Ligações de LEDs e Chaves

Os LEDs são as saídas básicas de um sistema microprocessado. Na forma digital (onde o que só interessa é se estão ligados ou não) podem sinalizar 2 estados para o usuário. As chaves *pushbutton* por sua vez são as entradas mais simples e o usuário pode pressioná-las para enviar uma informação para o sistema. A chave pode ser com ou sem retenção. É mais comum trabalhar com as chaves sem retenção pois o usuário não precisa se preocupar em retornar a um estágio inicial da chave.

Neste curso, estaremos usando a plataforma Arduino, que é composta dos microcontroladores Atmel AVR, uma placa que inclui um gravador USB e disponibilização fácil dos pinos do microcontrolador para uma montagem de hardware rápida e ferramentas de desenvolvimento em linguagem C.

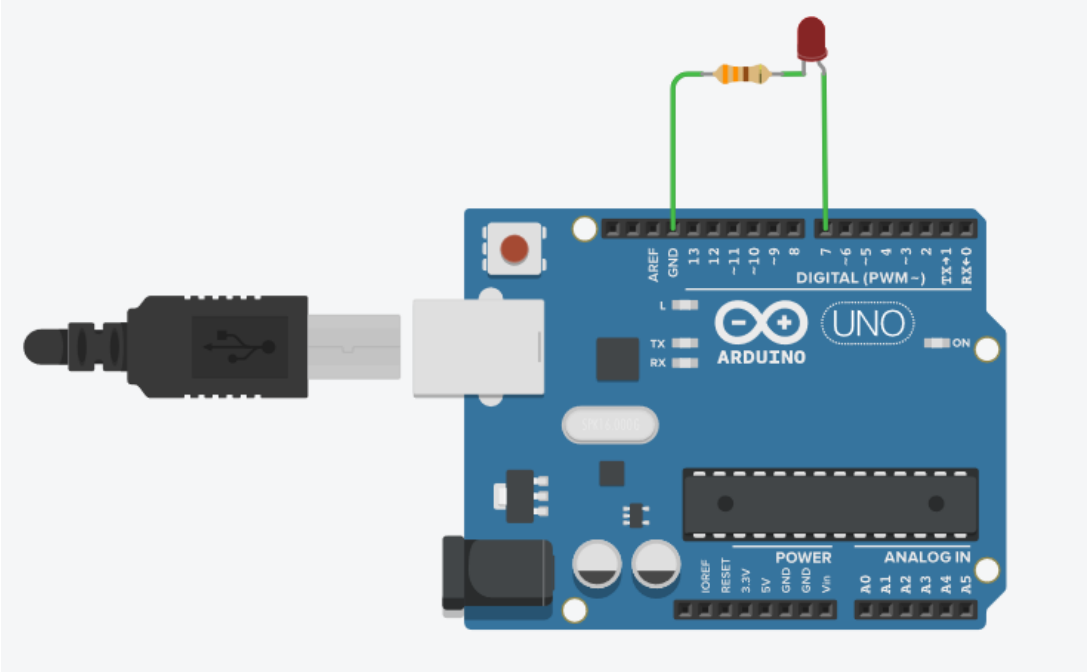
Portas digitais

Utilizaremos as portas digitais do Arduino. Estas portas digitais podem ser endereçadas individualmente (bit a bit) e podem ser de entrada ou de saída. Existem registradores internos ao microcontrolador capaz de manipular estas portas. No Arduino estes registradores são o DDR e o PORT e podem ter no máximo 8 bits. Os registradores DDR são responsáveis por armazenar a informação de sentido, ou seja, se os bits da porta são de entrada ou saída. Os registradores PORT espelham a saída a partir dos seus valores lógicos (1 equivale a 5V e 0 equivale a 0V)

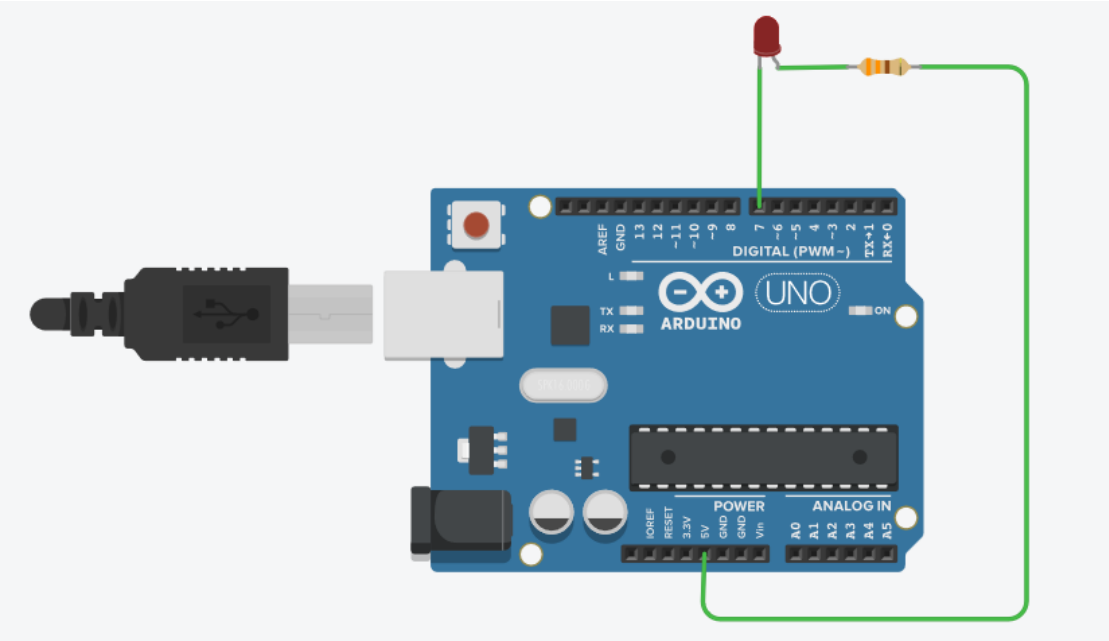
Como exemplo os pinos 0 a 7 do Arduino estão associados aos registradores DDRD e PORTD. Um valor binário de 01000101 no registrador DDRD indica que apenas os pinos 0 (0 o da direita até 7, o bit mais à esquerda), 2 e 6 são de saída e o restante de entrada. Da mesma forma, se um valor binário de 10110000 for encontrado no registrador PORTD isto significa que os pinos 4, 5 e 7 estão em 5V, os demais em 0V. Na manipulação dos displays de 7 segmentos, usaremos estes registradores.

Ligação de LEDs

Para ligarmos um LED ao microcontrolador, precisamos de um resistor para limitar a corrente pois sabemos que pela característica exponencial da curva IxV, um pequeno aumento da tensão pode causar um aumento muito alto da corrente. Podemos ligar o LED de duas formas: Acendendo quando o microcontrolador escrever 1 no pino que está ligado ao LED ou acendendo quando ele escrever 0. As figuras abaixo ilustram os dois tipos de ligação. Veja que no desenho gerado pelo simulador Tinkercad o catodo do LED é a linha reta (além disso tem o chanfro do lado deste terminal) dos terminais do LED.



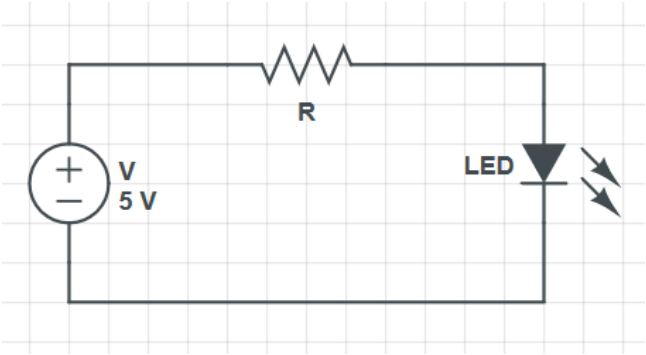
Ligação do LED para que ele acenda quando o microcontrolador escrever 1 no pino 7



Ligação do LED para que ele acenda quando o microcontrolador escrever 0 no pino 7

A alimentação do Arduino é de 5V, o que nos dá 0V na saída quando se escreve 0 lógico e 5V quando se escreve 1 lógico. Notem que não podemos trocar o sentido do anodo e catodo nas figuras pois senão teríamos uma polarização reversa do LED, o que não o faria acender em nenhuma situação. O LED só acenderá, em qualquer um dos casos, quando houver uma diferença de potencial entre os terminais do LED em que o catodo fique positivo em relação ao anodo.

Nos dois casos, quando o LED acende, podemos reduzir o circuito para o que está representado na figura abaixo. Neste caso, vamos usar uma corrente de 10mA (bem abaixo da corrente máxima que o Arduino pode oferecer em um pino digital) mas suficiente para gerar uma forte iluminação.



Circuito reduzido quando o LED acende

Para calcular o resistor, podemos usar a lei das malhas de Kirchhoff:

V = VR + VD

Onde V é a tensão da fonte, VR, é a tensão nos terminais do resistor e VD é a tensão nos terminais do LED. Um LED vermelho comum de 3mm ou 5mm tem uma queda de tensão aproximada de 1,4V (tensão de joelho).

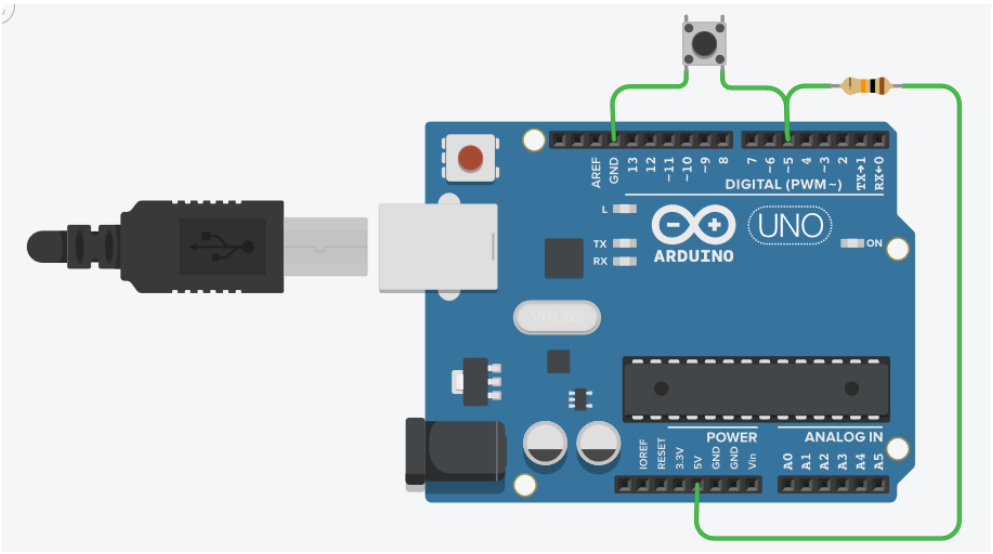
5V = R.(10mA) + 1,4V

De onde tiramos R = 360Ω

Como não há necessidade de precisão neste caso, podemos usar resistores com valores próximos a este. No caso, o valor escolhido nos exemplos foi 330Ω.

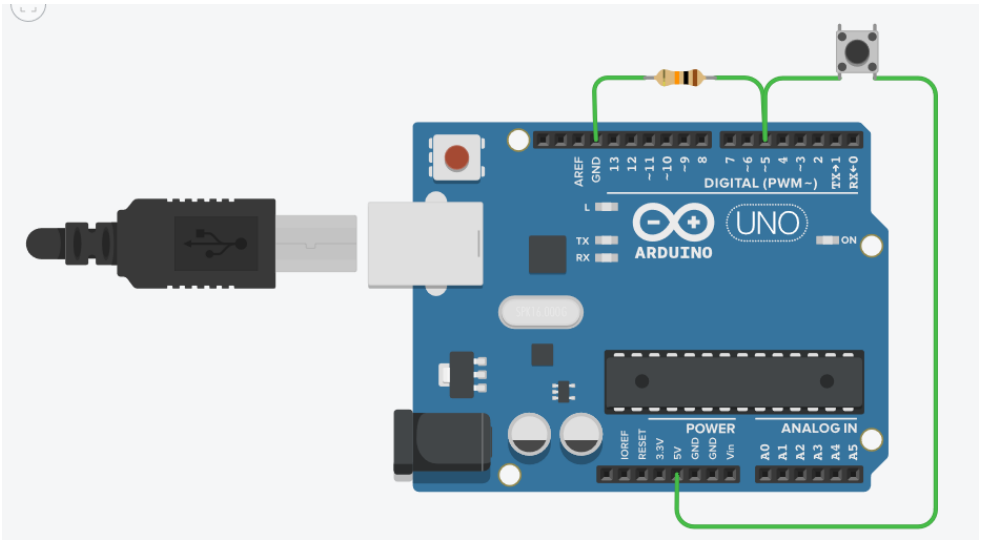
Ligação de Chaves

A ligação das chaves pode ser feita de duas formas, assim como os LEDs. Numa das formas, quando a chave é pressionada, entra 1 e a outra forma é a entrada de 0 quando a chave é pressionada. Nas figuras abaixo vemos essas ligações e também um resistor importante para este tipo de ligação.



Neste exemplo, quando a chave é pressionada, o pino 5 recebe GND (0V)

Na figura acima, a chave *pushbutton* não tem retenção e quando ela é pressionada, há o curto entre seus terminais (terminal 5 do Arduino e GND) o que faz com que o GND seja ligado ao terminal 5. Quando ela não está pressionada o pino 5 recebe o nível lógico 1 (5V) através do resistor. Veja que o resistor é importante pois quando a chave é pressionada, haveria um caminho sem resistência entre o GND e o 5V do Arduino. Este resistor é chamado de pull-up. O Arduino tem pull-ups internos que podem ser configurados por programação.

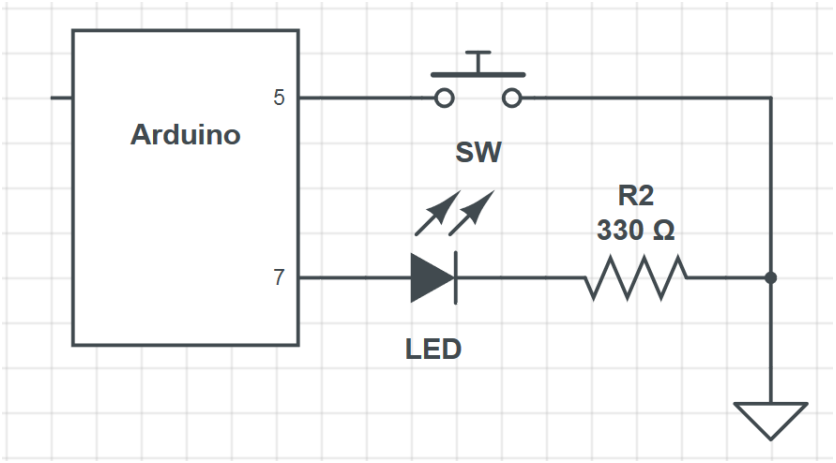


Neste exemplo, quando a chave é pressionada, o pino 5 recebe 5V

A figura acima mostra a ligação da chave quando queremos que ela envie 1 (5V) quando pressionada. O resistor agora se chama pull-down e tem o mesmo objetivo do anterior já explicado: Não deixa que haja curto entre 5V e GND quando a chave é pressionada e disponibiliza um caminho para o GND quando a chave não está pressionada. Um bom valor para estes registradores é de 10KΩ.

Exemplos

O exemplo abaixo mostra um LED ligado a uma saída do Arduino (pino 7) e a chave pushbutton ligado a uma entrada (pino 5). O programa será feito para que quando a chave for pressionada, o LED acenda. Caso contrário o LED será apagado.



Exemplo - LED e chave ligados em um Arduino

Programa:

```
#define LED 7
#define SW  5

void setup() {
  pinMode (LED, OUTPUT);
  pinMode (SW, INPUT_PULLUP);
}

void loop() {
  if (digitalRead (SW) == LOW) // a chave foi pressionada
    digitalWrite (LED, HIGH);
  else
    digitalWrite (LED, LOW);
}
```

Os primeiros comandos deste programa são diretivas para o compilador *#define*. Esta diretiva faz com que toda vez que o compilador encontrar LED, substitua por 7 (pino ao qual está ligado) e quando encontrar SW, substitua por 5. É sempre importante usar essas diretivas para o nosso circuito para que o programa fique bem legível e além disso caso troquemos um pino de algum componente, basta alterar no *#define*, sem a necessidade de percorrer todo o programa para as substituições.

Em *setup()* colocamos o sentido dos pinos. LED está como *OUTPUT* e SW está como *INPUT_PULLUP*. Como foi falado antes, os pinos digitais têm resistores *pull-up* associados a eles. Olhando o hardware, vemos que o LED acende quando a saída 7 vai para 1 e que a chave quando pressionada envia 0 e se não está pressionada, entra 1 pelo resistor interno de *pull-up*.

Na função *loop()* a função de biblioteca *digitalRead()* lê um pino e retorna *HIGH* ou *LOW* (definidos como 1 e 0 respectivamente no Arduino). Neste caso, lê o pino SW (pino digital 5) e se o valor for zero, a chave está pressionada naquele momento. Estando pressionada, o LED é aceso pois a função *digitalWrite()* escreve no pino LED (7) o valor *HIGH* e isso se traduz em colocar 5V no pino do LED e se não estiver pressionada, é escrito *LOW* no pino 7, apagando o LED.

Veja o vídeo explicativo deste exemplo