EE4144 – Intro to Embedded Systems

Lab Exercise 3

Interrupt Subroutines and UART Serial Communications

Objective:

This lab exercise demonstrates how to implement interrupt subroutines and use the UART provided by the Atmel 32U4 microcontroller. We begin by setting up the UART for serial communications between two Atmel 32U4 controllers. Two controllers are required for this lab so you should form teams of two. One team (A) will set up a simple push-button sensor that will activate the onboard LED on the other team (B) using interrupts and the UART serial interface. Team (B) will then respond to (A) confirming the status of its LED. Team (A) will then illuminate its own LED to match the status of (B)'s LED

Setup:

The following equipment will be required for this lab exercise:

1. 2 Adafruit Playground Classic controllers, including a momentary push button (Indicate Team A and Team B)
2. Interconnect wires, with alligator clips

Setting up the UART:

There is only one UART available on the Playground Classic on the Atmel 32U4, which utilizes Pin 0 as Rx and Pin 1 as Tx. These pins are clearly labeled on your microcontroller. We wish to set up the following configuration for the UART serial bus on both controllers:

• Bit Rate:	9,600
• Data Bits:	8
• Parity :	None
• Stop Bits:	1

- Asynchronous/Normal Speed
- Rx Interrupt Enabled

Using C, write the setup code for the following registers:

UCSRA, UCSRB, UCSRC and UBRR

→(0) Show the lab instructor your setup code

Be sure to set up both microcontrollers (both teams) to the same exact serial parameters.

Setting up the Hardware:

In this step we set up the serial link between teams.

1. Wire team (A) Tx to team (B) Rx
2. Wire team (A) Rx to team (B) Tx
3. Wire team (A) Ground to team (B) Ground

→(1) Show your connection setup to your lab instructor.

Setting up the UART Interrupts on Team A and B

We wish to implement a subroutine that handles an RX and TX UART interrupt for both teams. On both teams, set up the following interrupt register using C:

UCSRB

Also, each team should set up one of the push buttons to an Input that can be read by your code.

→(2) Show the code that sets up these registers and explain your bit selections.

Now, write the ISR that handles the interrupt, ISR(USART_RX_vect) on both controllers.

In your Team A loop, write the code that will send a "1" to Team B over the UART serial bus if the button is pressed and a "0" if the button is released (by writing the UDR register accordingly). Recall, the UDR register should contain "0" (0x30) if the button is released, "1"(0x31) if the button is depressed.

→(3) Show your lab instructor your loop routine.

TeamA is now set up to transmit one byte over the UART serial link to Team B each time the button changes states.

Setting up Team B

The serial parameters of Team B should already be setup in the previous steps. Be sure to double check that the Tx and Rx interrupts on the UART is enabled on BOTH controllers.

Setup the DDR register such that Pin 13 is set as an output.

To handle the Rx interrupts on Team B, confirm the implementation of the ISR(USART_RX_vect) function. This function will run each time a character is received by the UART. In this function, read the UDR register and see if it contains 0x30 or 0x31. Based on the value, illuminate the onboard LED (Pin 13) using the PORT register.

Immediately after assigning the PORT register, send back to Team A 0x30(OFF) or 0x31(ON) depending on the state of the LED.

→(4)Show the lab instructor the ISR(USART_RX_vect) subroutine and the setup for Pin13.

Completing the setup for Team A

Team A is already setup to detect the push button state. In addition, Team A is programmed to notify Team B of the button change by sending a byte over the serial bus. We now need Team A to implement the USART_RX_vect ISR so that when it receives a byte from Team B such that it can illuminate it's LED accordingly.

Setup the DDR register such that Pin 13 is set as an output.

To handle the Rx interrupts on Team A, implement the ISR(USART_RX_vect) function. This function will run each time a character is received by the UART. In this function, read the UDR register and see if it contains 0x30 or 0x31. Based on the value, illuminate the onboard LED (Pin 13) using the PORT register.

→(4)Show the lab instructor the ISR(USART_RX_vect) subroutine and the setup for Pin13.

Testing the setup

Download your code for Team A and Team B and confirm the hardware connections made previously in the lab. Each time you press the push button, a message is sent to illuminate the LED on Team B. A confirmation is then sent back to Team A illuminating the LED on Team A. The net result: Each time the button is depressed, the LED on BOTH controllers should illuminate and when released, both LEDs should go dark.

→(5)Demonstrate the proper function to the lab instructor.

Troubleshooting Tips:
If the setup is not working….

1. Confirm that the serial setup on both controllers is identical
2. Confirm Tx-Rx and Rx-Tx
3. Don't forget to setup the LEDs as outputs and the button as an input
4. Check the spelling of your ISRs; they must be exact.