Lab1

Lab results

1. **GitHub**: https://github.com/FilipPaul/Digital-Electronics-2

2. **Blink example**.

   - Meaning of | is logical OR, in blink example is this operator used for writing right value to DDRB register (here you can set if GPIO is output or input).

```
DDRB = DDRB | (1 << LED_GREEN);
```

   - Meaning of & is logical AND, in blink example is this operator used together with ~ (logical NOT) and with <<(operator for shifting bits left) for setting 5th LSB bit in register PORTB to zero without changing other bits in register PORTB.

```
PORTB = PORTB & ~(1 << LED_GREEN);
```

   - Meaning of ^ is logical XOR, in sketch is used for toggling only 1 bit in register

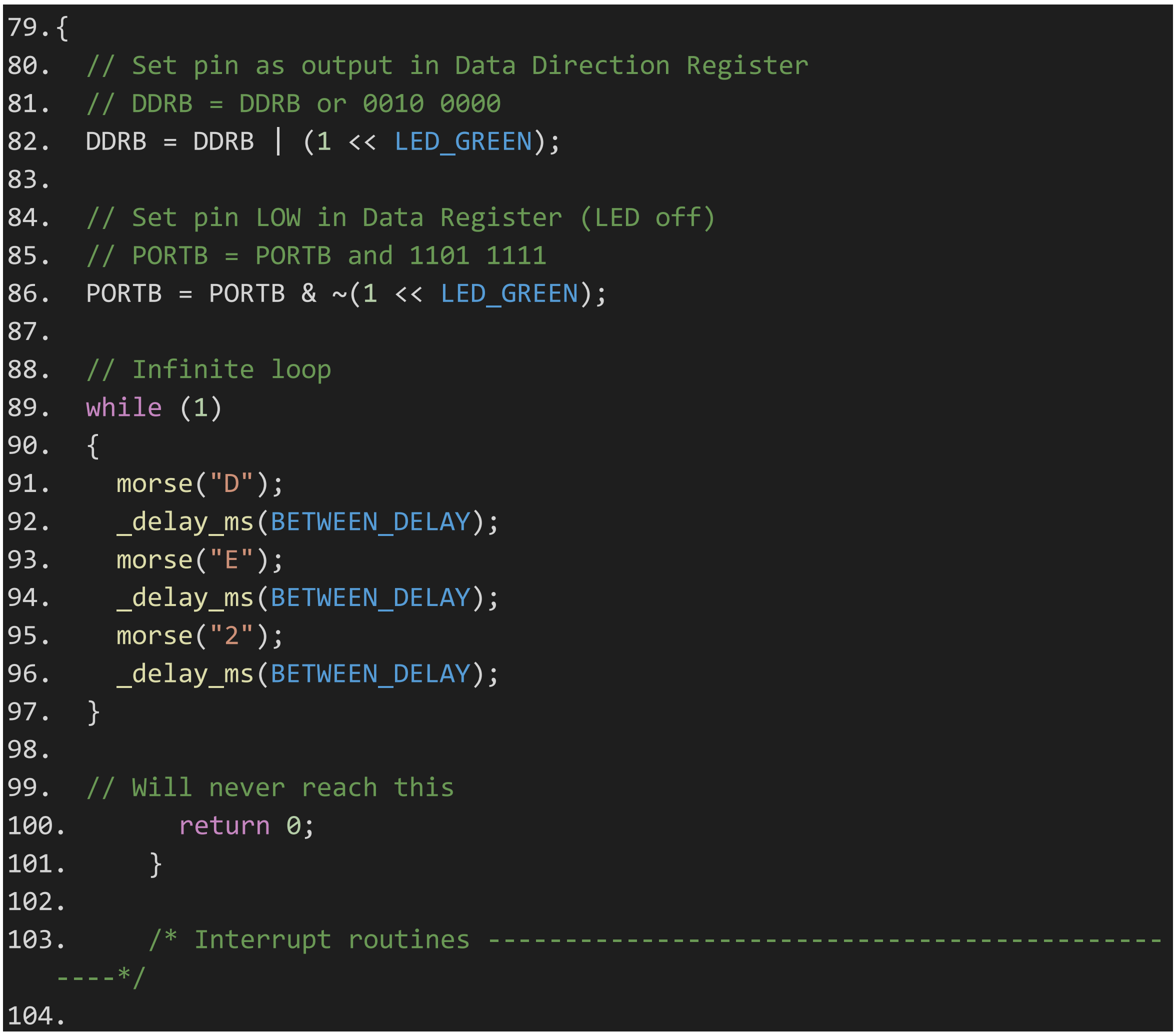
```
PORTB = PORTB ^ (1 << LED_GREEN);
```

6. **Morse code. Submit**:

   - C code (main.c). **also can be found in GITHUB repository with firmware.hex!!**

```
#include <Arduino.h>

/* Defines -----------------------------------------------------------*/
#define LED_GREEN PB5   // AVR pin where green LED is connected
#define SHORT_DELAY 300 // Delay in miliseconds
#define LONG_DELAY 600
#define BETWEEN_DELAY 300
#ifndef F_CPU
#define F_CPU 16000000 // CPU frequency in Hz required for delay func
#endif

/* Includes ----------------------------------------------------------*/
#include <util/delay.h> // Functions for busy-wait delay loops
#include <avr/io.h>     // AVR device-specific IO definitions
void morse(String letter)
{
  if (letter == "D")
  {
    PORTB = PORTB ^ (1 << LED_GREEN);
    _delay_ms(LONG_DELAY);
    PORTB = PORTB ^ (1 << LED_GREEN);
    _delay_ms(LONG_DELAY);
    PORTB = PORTB ^ (1 << LED_GREEN);
```

```c
    _delay_ms(SHORT_DELAY);
    PORTB = PORTB ^ (1 << LED_GREEN);
    _delay_ms(SHORT_DELAY);
    PORTB = PORTB ^ (1 << LED_GREEN);
    _delay_ms(SHORT_DELAY);
    PORTB = PORTB ^ (1 << LED_GREEN);
    _delay_ms(SHORT_DELAY);
  }
  else if (letter == "E")
  {
    PORTB = PORTB ^ (1 << LED_GREEN);
    _delay_ms(SHORT_DELAY);
    PORTB = PORTB ^ (1 << LED_GREEN);
    _delay_ms(SHORT_DELAY);
  }
  else if (letter == "2")
  {
    PORTB = PORTB ^ (1 << LED_GREEN);
    _delay_ms(SHORT_DELAY);
    PORTB = PORTB ^ (1 << LED_GREEN);
    _delay_ms(SHORT_DELAY);
    PORTB = PORTB ^ (1 << LED_GREEN);
    _delay_ms(SHORT_DELAY);
    PORTB = PORTB ^ (1 << LED_GREEN);
    _delay_ms(SHORT_DELAY);
    PORTB = PORTB ^ (1 << LED_GREEN);
    _delay_ms(LONG_DELAY);
    PORTB = PORTB ^ (1 << LED_GREEN);
    _delay_ms(LONG_DELAY);
    PORTB = PORTB ^ (1 << LED_GREEN);
    _delay_ms(LONG_DELAY);
    PORTB = PORTB ^ (1 << LED_GREEN);
    _delay_ms(LONG_DELAY);
    PORTB = PORTB ^ (1 << LED_GREEN);
    _delay_ms(LONG_DELAY);
    PORTB = PORTB ^ (1 << LED_GREEN);
    _delay_ms(LONG_DELAY);
  }
}

/* Variables ----------------------------------------------------------*/

/* Function prototypes ------------------------------------------------*/

/* Functions ----------------------------------------------------------*/
/**
 * Toggle one LED and use the function from the delay library.
 */
int main(void)
```



```c
{
  // Set pin as output in Data Direction Register
  // DDRB = DDRB or 0010 0000
  DDRB = DDRB | (1 << LED_GREEN);

  // Set pin LOW in Data Register (LED off)
  // PORTB = PORTB and 1101 1111
  PORTB = PORTB & ~(1 << LED_GREEN);

  // Infinite loop
  while (1)
  {
    morse("D");
    _delay_ms(BETWEEN_DELAY);
    morse("E");
    _delay_ms(BETWEEN_DELAY);
    morse("2");
    _delay_ms(BETWEEN_DELAY);
  }

  // Will never reach this
  return 0;
}

/* Interrupt routines ------------------------------------------------*/

```