11/13/2024

**Lab 08 Report**

Ryan Beck

Jared Bronson

**Objectives**

For this lab, build a project on the DE10-LITE that takes an ADC measurement at a 1 Hz rate. Display the reading from the ADC on the appropriate number of seven-segment displays using hex values. Interface a potentiometer to the system and show the displayed value changing as the pot is adjusted. This should be implemented entirely in VHDL.

**Procedures**

Using the system builder create a Quartus project file with the on-board clock, the Arduino header, and the 6 seven-segment displays. Take the generated Verilog file and create a VHDL file with matching port names. After creating the VHDL file, take the Verilog file out of the project. In the generic, create a value for to compare to the sample counter.

Using the IP catalog, create an ADC. The ADC will have an input clock of 10 MHz and a sample rate of 1 MHz. Change the reference voltage to internal, and activate channel zero. Using the IP catalog again create a PLL to pass into the ADC. Change the frequency of the input clock to 10 MHz to match the ADC.

In the architecture, create components to match the generated files of the ADC and PLL. Declare signals to use for the ADC and PLL signals. Next, create a table with values that correspond to each of the 16 hex values for display. Last, create signals to hold the display value, a counter for samples, and a value for when the display updates.

Instantiate the IP blocks for the ADC and PLL. Then, define a process for the on-board clock to add to the sample counter if it is less than the sample period value in the generic. If the sample counter is greater than the sample period, the counter will reset, and the display trigger will be driven high.

Define another process sensitive to the on-board clock. This process will look at the response valid output from the ADC. If the response valid goes high, the response data output will be scaled to a voltage and stored in the display.

The last process will again be sensitive to the on-board clock. This process will update the seven-segment display with the reading from the ADC. Next, compile the design and program to the development board for final testing.

**Results**

The ADC file, shown in figures, successfully reads voltage passed into channel zero of the ADC. The display is updated at a 1 Hz rate on the last three of the seven-segment displays. This is because the data read from the ADC is a 12-bit value. Using a potentiometer interfaced to the ADC, the voltage read decreases with more resistance, and increases with less resistance. This is done using only VHDL.

**Conclusion**

In conclusion, we were able to read an input voltage passed into the ADC and display it on 3 seven-segment displays. This was done using only VHDL in our Quartus project. The value of the voltage is shown using hex values, and is updated every second, or at a 1 Hz rate. A potentiometer successfully shows the value increasing and decreasing with varying voltage.

**A screenshot of a computer program

Description automatically generatedFigures**

Figure 1: ADC.vhd Pt 1

A screenshot of a computer program

Description automatically generatedFigure 2:ADC.vhd Pt 2

*A screenshot of a computer program

Description automatically generatedFigure 3: ADC.vhd Pt* *3*

*A screenshot of a computer program

Description automatically generated**Figure 4: ADC.vhd Pt 4*

A screen shot of a computer code

Description automatically generated

Figure 5: ADC.vhd Pt 5

A computer screen shot of a program code

Description automatically generatedFigure 6: VGA.vhd Pt 6