

Universidad de Costa Rica
Facultad de Ingeniería
Escuela de Ingeniería Eléctrica
IE-0624 Laboratorio de Microcontroladores
II ciclo 2022

Propuesta de proyecto

Sistema de jardinería automatizada con arduino

Estudiante:
Meybelle Castro Valverde - B91887
Sinaí Zamora Vega - B88719
Grupo 01

Profesor: MSc. Marco Villalta Fallas

20 de septiembre, 2022

1. Introducción

En este proyecto se llevará a cabo la implementación y el desarrollo del código de una jardinería automatizada con Arduino. Esta jardinería sera capaz de activar un sistema de riego e iluminación automático al detectar una humedad, temperatura o iluminación por debajo de la ideal para las plantas del jardín. Para esto, el sistema utiliza un sensor de temperatura y humedad DHT11, un sensor de humedad del suelo para medir las variables físicas y decidir si se debe regar o no el jardín, para el riego, se utiliza un relé que alimenta a una bomba de agua sumergible, y para la iluminación, se utilizo un led que se activará de acuerdo a las necesidades de iluminación de las plantas. Asimismo, se hizo una implementación de una aplicación móvil que recibe los datos vía bluetooth.

El repositorio con los archivos del proyecto se encuentra en la dirección:
<https://github.com/SinaiZamora/Proyecto—Laboratorio-de-Microcontroladores>

2. Justificación

Los jardines y huertas son un componente esencial en la supervivencia de los seres humanos, ya que el oxígeno y gran parte de nuestra alimentación proviene de estas, sin embargo, el hacerse cargo de un jardín o una huerta puede convertirse en un trabajo arduo para aquellas personas que deben dedicar la mayor parte de su día a otras actividades laborales, académicas o familiares, por lo que una solución a esto es una huerta o jardín automático, que se riegue en cantidades y en el tiempo adecuado midiendo las condiciones físicas actuales en las que se encuentra la tierra y el ambiente del jardín.

3. Objetivos y alcances

- Investigar en internet sobre sistemas de riego automático para plantas.
- Determinar los componentes eléctricos necesarios para el sistema.
- Diseñar e implementar el software necesario del Arduino.
- Comprobar el correcto funcionamiento del sistema mediante diferentes pruebas.

4. Metodología

Para la implementación del hardware, se utilizaron como base los componentes: Arduino UNO, sensor de humedad y temperatura DHT11, sensor de humedad del suelo, bomba de agua, relé, sensor de nivel, LED's y módulo Bluetooth. Se realizó primero un esquemático en el simulador Fritzing, y se implementó el código en lenguaje C++ bajo el framework de Arduino. Para el código, se hizo uso de la librería DHT.h, la cual corresponde a una librería del sensor de humedad y temperatura que se utilizó, cuando el código y la esquemático estuvieron listos, se procederá a armar el circuito electrónico de manera física y realizar las pruebas respectivas.

5. Marco teórico

5.1. Arduino UNO

El microcontrolador Arduino Uno es un MCU que se encuentra basado en el ATmega328P, este funciona como una interfaz para este microcontrolador. El uso del Arduino facilita la comunicación con la computadora por medio del puerto USB. En la siguiente imagen se muestra los 14 pines de I/O digitales. [1]

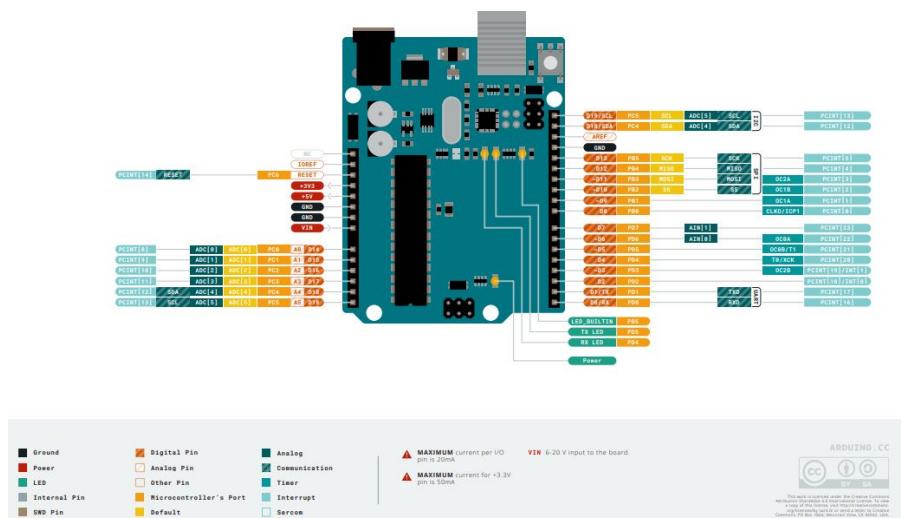


Figura 1: Diagrama del Arduino UNO [1]

Las características eléctricas del Arduino UNO son: [1]

- Voltaje de operación: 5 V
- Velocidad de operación máxima: 20 MHz
- Rango de operación de temperatura: -10 °C a 105 °C
- Velocidad de reloj: 16 MHz
- Corriente máxima por pin: 40 mA

Las funciones más importantes de programación del Arduino UNO son:

- **analogRead():** este tiene un único parámetro que es un pin analógico, entonces devuelve el valor leído de este. El valor puede ir entre 0 y 1023, que corresponde a un mapeo de 0 V a 5 V en 10 bits, por medio del convertidor analógico-digital del microcontrolador.
- **pinMode():** este recibe dos parámetros, un pin y un indicador de I/O, en donde configura al pin como entrada (INPUT) o como salida (OUTPUT).
- **digitalWrite():** este recibe dos parámetros: un pin y un indicador de H/L, en donde configura al pin como alto (HIGH) o bajo (LOW).
- **digitalRead():** este tiene un único parámetro que es un pin, entonces devuelve el valor leído que puede ser HIGH o LOW.

5.2. Microcontrolador ATmega328P

El arduino UNO posee un microcontrolador ATmega328/P, el cual posee 8 bits de CMOS y se basa en la arquitectura AVR RISC con 131 operaciones, la mayoría de estas se ejecutan en un solo ciclo de reloj, además cuenta con 32 registros de propósito general. Tiene 32 líneas de I/O programables, 28 pines PDIP y pad QFN/MLF. Cuenta con 2 timers de 8 bits con prescaler y modo de comparación, y otro de 16 bits con los mismos modos. Tiene 6 canales PWM, un comparador analógico, un timer tipo watching con oscilador, USI, USART y una interfaz de 2 cables. [2]

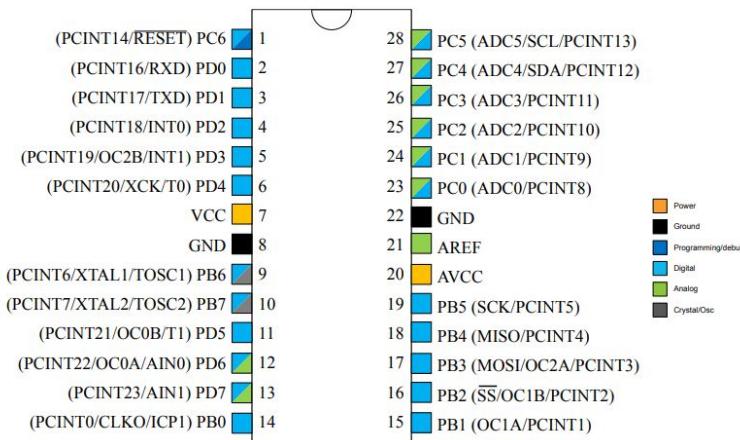


Figura 2: ATmega328P [2]

Funcionalidad de los pines del microcontrolador son: [2]

- **Pin VCC:** tensión de alimentación digital.
- **Pin GND:** tensión de alimentación tierra.
- **Puerto B:** este puerto tiene 8 pines de I/O bidireccionales, con resistencias pull-up internas, los buffers en la salida poseen características de control simétricas con alta capacidad de fuente y drenaje.
- **Puerto C:** este puerto tiene 6 pines de I/O bidireccionales, con resistencias de pull-up internas, los buffers en la salida poseen características de control simétricas de alta capacidad de fuente y drenaje.
- **PC6/RESET:** Si el fusible RSTDISBL está programado, entonces PC6 se usa como pin de I/O, pero, si el fusible RSTDISBL no está programado, el PC6 se usa como entrada de Reset. Y si está en bajo por más tiempo que la duración mínima del pulso, se generará un Reset, incluso si el reloj no está corriendo.
- **Puerto D:** este puerto tiene 8 pines de I/O bidireccional, con resistencias pull-up internas, los buffers en la salida poseen características de control simétricas de alta capacidad de fuente y drenaje.
- **AV_{cc}:** es el pin de alimentación de tensión para el convertidor analógico/digital, compuesto por PC[3:0] y PE[3:2]. Debe estar conectado externamente a VCC, incluso si no se use el convertidor. Y en el caso que se utilice se debe conectar a VCC a través de un filtro de paso bajo.

- **AREF:** este pin es el de referencia analógica para el convertidor analógico-digital.
- **ADC:** este sirve como entradas análogas para el convertidor

5.3. Comunicación serial USART

El microcontrolador ATmega328/P tiene el periférico USART, por sus siglas en inglés (Universal Synchronous Asynchronous Receiver Transceiver), que se usa para la comunicación con datos seriales a la computadora, esto se realiza por medio de la conexión directa de 2 puertos USART, en donde el transmisor logra convertir la información paralela del bus de entrada en una forma serial, y esta se envía al receptor que se encarga de des-serializar estos datos para así poder leerlos.

Los datos seriales del Arduino UNO se manejan con el uso de las siguientes funciones:

- **Serial.begin():** inicia la comunicación serial.
- **Serial.print():** envía datos de cualquier tipo.
- **Serial.end():** cierra la comunicación.

5.4. Sensor de humedad y temperatura del ambiente DHT11

El componente eléctrico DHT11 es un sensor digital de temperatura y humedad relativa de bajo costo, este integra un sensor capacitivo de humedad y un termistor para medir el aire circundante, y muestra los datos mediante una señal digital en el pin de datos, y posee una dimensión de 16*12*5 mm y un peso de 1 gramo. Para el hardware, se debe de conectar el pin VCC de alimentación a 3-5V, el pin GND a Tierra (0V) y el pin de datos a un pin digital del Arduino UNO, y para el software se dispone de librerías para Arduino. [7]

Las características son:

- Voltaje de Operación: 3V - 5V DC
- Rango de medición de temperatura: 0 a 50 °C
- Precisión de medición de temperatura: ±2.0 °C
- Rango de medición de humedad: 20 % a 90 % RH.
- Precisión de medición de humedad: 5 % RH.

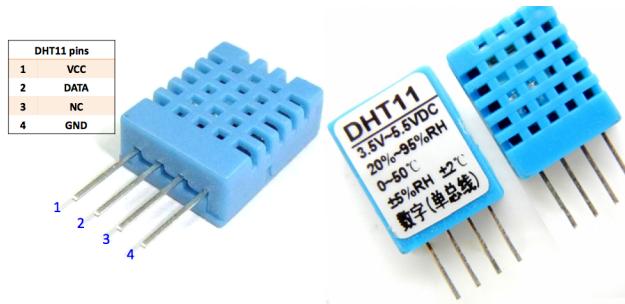


Figura 3: Elemento eléctrico sensor de humedad y temperatura DHT11 [7]

5.5. Sensor capacitivo de humedad del suelo v1.2

Es un sensor de humedad del suelo capacitivo analógico, que mide los niveles de humedad del suelo mediante detección capacitiva , es decir, la capacitancia varía según el contenido de agua presente en el suelo. La capacitancia se convierte en nivel de voltaje básicamente de 1,2 V a 3,0 V como máximo. La ventaja del sensor de humedad del suelo capacitivo es que están hechos de un material resistente a la corrosión que le da una larga vida útil. [8]

Una característica importante es que admite interfaz de sensor de 3 pines.

- **GND:** Tierra, debe ser común con la MCU
- **VCC:** 3,3 V – 5,5 V CC
- **AOUT:** Salida analógica que se conectada a una entrada analógica en una MCU

Las características son:

- Voltaje de Operación: 3,3V - 5,5V CC
- Voltaje de salida: 0v - 3,0V CC
- Corriente: 5mA
- Interfaz: PH2.0-3P
- Tamaño: 99x16 mm/3,9x0. 63



Figura 4: Elemento eléctrico sensor capacitivo de humedad del suelo [8]

5.6. Sensor de nivel del agua

El Sensor de Nivel Vertical es un dispositivo electromecánico que permite detectar el nivel de agua, actúa como un switch o interruptor que cambia de estado cuando el nivel de agua alcanza la pieza móvil del sensor y cuando se baja densidad se correrá según la altura del agua. Este sensor es utilizado para medir el nivel de agua dentro de un tanque, entonces el sensor puede activar una bomba, una alarma u otros dispositivos. [5]

Las características son:

- Voltaje de operación: 0-100 V
- Corriente de operación: 0.5 A
- Potencia máxima: 10W
- Temperatura de funcionamiento: -10°C a 60 °C
- Dimensiones: 68 mm x 24 mm
- Posición de uso: Vertical

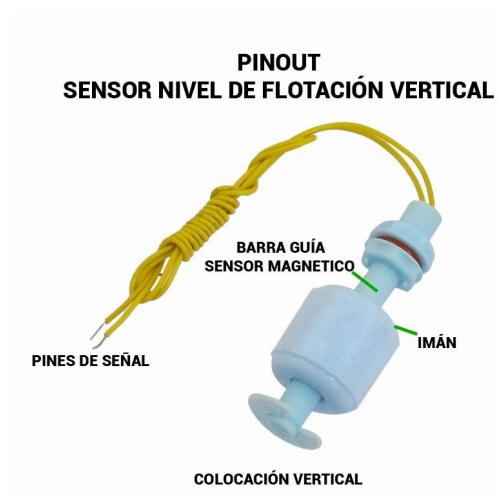


Figura 5: Elemento eléctrico microbomba de agua [5]

5.7. Módulo relay

El relé es un dispositivo electromecánico que actúa como un interruptor, posee 1 Relays de alta calidad, capaz de manejar cargas de hasta 250V/10A. Cada canal posee aislamiento eléctrico por medio de un optoacoplador y un LED indicador de estado. Este módulo Relay activa la salida normalmente abierta (NO: Normally Open) al recibir un 0 lógico, o sea 0 V y desactiva la salida con un 1 lógico, o sea 5 V. Consta de contactos normalmente abiertos (NO), contactos normalmente cerrados (NC) y un contacto común (COM). [6]

Los pines de entrada son:

- **GND:** Pin de tierra
- **VCC:** Pin de alimentación 5 V
- **In1:** Este pin puede estar en bajo o en alto

Las características son:

- Voltaje de operación: 5V DC
- Corriente de operación: 0.2 A
- Corriente máximo en contactos: 250V AC / 10A
- Tiempo de acción: 10 ms / 5 ms
- Dimensiones: 4.2cm x 2.2cm x 2cm

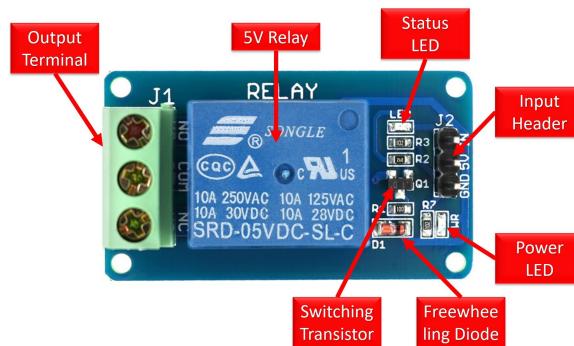


Figura 6: Elemento eléctrico microbomba de agua [6]

5.8. Módulo de Bluetooth HC-06

El módulo Bluetooth HC-06 es ideal para utilizar en todo tipo de proyectos donde se necesitan conexiones inalámbrica fiables y sencillas de utilizar. Se configura mediante comandos AT y solo puede funcionar en modo esclavo. La comunicación siempre será desde un dispositivo maestro hacia el receptor HC-06 esclavo. Además, puede alimentarse con una tensión de entre 3.3 y 6V (normalmente 5V), pero los pines TX y RX utilizan niveles de 3,3V por lo que no se puede conectar directamente a placas de 5V. Módulo montado en tarjeta con regulador de voltaje y 4 pines suministrando acceso a VCC, GND, TXD, y RXD. Tiene un LED incorporado que indica el estado de la conexión y si está emparejado o no en función de la velocidad del parpadeo, la velocidad por defecto (baud rate): 9600. [3]

Los 4 pines son: [4]

- **RX:** Pin para recepción de datos
- **TX:** Pin para transmisión de los datos
- **GND:** Va conectado al pin GND de la placa de Arduino
- **VCC:** Es el pin de alimentación de 3.3 V a 6 V

Las características son: [4]

- Voltaje de operación: 3.3 V a 6 V
- Corriente de operación: 50 mA
- Frecuencia: banda ISM de 2,4 GHz
- Temperatura de operación: -25 °C a +75 °C
- Velocidad: Asincrónica: 2 Mbps (max.)/160 kbps, sincrónica: 1 Mbps/1 Mbps
- Sensibilidad: -80 dBm a 0,1% BER
- Dimensiones totales: 1.7 cm x 4 cm aprox.
- Potencia de emisión: 6 dBm, Clase 2

- Alcance 5 a 10 metros

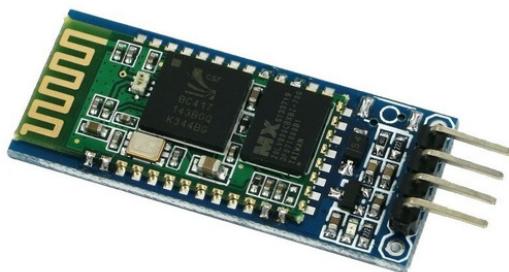


Figura 7: Elemento eléctrico módulo Bluetooth [4]

5.9. Microbomba de agua

La bomba de agua sumergible es un componente encargado de bombejar agua o bien líquidos de un lugar a otro al recibir una corriente eléctrica, el flujo es de 80-120 I/H. En este proyecto la bomba de agua se activara cuando el sensor de humedad detecte que la tierra se encuentra seca o pase un lapso de tiempo en el que se determine la planta necesita de agua, de esta forma se puede asegurar mantener una tierra húmeda apta para la planta y evitar que esta se seque. [9]

Las características son:

- Voltaje de entrada: 2,6V - 6V
- Corriente: 130 a 220 mA
- Flujo: 80-120 I/H
- Consumo: 0,4 a 1,5 W
- Altura máxima de bombeo: 1 metro
- Entrada de agua: Diámetro de 5 mm



Figura 8: Elemento eléctrico microbomba de agua [9]

5.10. Luz LED

Las plantas que crecen en interiores necesitan de luz artificial para desarrollarse, para equilibrar la calidez y frialdad que recibe la planta mediante esta luz, se determina que la luz LED es ideal para sustituir la luz natural en estos casos, ya que se puede implementar los colores deseados acorde a las necesidades de la planta, en este caso, una combinación entre la luz roja y azul son los mayores beneficiarios para el crecimiento de las plantas ya que proporcionan un equilibrio entre lo cálido y frío.

5.11. Sistemas automatizados de riego y luz para plantas

Los sistemas de riego y luz automatizados para plantas deben de contar con una programación específica dedicada a las condiciones para el tipo de planta que se va a cuidar, sin embargo, estos parámetros son modificables para ajustarse de acuerdo a la planta deseada cuidar. En el caso de este proyecto, se van a estandarizar los valores para una de las plantas ornamentales mas comunes en Costa Rica: la *Aglaoonema*. Esta planta se encuentra a lo largo de todo el territorio nacional, sus cuidados son muy estandarizados y puede crecer con luz artificial, por lo que resulta una buenas opción para medir el comportamiento del sistema de riego y luz automatizado a desarrollar. A continuación se definen los parámetros que necesita la Aglaneoma, los cuales serán los utilizados como referencia de cuido en la programación del sistema de riego automatizado

5.11.1. Parámetros para la planta: Aglaonema

- Temperatura: Idealmente necesita mantenerse entre temperaturas de 16° a 18°, por lo que se regulara la intensidad de los leds y se mantendrá en este rango midiendo su temperatura constantemente.
- Riego: La Aglaonema no necesita exceso de agua, únicamente necesita ser regada cuando su suelo esta seco, por lo tanto en este caso la bomba de agua se activara cuando detecte que no existe humedad en el suelo.
- Nivel de PH: La Aglaonema necesita un nivel de PH ligerante ácido, de aproximadamente 6 a 6,5. Para cuidar que la planta se encuentre en este rango, el nivel de ph se medirá constantemente, y si se detecta que se encuentra fuera de este rango, enviara un mensaje de alerta al usuario para que este se encargue de nivelarlo cambiando el sustrato de la planta.

5.12. Aplicación MIT INVENTOR

App inventor es un entorno de programación que permite crear aplicaciones móviles. Está diseñado para programar aplicaciones sencillas, pero totalmente funcionales para smartphones y tablets de dispositivos Android o iOS. El usuario puede, de forma visual y a partir de un conjunto de herramientas básicas, ir enlazando una serie de bloques para crear la aplicación. Y tiene tres partes fundamentales:

- **Diseñador:** Es el lugar donde se seleccionan las componentes para la aplicación.

- **Editor de bloques:** es el lugar donde se crea la lógica del programa. Se ejecuta en una ventana independiente del diseñador de componentes y está implementado como una aplicación de Java Web Start
- **Emulador:** Es el software que imita el funcionamiento de un dispositivo móvil Android real.

En la Figura 9 se muestra el código de bloques de la aplicación desarrollada. En la primer parte del código, se define el mostrar un estado de desconectado cuando se inicializa la aplicación y no mostrar datos de nada en la sección de *Datos de Jardín*, el cual corresponde a Label2, asimismo también se programa que se cree la lista de dispositivos bluetooth's disponibles ante de presionar el botón de Conectar que corresponde al ListPicker1. En la tercer parte, se programa que luego de presionar el botón de Conectar que se conecte al seleccionado de la lista y que imprima en el Label1 si si se conecto o si hubo un error.

Luego, en el cuarto bloque, se programa lo que seria el refreshamiento de los datos haciendo uso del Clock1, para esto primero se evalúa si esta conectado al bluetooth para evitar recibir basura, y evalúa si el dispositivo bluetooth posee datos, es decir, es mayor a cero, si esto se cumple entonces los envía al Label2 que seria la sección donde se despliegan los datos en la aplicación, estos datos se reciben como bytes y se transforman a texto.

Por ultimo, se programa que al presionar el botón de Desconectado, el cual corresponde al Button1, se desconecte del dispositivo Bluetooth y cambie el estado a desconectado.



Figura 9: Código de bloques de la aplicación de App Inventor

6. Componentes utilizados

Cuadro 1: Lista de componentes usados en el proyecto

Componente	Cantidad	Precio \$
Arduino UNO	1	23
Sensor DHT11	1	5,95
Sensor nivel de agua	1	2.54
Módulo Bluetooth HC-06	1	9,95
Microbomba	1	2,49
Sensor capacitivo	1	5,95
Módulo Relay	1	2
LED RGB	1	0.55
Protoboard	2	10
kit resistores	1	8,95

7. Diseño del circuito

En la Figura 10 se encuentran el diseño del circuito con todas sus respectivos componentes. A destacar que para el módulo bluetooth, el VCC no se conectó directamente a un pin de alimentación, sino al pin digital 12, esto se debe a que si se conecta directamente a la alimentación el módulo se alimentaría antes de haber cargado el sketch y el pin RX se encontrara ocupado, lo que generaría un error de comunicación, por lo que si se conecta en un pin digital se mantiene el pin con un LOW y una vez que se cargue el sketch se envía un HIGH en el pin digital, de esta forma no debemos conectar y desconectar el VCC cada vez que cargamos un sketch.

Para el diseño de los resistores, se tomó en cuenta los valores corriente soportados por los componentes del DHT11 y el led. Para el led, acorde a la hoja de datos del fabricante, se tiene una corriente máxima de 2mA, por lo que se eligió un valor comercial de 220Ω , lo cual según la ley de Ohm, si lo tenemos conectado a un voltaje de 5V, obtenemos una corriente de 1.1mA aproximadamente. Asimismo, se utilizó una resistencia de pull-down de $47k\Omega$, se eligió este valor comercial alto para asegurar una baja resistencia cuando el valor que lee el sensor de DHT11 es cero.

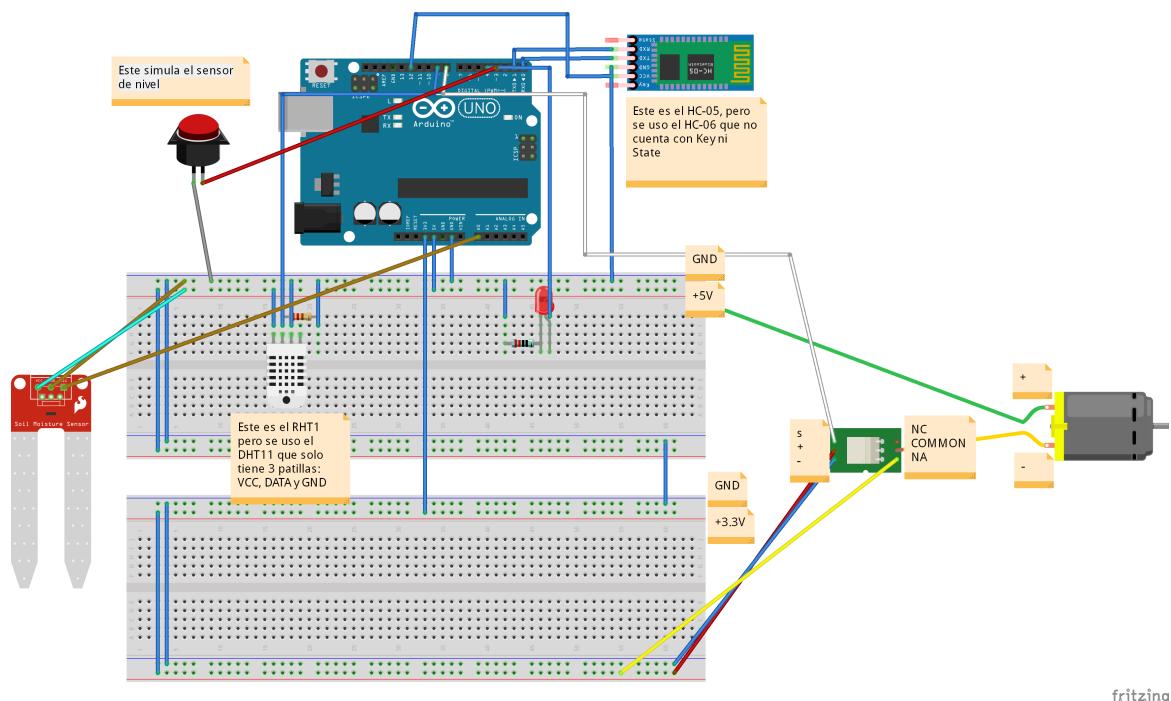


Figura 10: Esquemático del circuito

[Diseño de resistencias]

8. Análisis de resultados

8.1. Desarrollo de código

8.1.1. Lectura de temperatura, humedad y sensación térmica del ambiente mediante el DHT11

Para lograr leer los valores de la temperatura, humedad y sensación térmica del ambiente se hizo uso del sensor DHT11, el cual es capaz de medir la temperatura y expresarla en grados celsius y la humedad en porcentaje. Para obtener esto se utilizó la librería del sensor *DHT.h* y *DHT_U.h*, la entrada de DATA del sensor se conectó al pin digital 9 del arduino. Se creó un objeto sensor y se inicializó en la función de **setup**. Finalmente, mediante las funciones *readTemperature*, *readHumidity* y *computeHeatIndex* se obtienen los parámetros de temperatura, humedad y sensación térmica respectivamente.

```

1 temp = dhtsensor.readTemperature();
2 hum = dhtsensor.readHumidity();
3 sen_ter = dhtsensor.computeHeatIndex(temp, hum, false);

```

8.2. Lectura de la humedad del la tierra mediante el sensor capacitive del suelo

Para poder obtener el estado de humedad de la tierra haciendo uso del sensor de humedad del suelo, primero se realizo una calibración para obtener los valores correspondientes a seco estando totalmente en el aire y a mojado estando sumergido en el agua, este código se obtuvo de la hoja de datos proporcionada por el vendedor, el cual se muestra a continuación:

```

1 void setup() {
2     Serial.begin(9600); // open serial port, set the baud rate as 9600 bps
3 }
4 void loop() {
5     int val;
6     val = analogRead(0); //connect sensor to Analog 0
7     Serial.print(val); //print the value to serial port
8     delay(100);
9 }
```

Con el código de calibración anterior se obtuvo que el valor en seco es de 442 y el valor en agua es de 186, a partir de estos valores se procedió a dividir los tres casos posibles que se puede obtener: tierra seca, tierra mojada y tierra húmeda. Para esto se obtuvo la división del rango en tres partes, este intervalo corresponde a la variable **intervals**, el cual si lo visualizamos con los valores corresponde a:

- Caso tierra mojado: [186 - 271.33]
- Caso tierra húmeda: [271.34 - 356.67]
- Caso tierra seca: [356.68 - 442]

De esta forma, en el código al entrar a la función **loop** se subdividen los casos con **if's** a partir de este valor y en que rango se encuentre. El valor se lee a partir del pin analógico A0.

8.2.1. Conexión bluetooth mediante modulo HC-06

Para establecer una conexión bluetooth con un dispositivo externo, se hizo uso del modulo esclavo HC-06 y la librería *SoftwareSerial.h*, para su configuración se utilizaron comandos AT, los cuales se ingresaron imprimiéndolos en pantalla con sus respectivos valores deseados tal como se muestra en el código a continuación:

```

1 Serial.print("AT"); //comando AT para indicar que va a configurar
2 delay(1000); //delay para que sepa que se termino el primer comando
3
4 Serial.print("AT+NAME"); //se le cambia el nombre
5 Serial.print(name);
6 delay(1000);
7
8 Serial.print("AT+BAUD"); //se configura la velocidad
9 Serial.print(speed);
10 delay(1000);
11
12 Serial.print("AT+PIN");
13 Serial.print(pin);
14 delay(1000);
```

En las variables **name**, **speed** y **pin** se guardaron previamente los valores para el nombre del dispositivo bluetooth, la velocidad de transmisión de datos y el pin respectivamente. Para el nombre se eligió *MySmartGarden* ya que este es el nombre que se le dio a la aplicación móvil, el pin se dejó como uno genérico *0000* la velocidad se configuró en 9600 baudios, el cual corresponde a un *4*.

8.2.2. Lógica del código

Al inicio del código se incluyen las librerías y se declaran todas las variables a utilizar, asimismo se crea el; objeto del sensor DHT.

```

1 #include <DHT.h>
2 #include <DHT_U.h>
3 #include <SoftwareSerial.h>
4
5
6 const int dry_value = 442; //valor que mide el sensor en seco
7 const int water_value = 186; //valor que mide el sensor cuando esta en agua
8 int intervals = (dry_value - water_value)/3;
9 int soil_moisture_value = 0; // variable para guardar el valor de humedad de la
10 tierra
11 int water_pulp = 8; // pin digital donde se conecta la bomba de agua
12 int state = 1; // valor para medir estado de la humedad, 1 es mojado, 0 es seco
13 int temp_pin = 9; //pin digital donde se conecta el sensor
14 int temp; //variable para almacenar la temperatura
15 int hum; //variable para almacenar la humedad
16 float sen_ter; // variable para almacenar la sensacion termica
17 int water_sensor = 3;
18 int water_level;
19 int temp_led = 4;
20
21 //variables para para modulo bluetooth
22 const int LED = 13;
23 const int BT_power = 12;
24 char name[14] = "MySmartGarden";
25 char speed ='4';//9600
26 char pin [5]= "0000";
27
28 DHT dhtsensor(temp_pin, DHT11); //se crea objeto sensor
SoftwareSerial blue(2,3);

```

Luego, en la función **setup** se inicia la comunicación serial, el objeto del sensor DHT, se definen los pines necesarios como salidas y se realiza la configuración del modulo bluetooth con los comandos AT descritos en la sección de *Conexión bluetooth mediante modulo HC-06*.

En la función **loop** se comienza leyendo los datos de la temperatura, humedad, sensación térmica mediante el sensor DHT11 y el nivel de agua mediante el sensor de nivel de agua, el cual devuelve un 0 si detecta agua y un 1 si su flotador se encuentra abajo por falta de presión de agua. Seguido de esto, se procede a evaluar si la temperatura del ambiente es menor a 10°C, si la temperatura es menor entonces se entra al **if** y se enciende un led, el cual simula la situación de tener una fuente de luz mas grande que al encender proporcione calor a la planta en caso de que la temperatura sea muy baja.

```

1 if (temp < 10){
2     Serial.println("Baja temperatura , led encendido");
3     digitalWrite(temp_led, HIGH);

```

```

4 }
5
6 else{
7   digitalWrite (temp_led , LOW);
8 }
```

Después de evaluar la temperatura, se procede a imprimir los datos con comunicación serial.

Luego se lee el dato indicador del nivel de humedad de la tierra como se mencionó anteriormente en la sección *Lectura de la humedad de la tierra mediante el sensor capacitivo del suelo*, se ingresa a cada caso según el valor, donde en los casos si el dato leído del sensor se encuentre en los rangos de tierra húmeda y tierra mojada no sucede nada y solo se indica el estado de la tierra, sin embargo si el dato de la humedad del suelo se encuentra en el rango de tierra seca, se indica que se está intentando encender la bomba, para guardar el estado de esta evaluación, se actualiza la variable **state**, donde un **1** indica que la tierra está mojada o húmeda, y un **0** que la tierra se encuentra seca.

```

1 if(soil_moisture_value > water_value && soil_moisture_value < (water_value +
2   intervals)){
3
4   state = 1;
5   Serial.println("Estado de la tierra: Mojada");
6   Serial.println("\n");
7   delay(1000);}
8
9 else if(soil_moisture_value > (water_value + intervals) && soil_moisture_value
10 < (dry_value - intervals)){
11
12   state = 1;
13   Serial.println("Estado de la tierra: Humeda");
14   Serial.println("\n");
15   delay(1000);}
16
17 else if(soil_moisture_value < dry_value && soil_moisture_value > (dry_value -
18   intervals)){
19
20   state = 0;
21   Serial.println("Estado de la tierra: Seca");
22   Serial.println("\n");
23   delay(1000);
24   Serial.println("Encendiendo la bomba de agua...");}
```

De esta forma, luego de saber el estado de humedad de la tierra, se vuelve a evaluar mediante un **if** si el estado es **1** o **0**, si el estado es 1 la bomba de agua se mantiene apagada ya que la tierra ya se encuentra mojada o húmeda, sin embargo en el caso de que el estado sea 0 que indica que la tierra está seca, se debe encender la bomba, no obstante antes de esto, se debe verificar el nivel de agua, se revisa si el nivel de agua es alto (cero en la variable *water_level*) y si es así, se enciende la bomba y se indica que se está suministrando agua, en caso de que no haya agua suficiente en el tanque (uno en la variable *water_level*), no se enciende la bomba, sino que se le envía una señal de bajo para apagarla ya que puede encontrarse encendida debido al ciclo anterior cuando aún había agua, y se envía un mensaje de que el nivel de agua es insuficiente y se debe de llenar el tanque.

```

1 if(state == 0){
2
3   if(water_level == 0){
4     delay(2000);
```

```

5   digitalWrite(water_pulp, HIGH);
6   Serial.println("Suministrando agua... ");
7   delay(5000);

8
9   else if (water_level == 1){
10    digitalWrite(water_pulp, LOW);
11    Serial.println("Insuficiente nivel de agua, llenar el tanque");
12    delay(5000);
13 }

14
15  else if(state == 1){

16    digitalWrite(water_pulp, LOW);
17    delay(1000);

18
19 }

20

```

8.3. Desarrollo de hardware

Para el desarrollo del hardware, se utilizó un arduino UNO capaz de controlar los sensores, el módulo Bluetooth, un led y la microbomba dependiendo de lo que se definió en el código, explicado anteriormente en la sección *Desarrollo del código*. Este está conectado a la computadora por medio de un cable USB, y a una protoboard en dónde se encuentra los demás componentes eléctricos. En donde las conexiones de los pines son:

- **Pin 0:** conexión con el pin TX del módulo Bluetooth.
- **Pin 1:** conexión con el pin RX del módulo Bluetooth.
- **Pin 3:** conexión con el sensor del nivel del agua.
- **Pin 4:** conexión con el LED.
- **Pin 8:** conexión con la bomba de agua.
- **Pin 9:** conexión con el sensor DHT11.
- **Pin 12:** conexión con el pin VCC del módulo Bluetooth.
- **Pin A0:** conexión analógica con el sensor DHT11 que lee el valor de humedad.
- **Pin 5V**
- **Pin 3.3V**
- **Pin GND**



Figura 11: Sistema de riego completo



Figura 12: Planta con sensor de humedad de suelo

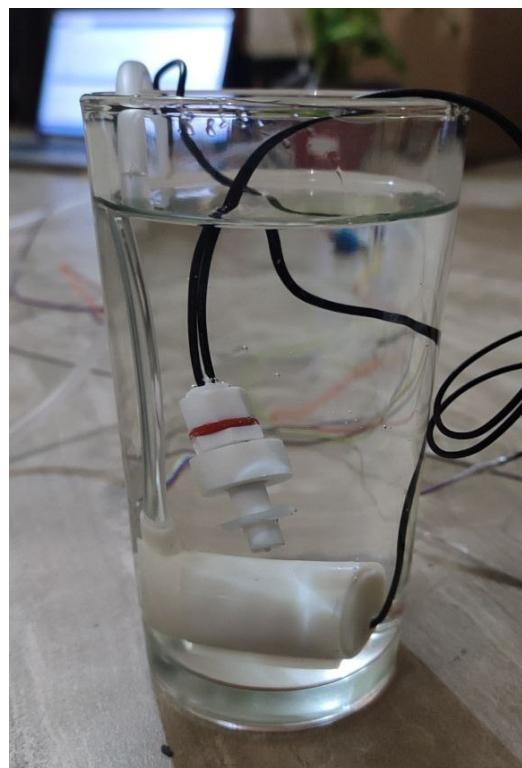


Figura 13: Bomba de agua y sensor de nivel de agua

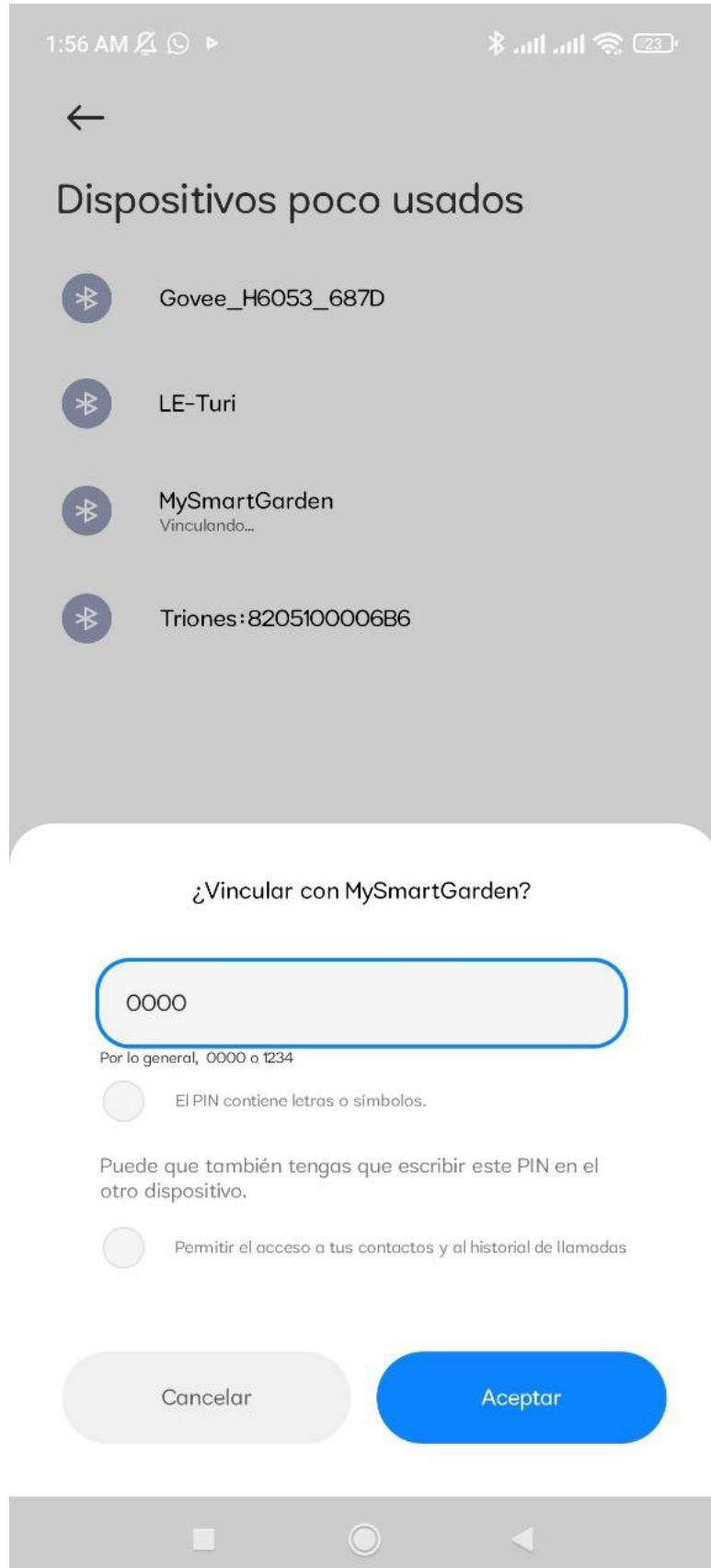
El sensor DHT11 está conectado

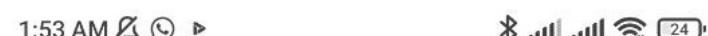
El módulo Bluetooth

8.4. Desarrollo de aplicación

8.5. Pruebas del funcionamiento

8.5.1. Conexión de aplicación móvil y recepción de datos





Bluetooth

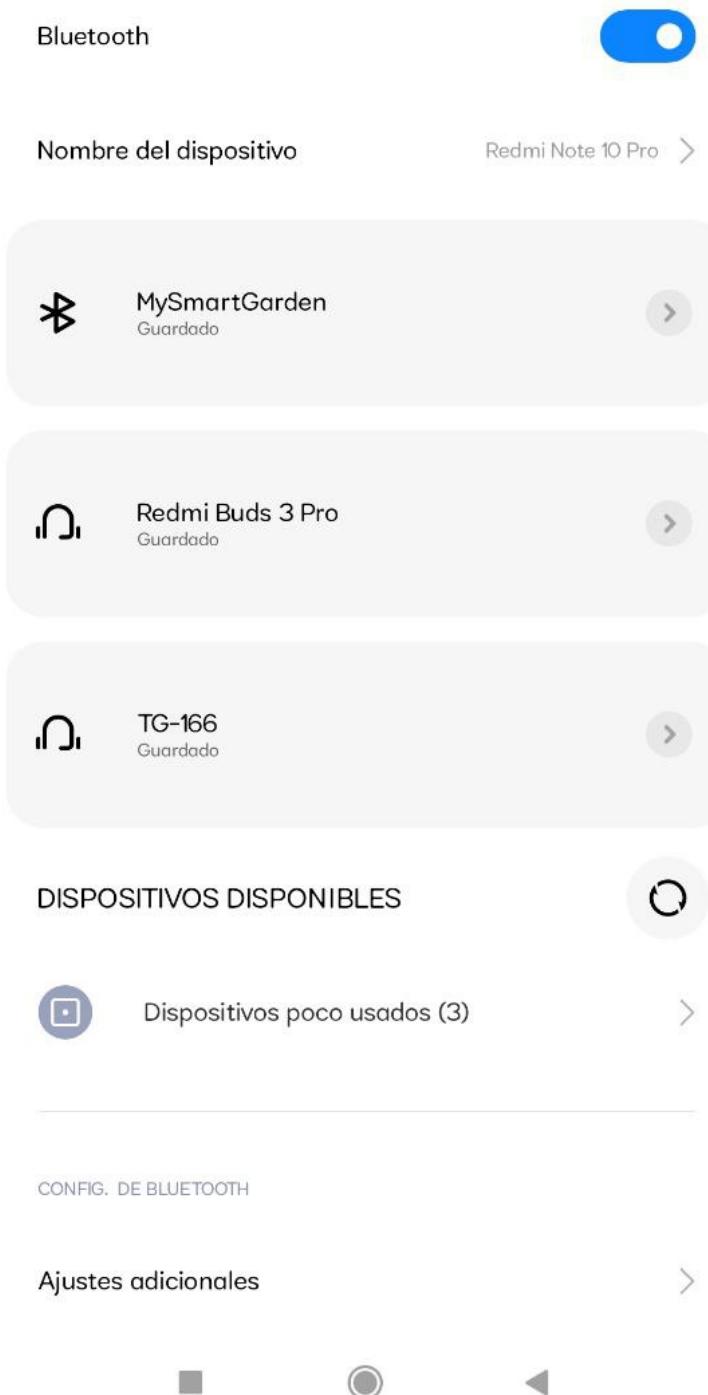


Figura 15: Dispositivo bluetooth vinculado



Figura 16: Dispositivos bluetooth desde la aplicación móvil



Figura 17: Recepción de estado desde la aplicación móvil

8.5.2. Caso 1: tierra seca

En el caso que se detecta que la tierra se encuentra seca, la bomba se enciende siempre y cuando haya agua en el tanque. En las Figuras ?? y 19 se encuentra la información presentada en el monitor serial y aplicación móvil cuando este caso sucede.

Estado de la tierra: Seca

Encendiendo la bomba de agua...

Suministrando agua...

Temperatura del ambiente: 23 C

Humedad del ambiente: 56%

Sensacion termica del ambiente: 22.82 C

Figura 18: Monitor serial en caso de tierra seca



Figura 19: Aplicación móvil en caso de tierra seca

8.5.3. Caso 2: tierra húmeda

En el caso de que la tierra este húmeda, se notifica que esta está húmeda y prosigue tomando datos de la temperatura y humedad, sin encender la bomba de agua. En las Figuras 20 y 21 se encuentra la información presentada en el monitor serial y y aplicación móvil cuando este caso sucede.

Temperatura del ambiente: 25 C
Humedad del ambiente: 49%
Sensacion termica del ambiente: 24.83 C

Estado de la tierra: Humeda

Temperatura del ambiente: 25 C
Humedad del ambiente: 49%
Sensacion termica del ambiente: 24.83 C

Figura 20: Monitor serial en caso de tierra húmeda

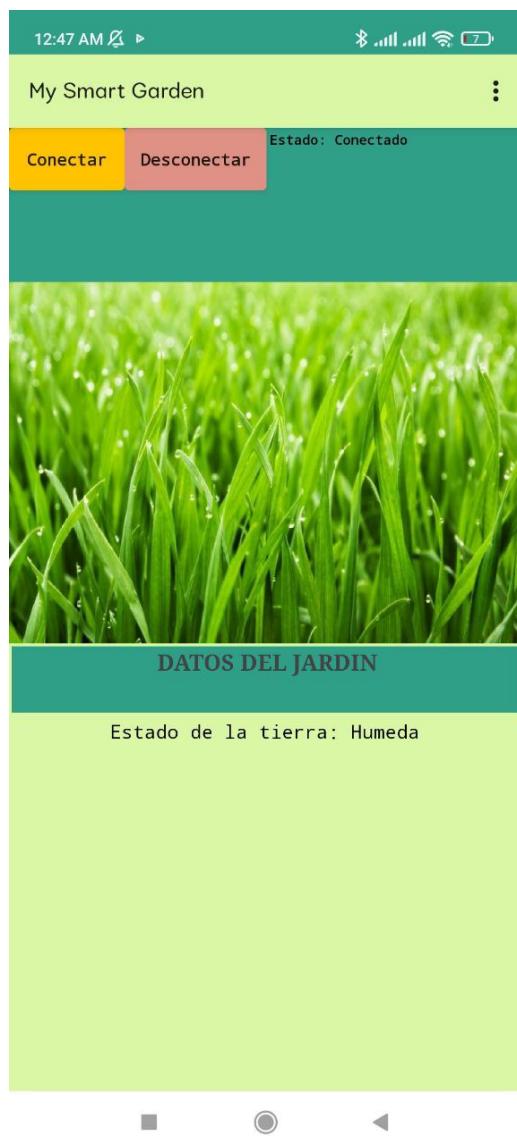


Figura 21: Aplicación móvil en caso de tierra húmeda

8.5.4. Caso 3: tierra mojada

En el caso de que la tierra este mojada, se notifica que esta esta mojada y prosigue tomando datos de la temperatura y humedad, sin encender la bomba de agua. En las Figuras 22 y 23 se encuentra la información presentada en el monitor serial y aplicación móvil cuando este caso sucede.

Temperatura del ambiente: 23 C
Humedad del ambiente: 53%
Sensacion termica del ambiente: 22.74 C

Estado de la tierra: Mojada

Temperatura del ambiente: 23 C
Humedad del ambiente: 53%
Sensacion termica del ambiente: 22.74 C

Figura 22: Monitor serial en caso de tierra mojada



Figura 23: Aplicación móvil en caso de tierra mojada

8.5.5. Caso 4: sin agua en el tanque

En el caso de que la tierra este seca y se vaya o este suministrando agua, se notifica que no hay agua en el tanque y que se debe de llenar, también se detiene la bomba de agua. En las Figuras 24 y 25 se encuentra la información presentada en el monitor serial y aplicación móvil cuando este caso sucede.

Temperatura del ambiente: 23 C
Humedad del ambiente: 53%
Sensacion termica del ambiente: 22.74 C

Insuficiente nivel de agua, llenar el tanque

Temperatura del ambiente: 23 C
Humedad del ambiente: 53%
Sensacion termica del ambiente: 22.74 C

Figura 24: Monitor serial en caso de insuficiente agua



Figura 25: Aplicación móvil en caso de insuficiente agua

8.5.6. Caso 5: temperatura menor a 10°C

En el caso de que la temperatura sea menor a 10°C, se notifica que la temperatura es muy baja y que se encenderá la luz al mismo tiempo que se enciende la luz para brindar calor a la planta. En las Figura 26 y 27 se encuentra la información presentada en el monitor serial y aplicación móvil cuando este caso sucede. En esta prueba lo que se hizo para obtener una temperatura menor a 10°C de manera inmediata ya que las temperaturas donde habitamos las integrantes son mayores a este valor, es que se desconectó un cable del sensor DHT11 al pin digital 9 del arduino para que el dato a leer sea cero y poder obtener el caso donde la luz se enciende por baja temperatura, es por esto que los datos que se muestran en la imagen tienen valores de 0°C y -3.94°C, ya que los datos medidos son erróneos a causa de la desconexión de su alimentación, sin embargo otra forma de obtener este resultado también es modificando el valor límite de cuando se enciende la luz.

```
Temperatura del ambiente: 23 C
Humedad del ambiente: 56%
Sensacion termica del ambiente: 22.82 C
```

```
Suministrando agua...
Baja temperatura, led encendido
```

```
Temperatura del ambiente: 0 C
Humedad del ambiente: 0%
Sensacion termica del ambiente: -3.94 C
```

Figura 26: Monitor serial en caso de temperatura menor a 10°C



Figura 27: Aplicación móvil en caso de temperatura menor a 10°C

9. Conclusiones

- Se logró desarrollar un sistema de jardinería de riego y luz automática, en donde el usuario puede visualizar las condiciones de este a través de una aplicación desde el celular.
- Es necesario recordar que las corrientes y tensiones manejadas por un Arduino son bastante bajas, esto imposibilita que dicha plataforma pueda controlar directamente dispositivos de alta potencia, como una bomba de mayor magnitud, por lo que es necesario usar un componente intermediario entre el Arduino y el dispositivo de alta potencia.
- A modo de recomendación, sería usar un bombillo o una banda de led's, para lograr una temperatura ambiente adecuada para la planta.
- Entre las diferencias entre propuesto y lo realizado es que en lugar de utilizar un modulo wifi para la transmisión de datos, se utilizo un modulo Bluetooth, esto debido a cuestiones de presupuesto y tiempo de conseguir todas las piezas que se necesitaban para la

implementación, asimismo, no se implemento un sensor de PH como originalmente decía la propuesta ya que no se encontró en ninguna de las tiendas un sensor de este tipo.

10. Cronograma

Semana 1-2:

- Maybelle: Investigación en internet sobre sistemas de riego automático para plantas.
- Sinaí: Investigación en internet los componentes necesarios para el sistema.

Semana 3-4:

- Maybelle: Diseño e implementación del hardware del sistema.
- Sinaí: Diseño e implementación del software del sistema.

Semana 5-6:

- Maybelle: Validación del sistema cuando las plantas no necesitan riego.
- Sinaí: Validación del sistema cuando las plantas necesitan riego.

Semana 7-8:

- Maybelle: Elaboración del informe y la presentación del proyecto.
- Sinaí: Elaboración del informe y la presentación del proyecto. [[knuthwebsite](#)]

Referencias

- [1] “Arduino UNO”. En: *Data Sheet*. 2001.
- [2] “ATmega328P”. En: *Data Sheet*. 2011.
- [3] BricoGeek. *Módulo Bluetooth HC-06*. 2015. URL: <https://tienda.bricogeek.com/modulos-bluetooth/1351-modulo-bluetooth-hc-06.html>.
- [4] Electronicos. *HC-06*. 2015. URL: <https://www.electronicoscaldas.com/es/modulos-rf/482-modulo-bluetooth-hc-06.html>.
- [5] UNIT electronics. *Sensor De Nivel Flotador Vertical*. 2014. URL: <https://uelectronics.com/producto/sensor-de-nivel-flotador-vertical/>.
- [6] Microcontrollerslab. *5V Single Channel Relay Module*. 2003. URL: <https://microcontrollerslab.com/5v-single-channel-relay-module-pinout-working-interfacing-applications-datasheet/>.

- [7] Naylamp. *SENSOR DE TEMPERATURA Y HUMEDAD RELATIVA DHT11*. 2002. URL: <https://naylampmechatronics.com/sensores-temperatura-y-humedad/57-sensor-de-temperatura-y-humedad-relativa-dht11.html>.
- [8] ProtoSuplies. *Capacitive Soil Moisture Sensor Module*. 2010. URL: <https://protosupplies.com/product/capacitive-soil-moisture-sensor-module/>.
- [9] Steren. *Micro bomba de agua con flujo de 80-120 l/h.* 2018. URL: <https://www.steren.cr/micro-bomba-de-agua-con-flujo-de-80-120-l-h.html>.

11. Anexos

Introduction

The Atmel® picoPower® ATmega328/P is a low-power CMOS 8-bit microcontroller based on the AVR® enhanced RISC architecture. By executing powerful instructions in a single clock cycle, the ATmega328/P achieves throughputs close to 1MIPS per MHz. This empowers system designer to optimize the device for power consumption versus processing speed.

Feature

High Performance, Low Power Atmel®AVR® 8-Bit Microcontroller Family

- Advanced RISC Architecture
 - 131 Powerful Instructions
 - Most Single Clock Cycle Execution
 - 32 x 8 General Purpose Working Registers
 - Fully Static Operation
 - Up to 20 MIPS Throughput at 20MHz
 - On-chip 2-cycle Multiplier
- High Endurance Non-volatile Memory Segments
 - 32KBytes of In-System Self-Programmable Flash program Memory
 - 1KBytes EEPROM
 - 2KBytes Internal SRAM
 - Write/Erase Cycles: 10,000 Flash/100,000 EEPROM
 - Data Retention: 20 years at 85°C/100 years at 25°C⁽¹⁾
 - Optional Boot Code Section with Independent Lock Bits
 - In-System Programming by On-chip Boot Program
 - True Read-While-Write Operation
 - Programming Lock for Software Security
- Atmel® QTouch® Library Support
 - Capacitive Touch Buttons, Sliders and Wheels
 - QTouch and QMatrix® Acquisition
 - Up to 64 sense channels

- Peripheral Features
 - Two 8-bit Timer/Counters with Separate Prescaler and Compare Mode
 - One 16-bit Timer/Counter with Separate Prescaler, Compare Mode, and Capture Mode
 - Real Time Counter with Separate Oscillator
 - Six PWM Channels
 - 8-channel 10-bit ADC in TQFP and QFN/MLF package
 - Temperature Measurement
 - 6-channel 10-bit ADC in PDIP Package
 - Temperature Measurement
 - Two Master/Slave SPI Serial Interface
 - One Programmable Serial USART
 - One Byte-oriented 2-wire Serial Interface (Philips I²C compatible)
 - Programmable Watchdog Timer with Separate On-chip Oscillator
 - One On-chip Analog Comparator
 - Interrupt and Wake-up on Pin Change
- Special Microcontroller Features
 - Power-on Reset and Programmable Brown-out Detection
 - Internal Calibrated Oscillator
 - External and Internal Interrupt Sources
 - Six Sleep Modes: Idle, ADC Noise Reduction, Power-save, Power-down, Standby, and Extended Standby
- I/O and Packages
 - 23 Programmable I/O Lines
 - 28-pin PDIP, 32-lead TQFP, 28-pad QFN/MLF and 32-pad QFN/MLF
- Operating Voltage:
 - 1.8 - 5.5V
- Temperature Range:
 - -40°C to 105°C
- Speed Grade:
 - 0 - 4MHz @ 1.8 - 5.5V
 - 0 - 10MHz @ 2.7 - 5.5V
 - 0 - 20MHz @ 4.5 - 5.5V
- Power Consumption at 1MHz, 1.8V, 25°C
 - Active Mode: 0.2mA
 - Power-down Mode: 0.1µA
 - Power-save Mode: 0.75µA (Including 32kHz RTC)

Table of Contents

Introduction.....	1
Feature.....	1
1. Description.....	9
2. Configuration Summary.....	10
3. Ordering Information	11
3.1. ATmega328	11
3.2. ATmega328P	12
4. Block Diagram.....	13
5. Pin Configurations.....	14
5.1. Pin-out.....	14
5.2. Pin Descriptions.....	17
6. I/O Multiplexing.....	19
7. Resources.....	21
8. Data Retention.....	22
9. About Code Examples.....	23
10. Capacitive Touch Sensing.....	24
10.1. QTouch Library.....	24
11. AVR CPU Core.....	25
11.1. Overview.....	25
11.2. ALU – Arithmetic Logic Unit.....	26
11.3. Status Register.....	26
11.4. General Purpose Register File.....	28
11.5. Stack Pointer.....	29
11.6. Instruction Execution Timing.....	31
11.7. Reset and Interrupt Handling.....	32
12. AVR Memories.....	34
12.1. Overview.....	34
12.2. In-System Reprogrammable Flash Program Memory.....	34
12.3. SRAM Data Memory.....	35
12.4. EEPROM Data Memory.....	36
12.5. I/O Memory.....	37
12.6. Register Description.....	38
13. System Clock and Clock Options.....	48

13.1. Clock Systems and Their Distribution.....	48
13.2. Clock Sources.....	49
13.3. Low Power Crystal Oscillator.....	51
13.4. Full Swing Crystal Oscillator.....	52
13.5. Low Frequency Crystal Oscillator.....	53
13.6. Calibrated Internal RC Oscillator.....	54
13.7. 128kHz Internal Oscillator.....	55
13.8. External Clock.....	56
13.9. Timer/Counter Oscillator.....	57
13.10. Clock Output Buffer.....	57
13.11. System Clock Prescaler.....	57
13.12. Register Description.....	58
14. PM - Power Management and Sleep Modes.....	62
14.1. Overview.....	62
14.2. Sleep Modes.....	62
14.3. BOD Disable.....	63
14.4. Idle Mode.....	63
14.5. ADC Noise Reduction Mode.....	63
14.6. Power-Down Mode.....	64
14.7. Power-save Mode.....	64
14.8. Standby Mode.....	65
14.9. Extended Standby Mode.....	65
14.10. Power Reduction Register.....	65
14.11. Minimizing Power Consumption.....	65
14.12. Register Description.....	67
15. SCRST - System Control and Reset.....	72
15.1. Resetting the AVR.....	72
15.2. Reset Sources.....	72
15.3. Power-on Reset.....	73
15.4. External Reset.....	74
15.5. Brown-out Detection.....	74
15.6. Watchdog System Reset.....	75
15.7. Internal Voltage Reference.....	75
15.8. Watchdog Timer.....	76
15.9. Register Description.....	78
16. Interrupts.....	82
16.1. Interrupt Vectors in ATmega328/P.....	82
16.2. Register Description.....	84
17. EXINT - External Interrupts.....	87
17.1. Pin Change Interrupt Timing.....	87
17.2. Register Description.....	88
18. I/O-Ports.....	97
18.1. Overview.....	97
18.2. Ports as General Digital I/O.....	98

18.3. Alternate Port Functions.....	101
18.4. Register Description.....	113
19. TC0 - 8-bit Timer/Counter0 with PWM.....	125
19.1. Features.....	125
19.2. Overview.....	125
19.3. Timer/Counter Clock Sources.....	127
19.4. Counter Unit.....	127
19.5. Output Compare Unit.....	128
19.6. Compare Match Output Unit.....	130
19.7. Modes of Operation.....	131
19.8. Timer/Counter Timing Diagrams.....	135
19.9. Register Description.....	137
20. TC1 - 16-bit Timer/Counter1 with PWM.....	149
20.1. Overview.....	149
20.2. Features.....	149
20.3. Block Diagram.....	149
20.4. Definitions.....	150
20.5. Registers.....	151
20.6. Accessing 16-bit Registers.....	151
20.7. Timer/Counter Clock Sources.....	154
20.8. Counter Unit.....	154
20.9. Input Capture Unit.....	155
20.10. Output Compare Units.....	157
20.11. Compare Match Output Unit.....	159
20.12. Modes of Operation.....	160
20.13. Timer/Counter Timing Diagrams.....	168
20.14. Register Description.....	169
21. Timer/Counter 0, 1 Prescalers.....	186
21.1. Internal Clock Source.....	186
21.2. Prescaler Reset.....	186
21.3. External Clock Source.....	186
21.4. Register Description.....	187
22. TC2 - 8-bit Timer/Counter2 with PWM and Asynchronous Operation.....	189
22.1. Features.....	189
22.2. Overview.....	189
22.3. Timer/Counter Clock Sources.....	191
22.4. Counter Unit.....	191
22.5. Output Compare Unit.....	192
22.6. Compare Match Output Unit.....	194
22.7. Modes of Operation.....	195
22.8. Timer/Counter Timing Diagrams.....	199
22.9. Asynchronous Operation of Timer/Counter2.....	200
22.10. Timer/Counter Prescaler.....	202
22.11. Register Description.....	202

23. SPI – Serial Peripheral Interface.....	215
23.1. Features.....	215
23.2. Overview.....	215
23.3. \overline{SS} Pin Functionality.....	219
23.4. Data Modes.....	219
23.5. Register Description.....	220
24. USART - Universal Synchronous Asynchronous Receiver Transceiver.....	225
24.1. Features.....	225
24.2. Overview.....	225
24.3. Block Diagram.....	225
24.4. Clock Generation.....	226
24.5. Frame Formats.....	229
24.6. USART Initialization.....	230
24.7. Data Transmission – The USART Transmitter.....	231
24.8. Data Reception – The USART Receiver.....	233
24.9. Asynchronous Data Reception.....	237
24.10. Multi-Processor Communication Mode.....	239
24.11. Examples of Baud Rate Setting.....	240
24.12. Register Description.....	243
25. USARTSPI - USART in SPI Mode.....	254
25.1. Features.....	254
25.2. Overview.....	254
25.3. Clock Generation.....	254
25.4. SPI Data Modes and Timing.....	255
25.5. Frame Formats.....	255
25.6. Data Transfer.....	257
25.7. AVR USART MSPIM vs. AVR SPI.....	258
25.8. Register Description.....	259
26. TWI - 2-wire Serial Interface.....	260
26.1. Features.....	260
26.2. Two-Wire Serial Interface Bus Definition.....	260
26.3. Data Transfer and Frame Format.....	261
26.4. Multi-master Bus Systems, Arbitration and Synchronization.....	264
26.5. Overview of the TWI Module.....	266
26.6. Using the TWI.....	268
26.7. Transmission Modes.....	271
26.8. Multi-master Systems and Arbitration.....	289
26.9. Register Description.....	291
27. AC - Analog Comparator.....	299
27.1. Overview.....	299
27.2. Analog Comparator Multiplexed Input.....	299
27.3. Register Description.....	300
28. ADC - Analog to Digital Converter.....	305

28.1. Features.....	305
28.2. Overview.....	305
28.3. Starting a Conversion.....	307
28.4. Prescaling and Conversion Timing.....	308
28.5. Changing Channel or Reference Selection.....	310
28.6. ADC Noise Canceler.....	312
28.7. ADC Conversion Result.....	315
28.8. Temperature Measurement.....	316
28.9. Register Description.....	316
29. DBG - debugWIRE On-chip Debug System.....	327
29.1. Features.....	327
29.2. Overview.....	327
29.3. Physical Interface.....	327
29.4. Software Break Points.....	328
29.5. Limitations of debugWIRE.....	328
29.6. Register Description.....	328
30. BTLD - Boot Loader Support – Read-While-Write Self-Programming.....	330
30.1. Features.....	330
30.2. Overview.....	330
30.3. Application and Boot Loader Flash Sections.....	330
30.4. Read-While-Write and No Read-While-Write Flash Sections.....	331
30.5. Boot Loader Lock Bits.....	333
30.6. Entering the Boot Loader Program.....	334
30.7. Addressing the Flash During Self-Programming.....	335
30.8. Self-Programming the Flash.....	336
30.9. Register Description.....	344
31. MEMPROG- Memory Programming.....	347
31.1. Program And Data Memory Lock Bits.....	347
31.2. Fuse Bits.....	348
31.3. Signature Bytes.....	350
31.4. Calibration Byte.....	351
31.5. Page Size.....	351
31.6. Parallel Programming Parameters, Pin Mapping, and Commands.....	351
31.7. Parallel Programming.....	353
31.8. Serial Downloading.....	360
32. Electrical Characteristics.....	365
32.1. Absolute Maximum Ratings.....	365
32.2. Common DC Characteristics.....	365
32.3. Speed Grades.....	368
32.4. Clock Characteristics.....	369
32.5. System and Reset Characteristics.....	370
32.6. SPI Timing Characteristics.....	371
32.7. Two-wire Serial Interface Characteristics.....	372
32.8. ADC Characteristics.....	374
32.9. Parallel Programming Characteristics.....	375

33. Typical Characteristics ($T_A = -40^{\circ}\text{C}$ to 85°C).....	378
33.1. ATmega328 Typical Characteristics.....	378
34. Typical Characteristics ($T_A = -40^{\circ}\text{C}$ to 105°C).....	403
34.1. ATmega328P Typical Characteristics.....	403
35. Register Summary.....	428
35.1. Note.....	430
36. Instruction Set Summary.....	432
37. Packaging Information.....	436
37.1. 32-pin 32A.....	436
37.2. 32-pin 32M1-A.....	437
37.3. 28-pin 28M1.....	438
37.4. 28-pin 28P3.....	439
38. Errata.....	440
38.1. Errata ATmega328/P.....	440
39. Datasheet Revision History.....	443
39.1. Rev. A – 06/2016.....	443

1. Description

The Atmel AVR® core combines a rich instruction set with 32 general purpose working registers. All the 32 registers are directly connected to the Arithmetic Logic Unit (ALU), allowing two independent registers to be accessed in a single instruction executed in one clock cycle. The resulting architecture is more code efficient while achieving throughputs up to ten times faster than conventional CISC microcontrollers.

The ATmega328/P provides the following features: 32Kbytes of In-System Programmable Flash with Read-While-Write capabilities, 1Kbytes EEPROM, 2Kbytes SRAM, 23 general purpose I/O lines, 32 general purpose working registers, Real Time Counter (RTC), three flexible Timer/Counters with compare modes and PWM, 1 serial programmable USARTs , 1 byte-oriented 2-wire Serial Interface (I2C), a 6-channel 10-bit ADC (8 channels in TQFP and QFN/MLF packages) , a programmable Watchdog Timer with internal Oscillator, an SPI serial port, and six software selectable power saving modes. The Idle mode stops the CPU while allowing the SRAM, Timer/Counters, SPI port, and interrupt system to continue functioning. The Power-down mode saves the register contents but freezes the Oscillator, disabling all other chip functions until the next interrupt or hardware reset. In Power-save mode, the asynchronous timer continues to run, allowing the user to maintain a timer base while the rest of the device is sleeping. The ADC Noise Reduction mode stops the CPU and all I/O modules except asynchronous timer and ADC to minimize switching noise during ADC conversions. In Standby mode, the crystal/resonator oscillator is running while the rest of the device is sleeping. This allows very fast start-up combined with low power consumption. In Extended Standby mode, both the main oscillator and the asynchronous timer continue to run.

Atmel offers the QTouch® library for embedding capacitive touch buttons, sliders and wheels functionality into AVR microcontrollers. The patented charge-transfer signal acquisition offers robust sensing and includes fully debounced reporting of touch keys and includes Adjacent Key Suppression® (AKS™) technology for unambiguous detection of key events. The easy-to-use QTouch Suite toolchain allows you to explore, develop and debug your own touch applications.

The device is manufactured using Atmel's high density non-volatile memory technology. The On-chip ISP Flash allows the program memory to be reprogrammed In-System through an SPI serial interface, by a conventional nonvolatile memory programmer, or by an On-chip Boot program running on the AVR core. The Boot program can use any interface to download the application program in the Application Flash memory. Software in the Boot Flash section will continue to run while the Application Flash section is updated, providing true Read-While-Write operation. By combining an 8-bit RISC CPU with In-System Self-Programmable Flash on a monolithic chip, the Atmel ATmega328/P is a powerful microcontroller that provides a highly flexible and cost effective solution to many embedded control applications.

The ATmega328/P is supported with a full suite of program and system development tools including: C Compilers, Macro Assemblers, Program Debugger/Simulators, In-Circuit Emulators, and Evaluation kits.

2. Configuration Summary

Features	ATmega328/P
Pin Count	28/32
Flash (Bytes)	32K
SRAM (Bytes)	2K
EEPROM (Bytes)	1K
General Purpose I/O Lines	23
SPI	2
TWI (I ² C)	1
USART	1
ADC	10-bit 15kSPS
ADC Channels	8
8-bit Timer/Counters	2
16-bit Timer/Counters	1

3. Ordering Information

3.1. ATmega328

Speed [MHz] ⁽³⁾	Power Supply [V]	Ordering Code ⁽²⁾	Package ⁽¹⁾	Operational Range
20	1.8 - 5.5	ATmega328-AU ATmega328-AUR ⁽⁵⁾ ATmega328-MMH ⁽⁴⁾ ATmega328-MMHR ⁽⁴⁾⁽⁵⁾ ATmega328-MU ATmega328-MUR ⁽⁵⁾ ATmega328-PU	32A 32A 28M1 28M1 32M1-A 32M1-A 28P3	Industrial (-40°C to 85°C)

Note:

1. This device can also be supplied in wafer form. Please contact your local Atmel sales office for detailed ordering information and minimum quantities.
2. Pb-free packaging, complies to the European Directive for Restriction of Hazardous Substances (RoHS directive). Also Halide free and fully Green.
3. Please refer to *Speed Grades* for Speed vs. V_{CC}
4. Tape & Reel.
5. NiPdAu Lead Finish.

Package Type	
28M1	28-pad, 4 x 4 x 1.0 body, Lead Pitch 0.45mm Quad Flat No-Lead/Micro Lead Frame Package (QFN/MLF)
28P3	28-lead, 0.300" Wide, Plastic Dual Inline Package (PDIP)
32M1-A	32-pad, 5 x 5 x 1.0 body, Lead Pitch 0.50mm Quad Flat No-Lead/Micro Lead Frame Package (QFN/MLF)
32A	32-lead, Thin (1.0mm) Plastic Quad Flat Package (TQFP)

3.2. ATmega328P

Speed [MHz] ⁽³⁾	Power Supply [V]	Ordering Code ⁽²⁾	Package ⁽¹⁾	Operational Range
20	1.8 - 5.5	ATmega328P-AU ATmega328P-AUR ⁽⁵⁾ ATmega328P-MMH ⁽⁴⁾ ATmega328P-MMHR ⁽⁴⁾⁽⁵⁾ ATmega328P-MU ATmega328P-MUR ⁽⁵⁾ ATmega328P-PU	32A 32A 28M1 28M1 32M1-A 32M1-A 28P3	Industrial (-40°C to 85°C)
		ATmega328P-AN ATmega328P-ANR ⁽⁵⁾ ATmega328P-MN ATmega328P-MNR ⁽⁵⁾ ATmega328P-PN	32A 32A 32M1-A 32M1-A 28P3	Industrial (-40°C to 105°C)

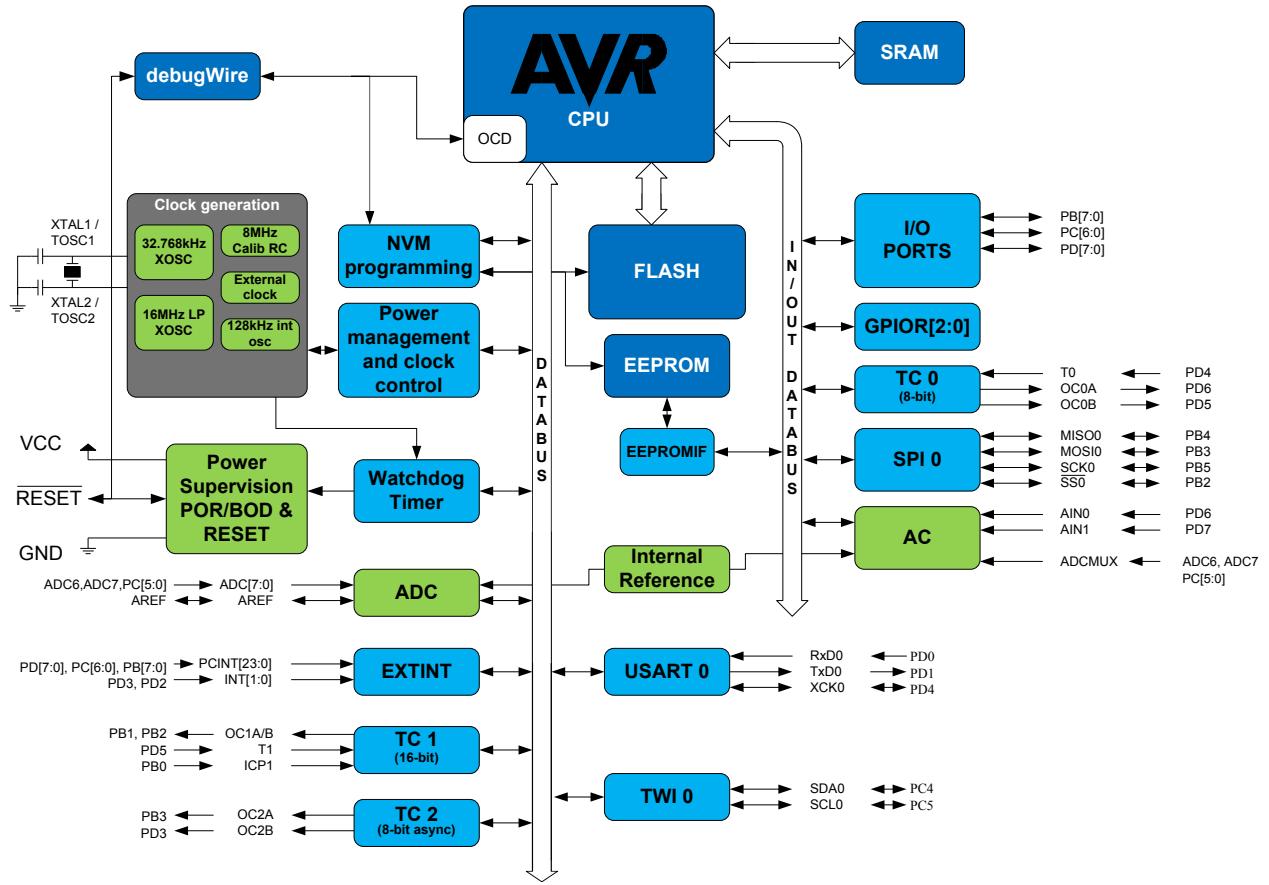
Note:

1. This device can also be supplied in wafer form. Please contact your local Atmel sales office for detailed ordering information and minimum quantities.
2. Pb-free packaging, complies to the European Directive for Restriction of Hazardous Substances (RoHS directive). Also Halide free and fully Green.
3. Please refer to *Speed Grades* for Speed vs. V_{CC}
4. Tape & Reel.
5. NiPdAu Lead Finish.

Package Type	
28M1	28-pad, 4 x 4 x 1.0 body, Lead Pitch 0.45mm Quad Flat No-Lead/Micro Lead Frame Package (QFN/MLF)
28P3	28-lead, 0.300" Wide, Plastic Dual Inline Package (PDIP)
32M1-A	32-pad, 5 x 5 x 1.0 body, Lead Pitch 0.50mm Quad Flat No-Lead/Micro Lead Frame Package (QFN/MLF)
32A	32-lead, Thin (1.0mm) Plastic Quad Flat Package (TQFP)

4. Block Diagram

Figure 4-1. Block Diagram



5. Pin Configurations

5.1. Pin-out

Figure 5-1. 28-pin PDIP

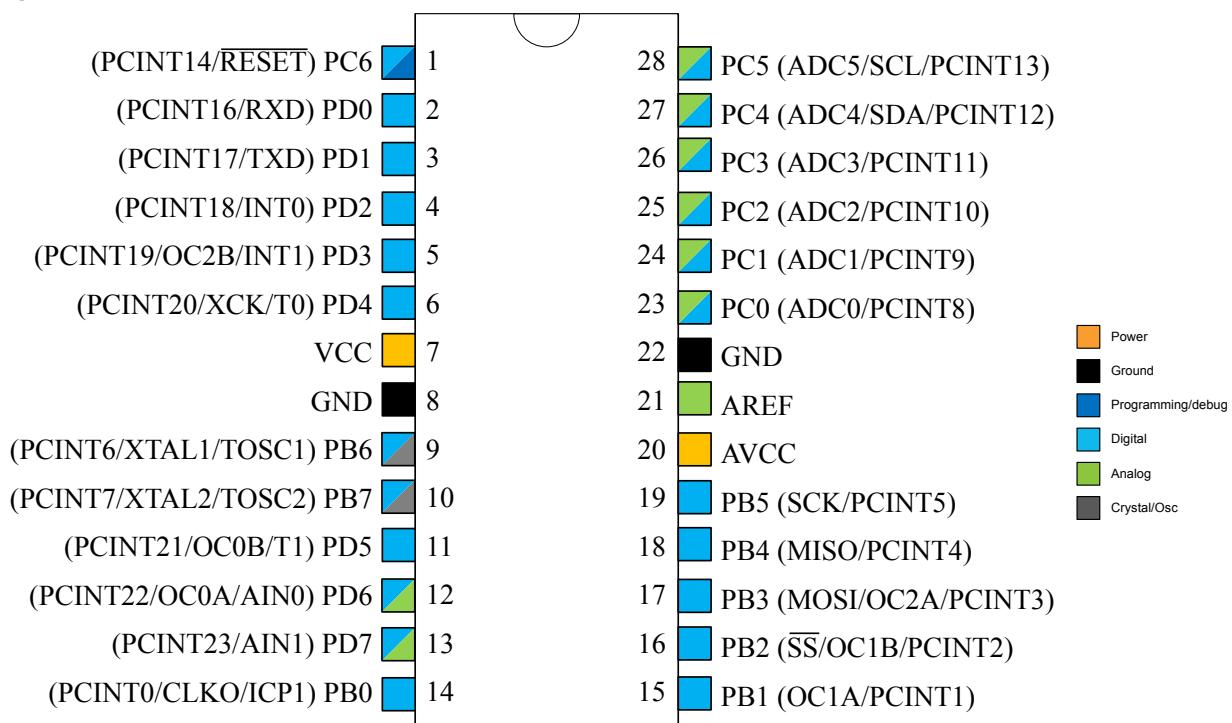


Figure 5-2. 28-pin MLF Top View

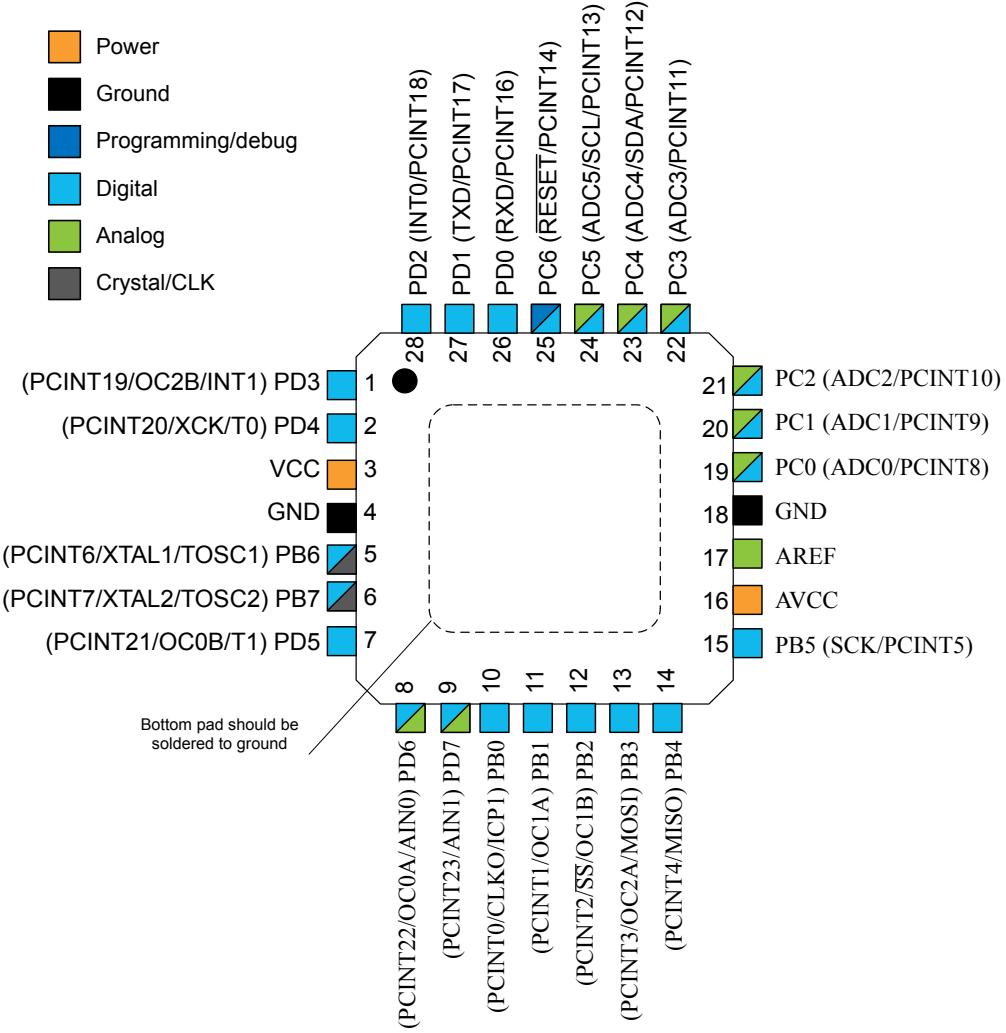


Figure 5-3. 32-pin TQFP Top View

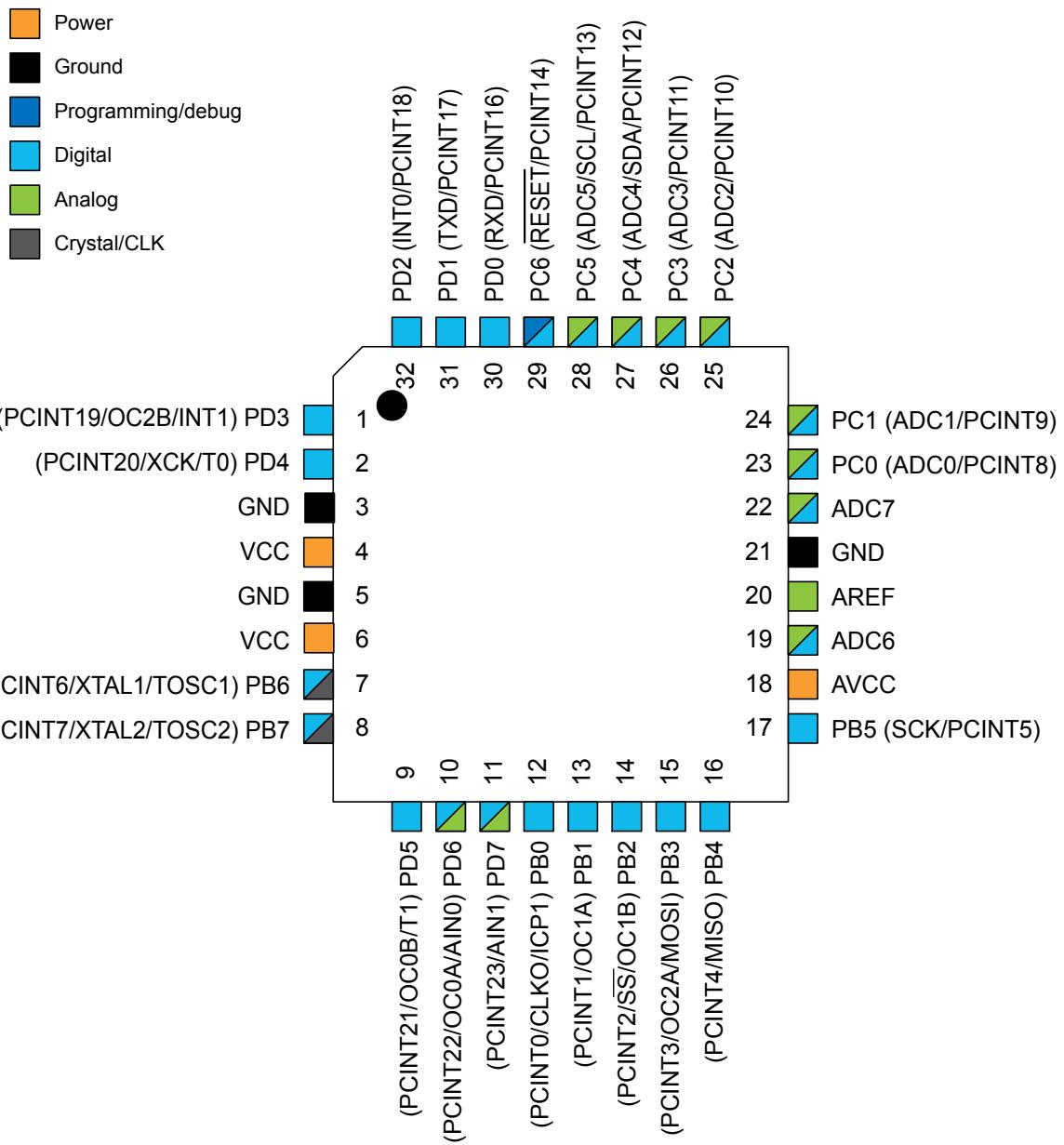
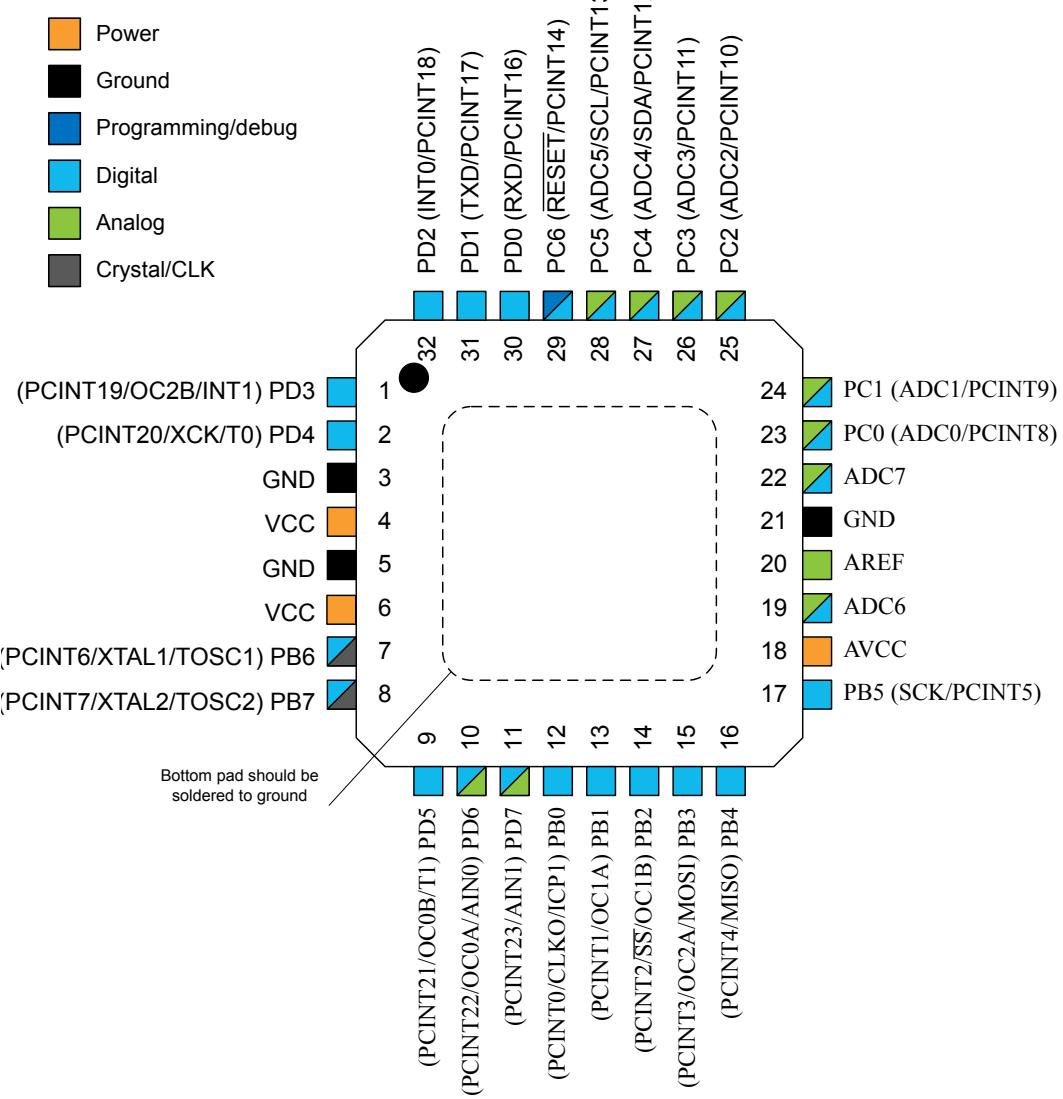


Figure 5-4. 32-pin MLF Top View



5.2. Pin Descriptions

5.2.1. VCC

Digital supply voltage.

5.2.2. GND

Ground.

5.2.3. Port B (PB[7:0]) XTAL1/XTAL2/TOSC1/TOSC2

Port B is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port B output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port B pins that are externally pulled low will source current if the pull-up resistors are activated. The Port B pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Depending on the clock selection fuse settings, PB6 can be used as input to the inverting Oscillator amplifier and input to the internal clock operating circuit.

Depending on the clock selection fuse settings, PB7 can be used as output from the inverting Oscillator amplifier.

If the Internal Calibrated RC Oscillator is used as chip clock source, PB[7:6] is used as TOSC[2:1] input for the Asynchronous Timer/Counter2 if the AS2 bit in ASSR is set.

5.2.4. Port C (PC[5:0])

Port C is a 7-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The PC[5:0] output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port C pins that are externally pulled low will source current if the pull-up resistors are activated. The Port C pins are tri-stated when a reset condition becomes active, even if the clock is not running.

5.2.5. PC6/RESET

If the RSTDISBL Fuse is programmed, PC6 is used as an I/O pin. Note that the electrical characteristics of PC6 differ from those of the other pins of Port C.

If the RSTDISBL Fuse is unprogrammed, PC6 is used as a Reset input. A low level on this pin for longer than the minimum pulse length will generate a Reset, even if the clock is not running. Shorter pulses are not guaranteed to generate a Reset.

The various special features of Port C are elaborated in the *Alternate Functions of Port C* section.

5.2.6. Port D (PD[7:0])

Port D is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port D output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port D pins that are externally pulled low will source current if the pull-up resistors are activated. The Port D pins are tri-stated when a reset condition becomes active, even if the clock is not running.

5.2.7. AV_{CC}

AV_{CC} is the supply voltage pin for the A/D Converter, PC[3:0], and PE[3:2]. It should be externally connected to V_{CC}, even if the ADC is not used. If the ADC is used, it should be connected to V_{CC} through a low-pass filter. Note that PC[6:4] use digital supply voltage, V_{CC}.

5.2.8. AREF

AREF is the analog reference pin for the A/D Converter.

5.2.9. ADC[7:6] (TQFP and VFQFN Package Only)

In the TQFP and VFQFN package, ADC[7:6] serve as analog inputs to the A/D converter. These pins are powered from the analog supply and serve as 10-bit ADC channels.

Guangzhou HC Information Technology Co., Ltd.

Product Data Sheet

Module Data Sheet

Rev 1

1. 0	2.0	2.1	2.2				
2006/6/18	2006/9/6	2010/4/22	2011/4/6				

DRAWN BY :	Ling Xin		MODEL : HC-06
CHECKED BY :	Eric Huang		Description:: BC04 has external 8M Flash and EDR module HC-06 is industrial, and compatible with civil HC-04
APPD. BY:	Simon Mok		REV: 2.0
Former version introduction	HC-06 is the higher version of LV_BC_2.0. Linvor is the former of wavesen.		Page :

Contents

1. Product's picture
2. Feature
3. Pins description
4. The parameters and mode of product
5. Block diagram
6. Debugging device
7. Characteristic of test
8. Test diagram
9. AT command set

1. Product's picture

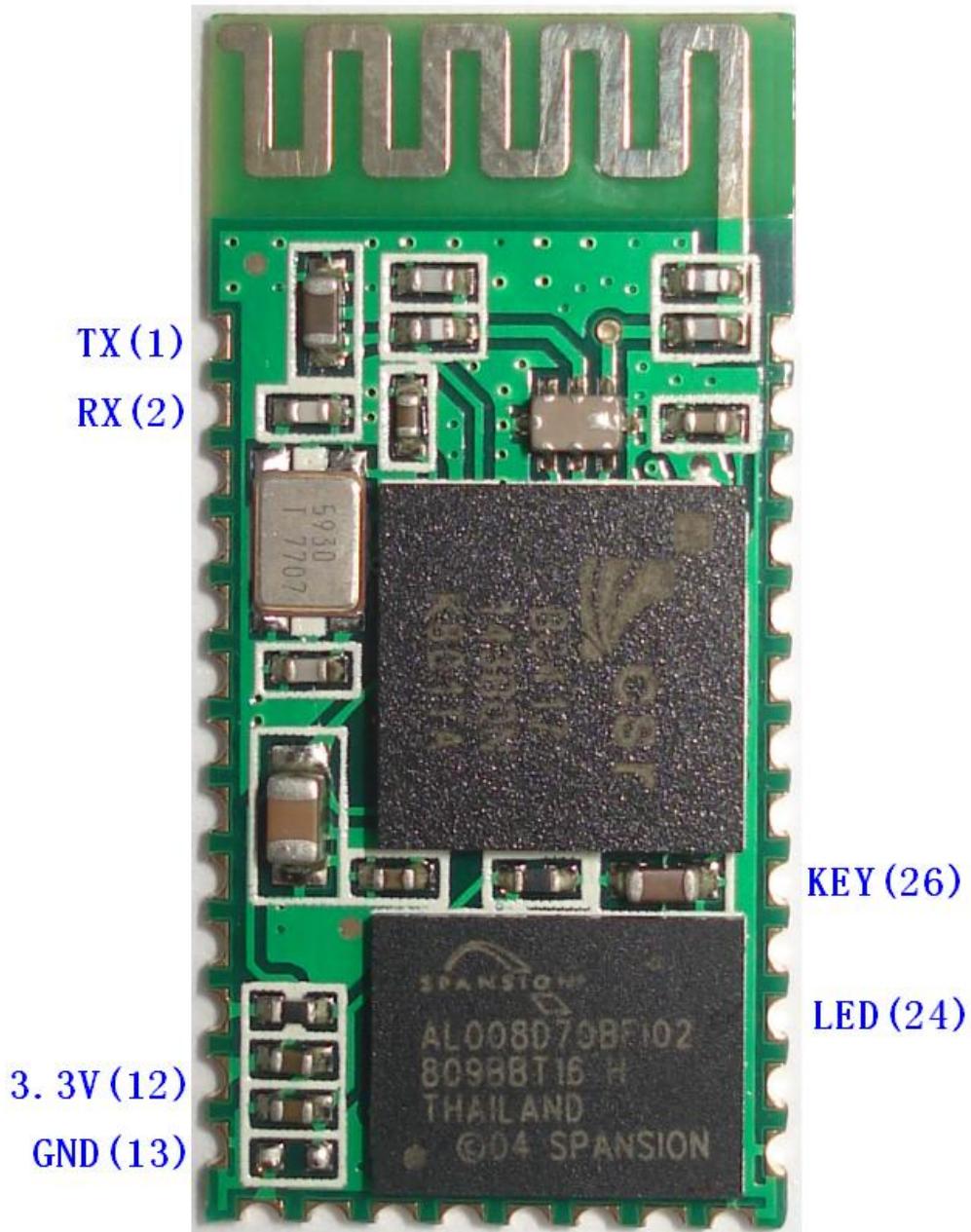


Figure 1 A Bluetooth module

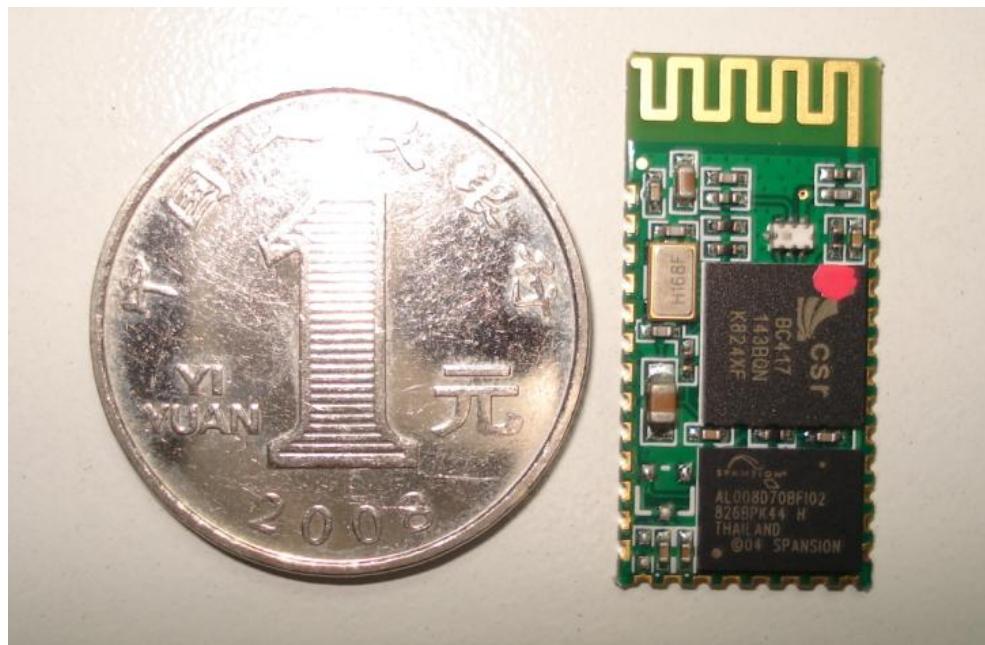


Figure 2. A Bluetooth module size



Figure 3 50 pieces chips in an anti-static blister package.

2. Feature

- Wireless transceiver
 - Sensitivity (Bit error rate) can reach -80dBm.
 - The change range of output's power: -4 - +6dBm.
- Function description (perfect Bluetooth solution)
 - Has an EDR module; and the change range of modulation depth: 2Mbps - 3Mbps.
 - Has a build-in 2.4GHz antenna; user needn't test antenna.
 - Has the external 8Mbit FLASH
 - Can work at the low voltage (3.1V~4.2V). The current in pairing is in the range of 30~40mA.
The current in communication is 8mA.
 - Standard HCI Port (UART or USB)
 - USB Protocol: Full Speed USB1.1, Compliant With 2.0
 - This module can be used in the SMD.
 - It's made through RoHS process.
 - The board PIN is half hole size.
 - Has a 2.4GHz digital wireless transceiver.
 - Bases at CSR BC04 Bluetooth technology.
 - Has the function of adaptive frequency hopping.
 - Small (27mm×13mm×2mm)
 - Peripherals circuit is simple.
 - It's at the Bluetooth class 2 power level.
 - Storage temperature range: -40 °C - 85°C , work temperature range: -25 °C - +75°C
 - Any wave inter Interference: 2.4MHz, the power of emitting: 3 dBm.
 - Bit error rate: 0. Only the signal decays at the transmission link, bit error may be produced. For example, when RS232 or TTL is being processed, some signals may decay.
- Low power consumption
- Has high-performance wireless transceiver system
- Low Cost

- Application fields:
 - Bluetooth Car Handsfree Device
 - Bluetooth GPS
 - Bluetooth PCMCIA , USB Dongle
 - Bluetooth Data Transfer
- Software
 - CSR

3. PINs description

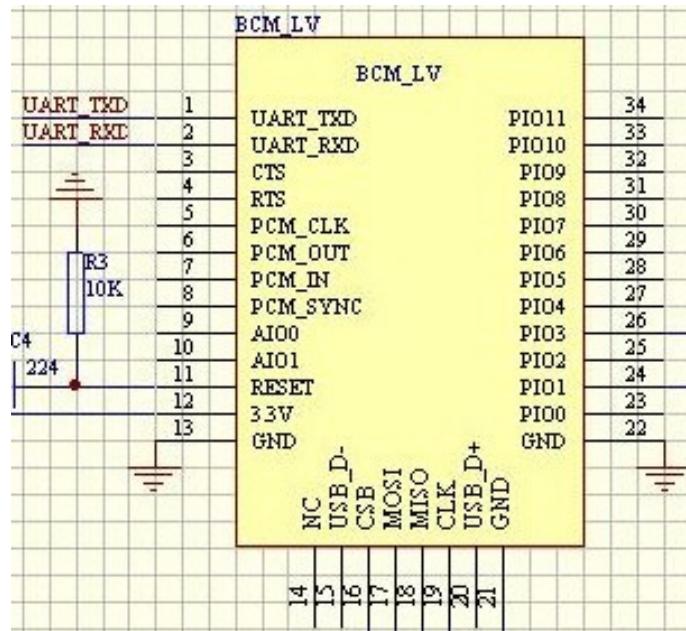


Figure 3 PIN configuration

The PINs at this block diagram is as same as the physical one.

PIN Name	PIN #	Pad type	Description	Note
GND	13 21 22	VSS	Ground pot	
1V8	14	VDD	Integrated 1.8V (+) supply with On-chip linear regulator output within 1.7-1.9V	
VCC	12	3.3V		
AIO0	9	Bi-Directional	Programmable input/output line	
AIO1	10	Bi-Directional	Programmable input/output line	

PIO0	23	Bi-Directional RX EN	Programmable input/output line, control output for LNA(if fitted)	
PIO1	24	Bi-Directional TX EN	Programmable input/output line, control output for PA(if fitted)	
PIO2	25	Bi-Directional	Programmable input/output line	
PIO3	26	Bi-Directional	Programmable input/output line	
PIO4	27	Bi-Directional	Programmable input/output line	
PIO5	28	Bi-Directional	Programmable input/output line	
PIO6	29	Bi-Directional	Programmable input/output line	CLK_REQ
PIO7	30	Bi-Directional	Programmable input/output line	CLK_OUT
PIO8	31	Bi-Directional	Programmable input/output line	
PIO9	32	Bi-Directional	Programmable input/output line	
PIO10	33	Bi-Directional	Programmable input/output line	
PIO11	34	Bi-Directional	Programmable input/output line	
RESETB	11	CMOS Input with weak internal pull-down		
UART_RTS	4	CMOS output, tri-stable with weak internal pull-up	UART request to send, active low	
UART_CTS	3	CMOS input with weak internal pull-down	UART clear to send, active low	
UART_RX	2	CMOS input with weak internal pull-down	UART Data input	
UART_TX	1	CMOS output, Tri-stable with weak internal pull-up	UART Data output	
SPI_MOSI	17	CMOS input with weak internal pull-down	Serial peripheral interface data input	
SPI_CS _B	16	CMOS input with weak internal	Chip select for serial peripheral interface, active low	

		pull-up		
SPI_CLK	19	CMOS input with weak internal pull-down	Serial peripheral interface clock	
SPI_MISO	18	CMOS input with weak internal pull-down	Serial peripheral interface data Output	
USB_-	15	Bi-Directional		
USB_+	20	Bi-Directional		
1.8V	14		1.8V external power supply input	Default : 1.8V internal power supply.
PCM_CLK	5	Bi-Directional		
PCM_OUT	6	CMOS output		
PCM_IN	7	CMOS Input		
PCM_SYNC	8	Bi-Directional		

4. The parameters and mode of product

LINVOR BLUE T

www.linvor.com

Bluetooth Module
Bluetooth

CSR,BC417143B

V 2.0

2006/09/6

蓝牙 RF 模块

1. 采用 CSR BC4 +8M FLASH 方案
2. 具有 PIO0-PIO11、AIO0、AIO1、
USB、PCM、UART 及 SPI 接口，
模块内置 8MFLASH，功能强大，
用户可定制软件,适用于各种蓝牙
设备，内置 RF 天线,便于调试。

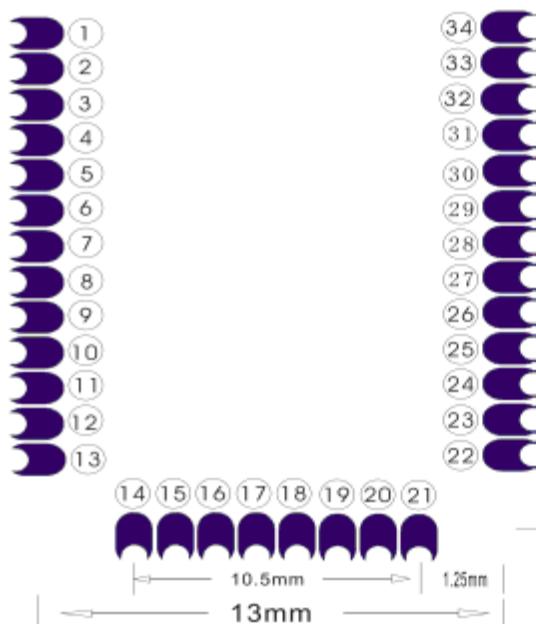
蓝牙协议版本	Bluetooth Specification V2.0 With EDR
USB 协议 USB Protocol	Full Speed USB V1.1 Compliant With USB V2.0
频率	2.4Ghz ISM band
调制方式	GFSK(Gaussian Frequency Shift Keying)
发射功率	-4 ~+4 dBm, Class 2
灵敏度	≥ -80dBm at 0. 1% BER
通讯速率	Asynchronous:2Mbps(Max)
供电电源	3.3V
工作温度	-20~+55 Centigrade
封装尺寸	27mmX13mmX2mm

LINVOR BLUE T

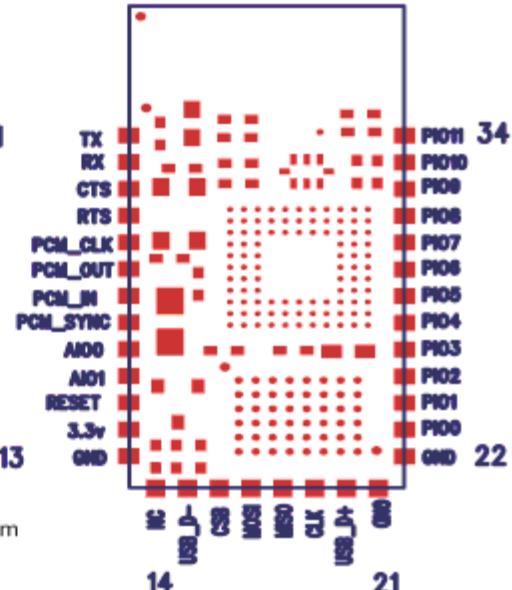
www.linvor.com

LV-BC-2.0

单位：mm



1
18mm
13
1.75mm
14 21
10.5mm 1.25mm
13mm



NO	PIN NAME	NO	PIN NAME
1	TX	20	USB D+
2	RX	21	GND
3	CTS	22	GND
4	RTS	23	PI00
5	PCM CLK	24	PI01
6	PCM OUT	25	PI02
7	PCM IN	26	PI03
8	PCM SYNC	27	PI04
9	AIO0	28	PI05
10	AIO1	29	PI06
11	RESET	30	PI07
12	3.3V	31	PI08
13	GND	32	PI09
14	NC	33	PI010
15	USB D-	34	PI011
16	CSB		
17	MOSI		
18	MISO		
19	CLK		

PCB Layout 请参考实物

5. Block diagram

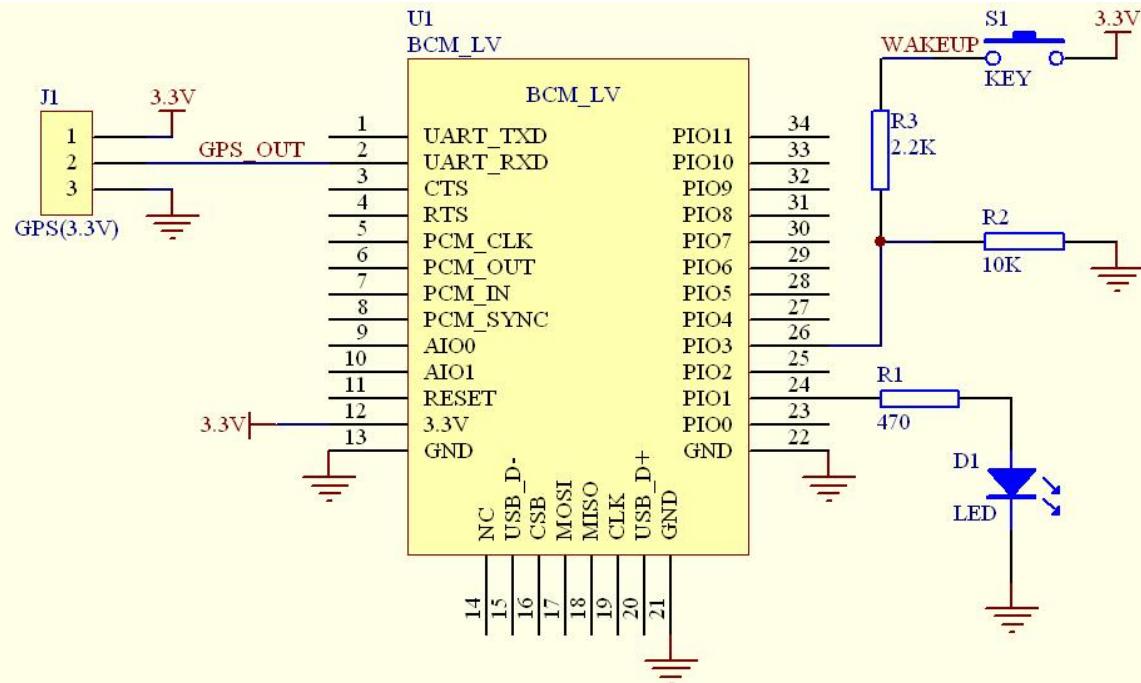
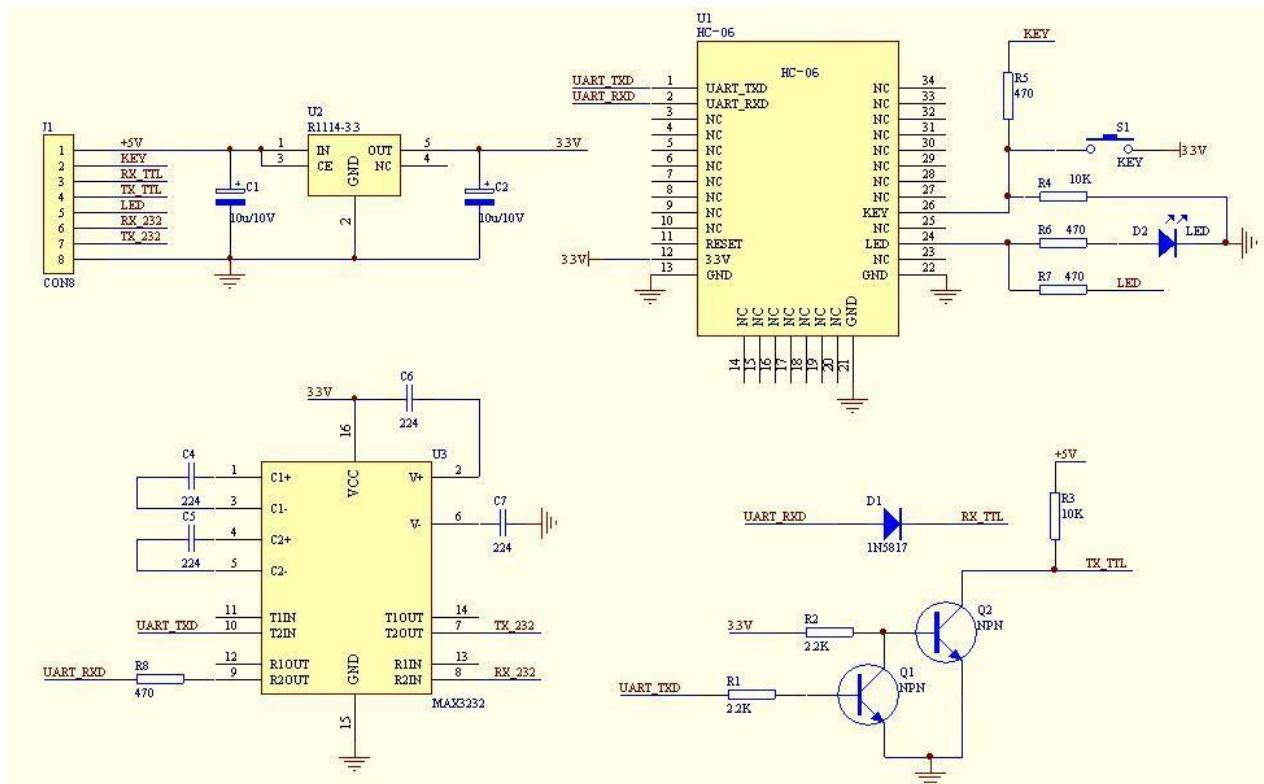


Figure 5 Block diagram 1



HC-04/06 master device has a function of remembering the last paired slave device. As a master device, it will search the last paired slave device until the connection is built. But if the WAKEUP bottom is pressed, HC-04/06 will lose the memory and research the new slave device.

6. Debugging device

6.1 Device

PC, hardware, 3G, 3G Frequency Counter (SP3386), 3.15V DC power supply, Shielding, Bluetooth Test box.

6.2 Software

7. Characteristic of test

	Test Condition 25°C RH 65%			
	Min	Typ	Max	Unit
1. Carrier Freq. (<i>ISM Band</i>)	2.4		2.4835	MHz
2. RF O/P Power	-6	2	4	dBm
3. Step size of Power control	2		8	dB
4. Freq. Offset (<i>Typical Carrier freq.</i>)	-75		75	KHz
5. Carrier Freq. drift (<i>Hopping on, drift rate/50uS</i>)	-20		20	KHz
1 slot packet	-25		25	KHz
3 slot packet	-40		-40	KHz
6. Average Freq. Deviations (<i>Hopping off, modulation</i>)	140		175	KHz
Freq. Deviation	115			KHz
Ratio of Freq. Deviation	0.8			
7. Receive Sensitivity @< 0.1% BER(<i>Bit error rate</i>)	-83			dBm

8. Test diagram

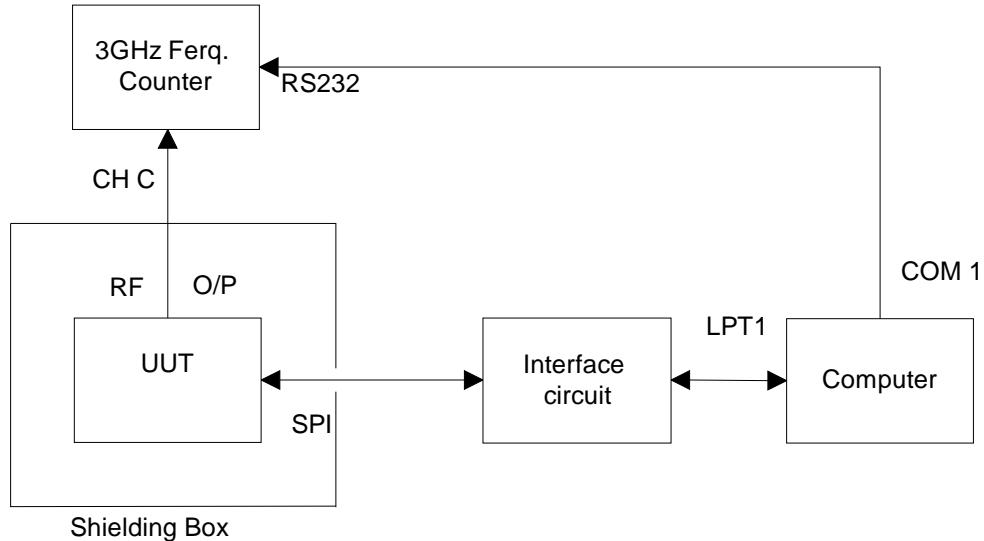


Fig 1. Programming and Freq. Alignment

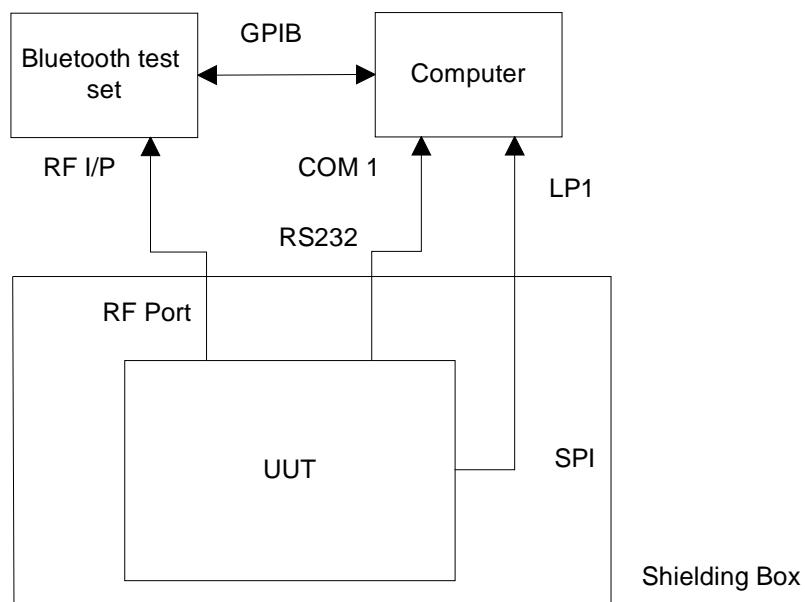


Fig 2 RF parameter Test Procedure

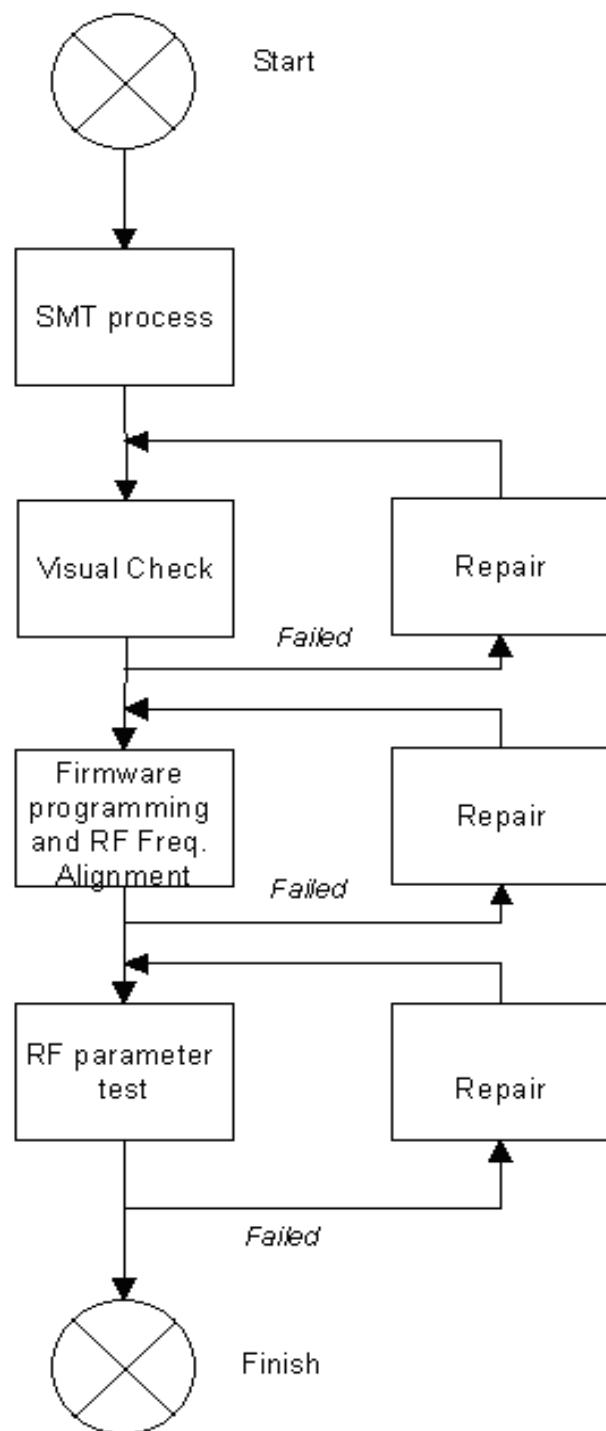


Fig 3 Assemble/Alignment/Testing Flow Chart

9. AT command set

The way to the AT command mode: supply power to the module, it will enter to the AT mode if it needn't pair. The interval of command is about 1 second.

Default parameter: Baud rate:9600N81, ID: linvor, Password:1234

1. Test communication

Send: AT (please send it every second)

Back: OK

2. Reset the Bluetooth serial baud rate

Send: AT+BAUD1

Back: OK1200

Send: AT+BAUD2

Back: OK2400

.....

1-----1200

2-----2400

3-----4800

4-----9600 (Default)

5-----19200

6-----38400

7-----57600

8-----115200

9-----230400

A-----460800

B-----921600

C-----1382400

PC can't support the baud rate lager than 115200. The solution is: make the MCU have higher baud rate (lager than 115200) through programming, and reset the baud rate to low level through the AT command.

The baud rate reset by the AT command can be kept for the next time even though the power is cut off.

3. Reset the Bluetooth name

Send: AT+NAMEname

Back: OKname

Parameter name: Name needed to be set (20 characters limited)

Example:

Send: AT+NAMEbill_gates

Back: OKname

Now, the Bluetooth name is reset to be “bill_gates”

The parameter can be kept even though the power is cut off. User can see the new Bluetooth name in PDA refresh service. (Note: The name is limited in 20 characters.)

4. change the Bluetooth pair password

Send: AT+PINxxxx

Back:OKsetpin

Parameter xxxx: The pair password needed to be set, is a 4-bits number. This command can be used in the master and slave module. At some occasions, the master module may be asked to enter the password when the master module tries to connect the slave module (adapter or cell-phone). Only if the password is entered, the successful connection can be built. At the other occasions, the pair can be finish automatically if the master module can search the proper slave module and the password is correct. Besides the paired slave module, the master can connect the other devices who have slave module, such as Bluetooth digital camera, Bluetooth GPS, Bluetooth serial printer etc.

Example:

Send: AT+PIN8888

Back: OKsetpin

Then the password is changed to be 8888, while the default is 1234.

This parameter can be kept even though the power is cut off.

5. No parity check (The version, higher than V1.5, can use this command)

Send: AT+PN (This is the default value)

Back: OK NONE

6. Set odd parity check (The version, higher than V1.5, can use this command)

Send: AT+PO

Back: OK ODD

7. Set even parity check(The version, higher than V1.5, can use this command)

Send: AT+PE

Back: OK EVEN

8. Get the AT version

Send: AT+VERSION

Back: LinvorV1.n



Capacitive Soil Moisture Sensor SKU:SEN0193



Capacitive Soil Moisture Sensor

Contents

- [1 Introduction](#)
- [2 Specification](#)
- [3 Tutorial](#)
 - [3.1 Requirements](#)
 - [3.2 Connection Diagram](#)
 - [3.3 Calibration Code](#)
 - [3.4 Calibration](#)
 - [3.4.1 Calibration Range](#)
 - [3.4.2 Section Settings](#)
 - [3.5 Test Code](#)
- [4 FAQ](#)

Introduction

Our soil moisture sensor measures soil moisture levels by capacitive sensing rather than resistive sensing like other sensors on the market. It is made of corrosion resistant material which gives it an excellent service life.

Insert it in to the soil around your plants and impress your friends with real-time soil moisture data! This module includes an on-board voltage regulator which gives it an operating voltage range of 3.3 ~ 5.5V. It is perfect for low-voltage MCUs, both 3.3V and 5V. For compatibility with a Raspberry Pi it will need an ADC converter.

This sensor is compatible with our 3-pin "Gravity" interface, which can be directly connected to the Gravity I/O expansion shield.

Specification

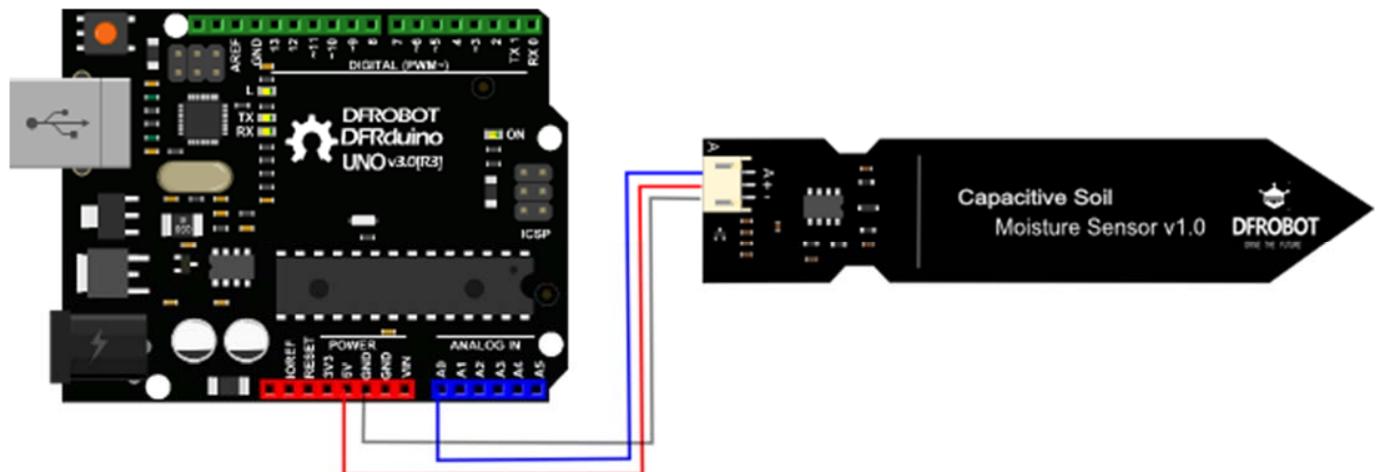
- Operating Voltage: 3.3 ~ 5.5 VDC
- Output Voltage: 0 ~ 3.0VDC
- Operating Current: 5mA
- Interface: PH2.0-3P
- Dimensions: 3.86 x 0.905 inches (L x W)
- Weight: 15g

Tutorial

Requirements

- **Hardware**
UNO x1
Capacitive Soil Moisture Sensor x1
Jumper Cable x3
- **Software**
Arduino IDE V1.6.5 [Click to Download Arduino IDE](#)

Connection Diagram

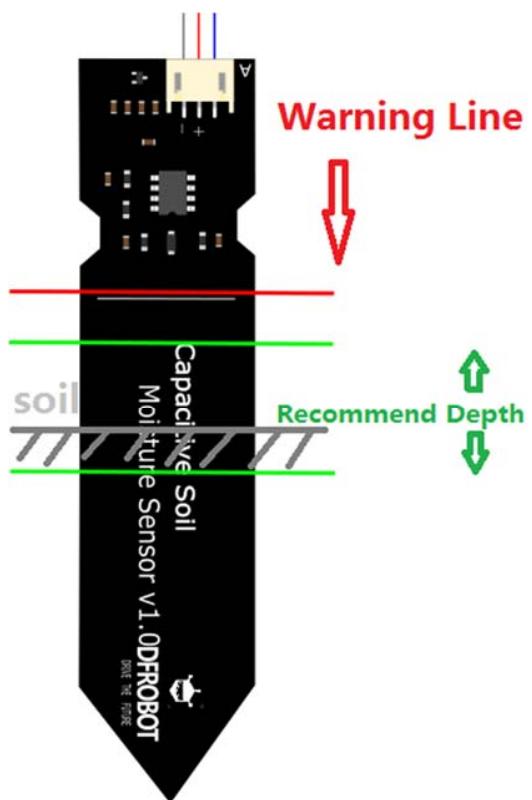


Calibration Code

```
void setup() {  
    Serial.begin(9600); // open serial port, set the baud rate as 9600 bps  
}  
void loop() {  
    int val;  
    val = analogRead(0); //connect sensor to Analog 0  
    Serial.print(val); //print the value to serial port  
    delay(100);  
}
```

Calibration

Calibration Range



1. Open the serial port monitor and set the baud rate to 9600
2. Record the sensor value when the probe is exposed to the air as "Value 1". This is the boundary value of dry soil "Humidity: 0%RH"
3. Take a cup of water and insert the probe into it no further than the red line in the diagram
4. Record the sensor value when the probe is exposed to the water as "Value 2". This is the boundary value of moist soil "Humidity: 100%RH"



The components on this board are NOT waterproof, do not expose to moisture further than the red line. (If you want to protect components from the elements, try using a length of wide heat shrink tubing around the upper-section of the board.)
There is an inverse ratio between the sensor output value and soil moisture.

Section Settings

The final output value is affected by probe insertion depth and how tight the soil packed around it is. We regard "value_1" as dry soil and "value_2" as soaked soil. This is the sensor detection range.

For example: Value_1 = 520; Value_2 = 260.

The range will be divided into three sections: dry, wet, water. Their related values are:

- Dry: (520 430]
- Wet: (430 350]
- Water: (350 260]

Test Code

```
*****
This example reads Capacitive Soil Moisture Sensor.

Created 2015-10-21
By berinie Chen <bernie.chen@dfrobot.com>

GNU Lesser General Public License.
See <http://www.gnu.org/licenses/> for details.
All above must be included in any redistribution
***** /
```

```
*****Notice and Trouble shooting*****
1.Connection and Diagram can be found here: https://www.dfrobot.com/wiki/index.php?title=Capacitive_Soil_Moisture_Sensor_SKU:SEN0193
2.This code is tested on Arduino Uno.
3.Sensor is connect to Analog 0 port.
*****/



const int AirValue = 520;    //you need to replace this value with Value_1
const int WaterValue = 260;  //you need to replace this value with Value_2
int intervals = (AirValue - WaterValue)/3;
int soilMoistureValue = 0;
void setup() {
    Serial.begin(9600); // open serial port, set the baud rate to 9600 bps
}
void loop() {
soilMoistureValue = analogRead(A0); //put Sensor insert into soil
if(soilMoistureValue > WaterValue && soilMoistureValue < (WaterValue + intervals))
{
    Serial.println("Very Wet");
}
else if(soilMoistureValue > (WaterValue + intervals) && soilMoistureValue < (AirValue - intervals))
{
    Serial.println("Wet");
}
else if(soilMoistureValue < AirValue && soilMoistureValue > (AirValue - intervals))
{
    Serial.println("Dry");
}
delay(100);
}
```