# Lab 1: Masauso Lungu

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

Link to Lab Assignment 1

## Blink example

1. What is the meaning of the following binary operators in C?
    - `|` - Bitwise operator `OR`
    - `&` - Bitwise operator `AND`
    - `^` - Bitwise operator `XOR`
    - `~` - Bitwise operator `NOT`
    - `<<` - Left shift operator
    - `>>` - Right shift operator
2. Complete truth table with operators: `|`, `&`, `^`, `~`

| b | a | b or a | b and a | b xor a | not b |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 1 |
| 0 | 1 | 1 | 0 | 1 | 1 |
| 1 | 0 | 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 1 | 0 | 0 |

## Morse code

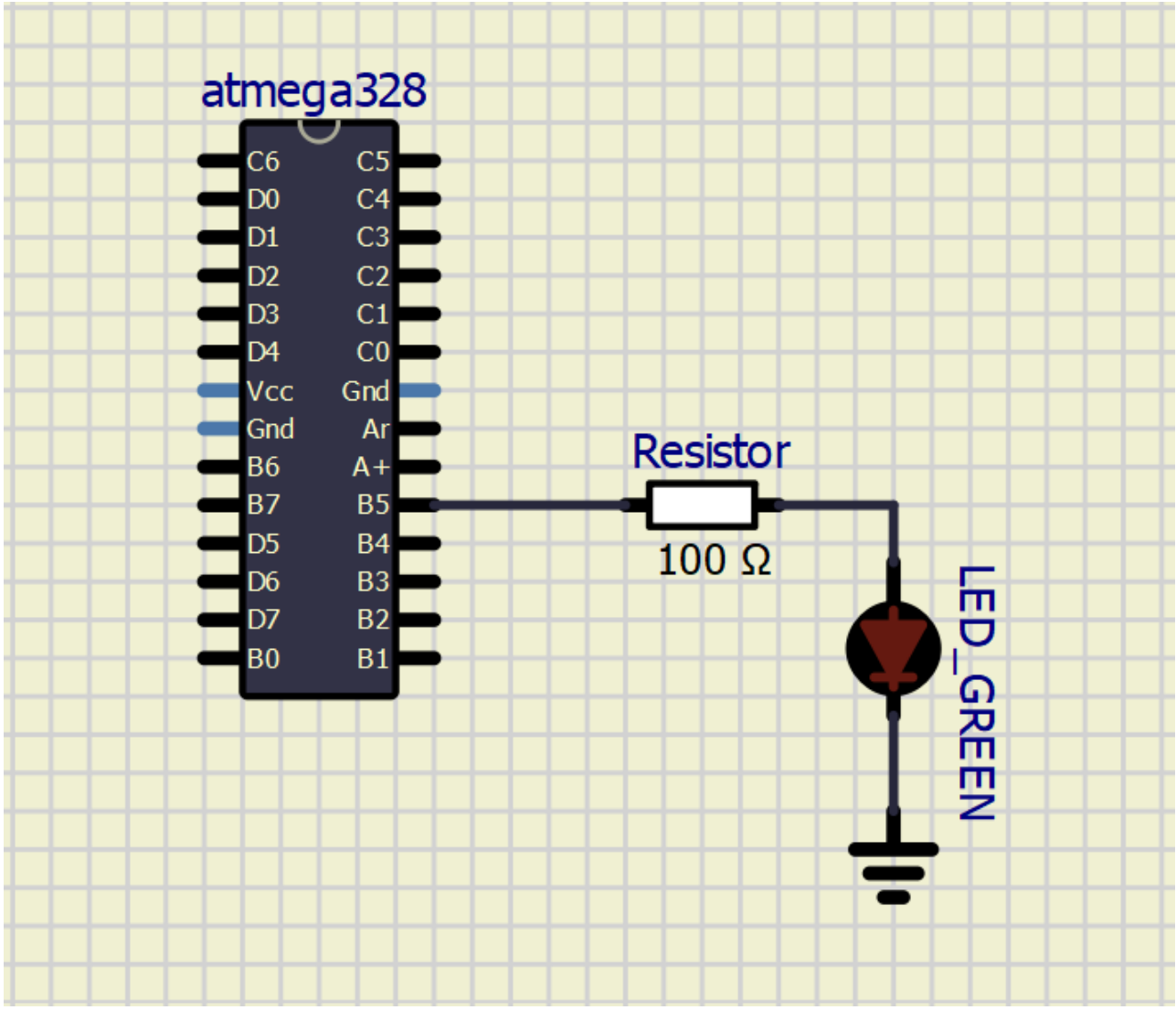
1. Listing of C code with syntax highlighting which repeats one "dot" and one "comma" (BTW, in Morse code it is letter `A`) on a LED:

```
/*
 * Morse_code.c
 *
 * Created: 9/28/2021 6:01:07 PM
 * Author : masau
 */

#define LED_GREEN   PB5   // AVR pin where green LED is connected
#define DOT_DELAY   500   // Delay in milliseconds
#define DASH_DELAY  1500  // DOT_DELAY*3

#define SHORT_DELAY 300    // Delay in milliseconds
#define LONG_DELAY  900    // Delay in milliseconds

#ifndef F_CPU           // Preprocessor directive allows for conditional
                        // compilation. The #ifndef means "if not defined".
```

```
# define F_CPU 16000000 // CPU frequency in Hz required for delay
#endif                  // The #ifndef directive must be closed by #endif

/* Includes ------------------------------------------------------------*/
/* Include another C language file into the current file at the location
 * of the #include statement prior to compiling the source code.
 */
#include <util/delay.h> // Functions for busy-wait delay loops
#include <avr/io.h>     // AVR device-specific IO definitions

/* Function definitions -----------------------------------------------*/
/**********************************************************************
 * Function: Main function where the program execution begins
 * Purpose:  Toggle one LED to display DE2 in Morse and use delay library.
 * Returns:  none
 **********************************************************************/
int main(void)
{
    // 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)
    {
        // Pause several milliseconds
        _delay_ms(LONG_DELAY*3);  // pause before the next cycle

        // Invert LED in Data Register
        // PORTB = PORTB xor 0010 0000

        // Sending symbol D (  _ • • )
        PORTB = PORTB ^ (1<<LED_GREEN);  // dash on
        _delay_ms(DASH_DELAY);
        PORTB = PORTB ^ (1<<LED_GREEN);  // off
        _delay_ms(SHORT_DELAY);          // pause between characters

        PORTB = PORTB ^ (1<<LED_GREEN);  // dot on
        _delay_ms(DOT_DELAY);
        PORTB = PORTB ^ (1<<LED_GREEN);  // dot off
        _delay_ms(SHORT_DELAY);

        PORTB = PORTB ^ (1<<LED_GREEN);  // dot on
        _delay_ms(DOT_DELAY);
        PORTB = PORTB ^ (1<<LED_GREEN);  // dot off
        _delay_ms(LONG_DELAY);           // pause between D and E

        // Sending Symbol E ( • )
        PORTB = PORTB ^ (1<<LED_GREEN);  // dot on
        _delay_ms(DOT_DELAY);
```

```
            PORTB = PORTB ^ (1<<LED_GREEN);  // dot off
            _delay_ms(LONG_DELAY);           // pause between E and 2

            //Sending 2 ( • • _ _ _ )
            PORTB = PORTB ^ (1<<LED_GREEN);  // first dot on
            _delay_ms(DOT_DELAY);
            PORTB = PORTB ^ (1<<LED_GREEN);  // dot off
            _delay_ms(SHORT_DELAY);          // pause between characters

            PORTB = PORTB ^ (1<<LED_GREEN);  // 2nd dot on
            _delay_ms(DOT_DELAY);
            PORTB = PORTB ^ (1<<LED_GREEN);  // dot off
            _delay_ms(SHORT_DELAY);

            PORTB = PORTB ^ (1<<LED_GREEN);  // 1st dash on
            _delay_ms(DASH_DELAY);
            PORTB = PORTB ^ (1<<LED_GREEN);  // off
            _delay_ms(SHORT_DELAY);

            PORTB = PORTB ^ (1<<LED_GREEN);  // 2nd dash on
            _delay_ms(DASH_DELAY);
            PORTB = PORTB ^ (1<<LED_GREEN);  // off
            _delay_ms(SHORT_DELAY);

            PORTB = PORTB ^ (1<<LED_GREEN);  // 3rd dash on
            _delay_ms(DASH_DELAY);
            PORTB = PORTB ^ (1<<LED_GREEN);  // off


    }

    // Will never reach this
    return 0;
}
```

2. Scheme of Morse code application, i.e. connection of AVR device, LED, resistor, and supply voltage. The image can be drawn on a computer or by hand. Always name all components and their values!


atmega328
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
Resistor
100 Ω
LED_GREEN