Índice de imágenes

Contenido

[Imagen 1: Uso del “if” 3](#_Toc157005205)

[Imagen 2: Introducción de la nueva entrada en el “setup()” 3](#_Toc157005206)

[Imagen 3: Introducción de la nueva entrada en el “loop()” 4](#_Toc157005207)

[Imagen 4: Introducción de delay entre estadoboton\_I y \_D 4](#_Toc157005208)

[Imagen 5: “Or” y “AND” en la condicional “if” 5](#_Toc157005209)

[Imagen 6: Impresión por pantalla de “estadoboton\_D” 6](#_Toc157005210)

[Imagen 7: Mejora de la impresión por pantalla 7](#_Toc157005211)

[Imagen 8: Resultado de la impresión por pantalla 7](#_Toc157005212)

Fuente:

<https://www.youtube.com/watch?v=fTjk9wKo5Ws&list=PLEzmH7aN82FEh2JjYuNCvFu6wFolHai32&index=7>

Inicialmente se ha de realizar un montaje similar al del PDF anterior, (Completar explicación)

Una vez montado los circuitos, hemos de usar el IDE para programarlo. En el anterior PDF nos había quedado un código tal que así:

|  |
| --- |
|  |
| Imagen : Uso del “if” |

Hemos de modificar ese código para que reciba el input tanto del pin 2 como del pin 3. Para ello definimos en el “Setup()” la nueva entrada del programa. La salida sigue siendo el pin 13. Una vez definida nos vamos al bucle.

En el bucle añadimos un “digitalRead()” ya que queremos saber el estado del pin 3 y cambiamos el nombre de la variable para que no se sobrescriba. Renombramos variables para poder organizarnos mejor.

Las llamamos “estadoboton\_D” y “estadoboton\_I”. Lo que quedaría sería repetir el mismo proceso que el de la Imagen 1 con las variables renombradas.

|  |
| --- |
|  |
| Imagen : Introducción de la nueva entrada en el “setup()” |

Y para el “void loop()” quedaría la :

|  |
| --- |
|  |
| Imagen : Introducción de la nueva entrada en el “loop()” |

Si le damos al botón de cargar el programa e intentamos encender los botones veremos que el derecho no se enciende mientras que el izquierdo sí. Esto ocurre porque Arduino ejecuta el bucle muy rápido, por tanto hemos de introducir dos delay, (ya explicado), para poder apreciarlo. El código, (solo en el “loop()”), quedaría como el de la Imagen 4:

|  |
| --- |
|  |
| Imagen : Introducción de delay entre estadoboton\_I y \_D |

Con eso ya lo veríamos. Ahora bien lo que queremos es que el LED este encendido todo el rato. Para ello usaremos un condicional un poco distinto llamado “OR” que en el IDE de Arduino se escribiría como “||”. El “AND” se escribe como “&&”. Quedaría como en la Imagen 5:

|  |
| --- |
|  |
| Imagen : “Or” y “AND” en la condicional “if” |

La explicación de la Imagen 5 es que mientras no estén los dos botones sin apretar, el LED va a encenderse. Si el “AND”, (“&&”), lo ponemos en el encendido ocurrirá la contrario que es que deben pulsarse los dos botones a la vez para que se encienda el LED.

También pueden usarse operaciones como <, >, >= o <= para definir en las condicionales. Solo han de sustituirse && y || por esos símbolos. El no es igual sería “¡=”.

Existe un comando en Arduino, (y en otros lenguajes de programación como Python), que es el **print.** Este comando te imprime por pantalla la variable que quieras ver. Para ello hemos de retocar tanto el setup() como el loop().

En el setup() hemos de añadir un comando llamado “**Serial.begin()**” que lo que establece la velocidad de entrada en bits/seg, (también conocido como Baudios), para transmitir datos desde el Arduino hasta el IDE. Hay que especificarle dicha velocidad, y pondremos una de las que viene por defecto que es 9600.

Para más información, buscar en el enlace:

<https://www.arduino.cc/reference/es/language/functions/communication/serial/begin/>

Una vez definido el inicio, (“begin”), hay que poner el propio print que lo escribiremos a la altura del código en el que queramos sacar por pantalla la información. El código quedaría como en la Imagen 6:

|  |
| --- |
|  |
| Imagen : Impresión por pantalla de “estadoboton\_D” |

Otro comando muy útil es “printnl” que sirve para sacar los valores en 1 columna en lugar de una fila.

El código sería como el de la Imagen 6 salvo que en la línea 15 de esa misma imagen escribimos “Serial.printnl(estadoboton\_D)” en lugar de “Serial.print”. Usando esos 2 comandos quedaría como en la Imagen 7:

|  |
| --- |
|  |
| Imagen : Mejora de la impresión por pantalla |

Y el resultado sería:

|  |
| --- |
|  |
| Imagen : Resultado de la impresión por pantalla |

En la Imagen 8 tenemos 0 cuando no están pulsados ninguno de los dos botones y 1 cuando si lo están.

Para ver el resultado por pantalla nos vamos a “Tools 🡪 Serial monitor”. Tal y como se describe en la Imagen 9:

|  |
| --- |
|  |
| Imagen 9: Acceso a Serial Monitor |