# Informe practica de laboratorio 6

**Universidad Sergio Arboleda**
**Sistemas Embebidos**

**Estudiantes:**

- Sharay Valentina Galvis Prieto.
- Juan Diego Moreno Cardenas.

## Introducción:

El presente informe describe el desarrollo e implementación de un sistema embebido utilizando el microcontrolador STM32F411. El objetivo principal del laboratorio fue diseñar un sistema capaz de realizar mediciones de distancia (en mm) por medio del uso de cuatro pares de LEDs IR (emisor y receptor), las cuales son recibidas y procesadas en el microcontrolador por pines análogicos y por el periférico ADC para poder obtener su equivalencia en voltaje; además, realizar la conversión de este voltaje al valor de distancia correspondiente para cada sensor. Se implementó una función que permite enviar los datos e información calculada desde el microcontrolador hasta una aplicación de ventana diseñada en el software QT Creator que permitirá visualizar la distancia detectada por cada sensor.

Entre los diferentes objetivos que se plantearon se encuentra, el establecer una comunicación USB mediante el puerto COM, realizar el montaje del circuito propuesto por el profesor en donde se disponen los LEDs IR emisor y receptor, obtener los datos generados por el led de manera analogica y asi mismo realizar un proceso de conversion ADC (convertidor analogico digital) para tener un valor de voltaje equivalente al valor recibido desde el GPIO, generar una funcion para pasar desde los valores dados a una distancia en mm que será la recibida desde el sensor; y por último, realizar múltiples mediciones de calibración para cada sensor desde cierta distancia hasta la mínima con intervalos de 5 mm, así mismo, graficar el comportamiento de cada sensor para estandarizar la conversión de datos.

Para cumplir con estos objetivos, se trabajó en la integración del circuito brindado por el profesor, y se configuró tanto la comunicación USB para la transmisión de datos en tiempo real, como los diferentes periféricos y módulos utilizados (Timers, ADC, DMA). Además, todo el desarrollo de este laboratorio está anexo al laboratorio anterior, de esta manera cuenta con todas las funcionalidades que se desarrollaron tanto en el laboratorio #5 como en el laboratorio #6. Este laboratorio permitio reforzar conceptos sobre lectura de valores análogos, conversión ADC, funcionamiento de la resolución de lectura y manejo de DMA (Acceso directo a memoria).

**Materiales utilizados:**

- Microcontrolador STM32F411.
- ST-Link V2.
- Cable USB.
- Protoboard.
- Capacitores.
- Resistencias.
- MOSFET 2n7000.
- Led IR emisor.
- Led IR receptor.
- Cables de conexión.

**Softwares utilizados:**

- Stmcube IDE.
- QT Creator.
- Excel.

**Procedimiento:**

El procedimiento seguido para la implementación del sistema fue el siguiente:

1. Inicialmente, se llevó a cabo la implementación del circuito brindado por el profesor en un montaje físico en protoboard, en donde se presentaban dos partes; la primera corresponde al control de la activación de los leds IR emisores, se utilizaron resistencias de 120 ohm para limitar la corriente que ingresaba al led y se utilizó un mosfet 2n7000 para poder controlar por medio de una señal digital (enviada desde el microcontrolador) cuando se cierre el circuito y comunique la tierra con el led; la segunda parte del circuito corresponde a la configuración de los leds receptores, se utilizó una resistencia de 10k y un conector desde el cátodo del led que va hacia una un pin GPIO analógico del microcontrolador para posteriormente leer la señal. Para este montaje se utilizó un adaptador de voltaje de 3.3v a 1A, además se interconectaron las tierras del montaje con las del microcontrolador y el adaptador; finalmente se cubrieron las partes laterales de todos los leds IR emisores con cinta aislante para evitar que las mediciones de los sensores se vieran afectadas por alguna señal intrusiva o no correspondiente a la emitida.

2. Ahora, se comprobó el correcto funcionamiento del circuito anteriormente implementado, se realizaron pruebas tanto de continuidad como de voltaje presente en los leds. Iniciamos verificando el funcionamiento de los mosfets mandando señales de 1 segundo de duración desde el microcontrolador y verificando por medio de una cámara de un dispositivo móvil si realmente el led IR emisor emite alguna señal; para probar el led IR receptor, mantuvimos el led emisor encendido, mientras con el uso del multímetro verificamos cómo variaba el voltaje sobre el led IR receptor a medida que variaba la distancia de una superficie a la cual apuntaban ambos leds.

3. Al haber verificado el funcionamiento del montaje, comenzamos a configurar los parámetros del proyecto desde STM32IDE. Como se mencionó anteriormente, el proyecto debe contar con las funcionalidades del laboratorio anterior, con lo cual, utilizamos el proyecto creado para ese caso y a partir de este modificamos ciertas configuraciones. Basándonos en el ejemplo realizado en clase, en donde se configuraba tanto las entradas analógicas (ADC)(Analog-to-Digital Converter) como la conexión del DMA (Direct Memory Access), iniciamos estableciendo la conectividad USB_OTG_FS en modo Device_Only, se configuró el USB_DEVICE en la clase Communication Device Class. Al utilizar este medio de transmisión, era necesario establecer el reloj interno del microcontrolador a una velocidad de 96MHz. Además, se configuró un timer (tim4) el cual se utiliza para controlar la duración de los pulsos que se enviaran a los leds IR emisores, con lo cual utilizamos un prescaler de (48000-1) y de esta manera obtenemos un valor de 2 pulsos en el tim, cada vez que transcurre 1 milisegundo, como se observa en la *Ec.1* se activó el periférico ADC1, para poder obtener los datos de los 4 leds IR, específicamente se utilizó IN4, IN5, IN6, IN7 para recibir los datos. Se configuró el DMA2 en modo circular, con baja prioridad y con dirección desde el periférico hacia la memoria del microcontrolador; además, para la recepción de los datos, se habilitó la opción “scan conversion” para que el ADC recorra todos los canales que utilizamos y realice la conversión una tras otra, además se configuró una resolución de 12 bits y ordenamos la lectura de los canales desde el cuarto hasta el séptimo. Definimos el “sampling time” en 3 ciclos del reloj para cada canal, es decir, 125 ns de lectura en cada canal.

$$\frac{96\,000\,000 \; -> HCLK}{48\,000 -> Prescaler} = 2000 \rightarrow cada\ segundo$$

$$\frac{2000}{1000} = 2 \rightarrow cada\ milisegundo$$

*Ec1. Cálculo prescaler tim4*

4. Una vez configurados los timers del microcontrolador, se configuraron los siguientes pines en la STM32 (además de los ya configurados en el laboratorio anterior):

- **PIN A4, PIN A5, PIN A6 y PIN A7**: Estos cuatro pines representan las entradas analógicas de los canales IN4, IN5, IN6 e IN7 respectivamente. Son los pines encargados de recibir la señal generada por cada sensor IR receptor.
- **PIN B12, PIN B13, PIN B14 y PIN B15**: Estos cuatro pines representan las salidas digitales que se encargan de encender cada led emisor cada milisegundo.

5. Posteriormente, en base a el código realizado en clase por el profesor, utilizamos una ISR que se ejecuta cada vez que un canal del ADC1 realiza una lectura y se almacena dicho valor dentro de un array de 4 posiciones, en donde cada índice corresponde a cada canal de lectura; de esta manera, este array se actualiza constantemente con las lecturas renovadas del ADC1.

6. Con los valores del adc, se procedió a crear una función encargada de convertir estos datos (0 - 4095) a una escala de voltaje (0 - 3.3), se utilizó la *Ec.2* para calcular esta equivalencia, e igualmente que en el caso anterior, se almacenan dichos valores en otro array de 4 posiciones el cual se actualiza constantemente con los datos renovados del ADC1.

$$\frac{Valor\ ADC * 3.3}{4095} = Valor\ Voltaje$$

*Ec2. Cálculo de valor adc a valor de voltaje*

7. En este punto, utilizamos el protocolo de comunicaciones que se desarrolló en el laboratorio #5, cómo se puede ver en la *Fig 1.0*. El protocolo que utilizamos consta de 5 elementos principales; el primer byte representa el inicio de los datos a enviar, por lo cual siempre es el mismo; en el segundo byte se asigna el tamaño de bytes del paquete completo; luego de este, se agregan todos los bytes que contienen la información que deseamos transmitir (valor del adc, del voltaje y de la distancia para cada canal), cada uno con un identificador respectivamente; posteriormente, se encuentra un byte que representa el CRC (Comprobación de redundancia cíclica) de todos los datos adjuntos al protocolo (excepto el byte final), este byte representa una técnica para detectar errores en la transmisión de datos y funciona realizando una operación XOR byte a byte desde el primer primer byte del paquete hasta el último byte que contiene la información que se desea transmitir; finalmente, el último byte representa el final del protocolo, por lo cual también siempre tiene el mismo valor.

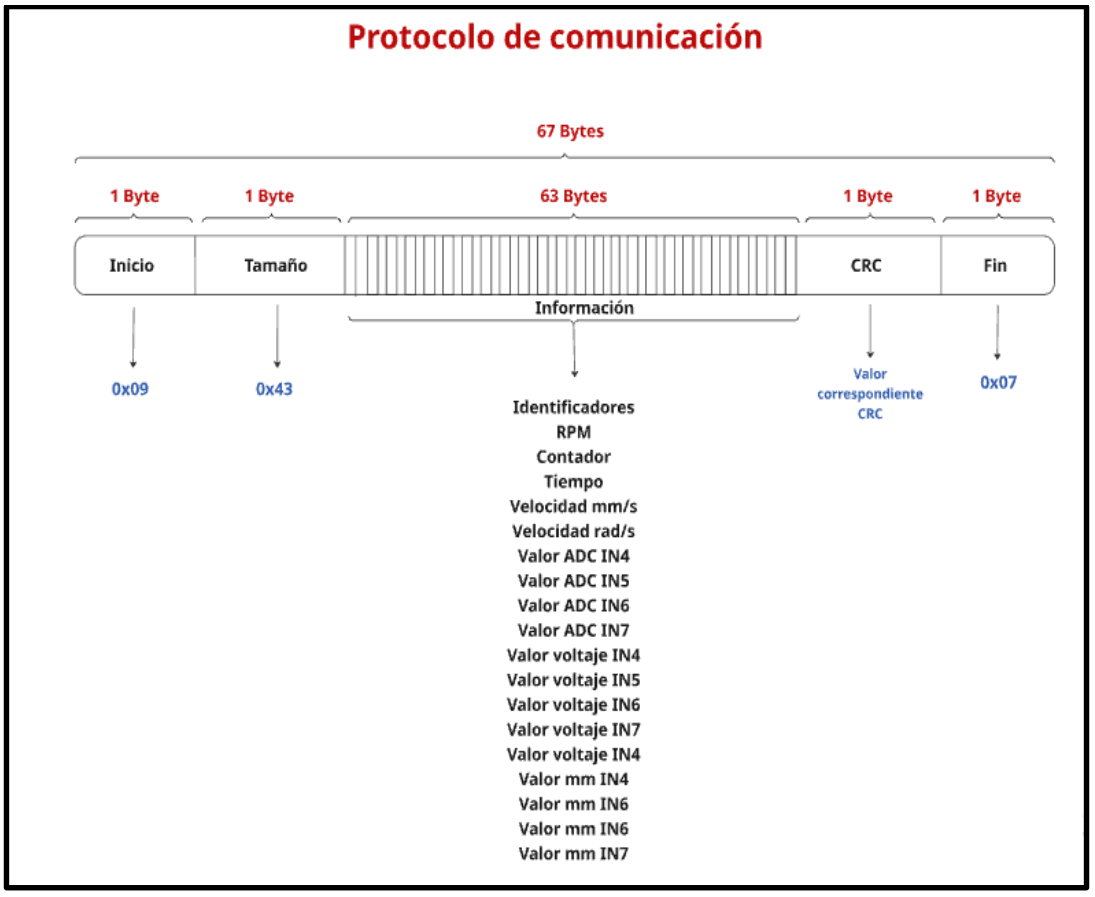


*Fig 1.0 Visualización de protocolo de comunicación desarrollado.*

8. Al tener los datos que requerimos transmitir (valor adc y de voltaje de 4 canales), separamos cada variable en bytes dependiendo de su tamaño y tipo, y se agregan dentro del mismo arreglo que ya teníamos en el laboratorio anterior con diferentes variables (RPM, tiempo, velocidad, etc.) separándolas por su respectivo identificador como lo podemos ver en la *Fig 1.1*. De esta manera, se agrega el conjunto de datos dentro del paquete creado en el inciso 7. y finalmente se envía byte a byte, todo el paquete desde la función CDC_Transm_FS() para el receptor por el puerto serial.

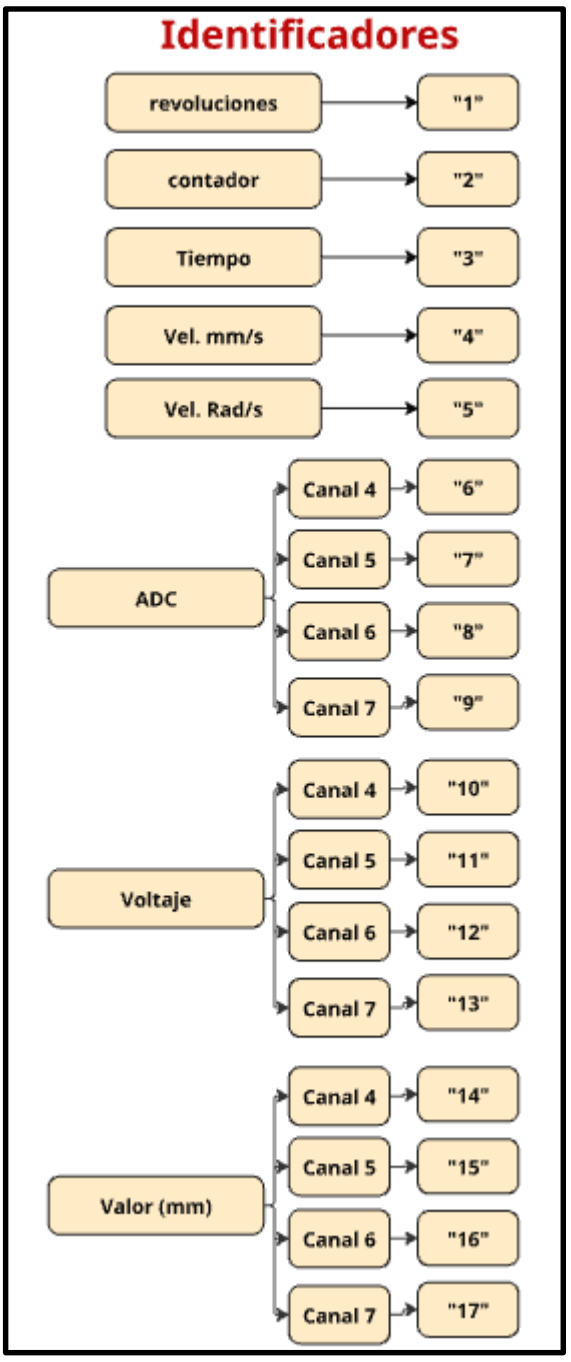


*Fig 1.1 Especificación de identificadores creados para el protocolo de comunicación desarrollado.*

9. Posteriormente, se actualizó el proyecto creado en el laboratorio anterior en la aplicación QT Creator, en donde se generó una disposición de barras de progreso, respecto a una imagen que hace referencia a los 4 leds IR receptores, en donde debe variar el valor de la barra respecto a la distancia detectada por cada sensor. Por otra parte, se agregó una tabla en donde se visualizan los datos recibidos de cada sensor IR, así se puede observar el valor del ADC recibido, de la conversión a voltaje y de la distancia calculada; estos datos se agregaron al mismo cuadro de texto (en donde en el anterior laboratorio se visualizaban los datos de las RPM del motor y el tiempo real transcurrido) para de esta manera poder exportar los datos de cada sensor a un archivo .log para y poder interactuar con ellos.

   En este programa se utilizó la función (desarrollada en el laboratorio #5) que valida la correcta recepción del paquete de datos según el protocolo establecido, el cual comprueba que los datos recibidos tengan en su primer índice el valor de (0x09) que es el valor de inicio del protocolo, verifica que el tamaño (índice 2) sea igual al tamaño del paquete recibido, calcula el CRC del paquete recibido y se compara con el penúltimo índice recibido y finalmente comprueba que el último valor del paquete sea (0x07) que es el valor del final del protocolo; si cumple con estas condiciones se prosigue a combinar los datos según sus identificadores y se muestran en la interfaz en cada una de las etiquetas creadas anteriormente.

10. Ahora, procedimos a realizar la calibración de los sensores para poder obtener una conversión de los datos recibidos por el ADC a su equivalente en distancia (milímetros). Ubicamos cada pareja de sensores de manera paralela en donde apuntarán hacia una misma dirección; además, se utilizó una cartulina negra como base, en la cual se marcaron 40 líneas a una distancia de separación de 5mm entre cada una. Para realizar la obtención de los datos de las mediciones se utilizó el programa creado en QT Creator para poder almacenar y exportar los datos recibidos por cada sensor.

    El proceso para la calibración fue:
    - Colocar una tabla de color blanco mate paralelamente a la ubicación de todos los leds IR (en un ángulo de 90°, con ayuda de una escuadra).
    - Acercar la tabla a una distancia de 20mm de los leds, y abrir el puerto serial dentro del programa de Qt para de esta manera, esperar 1 segundo aproximadamente por cada medición y proceder a exportar todos los valores de la lectura ADC que se presentaron en ese tiempo transcurrido y a esa determinada distancia. Realizamos este mismo proceso 40 veces más, alejando la tabla cada 5mm hasta llegar a 220mm.
    - Al obtener los 40 archivos .log, se utilizó el software Excel para el procesamiento de los datos exportados, cada archivo contiene una cantidad considerable de lecturas de valor ADC de los 4 canales. Por lo tanto, se creó un libro de Excel, al cual se le agregaron todas las mediciones separándolas en hojas, para tener un orden específico y calcular los promedios por cada medida tomada.
    - Luego dentro del mismo libro de Excel, se creó una hoja llamada “Promedios” en la cual se unificaron y organizaron los promedios de los valores ADC de cada medición, para poder graficar el valor ADC de cada pareja de sensores respecto a la distancia en mm como se puede observar en las siguientes figuras: *Fig.2, Fig. 2.1, Fig.2.2, Fig.2.3.*

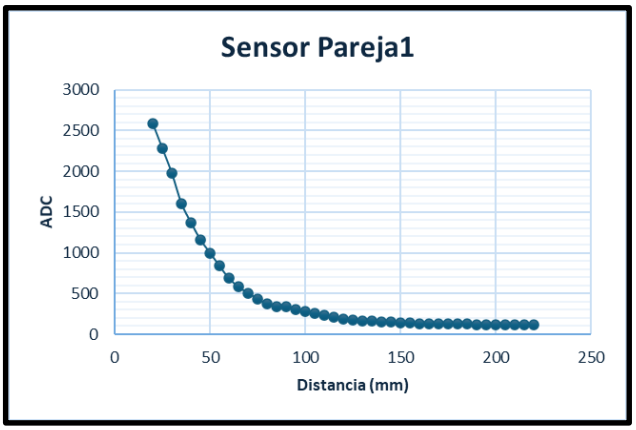


*Fig 2.0. Gráfica de valor ADC respecto a distancia de la pareja 1 de sensores IR.*

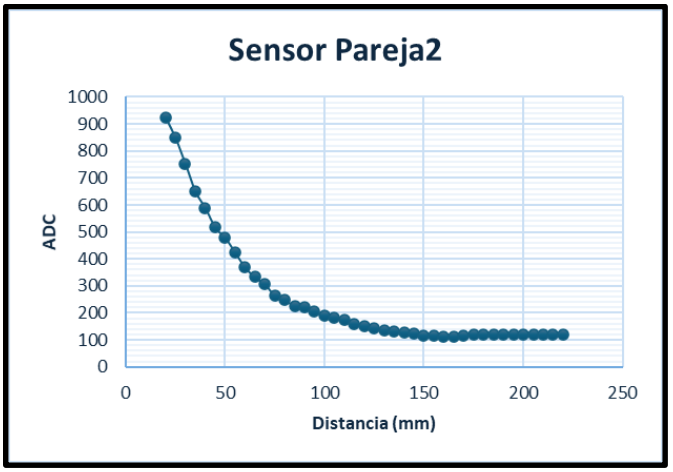


*Fig 2.1. Gráfica de valor ADC respecto a distancia de la pareja 2 de sensores IR.*

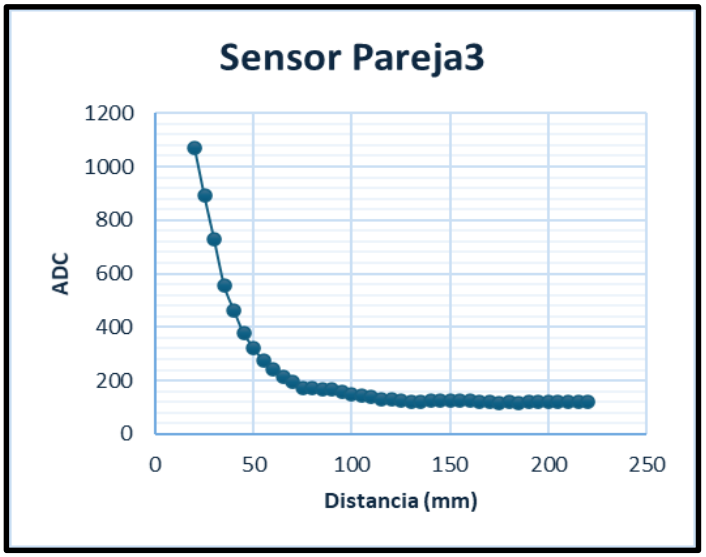


*Fig 2.2. Gráfica de valor ADC respecto a distancia de la pareja 3 de sensores IR.*

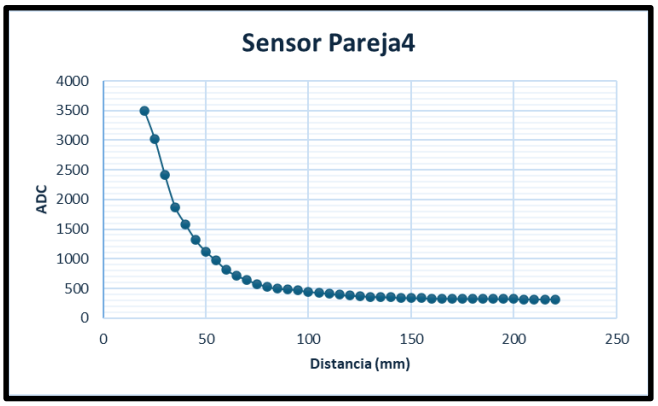


*Fig 2.3. Gráfica de valor ADC respecto a distancia de la pareja 4 de sensores IR.*

11. Posteriormente, se modificó el programa de la STM32 en donde se crearon 5 arrays de datos, donde se almacenan:

    - Las distancias en mm a las cuales se realizaron las pruebas de calibración.
    - Los valores ADC del led IR #1.
    - Los valores ADC del led IR #2.
    - Los valores ADC del led IR #3.
    - Los valores ADC del led IR #4.

    Es importante ordenar los valores del ADC de cada sensor respecto al mismo índice de la lista donde se almacenan las distancias medidas.

12. En este punto, se procedió a realizar una función dentro del mismo programa del microcontrolador en donde se logran interpolar los datos y así obtener un correcto cálculo de la distancia (mm) detectada por cada pareja de sensores, respecto a los valores ADC recibidos de cada par led. La función consiste en obtener el valor del ADC (de cada canal) en un instante de tiempo, este valor (depende del canal) va a recorrer el array de datos creado anteriormente y determina entre qué rango de valores está. De esta forma, tomamos los dos valores ADC más cercanos a los que se encuentra y se procede a realizar la obtención de una ecuación de recta característica para esos dos puntos; tomando esos dos extremos, se ubica el índice de estos datos en el array de los valores de distancia medida para conocer esos datos ADC a que distancia corresponden, con estos valores se calcula el valor de la pendiente (m) como se observa en la *Ec.3* y de igual forma obtenemos el valor de la intersección en el eje y (b) con la *Ec.4*. Al hallar estos valores se crea la ecuación de la recta como se ve en la *Ec.5* y a partir de esta, se ingresa el valor del ADC para obtener un valor de distancia (mm) preciso.

$$\frac{Y2 - Y1}{X2 - X1} = Pendiente\ (m)$$

*Ec3. Cálculo pendiente de la recta*

$$valor\ mm\ - (m * valor\ adc)\ = Intersección\ en\ y\ (b)$$

*Ec4. Cálculo intersección en y de la recta*

$$valor\ mm\ = (m * valor\ adc)\ + b$$

*Ec5. Ecuación de la recta para hallar distancia según valor ADC*

13. Al determinar la distancia en mm, se estandariza la función para la lectura de los 4 canales ADC, y se almacenan los datos en otro array. Este se añade dentro del paquete de datos que se transmite por la comunicación USB para poder visualizarlo en el programa gráfico de Qt Creator.

**Dificultades presentadas:**

Durante el desarrollo del laboratorio, enfrentamos varios retos que complicaron el avance del proyecto, pero logramos resolverlos con análisis y apoyo del profesor, de la siguiente manera:

- **Problemas con la protoboard:** El montaje en la protoboard generó inconvenientes desde el principio. presentaba malos contactos o falta de continuidad en las conexiones. A veces el circuito funcionaba bien, pero de repente dejaba de responder sin que supiéramos por qué. Al no conocer si el error estaba en los LEDs, en el código o en el montaje mismo. Nos tocaba desarmar todo, revisar cada cable con el multímetro y volver a conectar, lo que nos tomaba tiempo. Al final, nos dimos cuenta de que la protoboard, no siempre generaba buen contacto entre las pistas de alimentación y que el cable que utilizamos para comunicarlas, no estaba en buen estado. Luego de este inconveniente se volvió a realizar el montaje del circuito en otra protoboard revisando cada conexión hasta evidenciar su correcto funcionamiento.

- **Mediciones inestables de los LEDs IR**: Cuando se estaban calibrando las medidas de los LEDs IR y haciendo las pruebas, tuvimos problemas con las lecturas. Usamos un multímetro para medir el voltaje en el receptor según la distancia de la tabla de madera, pero los resultados no eran estables. Cuando la tabla estaba lejos, el voltaje apenas cambiaba y los valores del ADC se quedaban muy cerca de cero. Pero al acercarla mucho, el voltaje subía rápidamente. Esto hacía que las mediciones no fueran tan lineales y dificultaba realizar la conversión a distancia. Además, al revisar en STM32Cube IDE con "Live Expressions", los valores del ADC de los LEDs IR variaban demasiado, incluso sin mover la tabla. Pensamos que podía ser ruido en el circuito o que los LEDs no estaban bien puestos, así que tuvimos que ajustar el montaje y repetir las pruebas varias veces hasta que los datos se estabilizaron.

- **Alineación de los LEDs:** Otra dificultad fue lograr que los LEDs emisores y receptores estuvieran bien alineados. Cómo usamos cinta aislante para evitar señales intrusas, cualquier pequeño movimiento en el montaje hacía que las lecturas cambiaran. No fue tan sencillo buscar la posición exacta entre emisor y receptor ya que era clave, y tuvimos que fijarlos mejor en la protoboard para que las mediciones fueran consistentes.

**Diagramas:**

- Diagrama de flujo (*Archivo anexo ¨Link diagrama de flujo¨*).

- Diagrama esquemático (*Archivo anexo ¨Esquemático laboratorio 6¨*).

**Conclusiones:**

1. Los problemas con la protoboard nos enseñaron que un buen contacto en el hardware es tan importante como el código. Si las conexiones fallan, todo el proyecto empieza a presentar inconsistencias, y revisar con el multímetro nos ayudó a comprender donde se presentaban estos errores de continuidad y variación de voltaje.
2. Las mediciones dispersas de los LEDs IR nos mostraron que calibrar sensores como estos, no es tan sencillo. Luego de hacer las pruebas, y evidenciar cómo el voltaje subía repentinamente o se quedaba en cero, se procedió a buscar soluciones como; cambiar parámetros del valor de las resistencias utilizadas en cada sensor para ajustar los valores recibidos y reducir el ruido. Nos quedó claro que la alineación entre emisor y receptor tiene que ser precisa para que los datos sean concisos.
3. La integración y el uso adecuado del convertidor analógico-digital (ADC) y el uso de DMA permitieron realizar lecturas precisas de los sensores, optimizando el procesamiento de datos y reduciendo tanto la carga en la CPU del microcontrolador como el tiempo de ejecución.
4. Se implementó una estrategia de calibración basada en la recopilación de datos a intervalos de 5 mm de distancia, lo que nos permitió generar modelos de interpolación lineal para convertir las lecturas del ADC en valores de distancia con una mayor precisión.
5. La automatización de los procesos resultó fundamental para optimizar la eficiencia para la toma y el procesamiento de los datos. El uso del programa en Qt nos permitió registrar los valores de manera más rápida y precisa, mientras que la implementación de Excel facilitó la actualización automática de los cálculos al ingresar nuevos datos en la hoja de medidas correspondiente. Esto nos ayudó significativamente, ya que al volver a realizar la calibración de los nuevos sensores fue mucho más eficiente el procesamiento de los nuevos datos.
6. La integración de ambos programas correspondientes al laboratorio #5 y al programa actual se lograron de manera exitosa, sin presentar fallas en su funcionamiento. Esta unión permitió una comunicación eficiente entre las herramientas, asegurando la correcta transmisión y procesamiento de los datos.