EE332 Lab2: Simulation of Full Adder on Nexys 4 DDR

1st Qiu Kunyuan
EEE. Southern University of Science and Technology
Shenzhen, PRC
11913019@mail.sustech.edu.cn

***Abstract*—This report focuses on the phenomenons encountered in the simulation of a simple full adder, to reveal some critical features of FPGA programming comparing to ordinary programmable devices. Among these phenomenons, jitters and the race-hazard conditions are of the most concerns.**

***Index Terms*—FPGA, programmable logic, full adder, race-hazard condition, jitter, delay**

I. Introduction

Half adder and full adder are basic elements in digital circuits used to perform addition operations on binary numbers.

A half adder

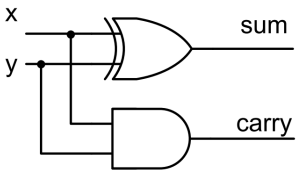

Fig. 1: Gate Level Description of Half Adder

is a combinational logic circuit that takes two binary inputs and produces two binary outputs: sum and carry.

$$\mathrm{HA} : (x_1, x_2) \mapsto (Q, C) := \begin{cases} Q & = \mathrm{xor}(x_1, x_2) \\ C & = \mathrm{and}(x_1, x_2) \end{cases} \tag{1}$$

The main limitation of the half adder is that it cannot handle carry inputs and therefore can only be used for 1-bit addition.

A full adder

is an extension of the half adder that accepts three binary inputs: two additions and a rounding input, and produces two binary outputs: sum and carry. Full adders can be connected in series to achieve binary addition of any number of bits.

Implementation of full adder can be derived directly from 3-digit addition, where the final carry output is toggled if any of $X_1 + X_2$ or $C_i + Q_1$ produces a carry:

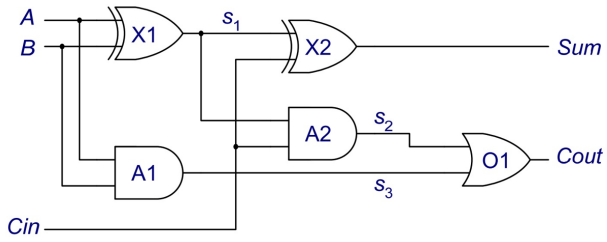

Fig. 2: Gate Level Description of Full Adder

$$(q_1, c_1) = \mathrm{HA}(x_1, x_2) \tag{2}$$
$$(q_2, c_2) = \mathrm{HA}(q_1, c_i) \tag{3}$$
$$c_o = \mathrm{or}(c_1, c_2) \tag{4}$$

Therefore the gate-level circuit of the full adder can be easily captured, as shown of Figure. (I)

II. Verilog Modeling

There are two levels of HDL modeling for full adder, one for gate level description and one for behavioral level description.

A. Gate-Level Implementation

For the gate level modeling, the DNF (OR-AND) of the output ports are required:

$$\begin{aligned} Q_2 &= HA(HA(x_1, x_2)[0], c_i)[0] \\ &= \mathrm{xor}(\mathrm{xor}(x_1, x_2), c_i) \\ &= \overline{A}BC + \overline{A}B\overline{C} + A\overline{B}\overline{C} + ABC \\ C_{\text{out}} &= AB + BC + AC \end{aligned} \tag{5}$$

Therefore, the Verilog code is pretty straight forward:

```
1 `timescale 1ns / 1ps
2
3
4
5 module full_adder_g (
6     input  wire c0,
7     input  wire a,
8     input  wire b,
```

```
9     output wire s,
10    output wire c1
11 );
12
13   assign s  = ((~c0) & (~a) & b) | ((~c0
       ) & a & (~b)) | (c0 & (~a) & (~b)) |
        (c0 & a & b);
14   assign c1 = (c0 & a) | (a & b) | (c0 &
       b);
15
16 endmodule
```

Listing 1: Gate Level Modeling of Full Adder in Verilog