



- (51) Classification internationale des brevets :
G06F 9/48 (2006.01) **G06F 9/50** (2006.01)
- (21) Numéro de la demande internationale :
PCT/FR2013/052633
- (22) Date de dépôt international :
5 novembre 2013 (05.11.2013)
- (25) Langue de dépôt : français
- (26) Langue de publication : français
- (30) Données relatives à la priorité :
1260587 8 novembre 2012 (08.11.2012) FR
- (71) Déposant : BULL SAS [FR/FR]; rue Jean Jaurès, F-78340 les Clayes sous Bois (FR).
- (72) Inventeurs : MARTINEZ, Patrice; 22, rue Alfred de Musset, F-38100 Grenoble (FR). ZENG EYINDANGA, Landry Stéphane; 88, rue Léon Jouhaux, F-38100 Grenoble (FR). CAYUELA, Jacques; Allée Verte, F-38700 La Tronche (FR).
- (74) Mandataire : SANTARELLI; BP 237, 14 avenue de la Grande-Armée, F-75822 Paris Cedex 17 (FR).

(81) États désignés (sauf indication contraire, pour tout titre de protection nationale disponible) : AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, JP, KE, KG, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) États désignés (sauf indication contraire, pour tout titre de protection régionale disponible) : ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, SZ, TZ, UG, ZM, ZW), eurasien (AM, AZ, BY, KG, KZ, RU, TJ, TM), européen (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

Déclarations en vertu de la règle 4.17 :

— relative à la qualité d'inventeur (règle 4.17.iv))

[Suite sur la page suivante]

(54) Title : METHOD, DEVICE AND COMPUTER PROGRAMME FOR THE PLACEMENT OF TASKS IN A MULTI-CORE SYSTEM

(54) Titre : PROCÉDÉ, DISPOSITIF ET PROGRAMME D'ORDINATEUR DE PLACEMENT DE TÂCHES DANS UN SYSTÈME MULTI-COEURS

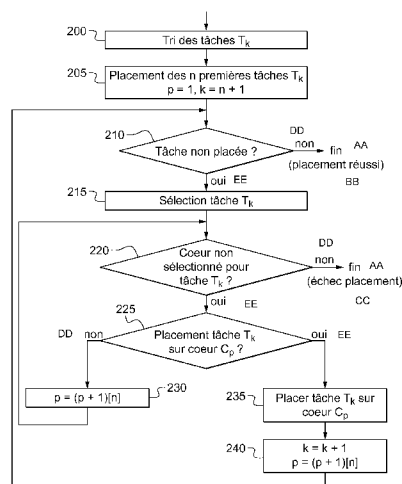


Fig.2

200 Sorting of tasks T_k
205 Placement of n first tasks T_k $p=1, k=n+1$
210 Task not placed?
215 Selection of task T_k
220 Core not selected for task T_k ?
225 Placement of task T_k on core C_p ?
AA end
BB placement successful
CC placement failed
DD no
EE yes

(57) Abstract : The invention concerns, in particular, the placement of tasks in a multi-core system, time constraints being associated with each task. After allocating (205) at least one separate task to each core, an unallocated task is selected (220). A core capable of executing the selected task according to time constraints associated with this task and time constraints associated with tasks previously allocated to this core is then identified (225). The selected task is then allocated (235) to the identified core.

(57) Abrégé : L'invention a notamment pour objet le placement de tâches dans un système multi-cœurs, des contraintes de temps étant associées à chaque tâche. Après avoir affecté (205) au moins une tâche distincte à chaque cœur, une tâche non affectée est sélectionnée (220). Un cœur susceptible d'exécuter la tâche sélectionnée selon des contraintes de temps associées à cette tâche et des contraintes de temps associées à des tâches préalablement affectées à ce cœur est alors identifié (225). La tâche sélectionnée est ensuite affectée (235) au cœur identifié.



Publiée :

— avec rapport de recherche internationale (Art. 21(3))

— avant l'expiration du délai prévu pour la modification des revendications, sera republiée si des modifications sont reçues (règle 48.2.h))

**Procédé, dispositif et programme d'ordinateur de placement
de tâches dans un système multi-cœurs**

5 La présente invention concerne la gestion de processus dans les systèmes d'information et plus particulièrement un procédé, un dispositif et un programme d'ordinateur de placement de tâches dans un système multi-cœurs, permettant notamment de certifier le comportement d'une application multitâche et temps réel en calculant le placement de ses tâches sur une machine multi-
10 cœurs en fonction de contraintes de temps réel de chacune des tâches.

 Un problème important rencontré lors du développement d'applications logicielles multitâches et temps réel est relatif à la garantie du fait que toutes les tâches de cette application puissent s'exécuter en respectant des contraintes de temps. En particulier, des interruptions affectant une tâche ne
15 doivent pas empêcher cette dernière de s'exécuter dans un délai imparti.

 Il est rappelé ici qu'un système est dit temps réel lorsqu'il est capable de réagir de façon fiable et répétée selon des délais d'exécution prédéterminés. A ces fins, une tâche exécutée par ce système doit s'exécuter en fonction d'autres tâches exécutées par celui-ci selon des contraintes de temps
20 prédéterminées. Le principe général du temps réel consiste à contrôler quelle partie du programme intervient à chaque instant, sans laisser de place à l'aléatoire.

 Lorsque le système de traitement de l'information utilisé est un système multi-cœurs, c'est-à-dire un système comprenant plusieurs unités de
25 traitement, le placement des tâches sur chaque cœur est particulièrement important pour le comportement temps réel de l'application considérée.

 Une solution couramment utilisée aujourd'hui pour optimiser le comportement d'une application multitâche et temps réel vis-à-vis de cette dernière caractéristique consiste à affecter, à chaque tâche, de façon
30 empirique, une priorité de placement d'autant plus élevée que le temps de réponse associé à la tâche considérée est petit afin d'influencer le placement (automatique) de la tâche. En effet, un séquenceur temps réel de tâches,

appelé *scheduler* en terminologie anglo-saxonne, ordonne et place les tâches d'une application en partant du principe qu'à un instant donné, les n cœurs d'un système de traitement doivent exécuter les n tâches les plus prioritaires.

5 Dans ce contexte, une garantie liée au comportement temps réel effectif de l'application peut être obtenue en effectuant une simulation de ce comportement à partir de scénarii d'exécution de l'application.

Cependant, une telle approche a ses limites car les scénarii utilisés doivent décrire de façon fidèle le comportement de l'application, ce qui peut être difficile à démontrer. Ainsi, à titre d'illustration, plusieurs tâches ayant chacune
10 une forte priorité peuvent empêcher une tâche de moindre priorité de respecter sa propre contrainte de temps d'exécution et, par conséquent, empêcher un fonctionnement normal de l'application.

Pour contenir un tel risque, une solution couramment utilisée consiste à sur-dimensionner le système, notamment au niveau de sa puissance
15 de calcul et/ou du nombre de cœurs mis en œuvre. Cependant, l'inconvénient majeur d'une telle approche est lié à son coût, sans que le surcoût ne garantisse le comportement temps réel de l'application.

Il existe donc un besoin pour améliorer le placement de tâches d'une application multitâche et temps réel dans un système multi-cœurs.

20 L'invention a ainsi pour objet un procédé pour ordinateur de placement d'une pluralité de tâches dans un système comprenant une pluralité de cœurs, avant l'exécution de chacune desdites tâches, des contraintes de temps étant associées à chaque tâche de ladite pluralité de tâches, ce procédé comprenant les étapes suivantes :

- 25 - affectation d'au moins une tâche distincte de ladite pluralité de tâches à chaque cœur de ladite pluralité de cœurs ;
- sélection d'une tâche non affectée de ladite pluralité de tâches ;
- identification d'un cœur de ladite pluralité de cœurs susceptible d'exécuter ladite tâche sélectionnée selon des contraintes de temps associées
30 à ladite tâche sélectionnée et des contraintes de temps associées à des tâches préalablement affectées audit cœur identifié ; et
- affectation de ladite tâche sélectionnée audit cœur identifié.

Le procédé selon l'invention permet ainsi d'affecter des tâches à des cœurs selon des contraintes de temps, notamment des tâches d'applications multitâches et temps réel.

5 Selon un mode de réalisation particulier, lesdites étapes de sélection d'une tâche, d'identification d'un cœur et d'affectation d'une tâche sélectionnée à un cœur identifié sont répétées pour affecter un ensemble de tâches de ladite pluralité de tâches à au moins un cœur de ladite pluralité de cœurs. Le procédé selon l'invention peut ainsi affecter automatiquement un ensemble de tâches à un ensemble de cœurs selon des contraintes de temps.

10 Toujours selon un mode de réalisation particulier, le procédé comprend en outre une étape d'assignation d'une priorité à chaque tâche affectée à un cœur, ladite priorité étant déterminée en fonction de l'ordre d'assignation des tâches audit cœur. Le procédé selon l'invention peut ainsi contrôler le respect de contraintes de temps.

15 Lesdites contraintes de temps comprennent, par exemple, au moins un pire temps d'exécution de la tâche associée, un temps de réponse imparti à la tâche associée et une période d'exécution de la tâche associée.

20 Selon un mode de réalisation particulier, le procédé comprend en outre une étape préalable d'ordonnancement des tâches de ladite pluralité de tâches selon au moins une contrainte de temps. Le procédé selon l'invention peut ainsi contrôler le respect de contraintes de temps.

Toujours selon un mode de réalisation particulier, ledit ordonnancement est effectué selon une période d'exécution de la tâche associée.

25 Toujours selon un mode de réalisation particulier, ladite étape d'identification d'un cœur comprend les étapes suivantes :

- sélection d'un cœur de ladite pluralité de cœurs ;
 - évaluation de la capacité dudit cœur sélectionné à exécuter ladite tâche sélectionnée selon des contraintes de temps associées à ladite tâche
- 30 sélectionnée et les contraintes de temps associées aux tâches préalablement affectées audit cœur sélectionné ; et

- si ledit cœur sélectionné n'est pas capable d'exécuter ladite tâche sélectionnée selon des contraintes de temps associées à ladite tâche sélectionnée et des contraintes de temps associées aux tâches préalablement affectées audit cœur sélectionné, répétition des étapes de sélection d'un cœur
5 et d'évaluation.

Le procédé selon l'invention permet ainsi de sélectionner des cœurs auxquels doivent être affectées des tâches, selon des contraintes de temps.

Ladite pluralité de cœurs forme par exemple une liste ordonnée de cœurs, un cœur étant sélectionné de façon incrémentale dans ladite liste
10 ordonnée de cœurs.

L'invention a également pour objet un programme d'ordinateur comprenant des instructions adaptées à la mise en œuvre de chacune des étapes du procédé décrit précédemment lorsque ledit programme est exécuté sur un ordinateur ainsi qu'un dispositif comprenant des moyens adaptés à la
15 mise en œuvre de chacune des étapes du procédé décrit précédemment. Les avantages procurés par ce programme d'ordinateur et ce dispositif sont similaires à ceux évoqués précédemment.

D'autres avantages, buts et caractéristiques de la présente invention ressortent de la description détaillée qui suit, faite à titre d'exemple non limitatif,
20 au regard des dessins annexés dans lesquels :

- la figure 1 illustre une représentation du temps d'exécution d'une tâche, de son pire temps d'exécution, du temps de réponse imparti à cette tâche et de sa période ;

- la figure 2 illustre un exemple d'algorithme de placement de
25 tâches d'une application multitâche et temps réel exécutée dans un système multi-cœurs, conformément à un mode de réalisation de l'invention ;

- la figure 3 illustre un exemple de placement de trois tâches sur un cœur, conformément à l'algorithme décrit en référence à la figure 2 ; et

- la figure 4 illustre un exemple d'architecture matérielle adaptée à
30 mettre en œuvre certaines étapes du procédé selon un mode de réalisation de l'invention.

De façon générale, l'invention a pour objet le placement de tâches dans un système multi-cœurs, permettant notamment de certifier le comportement d'une application multitâche et temps réel en calculant le placement de ses tâches sur une machine multi-cœurs en fonction de contraintes de temps de chacune des tâches.

Un tel placement est ici effectué avant l'exécution des tâches, de façon prédéterminée. En d'autres termes, le placement des tâches sur des cœurs est statique.

De telles contraintes de temps peuvent notamment être exprimées par les paramètres suivants, pour chaque tâche de l'application visée :

- le pire temps d'exécution de la tâche, aussi appelé *WCET* (sigle de *Worst Case Execution Time* en terminologie anglo-saxonne) ;
- le temps de réponse imparti à la tâche, aussi appelé *deadline* et noté *DL*, qui doit être respecté chaque fois que la tâche est exécutée ; et
- la période d'exécution de la tâche, ou intervalle de temps minimum entre deux exécutions consécutives de la tâche, aussi nommée période pour une tâche réellement périodique ou pseudo-période si seul l'intervalle de temps minimum entre deux exécutions consécutives de la tâche est considéré.

La période est une valeur héritée du domaine fonctionnel, elle correspond à un besoin applicatif, par exemple une fréquence d'échantillonnage sur un capteur. La pseudo-période exprime, elle aussi, un besoin fonctionnel. Cependant, elle correspond à un intervalle de temps minimum entre deux exécutions consécutives de la tâche. Une pseudo-période est ici assimilée à une période, ces deux notions étant gérées de façon identique dans un mode de réalisation particulier de l'invention. Les tâches non périodiques ont une période fixée ici à zéro.

Le pire temps d'exécution d'une tâche (*WCET*) correspond au temps d'exécution, au pire, de la tâche, quel que soit le traitement qu'elle doit effectuer et en considérant qu'elle n'est pas interrompue. Ce temps dépend notamment de la puissance de calcul du cœur exécutant cette tâche ainsi que de la nature et la quantité de données traitées par la tâche.

Le pire temps d'exécution d'une tâche peut-être obtenue, de façon empirique, en exécutant la tâche avec le jeu de données produisant la plus longue durée de traitement et en mesurant le temps de traitement. Cette mesure peut être effectuée par mesure directe ou à l'aide d'un simulateur ou d'un émulateur. La valeur de ce paramètre est, de préférence, fournie par le concepteur ou le développeur de l'application ou de la tâche. Elle peut également être obtenue, pour un système de traitement de données particulier, par extrapolation, par exemple par extrapolation linéaire, à partir d'une valeur de référence connue liée à un autre système de traitement de données particulier.

Le temps de réponse imparti à une tâche (*deadline*), pour effectuer un traitement, correspond également à un besoin applicatif. Il s'agit typiquement d'une contrainte forte qui, si elle n'est pas respectée, peut compromettre le bon fonctionnement de l'application temps réel visée. Il est typiquement défini par l'instant auquel une autre tâche a besoin d'un résultat produit par la tâche considérée.

La figure 1 illustre un exemple de représentation du temps d'exécution d'une tâche T_1 , de son pire temps d'exécution, noté $WCET(T_1)$, du temps de réponse imparti à cette tâche, noté $DL(T_1)$, et de sa période, notée $\delta(T_1)$.

La tâche T_1 , mise en œuvre par un cœur donné, est ici exécutée de façon périodique au temps t_0 , puis au temps t_1 suivant le temps t_0 , après une durée $\delta(T_1)$. $\delta(T_1)$ définit ainsi la période d'exécution de la tâche T_1 ($t_1 = t_0 + \delta(T_1)$). La $n^{ième}$ exécution de la tâche T_1 se produit au temps t_{n-1} ($t_{n-1} = t_0 + (n - 1) \cdot \delta(T_1)$).

Comme représenté, si le pire temps d'exécution de la tâche T_1 ($WCET(T_1)$), ainsi que le temps de réponse imparti de cette tâche ($DL(T_1)$) sont constants, le temps d'exécution réel de la tâche, noté $EXEC(T_1)$, ne l'est pas car il dépend de plusieurs facteurs, notamment, par exemple, des données à traiter. Ainsi, en d'autres termes, si les variables t'_i , t''_i et t'''_i sont définies de la façon suivante :

$$t'_i = t_i + EXEC(T_i)$$

$$t''_i = t_i + WCET(T_i)$$

$$t'''_i = t_i + DL(T_i)$$

$$\text{alors } (t''_i - t_i) = (t''_j - t_j) \text{ et } (t'''_i - t_i) = (t'''_j - t_j)$$

$$\text{mais } (t'_i - t_i) = (t'_j - t_j) \text{ ou } (t'_i - t_i) \neq (t'_j - t_j)$$

5 où les indices i et j représentent des indices sur des cycles d'exécution.

Selon un mode de réalisation particulier de l'invention, d'autres paramètres supplémentaires, non temporels et optionnels, peuvent être utilisés pour le placement de tâches dans un système multi-cœurs. Ainsi, par exemple
10 il est possible de prendre en compte l'ordre d'exécution de tâches et l'appel d'une tâche par une autre.

Il est observé ici que l'ordre d'exécution de tâches permet notamment de simplifier le calcul de placement si, du point de vue applicatif, des tâches de même période s'enchaînent. Dans ce cas, l'exécution de la $n^{\text{ième}}$
15 tâche à exécuter est lancée suite à une durée égale à la somme des temps de réponse impartis à chacune des $(n - 1)$ tâches précédentes. La somme des temps de réponse impartis aux tâches qui s'enchaînent doit être inférieure à la période commune de ces tâches.

Un appel de tâches consiste, pour une tâche donnée, à appeler une
20 autre tâche, de façon explicite, et à en attendre un résultat. Un tel appel peut être bloquant ou non. Dans le cas d'un appel non bloquant, seule une resynchronisation explicite de tâches est effectuée. Qu'ils soient bloquants ou non, il convient de vérifier que les conditions suivantes sont vérifiées :

$$WCET(T_{\text{appelante}}) + DL(T_{\text{appelée}}) < DL(T_{\text{appelante}})$$

25 où, à nouveau, $WCET(T_{\text{appelante}})$ représente le pire temps d'exécution de la tâche appelante et $DL(T_{\text{appelée}})$ et $DL(T_{\text{appelante}})$ représentent les temps de réponse impartis aux tâches appelées et appelantes, respectivement.

Il est observé que, dans ce cas, la tâche appelée n'est pas traitée comme périodique car elle est explicitement déclenchée par la tâche appelante.

30 Conformément à un mode de réalisation particulier de l'invention, des tâches dont les paramètres temporels sont compatibles entre eux sont placés sur un même cœur. En d'autres termes, toutes les tâches placées sur un même

cœur peuvent s'exécuter entièrement dans le respect de leur temps de réponse, quelles que soient les tâches qui viennent les interrompre.

Ainsi, de façon générale, l'invention a notamment pour objet, selon un mode de réalisation particulier, de calculer formellement, en fonction de
5 leurs caractéristiques temps réel, quelles tâches d'une application peuvent cohabiter au sein d'un même cœur d'un système multi-cœurs donné.

A ces fins, il est admis que les cœurs dédiés à l'exécution de tâches de l'application considérée sont exempts de perturbations. Ainsi, tous les processus non applicatifs tels que des processus du système d'exploitation
10 utilisé ainsi que des processus de traitement des interruptions sont avantageusement migrés sur d'autres cœurs dédiés, notamment, à l'exécution du système d'exploitation du système multi-cœurs.

Par ailleurs, il convient de prendre en compte le temps nécessaire au système d'exploitation utilisé pour commuter de l'exécution d'une tâche à celle
15 d'une autre. À ces fins, un temps de commutation prenant en compte un surcoût de temps ajouté par la plate-forme matérielle utilisée est ajouté au temps d'exécution de chaque tâche. Un tel temps de commutation peut être mesuré ou estimé. Dans ce dernier cas, il convient d'effectuer une majoration raisonnable.

20 Pour permettre le placement, les tâches sont ici triées en fonction de leurs caractéristiques, notamment en fonction de leur période ou pseudo-période d'exécution, de leur pire temps d'exécution (WCET) et/ou du temps de réponse qui leur est imparti (*deadline*). Leur tri peut également être basé, de façon optionnelle, sur l'ordre d'exécution des tâches et des appels entre tâches.

25 La figure 2 illustre un exemple d'algorithme de placement de tâches T_k d'une application multitâche et temps réel devant être exécutée dans un système multi-cœurs, comprenant ici n cœurs, conformément à un mode de réalisation de l'invention. Selon cet exemple, il convient de placer m tâches sur n cœurs.

30 Une première étape (étape 200) a pour objet de trier les m tâches T_k (k variant de 1 à m). Ce tri est ici effectué par période décroissante (selon la période associée à chaque tâche), puis pour des tâches ayant des périodes

identiques, par pire temps de réponse décroissant et, pour des tâches ayant des périodes et des pires temps de réponse identiques, par temps de réponse imparti décroissant.

Ainsi, par exemple, une tâche T_i est placée avant une tâche T_j si $\delta(T_i) > \delta(T_j)$ ou, si $\delta(T_i) = \delta(T_j)$, si $DL(T_i) > DL(T_j)$.

Au cours d'une étape suivante (étape 205), les n premières tâches T_k ($k = 1$ à n) sont placées sur les n cœurs du système multi-cœurs mettant en œuvre l'application considérée. Chacune de ces n premières tâches est placée sur un cœur distinct. Ainsi, en considérant que les cœurs sont notés C_p ($p = 1$ à n), la tâche T_k est placée sur le cœur C_p , k et p variant simultanément de 1 à n .

La valeur d'un index p , représentant un index sur les cœurs du système (variant de 1 à n), est alors mise à un et celle de l'index k à la valeur $n + 1$.

Un test est ensuite effectué (étape 210) pour déterminer s'il existe au moins une tâche de la liste des tâches ordonnées qui n'a pas été placée sur un cœur, c'est-à-dire s'il existe une tâche T_k . Dans la négative, l'algorithme prend fin, chacune des tâches de l'application considérée ayant été placée avec succès sur un cœur.

Au contraire, s'il existe au moins une tâche T_k de la liste des tâches ordonnées qui n'a pas été placée sur un cœur, la première tâche non placée de la liste des tâches ordonnées est sélectionnée (étape 215).

Un autre test est alors effectué pour déterminer s'il existe au moins un cœur qui n'a pas encore été sélectionné pour tenter d'y placer la tâche T_k sélectionnée (étape 220), c'est-à-dire pour déterminer si le cœur C_p a déjà été sélectionné pour tenter d'y placer la tâche T_k . Dans la négative, s'il a été tenté de placer la tâche T_k sur chacun des cœurs du système multi-cœurs, l'algorithme prend fin. Le placement des tâches est alors un échec.

Dans ce cas, une solution consiste par exemple à ajouter au moins un cœur au système multi-cœurs et à ré-exécuter l'algorithme décrit ici.

Au contraire, si le cœur C_p n'a pas encore été sélectionné pour tenter d'y placer la tâche T_k sélectionnée, un test est effectué pour déterminer si la tâche T_k sélectionnée peut être placée sur le cœur C_p sélectionné (étape 225).

Si la tâche T_k sélectionnée ne peut être placée sur le cœur C_p sélectionné, la valeur de l'index p est incrémentée de un, modulo n , c'est-à-dire modulo le nombre de cœurs du système multi-cœurs pouvant être utilisés pour placer des tâches de l'application considérée (étape 230). L'algorithme reboucle
 5 alors à l'étape 220 pour déterminer si la tâche T_k peut être placée sur le cœur suivant (cœur C_p).

Si, au contraire, la tâche T_k sélectionnée peut être placée sur le cœur C_p sélectionné, elle l'est (étape 235). La valeur de l'index k est alors incrémentée de un et celle de l'index p est également incrémentée de un,
 10 modulo n (étape 240). L'algorithme reboucle ensuite à l'étape 210 pour déterminer s'il reste au moins une tâche T_k à placer sur un cœur.

Pour déterminer si une tâche périodique ou pseudo-périodique T_k peut être placée sur un cœur C_p , il convient, de préférence, de prendre en compte la fréquence de cette tâche à placer par rapport à celles des tâches
 15 déjà placées sur le même cœur, ainsi que leurs pires temps d'exécution et leurs temps de réponse impartis. À ces fins, des règles peuvent être utilisées.

Ainsi, par exemple, il peut être décidé qu'une tâche T_r ne peut être placée sur un cœur C_p sur lequel ont été préalablement placées un ensemble de tâches T_s (s variant de 1 à $(r - 1)$) que si, et seulement si, pour tout k
 20 appartenant à l'intervalle $[1, r]$,

$$\sum_{i=k}^r WCETREEL(T_i) < DL(T_k)$$

avec

$$WCETREEL(T_i) = CEIL\left(\frac{DL(T_k)}{\delta(T_i)} \cdot WCET(T_i)\right)$$

où la fonction $CEIL(x)$ est une fonction qui retourne l'entier immédiatement supérieur à la valeur de la variable x .

De telles règles permettent de prendre en compte l'exécution d'une
 25 tâche de plus faible période et de calculer son impact réel sur des tâches déjà placées sur le même cœur.

Naturellement il est possible d'utiliser d'autres règles.

Lorsqu'une tâche est placée sur un cœur, une priorité d'exécution est ici associée à la tâche. Cette priorité correspond, par exemple, à son rang d'assignation sur le cœur. Ainsi, la première tâche placée sur un cœur a une priorité égale à 1, la seconde tâche placée sur ce cœur a une priorité égale à 2 et ainsi de suite.

Selon un mode de réalisation particulier, lorsqu'une tâche est placée sur un cœur, elle ne peut être déplacée ultérieurement pour y placer d'autres tâches.

Lorsqu'un ensemble de tâches a été placé avec succès sur un ensemble de cœurs d'un système multi-cœurs, une liste de tâches est obtenue pour chaque cœur du système. Les tâches des listes obtenues sont classées, par liste, de la tâche ayant la plus grande période à la tâche ayant la plus petite période, c'est-à-dire de la tâche ayant la plus faible priorité à la tâche ayant la plus haute priorité. En d'autres termes, la priorité d'une tâche est ici déterminée par sa position dans la liste de tâches affectées au cœur considéré.

Lorsque toutes les tâches d'une application ont été placées sur un ensemble de cœurs d'un système multi-cœurs, le comportement temps réel de l'application correspondante est vérifié car chaque tâche a le temps de s'exécuter avant d'atteindre la fin de son temps de réponse imparti malgré des interruptions de tâches plus prioritaires situées sur le même cœur.

La figure 3 illustre un exemple de placement de trois tâches sur un cœur, conformément à l'algorithme décrit en référence à la figure 2. Un rectangle blanc représente ici un temps de réponse imparti à une tâche tandis qu'un rectangle noir représente le temps de traitement effectivement alloué par le nœud à la tâche.

L'algorithme décrit en référence à la figure 2 peut être utilisé à différentes fins. Comme décrit précédemment, il peut être utilisé pour une certification formelle. L'algorithme détermine alors un placement qui garantit le comportement d'une application multitâche sur une plate-forme informatique donnée.

Il peut également être utilisé pour déterminer le dimensionnement minimum d'une plate-forme devant être mise en œuvre. A ces fins, l'algorithme

est lancé de façon successive en réduisant ou en augmentant, à chaque exécution, le nombre de cœurs pouvant être utilisés, jusqu'à obtenir le nombre de cœurs optimal.

L'algorithme permet également de valider un placement effectué de façon manuelle. À ces fins, la liste des tâches et de leur affectation aux différents cœurs est fournie à l'algorithme qui n'effectue pas, dans ce cas, l'étape de tri (étape 200). Si la validation réussit sans modifier l'ordre des tâches, le placement manuel était, au sens de l'algorithme, optimal. Si la validation modifie l'ordre des tâches, une solution plus adaptée est trouvée. Au contraire, si la validation échoue, le placement manuel était incorrect.

L'algorithme peut aussi être utilisé pour la certification d'une application répartie. Cet algorithme est utilisable pour calculer le placement des tâches sur un cluster de machines reliées entre elles par un réseau haut débit et dans lequel les horloges sont synchronisées. Cela permet notamment de mettre en œuvre des applications temps réel réparties de grande taille. A ces fins, il convient de dédier à l'application un certain nombre de cœurs, isolés de toute perturbation, par machine physique. Il convient également de prendre en compte, dans le temps de commutation, les aspects communication réseau entre machines.

La figure 4 illustre un exemple d'architecture matérielle d'un dispositif 400 adapté à mettre en œuvre certaines étapes de l'invention, en particulier les étapes décrites en référence à la figure 2. Le dispositif 400 est, par exemple, un ordinateur ou une station de travail.

Il comporte ici un bus de communication 405 auquel sont reliés :

- une ou plusieurs unités centrales de traitement ou microprocesseurs 410 (CPU, sigle de *Central Processing Unit* en terminologie anglo-saxonne) ;

- une mémoire morte 415 (ROM, acronyme de *Read Only Memory* en terminologie anglo-saxonne) pouvant comporter des programmes (prog, prog1 et prog2) nécessaires à la mise en œuvre de l'invention ;

- une mémoire vive ou mémoire cache 420 (RAM, acronyme de *Random Access Memory* en terminologie anglo-saxonne) comportant des

registres adaptés à enregistrer des variables et paramètres créés et modifiés au cours de l'exécution des programmes précités ; et

- une interface de communication 450 adaptée à transmettre et à recevoir des données.

5 Le dispositif 400 dispose également, de préférence, d'un disque dur 435 pouvant comporter les programmes précités ainsi que des informations traitées ou à traiter selon l'invention et d'un lecteur de cartes mémoires 440 adapté à recevoir une carte mémoire 445 et à y lire ou à y écrire des données traitées ou à traiter selon l'invention.

10 Le bus de communication permet la communication et l'interopérabilité entre les différents éléments inclus dans le dispositif 400 ou reliés à lui. La représentation du bus n'est pas limitative et, notamment, l'unité centrale est susceptible de communiquer des instructions à tout élément du dispositif 400 directement ou par l'intermédiaire d'un autre élément du dispositif
15 400.

Le code exécutable de chaque programme permettant au dispositif programmable de mettre en œuvre les processus selon l'invention, peut être stocké, par exemple, dans le disque dur 435 ou en mémoire morte 415.

20 Selon une variante, la carte mémoire 445 peut contenir des informations, notamment des informations à traiter selon l'invention, ainsi que le code exécutable des programmes précités qui, une fois lu par le dispositif 400, est stocké dans le disque dur 435.

25 Selon une autre variante, le code exécutable des programmes et les informations à traiter selon l'invention pourront être reçus, au moins partiellement, par l'intermédiaire de l'interface 450, pour être stocké de façon identique à celle décrite précédemment.

De manière plus générale, le ou les programmes ainsi que les informations à traiter selon l'invention pourront être chargés dans un des moyens de stockage du dispositif 400 avant d'être exécutés.

30 L'unité centrale 410 va commander et diriger l'exécution des instructions ou portions de code logiciel du ou des programmes selon l'invention, instructions qui sont stockées dans le disque dur 435 ou dans la

mémoire morte 415 ou bien dans les autres éléments de stockage précités. Lors de la mise sous tension, le ou les programmes qui sont stockés dans une mémoire non volatile, par exemple le disque dur 435 ou la mémoire morte 415, sont transférés dans la mémoire vive 420 qui contient alors le code exécutable
5 du ou des programmes selon l'invention, ainsi que des registres pour mémoriser les variables et paramètres nécessaires à la mise en œuvre de l'invention.

Naturellement, pour satisfaire des besoins spécifiques, une personne compétente dans le domaine de l'invention pourra appliquer des modifications
10 dans la description précédente.

REVENDEICATIONS

1. Procédé pour ordinateur de placement d'une pluralité de tâches
5 dans un système comprenant une pluralité de cœurs, avant l'exécution de
chacune desdites tâches, des contraintes de temps étant associées à chaque
tâche de ladite pluralité de tâches, lesdites contraintes de temps comprenant au
moins un pire temps d'exécution de la tâche associée, un temps de réponse
imparti à la tâche associée et une période d'exécution de la tâche associée, ce
10 procédé étant caractérisé en ce qu'il comprend les étapes suivantes :

- affectation (205) d'au moins une tâche distincte de ladite pluralité
de tâches à chaque cœur de ladite pluralité de cœurs ;
- sélection (220) d'une tâche non affectée de ladite pluralité de
tâches ;
- 15 - identification (225) d'un cœur de ladite pluralité de cœurs
susceptible d'exécuter ladite tâche sélectionnée selon des contraintes de temps
associées à ladite tâche sélectionnée et des contraintes de temps associées à
des tâches préalablement affectées audit cœur identifié ; et
- affectation (235) de ladite tâche sélectionnée audit cœur identifié,
- 20 le procédé comprenant une étape préalable d'ordonnancement (200) des
tâches de ladite pluralité de tâches selon lesdites contraintes de temps.

2. Procédé selon la revendication 1 selon lequel lesdites étapes de
sélection d'une tâche, d'identification d'un cœur et d'affectation d'une tâche
sélectionnée à un cœur identifié sont répétées pour affecter un ensemble de
25 tâches de ladite pluralité de tâches à au moins un cœur de ladite pluralité de
cœurs.

3. Procédé selon la revendication 1 ou la revendication 2
comprenant en outre une étape d'assignation d'une priorité à chaque tâche
affectée à un cœur, ladite priorité étant déterminée en fonction de l'ordre
30 d'assignation des tâches audit cœur.

4. Procédé selon l'une quelconque des revendications 1 à 3 selon
lequel ladite étape d'identification d'un cœur comprend les étapes suivantes :

- sélection d'un cœur de ladite pluralité de cœurs ;
 - évaluation de la capacité dudit cœur sélectionné à exécuter ladite tâche sélectionnée selon des contraintes de temps associées à ladite tâche sélectionnée et les contraintes de temps associées aux tâches préalablement affectées audit cœur sélectionné ; et
- 5
- si ledit cœur sélectionné n'est pas capable d'exécuter ladite tâche sélectionnée selon des contraintes de temps associées à ladite tâche sélectionnée et des contraintes de temps associées aux tâches préalablement affectées audit cœur sélectionné, répétition des étapes de sélection d'un cœur
- 10 et d'évaluation.

5. Procédé selon la revendication 4 selon lequel ladite pluralité de cœurs forme une liste ordonnée de cœurs, un cœur étant sélectionné de façon incrémentale dans ladite liste ordonnée de cœurs.

6. Procédé selon l'une quelconque des revendications précédentes
- 15 selon lequel ladite étape d'identification d'un cœur comprend une étape de décision pour décider qu'une tâche T_r ne peut être placée sur un cœur C_p sur lequel ont été préalablement placées un ensemble de tâches T_s , s variant de 1 à $(r - 1)$, que si, et seulement si, pour tout k appartenant à l'intervalle $[1, r]$,

$$\sum_{i=k}^r \text{CEIL}\left(\frac{DL(T_k)}{\delta(T_i)} \cdot WCET(T_i)\right) < DL(T_k)$$

- où $\text{CEIL}()$ est une fonction retournant l'entier immédiatement supérieur à la
- 20 valeur de l'argument, $WCET(T_i)$ représente le pire temps d'exécution de la tâche T_i , $DL(T_k)$ représente le temps de réponse imparti à la tâche T_k et $\delta(T_i)$ la période d'exécution de la tâche T_i .

7. Programme d'ordinateur comprenant des instructions adaptées à la mise en œuvre de chacune des étapes du procédé selon l'une quelconque
- 25 des revendications 1 à 6 lorsque ledit programme est exécuté sur un ordinateur.

8. Dispositif comprenant des moyens adaptés à la mise en œuvre de chacune des étapes du procédé selon l'une quelconque des revendications 1 à 6.

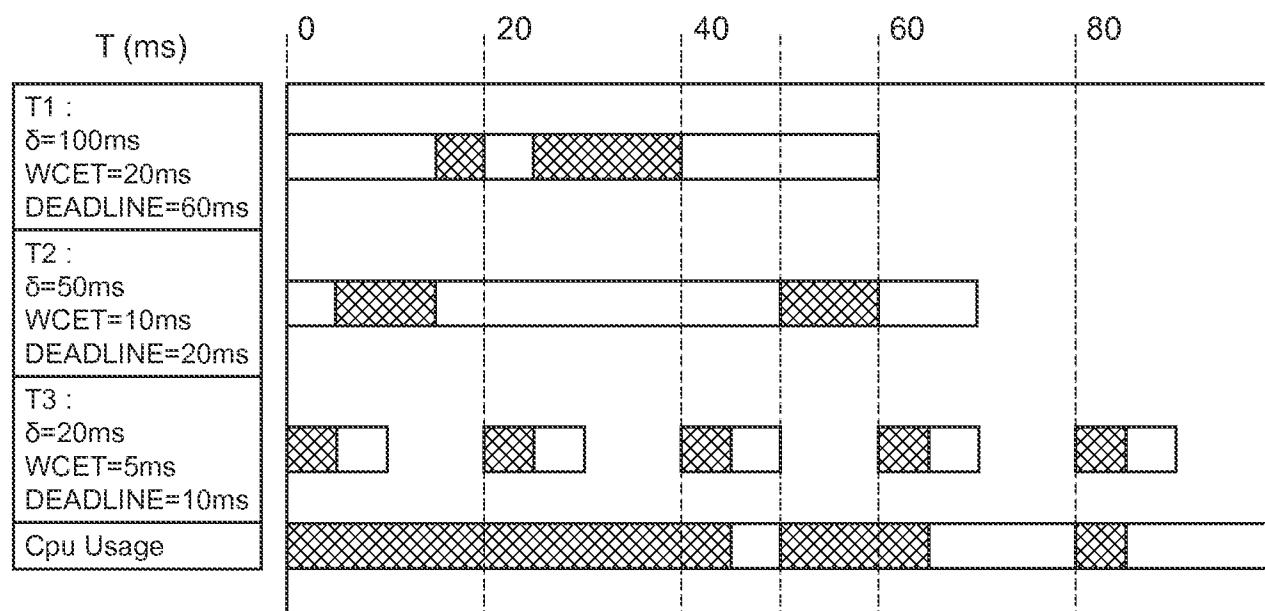
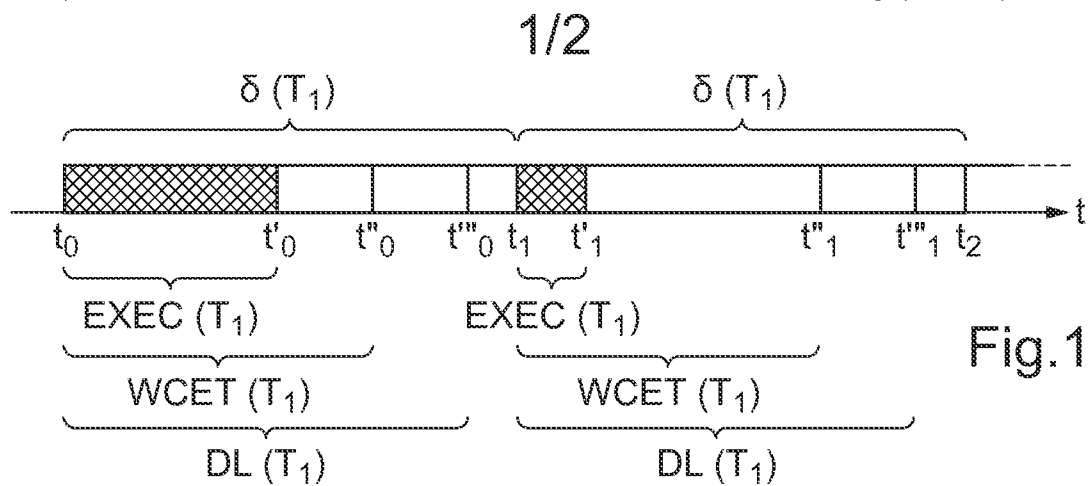
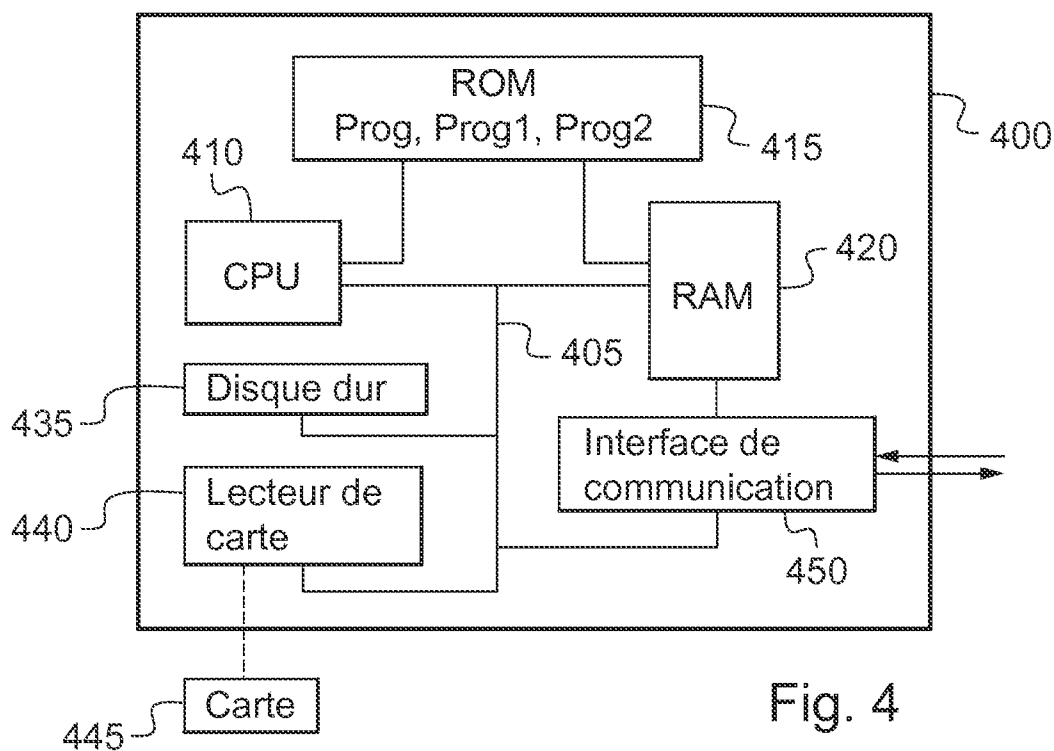


Fig.3



2/2

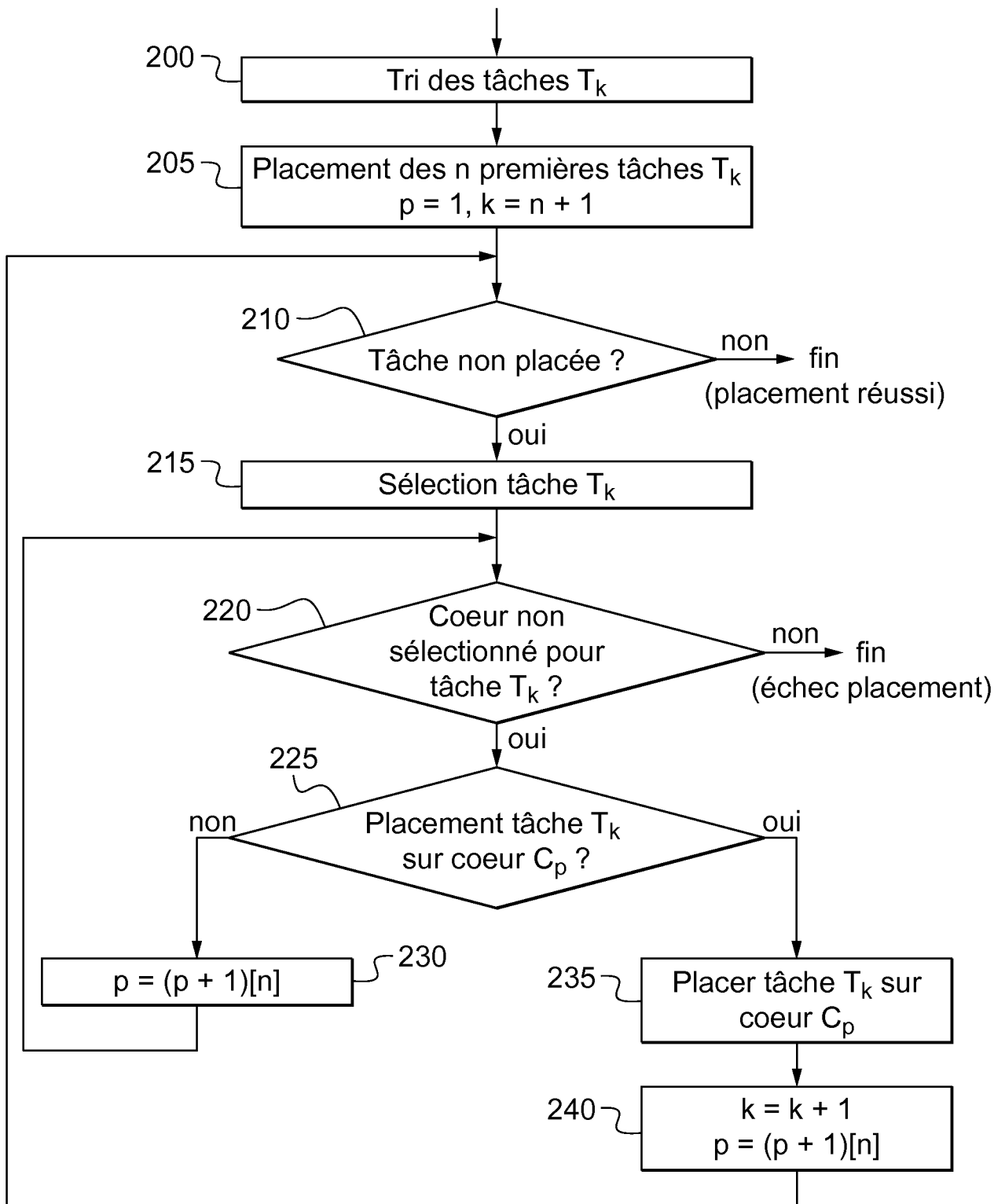


Fig.2

INTERNATIONAL SEARCH REPORT

International application No
PCT/FR2013/052633A. CLASSIFICATION OF SUBJECT MATTER
INV. G06F9/48 G06F9/50
ADD.

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
G06F

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

EPO-Internal

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	B. Berna: "Joint tasks and cache partitioning for real-time systems", Master Thesis, 5 June 2012 (2012-06-05), pages 1-34, XP055078069, Université de Rennes I ENS Cachan Retrieved from the Internet: URL: http://dumas.ccsd.cnrs.fr/docs/00/72/51/72/PDF/Berna.pdf [retrieved on 2013-09-06] the whole document ----- -/--	1-8



Further documents are listed in the continuation of Box C.



See patent family annex.

* Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier application or patent but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"&" document member of the same patent family

Date of the actual completion of the international search

28 March 2014

Date of mailing of the international search report

04/04/2014

Name and mailing address of the ISA/

European Patent Office, P.B. 5818 Patentlaan 2
NL - 2280 HV Rijswijk
Tel. (+31-70) 340-2040,
Fax: (+31-70) 340-3016

Authorized officer

Michel, Thierry

INTERNATIONAL SEARCH REPORT

International application No
PCT/FR2013/052633

C(Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT		
Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	BARUAH S ET AL: "The Partitioned Scheduling of Sporadic Real-Time Tasks on Multiprocessor Platforms", PARALLEL PROCESSING, 2005. ICPP 2005 WORKSHOPS. INTERNATIONAL CONFERENCE WORKSHOPS ON OSLO, NORWAY 14-17 JUNE 2005, PISCATAWAY, NJ, USA, IEEE, 14 June 2005 (2005-06-14), pages 346-353, XP010820985, DOI: 10.1109/ICPPW.2005.83 ISBN: 978-0-7695-2381-1 page 348, right-hand column, line 15 - page 349, right-hand column, line 11 -----	1-8
A	ALAN BURNS ET AL: "Delivering Real-Time Behaviour", 17 September 2007 (2007-09-17), DOMAIN MODELING AND THE DURATION CALCULUS; [LECTURE NOTES IN COMPUTER SCIENCE], SPRINGER BERLIN HEIDELBERG, BERLIN, HEIDELBERG, PAGE(S) 1 - 50, XP019100284, ISBN: 978-3-540-74963-9 the whole document -----	1,6-8
A	Xu Chao ET AL: "An Improved Static-Priority Scheduling Algorithm for Multi-Processor Real-Time Systems", Master of Science Thesis in Secure and Dependable Computer System, 1 June 2010 (2010-06-01), pages 1-99, XP055078096, Chalmers University of Technology University of Gothenburg Retrieved from the Internet: URL: http://publications.lib.chalmers.se/records/fulltext/129034.pdf [retrieved on 2013-09-06] the whole document -----	1-8
A	Ajit Ramachandran ET AL: "Real-Time Scheduling methods for High Performance Signal Processing Applications on Multicore platform", Master report, IDE 1260, 1 August 2012 (2012-08-01), pages 1-62, XP055078097, School of Information Science, Computer and Electrical Engineering Halmstad University, Sweden Retrieved from the Internet: URL: http://hh.diva-portal.org/smash/get/diva2:549983/FULLTEXT01.pdf [retrieved on 2013-09-06] the whole document -----	1-8

RAPPORT DE RECHERCHE INTERNATIONALE

Demande internationale n°

PCT/FR2013/052633

A. CLASSEMENT DE L'OBJET DE LA DEMANDE INV. G06F9/48 G06F9/50 ADD.		
Selon la classification internationale des brevets (CIB) ou à la fois selon la classification nationale et la CIB		
B. DOMAINES SUR LESQUELS LA RECHERCHE A PORTE Documentation minimale consultée (système de classification suivi des symboles de classement) G06F		
Documentation consultée autre que la documentation minimale dans la mesure où ces documents relèvent des domaines sur lesquels a porté la recherche		
Base de données électronique consultée au cours de la recherche internationale (nom de la base de données, et si cela est réalisable, termes de recherche utilisés) EPO-Internal		
C. DOCUMENTS CONSIDERES COMME PERTINENTS		
Catégorie*	Identification des documents cités, avec, le cas échéant, l'indication des passages pertinents	no. des revendications visées
X	B. Berna: "Joint tasks and cache partitioning for real-time systems", Master Thesis, 5 juin 2012 (2012-06-05), pages 1-34, XP055078069, Université de Rennes I ENS Cachan Extrait de l'Internet: URL: http://dumas.ccsd.cnrs.fr/docs/00/72/51/72/PDF/Berna.pdf [extrait le 2013-09-06] le document en entier ----- -/--	1-8
☒ Voir la suite du cadre C pour la fin de la liste des documents ☐ Les documents de familles de brevets sont indiqués en annexe		
* Catégories spéciales de documents cités: "A" document définissant l'état général de la technique, non considéré comme particulièrement pertinent "E" document antérieur, mais publié à la date de dépôt international ou après cette date "L" document pouvant jeter un doute sur une revendication de priorité ou cité pour déterminer la date de publication d'une autre citation ou pour une raison spéciale (telle qu'indiquée) "O" document se référant à une divulgation orale, à un usage, à une exposition ou tous autres moyens "P" document publié avant la date de dépôt international, mais postérieurement à la date de priorité revendiquée "T" document ultérieur publié après la date de dépôt international ou la date de priorité et n'appartenant pas à l'état de la technique pertinent, mais cité pour comprendre le principe ou la théorie constituant la base de l'invention "X" document particulièrement pertinent; l'invention revendiquée ne peut être considérée comme nouvelle ou comme impliquant une activité inventive par rapport au document considéré isolément "Y" document particulièrement pertinent; l'invention revendiquée ne peut être considérée comme impliquant une activité inventive lorsque le document est associé à un ou plusieurs autres documents de même nature, cette combinaison étant évidente pour une personne du métier "&" document qui fait partie de la même famille de brevets		
Date à laquelle la recherche internationale a été effectivement achevée	Date d'expédition du présent rapport de recherche internationale	
28 mars 2014	04/04/2014	
Nom et adresse postale de l'administration chargée de la recherche internationale Office Européen des Brevets, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Fax: (+31-70) 340-3016		Fonctionnaire autorisé Michel, Thierry

C(suite). DOCUMENTS CONSIDERES COMME PERTINENTS		
Catégorie*	Identification des documents cités, avec, le cas échéant, l'indication des passages pertinents	no. des revendications visées
X	BARUAH S ET AL: "The Partitioned Scheduling of Sporadic Real-Time Tasks on Multiprocessor Platforms", PARALLEL PROCESSING, 2005. ICPP 2005 WORKSHOPS. INTERNATIONAL CONFERENCE WORKSHOPS ON OSLO, NORWAY 14-17 JUNE 2005, PISCATAWAY, NJ, USA, IEEE, 14 juin 2005 (2005-06-14), pages 346-353, XP010820985, DOI: 10.1109/ICPPW.2005.83 ISBN: 978-0-7695-2381-1 page 348, colonne de droite, ligne 15 - page 349, colonne de droite, ligne 11 -----	1-8
A	ALAN BURNS ET AL: "Delivering Real-Time Behaviour", 17 septembre 2007 (2007-09-17), DOMAIN MODELING AND THE DURATION CALCULUS; [LECTURE NOTES IN COMPUTER SCIENCE], SPRINGER BERLIN HEIDELBERG, BERLIN, HEIDELBERG, PAGE(S) 1 - 50, XP019100284, ISBN: 978-3-540-74963-9 le document en entier -----	1,6-8
A	Xu Chao ET AL: "An Improved Static-Priority Scheduling Algorithm for Multi-Processor Real-Time Systems", Master of Science Thesis in Secure and Dependable Computer System, 1 juin 2010 (2010-06-01), pages 1-99, XP055078096, Chalmers University of Technology University of Gothenburg Extrait de l'Internet: URL: http://publications.lib.chalmers.se/records/fulltext/129034.pdf [extrait le 2013-09-06] le document en entier -----	1-8
A	Ajit Ramachandran ET AL: "Real-Time Scheduling methods for High Performance Signal Processing Applications on Multicore platform", Master report, IDE 1260, 1 août 2012 (2012-08-01), pages 1-62, XP055078097, School of Information Science, Computer and Electrical Engineering Halmstad University, Sweden Extrait de l'Internet: URL: http://hh.diva-portal.org/smash/get/diva2:549983/FULLTEXT01.pdf [extrait le 2013-09-06] le document en entier -----	1-8