Refrigerator Thermometer

Keep Food at Safe Temperatures

Joseph Asteefan, Rafal Astifo, Nathan Gilmer, Jacob Monks
Electrical and Computer Engineering Department
School of Engineering and Computer Science
Oakland University, Rochester, MI
jasteefan@oakland.edu, rafalastifo@oakland.edu, nagilmer@oakland.edu, jacobmonks@oakland.edu

***Abstract* - The purpose of this project is to program an Artix-A7 FPGA board using VHDL code to function as a thermometer. This was achieved using processes, temperature sensor, VGA controller, and seven-segment displays. The final code was generated to the FPGA board and demonstrated using an external VGA display.**

I. INTRODUCTION

Everywhere that serves food needs to keep their inventory at appropriate temperatures. The purpose of this project is to create a system for a thermometer that will visually indicate how high or low the temperature is, based on the reading of a temperature sensor. The reprogrammable nature of a FPGA enables exploration for different temperature ranges for different uses. The applications of this include the aforementioned refrigeration of food, keeping track of water heaters and radiators, and industrial uses such as melting down materials or keeping bacteria in check.

II. METHODOLOGY

The basic structure of the thermometer is very simple. A sensor on an FPGA will read the air temperature of the room and will output a binary signal that gives enough precision to accurately determine the Celsius temperature at that time. This can also be reconfigured to give Fahrenheit temperatures, but for the sake of this project, the Celsius scale is used, and it will be applied to refrigeration. The reading will then be sent to the system that converts the binary signal into a decimal number. This is when the Celsius conversion happens. Once that is finished, the temperature will be shown on the 7-segment display, and a VGA display is programmed to show a meter with blue, cyan, green, orange, and red indicators. The blue indicator represents a safe temperature for food to be stored at (below 40°F or 4°C) [1]. For the sake of presentation, however, the display will show the blue block only for temperatures under 28°C. Cyan will be shown for 28 to 31°C, green for 31 to 34°C, orange for 34 to 37°C, and red will show for anything greater than 37°C. Each of these colors are equally divided into 96 blocks as well. For food storage application, if the temperature is above 4°C for any period of time, the food will need to be measured with an analog thermometer and moved to another refrigerator as soon as possible if it is still at a safe temperature or thrown away if it has spoiled.

A. Temperature Sensor

The Nexys-A7 board is equipped with an ADT7420 temperature sensor with resolution of 16 bits and precision for up to 0.25 °C. It reads the atmospheric temperature and stores the value into two 8-bit registers, refreshing every 32 milliseconds. In order to convert the signal to a temperature reading that humans can understand, it must be bit-shifted to the right three times, and then multiplied by 0.0625. The result is a signed floating-point number in Celsius [2]. If it is desired to convert into Fahrenheit instead, take the Celsius value, multiply it by 1.8, and then add 32 [3]. This is a very simple code change that can be done in the program top file. Professor Llamocca's website for VHDL tutorials gives a good explanation of how the sensor works and how it is written [4].

B. VGA Controller

The goal is to use the output signal from the temperature sensor to display certain colors on the VGA display. To do this, the VGA controller must be configured to read the temperature from memory, and the input to the display screen will be a predetermined signal that represents the needed color. The way that VGA display control works is defined by the H-sync and V-sync. These signals tell the system what information to show on the display, based on a 12-bit input to the system. Moreover, the output signals HC and VC tell the system which pixels on the display need to be used [2]. The range for the colors on the display should scale to whatever kind of monitor is used, but for the sake of

presentation, the display used is a 640x480, 30 Hz monitor. The basic structure for VGA control was found from Professor Llamocca's website [4].

C. Seven-Segment Serializer

The 7-segment display on the Nexys board is programmed to show the Celsius temperature that the sensor is reading. The 9 MSB of the temperature are used. These 9 bits are equivalent to the whole number of that temperature. This 9-bit value is converted to an integer then divided by 10 get the tens digit. Then the modulus operator is used to get the ones digit. The way the display works is that each digit has an anode signal that needs to be set to show anything, and each of the seven segments on an individual digit has a cathode signal for turning each segment on and off. However, each digit on the display is connected to the same cathode signals, which means at any given time only one number can be shown [2]. In order to circumvent this, the display needs to be serialized. This means that it will cycle through the digits on the display and show the numbers necessary for those digits at a fast enough speed that the human eye cannot notice it is even changing. This is done by using a counter and state machine to set up the component to operate on a certain frequency. And a multiplexor is used to determine which digit is being displayed at any given time. Also, to display the correct digit, an integer to seven-segment decoder is used. For the serializer, the code from Professor Llamocca's website was sufficient with few changes necessary [4].

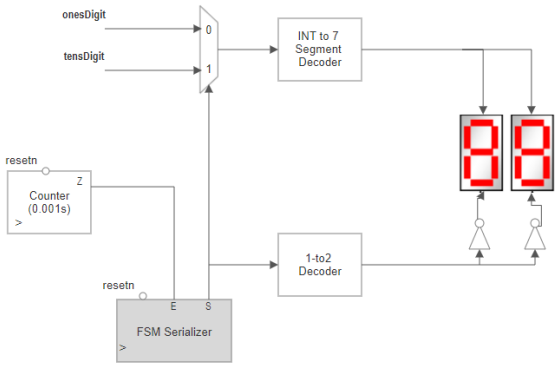

Figure 1: Seven-Segment Serializer

D. Top File and Processes

In the top file, two processes are used to correlate the temperature readings and VGA control. These processes execute the data through the use of if-statements to display the correct colors and blocks on the VGA display.

D1. Color/Block Placement Process

This process practically decides what color to display based on the temperature. This also sets the parameters for each block using the H-count and V-count variables. Figure 2 shows exactly what parameters are set.

Color	Temperature (°C)	Height (V-count)	Width (H-count)
Blue	T < 28	480 to 384	160 to 480
Cyan	28 < T < 31	384 to 288	160 to 480
Green	31 < T < 34	288 to 192	160 to 480
Orange	34 < T < 37	192 to 96	160 to 480
Red	T > 37	96 to 0	160 to 480

Figure 2: Color and Block Parameters

D2. Block Height Process

This process accounts for the gradual increase of the thermometer. The red and blue blocks do not gradually increase, as they are just there to show if the meat is completely freezing or completely overheating. The middle blocks however, all need to gradually increase with the temperature like an actual thermometer. This was done by using some simple arithmetic. Each block is 96 rows, and each middle block has a temperature range of 3°C. Now 3°C divided by 96 rows is equal to 0.03125°C per row. This temperature is equivalent to having "0000000000000100" or 4 bits for 16-bit temperature value. Essentially, the thermometer needs to increase or decrease a row every 4 bits of temperature. This was done by subtracting the temperature reading by the block's minimum temperature and then dividing by 4 bits. Diving by 4 is equivalent to bit shifting twice to the right in binary. Once this is done, the block's height has been found for that temperature. When setting the v-count parameter for that temperature reading, the base height of the block was subtracted by the block's height to correctly display the temperature on the thermometer. Note that the block's height is subtracted and not added because the thermometer needs to go up the screen, and this is done by subtracting for VGA.

Back to the top file as a whole, it consists of all these components put together, as seen in Figure 3. The temperature sensor values are fed into the serializer and VGA control to display the temperature correctly.

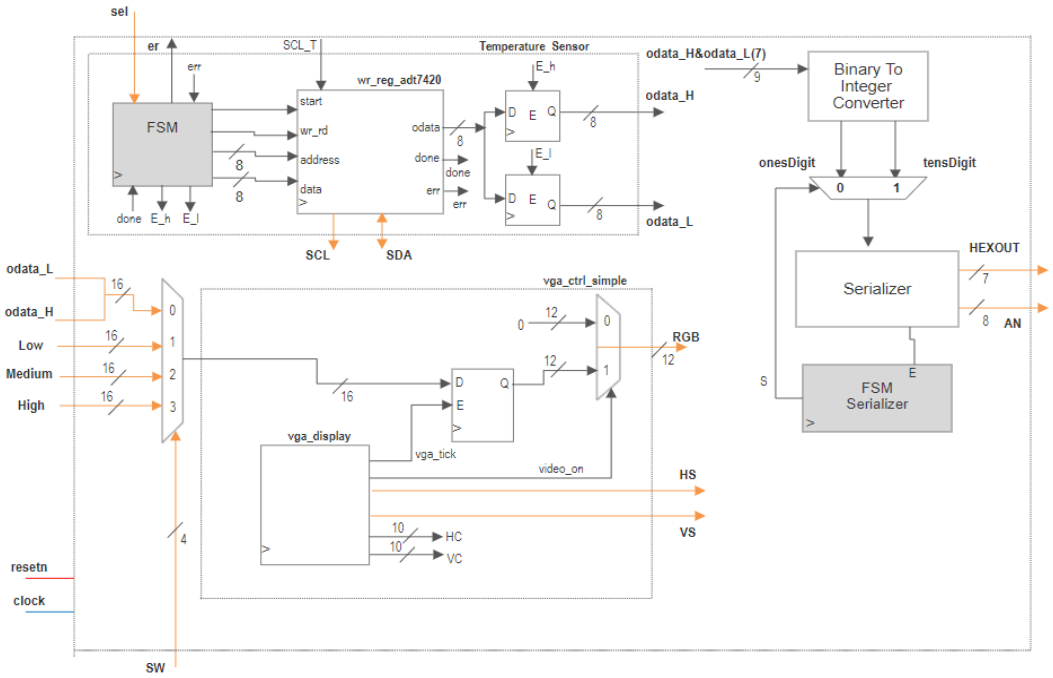

Figure 3: Top File Schematic

III. EXPERIMENTAL SETUP

VHDL code and Vivado was used to program the Artix-A7 FPGA board. A reset button, built-in clock, temperature sensor, VGA port, and seven-segment display were utilized in this project. A VGA cable and monitor were also used to actually display the thermometer. Switches 0 to 4 were used to decide what temperature to read. Switches 1 to 3 have pre-set temperatures, SW[1] being low temperature (below 28℃), SW[2] being medium temperature (between 31℃ and 34℃), and SW[3] being low temperature (above 37℃). These are primarily used for simulation purposes and to verify the project works. SW[0] turns on the temperature sensor reading, and SW[4] allows the temperature sensor to write that data onto the board. These two switches must both be ON to make the temperature work correctly.

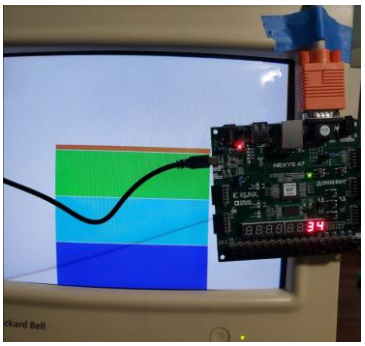

Figure 4: Hardware Setup

IV. RESULTS

The final outcome of our experiment resulted exactly as expected with a few minor complications. The data retrieved from the temperature sensor is sent directly to the seven-segment and the VGA display, showing the resulting temperature value. The seven-segment display demonstrates the temperature results as an integer value as expected. While the VGA display demonstrates the step-by-step increase and decrease in temperature data retrieved from the sensor, to better resemble a thermometer. The color specified for each temperature range appeared as expected, but minor glitches were noticed on the VGA display when the color blocks were increasing/decreasing in size. While the color blocks changed in size per temperature data, the color for the specified range would glitch and disappear for a brief second and reappear as it continuously changed. We did not notice this error when we did the simulation test. Although we were unable to test the temperature sensor through the simulation because it is not possible to do so, we implemented switches to show the expected outcome of our application. Due to the restraint of the time length and number of cycles required to run through every pixel on the VGA display, only one switch at a time can be tested through simulation. Each switch was tested, and the results were as expected. When the switch for the low value is set to ‘high’, only the blue block is displayed

at that temperature value and the remaining pixels are set to white. The results for each of the switches appeared in the same manner in accordance with the defined ranges for each temperature value. The simulation results also showed that when the temperature value reached a different temperature range, there is a row of white blocks in between the transition from one color block to the next. This is better observed in Figure 5, demonstrating the space between the orange and green block as the VGA is scanned through from top to bottom.

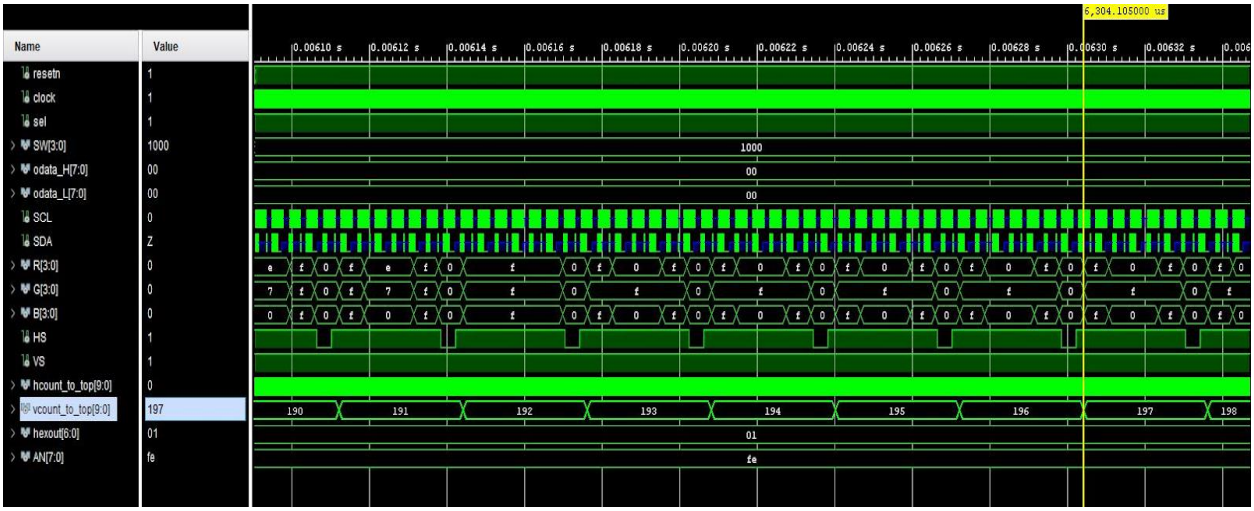

Figure 5: High Temperature Simulation Results (Green Block)

CONCLUSIONS

Our application for a thermometer was a success. The temperature ranges selected for our project were set so high solely for the purpose of seeing the resulting outcome for each scenario. Those ranges can be adjusted for future uses to better fit the requirements of the application. While we did have some difficulties in showing the simulation results of our project, the live demonstration of our application ran successfully. Since our goal was to create a thermometer, our project can be improved by showing the integer value of the temperature on the VGA display rather than the seven-segment display to make it easier to read. Another possible improvement could also be showing the color density for each block increase gradually as the temperature increases. And possibly, incorporate noise to indicate overheating or freezing temperature ranges. These improvements will each benefit our application in showing a more similar resemblance to a realistic thermometer.

REFRENCES

[1] “Are You Storing Food Safely?” U.S. Food and Drug Administration. 9 February 2021. https://www.fda.gov/consumers/consumer-updates/are-you-storing-food-safely

[2] Brown, Arthur. “Nexys A7 Reference Manual.” Nexys A7 Reference Manual - Digilent Reference, Digilent, reference.digilentinc.com/reference/programmable-logic/nexys-a7/reference-manual

[3] “Celsius to Fahrenheit (°C to °F).” Metric Conversion, Wight Hall Ltd., 2020, www.metric-conversions.org/temperature/celsius-to-fahrenheit.htm.

[4] Llamocca, Daniel. VHDL Coding for FPGAs, www.secs.oakland.edu/~llamocca/VHDLforFPGAs.html