

Instituto Politécnico Nacional.

Escuela Superior de Cómputo.

Alumnos:

Rojas Montaño Marcell Douglas

Mendoza Parra Sergio

Sanchez Merino Maximiliano

Torrijos Luna Irwing Omar

Materia:

Application Development for Mobile Devices

Profesor:

Cifuentes Álvarez Alejandro Sigfrido

**Reporte Proyecto 2**

Grupo:

3CM3

Fecha de entrega: 23/11/2018

Contenido

[**Objetivo** 1](#_Toc525578681)

[**Conceptos** 1](#_Toc525578682)

[**Desarrollo** 3](#_Toc525578683)

[**Código** 4](#_Toc525578684)

[**Conclusión** 4](#_Toc525578684)

[**Referencias** 4](#_Toc525578685)

# **Objetivo**

Desarrollar una aplicación móvil con ANDROID STUDIO que utilice la comunicación Bluettooth para mostrar en una pantalla LCD la ubicación seleccionada en el dispositivo móvil por medio de la app. La medición de estas variables son enviadas a través del módulo Bluetooth de un Arduino Uno.

# **Conceptos**

Bluetooth es una especificación industrial para Redes Inalámbricas de Área Personal (WPAN) que posibilita la transmisión de voz y datos entre diferentes dispositivos mediante un enlace por radiofrecuencia en la banda ISM de los 2.4GHz. Los principales objetivos que se pretenden conseguir con esta norma son:

* Facilitar las comunicaciones entre equipos móviles.
* Eliminar los cables y conectores entre estos.
* Ofrecer la posibilidad de crear pequeñas redes inalámbricas y facilitar la sincronización de datos entre equipos personales.

Los dispositivos BlueTooth pueden actuar como Masters o como Slaves. La diferencia es que un BlueTooth Slave solo puede conectarse a un master y a nadie más, en cambio un master BlueTooth, puede conectarse a varios Slaves o permitir que ellos se conecten y recibir y solicitar información de todos ellos, arbitrando las transferencias de información (Hasta un máximo de 7 Slaves) [1]. Para dotar de comunicación bluetooth con Arduino podemos hacerlo de varias formas:

* Modulo bluetooth externo. [2]
* Módulo HC-05 o HC-06.
* Módulo Bluetooth 4.0 HC-08 y HC-09.

Como el módulo BlueTooth es básicamente un nodo BT conectado a una interface serie, podríamos en principio conectar los pines RX y Tx a los equivalentes de Arduino en los pines 0 y 1 digitales, sin más que cruzarlos (BT Tx a Arduino Rx y BT Rx a Aduano Tx) y de hecho muchos ejemplos en Internet utilizan este esquema y se comunican con el BT mediante las familiares instrucciones de Serial.print (). Sin embargo, puesto que los pines 0 y 1 se utilizan en la comunicación serie de Arduino con el PC a través del USB y por tanto, si los usamos para comunicar con el módulo BT, perderíamos la conexión con el PC, es mejor usar otros pines.

Fijarse que, al hacer esta conexión, el LED del módulo HC-05, parpadea continuamente. Esto indica que no está pareado o vinculado. Cuando conectes algo al módulo, esta luz se quedará fija y es la forma de saber si hay conexión o no.

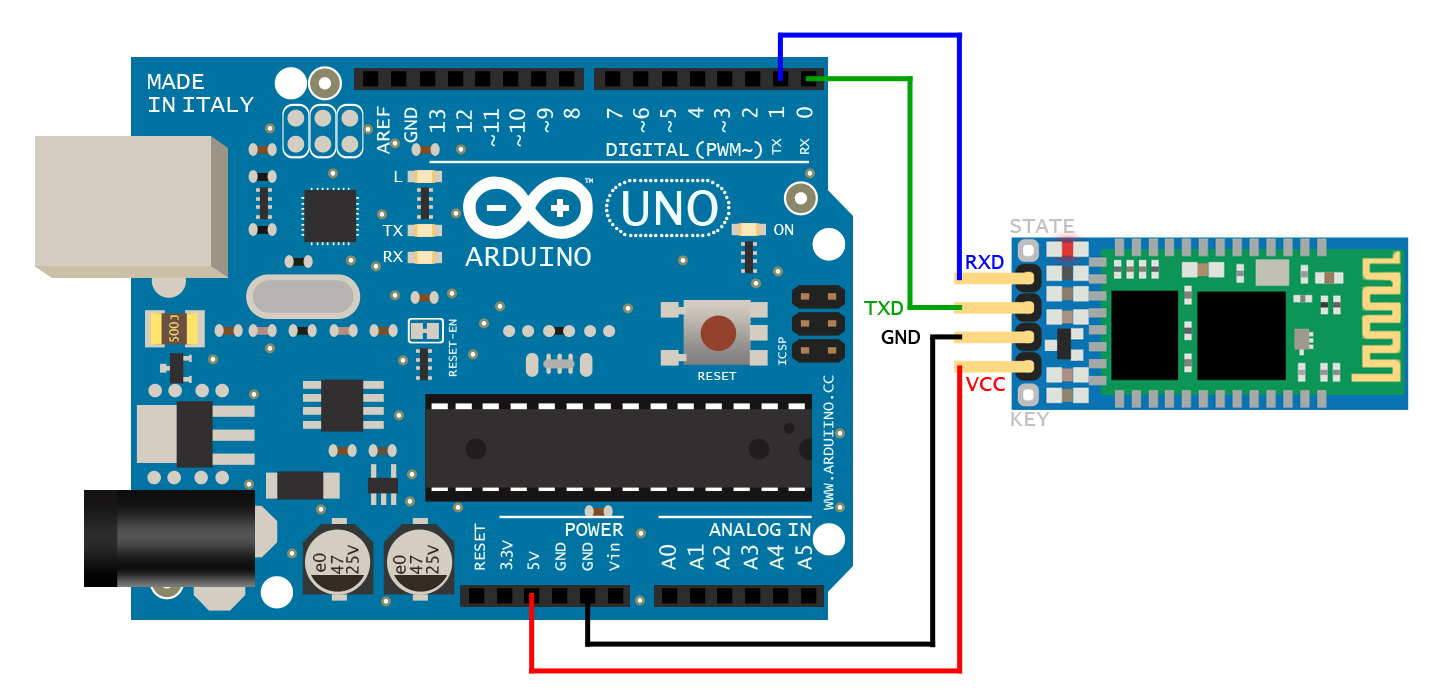


Imagen 1 Conexión del módulo Bluetooth al Arduino

El convertidor analógico-digital ADC del Arduino es de 10 bits; es decir, distingue 1024 valores diferentes. El nivel de voltaje en una entrada ADC es de 5 volts, por tanto: 5/1024 = 0.0049 volts = 4.9 mV

Por lo que un incremento en 1 significa 4.9 mV. Por ejemplo, si se conecta un potenciómetro a la entrada analógica A0, un 0 representa 0V y 1023 representa 5V. Se necesita un retardo para realizar la conversión de una señal analógica en digital, por ejemplo, de 10 milisegundos entre lecturas.

La función Serial.println(valor, DEC) envía datos decimales, pero también en hexadecimal con HEX, binarios con BIN y octales con OCT. La función analogRead(pin) lee el pin 0, 1, 2, 3, 4 o 5 como entradas ADC predeterminadas de 10 bits.

# **Desarrollo**



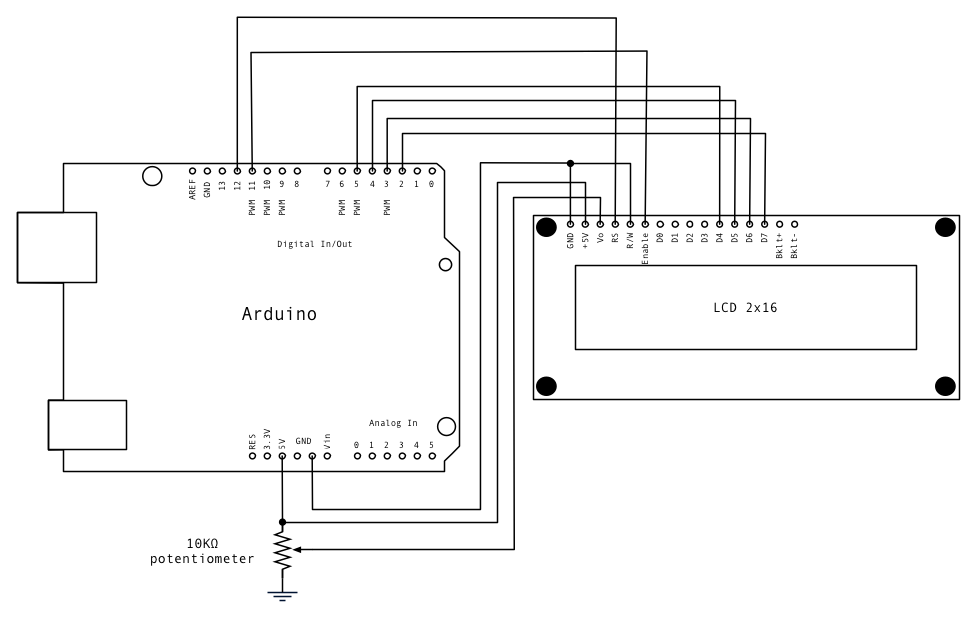


Imagen 1 Conexión del Arduino a la pantalla LCD.

# **Código**

#include <SoftwareSerial.h>

SoftwareSerial ModBluetooth(2, 3); // RX | TX

void setup()

{

pinMode(13, OUTPUT);

digitalWrite(13, LOW);

ModBluetooth.begin(9600);

Serial.begin(9600);

ModBluetooth.println("MODULO CONECTADO");

ModBluetooth.print("#");

}

void loop()

{

if (ModBluetooth.available())

{

char VarChar;

VarChar = ModBluetooth.read();

if(VarChar == '1')

{

digitalWrite(13, HIGH);

delay(100);

ModBluetooth.print("LED ENCENDIDO");

Serial.print("LED ENCENDIDO");

ModBluetooth.print("#");

}

if(VarChar == '0')

{

digitalWrite(13, LOW);

delay(100);

ModBluetooth.print("LED APAGADO#");

Serial.print("LED APAGADO#");

}

}

}

# **Conclusión**

Android es una herramienta muy importante para la creación de app´s, en este proyecto se ve reflejado el resultado indicando la posición en la que se encuentra nuestro dispositivo en el mapa por medio de la pantalla LCD.

Una pantalla LCD son dispositivos diseñados para mostrar información en forma gráfica. LCD significa Liquid Crystal Display (Display de cristal líquido). La mayoría de las pantallas LCD vienen unidas a una placa de circuito y poseen pines de entrada/salida de datos. Como se podrán imaginar, Arduino es capaz de utilizar las pantallas LCD para desplegar datos LCD.

# **Referencias**

[1]"Bluetooth en Arduino", *Aprendiendo Arduino*, 2018. [En linea]. Disponible en: https://aprendiendoarduino.wordpress.com/2016/11/13/bluetooth-en-arduino/. [Accedido el: 24- Sep- 2018].

[2]A. Industries, "Bluefruit EZ-Key - 12 Input Bluetooth HID Keyboard Controller", *Adafruit.com*, 2018. [En linea]. Disponible en: https://www.adafruit.com/products/1535. [Accedido el: 24- Sep- 2018].