EECE 359 – Computer Communications and Networking **Spring 2020**

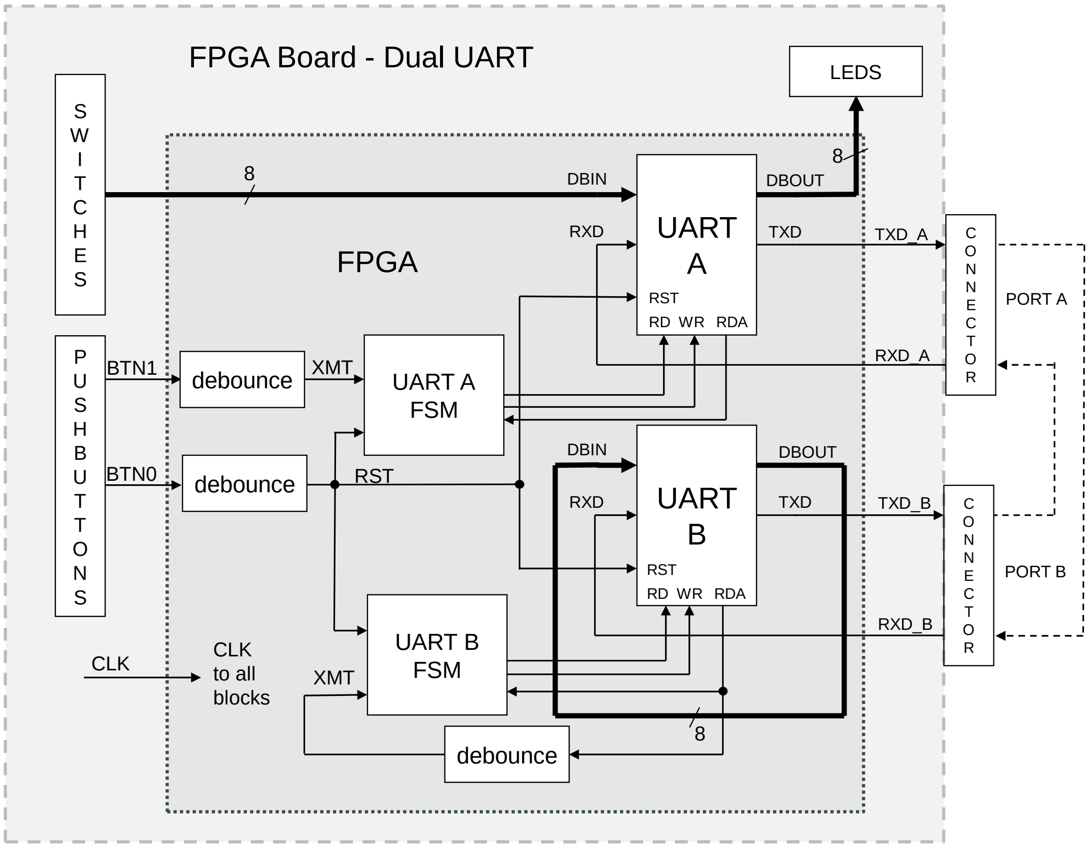
Lab #2 – Asynchronous Data Link

Objectives

This lab will use the UART (Universal Asynchronous Receiver/Transmitter) module that was introduced and studied in Lab #1, to establish an asynchronous data link. In this lab you will develop a system consisting of two UARTs. The UARTs will be set up so that they can communicate to each other through a board I/O connector. You will design the top-level VHDL module, synthesize the hardware design, simulate it, implement it on the FPGA board, and demonstrate its operation.

System Description

The system will consist of two UARTs, UART_A and UART_B (refer to the block diagram below). The 8 switches on the FPGA board will supply a byte of data to UART_A to send out on Port A. Two wires will be used to connect Port A to Port B. UART_B will receive the serial data coming into Port B. UART_B will immediately retransmit the data back to UART_A. The data received by UART_A will then be displayed on the 8 LEDs. Pushbutton 0 will be used to provide a reset to the system to initialize it. Pushbutton 1 will be used to command UART_A to transmit the data on its parallel input data bus. The RDA of UART_B will be used to command UART_B to transmit the data back to UART_A.



Design Hierarchy/Files

Implementation

- dualUART – Behavioral (dualUART .vhd)
  - UART_A – UART – Behavioral (UART.vhd)
    - Inst_UART_Tx – UART_Tx – Behavioral (UART_Tx.vhd)
    - Inst_UART_Rx – UART_Rx – Behavioral (UART_Rx.vhd)
  - UART_B – UART – Behavioral (UART.vhd)
    - Inst_UART_Tx – UART_Tx – Behavioral (UART_Tx.vhd)
    - Inst_UART_Rx – UART_Rx – Behavioral (UART_Rx.vhd)
  - UART_A_FSM – UART_FSM – Behavioral (UART_FSM.vhd)
  - UART_B_FSM – UART_FSM – Behavioral (UART_FSM.vhd)
  - RST_debounce – debounce – Behavioral (real_debounce.vhd)
  - XMT_A_debounce – debounce – Behavioral (real_debounce.vhd)
  - XMT_B_debounce – debounce – Behavioral (real_debounce.vhd)
  - dualUART-PapilioDuo.ucf

Simulation

- Test_dualUART.vhd – Test Bench
  - dualUART – Behavioral (dualUART .vhd)

All source files are provided, except the Test Bench. But, you will complete the UART finite state machine design in UART_FSM.vhd and complete the circuit interconnections of the dualUART in dualUART.vhd. You will create the Test Bench file, Test_dualUART.vhd.

PROCEDURE

Create a Xilinx ISE Project and complete the design of the dual UART

1. Create a project called "**Lab2**" in your **"EECE359Labs"** folder on your U-Drive.
2. set the **Working Directory** field to your user directory on the C-drive of the Windows client (<u>this is important to prevent the ISE software from cluttering your U-Drive with temporary working files and cause longer process times when running the Xilinx ISE software tools</u>),
3. Add the following source files (provided in myCourses):
   a. dualUART.vhd
   b. UART_FSM.vhd
   c. real_debounce.vhd
   d. UART.vhd
   e. UART_Rx.vhd
   f. UART_Tx.vhd
   g. dualUART-PapilioDuo.ucf

4. The top-level design of the dualUART system has been setup for you in the file dualUART.vhd. The circuit interconnections have been left for you to complete. Use the system block diagram as your guide.
    a. Complete the port maps for the two UARTS, components UART_A and UART_B
    b. Complete the port maps for the two UART FSMs, UART_A_FSM and UART_B_FSM
    c. Add connections to the LEDs and Switches, and for the UART_B data busses as noted at the bottom of the file
5. Complete the design of the UART finite state machine in UART_FSM.vhd for the UARTs to accomplish the system operation as outlined in the “system description” above. Each UART will have its own FSM, but both will be the same. Use a Moore type FSM.
    a. The state type declaration defines the four states that you will need. Decide on the appropriate default state at initialization and develop the process for the state register using RST as a synchronous reset.
    b. Consider the following as you develop the FSM state diagram.
        i. When the UART_FSM receives an XMT input of ‘1’, it should output a WR pulse for exactly one clock cycle to the UART to cause it to transmit the data from its DBIN bus.
        ii. When the UART_FSM receives an RDA input of ‘1’, it will set the RD output to a ‘1’ and hold it until the RDA input is ‘0’ again.
        iii. It is possible for both XMT and RDA to occur at the same time (that is if the receiving and transmitting of data are independent events). Write your code such that it will handle any timing of XMT and RDA. One thing to consider is that XMT is only one system clock cycle in duration, whereas RDA remains at a 1-level until the RDA is reset by the RD signal on the next rising edge of the rClk in the UART.
    c. Draw your FSM state diagram and submit an image file to the lab assignment (preferred) or a hard copy to the TA
    d. Develop the code for the FSM
6. Perform “Check Syntax” and correct any errors
7. Perform “Synthesis” and correct any errors

    The following are expected warnings because of unused signals/ports

Synthesis Messages - Errors, Warnings, and Infos
Xst:753 - "C:/Lab2/dualUART.vhd" line 134: Unconnected output port 'TBE' of component 'UART'.
Xst:753 - "C:/Lab2/dualUART.vhd" line 134: Unconnected output port 'PE' of component 'UART'.
Xst:753 - "C:/Lab2/dualUART.vhd" line 134: Unconnected output port 'FE' of component 'UART'.
Xst:753 - "C:/Lab2/dualUART.vhd" line 134: Unconnected output port 'OE' of component 'UART'.
Xst:753 - "C:/Lab2/dualUART.vhd" line 148: Unconnected output port 'TBE' of component 'UART'.
Xst:753 - "C:/Lab2/dualUART.vhd" line 148: Unconnected output port 'PE' of component 'UART'.
Xst:753 - "C:/Lab2/dualUART.vhd" line 148: Unconnected output port 'FE' of component 'UART'.
Xst:753 - "C:/Lab2/dualUART.vhd" line 148: Unconnected output port 'OE' of component 'UART'.

Xst:2677 - Node <UART_A/Inst_UART_Rx/OEint> of sequential type is unconnected in block <dualUART>.
Xst:2677 - Node <UART_A/Inst_UART_Rx/PEint> of sequential type is unconnected in block <dualUART>.
Xst:2677 - Node <UART_B/Inst_UART_Rx/OEint> of sequential type is unconnected in block <dualUART>.
Xst:2677 - Node <UART_B/Inst_UART_Rx/PEint> of sequential type is unconnected in block <dualUART>.

Develop the Test Bench

1. Generate the test bench file using the **Project → New Source** menu option and name the file **Test_dualUART.vhd**.
2. Copy the "generic" statement from the dualUART entity in the dualUART.vhd file and paste it before the port list statement in the component declaration for the dualUART.
3. Insert a "generic map" statement into the uut dualUART component instantiation. In that generic map, map the debounceDELAY parameter to a value of 3. This will set the debounce delay to a small value to keep the simulation time to a reasonable length while allowing the debounce to function as desired on the FPGA board.
4. Set the clock period for CLK to be the period of the system clock on your FPGA board.
5. Define the external connections to connect the UART_A output TXD_A to the UART_B input RXD_B and the UART_B output TXD_B to the UART_A input RXD_A. Write regular VHDL signal assignment statements in the test bench architecture before the stimulus process to accomplish this.
6. Add stimuli to the Test Bench to place a data pattern on the switches, generate a pulse on the RST to initialize the unit under test (uut), and generate a pulse on the XMT_A input of the uut to start the data transmission, using the following steps:
    a. Set the switches to have a data pattern of "01010110"
    b. Simulate a reset (RST) from a momentary switch (e.g. set RST to '0' for 20 µs then to '1' for 20 µs, then back to '0'). Note: It is important that the reset signal begins at a '0' level so that the debounce circuit will see a rising transition. Otherwise, the debounce circuit will not generate a pulse.
    c. Wait for 20 µs of time.
    d. Simulate a transmit command (XMT_A) from a momentary switch (e.g. generate another 20 µs pulse).

Run Simulation and Analyze the Operation of the Dual UART

1. Run a simulation using the Test_dualUART test bench for 3 milliseconds.
    a. Group the default waveforms with the name "**Test Bench**".
    b. You should see the two data transmissions, one from TXD_A to RXD_B followed by the other from TXD_B to RXD_A.
    c. You should see the LEDs change from "00000000" to "01010110" after UART_A receives the data transmission from UART_B.
    d. Drag the "uut" instance into the waveform viewer and name the group of signals "**dualUART**".
    e. Drag the "**UART_A_FSM**" instance into the waveform viewer and name the group of signals "**UART_A_FSM**".

f. Drag the “UART_B_FSM” instance into the waveform viewer and name the group of signals “UART_B_FSM”.

g. Save the waveform configuration file.

h. Rerun the simulation to capture all of the additional signal waveforms

i. In the Test Bench group, you will first see a pulse on the “rst” signal and then a pulse on the “xmt_a” signal. Then you will see the data frame transmitted on the “txd_a” signal and received at the same time on the “rxd_b” signal. Soon after that, you will see the same data frame transmitted on the “txd_b” signal and received on the "rxd_a“ signal. Soon after that transmission is finished, you will see the LED signal change from “00000000” to “01010110”

j. In the UART_A_FSM group after the reset, you should see the UART_A WR being generated by UART_A_FSM from the XMT_A input. Zoom in to see the timing of these signals and the FSM states.

k. After UART_B receives the data frame from UART_A, you should see its RDA signal go active causing the XMT_B input to UART_B_FSM to be generated. You should see the UART_B_FSM output the WR and RD signals for UART_B.

l. <u>Save your waveforms, with all of the groups expanded, to a PDF by printing with the “Microsoft Print to PDF” printer option. Include this PDF with your lab submission on myCourses.</u>

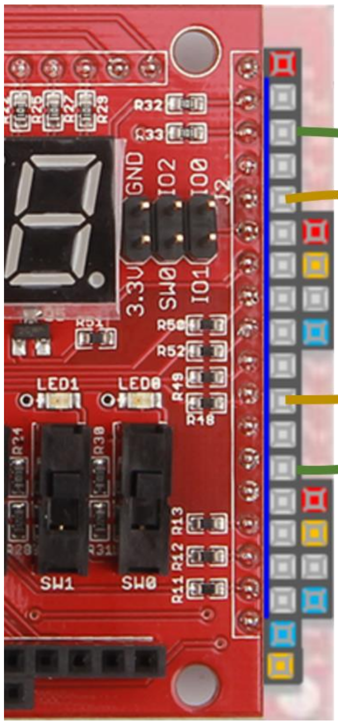
2. Answer the following questions as you analyze the operation of the Dual UART in simulation:
   a. What is the sequence of states (stcur) that you observe in the **UART_A_FSM** during the simulation?
   b. Describe what happened in **UART_A** during each of the states and what caused each state transition to occur. Include in your description all inputs/outputs of the UART_A that show activity as well as those of the FSM.
   c. What is the sequence of states (stcur) that you observe in the **UART_B_FSM** during the simulation?
   d. Describe what happened in **UART_B** during each of the states and what caused each state transition to occur. Include in your description all inputs/outputs of the UART_B that show activity as well as those of the FSM.

Implement the dualUART and generate a programming file

1. Use the provided UCF file for your board.
2. Run the Translate, Map, and Place & Route processes in ISE
3. If necessary, correct any errors and rerun the processes (These processes should be warning free with green check marks)
4. Generate the programming file (remember to set the start-up clock to “JTAG Clock” in the process properties)

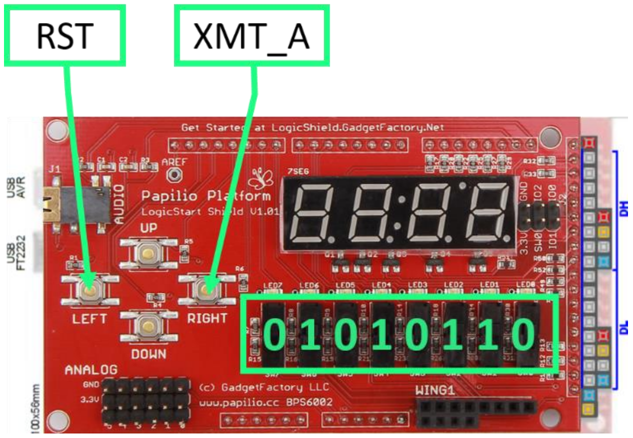
Test out the dualUART system on your FPGA board

1. Make the necessary external connections to your Papilio Duo/LogicStart Shield FPGA board according to the diagram. Use jumper wires to connect TXD_A to RXD_B and TXD_B to RXD_A as shown.



5V		
D0		
D1	TXD_A	
D2		
D3	RXD_A	
D4		5V
D5		3.3V
D6		
D7		GND
D8		
D9	TXD_B	
D10		
D11	RXD_B	
D12		5V
D13		3.3V
D14		
D15		GND
GND		
3.3V		

2. Connect your FPGA board to a USB port on the PC
3. Program the FPGA on your FPGA board using the Papilio Loader app.
4. Set the slide switches for a “01010110” data pattern
5. Activate the momentary switch for RST to reset the system
6. Activate the momentary switch for XMT_A to start data transmission



7. The data pattern of the switches should now appear on the LEDs
8. Try sending other data patterns
9. Use the digital scope to view the two signals traversing the “cable” between the two ports. You can do this by stripping additional insulation from one end of each of the

two wires to allow clipping the scope cables (red clip) to them. Be sure to connect the black clip of the two scope cables to GND on the board connector.

10. Capture an image of the two signals on the scope using the single trigger mode.
11. Compare the signals to those in the simulation using the same data pattern.
12. Measure the bit period of the data on the digital scope using the two cursors. Verify that the bit period is correct for a transmission rate of 9600 Baud
13. <u>Save the scope image in a file and submit it to the lab assignment</u>.

    To save a PNG image file:

    1. Image files can be saved to an external USB storage device. Plug a USB flash drive into the USB connector on the front of the scope
    2. Press **[Save/Recall] > Save > Format**; then, turn the Entry knob to select **24-bit image (*.png)**.
    3. Press the softkey in the second position and use the Entry knob to navigate to the save location.
    4. Press the **Settings** softkey.
    5. In the File Settings Menu, you have these softkeys and options:
        - **Setup Info** — setup information (vertical, horizontal, trigger, acquisition, math, and display settings) is also saved in a separate file with a TXT extension.
        - **Invert Grat** — the graticule in the image file has a white background instead of the black backgound that appears on- screen.
        - **Palette** — lets you choose between Color or Grayscale images.
    6. Finally, press the **Press to Save** softkey.

       A message indicating whether the save was successful is displayed.
14. <u>Demonstrate your system to a lab proctor and get checked off on their sheet</u>.

Submit your lab on myCourses:

Submit the following:

1. FSM State Diagram (10 points)
2. PDF of simulation waveforms (10 points)
3. Answers to questions (20 points)
4. Scope screen image of TXD_A / RXD_B and TXD_B / RXD_A waveforms (10 points)
5. The following VHDL code files: (30 points)
    - Test_dualUART.vhd
    - dualUART.vhd
    - UART_FSM.vhd

Proctor checkoff of FPGA board demo on checkoff sheet (20 points)