Name: **ID:**

ECSE 425 -- Computer Organization and Architecture
Winter 2011

Midterm Examination

1:00 – 2:30 pm, March 4, 2011

Duration: 90 minutes

- **There are 5 questions for a total of 100 points. There are 12 pages. Please check that you have all 12 pages.**
- **This is a closed-book exam. You can bring 1 single-sided sheet of hand-written notes. This sheet of notes must be entirely hand-written, no portions may be machine-produced of photocopied.**
- **Calculators are permitted, but no cell phones or laptops are allowed.**
- **Write your name and student number in the space below. Do the same on the top of each sheet of this exam.**
- **State any assumption.**
- **Write your answers in the space provided.**

Name: _____

Student Number: _____

Q1: _____ **Q2:** _____

Q3: _____ **Q4:** _____

Total: _____

Name: **ID:**

Question 1. Short Answers (20 points)
There are 10 sub question (2 points each)

For each question below, provide a short answer in 1-2 sentences.

a) List two of the 3 principles in computer design.

b) What is a balanced pipeline and why is it desirable?

c) Name two factors that limit the speed-up of a pipeline.

d) What hazards can be caused by name dependency?

e) What is precise exception?

f) Why does a pipeline complicate the implementation of precise exception?

g) How does dynamic scheduling and speculation differ with regards to changing the processor state?

h) How is forwarding implemented in Tomasulo organization?

i) How does in-order commit in speculative machine affect exception handling?

j) What is the main difference between superscalar and VLIW processors?

Name: **ID:**

Question 2. Branch Prediction (20 points)

Use the following snippet of code for this question:

```
a = 0;
b = 0;
for (i = 0 ; i < 4 ; i++) // B1
   if ((x[i] % 2) == 0) && (x[i] > 0) // B2
      a = a + 1;
   if ((x[i] % 4) == 0) // B3
      b = b + 1;
```

Given x = [2 -6 3 8], fill in the table on the next page for a (1,1) correlating branch predictor.

The branch predictors are all initialized to the *taken state*, and the last branch prior to this snippet of code was *taken*.

Name: **ID:**

```
x = [2 -6 3 8]

a = 0;
b = 0;
for (i = 0 ; i < 4 ; i++) // B1
   if ((x[i] % 2) == 0) && (x[i] > 0) // B2
      a = a + 1;
   if ((x[i] % 4) == 0) // B3
      b = b + 1;
```

#	Branch	State when prev. branch NT	State when prev. branch T	Prediction	Outcome	Update NT	Update T
1	B1						
2	B2						
3	B3						
4	B1						
5	B2						
6	B3						
7	B1						
8	B2						
9	B3						
10	B1						
11	B2						
12	B3						
13	B1						

Name: **ID:**

Question 3. Pipelining (20 points)

For this question, use the following snippet of code, which implements x[i] = a x[i] + b.

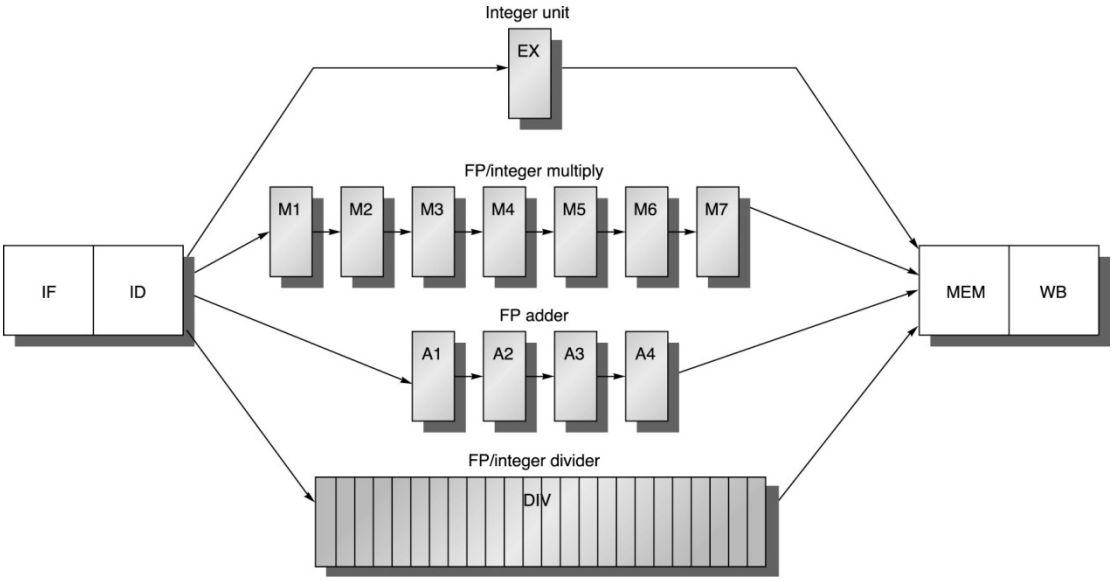
```
loop:  L.D     F4, 0(R1)
       MULT.D  F4, F0, F4
       ADD.D   F4, F2, F4
       S.D     F4, 0(R1)
       SUBI    R1, R1, #8
       BNEZ    R1, loop
```

The code is run on a floating point pipeline as shown below with the following cycles to execute and initiation intervals:

Functional unit	Cycles to execute	Initiation interval
Integer ALU	1	1
FP add	4	1
FP multiply	7	1
FP divide	25	25 (i.e. not pipelined)

Assume that the pipeline has full hazard detection and forwarding hardware. Assume a register read and a write in the same clock cycle "forward" through the register file. Assume all memory accesses take 1 clock cycle, and that the branch is handled by flushing the pipeline.

Fill in the chart on the next page to show the timing of the first iteration and the first instruction of the second iteration. What is the average number of clock cycles per iteration?



Name: **ID:**

																				Instr.
																				1
																				2
																				3
																				4
																				5
																				6
																				7
																				8
																				9
																				10
																				11
																				12
																				13
																				14
																				15
																				16
																				17
																				18
																				19
																				20
																				21
																				22
																				23
																				24
																				25
																				26
																				27

Name: **ID:**

Question 4. Dynamic Scheduling (20 points)

Now consider a dynamically scheduled machine using the Tomasulo algorithm (*without* hardware speculation). Assume that the functional units are *fully pipelined* and that all memory accesses hit the cache (so that memory accesses take 1 cycle). There is a memory unit with 5 load and 5 store buffers. Each load or store takes 2 cycles to execute, 1 to calculate the effective address, and 1 to load or store the data. Assume there are dedicated integer functional units for effective address calculation and branch condition evaluation. The other functional units are described in the following table.

Func. unit type	Cycles to execute	Number of func. units	Number of reservation stations
Integer ALU	1	1	5
FP adder	4	1	3
FP multiplier	7	1	2
Load	2	1	5
Store	2	1	5

For the same snippet of code repeated below, fill in the table on the next page with the clock cycle number that each instruction issues, begins and ends execution and writes its result for the first 2 iterations of the loop. Assume that there is at least one cycle delay between successive steps of every instruction execution sequence: issue, start execution and write back.

```
loop:  L.D     F4, 0(R1)
       MULT.D  F4, F0, F4
       ADD.D   F4, F2, F4
       S.D     F4, 0(R1)
       SUBI    R1, R1, #8
       BNEZ    R1, loop
```

Name: **ID:**

Func. unit type	Cycles to execute	Number of func. units	Number of reservation stations
Integer ALU	1	1	5
FP adder	4	1	3
FP multiplier	7	1	2
Load	2	1	5
Store	2	1	5

```
loop:  L.D     F4, 0(R1)
       MULT.D  F4, F0, F4
       ADD.D   F4, F2, F4
       S.D     F4, 0(R1)
       SUBI    R1, R1, #8
       BNEZ    R1, loop
loop:  L.D     F4, 0(R1)
```

Loop iteration	Code		Issue	Exec. Start	Exec. End	Write Back
1	L.D	F4, 0(R1)	1			
1	MULT.D	F4, F0, F4				
1	ADD.D	F4, F2, F4				
1	S.D	F4, 0(R1)				
1	SUBI	R1, R1, #8				
1	BNEZ	R1, loop				
2	L.D	F4, 0(R1)				
2	MULT.D	F4, F0, F4				
2	ADD.D	F4, F2, F4				
2	S.D	F4, 0(R1)				
2	SUBI	R1, R1, #8				
2	BNEZ	R1, loop				

Name: **ID:**

Question 5. Performance Analysis (20 points)

Part a) Reliability and Amdahl's law. (10 points)
Consider a system in which the components have the following MTTF (in hours):

CPU	1,000,000
Hard disk	200,000
Memory	500,000
Power supply	400,000

Part a.i)
Assume that if any component fails, then the system fails. What is the system MTTF?

Part a.ii)
You buy an additional hard drive and bring the total hard disk MTTF to 600,000 hours, which provides 3 times improvement. **Using Amdahl's law**, compute the improvement in the whole system reliability.

Part b) (10 points)

Suppose that we have a machine with 7-stage pipeline:

IF, ID1, ID2, EX, M1, M2, WB

Branch target addresses are calculated in ID2 and branch conditions are evaluated in EX.

Assume a base CPI of 1 without any stall. Assume 30% conditional branch frequency and 80% of these are taken. How much faster is the machine with predicted-taken branch prediction than with predicted-not-taken scheme?

Name: **ID:**

Extra blank page