

```
/*
 * asynch_sw_read.c
 *
 * Created: 2/15/2022 8:55:44 AM
 * Author : jason
 */

#include <avr/io.h>
#define F_CPU 4000000
#include <util/delay.h>

#define BAUD_RATE 9600 //baud rate value 2

//#define USART3_BAUD_RATE(BAUD_RATE) ((float)(F_CPU * 64 / (16 * (float)BAUD_RATE)) +
0.5)

//header functions that will be in use
uint8_t UART_sw_read();

//global character variable to store the recent data
char receiveData;
uint8_t startBit = 1;
uint8_t rxData;

int main(void)
{
    //unsigned char receiveData;
    //USART Full Duplex Initialization
    PORTB.DIR = ~PIN1_bm; //Set PB1 as input to receiving data

    while (1)
    {
        receiveData = (char)UART_sw_read(); //Received character from transmitter
    }
}

/*
The USART receiver samples the RX line to detect and
interpret the received data. The RX pin must be configured
as an input.

? The receiver accepts data when its hardware detects a valid
start bit. The data is shifted into the Receive Shift register
until the first Stop bit is received.

? The contents of the Receive Shift register is loaded into the
```

receive buffer and the Receive Complete Interrupt Flag (the RXCIF bit in the USARTn.STATUS register) is set.

? The RXDATA registers are the part of the double-buffered RX buffer that can be read by the application software when RXCIF is set.

\*/

```
uint8_t UART_sw_read(){
    uint8_t tempStoreReceiveData;
    uint8_t ReceiverData [8];
```

```
    startBit = 0;          //Start having the data being received by setting it to 0
```

```
    //Sample at falling edge when RX pin is 0 and check start bit is 0
    while(startBit == 0){
```

```
        //Purpose is to check the false start at half the start bit
```

```
        if(BAUD_RATE == 4800){
            _delay_us(208.3/2);
```

```
        }
        if(BAUD_RATE == 9600){
            _delay_us(104.2/2);
```

```
        }
        if(BAUD_RATE == 19200){
            _delay_us(52.1/2);
```

```
        }
    }
```

```
    //Check in the middle of bit time if PB1 is actually 0 or not. If PB1 is not
    //0, else go back in the beginning.
```

```
    if(startBit != 0){
        continue;
    }
```

```
    //False start to check the conditions if true or just wait till next start bit
```

```
    while(startBit == 0){
```

```
        //Bit time to be able to read data from the RX pin
```

```
        if(BAUD_RATE == 4800){
            _delay_us(208.3);
```

```
        }
        if(BAUD_RATE == 9600){
            _delay_us(104.2);
```

```
        }
        if(BAUD_RATE == 19200){
            _delay_us(52.1);
```

```
        }
    }
```

```

//Bit time for corresponding baud ratings
for(uint8_t i = 0; i < 8; i++){

    //Try to read each bit that is stored into PB1 to receive
    tempStoreReceiveData = (PORTB_IN & 0x02);

    //Align so that the bit data fetched from PB1 only exist in bit 0
    position
    tempStoreReceiveData >>= 1;

    //Mask to only care about the one bit bit every time the bit from PB1
    is read
    ReceiverData[i] = (tempStoreReceiveData & 0x01);

    //Bit time to receive each bit that is transmitted
    if(USART3.BAUD == 4800){
        _delay_us(208.3);
    }
    if(USART3.BAUD == 9600){
        _delay_us(104.2);
    }
    if(USART3.BAUD == 19200){
        _delay_us(52.1);
    }
}

//Store all of the data bits read from RX pin into RX output
rxData = ReceiverData[0] << 0 | ReceiverData[1] << 1 | ReceiverData[2] <<
2 |
ReceiverData[3] << 3 | ReceiverData[4] << 4 | ReceiverData[5] << 5 |
ReceiverData[6] << 6 | ReceiverData[7] << 7;

    startBit = 1;
}
    startBit = 1;
}
startBit = 1;

return (char) rxData;

}

```