Serial Data Receiver IP - Micro-Architecture Specification

Table of Contents

Section	Title	Page
1	Document Control	3
2	Introduction	3
3	Contents Overview	4
4	High-Level IP Overview	4
5	IP Input/Output Ports	5
6	Internal Sub-Block Descriptions	6
7	Data Path Details	7
8	Control Path Details	7
9	Pipeline & Timing	8
10	Clocking & Reset	8
11	Registers & Configuration	9
12	Performance	9

13	Error Correction / Security / Safety	10
14	Advanced Debug & QoS	10
15	Parameterization & Configurability	10
16	Error Handling & Debug Hooks	10
17	Example Scenarios & Waveforms	11
18	Implementation & RTL Notes	11
19	Limitations & Future Extensions	12
20	Revision History	12
21	Errata	12

1. Document Control

1.1 Title / ID

Serial Data Receiver IP - Micro-Architecture Specification

Doc ID: MAS-SERIAL-RX-001

1.2 Version / Revision

Version 1.0

1.3 Owner / Authors

Shashank, Micro-Architecture Team

1.4 Approvals

Jane (Design Lead), Ana (Verification Lead)

1.5 Distribution / Confidentiality

Internal Use Only – Confidential

2. Introduction

2.1 Purpose and Scope

This document provides the detailed micro-architecture specification for a serial data receiver IP core. The IP is designed to receive data serially in a LSB-first format, validate it using a start bit, odd parity checking, and a stop bit, and then output a byte of received data along with a done flag. This specification outlines its internal FSM, datapath, control logic, and timing behavior. It covers only the reception path and does not include transmission or higher-level protocol handling.

2.2 References

- UART Protocol Overview

- Architecture Specification

2.3 Document Organization

The document begins with a high-level overview, progresses through I/O interface details, internal design components, timing and control flow, and concludes with implementation notes and future considerations.

3. Contents Overview

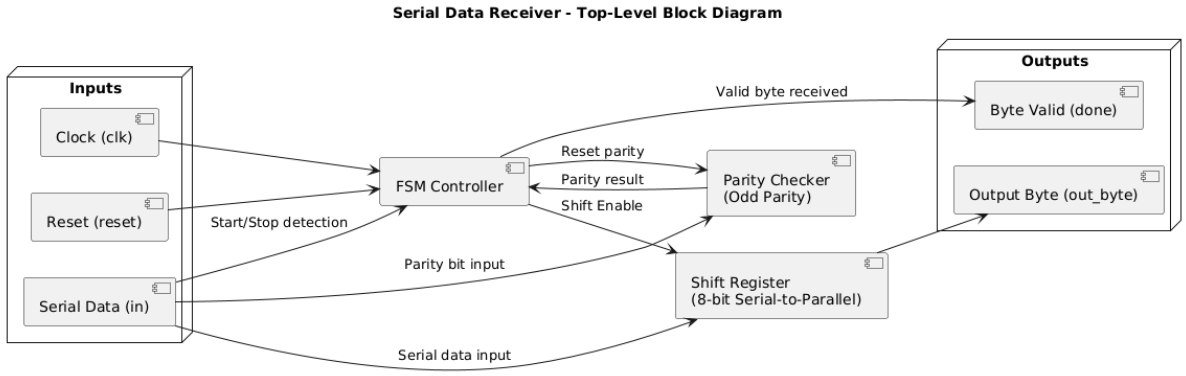
3.1 Design Goals

- Simple and robust serial data reception with start, parity, and stop bit verification.
- Synchronous design targeting single-clock domains.
- Minimum resource usage suitable for lightweight embedded applications.

3.2 Key Features

- Finite State Machine (FSM) based control logic.
- 8-bit serial-to-parallel shift register.
- Parity checker for error detection.
- LSB-first data reception protocol.
- Byte output with one-cycle valid signal.

4. High-Level IP Overview



4.1 Key Features

- Serial data reception with start, data, parity, and stop bit verification.
- FSM-based byte reception and error detection.
- Odd parity check for received data.

- LSB-first serial protocol support.
- Outputs received byte and a "done" signal for valid transactions.

4.2 Top-Level Block Diagram

The IP comprises the following primary blocks:

- **FSM (Finite State Machine)**: Orchestrates the reception logic.
- **Shift Register**: Captures incoming serial bits and forms the 8-bit output byte.
- **Parity Checker**: Validates odd parity for data integrity.

4.3 Module / File Hierarchy

- top_module.sv: Implements the serial receiver including FSM and control logic.
- parity.sv: Contains the parity checking logic.

4.4 Design Constraints

- Operates under a single clock domain.
- Requires detection of start bit (0) and validation of stop bit (1).
- Parity bit must maintain odd parity for data to be considered valid.

5. IP Input/Output Ports

5.1 Function

Facilitates serial reception and outputs validated byte with status flag.

5.2 Key Interfaces

- **Inputs**:
  - clk (1-bit): System clock.
  - reset (1-bit): Active-high synchronous reset.
  - in (1-bit): Serial data input.
- **Outputs**:
  - out_byte (8-bit): Parallel byte output after successful reception.
  - done (1-bit): Indicates a complete and valid data byte.

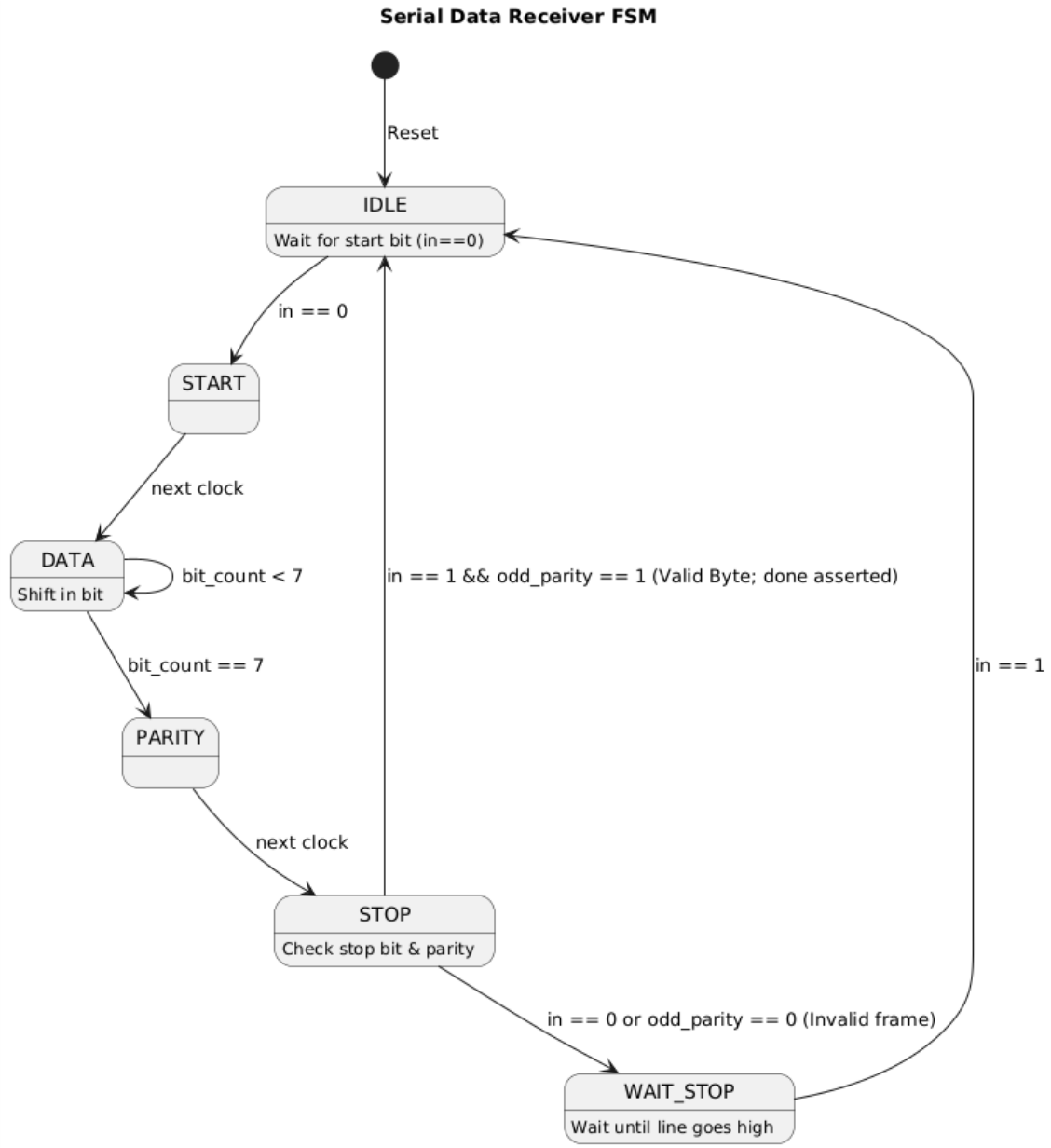
5.3 Parameterization

- No compile-time parameters used in current version, but extension-ready.

5.4 Timing

- Data is captured over 10 cycles (1 start + 8 data + 1 parity + 1 stop).
- Done signal is asserted for a single clock cycle on successful reception.

6. Internal Sub-Block Descriptions



6.1 FSM (Finite State Machine)

- **Function**: Manages all control logic for serial reception.
- **Inputs**: clk, reset, in
- **Outputs**: done, out_byte

- **Implementation**: 6-state FSM with sequential logic.
- **States**:
    - IDLE: Waits for start bit.
    - START: Validates start bit.
    - DATA: Shifts 8 bits.
    - PARITY: Verifies parity.
    - STOP: Confirms stop bit.
    - WAIT_STOP: Output stage.

6.2 Shift Register

- **Function**: Captures serial data into 8-bit format.
- **Inputs**: clk, reset, in
- **Output**: data_out (8-bit)
- **Implementation**: 8-bit left-shifting register.

6.3 Parity Checker

- **Function**: Verifies odd parity across data and parity bit.
- **Inputs**: clk, reset, in
- **Output**: odd (1-bit)
- **Implementation**: XOR toggle to track bit count.

7. Data Path Details

7.1 Data Flow

- Serial bit enters through in.
- FSM routes bits to shift register during DATA state.
- Parity bit enters checker in PARITY state.
- out_byte latched when STOP bit is verified.

7.2 Bit Widths & Encodings

- in: 1-bit serial input
- out_byte: 8-bit parallel output

7.3 Buffering / Queues

- No FIFO or queues; direct serial-to-parallel conversion.

8. Control Path Details

8.1 Status / Debug

- FSM state output can be used for debug.
- Error flags are generated internally (not exposed via ports).

8.2 Global Control FSM

- One global FSM governs all reception and validation steps.

8.3 Error / Security Handling

- If parity check fails or stop bit is not 1, data is discarded.

9. Pipeline & Timing

9.1 Pipeline Stage Breakdown

- FSM stages control flow sequentially, no true pipeline.

9.2 Latency

- Nominal: 10 cycles from start to done signal.

9.3 Hazards & Ordering

- Serial process avoids hazards through strict sequencing.

9.4 Multi-Power or Voltage Domains

- Not applicable.

10. Clocking & Reset

10.1 Clock Domains

- Single synchronous clock.

10.2 Reset Types

- Active-high, synchronous reset.

10.3 Reset / Startup Sequence

- FSM enters IDLE on reset and waits for a start bit.

11. Registers & Configuration

11.1 Configuration

- No runtime-configurable registers are present.

11.2 Register Map

- Not applicable.

12. Performance

12.1 Throughput / Bandwidth

- Processes 1 byte every 10 cycles.

12.2 Latency

- Fixed latency: 10 cycles.

12.3 Resource Utilization

- Estimated low: 1 FSM, 1 shift register, 1 parity checker.

12.4 Scalability

- Easily extensible by parameterizing data width.

13. Error Correction / Security / Safety

13.1 ECC or Data Integrity

- Uses odd parity for integrity checks.

13.2 Security Model

- Basic integrity, no encryption or key handling.

13.3 Functional Safety

- Discards corrupted data based on parity or stop bit failure.

14. Advanced Debug & QoS

14.1 Debug / Trace

- FSM state and done signals can be observed.

14.2 QoS / Scheduling

- Not applicable for single-stream receiver.

15. Parameterization & Configurability

- DATA_WIDTH: Default 8 bits.
- PARITY_TYPE: Currently hardcoded to odd.

16. Error Handling & Debug Hooks

16.1 Assertions / Monitors

- FSM transitions are trackable for test coverage.

16.2 Error Reporting

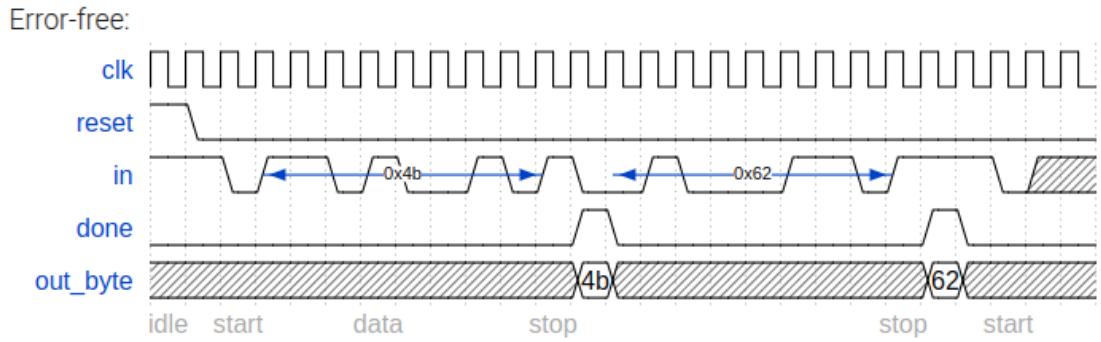
- Internal flagging of parity and stop errors.

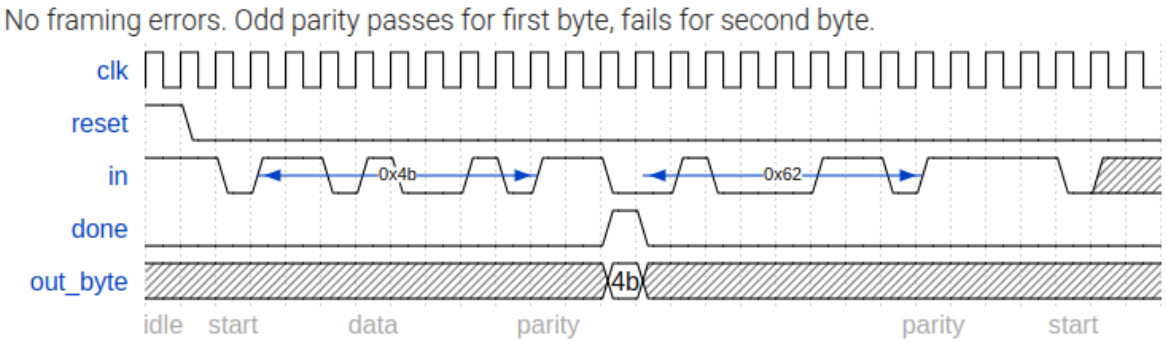
16.3 Integration with System Debug

- Debug hooks can connect to system monitor buses.

17. Example Scenarios & Waveforms

- **Valid Reception**: Start → 8 data bits → Parity → Stop → done.
- **Stop Bit Error**: Stop bit != 1 → byte discarded.
- **Parity Error**: Odd parity invalid → byte discarded.





18. Implementation & RTL Notes

18.1 Coding Style & Synthesis

- RTL is modular, synthesizable.

18.2 Power Optimization

- Not implemented in current version.

18.3 DFT & Test

- Scan chains and BIST not included.

19. Limitations & Future Extensions

Known Limitations

- No FIFO or streaming mode.
- No parity type selection.

Potential Future Work

- Add configurable parity.
- Extend to 9-bit UART.
- Add interrupt-based notification.

20. Revision History

Version	Date	Author	Changes
1.0	2025-04-15	Shashank	Initial version

21. Errata

- None known at this time.