Lab 8: UART Controller

UART Receiver

Objective

The objective of this lab is to design, implement, and test a UART (Universal Asynchronous Receiver/Transmitter) receiver module. The design must handle start bit detection, data sampling at optimal instants, error detection, and data buffering using a FIFO structure.

Design Requirements

- **Functionality**:
    - Detect start bit and validate frame structure.
    - Correctly sample incoming serial data using 16× oversampling.
    - Receive and store 8-bit data frames.
    - Validate stop bit and detect frame errors.
    - Provide valid received data to the system.
    - Store received data in a FIFO for reliable communication.
- **Specifications**:
    - Baud rate: **115200 bps**
    - System clock: **50 MHz**
    - Data bits: **8 bits (LSB first)**
    - Oversampling: **16 samples per bit**
    - FIFO depth: **16 entries** with *almost full* and *almost empty* indicators

Methodology

1 Baud Rate Generator

- Generates timing ticks for both transmission and reception.
- For reception, oversampling is achieved by dividing the system clock to produce 16 ticks per bit duration.
- This ensures reliable mid-bit sampling and error detection.

2 Receiver Controller (FSM)

- Implements a finite state machine (FSM) with states:
    - **IDLE** – Wait for the start bit (line goes low).
    - **START** – Validate start bit after half-bit delay.
    - **DATA** – Shift in 8 bits, sampling each at the center.
    - **STOP** – Verify stop bit and detect frame errors.
- Outputs include control signals for shifting data, validating frames, and FIFO write enable.

3 Receiver Datapath

- Shift register collects serial input bits at each data sampling tick.
- Once a full byte is received, it is latched into an output register.
- Also handles parity bit collection for error checking (if enabled).

4 FIFO Buffer

- Stores received bytes before they are read by the processor/system.
- Implements standard FIFO operations:

- **Write Enable** – stores new valid data.
- **Read Enable** – outputs stored data.
- **Full/Empty Flags** – indicate FIFO status.
- **Almost Full/Almost Empty** thresholds – improve system response.

5 Top Module Integration

- Connects the baud rate generator, FSM controller, datapath, and FIFO.
- Provides the system with:
  - Received data
  - Frame error indication
  - FIFO status signals

6 Testbench

- Generates clock and reset.
- Simulates data transmission at correct baud rate.
- Test cases:
  1. Receive ASCII character `'A'` (0x41).
  2. Receive ASCII character `'U'` (0x55).
  3. Simulate frame error by forcing stop bit low.

Results

- The UART receiver correctly received and reconstructed transmitted characters.
- **Case 1**: Received `'A'` (0x41) successfully.

- **Case 2**: Received `'U'` (0x55) successfully.
- **Case 3**: Frame error was successfully detected when the stop bit was invalid.
- FIFO stored the received data and provided flow control using full/empty flags.

Observations

- Oversampling (16×) ensures reliable bit sampling and reduces error risk due to clock mismatch.
- FIFO buffering decouples data reception from data processing, allowing robust communication.
- Frame error detection is essential for communication reliability.
- Proper synchronization between FSM, datapath, and FIFO is critical for correct operation.

7. Conclusion

In this lab, a UART receiver was successfully designed and tested. The design used a baud rate generator, FSM controller, datapath, and FIFO buffer to handle serial data reception. The implementation demonstrated reliable reception of characters, correct error detection, and effective data buffering. This design can be extended to include parity error detection, variable baud rates, and integration with a complete UART transceiver.