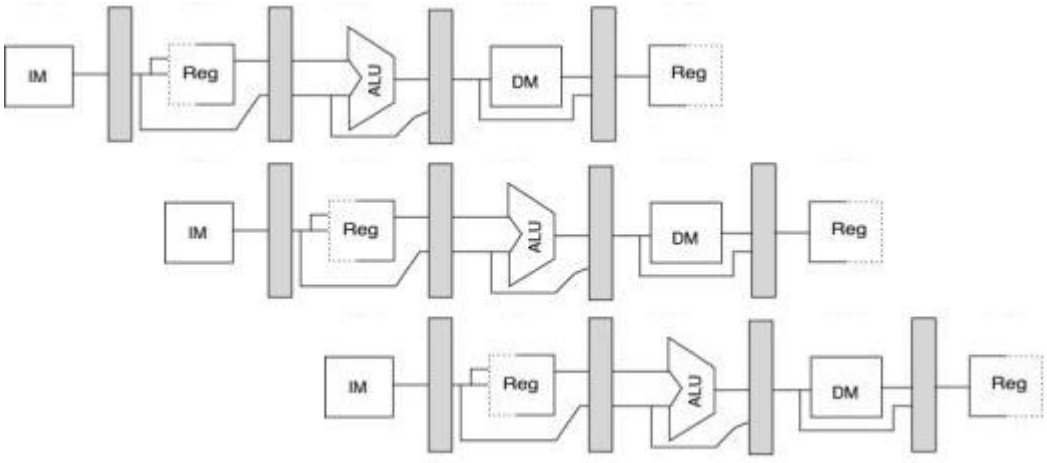
CS430 Problem Set #7

Date assigned: Monday, April 27, 2015
Date due: Tuesday, May 5, 2015 by 3pm
Points: 50

1) (15 pts) Consider the following MIPS program:

```
dadd r1,r2,r3
ld r4,0(r1)
sd r4,12(r1)
halt
```

a) Identify all of the data dependencies in the above program drawing arrows as we've done in class.
b) Given the 5 stage MIPS pipeline, show the instructions as they move through the pipeline using an S for stall when necessary. Assume no forwarding.
c) Rework b) assuming the maximum amount of forwarding possible.
d) Complete the following diagram showing where each forwarded piece of data comes from.



2) (5 pts) Consider the following MIPS program:

```
ld r1,0(r2)
dsub r4,r1,r5
and r6,r1,r7
or r8,r1,r9
halt
```

I make the claim that using the maximum amount of forwarding, no stalls are necessary. Either accept or reject my claim. Which ever stance you take, justtify your answer as you did in 1) d) above.

3) (12 pts) To compute the inner product of two vectors, we might write the loop:

```
for (i = 0; i < n; ++i)
{
  sum = sum + a[i] * b[i]
}
```

The code generated for this loop by an optimizing compiler might be as follows:

```
loop:

     ld r4,0(r9)          ; r9 has address a[i]
     daddi r9,r9,8        ; make r9 point to a[i+1], 8 bytes away
     ld r5,0(r19)         ; r19 has address of b[i]
     daddi r19,r19,8      ; make r19 point to b[i+1], 8 bytes away
     dmul r4,r4,r5        ; a[i]*b[i]
     dadd r8,r8,r4        ; sum = sum + a[i] * b[i]
     daddi r6,r6,1        ; i = i + 1
     slt r1,r6,r7         ; compare i < n
     bnez r1, loop        ; branch if i < n
```

The code before the loop would have to set up the following values in registers:

```
r9   address of a[1]
r19  address of b[1]
r8   initial value of sum
r6   initial value of i
r7   value of n
```

We can reduce the number of instructions by using a load-with-update instruction as on the PowerPC. An example of the proposed instruction would be: ld.u r4,8(r9) which would do the following:

1. compute the EA of r9+4
2. store the computed address back into register r9
3. fetch the word at that location storing it into register 4

a) Rewrite the code for the above loop using the load-with-update removing the instructions that update the address registers. Note any changes in the initial values of registers that need to be made.

b) Suppose you have the choice of having no load-with-update instruction or making all load instructions load-with-update for a new processor with a new instruction set. How would you make this decision?

c) If the vector a is defined as a: .word 1, 2, 3 How would you set r9 to point to the address of the first element of a?

4) (18 pts) You are to write an assembly language program for the winMIPS64 machine that will merge two ordered arrays into a third ordered array. The arrays are ordered in increasing order and the size of the array is in the first word of each array. The max size for each array is 100 elements (including the number of elements in each array). If the size of the merged array exceeds 99, terminate the program after merging 99 values. The array names are **arrayOne**, **arrayTwo**, and **arrayMerged**. Each statement of your program must be properly documented as shown in question 3).

Note1: Please make sure your problem sets are typed, answered in order, and stapled together.

Note2: A hard copy of your Problem Set Solution is due by the date and time specified. Create a folder 07PUNetID and place a copy of your solution 07PUNetID.doc (or .pdf) and a fully documented copy of the assembly language program from 4) 07PUNetID.s into the folder. Then place the folder 07PUNetID in the CS430 Drop Box.

There is no late exception for this last assignment!!!!!