(Développeur en conception, logiciels et plus)

simulation IdO simplifiée avec Wokwi

le développeur Uri Shaked parle de design, de logiciels et d'autres choses encore

Questions de Roberto Armani (Elektor)

Vous souhaitez simuler des processeurs et des cartes ? Découvrez Wokwi, un simulateur open-source innovant pour les systèmes embarqués et les appareils IdO. Son auteur, Uri Shaked, parle de la solution, ainsi que de son parcours et de ses centres d'intérêts.

Elektor : tout d'abord, merci pour votre temps, Uri. Pourriez-vous nous parler de vous, de votre parcours ? Où vous trouvez-vous actuellement ?
Uri Shaked : je suis ingénieur en logiciel, basé en Israël. Enfant, je démontais toujours mes jouets, curieux de savoir comment ils fonctionnaient. En CE2, j'ai cassé mon ordinateur PC/XT. J'avais envie d'apprendre à le réparer et ma tante m'a acheté un livre sur MS-DOS. Les deux derniers chapitres étaient une brève introduction à QBasic. J'ai été séduit. À partir de là, j'ai continué à programmer tous les jours.

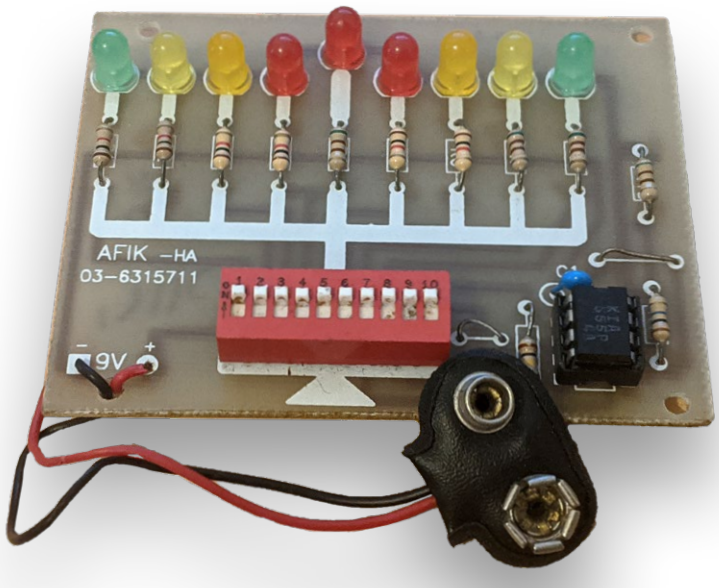


Figure 1. Mon projet de Menorah de Hanoukka, que j'ai assemblé en CM1 et qui est toujours fonctionnel !

Elektor : qu'est-ce qui a déclenché votre intérêt pour l'électronique ? S'agissait-il d'un projet spécifique, d'un cours particulier, de la passion d'un parent pour l'électronique ou peut-être d'un professeur ?
Uri Shaked : à l'école primaire, nous avions un cours hebdomadaire d'électronique. Il s'agissait d'un cours très pratique : nous apprenions à souder et à assembler des circuits imprimés que le professeur nous donnait, tout en apprenant les bases. Je me souviens encore à quel point j'aimais ces cours, et du sentiment d'accomplissement que je ressentais lorsque je tenais un projet terminé dans ma main (**figure 1**). En effet, ces leçons m'ont apporté les connaissances et l'expérience fondamentales et m'ont transmis la passion.

Elektor : personne n'oublie son « premier amour ». Parlez-nous de votre premier projet fonctionnel en électronique. Sur quoi avez-vous travaillé et pourquoi ?
Uri Shaked : lorsque j'étais en quatrième, je passais beaucoup de temps sur les serveurs de discussion IRC. Un jour, un ami a partagé un lien expliquant comment construire un système de haut-parleurs à partir de vieux disques durs en connectant le moteur de l'actionneur directement à un amplificateur. L'idée m'a fasciné et j'ai décidé de construire le mien. La qualité du son n'était pas parfaite, mais après quelques essais et erreurs, j'ai découvert qu'en plaçant un pot lourd sur le disque dur, la qualité du son et le volume s'en trouvaient grandement améliorés (le son résonnait à travers le corps du pot). Je me suis retrouvé avec une tour de pots dans le salon, jouant le vinyle de Pink Floyd de ma mère à travers cette configura-

tion. Inutile de préciser qu'elle n'était pas contente. Au fil des ans, les disques durs ont continué à être un thème dans mes projets, par exemple dans mon premier projet de microcontrôleur, un affichage de persistance de la vision à base d'un PIC18 monté sur disque dur ou, comme vous pouvez le voir dans la **figure 2**, l'un de mes premiers projets Arduino, un xylophone [1] alimenté par un disque dur qui jouait des chansons sur des bouteilles de bière recyclées.

Elektor : les ingénieurs et les programmeurs apprennent beaucoup de leurs erreurs. Certains de vos projets ont-ils tourné au désastre ? Avez-vous quelque chose à dire là-dessus ?
Uri Shaked : sans aucun doute mon projet GeekCon. La GeekCon est un événement qui se déroule sur un week-end et au cours duquel un groupe d'intellos se réunit pour construire des objets inutiles mais géniaux. Le but est d'échouer, comme le dit la devise, « si votre projet n'a pas échoué, c'est que vous n'avez pas visé assez haut ». En 2018, nous avons essayé de construire un robot qui joue de la trompette [2]. Nous avons utilisé un gant en latex rempli d'eau pour fabriquer des lèvres artificielles, un pot de pâtes pour un régulateur de débit d'air, et des servomoteurs à engrenages métalliques très résistants pour contrôler les doigts artificiels (**figure 3**). Une minute avant la présentation du projet final, il y a eu une fuite et deux des servomoteurs ont lâché. Un échec cuisant. Je n'aime pas échouer, alors je suis rentré chez moi et j'ai passé le mois suivant à essayer de réparer le robot. J'ai promis de l'apporter à l'événement annuel des développeurs de Chrome, le Chrome Dev Summit. J'ai fini par trouver comment fabriquer un mécanisme robuste pour les doigts, mais le mécanisme des lèvres était si délicat que je n'arrivais pas à le faire fonctionner. J'ai fini par « tricher », en utilisant un Raspberry Pi qui jouait des échantillons de trompette préenregistrés à travers l'embouchure. Mais au moins, le résultat final sonnait bien, et mon robot a beaucoup amusé les participants du Chrome Dev Summit [3].

Elektor : quelqu'un a dit un jour qu'« un bureau propre est le signe d'un esprit malade ». Auriez-vous le courage de nous montrer une photo de votre lieu de travail ?
Uri Shaked : la **figure 4** représente ce à quoi ressemblait mon bureau lorsque j'ai essayé de construire une version réelle du jeu « Offline Dinosaur » [4] de Google Chrome. Mon vrai problème était le stockage des composants. J'avais plus de 4 600 items, conservés dans 40 boîtes de rangement différentes, suivis dans plusieurs feuilles de calcul Excel, consommant un espace précieux dans mon petit appartement. L'année dernière, nous avons déménagé, et j'ai décidé de donner presque tout à un maker space

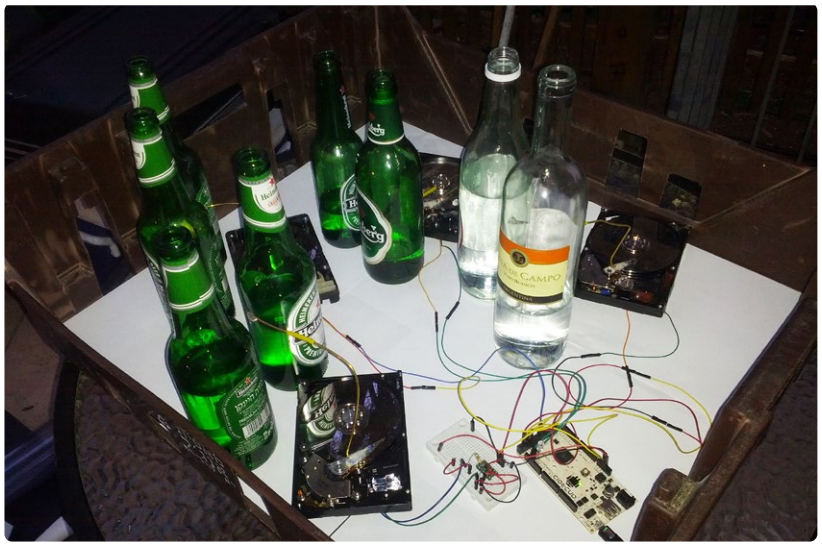


Figure 2. Le Xylophone alimenté par un disque dur.

Figure 3. Mon ami Avi avec notre robot trompette, quelques secondes avant qu'il n'explose !

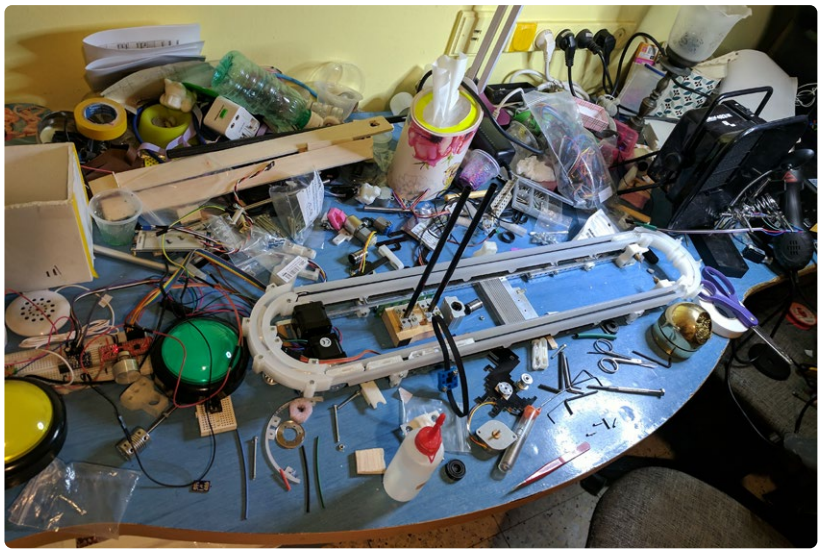


Figure 4. Mon espace de travail bien rangé…

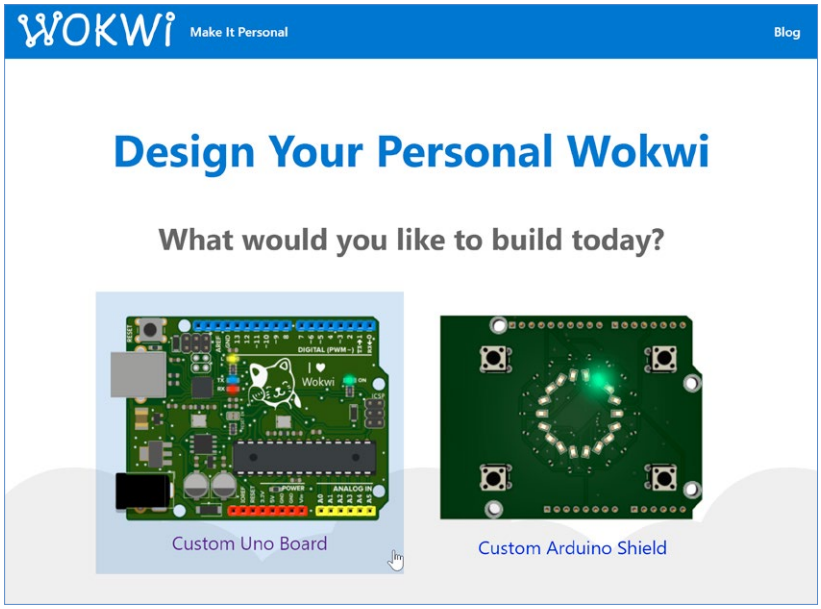


Figure 5. L'ancienne page d'accueil de Wokwi.

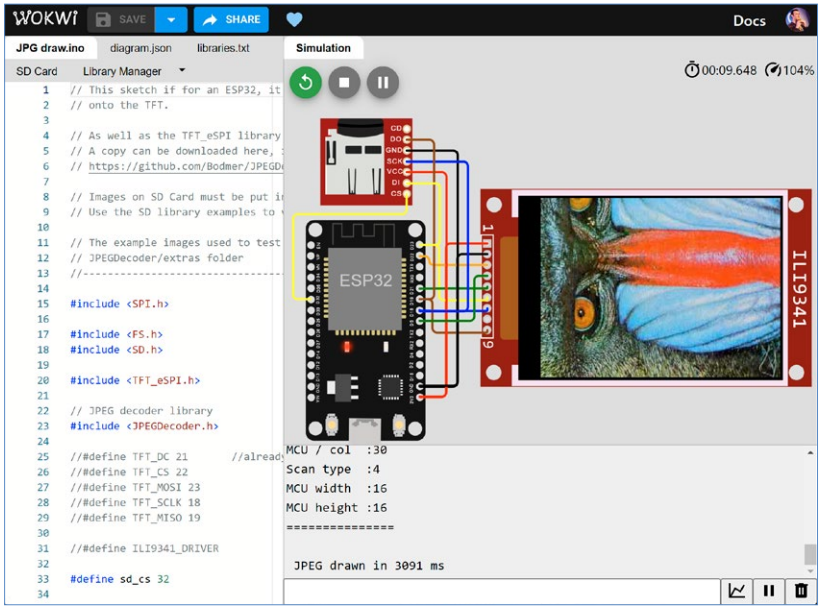


Figure 6. Wokwi aujourd'hui.

local. Aujourd'hui, j'utilise principalement des pièces virtuelles. J'ai encore quelques Raspberry Pi, un analyseur logique Saleae, et une petite collection de microcontrôleurs (ESP32s, STM32s, Raspberry Pi Picos, et Arduino), me permettant de comparer le comportement de la puce physique avec celui de la puce simulée.

Elektor : si vous deviez suggérer aux débutants quelques « trucs et astuces » sur l'aménagement d'une bonne espace de travail, que diriez-vous ?
Uri Shaked : ayez une bonne alimentation électrique !

Elektor : passons à Wowki. Qu'est-ce que c'est ? D'où vient le nom ? Qu'est-ce qui vous a amené à le créer ?
Uri Shaked : Wokwi est un simulateur pour les systèmes embarqués et les appareils IdO. Il existe une version en ligne gratuite, ainsi qu'un plugin Visual Studio Code. Wokwi peut simuler un ESP32, Raspberry Pi Pico, STM32, Arduino, et un tas d'autres microcontrôleurs, ainsi qu'une grande variété de dispositifs d'entrée et de sortie : écrans LCD, capteurs, moteurs, LED, boutons, haut-parleurs, potentiomètres, et plus encore. Comme vous pouvez le voir sur la **figure 5**, Wokwi a débuté comme un service qui vous permet de fabriquer des cartes Arduino personnalisées avec une interface facile à glisser-déposer. Commencez à partir d'un modèle de carte, ajoutez des périphériques (LED, capteurs, etc), placez-les dessus, et nous transformons votre conception en un circuit imprimé, nous le fabriquons et l'assemblons, et nous vous l'expédions. J'ai commencé à bloguer sur Arduino pour aider à faire passer le mot, et je voulais vraiment inclure des démonstrations interactives et en direct dans mes articles. Je n'ai pas trouvé de bonne solution pour partager facilement des projets Arduino et les simuler sur le web. J'ai donc commencé à travailler sur une bibliothèque JavaScript open-source capable de simuler le microcontrôleur ATmega328P. Le reste appartient à l'histoire. Sur la **figure 6**, vous pouvez voir à quoi cela ressemble aujourd'hui. Lorsque j'ai choisi le nom de Wokwi, j'ai cherché un mot court, facile à prononcer et qui n'avait pas encore de signification.

Elektor : qui s'en sert ? Son utilisation est-elle totalement gratuite ?
Uri Shaked : environ 55 % des utilisateurs sont des amateurs, 35 % sont des étudiants et des enseignants (universités et lycées) et 10 % sont des professionnels en microprogrammation. Certains cas d'utilisation populaires sont le prototypage de systèmes IdO (MQTT, Blynk, Thingsboard, IBM Cloud), les cours et ateliers en ligne (par exemple le cours *Making Embedded Systems*), et nous constatons un intérêt croissant de la part de la communauté Embedded Rust. Wokwi a une version gratuite et une version payante (*The Club*). La version gratuite est très performante, et possède les mêmes capacités de simulation que la version payante. La version payante ajoute le téléchargement de fichiers et de bibliothèques, le réseau local pour l'ESP32, et la possibilité de sauvegarder des projets privés (non listés).

Elektor : quels sont les éléments ou les aspects du développement et de la conduite du projet Wokwi qui se sont avérés les plus difficiles ?
Uri Shaked : Wokwi vise à simplifier les systèmes complexes et à offrir une excellente expérience à l'utilisateur. Au lieu de télécharger et d'installer des compilateurs, des pilotes de système d'exploitation, de se battre avec les autorisations, les câbles USB défectueux et les connexions bancales, il vous suffit de cliquer sur un bouton. Votre code se compile « comme par magie » et vous pouvez voir le résultat dans votre navigateur web. Pour créer ce type d'expé-

rience « ça marche tout seul », nous devons gérer nous-mêmes une grande partie des complexités. Nous configurons la chaîne d'outils du compilateur dans le cloud pour de multiples plateformes (par exemple Rust, Arduino, ESP-IDF, verilog), nous nous assurons qu'elle fonctionne rapidement à grance échelle (nous exécutons environ 2,3 millions de compilations par mois), et nous faisons la chasse aux bogues mystérieux. Par exemple, le mois dernier, j'ai passé une journée à comprendre pourquoi l'utilisation d'une bibliothèque spécifique dans un projet Arduino provoquait le blocage du serveur. Il s'est avéré qu'il y avait un problème avec l'Arduino CLI [5] qui provoquait un blocage lors de l'installation de cette bibliothèque. Offrir une bonne expérience utilisateur signifie également aider les utilisateurs à comprendre pourquoi le code ne fonctionne pas comme ils l'attendent. Je pense que nous avons encore un long chemin à parcourir, mais des fonctionnalités telles que l'analyseur logique virtuel, et la fonction Pin Function Dump récemment ajoutée sont absolument un pas en avant.

Elektor : avez-vous un projet de simulation préféré ? Parlez-nous en.
Uri Shaked : difficile de choisir, mais si je devais n'en retenir qu'un, ce serait 32 Servos Dancing [6], également visible sur la **figure 7**, qui est un Arduino Mega contrôlant 32 servos et animant leurs bras. C'est plutôt un projet artistique qu'utile et c'est ce que j'aime dedans. Le construire dans la vraie vie nécessiterait une alimentation puissante et de dépenser quelques centaines d'euros en pièces détachées, mais, dans le simulateur, c'est une évidence. Il suffit de cliquer sur *Play* pour le voir à l'œuvre. Pour pimenter les choses, même le diagramme de ce projet a été généré par le code Arduino. Jetez un coup d'œil à la fonction `GenerateDiagram()`, qui utilise des fonctions trigonométriques pour placer les servos. Elle imprime un fichier JSON, que vous pouvez ensuite coller dans *diagram.json* dans votre projet Wokwi, fournissant à la fois le circuit et sa représentation visuelle. C'est ce que j'appelle de l'ingénierie innovante ! En parlant de créativité des utilisateurs : Wokwi fournit une *Custom Chips API* [7], vous permettant de coder de nouveaux modèles de simulation pour des pièces qui n'existent pas encore dans Wokwi. L'API fournit une interface `framebuffer`, qui vous permet de créer vos propres écrans (LCD, E-Paper, etc.). Ce que je n'avais pas prévu, c'est que les utilisateurs créeront leur propre oscilloscope en utilisant l'interface framebuffer, comme le montre la **figure 8**.

Elektor : l'ESP32 est un module assez complexe en soi. Parlez-nous de l'intégration de l'ESP32 dans votre projet. Cela a-t-il été difficile ?
Uri Shaked : wow, c'était une question difficile. L'ESP32 utilise le jeu d'instructions Xtensa, qui en contient plus de 250. La première étape a consisté à écrire du code pour décoder et simuler chacune d'entre elles, ce qui a représenté environ 4 500 lignes de code. Ensuite, j'ai créé un pont GDB (pour pouvoir déboguer le code exécuté dans le simulateur), et j'ai commencé à travailler sur les périphériques de la puce. Mon objectif initial était de passer la ROM et le bootloader, et d'exécuter l'application *Hello World* à partir de l'ESP-IDF. Après avoir implémenté l'UART, la SPI, les timers (TIMG), et quelques autres périphériques obscurs (DPORT, EFUSE), j'ai finalement pu obtenir la première sortie utilisable de la console série du simulateur. Le plus grand défi, cependant, était de simuler le Wi-Fi. Contrairement aux autres parties de la puce, le périphérique Wi-Fi n'est pas du tout documenté, et le code source des pilotes n'est pas disponible. Il a fallu des semaines de rétro-ingénierie intensive [8] pour découvrir les registres qui contrôlent la radio Wi-Fi et pour les simuler correctement. L'ESP32 est en train de devenir la puce la plus populaire sur Wokwi, et Espressif, la société à l'origine de l'ESP32, soutient fortement le projet. Leurs ingénieurs font avancer le support de Rust, en utilisant Wokwi dans leurs formations et ateliers, et font même connaître le simulateur dans des conférences professionnelles.

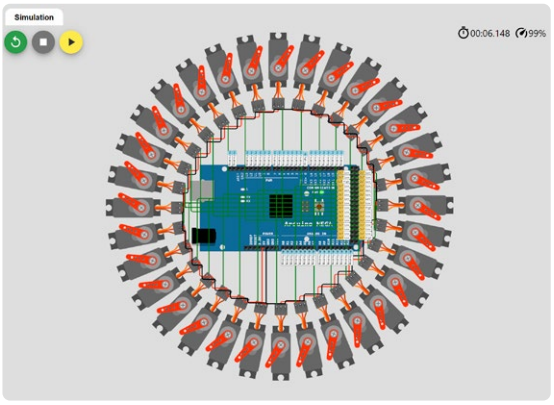


Figure 7. 32 servos dansants, pilotés par une carte Arduino Mega.

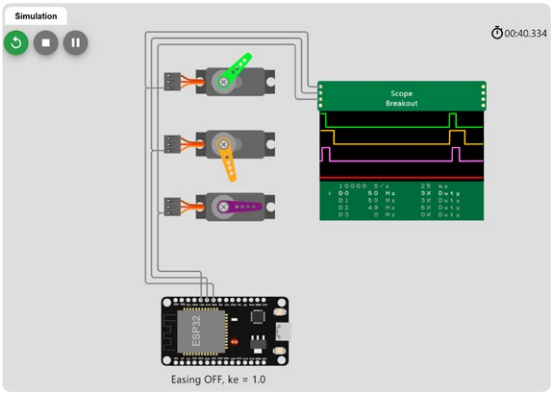


Figure 8. La puce Scope, montrant les signaux des servos en temps réel.

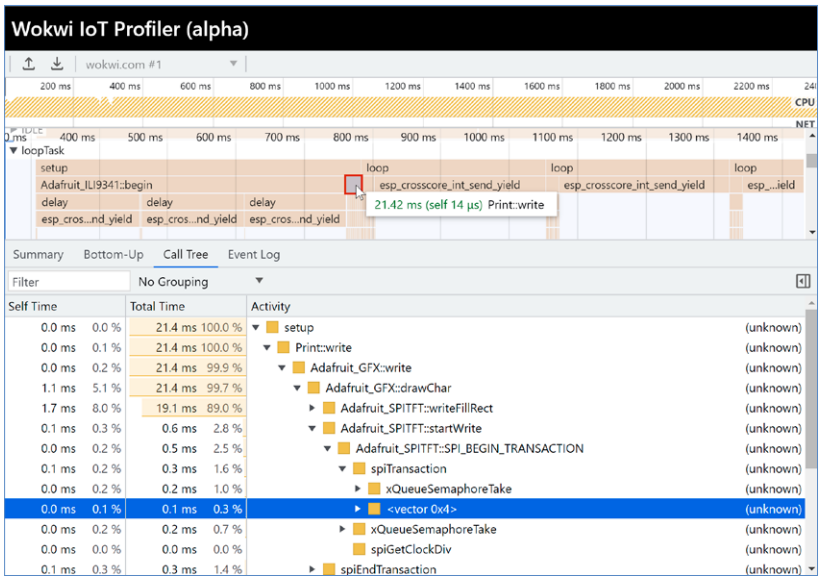
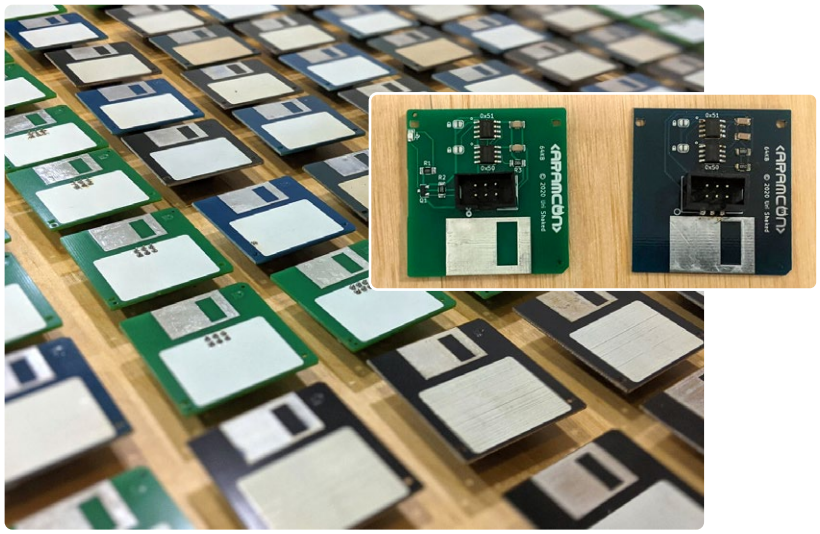


Figure 9. Wokwi Embedded Profiler (profiler embarqué).



▲

Figure 10. Les disquettes, avec un gros plan sur l'électronique (deux EEPROM, des cavaliers de soudure de verrouillage d'écriture, et une LED d'indication).

Elektor : parlez-nous de la réaction des utilisateurs à Wowki. Que faites-vous de leurs commentaires ?
Uri Shaked : nos utilisateurs sont généralement satisfaits du simulateur, qui leur fait gagner beaucoup de temps. Mais ils demandent toujours plus de fonctions, bien plus que ce que notre petite équipe peut gérer. Pour nous aider à nous concentrer sur ce qui est vraiment important pour les utilisateurs, j'ai ouvert la page « Votez pour de nouvelles fonctions » [9]. Tout le monde peut suggérer de nouvelles fonctions, et les gens peuvent voter avec leur argent pour les fonctions qui leur importent. Cela nous aide à établir des priorités, et c'est aussi un excellent moyen pour les utilisateurs de savoir ce qui les attend.

Elektor : où aimeriez-vous voir Wokwi dans les six à douze prochains mois ?
Uri Shaked : j'aimerais que Wokwi gagne en popularité auprès des utilisateurs professionnels. Au début de l'année, j'ai lancé Wokwi pour Visual Studio Code. Il vous permet d'exécuter la simulation directement dans votre IDE et de travailler beaucoup plus rapidement sur votre code. Je vois déjà des développeurs Rust l'utiliser pour améliorer leur productivité. Wokwi peut également s'intégrer au débogueur de VS Code et, contrairement au matériel réel, vous pouvez définir un nombre illimité de points d'arrêt. Je travaille actuellement sur un nouveau profileur embarqué, voir **figure 9**. Wokwi trace votre code pendant qu'il s'exécute dans le simulateur. Vous pouvez voir quelles fonctions sont appelées, dans quel ordre, et combien de temps le microcontrôleur passe dans chaque fonction. Je prévois également d'étendre la gamme ESP32 (avec ESP32-H2, ESP32-P4), d'ajouter de nouveaux dispositifs STM32, et peut-être d'entrer en contact avec d'autres fournisseurs de silicium et d'ajouter leurs microcontrôleurs également. Enfin, j'aimerais voir Wokwi utilisé dans des environnements CI (intégration continue). L'idée est de tester votre code embarqué à chaque validation. C'est beaucoup plus facile à mettre en place et à faire évoluer avec Wokwi, comparé au maintien d'une installation matérielle physique. J'utilise déjà une configuration CI en interne pour m'assurer que les composants critiques, tels que l'ESP32 Wi-Fi, ne se cassent pas avec les mises à jour du code et les nouvelles versions de l'ESP-IDF, et je travaille maintenant sur une CLI (interface de ligne de commande) pour Wokwi, afin de faciliter l'utilisation du simulateur dans les actions GitHub.

Elektor : vous semblez être à la fois un artiste et un ingénieur, comme on peut s'en rendre compte en lisant votre article intitulé *A Practical Guide to Designing PCB Art* **[10]. Parlez-nous de votre approche créative de l'art et de l'électronique.**
Uri Shaked : je suis assis avec des amis et je prépare un badge intelligent pour une conférence. Après avoir discuté du matériel à ajouter au badge, quelqu'un mentionne qu'il serait également intéressant que les participants puissent écrire de petits jeux pour le badge et disposer d'une sorte de magasin d'applications pour partager leurs créations. À ce moment-là, une idée a germé dans mon esprit. Nous n'avons pas besoin d'un magasin d'applications en ligne, nous pouvons créer un module matériel avec une petite puce EEPROM. Les participants écriront leurs applications et leurs jeux sur ce module, puis l'offriront à leurs amis pour qu'ils puissent copier le code sur leur badge. C'est ainsi qu'est né le module complémentaire Floppy Disk [11]. Une fois que j'ai eu l'idée, il m'a suffi de rechercher les dimensions d'une disquette, de dessiner un croquis avec Inkscape et de décider quelle couche de PCB utiliser pour chaque élément : la sérigraphie pour l'autocollant, du cuivre étamé (HASL) pour l'obturateur métallique et cuivre recouvert d'un masque de soudure pour le symbole HD. J'ai réalisé le schéma et le routage dans KiCad, et le résultat, eh bien, vous pouvez le voir sur la **figure 10**.

Elektor : avez-vous d'autres projets importants en vue ?

Uri Shaked : Tiny Tapeout [12]. Je travaille dessus avec Matt Venn, et notre objectif est de rendre le silicium personnalisé plus abordable et plus accessible. Vous pouvez créer une conception numérique avec Verilog ou utiliser Wokwi pour créer une conception à partir de portes logiques individuelles, puis renforcer votre conception à l'aide d'OpenLane (un outil de conception numérique open-source), et la soumettre pour qu'elle soit incluse dans notre puce. Nous prenons votre conception, ainsi que des centaines venant d'autres personnes, et nous combinons le tout en une conception de puce unique. Nous fabriquons ensuite les puces, les assemblons sur un circuit imprimé et vous envoyons une version physique. Le circuit imprimé que vous voyez à la **figure 11** comporte un ensemble de commutateurs DIP pour la sélection d'un modèle, de sorte que vous pouvez également jouer avec les

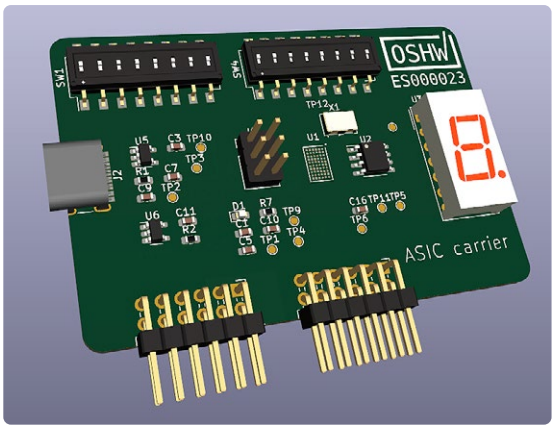


Figure 11. Tiny Tapeout breakout PCB (rendu préliminaire).

autres modèles de la puce. En fait, nous encourageons cela, toutes les conceptions sont open-source, et nous compilons une fiche technique avec toute la documentation. Nous commençons maintenant à collaborer avec des universités pour apporter Tiny Tapeout à leurs étudiants, et à créer du matériel pédagogique pour aider les amateurs et les passionnés de silicium à se familiariser avec les semi-conducteurs et la conception numérique. Plus tôt cette année, nous avons publié Siliwiz, une application open-source qui intègre un éditeur de disposition visuelle, un moteur DRC et une simulation SPICE, le tout fonctionnant à l'intérieur du navigateur. Il y a beaucoup plus à dire sur Tiny Tapeout, la communauté, et les conceptions créatives que nos utilisateurs mettent en place, mais, cette histoire est pour une prochaine fois !

VF : Maxime Valens - 230366-04

À propos d'Uri Shaked

Uri Shaked est un créateur de longue date. Il travaille actuellement sur Wokwi, une plateforme de simulation en ligne d'IdO et de systèmes embarqués, et sur Tiny Tapeout, qui rend la fabrication d'ASIC personnalisés abordable et accessible. Ses projets et son blog sont disponibles à l'adresse [13], ainsi que de nombreuses conférences techniques et interviews vidéo [14].

À propos de l'auteur

Roberto Armani est un ingénieur en électronique qui a plus de trente-cinq ans d'expérience dans différents secteurs. Avant de rejoindre l'équipe d'Elektor en tant que rédacteur, il a acquis de l'expérience et des connaissances dans l'industrie informatique, l'imagerie électronique, les télécommunications, les équipements d'essai des matériaux et la publication sur le web. Outre l'électronique, il aime écouter (et chanter) de la musique classique et faire des promenades en haute montagne.

LIENS

[1] Projet Arduino de Xylophone sur disque dur : https://youtu.be/dw9U0WxtK9c
[2] Expériences avec une trompette robotisée : https://bit.ly/3N2FfCz
[3] Démonstration de trompette au Chrome Dev Summit : https://youtu.be/PEVAczB9uUQ
[4] Version réelle du « Offline Dinosaur Game » de Google Chrome : https://bit.ly/3qgsifz
[5] Problèmes de CLI avec Arduino : https://github.com/arduino/arduino-cli/issues/2135
[6] Projet de servos dansants : https://wokwi.com/projects/305336312628511297
[7] Puces personnalisées API : https://docs.wokwi.com/chips-api/getting-started
[8] Vidéo « Inversion de l'ESP32 Wi-Fi » : https://youtu.be/XmaT8bMssyQ
[9] Page de vote de Wowki : https://wokwi.com/features
[10] Guide pratique pour la conception de circuits imprimés : https://blog.wokwi.com/a-practical-guide-to-designing-pcb-art/
[11] Disquette Add-On : https://github.com/urish/floppy-disk-sao
[12] Tiny Tapeout: https ://tinytapeout.com
[13] Page web des projets d'Uri : https://urish.org/#projects
[14] Vidéos et interviews d'Uri : https://urish.org/#talks