IO Port programming

For ATMega328p AVR interfacing

ATmega328p IO lines

- ADC

  [5:0] and they are located on the same physical ports as PORTC

- PORTB
- PORTC
- PORTD

Each PORT (B, C, D) consists of:

1- Input register: RO (PINx) Reads data from physical outer ports.

2- Data register: RW (PORTx) The register that contains the data of the PORT. It can be programmed to get hardcoded or read by any means. Depending on the nature of the port.

3- Direction register: RW (DDRx) The register that determines the nature of each bit in the outer physical port register 0->input, 1->output. If the bit is 1, the value of the outer port is the value of the corresponding bit in PORTx. While if it’s 0, the value has to be read from PINx. If a bit was input, then the value is read from PINx using PINx explicitly, and it’s High impedance by nature If you want to make it pull up, then set the corresponding but in PORTx.

Reads from outside world are carried out into port PINx. Writes to the outside world is handled from PORTx.

Syntax:

Set value in a register

```
PORTC = 5; // 5 = 0x05 =  0b101 // You can use any of those.
```

Deal with a specific Pin without affecting other pins.

```
PORTC = PORTC | (1 << PC5); // set pin // PC5 not 5. Has to match PORTx

PORTD = PORTD & ~(1 << PD4); // reset pin // PD4 not 4

PORTB = PORTB ^ ( 1 << PB2); // Toggle pin
```

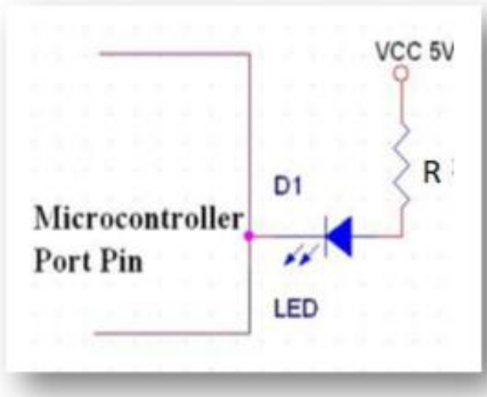
Let's set pin2 of portB to be Active low input (With Pullup resistor)

```
DDRB = 0b1111011; // You can completely set the value.

DDRB = DDRB & ~( 1 << PB2); // Or deal with pin solely.

PORTB = PORTB | ( 1 << PB2); // Toggle Pullup.
```

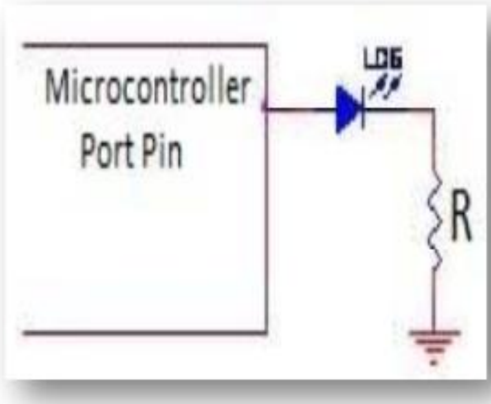
To check a bit in PINx. (Has to be input of course)

```
DDRB = 0xFB; // PB2 input.

PORTB |= (1 << PB2); // pullup

If (PINB & 0x04) // PINB => Input _ _ _ _ 1 _ _. To check PB2 is 1
```

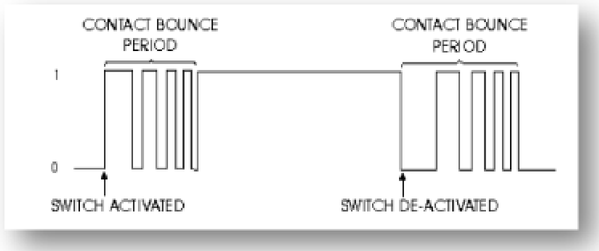
Negative Logic



Positive Logic

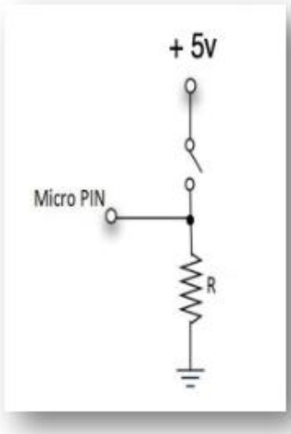


Switch de-bounce problem

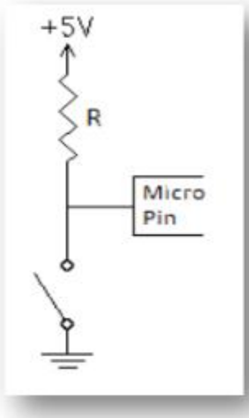


- Could be handled using software or hardware.
- It relies on the fact that bouncing takes a maximum period of 20-30 ms.
- The basic idea is to implement a delay after the first detected edge, during which no scanning for the switch is done. after the delay period is finished, scanning can proceed (Exercise 3).

Pull Down Resistor



Pull UP Resistor


+5V
R
Micro Pin