

①②

DEMANDE DE BREVET D'INVENTION

A1

②② Date de dépôt : 27.01.00.

③⑦ Priorité :

④③ Date de mise à la disposition du public de la
demande : 03.08.01 Bulletin 01/31.

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

⑥⑦ Références à d'autres documents nationaux
apparentés :

⑦① Demandeur(s) : STMICROELECTRONICS SA
Société anonyme — FR.

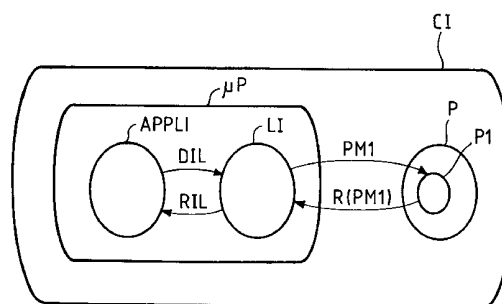
⑦② Inventeur(s) : CHABAUD MYRIAM et ALBARET
THIERRY.

⑦③ Titulaire(s) :

⑦④ Mandataire(s) : CABINET BALLOT.

⑤④ SYSTEME DE GESTION DE PERIPHERIQUES DANS UN CIRCUIT INTEGRE.

⑤⑦ Dans un circuit intégré à microprocesseur μP pour
carte à puce comprenant une couche logicielle d'application
APPLI, un système de gestion de périphériques comprend
une couche logicielle intermédiaire LI pour gérer les proces-
sus matériels sur des périphériques du circuit intégré, appe-
lés par un ou des processus de la couche logicielle
d'application.



SYSTEME DE GESTION DE PERIPHERIQUES DANS UN CIRCUIT INTEGRE

La présente invention concerne un système de gestion de périphériques dans un circuit intégré à microprocesseur. Elle concerne plus particulièrement les applications du type carte à puce.

5 Un circuit intégré pour une application de type carte à puce, appelé couramment microcircuit, ou encore micromodule électronique, comprend, en plus d'un microprocesseur, un certain nombre d'autres ressources matérielles, appelées généralement périphériques, tel
10 que des compteurs, des circuits d'interface série, des générateurs de nombre aléatoires, de signaux d'horloge... On a habituellement, associées à ces périphériques, des mémoires dont une mémoire programme ROM, des mémoires reprogrammables pour contenir des
15 données d'application (EEPROM) ou encore une mémoire de travail (RAM), et un contrôleur d'accès aux périphériques, qui gère notamment les bus d'adresse et de données.

Un tel circuit intégré est habituellement livré à
20 un client, qui n'est pas nécessairement l'utilisateur final du circuit, pour lequel on a implémenté dans ce circuit intégré selon sa destination finale, au moins une application. La tendance actuelle serait à l'implémentation de plusieurs applications, dont une
25 est activée par une application externe, à un moment donné.

Ces applications peuvent être de différents types. Elles peuvent par exemple consister en un système d'exploitation de carte à puce (*operating system*), une

librairie cryptographique, qui pourra permettre des calculs ou vérification de signature, du chiffrement, ou encore un logiciel de test embarqué, cette liste d'applications n'étant pas exhaustive.

- 5 Ces applications sont implémentées dans le circuit intégré sous forme de processus logiciels dans une couche logicielle d'application. Ces processus sont exécutés par le microprocesseur du circuit intégré.

10 Pour réaliser certains traitements d'un processus logiciel donné, la couche logicielle d'application doit appeler des processus sur des périphériques, appelées processus matériels dans la suite (par opposition aux processus logiciels).

15 Habituellement, elle le fait directement, c'est à dire qu'elle gère directement les périphériques du circuit intégré. Le fabricant du circuit intégré fournit à cet effet toutes les informations nécessaires, dont les adresses des différents périphériques du circuit intégré, les adresses en
20 mémoire programme ROM des tables des vecteurs d'interruption matérielle,.... Ces informations sont spécifiques du circuit intégré. Elles permettent à la couche logicielle d'application d'appeler directement un processus matériel sur un périphérique. Elles
25 permettent aussi à la couche logicielle de gérer ces périphériques en mode interruption, ce qui lui permet de continuer à travailler tant que le périphérique n'a pas terminé d'exécuter un processus matériel demandé (par exemple, la programmation d'une page mémoire
30 EEPROM). Dans ce cas, c'est la couche logicielle d'application qui gère directement les interruptions matérielles émises par les périphériques, ce qui

implique qu'elle doit lire et écrire des registres et accéder à la table des vecteurs d'interruption matérielle, en mémoire programme ROM, pour identifier une source d'interruption et engager les actions
5 nécessaires.

Or, on cherche à améliorer toujours et encore la sécurité des cartes à puce, pour empêcher toute utilisation de ces cartes à des fins frauduleuses.

Un objet de l'invention est de sécuriser le système
10 de gestion des périphériques dans un circuit intégré de carte à puce.

Une solution à ce problème technique a été trouvée dans l'invention, dans l'utilisation d'une couche logicielle intermédiaire agissant comme une barrière de
15 sécurité entre la couche logicielle d'application et les ressources matérielles du circuit intégré, c'est à dire le microprocesseur et les périphériques.

Selon l'invention, cette couche logicielle intermédiaire assure la gestion de tous les
20 périphériques pour le compte de la couche logicielle d'application. Ainsi, lorsqu'un processus de la couche logicielle d'application appelle un processus matériel sur un périphérique, cet appel est géré par la couche logicielle intermédiaire. Pour cela, on utilise une
25 instruction d'interruption logicielle, par laquelle on appelle une fonction de la couche logicielle intermédiaire.

Lorsque la couche logicielle intermédiaire a terminé le traitement lié à la fonction appelée, elle
30 fait un retour à la couche logicielle d'application, par une instruction de retour d'interruption.

Par ce mécanisme, la couche logicielle d'application n'a plus à connaître les spécificités matérielles du produit, comme l'adresse de la table des vecteurs d'interruptions matérielles, l'adresse des périphériques....On crée ainsi une barrière de sécurité
5 entre les ressources matérielles du circuit intégré et l'application.

Par ailleurs, le temps d'exécution d'un processus matériel particulier sur un périphérique peut-être
10 assez long (par exemple, la programmation d'un ou plusieurs mots en mémoire EEPROM).

De préférence, pour améliorer les performances de la carte, on prévoit que la couche logicielle intermédiaire libère tout de suite la couche logicielle d'application qui a appelé un processus matériel. Dans
15 ce cas, la couche logicielle intermédiaire elle-même gère les périphériques en mode interruption. Ceci permet à la couche logicielle d'application de continuer à travailler et à la couche logicielle intermédiaire de traiter d'autres appels de processus
20 matériels, le cas échéant.

Dans ce cas, lorsqu'un périphérique a terminé l'exécution d'un processus matériel, il émet une interruption matérielle, qui stoppe l'exécution en
25 cours dans le microprocesseur et le renvoie à la gestion des interruptions matérielles dans la couche logicielle intermédiaire.

Selon l'invention, on prévoit alors que le traitement de l'interruption matérielle dans la couche
30 logicielle intermédiaire comprend l'interruption de la couche logicielle d'application pour permettre à cette dernière de terminer le processus correspondant. Ainsi,

selon l'invention, l'on a un partage du traitement de l'interruption matérielle, la couche logicielle intermédiaire assurant sa fonction de barrière de sécurité en effectuant le traitement de l'interruption
5 correspondant à l'aspect matériel proprement dit, notamment la lecture et l'écriture de registres pour identifier la source de l'interruption matérielle, tandis que la couche logicielle d'application assure le reste du traitement de l'interruption, lié à
10 l'application.

Ainsi, l'invention concerne un système de gestion de périphériques dans un circuit intégré à microprocesseur pour une carte à puce, comprenant une couche logicielle d'application comprenant au moins un
15 processus nécessitant l'exécution de processus matériels sur au moins un périphérique du circuit intégré, caractérisé en ce qu'il comprend une couche logicielle intermédiaire pour gérer les processus matériels appelés par un processus de la couche
20 logicielle d'application.

D'autres caractéristiques et avantages de l'invention sont indiqués dans la description suivante faite à titre indicatif et non limitatif de l'invention, et en référence aux dessins annexés dans
25 lesquels :

la figure 1 représente schématiquement le système de gestion des périphériques selon l'invention ;

la figure 2 illustre un exemple de traitement d'un appel de fonction de la couche logicielle d'application
30 par la couche logicielle intermédiaire; et

la figure 3 illustre un exemple de mise en œuvre de l'invention utilisant des macro-instructions pour

assurer les appels entre la couche logicielle d'application et la couche logicielle intermédiaire.

Le système de gestion de périphériques selon l'invention s'applique de manière générale à un circuit
5 intégré CI comprenant un microprocesseur μP et des périphériques P. Il s'applique plus particulièrement à des circuits intégrés pour des applications de type carte à puce. Les périphériques de tels circuits intégrés comprennent habituellement au moins une
10 mémoire programme ROM contenant les processus logiciels de la couche d'application APPLI et le programme correspondant à la couche logicielle intermédiaire LI selon l'invention. Les processus ou programmes des couches d'application APPLI et intermédiaire LI sont
15 exécutés par le microprocesseur. C'est pourquoi on a choisi de les représenter sur la figure 1, à l'intérieur d'une boîte logique représentant le microprocesseur μP . A un moment donné, le microprocesseur exécute un processus ou programme donné
20 de la couche application APPLI. Il peut être en pratique interrompu dans cette exécution par une interruption matérielle ou logicielle.

Dans l'invention, lorsque le traitement de la couche d'application APPLI effectué par le
25 microprocesseur μP nécessite l'exécution d'un processus matériel dans un périphérique, il utilise l'interruption logicielle DIL pour appeler une fonction correspondante de la couche LI. Cette instruction de départ en interruption logicielle DIL est exécutée par
30 le microprocesseur μP . Le traitement de cette interruption logicielle comprend le lancement du processus matériel correspondant PM1 sur le

périphérique concerné P1. Le résultat de ce processus matériel R(PM1), qui peut tout simplement être un indicateur pour signifier que le processus est effectué, est retourné à la couche d'application avec
5 l'instruction de retour d'interruption logicielle, RIL.

Plus précisément, et comme représenté sur la figure 2, le microprocesseur exécute (1) un processus PL1 de la couche logicielle d'application. Dans ce processus PL1, il a besoin d'exécuter un processus matériel PM1.
10 Il émet donc une interruption logicielle DIL (2) pour appeler une fonction correspondante F(PM1) de la couche logicielle intermédiaire. Cette interruption logicielle est traitée par la couche logicielle intermédiaire LI, qui appelle (3) le processus matériel PM1 correspondant
15 sur le périphérique P1 concerné, en mode interruption. La couche logicielle intermédiaire LI rend alors la main (4) à la couche logicielle d'application qui peut continuer (5) d'autres traitements. Notamment, elle peut appeler d'autres fonctions différentes de la
20 couche logicielle intermédiaire, pour lancer d'autres processus matériels.

Lorsque le processus matériel PM1 est terminé, le périphérique P1 concerné émet (6) une interruption matérielle IT, ce qui stoppe (7) le traitement en cours
25 dans le microprocesseur μP , dans l'exemple un traitement de la couche logicielle d'application (ce pourrait être un traitement d'une autre fonction LI), pour le renvoyer vers la gestion de l'interruption (8), par la couche logicielle intermédiaire LI. Cette gestion
30 comprend la lecture et l'écriture de la table des vecteurs d'interruption matérielle, permettant d'identifier la source de l'interruption matérielle, P1

dans l'exemple, puis le renvoi (9) de cette interruption vers la couche logicielle d'application APPLI, pour permettre le traitement (10) de l'interruption par le processus logiciel appelant. Ce
5 renvoi se fait par une interruption logicielle DIL émise cette fois par la couche logicielle intermédiaire LI. Lorsque le processus concerné PL1 de la couche logicielle d'application a terminé son traitement de l'interruption, il libère (11) la couche logicielle
10 intermédiaire LI, qui a son tour, libère (12) le périphérique Pl.

Le microprocesseur peut alors reprendre (13) le traitement en cours, là où il avait été arrêté.

Les considérations liées au mode interruption ne
15 vont pas être rappelées en détail. Il s'agit d'un mode de communication avec des périphériques habituel, bien connu de l'homme du métier. On pourra simplement noter que dans le système de gestion selon l'invention qui utilise l'instruction assembleur de départ en
20 interruption logicielle DIL, on associe en pratique un domaine à chacun des processus PL1 de la couche logicielle d'application et un domaine à la couche logicielle intermédiaire LI, en sorte que l'on a un vecteur d'interruption logicielle associé à chacun de
25 ces domaines. Ainsi, lorsqu'une interruption logicielle DIL est émise, la table des vecteurs d'interruption permet d'identifier l'émetteur et de lancer le traitement d'interruption correspondant. Cette table se trouve dans la couche logicielle intermédiaire.

30 Par ailleurs, si un périphérique sur laquelle la couche logicielle intermédiaire LI doit lancer un processus matériel est occupé, l'étape (4) de la figure

2 dans laquelle la couche logicielle intermédiaire LI libère la couche logicielle d'application comprend le renvoi d'une information correspondante, lui signifiant que le périphérique est indisponible. On peut aussi
5 prévoir que pour chaque domaine, on définit les fonctions de la couche logicielle intermédiaire LI qui peuvent être appelées. Dans ce cas, si la fonction de la couche logicielle intermédiaire LI appelée par un processus donné ne fait pas partie de la définition du
10 domaine associé à ce processus, on prévoit que la couche logicielle intermédiaire LI renvoie avec l'instruction de retour d'interruption RIL une information indiquant que la fonction n'est pas accessible.

15 Si au contraire, le périphérique est disponible, il libère la couche logicielle d'application en lui renvoyant une information correspondante, lui indiquant que sa demande est en cours de traitement dans le périphérique concerné.

20 Dans l'invention, on utilise donc les possibilités offertes par le mode d'interruption logicielle du microprocesseur μp pour créer une barrière de sécurité entre la couche logicielle d'application et les périphériques. En particulier, la couche logicielle
25 d'application n'a plus besoin de connaître les adresses des périphériques, de la tables des vecteurs d'interruption matérielle et autres informations qui sont des informations particulièrement importantes sur le plan sécuritaire. Elle doit seulement mettre en
30 oeuvre le mode d'interruption logicielle, mode par lequel elle transmet ses appels de fonctions sur la

couche logicielle intermédiaire qui gère les périphériques à sa place.

En pratique, dans le cas des circuits intégrés pour des applications de type carte à puce, les compilateurs
5 de programme en code source (assembleur) n'intègrent généralement pas l'instruction assembleur de départ en interruption logicielle DIL et l'instruction assembleur de retour d'interruption RIL associée.

De préférence, pour éviter aux développeurs des
10 processus de la couche d'application et aux développeurs de la couche logicielle intermédiaire LI d'avoir à programmer en assembleur, on prévoit selon l'invention que la couche logicielle intermédiaire LI contient des macro-instructions pour chaque appel de
15 fonction.

La couche logicielle d'application doit donc seulement appeler une macro-instruction de la couche logicielle intermédiaire LI. Cette macro-instruction comprend le transfert des paramètres liés à la fonction
20 LI à exécuter et l'exécution de l'instruction assembleur DIL. Par ce mécanisme, c'est la couche logicielle intermédiaire LI qui émet l'interruption logicielle DIL. Le traitement de cette interruption DIL par la couche logicielle intermédiaire LI renvoie à son
25 tour vers une fonction LI. De cette façon, seule la partie de la couche logicielle intermédiaire LI qui contient les macro-instructions doit être développée en langage assembleur. La couche logicielle d'application et les fonctions de la couche logicielle intermédiaire
30 LI sont elles développées en langage évolué, typiquement en langage C. Un synoptique correspondant est représenté sur la figure 3.

L'invention qui vient d'être décrite permet de créer une barrière de sécurité entre la couche d'application et les périphériques du circuit intégré.

En outre, par le partage du traitement des
5 interruptions matérielles entre les couches logicielles intermédiaire et d'application, on optimise la communication entre ces deux couches, en permettant à l'application concernée d'effectuer elle-même le traitement de l'interruption matérielle lié à
10 l'application, tout en laissant la couche logicielle intermédiaire LI assurer son rôle de barrière, en lui laissant tous les aspects du traitement liés au matériel, notamment, l'identification de la source de l'interruption.

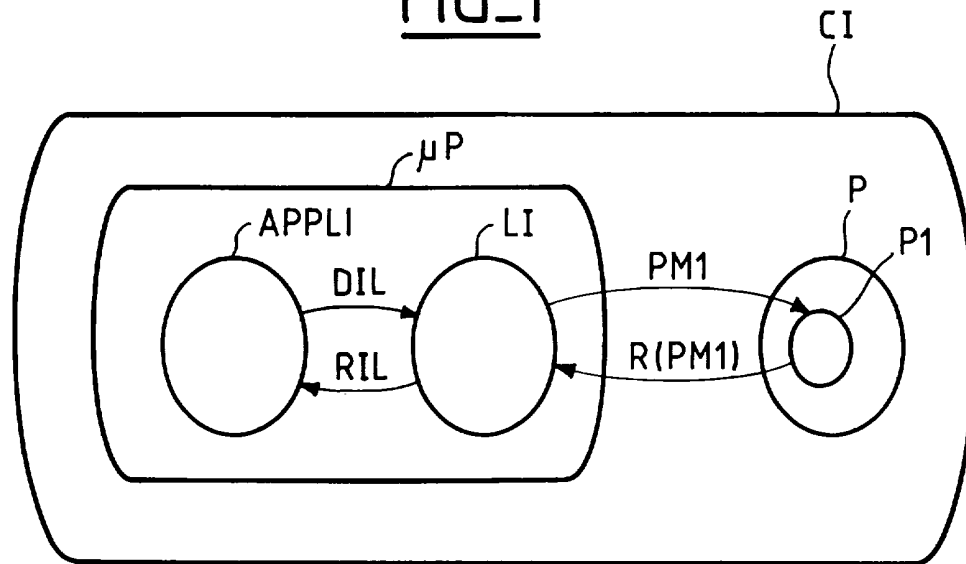
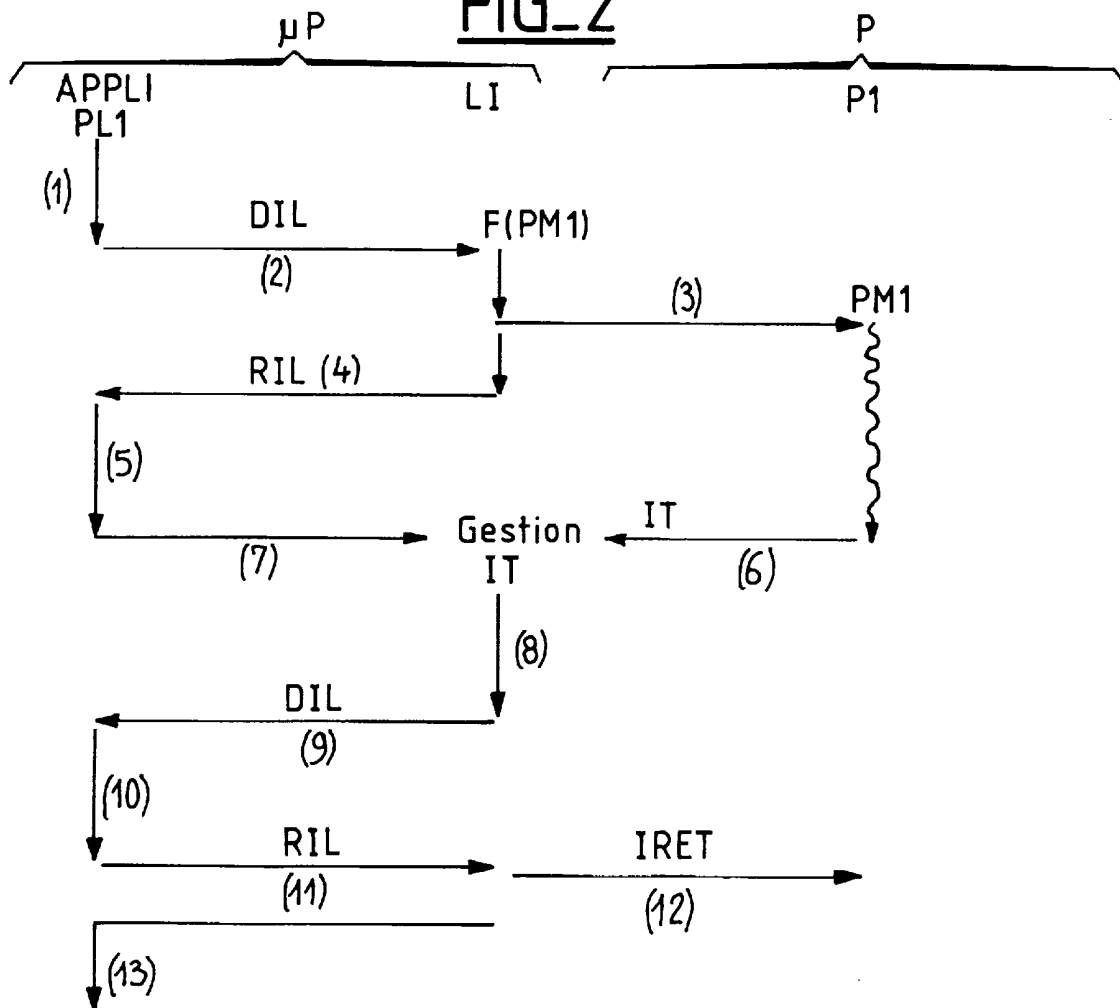
REVENDECATIONS

1. Système de gestion de périphériques dans un circuit
intégré à microprocesseur (μP) pour carte à puce,
comprenant une couche logicielle d'application
(APPLI) comprenant au moins un processus (P11)
5 nécessitant l'exécution de processus matériels sur au
moins un périphérique (P1) du circuit intégré,
caractérisé en ce qu'il comprend une couche
logicielle intermédiaire (LI) pour gérer les
processus matériels appelés par un processus de la
10 couche logicielle d'application.
2. Système de gestion de périphériques selon la
revendication 1, caractérisé en ce que la couche
logicielle d'application (APPLI) appelle une fonction
15 (F(PM1)) de la couche logicielle intermédiaire (LI)
correspondant au processus matériel (PM1) à exécuter.
3. Système de gestion de périphériques selon la
revendication 2, caractérisé en ce que cet appel de
20 fonction (F(PM1)) est effectué par l'émission par la
couche logicielle d'application (APPLI) d'une
interruption logicielle (DIL).
4. Système de gestion de périphériques selon la
25 revendication 2, caractérisé en ce que cet appel de
fonction (F(PM1)) est effectué par l'appel par la
couche logicielle d'application (APPLI) d'une macro-
instruction de la couche logicielle intermédiaire
(LI), ladite macro-instruction comprenant l'émission
30 par la couche logicielle intermédiaire (LI) d'une

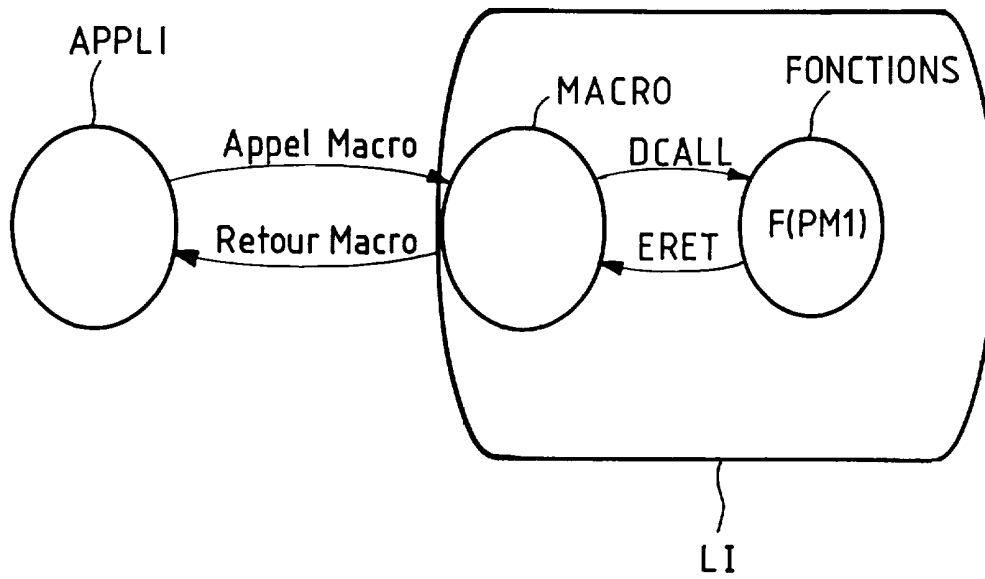
interruption logicielle (DIL) par laquelle ladite fonction (F(PM1)) est appelée.

- 5 5. Système de gestion de périphériques selon la revendication 3 ou 4, caractérisé en ce que le traitement de l'interruption logicielle (DIL) par la couche logicielle intermédiaire (LI) comprend la vérification de la disponibilité de la fonction appelée pour le processus appelant de la couche
10 logicielle d'application.
6. Système de gestion de périphériques selon l'une quelconque des revendications 3 à 5, caractérisé en ce que la couche intermédiaire (LI) gère les
15 périphériques en mode interruption.
7. Système de gestion de périphériques selon la revendication 6, caractérisé en ce que le traitement des interruptions matérielles (IT) des périphériques
20 est effectué par la couche logicielle intermédiaire (LI), le traitement d'une interruption matérielle (IT) comprenant l'interruption logicielle (DIL) de la couche logicielle d'application (APPLI) pour permettre un traitement correspondant à ladite
25 interruption matérielle dans cette couche, le périphérique étant libéré après la fin des traitements des dites interruptions matérielle et logicielle.
- 30 8. Carte à puce comprenant un circuit intégré muni d'un système de gestion de périphériques selon l'une quelconque des revendications précédentes.

1/2

FIG_1FIG_2

2/2

FIG_3

**RAPPORT DE RECHERCHE
PRÉLIMINAIRE**

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

2804525

N° d'enregistrement
national

FA 588478
FR 0001062

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 6 003 134 A (LO MINWEN ET AL) 14 décembre 1999 (1999-12-14)	1-4,6,8	G06F13/10 G06F13/38
Y	* abrégé; revendications 1-3,6,7,11,13,16; figures 3-5 *	4,7	G06K19/07
Y	TEO C Y ET AL: "MICROCOMPUTER-BASED MULTI-TASKING ON A SCADA SYSTEM USING INTERRUPTS IN MS-DOS" COMPUTERS IN INDUSTRY,NL,ELSEVIER SCIENCE PUBLISHERS. AMSTERDAM, vol. 22, no. 1, 1 juin 1993 (1993-06-01), pages 93-102, XP000381059 ISSN: 0166-3615 * abrégé * * page 95, colonne de gauche, ligne 6 - colonne 6, ligne 24 * * page 101, colonne de gauche, ligne 23 - colonne de droite, ligne 14 *	4,7	
			DOMAINES TECHNIQUES RECHERCHÉS (Int.CL.7)
			G06F
Date d'achèvement de la recherche		Examineur	
30 novembre 2000		Kingma, Y	
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	