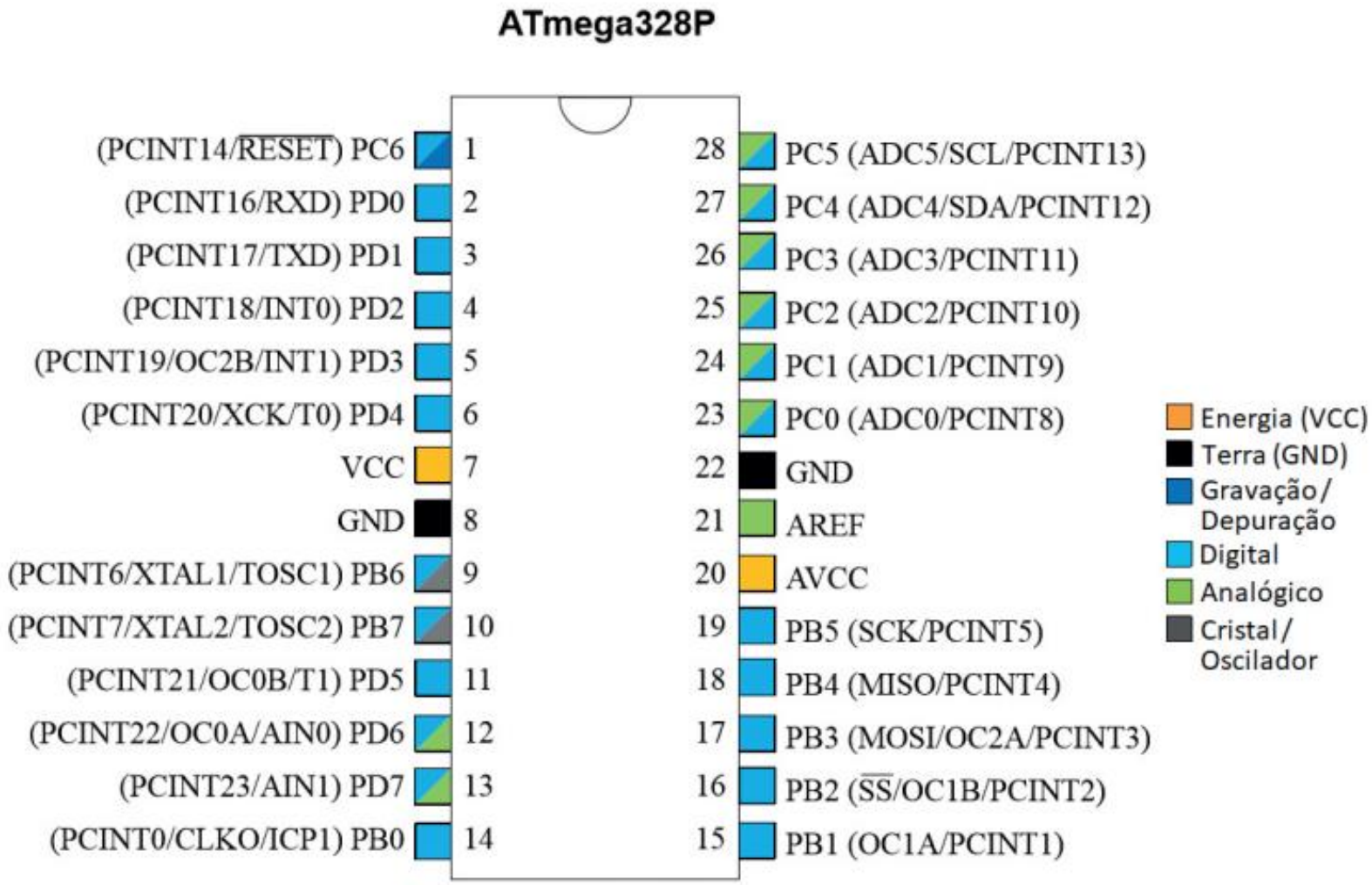
ATMEGA 328
ENTRADAS E SAÍDAS

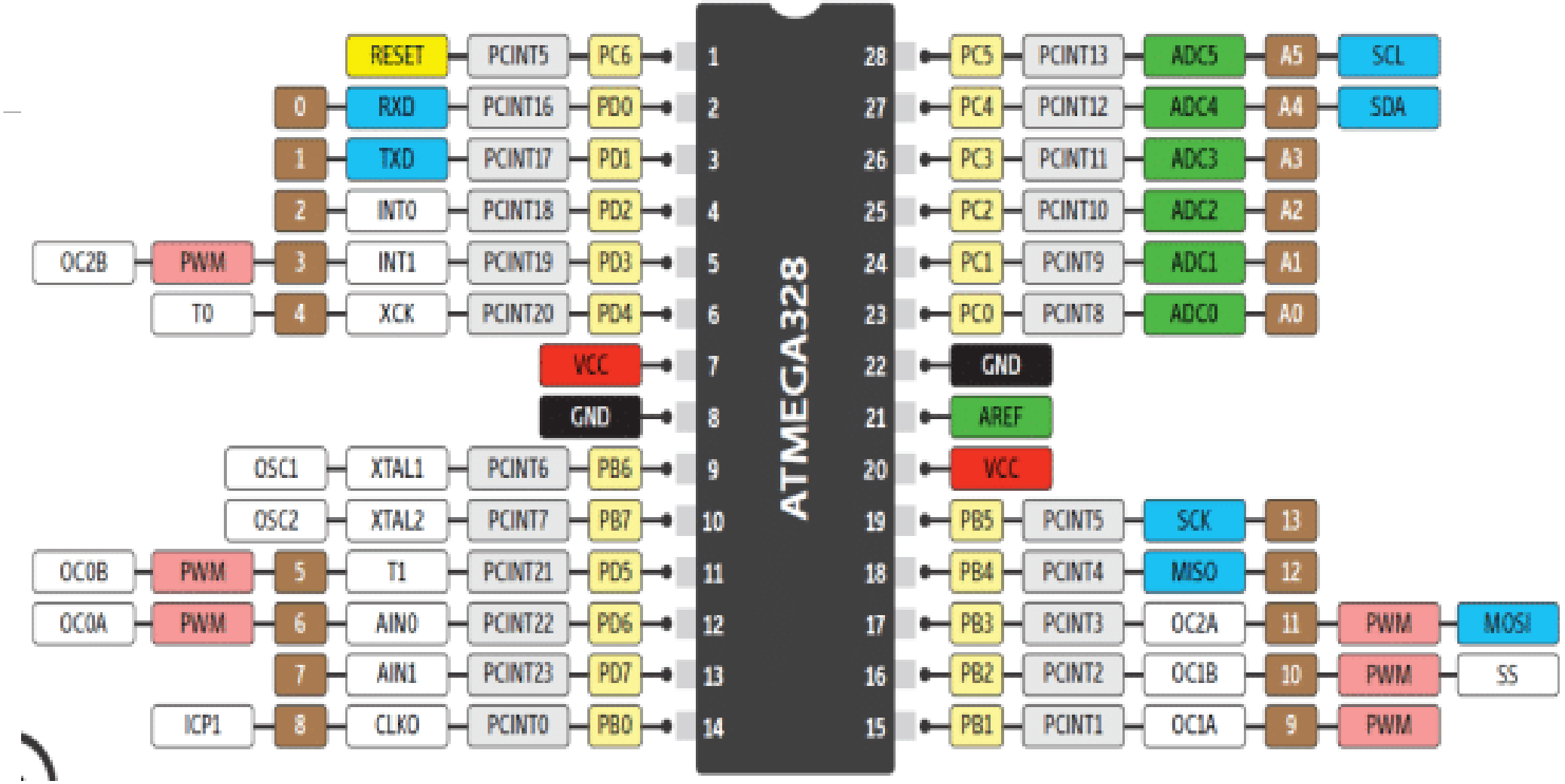
PROF. ME. HÉLIO ESPERIDIÃO

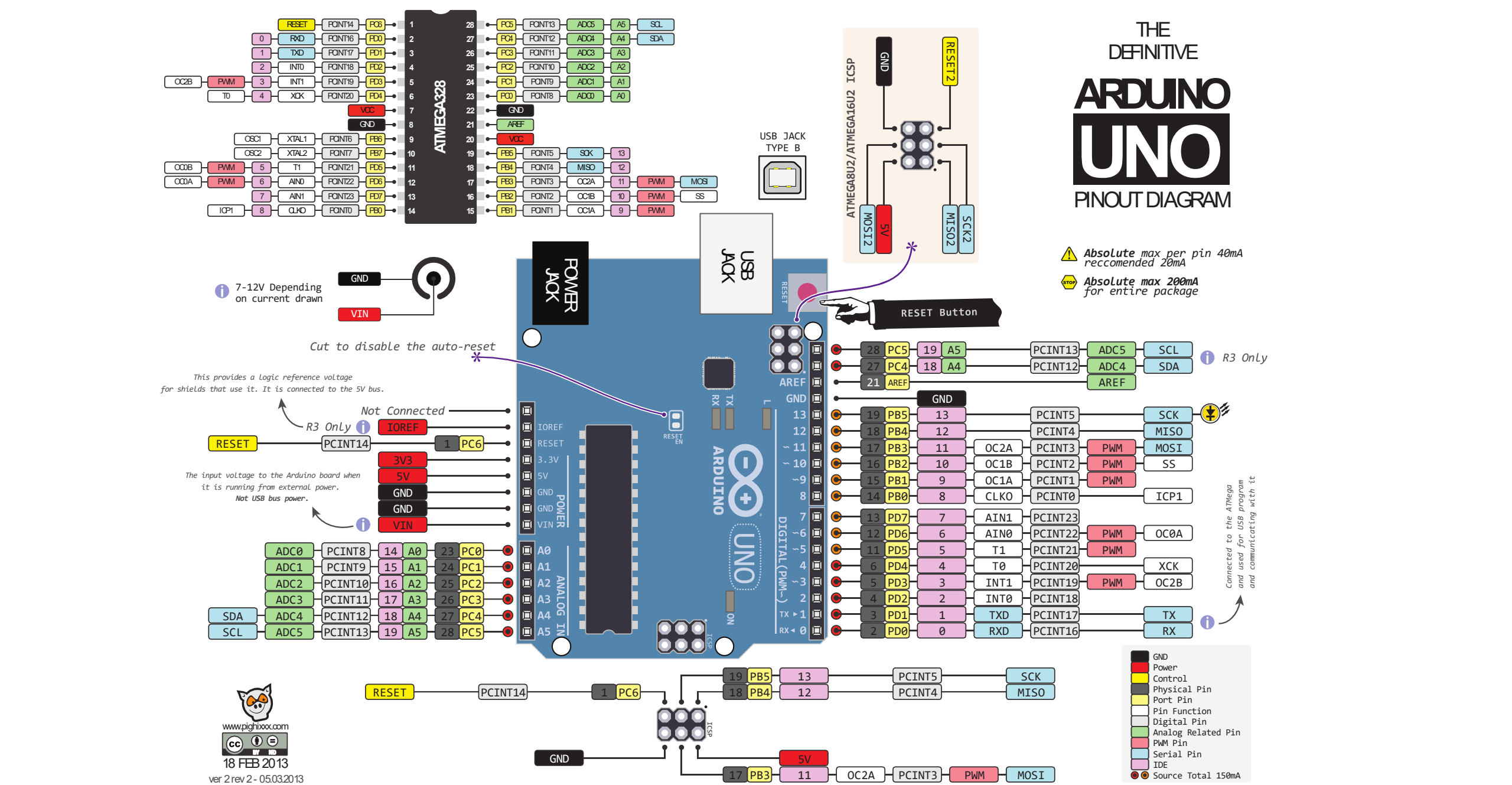
ATMega328

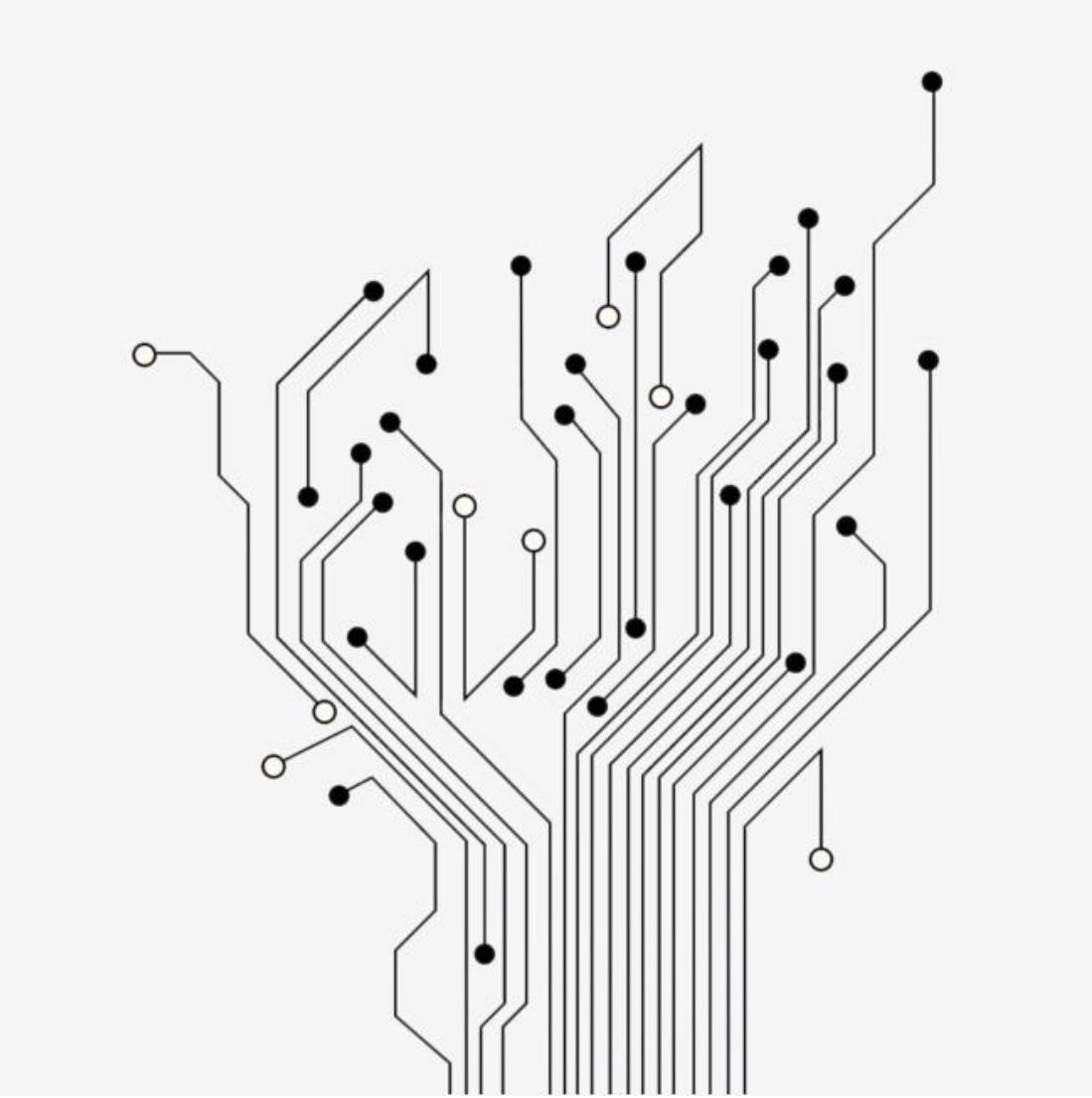
Os pinos dentro de um determinado PORT recebem um número sequencial, que se inicia com o valor “0”, e iremos nos referir a esse pino a com **letra de seu PORT, seguida de seu número.**

Exemplo: PD4 é o pino 4 do **PORTD**.


ATmega328P
(PCINT14/RESET) PC6 1
(PCINT16/RXD) PD0 2
(PCINT17/TXD) PD1 3
(PCINT18/INT0) PD2 4
(PCINT19/OC2B/INT1) PD3 5
(PCINT20/XCK/T0) PD4 6
VCC 7
GND 8
(PCINT6/XTAL1/TOSC1) PB6 9
(PCINT7/XTAL2/TOSC2) PB7 10
(PCINT21/OC0B/T1) PD5 11
(PCINT22/OC0A/AIN0) PD6 12
(PCINT23/AIN1) PD7 13
(PCINT0/CLKO/ICP1) PB0 14
28 PC5 (ADC5/SCL/PCINT13)
27 PC4 (ADC4/SDA/PCINT12)
26 PC3 (ADC3/PCINT11)
25 PC2 (ADC2/PCINT10)
24 PC1 (ADC1/PCINT9)
23 PC0 (ADC0/PCINT8)
22 GND
21 AREF
20 AVCC
19 PB5 (SCK/PCINT5)
18 PB4 (MISO/PCINT4)
17 PB3 (MOSI/OC2A/PCINT3)
16 PB2 (SS/OC1B/PCINT2)
15 PB1 (OC1A/PCINT1)
Energia (VCC)
Terra (GND)
Gravação/ Depuração
Digital
Analógico
Cristal/ Oscilador


ATMEGA328
RESET PCINT5 PC6 1
0 RXD PCINT16 PD0 2
1 TXD PCINT17 PD1 3
2 INT0 PCINT18 PD2 4
OC2B PWM 3 INT1 PCINT19 PD3 5
T0 4 XCK PCINT20 PD4 6
VCC 7
GND 8
OSC1 XTAL1 PCINT6 PB6 9
OSC2 XTAL2 PCINT7 PB7 10
OC0B PWM 5 T1 PCINT21 PD5 11
OC0A PWM 6 AIN0 PCINT22 PD6 12
7 AIN1 PCINT23 PD7 13
ICP1 8 CLKO PCINT0 PB0 14
28 PC5 PCINT13 ADC5 A5 SCL
27 PC4 PCINT12 ADC4 A4 SDA
26 PC3 PCINT11 ADC3 A3
25 PC2 PCINT10 ADC2 A2
24 PC1 PCINT9 ADC1 A1
23 PC0 PCINT8 ADC0 A0
22 GND
21 AREF
20 VCC
19 PB5 PCINT5 SCK 13
18 PB4 PCINT4 MISO 12
17 PB3 PCINT3 OC2A 11 PWM MOSI
16 PB2 PCINT2 OC1B 10 PWM SS
15 PB1 PCINT1 OC1A 9 PWM


THE DEFINITIVE
ARDUINO UNO
PINOUT DIAGRAM
ATMEGA328
ATMEGA8U2/ATMEGA16U2 ICSP
USB JACK TYPE B
POWER JACK
USB JACK
RESET Button
7-12V Depending on current drawn
Cut to disable the auto-reset
This provides a logic reference voltage for shields that use it. It is connected to the 5V bus.
Not Connected
R3 Only
The input voltage to the Arduino board when it is running from external power.
Not USB bus power.
Absolute max per pin 40mA reccomended 20mA
Absolute max 200mA for entire package
Connected to the ATMega and used for USB program and communicating with it
www.pighixxx.com
18 FEB 2013
ver 2 rev 2 - 05.03.2013
GND
Power
Control
Physical Pin
Port Pin
Pin Function
Digital Pin
Analog Related Pin
PWM Pin
Serial Pin
IDE
Source Total 150mA



Pinos I/O

Internamente, cada um dos pinos de I/O **apresenta um circuito digital responsável por configurar seu modo de operação,** ou seja, atuar como saída digital e controlar o pino entre os níveis de saída lógico zero (GND ou 0V) e um (VCC), ou atuar como entrada digital de alta impedância e realizar leitura dos mesmos níveis lógicos.

Registradores de controle

Para o controle e configuração de seus pinos, cada PORT implementa três registradores distintos.

Cada registrador é do tamanho de 8 bits (1 byte),

- Cada bit possui influência direta sobre o pino da mesma ordem, ou seja, o bit 0 atua sobre o pino 0, o bit 1 sobre o pino 1 e assim sucessivamente até o bit 7.

1º Registrador DDRB, DDRC e DDRD:

DDRB, DDRC e DDRD:

Responsáveis por configurar a **direção dos pinos em seu respectivo PORT**.

Ao configurar um bit desse registrador como 0, o respectivo pino atuará como entrada, e caso configurado como 1, o pino atuará como saída.

Exemplo: em **DDRD** = 0b**1111**0000 (em binário), os pinos PD0, PD1, PD02 e PD3 foram configurados como entrada, e os pinos PD4, PD5, PD6 e PD7 foram configurados como saída.

2º Registrador
PINB, PINC e PIND

Responsáveis por realizar a leitura digital dos pinos, quando os mesmos são configurados como entrada pelo registrador DDR.

Cada bit desse registrador representará o valor lógico do pino externo respectivo.

Exemplo: PINB em 1000001 (binário), os pinos PB0 e PB7 apresentam o nível lógico 1 (VCC) em seus terminais, e os demais pinos, o nível lógico 0 (GND) em seus terminais

3º Registrador
PORTB, PORTC e PORTD

Responsáveis por realizar o acionamento digital dos pinos, **quando** os mesmos são **configurados como saída pelo registrador DDR**.

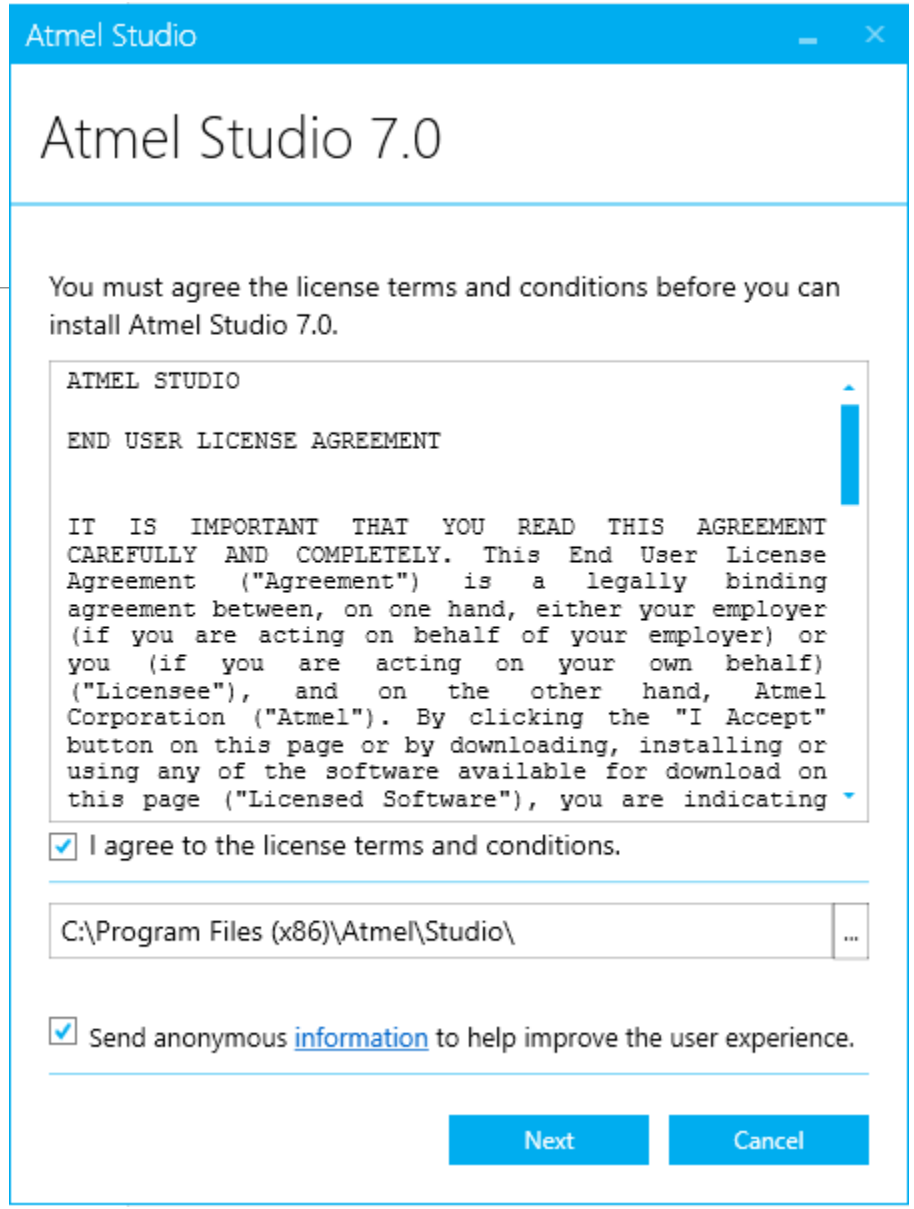
Cada bit desse registrador será responsável por gerar um nível lógico no respectivo pino.

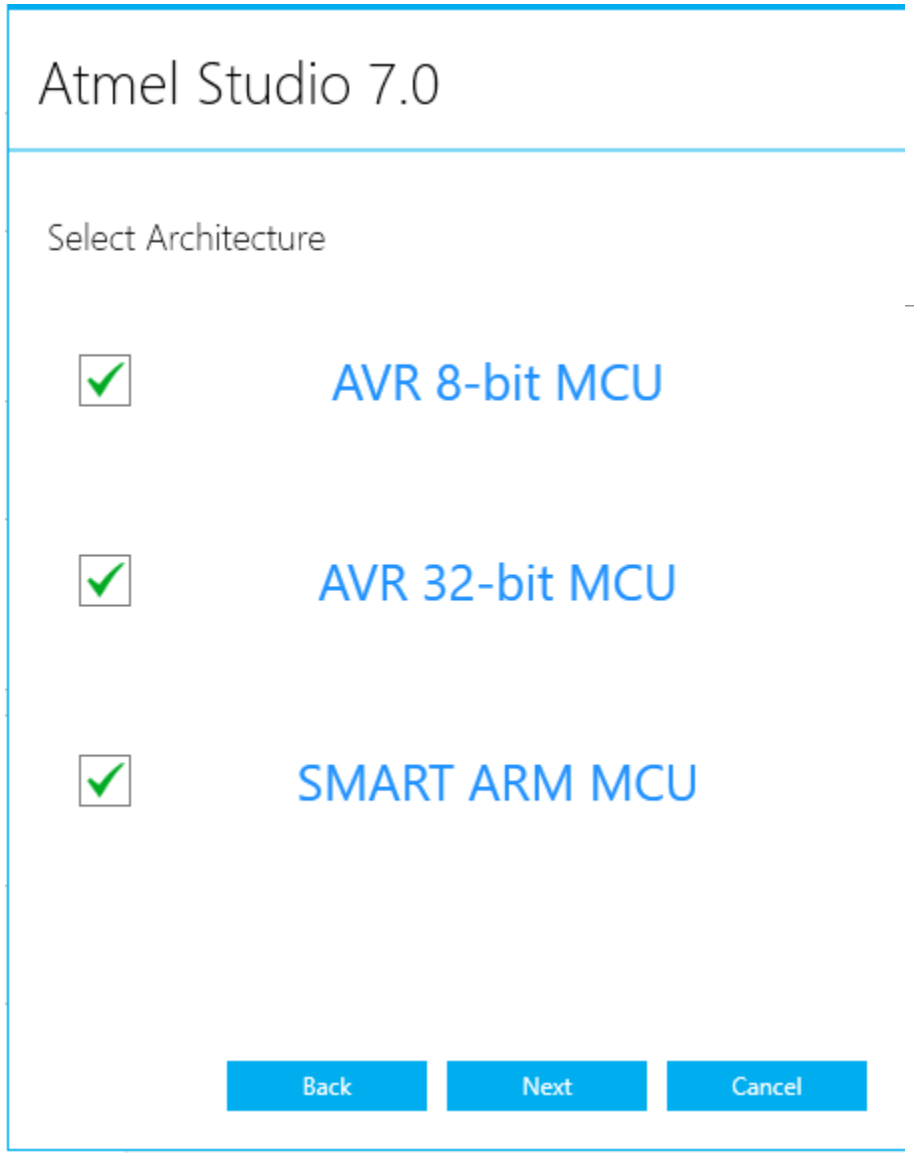
Exemplo: em PORTC = 0b11100000, os pinos PC0 a PC4 estarão com seus terminais em nível lógico 0 (GND), e os pinos PC5 a PC7 estarão em nível lógico 1 (VCC).

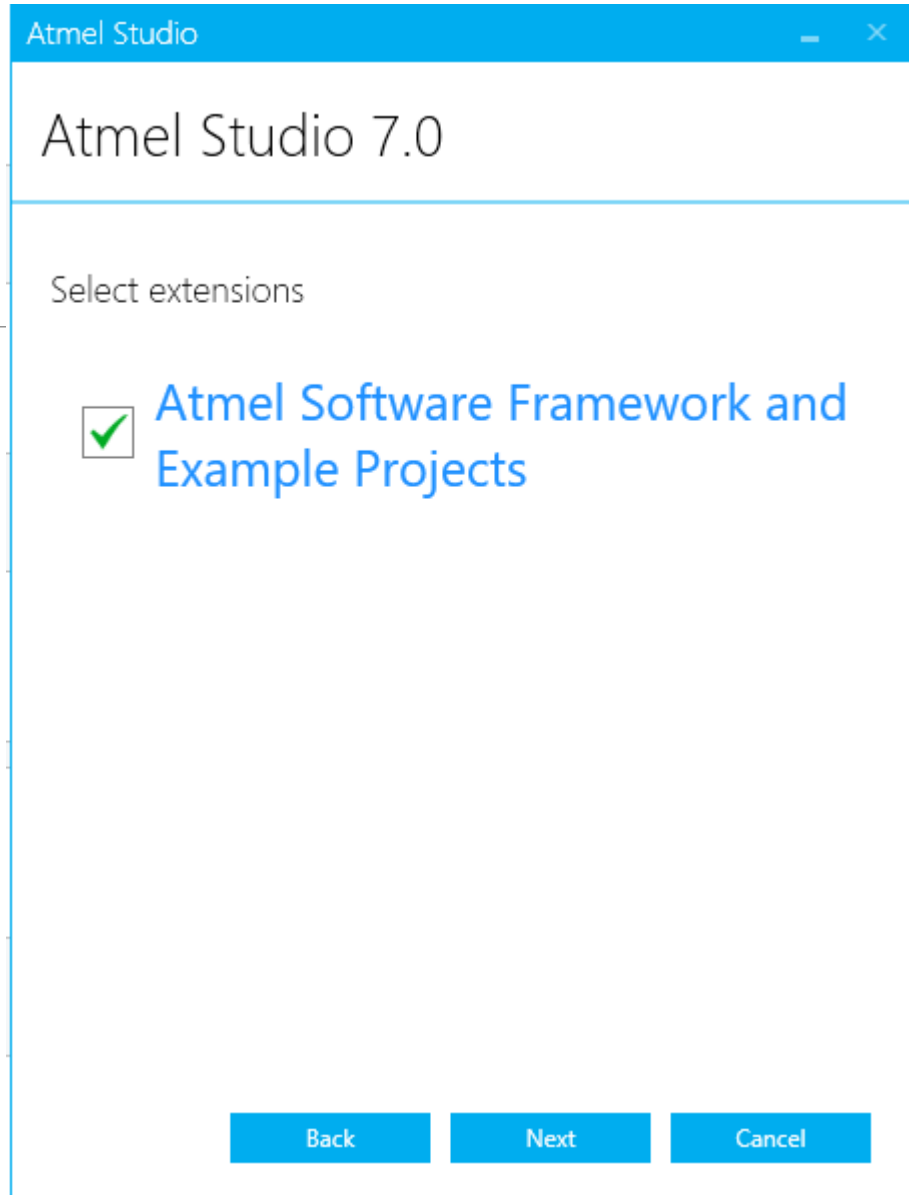
Atmel Studio 7.0

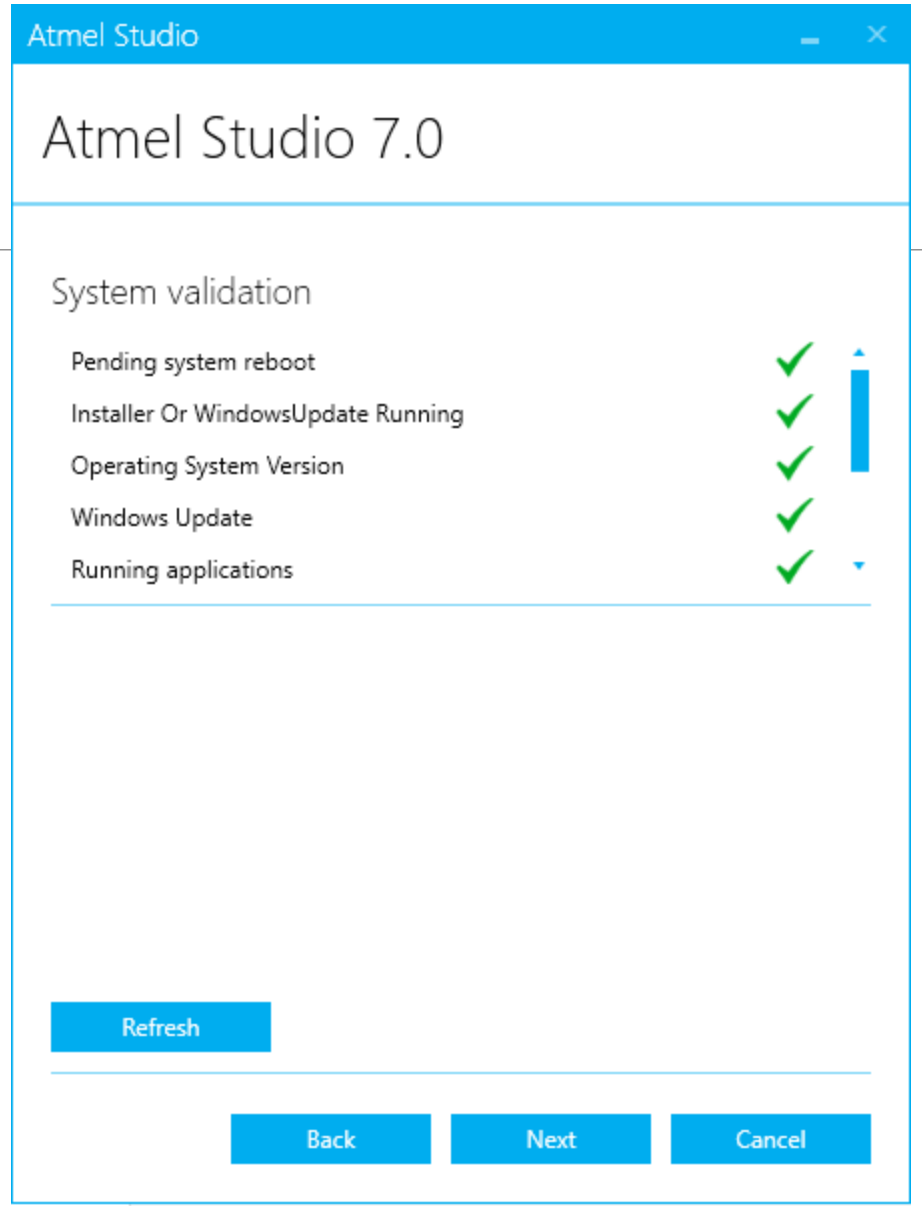
é a plataforma de desenvolvimento disponibilizada gratuitamente pela Microchip para desenvolver projetos com seus microcontroladores. Suporta tanto os Atmel AVR quanto os Atmel ARM Cortex.

http://www.microchip.com/mplab/avr-support/atmel-studio-7




Atmel Studio 7.0
Select Architecture
AVR 8-bit MCU
AVR 32-bit MCU
SMART ARM MCU
Back
Next
Cancel


Atmel Studio
Atmel Studio 7.0
Select extensions
Atmel Software Framework and Example Projects
Back
Next
Cancel


Atmel Studio
Atmel Studio 7.0
System validation
Pending system reboot
Installer Or WindowsUpdate Running
Operating System Version
Windows Update
Running applications
Refresh
Back
Next
Cancel

O AVRDUDE

O AVRDUDE é um programador em linha de comando muito popular para linha de microcontroladores Atmel AVR. É um projeto open-source e também é utilizado para upload dos programas na plataforma Arduino.

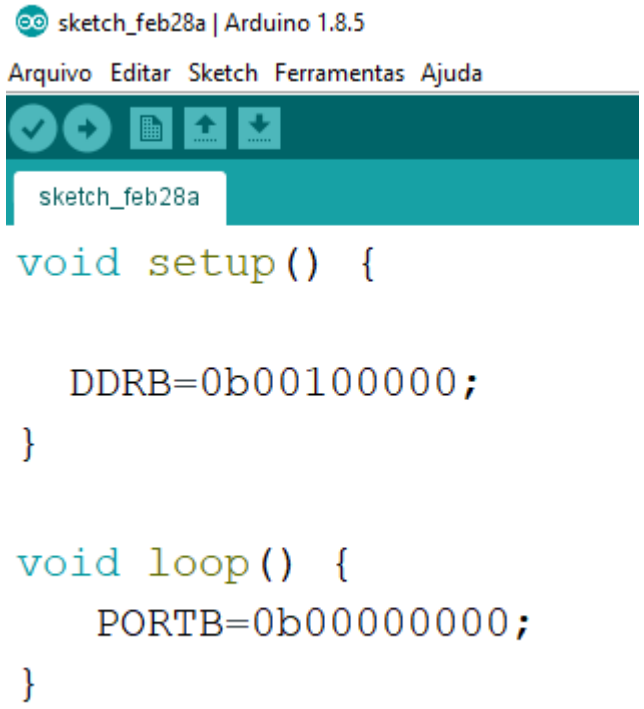
download

```
#include <avr/io.h>          //biblioteca para acesso aos registradores do uC
#include "util/delay.h"      //biblioteca para funções de delay

int main(void){
 DDRB = 0b00100000;    //configura pino PB5 como saída e demais como entradas
 while(1){
   PORTB= 0b00000000; // todas as portas são definidas como baixo
   _delay_ms(250);
   PORTB= 0b00100000;  // muda o estado do PB5 para alto
   _delay_ms(250);
 }
}
```

Configuração utilizando binários

Para IDE do Arduino



```c
#include <avr/io.h>          //biblioteca para acesso aos registradores do uC
#include "util/delay.h"      //biblioteca para funções de delay

int main(void){
 DDRB = 0b00100000;    //configura pino PB5 como saída e demais como entradas
 while(1){
    PORTB=(0<<PB5);  // muda o estado do PB5 para baixo
    _delay_ms(250);
    PORTB=(1<<PB5);  // muda o estado do PB5 para alto
    _delay_ms(250);
 }
}
```

```
#include <avr/io.h>          //biblioteca para acesso aos registradores do uC
#include "util/delay.h"      //biblioteca para funções de delay

int main(void){
 byte l=32; //0010 0000
 byte d=0;  //0000 0000
 DDRB = l;   //configura pino PB5 como saída e demais como entradas
 while(1){
   PORTB=(d);  // muda o estado do PB5 para baixo
   _delay_ms(250);
   PORTB=(l);  // muda o estado do PB5 para alto
   _delay_ms(250);

 }
}
```

```
#include <avr/io.h>          //biblioteca para acesso aos registradores do uC
#include "util/delay.h"      //biblioteca para funções de delay

int main(void){
 byte b=32;//0010000
 byte c=32;
 DDRB = c;   //configura pino PB5 como saída e demais como entradas
 while(1){
   PORTB^=(b);  // muda todos os estados de b
   _delay_ms(250);
   PORTB=(b);  // muda o estado do PB5 para alto
   _delay_ms(250);

 }
}
```

```
#include <avr/io.h>          //biblioteca para acesso aos registradores do uC
#include "util/delay.h"      //biblioteca para funções de delay

 byte sequencia[] = {0, 1, 2, 4, 8, 16, 32, 64, 128};
int main(void){
 DDRB = 0b00100000;    //configura pino PB5 como saída e demais como entradas
 int qtd=0;
 while(1){
  if(qtd>7){
    qtd=0;
  }
   PORTB=(sequencia[qtd]);  // muda todos os estados de b
   _delay_ms(1000);
   qtd++;

 }
}
```