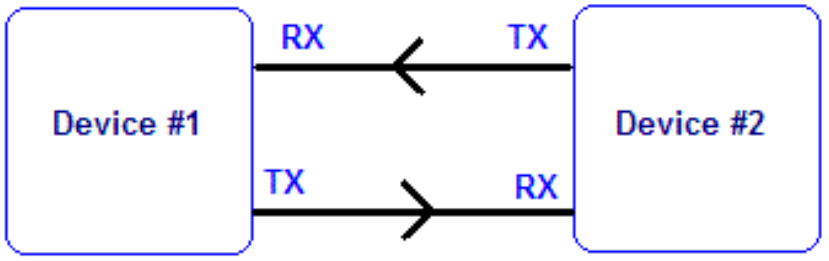
Serielle Schnittstellen

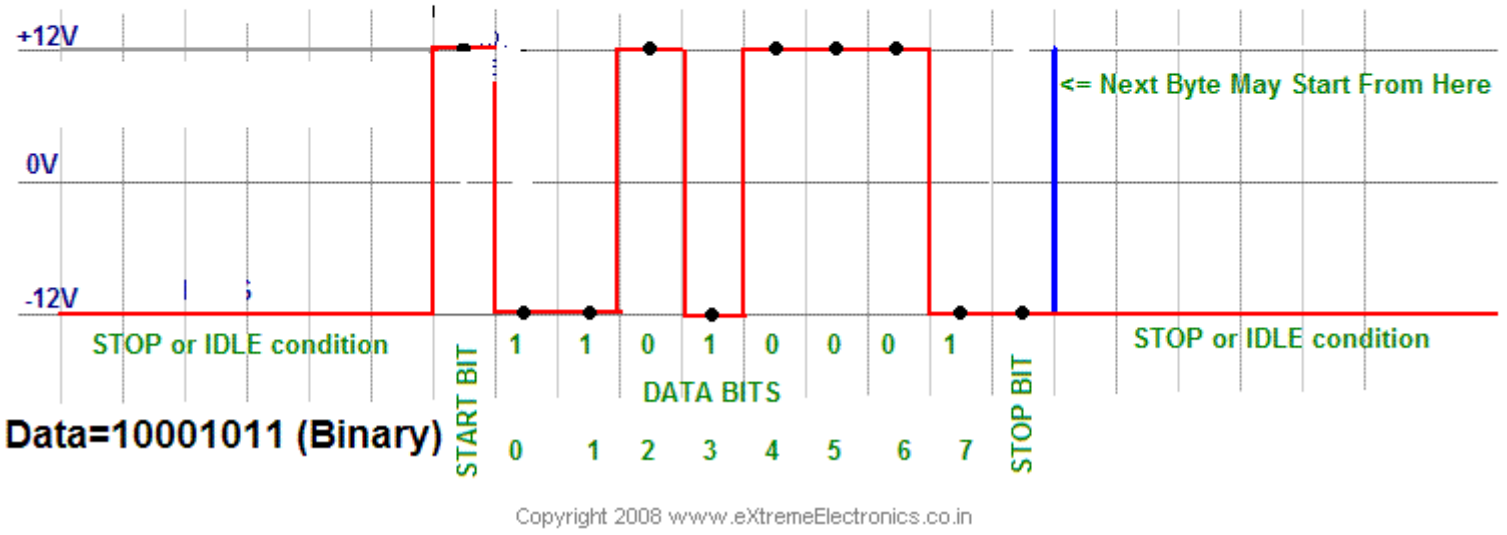
Einfuehrung
Projektlabor 2011

Eine Leitung zum Senden, eine zum Empfangen

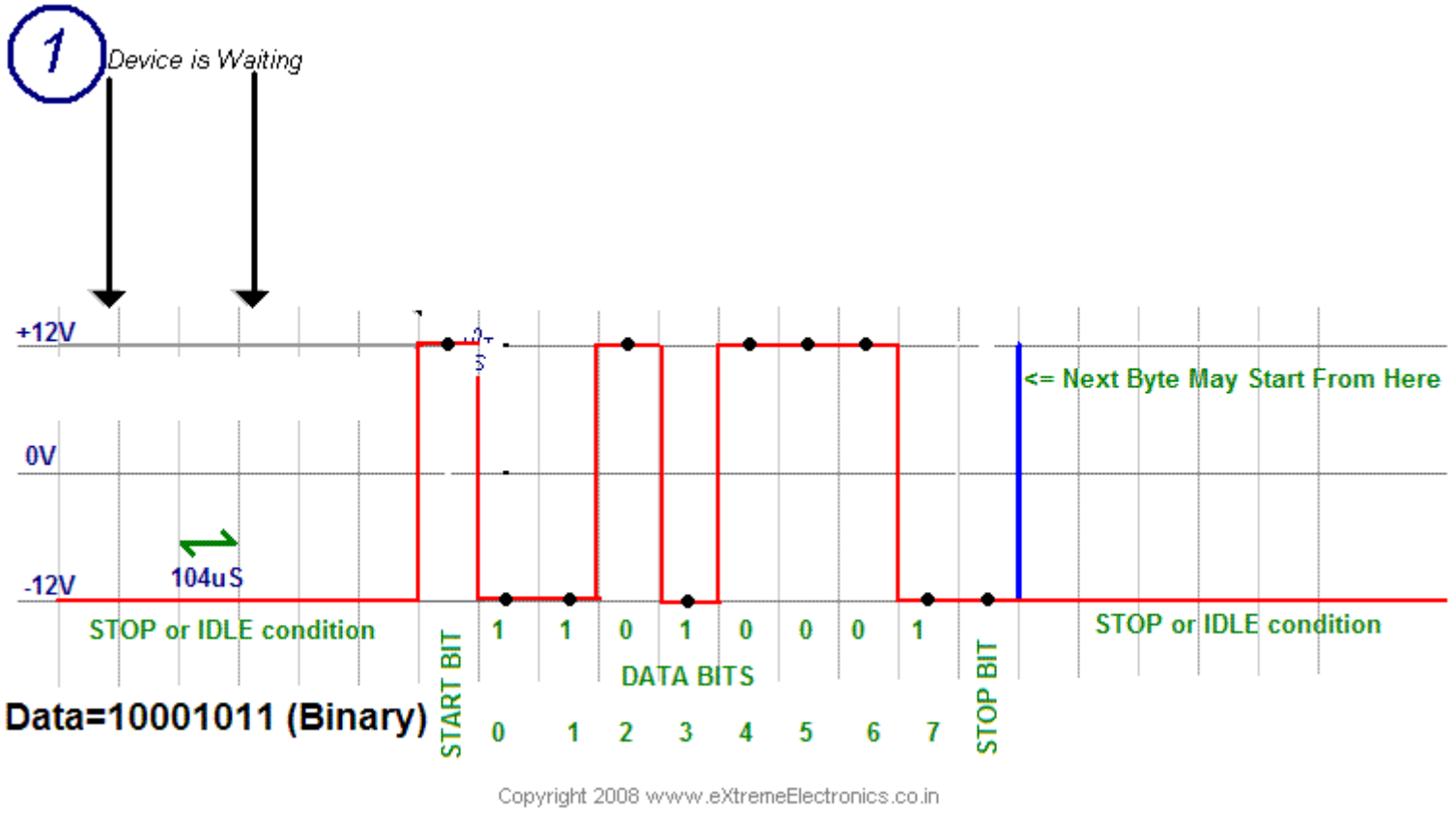


- serieller Datenstrom
- Rx,Tx und GND

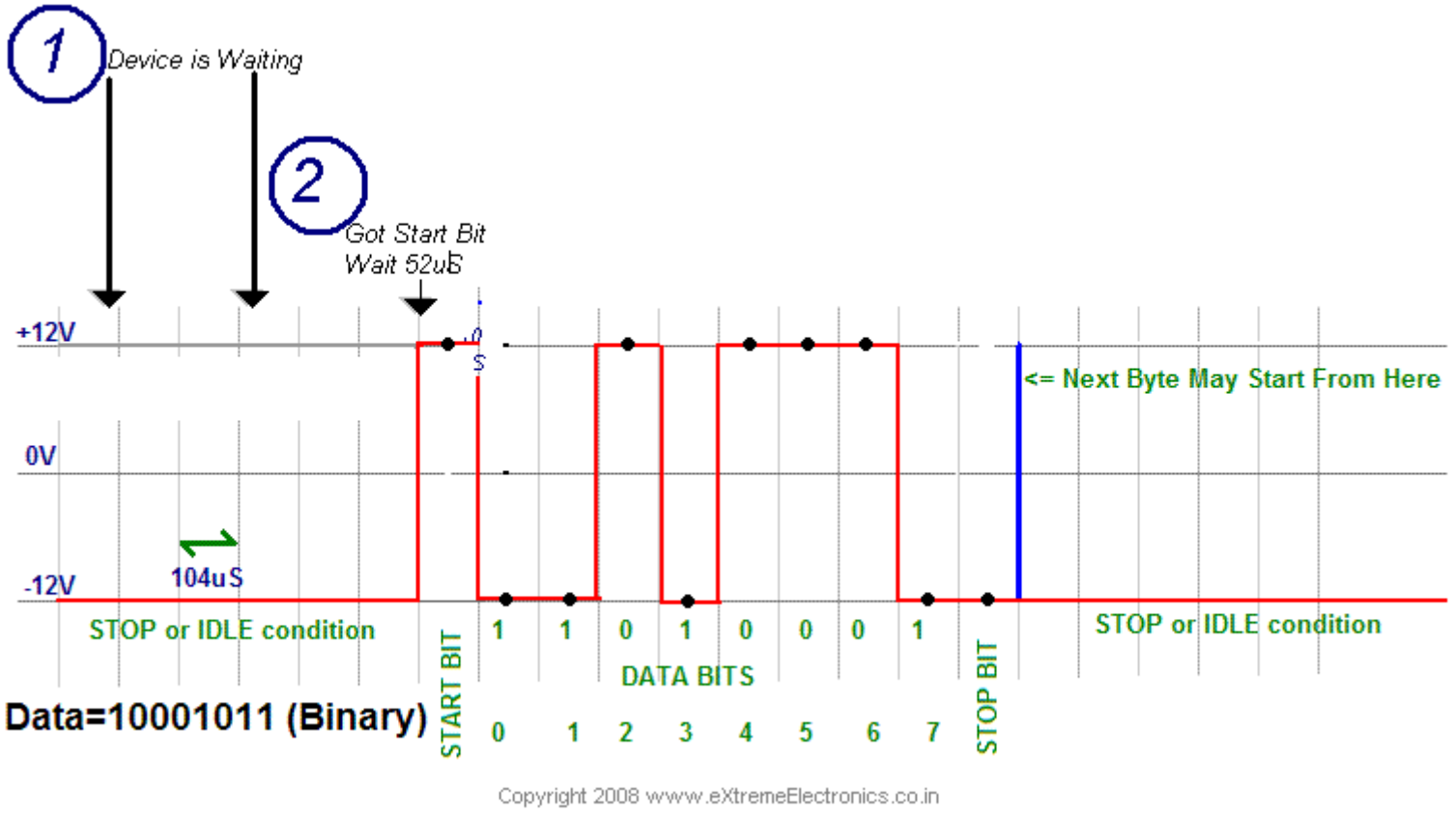
Senden


+12V
0V
-12V
<= Next Byte May Start From Here
STOP or IDLE condition
START BIT
1 1 0 1 0 0 0 1
DATA BITS
STOP BIT
STOP or IDLE condition
Data=10001011 (Binary)
0 1 2 3 4 5 6 7
Copyright 2008 www.eXtremeElectronics.co.in

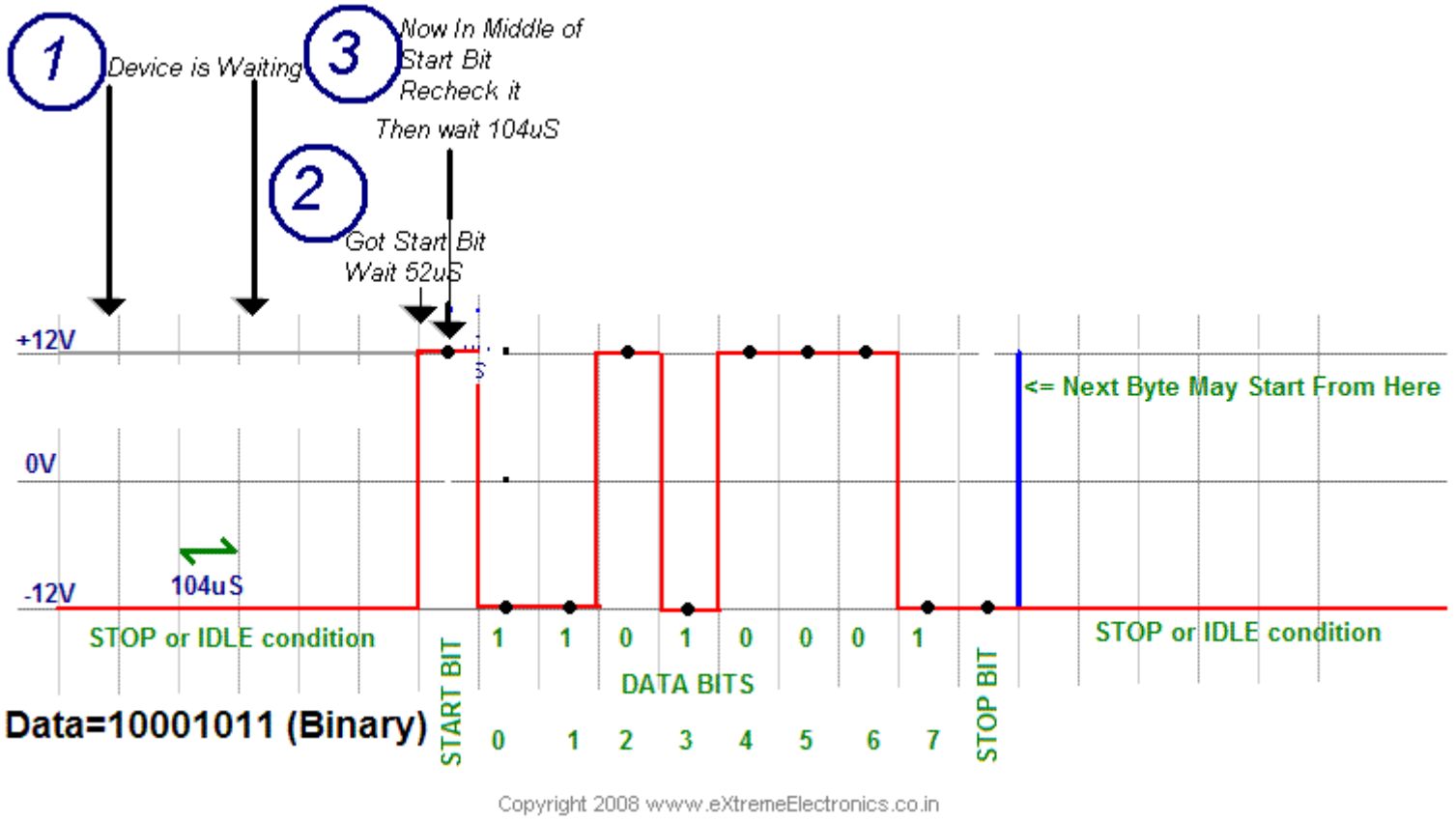
Empfangen


1
Device is Waiting
+12V
0V
-12V
104uS
<= Next Byte May Start From Here
STOP or IDLE condition
START BIT
1 1 0 1 0 0 0 1
DATA BITS
STOP BIT
STOP or IDLE condition
Data=10001011 (Binary)
0 1 2 3 4 5 6 7
Copyright 2008 www.eXtremeElectronics.co.in

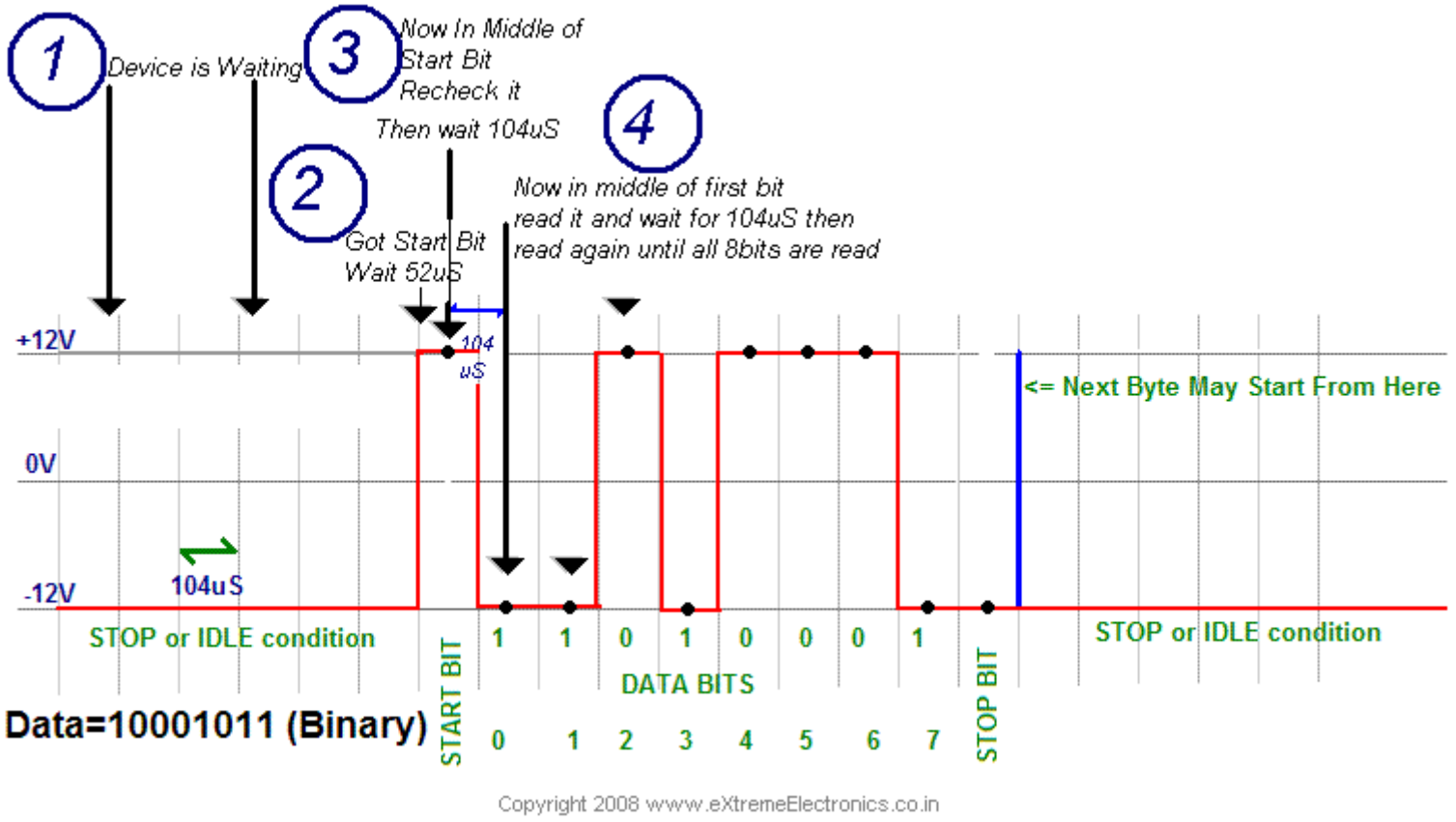
Empfangen



Empfangen



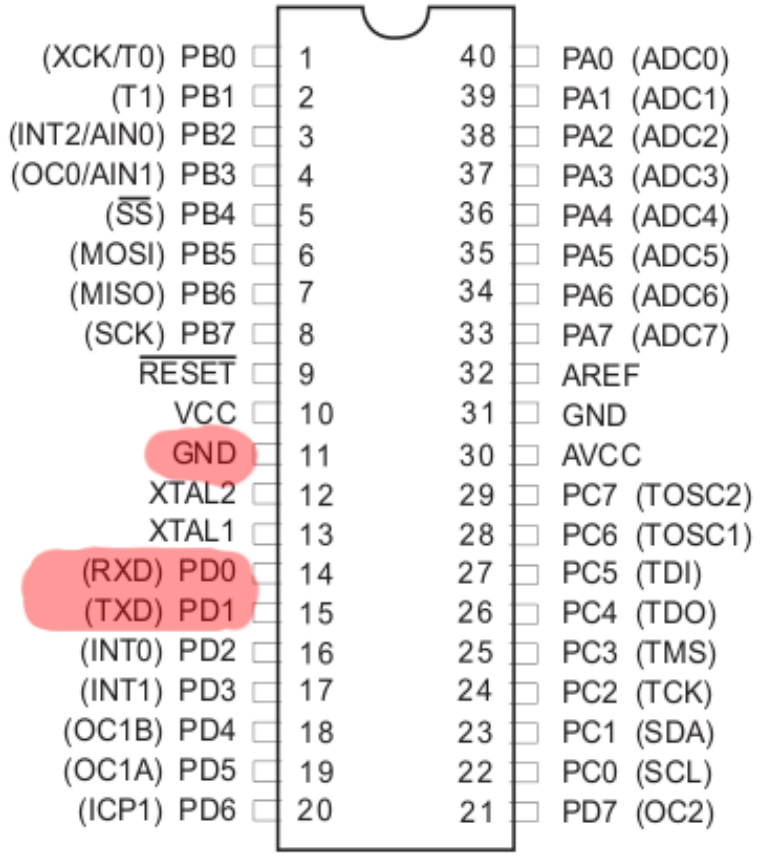
Empfangen



Pinouts am ATmega16

USART in den Mikrocontroller integriert

zwei Leitungen, keine zusaetzlichen Bausteine



Register

Schnittstellenparameter,
Interruptkonfiguration,
Daten

Codebeispiel & Anwendung

Senden und Empfangen mit
zwei Mikrocontrollern

```c
//Baudratenregister, 8MHz, 19200
#define UBRR_VAL 25

//Timer0 Overflow IRQ
ISR(TIMER0_OVF_vect)
{
	//warten auf TX-Complete
	while (!(UCSRA & (1<<UDRE)));

	//Tasterport lesen und versenden...
	UDR = PINB;
}

//FUNCTION "main"
int main (void)
{

	//PORTB als Input
	DDRB = 0x00;

	//Timer initialisieren: prescaler 1024, 2mhz quarz, overflow-IRQ
	TCCR0 |= (1<<CS00)|(1<<CS02);
	TIMSK |= (1<<TOIE0);

	//UART init: 8N1, nur TX
	UCSRB |= (1<<TXEN);                              // TX einschalten
	UCSRC = (1<<URSEL)|(1<<UCSZ1)|(1<<UCSZ0);        // 8N1

	//Baudratenregister schreiben
	UBRRH = UBRR_VAL >> 8;
	UBRRL = UBRR_VAL & 0xFF;

	//IRQs global aktivieren
	sei();

	while(1);
	return 0;
}
```

```c
//Baudratenregister, 8MHz, 19200
#define UBRR_VAL 25

//Timer0 Overflow IRQ
ISR(TIMER0_OVF_vect)
{
        //warten auf TX-Complete
        while (!(UCSRA & (1<<UDRE)));

        //Tasterport lesen und versenden...
        UDR = PINB;
}

//FUNCTION "main"
int main (void)
{

        //PORTB als Input
        DDRB = 0x00;

        //Timer initialisieren: prescaler 1024, 2mhz quarz, overflow-IRQ
        TCCR0 |= (1<<CS00)|(1<<CS02);
        TIMSK |= (1<<TOIE0);

        //UART init: 8N1, nur TX
        UCSRB |= (1<<TXEN);                             // TX einschalten
        UCSRC = (1<<URSEL)|(1<<UCSZ1)|(1<<UCSZ0);       // 8N1

        //Baudratenregister schreiben
        UBRRH = UBRR_VAL >> 8;
        UBRRL = UBRR_VAL & 0xFF;

        //IRQs global aktivieren
        sei();

        while(1);
        return 0;
}
```

```c
//Baudratenregister, 8MHz, 19200
#define UBRR_VAL 25

//UART Receive IRQ
ISR(USART_RXC_vect)
{
	//Empfangenes Byte auf Leds ausgeben...
	PORTA = UDR;
}


//FUNCTION "main"
int main (void)
{

	//PORTA als Output
	DDRA = 0xff;

	//UART init: 8N1, nur TX
	UCSRB |= (1<<RXEN)|(1<<RXCIE);                  // RX einschalten und RX-IRQ aktivieren
	UCSRC = (1<<URSEL)|(1<<UCSZ1)|(1<<UCSZ0);       // 8N1

	//Baudratenregister schreiben
	UBRRH = UBRR_VAL >> 8;
	UBRRL = UBRR_VAL & 0xFF;

	//IRQs global aktivieren
	sei();


	while(1);
	return 0;
}
```

```c
//Baudratenregister, 8MHz, 19200
#define UBRR_VAL 25

//UART Receive IRQ
ISR(USART_RXC_vect)
{
	//Empfangenes Byte auf Leds ausgeben...
	PORTA = UDR;
}


//FUNCTION "main"
int main (void)
{

	//PORTA als Output
	DDRA = 0xff;

	//UART init: 8N1, nur TX
	UCSRB |= (1<<RXEN)|(1<<RXCIE);                  // RX einschalten und RX-IRQ aktivieren
	UCSRC = (1<<URSEL)|(1<<UCSZ1)|(1<<UCSZ0);       // 8N1

	//Baudratenregister schreiben
	UBRRH = UBRR_VAL >> 8;
	UBRRL = UBRR_VAL & 0xFF;

	//IRQs global aktivieren
	sei();


	while(1);
	return 0;
}
```

Demonstration