xbarto0c / Digital-electronics-2 Public

Code Issues Pull requests Actions Projects Wiki Security Insights

main

Digital-electronics-2 / Labs / 04-interrupts / Assignment.md

xbarto0c Update Assignment.md History

1 contributor

56 lines (37 sloc) 2.79 KB

Lab 4: Jan Bartoň

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

ttps://github.com/xbarto0c/Digital-electronics-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	--	1024u	--	4096u	16384u
Timer/Counter1	16	4096u	32768u	--	262144u	--	1048576u	4194304u
Timer/Counter2	8	16u	128u	512u	1024u	2048u	4096u	16384u

Timer library

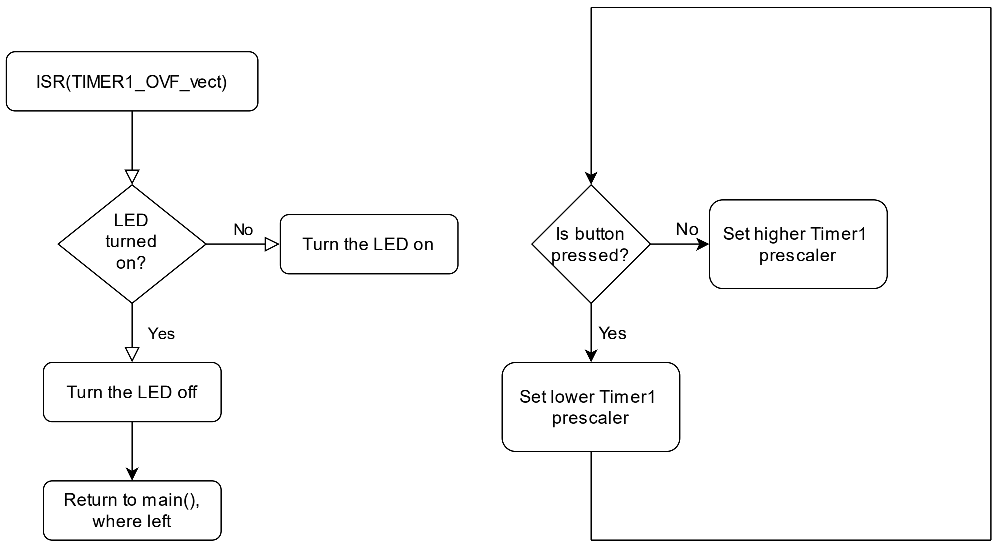
1. In your words, describe the difference between common C function and interrupt service routine.
   - Function - gets executed, whenever we call it from inside the running program, can do complex operations, unlike ISR
   - Interrupt service routine - gets executed, whenever a defined hardware event occurs (i.e. counter overflow), should be kept as small as possible
2. Part of the header file listing with syntax highlighting, which defines settings for Timer/Counter0:

```c
/**
 * @name  Definitions of Timer/Counter0
 * @note  F_CPU = 16 MHz
 */
#define TIM0_stop()           TCCR0B &= ~((1<<CS02) | (1<<CS01) | (1<<CS00));        // 000 --> STOP
#define TIM0_overflow_4ms()   TCCR0B &= ~((1<<CS02) | (1<<CS01)); TCCR0B |= (1<<CS00);// 001 --> 1
#define TIM0_overflow_33ms()  TCCR0B &= ~((1<<CS02) | (1<<CS00)); TCCR0B |= (1<<CS01);// 010 --> 8
#define TIM0_overflow_262ms() TCCR0B &= ~(1<<CS02); TCCR0B |= (1<<CS01) | (1<<CS00);  // 011 --> 64
#define TIM0_overflow_1s()    TCCR0B &= ~((1<<CS01) | (1<<CS00)); TCCR0B |= (1<<CS02);// 100 --> 256
#define TIM0_overflow_4s()    TCCR0B &= ~(1<<CS01); TCCR0B |= (1<<CS02) | (1<<CS00);  // 101 --> 1024

/**
 * @brief Defines interrupt enable/disable modes for Timer/Counter0.
 */
#define TIM0_overflow_interrupt_enable()    TIMSK0 |= (1<<TOIE0);   // 1 --> enable
```

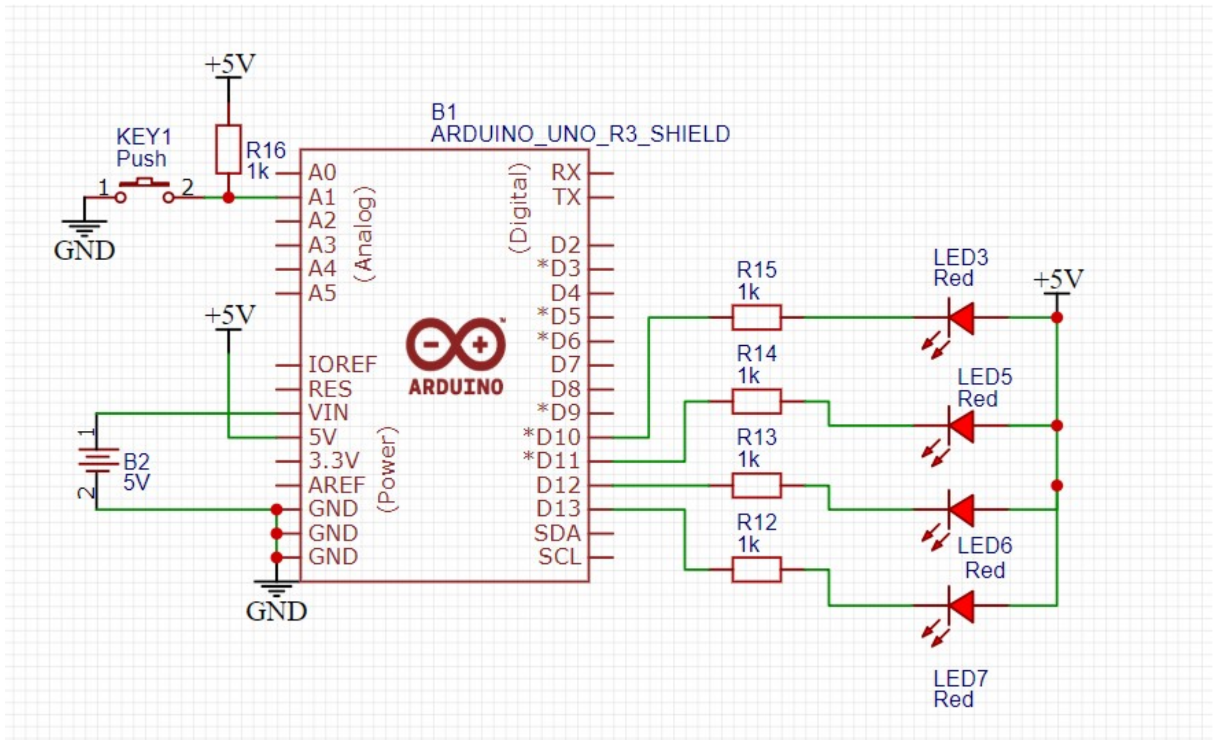
```c
#define TIM0_overflow_interrupt_disable()   TIMSK0 &= ~(1<<TOIE0);  // 0 --> disable
```

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!


B1
ARDUINO_UNO_R3_SHIELD
+5V
KEY1
Push
R16
1k
GND
+5V
B2
5V
GND
R15 1k
R14 1k
R13 1k
R12 1k
LED3 Red
LED5 Red
LED6 Red
LED7 Red
+5V
ARDUINO