Aufgabe 1: Variabel schnelles AVR-Lauflicht (ca. 30 Minuten)

Schreiben Sie ein AVR-Lauflicht, das in einer Reihe von 3 Leuchtdioden jeweils eine Diode für eine bestimmte Zeitspanne leuchten lässt. Nach Ablauf der Zeitspanne wird die Leuchtdiode deaktiviert und die nachfolgende Leuchtdiode für die Zeitspanne aktiviert. Am Ende der Reihe wird wieder mit der ersten Diode angefangen. Die Zeitintervalle zwischen den Schaltvorgängen sollen über zwei Taster im Intervall 100ms bis 5s in 100ms-Schritten erhöht bzw. verringert werden können, zu Beginn beträgt das Intervall 1s. Die Zeitspannen sollen mit einem 8-bit Timer mit Compare-Match-Einheit realisiert werden, wobei der Mikrokontroller während der Wartezeiten einen Sleep-Modus betreten soll. Änderungen des Schaltintervalls sollen sofort aktiv werden; wird ein Zeitintervall während einer Wartephase verringert und liegt das neue Intervall unter der bereits verstrichenen Zeit, so soll die Wartephase sofort beendet werden.

Schreiben Sie zunächst eine Funktion `void init();`, welche die benötigten IO-Ports, den Timer und die Interruptquellen konfiguriert. Diese wird zum Programmstart von der `main()`-Funktion einmalig aufgerufen. Danach durchläuft die `main()`-Funktion in einer Endlosschleife die Leuchtsequenz, wobei zur Realisierung der Wartezeit eine ebenfalls zu implementierende Funktion `void wait();` aufgerufen wird, die für das aktuell gültige Warteintervall den Prozessor in einem Stromsparmodus setzt. Implementieren Sie zudem die Unterbrechungsbehandlungsfunktionen für die beiden externen Interrupts zur Erhöhung bzw. Verringerung des Schaltintervalls und die Unterbrechungsbehandlungsfunktion für den Timer-Interrupt. Verwenden Sie einen Softwarezähler um Zeitspannen zu realisieren, die mit der gegebenen Hardwarekonfiguration nicht zu erreichen sind.

Information über die Hardware

Taster *Intervall erhöhen*: **Port D, Pin 2,** externe Interruptquelle **INT0**, libc-Makro **INT0_vect**
Taster *Intervall verkürzen*: **Port D, Pin 3,** externe Interruptquelle **INT1**, libc-Makro **INT1_vect**
Die Aktivierung der entsprechenden Interruptquellen erfolgt durch Setzen des **INT0** bzw. **INT1**-Bits im Register **GICR**.
Beide Taster verbinden den Pin mit Masse, es muss der interne Pullup-Widerstand verwendet werden.

ISC11/ISC01	ISC10/ISC00	Beschreibung
0	0	Interrupt bei low Pegel
0	1	Interrupt bei beliebiger Flanke
1	0	Interrupt bei fallender Flanke
1	1	Interrupt bei steigender Flanke

Table 1: Konfiguration der externen Interruptquellen 1/0 (Bits in Register MCUCR)

3 Leuchtdioden: **Port D, Pin 5, Pin 6 und Pin 7**, Start des Lauflichts mit der Leuchtdiode an Pin 5.
8-bit Timer: Kein Prescaler (**Takt 125 kHz**), Timer-Register **TCNT2**, Output-Compare-Register **OCR2**, Kontrollregister **TCCR2,** Interrupt demaskieren durch Setzen des **OCIE2**-Bit in Register **TIMSK**, avr-libc Makro **TIMER2_COMP_vect**.

WGM21	WGM20	
0	0	Normaler Modus
1	0	Clear-Timer on Compare Match Modus (CTC)

Table 2: Timer Moduskonfiguration (Bits in Register TCCR2)

Ergänzen Sie das folgende Codegerüst so, dass ein vollständig übersetzbares Programm entsteht.

```
#include <avr/io.h>
#include <avr/interrupt.h>
#include <avr/sleep.h>

/* Makros, Funktiondeklarationen, etc. */

----------
----------
----------
----------
----------
----------
----------
----------
----------

/* Behandlungsfunktion Externer Interrupt 0 */

----------
----------
----------
----------

/* Behandlungsfunktion Externer Interrupt 1 */

----------
----------
----------
----------

/* Behandlungsfunktion Timer Interrupt */

----------
----------
----------
----------
```

```
/* Funktion main() */

  /* Lauflicht-Schleife */

} /* Ende der Funktion main() */

/* Funktion wait() */

} /* Ende der Funktion wait() */
```

```
/* Initialisierungsfunktion init() */

----------------------------------------------------------------------------
----------------------------------------------------------------------------
  /* LEDs konfigurieren */

----------------------------------------------------------------------------
----------------------------------------------------------------------------
----------------------------------------------------------------------------
----------------------------------------------------------------------------
  /* Taster inkl. Interruptquellen konfigurieren */

----------------------------------------------------------------------------
----------------------------------------------------------------------------
----------------------------------------------------------------------------
----------------------------------------------------------------------------
----------------------------------------------------------------------------
----------------------------------------------------------------------------
----------------------------------------------------------------------------
----------------------------------------------------------------------------
----------------------------------------------------------------------------
----------------------------------------------------------------------------
----------------------------------------------------------------------------
  /* Timer inkl. Interruptquelle konfigurieren */

----------------------------------------------------------------------------
----------------------------------------------------------------------------
----------------------------------------------------------------------------
----------------------------------------------------------------------------
----------------------------------------------------------------------------
----------------------------------------------------------------------------
----------------------------------------------------------------------------
----------------------------------------------------------------------------
----------------------------------------------------------------------------
----------------------------------------------------------------------------
----------------------------------------------------------------------------
} /* Ende der Funktion init() */
```