Name _____

Computer Architecture
EE 4720
Final Examination

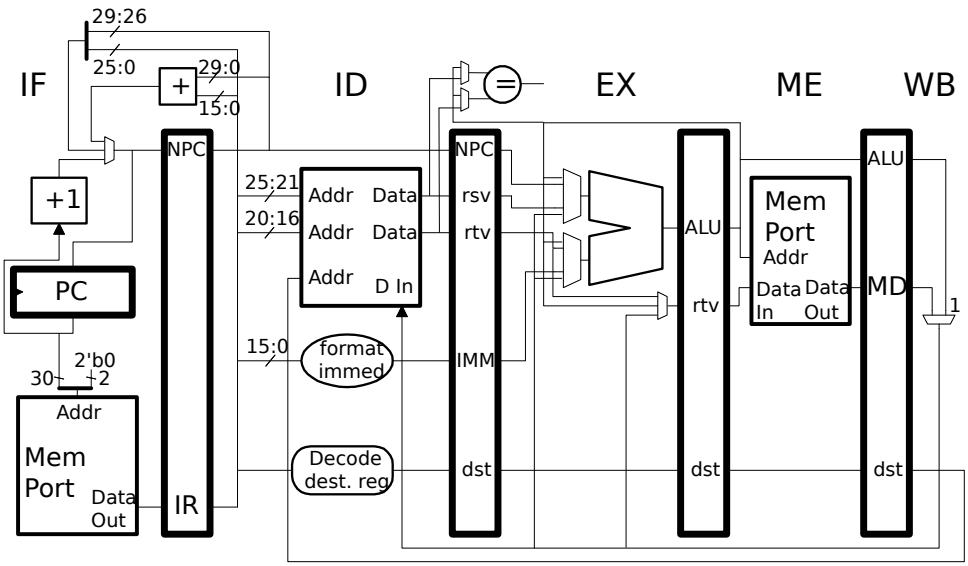
9 May 2011, 15:00–17:00 CDT

Problem 1	_____	(20 pts)
Problem 2	_____	(10 pts)
Problem 3	_____	(20 pts)
Problem 4	_____	(20 pts)
Problem 5	_____	(30 pts)
Exam Total	_____	(100 pts)

Alias _____

Good Luck!

Problem 1: (20 pts) The MIPS implementation below is similar to the one frequently used in class, except that it has bypass paths to the branch condition comparison unit.



(*a*) Show the execution of the code fragments below on the illustrated implementation up until the fetch of the first instruction of the second iteration. Be sure to account for the dependence on the branch condition.

☐ Pipeline execution diagram of code below.

```
LOOP1:
 lw r1, 0(r2)

 addi r2, r2, 4

 bne r2, r3 LOOP1

 add r4, r4, r1
```

☐ Pipeline execution diagram of code below.

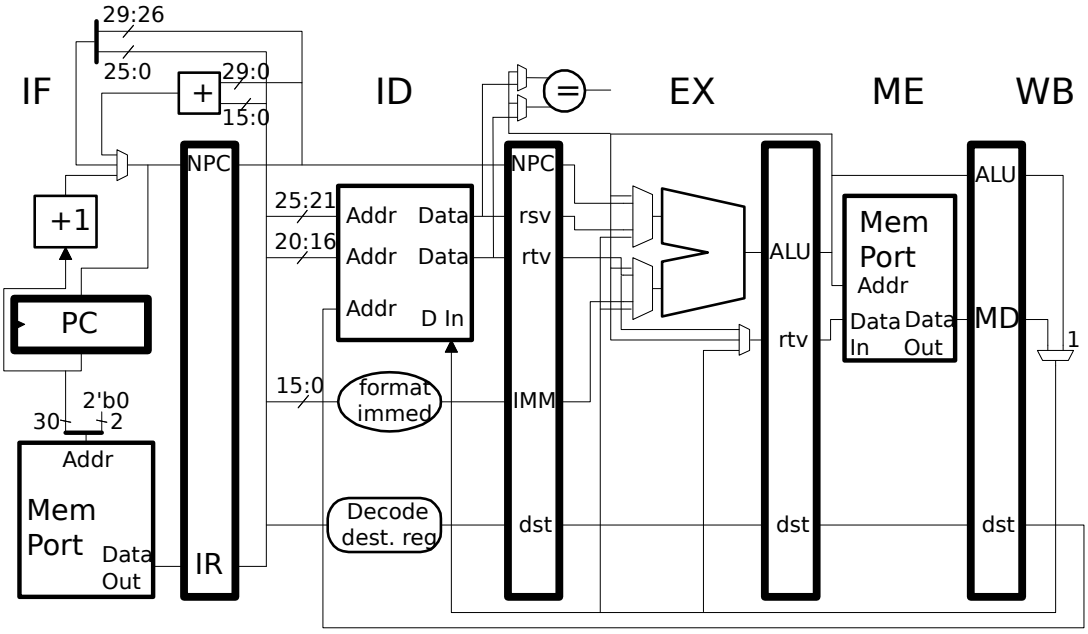
```
LOOP2:
 lw r1, 0(r2)

 lw r2, 4(r2)

 bne r2, r0 LOOP2

 add r4, r4, r1
```

(*b*) The implementation below has hardware in `IF` (not shown) to predict branches. Since it has prediction hardware the branch resolution hardware (the hardware that computes the branch condition and branch target) could be moved to the `ME` stage. Move the branch resolution hardware. Note that the resolution hardware will be used only if the branch is mispredicted.



☐ Move resolution hardware to the `ME` stage.

☐ Add any bypasses to resolution hardware needed by code samples in this problem.

☐ Cross out unused hardware in `ID`.

☐ Avoid adding unnecessary hardware.

Problem 1, continued: Consider the implementation from the previous part.

(*c*) Resolving the branch in `ME` with prediction can hurt performance with the code below compared to resolving in `ID` with prediction when the branch is hard to predict. By how much will it hurt performance? *Note: An exact number was not required on the original exam.* Explain, use a diagram if necessary. *Hint: Resolve is where recovery starts for a misprediction. Another hint: pay attention to dependencies on the branch condition.*

```
LOOP1:
 lw r1, 0(r2)
 addi r2, r2, 4
 bne r2, r3 LOOP1
 add r4, r4, r1
```

☐ Resolving in `ME` hurts performance here by ____ cycles.

☐ Explanation.

(*d*) Resolving the branch in `ME` with prediction should help performance with the code below compared to resolving in `ID` with prediction. Explain why, preferably with an execution diagram. *Hint: Same hints as previous part.*

```
LOOP2:
 lw r1, 0(r2)
 lw r2, 4(r2)
 bne r2, r0 LOOP2
 add r4, r4, r1
```

☐ Resolving in `ME` helps performance because...

Problem 2: (10 pts) The diagrams below show the execution of code on two-way superscalar dynamically scheduled systems of the type described in class. Each diagram **has mistakes**. Identify and fix them. Also explain why code should not execute as illustrated. The answer should specify what would go wrong, or why the system would be particularly inefficient, as appropriate.

(*a*) Find and fix the two problems with commit in the execution below, and explain why execution would be incorrect or inefficient.

```
# Cycle          0  1  2  3  4  5  6  7  8
lw r1, 0(r2)     IF ID Q  RR EA ME WB C
sub r4, r5, r6   IF ID Q  RR EX WB C
or  r7, r4, r8      IF ID Q  RR EX WB    C
# Cycle          0  1  2  3  4  5  6  7  8
```

☐ Fix the two problems above.

☐ Describe one error or inefficiency with execution.

☐ Describe another error or inefficiency with execution.

(*b*) Find and fix the problem with the code below. Note that the processor has two FP adders, named A and B. *Hint: Check dependencies.*

```
add.d f0, f2, f4      IF ID Q  RR A1 A2 A3 A4 WB C
sub.d f6, f0, f8      IF ID Q              RR A1 A2 A3 A4 WB C
add.d f10, f11, f12      IF ID Q           RR B1 B2 B3 B4 WB C
addi r1, r1, 4           IF ID Q              RR EX WB         C
```

☐ Fix the problem above.

☐ Describe one error or inefficiency with execution.

Problem 3: (20 pts) Code producing the branch patterns shown below is to run on three systems, each with a different branch predictor. All systems use a 2^{14}-entry BHT. One system uses a bimodal predictor, one system uses a local predictor with a 10-outcome local history, and one system uses a global predictor with a 10-outcome global history. The outcomes of branch `B1` form a repeating pattern, the outcome of `B2` can be modeled by a Bernoulli random variable with $p = .7$ (the branch is taken with probability .7), and `B3` is always taken.

```
B1:  N   T   T   T   T   T   N   N     N   T   T   T   T   T   N   N
B2:    R   R   R   R   R   R   R   R     R   R   R   R   R   R   R   R
B3:     T   T   T   T   T   T   T   T     T   T   T   T   T   T   T   T
```

For the questions below accuracy is after warmup.

☐ What is the accuracy of the bimodal predictor on `B1`?

☐ What is the approximate accuracy of the bimodal predictor on `B2`? Explain. *Hint: A correct answer would be slightly more or less (pick one) than.... For the exact answer one would need the state probability formula for a Markov chain.*

☐ What is the accuracy of the local predictor on branch `B1`?

☐ What is the warmup time of the global predictor on branch `B1`? Be sure to consider the effect of `B2`.

☐ What is the minimum local history size needed for a local predictor to predict `B1` with 100% accuracy?

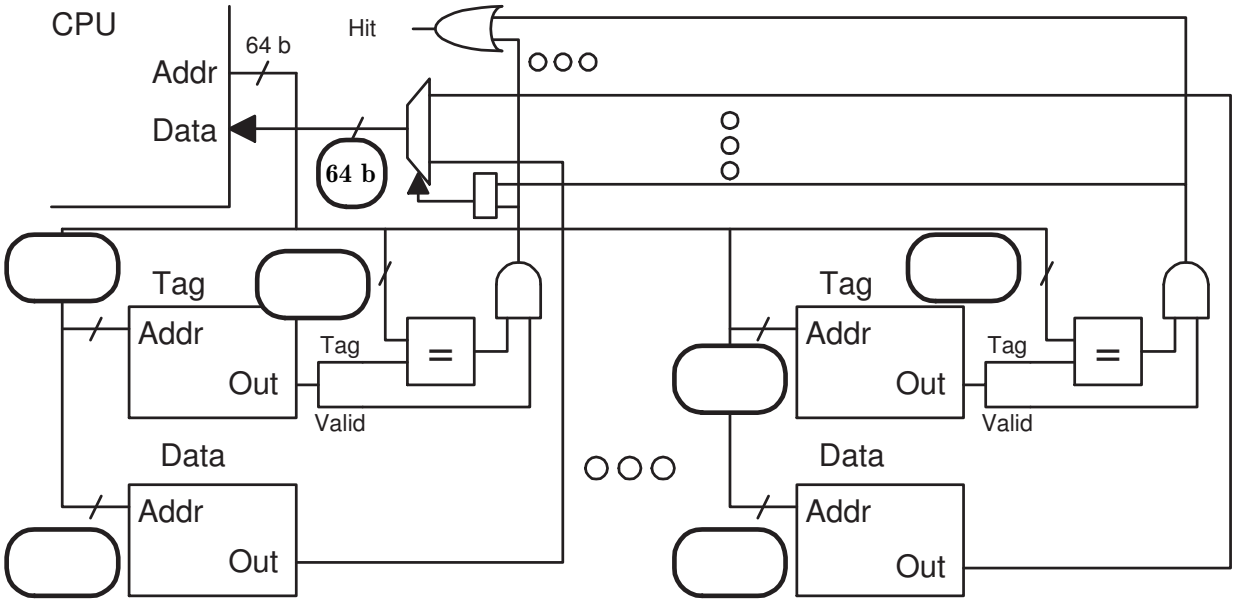
☐ What is the accuracy of a global predictor with a three-outcome global history on branch `B1`?

☐ What is the minimum global history size needed for a global predictor to predict `B1` with 100% accuracy?

Problem 4: (20 pts) The diagram below is for a set-associative cache with a capacity of 16 MiB (2^{24} bytes), and a line size of 64 bytes. The system has the usual 8-bit characters. Each data store below has a capacity of 2^{20} bytes.

(*a*) Answer the following, formulæ are fine as long as they consist of grade-time constants.

☐ Fill in the blanks in the diagram.



☐ Show the address bit categorization. Label the sections appropriately. (Alignment, Index, Offset, Tag.)

Address: | | | |

☐ Associativity:

☐ Memory Needed to Implement (Indicate Unit!!):

☐ Show the bit categorization for a direct mapped cache with the same capacity and line size.

Address: | | | |

Problem 4, continued: The problems on this page are **not** based on the cache from the previous page. The code fragments below run on a 32 MiB (2^{25} byte) direct-mapped cache with a 256-byte line size. Each code fragment starts with the cache empty; consider only accesses to the arrays, `a`, `ax`, and `ay`.

(*b*) Find the hit ratio executing the code below.

☐ What is the hit ratio running the code below? Explain

```
double sum = 0.0;
double *a = 0x20000000; // sizeof(double) == 8
int i;
int ILIMIT = 1 << 11;    // = 2^11

for (i=0; i<ILIMIT; i++) sum += a[ i ];
```

(*c*) The two code fragments below, labeled Method 1 and Method 2, perform the same computation but organize the data differently.

```
// Method 1
 struct Our_Struct { int x;  int y; };
 Our_Struct a[SIZE];

 for (int i=0; i<SIZE; i++) sum += a[ i ].x;
 for (int i=0; i<SIZE; i++) myfunc(sum,a[ i ].y);

// Method 2
 int ax[SIZE];
 int ay[SIZE];

 for (int i=0; i<SIZE; i++) sum += ax[ i ];
 for (int i=0; i<SIZE; i++) myfunc(sum,ay[ i ]);
```

☐ Which method has a higher hit ratio if the cache is small (or `SIZE` is large)? Explain

Problem 5: (30 pts) Answer each question below.

(*a*) Eliminating bypass paths from an n-way superscalar statically scheduled processor would have a much larger effect than eliminating them from a scalar statically scheduled processor (like our 5-stage MIPS implementation).

☐ Describe a positive benefit of eliminating the bypass paths that's much larger for the n-way superscalar.

☐ Describe a disadvantage of eliminating the bypass paths that's a much bigger disadvantage for the n-way superscalar.

(*b*) Consider a deeply pipelined system with $5n$ stages without bypass paths and that runs programs in which dependent instructions are far apart. Name two factors that will limit clock frequency for larger values of n. *Note: bypass paths were not mentioned in the original exam.*

☐ One frequency limiter.

☐ Another frequency limiter.

(*c*) Consider the BHT of a bimodal branch predictor. An entry would contain a CTI type, a two-bit counter, a branch target, and perhaps a tag. The tag would be used like a cache tag, but does not need to be as large. For this problem only consider branches.

☐ How large would the target need to be to predict MIPS branches? *Hint: The answer is not 32 bits.*

☐ Explain what the predictor can do with the tag to improve prediction accuracy.

(*d*) A four-way statically scheduled processor is much less expensive than a four-way dynamically scheduled processor. Why would the dynamically scheduled processor run some code faster than the statically processor even when the code was compiled with a good compiler that targeted the specific four-way implementation?

☐ Specific advantage of dynamically scheduled processor for some code.

☐ Describe properties of code that would run at the same rate on the two processors.

(*e*) In many VLIW ISAs the bundle size is 3 slots. Describe a disadvantage of making a VLIW ISA with a larger bundle size.

☐ Disadvantage of larger bundle size.

(*f*) A tester measuring the performance of a system using SPECcpu has the source code to SPECcpu programs. Why is that important to the goals of the suite?

☐ Source code important to SPECcpu goals?