Compas'2023 : Parallélisme/ Architecture / Système
LISTIC/USMB - Annecy France, 4-7 juillet 2023

TrustSoC : Architecture SoC hétérogène légère et efficace sécurisée par conception

Raphaële Milan[1], Lilian Bossuet[1], Loic Lagadec[2], Carlos Andres Lara-Nino[1]

[1] Université Jean Monnet Saint-Etienne, CNRS,
Laboratoire Hubert Curien UMR 5516
Saint-Etienne, France
prenom.nom@univ-st-etienne.fr

[2] Lab-STICC
ENSTA Bretagne
Brest, France
loic.lagadec@ensta-bretagne.fr

Résumé
Au cours des dernières années, les SoC (System-on-a-Chip) hétérogènes embarquant des processeurs à plusieurs cœurs et de la logique programmable ont progressé en terme de complexité et hétérogénéité. D'un point de vue sécurité, cela entraîne une augmentation de la surface d'attaque exploitable par un attaquant pour prendre le contrôle du système et/ou avoir accès à des données sensibles. Pour adresser ce problème, dans cet article, nous proposons les bases d'une architecture de SoC hétérogène de confiance sécurisée par conception appelée TrustSoC. Nous montrons que la sécurité ne doit pas être ajoutée après design, mais plutôt pensée depuis la phase de conception. Nous démontrons aussi que cette sécurité doit considérer tous les composants du SoC : matériels et logiciels. Nous basons notre proposition sur l'extension de la technologie ARM TrustZone, des contrôleurs de communication, des règles de fonctionnement et une isolation entre les composants logiciels et matériels et les partitions mémoires.

Mots-clés : FPGA-SoC, TrustZone étendue, contrôleurs de communication, AXI-4

1. Introduction

Les SoC hétérogènes sont présents dans de nombreux domaines applicatifs du fait de la flexibilité qu'ils offrent. Cela peut aller d'applications générales pour le grand public aux applications sensibles du domaine militaire : trading haute fréquence, Cloud, télécommunications militaires, etc. Pour s'adapter à un plus grand nombre d'applications, le nombre et la diversité des composants au sein du SoC ont augmenté. Parmi les SoC les plus complexes se trouvent les SoC-FPGA (comme le SoC-FPGA Intel Agilex family, et Zynq UltraScale+™ MPSoC de AMD Xilinx) auxquels nous nous intéressons dans cet article. Cependant, les concepts présentés dans ce papier sont transposables aux autres types de SoC.
Cet article présente une solution centrée sur le bus de communication et basée sur des petits contrôleurs distribués de communication pour sécuriser les SoC hétérogènes. Afin d'estimer le coût de la solution proposée, nous avons prototypé l'architecture de la solution proposée, appelée TrustSoC, avec un SoC AMD-Xilinx-Zynq-7000. Nous donnons des résultats expérimentaux d'implémentation et de performances avec quatre IP matérielles différentes pour des applications de traitement du signal et de cryptographie.
Le papier est organisé de la manière suivante : la section 2 décrit le contexte de ce travail. La section 3 présente le modèle de menaces que l'on considère dans cet article. La section 4

présente les différentes stratégies de protection du bus de communication du SoC et traite les avantages et inconvénients de chaque solution. La section 5 présente la solution proposée de SoC hétérogène sécurisée par conception. La section 6 présente les résultats expérimentaux dont l'analyse conclut l'article.

2. Contexte

Les SoC sont de plus en plus utilisés pour traiter des données sensibles, ainsi ils deviennent des cibles privilégiées pour des entités malveillantes. Les finalités de ces entités sont de voler de l'information ou de la modifier, de pirater le système ou de le désactiver. Les attaques qu'elles mettent en œuvre ciblent la partie logicielle. Celles-ci sont rendues possibles, en partie, à cause du partage des différentes ressources du SoC entre les différentes applications logicielles. Par exemple, dans des architectures récentes de SoC multi-processeurs, le dernier niveau de cache est partagé entre les différents cœurs. Un logiciel malveillant peut utiliser cette fonctionnalité à son avantage et peut déterminer grâce à une analyse des temps d'accès à la mémoire cache si l'application qu'il cible a accédé à certaines données [13]. En finalité, cela fournit à l'adversaire des renseignements utiles sur l'application cible. Récemment, Bossuet et Benhani ont montré que ce type d'attaque était aussi possible sur les SoC-FPGA et qu'il est possible d'exploiter les vulnérabilités de la mémoire cache depuis la PL (Partie Logique) pour attaquer une application logicielle qui s'exécute dans la PS (partie Processing System) [5]. Pour sécuriser l'exécution d'applications dans les SoC modernes, il est commun d'utiliser des solutions d'exécution sécurisée comme la technologie ARM TrustZone utilisée par exemple dans les SoC AMD-Xilinx [1],[10]. Cette solution sépare les ressources du processeur en deux mondes différents : un monde sécurisé et un monde non-sécurisé. La fig.1 (en annexe) donne un exemple typique de cette technologie appliquée à un SoC-FPGA : les blocs de couleur foncée représentent le monde non sécurisé et les blocs de couleur claire représentent le monde sécurisé.
Un bit d'information (le bit « NS ») permet au système de déterminer dans quel monde il se trouve. Comme l'illustre la fig.1 (en annexe), la stratégie de partitionnement d'AMD-Xilinx exploitant la technologie ARM TrustZone est appliquée à la PS (chaque cœur CPU peut exécuter des applications logicielles d'un des deux mondes), aux mémoires et à la PL (chaque IP intégrée à la PL est liée à un des deux mondes). Le bit NS est transmis via le bus AXI (Advanced eXtensible Interface) pour permettre à la PL de savoir quel monde (sécurisé ou non sécurisé) exécute l'application logicielle en cours. Chaque transaction de lecture et/ou écriture sur le bus AXI porte l'information du bit NS afin d'empêcher les ressources non sécurisées (dans la PS, PL et les mémoires) d'accéder aux ressources sécurisées. Le code et les données contenus dans le monde sécurisé sont supposés être protégés des intrus. Cela pourrait sembler être une approche de sécurité efficace. Malheureusement, des travaux récents ont montré qu'en dépit de la sécurité logicielle amenée par cette solution de sécurité, beaucoup de vulnérabilités peuvent être exploitées pour effectuer des attaques et corrompre le partitionnement de sécurité [3],[4],[6] et [11].
Ces récentes attaques montrent qu'il n'est pas suffisant de considérer la sécurité depuis un seul point de vue : logiciel ou matériel. Il faut plutôt considérer la sécurité comme un procédé holistique. Les solutions de sécurité doivent être mûrement réfléchies et considérer les deux facteurs : logiciels (OS, Boot) et matériels (architecture, PL, système matériel). Cependant, pour assurer un haut niveau de sécurité, la première étape devrait être de définir un modèle de menaces avant d'étudier les principales solutions architecturales disponibles dans la littérature. Nous proposons un tel modèle dans la section suivante.

3. Modèle de menaces

Dans cet article, nous envisageons plusieurs menaces liées aux attaques logicielles et matérielles à distance. Nous considérons les menaces sur la mémoire par des attaques de type analyse des temps d'accès aux mémoires cache depuis la PS et depuis la PL, mais aussi les accès illégitimes et les modifications du contenu de la mémoire. Nous envisageons une application logicielle malveillante qui tente d'accéder à des renseignements sensibles provenant d'autres applications logicielles ou d'IP matérielles. Nous envisageons une IP matérielle malveillante essayant d'accéder à des informations sensibles provenant d'autres IP matérielles ou d'applications logicielles. Nous considérons que l'attaquant peut essayer d'effectuer des accès illégaux au monde sécurisé à partir du monde non sécurisé en manipulant de manière malveillante le bus AXI et les périphériques.
Par conséquent, l'exécution logicielle proposée par la technologie TrustZone n'est pas suffisante lorsque l'architecture n'est pas sécurisée par conception. Il ressort que la communication à l'intérieur du SoC est l'élément de sécurité clé. Par conséquent, la section suivante étudie la possibilité de sécuriser le SoC par l'architecture de communication selon ce modèle de menaces.

4. Contrôleurs de communication sécurisés

Dans cette section, nous présentons différentes propositions d'architectures issues de la littérature pour sécuriser les bus de communication de SoC en utilisant des contrôleurs de communication de bus. En effet, les vulnérabilités associées aux bus de communication peuvent être atténuées en utilisant un contrôleur de communication qui va analyser toutes les transactions demandées et déterminer leurs légitimités. Pour cela, le contrôleur de communication applique des politiques de sécurité. Ces politiques stipulent les spécifications d'accès (lecture seule, lecture-écriture, non accessible) et les formats des données. Après avoir vérifié ces paramètres, le contrôleur autorise ou empêche l'accès à la ressource. Dans la littérature nous pouvons identifier trois architectures principales de contrôleurs de communication. La fig.2 (en annexe) illustre ces trois architectures nommées architecture centralisée, architecture distribuée et architecture mixte. Elles sont présentées dans les sous-sections suivantes.

4.1. L'approche centralisée

La première approche, (a) dans fig.2 (en annexe), utilise un unique contrôleur de communication, d'où le nom d'architecture centralisée. Ce contrôleur analyse toutes les transactions. Nous pouvons trouver des exemples de cette implémentation dans [7] et [8]. Le principal inconvénient de ce type d'architecture est la latence induite par le contrôleur unique. En effet, seul un contrôleur traite toutes les requêtes. Cependant, cette approche permet d'éviter la duplication des contrôleurs de communication si les ressources ne le permettent pas.

4.2. L'approche distribuée

La seconde approche, (b) dans fig.2 (en annexe), est l'approche distribuée. Dans cette architecture, un contrôleur de communication est instancié à chaque connexion avec le bus de communication. Les travaux [9] et [14] illustrent des exemples de ce type d'architecture. L'un des avantages principaux de cette approche est le gain de performances du système. Plusieurs contrôleurs de communication peuvent analyser simultanément les transactions demandées contrairement à l'approche centralisée. Un autre avantage de cette approche est que les transactions sont analysées avant d'êtres propagées sur le bus. Si elles sont déterminées illégitimes elles ne sont pas transmises. En contrepartie, ce type d'architecture nécessite plus de ressources.

4.3. L'approche mixte

La troisième et dernière architecture, fig.2 (c) (en annexe), est une combinaison des deux approches précédentes. Elle dispose d'un contrôleur de communication centralisé et de contrôleurs distribués à chaque point de connexion avec le bus de communication. Du fait de la présence du contrôleur centralisé, les contrôleurs distribués se retrouvent plus petits que dans l'architecture distribuée. Le contrôleur centralisé est lui aussi moins complexe que dans l'approche centralisée puisqu'une partie de ses tâches est déléguée aux petits contrôleurs distribués. Des exemples d'implémentation de cette architecture sont présentés dans [2] et [12]. En comparaison avec l'approche centralisée, le principal avantage de cette architecture est la présence des petits contrôleurs distribués qui permettent de réduire le problème de latence. Mais, d'un autre côté, il y a une duplication de ressources donc elles partagent le même inconvénient pour l'utilisation de ressources.

4.4. Synthèse des différentes approches

La table 1 (en annexe) synthétise les différentes propositions d'architectures pour les contrôleurs de communication suivant les différents aspects : surface, performances temporelles et sécurité. L'évaluation qualitative qui découle de la table 1 (en annexe) montre que l'architecture qui offre le meilleur compromis entre tous les critères (performances temporelles, ressources et sécurité) est l'architecture distribuée. C'est donc ce type d'architecture que nous sélectionnons pour l'architecture TrustSoC que nous présentons dans la suite de cet article.

5. Proposition d'architecture

Dans cette section, nous proposons une architecture sécurisée par conception de SoC hétérogène appelée TrustSoC. TrustSoC est une architecture personnalisable qui peut être ajustée aux besoins du concepteur. Comme cela est présenté dans fig.3 (en annexe), TrustSoC dispose d'une PS multi-cœurs (deux cœurs dans l'exemple de la fig.3), une PL et une zone mémoire et des périphériques. Cependant, il existe une différence importante avec le partitionnement sécurisé du système proposé par la technologie TrustZone [1] et celui utilisé dans notre proposition. Au lieu d'un seul monde sécurisé, nous envisageons plusieurs mondes sécurisés séparés (comme exemple, deux mondes sécurisés sont illustrés dans fig.3 (en annexe)). Ils sont identifiés avec différents niveaux de gris clair. Le bit NS utilisé pour TrustZone est étendu à un multi-mondes *world ID*. Dans l'exemple donné dans fig.3 (en annexe), le *world ID* est encodé sur deux bits : «00» pour le monde non sécurisé, «01» pour le premier monde sécurisé, «10» pour le deuxième monde sécurisé. Cet encodage est arbitraire et est amené à évoluer avec le nombre de mondes sécurisés choisi. TrustSoC intègre aussi une PL, divisée en régions non-sécurisées et sécurisées intégrant les IP matérielles du concepteur. Dans ces différentes régions, chaque IP matérielle possède un identifiant qui est géré par un wrapper de sécurité dédié (s_wrapper sur fig. 3 (en annexe)) et qui est automatiquement inclus par l'outil CAO et fixé au moment de la conception. Comme discuté dans la section 4.4, l'architecture distribuée sélectionnée pour TrustSoC, offre le meilleur compromis pour les contrôleurs de communication du SoC. Les accès sécurisés à la mémoire sont également gérés par un wrapper de sécurité dédié avec un ID spécifique et les partitions mémoires sont isolées physiquement entre elles. Cela signifie que les applications logicielles ou les IP matérielles qui appartiennent à un des mondes sécurisés ne peuvent pas accéder aux données d'un autre monde sécurisé sans permission. Évidemment, les applications logicielles ou les IP matérielles appartenant au monde non sécurisé ne peuvent pas accéder aux données des mondes sécurisés. Ces restrictions d'accès sont gérées par les contrôleurs de communication. TrustSoC embarque plusieurs mémoires : Flash, RAM et ROM qui ne sont pas

partagées. Chaque monde a une partition dans chaque wrapper de sécurité qui encapsule les différentes mémoires et leurs politiques de sécurité.

5.1. Fonctionnement dans un monde non sécurisé

Une application logicielle s'exécutant dans le monde non sécurisé ou une IP matérielle appartenant à ce monde ne peuvent pas communiquer directement avec une application logicielle ou une IP matérielle appartenant à un des mondes sécurisés sans leurs autorisations. Une telle communication pourrait apparaître lorsqu'un processus (application logicielle ou IP matérielle) exécuté dans un monde sécurisé doit déléguer certains calculs à une IP matérielle non sécurisée. Dans ce cas, après la fin du traitement, l'IP matérielle est automatiquement réinitialisée par son wrapper de sécurité ce qui empêche la réutilisation des données sensibles. Pour illustrer ce fonctionnement, chaque ressource liée au monde non sécurisé est représentée en couleur gris foncé sur fig.3 (en annexe) et fig.4 (en annexe). La figure 4 (en annexe) décrit le partitionnement des ressources de l'architecture présentée sur la fig.3 (en annexe) lorsque celle-ci fonctionne depuis un monde sécurisé (fig.4 a (en annexe)) et lorsque celle ci fonctionne avec un monde non sécurisé (fig.4 b (en annexe)).

5.2. Fonctionnement dans un des mondes sécurisés

Plusieurs différences apparaissent quand nous considérons le fonctionnement depuis un des mondes sécurisés. Un processus d'un des mondes sécurisés peut avoir accès à une application logicielle ou une IP matérielle depuis un monde non sécurisé afin d'accélérer un traitement non critique si cela a été autorisé par le design. Une application logicielle ou une IP matérielle d'un monde sécurisé ne peut pas accéder à des partitions mémoires d'autres mondes ou les modifier sans autorisation préalable. Les autorisations sont délivrées par les wrappers de sécurité. Chaque wrapper de sécurité des IP matérielles appartenant à un des deux mondes sécurisés réinitialise automatiquement l'IP après traitement pour empêcher un attaquant d'exploiter des données résiduelles. Cette règle est aussi appliquée pour les partitions de la mémoire cache. Enfin, chaque échange qui n'a pas été autorisé au préalable par les wrappers de sécurité dédiés du SoC ne peut pas avoir lieu.

6. Prototype d'implémentation matérielle et résultats expérimentaux

Dans cette section, nous présentons les résultats expérimentaux matériels obtenus sur un SoC AMD-Xilinx avec un Zynq-7000 avec la version 2020.2 du logiciel de conception Xilinx Vivado.

6.1. Contrôleurs de communication distribués « wrappers de sécurité »

Pour le prototype, la PL est réalisée avec quatre IP avec pour chacune un wrapper de sécurité dédié comme présenté dans la fig.5 (en annexe). Chaque IP matérielle a un identifiant (*IP ID* dans la fig.5 (en annexe)), une liste de permissions et un *world ID* qui sont tous codés matériellement dans le wrapper de sécurité. Les listes sont personnalisables par le concepteur et sont fixées à l'étape de synthèse. Chaque fois qu'une transaction est émise, l'identifiant source est transmis ainsi que le *world ID* en utilisant les signaux utilisateurs AXI-4. Cela permet une transmission sans surcharge supplémentaire puisque ces signaux sont intégrés dans le protocole AXI-4. Dans notre implémentation, l'identifiant est codé sur trois bits et le *world ID* sur deux. En se basant sur le prototype matériel implémenté (fig.5 (en annexe)), nous évaluons différents scénarios selon les autorisations de l'IP :
1-a. *Fonctionnement normal* : Dès lors que les IP ont la permission de communiquer, les données sont envoyées à l'IP destinataire une fois que le wrapper de sécurité autorise l'accès.
1-b. *Fonctionnement anormal* : Dès lors que les IP n'ont pas la permission de communiquer, le

wrapper de sécurité prend la main et répond à la place. Il force l'écriture du code d'erreur SLVERR sur le bus AXI et ne transmet jamais la requête à l'IP.

6.2. Conversion entre PS et PL

Dans la section précédente, nous avons présenté différents scénarios associés avec le fonctionnement de la PL, mais nous avons aussi implémenté une IP pour faire la conversion entre les signaux AXI de la PS et les signaux AXI de la PL comme illustré dans la fig.5 (en annexe). Cette IP, *Conversion_PL_PS*, a pour objectif d'agir comme un wrapper de sécurité global pour la PS. Elle ajoute un identifiant à chaque communication de la PS puisqu'elle ne dispose pas d'interface AXI-4. L'IP transmet ensuite la transaction au bon wrapper de sécurité selon l'identifiant demandé par la PS. Elle transmet aussi le *world ID* de la transaction. Nous avons évalué différents scénarios selon les droits d'accès :
2-a. *Fonctionnement normal* : Les permissions sont vérifiées par le wrapper de sécurité et les données sont transmises à l'IP.
2-b. *Fonctionnement anormal* : Les droits ne sont pas accordés, le wrapper de sécurité force l'écriture du code d'erreur SLVERR sur le bus AXI. Les données ne sont jamais transmises à l'IP.

6.3. Résultats expérimentaux

Afin d'estimer les coûts matériels du wrapper de sécurité, nous avons implémenté quatre IP matérielles dédiées à des applications de cryptographie ou de traitement du signal. La taille de ces IP varie de 2566 à 5951 LUT. La table 2 (en annexe) fournit des résultats expérimentaux. Toutes les IP matérielles utilisées sont disponibles en ligne (https ://opencores.org - https ://github.com/emse-sas-lab/SCAbox-ip) et [15]. Nous avons évalué nos implémentations matérielles, avec et sans le wrapper de sécurité, selon l'utilisation des ressources matérielles en nombre de LUT et en nombre de registres. Nous avons aussi évalué les performances temporelles avec une estimation de la fréquence maximale de fonctionnement avec et sans le wrapper de sécurité fournit par l'outil Vivado. Les trois dernières colonnes de la table 2 (en annexe) donne le surplus (en pourcentage) selon les trois critères (en nombre de LUT, en nombre de registres et fréquence maximale) de l'implémentation de l'IP matérielle avec le wrapper de sécurité par rapport à celle sans le wrapper de sécurité. Le surplus en ressources induit par le wrapper de sécurité en terme de nombre de LUT est de 8.8% au maximum et en nombre de registres il est de 21% au maximum. En terme de fréquence maximum de fonctionnement, le wrapper de sécurité ne change pas le chemin critique de l'IP matérielle et donc, ne change pas la fréquence maximum de fonctionnement (les fluctuations figurant au tableau 2 (en annexe) sont dues au processus de synthèse de l'outil Vivado). En conclusion, le wrapper de sécurité a un coût très limité en terme de ressources matérielles et un coût négligeable en terme de performances temporelles avec une amélioration importante de la sécurité.

7. Conclusion

Cet article a posé les bases d'une architecture SoC hétérogène de confiance sécurisée par conception appelée TrustSoC. Nous avons démontré que la sécurité doit être soigneusement pensée dès la phase de conception de l'architecture. Cette sécurité doit aussi être double : logicielle et matérielle. Ce travail conduit à la proposition d'une architecture sécurisée par conception efficace et légère. Il s'agit aussi d'une architecture personnalisable selon les besoins du concepteur et fixée à l'étape de conception. Le prototypage que nous avons réalisé permet d'estimer que les coûts matériels et en performance de TrustSoC sont limités pour un apport indéniable en termes de sécurité.

Bibliographie

1. Alves (T.) et Felton (D.). – Trustzone : Integrated hardware and software security. vol. 3, pp. 18–24.
2. Basak (A.), Bhunia (S.) et Ray (S.). – A flexible architecture for systematic implementation of SoC security policies. – In *2015 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*.
3. Benhani (E. M.) et Bossuet (L.). – DVFS as a security failure of TrustZone-enabled heterogeneous SoC. – In *2018 25th IEEE International Conference on Electronics, Circuits and Systems (ICECS)*, pp. 489–492.
4. Benhani (E. M.), Bossuet (L.) et Aubert (A.). – The security of ARM TrustZone in a FPGA-based SoC. vol. 68, n8. – Conference Name : IEEE Transactions on Computers.
5. Bossuet (L.) et Benhani (E. M.). – Security assessment of heterogeneous SoC-FPGA : On the practicality of cache timing attacks. – In *2021 IFIP/IEEE 29th International Conference on Very Large Scale Integration (VLSI-SoC)*. – ISSN : 2324-8440.
6. Bossuet (L.) et Lara Nino (C.). – Advanced covert-channels in modern SoCs.
7. Brunel (J.), Pacalet (R.), Ouaarab (S.) et Duc (G.). – SecBus, a software/hardware architecture for securing external memories. – In *2014 2nd IEEE International Conference on Mobile Cloud Computing, Services, and Engineering*. IEEE.
8. Coburn (J.), Ravi (S.), Raghunathan (A.) et Chakradhar (S.). – SECA : security-enhanced communication architecture. – In *Proceedings of the 2005 international conference on Compilers, architectures and synthesis for embedded systems*, *CASES '05*, CASES '05. Association for Computing Machinery.
9. Cotret (P.), Crenne (J.), Gogniat (G.) et Diguet (J.-P.). – Bus-based MPSoC security through communication protection : A latency-efficient alternative. – In *2012 IEEE 20th International Symposium on Field-Programmable Custom Computing Machines*, pp. 200–207.
10. Gosain (Y.) et Palanichamy (P.). – TrustZone technology support in zynq-7000 all programmable SoCs.
11. Gross (M.), Jacob (N.), Zankl (A.) et Sigl (G.). – Breaking TrustZone memory isolation and secure boot through malicious hardware on a modern FPGA-SoC.
12. Nasahl (P.), Schilling (R.), Werner (M.) et Mangard (S.). – HECTOR-v : A heterogeneous CPU architecture for a secure RISC-v execution environment. – In *Proceedings of the 2021 ACM Asia Conference on Computer and Communications Security*, *ASIA CCS '21*, ASIA CCS '21. Association for Computing Machinery.
13. Osvik (D. A.), Shamir (A.) et Tromer (E.). – Cache attacks and countermeasures : The case of AES. *In : Topics in Cryptology – CT-RSA 2006*, éd. par Pointcheval (D.). – Springer Berlin Heidelberg.
14. Siddiqui (F.), Hagan (M.) et Sezer (S.). – Pro-active policing and policy enforcement architecture for securing MPSoCs. – In *2018 31st IEEE International System-on-Chip Conference (SOCC)*. – ISSN : 2164-1706.
15. Sutter (G. D.), Deschamps (J.-P.) et Imana (J. L.). – Modular multiplication and exponentiation architectures for fast RSA cryptosystem based on digit serial computation. vol. 58, n7. – Conference Name : IEEE Transactions on Industrial Electronics.

Annexes

TABLE 1 – Synthèse des différentes architectures de contrôleurs de communication pour un SoC hétérogène

	Critères d'évaluation des différentes approches		
Architecture	Surface	Performances temporelles	Sécurité
Centralisé	+	- -	-
Distribuée	-	+ +	+
Mixte	-	+	+

TABLE 2 – Résultats d'implémentation pour les différentes IP de open cores, github et [15] avec ou sans le wrapper de sécurité

	IP seule			IP avec security wrapper			Différence [%]		
IP	LUT	Registres	Fmax (MHz)	LUT	Registres	Fmax (MHz)	LUT	Registres	Fmax (MHz)
Edge Sobel	2566	3919	219	2792	4357	220	+8,8	+11,2	0
AES	2978	1685	128	3148	2039	126	+5,7	+21,0	-1.6
Karatsuba multiplier 128 bit	4770	3701	194	5034	4043	210	+5,5	+9,2	+8,2
Montgommery multiplier 224 bit	5951	2278	79	6198	2684	78	+4,2	+17,8	-1,3

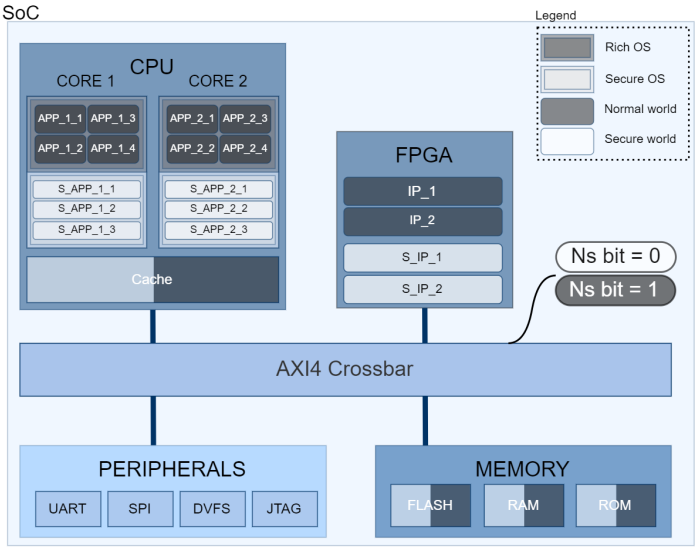

FIGURE 1 – Architecture de SoC hétérogène intégrant la technologie ARM-TustZone avec le monde non sécurisé en gris foncé et le monde sécurisé en blanc

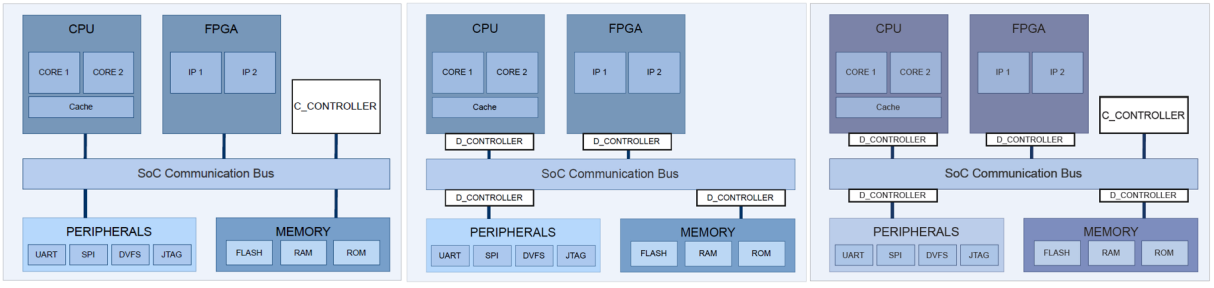

FIGURE 2 – Différentes architectures de contrôleurs de communication : (a) centralisée ; (b) distribuée ; (c) mixte

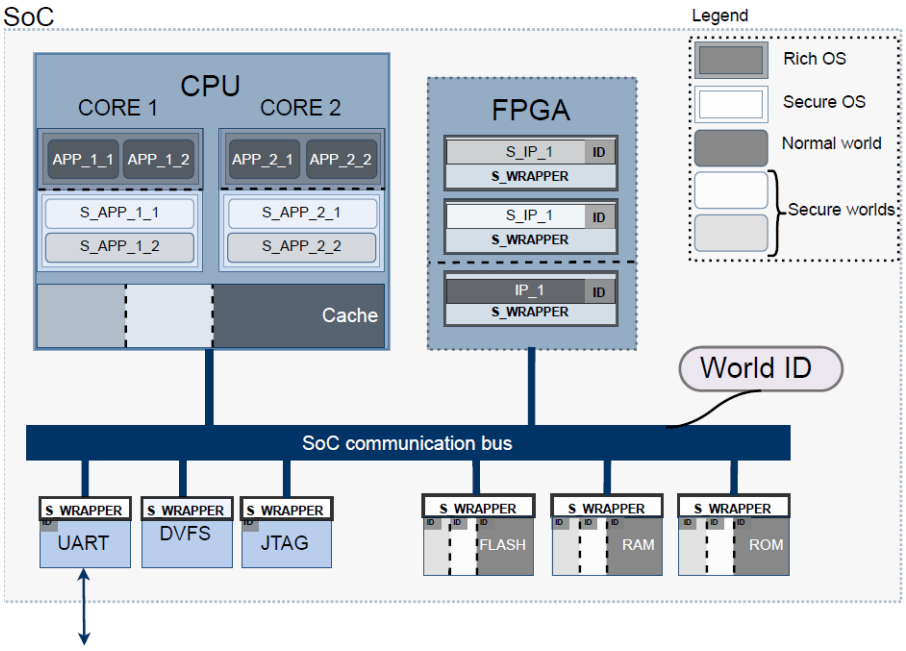

FIGURE 3 – Architecture de SoC hétérogène sécurisée «TrustZone étendue»

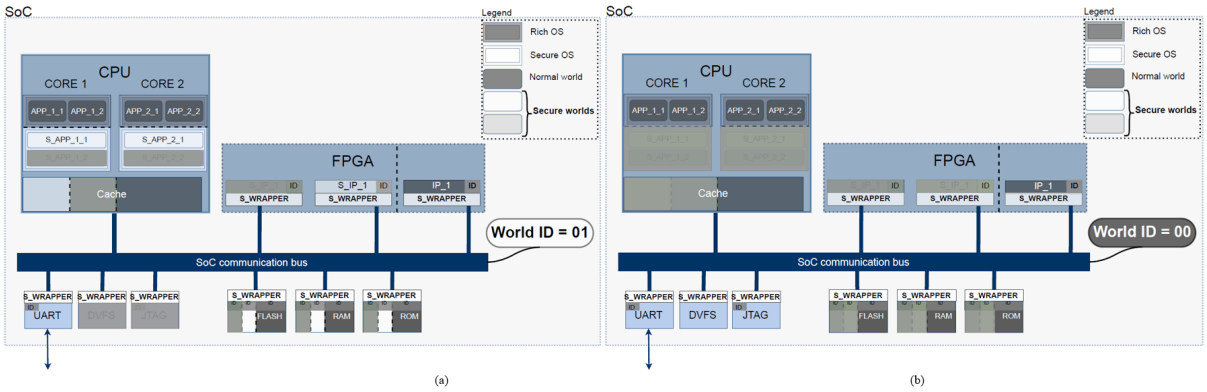

FIGURE 4 – Architecture de TrustSoC en fonctionnement dans un des mondes sécurisés (a) et dans le monde non sécurisé (b)

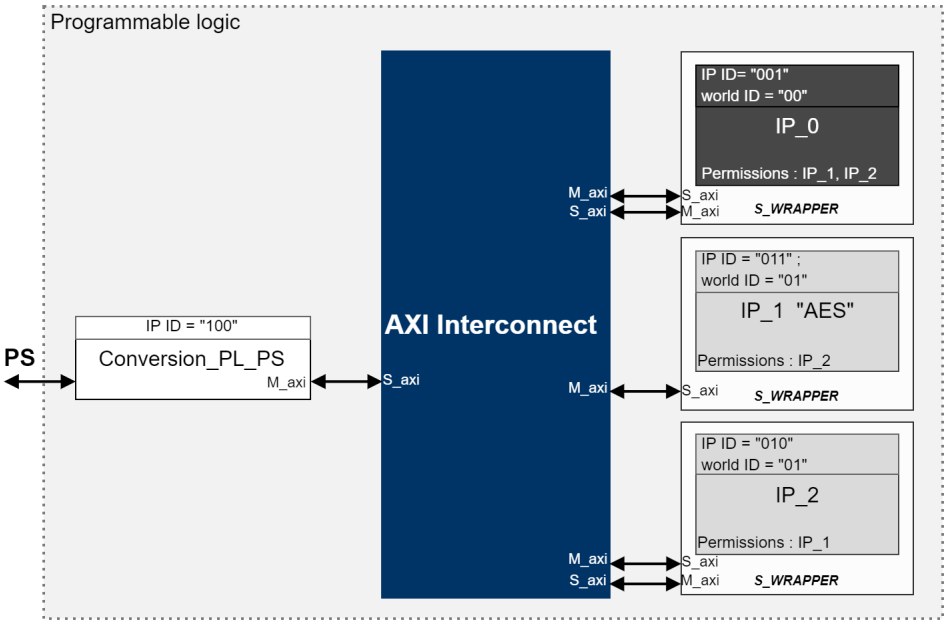

FIGURE 5 – Architecture matérielle de la PL de TrustSoC