Name: MASAUSO LUNGU 209533

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

Masauso Lungu

Lab 3: User library for GPIO control

Learning objectives

The purpose of this laboratory exercise is to learn how to create your own library in C. Specifically, it will be a library for controlling GPIO (General Purpose Input/Output) pins.

1 Preparation tasks (done before the lab at home)

1. Fill in the following table and enter the number of bits and numeric range for the selected data types defined by C.

Data type	Number of bits	Range	Description
`uint8_t`	8	0, 1, ..., 255	Unsigned 8-bit integer
`int8_t`	8	-128,-127...127	Signed 8-bit integer
`uint16_t`	16	0,1...65538	Unsigned 16-bit integer
`int16_t`	16	-32768...32767	Signed 16-bit integer
`float`	32	-3.4e+38 ,..., 3.4e+38	Single-precision floating-point
`void`	*	*	Used for pointers

*Varies with the system type

2. Any function in C contains a declaration (function prototype), a definition (block of code, body of the function); each declared function can be executed (called).

Study this article and complete the missing sections in the following user defined function declaration, definition, and call.

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

// Function declaration (prototype)
uint16_t calculate(uint8_t, uint8_t );

int main(void)
{
```

```
    uint8_t a = 156;
    uint8_t b = 14;
    uint16_t c;

    // Function call
    c = calculate(a, b);

    while (1)
    {
    }
    return 0;
}

// Function definition (body)
uint16_t calculate(uint8_t x, uint8_t y)
{
    uint16_t result;    // result = x^2 + 2xy + y^2

    result = x*x;
    result += 2*x*y;
    result += y*y;
    return result;
}
```

2. GPIO library

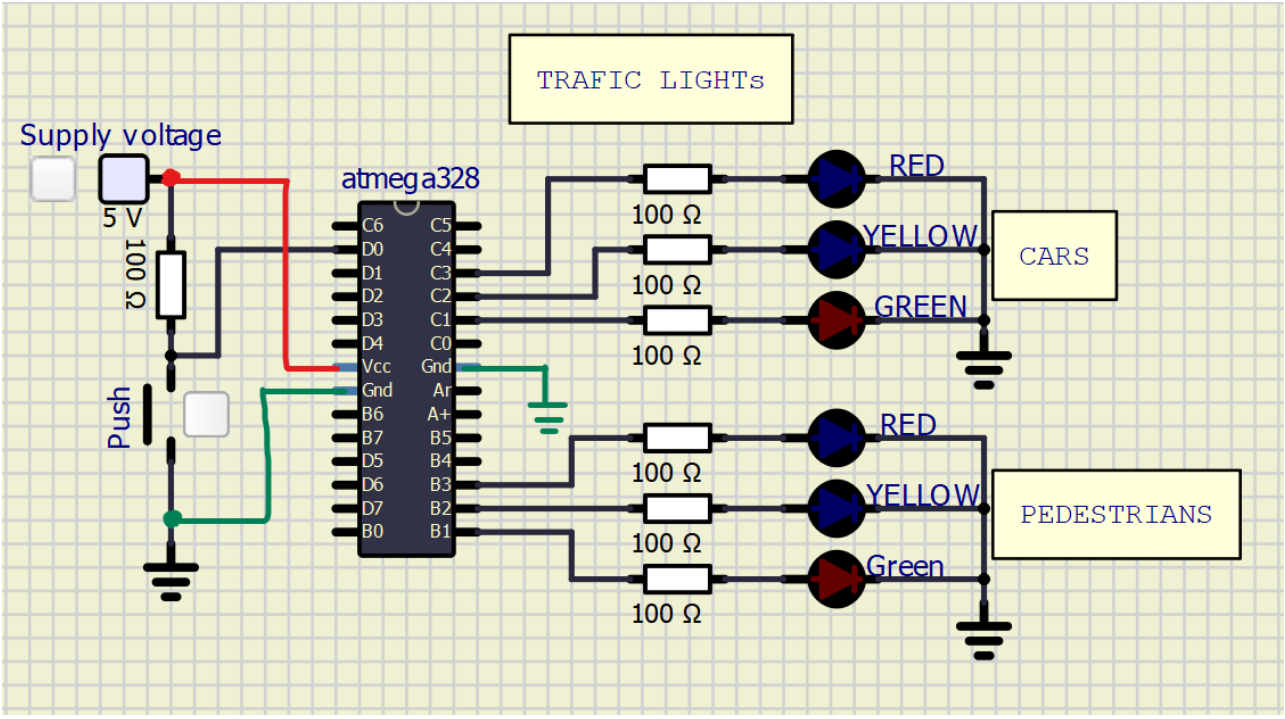
1. In your words, describe the difference between the declaration and the definition of the function in C.
    - Function declaration: Provides the compiler with information about the function's `name`, `return type`, and `parametors` it takes.
    - Function definition : Provides the actual body of the function.
2. Part of the C code listing with syntax highlighting, which toggles LEDs only if push button is pressed. Otherwise, the value of the LEDs does not change. Use function from your GPIO library. Let the push button is connected to port D:

```
    /* Defines -----------------------------------------------------------*/
#define LED_GREEN   PB5     // AVR pin where green LED is connected
#define LED_RED     PC0     // AVR pin where green LED is connected
#define BTN         PD0
#define BLINK_DELAY 500
#ifndef F_CPU
# define F_CPU 16000000     // CPU frequency in Hz required for delay
#endif

/* Includes ----------------------------------------------------------*/
#include <util/delay.h>     // Functions for busy-wait delay loops
#include <avr/io.h>         // AVR device-specific IO definitions
#include "gpio.h"           // GPIO library for AVR-GCC
```

```c
/* Function definitions ----------------------------------------------*/
/**************************************************************************
 * Function: Main function where the program execution begins
 * Purpose:  Toggle two LEDs when a push button is pressed. Functions
 *           from user-defined GPIO library is used.
 * Returns:  none
 *************************************************************************/
int main(void)
{
    // Green LED at port B
    GPIO_config_output(&DDRB, LED_GREEN);
    GPIO_write_low(&PORTB, LED_GREEN);

    // Configure the second LED at port C
    GPIO_config_output(&DDRC, LED_RED);
    GPIO_write_high(&PORTC, LED_RED);

    // Configure Push button at port D and enable internal pull-up resistor
    GPIO_config_input_pullup(&DDRD, BTN);


    // Infinite loop
    while (1)
    {
        // Pause several milliseconds
        _delay_ms(BLINK_DELAY);

        // WRITE YOUR CODE HERE
        if(!GPIO_read(&PIND, BTN))
        {
            GPIO_toggle(&PORTB, LED_GREEN);
            GPIO_toggle(&PORTC, LED_RED);
        }
    }

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

Traffic light

1. Scheme of traffic light application with one red/yellow/green light for cars and one red/green light for pedestrians. Connect AVR device, LEDs, resistors, one push button (for pedestrians), and supply voltage. The image can be drawn on a computer or by hand. Always name all components and their values!


TRAFIC LIGHTs
Supply voltage
5 V
100 Ω
Push
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
100 Ω
100 Ω
100 Ω
100 Ω
100 Ω
100 Ω
RED
YELLOW
GREEN
CARS
RED
YELLOW
Green
PEDESTRIANS