Bilkent University
CS223 Laboratory Project
Calculator

Section 03

Olcay Akman

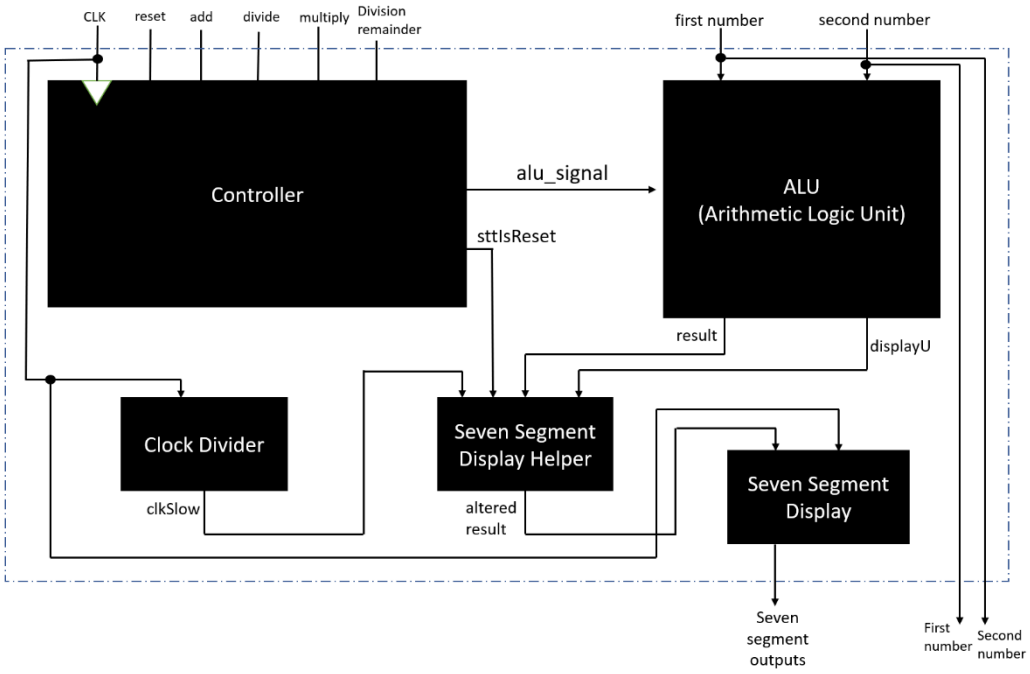
ID: 21702671

13.05.2019

1. Introduction

The aim of this project is to design a calculator using the FPGA. By using the switches on the BASYS3, the calculator takes the inputs and via the buttons it takes the operation commands. There are five buttons on the FPGA and each of them are assigned a different command: one does addition, one does multiplication, one does division, one gives the division remainder and one resets the display. All results are displayed on the Seven Segment four-digit display. The inputs are taken in signed binary number form (two's complement form) and the result is displayed in hexadecimal notation. The result blinks on the Seven Segment display.

2. Block Diagram



Above is the block diagram of the HLSM (High Level State Machine) that I have designed for my calculator design. In it I have blocks that serve different purposes, they are as follows:

- A controller.
- An Arithmetic Logic Unit
- A clock divider
- A helper part for the Seven Segment Display
- The Seven Segment Display module

Each of those blocks feed one another to create a fully functioning and complete calculator. In the next section of this report I have provided detailed explanations for each black box in this design as well as their relations with each other to finally contribute to the full calculator.

3. Detailed Explanation of the Work

The entire design contributes to a fully functioning calculator to which each input is given by using the switches on the FPGA board. For each black box given in the schema in section 2, I will now provide detailed explanation below.

3.1 Arithmetic Logic Unit (ALU)

The arithmetic logic unit is the part where the calculator does the simple math operations. These are as follows:

- Addition – covers subtraction
- Multiplication
- Division
- Modulus (division remainder)

Into the calculator, each number, namely first number and second number, is taken in binary form. That is, as switched are used to get the inputs, each input is fed into the calculator by using eight switches, where each switch is a bit. Both of those inputs are taken in signed (two's complement) form. Thus, the calculator can be fed negative numbers, and this is how subtraction is covered by the addition operation. If the user wishes to subtract a number B from a number A, that is, they wish to calculate "A – B", they will have to feed the two's complement form of B in its 8-bit binary equivalent. Similarly, any result, whether negative or positive, is displayed on the seven-segment display. If the result is negative, a minus sign appears on the most significant digit of the seven-segment display. The seven-segment display shows the result in its hexadecimal equivalent value. The decision whether to display a minus sign on the display is determined in another box, in the "Seven-Segment Display Helper" part, which is explained in section 3.4 of this report.

The arithmetic logic unit takes three inputs, gives two outputs. The first two inputs are the two numbers on which the calculations will be made, and the third is the "alu_signal", which is fed to the ALU from the Controller (explained in section 3.2). One output is the result itself, and the other is a signal called "displayU", which is used in Seven-Segment Display Helper (refer to section 3.4 for more information) part to determine whether the display show "_ _ _ U" or not. The display shows this message only if the result is undefined – this happens when the user tries to divide a number by zero. Other than this, inside the ALU module, according to the specific value of the alu_signal, the module determines which operation it is that needs to be done. It then does this specific operation, assigns the second output, named "result" to this result obtained. In my code, the following signals are used to carry out the corresponding operations:

- 3'b000 – reset
- 3'b001 – addition
- 3'b010 – multiplication
- 3'b011 – division
- 3'b100 – modulus (division remainder)

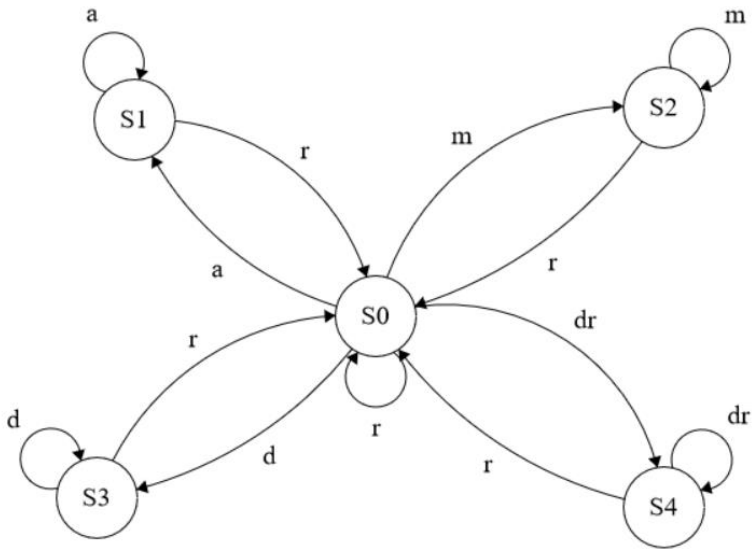
When the outputs of this logic unit are obtained, they are later used in the Seven-Segment Display Helper part to carry out the remaining internal operations needed to display the result properly.

3.2 Controller

The controller is the part where the required operation to be carried out is determined, which is kept as "alu_signal", and then fed into the ALU. This part this designed entirely as a Finite State Machine (FSM), thus has the following components:

- Next state logic
- State register
- Output logic

Like the design of any FSM, the machine has certain states and their transitions are given in the following diagram:



In this state transition diagram, each state is defined as follows:

- S0: The reset state. In this state, the seven-segment display is set to display 0000, which is the default case when the calculator has not yet done any calculations, and the alu_signal is set to 3'b000
- S1: The addition state. In this state, the output logic sets the alu_signal to 3'b001, which is the addition signal in the ALU
- S2: The multiplication state. In this state, the output logic sets the alu_signal is set to 3'b010, which is the multiplication signal in the ALU
- S3: The division state. In this state, the output logic sets the alu_signal to 3'b011, which is the division signal in the ALU
- S4: The modulus (division remainder) state. In this state the output logic sets the alu_signal to 3'b100, which is the modulus signal in the ALU

The state transitions are handled in the next state logic component of the FSM. When in the reset state, the machine checks for the button inputs. If any button is pressed, the next state is set to that specific state, otherwise it stays in the reset state. When in any state other than the reset state, the machine waits for the user to press the reset button again, otherwise it stays in the current state it is in.

The controller, in total, has six inputs and two outputs. The first output of the FSM is the alu_signal, which is determined in the output logic component of the FSM. The second output is a signal, named sttIsReset, which is used to determine whether the current state of the FSM is the reset state or not. This flag is significant because we wish for the display to show a blinking output only if it is displaying a result of a given calculation operation, and not blink but plainly display "0000" on the four digit seven-segment display if it is not displaying any particular result of a given calculation command. The purpose of this signal is further explained in section 3.4. One of the inputs is the system clock (clk), and the other five are the distinct buttons for each operation that the calculator can do: reset, addition, multiplication, division, modulus (division remainder).

3.3 Clock Divider

The clock divider module in this design is used to slow down the clock that is provided by the FPGA board. This is a necessary step because the calculator shows the result blinking. In order to implement this clock divider module, I have taken the module I have used for lab 4 this semester, modified it to get the calculator to function properly.

3.4 Seven-Segment Display Helper

This component of the entire calculator is solely designated to alter the result obtained from the ALU to finally send the seven-segment display module the correct values to display the result properly in the seven-segment display of the BASYS3. The facts that this component needs to check are as follows:

- Whether the result is negative
- If it is negative, whether it has overflow (that is, it does not fit in 3 digits as its hexadecimal equivalent)
- If the result is undefined (that is, if the second operand is zero and division is performed)

As it is necessary to display a minus sign (a dash) on the most significant digit of the display, we need to check whether the result is negative or not. If the result is positive, all four possible digits can be used to display the result of the operation. However, if the result is negative, since the most significant digit is used to display the minus, we are only left with three possible digits that can be used to display the result of the calculation in hexadecimal. Thus, as inputs are 8-bits each, it is likely that the result is too big to fit in three digits of its hexadecimal equivalent. If that is the case, the machine only displays the most significant three digits of the hexadecimal equivalent of the result in the display. Otherwise, if the negative result can be displayed using three digits or less, the result is displayed normally. Also, as it was determined and saved in a flag called “displayU” by the ALU (refer to section 3.1), and as this part checks the value of the displayU flag, the Seven-Segment Display Helper component determines whether the sign “_ _ _ U” is displayed on the display or not by providing the necessary inputs to the Seven-Segment Module. It is this part of the code that checks these conditions to finally determine the proper inputs to be sent to the Seven-Segment Display module of the calculator.

Another flag that is fed into this component of the calculator is called “sttIsReset”. It is used to determine whether the controller’s FSM is at any given moment in the reset state or not. It is desired that the seven-segment display blinks only when it is displaying the result of a given calculation operation. If it is not displaying a result, but rather displaying the default value “0000” on the seven-segment four-digit display, as is the case in the reset state – hence the flag, it will need to blink. Therefore, such a flag is needed to get the display to blink at the required moments.

This component also takes the output of the clock divider module, which is the slowed down version of the system clock called “clkSlow”. This slowed down clock is later used to make the result blink on the display.

Once the “altered result” is obtained from this component of the calculator, that is it has been set to display the proper result on the display, it is sent to the Seven-Segment Display module.

3.5 Seven-Segment Display

The Seven-Segment Display module takes the necessary inputs that are fed into it by the Seven-Segment Display Helper (refer to section 3.4) and then gives the necessary outputs which will later be connected to the proper ports via the constraint file to eventually display the result of the calculation on the FPGA’s four digit seven segment display. I have taken the SevSeg_4digit module that I have

previously used in lab 5 of this semester, modified it and made the appropriate changes to it to eventually make it fit to display the desired results for this calculator.

3.6 A Broad Overview

All the components given in this design, as explained above, feed each other the necessary signals to eventually create a fully functioning calculator in which all components work in alignment. Additionally, although it was not a specific member of any component explained above, there was another action performed by the top module (the calculator module) of the design, which contains all the components given above. It is to assign the corresponding LEDs above each switch to further enhance the usage of the calculator. The reason to do this is to comfortably see the 8-bit representation of both the numbers given to the calculator as inputs via the switches.

4. Conclusion

The calculator designed and implemented during the project is a simple one: it takes two numbers, it can only perform 4 operations (5 if we include subtraction too, which is covered by the addition key, further explained in section 3.1) and it displays the result in hexadecimal form. Each component of this calculator design work together to create a fully functioning calculator, as explained in this report.

5. References

While creating this project, I have written my own SystemVerilog code. However, there are only two modules of my entire project code that I have not written from scratch: these are the ClockDivider and SevSeg_4digit modules. I have taken the SystemVerilog codes for these two modules from the lab 4 and lab 5 assignments, respectively, that I have submitted to Unilica this semester. I had to modify them slightly to fit the needs of this calculator to make it function properly, as the project description requires.