**CSE 664 Introduction to System-on-Chip Design**

**Final Project – 8-Bit Microcontroller**



Revision: 1.0

Date Prepared: June 2023

Prepared by: Austin Benedetti, Daniel Ems, Nicholas Lyons, Mike Pena, Trent Reeves, Mimansa Verma, Derrick Ward, Jiawen Zhen

Prepared for: CSE 664 Introduction to System-on-Chip Design, Spring 2023, section M401, Professor Mohammed Abdallah

Contents

[1 Design Documentation 4](#_Toc137675953)

[1.1 Entity: acc\_simple 4](#_Toc137675954)

[1.1.1 Diagram 4](#_Toc137675955)

[1.1.2 Description 4](#_Toc137675956)

[1.1.3 Ports 4](#_Toc137675957)

[1.1.4 Signals 5](#_Toc137675958)

[1.1.5 Processes 5](#_Toc137675959)

[1.2 Entity: ALU\_8bit 6](#_Toc137675960)

[1.2.1 Diagram 6](#_Toc137675961)

[1.2.2 Description 6](#_Toc137675962)

[1.2.3 Ports 6](#_Toc137675963)

[1.2.4 Processes 6](#_Toc137675964)

[2 Entity: controller\_fsm 8](#_Toc137675965)

[2.1 Diagram 8](#_Toc137675966)

[2.2 Description 8](#_Toc137675967)

[2.3 Ports 8](#_Toc137675968)

[2.4 Processes 9](#_Toc137675969)

[3 Entity: nbit\_two\_one\_mux 10](#_Toc137675970)

[3.1 Diagram 10](#_Toc137675971)

[3.2 Description 10](#_Toc137675972)

[3.3 Ports 10](#_Toc137675973)

[4 Simulations 11](#_Toc137675974)

[4.1 Modelsim - accumulator.v 11](#_Toc137675975)

[4.2 Modelsim – ALU\_8bit.v 11](#_Toc137675976)

[4.3 Modelsim – controller\_fsm.v 12](#_Toc137675977)

[4.4 Modelsim – nbit\_two\_one\_mux.v 12](#_Toc137675978)

[4.5 Modelsim – program\_counter.v 12](#_Toc137675978)

[5 References 13](#_Toc137675979)

[APPENDIX A: 8-Bit Microcontroller Code Base 13](#_Toc137675980)

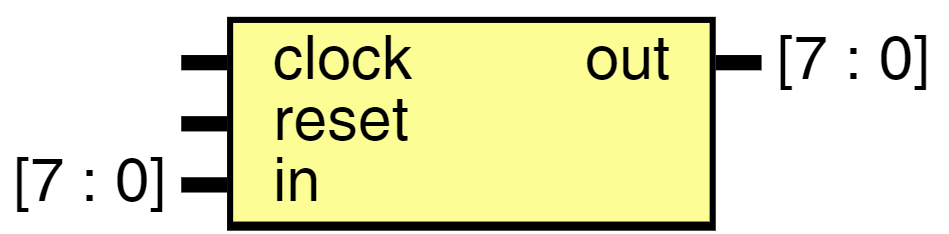
THIS PAGE INTENTIONALLY LEFT BLANK

# Design Documentation

## Entity: acc\_simple

* **File**: accumulator.v

### Diagram



### Description

A register in which intermediate arithmetic logic unit results are stored. This simple 8-bit accumulator's output will continuously grow by increments of the input until reset.

Note - Overflow is not handled

### Ports

| **Port name** | **Direction** | **Type** | **Description** |
| --- | --- | --- | --- |
| clock | input |  | Clock. Output is posedge triggered |
| reset | input |  | Clears register value (1 = clear, 0 = hold) |
| in | input | [7 : 0] | Input from ALU |
| out | output | [7 : 0] | Output return to ALU |

### Signals

| **Name** | **Type** | **Description** |
| --- | --- | --- |
| accumulator | reg [7 : 0] | Temporary storage for accumulator output |

### Processes

* unnamed: ( @(posedge clock or posedge reset) )

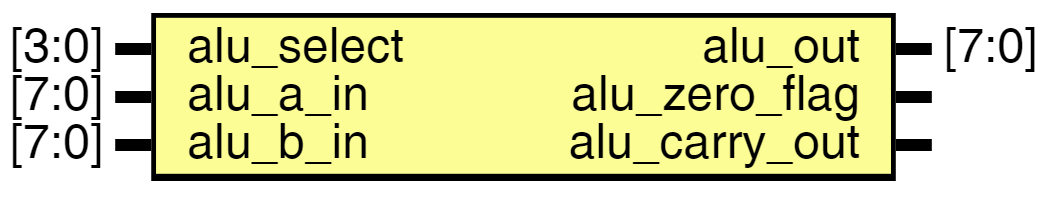
**Type:** always

**Description**  
Increments output by input at rising edge of clock.  
If reset is high then clear output.

## Entity: ALU\_8bit

* **File**: ALU\_8bit.v

### Diagram



### Description

A combinational digital circuit that performs arithmetic and bitwise operations on integer binary numbers.

### Ports

| **Port name** | **Direction** | **Type** | **Description** |
| --- | --- | --- | --- |
| alu\_out | output | [7:0] | 8 bit result (connected to mux that is connected to ACC) |
| alu\_zero\_flag | output |  | Set if result is zero (connected to controller) |
| alu\_carry\_out | output |  | Set if carry bit is needed (connected to controller) |
| alu\_select | input | [3:0] | Select ALU operation (received from controller) |
| alu\_a\_in | input | [7:0] | ACC ALU input |
| alu\_b\_in | input | [7:0] | REG ALU input |

### Processes

* unnamed: ( @(alu\_a\_in or alu\_b\_in or alu\_select) )

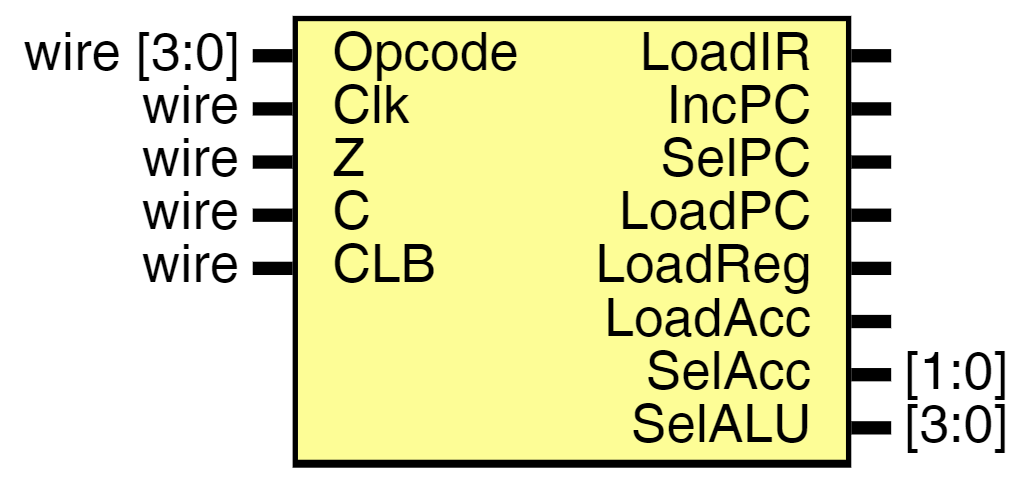
**Type:** always

**Description**  
The ALU outputs change anytime the inputs change (combinational logic).

# Entity: controller\_fsm

* **File**: controller\_fsm.v

## Diagram



## Description

Model that uses a known set of inputs to define several combinations of outputs and states.

## Ports

| **Port name** | **Direction** | **Type** | **Description** |
| --- | --- | --- | --- |
| LoadIR | output |  | Load instruction register with next instruction - (Should be checked before update) |
| IncPC | output |  | Program counter which is incremented only to next instruction |
| SelPC | output |  | Used to increment PC by immediate or val in reg |
| LoadPC | output |  | Signal to update PC value - (Should be checked before JUMP ONLY) |
| LoadReg | output |  | Signal to update register - (Should be checked before update) |
| LoadAcc | output |  | Signal to update accumulator - (Should be checked before update) |
| SelAcc | output | [1:0] | Select signal for ACC muxes - (SelAcc[1] = SelAcc1, SelAcc[0] = SelAcc0) |
| SelALU | output | [3:0] | Select signal for ALU operation (opcode) |
| Opcode | input | wire [3:0] | Opcode from instruction register |
| Clk | input | wire | Clock signal |
| Z | input | wire | Zero bit |
| C | input | wire | Carry bit |
| CLB | input | wire | TODO: WHAT IS THIS? |

## Processes

* unnamed: ( @(Clk) )

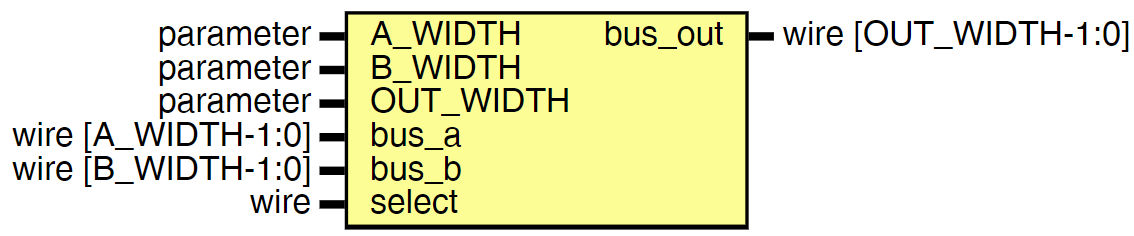
**Type:** always

**Description**  
Case statement for setting control signals

# Entity: nbit\_two\_one\_mux

* **File**: nbit\_two\_one\_mux.v

## Diagram



## Description

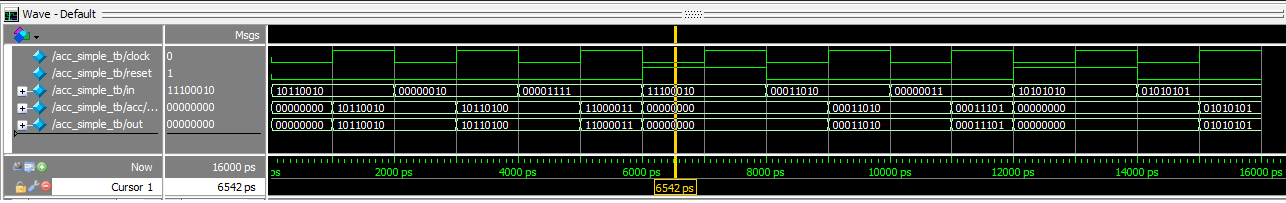
A data selector device that selects between several analog or digital input signals and forwards the selected input to a single output line.

## Ports

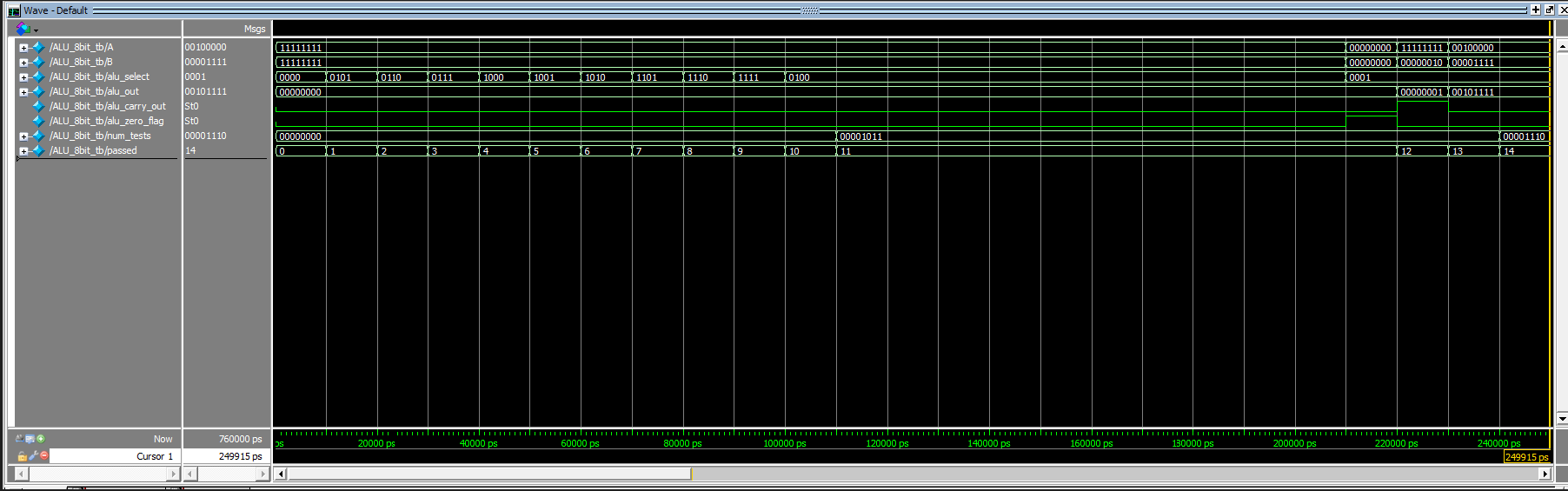
| **Port name** | **Direction** | **Type** | **Description** |
| --- | --- | --- | --- |
| A\_WIDTH | input | parameter |  |
| B\_WIDTH | input | parameter | Width of INPUT BUS A |
| OUT\_WIDTH | input | parameter | Width of INPUT BUS B |
| bus\_out | output | wire [OUT\_WIDTH-1:0] | Output bus |
| bus\_a | input | wire [A\_WIDTH-1:0] | Input bus A |
| bus\_b | input | wire [B\_WIDTH-1:0] | Input bus B |
| select | input | wire | Select Signal |

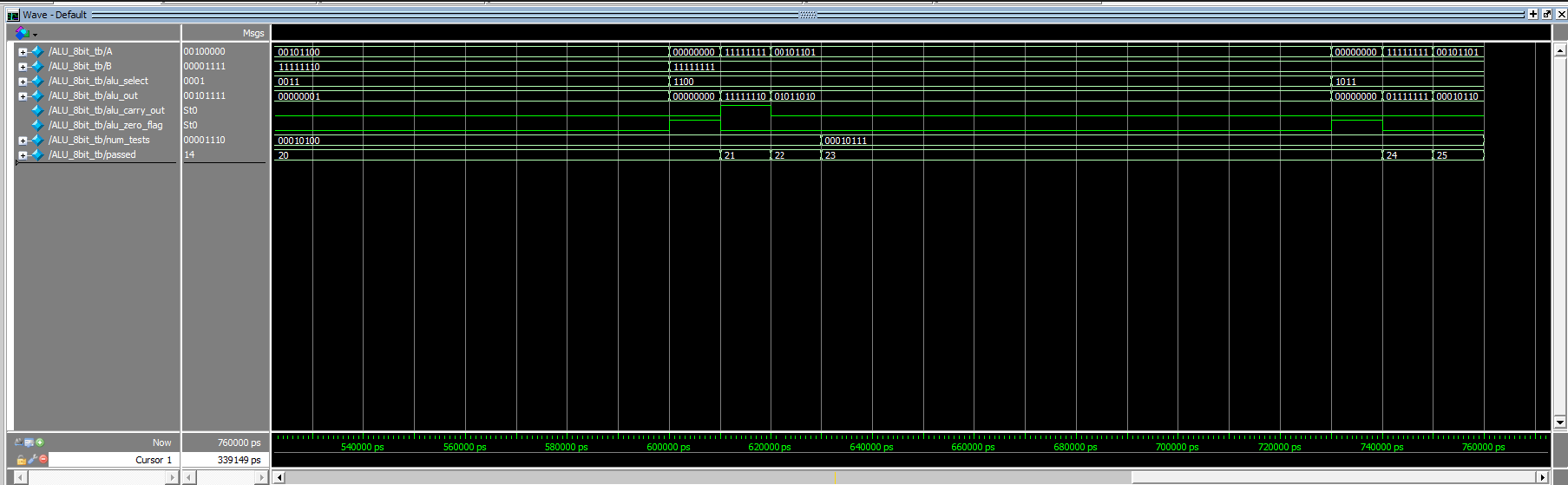
# Simulations

## Modelsim - accumulator.v



## Modelsim – ALU\_8bit.v

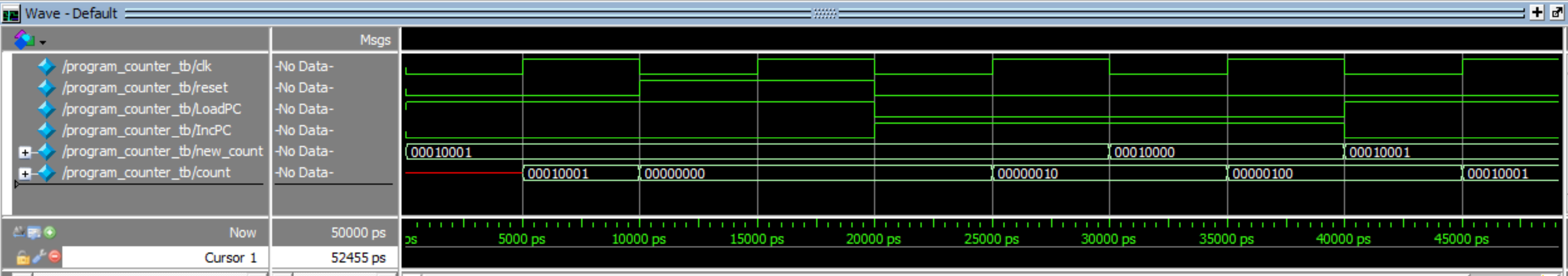




## Modelsim – controller\_fsm.v

## Modelsim – nbit\_two\_one\_mux.v

## Modelsim – program\_counter.v



# References

## APPENDIX A: 8-Bit Microcontroller Code Base



Prepared by:

Austin Benedetti, Daniel Ems, Nicholas Lyons, Mike Pena, Trent Reeves, Mimansa Verma, Derrick Ward, Jiawen Zhen