CDA 3103 – Computer Organization and Logic

Final, Spring 2014

Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

PID:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Instructions

* Write your name and PID on each page
* Be lucid and neat
* Justify all your answers
* 75 minutes
* Closed book, Closed notes
* Non-graphing calculator permitted

NAME: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ PID:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Question 1 – 5 points

What characteristics of program memory accesses do caches exploit?

1. Spatial Locality
2. Temporal Locality
3. Continual Locality
4. Both a and b
5. Both b and c

Question 2 – 5 points

Which of the following is not a type of cache miss?

1. Conflict
2. Compulsory
3. Capacity
4. Contradictory

Question 3 – 5 points

Which of the following is not true about Virtual Memory?

1. Virtual addresses must be translated to physical addresses
2. Misses are called memory crashes
3. Pages have a fixed size
4. Virtual memory is main memory treated as a cache

Question 4 – 5 points

Which of the following may cause hazards to occur?

1. Competition for resources
2. Slower processors
3. Order of MIPS instructions
4. Adding no-ops
5. Both a and b
6. Both a and c
7. a, b, and c

Question 5 – 16 points

The following truth table corresponds to the addition of 2-bit binary numbers: A1A0 + B1B0 = C1C0.

1. Complete the truth table assuming that overflow results in DON’T CARE.
2. Give the most simplified expression for C1 and C0 in sum-of-product form using karnaugh maps.

C0 =

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **A1** | **A0** | **B1** | **B0** | **C1** | **C0** |
| **0** | **0** | **0** | **0** |  |  |
| **0** | **0** | **0** | **1** |  |  |
| **0** | **0** | **1** | **0** |  |  |
| **0** | **0** | **1** | **1** |  |  |
| **0** | **1** | **0** | **0** |  |  |
| **0** | **1** | **0** | **1** |  |  |
| **0** | **1** | **1** | **0** |  |  |
| **0** | **1** | **1** | **1** |  | C1 = |
| **1** | **0** | **0** | **0** |  |  |
| **1** | **0** | **0** | **1** |  |  |
| **1** | **0** | **1** | **0** |  |  |
| **1** | **0** | **1** | **1** |  |  |
| **1** | **1** | **0** | **0** |  |  |
| **1** | **1** | **0** | **1** |  |  |
| **1** | **1** | **1** | **0** |  |  |
| **1** | **1** | **1** | **1** |  |  |

NAME: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ PID:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Question 6 – 16 points

Write a sequence of MIPS instructions to implement the following segment of C code:

count = 0;

for(index=head; index<=n; index++)

if(C[index] == target)

count++;

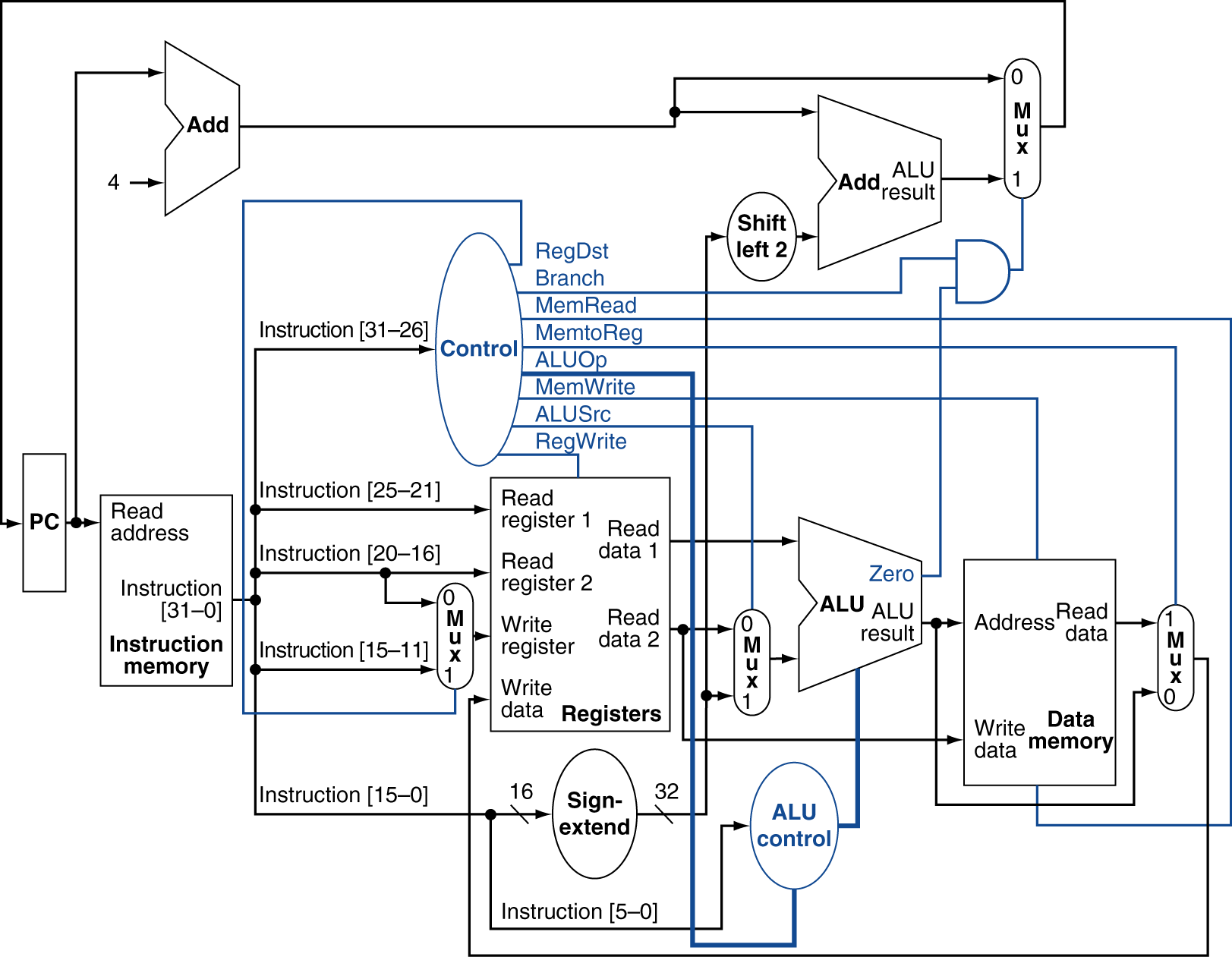
Assume the following register allocation:

$s0 count  
 $s1 index  
 $s2 head  
 $s3 base address of C array  
 $s4 target  
 $s5 n

Question 7 – 16 points

Different instructions utilize different hardware blocks in the basic single-cycle implementation. Determine the values of control signals (0 or 1) generated by the control for the following instruction and record them in the following table. For ALUOp, write in the function (ex: “and”) the ALU should perform.

sw $t0, 32($s3)



|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| RegDst | Branch | MemRead | MemtoReg | MemWrite | ALUOp | ALUSrc | RegWrite |
|  |  |  |  |  |  |  |  |

NAME: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ PID:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Question 8 – 16 points

Assume we have a small data cache capable of holding eight words. Each cache block consists of one word. When a given program is executed, the processor reads data from the following sequence of decimal addresses:

0, 4, 8, 11, 4, 5, 0, 4, 13, 17, 4, 15

Show the contents of the cache below at the end of the above reading operations if a four-way set-associative cache is used. For example, content at address 0can be shown as [0]. Also indicate the total number of cache hits and total number of block replacements. Assume LRU (Least Recently Used) replacement algorithm is used for block replacement in the cache, and the cache is initially empty.

|  |  |  |
| --- | --- | --- |
| Set # | Block Position | Contents of data cache |
| 0 | 0 |  |
| 1 |  |
| 2 |  |
| 3 |  |
| 1 | 0 |  |
| 1 |  |
| 2 |  |
| 3 |  |

Total number of cache hits: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Total number of block replacements: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Question 9 – 16 points

Suppose that you have a single-cycle processor which requires 10 clock cycles to perform a memory access, 4 cycles to perform a branch, and 2 cycles to perform all other instructions. Suppose further that the processor clock frequency is 1 GHz.

1. Suppose a program runs for 100 million instructions. The instructions consist of 30% memory accesses, 10% branches, and 60% other instructions. What is the average clock cycles per instruction? How much time will the program take to run?
2. A new design technique allows you to run the processor at 2000 MHz, but memory accesses now take 15 cycles. How much faster would a processor using the new design technique be?