SOLUTIONS EMBARQUÉES POUR APPLICATIONS HAUTE SÉCURITÉ

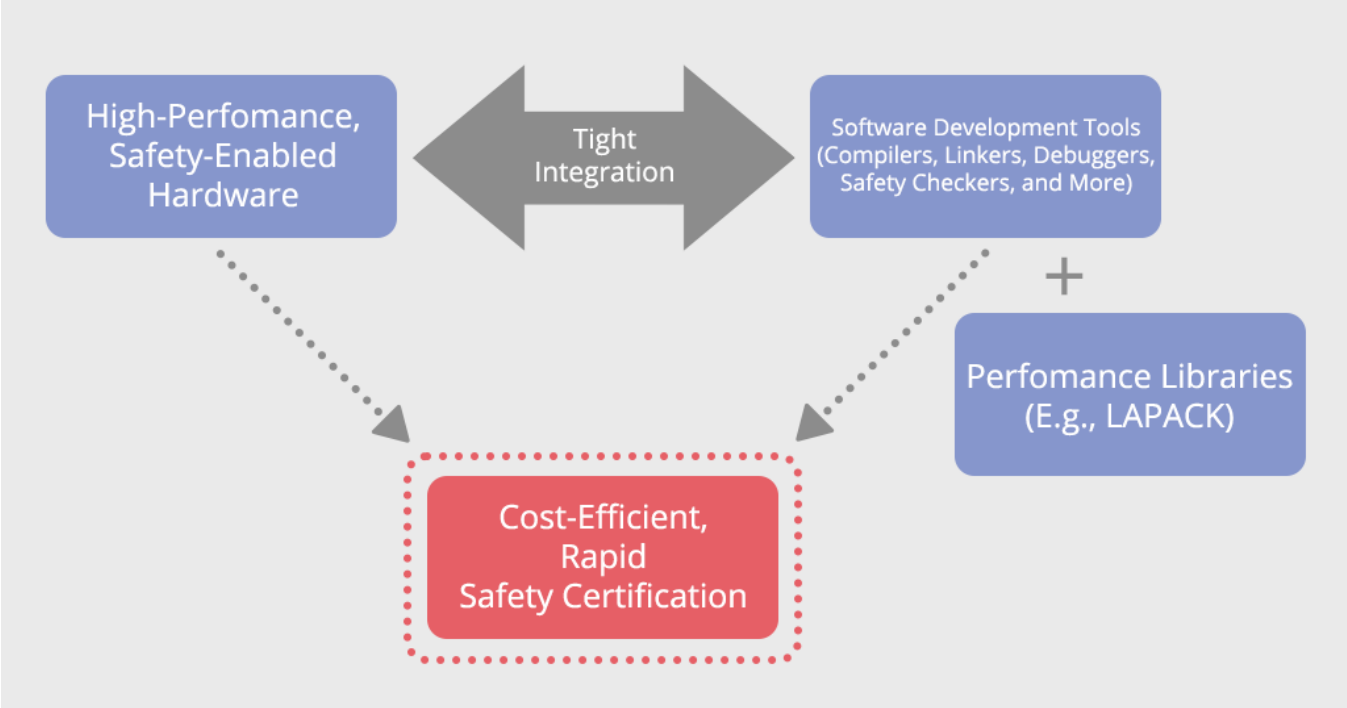
INTRODUCTION

Au début de l'ère automobile, les voitures étaient strictement mécaniques. Aujourd'hui, les fonctions d'une voiture sont de plus en plus contrôlées par des solutions embarquées, qui réalisent la synthèse des aspects matériels et logiciels. Par exemple, les modules de commande du groupe motopropulseur (PCM) contrôlent, entre autres, l'injection de carburant, la transmission et le freinage. Les constructeurs automobiles investissent dans une gamme de plus en plus large de systèmes avancés d'assistance au conducteur (ADAS), comme le cruise control adaptatif ou la détection d'angle mort.

La norme ISO 26262 fut créée pour optimiser la sécurité d'utilisation des applications destinées aux automobiles. Elle définit plusieurs niveaux d'intégrité de sécurité automobile (ASIL). Toute solution embarquée doit associer hardware et software sans perdre de vue la sécurité – il faut prévoir des procédures de secours, la redondance, l'autocontrôle et le test des composants, dans le respect de la classification ISIL de l'application concernée. La cotation ASIL d'une application est établie en mesurant les conséquences négatives entraînées par un échec de l'application (ou de l'un de ses constituants) dans différents scénarios. Les composants qui n'ont aucun impact sur la sécurité (par exemple, l'interface graphique d'une radio) ont une cotation ASIL plus basse que les fonctions ADAS dont la sécurité est primordiale. Dans la mesure où les systèmes de type ADAS prennent des décisions cruciales pour le conducteur, ces fonctions reçoivent généralement une cotation ASIL élevée – ASIL-D, par exemple.

En termes économiques, le marché mondial des UCE, les unités de commande électroniques (PCM inclus), devrait représenter près de 46 milliards en 2020 (1), tandis que le marché mondial des ADAS devrait connaître une croissance à deux chiffres entre 2014 et 2024, partant de 18,2 milliards (2). Voilà qui devrait encourager les fabricants d'équipement d'origine et les autres acteurs du secteur à intensifier leurs efforts pour l'innovation et le développement de ces systèmes.

Ceci étant, la certification de sécurité des ADAS est un processus long et difficile. La certification des systèmes peut être facilitée par le recours à des composants hardware (comme les microprocesseurs) et software (comme les compilateurs et les bibliothèques de performance de type LAPACK) pré-certifiés et testés en profondeur, ce qui contribue directement à améliorer la sécurité.



LES APPLICATIONS HAUTE SÉCURITÉ ONT BESOIN D'UN HARDWARE EMBARQUÉ AVANCÉ

Afin de réaliser des applications dont la sécurité est primordiale (comme pour les fonctions PCM et ADAS), les ingénieurs automobiles doivent s'assurer que leurs calculs sont exempts d'erreurs et réalisés dans un délai prédéfini. Une section de code ne peut en aucun cas interférer avec l'exécution d'autres sections, en changeant des données ou en consommant des ressources précieuses. En outre, certaines instructions doivent être traitées dans un ordre prédéfini. Le timing est donc très important. Les données provenant de capteurs différents (des radars et des caméras, par exemple) doivent être récoltées et intégrées en temps réel pour façonner une carte cohérente et sûre de l'environnement du véhicule.

Ces conditions strictes font du choix du hardware l'étape principale dans le développement d'un bon système de PCM ou ADAS. En étant particulièrement attentif à cet aspect, vous pouvez accélérer la certification et amener votre produit sur le marché dans le respect du délai et du budget prévu. Par exemple, le processeur innovant Infineon AURIX™ 2G TriCore™ est tout particulièrement adapté aux besoins de l'industrie automobile en matière de performance et de sécurité. Son architecture multicœurs innovante, basée sur jusqu'à six CPU 32-bits TriCore indépendants, a été conçue dans le respect des normes de sécurité les plus strictes, tout en offrant une amélioration significative de la performance. Les développeurs qui recourent au processeur AURIX 2G TriCore ménageront leurs efforts pour atteindre la norme ASIL-D par rapport à une architecture basée sur un processeur ancien. Les clients qui souhaitent lancer leur produit rapidement peuvent aujourd'hui réduire le coût du développement sécurisé de leur microcontrôleur (MCU) jusqu'à 30 %. (3)

Le processeur TriCore est optimisé pour les environnements où la sécurité est primordiale, et propose différentes fonctions liées à la sécurité que nous détaillerons ci-dessous.

Détection plus rapide des erreurs grâce à l'architecture multicœurs et au parallélisme lockstep

Avec le processeur AURIX 2G TriCore processor, l'exécution dite « lockstep », qui consiste à employer deux cœurs avec les mêmes instructions en parallèle pour détecter les erreurs, est disponible sur jusqu'à quatre cœurs sur six (figure 1). Cette technique permet d'obtenir une puissance de calcul inédite en toute sécurité et dans le respect de la norme ISO 26262 sur un seul dispositif intégré, et de réduire les efforts et les coûts de développement tout en observant la cotation ASIL-D applicable.

L'architecture parallèle d'AURIX est conçue pour contrôler et réduire l'impact des erreurs causées par l'isolation physique, la diversité d'exécution au niveau des instructions, ainsi que par la conception des circuits et les différences de timing au niveau des circuits. Des fonctionnalités spéciales ont été ajoutées, telles qu'un cycle double, une conception unique de réseaux de réinitialisation et d'horloge, ainsi qu'un comparateur lockstep au design innovant. Les cœurs parallèles AURIX ont été améliorés pour offrir une plus grande variété architecturale tout en réduisant les erreurs les plus courantes.

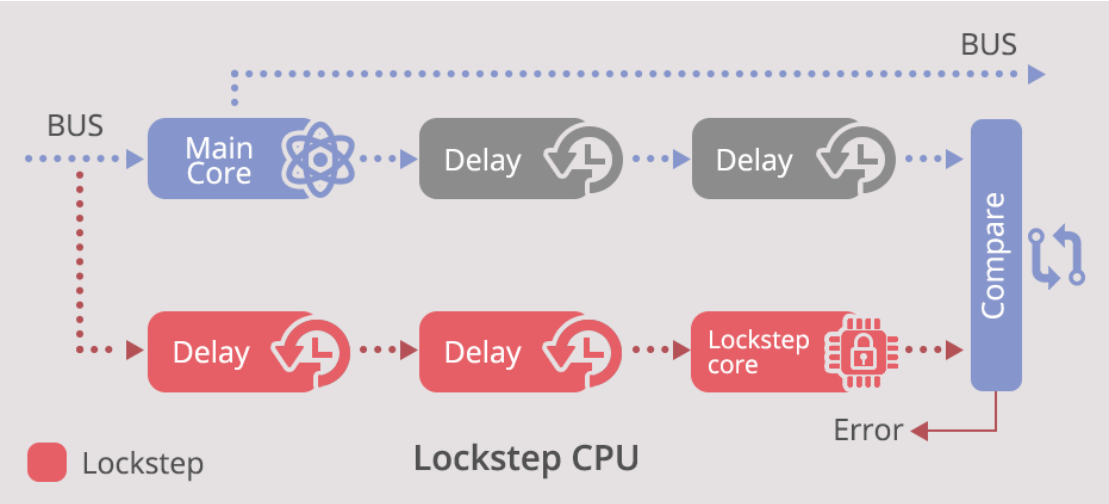


Figure 1. La diversité architecturale entre le cœur principal et les cœurs parallèles aide à réduire les défaillances les plus communes. Avec l'aimable autorisation de Infineon.

Affranchissez-vous des interférences

Si l'architecture multicœurs offre de nombreux avantages en matière de performances et de sécurité, elle doit également éviter les interférences : le logiciel assigné à une cotation ASIL donnée n'est pas censé interférer avec un logiciel de cotation ASIL supérieure. Cette absence d'influence doit être effective au niveau de l'espace (accès à la mémoire), de la communication et du timing. (4). Par exemple, le logiciel ne doit pas corrompre les données utilisées par d'autres composants software, et les applications logicielles qui partagent des périphériques ne doivent pas corrompre la configuration ou la production de ces périphériques. Puisque les cœurs peuvent partager des ressources comme les bus ou la mémoire, le timing de chaque application doit prendre en compte l'utilisation de ces ressources partagées.

Le processeur TriCore s'affranchit des interférences grâce à différentes méthodes:

- Au niveau de la mémoire et de la communication, l'unité de protection de la mémoire CPU de chaque cœur (CPU-MPU) régule le trafic sortant vers le bus.
- Les MPU qui gèrent le bus permettent de diviser la mémoire en affectant de manière statique des zones mémoire à des cœurs spécifiques.
- L'accès réservé RAP permet l'affectation statique de périphériques à des cœurs spécifiques au niveau du bus, ce qui évite que les autres cœurs n'aient accès à ces périphériques.
- Tous les registres de périphériques globaux en lien avec la sécurité sur le processeur sont protégés par Sendinit. Pour écrire dans ces registres, le cœur doit avoir accès au watchdog global.
- La gestion des erreurs s'enrichit des fonctions suivantes:
    - Des détecteurs spécifiques aux cœurs et définis par les utilisateurs traquent les chaînes d'opération illégales et règlent les problèmes
    - Des « signaux d'alarme » généraux signalent tout bit flipping ou surchauffe anormale des die, en connexion directe avec l'unité de gestion de la sécurité du matériel (SMU).
    - La prise en charge des alarmes logicielles définies par l'utilisateur
    - Une fonction d'arrêt d'urgence capable de faire basculer automatiquement les broches dans un état prédéfini sans intervention du logiciel

Un timing parfait grâce à un module timer générique intégré (GTM)

Le processeur AURIX 2G TriCore utilise un GTM à l'architecture unique, compatible avec les fonctions hardware pré-implémentées et avec un séquenceur multicanaux (MCS), un cœur de traitement simple et programmable. De cette façon, le GTM peut exécuter des fonctions complexes dans le logiciel tout en garantissant une latence basse et un timing parfaitement prévisible.

La conception modulaire du GTM facilite l'ajout et le retrait de fonctions dans un système donné en évitant d'affecter les autres, ce qui simplifie le développement et l'affranchit progressivement des plateformes. Le GTM peut être programmé en C ou dans un langage assembleur, à condition de choisir des outils dont les compilateurs prennent en charge le GTM.

Calcul rapide des résultats des transformations rapides de Fourier (FFT) effectuées sur le hardware

Les systèmes de détection radar utilisés dans de nombreuses applications ADAS utilisent les capacités de calcul FFT du microprocesseur, qui transforme les fréquences radar en informations spatiales. De la même façon, les PCM utilisent parfois les FFT pour surveiller la santé du moteur et analyser les vibrations. Par nature, les fonctions FFT nécessitent des calculs intensifs; le processeur TriCore offre un traitement haute efficacité du FFT, ce qui permet d'améliorer les performances des systèmes

radar. Les applications ADAS embarquées qui utilisent les capacités de calcul FFT du processeur TriCore sont conformes à la norme ISO 26262.

AMÉLIORER LA SÉCURITÉ GRÂCE À LA PROGRAMMATION MULTICŒURS

Les développeurs de l'industrie automobile ont accès à de nombreux outils de développement logiciel. Effectuer un choix parmi une telle offre est parfois délicat, mais il n'en est pas moins crucial pour réussir le développement d'une solution embarquée.

Choix d'un compilateur fortement intégré dans le hardware

Les développeurs automobiles qui utilisent le processeur AURIX 2G TriCore peuvent choisir entre plusieurs compilateurs C (ou assembleurs). Développer des critères pour choisir un compilateur peut aider à identifier le meilleur choix.

Une différence majeure réside dans la nature open source ou non du compilateur. Les compilateurs open source sont souvent peu coûteux, mais leurs inconvénients peuvent entraver le développement d'une solution embarquée haute sécurité. Les trois principaux problèmes liés aux compilateurs open source sont les suivants :

1. **Le compilateur n'est pas sous le contrôle du fabricant:** Le premier problème des compilateurs open source est que ceux-ci ne sont pas contrôlés par le fabricant. Si un bug est repéré, le client du compilateur voudra le voir réglé dans un délai raisonnable. En open source, le fabricant du compilateur n'a peut-être pas le personnel qualifié pour s'en occuper, car certaines parties du compilateur sont développées sur place et d'autres exploitent des technologies open source. Dès lors, la gestion des bugs peut prendre plus de temps.
2. **Les compilateurs open source ne prennent pas forcément en compte le hardware:** Les compilateurs génériques élaborés en open source ne tiennent généralement pas compte des capacités spécifiques du hardware, comme celles du processeur AURIX 2G TriCore, ou sont parfois incapables d'exploiter ces fonctions. Par exemple, de nombreux compilateurs open source ne sont pas compatibles avec le GTM du processeur TriCore, tandis que d'autres sont incapables de tirer parti du code compilé pour les architectures multicœurs. Les compilateurs prennent en charge ces fonctions à l'aide d'extensions de langage, que l'on peut installer plus facilement si le compilateur n'est pas open source.
3. **Les compilateurs officiels sont généralement mieux codés:** Enfin, les compilateurs entièrement élaborés par une équipe dédiée offrent souvent de meilleures performances avec le code compilé. Par exemple, pour obtenir la plus petite empreinte mémoire et le code le plus rapide possible, le linker choisi doit vous permettre de gérer quel code accède à quels blocs mémoire. L'architecture mémorielle du processeur TriCore propose différent types de mémoire et une protection ECC, pour un maximum de flexibilité dans la conception des applications (figure 2). On peut accéder à la mémoire bloc-note (ou mémoire « proche ») sur chaque cœur sans utiliser le bus, ce qui accélère l'accès à la mémoire. Les mémoires flash on-die et RAM ajoutent de la mémoire et des capacités, même si leur accès n'est pas aussi aisé qu'à la mémoire bloc-note. À moins que le compilateur et le linker ne soient compatibles avec un mappage rapide des objets correspondants dans chaque type de mémoire, l'architecture flexible de la mémoire du processeur TriCore est inutile et entrave les performances de l'application.

Il en va de même pour la programmation d'une architecture multicœurs. L'encodage dédié à un processeur multicœurs est beaucoup plus compliqué que la programmation destinée à un seul cœur, et nécessite que le linker autorise le développeur à assigner des instructions spécifiques à des cœurs donnés, à l'aide des ID de ces cœurs.

Opter pour un compilateur fortement intégré avec le hardware a un autre avantage : la disponibilité d'une assistance qualifiée, que l'on peut trouver directement dans le compilateur ou auprès des sociétés de hardware. Au lieu de devoir se débrouiller seuls, les développeurs peuvent poser leurs questions. En outre, il est parfois possible d'obtenir rapidement des améliorations du produit, ce qui est rarement possible dans la communauté open source, qui a tendance à adopter les nouvelles technologies moins vite.

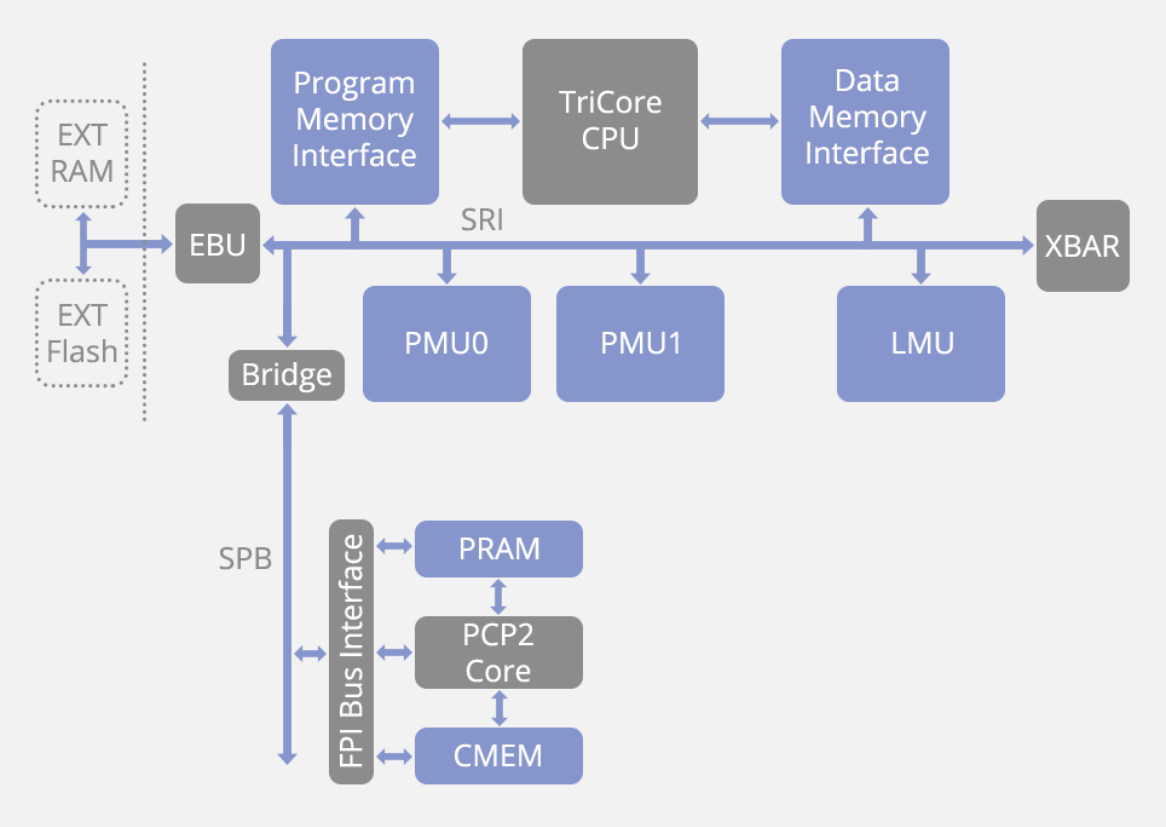


Figure 2. L'utilisation d'une combinaison compilateur/linker capable de tirer parti de l'architecture mémorielle du processeur aide les architectes logiciel à optimiser la performance de leurs applications.

Le code est rarement parfait du premier coup. Même les meilleurs développeurs de l'industrie automobile ne peuvent pas envisager tous les scénarios possibles. C'est pourquoi il est important d'investir dans des outils de développement performants pour déboguer et vérifier la sécurité des logiciels. Par exemple, il est possible d'obtenir des produits qui effectuent des tests de sécurité sur le code. Ces tests permettent d'éviter les interférences en débusquant les problèmes du MPU, très difficiles à repérer avec des tests classiques.

POURQUOI CHOISIR L'ASPICE ?

L'Automotive SPICE® (ASPICE) est un cadre de référence pour la conception et l'évaluation des procédés de développement de logiciels pour automobiles. L'efficacité des implémentations permet d'obtenir de meilleurs procédés et d'améliorer la qualité des produits. Opter pour un compilateur certifié ASPICE de niveau 2 vous garantit que le produit a été développé au moyen de procédés approuvés, dans le respect des normes de sécurité requises. Ainsi, votre application sera plus probablement exempte d'erreurs.

Le recours à un compilateur certifié ASPICE permet non seulement d'obtenir du code de meilleure qualité, mais aussi d'économiser de l'argent. Les coûts liés à la correction des erreurs augmentent de façon exponentielle au fur et à mesure que le projet avance (figure 3). (5)

Concept	$ 1,300
A sample	$ 4,550
B sample	$ 5,200
C sample	$ 7,800
PV series	$ 84,500
Production	$ 104,000
Post Production	$ 117,000

Figure 3. L'utilisation d'outils de développement sûrs et certifiés ASPICE pour corriger les erreurs au début d'un projet permet de réduire significativement les coûts liés à la correction de ces problèmes (données récoltées auprès d'Audi, BMW, Daimler, Porsche et Volkswagen).

CRÉEZ VOTRE APPLICATION HAUTE SÉCURITÉ SUR DES BASES SOLIDES

La sécurité est une préoccupation de plus en plus importante dans l'industrie automobile, à mesure que les fonctions ADAS et PCM se multiplient et dépendent de processus et de logiciels embarqués de plus en plus complexes. En réaction, le secteur définit des spécifications hardware et software et des normes de sécurité de plus en plus strictes. Le budget est également un poste important pour la plupart des développeurs d'applications haute sécurité et pour les fabricants OEM. Ils doivent élaborer des solutions sécurisées dans le respect du budget et des délais pour rester compétitifs. Pour obtenir des performances optimales dans un environnement haute sécurité, les développeurs ont besoin de hardware certifié, ainsi que de compilateurs respectant la norme ASPICE et optimisés pour le hardware utilisé.

RÉFÉRENCES

(1) Analyse du marché des ECU pour l'automobile par application et prévisions par segment jusqu'en 2020, https://www.marketresearchandstatistics.com/ad/automotive-ecu-market-analysis-by-application-powertrain-chassis-electronics-safety-security-entertainment-communication-navigation-and-segment-forecasts-to-2020/

(2) ADAS : en route vers l'adoption généralisée http://embedded-computing.com/articles/adas-the-road-widespread-adoption/

(3) "TriCore™ Architecture & Core," http://www.infineon.com/cms/en/product/microcontroller/32-bit-tricore-tm-microcontroller/tricore-tm-architecture-and-core/channel.html?channel=ff80808112ab681d0112ab6b73d40837

(4) Infineon, AURIX Safety Manual AP32224 V1.2, Infineon Munich, 2015

(5) Études de cas simples et leçons de l'ASPICE, https://www.ibm.com/developerworks/community/files/basic/anonymous/api/library/9e66f5de-701e-4994-9291-75646d558240/document/d88a5328-3d2a-4e82-9a3f-e4e5bf9f41be/media/Session22-A-SPICE%20compliance%20made%20easy%20Case%20studies%20and%20lessons%20learned.pdf

EN SAVOIR PLUS

https://adas.org.au/need-adas-qualification/

https://www.lifewire.com/advanced-driver-assistance-systems-534859