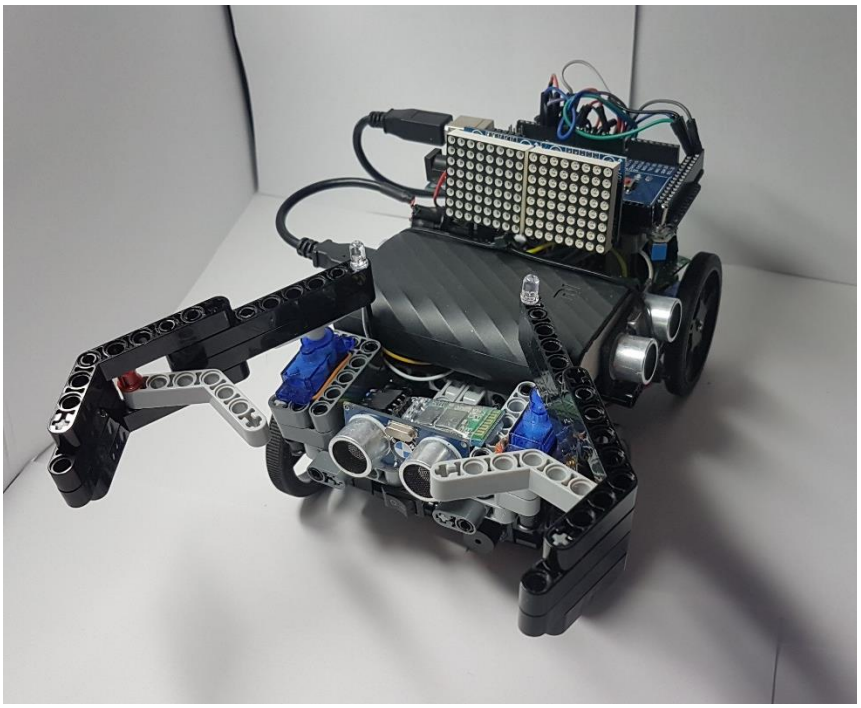
UNIVERSIDAD DE GRANADA

ROBOT BASADO EN ARDUINO



INGENIERÍA ELECTRÓNICA INDUSTRIAL

ALBERTO BARROSO LÓPEZ

INFORME DEL PROYECTO

BREVE DESCRIPCIÓN DEL PROPÓSITO DEL PROYECTO

Se trata de un robot formado por dos Arduinos: Arduino MEGA y Arduino UNO.

- El **Arduino MEGA** tendrá dos modos seleccionables mediante un interruptor con resistencia de pull-up para evitar así las posibles interferencias. Si se encuentra a nivel alto este interruptor, el robot entrará en *modo esquiva-obstáculos*, el cual autónomamente, mediante los tres sensores ultrasónicos de distancia y los cuatro servomotores de rotación continua, será capaz de moverse sin interceptar ningún objeto. Si por el contrario el interruptor se encuentra a nivel bajo, el robot se encontrará en el *modo control remoto mediante mando infrarrojos.* Por medio de este modo, el usuario es capaz de interaccionar con el robot a través de un mando infrarrojo, pudiendo desplazar el robot en la dirección que se desee y también pudiendo accionar las pinzas incorporadas en el robot para desplazar objetos.

- El **Arduino UNO** basará su funcionalidad en base a la conexión de un módulo Bluetooth, al cual se le enviará desde un teléfono móvil, equipado con esta tecnología, las funciones a realizar. Entre ellas estará el control de dos matrices led 8x8 en las cuales aparecerá un texto en scrolling y diversas figuras, un buzzer que será posible seleccionar la canción a reproducir y 2 led's (uno rojo y otro azul) que se combinaran con el buzzer (simulando el sonido de una sirena) para hacer un *modo policía*.

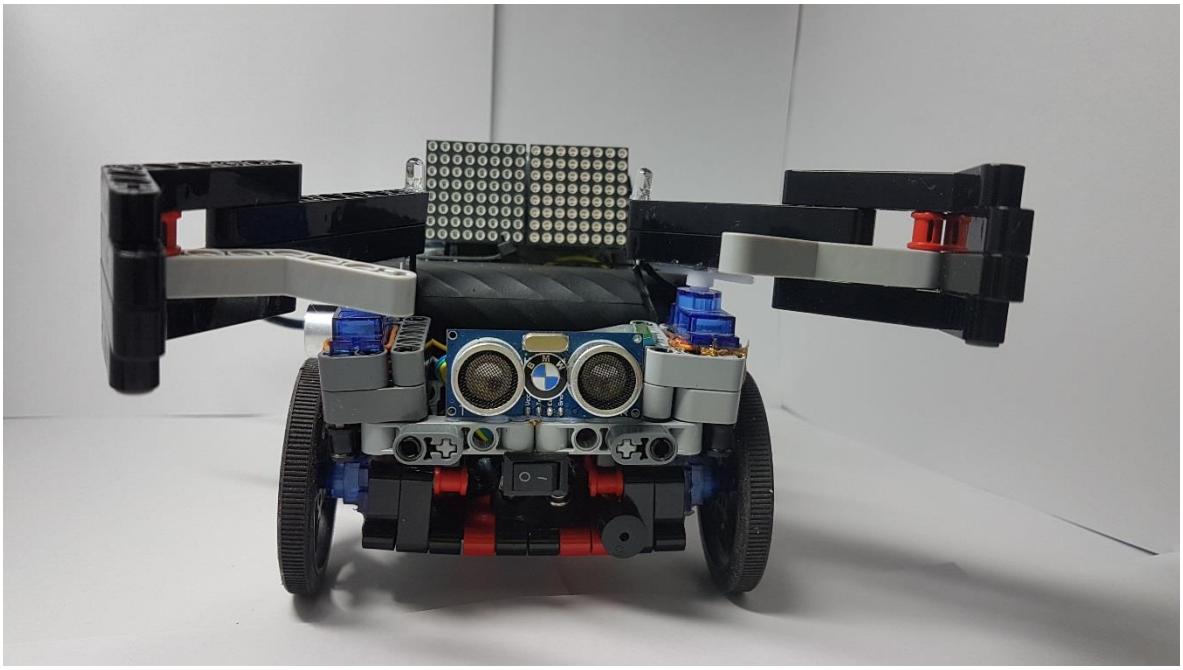
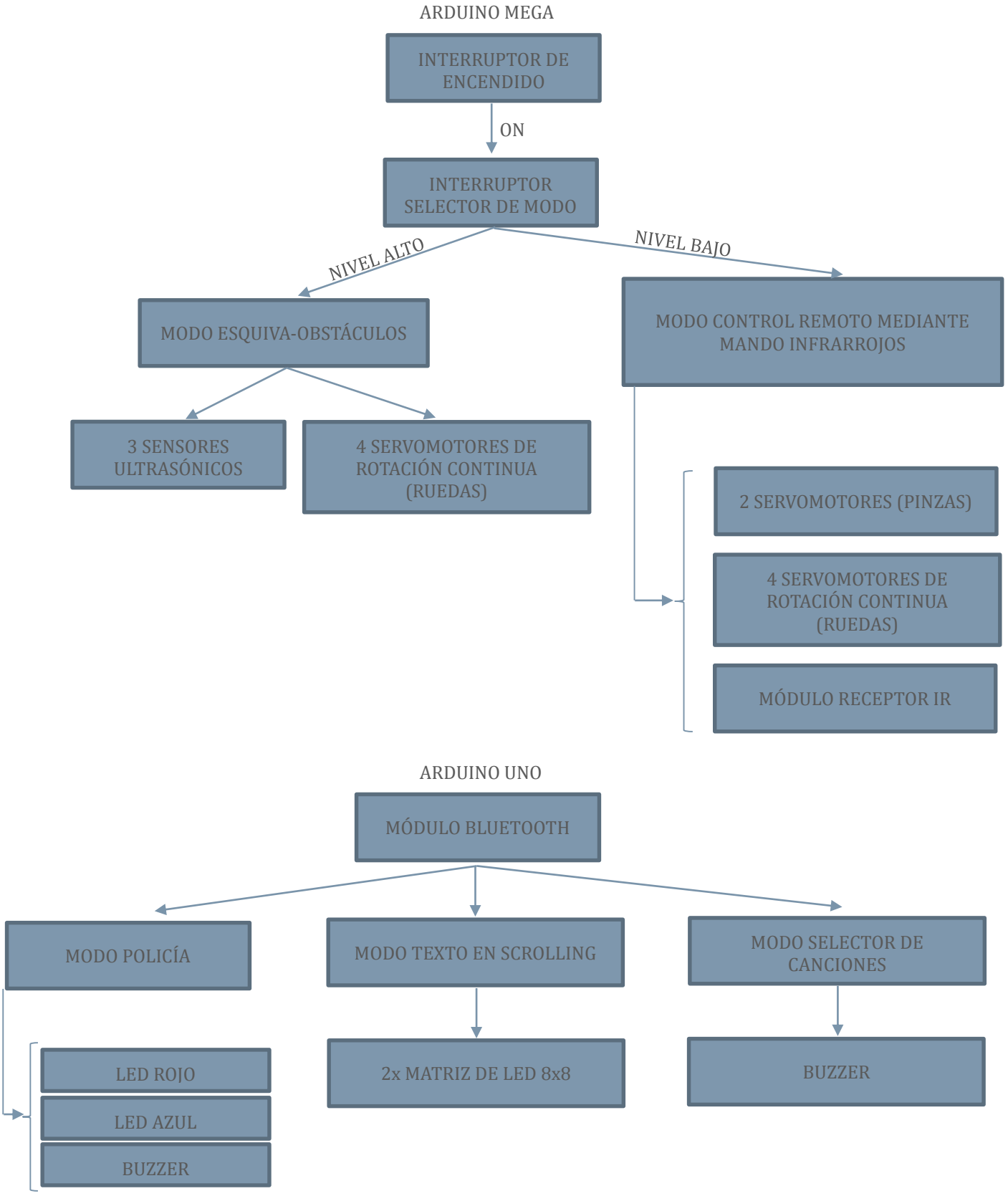
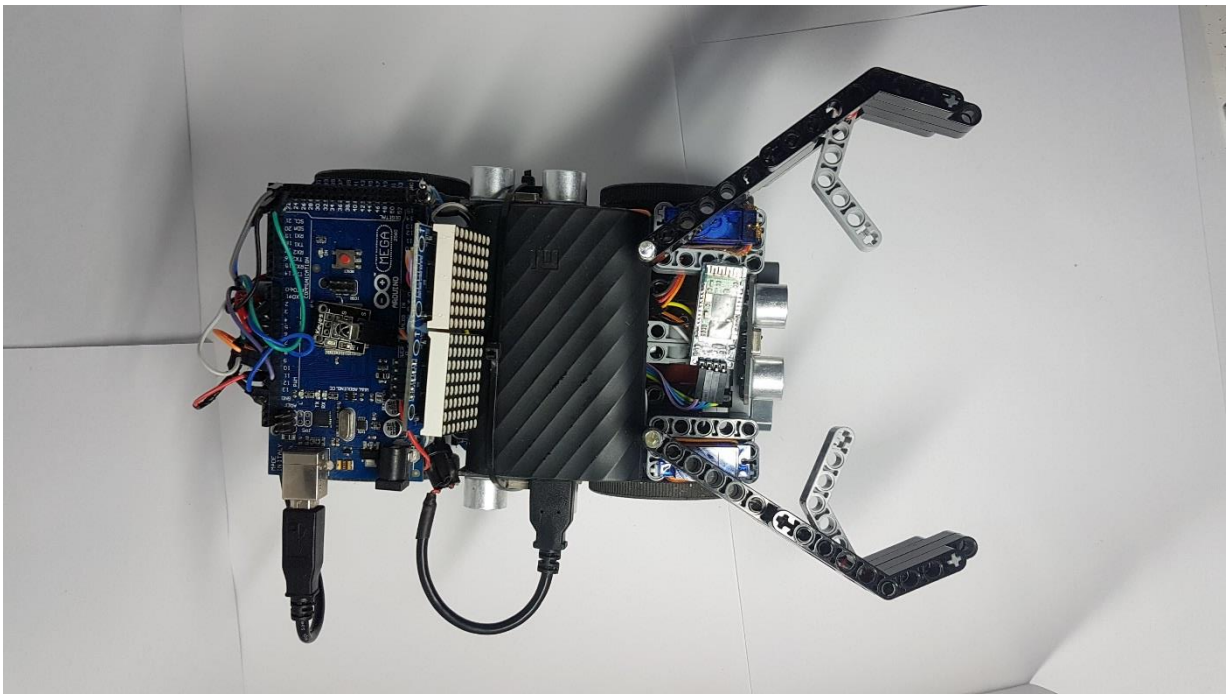


DIAGRAMA DE FLUJO



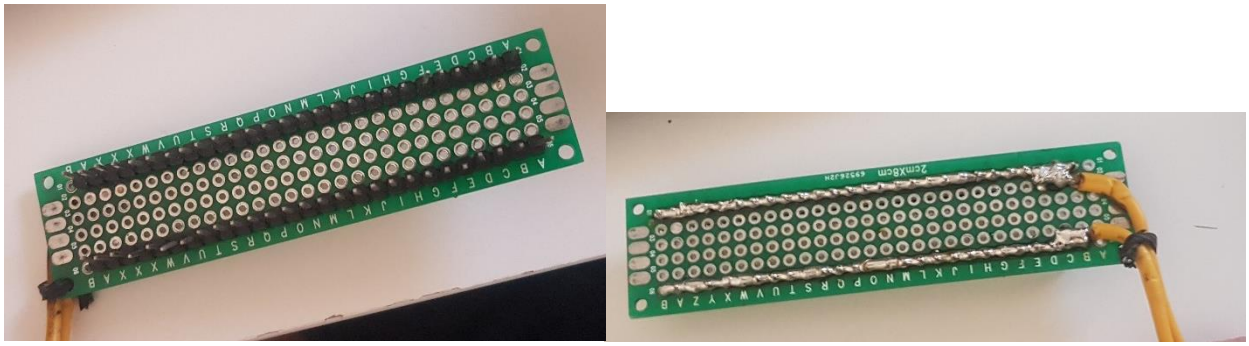
MATERIALES EMPLEADOS Y PRESUPUESTO

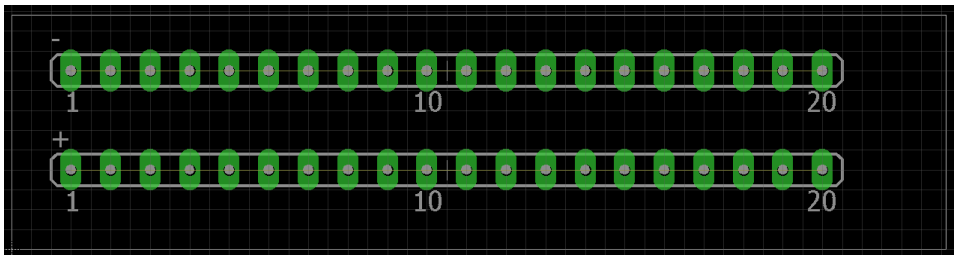
CANTIDAD	DESCRIPCIÓN	PRECIO UNIDAD	PRECIO TOTAL
1	ARDUINO UNO	3,10€	3,10€
1	ARDUINO MEGA	6,25€	6,25€
2	MATRIZ LED 8X8 + CONTROLADOR MAX7219	1,30€	2,60€
4	SERVOMOTOR DE ROTACIÓN CONTINUA (POLOLU-1053)	4,90€	19,60€
2	PACK 2 RUEDAS PARA SERVOMOTOR ROTACIÓN CONTINUA (POLOLU-1420)	7,95€	15,90€
2	SERVOMOTOR TOWER PRO 9G (SG90)	0,98€	1,96€
3	SENSOR DE DISTANCIA ULTRASÓNICO (HC-SR04)	0,90€	2,70€
1	MODULO BLUETOOTH (HC-06)	2,70€	2,70€
1	BUZZER	0,20€	0,20€
1	LED AZUL	0,20€	0,20€
1	LED ROJO	0,20€	0,20€
1	RESISTENCIA PULL-UP 20KΩ 1/4W	0,05€	0,05€
1	INTERRUPTOR ON/OFF	0,30€	0,30€
1	PLACA PERFORADA 8cmx2cm	0,25€	0,25€
2	TIRA DE 24 PINES	0,12€	0,24€
1	POWERBANK XIAOMI 5V 10.000mAh	15€	15€
1	MÚLTIPLES PIEZAS LEGO PARA ESTRUCTURA DEL ROBOT	14,20€	14,20€
47	CABLES DE CONEXIÓN	0,03€	1,41€
		TOTAL:	**86,86€**



PROBLEMAS Y SOLUCIONES. OPTIMIZACIONES.

A lo largo del proyecto se han encontrado múltiples problemas como, por ejemplo, debido al gran número de sensores/actuadores instalados en el robot, todos estos necesitan una alimentación a 5V y GRND. En total 10 sensores/actuadores, teniendo solo 2 pines de 5V y 2 pines de GRND en el Arduino, aparte de su limitación de consumo máximo de 500mA. Debido a esto, se decidió fabricar una placa de alimentación con 5V y GRND directos a la batería powerbank de 10.000mAh, solucionándose dicho problema.





En esta imagen se muestra el diseño de la placa de alimentación realizado con el programa EAGLE PCB 7.3.

También se ha procedido a optimizar el código de programa mediante la creación de múltiples funciones y la modificación de librerías, con el objetivo de eliminar de dichas librerías información que no se fuera a utilizar.

La librería utilizada para los protocolos IR, fue optimizada para que solo cargará el protocolo NEC, ahorrando un gran porcentaje de memoria en el Arduino.

```
El Sketch usa 11466 bytes (35%) del espacio de almacenamiento de programa. El máximo es 3225
Las variables Globales usan 925 bytes (45%) de la memoria dinámica, dejando 1123 bytes para
```

Librería cargando todos los protocolos.

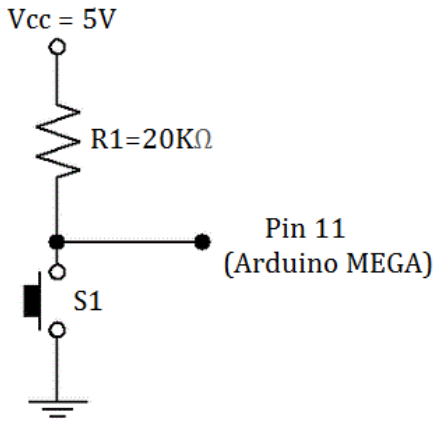
```
El Sketch usa 4832 bytes (14%) del espacio de almacenamiento de programa. El máximo es 3225
Las variables Globales usan 453 bytes (22%) de la memoria dinámica, dejando 1595 bytes para
```

Librería cargando únicamente el protocolo NEC.

Como se puede observar el sketch utilizaría 6.634 bytes menos, un 21% menos de uso de la memoria.

Debido a que la *powerbank* no tiene botón on/off se tuvo que fabricar un interruptor de encendido y apagado, con sus respectivos cables para alimentar al Arduino, a la placa de alimentación y a la propia *powerbank*.

Ya que la idea principal del robot era que tuviese dos modos principales mediante el Arduino MEGA, se diseñó un interruptor con resistencia de pull-up de 20KΩ, para evitar posibles interferencias y que estas provocaran un cambio de modo inesperado.



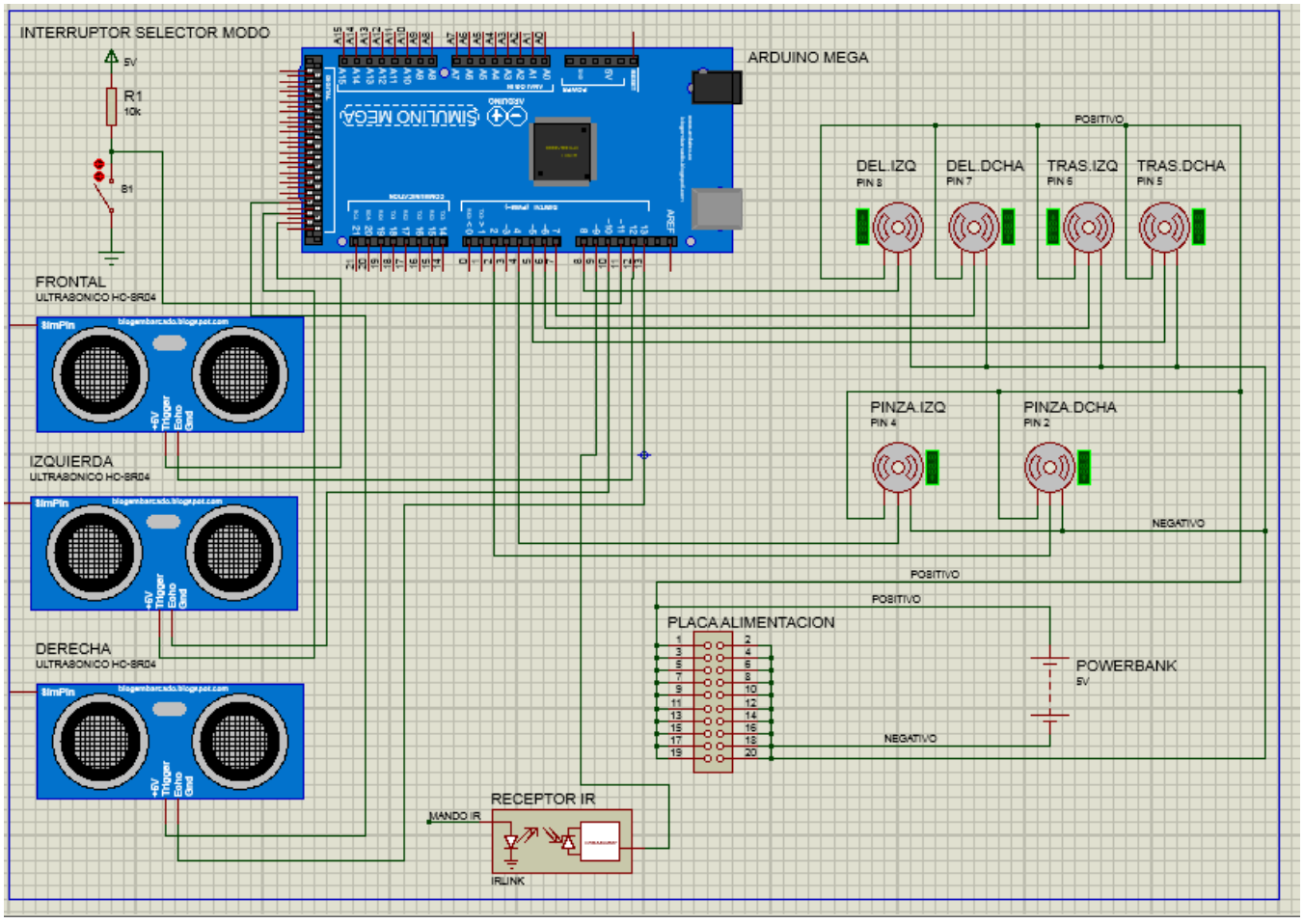
Por último, al usar el protocolo NEC se encontró un problema ya que dicho protocolo es especial cuando se deja pulsado un botón ininterrumpidamente. Esto supuso un problema ya que dejando pulsado el botón '1', el botón '2' o cualquier botón, enviaba el mismo identificador (0xFFFFFFFF). Debido a que el principal objetivo del mando infrarrojo era tener un movimiento fluido al dejar pulsado cualquier botón de dirección, se utilizó una nueva variable que guardase el valor del último valor recibido por infrarrojos y lo comparara al nuevo recibido. En caso de que recibiese el valor *del botón pulsado continuamente (0xFFFFFFFF)*, se reescribiría con el valor anterior.

```
if (miReceptor.getResults()) {
      miDecod.decode();
      Serial.println(miDecod.value,HEX);
        if(miDecod.protocolNum==MY_PROTOCOL) { //Comprobacion protocolo NEC
            if(miDecod.value==0xFFFFFFFF)     //Sustitucion de Repeticion por valor correcto
              miDecod.value=Anterior;

         switch(miDecod.value) {
           case IZQUIERDA:   giroIZQ();      break;

           case DERECHA:     giroDCHA();     break;

           case DELANTE:     avance();       break;

           case DETRAS:      retroceso();    break;

           case PINZAS:
               if(pinzas==1){
               abrirPINZAS();
               pinzas=!pinzas;}
               else if (pinzas==0){
               cerrarPINZAS();
               pinzas=!pinzas;}
           break;}

    Anterior=miDecod.value;}
 miReceptor.enableIRIn();}
```

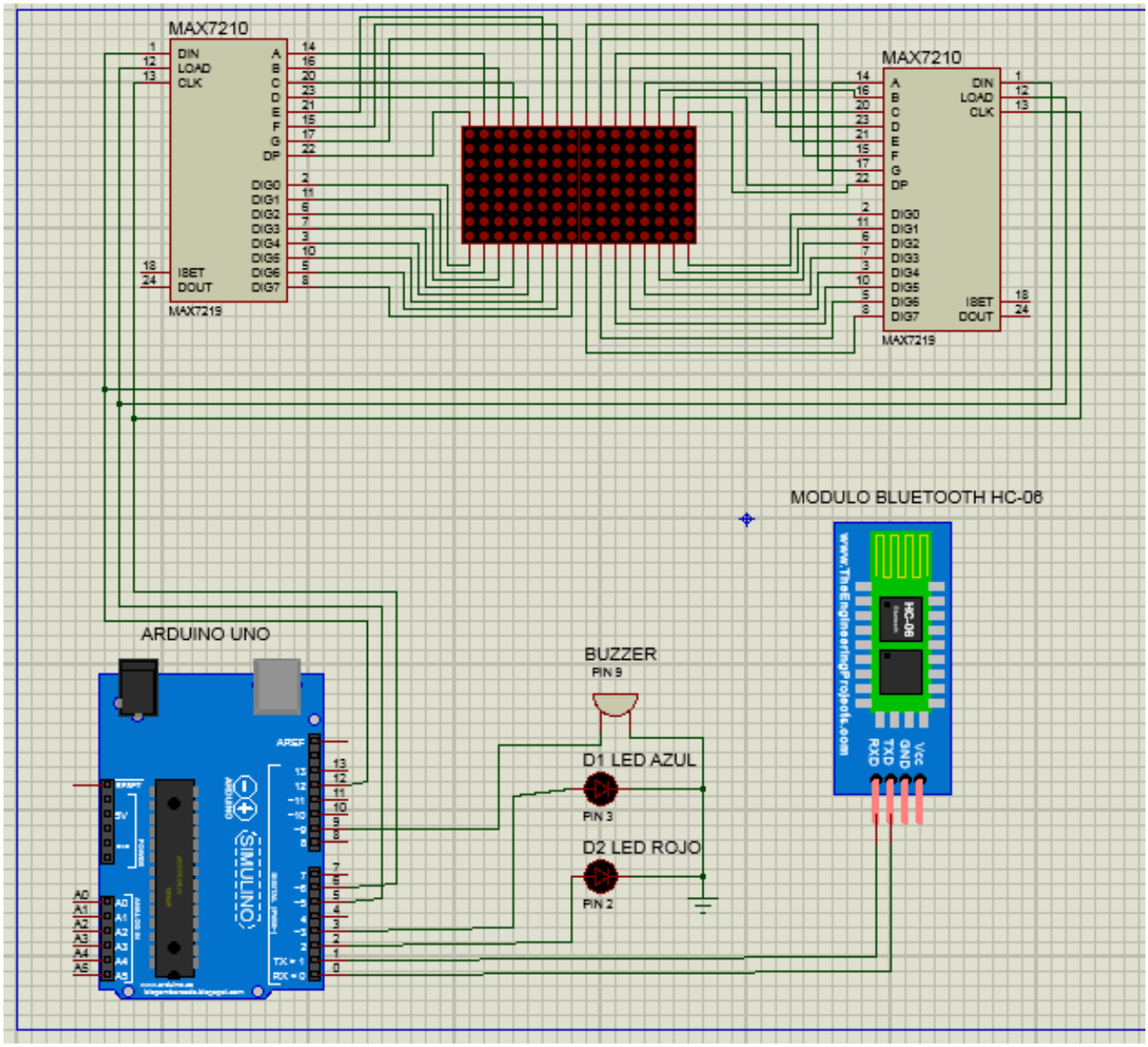
ESQUEMÁTICO DEL HARDWARE

El esquemático del hardware se ha realizado mediante el programa Proteus 8, en dos ficheros separados ya que el robot es controlado por dos Arduinos (ficheros adjuntados en el rar).

ARDUINO MEGA



ARDUINO UNO


MAX7210
DIN
LOAD
CLK
ISET
DOUT
MAX7219
MAX7210
DIN
LOAD
CLK
ISET
DOUT
MAX7219
MODULO BLUETOOTH HC-08
www.TheEngineeringProjects.com
HC-06
Vcc
GND
TXD
RXD
ARDUINO UNO
BUZZER
PIN 9
D1 LED AZUL
PIN 3
D2 LED ROJO
PIN 2

Código ARDUINO MEGA

```
#include <Arduino.h>
#include <IRLibAll.h>     //Libreria para mando IR
#include <Ultrasonic.h>   //Libreria para sensores de distancia
#include <Servo.h>        //Libreria para servomotores

// Definicion Pin interruptor/selector de modo
#define SEL1 11

// Definiciones Botones Mando IR
#define MY_PROTOCOL NEC
#define IZQUIERDA 0xFF22DD
#define DERECHA   0xFFC23D
#define DELANTE   0xFF629D
#define DETRAS    0xFFA857
#define PINZAS    0xFF02FD

// Configuracion Mando IR
IRrecv miReceptor(9); //Pin del receptor IR
IRdecode miDecod;
uint32_t Anterior;  // Variable para cuando se repita por usar protocolo NEC
bool pinzas;        // Variable para cambiar de estado las pinzas
// Definicion Servomotores
Servo motor_FRONTizq;
Servo motor_FRONTder;
Servo motor_BACKizq;
Servo motor_BACKder;
Servo pinza_izq;
Servo pinza_dcha;

// Definicion angulos servomotores pinzas
#define cerrarD 50
#define cerrarI 110
#define abrirD 0
#define abrirI  180
#define cerrarDesq 30
#define cerrarIesq 130

// Definiciones Sensores distancia ultrasonicos
#define ECHOizq      10    // PIN ECHO IZQ
#define TRIGGERizq    22   // PIN TRIG IZQ
#define ECHOdcha      13  // PIN ECHO DCHA
#define TRIGGERdcha    24 // PIN TRIG DCHA
#define ECHOfrente  12    // PIN ECHO FRENTE
#define TRIGGERfrente 23  // PINT TRIG FRENTE
```

```
Ultrasonic SonarFRENTE(TRIGGERfrente,ECHOfrente);  //SONAR FRONTAL
Ultrasonic SonarIZQ(TRIGGERizq,ECHOizq);           //SONAR IZQUIERDA
Ultrasonic SonarDCHA(TRIGGERdcha,ECHOdcha) ;       //SONAR DERECHA

  void setup() {
    motor_FRONTder.attach(7); //Configuracion de pines de los servos (ruedas)
    motor_FRONTizq.attach(8);
    motor_BACKder.attach(5);
    motor_BACKizq.attach(6);
    pinza_dcha.attach(2);     //Configuracion de pines de servos (pinzas)
    pinza_izq.attach(4);
    Serial.begin(9600);
    miReceptor.enableIRIn(); //Inicializacion del receptor IR
    abrirPINZAS();           //El robot siempre comienza con las pinzas abiertas
    pinzas=0;                //Variable de estado de las pinzas
   }//END SETUP


void loop() {

  int selector1 = digitalRead(SEL1);         //Leemos el valor del interruptor para saber en que modo se encuentra
  int sonarFRENTE = SonarFRENTE.Ranging(CM); //Se convierta la distancia a centimetros de los ultrasonicos
  int sonarDCHA = SonarDCHA.Ranging(CM);

  detener();

  if (selector1==HIGH){      //el robot esta con el modo esquivaobstaculos activado
              cerrarPINZASesq();     //Se cierran las pinzas parcialmente para no interferir en el sensor de distancia
              avanceESQ();

                   if(sonarFRENTE>3 && sonarFRENTE<20){   //Obstaculo frontal
                     retrocesoESQ();
                     giroDCHAESQ();}

                  else if(sonarDCHA>3 && sonarDCHA<25)   //Obstaculo derecha
                      giroIZQESQ();

                  else if(sonarIZQ>3 && sonarIZQ<25)     //Obstaculo izquierda
                      giroDCHAESQ();
   }//END IF

  else if (selector1==LOW){      //Modo control remoto mediante mando IR
     detener();                  //Detiene el robot
     if (miReceptor.getResults()) {
           miDecod.decode();          //Decodifica la direccion del boton en hexadecimal
              if(miDecod.protocolNum==MY_PROTOCOL) { //Comprobacion protocolo NEC
                   if(miDecod.value==0xFFFFFFFF)     //Sustitucion de Repeticion por valor correcto
                     miDecod.value=Anterior;
```

```
            switch(miDecod.value) {       //Dependiendo del boton pulsado, realiza su funcion

              case IZQUIERDA:   giroIZQ();      break;

              case DERECHA:     giroDCHA();     break;

              case DELANTE:     avance();       break;

              case DETRAS:      retroceso();    break;

              case PINZAS:

                 if(pinzas==1){
                  abrirPINZAS();
                  pinzas=!pinzas;}

                 else if (pinzas==0){
                  cerrarPINZAS();
                  pinzas=!pinzas;}

              break;
            } //END SWITCH

        Anterior=miDecod.value;}   //Almacena el valor anterior

    miReceptor.enableIRIn();}     //Vuelve a activar el receptor IR
     }
}


//Funciones
void abrirPINZAS(){  //Abre anmbas pinzas(servos)
  pinza_izq.write(abrirI);
  pinza_dcha.write(abrirD);
  delay(100);}

void cerrarPINZAS(){  //Cierra ambas pinzas(servos)
  pinza_izq.write(cerrarI);
  pinza_dcha.write(cerrarD);
  delay(100);}

void avanceESQ(){   // Movimiento continuo hacia delante en modo esquiva-obstaculos
  motor_FRONTizq.write(160);
   motor_FRONTder.write(70);
   motor_BACKizq.write(160);
   motor_BACKder.write(70);
   delay(50);}
```

```
void giroDCHAESQ(){   //GIRO derecha en modo esquiva-obstaculos
   motor_FRONTizq.write(180);
   motor_FRONTder.write(180);
   motor_BACKizq.write(180);
   motor_BACKder.write(180);
   delay(600);}

void giroIZQESQ(){   //GIRO izquierda en modo esquiva-obstaculos
   motor_FRONTizq.write(0);
   motor_FRONTder.write(0);
   motor_BACKizq.write(0);
   motor_BACKder.write(0);
   delay(600);}

void retrocesoESQ(){  //Movimiento continuo hacia detras
   motor_FRONTizq.write(70);
   motor_FRONTder.write(160);
   motor_BACKizq.write(70);
   motor_BACKder.write(160);
   delay(500);}

void avance(){  //Movimiento continuo hacia delante
   motor_FRONTizq.write(180);
   motor_FRONTder.write(0);
   motor_BACKizq.write(180);
   motor_BACKder.write(0);

void giroDCHA(){   //GIRO derecha
  motor_FRONTizq.write(180);
  motor_FRONTder.write(180);
  motor_BACKizq.write(180);
  motor_BACKder.write(180);
  delay(100);
  detener();}

void giroIZQ(){   //GIRO izquierda
  motor_FRONTizq.write(0);
  motor_FRONTder.write(0);
  motor_BACKizq.write(0);
  motor_BACKder.write(0);
  delay(100);
  detener();
  }
```

```
void detener(){    //Detiene el robot
  motor_FRONTizq.write(90);
   motor_FRONTder.write(90);
   motor_BACKizq.write(90);
   motor_BACKder.write(90);}

 void retroceso(){
   motor_FRONTizq.write(0);  //Movimiento continuo hacia detras
   motor_FRONTder.write(180);
   motor_BACKizq.write(0);
   motor_BACKder.write(180);
   delay(200);
   detener();}

void cerrarPINZASesq(){   //Cierra las pinzas parcialmente para no afectar al sensor de distancia
  pinza_izq.write(cerrarIesq);
  pinza_dcha.write(cerrarDesq);}
```

Código ARDUINO UNO

```
#include <LedControlMS.h> //Control encendido matrices LEDs
#include <MaxMatrix.h>       //Control matrices LEDs para transicion
#include <avr/pgmspace.h>

//Formas para Matriz LED
byte OjosON[] = {B00111100,
B01100110,
B11000011,
B10011001,
B10011001,
B11000011,
B01100110,
B00111100};

byte OjosOFF[] = {B00000000,
B00000000,
B00111100,
B01111110,
B11000011,
B10000001,
B00000000,
B00000000};
```

```
byte HI[] = {B00000000,
B00000100,
B00000000,
B01010100,
B01110100,
B01010100,
B00000000,
B00000000};

//Cargar en memoria letras codificadas en binario para texto en scrolling
PROGMEM const unsigned char CH[] = {
3, 8, B00000000, B00000000, B00000000, B00000000, B00000000, // space
1, 8, B01011111, B00000000, B00000000, B00000000, B00000000, // !
3, 8, B00000011, B00000000, B00000011, B00000000, B00000000, // "
5, 8, B00010100, B00111110, B00010100, B00111110, B00010100, // #
4, 8, B00100100, B01101010, B00101011, B00010010, B00000000, // $
5, 8, B01100011, B00010011, B00001000, B01100100, B01100011, // %
5, 8, B00110110, B01001001, B01010110, B00100000, B01010000, // &
1, 8, B00000011, B00000000, B00000000, B00000000, B00000000, // '
3, 8, B00011100, B00100010, B01000001, B00000000, B00000000, // (
3, 8, B01000001, B00100010, B00011100, B00000000, B00000000, // )
5, 8, B00101000, B00011000, B00001110, B00011000, B00101000, // *
5, 8, B00001000, B00001000, B00111110, B00001000, B00001000, // +
2, 8, B10110000, B01110000, B00000000, B00000000, B00000000, // ,
4, 8, B00001000, B00001000, B00001000, B00001000, B00000000, // -
2, 8, B01100000, B01100000, B00000000, B00000000, B00000000, // .
4, 8, B01100000, B00011000, B00000110, B00000001, B00000000, // /
4, 8, B00111110, B01000001, B01000001, B00111110, B00000000, // 0
3, 8, B01000010, B01111111, B01000000, B00000000, B00000000, // 1
4, 8, B01100010, B01010001, B01001001, B01000110, B00000000, // 2
4, 8, B00100010, B01000001, B01001001, B00110110, B00000000, // 3
4, 8, B00011000, B00010100, B00010010, B01111111, B00000000, // 4
4, 8, B00100111, B01000101, B01000101, B00111001, B00000000, // 5
4, 8, B00111110, B01001001, B01001001, B00110000, B00000000, // 6
4, 8, B01100001, B00010001, B00001001, B00000111, B00000000, // 7
4, 8, B00110110, B01001001, B01001001, B00110110, B00000000, // 8
4, 8, B00000110, B01001001, B01001001, B00111110, B00000000, // 9
2, 8, B01010000, B00000000, B00000000, B00000000, B00000000, // :
2, 8, B10000000, B01010000, B00000000, B00000000, B00000000, // ;
3, 8, B00010000, B00101000, B01000100, B00000000, B00000000, // <
3, 8, B00010100, B00010100, B00010100, B00000000, B00000000, // =
3, 8, B01000100, B00101000, B00010000, B00000000, B00000000, // >
4, 8, B00000010, B01011001, B00001001, B00000110, B00000000, // ?
5, 8, B00111110, B01001001, B01010101, B01011101, B00001110, // @
4, 8, B01111110, B00010001, B00010001, B01111110, B00000000, // A
4, 8, B01111111, B01001001, B01001001, B00110110, B00000000, // B
4, 8, B00111110, B01000001, B01000001, B00100010, B00000000, // C
4, 8, B01111111, B01000001, B01000001, B00111110, B00000000, // D
```

```
4, 8, B01111111, B01001001, B01001001, B01000001, B00000000, // E
4, 8, B01111111, B00001001, B00001001, B00000001, B00000000, // F
4, 8, B00111110, B01000001, B01001001, B01111010, B00000000, // G
4, 8, B01111111, B00001000, B00001000, B01111111, B00000000, // H
3, 8, B01000001, B01111111, B01000001, B00000000, B00000000, // I
4, 8, B00110000, B01000000, B01000001, B00111111, B00000000, // J
4, 8, B01111111, B00001000, B00010100, B01100011, B00000000, // K
4, 8, B01111111, B01000000, B01000000, B01000000, B00000000, // L
5, 8, B01111111, B00000010, B00001100, B00000010, B01111111, // M
5, 8, B01111111, B00000100, B00001000, B00010000, B01111111, // N
4, 8, B00111110, B01000001, B01000001, B00111110, B00000000, // O
4, 8, B01111111, B00001001, B00001001, B00000110, B00000000, // P
4, 8, B00111110, B01000001, B01000001, B10111110, B00000000, // Q
4, 8, B01111111, B00001001, B00001001, B01110110, B00000000, // R
4, 8, B01000110, B01001001, B01001001, B00110010, B00000000, // S
5, 8, B00000001, B00000001, B01111111, B00000001, B00000001, // T
4, 8, B00111111, B01000000, B01000000, B00111111, B00000000, // U
5, 8, B00001111, B00110000, B01000000, B00110000, B00001111, // V
5, 8, B00111111, B01000000, B00111000, B01000000, B00111111, // W
5, 8, B01100011, B00010100, B00001000, B00010100, B01100011, // X
5, 8, B00000111, B00001000, B01110000, B00001000, B00000111, // Y
4, 8, B01100001, B01010001, B01001001, B01000111, B00000000, // Z
2, 8, B01111111, B01000001, B00000000, B00000000, B00000000, // [
4, 8, B00000001, B00000110, B00011000, B01100000, B00000000, // \ backslash
2, 8, B01000001, B01111111, B00000000, B00000000, B00000000, // ]
3, 8, B00000010, B00000001, B00000010, B00000000, B00000000, // hat
4, 8, B01000000, B01000000, B01000000, B01000000, B00000000, // _
2, 8, B00000001, B00000010, B00000000, B00000000, B00000000, // `
4, 8, B00100000, B01010100, B01010100, B01111000, B00000000, // a
4, 8, B01111111, B01000100, B01000100, B00111000, B00000000, // b
4, 8, B00111000, B01000100, B01000100, B00101000, B00000000, // c
4, 8, B00111000, B01000100, B01000100, B01111111, B00000000, // d
4, 8, B00111000, B01010100, B01010100, B00011000, B00000000, // e
3, 8, B00000100, B01111110, B00000101, B00000000, B00000000, // f
4, 8, B10011000, B10100100, B10100100, B01111000, B00000000, // g
4, 8, B01111111, B00000100, B00000100, B01111000, B00000000, // h
3, 8, B01000100, B01111101, B01000000, B00000000, B00000000, // i
4, 8, B01000000, B10000000, B10000100, B01111101, B00000000, // j
4, 8, B01111111, B00010000, B00101000, B01000100, B00000000, // k
3, 8, B01000001, B01111111, B01000000, B00000000, B00000000, // l
5, 8, B01111100, B00000100, B01111100, B00000100, B01111000, // m
4, 8, B01111100, B00000100, B00000100, B01111000, B00000000, // n
4, 8, B00111000, B01000100, B01000100, B00111000, B00000000, // o
4, 8, B11111100, B00100100, B00100100, B00011000, B00000000, // p
4, 8, B00011000, B00100100, B00100100, B11111100, B00000000, // q
4, 8, B01111100, B00001000, B00000100, B00000100, B00000000, // r
4, 8, B01001000, B01010100, B01010100, B00100100, B00000000, // s
3, 8, B00000100, B00111111, B01000100, B00000000, B00000000, // t
4, 8, B00111100, B01000000, B01000000, B01111100, B00000000, // u
5, 8, B00011100, B00100000, B01000000, B00100000, B00011100, // v
5, 8, B00111100, B01000000, B00111100, B01000000, B00111100, // w
5, 8, B01000100, B00101000, B00010000, B00101000, B01000100, // x
4, 8, B10011100, B10100000, B10100000, B01111100, B00000000, // y
3, 8, B01100100, B01010100, B01001100, B00000000, B00000000, // z
3, 8, B00001000, B00110110, B01000001, B00000000, B00000000, // {
1, 8, B01111111, B00000000, B00000000, B00000000, B00000000, // |
3, 8, B01000001, B00110110, B00001000, B00000000, B00000000, // }
4, 8, B00001000, B00000100, B00001000, B00000100, B00000000, // ~
};
```

```
//Pines de LEDs y altavoz
int ledPinR = 2;
int ledPinB = 3;
int speakerOut = 4;

//Matrices LEDs
int data = 12;     // DIN, data in
int load = 5;      // CS, load
int clock = 6;     // CLK, clock
int maxInUse = 2;  //Numero Matrices
MaxMatrix m(data, load, clock, maxInUse); //Definicion pines Matriz LED
LedControl lc1 = LedControl (data, clock, load, maxInUse); //Definicion pines Matriz LED
char frase[] = " Robot de ALBERTO   ";  //Texto a mostrar en matrices LEDs en scrolling
byte buffer[10];

//BT
char rxChar;    // Variable para recibir datos del puerto serie

//Cancion Cucaracha
int timeUpDown[] = {3822, 3606, 3404, 3214, 3032, 2862,
2702, 2550, 2406, 2272, 2144, 2024,
1911, 1803, 1702, 1607, 1516, 1431,
1351, 1275, 1203, 1136, 1072, 1012};
byte song[] = {7,7,7,12,12,12,16,16,7,7,7,12,12,12,16,16,16,16,16,16,16,12,12,11,11,9,9,7,7,7,7,7};
byte beat = 0;
int MAXCOUNT = 32;
float TEMPO_SECONDS = 0.2;
byte statePin = LOW;
byte period = 0;
int i, timeUp;
unsigned long delayTime=400;

void setup(){
  m.init(); // Inicializacion modulo MAX7219
  m.setIntensity(10); // Intensidad LED's 0-15
  pinMode(ledPinR, OUTPUT); //Definicion pines salida
  pinMode(ledPinB, OUTPUT);
  pinMode(speakerOut, OUTPUT);
  Serial.begin(9600);
  lc1.shutdown(0,false);    //Iniciamos la matriz led #1
  lc1.shutdown(1,false);  //Iniciamos la matriz led #2
  lc1.setIntensity(0,10);    //Intensidad de los led en la matriz #1
  lc1.setIntensity(1,10);  //Intensidad de los led en la matriz #2
  lc1.clearDisplay(0);      //Apagamos todos los led de la matriz #1
  lc1.clearDisplay(1);    //Apagamos todos los led de la matriz #2
}//FIN SETUP
```

```
void loop(){
  int cont=0;   //Variable para bucle Modo Policia

  Ojos1();     //El Robot comienza abriendo los ojos
  delay(500);
  Ojos2();     //Y cerrandolos
  delay(500);

  Serial.println("Presione L --> Letras en scrolling");  //Envia por BT los posibles modos
  Serial.println("Presione P --> Modo Policia");
  Serial.println("Presione B --> Hi");
  Serial.println("Presione C --> Cancion");
  Serial.println("-------------------------------------"); //Separacion

 if( Serial.available() ){      // Si hay datos disponibles en el buffer
      rxChar = Serial.read();   // Leer un byte y colocarlo en variable

    if( rxChar == 'L'){         //Modo texto en scrolling
        m.shiftLeft(false, true);
        printStringWithShift(frase, 100);}

    else if( rxChar == 'P'){   //Modo Policia
        while(cont<20){
             digitalWrite(ledPinR,HIGH);
             tone(speakerOut, 10, 5000);
             delay(200);
            digitalWrite(ledPinR,LOW);
            digitalWrite(ledPinB,HIGH);
            tone(speakerOut, 1000, 5000);
            delay(200);
            digitalWrite(ledPinB,LOW);
            cont++;}
       noTone(speakerOut);}

    else if ( rxChar == 'C'){    //Modo Cancion
       digitalWrite(speakerOut, LOW);
           for (beat = 0; beat < MAXCOUNT; beat++) { //Comienza a reproducir la cancion
             statePin = !statePin;
             timeUp = timeUpDown[song[beat]];
             period = ((1000000 / timeUp) / 2) * TEMPO_SECONDS;
                         for (i = 0; i < period; i++) {
                               digitalWrite(speakerOut, HIGH);
                               delayMicroseconds(timeUp);
                               digitalWrite(speakerOut, LOW);
                               delayMicroseconds(timeUp);}
                          }
       digitalWrite(speakerOut, LOW);}

       else if( rxChar == 'B'){  //Representa 'hi' en la matriz LED
         Representar(HI,3000);}

     }
   delay(2000);
 } //FIN LOOP
```

```
// Representar caracter en la matriz
void printCharWithShift(char c, int shift_speed){
  if (c < 32) return;
  c -= 32;
  memcpy_P(buffer, CH + 7*c, 7);
  m.writeSprite(maxInUse*8, 0, buffer);
  m.setColumn(maxInUse*8 + buffer[0], 0);

        for (int i=0; i<buffer[0]+1; i++)
            {
              delay(shift_speed);
            m.shiftLeft(false, false);
             }
}

// Extraer caracter del texto en movimiento
void printStringWithShift(char* s, int shift_speed){
  while (*s != 0){
    printCharWithShift(*s, shift_speed);
    s++;
  }
}

// Funcion para representar figuras en matriz led 8x8
void Representar(byte *Datos,int retardo)
{
  for (int i = 0; i < 8; i++)
  {
    lc1.setColumn(0,i,Datos[7-i]);
  }
  delay(retardo);
}

//Funcion creada Ojos abiertos en matriz LED
void Ojos1(){
  for (int i = 0; i < 8; i++){
    lc1.setColumn(0,i,OjosON[i]);
    lc1.setColumn(1,i,OjosON[i]);}}

//Funcion creada Ojos cerrados en matriz LED
void Ojos2(){
  for (int i = 0; i < 8; i++){
    lc1.setColumn(0,i,OjosOFF[i]);
    lc1.setColumn(1,i,OjosOFF[i]);}}
```

BIBLIOGRAFÍA O ENLACES EMPLEADOS

Generador online código matriz 8x8 LED HEX/BINARY para ARDUINO: https://www.riyas.org/2013/12/online-led-matrix-font-generator-with.html

Librería para Arduino del módulo Ultrasonic Ranging HC-SR04: http://www.ardublog.com/library-for-arduino-ultrasonic-ranging-hc-sr04/

Servomotor Arduino tutorial de programación: https://programarfacil.com/tutoriales/fragmentos/servomotor-con-arduino/

Controlar un servo de rotación continua con Arduino: https://www.luisllamas.es/controlar-un-servo-de-rotacion-continua-con-arduino/

Módulo BlueTooth HC-06 Algunos conceptos básicos de BlueTooth: https://www.prometec.net/bt-hc06/

Matrices LED de 8×8 con Arduino y MAX7219: https://www.minitronica.com/matrices-led-8x8-arduino-max7219/

Scrolling con MAX7219. Desplazamiento lateral de mensajes en el display: https://www.prometec.net/scroll-max7219/

Reproducir sonidos con Arduino y un buzzer pasivo o altavoz: https://www.luisllamas.es/reproducir-sonidos-arduino-buzzer-pasivo-altavoz/

Prácticas de Código Abierto. La Cucaracha con el Arduino: http://codigoabierto.geografias.org/?p=113