CS 61C Final Solution — August 10, 2000

Your name ____

login cs61c–_______

TA name ____

This exam is worth 40 points, or 40% of your total course grade. The exam contains 4 substantive questions, plus the following:

Question 0 (1 point): Fill out this front page correctly and put your name and login correctly at the top of each of the following pages. **DO IT NOW!**

This booklet contains 12 numbered pages including the cover page. Put all answers on these pages, please; don't hand in stray pieces of paper. The exam is open book, open notes, but close calculators.

Our expectation is that many of you will not complete one or two parts of the exam. If you find one question especially difficult, leave it for later; start with the ones you find easier.

Please read all the instructions carefully before you start writing the answers. If any part of the exam is unclear, raise your hand and one of the staff members will go assist you. Good luck.

0	/1
1	/3
2	/18
3	/6
4	/12
total	/40

READ AND SIGN THIS:

I certify that my answers to this exam are all my own work, and that I have not discussed the exam questions or answers with anyone prior to taking this exam.

____

Your name ________________________________ login cs61c–______

Question 1 (1 point per question):

Grading: 1 point per question, all or nothing

1a) Convert the value `3411` from base 5 to base 10.

481

1b) Given the following C declaration:

```
int A[3][5] = {{0,1,2,3,4}, {1,3,5,7,9}, {0,1,1,2,3}};
```

Give 3 unique ways to extract the value 3 out of the array.

A[0][3], A[1][1], A[2][4]

1c) Assume **charArray** is the memory location for a character array with 20 elements, and **intArray** is the memory location for a integer array with 20 elements. Give the memory locations for element 12 of both arrays.

Because the wording is a bit ambiguous, we accept both charArray+11 and charArray+12 for the charArray, and intArray+44 and intArray+48 for the intArray.

Your name ____ login cs61c–_______

Question 2 (2 points per question):

Grading: 2 points per question, all or nothing unless otherwise noted.

2a) You forgot to write the **RFE** instruction at the end of your exception handler. How can a malicious program potentially exploit this bug?

The malicious program will have kernel privileges after the exception handler returns.

2b) To get the input from a mouse, is it better to use polling or interrupt? Explain your reasoning in 20 words or less.

Polling is better, because the I/O rate for mouse is low, the overhead to the processor is insignificant.

2c) Give an example of when your program will experience low spatial locality, and an example of when your program will experience low temporal locality.

Grading: 1 point for spatial, 1 point for temporal.

low spatial locality: linked list. low temporal locality: program with no loops and only uses data once.

Your name ____ login cs61c–_______

Question 2 continued:

2d) Let's modify the instruction set architecture and remove the ability to specify an offset for memory access instructions. Specifically, all load-store instructions with nonzero offsets would become pseudoinstructions and would be implemented using two instructions. For example,

```
addi $at, $t1, 104  # add the offset to a temporary
lw $t0, $at         # new way of doing lw $t0, 104($t1)
```

In 20 words or less, explain what changes you would make to the datapath to improve latency, and why.

Grading: 1 point for modified datapath, 1 point for explanation.

Combine ALU and memory stages in datapath for a total of 4 stages, because no instruction needs all 5 stages now.

2e) In project 5, you saved $at at the beginning of the interrupt handling routine, and restored it at the end. In 20 words or less, explain why.

£at may be changed by the assembler if any pseudoinstructions is used.

Your name ____ login cs61c–_______

Question 2 continued:

2f) Given the following instruction mix, what is the CPI for this processor?

Operation	Frequency	CPI
A	50%	1
B	15%	4
C	15%	3
D	10%	4
E	5%	1
F	5%	2

2.1

2g) Given the following set of MIPS instructions: (**add ori srl lw sw lui bne slt j jal**), list the ones that do **not** use

Grading: 1 point for correct first, second, and fourth stages, 1 point for correct third and fifth stages.

first stage of pipeline: none

second stage of pipeline: none

third stage of pipeline: bne, j, jal

fourth stage of pipeline: add ori srl lui bne slt j jal

fifth stage of pipeline: bne, j, sw

Question 2 continued:

2h) Construct a 1-bit comparator using the minimum number of 2-input **AND** gates, 2-input **OR** gates, 2-input **muxes**, and 1-input **INV** gates. The inputs to the comparator are the two bits being compared, and a control bit. If the control bit is 1, implement the EQUAL function: the comparator returns 1 if and only if the two inputs are equal. If the control bit is 0, implement the NEQUAL function: comparator returns 1 if and only if the two inputs are not equal. Label your inputs **A**, **B**, and **Control**; label your output **Out**. Please be neat and label all your inputs and outputs clearly.

Grading: 1 point if your solution works, 1 point if you use 7 gates or less

```
        ---
A ----O|   |
       |AND|---+
B -----|___|   |   __
               +--|  |      ___
                  |OR|--+--|   |
        ___    +--|__|  |  |MUX|--- Out
A -----|   |   |        +-O|___|
       |AND|---+             |
B ----O|___|                 |
                          Control
```

2i) Using one 2-input multiplexor, construct a 2-input **AND** gate. Label your inputs **A** and **B**, label your output **Out**. Please be neat and label all inputs and outputs clearly.

```
      ---
A ---|   |
     |MUX|--- Out
B ---|___|
       |
       |
       A
```

Your name ________________________________ login cs61c–______

Question 3 (3 points per question):

3a) The following piece of code has pipeline hazard(s) in it.

```
haz:    move    $5, $0
        lw      $10, 1000($20)
        addiu   $20, $20, -4
        addu    $5, $5, $10
        bne     $20, $0, haz
```

Reorder the instructions and insert the **least number of NOOPs** to make it hazard-free. Do not change the instructions themselves, and assume all necessary forwarding logics exist.

Grading: -1 if solution has hazard, -1 if any noop is used, -2 if branch delay slot is not filled, -2 if resulted program flow is not same as original.

```
haz: lw    $10, 1000($20)
     addiu $20, $20, -4
     move  $5, $0
     bne   $20, $0, haz
     addu  $5, $5, $10
```

Your name ______________________________ login cs61c–______

3b) Given the following virtual memory system:

```
52-bit virtual address
32KB pages
32-bit physical byte address
4-way set associative TLB
256 TLB entries.
```

Show the virtual-to-physical mapping in the diagram below, make sure to label all fields as well as the width of all fields; draw arrows to indicate which bits go where.

1 point for virtual address breakdown, 1 point for TLB, 1 point for physical address breakdown.

```
Arrows: tag in virtual address to virtual page number in TLB; index
in virtual address points into left side of TLB; physical page number
from TLB points to physical page number in physical address; offset
from virtual address points to offset from physical address.

Virtual address
---------------------------------------------
|      tag       |   index    |    offset   |
---------------------------------------------
    31 bits         6 bits       15 bits

TLB
-------------------------------------------------------------------------
|  Virtual page number              |        Physical page number        |
|-----------------------------------+------------------------------------|
|                                   |                                    |
|                                   |                                    |
-------------------------------------------------------------------------
          31 bits                                17 bits

Physical address
---------------------------------------------
|  Physical page number       |    offset   |
---------------------------------------------
         17 bits                  15 bits
```

Question 4 (4 points):

4a) In your MIPS machine, you are given a 2-way set-associative cache with 2-word blocks, and a total size of 32 words. Assume that the cache is initially empty, and that it uses an LRU replacement policy.

Given the following memory accesses in sequence, label whether they'll be hits or misses. Calculate the hit rate as well.

Grading: 1 point per 4 sequential accesses, 1 point for hit rate

```
0ff00f70 miss
0ff00f60 miss
0fe0012c miss
0ff00f5c miss
0fe0012c hit
0fe001e8 miss
0f000f64 miss
0f000144 miss
0fe00204 miss
0ff00f74 hit
0f000f64 hit
0f000128 miss

Hit rate: 25 %
```

Your name ____ login cs61c–_______

Question 4 continued:

4b) Answer questions on the next page according to the following output from cache.c (lab 8), assume there is only one level of cache in this system:

```
Size:   4096 Stride:      4 read+write: 100 ns
Size:   4096 Stride:      8 read+write: 99 ns
Size:   4096 Stride:     16 read+write: 100 ns
Size:   4096 Stride:     32 read+write: 101 ns
Size:   4096 Stride:     64 read+write: 100 ns
Size:   4096 Stride:    128 read+write: 100 ns
Size:   4096 Stride:    256 read+write: 100 ns
Size:   4096 Stride:    512 read+write: 100 ns
Size:   4096 Stride:   1024 read+write: 99 ns
Size:   4096 Stride:   2048 read+write: 101 ns
Size:   8192 Stride:      4 read+write: 100 ns
Size:   8192 Stride:      8 read+write: 101 ns
Size:   8192 Stride:     16 read+write: 99 ns
Size:   8192 Stride:     32 read+write: 100 ns
Size:   8192 Stride:     64 read+write: 101 ns
Size:   8192 Stride:    128 read+write: 101 ns
Size:   8192 Stride:    256 read+write: 100 ns
Size:   8192 Stride:    512 read+write: 99 ns
Size:   8192 Stride:   1024 read+write: 100 ns
Size:   8192 Stride:   2048 read+write: 100 ns
Size:   8192 Stride:   4096 read+write: 101 ns
Size:  16384 Stride:      4 read+write: 149 ns
Size:  16384 Stride:      8 read+write: 201 ns
Size:  16384 Stride:     16 read+write: 302 ns
Size:  16384 Stride:     32 read+write: 601 ns
Size:  16384 Stride:     64 read+write: 600 ns
Size:  16384 Stride:    128 read+write: 601 ns
Size:  16384 Stride:    256 read+write: 601 ns
Size:  16384 Stride:    512 read+write: 599 ns
Size:  16384 Stride:   1024 read+write: 600 ns
Size:  16384 Stride:   2048 read+write: 600 ns
Size:  16384 Stride:   4096 read+write: 600 ns
Size:  16384 Stride:   8192 read+write: 100 ns
Size:  32768 Stride:      4 read+write: 149 ns
Size:  32768 Stride:      8 read+write: 201 ns
Size:  32768 Stride:     16 read+write: 301 ns
Size:  32768 Stride:     32 read+write: 600 ns
```

Your name ________________________________ login cs61c–______

Question 4 continued:

```
Size:  32768 Stride:     64 read+write: 601 ns
Size:  32768 Stride:    128 read+write: 600 ns
Size:  32768 Stride:    256 read+write: 600 ns
Size:  32768 Stride:    512 read+write: 600 ns
Size:  32768 Stride:   1024 read+write: 601 ns
Size:  32768 Stride:   2048 read+write: 600 ns
Size:  32768 Stride:   4096 read+write: 601 ns
Size:  32768 Stride:   8192 read+write: 400 ns
Size:  32768 Stride:  16384 read+write: 100 ns
Size:  65536 Stride:      4 read+write: 150 ns
Size:  65536 Stride:      8 read+write: 199 ns
Size:  65536 Stride:     16 read+write: 303 ns
Size:  65536 Stride:     32 read+write: 600 ns
Size:  65536 Stride:     64 read+write: 600 ns
Size:  65536 Stride:    128 read+write: 600 ns
Size:  65536 Stride:    256 read+write: 600 ns
Size:  65536 Stride:    512 read+write: 600 ns
Size:  65536 Stride:   1024 read+write: 600 ns
Size:  65536 Stride:   2048 read+write: 600 ns
Size:  65536 Stride:   4096 read+write: 600 ns
Size:  65536 Stride:   8192 read+write: 600 ns
Size:  65536 Stride:  16384 read+write: 399 ns
Size:  65536 Stride:  32768 read+write: 100 ns
```

Grading: 1 point for cache & block size, 1 point for times, 1 point for associativity, 1 point for replacement policy

What is the cache size? 8KB

What is the block size? 32 bytes

What is the miss time? 600 ns

What is the hit time? 100 ns

What is the set associativity? 2-way

What is the cache replacement policy? random

Your name ____ login cs61c–______

Question 4 continued:

4c) Trace a jal instruction through the datapath stages, starting with an empty cache and TLB. Assume no page fault is generated by this instruction. Be very specific and indicate TLB hit/miss, cache hit/miss, and describe the actions taken. Label all stages clearly.

Grading: 1 point for TLB/pagetable access in first stage, 1 point for cache access in first stage, 1 point for second stage, 1 point for remaining 3 stages

fetch: TLB miss, check page table. Pagetable hits, bring entry back to TLB. Physical address is then sent to cache, which will result in a miss. Bring data from memory to cache, then return data to program. Increment PC.

decode: decode instruction, compute jump target address, update PC

ALU: idle

memory: idle

writeback: write PC+4 to £ra