EECS 470 W25 Midterm Review Session Worksheet Answer Key

Peppa Pig's Power, Performance, and Pipelining Problems (PPPPPP):

The 5-stage in-order pipeline is modified such that branches are resolved in the WB stage instead of the MEM stage. Is this a safe modification? Why or why not.

This is not a safe modification. Mispredicted store instructions would still be able to write to memory.

Qualcomm is working to improve the energy-efficiency of its next-generation mobile processor. The baseline processor consumes 12W. 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 8W. Should they add the new low power mode? Why or why not?

They should add the new low power mode. Applying DVFS to the original processor with a 10% performance hit, we find that the original processor consumes (0.9^3)*(12W) = 8.748 W. The original processor would consume more power than the low power mode operating at the same performance level.

Consider a program where 25% 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. What is the speedup if parallelized across 8 cores?

Applying Amdahl's law with 75% of the program eligible for a factor of 8 speedup:

Speedup = 1 / (1-f)+(f/S) = 1 / (0.25 + 0.75/8) = 2.909

P6 Problems:

From where can an instruction receive its source register values (Hint: there are 3 places)? What about in R10K?

ARF (architected register file), CDB, and ROB

Bradley's #1 Fan Club is implementing a P6 architecture for their 470 final project. They want to write to the ARF on complete rather than commit. Is this a good idea? Why or why not?

This is not a good idea. Out-of-order writes to the register file makes us unable to recover the correct state of the processor on an exception or branch misprediction. We lose precise state.

Given the following state of a P6-style machine, walk the machine forward three cycles. Assume that all operations have a 1 cycle latency, and there are no structural hazards on functional units. The machine is 2-way superscalar. Values of ROB #0 and #2 are broadcasted on the CDB on cycle 3. The tail of the ROB points to the next free entry

There are two more instructions that need to be dispatched: Inst 4: R4 = R2 + R1 Inst 5: R4 = R4 + R4

After cycle 3:

ROB							
ht	Entry #	Inst #	R	V	S	X	C
ht	0	0	R1	[R1]	1	2	3
	1	1	R2		3		
	2	2	R3	[R3]	1	2	3
	3	3	R1		3		
	4	4	R4				

Map Table	
Reg	T
R1	ROB #3
R2	ROB #1
R3	ROB #2+
R4	ROB #4

Reservation Station						
Entry #	Inst #	T	T1	T2	V1	V2
1	3	ROB #3	ROB #0	ROB #2	CDB0.V	CDB1.V
2	1	ROB #1	ROB #0		CDB0.V	[R4]
3	4	ROB #4	ROB #1	ROB #3		

CDB	
T	V
ROB #0	[R1]
ROB #2	[R3]

After cycle 4:

ROB							
ht	Entry #	Inst #	R	V	S	X	C
	0	5	R4				
ht	1	1	R2		3	4	
	2	2	R3	[R3]	1	2	3
	3	3	R1		3	4	
	4	4	R4				

Map Table	
Reg	T
R1	ROB #3
R2	ROB #1
R3	ROB #2+
R4	ROB #0

Reservation Station						
Entry #	Inst #	T	T1	T2	V1	V2
1	5	ROB #0	ROB #4	ROB #4		
2						
3	4	ROB #4	ROB #1	ROB #3		

CDB	
T	V

After cycle 5:

ROB							
ht	Entry #	Inst #	R	V	S	X	C
	0	5	R4				
ht	1	1	R2		3	4	5
	2	2	R3	[R3]	1	2	3
	3	3	R1		3	4	5
	4	4	R4		5		

Map Table	
Reg	T
R1	ROB #3+
R2	ROB #1+
R3	ROB #2+
R4	ROB #0

Reservation Station						
Entry #	Inst #	T	T1	T2	V1	V2
1	5	ROB #0	ROB #4	ROB #4		
2						
3	4	ROB #4	ROB #1	ROB #3	CDB0.V	CDB1.V

CDB	
T	V
ROB #1	[R2]
ROB #3	[R1]

Note: ROB#1 and ROB#3 should have their values populated inside of the ROB for cycle 5

R10K Problems:

When is the register file written to?

On complete

When can a physical register be freed? Why can it be freed then?

A physical register can be freed when the instruction that remapped the architectural register that it was assigned to retires. It can be freed then because we are guaranteed that there exists no instruction in the processor that still requires the original mapping.

How are branch mispredictions handled (assume no early branch resolution/fast branch recovery)?

When the mis-predicted branch becomes the head of the ROB, we have the following options:

- Copy the architected map table to the map table and restore the free list with registers not mapped in the architected map table.
- Make a checkpoint when the branch is dispatched and restore to that checkpoint in the case that the branch is mispredicted.

- Rollback the ROB, undoing the mappings of the instructions after the branch, until the tail of the ROB reaches the head.

Given the following state of a R10K-style machine, walk the machine forward four cycles. The machine is 2-way superscalar. There is one add functional unit with a 1 cycle latency, and one fully pipelined multiplier with a 2-cycle latency. The ROB has been prepopulated with the instructions to be executed. There are no further instructions. Feel free to cross out/replace entries in the given tables. Start at cycle 1.

After cycle 1:

ROB							
ht	Entry #	Instruction	T	Told	S	X	C
h	1	R1 = R1 + R2	P5	P1			
	2	R2 = R3 * R4	P6	P2			
t	3	R3 = R1 + R1					
	4	R4 = R4 * R4					

Map Table	
Reg	T+
R1	P5
R2	P6
R3	P3+
R4	P4+

Arch. Map Table	
Reg	T
R1	P1
R2	P2
R3	P3
R4	P4

Reservation Station				
Entry #	Operation	T	T1	T2
1	+	P5	P1+	P2+
2	*	P6	P3+	P4+
3				

CDB
T

Free List
P7, P8

After cycle 2:

ROB							
ht	Entry #	Instruction	T	Told	S	X	C
h	1	R1 = R1 + R2	P5	P1	2		
	2	R2 = R3 * R4	P6	P2	2		
	3	R3 = R1 + R1	P7	P3			
t	4	R4 = R4 * R4					

Map Table	
Reg	T+
R1	P5
R2	P6
R3	P7
R4	P4+

Arch. Map Table	
Reg	T
R1	P1
R2	P2
R3	P3
R4	P4

Reservation Station				
Entry #	Operation	T	T1	T2
1	+	P5	P1+	P2+
2	*	P6	P3+	P4+
3	+	P7	P5	P5

CDB
T

Free List
P8

After cycle 3:

ROB							
ht	Entry #	Instruction	T	Told	S	X	C
ht	1	R1 = R1 + R2	P5	P1	2	3	
	2	R2 = R3 * R4	P6	P2	2	3+	
	3	R3 = R1 + R1	P7	P3			
	4	R4 = R4 * R4	P8	P4			

Map Table	
Reg	T+
R1	P5
R2	P6
R3	P7
R4	P8

Arch. Map Table	
Reg	T
R1	P1
R2	P2
R3	P3
R4	P4

Reservation Station				
Entry #	Operation	T	T1	T2
1	*	P8	P4+	P4+
2				
3	+	P7	P5	P5

CDB
T

Free List

After cycle 4:

ROB							
ht	Entry #	Instruction	T	Told	S	X	C
ht	1	R1 = R1 + R2	P5	P1	2	3	4
	2	R2 = R3 * R4	P6	P2	2	3+	
	3	R3 = R1 + R1	P7	P3	4		
	4	R4 = R4 * R4	P8	P4	4		

Map Table	
Reg	T+
R1	P5+
R2	P6
R3	P7
R4	P8

Arch. Map Table	
Reg	T
R1	P1
R2	P2
R3	P3
R4	P4

Reservation Station				
Entry #	Operation	T	T1	T2
1	*	P8	P4+	P4+
2				
3	+	P7	P5+	P5+

CDB
T
P5

Free List