Utile PS1

## **1. Setup**

## **Datasheet ATmega328P**

* <http://www.atmel.com/Images/Atmel-8271-8-bit-AVR-Microcontroller-ATmega48A-48PA-88A-88PA-168A-168PA-328-328P_datasheet_Complete.pdf>

## **Atmel Studio 7**

<http://www.atmel.com/tools/atmelstudio.aspx?tab=overview>

## **Arduino Drivers**

<https://www.arduino.cc/en/Main/Software>

*Pentru Nano chinezesc va mai trebuie un driver pentru CH340g (*[*http://www.instructables.com/id/Arduino-Nano-CH340/*](http://www.instructables.com/id/Arduino-Nano-CH340/)*)*

## **Instalare external tool in Atmel Studio: (pentru a putea incarca programul pe placa dupa build)**

<http://www.discoveringelectronics.com/arduino-uno-programming-with-atmel-studio-6-1/>

Pentru Arduino Nano:

Command: C:\Program Files (x86)\Arduino/hardware/tools/avr/bin/avrdude.exe

Arguments -C"C:\Program Files (x86)\Arduino/hardware/tools/avr/etc/avrdude.conf" -v -patmega328p -carduino -PCOM4 -b57600 -D -Uflash:w:$(ProjectDir)Debug\$(TargetName).hex:i

## **2. Operatii pe biti**

**Reprezentare:**

PORTB = 1; // decimal   
PORTB = 0x01; // hex  
PORTB = 0b00000001; // binar

**Setare biti:**

Setarea unui bit pe 1 logic se face prin operatia **SAU** folosind operatorul **“ | ”.**

ex. Setare bitului 5 pe 1 logic in PORTB.

PORTB = 0; /\* 00000000 \*/

PORTB |= (1 << 5); /\* 00100000 \*/

**Resetare biti:**

Resetarea unui bit (aducerea lui pe 0 logic) se face prin operatia **SI** folosind operatorul **“&”**

ex. Resetare bitului 5 in PORTB.

PORTB **=** 0x20**;** */\** 00100000 *\*/*  
 PORTB **&=** **~(**1**<<**5**);** */\* 00000000 \*/*

**Comutare bit(toggle):**

Comutarea unui bit se face prin operatia **SAU EXCLUSIV** folosind operatorul **“^”**

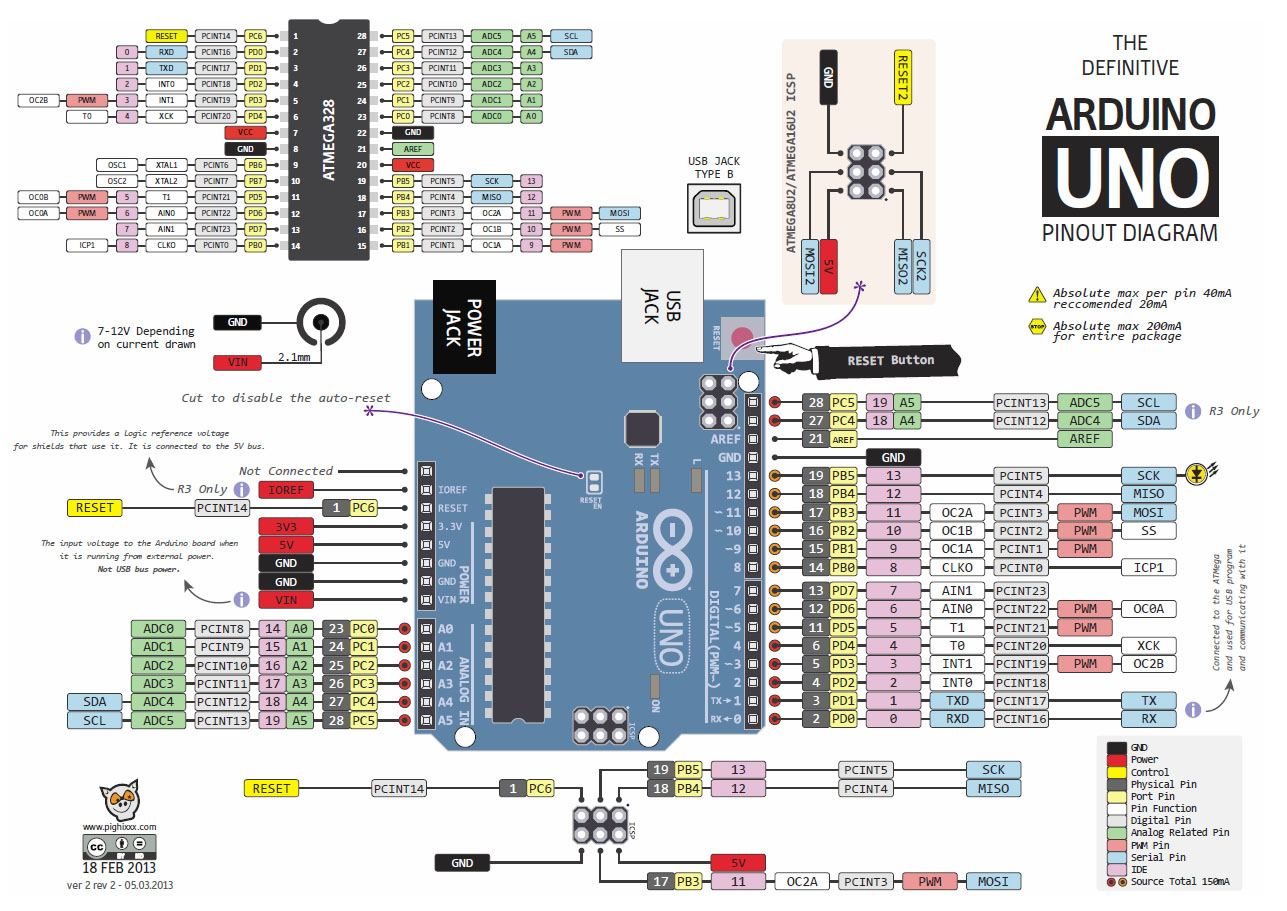
ex. Comutarea bitului 5 in PORTB.

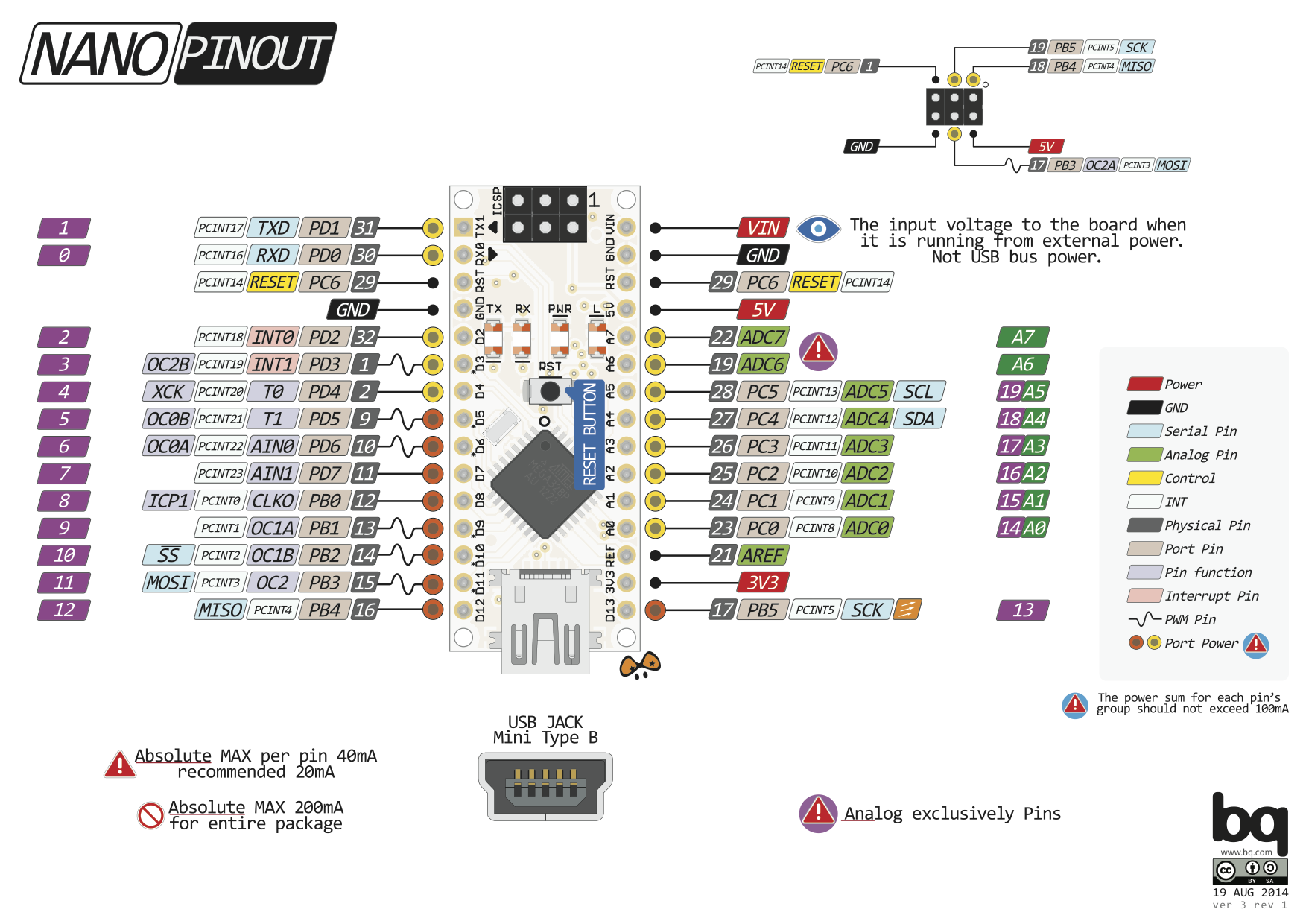
PORTB **=** 0x20**;** */\** 00100000 *\*/*

PORTB ^**=** 1<<5**;** */\** 00000000 *\*/*

PORTB ^**=** 1<<5**;** */\** 00100000 *\*/*

## 3. Pinout Arduino Uno/Nano





Each of the AVR Digital I/O ports is associated with three (3) I/O register. A Data Direction Register (**DDRx**), A Pin Register (**PINx**) and a Port Register (**PORTx**). Where **x** is the port **A, B, C, etc.** .

**DDRx - Port X Data Direction Register**

AVR Microcontroller DDR Register

DDRx is an 8-bit register which stores configuration information for the pins of Portx. Writing a **1** in the pin location in the DDRx makes the physical pin of that port an output pin and writing a **0** makes that pin an input pin.

**Note:** Each physical pin of a port is configured independently and thus a port can have some of its pins configured as input an the others as output pins.

**PINx - Port X Input Pins Register**

AVR Microcontroller Pin Register

PINx is an 8-bit register that stores the logic value, the current state, of the physical pins on Portx. So to read the values on the pins of Portx, you read the values that are in its PIN register.

**PORTx - Port X Data Register**

AVR Microcontoller Port Register

PORTx is an 8-bit register which stores the logic values that currently being outputted on the physical pins of Portx if the pins are configured as output pins. So to write values to a port, you write the values to the PORT register of that port.

sursa: http://www.avr-tutorials.com/digital/about-avr-8-bit-microcontrollers-digital-io-ports