The University of Alabama in Huntsville
ECE Department
CPE 431 01, CPE 531 01/01R
Fall 2022
Parallel Processors from Client to Cloud

Due November 29, 2022, 1.0(10), 2.0(10), 3.0(10)

1.0 Consider the following piece of C code:

```
for (j = 2; j < 1000; j++)
  D[j] = D[j-1] + D[j-2];
```

The MIPS code corresponding to the above fragment is:

```
       addiu   $s2, $zero, 7992
       addiu   $s1, $zero, 16
Loop:  l.d     $f0, -16($s1)
       l.d     $f2, -8($s1)
       add.d   $f4, $f0, $f2
       s.d     $f4, 0($s1)
       addiu   $s1, $s1, 8
       bne     $s1, $s2, loop
```

Instructions have the following associated latencies (in cycles):

add.d	l.d	s.d	addiu
3	4	2	1

1.0.1 How many cycles does it take for all instructions in a single iteration of the above loop to execute?

1.0.2 When an instruction in a later iteration of a loop depends upon a data value produced in an earlier iteration of the same loop, we say that there is a loop carried dependence between iterations of the loop. Identify the loop-carried dependences in the above code. Identify the dependent program variable and assembly-level registers. You can ignore the loop induction variable `j`.

2.0 Consider the following portions of two different programs running at the same time on four processors in a symmetric multi-core processor (SMP). Assume that before this code is run w, x, y, and z are -1, 4, 3 and 2 respectively and they are declared as int,

Core 1: x = z + w;
Core 2: y = x + z;
Core 3: w = x*y + 1;
Core 4: z = x/w;

What are all the possible resulting values of w, x, y, and z? For each possible outcome, explain how we might arrive at these values. You will need to examine all possible interleavings of instructions.

3.0 When performing computations on sparse matrices, latency in the memory hierarchy becomes much more of a factor. Sparse matrices lack the spatial locality in the data stream typically found in matrix operations. As a result, new matrix representations have been proposed.

One of the earliest sparse matrix representations is the Yale Sparse matrix Format. It stores an initial sparxe m x n matrix, M in row form using three one-dimensional arrays. Let R be the number of nonzero entries in M. We construct an array A of length R that contains all nonzero entries of M (in left-to-right top-to-bottom order). We also construct a second array IA of length m + 1 (i.e., one entry per row, plus one). IA(i) contains the index in A of the first nonzero of element of fow i. Row I of the original matrix extends from A(IA(i)) to A(IA(I+1)-1). The third array, JA, contains the column index of each element of A, so it also is of length R.

```
Row 0  [0, 0, -1, 0, 0, 4, 0, 0, 0, 7]
Row 1  [0, 2, 0, 3, 0, 0, 0, 0, 10, 0]
Row 2  [0, 0, 5, 8, 0, 0, 0, 0, 0, 9]
Row 3  [0, 0, 0, 0, 1, -2, 0, 0, 0, 0]
Row 4  [0, 6, 0, 0, 0, 0, 0, 0, 0, 0]
Row 5  [0, 0, 4, 0, 9, 0, 0, -2, 0, 0]
Row 6  [1, 2, 0, 0, 0, 0, 0, 0, -3, 0]
Row 7  [0, 11, 0, 0, 0, 0, 14, 0, 0, 0]
Row 8  [0, 6, 0, 11, -3, 0, 0, 0, 0, 0]
Row 9  [0, 0, 9, 0, 0, 0, 0, 7, 0, 0]
Row 10 [5, 0, 0, 0, 0, 0, 0, 0, -4, 0]
Row 11 [0, 1, 0, 0, 0, -1, 0, 0, 0, 0]
```

In terms of storage space, assuming that each element in matrix X is single precision floating point, compute the amount of storage used to store the Matrix above in Yale Sparse Matrix Format.