Periferico ADC

El conversor ADC es un módulo que convierte señales de tensión analógicas a valores enteros en un rango determinado; la resolución de la conversión está determinada por el tamaño de bits del conversor, es decir, si el ADC es de 16 bits, el número máximo a obtener en una conversión será de $2n^{16} - 1 = 65535$.

ESQUEMÁTICO: Este diagrama representa la solución a realizar, solo se ha agregado en sí el circuito para el sensor de Luz que es alimentado por la tensión ofrecida desde el pin de 3.3v (3v3 o 3V) de la tarjeta de desarrollo, para crear los demás elementos del circuito deberá remitirse a los conocimientos previos de los anteriores talleres.

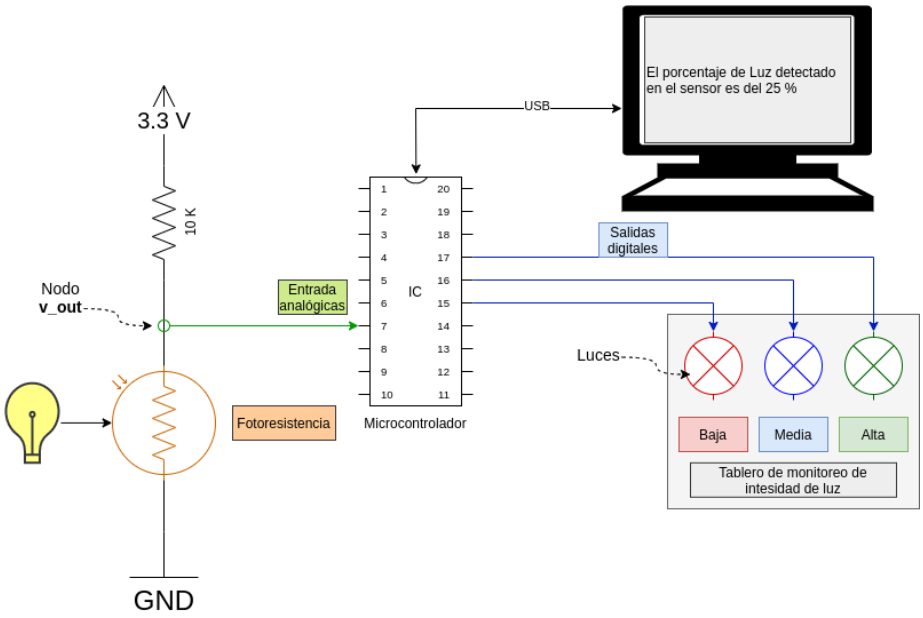

Figure 1: Esquema circuital

PROGRAMA A REALIZAR:

Traduzca este algoritmo a una tecnología/lenguaje, en este caso a micropython, si tiene dificultades para crear las diferentes instrucciones, remítase a algún tutorial del lenguaje python3.

RECOMENDACIÓN PARA LA REALIZACIÓN DEL PROGRAMA:

Para que pueda resolver este problema podrá seguir estas recomendaciones:

1. Realice el circuito planteado e identifique el nodo de voltaje *v_out*
2. Como es un sensor de Luz deberá provocar los dos casos extremos: cuando hay mucha luz 100 % o poca luz 0 %; para provocar estas dos situaciones

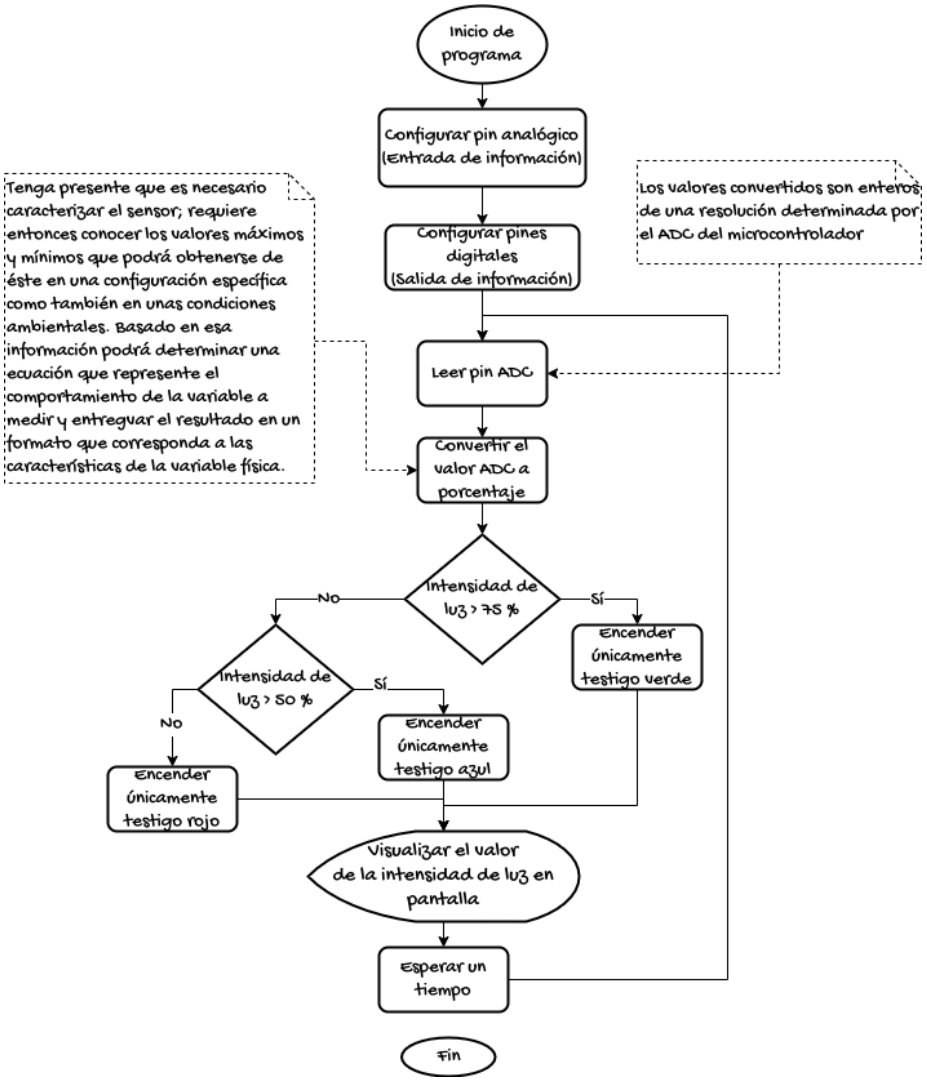

Figure 2: Diagrama de flujo

use sus manos para tapar el sensor y que no reciba luz 0% y por medio de la linterna del celular estimule el sensor para obtener el 100 % de la intensidad de la luz.

3. Simultáneamente al paso 2 mida el voltaje con un multímetro entre el nodo *v_out* y *GND* observe el cambio de voltaje que existe en proporción de la intensidad de luz percibida por el sensor e indique qué tipo de comportamiento tiene el voltaje en función de la luz recibida.
4. Teniendo presente que el ADC convierte señales de voltaje a valores de unidades enteras haga un pequeño programa que imprima el valor del voltaje convertido en esas unidades (ver abajo ejemplos de código según tarjeta de desarrollo), para ello acompañe su código de la función `print()` en un `while True` acompañado con un `time.sleep(segundos)` y mire la relación que existe entre el voltaje medido con el multímetro y las unidades imprimidas en consola; determine la relación existente de las unidades en función del voltaje convertido por el ADC.
5. Teniendo en cuenta los dos casos extremos realizados desde el paso 2, tome el valor de unidades obtenido para ambos casos (0% de luz y 100 % de luz), recuerde que estos valores se observan en la terminal y deben corresponder a la acción de aplicar los dos casos extremos del paso 2.
6. Al tener los dos casos extremos que puede llamar P_1 y P_2 donde cualquier punto es una coordenada de la forma $P(unidades\_adc, porcentaje\_luz)$ y que en un plano cartesiano $P(x, y)$, $x = unidades\_luz$ y $y = porcentaje\_luz$ trace una recta que una a esos dos puntos y construya una ecuación de la recta. Recuerde que la ecuación de la recta es de la forma $y(x) = mx + b$, la pendiente $m = \frac{y_2 - y_1}{x_2 - x_1}$ y que el punto de corte $b = y_1 - mx_1 = y_2 - mx_2$
7. Con la información obtenida en el paso 6 construya una función en python que haga esa transformación, pruebe su funcionamiento estimulando el sensor como fue planteado en el paso 2 y si es coherente el resultado, construya el algoritmo planteado en este taller (PROGRAMA A REALIZAR)

```
# Fije el valor de la pendiente m
# m = ...
# Fije el valor de B
# b = ...
# Los anteriores valores los puede sustituir directamente en la ecuación.

def calcularPorcentajeDeLuz(val_adc):
    return m*val_adc + b
```

USO DE ADC EN MICROPYTHON

Para decidir cuál pin va usar para la conexión deberá tener presente el *pinout* de la tarjeta de desarrollo en cuestión, revise que el pin a seleccionar debe permitir la configuración de un periférico ADC.

- ADC ESP32

```
# On the ESP32, ADC functionality is available on pins 32-39
# (ADC block 1) and pins 0, 2, 4, 12-15 and 25-27 (ADC block 2).

from machine import Pin
from machine import ADC

pin = Pin(pin_number, Pin.IN)
adc = ADC(pin)        # create an ADC object acting on a pin
val = adc.read_u16()  # read a raw analog value in the range 0-65535
```

- ADC Raspberry Pico

```
# RP2040 has five ADC channels in total, four of which are 12-bit
# SAR based ADCs: GP26, GP27, GP28 and GP29. The input signal for
# ADC0, ADC1, ADC2 and ADC3 can be connected with GP26, GP27, GP28,
# GP29 respectively (On Pico board, GP29 is connected to VSYS).
# The standard ADC range is 0-3.3V. The fifth channel is connected
# to the in-built temperature sensor and can be used for measuring
# the temperature.

from machine import ADC, Pin
adc = ADC(Pin(pin_number))     # create ADC object on ADC pin
val = adc.read_u16()         # read value, 0-65535 across voltage range 0.0v - 3.3v
```