# Concevez séparément puis intégrez facilement grâce à un contrôleur DSC double cœur

Alors que le niveau d'intégration logicielle se complexifie au sein des applications embarquées, les concepteurs cherchent des moyens permettant de simplifier le processus. Un contrôleur de signal numérique à double cœur peut offrir une solution appréciable. Explications de Microchip.

Les applications embarquées deviennent de plus en plus complexes et sophistiquées afin de pouvoir répondre à plusieurs objectifs. Tout d'abord, les applications doivent améliorer leur efficacité, ce qui suppose un niveau de performance suffisant de la part du contrôleur pour faire tourner des algorithmes complexes. Ensuite, la disponibilité omniprésente d'Internet permet aux applications embarquées de devenir plus « intelligentes » et plus « connectées ». Le troisième objectif consiste à réduire les coûts en intégrant différentes fonctions telles qu'une interface pour capteurs, la connectivité, la commande de moteur, la conversion de puissance numérique, la sécurité et la sûreté, le tout sur un seul et unique contrôleur. Un tel niveau d'intégration nécessite l'intervention d'experts de tous ces domaines respectifs afin de gérer les domaines ou modules de fonctionnement spécifiques, puis intégrer plusieurs fonctions dans l'application finale. Souvent, les entreprises multinationales ont leurs équipes dispersées dans le monde entier, ce qui rend d'autant plus important le fait que plusieurs modules puissent être conçus séparément puis intégrés facilement afin de réduire les risques et les efforts de développement.

AUTEUR

Harsha Jagadish, responsable du marketing produit, département des microcontrôleurs 16 bits, Microchip Technology.

## Pour une efficacité améliorée

Tout d'abord, si l'on s'intéresse à l'objectif qui consiste à améliorer l'efficacité énergétique, on constate qu'il nécessite des performances accrues du contrôleur. Prenons l'exemple d'une application de commande de moteur. Le secteur a délaissé les moteurs DC à balais, qui offrent une efficacité de 75-80 % pour se tourner vers les moteurs DC sans balais (BLDC, Brushless Direct Current), ou encore les récents moteurs synchrones à aimants permanents (PMSM, Permanent Magnet Synchronous Motors). Ces moteurs affichent une efficacité améliorée atteignant 85-90 %, un bruit acoustique réduit et une meilleure durée de vie.

Une application de commande de moteur DC à balais typique nécessite des techniques très simples de commande de la direction et de la vitesse, qui peuvent être mises en œuvre à l'aide d'un microcontrôleur 8 bits d'entrée de gamme. Comparativement, la commande d'un moteur BLDC sans capteur ou synchrone à aimant permanent (PMSM) avec contrôle vectoriel FOC (Field Oriented Control) est plus complexe et plus gourmande en ressources. Elle permet de contrôler au plus près l'énergie utilisée par le moteur sur une large plage de charges ou de vitesses et permet d'améliorer significativement l'efficacité. Des algorithmes de commande supplémentaires peuvent également être mis en œuvre en fonction des exigences de l'application, tels que la détection et la sortie de décrochage du rotor, l'autorotation, la saturation de la boucle PID et la compensation anti-windup, l'affaiblissement de flux et le couple maximum par ampère. Ces algorithmes permettent d'améliorer les performances et les réponses à une charge dynamique et d'augmenter l'efficacité globale.

Toutes ces techniques de commande avancées sont très gourmandes en ressources de calcul et nécessitent des opérations mathématiques telles que divisions, multiplications, racines carrées et calculs trigonométriques, qui requièrent une bande passante CPU (Central Processing Unit) significative. Comme ces fonctions de commande doivent être exécutées régulièrement à haute fréquence, il est nécessaire que le CPU s'y consacre selon un intervalle de temps spécifique.

Une exécution de boucle de commande aussi précise peut occuper la quasi-totalité de la bande passante du CPU et peut affecter d'autres fonctions à contraintes temporelles au sein d'une application complexe. Les développeurs de systèmes embarqués disposent dès lors de peu de flexibilité pour ajouter des fonctionnalités supplémentaires, telles que la communication, la surveillance de la sûreté de fonctionnement ou des fonctions système ou de gestion interne qui pourraient interférer avec la commande critique du moteur. Le défi se corse encore sur les applications de puissance numérique, où les fonctions de boucle de contrôle à contraintes temporelles doivent être exécutées selon une fréquence encore plus élevée.

## Des logiciels complexes

Intéressons-nous à présent à l'objectif suivant, lié à la connectivité à Internet ou au cloud. Les dernières tendances du marché consistent à disposer d'applications « intelligentes » et « connectées », offrant intelligence et accessibilité n'importe où. Ces exigences supposent que les applications embarquées intègrent de multiples piles logicielles multiples telles que :

- Le logiciel de fonctionnement principal de l'application. Dans notre exemple, cette fonction met en

œuvre la commande du moteur, les tâches de gestion interne et les opérations de l'interface utilisateur, dont la plupart des applications ont habituellement besoin.
- Le logiciel de communication qui met en œuvre les protocoles réseau nécessaires pour la connectivité de l'application.
- Le logiciel de sécurité pour la protection de la propriété intellectuelle et des données personnelles, l'intégrité des données, l'authentification, le contrôle d'accès et la protection contre les éventuelles tentatives de piratage.
Si les applications impliquent des opérations humaines et peuvent causer des dommages corporels en cas de dysfonctionnement, alors le logiciel de sécurité fonctionnelle doit également faire partie des applications critiques.
Certaines applications finales peuvent également avoir des exigences de personnalisation pour lesquelles une ou plusieurs fonctionnalités seront uniques et ciblées en fonction d'un segment de marché spécifique.
Toutes ces exigences au niveau des fonctionnalités supposent que plusieurs équipes d'experts, dans des domaines différents, développent des piles logicielles spécifiques et soient capables de les intégrer rapidement de façon optimale à l'application finale. Ces experts issus de différents domaines devront collaborer très étroitement pour créer l'architecture de l'application finale et la mettre en œuvre. Ce scénario se complique encore dans le cas des entreprises multinationales, où les équipes d'experts peuvent être dispersées dans le monde entier.

## Réduction des coûts

Enfin, l'optimisation des coûts constitue un objectif important, commun à toutes les applications finales. Souvent, les ingénieurs de systèmes embarqués ne disposent pas d'un budget assez important pour pouvoir envisager un système avec plusieurs microcontrôleurs, dans lequel chaque pile logicielle individuelle peut être exécutée sur des microcontrôleurs différents avec très peu de coordination. Le choix d'utiliser un seul contrôleur pour leur système avec un haut niveau d'intégration reste souvent la meilleure solution. Les coûts sont ainsi d'autant plus réduits, grâce à un encombrement minime sur la carte électronique et à la réduction du nombre de composants externes tels que les oscillateurs à quartz et les composants passifs.

## Quels sont les défis côté développement ?

Afin de pouvoir mettre en œuvre des algorithmes complexes et exécuter plusieurs piles logicielles, les développeurs de systèmes embarqués optent souvent pour un microcontrôleur offrant de meilleures performances. Cependant, ce n'est pas toujours le meilleur choix à cause des problèmes inhérents à l'exécution avec des contraintes temporelles, au développement de plusieurs piles logicielles, à l'intégration et aux phases de tests. Un simple ordonnanceur ou un système d'exploitation temps réel RTOS (Real-time Operating System) peuvent être utilisés pour programmer des tâches multiples ainsi que leur exécution au sein de différentes piles sur un CPU à hautes performances, et ce en leur affectant des créneaux temporels différents. Toutefois, un ordonnanceur ou un système RTOS ajoute un surcoût qui consomme de la bande passante CPU, de la mémoire ainsi que d'autres ressources du microcontrôleur. L'attribution de créneaux temporels augmente également la surcharge de commutation, réduisant l'efficacité d'utilisation du CPU. Le scénario se complique lorsque deux boucles de contrôle complexes, avec des contraintes temporelles, doivent être exécutées périodiquement selon des intervalles de temps précis, qui se chevauchent, ou lorsque deux fonctions critiques asynchrones doivent être exécutées simultanément en temps réel. Dans de telles situations, l'utilisation d'un microprocesseur encore plus performant ne permettra pas toujours de répondre aux exigences du système. Même si un microcontrôleur monocœur hautes performances possède une bande passante suffisante pour gérer plusieurs piles logicielles, éventuellement associées à un système RTOS, il existe tout un tas d'autres complications possibles à prendre en compte. Le développement, l'intégration et la vérification de plusieurs piles logicielles requièrent un très haut niveau de coordination entre les différents experts. Il faut développer une architecture logicielle compatible et modulaire qui permette de partager dynamiquement des ressources et d'échanger des informations. Les complications se multiplient si le système inclut d'anciennes piles logicielles ne disposant pas d'une architecture compatible.
- Les piles anciennes peuvent être dotées d'architectures différentes,

● La disponibilité omniprésente d'Internet permet aujourd'hui aux applications embarquées de devenir plus « intelligentes » et plus « connectées ».

reposant sur un mode d'attente active (polling) ou d'interruption.
- Les piles logicielles anciennes peuvent utiliser les mêmes ressources du microcontrôleur, qui doivent donc être désormais partagées sans conflit pour éviter toute situation de concurrence ou de blocage du système.
- Les piles peuvent utiliser plusieurs variables et fonctions globales courantes qui possèdent les mêmes noms.
- Chaque pile peut fonctionner à la perfection si elle est exécutée seule, mais présenter un dysfonctionnement après intégration. Le débogage d'une solution intégrée de ce type peut s'avérer un vrai cauchemar, qui ne fera qu'augmenter le temps de développement.
- Une pile autonome déjà disponible n'aidera pas toujours à réduire le temps de développement si elle est implémentée sur un microcontrôleur à un seul cœur. Toutes ces difficultés entraînent des risques de développement significatifs et augmentent les délais de mise sur le marché.

## Un contrôleur à double coeur

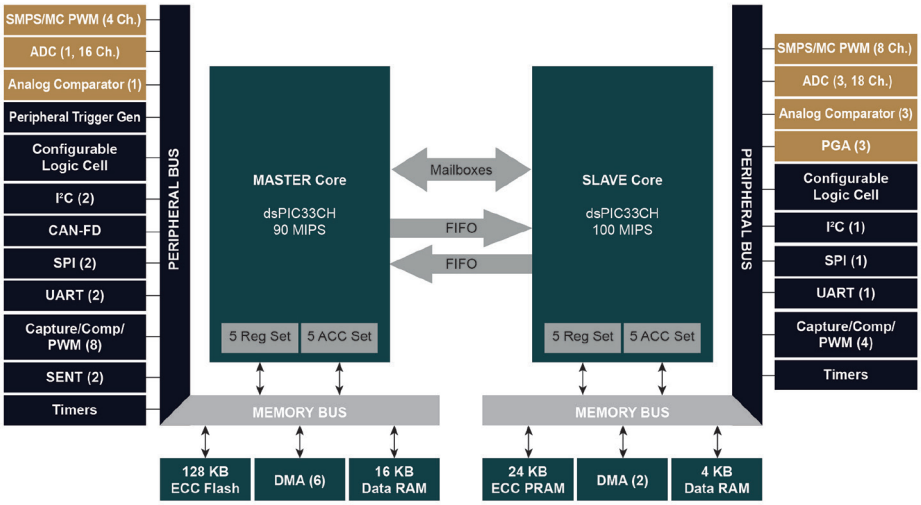
Un contrôleur double cœur aide à améliorer l'efficacité, à simplifier les efforts de développement et à réduire les coûts grâce aux avantages suivants :
- Il offre des performances accrues par rapport à un contrôleur à un seul cœur similaire qui fonctionnerait deux fois plus vite, et se révèle idéal pour les applications qui comportent au moins deux fonctions à contraintes temporelles.
- Il simplifie le développement logiciel grâce à deux cœurs indépendants qui permettent le développement logiciel avec des équipes dispersées sur le plan géographique, une intégration facile avec une coordination vraiment minime, et une personnalisation facile des fonctionnalités pour délivrer de multiples modèles d'une même ligne de produits (figure 1).

## Un contrôleur double cœur pour des performances accrues

Un contrôleur double cœur facilite l'intégration logicielle à un niveau élevé en permettant à différentes fonctions d'être exécutées sur deux cœurs indépendants. Ce qui se révèle très pratique si une application nécessite l'exécution périodique de deux fonctions à contraintes temporelles à un moment précis ou en réponse à des évènements asynchrones. Si chaque fonction à contraintes temporelles est exécutée sur un cœur différent, il n'y aura aucun conflit entre les fonctions. Le CPU est ainsi globalement mieux utilisé grâce à une réduction ou à l'absence totale de surcharge de commutation de contexte entre les fonctions. De nombreux contrôleurs double cœur sont livrés avec des ressources spécifiques qui réduisent les surcharges de commutation et d'arbitrage. Certains contrôleurs à double cœur intègrent également une mémoire PRAM spécifique rapide (Program RAM) associée à l'un des cœurs, en général le cœur esclave, ce qui améliore encore les performances. C'est pourquoi un contrôleur double cœur affiche des performances supérieures par rapport à un contrôleur monocœur fonctionnant à une vitesse deux fois plus élevée.

**1 SCHÉMA ÉLECTRONIQUE TYPIQUE D'UN CONTRÔLEUR À DEUX CŒURS INDÉPENDANTS**

Un contrôleur à deux cœurs offre des performances accrues par rapport à un contrôleur à un seul cœur similaire qui fonctionnerait deux fois plus vite, et se révèle idéal pour les applications qui comportent au moins deux fonctions à contraintes temporelles.

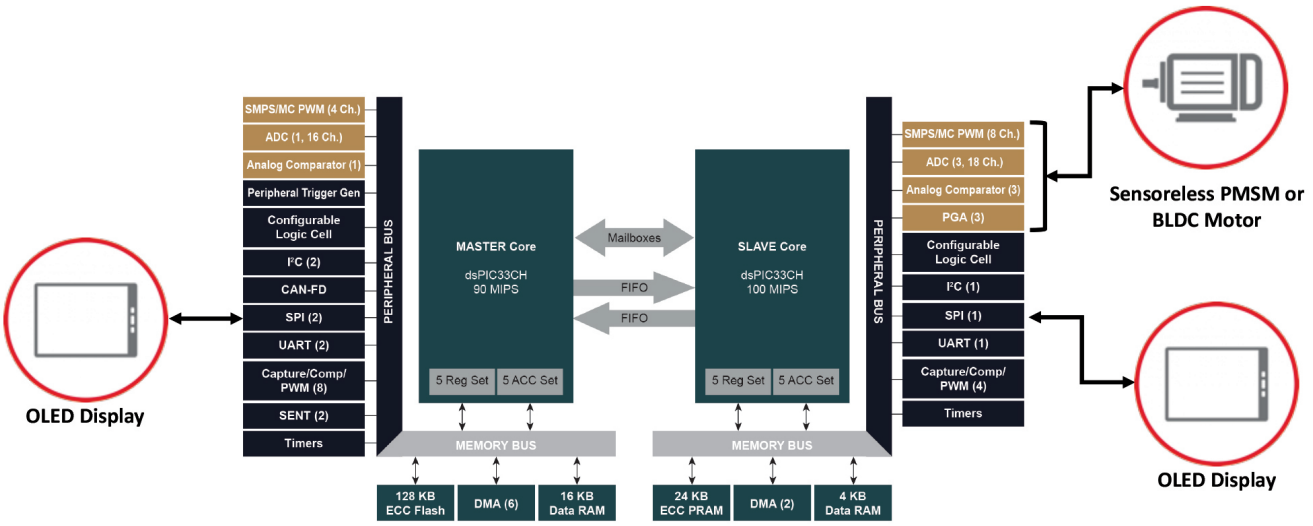



## Un contrôleur double cœur pour un développement simplifié

De nombreux contrôleurs double cœur offrent de la mémoire, des périphériques et un support de débogage sur chaque cœur. Un schéma de gestion des ressources flexibles permet par ailleurs l'attribution de ressources partagées à l'un ou l'autre des cœurs en fonction des besoins des applications.
Ce type d'architecture de microcontrôleur permet le développement de logiciels indépendants, avec très peu de coordination entres les experts des différents domaines, et facilite l'intégration. En particulier, les contrôleurs à double cœur simplifient l'intégration de deux piles logicielles reposant sur des architectures différentes ou requérant des ressources similaires du microcontrôleur, et qui peuvent dans ce cas tourner sur deux cœurs indépendants. Le processus est similaire au développement de piles à exécuter sur deux contrôleurs différents, mais avec les avantages des performances accrues, de l'utilisation optimale des ressources et de la réduction des coûts. Les complications inhérentes à l'intégration des piles sont ainsi évitées, tout comme celles liées à l'affectation de ressources sur des créneaux temporels différents et aux risques que cela suppose. Un contrôleur à double cœur facilite également le débogage post-intégration, étant donné que chaque cœur dispose de sa propre interface de débogage. Grâce à la très faible interdépendance entre les piles, il devient extrêmement simple d'isoler les problèmes et de les corriger lors du débogage. Grâce à ces nombreux avantages, les contrôleurs à double cœur réduisent significativement les risques de développement et accélèrent les délais de mise sur le marché.
On peut également ajouter à la liste des avantages des contrôleurs double

**2 SCHÉMA ÉLECTRONIQUE D'UNE APPLICATION COMPLEXE À FONCTIONS MULTIPLES**

Ici, l'un des cœurs met en œuvre la commande de moteur à l'aide d'un algorithme vectoriel FOC pour commander un moteur DC sans balais. Pour fournir une interface utilisateur graphique, l'autre coeur exécute une pile graphique pour connecter un écran OLED et met en œuvre les fonctions système pour connecter un potentiomètre et les boutons qui commandent la vitesse et l'état du moteur.




cœur qu'ils facilitent la personnalisation sans pour autant modifier les fonctionnalités principales. En définissant une architecture où les fonctionnalités principales s'exécutent sur l'un des cœurs, les fonctionnalités personnalisées peuvent être mises en œuvre sur l'autre cœur. Tous ces avantages des contrôleurs double cœur simplifient la conception logicielle, y compris lorsque plusieurs équipes dispersées aux quatre coins du monde sont impliquées, et permettent une intégration aisée avec très peu d'efforts de coordination.

## Un contrôleur double cœur pour une réduction des coûts

Grâce aux performances accrues qu'ils offrent, les contrôleurs à double cœur permettent aux développeurs de systèmes embarqués de réaliser des applications complexes en n'utilisant qu'un seul microcontrôleur. En simplifiant le développement, ils réduisent significativement le temps de conception et les risques associés et permettent de créer des systèmes concurrentiels grâce à des coûts réduits et à des délais de commercialisation plus rapides.

Pour bien se rendre compte des avantages des contrôleurs double cœur ci-dessus exposés, une petite expérience a été menée. Dans cette démonstration, l'un des cœurs (en général le coeur esclave) met en œuvre la commande de moteur à l'aide d'un algorithme vectoriel FOC pour commander un moteur DC sans balais. Pour fournir une interface utilisateur graphique, l'autre coeur (le coeur maître) exécute une pile graphique pour connecter un écran OLED et met en œuvre les fonctions système pour connecter un potentiomètre et les boutons qui commandent la vitesse et l'état du moteur (figure 2).

Pour faire la démonstration de la simplicité d'un composant double cœur, la pile graphique et le logiciel de commande de moteur ont été développés par deux équipes différentes, distantes sur le plan géographique. Grâce à la flexibilité de maintenir une architecture logicielle indépendante, les deux équipes n'ont pas vraiment eu besoin de se coordonner. L'une des équipes, spécialisée dans la commande de moteur, a pu mettre en œuvre très rapidement l'algorithme FOC pour commander le moteur BLDC. L'autre équipe étant spécialisée dans le développement d'interfaces utilisateur graphiques, les deux équipes ont pu mettre à profit leur expérience dans leur domaine respectif et terminer rapidement le projet. Très peu de coordination a suffi pour que les deux équipes établissent un accord qui transmette le statut des boutons et du potentiomètre entre les deux cœurs.

Pour cette expérience, les deux équipes ont utilisé les bibliothèques logicielles disponibles pour mettre en œuvre la commande de moteur et l'interface graphique. Le projet a été achevé en un rien de temps, et en passant très peu de temps à intégrer les deux piles existantes. Grâce aux excellentes performances du cœur, une grande partie de la bande passante du CPU est restée disponible sur les deux cœurs. Pour aller plus loin, une interface d'écran OLED a également été ajoutée sur le cœur esclave pour afficher en temps réel les paramètres du moteur sans pour autant affecter ses performances. Vous pouvez retrouver sur YouTube (https://www.youtube.com/watch?v=b8iHXJw_PEE) une vidéo de démonstration de l'application de commande de moteur avec l'interface graphique fonctionnant avec un contrôleur de signal numérique double cœur.

Le contrôleur de signal numérique (DSC, Digital Signal Controller) dsPIC33CH128MP508 de Microchip est un bel exemple de contrôleur double cœur qui offre tous ces avantages. Le dsPIC33CH double cœur affiche des performances accrues avec une mémoire dédiée et des périphériques spécifiques aux applications, le rendant idéal pour les applications embarquées hautes performances de commande de moteur et de conversion de puissance numérique. Le contrôleur double cœur de cette famille permet aux développeurs de mettre au point des firmwares pour différentes fonctions système séparément, et de les rassembler ensuite, sans que les blocs de code n'interfèrent entre eux. ■