Design 1: Binary-to-BCD Converter

Overview

The binary-to-BCD converter implements the "shift-and-add-3" algorithm to convert a 5-bit binary input to an 8-bit BCD output (4 bits for tens place, 4 bits for ones place).

Design Iterations

Iteration 1

Prompt:

I am trying to create a Verilog model binary_to_bcd_converter for a binary to binary-coded-decimal converter. It must meet the following specifications:

- Inputs: Binary input (5-bits)

- Outputs: BCD (8-bits: 4-bits for the 10's place and 4-bits for the 1's place)

How would I write a design that meets these specifications?

Response: The AI correctly suggested using the shift-and-add-3 algorithm but failed to implement proper left-shifting of the binary input.

Simulation Results:

- **Status:** FAILED
- **Issue:** 6th test case failed
- **GTKWave Analysis:** Failure occurred when input values exceeded 5, indicating improper bit extraction from binary input

Iteration 2

Refined Prompt:

I am trying to create a Verilog model binary_to_bcd_converter for a binary to binary-coded-decimal converter. It must meet the following specifications:

- Inputs: binary(5-bits)

- Outputs: bcd (8-bits: 4-bits for the 10's place and 4-bits for the 1's place)

So you will do shift and add 3 algorithm.

First you will take MSB of binary input and put it in LSB of bcd. Then left shift binary input by 1 bit.

Check for greater than 5.

Response: The AI introduced unnecessary padding and incorrect port naming. Failed to implement proper bit shifting in each iteration.

Simulation Results:

- **Status:** FAILED

- **Issue:** "Port doesn't exist in uut" - port naming mismatch with testbench

Iteration 3

Further Refined Prompt:

I am trying to create a Verilog model binary_to_bcd_converter for a binary to binary-coded-decimal converter. It must meet the following specifications:
- Inputs: binary_input (5-bits)
- Outputs: bcd_output (8-bits: 4-bits for the 10's place and 4-bits for the 1's place)

So you will do shift and add 3 algorithm.
In every iteration first you will take MSB of binary input and put it in LSB of bcd.
Then left shift binary input by 1 bit. Then Check both bcd digits for greater than 5.
No padding needed at start.

Response: Implemented incorrect algorithm sequence.

Simulation Results:

- **Status:** FAILED
- **Issue:** 1st test case failed due to wrong algorithm implementation

Iteration 4 (Final)

Optimized Prompt:

I am trying to create a Verilog module named `binary_to_bcd_converter` for a binary-to-BCD converter.
The module must meet the following specifications:
Inputs: binary_input: A 5-bit input.
Outputs: bcd_output: An 8-bit output, where the most significant 4 bits represent the 10's place and the least significant 4 bits represent the 1's place.

The design must implement the "shift-and-add-3" algorithm. The implementation should be sequential and follow these steps for five iterations (one for each bit of the binary_input):
1. Check and Adjust BCD Digits: Before the shift, check each 4-bit BCD digit. If a digit is greater than or equal to 5, add 3 to it.
2. Shift: Left-shift the combined BCD and binary register by one bit. The most significant bit of the binary input is shifted into the least significant bit of the BCD register.

Response: Generated perfect implementation with correct algorithm sequence.

Simulation Results:

- **Status:** PASSED
- **All test cases passed successfully**

Design 2: Shift Register

Overview

An 8-bit shift register with clock, active-low reset, data input, and shift enable control.

Design Process

Iteration 1 (Successful)

Prompt:

```
I am trying to create a Verilog model for a shift register. It must meet the following specifications:
- Inputs:
  - Clock
  - Active-low reset
  - Data (1 bit)
  - Shift enable
- Outputs:
  - Data (8 bits)
How would I write a design that meets these specifications?
```

Response: Generated correct implementation on first attempt.

Simulation Results:

- **Status:** PASSED
- **All test cases passed after correct testbench implementation**

Design 3: Linear Feedback Shift Register (LFSR)

Overview

An 8-bit LFSR with specified initial state (10001010) and tap positions at locations 1, 4, 6, and 7.

Design Process

Iteration 1

Prompt:

I am trying to create a Verilog model for an LFSR. It must meet the following specifications:
- Inputs:
  - Clock
  - Active-low reset
- Outputs:
  - Data (8-bits)

The initial state should be 10001010, and the taps should be at locations 1, 4, 6, and 7.

Response: Generated functionally correct code with minor port naming issues.

Simulation Results:

- **First Attempt:** FAILED - Port naming mismatch with testbench
- **After Port Correction:** PASSED

Results Summary

Design	Iterations Required	Primary Challenges	Final Status
Binary-to-BCD	4	Algorithm complexity, port naming, bit manipulation	PASSED
Shift Register	1	None (testbench issues only)	PASSED
LFSR	1	Minor port naming	PASSED

Conclusions:

• **Detailed Algorithm Specification:** Complex algorithms require step-by-step breakdown
• **Explicit Port Naming:** Specify exact port names to match testbenches
• **Clear Functional Requirements:** Unambiguous specification of desired behaviour.
• **Sequential Process Description:** For multi-step algorithms, describe each iteration