CS-223

Calculator Project

Section 03

Oğuz Kaan İmamoğlu

21702233

13.05.2019

2. Introduction

In this project, a simple calculator must be designed with FSM/HLSM logic. This calculator takes two 8 bit inputs and makes four simple operations with these inputs. It also does remainder operation. The output must be showed with seven segment display and also the output must blink with 500 ms period.

User enters two 8 bit inputs with switches of FPGA and he/she chooses an operation with push buttons on FPGA. Calculator makes the operation that be chose and display it on seven segment display until user push the reset button. After resetting user can enter new inputs also he/she can choose new operations.

Calculator includes negative numbers.

Subtraction operation will be represented as addition with a negative number.

Inputs will be represented as Two's Complement Numbers and output will be displayed as hexadecimal numbers.

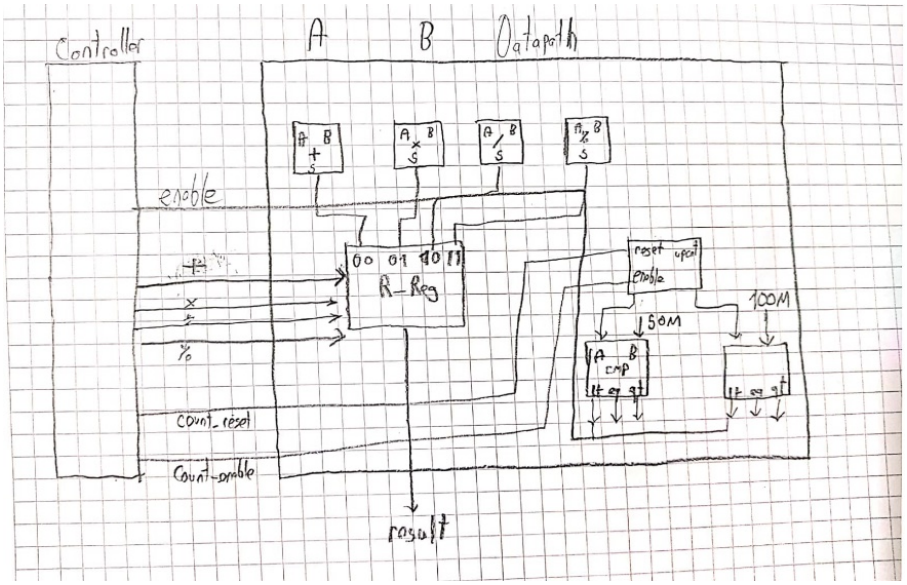
In my project there is only one missing requirement:

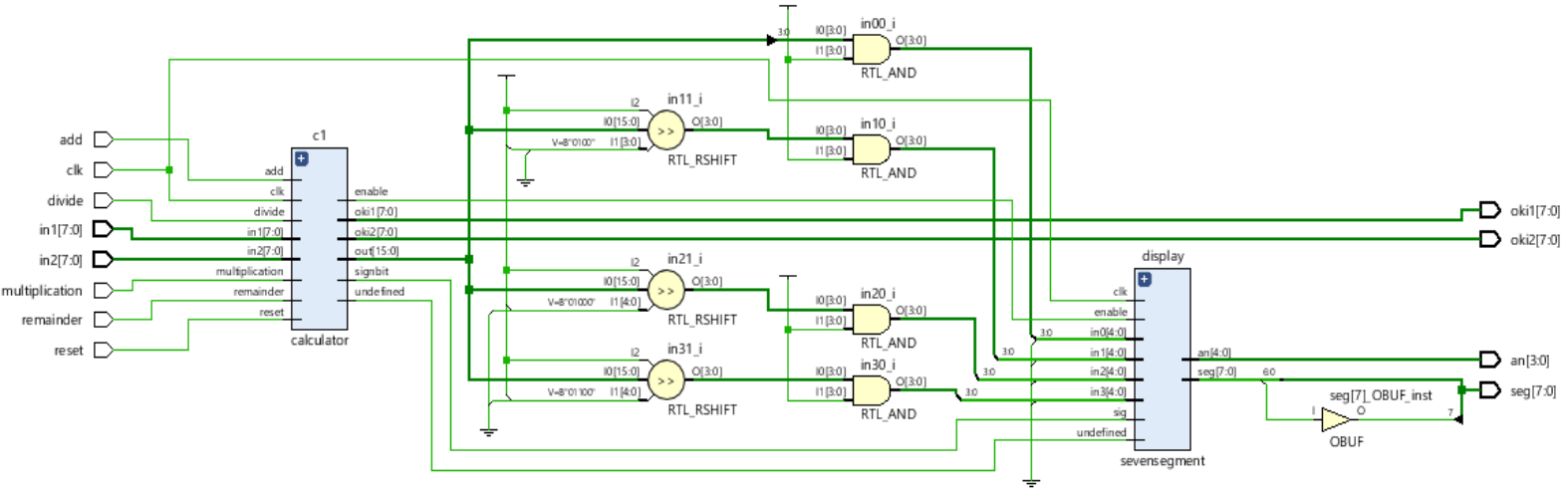
When the output size exceed 4 nibbles, for example when the output is –69CE ideally it must show -69C (by ignoring least significant nibble) . But in my project it will be shown as -9CE. It is a minor problem in my opinion.

All other features like including showing '___U', showing minus sign, showing values correctly, resetting inputs, showing blinking output are working without any problem.

3. Block Diagram

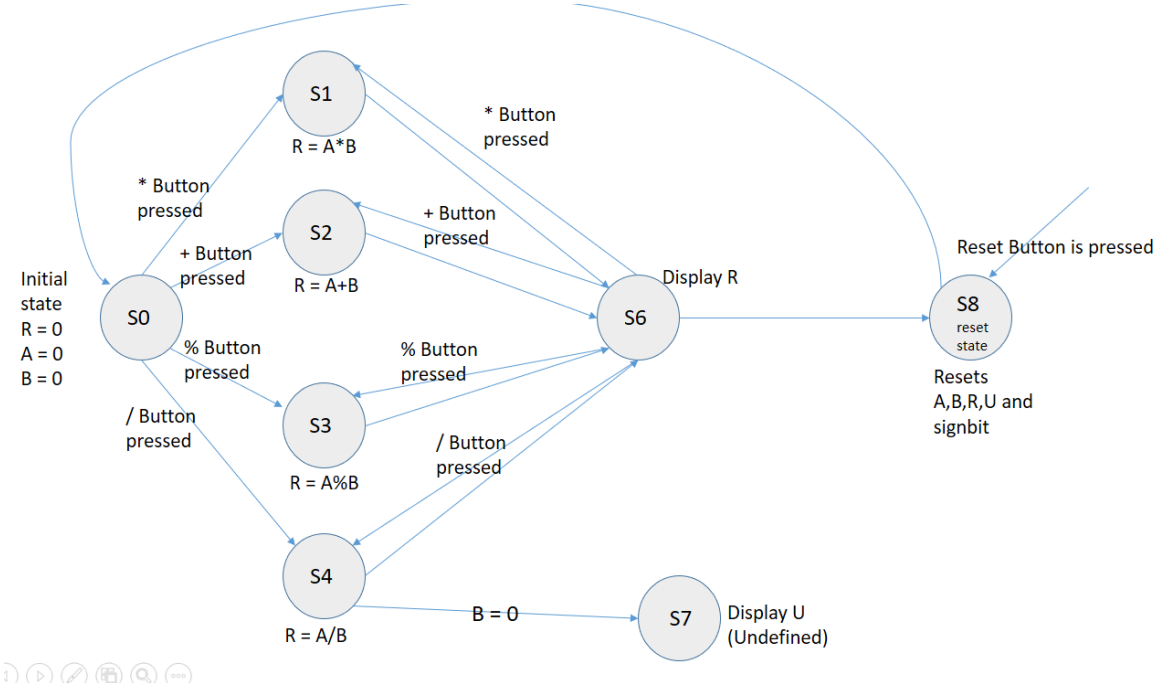
Block diagram (datapath) of the project will be given below. First one is my initial design and I made some changes in this design while I coding it in Vivado. Second design is created by Vivado's itself. It is more complicated. These will be explained more detailed in part 4 thanks to HLSM and FSM's of the project.





4. Detailed Explanation of the Work

HLSM of this project is be given below.



State 0 is the first state of HLSM. It initializes all inputs and outputs with 0. There are three extra variables. One is ‘undefined’, one is ‘sign bit’ and last one is ‘enable’. They will be explained in other states.

- If * button is pressed S1 will be next state. S1 does A*B operation.
- If + button is pressed S2 will be next state. S2 does A+B operation.
- If % button is pressed S3 will be next state. S3 does A%B operation.
- If / button is pressed S4 will be next state. S4 does A/B operation.

Except for S4, all these states will precede to S6 automatically. S4 precedes to S7 if B = 0 in order to show ‘___U’ which represents undefined. Otherwise it continues to S6 to show result on seven segment.

On any stage, if reset button is pressed, state will be S8 which is reset state. It resets all variables and precedes to S0. It means unless reset button is pressed user cannot update inputs. To show this, I assign inputs to leds. So, even user changes switches, leds (the real inputs) will not change until user press reset button. User can only update operations without pressing reset button.

Let us continue with displaying states. For displaying outputs there is another FSM inside the S6 and S7. These states are implemented in sevensegment module. I modified sevensegment module which is provided by instructors, in Unilica. These 3 variables I mentioned before for this purpose.

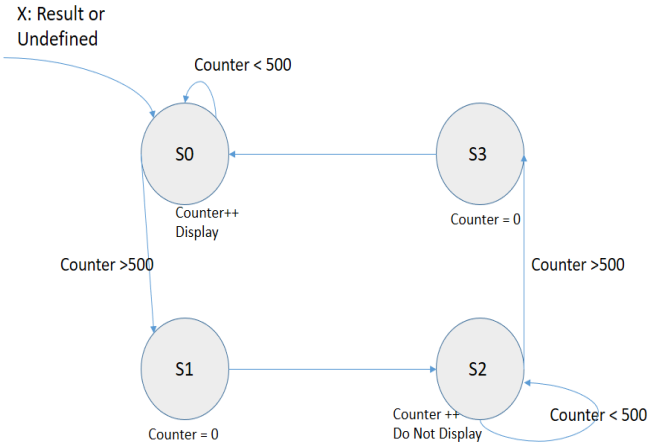
- Undefined variable becomes 1 if user press / button and denominator is 0. It will be send to sevensegment module and if it is 1 ‘ ___U’ will be displayed in seven segment.
- Other variable is enable, it is controlled by a clock divider. It changes 0 to 1 and 1 to 0 with 500 ms period and it will be send to sevensegment module. When it is 1, result will be displayed and when it is 0 nothing will be displayed. This causes blink. Result will blink with 500 ms period in seven segment display.
- The last variable is signbit. It is 1 if first bit of output is 1 and 0 if first bit of output is 0. Since output is signed variable it is represented as Two’s Complement. So first bit reflects it’s sign. The signbit variable will be send to sevensegment module and if it is 1 minus sign will be displayed in seven segment display. If signbit = 0 result will be displayed normally.

This is the FSM inside the S6 and S7. This is designed for displaying blinking output.

Also some other FSM’s can be designed for handling the undefined case and handling the minus sign case, but I only put this FSM on this report.

I must also mention my top module. In my top module I provoke a calculator module with user inputs. After that I used output of calculator module. I convert the result from decimal to hexadecimal by using mod 16. Then I provoke sevensegment module in order to display the output.

I think I mentioned all important details of my project.



5. Conclusion

This project contributes me lots of things. It was quite annoying process for me because in some cases I challenged with some problems. Some problems I challenged with and somethings this project contributes me:

- Designing datapath: I can design FLSM's and HSM's without significant hardships however; when I was trying derive a datapath from state machines, it was quite challenging for me. I am still not sure about my datapath whether it is correct or not.
- When I tried to add some features to my project, several times it causes some important problems in already working functions. In other words, fixing something leads to spoiling other things.
- This project taught me to be patient. Because generating bitstream was very time consuming and I did it many times. Waiting for bitstream is quite annoying.
- Another challenging situation is: Coding my designs on Vivado. My Verilog skills are not so well before this project but now, I can say that I am better at Verilog. However, there is still a long way before me.

As I mentioned in introduction there is only one malfunction in my project. If output exceeds 4 nibbles it does not shift the output to right. It ignores most significant nibble and display minus instead of it.

6. References

Seven segment code provided in Unilica.
Constraint file given in Unilica.

7. Appendix

https://www.unilica.com/index.php?v=course&s=home&cid=203&mv=11&aid=28392
https://www.unilica.com/index.php?v=course&s=home&cid=203&mv=11&aid=28387