(19) RÉPUBLIQUE FRANÇAISE

INSTITUT NATIONAL DE LA PROPRIÉTÉ INDUSTRIELLE

PARIS

(11) **N° de publication :** **2 977 694**

(à n'utiliser que pour les commandes de reproduction)

(21) **N° d'enregistrement national :** **11 56210**

(51) Int Cl[8] : ***G 06 F 21/52*** *(2013.01)*

(12) DEMANDE DE BREVET D'INVENTION A1

(22) **Date de dépôt :** 08.07.11.

(30) **Priorité :**

(43) **Date de mise à la disposition du public de la demande :** 11.01.13 Bulletin 13/02.

(56) **Liste des documents cités dans le rapport de recherche préliminaire :** *Se reporter à la fin du présent fascicule*

(60) **Références à d'autres documents nationaux apparentés :**

(71) **Demandeur(s) :** *STMICROELECTRONICS (ROUSSET) SAS Société par actions simplifiée* — FR.

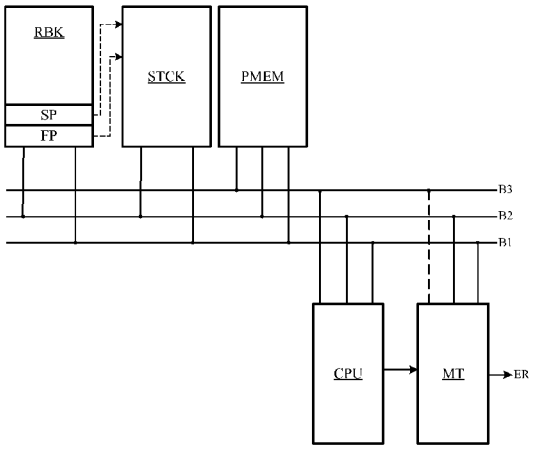
(72) **Inventeur(s) :** ORLANDO WILLIAM et GUILLEMIN PIERRE.

(73) **Titulaire(s) :** STMICROELECTRONICS (ROUSSET) SAS Société par actions simplifiée.

(74) **Mandataire(s) :** OMNIPAT Société anonyme.

(54) MICROPROCESSEUR PROTEGE CONTRE UN DEBORDEMENT DE PILE.

(57) L'invention concerne un microprocesseur comprenant une unité centrale (CPU), au moins une pile d'exécution (STCK), un pointeur de pile (SP), un bus d'adresse (B1) et un bus de données (B2). Le microprocesseur comprend également un moniteur hardware (MT) configuré pour fournir des codes témoins (C1, C2), insérer les codes témoins dans la pile ou laisser l'unité centrale les insérer, puis générer un signal d'erreur (ER) en réponse à une tentative de modification d'un code témoin présent dans la pile.



FR 2 977 694 - A1

MICROPROCESSEUR PROTEGE CONTRE UN DEBORDEMENT DE PILE

La présente invention concerne la protection d'un microprocesseur contre un débordement de pile. La pile d'exécution ("call stack") d'un microprocesseur, appelée généralement "pile" ("stack"), est une zone mémoire volatile dédiée à l'enregistrement de données concernant des fonctions exécutées par le microprocesseur. La pile permet notamment de mémoriser une adresse de retour à laquelle le microprocesseur doit revenir après exécution d'une fonction.

Ainsi, lorsqu'une première fonction ou "fonction appelante" appelle une seconde fonction ou "fonction appelée", la fonction appelante place son adresse de retour dans la pile, et la fonction appelée, quand elle termine l'exécution de la tache pour laquelle elle a été conçue, récupère l'adresse de retour dans la pile. Lorsque des fonctions s'appellent mutuellement, des adresses de retour s'accumulent dans la pile et sont récupérées les unes après les autres à chaque fin d'exécution d'une fonction. En sus des adresses de retour, la pile reçoit des données associées aux fonctions appelées ou appelantes, telles que des variables locales de la fonction appelée, des paramètres de la fonction appelée, un pointeur de trame de la fonction appelante, etc.

Lorsqu'un trop grand nombre d'information est enregistré dans la pile, il se produit un phénomène appelé "débordement de pile" ("stack overflow"). Il s'agit d'un débordement externe à la pile, c'est-à-dire en dehors de l'espace mémoire attribué à la pile. Il existe également un phénomène de débordement interne, lorsque l'écriture d'une zone de données déborde sur une autre zone de données, notamment une zone contenant une adresse de

retour. Un tel débordement interne peut être causé sciemment par un attaquant cherchant à prendre le contrôle du microprocesseur, et peut permettre de remplacer l'adresse de retour pointant vers la fonction appelante par une adresse de retour pointant vers un programme malicieux. Dans la présente demande, le terme "débordement de pile" désigne un débordement interne.

Une méthode connue pour contrer les attaques par débordement de pile consiste à insérer dans la pile des codes témoins, parfois appelés "canaris". De tels codes témoins sont généralement de petits entiers de valeur aléatoire, placés dans la pile à des endroits stratégiques, de préférence avant chaque adresse de retour. Pour s'assurer que la pile n'a pas subi de débordement frauduleux, on s'assure que la valeur du code témoin n'a pas changé, avant que la fonction appelée utilise l'adresse de retour présente dans la pile.

Cette technique permet d'augmenter considérablement la difficulté d'exploiter un débordement de pile, car elle oblige l'attaquant à prendre le contrôle du pointeur d'instruction par des moyens complexes.

Cette technique n'est toutefois pas infaillible car elle repose entièrement sur la prévision, dans le programme exécuté par le microprocesseur, d'instructions d'insertion et de vérification de codes témoins. Une attaque sur le programme lui-même pourrait donc permettre une attaque subséquente sur la pile en neutralisant les instructions de vérification des codes témoins.

Également, une lecture d'un code témoin pourrait permettre à un fraudeur de provoquer un débordement de pile non détectable, en faisant en sorte que le débordement de pile réécrive la bonne valeur du code

témoin à son emplacement initial tout en modifiant l'adresse de retour.

Il pourrait donc être souhaité de renforcer la sécurité offerte par l'insertion de codes témoins dans la pile d'exécution d'un microprocesseur.

A cet effet, des modes de réalisation de l'invention concernent un microprocesseur comprenant une unité centrale, au moins une pile d'exécution, un pointeur de pile, un bus d'adresse et un bus de données, e un moniteur hardware configuré pour : générer des codes témoins, insérer des codes témoins dans la pile ou laisser l'unité centrale les insérer, mémoriser des adresses de codes témoins insérés dans la pile, et générer un signal d'erreur en réponse à une tentative de modification d'un code témoin présent dans la pile.

Selon un mode de réalisation, le moniteur est également configuré pour générer le signal d'erreur en cas de tentative de lecture d'un code témoin dans la pile.

Selon un mode de réalisation, le moniteur est configuré pour générer des codes témoins aléatoires ou pseudo-aléatoires.

Selon un mode de réalisation, le moniteur est configuré pour générer des codes témoins déterministes et reproductibles.

Selon un mode de réalisation, le moniteur est configuré pour surveiller le bus d'adresse, et générer le signal d'erreur si une adresse de code témoin mémorisée apparaît sur le bus d'adresse.

Selon un mode de réalisation, le moniteur comprend un premier registre accessible en écriture pour l'unité

centrale, et est configuré pour générer un code témoin et appliquer la valeur du code témoin sur le bus de données en réponse à une écriture d'une donnée dans le registre par l'unité centrale.

Selon un mode de réalisation, le moniteur est également configuré pour insérer le code témoin dans la pile, à une adresse présente dans le premier registre.

Selon un mode de réalisation, le moniteur est configuré pour, en réponse à une demande de suppression de code témoin de l'unité centrale ou du programme exécuté par l'unité centrale : lire le code témoin dans la pile, à une adresse spécifiée par la demande de suppression, comparer le code témoin avec une valeur attendue du code témoin, et générer le signal d'erreur si la valeur lue est différente de la valeur attendue.

Selon un mode de réalisation, le moniteur est configuré pour supprimer lui-même le code témoin dans la pile, après l'avoir vérifié.

Selon un mode de réalisation, le moniteur comprend un second registre accessible en écriture pour l'unité centrale, et est configuré pour interpréter une écriture du second registre comme une demande de suppression d'un code témoin.

Selon un mode de réalisation, le moniteur est configuré pour interpréter une écriture du second registre comme une demande de suppression d'un code témoin à une adresse présente dans le second registre.

Des modes de réalisation de l'invention seront décrits plus en détail dans ce qui suit se référant à titre non limitatif aux figures jointes parmi lesquelles :

- la figure 1 représente un microprocesseur comprenant un moniteur de pile selon l'invention,
- la figure 2 représente un exemple de contenu de la pile, et
- la figure 3 représente un exemple de réalisation du moniteur de pile.

La figure 1 représente un mode de réalisation d'un microprocesseur selon l'invention. Le microprocesseur comprend un processeur CPU ("Central Processing Unit") appelé par la suite "le CPU", une mémoire programme PMEM, une pile d'exécution STCK, et une banque de registres RBK. Ces différents éléments sont reliés à un bus d'adresse B1, un bus de données B2 et un bus d'instruction B3.

La banque de registre RBK comprend un registre pointeur de pile SP ("Stack Pointer") et un registre pointeur de trame FP ("Frame pointer"). Le pointeur de pile contient l'adresse du haut de la pile STCK et le pointeur de trame contient l'adresse de début d'une trame d'une fonction en cours d'exécution. La mémoire programme PMEM contient un programme exécuté par le CPU.

Il sera noté que le bus d'instruction B3 est en soi un élément optionnel du microprocesseur, sa prévision étant fonction de l'architecture retenue, ici de type Harvard. Une architecture de type Von Neumann ne comporterait que le bus B2 pour véhiculer à la fois les données et les instructions.

Le microprocesseur comporte également un moniteur de pile MT selon l'invention. Le moniteur MT est configuré pour placer des codes témoins ("canaris") dans la pile STCK lors de son remplissage par le CPU. Le moniteur est également configuré pour détecter des tentatives d'écriture de la pile aux adresses où se trouvent les

codes témoins, soit des tentatives d'altération des codes témoins, et, de préférence, de détecter également des tentatives de lecture des codes témoins.

Dans un mode de réalisation, le moniteur surveille uniquement le bus d'adresse B1 et déclenche une alerte lorsque l'adresse d'un code témoin apparaît sur le bus d'adresse B1. Dans un tel cas, le moniteur ne cherche pas à déterminer s'il s'agit d'une tentative d'écriture ou de lecture, et ne surveille pas le bus d'instruction B3 (ou le bus B2 dans une architecture de type Von Neumann). L'alerte est inconditionnelle et est par exemple émise sous forme de signal d'erreur ER.

Le signal ER est appliqué à un décodeur d'interruption qui envoie le CPU dans un sous-programme sécurisé de traitement des erreurs (par exemple interruption sécurisée sur plateforme dite "trustzone", dont le traitement s'effectue en mode sécurisé, dans des mémoires protégées). Alternativement, le signal ER est utilisé pour provoquer la remise à zéro (reset) du CPU.

Lors du retour à une fonction appelante, le moniteur MT assure aussi la levée de la surveillance du code témoin, avant sa suppression. Avant de lever la surveillance du code témoin, le moniteur vérifie que la valeur du code témoin n'a pas été altérée. A cet effet, le moniteur lit le code témoin et le compare à une valeur initiale qu'il a conservée dans une mémoire interne. Dans un mode de réalisation, le moniteur assure lui-même la suppression du code témoin dans la pile. Dans un autre mode de réalisation, cette suppression est assurée par le CPU.

Ainsi, le moniteur MT forme un moyen hardware de génération et de surveillance des codes témoins qui ne peut être corrompu par une altération frauduleuse du programme exécuté par le CPU. Il forme une sorte

d'arbitre impartial indépendant du programme lui-même et conférant un haut niveau de sécurité dans la génération et la surveillance des codes témoins.

La figure 2 montre un exemple d'insertion de codes témoins dans la pile STCK et un exemple de contenu de celle-ci. Une flèche DIR1 indique le sens des adresses croissantes et une flèche DIR2 indique le sens de remplissage de la pile. Le remplissage s'effectue ici des adresses de plus forte valeur vers les adresses de plus faible valeur. Le "haut" de la pile, correspondant à la valeur courante du pointeur de pile SP, correspond donc ici à l'adresse de plus faible valeur de la pile. Il sera noté que dans d'autres modes de réalisation, la pile pourrait avoir un sens de remplissage inverse, correspondant au sens des adresses croissantes.

Dans l'exemple représenté, la pile contient la trame FFA d'une fonction FA, la trame FFB d'une fonction FB, et la trame FFC d'une fonction FC en cours d'exécution. On suppose que la fonction FC a été appelée par la fonction FB et que la fonction FB a été appelée par la fonction FA. Chaque trame contient des données contextuelles de la fonction concernée et des données de retour à la fonction appelante.

Ainsi, la trame FFB de la fonction FB comprend une adresse RAFA de retour à la fonction FA, une valeur FPFA du pointeur de trame de la fonction FA, et des variables locales LVFB de la fonction FB. Un code témoin C1 a été inséré par le moniteur MT dans la trame FFB, par exemple entre l'adresse de retour RAFA et la valeur FPFA du pointeur de trame.

La trame FFC de la fonction FC comprend une adresse RAFB de retour à la fonction FB, une valeur FPFB du pointeur de trame de la fonction FB, et des variables locales LVFC

de la fonction FC. Un code témoin C2 a été inséré par le moniteur MT dans la trame FFC, par exemple entre l'adresse de retour RAFB et la valeur FPFB du pointeur de trame.

La valeur courante du pointeur de pile SP désigne le haut de la pile (adresse la plus basse), et la valeur courante du pointeur de trame désigne l'emplacement de l'adresse RAFB de retour vers la fonction FB.

Les codes témoins C2 et C1 sont surveillés en temps réel par le moniteur MT. Ainsi, toute tentative de débordement de la pile visant à écraser l'adresse de retour RAFB ou les adresses de retour RAFA et RAFB implique une tentative d'écriture de la pile aux emplacements des codes témoins C2 et C1. Cette tentative est détectée par le moniteur MT et le conduit à émettre le signal d'erreur ER. De même, toute tentative de lecture des codes témoins est de préférence détectée par le moniteur qui émet également le signal d'erreur.

Pour l'utilisation du moniteur MT, des instructions d'insertion et de suppression de codes témoins sont prévues dans le programme exécuté par le CPU (programme enregistré dans la mémoire programme PMEM).

La mise en œuvre de modes de réalisation de l'invention peut faire l'objet de diverses variantes qui seront évoquées dans ce qui suit, avant de décrire un exemple détaillé de réalisation du moniteur MT en relation avec la figure 3.

Modification du programme pour utiliser le moniteur

L'utilisation du moniteur MT suppose l'insertion dans le programme exécuté par le CPU d'instructions de placement

de codes témoins dans la pile, et d'instructions de suppression de codes témoins.

On entend par "instruction de suppression de code témoin" une instruction qui conduit le moniteur MT à lever la surveillance du code témoin visé par l'instruction, en vue de sa suppression par le CPU, ou qui le conduit à lever la surveillance du code témoin et à le supprimer de la pile lui-même.

A cet effet, plusieurs options peuvent être prévues :
- une modification explicite du programme par le programmeur (appel à des macros ou des fonctions dédiées),
- une modification du programme par le compilateur pour que cette opération soit transparente au programmeur,
- la prévision d'une étape de post-compilation sur le code binaire généré, toujours pour rendre l'opération transparente au programmeur,
- la modification du compilateur afin que le programmeur puisse indiquer au moyen d'une commande dite de "preprocessing" (par exemple une commande de type "pragma" spécifiée par la norme du langage C) comment le compilateur doit se comporter pour compiler une section de programme ou une fonction. Cela permet de faire ajouter par le compilateur des instructions d'insertion et de suppression de codes témoins, en laissant le choix au programmeur de décider quelles sections de programme ou quelles fonctions doivent être protégées.

Génération des codes témoins

La valeur d'un code témoin peut être :
- arbitraire et fournie par un générateur aléatoire ou pseudo-aléatoire),
- déterministe et reproductible : la valeur du code témoin est par exemple déterminée par le moniteur en

fonction de variables connues telles que l'adresse à laquelle est placé le code témoin, et d'une valeur secrète, éventuellement aléatoire, connue du moniteur seulement, ou encore d'un identifiant de la fonction appelante fourni par le compilateur.

La valeur du code témoin comprend de préférence un octet à 0 pour assurer une protection contre les failles liées aux manipulations de chaînes de caractères.

Insertion des codes témoins dans la pile

Les codes témoins sont placés entre les zones de variables locales et les adresses de retour. Il est en effet souhaitable qu'ils soient situés entre les zones sensibles au débordement et les zones à protéger d'un débordement. Dans un mode de réalisation, des codes témoins additionnels sont placés à d'autres endroits de la pile pour réaliser l'équivalent d'un "champ de mines" offrant des perspectives d'application allant au-delà de la simple protection des adresses de retour.

L'insertion d'un code témoin dans la pile peut être déclenchée par le moniteur :

- automatiquement, sur détection d'une instruction de type "call" provoquant un changement de contexte dans la pile. Dans ce cas, le moniteur est relié au bus d'instruction B3, comme montré en traits pointillés sur la figure 1, et comprend des moyens de décodage d'une telle instruction (dans une architecture de type Von Neumann, cette surveillance serait assurée sur le bus B2);
- sur demande explicite du programme. Une demande explicite peut consister en une instruction spécifique d'insertion d'un code témoin, que le moniteur doit décoder, ou plus simplement une instruction d'écriture d'un registre du moniteur, interprétée par celui-ci comme

une instruction d'insertion, comme cela sera décrit plus loin en relation avec la figure 3.

Gestion des adresses des codes témoins

Pour surveiller les différents codes témoins placés dans la pile, le moniteur doit savoir où ils se trouvent. A cet effet, les adresses des codes témoins sont conservées dans une mémoire interne du moniteur ou sont stockées dans une mémoire externe (zone mémoire du CPU) accessible par l'intermédiaire des bus d'adresse et de donnée. Alternativement, une mémoire interne de faible capacité peut être prévue pour stocker les adresses des codes témoins les plus récemment introduits dans la pile, et une mémoire complémentaire être prévue pour stocker les adresses des autres codes témoins. On distingue alors les codes témoins "actifs" dont les adresses sont sous surveillance car chargées dans la mémoire interne du moniteur, et des codes témoins "inactifs" dont les adresses ne sont pas surveillées. Lors d'un changement de contexte, les adresses des codes témoins présentes dans la mémoire externe sont transférées dans la mémoire interne pour être surveillées par le moniteur. Les codes témoins dont les adresses sont transférées deviennent alors "actifs".

Les adresses stockées dans une mémoire externe sont de préférence protégées en confidentialité et intégrité, par exemple en étant chiffrées et associées à un code de correction d'erreur.

De même, les valeurs des codes témoins, si elles ne sont pas déterministes et reproductibles, peuvent être conservées dans une mémoire interne du moniteur ou être stockées dans une mémoire du CPU, sous une forme protégée en confidentialité et en intégrité.

Surveillance des adresses des codes témoins

Le moniteur doit tout d'abord détecter et empêcher une tentative d'écriture de la pile aux adresses des codes témoins. Il peut s'agir d'une tentative de modification faite par le CPU ou par un dispositif maître pouvant accéder au bus de données et d'adresse.

De préférence, le moniteur détecte et empêche également toute tentative de lecture de la pile à une adresse contenant un code témoin. En effet, certaines attaques nécessitent la connaissance préalable de la valeur des codes témoins.

A cet effet, le moniteur compare l'adresse courante présente sur le bus d'adresse aux adresses des codes témoins "actifs".

La surveillance effectuée par le moniteur peut concerner les adresses de tous les codes témoins insérés dans la pile (surveillance exhaustive) ou seulement les adresses du ou des codes témoins associés à la tache courante (fonction en cours d'exécution). Dans le second cas, le moniteur doit être informé du changement de la tache courante, pour mise à jour des adresses de codes témoins à surveiller.

La surveillance de l'adresse courante est assurée par exemple par des comparateurs recevant chacun l'adresse courante sur une entrée et recevant sur une autre entrée l'une des adresses de codes témoins à surveiller. Le temps de détection est de préférence constant et de faible valeur. Le nombre maximum de codes témoins surveillés simultanément peut être limité au nombre de comparateurs inclus dans le moniteur. Un comparateur à valeurs multiples peut aussi être prévu, pour

successivement comparer l'adresse actuelle avec chacune des adresses des codes témoins.

Si la surveillance ne concerne que les codes témoins de la tache courante, le moniteur recharge ses comparateurs internes lors d'un changement de contexte, en leur appliquant des adresses de codes témoins associés à la nouvelle tache.

Les nouvelles adresses appliquées aux comparateurs sont importées d'une mémoire externe après vérification d'intégrité et correction d'erreur, ou sélectionnées dans une zone de sa mémoire interne qui est associée à la tache considérée.

Suppression d'un code témoin

La suppression d'un code témoin comprend une étape préalable de vérification de la valeur du code témoin par le moniteur, par comparaison avec une valeur attendue.

A cet effet, le moniteur lit le code témoin dans la pile et la compare à une valeur attendue. Si la valeur du code témoin n'est pas celle attendue, le moniteur émet le signal d'erreur ER.

La suppression d'un code témoin peut être déclenchée sur détection d'une instruction de type "return" sur le bus d'instruction B3 (ou sur le bus B2 dans une architecture de type Von Neumann), ou par l'intermédiaire d'une requête explicite présente dans le programme, indiquant l'adresse du code témoin à supprimer, ou encore par l'intermédiaire d'un accès à un registre du moniteur, comme cela sera décrit dans ce qui suit en relation avec la figure 3.

La figure 3 représente un exemple de réalisation du moniteur MT suivant les lignes directrices qui viennent d'être décrites.

Le moniteur MT comprend un circuit de contrôle CCT relié au bus d'adresse B1 et au bus de données B2 du CPU, une mémoire volatile CAM, une mémoire volatile CVM, des comparateurs d'adresse CA0, CA1,... CAi et un comparateur de code témoin CDT. Le circuit de contrôle CCT est un circuit à logique câblée de type machine d'états ("state machine"). Il est équipé d'un générateur aléatoire ou pseudo-aléatoire CGEN, et de deux registres R1, R2 accessibles en écriture pour le CPU. La mémoire CAM est prévue pour mémoriser des adresses de codes témoins, tandis que la mémoire CVM est prévue pour mémoriser des valeurs de codes témoins. Les sorties des comparateurs CA0-CAi ainsi que la sortie du comparateur CDT sont envoyées dans une porte G1 de type OU dont la sortie fournit le signal d'erreur ER.

La mémoire CAM a une structure de type table de correspondance ("look up table") et comporte N sorties parallèles fournissant, en mode lecture, i+1 adresses de codes témoins associées à un index appliqué à l'entrée de la mémoire. Chaque valeur d'un code témoin fournie par la mémoire CAM est appliquée sur une entrée d'un comparateur CA0-CAi dont l'autre entrée est reliée au bus d'adresse B1.

La mémoire CVM a également une structure de type table de correspondance et comporte une sortie fournissant, en mode lecture, une valeur de code témoin associée à une valeur d'adresse fournie à l'entrée de la mémoire. Cette valeur est appliquée à une entrée du comparateur CDT dont l'autre entrée est reliée au bus de données B2.

Le registre R1 reçoit une adresse d'insertion de code témoin fournie par le CPU, et optionnellement un identifiant de la fonction appelée ou identifiant de la fonction appelante, ou les deux.

Le circuit de contrôle CCT est configuré pour détecter une écriture du registre R1 et interpréter cette écriture comme une demande d'écriture d'un code témoin dans la pile STCK, à l'adresse présente dans le registre.

Ainsi, pour l'écriture d'un code témoin, le programmeur ou le compilateur insère simplement une instruction d'écriture du registre R1 dans le programme. Ce mode de réalisation permet d'adapter l'invention à tout type de microprocesseur sans devoir prévoir une instruction spécifique d'insertion de codes témoins.

En réponse à l'écriture du registre R1, le circuit de contrôle CCT :

- génère un code témoin, ici au moyen du générateur CGEN,
- prend le contrôle des bus d'adresse B1 et de données B2 et écrit le code témoin dans la pile STCK à l'adresse spécifiée,
- enregistre l'adresse du code témoin dans sa mémoire CAM en relation avec un index associé qui peut être l'identifiant de la fonction appelée ou appelante,
- enregistre la valeur du code témoin dans la mémoire CVM en relation avec l'adresse du code témoin,
- place la mémoire CAM en mode lecture et lui applique l'index associé à l'adresse du code témoin.

Ainsi, les adresses de tous les codes témoins rattachés à cet index (par exemple tous les codes témoins associés à l'identifiant de la fonction appelée ou appelante) sont appliqués aux comparateurs CA0-CAi et sont sous surveillance. Si l'adresse de l'un des codes témoins sous surveillance apparaît sur le bus d'adresse B1, la sortie

de l'un des comparateurs CA0-CAi passe à 1 et le signal ER passe à 1 à la sortie de la porte G1, ce qui déclenche le processus de protection du système décrit plus haut (remise à zéro du microprocesseur ou traitement sécurisé de l'erreur).

Dans une variante, l'adresse d'écriture du code témoin est appliquée sur le bus d'adresse par le CPU. Le registre R1 est utilisé pour communiquer au circuit de contrôle CCT l'identifiant du processus en cours ou toute information autre que l'adresse du code témoin. Le circuit de contrôle CCT lit cette adresse sur le bus d'adresse pour la mémoriser dans la mémoire CVM, et fournit seulement la valeur du code témoin sur le bus de données. Le CPU applique ensuite lui-même une commande d'écriture à la pile STCK.

En réponse à l'écriture d'une adresse dans registre R2, le circuit de contrôle CCT :

- applique sur le bus l'adresse présente dans le registre R2 et lit la valeur du code témoin dans la pile. Cette valeur se trouve alors sur le bus de données B2 et sur une entrée du comparateur CDT,
- place la mémoire CVM en mode lecture et lui applique l'adresse présente dans le registre R2 en tant qu'index de lecture. La valeur initialement enregistrée du code témoin en relation avec cette adresse se trouve alors sur une seconde entrée du comparateur CDT.

Si les deux valeurs sont différentes, la sortie du comparateur CDT passe à 1 et le signal ER passe à 1 à la sortie de la porte G1, ce qui déclenche le processus de protection du système.

En l'absence du signal d'erreur, le circuit de contrôle CCT peut ensuite assurer lui-même l'effacement du code témoin dans la pile en accédant à celle-ci en mode

écriture, puis supprime l'adresse du code témoin de sa mémoire interne. Alternativement, le circuit CCT peut laisser le soin au CPU de procéder à cet effacement, après avoir supprimé l'adresse du code témoin de sa mémoire interne afin qu'elle ne soit plus sous surveillance.

Dans le cas où le circuit CCT n'est pas configuré pour effectuer lui-même cet effacement, le programmeur ou le compilateur doit prévoir, après une instruction d'écriture du registre R2, une instruction d'effacement de la pile à l'attention du CPU.

Dans une variante évoquée plus haut, le circuit de contrôle CCT génère des valeurs de codes témoins déterministes et les régénère en réponse à une écriture du registre R2. Dans ce cas, la mémoire CVM n'est pas nécessaire.

Divers autres modes de réalisation de l'invention peuvent être prévus par l'homme de l'art. Notamment, dans certains modes de réalisation, le CPU est équipé d'une mémoire cache agencée entre la mémoire programme et le CPU et recevant des instructions à exécuter plusieurs cycles d'horloge avant leur exécution effective. Dans ce cas, le moniteur est de préférence placé entre le CPU et le cache et est configuré pour observer les transactions exécutées par le CPU.

REVENDICATIONS

1. Microprocesseur comprenant une unité centrale (CPU), au moins une pile d'exécution (STCK), un pointeur de pile (SP), un bus d'adresse (B1) et un bus de données (B2),

caractérisé en ce qu'il comprend un moniteur hardware (MT) configuré pour :

- générer des codes témoins (C1, C2),
- insérer des codes témoins dans la pile ou laisser l'unité centrale les insérer,
- mémoriser (CAM) des adresses de codes témoins insérés dans la pile, et
- générer un signal d'erreur (ER) en réponse à une tentative de modification d'un code témoin présent dans la pile.

2. Microprocesseur selon la revendication 1, dans lequel le moniteur (MT) est également configuré pour générer le signal d'erreur en cas de tentative de lecture d'un code témoin dans la pile.

3. Microprocesseur selon l'une des revendications 1 et 2, dans lequel le moniteur est configuré pour générer des codes témoins aléatoires ou pseudo-aléatoires.

4. Microprocesseur selon l'une des revendications 1 et 2, dans lequel le moniteur est configuré pour générer des codes témoins déterministes et reproductibles.

5. Microprocesseur selon l'une des revendications 1 à 4, dans lequel le moniteur est configuré pour :

- surveiller le bus d'adresse (B1), et
- générer le signal d'erreur si une adresse de code témoin mémorisée apparaît sur le bus d'adresse.

6. Microprocesseur selon l'une des revendications 1 à 5, dans lequel le moniteur comprend un premier registre (R1) accessible en écriture pour l'unité centrale (CPU), et est configuré pour générer un code témoin et appliquer la valeur du code témoin sur le bus de données (B2) en réponse à une écriture d'une donnée dans le registre par l'unité centrale.

7. Microprocesseur selon la revendication 6, dans lequel le moniteur est également configuré pour insérer le code témoin dans la pile, à une adresse présente dans le premier registre (R1).

8. Microprocesseur selon l'une des revendications 1 à 7, dans lequel le moniteur (MT) est configuré pour, en réponse à une demande de suppression de code témoin de l'unité centrale ou du programme exécuté par l'unité centrale :
- lire le code témoin dans la pile, à une adresse spécifiée par la demande de suppression,
- comparer le code témoin avec une valeur attendue du code témoin, et
- générer le signal d'erreur (ER) si la valeur lue est différente de la valeur attendue.

9. Microprocesseur selon la revendication 8, dans lequel le moniteur est configuré pour supprimer lui-même le code témoin dans la pile, après l'avoir vérifié.

10. Microprocesseur selon l'une des revendications 8 et 9, dans lequel le moniteur comprend un second registre (R2) accessible en écriture pour l'unité centrale (CPU), et est configuré pour interpréter une écriture du second registre comme une demande de suppression d'un code témoin.

11. Microprocesseur selon la revendication 10, dans lequel le moniteur est configuré pour interpréter une écriture du second registre comme une demande de suppression d'un code témoin à une adresse présente dans le second registre.

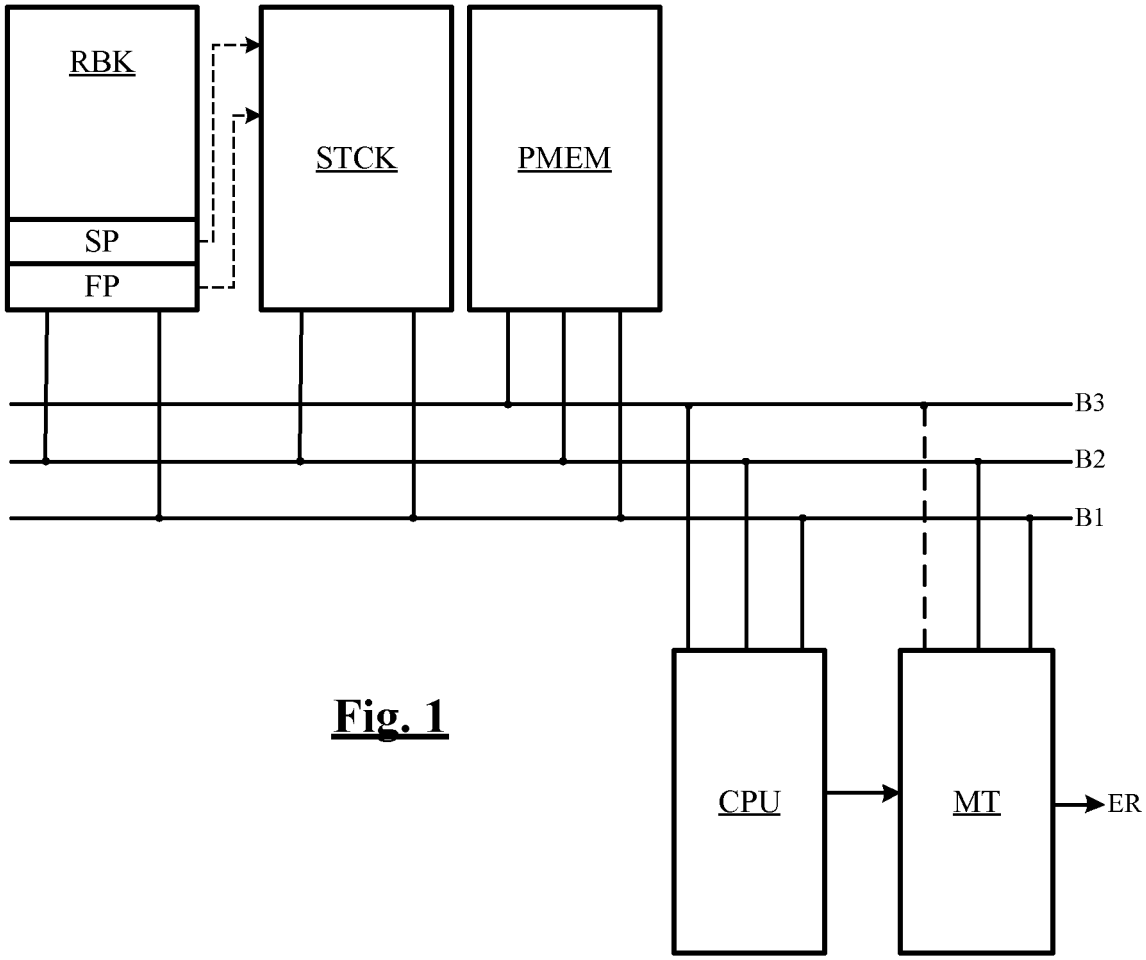

RBK
SP
FP
STCK
PMEM
B3
B2
B1
CPU
MT
ER

Fig. 1

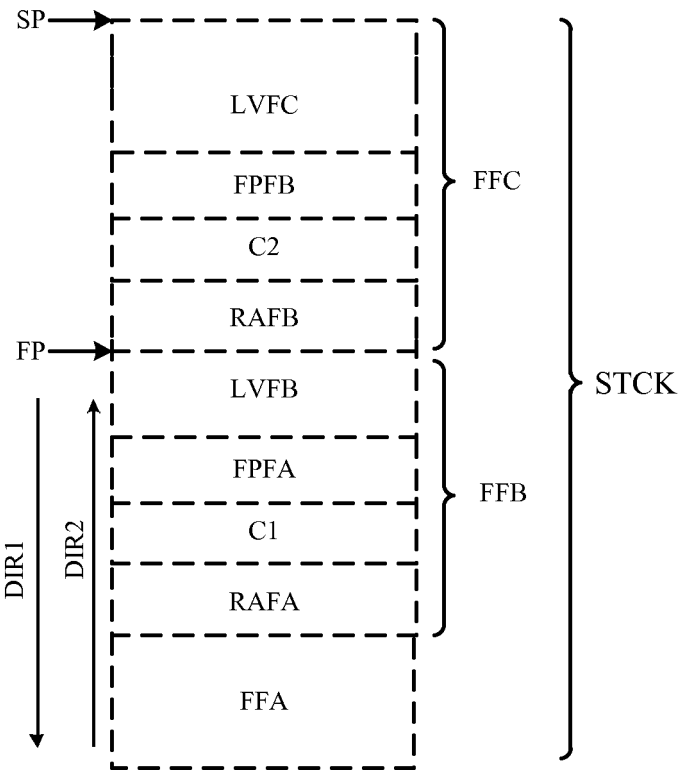

SP
LVFC
FPFB
C2
RAFB
FP
LVFB
FPFA
C1
RAFA
FFA
FFC
FFB
STCK
DIR1
DIR2

Fig. 2

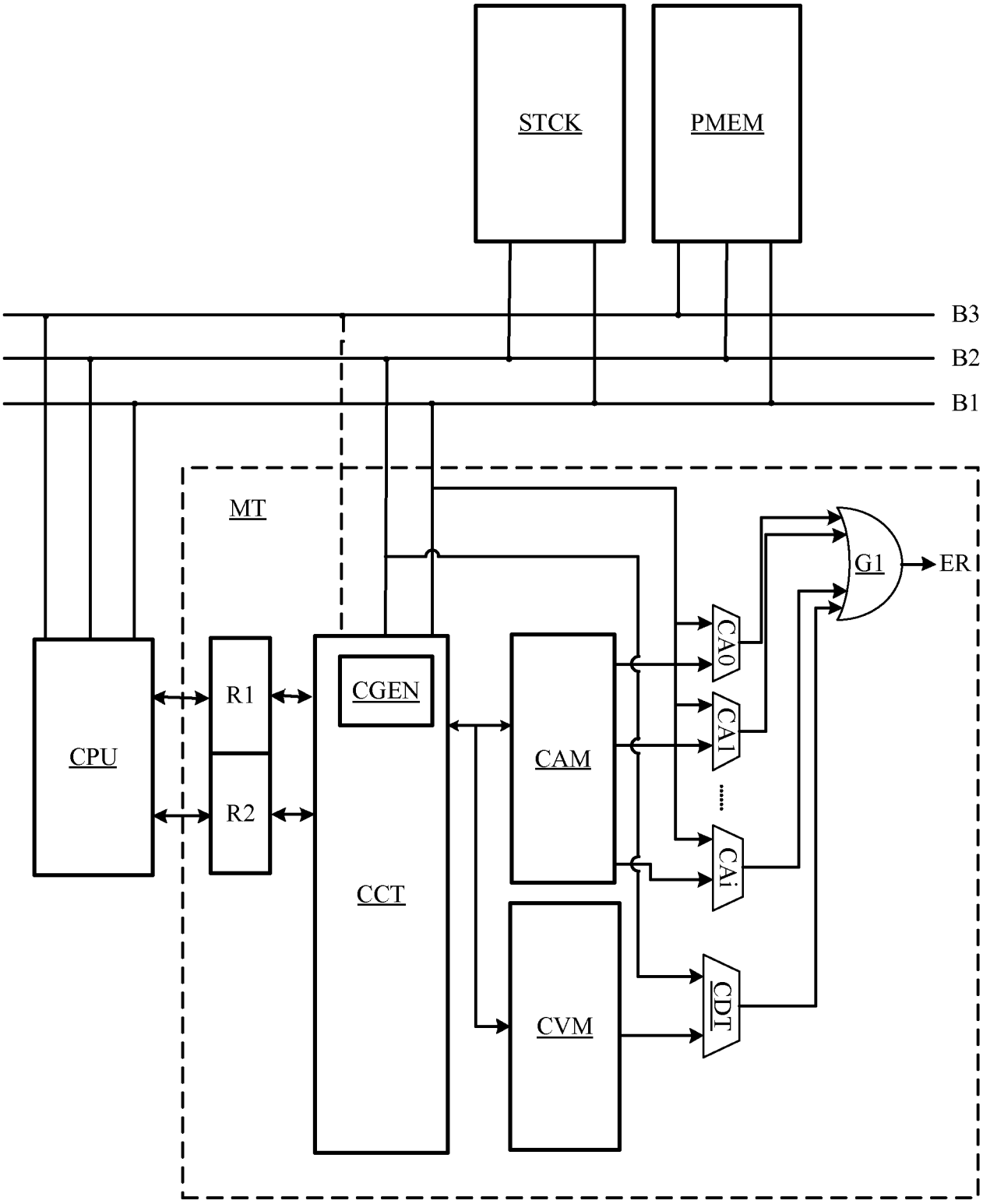

STCK
PMEM
B3
B2
B1
MT
G1
ER
CA0
CA1
CAi
CPU
R1
R2
CGEN
CCT
CAM
CVM
CDT

Fig. 3

RÉPUBLIQUE FRANÇAISE

INSTITUT NATIONAL DE LA PROPRIÉTÉ INDUSTRIELLE

RAPPORT DE RECHERCHE PRÉLIMINAIRE

établi sur la base des dernières revendications déposées avant le commencement de la recherche

N° d'enregistrement national

FA 755829
FR 1156210

DOCUMENTS CONSIDÉRÉS COMME PERTINENTS		Revendication(s) concernée(s)	Classement attribué à l'invention par l'INPI
Catégorie	Citation du document avec indication, en cas de besoin, des parties pertinentes		
X	US 2007/089088 A1 (BORDE SHRIKRISHNA V [US] ET AL) 19 avril 2007 (2007-04-19) * abrégé * * alinéa [0004] * * alinéas [0008] - [0012] * * alinéas [0022] - [0024] * * alinéas [0041] - [0051] * * alinéas [0057] - [0058]; figures 1-7 * -----	1-4	G06F21/06
X	US 2001/013094 A1 (ETOH HIROAKI [JP] ET AL) 9 août 2001 (2001-08-09) * abrégé * * alinéa [0009] * * alinéas [0029] - [0041] * * alinéas [0066] - [0083] * * figures 1-11 * -----	1-3	
A	US 2005/144471 A1 (SHUPAK RICHARD M [US] ET AL) 30 juin 2005 (2005-06-30) * abrégé * * alinéas [0008] - [0012] * * alinéas [0033] - [0039] * * alinéas [0045] - [0051] * * figures 1-5 * -----	1-11	DOMAINES TECHNIQUES RECHERCHÉS (IPC) G06F
A	US 2003/217277 A1 (NARAYANAN RAM GOPAL LAKSHMI [US]) 20 novembre 2003 (2003-11-20) * abrégé * * alinéas [0013] - [0015] * * alinéas [0025] - [0026] * * alinéas [0034] - [0040] * * figures 1-5 * * revendications 1-12 * -----	1-11	

Date d'achèvement de la recherche	Examinateur
23 décembre 2011	Bichler, Marc

CATÉGORIE DES DOCUMENTS CITÉS

X : particulièrement pertinent à lui seul
Y : particulièrement pertinent en combinaison avec un autre document de la même catégorie
A : arrière-plan technologique
O : divulgation non-écrite
P : document intercalaire

T : théorie ou principe à la base de l'invention
E : document de brevet bénéficiant d'une date antérieure à la date de dépôt et qui n'a été publié qu'à cette date de dépôt ou qu'à une date postérieure.
D : cité dans la demande
L : cité pour d'autres raisons

& : membre de la même famille, document correspondant

EPO FORM 1503 12.99 (P04C14)

ANNEXE AU RAPPORT DE RECHERCHE PRÉLIMINAIRE RELATIF A LA DEMANDE DE BREVET FRANÇAIS NO. FR 1156210 FA 755829

La présente annexe indique les membres de la famille de brevets relatifs aux documents brevets cités dans le rapport de recherche préliminaire visé ci-dessus.
Les dits membres sont contenus au fichier informatique de l'Office européen des brevets à la date du 23-12-2011
Les renseignements fournis sont donnés à titre indicatif et n'engagent pas la responsabilité de l'Office européen des brevets, ni de l'Administration française

Document brevet cité au rapport de recherche	Date de publication	Membre(s) de la famille de brevet(s)	Date de publication
US 2007089088 A1	19-04-2007	AUCUN	
US 2001013094 A1	09-08-2001	JP 3552627 B2	11-08-2004
		JP 2001216161 A	10-08-2001
		US 2001013094 A1	09-08-2001
US 2005144471 A1	30-06-2005	AUCUN	
US 2003217277 A1	20-11-2003	AUCUN	

EPO FORM P0465