**Oefeningenreeks 7 23 Nov 2020**

7-1) Schrijf een programma dat weergeeft (print en/of GUI) hoeveel milliseconden je een drukknop, verbonden met je RP, had ingedrukt. Je kan deze waarde updaten bij het loslaten.

import RPi.GPIO as GPIO

import time

GPIO.setwarnings(False)

GPIO.setmode(GPIO.BCM)

drukknop1 = 18

vorige\_toestand\_drukknop1=1

start\_tijd=0

stop\_tijd=0

GPIO.setup(drukknop1,GPIO.IN,pull\_up\_down=GPIO.PUD\_UP)

print("Druk om de tijdsmeting te starten!")

while True:

toestand\_drukknop1= GPIO.input(drukknop1)

if vorige\_toestand\_drukknop1 != toestand\_drukknop1:

vorige\_toestand\_drukknop1 = toestand\_drukknop1

if toestand\_drukknop1==0:

start\_tijd=time.time()

print("Tijdsmeting gestart!")

if toestand\_drukknop1==1:

stop\_tijd=time.time()

print("Knop was",stop\_tijd-start\_tijd,"seconden ingedrukt!")

time.sleep(0.2)

7-1b) Schrijf een programma dat weergeeft (print en/of GUI) hoeveel milliseconden je een drukknop op je GUI had ingedrukt. Je kan deze waarde updaten bij het loslaten.

import tkinter as tk

import time

start\_tijd=0

stop\_tijd=0

def button\_press(p):

global start\_tijd

start\_tijd=time.time()

print("Start chrono!")

def button\_release(p):

global stop\_tijd

stop\_tijd=time.time()

print("Stopped chrono, total time =", stop\_tijd-start\_tijd)

#defining our window

window = tk.Tk()

window.geometry("800x400") # do x , not \*

window.title("Button")

#defining what is in the window

schakelaar = tk.Button(window, text="COUNT")

schakelaar.place(x = "100", y = "100")

schakelaar.bind("<Button>", button\_press) #event indrukken toevoegen aan widget

schakelaar.bind("<ButtonRelease>", button\_release) #event loslaten toevoegen aan widget

window.mainloop()

7-2) Schrijf een programma dat een led vanop 4 plaatsen kan aansturen, zowel met 4 GPIO drukknoppen als met 4 GUI knoppen.

import RPi.GPIO as GPIO

import time

GPIO.setmode(GPIO.BCM)

GPIO.setwarnings(False)

GPIO.cleanup()

#drukknop1 = 18

#drukknop2= 12 #drukknop3= 5 #drukknop4= 6

list\_drukknoppen = [ 18,12,5,6]

GPIO.setup(list\_drukknoppen, GPIO.IN,pull\_up\_down=GPIO.PUD\_UP) # [18,12,5,6] is een Python list

led\_rood=24

status\_led\_rood=0

GPIO.setup(led\_rood,GPIO.OUT)

vorige\_toestand\_drukknoppen = [ 1,1,1,1]

toestand\_drukknoppen = [ 1,1,1,1]

def scan\_4\_drukknoppen():

global vorige\_toestand\_drukknoppen ,toestand\_drukknoppen, list\_drukknoppen

ingedrukt = 0

for x in range(4):

toestand\_drukknop\_nu = GPIO.input(list\_drukknoppen[x])

if toestand\_drukknop\_nu != vorige\_toestand\_drukknoppen[x]:

vorige\_toestand\_drukknoppen[x] = toestand\_drukknop\_nu

if toestand\_drukknop\_nu == 0:

print("knop aan pin", list\_drukknoppen[x] , "ingedrukt")

ingedrukt = 1

else:

print("knop aan pin", list\_drukknoppen[x] , "losgelaten")

time.sleep(0.2)

return ingedrukt

print("Druk op 1 vd 4 drukknoppen om de led te toggelen!")

while True:

ingedrukt= scan\_4\_drukknoppen()

if ingedrukt == 1:

print("ingedrukt=1")

status\_led\_rood = not status\_led\_rood

print("status\_led\_rood=", status\_led\_rood)

GPIO.output(led\_rood , status\_led\_rood)

7-3) Schrijf een programma dat een teller verhoogt en ze’n waarde uitprint telkens een schakelaar wordt ingedrukt. Kan je die waarde ook tonen in een GUI? Je kan geen GUI aanpassen vanuit de callback van je schakelaar, hoe los je dit op?

import time

import RPi.GPIO as GPIO

from tkinter import \*

window = Tk()

window.geometry("800x400")

window.title("Oef 7.3")

knop1 = 18

counter = 0

counter\_before=0

GPIO.setwarnings(False)

GPIO.setmode(GPIO.BCM)

GPIO.setup(knop1,GPIO.IN,pull\_up\_down = GPIO.PUD\_UP)

def button\_pressed(channel):

global counter

counter = counter + 1

# my\_label["text"] = counter tkinter updates mogen alleen in main tthread, niet in thread van GPIO callbacks

print(counter)

my\_label = Label(window, text = "0",relief = GROOVE)

my\_label.place(x=10,y=100)

GPIO.add\_event\_detect(knop1, GPIO.FALLING, callback=button\_pressed, bouncetime = 200)

while True:

if counter != counter\_before:

counter\_before = counter

my\_label["text"] = counter

window.update()

7-4) Schrijf een programma dat met een start en een stop drukknop de tijd registreert zoals een chronometer

import time

from tkinter import \*

import RPi.GPIO as GPIO

window = Tk()

window.geometry("500x250")

window.title("Oef 7.4")

start\_knop = 18

stop\_knop = 12

GPIO.setwarnings(False)

GPIO.setmode(GPIO.BCM)

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

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

my\_label = Label(window, text = "0", font=('Times', '128'),relief = GROOVE) #Label(window, text=0 , font=('Times', '128'), relief = SUNKEN)

my\_label.place(x=10,y=10)

start\_tijd = 0

chrono\_running = False

def start\_pressed(channel):

global start\_tijd,chrono\_running

start\_tijd = time.time()

chrono\_running = True

def stop\_pressed(channel):

global chrono\_running

chrono\_running = False

GPIO.add\_event\_detect(start\_knop, GPIO.FALLING, callback=start\_pressed, bouncetime = 200)

GPIO.add\_event\_detect(stop\_knop, GPIO.FALLING, callback=stop\_pressed, bouncetime = 200)

while True:

if chrono\_running:

my\_label["text"] = round(time.time() - start\_tijd,3)

window.update()

7-5) Schrijf een programma dat de rode, oranje, groene led toggelt bij respektievelijk indrukken van slechts 1 GPIO drukknop 1,2,3 keer in een bepaalde tijd die je zelf kiest. ( 1.5 seconde in onderstaand vb)

import RPi.GPIO as GPIO

import time

red=23 # refereer naar GPIO pin met variablele

orange=24 # refereer naar GPIO pin met variablele

green= 25 # refereer naar GPIO pin met variablele

drukknop = 18

GPIO.setwarnings(False)

GPIO.setmode(GPIO.BCM)

GPIO.setup(red,GPIO.OUT)

GPIO.setup(orange,GPIO.OUT)

GPIO.setup(green,GPIO.OUT)

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

print("Druk CTRL C om dit programma te beëindigen!")

status\_leds=0

drukknop\_status\_eerder=0

time\_eerder = 0

max\_indruk\_tijd = 2

time\_indrukken=0

time\_ingedrukt=0

status\_red=0

status\_orange=0

status\_green = 0

start\_checking\_tijd\_ingedrukt =0

status\_key\_detection = 0

press\_within\_interval\_counter =0

try:

while True:

drukknop\_status= GPIO.input(drukknop)

if drukknop\_status != drukknop\_status\_eerder:

drukknop\_status\_eerder=drukknop\_status

if drukknop\_status==0 :

print("knop nu ingedrukt")

if status\_key\_detection == 0:

# first press after previous cyclus is handled

print("cyclus start")

time\_indrukken=time.time()

press\_within\_interval\_counter=1

status\_key\_detection =1

else:

press\_within\_interval\_counter+=1

time.sleep(0.05)

if status\_key\_detection!=0 and (time.time() - time\_indrukken > 1.5):

status\_key\_detection=0

print("cyclus stopt")

if press\_within\_interval\_counter==1:

status\_red= not status\_red

GPIO.output(red,status\_red)

print("1 keer gedrukt in 1.5 seconden")

elif press\_within\_interval\_counter==2:

status\_orange= not status\_orange

GPIO.output(orange,status\_orange)

print("2 keer gedrukt in 1.5 seconden")

elif press\_within\_interval\_counter==3:

status\_green= not status\_green

GPIO.output(green,status\_green)

print("3 keer gedrukt in 1.5 seconden")

except:

GPIO.cleanup()

print("cleaned all gpio's")

7-6) Schrijf een programma dat rode, oranje, groene led toggelt bij respektievelijk indrukken van slechts 1 GPIO drukknop gedurende 1,2,3 seconden.

import RPi.GPIO as GPIO

import time

red=23 # refereer naar GPIO pin met variablele

orange=24 # refereer naar GPIO pin met variablele

green= 25 # refereer naar GPIO pin met variablele

drukknop = 18

GPIO.setwarnings(False)

GPIO.setmode(GPIO.BCM)

GPIO.setup(red,GPIO.OUT)

GPIO.setup(orange,GPIO.OUT)

GPIO.setup(green,GPIO.OUT)

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

print("Druk CTRL C om dit programma te beëindigen!")

status\_leds=0

drukknop\_status\_eerder=0

time\_eerder = 0

max\_indruk\_tijd = 2

time\_indrukken=0

time\_ingedrukt=0

status\_red=0

status\_orange=0

status\_green = 0

start\_checking\_tijd\_ingedrukt =0

try:

while True:

drukknop\_status= GPIO.input(drukknop)

if drukknop\_status != drukknop\_status\_eerder:

drukknop\_status\_eerder=drukknop\_status

if drukknop\_status==0 :

time\_indrukken=time.time()

start\_checking\_tijd\_ingedrukt = 1

elif drukknop\_status==1 :

start\_checking\_tijd\_ingedrukt = 0

if time\_ingedrukt > 0.1 and time\_ingedrukt < 0.3:

status\_red= not status\_red

GPIO.output(red,status\_red)

print("time\_ingedrukt > 0.3 and time\_ingedrukt < 0.6 => toggle red")

elif time\_ingedrukt > 0.3 and time\_ingedrukt < 0.8:

status\_orange= not status\_orange

GPIO.output(orange,status\_orange)

print("time\_ingedrukt > 0.6 and time\_ingedrukt < 1.5 => toggle orange")

elif time\_ingedrukt > 0.8 :

status\_green= not status\_green

GPIO.output(green,status\_green)

print("time\_ingedrukt > 1.5 => toggle green")

time.sleep(0.05)

if start\_checking\_tijd\_ingedrukt == 1:

time\_ingedrukt= time.time() - time\_indrukken

except:

GPIO.cleanup()

print("cleaned all gpio's")

7-7) PWM 19, Gebruik knop1 dim+, drukknop2 dim-, knop3 om de leds R,G of O te selecteren.

import RPi.GPIO as GPIO

import time

GPIO.setmode(GPIO.BCM)

GPIO.setwarnings(False)

GPIO.cleanup()

dim\_up = 18

dim\_down = 12

select\_ROG\_LEDS = 26

list\_drukknoppen = [ dim\_up, dim\_down , select\_ROG\_LEDS]

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

led\_rood = 23

led\_oranje = 24

led\_groen = 25

list\_leds = [ led\_rood, led\_oranje, led\_groen]

GPIO.setup(list\_leds, GPIO.OUT)

freq=500

pwm\_red= GPIO.PWM(led\_rood,freq)

pwm\_orange= GPIO.PWM(led\_oranje,freq)

pwm\_green= GPIO.PWM(led\_groen,freq)

pwm\_red.start(0)

pwm\_orange.start(0)

pwm\_green.start(0)

pwm=[pwm\_red,pwm\_orange,pwm\_green] # list die 3 pwm objecten bevat

pwm\_levels= [0,0,0] # list die 3 PWM levels bevat

selected\_rog=0 # 0 = R, 1 = O, 2 = G

def fct\_dim\_down():

global selected\_rog , pwm\_levels , pwm

if pwm\_levels[selected\_rog] > 0:

pwm\_levels[selected\_rog] -= 1

pwm[selected\_rog].ChangeDutyCycle(pwm\_levels[selected\_rog])

print("dim niveau2", selected\_rog, "=", pwm\_levels[selected\_rog])

def fct\_dim\_up():

global selected\_rog , pwm\_levels , pwm

if pwm\_levels[selected\_rog] < 100:

pwm\_levels[selected\_rog] += 1

pwm[selected\_rog].ChangeDutyCycle(pwm\_levels[selected\_rog])

print("dim niveau", selected\_rog, "=", pwm\_levels[selected\_rog])

def select\_r\_o\_g(p):

global selected\_rog

if selected\_rog < 2:

selected\_rog +=1

else:

selected\_rog =0

print("selected color is now",selected\_rog)

GPIO.add\_event\_detect(select\_ROG\_LEDS, GPIO.RISING, callback=select\_r\_o\_g , bouncetime = 250)

while True:

if GPIO.input(dim\_up) == 0 :

fct\_dim\_up()

if GPIO.input(dim\_down) == 0 :

fct\_dim\_down()

7-8) PWM 20, Gebruik slechts 1 drukknop om zowel de 3 leds R,G of O te selecteren en te dimmen.

7-9) MCP3008 regel met een potentiometer aangesloten op de MCP de snelheid van je knipperende rode led tussen 1 en 5 keer per seconde. Gebruik de volledige rondgang van de potentiometer tussen min en max om de snelheid in te stellen.

import RPi.GPIO as GPIO

import time

from gpiozero import MCP3008

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

GPIO.setwarnings(False)

GPIO.setmode(GPIO.BCM)

led\_rood=23

GPIO.setup(led\_rood, GPIO.OUT)

time\_stamp =0

status\_rood=0

def map\_fct(x, in\_min, in\_max, out\_min, out\_max):

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

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

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

while True:

analoge\_waarde=mcp3008\_channel\_0.value

delay\_time\_led= map\_fct(analoge\_waarde ,0,1,0.5,0.1) # map\_fct(analoge\_waarde ,0,1,1,5) will never give 1...5

print(delay\_time\_led) # to check your setting

time.sleep(0.10)

if time.time()-time\_stamp > delay\_time\_led:

time\_stamp=time.time()

status\_rood= not status\_rood

GPIO.output(led\_rood, status\_rood)

7.10) MCP3008 selecteer met de potentiometer welke led r,g,b er brandt. Verdeel het regelgebied van de potentiometer netjes in 3. Er brandt altijd slechts 1vd 3 leds.

import RPi.GPIO as GPIO

import time

from gpiozero import MCP3008

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

GPIO.setwarnings(False)

GPIO.setmode(GPIO.BCM)

led\_rood=23

led\_oranje =24

led\_groen = 25

leds= [led\_rood,led\_oranje,led\_groen] # list

GPIO.setup(leds, GPIO.OUT)

time\_stamp =0

status\_rood=0

led\_on\_before=1

def stuur\_led(led):

global led\_on\_before

if led != led\_on\_before:

GPIO.output(leds[led\_on\_before],0)

led\_on\_before=led

GPIO.output(leds[led],1)

while True:

analoge\_waarde=mcp3008\_channel\_0.value

print(analoge\_waarde) # to check your setting

time.sleep(0.10)

# led0 = rood led1 = oranje led2 = groen

if analoge\_waarde < 0.33:

stuur\_led(0)

elif analoge\_waarde >= 0.33 and analoge\_waarde <= 0.66:

stuur\_led(1)

else:

stuur\_led(2)

7.11) Stuur je servo-motor van links naar rechts met je potentiometer.

import RPi.GPIO as GPIO

import time

from gpiozero import MCP3008

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

GPIO.setwarnings(False)

GPIO.setmode(GPIO.BCM)

servo=17

GPIO.setup(servo, GPIO.OUT)

p = GPIO.PWM(servo, 50) # channel=23 frequency=50Hz

p.start(0)

servo\_dc\_before=0.0

def map\_fct(x, in\_min, in\_max, out\_min, out\_max):

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

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

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

while True:

analoge\_waarde = mcp3008\_channel\_0.value

servo\_dc = map\_fct(analoge\_waarde, 0 , 1 , 2.5 , 12.5)

if servo\_dc\_before != servo\_dc:

servo\_dc\_before = servo\_dc

p.ChangeDutyCycle(servo\_dc)

print("servo\_dc",servo\_dc) # to check your setting

7.12) Stuur je servo-motor van links naar rechts met je GUI schuifregelaar en 3 knoppen ( -90/0/+90)

import RPi.GPIO as GPIO

import time

from tkinter import \*

from gpiozero import MCP3008

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

window = Tk()

window.geometry("320x150") # not \*

window.title("CVOFOCUS THERMOSTAAT")

def links():

global servo\_dc

servo\_dc= 2.5

def midden():

global servo\_dc

servo\_dc= 7.5

def rechts():

global servo\_dc

servo\_dc= 12.5

knop\_links = Button(window , text = "links" , width = 8 , height = 1, command = links)

knop\_links.place(x=10,y=10)

knop\_midden = Button(window , text = "midden" , width = 8 , height = 1, command = midden)

knop\_midden.place(x=110,y=10)

knop\_rechts = Button(window , text = "rechts" , width = 8 , height = 1, command = rechts)

knop\_rechts.place(x=210,y=10)

def servo\_slider(p):

global servo\_dc

servo\_dc= float(p)

slider= Scale(window, from\_=2.5 , to = 12.5 , length= 300, orient="horizontal" , resolution = 0.5 ,command = servo\_slider)

slider.place(x=10,y=50)

GPIO.setwarnings(False)

GPIO.setmode(GPIO.BCM)

servo=17

GPIO.setup(servo, GPIO.OUT)

p = GPIO.PWM(servo, 50) # channel=23 frequency=50Hz

p.start(0)

servo\_dc\_before=0.0

servo\_dc=0

while True:

window.update()

if servo\_dc\_before != servo\_dc:

servo\_dc\_before = servo\_dc

p.ChangeDutyCycle(servo\_dc)

print("servo\_dc",servo\_dc) # to check your setting