UNIVERSIDAD SAN CARLOS DE GUATEMALA
ARQUITECTURA DE COMPUTADORAS Y ENSAMBLADORES 1
SECCIÓN B
PRIMER SEMESTRE 2021

PRÁCTICA 2

201801263 Audrie Annelisse del Cid Ochoa
201700499 Grethel Vilchez Suarez
201700569 Keila Avril Vilchez Suarez
201731087 José Fernando Guerra Muñoz

MENSAJES

Para el método mensaje se utilizó una variable global llamada velocidad la cual seria usada para el movimiento del texto con un valor de 150 y se utilizó la librería <LiquidCrystal.h> para el uso de LCD en proteus.

```
#include <LiquidCrystal.h>
```

Para indicar los pines que utilizará el LCD se hizo con la siguiente línea de código.

```
LiquidCrystal lcd(13,12,11,10,9,8); //RS,E,D4,D5,D6,D7
```

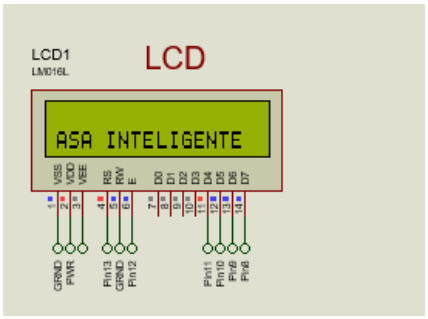
Se realizó una función la cual se muestra a continuación, la cual tiene de parámetro una string llama texto lo cual es el mensaje a mostrar en el LCD, se creó una variable de tipo int la cual almacenara el tamaño del mensaje a mostrar, para poder desplazar todas las letras del texto se creo un ciclo, usamos el .clear para limpiar el lcd, el .setCursor este sirve para poner las coordenadas del LCD donde queremos que inicie nuestro texto, el .print el cual nos imprime el texto en el LCD y el delay es la velocidad de como pasara el texto, se realizó otra ciclo el cual recorre el tamaño del texto con el objetivo que al estar desplazando este pase letra por letra hasta que termine en todo el LCD, utilizamos .substring se utiliza para devolver el texto.

```
String mostrarMensaje (String texto){
 const int tamano_texto = texto.length();
 for(int i=16;i>=1;i--)    //ciclo para mostrar el texto por toda la pantalla dezplanzando
 {
   lcd.clear();
   lcd.setCursor(i, 1); //coordenada de posicion
   lcd.print(texto); // el texto
   delay(VELOCIDAD_TEXTO);// la espera
 }
 for(int i=1; i<=tamano_texto ; i++)
 {
   String mensaje = texto.substring(i-1);//devuelve el texto
   lcd.setCursor(0, 1);
   lcd.print(mensaje);
   delay(VELOCIDAD_TEXTO);
   lcd.clear();
 }
 return texto;
}
```

Se realizó un método para poder mostrar el mensaje inicial el cual solo se manda a llamar la función anteriormente descrita con el texto que se deseó mostrar tal y como se muestra a continuación.

```
void mensajeInicio(){
    mostrarMensaje("CASA INTELIGENTE ACE1");
    delay(VELOCIDAD_TEXTO);
    lcd.clear();
    mostrarMensaje("ACE1-B-G4-S1");

}
```

Su funcionalidad en proteus se veria de la siguiente manera.

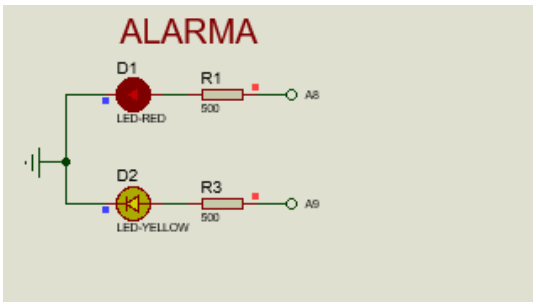


ALARMA

Se creó una variable global de tipo int la cual será igual al valor de milisegundos que se quería para que la alarma se active, adicional se creó una variable de tipo int dentro del método igualada a 0 la cual sirvió dentro de un bucle el cual este detendrá hasta que la variable tiempo sea igual al tiempo de alarma con el objetivo que en el transcurso de ejecución del bucle se activen los leds en proteus y dentro del LCD salga el mensaje de “Acceso no autorizado” para esto se activo los dos pines del Arduino los cuales darán voltaje a los led esto fue escribien pinmode, digitalwrite para encender llamando el numero de pin coma high y para apagar se usó el low y cada vez en su tiempo de ejecución a la variable tiempo se le sumaba 300 hasta llegar el valor de tiempo de alarma.

```
void ActivarAlarma(){

    int tiempo = 0;
    while(tiempo < TiempoAlarma){
      pinMode(A8, OUTPUT);
      pinMode(A9, OUTPUT);
      digitalWrite(A8, HIGH); // valor alto,LED ON
      digitalWrite(A9, HIGH);
      lcd.setCursor(3, 0); //coordenada de posicion
      lcd.print("Acceso No");
      lcd.setCursor(3, 1); //coordenada de posicion
      lcd.print("Autorizado");
      delay(150); // espero
      digitalWrite(A8, LOW);
      digitalWrite(A9, LOW);//valor bajo, LED OFF
      delay(150); // espero
      tiempo = tiempo + 300;
    }
}
```

En proteus se ve de la siguiente manera al ser activada, los se apagan y se enciende intermediaramente.



SENSOR ULTRASONICO DE ENTRADA Y DE SALIDA

IMAGEN CREACION DE VARIABLES BOOLEANAS

```
bool sensorultrasonico=true;
bool sensorultrasonicosalida=false
```

Como primera instancia se crearon dos variables booleanas del sensor, una servirá para la entrada que es verdadera y la de salida que es false. Mas adelante se explicará para que se utilizaron estas variables y cuál es su función.

IMAGEN CREACION DE VARIABLES PARA USO DEL SENSOR ULTRASONICO

```
long tiempo=0; //Donde se va a guardar el tiempo de duración del pulso generado por el pin Ech
long distancia=0; //Donde se va a guardar la distancia calculada
int pinTrigger=A10;
int pinEcho=A11;
int pinTriggerS=A13;
```

En la siguiente imagen se presentan cinco declaraciones de variables a continuación se especificará el uso de cada una:

Tiempo: Esta variable es para guardar el tiempo de duración por cada pulso generado por el pin Echo.

Distancia: Esta variable es para guardar la distancia calculada.

pinTrigger: En esta variable se va a guardar el pin del Arduino que se conecta con el pin del sensor ultrasónico de entrada.

pinEcho: En esta variable se va a guardar el pin del Arduino que se conecta con el pin del sonsorUltrasonico de entrada

PinTriggerS: En esta variable se va a guardar el pin del Arduino que se conecta con el pin del sonsorUltrasonico de salida.

pinEchoS: En esta variable se va a guardar el pin del Arduino que se conecta con el pin del sonsorUltrasonico de salida.

IMAGEN METODO DEL SENSOR ULTRASONICO DE ENTRADA

```
void sensorUltrasonico(){
    digitalWrite(pinTrigger,LOW); //S
    delayMicroseconds(5);//Restardo d
    digitalWrite(pinTrigger,HIGH); //
    delayMicroseconds(10); //retardo
    digitalWrite(pinTrigger,LOW);//Se
    tiempo=pulseIn(pinEcho,HIGH);//Se
    distancia=tiempo*0.0343/2; //Calc
    //Impresión en el LCD
    if (distancia < 150) {

     lcd.clear();
     mensajeinicial=false;
     sensorultrasonico=false;
     contra=true;
     password();
    }else {
      sensorultrasonico=true;
      mensajeinicial=true;
      lcd.clear();

    }

//delay(50);
}
```

En este método es donde se realizará toda la función que tiene el sensor ultrasónico, en la primera línea tenemos es para asegurarnos que el pin Trigger empiece en cero, a continuación, se hace un retardo de 5 microsegundos. Establecemos en alto el pin del trigger para comenzar el pulso de inicio del sensor con un retardo de 10 microsegundos que es el tiempo mínimo para inicializar el trigger del sensor. Se establece en bajo el pin del trigger para terminar el pulso de inicio del sensor y se inicia la función para que mida el tiempo del pulso generado por el Echo del sensor, definimos la ecuación para pasar las pulsaciones a centímetros. Debemos establecer una condición que es si la distancia es menor 150 entrara al if para establecer la inicialización del teclado de la práctica, de no ser así la variable sensor ultrasónico es verdadera y se sigue mostrando el mensaje inicial de la casa.

IMAGEN SENSOR ULTRASONICO DE SALIDA

```
void sensorUltrasonicoSalida(){
    digitalWrite(pinTriggerS,LOW); //Se asegura un cero en el pir
    delayMicroseconds(5);//Restardo de 5 microsegundo
    digitalWrite(pinTriggerS,HIGH); //Se establece en alto el pir
    delayMicroseconds(10); //retardo de 10 microsegundos (tiempo
    digitalWrite(pinTriggerS,LOW);//Se establece en bajo el pin c
    tiempo=pulseIn(pinEchoS,HIGH);//Se inicia la función pulseIn
    distancia=tiempo*0.0343/2; //Calculo de distancia a la cual s
    //Impresión en el LCD
    if (distancia < 150) {
      segundaPasada=true;
      primeraPasada=false;
      PassRight=false;
   //   sensorUltrasonicoSalida=false;
     //mov2();
     //mov1();
     lcd.clear();
     //lcd.setCursor(3, 0); //coordenada de posicion
     //lcd.print("SALIDA");
     //codigode motor de salida
    }else {
  //    sensorultrasonico=true;
  //    mensajeinicial=true;
  //    lcd.clear();

    }

//delay(50);
}
```

Para la parte del método del sensor ultrasónico de salida realizamos las mismas funciones que el del método de sensor ultrasónico de entrada con la diferencia que las condiciones cambian, las condiciones de este método es que si la distancia es menor a 15º0 se empezara a girar el motor que es la simulación de una puerta de salida a un giro de 90 grados.

IMAGEN METODO SETUP

```
void setup(){
  lcd.begin(16,2); //LCD (16 COLUMNAS Y 2 FILAS)

//  pinMode(A0,OUTPUT); //TRUE PASSWORD CORRECTO LED YELLOW.
  //pinMode(A1,OUTPUT); //FALSE PASSWORD INCORRECTO LED RED.
  pinMode(pinEcho,INPUT); // Configuración del pin 6 como entrada
  pinMode(pinTrigger,OUTPUT); //Configuración del pin 7 como salida
   pinMode(pinEchoS,INPUT); // Configuración del pin 6 como entrada
  pinMode(pinTriggerS,OUTPUT); //Configuración del pin 7 como salida
  m1.attach(pin);// asigna el pin
  m2.attach(pin111);// asigna el pin
}
```

En nuestro método setup hacemos las configuraciones para los pines de entrada y salida del sensor ultrasónico y del Arduino, definimos los pines de estrada de Echo y el pin de Trigger como salida esto es para el sensor ultrasónico de entrada, para el sensor ultrasónico de salida definimos los mismos pines antes mencionados.

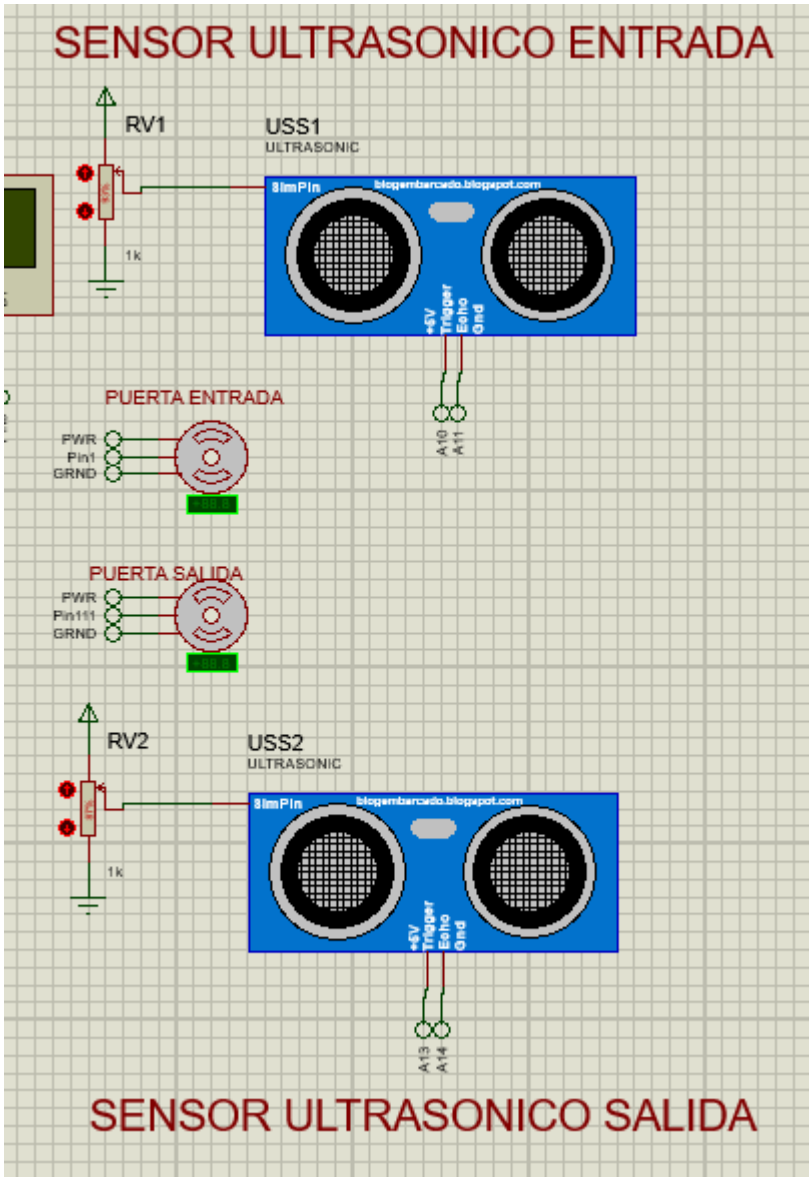
IMAGEN METODO DE LOOP

```
void loop(){
  //sensorUltrasonicoSalida();
  if(mensajeinicial==true){
    mensajeInicio();
  }
  if(sensorultrasonico==true){
    sensorUltrasonico();
  }
  if(contra==true && activacionAlarma==false){

    password();
  }
  if(mostrarmensaje==true){
     lcd.clear();
     lcd.setCursor(3,0);
     lcd.print("BIENVENIDO A");
     lcd.setCursor(3,1);
     lcd.print("CASA ^_^");
     mostrarmensaje=false;
  }
  if(primeraPasada==false && PassRight==true){
    mov2();
    delay(300);
    mov1();
    delay(80);
    primeraPasada=true;
  }
  if(PassRight==true){
    CheckControlLight();
  }
  if(sensorultrasonicosalida==true){
     sensorUltrasonicoSalida();
  }
```

Para finalizar con la programación de nuestro sensor ultrasónico mandamos a llamar los métodos ya mencionado con sus respectivas condiciones para su funcionamiento óptimo.

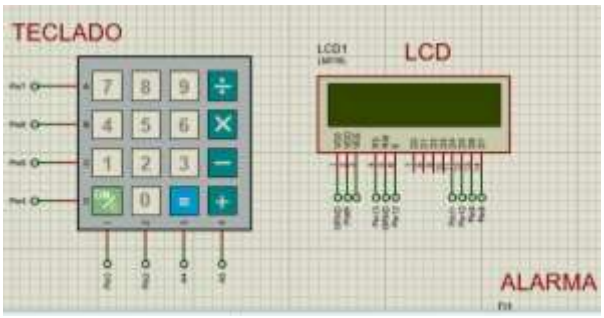
A continuación, se presenta el esquema que se utilizo en proteus para la creación del sensor ultrasónico.

IMAGEN SENSOR ULTRASONICO EN PROTEUS



Los materiales que utilizamos fueron dos sensores ultrasónicos uno para la entrada y otro para la salida, se utilizó un potenciómetro que es quien medirá la distancia por cada pulsación que genere nuestro sensor ultrasónico, cada uno conectado a tierra y a su poder.

TECLADO



Para la verificación de contraseña por teclado de uso la librería de proteus keypad-smallcalc y LM016L para la pantalla LCD donde se mostrarán los mensajes, el keypad se manejó como una matriz

```
int a=0, b=0, c=0, d=0, e=0, g=0;//acumuladores de datos enteros para la contrseña.
int var=0; //incremento apara el switch.
int C1=2, C2=0, C3=2, C4=1, C5=0, C6=4;//contraseña....Ustedes pueden codificarlo la contraseña
char f='*';  //caracter para cubrir la contraseña.
int veces=0,incorrecto=0; //seguridad de solo 3 intentos para ingresar la contraseña correcta.
int aviso=3; //aviso para mostrar los intentos como seguridad para el usuario.
const byte filas = 4; //cuatro  filas.
const byte columnas = 4; //cuatro columnas.
char tecla[filas][columnas] = {
  {'7','8','9','A'},
  {'4','5','6','B'},
  {'1','2','3','C'},
  {'*','0','#','D'}
};

byte pinFilas[filas] = {7, 6, 5, 4}; //conectarse a las patillas de salida de fila del teclado.
byte pinColumnas[columnas] = {3, 2, A4, A5}; //conectarse a las patillas de las columnas del teclado.

Keypad keypad = Keypad( makeKeymap(tecla), pinFilas, pinColumnas, filas, columnas );
LiquidCrystal lcd(13,12,11,10,9,8); //RS,E,D4,D5,D6,D7
```

Declaramos las variables numéricas a, b, c, d, e y g para guardar los números ingresados por el usuario la variables var para verificar el incremento de números ingresados y las variables C1 a C6 para guardar la contraseña, luego la variable veces para guardar los intentos y la cantidad incorrecta, el número de oportunidades que le quedan y la cantidad de columnas y filas de la matriz del teclado, luego definimos la estructura matricial del teclado, y los pines de acceso, por ultimo declaramos los controles que usaremos en este caso Keypad y LiquidCrystal.

El Método para verificar la contraseña “password()” vamos a usar el controlador para capturar cada pulsación de la matriz y guardarla en la variable key, si key no es nula posicionamos el cursor en la posición de 5 más la cantidad de caracteres ingresados para imprimir el asterisco, luego convertimos el carácter a entero según el código ascii y verificamos la cantidad de caracteres ingresados y la almacenamos en su variable respectiva

```
void password(){
  char key = keypad.getKey();
  if (key){

     lcd.setCursor(5+var,1);
     lcd.print(key),lcd.setCursor(5+var,1),lcd.print(f);//imprimimos el caracter en el lcd
     key=key-48; //COVERCIÓN DE CHAR A ENTEROS -48 SEGUN EL CÓDIGO ASCII.
     var++; //var se incrementa para los case1,case2,case3,case4.
     switch(var){
        case 1:
          a=key; //almacenamos primer dígito para la contraseña que seria el 1
        break;
        case 2:
          b=key; //almacenamos segundo dígito para la contraseña que seria el 2
        break;
        case 3:
          c=key; //almacenamos tercer dígito para la contraseña que seria el 3
        break;
        case 4:
          d=key; //almacenamos cuarto dígito para la contraseña que seria el 4
        break;
        case 5:
          e=key; //almacenamos cuarto dígito para la contraseña que seria el 4
        break;
        case 6:
          g=key; //almacenamos cuarto dígito para la contraseña que seria el 4
```

Si la cantidad de caracteres es 6 pasamos a verificar que cada carácter sea igual al carácter de la contraseña en esa posición de ser así imprimimos el mensaje de bienvenida y activamos el led de verificación positiva, de otra manera imprimirá el mensaje de error y pasaremos a el ingreso de otra contraseña para verificar

```
        case 6:
          g=key; //almacenamos cuarto dígito para la contraseña que seria el 4
          delay(100);
          if(a==C1 && b==C2 && c==C3 && d==C4 && e==C5 && g==C6){
              lcd.clear();
              lcd.setCursor(3,0);
              lcd.print("BIENVENIDO A");
              lcd.setCursor(3,1);
              lcd.print("CASA ^_^");
              PassRight=true;
              contra=false;
              //lcd.clear();
              movl();
              digitalWrite(A0,HIGH);
              //delay(700);
              digitalWrite(A0,LOW);
              //llamar metodo iluminacion

          }else{
              lcd.clear();
              lcd.setCursor(3,0);
              lcd.print("ERROR EN");
              lcd.setCursor(2,1);
              lcd.print("CONTRASEÑA");

              digitalWrite(A1,HIGH);
```

```
        digitalWrite(A0,LOW);
        //llamar metodo iluminacion

    }else{
        lcd.clear();
        lcd.setCursor(3,0);
        lcd.print("ERROR EN");
        lcd.setCursor(2,1);
        lcd.print("CONTRASEÑA");

        digitalWrite(A1,HIGH);
        delay(400);
        lcd.clear();
        digitalWrite(A1,LOW);
        //ontra=false;
    }
//------Seguridad para la contraseña y sus restricciones-------------------//

    if(a==C1 && b==C2 && c==C3 && d==C4 && e==C5 && g==C6){
        veces=0;//si es correcto el password ,variable veces no se incremeta.
        aviso=3;//variable aviso se mantiene en 3
    }else{
        veces ++; //incrementamos los intentos incorrectos de password para el bloqueo.
        aviso --; //decremento de variable aviso ,de 3 hasta 0 según las veces de fallas al ingresar el password.
        lcd.setCursor(2,0);
        lcd.print("LE QUEDA: ");
```

si el password es correcto hacemos un reinicio de oportunidades de lo contrario mostramos cuantas oportunidades le quedan si ya no le quedan oportunidades mostrara el mensaje de alerta y activara la alarma

```
    }else{
        veces ++; //incrementamos los intentos incorrectos de password para el bloqueo.
        aviso --; //decremento de variable aviso ,de 3 hasta 0 según las veces de fallas al ingresar el password.
        lcd.setCursor(2,0);
        lcd.print("LE QUEDA: ");
        lcd.setCursor(13,0);
        lcd.print(aviso);
        lcd.setCursor(2,1);
        lcd.print("OPORTUNIDAD");
        if(aviso==0){
            lcd.clear();
            lcd.setCursor(5,0);
            lcd.print("ALARMA");
            lcd.setCursor(4,1);
            lcd.print("ACTIVADO");
            contra=false;
            activacionAlarma=true;
           // ActivarAlarma();

        }
        delay(300);lcd.clear();
    }

    while(veces>=3){
       activacionAlarma=true;
       ActivarAlarma();
```

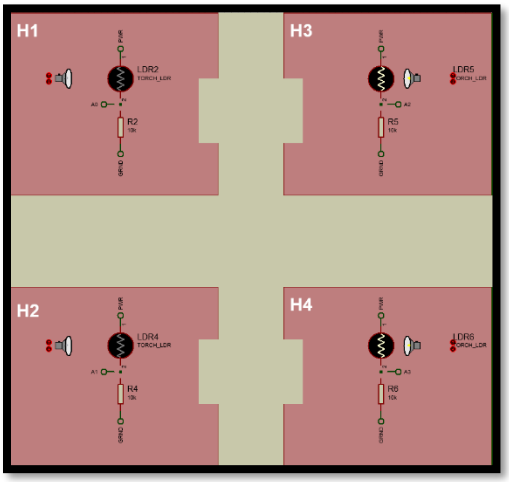
Mientras la alarma este activada el teclado permanecerá bloqueado y posteriormente se reiniciará y dará nuevas oportunidades

```
                contra=false;
                activacionAlarma=true;
               // ActivarAlarma();

            }
            delay(300);lcd.clear();
        }

        while(veces>=3){
           activacionAlarma=true;
           ActivarAlarma();
           contra=true;
           activacionAlarma=false;
           reinicio();
        }//while es Bucle infinito de seguridad para bloquear los re intentos del password

        var=0;
        lcd.clear();
      break;//se  termina el
    }
  }
  if(!key ){lcd.setCursor(0,0),lcd.print("INGRESE SU PASS");}//portada de inicio en el LCD
  delay(2);
}
//metodos del motor
```

Control de Luces

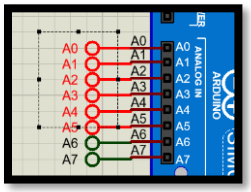
Componentes utilizados:

Para el control de luces se utilizaron los siguientes materiales en proteus:



- 4 Fotoresistencias LDR
- 4 Resistencias 10 k

Se utilizaron las entradas Análogas A0, A1,A2,A3.



Lógica utilizada:

Se utilizaron las siguientes variables.

CCLight: Variable booleana para activar motor si la contraseña es correcta.

Hi_n : Estado inicial de Luces

Ha_n : Estado actual de Luces

```
LiquidCrystal lcd(7, 6, 5, 4, 3, 2);
//Variables Control de Luces
bool CCLight=false;

//Habitaciones: Estado Inicial - Estados Actual
int Hi1=0; int Ha1=0;
int Hi2=0; int Ha2=0;
int Hi3=0; int Ha3=0;
int Hi4=0; int Ha4=0;
```

El propósito de estas variables es para comparar. En caso que haya algún cambio en el estado en el que se encontraban las luces al momento de hacer el chéquelo inicial del

control de luces y el momento actual. En caso que exista un cambio, se repetirá el chequeo.El control de luces se manejó con la siguiente función “CheckControlLight”.

```
void CheckControlLight(){
  Hi1=Ha1; Ha1=analogRead(A0);
  Hi2=Ha2; Ha2=analogRead(A1);
  Hi3=Ha3; Ha3=analogRead(A2);
  Hi4=Ha4; Ha4=analogRead(A3);
  if((Hi1+Hi2+Hi3+Hi4)==0 && (Ha1+Ha2+Ha3+Ha4)==0){ //Estado Inicial
    RecorrerHabitaciones();
  }else if(((Hi1-Ha1)+(Hi2-Ha2)+(Hi3-Ha3)+(Hi4-Ha4))==0){ //Ya hizo un recorrido y nada ha cambiado
    lcd.setCursor(1,0);
    lcd.print("Control Luces");
    lcd.setCursor(1,1);
    lcd.print("Finalizado   ");
    CCLight=true;
    delay(800);
  }else{  //Ya hizo un recorrido y hay cambios
    RecorrerHabitaciones();
    CCLight=false;
  }
}
```

Se utilizaron 3 condiciones:

1. Estado inicial: Todas las variables de comparación están en cero. Eso significa que si estas son sumadas su resultado será cero.
2. No hay Cambios: No hubo ningún cambió en el manejo de luces, el sistema para esperando nuevos cambios u otra instrucción.
3. Cambio de estado: Hubo un cambio de estado en alguna luz de las habitaciones. Eso significa que si se comparan las variables del estado anterior con las actuales, por medio de operaciones aritméticas, su estado será diferente de cero y se necesitará actualizar el control de luces.

La función “CheckControlLight”, hizo uso de la función “RecorrerHabitaciones”, para actualizar los estados de las luces en la pantalla lcd.

```
}
int ldr=0;
void RecorrerHabitaciones(){
  for (int i=1; i<=4;i++){
    if (i==1){    //Habitacion 1
      ldr=analogRead(A0);
      Ha1=ldr;
      lcd.setCursor(1,0);
      lcd.print("Habitacion 1:");
      if(ldr<=105){
        lcd.setCursor(1,1);
        lcd.print("Luz Apagada");
        delay(3000);
      }else{
        lcd.setCursor(1,1);
        lcd.print("Luz Encendida");
        delay(3000);
      }
    }else if (i==2){
      ldr=analogRead(A1);
      Ha2=ldr;
      lcd.setCursor(1,0);
      lcd.print("Habitacion 2:");
      if(ldr<=105){
        lcd.setCursor(1,1);
        lcd.print("Luz Apagada");
        delay(3000);
      }else{
        lcd.setCursor(1,1);
        lcd.print("Luz Encendida");
        delay(3000);
      }
```

Se utilizó la condición for para recorrer las habitaciones y se utilizó la función AnalogRead, para obtener el valor de las luces según el pin de entrada análogo que se le especificó.