Sistemas Baseados em Microprocessadores

Mestrado Integrado em Engenharia Eletrotécnica e de Computadores

U. PORTO
Faculdade de Engenharia

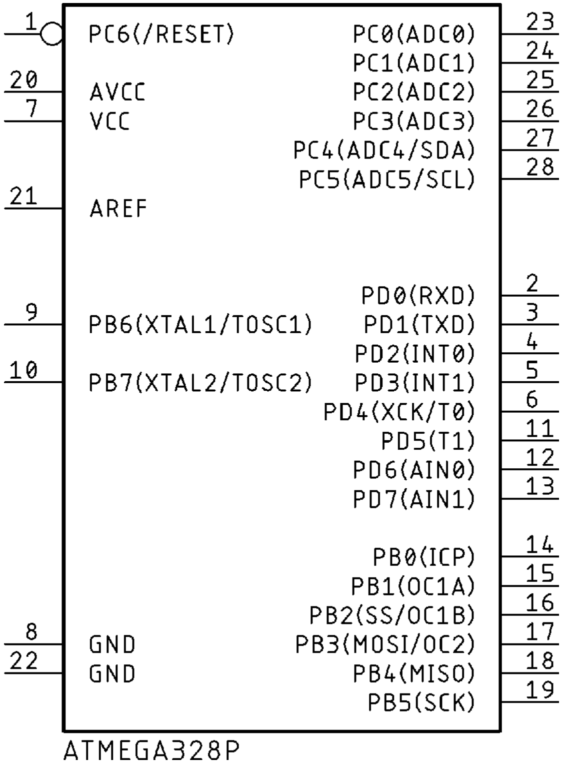
ATmega328p – Input/Output Ports

João Paulo de Sousa

ATmega328p - IO Ports

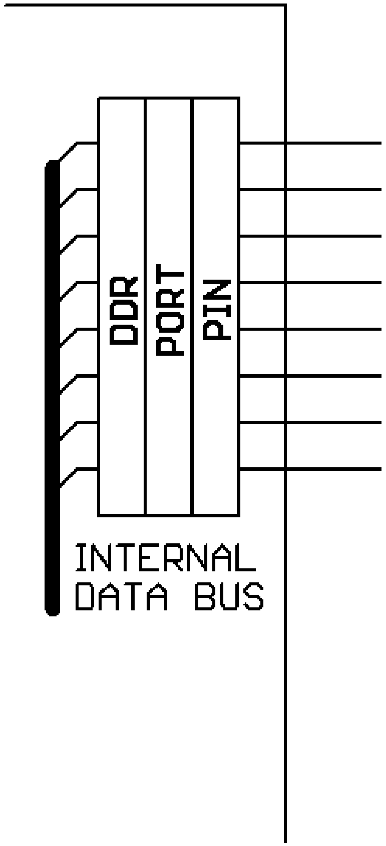
- 3 Bidirecional ports (PB, PC, PD)
- Multifunctional pins:
  - 23 in total
    - PB0..5, PC0..5, PD0..7
    - PB6,PB7: Clk, PC6: RST
  - Internal pull-up resistors
  - Maximum currents
    - 40 mA per pin
    - 200 mA in total

Challenge:
Find this in the datasheet



ATmega328p - IO Ports

- Each port has 3 associated registers: **DDR**, **PORT**, **PIN**
  - DDR: Configuration register (1:Out, 0:In)
  - PORT: output register
  - PIN: input register
- 9 registers in total:
  - DDRB, PORTB and PINB
  - DDRC, PORTC and PINC
  - DDRD, PORTD and PIND



ATmega328p - IO Ports as digital outputs

- Configuration as output port:
  - Write 1 in the desired DDR register bits
- Use as output port:
  - Write to the PORT register

ATmega328p - IO Ports as digital outputs

- Configuration as output port:
  - Write 1 in the desired DDR register bits
- Use as output port:
  - Write to the PORT register

```
/* PB3..PB0 as outputs (PB7..PB4 as inputs) */
DDRB = 0b00001111;
DDRB = 0x0F;

/* Set PB0 and PB1, clear PB2 and PB3 */
PORTB = 0x03;
PORTB = 0b00000011;

/* Set PB0 and PB1, don’t change the others */
PORTB = PORTB | 0x03;
PORTB = PORTB | 0b00000011;

/* Toggle PB1 and PB2 */
PORTB = PORTB ^ 0x06;
PORTB = PORTB ^ 0b00000110;



/* PB7, PB6 and PB2 as outputs
   remaining pins unconfigured */
DDRB = DDRB | 0xC4;
DDRB = DDRB | 0b11000100;
DDRB = DDRB | (1<<2) | (3<<6);
```

Some syntax details in [embedded] C

- Logical operators (&& , | |)
  - Work on boolean values
  - Return a boolean value
- Bitwise operators (& , |)
  - Work on Integral values
  - Return Integral value
- Examples:

```
0x23 && 0x0F = 0x01
0x23 & 0x0F = 0x03
```

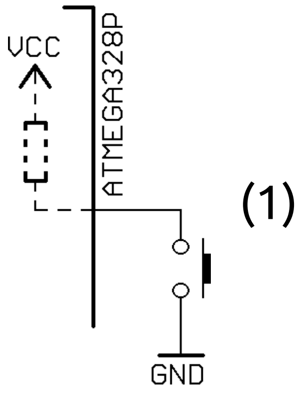
- Shift operators:
  - 1<<N = 2^N
  - 0<<N = 0
- Examples:

```
1<<5 = 100000b
1<<0 = 000001b
3<<4 = 110000b
0<<2 = 000000b
```

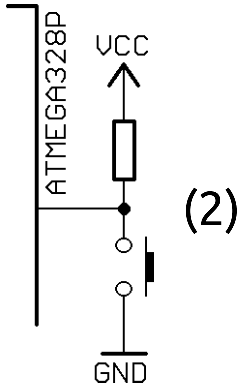
ATmega328p - IO Ports as digital inputs

- There must always be a pull-up resistor (R_P) at each input (why?)
  - It can be internal (1) or external (2)
  - Which value for the internal R_P ?

Challenge: find this in the datasheet
Hint: v_i=f(i_i)



- To activate an internal pull-up:
  - Write 0 at the required DDR bit
  - Write 1 at the required PORT bit



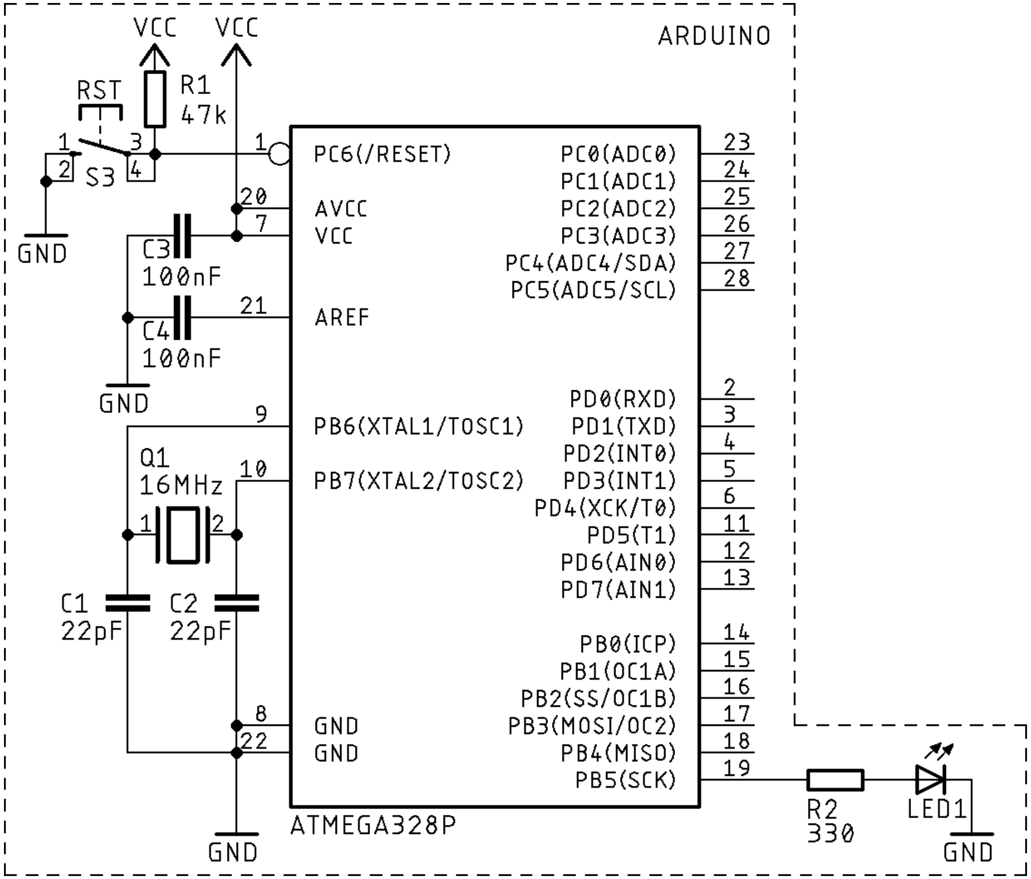
ATmega328p - IO Ports as digital inputs

- Configuration as input port:
  - Write 0 in the desired DDR register bits
  - [Write 1 in the desired PORT register bits]
- Use as input port:
  - Read from the PIN register

```
#define CAFOK PB7     /* CAFOK Sensor */
#define DEVOK PB6     /* DEVOK Sensor */
#define STOP PB5      /* STOP Button */

unsigned char nstate,sensors;

/* PB7..PB5 as inputs with internal pull-up
 * resistors (PB4..PB0 as outputs) */
DDRB = 0b00011111;
DDRB = 0x1F;
DDRB = ~((1<<CAFOK)|(1<<DEVOK)|(1<<STOP));
DDRB = ~(7<<5);

PORTB = 0b1110000;
PORTB = 0xE0;
PORTB = PORTB |
       (1<<CAFOK)|(1<<DEVOK)|(1<<STOP));

sensors = PINB & 0xE0;  /* Read PB7,PB6,PB5*/

/* if STOP==0 goto state 7 */
if (!(PINB&(1<<STOP))) nstate = 7;
```

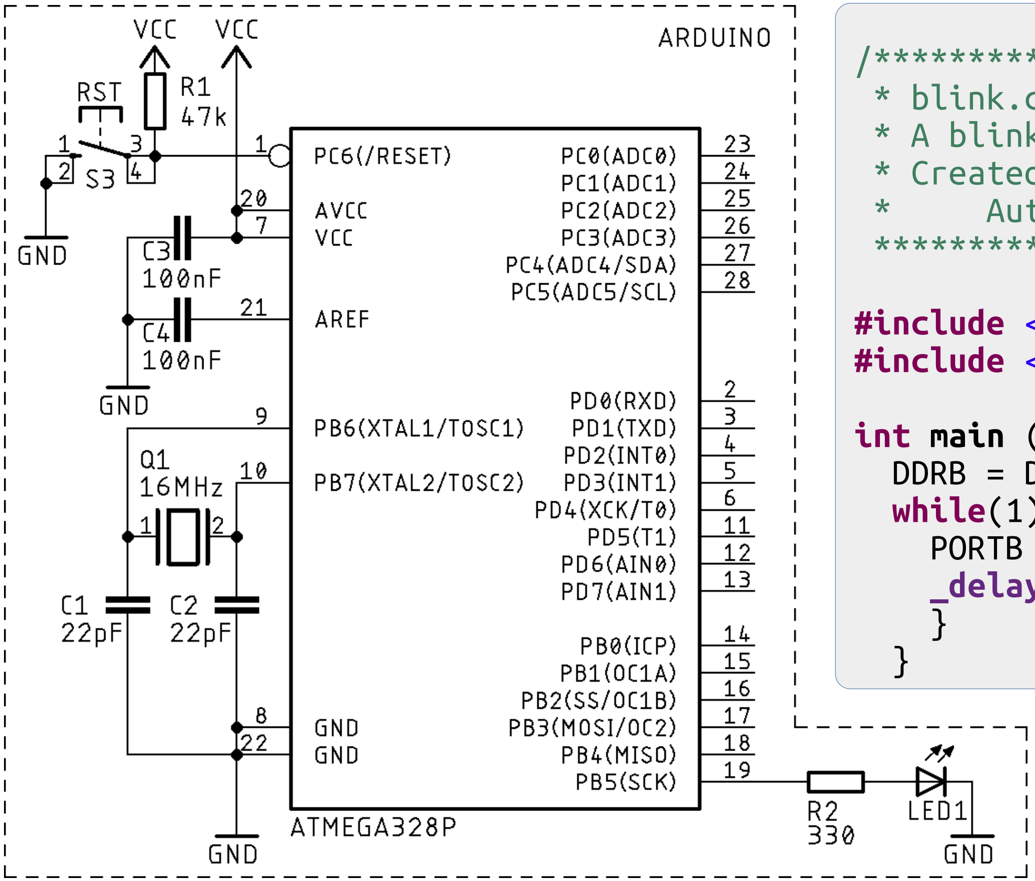
ATmega328p - IO Ports

Example 1 - Blinking LED



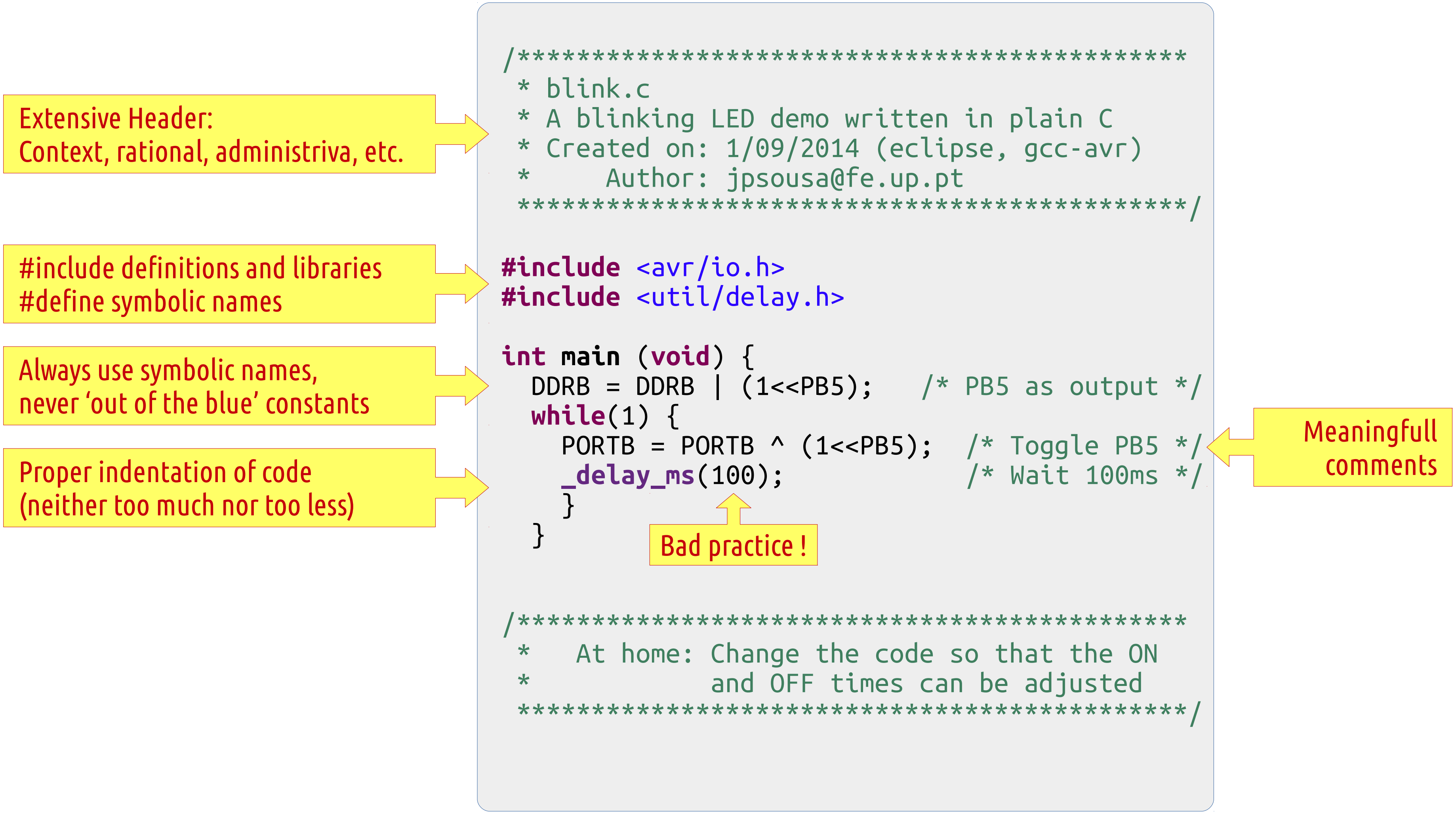
ATmega328p - IO Ports

Example 1 - Blinking LED



```
/*********************************************
 * blink.c
 * A blinking LED demo written in plain C
 * Created on: 1/09/2014 (eclipse, gcc-avr)
 *      Author: jpsousa@fe.up.pt
 *********************************************/

#include <avr/io.h>
#include <util/delay.h>

int main (void) {
  DDRB = DDRB | (1<<PB5);    /* PB5 as output */
  while(1) {
    PORTB = PORTB ^ (1<<PB5);  /* Toggle PB5 */
    _delay_ms(100);            /* Wait 100ms */
    }
  }
```

Good practices in [embedded] C



Extensive Header:
Context, rational, administriva, etc.

#include definitions and libraries
#define symbolic names

Always use symbolic names,
never ‘out of the blue’ constants

Proper indentation of code
(neither too much nor too less)

Meaningfull
comments

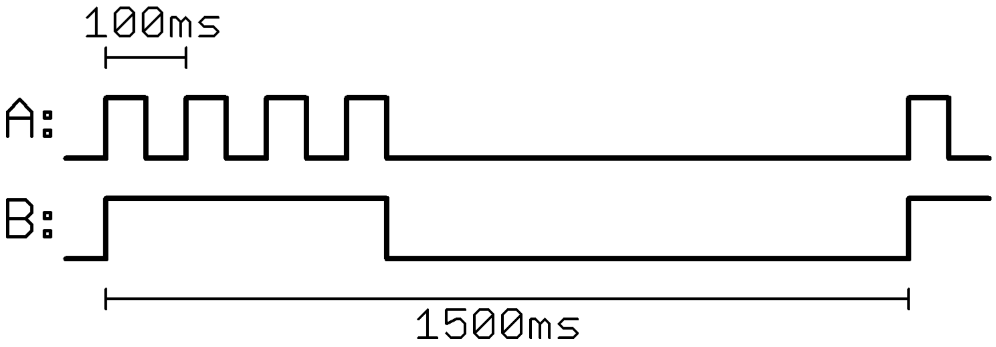
Bad practice !

```
/*********************************************
 * blink.c
 * A blinking LED demo written in plain C
 * Created on: 1/09/2014 (eclipse, gcc-avr)
 *     Author: jpsousa@fe.up.pt
 *********************************************/

#include <avr/io.h>
#include <util/delay.h>

int main (void) {
  DDRB = DDRB | (1<<PB5);   /* PB5 as output */
  while(1) {
    PORTB = PORTB ^ (1<<PB5);  /* Toggle PB5 */
    _delay_ms(100);            /* Wait 100ms */
    }
  }


/*********************************************
 *   At home: Change the code so that the ON
 *            and OFF times can be adjusted
 *********************************************/
```

ATmega328p - IO Ports

Example 2 - Visual and audible alarm

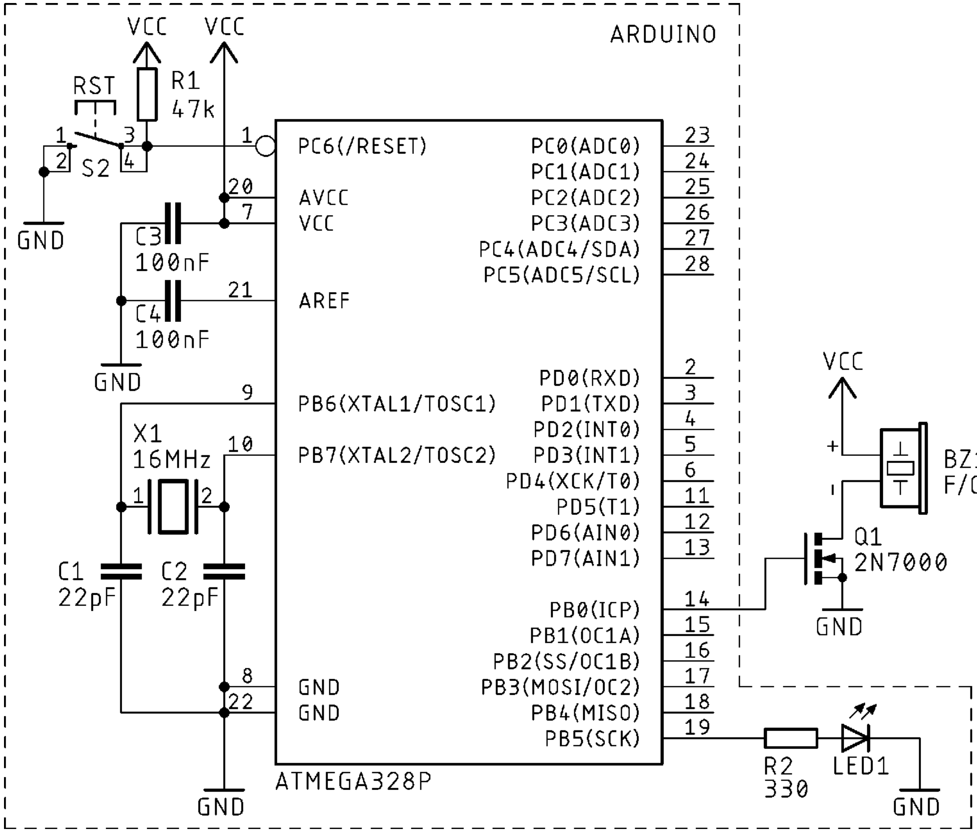
- Generate two distinct signals, at two different output pins of the uC, according to the waveforms shown:
  - Signal A: connected to a piezo buzzer
  - Signal B: connected to an LED



ATmega328p - IO Ports

Example 2 - Visual and audible alarm (hardware)

- Arduino boards already have an LED at PB5
- PB0 and PB5 as outputs
- Danger:
  - uC vs Arduino
  - Active buzzer!



ATmega328p - IO Ports

Example 2 - Visual and audible alarm (software)

```c
/*
 * alarm.c
 * A visual and audible alarm
 * written in plain C
 *  Created on: 14/09/2014 (eclipse, gcc-avr)
 *       Author: jpsousa@fe.up.pt
 */

/* Include register definitions */
#include <avr/io.h>

/* Include delays library */
#include <util/delay.h>
```

```c
void main(void) {
  unsigned char i;

  /* PB0 and PB5 as output */
  DDRB = DDRB | 0b00100001;

  while(1) {
    /* Set LED and Buzzer */
    PORTB = PORTB | 0b00100001;

    /* Toggle Buzzer 6 times */
    for(i=1;i<7;i++) {
      _delay_ms(50);
      PORTB = PORTB ^ 0b00000001;
    }
    _delay_ms(50);
    /* Clear LED and Buzzer */
    PORTB = PORTB & ~0b00100001;
    _delay_ms(1150);
  }
}
```

ATmega328p - IO Ports

Example 2 - Visual and audible alarm (better software)

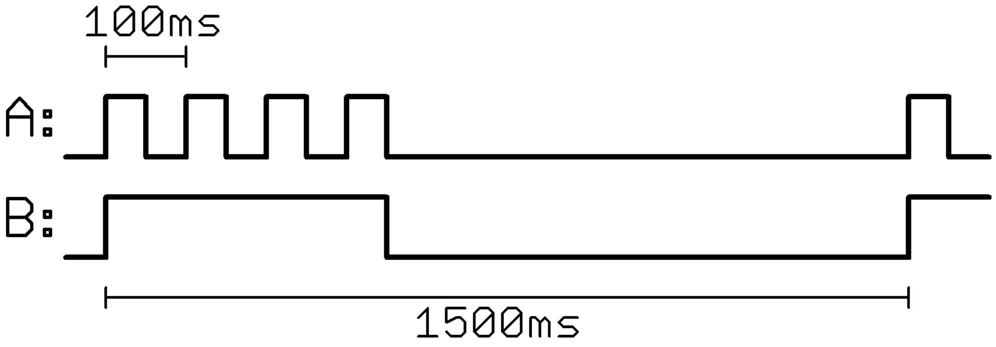
```c
/*********************************************
 * alarm.c
 * A visual and audible alarm written in C
 *  Created on: 14/09/2014 (eclipse, gcc-avr)
 *      Author: jpsousa@fe.up.pt
 ********************************************/
#include <avr/io.h>        /* Register defs */
#include <util/delay.h>   /* Delays library */

#define LED PB5           /* LED at pin PB5 */
#define BUZ PB0        /* Buzzer at pin PB0 */

#define SHORT 50    /* Short duration pulse */
#define LONG 1150    /* Long duration pulse */
```

```c
void main(void) {
  uint8_t i;               /* Pulse counter */
  DDRB=DDRB|(1<<LED)|(1<<BUZ);   /* Outputs */

  while(1) {           /* Repeat forever... */
    /* Set only LED and Buzzer */
    PORTB = PORTB | (1<<LED) | (1<<BUZ);

    /* Toggle Buzzer 6 times */
    for(i=1;i<7;i++) {
      _delay_ms(SHORT);
      PORTB = PORTB ^ (1<<BUZ);
    }
    _delay_ms(SHORT);

    /* Clear only LED and Buzzer */
    PORTB = PORTB & ~((1<<LED) | (1<<BUZ));
    _delay_ms(LONG);
  }
}
```

ATmega328p - IO Ports

Example 3 - Special visual and audible alarm

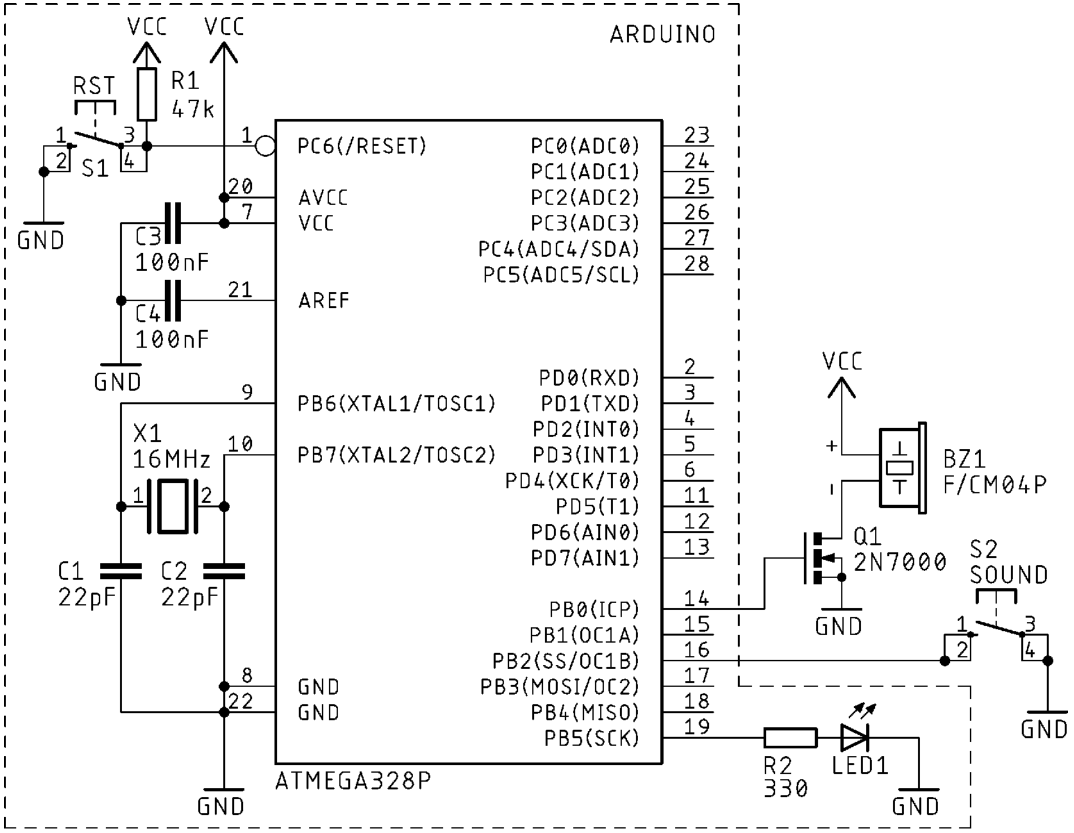
- Generate two distinct signals, at two different output pins of the uC, according to the waveforms shown
  - Signal A: connected to a piezo buzzer
  - Signal B: connected to an LED
- Signal B is continuously generated, signal A should only be generated when a pushbutton is pressed



ATmega328p - IO Ports

Example 3 - Hardware

- PB0 and PB5 as outputs
- PB2 as input
- Danger:
  - uC vs Arduino
  - Active buzzer
  - PB2 Internal pull-up



ATmega328p - IO Ports

Example 3 - Software

```c
/********************************************
 * special-alarm.c
 * A new visual and audible alarm in C
 *  Created on: 14/09/2014 (eclipse, gcc-avr)
 *      Author: jpsousa@fe.up.pt
 ********************************************/
#include <avr/io.h>        /* Register defs */
#include <util/delay.h>   /* Delays library */

#define LED PB5           /* LED at pin PB5 */
#define BUZ PB0        /* Buzzer at pin PB0 */
#define KEY PB2           /* KEY at pin PB2 */

#define SHORT 50    /* Short duration pulse */
#define LONG 1150    /* Long duration pulse */
```

```c
void hw_init(void) {
  /* set LED and Buzzer pins as outputs */
  DDRB = DDRB | (1<<LED)
              | (1<<BUZ);

  /* Set KEY pin as input
   * and activate its internal pull-up */
  DDRB = DDRB & ~(1<<KEY);
  PORTB = PORTB | 1<<KEY;
}
```

ATmega328p - IO Ports

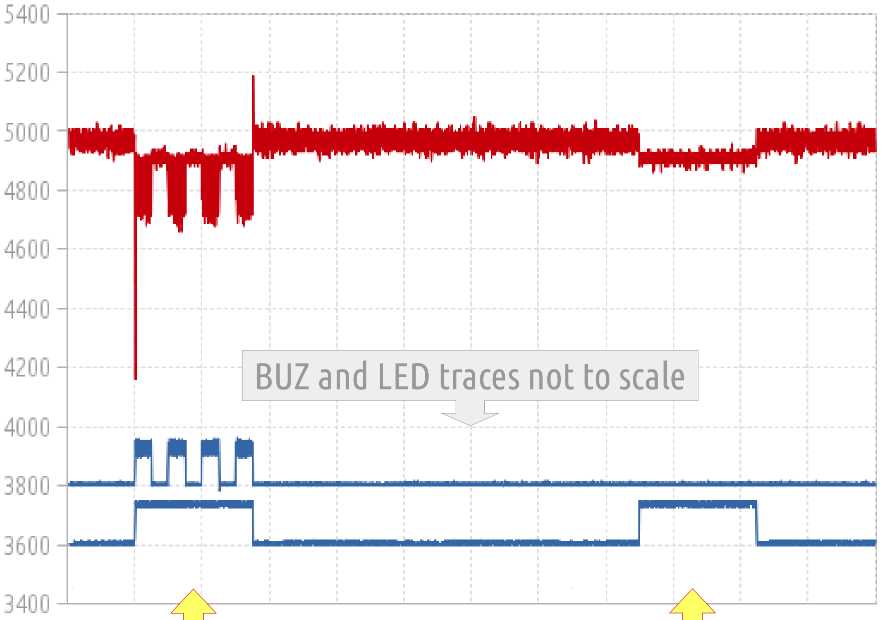
Example 3 - Software

```c
/* Generate n intervals of 50ms testing the
 * KEY bit at the beginning of each interval
 * and toggling the buzzer only if the key
 * is found pressed (PB2=0)
 */
void pulse50(unsigned char n) {
  while(n--) {
    if (PINB & (1<<KEY)) {      /* Read key */
      PORTB=PORTB & ~(1<<BUZ);  /* No sound */
    } else {
      PORTB=PORTB ^ (1<<BUZ);      /* Sound */
    }
    _delay_ms(SHORT);
  }
}
```

```c
void main(void) {

  hw_init();

  while(1) {
    /* Set LED */
    PORTB = PORTB | (1<<LED);

    /* Handle Buzzer (8 edges)*/
    pulse50(8);

    /* Clear LED and Buzzer */
    PORTB = PORTB & ~((1<<LED) | (1<<BUZ));
    _delay_ms(LONG);
  }
}
```

ATmega328p - IO Ports

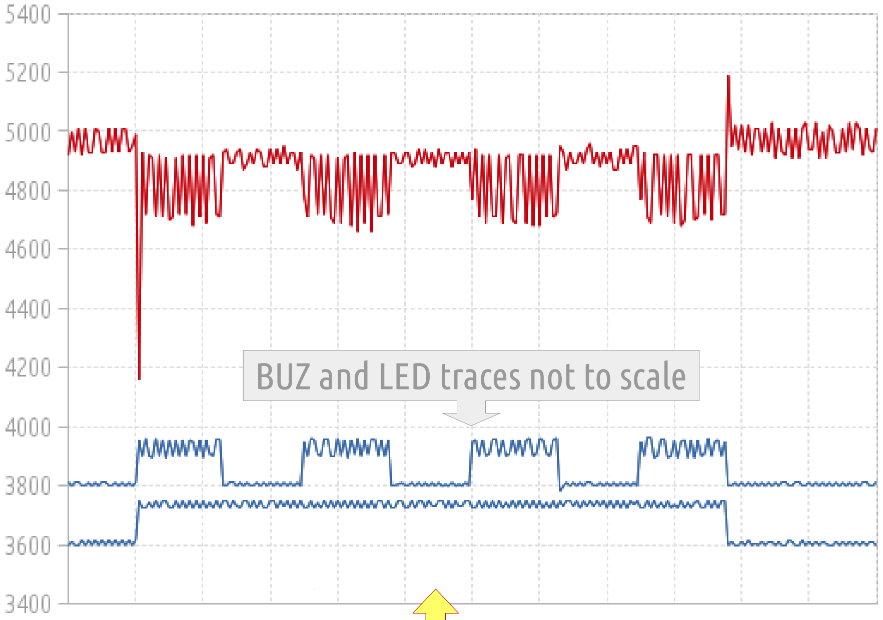
Example 3 - Power supply noise (mV)



BUZ and LED traces not to scale

KEY pressed

KEY not pressed



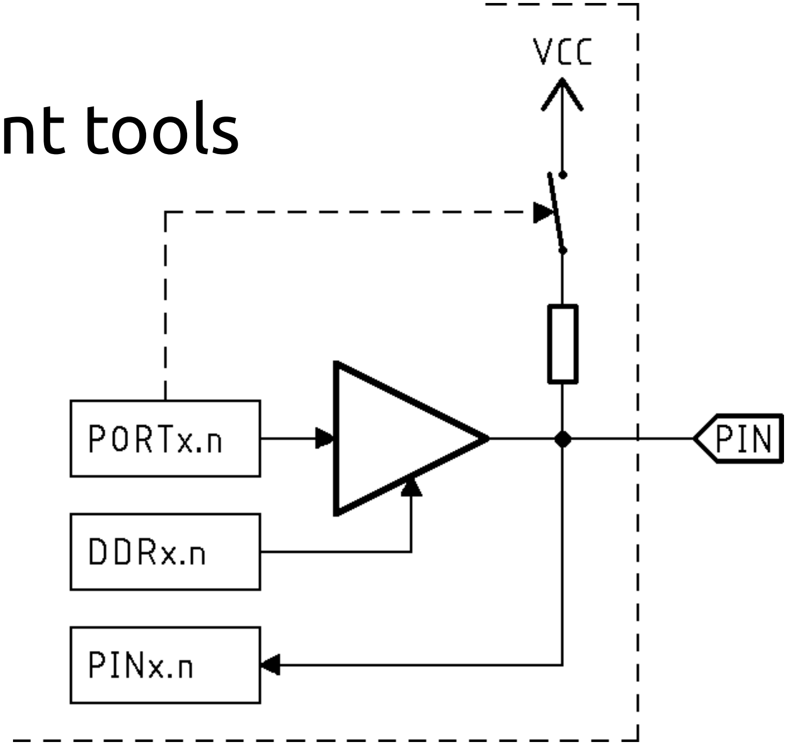
BUZ and LED traces not to scale

KEY pressed

To further explore...

- Chapter 18 of the datasheet (Moodle)
- Tutorial on I/O Ports
- Most important:
  - Install the development tools
  - Try the the examples and explore variants of your own
  - Quizz: is this correct?

```
(a) varX = PINB;
(b) varY = PORTB;
```


VCC
PORTx.n
DDRx.n
PINx.n
PIN