Lab 1B: Priority Encoder with Enable

Objective

The objective of this lab is to design and verify an **8-to-3 priority encoder** with an input enable signal. The encoder should identify the highest-priority active input and generate a 3-bit encoded output along with a valid flag. This experiment introduces the concept of **priority logic, input enable control, and handling don't-care conditions**.

Design Requirements

The encoder must meet the following requirements:

- **Inputs**:
    - 8-bit active-high input (`data_in[7:0]`)
    - Enable signal (`enable`)
- **Outputs**:
    - 3-bit encoded output (`encoded_out`)
    - Valid signal (`valid`)
- **Functionality**:
    - Input with the highest priority (MSB = `data_in[7]`) determines the output.
    - **Valid = 1** when any input is active; **Valid = 0** otherwise.
    - When **enable = 0**, the encoder is disabled and outputs are invalid.
    - Must handle the **all-zero input case** by setting valid = 0.

Design Methodology

1. **Truth Table**

    - A truth table was constructed covering all possible input cases.
    - For multiple active inputs, the **highest-order ‘1’ bit** was given priority.
    - For all inputs = 0, valid = 0 and encoded_out = 000.

2. **K-map Simplification**
    - Boolean expressions for the three output bits were derived using **Karnaugh maps**.
    - This optimization minimized gate usage for FPGA implementation.

3. **Casez Optimization**
    - A `casez` statement was chosen in SystemVerilog to simplify don’t-care handling.
    - This made the design concise and efficient for priority encoding.

4. **Enable Handling**
    - When enable = 0, the outputs default to encoded_out = 000 and valid = 0.
    - This ensures predictable behavior in disabled mode.

Simulation and Verification

Simulation Setup

- A testbench applied different input vectors to verify all conditions.
- Both single-bit active inputs and multiple active inputs were tested.
- The enable signal was toggled to test encoder activation and deactivation.

Observed Results

1. **Enable = 0**: Encoder outputs were disabled regardless of input values.
2. **Single Active Input**: Correct encoded value was generated for all eight inputs.

    - Example: `data_in = 00010000 → encoded_out = 100`.

3. **Multiple Active Inputs**: Encoder always prioritized the **highest-order 1**.
    - Example: `data_in = 01010101 → encoded_out = 110`.
4. **All-Zero Input**: Output was 000 with valid = 0.
5. **Valid Flag**: Correctly asserted only when at least one input was active.

Results and Discussion

- The **8-to-3 priority encoder** functioned as expected in all tested scenarios.
- The **enable signal** effectively controlled encoder activation.
- The **valid output** provided a useful status signal for downstream circuits.
- Using **casez** simplified the logic and reduced FPGA resource usage.
- The design can be extended to larger encoders (e.g., 16-to-4) by cascading smaller blocks.

Conclusion

In this lab, an **8-to-3 priority encoder with enable** was designed, implemented, and simulated. The encoder successfully handled single and multiple input cases, correctly prioritized higher-order inputs, and provided a valid output indicator. The design demonstrated the principles of **priority logic, don't-care optimization, and enable-controlled functionality**, making it suitable for FPGA-based digital systems.