# 5-bit Signed 5-Function Calculator

Joshua Arnott, Nicholas Musienko
Electrical and Computer Engineering Department
School of Engineering and Computer Science
Oakland University, Rochester, MI
e-mails: jfarnott@oakland.edu, nmusienko@oakland.edu

## I. Introduction

This project involves the creation of a basic 5-bit input calculator that can add, subtract, multiply, divide, and provide the modulo of those two numbers. The following are the results of our project and implementation onto a Nexys-DDR board via VHDL.

People use a calculator like the one achieved by this project daily. These calculators have become so ubiquitous that nearly everyone has access to one via the phones in their pockets. To do this, the usage of full adders, shift registers, multiplexors, decoders, and state machines as learned in class was necessary. Also, the use of outside sources, in this case a keyboard, was used to input data which intern was output via on board LEDs.

## II. Methodology

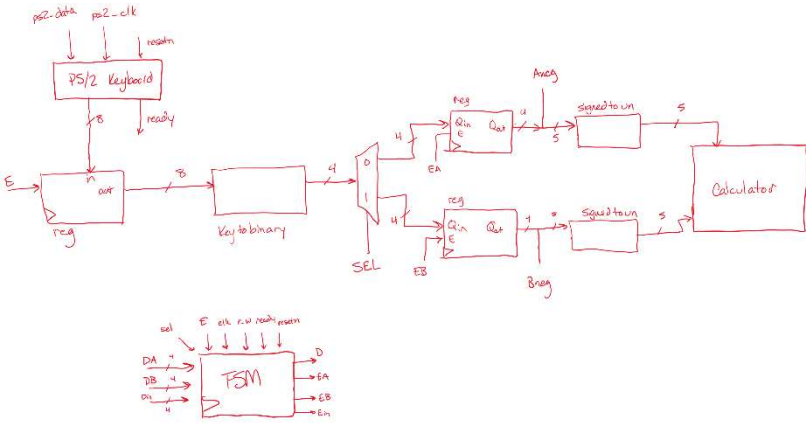


**Figure 1: Block Diagram of Basic Program Flow**

### A. Inputs

To verify that the components work as required, the input method to start was the switches on the Nexys-DDR board. There were 2 rows of 5 switches dedicated for data input, 3 switches for functions, and a read/write switch. As soon as the circuit was verified working with the switches, a keyboard was connected to the Nexys board via USB. The input signal processing was modified slightly, but the rest of the program adapted to the different input method with no problem once the keyboard input was fully working. To use a keyboard input, a finite state machine was needed to read the incoming signal from the keyboard and output the data in binary along with a done signal, signifying the presence of new data from the keyboard. Code used from the professor's website comprised the majority of the interface with the PS/2 keyboard, but some modifications were made to alert the rest of the circuit to the presence of new data [1]. We were then able to use this data in the calculator portion of the circuit. Also, the switches were still used in this implementation for signing the unsigned input from the keyboard, toggling which input register the keyboard writes data to, the different math functions, an enable switch, and a read/write switch. These switches along with the inputs from the keyboard allowed for the math functions to occur with the proper input. This combination of keyboard and switches allows for the most intuitive input scheme manageable while also keeping overall complexity within tolerable limits.

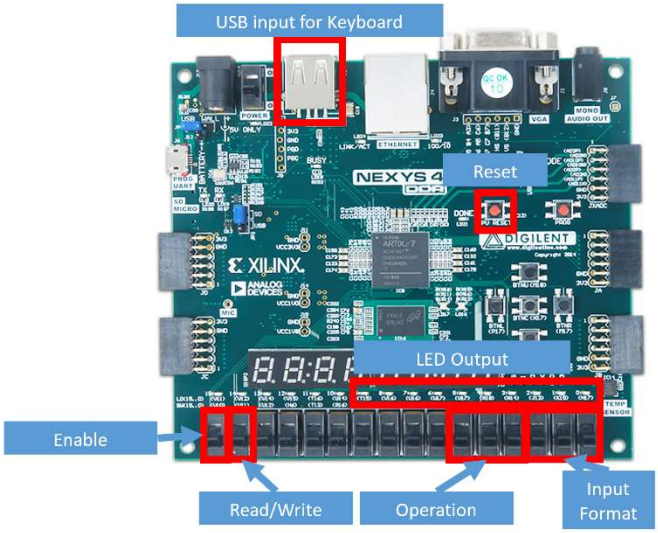


**Figure 2: Picture of inputs used on the Nexys-DDR board for the final implementation**

### B. Functions

For the calculator to function properly, an array of functions will need to be created or modified from code created from class (as seen in Figure 1). After being processed

through a keyboard to binary look-up table, the two inputs, now binary sign and magnitude numbers, will then be written to the outputs, and then gets processed through a sign and magnitude to 2's complement converter, as the calculator is made to accept 2's complement numbers for its processes.

When the read/write switch goes high, then the two 2's complement numbers get written to registers that allow the numbers to pass into the calculator circuit. Inside the calculator circuit, one major finite state machine handles the inputs and outputs overall, toggling on and off the multiplication and division portions of the circuit. The read/write switch going high actually causes the state machine to move to its next phase, activating the multiplication and division circuits.

The addition and subtraction units use the 2's complement numbers and a pair of 5-bit adders, outputting two 6-bit numbers including the carry-out. These 6-bit numbers go through a pair of formatters that convert those numbers to signed 10-bit numbers for storage in a pair of registers.

The multiplier and divider circuits are slightly more complex. First, the two inputs get formatted into two unsigned numbers, the most significant bits of each getting run through an XOR gate to get the sign of the output. Upon receiving an enable input, the divider, a modified version of one constructed for a laboratory activity, activates, performing a series of iterative subtractions controlled by its own finite state machine. This circuit outputs the division and modulo results of the two inputs and a done output. Likewise, the multiplier circuit activates with an enable input. This iterative adder was created based of notes from the class and is controlled by its own independent state machine [2]. This circuit outputs the product of the two inputs and a done output.

When the main FSM receives these two done outputs, it transitions into its final state. It enables the multiplexor that allows for the output of the calculator circuit and enables the registers that store the results from the independent circuits. Switches on the Nexys board drive the select lines for that multiplexor, allowing to change the operation requested without having to re-run the calculator circuit.

### C. *Outputs*

As described above, the output of the functions is eventually displayed on the onboard LEDs on the Nexys-DDR board. When the read/write switch is in the write state, ten LEDs are illuminated depending on the sign and magnitude number that is being displayed. The first five LEDs, zero through four, are for the first number input in sign and magnitude form. Similarly, LEDs five through nine display the second input in sign and magnitude form. Then when the read/write switch is set to read, the output value from the calculator is shown along with a done LED which is LED fifteen, signifying the completion of the FSM controlling the calculator. The output value is also in sign and magnitude for ease of reading. To achieve this, a multiplexor is used to choose between the input values and the output value, with a select line connected to the read/write switch.

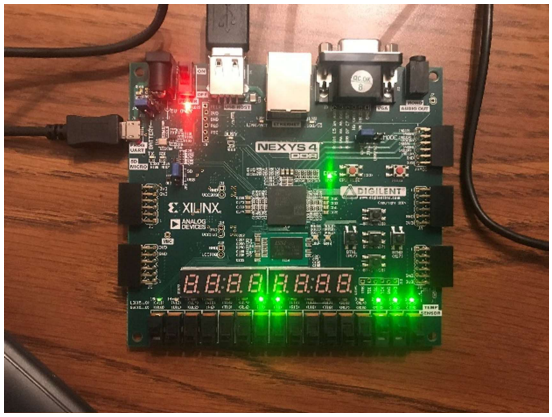


**Figure 3: LED output display, showing two inputs of 7 and 8.**

## III. Experimental Setup

The initial phase of testing of the project was developing a test bench simulation source to test if the calculator was functioning properly. To test our system on the board we chose to initially start by just isolating the calculator and input to the calculator directly from the onboard switches. This was tested rigorously until it was verified that its operation was correct. Similarly, when testing the inputs from the keyboard, the code to receive the data from the keyboard was developed independently from the calculator and eventually implemented once tested into the full project. An iterative design process was used, where changes were made, then tested via test bench and bitstream programming. The results from this testing, both in experience and function, was then used in the creation of further changes.

## IV. Results

The biggest results learned through this project were the development and implementation of finite state machines. Through this project, finite state machines were used in the calculator and in the program to read the keyboard, and we learned how effective and useful these can be for implementing multiple modules concerning different outputs at once.

Another big result of this project was the development of a calculator with a keyboard input that can display outputs on LEDs. A human-friendly input method was used to push data into our system, and the data was then readable, what we set out to do. We were successful in our goal of creating a calculator that could do math on two five-bit signed numbers and output an answer in a human-readable way, as demonstrated by the figures below.

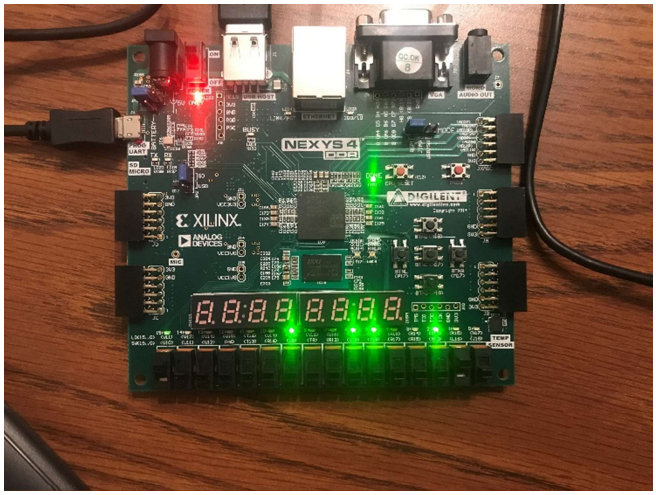

**Figure 4: LED output of the two input numbers from the keyboard, 4 on the right and -3 on the left 5 LEDs.**

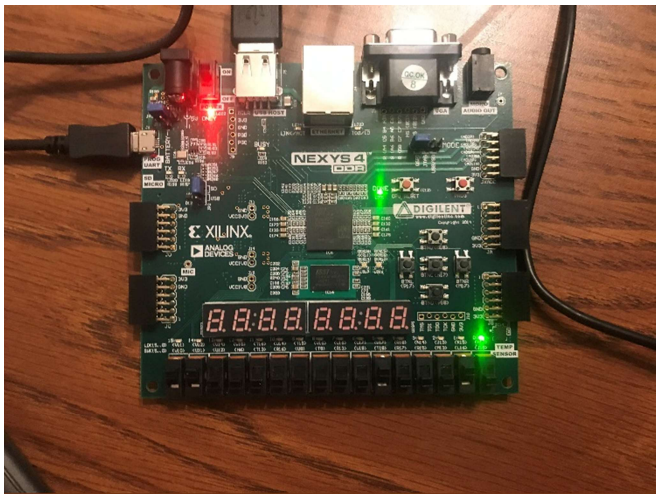

**Figure 5: LED output of the sum of 4 and -3 (4 + -3 = 1)**

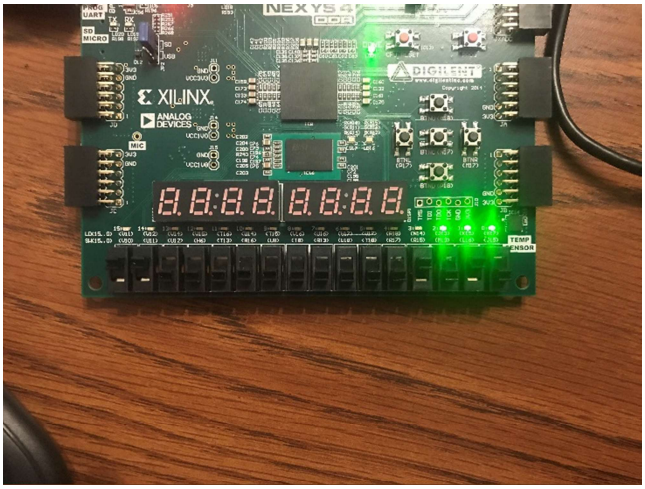

**Figure 6: LED output of the difference of 4 and -3 (4 - -3 = 7)**

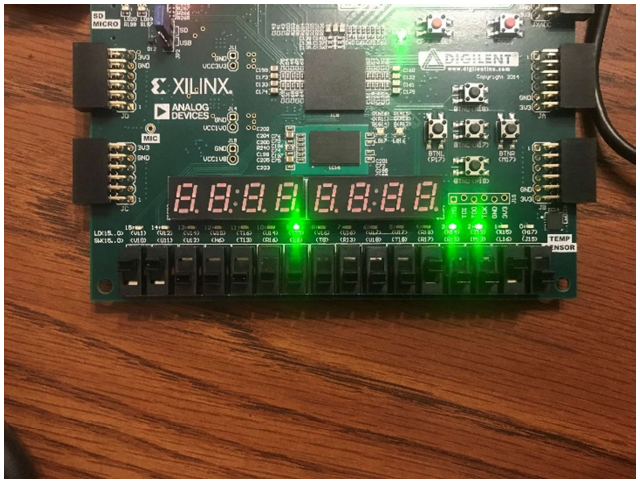

**Figure 7: LED output of the product of 4 and -3 (4 * -3 = -12)**

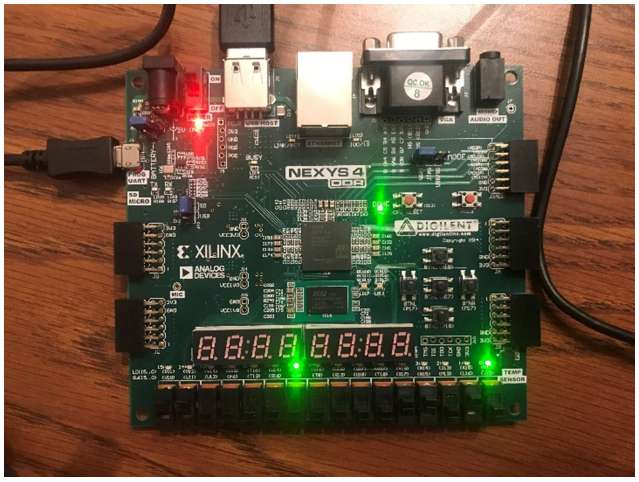

**Figure 8: LED output of the quotient of 4 and -3, rounded down (4 / -3 = -1)**

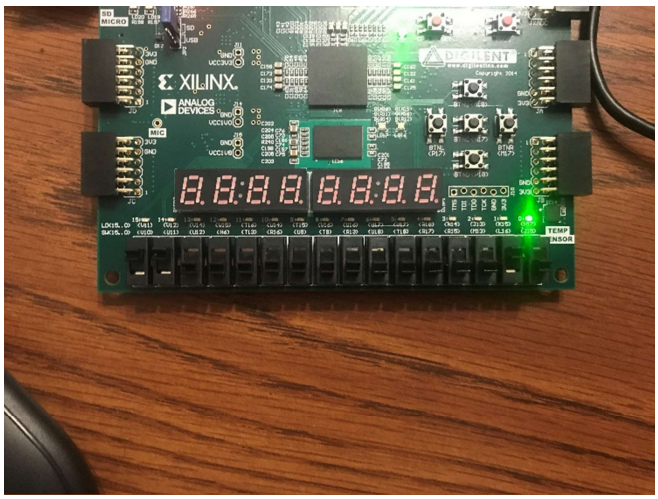


**Figure 9: LED output of the modulo of 4 and -3 (4 mod -3 = 4 % -3 = 1)**

## V. Conclusions

The main take away our group had was a true understanding of the complexities of a digital logic system and the usefulness of things like finite state machines. We also grappled with the differences between VHDL and other programming languages. Practices that would work in things like C++ or Python would not work in VHDL, bringing with it a whole collection of considerations and problems that normally would not be considered. These considerations could account for much of our development time over the course of the project.

Some of the problems still needing to get solved involve the input method. The project was designed with the keyboard in mind, trying to use as little of the inputs from the board as possible, especially the switches. It would lead to greater usability to push as many of the inputs to the keyboard as possible, but the structure of the program used to get the results we did does not lend itself to moving things like negating numbers and selecting registers. Moving in this direction would indeed be an improvement.

In the future we would like to develop binary to BCD converters and display the information on the seven segment displays so it is easier for the user to read and understand. Also, in the future it might be good to add more bits to be able to do math on larger numbers. If we had more time, we would also like to resolve an issue we found that certain numbers would result in zero even though that was not the correct answer. We think this is cause by the two's compliment to sign and magnitude converter, but more investigation would need to be done to be sure. But in its present state, our program functions more than acceptably as a signed calculator.

## References

[1] Llamocca, Daniel. "PS/2 Keyboard Controller." VHDL Coding for FPGAs. http://www.secs.oakland.edu/~llamocca/Tutorials/VHDLFPGA/Vivado/Unit_7/my_ps2keyboard.zip

[2] Llamocca, Daniel. "Notes – Unit 7." Introductions to Digital Systems Design, Nov. 2018, pp. 5-9. https://moodle.oakland.edu/pluginfile.php/4752379/mod_resource/content/8/Notes%20-%20Unit%207.pdf.

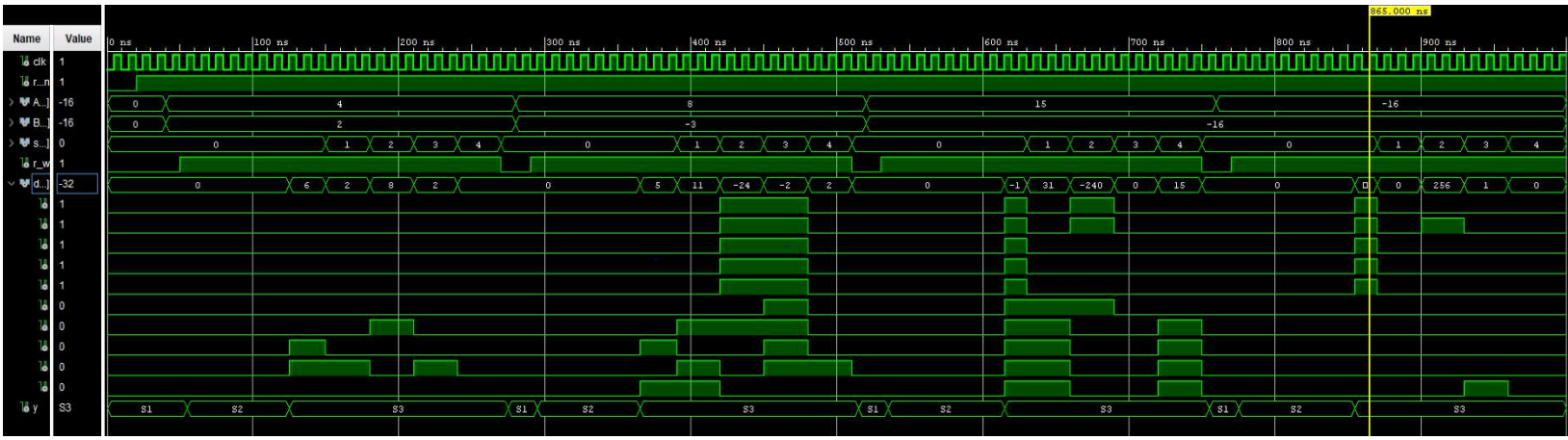


**Figure 10: Test Bench output of main calculator circuit used in testing**

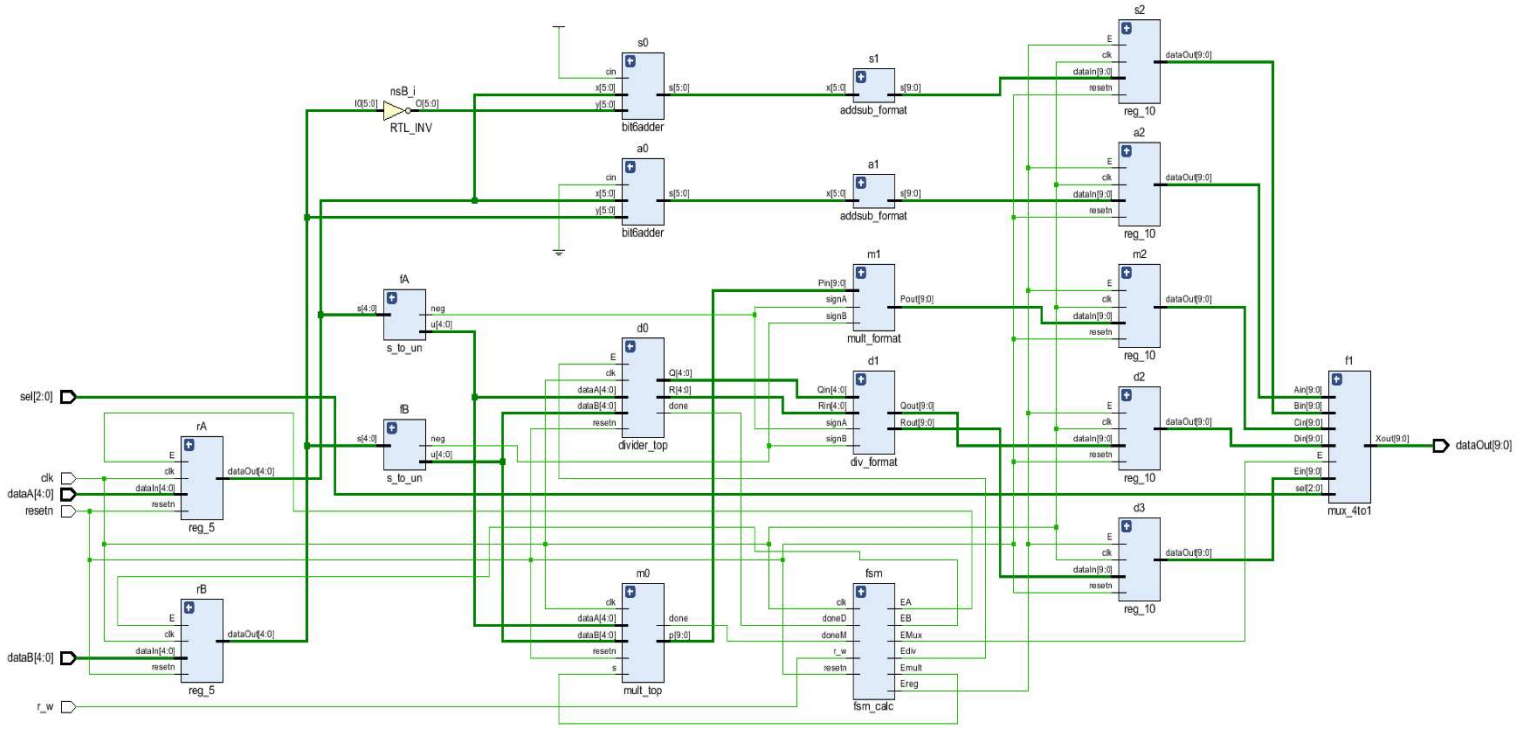


**Figure 11: Calculator Block Diagram**