**The University of Alabama in Huntsville**
**ECE Department**
**CPE 431 01**
**Test 2**
**November 9, 2017**

**Name: _____**

1. (1 point) The only dependence/hazard where full forwarding cannot save the day is between a _____ and a _____.
2. (1 point) The minimum unit of information that may be either present or not present in physical memory is a _____.
3. (1 point) _____ is an advanced pipelining technique that enables the processor to execute more than one instruction per clock cycle.
4. (10 points) It is possible to have an even greater cache hierarchy than two levels. Consider a processor with the following parameters.

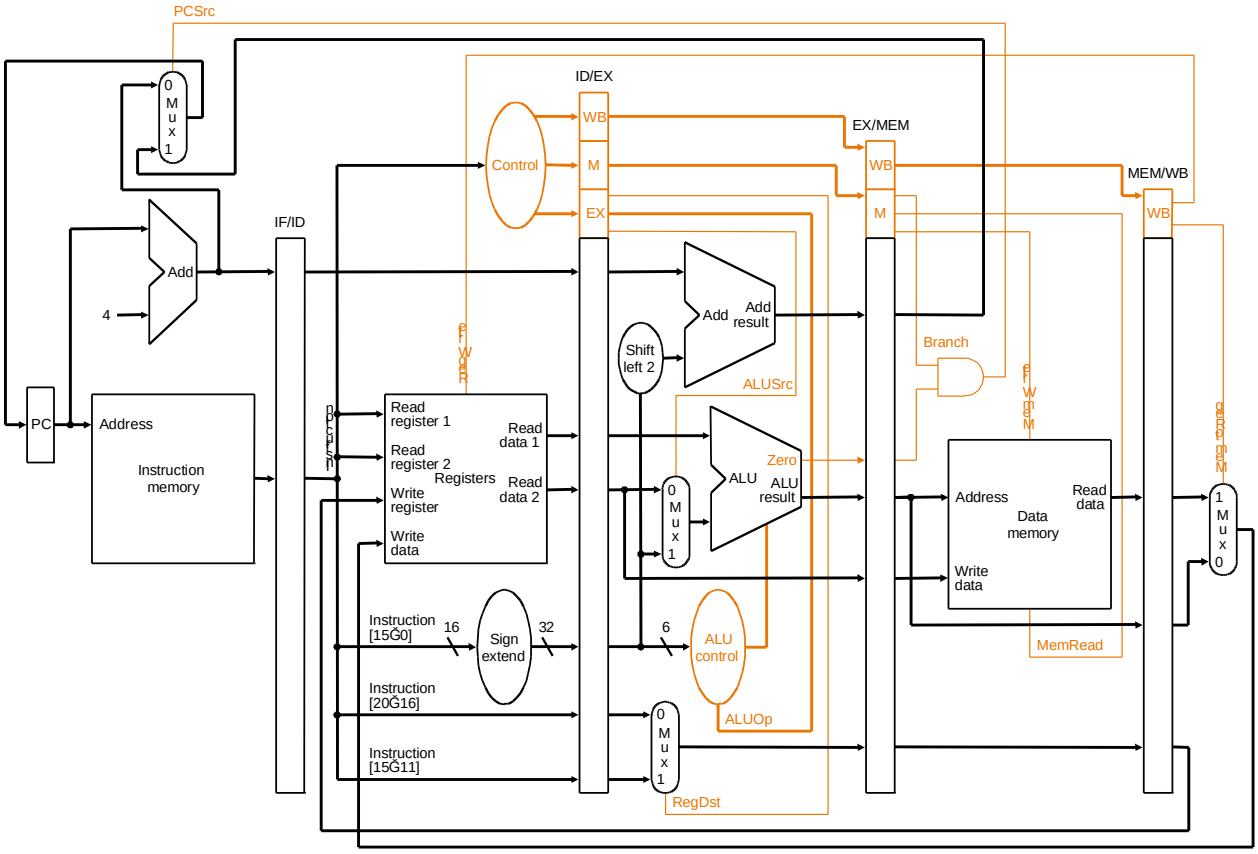
| Base CPI, no memory stalls | Processor speed | Main memory access time | First-level cache miss rate | Second-level cache, direct-mapped speed | Local miss rate with second-level cache, direct-mapped | Third-level cache, eight-way set associative speed | Local miss rate with third-level cache, eight-way set associative |
|---|---|---|---|---|---|---|---|
| 2 | 3.5 GHz | 90 ns | 5 % | 20 cycles | 40 % | 50 cycles | 30 % |

Calculate the CPI for the processor given that the Memory accesses/instruction = 1.36 and that hits in the L1 cache incur a 1 cycle stall.

5. (15 points) Here is a series of address references given as byte addresses: 118, 483, 2069, 321, 368, 1077, 1505, 812, 2832, 373, 1411, 511, 1463, 690, 4820, 1714, 1508, 1080. Assuming a two-way set associative-mapped cache with two-word blocks and a total size of 16 64-bit words that is initially empty and uses LRU, (a) label each reference in the list as a hit or a miss and (b) show the entire history of the cache, including tag and data.

| Byte Address (Decimal) | Byte Address (Hexadecimal) | Byte Address (Binary) |
|---|---|---|
| 118 | 76 | 0000 0000 0111 0110 |
| 483 | 1E3 | 0000 0001 1110 0011 |
| 2069 | 815 | 0000 1000 0001 0101 |
| 321 | 141 | 0000 0001 0100 0001 |
| 368 | 170 | 0000 0001 0111 0000 |
| 1505 | 5E1 | 0000 0101 1110 0001 |
| 812 | 32C | 0000 0011 0010 1100 |
| 2832 | B10 | 0000 1011 0001 0000 |
| 373 | 175 | 0000 0001 0111 0101 |
| 1411 | 583 | 0000 0101 1000 0011 |
| 511 | 1FF | 0000 0001 1111 1111 |
| 122 | 7A | 0000 0000 0111 1010 |
| 690 | 2B2 | 0000 0010 1011 0010 |
| 4820 | 12D4 | 0001 0010 1101 0100 |
| 1714 | 6B2 | 0000 0110 1011 0010 |
| 1508 | 5E4 | 0000 0101 1110 0100 |
| 2070 | 816 | 0000 1000 0001 0110 |
| 1080 | 438 | 0000 0100 0011 1000 |

6. (15 points) Consider executing the following code on a pipelined datapath like the one shown except that 1) it supports j instructions that complete in the ID stage, and 2) it has EX forwarding only. The register file does support writing in the first half cycle and reading in the second half cycle.




```
sort:    addi  $sp, $sp, -20
         sw    $ra, 16($sp)
         sw    $s3, 12($sp)
         sw    $s2, 8($sp)
         sw    $s1, 4($sp)
         sw    $s0, 0($sp)
         add   $s2, $a0, $zero
         add   $s3, $a1, $zero
         add   $s0, $zero, $zero
for1tst: slt   $t0, $s0, $s3
         beq   $t0, $zero, exit1
         addi  $s1, $s0, -1
for2tst: slt   $t0, $s1, $zero
         bne   $t0, $zero, exit2
         add   $t1, $s1, $s1
         add   $t1, $t1, $t1
         add   $t2, $s2, $t1
         lw    $t3, 0($t2)
         lw    $t4, 4($t2)
         slt   $t0, $t4, $t3
         beq   $t0, $zero, exit2
         add   $a0, $s2, $zero
         add   $a1, $s1, $zero
         jal   swap
         addi  $s1, $s1, -1
         j     for2tst
exit2:   addi  $s0, $s0, 1
         j     for1tst
exit1:   lw    $s0, 0($sp)
         lw    $s1, 4($sp)
         lw    $s2, 8($sp)
         lw    $s3, 12($sp)
         lw    $ra, 16($sp)
         addi  $sp, $sp, 20
         jr    $ra
```

If the `add $t2` instruction four instructions after the `for2tst` label begins executing in cycle 1 and the `beq $t0, $zero, exit2` is taken, what instructions are found in each of the five stages of the pipeline in the 11th cycle? Show the instructions being executed in each stage of the pipeline during each cycle. What value is stored in the ALUResult of the IF/ID pipeline register in the 11th cycle? Assume that before the instructions are executed, the state of the machine was as follows:
The PC has the value $200_{10}$, the address of the `add $t2` instruction
Every register has the initial value $20_{10}$ plus the register number.
Every memory word accessed as data has the initial value $10000_{10}$ plus the byte address of the word.

| Cycle | IF | ID | EX | MEM | WB |
|---|---|---|---|---|---|
| 1 | | | | | |
| 2 | | | | | |
| 3 | | | | | |
| 4 | | | | | |
| 5 | | | | | |
| 6 | | | | | |
| 7 | | | | | |
| 8 | | | | | |
| 9 | | | | | |
| 10 | | | | | |
| 11 | | | | | |

IF/ID.PCInc = _____

7. (1 point) _____ (True/False) Knowing the number of words in a block is enough to know how many sets there are in a cache.

8. (1 point) _____ (True/False) A 4-way set associative cache always has 4 sets.

9. (10 points) The importance of having a good branch predictor depends on how often conditional branches are executed. Together with branch predictor accuracy, this will determine how much time is spent stalling due to mispredicted branches. In this exercise, assume that the breakdown of dynamic instructions into various instruction categories is as follows:

| R-type | BEQ | JMP | LW | SW |
|---|---|---|---|---|
| 50% | 20% | 5% | 20% | 5% |

Also, assume the following branch predictor accuracies:

| Always-Taken | Always-Not-Taken | 2-Bit |
|---|---|---|
| 65% | 70% | 90% |

Stall cycles due to mispredicted branches increase the CPI. What is the extra CPI due to mispredicted branches with the (a) always-taken predictor, (b) always-not-taken predictor and (c) the 2-bit predictor? Assume that BEQ outcomes are determined in the EX stage, that JMP outcomes are determined in the ID stage, that there are no data hazards, and that no delay slots are used.

10. (15 points) Virtual memory uses a page table to track the mapping of virtual addresses to physical addresses. The following table is a stream of virtual addresses as seen on a system. Assume 16 KiB pages, byte addressing, a four-entry fully associative TLB, and true LRU replacement. If pages must be brought in from disk, increment the next largest page number. Given the address stream, and the shown initial state of the TLB and page table, show the final state of the system. Also list for each reference if it is a hit in the page table, or a page fault.

**TLB**

| Valid | Tag | Physical Page Number |
|---|---|---|
| 1 | 11 | 12 |
| 1 | 7 | 4 |
| 1 | 3 | 6 |
| 0 | 4 | 9 |

**Page table**

| VPN | Valid | Physical page or in disk |
|---|---|---|
| 0 | 1 | 5 |
| 1 | 0 | Disk |
| 2 | 0 | Disk |
| 3 | 1 | 6 |
| 4 | 1 | 9 |
| 5 | 1 | 11 |
| 6 | 0 | Disk |
| 7 | 1 | 4 |
| 8 | 0 | Disk |
| 9 | 0 | Disk |
| 10 | 1 | 3 |
| 11 | 1 | 12 |
| 12 | 0 | Disk |

| Address | TLB Hit/Miss | Page Table Hit/Miss | Page Fault Y/N |
|---|---|---|---|
| 184,760 | | | |
| 110,824 | | | |
| 56,312 | | | |
| 119,200 | | | |
| 136,100 | | | |
| 151,692 | | | |
| 126,856 | | | |
| 40,000 | | | |

11. (15 points) Using the code below, unroll the loop so that three iterations are executed. Arrange the unrolled code to maximize performance. You may assume that the loop executes a multiple of three times. Calculate the number of cycles for the original and for the unrolled, rearranged code. Assume full forwarding and one cycle delay for taken branches.

```
       addi  $t2, $zero, $zero      # i = 0
Loop:  sll   $t3, $t2, 2            # $t3 = i*4
       add   $t0, $t3, $s6          # $t0 = &A[i]
       add   $t1, $t3, $s8          # $t1 = &C[i]
       lw    $t0, 0($t0)            # $t0 = A[i]
       add   $t0, $t0, $s3          # $t0 = A[i]+j
       lw    $t0, 0($t0)            # $t0 = A[A[i]+j]
       sw    $t0, 0($t1)            # C[i] = A[A[i]+j]
       addi  $t2, t2, 1             # i++
       slt   $t4, $t2, 90           # $t4 = 1 if i < 90
       bne   $t4, $zero, Loop       # go back if i < 90
```

12. (15 points) In this exercise we look at memory locality properties of matrix computation. The following code is written in MATLAB, where elements within the same column are stored contiguously. Assume each word is a 32-bit integer.

```
for I = 1:8
  for J = 1:8000
    for K = 1:2000
      A[I][J][K] = B[J][1][I] + A[I][J][K];
```

(a) References to which variables exhibit temporal locality?
(b) References to which variables exhibit spatial locality?