage of the producer instruction is the "source" of the value that must be "received" by the consumer instruction.

Producer Instruction	Consumer Instruction	Latency
FP ADD	FP ALU/M	
FP DIV	FP ALU/M	
FP MUL	FP ALU/M	
FP Load	FP ALU/M	
INT ALU/M	INT ALU	
INT ALU/M	INT ALUM	

3.10, 3.11 Determine the latencies associated with the pairs of operations given below. The notation is described in the previous problem's directions. Again, two points per correct row.

Producer Instruction	Consumer Instruction	Latency
INT ALU/M	FP Load	
INT ALU/M	FP Store	
INT ALU/M	INT Store	
FP ALU/M	FP Load	
FP ALU/M	FP Store	
INT ALU/M	Conditional Branch	

3.12 The is the last of the **MMM-DLX** latency problems. The previous four questions (3.8–3.11) never examined the potential latencies between INT Load producers or INT ALU/M producers and FP ALU and FP ALUM consumers. So, you get to. Make sure you explain your reasoning clearly.

Just some silly scratch paper.