VIETNAM NATIONAL UNIVERSITY
HO CHI MINH CITY UNIVERISTY OF TECHNOLOGY
Department of Electronics

EE4449: Logic Synthesis

LAB 01:

CLASS TT01 --- GROUP 08 --- SEMESTER 232

Supervisor : Dr. Linh Tran

Name	**ID**
Nguyen Pham Tien Dung	2051100

Ho Chi Minh City– 2023

TABLES OF CONTENTS

Warmup...2

1. Objectives 2
2. Analyses 2
3. Specification 3
4. Results..3
5. Discussion 5

LAB 1

Warmup

1. Objectives

- Review understanding of basic logic design and FSM
- Develop a combinational logic and FSM using SystemVerilog in the context of creating a '**downstream**' module for an FPGA programming project. This module should be capable of captures the calculated **sum** and displaying the result on a hex display.
- Write the **'p1'** module to be the top module of the design, connecting everything and synthesize on Quartus.

2. Analyses

In this segment, we will analyze the requirements of the laboratory and present a solution:

- **What are we addressing?** We are handling the design and implementation of a 'downstream' module using SystemVerilog. This module is designed to work in conjunction with the already designed 'sumitup' module in an FPGA programming project.
- **Why this approach?** This approach allows us to apply our understanding of basic logic design and Finite State Machines (FSM) in a practical context. It also provides an opportunity to work with FPGA programming and get familiar with the Altera DE10 board.
- **How is the module designed?** The 'downstream' module is designed to capture the sum calculated by the 'sumitup' module and hold it for display while the next sum is being calculated. The operation is controlled using two signals, 'clk' and 'reset_l'. On every positive edge of the 'clk' signal, if the 'reset_l' signal is not asserted, the 'sum_out' signal is set to zero. If the 'done' signal is asserted, indicating that the 'sumitup' module has finished calculating a sum, the 'sum_out' signal is set to the value of the 'sum_in' signal. The design is synthesized on Quartus and demonstrated on the Altera DE10 board.

3. Specification

Table 1.1: Table for Specification

Signal	Direction	Size	Description
clk	input	1	Clock signal.
reset_l	input	1	Active low reset signal
done	input	1	Signal indicating the ‘sumitup’ module has finished calculation
sum_in	input	8	Input sum from the ‘sumitup’ module
sum_out	output	8	Output sum to be displayed

4. Results

In this segment, I will validate the accuracy ofmy design. Here’s a part discussing the methods of verification, in this case, using the Verilator library in C++ for testing and waveform visualization.

- **Verification Method**: To ensure the correctness of our design, we utilized the Verilator library in C++. Verilator is a free and open-source tool that translates Verilog to a cycle-accurate C++ model. This allows us to write testbenches in C++, which can be significantly faster and more flexible than traditional Verilog testbenches.
- **Testing**: We wrote a C++ testbench using Verilator that applies a series of inputs to the ‘downstream’ module and checks the output against the expected results. This helped us identify any discrepancies and correct them in our design.
- **Waveform Visualization**: Verilator also enabled us to generate Value Change Dump (VCD) files, which we visualized using a waveform viewer. This allowed us to see the timing of the signals in our design and verify that they behaved as expected.

By using Verilator for verification, we were able to thoroughly test our design and ensure its correctness. The waveforms provided a visual confirmation of the correct operation of our ‘downstream’ module, reinforcing the results of our tests. This approach proved to be an effective way to verify our design.

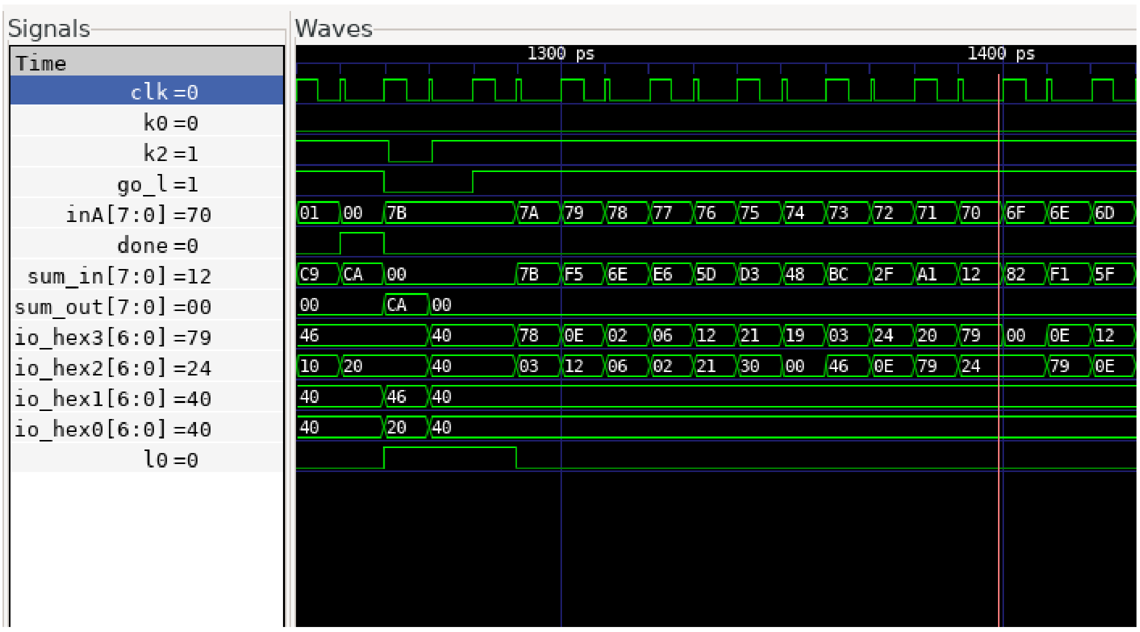

Figure 1.1: Waveform

- **Reset Operation (k2)**: When KEY2 is pressed, it acts as a reset button, initializing the display to zeros, turning off LEDR0, and resetting the Finite State Machine (FSM). This ensures that the system starts from a known state.
- **Start Operation (k0)**: KEY0 is used to start the operation. Upon pressing and holding KEY0, the hardware testbench sends a series of numbers to the **'sumitup'** module at . The sum of the numbers sent by the testbench is displayed on the upper hex displays (HEX3 and HEX2), while the result calculated by our code is displayed on the lower hex displays (HEX1 and HEX0). This result is the value captured by the 'downstream' module.
- **Verification of Results**: If the sum calculated by the testbench and our code match, LEDR0 lights up (l0 = 1), indicating the correctness of our design. When KEY0 is released, the hardware testbench zeros its hex displays, while our code continues to display the calculated sum in the lower digits, waiting for the next press of KEY0.
- **Continued Operation**: Upon pressing KEY0 again, a new series of numbers are sent and displayed, providing a new sum. Pressing KEY2 at any point will reset the system, zeroing all the displays.
- **Note**: The buttons are active low, meaning they present a logic 0 when pressed.

While the design has been tested and verified in a simulated environment using Verilator, it has not yet been tested on the actual Altera DE10 board. This is a crucial next step, as real-world testing can often reveal issues not caught in simulation due to the complexities of hardware-software interaction. Therefore, the next step in this project would be to synthesize the design onto the DE10 board and perform comprehensive testing to ensure the design works as expected in a real-world scenario. This will provide a more robust validation of the design. However, due to constraints such as time and resources, this testing could not be completed at this stage. It remains an important task for future work.

5. Discussion

While it’s true that the design operates as expected, it’s important to delve deeper into the evaluation and potential improvements. The ‘downstream’ module successfully captures the sum calculated by the ‘sumitup’ module and holds it for display, demonstrating the practical application of basic logic design and Finite State Machines (FSM) in FPGA programming.
However, perfection is a continuous journey and there are always areas for improvement. For instance, the current design operates at a fixed clock speed of 50 MHz. In future iterations, it could be beneficial to explore dynamic clock scaling techniques to optimize power consumption and performance based on the workload.

Additionally, the current verification method relies on the comparison of the sum calculated by the ‘sumitup’ module and the testbench. While this method is effective, it could be supplemented with formal verification techniques to prove the correctness of the design under all possible inputs and states.

Lastly, the user interface currently uses hex displays and LEDs on the Altera DE10 board. To enhance user experience, a more sophisticated interface could be developed, perhaps using an LCD display or even a web-based interface.

In conclusion, while the design works properly and meets the requirements, there are several potential enhancements that could be explored to strive for perfection. These improvements, however, could not be implemented due to constraints such as time and resources. Nevertheless, they provide a roadmap for future work and learning opportunities.