

CSCE 491

Computer Engr. Design Project

2005/10/6

802.11 MAC Layer Design

Shifter Functions (Receiver & Transmitter)

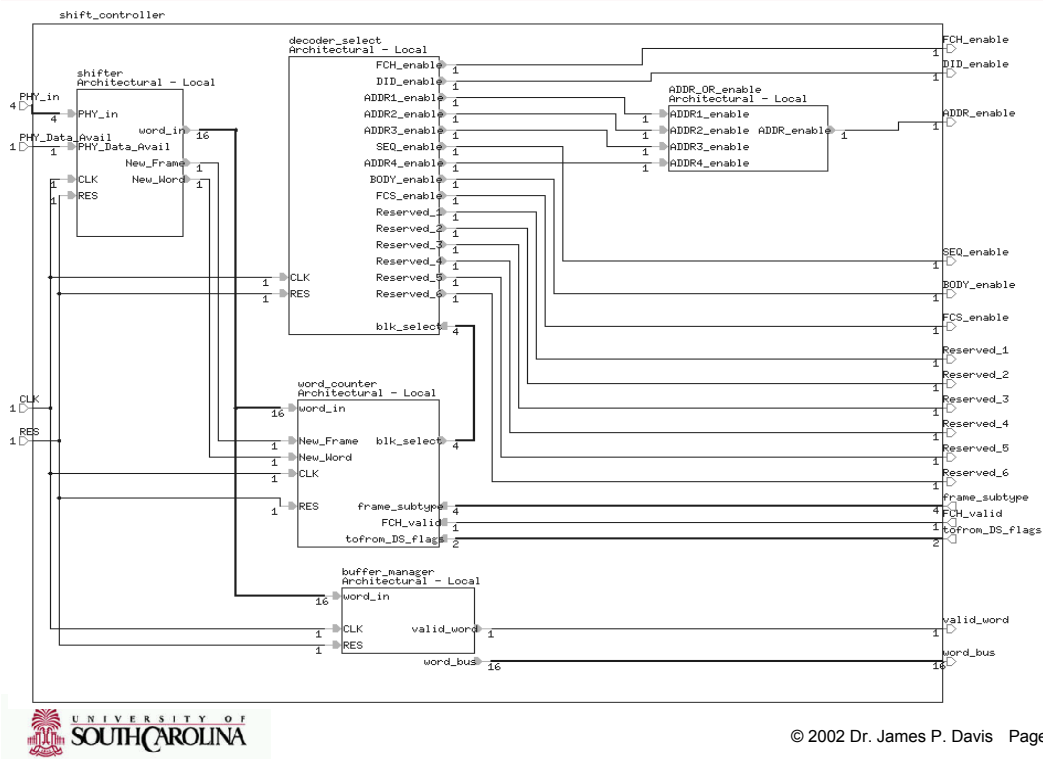
Assignment #5 (Project Task 1)

© 2005 Dr. James P. Davis

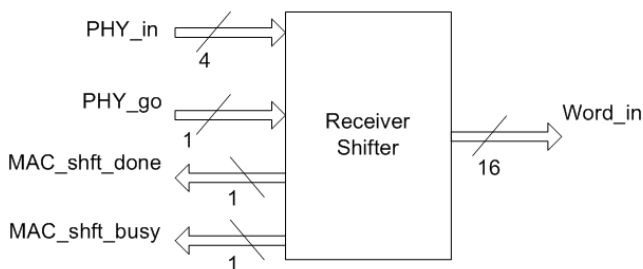
Assignment #5 Description

- **MAC Receiver teams**
 - ✓ Design the ASM thread to implement the 4-bit to 16-bit Shifter. We obtain 4-bit data words from the PHY Layer, and we want to take 4 of these and create a 16-bit internal word for processing inside the MAC Receiver. Carry out some initial simulation debugging using the Bus Table entry of data values for the 4-bit data.
 - ✓ Design the test thread that will automate passing data to the Shifter. This thread will “handshake” with the Shifter, in that it does the following: (1) polls for Shift_In to be ‘NotBusy’. It polls the flag until Shift_In is not busy, then it will set the Shifter’s Enable flag. The Shifter, if it is not busy shifting, will be polling the Enable signal, waiting for the PHY thread to set it high. When it is set high, it grabs 4 bit word, and sets its Busy flag. PHY waits until Shifter clears it before putting another 4-bit word up and setting the Enable. Once Shifter is done, it clears Busy and loops back to its Enable poll loop.
- **MAC Transmitter teams**
 - ✓ Design the ASM thread to shift out a 4-bit data word to the PHY Layer, taking the internally-generated 16-bit word and passing each chunk to the PHY Layer. You’ll also use the Bus Table to do simulation debugging.
 - ✓ Once the thread is working, you’ll add a thread that will emulate the PHY Layer for the Transmitter. Just like the Receiver side, the Transmitter side uses handshaking with PHY. But your PHY will have a poll loop waiting for Shift_out to indicate a 4-bit word is ready. Once it is enabled, it sets its PHY_Busy flag, which keeps Shift_out from overwriting the word just posted.
 - ✓ The Transmitter’s Shift_out also must interface with a thread that passes 16-bit words to the Shift_out.

Receiver Shift Controller – Block Diagram



MAC Receiver Shifter – Block Diagram



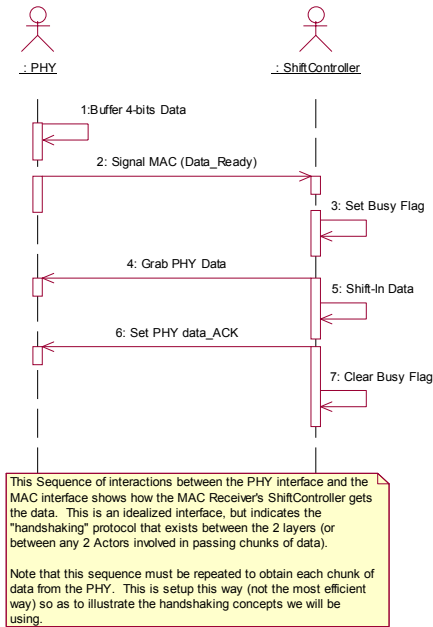
- Receiver_shifter block takes in 4-bits on each clock cycle, and after 4 clock cycles, provides a 16-bit word to the rest of the MAC Receiver block.

This block handshakes with the PHY block, in that PHY does the following: (1) test to see if the shifter is busy (MAC_shft_busy), and waits until it is not busy, then (2) once not busy, it sets the flag PHY_go, and posts its data on the PHY_in bus, then (3) the PHY block waits until Receiver_shifter block posts the signal MAC_shft_done, and then (2) PHY goes to the top of its loop to start this process again.

The Receiver_shifter block sits in a poll loop waiting for PHY_go to be set. Once set by PHY block, it sets its MAC_shft_busy flag, and starts its shift operation. Once it is done, it clears MAC_shft_busy and sets MAC_shft_done. When this thread starts, it also sets MAC_shft_done = 0 and busy = 0.



Receiver – Basic Scenario

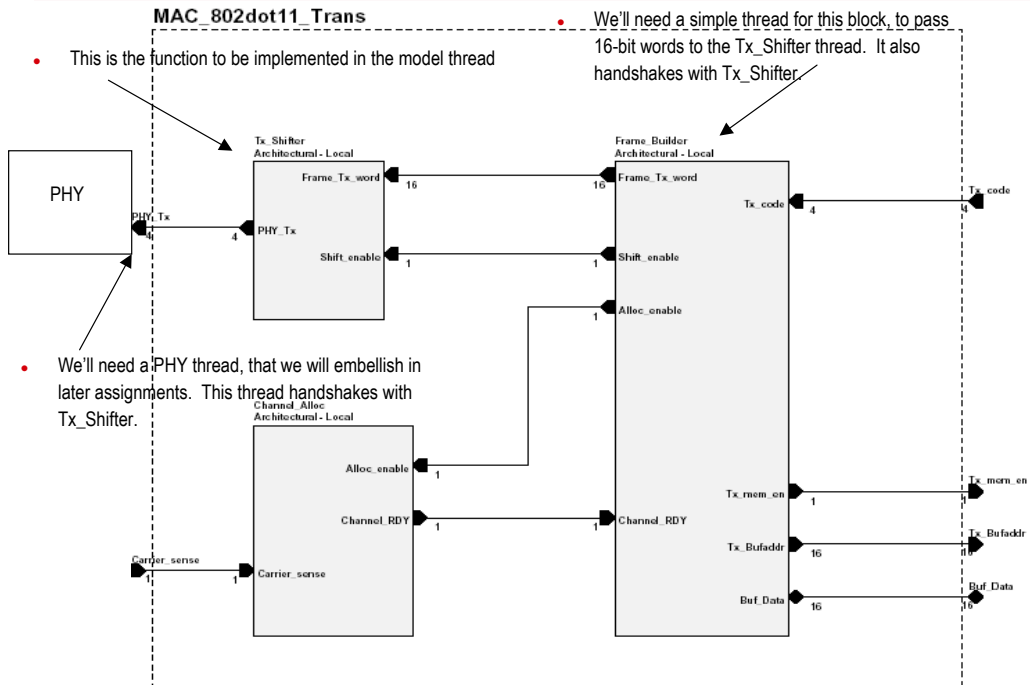


Outlining the scenario:

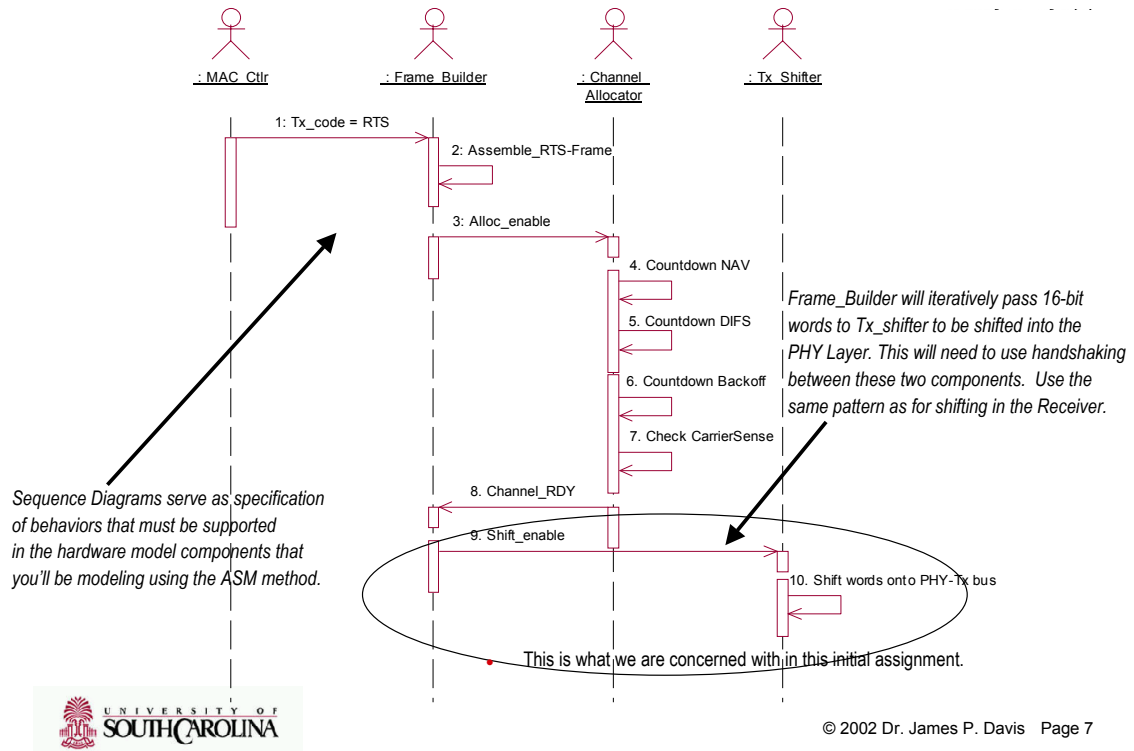
- ✓ We are receiving the data stream from the PHY Layer.
- ✓ We assume there is some means for the PHY to signal the MAC that a new data stream is ready.
- ✓ We assume the data will be passed to the MAC 4-bits at a time.
- ✓ We assume that we'll need to have some "handshaking" between the blocks in order to manage the flow of data—we don't want to overflow the MAC while it is shifting data.
- ✓ *Pipelined data processing with handshaking* is a basic design "pattern" used often in digital systems design.



Transmitter – Block Diagram



Transmitter – Basic Scenario



802.11 Transmitter – Shifting the Frame

• Tasks:

- ✓ The Transmitters TX_Shifter thread will be much like the Shifter thread in the Receiver block.
- ✓ We take each 16-bit word and shift it onto the 4-bit *MAC_out* channel. When we have shifted out all of the words, we set a *Tx_Done* flag and go back into our pool loop (waiting for the next *Tx_Shift_Enable* event).
- ✓ Question: how do we keep track of which words to shift out and in what order? It depends on the type/subtype of frame we are transmitting.
- ✓ *Suggestion*: you could build a thread in Transmitter block that maintains *current_Tx_state*, similar to what we will do for the *frame_sequencer* thread in the Receiver block. It keeps track of what frame fields need to be transmitted and in what order.

Assignment Specifics

- Create the model as follows:
 - ✓ Model the shifter function, and debug test using Bus Table and simulator. Devise 4 test cases.
 - ✓ For Transmitter, you'll need signals in Bus Table for both sides of the communication stream (16bit side of FrameBuilder and 4-bit side of PHY). For Receiver, you'll just place shifted 16-bit word onto the Word_bus.
 - ✓ Create test threads for PHY (both Receiver and Transmitter) , and a thread for Frame_Builder (Transmitter only, but it is trivial).
- Follow documentation requirements for assignment deliverables.
 - ✓ Test planning worksheet, simulation output, Effort Distribution worksheet, Bus Table and ASM Thread print outs (make sure everything is visible on the sheets). Also, enter your design information in the flowHDL file (Design Information menu).

