CS 223 – DIGITAL DESIGN

TWO'S COMPLEMENT CHECKSUM FOR MEMORY

MELİKE DEMİRCİ
21702346
SECTION 04

17.05.2020

1. Introduction

In this project, Two's Complement Checksum Machine is designed. A data input system and memory integrated with Checksum algorithm calculates the sum of memory elements in two's complement number system.

In order to deal with this problem, modules that store and manage data, which is coming through Basys3 buttons and switches, are created in Systemverilog. Besides, Basys3 leds are used to display input data and seven segment to display the memory elements, checksum and amount of clock cycles in computation.

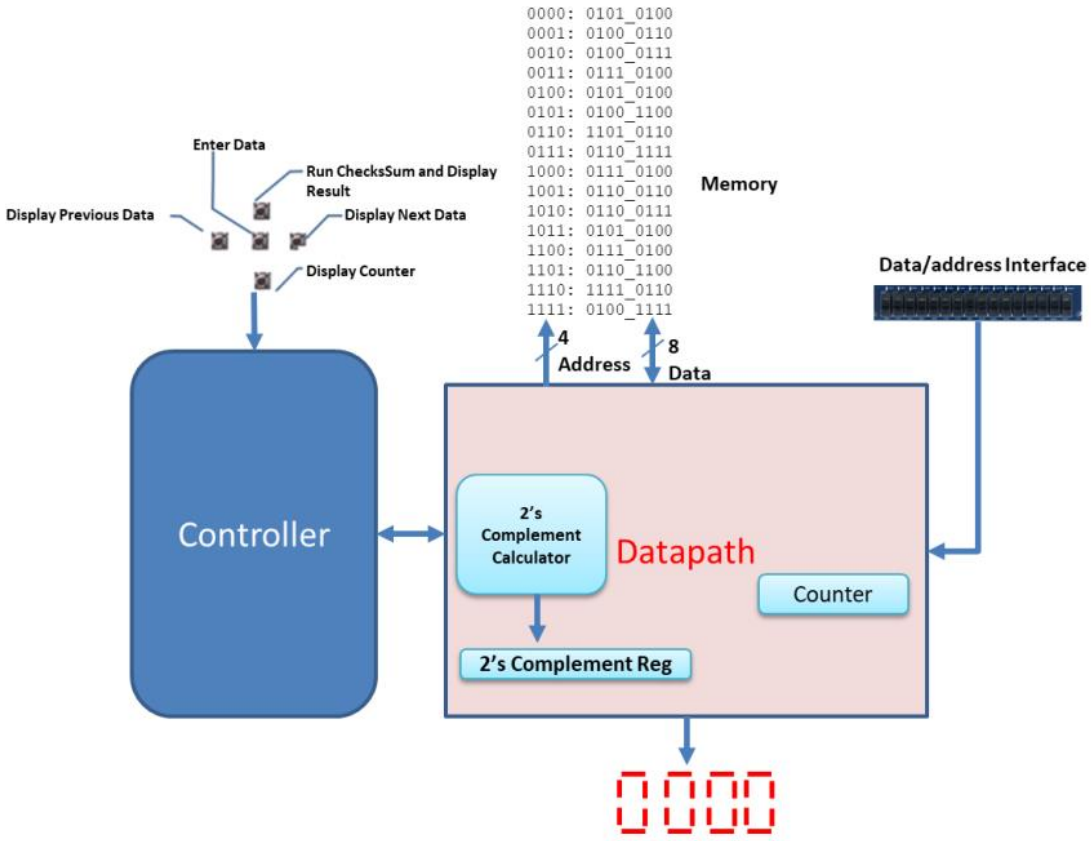

Figure 1: Operation Flow Architecture

2. Block Diagram

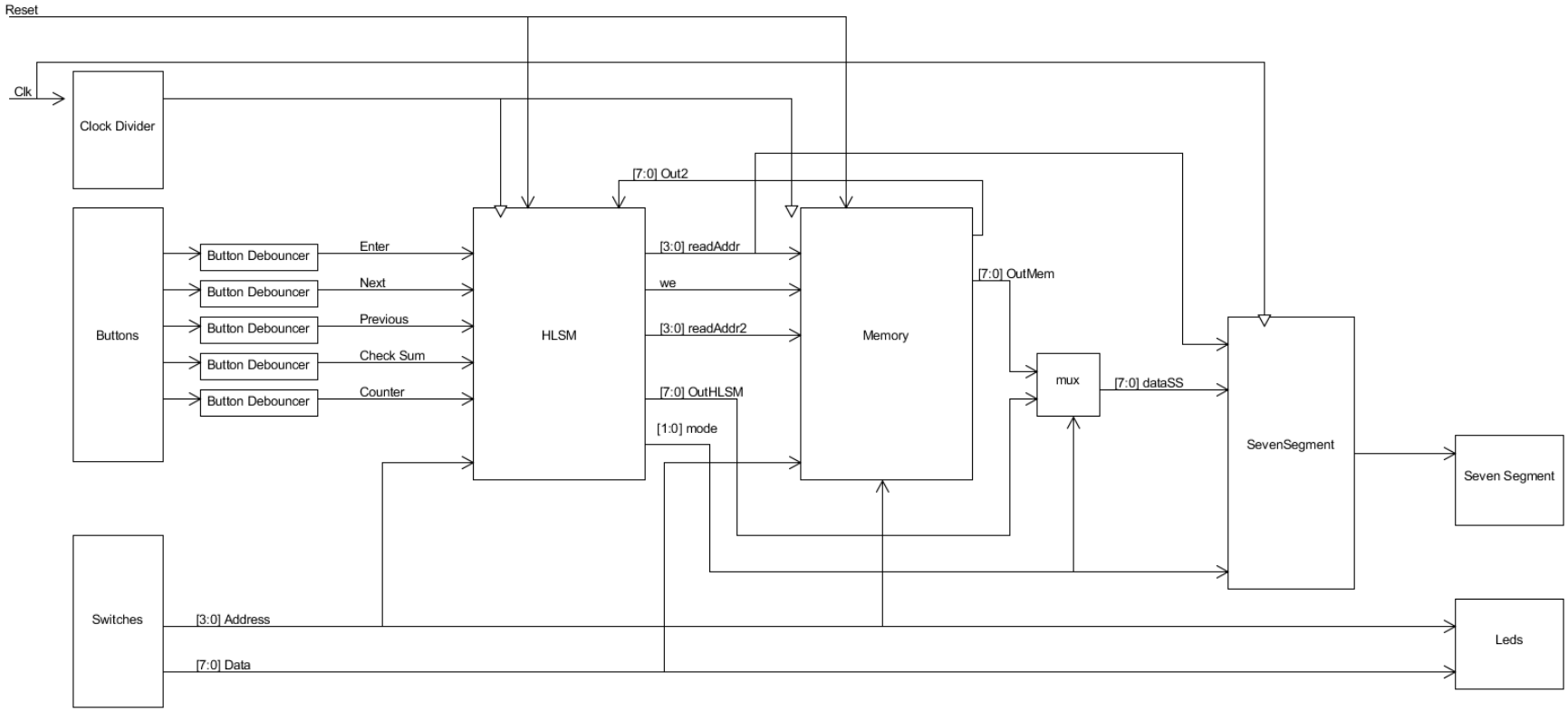

Figure 2: Block Diagram

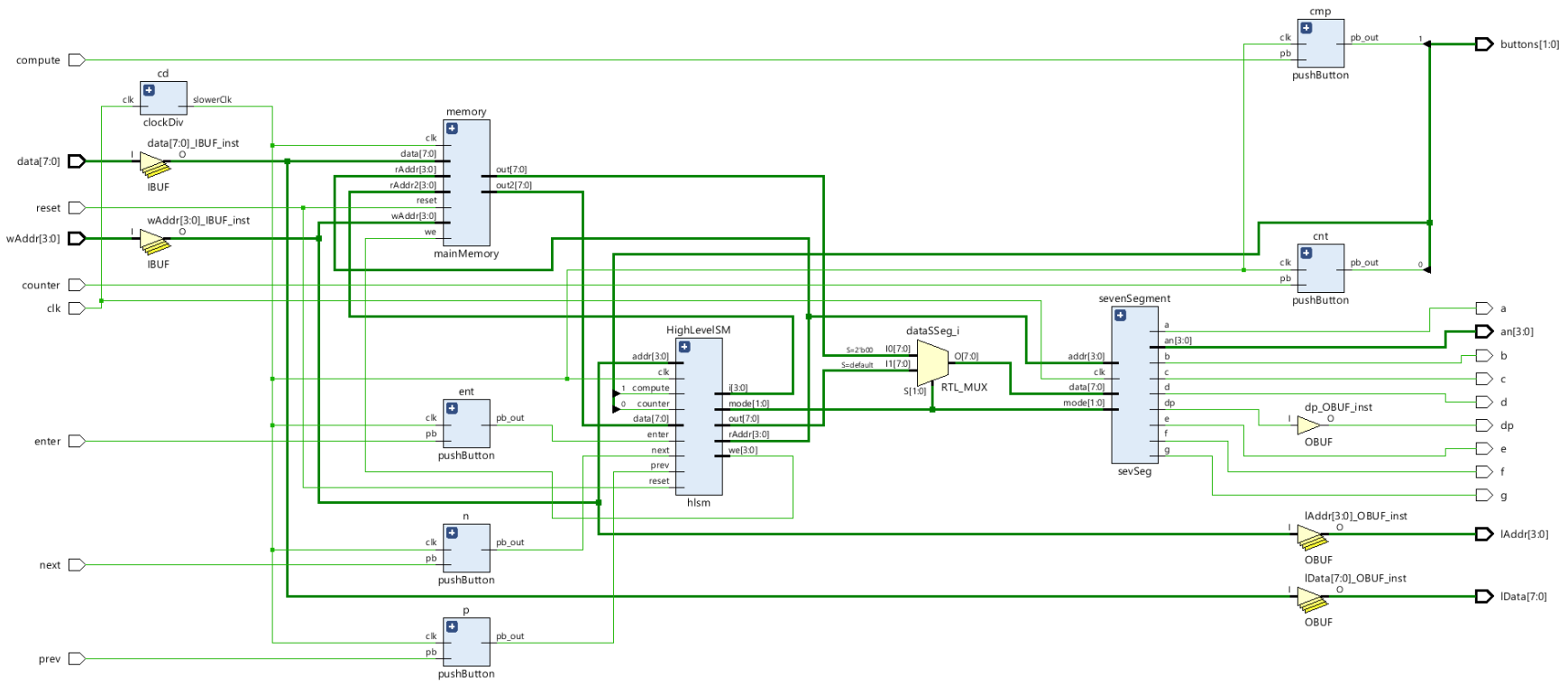

Figure 3: Elaborated Design from Vivado

3. Detailed Explanation of the Work

There are 3 main modules such as hlsm, mainMemory and sevSeg. These 3 module designed in order to execute main operations. Thus, there are pushButton and clockDiv modules in order to adjust clk and button inputs.

module *hlsm*

inputs: logic clk, reset, next, prev, compute, counter, enter, [3:0] addr,[7:0] data
outputs: [3:0]rAddr, we, [1:0] mode, [7:0] out, [3:0] i

This module is a high level state machine module as it can be understood from its name. If *reset switch becomes on*, initialized memory starts to be projected onto the seven segment on Basys3. Initially, address zero is displayed in memory display mode, which is adjusted by mode output passing to sevSeg module. When next button is pushed displayed address increments one and when prev button is pushed it decrements one. When address switches change, the data of the memory, which has index value coming from the switches, starts to be displayed.

Management of the memory display organizations are not included in the controller part of the hlsm. There is an address register which keeps the proper address according to the inputs. In controller, with mode signal, sevSeg module will be able to display content properly.

By using enter button, we becomes 1 for one clock cycles which enables the memory content to be adjusted according to the data and address coming from switches. If compute button is pushed, machine starts to calculate checksum in 16 clock cycles. In every cycle, output i increments by one and data flows from the mainMemory according to i. After the computation, sum and mode adjust checksum display mode in sevSeg module. When counter button is pushed, data of counter register and proper mode output goes into sevSeg in order to display clock cycle count.

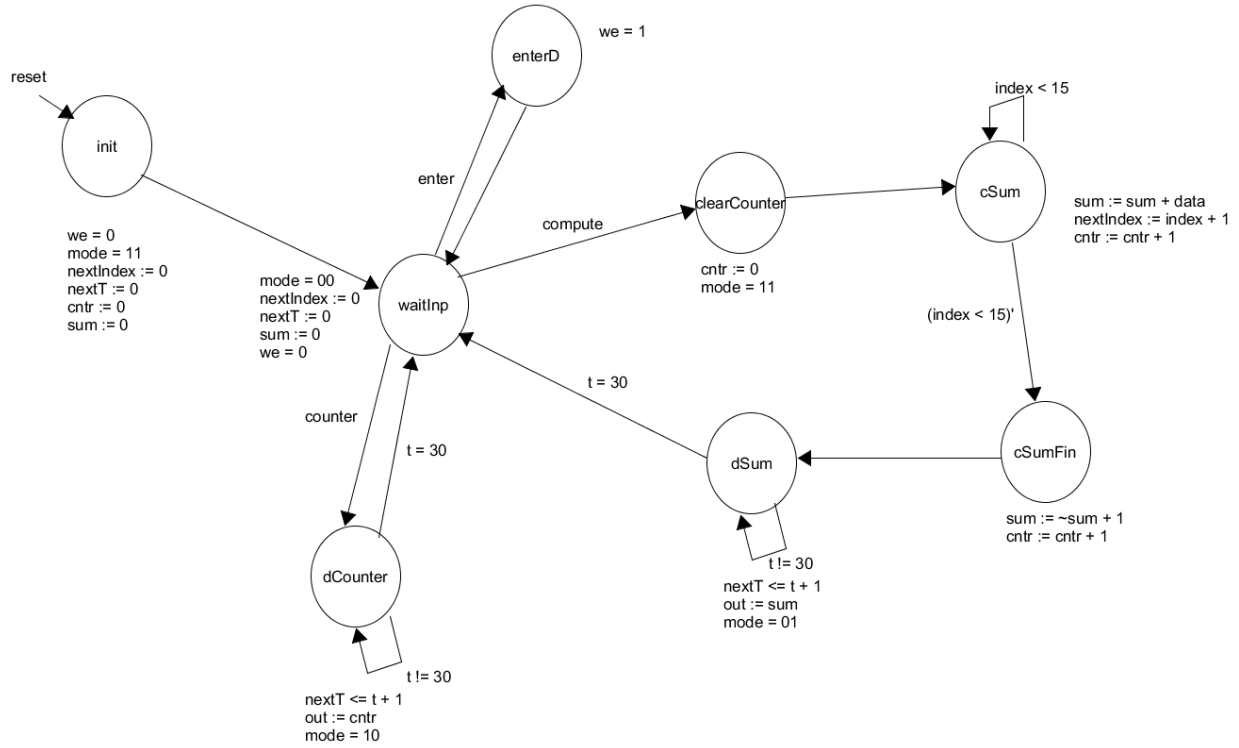

Figure 4: State Transition Diagram of HLSM

module *mainMemory*

inputs: logic clk, reset, we, [3:0] wAddr, [3:0] rAddr, [3:0] rAddr2, [7:0] data
outputs: logic [7:0] out, [7:0] out2

This module is a main memory of the machine. When reset is 1, memory is initialized with the following content 0x00, 0x01, 0x02, 0x03, 0x04 0x05, 0x06, 0x08, 0x09, 0x0a, 0x0b, 0x0c, 0x0d, 0x0e, 0x0f . When we is 1, module writes the data into the corresponding address. Out outputs the data in memory[rAddr], out2 outputs the data in memory[rAddr2].

module *sevSeg*

inputs: clk, [1:0] mode, [7:0] data, [3:0] addr
outputs: a, b, c, d, e, f, g, dp, [3:0] an

This module manages the lights on the seven-segment display. When mode signal is 00, seven-segment displays data and addr in memory diplay mode such as

“Address in hex – data in hex”. When mode is 01, data is displayed in checksum display mode which is “C = data in hex”. When mode is 10, data is displayed on two left digits in hex. It displays 0 if the checksum didn’t computed and diplays F if the calculation is done. F in decimal is 16, which is amount of clock cycles needed for calculation. Mode 11 is used to display “----“ while checksum is calculated and reset becomes 1.

module *clockDiv*
inputs: clk
outputs: slowerClk

This module slower the 100Mhz clock of the Basys3. Frequency of the slowerClk is 10hz.

module *pushButton*
inputs: clk, pb
outputs: pb_out

This module deals with the bounces in push button signal. The pb_out is generated as a single pulse with a period of the slow clock without bouncing as it is expected.[1]

4. Conclusion

To summarize, these modules and their implementations provide a properly working Two’s Complement Checksum Machine. The memory can be initialized and new data can be entered via switches and buttons. Also, memory content can be displayed with the control of switches and buttons. When memory is filled, machine can output the sum of the memory in two’s complement.

Finally, there is no observed problem in the design.

Referances

1- https://www.fpga4student.com/2017/04/simple-debouncing-verilog-code-for.html