

# Exploración de conceptos de conversión A/D con Raspberry Pi Pico 2W

Caraballo Cárdenas Andres Felipe [est.andres.caraballo@unimilitar.edu.co](mailto:est.andres.caraballo@unimilitar.edu.co)

**Resumen** – En este laboratorio los estudiantes tendrán la oportunidad de trabajar con el Raspberry Pi Pico 2W para entender cómo una señal analógica puede convertirse en datos digitales dentro de un sistema de comunicación. A través del uso del ADC (convertidor analógico–digital), aprenderán a capturar diferentes señales y a procesarlas con funciones como `read_u16()`, entendiendo también cómo se interpreta la resolución real de 12 bits. Las prácticas incluyen tomar muestras de señales y analizarlas tanto en MATLAB como en Python. Para ello, se generarán archivos en formato CSV, se crearán gráficas y se compararán los resultados obtenidos entre lo que mide el microcontrolador y lo que se reconstruye en los programas, así mismo se realizarán ejercicios de análisis estadístico que les permitirán calcular el promedio, la desviación estándar y observar la distribución de las muestras mediante histogramas. De esta forma, podrán validar y contrastar los datos procesados en el computador con los valores obtenidos directamente desde la tarjeta.

## I. INTRODUCCIÓN

La conversión de señales analógicas a digitales (A/D) es un elemento fundamental en los sistemas de comunicación modernos, ya que posibilita transformar información continua en valores discretos que pueden ser procesados, almacenados o transmitidos por equipos electrónicos. En este laboratorio se trabaja con el microcontrolador Raspberry Pi Pico 2W con el propósito de comprender los principios de muestreo y cuantización, además de analizar los datos obtenidos mediante el apoyo de herramientas computacionales.

A lo largo de las prácticas, los estudiantes aprenden a registrar señales analógicas, traducirlas a su correspondiente representación digital e interpretar los resultados utilizando MATLAB y Python. Esto permite unir la teoría con la experimentación en torno al uso de conversores A/D, la reconstrucción de señales y la aplicación de métodos estadísticos sobre las mediciones, lo cual contribuye al fortalecimiento de habilidades esenciales dentro del ámbito de las telecomunicaciones digitales.

## II. PROCEDIMIENTO

El laboratorio se organiza en tres etapas principales. En la primera, los estudiantes trabajan directamente con el

convertidor A/D del Raspberry Pi Pico 2W, eligiendo una entrada adecuada y aplicando un voltaje dentro del rango permitido (0 a 3.3 V). A través de la función `read_u16()`, se observa cómo ocurre la conversión y los resultados se representan con el Plotter de Thonny, complementando la práctica con ejemplos que muestran la relación entre los 16 bits leídos y la resolución efectiva de 12 bits. En la segunda fase, la atención se centra en el muestreo de señales. Para ello, se utilizan programas en MATLAB y Python que permiten capturar datos, graficarlos y compararlos con las señales reconstruidas a partir de archivos CSV. Como entrada se emplea una señal de 3 Vpp con un nivel DC de 1.6 V, ajustada desde el generador de funciones. La tercera parte está dedicada al análisis estadístico. En esta sección, se conecta el conversor a distintos valores de tensión continua, se realizan cinco mediciones y los datos recolectados se procesan en MATLAB o Python. A partir de ello se calculan parámetros como la media, la desviación estándar y se construyen histogramas, contrastando los resultados con los obtenidos directamente en el microcontrolador.

## III. ANÁLISIS DE RESULTADOS

Al comenzar la práctica se armó un montaje con el potenciómetro, con el propósito de observar la variación del voltaje dentro del rango de 0 a 3.3 V. Dichos cambios pudieron comprobarse mediante la herramienta Plotter del entorno Thonny, que permitió generar una gráfica en tiempo real y visualizar cómo pequeñas modificaciones en el ajuste del potenciómetro se reflejaban claramente en la señal.

Como complemento, se llevó a cabo un ejercicio adicional de conversión empleando tres valores diferentes a los propuestos en el anexo de la guía, con el fin de ampliar la comprensión del procedimiento.

### Ejemplo 4: Escala (512)

```
Code12 = 512 = 0010 0000 0000 (bin, 12b)
raw16 = 512 << 4 = 8192 = 0010 0000 0000 0000 (bin, 16b)
raw16 >> 4 = 0010 0000 0000 (bin, 12b) = 512
```

### Ejemplo 5: Escala (256)

```
code12 = 256 = 0001 0000 0000 (bin, 12b)
raw16 = 256 << 4 = 4096 = 0001 0000 0000 0000 (bin, 16b)
```

$\text{raw16} \gg 4 = 0001\ 0000\ 0000\ (\text{bin}, 12\text{b}) = 256$

### Ejemplo 6: Escala (128)

$\text{code12} = 128 = 0000\ 1000\ 0000\ (\text{bin}, 12\text{b})$

$\text{raw16} = 128 \ll 4 = 2048 = 0000\ 1000\ 0000\ 0000\ (\text{bin}, 16\text{b})$

$\text{raw16} \gg 4 = 0000\ 1000\ 0000\ (\text{bin}, 12\text{b}) = 128$

Posteriormente, en el laboratorio se realizó una comparación entre una señal ideal y una señal generada con el generador de funciones, utilizando para ello las muestras registradas en el archivo CSV obtenido de la Raspberry Pi Pico 2W.

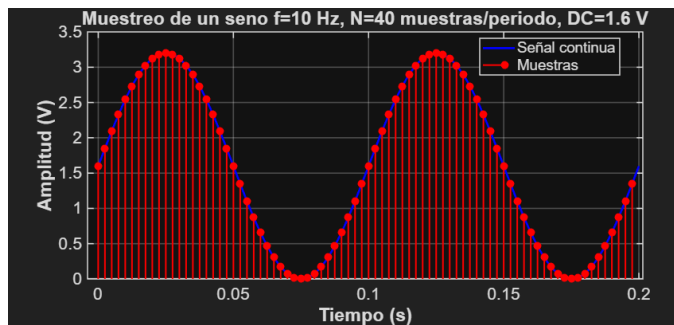


Fig.1 Señal ideal generada en Matlab con nivel DC y frecuencia de 10 Hz

Al realizar la comparación entre la señal teórica y la señal obtenida experimentalmente con el generador de funciones y capturada por la Raspberry Pi Pico 2W, se aprecia que ambas conservan la misma tendencia general. No obstante, mientras que la señal ideal exhibe un muestreo totalmente uniforme, la experimental presenta un espaciado entre muestras que no es constante. Esta diferencia se debe a que, en condiciones reales, el proceso de adquisición no ocurre de manera perfectamente regular: el microcontrolador introduce ligeros retrasos en la lectura y el almacenamiento de los datos, provocando que algunas muestras se registren más cercanas entre sí y otras más separadas. A ello se suman efectos como la cuantización propia del ADC, la presencia de ruido y pequeñas variaciones introducidas por el generador de funciones.

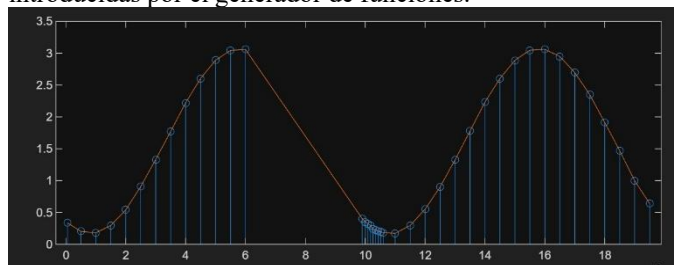


Fig.2 Señal experimental obtenida del csv

En la tercera parte de la práctica se trabajó con diferentes valores de voltaje para calcular la media y la desviación estándar, con el objetivo de analizar la variabilidad presente en las mediciones. Los resultados obtenidos se organizaron en una tabla de datos, la cual se presenta a continuación.

Test	V in (DC)	Media	Desviación estándar	Nombre de los archivos
1	1.5V	0.3295V	0.0129V	Medicion1
2	2V	0.4393V	0.0165V	Medicion2
Test	V in (DC)	Media	Desviación estándar	Nombre de los archivos
3	2.5V	0.5498V	0.0206V	Medicion3
4	3V	0.6592V	0.0249V	Medicion4
5	3.2V	0.7030V	0.0268V	Medicion5

Tabla 1. Análisis estadístico

De esta manera, se procedió a validar los resultados tomando como referencia las mediciones obtenidas. Para ello, se realizó una comparación entre los valores procesados en MATLAB y aquellos calculados mediante el programa en MicroPython. El análisis mostró que las diferencias entre ambos conjuntos de datos fueron mínimas. Estas pequeñas variaciones se deben principalmente al redondeo en los cálculos, a la forma en que cada entorno gestiona la precisión numérica y a la naturaleza discreta del muestreo, que puede generar ligeras discrepancias.

```
>> calculos
Media del voltaje: 0.65924
Desviación estándar del voltaje: 0.024907
```

Fig.3 Media y desviación estándar de la medición 4

Así como se verificaron los resultados de la **media** y la **desviación estándar**, también se utilizó **MATLAB** para elaborar los **histogramas** correspondientes a las mediciones obtenidas. Con ello se generó la siguiente gráfica, que facilita una mejor comprensión de la **distribución de los valores** capturados por el conversor A/D.

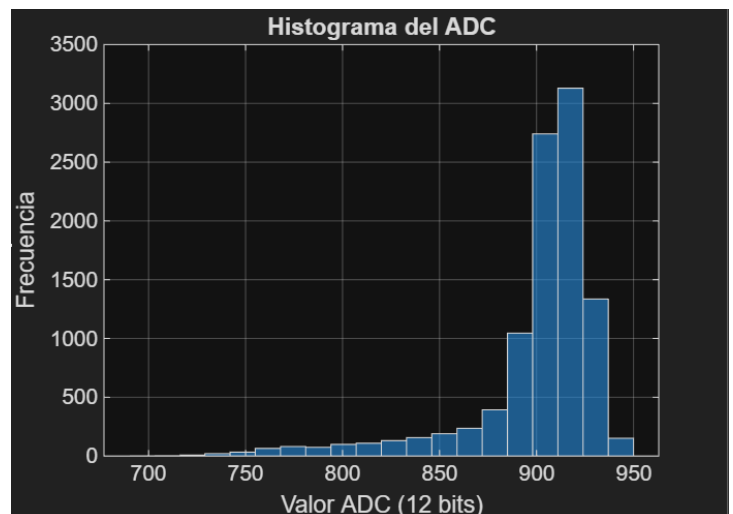


Fig.4 Grafica histograma de prueba 4

<https://randomnerdtutorials.com/raspberry-pi-pico-analog-inputs-micropython/>

[3] Circuit Schools Staff. (2021, diciembre 10). *How to use ADC on Raspberry Pi Pico in detail with MicroPython example*. Circuit Schools. Recuperado de <https://www.circuitschools.com/how-to-use-adc-on-raspberry-pi-pico-in-detail-with-micropython-example/>

El histograma del ADC, con resolución de 12 bits, muestra en el eje X los valores entregados por el conversor y en el eje Y la frecuencia con la que cada valor aparece. Los resultados evidencian que la mayoría de las mediciones se concentran en un rango estrecho entre 890 y 940, con un pico marcado entre 900 y 920. Esto indica que la señal medida es constante, ya que no se distribuye a lo largo de todo el rango disponible del conversor (0–4095), sino alrededor de un único punto. Sin embargo, se observa cierta dispersión hacia valores inferiores a 900, lo cual se atribuye al ruido o a las limitaciones de precisión del ADC y a ligeras variaciones en la fuente de señal.

#### IV. CONCLUSIONES

- La práctica permitió comprender cómo un conversor analógico–digital (ADC) transforma señales continuas en valores discretos, mostrando en la Raspberry Pi Pico 2W un funcionamiento estable y confiable dentro de su rango de operación.
- Se evidenció que, aunque la señal ideal y la experimental mantienen la misma forma general, la real siempre presenta pequeñas variaciones. Estas diferencias se deben al ruido, a la cuantización del ADC y a los tiempos de muestreo no perfectamente uniformes.
- El análisis con MATLAB resultó clave para visualizar, comparar y validar los datos obtenidos. Estas herramientas facilitaron la reconstrucción de señales, la creación de gráficas y la generación de histogramas que permitieron entender mejor la distribución de los valores medidos.
- Los cálculos de media y desviación estándar confirmaron que las mediciones son estables, con baja dispersión alrededor del valor central. Esto demuestra que, a pesar de las limitaciones del hardware, el sistema es adecuado para el estudio del muestreo y la cuantización.

#### V. REFERENCIAS

[1] Alam, M. (2024, noviembre 28). *How to use ADC in Raspberry Pi Pico | ADC example code*. How2Electronics. Recuperado de <https://how2electronics.com/how-to-use-adc-in-raspberry-pi-pico-adc-example-code/>

[2] Random Nerd Tutorials. (s. f.). *Raspberry Pi Pico: Read Analog Inputs (MicroPython)*. Recuperado de