EE529 **EMBEDDED SYSTEMS**

Assignment-2

1. Write a VHDL program to model a two input AND gate and a two input XOR gate using dataflow style of modeling. Save the program files in your working library.

2. Use the models of the AND gate and XOR gate created in problem 1 to develop the structural model of a half adder.

3. Write a VHDL program to model a half adder using dataflow style of modeling.

4. Write a VHDL program to model a half adder using behavioral style of modeling.

5. Use the model of the half adder developed in problem 2 to develop the model of a single bit full adder.

6. Use the model of the single bit full adder developed in problem 5 to develop the model of a four bit ripple carry adder.

7. Consider a circuit to control the motor of a tape player. The circuit has four inputs PL, RE, ST and M and two outputs P and R. Four of the inputs are the control buttons on the tape player. The input PL is 1 if the play button is pressed, the input RE is 1 if the rewind button is pressed, and the input ST is 1 if the stop button is pressed. The fourth input to the control circuit is M, which is 1 if the special music sensor detects music in the current tape position. The two outputs P and R when 1 make the tape operate in play and rewind modes respectively. No more than one output should be ever on at a time and all the outputs zero cause the tape motor to stop. The buttons control the tape player as follows. If the play button is pressed, the tape player starts to play. If the play button is pressed and the rewind button is also pressed, then the music sensor detects whether music is there in the current tape position and if music is there then the motor rewinds until the music sensor detects no music. Pressing the stop button at any instant of time will make the tape player motor to come to a halt. Write a behavioral VHDL model to represent the circuitry of the controller.

8. A complex MAC unit operates on two sequences of complex numbers, $\{x_i\}$ and $\{y_i\}$. The MAC unit multiplies corresponding elements of the sequences and accumulates the sum of products. The result is $\sum_{i=1}^{N} x_i y_i$, where N is the length of the sequences. Each complex number is represented in Cartesian form, consisting of a real and an imaginary part. If we are given two complex numbers x and y, their product is a complex number p, calculated as follows:

$$p_{real} = x_{real} \times y_{real} - x_{imag} \times y_{imag}$$

$$p_{imag} = x_{real} \times y_{imag} + x_{imag} \times y_{real}$$

The sum is a complex number s calculated as follows:

$$s_{real} = x_{real} + y_{real}$$
$$s_{imag} = x_{imag} + y_{imag}$$

The MAC unit calculates its result by taking successive pairs of complex numbers, one each from two input sequences, forming their complex product and adding an accumulator register. The accumulator is initially cleared to zero and is reset after each pair of sequences has been processed. If we count the operations required for each pair of input numbers, we see that the MAC must perform four multiplications to form partial products, then a subtraction and an addition to form the full product and finally two additions to accumulate. This is diagrammatically represented in figure 1. Since the operation must be performed in this order, therefore the time taken to complete processing one pair of inputs is the sum of the delays of the three steps. In a high performance digital signal processing application, this delay may cause the bandwidth of the system to be reduced below a required minimum. We can avoid the delay by pipelining the MAC unit that is, organizing it like an assembly line as shown in figure 2. Assume a delay of 2 ns for the registers, 8 ns for the multiplier and 3 ns for the adder. Model the MAC unit structurally and behaviorally using VHDL assuming both the un-pipelined and pipelined architectures as shown in figures 1 and 2. Enumerate through your simulation any advantage that you are obtaining out of the pipelined architecture. Compare your simulation results with any performance gain that you might be obtaining through mathematical calculations. Use 16 bit two's complement fixed point binary representation. Bit 15 is assumed to be the sign bit and binary point is assumed to be between bits 15 and 14. Using this format we can represent numbers in the range of -1 (inclusive) to +1 with a resolution of 2^{-15}. Use overflow bit(s) if needed in your circuit model.

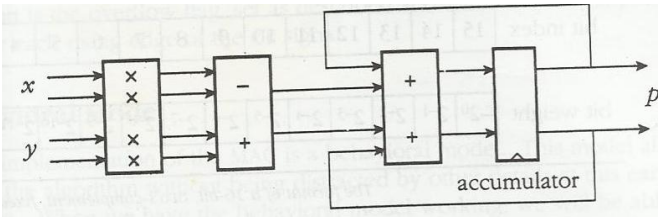

Figure 1

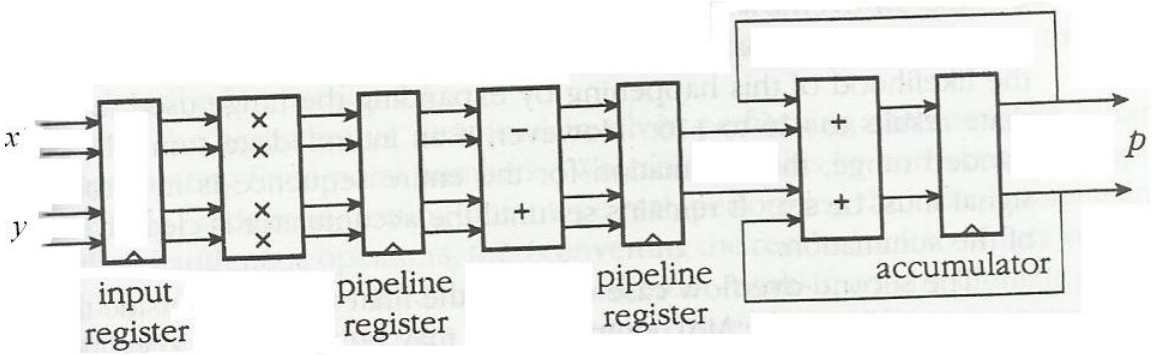

Figure 2

9. Consider the same problem in problem no. 8. Devise a sequence of inputs values for the MAC that can cause the sum in the accumulator to overflow.

10. The following process stamen models a producer of data:
    producer: **process** is
      variable next_data: natural:=0;

```
begin
   data<=next_data;
   next_data:=next_data+1;
   data_ready<='1';
  wait until data_ack='1';
data read<='0';
wait until data_ack='0';
end process producer;
```

The process uses a four phase handshaking protocol to synchronize data transfer with a consumer process. Develop a process statement to model the consumer. It too should use delta delays in the handshaking protocol. Simulate the handshaking process.