

Arduino Standalone: I

Abrantes Araújo Silva Filho

2025-11-18

Resumo

Este é o primeiro de uma série de artigos que discute e mostra, em detalhes, como montar um Arduino Standalone, ou seja, um Arduino montado apenas em uma protoboard. Neste artigo aprenderemos a fazer a montagem da fonte de alimentação, do microcontrolador e de outros componentes fundamentais, como o cristal oscilador. Em outros artigos desta série discutiremos as opções disponíveis para a programação do microcontrolador diretamente na protoboard.

Sumário

1	Introdução	2
2	Materiais	4

1 Introdução

Depois de algum tempo utilizando o Arduino e ganhando experiência em diversos projetos interessantes com sensores e atuadores, você começa a sentir necessidade de entender realmente como essa placa funciona e como o microcontrolador realiza a mágica de transformar o código de seu programa em ações no mundo real.

Montar um “Arduino Standalone” (usar apenas o microcontrolador — geralmente o ATmega328P — na protoboard sem a placa do Arduino) é um verdadeiro **rito de passagem** no aprendizado de eletrônica, programação e sistemas embarcados: é o momento em que você deixa de ser apenas um usuário do Arduino para entender como a engenharia por trás dele funciona.

Pense no Arduino como uma bicicleta com rodinhas de treinamento: você consegue fazer muitas coisas legais e interessantes mas chega um momento em que, para avançar, as rodinhas de treinamento precisam ser removidas. Montar seu primeiro Arduino Standalone equivale à seu primeiro passeio de bicicleta sem as rodinhas de treinamento.

Existem diversas razões pelas quais isso é interessante e importante:

- **Desmistificação do hardware:** quando usamos o Arduino, muitas coisas estão pré-configuradas e ocultas para nós. Ao montar o circuito independente na protoboard você aprenderá na prática a função de diversos componentes essenciais como o cristal oscilador, a regulação de tensão para a alimentação do microcontrolador e a montagem do circuito de reset. Além disso você entenderá de verdade o que o microcontrolador precisa obrigatoriamente para funcionar (alimentação estável, clock, reset em nível definido, capacitores de desacoplamento, etc.) e o que é opcional e fornecido pelo Arduino apenas como um “conforto” para o usuário (conversor USB-Serial, LEDs, etc.).
- **Eficiência energética:** a placa do Arduino possui componentes que constantemente consomem energia (LEDs, conversor USB-Serial) mesmo que seu código não esteja fazendo nada. Ao eliminar todos os componentes não essenciais e configurar corretamente o microcontrolador (usando o modo de *sleep*) você pode fazer seu Arduino Standalone funcionar por vários meses usando apenas duas pilhas AA (um Arduino completo acabaria com as pilhas em alguns dias).
- **Custo e permanência:** imagine que você criou um pequeno sistema de automação para o portão da garagem de sua casa. Você deixaria seu Arduino preso lá no portão para sempre? Correndo o risco de um fio se desconectar com o tempo? Você vai perder uma placa relativamente cara e que tem diversas outras possibilidades de usos além do portão? Provavelmente não. O melhor é que você utilize apenas o microcontrolador e os componentes básicos para fazer seu sistema de automação funcionar e, ao montar o Arduino Standalone, você aprenderá a como fazer isso.

- **Transição para o produto final:** nenhum produto final acabado tem um Arduino colado dentro dele. Produtos reais utilizam apenas o microcontrolador e os demais componentes eletrônicos soldados em uma placa de circuito impressa (PCB¹) fabricada profissionalmente. Aprender a montar um Arduino Standalone é um passo intermediário obrigatório para que, no futuro, você aprenda a projetar suas próprias PCB profissionais. Você entenderá na prática a diferença entre uma placa de desenvolvimento e a versão de produção final.
- **Flexibilidade:** ao usar o microcontrolador independente você não fica preso aos 5,0 V e 16 MHz do Arduino: você pode rodar o chip em 3,3 V e se comunicar com sensores modernos sem conversores de nível, ou pode usar o clock interno de 8 MHz do microcontrolador e liberar mais dois pinos digitais extras para uso.
- **Entendimento do datasheet:** para entender certas funções mais avançadas do microcontrolador você será forçado a ler o datasheet e procurar coisas como os tipos de clock possíveis, configuração de fuses e a organização dos pinos.
- **Liberdade de forma e conectores:** ao usar o Arduino você é obrigado a utilizar os conectores pin header fêmea que, apesar de bons para prototipagem, são ruins para o produto final (não oferecem fixação segura dos fios jumper). Com o microcontrolador independente você é quem decide qual o tamanho e o formato da placa final, e quais serão os conectores mais adequados para seu produto (bornes de parafuso, conectores Molex e outros).
- **Exercício de engenharia:** montar um Arduino Standalone lhe proporcionará um exercício de engenharia de sistemas embarcados completo pois você precisará montar e/ou entender a parte elétrica (fonte, reguladores, capacitores, proteções), a parte digital (clock, reset, sinais de entrada/saída, interface com sensores e atuadores), o firmware (fuses, bootloader, programação ISP², debug), documentação (projeto esquemática, projeto da PCB) e fabricação do seu produto final.

Em resumo, ao montar seu próprio Arduino Standalone você estará dando seus primeiros passos para se tornar um verdadeiro engenheiro de sistemas embarcados. Obviamente esse é um objetivo ambicioso demais para ser tratado em uma pequena série de artigos como esta, então resolvi me limitar aqui aos aspectos básicos e mais simples de criar um Arduino Standalone na esperança de que munido desta base você seja capaz de aprender cada vez mais sobre sistemas embarcados e o fantástico mundo dos microcontroladores.

Neste primeiro artigo faremos a montagem básica do Arduino Standalone, ou seja, a montagem inicial do microcontrolador na PCB, incluindo a fonte de alimen-

¹Printed Circuit Board (PCB).

²In-System Programming (ISP), também chamado por *In-Circuit Serial Programming* (ICSP).

tação, o circuito de reset, e o circuito de clock externo. Meu objetivo é que este artigo seja um tutorial completo e detalhado o suficiente para você montar seu microcontrolador independente de forma segura³. Alguns pressupostos que estou assumindo:

- Montaremos um Arduino Standalone usando o microcontrolador Microchip ATmega328P-PU, que é o utilizado no Arduino Uno.
- Utilizaremos como fonte de alimentação uma bateria ou fonte de 9 V, conectada diretamente à protoboard.
- Nossa montagem será a mais fiel possível ao Arduino Uno⁴.

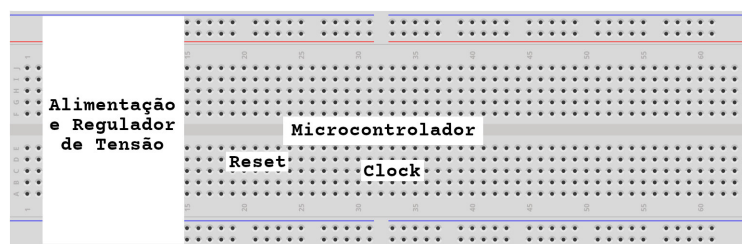
2 Materiais

Nosso Arduino Standalone terá, na protoboard, os seguintes grandes componentes:

1. **Alimentação:** usaremos uma bateria ou fonte de alimentação de 9 V mas o ATmega328P suporta, no máximo 5,5 V. Precisaremos então de um circuito regulador de tensão que receba os 9 V e entregue 5 V para o microcontrolador.
2. **Microcontrolador:** é o ATmega328P em si. Você pode remover o microprocessador do Arduino, ou comprar um novo.
3. **Clock externo:** usaremos um circuito de clock externo de 16 MHz com um cristal oscilador.
4. **Circuito de reset:** um botão na protoboard permitirá que o microcontrolador seja reiniciado rapidamente, sem a necessidade de desconectar a fonte de energia.

A organização desses grandes componentes na protoboard será aproximadamente a exibida na figura 1, a seguir:

Figura 1: Organização geral dos componentes



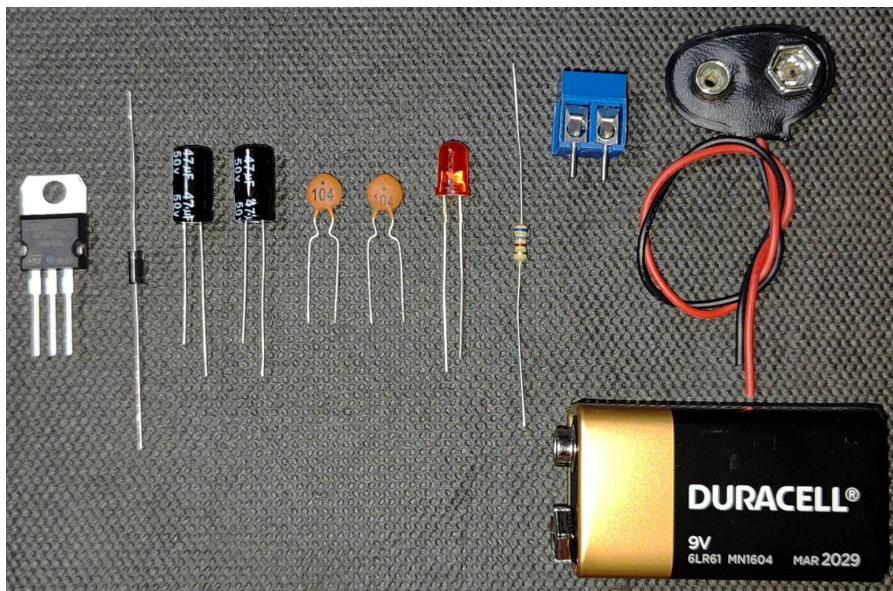
³Este artigo é, na verdade, o tutorial que eu queria ter tido acesso quando eu tentei montar um Arduino Standalone pela primeira vez.

⁴Este item parece um contra-senso pois acabamos de citar todas as vantagens de abandonar o Arduino e usar o microcontrolador independente. Sim, isso é meio que um contra-senso mesmo, mas estou partindo do pressuposto que você está montando seu primeiro Arduino Standalone e manter a arquitetura o mais próxima possível do Arduino, nesse momento, facilitará sua aprendizagem.

Os materiais e componentes necessários são os seguintes:

- **Alimentação:** a lista a seguir e a figura 2 mostram os componentes que utilizaremos para a alimentação do microcontrolador. Alguns itens são opcionais mas recomendados neste estágio inicial de seu aprendizado.
 - 1 regulador linear de tensão L7805CV
 - 1 diodo 1N4007
 - 2 capacitores eletrolíticos de 47 μ F
 - 2 capacitores cerâmicos de 100 nF
 - 1 LED vermelho de 5 mm (opcional, apenas para indicar que a protoboard está energizada)
 - 1 resistor de 680 Ω (opcional, apenas para indicar que a protoboard está energizada)
 - 1 conector borne de duas vias
 - 1 clipe para bateria de 9 V e 1 bateria de 9 V

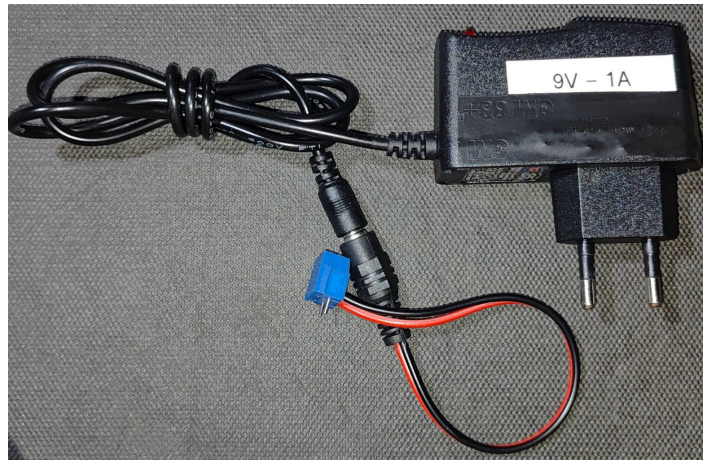
Figura 2: Componentes para a alimentação



Se você não quiser usar uma bateria, pode utilizar uma fonte de alimentação de 9 V (1 A)⁵, com um adaptador com borne de duas vias conforme a figura 3 (na página 6), para conectar a fonte à protoboard. Para mantermos nosso Arduino Standalone o mais portátil possível usaremos uma bateria comum mesmo.

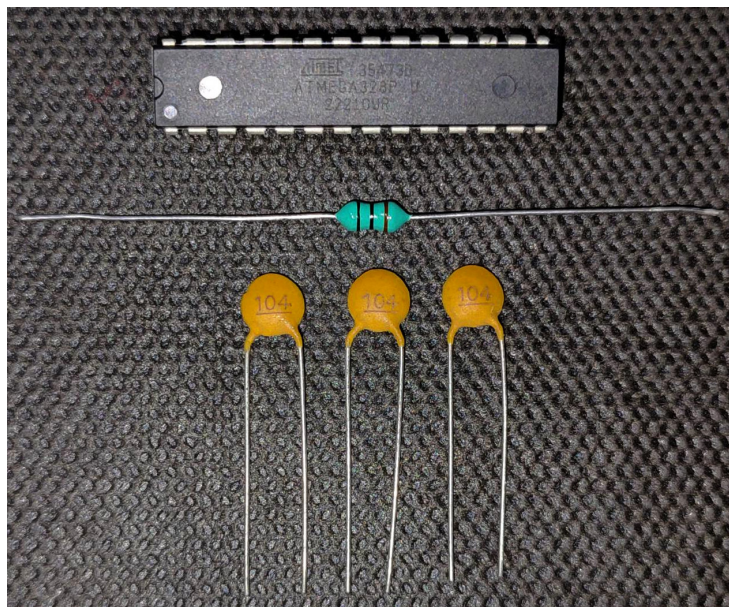
⁵Estou usando uma fonte que fornece corrente de até 1 A, mas você pode utilizar qualquer outra fonte de 9 V que tiver disponível.

Figura 3: Alternativa para alimentação



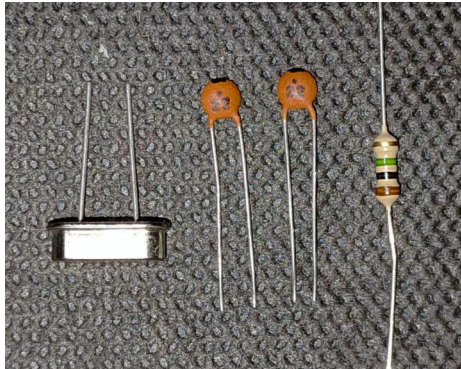
- **Microcontrolador:** precisaremos dos materiais ilustrados na lista abaixo e na figura 4 a seguir:
 - 1 microcontrolador ATmega328P-PU (você pode retirar, com muito cuidado, o microcontrolador de sua placa Arduino atual, ou comprar um novo microcontrolador)
 - 1 indutor axial de 10 μ H
 - 3 capacitores cerâmicos de 100 nF

Figura 4: ATmega328P-PU



- **Clock:** para montar o circuito de clock precisamos dos materiais a seguir (ilustrados na Figura 5):
 - 1 cristal oscilador de 16 MHz
 - 2 capacitores cerâmicos de 22 pF
 - 1 resistor de 1 M Ω

Figura 5: Clock externo



Além dos materiais e componentes principais listados anteriormente, também será necessário uma protoboard de 830 furos (eu prefiro os modelos que têm quatro linhas de energia separadas), ferramentas diversas (alicates de corte, decapadores de fio, alicates de bico fino, pinças) e, se disponível, um multímetro para verificar tensões:

Figura 6: Ferramentas úteis

