Laboratorio uno

Samuel Enrique Bejarano Soto [est.samuel.bejarano@unimilitar.edu.co](mailto:est.samuel.bejarano@unimilitar.edu.co) Juan David Medina [est.juan.medina@unimilitar.edu.co](mailto:est.juan.medina@unimilitar.edu.co) Andrés Felipe Bernal - 7003748 [est.andres.bernal1@unimilitar.edu.co](mailto:est.andres.bernal1@unimilitar.edu.co)

Profesor: Diego.

***Resumen:*** Este informe de laboratorio detalla una práctica donde se utilizó el micro controlador STM32F767. Con acceso a todos los pines disponibles, se llevaron a cabo cuatro ejercicios distintos para abordar diversos aspectos de la programación y el uso de los pines. Los ejercicios cubrieron desde la configuración de puertos GPIO hasta la comunicación serial. A lo largo del informe, se describirán en detalle los procedimientos y los resultados obtenidos en cada ejercicio, destacando la importancia de la exploración de pines en el contexto de la STM32F767.

**Palabras clave:** Stm32F767 – Keil Uvision – leds – motor paso a paso – corriente – display.

**Abstract:** This laboratory report details an experiment where the STM32F767 microcontroller was utilized. With access to all available pins, four distinct exercises were conducted to address various aspects of programming and pin usage. The exercises ranged from configuring GPIO ports to serial communication. Throughout the report, the procedures and outcomes of each exercise will be described in detail, emphasizing the significance of pin exploration within the context of the STM32F767.

**Keywords**: Stm32F767 – Keil Uvision – leds – motor step by step – current – display.



1. INTRODUCCION.

En el mundo de la electrónica y la programación de sistemas, la STM32F767 ha surgido como una herramienta esencial que combina potencia y versatilidad en un solo microcontrolador. Este informe de laboratorio se adentra en el emocionante universo de la STM32, centrándose en su programación y aplicación práctica a través de una serie de ejercicios diseñados para expandir nuestra comprensión y habilidades en este campo.

Cada ejercicio aborda un aspecto específico de la programación y uso de la STM32F767, desde la configuración inicial hasta la interacción con periféricos como puertos GPIO, convertidores analógico-digitales (ADC) y comunicación serie. A medida que progresamos en el informe, examinaremos cómo los conceptos teóricos se traducen en la práctica, permitiéndonos comprender mejor cómo la programación eficiente de la STM32F767 puede desbloquear un mundo de posibilidades.

La relevancia de este informe radica en la creciente demanda de profesionales que pueden trabajar con microcontroladores avanzados. Además, el informe busca fomentar la comprensión de cómo la programación y la experimentación práctica se combinan para fortalecer nuestra competencia en la resolución de problemas de electrónica.

A medida que nos sumergimos en los ejercicios planteados en la guía, descubriremos la utilidad y la versatilidad de la STM32F767, y cómo su programación puede abrir puertas a la innovación en una amplia gama de aplicaciones.

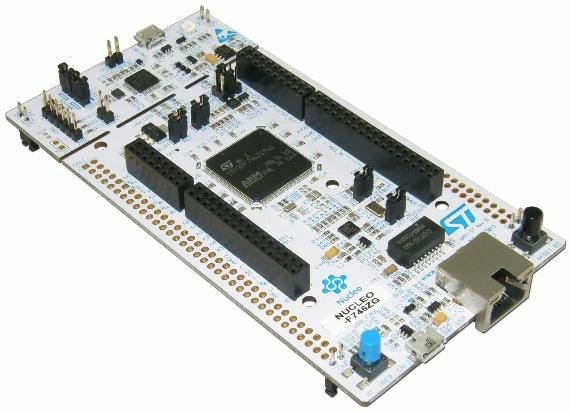
1. MARCO TEORICO.

## STM32:

El STM32 es una familia de microcontroladores de 32 bits desarrollados por la compañía STMicroelectronics. Estos microcontroladores ofrecen una amplia variedad de características y opciones de conectividad para una amplia gama de aplicaciones. Son muy populares en el

campo de la electrónica, la robótica y la industria, gracias a su potencia y versatilidad.





**Imagen 1**. STM32

## KEIL:

Keil uVision es un entorno integrado de desarrollo (IDE) utilizado principalmente para programar microcontroladores de la familia ARM, aunque también admite otros microcontroladores. Fue desarrollado por la compañía Keil, que fue adquirida por ARM en 2005 y ahora es parte de la empresa de software de desarrollo ARM. Keil uVision ofrece una plataforma completa para el desarrollo de proyectos de microcontroladores, que incluye herramientas de compilación, depuración, simulación y programación. Es una herramienta popular entre los desarrolladores de sistemas embebidos debido a su facilidad de uso y su amplio soporte para microcontroladores de diferentes fabricantes**.**

## RESISTENCIAS:

Una resistencia es un componente eléctrico pasivo diseñado para limitar el flujo de corriente eléctrica en un circuito. Su función principal es ofrecer resistencia al paso de la corriente, lo que regula la cantidad de corriente que fluye a través de ella y también afecta la caída de voltaje en el circuito.



**Imagen 2**. Resistencias.

## LEDS:

Un LED, o Diodo Emisor de Luz, es un dispositivo semiconductor que emite luz cuando pasa corriente eléctrica a través de él. Su capacidad para generar luz en diversos colores, su eficiencia energética y durabilidad lo convierten en una opción ampliamente utilizada en indicadores visuales, pantallas, iluminación y una variedad de aplicaciones electrónicas. Su tamaño compacto, encendido instantáneo y la posibilidad de controlar su intensidad hacen que los LEDs sean esenciales en la tecnología moderna.



**Imagen 3**. Leds

## MOTOR PASO A PASO:

Un motor paso a paso es un dispositivo electromecánico que convierte pulsos eléctricos en movimientos discretos o "pasos". Estos movimientos angulares se logran mediante la activación secuencial de bobinas electromagnéticas ubicadas en el motor. Cada pulso eléctrico hace que el motor avance un paso, lo que lo hace ideal para aplicaciones que requieren precisión en el control del movimiento, como impresoras 3D, máquinas CNC, sistemas de posicionamiento y robots, entre otros. La característica distintiva del motor paso a paso es su capacidad para moverse en incrementos predefinidos sin necesidad de retroalimentación, lo que lo hace muy útil en situaciones donde la posición precisa es esencial.

1. PROCEDIMIENTO.

Para esta práctica se aprovechó de las librerías LCD-Comm-Write y SysTick, desarrolladas en prácticas anteriores para la agilización del desarrollo de esta práctica.

Para iniciar con el desarrollo de esta práctica se asignaron los pines necesarios para su desarrollo. Debido a que se tienen en cuenta los pines que están asignados para comunicación serial, fue necesario recurrir al datasheet para verificar que los pines seleccionados eran los correctos para cumplir con los objetivos de la práctica:

|  |  |  |
| --- | --- | --- |
| LCD |  |  |
| Salidas | Pin | Ubicación |
| E | PC9 | CN12 - 1 |
| RS | PC8 | CN12 - 2 |
| d7 | PC7 | CN12 - 19 |
| d6 | PC6 | CN12 - 4 |
| d5 | PC5 | CN12 - 6 |
| d4 | PC4 | CN12 - 34 |
| d3 | PC3 | CN11 - 37 |
| d2 | PC2 | CN11 - 35 |
| d1 | PC1 | CN11 - 36 |
| d0 | PC0 | CN11 - 38 |

Tabla 1. Pines asignados para control de la LCD

|  |  |  |
| --- | --- | --- |
| Teclado Matricial | |  |
| Entradas | Pin | Ubicación |
| Column 1 | PD0 | CN9 - 25 |
| Column 2 | PD1 | CN9 - 27 |
| Column 3 | PD2 | CN8 - 12 |
| Column 4 | PD3 | CN9 - 10 |
| Salidas | Pin | Ubicación |
| Row 1 | PD4 | CN9 - 8 |
| Row 2 | PD5 | CN9 - 6 |
| Row 3 | PD6 | CN9 - 4 |
| Row 4 | PD7 | CN9 - 2 |

Tabla 2. Pines necesarios para la lectura del teclado matricial

|  |  |  |
| --- | --- | --- |
| Entradas | Pin | Ubicación |
| Clear | PC13 | BTN-User |

Tabla 3. Asignación de botón para limpiar la LCD

|  |  |  |
| --- | --- | --- |
| USART3 | Pin | Ubicación |
| RX | PD8 | CN12 - 10 |
| TX | PD9 | CN11 - 69 |
|  |  |  |
| USART6 | Pin | Ubicación |
| RX | PG9 | CN10 - 16 |
| TX | PG14 | CN10 - 14 |

Tabla 4 y 5. Asignación de pines para USART

Una vez se asignaron los pines se empezó con el desarrollo de una librería que permite la configuración de cualquier USART presente en la tarjeta STM32F767ZI y algunas otras que comparten estas configuraciones. Esta librería usa una función las cual debe recibir la siguiente información:

* USART X (un numero entero que permita identificar el USART que se desea usar)
* PUERTO ‘X’ (la letra que indica el puerto que se desea usar)
* PINES XxXx (número entero que indique los pines a usar, ejemplo, si se quieren usar los pines 7 y 8, se debe escribir 78, siempre poniendo el número menor primero)
* Bits de Parada (número flotante indicando los bits de parada siendo permitidos 0.5, 1, 1.5 y 2)
* BAUDIOS (un numero entero que indique los baudios deseados para la comunicación)

Una vez realizada esta función y teniendo en cuenta los registro necesarios se creó la librería y se implementó en el código principal del proyecto.

Una vez realizado esto, se procedió a realizar la lógica que permitía la lectura del teclado matricial, para esto se realizó la configuración de 8 pines, 4 de ellos como salidas generales y los otras 4 se usaron como interrupciones.

En el código principal se creó una secuencia que cada cierto tiempo enviada una salida en lato a cada una de las columnas por separado, y las interrupciones están asignadas a las filas, por lo que al activar una de estas interrupciones el código compara la columna que estaba siendo activada y la interrupción que se activó para determinar la posición de la tecla que se pulso en el teclado. Esta comparación le permite seleccionar la letra de esa misma ubicación en una matriz generada en el código para seleccionar entre 14 letras:

char matriz[4][4] = {'A','I','E','O',

'H','L','S','M',

'N','B','T','P',

'\*','R','#',' '};

Una vez se tenía el funcionamiento de cada una de las partes que componen el laboratorio, se procedió a juntarlo todo teniendo en cuenta los objetivos y procedimientos requeridos en la guía, así como el envió de información mediante un USART inhabilitando la recepción de información mediante el OTRO USART, es decir, solo podía responder el USART que había

recibido la información desde el teclado matricial.

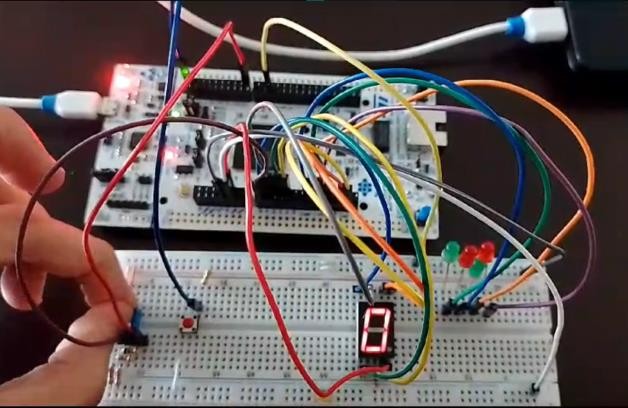
1. ANALISIS

Al intentar cumplir con una de las condiciones del laboratorio se evidencio que deshabilitar la interrupción de los USART no es una buena opción, ya que el proyecto empezaba a generar errores que al cabo de pocas interacciones quedaba completamente congelado en un bucle de la interrupción deshabilitada.

Además, se propusieron varias formas de realizar la lectura del teclado matricial, y aunque todas podían funcionar perfectamente, se decidió usar el método antes mencionado ya que facilitaba algunos procesos y era el más óptimo en lo que a código se refiere, teniendo como comparación los otros métodos propuestos en el grupo.

Debido a que la recepción de información de USART se hace mediante interrupciones, fue necesario implementar las funciones que permiten determinar la prioridad de las interrupciones, poniendo como interrupción inmediata después del botón de reset de la tarjeta en cuanto a prioridad, a la interrupción del botón de usuario que permite el reseteo de la LCD así como de las configuraciones iniciales de los USART.

1. RESULTADOS.



**Imagen 14.** Montaje motor, nombre y contador.

1. CONCLUCIONES.
2. Una vez realizada la guía se determinó que la mejor forma de limitar la recepción de información mediante USART, es inhabilitando la recepción de información mediante los registros del USART
3. Al encontrar varias soluciones para una necesidad, es necesario determinar cuál es la opción n más óptima que permita desarrollar de manera más eficiente el cumplimento de dicha necesidad