**Oefeningen ledstrips :**

oefeningen WS2812

https://www.tinytronics.nl/shop/nl/verlichting/led-digitaal/ws2812b-digitale-5050-rgb-led-ring-12-leds

oef 8-1 ) 1 digitale led en stuur ze achter elkaar met pauze 1sec rood, groen , blauw met intensiteit max.

from rpi\_ws281x import \*

import time

# LED strip configuration:

LED\_COUNT = 12 # Number of LED pixels.

LED\_PIN = 21 # GPIO pin connected to the pixels (18 uses PWM!).

#LED\_PIN = 10 # GPIO pin connected to the pixels (10 uses SPI /dev/spidev0.0).

LED\_FREQ\_HZ = 800000 # LED signal frequency in hertz (usually 800khz)

LED\_DMA = 5 # DMA channel to use for generating signal (try 10)

LED\_BRIGHTNESS = 255 # Set to 0 for darkest and 255 for brightest

LED\_INVERT = False # True to invert the signal (when using NPN transistor level shift)

LED\_CHANNEL = 0 # set to '1' for GPIOs 13, 19, 41, 45 or 53

strip = Adafruit\_NeoPixel(LED\_COUNT, LED\_PIN, LED\_FREQ\_HZ, LED\_DMA, LED\_INVERT, LED\_BRIGHTNESS) # last par LED\_CHANNEL not used

strip.begin()

while True:

strip.setPixelColorRGB(0, 255, 0, 0)

strip.show()

time.sleep(1)

strip.setPixelColorRGB(0, 0, 255, 0)

strip.show()

time.sleep(1)

strip.setPixelColorRGB(0, 0, 0, 255)

strip.show()

time.sleep(1)

oef 8-2 ) 1 digitale led en stuur ze achter elkaar met pauze 1sec rood, groen , blauw met intensiteit min.

from rpi\_ws281x import \*

import time

# LED strip configuration:

LED\_COUNT = 12 # Number of LED pixels.

LED\_PIN = 13 # GPIO pin connected to the pixels (18 uses PWM!).

#LED\_PIN = 10 # GPIO pin connected to the pixels (10 uses SPI /dev/spidev0.0).

LED\_FREQ\_HZ = 800000 # LED signal frequency in hertz (usually 800khz)

LED\_DMA = 5 # DMA channel to use for generating signal (try 10)

LED\_BRIGHTNESS = 255 # Set to 0 for darkest and 255 for brightest

LED\_INVERT = False # True to invert the signal (when using NPN transistor level shift)

LED\_CHANNEL = 1 # set to '1' for GPIOs 13, 19, 41, 45 or 53

strip = Adafruit\_NeoPixel(LED\_COUNT, LED\_PIN, LED\_FREQ\_HZ, LED\_DMA, LED\_INVERT, LED\_BRIGHTNESS,LED\_CHANNEL) # last par LED\_CHANNEL not used

strip.begin()

while True:

strip.setPixelColorRGB(0, 1, 0, 0)

strip.show()

time.sleep(1)

strip.setPixelColorRGB(0, 0, 1, 0)

strip.show()

time.sleep(1)

strip.setPixelColorRGB(0, 0, 0, 1)

strip.show()

time.sleep(1)

oef 8-3 ) 1 digitale led en stuur ze op 2.55 seconden van min intensiteit rood naar max intensiteit rood.

from rpi\_ws281x import \*

import time

# LED strip configuration:

LED\_COUNT = 12 # Number of LED pixels.

LED\_PIN = 13 # GPIO pin connected to the pixels (18 uses PWM!).

#LED\_PIN = 10 # GPIO pin connected to the pixels (10 uses SPI /dev/spidev0.0).

LED\_FREQ\_HZ = 800000 # LED signal frequency in hertz (usually 800khz)

LED\_DMA = 5 # DMA channel to use for generating signal (try 10)

LED\_BRIGHTNESS = 255 # Set to 0 for darkest and 255 for brightest

LED\_INVERT = False # True to invert the signal (when using NPN transistor level shift)

LED\_CHANNEL = 1 # set to '1' for GPIOs 13, 19, 41, 45 or 53

strip = Adafruit\_NeoPixel(LED\_COUNT, LED\_PIN, LED\_FREQ\_HZ, LED\_DMA, LED\_INVERT, LED\_BRIGHTNESS,LED\_CHANNEL) # last par LED\_CHANNEL not used

strip.begin()

while True:

for i in range(256): # range 0...255

strip.setPixelColorRGB(0, i, 0, 0)

strip.show()

time.sleep(0.01)

oef 8-4 ) 1 digitale led en stuur ze op 2.55 seconden van min intensiteit rood naar max intensiteit rood, wacht 2 seconden en dan op 2.55 seconden van max intensiteit rood terug naar min intensiteit rood.

from rpi\_ws281x import \*

import time

# LED strip configuration:

LED\_COUNT = 12 # Number of LED pixels.

LED\_PIN = 13 # GPIO pin connected to the pixels (18 uses PWM!).

#LED\_PIN = 10 # GPIO pin connected to the pixels (10 uses SPI /dev/spidev0.0).

LED\_FREQ\_HZ = 800000 # LED signal frequency in hertz (usually 800khz)

LED\_DMA = 5 # DMA channel to use for generating signal (try 10)

LED\_BRIGHTNESS = 255 # Set to 0 for darkest and 255 for brightest

LED\_INVERT = False # True to invert the signal (when using NPN transistor level shift)

LED\_CHANNEL = 1 # set to '1' for GPIOs 13, 19, 41, 45 or 53

strip = Adafruit\_NeoPixel(LED\_COUNT, LED\_PIN, LED\_FREQ\_HZ, LED\_DMA, LED\_INVERT, LED\_BRIGHTNESS,LED\_CHANNEL) # last par LED\_CHANNEL not used

strip.begin()

while True:

for i in range(1,256): # 1...255

strip.setPixelColorRGB(0, i, 0, 0)

strip.show()

time.sleep(0.01)

time.sleep(2)

for i in range(255,0,-1): # 255...1

strip.setPixelColorRGB(0, i, 0, 0)

strip.show()

time.sleep(0.01)

time.sleep(2)

oef 8-5 ) 1 digitale led en stuur ze op 2.55 seconden van min intensiteit wit naar max intensiteit wit, wacht 2 seconden, en daarna op 2.55 seconden van max intensiteit wit terug naar min intensiteit wit.

from rpi\_ws281x import \*

import time

# LED strip configuration:

LED\_COUNT = 12 # Number of LED pixels.

LED\_PIN = 13 # GPIO pin connected to the pixels (18 uses PWM!).

#LED\_PIN = 10 # GPIO pin connected to the pixels (10 uses SPI /dev/spidev0.0).

LED\_FREQ\_HZ = 800000 # LED signal frequency in hertz (usually 800khz)

LED\_DMA = 5 # DMA channel to use for generating signal (try 10)

LED\_BRIGHTNESS = 255 # Set to 0 for darkest and 255 for brightest

LED\_INVERT = False # True to invert the signal (when using NPN transistor level shift)

LED\_CHANNEL = 1 # set to '1' for GPIOs 13, 19, 41, 45 or 53

strip = Adafruit\_NeoPixel(LED\_COUNT, LED\_PIN, LED\_FREQ\_HZ, LED\_DMA, LED\_INVERT, LED\_BRIGHTNESS,LED\_CHANNEL) # last par LED\_CHANNEL not used

strip.begin()

while True:

for i in range(1,256): # 1...255

strip.setPixelColorRGB(0, i, i, i)

strip.show()

time.sleep(0.01)

time.sleep(2)

for i in range(255,0,-1): # 255...1

strip.setPixelColorRGB(0, i, i, i)

strip.show()

time.sleep(0.01)

time.sleep(2)

oef 8-6 )cirkel met 12 digitale RGB leds, laat 1 rode led met max intensisteit ronddraaien in de richting van de klok met 1 stap per 500 ms.

from rpi\_ws281x import \*

import time

# LED strip configuration:

LED\_COUNT = 12 # Number of LED pixels.

LED\_PIN = 13 # GPIO pin connected to the pixels (18 uses PWM!).

#LED\_PIN = 10 # GPIO pin connected to the pixels (10 uses SPI /dev/spidev0.0).

LED\_FREQ\_HZ = 800000 # LED signal frequency in hertz (usually 800khz)

LED\_DMA = 5 # DMA channel to use for generating signal (try 10)

LED\_BRIGHTNESS = 255 # Set to 0 for darkest and 255 for brightest

LED\_INVERT = False # True to invert the signal (when using NPN transistor level shift)

LED\_CHANNEL = 1 # set to '1' for GPIOs 13, 19, 41, 45 or 53

strip = Adafruit\_NeoPixel(LED\_COUNT, LED\_PIN, LED\_FREQ\_HZ, LED\_DMA, LED\_INVERT, LED\_BRIGHTNESS,LED\_CHANNEL) # last par LED\_CHANNEL not used

strip.begin()

while True:

for i in range(LED\_COUNT):

strip.setPixelColorRGB(i,255,0,0)

strip.show()

time.sleep(0.5)

strip.setPixelColorRGB(i,0,0,0)

oef 8-7) cirkel met 12 digitale RGB leds, laat 1 rode led met max intensisteit ronddraaien in de richting tegen de klok met 1 stap per 500 ms.

from rpi\_ws281x import \*

import time

# LED strip configuration:

LED\_COUNT = 12 # Number of LED pixels.

LED\_PIN = 13 # GPIO pin connected to the pixels (18 uses PWM!).

#LED\_PIN = 10 # GPIO pin connected to the pixels (10 uses SPI /dev/spidev0.0).

LED\_FREQ\_HZ = 800000 # LED signal frequency in hertz (usually 800khz)

LED\_DMA = 5 # DMA channel to use for generating signal (try 10)

LED\_BRIGHTNESS = 255 # Set to 0 for darkest and 255 for brightest

LED\_INVERT = False # True to invert the signal (when using NPN transistor level shift)

LED\_CHANNEL = 1 # set to '1' for GPIOs 13, 19, 41, 45 or 53

strip = Adafruit\_NeoPixel(LED\_COUNT, LED\_PIN, LED\_FREQ\_HZ, LED\_DMA, LED\_INVERT, LED\_BRIGHTNESS,LED\_CHANNEL) # last par LED\_CHANNEL not used

strip.begin()

while True:

for i in range(LED\_COUNT-1,-1,-1): # LED\_COUNT-1 ... 0

strip.setPixelColorRGB(i,255,0,0)

strip.show()

time.sleep(0.5)

strip.setPixelColorRGB(i,0,0,0)

oef 8-8) cirkel met 12 digitale RGB leds, laat 1 rode led met max intensisteit ronddraaien. De richting wordt bepaald door een schakelaar aan input 7. ( input hoog CW , laag ACW)

from rpi\_ws281x import \*

import time

import RPi.GPIO as GPIO

GPIO.setwarnings(False)

GPIO.setmode(GPIO.BCM)

# LED strip configuration:

LED\_COUNT = 12 # Number of LED pixels.

LED\_PIN = 13 # GPIO pin connected to the pixels (18 uses PWM!).

#LED\_PIN = 10 # GPIO pin connected to the pixels (10 uses SPI /dev/spidev0.0).

LED\_FREQ\_HZ = 800000 # LED signal frequency in hertz (usually 800khz)

LED\_DMA = 5 # DMA channel to use for generating signal (try 10)

LED\_BRIGHTNESS = 255 # Set to 0 for darkest and 255 for brightest

LED\_INVERT = False # True to invert the signal (when using NPN transistor level shift)

LED\_CHANNEL = 1 # set to '1' for GPIOs 13, 19, 41, 45 or 53

strip = Adafruit\_NeoPixel(LED\_COUNT, LED\_PIN, LED\_FREQ\_HZ, LED\_DMA, LED\_INVERT, LED\_BRIGHTNESS,LED\_CHANNEL) # last par LED\_CHANNEL not used

strip.begin()

GPIO.setup(26,GPIO.IN , pull\_up\_down=GPIO.PUD\_UP)

while True:

x=GPIO.input(26)

print(x)

if x==1:

for i in range(LED\_COUNT):

strip.setPixelColorRGB(i, 255, 0, 0)

strip.show()

time.sleep(0.5)

strip.setPixelColorRGB(i, 0, 0, 0)

else:

for i in range(LED\_COUNT-1,-1,-1):

strip.setPixelColorRGB(i, 255, 0, 0)

strip.show()

time.sleep(0.5)

strip.setPixelColorRGB(i, 0, 0, 0)

oef 8-9) cirkel met 12 digitale RGB leds, laat 1 rode led ronddraaien. De richting wordt bepaald door een schakelaar. De intensiteit wordt ingesteld met behulp van een potentiometer.

from rpi\_ws281x import \*

import time

import RPi.GPIO as GPIO

GPIO.setwarnings(False)

GPIO.setmode(GPIO.BCM)

GPIO.setup(26,GPIO.IN , pull\_up\_down=GPIO.PUD\_UP)

from gpiozero import MCP3008

mcp3008\_channel\_0 = MCP3008(channel=0, device=0)

# LED strip configuration:

LED\_COUNT = 12 # Number of LED pixels.

LED\_PIN = 13 # GPIO pin connected to the pixels (18 uses PWM!).

#LED\_PIN = 10 # GPIO pin connected to the pixels (10 uses SPI /dev/spidev0.0).

LED\_FREQ\_HZ = 800000 # LED signal frequency in hertz (usually 800khz)

LED\_DMA = 5 # DMA channel to use for generating signal (try 10)

LED\_BRIGHTNESS = 255 # Set to 0 for darkest and 255 for brightest

LED\_INVERT = False # True to invert the signal (when using NPN transistor level shift)

LED\_CHANNEL = 1 # set to '1' for GPIOs 13, 19, 41, 45 or 53

strip = Adafruit\_NeoPixel(LED\_COUNT, LED\_PIN, LED\_FREQ\_HZ, LED\_DMA, LED\_INVERT, LED\_BRIGHTNESS,LED\_CHANNEL) # last par LED\_CHANNEL not used

strip.begin()

while True:

intensiteit = int(mcp3008\_channel\_0.value \*255)

richting = GPIO.input(26)

if richting == 1:

for i in range(LED\_COUNT):

strip.setPixelColorRGB(i, intensiteit , 0, 0)

strip.show()

time.sleep(0.5)

strip.setPixelColorRGB(i, 0 , 0, 0)

else:

for i in range(LED\_COUNT-1,-1,-1):

strip.setPixelColorRGB(i, intensiteit , 0, 0)

strip.show()

time.sleep(0.5)

strip.setPixelColorRGB(i, 0 , 0, 0)

oef 8-9-b) cirkel met 12 digitale RGB leds, laat 1 rode led ronddraaien. De richting wordt bepaald door een schakelaar. De intensiteit wordt ingesteld met behulp van een potentiometer. Veranderingen reageren onmiddellijk!

oef 8-10) ledstrip met 10 digitale RGB leds, gebruik ze als een volume-level of spaningsmeter. Despanning 0-3V3 zo goed mogelijk weergeven met de 10 leds, alle leds zelfde kleur bv blauw.  
  
#oef 8-10) ledstrip met 10 digitale RGB leds, gebruik ze als een volume-level of spaningsmeter.

# Despanning 0-3V3 zo goed mogelijk weergeven met de 10 leds, alle leds zelfde kleur bv blauw.

from rpi\_ws281x import \*

import time

from gpiozero import MCP3008

# LED strip configuration:

LED\_COUNT = 10 # Number of LED pixels.

LED\_PIN = 13 # GPIO pin connected to the pixels (18 uses PWM!).

#LED\_PIN = 10 # GPIO pin connected to the pixels (10 uses SPI /dev/spidev0.0).

LED\_FREQ\_HZ = 800000 # LED signal frequency in hertz (usually 800khz)

LED\_DMA = 5 # DMA channel to use for generating signal (try 10)

LED\_BRIGHTNESS = 255 # Set to 0 for darkest and 255 for brightest

LED\_INVERT = False # True to invert the signal (when using NPN transistor level shift)

LED\_CHANNEL = 1 # set to '1' for GPIOs 13, 19, 41, 45 or 53

strip = Adafruit\_NeoPixel(LED\_COUNT, LED\_PIN, LED\_FREQ\_HZ, LED\_DMA, LED\_INVERT, LED\_BRIGHTNESS, LED\_CHANNEL) # last par LED\_CHANNEL not used

strip.begin()

mcp3008\_channel\_0 = MCP3008(channel=0, device=0)

while True:

waarde= round(mcp3008\_channel\_0.value \* 10 )# 0....10

print(waarde)

for \_ in range(10): # 0.....9 = ledpixel 0..9

if waarde > \_:

strip.setPixelColorRGB(\_, 0, 0, 255)

else:

strip.setPixelColorRGB(\_, 0, 0, 0)

strip.show()

time.sleep(0.1)

oef 8-11) ledstrip met 10 digitale RGB leds, gebruik ze als een volume-level of spaningsmeter. Spanning 0-5V zo goed mogelijk weergeven door de kleur van de 10 leds te veranderen van groen naar oranje naar rood,dit zo geleidelijk mogelijk wanneer de spanning verandert van 0-3V3.

#oef 8-11) ledstrip met 10 digitale RGB leds, gebruik ze als een volume-level of spaningsmeter.

#Spanning 0-3V3 zo goed mogelijk weergeven door de kleur van de 10 leds te veranderen van groen naar oranje naar rood,dit zo

# geleidelijk mogelijk wanneer de spanning verandert van 0-3V3.

#def my\_map(x, in\_min, in\_max, out\_min, out\_max):

#return int((x - in\_min) \* (out\_max - out\_min) / (in\_max - in\_min) + out\_min)

from rpi\_ws281x import \*

import time

from gpiozero import MCP3008

# LED strip configuration:

LED\_COUNT = 10 # Number of LED pixels.

LED\_PIN = 13 # GPIO pin connected to the pixels (18 uses PWM!).

#LED\_PIN = 10 # GPIO pin connected to the pixels (10 uses SPI /dev/spidev0.0).

LED\_FREQ\_HZ = 800000 # LED signal frequency in hertz (usually 800khz)

LED\_DMA = 5 # DMA channel to use for generating signal (try 10)

LED\_BRIGHTNESS = 255 # Set to 0 for darkest and 255 for brightest

LED\_INVERT = False # True to invert the signal (when using NPN transistor level shift)

LED\_CHANNEL = 1 # set to '1' for GPIOs 13, 19, 41, 45 or 53

strip = Adafruit\_NeoPixel(LED\_COUNT, LED\_PIN, LED\_FREQ\_HZ, LED\_DMA, LED\_INVERT, LED\_BRIGHTNESS, LED\_CHANNEL) # last par LED\_CHANNEL not used

strip.begin()

def my\_map(x, in\_min, in\_max, out\_min, out\_max):

return int((x - in\_min) \* (out\_max - out\_min) / (in\_max - in\_min) + out\_min)

mcp3008\_channel\_0 = MCP3008(channel=0, device=0)

volume\_before=0

while True:

volume = round(mcp3008\_channel\_0.value \* 1023 )# 0....10

print(volume)

if volume != volume\_before:

volume\_before=volume

# 0.....9 = ledpixel 0..9

if volume < 512:

rood\_gehalte= my\_map(volume , 0,1023 , 0,510); # rood moet 255 bij pot midden

groen\_gehalte = my\_map(volume , 0,1023 , 255,128); #oranje = ( 255/128/0)

else:

rood\_gehalte= 255;

groen\_gehalte = my\_map(volume , 0,1023 , 255,0); # pot midden moet G=128 gevn

for i in range(LED\_COUNT):

strip.setPixelColorRGB(i,rood\_gehalte, groen\_gehalte, 0)

print(rood\_gehalte,"/",groen\_gehalte)

strip.show();

time.sleep(0.1)

oef 8-12) ledstrip met 10 digitale RGB leds, gebruik ze als een volume-level of spaningsmeter. Spanning 0-3V3 zo goed mogelijk weergeven door 1 tot 10 leds te sturen en de brandende leds hun kleur samen te veranderen van groen (0V) naar oranje (1.66V) naar rood (3V3) , dit zo geleidelijk mogelijk wanneer de spanning verandert tussen 0-3V3.

from rpi\_ws281x import \*

import time

from gpiozero import MCP3008

# LED strip configuration:

LED\_COUNT = 10 # Number of LED pixels.

LED\_PIN = 13 # GPIO pin connected to the pixels (18 uses PWM!).

#LED\_PIN = 10 # GPIO pin connected to the pixels (10 uses SPI /dev/spidev0.0).

LED\_FREQ\_HZ = 800000 # LED signal frequency in hertz (usually 800khz)

LED\_DMA = 5 # DMA channel to use for generating signal (try 10)

LED\_BRIGHTNESS = 255 # Set to 0 for darkest and 255 for brightest

LED\_INVERT = False # True to invert the signal (when using NPN transistor level shift)

LED\_CHANNEL = 1 # set to '1' for GPIOs 13, 19, 41, 45 or 53

strip = Adafruit\_NeoPixel(LED\_COUNT, LED\_PIN, LED\_FREQ\_HZ, LED\_DMA, LED\_INVERT, LED\_BRIGHTNESS, LED\_CHANNEL) # last par LED\_CHANNEL not used

strip.begin()

def my\_map(x, in\_min, in\_max, out\_min, out\_max):

return int((x - in\_min) \* (out\_max - out\_min) / (in\_max - in\_min) + out\_min)

mcp3008\_channel\_0 = MCP3008(channel=0, device=0)

volume\_before=0

while True:

volume = round(mcp3008\_channel\_0.value \* 1023 )# 0....10

print(volume)

if volume != volume\_before:

volume\_before=volume

# 0.....9 = ledpixel 0..9

if volume < 512:

rood\_gehalte= my\_map(volume , 0,1023 , 0,510); # rood moet 255 bij pot midden

groen\_gehalte = my\_map(volume , 0,1023 , 255,128); #oranje = ( 255/128/0)

else:

rood\_gehalte= 255;

groen\_gehalte = my\_map(volume , 0,1023 , 255,0); # pot midden moet G=128 gevn

aantal\_leds = my\_map(volume , 0,1023 , 0 , 10)

print(aantal\_leds)

for i in range(LED\_COUNT):

if aantal\_leds > i:

strip.setPixelColorRGB(i,rood\_gehalte, groen\_gehalte, 0)

else:

strip.setPixelColorRGB(i,0, 0, 0)

print(rood\_gehalte,"/",groen\_gehalte)

strip.show();

time.sleep(0.1)

oef 8-13) Simuleer de beroemde Knight rider KITT led-sequentie.

from rpi\_ws281x import \*

import time

from gpiozero import MCP3008

# LED strip configuration:

LED\_COUNT = 10 # Number of LED pixels.

LED\_PIN = 13 # GPIO pin connected to the pixels (18 uses PWM!).

#LED\_PIN = 10 # GPIO pin connected to the pixels (10 uses SPI /dev/spidev0.0).

LED\_FREQ\_HZ = 800000 # LED signal frequency in hertz (usually 800khz)

LED\_DMA = 5 # DMA channel to use for generating signal (try 10)

LED\_BRIGHTNESS = 255 # Set to 0 for darkest and 255 for brightest

LED\_INVERT = False # True to invert the signal (when using NPN transistor level shift)

LED\_CHANNEL = 1 # set to '1' for GPIOs 13, 19, 41, 45 or 53

strip = Adafruit\_NeoPixel(LED\_COUNT, LED\_PIN, LED\_FREQ\_HZ, LED\_DMA, LED\_INVERT, LED\_BRIGHTNESS, LED\_CHANNEL) # last par LED\_CHANNEL not used

strip.begin()

while True:

for t in range(LED\_COUNT):

strip.setPixelColorRGB(t, 255, 0, 0)

strip.show()

time.sleep(0.1)

strip.setPixelColorRGB(t, 0, 0, 0)

time.sleep(0.1)

for t in range(LED\_COUNT-1,-1,-1):

strip.setPixelColorRGB(t, 255, 0, 0)

strip.show()

time.sleep(0.1)

strip.setPixelColorRGB(t, 0, 0, 0)

time.sleep(0.1)

oef 8-14) Met 3 drukknoppen stuur je 3 voorgeprogrammeerde kleuren naar de ledstrip. Een vierde knop schakelt de leds aan/uit.

from rpi\_ws281x import \*

import RPi.GPIO as GPIO

import time

from gpiozero import MCP3008

# LED strip configuration:

LED\_COUNT = 10 # Number of LED pixels.

LED\_PIN = 13 # GPIO pin connected to the pixels (18 uses PWM!).

#LED\_PIN = 10 # GPIO pin connected to the pixels (10 uses SPI /dev/spidev0.0).

LED\_FREQ\_HZ = 800000 # LED signal frequency in hertz (usually 800khz)

LED\_DMA = 5 # DMA channel to use for generating signal (try 10)

LED\_BRIGHTNESS = 255 # Set to 0 for darkest and 255 for brightest

LED\_INVERT = False # True to invert the signal (when using NPN transistor level shift)

LED\_CHANNEL = 1 # set to '1' for GPIOs 13, 19, 41, 45 or 53

strip = Adafruit\_NeoPixel(LED\_COUNT, LED\_PIN, LED\_FREQ\_HZ, LED\_DMA, LED\_INVERT, LED\_BRIGHTNESS, LED\_CHANNEL) # last par LED\_CHANNEL not used

strip.begin()

but\_kleur1= 5

but\_kleur2= 6

but\_kleur3= 26

but\_aan\_uit= 18

GPIO.setwarnings(False)

GPIO.setmode(GPIO.BCM)

GPIO.setup(but\_kleur1,GPIO.IN, pull\_up\_down=GPIO.PUD\_UP)

GPIO.setup(but\_kleur2,GPIO.IN, pull\_up\_down=GPIO.PUD\_UP)

GPIO.setup(but\_kleur3,GPIO.IN, pull\_up\_down=GPIO.PUD\_UP)

GPIO.setup(but\_aan\_uit,GPIO.IN, pull\_up\_down=GPIO.PUD\_UP)

kleur=0

kleur1=0xFF0000

kleur2=0x00FF00

kleur3=0x0000FF

aan\_uit=0

def update\_leds():

print("update\_leds",(kleur & 0xFF0000)>> 16,(kleur & 0x00FF00)>> 8,(kleur & 0x0000FF))

for i in range(LED\_COUNT):

strip.setPixelColorRGB(i,(kleur & 0xFF0000)>> 16, (kleur & 0x00FF00)>> 8, (kleur & 0x0000FF)) # bitwise operators & and >>

strip.show()

def fct\_kleur1(p):

print("fct\_kleur1")

global kleur

kleur=kleur1

update\_leds()

def fct\_kleur2(p):

print("fct\_kleur2")

global kleur

kleur=kleur2

update\_leds()

def fct\_kleur3(p):

print("fct\_kleur3")

global kleur

kleur=kleur3

update\_leds()

def fct\_aan\_uit(p):

print("fct\_aan\_uit")

global aan\_uit

aan\_uit= not aan\_uit

if aan\_uit:

for i in range(LED\_COUNT):

strip.setPixelColorRGB(i,(kleur & 0xFF0000)>> 16, (kleur & 0x00FF00)>> 8, (kleur & 0x0000FF)) # bitwise operators

else:

for i in range(LED\_COUNT):

strip.setPixelColorRGB(i,0, 0, 0)

strip.show()

GPIO.add\_event\_detect(but\_kleur1, GPIO.FALLING, callback = fct\_kleur1) # lambda gebruiken?

GPIO.add\_event\_detect(but\_kleur2, GPIO.FALLING, callback = fct\_kleur2)

GPIO.add\_event\_detect(but\_kleur3, GPIO.FALLING, callback = fct\_kleur3)

GPIO.add\_event\_detect(but\_aan\_uit, GPIO.FALLING, callback = fct\_aan\_uit)

while True:

pass # callbacks aan buttons toegewezen

oef 8-15) Met 3 drukknoppen stuur je 3 voorgeprogrammeerde kleuren naar de ledstrip. Een vierde knop schakelt de leds aan/uit.

Probeer de kleuren flexibel toe te wijzen en op te slaan. ( mogelijk pas voor module 2 via bv bluetooth App)

Opmerking :

rpi\_ws281x import kan je niet starten vanuit Thonny IDE als user “pi”.

Werken als user “root” lukt wel maar wordt afgeraden. Je moet dan eerst een pwd aanmaken voor de “root” user.

sudo su

sudo passwd root

Daarom starten we de ledpixel programma’s vanop de terminal met : **sudo python3 naamprogramma.py**