Lab 4: DANIEL HAVRÁNEK

Link to your `Digital-electronics-2` GitHub repository:

https://github.com/Dan5049/Digital-electronic-2

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	16,384m
Timer/Counter1	16	4,096m	32,768m	-	262,144m	--	1,048576	4,194304
Timer/Counter2	8	16u	128u	512u	1,024m	2,048m	4,096m	16,384m

Timer library

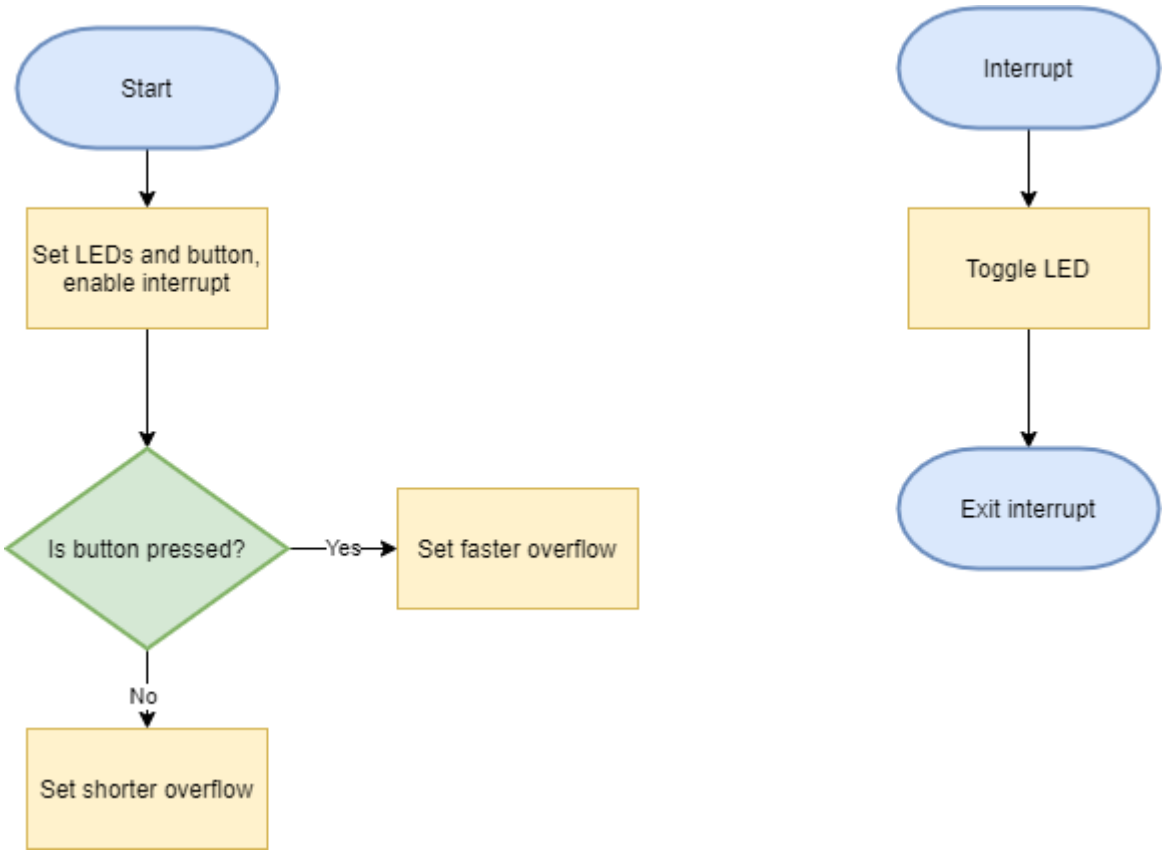
1. In your words, describe the difference between common C function and interrupt service routine.
    - Function - a group of statements that together perform a task
    - Interrupt service routine - when interrupt occurs, the microcontroller runs the ISR, its similar to functinon, but they are called by occurrence of an interrupt and terminate with a return from interrupt instruction

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

```
/**
 * @name  Definitions for 8-bit Timer/Counter0
 * @note  t_OVF = 1/F_CPU * prescaler * 2^n where n = 8, F_CPU = 16 MHz
 */
/** @brief Stop timer, prescaler 000 --> STOP */
#define TIM0_stop()              TCCR0B &= ~((1<<CS02) | (1<<CS01) | (1<<CS00));
/** @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_16ms()     TCCR0B &= ~(1<<CS01); TCCR0B |= (1<<CS02) |
```

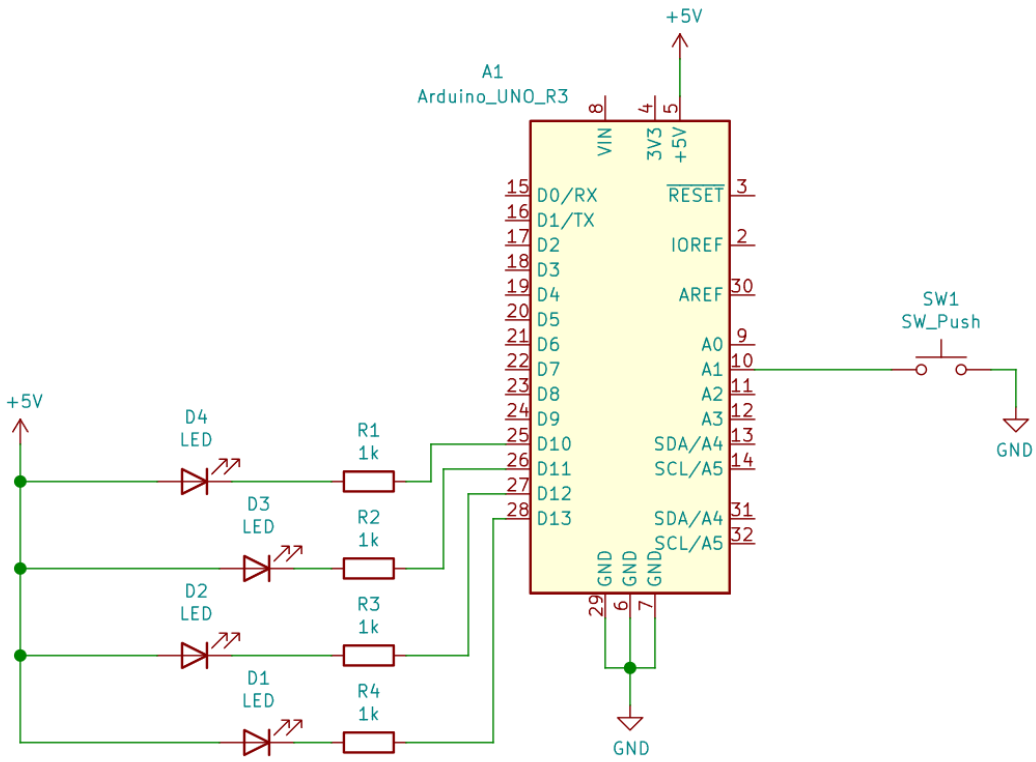
```
(1<<CS00);
/** @brief Enable overflow interrupt, 1 --> enable */
#define TIM0_overflow_interrupt_enable()    TIMSK0 |= (1<<TOIE0);
/** @brief Disable overflow interrupt, 0 --> disable */
#define TIM0_overflow_interrupt_disable()   TIMSK0 &= ~(1<<TOIE0);
```

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!


+5V
A1
Arduino_UNO_R3
8
4
5
VIN
3V3
+5V
15
D0/RX
16
D1/TX
17
D2
18
D3
19
D4
20
D5
21
D6
22
D7
23
D8
24
D9
25
D10
26
D11
27
D12
28
D13
RESET
3
IOREF
2
AREF
30
A0
9
A1
10
A2
11
A3
12
SDA/A4
13
SCL/A5
14
SDA/A4
31
SCL/A5
32
GND
GND
GND
29
6
7
GND
SW1
SW_Push
GND
+5V
D4
LED
R1
1k
D3
LED
R2
1k
D2
LED
R3
1k
D1
LED
R4
1k