**b) Prelab Questions**

1. What is the difference between synchronous and asynchronous communication?
   1. With synchronous communication, an external clock is used to synchronize sending and receiving. With asynchronous communication, special signals such as “DREIF” on the XMEGA are used to synchronize sending and receiving.
2. What is the difference between serial and parallel communication?
   1. Serial communication uses a single transmission medium, and transmits data one bit at a time. Parallel communication uses multiple transmission mediums simultaneously to transmit multiple bits at once.
3. List the XMEGA’s USART registers used in your programs and briefly describe their functions.
   1. USARTp#\_STATUS: Provides the current status of the Rx/Tx. Checking appropriate values allows for receiving or transmitting at the right times.
   2. USARTp#\_DATA: Register to check when it is time to transmit or receive.
   3. USARTp#\_CTRLA: Controls USART interrupt levels.
   4. USARTp#\_CTRLB: Configures USART, enables Rx/Tx.
   5. USARTp#\_CTRLC: Configures USART’s communication mode, parity mode, stop bit mode, character size, and data order.
   6. USARTp#\_BAUDCTRLA: Holds the lower 8 bits of the BSel.
   7. USARTp#\_BAUDCTRLB: Holds the BScale and upper 4 bits of the BSel.
4. List the number of bounces from part A of this lab. How long (in ms) is your delay routine for debouncing?
   1. Variable from 0 to ~8. My delay routine is approximately 10ms.
5. What is the maximum possible baud you can use for asynchronous communication if your board runs at 2MHz? Support your answer with the values you would place in any special registers that are needed.
   1. Max Baud = 125000Hz; BSel = 0; BScale = 0

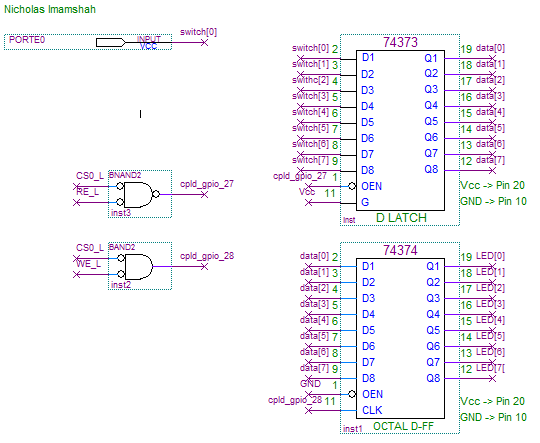
**c) Problems Encountered**

1. Initially I forgot to include the null character at the end of my strings, so I was having trouble debugging the OUT\_STRING method in part E.
2. I did not understand how to verify the baud rate initially; mainly I misunderstood what was meant by the lab document when it mentioned using Port E or Port F for the USART.

**d) Future Work/Applications**

In the lab we used serial communication to transmit ASCII characters from our processor to our computer; this concept represents the much more general process of sending a stream of data between devices. Such a configuration is reminiscent of a networking scenario, where there is a sending a receiving device, the receiving device constantly accepting packets of some sort from the sending device.

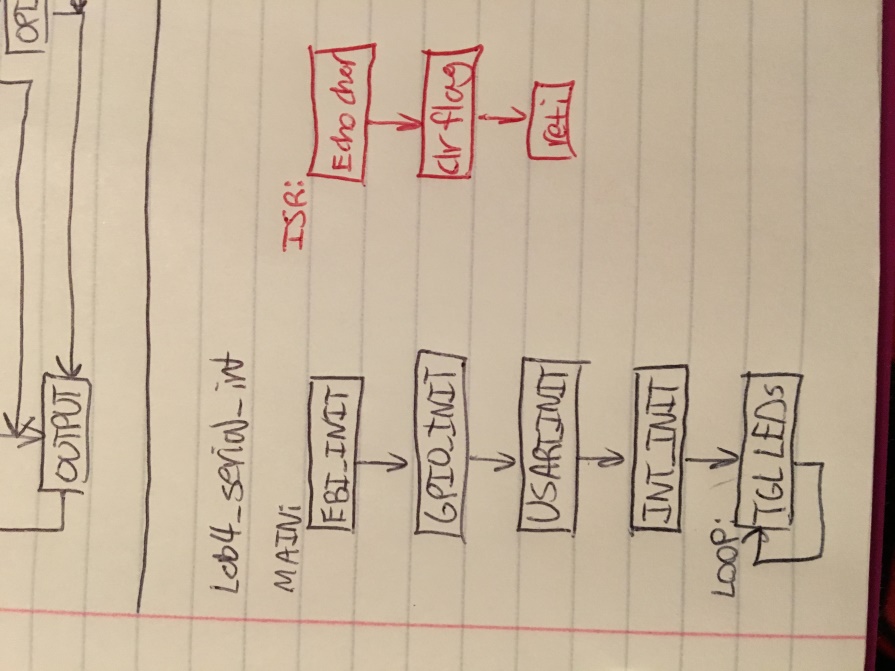
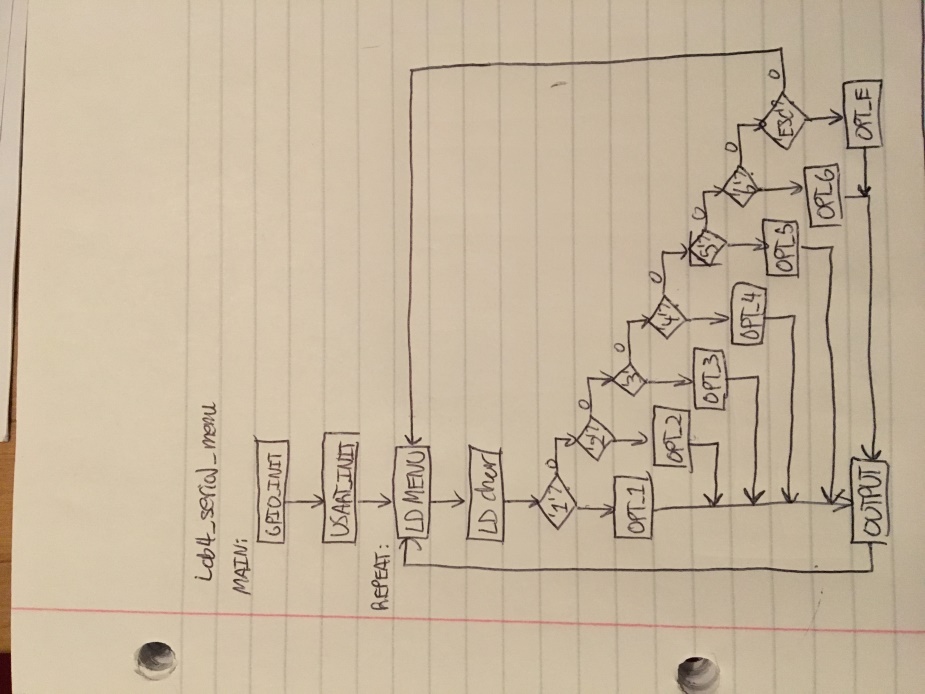
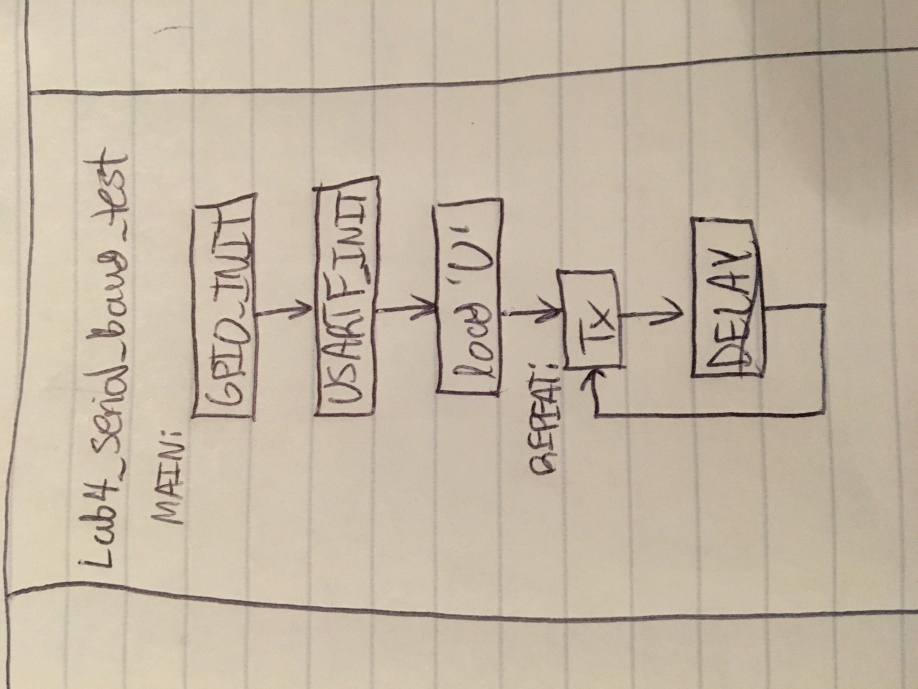
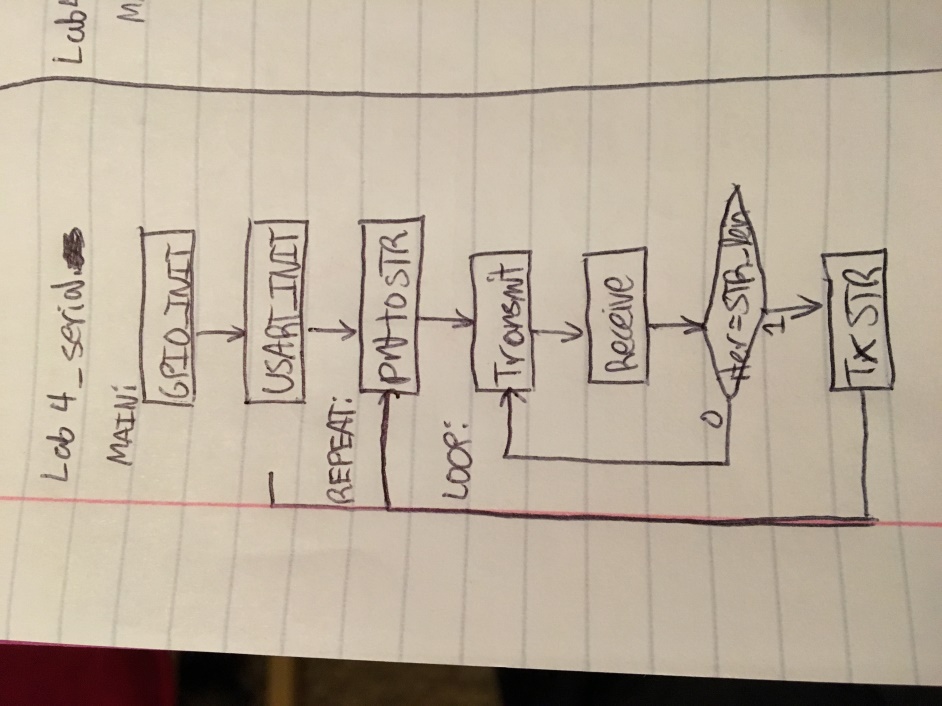
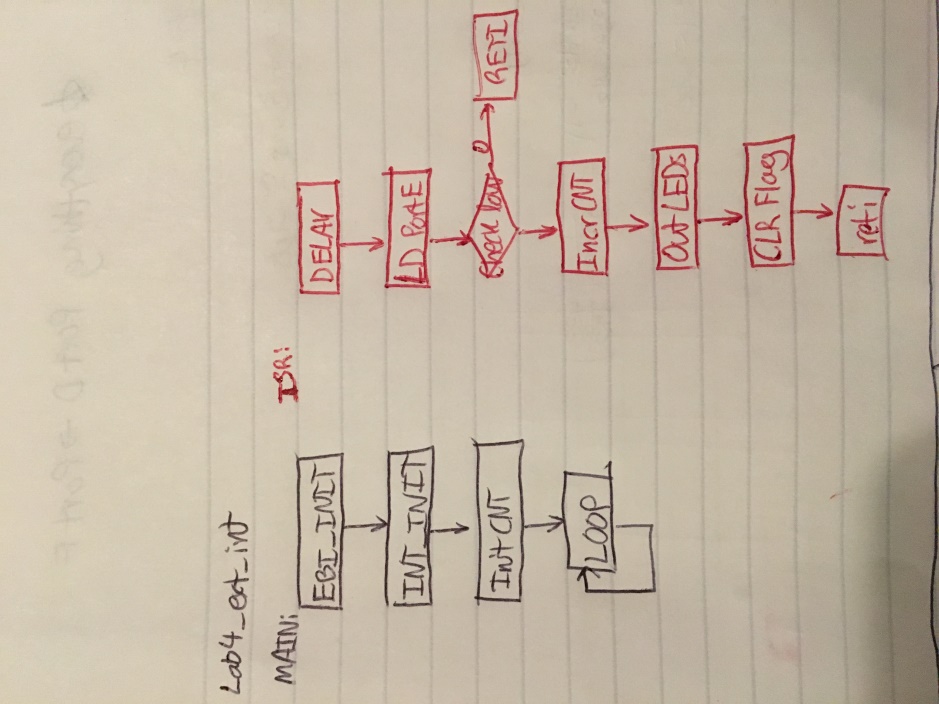
**e) Schematics**

****

**f) Decoding Logic**

N/A

**g) Pseudocode/Flowcharts**

****

**h) Program Code**

Lab4\_ext\_int.asm

/\* Lab4\_ext\_intr.asm

\*

\* Lab 4 External Interrupt

\* Name: Nicholas Imamshah

\* Section: 6957

\* TA Name: Daniel Gonzalez

\* Description: The purpose of this program is to configure an interrupt on the XMEGA that will watch for a falling edge.

\*/

.nolist

.include "ATxmega128A1Udef.inc"

.list

.org 0x0000

rjmp MAIN

.org PORTE\_INT0\_VECT ;place code at the interrupt vector for the PORTE\_INT0 interrupt

jmp ISR\_LED\_COUNT ;jump to our interrupt routine

;must org program at 0x0200 so it doesn't conflict with interrupt vectors that are at 0x0000-0x00FE

.org 0x0200

MAIN:

rcall CONFIG\_EBI

nop

rcall INIT\_INTERRUPT

nop

ldi r18, 0 ;initialize count

LOOP:

rjmp LOOP

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Name: INIT\_INTERRUPT

\* Purpose: Subroutine to initialize the PortE external pin interrupt PE0 using INT0

\* Inputs: None

\* Outputs: None

\* Affected: r16, PMIC\_CTRL, PORTE: \_INT0MASK\_OUT, \_DIRCLR, \_INTCTRL, \_PIN0CTRL

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

INIT\_INTERRUPT:

ldi r16, 0x01 ;set output to default to '1'.

sts PORTE\_OUT, r16

sts PORTE\_DIRCLR, r16 ;Set external interrupt pin (PE0) as an input

ldi r16, 0x01 ;select the external interrupt as a low-level priority

sts PORTE\_INTCTRL, r16

ldi r16, 0x01 ;select PORTE\_PIN0 as the interrupt source

sts PORTE\_INT0MASK, r16

ldi r16, 0x02 ;select falling edge for external interrupt

sts PORTE\_PIN0CTRL, r16

ldi r16, 0x01 ;enable low-level interrupts

sts PMIC\_CTRL, r16

sei ;set global interrupt flag LAST!

ret

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Name: ISR\_LED\_COUNT

\* Purpose: Interrupt service routine to increment the count of executions on LEDs

\* Inputs: None

\* Outputs: None

\* Affected: r17, r18

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

ISR\_LED\_COUNT:

ldi r16, 255

rcall DELAY

nop

lds r16, PORTE\_IN

nop

sbrc r16, 0 ;if bit0 is cleared, then continue

reti ;else, we got a rising edge

inc r18 ;inc each time ISR runs

st X, r18 ;store to LEDs

ldi r17, 0x01

sts PORTE\_INTFLAGS, r17 ;clear the PORTE\_INTFLAGS

reti ;return from the interrupt service routine

DELAY:

ldi r17, 50

rcall DELAY\_INNER

dec r16

nop

brne DELAY

ret

DELAY\_INNER:

dec r17

nop

brne DELAY\_INNER

ret

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Name: CONFIG\_EBI

\* Purpose: Subroutine to initialize Ports H,J,K/EBI for additional Input/Output ports

\* Inputs: None

\* Outputs: None

\* Affected: R16, X, Z, EBI\_CTRL, PORTH,J,K,

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

.set PORT = 0x288000

.set PORT\_END = 0x289FFF

CONFIG\_EBI:

ldi r16, 0x17 ;Configure PORTH bits 4, 2, 1, and 0 as outputs.

sts PORTH\_DIRSET, r16 ; These are the CS0(L), ALE1(H), RE(L), and WE(L) outputs.

; (CS0 is bit 4; ALE1 is bit 2; RE is bit 1; WE is bit 0)

ldi r16, 0x13 ;Default CS0(L), RE(L), and WE(L) to H = false.

sts PORTH\_OUTSET, r16 ; ALE defaults to 0 = L = false.

ldi r16, 0xFF ;Set all PORTK pins (A15-A0) to be outputs.

sts PORTK\_DIRSET, r16

ldi r16, 0xFF ;Set all PORTJ pins (D7-D0) to be outputs.

sts PORTJ\_DIRSET, r16

ldi r16, 0x01 ;Store 0x41 in EBI\_CTRL reg to select 3 port EBI(H,J,K)

sts EBI\_CTRL, r16 ; mode and SRAM ALE1 mode

;Initialize the Z pointer to point to the base address for CS0 in memory

ldi ZH, high(EBI\_CS0\_BASEADDR)

ldi ZL, low(EBI\_CS0\_BASEADDR)

;Load the middle byte (A15:8) of the three byte addr into a reg and store it as the

; LOW byte of the Base Address, BASEADDRL.

ldi r16, byte2(PORT)

st Z+, r16

;Load the highest byte (A23:16) of the three byte addr into a reg and store it as the

; HIGH byte of the Base Address, BASEADDRH.

ldi r16, byte3(PORT)

st Z, r16

ldi r16, 0x15 ;Set to 8KB CS space and turn on SRAM mode, 0x288000 - 0x289FFF

sts EBI\_CS0\_CTRLA, r16

;Steps for using the port expansion

ldi r16, byte3(PORT) ;initialize a pointer to point to the base addr of the PORT

sts CPU\_RAMPX, r16 ;use the CPU\_RAMPX reg to set the third byte of the pointer

ldi XH, high(PORT) ;set the middle (XH) and low (XL) bytes of the pointer as usual

ldi XL, low(PORT)

ret

Lab4\_serial.asm

/\* Lab4\_serial.asm

\*

\* Lab 4 XMEGA USART System

\* Name: Nicholas Imamshah

\* Section: 6957

\* TA Name: Daniel Gonzalez

\* Description: The purpose of this program is to configure the XMEGA USART for communication with a terminal program.

\*/

.nolist

.include "ATxmega128A1Udef.inc"

.list

.equ BSel = 289

.equ BScale = -7

.equ Prompt = '?'

.equ StringLength = 10

.dseg

.org 0x2000 ;where inputs will be stored, outputs written to

STRING: .byte StringLength

.cseg

.org 0x0000

rjmp MAIN

.org 0x0200

MAIN:

rcall GPIO\_INIT

rcall USART\_INIT

REPEAT:

ldi r16, Prompt ;load Prompt character

ldi r17, StringLength

ldi ZL, low(STRING)

ldi ZH, high(STRING)

LOOP:

rcall OUT\_CHAR ;echo character

rcall IN\_CHAR

st Z+, r16 ;store input char

dec r17

brne LOOP

rcall OUT\_CHAR ;echo last entered char

ldi r16, '!'

rcall OUT\_CHAR ;signify end of string

rcall OUT\_STRING

rjmp REPEAT

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Name: USART\_INIT

\* Purpose: Subroutine to initialize the XMEGA USART system

\* Inputs: None

\* Outputs: None

\* Affected: r16, USARTD0\_CTRLB, USARTD0\_CTRLC, USARTD0\_BAUDCTRLA, USARTD0\_BAUDCTRLB

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

USART\_INIT:

ldi r16, 0x18

sts USARTD0\_CTRLB, r16 ;turn on TXEN and RXEN lines

ldi r16, 0x23

sts USARTD0\_CTRLC, r16 ;Parity = even, 8 bit frame, 1 stop bit

ldi r16, (BSel & 0xFF) ;select only lower 8 bits of BSel

sts USARTD0\_BAUDCTRLA, r16 ;set BAUDCTRLA to lower 8 bits of BSel

ldi r16, ((BScale << 4) & 0xF0) | ((BSel >> 8) & 0x0F)

sts USARTD0\_BAUDCTRLB, r16 ;set BAUDCTRLB to BScale | Bsel.

; Lower 4 bits are upper 4 bits of BSel.

; Upper 4 bits are upper 4 bits of BScale.

ret

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Name: GPIO\_INIT

\* Purpose: Subroutine to initialize the XMEGA GPIO for use with the USART System

\* Inputs: None

\* Outputs: None

\* Affected: r16, PORTD\_DIR, PORTD\_OUT, PORTQ\_DIR, PORTQ\_OUT

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

GPIO\_INIT:

ldi r16, 0x08

sts PORTD\_DIRSET, r16 ;set PORTD\_PIN3 as output for TX pin of USARTD0

sts PORTD\_OUTSET, r16 ;set the TX line to default to '1'

ldi r16, 0x04

sts PORTD\_DIRCLR, r16 ;set RX pin for input

ldi r16, 0xA

sts PORTQ\_DIRSET, r16 ;set pins 1 and 3 of PORTQ to output

sts PORTQ\_OUTCLR, r16 ;set USB\_SW\_EN = '0', USB\_SW\_SEL = '0'

ret

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Name: OUT\_CHAR

\* Purpose: Subroutine to output a single character to the transmit pin of the USART

\* Inputs: None

\* Outputs: Transmits data via USART

\* Affected: USARTD0\_STATUS, USARTD0\_DATA

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

OUT\_CHAR:

push r17

TX\_POLL:

lds r17, USARTD0\_STATUS ;load status register

sbrs r17, 5 ;proceed to writing out the char if

; the DREIF flag is set

rjmp TX\_POLL ;else go back to polling

sts USARTD0\_DATA, r16 ;send the character out over the USART

pop r17

ret

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Name: OUT\_STRING

\* Purpose: Subroutine to output character strings stored in memory

\* Inputs: None

\* Outputs: Transmits data via USART

\* Affected: r16

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

OUT\_STRING:

ldi ZL, low(STRING)

ldi ZH, high(STRING)

PRINT\_CHAR:

ld r16, Z+ ;load char pointed to by Z, POST-increment

cpi r16, 0x00 ;check if char is null

breq PRINT\_DONE ; if null -> DONE printing string

rcall OUT\_CHAR ; else OUTPUT that char

rjmp PRINT\_CHAR ;repeat

PRINT\_DONE:

ret

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Name: IN\_CHAR

\* Purpose: Subroutine to receive a single character from receiver pin of the USART

\* Inputs: None

\* Outputs: r16 loaded with input from SCI

\* Affected: r16, USARTD0\_STATUS, USARTD0\_DATA

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

IN\_CHAR:

RX\_POLL:

lds r16, USARTD0\_STATUS ;load the status register

sbrs r16, 7 ;proceed to reading in a char if

; the RXB8 flag is set

rjmp RX\_POLL ;else continue polling

lds r16, USARTD0\_DATA ;read the character into r16

ret

Lab4\_serial\_baud\_test.asm

/\* Lab4\_serial\_baud\_test.asm

\*

\* Lab 4 XMEGA USART Baud Rate Test

\* Name: Nicholas Imamshah

\* Section: 6957

\* TA Name: Daniel Gonzalez

\* Description: The purpose of this program is to test the Baud Rate of the XMEGA's USART System.

\*/

.nolist

.include "ATxmega128A1Udef.inc"

.list

.equ BSel = 289

.equ BScale = -7

.org 0x0000

rjmp MAIN

.org 0x0200

MAIN:

rcall GPIO\_INIT

rcall USART\_INIT

ldi r16, 0x55 ;load U

REPEAT:

rcall OUT\_CHAR ;echo U to console

rcall DELAY\_1MS ;delay 1000us

rjmp REPEAT

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Name: USART\_INIT

\* Purpose: Subroutine to initialize the XMEGA USART system

\* Inputs: None

\* Outputs: None

\* Affected: r16, USARTE0\_CTRLB, USARTE0\_CTRLC, USARTE0\_BAUDCTRLA, USARTE0\_BAUDCTRLB

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

USART\_INIT:

ldi r16, 0x18

sts USARTE0\_CTRLB, r16 ;turn on TXEN and RXEN lines

ldi r16, 0x23

sts USARTE0\_CTRLC, r16 ;Parity = even, 8 bit frame, 1 stop bit

ldi r16, (BSel & 0xFF) ;select only lower 8 bits of BSel

sts USARTE0\_BAUDCTRLA, r16 ;set BAUDCTRLA to lower 8 bits of BSel

ldi r16, ((BScale << 4) & 0xF0) | ((BSel >> 8) & 0x0F)

sts USARTE0\_BAUDCTRLB, r16 ;set BAUDCTRLB to BScale | Bsel.

; Lower 4 bits are upper 4 bits of BSel.

; Upper 4 bits are upper 4 bits of BScale.

ret

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Name: USART\_INIT

\* Purpose: Subroutine to initialize the XMEGA USART system

\* Inputs: None

\* Outputs: None

\* Affected: r16, PORTE\_DIR, PORTE\_OUT, PORTQ\_DIR, PORTQ\_OUT

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

GPIO\_INIT:

ldi r16, 0x08

sts PORTE\_DIRSET, r16 ;set PORTE\_PIN3 as output for TX pin of USARTE0

sts PORTE\_OUTSET, r16 ;set the TX line to default to '1'

ldi r16, 0x04

sts PORTE\_DIRCLR, r16 ;set RX pin for input

ldi r16, 0xA

sts PORTQ\_DIRSET, r16 ;set pins 1 and 3 of PORTQ to output

sts PORTQ\_OUTCLR, r16 ;set USB\_SW\_EN = '0', USB\_SW\_SEL = '0'

ret

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Name: OUT\_CHAR

\* Purpose: Subroutine to output a single character to the transmit pin of the USART

\* Inputs: None

\* Outputs: Transmits data via USART

\* Affected: USARTE0\_STATUS, USARTE0\_DATA

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

OUT\_CHAR:

push r17

TX\_POLL:

lds r17, USARTE0\_STATUS ;load status register

sbrs r17, 5 ;proceed to writing out the char if

; the DREIF flag is set

rjmp TX\_POLL ;else go back to polling

sts USARTE0\_DATA, r16 ;send the character out over the USART

pop r17

ret

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Name: OUT\_STRING

\* Purpose: Subroutine to output character strings stored in memory

\* Inputs: None

\* Outputs: Transmits data via USART

\* Affected: r16

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

OUT\_STRING:

PRINT\_CHAR:

ld r16, Z+ ;load char pointed to by Z, POST-increment

cpi r16, 0x00 ;check if char is null

breq PRINT\_DONE ; if null -> DONE printing string

rcall OUT\_CHAR ; else OUTPUT that char

rjmp PRINT\_CHAR ;repeat

PRINT\_DONE:

ret

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Name: IN\_CHAR

\* Purpose: Subroutine to receive a single character from receiver pin of the USART

\* Inputs: None

\* Outputs: r16 loaded with input from SCI

\* Affected: r16, USARTE0\_STATUS, USARTE0\_DATA

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

IN\_CHAR:

RX\_POLL:

lds r16, USARTE0\_STATUS ;load the status register

sbrs r16, 7 ;proceed to reading in a char if

; the RXCIF flag is set

rjmp RX\_POLL ;else continue polling

lds r16, USARTE0\_DATA ;read the character into r16

ret

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Name: DELAY\_1MS

\* Purpose: Subroutine to delay for X\*10ms

\* Inputs: X stored in r20

\* Outputs: None

\* Affected: r20

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

.equ jcycles = 2

.equ kcycles = 246

DELAY\_1MS:

dec r20

rcall JLOOP\_INIT

cpi r20, 0

brne DELAY\_1MS

ret

JLOOP\_INIT:

push r20

ldi r20, jcycles

JLOOP:

dec r20

rcall KLOOP\_INIT

cpi r20, 0

brne JLOOP

pop r20

ret

KLOOP\_INIT:

push r20

ldi r20, kcycles

KLOOP:

dec r20

nop

brne KLOOP

pop r20

ret

Lab4\_serial\_menu.asm

/\* Lab4\_serial\_menu.asm

\*

\* Lab 4 Serial Menu

\* Name: Nicholas Imamshah

\* Section: 6957

\* TA Name: Daniel Gonzalez

\* Description: The purpose of this program is to provide a menu with responses by using the XMEGA USART System.

\*/

.nolist

.include "ATxmega128A1Udef.inc"

.list

.equ BSel = 289

.equ BScale = -7

.equ Prompt = '?'

.equ CR = 0x0D

.equ LF = 0x0A

.equ ESC = 0x1B

.equ TAB = 0x09

.equ NUL = 0x00

.org 0x100

MENU: .db "Nick's favorite:", CR, LF

.db "1.", TAB, "OS/Computer (Mac or PC)", CR, LF

.db "2.", TAB, "EE/CE Course ", CR, LF

.db "3.", TAB, "Hobby", CR, LF

.db "4.", TAB, "Quote", CR, LF

.db "5.", TAB, "Movie", CR, LF

.db "6.", TAB, "Re-display menu", CR, LF

.db "ESC: exit", CR, LF, CR, LF, NUL

FAV\_OS: .db "Windows, unless Danny is looking, then Linux ", CR, LF, CR, LF, NUL

FAV\_CE: .db "Design Patterns", CR, LF, CR, LF, NUL

FAV\_HO: .db "Gaming with friends", CR, LF, CR, LF, NUL

FAV\_QT: .db "From an archery instructor: 'The reason you are missing, is because you are focusing on the target and not on your actions.' ", CR, LF, CR, LF, NUL

FAV\_MV: .db "Kung Fury", CR, LF, CR, LF, NUL

PR\_ESC: .db "Done!", CR, LF, CR, LF, NUL

.org 0x0000

rjmp MAIN

.org 0x0200

MAIN:

rcall GPIO\_INIT

rcall USART\_INIT

REPEAT:

ldi r16, Prompt ;load Prompt character

ldi ZL, low(MENU << 1)

ldi ZH, high(MENU << 1)

rcall OUT\_STRING

rcall IN\_CHAR ;read input

rcall OUT\_CHAR ;echo to console

push r16 ;push to stack

ldi r16, CR ;CRLF

rcall OUT\_CHAR

ldi r16, LF

rcall OUT\_CHAR

pop r16 ;pop input off stack

cpi r16, '1'

breq OPT\_1 ;option 1 chosen

cpi r16, '2'

breq OPT\_2 ;option 2 chosen

cpi r16, '3'

breq OPT\_3 ;option 3 chosen

cpi r16, '4'

breq OPT\_4 ;option 4 chosen

cpi r16, '5'

breq OPT\_5 ;option 5 chosen

cpi r16, '6'

breq REPEAT ;option 6 chosen

cpi r16, ESC

breq OPT\_E ;option ESC chosen

rjmp REPEAT

OUTPUT:

rcall OUT\_STRING

rjmp REPEAT

OPT\_1:

ldi ZL, low(FAV\_OS << 1)

ldi ZH, high(FAV\_OS << 1)

rjmp OUTPUT

OPT\_2:

ldi ZL, low(FAV\_CE << 1)

ldi ZH, high(FAV\_CE << 1)

rjmp OUTPUT

OPT\_3:

ldi ZL, low(FAV\_HO << 1)

ldi ZH, high(FAV\_HO << 1)

rjmp OUTPUT

OPT\_4:

ldi ZL, low(FAV\_QT << 1)

ldi ZH, high(FAV\_QT << 1)

rjmp OUTPUT

OPT\_5:

ldi ZL, low(FAV\_MV << 1)

ldi ZH, high(FAV\_MV << 1)

rjmp OUTPUT

OPT\_E:

ldi ZL, low(PR\_ESC << 1)

ldi ZH, high(PR\_ESC << 1)

rcall OUT\_STRING

rjmp DONE

DONE:

rjmp DONE

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Name: USART\_INIT

\* Purpose: Subroutine to initialize the XMEGA USART system

\* Inputs: None

\* Outputs: None

\* Affected: r16, USARTD0\_CTRLB, USARTD0\_CTRLC, USARTD0\_BAUDCTRLA, USARTD0\_BAUDCTRLB

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

USART\_INIT:

ldi r16, 0x18

sts USARTD0\_CTRLB, r16 ;turn on TXEN and RXEN lines

ldi r16, 0x23

sts USARTD0\_CTRLC, r16 ;Parity = even, 8 bit frame, 1 stop bit

ldi r16, (BSel & 0xFF) ;select only lower 8 bits of BSel

sts USARTD0\_BAUDCTRLA, r16 ;set BAUDCTRLA to lower 8 bits of BSel

ldi r16, ((BScale << 4) & 0xF0) | ((BSel >> 8) & 0x0F)

sts USARTD0\_BAUDCTRLB, r16 ;set BAUDCTRLB to BScale | Bsel.

; Lower 4 bits are upper 4 bits of BSel.

; Upper 4 bits are upper 4 bits of BScale.

ret

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Name: GPIO\_INIT

\* Purpose: Subroutine to initialize the XMEGA GPIO for the USART System

\* Inputs: None

\* Outputs: None

\* Affected: r16, PORTD\_DIR, PORTD\_OUT, PORTQ\_DIR, PORTQ\_OUT

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

GPIO\_INIT:

ldi r16, 0x08

sts PORTD\_DIRSET, r16 ;set PORTD\_PIN3 as output for TX pin of USARTD0

sts PORTD\_OUTSET, r16 ;set the TX line to default to '1'

ldi r16, 0x04

sts PORTD\_DIRCLR, r16 ;set RX pin for input

ldi r16, 0xA

sts PORTQ\_DIRSET, r16 ;set pins 1 and 3 of PORTQ to output

sts PORTQ\_OUTCLR, r16 ;set USB\_SW\_EN = '0', USB\_SW\_SEL = '0'

ret

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Name: OUT\_CHAR

\* Purpose: Subroutine to output a single character to the transmit pin of the USART

\* Inputs: None

\* Outputs: Transmits data via USART

\* Affected: USARTD0\_STATUS, USARTD0\_DATA

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

OUT\_CHAR:

push r17

TX\_POLL:

lds r17, USARTD0\_STATUS ;load status register

sbrs r17, 5 ;proceed to writing out the char if

; the DREIF flag is set

rjmp TX\_POLL ;else go back to polling

sts USARTD0\_DATA, r16 ;send the character out over the USART

pop r17

ret

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Name: OUT\_STRING

\* Purpose: Subroutine to output character strings stored in memory

\* Inputs: Z pointing to desired output string

\* Outputs: Transmits data via USART

\* Affected: r16

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

OUT\_STRING:

;push r16

PRINT\_CHAR:

lpm r16, Z+ ;load char pointed to by Z, POST-increment

cpi r16, NUL ;check if char is null

breq PRINT\_DONE ; if null -> DONE printing string

rcall OUT\_CHAR ; else OUTPUT that char

rjmp PRINT\_CHAR ;repeat

;pop r16

PRINT\_DONE:

ret

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Name: IN\_CHAR

\* Purpose: Subroutine to receive a single character from receiver pin of the USART

\* Inputs: None

\* Outputs: r16 loaded with input from SCI

\* Affected: r16, USARTD0\_STATUS, USARTD0\_DATA

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

IN\_CHAR:

RX\_POLL:

lds r16, USARTD0\_STATUS ;load the status register

sbrs r16, 7 ;proceed to reading in a char if

; the RXCIF flag is set

rjmp RX\_POLL ;else continue polling

lds r16, USARTD0\_DATA ;read the character into r16

ret

Lab4\_serial\_int.asm

/\* Lab4\_serial\_int.asm

\*

\* Lab 4 Serial Interrupt

\* Name: Nicholas Imamshah

\* Section: 6957

\* TA Name: Daniel Gonzalez

\* Description: This is an interrupt driven echo program.

\*/

.nolist

.include "ATxmega128A1Udef.inc"

.list

.equ BSel = 289

.equ BScale = -7

.org 0x0000

rjmp MAIN

.org USARTD0\_RXC\_VECT

jmp ISR\_ECHO

.org 0x0200

MAIN:

rcall EBI\_INIT

rcall GPIO\_INIT

rcall USART\_INIT

rcall INTERRUPT\_INIT

ldi r16, 0x55

st X, r16

ldi r17, 0xFF ;for XORing to create a toggle

LOOP:

EOR r16, r17 ;toggle r16

st X, r16 ;output to LEDs

ldi r20, 50

rcall DELAY\_X\_MS

rjmp LOOP

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Name: OUT\_CHAR

\* Purpose: Subroutine to output a single character to the transmit pin of the USART

\* Inputs: None

\* Outputs: Transmits data via USART

\* Affected: USARTD0\_STATUS, USARTD0\_DATA

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

OUT\_CHAR:

push r17

TX\_POLL:

lds r17, USARTD0\_STATUS ;load status register

sbrs r17, 5 ;proceed to writing out the char if

; the DREIF flag is set

rjmp TX\_POLL ;else go back to polling

sts USARTD0\_DATA, r16 ;send the character out over the USART

pop r17

ret

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Name: USART\_INIT

\* Purpose: Subroutine to initialize the XMEGA USART system

\* Inputs: None

\* Outputs: None

\* Affected: r16, USARTD0\_CTRLB, USARTD0\_CTRLC, USARTD0\_BAUDCTRLA, USARTD0\_BAUDCTRLB

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

USART\_INIT:

ldi r16, 0x10

sts USARTD0\_CTRLA, r16 ;enable low-level interrupts on USART receive

ldi r16, 0x18

sts USARTD0\_CTRLB, r16 ;turn on TXEN and RXEN lines

ldi r16, 0x23

sts USARTD0\_CTRLC, r16 ;Parity = even, 8 bit frame, 1 stop bit

ldi r16, (BSel & 0xFF) ;select only lower 8 bits of BSel

sts USARTD0\_BAUDCTRLA, r16 ;set BAUDCTRLA to lower 8 bits of BSel

ldi r16, ((BScale << 4) & 0xF0) | ((BSel >> 8) & 0x0F)

sts USARTD0\_BAUDCTRLB, r16 ;set BAUDCTRLB to BScale | Bsel.

; Lower 4 bits are upper 4 bits of BSel.

; Upper 4 bits are upper 4 bits of BScale.

ret

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Name: GPIO\_INIT

\* Purpose: Subroutine to initialize the XMEGA GPIO for the USART System

\* Inputs: None

\* Outputs: None

\* Affected: r16, PORTD\_DIR, PORTD\_OUT, PORTQ\_DIR, PORTQ\_OUT

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

GPIO\_INIT:

ldi r16, 0x08

sts PORTD\_DIRSET, r16 ;set PORTD\_PIN3 as output for TX pin of USARTD0

sts PORTD\_OUTSET, r16 ;set the TX line to default to '1'

ldi r16, 0x04

sts PORTD\_DIRCLR, r16 ;set RX pin for input

ldi r16, 0xA

sts PORTQ\_DIRSET, r16 ;set pins 1 and 3 of PORTQ to output

sts PORTQ\_OUTCLR, r16 ;set USB\_SW\_EN = '0', USB\_SW\_SEL = '0'

ret

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Name: EBI\_INIT

\* Purpose: Subroutine to initialize Ports H,J,K/EBI for additional Input/Output ports

\* Inputs: None

\* Outputs: None

\* Affected: R16, X, Z, EBI\_CTRL, PORTH,J,K,

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

.set PORT = 0x288000

.set PORT\_END = 0x289FFF

EBI\_INIT:

ldi r16, 0x17 ;Configure PORTH bits 4, 2, 1, and 0 as outputs.

sts PORTH\_DIRSET, r16 ; These are the CS0(L), ALE1(H), RE(L), and WE(L) outputs.

; (CS0 is bit 4; ALE1 is bit 2; RE is bit 1; WE is bit 0)

ldi r16, 0x13 ;Default CS0(L), RE(L), and WE(L) to H = false.

sts PORTH\_OUTSET, r16 ; ALE defaults to 0 = L = false.

ldi r16, 0xFF ;Set all PORTK pins (A15-A0) to be outputs.

sts PORTK\_DIRSET, r16

ldi r16, 0xFF ;Set all PORTJ pins (D7-D0) to be outputs.

sts PORTJ\_DIRSET, r16

ldi r16, 0x01 ;Store 0x41 in EBI\_CTRL reg to select 3 port EBI(H,J,K)

sts EBI\_CTRL, r16 ; mode and SRAM ALE1 mode

;Initialize the Z pointer to point to the base address for CS0 in memory

ldi ZH, high(EBI\_CS0\_BASEADDR)

ldi ZL, low(EBI\_CS0\_BASEADDR)

;Load the middle byte (A15:8) of the three byte addr into a reg and store it as the

; LOW byte of the Base Address, BASEADDRL.

ldi r16, byte2(PORT)

st Z+, r16

;Load the highest byte (A23:16) of the three byte addr into a reg and store it as the

; HIGH byte of the Base Address, BASEADDRH.

ldi r16, byte3(PORT)

st Z, r16

ldi r16, 0x15 ;Set to 8KB CS space and turn on SRAM mode, 0x288000 - 0x289FFF

sts EBI\_CS0\_CTRLA, r16

;Steps for using the port expansion

ldi r16, byte3(PORT) ;initialize a pointer to point to the base addr of the PORT

sts CPU\_RAMPX, r16 ;use the CPU\_RAMPX reg to set the third byte of the pointer

ldi XH, high(PORT) ;set the middle (XH) and low (XL) bytes of the pointer as usual

ldi XL, low(PORT)

ret

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Name: INTERRUPT\_INIT

\* Purpose: Subroutine to initialize the PortE external pin interrupt PE0 using INT0

\* Inputs: None

\* Outputs: None

\* Affected: r16, PMIC\_CTRL, PORTE: \_INT0MASK\_OUT, \_DIRCLR, \_INTCTRL, \_PIN0CTRL

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

INTERRUPT\_INIT:

ldi r16, 0x01 ;enable low-level interrupts

sts PMIC\_CTRL, r16

sei ;set global interrupt flag LAST!

ret

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Name: ISR\_ECHO

\* Purpose: Interrupt service routine to echo char on USART receive pin to transmit

\* Inputs: None

\* Outputs: None

\* Affected: USARTD0\_STATUS

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

ISR\_ECHO:

push r16 ;push any registers used to ensure they can be used after

lds r16, USARTD0\_DATA ;read character into r16

rcall OUT\_CHAR ;echo character

ldi r16, 0x80

sts USARTD0\_STATUS, r16 ;ensure RXCIF is cleared

pop r16 ;restore registers used in ISR

reti

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Name: DELAY\_X\_MS

\* Purpose: Subroutine to delay for X\*10ms

\* Inputs: X stored in r20

\* Outputs: None

\* Affected: r20

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

.equ jcycles = 20

.equ kcycles = 246

DELAY\_X\_MS:

dec r20

rcall JLOOP\_INIT

cpi r20, 0

brne DELAY\_X\_MS

ret

JLOOP\_INIT:

push r20

ldi r20, jcycles

JLOOP:

dec r20

rcall KLOOP\_INIT

cpi r20, 0

brne JLOOP

pop r20

ret

KLOOP\_INIT:

push r20

ldi r20, kcycles

KLOOP:

dec r20

nop

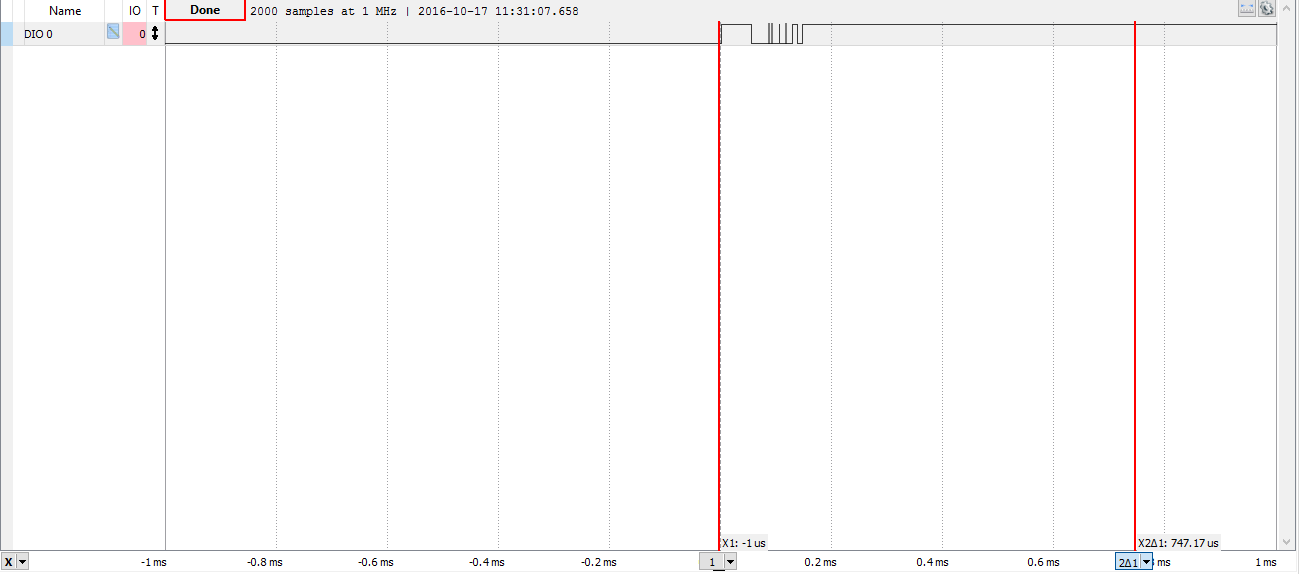
brne KLOOP

pop r20

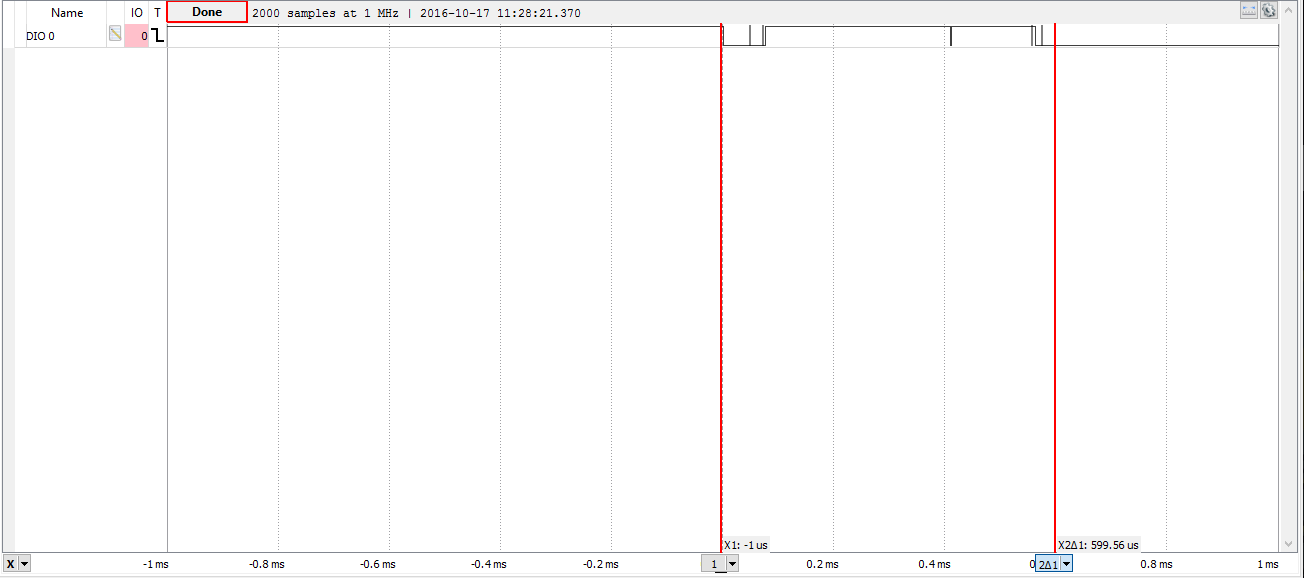
ret

**i) Appendix**

Low-to-High Bouncing



High-to-Low Bouncing



Baud Rate Verification

