EECS 470 *Midterm Exam - Solutions*

Fall 2016

Name: _____ unique name: _____

Sign the honor code:

I have neither given nor received aid on this exam nor observed anyone else doing so.

_____

Scores:

#	Points
1	**/ 18**
2	**/ 16**
3	**/ 15**
4	**/ 16**
5	**/ 15**
6	**/ 20**
Total	***/ 100***

NOTES:

- Closed book. Closed notes.
- Calculators are allowed, but no PDAs, Portables, Cell phones, etc.
- Don't spend too much time on any one problem.
- You have 90 minutes for the exam (avg. 15 minutes per problem).
- There are 10 pages in the exam (including this one), including a 1-page answer sheet for problem number 6. Please ensure you have all pages.
- **Be sure to show work and explain what you've done when asked to do so.**

1) Short answer [points]

a) Does pipelining improve latency or throughput? **[2 points]**

throughput

b) Give an example of a microprocessor component that exploits locality. **[2 points]**

cache

c) Explain why it doesn't make sense for a 2-wide processor with a 2-cycle L1 cache access time and a 12 cycle L2 cache access time and 32 architectural registers to have a 16 entry re-order buffer (ROB). Should the ROB be larger or smaller? **[3 points]**

Needs to be larger to hide L2 hit latency (2 wide * 12 cycles = 24 entries minimum)

d) Name and briefly explain a technique to reduce the **dynamic power dissipation** of a processor. **[3 points]**

Dynamic frequency & voltage scaling

e) Name two kinds of data hazards that the Tomasulo algorithm can eliminate for which a scoreboard machine must stall. **[4 points]**

WAR (anti) dependence and WAW (write) hazards.

f) In class, we studied the reorder buffer as a mechanism to implement precise exceptions. Name (or, if you can't remember the names, describe) **two** other mechanisms to implement precise exceptions, as described by D. Sima. **[4 points]**

History buffer and future file.

2) Power & Performance [16 total points]

a) Qualcomm is working to improve the energy-efficiency of its next-generation mobile processor. The baseline processor consumes 9W. They are considering adding a new low-power mode that shuts off 75% of the on chip caches. The low power mode incurs a 10% performance hit, but reduces power consumption to 7W. Should they add the new low power mode? Why or why not? **[8 points]**

If clock rate and voltage are reduced 10% (giving 10% performance degradation), power is reduced to (0.9)^3 * 9W = 6.56W. Hence, voltage and frequency scaling is better than the proposed low power mode.

b) ARM has recently been pushing the use of its new 64-bit cores in server-class systems for applications like online transaction processing. The key idea of this push is to improve energy efficiency (transactions per Joule). ARM is now evaluating systems that replace a single Intel server CPU with a cluster of several ARM cores.

Suppose an Intel server CPU, operating at a nominal voltage of 1.1V consumes 60W and can provide 16800 transactions per minute (tpm). ARM cores operate at 0.9V, consume 2.5W, and provide 800 tpm each. The Intel CPU can scale its voltage down to 0.9V, while the ARM cores' operating voltage is fixed. Assume that transaction processing performance is proportional to clock frequency. Further assume that the transaction processing workload can be parallelized across an arbitrary number of ARM cores (performance scales linearly with the number of cores).

Which approach is more energy efficient? Justify your answer by comparing the best transactions per Joule achievable under each design. **[8 points]**

Intel @ 1.1V: power 60W; 16800 tpm = 280 tps; 4.66 trans / joule
Intel @ 0.9V: power 32.86W; 13745 tpm = 342.48 tps; 10.4 trans / joule
ARM @ 0.9V: power 2.5W; 800 tpm = 13.3 tps; 5.3 trans / joule

Intel CPU scaled to 0.9V is the most energy-efficient design

3) Iron Law [15 total points]

The "Iron Law" of processor performance breaks execution time of a program into three factors. These factors help us to understand the impact of various design changes.

a) What are the three factors of the Iron Law? **[3 points]**

clock frequency, CPI, instructions per program

For each of the following system modifications, indicate their effect on each factor of the Iron Law (increase, decrease, no change). For each increase or decrease, explain why the change occurs (no more than one sentence per change).

b) Change data structure memory layouts to improve spatial locality. **[4 points]**

reduces CPI

c) Increase the processor issue, dispatch, and retirement widths from 1 to 4. **[4 points]**

reduces CPI, increases clock frequency

d) Add a "multiply-accumulate" instruction to the ISA to replace sequences of multiply and add instructions. **[4 points]**

reduces instructions per program, may or may not affect CPI and clock frequency.

4) Amdahl's Law [16 points]

Consider a program where 20% of its execution is serial and the remainder is "embarrassingly parallel" (i.e., its performance scales linearly in the number of cores for an arbitrary number of cores). The performance of the serial portion of the program is directly proportional to memory access latencies.

a) What is the maximum possible speedup that can be achieved for the program? **[4 points]**

5x.

b) For a system with 8 cores, what is the maximum speedup that can be achieved? **[4 points]**

3.33x

Recent advances in memory technology can reduce memory access latency in half. With the new memory technology:

c) What is the maximum possible speedup that can be achieved for the program? **[4 points]**

9x.

d) For a system with 8 cores, what is the maximum speedup that can be achieved? **[4 points]**

4.5x.

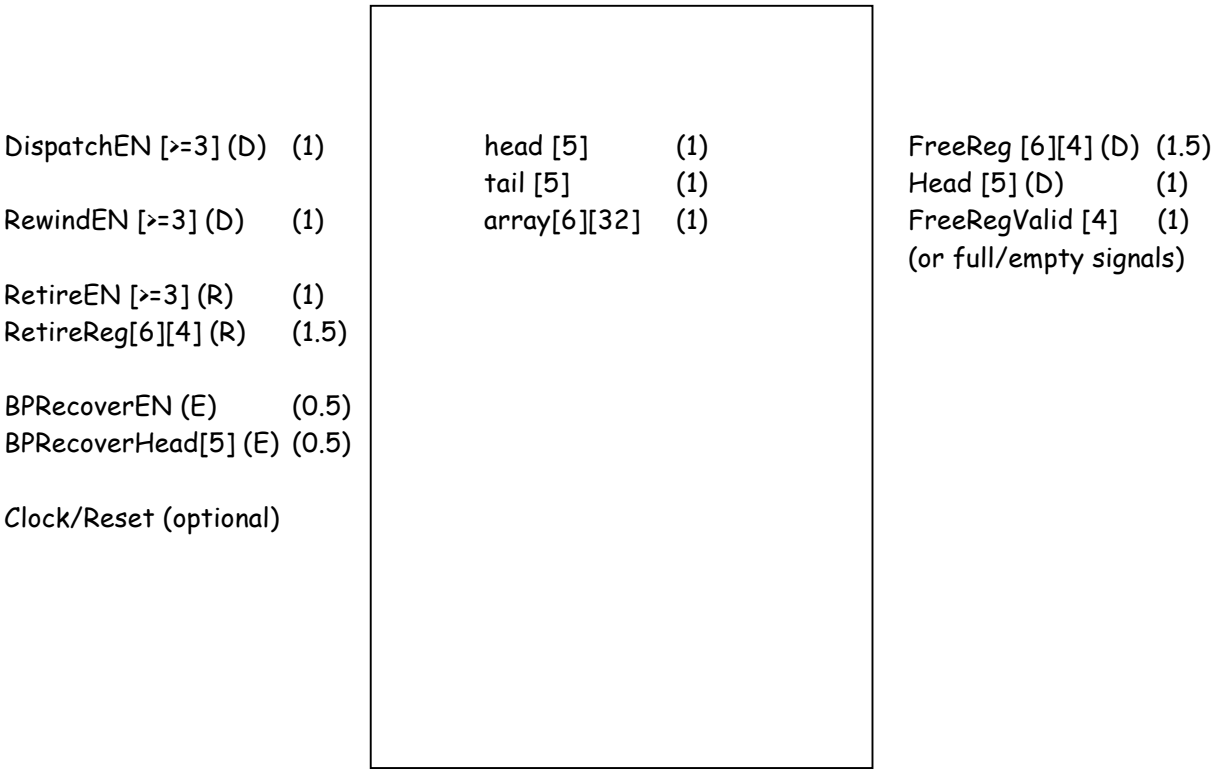
5) R10K Free List [15 points]

The Free List is often one of the trickier components of an R10K microarchitecture to implement correctly. Complete the block diagram below for a free list for a 4-wide superscalar machine with a 32-entry ROB, 32 architectural registers, and a 64-entry physical register file. The free list must support 4-wide dispatch and exception rewind of 4 instructions per cycle (note: each cycle, the free list will perform either dispatch or rewind, but not both. Note further that the free list can do exception rewind and retire at the same time). It should also support single-cycle recovery of a mispredicted branch (note: you only need to support fast recovery for one outstanding branch). **[12 points]**

Be sure your diagram shows:

- all state stored in the free list, including the free list entries and any additional book-keeping storage your design requires. **Label the bit width** of all storage.
- All input and output ports. For each port **label its width** and indicate during **which pipeline stage** the port is used (assume the R10K pipeline stages discussed in class: **F**etch, **D**ispatch, i**S**sue, e**X**ecute, **C**omplete, **R**etire). Ports used for exception rewind should be marked **D**ispatch.
- You do not need to show details of the combinational logic within the free list, just the inputs, outputs, and state.

(Note: this problem is harder than it looks at first glance; I suggest doing it **last**.)



e) In your free list design, briefly explain how single-cycle branch rewind is accomplished. How does your design ensure that the registers allocated by squashed instructions are restored to the free list in just one cycle? **[3 points]**

The BPRecoverEN signal overwrites head with BPRecoverHead, which backs the head pointer up, instantly restoring all the entries (still in array) that have been allocated since the branch instruction was dispatched.

6) MIPS R10K Execution. [20 points]

On the next pages, you will find a set of charts showing a snapshot of a MIPS R10K-like microarchitecture after one cycle executing a sequence of instructions. You must advance this machine 5 additional clock cycles (to the end of cycle #6). Use the cycle-by-cycle state tables to record the contents of each hardware structure at the end of each clock cycle.

Assume the following:

- Assume the machine is a **scalar** (i.e., it can issue, complete, and retire at most 1 instruction per cycle). If there are conflicts among instructions, the machine always selects the oldest instructions first.
- Ignore the fetch stage. Assume all instructions have been fetched and are ready for dispatch whenever the out-of-order core allows.
- This machine has 4 architectural registers, a 5-entry ROB, 4 reservation stations, and 9 physical registers. Reservation stations may hold any type of instruction.
- There is **1 add** functional unit with a **1-cycle** latency, and **1 pipelined load** functional unit with a **2-cycle** latency (i.e., 2 cycles in X stages; pipelined means that a new load can start when a preceding load reaches the 2nd X stage). Assume that loads do not cause cache misses.
- Assume no bypassing. A dependent instruction wakes only when its predecessor reaches C. Assume that any instructions in the C stage will bypass to S through the physical register file.
- The load and add units each have dedicated write ports on the register file, but there is only a single CDB to broadcast tags (i.e., the two units can't both do tag broadcast at the same time, but they may both be in the C stage). In the case of a CDB conflict, the load gets the CDB and the add proceeds to C stage; the add will then do its tag broadcast from C stage. If a load conflicts for CDB with an add in C stage, the load wins and the add pipeline stalls. If adds in both X and C need the CDB, the add in C wins.
- Assume reservation stations are freed as early as possible and can be reused as soon as they are freed.

Here is the instruction sequence:

(1) Load R3 = Mem[R1 + R2]
(2) R2 = R1 + 10
(3) Load R4 = Mem[R2 + R3]
(4) R3 = R1 + R2
(5) R1 = R4 + 10

To save you time in writing, the instructions have been pre-filled into the ROB in the solution sheet – be sure to update the head and tail pointers correctly to show when instructions are dispatched and retired. Cycle 1 has been completed for you. Pay attention to the cycle number on each chart—be sure you fill them out in correct order!

If you make a mistake and need additional blank copies of the fill-in sheet, ask the exam proctor. Make sure the old sheets are torn up and the new ones are stapled to your exam!!!

R10K Cycle # 1

ROB							
ht	#	Instruction	T	Told	S	X	C
th	1	Ld R3=M[R1+R2]	p5	p3			
	2	R2=R1+10					
	3	Ld R4=M[R2+R3]					
	4	R3=R1+R2					
	5	R1=R4+10					

Map Table	
Reg	T+
r1	p1+
r2	p2+
r3	p5
r4	p4+

Arch Map	
Reg	T+
r1	p1+
r2	p2+
r3	p3+
r4	p4+

Free List
~~p5~~,p6,p7, p8,p9

CDB
T

Reservation Stations				
#	instruction #	T	T1	T2
1	(1)	p5	p1+	p2+
2				
3				
4				

R10K Cycle # 2

ROB							
ht	#	Instruction	T	Told	S	X	C
h	1	Ld R3=M[R1+R2]	p5	p3	2		
t	2	R2=R1+10	p6	p2			
	3	Ld R4=M[R2+R3]					
	4	R3=R1+R2					
	5	R1=R4+10					

Map Table	
Reg	T+
r1	p1+
r2	p6
r3	p5
r4	p4+

Arch Map	
Reg	T+
r1	p1+
r2	p2+
r3	p3+
r4	p4+

Free List
~~p6~~,p7, p8,p9

CDB
T

Reservation Stations				
#	instruction #	T	T1	T2
1	(1)	p5	p1+	p2+
2	(2)	p6	~~p~~1+	
3				
4				

R10K Cycle # 3

ROB							
ht	#	Instruction	T	Told	S	X	C
h	1	Ld R3=M[R1+R2]	p5	p3	2	3	
	2	R2=R1+10	p6	p2	3		
t	3	Ld R4=M[R2+R3]	p7	p4			
	4	R3=R1+R2					
	5	R1=R4+10					

Map Table	
Reg	T+
r1	p1+
r2	p6
r3	p5
r4	p7

Arch Map	
Reg	T+
r1	p1+
r2	p2+
r3	p3+
r4	p4+

Free List
~~p7~~, p8,p9

CDB
T

Reservation Stations				
#	instruction #	T	T1	T2
1	~~(1)~~	~~p5~~	~~p1+~~	~~p2+~~
2	(2)	p6	p1+	
3	(3)	p7	p6	p5
4				

R10K Cycle # 4

ROB							
ht	#	Instruction	T	Told	S	X	C
h	1	Ld R3=M[R1+R2]	p5	p3	2	3-4	
	2	R2=R1+10	p6	p2	3	4	
	3	Ld R4=M[R2+R3]	p7	p4			
t	4	R3=R1+R2	p8	p5			
	5	R1=R4+10					

Map Table	
Reg	T+
r1	p1+
r2	p6
r3	p8
r4	p7

Arch Map	
Reg	T+
r1	p1+
r2	p2+
r3	p3+
r4	p4+

Free List
~~p8~~,p9

CDB
T

Reservation Stations				
#	instruction #	T	T1	T2
1				
2	~~(2)~~	~~p6~~	~~p1+~~	
3	(3)	p7	p6	p5
4	(4)	p8	p1+	p6

R10K Cycle # 5

ROB							
ht	#	Instruction	T	Told	S	X	C
h	1	Ld R3=M[R1+R2]	p5	p3	2	3-4	5
	2	R2=R1+10	p6	p2	3	4	
	3	Ld R4=M[R2+R3]	p7	p4			
	4	R3=R1+R2	p8	p5			
t	5	R1=R4+10	p9	p1			

Map Table	
Reg	T+
r1	p9
r2	p6
r3	p8
r4	p7

Arch Map	
Reg	T+
r1	p1+
r2	p2+
r3	p3+
r4	p4+

Free List
~~p9~~

CDB
T
p5

Reservation Stations				
#	instruction #	T	T1	T2
1	(5)	p9	p7	
2				
3	(3)	p7	p6	p5+
4	(4)	p8	p1+	p6

R10K Cycle # 6

ROB							
ht	#	Instruction	T	Told	S	X	C
	1	Ld R3=M[R1+R2]	p5	p3	2	3-4	5
h	2	R2=R1+10	p6	p2	3	4	6
	3	Ld R4=M[R2+R3]	p7	p4	6		
	4	R3=R1+R2	p8	p5			
t	5	R1=R4+10	p9	p1			

Map Table	
Reg	T+
r1	p9
r2	p6+
r3	p8
r4	p7

Arch Map	
Reg	T+
r1	p1+
r2	p2+
r3	p5+
r4	p4+

Free List
p3

CDB
T
p6

Reservation Stations				
#	instruction #	T	T1	T2
1	(5)	p9	p7	
2				
3	(3)	p7	p6+	p5+
4	(4)	p8	p1+	p6+