wemos, esp8266, riego, autónomo

Riego Idependiente

La idea de este proyecto es crear un sistema de riego automático que funcione independiente de una toma de agua. Para ello utilizaremos un motor sumergible que meteremos dentro de una garrafa y un wemos con display oled para controlar todo el sistema.

Materiales:

- Wemos D1 mini
- Display oled 0.96
- Sensor de humedad
- circuito para cargar batería Lipo (TP 4506)
- batería Lipo

Circuito

Las equivalencias con los GPIOs lo pueden encontrar e http://escapequotes.net/wemos-d1mini-arduinoide/ Necesitaremos usar un transistor bipolar o un mosfet ya que la corriente máxima que da arduino por pin es de 40mA. Insuficiente para poner en marcha el motor.

Si forzamos la corriente o voltaje máximo que soporta nuestro Arduino es muy probable que lo acabemos quemando

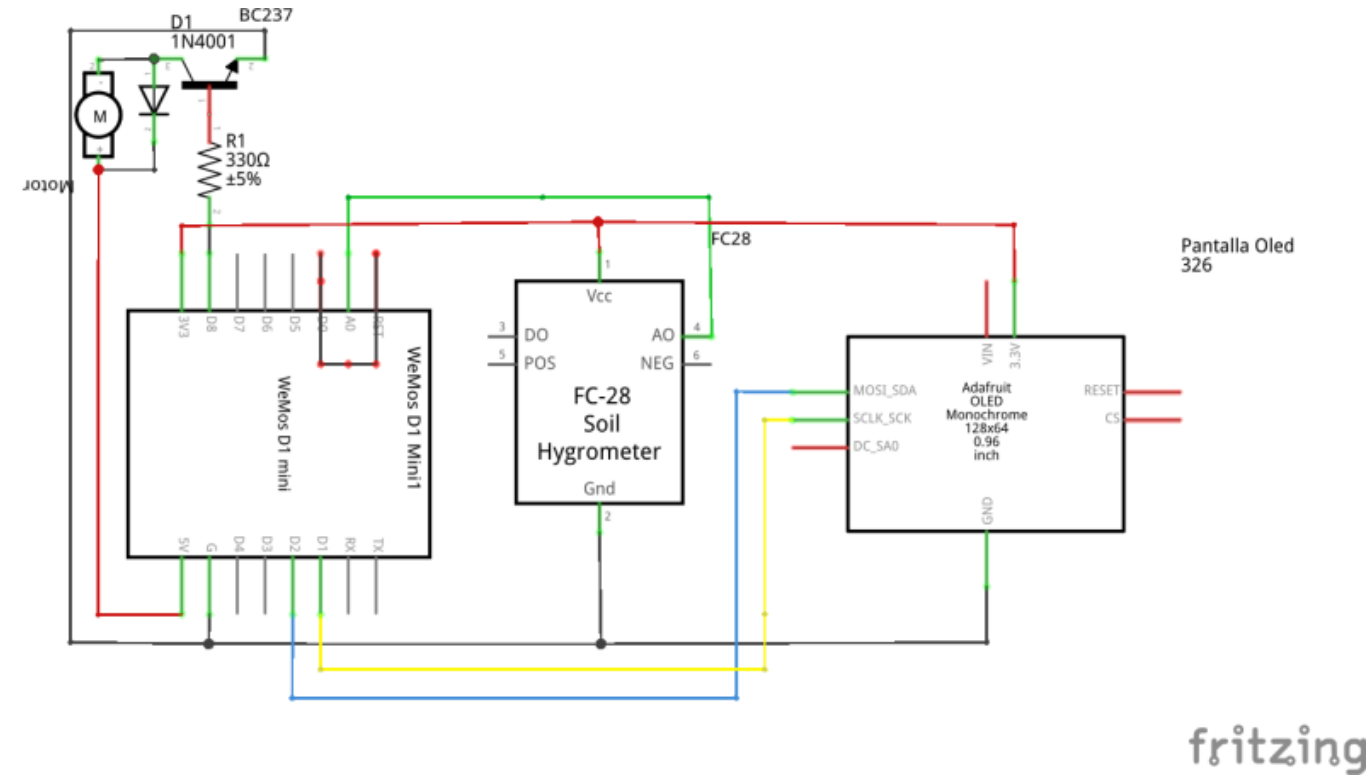
Yo he utilizado un transistor BCP337 y un motor con una resistencia de unos 10 ohm . Utilizando la calculadora (https://www.luisllamas.es/calculadora-de-transitor-bjt-como-interruptor/) me sale una resistencia de base de unos 330 ohm.

Independiente de la resistencia de base necesitamos también poner un diodo en paralelo con el motor. A dicho diodo se le denomina diodo de flyback, ya que proporciona un camino de baja resistencia que permite disipar las corrientes inducidas por las cargas inductivas, protegiendo el resto de dispositivos.

Pantalla Oled → Vdd a +3V, GND al menos , el pin SCL al GPIO5 (D1) y SDA al GPIO4 (D2) del Wemos **FC28** → Vcc a +3V, GND, A0 a la patilla A0 del Wemos

Para habilitar el modo Deep-sleep, necesitamos conectar un cable entre el pin RST y el al pin GPIO 16 (D0 en el Wemos)

El circuito queda de la siguiente forma:



En el esquema no aparece el módulo de carga tp4506 ya que simplemente es conectar los terminales OUT+ y OUT- del módulo TP4506 a nuestra placa y conectar la batería a los terminales B+ y B- del TP4506

Programa

Lo primero debemos descargar e instalar las librerías para arduino del display Oled → Adafruit_GFX y Adafruit_SSD1306

```
/------------ Programa para  sistema de riego portátil con wemos
//-------------por LC de intrusos.info
//-------------wiki.intrusos.info

#include <SPI.h>
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>

//----------------Definimos a que patilla esta conectado el FC-28
byte pinSensorHumedad = A0;

//--------------- Definimos el pin donde está el motor
byte pinMotor = 15; //GPIO15 que es el D8 en el ESP8266

//---------------------------- Pantalla OLED
// SCL GPIO5
```

```
// SDA GPIO4
#define OLED_RESET 0  // GPIO0
Adafruit_SSD1306 display(OLED_RESET);

#define NUMFLAKES 10
#define XPOS 0
#define YPOS 1
#define DELTAY 2

#define LOGO16_GLCD_HEIGHT 16
#define LOGO16_GLCD_WIDTH  16


//-------------- Constantes para los posibles modos de funcionamiento
byte MODO_SOLO_HUMEDAD = 0;


int modoElegido = 0;        // Modo de funcionamiento

int umbralHumedad = 0;      // Umbral de humedad seleccionado para empezar a
regar

int duracionRiego = 0;      // Duración del riego en segundos una vez
alcanzado el evento de activación

String linea1;          // Contenido para la línea superior del display
String linea2;          // Contenido para la línea inferior del display

int humedadMinima = 0;      // Lectura mínima por defecto para el sensor de
humedad (se ajusta dinámicamente)
int humedadMaxima = 100;    // Lectura máxima por defecto para el sensor de
humedad (se ajusta dinámicamente)

int lecturasHumedad[10];    // Ultimas 10 lecturas del sensor para hacer la
media
int indiceLecturasHumedad = 0;  // Indice para saber que valor toca rellenar
del array previo
boolean mediaLista = false; // Indicador de que ya están rellenos los 10
valores del array
int mediaHumedad = 0;       // Media de las últimas 10 lecturas de humedad
int riegos = 0;                 // Numero de riegos realizados
int limiteRiegos = 10;          // Limite de seguridad del número de riegos
const int dormir = 10;       //tiempo de reposo (deep-sleep)

//----------------------------------Inicio Setup ---------------------------
-----------------------------
void setup()   {
  Serial.begin(9600);

  // Establece a modo salida el pin para controlar el motor
  pinMode(pinMotor, OUTPUT);
```

```
  //-----------Umbral de Humedad ----------------
  umbralHumedad = 70;

  //------------Duración Riego ------------------------
  duracionRiego = 10;

  //---------------------------- Inicializa el display  --------------------
-----
  display.begin(SSD1306_SWITCHCAPVCC, 0x3C);  // initialize with the I2C
addr 0x3C (for the 64x48)
  // init done

  display.display();
  delay(2000);

  // ------------------------Limpiar el buffer. -------------------------
  display.clearDisplay();

  // ------------------------Muestra el tiempo entre riegos
  linea1 = "Humedad: " + str(mediaHumedad) + "\x25" + " [" +
str(umbralHumedad) + "]";  // \x25 es el símbolo ascii de %
  linea2 = "Riegos: " + str(riegos);
  mostrarTexto();
  delay(1000);

}
void loop() {


  // ----------------------------------Continua el bucle hasta que el tiempo
llegue a cero
  while (true)
  {
    delay(1000);

    // ---------------------------- Actualiza la variable con la humedad
actual
    leerHumedad();

    // ---------------------------------Actualiza el display con el tiempo
hasta el siguiente riego y la humedad actual
    linea1 = "Humedad:" + str(mediaHumedad) + "\x25" + " [" +
str(umbralHumedad) + "]";
    linea2 = "Riegos: " + str(riegos);
    mostrarTexto();

// Si la media de humedad de las últimas 10 lecturas está lista y es
inferior al umbral configurado, activa el riego
    if ((mediaHumedad < umbralHumedad) && mediaLista)
    {
      regar();
```

```
      // Reinicia la media de humedad para que le tiempo a la tierra a
empaparse
      indiceLecturasHumedad = 0;
      mediaLista = false;
    }
// Si la media de humedad de las últimas 10 lecturas está lista y es
superior al umbral configurado, a dormir
    if ((mediaHumedad > umbralHumedad) && mediaLista)
    {
      reposo();
    }
  }

}


// ------------------------------------- Muestra el texto configurado en el
display ---------------------
void mostrarTexto()
{
  display.clearDisplay();
  display.setTextSize(1);
  display.setTextColor(WHITE);
  display.setCursor(0, 0);
  display.println(linea1);
  display.setCursor(0, 20);
  display.println(linea2);
  display.display();
  delay(2000);



}
// ----------------------------Actualiza la variable con la media de humedad
de las últimas 10 lecturas -------------------------
void leerHumedad()
{
  lecturasHumedad[indiceLecturasHumedad] = analogRead(pinSensorHumedad);

  // Serial.println (lecturasHumedad[indiceLecturasHumedad]); para saber el
valor que da el sensor de humedad

  lecturasHumedad[indiceLecturasHumedad] = map
(lecturasHumedad[indiceLecturasHumedad], 140, 395, 100, 0); // Mapeamos el
valor del sensor de 0 a 100
  //los valores 140 y 395 son los que me daba mi sensor ->sumergido en agua
140 y al aire 395
  indiceLecturasHumedad++;
  if (indiceLecturasHumedad > 9)
  {
    indiceLecturasHumedad = 0;
    mediaLista = true;
```

```
  }

  mediaHumedad = 0;
  for (int i = 0; i < 10; i++) mediaHumedad += lecturasHumedad[i];
  mediaHumedad /= 10;

  Serial.println (mediaHumedad);

  if (mediaHumedad > humedadMaxima) humedadMaxima = mediaHumedad;
  if (mediaHumedad < humedadMinima) humedadMinima = mediaHumedad;
  mediaHumedad -= humedadMinima;

  mediaHumedad = (double)((double)mediaHumedad / (double)(humedadMaxima -
humedadMinima)) * 100;

}

// ----------------------------------------Devuelve una cadena numérica de
al menos 2 caractes, rellenando con un cero por la izquierda si hace falta

String str(int valor)
{
  if (valor < 10) return "0" + String(valor);
  else return (String(valor));
}


// ---------------------------Activa lo bomba de riego durante el tiempo
configurado
void regar()
{

  int riegoPendiente = duracionRiego;
  digitalWrite(pinMotor, HIGH);
  while (riegoPendiente > 0)
  {
    linea1 = " -- REGANDO -- ";
    linea2 = "Restante: " + str(riegoPendiente);
    mostrarTexto();
    delay(990);
    riegoPendiente--;
  }
  digitalWrite(pinMotor, LOW);
  riegos++;
}

// ----------------- Entra en modo de ahorro energía -------------
void reposo()
{
  // a Dormir un rato (ojo se resetea y pierde el valor de las variables)
  Serial.println("ESP8266 in sleep mode");
```

```
  ESP.deepSleep(dormir * 1000000);
}
```

Referencias

- https://github.com/adafruit/Adafruit_SSD1306
- https://github.com/klarsys/esp8266-OLED
- http://www.prometec.net/transistores/
- https://www.luisllamas.es/salidas-mayor-potencia-arduino-transistor-bjt/
- https://www.luisllamas.es/arduino-transistor-mosfet/
- http://www.addicore.com/TP4056-Charger-and-Protection-Module-p/ad310.htm