[Fecha]

REPORTE DEL CIRCUITO REGULADOR DE VOLTAJE

irfz44n

JINMI JORDANI FLORES ALTAMIRANO

Reporte del proyecto del dispositivo hr411

**El proyecto realizado se fue creando de la siguiente manera**

**MOSFET IRFZ44N:** Es un [transistor](https://www.ecured.cu/Transistor) de tecnología MOS-FET (Metal–Oxide–Semiconductor – Field Effect Transistor) y de alta potencia que posee destacadas características que lo hacen ideal para aplicaciones de conmutación y en la modulación por ancho de pulso (PWM).

Este tipo de componente es fabricado por la [International Rectifier](https://www.ecured.cu/index.php?title=International_Rectifier&action=edit&redlink=1)entre otros fabricantes que más adelante detallaremos, utiliza técnicas avanzadas de procesamiento para lograr muy baja [resistencia](https://www.ecured.cu/Resistencia) por área de [silicio](https://www.ecured.cu/Silicio). Este beneficio, combinado con la [velocidad](https://www.ecured.cu/Velocidad) de conmutación y su diseño solido de dispositivos de los [MOSFET](https://www.ecured.cu/Transistor_MOSFET) de potencia HEXFET, ofrece al diseñador una extremadamente eficiencia y un dispositivo fiable para su uso en una amplia variedad de aplicaciones.

# **resistencia**

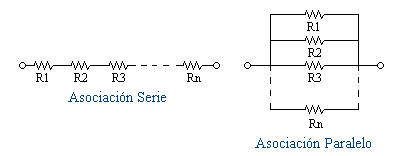
**e**ste componente imprescindible en la construcción de cualquier equipo electrónico, ya que permite distribuir adecuadamente la corriente y voltaje a todos los puntos necesarios.

El valor de la resistencia se expresa en ohmio, al cual representamos con el símbolo 

Si sometemos los extremos de una resistencia al paso de una corriente continua se producirá en la misma una caída de tensión proporcional a su valor. La intensidad que la atraviese será también proporcional a la tensión aplicada y al valor en ohmios de la resistencia. Para calcular dicha relación no hay más que aplicar la Ley de Ohm:



Hay dos formas de asociar resistencias en un circuito: asociación serie y asociación paralela:



La resistencia equivalente de un circuito serie es:

RT = R1 + R2 + R3 + ... + Rn

lo cual nos indica que una sola resistencia de valor RT se comportará de la misma forma que las n resistencias R1, R2, R3… Rnconectadas en serie.

Si el circuito es en paralelo entonces la resistencia equivalente es:

RT = 1/ (1/R1 + 1/R2 + 1/R3 + ... + 1/Rn)

Cuando se trata de dos resistencias en paralelo se tiene:

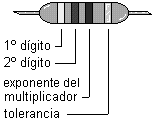
http://www.viasatelital.com/proyectos_electronicos/resist4.gif

### Código de colores



Identificar un resistor no es una tarea muy complicada, se observa que estos poseen 4 bandas de colores, 3 de idénticas proporciones y una más alejada de éstas. Estas bandas representan el valor real del resistor incluyendo su porcentaje de tolerancia o error siguiendo un código de colores estándar.

En primer lugar, tratamos de identificar el extremo que corresponde a la banda de tolerancia del resistor, que en la mayoría de los casos suele ser dorada (5%) o (algo más raro) plateada (10%). Una vez localizada ésta la dejamos de lado, (literalmente a la derecha), vamos al otro extremo y leemos la secuencia:



# **ARDUINO UNO R3**

Este es el nuevo Arduino Uno R3 utiliza el microcontrolador ATmega328. En adición a todas las características de las tarjetas anteriores, el Arduino Uno utiliza el ATmega16U2 para el manejo de USB en lugar del 8U2 (o del FTDI encontrado en generaciones previas). Esto permite ratios de transferencia más rápidos y más memoria. No se necesitan drivers para Linux o Mac (el archivo inf para Windows es necesario y está incluido en el IDE de Arduino).

La tarjeta Arduino Uno R3 incluso añade pins SDA y SCL cercanos al AREF. Es más, hay dos nuevos pines cerca del pin RESET. Uno es el IOREF, que permite a los shields adaptarse al voltaje brindado por la tarjeta. El otro pin no se encuentra conectado y está reservado para propósitos futuros. La tarjeta trabaja con todos los shields existentes y podrá adaptarse con los nuevos shields utilizando esos pines adicionales.

El Arduino es una plataforma computacional física open-source basada en una simple tarjeta de I/O y un entorno de desarrollo que implementa el lenguaje Processing/Wiring. El Arduino Uno R3 puede ser utilizado para desarrollar objetos interactivos o puede ser conectado a software de tu computadora (por ejemplo, Flash, Processing, MaxMSP). El IDE open-source puede ser descargado gratuitamente (actualmente para Mac OS X, Windows y Linux).

Nota: Esta plataforma requiere la carpeta de drivers Arduino 1.0 para poder instalarlo de forma apropiada en algunos computadores. Hemos testeado y confirmado que el Arduino Uno R3 puede ser programado en versiones anteriores del IDE. Sin embargo, la primera vez que uses el Arduino en una nueva computadora deberás tener el Arduino 1.0 instalado en la máquina. Si estás interesado en leer más acerca de los cambios en el IDE, revisa las notas oficiales de Arduino 1.0.

Características:

Microcontrolador ATmega328.

Voltaje de entrada 7-12V.

14 pines digitales de I/O (6 salidas PWM).

6 entradas análogas.

32k de memoria Flash.

Reloj de 16MHz de velocidad.

/\*

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

{==[=======> (CODIGO DEL ARDUINO) <=======]==}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\*/

int Motor = 6; // declaramos las variables de tipo entero

int entrada\_analogica= A0;

void setup() {

pinMode (Motor, OUTPUT); // designamos los pines como pines de salida y entrada

pinMode (entrada\_analogica, INPUT); // entrada

}

void loop() { //entramos a un siclo repetitivo

int valor\_entrada\_analogica =0; // declaramos las variables de tipo entero

int valor\_salida\_pwm=0; // declaramos la variable de tipo entero para la ulizacion de la modulacion por ancho de pulso

// ojo ( estos pines tienes que ser pines de (PWM)

valor\_entrada\_analogica = analogRead(entrada\_analogica); // Guardamos la lectura analogica en la variable (valor\_entrada\_analógica)

valor\_salida\_pwm =map(valor\_entrada\_analogica,0, 1023,0,255); // guardamos el mapeo en la variable (valor salida)

analogWrite(Motor, valor\_salida\_pwm); // manifestamos en el led la lectura analogica que hemos reducido de( 0 - 255)

// donde 0 es 0 voltios y 255 viene ser 5 voltios

}

