Serial Artihmetic Processor

Deyvi Andrade- Aviles

Department of Electrical and Computer Engineering, University of Massachusetts Lowell
Submitted: 22 December 2023

Abstract

Serial arithmetic processors are a type of processor that performs operations on data one bit at a time, rather than processing multiple bits simultaneously like parallel processors. In this particular project, the design that is being constructed has specifications that would allow the user to perform a set of operations on a 4-bit operation code — OpCode. The design consits of three main modules; a data path, state generator, and a control circuit. These three modules will work in conjunction to output a 4-bit result.

1 Design

DATAPATH Module

ALU Function Set This particular module specifications include two registers that corresponds to one 4-digit bit. These registers have control inputs **s1rA** and **s0rA** for register A, and **s1rB s0rB** for register B. These registers sport a 74194 chip that are used as shift/parallel load registers. Towards the center of the diagram is where the single 2:1 input processor multiplexer is located. The bottom right portion of the diagram features the carry portion of the full adder (which is located towards the left of item labeled **CARRY MUX**).

In addition to the hardware, the module also includes inputs which are employed in hopes of inputting the proper configuration for the machine once it is "powered on". Note that the machine is also configured with asynchronous active-low reset (**rst** as seen in the diagram) CLK is simply the clock input meant to keep track of the different stages the machine may be in during its operation. **CS** and **CSEL** represent the carry MUX selector bits and the carry enable respectively.

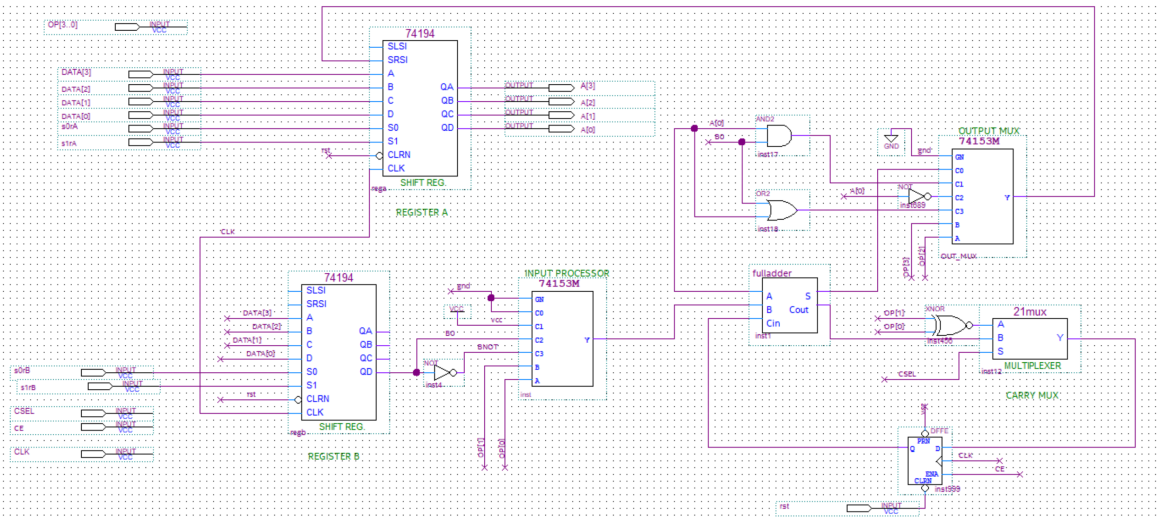

Figure 1: Block diagram schematic of the DATAPATH module in Quartus Prime. The design includes inputs that are fed into registers, muxes, input processesors, etc.

DATAPATH Table The following table was derived from the DATAPATH module.

Table 1: Truth table that was derived from the DATAPATH module. The inputs are the 4-bit OpCodes that each represent a unique operation. This oepraiton corresponds to a certain instruction which is lsited on the far right column of the table. Note that the outputs for x_i and y_i each represent the 4-bit value for A and B; which make up the two integers whom of which are being operated on.

$OpCode[3:0]$	MUX_{OUT}	x_i	y_i	c_0	$Instruction$
0 0 0 0	c_0	A_i	0	1	dec: $A-1 \rightarrow A$
0 0 0 1	c_0	A_i	1	0	inc: $A+1 \rightarrow A$
0 0 1 0	c_0	A_i	B_i	0	add: $A+B \rightarrow A$
0 0 1 1	c_0	A_i	$\overline{B_i}$	1	sub: $A-B \rightarrow A$
0 1 0 1	c_1	A_i	1	0	and: A & $B \rightarrow A$
1 0 1 0	c_2	A_i	B_i	0	comp: $\overline{A} \rightarrow A$
1 1 1 1	c_3	A_i	$\overline{B_i}$	1	or: A — B → A

Stage Generator

State Diagram As stated prior, this machine is configured to have an asynchronous reset; in doing so, the machine needs to be able to decide what operations it must be accomplishing. Figure 2 depicts an interpretation of the ASM chart of the serial arithmetic processor. The diagram depicts the different states that the machine may undergo during its operation. Note the implicit timing information that is denoted by box which indicates exactly what operation the machine is undergoing during said state (denoted by T_i where i represents a state from 1-5).

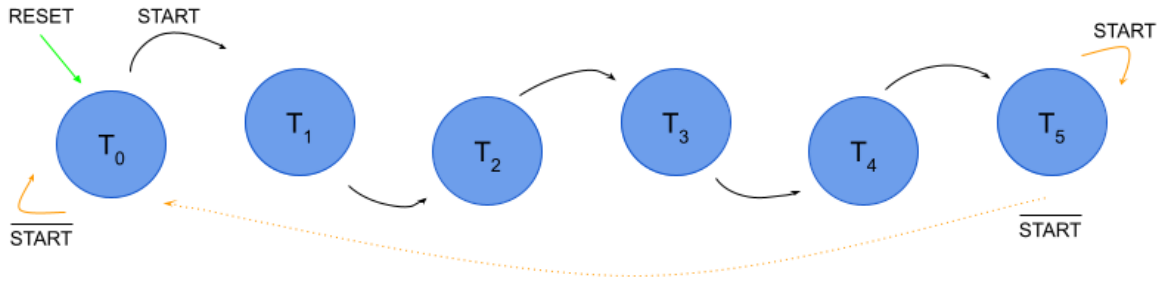

Figure 2: State diagram for the serial arithmetic prossesor. The ASM for this machine can be summed up in the following sentences: Hold in the initial state (T0) until **START** is set to 1. Move through the remaining 5 states and return to $T0$ when completed. The value of **START** does not matter while in states 1-4. The system will pause in $T5$ if **START** is left at 1, and will only return to $T0$ if **START** $= 0$. Asynchronously return to the initial state (T0) when the system **RESET** is '0'

State Table & Formulas The following is the state table and the formulas derived from the State table. Note that the inputs for the table are PS and START

Table 2: Transition table for the stage generator portion of the machine.

PS	START	NS
T_0	0	T_0
T_0	1	T_1
T_1	x	T_2
T_2	x	T_3
T_3	x	T_4
T_4	x	T_5
T_5	1	T_5
T_5	0	T_0

$DFF \rightarrow D_0 = T_0^+$
$D_1 = T_0 \cdot START$
$D_2 = T_1$
$D_3 = T_2$
$D_4 = T_3$
$D_5 = T_4 + (T_5 \cdot START)$

State Generator Block Diagram Schematic The first diagram shown in this project is of the state generator. The main function behind this module is to generate the state in which the machine is operating in. Note that the clock is wired to reset asynchronously.

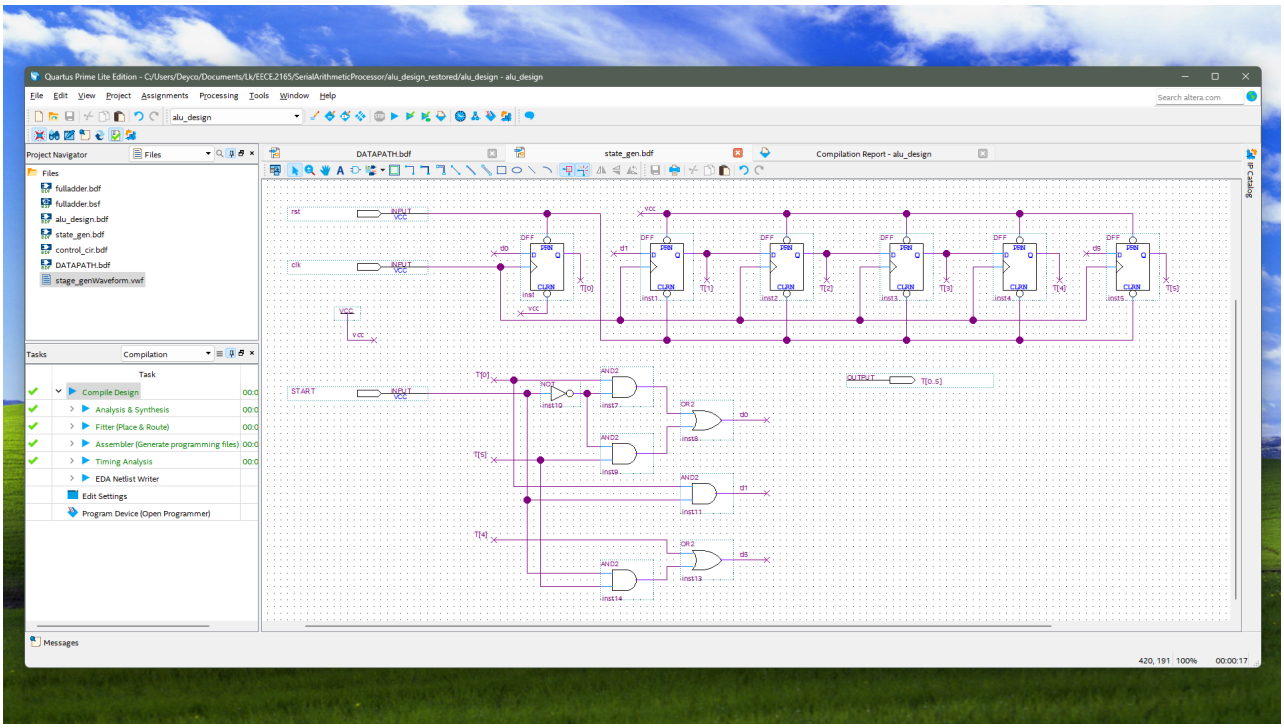

Figure 3: Block diagram figure of the state generator module used in the serial arithmetic processesor in Quartus Prime.

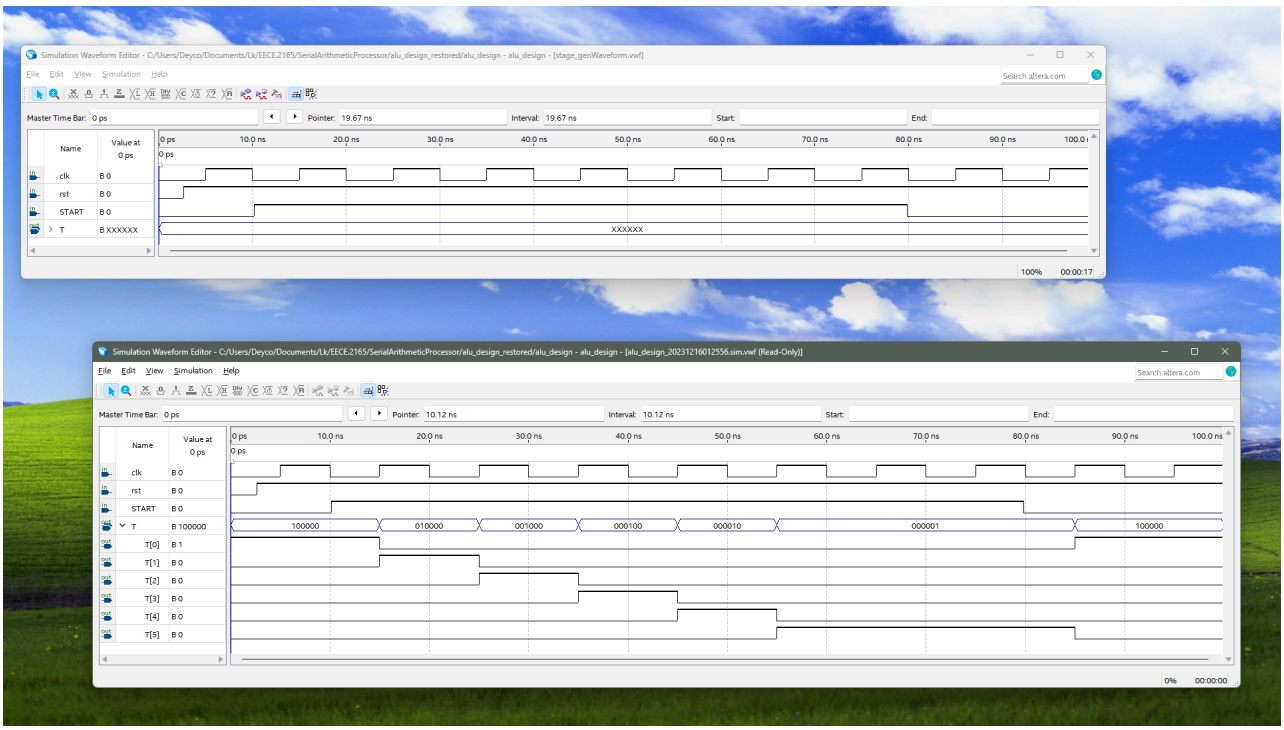

Figure 4: Waveform diagram of the state generator module indicating that the module is functioning as intended.

Control Circuit

Control Circuit Truth Table The last module that must be configured for this machine is the control circuit. The purpose behind this module is it is essentially a ccombinational circuit whose inputs sets the control signals; these signals are based on the PS and current inputs.

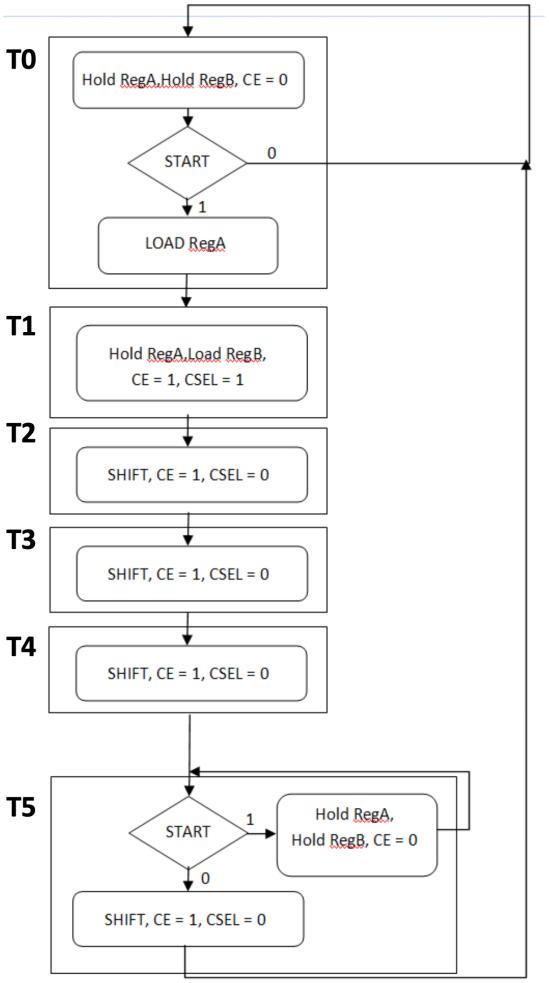

Figure 5: ASM chart of the serial arithmetic processor. This chart in particular was used to develop the formulas for the control circuit. Note that the register instructions are purposely left implicit for the sake of a more approachable visual interpretation of the machine's instructions and operation.

Table 3: Truth table derived from the ASM chart in Figure 5. Since the instructions are implicitly stated, statements had to be translated. "Hold Reg A" will refer to literally holding the 4-bit number held in register A, but also represents the operation that the register will accomplish with the said number.

PS	$START$	s1rA s0rA	s1rB s0rB	$CSEL$	CE
T_0	0	0 0	0 0	d	0
T_0	1	1 1	d d	d	d
T_1	d	0 0	1 1	1	1
T_2	d	0 1	0 1	0	1
T_3	d	0 1	0 1	0	1
T_4	d	0 1	0 1	0	1
T_5	1	0 0	0 0	d	0
T_5	0	0 1	0 1	0	1

Control Circuit & APU Circuit Prior to constructing the block diagram, formulas must be derived once more from its respective table Note that the formulas for CE, s0rB, and s1rB, CSEL are shared respectively. This makes sense as if you locate the outputs on the control circuit truth table, depend on what values you set the Don't Cares to, you can have the same outputs.

$$s1rA = T_0 \cdot START \qquad \overline{s0rA} = (T_0 \cdot \overline{START}) + T_1 + (T_5 \cdot START) \tag{1}$$

$$s1rB = CSEL = T_1 \qquad s0rB = CE = T_0 \oplus \overline{(T_5 \cdot START)} \tag{2}$$

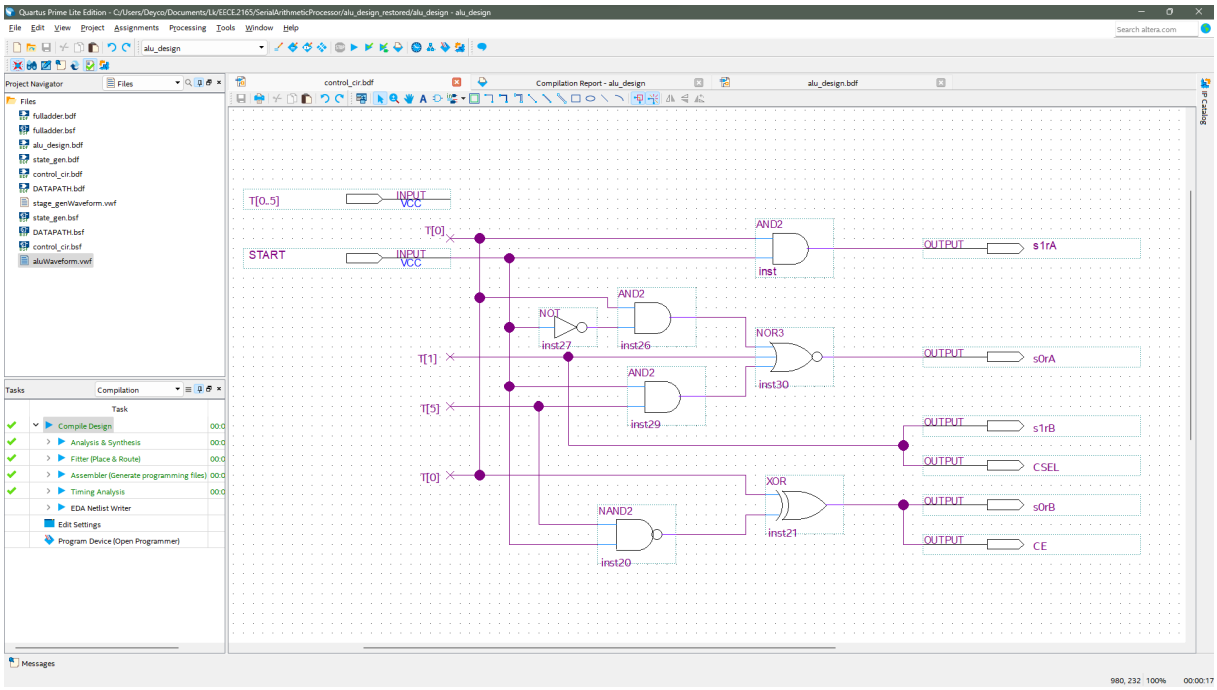

Figure 6: Block diagram schematic of the control circuit module.

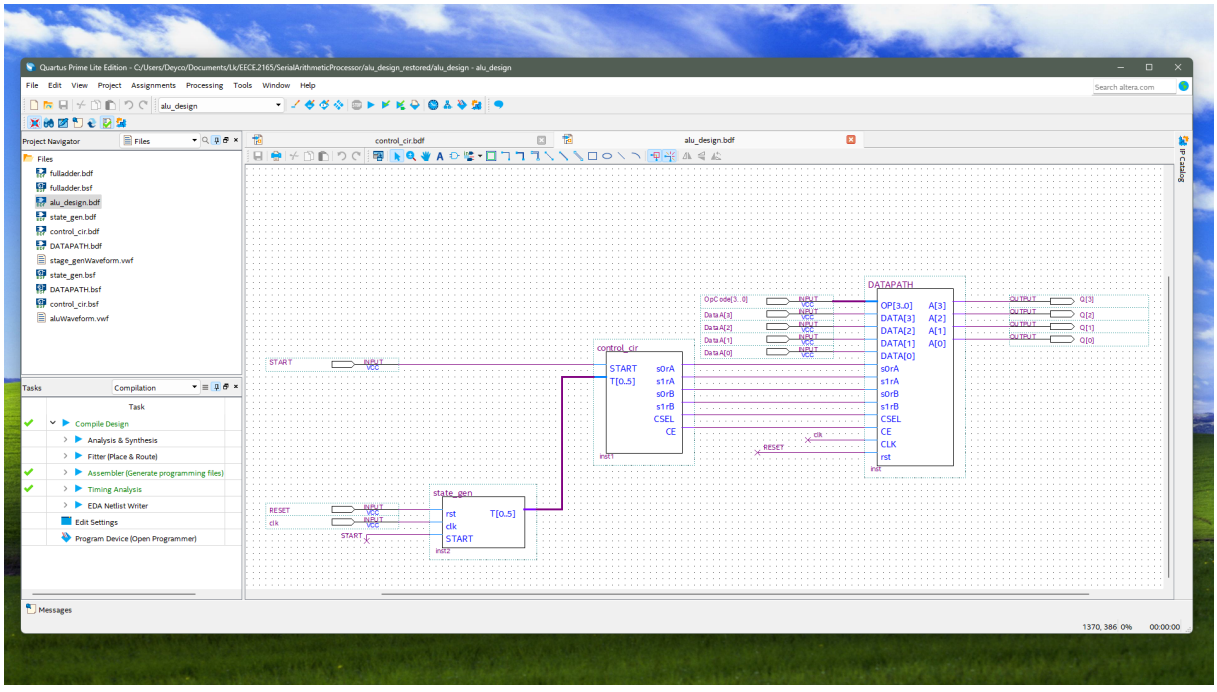

Figure 7: Block diagram schematic of all three modules wired together to compose the serial arithmetic processor.

ALU Tests

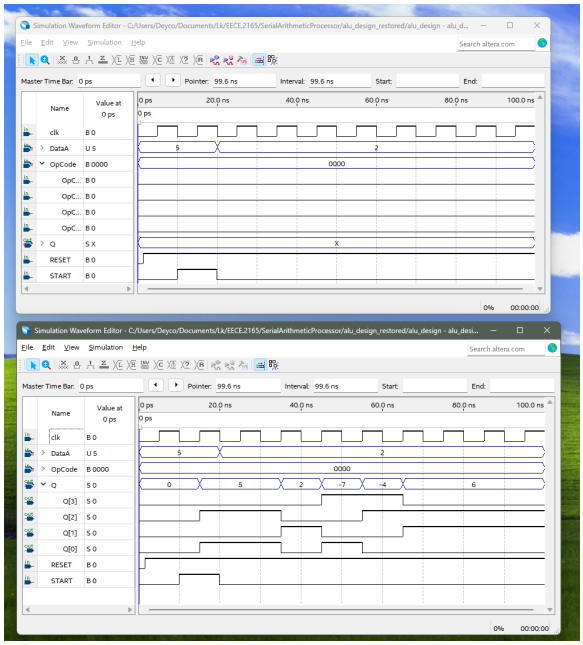

Figure 8: OpCode 0000 (DEC)

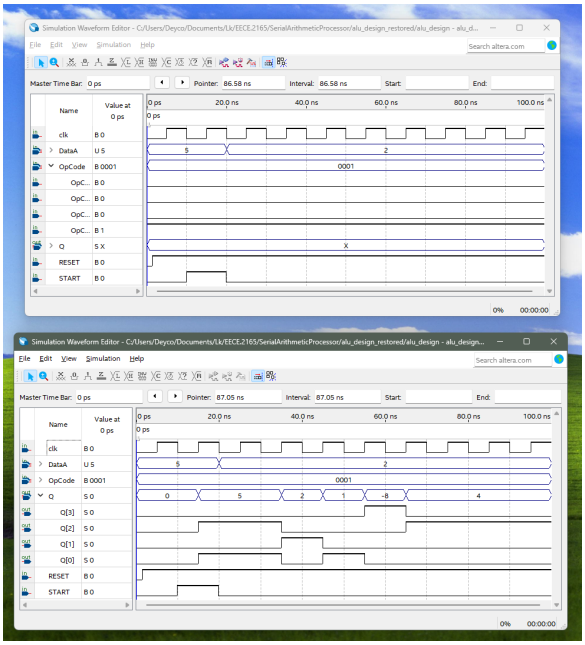

Figure 9: OpCode 0001 (INC)

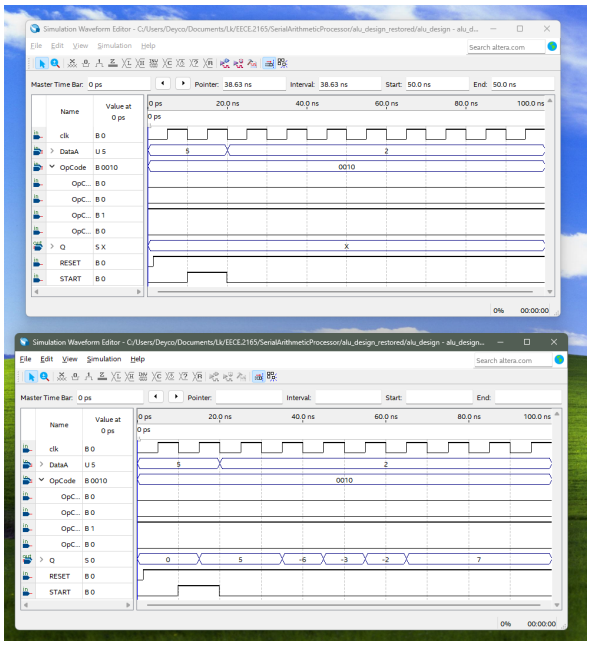

Figure 10: OpCode0010 (ADD)

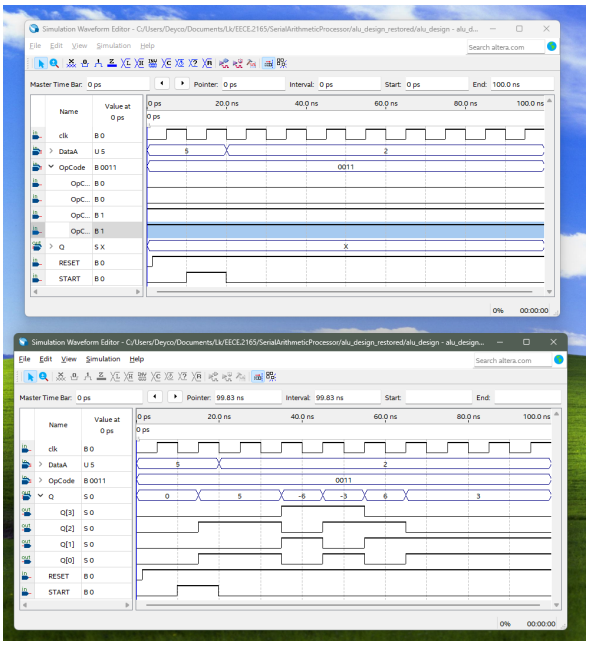

Figure 11: OpCode 0011 (SUB)

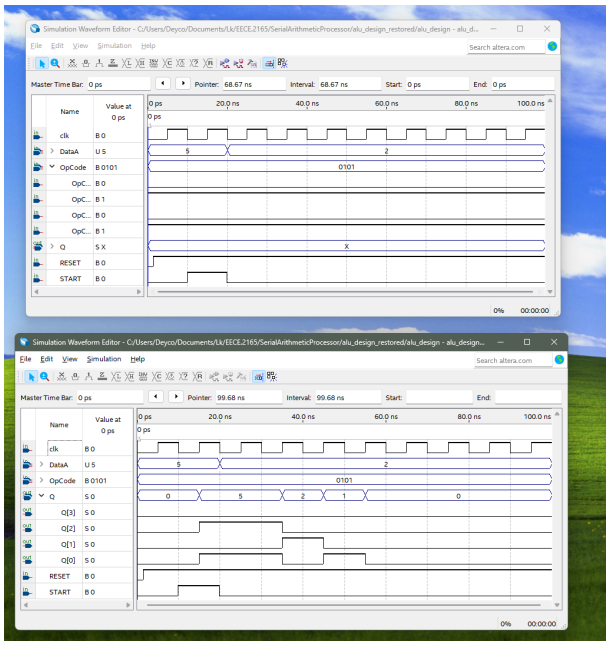

Figure 12: OpCode 0101 (AND)

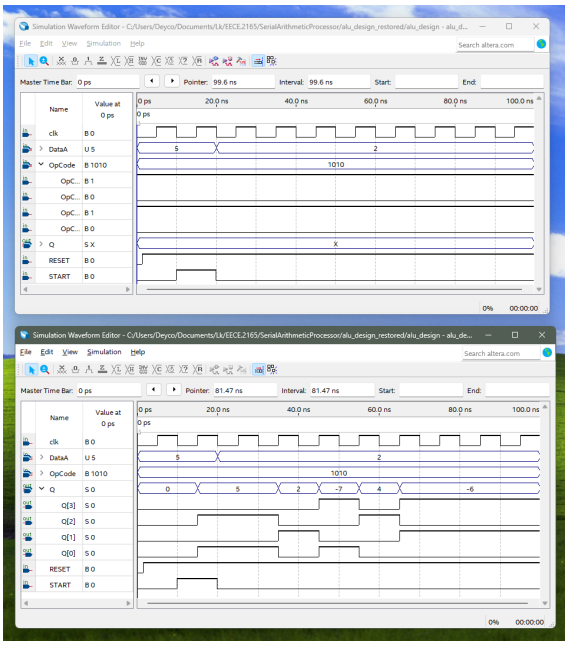

Figure 13: OpCode 1010 (COMP)

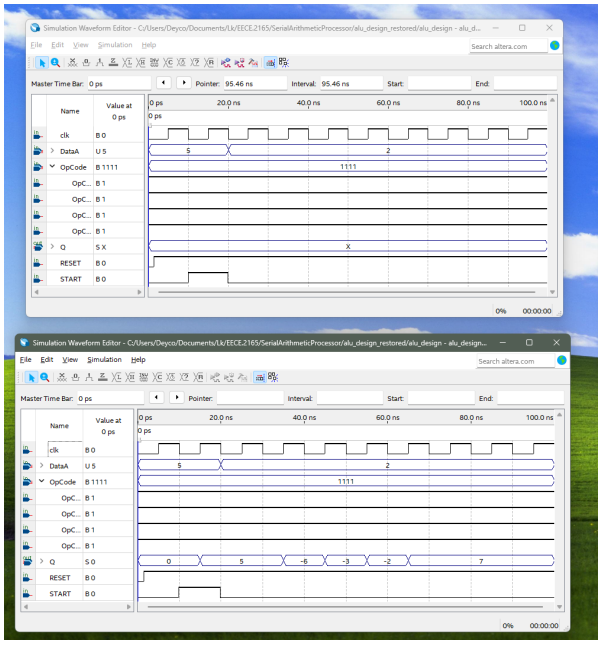

Figure 14: Opcode 1111 (OR)