

Usando um transistor para controlar mais de um LED

Em nosso projeto do *Cybertrem* usamos 4 transistores, cada um para controlar 4 LEDs colocados em nosso protótipo.

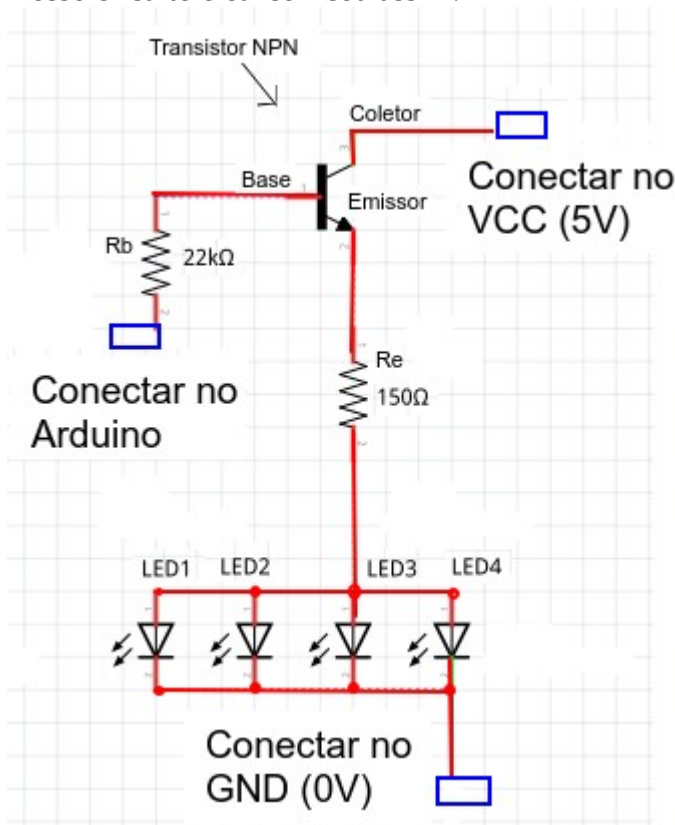
Fizemos isso, pois o Arduino não possui capacidade de controlar correntes maiores do que 40mA (quarenta miliamperes) em cada um de seus pinos (nem possui capacidade total de mais do que 200mA, duzentos miliamperes). O transistor, então, controla a corrente vinda da interface USB do computador (ou da bateria que usamos nas demonstrações) e age como uma espécie de chave: quando o transistor recebe um sinal positivo (ou alto, HIGH) em sua base, ele permite a circulação da corrente elétrica entre seu coletor e seu emissor. Isto é o funcionamento para o transistor do tipo NPN (e sabemos que existe também o tipo PNP). Para este funcionamento o transistor exige muito pouca corrente do Arduino, o que permite um funcionamento seguro.

Existem transistores capazes de controlar grandes quantidades de corrente. Em nosso caso, a quantidade é relativamente pequena, de poucos miliamperes, de forma que utilizamos um transistor de pequena potência e de uso geral (o disponível, e que foi utilizado, foi o 2N2222).

Em nosso experimento, o transistor utilizado recebe o apoio de dois componentes do tipo resistor: um em sua base (R_b) e outro em seu emissor (R_e). Na sequência, veremos como eles são calculados. Por enquanto, vamos ver as ligações.

Estamos trabalhando agora com o circuito eletrônico associado à nossa solução, não com o *protoboard*. Desta forma, veremos os símbolos dos componentes e iremos efetuar alguns cálculos, para que vocês possam reproduzir outros circuitos quando necessário.

Nosso circuito elétrico ficou assim:

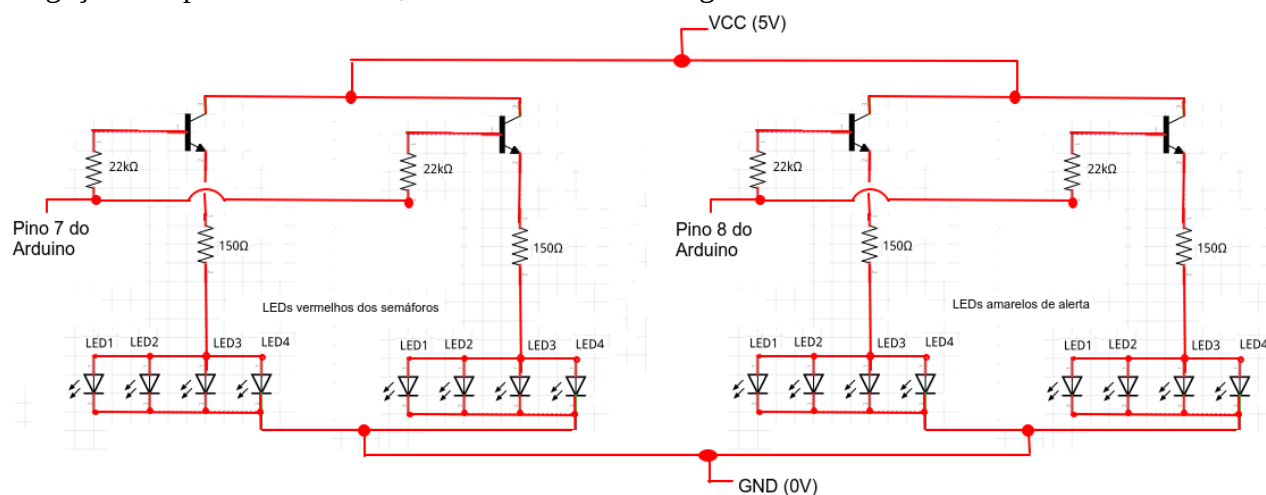


Desta forma, o transistor recebe um comando vindo do Arduino, o qual passa pelo resistor que está ligado em sua base e, a partir das características de seu funcionamento elétrico, permite a passagem

de corrente entre seu coletor e seu emissor. O coletor está ligado ao +5V (cinco volts positivos) da placa do Arduino. O emissor, por meio de um resistor limitador de corrente, está ligado a 4 LEDs, ligados em paralelo, os quais são ligados ao 0V (zero volts, ou *ground*, GND), completando o circuito elétrico.

Este circuito foi repetido 4 vezes: 2 foram usados para controlar os LEDs dos semáforos (dois semáforos de 4 LEDs vermelhos cada) e 2 foram usados para controlar os LEDs amarelos que ficam na pista para alerta (4 LEDs de cada lado, cada conjunto de 4 LEDs controlado por um transistor).

A ligação completa ficou assim, com os transistores ligados 2 a 2:



Lembrem-se de que as conexões estão representadas assim: 

E, quando não existe conexão, somente um fio passando por cima do outro, está representado assim: 

Os transistores cujas bases estão ligadas ao pino digital 7 do Arduino controlam os semáforos. Os transistores cujas bases estão ligadas ao pino digital 8 do Arduino controlam os LEDs amarelos de alerta. A escolha dos pinos do Arduino foi aleatória. Se quiséssemos controlar os semáforos individualmente poderíamos conectar as bases dos transistores separadamente, cada uma a um pino, por exemplo, um deles no pino 7 e o outro no pino 6 (obviamente, teríamos que agir no código também, e não somente nas ligações). O mesmo poderia ser feito para os transistores que controlam os LEDs amarelos de alerta.

Mas, de onde vieram os valores dos resistores?

Novamente, como o circuito não é extremamente crítico, e considerando-se a existência de uma certa tolerância dos componentes, foram utilizados os componentes que estavam disponíveis no momento da montagem. Porém, houve um cálculo prévio, para que os valores ideais fossem determinados, para que pudessem ser utilizados como parâmetros. Foi feito como segue.

Existe um resistor ligado no emissor do transistor, identificado como R_e , o qual controla a quantidade de corrente que poderá passar pelos LEDs. E, um resistor ligado na base do transistor, identificado como R_b , o qual controla a quantidade de corrente necessária para acionar o transistor. Eles foram calculados de forma simplificada, seguindo o seguinte raciocínio:

Primeiramente, como os LEDs estão ligados em paralelo, as correntes e tensões que os percorrem são as mesmas (já que eles possuem as mesmas características). Assumimos que a tensão que os LEDs precisam para funcionar é de 1,7V (um vírgula sete volts), e, que a corrente que vai circular pelo conjunto de LEDs será de 30mA (trinta miliamperes - ou 0,03A, zero vírgula zero 3 amperes). Estas informações podem ser buscadas nas folhas de dados (*datasheets*) fornecidas pelo fabricantes dos componentes.

Quando um transistor de silício NPN, como o utilizado, está em funcionamento, existe uma queda de tensão entre seus terminais de coletor e emissor, de 0,7V (zero vírgula sete volts).

A queda de tensão no LED (1,7V) e a do transistor (0,7V) serão somadas, e reduzidas da tensão de alimentação extraída da placa do Arduino (que é de 5V). Esta diferença, será dividida pela quantidade de corrente desejada. O resultado será o valor da resistência a utilizar.

Vamos ver os números em uma equação:

Resistor de emissor (R_e) = [Tensão do Arduino – (queda no LED + queda no transistor)] / corrente desejada

$$R_e = [5V - (1,7V + 0,7V)] / 0,03A \quad (30mA = 0,03A)$$

$$R_e = [5V - (2,4V)] / 0,03A$$

$$R_e = [2,6V] / 0,03A$$

$$R_e = 87\Omega$$

O valor encontrado, 87Ω , é o valor do resistor a ser ligado entre o emissor do transistor e o conjunto de LEDs. Mas, este valor não é um valor de mercado; comercialmente, como valores mais próximos, temos 82Ω e 91Ω . Isto ocorre por não ser comercialmente viável entregar todos os valores possíveis de serem calculados. E, porque podemos realizar associação de resistores de forma a chegar no valor desejado. E, ainda, porque os componentes possuem tolerância quanto a seus valores nominais.

O valor pelo qual é identificado um componente é chamado de valor nominal. O valor real do componente depende ainda de sua tolerância. Ou seja, ele pode não ser exatamente o valor nominal. Vamos assumir que o resistor possua uma tolerância de 5%. Isto quer dizer que ele poderá ter um valor até 5% menor, ou até 5% maior do que seu valor nominal. Normalmente a tolerância é especificada assim: $\pm 5\%$ (5% para mais ou para menos – existem valores de tolerância comuns no mercado de 1%, 2%, 5%, 10% e 20%).

Vamos fazer a conta para o resistor comercial de 82Ω : calculamos 5% dividindo o 5 por 100, o que dá $5/100 = 0,05$. Depois, calculamos 0,05 de 82Ω , o que é realizado multiplicando o 0,05 por 82, o que resulta em 4,1. Finalmente, subtraímos (-5%) e somamos (+5%) este valor obtido para a tolerância ao valor nominal. Teremos: $82 - 4,1 = 77,9\Omega$, e, $82 + 4,1 = 86,1\Omega$. portanto, um resistor de 82Ω com tolerância de $\pm 5\%$ poderá ter, na prática, qualquer valor de resistência entre $77,9\Omega$ e $86,1\Omega$.

O mesmo cálculo pode ser realizado para o resistor de valor comercial de 91Ω . Para $\pm 5\%$ de tolerância (que é a mais comum), teremos valores entre $86,45\Omega$ e $95,55\Omega$.

Notem que o maior valor do resistor de valor comercial mais baixo (82Ω) é de $86,1\Omega$, o que é muito próximo do menor valor do resistor de valor mais alto (91Ω), que é de $86,45\Omega$. E, que ambos são muito próximos do valor calculado, que é de 87Ω .

Se usarmos um resistor de valor menor, a corrente aumenta. Se usarmos um resistor de valor maior, a corrente diminui. Por uma questão de segurança, vamos usar o de valor comercial maior, que é o de 91Ω (veja na plaquinha que, na verdade, foram utilizado resistores de 150Ω , por estarem disponíveis, e por estarem acima do valor calculado – o que implica em corrente menor, mas ainda funcional).

Agora, vamos calcular o valor do resistor da base (R_b), seguindo a equação:

$$\text{Resistor de base } (R_b) = [\text{Tensão do Arduino} - \text{queda no transistor}] / \text{corrente de base } (I_b)$$

A queda no transistor é de 0,7V; a tensão do Arduino é

A corrente de base (I_b) é calculada como $I_b = \text{Corrente de coletor } (I_c) / \beta$ (letra grega *beta*, que representa o ganho do transistor)

Aqui, cabem observações: na eletrônica, usualmente, indicamos os resistores e/ ou valores de resistência com a letra R, a tensão com E (ou V) e a corrente com I. A resistência, é medida em ohms (com símbolo Ω , que é a letra grega ômega maiúscula); a tensão, é medida em volts (símbolo V); e, a corrente é medida em amperes (símbolo A). Não é incomum termos valores múltiplos e submúltiplos, como, por exemplo: k Ω para quiloohms (milhares de ohms – notem que o ‘k’ é minúsculo), M Ω para megaohms (milhões de ohms – notem que o ‘M’ é maiúsculo), mA para miliamperes (milésimos de ampere – notem que o ‘m’ é minúsculo). Outra observação importante refere-se à questão do ‘ganho’ do transistor, que é identificado pela constante beta (e representado pela letra β , que é a letra grega beta). A expressão ganho tem a ver com o funcionamento do componente, e refere-se ao fato de que, para uma pequena corrente de base, o transistor é capaz de manipular uma grande corrente de coletor. O valor do β é extraído da folha de especificações fornecida pelo fabricante do componente, usualmente chamada de *datasheet* (folha de dados).

Voltando aos cálculos, já que nós calculamos o valor do resistor de emissor para uma corrente de 30mA nos LEDs, e como esta corrente tem que passar pelo coletor do transistor, vamos assumir que a corrente de coletor será de 30mA. Já para o valor de β , conforme o fabricante ele varia de 100 a 800, sendo o valor típico de 150. Assumiremos este 150 e teremos:

Corrente de base (I_b) = Corrente de coletor (I_c) / β
Corrente de base (I_b) = 0,03 / 150
Corrente de base (I_b) = 0,0002 A

Notem que 0,0002A são 0,2mA (zero vírgula dois miliamperes), ou, 200uA (duzentos microamperes), que são 200 milionésimos de ampere. Portanto, fica evidenciado o ‘ganho’ do transistor: com apenas 0,2mA extraídos dos pinos do Arduino ele vai comandar 30mA para os LEDs (150 vezes mais, ou, um ‘ganho’ de corrente de 150 vezes). Por isso usamos o transistor, para evitar retirar dos circuitos internos do Arduino correntes muito elevadas (que podem ser ainda maior no caso de motores, por exemplo).

Continuando...

Resistor de base (R_b) = [Tensão do Arduino – queda no transistor] / corrente de base (I_b)
 R_b = [5V – 0,7V] / 0,0002A
 R_b = [4,3V] / 0,0002A
 R_b = 21500 Ω

Finalmente, temos o valor de R_b , que será de 21500. Os valores comerciais mais próximos são os de 20000 Ω (20k Ω) e 22000 Ω (22k Ω) – com as questões de tolerância já comentadas. Ficaremos com o valor de 22k Ω (no comércio, este valor pode ser pedido como ‘vinte e dois k’).

Volte à primeira figura, na primeira folha, e veja que, no circuito desenhado, foram utilizados estes valores escolhidos – resistor de base de 22k Ω , resistor de emissor de 150 Ω , quatro LEDs ligados em paralelo e um transistor NPN tipo 2N2222.

Um último cálculo: quando a corrente elétrica circula em um circuito ela causa uma dissipação de calor, advinda do cálculo de potência. Os resistores que usualmente utilizamos em nossos projetos possuem uma potência máxima de operação de 1/4W (um quarto de Watt, ou 0,25W). portanto, não podemos superar esta potência se quisermos que o resistor opere dentro de sua condição ideal.

Para calcular a potência que o resistor deverá dissipar quando a corrente prevista circular por ele, faremos assim:

Potência do resistor = Valor da resistência * (Valor da corrente)²

Potência do resistor de emissor = 150 Ω * (0,03A)²
(0,03A)² é o mesmo que 0,03A * 0,03A, que é 0,0009A
Potência do resistor de emissor = 150 Ω * 0,0009A
Potência do resistor de emissor = 0,135W

Potência do resistor de base = 22000 Ω * (0,0002A)²

Potência do resistor de base = $22000\Omega * 0,00000004A$

Potência do resistor de base = $0,00088W$

Em ambos os casos a potência a ser dissipada estará abaixo do valor permitido pelo resistor, pois $\frac{1}{4}W$ corresponde a $0,25W$. No comércio, temos resistores de $\frac{1}{8}W$, $\frac{1}{4}W$, $\frac{1}{2}W$, $1W$, $2W$, encontrados mais comumente. E, valores acima disto como $5W$, $10W$ e $20W$; ou, muito acima, mais difíceis de encontrar. Os mais comuns são os de $\frac{1}{8}W$, $\frac{1}{4}W$ e $\frac{1}{2}W$. O de $\frac{1}{4}W$, que usamos em nosso circuito, é um dos mais utilizados com Arduino, pois seu valor atende à maioria dos circuitos que são geralmente propostos.

NÃO SE ASSUSTEM COM OS CÁLCULOS. A maioria das aplicações utiliza valores padrão; muitos cálculos podem ser realizados em páginas da internet, emplanilhas ou outros programas de auxílio; muitas vezes vamos utilizar módulos, que já vem com ‘tudo pronto’. Aqui, colocamos e discutimos os cálculos para que vocês possam ir além do trivial.

Este tipo de cálculo, de corrente, tensão e resistência, faz parte dos conteúdos de aprendizagem que serão vistos nas aulas de física no ensino médio (ou, para quem quiser fazer cursos técnicos, em especial nos cursos de eletrotécnica, eletrônica e telecomunicações).