ESKISEHIR TECHNICAL UNIVERSITY

DEPARTMENT OF ELECTRICAL AND ELECTRONICS ENGINEERING

LAB-3

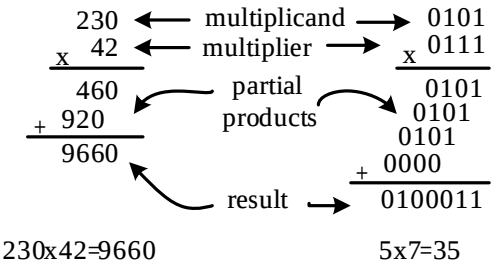
SEQUENTIAL MULTIPLER

1 Introduction

In this third lab, you will implement an 8-bit sequential multiplier. You have to use the shift and add algorithm whose details will be described in next chapters.

2 Binary multiplication

Binary multiplication is similar to decimal multiplication but there is only 1's and 0's.



While the multiplication is doing, partial products are produced by multiplying a single digit of the multiplier with the entire multiplicand. The shifted partial products are added to get the result. In general, a NxN multiplier multiplies two N-bit numbers and gives a 2N-bit result. The partial products in binary multiplication are either the multiplicand or all 0's since there is only 1's and 0's.

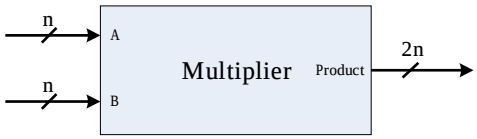


Figure 1: Multiplier block diagram

3 Sequential Multiplier

A sequential multiplier needs some extra signals for synchronization purpose. You should define these signals in entity section of your design.

Input **clk**: clock signal to synchronize the system.
Input **reset**: asynchronous reset signal to initialize the system.
Input **start**: synchronous signal that must be high to start a new operation.
Output **ready**: synchronous signal that is set during 1 cycle by the multiplier when the result of the operation is available.

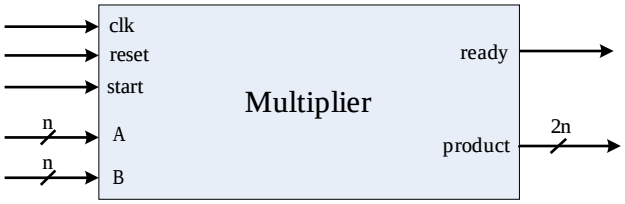


Figure 2: Multiplier port declaration.

Shift and add algorithm is as follows; you sequentially multiply the multiplicand (**B**) by each digit of the multiplier (**A**) and then summing up the partial products which are properly shifted to get the final result.
There is an architecture view of the multiplier that you will design.

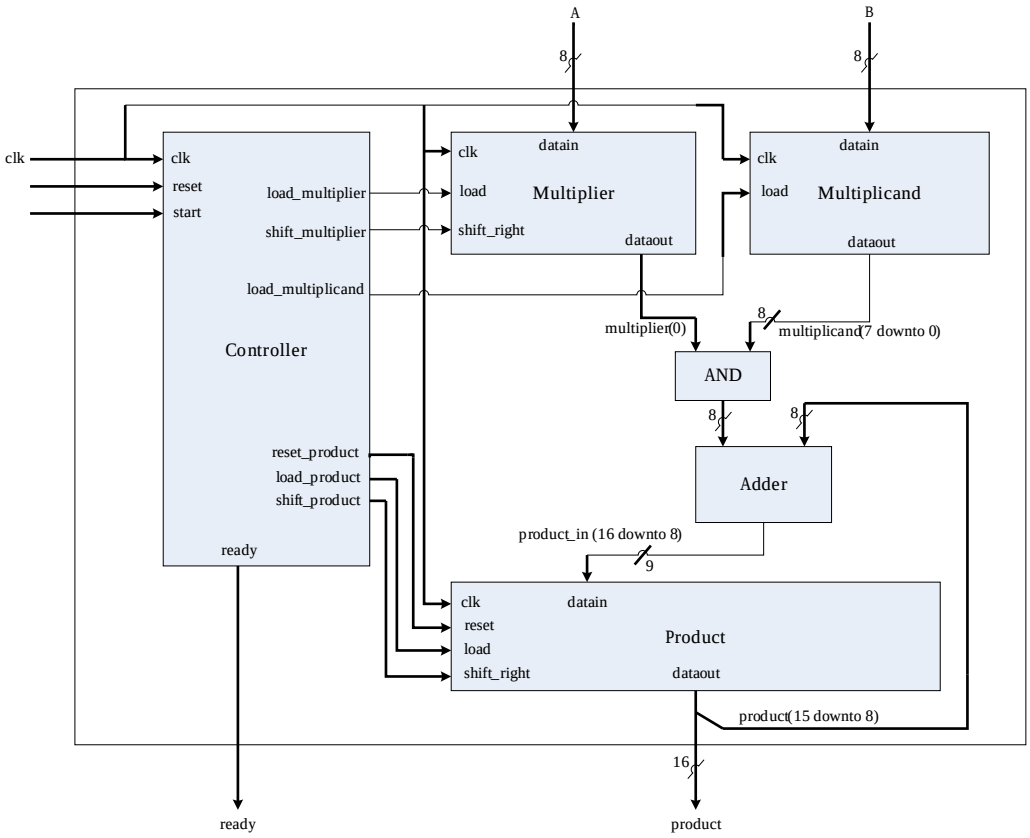


Figure 3: 8-bit Sequential multiplier components

The expressions of the components that you have to implement are listed below.

8-bit register **Multiplicand**:

- Operand B is loaded into internal register inside the multiplicand component when load signal is high.

8-bit shift register **Multiplier**:

- Operand A is loaded into internal register inside the multiplier component when load signal is high.
- Register which keeps the value of A is shifted to the right when shift_right signal is high.
- The dataout output signal of this component is 1-bit signal. It is the least significant bit of the register.

17-bit shift register **Product**:

- Register is initialized to 0 when reset signal is high.
- Register loads the addition result from adder in its most significant bits when load signal is high.
- Register is shifted to the right when shift input is high.
- The dataout output signal bits are its 16 least significant bits.

AND component:

- The block diagram consists of 8 AND gates logically.
- It performs the AND logical operation on each multiplicand bits with the least significant bit of the multiplier.

8-bit **Adder**:

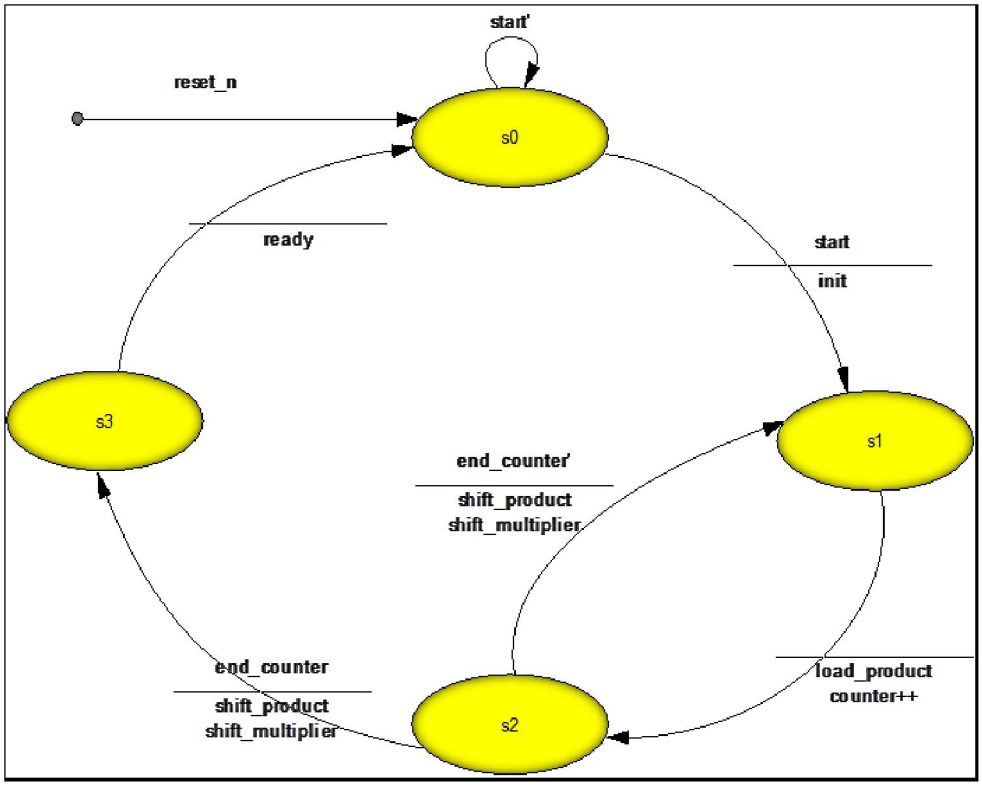
- The component sums the result of the AND gates with the 8 most significant bits of the Product output.

The system **Controller**:

- It includes a state machine and it produces control signals of all other components at right time.

4 State Machine

The controller contains the state machine which is given below.



The asynchronous reset signal is initialized to state machine to s0.

State s0: Waits in this state till start signal is high before going into state s1 and initializing the register in which:

- The multiplicand and multiplier registers load the input values.
- The product register and the loop counter are initialized to 0.

State s1: The next state is always s2.

- In this state, Adder result is loaded into the product register.
- The loop counter is always incremented by 1.

State s2: If the loop counter reaches the end value, continue with state s3, otherwise return to the state s1.

- In both cases, the multiplier and product registers are shifted. Therefore, necessary control signals must be produced in this state.

State s3: The next state is always s0.

- In this state, multiplication operation is complete. **ready** output signal must be set active to indicate operation is done.

5 Procedure

In this third lab, you will design a sequential system to perform multiplication operation on two 8-bit operands.

- Write the VHDL description of the multiplication unit whose details are given above.
- You should simulate your design using ISE or another VHDL simulator to prove the correctness of your design.
- There is no hardware implementation for this lab, therefore only your simulation results are graded.