

h e p i a

Haute école du paysage, d'ingénierie  
et d'architecture de Genève

# HEPIALIGHT3 - $R\pi$ PICO

Michael Divià  
Alejandro Escribano Martín  
Gaspard Le Gouic

Bachelor I.S.C - HEPIA  
28 Juin 2024  
Genève, Suisse

version 1.1

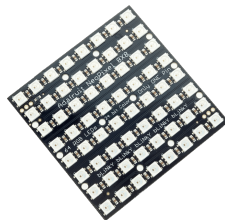
- 1 Introduction
- 2 Points clés du projet
- 3 Problèmes rencontrés
- 4 Conclusion
- 5 Bibliographie

## Idées de départ :

- Évaluer le portage de HepiaLigh2 sur différentes nouvelles architectures
- RP2040-PICO-HDR & Adafruit NeoPixel NeoMatrix 8x8 - 64 RGB LED Pixel Matrix
- MicroPython



RPi Pico



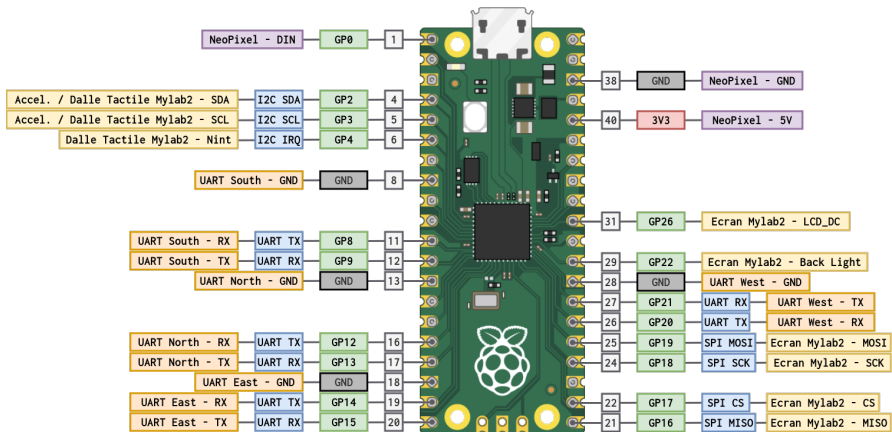
NeoPixel NeoMatrix

- Maintient d'un journal de développement
- Utilisation de Git comme système de contrôle de versions
- Répartition des tâches principales entre les différents intégrants
- Utilisation de Thonny pour itérer et flasher les nouvelles versions du code sur la plateforme
- Mise en commun régulière du travail réalisé

# Avantages / Désavantages de la plateforme

- + Facilité / Rapidité de développement
- + Bas Prix : RP2040 - CHF 0.644 // NeoMatrix - CHF 27
- + Communauté active, documentation abondante
- + Faible consommation

- + Facilité / Rapidité de développement
- + Bas Prix : RP2040 - CHF 0.644 // NeoMatrix - CHF 27
- + Communauté active, documentation abondante
- + Faible consommation
- Limitation du nombre de périphérique
- Fonctionnalité avancée demande des connaissances poussées
- Pas de sortie 5V, seul 3V3



Raspberry Pi Pico Pinout for HepiaLight3

## Affichage

```
Matrix.clear(Color)
Matrix.set_line(line, Color)
Matrix.set_column(line, Color)
Matrix.set_led(column, line, Color)
Matrix.get_led(column, line)
show_text(text, Color, speed)
set_img(matrix)
```

## Communication

```
Uart(Direction, baudrate, parity, bits, stop)
x.send(string)
x.sendline(string)
x.receive(length)
x.receivevline(length)
```

## Capteur externe - accéléromètre (I2C)

```
Accel.init()  
Accel.get_x()  
Accel.get_y()  
Accel.get_z()  
Accel.facing(side)  
Accel.tilting(dir)  
accel_test()
```

## Capteur externe - touchscreen (I2C)

```
Touch.attach(cb)
Touch.detach()
Touch.callback_do()
Touch.init()
Touch.read(pos)
Touch.compute_pos(x, y)
Touch.read_pos()
Touch.touch_cb(pos)
Touch.handler()
touch_test()
touch_irq_test()
```

## Display externe LCD (SPI)

```
Lcd.write_cmd(cmd)
Lcd.write_data(data)
Lcd.init()
Lcd.set_window
lcd_test()
```

- Communication sans fil, CYW43439 : Wi-Fi+Ble (CHF 3.48)
- Communication sans fil, ESP32-WROOM-32 (CHF ~3.00)

- Communication SPI avec l'écran lente
- UART supplémentaire : besoin impératif de `asm_pio`
- Surcouche de complexité pour faire fonctionner ensemble le RPi Pico avec ESP32
  - Besoin d'un "set de commandes" supplémentaire (AT Command Set)
  - Rajoute un  $\mu$ C/co-processeur à l'ensemble

## Projet :

- RP2040 : puissant, robuste, économique & efficace
- Communauté active et documentation abondante
- Modernisation et conservation des fonctionnalités clés de HepiaLight2
- Faisabilité technique démontrée malgré les défis

## Projet :

- RP2040 : puissant, robuste, économique & efficace
- Communauté active et documentation abondante
- Modernisation et conservation des fonctionnalités clés de HepiaLight2
- Faisabilité technique démontrée malgré les défis

## Personnelle :

- Compétences en programmation embarquée et gestion de projet consolidées
- Familiarisation avec bibliothèques et communication UART
- Objectifs atteints & adaptation réussie

## Questions ?



Merci de nous avoir écouté !

- [1] HepiaLight 2. Github. 2024. url : <https://gitedu.hesge.ch/cores/projects/hepialight2>.
- [2] MicroPython documentation. MicroPython. 2024. url : <https://docs.micropython.org/en/latest/>.
- [3] MyLab2. hepialsn - HESO-SO. 2024. url : <https://hepialsn.hesge.ch/myLab2/>.
- [4] Raspberry Pi Pico documentation. Raspberrypi. 2024. url : <https://www.raspberrypi.com/documentation/microcontrollers/raspberry-pi-pico.html>.
- [5] Thonny IDE. Thonny. 2024. url : <https://github.com/thonny/thonny/>.

