Lab assignment 1 / Week 1

Preparation for the lab

You should have issued Icoboard and SD card from Lab 419. The SD card will serve as a storage for operating system you are going to run on Raspberry Pi. The Raspberry Pi will be used to program IcoBoard which contains an FPGA from iCE40 series from Lattice Semiconductors. Raspberry Pi's are available in Lab A719.

To start the system, you will put the SD card in Raspberry Pi's SD card slot and attach icoboard to Pi (it will be shown in Lab or class, if it is not obvious). Making sure that the monitor, keyboard and mouse are connected to Raspberry Pi, you power it up.

You can use a text editor on Raspberry Pi to enter your verilog programs. The compile it using the tool-chain described below, then send the binary generated to the icoboard. Finally, icoboard is initiated to execute according to the binary loaded on it. All these steps can be done by executing commands on Raspberry Pi.

Please download http://intranet.iith.ac.in/files/misc/2017-03-02-raspbian-jessie-icotools.img (~8GB) and copy it onto the SD card. Use this link to do that: https://www.raspberrypi.org/documentation/installation/installing-images/linux.md. The SD card then can be put in raspberry pi. The image provides the tools needed to compile Verilog and send that output to Icoboard.

1. **CSwap and its *simulation***: Go through its description on https://en.wikipedia.org/wiki/Fredkin_gate. Like a NAND gate, this gate can be used to create all logic. It is also reversible, therefore the inputs can be created from the outputs, which is not true for the NAND.
The implementation of cswap gate will be discussed in class.
We will cover
A. How to write cswap gate in structural or behavioral styles.
B. How to assemble multiple cswap gate to make one bit full adder.
C. How to use this full adder to make arbitrary width full adder.
D. How to write testbenches in for all of the above cases.

You will be downloading the verilog examples using a tool called git. We will take a very brief look at how it works.
To use this tool on raspberry pi, type following on the text terminal:

```
t
```

This will produce a folder named intro-hdl in your current folder. A folder named "cswap" resides inside "intro-hdl". The files for this lab session are inside "cswap". You will be compiling the verilog files and running simulation of those circuits using iverilog. Take a look at README file contained in that folder.

You can access future lab session by updating the folder, when I make updates to the repository above or by downloading it afresh. You can save your changes too, for that you have to create an account with github.com or bitbucket.org (which allows for private repositories) and save your work in a repository there.

[In “cswap” folder you can also find fa.json, which is a format that is used by a program called wavedrom (https://wavedrom.com/) to create pretty looking timing diagrams. Download and install if you find need for it.]

Your task for this exercise is to compile all the verilog code and look at simulation results in gtkwave to see if they make sense. Try to modify code somewhere and see what happens.

2. **Toffoli gates**: In this part, you will be writing implementation other gates using the Toffoli gate (https://en.wikipedia.org/wiki/Toffoli_gate).
A. First write Toffoli gates structural and behavioural code in Verilog.
B. Write AND, OR, NOT and Fredkin gates by using Toffoli gates.
C. Write a 2-to-1 multiplexer using Toffoli gates.
D. Construct 2^n-to-1 using multiplexers made in C.

You should have written testbenches for A, B, C, D.