REQUIREMENTS NOT MET

N/A

PROBLEMS ENCOUNTERED

N/A

FUTURE WORK/APPLICATIONS

Almost all types of communications between computers is performed serially. Such as SPI, USART, USB, and others not mentioned. This lab allows me to understand how these communication protocols work at the hardware level.

PRE-LAB EXERCISES

i. The sampling rate of a UART receiver is usually faster than the baud rate of the overall system. Why is this so?

The receivers needs to be able to read data at any time since its asynchronous. In addition, 3 samples need to be taken in order for the clock synchronizer to determine if it's a legitimate start bit or just noise. This requires a faster sampling rate on the receiver.

ii. What is the maximum possible baud rate for asynchronous communication within the USART system of the ATxmega128A1U, assuming that the microcontroller has a system clock frequency of 2 MHz and that the USART "double-speed mode" is disabled (i.e., the relevant bit CLK2X is set to 0)? In addition to the maximum rate, provide the values of the relevant registers used to configure that rate. Whenever appropriate, support your answer with calculations.

BAUDCTRLA would need to be 0b00000000
BAUDCTRLB would need to be 0b00000000

$$Baud \leq \frac{2000000}{16} = 125000\ BPS$$

$$Baud = \frac{f_{Per}}{2^{BScale} \cdot 16(BSel+1)} = \frac{2000000}{2^{BScale} \cdot 16(BSel+1)} = \frac{2000000}{2^{0} \cdot 16(0+1)} = \frac{2000000}{1 \cdot 16} = 125000\ BPS$$

Figure 1: Baud rate calculations

iii. In the context of the USART system within the ATxmega128A1U, how many buffers (i.e., memory locations that store temporary data) are used by a transmitter? How many are used by a receiver? Additionally, for both transmitters and receivers, explain how the use of buffers provides greater flexibility to an application involving these components.

The transmitter contains 2 registers:
1: The shift register, which is used for transmitting receiving data
2. The data register, which is used for holding data while the shift register receives or transmits data

The receiver contains 3 registers
1: The shift register, which is used for receiving data
2 and 3: The data register, which is used for holding data for parity checking and synchronization while the shift register receives data

The buffers allow data to be received/transmitted while being written to/read from by the microcontroller.

iv. If an asynchronous serial communication protocol of 7 data bits, one start bit, one stop bits, odd parity, and baud rate of 15.6 kHz was chosen, calculate how many seconds it would take to transmit the ASCII character string "Dr. Schwartz saw seven slick slimy snakes slowly sliding southward." (This string has 67 characters.) Note that ASCII is a 7-bit (not an 8-bit) code. Show all work.

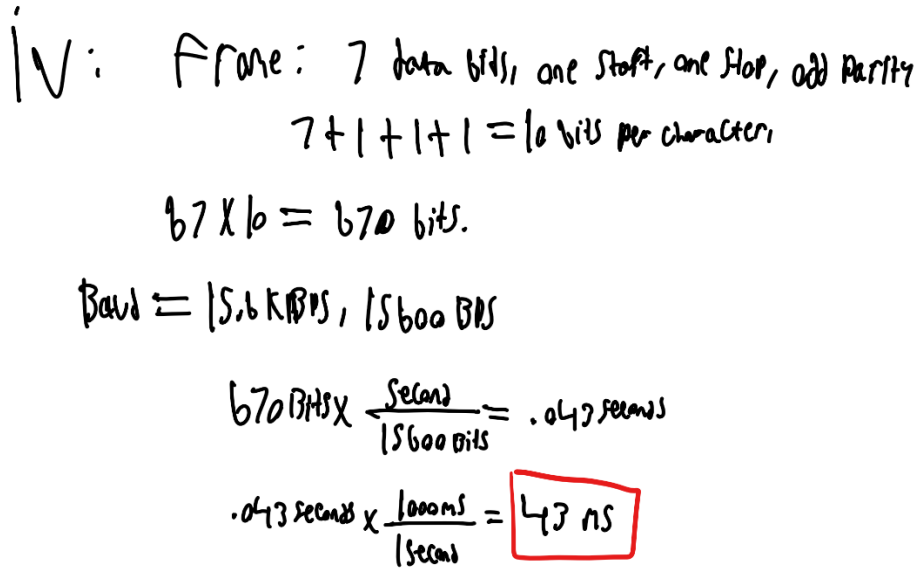

Figure 2: Baud rate calculations

PSEUDOCODE/FLOWCHARTS

SECTION 2

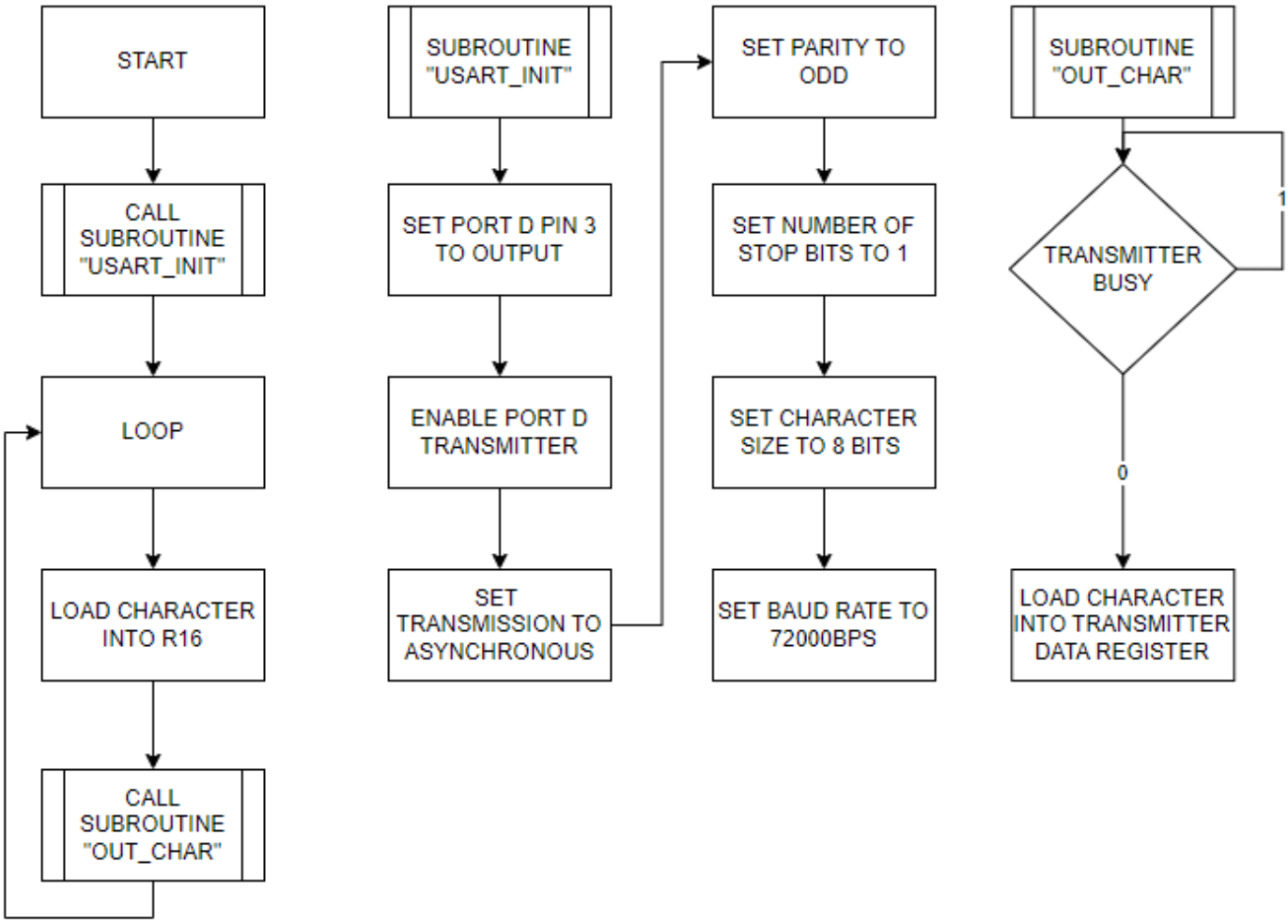

Figure 3: Flowchart for "lab5_2.asm"

SECTION 3

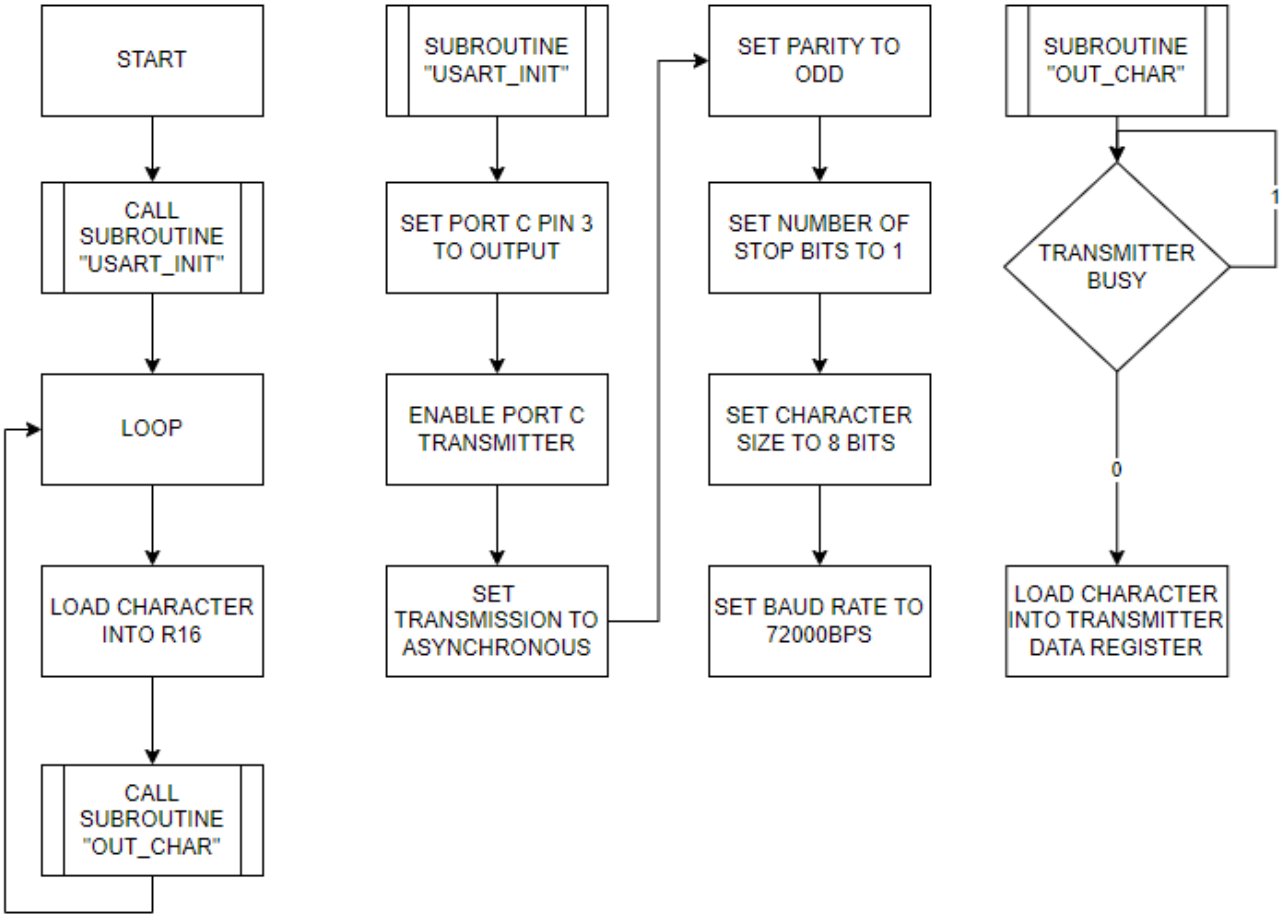

Figure 4: Flowchart for "lab5_3.asm"

SECTION 4

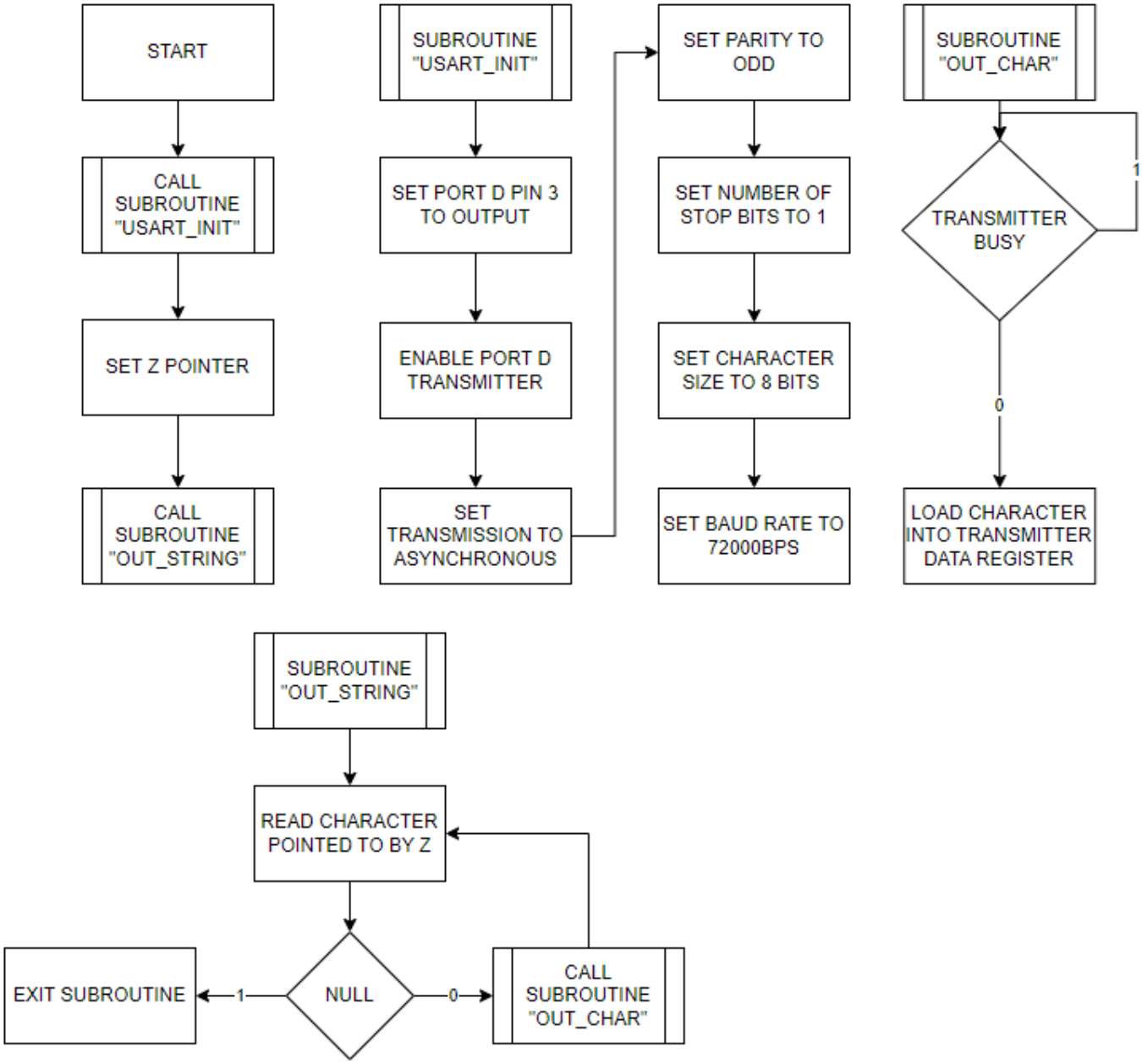

Figure 5: Flowchart for "lab5_4.asm"

SECTION 5

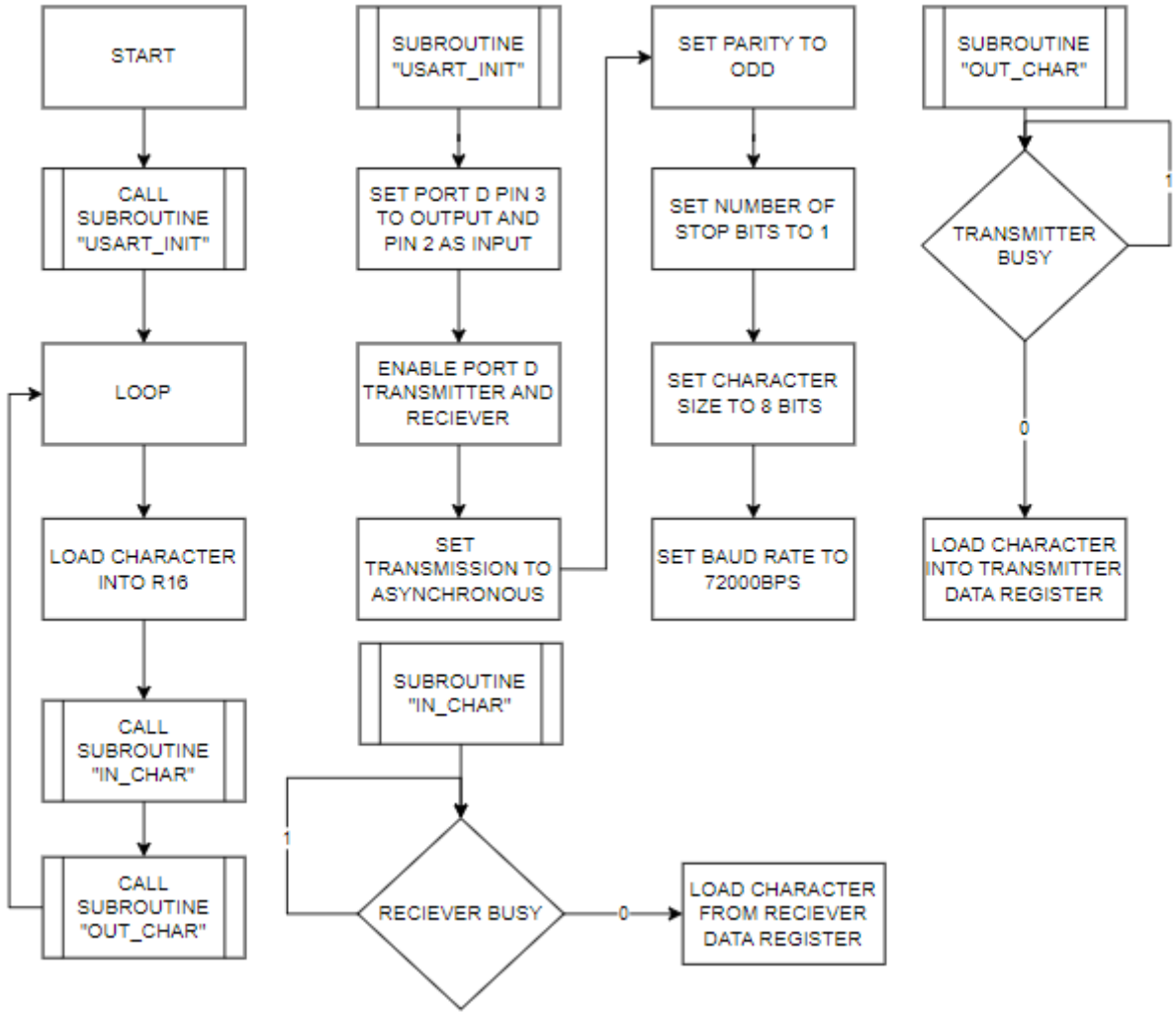

Figure 6: Flowchart for "lab5_5.asm"

SECTION 6

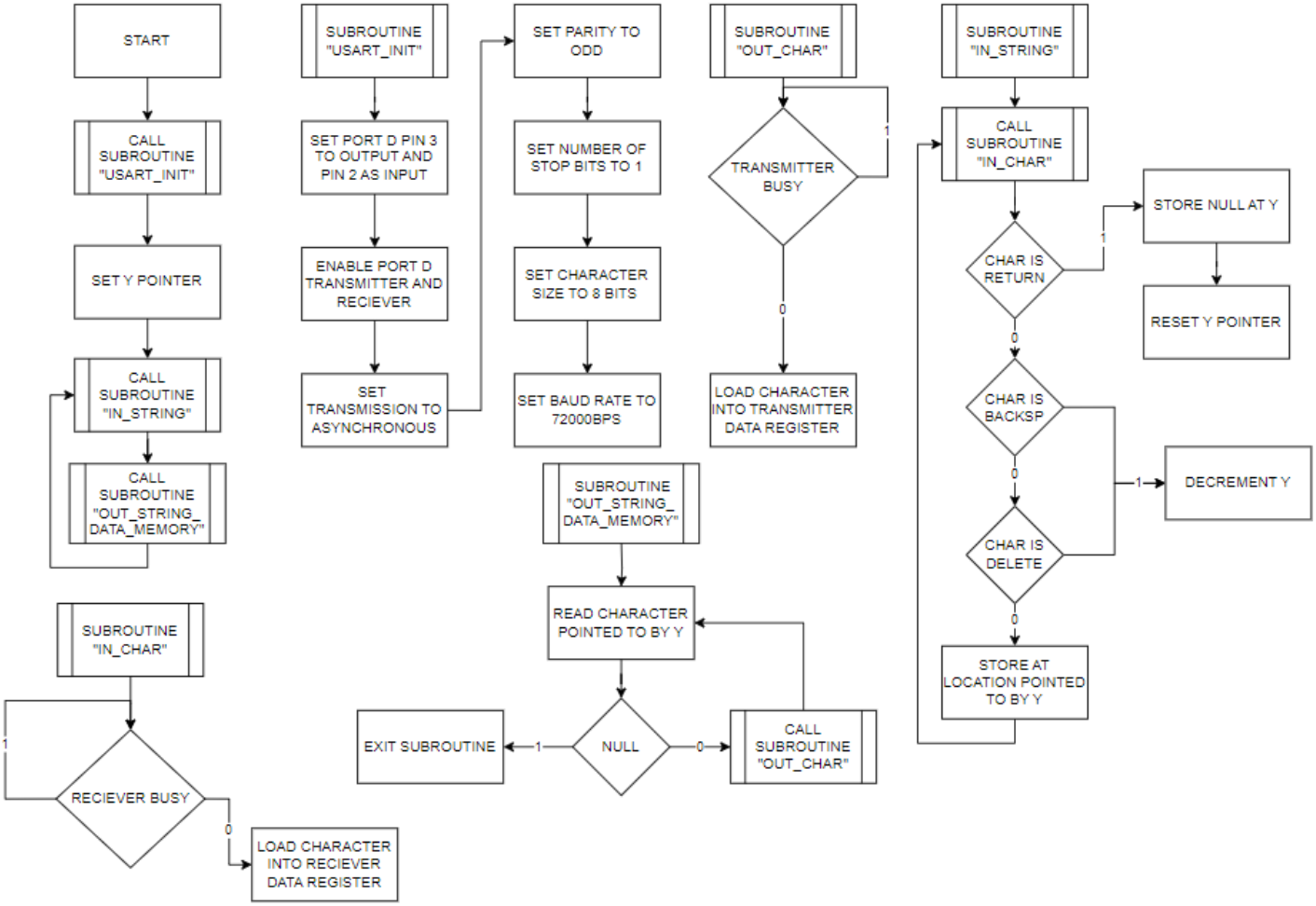

Figure 7: Flowchart for "lab5_6.asm"

SECTION 7

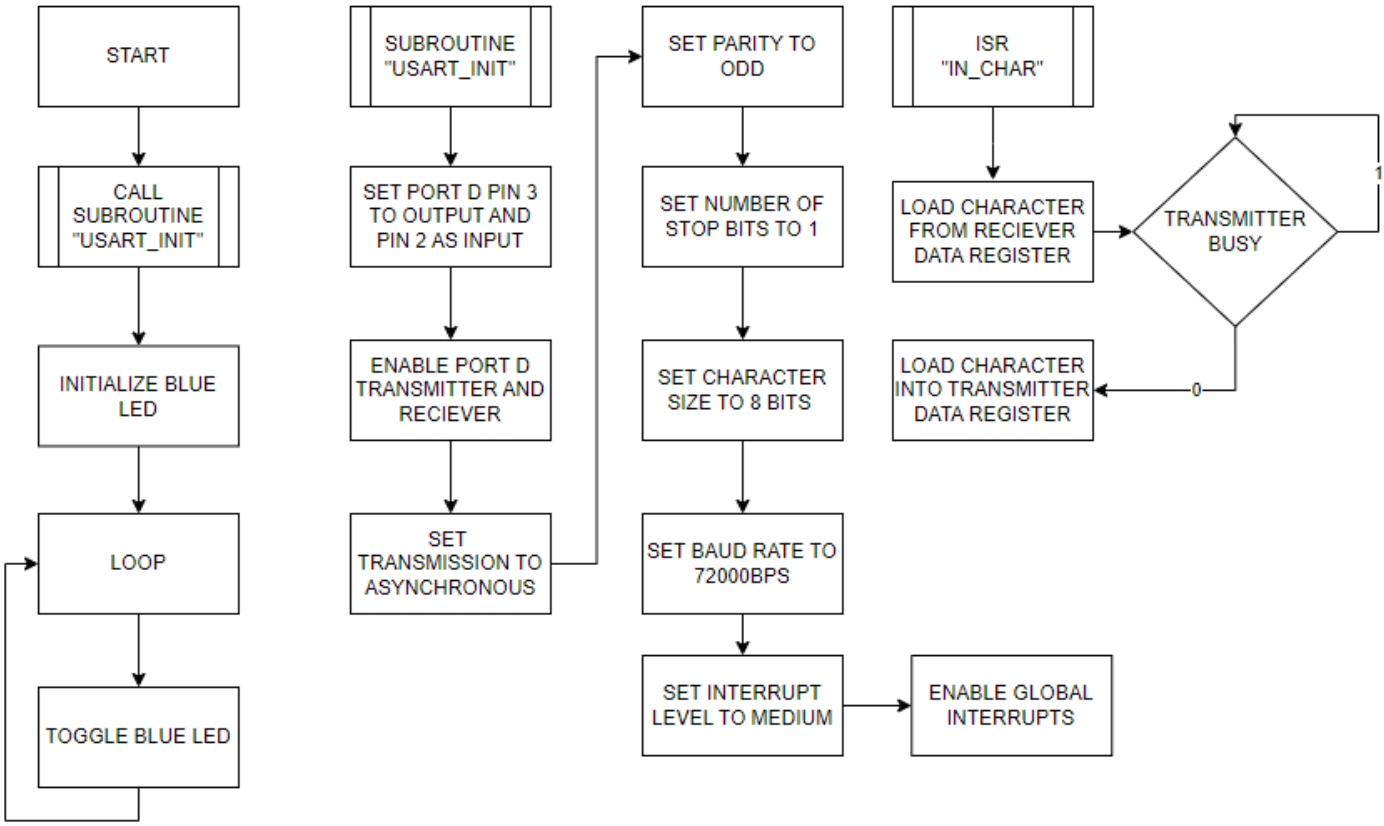

Figure 8: Flowchart for "lab5_7.asm"

PROGRAM CODE

SECTION 2

```
;****************************************
;Lab 5, Section 2
;Name: Steven Miller
;Class #: 11318
;PI Name: Anthony Stross
;Description: transmits character over serial usb line
;****************************************

;***************INCLUDES*****************************************
.include "ATxmega128a1udef.inc"
;***************END OF INCLUDES*****************************

;*********************************EQUATES*********************************
.equ bsel = 47
.equ bscale = -6
;*********************************END OF EQUATES*****************************

;*********************************DEFS*********************************
;*********************************END OF DEFS*********************************

;***********PROGRAM MEMORY CONFIGURATION*************************

;***********END OF PROGRAM MEMORY CONFIGURATION***************

;***********DATA MEMORY CONFIGURATION*************************

;***********END OF DATA MEMORY CONFIGURATION***************

;***********MAIN PROGRAM*********************************
.CSEG
.org 0x0000
	rjmp main
.org 0x0200

main:
	;initialize stack
	ldi r16, low(0x3fff)
	out CPU_SPL, r16
	ldi r16, high(0x3fff)
	out CPU_SPH, r16
	rcall USART_INIT
	loop:
		ldi r16,'U'
		rcall OUT_CHAR
	rjmp loop
end:
rjmp end
```

```
;***************************************************
; Name: USART_INIT
; Purpose: INITIALIZE USART MODULE ON PORT D
; Input(s): N/A
; Output: N/A
;***************************************************
USART_INIT:
	push r16
	;set port d pin 3 as output
	ldi r16, 0b00001000
	sts PORTD_OUTSET,r16
	sts PORTD_DIRSET, r16
	;enable transmitter
	ldi r16, 0b00001000
	sts USARTD0_CTRLB, r16
	;set transmission to asynchronous and parity to odd
	;and set number of stop bits to 1 and set character size to 8 bits
	ldi r16,(USART_PMODE_ODD_gc|USART_CMODE_ASYNCHRONOUS_gc|USART_CHSIZE_8BIT_gc)
	sts USARTD0_CTRLC, r16
	;set baud rate to 72000 bps
	;bsel = 47
	;bscale = -6
	ldi r16, low(bsel)
	sts USARTD0_BAUDCTRLA, r16
	ldi r16, ((bscale<<4) | (high(bsel))) ;1010 = -6
	sts USARTD0_BAUDCTRLB, r16
	pop r16
ret
;***************************************************
; Name: OUT_CHAR
; Purpose:TRANSMIT CHARACTER OUT OF PORT D TO USB
; Input(s): N/A
; Output: USARTD0_DATA
;***************************************************
OUT_CHAR:
	push r17
	;check if transmitter busy
	transmitter_busy:
	lds r17, USARTD0_STATUS
	sbrs r17, USART_DREIF_bp
	rjmp transmitter_busy
	;ldi r17, 0b0
	;load character into transmitter data register
	sts USARTD0_DATA, r16
	pop r17
ret
```

SECTION 3

```
;****************************************
;Lab 5, Section 3
;Name: Steven Miller
;Class #: 11318
;PI Name: Anthony Stross
;Description: transmits character over serial usb line but using a scope readable pin
;****************************************

;***************INCLUDES*************************************
.include "ATxmega128a1udef.inc"
;***************END OF INCLUDES*****************************

;********************************EQUATES*********************************
.equ bsel = 47
.equ bscale = -6
;********************************END OF EQUATES*********************************

;********************************DEFS*********************************
;********************************END OF DEFS*********************************

;***********PROGRAM MEMORY CONFIGURATION************************

;***********END OF PROGRAM MEMORY CONFIGURATION***************

;***********DATA MEMORY CONFIGURATION************************

;***********END OF DATA MEMORY CONFIGURATION***************

;***********MAIN PROGRAM*********************************
.CSEG
.org 0x0000
	rjmp main
.org 0x0200

main:
	;initialize stack
	ldi r16, low(0x3fff)
	out CPU_SPL, r16
	ldi r16, high(0x3fff)
	out CPU_SPH, r16
	rcall USART_INIT
	loop:
		ldi r16,'U'
		rcall OUT_CHAR
	rjmp loop
end:
rjmp end
```

```
;***************************************************
; Name: USART_INIT
; Purpose: INITIALIZE USART MODULE ON PORT C
; Input(s): N/A
; Output: N/A
;***************************************************
USART_INIT:
	push r16
	;set port C pin 3 as output
	ldi r16, 0b00001000
	sts PORTC_OUTSET,r16
	sts PORTC_DIRSET, r16
	;enable transmitter
	ldi r16, 0b00001000
	sts USARTC0_CTRLB, r16
	;set transmission to asynchronous and parity to odd
	;and set number of stop bits to 1 and set character size to 8 bits
	ldi r16,(USART_PMODE_ODD_gc|USART_CMODE_ASYNCHRONOUS_gc|USART_CHSIZE_8BIT_gc)
	sts USARTC0_CTRLC, r16
	;set baud rate to 72000 bps
	;bsel = 47
	;bscale = -6
	ldi r16, low(bsel)
	sts USARTC0_BAUDCTRLA, r16
	ldi r16, ((bscale<<4) | (high(bsel))) ;1010 = -6
	sts USARTC0_BAUDCTRLB, r16
	pop r16
ret
;***************************************************
; Name: OUT_CHAR
; Purpose:TRANSMIT CHARACTER OUT OF PORT C TO USB
; Input(s): N/A
; Output: USARTC0_DATA
;***************************************************
OUT_CHAR:
	push r17
	;check if transmitter busy
	transmitter_busy:
	lds r17, USARTC0_STATUS
	sbrs r17, USART_DREIF_bp
	rjmp transmitter_busy
	;ldi r17, 0b0
	;load character into transmitter data register
	sts USARTC0_DATA, r16
	pop r17
ret
```

SECTION 4

```
;****************************************
;Lab 5, Section 4
;Name: Steven Miller
;Class #: 11318
;PI Name: Anthony Stross
;Description: transmits character string over serial usb line
;****************************************

;***************INCLUDES*************************************
.include "ATxmega128a1udef.inc"
;***************END OF INCLUDES******************************

;*********************************EQUATES*********************************
.equ bsel = 47
.equ bscale = -6
;*********************************END OF EQUATES*********************************

;*********************************DEFS*********************************
;*********************************END OF DEFS*********************************

;***********PROGRAM MEMORY CONFIGURATION************************
.CSEG
.ORG 0X3744
TABLE:
.DB "STEVEN MILLER"
;***********END OF PROGRAM MEMORY CONFIGURATION***************

;***********DATA MEMORY CONFIGURATION************************

;***********END OF DATA MEMORY CONFIGURATION**************

;***********MAIN PROGRAM*********************************
.CSEG
.org 0x0000
	rjmp main
.org 0x0200

main:
	;initialize stack
	ldi r16, low(0x3fff)
	out CPU_SPL, r16
	ldi r16, high(0x3fff)
	out CPU_SPH, r16
	rcall USART_INIT
	;set z pointer

	ldi ZL, byte1(TABLE<<1)
	ldi ZH, byte2(table<<1)
	ldi r16, byte3(table<<1)
	sts CPU_RAMPZ, r16

	rcall OUT_STRING

end:
rjmp end
```

```
;***************************************************
; Name: USART_INIT
; Purpose: INITIALIZE USART MODULE ON PORT D
; Input(s): N/A
; Output: N/A
;***************************************************
USART_INIT:
	push r16
	;set port d pin 3 as output
	ldi r16, 0b00001000
	sts PORTD_OUTSET,r16
	sts PORTD_DIRSET, r16
	;enable transmitter
	ldi r16, 0b00001000
	sts USARTD0_CTRLB, r16
	;set transmission to asynchronous and parity to odd
	;and set number of stop bits to 1 and set character size to 8 bits
	ldi r16,(USART_PMODE_ODD_gc|USART_CMODE_ASYNCHRONOUS_gc|USART_CHSIZE_8BIT_gc)
	sts USARTD0_CTRLC, r16
	;set baud rate to 72000 bps
	;bsel = 47
	;bscale = -6
	ldi r16, low(bsel)
	sts USARTD0_BAUDCTRLA, r16
	ldi r16, ((bscale<<4) | (high(bsel))) ;1010 = -6
	sts USARTD0_BAUDCTRLB, r16
	pop r16
ret
;***************************************************
; Name: OUT_CHAR
; Purpose:TRANSMIT CHARACTER OUT OF PORT D TO USB
; Input(s): N/A
; Output: USARTD0_DATA
;***************************************************
OUT_CHAR:
	push r17
	;check if transmitter busy
	transmitter_busy:
	lds r17, USARTD0_STATUS
	sbrs r17, USART_DREIF_bp
	rjmp transmitter_busy
	;ldi r17, 0b0
	;load character into transmitter data register
	sts USARTD0_DATA, r16
	pop r17
ret
```

```
;***************************************************
; Name: OUT_STRING
; Purpose:TRANSMIT STRING OUT OF PORT D TO USB
; Input(s): N/A
; Output:N/A
;***************************************************
OUT_STRING:
	push r16
	read_string:
	;read character pointed to by z
	elpm r16, z+
	;check if null
	cpi r16, 0x00
	breq null
	not_null:
	rcall OUT_CHAR
	rjmp read_string
	null:
	pop r16
ret
```

SECTION 5

```
;****************************************
;Lab 5, Section 5
;Name: Steven Miller
;Class #: 11318
;PI Name: Anthony Stross
;Description: recieves serial data from computer
;****************************************

;***************INCLUDES*************************************
.include "ATxmega128a1udef.inc"
;***************END OF INCLUDES*****************************

;*********************************EQUATES*********************************
.equ bsel = 47
.equ bscale = -6
;*********************************END OF EQUATES*********************************

;*********************************DEFS*********************************
;*********************************END OF DEFS*********************************

;***********PROGRAM MEMORY CONFIGURATION*************************
;***********END OF PROGRAM MEMORY CONFIGURATION***************

;***********DATA MEMORY CONFIGURATION*************************

;***********END OF DATA MEMORY CONFIGURATION***************

;***********MAIN PROGRAM*****************************
.CSEG
.org 0x0000
	rjmp main
.org 0x0200

main:
	;initialize stack
	ldi r16, low(0x3fff)
	out CPU_SPL, r16
	ldi r16, high(0x3fff)
	out CPU_SPH, r16
	rcall USART_INIT
	loop:
		rcall IN_CHAR
		rcall OUT_CHAR
	rjmp loop
end:
rjmp end

;*************************************************
; Name: USART_INIT
; Purpose: INITIALIZE USART MODULE ON PORT D
; Input(s): N/A
; Output: N/A
;*************************************************
USART_INIT:
	push r16
	;set port d pin 2 as input and port d pin 3 as output
	ldi r16, 0b00000100
	sts PORTD_OUTCLR,r16
```

```
	sts PORTD_DIRCLR, r16
	ldi r16, 0b00001000
	sts PORTD_OUTSET,r16
	sts PORTD_DIRSET, r16
	;enable transmitter and reciever
	ldi r16, 0b00011000
	sts USARTD0_CTRLB, r16
	;set transmission to asynchronous and parity to odd
	;and set number of stop bits to 1 and set character size to 8 bits
	ldi r16,(USART_PMODE_ODD_gc|USART_CMODE_ASYNCHRONOUS_gc|USART_CHSIZE_8BIT_gc)
	sts USARTD0_CTRLC, r16
	;set baud rate to 72000 bps
	;bsel = 47
	;bscale = -6
	ldi r16, low(bsel)
	sts USARTD0_BAUDCTRLA, r16
	ldi r16, ((bscale<<4) | (high(bsel))) ;1010 = -6
	sts USARTD0_BAUDCTRLB, r16
	pop r16
ret
;***************************************************
; Name: OUT_CHAR
; Purpose:TRANSMIT CHARACTER OUT OF PORT D TO USB
; Input(s): N/A
; Output: USARTD0_DATA
;***************************************************
OUT_CHAR:
	push r17
	;check if transmitter busy
	transmitter_busy:
	lds r17, USARTD0_STATUS
	sbrs r17, USART_DREIF_bp
	rjmp transmitter_busy
	;load character into transmitter data register
	sts USARTD0_DATA, r16
	pop r17
ret

;***************************************************
; Name: IN_CHAR
; Purpose: RECIEVES CHARACTER FROM USB TO RECIEVER
; Input(s):USARTD0_DATA
; Output:N/A
;***************************************************
IN_CHAR:
	push r17
	;check if reciever busy
	reciever_busy:
	lds r17, USARTD0_STATUS
	sbrs r17, USART_RXCIF_bp
	rjmp reciever_busy
	;load character into reciever data register
	lds r16, USARTD0_DATA
	pop r17
ret
```

SECTION 6

```
;****************************************
;Lab 5, Section 6
;Name: Steven Miller
;Class #: 11318
;PI Name: Anthony Stross
;Description: recieves serial data from computer and allows backspace
;****************************************

;***************INCLUDES*************************************
.include "ATxmega128a1udef.inc"
;***************END OF INCLUDES*****************************

;*********************************EQUATES*********************************
.equ bsel = 47
.equ bscale = -6
.EQU SRAM_TABLE_LOC = 0x2000
.EQU SRAM_TABLE_SIZE = 500
.EQU return_char = 0x0d
.EQU backspace_char = 0x08
.EQU delete_char = 0x7f
;*********************************END OF EQUATES*****************************

;*********************************DEFS************************************
;*********************************END OF DEFS*****************************

;***********PROGRAM MEMORY CONFIGURATION*************************
;***********END OF PROGRAM MEMORY CONFIGURATION***************

;***********DATA MEMORY CONFIGURATION*************************
.DSEG
.org SRAM_TABLE_LOC
DATA_MEMORY:
.BYTE SRAM_TABLE_SIZE

;***********END OF DATA MEMORY CONFIGURATION***************

;***********MAIN PROGRAM*********************************
.CSEG
.org 0x0000
	rjmp main
.org 0x0200

main:
	;initialize stack
	ldi r16, low(0x3fff)
	out CPU_SPL, r16
	ldi r16, high(0x3fff)
	out CPU_SPH, r16
	rcall USART_INIT
	ldi yl, low(sram_table_loc)
	ldi yh, high(sram_table_loc)
	loop:
		rcall IN_STRING
		rcall OUT_STRING_DATA_MEMORY
	rjmp loop
end:
rjmp end
```

```
;***************************************************
; Name: USART_INIT
; Purpose: INITIALIZE USART MODULE ON PORT D
; Input(s): N/A
; Output: N/A
;***************************************************
USART_INIT:
	push r16
	;set port d pin 2 as input and port d pin 3 as output
	ldi r16, 0b00000100
	sts PORTD_OUTCLR,r16
	sts PORTD_DIRCLR, r16
	ldi r16, 0b00001000
	sts PORTD_OUTSET,r16
	sts PORTD_DIRSET, r16
	;enable transmitter and reciever
	ldi r16, 0b00011000
	sts USARTD0_CTRLB, r16
	;set transmission to asynchronous and parity to odd
	;and set number of stop bits to 1 and set character size to 8 bits
	ldi r16,(USART_PMODE_ODD_gc|USART_CMODE_ASYNCHRONOUS_gc|USART_CHSIZE_8BIT_gc)
	sts USARTD0_CTRLC, r16
	;set baud rate to 72000 bps
	;bsel = 47
	;bscale = -6
	ldi r16, low(bsel)
	sts USARTD0_BAUDCTRLA, r16
	ldi r16, ((bscale<<4) | (high(bsel))) ;1010 = -6
	sts USARTD0_BAUDCTRLB, r16
	pop r16
ret
;***************************************************
; Name: OUT_CHAR
; Purpose:TRANSMIT CHARACTER OUT OF PORT D TO USB
; Input(s): N/A
; Output: USARTD0_DATA
;***************************************************
OUT_CHAR:
	push r17
	;check if transmitter busy
	transmitter_busy:
	lds r17, USARTD0_STATUS
	sbrs r17, USART_DREIF_bp
	rjmp transmitter_busy
	;load character into transmitter data register
	sts USARTD0_DATA, r16
	pop r17
ret

;***************************************************
; Name: IN_CHAR
; Purpose: RECIEVES CHARACTER FROM USB TO RECIEVER
; Input(s):USARTD0_DATA
; Output:N/A
;***************************************************
IN_CHAR:
	push r17
	;check if reciever busy
	reciever_busy:
	lds r17, USARTD0_STATUS
	sbrs r17, USART_RXCIF_bp
```

```
	rjmp reciever_busy
	;load character into reciever data register
	lds r16, USARTD0_DATA
	pop r17
ret
;***************************************************
; Name: IN_STRING
; Purpose: RECIEVES CHARACTER STRING FROM USB TO RECIEVER
; Input(s):USARTD0_DATA
; Output:N/A
;***************************************************
IN_STRING:

	read_string:
		RCALL IN_CHAR
		;check if character is return
		cpi r16, return_char
		breq is_return
		;check if character is backspace
		cpi r16, backspace_char
		breq is_backspace_or_delete
		;check if character is delete
		cpi r16, delete_char
		breq is_backspace_or_delete
		;if none
		st y+, r16
		rjmp read_string
		;if character is backspace or delete
		is_backspace_or_delete:
		sbiw y, 1
	rjmp read_string
	;if return
	is_return:
	ldi r16, 0x00
	st y, r16
	;reset y pointer
	ldi yl, low(sram_table_loc)
	ldi yh, high(sram_table_loc)
ret
;***************************************************
; Name: OUT_STRING_DATA_MEMORY
; Purpose:TRANSMIT CHARACTER STRING IN DATA MEMORY OUT OF PORT D TO USB
; Input(s): N/A
; Output: USARTD0_DATA
;***************************************************
OUT_STRING_DATA_MEMORY:
	push r16
	read_string_data_mem:
		;read character pointed to by y
		ld r16, y+
		;check if null
		cpi r16, 0x00
		breq null
		not_null:
		rcall OUT_CHAR
		rjmp read_string_data_mem
		null:
		;ldi yl, low(sram_table_loc)
		;ldi yh, high(sram_table_loc)
	pop r16
ret
```

SECTION 7

```
;****************************************
;Lab 5, Section 7
;Name: Steven Miller
;Class #: 11318
;PI Name: Anthony Stross
;Description: recieves serial data from computer using interrupts
;****************************************
;*********************************EQUATES*********************************
.equ bsel = 47
.equ bscale = -6
;*****************************END OF EQUATES*****************************

;*********************************DEFS*********************************
;*****************************END OF DEFS*****************************

;***********PROGRAM MEMORY CONFIGURATION************************
;***********END OF PROGRAM MEMORY CONFIGURATION***************

;***********DATA MEMORY CONFIGURATION************************

;***********END OF DATA MEMORY CONFIGURATION***************

;***********MAIN PROGRAM*********************************
.CSEG
.org USARTD0_RXC_vect
	rjmp in_char
.org 0x0000
	rjmp main
.org 0x0200

main:
	;initialize stack
	ldi r16, low(0x3fff)
	out CPU_SPL, r16
	ldi r16, high(0x3fff)
	out CPU_SPH, r16
	rcall USART_INIT
	;initialize blue led
	ldi r16, 0b01000000
	sts PORTD_DIRSET,r16
	ldi r16, 0b10111111
	sts PORTD_OUT,r16
	ldi r16, 0b01000000
	loop:
		sts PORTD_OUTTGL,r16
	rjmp loop
end:
rjmp end
```

```
;***************************************************
; Name: USART_INIT
; Purpose: INITIALIZE USART MODULE ON PORT D
; Input(s): N/A
; Output: N/A
;***************************************************
USART_INIT:
	push r16
	;set port d pin 2 as input and port d pin 3 as output
	ldi r16, 0b00000100
	sts PORTD_OUTCLR,r16
	sts PORTD_DIRCLR, r16
	ldi r16, 0b00001000
	sts PORTD_OUTSET,r16
	sts PORTD_DIRSET, r16
	;enable transmitter and reciever
	ldi r16, 0b00011000
	sts USARTD0_CTRLB, r16
	;set transmission to asynchronous and parity to odd
	;and set number of stop bits to 1 and set character size to 8 bits
	ldi r16,(USART_PMODE_ODD_gc|USART_CMODE_ASYNCHRONOUS_gc|USART_CHSIZE_8BIT_gc)
	sts USARTD0_CTRLC, r16
	;set baud rate to 72000 bps
	;bsel = 47
	;bscale = -6
	ldi r16, low(bsel)
	sts USARTD0_BAUDCTRLA, r16
	ldi r16, ((bscale<<4) | (high(bsel))) ;1010 = -6
	sts USARTD0_BAUDCTRLB, r16
	;set interrupt level to medium
	ldi r16, 0b00100000
	sts USARTD0_CTRLA, r16
	;enable global interrupts
	ldi r16, 0b00000010
	sts pmic_ctrl, r16
	sei
	pop r16
ret
```

```
;***************************************************
; Name: IN_CHAR
; Purpose: reciever ISR
; Input(s):USARTD0_DATA
; Output:N/A
;***************************************************
IN_CHAR:
	push r17
	push r16
	lds r16, CPU_SREG
	push r16
	;load character from reciever data register
	lds r18, USARTD0_DATA
	;check if transmitter busy
	transmitter_busy:
	lds r17, USARTD0_STATUS
	sbrs r17, USART_DREIF_bp
	rjmp transmitter_busy
	;load character into transmitter data register
	sts USARTD0_DATA, r18
	pop r16
	sts CPU_SREG, r16
	pop r16
	pop r17
reti
```

APPENDIX

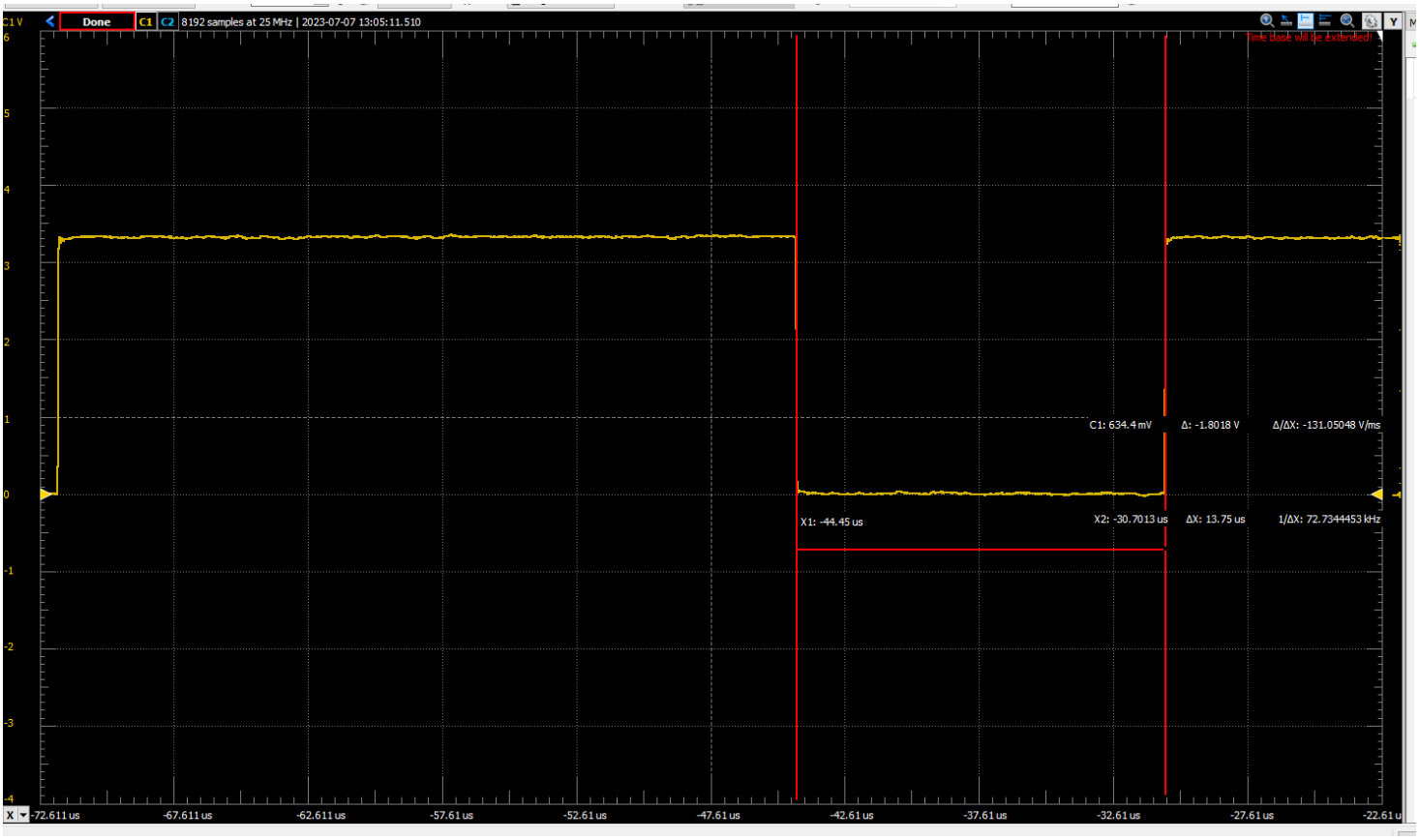

Figure 9: Measurement of first data bit

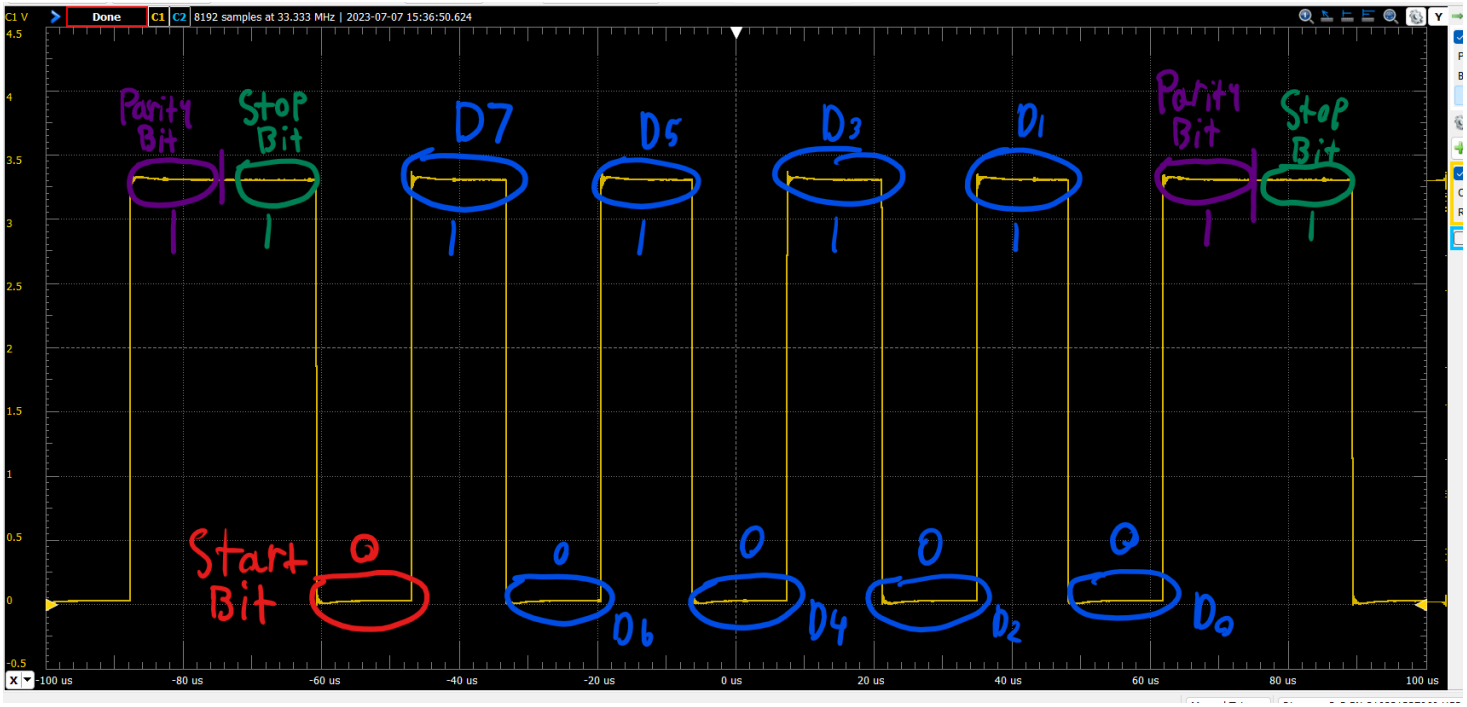

Figure 10: Measurement of data frame