**Quartus Prime Introduction Using Verilog Designs**

**Andy Cha**

**1. Introduction:**

This tutorial provided by Intel introduces the user to Quartus Prime CAD system. This tutorial will provide a general overview of a typical CAD flow for designing circuits implemented by using FPGA devices. This tutorial makes use of the Verilog design entry method.Then involves configuring the designed circuit onto a FPGA device. The device being used in this lab will be Terasic’s DE10-Lite.

**2. Background:**

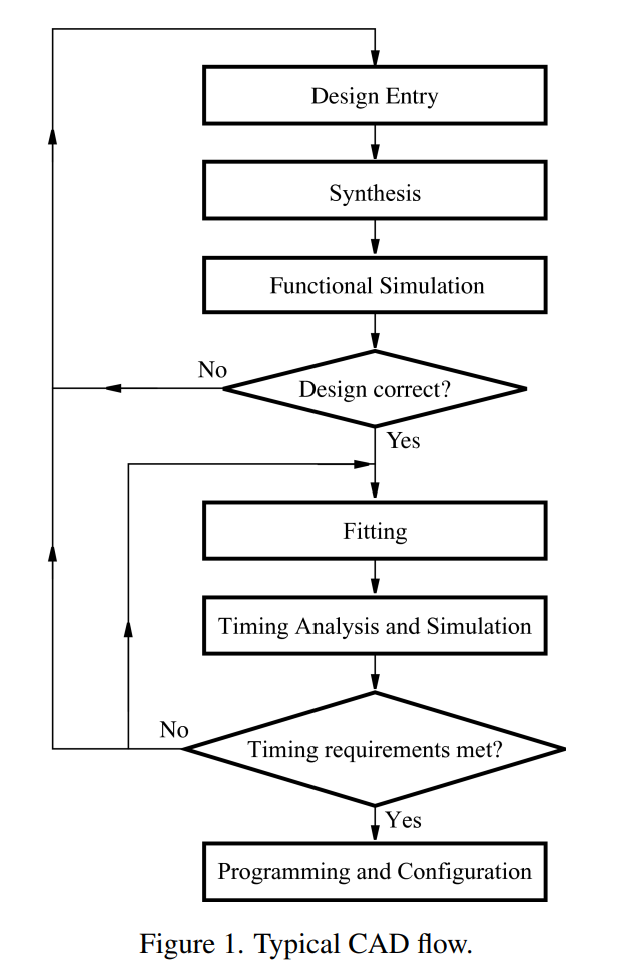


Figure 1 displays a typical CAD (Computer Aided Design) flow. CAD software makes it easy to implement a desired logic circuit by using a programmable logic device.

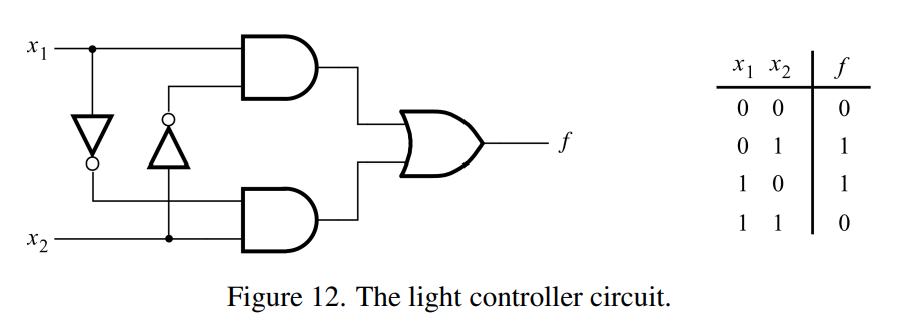
**3. Getting Started:**

The first step of this tutorial walks us through creating a directory where we will place all our files. The directory name will be called “introtutorial”. Then we create our project that will be named “light” and pick the FPGA device associated with DE0-Lite: *MAX 10 10M50DAF484C7G*. Once we have set up our project we create a file and save it as “light.v” (the same as our project name).

**4. Design Entry Using Verilog Code:**

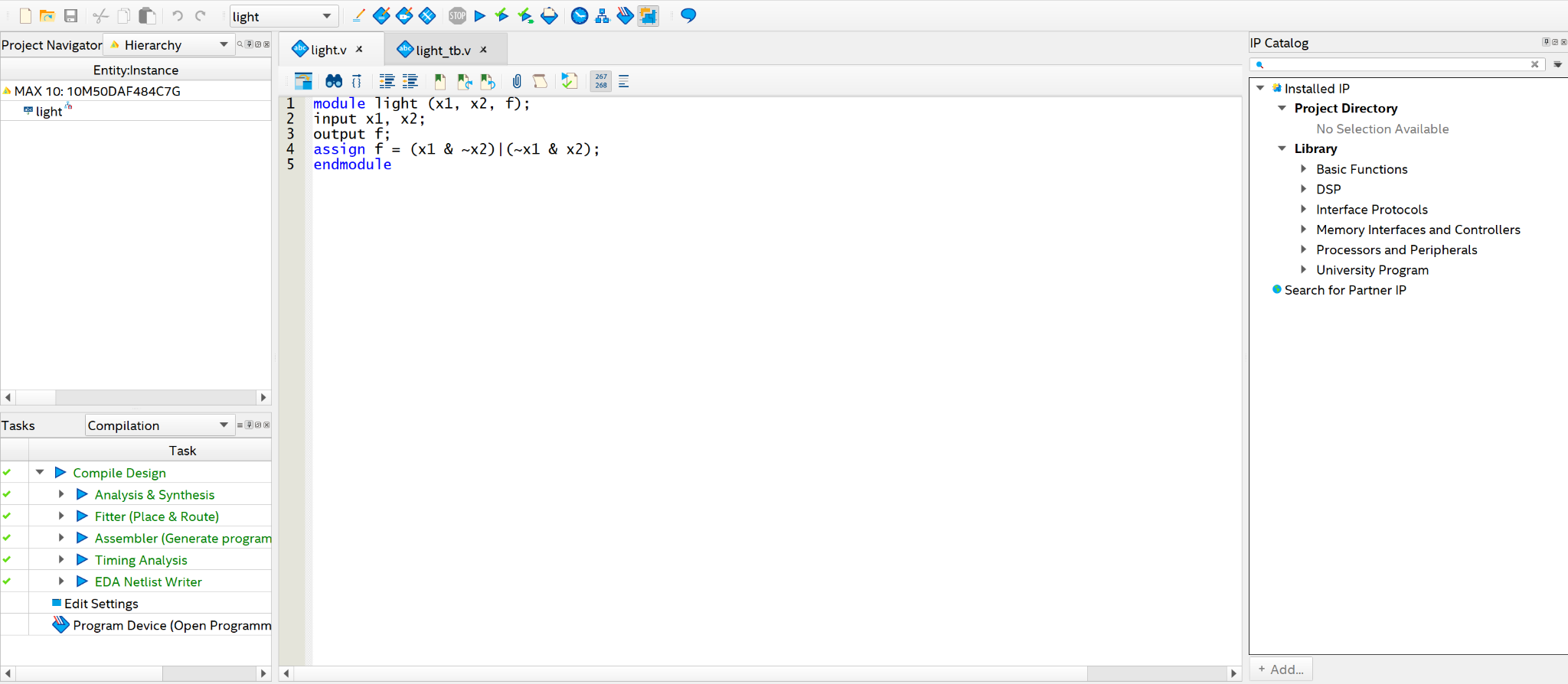
We will design a two-way light controller circuit as shown in the figure below.

A single light can be controlled by either of the two switches *x1* and *x2*. The truth table is provided as well to indicate input and output relations.



**Design file:**

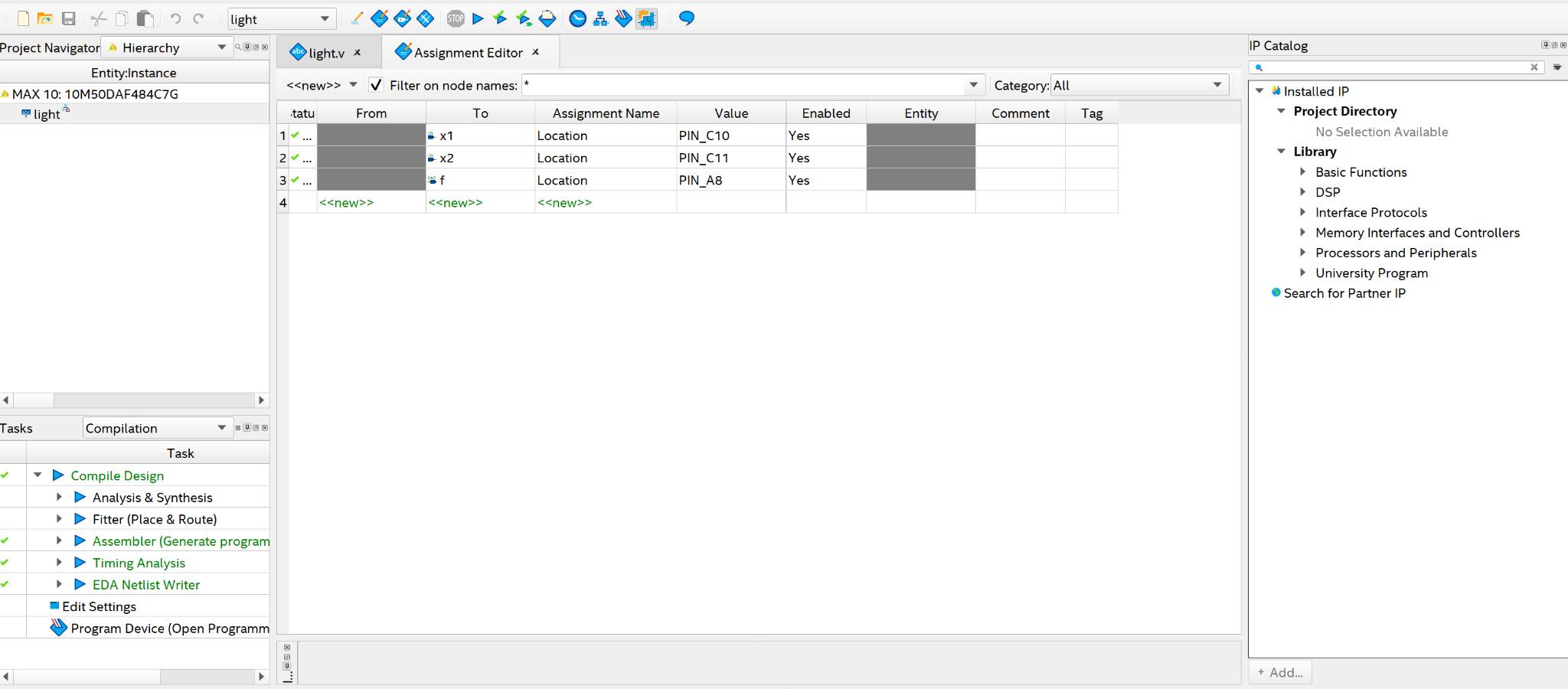
Here is the Verilog code for the circuit “light.v”. We can save and compile to check whether there are any errors present.



**5. Pin Assignment:**

Now that we have compiled the Verilog code and make sure that no errors are present, we can move on to pin assignment. Using the assignment editor, We will assign two toggle switches on the board as *x1* and *x2* inputs. Then 1 red LED as the output *f*.

**Assignment Editor:**

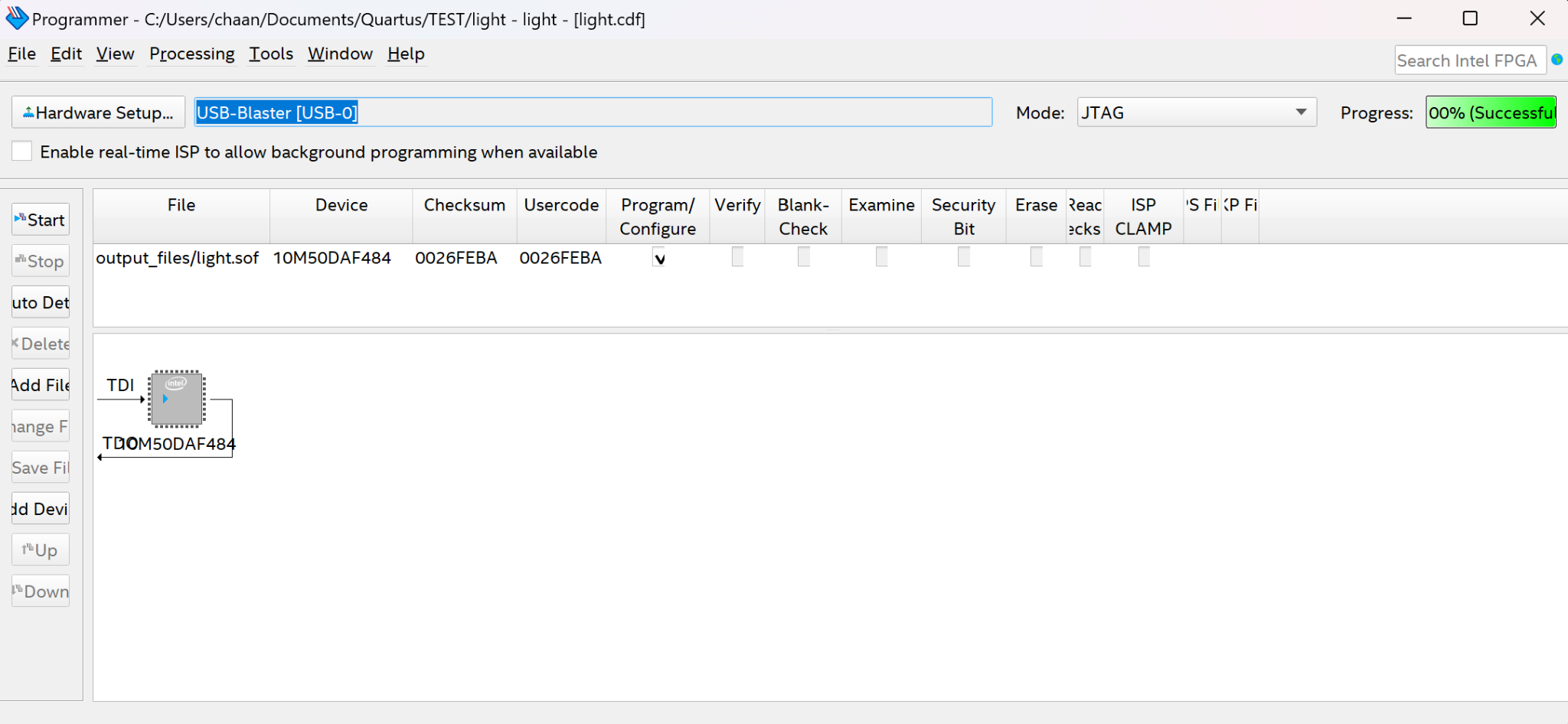


**6. JTAG Programming:**

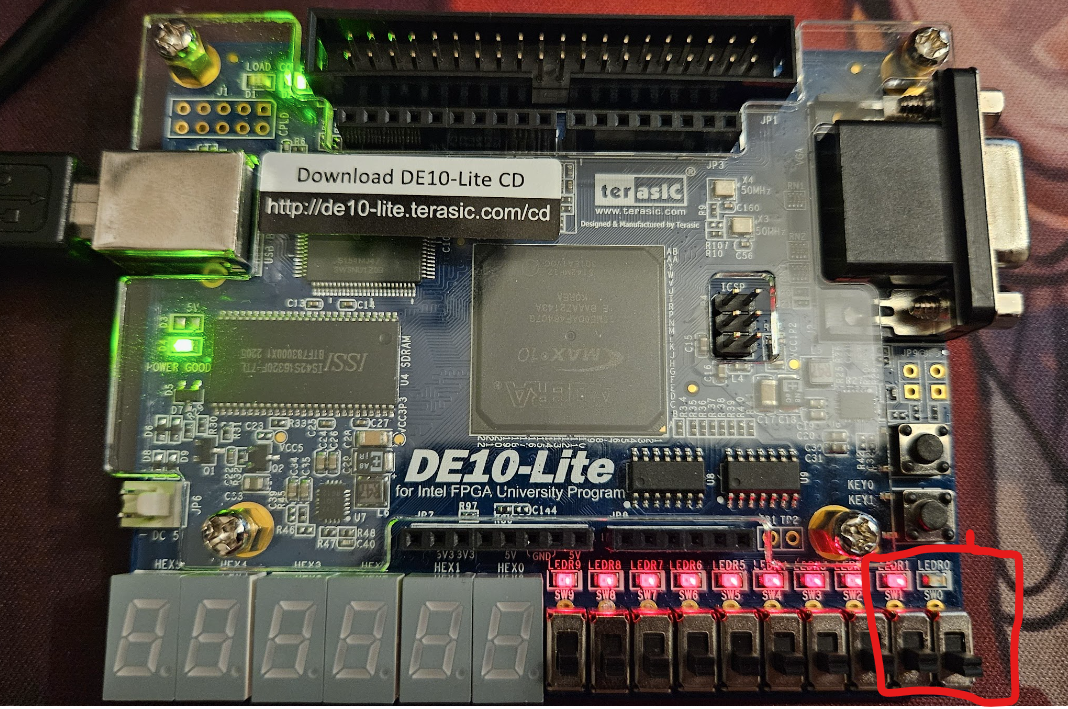
In the JTAG mode, the configuration data is loaded directly into the FPGA device. JTAG or Joint Test Action Group, is a simple way for testing digital circuits and loading data into them, which became an IEEE\* standard. If the FPGA is configured in this manner, it will retain its configuration as long as the power remains turned on. The configuration information is lost when the power is turned off.

The Active Serial (AS) mode is a configuration device that includes some flash memory that is used to store the configuration data. Quartus Prime software places the configuration data into the configuration device on the DE-series board. Then, this data is loaded into the FPGA upon power-up or reconfiguration. Thus, the FPGA need not be configured by the Quartus Prime software if the power is turned off and on. The choice between configuring your board with one of these two modes is made by switches.

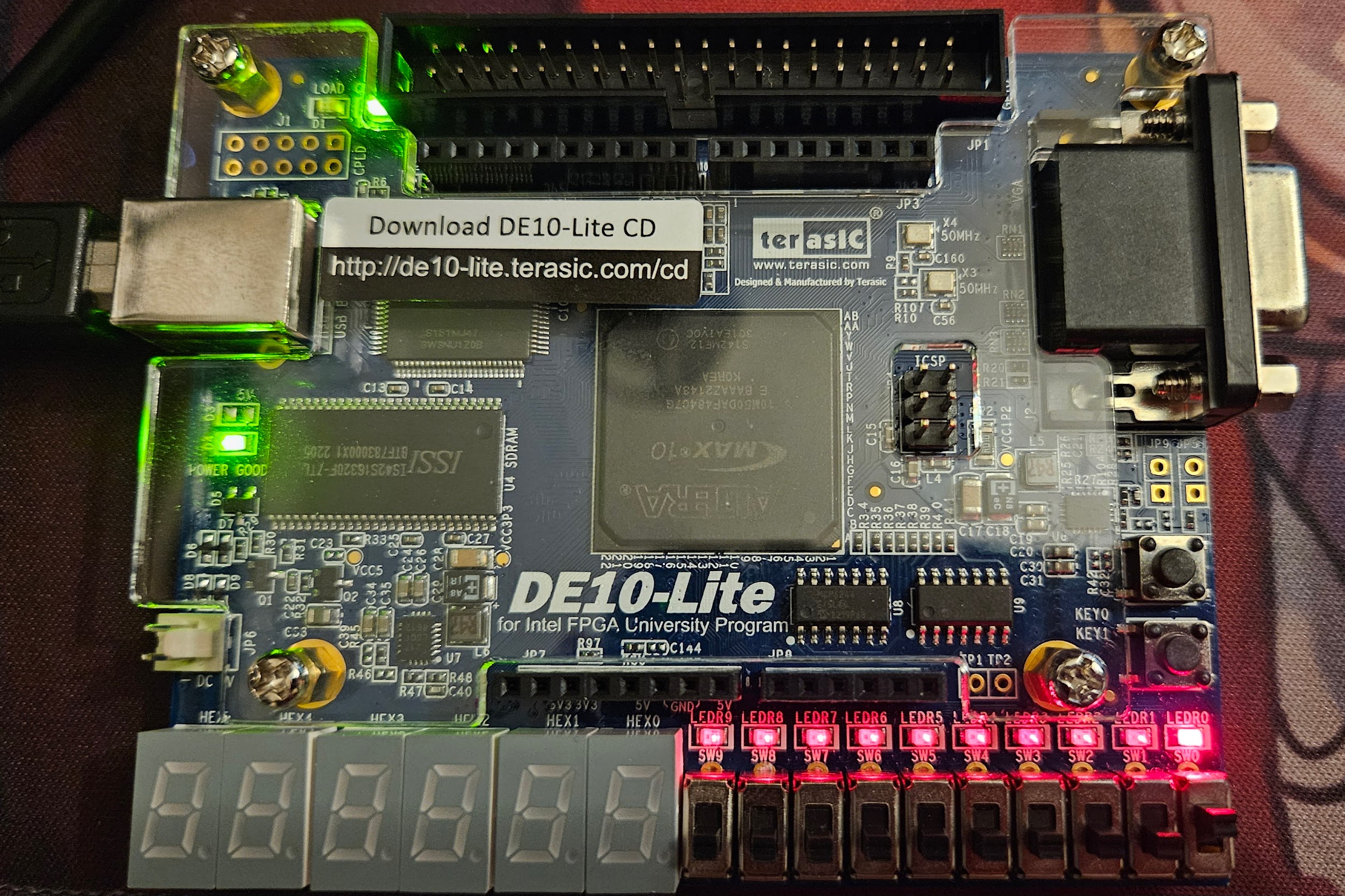
**Programmer (hardware setup):**

****

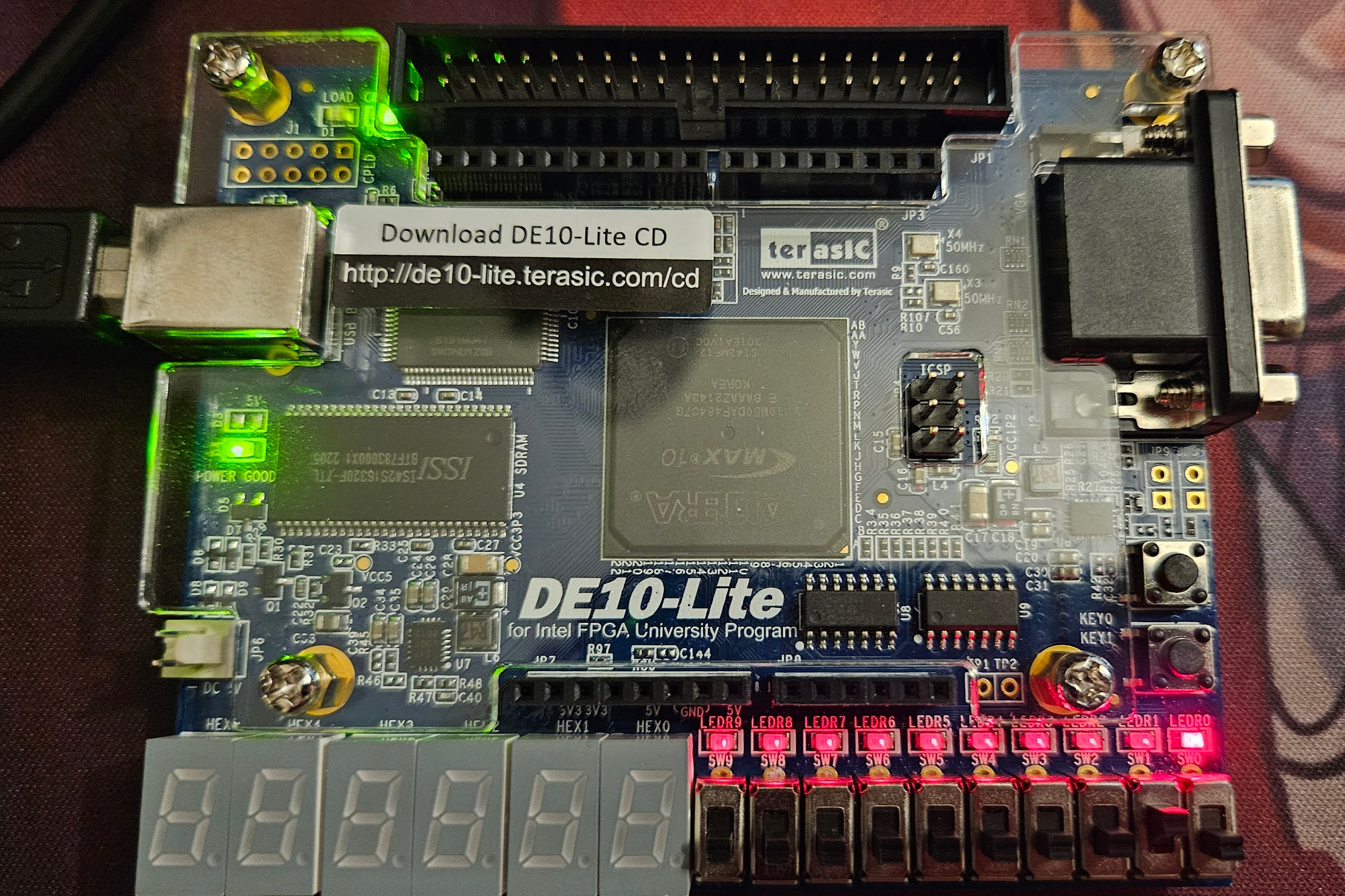
After pin assignment, To load the design into the FPGA we go to Tools > Programmer. There we can verify that the hardware is identified through the USB and the output file is listed. Note that we can also identify the mode as JTAG or AS here. Then we can hit “start” to load the design. Once the progress displays (100% Successful) on the top right, then the design is loaded completely to the FPGA.

****

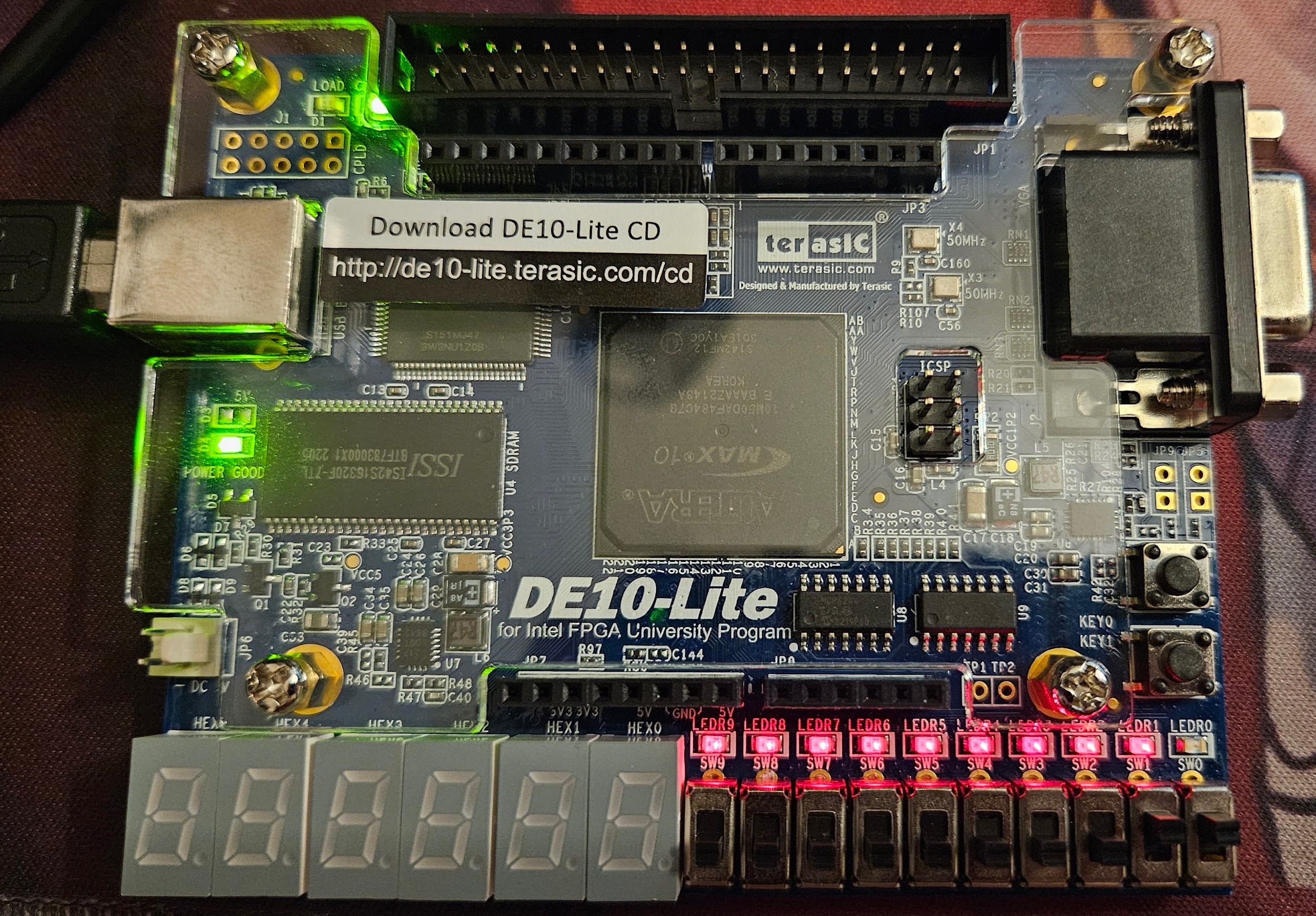
After the design is loaded to the FPGA device we can notice the changes to the device. *SW0* = x1, *SW1* = x2, and *LEDR0* = f. When *SW1* and *SW0* are 0, then *LEDR0* is OFF.



When *SW0* is ON or 1, then *LEDR0* turns on.



If we turn *SW0* to OFF again and switch *SW1* to ON, then we can see that *LEDR0* is still on.



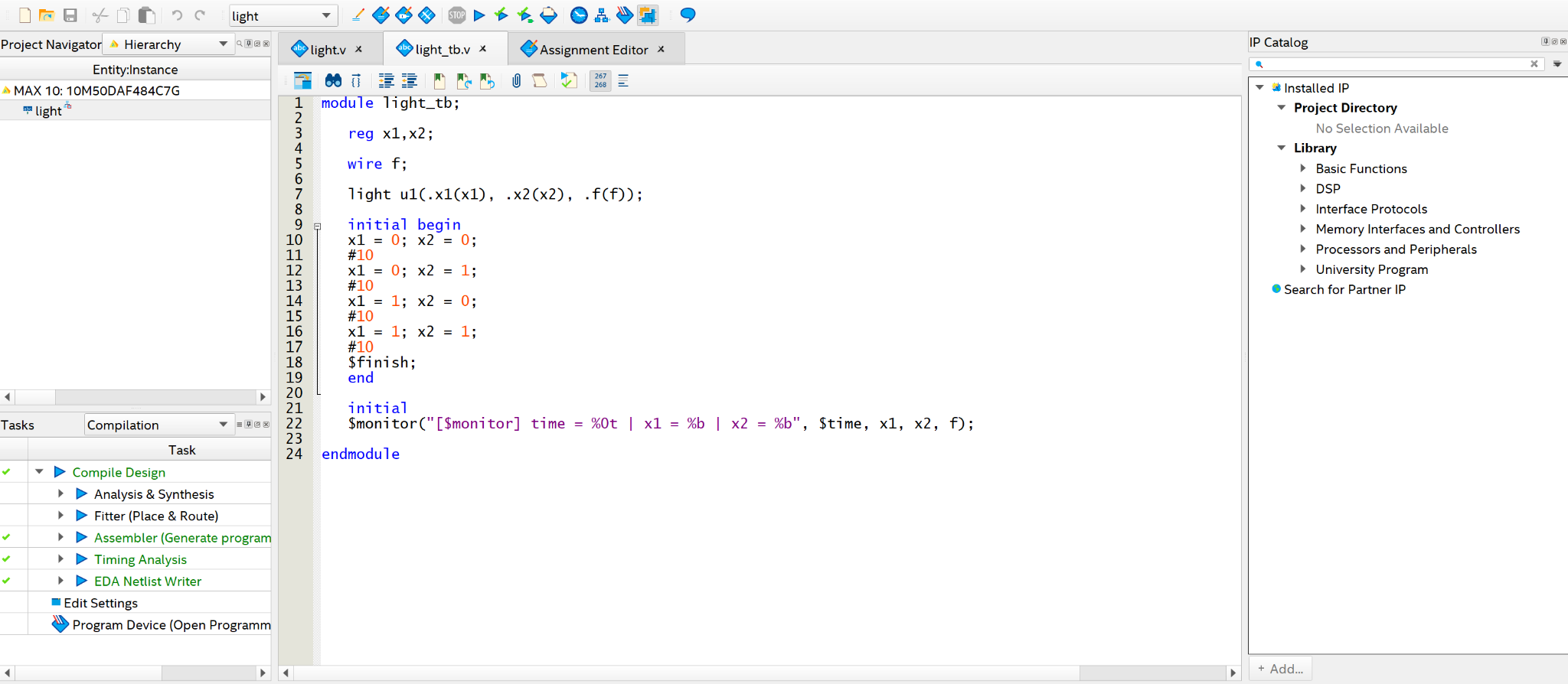
When both *SW0* and *SW1* are ON, then *LEDR0* is OFF.

If we look back at the truth table and compare the inputs and outputs to it, we see that we get the same thing. This proves that the design successfully loaded into the FPGA device and worked as intended.

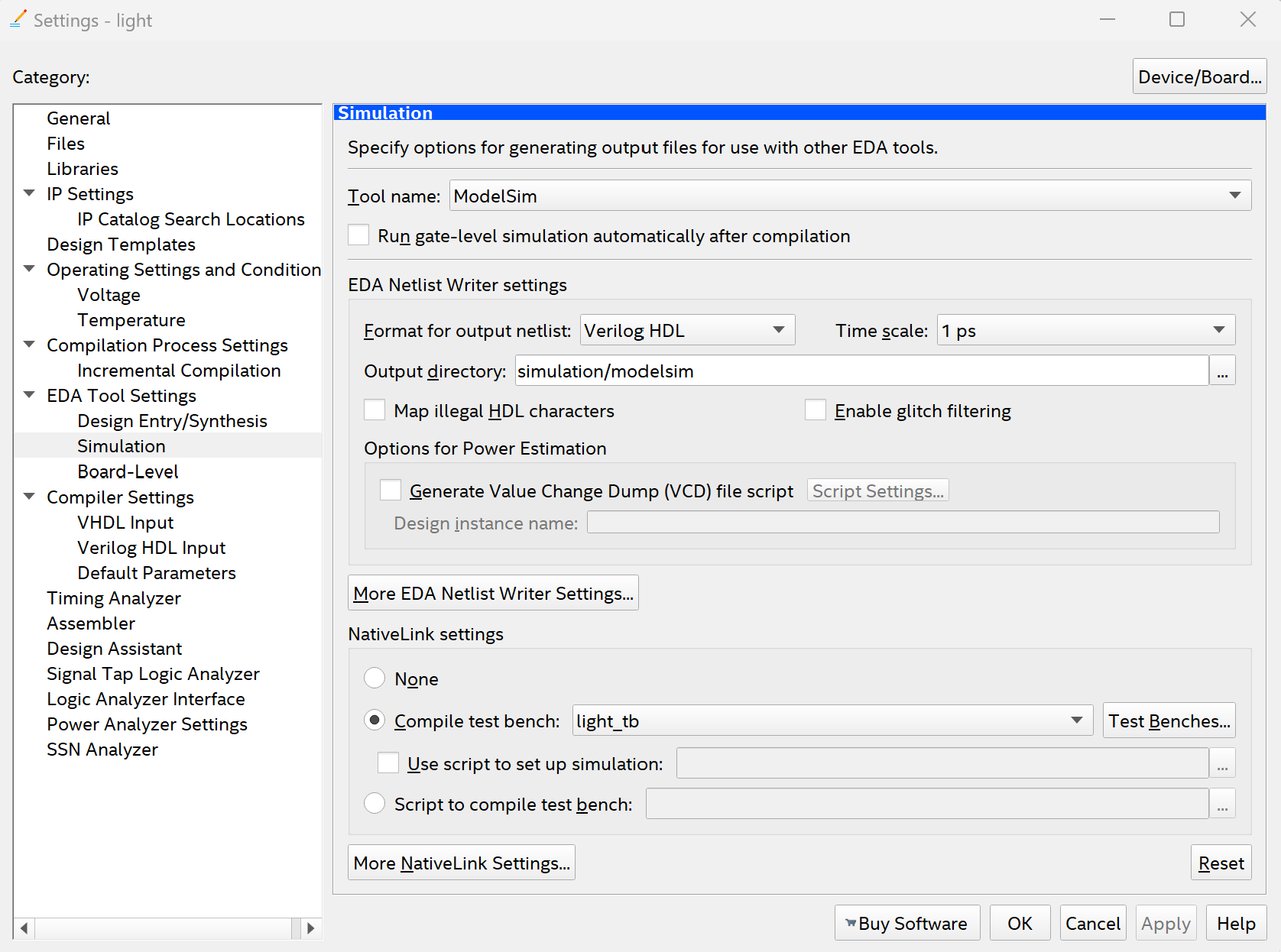
**7. Simulation:**

Another way to test the design is by simulation. We will use three files included in the ModelSim subfolder to control the ModelSim simulator. The files are named testbench.v, testbench.tcl, and wave.do. The testbench.v file is a style of Verilog code known as a testbench. In our case, our testbench file will be named “light\_tb.v”. The purpose of a testbench is to instantiate a Verilog module that is to be simulated, and to specify values for its inputs at various simulation times. In this case the module to be simulated is our two way light controller, which we refer to as the design under test (DUT).

**Testbench:**



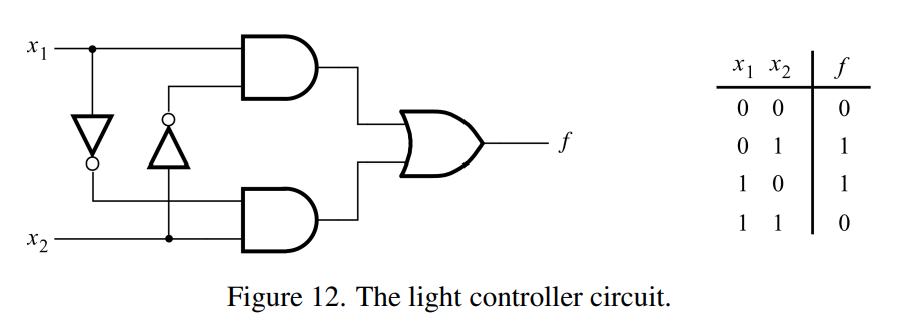
To set up our project for simulation, we go to settings and simulation. Then we identify the tool name to be ModelSim. Then under “Compile test bench: “ we select the name of the testbench to be “light\_tb” and add the file ”light\_tb.v”.



**Waveform:**

To generate the waveform, we click on Tools > Run Simulation tool > RTL Simulation. Waveform should open and display waveform. To do an exact fit of the waveform to the display, we can press “f”.





From the waveform generated, we can compare that to the truth table provided. We see that inputs match the corresponding output, therefore the design is working as intended.