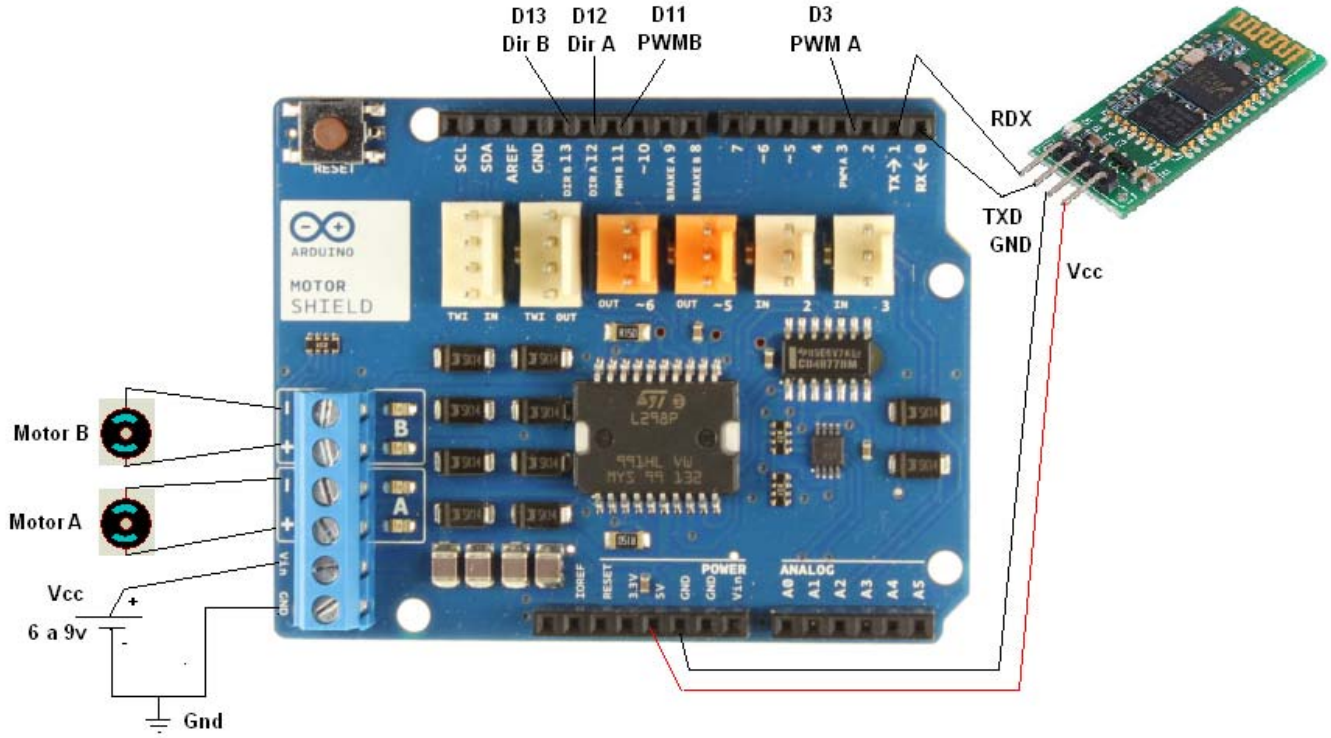
Arduino Motor Shield y Bluetooh

Placa de control para dos motores de cc. con arduino.



Pines de control			
		Motor A	Motor B
On/off velocidad		D3	D11
Dirección		D12	D13
no utilizados	Freno	D9	D8
	Corriente	A0	A1

Funcionamiento		
	On/off velocidad	Dirección
Parar	LOW	LOW
Giro horario	HIGH	LOW
Giro antihorario	HIGH	HIGH

Arduino Motor Shield.

El Arduino Motor Shield se basa en el c.i. L298 (ficha técnica), que es un doble puente completo controlador diseñado para manejar cargas inductivas tales como relés, solenoides, DC y motores paso a paso. Le permite manejar dos motores de corriente continua con la placa Arduino, controlar la velocidad y dirección de cada uno de forma independiente. También se puede medir la absorción de corriente del motor de cada motor, entre otras características. El conector es TinkerKit compatibles.

Resumen

Tensión de funcionamiento	5V a 12V
Controlador de motor	L298P , unidades de 2 motores de corriente continua o un motor paso a paso
Corriente máxima	2A por canal o máx 4A (con fuente de alimentación externa)
Detección de corriente	1.65V / A

Parada de funcionamiento libre y función de freno	

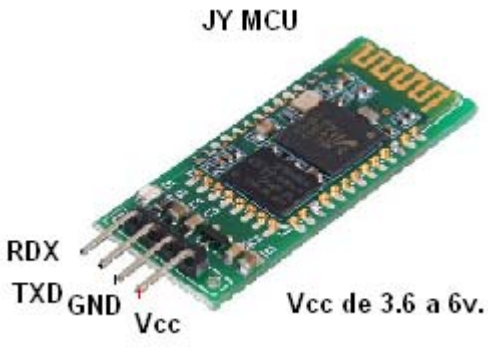
Esquema de diseño.

EAGLE archivos: arduino_MotorShield_Rev3-reference-design.zip
Esquema: arduino_MotorShield_Rev3-schematic.pdf

Módulo Bluetooth:

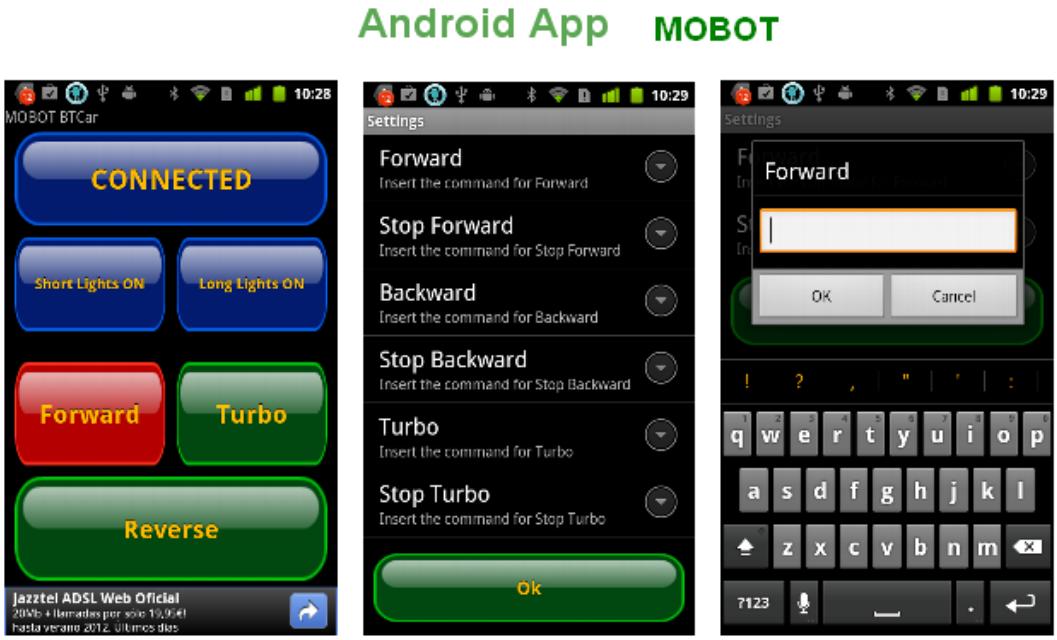
El módulo JY-MCU podemos utilizarlo para comunicarse con arduino mediante bluetooth.

Conectar a la placa arduino RDX a D1(TX), TXD a D0(RX), GND y Vcc a los pines correspondientes. La información se recibe vía serie.



Para utilizar el teléfono móvil como elemento de control, vincular el módulo como **livor** (contraseña **1234**).

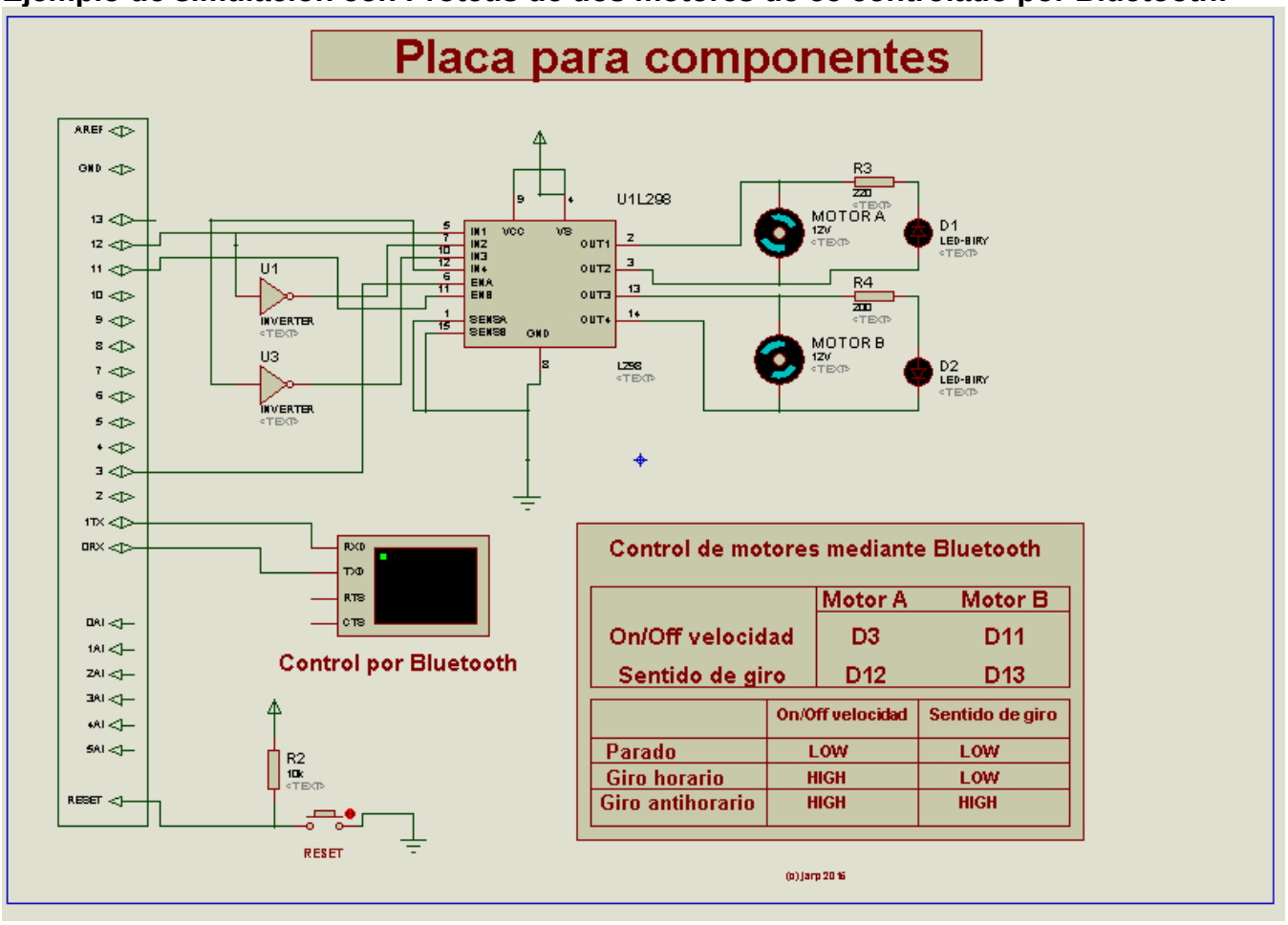
Hay aplicaciones en android para descargar como Motbot (http://www.mobot.es/) o realizarla nosotros con ayuda de APP Inventor (http://ai2.appinventor.mit.edu/).



También podemos configurar el módulo mediante comandos AT (http://www.extremadura-web.es/Blog/2012/10/29/comunicacion-bluetooth-serie-arduino-y-basic4android/)

Para simular el módulo en **proteus** utilizamos el **VIRTUAL TERMINAL** (configurar velocidad del simulador a 1200). Simulamos la información recibida por bluetooth con ayuda del teclado.

Ejemplo de simulación con Proteus de dos motores de cc controlado por Bluetooth:



Programa arduino:

```
//Control motores con Arduino MotorShieds mediante bluetooth
int motor_A = 3;                  //control on/off velocidad motor A
int motor_A_giro = 12;            //dirección giro motor A
int motor_B = 11;                 //control on/off velocidad motor B
int motor_B_giro = 13;            //dirección giro motor B
char valor ;                      //almacena carácter recibido por bluetooth
void setup () {
 pinMode (motor_A,OUTPUT);
 pinMode (motor_A_giro,OUTPUT);
 pinMode (motor_B,OUTPUT);
 pinMode (motor_B_giro,OUTPUT);
 Serial.begin(9600);              //configura terminal para simular control por bluetooh
                                  //IMPORTANTE  Poner en proteus velocidad simulador a 1200
 Serial.print("Pulsar tecla 8:avanza, 2:retrocede, 4: izquierda, 6:derecha, 5:stop");
 Serial.println();                //línea siguiente
 } // setup

// define acciones
void avanza(){
 digitalWrite(motor_A, HIGH);
 digitalWrite(motor_A_giro, HIGH);
 digitalWrite(motor_B, HIGH);
 digitalWrite(motor_B_giro, HIGH);
}
```

```
void retrocede(){
 digitalWrite(motor_A, HIGH);
 digitalWrite(motor_A_giro, LOW);
 digitalWrite(motor_B, HIGH);
 digitalWrite(motor_B_giro, LOW);
}
void gira_izquierda(){
 digitalWrite(motor_A, HIGH);
 digitalWrite(motor_A_giro, LOW);
 digitalWrite(motor_B, HIGH);
 digitalWrite(motor_B_giro, HIGH);
}
void gira_derecha(){
 digitalWrite(motor_A, HIGH);
 digitalWrite(motor_A_giro, HIGH);
 digitalWrite(motor_B, HIGH);
 digitalWrite(motor_B_giro, LOW);
}
void parar(){
 digitalWrite(motor_A, LOW);
 digitalWrite(motor_A_giro, LOW);
 digitalWrite(motor_B, LOW);
 digitalWrite(motor_B_giro, LOW);
}

void loop(){
 if(Serial.available()>0){
   valor=Serial.read();
    Serial.println(valor);
  switch (valor) {
 case '8':    // Avanza
   Serial.println("Avanza");
    avanza();
   break;
 case '2':    // Retrocede
   Serial.println("Retrocede");
    retrocede();
   break;
 case '4':    // Gira izquierda
   Serial.println("Gira izquierda");
   gira_izquierda();
   break;
 case '6':    // Gira derecha
   Serial.println("Gira derecha");
   gira_derecha();
   break;
 case '5':    Parar
   Serial.println("Parar");
   parar();
   break;
 }  // switch
     }//serial
  delay(100);                                  //retardo para leer tecla
} //loop
```