# Lab 4: Tomáš Křička

## My repository

My git - Tomáš Křička, 223283

## Overflow times

1. Complete table with overflow times.

| Module | Number of bits | 1 | 8 | 32 | 64 | 128 | 256 | 1024 |
|---|---|---|---|---|---|---|---|---|
| Timer/Counter0 | 8 | 16u | 128u | -- | 1.024m | -- | 4.096m | 16.38m |
| Timer/Counter1 | 16 | 4.096m | 32.768m | -- | 262.144m | -- | 1.0486 | 4.1943 |
| Timer/Counter2 | 8 | 16u | 128u | 512u | 1.024m | 2.048m | 4.096m | 16.38m |

## Timer library

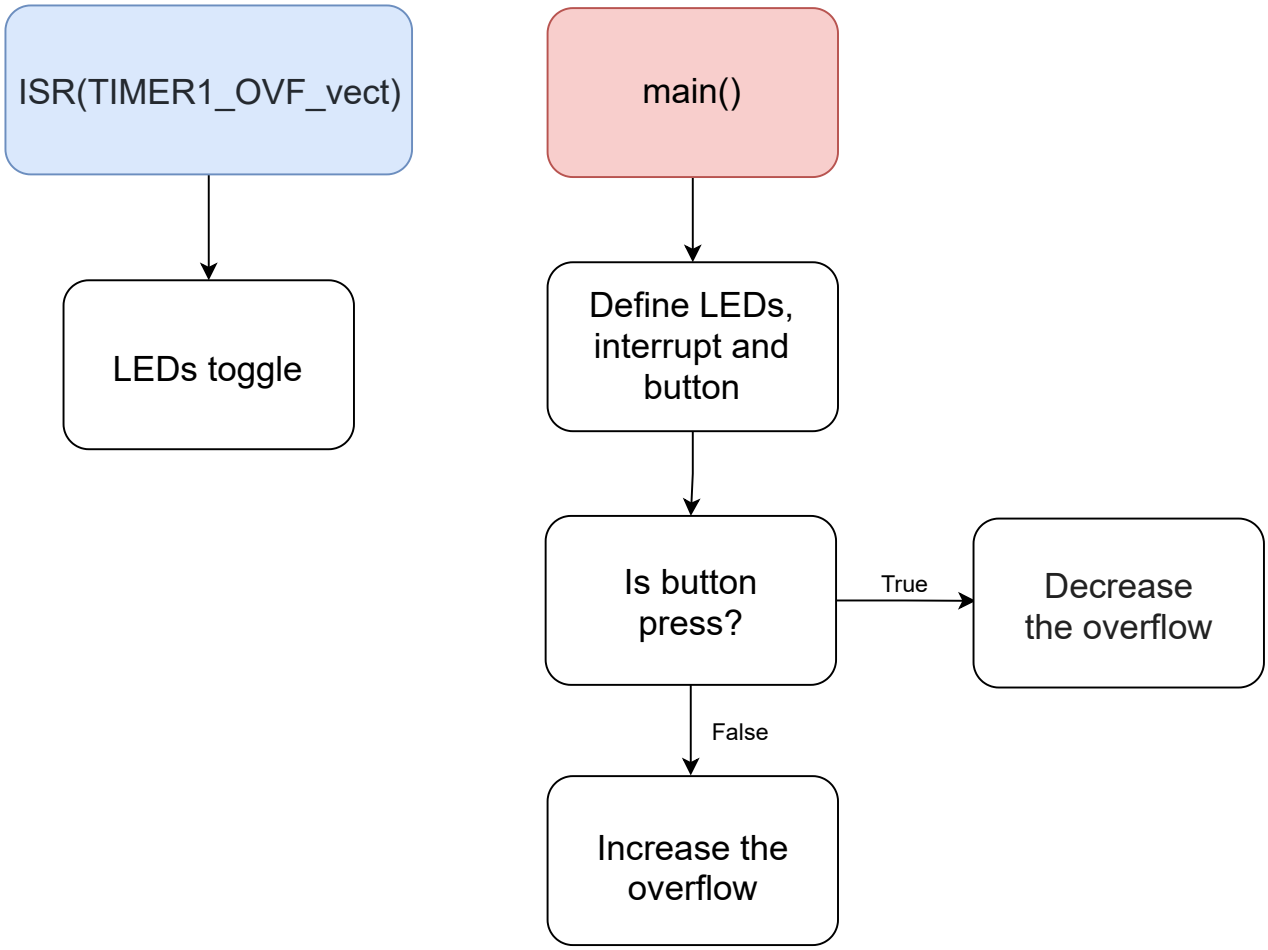
1. In your words, describe the difference between common C function and interrupt service routine.
    - Function - blok of statements thats take inputs and do specific task
    - Interrupt service routine - if came an interrurt from periphery the program started do another function

2. Part of the header file listing with syntax highlighting, which defines settings for Timer/Counter0:

```
/**
 * @name  Definitions of Timer/Counter0
 * @note  F_CPU = 16 MHz
 */
/** @brief Stop timer, prescaler 000 --> STOP */
#define TIM0_stop()             TCCR0B &= ~((1<<CS02) | (1<<CS01) | (1<<CS10));
/** @brief Set overflow 16us, prescaler 001 --> 1 */
#define TIM0_overflow_16us()    TCCR0B &= ~((1<<CS02) | (1<<CS01)); TCCR0B |=
(1<<CS00);
/** @brief Set overflow 128us, prescaler 010 --> 8 */
#define TIM0_overflow_128us()   TCCR0B &= ~((1<<CS02) | (1<<CS00)); TCCR0B |=
(1<<CS01);
/** @brief Set overflow 1ms, prescaler 011 --> 64 */
#define TIM0_overflow_1ms()     TCCR0B &= ~(1<<CS02); TCCR0B |= (1<<CS01) |
(1<<CS00);
/** @brief Set overflow 4ms, prescaler 100 --> 256 */
#define TIM0_overflow_4ms()     TCCR0B &= ~((1<<CS01) | (1<<CS00)); TCCR0B |=
(1<<CS02);
```

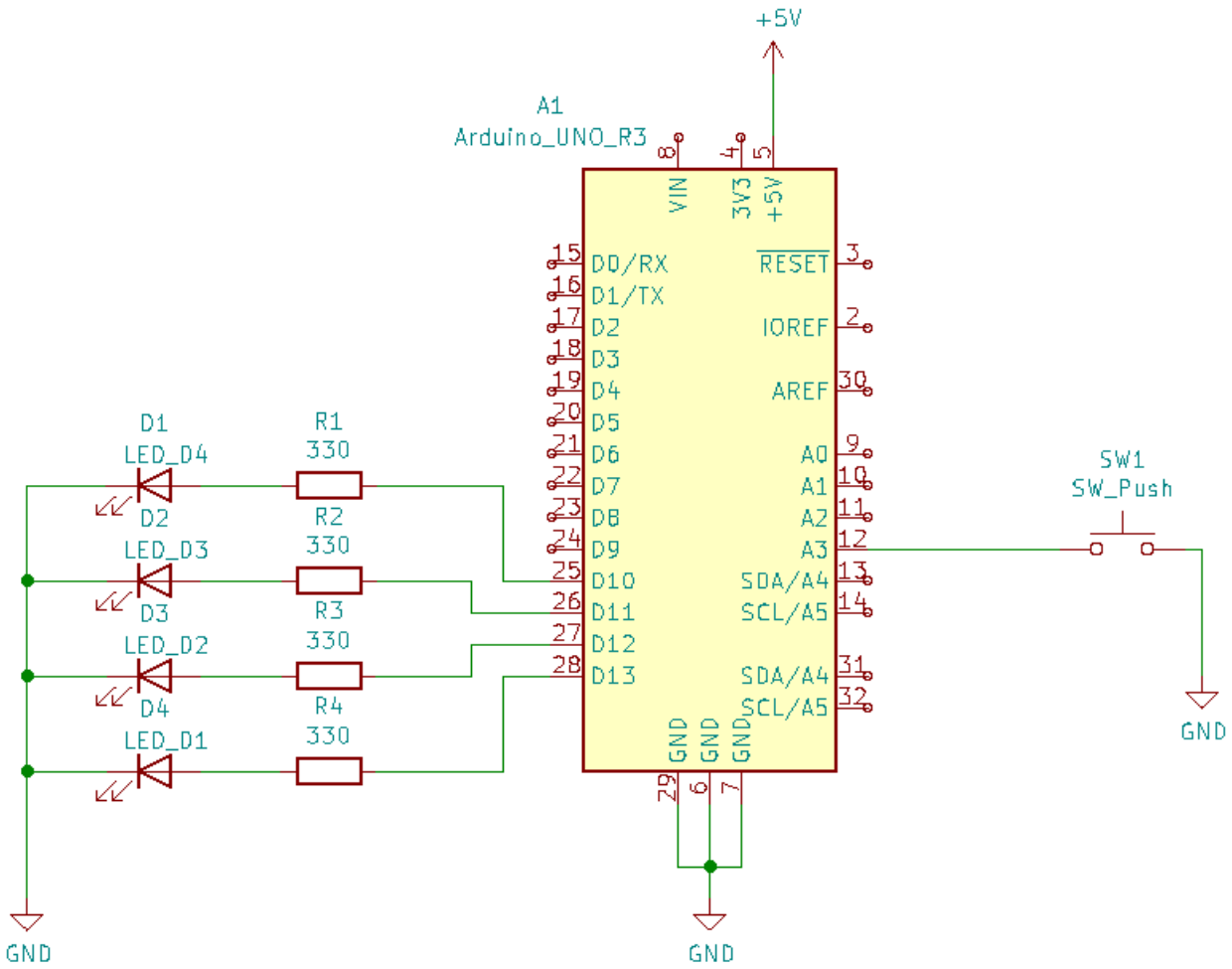
```
/** @brief Set overflow 16ms, prescaler // 101 --> 1024 */
#define TIM0_overflow_16s()     TCCR0B &= ~(1<<CS01); TCCR0B |= (1<<CS00) |
(1<<CS02);
/** @brief Enable overflow interrupt, 1 --> enable */
#define TIM0_overflow_interrupt_enable()  TIMSK0 |= (1<<TOIE0);
/** @brief Disable overflow interrupt, 0 --> disable */
#define TIM0_overflow_interrupt_disable() TIMSK0 &= ~(1<<TOIE0);
```

3. Flowchart figure for function `main()` and interrupt service routine `ISR(TIMER1_OVF_vect)` of application that ensures the flashing of one LED in the timer interruption. When the button is pressed, the blinking is faster, when the button is released, it is slower. Use only a timer overflow and not a delay library.




## Knight Rider

1. Scheme of Knight Rider application with four LEDs and a push button, connected according to Multi-function shield. Connect AVR device, LEDs, resistors, push button, and supply voltage. The image can be drawn on a computer or by hand. Always name all components and their values!




Tabulka (datasheet page 116)

| Module | Operation | I/O register(s) | Bit(s) |
|---|---|---|---|
| Timer/Counter0 | Prescaler<br>8-bit data value<br>Overflow interrupt enable | | CS02, CS01, CS00 |
| Timer/Counter1 | Prescaler<br>16-bit data value<br>Overflow interrupt enable | TCCR1B<br>TCNT1H, TCNT1L<br>TIMSK1 | CS12, CS11, CS10 (000: stopped, 001: 1, 010: 8, 011: 64, 100: 256, 101: 1024)<br>TCNT1[15:0]<br>TOIE1 (1: enable, 0: disable) |
| Timer/Counter2 | Prescaler<br>8-bit data value<br>Overflow interrupt enable | | CS22, CS21, CS20 |

Tabulka 2 (datasheet page 74)

| Program address | Source | Vector name | Description |
|---|---|---|---|
| 0x0000 | RESET | -- | Reset of the system |
| 0x0002 | INT0 | `INT0_vect` | External interrupt request number 0 |
| 0x0004 | INT1 | `INT1_vect` | External interrupt request number 1 |
| 0x0006 | PCINT0 | | Pin Change Interrupt Request 0 |
| 0x0008 | PCINT1 | | Pin Change Interrupt Request 1 |
| 0x00A | PCINT2 | | Pin Change Interrupt Request 2 |
| 0x00C | WDT | | Watchdog Time-out Interrupt |
| 0x0012 | TIMER2_OVF | | Timer/Counter2 Overflow |
| 0x0018 | TIMER1_COMPB | `TIMER1_COMPB_vect` | Compare match between Timer/Counter1 value and channel B compare value |
| 0x001A | TIMER1_OVF | `TIMER1_OVF_vect` | Overflow of Timer/Counter1 value |
| 0x0020 | TIMER0_OVF | | Timer/Counter0 Overflow |
| 0x0024 | USART_RX | | USART Rx Complete |
| 0x002A | ADC | | ADC Conversion Complete |
| 0x0030 | TWI | | 2-wire Serial Interface |