Campana Química Inteligente

Universidad del Valle de Guatemala

Judah Sebastian Pérez Zeiset, Carnet no. 21536
Carlos Daniel Valdez Coreas, Carnet no. 21976

September 1, 2023

El proyecto consiste en crear una campana química que integra sensores y comunicación con la nube. Para esto se emplearon cuatro microcontroladores PIC16F887, un SoC ESP32, una pantalla LCD, un servomotor, un motor DC; los sensores DTH11 (temperatura y humedad del ambiente), LM35 (temperatura puntual) y MQ2 (gas inflamable y humo). También se implementó la comunicación entre microcontroladores con los protocolos I2C y UART, y con la nube a través de Wi-Fi.

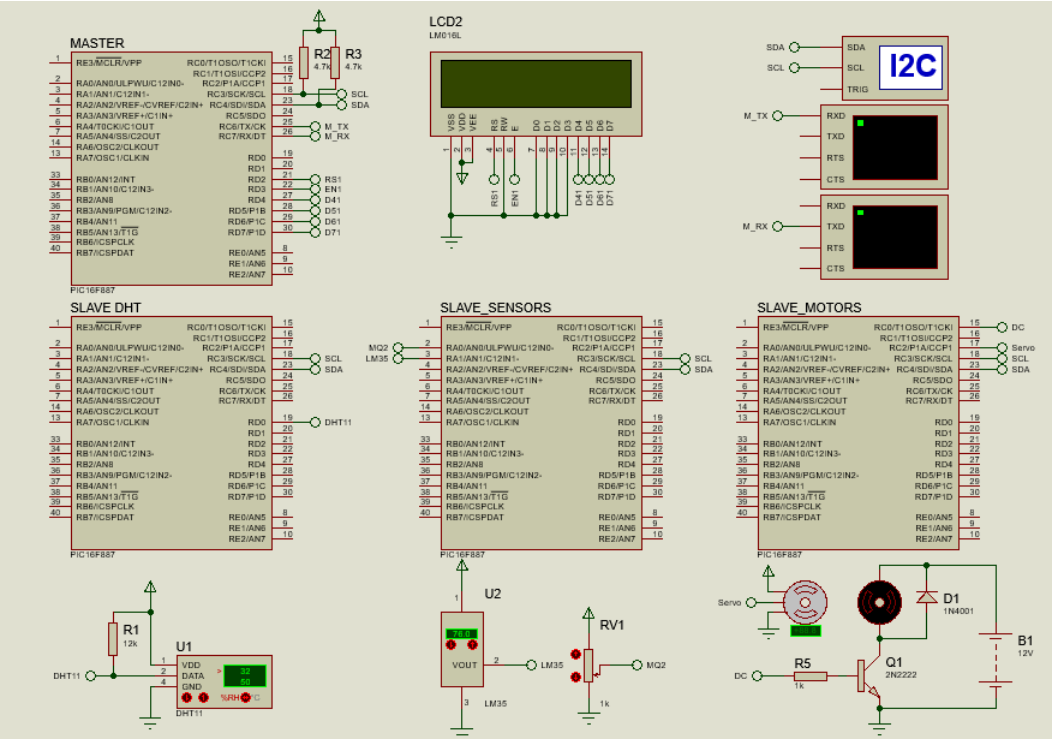

Figura 1: Circuito de la Campana Química Inteligente.

A grandes rasgos, los microcontroladores esclavos se ocupan de leer los sensores y controlar los motores; todos estos se comunican con el maestro, el cual solicita los valores de cada sensor y los despliega en una pantalla LCD, a su vez envía la información al ESP32. Finalmente el ESP32 envía los datos a la nube para ser desplegados en un panel de control de Adafruit IO. A continuación se detallan las tareas y funcionamiento de cada microcontrolador.

1 PIC Maestro

Este PIC se encarga de solicitar los valores de los sensores aproximadamente cada 300 ms, a excepción del sensor DTH11, el cual se solicita cada 2500 ms debido a que la lectura de este sensor tarda cerca de 2 segundos. Posteriormente envía el valor de cada sensor al ESP32 y a su vez los muestra en una pantalla LCD.

Este microcontrolador también actualiza el estado de los motores, continuamente sobreescribe la posición del servomotor de acuerdo al valor recibido desde el panel de Adafruit; a su vez revisa que la temperatura sea menor a 50 °C y la concentración del aire menor a 400 ppm, en dado caso que ambos superen estos límites se interpreta como que la Campana Química tiene llamas en su interior y envía la señal para activar la bomba de agua.

1.1 Registro MOTORCON

Este registro está dedicado al control de los motores, donde el nibble alto activa y desactiva la bomba de agua y el nibble bajo establece la posición del servomotor.

-	-	-	DC	-	Servo2	Servo1	Servo0
Motor DC				Servomotor			

bit 7-5 No implementado.
bit 4 DC: Control motor DC.
1 = Activar motor DC.
0 = Desactivar motor DC.
bit 3 No implementado.
bit 2-0 Servo<2:0>: Servomotor position.
0010 = Servo a 0 grados.
0011 = Servo a 90 grados.
0100 = Servo a 180 grados.
else = No implementado.

1.2 Comunicación

Para leer o escribir información a los microcontroladores esclavos utiliza el módulo MSSP configurado con el protocolo I2C. La lectura de cada sensor se realiza uno a la vez en el *loop* y sigue el siguiente formato:

- Se envía la señal de start.
- Se escribe la dirección del esclavo del cual se desea leer.
- Se escribe un caracter que indica qué dato es el solicitado.
- Se envía la señal de repeated start.
- Se espera a recibir el dato enviado por el esclavo.
- Si el dato requiere una conversión se realiza con la función de map.

Por otro lado, la comunicación con el ESP32 se realiza por medio del módulo EUSART configurado como UART. En este canal la información de los sensores es enviada en una larga cadena, iniciando con el carácter *new line* (0x0A) y separando cada uno de los datos con el carácter *space* (0x20).

1.3 Funciones

- requestTemp(): solicita la lectura de temperatura del sensor LM35 al PIC Sensores.
- requestGas(): solicita la lectura de gases del sensor MQ2 al PIC Sensores.
- requestHum(): solicita la lectura de humedad ambiente del sensor DHT11 al PIC DHT11.
- writeMotors(): envía el registro MOTORCON al PIC Motores.
- LDC_output(): despliega el valor de los sensores en la pantalla LCD.
- sendDataUART(): envía el valor de los sensores al ESP32.

- num_to_string(): convierte el valor numérico de un registro a una cadena que representa su valor en números decimales.
- map(): convierte un valor a su equivalente en una escala distinta, manteniendo proporcionalidad lineal entre los parámetros de la función.

2 PIC Esclavo Sensores

Este microcontrolador tiene la función de leer la información capturada por los sensores LM35 y MQ2, los cuales corresponden a temperatura y detección de gases combustibles. Ambos sensores son analógicos por lo que su lectura se obtiene a través del módulo ADC, alternando el canal de entrada del módulo cada vez se completa una lectura para leer el siguiente sensor.

2.1 Comunicación

Como se mencionó anteriormente, utiliza el protocolo I2C para enviar la lectura de los sensores al maestro. En el *loop* se verifica constantemente el valor que le fue solicitado y actualiza el registro *send_data* para enviar el dato correcto.

3 PIC Esclavo DHT11

Este PIC está dedicado a la lectura del sensor DHT11, el cual mide la temperatura y humedad del ambiente; a pesar haber sido posible implementarlo junto a los otros dos sensores, se optó por manejarlo por aparte por dos razones, siendo la primera que la lectura de este sensor requiere un lapso mínimo de 2000 ms, y la segunda que requiere del protocolo *single-wire two-way* para la transmisión de datos.

3.1 Comunicación

Su modo de operación es muy similar al PIC Sensores, con la excepción que tiene un contador para retrasar la lectura del DHT11; de esta manera se logra actualizar el dato a enviar sin verse afectado por los tiempos del sensor.

La lectura del sensor inicia con una señal de start y el DTH11 responde enviando pulsos de distinta duración para que el microcontrolador pueda distinguir y convertir la información a 0 y 1.

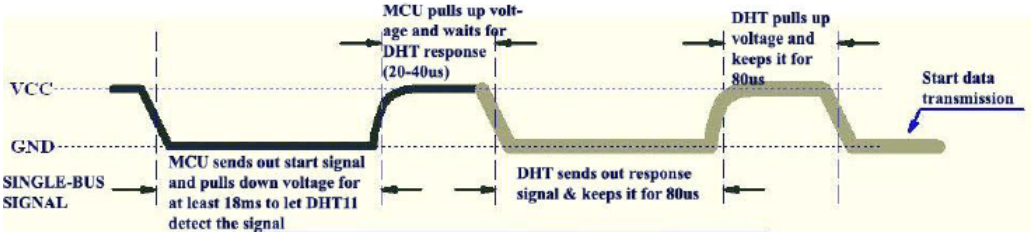

Figura 2: Comunicación serial entre el microcontrolador y sensor DHT11.

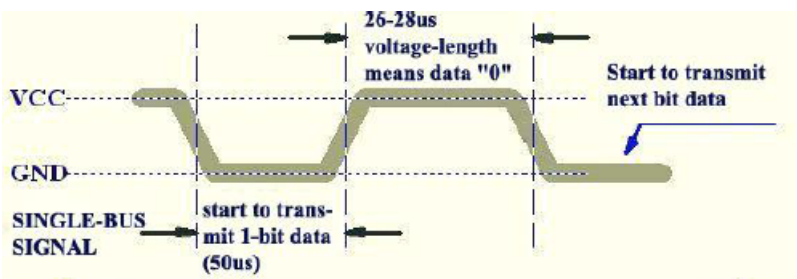

Figura 3: Ancho de pulsos interpretados como 0 y 1.

4 PIC Esclavo Motores

Este último PIC es el encargado de controlar los motores, de acuerdo con la información recibida desde el maestro. Para el control del servomotor se auxilia del módulo del Timer 0 para generar una señal PWM; y para el motor es simplemente una salida digital adjunta a un transistor que funciona como interruptor para activar al motor.

4.1 Comunicación

A diferencia de los otros dos esclavos, este PIC únicamente recibe datos del maestro. Luego de recibir una escritura, el *loop* se encarga de separar el registro por *nibbles*, para corresponder al control del motor DC y servomotor por separado.

4.2 Funciones

- initPWM(): inicializa el módulo del TMR0.
- angle_to_PWM(): controla la señal PWM para controlar el servomotor al ángulo establecido.

5 ESP32

Se utilizó el ESP32 con el framework de Arduino, con el objetivo de implementar el uso de la nube en el proyecto. El ESP32 recibe del PIC Maestro, por medio de comunicación UART, los datos correspondientes a las lecturas de cada sensor. Asimismo, el ESP32 envía un dato al PIC Maestro que indica la posición a la que se debe mover el servomotor que representa la puerta de la campana. Se utilizaron los pines RX2 y TX2 para la comunicación UART, además de un convertidor lógico digital bidireccional para poder operar entre los 5V del PIC y los 3.3V del ESP32.

Debido a que el PIC envía todos los datos separados por espacios, el ESP32 lee la cadena completa de datos y encuentra la posición de cada uno de los espacios. Conociendo la posición de cada dato en la cadena, se separa y se almacena en diferentes variables.

Para la implementación de Adafruit.io se utilizó el ejemplo creado por Todd Treece para Adafruit Industries. En el archivo config.h se describe el "username" y "key" del propietaro de los canales. Asimismo se configura el nombre y la contraseña de la red WiFi a la que se conectará el ESP32. En el archivo main se guardan los datos de las variables en su respectivo canal. Cabe mencionar que se utilizó la librería Ticker.h para crear una función que se ejecute cada diez segundos, la cual envía los datos a los canales de Adafruit respetando las limitaciones del mismo, ya que solamente se puede enviar un dato cada dos segundos para evitar el *throttling*. Para leer el slider que abre o cierra la puerta se utilizó la función "handleMessage", la cual cambia el estado de una variable booleana según sea el caso.

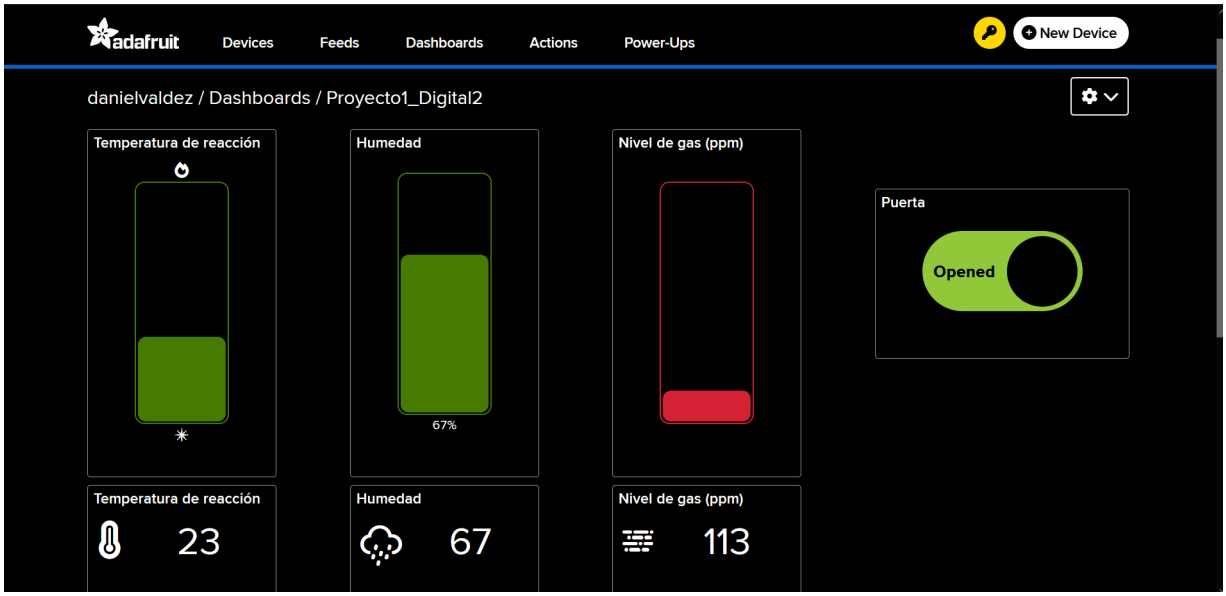

Figura 4: Dashboard de Adaf

6 Enlaces

- Video demostrativo: https://youtu.be/NTZnpFPvyLI?si=0j6IwjBXolnQrjfe
- Repositorio GitHub: https://github.com/Ritmop/CampanaQuimica_ProyectoElectronicaDigital2

7 Bibliografía

- Microchip Technology Inc.(2009). PIC16F882/883/884/886/887 Data Sheet.
- DHT11 Humidity & Temperature Sensor.(s.f.). List of Unclassified Manufacturer. Based on OSEPP Data Sheet.
- National Semiconductor Corporation.(2000). LM35 Precision Centigrade Temperature Sensors.
- Pololu Robotics & Electronics.(s.f.). MQ-2 Semiconductor Sensor for Combustible Gas.