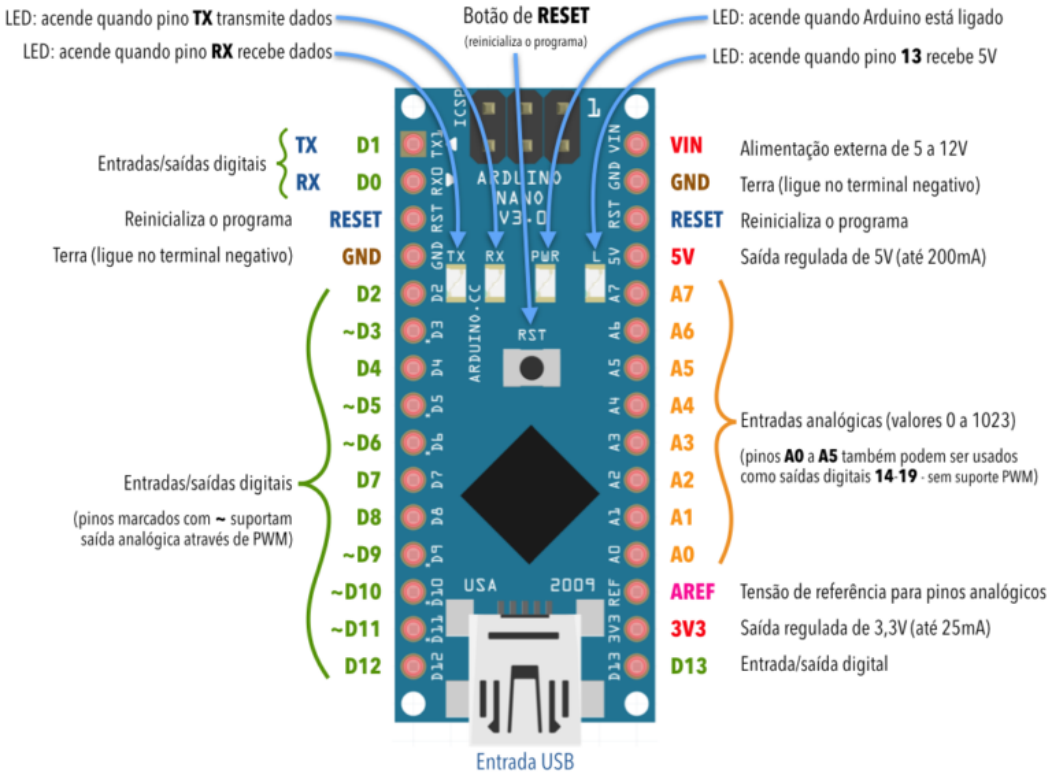
ELETRÔNICA PARA ARTISTAS
Coisas criativas para fazer com elétrons

JUNE 11, 2017 BY ETACARINAE

Arduino 2: Configuração do Arduino Nano

O Arduino que usaremos na oficina é um **Arduino Nano**. Tem dimensões de 43 x 15 mm. Ele possui uma entrada USB que permite a ligação direta a um computador (não precisa de adaptador), e que também fornece alimentação de 5V enquanto estiver conectado. Depois de programado e desconectado do computador, ele pode ser alimentado de forma independente por **7 a 12V** aplicados nos pinos **VIN** (ligado ao positivo da bateria ou fonte) e **GND** (ligado ao negativo).

O Arduino Nano também possui saídas de tensão reguladas em **3,3 V** (Pino 3V3) e **5V** (Pino 5V). Os pinos A0 a A7 são de **entrada analógica** (recebem valores *entre* 0 e 5V), e D0 a D13 suportam **entrada digital** (reconhecem dois valores: 0V – nível lógico BAIXO ou 5V – nível lógico ALTO). A **saída analógica** é *simulada* via PWM apenas através dos pinos digitais D3, D5, D6, D9, D10 e D11. Os outros pinos digitais, e também os pinos A0 a A5, podem operar como **saída digital**. O diagrama abaixo ilustra a pinagem do Arduino Nano:



As especificações de corrente e tensão referem-se ao clone chinês CH340 do Arduino Nano que está incluído no kit, e não ao Arduino Nano original italiano (que são um pouco diferentes).

Algumas observações e cuidados importantes:

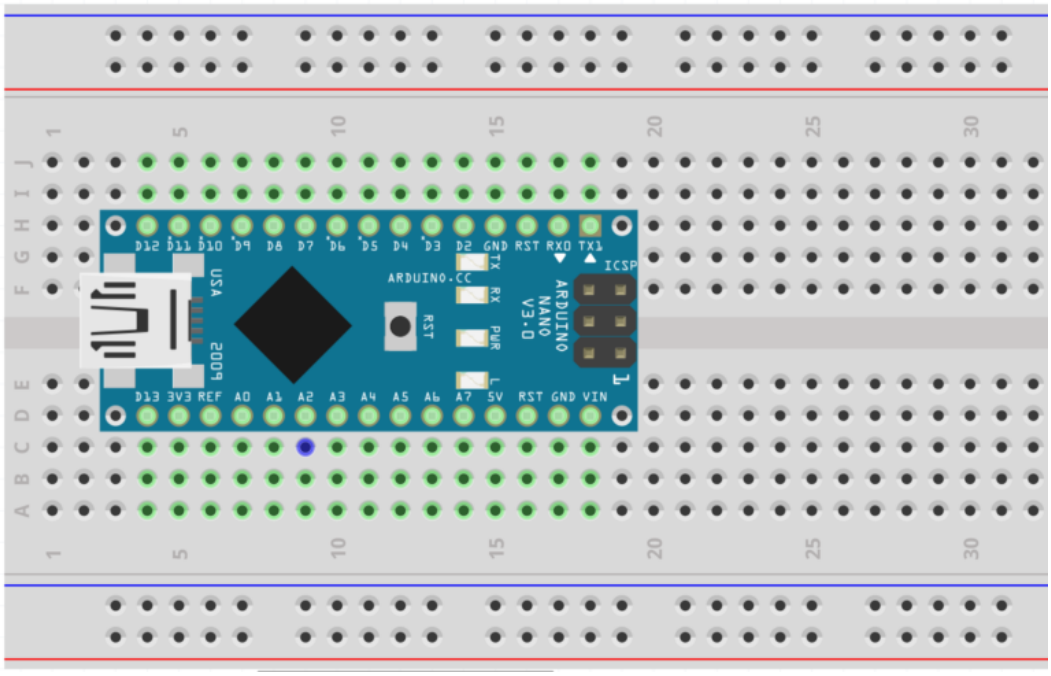
- **Os pinos** do Arduino suportam **no máximo 40mA** (ligar em um circuito que deixa passar mais corrente pode queimar o pino). É necessário calcular resistores para limitar a corrente.
- **O Arduino inteiro fornece no máximo 200mA**. Mas é possível controlar circuitos que consomem bem mais corrente, desde os sinais enviados e recebidos pelos pinos sejam intermediados por circuitos que reduzam as correntes e tensões a níveis suportados. Isto pode ser feito com resistores, capacitores, transistores, relés e outros dispositivos.
- Também é necessário ter cuidado para não curto-circuitar as saídas (**5V** ou **3V3** ligadas diretamente em **GND**). Os pinos analógicos e digitais podem ser ligados diretamente em 5V ou 0V somente **se forem usados como entradas**. Esses valores são tratados como informação (nível lógico ALTO e BAIXO) pelo Arduino. Para usá-los como **saídas**, é necessário configurar essa funcionalidade na programação, e ter o mesmo cuidado que as saídas 5V e 3V3 (não ligar diretamente em GND), além de usar resistores para manter o fluxo de corrente dentro do limite.
- O pino AREF é usado para ajustar a tensão de referência usada para os pinos analógicos. Ela está internamente conectada ao pino 5V, mas pode ser desligada via programação. Ligar uma tensão qualquer neste pino sem primeiro fazer essa alteração via código irá queimar o regulador de tensão (e provavelmente a entrada USB).

Um programa escrito para um tipo de Arduino pode ser usado em outro tipo de Arduino. Pode-se aproveitar programas prontos e fazer pequenas adaptações sem que seja necessário entender todo o código. Portanto, sabendo o mínimo da programação do Arduino, você pode baixar programas da Internet e adaptar para seus circuitos. É preciso garantir que os números de pinos, **declarados no código dos programas**, e os pinos reais, **usados no circuito** estejam de acordo. Em geral qualquer pino digital ou analógico pode ser usado. Eles podem até ser reprogramados. Alguns pinos têm capacidades especiais. Por exemplo, os pinos digitais 3, 5, 6, 9, 10 e 11, no Arduino Nano, permitem gerar saída analógica usando PWM.

Não se preocupe se você não entendeu tudo. São muitos conceitos e é sempre mais fácil entender com um ou mais exemplos. Nas próximas seções mostraremos como instalar e configurar o Arduino, e depois como usá-lo através de vários experimentos. Depois que você fizer os experimentos, releia esta seção. Vários conceitos irão ficar mais claros.

Preparação e teste do Arduino

Como vamos construir circuitos, e o Arduino Nano não possui soquetes onde podemos inserir terminais de componentes, precisamos usar o protoboard. Encaixe o Arduino com cuidado ocupando a parte central, de forma que possamos ter acesso a todos os seus pinos através dos pinos laterais. Da forma mostrada abaixo, cada pino terá dois a três furos.



O protoboard deve estar livre de outros circuitos (principalmente, não deve haver nenhuma fonte de energia conectado a ele).

Depois de encaixado o Arduino, encaixe uma das pontas do cabo USB no Arduino, e a outra em alguma saída USB do seu computador. O LED PWR do Arduino deverá acender, indicando que ele está sendo alimentado pela porta USB. Para haver comunicação, no entanto, é preciso instalar o driver.

Instalação do ambiente de desenvolvimento

Para habilitar um computador para programar o Arduino Nano do kit são necessárias duas etapas:

1. Instalar o **driver** (programa que permite a comunicação com o Arduino via porta USB do computador) **do adaptador USB-Serial** (embutido no Arduino).
2. Instalar o programa com o **ambiente de programação** (Arduino IDE).

A **IDE** (aplicação com ambiente gráfico para programação) é distribuída pelo site oficial do Arduino (**arduino.cc**) e existe para Mac, Windows e Linux. Roda de maneira praticamente idêntica nas três plataformas.

O **driver** é mais complicado de instalar, e pode variar dependendo do Arduino usado, se é um clone ou se é um autêntico italiano. O Arduino original (italiano) não requer a instalação de drivers no Mac, mas a instalação ainda pode ser necessária em algumas versões de Windows.

Instalação do driver

O Arduino Nano incluído no kit é um clone e usa um **adaptador USB-Serial chinês** (chip CH341). Para que ele seja reconhecido pelo computador, seja Mac, PC ou Linux, ele **precisa** ter o **driver** instalado antes. O driver é um programa de instalação que deve ser baixado do site do fabricante e executado. Ele não faz nada além disso. A instalação termina depois que o computador for reiniciado. Veja as instruções abaixo. Elas podem ser diferentes dependendo do sistema que você estiver usando.

Windows

Se você usa **Windows 10** baixe o arquivo EXE disponível em

http://www.wch.cn/download/CH341SER_EXE.html

(clique no botão de Download), execute-o. Você deve ter as permissões para executar este programa no computador, pois ele vai gravar arquivos do sistema. Siga o passo-a-passo (em inglês). Depois é necessário reiniciar o computador para completar a instalação. Quando terminar e reiniciar, pule para a seção seguinte (IDE) para instalar o ambiente de programação.

Mac

Se usa **Mac OS Sierra (10.12)**, baixe o arquivo ZIP em

http://www.wch.cn/download/CH341SER_MAC_ZIP.html

(clique no botão de Download) e abra o ZIP. Dentro dele há um arquivo **CH34x_Install_V1.4.pkg**. Execute esse arquivo e siga as instruções (em inglês). Depois é necessário reiniciar o computador para completar a instalação. Quando reiniciar, pule para a seção seguinte (IDE) para instalar o ambiente de programação.

Linux

E se você usa Linux baixe o arquivo localizado em

http://www.wch.cn/download/CH341SER_LINUX_ZIP.html

(clique no botão de Download). Abra o ZIP em uma pasta. Abra uma janela do terminal e execute as linhas abaixo:

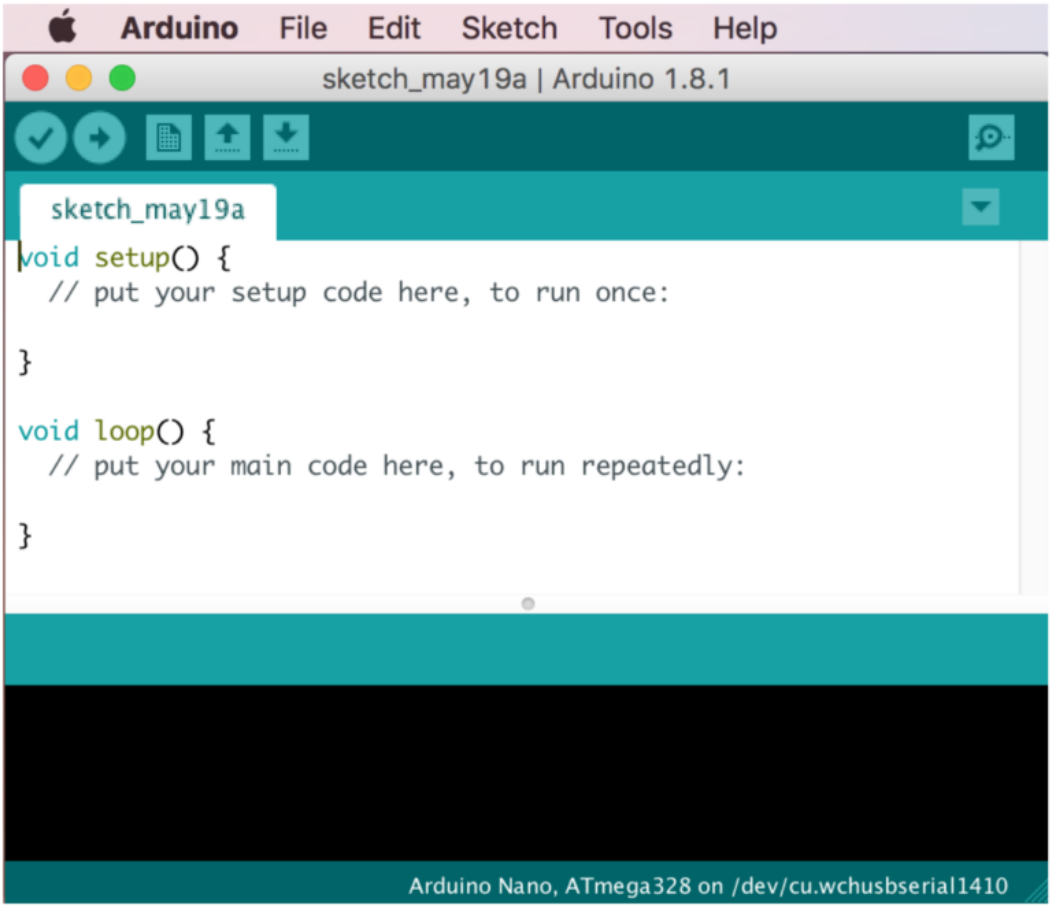
```
sudo make
sudo make load
```

Instalação do ambiente de programação (IDE)

A programação do Arduino é feito na linguagem *Processing*, que é baseada na linguagem C e similar a linguagens de programação populares como C# e Java. Embora possam ser escritos programas bastante complexos usando essa linguagem, é possível fazer muita coisa escrevendo programas bem simples e fáceis de entender mesmo para quem é leigo em programação. Aprendendo o mínimo, você conseguirá baixar programas disponíveis na Internet e adaptar para rodar com seus circuitos. Para isto, precisamos instalar o ambiente de desenvolvimento integrado (*IDE – Integrated Development Environment*) do Arduino. Baixe o programa de instalação para o seu sistema operacional (Windows, Mac ou Linux) na página

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

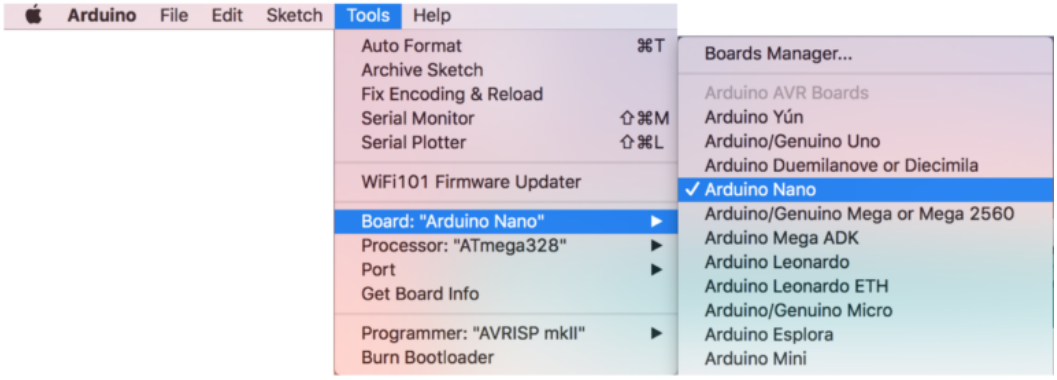
Execute o instalador e siga o passo-a-passo. Depois rode o programa. Ele deverá abrir a janela abaixo:



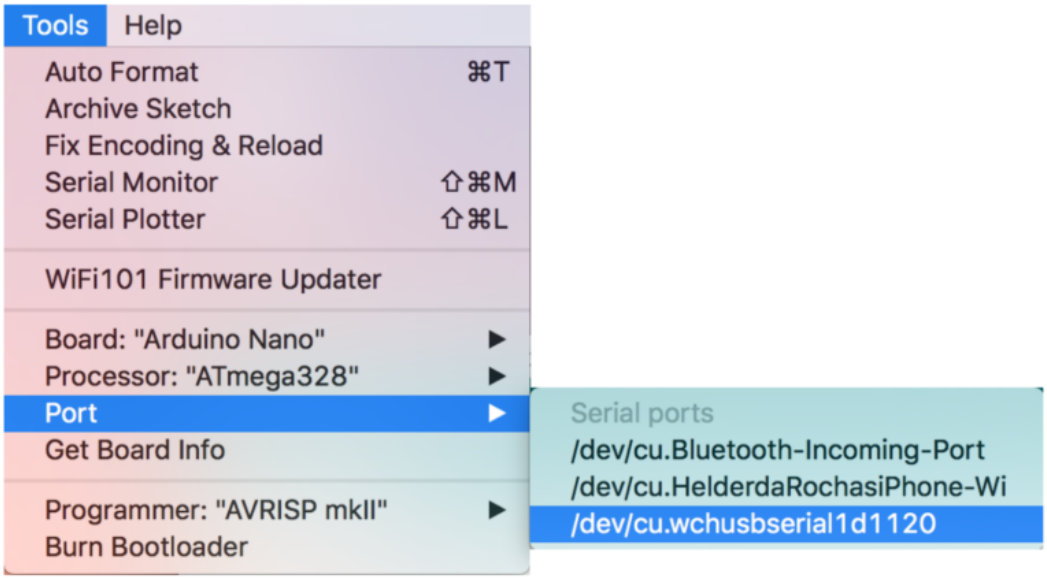
Comunicação do Arduino com o computador

Depois de instalados o driver e o IDE, é preciso ainda **selecionar a placa usada** e a **identificar a porta de comunicação** onde ela está conectada. Isto só precisa ser feito uma vez para cada placa diferente que você usar, mas requer que o Arduino esteja conectado. Portanto, se você ainda não conectou o Arduino a uma porta USB do seu computador, faça isto agora.

Selecione no menu **Ferramentas** (*Tools*), na opção **Placa** (*Board*). Na lista há várias placas. Selecione **Arduino Nano**.



Depois selecione a porta de comunicação. No Windows deve ser algo como **COM4**. No Linux e Mac, um caminho que inicia com **/dev/cu.wchusbserial** (ex: /dev/cu.wchusbserial123456).



Se você usar outro tipo de Arduino posteriormente, terá que modificar esses parâmetros ou a transferência do programa não será possível (o programa apresentará

mensagens de erro informando isto). Alguns clones de Arduino são identificados diferentemente (ex: alguns clones chineses de LilyPad são identificados como Arduino Uno) e outros requerem a instalação de bibliotecas externas (que podem ser baixadas) para funcionar.

No próximo post iniciaremos a programação do Arduino e faremos um primeiro circuito.