```
// ====================================================================
// Include C Libriaries
// ====================================================================

#include <avr/io.h>
#include <avr/delay.h>
#include <avr/interrupt.h>

// ====================================================================
// Portdefinitionen
// ====================================================================

#define IR_OUT_PORT PORTB
#define IR_OUT_PIN PB0

// ====================================================================
// Timerdefinitionen
// ====================================================================

// Phasenkorrekter PWM / Vorteiler 1 / Compare-Wert = 50
// Berechnung: 4 MHz / (2*50) = 40 kHz (40.000 Interrupts und LED-Toogle pro Sekunde)
#define timer_config TCCR0A=(1<<WGM00);TCCR0B=(1<<WGM02)|(1<<CS00);OCR0A=55

// mit Kopplung IR LED (Toogle PB0)
#define timer_start_Pulse TCCR0A|=(1<<COM0A0);TCNT0=0

// ohne Kopplung IR LED (Toogle PB0)
#define timer_start_Pause TCCR0A&=~(1<<COM0A0);TCNT0=0

// Timer anhalten
#define timer_stop TCCR0A=0;TCCR0B=0;OCR0A=0

// ====================================================================
// NEC Code: Dauer der Puls- und Pausephasen [Anzahl der Interrupts pro Bit/Phase]
// ====================================================================

//#define dauer_count 25                                  // Dauer zw. 2 Interrupts = Frequenz LED
//// 40 kHz => 1/40.000 = 25 us
//#define startbit_pulse (9000/dauer_count)                                       // Startbit Pulse = 360
//#define startbit_pause (4500/dauer_count)                                       // Startbit Pause = 180
//#define datenbit_pulse (560/dauer_count)                                        // Datenbit Pulse = 22
//#define datenbit0_pause (560/dauer_count)                                       // 0-Bit Pause = 22
//#define datenbit1_pause (1690/dauer_count)                                      // 1-Bit Pause = 67
//#define stopbit_pulse (560/dauer_count)                                                 // Stopbit Pulse
= 22
//#define stopbit_pause (9000/dauer_count)                                        // Stopbit Pause = 360
(mindestens)

// TEST >>>
#define dauer_count 1                                   // Dauer zw. 2 Interrupts = Frequenz LED

#define startbit_pulse (50000/dauer_count)                                      // Startbit Pulse = 360
#define startbit_pause (50000/dauer_count)                                      // Startbit Pause = 180
#define datenbit_pulse (10000/dauer_count)                                      // Datenbit Pulse = 22
#define datenbit0_pause (20000/dauer_count)                                     // 0-Bit Pause = 22
#define datenbit1_pause (10000/dauer_count)                                     // 1-Bit Pause = 67
#define stopbit_pulse (500/dauer_count)                                         // Stopbit Pulse = 22
#define stopbit_pause (500/dauer_count)                                         // Stopbit Pause = 360
(mindestens)
// << TEST

// Zaehler fuer Anzahl Interrupts (= Anzahl LED-Impulse)
volatile long int interrupt_count;

// ====================================================================
// Hauptprogramm
// ====================================================================
```

```
void main(void)
{
        // 4 Bytes der Sequenz
        unsigned char byte1;
        unsigned char byte2;
        unsigned char byte3;
        unsigned char byte4;

        // TEST
        unsigned char i;

        // PB0 als Ausgang (Toogle IR LED)
        DDRB = (1<<IR_OUT_PIN);

        // Konfiguration Timer
        // Phasenkorrekter PWM / Vorteiler 1 / Compare-Wert = 50
        timer_config;

        // Interrupt definieren
        TIMSK0 = (1<<OCIE0A);
        sei();

        // kurz warten
        _delay_ms(2000);

        // Beispiel-Sequenz senden
        byte1 = 0b11111111;
        byte2 = 0b00000000;
        byte3 = 0b10101010;
        byte4 = 0b11001100;

        sendStartBit();

        sendDatenByte(byte1);
        sendDatenByte(byte2);
        sendDatenByte(byte3);
        sendDatenByte(byte4);

        sendStopBit();

        TIMSK0 = 0;
        timer_stop;
        IR_OUT_PORT = (1<<IR_OUT_PIN);
}

// ====================================================================
// Funktionen zum Senden einzelner Bits
// ====================================================================

void sendStartBit()
{
        sendBit(startbit_pulse,startbit_pause);
}

void sendStopBit()
{
        sendBit(stopbit_pulse,stopbit_pause);
}

void sendBit(long int countPulse,long int countPause)
{
        interrupt_count = 0;
        timer_start_Pulse;
        while (interrupt_count<countPulse);

        interrupt_count = 0;
        timer_start_Pause;
        while (interrupt_count<countPause);
```

```
}

void sendDatenBit(unsigned char bit_0_1)
{
        if (bit_0_1 == 0) {
        sendBit(datenbit_pulse,datenbit0_pause);}
        else {
        sendBit(datenbit_pulse,datenbit1_pause);}
}

void sendDatenByte(unsigned datenByte)
{
        // Bit-Zaehler
        unsigned char bit_count;
        unsigned char current_bit;

        // Schleife ueber 8 Bits des Datenbyte
        for (bit_count=8;bit_count>=1;bit_count--)
        {
                current_bit = (datenByte & (1 << (bit_count-1))) >> (bit_count-1);
                sendDatenBit(current_bit);
        }
}

// ====================================================================
// Interrupt Service Routine
// ====================================================================

ISR(TIM0_COMPA_vect)
{
        interrupt_count++;                              // Interrupts (= IR Pulse) zaehlen
}
```