8x8 Signed Serial-Parallel Multiplier

Project Report for CSCE 2301 – Digital Design I

Marcelino Sedhum, Kirollos Mounir, Joshua Samuel
The American University in Cairo

May 21, 2025

Contents

1 Introduction

In modern digital systems, multiplication is a fundamental operation used in various domains such as computer arithmetic, digital signal processing, and embedded systems. The trade-off between speed and hardware complexity becomes critical when designing arithmetic units.

This project aimed to implement an 8x8 Signed Serial-Parallel Multiplier (SPM) using the Basys 3 FPGA development board. The SPM architecture processes one operand serially and the other in parallel, achieving a compromise between performance and resource usage.

Our design reads signed 8-bit binary inputs using switches, computes the signed product, and displays the result in decimal format on a 4-digit 7-segment display with scrolling capability.

2 System Overview

The 8x8 Signed SPM system is designed to accept two signed 8-bit inputs:

- Multiplier: SW[7:0]
- Multiplicand: SW[15:8]

Once the start button (BTNC) is pressed, the system initiates the multiplication process. The output is displayed on the 7-segment display in decimal format, with BTNL and BTNR enabling scrolling. LD0 LED signals the end of computation.

3 Block Diagram

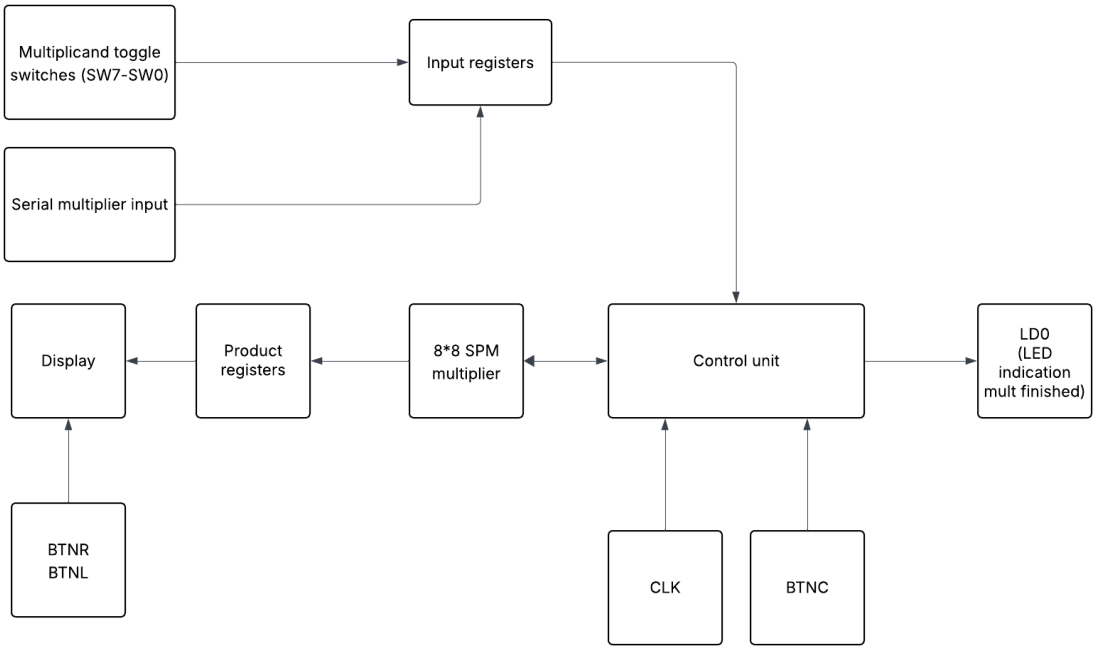

Figure 1: Top-level block diagram of the 8x8 Signed SPM system.

4 Design in Logisim Evolution

4.1 7-Segment Display Driver

We implemented a display driver module in Logisim that converts binary input into decimal output using a binary-to-BCD conversion and displays it on the 7-segment modules. BTNL and BTNR allow users to scroll through the digits.

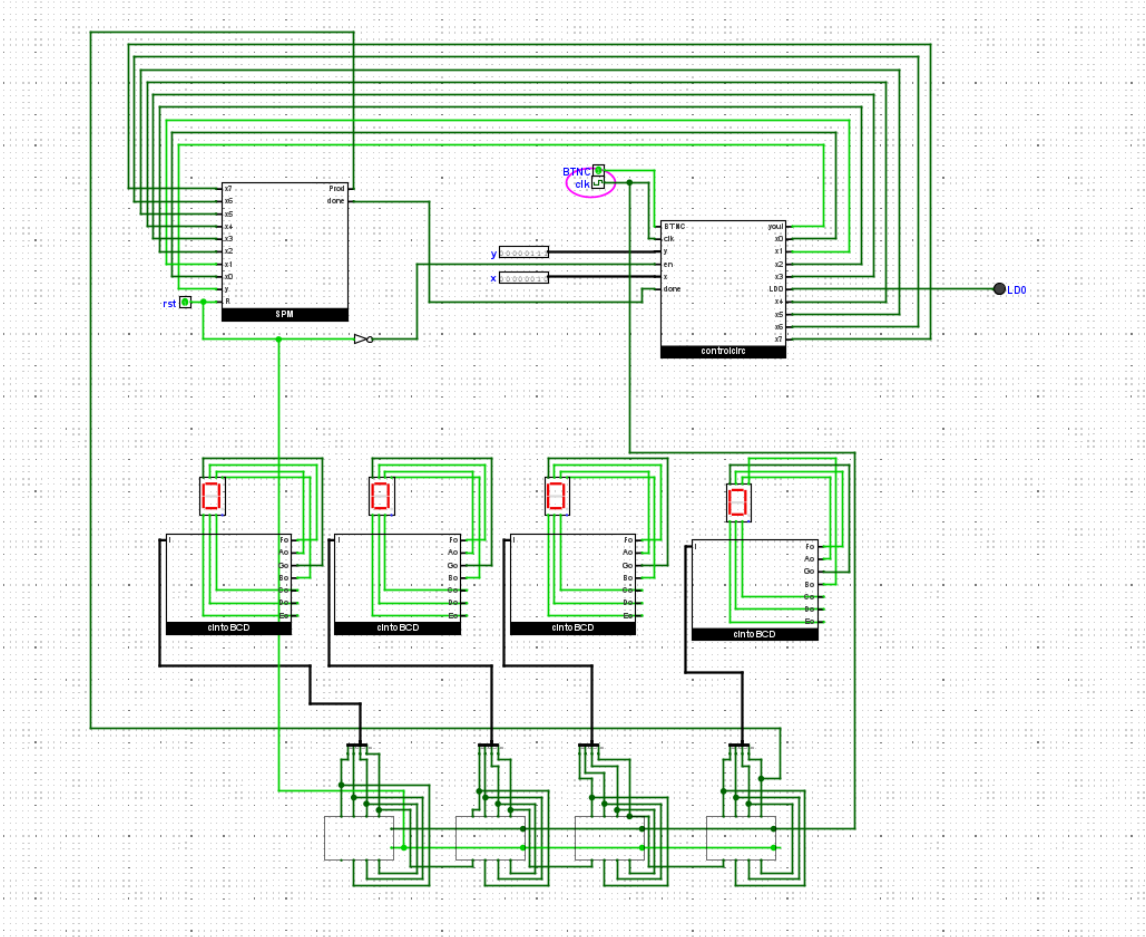

Figure 2: 7-Segment driver simulation in Logisim.

4.2 Serial-Parallel Multiplier

We simulated the SPM circuit using a finite state machine to control each multiplication step, handle signs correctly, and shift the partial product accordingly.

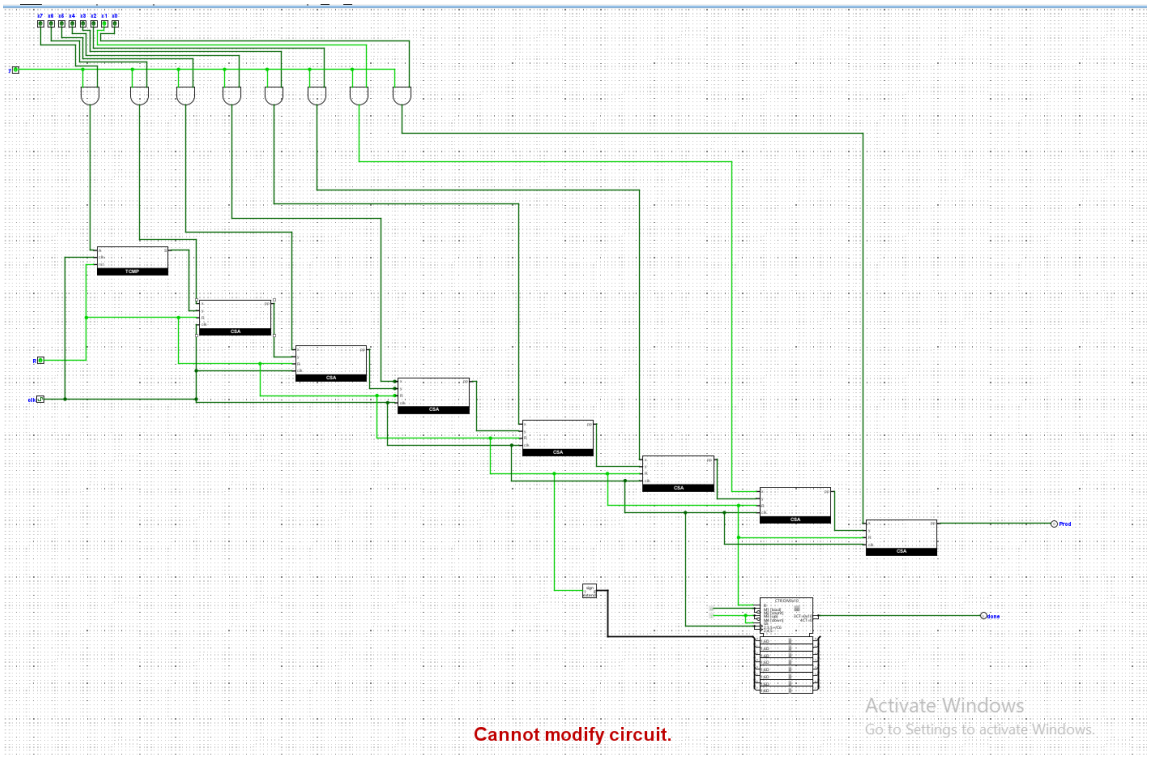

Figure 3: Serial-Parallel Multiplier simulation in Logisim.

5 Verilog Implementation

The full system was implemented in Verilog. It consists of several modular files:

- `bin_to_bcd.v` – Switch binary into binary coded decimal values.
- `helper_bsd_to_7seg.v` – transforms BCD into 7seg display using combinational.
- `serial_parallel_multiplier.v` – SPM logic.
- `seven_seg_display.v` – works the display.
- `top_control_unit.v` - top module integrating everything together.

6 FPGA Integration and Test

The design was synthesized and uploaded to the Basys 3 FPGA. Testing was done using different 8-bit signed inputs. Some of the test cases included:

- Positive × Positive
- Positive × Negative
- Negative × Negative

The BTNC button was used to trigger computation, BTNL/BTNR to scroll, and LD0 to indicate completion. Results were displayed correctly with sign and decimal digits.

7 Implementation Challenges

Throughout the development process, we faced several challenges:

- **Sign Extension:** Properly sign-extending negative numbers to 16 bits in a 2's complement system.
- **Display Overflow:** Managing products that exceed the display limit (up to 5 digits).
- **Operator Behavior:** Special care was required to make signed multiplication behave consistently across Logisim and Verilog due to differences in handling of the '+', '*', and '-' operators.
- **Clock Synchronization:** Ensuring button inputs and FSM transitions were synchronized with the FPGA's clock.

8 Validation

We validated the system by testing:

- Functional correctness: correct decimal output for all input combinations.
- Edge cases: multiplying with zero, with maximum and minimum signed 8-bit values.
- Scrolling operation: successfully displaying all 5 digits using BTNL and BTNR.

Simulation and on-board testing confirmed that the system performs correctly.

9 Conclusion

This project successfully demonstrated the implementation of an 8x8 Signed Serial-Parallel Multiplier on an FPGA. The system efficiently performs signed multiplication using a combination of serial-parallel processing, control logic, and display formatting.

Beyond meeting the functional requirements, the project reinforced our understanding of binary arithmetic, FSMs, BCD conversion, and hardware-software integration on FPGAs.