NESTE CAPÍTULO

- » Hackeando um hardware
- » Assumindo um controle remoto
- » Controlando uma câmera
- » Fingindo ser um teclado

Capítulo Online

Hackeando Outros Hardwares

Talvez lhe pareça familiar a ideia de que usar a ferramenta adequada para o trabalho apropriado faz toda a diferença no desempenho. O mundo está repleto de produtos cuja finalidade é fazer um bom trabalho, e gastou-se uma boa quantidade de tempo e dinheiro para desenvolvê-los e aprimorá-los. Infelizmente, nem tudo é desenvolvido levando o Arduino em consideração; desse modo, você pode encontrar uma barreira para transformar certos dispositivos na "ferramenta adequada" para o que tem em mente.

Hackear um hardware tem a ver com a reprogramação de dispositivos e sistemas para realizar as coisas que você quer que eles façam. Neste capítulo, você aprenderá a transformar com segurança esses dispositivos existentes, como controles remotos, em algo valioso para seu próprio projeto, possibilitando que seu Arduino vá além, sem comprometer a sua placa ou a sua vida.

Desmontando os Dispositivos sem Destruí-los

Antes de começar a falar sobre a reconstrução de hardware e dispositivos (o que faço na próxima seção com um controle de PlayStation), você deve analisar algumas regras importantes para se certificar de que esta não será a última vez que o hardware em questão funcionará:

- **Verifique se alguém já fez isso antes.** A internet abriga uma verdadeira abundância de conhecimento, por isso, não deixe de pesquisar antes de se aventurar pelo desconhecido. Há grande chance de que, se você quiser fazer alguma coisa, alguém já tentou antes (ou, pelo menos, tentou algo parecido). Nada melhor do que encontrar um site no qual todas as especulações vão ao encontro de suas necessidades. Isso o poupa da tentativa e erro de escolher qual controle remoto ou controle de jogos comprar. Caso tenha sorte, pode até encontrar instruções passo a passo completas, fotos e listas de compras para peças específicas de hardware.
- **Desligamento.** Certifique-se de que o hardware que você planeja desmontar não esteja ligado; isto é, retire as baterias e desconecte os cabos. Se você estiver cutucando alguma coisa com uma chave de fenda ou puxando os fios, a última coisa que quer é um curto-circuito danificando a placa (ou até você).
- **Tire fotos antes.** Fotos digitais são gratuitas e a maioria de nós tem acesso a uma câmera digital de algum tipo. Use-a. É bem fácil esquecer onde partes ou conexões estavam unidas quando já estão desmontadas, e uma simples foto de cada etapa da desmontagem pode ajudar muito.
- **Conheça suas juntas.** A maioria dos controladores de plástico e outros hardwares apresenta pequenas juntas de parafusos. Você pode removê-las com uma chave de fenda bem precisa e cuidadosamente desmontar o controle. Caso todos os parafusos estejam do lado de fora e você ainda não consiga desmontá-lo, confira se há mais parafusos debaixo dos adesivos. Os parafusos podem estar escondidos aí; muita vezes, é uma tática inteligente para os varejistas ou os fabricantes verificarem se o produto foi adulterado.
- **Utilize uma bandeja de peças.** Como está removendo peças, é extremamente fácil perder parafusos e componentes, portanto, faça questão de ter uma bandeja de peças. Ela pode ser um case ou uma caixa com divisores para organização extra, ou até mesmo uma bandeja de parafusos magnéticos, encontrada na maioria das boas lojas de ferramentas. Ou seja um tremendo mão de vaca e use uma jarra, ou até mesmo um envelope.
- **Não force as peças.** E, se fizer isso, arque com as consequência. Muitas peças de plástico em hardware não são feitas para serem desmontadas, e os acessórios que encaixam no lugar podem ser bem complicados na hora de

desmontar. Às vezes — com muita paciência, e bota paciência nisso — você pode encontrar esses clipes e soltá-los com chaves de fenda, usando um pouco de força. Só que eles podem quebrar, e é assim que as coisas são. O uso de fixações internas ou cola significa que a força bruta é às vezes a única maneira de abrir gabinetes de hardware. Em geral, como você quer somente as peças que funcionam internamente, peças rachadas e quebradas não são um problema, mas tome cuidado para não danificar o que está dentro quando forçar algo para que se abra.

- **Conheça os termos de sua garantia estendida.** Nem espere recuperar seu dinheiro se você quebrar um pedaço de hardware. Esses dispositivos foram fechados desde que saíram da fábrica para garantir que os padrões sejam mantidos. O que você está fazendo é uma grande cirurgia em um dispositivo pré-embalado; desse modo, é bem improvável que uma empresa o devolva após uma "otimização" que não deu certo.

Hackeando um Controle Remoto

Um controle remoto é um jeito inacreditavelmente prático de se burlar dispositivos difíceis ou desconhecidos que você quer incluir em seu projeto. Ao usar um componente chamado acoplador óptico, você consegue acionar botões em um controle remoto da mesma maneira que consegue piscar um LED com o seu Arduino. Compreender como utilizar esse componente simples lhe permite fazer coisas incríveis com o seu Arduino, de maneira rápida, fácil e segura.

Bem antes de existirem as placas Arduino, já existiam truques de hacking para dispositivos, sobretudo no mundo da arte. Se já esteve em um show de belas-artes, certamente viu exibições de arte de vídeo com muitos monitores ou projetores rodando em aparelhos de DVD (ou, antes dos DVDs, em videocassetes) que estavam sincronizados. Na maioria dos casos, isso era feito no olho mesmo, pressionando simultaneamente o botão play nos aparelhos de DVD. Isso é bom para sequenciar os vídeos, mas e se quiser ter um temporizador mais preciso para o dispositivo, ou, melhor ainda, fazer com que uma exposição de arte seja mais responsiva, reproduzindo alguma coisa somente quando alguém entrar na sala? Obviamente que o Arduino pode ajudar.

Uma parte de um DVD player *é* de baixa voltagem: o controle remoto. Via de regra, isso vale para qualquer controle remoto sem fio e sem pilhas AA ou AAA. Tal fato é de suma importância, pois o controle remoto é feito para controlar dispositivos maiores sem qualquer conexão elétrica. Ao hackeá-lo, você consegue controlar diretamente qualquer coisa que esteja se comunicando "nos bastidores" do dispositivo perigoso alimentado pela rede elétrica.

Você geralmente encontra dois tipos de controle remoto. O primeiro é o controle infravermelho, normalmente usado para televisores e aparelhos de DVD,

por exemplo. Esse tipo de controle remoto pisca um LED infravermelho invisível ao olho humano, mas altamente visível para o receptor de infravermelho (parecido com um sensor de luz) no dispositivo. Para evitar a interferência de outras fontes de luz e iluminação, como sol ou lâmpadas, o dispositivo é dotado de um filtro que permite que somente um comprimento de onda muito preciso de luz infravermelha seja recebido.

O outro tipo de controle remoto é similar àquele encontrado em carros ou aeronaves de controle remoto (que às vezes as crianças ganham no Natal e não duram nem dez minutos). Em vez de luz, esses controles remotos utilizam ondas eletromagnéticas invisíveis de frequência extremamente baixa, porém conseguem se deslocar muito além dessa frequência.

O tipo de comunicação que seu controle remoto usa determina o alcance dele, mas o hardware físico interno é o que de fato importa. Na hipótese de você desmontar um controle remoto do qual não precise mais, e caso se sinta à vontade para montá-lo de volta, tente descobrir como ele funciona por baixo. A maioria dos controles remotos da TV tem botões de borracha e silicone. Em geral, cada botão de borracha tem um amortecedor pequeno condutor nas cores cinza ou preta, na parte inferior. Esse amortecedor pressiona em lotes as tiras de cobre que se interligam como dedos, mas, na realidade, elas não estão conectadas. Quando o bloco condutor pressiona essas tiras, os dois lados se conectam e acionam o botão. Com alguns controles remotos, como aqueles de carrinhos e aviões de controle remoto, você geralmente descobre que alguns botões são montados de várias maneiras e são acionados por botões de controle. Eles são semelhantes aos botões do seu kit Arduino.

Para controlá-los, você precisa manusear acopladores ópticos. Falarei sobre eles na seção a seguir.

Fazendo um circuito com acopladores ópticos

Como você está usando um acoplador óptico para controlar um botão, a primeira tarefa é encontrar esse botão. Existem muitos tipos diferentes de controles remotos disponíveis. Você pode escolher um controle remoto de TV ou um controle remoto para um carrinho de controle remoto, mas, para este exemplo, eu uso um controle remoto com fio parecido com um antigo controle de PlayStation.

Todos esses botões também assumem diferentes formas, por isso, é importante saber o que procurar. Basicamente, cada botão é uma conexão interrompida conectada por algo condutor. Na maioria dos controles remotos de TV e console de jogos, o botão está frequentemente escondido embaixo de uma camada de borracha ou plástico. Em geral, essa camada de borracha ou

plástico tem um estofo protetor condutor em sua parte inferior que conecta duas superfícies condutoras para completar um circuito e desencadear uma ação.

Depois de localizar esses dois estofos condutores, você precisa de algo que possa substituir o botão e ser controlado pelo seu Arduino: para isso, você usa um *acoplador óptico*. Um acoplador óptico é um interruptor ativado pela luz — dê uma olhada na Figura CO-1 — e é chamado de muitos nomes, incluindo opto-isolador, acoplador fotográfico e isolador óptico. A luz é colocada dentro do gabinete preto desse minúsculo circuito integrado para que você nunca a veja ligar e desligar, porém o interruptor o faz.

Um acoplador óptico tem quatro pernas, duas para cada lado do circuito. De um lado é um LED, e do outro, um fototransistor. O fototransistor é o mesmo que o transistor usado no Capítulo 8, mas, em vez de fechar o circuito quando uma pequena tensão é aplicada, ele se fecha quando a luz é acesa. Esse design possibilita que parte do circuito Arduino permaneça eletricamente isolada do controle remoto, o que significa que ele pode se integrar ao circuito de controle remoto com muito pouca interrupção no circuito. No exemplo desta seção, você aprende como conectar um acoplador óptico a um botão existente para transformá-lo em um botão controlado pelo Arduino. Essa é uma tarefa simples que exige muito pouca soldagem.

Para este projeto, você precisa de:

- Um controle remoto de sua escolha
- Um multímetro
- Um ferro de solda
- Dois fios conectores cortados no comprimento
- Um Arduino Uno
- Uma breadboard
- Um acoplador óptico
- Fios de jumper

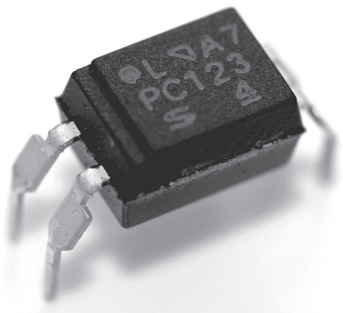

FIGURA CO-1: Um típico acoplador óptico.

Primeiro, você precisa soldar seus dois fios conectores em cada lado da chave eletrônica existente. Isso facilita a incorporação da chave eletrônica à sua breadboard. Faça questão de ter fio suficiente para a placa de circuito se assentar comodamente ao lado do Arduino e da breadboard.

Apare as extremidades do fio e torça-as entre o polegar e o indicador. Quando elas estiverem perfeitas e uniformes, estanhe-as com uma pequena quantidade de solda, como na Figura CO-2. Você faz isso para garantir que elas não sejam desmanteladas e para facilitar seu encaixe em uma breadboard e sua solda em outras superfícies. Ao fazer a estanhagem, recomenda-se deslizar o ferro de solda até a ponta do fio exposto para remover qualquer excesso de solda. Se um cordão de solda se formar na ponta, você sempre pode cortá-lo para deixar uma ponta limpa.

CUIDADO

Caso esteja hackeando um controle com uma bateria, lembre-se de remover as baterias antes de começar a aplicar calor nos fios energizados.

O primeiro passo é encontrar o botão que você deseja controlar. Para o controle PlayStation, um dos lados do botão é um pad de metal, e a parte que os conecta é anexada ao case de plástico. Depois de remover o case, você pode facilmente encontrar os pads de metal e prender os fios a eles, como pode observar na Figura CO-2.

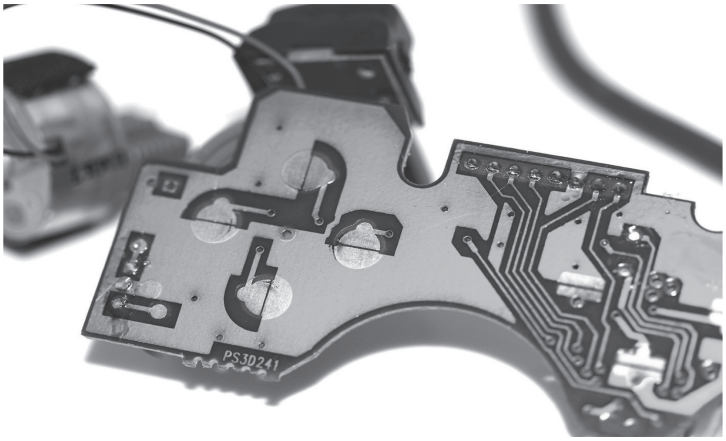

FIGURA CO-2: Dentro de um controle PlayStation, você pode ver os contatos de cada botão que está esperando para ser hackeado!

O controle também pode ser um botão ou uma chave eletrônica, então talvez seja necessário um pouco de criatividade para conectar os fios ao botão. Ache o botão que você quer controlar.

Primeiro, você precisa remover as baterias do seu controle remoto. Precisa descobrir de que maneira ele está conectado, e o método mais confiável é usar o multímetro para verificar a continuidade. Se bipar, há uma conexão.

Os botões de quatro pernas são muitas vezes conectados em pares, portanto, ao usar seu multímetro para verificar a continuidade (veja o Capítulo 5), certifique-se de encontrar um par de pernas que emita o bipe apenas quando o botão é pressionado. Marque as pernas ou procure se lembrar delas.

Você precisa encontrar uma superfície para soldar. Os pads do PlayStation são planos e fáceis de aplicar solda. No que diz respeito aos botões, você pode remover o botão completamente com um alicate e soldar diretamente na placa de circuito ou soldar um fio na própria perna do botão. A perna em si é geralmente a opção mais segura.

Estanhe o pad de metal ou a perna da mesma maneira que você estanha o fio, deixando uma pequena camada de solda.

Você também precisa descobrir de que modo a chave eletrônica está conectada — ou seja, qual lado é o positivo e qual é o negativo. Como no sketch Button do Arduino (descrito no Capítulo 7), um botão é conectado com frequência ao terra quando não está em uso, e uma pequena tensão passa por ele quando pressionado. Para encontrar o terra, você pode usar o verificador de continuidade do multímetro com outras partes que sabe que estão aterradas, como o terra da bateria. Quando ele emite um bipe, você tem uma conexão e, por um processo de eliminação, sabe que esse lado é aterrado e que o lado oposto está fornecendo a tensão.

Neste exemplo, eu soldei os fios aos pads de metal. Segure a ponta estanhada do fio no pad estanhado do controlador e aplique calor com o ferro de solda. Esse procedimento derrete a solda uniformemente e ajuda a obter um bom contato. Depois que a solda tiver esfriado, você terá dois cabos auxiliares para conectar seu controle remoto à sua placa de montagem.

Você pode levar em consideração três complementos extras que acrescentam mais robustez à sua fiação. Primeiro, você pode usar algumas gotas de cola quente para segurar os fios no lugar e evitar que a tensão comprometa a junta de solda. A cola deve estar no isolamento de plástico do fio, que é extremamente forte quando puxado, e na própria placa de circuito, que proporciona uma área de superfície maior para a fixação da cola. Segundo, você pode torcer os dois fios juntos para dar a eles uma força maior e deixar a conexão com uma aparência mais arrumada. Terceiro, você pode conectar um bloco de terminais (bloco shock) ao final dos fios, facilitando a conexão e desconexão do controle conforme necessário, como na Figura CO-3.

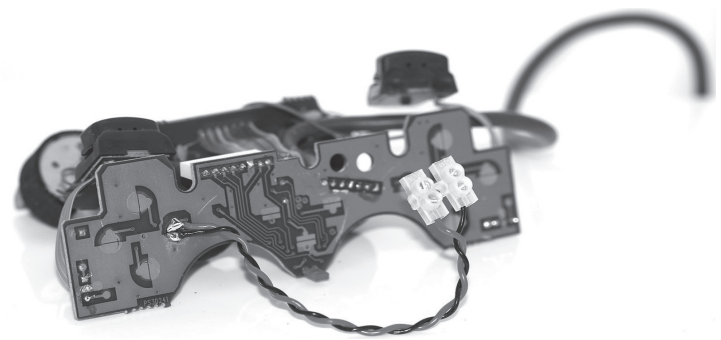

FIGURA CO-3: Dois fios de jumper conectados a um botão PlayStation.

Agora que você tem todos os componentes, pode montar a placa de montagem e testar o circuito. Complete o circuito conforme mostrado nas Figuras CO-4 e CO-5 para controlar seu controle remoto com seu Arduino.

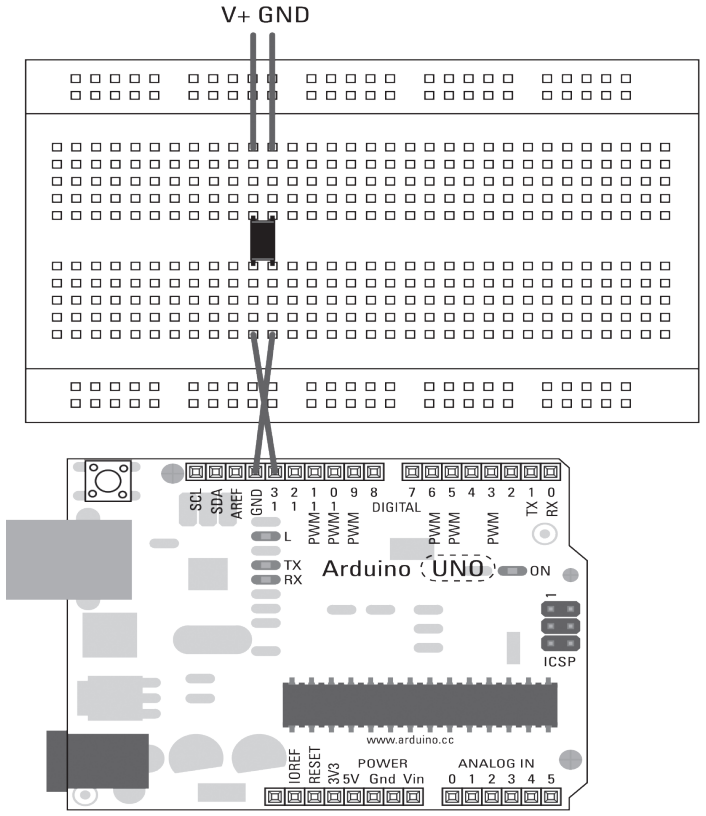

FIGURA CO-4: O layout do circuito para o circuito do acoplador óptico.

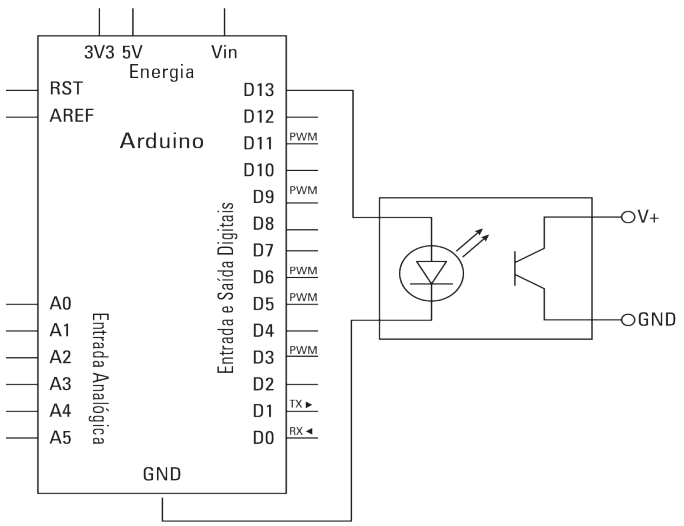

FIGURA CO-5: O esquema do circuito do acoplador óptico.

Você pode dividir o acoplador óptico em duas metades. De um lado está a entrada, que é basicamente uma fonte de luz LED, semelhante ao LED no pino 13 do seu Arduino. Ele tem uma polaridade, portanto, a energia deve ser aplicada ao ânodo (+), e o cátodo (-) deve estar conectado ao terra para que funcione. A outra metade é um transistor operado por luz. A energia também deve fluir através dessa metade do jeito correto para evitar danificar o acoplador óptico. Talvez seja necessário dar uma de Sherlock Holmes (como mencionado anteriormente no livro) usando o multímetro para descobrir qual lado da chave eletrônica está fornecendo energia e qual lado está aterrando.

Conecte seu botão ao acoplador ótico utilizando os fios conectores e você estará pronto para assumir o controle dele!

Depois de montar seu circuito, você precisa do software apropriado para usá-lo. Na barra de menu do Arduino, clique em Arquivo ➪ Exemplos ➪ 01.Basics ➪ Blink, e você deve ver o sketch Blink que segue.

```
/*
  Blink
  Acende um LED por um segundo, depois desliga por um segundo,
  repetidamente.

  Este código de exemplo está em domínio público.
*/

// O pino 13 tem um LED conectado na maioria das placas Arduino.
// Dê um nome a ele:
int led = 13;

// a rotina da setup é executada uma vez quando você pressiona reset:
void setup() {
  // inicializa o pino digital como uma saída.
  pinMode(led, OUTPUT);
}

// a rotina de loop é executada repetidamente e não para nunca:
void loop() {
  digitalWrite(led, HIGH);   // liga o LED (HIGH é o nível de voltage)
  delay(1000);               // espera por um segundo
  digitalWrite(led, LOW);    // liga o LED fazendo com que a voltagem
                             //fique LOW
  delay(1000);                 // espera por um segundo
}
```

Depois de carregar o sketch, o Arduino aciona o acoplador óptico a cada segundo (ligado por um segundo, desligado por um segundo). Se você seguiu as etapas anteriores, nada acontece, porque as baterias do controle remoto foram removidas. Peguei você!

Desconecte seu Arduino da porta USB, coloque as baterias de volta no controle e, em seguida, conecte seu Arduino de volta. Seja lá qual for o seu botão, deve acontecer isso: ligado por um segundo, depois desligado por um segundo. O controle remoto de um carrinho pode fazer com que ele acelere aos trancos; um controle remoto para a sua TV pode aumentar o volume um passo de cada vez.

Caso você não veja nada aceso, confira os seus fios e cabos:

- Confira se está usando o número de pinos correto.
- Confira as conexões na breadboard. Se os fios de jumper ou os componentes não estiverem conectados corretamente nas linhas da placa breadboard, eles não funcionarão.
- Confira se seu acoplador óptico está na posição correta e se os pinos adequados estão sendo usados. Em caso de dúvida, desconecte o fio do dispositivo e teste as pernas do acoplador óptico com o verificador de continuidade do multímetro para certificar-se de que a corrente seja alternada.

Se você não estiver familiarizado com o sketch Blink, confira o Capítulo 4.

Fazendo ainda mais com um circuito com acoplador óptico

O circuito com acoplador óptico é uma ótima maneira de controlar objetos controlados remotamente com um pequeno circuito integrado. No exemplo desta seção, você simplesmente usa o Blink para piscar e desligar o circuito, coisa básica, e muitas vezes é o necessário para controlar coisas maiores. Entretanto, também é possível utilizar os acopladores ópticos com o `analogWrite` dos circuitos de saída, gerando a mesma saída que desvanece e esmaece um LED ou controla a velocidade de um motor. No entanto, faça questão de examinar sua saída. Se você estiver ligando e desligando um ventilador do escritório, levará um bom tempo para que o ventilador aumente a velocidade, e os pulsos de alta frequência são desperdiçados nesse processo.

DICA

Seja o que for que você esteja tentando, comece sempre com algo simples como Blink e ajuste o tempo depois que estiver contente com a comunicação.

Fazendo uma Câmera Reflex Monobjetiva (SLR) com um Botão Remoto

Uma variedade enorme de softwares (alguns mencionados no Capítulo 16 do livro) manipulam webcams para capturar imagens, mas (pelo menos, por enquanto) a resolução e a qualidade das webcams costumam deixar, e muito, a desejar, resultando em imagens imprecisas e granuladas. Webcams com lentes de alta qualidade e boa resolução estão disponíveis, mas, não raro, custam os olhos da cara, em uma faixa de preço semelhante à maioria das SLRs digitais. Não seria ótimo se você pudesse tirar fotos remotamente com uma SLR digital de altíssima qualidade?

As SLRs digitais conseguem capturar imagens de alta resolução e armazená-las ou passá-las diretamente para um computador. Caso possa controlar remotamente uma dessas, você abre um leque gigantesco de possibilidades para fotografia com time-lapse, fotografia do tipo HDR (*high dynamic range*) e fotografia de alta velocidade. Neste exemplo, você aprende como conectar um botão disparador remoto e a conexão de autofoco da sua SLR digital à sua placa Arduino.

As versões do botão disparador remoto já estão consolidadas como um produto e normalmente são usadas em câmeras com tripé, para evitar movimentos da câmera ao pressionar o botão de liberação do disparador. Alguns são via cabo, outros, via fio, mas ambos têm um preço bem salgado para o que são. Eles estão disponíveis na maioria das lojas de varejo ou online de fotografia, entretanto, você encontra preços melhores em sites como o eBay. Neste exemplo, você aprenderá sobre um conector básico com fio.

Dentro do case de plástico estão algumas chaves eletrônicas. Normalmente, há um interruptor deslizante, que aciona o foco automático antes de disparar o botão disparador. Na câmera, há um soquete, que varia dependendo de sua marca. Existem muitos plugues e tomadas disponíveis para inúmeras marcas de câmeras, porém, infelizmente, não consigo abordar todos eles neste livro.

Minha câmera é uma Canon 60D, que tem um conector de 2,5mm jack com três polos. É pequeno como um fone de ouvido de 3,5mm, parece idêntico. O número de polos refere-se ao número de fios utilizados. Neste caso, existem três fios: um para o foco automático, um para o disparo do botão disparador e um para o terra, que aciona um ou ambos. Esses conectores estão amplamente disponíveis e você pode soldá-los sozinho se quiser enfrentar um desafio. A Figura CO-6 é uma ilustração do conector com os pinos apropriados.

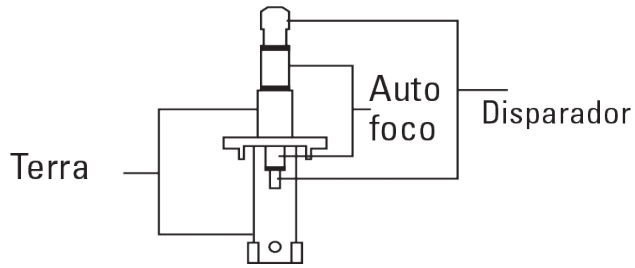

FIGURA CO-6: Diagrama da versão do botão disparador da Canon 60D.

Se você tiver uma versão de um botão disparador existente que não esteja usando, remova a extremidade do disparador desparafusando e desmontando o gabinete da chave ou simplesmente cortando o cabo e removendo-o para expor os três fios. Você pode validar a tensão dos fios usando o teste de continuidade do multímetro. Leia a box "Soldando um Conector Jack de 2,5mm" neste capítulo se pretende soldar o jack de 2,5mm por conta própria.

SOLDANDO UM CONECTOR JACK DE 2,5MM

Se você estiver soldando um conector jack de 2,5mm, tome muito cuidados com os outros conectores.

Dentro do conector de 2,5mm existe muito pouco espaço para fios muito longos ou gotas de solda que sejam muito grandes. As partes de plástico que ficam entre os contatos de metal também são extremamente sensíveis ao calor e, se você aplicar o ferro de solda por muito tempo, elas derretem e se deformam.

Primeiro, estanhe a ponta de cada fio. Recomenda-se um fio 7/0,2 de 7 núcleos ou mais fino. Você pode achar que o isolamento no fio encolhe quando o calor é aplicado; garanta que ele esteja totalmente estanhado até o final do isolamento e, em seguida, apare o fio estanhado em 2-3mm. Os contatos são minúsculos, então, você precisa apenas de um pouquinho de fio exposto.

Em seguida, estanhe cada contato no conector de 2,5mm. Somente uma pequena quantidade de solda é necessária, portanto, certifique-se de remover quaisquer bolhas com um sugador de solda. Com cuidado, solde o fio e os contatos juntos. Os fios ficam um em cima do outro, desse modo, comece primeiro com o menor (menos acessível) e trabalhe. Certifique-se também de que algum isolamento esteja entre os contatos. Quando você tem um soquete de trabalho e três fios desencapados, faça um teste rápido. Conecte o disparador à sua câmera e ligue-o. Aproxime o fio do disparador e os fios terra: a câmera deve tirar uma foto. Tente configurar sua câmera para foco automático e junte os fios de foco automático e aterramento. Tente ambos, e a câmera deve focar e tirar uma foto!

Fazendo um circuito transistor

Agora que você tem os fios expostos, tudo o que resta é fazer seu próprio circuito para alterná-los quando quiser. Isso é possível com um simples circuito transistor.

Para este projeto, você precisa de:

- Uma câmera digital SLR
- Uma versão de um botão disparador remoto (comprado ou feito em casa)
- Um multímetro
- Um ferro de solda
- Um Arduino Uno
- Uma placa breadboard
- Um transistor
- Fios de jumper

Se medir a voltagem entre o fio e o terra, ela deve estar em 2,5V, assim você pode ter certeza de que está lidando com um sinal de baixa voltagem. Muitos transistores funcionam com tensões muito baixas. Dois bons exemplos são o P2N2222AG, incluído em muitos kits, ou o TIP120 (ou 121/122). Outros transistores, como o MOSFET IRF510/520, são um exagero em relação às aplicações simples, porque são destinados a usos de alta potência.

O transistor atua da mesma forma que no Capítulo 8, abrindo e fechando o gate entre a(s) fonte(s) de energia e o terra para completar o circuito. Você pode optar por conectar somente um fio ou conectá-los a transistores independentes. Neste exemplo, você conecta ambos ao circuito transistor. Isso possibilita que a câmera foque e então dispare todas as vezes, enquanto estiver em foco automático. Quando definida para foco manual, a câmera tira fotos, o que é significativamente mais rápido e melhor para fotos de ação.

Utilizando o ferro de solda, estanhe a ponta dos fios desencapados a fim de prepará-los para uso na breadboard. Monte o circuito conforme mostrado nas Figuras CO-7 e CO-8 para controlar seu disparador com seu Arduino. Como observado no Capítulo 8, analise as informações do seu transistor antes de começar. Depois de examinar a ficha técnica, conecte o pino Base ao pino 13 do seu Arduino.

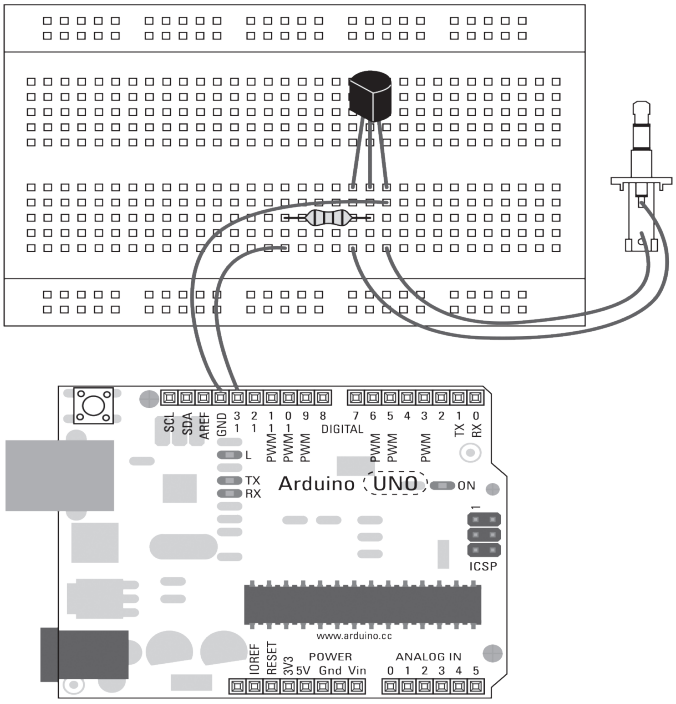

FIGURA CO-7: O layout do circuito para o circuito digital do transistor SLR.

LEMBRE-SE

Como um resistor embutido na placa está conectado ao pino 13, não é necessário acrescentar um, mas, se você usar outro pino, certifique-se de utilizar um resistor adequado, consultando a ficha técnica de seus transistores e os cálculos no Capítulo 6.

O Coletor deve ser conectado ao pino do disparador do conector jack de 2,5mm e o Emissor deve estar conectado a partir do terra para o conector jack.

Depois que seu circuito é montado, você precisa do software adequado para usá-lo. Na barra de menu do Arduino, cliquem em Arquivo ➪ Exemplos ➪ 01Basics ➪ Blink, e você dever ver o sketch Blink que segue. Para ver esse sketch, confira o Capítulo 4.

Como você faz interface com hardware, é essencial conhecer as configurações da sua câmera. A Canon 60D tem configurações diferentes para o disparo da câmera, tomada única, múltiplas tomadas de disparos de alta velocidade, temporizador e controle remoto. Na hipótese de a câmera estar configurada para tomada única, ela tira uma foto quando o disparador é pressionado e não tira outra até que ele seja totalmente liberado e pressionado novamente. Com o sketch Blink, você está na verdade segurando o disparador por um segundo, soltando por um segundo e depois repetindo todo o processo. Essa configuração funciona bem para tirar uma foto a cada dois segundos, entretanto, se você alterar o modo de disparo para muitas tomadas ou para inúmeras tomadas de

fotos de alta velocidade, a câmera tirará quantas fotos puder enquanto o disparador estiver pressionado, vai parar por um segundo e, em seguida, começará outra descarga de tomadas de fotos. Tal configuração é ótima para fotos com movimento e muita ação, mas consome sua bateria.

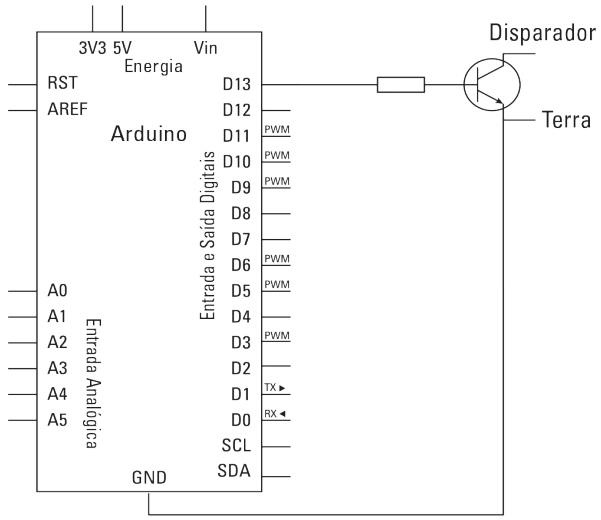

FIGURA CO-8: O esquema para o circuito digital do transistor SLR.

É melhor configurar sua câmera para uma única tomada de foto e editar o código para criar um delay menor entre `HIGH` e `LOW` e um delay maior de alguns segundos depois. Por exemplo, os valores de delay redefinidos no código a seguir são 100 e 3.000:

```
digitalWrite(led, HIGH);   // liga o LED (HIGH é o nível de voltage)
  delay(100);                 // espera por um segundo
digitalWrite(led, LOW);    // liga o LED fazendo com que a voltagem fique
                           // LOW
  delay(3000);                 // espera por um segundo
```

Depois de carregar o código, desconecte seu Arduino, conecte o conector jack de 2,5mm à sua câmera e ligue-a. Reconecte seu Arduino, e a câmera começa a tirar uma foto a cada quatro segundos.

Caso você não veja nada aceso, confira os seus fios e cabos:

- Confira se você está usando o número de pinos correto.
- Confira as conexões na breadboard. Se os fios de jumper ou os componentes não estiverem conectados corretamente nas linhas da placa breadboard, eles não funcionarão.

» Verifique se o seu transistor está no lugar correto e se os pinos adequados estão sendo usados. Em caso de dúvida, confira a voltagem através das pernas do coletor e do emissor do transistor com a regulagem de voltagem do multímetro. Certifique-se de que uma pequena voltagem seja exibida na tela do multímetro sempre que o pino for acionado.

Abordo o sketch Blink no Capítulo 4, logo, se você não tiver certeza do que está acontecendo, dê uma olhada lá.

Fazendo mais com um circuito transistor

Agora que você controla totalmente o disparador da sua câmera, é necessário alguma coisa para acioná-lo. Uma combinação excelente é um alarme a laser ou laser trip wires, conforme descrito no Capítulo 12. Você pode utilizá-lo como um dispositivo de segurança para pegar pessoas que estão cometendo um crime em flagrante, ou mesmo de um modo brincalhão para obter fotos espontâneas de alta qualidade de pessoas em uma festa. Para controlar esse dispositivo na prática, é preciso fazer com que ele seja resistente o bastante para durar mais. Você pode transferir facilmente esse circuito para uma placa de prototipagem (stripboard) ou para uma Proto Shield e alojar seu circuito em um case resistente, aproveitando as dicas do Capítulo 13.

Caso não goste dessa solução conectada fisicamente por fios ou não consiga achar os detalhes técnicos do seu conector, há outras maneira de se comunicar com as câmeras também, como por intermédio de controles remotos infravermelhos. Por outro lado, pode-se hackear um controle remoto existente ou encontrar uma biblioteca que pode piscar um LED nas frequências apropriadas para imitar um controle remoto. Muitas dessas soluções estão no Playground do Arduino (`http://www.arduino.cc/ playground/Main/InterfacingWithHardware#Camera` — conteúdo em inglês).

Substituindo um Mouse e um Teclado

Nos Capítulos 13 e 14 do livro, você aprende sobre a comunicação com outro software em um computador. É possível fazer isso de muitas formas, mas também há alguns truques práticos para fazê-lo de maneira rápida e fácil. Um desses métodos é disfarçar seu Arduino como outro periférico que seu computador já reconhece, como um teclado ou mouse. Usando um periférico reconhecido, você pode se comunicar com um computador praticamente sem nenhuma configuração, assim como pode conectar um mouse e usá-lo instantaneamente.

Há modos de hackear teclados e mouses que implicam retirar os periféricos da maneira que você fez no restante dos exemplos deste capítulo. Mas não seria

legal se o Arduino fingisse ser um mouse ou um teclado? Afinal de contas, ele é um dispositivo USB. Veja bem, até pouco tempo, isso não era possível, mas uma adição recente aos produtos da linha Arduino — a placa Leonardo — mudou essa situação. Confira a placa Leonardo na Figura CO-9.

FIGURA CO-9: Uma placa Leonardo novinha em folha.

O Arduino Leonardo usa um microcontrolador diferente daquele do Uno ou do Mega: ele utiliza um ATmega32u4 (o "u" significa USB). Ele tem comunicação USB embutida, então, em vez de converter de USB para serial e depois de serial para USB, o Arduino se comunica diretamente com a porta USB e é tratado como qualquer outro dispositivo USB.

A placa Leonardo tem uma série de outras vantagens:

- Pode atuar como um dispositivo de interface humana (HID). Isso significa que o Leonardo pode se comportar como um mouse ou teclado. Tal potencialidade é excelente para facilitar a interface com o software e também para criar periféricos de computador específicos.
- Ele também pode se comunicar em série, assim como o Uno faz em relação à comunicação tradicional do Arduino.
- Ele é mais barato também. Por ter menos componentes, gasta-se menos para montar a placa e, consequentemente, é mais barata de comprar. Você também pode comprar a placa sem os soquetes header para economizar mais.
- Ele tem um pino a mais de sinal PWM que um Arduino Uno. Não é uma bela de uma vantagem, mas é melhor que nada e já ajuda. O pino 13 também é capaz de manipular sinal PWM.

CUIDADO

Lembre-se de uma questão importante antes de começar a utilizar o Leonardo. Por ele estar agindo como um teclado ou mouse, você pode perder o controle das entradas do seu computador, coisa que dificulta a reprogramação. Se disser ao seu Leonardo para digitar um poema e fazer um loop nesse comando, é exatamente o que acontecerá — para sempre. Caso isso aconteça na janela do seu sketch, você já pode imaginar o caos generalizado. A melhor opção é ter as saídas de teclado e mouse controladas por um interruptor físico ou botão conectado ao seu Arduino, de modo que você não só consiga manter sua placa conectada, mas também pará-la quando necessário.

Pense nisso como um botão de parada de emergência, caso as coisas saiam erradas.

A placa Leonardo também apresenta outras diferenças em relação ao Arduino Uno normal:

- Não tem porta serial definida. Como a comunicação serial ocorre por meio de uma porta serial virtual, o Leonardo não tem um ID fixo. Ou seja, sempre que a placa é resetada, a porta serial recebe um novo ID (/dev/tty.usbmodemXXXXXX ou COM X). Isso é um problema para programas que procuram uma porta serial fixa, portanto, um Uno pode ser preferível nessas ocasiões.
- A porta serial não é redefinida quando aberta. Em um Uno, o sketch é redefinido sempre que a porta serial é aberta, seja para comunicação serial ou para o monitor serial. Este não é o caso do Leonardo, o que pode significar que as linhas Serial.print, Serial.println ou Serial.write na função `setup ()` não são vistas. Uma técnica para evitar a falta dessas instruções é verificar se há uma conexão serial:

```
// enquanto a conexão serial não estiver presente, não faça nada
while (!Serial);
```

- A comunicação serial é bem mais rápida. Como não há conversão serial para USB, a placa Leonardo é capaz de ocupar a memória serial do seu computador muito mais rápido do que as placas anteriores. Se achar que seu monitor serial está devagar ou ficando consideravelmente lento, inserir um pequeno delay de apenas 1 milissegundo ajuda.

```
// espera por 1 milissegundo
delay(1);
```

- Um soquete micro USB substitui o soquete USB convencional. Em vez do USB AB normal, é necessário usar um USB A para um micro B, como os da maioria dos dispositivos Kindle e dos telefones e tablets Android.
- Os LEDs indicadores foram movidos. Em vez do sistema usual de LEDs no Arduino Uno e nas placas anteriores, todos os LEDs são encontrados ao

longo da borda esquerda da placa entre o soquete micro USB e o soquete de energia (veja a Figura CO-10).

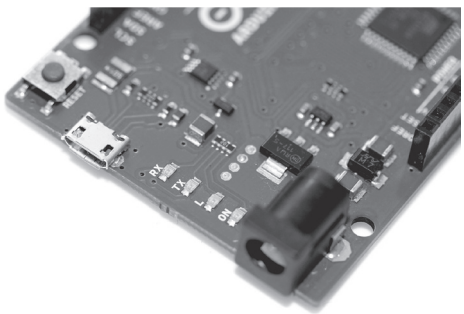

FIGURA CO-10: Disposição nova de LED para o Leonardo.

Configurando a placa Leonardo

Conforme ocorre com qualquer dispositivo novo em seu computador, você precisa garantir que este dispositivo seja reconhecido para que ele funcione adequadamente.

No MacOS, configurar a placa Leonardo é simples. Quando você conecta seu Leonardo pela primeira vez, o Assistente de Configuração do Teclado informa que "Seu teclado não pode ser identificado". Só isso, não precisa configurar mais nada; basta clicar no botão vermelho para fechar a janela e você está pronto para usá-la.

No Windows 7, você deve encontrar os drivers corretos. Ao conectar sua placa Leonardo pela primeira vez, o Assistente para Adicionar o Hardware novo tenta localizar os drivers. Caso contrário, siga estas etapas:

1. **Vá para o menu Iniciar e, na caixa Pesquisar Programas e Arquivos, digite** Gerenciador de Dispositivos.
2. **Selecione Gerenciador de Dispositivos na lista que aparece.**

   Você deverá ver uma lista de todos os dispositivos conectados ao seu computador.
3. **Encontre a placa Arduino Leonardo e clique com o botão direito nela.**
4. **Escolha Atualizar Driver e, na janela que aparece, escolha Procurar Meu Computador para o Software do Driver.**
5. **Acesse pasta do aplicativo Arduino e a pasta Drivers que ele contém e clique em OK.**

6. **Percorra com o mouse os menus para concluir a instalação dos drivers, e seu Leonardo estará pronto para uso.**

Caso esteja utilizando o Ubuntu 10.0.4 ou posterior, você não precisa instalar nenhum driver.

Implementando o sketch KeyboardMessage

Neste exemplo, você aprende a simular um teclado usando uma placa Arduino Leonardo. No sketch KeyboardMessage, sua placa Leonardo digita uma mensagem sempre que um botão físico é pressionado. A mensagem é uma mistura de texto predefinido e um temporizador variável que aumenta quando o botão é pressionado.

Você precisa de:

- Uma placa Arduino Leonardo
- Uma breadboard
- Um botão
- Um resistor de 10k ohm
- Fios de jumper

O circuito é o mesmo que o sketch Button básico (abordado no Capítulo 7), porém, com o Leonardo, você tem a opção adicional de saída do teclado ou do mouse. Essa funcionalidade pode ser ótima para se comunicar com outro software na tela ou até mesmo registrar dados em um documento de texto. Complete o circuito conforme mostrado nas Figuras CO-11 e CO-12.

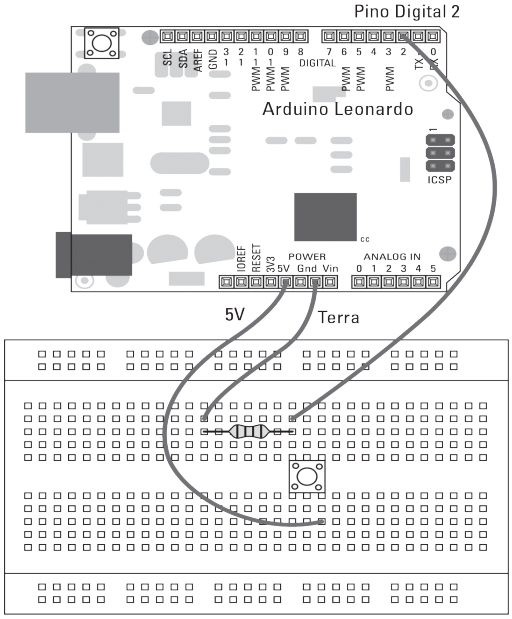

FIGURA CO-11: Um diagrama de circuito de um circuito de botão se comunicando com o pino digital 2.

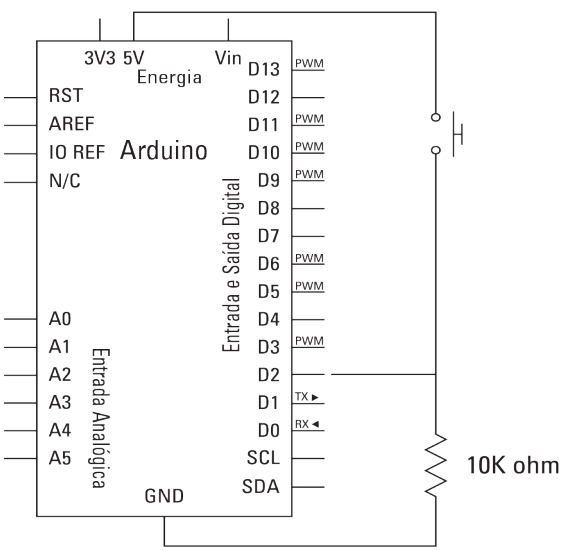

FIGURA CO-12: Um esquema do circuito de botão.

De um lado, o botão está conectado à fonte de 5V do seu Arduino. Do outro, ao pino 2 do Leonardo e também ao GND, através de um resistor de 10k.

Depois que seu circuito é montado, você precisa do software adequado para usá-lo. Na barra de menu do Arduino, clique em Arquivo ➪ Exemplos ➪ 09.USB ➪ Keyboard ➪ KeyboardMessage.

```
/*
 Teste sketch Keyboard Button

 Envia uma string de texto quando um botão é pressionado.

 O circuito:
 * botão ligado ao pino 2 de + 5V
 * Resistor de 10K ligado ao pino a partir do terra

 criado em 24 de outubro de 2011
 modificado em 27 de março de 2012 por
 Tom Igoe

 Este código de exemplo está em domínio público.

 http://www.arduino.cc/en/Tutorial/KeyboardButton (conteúdo em inglês)
 */

const int buttonPin = 2; // pino de entrada para botão
int previousButtonState = HIGH; // para verificar o estado de um botão
int contador = 0; // temporizador do botão

void setup() {
  // faz o pino do botão de entrada:
  pinMode(buttonPin, INPUT);
  // inicializa o controle do teclado:
  Keyboard.begin();
}

void loop() {
  // lê o botão:
  int buttonState = digitalRead(buttonPin);
  // se o estado do botão mudou,
  if ((buttonState != previousButtonState)
    // e atualmente é pressionado:
  && (buttonState == HIGH)) {
    // incrementa o temporizador de botões
    counter++;
    // digita uma mensagem
    Keyboard.print("You pressed the button ");
```

```
    Keyboard.print(counter);
    Keyboard.println(" times.");
  }
  // salva o estado do botão atual para comparação na próxima vez:
  previousButtonState = buttonState;
}
```

Depois de fazer o upload do sketch com sucesso, abra um editor de texto. Usando o mouse, confira se a parte de entrada de texto do aplicativo está selecionada e pressione o botão. Você deve ver uma linha de texto dizendo: "You pressed the button 1 times" [Você apertou o botão 1 vez]. A legibilidade do código não é lá aquelas coisas, mas é um excelente exemplo de uma emulação de teclado funcionando. Essa mensagem aparece sempre que o botão é pressionado, para mantê-lo informado sobre o número de vezes que isso acontece.

Observe que, se você desconectar sua placa Leonardo e conectá-la novamente, verá o LED do pino 13 marcado com "L" em efeito fader, enquanto a placa é inicializada. Durante este período, sua placa Leonardo não responde a entradas, e você pode receber respostas inconclusivas se alguém interagir com ela. É melhor esperar até que o efeito fader pare antes de usar a placa para evitar esses erros iniciais.

Se você não vir uma mensagem, confira seus fios e cabos:

- Confira se está usando o número de pinos correto.
- Confira as conexões na breadboard. Se os fios de jumper ou os componentes não estiverem conectados corretamente nas linhas da placa breadboard, eles não funcionarão.
- Verifique se o upload foi bem-sucedido e se a placa e a porta serial corretas estão selecionadas.

Compreendendo o sketch KeyboardMessage

Alguns comentários no sketch apresentam alguns erros que podem causar confusão.

A linha "* resistor de 10-quilohm conectado a partir do pino 4 ao terra" deve se referir ao pino 2, não ao pino 4.

O pino do botão recorrente é declarado como pino 2. Como esse sketch está monitorando quantas vezes o botão foi pressionado, é necessária uma variável para armazenar o estado do botão anterior. Como a primeira leitura será LOW, o valor anterior está definido como HIGH para começar. O valor do temporizador também é declarado e inicializado com um valor zero.

```
const int buttonPin = 2; // pino de entrada para botão
int previousButtonState = HIGH; // para verificar o estado de um
                                // botão
int contador = 0; // temporizador do botão
```

Na setup, o buttonPin é declarado como uma entrada.

```
void setup() {
  // faz o pino do botão de entrada:
  pinMode(buttonPin, INPUT);
```

A função específica da placa Leonardo, Keyboard.begin, é chamada para inicializar a comunicação como um teclado. Ela é semelhante ao modo como a Serial.begin() é chamada para inicializar a comunicação serial.

```
  // inicializa o controle do teclado:
  Keyboard.begin();
}
```

A primeira tarefa no loop é ler o estado do buttonPin. Em seguida, essa leitura é armazenada na variável local (temporária) buttonState, que é apagada na próxima vez que o sketch fizer um loop.

```
void loop() {
  // lê o botão:
  int buttonState = digitalRead(buttonPin);
```

Uma instrução if verifica se o atual buttonState não é igual a (!=), o previousButtonState, e (&&), se buttonState é igual ao valor (==) HIGH.

```
// se o estado do botão mudou,
if ((buttonState != previousButtonState)
  // e atualmente está pressionado:
```

```
&& (buttonState == HIGH)) {
```

Se essa instrução for true, o botão acabou de ser pressionado, logo, o temporizador é incrementado e a mensagem do teclado é exibida.

```
// incrementa o temporizador de botões
counter++;
```

A mensagem está dividida em três linhas. Uma linha de texto é seguida pelo valor numérico da variável do temporizador e, em seguida, uma linha de texto completa a frase e finaliza a linha. `Keyboard.println ()` indica um retorno de linha, iniciando uma linha nova para a próxima mensagem.

```
  // digita uma mensagem
  Keyboard.print("You pressed the button ");
  Keyboard.print(counter);
  Keyboard.println(" times.");
}
```

A instrução `if` termina e o valor atual se torna o valor anterior para o próximo loop.

```
  // salve o estado do botão atual para comparação na próxima vez:
  previousButtonState = buttonState;
}
```

Esse simples sketch interage de maneira rápida e fácil com uma interface de software em um computador. Ele pode ser utilizado para alguma coisa bem simples, como acertar a barra de espaço para avançar slides com um único botão ou para pressionar o botão que faz o mouse se movimentar, tornando seu PC um controlador de jogo.