Lab 2: Pavel Vaněk

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

https://github.com/Bobik77/Digital-electronics-2

Active-low and active-high LEDs

1. Complete tables according to the AVR manual.

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

DDRB	PORTB	Direction	Internal pull-up resistor	Description
0	1	input	yes*	MCU will source current if ext. pulled low
1	0	output	no	Output low (sink)
1	1	output	no	Output high (source)

must be globaly alloved in PUD register*

Port	Pin	Input/output usage?
A	x	Microcontroller ATmega328P does not contain port A
B	0	Yes (Arduino pin D8)
	1	Yes (Arduino pin D9)
	2	Yes (Arduino pin D10)
	3	Yes (Arduino pin D11)
	4	Yes (Arduino pin D12)
	5	Yes (Arduino pin D13)
	6	No
	7	No
C	0	Yes (Arduino pin A0/D14)
	1	Yes (Arduino pin A1/D15)
	2	Yes (Arduino pin A2/D16)
	3	Yes (Arduino pin A3/D17)
	4	Yes (Arduino pin A4/D18
	5	Yes (Arduino pin A5/D19)
	6	Yes (Arduino pin RST)
	7	No
D	0	Yes (Arduino pin RX D0)
	1	Yes (Arduino pin TX D1)
	2	Yes (Arduino pin D2)

Port	Pin	Input/output usage?
	3	Yes (Arduino pin D3)
	4	Yes (Arduino pin D4)
	5	Yes (Arduino pin D5)
	6	Yes (Arduino pin D6)
	7	Yes (Arduino pin D7)

2. Part of the C code listing with syntax highlighting, which blinks alternately with a pair of LEDs; let one LED is connected to port B and the other to port C:

```c
int main(void)
{
    // Green LED at port B
    // Set pin as output in Data Direction Register...
    DDRB = DDRB | (1<<LED_GREEN);
    PORTB = PORTB | (1<<LED_GREEN);
    // ...and turn LED off in Data Register

    // Configure the second LED at port C
    DDRC = DDRC | (1<<LED_EXTERNAL);
    PORTC = PORTC & ~(1<<LED_EXTERNAL);

    // Infinite loop
    while (1)
    {
        // Pause several milliseconds
        _delay_ms(BLINK_DELAY);
        PORTB = PORTB ^ (1<<LED_GREEN);
        PORTC = PORTC ^ (1<<LED_EXTERNAL);
        // WRITE YOUR CODE HERE
    }

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

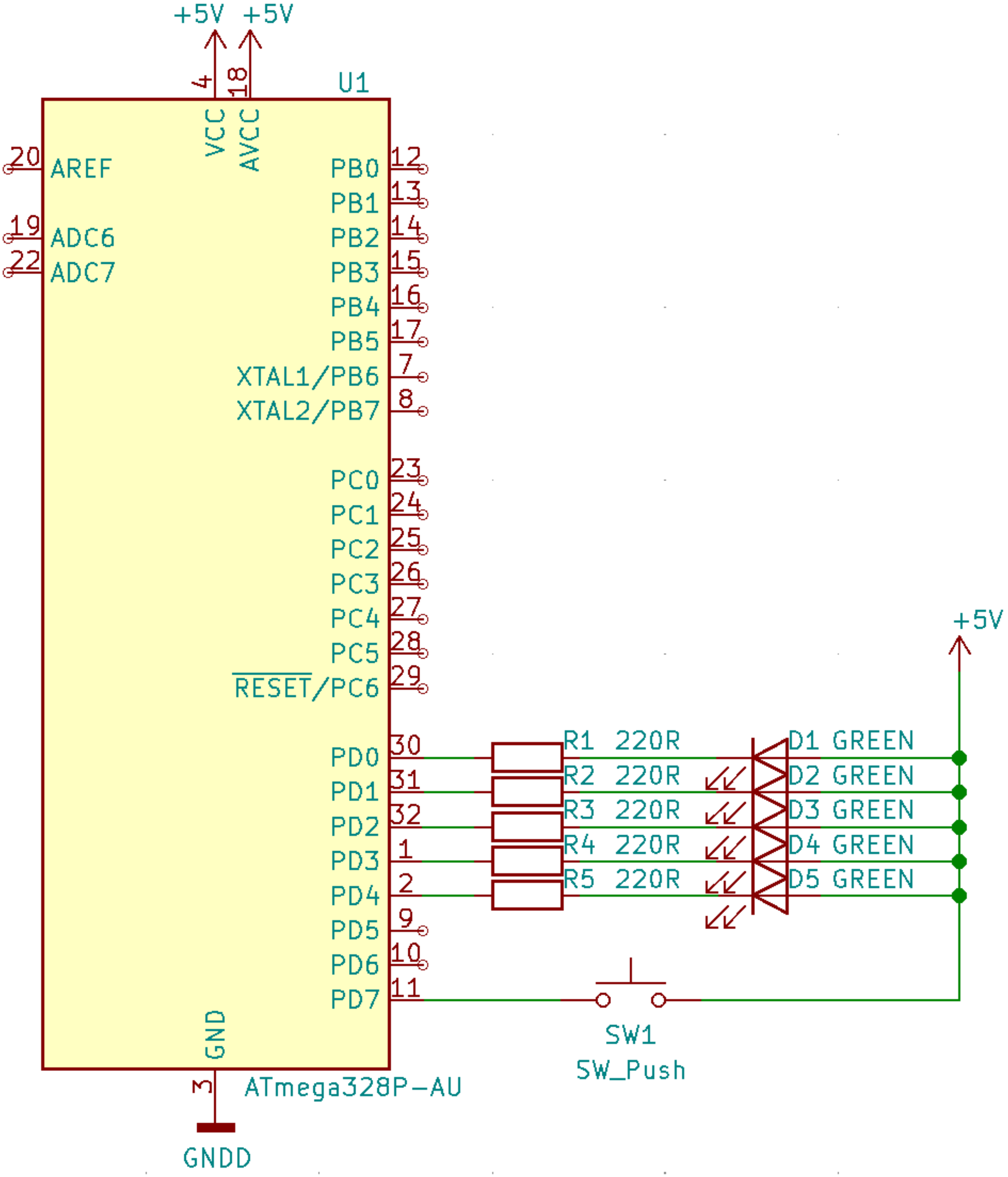
Push button

1. 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. Let the push button is connected to port D:

```c
int main(void)
{
    // Green LED at port B
    // Set pin as output in Data Direction Register...
    DDRB = DDRB | (1<<LED_GREEN);
    PORTB = PORTB | (1<<LED_GREEN);
    // ...and turn LED off in Data Register

    // Configure the second LED at port C
    DDRC = DDRC | (1<<LED_EXTERNAL);
    PORTC = PORTC & ~(1<<LED_EXTERNAL);

    // Configure Push button at port D and enable internal pull-up resistor
    DDRC = DDRC & ~(1<<PUSH_BTN);
    PORTC = PORTC | (1<<PUSH_BTN);

    // Infinite loop
    while (1)
    {
        if (bit_is_clear(PINC, PUSH_BTN))
        {
            PORTB = PORTB ^ (1<<LED_GREEN);
            PORTC = PORTC ^ (1<<LED_EXTERNAL);
            // Pause several milliseconds
            _delay_ms(BLINK_DELAY);
        }
    }

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

Knight Rider

1. Scheme of Knight Rider application, i.e. connection of AVR device, five LEDs, resistors, one push button, and supply voltage. The image can be drawn on a computer or by hand. Always name all components and their values!


+5V
+5V
4
18
U1
VCC
AVCC
20
AREF
19
ADC6
22
ADC7
PB0 12
PB1 13
PB2 14
PB3 15
PB4 16
PB5 17
XTAL1/PB6 7
XTAL2/PB7 8
PC0 23
PC1 24
PC2 25
PC3 26
PC4 27
PC5 28
RESET/PC6 29
PD0 30
PD1 31
PD2 32
PD3 1
PD4 2
PD5 9
PD6 10
PD7 11
GND
3
ATmega328P-AU
GNDD
+5V
R1 220R
R2 220R
R3 220R
R4 220R
R5 220R
D1 GREEN
D2 GREEN
D3 GREEN
D4 GREEN
D5 GREEN
SW1
SW_Push