

CMSC 411 Homework 4 (Fall 2009)

1. Branch prediction

Consider the following code fragment:

```
if( d == 1 )
    d = 2;
if( d == 2 )
    d = 3;
if( d == 3)
    d = 4;
```

A typical code sequence generated for this fragment, assuming that `d` is assigned to `R1`, looks like the following:

```
        DSUBUI R2, R1, #1
        BNEZ  R2, L1    ; Branch B1
        DADDIU R1, R0, #2
L1:     DSUBUI R3, R1, #2
        BNEZ  R3, L2    ; Branch B2
        DADDIU R1, R0, #3
L2:     DSUBUI R4, R1, #3
        BNEZ  R4, L3    ; Branch B3
        DADDIU R1, R0, #4
        . . .
L3:
```

For the following questions, show for 6 loop iterations the state of the branch predictors (and branch history tables, if needed), the predictions, and the actual branch actions. Report the number of mis-predicted branches. Assume that all predictors are initialized to not taken, and that the correlation bits are initially set to not taken. When multiple predictors may be used, circle (or underline) the predictors used to make a prediction.

- Construct a (1, 1) predictor w/global history + branch address. Show results with the value of `d` alternating between 2 and 0 on succeeding loop iterations. I.e., with the value of `d` = {2 0 2 0 2 0}.
- Construct a (0, 2) predictor w/saturating counter. Show results with the value of `d` = {0 0 2 2 2 2}.
- Construct a (1, 1) predictor w/local history + branch address. Show results with the value of `d` = {0 0 2 2 2 2}.
- Construct a (2, 1) global predictor (no branch address). Show results with the value of `d` = {0 0 2 2 2 2}.

- (e) Construct a 2-bit tournament predictor (w/ saturating counter) choosing between the (1,1) local+branch and (2,2) global predictors. Show results with the value of $d = \{0\ 0\ 2\ 2\ 2\ 2\}$.

Iteration	d=?	B1 state	B1 prediction	B1 action	B2 state	B2 prediction	B2 action	B3 state	B3 prediction	B3 action
1	2									
2	0									
3	2									
4	0									
5	2									
6	0									
7										

Iteration	d=?	B1 state	B1 prediction	B1 action	B2 state	B2 prediction	B2 action	B3 state	B3 prediction	B3 action
1	0									
2	0									
3	2									
4	2									
5	2									
6	2									
7										

2. Dynamic scheduling

The following code implements the vector operation $Y = aX + Y$ for a vector of length 100, and assumes that initially $R1=0$ and $F0$ contains the constant a :

```
foo:  L.D      F2,0(R1)      ;load X(i)
      MUL.D   F4,F2,F0     ;multiply a*X(i)
      L.D      F6,0(R2)    ;load Y(i)
      ADD.D   F6,F4,F6     ;add a*X(i)+Y(i)
      S.D      F6,0(R2)    ;store Y(i)
      DADDUI  R1,R1,#8     ;increment X index
      DADDUI  R2,R2,#8     ;increment Y index
      DSGTUI  R3,R1,#800   ;test if done
      BEQZ    R3,foo       ;loop if not done
```

The DSGTUI instruction is an integer instruction that sets the result register, R3, to a non-zero value if the value in R1 is greater than 800.

You are going to describe the execution of the loop for a single-issue Tomasulo-style MIPS pipeline, with Integer, FP adder, and FP multiplier functional units.

Assume the following:

- Functional units are not pipelined.
- No forwarding between functional units; results can only come from the CDB.
- The EX stage does both the effective address calculation and the memory access for loads and stores. So the pipeline is IF/ID/IS/EX/WB.
- The issue(IS) and write result (WB) stages each take 1 clock cycle.
- There are 5 load buffer slots and 5 store buffer slots.
- The BEQZ instruction takes 0 clock cycles.

For each problem, show the number of stall cycles for each instruction and what clock cycle each instruction begins execution (i.e., enters its first EX cycle), for two iterations of the loop. Also show when each instruction writes the CDB (store and branch instructions don't write the CDB). Describe how many clock cycles the 1st and succeeding loop iterations take. (may take longer).

- (a) Describe the execution of the processor for the following architecture. Assume loads take 1 clock cycle.

FU type	Cycles in EX	# of FUs	# of reservation stations
Integer	1	1	5
FP adder	3	1	3
FP multiplier	10	1	2

- (b) Describe the execution of the processor for the following architecture. Assume loads take 2 clock cycles.

FU type	Cycles in EX	# of FUs	# of reservation stations
Integer	1	1	5
FP adder	4	1	3
FP multiplier	8	1	2

You can use the following table as a template.

3. Cache

- (a) Suppose we have a byte addressable memory of size 4GB(2^{32} bytes) with a cache of size 128KB (2^{17} bytes), not including tag bits. Also suppose the cache block size is 64 bytes. For each of the following cache organizations, compute the length in number of bits for the tag, index and offset fields of the 32-bit memory address (show your calculations):
- direct mapped
 - 2-way set associative
 - fully associative
- (b) Given the following parameters, which performs better in terms of average memory access time, the direct mapped or the 2-way set associative cache? Assume a cache hit takes 1 cycle for either organization, and the cache miss penalty is 20 nanoseconds (both use the same memory system). Justify your answer.
- Direct mapped - miss rate 8%, 1GHz clock
2-way set associative - miss rate 5%, 900MHz clock
- (c) Consider adding a fast second-level cache to a computer with only a single level cache. Suppose data can be accessed from the second level cache 8 times faster than it can be accessed from the original cache, and that the second level cache can be used 25% of the time. How much speedup can we gain by adding the second-level cache? Assume all data accesses hit in the original cache.