Simple Calculator

By:
Matt Stopyak
Matt Wagner
Benley Mathew

Roles

Matt S: 8-bit Adder and Subtractor.

Benley M: 8-bit Multiplier and Divider.

Matt W: State Machine and Integration.

INTRODUCTION

What was Made? A simple calculator. Using the Nexys 4 FPGA board from class and a 4x4 PMOD keypad from Digilent.

How was it made?

1. The main math operations, addition, subtraction, multiplication and division, were created in VHDL and used to add, subtract, multiply and divide simple binary unsigned numbers.

2. Values would then be converted from BCD numbers into Binary numbers and stored in different registers once inputted in the system after the “equal” button is pressed.

What we learned?

1. To clear variables in the sensitivity of the process

2. How to converter BCD to Binary and Binary to BCD. This method is called Double-Dabble.

Uses for this project.

- Used for any person who needs to use a calculator for simple math, such as elementary school students who are just starting to learn basic mathematics.

Methodology (Inputs and Outputs)

- Eight-Bit Unsigned Numbers [0, 256]
  - Changed to two digit values only, [0, 99]

- Numbers are converted from BCD to Binary.

- After the numbers go through the operations, the numbers are converted back to BCD numbers and outputted to the Seven-Segment display.

- For division, the Seven-Segment display with have a remainder.

- Additionally, for the Seven-Segment display to work with the multiple digits, we had to create a multiplexer that would select each digit one at a time and display the correct value.

Methodology (Operations)

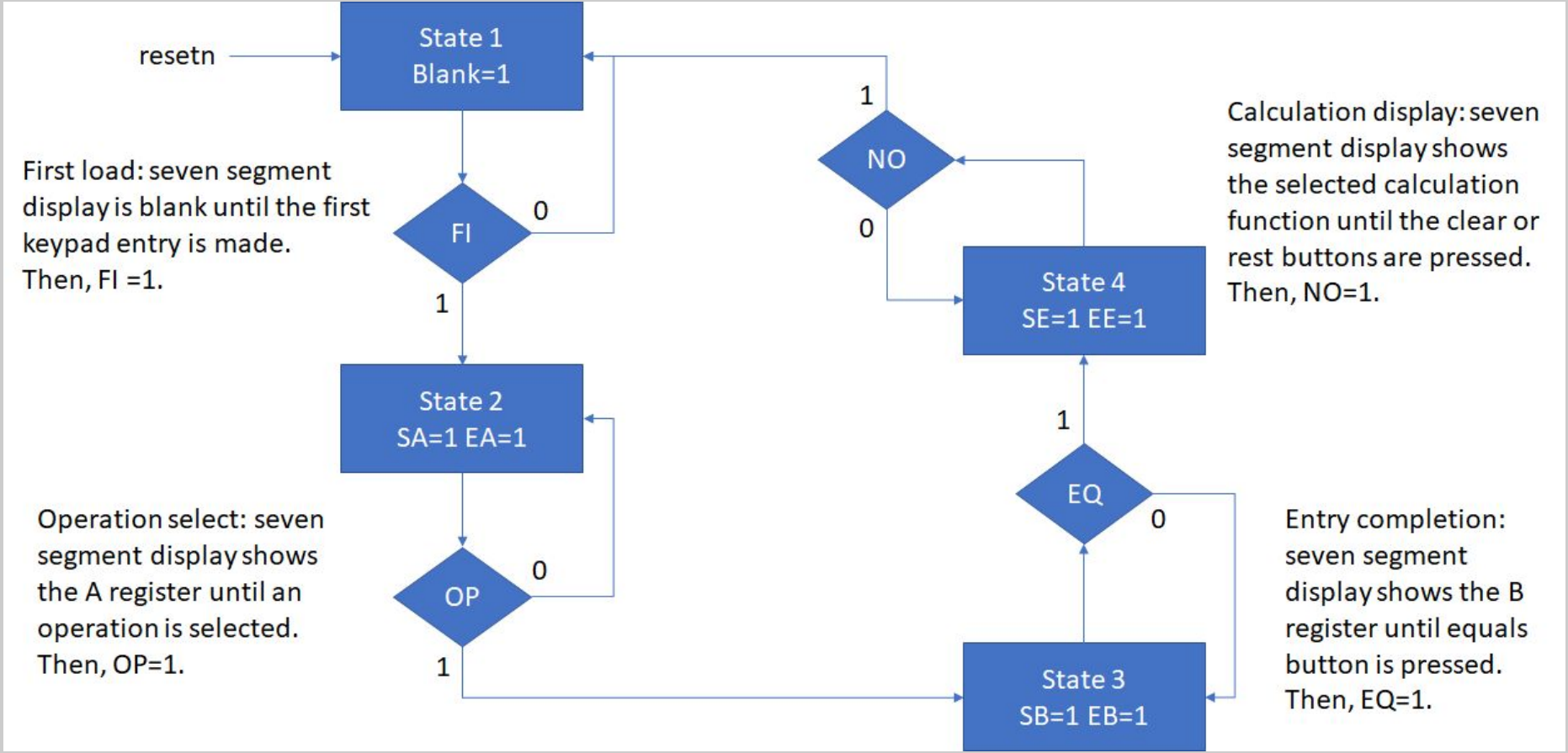
- Addition
  - Created from Adder, made in the lab.
  - Modified to add 8-bit numbers together.

- Subtraction
  - Also created from Adder, made in the lab.
  - Changed the addsub value to ‘1’ to do the subtraction
  - Modified to subtract 8-bit numbers together.

- Multiplication
  - Created from Multiplier, made in the lab.
  - Modified to multiply 8-bit numbers together.

- Division
  - Created from Divider, made in the lab.
  - Modified to divide 8-bit numbers together.

State Machine Diagram



Experimental Setup

- Supplies needed for project.
    - PmodKYPD: keypad from the Diligent
    - NEXYS 4 DDR Board: Vivado Board

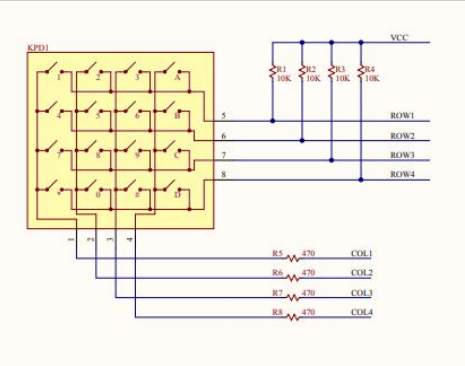
- Keypad Keys
    - Buttons 0-9 are numeric values 0-9
    - Button A is the Addition Key
    - Button B is the Subtraction Key
    - Button C is the Multiplication Key
    - Button D is the Division Key
    - Button E is the Equal Key
    - Button F is the Clear Key

- Vivado Board:
    - Reset button just resets the program
    - LED 15: State 1
    - LED 14: State 2
    - LED 13: State 3
    - LED 12: State 4
    - LED 11: Enable Register A
    - LED 10: Enable Register B
    - LED 9: Enable Register E
    - LED 17 (Blue): Key Press
    - LED 0: Carry Out
    - LED 1: Overflow
    - Seven-Segment:
        - Digit 8: “=” sign
        - Digit 5 and 6: Number Display
        - Digit 4: “r” for remainder
        - Digit 3: remainder value

Experimental Setup Continued...

- The keypad consists of 4 columns and 4 rows of interconnected buttons.
- Following the internal clock of the FPGA board, each column would have its pin dropped to a ‘zero’ and if any rows dropped to ‘zero’ in that same clock cycle the depressed keypad could be inferred from these coordinates.
- Once the depressed key was decoded it could be latched into the circuit for implementation in the final design.

```
decode: process(clk)
begin
    if clk'event and clk = '1' then
            if sclk = "00000000000000000000" then
                DC <= "11111";
                sclk <= sclk+1;
            end if;
            -- 1ms
            if sclk = "00011000011010100000" then
                --C1
                Col<= "0111";
                sclk <= sclk+1;
            -- check row pins
            elsif sclk = "00011000011010101000" then
                --R1
                if Row = "0111" then
                    DC1 <= "00001"; --1
                --R2
                elsif Row = "1011" then
                    DC1 <= "00100"; --4
                --R3
                elsif Row = "1101" then
                    DC1 <= "00111"; --7
                --R4
                elsif Row = "1110" then
                    DC1 <= "00000"; --0
                elsif Row = "1111" then
                    DC1 <= "11111"; --blank
                end if;
                sclk <= sclk+1;
            -- 2ms
             elsif sclk = "00110000110101000000" then
                --C2
                Col<= "1011";
                sclk <= sclk+1;
```



Results

Demo of Project Results Link:

https://photos.app.goo.gl/EvsWD4Uwr1oxDwWv5

Operation	Data A	Data B	Solution
Addition (+)	12	36	48
	3	27	30
	99	99	98 (Carry Out)
Subtraction (-)	48	12	36
	98	8	90
	54	52	2
Multiplication (*)	3	6	18
	20	4	80
	9	9	81
Division (/)	29	10	2 (r9)
	48	12	4
	56	5	11 (r1)

Conclusion

- Issues
  - Integration of keypad.
  - The decoding and button debouncing of the keypad.
  - The shift nature of the BCD output.

- Improvements
  - Using 2’s complement binary arithmetic.
  - Drop the leading zeroes on the display.
  - Institute an “all clear” and clear function.
  - Expand the arithmetic capabilities to handle a greater number of bits.
  - Rework the FSM and integration of coding so it is easier to read.
  - Replace the seven segment display with LCD display for greater functionality.

- Overall
  - Harder than anticipated.
  - When you forget that it is hardware descriptive and treat it is a coding language, the problems you will encounter with your code will intensify.
  - A good learning experience for beginners.