The Beckstreet Boys

Cameron Buchman

Jiang Bowen

Joshua Guzman

Albey Kappil

Swamynathan Singaravelu

System Design

November 22, 2019

Parts List

Multiplexers -

- muxSelector - A Multiplexer module with 12 channels of 16-bit input from the output of each of the operation modules of the ALU. This module uses a one-hot input select from the controlLogic to choose between the channels to output a 16-bit solution.
- mux_a - A Multiplexer with 2 channels of 16-bit inputs, one being the 'a' input number and the other being an internal reset input (16-bit 0) when the opcode is reset. Channel is selected based on the one-bit selector of the mux coming from 12th bit of the one-hot output of the control logic.
- mux_b - A Multiplexer with 2 channels of 16-bit inputs, one being the 'b' input number and the other being an internal reset input (16-bit 0) when the opcode is reset. Channel is selected based on the one-bit selector of the mux coming from 12th bit of the one-hot output of the control logic.

controlLogic-

- This module acts as a decoder and it takes an input of 4-bit opcode representing the operation taking place and outputs a corresponding one-hot number representing the input opcode.

Registers -

- reg_a - takes the result of the mux of a and outputs the result to the nine operation modules when the clock goes from low to high or high to low.
- reg_b - takes the result of the mux of b and outputs the result to the nine operation modules when the clock goes from low to high or high to low.

accumulator -

- takes the output of the muxSelector as an input and outputs it to the finalOutput wire
- This module takes in a 16-bit number for input and outputs a 16-bit output

Clk -

- The clock pulses every 5 ticks between low and high.

Input List:

input [15:0] a, b;

A: First input number for all of math and function gates.

B: Second input number for all of math and function gates except "shift" and "not".

input [3:0] opcode;

Opcode: An input to choose the operator of the calculation.

NOTE: Inputs and outputs for individual modules can be found in the Modules List section.

Output List:

wire [15:0] finalOutput;

finalOutput: This 16 bits output shows what the result is by the input a, b and opcode.

NOTE: Inputs and outputs for individual modules can be found in the Modules List section.

Interface List:

wire [11:0] select;

select: It is a 12 bits output transfer from the 4 bit opcode, and be used in “Add_Sub” and “muxSelector”.

wire [15:0] dff_a_out;

dff_a_out: It is a 16 bits output which is the output of the register with the input of “a”.

wire [15:0] dff_b_out;

dff_b_out: It is a 16 bits output which is the output of the register with the input of “b”.

wire [15:0] and_out, nand_out;

and_out: It is a 16 bits output from the “And” gate with the result of “and” of inputs “a” and “b”.
nand_out: It is a 16 bits output from the “Nand” gate with the result of “nand” of inputs “a” and “b”.

wire [15:0] or_output, nor_output;

or_output: It is a 16 bits output from the “Or” gate with the result of “or” of inputs “a” and “b”.
nor_output: It is a 16 bits output from the “Nor” gate with the result of “nor” of inputs “a” and “b”

wire [15:0] xor_output,xnor_output;

xor_output: It is a 16 bits output from the “Xor” gate with the result of “xor” of inputs “a” and “b”.
xnor_output: It is a 16 bits output from the “Xnor” gate with the result of “xnor” of inputs “a” and “b”.

wire [15:0] not_out;

not_out: It is a 16 bits output from the “Not” gate with the result of “not” of the input “a”.

wire [15:0] addsub_out;

addsub_out: It is a 16 bits output from the “Add_Sub” with the result of sum or difference of inputs “a” and “b”.

wire [15:0] shLeft_out, shRight_out;

shLeft_out: It is a 16 bits output from the “Shifter” with the result of “shiftLeft” of the input “a”.
shRight_out: It is a 16 bits output from the “Shifter” with the result of “shiftRight” of the input “a”.

wire [15:0] mux_out;

mux_out: It is a 16 bits output from “muxSelector” gate depends on the inputs “a”, “b” and “opcode”, which gives a final result of the calculate.

wire carry;

carry: It is a one bit value for the “Add_Sub”.

wire overflow;

overflow: It is a one bit value for the “Add_Sub”.

Local reset variable ;

- A reset input to zero of 16-bits to mux a and mux b channels.

Modules List:

muxSelector module
Inputs: addsub_out ,shRight_out, shLeft_out, and_out, or_output, xor_output, xnor_output, nand_out, nor_output, not_out, select
Outputs: mux_out

A Multiplexer module with 12 channels of 16-bit input from the output of each of the operation modules of the ALU. this module uses a one-hot input select from the controlLogic to choose between the channels to output a 16-bit solution.
This module has local parameter to define the one-hot selectors.

controlLogic Module
Inputs: opcode
Outputs: select

This module acts as a decoder and it takes an input of 4-bit opcode representing the operation taking place and outputs a corresponding one-hot number representing the input opcode. This module has local parameter to define both the binary inputs and the one-hot outputs.

accumulator Module
Inputs: mux_out
Outputs: finalOutput

This module takes in a 16-bit number for input and make it to be a 16-bit output.

Shifter module
Inputs: reg_a_out
Outputs: shLeft_out, shRight_out

This module takes in a 16-bit number and does a left and right shift of the number.

Add_Sub Module
Inputs: reg_a_out, reg_b_out, select[7]
Outputs: addsub_out, carry, overflow, reg_a_out

This module takes an 16-bit input of A and B and a select bit that determines if operation will be addition or subtraction and outputs the sum or difference, carry, and overflow of the operation. The value of select determines if the operation to be performed is addition or subtraction. If subtraction, the 2's complement of reg_b_out is calculated before the subtraction operation takes place.

Breadboard Module
Inputs: a,b,opcode,clk
Outputs: select

All modules except the Testbench Module, wires, registers have been instantiated in this module and assigned to the parameters inside the breadboard module. This module acts as a nexus between all the modules and receives and passes initial inputs and final outputs.

Testbench Module

Contains the local parameters of the 4bit binary opcode inputs. Contain input registers inputs of the ALU which are a, b, opcode, and clk. Contains the wire select.

XOR Module
Inputs: reg_a_out, reg_b_out
Outputs: xor_out

This module takes in two 16-bit numbers performing the exclusive-or operation on the two inputs and outputs the result.

XNOR Module
Inputs: reg_a_out, reg_b_out
Outputs: xnor_out

This module takes in two 16-bit numbers performing the xnor operation on the two inputs and outputs the result.

OR Module
Inputs: reg_a_out, reg_b_out
Outputs: or_out

This module takes in two 16-bit numbers and will return one 16-bit number, either inputs a or b as long as at least one is valid.

NOR Module
Inputs: reg_a_out, reg_b_out
Outputs: nor_out

This module takes in two 16-bit numbers and will return one 16-bit number, as long as neither inputs a or b are valid.

NOT Module
Inputs: reg_a_out,
Outputs: not_out

This module takes in a 16-bit number for input and make the inverses of it to be a 16-bit output.

AND Module -
Inputs: reg_a_out, reg_b_out
Outputs: and_out

This module takes in two 16-bit number for input and make the 'and' result of it to be a 16-bit output.

NAND Module-
Inputs: reg_a_out, reg_b_out
Outputs: nand_out

This module takes in two 16-bit number for input and make the 'nand' result of it to be a 16-bit output.

Mode List:

Ready : ALU is ready to perform the next operation and selects operation to perform. This is also the mode that the ALU remains on during no-operation actions.

Add: ALU is currently performing the addition operation on two inputs. Once completed, ALU returns to Ready mode. If an error such as carry or overflow occurs, ALU transitions to error mode from this mode.

Sub: ALU is currently performing the subtraction operation on two inputs. Once completed, ALU returns to Ready mode. If an error such as carry or overflow occurs, ALU transitions to error mode from this mode.

And: ALU is currently performing the and operation on two inputs. Once completed, ALU returns to Ready mode.

Or: ALU is currently performing the or operation on two inputs. Once completed, ALU returns to Ready mode.

ShiftR: ALU is currently performing the shift right operation on an input. Once completed, ALU returns to Ready mode.

ShiftL: ALU is currently performing the shift left operation on an input. Once completed, ALU returns to Ready mode.

Xor: ALU is currently performing the exclusive-or operation on two inputs. Once completed, ALU returns to Ready mode.

Nand: ALU is currently performing the not-and operation on two inputs. Once completed, ALU returns to Ready mode.

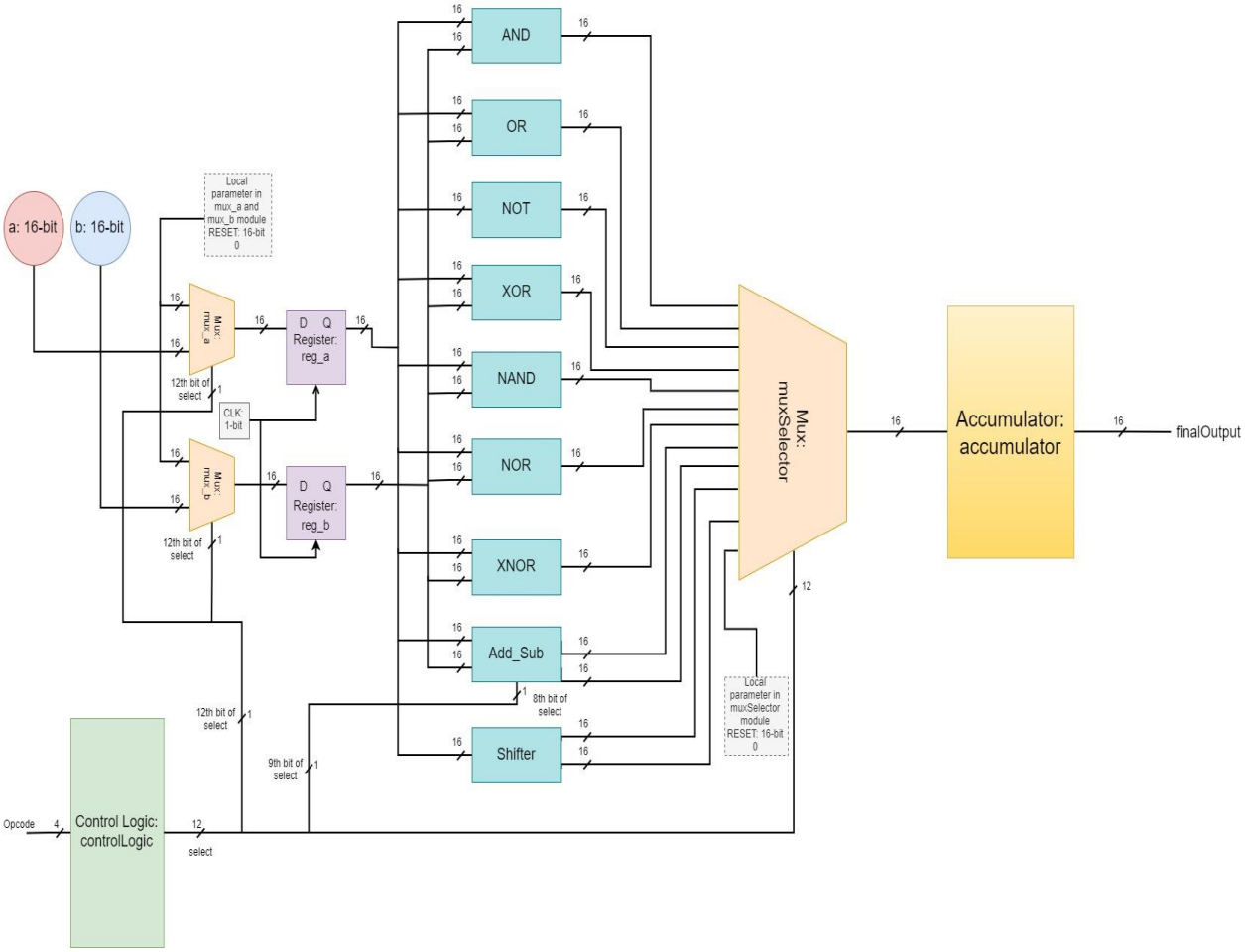
Nor: ALU is currently performing the not-or operation on two inputs. Once completed, ALU returns to Ready mode.

Xnor: ALU is currently performing the exclusive-not-or operation on two inputs. Once completed, ALU returns to Ready mode.

Not: ALU is currently performing the not operation on an input. Once completed, ALU returns to Ready mode.

Error: When the Add or Sub modes result in carry or overflow errors, those two modes transition to the error mode. The clear command has to be used to clear memory and allow the ALU to return to the Ready mode.

Circuit Diagram



State Machine