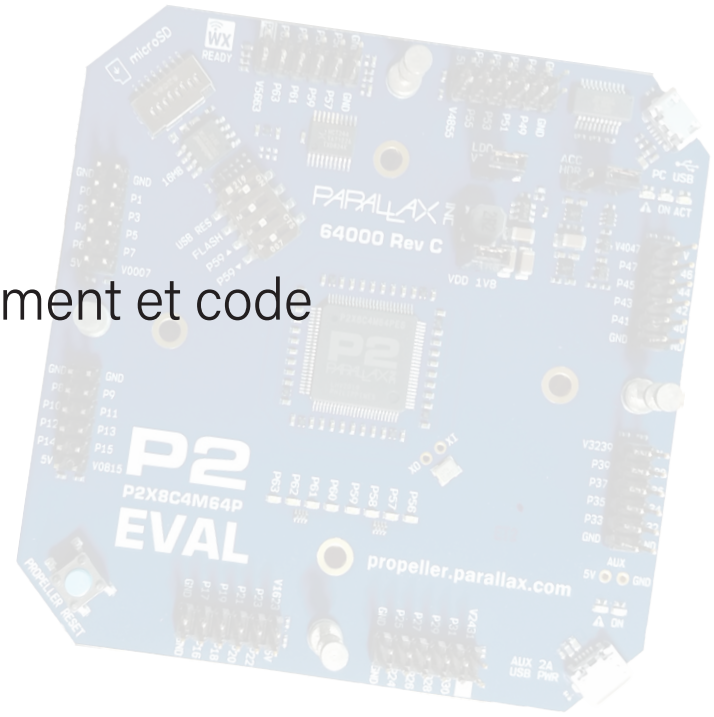
Propeller 2 de Parallax (2)

Environnement de développement et code



Mathias Claußen (Elektor)

Dans la première partie de cette série d'articles, je vous ai présenté le Propeller 2, le nouveau microcontrôleur multicœur avancé de Parallax. Voyons maintenant comment utiliser le langage Spin2 pour piloter une LED.

Maintenant que le Propeller 2 de Parallax et ses caractéristiques vous sont familiers, je vais vous faire découvrir l'environnement de développement et l'écriture du code. Lisez la suite pour apprendre à piloter une LED.

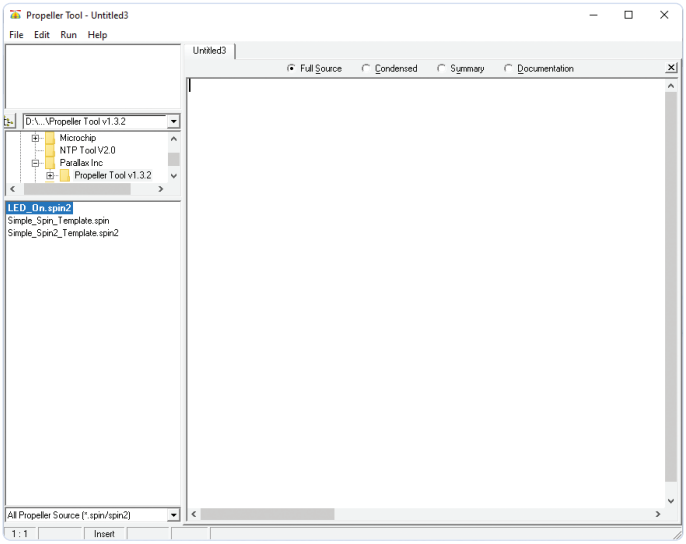


Figure 1. Interface utilisateur de Propeller Tool.

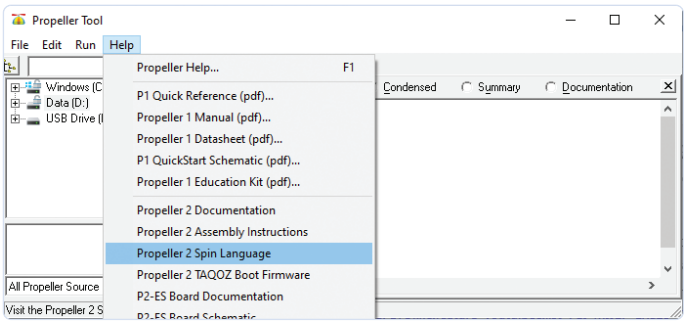


Figure 2. Accès à la documentation Spin2 via le menu Aide (Help).

Accéder aux LED intégrées

Commençons par l'environnement logiciel et la commande de l'une des LED intégrées. Nous ferons notre développement sous Windows 10 avec les outils fournis par le constructeur. Nous mettrons en place la configuration logicielle minimale pour compiler et charger le code sur le Propeller 2.

Parallax a inclus les langages Spin/Spin2 pour le Propeller 2 dans son outil de développement Propeller Tool Version 2.3 Alpha. Si vous préférez le langage assembleur, vous pouvez utiliser PNut comme environnement de développement ou un assembleur en ligne (*inline*). Si vous aimez coder en C/C++ comme moi, malheureusement, le compilateur et l'environnement de développement ne sont pas encore prêts pour ces langages. Actuellement, les équipes de développement s'efforcent de travailler sur le compilateur C/C++ et la vidéo de présentation [1] du Propeller 2 montre où ils en sont.

Environnement de développement

Vous pouvez télécharger le logiciel Propeller Tool 2.3 Alpha sur [2]. Téléchargez d'abord Propeller Tool 1.3.2 et placez l'exécutable de Propeller Tool 2.3 Alpha dans son répertoire d'installation. Une fois le processus terminé, vous pouvez démarrer l'éditeur et commencer à coder. L'interface utilisateur de l'outil est illustrée à la **figure 1.**

Assembleur ou Spin2 ?

La question est : Spin2 ou Assembler pour commencer l'écriture du code ? Pour rester simples, nous continuerons, pour l'instant, avec Spin2. Spin2 est un langage interprété, comme le BASIC, mais différent. Vous trouverez dans le menu d'aide (*Help*) les commandes et les références du langage Spin2 (**figure 2)**.

Pour démarrer avec le langage Spin2, vous pouvez utiliser le manuel de référence intégré, mais gardez à l'esprit qu'il n'est pas encore terminé.

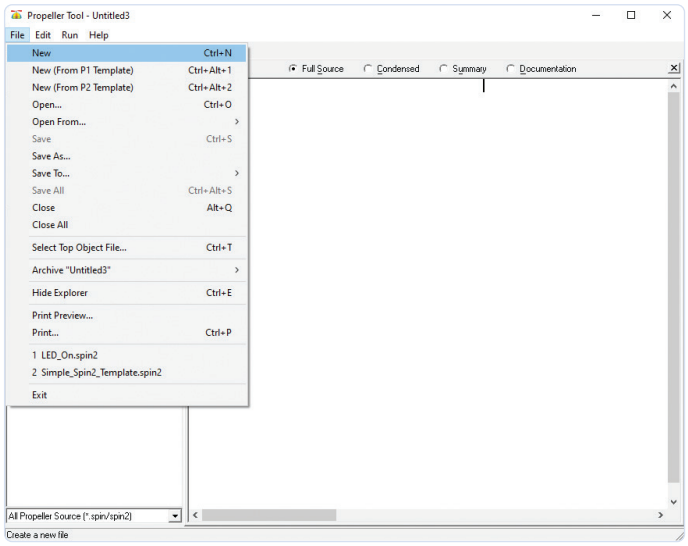


Figure 3. Création d'un nouveau projet Spin2.

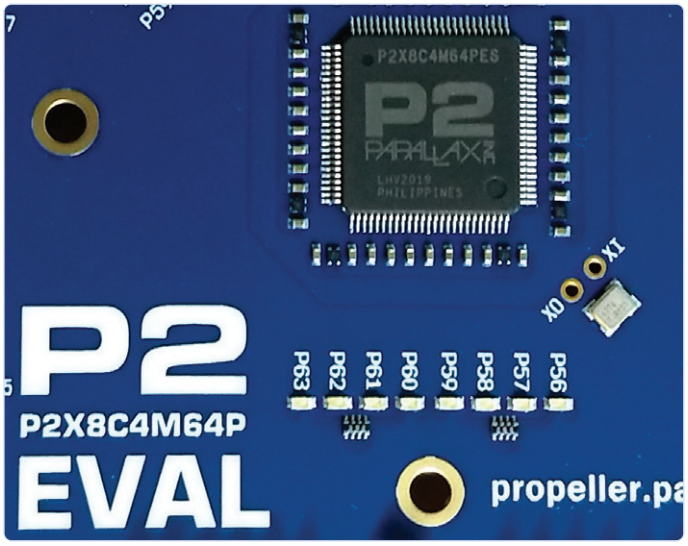


Figure 4. Détail des LED connectées aux broches 56 à 63 du microcontrôleur.

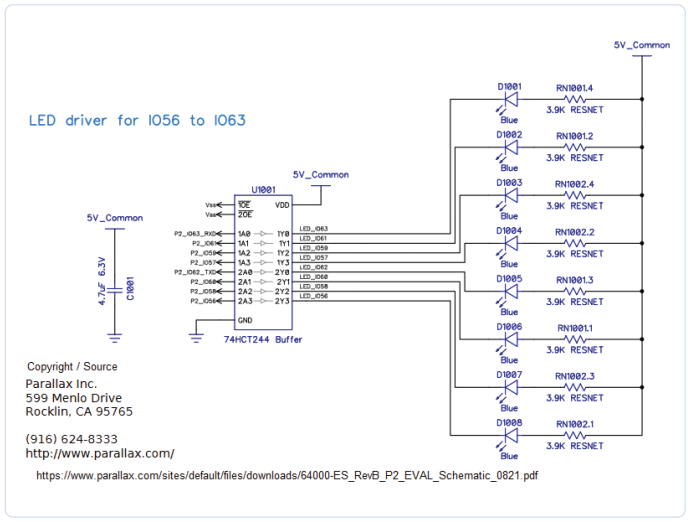


Figure 5. Schéma des LED pilotées par les tampons du 74HCT244.

Cela vous mènera à la documentation officielle des fonctions mises en place dans le microcontrôleur Propeller 2.
Comme ce langage est de type interprété, vous aurez besoin d'environ six cycles d'instructions, soit douze cycles d'horloge, pour exécuter une commande. Ce n'est pas le moyen le plus rapide de faire avancer les choses en matière de cycles d'horloge, mais le langage Spin2 est plus facile à prendre en main que l'assembleur. Comme le Spin2 est un langage de haut niveau, la plupart des commandes cachent beaucoup d'instructions en assembleur, mais elles sont plus intuitives. Vous pouvez créer un nouveau projet Spin2 en suivant la **figure 3**.

Broches d'entrées-sorties (E/S)

Avant de commencer à coder, examinons brièvement le fonctionnement des broches d'E/S. Nous les utilisons ici comme des broches d'E/S ordinaires et verrons plus tard toutes leurs fonctions en détail. En général, nous devons définir la broche choisie comme sortie pour pouvoir la placer à un niveau logique bas ou haut. Sur la carte d'évaluation, vous pouvez apercevoir clairement les LED intégrées, celles-ci sont marquées P56 à P63 (**figure 4**). Il s'agit d'un groupe de LED connectées aux broches 56 à 63 du Propeller 2, nous choisissons la première (en face du marquage P56) pour nos expériences. Si vous observez attentivement le schéma, vous remarquerez qu'elles ne sont pas directement reliées au microcontrôleur, mais qu'elles passent par un circuit intégré 74HCT244, contenant huit tampons TTL. Notez également que l'anode de chaque LED est connectée de manière permanente à la tension VCC. La cathode de chaque LED est reliée à une broche d'E/S du microcontrôleur, elle est ramenée à la masse (0 V) grâce au tampon, ce qui permet d'allumer la LED correspondante (**figure 5**).

LED inversées

Pour notre code, cela signifie que placer la broche du microcontrôleur à un niveau logique haut éteindra la LED alors qu'un niveau bas l'allumera. Avec cette logique inversée en tête, nous devons dire à notre code de placer la broche P56 à un niveau haut pour l'éteindre. De plus, si nous ne configurons rien, les broches sont configurées par défaut en entrées, les tampons agiront comme si elles étaient placées à un niveau logique haut et chaque LED connectée sera éteinte. En résumé, pour allumer notre LED avec le langage Spin2, nous devons :

- Initialiser le microcontrôleur et au moins un cœur (*cog* en anglais)
- Configurer la broche 56 en tant que sortie
- Placer la broche 56 à un niveau logique bas
- Ne plus rien faire

Première étape : l'initialisation. Dans le cas présent, elle est déjà effectuée pour nous et nous n'avons pas besoin de nous en soucier. Notre exemple en langage Spin2 n'a besoin que de quelques lignes. Notre code commencera par la fonction `pub main ()` suivie de la méthode `pinwrite()` (**figure 6**). La méthode `pinwrite()` est intégrée au langage Spin2 et permet de définir le niveau logique d'une ou plusieurs broches grâce à une valeur donnée. Ici, nous utilisons uniquement la broche 56, à laquelle notre LED est connectée. Nous passons en deuxième argument un '0' pour placer la broche à un niveau logique bas et ainsi, allumer notre LED. La dernière ligne `repeat` forcera le processus dans une boucle sans fin, car il n'y a plus de code sous l'instruction `repeat` l'obligeant à ne rien faire d'autre qu'à se répéter indéfiniment. Sans l'instruction `repeat` ici, le processus s'arrêterait après une itération, les broches d'E/S ne seraient plus pilotées.

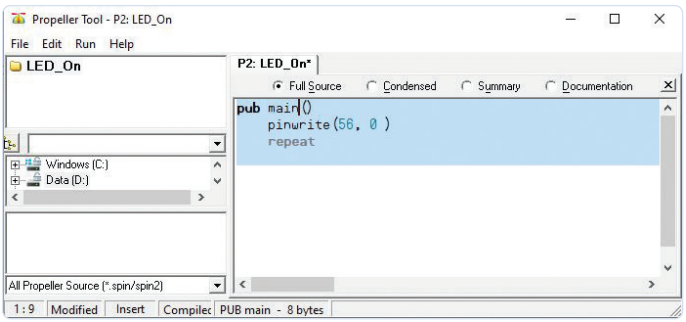


Figure 6. Code complet pour allumer la LED de la broche 56.

Chargement du code dans le Propeller 2

Si votre code est prêt, vous pouvez le charger sur la carte d'évaluation du Propeller 2 et l'exécuter. Pour cela, il suffit de connecter la carte à votre PC par l'un de ses ports USB et de sélectionner dans le menu *Run->Compile Current->Load RAM* pour charger le code directement dans la RAM du microcontrôleur et l'exécuter. La LED connectée à la broche 56 doit s'allumer, bravo, vous avez réussi une sorte de « Bonjour tout le monde ! » (*Hello, world!*). Maintenant vous vous demandez : « Et je peux la faire clignoter ? ». Oui, c'est possible. Le langage Spin2 contient l'instruction `WAITMS` qui permet de retarder l'exécution du code, par exemple `WAITMS(500)` provoque un retard de 500 ms. Vous savez que le code après `REPEAT` est exécuté en boucle. Maintenant vous devriez pouvoir adapter le code pour que la LED clignote. Utilisez les informations que vous avez rassemblées jusqu'à présent pour modifier le code afin d'obtenir une LED clignotante. Ne vous inquiétez pas, nous donnerons la solution plus tard.

Nous pouvons maintenant piloter une broche d'E/S, l'étape suivante consiste à déterminer comment envoyer des données série. La plupart d'entre nous sont habitués à cet apprentissage progressif, en particulier lors des premiers pas avec un microcontrôleur, comme un AVR. Comme nous aurons besoin des fonctions Smart-Pin, nous nous familiariserons également avec celles-ci.

200479-B-03

Des questions ou des commentaires ?
Vous avez des questions ou des commentaires sur cet article. Envoyez un courriel à l'auteur (mathias.claussen@elektor.com) ou contactez Elektor (redaction@elektor.fr).

Contributeurs
Auteur : **Mathias Claußen**
Rédacteurs : **Jens Nickel** et **CJ Abate**
Mise en page : **Giel Dols**
Traduction : **Nicolas Bishop**

LIENS

[1] **Parallax, « Propeller 2 Live Forum Early Adopter Series - C Programming with Eric Smith », 6 juillet 2020 :** https://bit.ly/propeller2-c

[2] **Téléchargement de Propeller Tool 2.3 Alpha :** https://propeller.parallax.com/p2.html#software