MicroPython sur Raspberry-Pi Pico

Tome 1 : découvrir le Pico

Présentation

La fondation Raspberry-Pi s’est fait connaître avec ses célèbres nano-ordinateurs Raspberry-Pi. En 2021, la fondation révolutionne une nouvelle fois le monde de l’apprentissage en éditant son premier microcontrôleur, le RP2040 qui équipe le **Raspberry-Pi Pico**. Le Pico fonctionne nativement sous MicroPython et à rapidement trouvé sa place auprès des Makers mais aussi des grands acteurs de l’électronique et de l’industrie !

Cet ouvrage permet de **découvrir** et **maîtriser MicroPython, Python** sur microcontrôleur, avec les cartes **Raspberry-Pi Pico**. Aller au-delà des concepts et découvrir les **aspects pratiques** du langage Python appliqué aux microcontrôleurs ! Cet ouvrage guide son lecteur de façon didactique dans la mise en œuvre des **cartes Pico**, sans oublier d'explorer la prise de contrôle de **composants électroniques** que l'on peut y raccorder.

Les Makers pourront facilement y trouver de quoi adapter leur savoir-faire Arduino sur MicroPython, ceux qui ne sont pas encore des **Makers** pourraient fort bien le devenir, **les programmeurs** ajouteront sans peine une nouvelle corde à leur arc, **les ingénieurs** bâtiront de nouveaux ponts entre différents domaines du savoir, **l'électronicien** découvrira un outil pour produire rapidement des prototypes et **les pédagogues** pourront conjuguer facilement théorie et expérimentation. Enfin, les utilisateurs de la MicroPython Pyboard originale se sentirons comme à la maison.

Chapitre après chapitre, le lecteur part à la découverte de la **mise en œuvre de la carte Pico**, à la **réalisation de montages électroniques** simples, jusqu'à l'**utilisation de techniques avancées** telles que la manipulation de différents **capteurs et interfaces**, des **sorties PWM et analogique** ou d'un **bus I2C**. Il peut ainsi apprendre par exemple à allumer une LED, activer un relais, commander des moteurs ou encore acquérir des données environnementales et afficher des informations sur des écrans.

Qu’est-ce que MicroPython ?

- 1. Mise en garde : euphorie imminente !
- 2. Avant-propos : premier contact MicroPython
- 3. Objectifs de l’ouvrage
- 4. Prérequis
- 5. Présentation de MicroPython
- 6. Comparaison MicroPython et Arduino
    - 6.1 Arduino
    - 6.2 MicroPython
- 7. Intérêt
    - 7.1 Python pour l’électronique
    - 7.2 Python, un langage populaire

        - 7.3 Python et apprentissage rapide
        - 7.4 Python et l’enseignement
- 8. Communauté
    - 8.1 Bibliothèques et pilotes
    - 8.2 Forums

Plateformes MicroPython

- 1. Préambule
- 2. À l’assaut du monde professionnel
- 3. Critères de sélection
    - 3.1 Tension logique
    - 3.2 Fréquence CPU
    - 3.3 Mémoire RAM
        - 3.3.1 Durant la compilation des scripts
        - 3.3.2 Durant le fonctionnement du script
    - 3.4 Mémoire flash
    - 3.5 Calcul en virgule flottante (FPU)
    - 3.6 Communication
    - 3.7 Communauté
    - 3.8 Support matériel de MicroPython
- 4. Vue d’ensemble des plateformes disponibles
    - 4.1 STMicroelectronics (Pyboard, Pyboard-D)
    - 4.2 Raspberry-Pi (Pico, RP2040)
    - 4.3 Microchip/Atmel (plateformes Adafruit)
    - 4.4 Nordic Semiconductor (Micro:bit)
    - 4.5 Espressif (ESP8266 et ESP32)
    - 4.6 Pycom.io
    - 4.7 Risc-V
    - 4.8 Autres plateformes
- 5. MicroPython et CircuitPython

Raspberry-Pi Pico

- 1. Microcontrôleur RP2040
- 2. Présentation du Pico
    - 2.1 MicroPython
    - 2.2 Raspberry-Pi Pico
        - 2.2.1 En cas de problème
        - 2.2.2 Premier survol du Pico
        - 2.2.3 Examen physique
    - 2.3 Système de fichiers MicroPython
    - 2.4 Quels connecteurs pour le Pico ?
    - 2.5 Pico et Fritzing
- 3. Le Pico en détail
    - 3.1 Interfaces matérielles
        - 3.1.1 LED utilisateur

        - 3.1.2 Bouton Boot0
        - 3.1.3 Ajouter le bouton Reset
        - 3.1.5 Mémoire Flash et système de fichier
        - 3.1.6 Alimentation du Pico
        - *** en option *** Horloge RTC, Support MicroSD
- 4. Pico : tension logique et courant
    - 4.1 Niveau logique et tensions
    - 4.2 Niveau logique et Python
    - 4.3 Tolérance 5 V ? NON !
    - 4.4 Courants maximum, source et sink
    - 4.5 Injection de courant
- 5. Les fonctions alternatives sur le Pico
    - 5.1 Sortie PWM
    - 5.2 Entrée analogique (ADC)
    - 5.3 Sortie analogique (DAC)
    - 5.4 Bus I2C
    - 5.5 Bus SPI
    - 5.6 UART (port série)
- 6. Brochage du Pico
    - 6.1 Partie droite
    - 6.2 Partie gauche
    - 6.3 Déboggeur
- 7. Brochage avancé et timers **** Contenu à confirmer ***
    - 7.1 Broches du RP2040
    - 7.2 Timers
- 8. Comment détruire sa Pico en sept leçons ?
    - 8.1 Placer une broche directement à la masse
    - 8.2 Brancher des GPIO ensemble
    - 8.3 Appliquer une surtension sur une broche d’entrée
    - 8.4 Appliquer une tension d’alimentation inversée sur V+
    - 8.5 Appliquer une tension supérieure à 3,3 V sur la broche 3V3
    - 8.6 Dépasser le courant max d’une broche
    - 8.7 Dépasser le courant max du microcontrôleur

Environnement de travail

- 1. Avant-propos
- 4. Console série et REPL
    - 4.1 PuTTY (multiplateforme)
    - 4.2 Picocom (Linux)
    - 4.3 Screen (Mac, Linux)
- 5. Outils intégrés
    - 5.1 Thonny
        - 5.2.1 Installation Linux, Raspberry-Pi OS, Windows, Mac
        - 5.2.2 Utilisation avec Pico
    - 5.2 Ampy

- 5,2 RShell
    - 5.1.1 Linux et Raspbian
    - 5.1.2 Windows
    - 5.1.3 Mac OS

Prise de contrôle

- 1. Communiquer avec MicroPython
- 2. Utiliser Thonny avec le Pico
- 3. Manipuler le Pico avec Ampy
- 4. REPL : l'invite en ligne de commande
    - 4.1 Séquence de contrôle REPL
    - 4.2 Options avancées sur REPL
        - 4.2.1 Édition de ligne
        - 4.2.2 Historique de commandes
        - 4.2.3 Autocomplétion
        - 4.2.4 Variable « _ »
    - 4.3 Outils Python avancés pour REPL
        - 4.3.1 Fonction help()
        - 4.3.2 Fonction dir()
        - 4.3.3 Fonction listdir()
        - 4.3.4 Afficher le contenu d’un fichier
    - 4.4 Développer avec REPL
- 6. RShell
    - 6.1 Ligne de commande RShell
    - 6.2 REPL sous RShell

Séquence de démarrage

- 1. Séquence de démarrage MicroPython
- 2. Fichier boot.py
- 3. Fichier main.py
- 4. En cas de problème

Programmer

- 1. Préambule
- 2. Les bibliothèques MicroPython
- 4. Bibliothèque machine
    - 4.1 Limitation de la portabilité
    - 4.2 Quel intérêt pour la portabilité ?
    - 4.3 Contenu de la bibliothèque machine
- 9. Entrées/sorties
    - 9.1 Entrée numérique
    - 9.2 Entrée numérique (pull-up interne)
    - 9.3 Entrée numérique et déparasitage
    - 9.4 Entrée numérique et interruption
    - 9.5 Sortie numérique
        - 9.5.1 Commander une LED

        - 9,5,2 Commande de relais
    - 9.6 Entrée analogique
        - 9.6.1 Potentiomètre
        - 9.6.2 Lecture analogique
        - 9,6,3 Photo-résistance
    - 9.7 Sortie PWM
        - 9.8.1 Commander l’intensité d’une LED
    - 9.8 Commande Servo
- 10. Identification et mode des broches

Apprentissage par la pratique

- 1, Préambule
- 2, Feu de circulation
  Rouge, Vert, Jaune, bouton
- 3, Jeu de rapidité
  2 boutons, 3 Leds… qui réagit le plus vite lorsque la LED s’allume
- 4, Détecteur de présence
  Allumer une LED (de la lumière) lorsqu’une présence est détectée.
- 5. Thermomètre simple
  Utiliser un capteur analogique TMP36 et modifier la luminosité d’une LED en fonction de la température
- 6. Thermomètre baregraphe
  Utiliser un bar-graphe pour indiquer la température.
- 7. DataLogger
  Utiliser le système de fichier MicroPython pour capturer la température (Tmp36), Luminosité (LDR) et humidité du sol (SEN0308, SEN0193). Information enregistrée toutes les 5 minutes.
- 8. Robot 2 roues (servo à rotation continue)
  Créer un petit robot 2 roues + capteurs ultrason (éviter qul ne rencontre un mur).
- 9. Créer un minuteur
  Créer un petit minuteur (minutes et seconde) à l’aide d’un afficheur 2 lignes et d’un bouton.

Tome 2 : Approfondir & maîtriser

Présentation

Le tome 1 de « MicroPython sur Raspberry-Pi Pico » survolait les possibilités offertes par la carte Raspberry-Pi PICO et la programmation Python sur microcontrôleur. Nulle doute que l'esprit du lecteur fourmille maintenant d'idées. Ce deuxième volet permet d'approfondir les connaissances permettant la réalisation de très nombreux autres projets. Cet ouvrage s'attarde sur l'utilisation de nouveaux capteurs, de cartes d'interface (carte relais), interface d'affichage, LEDs intelligentes et autres. Cet ouvrage s'appuie sur des gammes de produits faciles à se procurer.

Les périphériques d'affichages généralement relégués en fin d'ouvrage seront rapidement abordés dans ce manuel. Les afficheurs, avec les boutons, sont des éléments incontournables des interfaces homme-machine. La maîtrise précoce l'afficheur 4x20 caractères permet, par la suite, de proposer des exemples plus proches de « mini-projets » que de la fiche de cours.

Au terme de ce deuxième volet le lecteur d'envisager la création et réalisation de projet plus avancés.

Raspberry-Pi Pico (rappel)

- 1. Présentation du Pico
- 2, Brochage du Pico
- 3. Petit rappel sur Thonny (et copie de bibliothèque)

Programmer

- 1. Préambule
- 2. Les bibliothèques MicroPython
    - 2.1 Le préfixe u
    - 2.2 Bibliothèques dans le firmware
    - 2.3 Mécanisme de chargement
    - 2.4 Où placer les bibliothèques ?
    - 2.5 Écrire ses propres bibliothèques
        - 2.5.1 Script de test
        - 2.5.2 Création des bus
- 3. Bibliothèques disponibles
    - 3.1 Bibliothèques standards et micropythonifiées
    - 3.2 Bibliothèques propres à MicroPython
    - 3.3 Bibliothèques spécifiques à la carte de développement
- 4. Bibliothèque machine
    - 4.1 Limitation de la portabilité
    - 4.2 Quel intérêt pour la portabilité ?
    - 4.3 Contenu de la bibliothèque machine
- 5. Bibliothèque rp2
- 6. Bibliothèque os
- 7. Charger et exécuter un script à la volée
- 9. Entrées/sorties (rappel)

- 9.1 Entrée numérique
  Pull-up, déparasitage, interruption, utilisation d'un mini joystick
- 9.2 Sortie numérique
  - 9.5.1 Commander une LED
  - 9.5.2 Broche en sortie et courant de court-circuit
  - 9.5.3 Montage drain ou source
- 9.6 Entrée analogique
  - 9.6.2 Lecture analogique
  - 9.6.3 Précision des convertisseurs
  - 9.6.4 Échantillonnage
  - 9.6.5 Acquisition de signal et fréquence d'échantillonnage
- 9.7 Sortie PWM
- 9.8 Commande Servo
  - 9.9.1 Synchroniser des servomoteurs
  - 9.9.2 Servomoteurs à rotation continue
  - 9.9.3 Alimentation des servomoteurs
  - 9.9.4 Calibration des servomoteurs

- 10. Identification et mode des broches
- 11. Les timers *** A revoir ***
  - 11.1 Timers disponibles
  - 11.2 Fonction de rappel/d'interruption
  - 11.3 Fonctions d'interruption et bonnes pratiques
  - 11.4 Timers avancés
  - 11.5 Le timer WatchDog
- 12. Bus I2C
  - 12.1 I2C : comment ça marche ?
    - 12.1.1 À propos des bits d'adresse
    - 12.1.2 I2C comme un périphérique mémoire
  - 12.2 Capteur I2C et pilote MicroPython
    - 12.2.1 Où trouver des pilotes ?
    - 12.2.2 Comment utiliser un pilote I2C
    - 12.2.3 Faut-il créer le bus I2C hors du pilote ?
  - 12.3 Connecteurs standardisés pour I2C
    - 12.3.1 Connecteur UEXT d'Olimex
    - 12.3.2 Connecteur StemmaQt d'Adafruit
    - 12.3.3 Connecteur Qwiic de SparkFun
  - 12.4 Communication I2C par l'exemple
    - 12.4.1 Exemple 1 : afficheur 4x7 Segments
    - 12.4.2 Exemple 2 : ADS1115 (entrées analogiques supplémentaires)
    - 12.4.3 Exemple 3 : MOD-IO (carte relais I2C)
  - 12.5 Le bus I2C plus en détail
    - 12.5.1 Résistances pull-up et tension logique
    - 12.5.2 Vitesse limitée
    - 12.5.3 Effet capacitif et longueur de ligne
    - 12.5.4 LTC4311 pour augmenter la longueur du bus
- 13. Bus SPI

        - 13.1 Utilisation de l’API SPI
        - 13.2 Classe SPI du module machine
        - 13.3 Communication SPI par l’exemple
            - 13.3.1 Exemple 1 : ADS9833 pour générer du son
            - 13.3.2 Exemple 2 : matrice LED (Olimex)
- 14. Interface UART
    - 14.1 La trame série
    - 14.2 Configurer une ligne série
    - 14.3 Émission/réception
    - 14.4 Module GPS - communication UART par l’exemple
- 15. Horloge RTC *** A vérifier ****

Capteurs et interfaces

- 1. Introduction
- 2. Signal numérique
    - 2.1 Module relais
        - 2.1.1 Mise en garde
        - 2.1.2 À réaliser soi-même
        - 2.1.3 Modules relais préassemblés
        - 2.1.4 Plusieurs modules relais
        - 2.1.5 Relais et impulsions électromagnétiques (EMI)
        - 2.1.6 PowerSwitchTail
    - 2.2 Contact magnétique (interrupteur)
    - 2.3 Capteur à effet Hall
    - 2.4 Capteur de mouvement (PIR)
    - 2.5 Encodeur rotatif
    - 2.6 LED RGB
    - 2.7 Capteur ultrason HC-SR04
        - 2.7.1 Brancher le HC-SR04
        - 2.7.2 Tester le HC-SR04
        - 2.7.3 Bibliothèque
- 3. Signal analogique
    - 3.1 Capteur de température TMP36 (rappel)
    - 3.2 Capteur de pression différentielle MPXV5010DP
    - 3.3 Capteur de niveau sonore
    - 3.4 Mesure de courant avec ACS712
        - 3.4.1 Mise en garde
        - 3.4.2 Charge alternative et relevé alternatif
        - 3.4.3 Mesure avec la Pico
    - 3.5 Mesure d’humidité de sol par effet capacitif

- 4. Afficheurs
    - 4.1 LED NeoPixel (WS2812)
    - 4.2 LED DotStar (APA102)
    - 4.3 SD1306 : afficheur OLED
        - 4.3.1 Brancher l'afficheur OLED

        - 4.3.2 Installer la bibliothèque
        - 4.3.3 Tester l’afficheur OLED
        - 4.3.4 Manipulations du FrameBuffer
        - 4.3.5 Image PBM
        - 4.3.6 Ressources
    - 4.4 MOD-LCD : afficheur à cristaux liquides
    - 4.5 Afficheur LCD 4 lignes I2C (DFR-LCD-20x4-WHITE-I2C)
    - 4.6 ILI9341 : afficheur TFT couleur

- 5. Interface I2C
    - 5.0 MPR121 : capteur capactif
    - 5.1 MCP23017 : extension d’entrée/sortie
        - 5.1.1 Brancher le MCP23017
        - 5.1.2 Bibliothèque
        - 5.1.3 Tester le MCP23017
    - 5.2 ADS1115 : entrée analogique
        - 5.2.1 Brancher l' ADS1115
        - 5.2.2 Mille milliards de mille parasites !
        - 5.2.3 Le gain programmable
        - 5.2.4 Bibliothèque
        - 5.2.5 Tester l’ADS1115
    - 5.3 DHT11 / DHT22 : humidité et température
        - 5.3.1 Brancher le DHT 11
        - 5.3.2 Tester le DHT11
    - 5.4 BMP280 / BME280 : humidité, pression et température
        - 5.4.1 Brancher le BME280
        - 5.4.2 Bibliothèque
        - 5.4.3 Tester le BME280
        - 5.4.4 Calcul d’altitude
        - 5.4.5 Modes de fonctionnement du BME280
    - 5.5 TSL2591 : luminosité
        - 5.5.1 Brancher le TSL2591
        - 5.5.2 Bibliothèque
        - 5.5.3 Tester le TSL2591
    - 5.6 SI1145 : luminosité visible/infrarouge/index UV
        - 5.6.1 Brancher le SI1145
        - 5.6.2 Bibliothèque
        - 5.6.3 Tester le SI1145
    - 5.7 MOD-IO : relais et entrées optocoupleurs (24 V)
    - 5.8 Contrôleur Wii Nunchuck UEXT
        - 5.8.1 Brancher le Nunchuk
        - 5.8.2 Bibliothèque
        - 5.8.3 Tester
    - 5.9 MCP4725 : sortie analogique
        - 5.9.1 Brancher le MCP4725
        - 5.9.2 Bibliothèque

        - 5.9.3 Tester le MCP4725
        - 5.9.4 Une onde en dos de chameau
    - 5.10 MCP9808 : capteur de température de précision
        - 5.10.1 Brancher le MCP9808
        - 5.10.2 Bibliothèque
        - 5.10.3 Tester le MCP9808
- 6. Contrôle moteur
    - 6.1 Servomoteurs sur la Pyboard
    - 6.2 L298 : pont-H pour moteur continu
    - 6.3 L293D : pont-H pour moteur continu
    - 6.4 DRV8833 : contrôleur moteur continu
    - 6.5 PCA9685 : contrôleur PWM et servo moteur
    - 6.6 A4988 : contrôleur moteur pas-à-pas
        - 6.6.1 Découvrir le moteur pas-à-pas
        - 6.6.2 Contrôleur de moteur pas-à-pas A4988
        - 6.6.3 Tester le A4988

- 7. Interface UART
    - 7.1 Module GPS
        - 7.1.1 Brancher le module GPS
        - 7.1.2 Bibliothèque
        - 7.1.3 Tester le module GPS
    - 7.2 Lecteur de carte RFID
        - 7.2.1 Brancher le module RFID
        - 7.2.2 Bibliothèque
        - 7.2.3 Tester le module RFID
    - 7.3 Capteur ultrasonique A01NYUB
    - 7.4 Module MP3 (GRAV-MP3-VOICE-8MB)

Fonctionnalités avancées du Pico

- 1. PIO : Programmable I/O
- 2, Developpement multi-coeurs
- 3. Débogeur

Classes MicroPython courantes

- 1. Les classes MicroPython en français
- 2. La classe ADC
    - 2.1 Constructeur
    - 2.2 Méthodes
- 3. La classe ADCAll
    - 3.1 Constructeur
- 4. La classe DAC
    - 4.1 Constructeur
    - 4.2 Méthodes

- 5. La classe I2C
    - 5.1 Constructeur
    - 5.2 Méthodes
    - 5.3 Méthodes - opérations primitives
    - 5.4 Méthodes - opérations standards
    - 5.5 Méthodes - opérations mémoire
- 6. La classe Pin
    - 6.1 Constructeur
    - 6.2 Méthodes
- 7. La classe RTC **** A Revoir ***
    - 7.1 Constructeur
    - 7.2 Méthodes
- 8. La classe Servo
    - 8.1 Constructeur
    - 8.2 Méthodes
- 9. La classe Signal
    - 9.1 Constructeur
    - 9.2 Méthodes
- 10. La classe SPI
    - 10.1 Constructeur
    - 10.2 Méthodes
- 11. La classe Timer *** A revoir ****
    - 11.1 Constructeur
    - 11.2 Méthodes
- 12. Classe TimerChannel *** A revoir ****
    - 12.1 Méthodes
- 13. Classe UART
    - 13.1 Constructeur
    - 13.2 Méthodes

Annexes

- 1. À propos des annexes
- 2. Mise à jour du firmware
- 3. Interrupteur RunApp
- 4. Conversion des logiques 3,3 V et 5 V
    - 4.1 Logiques TTL 5 V et CMOS 3,3 V
    - 4.2 Sortie 3,3 V vers entrée 5 V
    - 4.3 Sortie 5 V vers entrée 3,3 V
    - 4.4 Conversion bidirectionnelle 5 V - 3,3 V
    - 4.5 Niveau logique et bus I2C
    - 4.6 Conversion et haut débit
- 5. PWM vers analogique
- 6. Plateformes RP2040 disponibles
- 7. Schéma du Pico