Name: MASAUSO LUNGU 209533

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

Masauso Lungu

Lab 4: Interupts, timers

Learning objectives

The purpose of the laboratory exercise is to understand the function of the interrupt, interrupt service routine, and the functionality of timer units. Another goal is to practice finding information in the MCU manual; specifically setting timer control registers.

PART1: Preparation tasks (done before the lab at home)

Consider an n-bit number that we increment based on the clock signal. If we reach its maximum value and try to increase it, the value will be reset. We call this state an **overflow**. The overflow time depends on the frequency of the clock signal, the number of bits, and on the prescaler value:

$$t_{ovf} = \frac{1}{f_{CPU}} \cdot 2^n \cdot N =$$

1. Calculate the overflow times for three Timer/Counter modules that contain ATmega328P if CPU clock frequency is 16 MHz. Complete the following table for given prescaler values. Note that, Timer/Counter2 is able to set 7 prescaler values, including 32 and 128 and other timers have only 5 prescaler values.

Module	Number of bits	1	8	32	64	128	256	1024
Timer/Counter0	8	16us	128us	--	1.024ms	--	4.096ms	16.384ms
Timer/Counter1	16	4.096ms	32.768ms	--	262.144ms	--	1.048576s	4.1943s
Timer/Counter2	8	16us	128us	512us	1.024ms	2.048ms	4.096ms	16.384ms

2. Shields are boards that can be attached to an Arduino board, significantly expand its capabilities, and makes prototyping much faster. See schematic of Multi-function shield and find out the connection of four LEDs (D1, D2, D3, D4) and three push buttons (S1-A1, S2-A2, S3-A3).

The LEDs are connected as follow: D1=PB5[13], D2=PB4[12], D3=PB3[~11], D4=PB2[~10],

And the push buttons are connected as follows: S1-A1=PC1[A1], S2-A2=PC2[A2], S3-A3=PC3[A3].

3. Timers

A timer (or counter) is a hardware block within an MCU and can be used to measure time events. ATmega328P has three timers, called:

- Timer/Counter0,
- Timer/Counter1, and
- Timer/Counter2.

T/C0 and T/C2 are 8-bit timers, where T/C1 is a 16-bit timer. The counter counts in synchronization with microcontroller clock from 0 up to 255 (for 8-bit counter) or 65,535 (for 16-bit). Different clock sources can be selected for each timer using a CPU frequency divider with fixed prescaler values, such as 8, 64, 256, 1024, and others.

The timer modules can be configured with several special purpose registers. According to the ATmega328P datasheet (eg in the **8-bit Timer/Counter0 with PWM > Register Description** section), which I/O registers and which bits configure the timer operations?

Module	Operation	I/O register(s)	Bit(s)
Timer/Counter0	Prescaler	TCCR0B	CS02, CS01, CS00 (000: stopped, 001: 1, 010: 64, 100: 256, 101: 1024)
	8-bit data value	TCNT0	TCNT0[7:0]
	Overflow interrupt enable	TIMSK0	TOIE0(1: enable, 0: disable)
Timer/Counter1	Prescaler	TCCR1B	CS12, CS11, CS10 (000: stopped, 001: 1, 010: 8, 011: 64, 100: 256, 101: 1024)
	16-bit data value	TCNT1H, TCNT1L	TCNT1[15:0]
	Overflow interrupt enable	TIMSK1	TOIE1 (1: enable, 0: disable)
Timer/Counter2	Prescaler	TCCR2B	CS22, CS11, CS10 (000: stopped, 001: 1, 010: 8, 011: 32, 100: 64, 101: 128, 110: 256, 111: 1024)
	8-bit data value	TCNT2	TCNT2[7:0]
	Overflow interrupt enable	TIMSK2	TOIE2(1: enable, 0; disable)

4. Source of interruptions

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 1
0x0006	PCINT0	`PCINT0_vect`	Pin change interrupt request 0
0x0008	PCINT1	`PCINT1_vect`	Pin change interrupt request 1

Program address	Source	Vector name	Description
0x00A	PCINT2	PCINT2_vect	Pin change interrupt request 2
0x00C	WDT	WDT_vect	Watchdaog Time-out interrupt
0x0012	TIMER2_OVF	TIMER2_OVF_vect	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	TIMER0_OVF_vect	Timer/Counter0 Overflow
0x0024	USART_RX	USART_vect	USART_Rx complete
0x002A	ADC	ADC_vect	ADC Conversion Complete
0x0030	TWI	TWI_vect	2-wire Serial interface

5. PWM

Module	Description	MCU pin	Arduino pin
Timer/Counter0	OC0A	PD6	6
	OC0B	PD5	5
Timer/Counter1	OC1A	PB1	9
	OC1B	PB2	10
Timer/Counter2	OC2A	PB3	11
	OC2B	PD3	3

Part 2: Lab Assignment

Timer library

1. In your words, describe the difference between common C function and interrupt service routine.

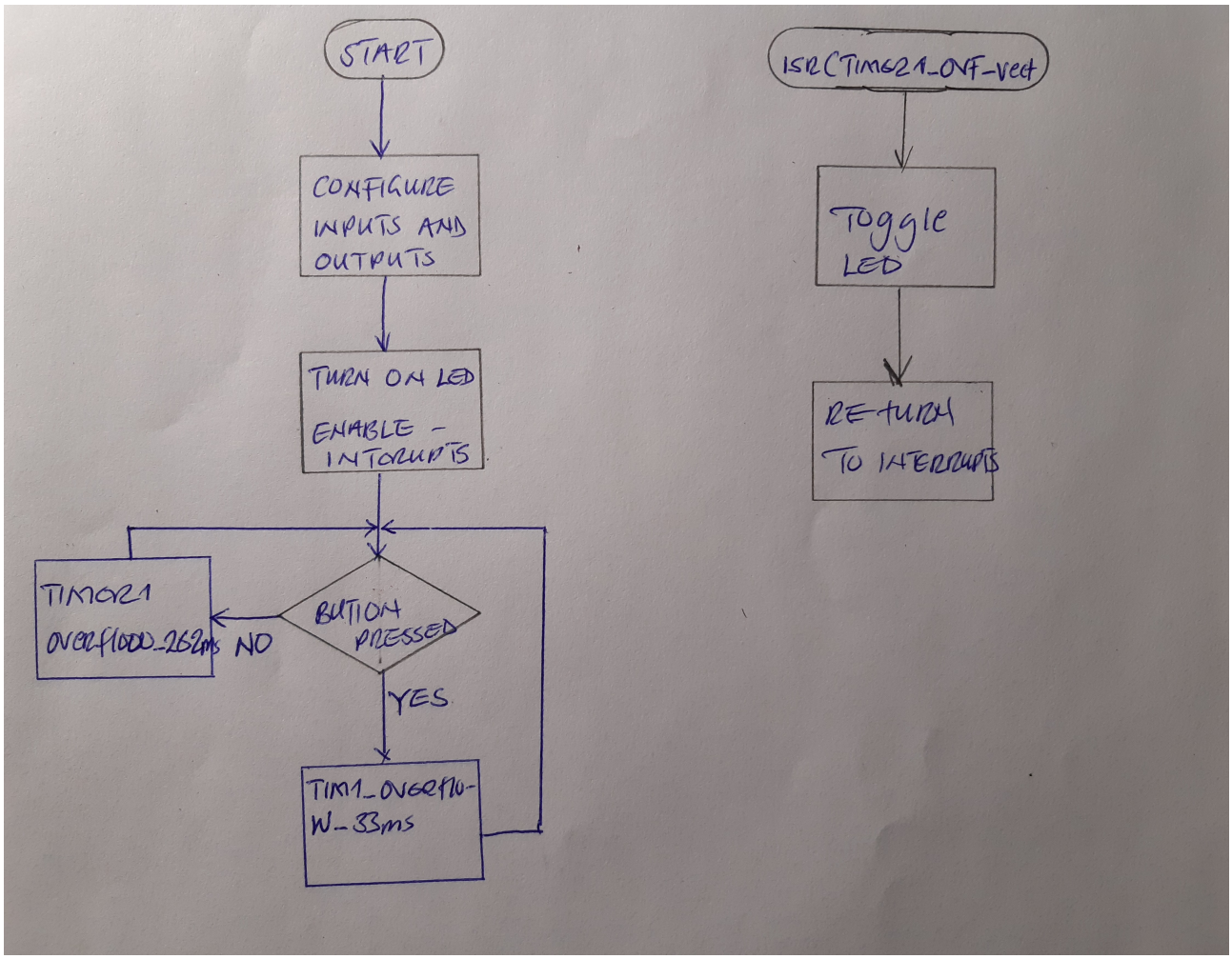
The difference between C function and ISR is in the way they get called.

- Function : The C function is generally called in the main program code and get executed.
- Interrupt service routine : get executed automatically whenever an interrupt is triggered in the hardware peripheral or processor exception signals. ISR takes no parameters and returns no results and unlike the normal function, ISR can be called at anytime when an interrupt occur.

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

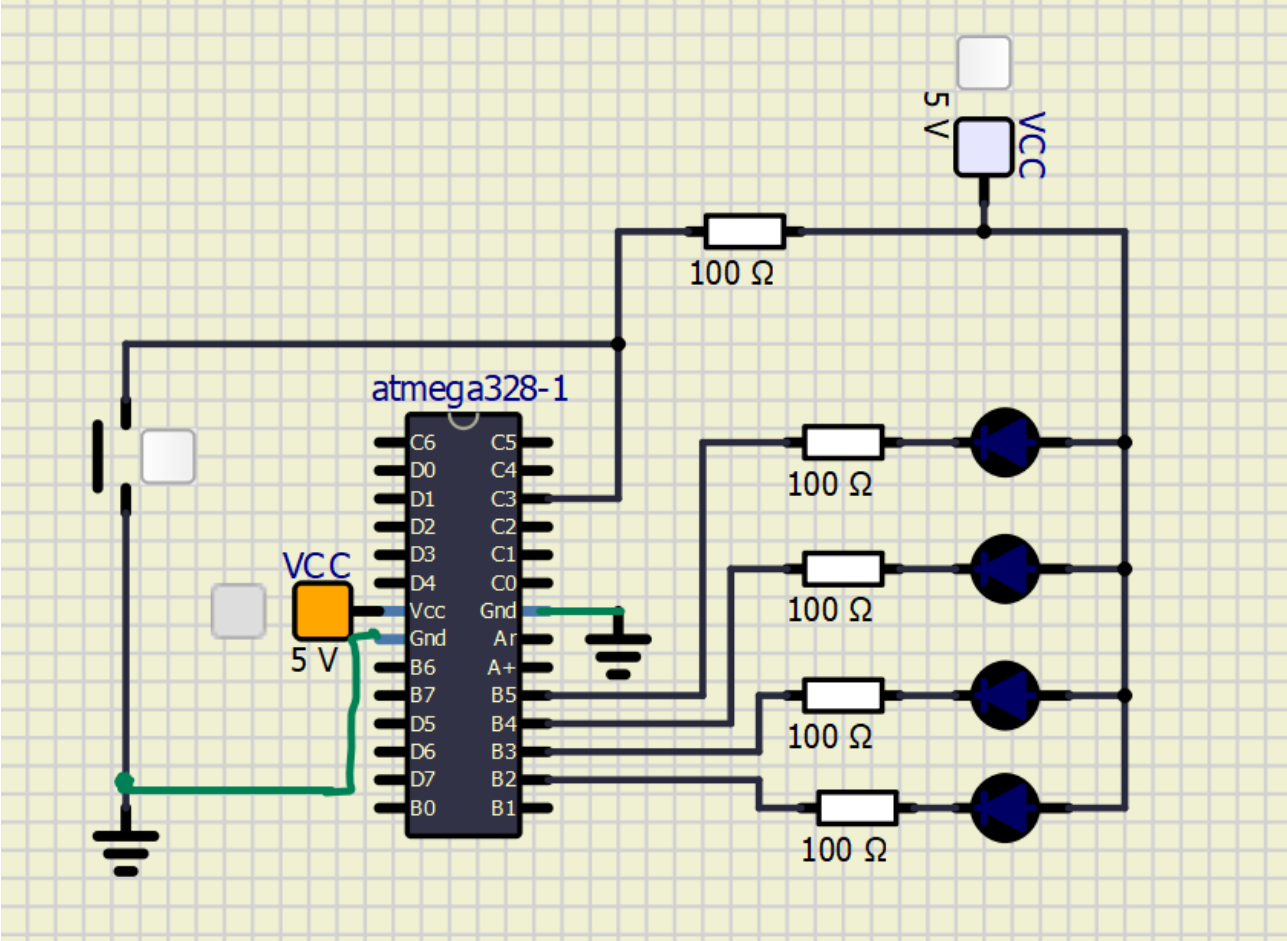
```
#ifndef TIMER_H_
#define TIMER_H_

// library
#include <avr/io.h>

/**
 * @name  Definitions for 8-bit Timer/Counter0
 * @note  t_OVF = 1/F_CPU * prescaler * 2^n where n = 8, F_CPU = 16 MHz
 */
// WRITE YOUR CODE HERE
#define TIM0_stop()              TCCR0B &= ~((1<<CS02) | (1<<CS01) | (1<<CS00));
#define TIM0_overflow_16us()     TCCR0B &= ~((1<<CS02) | (1<<CS01)); TCCR0B |=
(1<<CS00);
#define TIM0_overflow_128us()    TCCR0B &= ~((1<<CS02) | (1<<CS00)); TCCR0B |=
(1<<CS01);
#define TIM0_overflow_1ms()      TCCR0B &= ~(1<<CS02); TCCR0B |= (1<<CS01) |
(1<<CS00);
#define TIM0_overflow_4ms()      TCCR0B &= ~((1<<CS01) | (1<<CS00)); TCCR0B |=
(1<<CS02);
#define TIM0_overflow_16ms()     TCCR1B &= ~(1<<CS01); TCCR0B |= (1<<CS02) |
(1<<CS00);

/**
 * @brief Defines interrupt enable/disable modes for Timer/Counter1.
 */
#define TIM0_overflow_interrupt_enable()    TIMSK0 |= (1<<TOIE0);
#define TIM0_overflow_interrupt_disable()   TIMSK0 &= ~(1<<TOIE0);

#endif
```

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. The image can be drawn on a computer or by hand. Use clear descriptions of the individual steps of the algorithms.



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.


5 V
VCC
100 Ω
atmega328-1
C6
D0
D1
D2
D3
D4
Vcc
Gnd
B6
B7
D5
D6
D7
B0
C5
C4
C3
C2
C1
C0
Gnd
Ar
A+
B5
B4
B3
B2
B1
VCC
5 V
100 Ω
100 Ω
100 Ω
100 Ω