Lab 4: Ondřej Soukeník

https://github.com/ondrasouk/Digital-electronics-2/blob/main/Labs/04-interrupts/README.md

Overflow times

1. Complete table with overflow times.

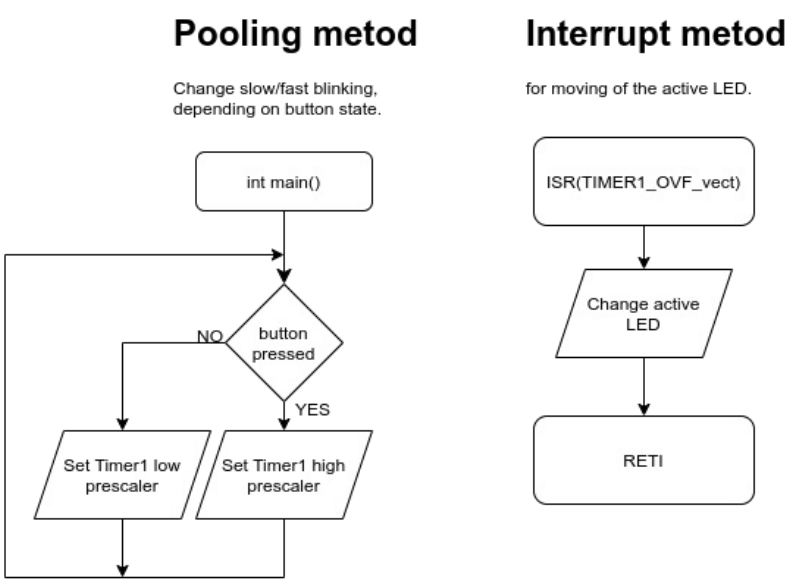
Module	Number of bits	1	8	32	64	128	256	1024
Timer/Counter0	8	16u	128u	--	1024u	--	4096u	16384u
Timer/Counter1	16	4096u	32768u	--	262144u	--	1048576u	4194304u
Timer/Counter2	8	16u	128u	512u	1024u	2048u	4096u	16384u

Timer library

1. In your words, describe the difference between common C function and interrupt service routine.
   - Function
     - Program calls function.
   - Interrupt service routine
     - CPU calls Interrupt service routine. Running program is paused (the Program counter is stored) when the interrupt occurs.
2. Part of the header file listing with syntax highlighting, which defines settings for Timer/Counter0:

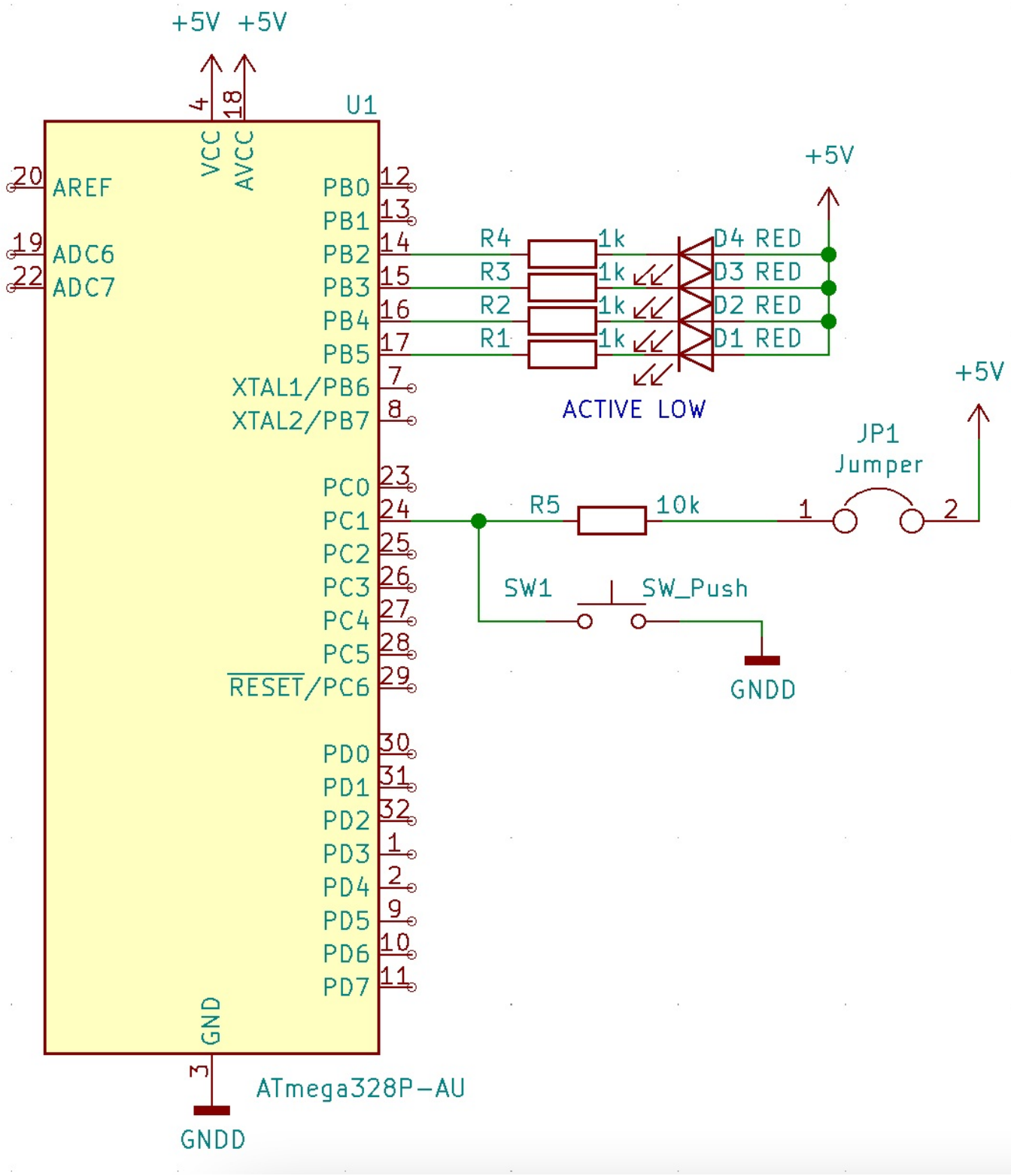
```
/**
 * @name  Definitions for 8-bit Timer/Counter0
 * @note  t_OVF = 1/F_CPU * prescaler * 2^n where n = 8, F_CPU = 16 MHz
 */
/** @brief Stop timer, prescaler 000 --> STOP */
#define TIM0_stop()           TCCR0B &= ~((1<<CS12) | (1<<CS11) | (1<<CS10));
/** @brief Set overflow 16us, prescaler 001 --> 1 */
#define TIM0_overflow_16us()   TCCR0B &= ~((1<<CS12) | (1<<CS11)); TCCR0B |= (1<<CS10);
/** @brief Set overflow 128us, prescaler 010 --> 8 */
#define TIM0_overflow_128us()  TCCR0B &= ~((1<<CS12) | (1<<CS10)); TCCR0B |= (1<<CS11);
/** @brief Set overflow 1ms, prescaler 011 --> 64 */
#define TIM0_overflow_1ms() TCCR0B &= ~(1<<CS12); TCCR0B |= (1<<CS11) | (1<<CS10);
/** @brief Set overflow 4ms, prescaler 100 --> 256 */
#define TIM0_overflow_4ms()    TCCR0B &= ~((1<<CS11) | (1<<CS10)); TCCR0B |= (1<<CS12);
/** @brief Set overflow 16.3ms, prescaler // 101 --> 1024 */
#define TIM0_overflow_16ms()    TCCR0B &= ~(1<<CS11); TCCR0B |= (1<<CS12) | (1<<CS10);
/** @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
+5V
4
18
U1
VCC
AVCC
20
AREF
19
ADC6
22
ADC7
PB0
PB1
PB2
PB3
PB4
PB5
12
13
14
15
16
17
XTAL1/PB6
XTAL2/PB7
7
8
PC0
PC1
PC2
PC3
PC4
PC5
RESET/PC6
23
24
25
26
27
28
29
PD0
PD1
PD2
PD3
PD4
PD5
PD6
PD7
30
31
32
1
2
9
10
11
GND
3
GNDD
ATmega328P-AU
+5V
R4
R3
R2
R1
1k
D4 RED
D3 RED
D2 RED
D1 RED
ACTIVE LOW
+5V
JP1
Jumper
R5
10k
1
2
SW1
SW_Push
GNDD