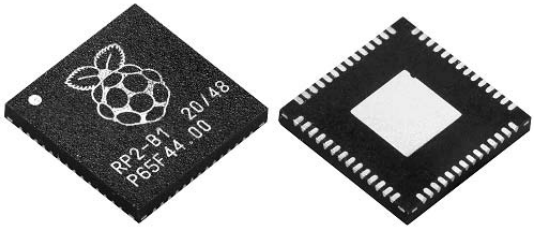
Chapitre 3
Raspberry-Pi Pico

1. Microcontrôleur RP2040

Le Raspberry-Pi Pico, dont il est question dans cet ouvrage, exploite le microcontrôleur RP2040 également conçu par la fondation Raspberry-Pi.

La conception du premier microcontrôleur de la fondation Raspberry-Pi est une grande réussite et surpasse, sur bien des points, les concurrents du marché.

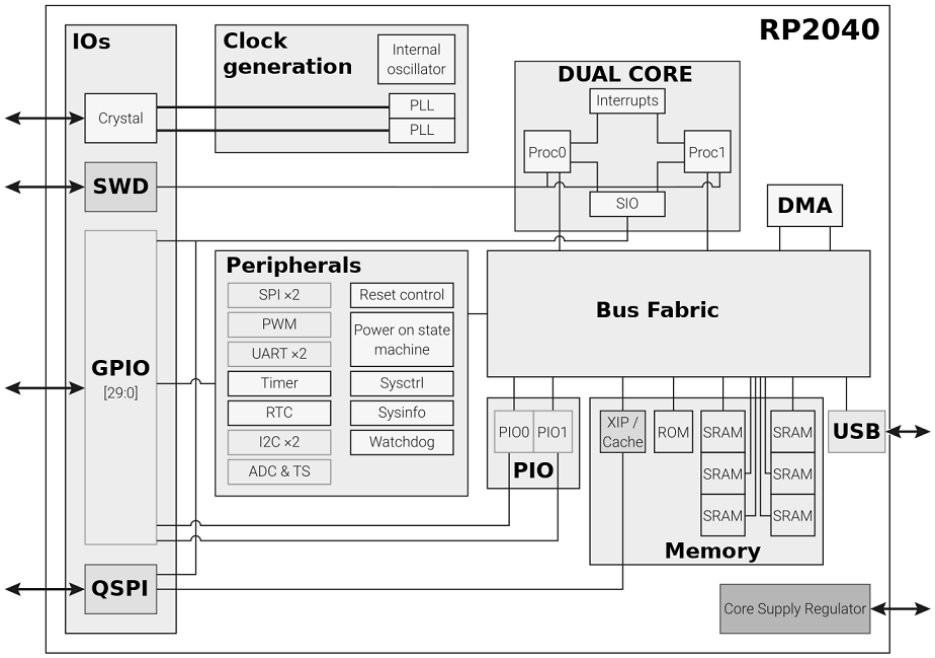


Microcontrôleur RP2040

Pour commencer, ce microcontrôleur dispose d'un double cœur, ce qui n'est pas commun pour un composant à 1,30 euro.

Ensuite, il s'agit d'un processeur haute performance cadencé à 133 Mhz disposant de tous les périphériques standards d'un microcontrôleur standard (bus I2C, bus SPI, bus série/UART, Timer), tout en étant accompagné d'autres fonctionnalités avancées.

Le graphique ci-dessous décrit les différents blocs fonctionnels du RP2040.



Blocs fonctionnels du RP2040

Voici ce qu'il est possible d'apprendre dans ces blocs fonctionnels :

- Le **bloc mémoire** (*memory*) reprend six banques de mémoire RAM pour un total de 264 Kio, ce qui est très conséquent par rapport à un Arduino UNO qui ne dispose que de 2 Kio. Le bloc mémoire contient aussi un élément ROM (mémoire en lecture seule) contenant, entre autres, le *bootloader* UF2 permettant de téléverser un nouveau firmware via USB.
- Le **bloc USB** indique que le RP2040 dispose d'un support USB natif. Celui-ci est utilisé pour téléverser un nouveau firmware mais permet aussi au RP2040 d'établir une connexion USB-Série, d'émuler un périphérique HID comme une souris ou un clavier.
- Le **bloc Dual Core** reprend les deux cœurs totalement indépendants ! Il sera donc possible de faire fonctionner deux scripts MicroPython indépendamment l'un de l'autre. Le sous-bloc SIO (*Software control gpIO*) concerne les modifications d'état de broche par l'exécution du programme utilisateur.

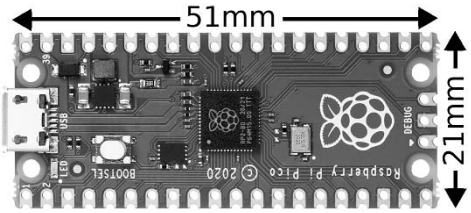
- Le **bloc DMA** (*Direct Memory Access*) permet des accès directs en mémoire, comme effectuer un échantillonnage en stockant directement les informations en mémoire sans passer par l'un des cœurs d'exécution.
- Le **bloc périphériques** (*Peripherals*) reprend la liste des périphériques usuels supportés sur un microcontrôleur. Cet ouvrage permettra d'aborder plusieurs des périphériques disponibles sur le RP2040.
- Le **bloc PIO** (*Programmable IO*) est une spécificité du RP2040. Il est composé de deux entités PIO0 et PIO1 reprenant des machines à états permettant d'exécuter, **de façon autonome**, du code rattaché à des entrées/sorties. Cela permet de supporter/créer un périphérique spécifique sans charger les deux cœurs. À titre d'exemple, ce procédé est utilisé pour l'émission des données vers un ruban NeoPixel, un processus très exigeant sur les temps d'exécution. PIO permet de mettre en place ce protocole matériel sans consommer ni bloquer des ressources processeurs. PIO peut aussi être utilisé pour créer un port parallèle à haut débit.
- Le **bloc GPIO** (*General Purpose Input Ouput*) concerne les trente broches d'entrées/sorties physiques réparties tout le long des bordures du microcontrôleur. Ce bloc est manipulé par les blocs périphériques, PIO, SIO. Le lecteur attentif remarquera une relation indirecte entre les GPIOs et les deux cœurs du microcontrôleur (ainsi que les PIOs) par l'intermédiaire du Bus Fabric.
- Le **bloc Bus Fabric** est un des points forts du microcontrôleur RP2040 qui sera surtout exploité dans des applications industrielles. Le Bus Fabric permet de connecter un bus de données (I2C, SPI, UART, etc.) sur un choix de broches du RP2040, un peu comme un tableau d'interconnexion géant au sein même du microcontrôleur. Il s'agit d'une fonctionnalité avancée qui ne sera pas exploitée dans cet ouvrage, pour ne pas compliquer inutilement les aspects techniques.
- Le **bloc QSPI** (*Quad* SPI) est un bus périphérique utilisé pour la communication avec un module de mémoire flash externe. Un bus Quad SPI peut utiliser jusqu'à quatre lignes de données pour accéder à de la mémoire externe permettant ainsi la lecture de plusieurs bits en simultané. Plus il est possible de lire des bits en même temps et moins d'opérations de lectures seront nécessaires pour lire une même quantité de données. Ce bus à haut débit est utilisé par le RP2040 pour y attacher la mémoire Flash de stockage.
- Le **bloc SWD** (*SoftWare Debug*) est une interface matérielle permettant de déboguer les programmes utilisateurs en cours de fonctionnement sur le microcontrôleur. Cette fonctionnalité avancée concerne surtout les programmes utilisateurs écrits en C/C++. Pour sa part, MicroPython offre d'autres possibilités plus simples d'emploi.
- Le **bloc Quartz** (*Crystal*), disponible sur les entrées/sorties du microcontrôleur, est utilisé pour y brancher l'oscillateur/quartz générant le signal d'horloge à 133 Mhz. Ce signal d'horloge est nécessaire pour cadencer l'exécution des instructions par les cœurs.

2. Présentation du Pico

2.1 Survol du Pico

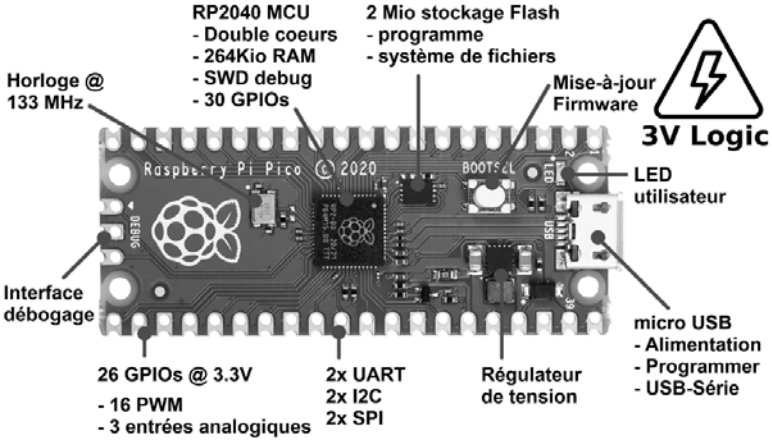
Le Raspberry-Pi Pico est la plateforme de référence de la fondation pour le microcontrôleur RP2040.

Cette plateforme contient l'implémentation minimale d'un microcontrôleur RP2040 pour la modique somme de 5 euros.



Raspberry-Pi Pico

Placée sur une carte de 5 x 2 cm, la plateforme apporte tous les éléments nécessaires au développement de solution en C/C++, Arduino, MicroPython.



Éléments du Raspberry-Pi Pico

2.1.1 Port micro-USB

Le connecteur USB peut être utilisé pour alimenter la plateforme en 5 V mais permet également de brancher le Pico sur un ordinateur pour :

- téléverser un nouveau programme sur la carte (croquis Arduino ou MicroPython) lorsque la carte est en mode de programmation,
- obtenir une liaison série via la connexion USB (terminal Arduino ou ligne de commande MicroPython/REPL),
- se comporter comme un hôte USB permettant ainsi d'émuler des périphériques tels que clavier ou souris.

2.1.2 Broches d'entrées/sorties

La carte Pico n'expose pas moins de 26 broches GPIOs pouvant être utilisées en entrée ou en sorties.

Remarque

Le Pico est un microcontrôleur en logique 3,3 V et n'est pas tolérant 5 V ! Il ne faut, en aucun cas, dépasser la tension de 3,3 V sur l'une des broches GPIOs du Pico.

16 de ces broches peuvent être utilisées en PWM (*Pulse Width Modulation*) pour moduler la largeur d'impulsion d'un signal. Dans le monde Arduino, cette fonctionnalité est abusivement appelée « sortie analogique », car elle permet de moduler la puissance de sortie de signal (par exemple, la luminosité d'une LED), faisant passer la fonctionnalité PWM pour une pseudo-sortie analogique.

Trois de ces broches étant aussi des entrées analogiques, elles permettent de lire une tension d'entrée entre 0 et 3,3 V.

La LED utilisateur, placée juste à côté du port USB, est également branchée sur la broche GPIO (25) pour le Pico. Attention, il n'en va pas de même pour le Pico Wireless.

Enfin, ces GPIOs servent également de support pour les différents bus de données (SPI, I2C, UART) pouvant être utilisés comme sur le Raspberry-Pi ou un Arduino.

À noter que le GPIO 25 n'est pas la seule broche non accessible sur les connecteurs de la carte (dite « *Not breakout* »). Parmi ces GPIOs, on trouve :

- GPIO 24 : permettant de savoir si une tension d'alimentation est présente sur le port micro-USB. Donc si VBUS = 5 V.
- GPIO 29 : permet de relever la tension présente sur la broche VSYS. Ce GPIO est l'entrée analogique 3 (ADC3) et la tension est mesurée par l'intermédiaire d'un pont diviseur de tension qui divise VSYS par 3 (ADC3 = VSYS/3). Cette broche est particulièrement utile pour surveiller la tension d'un accumulateur LiPo ou bloc pile utilisé pour alimenter le Pico.
- GPIO 23 : permet d'activer un mode d'économie d'énergie sur le régulateur d'alimentation du Pico en plaçant le GPIO 23 au niveau bas. Le mode d'économie ne peut être activé que lorsque le Pico consomme peu. Dans ce mode, le hacheur de tension passe en fréquence variable, ce qui permet de faire une économie de puissance significative sur le transistor de hachage. La fréquence de hachage est ajustée pour maintenir la tension à 3,3 V en sortie du régulateur, mais parfois au prix d'une ondulation plus importante du courant de sortie. En cas de forte consommation, l'état du GPIO 23 est ignoré.

2.1.3 Le microcontrôleur RP2040

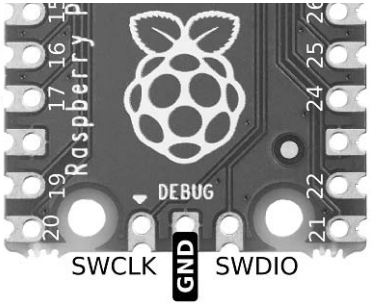
Déjà abordé en début de chapitre, le microcontrôleur RP2040 apporte à MicroPython (tout comme aux autres langages) des caractéristiques hors du commun pour un prix dérisoire. Si le RP2040 offre les services traditionnels d'un microcontrôleur, il apporte aussi toute sa puissance à la plateforme Raspberry-Pi Pico avec quelques caractéristiques remarquables :

- **264 Kio de mémoire RAM** : 132 fois plus qu'un Arduino UNO. Cette quantité de mémoire est très confortable pour exécuter tout script Python et bibliothèques.
- **Mémoire Flash externe** : le microcontrôleur est conçu pour être équipé d'une mémoire Flash externe (voir point suivant). De nombreuses plateformes à base de RP2040 utilisent cette particularité pour proposer une offre variée en matière de stockage.
- **Double cœurs** : permettant l'exécution en parallèle de scripts Python (module *_thread*) qui, au contraire de Python sur PC, offrent une vraie exécution multithread des deux processus. Il faut donc prévoir des mécanismes de synchronisation lors d'échanges de données entre les deux threads.
- **PIO** : Programmable IO est une fonctionnalité avancée du RP2040 permettant de créer des entrées/sorties autonomes (fonctionnant sans assistance des deux cœurs). PIO permet d'implémenter des protocoles de communication matériel à l'aide d'un ensemble de quelques instructions spécialisées. Cette fonctionnalité avancée ne sera pas abordée dans cet ouvrage.

- **Support natif USB** : supportant à la fois USB 1,1 client et hôte, le microcontrôleur peut être branché directement sur un ordinateur pour établir une communication USB-Série, offrir un service de stockage de masse (comme un lecteur USB Stick) et une émulation de clavier/souris ou autres périphériques HID (*Human Interface Device*).
- **Bus matériels** : des bus matériels SPI, I2C, UART comme sur tous les microcontrôleurs classiques mais en plusieurs exemplaires et accessibles via le Bus Fabric permettant de relocaliser ces bus sur différentes broches du Pico (voir description du Bus Fabric en début de chapitre - section Microcontrôleur RP20401). PIO permet, entre autres, de créer de nouveaux types de bus.
- **Port de débogage** : s'utilise en conjonction avec OpenOCD pour déboguer du code en C/C++.
- **Bouton Boot** : en cas de blocage complet du microcontrôleur, il est possible d'activer le mode **Boot** (presser le bouton en mettant la carte sous tension). Dans ce mode, la carte Pico apparaît comme un lecteur USB où il est possible de téléverser un nouveau firmware par simple glissé/déposé. Ce mode est exploité par les développeurs C/C++ pour placer une nouvelle version de leur programme (*firmware*) sur la carte Pico. Sous MicroPython, le firmware est « MicroPython » lui-même et ne doit être installé qu'une seule fois.

2.1.4 L'interface de débogage

Un peu à l'écart du brochage d'entrée/sortie, le port 3 broches portant la mention « *DEBUG* » sur la sérigraphie est composé d'un signal de données SWDIO (*SoftWare Debug Input/Output*) et d'un signal d'horloge SWDCLK (*SoftWare Debug CLock*).



Interface de débogage

Ce connecteur permet de déboguer des programmes écrits en C/C++ à l'aide d'OpenOCD (*Open On Chip Debugger*). La fondation recommande de souder un connecteur orienté vers le haut si cette interface doit être utilisée.