Trabajo Sistemas Electrónicos Digitales

APLICACIÓN DOMÓTICA CON UN MICROCONTROLADOR: MICROONDAS

Curso 2024/2025

Irene Santalices Acuña 56605
Pablo Sanchez Gonzalez 55453
Sara Aparicio Romeo 55726

Repositorio Github: https://github.com/SaraAparicio22/Microondas-Micros.git

ÍNDICE:

1. INTRODUCCIÓN

En este proyecto se desarrolla una simulación funcional de un microondas utilizando un microcontrolador STM32, utilizaremos lenguaje C además de la biblioteca de abstracción de hardware (HAL).

El objetivo principal es implementar y demostrar el comportamiento básico de un microondas real, incluyendo funcionalidades como la configuración de tiempo mediante un potenciómetro, la activación de procesos a través de un botón y la simulación del estado del microondas mediante indicadores LED.

El programa utiliza un potenciómetro B100K como convertidor analógico a digital para leer el valor que posteriormente se configurará en el temporizador del microondas. De esta forma el usuario puede elegir qué cantidad de tiempo quiere “calentar” su comida.

Con la pulsación del botón de la placa se enciende el LED verde (indicando que el proceso está funcionando) y comienza el temporizador previamente cargado con el valor del potenciómetro.

A medida que pasan los segundos el contador va decreciendo hasta llegar a 0, activando así el LED rojo que indica el fin del temporizador (su comida está lista).

Hay 4 LED configurados pese a que solo hacen falta 2. La razón es para tener mayor comodidad a la hora de la simulación con la maqueta. Se explica más adelante en el apartado “Problemas”.

2. COMPONENTES UTILIZADOS

Hardware

1. **Microcontrolador STM32F4**
    - Modelo: STM32F411VG
    - Arquitectura: ARM Cortex-M4
    - Temporizador por interrupción TIM2 configurado en el pin PA1.
2. **Potenciómetro:** Usado para ajustar el tiempo de funcionamiento del microondas.
    - Características: 100KΩ
    - Configuración de Pin (ADC): PA2
3. **LEDs:** Representan diferentes estados del microondas:
    - LED verde de la placa configurado en el pin PD12.
    - Diodo LED verde configurado en el pin PA3.
    - LED rojo de la placa configurado en el pin PD13.
    - Diodo LED rojo configurado en el pin PA4.
4. **Botón pulsador:** Empleado para simular la apertura o cierre de la puerta del microondas y la activación del sistema.
    - Conectado a un pin de interrupción externa (GPIO PA0).
5. **Cables y conexiones**
    - Conexiones a la placa base mediante cables Dupont.

Software

1. **STM32CubeIDE**
    - Entorno de desarrollo integrado utilizado para programar, depurar y cargar el código en el microcontrolador.
2. **HAL (Hardware Abstraction Layer)**
    - Biblioteca utilizada para interactuar con los periféricos del STM32.
3. **Lenguaje C**
    - Lenguaje de programación empleado para implementar el sistema.

3. FUNCIONAMIENTO

El microondas simulado sigue la siguiente lógica de operación:

1. **Ajuste de tiempo**:
    - El usuario ajusta el tiempo deseado utilizando el potenciómetro. El valor analógico del potenciómetro se convierte a digital mediante el módulo ADC del microcontrolador.
    - El tiempo se escala en función del valor del ADC.
2. **Inicio del proceso**:
    - Al presionar el botón, se activa el temporizador (TIM2), que comienza una cuenta regresiva según el tiempo configurado.
    - Durante el tiempo de operación, el LED verde permanece encendido, indicando que el microondas está "cocinando".
3. **Finalización del proceso**:
    - Cuando el temporizador alcanza cero, el LED verde se apaga y el LED rojo se enciende, señalando que el tiempo de operación ha finalizado.
    - El microcontrolador detiene el temporizador y espera una nueva configuración del tiempo o la reactivación mediante el botón.

El flujo del programa en el bucle principal está diseñado para leer continuamente el valor del ADC, mientras que las acciones relacionadas con el botón y temporizador se manejan en las interrupciones (“HAL_GPIO_EXTI_Callback” y “HAL_TIM_PeriodElapsedCallback”).

4. PROGRAMACIÓN

El código fuente desarrollado incluye los módulos necesarios para configurar y utilizar los periféricos de ADC, GPIO y TIM. A continuación, destacamos algunas partes importantes del código:

1. **ADC**

El ADC1 se utiliza para leer los valores del potenciómetro y pasarlos de analógico a digital.

- Inicialización

```
hadc1.Instance = ADC1;
hadc1.Init.ClockPrescaler = ADC_CLOCK_SYNC_PCLK_DIV2;
hadc1.Init.Resolution = ADC_RESOLUTION_8B;
hadc1.Init.ScanConvMode = DISABLE;
hadc1.Init.ContinuousConvMode = DISABLE;
hadc1.Init.DiscontinuousConvMode = DISABLE;
hadc1.Init.ExternalTrigConvEdge = ADC_EXTERNALTRIGCONVEDGE_NONE;
hadc1.Init.ExternalTrigConv = ADC_SOFTWARE_START;
hadc1.Init.DataAlign = ADC_DATAALIGN_RIGHT;
hadc1.Init.NbrOfConversion = 1;
hadc1.Init.DMAContinuousRequests = DISABLE;
hadc1.Init.EOCSelection = ADC_EOC_SINGLE_CONV;
if (HAL_ADC_Init(&hadc1) != HAL_OK)
{
  Error_Handler();
}
```

- Bucle infinito: espera a recibir los datos para terminar la conversión.

```
while (1)
{
  /* USER CODE END WHILE */

  /* USER CODE BEGIN 3 */
        //Leemos el valor del potenciometro
        HAL_ADC_Start(&hadc1);

              if(HAL_ADC_PollForConversion(&hadc1, HAL_MAX_DELAY) == HAL_OK){
                    ADC_val = HAL_ADC_GetValue(&hadc1);
              }
        HAL_ADC_Stop(&hadc1);
}
```

2. **Interrupción del botón**

```
void HAL_GPIO_EXTI_Callback(uint16_t GPIO_Pin)
{
  if (GPIO_Pin == GPIO_PIN_0)
  {
      //buttonPressed = 1;
      HAL_GPIO_WritePin(GPIOD, GPIO_PIN_12, GPIO_PIN_SET);
      HAL_GPIO_WritePin(GPIOA, GPIO_PIN_3, GPIO_PIN_SET);  // LED verde ON
      HAL_GPIO_WritePin(GPIOA, GPIO_PIN_4, GPIO_PIN_RESET); // LED naranja OFF
      __HAL_TIM_CLEAR_FLAG(&htim2, TIM_IT_UPDATE);
      __HAL_TIM_SET_COUNTER(&htim2, ADC_val*100);
      HAL_TIM_Base_Start_IT(&htim2);

  }
}
```

Cuando se pulsa el botón, el microondas empieza a calentar y se encienden ambos LED de color verde indicando que está funcionando.

3. **Interrupción del temporizador**

```
void HAL_TIM_PeriodElapsedCallback(TIM_HandleTypeDef *htim)
{
  if (htim->Instance == TIM2)
  {
      //time_end = 1;
      HAL_GPIO_WritePin(GPIOD, GPIO_PIN_13, GPIO_PIN_SET);
      HAL_GPIO_WritePin(GPIOA, GPIO_PIN_4, GPIO_PIN_SET);  // LED Naranja ON
      HAL_GPIO_WritePin(GPIOA, GPIO_PIN_3, GPIO_PIN_RESET); // LED naranja OFF
      HAL_TIM_Base_Stop_IT(&htim2);

  }
}
```

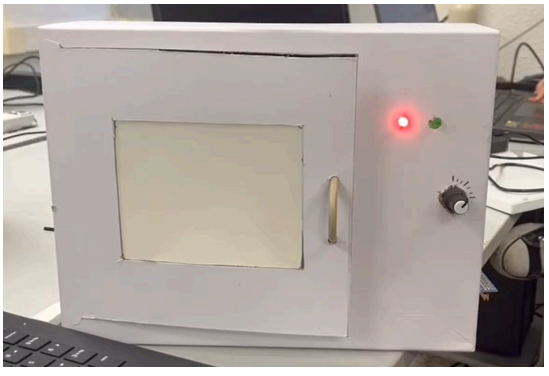
Cuando termina el tiempo que hemos puesto en el potenciómetro (el microondas ha terminado de calentar), se encienden ambos LED rojos.

5. MAQUETA

Para realizar la simulación hemos hecho un microondas clásico, con 2 LEDs, uno verde y un rojo que indican el estado en el que se encuentra el microondas y un potenciómetro para regular el tiempo.

Cuando se gira el potenciómetro para elegir el tiempo de cocinado, y se pulsa el botón, el LED verde se enciende indicando que está calentando la comida.

Cuando termina el tiempo, se apaga el LED verde y se enciende el LED rojo, indicando que ha finalizado el tiempo.



6. PROBLEMAS GENERADOS Y SOLUCIONES PROPUESTAS

1. Configuración del Reloj del Microcontrolador

El temporizador del microcontrolador no funcionaba de forma correcta debido a que la configuración inicial del reloj del proyecto no proporcionaba una frecuencia adecuada. Esto provocaba desajustes en las temporizaciones y, por ello, un mal funcionamiento del temporizador y de las funcionalidades dependientes, como el control del tiempo restante mostrado en el display.

Tras analizar la configuración predeterminada y consultar documentación llegamos a la conclusión que la fuente del reloj del microcontrolador debía ser ajustada para obtener un funcionamiento adecuado. Decidimos modificar la configuración del oscilador a HSI (High-Speed Internal), una fuente de reloj interna con una frecuencia estable de 16 MHz, lo que resultó en una base de tiempo adecuada para el temporizador.

Además, ajustamos los valores de los prescalers de los buses para garantizar que las divisiones de frecuencia fueran compatibles con nuestras necesidades (AHB a 1 división, APB1 a 4 divisiones, APB2 a 2 divisiones).

2. Diseño y Montaje de la Maqueta

Al diseñar la maqueta, nos encontramos con dificultades para implementar todos los componentes de manera óptima. El uso exclusivo de los LED integrados en la placa de desarrollo (STM32F4) generaba problemas de visibilidad y accesibilidad durante las pruebas y la demostración.

Decidimos implementar dos diodos LED externos (uno rojo y uno verde) conectados directamente al microcontrolador, lo que facilitó el montaje y permitió una demostración más clara de las funciones del proyecto. Sin embargo, mantuvimos los LED integrados de la placa para mayor flexibilidad y comodidad durante la defensa de la maqueta, ya que estos pueden ser fácilmente utilizados como respaldo en caso de fallos en los componentes externos.

3. Gestión del Display de 7 Segmentos

Durante la integración del display de 7 segmentos, tuvimos algunos problemas en la sincronización del tiempo restante y la correcta actualización de los dígitos. Inicialmente, esto generaba un efecto de parpadeo perceptible y actualizaciones incorrectas al cambiar el tiempo restante. Finalmente, debido a la falta de tiempo, decidimos no introducirlo y dejar el funcionamiento del microondas como uno más clásico.

7. CONCLUSIONES

Este proyecto de simulación de un microondas con un microcontrolador STM32F4 nos permitió poner en práctica varios conceptos importantes de programación y electrónica. Aprendimos a manejar periféricos como el ADC, GPIO y temporizadores, y a integrar componentes externos de forma efectiva. Aunque tuvimos problemas como ajustar correctamente el temporizador u optimizar la maqueta para su presentación, encontramos soluciones que mejoraron tanto el diseño como el funcionamiento general del sistema. Además consideramos que una futura mejora sería añadiendo un buzzer o señal sonora al finalizar la cuenta regresiva. Este sonido permitiría una notificación más clara y útil para el usuario, especialmente en situaciones donde no pueda observarse directamente el estado de los LEDs.