INFORME PRACTICA DE LABORATORIO 4

David Pérez – Camila Quecan – Ahmed Reyes

David.perez02@usa.edu.co – camila.quecan01@usa.edu.co – Ahmed.reyes01@usa.edu.co

Universidad Sergio Arboleda

¿QUE USAMOS?

- Microcontrolador STM32F411CEU.
- ST LINK V2.
- Programa STMCUBE IDE
- Pantalla Oled
- Base de giro y disco
- Motor DC
- Protoboard, Jumpers, Modulo Puente H, Mys 1838, Modulo de encoders
- Control remoto RM-Y173 Sony.

PROTOCOLO SENSOR IR Y CONTROL:

Para este laboratorio se implementó un sensor Mys 1838, este sensor es una de las partes esenciales para este laboratorio, comenzamos observando como funcionaba así que lo que se realizo fue la alimentación del sensor para ver como funcionaba, a esta alimentación le agregamos una resistencia de 220ohms para protegerlo, al realizar la implementación de este le mandamos la señal desde el control remoto, señal la cual nos presentaba los pulsos de cada uno de los botones del control, como se muestra en la imagen 1.1.

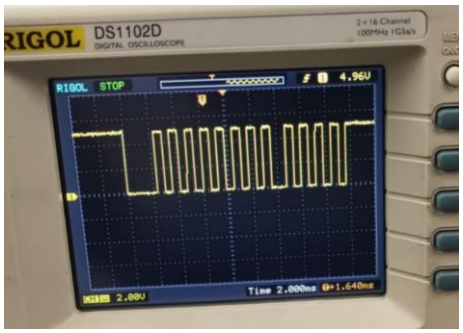


Imagen 1.1 Pulsos obtenidos del sensor oprimiendo el botón 1.

Ahora, ¿Cómo relacionamos ese cuadro de pulsos?, lo que hicimos fue lo siguiente revisar como es el protocolo IR del control Sony, donde encontramos el siguiente ejemplo de los pulsos.

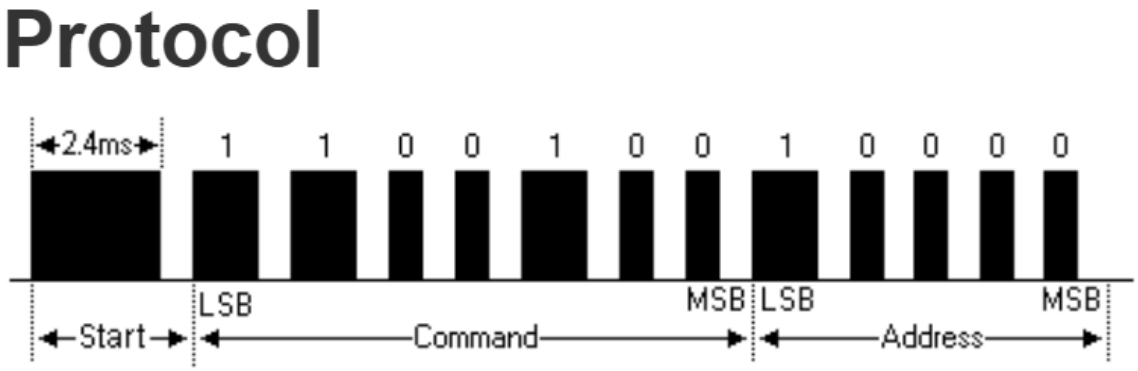


Imagen 1.2 Protocolo del control Sony.

Aquí empezamos a analizar lo que pasa entre las dos imágenes, al realizar la revisión en el osciloscopio “imagen 1.1”, observamos los tiempos de cada pulso, ósea el de “start” y los de “1 y 0”, con esto nos dimos cuenta que el tiempo de cada pulso concordaba con el presentado en la imagen 1.2, pero hay una pequeña diferencia, que si nos damos cuenta es que los pulsos de la imagen 1.1 están invertidos en comparación a la imagen 1.2, ¿Por qué sucede esto? , esto es debido a que el sensor infrarrojo realizan una modulación de la señal, donde si se detecta señal por el sensor la salida de este la toma como un bajo o 0V, por esta razón si miramos el sensor en el osciloscopio sin ninguna señal, se observa un DC de un alto o 5V.

MODULACION DEL CONTROL E IR Mys1838:

Entrando mas a detalle en lo realizado definimos ye encontramos que la señal posee una frecuencia portadora de 40KHz, dentro de esta frecuencia determinamos la duración de los pulsos 1 y 0 según lo investigado, de esto obtuvimos que efectivamente en el tren de pulsos un 1 logico toma el tiempo de 1.2 ms mientras que el 0, 0.6ms como se observa en la imagen 1.3.

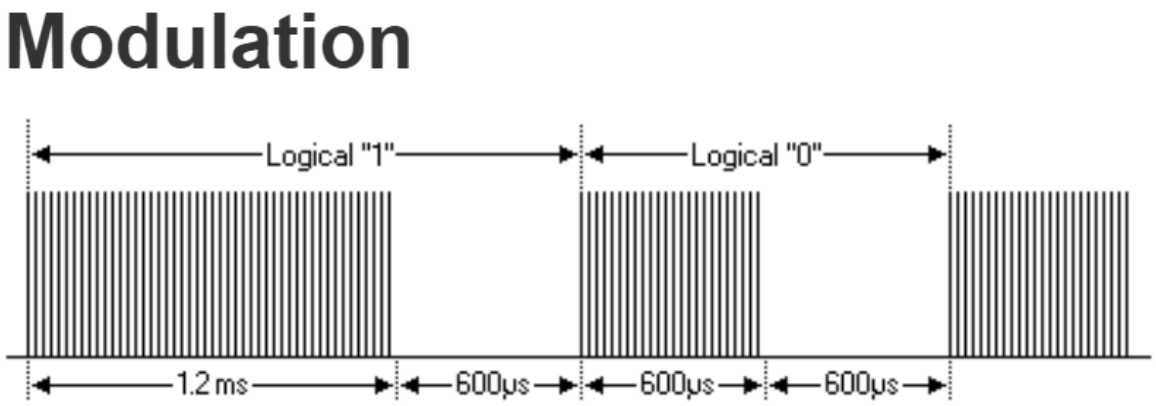


Imagen 1.3 Modulación y tiempo de cada bit en el tren de pulsos.

TRASMISION Y DECODIFICACION DE DATOS:

Realizamos la trasmisión de datos con ayuda del sensor IR, lo configuramos como entrada y recibimos los pulsos con ayuda del TIM, lo primero que hicimos fue poner el umbral de los pulsos de acuerdo a la modificación del pre escaler, entonces tomamos los valores del tiempo de los pulso 1 y0 y los acomodamos en el umbral para obtener exactamente si es un 1 y si es un 0, así decodificamos la señal, ahora al inicio en la pantalla oled solo imprimimos los 8 primeros bits del pulso, que es el start y los 7 bits de comando, con esto verificas en binario que el programa imprimiera en la pantalla los valores obtenidos en el osciloscopio.

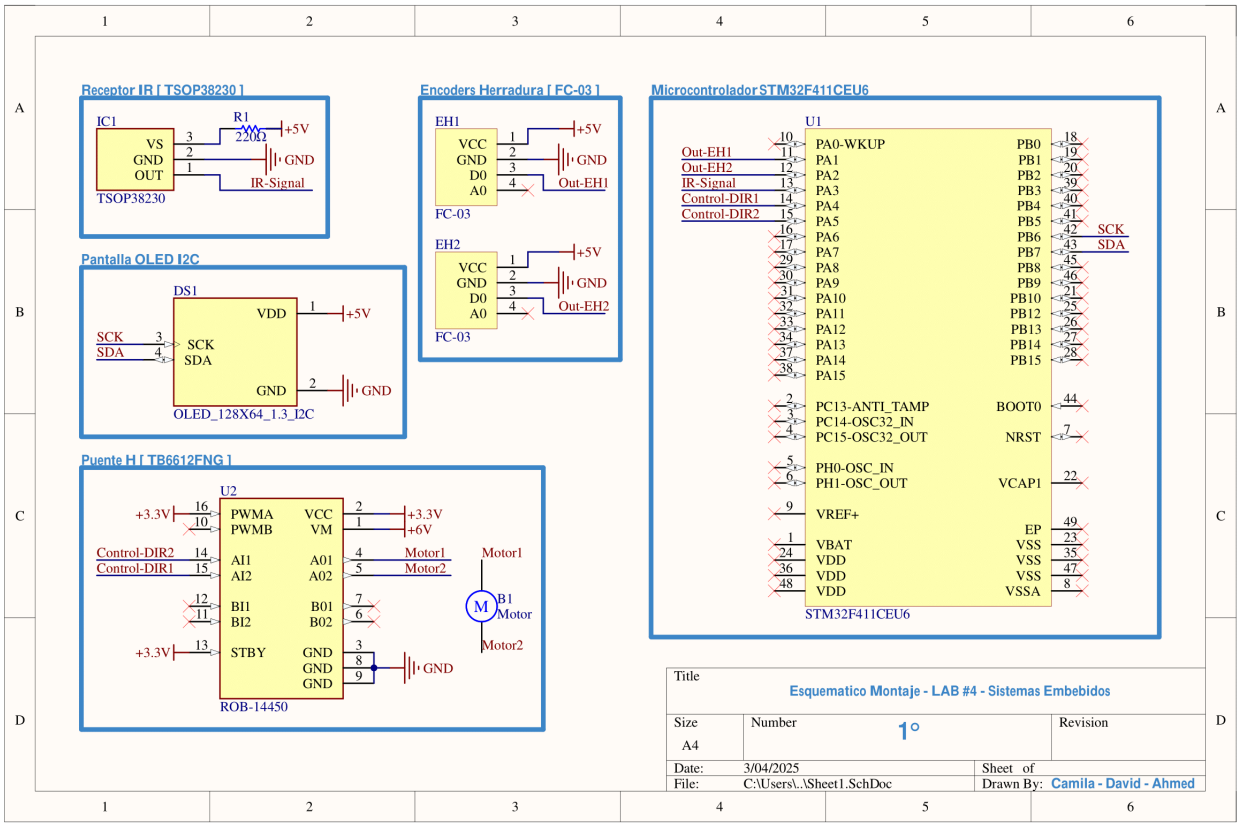
¿COMO FUE EL USO DEL TIM?

Implementamos las librerías del TIM para estar contando constantemente desde que se inicia el programa, gracias a esto pudimos acomodar el código a nuestros datos, con ayuda del umbral ajustamos los datos para que el TIM haga el conteo de manera correcta.

Usamos:

TTIM Base_Start, TIM_SET_COUNTER, TIM_GET_COUNTER.

ESQUEMATICO DE LAS CONEXIONES:



¿QUE SE REALIZÓ?

Implementamos el esquemático de conexiones en la protoboard tal cual como se muestra, realizamos la visualización de la señal de pulsos tal cual como se mostro anteriormente, ya revisado los pulsos comenzamos con la implementación del código, donde lo primero fue que solo imprimiera los valores del pulso de comando “los 8 primeros bits del pulso incluyendo el start”, cuando en la pantalla oled y en el TIM confirmamos que los valores estuvieran correctos, comenzamos ya con la implementación del puente H, que básicamente fue la señal del pulsos darle las condiciones que queríamos que el programa cumpliera, como lo era el comando de ir a la derecha a la izquierda etc. Una vez implementado esto empezamos a enviar la señal del control y observar como estaba funcionando el motor con el puente H, culminada esta sección se comenzó con la trasmisión de datos por el puerto USB, transmisión con la que se obtendrían las RPM del motor gracias a la información de los encoders.

¿QUÉ SUCEDIÓ?

Ahora en cuanto la implementación en la protoboard, fue bastante sencillo no hubo ninguna dificultad, lo único para resaltar es las conexiones del driver y las conexiones en general de la placa y del puente H, ahora en cuanto la implementación del código y la transmisión de datos, solo tuvimos errores fue en la definición de los pines a nivel de código, debido a que usamos un código anterior, y no caímos en cuenta en unas variables, solucionando esto la impresión de los comandos en la pantalla ya fue correcta.

Ahora una de las problemáticas, al momento de mandar los comandos del control al IR, todo funciona correctamente hasta que se alimenta el puente H, lo que sucedió es que la rueda gira normal y los comandos hacen lo que deben hacer, el problema esta en que en la pantalla al insertar un comando hacia el que era, pero empezó a imprimir otros comandos, otro error era que al hacer los cambios de giro el motor se detenía unos dos segundos y ahí si empezaba a girar, resultamos haciendo observaciones de código pero el error no estaba ahí. Ya que nos percatamos que el motor cada vez que va aumentando su velocidad pide corriente, al hacer esto los comandos comienzan a fallar, pero cuando el motor no ha llegado a su máxima velocidad comienza a afectar a los comandos.

DIAGRAMA DE FLUJO

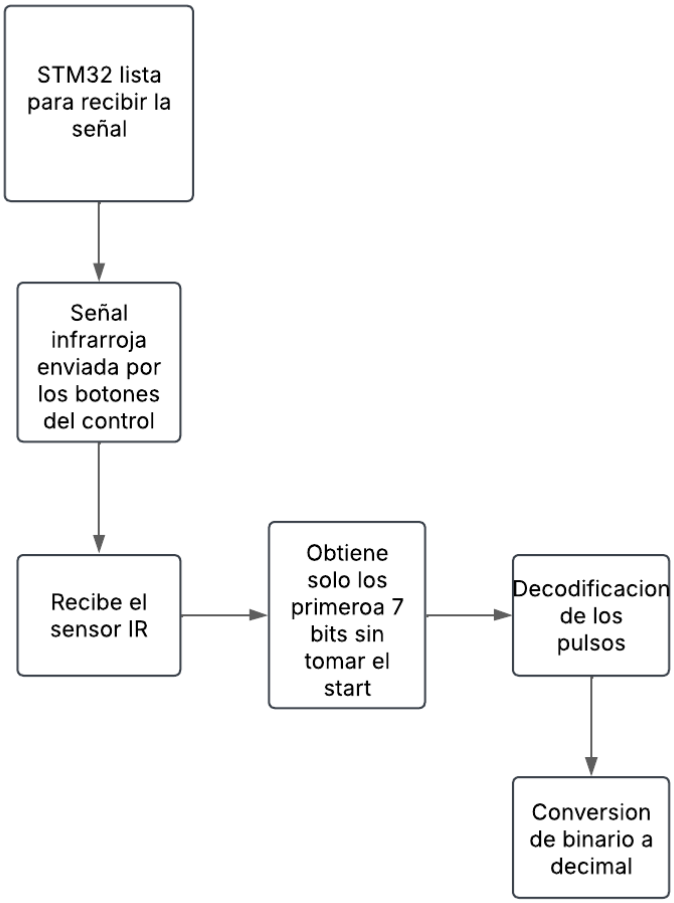


Imagen 1: Diagrama de flujo de la trasmisión y codificación de datos

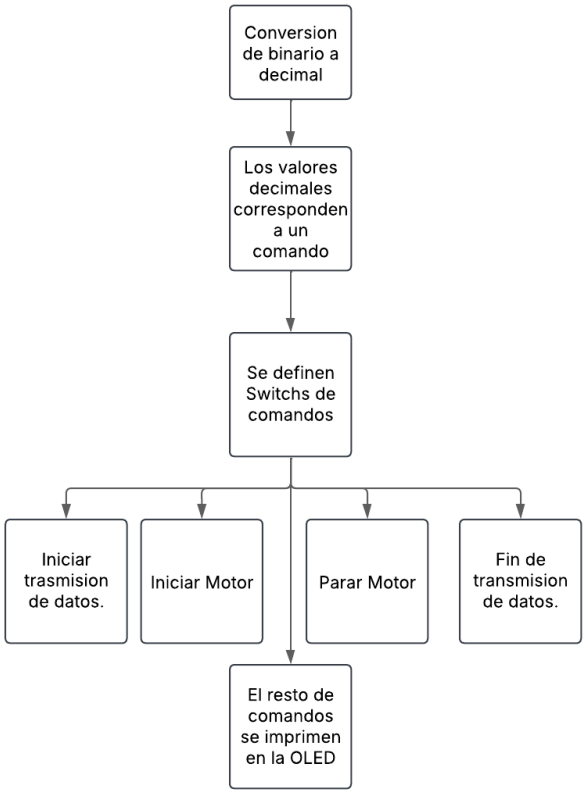


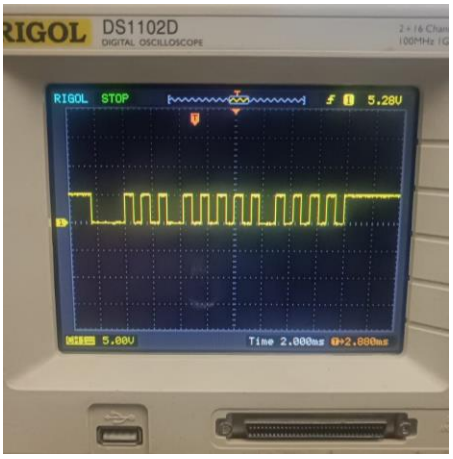
Imagen 2: Diagrama de flujo del puente H y Rpm

CONCLUSIONES

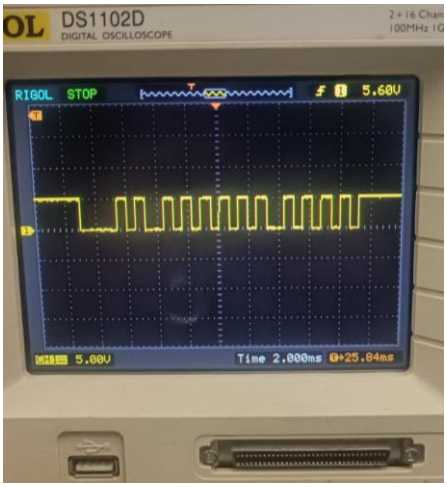
- La implementación de la base de giro y circuito electrónico fue el esperado.
- El uso de las librerías del TIM y la definición correctas de estas mismas sirvieron y ayudaron a la obtención correcta de los datos y la comprensión de estos mismos.
- El sensor infrarrojo IR demostró un gran desempeño en su uso, en los anexos se presentan todos los pulsos de los comandos recibidos por el sensor mostrando la precisión de la señal transmitida por el control, como así mismo la recibida por el receptor.
- En cuanto los comandos se pueden destacar que hay un mínimo margen de error en los pulsos observados en el osciloscopio “Anexos”, aun así, gracias a la implementación de los umbrales este margen no fue de afección.

- La importancia de saber el voltaje y corriente de los módulos y los motores, debido que estos generan ruidos en la señal o picos de corriente que pueden influir o alterar en el desarrollo del procedimiento del sistema.

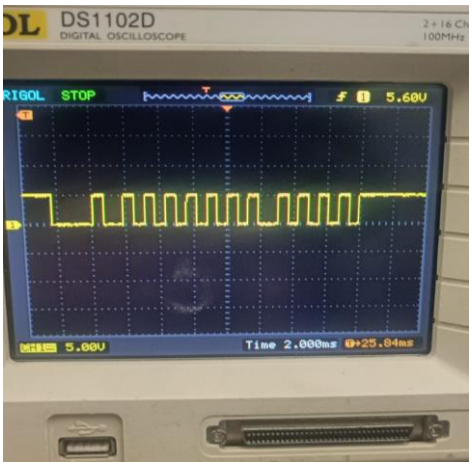
ANEXOS:



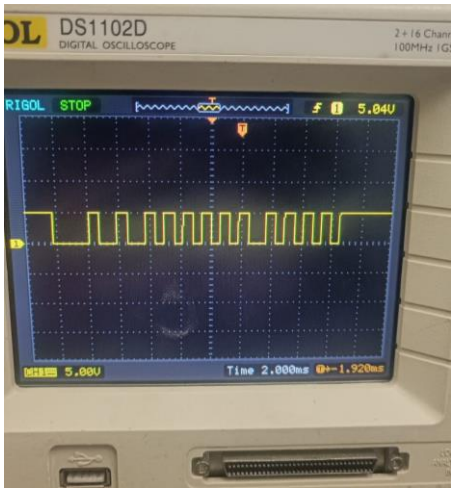
Anexo 1.1. Tren de pulsos. Botón 1



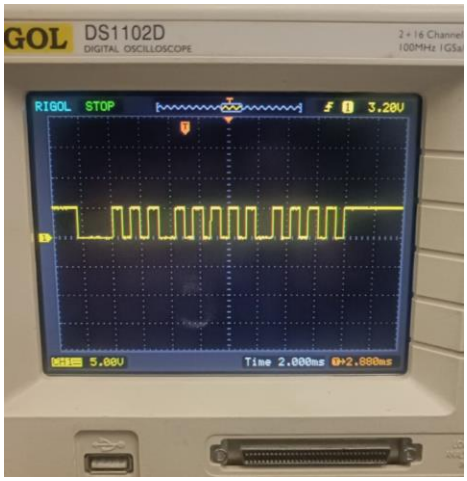
Anexo 1.3. Tren de pulsos. Botón 3



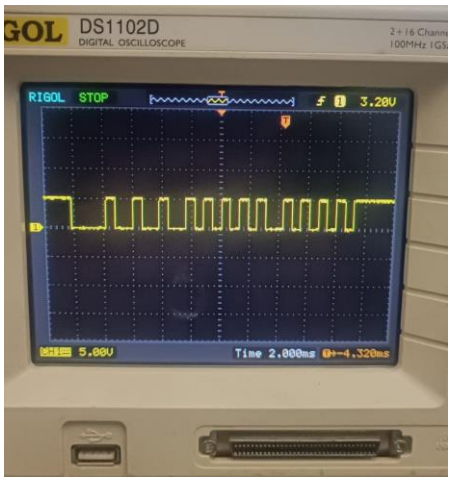
Anexo 1.2. Tren de pulsos. Botón 2



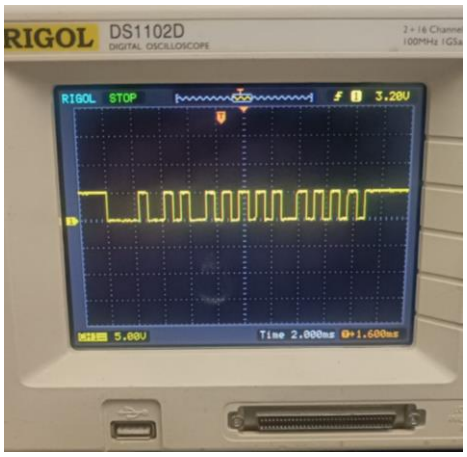
Anexo 1.4. Tren de pulsos. Botón 4



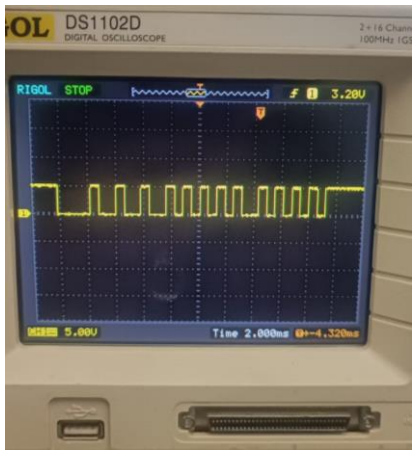
Anexo 1.5. Tren de pulsos. Botón 5



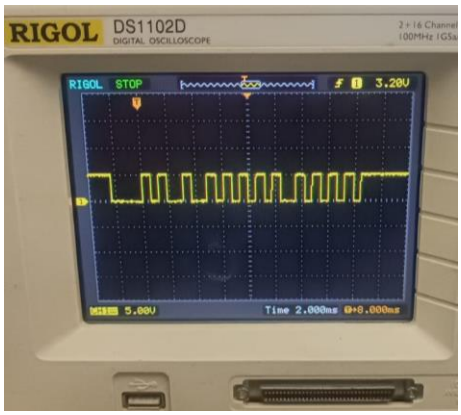
Anexo 1.8. Tren de pulsos. Botón 8



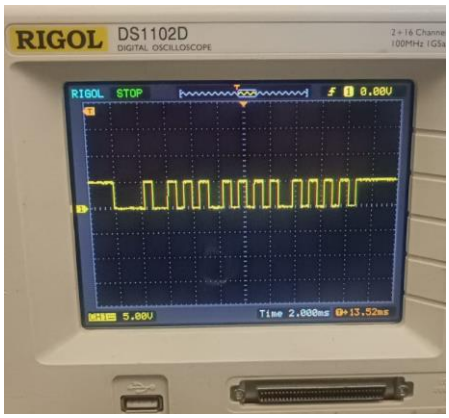
Anexo 1.6. Tren de pulsos. Botón 6



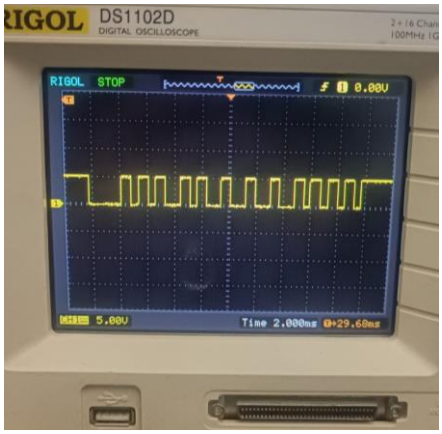
Anexo 1.9. Tren de pulsos. Botón 9



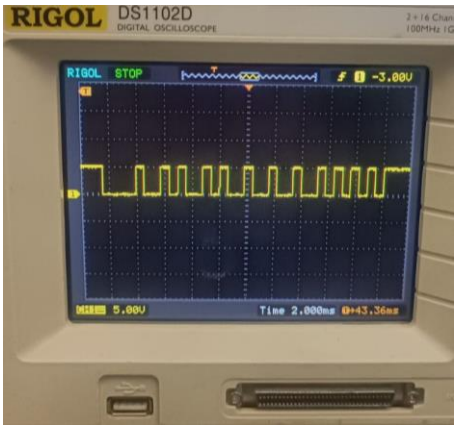
Anexo 1.7. Tren de pulsos. Botón 7



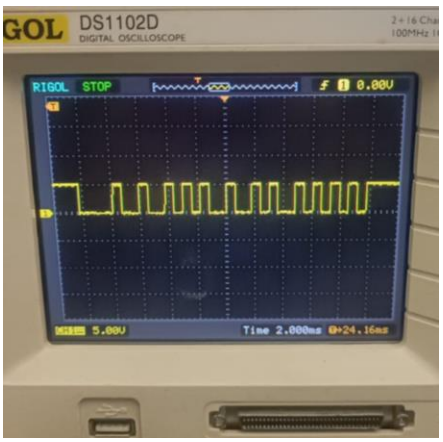
Anexo 2.1. Tren de pulsos. Botón 0



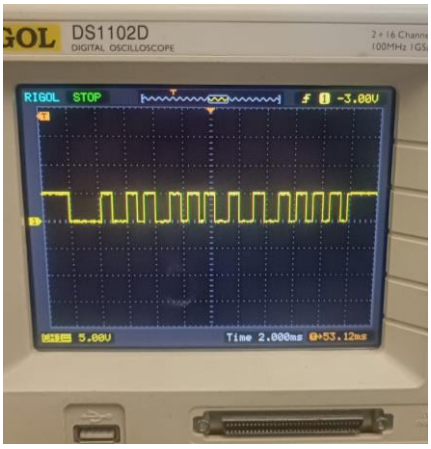
Anexo 2.2. Tren de pulsos. Botón UP



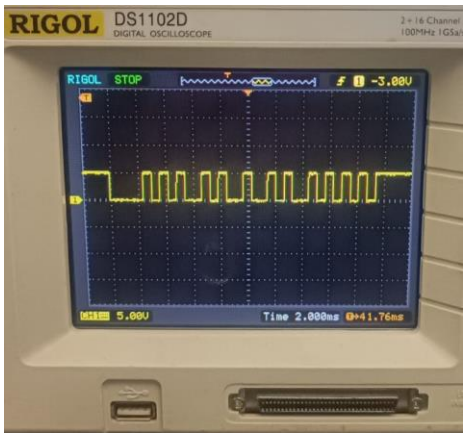
Anexo 2.4. Tren de pulsos. Botón DOWN



Anexo 2.3. Tren de pulsos. Botón RIGHT



Anexo 2.5. Tren de pulsos. Botón OK



Anexo 2.3. Tren de pulsos. Botón LEFT