Siguelíneas

- 6 Módulo siguelíneas
- 6.2 TLC1543
- 6.3 TLC1543.py y VARIABLES.py
- 6.4 Test-Sigue-lineas
- 6.5 Siguelineas

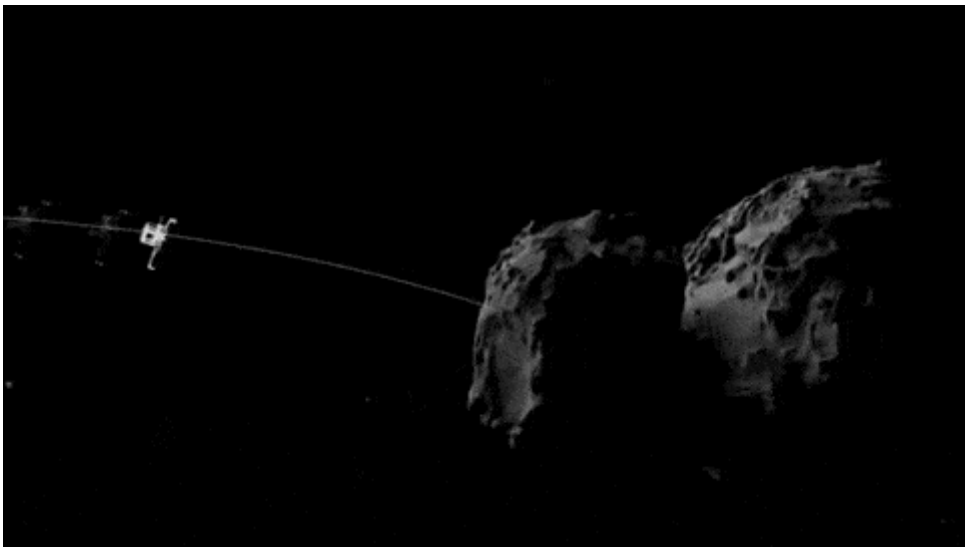
Copyright 2026

6 Módulo siguelíneas

ORBITAS

Esta claro que un rover de verdad no va a tener que seguir una línea pintada en el suelo. Pero, las sondas que lo transportan a los astros, sí que tiene que seguir unas órbitas o líneas imaginarias. Nosotros lo vamos a simular con una línea pintada en el suelo.

La programación básica es la misma siempre: si te desvías, corrige.



Philae rover's orbit in comet 67P

Pero no es fácil elegir la órbita :

https://www.youtube.com/embed/gYJsMBabjVY

NUESTRO ROBOT SIGUELINEAS

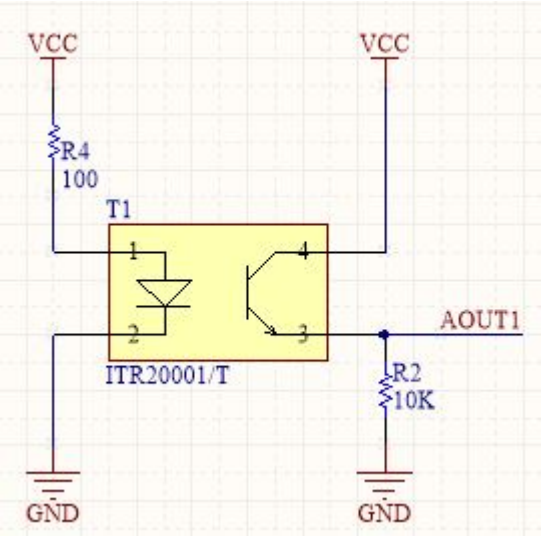
Nuestro robot tiene un módulo con 5 sensores al color :



Tiene un funcionamiento similar al Sensor obstáculos IR. El receptor tiene un sensor de reflexión de infrarrojos ITR20001/T. Un led emite luz IR contínuamente, la luz infraroja es reflejado por un obstáculo y lo recibe el receptor.

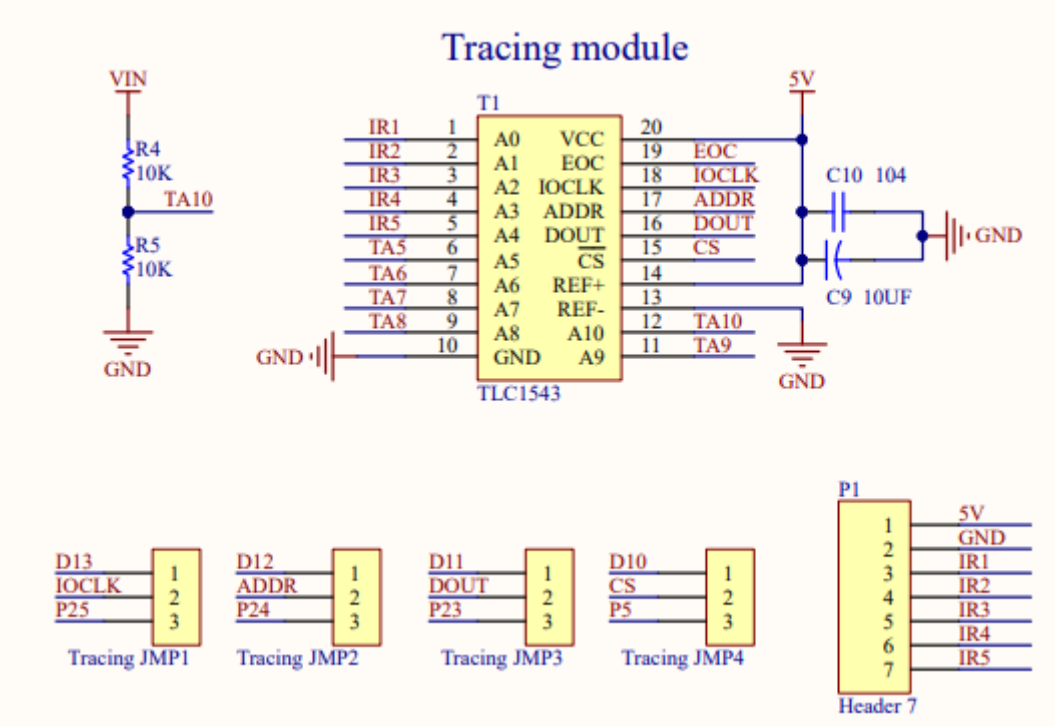
La salida del sensor es analógica y es sensible al color y la distancia del objeto detectado.

Tiene 5 canales de sensores. Chequeandolos se puede juzgar la posición de la línea oscura que esté en el suelo.



6.2 TLC1543

Este robot no nos lo pone fácil con el siguelíneas ¿Por qué? Porque los 5 sensores (IR1..IR5) están conectados **a un conversor analógico digital TLC1543** tal y como puedes ver en su esquema eléctrico:



¿Cómo está conectado con GPIO?

Pues con estos números:

- CS en GPIO 5
- Clock en GPIO 25
- Address en GPIO 24
- DataOut en GPIO 23

¿Cómo funciona este chip?

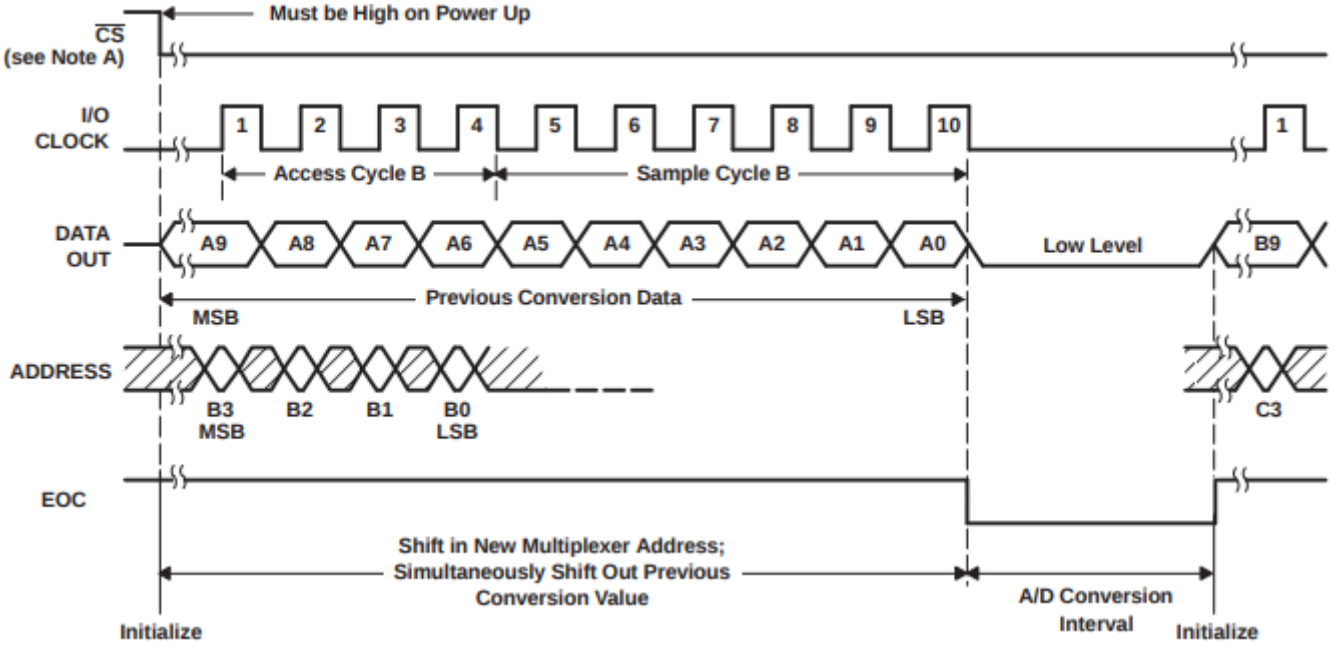
Pues léete su manual de instrucciones □□□□□□ aburrido ¿verdad?

Te lo resumimos :

1. Este chip se activa por nivel bajo del **CS**.
2. En ese momento LEE LA DIRECCIÓN definida por **ADDRESS** (*empezando por el bit más alto MSB)* donde ADDRESS es un número entre 0 y 4 en binario que corresponde canal o sensor infrarrojo que se quiere leer A0 hasta A4 (que corresponden a los sensores IR1 hasta IR5).
3. Los primeros 4 pulsos de **CLOCK** son para leer ADDRESS (*en el flanco de subida*). En la ilustración Accesss Cycle B.
4. Los otros 6 *no valen para nada*. En la ilustración Sample Cycle B.
5. Los siguientes 10 pulsos se emite por **DATAOUT** (*empezando por el bit más alto MSB)* el valor leído del sensor de infrarojos que has seleccionado en ADDRESS. En la ilustración Previous Conversion Data, es decir, está sacando la conversión de los valores leidos anteriormente que no están en la ilustración.

> **Mentirijillas**: Realmente el punto 4 no es verdad, lo que pasa es que si has leído el punto 5 durante los 10 pulsos de reloj está sacando la lectura del IR definido por ADDRESS de los 10 anteriores. Lo que es verdad es que la primera lectura de todas, esos 6 pulsos no valen para nada.

Copyright 2026



Extraido del Datacheet

- Los 4 primeros pulsos de reloj por flanco de subida leen ADDRESS: B3 B2 B1 y B0 mientras tanto por DATAOUT está sacando los valores A9..A0 definido en una anterior ADDRESS que no vemos (los 10 primeros pulsos de CLOCK no tiene sentido esos valores de DOUT).
- Los rayados significan que igual da lo que haya pues no lo lee. Por ejemplo entre el pulso 5 y 10 del rejoj, no se está leyendo ADDRESS.
- Si te fijas, en los siguientes 10 pulsos de CLOCK (sólo aparece el primer pulso) ya empieza a salir por DATAOUT los valores del sensor definidos en ADDRESS como B3B2B1B0.
- Entre 10 pulsos de reloj y los otros 10 se necesita un tiempo de conversión A/D Conversion interval.

Copyright 2026

6.3 TLC1543.py y VARIABLES.py

Tal y como hemos visto en la teoría del TLC1543 ¿Cómo está conectado? añadimos estas líneas al archivo **VARIABLES.py**

```
##SENSOR SIGUELINEAS

CS = 5
Clock = 25
Address = 24
DataOut = 23

##SENSOR SIGUELINEAS

CS = 5
Clock = 25
Address = 24
DataOut = 23
```

Script Damebit

En la teoría del TLC1543 ¿Cómo funciona? tenemos que obtener el bit de una posición dada de un número dado. Aquí hay un pequeño script para hacerlo (dale al play para ejecutarlo):

https://repl.it/@deleyva/ObtenerBitEntero?lite=true

Copyright 2026

TLC1543.py

Tal y como hemos visto en la teoría del TLC1543 ¿Cómo funciona? podemos hacer una librería que tenga una función **SENSORLINEA(cual)** que nos devuelva el valor que lee el sensor *cual*:

- Importamos las variables de **VARIABLES.py**
- Luego realizamos una función **SACADIRECCION** que active la salida ADDRESS según sus bits basándonos en la función **Damebit** que hemos visto.
- Activamos 4 golpes de reloj sacando la dirección **ADDRESS** con la función SACADIRECCION
- Hacemos 6 pulsos de **CLOCK** perdidos
- Hacemos 10 pulsos de CLOCK pero leyendo el valor **DATAOUT** y convirtiendo esos bits en un número decimal, ese será el valor que devolverá la función **SENSORLINEA(cual)**
- Grabamos esto en un archivo TLC1543.py

```
import RPi.GPIO as GPIO
import time

from VARIABLES import *

#########################################################
#función de manipulación de bits
#ver https://repl.it/@javierquintana/ObtenerBitEntero
#########################################################
def SACADIRECCION(x,n):
  if (x & (1<<n)):
    GPIO.output(Address,GPIO.HIGH)
  else:
    GPIO.output(Address,GPIO.LOW)

#########################################################
#función de obtener lectura del sensor siguelíneas
#cual = el número del siguelíneas a leer 0-4
#########################################################
def SENSORLINEA(cual):
  #activo el chip
```

```
    GPIO.output(CS,GPIO.LOW)
    for i in range(4):
      #Pongo en Address el bit de cual empezando por MSB
      SACADIRECCION(cual,3-i)
      #Flanco de subida de Clock para que lo lea
      GPIO.output(Clock,GPIO.LOW)
      GPIO.output(Clock,GPIO.HIGH)
    #ahora 6 pulsos perdidos
    for i in range(6):
      GPIO.output(Clock,GPIO.LOW)
      GPIO.output(Clock,GPIO.HIGH)
    #vamos a darle un tiempo para que calcue la convesión A/D
    #A/D Conversion Interval =
    time.sleep(0.001)
    #leemos el número
    #formula valor = sumatorio (potenciasde2 * bit leido)
    #potenciasde2 = 2 elevado al peso del bit
    #el peso del primer bit es MSB luego 9 y acaba en 0 o LSB
    valor=0
    for i in range (10):
      GPIO.output(Clock,GPIO.LOW)
      GPIO.output(Clock,GPIO.HIGH)
      valor=valor+GPIO.input(DataOut)*(2**(9-i))
    #desactivamos el chip
    GPIO.output(CS,GPIO.HIGH)
    return valor
```

Copyright 2026

6.4 Test-Sigue-lineas

Podemos hacer un test para ver cómo funciona este siguelíneas y cómo funciona la librería

- Importamos la librería TLC1543 creada anteriormente
- Vamos leyendo cada uno de los sensores.
- Que salga por pantalla los valores leídos.

https://www.youtube.com/embed/MFWnOyL_nzU

¿Te atreves?:

Fichero Testsiguelineas.py

```
import RPi.GPIO as GPIO
import time

import TLC1543

while True:
    for i in range(5):
        x=TLC1543.SENSORLINEA(i)
        print (" Sensor",i,"= ",x,end="")
    print(" ");
    time.sleep(0.5)
```

6.5 Siguelineas

Vamos a realizar un programa que siga la línea: * Pero **¡¡AL REVÉS!!!** ¿por qué marcha hacia atrás? (o sea, la cámara mira hacia la parte trasera del sentido de la marcha): mira al final de la página.

Fijaremos de antemano una velocidad pequeña de 25% y un incremento de velocidad de diferencia en los motores de 10% cuando no está centrado para que gire hasta que se centre. Veamoslo con un vídeo:

https://www.youtube.com/embed/rsdQ9fGOl-I

Solución

La solución es fácil con la librería TLC1543.py donde la función **SENSORLINEA(cual)** nos da el valor que lee los sensores IR. Recuerda que TLC1543.py en la misma carpeta que vamos a crear este programa y las incorporamos en el programa con **import**. * También incorporamos las variables definidas en **VARIABLES.py**

¿Te atreves?

- Leemos la lectura de los 5 sensores
- Ajustamos la velocidad de los motores según si hay lectura de línea negra y donde
- La potencia PWM no puede pasar de 0 a 100 por eso limitamos los valores
- Si no hay linea negra que vuelva hasta que recupera la línea negra:

Fichero Siguelineas.py

```
import RPi.GPIO as GPIO
import time

import TLC1543

from VARIABLES import *
```

```
from random import randint

x=[0,0,0,0,0]
vel=25
blanco=14
incremento=10
tiempo = 0.2



##hacia DELANTE
GPIO.output(IN1,GPIO.LOW)
GPIO.output(IN2,GPIO.HIGH)
GPIO.output(IN3,GPIO.HIGH)
GPIO.output(IN4,GPIO.LOW)

while True:
    lineanegra=0
    #Leemos los siguelíneas
    for i in range(5):
        x[i]=TLC1543.SENSORLINEA(i)
        if (x[i]90):
            velA=90
        if (velB90):
            velB=90
        ##activamos motores
        print ('con linea negra SENSORES=',x[0],'-',x[1],'-',x[2],'-',x[3],'-',x[4],'-
VELA=',velA,'VELB=',velB);
        PWMA.start(velA)
        PWMB.start(velB)
        #time.sleep(tiempo)
    else:           ##no tiene linea negra pues marcha atrás y que lo busque
        GPIO.output(IN1,GPIO.HIGH)
        GPIO.output(IN2,GPIO.LOW)
        GPIO.output(IN3,GPIO.LOW)
        GPIO.output(IN4,GPIO.HIGH)
        while (lineanegra==0):
```

Copyright 2026

```
velA=randint(vel-incremento,vel+incremento)
velB=vel-(velA-vel)  #el opuesto
print ('SIN linea negra VELA=',velA,'VELB=',velB);
PWMA.start(velA)
PWMB.start(velB)
time.sleep(tiempo)
for i in range(5):
    x[i]=TLC1543.SENSORLINEA(i)
    if (x[i]
```

¿Por qué en este ejercicio ALPHABOT va al revés?

Por que los sensores siguelineas por la parte de atrás del sentido de la marcha **PRODUCE UNA REALIMENTACIÓN POSITIVA** es decir, cuando detecta que hay que girar, gira, pero la cola se mueve demasiado deprisa que produce que pierda la línea. Controlarlo **es posible pero es difícil** la demo de Alphabot lleva el software para hacerlo.

□□□ No seas gallina !! □□□ Pruébalo.

- Cambia el código anterior las marchas es decir los GPIO.output de los motores, pon los HIGH por LOW y viceversa
- Y también el control de giro, es decir en vez de +incremento pon -incremento y viceversa
- ¿Funciona? ¡¡se vuelve loco !!!

Chocheando un poco.. esto me recuerda a una vieja historia..

Si los pioneros de la aviación americanos lo tuvieron difícil, más lo tuvieron los españores. Se lo podríamos preguntar al **pastor** del pueblo Coruña del Conde: **Diego Marín Aquilera** que en 1793 inventó un artilugio que volaba de forma controlada... la pena es que la Inquisición, el cura del pueblo junto con los lugareños no tenían ni idea que este español hubiera hecho historia, y que la aviación hubiera adelantado más de 100 años. Pensaban que eso era obra del demonio □□ por lo tanto quemaron todos sus inventos □□□□ □□ □□♂.



De Eulogia Merle - Fundación Española para la Ciencia y la Tecnología, CC BY-SA 4.0

Aaayyyy □□ si en España hicieramos caso a los genios que tenemos ...

Esa mala suerte no lo tuvieron **los hermanos Wright** que en 1903 volaron su primer artilugio:



Gregg Bryant saved to Aircraft-Vintage Pinterest

Pero... pusieron **el timón delante**, no como **Diego Marín Aquilera** que observaba bien las aves. Esto provocaba una **realimentación positiva** (al levantar el timón, levantaba el morro, y esto provocaba que se levantase aún más, y viceversa a la hora de bajarlo). Los hermanos Wright patentaron su invento y gastaron todo su dinero en abogados para defender que nadie copiase su control, pero la verdad es que ... que nadie lo hizo.

La industria de la aviación detectó el fallo y los elementos de control van por detrás del ala principal, esto crea una realimentación negativa, por lo tanto mayor estabilidad en el vuelo y ... la ruina de los hermanos Wright.

□□ ¿En el diseño de este Alphabot habrá participado algún descendiente de los hermanos Wright? □□

> Mentirijilla: En algunas ocasiones se usa el timón delante: En los cazas para conseguir giros muy rápidos aprovechando esa realimentación positiva como este Saab 39 Gripen:



Editado de De Ernst Vikne - originally posted to Flickr as JAS Gripen, CC BY-SA 2.0