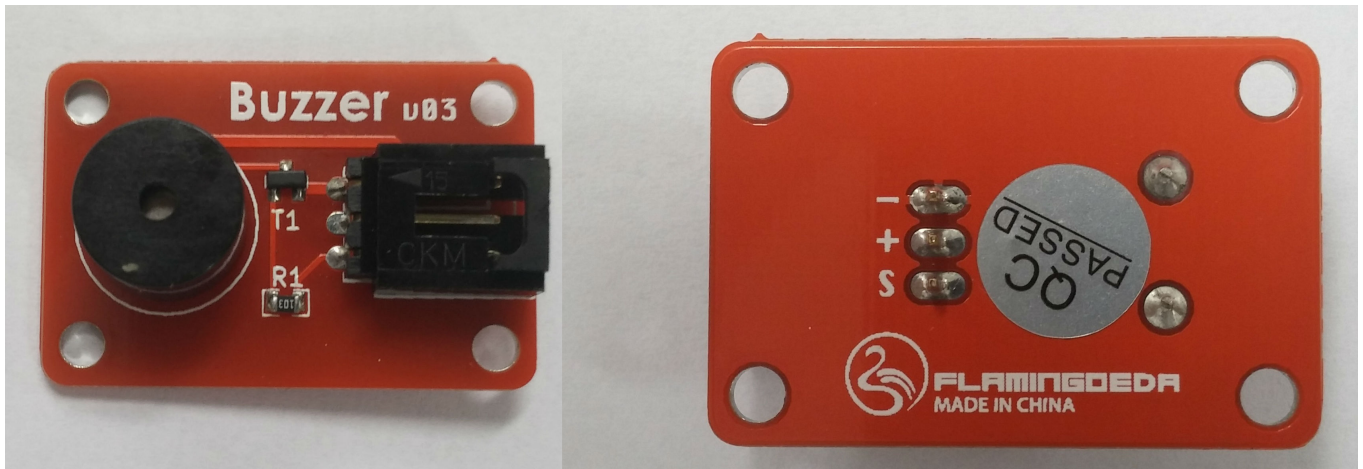
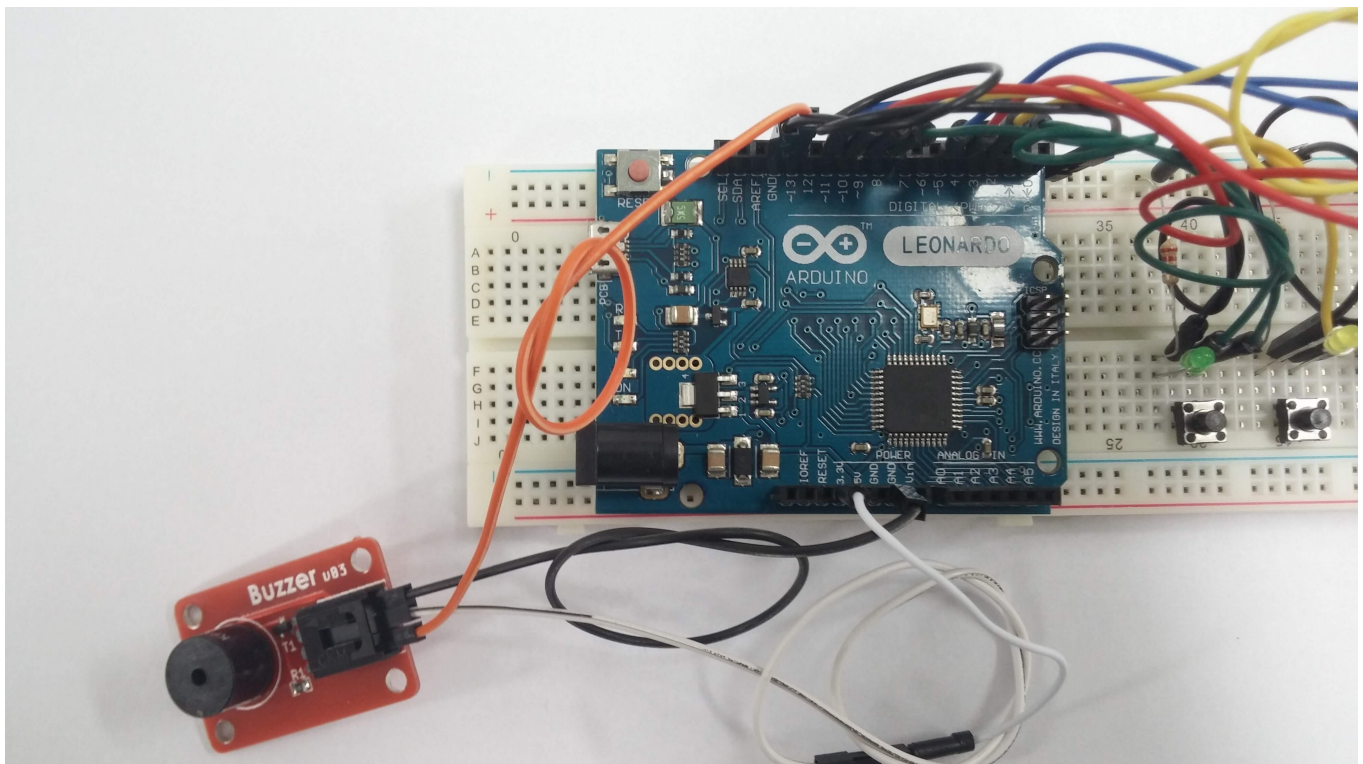


(Opcional) Colocando som no seu jogo

Para esse exercício, utilizaremos um buzzer que contém um pequeno circuito integrado, como o do modelo abaixo:



Repare que nele há 3 saídas. A positiva (+) nós iremos conectar no 5 volts, a negativa (-) no terra (GND) e a S ligaremos em uma porta digital, no nosso caso usaremos a 13:



Com o buzzer já conectado ao Arduino, vamos ao código!

1) Primeiro, criamos a constante referente ao buzzer:

```
#define BUZZER 13
```

2) Segundo, usamos o 13 porque o buzzer está ligado à porta 13 do Arduino. Agora, precisamos iniciar a porta do buzzer como OUTPUT, pois vamos emitir um som. Em `iniciaPortas()`, adicione:

```
pinMode(BUZZER, OUTPUT);
```

3) Vamos, agora, criar uma função chamada `tocaSom()`. Ela recebe a frequência do som por parâmetro e utiliza essa frequência para gerar um tom no buzzer. Portanto, a função `tone` recebe além do buzzer e da frequência, também o tempo de duração do som por parâmetro, que nós vamos colocar como 100 milissegundos:

```
void tocaSom(int frequencia) {  
    tone(BUZZER, frequencia, 100);  
}
```

4) Com a função criada, já é possível tocar um som através de uma frequência, basta chamar a `tocaSom()` ! Mas, quando exatamente queremos que o som seja tocado? Quando o LED piscar! Então, vamos lá na função `piscaLed()` e adicionamos a chamada da `tocaSom()` :

```
int piscaLed(int portaLed) {  
  
    tocaSom(2000);  
  
    digitalWrite(portaLed, HIGH);  
    delay(UM_SEGUNDO);  
    digitalWrite(portaLed, LOW);  
    delay(MEIO_SEGUNDO);  
  
    return portaLed;  
}
```

5) Agora, você pode compilar o código, enviá-lo ao Arduino e testar o som. Ao fazer isso repare que sempre é tocado o mesmo som! Isso ocorre porque só reproduzimos o tom da frequência 1000. Para melhorar nosso jogo podemos tocar um som distinto para cada LED do Arduino.

Por isso, vamos criar mais uma função e a chamaremos de `verificaSomDoLed()` . Essa função irá receber a porta do LED por parâmetro, verificar qual LED é e dependendo de qual for, um som diferente será tocado:

```
void verificaSomDoLed(int portaLed) {  
    switch (portaLed) {  
        case LED_VERDE:  
            tocaSom(2000);  
            break;  
        case LED_AMARELO:  
            tocaSom(2200);  
            break;  
        case LED_VERMELHO:  
            tocaSom(2400);  
            break;  
        case LED_AZUL:  
            tocaSom(2500);  
            break;  
    }  
}
```

6) Repare que cada LED terá um som em uma frequência distinta. Por isso, precisamos alterar a função `piscaLed()` para acrescentar a `verificaSomDoLed()` que acabamos de criar e não mais o `tocaSom()` :

```
int piscaLed(int portaLed) {  
  
    verificaSomDoLed(portaLed);  
  
    digitalWrite(portaLed,HIGH);  
    delay(calculaTempoLed());  
    digitalWrite(portaLed,LOW);  
    delay(MEIO_SEGUNDO);  
  
    return portaLed;  
}
```

7) Compile e envie o código do Arduino. Simule o jogo e veja... Não ficou melhor?!

8) Para finalizar, vamos adicionar mais um som, uma espécie de sirene para servir de alerta quando o jogo for finalizado com falha. Para isso basta adicionar **no começo** da função `jogoFinalizadoFalha()` :

```
void jogoFinalizadoFalha() {  
    tocaSom(300);  
    // Restante do código comentado  
}
```