WESTERN NEW ENGLAND UNIVERSITY | WNE

Department of Electrical and Computer Engineering

CPE 462/562 VHDL: Simulation & Synthesis

Final Project: Vending Machine Design and Implementation

Katelyn Charbonneau
Email: kc325844@wne.edu
Date: 12/8/2017

1 Objective

The objectives of the project are to design and implement a usable vending machine; be able to write VHDL code to design said machine using a state machine; and implement the system on the Altera DE-115 FPGA.

2 Design Procedure

The top level design block is shown in Figure **??**.

Where Alarm, Clock, Hour, and Minute Set are push buttons, Alarm Enable and Reset are switches, Alarm and AM/PM are LEDs, and the Hex Displays are 6 vectors; one for each seven segment display used in this project.

The digital clock supports setting the time and alarm time on a 12-hour clock with an asynchronous reset. When the current clock time is equal to the alarm time, the alarm will signal and can be turned off with the Alarm Enable switch.

The push buttons on the FPGA are not debounced well. As such, a custom debounce routine was written for each of the buttons; this was then included as a component in the main design. The process used in this code is shown on the next page in Figure **??**.

Because the clock needs to increment every second, we need a 1Hz clock to drive it. Since the FPGA's clock is 50MHz, a new 1Hz clock was made by "slowing down" the original. This was done by counting how many rising edges of the original clock are needed for a second to pass, and changing the value of the 1Hz clock when that many edges passed. The clock counts time by incrementing the one's place of the seconds value by 1 each second, which then cascades down. For example, a 9 in the one's place would next revert to 0 and increase the ten's place by 1; a 5 in the ten's place would next revert to 0 and increase the one's place of minutes, and so on. The hours reset from "12" back to "01", and the AM/PM signal changes when hours advance from "11" to "12". Hitting the reset switch sets both the clock time and alarm time to 12:00:00 midnight and locks them both at that value until reset is released. Reset will also disable the alarm while it is on, however, since both times are 12:00:00 midnight, the alarm will trigger immediately after reset is released. This can be mitigated by also having the Alarm Enable switch set when reset is released; it will suppress the trigger of the alarm.

When the clock and alarm times are equal, the alarm signal is set to high. Originally, it remained high for a minute. During this minute, it could be turned off by flipping the alarm switch, and will automatically turn itself off after a minute has passed. However, if the user turned the alarm switch back off during that minute, the alarm would go back on. This was not the desired behavior, so it was revised. The alarm should stay high for any length of time until the alarm enable switch is set, then remain 0 until the next alarm is triggered. Note that the alarm enable switch allows the alarm to trigger when it is a logic '0' or off, and prohibits the alarm from triggering when it is a logic '1' or on. The present behavior is described by this simple Moore State Machine:

3 Simulation Results

4 Demonstration on FPGA

The design was synthesized onto the Altera Cyclone IV E DE2-115F29C7 FPGA.

- User Controls:
  - Product Select: Switches 17 and 16
  - Product Choose: Key 3
  - Coin Select: Switches 1 and 0
  - Coin Choose: Key 2
  - Cancel: Key 0
- Output Displays:
  - Product Abbreviation: Hex 5
  - Main Display: Hex 2, 1, and 0
  - Delivery Alert: Green LED 8 (LEDG8)

5 Conlusion

The project simulated and synthesized correctly. The code required sufficient knowledge of structural, concurrent, and sequential VHDL. A package was designed to compartmentalize the code to make it easier to read and debug. Recently learned functions and procedures were used within this package, as well as the declaration of the debounce component to make the main code cleaner. There were two main issues in the original compilation of this project. First, the reset behaved in a non-user friendly manner as described in section 2. This issue has been solved as of the current revision. The second issue was that the clock stopped ticking while the user was setting it (it paused). This was considered an extremely poor user-friendly interface, and thus had to be changed. Again, as of the current revision, this issue is solved, and the clock will increment as usual no matter which input the user is interacting with (except for reset). The project could potentially still use optimization improvements to increase speed and reduce the size of the synthesized hardware. In particular, designing a small state machine for turning on/off the alarm is probably excessive for its application.