NCTU-EE IC LAB – Spring 2018

Lab01 Exercise

Design: Booth Multiplier

Data Preparation

1. Extract test data from TA's directory:
   % tar xvf ~iclabta01/Lab01.tar
2. The extracted LAB directory contains:
   a. Practice/ : example code
   b. Exercise/ : your design

Design Description and Examples

Booth algorithm is a multiplication operation that multiplies two numbers in two's complement notation. The detail algorithm is described as below:

1. Find the second and the third biggest of the five inputs and execute with Booth algorithm.

2. Assume that the multiplicand m is x-bit and multiplier r is y-bit. Initialize a register P for the final result, and the length is x+y+1 bits. The initial value of P is 0(x bits)_r(y bits)_0(1 bit).

3. The rightmost 2 bits used for the selection of different executions.

LSB	Execution
00	No execution
01	Add m to the left part of P
10	Subtract m from the left part of P
11	No execution

3. Arithmetically shift 1 bit on P

4. Repeat the step (2) and (3) for y times.

5. The final answer is obtained by dropping the LSB from P.

The summary of the description and specifications are as followings:

Input Signal	Bit Width	Description
in_1	6	signed input
in_2	6	signed input
in_3	6	signed input

in_4	6	signed input
in_5	6	signed input
Output Signal	**Bit Width**	**Description**
out	12	Product

Examples:

Assume in_1 = 0000 , in_2 = 0010 , in_3 = 0111 , in_4 = 0110 , in_5 =0001

The third biggest input is 0010,and the second biggest input is 0110

0010*0110

Iteration	**Multiplicand**	**Execution**	**P**
0	**0010**	**initial**	**0000_0110_0**
1	**0010**	**LSB:00 no execution**	**0000_01100**
	0010	Arithmetically shift 1 bit on P	**0000_00110**
2	**0010**	**LSB:10 Subtract m from the left part of P.**	**1110_00110**
	0010	Arithmetically shift 1 bit on P	**1111_00011**
3	**0010**	**LSB:11 no execution**	**1111_00011**
	0010	Arithmetically shift 1 bit on P	**1111_10001**
4	**0010**	**LSB:01 Add m to the left part of P.**	**0001_10001**
	0010	Arithmetically shift 1 bit on P	**0000_11000**

Out = 0000_1100

Inputs

1. The signal **in_1, in_2, in_3, in_4,in_5** are signed 6-bit inputs.

Outputs

The signal **out** is signed 12-bit. This represents the product of two input.

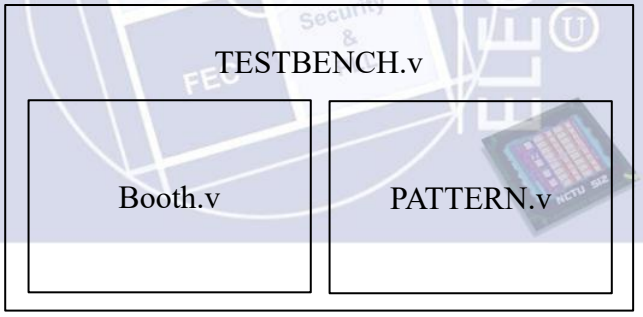
Specifications

1. You can ONLY use Booth algorithm to complete the multiplication .
2. Top module name : Booth (File name: Booth.v)
3. Input pins : in_1[5:0], in_2[5:0], in_3[5:0], in_4[5:0], in_5[5:0],
4. Output pins : out[11:0]
5. The maximal delay is **15ns**. You can choose the delay you want, but it must be less than 15ns. The delay of your design will influence your grade.
   Note: Change the parameter **Max_Delay** in 02_SYN/syn.tcl if you want to compress your cycle time.

```
#------------------------------------------------------
#  Global Parameters
#======================================================
set DESIGN "Booth"
set MAX_Delay 10
```

6. After synthesis, check the "Booth.area" and "Booth.timing" in the folder "Report". The area report is valid only when the slack in the end of "Booth.timing" is non-negative.
7. The synthesis result **cannot** contain any **latch**.
   Note: You can check if there is a latch by searching the keyword "**Latch**" in 02_SYN/**syn.log**

Block Diagram



Grading Policy

The performance is determined by the area and delay of your design. The less cost your design has, the higher grade you get.

Function Validity: 70%

Performance: 30% (Total simulation time: 15%, area: 15%)

Note

1. Please upload the following file on e3 platform before **12:00 at noon** on **Sep. 25**: **Booth_iclab??.v** and **latency_iclab??.txt** (latency is the smallest latency you use

in synthesis, ?? is your account no.)

Ex: Booth_iclab99.v, 5.5_iclab99.txt

2. Template folders and reference commands:

In RTL simulation, the name of template folder and reference commands is:

01_RTL:

"**./01_run**"

02_SYN/ (Synthesis):

./01_run_dc

(Check **latch** by searching the keyword "**Latch**" in 02_SYN/**syn.log**)

(Check the design's timing in /Report/Booth.timing)

03_GATE_SIM/ (GL simulation):

./01_run

You can key in **./09_clean_up** to clear all log files and dump files in each folder

Example Waveform

Input and output signal: