ECE353 In-Class Exercise

UART – Advanced Features

Problem 11B Objectives

- Generate the code for the producer of the transmit data flow
- Implement the transmit portion of the UART ISR

1. Copy gpioPort.c

A. Add your version of **`gpioPort.c`** from last week's UART polling exercise to the Keil uVision project in the drivers directory.

2. Add pc_buffer.c

A. Add your version of **`pc_buffer.c`** to the project in the drivers directory.

3. Modify uart.c

A. Modify the producer function **`uartTx()`**. See the transmit workflow on the ECE353 virtual book and the comments found in **`uart.c`**.

4. Modify interrupts.c

A. Copy your code from the **`Rx_Flow`** from the previous ICE.

B. Modify **`UART0_Tx_Flow`** so that data is moved from the transmit circular buffer into the transmit hardware FIFO. This routine should exit once either the hardware FIFO becomes full **or** the circular buffer is empty. If the circular buffer is empty, disable transmit empty interrupts.

5. What to Turn In

Turn in **`interrupts.c`** to the dropbox on the course website.

Notes: fputc and fgetc allow us to use printf, getc, scanf, etc to generate output on the serial debug interface. These libraries make a calls to fputc and fgetc for each character that gets transmitted/received using stdio.h. In many situations you many not want to use stdio.h since it consumes a large amount of code space and is not thread safe for embedded operating systems. For our class, we will not worry about those constraints. Having access to printf and scanf will be very handy for debugging the software we write in class.