Practica 2 -Grupo #3

Laboratorio Arquitectura de compiladores y ensambladores 1 1er semestre 2021

Facultad de ingeniería

Randall Chriss Ramos Saucedo	201602869
Bryan Alexander Portillo Alvarado	201602880
Gabriel Alejandro García Meza	201700725
Oscar Roberto Velásquez León	201709144
Daniel Antonio Granados Azurdia	201709361
Juan Antonio Solares Samayoa	201800496

Introducción

El siguiente proyecto presenta una simulación para una casa inteligente utilizando Proteus para la simulación del hardware, un Arduino serie Mega y programación de Arduino, la practica consiste en la realización de un sistema de seguridad e iluminación controlado automáticamente mediante diferentes sensores que ayudaran a la detección de usuarios y otros factores para el disparo de eventos.

Materiales

- Una pantalla LCD AGM1232G
- Un Arduino MEGA
- Un sensor ultrasónico [4.0]
- Resistencias 330 ohm
- Fotorresistencias
- Un teclado de 9 segmentos

Código Arduino

A continuación, veremos a detalle el código implementado para resolver la practica uno, primeramente las librerías importadas para la realización del proyecto son las que se muestran a continuación.

```
#include<SoftwareSerial.h>
#include <Keypad.h>
#include <LiquidCrystal.h>
#include <Servo.h>
```

Utilizando SoftwareSerial.h

La biblioteca SoftwareSerial ha sido desarrollada para permitir la comunicación en serie en otros pines digitales del Arduino, utilizando software para replicar la funcionalidad (de ahí el nombre "SoftwareSerial"). Es posible tener múltiples puertos serie de software con velocidades de hasta 115200 bps. Un parámetro habilita la señalización invertida para los dispositivos que requieren ese protocolo.

KeyPad.h

Es una biblioteca para el uso de teclados de estilo *de matriz* con el Arduino. A partir de la versión 3.0 ahora es compatible con las pulsaciones de teclas mulitple.

LiquidCrystal.h

Esta biblioteca permite que una placa Arduino controle las pantallas LiquidCrystal (LCD) basadasen el chipset Hitachi HD44780 (o un compatible), que se encuentra en la mayoría de las pantallas LCD basadasen texto. La biblioteca funciona en modo de 4 u 8 bits (es decir, utilizando 4 u 8 líneas de datos además de rs, habilitar y, opcionalmente, las líneas de control rw).

Servo.h

Permite que las placas Arduino/Genuino controlen una variedad de servomotores.
Esta biblioteca puede controlar un gran número de servos. Hace un uso cuidadoso de los temporizadores: la biblioteca puede controlar 12 servos usando sólo 1 temporizador. En el Arduino Due puedes controlar hasta 60 servos.

Configuración de pines

LiquidCrystal (A0, A1, A2, A3, A4, A5)	PINES DE PANTALLA LCD
const int trigPin = 13;	PIN DE TRIGGER SENSOR ULTRASONICO
const int echoPin = 12;	PIN DE ECHO DE SENSOR ULTRASONICO
const int trigPinSalida = 14;	PIN DE SALIDA DE TRIGGER
const int echoPinSalida = 15;	PIN DE SALIDA DE ECHO
const int ledSalida = 17;	PIN DE SALIDA DE LED
pinMode(10, OUTPUT);	PIN PARA SALIDA DE ALARMA

Programación

Utilizando un paradigma de **programación funcional** debido a esto toda la solución se basa en la utilización de funciones específicas que se encargan de realizar las diferentes tareas y se mostraran a continuación.

- void pedirClave()
  Esta función despliega un mensaje y activa el protocolo para reconocer el teclado e ingresar los números correspondientes para verificar una clave guardada en memoria.
  Tipo: void
  Parámetros: NADA
- void activarAlarma()
  Esta función activa el protocolo de alarma que bloquea el sistema siguiendo los pasos establecidos para una mayor seguridad.
  Tipo: void
  Parámetros: NADA
- void mensajeInicial()
  Esta función despliega un mesaje por default, lo despliega en la pantalla lcd.
  Tipo: void
  Parámetros: NADA

- void lights_state()
  Esta función utiliza el protocolo establecido para cambiar el estado de las luces del sistema.
  Tipo: void
  Parámetros: NADA

- void change_state(int index)
  Esta función cambia el estado de las luces de manera específica otorgando un índice que representa el cuarto en el que se encuentra cada luz.
  Tipo: void
  Parámetros: Un índice de tipo entero INT

- void UltraSonico()
  Esta función activa el sensor ultrasónico que produce un echo y recibe el rebote para así calcular distancia y detectar si se tiene que hacer un cambio en el sistema.
  Tipo: void
  Parámetros: NADA

Formula de distancia del sensor ultrasónico

```
distancia = duracion * 340 / (2 * 10000);
```

Simulación del sistema en Proteus