CS4223 Tutorial 5: Data Level Parallelism

1. Consider the following vector code

```
LV          V1,Rx       ;load vector X
MULVS       V2,V1,F0    ;vector-scalar mult
LV          V3,Ry       ;load vector Y
ADDV.D      V4,V2,V3    ;add
SV          Ry,V4       ;store the result
```

Assume 4 cycle vector add latency, 6 cycle vector multiply latency, and 10 cycle vector load/store latency. Adder, multiplier, and load/store units are all pipelined. Also assume 8-lane vector processor, that is, the processor contains 8 units of pipelined adder, multiplier, and load/store unit. Further assume that there is no conflict among the memory bank accesses. How long will it take to execute this code if vector length = 32, VLRMAX = 32 and there is no chaining?

2. Consider the following code fragment that produces 128 results.

```
  for (i=0; i<64; i++) {
       a[i] = u * b[i];
        c[i] =  b[i] + v;
}
```

Assume a vector pipeline with the following pipeline latencies and one lane per operation. Also assume there are no bank conflicts in the accesses for the above loop.

Operation	Start-up penalty
Vector load/store	12 cycles
Vector multiply	7 cycles
Vector add	6 cycles

(a) Write the vector version of this code in assembly language.

(b) Assuming no chaining and single memory pipeline, how many clock cycles will be required to execute this code fragment including start-up overheads.

(c) If the vector sequence is chained but still has single memory pipeline, how many clock cycles will be required to execute this code fragment including start-up overheads?

(d) Assuming three memory pipelines and chaining, how many clock cycles are required to execute this code fragment including start-up overheads?

2. Consider the following code, which multiplies two vectors that contain single-precision complex values:

```
 for (i=0; i< 300; i++){
          c_re[i] = a_re[i] * b_re[i] – a_im[i] * b_im[i];
          c_im[i] =a_re[i] * b_im[i] + a_im[i] * b_re[i];

}
```

Assume that the processor runs at 700 MHz and has a maximum vector length of 64. The load/store unit has a startup overhead of 15 cycles; the multiply unit, 8 cycles; the add/subtract unit, 5 cycles.

a) What is the arithmetic intensity (number of operations per byte transferred) of this kernel?

b) Convert this loop into VMIPS assembly code using strip mining.

c) Assuming chaining and a single memory pipeline, how many clock cycles are required per complex result value, including start-up overhead?

d) Now assume that the processor has three memory pipelines and chaining. If there are no bank conflicts in the loop's accesses, how many clock cycles are required per result?

3. The following kernel performs a portion of the finite-difference time-domain (FDTD) method for computing Maxwell's equations in a three-dimensional space, part of the SPEC06fp benchmarks:

```
  for (int x =0; x < NX-1; x ++) {

          for (int y =0; y < NY-1; y++) {

               for (int z=0; z < NZ-1; z++) {

                    int index = x*NY*NZ + y*NZ + z;

                     if (y > 0 && x > 0) {

                              material = IDx[index];

                              dH1 = (Hz[index] – Hz[index-incrementY]/dy[y];
                    dH2 = (Hy[index] – Hy[index-incrementZ]/dz[z];

                    Ex[index] = Ca[material] * Ex[index] + Cb[material] * (dH2 – dH1);

}}}}
```

Assume that dH1, dH2, Hy, Hz, dy, dz, Ca, Cb, Ex are all single-precision floating-point arrays. Assume IDx is an array of unsigned int.

a) What is the arithmetic intensity of this kernel?

b) Assume this kernel is to be executed on a processor that has 30 GB/sec of memory bandwidth. Will this kernel be memory bound or compute bound?

c) Develop a roofline model for this processor, assuming it has a peak computational throughput of 85 GFLOP/sec. [We will discussion roofline model in class]