# ÍNDICE

# 1. CAJA DE DISPOSITIVOS

Para la realización de las siguientes prácticas se usará una caja con distintos dispositivos. La caja debe permanecer conectada al ordenador en todo momento mediante un cable USB. A continuación se especifican los distintos elementos. Más adelante se presentarán las funciones y su modo de uso para el correcto funcionamiento de cada dispositivo.

Figura 1: Caja de dispositivos

- Botonera: Consta de cinco pulsadores:

Figura 2: Botonera

- RGB: LED multicolor. Puede programarse para que luzca rojo, verde, azul o cualquier combinación de estos tres colores:

Figura 3: RGB

- LDR: Fotorresistencia. Se realiza una lectura de la luz que incide en el dispositivo, tomando valores próximos a cero cuando se encuentra tapado (ausencia de luz) y valores elevados cuando hay mucha luz.

Figura 4: LDR

- SERVOMOTOR: Motor que gira una posición dada. A diferencia de un servo de rotación continua, este motor no da vueltas completas, sino que se sitúa en la posición especificada en grados.

Figura 5: Servo

- Conjunto de LED (Reloj binario): Diodos LED que pueden apagarse o encenderse. La disposición en columnas se ha realizado para formar un reloj binario donde las columnas, de izquierda a derecha, se corresponden a las decenas de las horas, las unidades de las horas, las decenas de los minutos y las unidades de los minutos respectivamente.

Figura 6: LEDS

# 2. PROYECTO ARDUINO

Para la realización de estas prácticas se usará la plataforma Code::Blocks Arduino IDE. En el interfaz de la aplicación se pulsa File ⟶ new ⟶ Project…

Y se selecciona Arduino Project:


Al crear un nuevo proyecto en Arduino aparecerán una serie de ventanas emergentes:

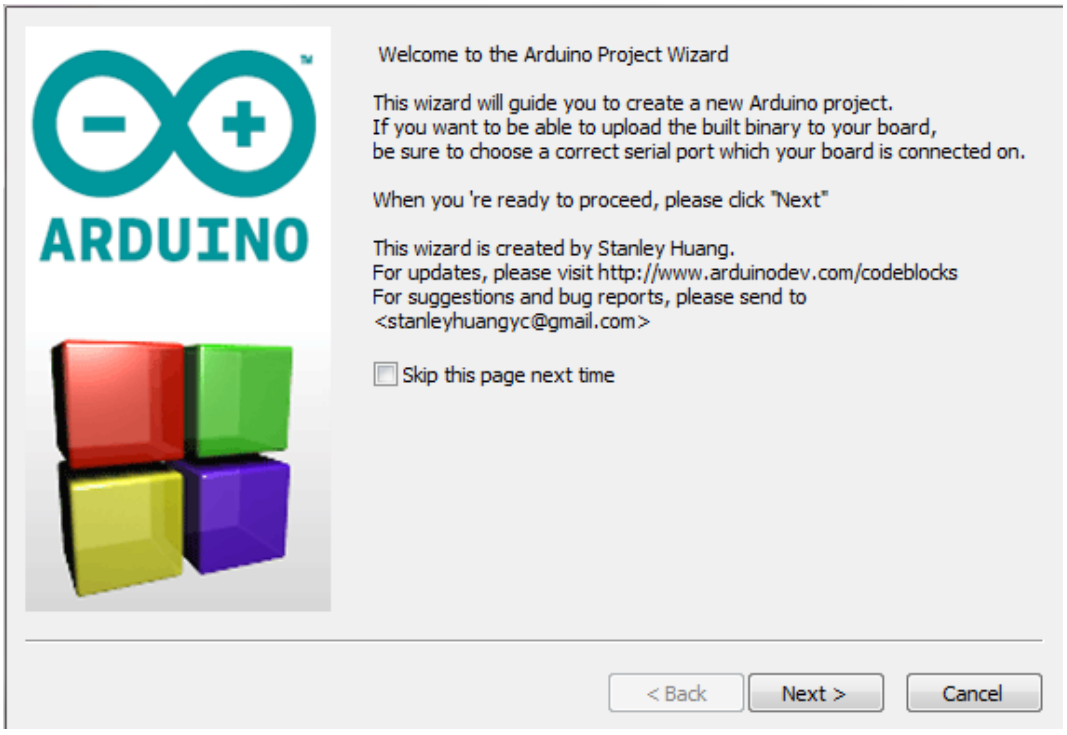


Figura 7: Creación de un proyecto en arduino

Pulsando en Next, se obtiene:

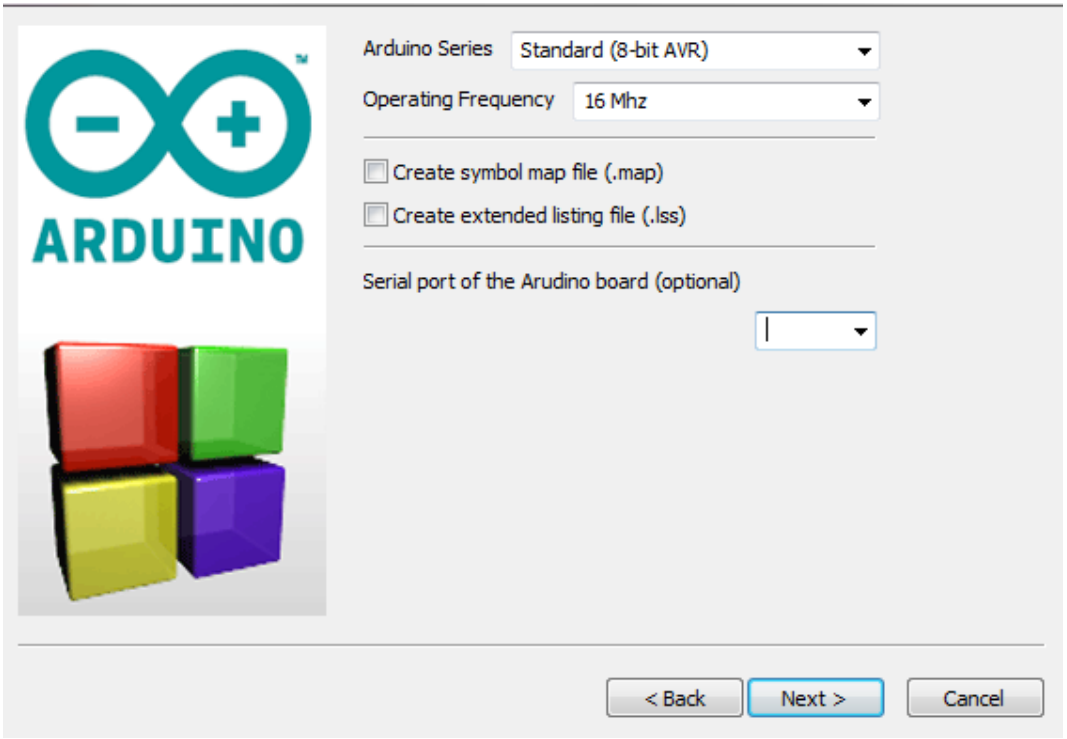


Figura 8: Configuración de parámetros

Se dejan las opciones que vienen por defecto y se pulsa Next.


Figura 9: Título y ruta de acceso

En esta ventana se dará título al proyecto en la pestaña Project title. Es muy importante no poner espacios en el nombre del proyecto, ya que creará un

conflicto interno que hará imposible cargar el programa, en su lugar se puede sustituir el espacio por una barra baja. Se definirá la ruta donde desea guardarse el nuevo proyecto en la pestaña Folder to create Project in. Una vez dado nombre al proyecto y definido el lugar donde se debe crear, se pulsa Finish.

Se abrirá un código de ejemplo sobre el que se va a trabajar:

```
1   #include <Arduino.h>
2
3   /*
4     Turns on an LED on for one second, then off for one second, repeatedly.
5   */
6
7   void setup()
8   {
9       Serial.begin(9600);
10
11      // initialize the digital pin as an output.
12      // Pin 13 has an LED connected on most Arduino boards:
13      pinMode(13, OUTPUT);
14  }
15
16  void loop()
17  {
18      Serial.println("Hello world!");
19
20      delay(1000);              // wait for a second
21      digitalWrite(13, HIGH);   // set the LED on
22      delay(1000);              // wait for a second
23      digitalWrite(13, LOW);    // set the LED off
24  }
25
```

Figura 10: Ejemplo de proyecto en arduino

En primer lugar el alumno debe eliminar todas las instrucciones del código, dejando únicamente el setup() y el loop(). Se trabajará a partir de este código:

```
#include <Arduino.h>


void setup()
{
    Serial.begin(9600);

}

void loop()
{

}
```

Figura 11: Código base para proyecto en arduino

En primer lugar se debe incluir la librería “Led.h” ya que es donde se encuentran todas las funciones que harán que el alumno pueda usar los distintos dispositivos de la caja.

```
#include <Arduino.h>

#include "Led.h"

void setup()
{
    Serial.begin(9600);

}

void loop()
{

}
```

Como se puede apreciar, un proyecto con Arduino consta de dos partes. Por un lado está la función setup(). Las instrucciones que se den en esta función se realizarán una única vez al principio del programa. Por esto es la parte donde se inicializan variables o donde, por ejemplo, se abre la comunicación con el puerto serie:

```
void setup()
{
    Serial.begin(9600);

}
```

La segunda parte del proyecto, el loop(), podría definirse como el main() que se ha venido usando hasta ahora. La principal diferencia es que esta función se ejecuta repetidamente en un bucle del que no se sale nunca. El programa está constantemente repitiendo las mismas instrucciones. Si por ejemplo, se desea que el programa haga una cosa diferente en función del botón pulsado, es necesario que se esté leyendo continuamente el valor del botón activo. Es por esto que en los presentes ejercicios, las instrucciones se darán siempre en el loop().

Excepciones: Como se ha mencionado anteriormente, los objetos deben definirse en el setup() ya que sólo se ejecuta una vez. Es por esto que al usar el servomotor, deben introducirse algunas instrucciones en esta parte. Esto se definirá más adelante, cuando se estudien cada una de las funciones de la librería “Led.h”.

# 3. MANUAL DE FUNCIONES DE LA LIBRERÍA “Led.h”

A lo largo de los próximos ejercicios se van a usar una serie de funciones que se encuentran en la librería “Led.h”. Para poder hacer uso de las mismas es imprescindible que en cada nuevo proyecto se incluya la librería:

```
#include "Led.h"
```

Las funciones disponibles son:

## escribir()

Esta función escribirá en el puerto serie la variable introducida por el alumno. Para abrir el puerto serie en Arduino Builder, una vez se ha cargado el código en la placa, se selecciona la velocidad de 9600 en la pestaña de la derecha y se pulsa open.

Ejemplo de uso:

```
escribir("Hola mundo!");
```

Al abrir el puerto serie se imprimirá la frase Hola mundo! repetidamente.

## retardo()

Esta función realizará una espera (delay) del tiempo introducido por el alumno. El tiempo debe introducirse en segundos.

## LedOn() y LedOff()

LedOn enciende y LedOff apaga el número de LED que se haya introducido respectivamente. La numeración de los LED es la siguiente:

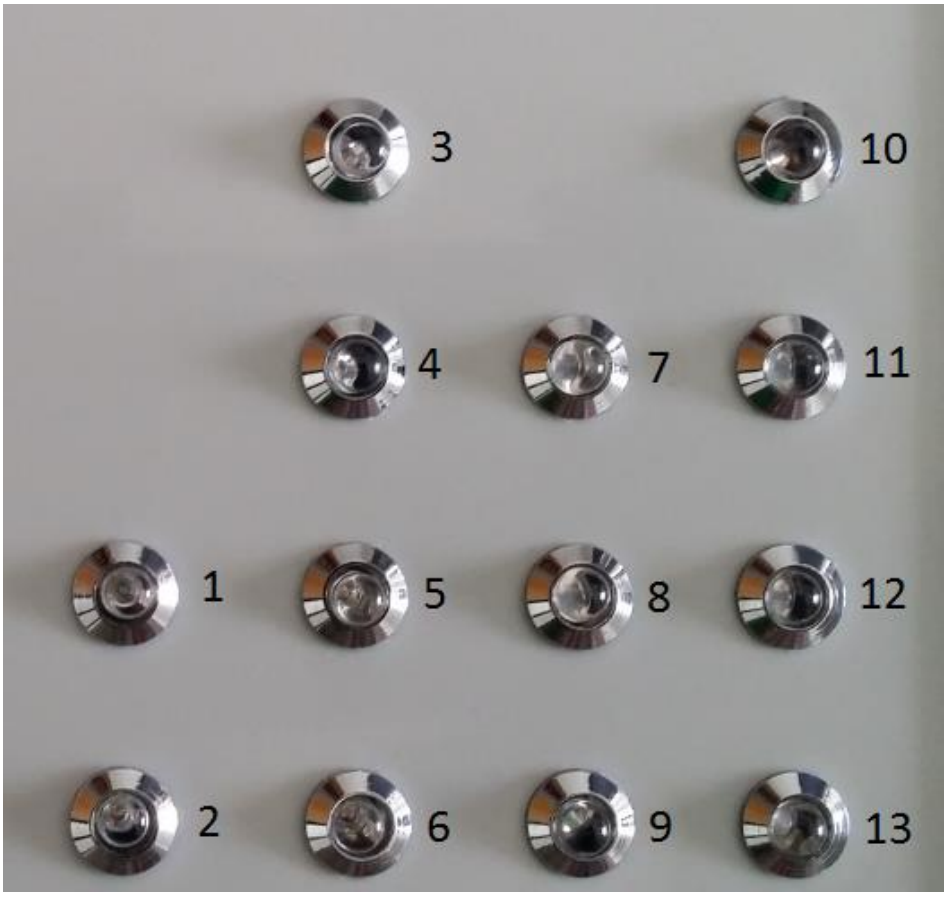


Figura 12: Numeración de los LEDS

Ejemplo de uso:

```
LedOn(5);
retardo(10);
LedOff(5);
retardo(8);
```

En este caso se encenderá el LED 5 durante 10 segundos y se apagará durante 8 segundos.

## botonera()

Dado que existen 5 botones, esta función devolverá el valor del botón que se haya pulsado (del 1 al 5). Se creará una variable en la que se almacenará el valor devuelto por la función. La numeración de los botones es la siguiente:

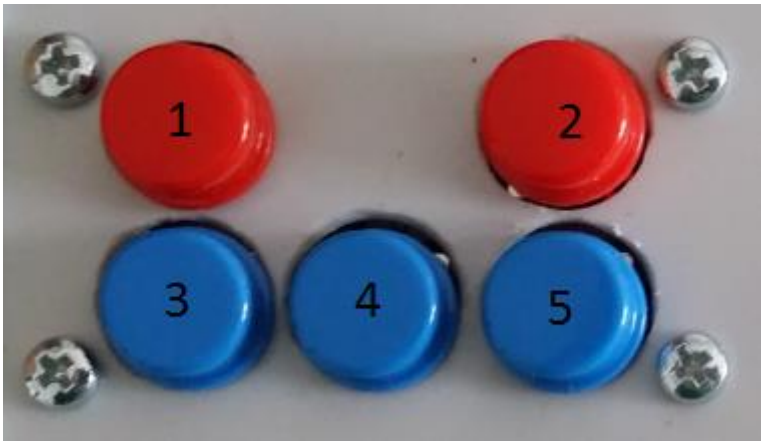


Figura 13: Numeración de los pulsadores

Ejemplo de uso:

```
int boton=botonera();
if(boton==1)
    LedOn(1);
```

En este caso cuando se pulse el botón 1 se encenderá el LED 1.

## LDR()

Esta función devuelve el valor analógico leído por el sensor LDR. Se almacenará en una variable para poder trabajar con él.

Ejemplo de uso:

```
int valor=LDR();
escribir(valor);
```

En este caso, al abrir el puerto serie en Arduino Builder (a la velocidad de 9600) aparecerán los valores leídos por el sensor LDR.

## RGB()

Esta función controla el LED RGB mediante 3 valores. El rango de operación es de 0 a 255, siendo el 0 nada de ese color y el 255 todo de ese color. Se deberán introducir los valores que se quieran para el rojo, el verde y el azul en ese orden.

Ejemplo de uso:

```
RGB(0, 255, 0);
```

De esta manera el RGB se iluminará en color verde.

## zumbador()

Esta función hace que suene el zumbador con la frecuencia introducida durante un tiempo determinado en segundos.

La frecuencia de las notas de la escala musical es:

| Nota | Frecuencia (Hz) |
|---|---|
| Do | 261.63 |
| Re | 293.66 |
| Mi | 329.63 |
| Fa | 349.23 |
| Sol | 392.00 |
| La | 440.000 |
| Si | 493.88 |

Tabla 1: Frecuencia de las notas musicales

Ejemplo de uso:

```
zumbador(350, 5);
```

En este caso sonará la nota FA durante 5 segundos.

## SERVO

La caja de dispositivos cuenta con un servo motor SG90. Éste no es un motor de rotación continua, sino que se mueve por posiciones. Es decir, el servo no puede realizar vueltas completas, en su lugar se sitúa en una posición definida en grados y que va de 0 a 180.

Para poder usar esta función es imprescindible incluir la librería “Servo.h”

```
#include "Servo.h"
```

En primer lugar hay que definir el objeto del servo, para ello se debe poner la siguiente línea de código después de incluir la librería y antes del setup:

```
Servo miservo;
```

Lo siguiente es asociar dicho servo al pin conectado. Para ello se debe copiar la siguiente frase dentro del setup:

```
void setup()
{
    Serial.begin(9600);

    miservo.attach(6);

}
```

Para escribir la posición en grados que se desea tomar el comando es miservo.write(grados) y para especificar el tiempo requerido para llegar a esa posición se usará la función retardo con el tiempo en segundos. De esta manera se puede variar la velocidad del servo.

Es muy importante tener en cuenta que la posición máxima del servo es de 180º y la mínima de 0º.

Ejemplo de uso:

```
#include <Arduino.h>
#include "Servo.h"
#include "Led.h"
Servo miservo;

void setup()
{
    Serial.begin(9600);

    miservo.attach(6);

}

void loop()
{
    miservo.write(0);
    retardo(0.5);
    miservo.write(179);
    retardo(0.5);
}
```

En este ejemplo el servo se moverá entre sus posiciones máximas y mínimas tardando medio segundo en realizar cada recorrido. El movimiento se repetirá en un bubcle. Si se modifica el retardo a 0.25 se consigue que la velocidad aumente realizando el recorrido en la mitad de tiempo.

## hora ( , , , )

Esta función sirve para manejar el reloj binario disponible en la caja de dispositivos. Como argumento de entrada a la función debe introducirse la hora en decimales que se desea representar en binario separando cada dígito por comas (ya que se trata de un vector). El modo de uso es:

**hora( decenas_horas , unidades_horas , decenas_minutos , unidades_minutos);**

**reloj ( , , , )**

Esta es otra de las funciones que maneja el reloj binario. En este caso cada uno de los 4 argumentos de entrada es un vector de diferentes tamaños. Cada vector se corresponde con cada una de las columnas del reloj. . El tamaño se corresponde al número de LED de cada columna. En primer lugar se deben crear los vectores con los valores 0 (LED apagado) o 1 (LED encendido). Al pasar a la función los valores de los vectores, se encenderán los LED indicados representando la hora introducida en números binarios.

Ejemplo de uso:

```
#include <Arduino.h>
#include "Led.h"


void setup()
{
    Serial.begin(9600);

}

void loop()
{
    int a[2]={1,1};
    int b[4]={1,1,0,1};
    int c[3]={1,1,1};
    int d[4]={1,0,1,0};
    reloj(a,b,c,d);

}
```

# 4. FUNCIONES IF, IF-ELSE

Para el estudio de estas estructuras se realizará el problema de una luz de escalera. Se irá resolviendo en distintas fases añadiendo, cada vez, un grado de complejidad. Se utilizará uno de los pulsadores de la botonera y uno de los diodos LED a elegir, ambos, por el alumno.

1. *La luz se enciende cuando se pulsa un botón. y se apaga cuando se vuelve a pulsar.*
   En este punto el alumno decidirá cuál de los botones será el que accione la luz y qué led se encenderá.

```
#include <Arduino.h>
#include <Led.h>

/*Enciende un LED al pulsar un boton*/

void setup()
{
    Serial.begin(9600);
}

void loop()
{
    int boton=botonera();
    if(boton==1)
        LedOn(1);
}
```

2. *La luz se enciende al pulsar un botón, espera un tiempo definido (por ejemplo 10 segundos) y se apaga. Si cuando la luz está encendida se vuelve a pulsar el botón no sucede nada.*

3. *La luz se enciende cuando se pulsa un botón y se apaga al volver a pulsarle.*
   En este punto será necesario crear una variable de estado que indique si el valor del botón ha cambiado, de manera que si se detecta algún cambio, el led se encenderá o se apagará.
   La variable será de tipo bool, de manera que su valor cambiará entre true o false. En primer lugar se inicializa como false y cada vez que se pulse el botón cambiará su valor, de manera que al ser true se encenderá el led y al ser false se apagará.

```
bool estado = false;
    if(botón==1)
estado=!estado;
```

4. *La luz se enciende automáticamente si se detecta que es de noche.*
   Para esto se usará un sensor LDR. En primer lugar el alumno leerá los valores que alcanza dicho sensor. Para ello utilizará la función escribir() en la que introducirá la variable en la que ha guardado el valor leído por el sensor LDR. Una vez abierto el puerto serie en Arduino Builder (velocidad de 9600), debe estudiar el rango de funcionamiento del sensor, es decir, qué valor toma cuando hay mucha luz incidente y cuál cuando está totalmente tapado. Una vez estudiados estos valores el alumno debe escoger un valor por debajo del cual debe encenderse el LED.
5. *La luz se enciende cuando se pulsa el botón sólo cuando es de noche.*

# 5. FUNCIÓN SWITCH

Para el estudio de esta función se utilizará la botonera.

1. *Encender el diodo RGB de un color diferente en función del botón pulsado.*
   Por ejemplo, al pulsar el botón 1 se pondrá azul, el 2 verde, el 3 rojo…
2. *Hacer sonar el zumbador con una nota diferente en función del botón pulsado.*
   Por ejemplo, al pulsar 1 hacer sonar la nota DO, 2 RE, 3 MI…
3. *Movimiento del servo mediante la botonera. Al pulsar el botón 1, el servo se situará en la posición de 0º e irá moviéndose un número definido de grados en función del botón pulsado. Además se especificará el tiempo para realizar cada movimiento*
   Por ejemplo, al pulsar el botón 1 se moverá a la posición de 30º en medio segundo, el botón 2 a 90 en un cuarto de segundo…

# 4. FUNCIONES DO, DO-WHILE

1. *Encender el LED cuando se detecta que es de noche.*
   Para resolver este problema se utilizará el sensor LDR y un LED a elegir por el alumno.
2. *Recorrer todos los colores del RGB.*
3. *Hacer que se enciendan uno a uno todos los LEDS con una espera entre encendidos a elegir por el alumno.*

# 5. FUNCIÓN FOR

Para el estudio de este bucle se usará una fila de LED. La disposición de los LED no va a ser lineal ya que para un ejercicio que se presentará más adelante es necesario que sean 4 filas diferentes pero para realizar estos ejercicios se considerará que es una misma línea y se contarán los LED del 1 al 13.

1. *Ir encendiendo uno a uno, en orden, cada LED.*
2. *Ir encendiendo uno a uno, en orden, cada LED y una vez que están todos encendidos ir apagándolos uno a uno en orden inverso.*

3. *Ir encendiendo uno a uno, en orden, cada LED pero una vez que se enciende uno se apaga el anterior, de manera que en cada momento solo habrá un LED encendido.*
4. *Ampliación del ejercicio anterior, se irá encendiendo cada vez un LED del 1 al 13 y luego cada vez un LED del 13 al 1.*
5. *Recorrer todos los colores del LED RGB.*

# 6. VECTORES

Para el estudio de vectores se va a realizar un reloj binario. Para ello se dispondrán cuatro columnas de LED. La primera con dos, la segunda con cuatro, la tercera con tres y la cuarta con cuatro. Las dos primeras columnas representan las horas y las dos segundas los minutos.

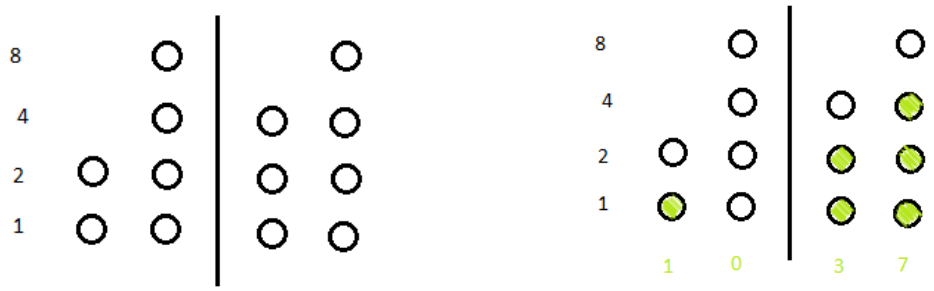


Figura 14: Reloj binario

Tal como se ve en la imagen de ejemplo, las columnas de la izquierda corresponden a las horas, en el ejemplo está el primer LED encendido, que corresponde al 1 y en la segunda columna no hay ninguno encendido, por lo que es un 0 (la hora se da en formato 24 h). Las columnas de la derecha corresponden a los minutos y, sumando los LED encendidos, se ve que son 3 y 7. Por lo que serán las 10:37

El objetivo final de este ejercicio es que el alumno comprenda y practique el uso de números binarios. Las actividades que se plantean se basan únicamente en el manejo del número binario. Para ir más allá en la parte de programación es posible que sea el alumno quién programe las funciones reloj() y hora().

1. *Representar una hora específica (por ejemplo, las 13:30).*
   Para este punto se utilizará la función reloj(1,3,0,0);
2. *Representar una hora binaria.*
   Para ello se configurarán los vectores mediante unos y ceros. El alumno debe entender la hora decimal que se ha representado mediante vectores en sistema binario.