Installing the *DESim* Application

This tutorial shows you how to download and install the *DESim* application on a computer that is running a Microsoft Windows operating system. To perform the steps given below you will need to have access to the Internet, and have the required permissions to install application software.

We do not give detailed instructions in this document for using the *DESim* application, but only show how to install it. Separate instructions for using the *DESim* application are provided in the tutorial called *Using the DESim Software with Verilog Code*. Before using the *DESim* application it is necessary to install the *ModelSim* simulator. Instructions for installing an appropriate version of *ModelSim* are provided in the tutorial *Using the ModelSim-Intel FPGA Simulator with Verilog Testbenches*, available from the Intel FPGA Academic Program.

Getting Started

The discussion below assumes that you are using the Google *Chrome* Internet Browser to navigate on the Internet. If you are using a different browser application, then you may notice some minor discrepancies from some of the material presented below.

You can download *DESim* software installer from its repository on *GitHub*. Open your Internet browser and navigate to https://github.com/fpgacademy/DESim. As indicated in Figure 1, the *DESim* repository includes the source-code for the application. You can browse through this code if interested, but it is not necessary to do so. To download the *DESim* installer onto your computer, use the `Releases` area on the *GitHub* display. As illustrated on the right-hand side of Figure 1 click your mouse on the item ⬦ Version 1.0 (Latest) . This action opens the repository's `Releases` page, part of which is shown in Figure 2.
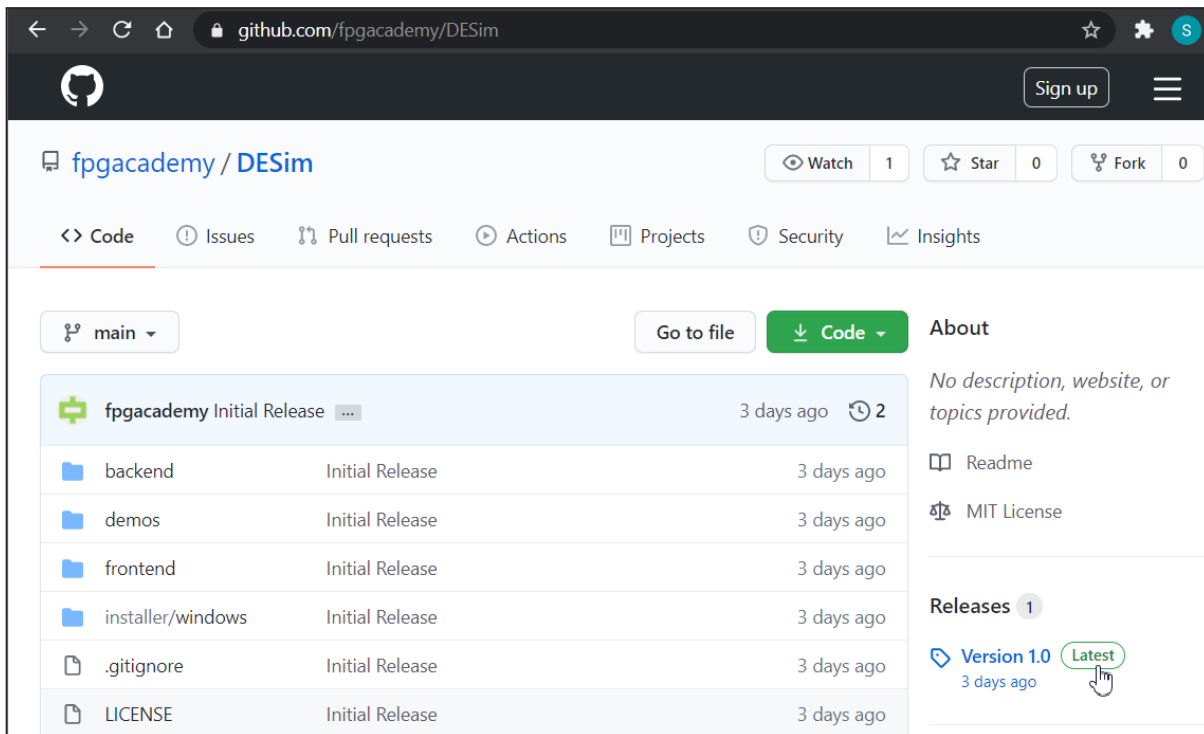

Figure 1: The *DESim* repository on *GitHub*.

Click on the filename *desim_setup.exe*, as illustrated in Figure 2, which downloads this file to your computer. You may be presented with a warning message in your browser, because the file that you are downloading is an *executable program*. Make the appropriate selections in your browser to keep the downloaded file so that it is saved onto your computer.
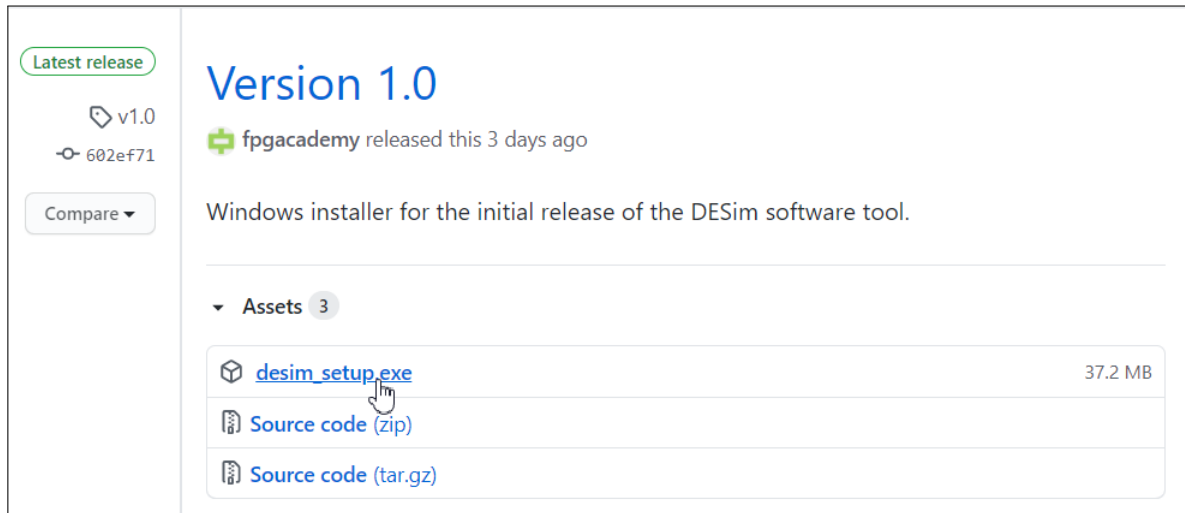

Figure 2: The `Releases` page for the *DESim* repository.

The *desim_setup.exe* file is the *installer* program for the *DESim* application. Open this file (execute the program) to reach the `Welcome` screen shown in Figure 3. Click `Next` to see the *License Agreement* for the *DESim* application, and then click `I Agree` if you accept the terms of the license. If you do not accept the terms of the agreement, then the installer will exit. Click `Next` to reach the screen displayed in Figure 4.
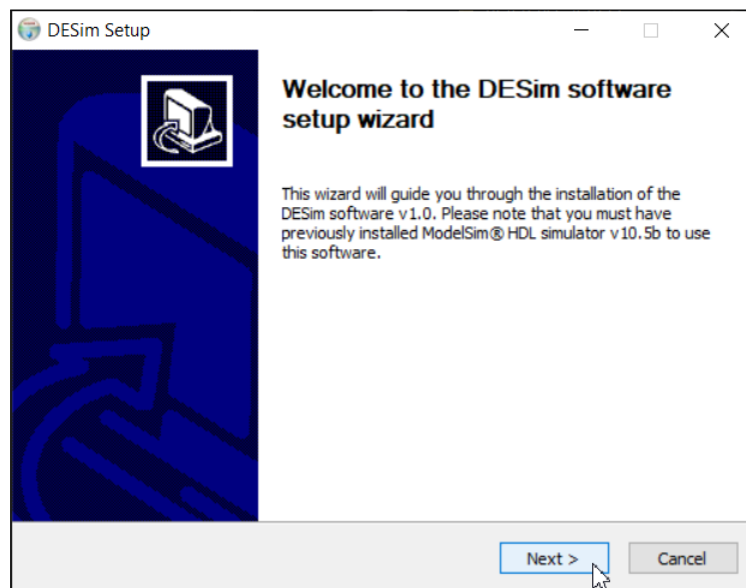

Figure 3: The first screen of the *DESim* installer.

2

In Figure 4 you can specify an installation folder. In the discussion below we assume that you have accepted the default location (`C:\DESim`), but you can change this selection. Click the `Install` button. During the installation process you have the option of placing an icon onto your `Desktop` for this application. This icon provides an easy way to run the *DESim* application, and so is recommended.
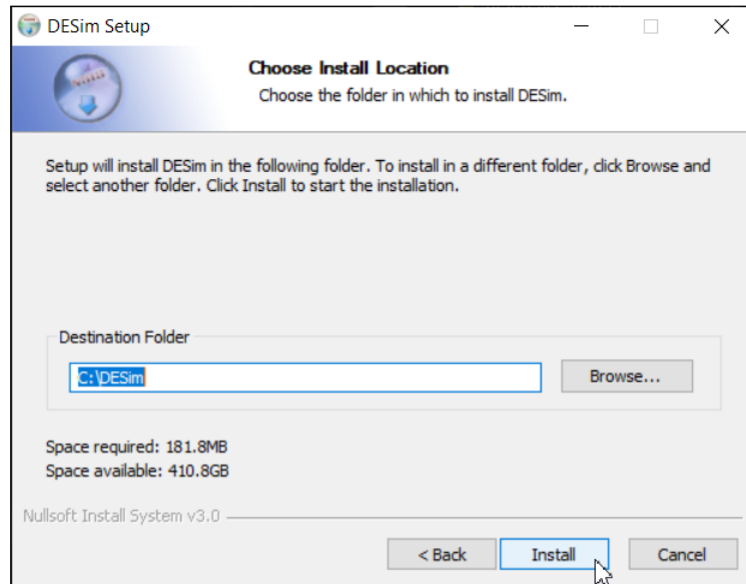

Figure 4: Installing the *DESim* application.

The `C:\DESim` folder created in the installation process contains the *DESim* software and some example projects (called *demos*). Start the *DESim* application by double-clicking its *icon* on your `Desktop`, or by selecting *DESim* from the *Windows Start* menu. Alternatively, you can use `File Explorer` to run the *DESim* application by navigating to the `C:\DESim` folder, right-clicking on the *batch* file `DESim_run.bat`, as illustrated in Figure 5, and then selecting `Open` (or, you can double-click on the batch file to open it). You should now see the *DESim* graphical user interface (GUI), illustrated in Figure 6. It should show the message "The server is running..." near the top of the *message pane* in the GUI.
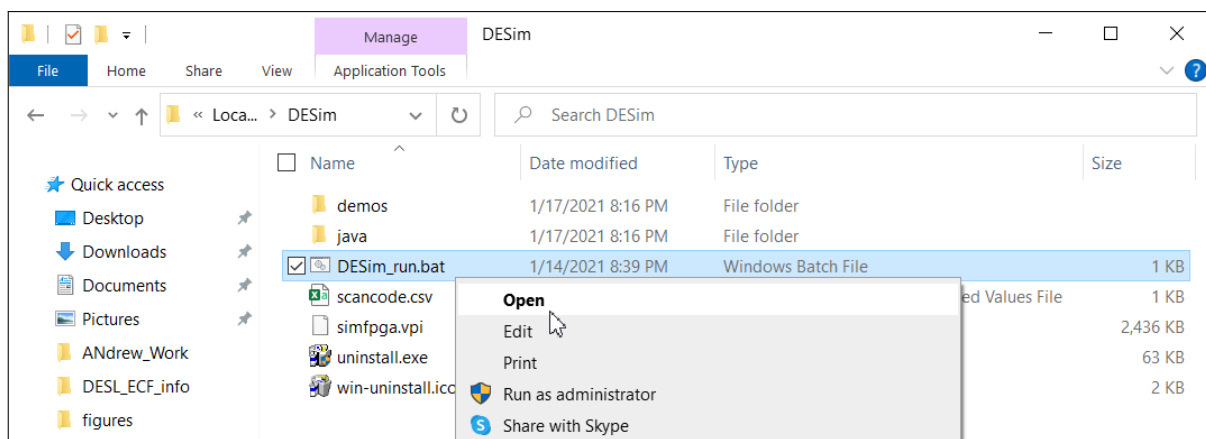

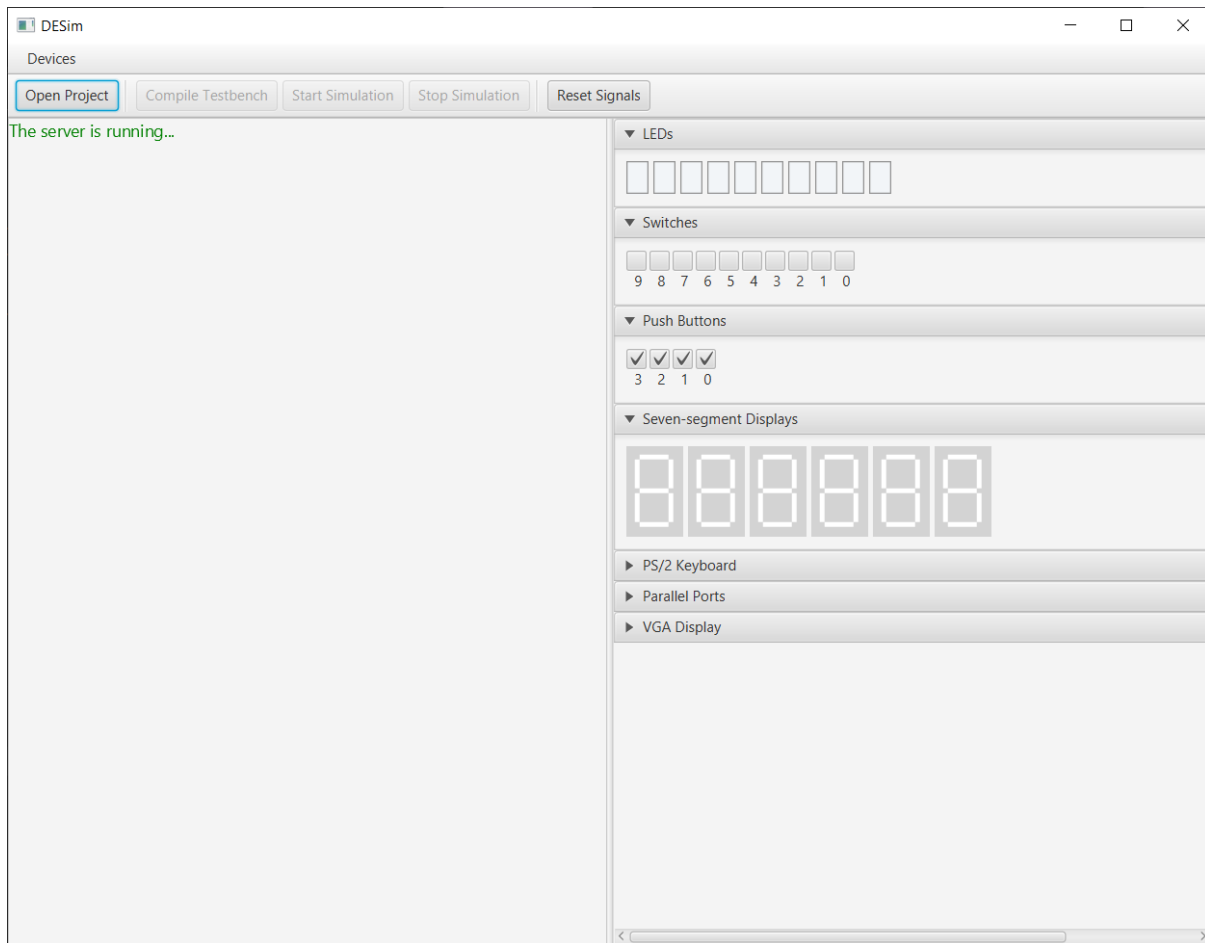Figure 5: Starting the *DESim* application.

Figure 6: The *DESim* window.

To ensure that the *DESim* application can communicate with the *ModelSim* simulator, you may wish to try out one (or more) of the *demo* projects that come with *DESim*. As an example, click the `Open Project` command in the *DESim* GUI and then navigate into the `demos` folder. As illustrated in Figure 7, click to select the folder named `LED_HEX` and then click on the `Select Folder` button.

Click the `Compile Testbench` command in *DESim*. As shown in Figure 8, the ModelSim simulator is executed to compile the Verilog code for the sample project, and the compilation messages that are produced by *ModelSim* are displayed in the *DESim* message pane.

In the *DESim* window select the `Start Simulation` command, which runs the ModelSim Verilog simulator. As illustrated in Figure 9 any messages produced by *ModelSim* are displayed in the message pane of the *DESim* window. To make your display look like the one in the figure, in the *DESim* GUI click on the `Switch` with index number 6, which causes the corresponding `LED` to turn red. To activate the `Seven-segment Display` output you have to reset the `LED_HEX` circuit. To do this, click on `Push Button 0` to press it, and then click again to release this button. To learn about the features of the `LED_HEX` project, you can follow the instructions in its *Readme.txt* file, shown in Figure 10, and/or read through the Verilog source-code file `LED_HEX.v`, displayed in Figure 11. These files can be found using the Microsoft Windows `File Explorer` in the `LED_HEX` folder.

You can stop the *ModelSim* simulation by selecting the `Stop Simulation` command in the *DESim* GUI. To close the *DESim* program click the × in the upper-right corner of the window.
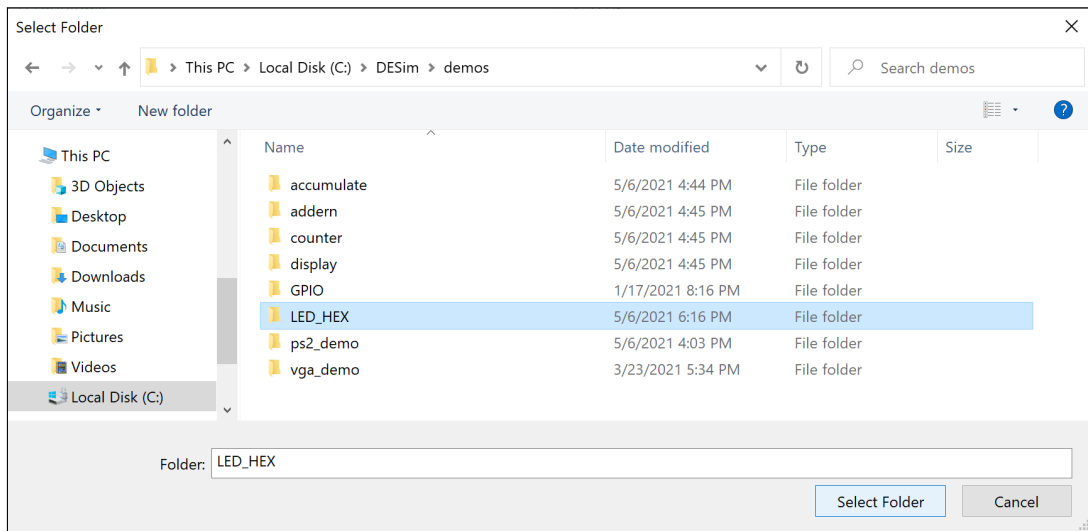
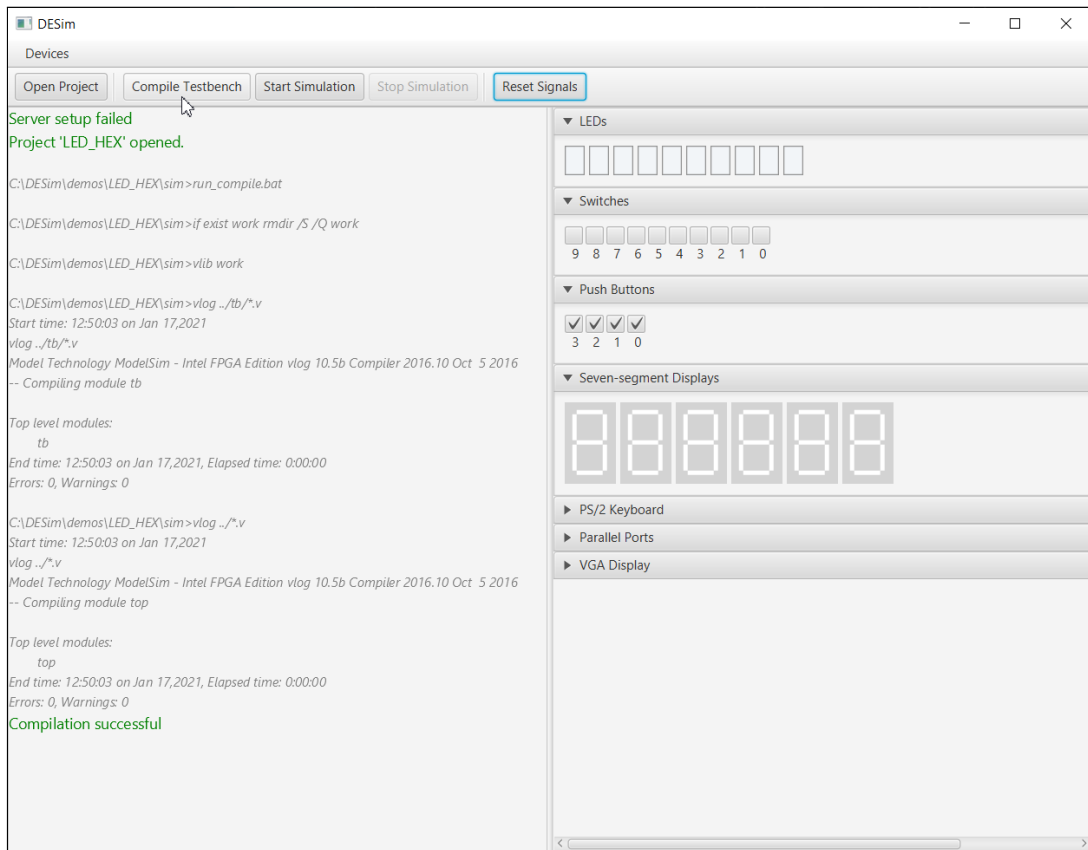Figure 7: Opening a sample project in the `demos` folder.

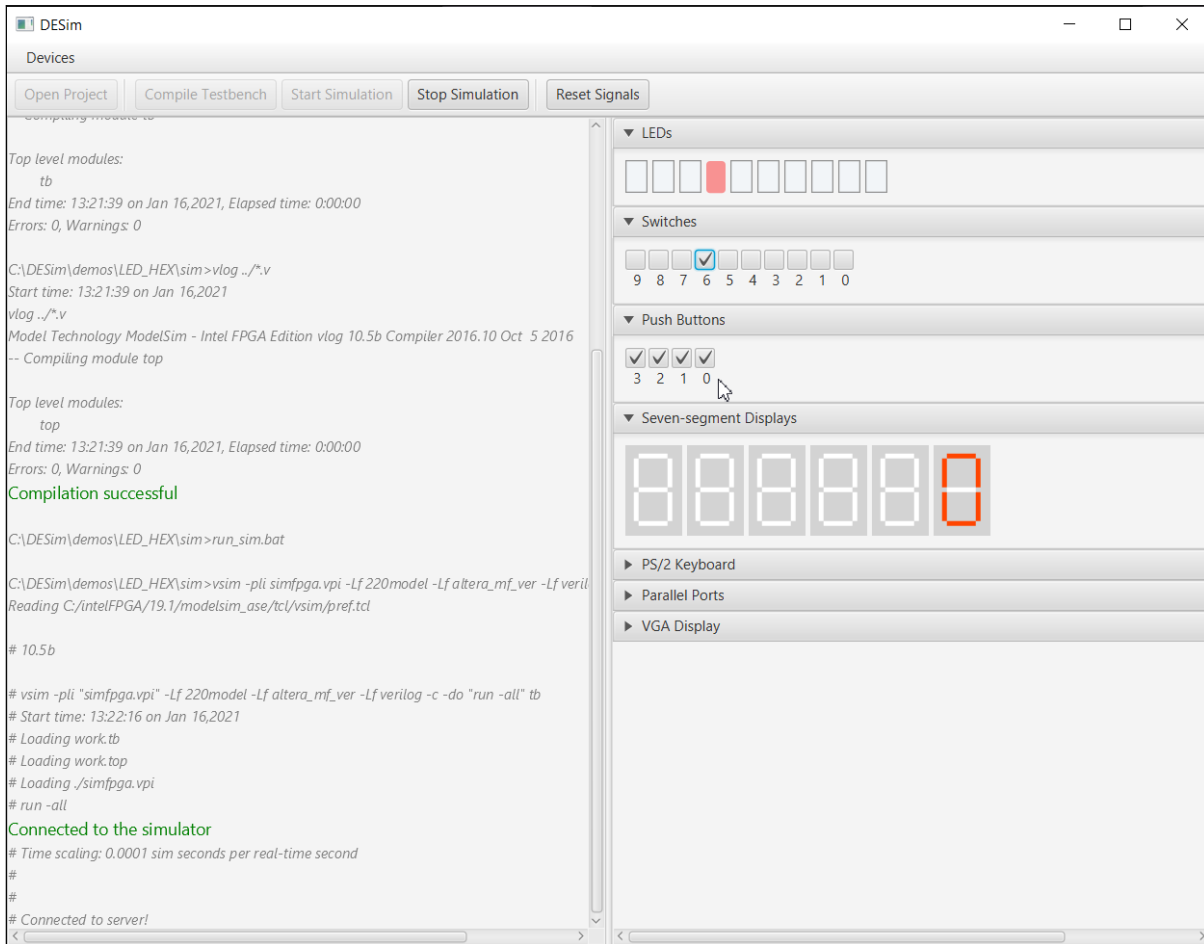Figure 8: Compiling the sample project.

Figure 9: Simulating the sample project.

```
To use this demo:

-- SW are displayed on LEDR
-- KEY[0] is the synchronous reset. It sets the HEX-display
    selector to 0.
-- KEY[1] provides the active-low enable for the HEX-display
    selector

To use:
1. First press/release KEY[0] to reset the circuit; HEX0 is
    selected
    -- HEX0 can be changed using SW[6:0]
2. Set SW[9:7] to select a different HEX display (from 0 to 5)
    -- press/release KEY[1] to store the selected HEX address
    -- the selected HEX display can now be changed using SW[6:0]
3. Set SW[9:7] to select another display, etc.
```

Figure 10: The *Readme.txt* file for the LED_HEX project.

```verilog
module LED_HEX (CLOCK_50, SW, KEY, LEDR, HEX0, HEX1, HEX2, HEX3, HEX4, HEX5);
    input wire CLOCK_50;
    input wire [9:0] SW;
    input wire [1:0] KEY;
    output wire [9:0] LEDR;          // DE-series LEDs

    output reg [6:0] HEX0;           // DE-series HEX displays
    output reg [6:0] HEX1;
    output reg [6:0] HEX2;
    output reg [6:0] HEX3;
    output reg [6:0] HEX4;
    output reg [6:0] HEX5;

    assign LEDR = SW;

    reg [2:0] Addr;                  // used to select a HEX display

    always @ (posedge CLOCK_50)
        if (KEY[0] == 0)             // sync reset
            Addr <= 3'b0;
        else if (KEY[1] == 0)        // select a HEX display
            Addr <= SW[9:7];

    always @ (posedge CLOCK_50)
        case (Addr)
            3'b000:  HEX0 <= SW[6:0];
            3'b001:  HEX1 <= SW[6:0];
            3'b010:  HEX2 <= SW[6:0];
            3'b011:  HEX3 <= SW[6:0];
            3'b100:  HEX4 <= SW[6:0];
            3'b101:  HEX5 <= SW[6:0];
            default: ;
        endcase

endmodule
```

Figure 11: The Verilog source-code file LED_HEX.v.