Lab 22-1 Verifying a Serial Interface Receiver

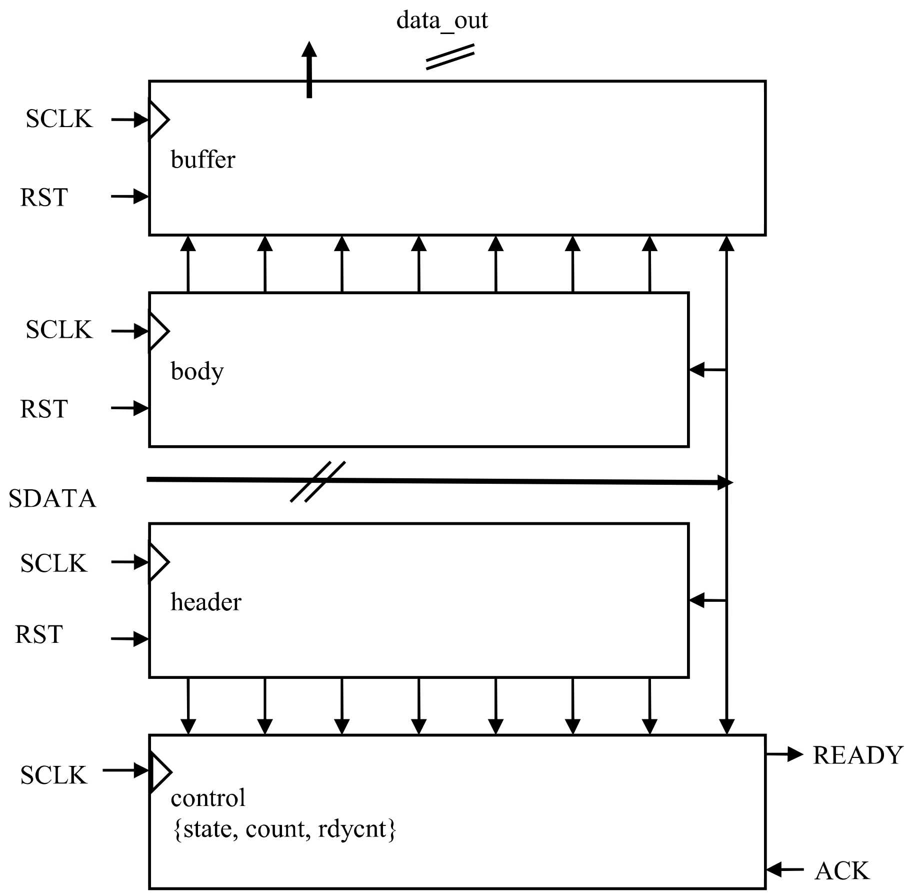
Objective: To test a serial interface receiver.

The lab model is a serial interface receiver. The receiver interfaces between a serial input stream and a parallel output stream.

In this lab, read the specification first and then follow the instructions. Please make sure to use the same variable name as the ones shown in block diagrams.

Specifications

- The receiver operates on the positive edge of the serial clock **SCLK** and has a synchronous active-high reset **RST**.
- A 24-bit packet represents the data.
- The packet consists of an 8-bit header with a value of 0xa5 followed by two 8-bit data bytes.
- The receiver initially shifts input data left into the header register until it detects the header value.
- For this reason, the header register must be initialized to a value opposite the leftmost header value bit.
- Upon detecting the packet header, the receiver clears the header register and shifts input data left into the body register while counting to 16.
- Upon the 16th count, the receiver moves the data to the output buffer, clears the counter, asserts the ready output, and again shifts input data left into the header register. The receiver can thus move a packet every 24 clocks.



Designing the Finite State Machine

1. The test environment reads the leftmost buffer byte while acknowledging the ready signal. The receiver shifts the output buffer left by one byte upon each such acknowledgment. The receiver counts acknowledge and drop the ready signal upon the last acknowledge. The test environment can delay the last acknowledge up to and including the clock that loads the next data into the output buffer. Failure of the test environment to retrieve data within that interval results in lost data.

2. In this lab, you have to generate a simple test of the serial interface receiver, which generates random data for four packets.

3. You statically construct a 256-bit stream containing four valid packets that are surrounded by values other than the header value.

4. Define HEADER_SIZE, BODY_SIZE and HEADER_VALUE as parameters with values 8, 16, and 8`ha5, respectively.

5. You reset the receiver and give this stream as an input to the receiver inputs. While adhering to the output protocol, you retrieve the receiver output data and verify that all packets are received and correct.

Verifying the Finite State Machine Design

1. Change to the *lab23-rcvr* directory and examine the following files.

rcvr.v	DUT

2. Simulate the design and test using the following commands with Xcelium.

```
xrun rcvr.v your_test_file_name -access +rwc
```

You should see the following result if simulation was successful.

```
121ns: Rcvd data 3524
201ns: Rcvd data 5e81
377ns: Rcvd data d609
489ns: Rcvd data 5663
489ns: Response process complete
517ns: Stimulus process complete
517ns: frames_sent=4, frames_rcvd=4
TEST PASSED
```

3. Correct your design as needed.

End of Lab