

```
/*
 * usart_avr128.c
 *
 * Created: 3/3/2022 4:20:17 PM
 * Author : jason
 */

#define F_CPU 4000000
#define USART3_BAUD_RATE(BAUD_RATE) ((float)(F_CPU * 64 / (16 *(float)BAUD_RATE))) // ↗
    Calculation of baud rate from data sheet
#include <avr/io.h>
#include <util/delay.h>
#include <avr/interrupt.h>

/* UART Buffer Defines */
#define USART_RX_BUFFER_SIZE 128    /* 2,4,8,16,32,64,128 or 256 bytes */
#define USART_TX_BUFFER_SIZE 128    /* 2,4,8,16,32,64,128 or 256 bytes */
#define USART_RX_BUFFER_MASK ( USART_RX_BUFFER_SIZE - 1 )
#define USART_TX_BUFFER_MASK ( USART_TX_BUFFER_SIZE - 1 )

#if ( USART_RX_BUFFER_SIZE & USART_RX_BUFFER_MASK )
    #error RX buffer size is not a power of 2
#endif

#if ( USART_TX_BUFFER_SIZE & USART_TX_BUFFER_MASK )
    #error TX buffer size is not a power of 2
#endif

/* Static Variables */
static unsigned char USART_RxBuf[USART_RX_BUFFER_SIZE];
static volatile unsigned char USART_RxHead;
static volatile unsigned char USART_RxTail;
static unsigned char USART_TxBuf[USART_TX_BUFFER_SIZE];
static volatile unsigned char USART_TxHead;
static volatile unsigned char USART_TxTail;
volatile int flag;

/* Prototypes */
void USART3_Init( unsigned int baudrate );
unsigned char USART3_Receive( void );
void USART3_Transmit( unsigned char data );

int main(void)
{
    unsigned int baudRate = 9600; //Baud rate value

    USART3_Init(baudRate); //function to initialize for USART
    sei(); //Enable interrupts => enable USART3 interrupts
    while(1)
    {
```

```

    USART3_Transmit(USART3_Receive()); /* Echo the received character */
}
}

//Initialize USART
void USART3_Init(unsigned int baudrate){
    unsigned char x;
    PORTB_DIR &= PIN1_bm; //Set PB1 as the input (RX pin)
    PORTB_DIR |= PIN0_bm; //Set PB0 as output (TX pin)
    USART3_BAUD = (uint16_t)USART3_BAUD_RATE(baudrate); //Taken from data sheet to ↗
        calculate baud rate
    USART3_CTRLB |= USART_TXEN_bm | USART_RXEN_bm; //Enable USART transmitter and ↗
        receiver
    USART3_CTRLA |= USART_RXCIE_bm; //Enable the Receive complete interrupt

    //Guess set the default to being asynchronous, disable parity, 1 stop bit, 8 bits

    //Flush receive buffer
    x = 0;

    USART_RxTail = x;
    USART_RxHead = x;
    USART_TxTail = x;
    USART_TxHead = x;
}

//Interrupt service routine for the receive
ISR(USART3_RXC_vect){
    unsigned char data;
    unsigned char tmphead;

    //Read the received data
    data = USART3_RXDATA1;

    /*Calculate the buffer index */
    tmphead = (USART_RxHead + 1) & USART_RX_BUFFER_MASK;
    USART_RxHead = tmphead; //Store new index

    if(tmphead == USART_RxTail){
    }
    USART_RxBuf[tmphead] = data; //Store received data in buffer
}

//Interrupt service routine for the transmit
ISR(USART3_DRE_vect){
    unsigned char tmptail;

    //Check if all data is transmitted
    if(USART_TxHead != USART_TxTail){
        //Calculate buffer index
    }
}

```

```

    tmptail = (USART_TxTail + 1) & USART_TX_BUFFER_MASK;
    USART_TxTail = tmptail;    //Store new index

    USART3_TXDATAL = USART_TxBuf[tmptail]; //Starts transmission
}
else{
    //Disable the USART data register empty interrupt enable
    USART3_CTRLA &= ~(1 << USART_DREIE_bp); //define USART_DREIE_bp 5 /* Data ↗
    Register Empty Interrupt Enable bit position. */
}
}

//Read and write function
unsigned char USART3_Receive(void){
    unsigned char tmptail;

    while(USART_RxHead == USART_RxTail); //Wait for incoming data

    tmptail = (USART_RxTail + 1) & USART_RX_BUFFER_MASK; //Calculate buffer index

    USART_RxTail = tmptail; //Store new index

    return USART_RxBuf[tmptail];    //return data
}

//Transmit the data
void USART3_Transmit(unsigned char data){
    unsigned char tmphead;

    //Calculate buffer index
    tmphead = (USART_TxHead + 1) & USART_TX_BUFFER_MASK; //Wait for free space in the ↗
    buffer

    while(tmphead == USART_TxTail);

    USART_TxBuf[tmphead] = data; //Store data in buffer
    USART3_TXDATAL = USART_TxBuf[tmphead];
    USART_TxHead = tmphead;    //Store new index

    USART3_CTRLA |= (1<< USART_DREIE_bp);    //Enable the DREIE interrupt
}

//Data received in the buffer
unsigned char DataInReceiveBuffer(void){
    /* Return 0 (FALSE) if the receive buffer is empty */
    return (USART_RxHead != USART_RxTail);
}

```

