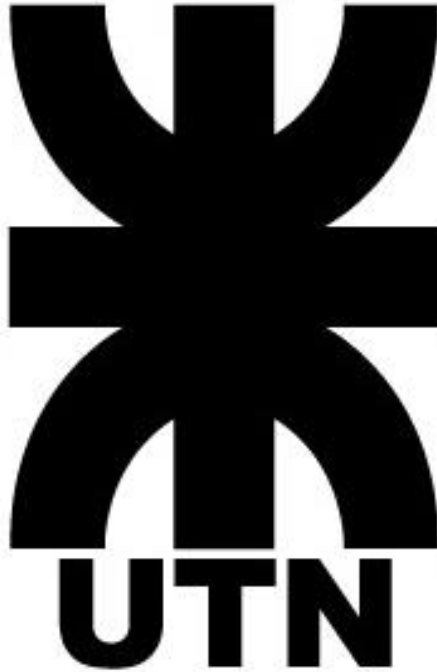


Universidad Tecnológica Nacional
Facultad Regional de Tucumán



Técnicas Digitales II
Sistema de seguridad con sensores de apertura y
comunicación Bluetooth

Carrera: Ing. Electrónica

Asignatura: Técnicas Digitales II

Año: 2024

Grupo N°: 8

Alumnos: Gao, Luciano

Jorrat, Tomás

Mitre, Emilio

Profesor: Ing. Mansilla Rubén Darío

Informe de Proyecto Final

1) Consideraciones sobre el hardware del proyecto:

Plataforma embebida: NUCLEO-STM32F413ZH

1.1) Descripción.

El proyecto tiene como objetivo implementar un sistema de seguridad con alarma, utilizando sensores magnéticos para detectar la apertura de puertas y ventanas y sensores PIR para detección de movimiento. El diseño implementa una placa de desarrollo STM32F413ZH comunicada mediante un módulo Bluetooth HC-05 con una terminal serial externa (Ej. app en un smartphone). El sistema se controlará mediante una entrada de teclado matricial de 4x3 y una pantalla LCD 16x2, que se comunica mediante un módulo adaptador I2C LCD basado en un controlador PCF8574. Una vez que el sistema se encuentre armado (alarma activada) si se activa uno o más sensores se envía una señal para el disparo de la alarma, la placa de desarrollo enviara una señal que provocara el sonido mediante una sirena conectada con una etapa de potencia.

Funcionamiento

En cuanto al funcionamiento del sistema, este cuenta con una interfaz de usuario que se visualiza en la pantalla LCD y se maneja mediante el teclado pudiendo elegir una opción con la tecla indicada en pantalla. La interfaz cuenta con un menú principal y una serie de submenús para acceder a las diferentes funciones, estas son.

Menú principal: Este es el menú que se muestra al iniciar el programa, estando la alarma desactivada, y al cual regresaremos por defecto en la mayoría de casos al salir de las demás pantallas y estados.



1.Activar alarma -> Nos muestra 2 opciones para activar la alarma, completo o sin el sensor PIR

2.Cambiar la contraseña -> Se debe ingresar la contraseña actual y luego una nueva

***.Mas** -> Nos muestra más opciones del menú principal, en este caso una opción extra para realizar una prueba de la alarma.

3.Prueba -> Simula el disparo de la alarma encendiendo la sirena durante 2 segundos, muestra en pantalla "Prueba de Alarma"

Activación: Al seleccionar la opción 1 del menú principal no mostrara un submenú para elegir de qué manera se activara la alarma.



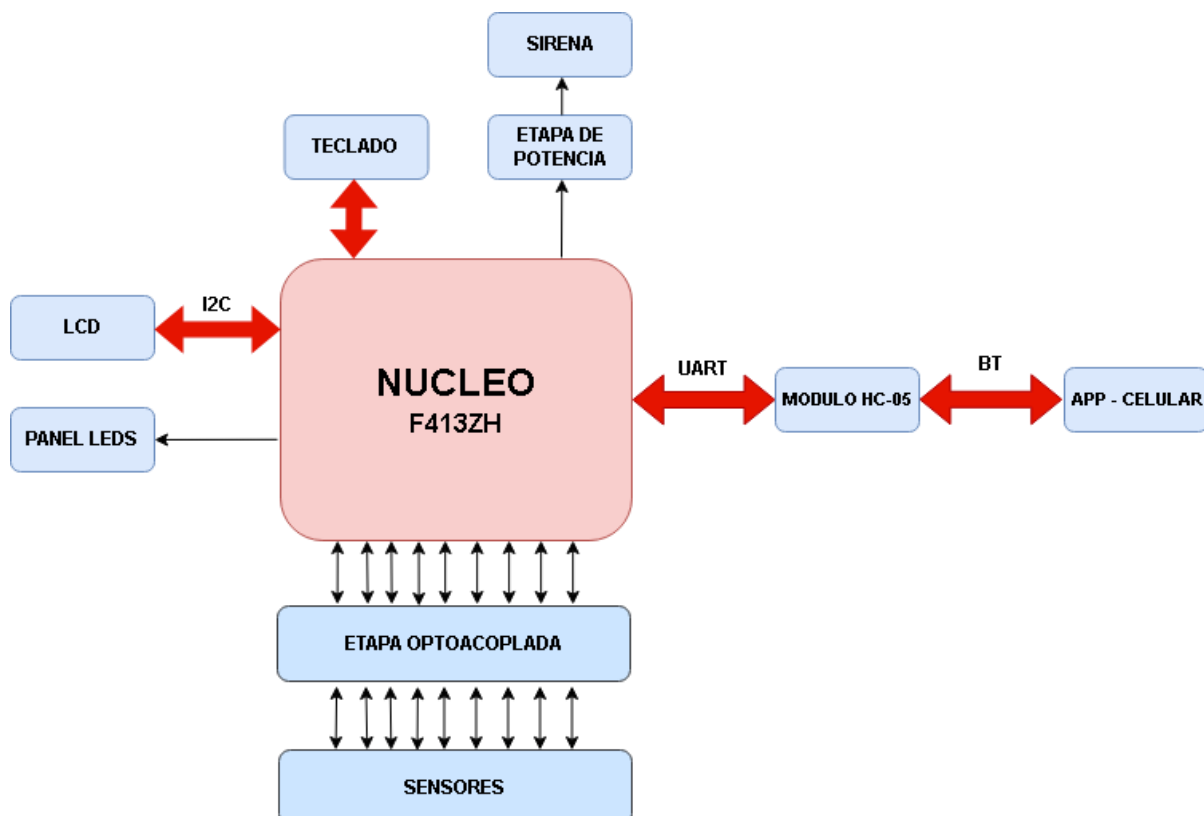
1. Sistema completo -> Incluye sensores magnéticos y sensor de movimiento PIR
2. Sistema sin sensor de movimiento -> Solo se incluyen los sensores magnéticos

Luego de elegir una opción se deberá ingresar la contraseña para que se active la alarma, tendremos 20 segundos antes de la activación en el cual se ignoran los sensores (para poder salir de la casa, por ejemplo). Al pasar los 20 segundos se mostrará en pantalla el mensaje "Alarma activa". En este estado se estará controlando el estado de los sensores.

Disparo: Estando la alarma activada, si se detecta la apertura de una puerta o ventana (sensor magnético) o se detecta movimiento (Si se seleccionó la opción con el sensor PIR), se produce el disparo de la alarma, la sirena comenzará a sonar y se deberá ingresar la contraseña correcta para desactivarla.

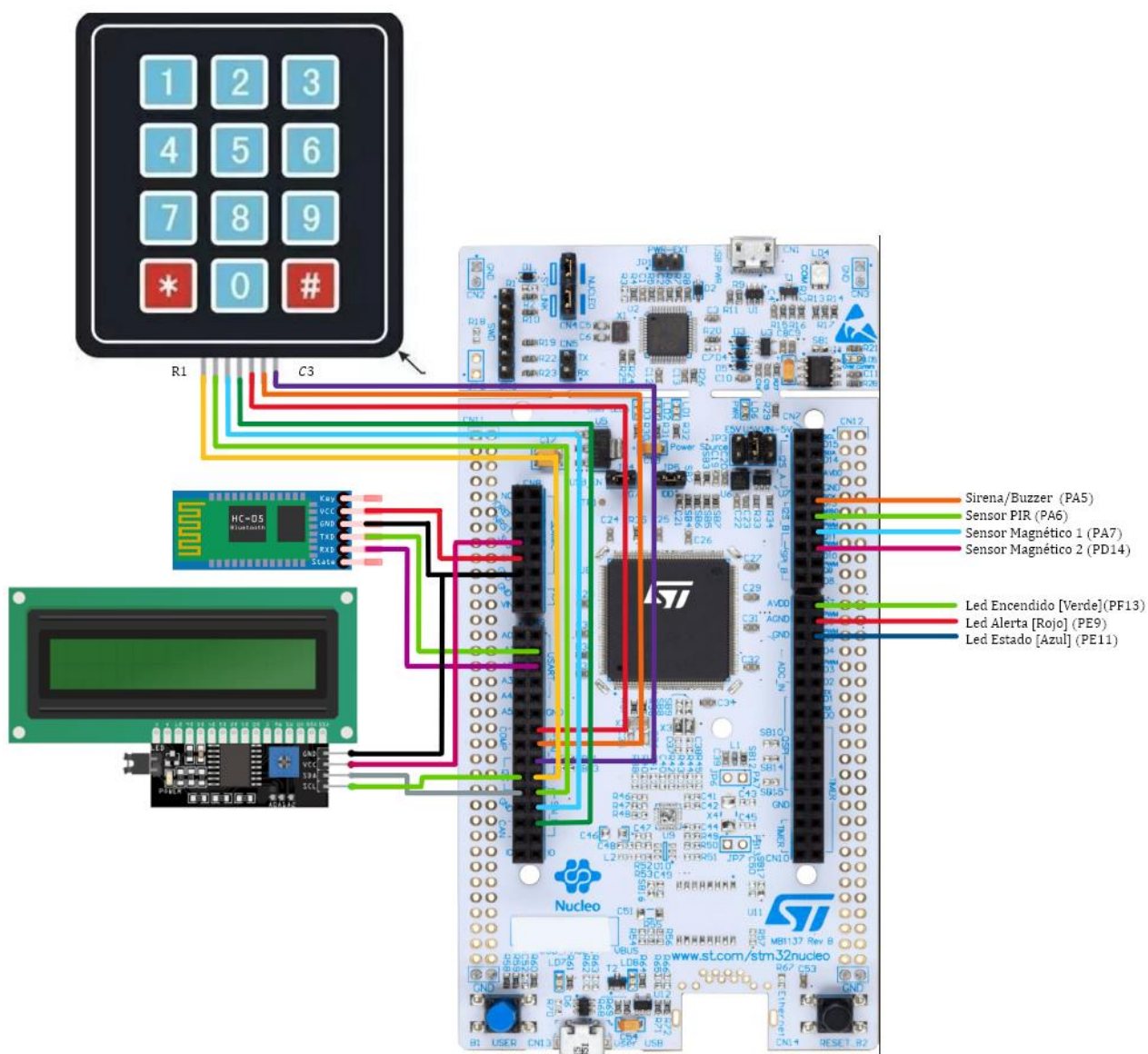
Diagrama de bloques:

El siguiente diagrama muestra los diferentes módulos/periféricos del proyecto y su interconexión.



1.2) Circuito del proyecto:

Las siguientes imágenes indican la conexión de cada periférico con la placa de desarrollo STM32F413ZH en los pines configurados para este proyecto. Los sensores y leds externos se encuentran en una PCB la cual se muestra su esquemático más abajo.



Pines de cada periférico:

Teclado 4x3:

Filas: R1(PE6) | R2(PE3) | R3(PF8) | R4(PF7)

Columnas: C1(PE2) | C2(PE4) | C3(PE5)

HC-05: Vcc=5V

TX->USART2_RX(PD6)

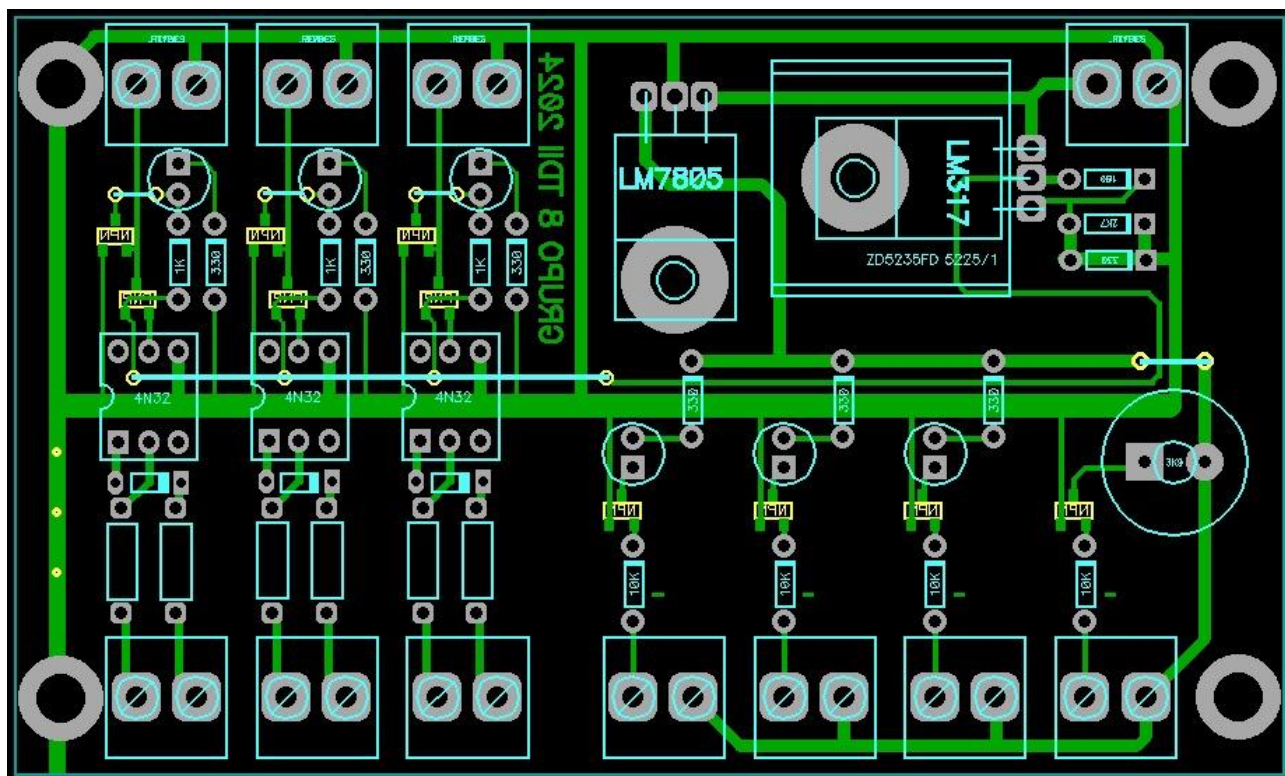
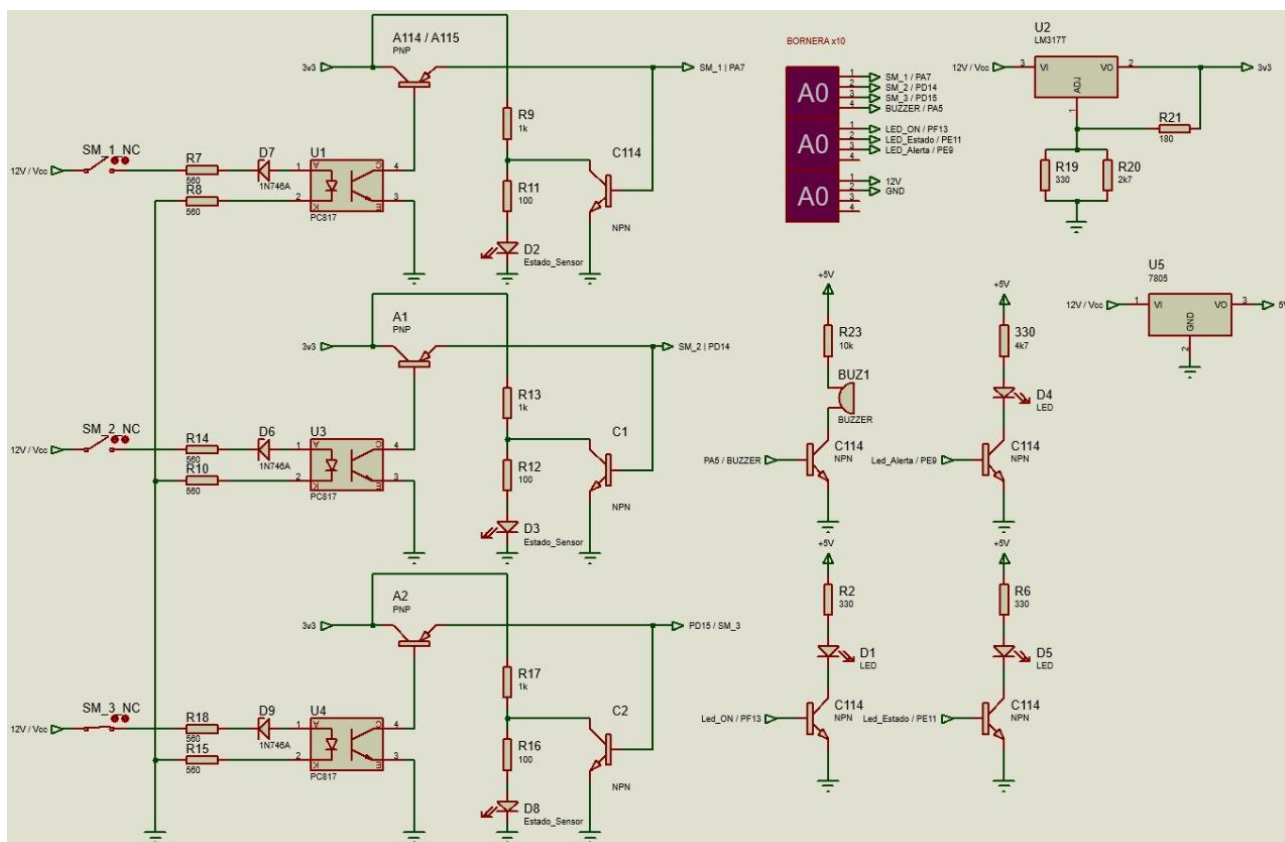
RX->USART2_TX(PD5)

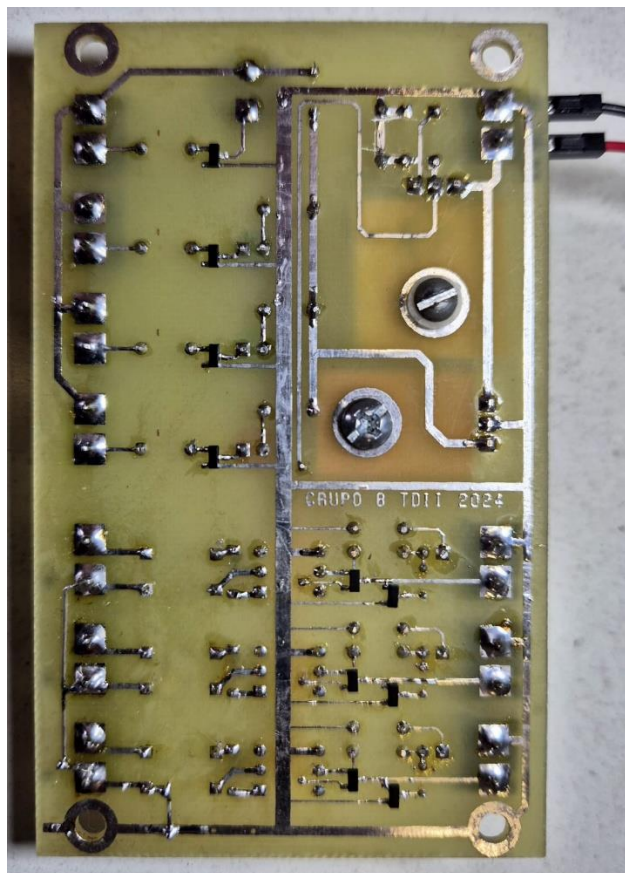
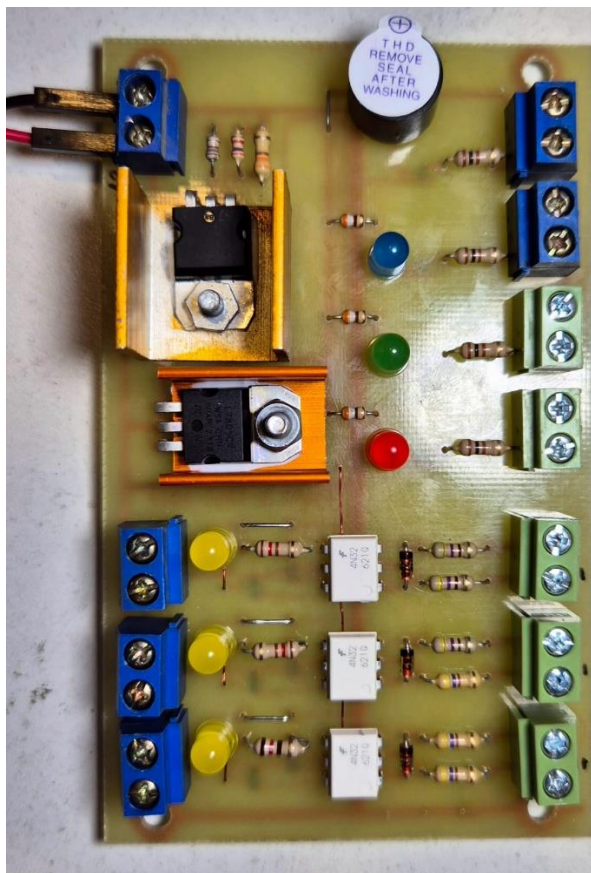
Display LCD - PCF8574: Vcc=3V3

I2C2_SDA(PF0)



I2C2_SCL(PF1)

Esquemático PCB externa: Se utilizan diferentes etapas de adaptación para la conexión de los sensores y leds externos hacia la placa de desarrollo.





1.3) Listado de componentes:

| Componente (Cantidad) | Imagen | Descripción |
|--------------------------|---|---|
| STM32F413ZH x1 |  | Placa de desarrollo Microprocesador ARM Cortex-M4 |
| HC-05 x1 |  | Modulo Bluetooth (UART) 4 pines Vcc=5V 9600 bps, 8N1 |

| | | |
|-----------------------|---|--|
| LCD 16x2 x1 |  | Display LCD 16x2 caracteres Vcc=3,3V-5V 16 pines |
| PCF8574 x1 |  | Modulo adaptador I2C Vcc=2,5-6V 4 pines Control regulable de backlight |
| Teclado 4x3 x1 |  | Teclado matricial de membrana 4x3 4 filas, 3 columnas |
| Sensor MS-14-BL x2 |  | Sensor Magnético Normal cerrado (cuando ambas partes están juntas) Vcc=3,3V |
| HC-SR501 x1 |  | Sensor PIR (Sensor de movimiento) Vcc=5-20V Salida TTL; 3V3 , 0V Ajuste de tiempo y sensibilidad Modos: Disparo único (L) o repetido (H) |

| | | |
|------------------|---|--|
| Buzzer x1 |  | Buzzer piezoeléctrico |
| Leds X3 |  | Se utilizan los leds como indicadores visuales |

2) Consideraciones de Software:

2.1) Repositorio AFP_5_GRUPO_8

https://github.com/TomasJorrat/AFP_5_GRUPO_8_2024

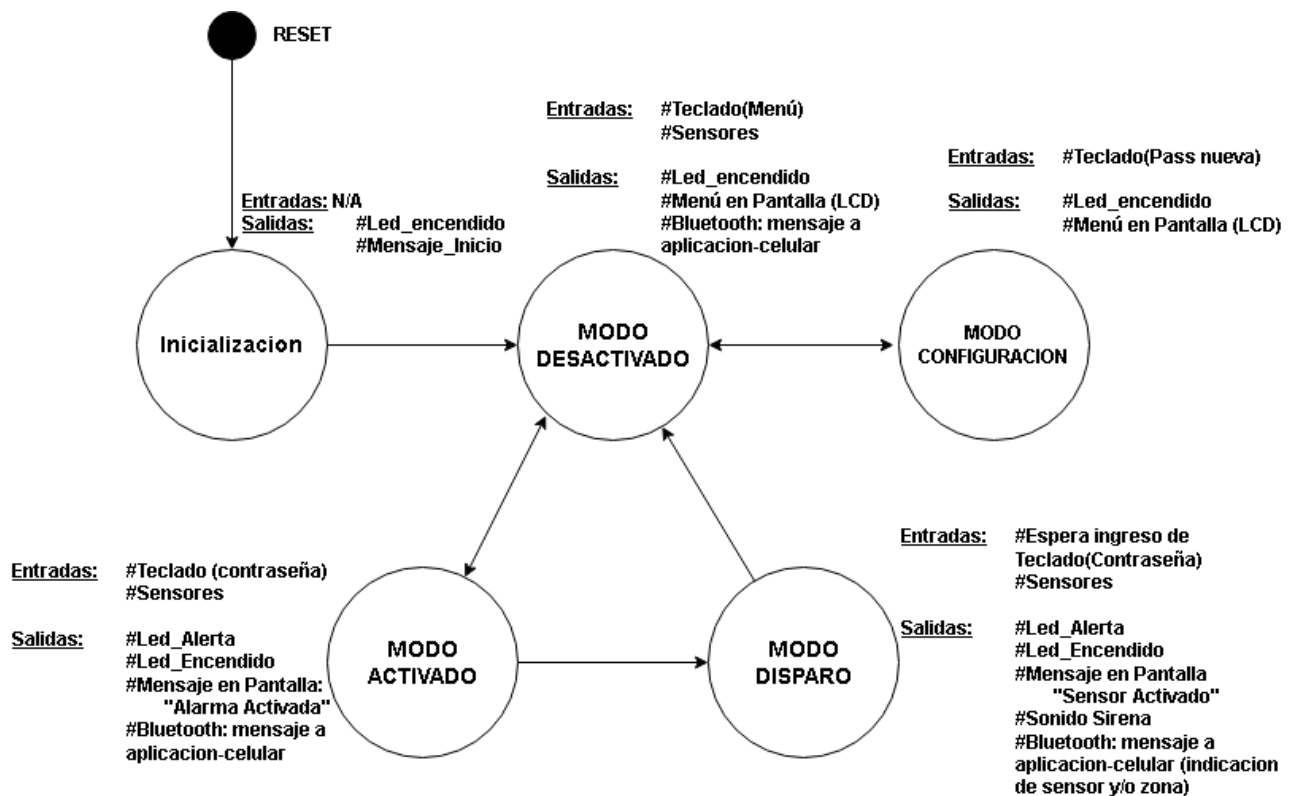
https://github.com/emiliomitre/Grupo_8_TDII_2024

Nombre del proyecto principal: AFP_5_GRUPO_8_Alarma

2.2) Funcionamiento de la aplicación:

La aplicación se implementó para una placa de desarrollo STM32-F413ZH utilizando sensores magnéticos y de movimiento PIR para detectar si hay un intruso, una pantalla LCD y teclado para el manejo de la interfaz de usuario, junto con un módulo bluetooth para comunicarse con una app en el celular y una sirena para dar una alerta sonora en caso de que se detecte apertura o movimiento.

El programa principal se basa en una MEF de 5 estados. El primer estado inicializa los módulos de software y luego ingresa en el bucle infinito del programa en el estado desactivado, mediante el uso del teclado, que en el programa se capturan como caracteres, podemos pasar de un estado a otro.



- **Inicialización:** En este estado se ejecuta las función que inicializa los módulos y funciones necesarias para el funcionamiento del programa, entre ellos los módulos GPIO, I2C, UART y los tiempos de retardo no bloqueante API_Delay. En este estado se enciende el led verde en la PCB y se envía un mensaje por bluetooth indicando si se establecio la conexión "HC-05 conectado con STM32". Se muestra en pantalla el menú principal una vez que se inicializan todos los módulos.

La MEF se manejo en el bucle infinito mediante una función SWITCH y los estados cambian según las opciones que seleccionemos en el menú mostrado en pantalla.

- **Modo Desactivado (MAIN_MENU):**

Se muestra en el LCD el menú principal con las opciones: 1. Activar, 2. Cambiar Pass, *. MAS. Y dentro del bucle infinito se escanea el teclado mediante la función "keypad_getkey()". El led verde permanecerá encendido en este estado.

Si se presionó una tecla y se capturo un carácter, la función "HandleMainMenuInput" se encarga de evaluar, cual opción se seleccionó y que debería hacer el programa.

1. Activar -> DisplayAlarmMenu();
2. Cambiar Pass -> DisplayChangePassMenu();
3. Mas -> HandleSubMenu();

Según que tecla se presione se mostrara otro menú o mensaje en pantalla y cambiara de estado la MEF.

- **Modo configuración (CHANGE_PASS_MENU):**

Al ingresar en este estado se pedirá la contraseña actual. La cual esta formada por 4 dígitos y por defecto es "1234", se debe presión "#" luego de ingresar los 4 dígitos para confirmar. Si la contraseña ingresada es incorrecta se volverá a pedir nuevamente, si pasan 20 segundos sin que se detecte una tecla se mostrara el mensaje "Tiempo expirado" y regresa al menú principal, si presionamos la tecla "*" también volveremos al menú principal. Al volver al menú principal se regresa al modo desactivado.

- **Modo Activado:**

- (ALARM_MENU)**

- Al seleccionar una opción en el sub "1.Activar" ya sea el sistema completo con todos los sensores o activar sin sensor de movimiento, se ingresa en este estado donde se pedirá ingresar la contraseña para activar la alarma.

- Si pasan 20 segundos sin que se detecte una tecla o se presiona "*", se volverá al menú principal. Si ingresamos la contraseña correcta el programa entra en la función "ActivateAlarm()" la cual nos muestra en pantalla que se activo la alarma y tenemos 20 segundos antes de que este armada, también se envía este mensaje mediante bluetooth.

- (ACTIVATE_ALARM)**

- Luego de que pasaron los 20 segundos y se armo la alarma, se mostrara en pantalla el mensaje "Alarma Activa" y se controlara el estado de los sensores, en caso de que se active uno o mas se disparara la alarma.

- Se puede desactivar sin que se dispare, se debe presionar una tecla y luego ingresar la contraseña. De ingresarla incorrectamente se disparara la alarma.

- **Modo Disparo (AlarmTriggered):**

- Se emite una alerta sonora mediante la sirena o buzzer, se envia un mensaje por bluetooth indicando el disparo y en pantalla se muestra el mensaje "ALERTA" además de pedirse ingresar la contraseña para poder desactivar la alarma.

- Una vez que se ingresa correctamente la contraseña para desactivar la alarma se regresa al modo desactivado, se envia un mensaje por bluetooth dando aviso y se muestra nuevamente en pantalla el menú principal.

2.3) Módulos de software:

Se incluyen en la carpeta Drivers/API los siguientes modulos de software para el manejo de la aplicación. Cada módulo incluye las inicializaciones respectivas de los periféricos que utiliza.

API_GPIO: Modulo de desarrollo propio para el manejo de los leds de usuario integrados en la placa y otros pines de requerirse.

API_Delay: Modulo de desarrollo propio que permite utilizar un delay no bloqueante mediante el uso del SysTick.

API_Debounce: Modulo de desarrollo propio que implementa una función de anti rebote para pulsadores.

API_BT: Modulo de desarrollo propio para el manejo de un modulo bluetooth HC-05, utiliza funciones de la HAL para el manejo de los puertos UART, en este caso específicamente para USART2. Contiene funciones que facilitan el envío o recepción de mensajes a través de los pines UART.

API_Keypad: Modulo de desarrollo propio para la lectura de un teclado matricial 4x3. Se deben configurar las filas como salidas y columnas como entradas para evaluar correctamente que tecla se presiona y capturar el dato.

API_LCD: Este modulo utiliza el puerto I2C2 para el manejo de un display LCD 16x2 conectado con un adaptador. Contiene funciones que permiten mover la posición del cursor y escribir en pantalla un mensaje.

2.4) Periféricos:

GPIO: Se utiliza para el manejo de leds, tanto de la placa de desarrollo como externos, el manejo de sensores magnéticos, sensor PIR y de la sirena o buzzer.

I2C2: Se utiliza para la comunicación con la pantalla LCD

USART2: Se utiliza en modo Asíncrono para la comunicación bluetooth con el modulo HC-05. Utiliza un baud rate de 9600 bps y una trama 8N1 en modo full dúplex.

3) Practicas de programación.

3.1) Ejemplo de variable global privada

https://github.com/TomasJorrat/AFP_5_GRUPO_8_2024/blob/016e67b179384246f129ea4ea23d04e76ee9e333/AFP_5_GRUPO_8_Alarma/Drivers/API/Src/API_Keypad.c#L12

3.2) Primera línea de la función de actualización de la MEF.

Las diferentes funciones de la MEF se manejan en main.c y el cambio de estado se produce dentro del bucle infinito del programa.

https://github.com/TomasJorrat/AFP_5_GRUPO_8_2024/blob/016e67b179384246f129ea4ea23d04e76ee9e333/AFP_5_GRUPO_8_Alarma/Core/Src/main.c#L161

3.3) Reglas de buena programación utilizadas.

El código compila sin errores ni warnings y no se utilizan funciones de control de flujo como goto.

Se restringe la visibilidad de la información de variables dentro de lo posible, como aquellas dentro los módulos de software que no se utilizan en main.c

https://github.com/TomasJorrat/AFP_5_GRUPO_8_2024/blob/016e67b179384246f129ea4ea23d04e76ee9e333/AFP_5_GRUPO_8_Alarma/Drivers/API/Src/API_LCD.c#L12

Se utiliza el preprocesador moderadamente, principalmente para los #include y #define

https://github.com/TomasJorrat/AFP_5_GRUPO_8_2024/blob/016e67b179384246f129ea4ea23d04e76ee9e333/AFP_5_GRUPO_8_Alarma/Core/Src/main.c#L24

Se controla el valor de retorno de las funciones non-void, como el caso de la función keypad_getkey que se almacena en la variable "char key" la cual se evalúa que sea diferente de nulo y posteriormente se evalúa su valor específico.

https://github.com/TomasJorrat/AFP_5_GRUPO_8_2024/blob/016e67b179384246f129ea4ea23d04e76ee9e333/AFP_5_GRUPO_8_Alarma/Core/Src/main.c#L163

https://github.com/TomasJorrat/AFP_5_GRUPO_8_2024/blob/016e67b179384246f129ea4ea23d04e76ee9e333/AFP_5_GRUPO_8_Alarma/Core/Src/main.c#L211

3.5) Un ejemplo de un comentario que explique el porque y no es el como es.

Un ejemplo de comentario que explica el porque y no el como es en la declaración de una variable
char newPassword[5] // Almacena la contraseña nueva

https://github.com/TomasJorrat/AFP_5_GRUPO_8_2024/blob/016e67b179384246f129ea4ea23d04e76ee9e333/AFP_5_GRUPO_8_Alarma/Core/Src/main.c#L66

4) Anexos:

Sensor PIR HC-SR501 - [Datasheet](#)

Bluetooth HC-05 - [Datasheet](#)

STM32 F413ZH – [Recursos y datasheet](#) - [PinOut](#)

LCD 16x2 - [Datasheet](#)

PCF8574 – [Recursos y Datasheet](#)

5) Bibliografía:

- Donald Norris, Programming with STM32, Getting Started with the Nucleo Board and C/C++.
- Carmine Noviello, Mastering STM32, 2018
- Martyn Currey, website, <https://www.martyncurrey.com>
- DeepBlue Embedded, STM32, <https://deepbluembedded.com/stm32-arm-programming-tutorials/>