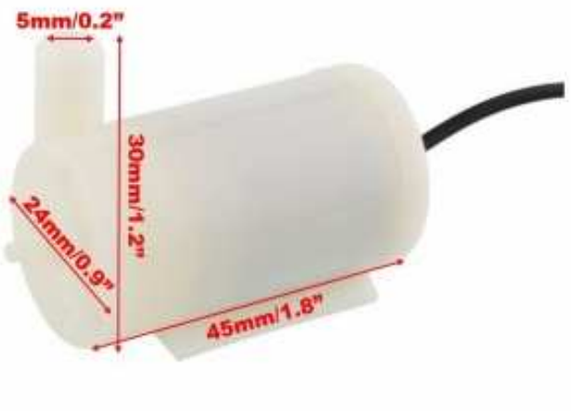
Mini Bomba De Agua Dc Sumergible 3v 5v Arduino - Pic

(Versión 13-3-19)

Para proyectos de riego automático



Esta bomba de agua se moverá 2 Litros por minuto, sirve para trabajo pesado con características de funcionamiento de 6V, 0.3A del motor y un cuerpo termo-plástico resistente. Es totalmente sumergible y refrigerado por agua. Utilice esta bomba para hacer una fuente o una cascada, incluso regar sus plantas.

Especificaciones:

- Voltaje DC: 2.5-6V
- Elevación máxima: 40-110cm / 15.75 "-43,4"
- Caudal: 80-120L / H
- Diámetro exterior de salida de agua: 7,5 mm / 0,3 "
- Dentro de diámetro de salida del agua: 5 mm / 0.2 "
- Diámetro: Aprox. 24mm / 0.95 "
- Duración: Aprox. 45mm / 1.8 "
- Altura: Aprox. 30mm / 1.2 "
- Material: plástico de ingeniería
- El modo de conducir: diseño de corriente continua sin escobillas, de conducción magnética

---

DISEÑO DE UN SISTEMA DE RIEGO PARA MACETAS

Dado el caudal entregado por la bomba y los requerimientos de agua de las plantas en las macetas, podría una misma bomba suministrar agua a más de una maceta, mediante un sistema de cañerías con conectores en T.

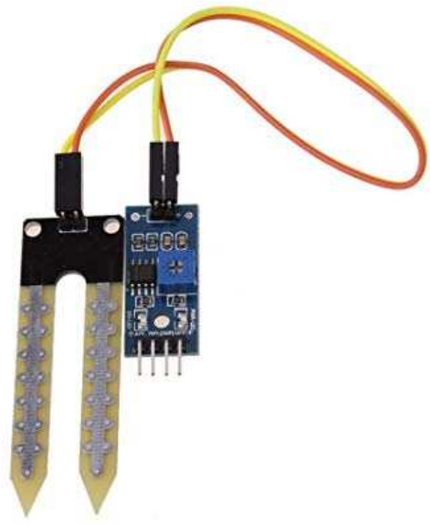
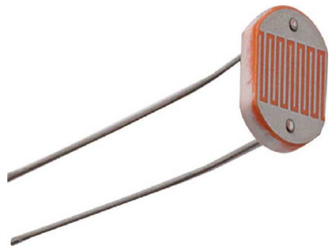
Si no es suficiente, puede que se necesiten mas de una bomba.

A partir de aquí podemos crear o buscar distintas alternativas de control para este sistema de riego:

Mediante sensores de humedad de tierra, temperatura, luminosidad etc., podemos darle distinta complejidad.

Por ejemplo mediante la lectura de la humedad de la tierra y la condición de luminosidad del ambiente, podremos establecer el momento para el riego. La temperatura también podría jugar un factor de decisión.

Los sensores serian:

 SENSOR DE HUMEDAD DE SUELO	 SENSOR DE LUZ (LDR)	 SENSOR DE TEMPERATURA

También se podría llevar algún tipo de registro visualizado en un **LCD.** O intercambiar información con el **móvil** y que el mismos usuario ejecute la orden de activación de bombas.

Pero una opción más sencilla y más segura, podría ser por medio de un reloj, programado el sistema para que inicie el riego en determinados momentos del día. Estamos hablando de un sistema de control no realimentado, es decir no se tomaría información de la salida para enviar a la entrada y corregir la salida.

El riesgo de utilizar un sistema u otro, depende del dueño de las plantas, de cuanto tiempo y dinero que desea gastar en ellas.

Un sistema controlado solo por reloj tendría el inconveniente que si las plantas no absorben el agua enviada en los tiempos considerados, estas podrían morir por exceso de agua.

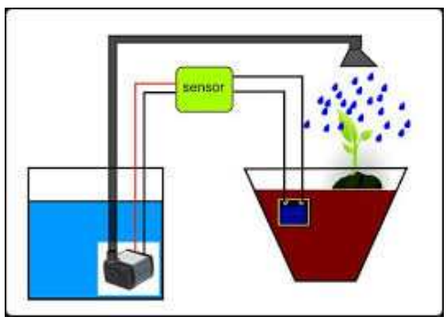
El sistema con los sensores también podría fallar como consecuencia de la falla en alguno de ellos en su lectura o funcionamiento.

Como en cualquier sistema de control, mediante el agregado de redundancias se consigue que el mismo cumpla con su función.

La redundancia en sistemas de control apunta a disponer elementos adicionales que garantizan su funcionamiento si uno de sus componentes falla.

SISTEMA DE CONTROL DE RIEGO DE MACETAS

MEDIANTE RELOJ *SOLAMENTE*



Módulo RTC DS1302.

Un módulo RTC (Real Time Clock) o "Reloj de tiempo real" consiste en un circuito integrado alimentado por una batería el cual, en todo momento, registra la fecha, día de la semana y hora al igual que un reloj digital convencional.

Crearemos el programa que accione 2 bombas mediante Reles.

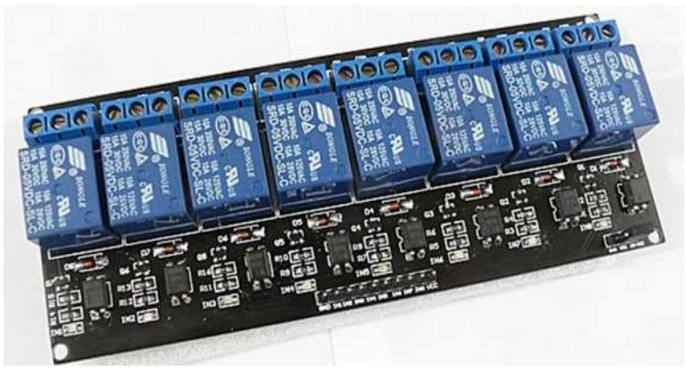
Sin embargo utilizaremos un hardware que eventualmente podría manejar 8 bombas. Esto es debido a que hemos conectado la placa Arduino a un modulo de 8 reles.

Adicionalmente con fines de práctica se estarán usando 2 de las entradas analógicas como salidas digitales.

MATERIALES PRINCIPALES EMPLEADOS:

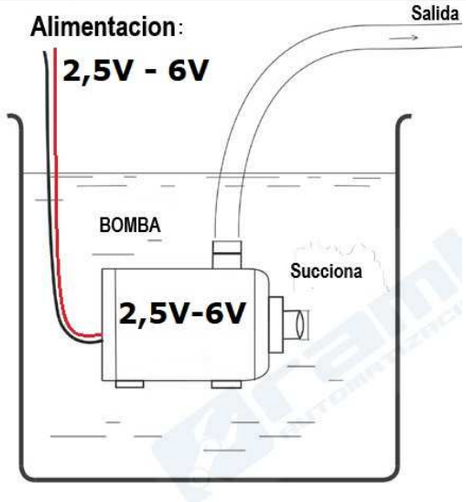
Placa Arduino Uno

Bomba sumergible que funciona con 5 v.



Modulo de 8 reles

Fuente de alimentación externa para alimentar las bombas (el consumo de las bombas no puede ser entregado por la placa Arduino). También la placa Arduino va a ser alimentada por esta fuente para protoboard.



Recipiente para el agua donde se sumerge la bomba.

Adaptador 220v a 9V

El siguiente programa se llama **RIEGO_HORAS**

```
//programa  RIEGO EN CIERTAS HORAS – HORA DE RIEGO 1 -2 - 3

//Las 2 bombas de este ejemplo se activan por medio de 0(cero) en los pines 3 y 14

#include <DS1302.h> // Librería del reloj

float TRIEGO;

float M1RIEGO= 8;  //Hora de riego 1

float M2RIEGO= 14; //Hora de riego 2

float M3RIEGO= 19; //Hora de riego 3

int DURARIEGO= 8000; // Dura DURARIEGO milisegundos el riego

int TESTIGO= 0; //Variable testigo para establecer un riego a MxRIEGO

                        // Un riego en cada hora elegida

// Inicializacion del modulo.

/* Conexion Modulo Reloj con  Arduino

 *

 *  GND = GND

 *  CLK = 8

 *  DAT = 7

 *  RST = 6

 */

//Configuración del reloj

DS1302 rtc(6, 7, 8); // Fijamos los pines de conexión con el Arduino

Time t;

void setup()

{


for (int i=2; i<6; i++) // declara i y prueba si es menor

                            //que 6, incrementa i.

{

pinMode(i, OUTPUT); // configura ‘pin’ como Salida

digitalWrite(i, HIGH); //todos los fija en alto

}

pinMode(13, OUTPUT); // configura ‘pin’ como Salida

digitalWrite(13, LOW); //fija en BAJO

pinMode(14, OUTPUT); // configura ‘pin’ como Salida
```

```
digitalWrite(14, HIGH); //fija en alto

//Arranca bombas DURARIEGO milisegundos por unica vez al iniciar o resetear Arduino

//Debemos recordar que los módulos reles tienen optoacoplador y se activan con 0 (cero)

delay(500); // Demora de estabilización

digitalWrite(14,LOW ); //Activa rele bomba 1

delay(300);

digitalWrite(3,LOW); //Activa rele bomba 2

delay(DURARIEGO); //tiempo en el cual se esta regando

digitalWrite(14,HIGH ); //Desactiva rele bomba 1

delay(300);

digitalWrite(3,HIGH ); //Desactiva rele bomba 2

//fin inicio activo de bombas al resetear programa

// Inicializacion del puerto serie.

Serial.begin(9600);

}


void loop()

{

  delay(5000);  // Demora para no sobrecargar las comunicaciones con el modulo.

//dos parpadeos LED 13 – Es una indicación visual para el usuario de que el programa esta corriendo

  digitalWrite(13,HIGH );

  delay(300);

  digitalWrite(13,LOW);

  delay(300);

  digitalWrite(13,HIGH );

  delay(300);

  digitalWrite(13,LOW);

//fin parpadeo LED 13

  // Obtencion de datos

  t = rtc.getTime();

 // Publicar  datos de fecha y hora  en MONITOR SERIE

 // Se publicara el dia de la semana

  Serial.print("HOY:");

  if (t.dow == 1) Serial.print("lunes");  // La variable t.dow (dia de la semana) tendra valor de 1
```

```
                                        //para día lunes y 7 para domingo.
  if (t.dow == 2) Serial.print("martes");
  if (t.dow == 3) Serial.print("miercoles");
  if (t.dow == 4) Serial.print("jueves");
  if (t.dow == 5) Serial.print("viernes");
  if (t.dow == 6) Serial.print("sabado");
  if (t.dow == 7) Serial.print("domingo");
    Serial.println();  // Linea aparte
    // Se publicaran datos de fecha, en números.
  Serial.print("DIA:");
  Serial.print(t.date, DEC);  // Dia del mes
   Serial.println();
    // Mes
  Serial.print("MES:");
  Serial.print(t.mon);
    Serial.println();
    // Año
  Serial.print("DE:");
  Serial.print(t.year, DEC);
    Serial.println();
    Serial.print("HORA:");  // Hora en formato 0-23.
  Serial.print(t.hour, DEC);
    Serial.print(", MIN:");  // Minutos.
  Serial.print(t.min, DEC);
//*************************************************
// Almacenamiento en variable de la HORA
TRIEGO =  t.hour, DEC;// carga el dato HORA  de tiempo actual
                                //El contenido de esta variable se utilizara para saber si se debe regar
//*************************************************
  //SEGUNDOS
  Serial.print(", SEG:");  // Segundos.
  Serial.print(t.sec, DEC);
  Serial.println();
  Serial.println();
```

```
  Serial.println();

// Se termina de mostrar por MONITOR SERIE datos de fecha y hora



//Se chequea si es momento de regar

if (((TRIEGO == M1RIEGO) || (TRIEGO == M2RIEGO)  || (TRIEGO == M3RIEGO))&& TESTIGO == 0)

{

digitalWrite(14,LOW );

delay(300);

digitalWrite(3,LOW );

TESTIGO= 1; //Hace testigo igual a 1 para que solo se riegue una vez cuando TESTIGO=0

Serial.println("-------------------------");

 Serial.print("Regando a las= ");

 Serial.println(TRIEGO);

  Serial.println("-------------------------");

delay(DURARIEGO);//Regando este tiempo

digitalWrite(14,HIGH );

delay(300);

digitalWrite(3,HIGH );

delay(1000);//Tiempo muerto

 }

//Va a recuperar el valor de testigo para que se pueda regar en proximo momento

  if ((TRIEGO == M1RIEGO +1) || (TRIEGO == M2RIEGO +1)  || (TRIEGO == M3RIEGO+1))

  {

  TESTIGO= 0; //Restaura TESTIGO a 0 para que se pueda regar en proximo momento de riego

    }

}
```

Lo que deberia recordar del uso de la librería de este reloj:

Obtención de datos del RELOJ y su carga en la variable t

t = rtc.getTime();

Entrega un numero entero que representa el nombre del día de la semana. Donde 1 lunes- 2 martes -3 miercoles – 4 jueves – 5 viernes – 6 sabado – 7 domimgo.

t.dow

Día del mes, 1.2.3…….31

t.date, DEC

Mes

t.mon

Año

t.year, DEC

Hora en formato 0 -23

t.hour, DEC

Minutos

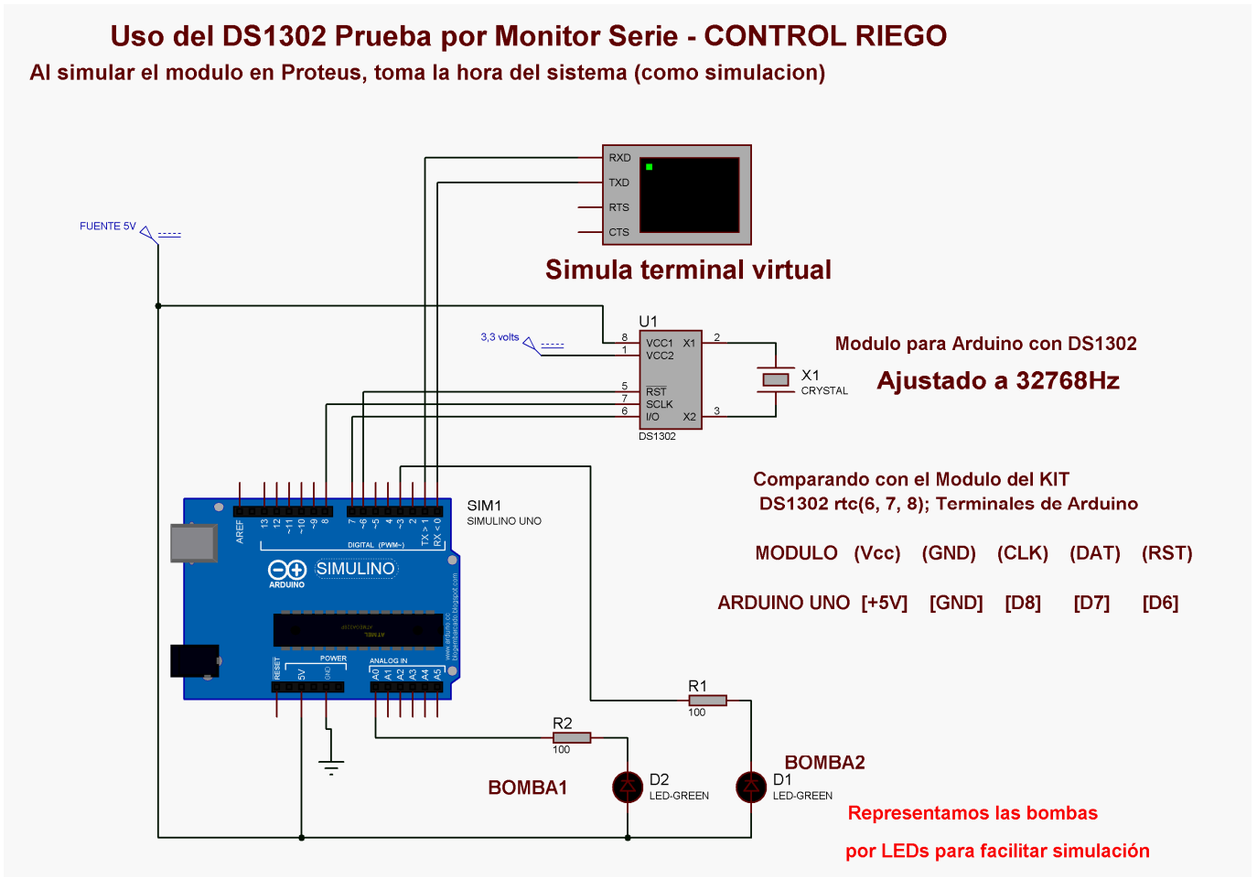
t.min, DEC

Segundos

t.sec, DEC

SIMULACION DEL EJERCICIO PROPUESTO

Para la simulación partiremos del programa anterior, pero cambiaremos las horas de riego por segundos, de este modo durante cada minuto se regara en 3 veces. Se ha de tener en cuenta que para facilitar la simulación se han convertido en comentario algunas lineas de programa, eliminándose además algunos retardos. RECUERDE que va a depender de la velocidad de su computadora.



A continuación el programa empleado con sus ajustes.

Se llama **RIEGO_SIMULACION**

```
//programa  RIEGO EN CIERTOS SEGUNDOS EN UN MINUTO – SEGUNDOS DE RIEGO 1 -2 - 3
//Las 2 bombas de este ejemplo se activan por medio de 0(cero) en los pines 3 y 14
#include <DS1302.h> // Librería del reloj
float TRIEGO;
float M1RIEGO= 15;  //Segundo de riego 1
float M2RIEGO= 30; //Segundo de riego 2
float M3RIEGO= 45; //Segundo de riego 3
int DURARIEGO= 900; // Dura DURARIEGO milisegundos el riego
int TESTIGO= 0; //Variable testigo no utilizada en esta version
                        // Un riego en cada hora elegida
// Inicializacion del modulo.
/* Conexion Modulo Reloj con  Arduino
 *
 *  GND = GND
 *  CLK = 8
 *  DAT = 7
 *  RST = 6
 */
//Configuración del reloj
DS1302 rtc(6, 7, 8); // Fijamos los pines de conexión con el Arduino
Time t;
void setup()
{
for (int i=2; i<6; i++) // declara i y prueba si es menor
                            //que 6, incrementa i.
{
pinMode(i, OUTPUT); // configura ‘pin’ como Salida
digitalWrite(i, HIGH); //todos los fija en alto
}
pinMode(13, OUTPUT); // configura ‘pin’ como Salida
digitalWrite(13, LOW); //fija en BAJO
```

```
pinMode(14, OUTPUT); // configura ‘pin’ como Salida

digitalWrite(14, HIGH); //fija en alto

//Arranca bombas DURARIEGO milisegundos por unica vez al iniciar o resetear Arduino

//Debemos recordar que los módulos reles tienen optoacoplador y se activan con 0 (cero)

delay(500); // Demora de estabilización

digitalWrite(14,LOW ); //Activa rele bomba 1

//delay(300);

digitalWrite(3,LOW); //Activa rele bomba 2

delay(DURARIEGO); //tiempo en el cual se esta regando

digitalWrite(14,HIGH ); //Desactiva rele bomba 1

//delay(300);

digitalWrite(3,HIGH ); //Desactiva rele bomba 2

//fin inicio activo de bombas al resetear programa

// Inicializacion del puerto serie.

Serial.begin(9600);

Serial.print("Sistema de Riego (SIMULACION)");

}

void loop()

{

 // delay(5000);  // Demora para no sobrecargar las comunicaciones con el modulo.

//dos parpadeos LED 13 – Es una indicación visual para el usuario de que el programa esta corriendo

//NO SE USA PARPADEO DE LED 13 EN ESTE EJEMPLO PARA SIMULAR

//fin parpadeo LED 13

  // Obtencion de datos

  t = rtc.getTime();

 // Publicar  datos de fecha y hora  en MONITOR SERIE

 // Se publicara el dia de la semana

 // Serial.print("HOY:");

 // if (t.dow == 1) Serial.print("lunes");  // La variable t.dow (dia de la semana) tendra valor de 1

                                            //para día lunes y 7 para domingo.

 // if (t.dow == 2) Serial.print("martes");

 // if (t.dow == 3) Serial.print("miercoles");

  //if (t.dow == 4) Serial.print("jueves");

 // if (t.dow == 5) Serial.print("viernes");
```

```
//  if (t.dow == 6) Serial.print("sabado");

//  if (t.dow == 7) Serial.print("domingo");

//    Serial.println();  // Linea aparte

    // Se publicaran datos de fecha, en números.

//  Serial.print("DIA:");

//  Serial.print(t.date, DEC);  // Dia del mes

//   Serial.println();

    // Mes

//  Serial.print("MES:");

//  Serial.print(t.mon);

//    Serial.println();

    // Año

//  Serial.print("DE:");

//  Serial.print(t.year, DEC);

//    Serial.println();

//    Serial.print("HORA:");  // Hora en formato 0-23.

//  Serial.print(t.hour, DEC);

//    Serial.print(", MIN:");  // Minutos.

//  Serial.print(t.min, DEC);

//*************************************************

// Almacenamiento en variable de la HORA (segundos en esta simulacion)

TRIEGO =  t.sec, DEC;// carga el dato seg  de tiempo actual

                                //El contenido de esta variable se utilizara para saber si se debe regar

//*************************************************

  //SEGUNDOS

 // Serial.print(", SEG:");  // Segundos.

//  Serial.print(t.sec, DEC);

//  Serial.println();

//  Serial.println();

//  Serial.println();

// Se termina de mostrar por MONITOR SERIE datos de fecha y hora

//Se chequea si es momento de regar

if (((TRIEGO == M1RIEGO) || (TRIEGO == M2RIEGO)  || (TRIEGO == M3RIEGO))&& TESTIGO == 0)

{
```

```
digitalWrite(14,LOW );

//delay(30);

digitalWrite(3,LOW );

//TESTIGO= 1; //Hace testigo igual a 1 para que solo se riegue una vez cuando TESTIGO=0

Serial.println("------------------------");

 Serial.print("Regando a las= ");

 Serial.println(TRIEGO);

  Serial.println("------------------------");

delay(DURARIEGO);//Regando este tiempo

digitalWrite(14,HIGH );

//delay(30);

digitalWrite(3,HIGH );

//delay(1000);//Tiempo muerto

 }

//Va a recuperar el valor de testigo para que se pueda regar en proximo momento

//  if ((TRIEGO == M1RIEGO +1) || (TRIEGO == M2RIEGO +1)  || (TRIEGO == M3RIEGO+1))

//  {

//  TESTIGO= 0; //Restaura TESTIGO a 0 para que se pueda regar en proximo momento de riego

//    }

delay(500); //demora para no sobrecargar la lectura del modulo

}
```

CONCLUSIONES:

Recuerde que la fuente de alimentación empleada debe no solo ser capaz de almentar la placa Arduino sino los reles y las bombas.