Name: **Id#**

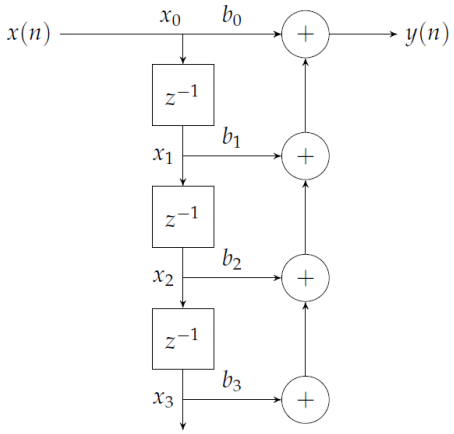
COE 405, Term 181

Design & Modeling of Digital Systems

Assignment# 5

Due date: Tuesday, Nov. 27, 2018

It is required to design a parametrizable m-tap finite impulse response filter (FIR). The FIR filter receives an n-bit input and produces an n-bit output. It has m n-bit tap coefficients. An example of a 4-tap FIR filter is shown below:



The 4-tap FIR filter performs the following function: Y(t) = X(t)*b0 + X(t-1)*b1 + (Xt-2)*b2 + X(t-3)*b3 (i.e., the current output Y(t) is equal to the current input X(t) times b0 plus the previous input X(t-1) times b1 plus …etc.). The m tap coefficients should be read from a file.

Q.1. First, we will consider an implementation of the FIR filter with a throughput of one value per clock cycle i.e., the FIR filter will receive an input value every clock cycle and produces an output value. In this case, we will use as many multipliers and adders as needed.

(i) Write a parametrizable Verilog module for modeling the m-tap FIR filter. It receives both m and n as parameters. The model has an n-bit input X and produces m*n-bit output Y. The FIR has an asynchronous input that resets the previously stored values to 0.

(ii) Write a test bench that tests a 4-tap FIR filter with 4-bit inputs using the following tap coefficients b[0]=1, b[1]=1, b[2]=2 and b[3]=2. Apply the following inputs {1, 2, 3, 4} by applying X=1 first.

(iii) Implement the FIR filter on FPGA using the parameter and test values in (ii) and include a video link demonstrating its correct functionality.

Q.2. Second, we will consider an implementation of the FIR filter using only a single adder and a single multiplier. To do that, we will use a circular buffer. The circular buffer is a data structure that allows efficient stream processing. The buffer definition and its implementation in C are given below. The buffer is initialized by having all of its values reset to 0. We can insert a value in the buffer using Put, or get a value from the buffer using Get providing the value I of the input to be obtained with i=0 indicating the last value inserted in the buffer.

```
#define SIZE 3
int buffer[SIZE];
int pos;

void init() {
  for (int i = 0; i < SIZE; i++)
     buffer[i] = 0;
  pos = SIZE - 1;
}

void put(int value) {
  pos = (pos + 1) % SIZE;
  buffer[pos] = value;
}

/* get the ith value earlier from the circular buffer; zero being the newest value */
int get(int i) {
int index = (pos - i);
If (index >=0)
   return buffer[index];
else return buffer[SIZE+index];
}
```

(i) Write a parametrizable Verilog module for modeling a circular buffer of size m with each buffer slot of size n. The buffer receives both m and n as parameters. The buffer receives three inputs, Init, Put, Get. When Init is asserted, the Buffer initializes all of its content to 0. When Put is asserted, the buffer inserts an n-bit input value X. When Get is asserted, the buffer will receive an input I of size log2(m) bits to indicate the position of needed value and will return an n-bit output value Y. Assume that the three

inputs will not be asserted simultaneously. Each of those actions will be done in a single clock cycle.

(ii) Write a test bench to test the correct functionality of your circular buffer using a buffer size of 4 and 4-bit input values. Apply the following input sequence in consecutive cycles: Init, Put(1), Put(2), Put(3), Put(4), Get(0), Get(1), Put(5), Get(0), Get1).

(iii) The FIR filter can be implemented in C language using the circular buffer as shown below:

```
int fir(int x) {
  int i, y;
    y = x*b[0];
    for (i = 0; i < SIZE; i++)
    y += b[i+1] * get(i);
    put(x);
return y;
}
```

In this implementation whenever a new input X is applied a Start signal will be asserted in the same cycle. Once the corresponding output is computed a Done signal will be asserted.

a) Show the design of the data path for implementing the FIR filter.
b) Show the ASM design for controlling the operation of the FIR filter.
c) Write a Verilog module for modeling your data path and ASM of the FIR filter.
d) Write a test bench a test bench that tests a 4-tap FIR filter with 4-bit inputs using the following tap coefficients b[0]=1, b[1]=1, b[2]=2 and b[3]=2. Apply the following inputs {1, 2, 3, 4} by applying X=1 first.
e) Implement the FIR filter on FPGA using the parameter and test values in (d) and include a video link demonstrating its correct functionality.