ECE 583 Final Project Report

Robert Lee
University of New Hampshire
Computer Engineering
Durham, New Hampshire
rl1105@wildcats.unh.edu

Ryan Conte
University of New Hampshire
Computer Engineering
Durham, New Hampshire
rc1095@wildcats.unh.edu

***Abstract*—This document is a report for Robert Lee's and Ryan Conte's final project for ECE583. It details the motivations, challenges, methods, results and more that were achieved during this project.**

I. Introduction

This document is a report on showing keyboard key scan values, translated to ASCII, on the Basys 3 seven segment LED and then saving the values into memory that is accessible through the switches. The document will also detail how the memory is pop-able through using the Basys 3's center button. This project was created to help understanding in SystemVerilog as well as different functionalities in the Basys 3 board.

A. Motivation

The motivation of this project was that the group really wanted to implement something the group had never learned or implemented. After bouncing many ideas off each other, the group had stumbled upon a GitHub project that used the USB HID host port on the Basys 3. Intrigued, the group decided to pursue this idea with a bit of flair. Coming up with the idea of using the switches on the Basys 3 as a way to access memory, the group decided to use the built in keyboard key scans, turned into ASCII values, as savable inputs, which could then be popped by using the Basys 3 center button.

II. Ease of Access

A. Abbreviations and Acronyms

- top - The "main" module that called to other modules
- keyReceiver - Secondary module that received the key scans from the keyboard
- PS2Clk - Clock from keyboard input
- kclk - Clock in top module
- oflag - Flag that sets when count2 variable reaches 3 inside PS2Receiver module
- rflag - Flag that sets when oflag is 1 inside top module
- key scan - 8-bit hex value that was pre-implemented on the Basys 3 for keyboard input
- ASCII - 8-bit hex value of keyboard's character encoding
- count - counter varible that counts incoming bitstream from keyboard
- count2 - counter variable that counts amount of 2 word strands coming from keyboard

III. Report

A. Challenges

The group had faced many challenges along the way while trying to implement this system. The first challenge was one the hardest by far due to neither of the members knowing how to use the USB HID host on the Basys 3. However with much perseverance and a lot of research, the group had been able to implement the input from the keyboard as the basic key scans that were already built into the Basys 3.

The next challenge was implementing the "shifting" part of the key scans. Although this part had seemed quite straight forward, there were many unforeseen difficulties in the timing, as well as in the logic, of figuring out when to shift the bits. This was overcome by much perseverance and trial and error.

The challenge faced after that was changing the key scans from the keyboard into the resulting ASCII values. This challenge was definitely more straight forward when it came to the major challenges of the project however it was still a challenge that the group had faced. It was rather a more tedious challenge rather than a logical challenge however once again it was overcome through trial and error and perseverance.

The second to last challenge that the group faced was using the switches to access memory addresses and saving the inputs from the keyboard into those addresses. In addition to this challenge, the group also had the challenge of making the LEDs light up on a save.

The final challenge was taking the saved inputs from each of the memory slots and being able to pop them, causing the saved memory in that address to be reset back to four zeros. The challenges that were faced happened in around this order allowing the group to be flexible with changing pieces back and forth from the Verilog.

B. Methods

The method used to receive the data from the keyboard was to implement a module called keyReceiver. The group had created this module due to the way that the Basys 3 had implemented the way it gets key scans from the keyboard. The way the key scans are inputted to the Basys 3 is a serial BUS containing a one-bit clock and a one-bit data stream. To grab the bits coming from the keyboard the group created this secondary module to help respond to each of the PS2Clk positive edge changes. It does so by using an always

@(posedge kclk) block, as every positive edge clock comes in, a bit of data from the key scan also comes in. On the inside of this module, there is a switch statement which branches based on the value inside of counter.

The counter variable keeps track of how many bits of the word have been received from the stream. When it is at zero, it represents that a start bit has not been transmitted yet. When it is greater than zero but less that 10, it represents that the data is in the process of being transmitted. When the counter reaches 10, it means the end bit received, and the module increments the count2 variable, which represents the number of words transmitted since the oflag has been set. Since there is one word-transfer on a key being press, and two words transferred on the key being released, the only relevant value is the last of the three words transmitted since the last word is the same as the first word. When the count2 variable reaches 3, the oflag is set, representing to the top module that the code for the key has been successfully transferred.

Contained in the top module is an always @(posedge rflag) which first checks the key scan against the enter key's key scan. If the enter key was pressed, the module saves the value currently displayed on the 7-segment display to memory. The memory address is determined by the switches on the board. If it was not the enter key, the module will write the newly processed byte onto the display after shifting the value on the display a byte to the left. This allows for address selection and flexible reading/writing to these addresses.

Creating the part where the ASCII values would shift over in the seven segment display was by far one of the more difficult challenges. Despite sound logic, the timing seemed to always be a bit off when trying to input the second ASCII value in. However, with many different attempts at trying the timing for the shift as well as changing the way that the different counter variables were used, the group was able to make the shift work smoothly and efficiently for one key press at a time.

Changing the key scan values to be ASCII values was quite straight forward although it was tedious. The Basys 3 has a built in keyboard of key scans for when a USB keyboard is implemented. So what the group had done was take every single one of the key scan codes from the keyboard and used a switch case table to change them into showing the ASCII value of the letter rather than showing the key scan. This was the best way to do this change due to the switch cases being easier and quicker to implement rather than using a large amount of if else statements.

A lot of the challenges that were encountered while during this project were really solved by a lot of trial and error and perseverance. Going thoroughly through the logic and syntax to make sure nothing was incorrect was the main way a large amount of the challenges were solved. Due to this, everything had worked out well together. Such as with the using a secondary module to grab the key scans and using the switch case to make the key scans read into ASCII. From the way that we had encountered and overcome every challenge, the module had worked out perfectly to what the group had set out to accomplish.

C. Results

In the end, the implementation worked the way it was intended to work. When the board is first powered on, it shows 4 zeros on the display. When a valid ASCII key is pressed, the ASCII code is displayed on the screen. The user can continue to type keys, and the board will display the most recent 2 presses. When the user hits the enter key, the LEDs will light up for 2 seconds, showing the value was saved into memory at the address corresponding to the switches on the board. When the switches change state, the value stored in the corresponding address is displayed on the 7-segment display. The user can press the center button to ”pop” the lower indexed value, and fill with zeros on the left. No changes to any given memory slot will be saved unless the ENTER key is pressed before switching the address again.

Our group had set out and successfully created a USB HID host keyboard input into 16-bit memory slots. All functionalities that were thought of were able to be implemented quite well and in a manner of being easily read when looking over the top module. Despite all the challenges and difficulties encountered, the group was able to persevere and overcome every challenge to create exactly what was detailed in this report.

IV. Conclusion

In conclusion, the group had accomplished exactly what the group had set out to do. By using the Basys 3's different functionalities, the group was able to implement a USB keyboard that could input different key scans that were then translated to ASCII into savable memory slots by using the switches as addresses to access the slots of memory. All of the inputs were able to be shown on the seven segment display, shifting every value to the left and the last two values that were entered would be able to be saved into the memory slot. Also the group was able to implement a pop feature that uses the Basys 3's center button to pop out the memory inside of an address. Despite all of the difficulties and challenges faced along the way, all of these functions were implemented correctly and efficiently allowing for a fast and easily used keyboard input saver.

This was an amazing learning experience for the group as it really honed in the groups different skills in Verilog and technical writing skills. It forced the group into learning while implementing and taught a valuable lesson in being movtiated to learn new and exciting things. With the project finished, it left the group to be more efficient in implementing Verilog, learning new things and better technical writing skills.

References

[1] *Basys 3 Reference Manual*. Retrieved from https://reference.digilentinc.com/basys3/refmanual

[2] *ASCII table , ASCII codes*. Retrieved from https://theasciicode.com.ar/

[3] *Basys 3 Keyboard Demo*. Retrieved from https://reference.digilentinc.com/learn/programmable-logic/tutorials/basys-3-keyboard-demo/start

[4] *GitHub Basys 3 Demo*. Retrieved from https://github.com/Digilent/Basys-3-Keyboard