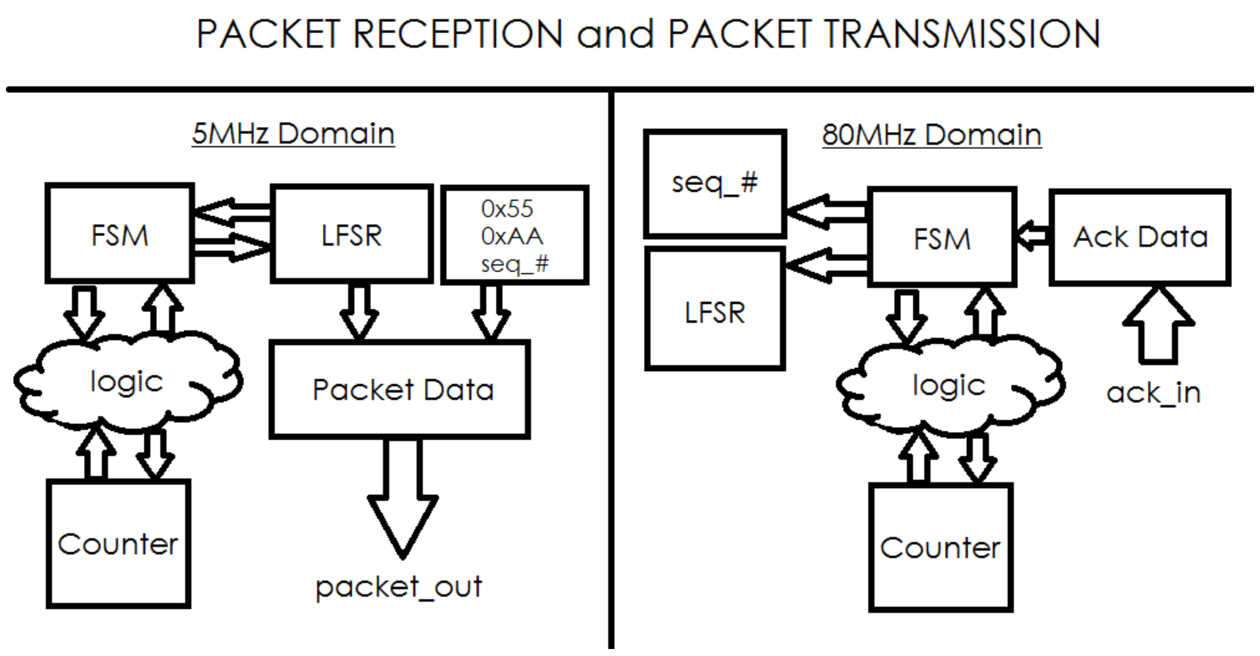
Diagrams



Design Structure

Both of the sub-systems are constructed the same way. In the receiver sub-system, "packet data" and "ack data" corresponds to the UART's transmitter and receiver respectively, each containing a FIFO to synchronize the data. Only difference between the sub-systems is rather than receiving messages, it is receiving acknowledgements and same concept holds for sending.

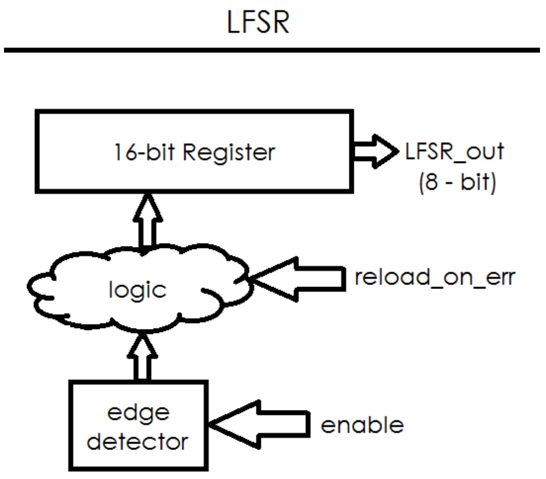
Data Flow

The transmission sub-system generates and begins sending the heading for the message (i.e. 0x55, 0xAA, seq. #). Then, it saves the LFSR seed in case it needs to be reloaded later due to error and makes requests to LFSR to create new 8-bit random numbers for any subsequent bytes.

The receiver sub-system waits for any incoming messages and keeps track of the LFSR data by syncing requests on its side. If any error was found, the acknowledgement is not sent because the transmission sub-system times out either way. Upon success, the sequence number increases on either side and the design is ready to send the next packet.

Bugs and Problems

When an acknowledgement is sent from the receiver sub-system, it does not assert that the other side received it. To fix this, I propose adding an assertion before receiving a message to indicate the acknowledgement has been received. I have not had luck implementing that in time.



Generation

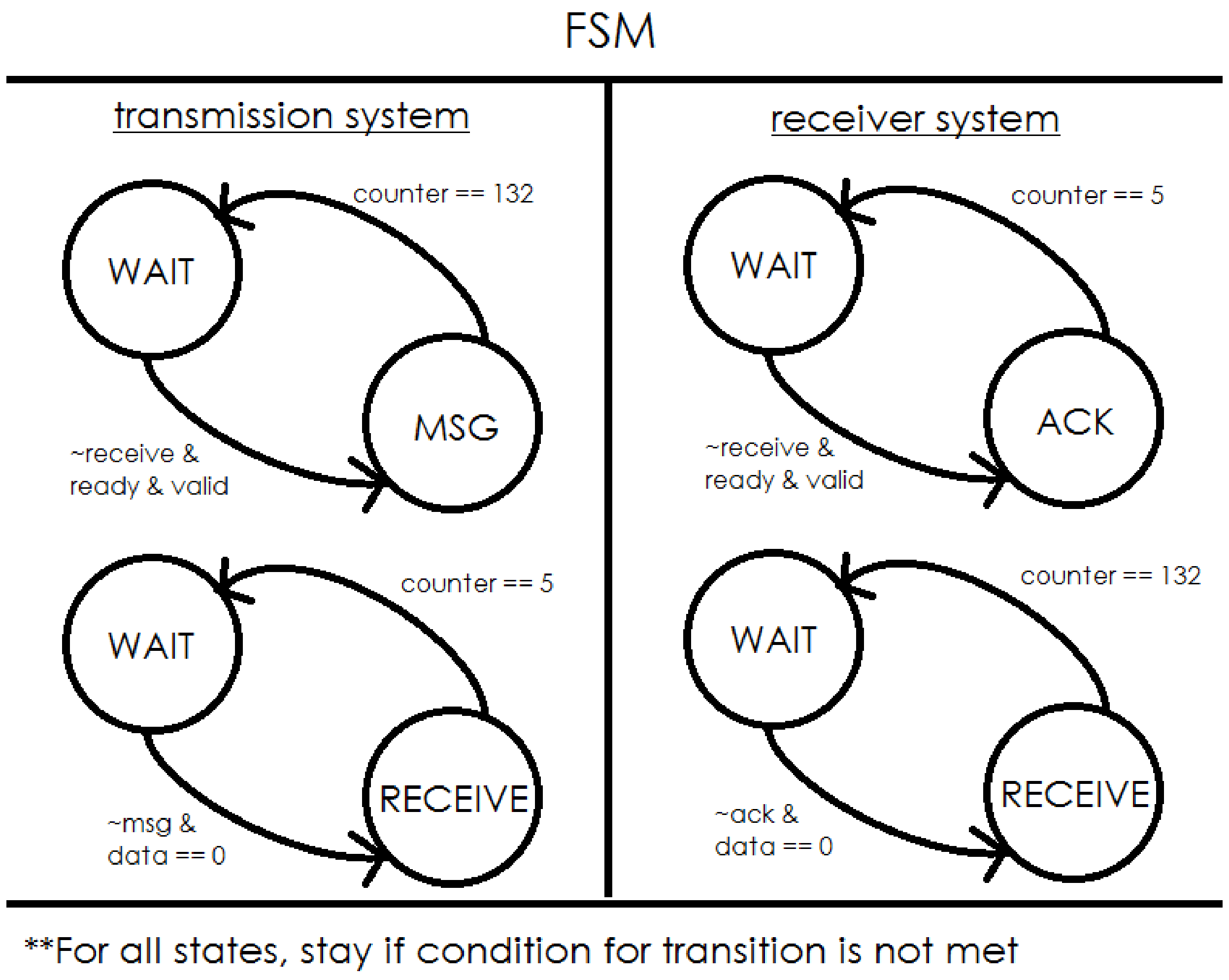
The LFSR propagates to generate 8 random bits each time by request and is initialized 0xABCD to ensure that it syncs ups with the other side.

Reload

In case of errors or an absence of acknowledgement, the LFSR restores its previous states saved in the sub-systems.

Enable

The edge detector is simply a mechanism for signal purposes.

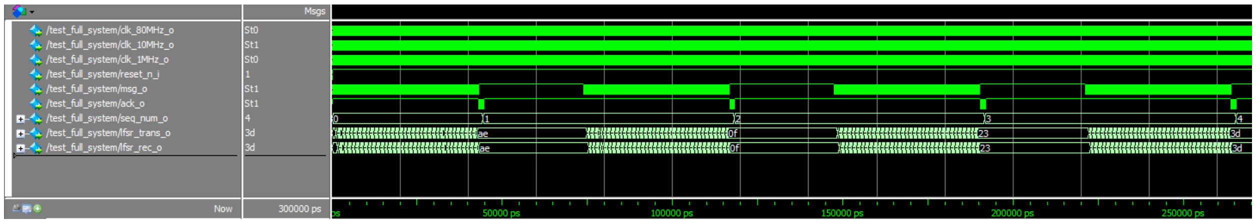


Since all FSM take turns making transitions because of the way data flows among them and each transition is given ample time to settle, the issue with cross domain transition should be non-existent.

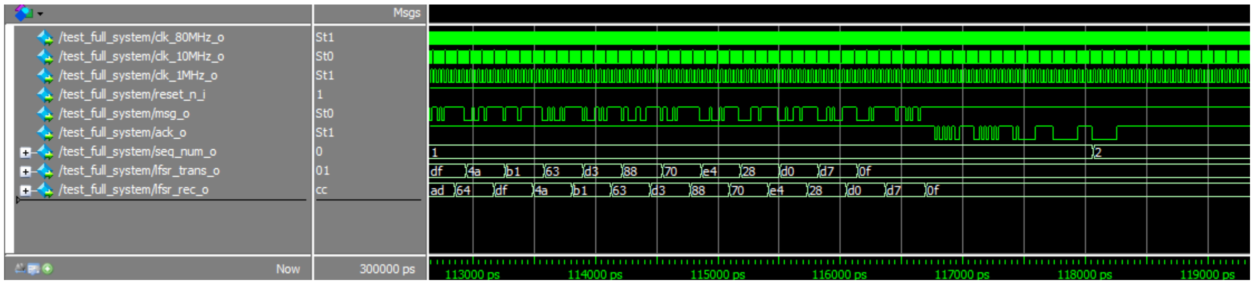
Simulation

NOTE: Please zoom to see details

Full System

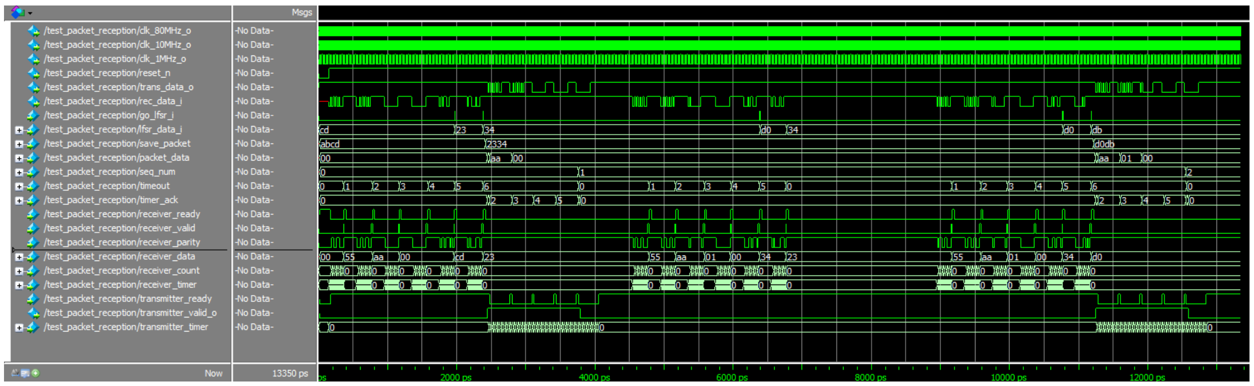

/test_full_system/clk_80MHz_o
/test_full_system/clk_10MHz_o
/test_full_system/clk_1MHz_o
/test_full_system/reset_n_i
/test_full_system/msg_o
/test_full_system/ack_o
/test_full_system/seq_num_o
/test_full_system/lfsr_trans_o
/test_full_system/lfsr_rec_o
Msgs
Now
300000 ps

1. Packets of 128-byte data are being sent and sequence number increases


/test_full_system/clk_80MHz_o
/test_full_system/clk_10MHz_o
/test_full_system/clk_1MHz_o
/test_full_system/reset_n_i
/test_full_system/msg_o
/test_full_system/ack_o
/test_full_system/seq_num_o
/test_full_system/lfsr_trans_o
/test_full_system/lfsr_rec_o
Msgs
Now
300000 ps

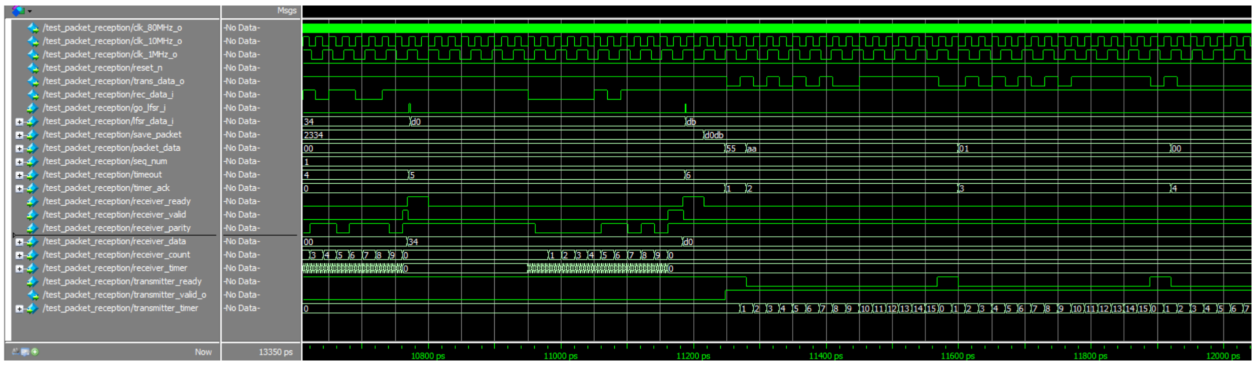
2. *LFSR data syncs across sub-systems with details of messages and acknowledgements*

Packet Reception Sub-system


/test_packet_reception/clk_80MHz_o
/test_packet_reception/clk_10MHz_o
/test_packet_reception/clk_1MHz_o
/test_packet_reception/reset_n
/test_packet_reception/trans_data_o
/test_packet_reception/rec_data_i
/test_packet_reception/go_lfsr_i
/test_packet_reception/lfsr_data_i
/test_packet_reception/save_packet
/test_packet_reception/packet_data
/test_packet_reception/seq_num
/test_packet_reception/timeout
/test_packet_reception/timer_ack
/test_packet_reception/receiver_ready
/test_packet_reception/receiver_valid
/test_packet_reception/receiver_parity
/test_packet_reception/receiver_data
/test_packet_reception/receiver_count
/test_packet_reception/receiver_timer
/test_packet_reception/transmitter_ready
/test_packet_reception/transmitter_valid_o
/test_packet_reception/transmitter_timer
Msgs
Now
13350 ps

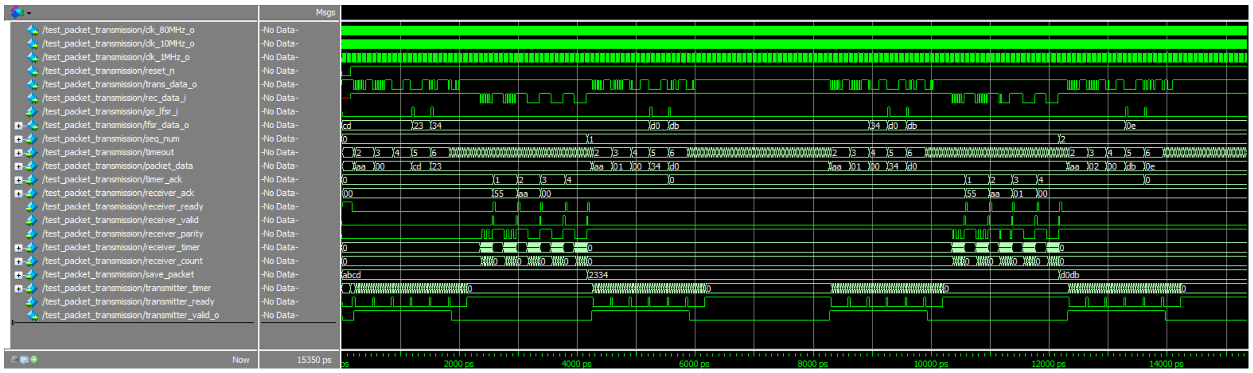
Note: The packet length is shortened to 2-byte of data to fit everything on the screen

1. *A) Test bench generates a valid packet at first and the system respond with an acknowledgement and the LFSR saves the new generation-seed for the next packet. B) The test bench uses an incorrect set of LFSR data and receiver system does not respond. C) The system reloads the old generation-seed to receive the resent packet.*

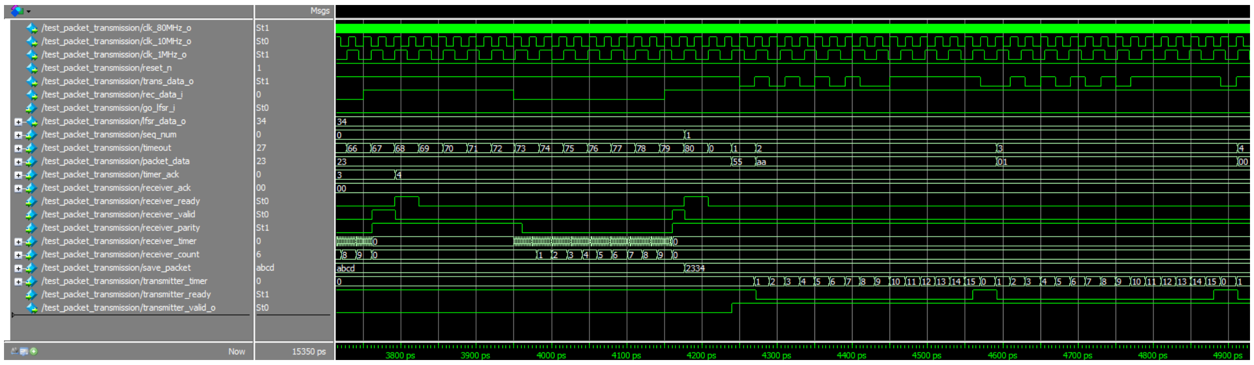


2. *The transmitter uses 16 10-MHz cycles to send each byte for the acknowledgement. Detail of the packet begins with 0x55, 0xaa, lower byte seq. #, upper byte seq. num, and x00. If there is an error, no acknowledgements are sent.*

Packet Transmission Sub-system



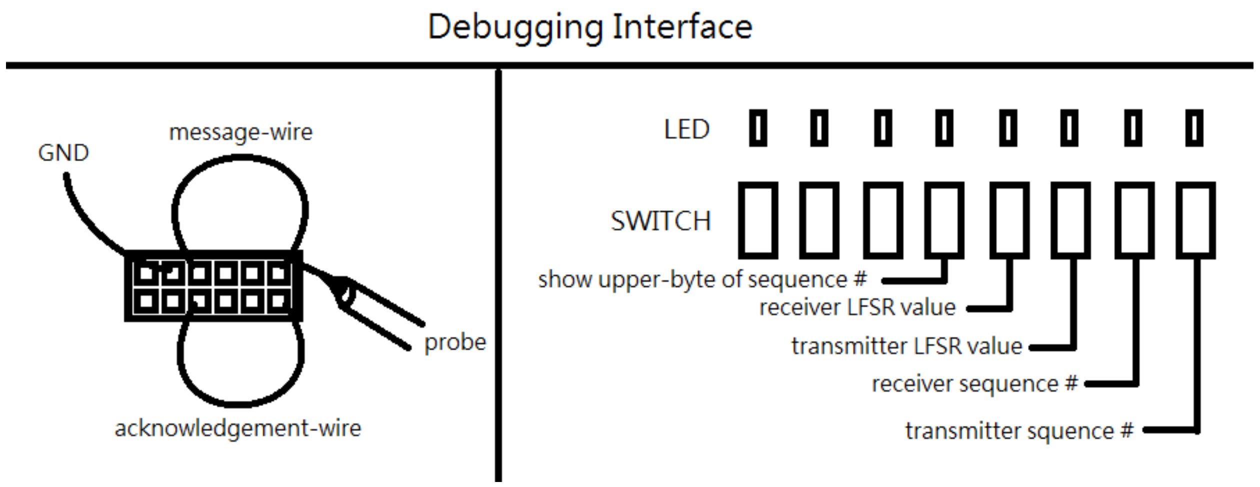
1. *A) The system begins generating packets once reset is released. The test-bench fakes an acknowledgement and the system increases its packet count. B) The system then creates the 2nd packet and the test-bench simulates not giving an acknowledgement so the system times out. C) The system resends the same packet and the process continues.*



2. *The system times out at counts of 80 (for the purpose of fitting things on the screen again) and either resend the packet or move on to the next one. In this case, the sequence number increases and the new LFSR seed is saved so it has moved on to the next packet.*

Verification

BOARD



The LED lights show any buses selected by the switches.

Testing general functionality

1) Message & Acknowledgement Send – The oscilloscope shows that message with headings and different LFSR values are being sent and the acknowledgement comes right after the message.

2) Message & Acknowledgement Receive – The sequence number of the each sub-systems increase and this occurs only after messages are received and acknowledgement is: sent for the receiver sub-system and received for the transmitter system.

3) Errors / Resend Packages – The loop-back wire is removed and reattached to test whether or not the system resumes correctly.

4) Reset – It is tested first on simulation to ensure its propagation is correct. Then it is mapped onto the reset button and debugged to figure out whether buttons are active-low or active-high.

Testing syncing of LFSR data across sub-systems

1) If the simulation shows that they do not sync up, the process is stopped and cannot resume when message-wire is removed and reattached.

2) When the simulation indicates that the LFSR values and sequence numbers are in sync, the process pauses upon removing the message-wire and resumes upon reattaching the wire.

***** NOTICED SOMETHING NEW BELOW***

3) The receiver sub-system pauses at a certain LFSR seed value when interrupted by pulling out the loop-back wire. ** The transmitter sub-system appears to be 0xFF (or flickers rapidly through many combinations) even when interrupted, but that is because the transmitter keeps trying to send the same message (as seen in the oscilloscope).

Testing syncing of sequence number

1) When loop-back wire is pulled out, then the sequence number stays and can be observed with the debugging interface. This demonstrates the acknowledgement not received case because the sequence number of the receiver sub-system is one ahead of the transmitter sub-system.

Conclusion

It is regrettable that I am not able to resolve the issue with missed acknowledgements but I made enough progress to say I did okay on the lab. It is a pretty good class but I wish we used the board a lot more and have more interesting / practical projects.