EEL 6764 Principles of Computer Architecture
Practice #2

(**1**) Consider the following code. Assume maximum vector length of 64.

```
for (i=0;i<300;i++) {
   A[i] = A[i] + B[i];
}
```

(a) Rewrite the code using strip mining.

```
for (i=0; i<44; i++) {
   A[i] = A[i] + B[i];
}
k = 44;
for (h=0; h<4; h++) {
   for (i=k; i < k+64; i++)
         A[i] = A[i] + B[i];
   k = k + 64;
}
```

Or follow the example shown in the book for a similar solution.

(b) Assume that ADDER is pipelined with 6 stages. Also assume that memory contains a single bank. It takes 5 cycles to load the first data element from memory, and 1 cycle for each following data element. How many cycles are required to finish the above code? Ignore write time.

$\mathbf{5 + 7 + 299 = 311}$

(c) Now assume there is an additional ADDER same as the above. Now find the total cycles to finish the above code. Also consider the case where there are 4 ADDERs.

With additional ADDER but without increasing memory bandwidth, the latency for the above code is the same as the previous question at the best.

(d) In the previous questin, do more ADDERs help reduce runtime? explain your answer. If yes, explain if the total runtime can be reduced further if more ADDERs are added? If not, propose a solution.

for two or more ADDERs added into the processor, memory bandwidth also needs to be increased. This can be done by adding more independent memory banks. For two ADDERs, suppose the memory bandwidth is doubled. Then, for each ADDER, which handles half of elements in the vector registers, it takes $5 + 7 + 149 = 161$ **cycles to finish the above code.**

(**2**) Consider the following code. Assume maximum vector length of 64, ADDER is pipelined with 6 stages, and the multiplier is pipelined with 8 stages.

```
for (i=0;i<300;i++) {
   A[i] = A[i] + B[i];
   C[i] = 2 * A[i];
}
```

(a) Assume that memory has sufficient bandwidth to supply data to keep FUs busy. Find the total number of cycles to finish the above code without chaining.

it takes $7 + 299 = 306$ **cycles to finish the first vector operation,** $9 + 299 = 308$ **to finish the second. Therefore, it takes** 614 **cycles to finish the above code.**

(b) Find the total number of cycles to finish the above code with chaining.

It takes $6+8+1$ for the element of index 0 to finish, while it takes 1 cycle for each of remaining 299 elements, therefore it takes $15+299=314$ cycles to finish the above code.

(**3**) Explain how the conditional execution of the following code is supported for a vector architecture? Be specific.

```
for (i = 0; i < 64; i=i+1)
   if (X[i] != 0)
      X[i] = X[i] - Y[i];
```

For a more general case shown below, how would it be processed by a vector machine? Propose necessary architecture features to support the following code.

```
for (i = 0; i < 64; i=i+1)
   if (X[i] > Y[i])
      X[i] = X[i] - Y[i];
   else
      X[i] = Y[i] - X[i];
```

In general, you can use additional MUXs to control which operations are applied to which elements, or, which elements are used to update vector registers.

(**4**) Memory addressing with strides is necessary to access non-continuous memory locations. Explain what problems it may cause? By specific.

Refer to the example in the slides for this questions.