PyramidTech Internship PTI 22

RTL Design, Validation, and Implementation of an Elevator Controller

1 Purpose

- The objectives of this project are:
  - RTL Design of an elevator controller using VHDL
  - Validation of the design using self-checking testbenches and simulation
  - The hardware implementation and validation of the elevator controller using the Terasic DE0-CV development board.

2 Required tools

Notepad++ (source code editing)
Quartus Prime Lite edition
Questa Intel FPGA starter Edition

3 Required Documents

DE0-CV User Manual, Terasic, May 4, 2015.

4 Design specification

Move the elevator either up or down to reach the requested floor. Once at the requested floor, open the door for at least 2 seconds. Ensure the door is never open while moving. Don't change directions unless there are no higher requests when moving up or no lower requests when moving down. Assume that the elevator moves from one floor to another in 2 seconds. The controller should use the 50 MHz clock on DE0-CV board. Use a 1 sec clock enable to the timer for the elevator movement and the opening of the door. The inputs and outputs of the controller are shown in the figure below. Also, the controller can be broken into a Request Resolver that resolves various floor requests into single requested floor and a Unit Control that moves elevator to this requested floor -see Figure 1-.
Design the controller to serve up to 10 floors (0 is the ground floor and 9 is the highest floor). Use VHDL generics for the number of floors. The floor output should be connected to a seven-segment display. All control signals and status signals will be connected to push button switches and LEDS as shown in the next section.

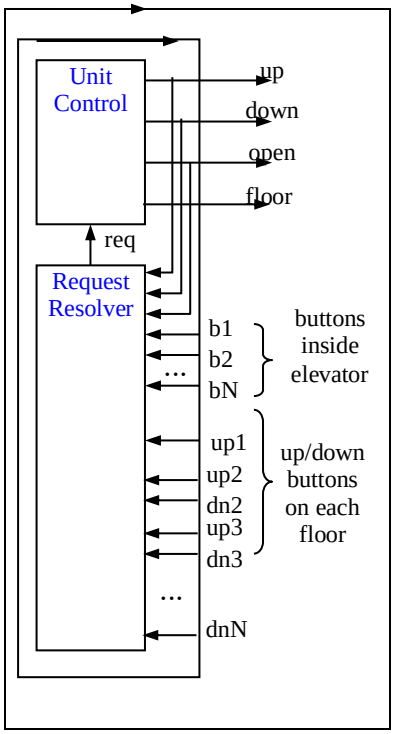

Figure 1: Elevator controller block diagram and interface ports

5 HW Validation Setup

The hardware validation setup of the elevator_ctrl.vhd module is shown in Figure 2. The setup is limited to 4 floors only. The buttons are limited to the ones inside the elevator only the other buttons are hardwired to '0' to be inactive during the test. The 4 control buttons are assumed to be KEY0, KEY1, KEY2, and KEY3. Those key correspond to bn0, bn1, bn2, and bn3. The reset_n is active low and is tied to push button KEY4. The floor count is connected to the units SSD. The status output signal mv_up, mv_down, and door_open are connected to LED0, LED1, and LED3; respectively.

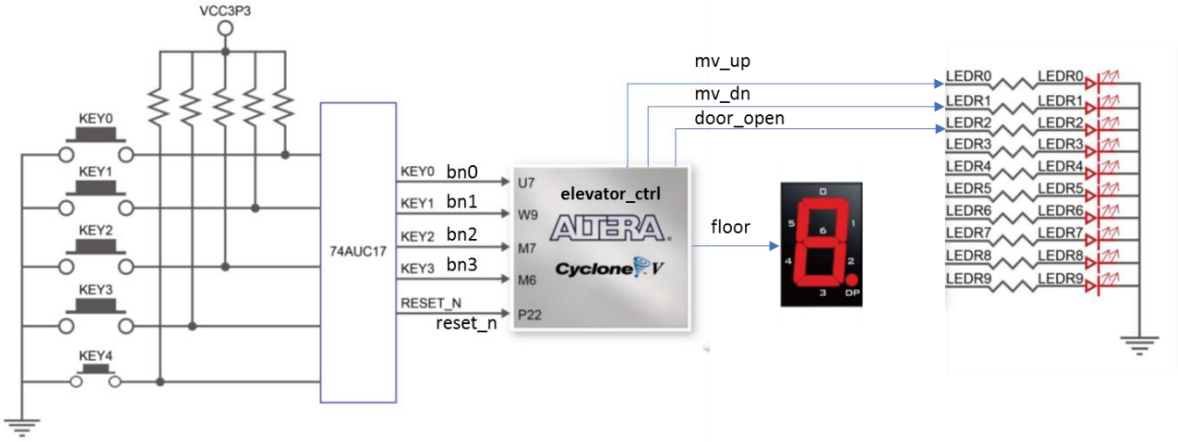

Figure 2: Validation setup for the elevator controller

6 Steps

1. Use the following features for a self-checking testbench for the elevator controller:
   a. Event Stimulus Type
   b. Assertion
   c. Read/Write Files
   d. Variables
   e. External Names and Aliases
   f. Procedure and Functions
   g. Package and Package Body
2. Place and Route
   a. Create a Quartus project and include your design under titles, elevator_ctrl.vhd and ssd.vhd.
   b. Edit the FPGA pin assignments of the inputs and outputs elevator_ctrl.vhd.
   c. Time constraint the input clock using the 2 lines below. Save the constraint file as prbs.sdc. Add this constraint file to the Quartus project using the same steps for adding VHDL files to the project.

```
set_time_format -unit ns -decimal_places 3
create_clock -name {clk} -period 20.000 [get_ports {clk}]
```

3. Generate the configuration file and configure the FPGA on the DE0-CV.
4. Examine the controller with single floor request for both up and down directions.
5. Examine the controller with multiple simultaneous floor requests

7 Needed Submission Files

Submit the following files in a submission directory contains your name on a Google form:

1. A design document showing the state diagram of the finite state machine of the elevator controller (elevator_ctrl_fsm.docx).
   Hint: You can take a print screen from the state machine viewer in Questa tool.
2. RTL files
   a. Seven segment interface (from lab 1) ssd.vhd
   b. The RTL implementation of the controller, elevator_ctrl.vhd.
3. Create at testbench to verify the functionality of elevator_ctrl.vhd. Name this testbench elevator_ctrl_tb.vhd. Make sure your testbench verifies multiple simultaneous floor requests.
4. The pin assignment file, <project name>.qsf
5. The timing constraint file, elevator_ctrl.sdc