INSTITUTO TÉCNICO INDUSTRIAL SAN JUAN BOSCO - CALI
TALLER DE ELECTRICIDAD Y ELECTRÓNICA
ELECTRICIDAD Y ELECTRÓNICA – GRADO 11

CONTENIDO: Arduino – 44496
ACTIVIDAD: Control de carga A.C.

NOMBRE:

FECHA: 13 de marzo de 2019

OBJETIVOS:

1. Conocer los aspectos básicos de Arduino.
2. Implementar un circuito eléctrico/electrónico para el control de una carga A.C.
3. Diseñar un programa para controlar una carga A.C. con un pulsador y Arduino.

TIEMPO PARA DESARROLLAR LA PRÁCTICA: 1 hora de clase para explicación de la guía y solución de dudas. 6 horas de clase para desarrollo práctico en protoboard. 6 horas de clase para desarrollo práctico en tarjeta universal. Fecha aproximada de revisión de la práctica: 20 de marzo protoboard. 27 de marzo tarjeta universal.

MARCO TEÓRICO:

```
const int buttonPin = 3;

// setup initializes serial and the button pin
void setup()
{
  Serial.begin(9600);
  pinMode(buttonPin, INPUT);
}

// loop checks the button pin each time,
// and will send serial if it is pressed
void loop()
{
  if (digitalRead(buttonPin) == HIGH)
    Serial.write('H');
  else
    Serial.write('L');

  delay(1000);
}
```

A continuación se detallan algunos conceptos importantes al trabajar con Arduino.
setup(): Se llama cuando se inicia un diseño o proyecto. Utilícelo para inicializar variables, modos de pines, iniciar para trabajo con librerías, etc. La función setup sólo se ejecutará una vez, después de cada encendido o reinicio de la placa Arduino.

loop(): Después de crear una función setup (), que inicializa y establece los valores iniciales, la función loop () hace precisamente lo que su nombre sugiere, y lazos consecutivamente, permitiendo que su programa cambie y responda. Utilícelo para controlar activamente la tarjeta Arduino.

Operadores De Comparación:

```
x == y (x is equal to y)
x != y (x is not equal to y)
x < y (x is less than y)
x > y (x is greater than y)
x <= y (x is less than or equal to y)
x >= y (x is greater than or equal to y)
```

Estructuras De Control

if:

```
if (someVariable > 50)
{
  // do something here
}
```

```
if (x > 120) digitalWrite(LEDpin, HIGH);

if (x > 120)
digitalWrite(LEDpin, HIGH);

if (x > 120){ digitalWrite(LEDpin, HIGH); }

if (x > 120){
  digitalWrite(LEDpin1, HIGH);
  digitalWrite(LEDpin2, HIGH);
}                                    // all are correct
```

if… else:

```
if (pinFiveInput < 500)
{
  // action A
}
else
{
  // action B
}
```

```
if (pinFiveInput < 500)
{
  // do Thing A
}
else if (pinFiveInput >= 1000)
{
  // do Thing B
}
else
{
  // do Thing C
}
```

Switch Case:
Var: La variable cuyo valor se va a comparar con los diferentes casos.
Label: Valor para comparar con la variable.

```
switch (var) {
  case label:
    // statements
    break;
  case label:
    // statements
    break;
  default:
    // statements
  break;
}
```

Estructuras De Repetición

While:

```
while(expression){
  // statement(s)
}
```

Do While:

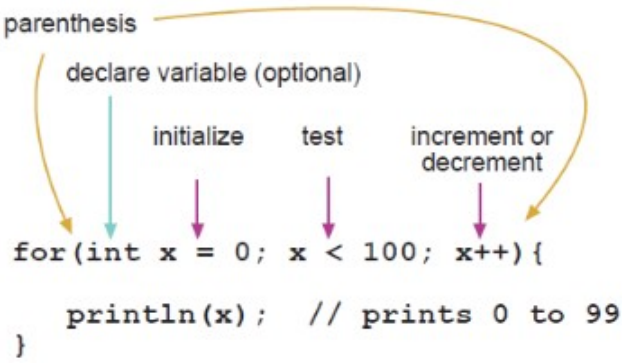
```
do
{
    // statement block
} while (test condition);
```

For:

```
for (initialization; condition; increment) {

//statement(s);

}
```



Funciones Especiales De Arduino

pinMode(): Configura el pin especificado para comportarse bien como entrada o como salida.
Sintaxis= pinMode(pin,mode)
pin: Número del pin cuyo modo desea establecer.
mode: INPUT, OUTPUT, INPUT_PULLUP.

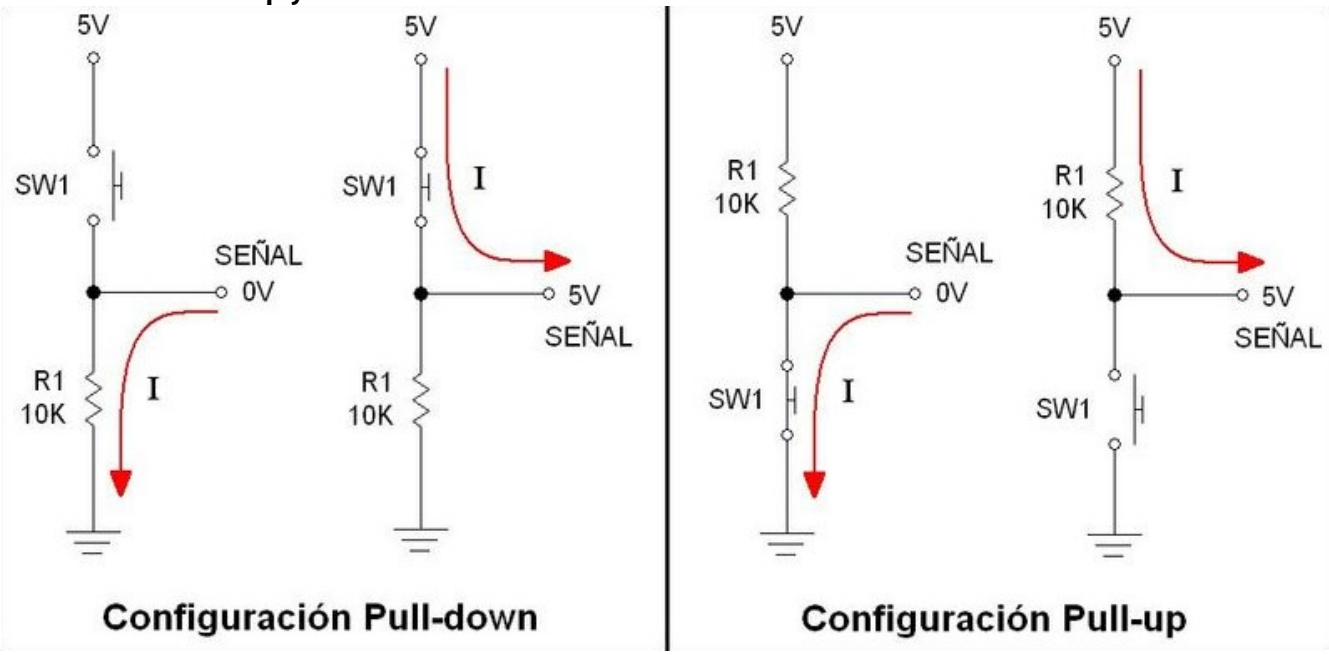
digitalWrite(): Escribe un valor ALTO o BAJO en un pin digital. Si el pin ha sido configurado como OUTPUT con pinMode (), su voltaje se ajustará al valor correspondiente: 5V (o 3.3V en placas de 3.3V) para HIGH, 0V (tierra) para LOW. Si el pin está configurado como INPUT, digitalWrite () activará (HIGH) o desactivará (LOW) el pullup interno en el pin de entrada. Se recomienda ajustar el pinMode () a INPUT_PULLUP para habilitar la resistencia pull-up interna.
Sintaxis= digitalWrite(pin,value)
pin: Número del pin.
value: HIGH o LOW.

PORTx : B00001111; : Sirve escribir valores lógicos en un puerto.
x: El nombre del puerto.
Después de la B van ocho dígitos (ceros o unos) que corresponden al nivel de cada pin del puerto.

digitalRead():Lee el valor de un pin digital especificado, ya sea HIGH o LOW.
Sintaxis= digitalRead(pin)
pin: El número del pin digital que desea leer (int).

Retorna o devuelve: HIGH o LOW. Si el pin no está conectado a nada, digitalRead () puede devolver HIGH o LOW (y esto puede cambiar aleatoriamente).

Resistores De PullUp y PullDown:



DESARROLLO PRÁCTICO:

1. Implementar en protoboard todo lo necesario (hardware y software) para que mediante un pulsador y Arduino se accione una carga A.C. (un bombillo). El funcionamiento debe ser el siguiente: El bombillo inicia apagado, una vez se presione el pulsador el bombillo enciende y se queda encendido por 3 segundos, pasados los 3 segundos el bombillo se apaga y queda listo para que se vuelva a oprimir el pulsador.
2. Implementar en tarjeta universal todo lo necesario (hardware y software) para que mediante un pulsador y Arduino se accione una carga A.C. (un bombillo). El funcionamiento debe ser el siguiente: El bombillo inicia apagado, una vez se presione el pulsador el bombillo enciende y se queda encendido por 3 segundos, pasados los 3 segundos el bombillo se apaga y queda listo para que se vuelva a oprimir el pulsador.

CRITERIOS DE EVALUACIÓN

Los aspectos que se tendrán en cuenta para la valoración final de la práctica son:

1. Disciplina durante el desarrollo de la práctica.
2. Vocabulario y respeto.
3. Uso del celular durante la práctica.
4. Estética y correcto funcionamiento en los circuitos implementados.
5. Cumplimiento del objetivo de la práctica en el tiempo acordado.
6. Herramientas y materiales necesarios para desarrollar la práctica.
7. Uso de casaca, guía de práctica y formato de seguimiento a prácticas.