Lab 2: Control of GPIO, LED, push button

- Tables for DDRB, PORTB, and their combination

DDRB	Description
0	Input pin
1	Output pin

PORTB	Description
0	Output low value
1	Output high value

DDRB	PORTB	Direction	Internal pull-up resistor	Description
0	0	input	no	Tri-state, high-impedance
0	1	input	no	Tri-state, high-impedance
1	0	output	no	Output Low
1	1	output	no	Output High

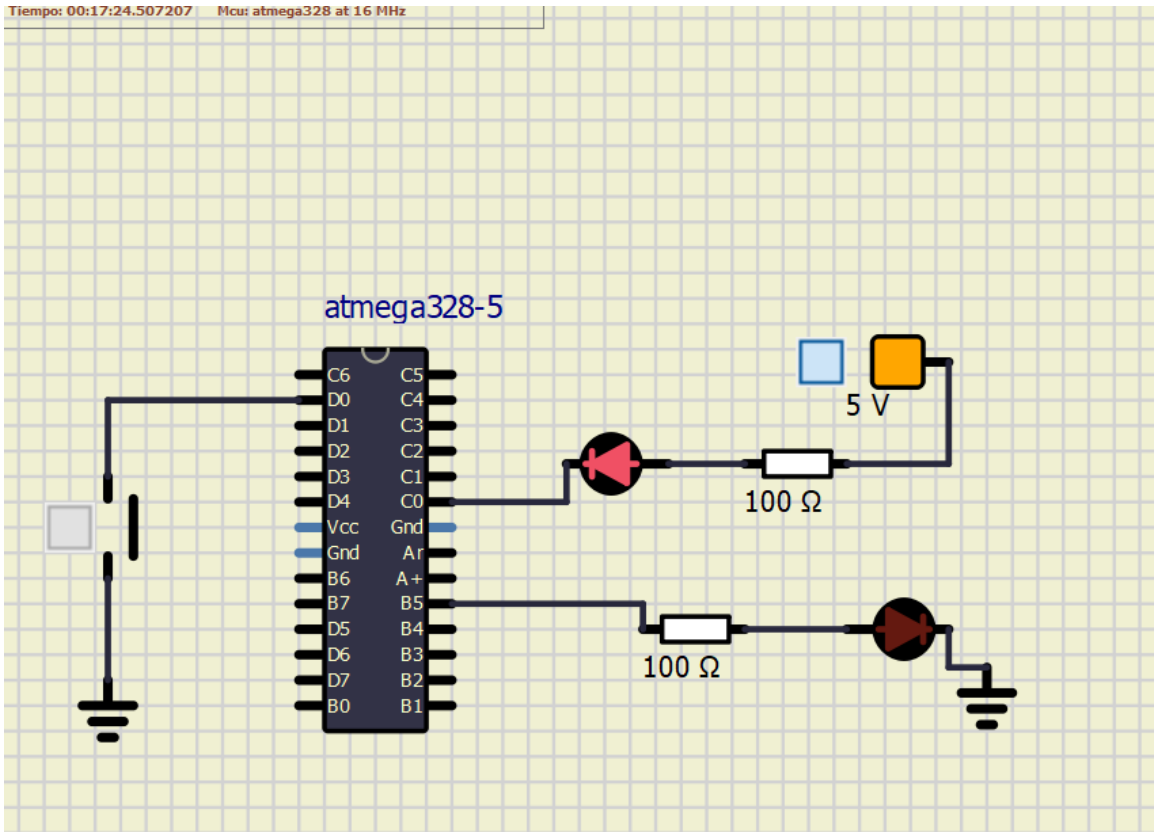
- Table with input/output pins available on ATmega328P

Port	Pin	Input/output usage?
A	x	Microcontroller ATmega328P does not contain port A
B	0	Yes (Arduino pin 8)
	1	Yes (Arduino pin -9)
	2	Yes (Arduino pin -10)
	3	Yes (Arduino pin -11)
	4	Yes (Arduino pin 12)
	5	Yes (Arduino pin 13)
	6	No
	7	No
C	0	Yes (Arduino pin A0)
	1	Yes (Arduino pin A1)
	2	Yes (Arduino pin A2)
	3	Yes (Arduino pin A3)
	4	Yes (Arduino pin A4)
	5	Yes (Arduino pin A5)
	6	No

	7	No
D	0	Yes (Arduino pin RX<-0)
	1	Yes (Arduino pin TX->1)
	2	Yes (Arduino pin 2)
	3	Yes (Arduino pin ~3)
	4	Yes (Arduino pin 4)
	5	Yes (Arduino pin ~5)
	6	Yes (Arduino pin ~6)
	7	Yes (Arduino pin 7)

```c
/*****************************************************************************
 *
 * Alternately toggle two LEDs when a push button is pressed.
 * ATmega328P (Arduino Uno), 16 MHz, AVR 8-bit Toolchain 3.6.2
 *
 * Copyright (c) 2018-2020 Tomas Fryza
 * Dept. of Radio Electronics, Brno University of Technology, Czechia
 * This work is licensed under the terms of the MIT license. *
 *****************************************************************************/

/* Defines -----------------------------------------------------------*/
#define LED_GREEN   PB5     // AVR pin where green LED is connected
#define LED_RED   PC0     // AVR pin where red LED is connected
#define BUTTON   PD0     // AVR pin where the button is connected

#define SHORT_DELAY 500      // Delay in milliseconds
#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

/* Functions ---------------------------------------------------------*/
/**
 * Main function where the program execution begins. Toggle one LED
 * and use function from the delay library.
 */
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);



       // Set pin as output in Data Direction Register
       // DDRC = DDRC or 0010 0000
       DDRC = DDRC | (1<<LED_RED);


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

       /*PUSH BUTTON*/

       DDRD = DDRD & ~(1<<BUTTON);  // input
       PORTD = PORTD | (1<<BUTTON); // enable internal pull up

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

              if(bit_is_clear(PIND,BUTTON)){
```

```
            // Invert LED in Data Register
            // PORTB = PORTB xor 0010 0000
            PORTB = PORTB ^ (1<<LED_GREEN);
            PORTC = PORTC ^ (1<<LED_RED);


            }


    }

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



- Knight Rider application.

```
/*****************************************************************
 *
 * Knight Rider application.
 * ATmega328P (Arduino Uno), 16 MHz, AVR 8-bit Toolchain 3.6.2
 *
 * Author: Elena Arjona Bustos
 *
 *****************************************************************/

/* Defines -----------------------------------------------------------*/
#define LED_RED1   PC0     // AVR pin where red LED 1 is connected
#define LED_RED2   PC1     // AVR pin where red LED 2 is connected
#define LED_RED3   PC2     // AVR pin where red LED 3 is connected
#define LED_RED4   PC3     // AVR pin where red LED 4 is connected
#define LED_RED5   PC4     // AVR pin where red LED 5 is connected
#define BUTTON   PD0     // AVR pin where the button is connected

#define SHORT_DELAY 250      // Delay in milliseconds

#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

/* Functions ---------------------------------------------------------*/
/**
 * Main function where the program execution begins.
 */

int leds[] ={LED_RED1,LED_RED2,LED_RED3,LED_RED4,LED_RED5};

int main(void)
{

    int i = 0;
    int r = 0;

    // Set pin as output in Data Direction Register
    // DDRC = DDRC or 0010 0000
    DDRC = DDRC | (1<<LED_RED1);
    DDRC = DDRC | (1<<LED_RED2);
    DDRC = DDRC | (1<<LED_RED3);
    DDRC = DDRC | (1<<LED_RED4);
    DDRC = DDRC | (1<<LED_RED5);


    // Set pin LOW in Data Register (LED off)
    // PORTC = PORTC and 1101 1111
    PORTC = PORTC | (1<<LED_RED1);
    PORTC = PORTC | (1<<LED_RED2);
    PORTC = PORTC | (1<<LED_RED3);
    PORTC = PORTC | (1<<LED_RED4);
    PORTC = PORTC | (1<<LED_RED5);

    /*PUSH BUTTON*/

    DDRD = DDRD & ~(1<<BUTTON);  // input
```

```
    PORTD = PORTD | (1<<BUTTON); // enable internal pull up
    // Infinite loop
    while (1)
    {
        // Pause several milliseconds
        _delay_ms(SHORT_DELAY);

          PORTC = PORTC | (1<<leds[i]);

          if(bit_is_clear(PIND,BUTTON)){


             if(i == 4){

                    r = 1;
                    PORTC = PORTC | (1<<leds[4]);

             }else if(i == 0){

                    r = 0;
                    PORTC = PORTC | (1<<leds[0]);

             }


             if(r == 0){

                    i++;

             }else{

                    i--;

             }

          PORTC = PORTC & ~(1<<leds[i]);

          }

          }

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