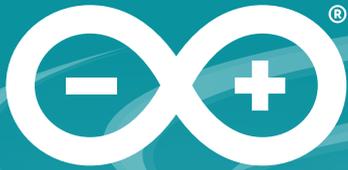


Édition spéciale
Rédaction invitée



ARDUINO

Édition bonus
dévoilée !

Domotique

Connectivité simplifiée

p. 4 **Rétro-gaming**
avec Arduino

p. 14 **Contrôleur**
pour Spotify

Du prototypage à la production

De nouveaux
articles bonus
gratuits chaque
semaine !



La carte d'apprentissage
Arduino d'Elektor

p. 10



Apporter l'aspect humain
au service de la production

p. 24



Disponible
la semaine 01
2023

Seuls les articles qui
répondent à nos critères

p. 32

De la part d'Arduino et d'Elektor par Amour de l'Ingénierie

En vente maintenant

Édition spéciale du magazine Elektor !
Rédaction invitée : Arduino !

Des projets électroniques

« maison », des idées pour les électroniciens, et plus encore, proposés par les experts d'Arduino et d'Elektor

De nombreux projets et tutoriels

Explorez des sujets d'actualité : **MicroPython**, **TinyML** et domotique avec Arduino

Découvrez l'Arduino : Les points de vue de **Fabio**, **Massimo** et **David**

Découvrez la **Portenta X8**

Achetez-le dès maintenant, dans votre **kiosque préféré** ou dans l'e-choppe d'Elektor.

Les liens dans les articles vous offrent un **accès facile** aux produits et solutions Arduino.



En savoir plus

www.elektor.fr/arduino-magazine



45^{ème} année
Édition bonus dévoilée

ISSN 0181-7450
Dépôt légal : décembre 2022
CPPAP 1125 T 83713
Directeur de la publication :
Donatus Akkermans

Elektor est édité par :
PUBLITRONIC SARL
c/o Regus Roissy CDG
1, rue de la Haye
BP 12910
FR - 95731 Roissy CDG Cedex

Pour toutes vos questions :
service@elektor.fr

www.elektor.fr | www.elektormagazine.fr

Publicité :
Raoul Morreau
Tél. : +31 (0)6 4403 9907
Courriel : raoul.morreau@elektor.com

DROITS D'AUTEUR :
© 2022 Elektor International Media B.V.

Elektor ne publie que du contenu (texte et images) produit par lui-même ou avec l'autorisation de son créateur. Avant publication, les droits d'auteur du contenu fourni par des tiers sont vérifiés. Si l'ayant droit est inconnu, Elektor s'efforce de le retrouver pour le rémunérer. Il n'est pas toujours possible de retrouver le détenteur des droits d'auteur. Si vous êtes ou connaissez le titulaire inconnu des droits d'auteur d'une publication, veuillez nous contacter à l'adresse redaction@elektor.fr.

Toute reproduction ou représentation intégrale ou partielle, par quelque procédé que ce soit, des pages publiées dans la présente publication, faite sans l'autorisation de l'éditeur est illicite et constitue une contrefaçon. Seules sont autorisées, d'une part, les reproductions strictement réservées à l'usage privé du copiste et non destinées à une utilisation collective, et, d'autre part, les analyses et courtes citations justifiées par le caractère scientifique ou d'information de l'oeuvre dans laquelle elles sont incorporées (Loi du 11 mars 1957 -art. 40 et 41 et Code Pénal art. 425).

Certains circuits, dispositifs, composants, etc. décrits dans cette revue peuvent bénéficier de droits propres aux brevets; la Société éditrice n'accepte aucune responsabilité du fait de l'absence de mention à ce sujet. Conformément à l'art. 30 de la Loi sur les Brevets, les circuits et schémas publiés dans Elektor ne peuvent être réalisés que dans des buts privés ou scientifiques et non commerciaux. L'utilisation des schémas n'implique aucune responsabilité de la part de la Société éditrice. La Société éditrice n'est pas tenue de renvoyer des articles qui lui parviennent sans demande de sa part et qu'elle n'accepte pas pour publication. Si la Société éditrice accepte pour publication un article qui lui est envoyé, elle est en droit de l'amender et/ou de le faire amender à ses frais; la Société éditrice est de même en droit de traduire et/ou de faire traduire un article et de l'utiliser pour ses autres éditions et activités, contre la rémunération en usage chez elle.

Imprimé aux Pays-Bas par
Senefelder Misset - Doetinchem
Distribué en France par M.L.P. et en Belgique par A.M.P.

Édition bonus dévoilée

L'édition du magazine *Elektor*, publiée début décembre 2022 avec Arduino en tant que rédaction invitée, n'était pas la fin de la collaboration créative entre Elektor et Arduino. Nous avons encore plus de projets, des idées et des articles informatifs pour vous inspirer pendant les mois à venir. Sur une période de quatre semaines, nous dévoilons le contenu de cette édition jusqu'à ce que vous disposiez du magazine bonus complet début janvier 2023. Quelle belle façon de démarrer la nouvelle année ! Que vous soyez un ingénieur professionnel travaillant sur un nouveau produit industriel ou un bricoleur à la recherche d'un projet amusant basé sur

Arduino pour le week-end, vous trouverez cette édition supplémentaire du magazine Elektor informative et motivante. Nous vous proposons des articles sur un large éventail de sujets et de projets liés à Arduino, notamment le rétrogaming avec Arduino, une carte d'apprentissage Arduino d'Elektor et un contrôleur portable basé sur Arduino pour Spotify. En lisant les projets et les articles de ce numéro, n'hésitez pas à partager vos idées avec nous sur elektormagazine.fr, arduino.cc et sur les réseaux sociaux. Nous avons hâte de recevoir vos commentaires. Bonne lecture !

C. J. Abate (Directeur du contenu, Elektor)

DANS CE NUMÉRO



- ◀ **4 Doom sur Portenta**
Rétro-gaming avec Arduino

- 10 Introduction rapide au monde de l'Arduino**
Carte de développement pour l'Arduino Nano

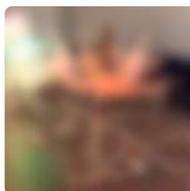


- ◀ **14 Contrôleur pour Spotify**
Le kit Oplà IoT contient (presque) tout ce dont vous avez besoin

- 20 Créez, déployez et maintenez des applications évolutives et sécurisées**
avec Arduino Portenta X8 équipé du mini processeur d'applications i.MX 8M de NXP et de l'élément de sécurité EdgeLock SE050



- ◀ **24 Apporter l'aspect humain au service de la production**
Rencontre avec Daria Baradel, la responsable de la production chez Arduino

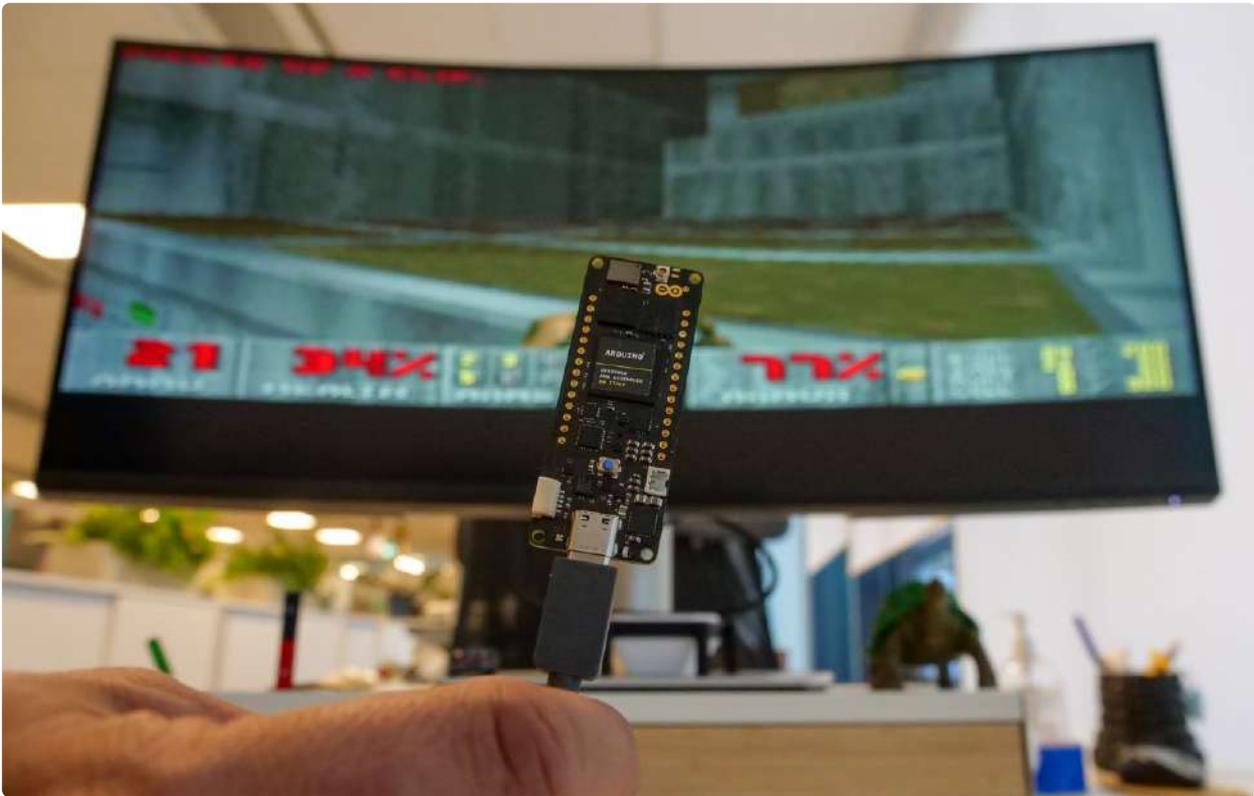


- 28 Les cartes de développement**
Passé, présent et futur

- ◀ **32 Art f... [redacted] ne**
Sc... [redacted] 2023

- ◀ **38 Ca... [redacted] s**
Mette... [redacted] no ! 2023





Vous pouvez faire tourner Doom sur une carte Portenta !

Doom sur Portenta

rétro-gaming avec Arduino

David Cuartielles (Arduino)

Saviez-vous que *Doom* pouvait être exécuté sur une carte *Portenta H7* ? Comment et pourquoi ce jeu ? Martino Facchin, chef de l'équipe en charge des micrologiciels d'Arduino, satisfait notre curiosité.

Sorti en 1993, *Doom* est probablement le jeu le plus populaire de l'histoire du jeu vidéo. Avec plus de 3,5 millions de titres vendus 50 \$ l'unité, ses développeurs sont devenus millionnaires du jour au lendemain. *Doom* est une création d'*id Software*, à l'époque un petit éditeur de jeux vidéo connu pour avoir publié l'année précédente le titre à succès *Wolfenstein 3D*. *Doom* est un jeu de tir en vue subjective,

un genre de jeu dans lequel le joueur combat des ennemis au moyen d'armes qu'il peut aussi se procurer sur le champ de bataille. *Doom* a été porté sur tous les systèmes d'exploitation, mais peut aussi être exécuté sur du matériel dépourvu d'OS. Le code est ouvert, actuellement distribué sous licence GPL. Sa compilation n'est pas triviale, mais certains sont parvenus à l'exécuter sur de très petits ordinateurs et au sein d'autres programmes. Microsoft lui a rendu hommage sous la forme d'un « œuf de Pâques » caché dans la version Excel 95.

Certains utilisent le jeu pour mesurer les performances de petits ordinateurs ou faire étalage de leurs talents de codeur. Lors du DEF CON 22 de Las Vegas, le codeur @sickcodes (son pseudo sur Twitter et GitHub) a ainsi lancé sur l'afficheur d'un tracteur John Deere une version de *Doom* comportant des scènes agricoles. Arduino ne fait pas exception. Nous aussi avons utilisé *Doom* pour tester les capacités du premier prototype de la *Portenta H7* – la plus puissante de nos cartes en 2018. J'ai invité Martino Facchin, responsable de l'équipe micrologiciel d'Arduino, à nous en dire plus sur cet épisode.



David Cuartielles : Parlons du rapport entre Doom et la carte Portenta H7. J'ai déjà esquissé à grands traits l'histoire du jeu et parlé de son succès phénoménal, un succès qui a rendu ses concepteurs millionnaires.

Martino Facchin : Des concepteurs qui par la suite ont publié le code source – c'est le point le plus important.

Cuartielles : Absolument ! À ce propos, sous quelle licence avaient-ils publié le code ?

Facchin : C'est à vérifier, mais je crois qu'il s'agit d'une licence compatible avec la GPL, la *Doom Source Code License*. Seul le moteur de jeu est open source, les éléments graphiques ne le sont pas. De fait on ne peut jouer qu'à la version *shareware* de Doom, pas au jeu complet. À moins de l'acheter bien sûr.

Note de David

J'ai vérifié, et effectivement *id Software* a publié Doom en 1997 sous la licence mentionnée par Martino – une ouverture du code à des fins éducatives. C'est après un incident de disque dur survenu chez les mainteneurs de *glDoom* – qui laissa le monde sans copie de ce portage openGL en raison de la clause de non-distribution de la licence Doom – qu'*id Software* accepta d'adopter pour sa licence les termes de la licence GPL.

Cuartielles : Le truc cool avec ce jeu, c'est que les joueurs pouvaient créer leurs propres « mods ». Je me rappelle d'une version de Wolfenstein 3D avec des personnages de Star Wars.

Facchin : Ça ne me dit rien.

Cuartielles : Eh oui, je suis vieux à ce point. Je me souviens pour nous deux.

Facchin : Ma foi, je me souviens bien avoir joué à Wolfenstein 3D quand j'étais enfant, mais comme nous n'avions pas de connexion internet, nous ne pouvions sans doute pas récupérer ce genre de mods.

Cuartielles : Avant d'en venir à Doom, pouvez-vous vous présenter ?

Facchin : [Rires] Martino Facchin, ingénieur micrologiciel chez Arduino.

Cuartielles : Quelle est votre fonction ?

Facchin : Je suis responsable des micrologiciels – le type qu'on appelle quand on a besoin d'aide sur un micrologiciel. J'étais seul au départ, mais aujourd'hui je suis entouré d'une équipe formidable, qui d'ailleurs s'agrandit encore. Nous essayons aussi d'y inclure la communauté en expliquant ce que nous faisons.

Cuartielles : À part le portage de Doom sur une Portenta (dont nous reparlerons), de quelle réalisation êtes-vous le plus fier ?

Facchin : Je dirais que c'est le *framework PluggableUSB*. Six mois après mon arrivée chez Arduino, nous avons dû répondre aux demandes des utilisateurs souhaitant exploiter le port USB de la carte *UNO Due* et celui de la toute nouvelle *Arduino Zero*. Chaque fois qu'un utilisateur reliait une carte *Leonardo* à son ordinateur, les pilotes de clavier, souris, etc., étaient activés, même s'ils n'étaient pas utilisés. Nous avons dû répondre à la hâte en écrivant un descripteur USB aidant l'utilisateur à voir uniquement ce qu'il voulait utiliser à ce moment-là. Nous avons également permis d'autres fonctions, par exemple l'USB MIDI réclamé par beaucoup d'utilisateurs. Pour moi c'était quelque chose d'énorme. J'étais assez jeune à l'époque, et pour trouver la meilleure stratégie j'avais dû m'adresser à la communauté avec l'aide de Matthijs Kooijman (www.stderr.nl pour en savoir plus sur son travail) et de Paul Stoffregen (créateur de Teensy). Et ça a marché. De temps en temps des personnes nous disent : « J'ai utilisé la bibliothèque MIDI pour ceci ou cela ». Je connais même un développeur qui exploite les fondations de ce code pour paramétrer les dispositions des claviers internationaux. Je suis très fier de cela.

Cuartielles : Vous dites que vous étiez « assez jeune à l'époque ». Depuis combien de temps travaillez-vous chez Arduino ?

Facchin : Depuis six ans et demi. Dans les bureaux de Turin.

Cuartielles : Une belle ancienneté déjà. Combien de personnes compte l'équipe Firmware ?

Facchin : Six personnes. Ça peut sembler beaucoup, mais nous nous occupons de tous les produits et avons d'autres activités, comme les certifications. À Turin travaille aussi l'équipe *Tooling*. Elle s'occupe de la CLI, l'interface en ligne de commande, et d'autres fonctions logicielles.



Doom s'est imposé naturellement, restait à implanter les fonctions nécessaires à son exécution.



Cuartielles : Tous les développeurs utilisent GitHub, n'est-ce pas ?

Facchin : Oui, et le code devient public une fois le développement achevé.

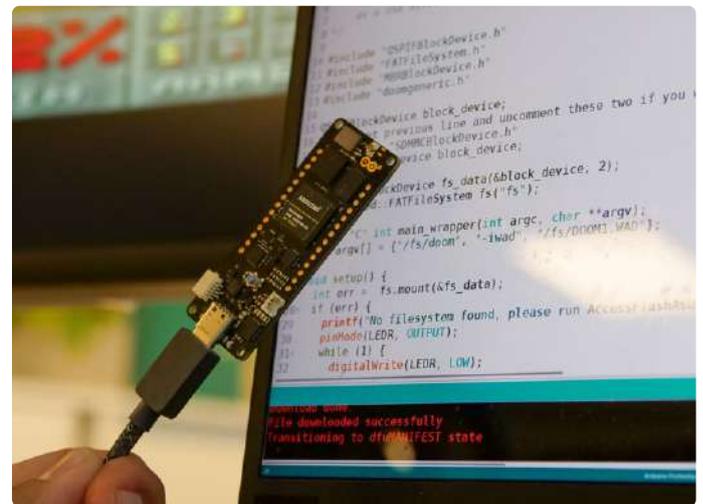
Cuartielles : Ce qui est super. Revenons à Doom. J'ai rappelé qu'il avait été vendu à plus de 3,5 millions d'unités et que ce succès, le premier dans son genre pour un jeu de tir à la première personne, avait rendu ses créateurs pleins aux as. Nous savons aussi que le code a été rendu public et porté sur toutes sortes d'appareils.

Facchin : Téléphones, calculatrices, systèmes d'exploitation...

Cuartielles : Et sur la Portenta H7, donc. Il s'agit d'une carte Arduino à double cœur conçue pour les environnements industriels. Qui a eu l'idée d'y faire tourner Doom ?

Facchin : C'est juste que nous venions d'en recevoir le premier prototype, une belle carte enrichie de nombreuses puces, et que tout restait à faire. Il y avait en particulier cette puce d'Analogix généralement utilisée pour convertir les signaux MIPI en signaux Display-Port et les diriger ensuite vers un moniteur externe. Nous n'avions aucune expérience avec les sous-systèmes de la puce chargés de cette conversion, et les exemples fournis ne nous aidaient pas. Nous sommes d'abord parvenus à afficher quelques boîtes jaunes, puis le logo Arduino entouré de formes graphiques, mais c'était loin d'être parfait. Nous ne comprenions pas pourquoi le rendu était mauvais, alors j'ai décidé d'essayer avec des graphismes et animations dont la qualité m'était connue.

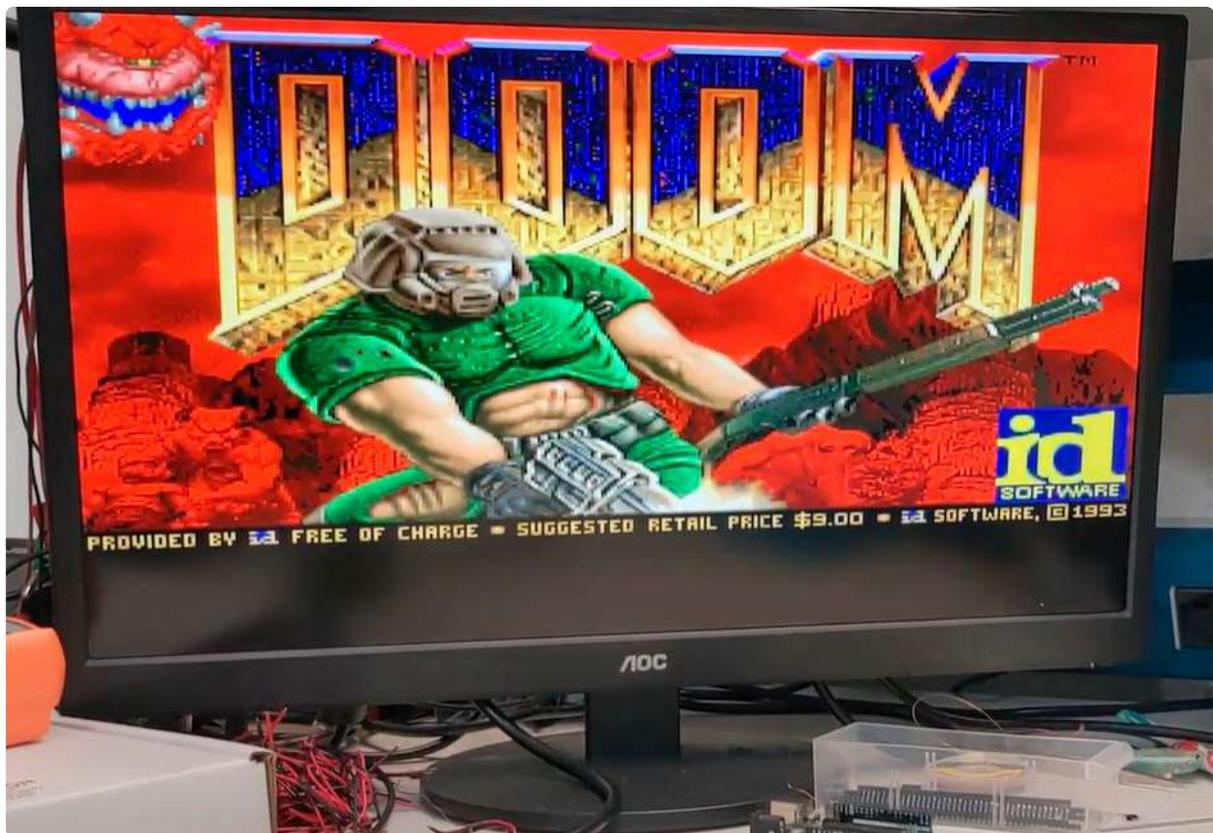
Doom s'est imposé naturellement, restait à implanter les fonctions nécessaires à son exécution. Le faire tourner sur un système sans OS demande relativement peu de travail, même si ce n'est pas vraiment la version de Doom avec laquelle vous passeriez des heures à jouer. Il suffit de modifier six ou sept fonctions pour l'adapter à votre matériel. Avant de lancer le simulateur, j'ai d'abord dû m'occuper de l'exécution des données en mémoire RAM ou externe, car nous n'avions aucun code de départ pour ça. Ensuite j'ai dû préparer un tampon de trame et procéder à un peu de magie afin d'obtenir, enfin, une sortie vidéo correcte par USB-C.



Pour jouer à Doom avec un périphérique de type manette, il vous faudra ajouter vous-même le code nécessaire.

Comment lancer Doom sur votre Portenta H7

1. Téléchargez la dernière version de l'EDI Arduino. Nous recommandons la version 2.0 ou ultérieure.
2. Téléchargez le cœur Portenta depuis le gestionnaire de cartes.
3. Sélectionnez le cœur M7 pour votre carte Portenta H7. Toutes les étapes suivantes doivent être exécutées sur ce cœur.
4. Assurez-vous que l'EDI a identifié le port auquel est relié la carte.
5. Vous trouverez sous Exemples / Doom un exemple contenant les instructions de base. Quelques exemples doivent d'abord être exécutés avant de procéder à l'installation de Doom.
6. [Optionnel] Lancez Exemples / STM32H747_System / STM32H747_manageBootloader pour mettre à jour le chargeur de démarrage de votre carte.
7. Formatez la mémoire flash externe avec Exemples / STM32H747_System / QSPIFormat. Après installation, ouvrez le terminal série et suivez les instructions données.
8. Lancez Exemples / USB As Mass Storage / AccessFlashAsUSBdisk pour transformer votre carte en périphérique de stockage USB.
9. Ouvrez le moniteur série et choisissez la façon dont la carte doit être formatée. Votre OS devrait ensuite reconnaître deux nouveaux lecteurs reliés à votre ordinateur.
10. Téléchargez DOOM1.WAD depuis doomwiki.org/wiki/DOOM1.WAD et copiez le fichier dans la partition la plus large du lecteur virtuel Portenta.
11. Retournez à l'exemple Doom.ino de l'étape 5 et flashez-le sur la carte (sur son cœur M7 !) Si vous n'aviez pas réussi à identifier le port de programmation, double-cliquez sur le bouton Reset de la carte avant de lancer le chargement du code, et assurez-vous que le port série est bien reconnu.
12. Déconnectez votre Portenta H7 et reliez-la à un hub USB-C comme s'il s'agissait d'un portable. Reliez votre hub à une alimentation externe, et utilisez un câble HDMI pour envoyer le signal vidéo à un écran.



Doom fait partie des classiques du jeu vidéo !

Cuartielles : Le processeur de la carte Portenta H7 possède deux cœurs Cortex M4 et M7. Lequel exécutait Doom ?

Facchin : À l'époque c'était le M4 parce qu'il était plus simple à mes yeux de programmeur de systèmes embarqués, mais aujourd'hui nous utilisons le M7. C'est une sorte de microcontrôleur type, sans caractéristiques spéciales. D'un autre côté le M7 possède un cache qu'il faut inspecter et invalider au bon moment lors de l'affichage des images. Il s'est montré très rapide la première fois que je l'ai utilisé, mais rien ne s'affichait correctement. A contrario la cadence qu'offrait le M4 suffisait (25 images par seconde, ou FPS), et l'affichage était parfait.

Cuartielles : 25 FPS, mon premier ordinateur était loin d'en faire autant. Résumons-nous : vous étiez parvenu à exécuter Doom sur le plus lent des deux processeurs d'une carte à modules Bluetooth et Wi-Fi. La vidéo est délivrée par USB-C. Ce port peut recevoir une souris, un clavier, un hub, que sais-je encore. Quel périphérique d'entrée utilisiez-vous pour Doom ?

Facchin : Le projet n'a hélas pas été plus loin. Après avoir réussi à afficher Doom, nous sommes passés à LVGL, une bibliothèque graphique bien plus utile aux développeurs pour l'écriture d'applications destinées à la Portenta. LVGL sait traiter les événements venant de l'hôte USB, p. ex. un clic de souris, ce qui permet d'écrire des interfaces pour toutes sortes de contextes professionnels.

Cuartielles : Ce serait cool d'avoir un automate programmable industriel à Portenta H7 sur lequel on pourrait jouer à Doom sur le M4 pendant que le M7 bosse, non ?

Facchin : Absolument !

Cuartielles : Merci Martino, c'était sympa de vous entendre narrer ce portage de Doom sur la Portenta H7. Nous ne manquons pas de partager les instructions permettant de le faire. Il restera au lecteur à ajouter au code le périphérique de son choix pour profiter du jeu. ◀

220542-04 — VF : Hervé Moreau

À propos de l'auteur

David Cuartielles est le cofondateur d'Arduino. Il est titulaire d'un doctorat en design interactif et d'un master en génie des télécommunications. Il enseigne à l'université de Malmö.

Des questions, des commentaires ?

Contactez Elektor par courriel à redaction@elektor.fr.



Produits

- **Arduino Portenta H7**
www.elektormagazine.fr/arduino-portenta-h7

20%

de réduction
sur la première année
de votre abonnement

Rejoignez la communauté Elektor

Souscrivez un
abonnement
ou



Gold Green

- Accès à l'archive numérique depuis 1978 !
- 8 x magazine imprimé Elektor
- 8 x magazine numérique (PDF)
- 10 % de remise dans l'e-choppe et des offres exclusives pour les membres
- Accès à plus de 5000 fichiers Gerber
- Livraison gratuite en France



www.elektormagazine.fr/arduino-member

Utilisez le code promo :

ARDUINO22

introduction rapide au monde de l'Arduino

carte de développement pour l'Arduino Nano

Wolfgang Trampert (Allemagne)

Elektor reste fidèle à sa mission éducative : nous vous présentons ici une toute nouvelle carte d'apprentissage basée sur l'Arduino Nano. Associée à une formation pratique et bien structurée, elle constitue une plateforme idéale pour améliorer vos compétences et explorer le monde des microcontrôleurs.

Note de l'éditeur

Au moment de la publication de cet article, le livre associé à ce kit n'est disponible qu'en allemand. Des traductions sont prévues très prochainement. Une fois terminée, la version française sera disponible dans l'e-choppe Elektor.

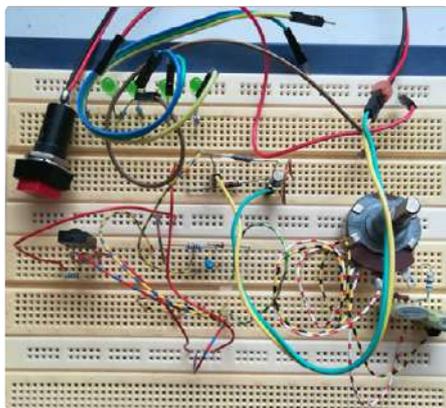


Figure 1. Montage simple pour tester un croquis Arduino.

On pourrait dire que la « philosophie Arduino » est associée à une approche de conception axée sur le matériel : dans la plupart des cas, le logiciel ou le croquis Arduino permet de commander des composants, notamment des interrupteurs, des boutons-poussoirs, des potentiomètres, des LED, des LCD, des buzzers piézoélectriques, des transistors de commande, etc. via les GPIO du microcontrôleur. D'autres types de périphériques ou de modules électroniques tels que des capteurs, des cartes d'affichage ou de pilotage utilisent diverses interfaces série telles que SPI, I²C ou le bus 1 fil pour communiquer avec le contrôleur. Afin de vous familiariser avec le monde des microcontrôleurs et des cartes Arduino, vous pourrez construire de nouveaux circuits pratiques et les contrôler avec une carte Arduino.

Cependant, la base de tout projet utilisant du matériel Arduino est le développement du logiciel (*sketch*). Le matériel n'est qu'un moyen d'arriver à ses fins. Vous pouvez bien sûr connecter tous les périphériques dont vous avez besoin pour un projet particulier en utilisant une carte de prototypage (**figure 1**). Cette approche vous donne un maximum de liberté pour installer les périphériques et acheminer les signaux selon vos besoins. Ce n'est pas toujours une bonne idée, surtout lorsque vous êtes débutant. La taille du montage peut rapidement augmenter et se compliquer en connectant un grand nombre de fils Dupont. Souvent, vous passerez plus de temps à déboguer le matériel et à résoudre les erreurs de câblage qu'à écrire le code. Cela ne fait qu'augmenter le niveau de frustration, et la correction d'erreurs ne vous apprendra pas nécessairement grand-chose d'utile.

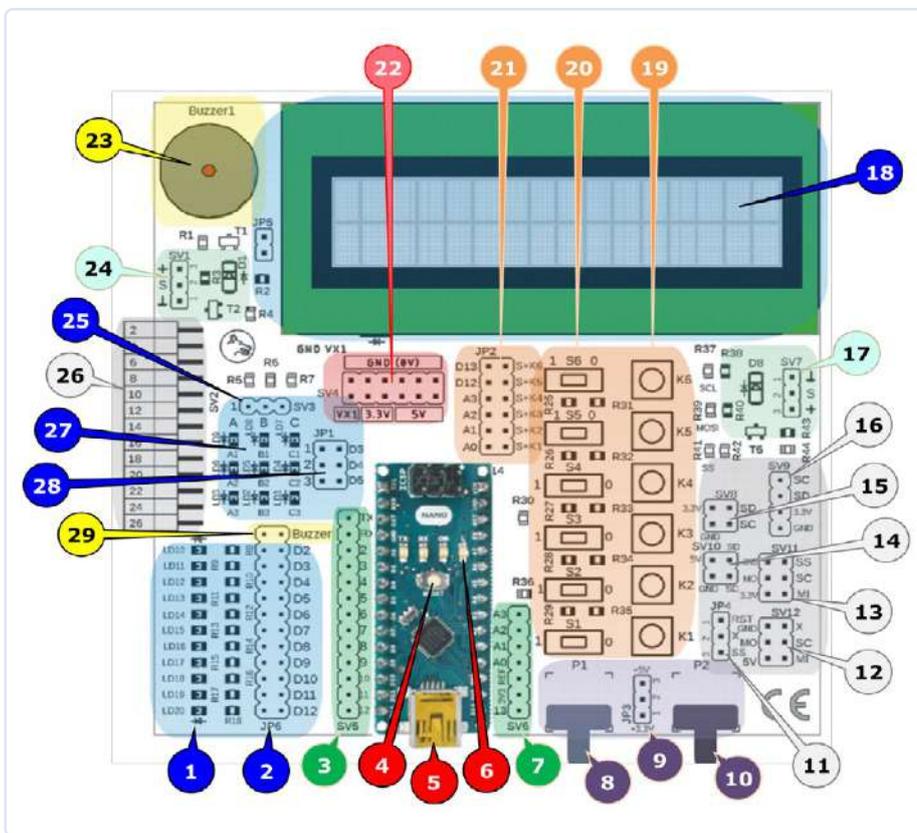


Figure 2. La carte d'apprentissage MCCAB, Rev. 3.3.

Carte d'apprentissage MCCAB

Pour contourner ces obstacles, nous avons développé la carte d'apprentissage Arduino d'Elektor, également connue sous le nom de *carte d'apprentissage MCCAB* (figure 2). Le cœur du système est une carte Arduino Nano que vous pouvez connecter sur la carte d'apprentissage MCCAB. Vous trouverez en plus de celle-ci un grand nombre de périphériques de base dont vous aurez généralement besoin pour construire de nouveaux prototypes pour de nombreuses applications telles qu'une configuration pour votre laboratoire, des circuits de test et d'expérimentation, des projets et des exercices pour faciliter vos études et votre apprentissage, ainsi que des projets de loisirs. Les E/S (GPIO) du microcontrôleur sont toutes accessibles grâce à deux connecteurs sur la carte d'apprentissage MCCAB, ce qui assure une flexibilité de déploiement maximale. Il est possible de connecter des périphériques supplémentaires ou de transmettre des signaux externes avec des fils Dupont, selon vos besoins. Vous n'aurez pas à vous soucier d'un câblage incorrect et vous passerez moins de temps à fouiller dans vos boîtes pour trouver le composant indispensable à la réalisation de votre circuit. Il est également possible de connecter facilement des circuits supplémentaires construits sur des platines à essai avec des câbles Dupont, puisque toutes les E/S du microcontrôleur de l'Arduino Nano sont connectées

aux deux connecteurs SV5 et SV6 de la carte d'apprentissage MCCAB (flèches 3 et 7 sur la figure 2). Sur le côté gauche de la carte, vous pouvez également voir la double rangée de 26 broches à angle droit du connecteur SV2 (flèche 26 sur la figure 2) où vous pouvez insérer une carte d'extension externe. Ce connecteur fournit tous les signaux des E/S importants du microcontrôleur. Les cartes externes destinées à réaliser des fonctions telles qu'un traceur de courbe de composants électroniques, une alimentation de laboratoire ou un contrôleur de feux de signalisation peuvent être connectées à la carte d'apprentissage MCCAB et commandées par celle-ci. Les informations et les résultats d'un croquis en cours d'exécution peuvent être affichés sur l'écran LCD intégré de 2 x 16 caractères (flèche 18), reliés par le bus I2C de la carte. Une matrice de LED 3 x 3 est également intégrée (flèche 27).

La carte d'apprentissage MCCAB est alimentée par une alimentation $V_{cc} = +5\text{ V}$. Cette alimentation est généralement fournie par le câble USB branché sur votre PC, dont vous avez besoin pour créer et téléverser le croquis de chaque exercice sur la MCCAB. Vous pouvez également alimenter la MCCAB avec un bloc d'alimentation externe. Dans le schéma de la carte d'apprentissage (figure 2), tous les composants associés à une fonction spécifique de la carte sont identifiés avec une couleur de fond commune.

Les commandes et indicateurs de la carte d'apprentissage MCCAB

- | | |
|----|---|
| 1 | 11 x LED (indication de l'état des entrées/sorties D2 à D12) |
| 2 | Connecteur reliant les LED LD10 à LD20 avec les E/S D2 à D12 |
| 3 | Entrées et sorties du microcontrôleur |
| 4 | Bouton <i>RESET</i> |
| 5 | Arduino NANO avec prise mini USB |
| 6 | LED <i>L</i> , associé à la broche D13 |
| 7 | E/S du microcontrôleur |
| 8 | Potentiomètre P1 |
| 9 | Tension d'alimentation de P1 et P2 |
| 10 | Potentiomètre P2 |
| 11 | Signal sur la broche X du SV12 |
| 12 | Interface SPI 5 V (le signal de la broche X est sélectionné par JP4.) |
| 13 | Interface SPI 3,3 V |
| 14 | Interface I2C 5 V |
| 15 | Interface I2C 3,3 V |
| 16 | Interface I2C 3,3 V |
| 17 | Sortie de commutation pour composants externes |
| 18 | LCD 2 x 16 |
| 19 | 6 x boutons-poussoirs K1 à K6 |
| 20 | 6 x commutateurs à glissière S1 à S6 |
| 21 | Connecteur pour relier les commutateurs aux E/S du microcontrôleur. |
| 22 | Diviseur de tension d'alimentation |
| 23 | <i>Buzzer1</i> |
| 24 | Sortie de commutation pour composants externes |
| 25 | Matrice de colonnes de LED 3 x 3 |
| 26 | 2 x 13 connecteurs pour connecter un module externe |
| 27 | 3 x 3 matrice de LED (rouge) |
| 28 | Connexions des lignes de la matrice 3 x 3 à D3, D4 et D5 |
| 29 | La position du cavalier relie <i>Buzzer1</i> à la broche D9 |

Tableau 1. Classes disponibles dans la bibliothèque MCCAB_Lib.

Classe	Utilisation
KeySwitch	État des commutateurs S1 à S6 et des boutons-poussoirs K1 à K6
Matrix	Commande de la matrice de LED 3 x 3.
LED	Commande d'allumage, d'extinction et de clignotement des 12 LED LD10 à LD20
LedBlock	Affichage d'un rythme binaire sur les 11 LED (LD10 à LD20)

La bibliothèque MCCAB_Lib à utiliser avec les cartes d'apprentissage

Le développement logiciel implique l'utilisation de l'EDI Arduino pour écrire le programme (ou *sketch*) qui commande le microcontrôleur. Le croquis est ensuite compilé et téléversé vers le microcontrôleur de l'Arduino Nano sur la carte d'apprentissage via un câble mini USB. Il est possible de configurer les E/S du microcontrôleur avec la fonction Arduino `pinMode()` et de lire ou de contrôler les données des signaux émis ou reçus par les composants connectés à la carte d'apprentissage avec `digitalRead()`, `digitalWrite()`, `analogRead()`, etc.

Cependant, une bibliothèque *MCCAB_Lib* [1] est disponible et aide les développeurs en fournissant des commandes supplémentaires pour contrôler les divers périphériques matériels de la carte d'apprentissage MCCAB. Vous pouvez télécharger gratuitement cette bibliothèque et l'intégrer dans votre propre croquis. Cette bibliothèque facilite grandement la gestion des périphériques de la carte. La bibliothèque *MCCAB_Lib* contient cinq classes permettant de commander les commutateurs, les LED et le buzzer sur la carte d'apprentissage et peut être facilement incluse dans le croquis de l'utilisateur selon les besoins. Le **tableau 1** présente une liste des classes disponibles.

Grâce à cette bibliothèque, l'utilisateur n'a pas à se soucier de définir des périodes de temps pour le débouclage des commutateurs, de générer des signaux de contrôle multiplex pour la matrice de LED 3 x 3 et de faire clignoter les LED LD10 à LD20 ou L, ou même de générer les fréquences de tonalité du buzzer. Les fonctions de la bibliothèque font cela automatiquement en tâche de fond du programme, sans que l'utilisateur s'en aperçoive.

Listage 1 est un exemple de croquis pour démontrer l'utilisation de la bibliothèque *MCCAB_Lib*.

Sur la ligne 15 du sketch, la variable objet `Led` est déclarée de la classe `LED` de la bibliothèque *MCCAB_Lib*. Le paramètre `LED_PIN` utilisé dans la déclaration de la variable objet `Led` est défini comme une constante dans la ligne 13 indiquant la broche à laquelle la LED est connectée. Cette broche est automatiquement configurée en sortie lors de l'instanciation.

La variable objet `Key` de la classe `KeySwitch` de la bibliothèque *MCCAB_Lib* (déclarée dans la ligne 22) surveille pendant l'exécution (en



Listage 1.

```
/*
 * Sketch which uses pushbutton K4 to toggle LED LD10 on and off using object variables in the
 * "KeySwitch" und "LED" classes in the MCCAB_Lib library.
 * To read the status of pushbutton K4 its necessary to insert a jumper to link position S+K4 (the switch
 * connection) with A3 (GPIO A3 of the microcontroller) on double header strip JP2 of the MCCAB.
 * Insert another jumper (in position D2 of the double header strip J6) to link LED LD10 with the
 * microcontroller GPIO of the MCCAB.
 */

11 #include <MCCAB_Lib.h> // bind the MCCAB_Lib Library to the Sketch
12
13 #define LED_PIN 2 // the LED is connected to pin D2
14
15 LED Led(LED_PIN); // Object-Variable
16
17 //function called by the object-variable "Key" when the switch is closed.
18 void switchTurnedOn() {
19     Led.toggle(); // toggle or flip the state of the LED
20 }
21
22 KeySwitch Key(SK4, ACTIVE_HIGH, switchTurnedOn, nullptr); // Object-Variable
23
24 void setup() { } // nothing to do here...
26 void loop() { } // or here
```

arrière-plan) l'état de l'entrée du commutateur sur la broche SK4, qui lui est passée en paramètre selon sa déclaration. Elle effectue le débouclage du commutateur lorsque le bouton-poussoir K4 est enfoncé ou relâché et appelle la fonction `switchTurnedOn()` lorsque le bouton est enfoncé. La méthode `toggle()` de la classe `LED` de la bibliothèque `MCCAB_Lib` est activée dans la fonction `switchTurnedOn()` de la ligne 19 pour inverser l'état courant de la diode électroluminescente LD10.

Puisque les broches de connexion du commutateur et de la LED sont automatiquement configurées comme entrée et sortie lors de la déclaration des objets, il n'y a rien à ajouter dans la fonction `setup()` de la ligne 24 de ce croquis.

La fonction `loop()` de la ligne 26 ne contient pas non plus d'instructions, car la seule tâche à effectuer dans ce sketch est de commuter l'état de la LED lorsque le bouton K4 est pressé. Cette action est déclenchée par la classe `KeySwitch` en appelant la fonction `switchTurnedOn()`.

En utilisant les classes de la bibliothèque `MCCAB_Lib`, dans des croquis plus longs, vous n'aurez pas besoin d'utiliser les deux fonctions `setup()` et `loop()` du modèle logiciel standard d'Arduino pour vérifier continuellement l'état des composants périphériques, ce qui les libérerait pour des tâches plus importantes.

12 projets, croquis, et 46 exercices

Un manuel d'instruction détaillé pour la carte d'apprentissage MCCAB est disponible et téléchargeable sur [1]. La carte d'apprentissage MCCAB et la bibliothèque `MCCAB_Lib` seront également décrites en détail dans un prochain livre qui sera bientôt disponible (voir note de l'éditeur).

Le livre explique en détail les bases matérielles et logicielles d'un système à microcontrôleur et introduit le langage de programmation C, qui est utilisé pour écrire des croquis Arduino. Le livre porte principalement sur les exercices pratiques. L'apprentissage par la pratique est le concept clé utilisé ici pour vous permettre d'acquérir les compétences dont vous aurez besoin lorsque vous vous lancerez dans la construction de vos propres projets. Dans une section pratique détaillée, vous trouverez 12 croquis de projets et 46 exercices qui vous permettent de développer vos connaissances au fur et à mesure que vous travaillez sur les nombreux exemples. Les exercices sont structurés de telle sorte que le lecteur se voit proposer une tâche à résoudre avec la carte d'apprentissage MCCAB en utilisant les connaissances acquises dans la partie théorique du livre. Pour chaque exercice, il y a ensuite une explication détaillée et un exemple de solution bien commenté qui aide à résoudre les problèmes. ◀

220450-04

Des questions, des commentaires ?

Contactez Elektor (redaction@elektor.fr).

À propos de l'auteur

Wolfgang Trampert a commencé à développer et à programmer des systèmes à microcontrôleurs après avoir terminé ses études en électronique. Son entreprise d'ingénierie a développé des solutions basées sur les microcontrôleurs pour répondre aux besoins des clients. Il est l'auteur de plusieurs ouvrages et articles spécialisés et anime des formations sur le thème des microcontrôleurs.



Produits

> **MCCAB Training Board (SKU 20295)**
www.elektor.fr/20295

> **Mikrocontroller-Praxiskurs für Arduino-Einsteiger (livre en allemand, SKU 20293)**
www.elektor.fr/20293

LIENS

[1] Bibliothèque `MCCAB_Lib` :
www.elektor.fr/20295



Arduino & Co – Measure, Control, and Hack

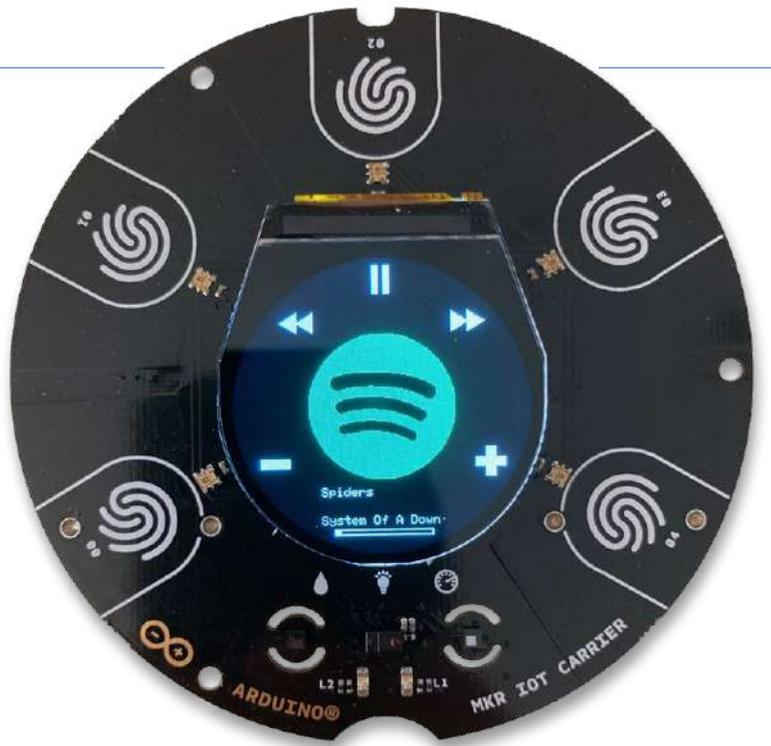
Avec une simple carte Arduino Pro Mini et quelques composants, il est possible de réaliser des projets, autrefois irréalisables (ou qui auraient coûté une fortune) facilement et à moindre coût grâce à ce nouveau livre (en anglais) : de simples projets d'éclairage de LED, une station de recharge et de test de batterie rechargeable, et encore plus!

www.elektor.fr/20243



contrôleur pour Spotify

le kit Oplà IoT contient
(presque) tout ce dont vous
avez besoin



Altuğ Bakan (Turquie)

Le kit Arduino Oplà IoT contient la carte pour maker MKR WiFi 1010 et une carte support qui intègre des relais, un écran OLED de forme ronde, des boutons tactiles capacitifs et quelques capteurs. Nous expliquons dans cet article comment construire un contrôleur portable pour le célèbre lecteur de musique Spotify. Certains éléments de sécurité sont bien entendu nécessaires.

Grâce à ses capacités wifi, la carte Arduino MKR WiFi 1010 constitue un cerveau parfait pour votre prochain projet IdO (Internet des objets). Mieux encore, avec le kit Arduino Oplà IoT, vous disposez à la fois d'une carte pour maker et d'une carte support (figure 1). Cette dernière contient des relais, un écran OLED de forme ronde et des

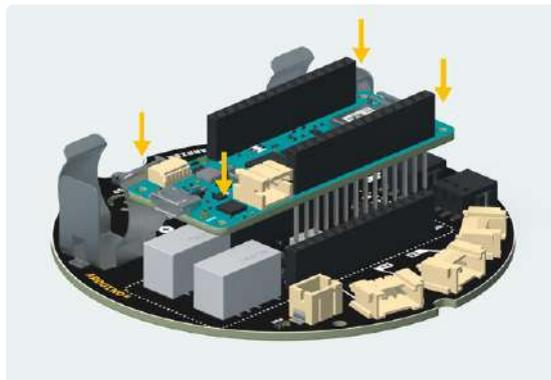


Figure 1, La carte pour maker MKR WiFi 1010 est placée sur la carte support, dotée de relais et d'autres périphériques nécessaires. .

boutons tactiles capacitifs. Le kit comprend également un capteur d'humidité et un capteur infrarouge passif (IRP) (figure 2). Il est ainsi facile de mettre en œuvre des dispositifs comme les alarmes de sécurité domestique et l'arrosage automatique des plantes.

La fonction wifi permet également de contrôler des programmes exécutés sur votre PC, s'ils disposent d'une interface réseau. Les boutons tactiles, l'écran, le support de batterie et le boîtier permettent de créer facilement un contrôleur portable pour différents types de logiciels PC, en complément d'une souris et d'un clavier (figure 3).

Je suis fan du lecteur musical Spotify, et j'ai donc utilisé le kit Oplà pour construire mon propre contrôleur Spotify sans fil. Vous pouvez appuyer sur des boutons pour passer à la chanson suivante ou précédente, ou encore la lire ou la mettre en pause, et augmenter ou diminuer le volume. Pour ce faire, il va sans dire que le lecteur Spotify doit être lancé sur votre PC ou votre smartphone.

Communication sécurisée

Spotify est livré avec une interface de programmation facile à utiliser, qui permet de contrôler votre lecteur via le réseau ; vous aurez besoin pour cela de la licence Spotify Plus. Certains éléments de sécurité sont bien entendu nécessaires. Pour utiliser l'API web de Spotify, basée sur REST, vous devez d'abord vous authentifier sur le serveur de comptes Spotify avec votre nom d'utilisateur et votre mot de passe de connexion. Une fois authentifié, votre logiciel doit envoyer un identifiant (ID) client et un secret client. Le serveur Spotify vous renvoie ensuite un jeton d'accès, que vous devrez émettre à chaque appel de l'API web pour contrôler votre lecteur Spotify. Ce flux



Figure 2 . Le kit Arduino Oplà IoT.



Figure 3 . Le support de batterie permet d'obtenir un kit Oplà IoT portable.

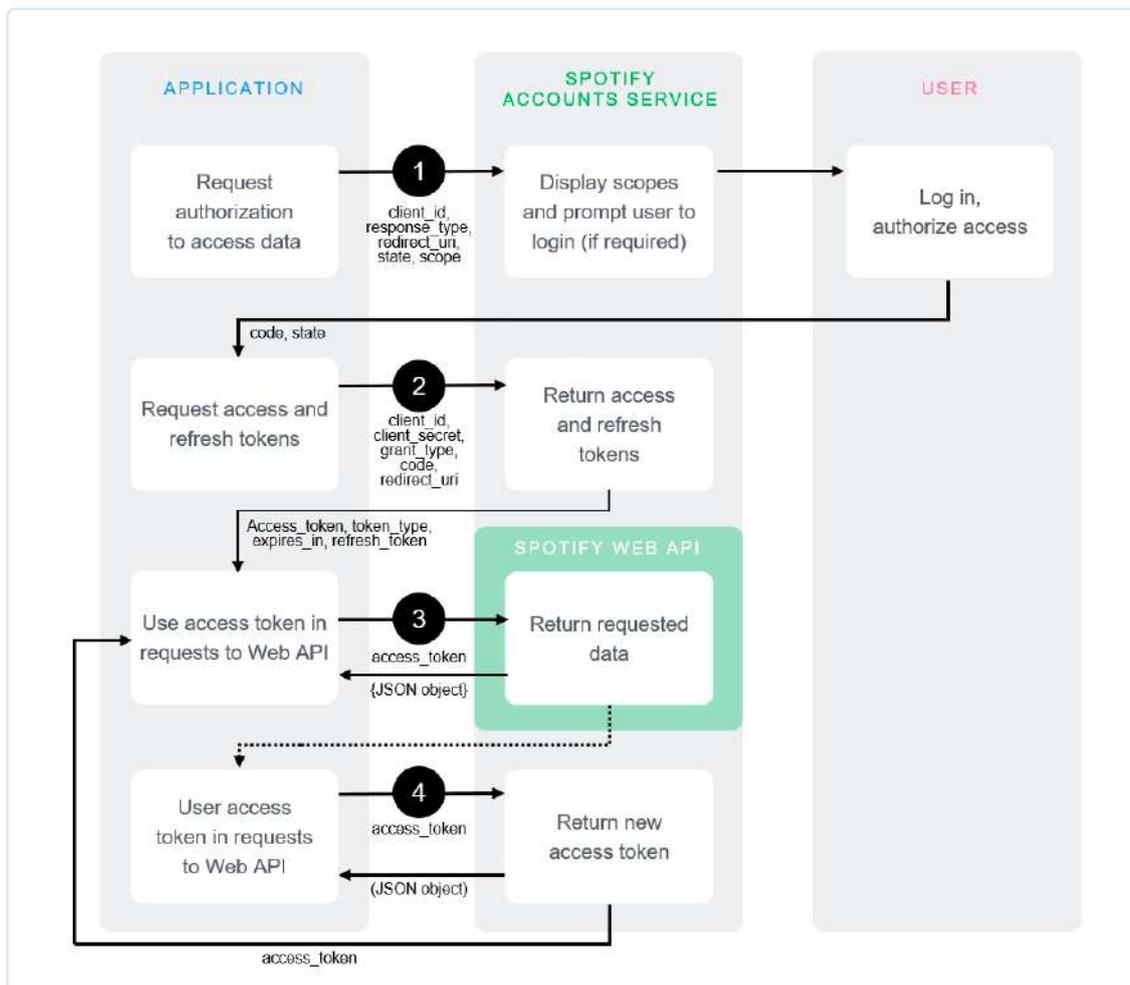


Figure 4 . Le flux d'authentification en plusieurs étapes est basé sur le processus OAuth 2.0 bien connu.

Figure 5. Vous devez créer une « appli » pour obtenir ...

Figure 6. ...votre identifiant et votre secret client.

d'authentification en deux étapes est basé sur le processus *OAuth2* bien connu (voir la **figure 4**).

Comment obtenir votre identifiant et votre secret client ? Il vous suffit d'utiliser le créateur d'applications Spotify App Builder [1] pour concevoir votre propre logiciel PC ou une application mobile servant à contrôler Spotify (**figure 5**). Cependant, nous ne le faisons pas ici ; nous voulons simplement les informations d'identification (voir la **figure 6**). L'identifiant et le secret client doivent être stockés sur notre carte Arduino MKR. Bien sûr, vous pourriez les coder « en dur » dans le croquis Arduino, mais il existe un moyen simplifié et plus sûr de le faire. L'éditeur web Arduino [2] propose un onglet *Secrets*, dans lequel vous pouvez définir des variables d'environnement utilisées ensuite dans votre code (**figure 7**). Il suffit de saisir l'identifiant client Spotify et le secret ainsi que le nom et le mot de passe de votre réseau wifi dans les champs de l'onglet. Si vous compilez et téléchargez le logiciel sur le contrôleur, vos valeurs secrètes individuelles seront également téléchargées pour être utilisées par le code du projet. Dans votre croquis, vous devez remplacer les chaînes de caractères contenant des données sensibles en écrivant une expression `SECRET_xxx` — par exemple : `SECRET_SPOTIFY_CLIENT`.

Authentification

Pour lancer le flux *OAuth2*, vous devez vous authentifier auprès de Spotify. Lorsque vous démarrez le contrôleur Spotify décrit ici, il se connecte au réseau wifi spécifié (maison) et affiche sur l'écran OLED l'adresse IP obtenue par le routeur. Je voulais donner à l'utilisateur une chance de s'authentifier facilement sur Spotify, et j'ai donc créé l'approche suivante. Le contrôleur Arduino génère une petite page web qui sera affichée dans un navigateur si

Figure 7. Saisissez toutes les valeurs privées dans l'onglet *Secrets* de l'éditeur web Arduino, avant de compiler et de télécharger le code.



Figure 8. Page web proposée par le contrôleur, pour se connecter à Spotify.

vous entrez l'adresse IP de votre contrôleur (**figure 8**). Cette petite page web contient un lien. (Référez-vous au **listage 1** pour voir comment la page web est produite dans le code Arduino). Si vous appuyez dessus, le navigateur se rend sur la page d'authentification de Spotify, où vous pouvez facilement vous connecter. Il vous sera alors demandé si vous donnez au contrôleur l'autorisation de contrôler Spotify (**figure 9**). À noter : pour que tout cela fonctionne, vous devez

également entrer l'adresse IP du contrôleur en tant que « Redirect URI » (URI de redirection) dans l'éditeur d'application Spotify (**figure 10**). À partir de là, le contrôleur Arduino peut obtenir le jeton d'accès à l'API en envoyant l'identifiant client et le secret à Spotify (**listage 2**). Le jeton d'accès doit être régulièrement réactualisé pendant le fonctionnement. Une fonction du croquis (**listage 3**), appelée toutes les 3000 secondes, permet de réaliser cette actualisation.



Listage 1. Page web pour s'authentifier sur Spotify, mise à disposition par le contrôleur Spotify.

```
String webpage = "<!DOCTYPE html>\n";
webpage += "<html><body>";
webpage += getStyle();
webpage += "<a href=\"https://accounts.spotify.com/authorize?client_id=";
webpage += SPOTIFY_CLIENT;
webpage += "&response_type=code&redirect_uri=http://";
webpage += ip_address;
webpage += "/redirect/&scope=user-read-playback-state user-modify-playback-state\">Authenticate Spotify</a>\n";
webpage += "</body></html>";
wifiClient.print(webpage);
```

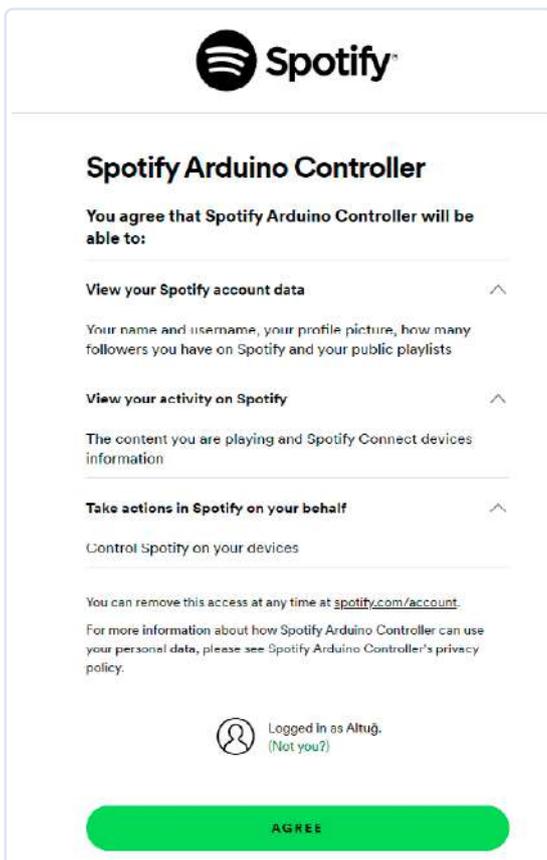
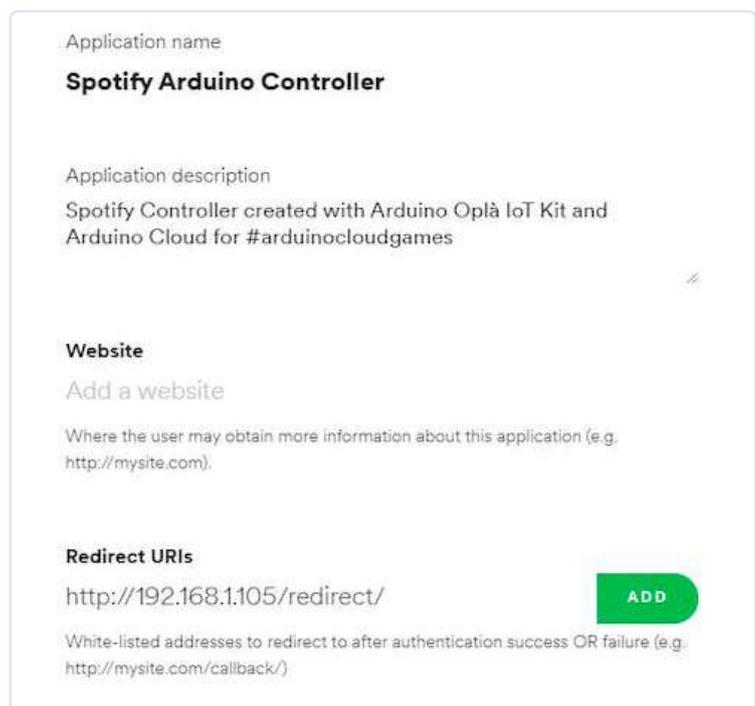


Figure 9. Si vous êtes connecté(e) à Spotify, vous devez donner au contrôleur l'autorisation d'agir en votre nom.

Figure 10. URI de redirection : l'adresse du contrôleur dans votre réseau domestique.





Listage 2. Fonction destinée à obtenir le jeton auprès de Spotify pour une utilisation ultérieure par l'API.

```
// Get the user authorization token
bool getAccessToken(String userCode) {
    String postData = "grant_type=authorization_code&code=" + userCode + "&redirect_uri="
        "http://" + ip_address + "/redirect/";
    authClient.beginRequest();
    authClient.post("/api/token");
    authClient.setHeader("Content-Type", "application/x-www-form-urlencoded");
    authClient.setHeader("Content-Length", postData.length());
    authClient.sendBasicAuth(SPOTIFY_CLIENT, SPOTIFY_SECRET);
    // send the client id and secret for authentication
    authClient.beginBody();
    authClient.print(postData);
    authClient.endRequest();

    // If successful
    if (authClient.responseStatusCode() == 200) {
        lastTokenTime = millis();
        DynamicJsonDocument json(512);
        deserializeJson(json, authClient.responseBody());
        accessToken = json["access_token"].as<String>();
        refreshToken = json["refresh_token"].as<String>();
        return true;
    }
    return false;
}
```



Listage 3. Fonction servant à actualiser le jeton.

```
// Refresh the user authentication token
void refreshAccessToken() {
    String postData = "grant_type=refresh_token&refresh_token=" + refreshToken;
    authClient.beginRequest();
    authClient.post("/api/token");
    authClient.setHeader("Content-Type", "application/x-www-form-urlencoded");
    authClient.setHeader("Content-Length", postData.length());
    authClient.sendBasicAuth(SPOTIFY_CLIENT, SPOTIFY_SECRET);
    // send the client id and secret for authentication
    authClient.beginBody();
    authClient.print(postData);
    authClient.endRequest();

    // If successful
    if (authClient.responseStatusCode() == 200) {
        lastTokenTime = millis();
        DynamicJsonDocument json(256);
        deserializeJson(json, authClient.responseBody());
        accessToken = json["access_token"].as<String>();
    }
}
```



Figure 11 . Les fonctions des boutons sont affichées à l'écran.



Listage 4. Exemple d'utilisation de l'API (morceau suivant et précédent).

```
// Skip a song towards a given direction
void skipSong(String direction) {
  apiClient.beginRequest();
  apiClient.post("/v1/me/player/" + direction);
  apiClient.setHeader("Content-Length", 0);
  apiClient.setHeader("Authorization", "Bearer " + accessToken);
  apiClient.endRequest();
}
```

Fonctionnement

Le reste du code est moins complexe. L'appareil affiche le logo Spotify et la fonction des boutons sur l'écran OLED (**figure 11**). Si l'utilisateur touche un bouton, la fonction d'API correspondante est appelée. Référez-vous au **listage 4** pour voir comment passer d'un morceau au précédent ou au suivant.

Une autre fonction du code demande le statut du lecteur à l'API Spotify. La réponse est une chaîne JSON. J'utilise la bibliothèque *ArduinoJson.h* et certaines de mes propres fonctions pour traiter plus facilement les chaînes JSON. Pour obtenir l'état des boutons, commander les LED et afficher des graphiques sur l'écran OLED, j'utilise la bibliothèque *Arduino_MKRIoTCarrier.h*. Vous pouvez vous plonger dans mon code pour vous en inspirer et réaliser vos propres projets avec le kit Oplà. Il est possible de télécharger mon logiciel à l'adresse [3].

Connexion à la plateforme Arduino Cloud

J'ai également établi une connexion avec la plateforme Arduino Cloud et créé un tableau de bord qui affiche la chanson en cours et le nom de l'artiste à côté du volume de l'appareil (**figure 12**). Vous pouvez bien sûr créer votre propre tableau de bord personnel, avec les données que vous souhaitez.

Mon projet a remporté la 3^e place des *Arduino Cloud Games 2022* !

(220407-04) VF : Pascal Godart

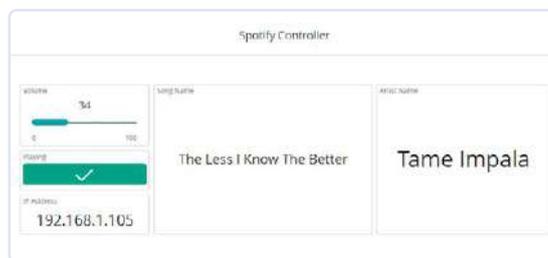


Figure 12 . Le morceau en cours et le nom de l'artiste sont envoyés à la plateforme Arduino Cloud, où ils sont visibles sur votre tableau de bord personnel.

Des questions, des commentaires ?

Envoyez un courriel à l'auteur (mail@alt.ug) ou contactez Elektor à (redaction@elektor.fr).

À propos de l'auteur

Altug Bakan a travaillé comme ingénieur électronique, principalement avec des systèmes embarqués. Il aime utiliser Arduino dans son travail pour ses qualités de prototypage rapide et de convivialité. Ses sujets préférés en électronique sont la programmation embarquée dite « bare-metal » (sans système d'exploitation associé) et l'Internet des objets (IdO)..

Produits

> **Arduino Oplà IoT Kit**
www.elektormagazine.fr/arduino-opla-iot-kit

LIENS

[1] Spotify App Builder : <https://developer.spotify.com/dashboard/>

[2] Éditeur web Arduino : <https://create.arduino.cc/editor>

[3] Le projet présent sur create.arduino.cc : <https://create.arduino.cc/projecthub/Altug/opla-spotify-controller-6e7bc4>

créez, déployez et maintenez des applications évolutives et sécurisées

avec Arduino Portenta X8 équipé du mini processeur d'applications i.MX 8M de NXP et de l'élément de sécurité EdgeLock® SE050

Contribué par NXP Semiconductors

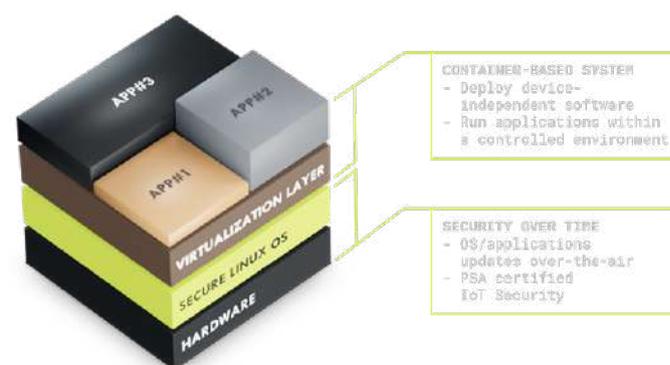
La mise sur le marché d'un dispositif **IdO** implique des efforts de conception et de développement considérables avec des problèmes d'évolutivité, des défis de sécurité et des limitations de dispositifs à chaque coin de rue. L'ajout d'intelligence complexifie encore plus. C'est pourquoi la sélection du bon matériel et du bon logiciel de développement est essentielle pour mettre plus rapidement sur le marché des produits périphériques sécurisés. Cet article présente la plateforme Arduino Portenta X8, un SoM sécurisé de qualité industrielle basé sur le microprocesseur d'applications i.MX 8M de NXP et un élément matériel de sécurité EdgeLock® SE050 intégré. Cette plateforme certifiée PSA est également Arm® SystemReady IR pour une sécurité assu.

Arduino Portenta X8 est un système puissant de qualité industrielle sur un module avec Linux® préchargé à bord, capable d'exécuter des logiciels indépendants du dispositif grâce à son architecture modulaire en conteneur. Il offre deux approches : la flexibilité d'utilisation de Linux combinée à des applications en temps réel via l'environnement Arduino. La connectivité wifi/BLE embarquée permet de mettre à jour le système d'exploitation et les applications à distance, tout en maintenant l'environnement du noyau Linux à un niveau de performance optimal.

Sécurité de pointe

Le système basé sur des conteneurs intègre différentes couches de sécurité, à commencer par la matérielle qui comprend le *Secure Element* de NXP. Il utilise la plateforme DevOps basée sur le cloud de Foundries.io [1] pour réinventer la manière dont les solutions

Linux embarquées sont construites, testées, déployées et maintenues. Le Portenta X8 comprend le système d'exploitation de microplateforme Linux open-source personnalisable, conçu selon les meilleures pratiques de l'industrie en matière de sécurité de bout en bout, de mises à jour *OTA* incrémentielles et de gestion de flotte.

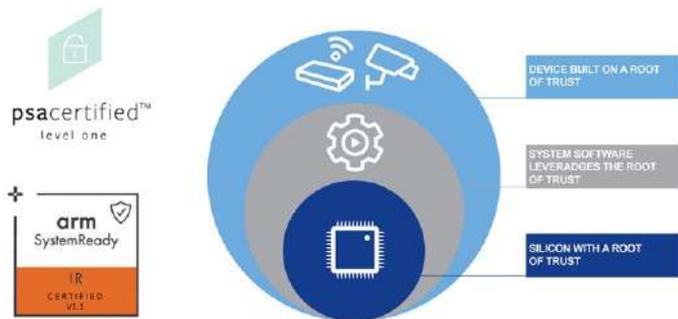


Portenta X8 Container and Security.

La couche de virtualisation permet aux utilisateurs de déployer des logiciels indépendants des appareils et fonctionnant dans un environnement contrôlé. Ils peuvent créer leurs propres conteneurs à l'aide de Docker et télécharger des images préfaites à partir de Docker Hub ou d'autres registres publics disponibles pour concevoir une application sur mesure. Si le développeur souhaite intégrer de l'embarqué, il peut le faire facilement en l'exécutant sur un conteneur, en la plaçant sur la carte et en la testant dès sa sortie de la boîte. Cela offre un large éventail de possibilités en mélangeant les capacités de Linux et d'Arduino.

Portenta X8 a obtenu la certification PSA et l'élément matériel de sécurité NXP EdgeLock SE050 assure la génération de clés, l'accélération des opérations cryptographiques et le stockage sécurisé. Il a également obtenu la certification Arm® SystemReady [2] et a intégré les services Parsec, ce qui en fait l'un des premiers produits Cassini ou dispositifs Edge natifs du cloud disponibles pour les développeurs sur le marché. Il fonctionne de manière transparente avec Fedora IoT, Fedora Server, Debian et la microplateforme Linux. En permettant la migration des charges de travail du cloud vers la périphérie, le

Portenta X8 contribue à une expérience de développeur cloud-native à travers l'écosystème IdO diversifié et sécurisé d'Arm.



Platform Security Architecture.

EdgeLock SE050 - Un ancrage de confiance pour l'IdO

L'EdgeLock SE050 [3] de NXP est un matériel de sécurité discret et inviolable destiné à protéger l'identité d'un dispositif, y compris les clés cryptographiques et les certificats. C'est un élément intégré relié au processeur principal via l'interface I²C. L'EdgeLock SE050 est certifié Critères Communs EAL 6+ pour le matériel et le système d'exploitation. Prêt à l'emploi pour les appareils IdO, il fournit une base de confiance au niveau du circuit intégré et offre une véritable sécurité de bout en bout, de la périphérie au cloud, sans qu'il soit nécessaire d'implémenter un code de sécurité ni de manipuler des clés et des informations d'identification critiques.



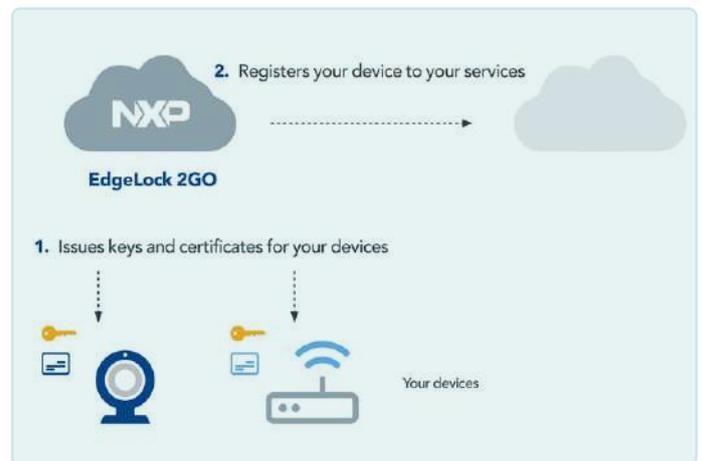
Une base de confiance fondée sur le silicium : élément de sécurité EdgeLock® SE050.

Livré sous la forme d'une solution prête à l'emploi, EdgeLock SE050 offre plusieurs algorithmes et protocoles cryptographiques pré-implémentés et est accompagné d'un paquet de support qui simplifie la conception et réduit le temps de mise sur le marché. En plus des bibliothèques pour différents microcontrôleurs (MCU) et MPU, le paquet de support offre également l'intégration avec les nombreux OS courants, y compris Linux, RTOS et Android.

Les concepteurs de dispositifs IdO sont confrontés à deux défis majeurs lors de la mise en œuvre de l'embarquement des dispositifs sur le

cloud. D'abord, le provisionnement de l'identité du dispositif faisant référence à l'installation des clés et des certificats. Le second défi est la gestion des identités des dispositifs une fois libérés sur le terrain, faisant référence à la mise à jour, l'ajout ou la révocation des clés et des certificats tout au long du cycle de vie du dispositif.

Pour aider les concepteurs à relever ces défis, NXP propose le service géré EdgeLock 2GO [4]. La plateforme est une combinaison de matériel et de services spécialement conçue qui établit une base de confiance fondée sur le silicium. EdgeLock 2GO émet les identités requises pour les dispositifs IdO et installe les références en toute sécurité dans le matériel EdgeLock SE050. Il enregistre également automatiquement le dispositif IdO directement auprès du cloud.



NXP gère les informations d'identification des dispositifs.

TCe service flexible prend en charge plusieurs types de justificatifs d'identité et applique différentes configurations en fonction du projet. Les accréditations peuvent être renouvelées ou ajoutées aux dispositifs mis en service sur le terrain. Avec la mise en place d'EdgeLock SE050 et de l'EdgeLock 2GO, les utilisateurs bénéficient d'une solution de bout en bout qui est simple, sûre et flexible.

L'IdO continue de se développer, mais les risques aussi. La combinaison EdgeLock de NXP, avec sa sécurité matérielle et son service de gestion des informations d'identification, offre aux fabricants de dispositifs un moyen plus sûr de faire des affaires. En prenant en charge le déploiement d'un appareil, NXP EdgeLock réduit les délais de mise sur le marché et les coûts quotidiens d'exploitation d'un déploiement IdO tout en ayant la certitude que les appareils sont protégés par une sécurité de haut niveau.

Libérez la puissance : plus de vitesse et une meilleure efficacité

Le SoC i.MX 8M Mini [5] est le premier processeur d'applications multi-cœur embarqué de NXP construit à l'aide de la technologie de processeur 14LPC FinFET avancée, offrant plus de vitesse et une meilleure efficacité énergétique. La famille de processeurs d'applications i.MX 8M Mini associe la haute performance informatique, l'efficacité énergétique et la sécurité embarquée nécessaires à la croissance rapide de l'informatique des nœuds de périphérie, du streaming multimédia et des applications de *machine learning*.



Le SoC i.MX 8M Mini est proposé dans des variantes à un, deux et quatre cœurs utilisant le processeur Arm® Cortex®-A53 et fonctionnant jusqu'à 1,8 GHz par cœur. Livré dans un processus avancé à faible consommation, le complexe de cœurs est optimisé pour un fonctionnement sans ventilateur, un faible coût thermique du système et une longue durée de vie de la batterie. Les cœurs Cortex-A peuvent être mis hors tension tandis que le sous-système Cortex-M4 assure la surveillance du système en temps réel et à faible consommation. Le contrôleur DRAM prend en charge les mémoires 32 bits/16 bits LPDDR4, DDR4 et DDR3L, offrant ainsi une grande souplesse de conception du système.

Les options de cœur du i.MX 8M Mini sont optimisées pour une consommation ultra-faible, voire inférieure à un watt dans certaines applications spécifiques, mais offrent la puissance de traitement nécessaire pour les applications grand public, audio, industrielles, de machine learning et d'inférence sur une gamme de fournisseurs de cloud. Le SoC i.MX 8M Mini intègre également l'accélération matérielle de la vidéo 1080p pour permettre des applications vidéo bidirectionnelles, des graphiques 2D et 3D pour offrir une expérience visuelle IHM riche, et des capacités audio avancées pour permettre des applications riches en sons. Une sélection étendue d'interfaces à haut débit permet une connectivité plus large du système et vise une qualification de niveau industriel.

Les exemples d'application comprennent :

► Automatisation industrielle

Le Portenta X8 peut alors faire office de passerelle multiprotocole, en envoyant des données au cloud ou au système ERP via wifi, LoRa, NB/IoT, LTE Cat.M1.

La disponibilité de conteneurs Linux comme ROS au sein de l'environnement Arduino fait du Portenta X8 une solution idéale pour les véhicules guidés autonomes.

► Automatisation des bâtiments

Interagissant avec des capteurs intelligents sur le plan environnemental, Portenta X8 permet la mise en œuvre de ML et de traitement d'images en temps réel sur la périphérie.

Les kiosques intelligents s'appuient généralement sur plusieurs composants (par exemple, des lecteurs de cartes, des caméras, des microphones), ce qui nécessite une sélection diversifiée d'E/S. Associé à un Max Carrier, le Portenta X8 assure la connectivité wifi et permet aux administrateurs de surveiller à distance l'utilisation des machines.

Le Portenta X8 peut simultanément contrôler les systèmes CVC, allumer/éteindre les appareils intelligents, régler de manière autonome l'éclairage et contrôler les accès en périphérie.

Commencez à développer dès aujourd'hui avec le SOM Portenta X8 [6] de qualité industrielle, sécurisé et doté d'une densité de calcul exceptionnelle. ◀

220576-04

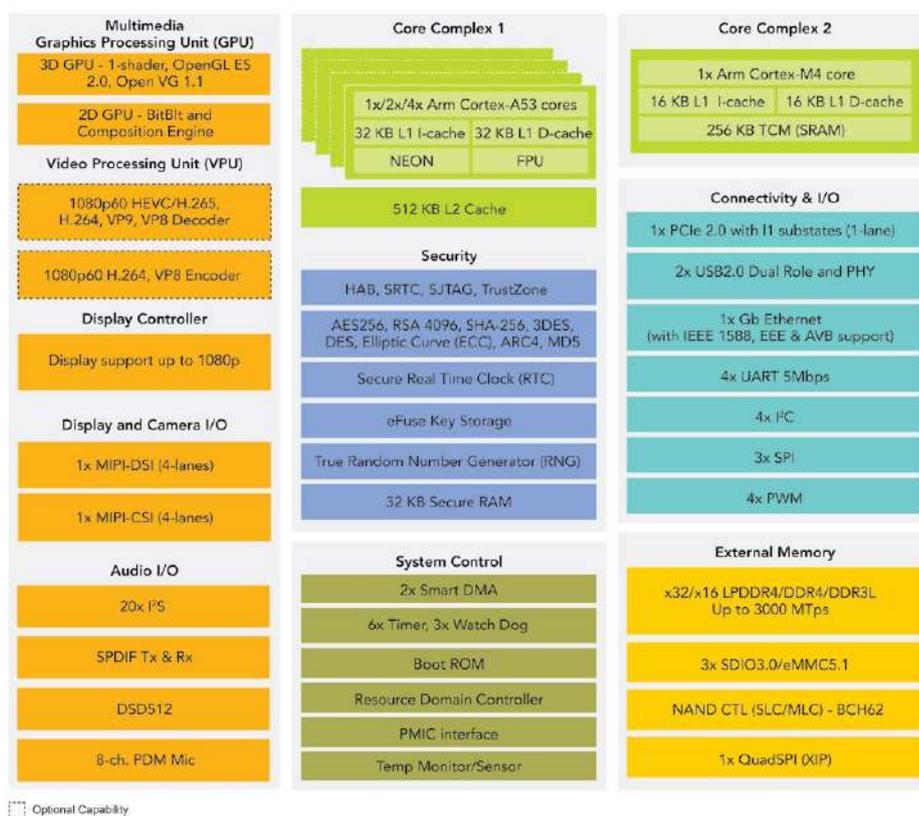


Schéma fonctionnel du processeur d'applications i.MX 8M Mini

LIENS

- [1] Foundries.io : <https://foundries.io/>
- [2] Arm SystemReady : <https://www.arm.com/architecture/system-architectures/systemready-certification-program>
- [3] EdgeLock SE050 : <https://bit.ly/EdgeLockSE050>
- [4] EdgeLock 2GO : <https://bit.ly/EdgeLock2GO>
- [5] i.MX 8M Mini : <https://bit.ly/iMX8MMini>
- [6] Portenta X8 SOM : <https://www.arduino.cc/pro/hardware/product/portenta-x8>

Nouveau projet Aduino ou électronique ?

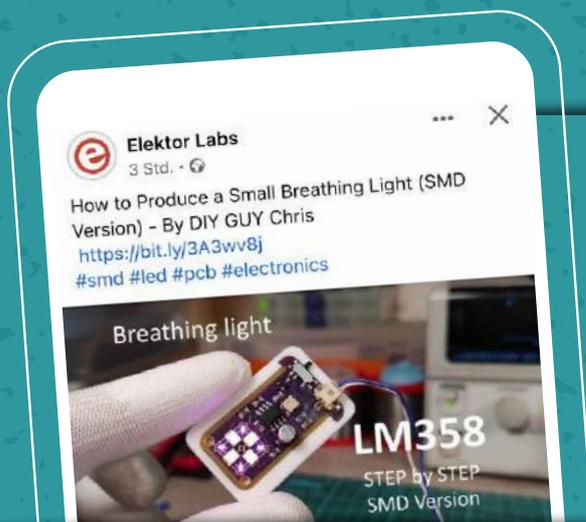
Partagez-le avec notre communauté !



Suivez-nous sur :



www.twitter.com/ElektorFR



www.instagram.com/elektorlabs



www.facebook.com/ElektorFR





▲
Daria Baradel (devant) et ses collègues d'Arduino.

rencontre avec Daria Baradel, la responsable de la production chez Arduino

Keith Jackson (Arduino)

Comment l'un des fabricants d'électronique les plus populaires au monde a-t-il réussi à relever les défis de la chaîne d'approvisionnement au cours des trois dernières années ? Grâce à une passion inébranlable pour l'innovation, une collaboration étroite avec des partenaires locaux et un facteur humain.

Keith Jackson : Bonjour, Daria. Avec tout ce que nous voyons dans les journaux sur les pénuries mondiales de composants électroniques cessant la production dans les usines du monde entier, cela a dû être une année difficile en tant que responsable de la production pour Arduino ?

Daria Baradel : Oui. Alors que tout le monde a entendu parler des pénuries dans l'industrie électronique affectant tout, des nouvelles voitures aux consoles de jeux, ce n'est pas seulement le manque de stock qui cause les plus grandes difficultés, ce sont les retards imprévisibles et les augmentations

constantes des coûts qui rendent le contrôle pratiquement impossible.

L'ensemble de l'équipe a dû adapter son approche et devenir plus flexible. Pour garder le contrôle, il faut de la prévisibilité et une gestion des risques, mais lorsque la prévisibilité n'existe plus, maintenir la production est un défi quotidien. N'importe quel jour, une livraison attendue peut être annulée, ce qui entraîne des arrêts de production à l'usine. Mais ce ne sont pas toujours les annulations qui entraînent des changements de dernière minute dans la programmation de la production. Il y a les cas exactement opposés où l'on a soudain la chance de recevoir une livraison aléatoire de composants qui n'était pas attendue avant 2023. Dans les cas où il s'agissait de la seule ligne en suspens dans une nomenclature, cela nous donne soudain l'occasion de remettre un modèle en production, de sorte que nous adaptions constamment les plans en fonction des disponibilités et des demandes. Nous avons dû revoir toutes les procédures standard et les adapter dans cette situation instable et imprévisible, en les rendant moins standard et plus flexibles afin d'être prêts à gérer chaque problème qui se serait présenté.

Keith : Pouvez-vous partager quelques exemples sur la façon dont vous et l'équipe avez adapté votre approche pour faire face à ces défis ?

Daria : Au début, j'essayais de prédire ce qui allait se passer, mais cela s'est avéré impossible. Ainsi, j'ai dû changer mon état d'esprit et adopter des procédures plus proactives. En tant qu'équipes, nous avons convenu qu'il était nécessaire d'avoir un meilleur contrôle sur les livraisons ; par conséquent, chaque vendredi, nous vérifions auprès de tous nos fournisseurs ce qui doit être livré la semaine prochaine. Pour les principaux composants, nous assurons ensuite un suivi quotidien.

Mais il ne s'agit pas seulement de la façon dont nous travaillons avec nos fournisseurs ; il est également devenu absolument nécessaire de travailler beaucoup plus étroitement avec l'équipe Arduino Hardware chargée de concevoir et de tester les produits. Avant de finaliser une nomenclature pour un nouveau produit ou même de confirmer les composants à monter sur une carte lors du prototypage, les deux équipes se rencontrent désormais pour évaluer la disponibilité et la fiabilité du fournisseur, dans le cadre du processus de conception.

La disponibilité d'un composant, ainsi que sa capacité, sont désormais prises en compte dans la conception finale. Tout n'est pas négatif pour autant, car l'équipe a désormais une bien meilleure compréhension des

cartes et des composants, puisqu'elle a été impliquée bien plus tôt dans le processus de conception. Ainsi, nous suivons tous chaque nouveau produit dès le concept initial.

En 2022, l'équipe de la chaîne d'approvisionnement, en collaboration avec l'équipe du matériel, a géré la validation de plus de 100 composants alternatifs et retravaillé la conception de 16 produits pour ajuster la nomenclature et pouvoir les produire.

Keith : Comment cela s'est passé avec les fournisseurs, étant donné qu'Arduino n'est qu'un des nombreux clients qui réclament des composants ?

Daria : Ironiquement, au cours des dernières années, j'avais investi beaucoup de temps pour automatiser autant de procédures d'achat que possible pour la commande de composants. Cette année, cependant, l'interaction humaine est revenue au premier plan. Il est essentiel de communiquer et de travailler avec les autres. Ce n'est que par des appels téléphoniques constants que nous pouvons suivre les derniers changements et les conséquences qui en découlent. Je serais probablement considérée comme une harceleuse avec les messages que je leur envoie constamment !

De nombreux de nos fournisseurs s'intéressent de près à l'ensemble du concept Arduino, et un bon nombre d'entre eux en sont passionnés. Par conséquent, il est clairement évident que nous avons obtenu un grand soutien de la part de certains fournisseurs et un mauvais soutien de la part d'autres ; mais dans l'ensemble, ceux nous ayant grandement aidés sont maintenant très proches, travaillant étroitement avec nous, s'intégrant dans une grande communauté.

Keith : Comment se passe la production et quelles sont vos perspectives pour l'avenir ?

Daria : Malgré tout ce qui s'est passé cette année, les volumes de production sont toujours en hausse de plus de 20 % par rapport à la même période en 2021. Ce résultat a été obtenu grâce à un effort transversal complet des équipes. L'équipe Hardware a énormément contribué en vérifiant et en approuvant littéralement des centaines de composants alternatifs, dans de nombreux cas en les montant et en les testant pour approbation le jour même afin de s'assurer que nous pouvions nous approvisionner en stock disponible. Ainsi, bien qu'il n'y ait pas de fin visible de la pénurie de composants, avec de nombreux délais de livraison s'étendant jusqu'à la fin de l'année 2023, je suis toujours convaincu que nous pouvons continuer à nous développer car, après tout, les entreprises sont faites par des humains. Les changements de mentalité





et de procédures que nous avons dû mettre en place cette année ne feront que nous rendre plus forts et plus efficaces à l'avenir.

Keith : Vous avez mentionné le test et l'approbation de composants alternatifs le jour même. Comment est-ce réalisable ?

Daria : Avoir nos producteurs à proximité des bureaux de R&D d'Arduino a été un énorme avantage pendant la pandémie de COVID et en ces temps de pénurie. Arduino a toujours été fier de fabriquer en Italie, avec toutes les cartes produites dans deux usines de la région du Piémont. En pratique, cela signifie que lorsque des problèmes surviennent, les ingénieurs sautent dans une voiture et se rendent immédiatement à l'usine voisine de Turin pour vérifier et changer les composants si nécessaire. Lorsque nous sommes informés de la disponibilité d'un autre composant, il est nécessaire de confirmer la commande le jour même, sinon le stock est pris par quelqu'un d'autre. Il faut être très rapide sur le terrain. Les ingénieurs se rendent donc à l'usine, testent le produit alternatif et nous donnent le feu vert, ce qui permet de passer la commande le jour même. Cela n'aurait clairement pas été possible si notre production avait été effectuée à l'étranger.

Keith : Arduino doit être l'une des rares entreprises d'électronique à utiliser encore des producteurs locaux pour ses cartes ?

Daria : Oui. La production locale était un choix délibéré, car Arduino a toujours adopté une approche positive durable, de l'impact environnemental et de la communauté locale. Nos objectifs de durabilité sont plus qu'un bout de papier : nous avons toujours essayé de garder une empreinte carbone aussi faible que possible et de limiter les déchets. Par exemple, nous donnons nos vieux stocks aux écoles locales, et la fabrication et l'emballage des produits sont effectués dans un rayon de 50 km autour de notre bureau de Turin. L'esprit communautaire est appliqué lors de la sélection de nos partenaires, ainsi que lors de la fabrication locale des produits. Le kitting et l'emballage sont réalisés par une entreprise locale qui emploie activement des personnes handicapées et leur donne l'occasion d'être indépendantes.

Keith : On dirait que vous êtes très fier de travailler chez Arduino. Mais à part gérer les pénuries quotidiennes de composants, qu'est-ce qui vous inspire ?

Daria : Je suis ingénieure de formation, avec deux masters en ingénierie de production et en gestion de l'ingénierie. En 2014, lorsque j'ai rejoint l'entreprise en tant que cheffe de projet, nous fabriquions entre 3 000 et 5 000 cartes par mois. Ce chiffre a été multiplié par plus de 50, ce qui signifie que j'ai maintenant une équipe complète de huit personnes pour gérer tout cela.

En dehors du travail, j'aime voyager et pratiquer de nombreux sports, en particulier l'escalade, la course à pied, la planche à voile et le beach-volley. Je trouve que le sport est le meilleur défouloir pour tous les stress de la journée. Mais si j'avais un rêve pour l'avenir, ce serait de posséder un jour une ferme, car j'adore les animaux, venant d'une famille qui élevait des vaches, des moutons et des chèvres. Je suis sûre que même à ce moment-là, j'aurai des Arduino partout dans la ferme pour la rendre aussi efficace que possible. ◀

220426-04 — VF : Maxime Valens

À propos de l'auteur

Keith Jackson travaille dans le département du marketing d'Arduino. Il est passionné par tout ce qui concerne Arduino, qui est plus qu'une entreprise ou une marque, mais toute une communauté diversifiée.

Des questions, des commentaires ?

Envoyez un courriel à l'auteur (k.jackson@arduino.cc) ou contactez Elektor (redaction@elektor.fr).

MicroPython Enters the World of Arduino

with Stuart Cording & Sebastian Romero

MicroPython has made it to the world of Arduino, providing the first significant alternative to programming in C and C++. So, what's all the fuss, how easy is it to use, and who can benefit from programming in this, for microcontrollers, relatively new language? Stuart Cording will speak with Sebastian Romero (Head of Content, Arduino) during our live webinar to find out more.

Join for free

www.elektor.com/webinar-MicroPython



les cartes de développement

passé, présent et futur

Mark Patrick (Mouser Electronics)

Ces dernières années, la signification du terme « carte de développement » s'est perdue dans une myriade d'autres termes désignant eux aussi des cartes matérielles utilisées à des fins de développement, parmi lesquels « cartes de démonstration (démo) », « kits d'évaluation » et « conceptions de référence ».

Dans cet article, nous expliquerons ce que l'on entend exactement par « carte de développement » et en quoi elle se distingue de son plus proche parent, l'ordinateur à carte unique (SBC). Nous retracerons ensuite l'évolution de la carte de développement depuis sa création jusqu'à aujourd'hui et examinerons enfin quelques tendances susceptibles de donner des indications sur son évolution future.

Qu'est-ce qu'une carte de développement ?

Il convient tout d'abord de s'accorder sur une définition claire de ce que l'on entend par « carte de développement » et de savoir en quoi celle-ci diffère d'un ordinateur à carte unique (SBC). Les cartes de développement sont généralement créées par les fabricants de microcontrôleurs, qui s'en servent de supports pour exposer les fonctionnalités de leurs produits (bien que cet usage s'étende aujourd'hui à d'autres types de composants que les microcontrôleurs). Un microcontrôleur est un circuit intégré qui comprend un processeur, de la RAM, un stockage flash ainsi que des entrées/sorties pour l'interfaçage avec le monde réel. C'est ni plus ni moins

un ordinateur miniature contenu dans un boîtier unique. Il fournit aux développeurs un moyen pratique de contrôler des composants externes tels que des lumières, des petits moteurs, etc. Un SBC fournit la même fonctionnalité, à la différence notable que le processeur, la RAM et le stockage sont chacun contenus dans des circuits intégrés distincts sur la carte. Des interfaces permettent alors de connecter le SBC à un clavier et/ou un écran.

Tandis que le microprocesseur d'un SBC nécessite un système d'exploitation pour fonctionner, le microcontrôleur est géré à l'aide d'un environnement de développement intégré (IDE) fourni par le fabricant. Il est de plus en plus fréquent de nos jours de voir des fabricants créer des cartes de développement non dans le but de servir de support de démonstration à un microcontrôleur, mais pour présenter les caractéristiques de capteurs ou d'autres circuits intégrés auxquels le microcontrôleur s'interface. Ce sont ces cartes que l'on appelle couramment « cartes de démonstration », « kits d'évaluation » ou – si elles ont été assemblées pour permettre la collecte de composants dans un but tangible – « conceptions de référence ».

Certaines cartes n'ont pas pour principale vocation de servir au développement de matériel, mais plutôt de fournir un accès aux données « réelles » dont ont besoin les développeurs de logiciels pour créer et affiner les algorithmes indispensables aux applications d'intelligence artificielle et d'apprentissage automatique. Ces types de cartes ne correspondent donc pas tout à fait à la définition et à l'usage originels d'une « carte de développement », mais ce terme s'emploie désormais pour désigner collectivement tout élément matériel utilisé dans le cadre du développement matériel et logiciel de nouveaux objets électroniques.

(Source : Shutterstock)

Le passé

C'est en 2006 qu'apparaît la carte de développement à microcontrôleur qui, la première, attira l'attention de la communauté des ingénieurs. Cette plateforme de prototypage, qui sera plus tard connue sous le nom d'Arduino [1] (**figure 1**), a été rapidement adoptée par une nouvelle catégorie de concepteurs en électronique constituée d'ingénieurs passionnés, d'amateurs et de bricoleurs dans l'âme. La popularité de l'Arduino a sans doute contribué au succès commercial des SBC et autres plateformes basées sur des microcontrôleurs qui l'ont suivie, dont la Beagleboard [2]. Apparue en 2008, cette carte offrait aux ingénieurs une plateforme de développement open source et peu coûteuse dont le développement était assuré par la communauté open source. Puis vint en 2012 le premier ordinateur monocarte, le Raspberry PI [3]. À l'instar de la Beagleboard, il s'agissait au départ de proposer une plateforme éducative bon marché que les étudiants pourraient utiliser pour apprendre à coder. Mais l'attrait pour le Raspberry PI a largement dépassé le cercle de l'enseignement et il a rapidement été adopté tant par les amateurs en électronique que par les ingénieurs professionnels.

Le présent

Les SBC se divisent aujourd'hui en deux grandes catégories, à savoir les SBC propriétaires et les open source. Les SBC propriétaires sont généralement conçus pour être utilisés dans des applications finales et ont été soumis au même type de test et d'assurance qualité que les autres produits finaux. Ils sont soit intégrés à des équipements électroniques, soit installés dans une configuration en racks. Au contraire, les SBC open source offrent aux utilisateurs la possibilité de modifier librement la conception et la disposition de leur matériel. En ayant accès au code source, les utilisateurs peuvent rapidement et facilement apprendre comment fonctionnent le logiciel et le matériel et adapter la conception à leurs besoins.

Aujourd'hui, de nombreux types de processeurs équipent les cartes de développement et les SBC : des processeurs X86 pour ordinateurs personnels traditionnels (AMD et Intel) aux processeurs ARM utilisés dans les applications industrielles et mobiles. Les systèmes d'exploitation les plus utilisés sur les SBC sont Linux et ses dérivés (Ubuntu, Fedora, Debian, etc.), mais aussi Android et Windows CE. De leur côté, les cartes de développement à microcontrôleur ne nécessitent pas de système d'exploitation, mais sont programmées à l'aide d'un IDE fourni par le fabricant. Les cartes de développement et les SBC ont tous deux évolué pour inclure des fonctionnalités comme la connectivité sans fil (wifi, Bluetooth) et les interfaces audio et vidéo les plus récentes. Ce faisant, certains SBC présentent aujourd'hui des fonctionnalités équivalentes à celles de nombreux ordinateurs personnels et tablettes.

Le futur : la carte de développement devient un produit final

Les fabricants avaient pris pour habitude de créer des outils de développement dans l'unique but de servir de support marketing à leurs microcontrôleurs. C'est ce qu'on appelait dans l'industrie le « design-in ». En effet, la carte de développement faisait gagner beaucoup de temps aux ingénieurs de conception désireux de tester le microcontrôleur dans leur laboratoire et d'en étudier les fonctionnalités. En leur facilitant ainsi la tâche, les fabricants pensaient que les concepteurs seraient également plus enclins à choisir leur microcontrôleur (avec naturellement les pièces auxiliaires qui s'y rapportent) pour le prototypage initial de leur produit et, *in fine*, pour la production en masse, ce qui rapporterait évidemment un volume de commande considérable au fabricant. Cette démarche tenait du bon sens dans le cas de produits dont les pièces présentent très peu de différences sur le plan des spécifications techniques d'un fournisseur à l'autre. Or, du côté des fabricants, cette approche a été, pour ainsi dire, victime de son propre succès. Ils se sont bientôt rendu compte qu'il leur fallait continuer à réduire sans cesse la charge de travail que représente l'essai de leur produit pour l'ingénieur chargé de la sélection. C'est alors que la carte de développement est devenue le principal élément différenciateur, d'autant plus lorsqu'il s'agit de produits globalement assez similaires à ceux de la concurrence.

C'en est au point que, même pour des microcontrôleurs présentant un avantage incontestable par rapport à la concurrence (sur le plan de la puissance ou de la vitesse, par exemple), les ingénieurs concepteurs continuent d'avoir des exigences relatives à la carte de développement, notamment des fonctionnalités *plug'n'play*.

Les fabricants ont amélioré leur proposition de valeur en proposant des conceptions de référence qui se composent d'un microcontrôleur et d'autres circuits

Figure 1. Carte de développement à microcontrôleur Arduino. (Source : Shutterstock)





▲
Figure 2. La HSP3.0 de Maxim Integrated.

intégrés (généralement des capteurs). Initialement, ces conceptions étaient destinées à montrer comment interconnecter les appareils pour émuler la fonctionnalité électrique d'un produit final et peu d'attention était accordée au format, à la taille de la conception ou à la facilité de fabrication. Cependant, certains fabricants n'ont pas hésité à franchir le pas en proposant des conceptions de référence qui sont devenues des prototypes de produits à part entière et même des produits achevés.

Un exemple de cette évolution sont les conceptions de référence de la plateforme de capteurs de santé (HSP) [4] de Maxim Integrated (qui fait désormais partie d'Analog Devices). Dans leur version initiale, ces conceptions de référence se composaient d'une petite carte de développement équipée d'une série de capteurs (température, pression, accéléromètre, biopotential, etc.) adaptés à des applications de forme et santé et que l'on peut configurer à l'aide d'un microcontrôleur. Le format de leurs successeurs, les HSP2.0 et HSP3.0, leur permettait d'être portés au poignet et ressemblaient de fait en tout point aux autres *wearables* disponibles sur le marché (figure 2).

Les développeurs avaient ainsi la possibilité d'évaluer la fonctionnalité de leurs capteurs dans des situations réelles. Un autre avantage non négligeable de ces conceptions est qu'elles permettaient aussi aux développeurs de logiciels d'accéder gratuitement aux relevés des capteurs, ce qui est rarement le cas avec d'autres *wearables* de ce type. Or, l'accès à ces informa-

▼
Figure 3. Un automate programmable industriel.
(Source : Shutterstock)



tions permet de développer des algorithmes d'apprentissage automatique et d'intelligence artificielle qui apportent de la valeur ajoutée à l'application.

En montrant comment leur matériel facilite l'accès à ces données, Maxim entend bien sûr inciter les développeurs de produits à opter pour certains circuits intégrés (voir tous) présents dans sa conception de référence afin de les intégrer à leurs propres produits. Fort de cette expérience, Maxim a ensuite développé le bracelet *MAX HEALTH BAND* [5] et la sangle de poitrine *MAX ECG MONITOR* [6]. Ces deux produits ont été spécialement conçus et fabriqués comme des *wearables* de forme et santé aboutis. Bien qu'ils ne soient pas destinés à être vendus directement aux consommateurs, ces produits pourraient très bien intéresser des entreprises prêtes à conclure un contrat de licence de marque.

Le fait de proposer ainsi un produit entièrement fonctionnel, qui ne nécessite plus aucun travail de développement, semble être un bon moyen d'élargir et de renouveler sa clientèle commerciale non technique. La Thingy:91 [7] de Nordic Semiconductor est un autre exemple de plateforme de développement où le matériel est devenu presque accessoire par rapport à la capacité de fournir aux développeurs un accès aux données dont ils ont besoin pour développer les logiciels et les algorithmes qui donnent sa valeur intrinsèque au matériel. Bien entendu, le but est de les faire opter pour cette même plateforme pour de nouvelles conceptions de produits qui exploitent les mêmes algorithmes. Sans doute cette technique fera-t-elle bientôt des émules parmi les autres fabricants.

L'utilisation accrue des cartes de développement dans les produits industriels

Il est de plus en plus courant de voir des cartes de développement et des SBC adaptés pour être utilisés dans des produits commerciaux. Une autre tendance est leur utilisation dans des applications à faible volume, mais à plus grande valeur, c'est-à-dire des produits finaux industriels comme des automates programmables industriels (API, ou PLC en anglais) (figure 3), qui sont soumis à des normes plus strictes que leurs équivalents commerciaux.

Les cartes de test pour applications industrielles

De nombreux SBC aujourd'hui disponibles sont intrinsèquement devenus des conceptions abouties, car ils sont composés de pièces qui ont été initialement développées pour être utilisées dans des produits finaux, ce qui signifie qu'elles ont déjà subi des tests et que leur qualité est garantie. À cela s'ajoute le fait que les conceptions open source sont constamment revues par une armée de concepteurs et de programmeurs compétents qui mettent à jour et évaluent les cartes et les logiciels qu'ils utilisent.

Les tests des cartes SBC sont désormais effectués par des entreprises de conception et de fabrication de

haute qualité. Elles sont par conséquent soumises à des contrôles de qualité aussi rigoureux que ceux de tout autre produit final, ce qui leur permet même d'obtenir les certifications CE ou FCC. Ce flux de test peut facilement être étendu pour répondre aux exigences des produits industriels.

Or, les cartes de développement à microcontrôleur fournies par des fabricants ou des tiers, bien qu'elles soient habituellement adaptées à une utilisation dans des produits commerciaux, ne sont quant à elles généralement pas soumises à des tests aussi rigoureux que le sont ceux des produits industriels. De ce fait, les fabricants ne les recommandent pas pour une utilisation immédiate (sous leur forme actuelle) dans ces applications.

Même si certaines cartes incluent des composants de qualité industrielle, la plupart répondent seulement aux normes de qualité commerciale, parce qu'elles ne sont conçues que pour fonctionner à température ambiante. Les prototypes de cartes de développement subissent généralement des tests à température ambiante pendant plusieurs jours ou semaines, mais cela varie selon le fabricant, car il n'y a pas de normes établies en la matière. La principale exigence de qualité pour les fabricants est que leurs cartes fonctionnent de manière fiable à température ambiante. Par conséquent, les acheteurs doivent être conscients qu'il est peu probable qu'elles aient été testées dans des conditions de température ou d'humidité extrêmes, de même qu'elles n'ont pas subi de tests de résistance aux contraintes associées à des vibrations ou à des chocs intenses.

Lors du choix d'une carte de développement pour une application industrielle, il est donc essentiel d'opter pour le moindre risque, notamment en vérifiant que les composants de la carte sont de la bonne classe de température. Il est également judicieux de tester simultanément plusieurs cartes à haute température pendant plusieurs jours. De même, si vous envisagez d'utiliser une carte de développement dans un produit susceptible de se retrouver dans un environnement très humide, il convient de la tester dans des conditions comparables. Enfin, les cartes destinées à être utilisées dans une application à fortes vibrations doivent être montées dans un cadre d'essai afin de tester leur résistance aux vibrations.

Conclusion

Les SBC et les cartes de développement à microcontrôleur offrent aux petites entreprises un moyen pratique de commercialiser rapidement leurs conceptions sans avoir à développer de nouveau matériel. Cela leur permet de se concentrer sur l'innovation logicielle et – de plus en plus – sur le développement d'algorithmes d'apprentissage automatique et d'intelligence artificielle. Les SBC et les cartes de développement peuvent aujourd'hui être utilisées dans bien plus d'applications que ce pour quoi elles étaient initialement destinées et leur importance dans l'histoire récente de l'industrie électronique n'est plus à démontrer. Elles ne cessent de gagner en puissance, en intelligence et en réactivité, tout en restant des produits à la portée tant des ingénieurs professionnels que des passionnés d'électronique. ◀

220597-04



À propos de l'auteur

En tant que directeur marketing technique de Mouser Electronics pour la région EMEA, Mark Patrick est responsable de la création et de la diffusion du contenu technique - un contenu essentiel à la stratégie de Mouser visant à soutenir, informer et inspirer son public d'ingénieurs.

Avant de diriger l'équipe de marketing technique, Patrick faisait partie de l'équipe de marketing achat de la région EMEA et jouait un rôle essentiel dans l'établissement et le développement des relations avec

les principaux partenaires et fournisseurs. En plus d'avoir occupé divers postes dans les départements techniques et marketing, Patrick a travaillé pendant huit ans chez Texas Instruments, dans les services support et ventes techniques.

Ingénieur expérimenté, passionné de synthétiseurs vintage et de motos, il n'hésite pas à les réparer. Patrick est titulaire d'un diplôme d'ingénieur en électronique avec mention très bien de l'université de Coventry.

LIENS

- [1] Produits présentés par Arduino : <https://elektor.link/MouserArduino>
- [2] Produits présentés par BeagleBoard : <https://elektor.link/MouserBeagleBoard>
- [3] Produits présentés par Raspberry Pi : <https://elektor.link/MouserRaspberryPi>
- [4] Produits présentés par Maxim Integrated : <https://elektor.link/MouserMaxim>
- [5] MAX HEALTH BAND : <https://elektor.link/MouserMaxHealthBand>
- [6] MAX-ECG-MONITOR : <https://elektor.link/MouserMaxECGMonitor>
- [7] Nordic Semiconductor Thingy:91™ Multisensor Prototyping Kit : <https://elektor.link/MouserThingy91>

Édition bonus dévoilée !

Disponible la semaine 01

▲
Figure 1
atelier «

l'
qu



...tant un logiciel permettant de créer des symphonie par le seul fait de jouer ensemble !



Édition bonus dévoilée !

Les
rent
cons
de r
rez
tens
noti
la n
de
des
Ch
où
LE
po
nic
Vo
re
Po
tic
vo

Disponible la semaine 01



our la



Figure

Figure

Édition bonus dévoilée !

Disponible la semaine 01



une grande variété de timbres venant de la puce. La correspond à la note que nous recommandons



Édition bonus dévoilée !

Disponible la semaine 01



1 2
100

Édition bonus dévoilée !

Disponible la semaine 01

Figure 9.
des bro





Édition bonus dévoilée !

Sel
mer
pro
trou

Co

Not
par
une
ave
pro

Disponible la semaine 01



Mettez la main

Édition bonus
dévoilée !

Disponible la semaine 01

B
C

La
rob
Arc
nou
les
Bra
de c
tâch
des
suiv
pan
men
offre
évol
un n
écran
teurs
totale

www

Ardu



projets de n'importe où dans le monde.

www.elektor.fr/19351

Édition bonus dévoilée !

Disponible la semaine 01

UNO, et commander le kit de
synthé commandé par la carte UNO !

www.elektor.fr/20330

Arduino Sensor Kit Base

www.elektor.fr/19944



à la possibilité d'exécuter des app.
d'Edge Computing (IA).
www.elektor.fr/19936

Supporting **Arduino Resellers**

Cette édition spéciale Arduino du magazine Elektor a été rendue réalisable grâce au soutien de ces membres de la communauté des revendeurs Arduino.

N'hésitez pas à les consulter pour tous vos besoins liés à l'Arduino !

GOTRON
AALST GENT HASSELT



www.gotron.be

HELLAS
digital



www.hellasdigital.gr

TINYTRONICS



www.tinytronics.nl

 **Paradisetronic.com**



www.paradisetronic.com

 **Techni Science.**



www.techniscience.com

 **reichelt**
elektronik – The best part of your project



www.reichelt.com/arduino

 **WHADDA**
MADE BY VELLEMAN



www.whadda.com

 **KUBII**



www.kubii.fr

GO TRONIC
ROBOTIQUE ET COMPOSANTS ÉLECTRONIQUES



www.gotronic.fr