CA226 – WinMIPS64 Floating point exercise I

Assume we have a double *x* and a positive integer *n*. We want to raise *x* to the *n*-th power. Here's one (pretty dumb) way of doing it.

```
        .data
    n:  .word 8
    x:  .double 0.5

        .text
        LD      R1,n(R0)
        L.D     F0,x(R0)
        DADDI   R2, R0, 1     ; R2 = 1
        MTC1    R2,F11        ; F11 = 1
        CVT.L.D F2,F11        ; F2 = 1
loop:   MUL.D   F2, F2, F0    ; F2 = F2*F0
        DADDI   R1, R1, -1    ; decrement R1 by 1
        BNEZ    R1, loop      ; if R1 != 0 continue

; result in F2

        HALT
```

Get this program working with WinMIPS64

Note how the MTC1 instruction moves an integer number into a floating-point register. The instruction CVT.L.D convert the integer to Double Precision format.

In the pipeline window observe in particular the execution of the MUL.D instruction.

Next implement this simple algorithm to calculate $w=x^n$. Try and minimize the number of clock cycles.

```
        w=1;
        forever
        {
            if (n%2!=0) w*=x;
            n/=2; if (n==0) break;
            x*=x;
        }
```

Convert this to MIPS64 assembly language.