ENGR 3410: The Final

Mark L. Chang

December 12, 2005

Instructions

This is a test. This is only a test. Actually, it can hardly even be called a real test. In fact, I think this test would shake with fear if it encountered a real test in a dark alley. This is really just a couple homeworks cleverly stapled together.

This test has no time limit, just an endpoint. My time estimate for you is around 2 hours. You can take this exam in one sitting, many sittings, even standing up or laying down. You may use any sources for solving your problems except for other people. You may use the internet, Google^{TM}, a calculator, lecture notes, and perhaps most importantly, the textbook. Just work *alone*.

The exam is due at 7PM, Thursday, December 15th, 2005. You may slide the exam under my door in East Hall or put it in my office in OC 357. I will also gladly accept email submissions of typeset solutions. Partial credit is given, so please show all work. Be clear with any assumptions you make.

I will be out of town from Wednesday morning until midnight Thursday. I will try and answer questions, but it will be difficult. Emails will work better than instant messages.

Problem 0

Please complete the following questions with a short, 1-3 sentence answer.

1. Which has a higher clock rate: a single-cycle CPU or a pipelined CPU?

2. How do we achieve this higher clock rate?

3. Why do modern high-performance CPUs need caches?

Problem 1

Consider this code fragment:

```
add  $3, $2, $1
lw   $4, 100($3)
sub  $6, $4, $3
add  $2, $3, $6
xor  $3, $8, $1
addi $5, $1, 100
```

On the *pipelined processor* designed in class:

1. How many cycles will it take to execute this code as written?
2. What is the minimum number of cycles it will take to execute this code?

Hint: You will have to write some assumptions down for this problem.

Problem 2

We have a program that contains 3 million instructions. 35% are ALU, 35% are stores, 25% are loads, and 5% are branches.

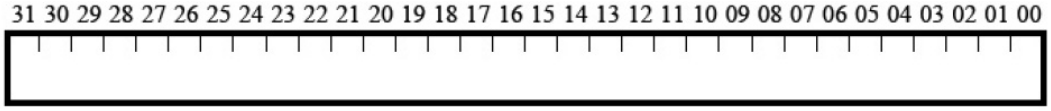
1. For a pipelined CPU as designed in class, with a clock period of 1ns, what is the execution time? Assume 50% of the delay slots are filled by the compiler. Ignore pipeline fill/drain effects.

2. If none of the delay slots can be filled on the pipeline CPU, what is the speedup compared to Part 1 of this problem?

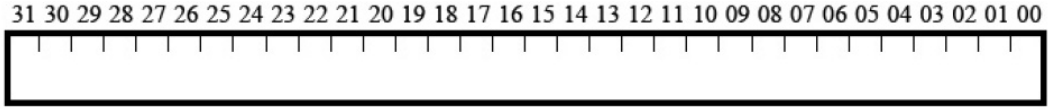
Problem 3

Assume a 32-bit memory address for our MIPS processor. For the following configurations, show the location and size for the byte select, cache index, and cache tag.

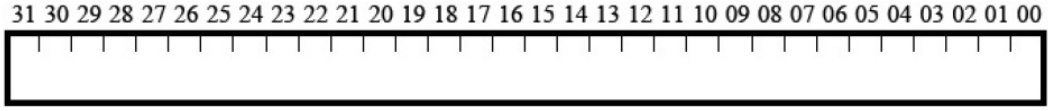
1. 16 kilobyte cache, 64 byte blocks, direct mapped



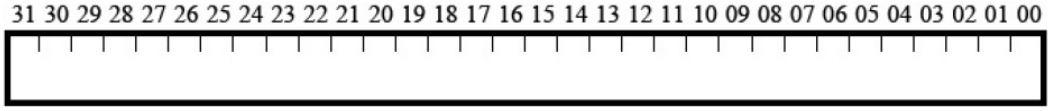
2. 16 kilobyte cache, 64 byte blocks, 2-way set associative



3. 32 kilobyte cache, 128 byte blocks, 4-way set associative



4. 32 kilobyte cache, 128 byte blocks, fully associative



Problem 4

Assume that the code below is loaded into memory location 0, $a1 = 4, and $a0 = 100.

```
sll $t0, $a1, 2
add $t0, $t0, $a0
lw  $t1, 0($t0)
lw  $t2, 4($t0)
sw  $t2, 0($t0)
sw  $t1, 4($t0)
```

Create a table with two columns. First column is **Address**, second column is **Cache miss type or hit**. Fill in this table as described below:

1. Show the sequence of memory addresses generated by the CPU when executing this code sequence. This is the left-hand side of the table.
2. Assume a direct-mapped cache of 32 bytes with 4 byte blocks. Assume a completely invalid and empty cache at the start of execution. Fill in the right-hand portion of your table with *hit* or *miss*, and indicate the type of cache miss if applicable.
3. Show the contents of the cache (data portion only) after executing the entire instruction sequence *in a separate table*.