DCC888 – Instruction Level Parallelism [1]

Name: _____ ID: _____

1. Consider the program below, which contains some dependencies:

```
1) a = b
2) c = d
3) b = c
4) d = a
5) c = d
6) a = b
```

   For each pair of statements, classify the dependence as (i) true dependence; (ii) antidependence; (iii) output dependence or (iv) no dependence.

   (a) Statements 1 and 4

   (b) Statements 3 and 5

   (c) Statements 1 and 6

   (d) Statements 3 and 6

   (e) Statements 4 and 6

2. Generate three address code for the expression $((u - v) + (w - x)) \times (y + z)$ in such a way to produce a result exactly as parenthesized. Give register-level machine code to provide the maximum possible parallelism. You will need instructions to perform load, store, addition, subtraction and multiplication. If we assume that we can execute independent instructions in parallel, and assuming that each operation takes one cycle to execute, how many cycles do you need to solve this expression?

[1] These exercises have been taken from the Dragon Book, Chapter 10

3. Consider the program below, which computes the expression from the previous question:

```
LD r1, u
LD r2, v
SUB r1, r1, r2
LD r2, w
LD r3, x
SUB r2, r2, r3
ADD r1, r1, r2
LD r2, y
LD r3, z
ADD r2, r2, r3
MUL r1, r1, r2
```

Assuming that we have as many functional units as we need, how many steps do we need to evaluate this expression? Notice that in this case we are using a minimum number of registers. This economy puts pressure on parallelism, as it creates more dependences between instructions.

4. Draw the instruction dependence graph for each sequence of instructions below:

(a)
```
LD r1, a
LD r2, b
SUB r3, r1, r2
ADD r2, r1, r2
ST a, r3
ST b, r2
```

(b)
```
LD r1, a
LD r2, b
SUB r1, r1, r2
ADD r2, r1, r2
ST a, r1
ST b, r2
```

(c)
```
LD r1, a
LD r2, b
SUB r3, r1, r2
ADD r4, r1, r2
ST a, r3
ST b, r4
```

5. Assume a machine with one ALU resource (for the ADD and SUB operations), and one MEM unit (for the LD and ST operations). Assume that all operations require one clock, except for the LD, which requires two. However, a ST on the same memory location can commence one clock after a LD on that location starts. Find a shortest schedule for each of the fragments in the three code snippets of the previous question.

6. Consider the code fragment below:

```
if (x == 0) {
    a = b;
} else {
    a = c;
}
d = a;
```

Assume a machine in which loads take two cycles, and every other operation takes only one cycle. Also, assume a VLIW machine that can run any two operations simultaneously.

(a) Draw a control flow graph for the code snippet in this question.

(b) Find a shortest possible scheduling for the instructions that you have used in your CFG. Arrange instructions that are going to be issued together side-by-side.