Project Report: One-Byte Digital Wallet

Name: Raman Tondro (40232106)

Date: 1/1/2026

Project: Digital Logic Design

1. Project Overview & Circuit Concept

The objective of this project was to design and simulate a **One-Byte Digital Wallet** capable of tracking shared expenses. The system operates on 8-bit binary values ranging from 0 to 255 and performs two primary operations: **Deposit (Addition)** and **Withdrawal (Subtraction)** based on a control signal. The core design principle relies on the **Ripple Carry Adder** architecture. To adhere to the "Clean Architecture" requirement and avoid using a separate subtraction circuit, the system utilizes **two's Complement arithmetic**. By manipulating the second operand and the initial carry bit, the same adder structure performs both mathematical operations efficiently. The system also includes a safety feature to detect invalid balances, such as overflows or negative results, by monitoring the final carry-bit of the calculation.

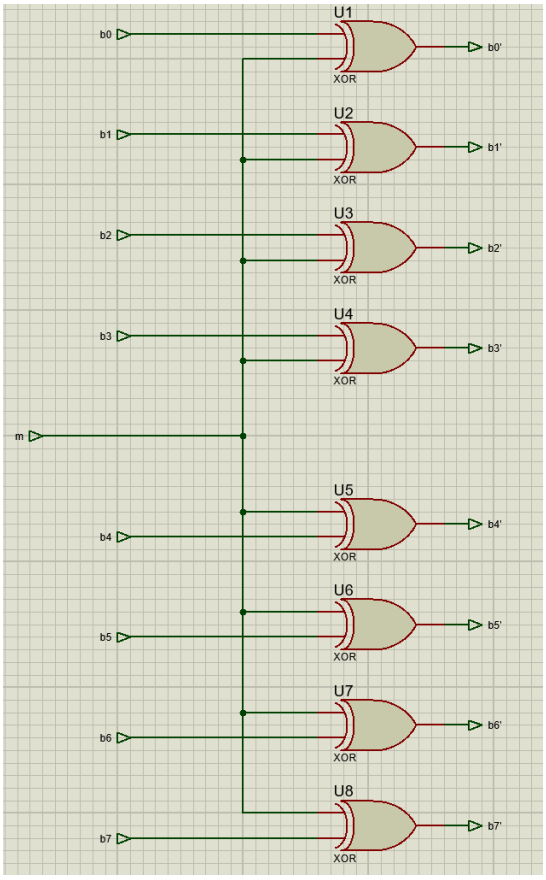
2. Block Diagram and Modular Design

To improve design quality (Bonus 2), the circuit is organized hierarchically using reusable child components (sub-circuits). This modular approach enhances clarity, maintainability, and scalability of the schematic, making it more professional and easier to debug. The main schematic integrates these components seamlessly, ensuring a clean and readable layout. The design emphasizes encapsulation of logic blocks such as the Input Conditioner and Full Adders, which are abstracted into individual modules, facilitating efficient testing and validation.

2.1. Sub-Circuit 1: The Input Conditioner (B XOR Mode)

This component prepares the transaction amount (B) for the adder. It takes the 8-bit input B and a 1-bit Mode selector. It performs a bitwise XOR operation on each bit of B against the Mode bit, allowing dynamic inversion of B when subtraction is required. This operation is crucial for implementing subtraction via two's complement arithmetic.

Schematic of Input Conditioner



The logic per bit is summarized in the following truth table:

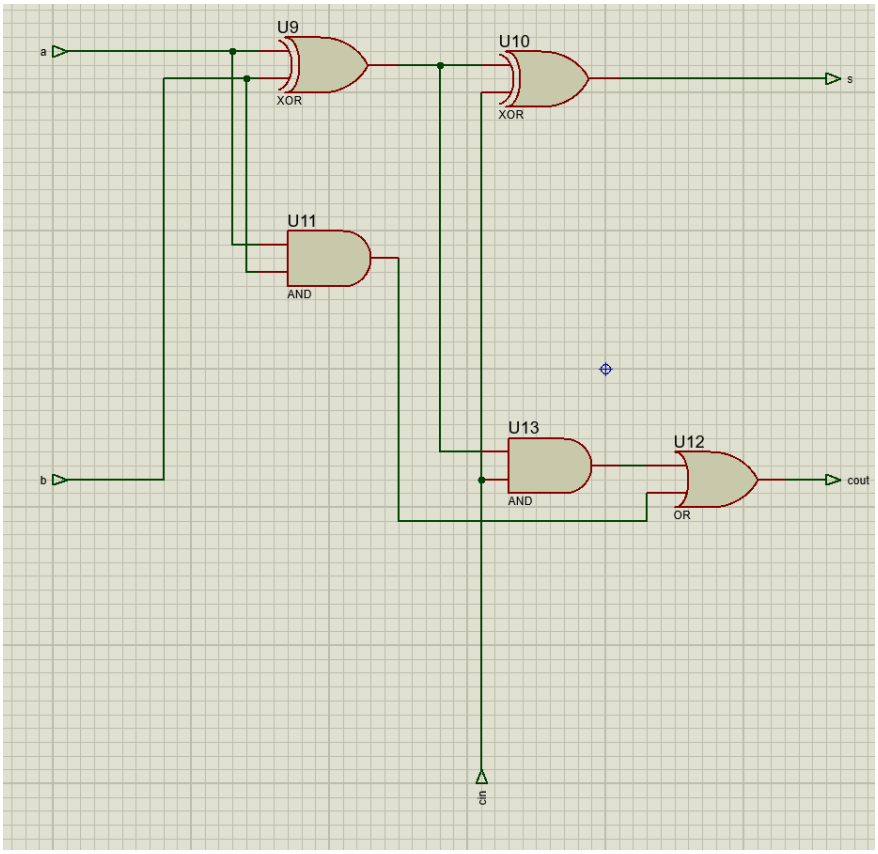
Input (B Bit)	Input (Mode)	Output (Processed Bit)	Logic Description
0	0	0	Pass-through (Mode 0 preserves B)
1	0	1	Pass-through (Mode 0 preserves B)
0	1	1	Invert (Mode 1 inverts B)
1	1	0	Invert (Mode 1 inverts B)

2.2. Sub-Circuit 2: The 1-Bit Full Adder

The arithmetic core is constructed by chaining eight 1-bit Full Adders. Each adder computes the Sum and Carry Out for its respective bit position, forming the basis of the 8-bit ripple carry adder.

The Full Adder's schematic and truth table are detailed below, illustrating the logic for each bit operation.

Schematic of Full Adder Component

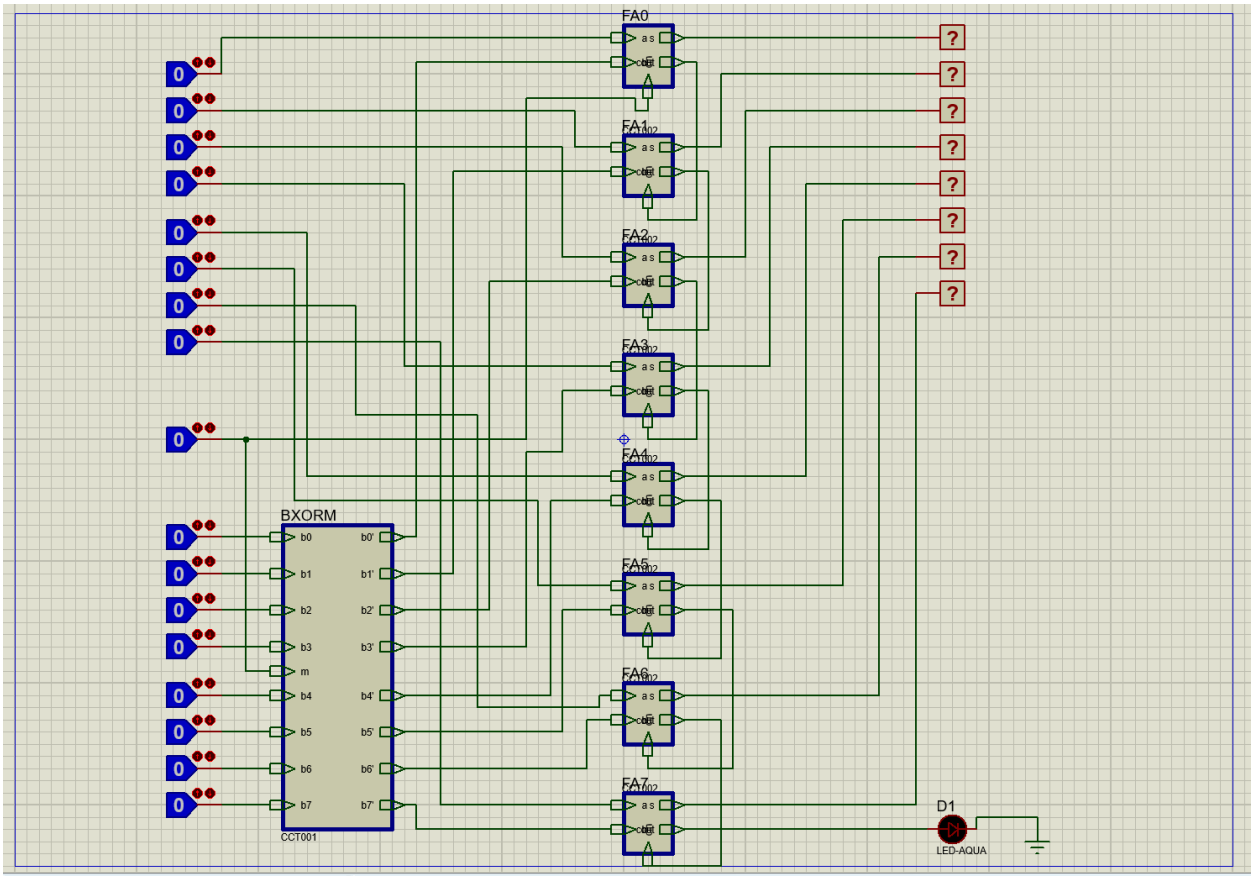


Truth Table (Full Adder)

A	B	Cin	Sum	Cout	State Description
0	0	0	0	0	No input
0	0	1	1	0	Carry In only
0	1	0	1	0	B only
0	1	1	0	1	B + Cin (Carry Generated)
1	0	0	1	0	A only
1	0	1	0	1	A + Cin (Carry Generated)
1	1	0	0	1	A + B (Carry Generated)
1	1	1	1	1	All High (Sum=1, Carry=1)

3. Final Circuit Implementation

The main circuit integrates the sub-circuits. The 8-bit inputs are processed through the conditioners and fed into the chain of Full Adders. The schematic layout ensures a logical flow from inputs to outputs, with clear demarcation of functional blocks.



3.1. Inputs and Outputs

- **Input A [7:0]:** Represents the current wallet balance.
- **Input B [7:0]:** Represents the amount to be deposited or withdrawn.
- **Input Mode (1 Bit):** The operation selector, where 0 indicates addition and 1 indicates subtraction.
- **Output Result [7:0]:** The new calculated balance after the operation.
- **Output LED (Warning):** Indicates if the balance is invalid, such as overflow or negative balance.

3.2. Handling Addition and Subtraction

The circuit employs a unified mathematical formula for both operations:

Result = A + (B XOR Mode) + Mode

- **For Addition (Mode = 0):**
- The XOR gate outputs B unchanged (B XOR 0 = B).
- **The initial Carry-In (C_{in}) is 0.**
- Calculation: A + B + 0.

- **For Subtraction (Mode = 1):**
- The XOR gate outputs the inverse of B (B XOR 1 = 1's complement of B).
- **The initial Carry-In (C_{in}) is 1.**
- Calculation: A + 1's complement of B + 1, which implements subtraction via two's complement.

4. Bonus Features

Bonus 1: Invalid Balance Warning

The wallet system includes a warning mechanism for invalid balances, such as exceeding 255 or dropping below 0. This is achieved by monitoring the final Carry Out (C_{out}) of the most significant bit (MSB). In addition, during subtraction, the Carry bit indicates whether a borrow has occurred, signaling an invalid or negative balance. The warning LED is connected to this final output, providing a visual alert for boundary conditions.

Bonus 2: Clean Architecture

The design emphasizes modularity by encapsulating logic into discrete components, such as Input Conditioners and Full Adders. This approach prevents "spaghetti wiring," simplifies debugging, and enhances scalability. The hierarchical structure facilitates easy updates and testing, ensuring the system remains robust and maintainable.

5. Test Cases and Validation

The system was validated through simulation in Proteus, covering various scenarios:

Test Case 1: Standard Deposit (Addition)

- **Inputs:** Balance (A) = 10, Amount (B) = 5, Mode = 0 (Add)
- **Process:** 10 + 5
- **Output:** Result = 15
- **LED Status:** OFF (Valid Balance)

Test Case 2: Standard Withdrawal (Subtraction)

- **Inputs:** Balance (A) = 20, Amount (B) = 5, Mode = 1 (Sub)
- **Process:** 20 - 5
- **Output:** Result = 15
- **LED Status:** OFF (Valid Balance)

Test Case 3: Wallet Overflow (Invalid Balance)

- **Inputs:** Balance (A) = 250, Amount (B) = 10, Mode = 0 (Add)
- **Process:** 250 + 10 = 260 (Exceeds 255)
- **Output:** Shows truncated value due to overflow.
- **LED Status:** ON (Indicates Overflow/Invalid State)

Test Case 4: Negative Balance (Invalid Balance)

- **Inputs:** Balance (A) = 10, Amount (B) = 20, Mode = 1 (Sub)
- **Process:** 10 - 20 = -10 (represented in 2's complement)
- **Output:** Shows negative value in 2's complement form.
- **LED Status:** ON (Indicates Invalid/Negative State)