Proyecto final.

Autor: Ahmed Ctatou Bougdadi.

Nombe: Alerta de proximidad para vehículos.

Tarea: Realización de un proyecto 1

1. Finalidad del sistema: explicar qué se pretende realizar, para qué sirve elsistema que queremos diseñar, y qué características debe tener. 2

2. Búsqueda de Información: información relevante que se haya buscado y usado para resolver el problema. 3

2.1. Software. 3

2.1.1. Funciones y llamadas a funciones. para realizar tareas repetitivas y para reducir el tamaño de un programa. Segmentar el código en funciones permite crear piezas de código que hacen una determinada tarea y volver al área del código desde la que han sido llamadas. 3

2.1.2. #define 4

Sintaxis 4

Código de ejemplo 4

Notas y advertencias 5

2.2. -Búsqueda de información harware 5

3. Hardware: esquema de entradas y salidas. lista de materiales, esquema de la protoboard y esquema electrónico.(FRITZING) 6

3.1. Montaje en fritzing. 6

3.2. Lista de mareriales. 7

3.3. Esquema electrónico 7

4. software 8

4.1. Software: diagrama de flujo 8

4.2. Funciones llamadas. 9

4.3. Software: Código comentado 11

5. Funcionamiento: fotos 18

6. Evaluación: qué funciona bien y que se puede mejorar. Problemas que hemos tenido y soluciones. Propuestas de mejora y ampliación de la actividad 20

1. **Finalidad del sistema: explicar qué se pretende realizar, para qué sirve elsistema que queremos diseñar, y qué características debe tener.** En este proyecto final montaremos un circuito que consiste en una alerta de proximidad para un vehículo cuando éste este maniobrando en distancia cortas menores de 30 cm hasta 0cm.Las exigencias que le impondremos al diseño de este proyecto serán las características siguientes:

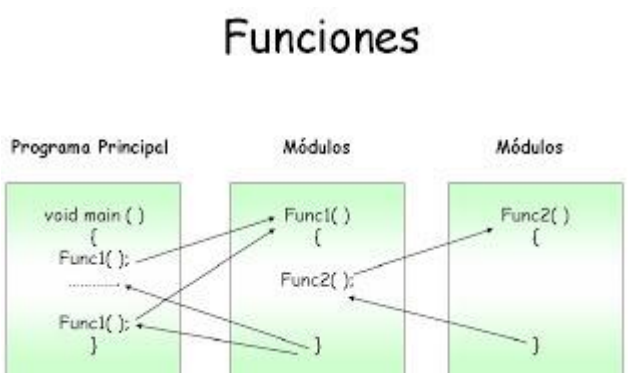
   Avisos tanto visuales como acústicos

   - Cuando se encienda el sistema y el vehículo esta a una distancia mayor o igual a 30 cm, en pantalla que salga el aviso "CUIDADO".
   - Cuando el vehículo se aproxime a una distancia mayor que 20 cm. Salga un mensaje que nos avisa que podemos proseguir "PROSIGA" Se encienda un led verde, suene el zumbador con un tono de 5000HZ durante 200ms.
   - En caso de que el vehículo se aproxime a una distancia mayor que 10 cm, salga un mensaje que nos avisa de que tengamos que tener precaución "PRECAUCIÓN" Se encienda un led amarillo , suene el zumbador con un tono de 2500HZ durante 200ms.
   - En caso que el vehículo se aproxime a una distancia este comprendida entre 0 cm y 10 cm salga un mensaje que nos avisa de que estamosen peligro "PELIGRO" Se encienda un led ROJO , suene el zumbador con un tono de 2000HZ durante 200ms.

2. Búsqueda de Información: información relevante que se haya buscado y usado para resolver el problema.

2.1. Software.

2.1.1. Funciones y llamadas a funciones. para realizar tareas repetitivas y para reducir el tamaño de un programa. Segmentar el código en funciones permite crear piezas de código que hacen una determinada tarea y volver al área del código desde la que han sido llamadas.

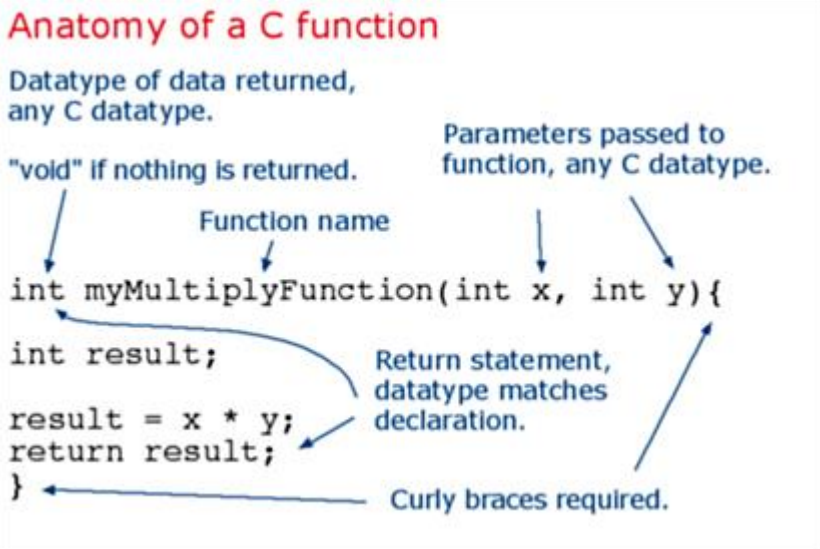


Las funciones se declaran asociadas a un tipo de valor. Este valor será el que devolverá la función, por ejemplo 'int' se utilizará cuando la función devuelva un dato numérico de tipo entero. Si la función no devuelve ningún valor entonces se colocará delante la palabra "void", que significa "función vacía"

Sintaxis:

```
1   tipo nombreFunción (parámetros) {
2      instrucciones;
3   }
```

Anatomía de una función en C:



Para llamar a una función, simplemente:

2.1.2. #define

#define es un componente C útil que le permite al programador dar un nombre a un valor constante antes de compilar el programa. Las constantes definidas en arduino no ocupan ningún espacio de memoria de programa en el chip. El compilador reemplazará las referencias a estas constantes con el valor definido en tiempo de compilación.

Sin embargo, esto puede tener algunos efectos secundarios no deseados, si, por ejemplo, un nombre de constante que ha sido #defined se incluye en alguna otra constante o nombre de variable. En ese caso, el texto se reemplazará por el número # definido (o texto).

En general, la palabra clave const es preferida para definir constantes y debe usarse en lugar de #define.

Sintaxis

```
#define constantName value
```

Tenga en cuenta que el # es necesario.

Código de ejemplo

```
#define ledPin 3 // The compiler will replace any mention of ledPin with the value 3 at compile time.
```

Notas y advertencias

No hay punto y coma después de la declaración #define. Si incluye uno, el compilador arrojará errores crípticos más abajo en la página.

```
#define ledPin 3; // this is an error
```

Del mismo modo, incluir un signo igual después de la declaración #define también generará un error de compilador críptico más abajo en la página.

```
#define ledPin = 3 // this is also an error
```

2.2. -Búsqueda de información harware

En los siguiente enlaces podremos ver el datesheet de todos los componentes usados:

HC-SR04 Sensor Ultrasónico :

http://bkargado.blogspot.com.es/2013/09/todosobrehc-sr04.html

Pantalla LCD:

https://www.sparkfun.com/datasheets/LCD/ADM1602K-NSW-FBS-3.3v.pdf

Zumbador TMB12A05:

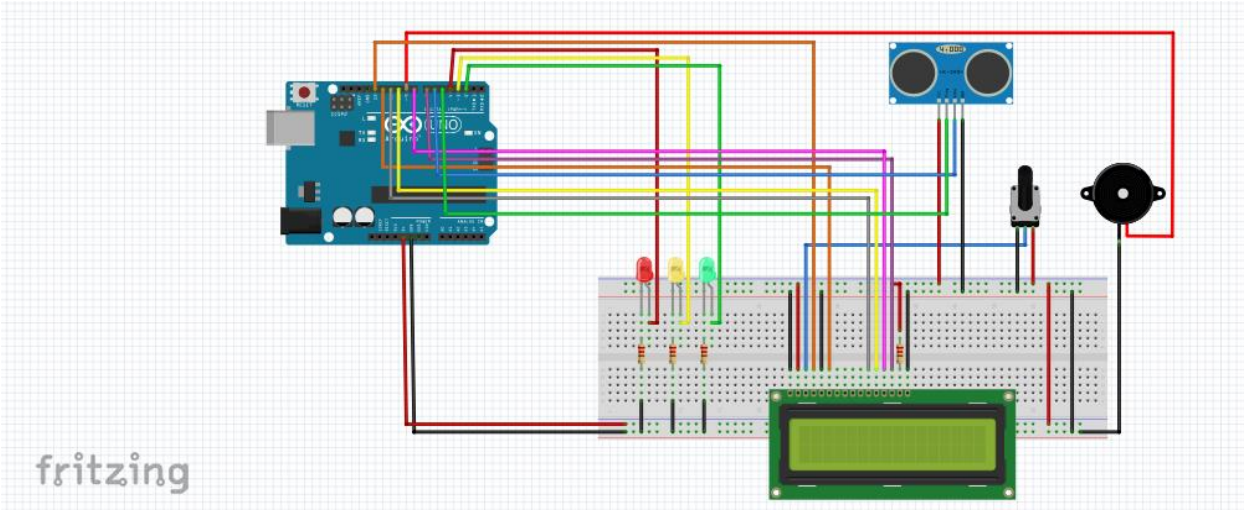
http://www.electronicaestudio.com/docs/SHT-TMB12A05.pdf

Potenciómetro:

http://www.mgelectronic.rs/ProductFilesDownload?Id=2864

3. Hardware: esquema de entradas y salidas. lista de materiales, esquema de la protoboard y esquema electrónico.(FRITZING)

3.1. Montaje en fritzing.



3.2. Lista de mareriales.

Placa arduino.

1 resistencias de 220 ohmios.

1 potenciómetro.

1 zumbador.

Pantalla LCD.

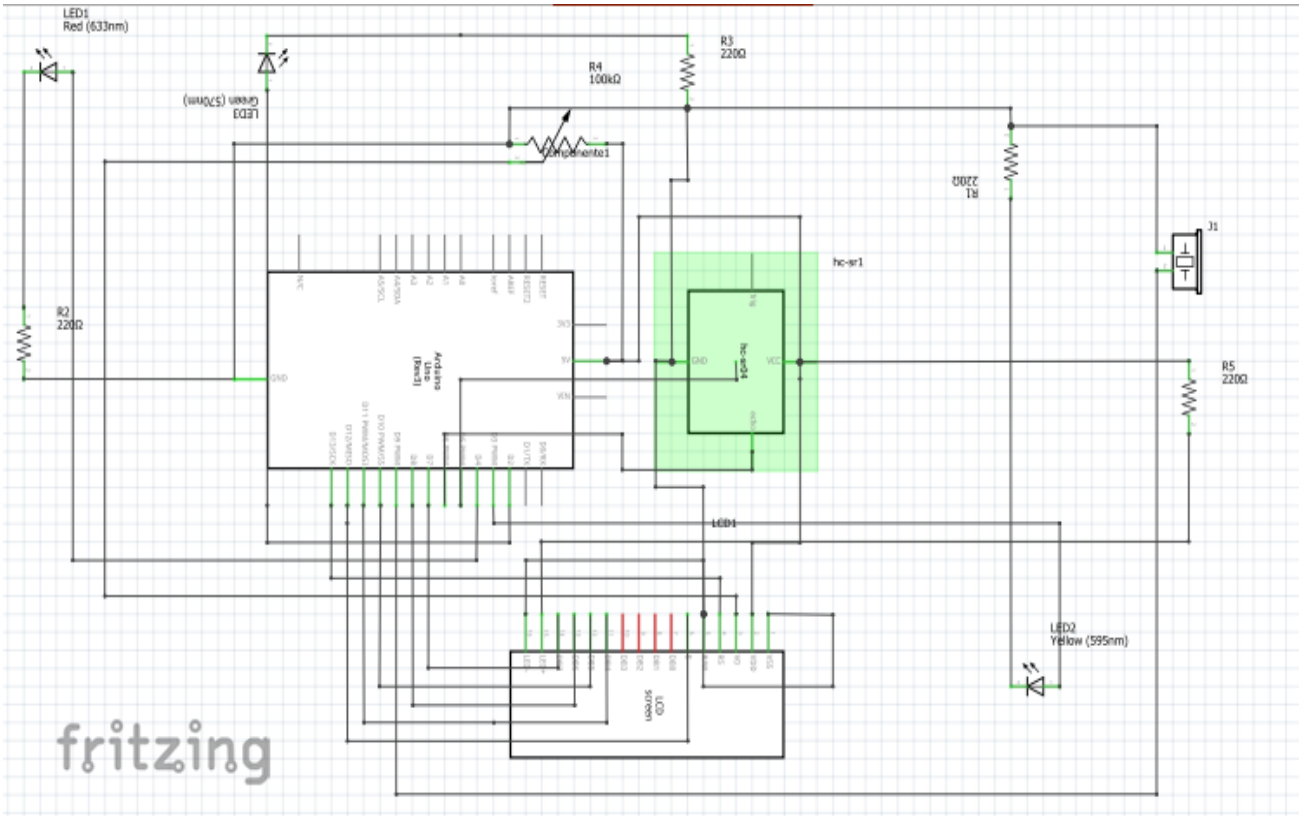
Sensor de distancia ultrasonidos.

2LEDs rojo, amarillo y verde.

Cables para puentear macho-macho y hembra-macho

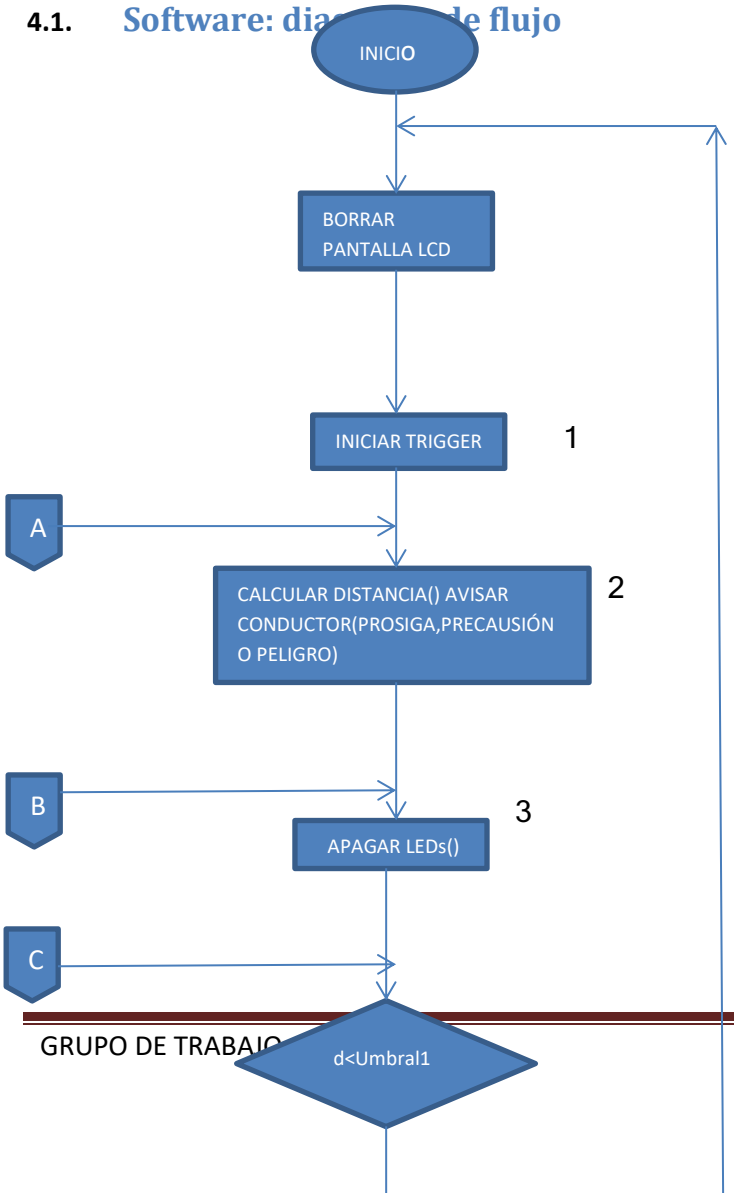
Para las características técnica ver el punto 2.2.

3.3. Esquema electrónico



4. software

4.1. Software: diagrama de flujo

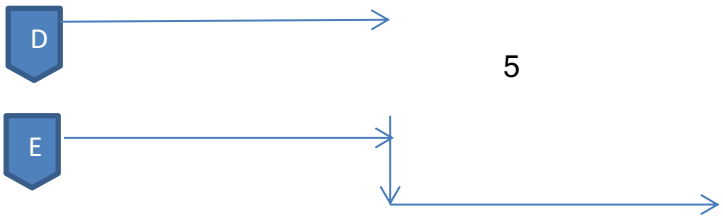


1. llamada a la función INICIAR TRIGGER()
2. llamada a la función CALCULAR DISTANCIA ()
3. llamada a la función APAGAR LEDs()
4. llamada a la función ALERTA DISTANCIAS().
5. llamada a la función APAGAR LEDs()

6. La llamada a la función distancia() 2, devuelve el valor de la distancia para después ser testeado la condición 6

6

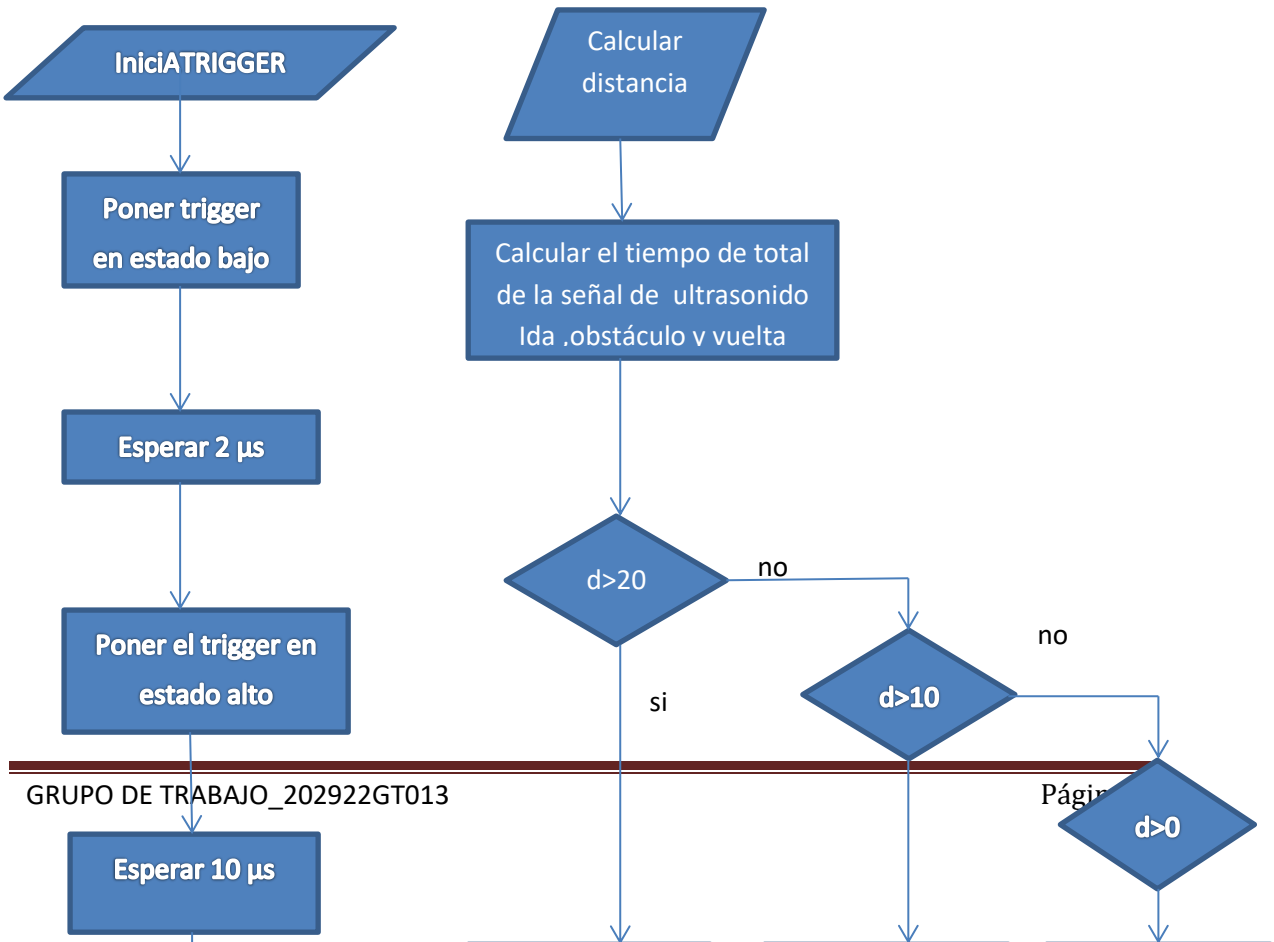
4



5

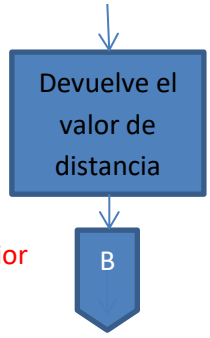
4.2. Funciones llamadas.

1. ***iniciarTrigger()***
2. ***calcularDistancia()***



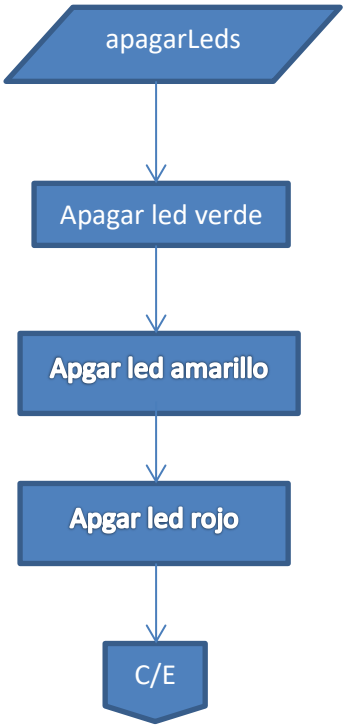
si

Sigue el la página anterior

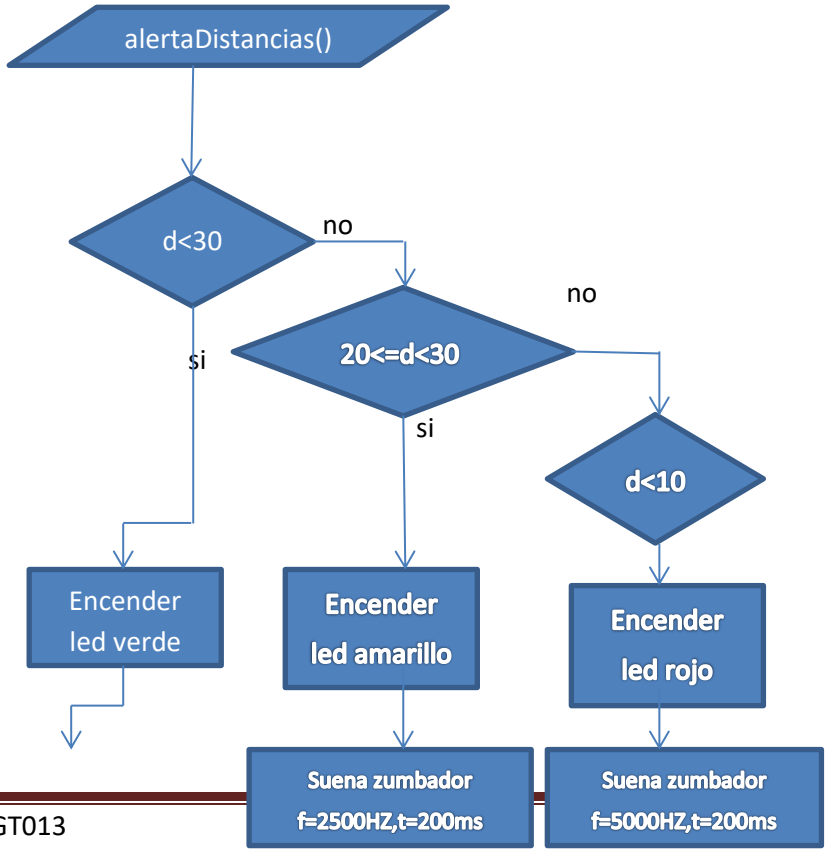


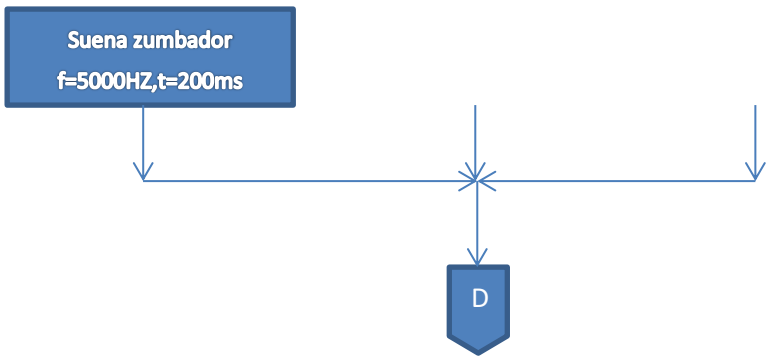
Sigue en la página anterior

3 y 5 apagarLeds()



4. AlertaDistancias()





4.3. Software: Código comentado

TÍTULO Y FINALIDAD DEL PROGRAMA

/*Este programa consiste en ALERTAR LA PROXIMIDAD DE UN VEHICULO con dos avisos,

uno sonoro y otro visual que nos indica que hay que hacer, proseguir, tener cuidado

o parar porque hay peligro de choque*/

INCLUIR LIBRERÍA PARA EL FUNCINAMIENTO DRE LA PANTALLA LCD

#include <LiquidCrystal.h>; /*incluir libreria para utilizar la pantalla LCD

No se ha utilizado la Pantalla LCD con integrado I2C. puesto que la tengo averiada*/

PINES UTILIZADOS

// Pines utilizados

#define LEDVERDE 2 /*SE PODRÍA HABER UTILIZADO const int LEDVERDE =2; PIN AL QUE

LED ESTA CONECTADO*/

```
#define LEDAMARILLO 3

#define LEDROJO 4

#define TRIGGER 5

#define ECHO 6

#define BUZZER 9
```

CONSTANTES

```
// Constantes

const float sonido = 34300.0; // Velocidad del sonido en cm/s

const float umbral1 = 30.0;

const float umbral2 = 20.0;

const float umbral3 = 10.0;
```

ASOCIAR PINES A LCD

```
LiquidCrystal lcd(13, 12, 11, 10, 8, 7); /*pines de la pantalla LCD

 inicializa la librería asociando los pines de la  interfaz LCD necesarios

 con el número de pin arduino al que está conectado*/
```

```
void setup() {

 Serial.begin(9600);      // Iniciamos el monitor serie

lcd.begin(16, 2); // configura el número de columnas(16) y filas(2) de la pantalla LCD

lcd.setCursor(1,0);  /*llamada a la función setCursor para colocar el cursor en columna

            segunda y primera fila */

lcd.print("CUIDADO"); //imprime el mensaje "cuidado el la pantalla LCD"
```

```
 delay(1000);
// Modo entrada/salida de los pines
 pinMode(LEDVERDE, OUTPUT);
 pinMode(LEDAMARILLO, OUTPUT);
 pinMode(LEDROJO, OUTPUT);
 pinMode(ECHO, INPUT);
 pinMode(TRIGGER, OUTPUT);
 pinMode(BUZZER, OUTPUT);
  // APAGAMOS TODOS LOS LEDS
  apagarLED();  //llamada a la función  apagarLED.    Ir a la pag 10
}
```

```
void loop() {

 lcd.clear(); //borrar datos de la pantalla

  //llamamos a esta función para preparamos el sensor de ultrasonidos
  iniciarTrigger();          Ir a la pag 14

        // Obtenemos la distancia
  float distancia = calcularDistancia(); /*Llamamos a la función calcular
  distancia()                    Ir a la pag 12

  // Apagamos todos los LEDs
  ApagarLEDs();/* a la función apagar apagarLEDs para que no haya confusión en
  en los avisos*/                                  Ir a la pag 10
```

```
/* Lanzamos alerta si estamos dentro del rango de peligro. el valor guardado en la
 la variable distancia de la linea 58,vemos si cumple la condición de que la
 la distancia sea menor que 30 cm   */
if (distancia < umbral1)
{
/*  en caso de que la distancia sea menor que 30 cm Lanzamos alertas,es decir llamamos
a la función alerta y le pasamos el parámetro (valor de distancia) */
```

alertas(distancia); /*llamada de esta función pasandole el valor de la distancia obtenido por la función calculardistancia() y guardado en la variabel distancia*/ ir pags 10 y 11

```
// Apaga todos los LEDs. función que apaga los leds, no devuelve ningún valor
void apagarLEDs()
{
 digitalWrite(LEDVERDE, LOW);
 digitalWrite(LEDAMARILLO, LOW);
 digitalWrite(LEDROJO, LOW);
}
```

FIN DE LA FUNCIÓN LLAMADA

```
// Función que comprueba si hay que lanzar alguna alerta visual o sonora
void alertas(float distancia)  /*función que recibe el valor  distancia declarado
como float para resevar memoria por si acaso fuera decimal,para después
evaluarlo. esta no devuelve ningún valor */
 if (distancia < umbral1 && distancia >= umbral2) //si 20<d<30
```

```
  tone(BUZZER, 2500, 200); /*el zunbador suena a una frecuencia de
2500HZ
                    durante una duración de 200 ms*/


 }
 else if (distancia <= umbral3) // sino y si d<10
{
 // Encendemos el LED rojo
 digitalWrite(LEDROJO, HIGH);
```

FIN DE LA FUNCIÓN LLAMADA

/* función que calcula la distancia a la que se encuentra un objeto. Devuelve una variable tipo float que contiene la distancia*/

float calcularDistancia()

{

// La función pulseIn obtiene el tiempo que tarda en cambiar entre estados, en este caso a HIGH

unsigned long tiempo = pulseIn(ECHO, HIGH);/*Espera que el pin echo sea sea alto inicia el cronometraje y luego espera que el pin echo 6 este bajo y se detiene .Este tiempo lo guardamos en la variable tiempo. Este tiempo es ,de ida y vuelta despué de que la señal de ultrasonido haya revotado contra el obstáculo, en

```
{
lcd.print("PROSIGA"); /*cuando la distancia es mayor de 20 cm aparece en la pantalla el mensaje deprosiga*/
 } else if(distancia > 10)
   { lcd.print("PRECAUCION");/*cuando la distancia esta comprendida entre 19,99cm y 10 cm aparece en la pantalla el mensaje de */
  }
 else if(distancia > 0)
 { lcd.print("PELIGRO"); /* cuando la distancia esta comprendida entre 9,99cm y 0 cm  aparece en la pantalla
                    el mensaje de peligro*/ }
 delay(500);
```

FIN DE LA FUNCIÓN LLAMADA

```
/* Función  que inicia la secuencia del Trigger para comenzar a medir,  no devuelve ningún
                                        valor*/

void iniciarTrigger()

        {

              // Ponemos el Triger en estado bajo y esperamos 2 microsegundo

digitalWrite(TRIGGER, LOW)

delayMicroseconds(2);
```

FIN DE LA FUNCIÓN LLAMADA

5. Funcionamiento: fotos

Ahora veremos unas imágenes del funcionamiento de como visualizamos en la pantalla LCD los mensajes y los led encendidos depende de la distancia y dentro de esta carpeta veremos el montaje de este proyecto en un video que muestra como se enciende cada LED a diferente distancia, el pitido del zumbador y el mensaje que aparece en la pantalla LCD.

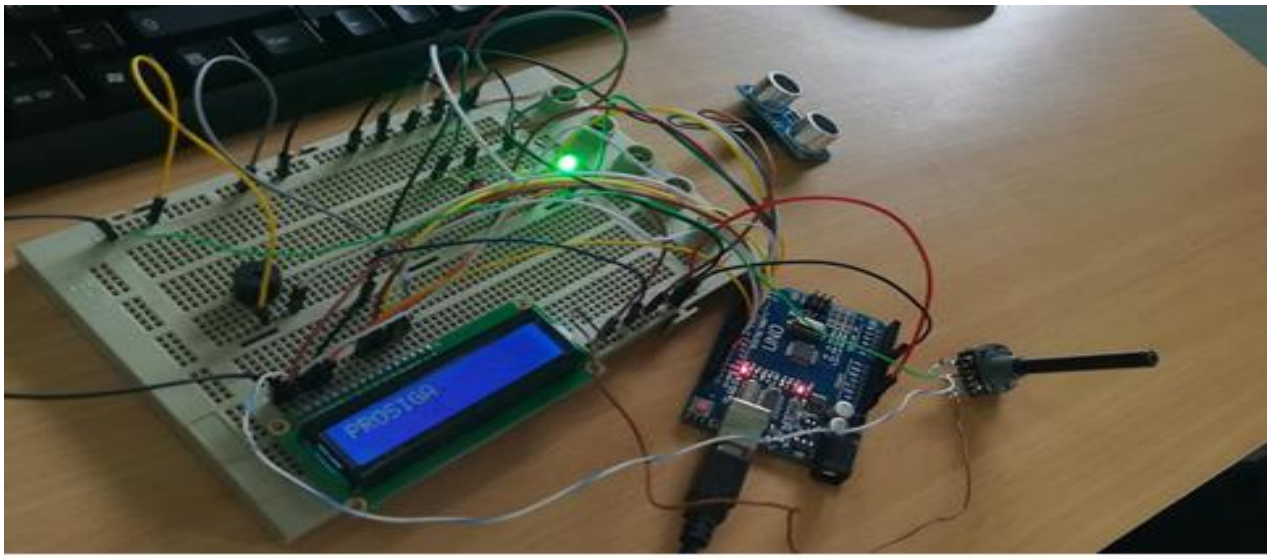


Fig. 1-Vemos el led verde encendido y el mensaje de prosiga en la pantalla lcd por la distancia que entre el sensor y la mano.

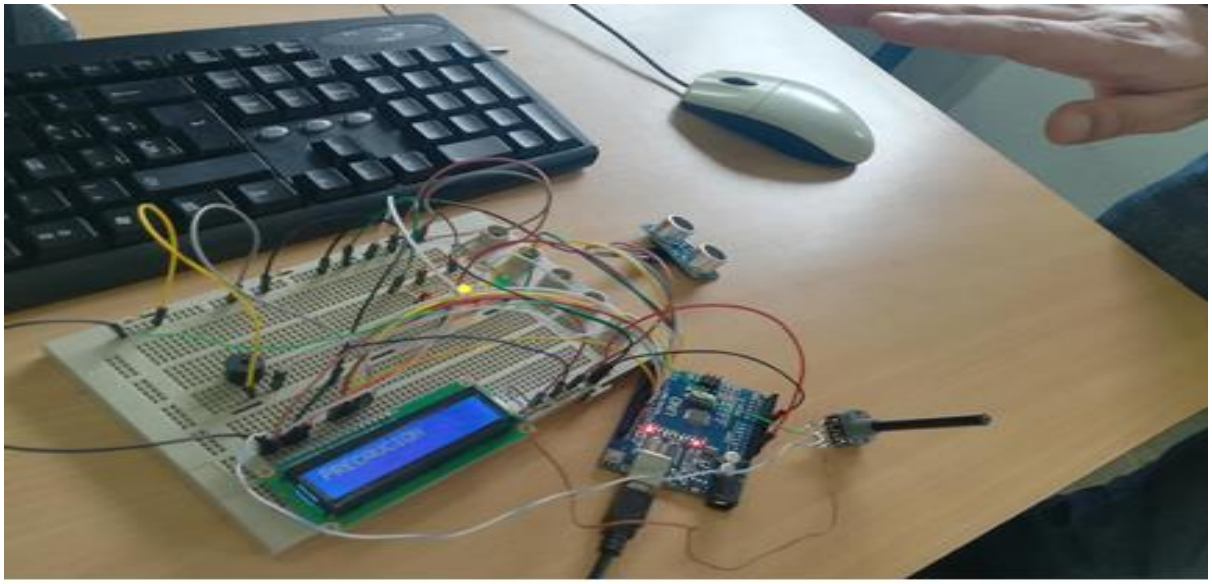


Fig.2- Vemos el led amarillo encendido y el mensaje de precaución en la pantalla lcd por la distancia que entre el sensor y la mano.

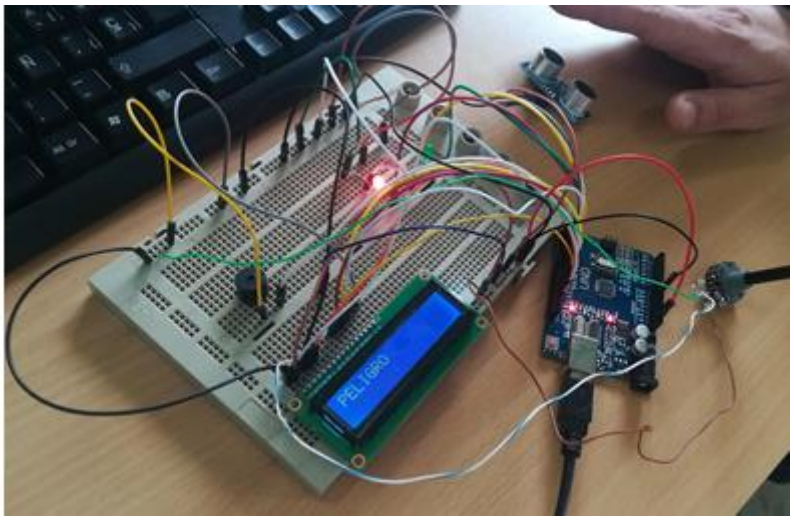


Fig.3Vemos el led rojo encendido y el mensaje de peligro en la pantalla lcd por la distancia que entre el sensor y la mano.

6. Evaluación: qué funciona bien y que se puede mejorar. Problemas que hemos tenido y soluciones. Propuestas de mejora y ampliación de la actividad

se podría mejorar de manera que se visualiza por la pantalla la distancia en cm.

También se podría utilizar 2 sistemas similares al presentado en este proyecto y añadir un par de sensores para detectar si el vehículo va hacia delante o hacia atrás y en función de esta decisión , que funcione un sistema u otro.

También hemos tenido un problema de funcionamiento con la placa protoboard que no funcionaba bien los componentes y se quedaba el led rojo siempre encendido y no aparecía mensaje alguno en la pantalla.