Mise en œuvre de la sécurité à l'ère de l'IoT sur… un microcontrôleur

La combinaison des mesures de sécurité sur un microcontrôleur, le SAML11, qui vont de la protection physique de la mémoire à la technologie de séparation Arm TrustZone, offre aux développeurs la possibilité de construire une sécurité globale dans leurs systèmes embarqués. Pour in fine garantir que leurs applications ne seront pas le maillon faible en termes de sécurité dans le monde de l'Internet des objets. Explications de Microchip.

L'Internet des objets (IoT) a la capacité potentielle de créer une grande valeur à partir des données collectées en temps réel, issues des capteurs et des systèmes embarqués, sur n'importe quel réseau. Mais cette valeur ajoutée comporte des risques. La capacité à se connecter à quasiment tout appareil via une connexion réseau implique la possibilité, pour des utilisateurs malveillants qui passent leurs journées à exploiter des failles et des faiblesses, d'identifier de nombreuses cibles potentielles supplémentaires. Une fois le chemin d'accès trouvé, ils peuvent détruire des réseaux entiers, voler des données et faire chanter leurs propriétaires (figure 1). D'après un rapport du Kaspersky Labs rédigé au terme d'une enquête sur les activités de piratage informatique dans le monde entier, on estime que des attaques ont été menées sur plus de 40% de la totalité des ordinateurs des systèmes de contrôle industriel rien qu'au cours du premier semestre 2018.

Bien que les serveurs soient les principales cibles, certaines attaques concernent des appareils comparativement plus simples et apparemment sans risques. Qui pourrait penser qu'un thermomètre électronique dans un bassin de poissons constitue une menace potentielle pour la sécurité réseau d'une entreprise? Eh bien, ce fut le cas pour un casino. Les hackers ont exploité les défenses comparativement plus faibles du thermomètre pour pénétrer dans le réseau central de l'établissement. Depuis ce point d'entrée, ils ont eu accès à l'ensemble du système et ont trouvé rapidement une base de données des clients qu'ils ont copiée. La faille n'a été découverte que lorsqu'un consultant en sécurité a analysé le journal des connexions réseau et a trouvé des données ayant été envoyées à un serveur distant situé en Finlande, en utilisant des protocoles en principe utilisés pour le streaming multimédia.

AUTEUR

Ramanuja Konreddy, ingénieur marketing produit pour les microcontrôleurs 32 bits, Microchip Technology.

Une imagination sans limite pour les hackers…

Des réseaux de banques ont été forcés via les systèmes de surveillance par caméra en circuit fermé (CCTV), initialement installés pour améliorer la sécurité physique, et des hackers ont trouvé des moyens d'espionner des utilisateurs à leur domicile en utilisant la caméra intégrée de leur robot aspirateur. Pourtant, ces attaques peuvent être évitées. Il est possible de se protéger contre elles et de maintenir les appareils et le réseau principal protégés. Le secret consiste à mettre en place plusieurs niveaux de barrières de sécurité, qui rendront toute attaque particulièrement difficile et chronophage pour les cybercriminels potentiels.

Car de nos jours, les hackers profitent souvent des mauvaises décisions prises par les équipes de développement. Une erreur courante consiste à ce que les appareils soient dotés d'un mot de passe par défaut permettant aux utilisateurs distants de se connecter à un serveur sécurisé. Ce type de mot de passe est souvent enregistré dans l'application destinée à paramétrer facilement les objets connectés. La bonne pratique consiste à attribuer un mot de passe unique à chaque appareil. Cependant, à mesure que les fabricants améliorent leurs compétences en sécurité de base, les cybercriminels emploient des tactiques plus complexes, semblables à celles qu'ils utilisent contre les serveurs. Dans ce contexte en constante évolution, les systèmes embarqués ne peuvent pas compter sur leur image de cible de moindre valeur pour les hackers. Comme les propriétaires du casino l'ont appris à leurs dépens, tout objet connecté peut servir de porte d'entrée au réseau principal en cas d'attaque ciblée.

Il existe de nombreux types d'attaques qui peuvent être menées sur un système réseau. Une technique couramment employée par les cybercriminels consiste à exploiter les failles logicielles. Si l'appareil reçoit, par exemple, plus de données dans un paquet de données qui a été attribué à un tampon sur la pile système, les octets supplémentaires vont dépasser du tampon et « se déverser » dans les structures de données voisines. Quand une autre routine arrive et chasse ces octets de la pile, les données sont susceptibles d'être utilisées par d'autres routines, ce qui entraîne un plantage ou des résultats erronés. Le processeur peut même interpréter ces valeurs comme des adresses de branchement cibles et tenter d'exécuter le mauvais code. Un hacker familier de la structure mémoire de l'appareil utilisera ses connaissances pour construire des miniprogrammes qui se feront ouvrir

l'appareil. Autre technique similaire : il est possible d'envoyer des données erronées à l'appareil entraînant la non-exécution des sous-programmes utilisés pour traiter les octets et ainsi rendre potentiellement vulnérable l'appareil.

D'autres types d'attaques ciblent les protocoles de communication plutôt que le logiciel de l'appareil et tentent de submerger le système jusqu'à le faire planter. Pendant que l'appareil essaie de se rétablir, les cybercriminels peuvent tenter de pénétrer le système à cet endroit. Si l'attaquant est capable de contrôler les équipements réseau à proximité ou d'obtenir un accès physique direct, il peut se faire passer pour les serveurs légitimes avec lesquels l'appareil veut communiquer. De telles attaques, dites du « singe intercepteur », peuvent être utilisées pour analyser des données issues de l'appareil, pouvant se révéler utiles pour une future attaque. Le hacker peut également essayer de charger son propre firmware sur l'appareil. Une fois redémarré avec le firmware malveillant, l'appareil effectue toutes les actions voulues par le cybercriminel.

Sécuriser un firmware

Dans l'idéal, le système rejettera les tentatives de communication par des machines qui ne peuvent pas prouver qu'elles ont les droits d'accès. Ainsi, l'appareil rejettera le faux firmware envoyé par le hacker. L'appareil pourrait également ignorer les tentatives de connexion en cas d'attaque de déni de service et donc éviter d'utiliser des ressources provoquant le plantage. Et il éviterait d'envoyer des données sensibles vers une machine impossible à identifier de manière fiable. Pour les paquets que l'appareil accepte, il vérifierait leur longueur et leur exactitude, rejetant toute trame de bits mal formée qui pourrait provoquer un dépassement du tampon ou une attaque par injection de commandes. Cependant, la mise en œuvre de toutes ces protections dans le firmware de l'appareil peut s'avérer hors de prix pour toute base de code de grande taille, en particulier pour celles qui ont beaucoup recours à des bibliothèques et applications logicielles tierces.

Une autre approche, plus réaliste, consiste à diviser le firmware en plusieurs sections, dont certaines ont besoin d'un degré de garantie de sécurité élevé, et d'autres sont autorisées à subir une attaque car elles ne mettront pas en péril les fonctions de sécurité. Par exemple, un sous-programme qui ne fait qu'encapsuler des données de température dans un format de données textuelle JSON (JavaScript Object Notation) pour qu'elles puissent être envoyées vers une application de smartphone, n'a pas besoin de subir des contrôles quant à ses propriétés de sécurité. Le code sécurisé, en revanche, s'assurera que l'authentification est effectuée avant que les données ne soient envoyées.

La part logicielle sur l'appareil qui nécessite un renforcement complet de la sécurité ne représente qu'une fraction de la base de code totale. Cependant, cette séparation n'est effective que s'il n'existe pas de porte dérobée entre le code non sécurisé et les routines ultrasécurisées, exploitable par les hackers via des attaques utilisant l'escalade des privilèges. Par exemple, si une attaque par dépassement de tampon parvient à charger dans la mémoire dite « sécurisée » des données qui attribuent au cybercriminel une identité d'administrateur système, alors toute protection est vaine. L'isolation de la mémoire sécurisée par rapport aux zones non sécurisées est capitale et ne peut être mise en œuvre de manière fiable qu'au niveau matériel.

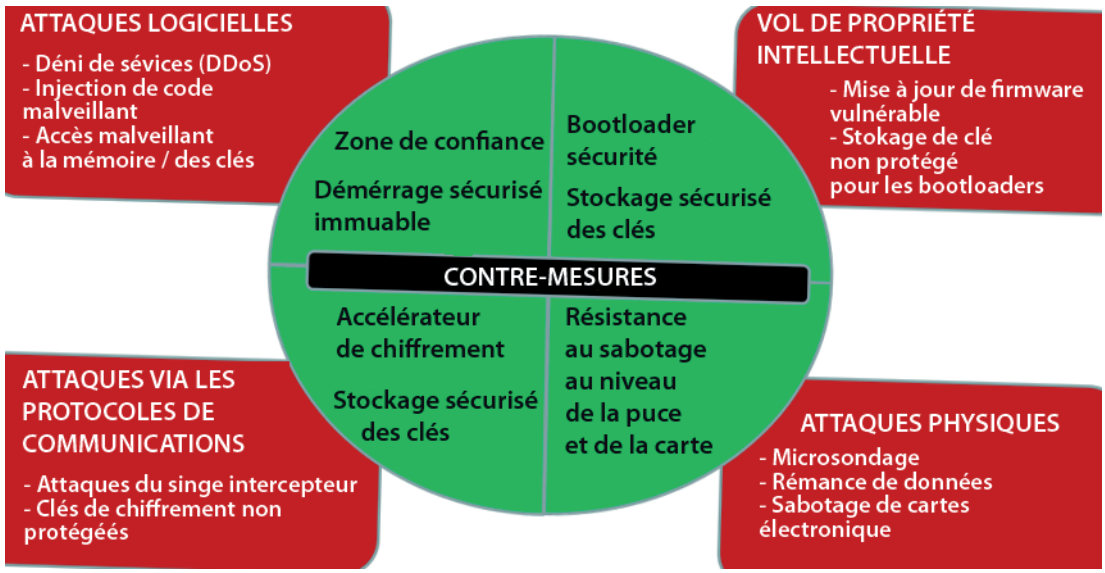
Utiliser les différents niveaux de protection matériels

Les microcontrôleurs embarqués tels que ceux de la famille SAML11 de Microchip contiennent des composants de mise en œuvre de la sécurité reposant sur une architecture Arm TrustZone, augmentée de protections propriétaires contre les attaques logicielles. La sécurité matérielle du SAML11 permet ici de construire une racine de confiance et de l'utiliser pour étendre les protections de sécurité au reste du système, formant la base d'un framework de sécurité complet. Approche qui peut s'étendre au-delà du circuit lui-même jusqu'à l'application finale, via plusieurs utilisateurs (figure 2).

La racine de confiance peut effectuer des opérations de chiffrement qui étendent cette zone de confiance hors de ses limites, pour y inclure d'autres éléments du système, per-

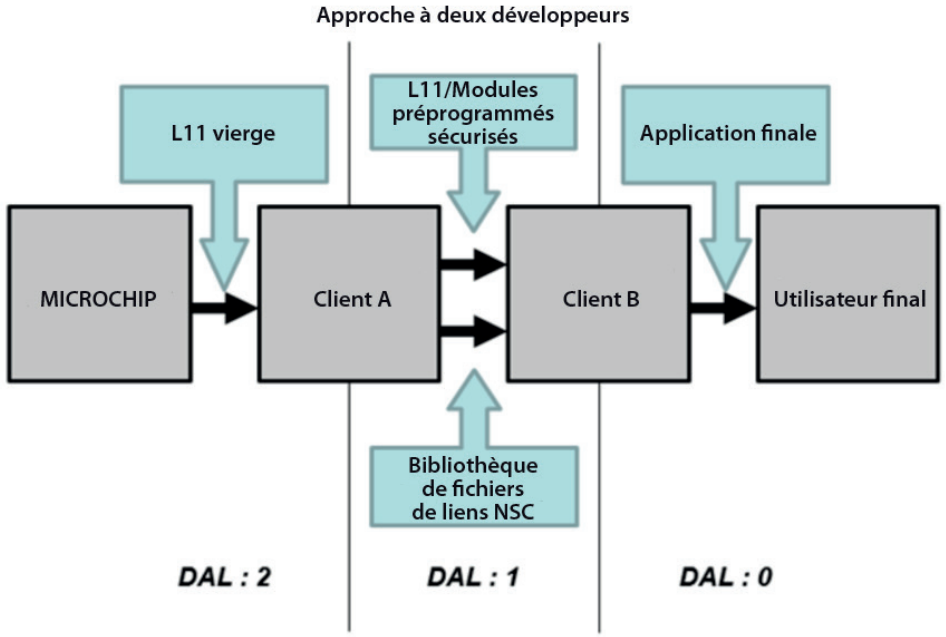
1 TOPOLOGIE DES DIFFÉRENTS TYPES D'ATTAQUES SUR UN SYSTÈME EMBARQUÉ

La capacité à se connecter de quasiment tout appareil via une connexion réseau implique la possibilité, pour des utilisateurs malveillants qui passent leurs journées à exploiter des failles et des faiblesses, d'identifier de nombreuses cibles potentielles.



2 APPROCHE À DEUX DÉVELOPPEURS

La sécurité matérielle du SAML11 permet de construire une racine de confiance et de l'utiliser pour étendre les protections de sécurité au reste du système, formant la base d'un framework de sécurité complet.



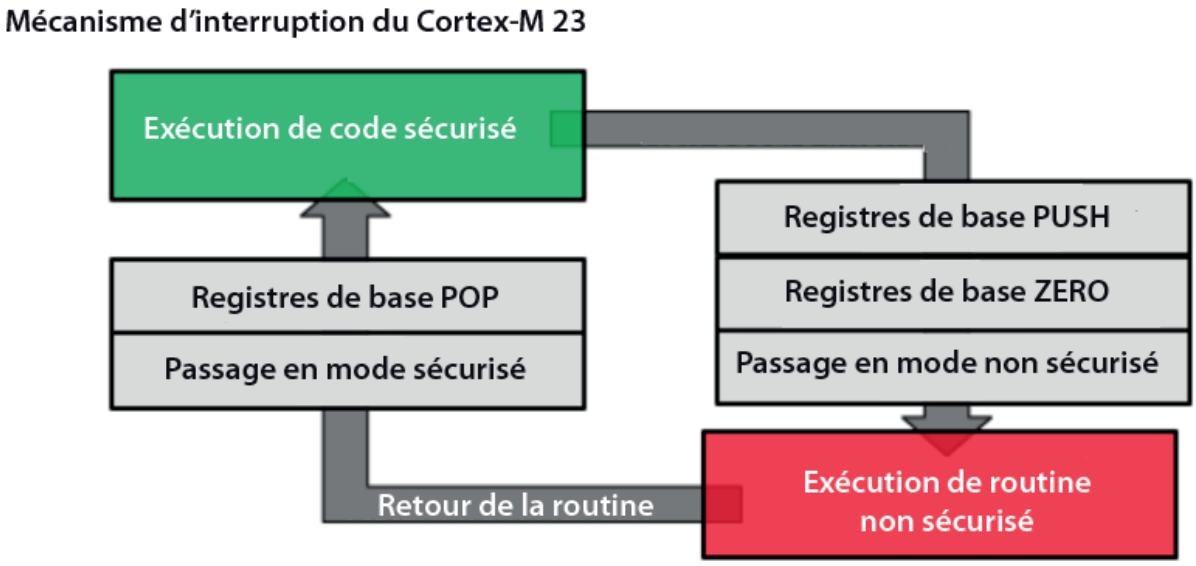
mettant des communications protégées sur un réseau considéré comme non fiable. Un contrôleur cryptographique intégré facilite de son côté les opérations de génération des clés par session ainsi que les opérations de chiffrement et de déchiffrement. Le contrôleur peut également aider à vérifier l'authenticité non seulement des messages entrants issus du réseau, mais également du code exécuté par le système. Les protections cryptographiques protègent même contre les contrefaçons matérielles. Dans ce cas, le logiciel exécuté dans la zone de confiance peut interroger les autres cartes d'un châssis en envoyant des défis auxquels seules les cartes autorisées peuvent répondre correctement.

Pour maintenir une racine de confiance, les circuits SAML11 utilisent une procédure de démarrage sécurisé. Pour empêcher toute erreur au démarrage, le séquence de démarrage initiale est écrite dans du code enregistré dans une zone de mémoire ROM spécifique qui ne peut être modifiée après fabrication et ne peut donc être contournée. Une fois le démarrage initial terminé, des services dans le code de la ROM de démarrage vérifient l'authenticité du reste du code du firmware nécessaire pour finaliser la procédure de démarrage. Cette opération est effectuée à l'aide d'un accélérateur de chiffrement, en vérifiant le code de hachage enregistré pour chaque segment du firmware, afin de s'assurer qu'il correspond bien au hachage de référence encodé par des fusibles programmés en usine. Toute valeur non conforme entraîne le redémarrage de l'appareil et relance la procédure de démarrage sécurisé. Ainsi, même si le cybercriminel réussit à modifier le firmware stocké sur la mémoire flash intégrée, l'appareil ne peut démarrer tant que la version d'usine n'a pas été restaurée.

Une fois que l'appareil a démarré avec succès et qu'il fonctionne avec un firmware connu et sûr, la mise en œuvre de la technologie Arm Trust Zone sur le SAML11 entre en jeu pour maintenir une séparation nette entre les logiciels sûrs et les logiciels potentiellement dangereux. La technologie Arm TrustZone dans le code du processeur Arm Cortex-M23 du SAML11 fournit ainsi un jeu d'ins-

3 MÉCANISMES D'INTERRUPTION DU CORTEX-M23

La technologie Arm TrustZone dans le code du processeur Arm Cortex-M23 du microcontrôleur SAML11 fournit un jeu d'instructions sécurisé qui garantit que tout appel de fonction effectué par du code non sécurisé envoyé vers le domaine sécurisé est vérifié pour y détecter un problème éventuel.



tructions sécurisé qui garantit que tout appel de fonction effectué par du code non sécurisé envoyé vers le domaine sécurisé est vérifié pour y détecter tout problème éventuel (figure 3). La technologie Arm TrustZone permet la création de domaines de sécurité logiciels restreignant l'accès de certaines sections de mémoire, de certains périphériques et de certaines entrées/sorties à certains logiciels de confiance, sans pour autant compromettre les performances du système. En autorisant le code sécurisé à être collecté et protégé, la technologie Arm TrustZone simplifie ainsi l'évaluation de la sécurité d'un appareil embarqué (figure 4).

Pour différencier et isoler le code sécurisé du code non sécurisé, la mémoire du SAML11 est partitionnée en plusieurs zones distinctes, chaque zone étant configurée par des fusibles pour résister aux attaques logicielles. Toute tentative d'accès aux zones identifiées comme sécurisées par du code non sécurisé, ou toute exécution de code ne correspondant pas à l'état de sécurité du système, entraîne une exception de panne matérielle qui empêche la demande d'accès d'aboutir ainsi que le redémarrage éventuel du système. Cette protection est maintenue y compris pendant les interruptions de maintenance et de débogage.

Par exemple, la mise en œuvre de la technologie Arm TrustZone maintient deux pointeurs de piles qui séparent l'exécution sécurisée de l'exécution non sécurisée et empêchent toute mise en danger des données sur la pile pouvant être tentée pendant la routine d'interruption. Pendant le débogage, le code sécurisé et le code non sécurisé sont traités de façon différente grâce à plusieurs niveaux d'accès de débogage. Un développeur travaillant sur des sections non sécurisées ne peut modifier du code sécurisé ou accéder directement aux informations de débogage qu'il contient. Les responsabilités sont ainsi clairement séparées, de sorte que seuls les développeurs dotés des autorisations de sécurité nécessaires peuvent travailler sur le code protégé.

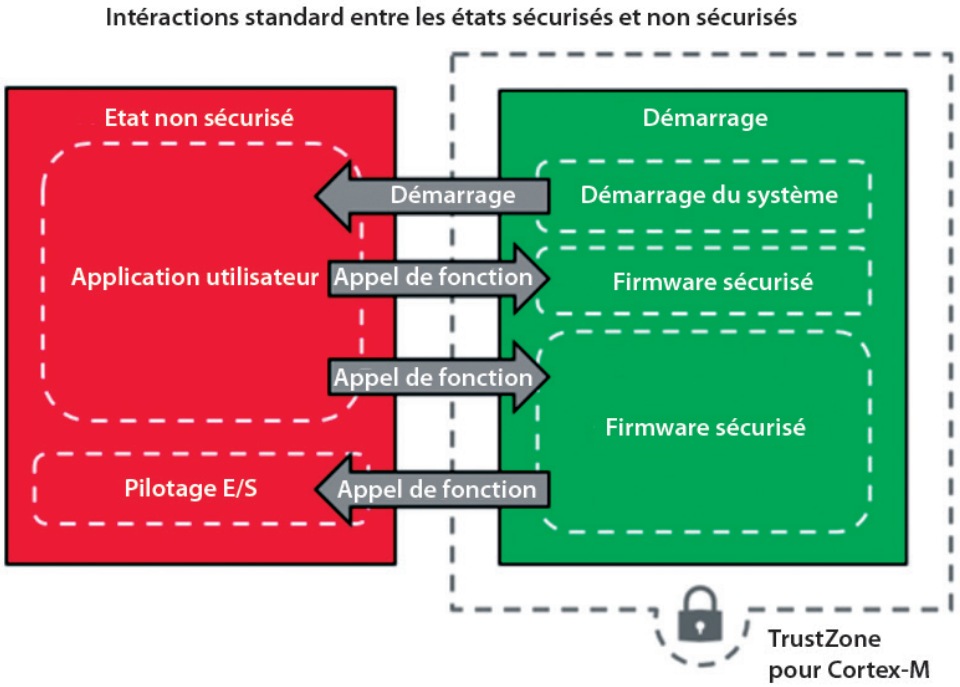
Typiquement, un développeur chargé d'écrire les applications sécurisées fournira les fichiers d'en-tête et les routines des bibliothèques permettant au code non sécurisé d'effectuer des requêtes, comme le chiffrement d'un paquet de données du réseau devant être envoyé sur Internet. L'application sécurisée est ensuite chargée dans une zone mémoire protégée. Quand un développeur sans autorisation de sécurité développe le code réseau, il doit utiliser la bibliothèque et les fichiers de lien fournis, mais il aura uniquement accès au niveau API : le code sécurisé et ses données restent dans la boîte noire. Les tentatives de modification du code sécurisé sont vouées à l'échec au moment du démarrage parce que le code de hachage ne correspond pas à l'exécution. C'est possible grâce aux vérifications de cohérence effectuées par la procédure de démarrage sécurisé immuable offerte par le SAML11. Par exemple, la manipulation de pointeurs pour se référer à un emplacement incorrect entraînera une erreur.

Les périphériques peuvent également être identifiés comme sécurisés ou non sécurisés, avec pour résultat que seuls les logiciels autorisés peuvent y accéder ou les contrôler directement. Pour les périphériques qui peuvent fournir des services aux deux types de zone, leur accès est protégé de façon identique que pour les appels de fonction. Le code non sécurisé effectue une requête via une commande API fournie par le développeur du code sécurisé. Ainsi, tout contrôle direct du périphérique est maintenu par le code authentifié, qui peut vérifier qu'il ne s'agit pas d'une tentative d'utilisation malveillante. Par exemple, le code non sécurisé peut être autorisé à lire l'état d'un compteur de temps, mais ne pas être autorisé à le remettre à zéro.

Les périphériques externes et les sous-systèmes peuvent être autorisés à effectuer des requêtes au microcontrôleur seulement s'ils fournissent une valeur de hachage qui correspond au certificat numérique contenu dans la clé stockée de manière sécurisée sur la puce et aux données qu'ils envoient. Ainsi, on empêche toute tentative d'utilisation d'un périphérique compromis par un hacker pour saboter le fonctionnement du système. C'est également une bonne façon pour le fabricant de garantir que ses appareils ne peuvent être utilisés avec des sous-systèmes contrefaits. Même si le cybercriminel peut sonder et modifier les 256 octets de RAM destinés au stockage de la clé, le SAML11 intègre des mécanismes qui annulent les clés et les données en cas de détection de ce type d'activité. ■

4 INTERACTIONS STANDARD ENTRE LES ÉTATS SÉCURISÉS ET NON SÉCURISÉS

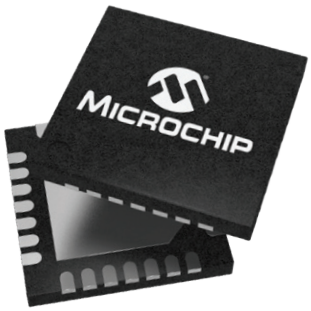
En autorisant le code sécurisé à être collecté et protégé, la technologie Arm TrustZone simplifie l'évaluation de la sécurité d'un appareil embarqué.



Boostez votre système grâce à notre écosystème complet

Nous avons ce qu'il vous faut, quand il vous le faut

Vous cherchez le chemin le plus rapide, le plus simple et le moins risqué entre le prototype et la production ? Microchip vous offre une aide à la conception complète à chaque étape de votre projet, grâce à notre écosystème de développement complet.



- Construisez des prototypes rapidement grâce à un environnement de conception et de débogage intuitif
- Démarrez rapidement votre projet grâce à des systèmes de référence et à des composants spécifiques à votre application
- Diminuez les risques grâce à des outils ayant fait leurs preuves et à un logiciel testé par des professionnels

Quel que soit votre besoin, nous vous offrons une aide à la conception complète à chaque étape de votre projet.

Commencez à développer dès maintenant en allant sur
www.microchip.com/Ecosystem

MICROCHIP