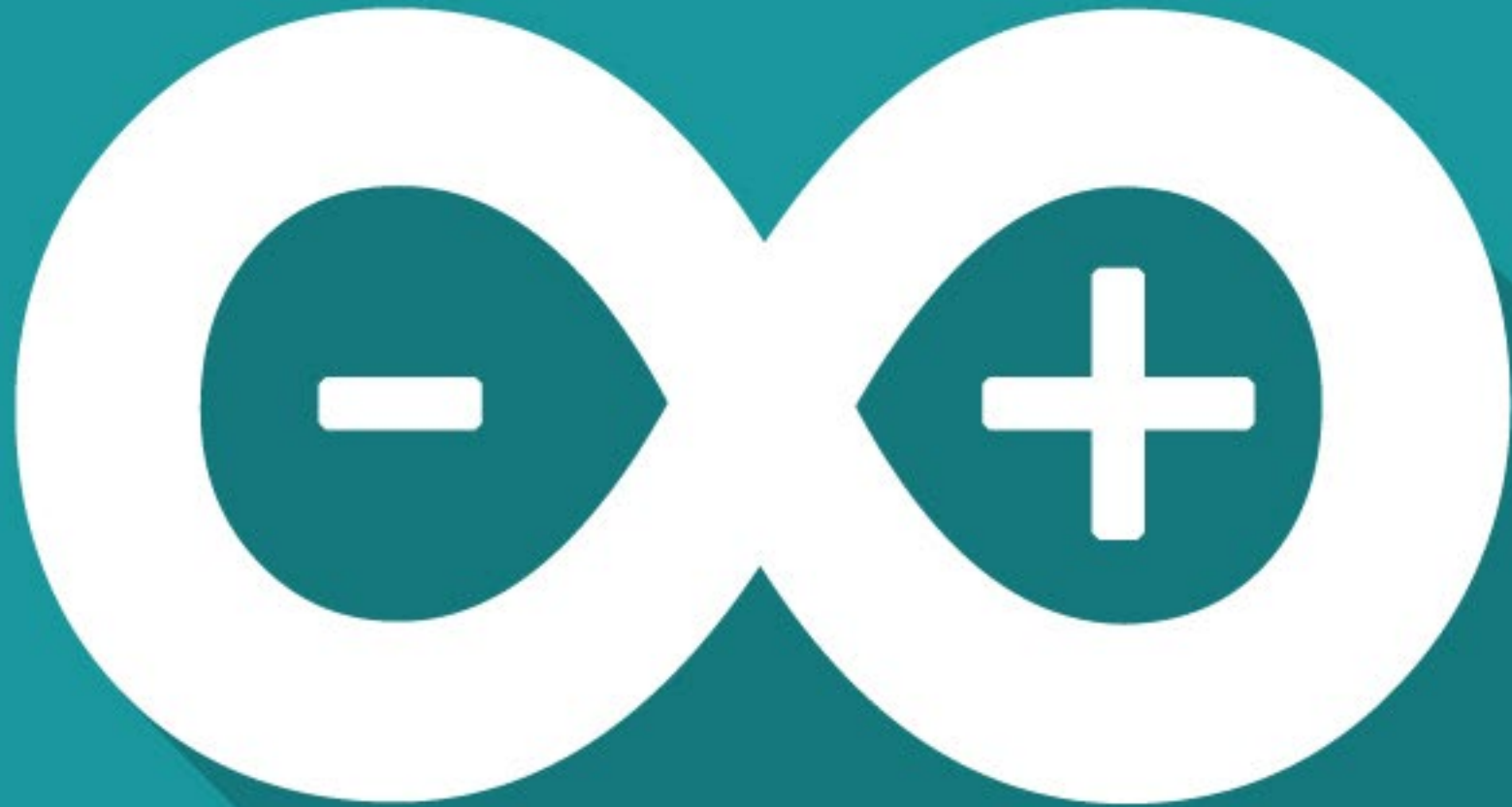
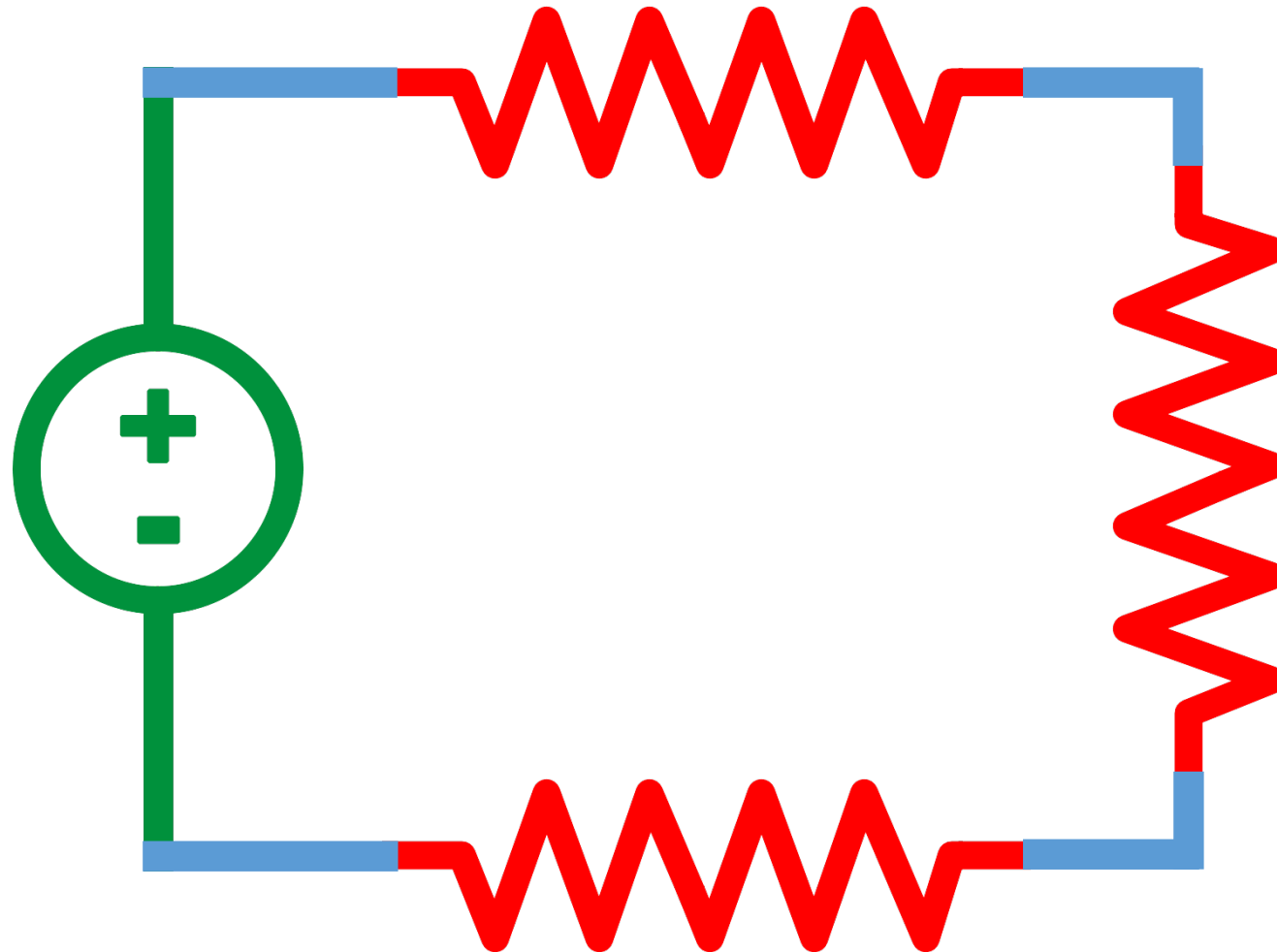


ARDUINO

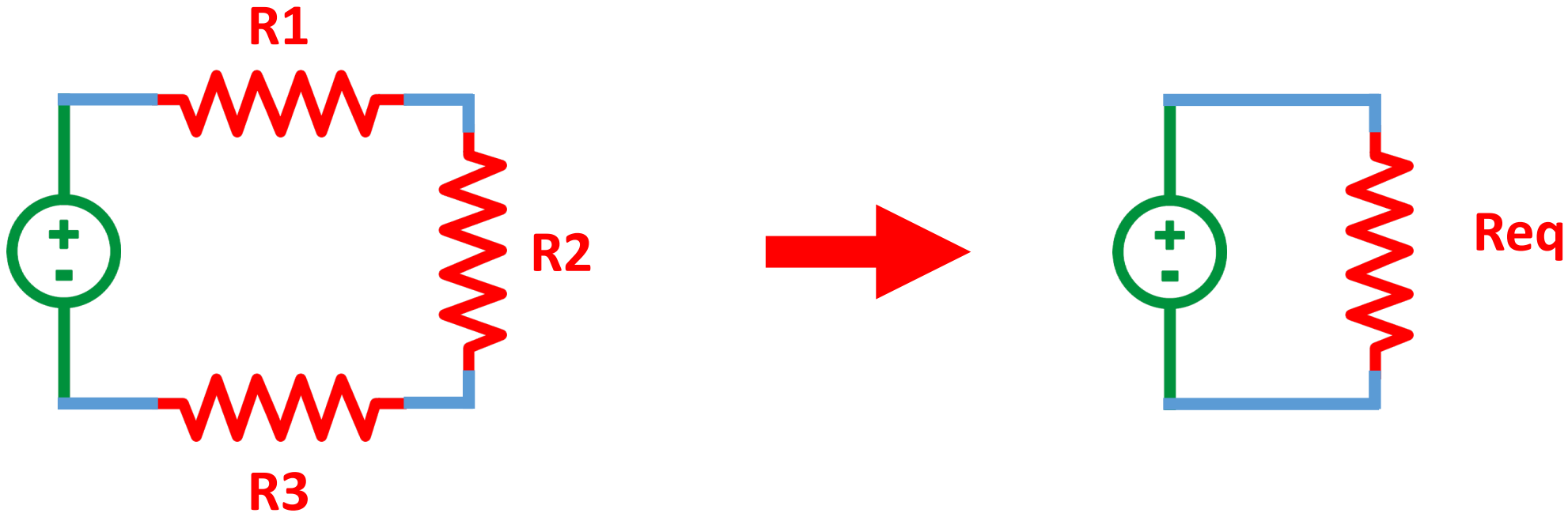


SESIÓN 02

RESISTENCIAS EN SERIE

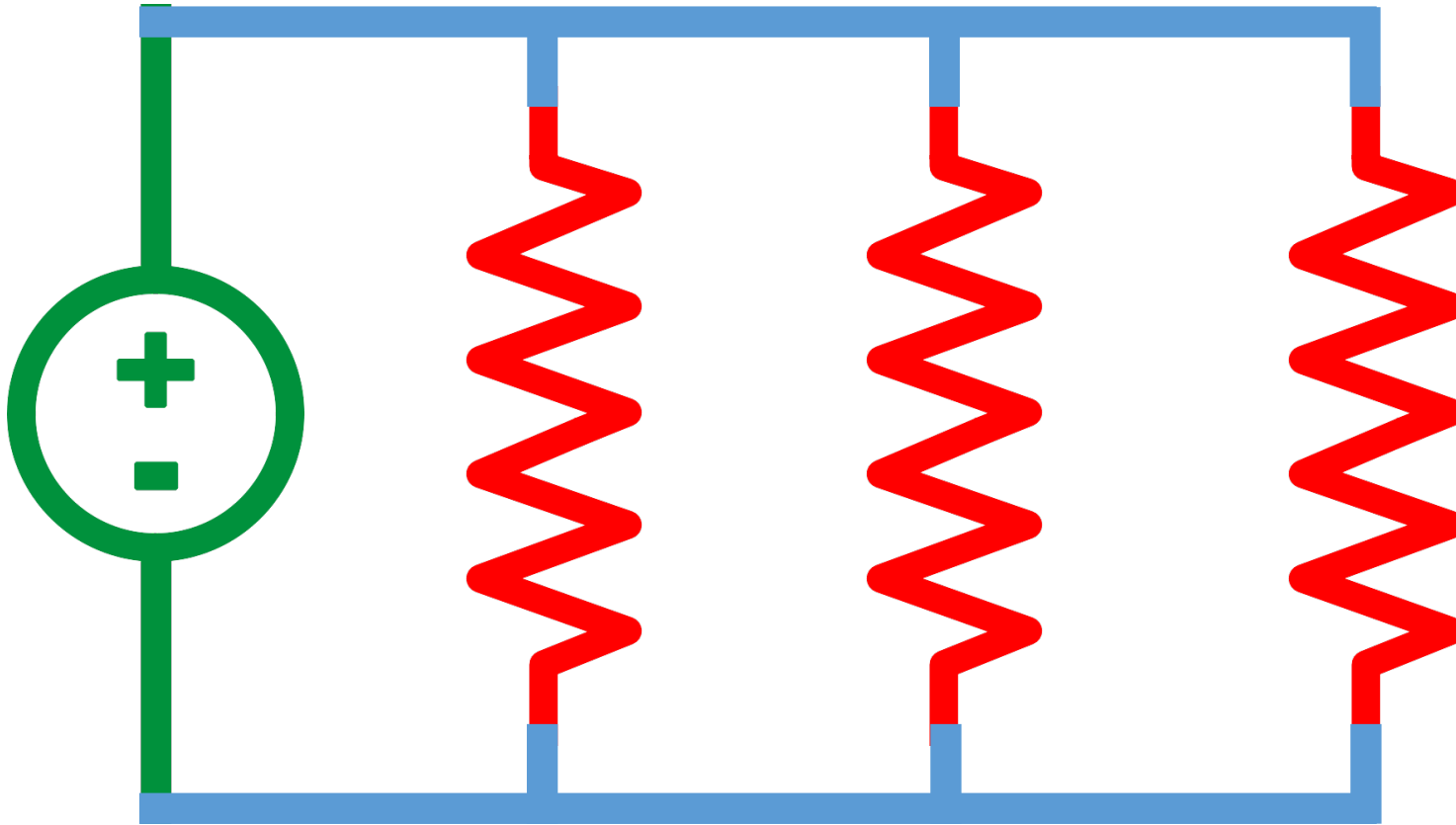


RESISTENCIAS EN SERIE

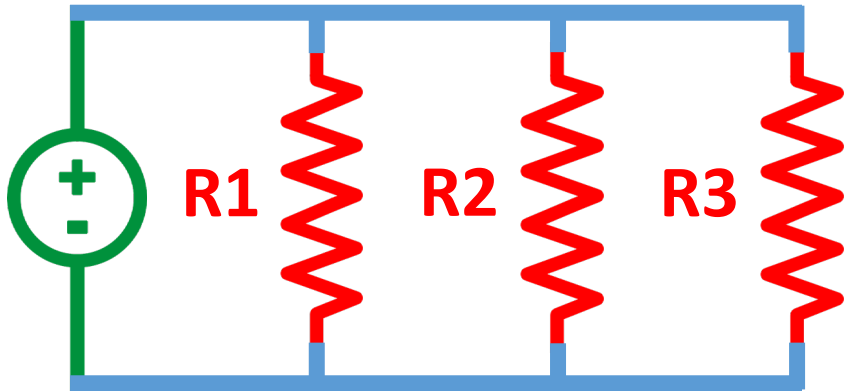


$$R_{eq} = R1 + R2 + R3$$

RESISTENCIAS EN PARALELO



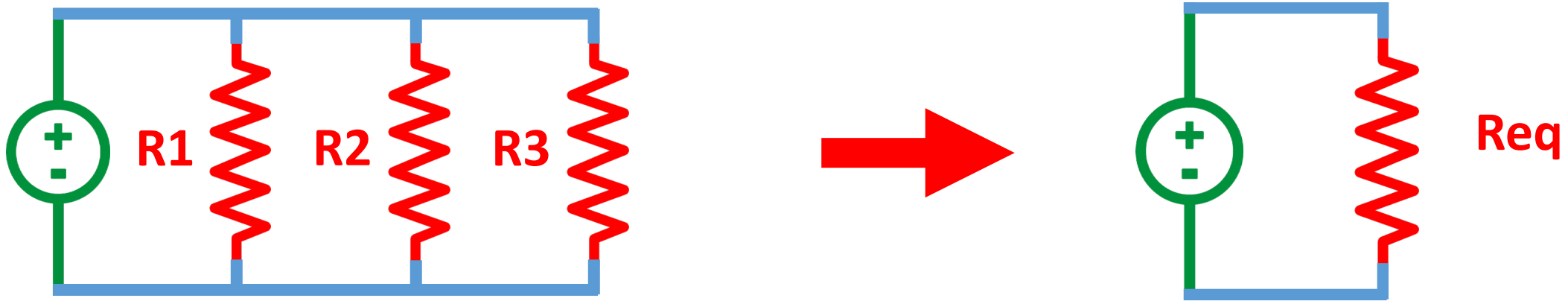
RESISTENCIAS EN PARALELO



$$\frac{1}{R_{eq}} = \frac{1}{R1} + \frac{1}{R2} + \frac{1}{R3}$$

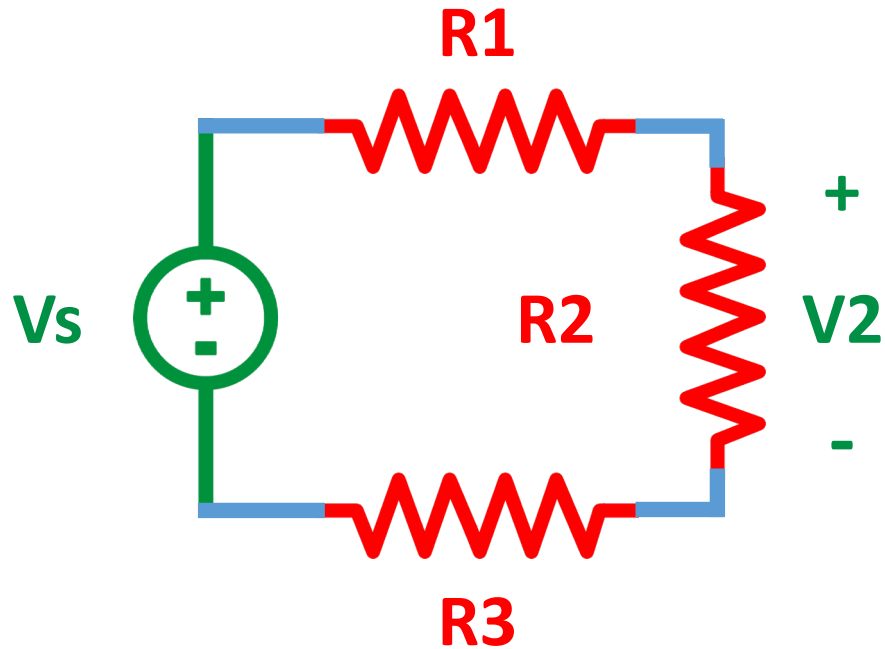
$$\frac{1}{R_{eq}} = \frac{R2 \ R3 + R1 \ R3 + R1 \ R2}{R1 \ R2 \ R3}$$

RESISTENCIAS EN PARALELO



$$R_{eq} = \frac{R1 \ R2 \ R3}{R2 \ R3 + R1 \ R3 + R1 \ R2}$$

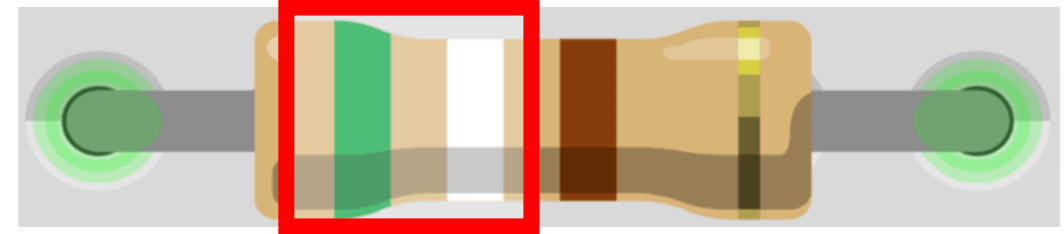
DIVISOR DE TENSIÓN



$$V_2 = \frac{V_s R_2}{R_1 + R_2 + R_3}$$

CÓDIGO DE COLORES RESISTENCIAS

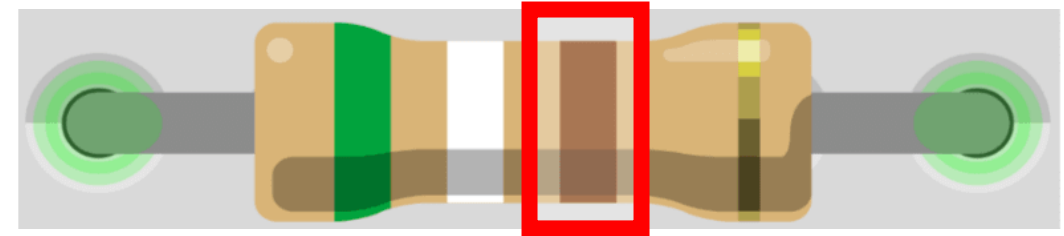
COLOR	VALOR 1	VALOR 2	MULTIPLICADOR	TOLERANCIA
NEGRO	0	0	1Ω	± 1%
MARRÓN	1	1	10Ω	± 2%
ROJO	2	2	100Ω	
NARANJA	3	3	1KΩ	
AMARILLO	4	4	10KΩ	
VERDE	5	5	100KΩ	± 0.5%
AZUL	6	6	1MΩ	± 0.25%
VIOLETA	7	7	10MΩ	± 0.10%
GRIS	8	8		± 0.05%
BLANCO	9	9		
ORO			0.1Ω	± 5%
PLATA			0.01Ω	± 10%



Las primeras dos bandas establecen el valor del resistor.

CÓDIGO DE COLORES RESISTENCIAS

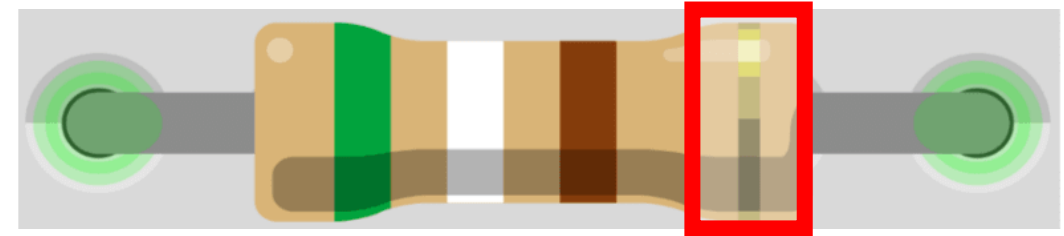
COLOR	VALOR 1	VALOR 2	MULTIPLICADOR	TOLERANCIA
NEGRO	0	0	1Ω	± 1%
MARRÓN	1	1	10Ω	± 2%
ROJO	2	2	100Ω	
NARANJA	3	3	1KΩ	
AMARILLO	4	4	10KΩ	
VERDE	5	5	100KΩ	± 0.5%
AZUL	6	6	1MΩ	± 0.25%
VIOLETA	7	7	10MΩ	± 0.10%
GRIS	8	8		± 0.05%
BLANCO	9	9		
ORO			0.1Ω	± 5%
PLATA			0.01Ω	± 10%



La tercera banda es el multiplicador, esta es la que nos indica los ceros al final.

CÓDIGO DE COLORES RESISTENCIAS

COLOR	VALOR 1	VALOR 2	MULTIPLICADOR	TOLERANCIA
NEGRO	0	0	1 Ω	$\pm 1\%$
MARRÓN	1	1	10 Ω	$\pm 2\%$
ROJO	2	2	100 Ω	
NARANJA	3	3	1K Ω	
AMARILLO	4	4	10K Ω	
VERDE	5	5	100K Ω	$\pm 0.5\%$
AZUL	6	6	1M Ω	$\pm 0.25\%$
VIOLETA	7	7	10M Ω	$\pm 0.10\%$
GRIS	8	8		$\pm 0.05\%$
BLANCO	9	9		
ORO			0.1 Ω	$\pm 5\%$
PLATA			0.01 Ω	$\pm 10\%$



La cuarta banda es la tolerancia, es decir, el porcentaje en el cual varía el valor de la resistencia.

EJERCICIOS

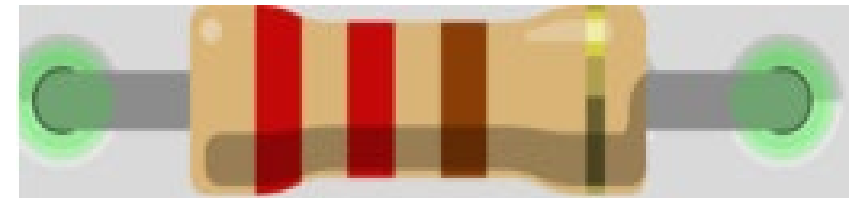
COLOR	VALOR 1	VALOR 2	MULTIPLICADOR	TOLERANCIA
NEGRO	0	0	1 Ω	$\pm 1\%$
MARRÓN	1	1	10 Ω	$\pm 2\%$
ROJO	2	2	100 Ω	
NARANJA	3	3	1K Ω	
AMARILLO	4	4	10K Ω	
VERDE	5	5	100K Ω	$\pm 0.5\%$
AZUL	6	6	1M Ω	$\pm 0.25\%$
VIOLETA	7	7	10M Ω	$\pm 0.10\%$
GRIS	8	8		$\pm 0.05\%$
BLANCO	9	9		
ORO			0.1 Ω	$\pm 5\%$
PLATA			0.01 Ω	$\pm 10\%$



100 [Ω]

EJERCICIOS

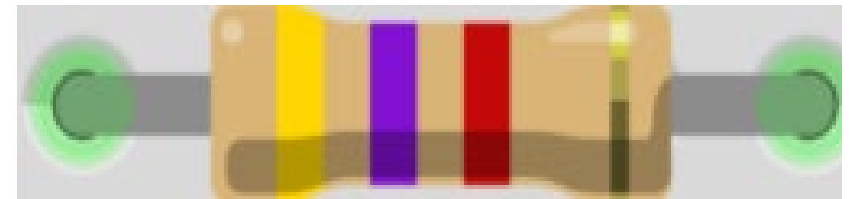
COLOR	VALOR 1	VALOR 2	MULTIPLICADOR	TOLERANCIA
NEGRO	0	0	1 Ω	$\pm 1\%$
MARRÓN	1	1	10 Ω	$\pm 2\%$
ROJO	2	2	100 Ω	
NARANJA	3	3	1K Ω	
AMARILLO	4	4	10K Ω	
VERDE	5	5	100K Ω	$\pm 0.5\%$
AZUL	6	6	1M Ω	$\pm 0.25\%$
VIOLETA	7	7	10M Ω	$\pm 0.10\%$
GRIS	8	8		$\pm 0.05\%$
BLANCO	9	9		
ORO			0.1 Ω	$\pm 5\%$
PLATA			0.01 Ω	$\pm 10\%$



220 [Ω]

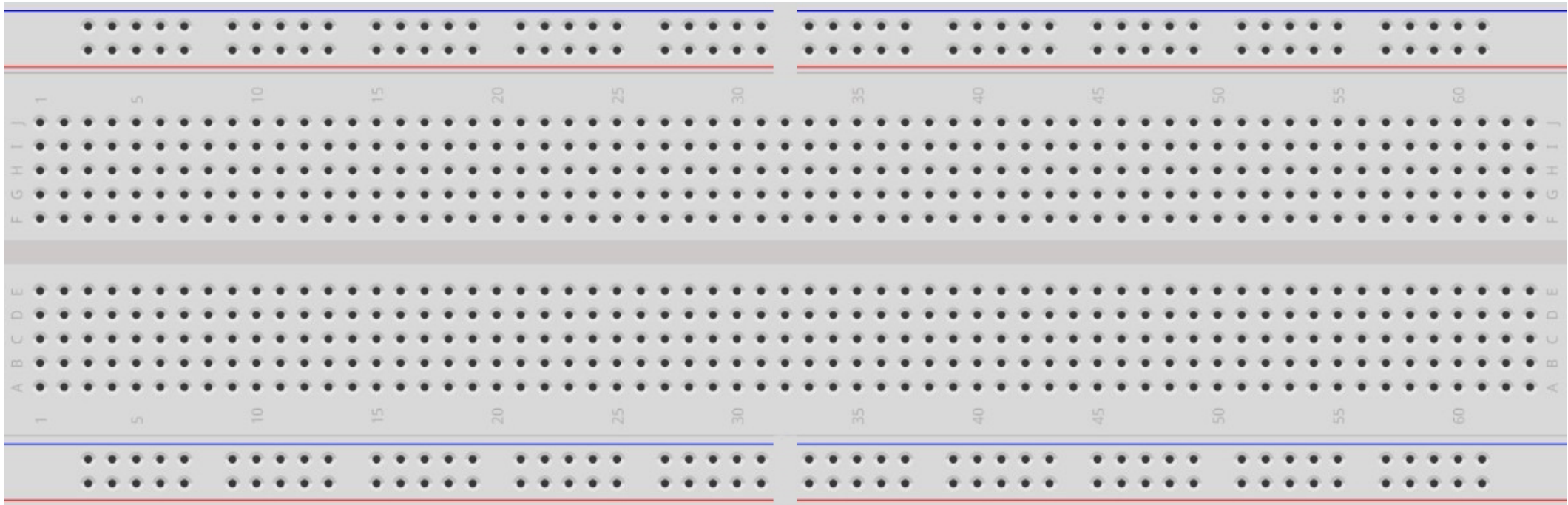
EJERCICIOS

COLOR	VALOR 1	VALOR 2	MULTIPLICADOR	TOLERANCIA
NEGRO	0	0	1 Ω	$\pm 1\%$
MARRÓN	1	1	10 Ω	$\pm 2\%$
ROJO	2	2	100 Ω	
NARANJA	3	3	1K Ω	
AMARILLO	4	4	10K Ω	
VERDE	5	5	100K Ω	$\pm 0.5\%$
AZUL	6	6	1M Ω	$\pm 0.25\%$
VIOLETA	7	7	10M Ω	$\pm 0.10\%$
GRIS	8	8		$\pm 0.05\%$
BLANCO	9	9		
ORO			0.1 Ω	$\pm 5\%$
PLATA			0.01 Ω	$\pm 10\%$

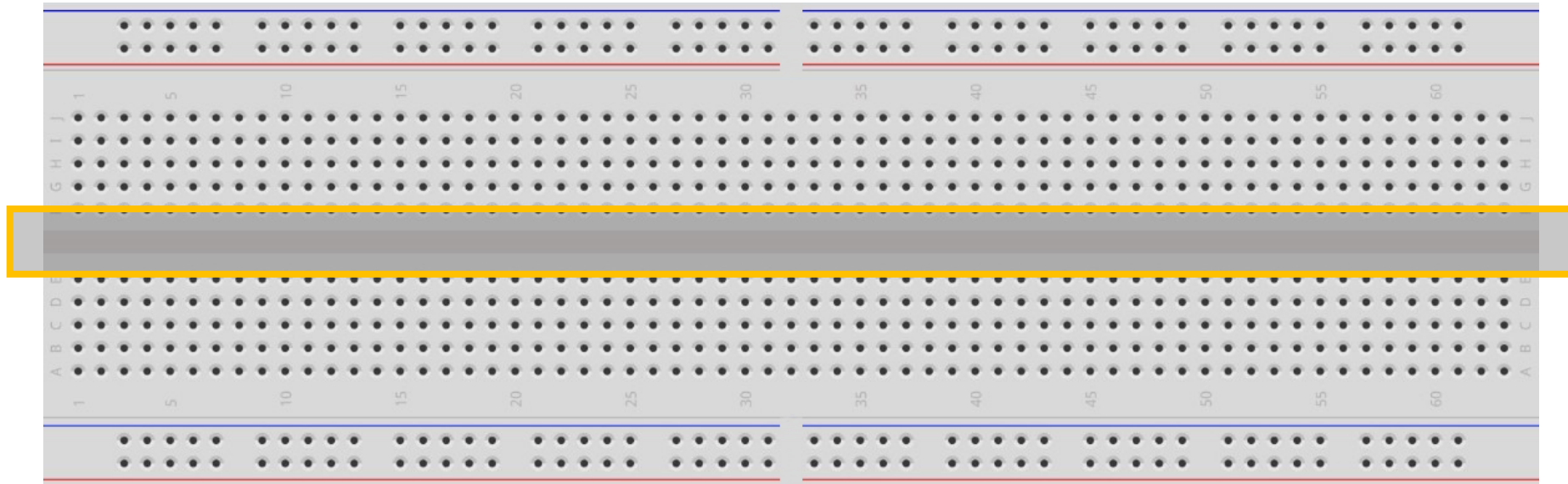


4700 [Ω]

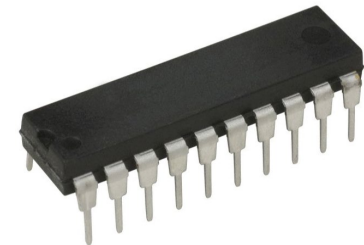
USO DE PROTOBOARD



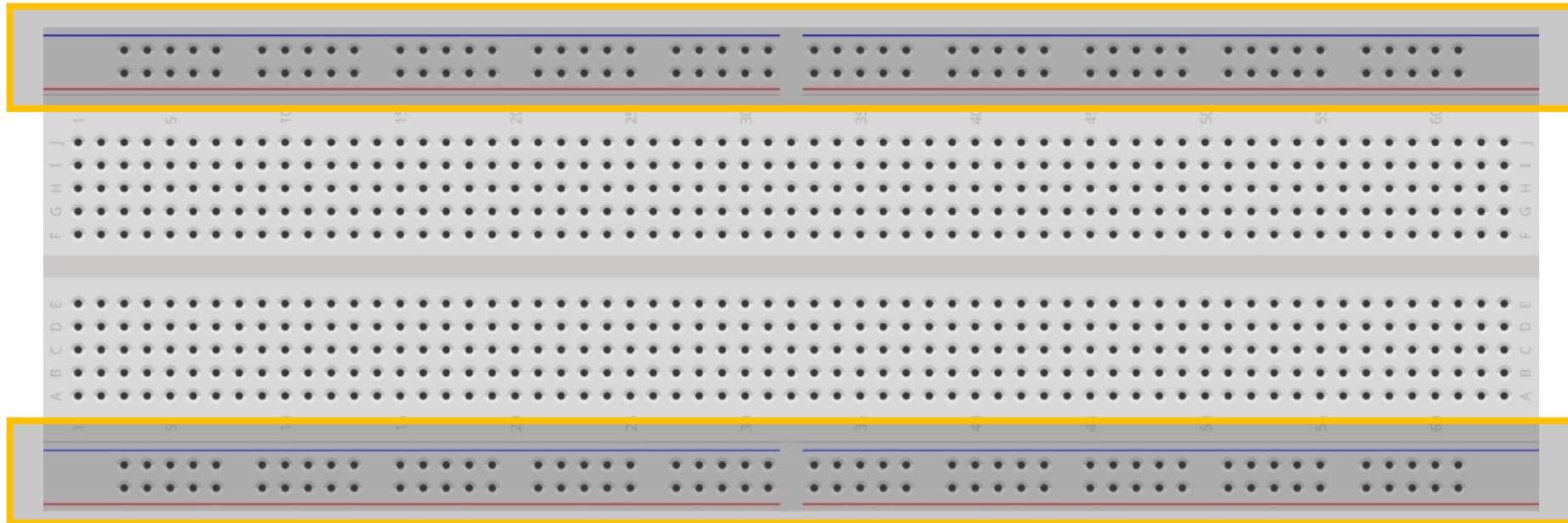
USO DE PROTOBOARD



Canal central: Es la región localizada en el medio del protoboard. Su mayor utilidad se da al momento de conectar circuitos integrados.

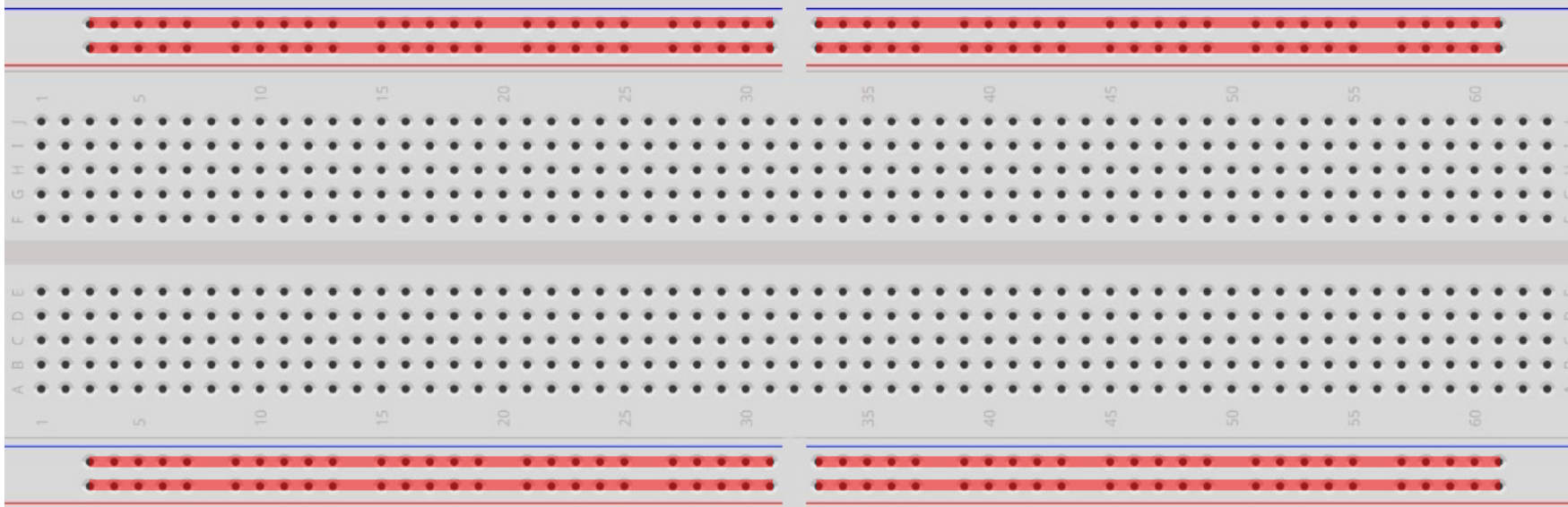


USO DE PROTOBOARD



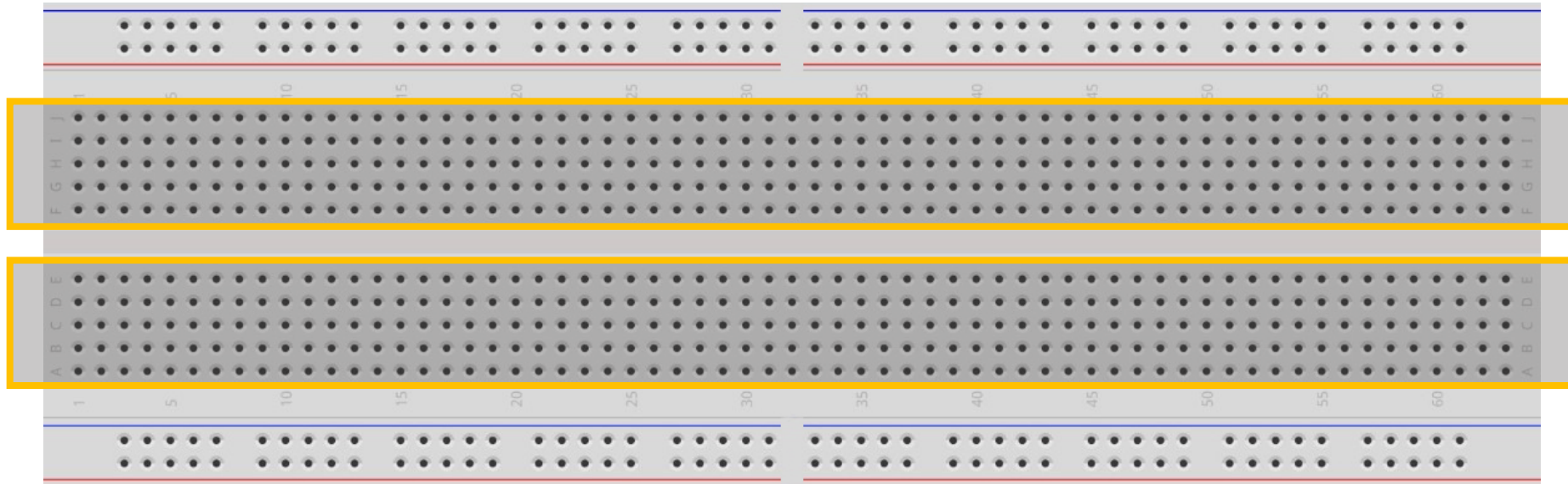
Buses: Se localizan en los extremos del protoboard.
Se utilizan para conectar la fuente de alimentación y tierra.

USO DE PROTOBOARD



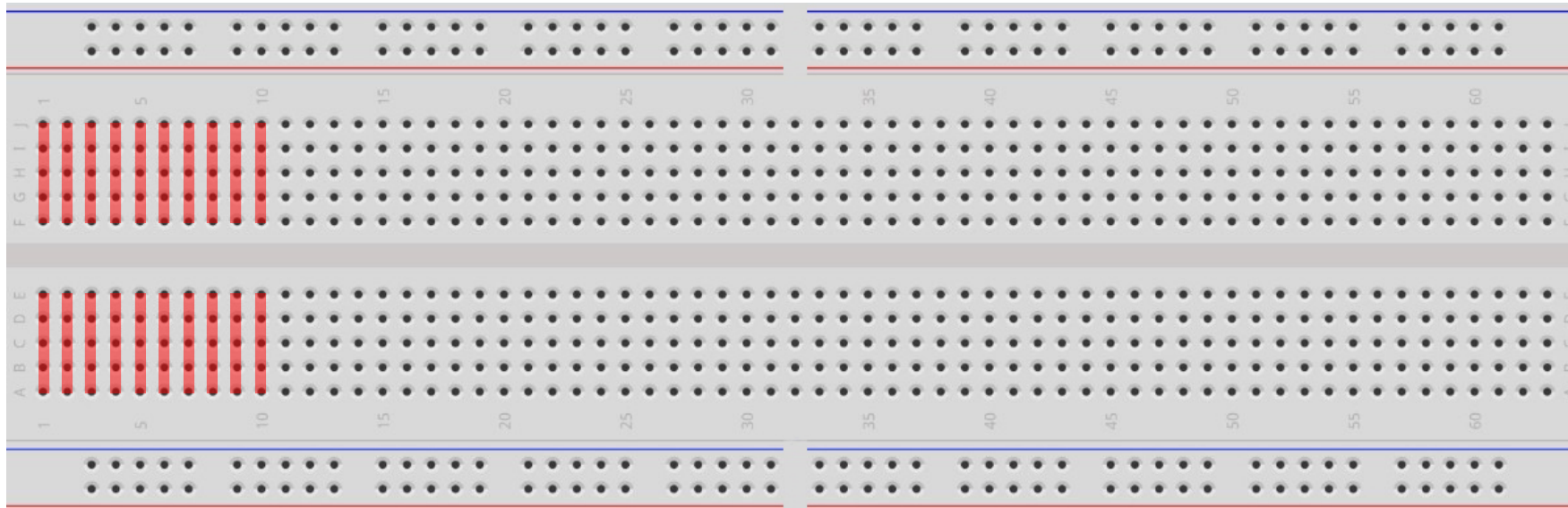
Buses: Se localizan en los extremos del protoboard. Se utilizan para conectar la fuente de alimentación y tierra.

USO DE PROTOBOARD



Nodos: También llamadas pistas, se localizan en la parte central del protoboard.

USO DE PROTOBOARD



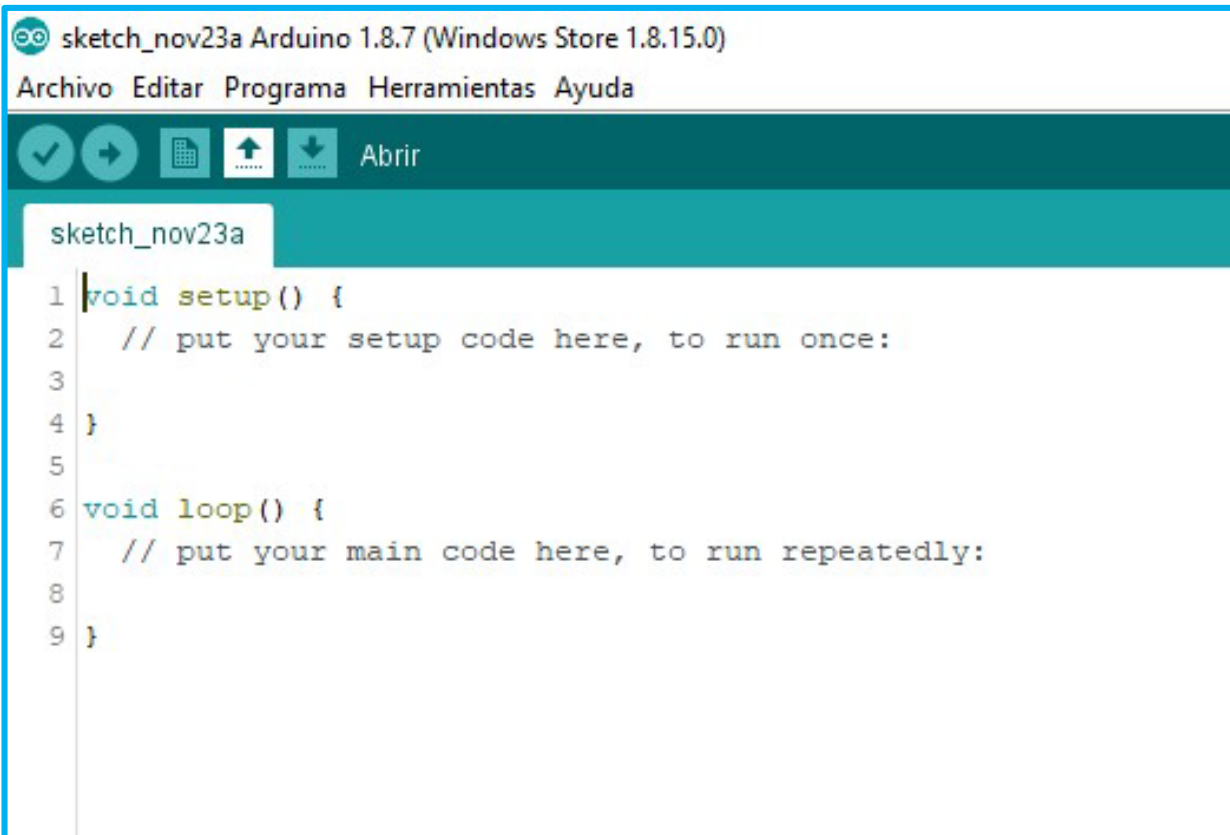
Nodos: También llamadas pistas, se localizan en la parte central del protoboard.



PROGRAMACIÓN EN ARDUINO

- Es la programación de un microcontrolador.
- El lenguaje de programación de Arduino está basado en C++.
- Lenguaje de programación es una estructura diseñada para describir el conjunto de acciones que un equipo debe ejecutar.
- Arduino incluye las herramientas necesarias para compilar y “quemar” el programa en la memoria flash del microcontrolador.

SKETCH DE ARDUINO



```
sketch_nov23a Arduino 1.8.7 (Windows Store 1.8.15.0)
Archivo Editar Programa Herramientas Ayuda

✓ → 📄 ⬆ ⬇ Abrir

sketch_nov23a
1 void setup() {
2   // put your setup code here, to run once:
3
4 }
5
6 void loop() {
7   // put your main code here, to run repeatedly:
8
9 }
```

- Un programa de Arduino se denomina sketch o proyecto y tiene la extensión .ino.
- Para que funcione el sketch, el nombre del archivo debe estar en una carpeta con el mismo nombre que el sketch.

ESTRUCTURA SKETCH

```
1 void setup() {  
2   // put your setup code here, to run once:  
3 }  
4  
5 void loop() {  
6   // put your main code here, to run repeatedly:  
7 }
```

- **void setup(){}**

Se utiliza para dar valores iniciales a las variables y establecer los modos de los pines (entrada o salida).

- **void loop(){}**

Es el bucle que se ejecuta infinitamente y donde reside la funcionalidad del sketch.

VARIABLES

- Una variable es una herramienta de programación que nos ayuda a almacenar y recuperar información en nuestros programas.
- Una variable debe tener un nombre.
- Una variable global es aquella que puede ser vista y utilizada por cualquier función.
- Una variable local es aquella que se define dentro de una función o como parte de un bucle.

```
sketch_nov23a $
1 int variable1;
2
3 void setup() {
4
5 }
6
7 void loop() {
8   int variable2;
9 }
10
```


DECLARACIÓN E INICIALIZACIÓN DE VARIABLES

`int temperaturaMaxima = 25;`




Diagram illustrating the components of the variable declaration and initialization code:

- `int`: tipo de dato
- `temperaturaMaxima`: nombre de la variable
- `=`: operador de asignación
- `25`: valor
- `;`: punto y coma



TIPOS DE DATOS

- **Byte**

Almacena un valor numérico de 8 bits sin decimales. Tienen un rango entre 0 y 255.

- **Entero (int)**

Almacena valores numéricos de 16 bits sin decimales comprendidos en el rango -32768 a 32767.

- **Entero largo (long)**

Puede almacenar números de 4 bytes (32-bit). El rango de valores es de -2.147.483.648 a 2.147.483.647. Puede almacenar números positivos y negativos.

TIPOS DE DATOS

- **Decimal (float)**

Se aplica a los números con decimales. Con este tipo de datos se pueden almacenar valores realmente grandes. Solo utiliza 4 bytes de memoria (32-bit).

- **Booleano (boolean)**

Es el tipo de dato que menos memoria ocupa, 1-bit. Puede tomar dos valores o 1 (true) o 0 (false).

- **Carácter (char)**

Utiliza 1 byte de memoria (8-bit). Permite almacenar una letra de una forma especial.

OPERADORES

- Un operador es un elemento de programa que se aplica a uno o varios operandos en una expresión o instrucción.
- Un operador, es un símbolo que indica al compilador que se lleve a cabo ciertas manipulaciones matemáticas o lógicas.

OPERADORES ARITMÉTICOS

= asignación

+ suma

***** producto

- resta

% módulo

OPERADORES COMPUETOS

$x++ \rightarrow x = x + 1$

$x-- \rightarrow x = x - 1$

$x += y \rightarrow x = x + y$

$x *= y \rightarrow x = x * y$

$x -= y \rightarrow x = x - y$

$x /= y \rightarrow x = x / y$

OPERADORES COMPARACIÓN

$x == y$ igual

$x != y$ diferente

$x < y$ menor

$x > y$ mayor

$x <= y$ menor igual



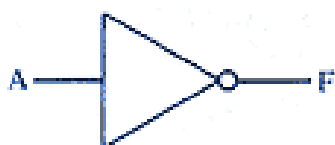
$x >= y$ mayor igual

OPERADORES BOOLEANOS

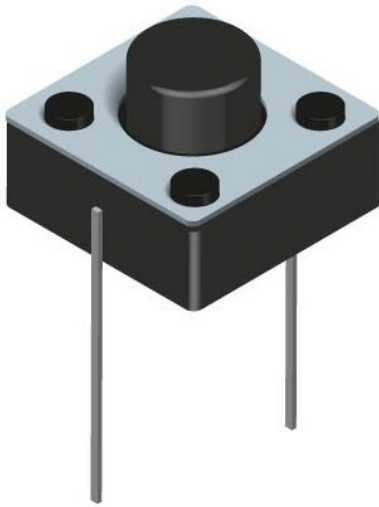
x && y

x || y

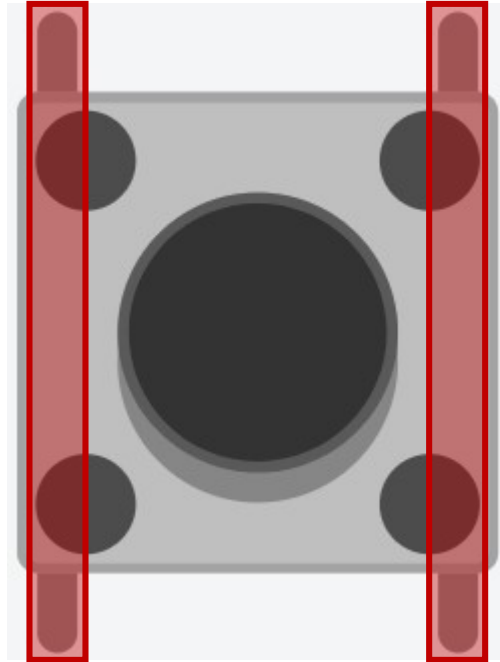
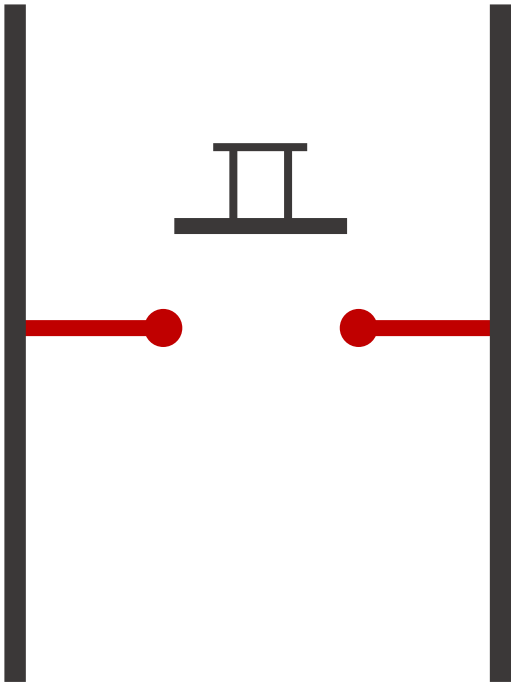
! x

Name	Graphic Symbol	Algebraic Function	Truth Table															
AND		$F = A \cdot B$ or $F = AB$	<table><tr><th>A</th><th>B</th><th>F</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	A	B	F	0	0	0	0	1	0	1	0	0	1	1	1
A	B	F																
0	0	0																
0	1	0																
1	0	0																
1	1	1																
OR		$F = A + B$	<table><tr><th>A</th><th>B</th><th>F</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	A	B	F	0	0	0	0	1	1	1	0	1	1	1	1
A	B	F																
0	0	0																
0	1	1																
1	0	1																
1	1	1																
NOT		$F = \bar{A}$ or $F = A'$	<table><tr><th>A</th><th>F</th></tr><tr><td>0</td><td>1</td></tr><tr><td>1</td><td>0</td></tr></table>	A	F	0	1	1	0									
A	F																	
0	1																	
1	0																	

PULSADOR

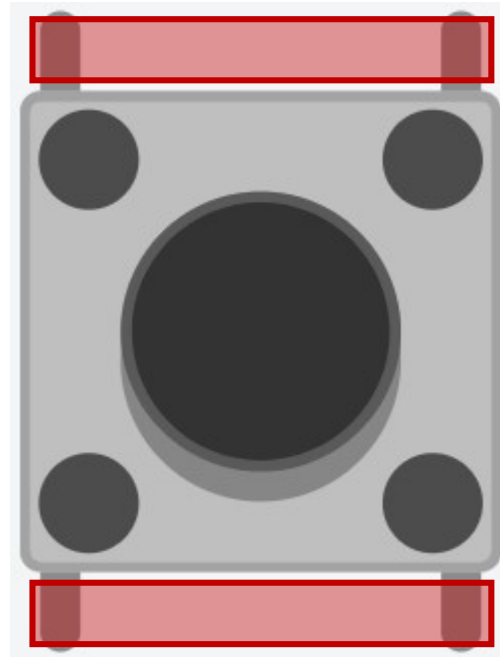
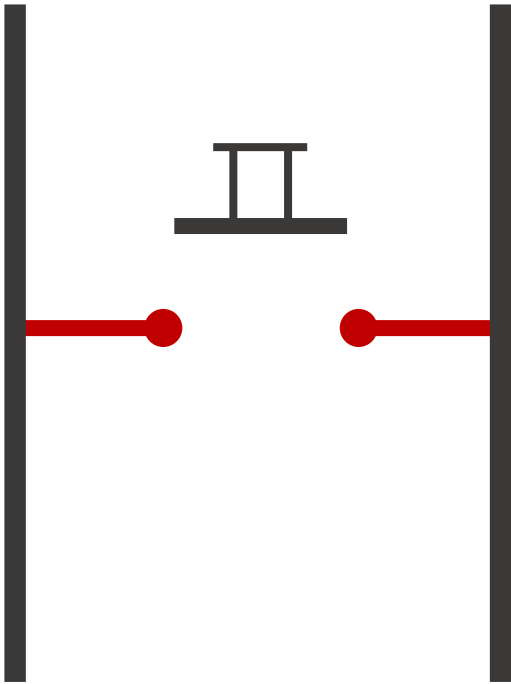


PULSADOR



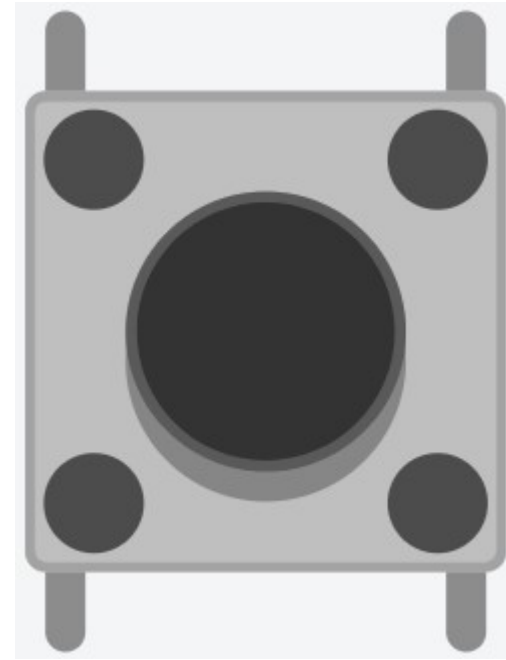
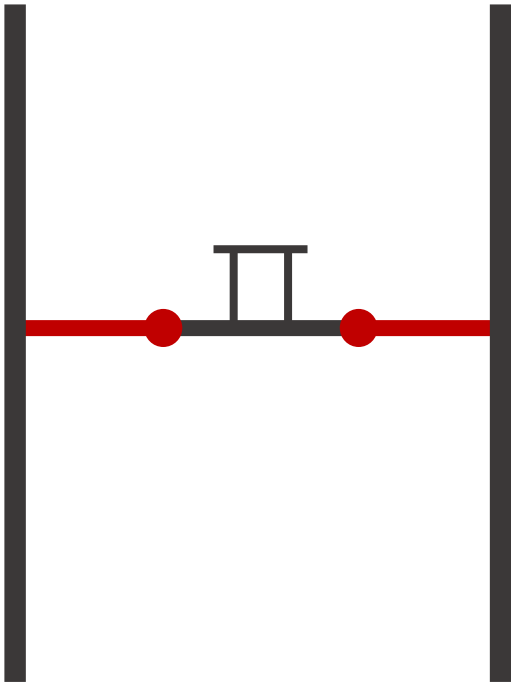
Estas dos patas
están conectadas
siempre una con
otra.

PULSADOR

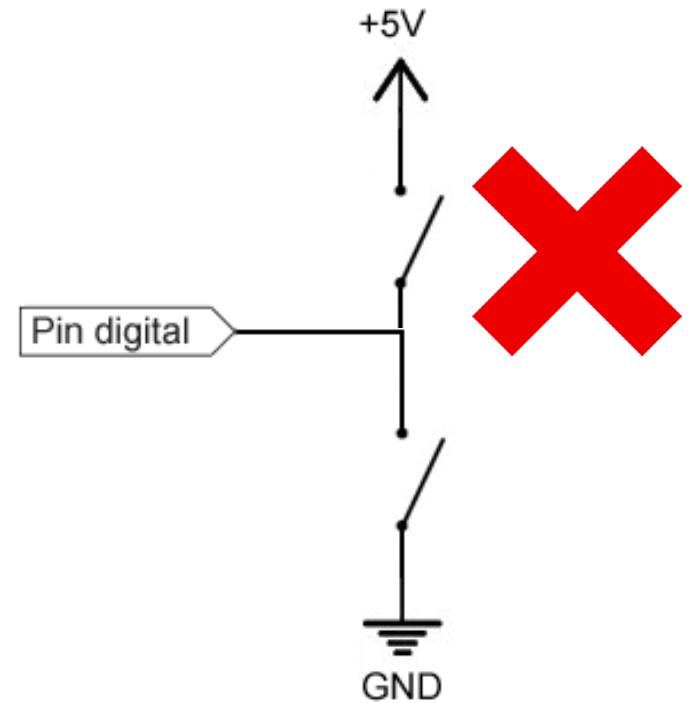
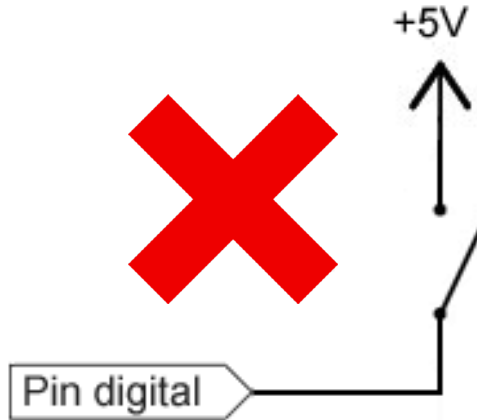
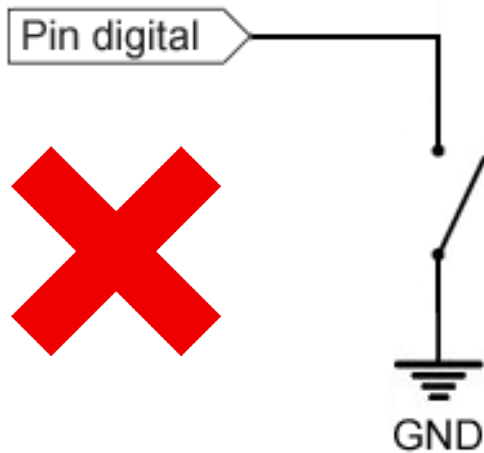


Estas dos patas
NO se encuentran
conectadas entre
sí.

PULSADOR



CONEXIÓN DE UN PULSADOR AL ARDUINO

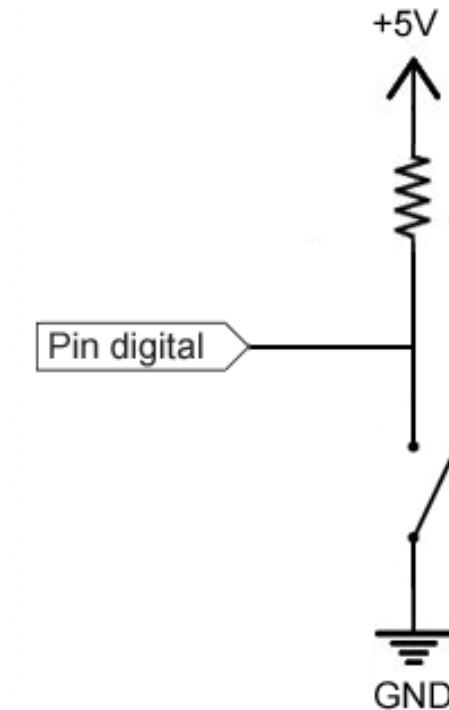


Cuando el pulsador está abierto, el pin digital no tiene un voltaje de referencia, por lo cual este queda en un estado de alta impedancia, es decir, puede asumir cualquier valor.

CONEXIÓN DE UN PULSADOR AL ARDUINO

Resistencia Pull-Up

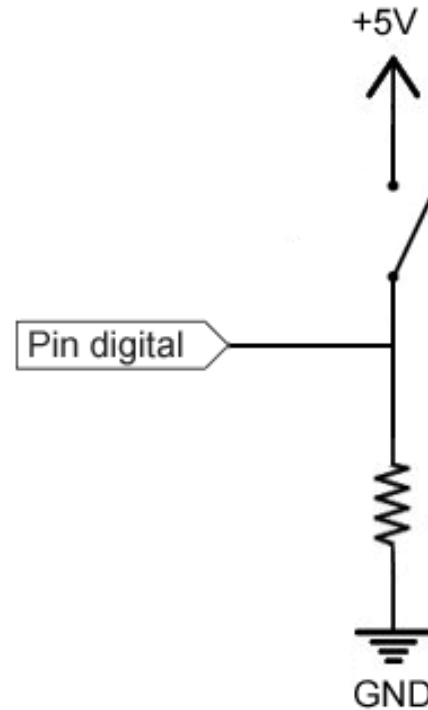
La resistencia de Pull-Up fuerza HIGH cuando el pulsador está abierto. Cuando está cerrado el PIN se pone a LOW, la intensidad que circula se ve limitada por esta resistencia.



CONEXIÓN DE UN PULSADOR AL ARDUINO

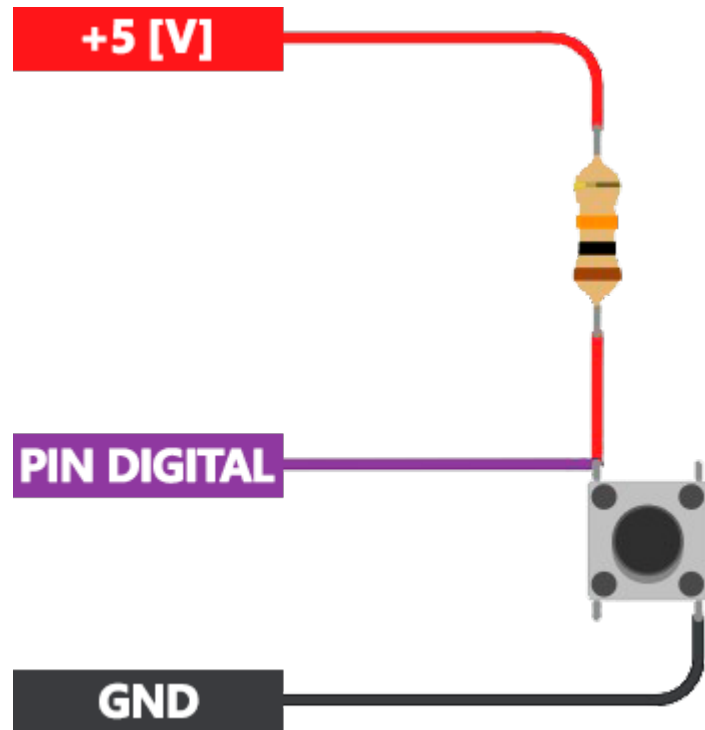
Resistencia Pull-Down

La resistencia de Pull-Down fuerza LOW cuando el pulsador está abierto. Cuando está cerrado el PIN se pone a HIGH, y la intensidad que circula se ve limitada por esta resistencia.



CONEXIÓN DE UN PULSADOR AL ARDUINO

Resistencia Pull-Up



Resistencia Pull-Down

