- Alexandra Zajda (260807517)
- Nicolas Barreyro (260730549)
- Gorkem Yalcinoz (260710053)
- Group 12

ECSE-222 Digital Logic
LAB REPORT #4
Lab-4 Basic calculator

- Alexandra Zajda (260807517)
- Nicolas Barreyro (260730549)
- Gorkem Yalcinoz (260710053)
- Group 12

- Alexandra Zajda (260807517)
- Nicolas Barreyro (260730549)
- Gorkem Yalcinoz (260710053)
- Group 12

INTRODUCTION

The goal of this lab was to design a basic calculator able to perform addition/substraction on two 8-bit numbers. In addition to previous labs' register and adder/substractor implementations that the present implementation builds upon, the latter contains a state machine controller that manages the system operations using four states. The calculator has inputs and outputs that can be mapped to buttons, switches and LED's on the FPGA board to be tested. The circuit contains a *clock* and *reset* lines.

DESIGN (I/O)

The circuit has as **inputs** :

- *clock* input bit (state transitions are only processed during rising edge).
- *reset* input bit (resets state to *s0*).
- *tr1* and *tr2* input bits (allow transition of state from *s0* to *s1* and *s1* to *s2* respectively).
- *operator* input bit (allows transition of state from *s2* to *s3*).
- 2-bit *operation* vector("00" to add inputs, "01" to do 1st input - 2nd input, "10" to do 2nd input - 1st input, "11" to prompt the system to reset).
- 8-bit *in_1* vector which forwards physical switch inputs to internal register.

The circuit has as **outputs**:

- 9-bit *out_1* vector (containing cout/bout as MSB and sum/diff of inputs as the rightmost bits) which is mapped to the 9 LEDs on the board.

The circuit makes use of a 32-bit internal signal that represents 4 8-bit registers. The registers' indexation starts from rightmost bit. These registers are used as follows:

- R0 : stores an 8-bit number, inputted by the user (*in_1*) during state *s1*.
- R1 : stores an 8-bit number, inputted by the user (*in_1*) during state *s2*.
- R2 : stores the result of an addition/subtraction during state *s3*.
- R3 : LSB of R3 stores *cout/bout* during state *s3.*

A mapping of the inputs/outputs (except clock line) to switches/buttons/LEDs the Altera board can be seen in Figure 1.

- Alexandra Zajda (260807517)
- Nicolas Barreyro (260730549)
- Gorkem Yalcinoz (260710053)
- Group 12

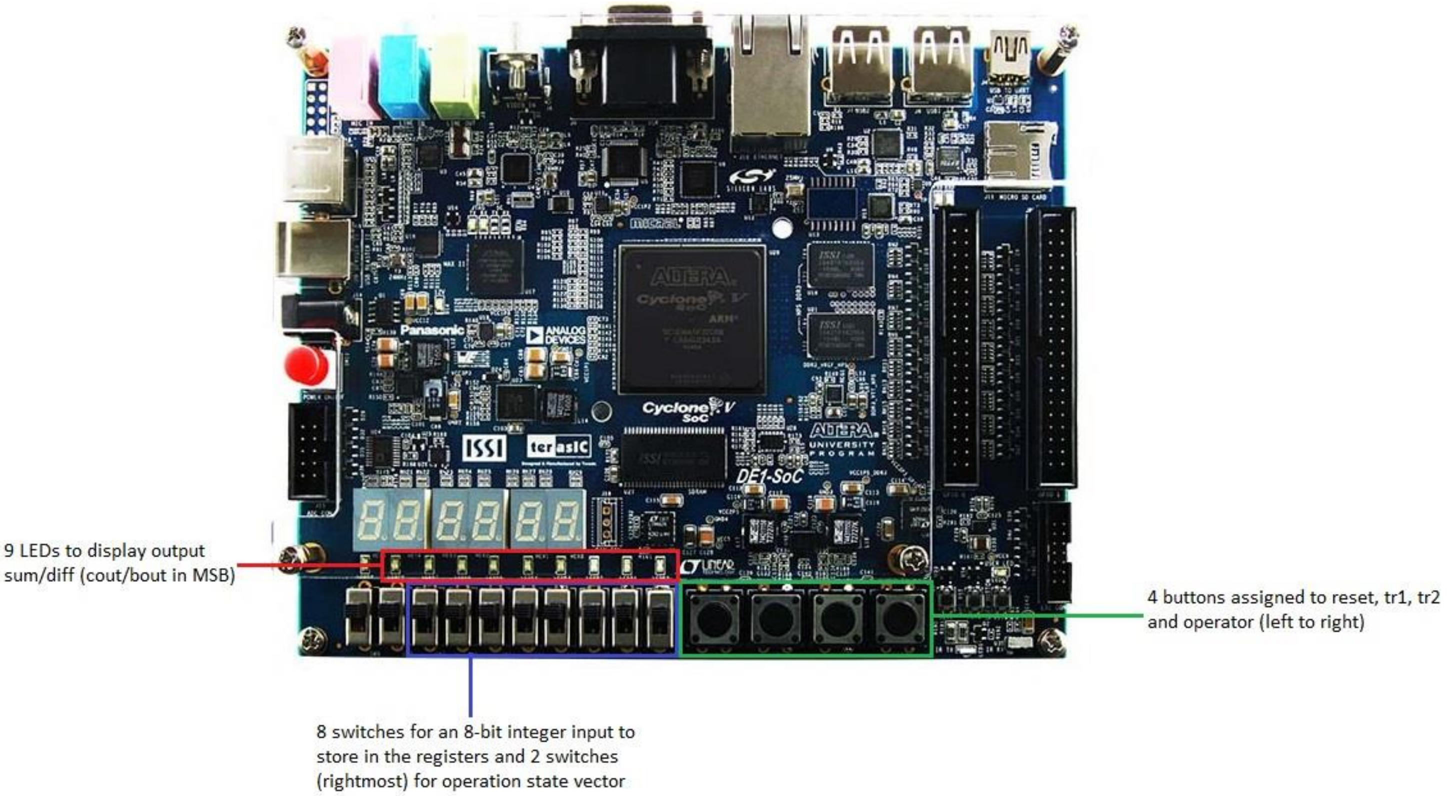

9 LEDs to display output sum/diff (cout/bout in MSB)
4 buttons assigned to reset, tr1, tr2 and operator (left to right)
8 switches for an 8-bit integer input to store in the registers and 2 switches (rightmost) for operation state vector

FIGURE 1 : Location of User-Input Pins/Buttons/Switches on the Altera Board

The circuit makes use of 3 components that were introduced in previous labs : registers, the 8-bit subtractor, and the 8-bit adder. The components are mapped to internal signals and perform addition/subtraction on the inputs stored in the registers R0/R1.

- Alexandra Zajda (260807517)
- Nicolas Barreyro (260730549)
- Gorkem Yalcinoz (260710053)
- Group 12

FUNCTIONALITY

The specific circuit uses user input to perform its various functions. This input is physically provided via an Altera Dec-Soc educational board. With the exception of the *clock* input, which was mapped to a predetermined 50kHz frequency clocking signal pin, all 4 other input bits (*reset*, *tr1*, *tr2*, *operation*) were user-inputs mapped to the 4 active-low buttons of the board. Thus, if the switches were pressed, the value of their signal was "0".

The circuit's operations are managed by a state machine controller. The controller makes use of 4 states which are *s0*, *s1*, *s2* and *s3*. The first state (*s0*) makes sure that the output is set to 0, thus the LEDs will be turned off. The second (*s1*) state allows the user to store a 8-bit integer in R0 using the board's switches shown in Figure 1 and the number is redirected to the output, so that the LEDs display its value. The third (*s2*) state allows the user to store a 8-bit integer in R1 and the number is also redirected to the output. The fourth state (*s3*) allows the user to input a 2-bit vector representing the operation desired. If the operation is "00", the inputs will be added and the output will display the sum and carry-out. If the operation is "01", the difference and borrow-owt of (R0 - R1) will be redirected to the output. If the operation is "10", the difference and borrow-out of (R1 - R0) will be instead be redirected to the output. Finally, if the operation is "11", the system will be prompted to reset (similar as activating the reset button).

Furthermore, state transitions happen only during a clock rising edge. If the system is at state *s0* and the button mapped to *tr1* is activated, then the system will transition to state *s1.* Similarly, it will transition from *s1* to *s2* if *tr2* is activated. At state *s2*, if the transition bit *operation* is activated, it will transition to *s3* and proceed to compute sums and differences.

The circuit is capable of showing negative numbers using two's complement notation.

SIMULATION PROCEDURE EXPLANATION

To test if the circuit was adequate, we first ran a timing waveform simulation of the circuit using Altera Quartus II software. Considering the amount of input variations and the fact that output was depending on timing (due to the clock signal) the waveform was used to evaluate specific carefully timed operations to see if the outputs *data_out* and *bit_out* were correct.

Atop of that, these same operations were reproduced on the Altera board once it was synched to the circuit.

Finally, a SignalTap II simulation was run, to evaluate real-time time delays that happened when our circuit was run on that specific piece of hardware. This was done to contrast the waveform simulation, which was a simulation that was not hardware-specific.

ANALYSIS

operation = “00” (ADDITION)

To analyze the efficiency of the circuit, given the massive amount of possible inputs, we decided to settle on a specific set of inputs. For the sake of simplicity, we will analyze a specific example where our circuit adds the numbers 3 and 4 stored in R0 and R1 respectively. The output is seen in the *out_1* vector.

As can be seen in Figure 3, at the first state *s0,* nothing is assigned to the output. At state *s1,* after the system transitioned since *tr1* is activated*,* the number “0000 0011” = 3 is assigned to R0. Later the number “0000 0100” = 4 is assigned to R1 since the system transitioned to state *s2*. Finally at the third rising edge, the system transitions to state *s3* and proceeds to add both integers in the register since *operation* is set to “00” producing an output of “0000 0111” = 7.

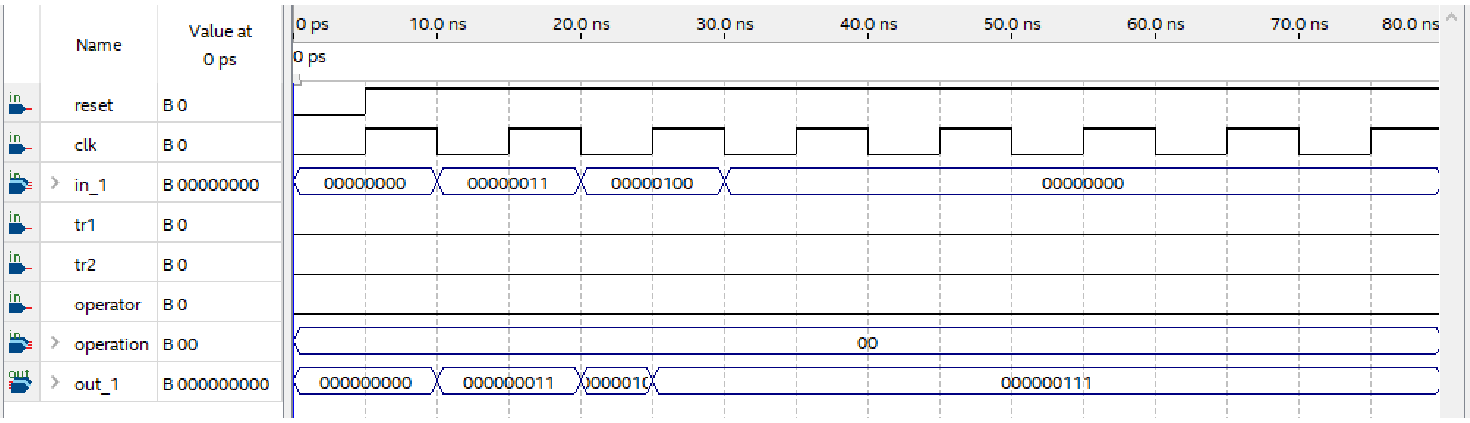
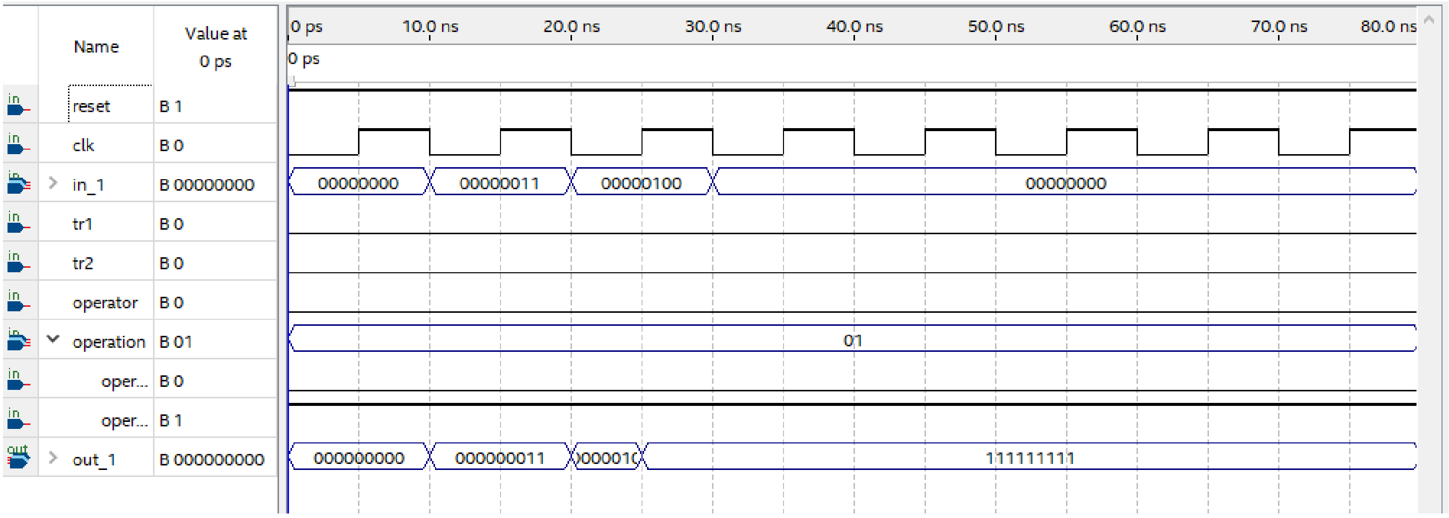


FIGURE 3 : Waveform showing Timing Simulation of circuit (R0 = 3 + R1 = 4).

operation = “01” (SUBSTRACTION)

For the sake of simplicity, we will analyze a specific example where our circuit subtracts the numbers 3 and 4 stored in R0 and R1 respectively. The output is seen in the *data_out* vector at the next clock rising edge.

As can be seen in Figure 3, at the first state *s0,* nothing is assigned to the output. At state *s1,* after the system transitioned since *tr1* is activated*,* the number “0000 0011” = 3 is assigned to R0. Later the number “0000 0100” = 4 is assigned to R1 since the system transitioned to state *s2*. Finally at the third rising edge, the system transitions to state *s3* and proceeds to substract both integers in the registers since *operation* is set to “01” producing an output of “1111 1111” = -1.



Name	Value at 0 ps
reset	B 1
clk	B 0
in_1	B 00000000
tr1	B 0
tr2	B 0
operator	B 0
operation	B 01
oper...	B 0
oper...	B 1
out_1	B 000000000

FIGURE 4 : Waveform showing Timing Simulation of circuit (R0 = 3 - R1 = 4).

DISCUSSION

Following the analysis of the waveform simulations of the circuit, we concluded that the design and implementation of the circuit was accurate. The addition mode accurately produced the result 3 + 4 = 7 and the subtraction mode produced the result 3 - 4 = -1. The design of our circuit (as seen in ANNEXE A) proved to be accurate.

Also, the state machine implementation proved to be accurate and efficient when testing on the board. It produced the right results and it was implemented the appropriate way as shown in class depending only on the present state of the system since it is a Moore state machine.

To improve the performance of the circuit, the combinational inner workings of the adder and subtractor could be optimized.

- Alexandra Zajda (260807517)
- Nicolas Barreyro (260730549)
- Gorkem Yalcinoz (260710053)
- Group 12

References

Clark, J. “Lab#4 - Basic Calculator“, ECSE 323, 2018.

- Alexandra Zajda (260807517)
- Nicolas Barreyro (260730549)
- Gorkem Yalcinoz (260710053)
- Group 12

ANNEXE A (lab4_g12 -- BASIC CALCULATOR RTL Circuit Diagram)