MEBO

Robot

Educativo

Introducción al mundo de la robótica

Iván Soormally Rodríguez
Jesús Centeno Ponce
Guillermo Cárdenas Fernández

1. NATURALEZA DEL PROYECTO

Hemos escogido la elaboración de este proyecto porque nos resultaba bastante interesante que niños de primaria empiecen a conocer cómo funciona la robótica.

A través de este robot educativo, **Mebo**, conseguimos que adquieran conceptos básicos sobre programación, que conozcan diferentes componentes, así como sus distintas aplicaciones. Todo ello sin dejar de lado que se trata de un juguete.

Queríamos que este robot fuese accesible para todos, por lo que hemos utilizado componentes que se puedan encontrar en la mayoría de las tiendas de electrónica a un precio razonable.

Mebo podrá construirse por cualquier persona que tenga conocimientos básicos de electrónica, pudiendo descargar tanto la programación como los esquemas electrónicos, para montarlo por su cuenta.

También pondremos a disposición la posibilidad de adquirir un kit con todo lo necesario para realizarlo.

2. BÚSQUEDA DE INFORMACIÓN

Para la realización de este proyecto hemos investigado otros robots que realizan una función similar, como el **Bee Bot**, pero detectamos que la mayoría son funciones muy básicas las que realizan, lo cual queríamos modificar para que **Mebo** fuese más completo.

A partir de ahí, surgieron varias ideas, como añadirle pinzas y el control mediante **Bluetooth**, elementos que no se encuentran juntos en otros robots del mismo segmento.

3. SOLUCIÓN ELEGIDA

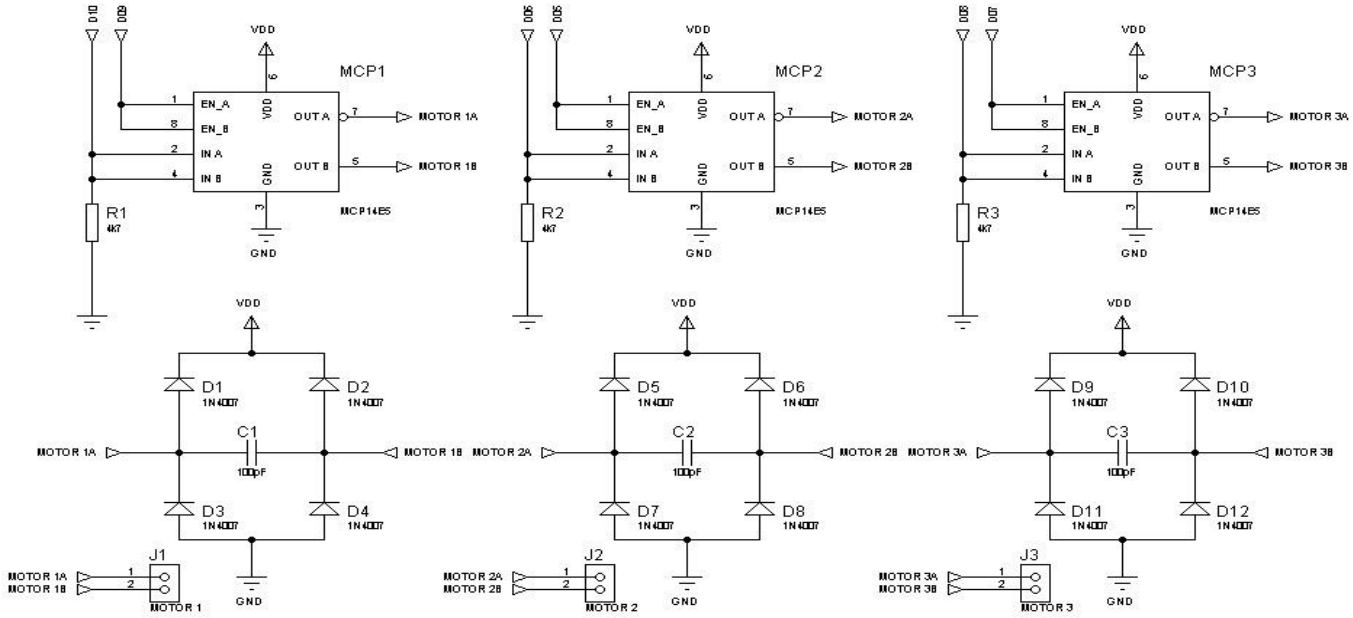
A partir de la información recopilada y de las ideas que teníamos sobre este robot decidimos que realizara las siguientes funciones:

- El robot debía moverse en un tablero cuadriculado, con unas dimensiones concretas, desplazándose por las casillas sin desviarse, para ello cada rueda dispondrá de un **Encoder** para controlar la velocidad de cada rueda y así evitar que ocurra.
- Dispondrá de una pinza en la parte frontal para poder agarrar y desplazar objetos de una casilla a otra.
- En el robot estará dotado de **LEDs** que se activen con el movimiento o por decisión del usuario.
- También dispondrá de un **Buzzer**, para poder reproducir canciones y dar aviso de alguna acción.
- Controlado por un dispositivo móvil, mediante una **APK**, podremos introducir las instrucciones de manera secuencial y enviarlas posteriormente al robot pulsando el botón de confirmar.
  Este recibirá la información y se pondrá en marcha automáticamente.
- Una placa de **Arduino Nano** será quién haga la función de cerebro, facilitando la carga del programa y las futuras modificaciones por parte de los usuarios.

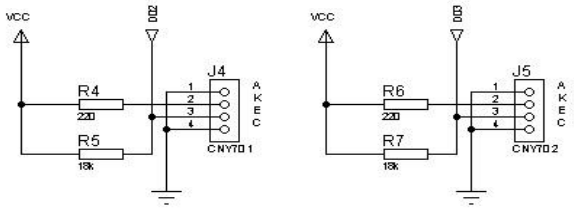
4. CONTENIDO

a. Esquemas electrónicos.

Contol de motores

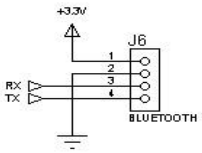


Encoders:

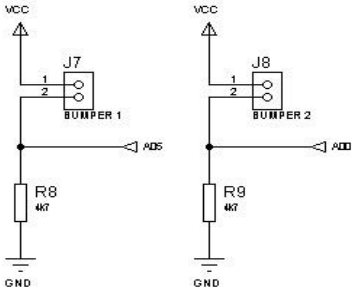


Arduino Nano:

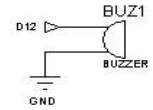
Bluetooth:



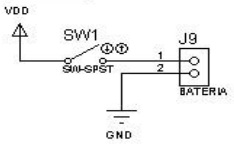
Finales de carrera:



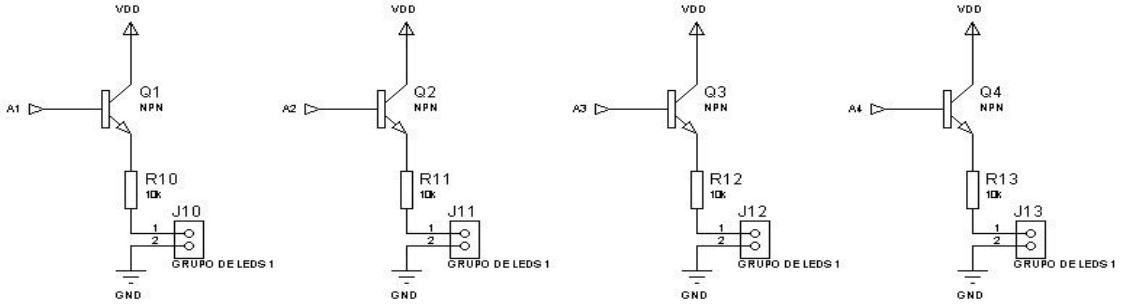
Buzzer



Batería:



Alimentación de leds:



En el **esquema electrónico** aquí mostrado podemos observar su composición en **8 partes**:

1. **Control de motores:**

Compuesto por el **driver MCP14E5** y un **puente en H** para proteger.
Driver elegido por su posibilidad de controlar tanto la dirección, como la velocidad de los motores (introduciéndole una señal PWM por la entrada EN) y tan solo utilizando dos pines de control.
También utilizado en el cierre y apertura de las pinzas, sin ser necesario tener en cuenta la velocidad.

2. **Encoder**:

Compuesto por un **sensor CNY70**, este nos informa mediante una de sus patillas del nivel alto o bajo, dependiendo si pasa o no por una superficie que es reflejada.

3. **Finales de carrera:**

Detectan si las pinzas están abiertas o cerradas para activar o desactivar el motor.

4. **Buzzer:**

El **Buzzer** es el encargado de reproducir los sonidos deseados.

5. **Batería:**

Con ella se conecta y desconecta la alimentación.

6. **Alimentación de LEDs:**

Consta de cuatro grupos de **LEDs** alimentados por un transistor para evitar un sobreconsumo en la placa **Arduino**.

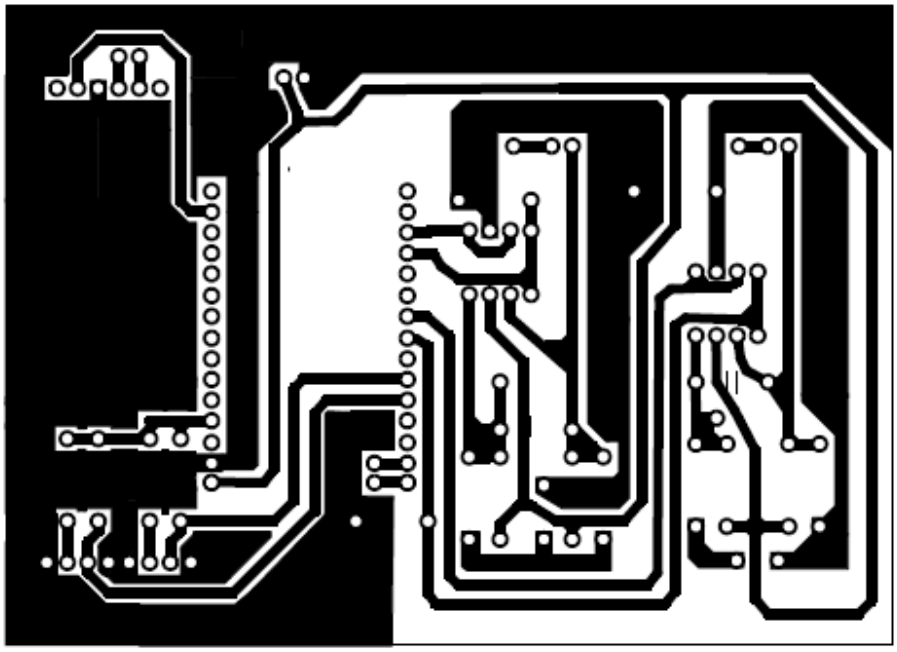
7. **Bluetooth:**

Conexionado del módulo **Bluetooth** a la placa de **Arduino**, sin olvidar que la conexión Tx y Rx va cruzada.

8. **Arduino:**

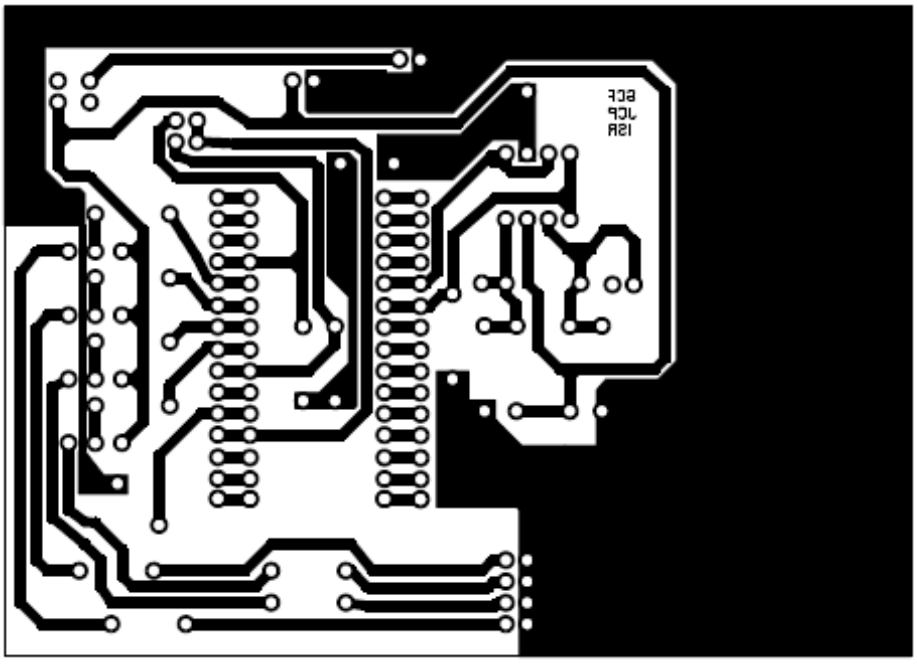
Se encarga de controlar todos los módulos conectados a él.

b. PCB.



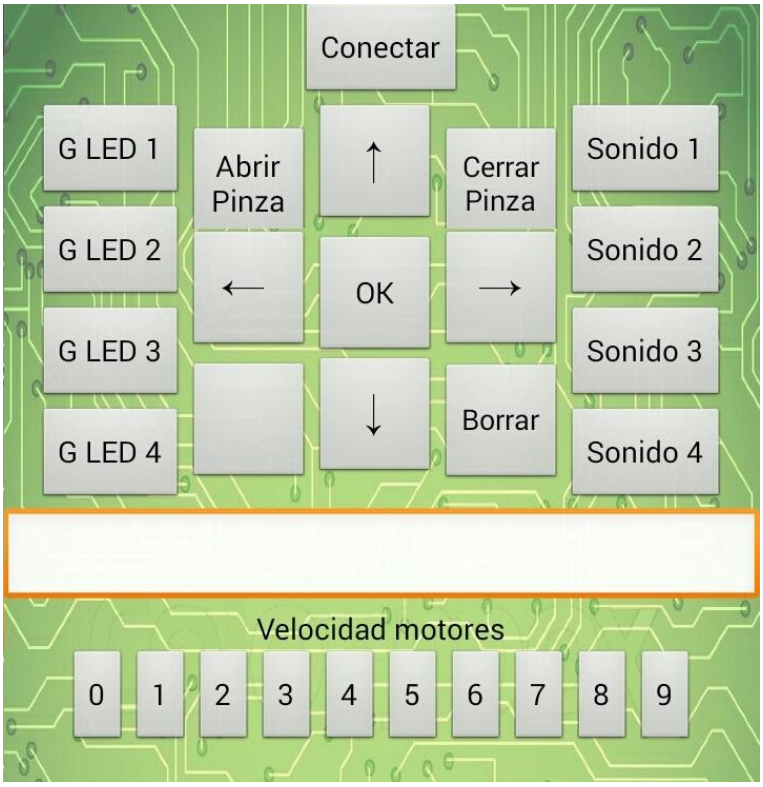
En estas imágenes observamos dos placas, las cuales irán insertadas una sobre otra de forma paralela.

Dado que nos resultaba imposible realizar todas las conexiones en una sola capa, decidimos proceder con dos placas, y así, también conseguir darle al robot un aspecto más robusto, mejorando su estética y a la vez ocultando parte del cableado entre ambas.



c. APK.

Aplicación con la que podremos controlar a **Mebo** realizada en el entorno de **APPInventor 2**.

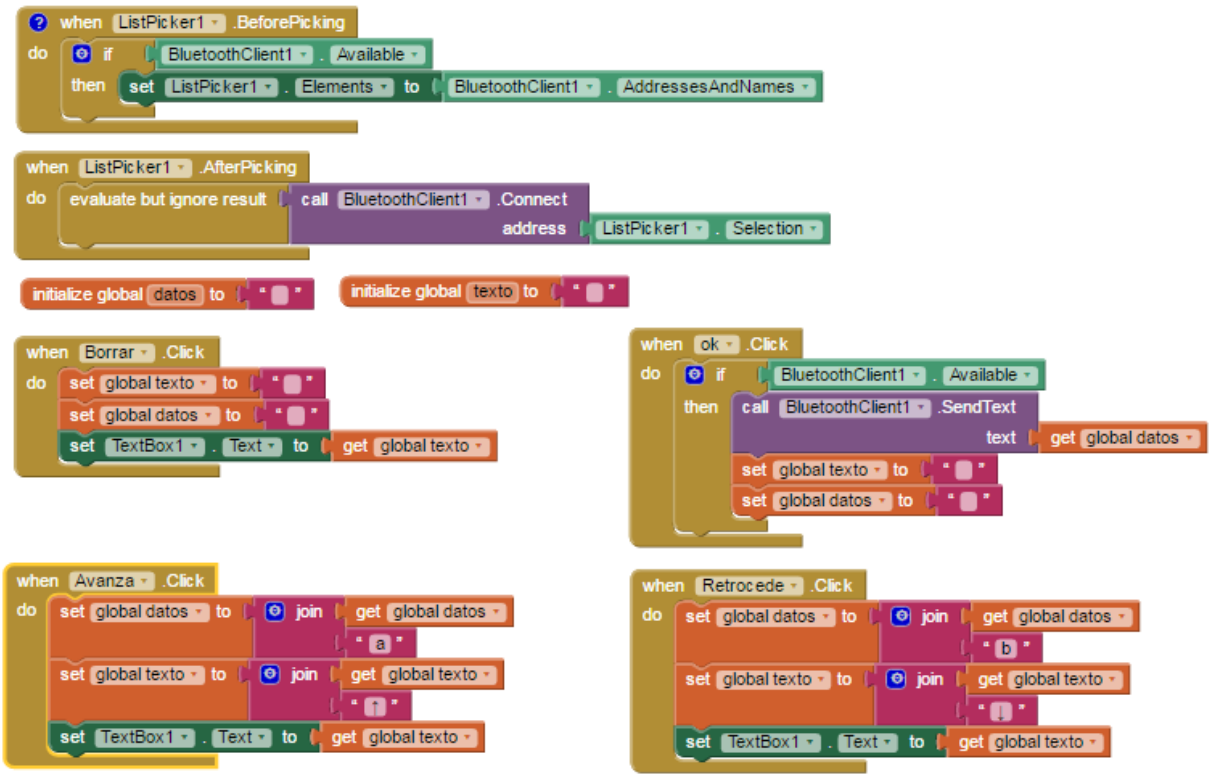


En la imagen observamos el entorno de usuario de la **APK**, en la que podemos encontrar de forma muy intuitiva todas las funciones que podremos realizar con nuestro robot.

En el recuadro blanco, podremos ir distinguiendo todas las instrucciones que introducimos y que, posteriormente, serán enviadas pulsando el **botón OK**.

Para vincular nuestro dispositivo con el robot, debemos conectar nuestro bluetooth del móvil y posteriormente pulsar el **botón Conectar** que vemos en la imagen.

En la imagen que mostrando a continuación podemos ver un resumen de los **bloques** que hacen funcionar la **APK**:



- Los dos primeros bloques se encargan de la conexión por **Bluetooth**.
  Cuando pulsamos el **botón Conectar**, comprueba que el **Bluetooth** del dispositivo esté disponible y nos muestra una lista de los dispositivos a los que estamos vinculados; una vez seleccionado el **Bluetooth** del robot, la conexión de ambos será efectiva.

- Utilizamos dos variables globales **DATOS** y **TEXTO**; en DATOS, se almacenan los caracteres que serán enviados por Bluetooth, y en TEXTO se almacenan los caracteres y símbolos que serán mostrados en el recuadro blanco del entorno de usuario, se almacenan cosas distintas para que lo mostrado al usuario sea entendible.

- El siguiente bloque que vemos se activa cuando pulsamos el **botón OK**, se encarga de enviar la variable DATOS por Bluetooth al robot y posteriormente inicializa las dos variables en blanco para que sean rellenadas de nuevo.

- También dispone de un bloque que se encarga de dejar en blanco las dos variables, al pulsar el botón **Borrar**, pudiendo corregir fallos a la hora de introducirle las instrucciones.

- Los siguientes bloques corresponden al resto de botones disponibles en la **APK**; aunque solo se muestren dos en la imagen, los bloques realizan la misma función pero añadiendo distintos caracteres a las variables. Cuando pulsamos cada botón, añade a la variable **DATOS** y **TEXTO** el carácter correspondiente a dicho botón y, por último, muestra en el recuadro blanco lo que hay almacenado en TEXTO.

d. Programación.

La programación se ha realizado en el entorno de **Arduino**.

El centro de la programación está en el **SWITCH**, este va leyendo los caracteres uno a uno y va realizando una función u otra.

Toda la programación aparece comentada para comprenderla más fácilmente. Más información acerca de la programación en el **ANEXO 1**.

.

5. RECURSOS

Material	Utilidad
Placa fotosensible	Realización de los circuitos impresos del robot
MCP14E5	Alimentación de los motores
Diodos 1N4007	Protección de MCP15E5
Condensadores de 100pF	
Interruptor	Activar o desactivar la alimentación de la batería
Arduino Nano	Control del robot
LEDs	
Módulo bluetooth	Comunicación entre robot y dispositivo móvil
Final de carrera	Detectar la apertura y cierre de las pinza
Resistencias	
Zócalos	Conexión de MCP14E5
Buzzer	Emisión de sonido
Pines hembra y macho	Conexionado
CNY70	Detectar posición de las ruedas
Motores	Movimiento y apertura y cierre de pinza
Rueda	Movimiento del robot
Pinza	Coger objetos
Transistores	Alimentación de LEDs

Herramientas	Utilidad
Taladro	Orificios de PADs
Destornillador	Apriete de tornillos
Insoladora	Revelado de placas
Sierra	Cortar PCBs y distintas piezas
Guillotina	Cortes rectos de PCBs
Agua oxigenada 110 Vol.	Realización de la mezcla para las PCBs
Agua fuerte	Realización de la mezcla para las PCBs
Sosa	Realización de la mezcla para las PCBs
Estaño	Soldadura de componentes
Soldador	Soldadura de componentes

6. SECUENCIA DE OPERACIONES

Pasos	Explicación	Tiempo
1	Pruebas de circuitos en protoboard	8h
2	Diseño de las PCBs	16h
3	Realización de las PCBs	3h
4	Soldado de componentes	4h
5	Montaje de Mebo y diseño de las pinzas	24h
6	Programación de Mebo	30h

7. EVALUACIÓN

La puesta en marcha de este proyecto ha sido bastante satisfactoria, ya que gracias a él hemos podido ser conscientes de todos los conocimientos adquiridos en esta etapa.

A su vez nos ha brindado la oportunidad de poner en práctica nuestras habilidades y continuar aprendiendo sobre el mundo de la robótica, un mundo muy complejo con el cual nos hemos dado cuenta de la importancia de trabajar en equipo.

La experiencia ha despertado aún más nuestro interés en la electrónica, animándonos a adquirir más conocimientos no solo para mejorar este prototipo de robot, sino para llevar a cabo muchas otras ideas que nos han venido a la mente mientras lo realizábamos.

La falta de tiempo y de conocimientos nos ha puesto hándicaps como la carencia de una carcasa ya diseñada que mejoraba su aspecto estético y aportaba rigidez al robot; un display que nos muestre información sobre los movimientos a realizar o cualquier dato almacenado; al igual que nos hubiese gustado que los movimientos de Mebo fuesen aún más precisos.

Dando por concluida la ejecución del proyecto trataremos de estudiar con profundidad los fallos que hemos tenido para una futura mejora del prototipo. Estás mejoras contribuirán a su calidad y como consecuencia directa a un aprendizaje continuo y eficaz.

ANEXO 1

```
//Definir todas las variables con sus puertos
#define M1 10   //Motor 1
#define M1E 9   //Motor 1 Enable
#define M2 6    //Motor 2
#define M2E 5   //Motor 2 Enable
#define BZ 12   //Buzzer
#define P 8     //Pinzas
#define PE 7    //Pinzas Enable
#define FC1 19   //Final de Carrera 1
#define FC2 14   //Final de Carrera 2
#define BZ 12   //Buzzer
#define L1 15   //LEDs 1 Adelante
#define L2 16   //LEDs 2 Atrás
#define L3 17   //LEDs 3 Izquierda
#define L4 18   //LEDs 4 Derecha
#define CNY2 3   //CNY 2
#define CNY1 2   //CNY 1

//Definir variables para la velocidad de cada motor al 50% de 255
float SPD1 = 126.5; //Velocidad para el motor 1
float SPD2 = 125.5; //Velocidad para el motor 2

//Variables de los Encoders de cada motor inicializada a 0
int E1 = 0; //Encoder del motor 1
int E2 = 0; //Encoder del motor 2

//Definir variables de las notas musicales
int Do[5] = {131, 262, 523, 1046, 2093}; // Nota Do
int DoS[5] = {139, 277, 554, 1108, 2217}; //Nota Do#
int Re[5] = {147, 294, 587, 1175, 2349}; //Nota Re
int ReS[5] = {156, 311, 622, 1244, 2489}; //Nota Re#
int Mi[5] = {165, 330, 659, 1319, 2637}; //Nota Mi
int Fa[5] = {175, 349, 698, 1397, 2794}; //Nota Fa
int FaS[5] = {185, 370, 740, 1480, 2960}; //Nota Fa#
int Sol[5] = {196, 392, 784, 1568, 3136}; //Nota Sol
int SolS[5] = {208, 415, 831, 1661, 3322}; //Nota Sol#
int La[5] = {220, 440, 880, 1760, 3520}; //Nota La
int LaS[5] = {233, 466, 932, 1866, 3729}; //Nota La#
int Si[5] = {247, 494, 988, 1976, 3951}; //Nota Si

//Funciones para las interrupciones de los encoders
void Encoder1() //Función para el Encoder 1
{
 E1++; //Contador de pasos para el Encoder 1
}
void Encoder2() //Función para el Encoder 2
{
 E2++; //Contador de pasos para el Encoder 2
}

void setup()
{ //Definir entradas y salidas
```

```
 pinMode(M1, OUTPUT); //IN del motor 1 como salida
 pinMode(M1E, OUTPUT); //Enable del motor 1 como salida
 pinMode(M2, OUTPUT);//IN del motor 2 como salida
 pinMode(M2E, OUTPUT); //Enable del motor 2 como salida
 pinMode(BZ, OUTPUT); //Buzzer como salida
 pinMode(P, OUTPUT); // IN de las pinzas como salida
 pinMode(PE, OUTPUT);  //Enable de las pinzas como salida
 pinMode(FC1, INPUT);  //Final de Carrera 1 como salida
 pinMode(FC2, INPUT);  //Final de Carrera 2 como salida
 pinMode(L1, OUTPUT); // Led derecho
 pinMode(L2, OUTPUT); // Led izquierdo
 pinMode(L3, OUTPUT); // Led Trasero central
 pinMode(L4, OUTPUT); // Led Trasero laterales
 pinMode(CNY2, INPUT);  // CNY2 como entrada
 pinMode(CNY1, INPUT);  // CNY2 como entrada
 attachInterrupt(0, Encoder1, CHANGE); //Definir interrupciones para la función del encoder 1 a
cada cambio de nivel
 attachInterrupt(1, Encoder2, CHANGE); //Definir interrupciones para la función del encoder 2 a
cada cambio de nivel
 Serial.begin(9600); //Conexión Bluetooth
}

void adelante() //Función para que ande hacia adelante
{
 while (E1 < 65) // Mientras que el encoder 1 cambie 65 veces
 {
   analogWrite(M1E, SPD1); // Activa Motor Derecho a la velocidad escogida, 50% si no elije
nada
   analogWrite(M2E, SPD2); // Activa Motor Izquierda a la velocidad escogida, 50% si no elije
nada
   digitalWrite(M1, HIGH); // Motor Derecho hacia adelante
   digitalWrite(M2, HIGH); // Motor Izquierda hacia adelante
 }
 digitalWrite(M1E, LOW); // Desactiva Motor Derecho
 digitalWrite(M2E, LOW); // Desactiva Motor Izquierda
}

void atras() //Función para que ande hacia atrás
{
 while (E1 < 65) // Mientras que el encoder 1 cambie 65 veces
 {
   analogWrite(M1E, SPD1); // Activa Motor Derecho a la velocidad escogida, 50% si no elije
nada
   analogWrite(M2E, SPD2); // Activa Motor Izquierda a la velocidad escogida, 50% si no elije
nada
   digitalWrite(M2, LOW); // Motor Izquierda hacia atras
   digitalWrite(M1, LOW); // Motor Derecho hacia atras
 }
 digitalWrite(M1E, LOW); // Desactiva Motor Derecho
 digitalWrite(M2E, LOW); // Desactiva Motor Izquierda
}

void derecha() //Función para que ande hacia la derecha
{
 while (E2 < 2) // Mientras que el encoder 2 cambie 2 vez; para que se ajuste a la siguente
celda
```

```
 {
   analogWrite(M1E, SPD1); // Activa Motor Derecho a la velocidad escogida, 50% si no elije
nada
   analogWrite(M2E, SPD2); // Activa Motor Izquierda a la velocidad escogida, 50% si no elije
nada
   digitalWrite(M1, HIGH); // Motor Derecho hacia adelante
   digitalWrite(M2, HIGH); // Motor Izquierda hacia adelante
 }
 digitalWrite(M1E, LOW); // Desactiva Motor Derecho
 digitalWrite(M2E, LOW); // Desactiva Motor Izquierda
 while (E1 < 58) // Mientras que el encoder 1 cambie 58 vez
 {
   analogWrite(M1E, SPD1); // Activa Motor Derecho a la velocidad escogida, 50% si no elije
nada
   digitalWrite(M1, HIGH); // Motor Derecho hacia adelante
 }
 digitalWrite(M1E, LOW); // Desactiva Motor Derecho
}

void izquierda() //Función para que ande hacia la izquierda
{
 while (E1 < 2) // Mientras que el encoder 2 cambie 2 vez; para que se ajuste a la siguente
celda
 {
   analogWrite(M1E, SPD1); // Activa Motor Derecho a la velocidad escogida, 50% si no elije
nada
   analogWrite(M2E, SPD2); // Activa Motor Izquierda a la velocidad escogida, 50% si no elije
nada
   digitalWrite(M1, HIGH); // Motor Derecho hacia adelante
   digitalWrite(M2, HIGH); // Motor Izquierda hacia adelante
   digitalWrite(M1E, LOW); // Desactiva Motor Derecho
   digitalWrite(M2E, LOW); // Desactiva Motor Izquierda
 }
 while (E2 < 58) // Mientras que el encoder 2 cambie 58 vez
 {
   analogWrite(M2E, SPD2); // Activa Motor Izquierda a la velocidad escogida, 50% si no elije
nada
   digitalWrite(M2, HIGH); // Motor Izquierda hacia adelante
 }
 digitalWrite(M2E, LOW); // Desactiva Motor Izquierda
}

void cerrar_pinzas() //Función para cerrar pinzas
{
 while (digitalRead(FC2) == 1) // Mientras el Final de Carrera no esté pulsado
 {
   digitalWrite(PE, HIGH); // Activa Pinzas
   digitalWrite(P, HIGH); // Pinzas para cerrar
 }

 if (digitalRead(FC2) == 0) // Si el Final de Carrera está pulsado
 {
   digitalWrite(PE, LOW); // Desactivar Pinzas
   nota(Si[2], 200); // Canción para avisar que ha cerrado la pinza y se ha parado el motor
   nota(La[2], 200);
   nota(Sol[2], 200);
```

```
  noTone(BZ); delay(1000);
 }
}

void abrir_pinzas() //Función para abrir pinzas
{
 while (digitalRead(FC1) == 0) // Mientras el Final de Carrera no esté pulsado
 {
  digitalWrite(PE, HIGH); // Activa Pinzas
  digitalWrite(P, LOW); // Pinzas para abrir
 }
 if (digitalRead(FC1) == 1)  // Si el Final de Carrera está pulsado
 {
  digitalWrite(PE, LOW); // Desactivar Pinzas
 }

 delay(30); //para asegurar el abrir se le repite el proceso

 while (digitalRead(FC1) == 0)  // Mientras el Final de Carrera no esté pulsado
 {
  digitalWrite(PE, HIGH); // Activa Pinzas
  digitalWrite(P, LOW); // Pinzas para abrir
 }

 if (digitalRead(FC1) == 1) // Si el Final de Carrera está pulsado
 {
  digitalWrite(PE, LOW); // Desactivar Pinzas
  nota(Si[2], 200); //Canción para avisar que ha abierto la pinza y se ha parado el motor
  nota(La[2], 200);
  nota(Sol[2], 200);
  noTone(BZ); delay(1000);
 }
}

void nota(int frec, int t) //Función para definir la música usando la frecuencia y el tiempo
{
 tone(BZ, frec); // Se le declara el pin del Buzzer a esta función
 delay(t);
}

void loop()
{
 char dato = Serial.read();  //lee el puerto serie (Rx y Tx)
 digitalWrite(L3, HIGH); // Se enciende el LED trasero centrar que simula la luz de encendido
 digitalWrite(L4, HIGH); // Se enciende los LEDs trasero laterales que simula la luz de parada
 switch (dato) // Se declara un switch para determinar varios carácteres que recoge por
bluetooth
 {
  case 'a': // Caso para ir adelante
    digitalWrite(L1, HIGH); // Se enciende los LEDs delanteros para empezar a andar
    digitalWrite(L2, HIGH);
    digitalWrite(L4, LOW); // Se apaga los LEDs trasero laterales que simula la luz de parada
para empezar a andar
    adelante(); // Llama la función ''adelante''
    digitalWrite(L1, LOW); // Se apagan los LEDs delanteros
    digitalWrite(L2, LOW);
```

```
    digitalWrite(L4, HIGH); //Se enciende los LEDs trasero laterales que simula la luz de parada
    E1 = 0; // Se inicializa los cuentapasos a 0
    E2 = 0;
    break;

  case 'b': // Caso para ir atrás
    digitalWrite(L4, LOW); //  Parpadeo del los LEDs traseros laterales para indicar que va a ir
marcha atrás
    delay(200);
    digitalWrite(L4, HIGH);
    delay(200);
    digitalWrite(L4, LOW);
    delay(200);
    digitalWrite(L4, HIGH);
    delay(200);
    digitalWrite(L4, LOW);
    delay(200);
    atras();
    digitalWrite(L4, HIGH);
    E1 = 0; // Se inicializa los cuentapasos a 0
    E2 = 0;
    break;

  case 'c': // Caso para girar a la izquierda
    digitalWrite(L2, HIGH); //Se enciende el LED de la izquierda para girar a la izquierda
    digitalWrite(L4, LOW); // Se apaga los LEDs trasero laterales que simula la luz de parada
para empezar a andar
    izquierda(); // Llama a la función de giro a la izquierda
    digitalWrite(L4, HIGH); //Se enciende los LEDs trasero laterales que simula la luz de parada
    digitalWrite(L2, LOW); // Se apaga el LED de giro a la izquierda
    E1 = 0; // Se inicializa los cuentapasos a 0
    E2 = 0;
    while (E2 < 40) // Mientras que el encoder 2 cambie 40 veces para ir hacia atrás y ajustarse
al giro
    {
      analogWrite(M1E, SPD1); // Activa Motor Derecho a la velocidad escogida, 50% si no elije
nada
      analogWrite(M2E, SPD2); // Activa Motor Izquierda a la velocidad escogida, 50% si no elije
nada
      digitalWrite(M1, LOW); // Motor Derecho hacia atrás
      digitalWrite(M2, LOW); // Motor Izquierda hacia atrás
    }
    digitalWrite(M1E, LOW); // Desactiva Motor Derecho
    digitalWrite(M2E, LOW); // Desactiva Motor Izquierda
    E1 = 0; // Se inicializa los cuentapasos a 0
    E2 = 0;
    break;

  case 'd': // Caso para girar a la derecha
    digitalWrite(L4, LOW); // Se apaga los LEDs trasero laterales que simula la luz de parada
para empezar a andar
    digitalWrite(L1, HIGH); //Se enciende el LED de la derecha para girar a la derecha
    derecha(); // Llama a la función de giro a la derecha
    digitalWrite(L4, HIGH); //Se enciende los LEDs trasero laterales que simula la luz de
    digitalWrite(L1, LOW); // Se apaga el LED de giro a la derecha
    E1 = 0;// Se inicializa los cuentapasos a 0
```

```
    E2 = 0;
    while (E2 < 40) // Mientras que el encoder 2 cambie 40 veces para ir hacia atrás y ajustarse
al giro
    {
      analogWrite(M1E, SPD1); // Activa Motor Derecho a la velocidad escogida, 50% si no elije
nada
      analogWrite(M2E, SPD2); // Activa Motor Izquierda a la velocidad escogida, 50% si no elije
nada
      digitalWrite(M1, LOW); // Motor Derecho hacia atrás
      digitalWrite(M2, LOW); // Motor Izquierda hacia atrás
    }
    digitalWrite(M1E, LOW); // Desactiva Motor Derecho
    digitalWrite(M2E, LOW); // Desactiva Motor Izquierda
    E1 = 0; // Se inicializa los cuentapasos a 0
    E2 = 0;
    break;

  case 'e': // Caso para abrir las pinzas
    digitalWrite(L1, HIGH); // Parpadeo del todos los LEDs para indicar que va a mover las
pinzas
    digitalWrite(L2, HIGH);
    digitalWrite(L3, HIGH);
    digitalWrite(L4, HIGH);
    delay(200);
    digitalWrite(L1, LOW);
    digitalWrite(L2, LOW);
    digitalWrite(L3, LOW);
    digitalWrite(L4, LOW);
    delay(200);
    digitalWrite(L1, HIGH);
    digitalWrite(L2, HIGH);
    digitalWrite(L3, HIGH);
    digitalWrite(L4, HIGH);
    delay(200);
    digitalWrite(L1, LOW);
    digitalWrite(L2, LOW);
    digitalWrite(L3, LOW);
    digitalWrite(L4, LOW);
    delay(200);
    abrir_pinzas();
    digitalWrite(L4, HIGH);
    break;

  case 'f':  // Función para cerrar las pinzas
    digitalWrite(L1, HIGH); // Parpadeo del todos los LEDs para indicar que va a mover las
pinzas
    digitalWrite(L2, HIGH);
    digitalWrite(L3, HIGH);
    digitalWrite(L4, HIGH);
    delay(200);
    digitalWrite(L1, LOW);
    digitalWrite(L2, LOW);
    digitalWrite(L3, LOW);
    digitalWrite(L4, LOW);
    delay(200);
    digitalWrite(L1, HIGH);
```

```
  digitalWrite(L2, HIGH);
  digitalWrite(L3, HIGH);
  digitalWrite(L4, HIGH);
  delay(200);
  digitalWrite(L1, LOW);
  digitalWrite(L2, LOW);
  digitalWrite(L3, LOW);
  digitalWrite(L4, LOW);
  delay(200);
  cerrar_pinzas();
  digitalWrite(L4, HIGH);
  break;

case 'k':// Música 1 Entre dos aguas (Paco de Lucía)
  nota(La[1], 400); noTone(BZ); delay(400);
  nota(Mi[1], 400); noTone(BZ); delay(400);
  nota(La[1], 400); noTone(BZ); delay(200);
  nota(Mi[1], 200); noTone(BZ); delay(200);
  nota(La[1], 200); noTone(BZ); delay(200);
  nota(LaS[1], 100); noTone(BZ); delay(100);
  nota(Si[1], 400); noTone(BZ); delay(400);
  nota(FaS[1], 400); noTone(BZ); delay(400);
  nota(Si[1], 400); noTone(BZ); delay(200);
  nota(FaS[1], 200); noTone(BZ); delay(200);
  nota(Si[1], 200); noTone(BZ); delay(200);
  nota(LaS[1], 100); noTone(BZ); delay(100);
  nota(La[1], 400); noTone(BZ); delay(400);
  nota(Mi[1], 400); noTone(BZ); delay(400);
  nota(La[1], 400); noTone(BZ); delay(400);
  break;

case 'l':// Música 2 Tema Principal (Star Wars)
  nota(Re[1], 150); noTone(BZ); delay(50);
  nota(Re[1], 150); noTone(BZ); delay(50);
  nota(Re[1], 150); noTone(BZ); delay(50);
  nota(Sol[1], 900); noTone(BZ); delay(150);
  nota(Re[2], 900); noTone(BZ); delay(50);
  nota(Do[2], 150); noTone(BZ); delay(50);
  nota(Si[1], 150); noTone(BZ); delay(50);
  nota(La[1], 150); noTone(BZ); delay(50);
  nota(Sol[2], 900); noTone(BZ); delay(150);
  nota(Re[2], 900); noTone(BZ); delay(100);
  nota(Do[2], 150); noTone(BZ); delay(50);
  nota(Si[1], 150); noTone(BZ); delay(50);
  nota(La[1], 150); noTone(BZ); delay(50);
  nota(Sol[2], 900); noTone(BZ); delay(150);
  nota(Re[2], 900); noTone(BZ); delay(100);
  nota(Do[2], 150); noTone(BZ); delay(50);
  nota(Si[1], 150); noTone(BZ); delay(50);
  nota(Do[2], 150); noTone(BZ); delay(50);
  nota(La[1], 1200); noTone(BZ); delay(2000);
  break;

case 'm':// Música 3 Marcha del imperio (Star Wars)
  nota(Sol[1], 500); noTone(BZ); delay(100);
  nota(Sol[1], 500); noTone(BZ); delay(100);
```

```
  nota(Sol[1], 500); noTone(BZ); delay(100);
  nota(ReS[1], 500); noTone(BZ); delay(1);
  nota(LaS[1], 125); noTone(BZ); delay(25);
  nota(Sol[1], 500); noTone(BZ); delay(100);
  nota(ReS[1], 500); noTone(BZ); delay(1);
  nota(LaS[1], 125); noTone(BZ); delay(25);
  nota(Sol[1], 500); noTone(BZ); delay(2000);
  break;

case 'n'://Música 4 HARRY POTTER
  nota(Si[1], 500);
  nota(Mi[2], 1000);
  nota(Sol[2], 250);
  nota(FaS[2], 250);
  nota(Mi[2], 1000);
  nota(Si[2], 500);
  nota(La[2], 1250);
  nota(FaS[2], 1000);
  nota(Si[1], 500);
  nota(Mi[2], 1000);
  nota(Sol[2], 250);
  nota(FaS[2], 250);
  nota(Re[2], 1000);
  nota(Mi[2], 500 );
  nota(Si[1], 1000 );
  noTone(BZ); delay(1000);
  nota(Si[1], 500);
  nota(Mi[2], 1000);
  nota(Sol[2], 250);
  nota(FaS[2], 250);
  nota(Mi[2], 1000);
  nota(Si[2], 500);
  nota(Re[3], 1000);
  nota(DoS[3], 500);
  nota(Do[3], 1000);
  nota(La[2], 500);
  nota(Do[3], 1000);
  nota(Si[2], 250);
  nota(LaS[2], 250);
  nota(Si[1], 1000);
  nota(Sol[2], 500);
  nota(Mi[2], 1000);
  noTone(BZ); delay(2000);
  break;

case 'g': // Grupo Leds 1
  digitalWrite(L1, HIGH);
  delay(200);
  digitalWrite(L1, LOW);
  digitalWrite(L4, HIGH);
  delay(200);
  digitalWrite(L4, LOW);
  digitalWrite(L2, HIGH);
  delay(200);
  digitalWrite(L2, LOW);
  digitalWrite(L3, HIGH);
```

```
  delay(200);
  digitalWrite(L3, LOW);
  digitalWrite(L1, HIGH);
  delay(200);
  digitalWrite(L1, LOW);
  digitalWrite(L4, HIGH);
  delay(200);
  digitalWrite(L4, LOW);
  digitalWrite(L2, HIGH);
  delay(200);
  digitalWrite(L2, LOW);
  digitalWrite(L3, HIGH);
  delay(200);
  digitalWrite(L3, LOW);
  digitalWrite(L1, HIGH);
  delay(200);
  digitalWrite(L1, LOW);
  digitalWrite(L4, HIGH);
  delay(200);
  digitalWrite(L4, LOW);
  digitalWrite(L2, HIGH);
  delay(200);
  digitalWrite(L2, LOW);
  digitalWrite(L3, HIGH);
  delay(200);
  digitalWrite(L3, LOW);
  delay(200);
  break;

case 'h': //Grupo Leds2
  digitalWrite(L1, HIGH);
  delay(200);
  digitalWrite(L3, HIGH);
  digitalWrite(L4, HIGH);
  delay(200);
  digitalWrite(L1, LOW);
  digitalWrite(L2, HIGH);
  delay(200);
  digitalWrite(L3, LOW);
  digitalWrite(L4, LOW);
  delay(200);
  digitalWrite(L2, LOW);
  delay(200);
  digitalWrite(L2, HIGH);
  delay(200);
  digitalWrite(L3, HIGH);
  digitalWrite(L4, HIGH);
  delay(200);
  digitalWrite(L1, HIGH);
  digitalWrite(L2, LOW);
  delay(200);
  digitalWrite(L3, LOW);
  digitalWrite(L4, LOW);
  delay(200);
  digitalWrite(L1, LOW);
  delay(200);
```

```
  break;

case 'i': //Grupo Leds3
  digitalWrite(L1, HIGH);
  delay(200);
  digitalWrite(L3, HIGH);
  delay(200);
  digitalWrite(L1, LOW);
  digitalWrite(L2, HIGH);
  delay(200);
  digitalWrite(L3, LOW);
  digitalWrite(L4, HIGH);
  delay(200);
  digitalWrite(L2, LOW);
  digitalWrite(L1, HIGH);
  delay(200);
  digitalWrite(L4, LOW);
  digitalWrite(L3, HIGH);
  delay(200);
  digitalWrite(L1, LOW);
  digitalWrite(L2, HIGH);
  delay(200);
  digitalWrite(L3, LOW);
  digitalWrite(L4, HIGH);
  delay(200);
  digitalWrite(L2, LOW);
  digitalWrite(L1, HIGH);
  delay(200);
  digitalWrite(L4, LOW);
  digitalWrite(L3, HIGH);
  delay(200);
  digitalWrite(L1, LOW);
  digitalWrite(L2, HIGH);
  delay(200);
  digitalWrite(L3, LOW);
  digitalWrite(L4, HIGH);
  delay(200);
  digitalWrite(L2, LOW);
  digitalWrite(L1, HIGH);
  delay(200);
  digitalWrite(L4, LOW);
  delay(200);
  digitalWrite(L1, LOW);
  delay(200);
  break;

case 'j':// Grupo Leds4
  digitalWrite(L1, HIGH);
  delay(200);
  digitalWrite(L2, HIGH);
  delay(200);
  digitalWrite(L1, LOW);
  digitalWrite(L3, HIGH);
  delay(200);
  digitalWrite(L2, LOW);
  digitalWrite(L4, HIGH);
```

```
  delay(200);
  digitalWrite(L3, LOW);
  delay(200);
  digitalWrite(L4, LOW);
  delay(200);
  digitalWrite(L4, HIGH);
  delay(200);
  digitalWrite(L3, HIGH);
  delay(200);
  digitalWrite(L4, LOW);
  digitalWrite(L2, HIGH);
  delay(200);
  digitalWrite(L3, LOW);
  digitalWrite(L1, HIGH);
  delay(200);
  digitalWrite(L2, LOW);
  delay(200);
  digitalWrite(L1, LOW);
  delay(200);
  break;

case '0': // Velocidad 0
  SPD1 = 0;
  SPD2 = 0;
  break;

case '1': // Velocidad 1
  SPD1 = 50;
  SPD2 = 51;
  break;

case '2': // Velocidad 2
  SPD1 = 75.5;
  SPD2 = 76.5;
  break;

case '3': // Velocidad 3
  SPD1 = 101;
  SPD2 = 102;
  break;

case '4':// Velocidad 4 (50%)
  SPD1 = 126.5;
  SPD2 = 127.5;
  break;

case '5':// Velocidad 5
  SPD1 = 152;
  SPD2 = 153;
  break;

case '6':// Velocidad 6
  SPD1 = 177.5;
  SPD2 = 178.5;
  break;
```

```
  case '7':// Velocidad 7
    SPD1 = 203;
    SPD2 = 204;
    break;

  case '8':// Velocidad 8
    SPD1 = 228.5;
    SPD2 = 229.5;
    break;

  case '9':// Velocidad 9
    SPD1 = 254;
    SPD2 = 255;
    break;
 }
}
```