# La flexibilité des mises à jour du firmware : un atout pour les objets connectés

Les développeurs de systèmes pour l'Internet des objets doivent fournir la flexibilité nécessaire pour mettre à jour le code et les données. Quel code et quelle quantité de code mettre à jour ? A quelle fréquence réaliser les mises à jour ? Combien de temps la mise à jour va-t-elle prendre ? Telles sont des questions qui doivent être étudiées pendant la phase de conception de l'objet connecté. Ici, la sélection de la mémoire non volatile a un impact sur ces questions et joue un rôle crucial dans le calcul du temps et de la vitesse des mises à jour du code, comme l'explique Microchip.

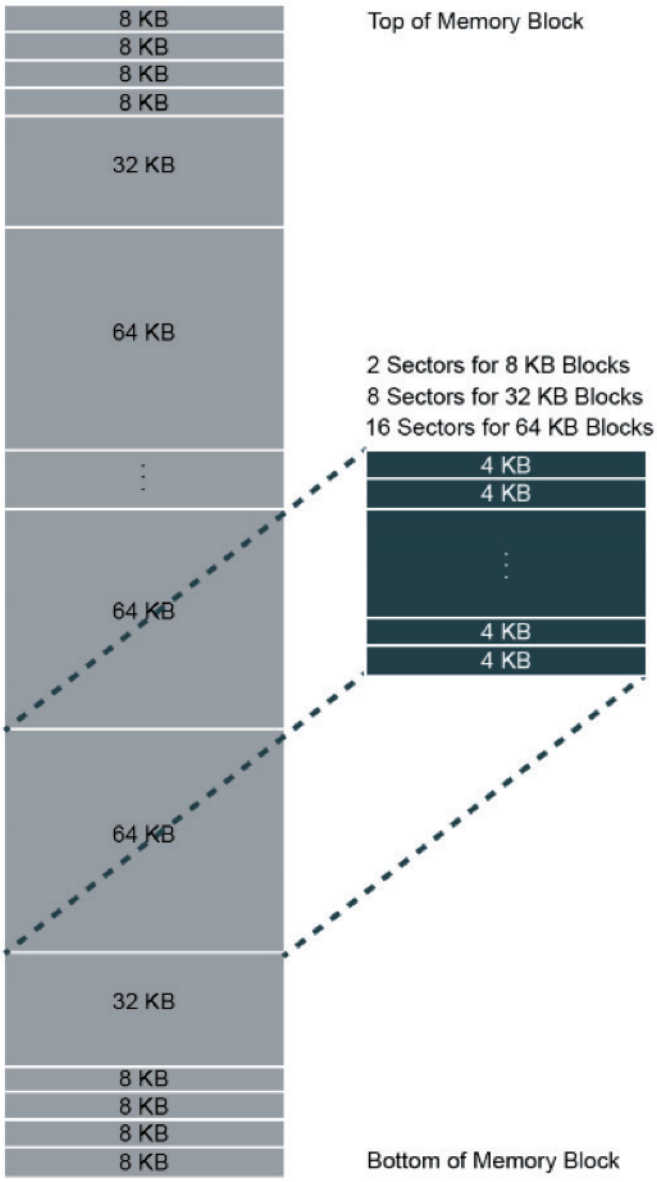
AUTEUR

Hardik Patel, ingénieur d'applications, Microchip.

Les objets connectés via l'Internet des objets (IoT, Internet of Things) pénètrent le marché à une vitesse vertigineuse, depuis les appareils électroménagers jusqu'aux appareils médicaux, en passant par les véhicules. Dans cette course en avant, les développeurs doivent intégrer la flexibilité à leurs produits pour s'adapter à un écosystème des objets connectés en constante évolution, à mesure que les nouvelles fonctionnalités et règlementations apparaissent. Dans ce cadre, les mises à jour des firmwares permettent non seulement la personnalisation pendant le déploiement initial sur le site du client, mais également d'ajouter de nouvelles fonctions/caractéristiques au produit une fois qu'il est installé sur le terrain, ou de corriger les problèmes logiciels en cours d'utilisation. Le code du firmware est souvent stocké sur un composant de mémoire non volatile du type mémoire flash NOR, en raison de leurs capacités de reprogrammation et de leur fiabilité. En réécrivant une partie du code logiciel de l'appareil qui est stocké dans la mémoire non volatile utilisée dans l'appareil, les fabricants peuvent facilement mettre à niveau les capacités de l'appareil. Pour mettre à jour des firmwares, trois paramètres doivent être pris en compte : Quel code et quelle quantité de code mettre à jour ? A quelle fréquence réaliser les mises à jour ? Et combien de temps la mise à jour va-t-elle prendre ?

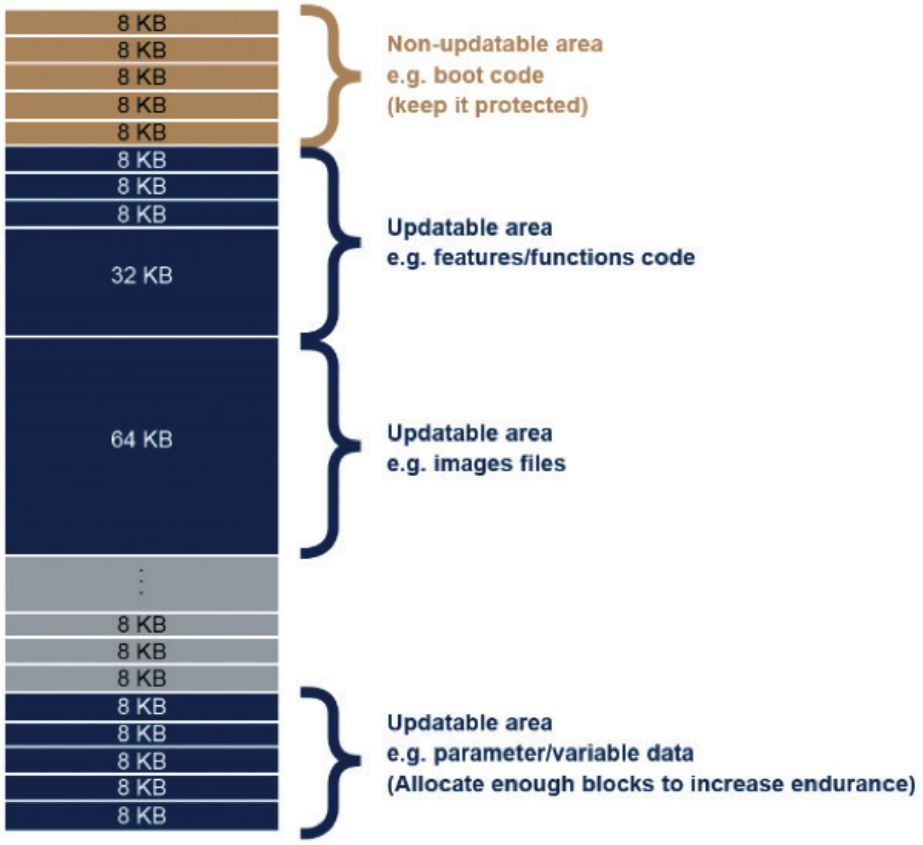
**1 ORGANISATION DE LA MÉMOIRE**

On voit ici comment la mémoire SST26VF064B est organisée avec huit blocs de 8 Ko, deux blocs de 32 Ko et 126 blocs de 64 Ko.



**2 ORGANISATION DE LA MÉMOIRE DANS LES ZONES NON ACTUALISABLES**

On voit ici l'organisation de la mémoire dans les zones non actualisables (par exemple le code de démarrage) et dans les zones actualisables (par exemple le code pour les fonctions/fonctionnalités, les fichiers image et les variables de paramètres).




## Quel code et quelle quantité de code mettre à jour ?

Il faut décider quelle partie du code et quelle quantité de code seront mises à jour pendant la phase de conception initiale de l'objet connecté. La portion de code du firmware actualisable doit être stockée dans une zone séparée de la mémoire flash NOR, différente du reste du code non actualisable. La mise à jour de tout composant flash NOR se fait en deux étapes : on efface d'abord la zone de la mémoire concernée, puis on programme les nouvelles informations dans cette zone. La mémoire flash NOR est organisée en différentes zones appelées secteurs ou blocs de mémoire, de différentes tailles. Les composants flash NOR, comme ceux de la technologie SuperFlash de SST (référencées SST-26VF064B pour 64 Mbit ou 64 Mo), sont constitués de plusieurs secteurs de 4 Ko uniformes qui peuvent être effacés et reprogrammés individuellement (4 Ko = $4 \times 10^{24}$ x 8 bits = 32 762 bits). Ils sont également constitués de plus gros blocs de 8, 32 et 64 Ko, qui peuvent également être effacés individuellement. Ainsi, un bloc de 8 Ko est constitué de deux secteurs, un bloc de 32 Ko est constitué de huit secteurs et un bloc de 64 Ko est constitué de seize secteurs (figure 1).

Chaque bloc peut également être protégé individuellement. Avant de réaliser toute mise à jour sur une zone de la mémoire flash, les blocs de cette zone doivent être déverrouillés pour permettre l'effacement et la programmation. Une fois la mise à jour effectuée, mieux vaut remettre en place la protection pour ces blocs, afin d'éviter tout écriture ou effacement inopiné dans ces zones. La portion de code du firmware pouvant être mise à jour doit être organisée en secteurs et en blocs, de sorte qu'il y ait assez de flexibilité pour permettre à la fois des mises à jour de fonctions/fonctionnalités minimales et maximales. La vitesse de réalisation des mises à jour étant déterminée par le nombre de secteurs et de blocs à effacer et à reprogrammer, mieux vaut considérer la vitesse et la flexibilité comme allant de pair au moment d'organiser la portion du firmware pouvant être mise à jour (figure 2). Les portions non actualisables, comme celle du code de démarrage, sont stockées dans des zones protégées. Les portions du logiciel actualisables, comme les fonctions et fonctionnalités, sont divisées en blocs plus petits ou plus grands, selon les besoins en flexibilité. Les fichiers image pouvant être mis à jour sont stockés dans des blocs plus gros tandis que les variables/paramètres du même type sont stockés dans des blocs plus petits.

**I.- SÉQUENCE D'INSTRUCTIONS DE COMMANDE FLASH POUR METTRE À JOUR 1, 2 OU 4 MO DE MÉMOIRE**

| Steps | Instruction Sequence for Commands | Number of Clocks |
|---|---|---|
| 1 | SPI_WREN | 8 |
| 2 | SPI_Enable_Quad_IO | 8 |
| 3 | SQI_WREN | 2 |
| 4 | SQI_ Write Block Protection Register (to unprotect portion of Flash) | 38 |
| 5 | SQI_Block Erase 64 KB blocks | 2 |
| 6 | SQI_WREN | 8 |
| 7 | **Wait 25 ms for completion of erase for SuperFlash technology memory (or 3000 ms for conventional Flash memory)** | |
| | Repeat steps 5, 6 and 7 till the required amount of memory is erased | |
| | To erase 1 Mb of data, steps 5, 6 and 7 will need to be repeated for (1024*1024)/(64*1024*8) = 2 times | |
| | To erase 2 Mb of data, steps 5, 6 and 7 will need to be repeated for (2*1024*1024)/(64*1024*8) = 4 times | |
| | To erase 4 Mb of data, steps 5, 6 and 7 will need to be repeated for (4*1024*1024)/(64*1024*8) = 8 times | |
| 8 | SQI_WREN | 2 |
| 9 | SQI_Page program command followed by programming 256 bytes of data (260 multiplied by 2) | 520 |
| 10 | **Wait 1.5 ms for completion of page programming for SuperFlash technology memory (or 5 ms for conventional Flash memory)** | |
| | Repeat steps 8, 9 and 10 until all the data is written | |
| | To program 1 Mb of data, steps 8, 9 and 10 will need to repeated for 1024*1024/8 divided by 256 = 512 times | |
| | To program 2 Mb of data, steps 8, 9 and 10 will need to repeated for 2*1024*1024/8 divided by 256 = 1024 times | |
| | To program 4 Mb of data, steps 8, 9 and 10 will need to repeated for 4*1024*1024/8 divided by 256 = 2048 times | |
| 11 | SQI_WREN | 2 |
| 12 | SQI_Write Block Protection Register (to again protect the portion of Flash) | 38 |

**II.- TEMPS DE PROGRAMMATION ET D'EFFACEMENT POUR LES SST26VF064B ET LES FLASH CONVENTIONNELLES**

| | SST26VF064B | Conventional Flash |
|---|---|---|
| Sector erase | 25 ms (max) | 150 ms to 3000 ms |
| Block erase | 25 ms (max) | 750 ms to 3 s |
| Chip erase | 50 ms (max) | 15 s to 80 s |
| Page program | 1.5 ms (max) | 1 ms to 5 ms |

## A quelle fréquence réaliser les mises à jour ?

La principale limite de la fréquence des mises à jour est liée à la durée de vie de la mémoire utilisée dans l'application. La technologie de mémoire SuperFlash, qui est celle du SST-26VF064B, possède une durée de vie de 100 000 cycles, ce qui signifie que chaque secteur peut être programmé et effacé 100 000 fois. La possibilité d'effectuer 100 000 mises à jour semble énorme. Cependant, de nombreux objets connectés collectent des données et stockent les informations recueillies dans une flash NOR pendant l'utilisation, ce que l'on doit prendre en compte lors du calcul du nombre maximum de

### III.- TEMPS NÉCESSAIRE POUR ACTUALISER 1, 2 OU 4 MO DE MÉMOIRE UTILISANT LA TECHNOLOGIE SUPERFLASH

| Calculations | Time |
|---|---|
| Maximum block erase time | 25 ms |
| Maximum page program time | 1.5 ms |
| Clock period for 104 Mhz frequency | 9.6 ns |
| CE high time between each instruction for 104 Mhz | 12 ns |
| **Time A.** Time for instruction 1 to 4 = (56 clocks * period) + (3 * CE high times) | 573.6 ns |
| **Time B.** Time for instruction 5 to 7 = (10 clocks * period) + (2 * CE high times) + (25ms wait for block erase) | 25000120 ns |
| **Time C.** Time for instruction 8 to 10 = (522 clocks * period) + (1 * CE high time) + (1.5 ms wait for page program) | 1505023.2 ns |
| **Time D.** Time for instruction 11 to 12 = (40 clocks * period) + (1 * CE high time) | 396 ns |
| **Total time for all the instructions for block erase and programming 1 Mb of data = (Time A) + (Time B * 2) + (Time C * 512) + (Time D)** | **0.820573088 s** |
| **Total time for all the instructions for block erase and programming 2 Mb of data = (Time A) + (Time B * 4) + (Time C * 1024) + (Time D)** | **1.641145206 s** |
| **Total time for all the instructions for block erase and programming 4 Mb of data = (Time A) + (Time B * 8) + (Time C * 2048) + (Time D)** | **3.282289443 s** |

cycles. Il est donc important d'allouer suffisamment de secteurs de la mémoire en tenant compte de la longévité. Pour être plus clair, prenons un exemple : imaginons que l'objet connecté collecte et stocke 16 octets d'informations et qu'on pense que ces informations seront collectées et stockées 100 millions de fois pendant la durée de vie du produit. Le nombre de secteurs devant être alloué peut se calculer comme suit :

- 1 secteur = 4 Ko ;
- tous les emplacements d'adresse dans le secteur sont utilisés pour stocker les informations, soit 16 octets de données en même temps, et ces données sont écrites dans un nouvel emplacement d'adresse jusqu'à ce que la fin du secteur soit atteinte (par exemple 0x0000-0x000F puis 0x0010-0x001F puis 0x0020-0x002F, etc.) ;
- 4 Ko/16 = 256, ce résultat est le nombre de fois où les informations stockées peuvent être écrites avant d'atteindre la capacité maximale du secteur et d'effacer une donnée du secteur ;
- la limite d'endurance d'un secteur est de 100 000 cycles ;
- ainsi, si on peut écrire 256 fois sur un secteur pendant 100 000 cycles, les données peuvent être collectées et stockées 25 600 000 fois ;
- et si une application nécessite d'avoir des données collectées et stockées 100 millions de fois, alors le nombre de secteurs à allouer est de 100 000 000/25 600 000 = 3,9. Par conséquent, dans cet exemple, il est nécessaire d'attribuer 4 secteurs pour stocker 16 octets de données pendant toute la durée de vie de l'application.

Les ingénieurs de l'Internet des objets doivent procéder à des calculs similaires afin d'allouer suffisamment de secteurs et de blocs de mémoire pour les paramètres d'enregistrement chronologique des données, de sorte de ne pas dépasser la limite d'endurance de leur composant flash NOR.

## Vitesse des mises à jour

La vitesse des mises à jour peut être calculée en fonction du nombre de blocs et de secteurs qui doivent être effacés et reprogrammés. Imaginons qu'il faille reprogrammer 1 Mo, 2 Mo ou 4 Mo du code ou des données du firmware, qui seraient stockés dans plusieurs blocs de 64 Mo du SST26VF064B. Le code ou les données peuvent être constitués de portions de code du firmware, de fichiers image ou d'autres programmes devant être mis à jour. La réalisation de la mise à jour suppose d'effectuer une séquence d'instructions de commande dans la mémoire flash. La séquence se décompose en plusieurs étapes, qui suivent toujours le même ordre : déverrouiller les blocs de mémoire, effacer ces blocs, les reprogrammer avec les données/le code actualisés, verrouiller à nouveau les blocs de mémoire. Pour le SST26VF064B, la séquence d'instructions requise pour actualiser 1, 2 ou 4 Mo de mémoire est représentée dans le tableau I.

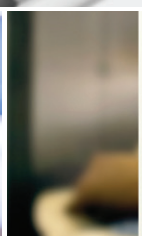
Sur ce tableau, on voit bien que les deux moments les plus significatifs sont la phase d'effacement et la phase de programmation. Le SST26VF064B utilise la technologie SuperFlash, qui offre d'excellentes performances d'effacement. Le tableau II présente quant à lui une comparaison entre les performances d'effacement et de programmation de la technologie SuperFlash par rapport aux flash conventionnelles. Les performances d'effacement largement supérieures offertes par la technologie SuperFlash par rapport aux flash conventionnelles se révèlent extrêmement utiles pour réduire le temps nécessaire pour une mise à jour. Le SST26VF064B supporte une fréquence d'horloge maximale de 104 MHz, un temps d'effacement de secteur minimum de 25 ms, un temps d'effacement de bloc maximum de 25 ms et un temps de programmation par page maximum de 1,5 ms. Un délai de 12 ns (temps maximum) est également requis entre chaque instruction de commande vers la mémoire flash qui fonctionne à une fréquence d'horloge de 104 MHz. En utilisant la séquence de commandes du tableau I et en connaissant le temps de programmation et le temps d'effacement, on peut calculer le temps nécessaire requis pour mettre à jour 1, 2 ou 4 Mo de mémoire utilisant la technologie SuperFlash et une mémoire flash conventionnelle, comme on peut le voir respectivement sur les tableaux III et IV. Ce type de calcul doit être effectué par des ingénieurs de l'Internet des objets pour estimer la vitesse de réalisation d'une mise à jour, dans le but de minimiser les temps d›arrêt des objets connectés pendant que la mise à jour est en cours. ■

### IV.- TEMPS NÉCESSAIRE POUR ACTUALISER 1, 2 OU 4 MO DE MÉMOIRE FLASH CONVENTIONNELLE

| Calculations | Time |
|---|---|
| Maximum block erase time | 3000 ms |
| Maximum page program time | 5 ms |
| Clock period for 104 Mhz frequency | 9.6 ns |
| CE high time between each instruction for 104 Mhz | 20 ns |
| **Time A.** Time for instruction 1 to 4 = (56 clocks * period) + (3 * CE high times) | 597.6 ns |
| **Time B.** Time for instruction 5 to 7 = (10 clocks * period) + (2 * CE high times) + (3000 ms wait for block erase) | 3000000136 ns |
| **Time C.** Time for instruction 8 to 10 = (522 clocks * period) + (1 * CE high time) + (5 ms wait for page program) | 5005031.2 ns |
| **Time D.** Time for instruction 11 to 12 = (40 clocks * period) + (1 * CE high time) | 404 ns |
| **Total time for all the instructions for block erase and programming 1 Mb of data = (Time A) + (Time B * 2) + (Time C * 512) + (Time D)** | **8.562577248 s** |
| **Total time for all the instructions for block erase and programming 2 Mb of data = (Time A) + (Time B * 4) + (Time C * 1024) + (Time D)** | **17.12515349 s** |
| **Total time for all the instructions for block erase and programming 4 Mb of data = (Time A) + (Time B * 8) + (Time C * 2048) + (Time D)** | **34.25030599 s** |

# Suite d'outils IoT Security

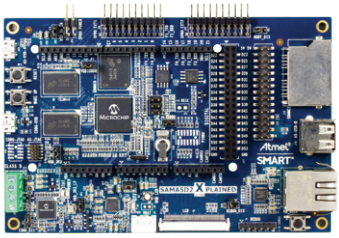
## Quand complexité rime avec simplicité




La suite d'outils IoT Security pour le microprocesseur SAMA5D2 permet une utilisation simple et une prise en main rapide de ses fonctionnalités de sécurité avancées, telles que la technologie ARM® TrustZone® et le moteur de chiffrement matériel, sans phase d'apprentissage longue et fastidieuse. La suite couvre les exigences de sécurité pour les fabricants d'objets connectés en un pack unique et facile à utiliser. Il permet le stockage, le chiffrement/déchiffrement et l'échange de clés entre appareils et applications, et ses interfaces de programmes d'application (API) faciles à utiliser vous font gagner un temps précieux.

### Caractéristiques

- **Démarrage fiable** – Démarrage vérifié par racine de confiance (RoT, Root of Trust)
- **Protection du firmware** – Chiffrement et exécution du firmware authentifié
- **ID de confiance de l'appareil** – Certificat unique de l'appareil lié à la RoT
- **Stockage sécurisé** – Stockage sécurisé des clés, certificats et données
- **Communications sécurisées** – Jumelage des appareils et communications via l'Internet des objets authentifiés
- **Mise à jour sécurisée du firmware** – Mise à niveau distante du firmware de manière sécurisée



Carte d'évaluation
Xplained Ultra SAMA5D2
(ATSAMA5D2-XULT)

Téléchargez gratuitement le kit d'évaluation de la suite IoT Security pour démarrer.

microchip DIRECT

MICROCHIP

**www.microchip.com/SAMA5D2**

Le nom et le logo de Microchip et le logo Microchip sont des marques déposées de Microchip Technology Incorporated aux États-Unis et dans d'autres pays. ARM et Cortex sont des marques déposées de ARM Limited (ou de ses filiales) aux Etats-Unis et dans d'autres pays. Toutes les autres marques commerciales ci-dessus mentionnées sont la propriété exclusive de leurs propriétaires respectifs. © 2018 Microchip Technology Inc. Tous droits réservés. DS60001511A. MEC2201Fre01/18