arduino, electrovávulas, riego, reles

5. Electroválvulas

En nuestro anterior proyecto teníamos un pequeño motor para regar independientemente de una toma de agua. Ahora lo vamos a modificar para conectarlo a una o varias tomas de agua y que sean unas electroválvulas las que al abrir o cerrar, dejen pasar el agua de la toma para regar nuestro huerto.

El esquema es idéntico al anterior, sólo necesitamos sustituir el motor por una placa de relés, que nos permita encender y apagar las electroválvulas que funcionan a 12 V .

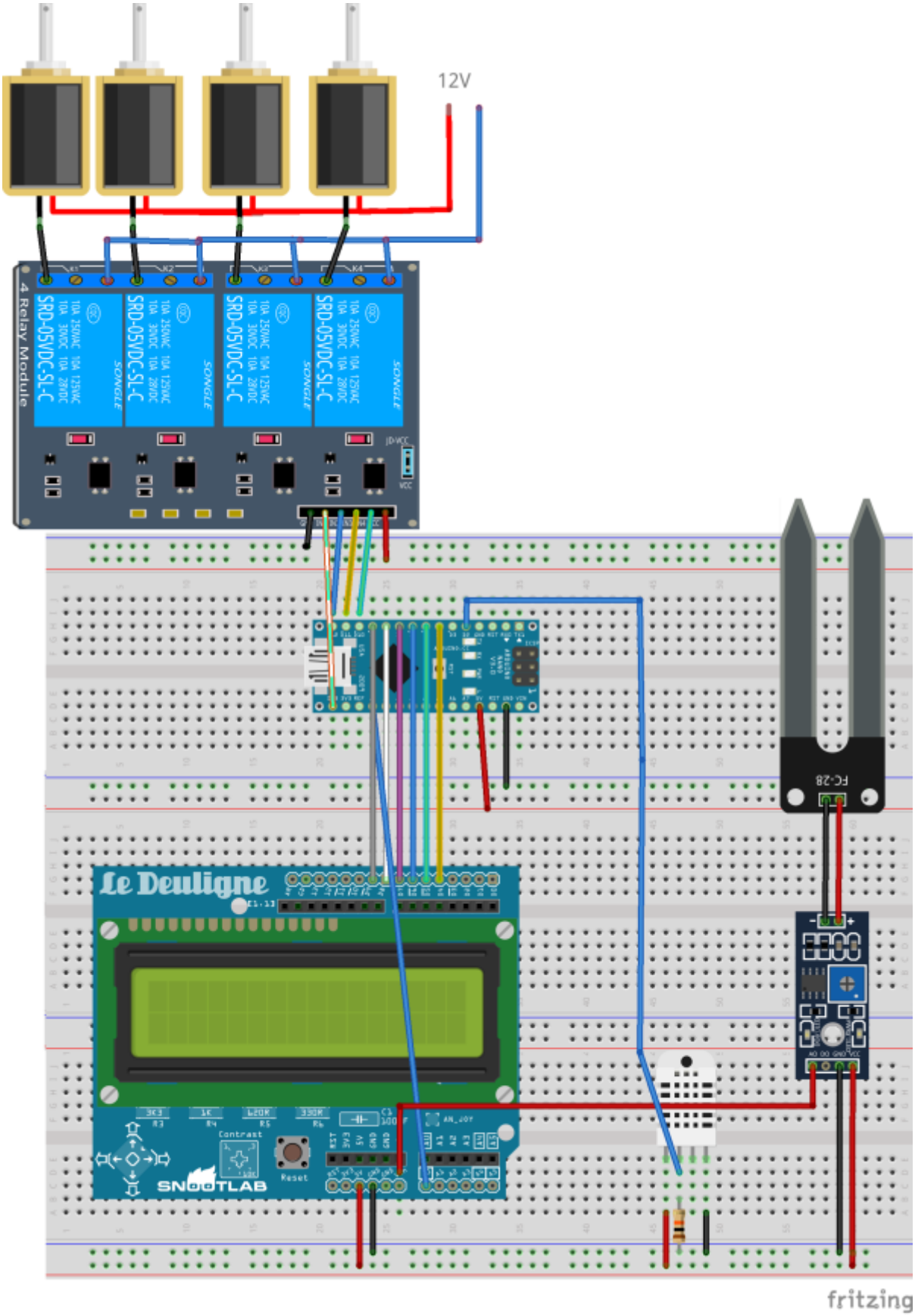


Como los relés nos permite aislar el arduino de cualquier otra fuente de alimentación, también podemos encender o apagar cualquier dispositivo que funcione con cualquier tensión que soporten los relés

En mi caso he comprado la siguiente placa de cuatro relés que permiten un máximo de 10A a 30VDC o 10 A a 250VAC.



El esquema quedaría de la siguiente forma:



Y modificamos el código :

```
//// wiki.intrusos.info
// modificación del código de http://webdelcire.com/wordpress/archives/2471
// riego v1.6 se cambia el motor por un relé de 4 salidas conectadas a
```

```
electrválvulas


//---------------------------Sensor DHT ----------------------------------
------------
#include "DHT.h" // Libreria para Sensores DHT
#define DHTPIN 2     // Pin del Arduino al cual esta conectado el pin 2 del
sensor
// Descomentar segun el tipo de sensor DHT usado
#define DHTTYPE DHT11   // DHT 11
DHT dht(DHTPIN, DHTTYPE);// Inicializa el sensor

//---------------LCD ------------------------------------------------------
--------------
#include <LiquidCrystal.h>  //Libreria LCD
// Inicializa el objeto LCD con los pines de la interfaz
LiquidCrystal lcd(8, 9, 4, 5, 6, 7);

//----------------- Reles --------------
byte pinRele1 = 10;
byte pinRele2 = 11;
byte pinRele3 = 12;
byte pinRele4 = 13;




// Pines analógicos
byte pinSensorHumedad = 1;

// definición teclado
int lcd_key     = 0;
int adc_key_in  = 0;
#define btnRIGHT  0
#define btnUP     1
#define btnDOWN   2
#define btnLEFT   3
#define btnSELECT 4
#define btnNONE   5

// Constantes para los posibles modos de funcionamiento
byte MODO_TIEMPO_HUMEDAD = 0;
byte MODO_SOLO_TIEMPO = 1;
byte MODO_SOLO_HUMEDAD = 2;


int modoElegido = 0;        // Modo de funcionamiento

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

```
int diasTemporizacion = 0;
int horasTemporizacion = 0;
int minutosTemporizacion = 0;

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

int read_LCD_buttons();  // para leer los botones

int read_LCD_buttons()
{ adc_key_in = analogRead(0);      // Leemos A0
  if (adc_key_in > 900)  return btnNONE;     // Ningun boton pulsado
  if (adc_key_in < 50)   return btnRIGHT;
  if (adc_key_in < 200)  return btnUP;
  if (adc_key_in < 400)  return btnDOWN;
  if (adc_key_in < 600)  return btnLEFT;
  if (adc_key_in < 800)  return btnSELECT;
  return btnNONE;  // Por si todo falla
}


//---------------------------------inicio setup ----------------------------
-------
void setup()
{

  // Inicializa el número de columnas y filas del LCD
  lcd.begin(16, 2);

  // Inicializamos los reles
  pinMode(pinRele1, OUTPUT);
  pinMode(pinRele2, OUTPUT);
  pinMode(pinRele3, OUTPUT);
```

```
  pinMode(pinRele4, OUTPUT);

  //Asegurar nivel ALTO en cada entrada de rele
  // los reles se activan con nivel BAJO (0)
  digitalWrite (pinRele1, RELAY_OFF);
  digitalWrite (pinRele2, RELAY_OFF);
  digitalWrite (pinRele3, RELAY_OFF);
  digitalWrite (pinRele4, RELAY_OFF);

  // Activa el led rojo y muestra la pregunta sobre el modo en el que va a
funcionar
  mostrarPregunta(modoElegido);

  // Pregunta por el modo de funcionamiento: tiempo + humedad, sólo tiempo o
sólo humedad
  boolean respuesta = false;
  while (!respuesta)
  {
    lcd_key = read_LCD_buttons();
    delay(200);  //para evitar lecturas seguidas
    if ( lcd_key == btnDOWN)
    {
      modoElegido--;
      if (modoElegido == -1) modoElegido = 2;
      mostrarPregunta(modoElegido);
    }
    else if ( lcd_key == btnUP)
    {
      modoElegido++;
      if (modoElegido == 3) modoElegido = 0;
      mostrarPregunta(modoElegido);
    }
    else if ( lcd_key == btnSELECT)
    {
      respuesta = true;
    }
  }


  // Si el modo elegido incluye riego por tiempo, pregunta cada cuantos
días, horas y minutos
  if (modoElegido == MODO_TIEMPO_HUMEDAD || modoElegido == MODO_SOLO_TIEMPO)
  {
    // No permite elegir un tiempo cero
    while (diasTemporizacion + horasTemporizacion + minutosTemporizacion ==
0)
    {
      diasTemporizacion = preguntarValor("Numero de dias:", 0, 30);
      horasTemporizacion = preguntarValor("Numero de horas:", 0, 23);
      minutosTemporizacion = preguntarValor("Numero minutos:", 0, 59);
    }
```

```
    // Muestra un resumen del tiempo elegido
    linea1 = "  DD : HH : MM";
    linea2 = "  " + str(diasTemporizacion) + " : " + str(horasTemporizacion)
+ " : " + str(minutosTemporizacion);
    mostrarTexto();
    delay(1000);
  }

  // Si el modo elegido incluye riego por sensor de humedad, pregunta cual
es el umbral de humedad para regar
  if (modoElegido == MODO_TIEMPO_HUMEDAD || modoElegido ==
MODO_SOLO_HUMEDAD)
  {
    umbralHumedad = preguntarValor("Umbral humedad:", 40, 99);
  }

  // Pregunta cuantos segundos deberá durar el riego cuando se active
  duracionRiego = preguntarValor("Duracion riego:", 15, 300);

}


void loop()
{
  // inicializa las variables de tiempo para el riego por temporización
  int diasPendientes = diasTemporizacion;
  int horasPendientes = horasTemporizacion;
  int minutosPendientes = minutosTemporizacion;
  int segundosPendientes = 0;

  if (modoElegido == MODO_TIEMPO_HUMEDAD)
  {
    // Continua el bucle hasta que el tiempo llegue a cero
    while ((diasPendientes + horasPendientes + minutosPendientes +
segundosPendientes) > 0)
    {
      // Espera un segundo y decrementa las variables el equivalente a un
segundo
      delay(987);
      segundosPendientes--;
      if (segundosPendientes == -1)
      {
        segundosPendientes = 59;
        minutosPendientes--;
      }
      if (minutosPendientes == -1)
      {
        minutosPendientes = 59;
        horasPendientes--;
      }
      if (horasPendientes == -1)
```

```
      {
        horasPendientes = 23;
        diasPendientes--;
      }

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

      // Actualiza el display con el tiempo hasta el siguiente riego y la
humedad actual
      linea1 = str(diasPendientes) + ":" + str(horasPendientes) + ":" +
str(minutosPendientes) + ":" + str(segundosPendientes);
      linea2 = "Humedad:" + str(mediaHumedad) + "\x25" + " [" +
str(umbralHumedad) + "]";
      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;
      }
    }

    // Activa lo bomba de riego durante el tiempo configurado
    regar();
  }
  else if (modoElegido == MODO_SOLO_TIEMPO)
  {
    // Continua el bucle hasta que el tiempo llegue a cero
    while ((diasPendientes + horasPendientes + minutosPendientes +
segundosPendientes) > 0)
    {
      delay(990);
      segundosPendientes--;
      if (segundosPendientes == -1)
      {
        segundosPendientes = 59;
        minutosPendientes--;
      }
      if (minutosPendientes == -1)
      {
        minutosPendientes = 59;
        horasPendientes--;
      }
      if (horasPendientes == -1)
      {
```

```
        horasPendientes = 23;
        diasPendientes--;
      }

      // Actualiza el display con el tiempo hasta el siguiente riego
      linea1 = "Proximo riego:";
      linea2 = str(diasPendientes) + ":" + str(horasPendientes) + ":" +
str(minutosPendientes) + ":" + str(segundosPendientes);
      mostrarTexto();
    }

    // Activa lo bomba de riego durante el tiempo configurado
    regar();
  }
  else if (modoElegido == MODO_SOLO_HUMEDAD)
  {
    while (true)
    {
      delay(1000);

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

      // Actualiza el display con la humedad actual y el número de riegos
efectuados hasta el momento
      linea1 = "Humedad: " + str(mediaHumedad) + "\x25" + " [" +
str(umbralHumedad) + "]";  // \x25 es el símbolo ascii de %
      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;
      }
    }
  }
}


// Pregunta el modo de funcionamiento con la última opción elegida
void mostrarPregunta(byte modo)
{
  lcd.clear();
  lcd.print("Modo de riego?");
  lcd.setCursor(0, 1);
```

```
  if (modo == MODO_TIEMPO_HUMEDAD) lcd.print("Tiempo + Humedad");
  else if (modo == MODO_SOLO_TIEMPO) lcd.print("Solo Tiempo");
  else if (modo == MODO_SOLO_HUMEDAD) lcd.print("Solo Humedad");
}

// Muestra el texto configurado en el display
void mostrarTexto()
{
  lcd.clear();
  lcd.print(linea1);
  lcd.setCursor(0, 1);
  lcd.print(linea2);
}


// Muestra una pregunta y recoge un valor numérico
int preguntarValor(String texto, int inicial, int maximo)
{
  linea1 = texto;
  linea2 = str(inicial);
  mostrarTexto();
  boolean respuesta = false;
  int valor = inicial;
  while (!respuesta)
  {
    lcd_key = read_LCD_buttons();
    delay(200); //para evitar lecturas seguidas
    if ( lcd_key == btnDOWN)
    { valor--;
      if (valor == -1) valor = maximo;
      linea2 = str(valor);
      mostrarTexto();
    }

    else if ( lcd_key == btnUP)
    {
      valor++;
      if (valor > maximo) valor = 0;
      linea2 = str(valor);
      mostrarTexto();
    }

    else if ( lcd_key == btnSELECT)
    {
      respuesta = true;
    }
  }
  return valor;
}

// 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));
}


// Actualiza la variable con la media de humedad de las últimas 10 lecturas
void leerHumedad()
{
  lecturasHumedad[indiceLecturasHumedad] = analogRead(pinSensorHumedad);
  lecturasHumedad[indiceLecturasHumedad] = map
(lecturasHumedad[indiceLecturasHumedad], 0, 1023, 100, 0); // Mapeamos el
valor del sensor de 0 a 100
  indiceLecturasHumedad++;
  if (indiceLecturasHumedad > 9)
  {
    indiceLecturasHumedad = 0;
    mediaLista = true;
  }

  mediaHumedad = 0;
  for (int i = 0; i < 10; i++) mediaHumedad += lecturasHumedad[i];
  mediaHumedad /= 10;
  if (mediaHumedad > humedadMaxima) humedadMaxima = mediaHumedad;
  if (mediaHumedad < humedadMinima) humedadMinima = mediaHumedad;
  mediaHumedad -= humedadMinima;

  mediaHumedad = (double)((double)mediaHumedad / (double)(humedadMaxima -
humedadMinima)) * 100;
  if (mediaHumedad == 100) mediaHumedad = 99;
}


// Activa lo bomba de riego durante el tiempo configurado
void regar()
{
  if (riegos == limiteRiegos)
  {
    linea1 = " - BLOQUEADO - ";
    linea2 = "Limite de riegos";
    mostrarTexto();
    while (true) {
      delay(999999);
    }
  }
  int riegoPendiente = duracionRiego;
  digitalWrite(pinRele1, LOW);
  digitalWrite(pinRele2, LOW);
  digitalWrite(pinRele3, LOW);
```

```
  digitalWrite(pinRele4, LOW);
  while (riegoPendiente > 0)
  {
    linea1 = " -- REGANDO -- ";
    linea2 = "Restante: " + str(riegoPendiente);
    mostrarTexto();
    delay(990);
    riegoPendiente--;
  }
  digitalWrite(pinRele1, HIGH);
  digitalWrite(pinRele2, HIGH);
  digitalWrite(pinRele3, HIGH);
  digitalWrite(pinRele4, HIGH);
  riegos++;
}
```

Referencias

- http://www.profetolocka.com.ar/2015/05/09/modulo-de-4-reles-para-arduino/

From:
https://wiki.intrusos.info/ - **LCWIKI**

Permanent link:
https://wiki.intrusos.info/doku.php?id=electronica:arduino:valvula

Last update: **182023/01/ 13:36**