



Un module de microcontrôleur à ultra-faible puissance offre de nouvelles perspectives aux ingénieurs — Première partie : création d'un projet Eclipse

Simon Bramble, ingénieur en chef chez Analog Devices

Résumé

Cet article présente un module de microcontrôleur à ultra-faible puissance offrant de nombreuses fonctions, et il explique comment le programmer et le déboguer à l'aide d'outils gratuits courants. À la différence de beaucoup d'autres modules de microcontrôleurs haut de gamme, celui-ci est disponible dans un format DIP permettant aux ingénieurs professionnels ou amateurs d'effectuer des prototypages faciles. La première partie indique comment créer un projet dans Eclipse, tandis que la [deuxième partie](#) explique comment configurer Eclipse pour qu'il fonctionne avec le module de microcontrôleur.

Introduction

Alors que la quête constante de miniaturisation repousse les limites de l'électronique, les ingénieurs et amateurs éprouvent de plus en plus de difficultés à réaliser des prototypes avec des boîtiers dont la taille ne cesse de diminuer. Bien qu'il soit évident que l'introduction d'un microcontrôleur dans une conception multiplie les capacités du circuit, bon nombre d'ingénieurs restent nostalgiques de l'époque où davantage de composants étaient disponibles en boîtiers DIP. Les microcontrôleurs plus évolués ne sont disponibles que dans des boîtiers qu'il est impossible d'utiliser pour le prototypage ou qui sont fournis avec un énorme kit d'évaluation incluant de nombreux composants inutiles.

Cet article présente le [MAX32625PICO](#), un module de microcontrôleur adapté au prototypage et incroyablement petit, mais conservant toutefois un format DIP qui facilite à l'extrême son intégration dans des circuits imprimés expérimentaux et de production. L'article indique également comment programmer et déboguer à l'aide d'[Eclipse](#), un environnement de développement intégré (EDI) gratuit et très populaire. Il encouragera les ingénieurs à laisser derrière eux le monde des processeurs 8 bits tout en gardant un pied dans celui des boîtiers DIP. Pour obtenir un guide de démarrage complet, prenez connaissance des instructions qui suivent.

Découvrez le MAX32625PICO

La figure 1 montre le MAX32625PICO, également appelé PICO. Il contient le microcontrôleur [MAX32625](#), un processeur à ultra-faible puissance 32 bits Arm® Cortex®-M4 avec 512 Ko de mémoire flash et 160 Ko de SRAM, et qui fonctionne à une fréquence jusqu'à 96 MHz. Les périphériques les plus utiles du microcontrôleur, à savoir l'interface SPI principale, de l'interface SPI secondaire, un port I2C, deux entrées vers un ADC 10 bits, une interface monocâble et deux UART, sont mis à disposition sur les broches du PICO. Ce dernier comprend également le circuit de gestion de puissance [MAX14750](#), un LED RVB, un interrupteur, un connecteur USB Micro B et une embase de débogage Cortex à 10 broches, et il peut être alimenté directement via le port USB ou une alimentation 5 V externe. De plus, il est muni de connecteurs mâles de 0,1 pouce de chaque côté du circuit imprimé, ce qui permet de souder les broches sur le PICO et de le monter sur une plaque de prototypage. Si un montage en surface est nécessaire, le cuivre des connecteurs mâles s'étend jusqu'au bord de la carte, et le PICO peut donc être utilisé comme composant de montage en surface également. Le PICO est suffisamment petit pour être introduit dans n'importe quelle conception, mais suffisamment grand pour permettre un prototypage facile. Voir la figure 1.

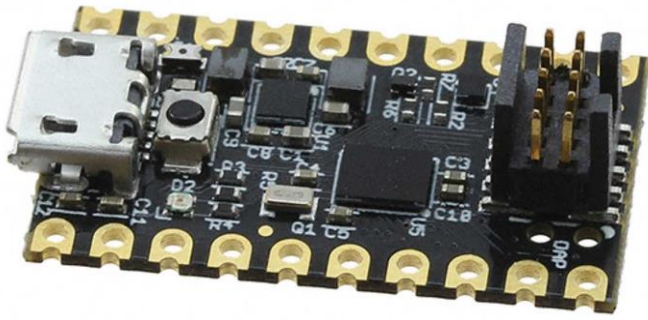


Figure 1. Le MAX32625PICO.

Les composants nécessaires pour programmer ou déboguer un PICO sont les suivants :

- Deux PICO
- Deux câbles USB Micro B (normalement fournis avec les PICO)
- Un câble de programmation, numéro de série TC2050-IDC-NL-050

Le câble TC2050 est muni d'un connecteur à 10 broches d'un côté et de broches pogo à ressort de l'autre côté. Il n'est utilisé que pendant le débogage ou la reprogrammation du PICO si le bootloader est écrasé. La programmation normale se fait par simple glisser-déposer.

Créer un projet dans l'EDI Eclipse

Le PICO a été conçu à l'origine pour être utilisé sur la plateforme de compilation en ligne Mbed, qui n'est plus disponible aujourd'hui. Très prisé par les professionnels comme les amateurs, l'EDI open-source Eclipse offre une alternative gratuite à la plateforme Mbed. Cet article indique au programmeur comment configurer Eclipse pour générer un fichier binaire qui peut être déposé sur le matériel cible à l'aide de Windows Explorer. Les fichiers de configuration peuvent être téléchargés dans un fichier .zip via le lien fourni à la fin du présent article.

L'EDI Eclipse peut être téléchargé depuis la page d'accueil du microcontrôleur MAX32625. Allez dans la section **Tools and Simulations** et cliquez sur le lien de téléchargement **Low Power ARM Micro SDK (Win)**. Vous pouvez alors télécharger et installer l'Arm Cortex Toolchain IDE. Utilisez la configuration d'installation par défaut.

Après l'installation, sélectionnez :

Fichier -> Nouveau -> Microcontrôleurs Maxim

comme sur la figure 2.

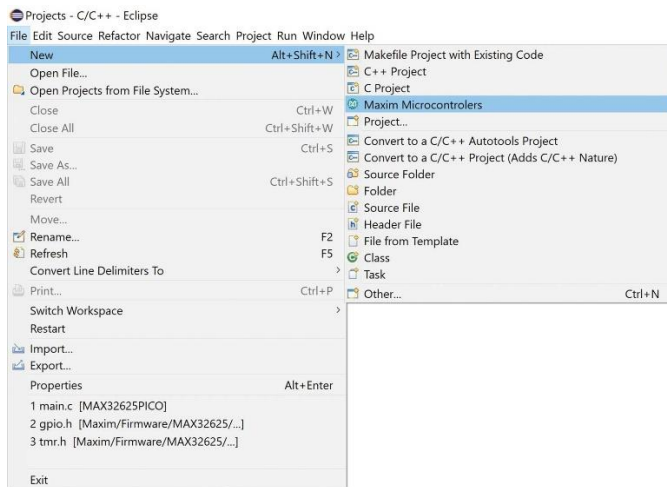


Figure 2. Créer le premier projet.

Sur la boîte **Créer un projet** qui apparaît, assurez-vous que le nom du projet ne contient aucun espace.

Remplissez la fenêtre **Sélectionnez la configuration du projet** comme sur la figure 3.

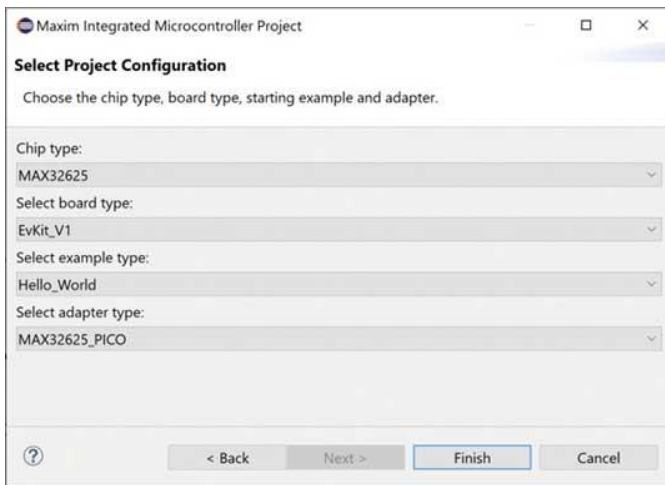


Figure 3. Configuration du projet.

Comment l'EDI communique-t-il avec le matériel cible ?

Il est intéressant de décrire ce qu'il se passe à l'intérieur d'Eclipse pour expliquer les messages d'erreur qui peuvent s'afficher pendant le processus de débogage. L'EDI communique avec le microcontrôleur cible (en l'occurrence, le PICO) à l'aide de deux programmes : le débogueur GNU (GDB) et le débogueur sur puce Open (OCD). GDB est un outil de débogage de haut niveau qui permet à l'utilisateur de préparer le code, d'exécuter les points de contrôle et d'examiner les valeurs de registre. GDB communique avec OpenOCD, un outil qui traduit les commandes de haut niveau pour qu'elles soient plus proches de ce que comprend le microcontrôleur cible. GDB et OpenOCD tournent tous deux sur le PC, dans Eclipse. Entre le PC et le matériel final est connecté un adaptateur de débogage qui traduit les commandes en signaux électriques compris par le port de débogage du microcontrôleur. Ce port peut avoir deux formes : un port de débogage JTAG ou une version JTAG comptant moins de broches, appelée port SWD (single wire debug). Tout ce qui précède permet au programmeur d'exécuter le code ligne par ligne sur le matériel cible plutôt que sur un simulateur, en observant les variations des valeurs de registre.

L'EDI Eclipse communique avec GDB, qui communique avec OpenOCD, et OpenOCD communique via l'adaptateur de débogage avec le port de débogage du PICO cible. GDB et OpenOCD se trouvent dans le kit de développement logiciel (SDK) et sont installés automatiquement avec ce dernier. Ils sont invoqués par Eclipse. Lorsqu'un adaptateur de débogage est connecté entre le PC et le matériel cible, la communication de bout en bout entre Eclipse et le PICO est transparente.

La configuration de projet présentée sur la figure 3 indique que l'EDI Eclipse sera connecté à un **MAX32625EVKIT** qui utilise un port JTAG. Toutefois, au lieu du débogueur JTAG, le PICO utilise le port SWD comprenant moins de broches. Pour assurer l'interface entre l'EDI Eclipse et le PICO, un deuxième PICO peut être connecté entre le PC hôte et le PICO cible, à la place du débogueur JTAG. Ce deuxième PICO sera appelé PICO programmeur, afin de le distinguer du PICO cible sur lequel tournera le code final. Le PICO programmeur doit être chargé avec un programme appelé DAPLink, qui traduit les signaux USB du PC en signaux compris par le PICO. Il s'agit d'une procédure simple décrite à la section Charger un fichier binaire, dans la deuxième partie du présent article.

Étant donné que l'on utilise ici un PICO programmeur au lieu du débogueur JTAG, sélectionnez le type d'adaptateur **MAX32625_PICO**. Voir la figure 4.



Figure 4. Sélectionner le type d'adaptateur.

Modifier Eclipse pour qu'il fonctionne avec le PICO.

Téléchargez le fichier .zip via le lien fourni à la fin de cet article et extrayez son contenu dans l'emplacement qui convient. Trouvez le fichier

MAX32625PICO.cfg

et copiez-le dans le répertoire suivant :

C:\Maxim\Toolchain\share\openocd\scripts\target

Ceci modifie le fichier original (MAX32625.cfg) et permet d'utiliser un deuxième PICO à la place du débogueur JTAG. Ce fichier écrase la commande de réinitialisation généralement émise par le débogueur JTAG et réinitialise certains des registres du processeur cible, ainsi que le compteur de programme et le pointeur de pile.

Une fois le projet créé, dans l'onglet **Explorateur de projets** d'Eclipse (en haut à gauche), cliquez avec le bouton droit sur le nom du projet et sélectionnez :

Déboguer en tant que -> Déboguer les configurations...

Allez dans **Débogage GDB OpenOCD** sur le menu de gauche, puis sélectionnez le projet en cours.

Dans l'onglet **Débogueur**, changez le fichier de configuration en remplaçant **MAX32625.cfg** par **MAX32625pico.cfg**, comme indiqué sur la figure 5.

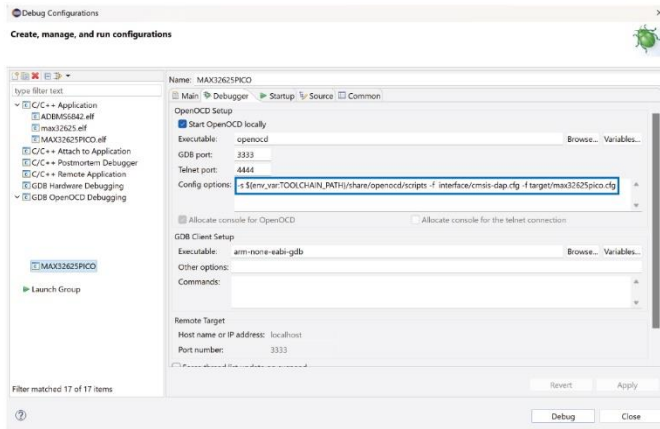


Figure 5. Sélectionner le nouveau fichier de configuration.

Cliquez sur **Appliquer**, puis **Fermer**.

Sur la fenêtre **Explorateur de projets**, cliquez de nouveau avec le bouton droit sur le nom du projet et sélectionnez **Propriétés**. Cliquez avec le bouton gauche sur l'en-tête **C/C++ Build** et, dans l'onglet **Paramètres builder**, assurez-vous que la commande **Build** est la suivante :

make ECLIPSE=1 release

comme sur la figure 6. Ceci indique à Eclipse de générer le fichier binaire (programme exécutable) qui peut être chargé sur le PICO.

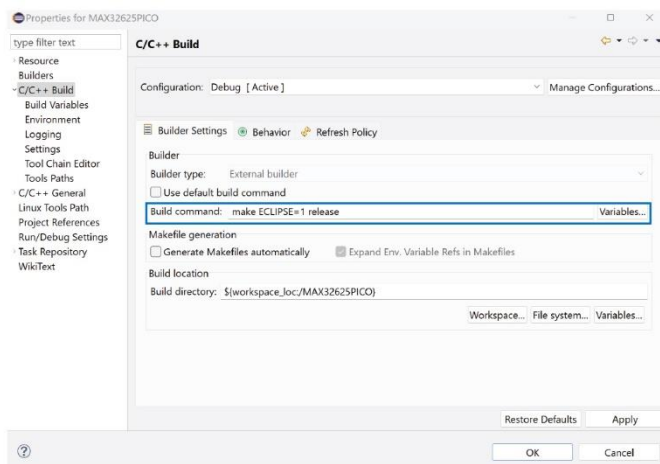


Figure 6. Générer le fichier binaire.

Cliquez sur **Appliquer** et **OK**.

Configurer Eclipse pour inclure d'autres fichiers

Pour faciliter la programmation des processeurs Arm, le microcontrôleur est fourni avec des pilotes de bas niveau pour permettre au code du programmeur de communiquer avec les nombreux registres et périphériques qui se trouvent dans le microcontrôleur. Ainsi, le programmeur n'a pas besoin de se soucier du contrôle du niveau de registre et peut se concentrer sur la fonctionnalité de niveau élevé de l'application finale. Ces pilotes répondent à la norme CMSIS (Common Microcontroller Software Interface Standard) et sont simplement une arborescence des fichiers nécessaires pour programmer le MAX32625. Ce microcontrôleur contient un cœur Arm qui est commun à chaque famille particulière de microcontrôleurs Arm. Il comprend un jeu de fichiers nécessaires pour le configurer. Autour du cœur Arm sont installés des périphériques (ADC, ports GPIO, minuteurs, compteurs, ports SPI, etc.), ce qui distingue ce produit des autres produits basés sur Arm. La configuration de ces périphériques requiert un jeu de fichiers. Le cœur Arm et ses périphériques constituent le microcontrôleur. Celui-ci est monté sur une carte et connecté par exemple à un écran, des port-headers, des interrupteurs, des LED et des émetteurs-récepteurs Bluetooth®. Une série de fichiers décrit comment il est connecté aux composants voisins présents sur la carte.

Depuis le cœur du MAX32625 (cœur Arm) vers l'extérieur, un jeu de fichiers permet de configurer le cœur Arm proprement dit, puis un autre jeu de fichiers permet de configurer les périphériques qui se trouvent autour du cœur Arm, et enfin un dernier jeu de

fichiers permet de configurer les composants du kit d'évaluation qui entourent le processeur. Eclipse doit être configuré pour trouver ces fichiers lors de la compilation du programme.

Dans Eclipse, sur la fenêtre **Explorateur de projets**, cliquez avec le bouton droit sur le nom du projet et sélectionnez **Propriétés**, puis agrandissez le menu **Général C/C++** et allez dans la section **Chemins et symboles**. Sur l'onglet **Inclut**, sélectionnez **GNU C**. Ajoutez les répertoires indiqués sur la figure 7 à l'aide des boutons **Ajouter** et **Système de fichiers**.

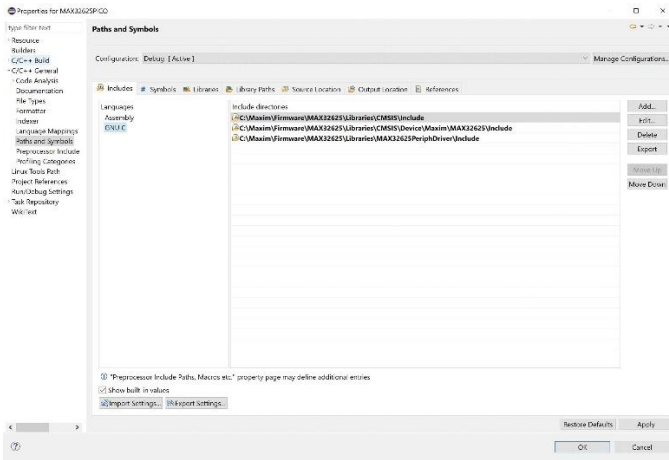


Figure 7. Ajouter des fichiers externes.

Après avoir ajouté les fichiers, cliquez sur **Appliquer** et **OK**.

Vous avez ci-après une description de la fonction de chaque répertoire inclus.

C:\Maxim\Firmware\MAX32625\Libraries\CMSIS\Include

Ce répertoire contient les fichiers utilisés par le cœur Arm du MAX32625.

C:\Maxim\Firmware\MAX32625\Libraries\CMSIS\Device\Maxim\MAX32625\Include

Ce répertoire contient les définitions des périphériques/registres du MAX32625.

C:\Maxim\Firmware\MAX32625\Libraries\MAX32625PeriphDriver\Include

Ce répertoire contient les fichiers permettant d'utiliser les périphériques du MAX32625.

Conclusion

La première partie du présent article décrit la création d'un projet dans Eclipse et le logiciel nécessaire pour programmer un microcontrôleur. La deuxième partie indique comment configurer Eclipse pour qu'il fonctionne avec le PICO.

Téléchargez les fichiers du logiciel via le lien suivant : <https://www.analog.com/media/en/software/software-configuration/eclipse-configuration.zip>

À propos de l'auteur

Simon Bramble est diplômé en génie électrique et électronique, avec spécialisation électronique analogique et puissance, de la Brunel University de Londres (1991). Il a fait carrière dans le domaine de l'électronique analogique et travaillé chez Linear Technology (qui fait maintenant partie d'Analog Devices).