Lab 2A: 32-bit Barrel Shifter

Objective

The objective of this lab is to design and verify a **32-bit barrel shifter** capable of performing left/right shifts as well as left/right rotations in a single cycle. The design should support variable shift amounts ranging from 0 to 31 and must be optimized for speed and hardware efficiency.

Design Requirements

- **Data Width**: 32-bit input and 32-bit output.
- **Shift Amount**: 5-bit control input to select the shift distance (0–31).
- **Direction Control**:
    - Left shift/rotate.
    - Right shift/rotate.
- **Mode Control**:
    - Shift mode (vacant positions filled with zeros).
    - Rotate mode (bits wrapped around).
- **Performance**: Single-cycle operation.

Design Methodology

1 Barrel Shifter Architecture

- The design is implemented as a **multi-stage shifter** using a series of multiplexers.
- Each stage shifts by a power of two (1, 2, 4, 8, 16) depending on the corresponding bit in the shift amount.

- The final output is obtained after passing through all stages, ensuring any shift from 0–31 positions can be achieved in a single cycle.

2 Control Signals

- **left_right**: Determines the direction of the shift/rotation.
    - `0` = Left
    - `1` = Right
- **shift_rotate**: Determines the mode of operation.
    - `0` = Logical Shift
    - `1` = Rotate

3 Optimization

- Each stage uses multiplexers to minimize combinational delay.
- The datapath structure is balanced to ensure maximum performance with minimal logic depth.
- FPGA routing resources are considered by limiting fan-out and reusing intermediate stage signals.

4 Testbench Methodology

- A **self-checking testbench** was developed to automatically compare the DUT output against expected results.
- Test cases included:
    1. **Shift Left** by different amounts.
    2. **Shift Right** by different amounts.
    3. **Rotate Left** by different amounts.

4. **Rotate Right** by different amounts.
5. Edge cases such as shift by 0, shift by 31, and large rotations.

Results

1 Functional Verification

- The barrel shifter successfully passed all test cases.
- **Shift Left**: Correctly inserted zeros on the right and shifted bits left.
- **Shift Right**: Correctly inserted zeros on the left and shifted bits right.
- **Rotate Left**: Wrapped the left-shifted bits around to the right end.
- **Rotate Right**: Wrapped the right-shifted bits around to the left end.

2 Observed Outputs (Sample Cases)

- Input: `0xA5A5A5A5`, Shift Left by 8 → Output: `0xA5A5A500`
- Input: `0xA5A5A5A5`, Shift Right by 8 → Output: `0x00A5A5A5`
- Input: `0xDEADBEEF`, Rotate Left by 8 → Output: `0xADBEEFDE`
- Input: `0xDEADBEEF`, Rotate Right by 8 → Output: `0xEFDEADBE`

3 Performance

- The design achieved **single-cycle operation**, as expected.
- The self-checking testbench confirmed that outputs matched the expected results for all tested scenarios.

Observations

- The barrel shifter enables **constant-time shifts/rotations**, unlike sequential shifters which require multiple cycles.

- Multiplexer-based design ensures **low latency**, but requires more logic resources compared to simple shifters.

- The ability to rotate as well as shift increases the flexibility of the unit, making it suitable for applications such as CPU instruction sets.

Conclusion

The 32-bit barrel shifter was successfully designed, implemented, and tested. The design met all functional and performance requirements, supporting variable left/right shifts and rotations within a single cycle. Simulation results confirmed correctness across all test cases. This design approach demonstrates how multiplexer-based architectures can be used to achieve efficient, high-speed shifting and rotation operations in digital systems.