(19) RÉPUBLIQUE FRANÇAISE

INSTITUT NATIONAL
DE LA PROPRIÉTÉ INDUSTRIELLE

PARIS

(11) **N° de publication :** **2 900 004**
(à n'utiliser que pour les commandes de reproduction)

(21) **N° d'enregistrement national :** **06 03431**

(51) Int Cl[8] : **H 03 M 7/40** (2006.01), H 04 N 7/24

(12) DEMANDE DE BREVET D'INVENTION A1

(22) **Date de dépôt :** 18.04.06.

(30) **Priorité :**

(43) **Date de mise à la disposition du public de la demande :** 19.10.07 Bulletin 07/42.

(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) :** *THOMSON LICENSING Société par actions simplifiée* — FR.

(72) **Inventeur(s) :** LE LANN JEAN CHRISTOPHE, COCHEREL GILDAS, JOLLIVET CHRISTOPHE et FOSSARD MICHAEL.

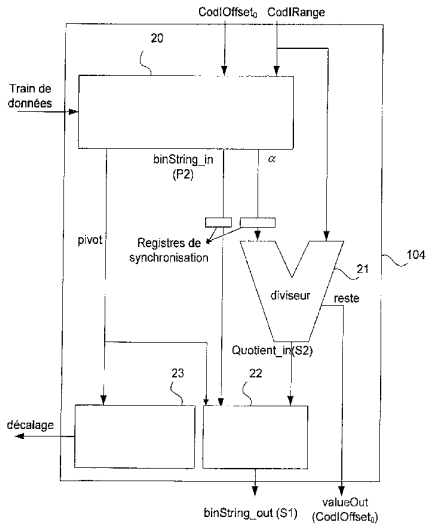
(73) **Titulaire(s) :**

(74) **Mandataire(s) :** THOMSON.

(54) PROCEDE ET DISPOSITIF DE DECODAGE ARITHMETIQUE.

(57) L'invention concerne un procédé de décodage d'éléments codés selon un procédé de codage arithmétique tel que CABAC. Le procédé décode au moins une partie du train binaire en un ensemble d'éléments à partir de premier et second paramètres de décodage $CodIOffset_0$ et CodIRange, l'ensemble d'éléments comprenant un préfixe (P2) composé de n premiers éléments et un suffixe (S2) comprenant m seconds éléments, le préfixe (P2) et le suffixe (S2) étant séparés par un élément appelé pivot (P). Le procédé comprend les étapes suivantes:

- déterminer la valeur de n à partir des premier et second paramètres de décodage et des valeurs de k bits consécutifs du train binaire, dits k premiers bits, pour en déduire le préfixe (P2) et la valeur de m; et
- déterminer le suffixe (S2) à partir des premier et second paramètres de décodage et des valeurs de m bits consécutifs du train binaire qui suivent les n premier bits des k premiers bits.



FR 2 900 004 - A1

PROCEDE ET DISPOSTIF DE DECODAGE ARITHMETIQUE

1. Domaine de l'invention

L'invention concerne un procédé et un dispositif de décodage d'éléments codés selon un procédé de codage arithmétique contextuel, lesdits éléments codés se présentant sous la forme d'un train de données.

2. Etat de l'art

Des données numériques de type audio ou vidéo peuvent être codées selon divers procédés de codage. Les procédés de codage arithmétique sont du domaine connu et permettent notamment de coder une donnée ou un élément syntaxique (p. ex. un vecteur de mouvement) avec un nombre non entier de bits. Ainsi, un même bit peut contenir de l'information relative à deux éléments syntaxiques différents contrairement à ce que permet un codage d'Huffman. Un tel bit est dans ce cas appelé bit fractionnaire.

La norme de codage vidéo H.264 également connu sous le nom de MPEG-4 partie 10 ou MPEG-4 AVC propose un procédé de codage entropique qui est un procédé de codage arithmétique contextuel connu sous le nom de CABAC («Context-based Adaptive Binary Arithmetic Coding» en anglais). Ce procédé de codage est adaptatif dans la mesure où les valeurs précédemment codées influent sur le codage des éléments syntaxiques courants. Afin d'améliorer les temps de décodage des éléments syntaxiques entre les étapes de décodage entropique et les traitements ultérieurs (p.ex. quantification inverse) et donc réduire la taille mémoire nécessaire, il est souhaitable de paralléliser au moins en partie le procédé de décodage arithmétique contextuel, ce qui est relativement complexe en raison de la nature séquentielle dudit procédé de décodage.

3. Résumé de l'invention

L'invention a pour but de pallier au moins un des inconvénients de l'art antérieur. Plus particulièrement, l'invention concerne un procédé de décodage d'éléments codés selon un procédé de codage arithmétique tel que CABAC, les éléments codés se présentant sous la forme d'un train binaire. Le procédé décode au moins une partie du train binaire en un ensemble d'éléments à

partir de premier et second paramètres de décodage prédéfinis définissant respectivement la borne inférieure ($CodIOffset_0$) et la taille (CodIRange) d'un intervalle, l'ensemble d'éléments comprenant un préfixe (P2) composé de n premiers éléments de valeur identique prédéterminée et un suffixe (S2) comprenant m seconds éléments, m dépendant de n selon une fonction prédéterminée, le préfixe (P2) et le suffixe (S2) étant séparés par un élément appelé pivot (P). Le procédé comprend les étapes suivantes :

- déterminer la valeur de n à partir des premier et second paramètres de décodage et des valeurs de k bits consécutifs du train binaire, dits k premiers bits, k étant un entier prédéterminé supérieur ou égal à n, pour en déduire le préfixe (P2) et la valeur de m; et
- déterminer le suffixe (S2) à partir des premier et second paramètres de décodage et des valeurs de m bits consécutifs du train binaire qui suivent les n premier bits des k premiers bits.

Préférentiellement, les n premiers éléments, le pivot (P) et les m seconds éléments sont des bits.

Selon un mode de réalisation particulier, la valeur de n est déterminée de telle sorte que n soit le plus petit entier satisfaisant la relation suivante :

$$2^n\ CodIOffset_0 + val(n) - (2^n - 1) * CodIRange < CodIRange$$

Où : - val(n) est la valeur correspondant aux n premiers bits du train binaire;

- $CodIOffset_0$ est la valeur du premier paramètre définissant la borne inférieure de l'intervalle; et
- CodIRange est la valeur du second paramètre définissant la taille de l'intervalle.

En outre, le suffixe (S2) est déterminé de telle sorte que sa valeur soit égale au quotient $(2^n\ CodIOffset_0 + val(n))/codIRange$.

L'invention concerne également un dispositif de décodage d'éléments codés selon un dispositif de codage arithmétique tel que CABAC, les éléments codés se présentant sous la forme d'un train binaire. Le dispositif décode au moins une partie du train binaire en un ensemble d'éléments à partir de premier et second paramètres de décodage prédéfinis définissant respectivement la borne inférieure ($CodIOffset_0$) et la taille (CodIRange) d'un intervalle, l'ensemble d'éléments comprenant un préfixe (P2) composé de n

premiers éléments de valeur identique prédéterminée et un suffixe (S2) comprenant m seconds éléments, m dépendant de n selon une fonction prédéterminée, le préfixe (P2) et le suffixe (S2) étant séparés par un élément appelé pivot (P). Le dispositif comprend:

- des moyens pour déterminer la valeur de n à partir des premier et second paramètres de décodage et des valeurs de k bits consécutifs du train binaire, dits k premiers bits, k étant un entier prédéterminé supérieur ou égal à n, pour en déduire le préfixe (P2) et la valeur de m; et
- des moyens pour déterminer le suffixe (S2) à partir des premier et second paramètres de décodage et des valeurs d'au plus m bits consécutifs du train binaire qui suivent les n premier bits des k premiers bits.

Préférentiellement, les moyens pour déterminer le suffixe (S2) comprennent un diviseur sur silicium.

Avantageusement, le dispositif de décodage comprend en outre des moyens pour concaténer le préfixe P2, le pivot P et le suffixe S2.

Préférentiellement, caractérisé en ce que le dispositif de décodage comprend en outre des moyens pour décaler le train binaire d'un nombre entiers de bits égal à m+n+1.

L'invention concerne également un produit programme d'ordinateur qui comprend des instructions de code de programme pour l'exécution des étapes du procédé selon l'invention, lorsque le programme est exécuté sur un ordinateur.

4. Listes des figures

L'invention sera mieux comprise et illustrée au moyen d'exemples de modes de réalisation et de mise en œuvre avantageux, nullement limitatifs, en référence aux figures annexées sur lesquelles :

- la figure 1 illustre un procédé de décodage, selon l'état de l'art, de bits appelés « bins » représentant des éléments syntaxiques ou une partie de tels éléments codés dans un mode particulier appelé mode bypass conformément à la norme H.264;

- la figure 2 représente les bins d'un élément syntaxique particulier présentant une certaine régularité;
- la figure 3 représente un dispositif de décodage arithmétique contextuel selon l'invention sous forme de modules;
- la figure 4 représente une partie d'un module du dispositif de décodage arithmétique contextuel représenté par la figure 3;
- la figure 5 représente un premier module de calcul de la partie du dispositif de décodage représentée par la figure 4;
- la figure 6 représente un premier bloc de calcul du module illustré par la figure 5;
- la figure 7 représente un deuxième bloc de calcul du module illustré par la figure 5;
- la figure 8 représente un troisième bloc de calcul du module illustré par la figure 5;
- la figure 9 représente un second module de calcul de la partie du dispositif de décodage représentée par la figure 4; et
- la figure 10 représente un troisième module de calcul de la partie du dispositif de décodage représentée par la figure 4.

5. Description détaillée de l'invention

L'invention est décrite dans le cadre d'une application de décodage d'une séquence d'images conformément à la norme H.264 lorsque le mode de codage arithmétique contextuel CABAC est activé. Lors du codage d'une séquence d'images, certains éléments syntaxiques, p. ex. les vecteurs de mouvement, apparaissent fréquemment. Il est donc important de pouvoir les décoder de manière efficace. A cet effet, l'invention a pour objectif de paralléliser au moins partiellement le procédé de décodage de données numériques codées préalablement un procédé de codage arithmétique contextuel tel CABAC.

Conformément à la norme H.264, les éléments syntaxiques ne sont pas directement codés par un procédé de codage arithmétique. En effet, un élément syntaxique est, lors d'une première étape de codage appelé étape de binarisation, codé en utilisant un code à longueur variable constitué de bits

appelés « bins » pour les distinguer des bits du train de données final à l'aide de tables et/ou procédés définis dans le document intitulé « Draft of version 4 of H.264/AVC (ITU-T Recommendation H.264 and ISO/IEC 14496-10 (MPEG-4 part 10) Advanced Video Coding) » de l'ISO/IEC MPEG et ITU-T VCEG référencé JVT-M050d4. L'ensemble des bins ainsi généré est codé, lors d'une deuxième étape, par un procédé de codage arithmétique contextuel en un ensemble de bits qui est le train de données codées (« bitstream » en anglais). Dès lors, à la différence d'un procédé de décodage entropique classique (p.ex. tel que celui utilisé dans le cadre de la norme MPEG-2), un procédé décodage entropique tel que celui utilisé dans le cadre de la norme H.264 comprend les étapes suivantes :

- lire un certain nombre de bits dans le train de données codées ;
- générer lors d'une étape de décodage arithmétique contextuel, à partir de ces bits, un second train de bits successifs ou bins représentant un élément syntaxique (comme par exemple un coefficient DCT ou une composante de vecteur) de manière séquentielle ; et
- générer l'élément syntaxique à partir des bins générés lors de l'étape précédente.

La dernière étape du procédé de décodage est appelée étape de débinarisation. Un nombre donné de bits peut être transformé en un nombre de bins différents. Ces bins sont mémorisés et interprétés afin de retrouver la valeur de l'élément syntaxique complet. Il est à noter que lorsqu'une valeur d'élément syntaxique a été décodée, il est possible de continuer à extraire à partir du même dernier bit de l'information pour l'élément syntaxique suivant. Dans le cas de la norme H.264, le procédé de codage peut se placer dans un mode de codage arithmétique dégradé (appelé mode « bypass » en anglais) suivant le type d'élément syntaxique qu'il cherche à coder. Le procédé de codage met alors la valeur d'un paramètre particulier appelé bypassFlag à 1. La valeur de ce paramètre est déterminée par le procédé de décodage selon l'invention en fonction du type d'élément syntaxique que celui-ci doit décoder. En effet, la norme H.264 spécifie si un élément syntaxique donné doit être codé dans le mode dégradé ou le mode non dégradé. Le mode dégradé est notamment utilisé pour coder des éléments syntaxiques ou des parties de tels éléments dont la distribution est approximativement uniforme. La figure 2

représente les bins d'un élément syntaxique (par exemple les vecteurs de mouvement mvd_l0 et mvd_l1 et les coefficients coef_abs_level_minus1 définis dans la norme H.264) dont au moins une partie est codée selon le mode dégradé. Par exemple, les suffixes S1 des éléments syntaxiques vecteurs de mouvement et des coefficients coef_abs_level_minus1 sont, selon la norme H.264, codés dans ce mode dégradé. Pour les vecteurs de mouvement, les bins du suffixe S1 suivent les 9 premiers bins d'un préfixe P1. Pour les coefficients coef_abs_level_minus1 les bins du suffixe S1 suivent les 14 premiers bins d'un préfixe P1. Si la valeur de tels éléments syntaxiques est inférieure à certaines valeurs (dépendant du nombre de bins dans le préfixe P1) il peut n'y avoir aucun suffixe S2. Les préfixes P1 codés dans le mode non dégradé, i.e. bypassFlag = 0, sont décodés selon un procédé de décodage arithmétique contextuel classique tel que décrit dans le document JVT-M050d4. Les suffixes S1 sont décodés selon le procédé de l'invention.

La figure 1 représente le procédé de décodage selon l'état de l'art et décrit dans le document JVT-M050d4. Ce procédé permet de générer la succession de bins représentant un élément syntaxique ou une partie d'un tel élément codé dans le mode dégradé. Les opérateurs utilisés sont définis dans le document JVT-M050d4 : l'opérateur '<<' est l'opérateur « décalage à gauche », l'opérateur '|' est l'opérateur « OU » et l'opérateur '&' est l'opérateur « ET ». La fonction val(m) retourne la valeur obtenue en « lisant m bits » dans le train de données codées. Cette fonction. Les paramètres de décodage $CodIOffset_0$ et CodIRange qui caractérisent l'état courant du décodeur arithmétique définissent respectivement la borne inférieure et la taille d'un intervalle. Ces paramètres de décodage sont initialisés dans les images au début de chaque tranche (« slice » en anglais) : $CodIOffset_0=0$ et CodIRange=1. binVal désigne la valeur du bin courant. En déroulant le procédé illustré par la figure 1, i.e. en l'appliquant plusieurs fois de suite, on se heurte a priori à la combinatoire de toutes les possibilités de branchement rencontré à chaque itération (branche OUI et NON). Toutefois, il est possible de simplifier ce procédé en le parallèlisant dans le cas du décodage des suffixes S1 des éléments syntaxiques considérés, i.e. vecteurs de mouvements et coefficients. En effet, les suites de bins représentant ces

suffixes S1 ne sont pas aléatoires. Elles présentent des régularités et notamment elles comprennent:

- un préfixe P2 constitué d'une suite de n bins de valeur '1' ;
- un bin '0' appelé pivot et référencé P; et
- un suffixe S2 constitué d'une suite aléatoire de m bins, où m dépend de n selon une fonction prédéterminée par exemple $m=n+n_0$, n_0 étant un entier dont la valeur est connue et dépend de l'élément syntaxique courant.

En déroulant k fois le procédé selon l'état de l'art, on obtient les k relations suivantes :

relation d'ordre 1 :

$$codIOffset_1 = (2 \times codIOffset_0 + val(1)) - codIRange$$

relation d'ordre 2 :

$$\begin{aligned} codIOffset_2 &= (2 \times codIOffset_1 + val(1)) - codIRange \\ &= (2^2 \times codIOffset_0 + val(2)) - (2^2 - 1) \times codIRange \end{aligned}$$

relation d'ordre 3 :

$$codIOffset_3 = (2^3 \times codIOffset_0 + val(3)) - (2^3 - 1) \times codIRange$$

...

relation d'ordre k :

$$codIOffset_k = (2^k \times codIOffset_0 + val(k)) - (2^k - 1) \times codIRange$$

Ainsi, il est possible lors d'une étape 1 à partir des seules valeurs de $CodIOffset_0$ et CodIRange de déterminer la position du pivot P, i.e. la valeur de n et par conséquent la longueur m du suffixe S2. En effet, le plus petit indice p tel que la relation $codIOffset_p < CodIRange$ soit vérifiée détermine précisément la position du pivot. Si l'on suppose que l'indice du premier bin de P2 est zéro alors p=n. Sachant que p est toujours inférieur à k (p.ex. k=14), il suffit dès lors pour déterminer cette valeur de calculer en parallèle k valeurs de $CodIOffset_k$ et de comparer toutes ces valeurs à CodIRange. A l'issue de cette étape, la valeur de $CodIOffset_0$ est mise à jour : $codIOffset_0 = codIOffset_p$. Par la suite, on se place dans le train de données après le $p^{ième}$ bit pour déterminer des valeurs val(1) ... val(m).

Après avoir déterminé la position p du pivot P, le nombre m de bins qu'il reste à générer lors d'une étape 2 pour reconstruire le suffixe S2 du suffixe S1 est connu : $m=n+n_0$. Cependant la valeur de chacun de ces bins

reste à déterminer lors de cette seconde étape. En déroulant à nouveau m fois le procédé selon l'état de l'art, on obtient les m relations suivantes :

$$codIOffset_1 = 2^1 \times codIOffset_0 + val(1) - b_0 \times codIRange$$

$$codIOffset_2 = 2^2 \times codIOffset_0 + val(2) - (2 \times b_0 + b_1) \times codIRange$$

$$codIOffset_3 = 2^3 \times codIOffset_0 + val(3) - (2^2 \times b_0 + 2 \times b_1 + b_2) \times codIRange$$

....

$$codIOffset_m = 2^m codOffset_0 + val(m) - codIRange \times \left\lfloor 2^{m-1} b_0 + 2^{m-2} b_1 + ... + b_{m-1} \right\rfloor$$

Posons $\alpha = 2^m codOffset_0 + val(m)$ et $B = \left\lfloor 2^{m-1} b_0 + 2^{m-2} b_1 + ... + b_{m-1} \right\rfloor$. α est une constante qui dépend des valeurs de m bits du train de données, m étant connue, et de la valeur de $CodIOffset_0$ également connue. La norme impose en outre que la valeur $codIOffset_m$ soit inférieure à *codIRange*.

Dès lors, B est égal au quotient α / *codIRange* et $codIOffset_m$ au reste de cette même division car $\alpha = CodIRange * B + codIOffset_m$ et $codIOffset_m$< *codIRange*.

La valeur *codIRange* est ainsi soustraite un maximum de fois de la valeur α conformément à la norme H.264.

Ayant déterminé la position du pivot lors de l'étape 1 et donc le préfixe P2, puis ayant déterminé la valeur des bins du suffixe S2, i.e. la valeur B lors de l'étape 2, le suffixe S1 est entièrement déterminé par concaténation de P2, S2 et du pivot. A l'issue des étapes 1 et 2, la valeur de CodIRange n'est pas modifiée et la valeur de $CodIOffset_0$ est mise à jour avec la valeur de $CodIOffset_m$. Ces valeurs de CodIRange et $CodIOffset_0$ éventuellement mises à jour sont celles qui seront utilisées lorsque le procédé de codage arithmétique contextuel sera de nouveau utilisé pour décoder des bins que ce soit en mode dégradé ou non.

L'invention concerne également un dispositif de décodage arithmétique contextuel 10 qui implémente le procédé de décodage arithmétique selon l'invention. Sur les figures qui suivent, les modules représentés sont des unités fonctionnelles, qui peuvent ou non correspondre à des unités physiquement distinguables. Par exemple, ces modules ou certains d'entre eux peuvent être regroupés dans un unique composant, ou constituer des fonctionnalités d'un même logiciel. *A contrario*, certains modules peuvent

éventuellement être composés d'entités physiques séparées. En référence à la figure 3, le dispositif de décodage 10 comprend notamment un module de contrôle 100 dont le rôle est de contrôler le chemin de données (« datapath » en anglais) complet. Le module de contrôle 100 se présente par exemple sous la forme d'une machine d'état (FSM - « Finate State Machine » en anglais). Le dispositif de décodage 10 comprend en outre un module 101 permettant de sélectionner à partir des données voisines un contexte stocké dans une mémoire référencée cts_mem sur la figure 3. Le dispositif de décodage 10 comprend par ailleurs un module de décodage arithmétique 102 qui met en oeuvre le procédé de décodage selon l'invention décrit précédemment. Afin de synchroniser les deux étages du traitement pipeline le dispositif de décodage 10 comprend un registre R localisé entre le module de contrôle 100 et le module de décodage arithmétique 102. Le module de décodage arithmétique 102 va éventuellement lire une valeur de contexte dans la mémoire cts_mem et génère un bin à partir du train de données reçu en entrée. Il va également mettre à jour le contexte dans la mémoire cts_mem. Ce bin est transmis à un module de débinarisation 103. Il envoie également une commande de décalage à un décaleur (« shifter » en anglais) externe qui décale le train de données d'un certain nombre de bits. Le module de débinarisation 103 permet de générer les éléments syntaxiques à partir des bins générés par le décodeur arithmétique 102.

Le module de décodage arithmétique 102 comprend un premier module qui met en oeuvre le procédé de décodage arithmétique classique proposé par la norme pour décoder les bins d'éléments syntaxiques ou de parties de tels éléments autre que ceux pour lesquels le procédé de décodage selon l'invention est utilisé (i.e. suffixes des vecteurs de mouvement et des coefficients coef_abs_level_minus1). Le module de décodage arithmétique 102 comprend également un second module 104 représenté sur la figure 4 qui met en œuvre le procédé selon l'invention. Le module 104 comprend notamment un premier module de calcul 20 permettant de calculer la position du pivot P selon l'étape 1 du procédé selon l'invention, de générer le préfixe P2 du suffixe S1 et de calculer la valeur de α. Il comprend en outre un module de calcul 21 permettant de faire la division de α par le paramètre CodIRange afin de calculer la valeur du suffixe S2 et la

nouvelle valeur de $CodIOffset_0$. Ce module de calcul 21 est par exemple un diviseur sur silicium. Le module 104 comprend également un module de concaténation 22 représenté sur la figure 9 permettant de générer le suffixe S1 en concaténant le préfixe P2 généré par le module de calcul 20, le pivot P et le suffixe S2 généré par le module de calcul 21. Le module 104 comprend un module 23 représenté sur la figure 10 permettant de calculer un nombre entier de bits (m+n+1), ledit nombre étant transmis à un décaleur (« shifter » en anglais) chargé de décaler le train de données de ce nombre de bits.

Le module de calcul 20 représenté sur la figure 5 comprend des premiers modules de calcul 200 représentés sur la figure 6 permettant de calculer en parallèle les k relations suivantes :

$$\text{Calcul\_1(k)} = (2^k \times codIOffset_0 + \text{var}(k)) - 2^k \times codIRange)$$

Il comprend également un module 201 représenté sur la figure 7 permettant de déterminer la position p du pivot P en comparant à la valeur nulle les k valeurs Calcul_1(k) calculées par les premiers modules de calcul 200 ce qui revient à comparer les k valeurs de $CodIOffset_k$ à la valeur CodIRange. Il comprend également un premier multiplexeur M_1 permettant de sélectionner la valeur Calcul_1(p) parmi les k valeurs calculées. Le module de calcul 20 comprend en outre des seconds modules de calcul 202 représentés sur la figure 8 permettant de calculer les k relations suivantes :

$$\text{Calcul\_2(k)} = 2^k codOffset_0 + \text{var}(k)$$

Il comprend en outre un multiplexeur M_2 permettant de sélectionner la valeur Calcul_2(m) parmi les k valeurs calculées, la valeur de m étant connue suite à la détermination de la position du pivot. La valeur du paramètre α qui est égale à Calcul_2(m) est utilisée par le module de calcul 21 pour calculer la valeur du suffixe S2 et la nouvelle valeur de l'offset $CodIOffset_0$.

Bien entendu, l'invention n'est pas limitée aux exemples de réalisation mentionnés ci-dessus. En particulier, l'homme du métier peut apporter toute variante dans les modes de réalisation exposés et les combiner pour bénéficier de leurs différents avantages. L'invention décrite dans le cadre de données vidéo n'est pas limitée à ce type de données et peut notamment s'appliquer à des données de type audio, voix. Il n'est pas non plus limité à la

norme H.264 et peut notamment s'appliquer à d'autres normes mettant en œuvre un procédé de codage arithmétique contextuel conformément à H.264.

Revendications

1. Procédé de décodage d'éléments codés selon un procédé de codage arithmétique tel que CABAC, lesdits éléments codés se présentant sous la forme d'un train binaire, ledit procédé décodant au moins une partie dudit train binaire en un ensemble d'éléments à partir de premier et second paramètres de décodage prédéfinis définissant respectivement la borne inférieure ($CodIOffset_0$) et la taille (CodIRange) d'un intervalle, ledit ensemble d'éléments comprenant un préfixe (P2) composé de n premiers éléments de valeur identique prédéterminée et un suffixe (S2) comprenant m seconds éléments, m dépendant de n selon une fonction prédéterminée, ledit préfixe (P2) et ledit suffixe (S2) étant séparés par un élément appelé pivot (P), ledit procédé étant caractérisé en ce qu'il comprend les étapes suivantes :

- déterminer la valeur de n à partir desdits premier et second paramètres de décodage et des valeurs de k bits consécutifs dudit train binaire, dits k premiers bits, k étant un entier prédéterminé supérieur ou égal à n, pour en déduire ledit préfixe (P2) et la valeur de m; et
- déterminer le suffixe (S2) à partir desdits premier et second paramètres de décodage et des valeurs de m bits consécutifs dudit train binaire qui suivent les n premier bits desdits k premiers bits.

2. Procédé selon la revendication 1, caractérisé en ce que lesdits n premiers éléments, le pivot (P) et lesdits m seconds éléments sont des bits.

3. Procédé selon la revendication 2, caractérisé en ce que la valeur de n est déterminée de telle sorte que n soit le plus petit entier satisfaisant la relation suivante :

$$2^n\ CodIOffset_0 + val(n) - (2^n - 1)*\ CodIRange < CodIRange$$

Où : - val(n) est la valeur correspondant aux n premiers bits dudit train binaire;

- $CodIOffset_0$ est la valeur dudit premier paramètre définissant la borne inférieure dudit intervalle; et

- CodIRange est la valeur dudit second paramètre définissant la taille dudit intervalle.

4. Procédé selon la revendication 3, caractérisé en ce que le suffixe (S2) est déterminé de telle sorte que sa valeur soit égale au quotient (2^n $CodIOffset_0$ + val(n))/codIRange.

5. Dispositif (104) de décodage d'éléments codés par un dispositif de codage arithmétique tel que CABAC, lesdits éléments codés se présentant sous la forme d'un train binaire, ledit dispositif décodant au moins une partie dudit train binaire en un ensemble d'éléments à partir de premier et second paramètres de décodage prédéfinis définissant respectivement la borne inférieure ($CodIOffset_0$) et la taille (CodIRange) d'un intervalle, ledit ensemble d'éléments comprenant un préfixe (P2) composé de n premiers éléments de valeur identique prédéterminée et un suffixe (S2) comprenant m seconds éléments, m dépendant de n selon une fonction prédéterminée, ledit préfixe (P2) et ledit suffixe (S2) étant séparés par un élément appelé pivot (P), ledit dispositif (104) étant caractérisé en ce qu'il comprend:

- des moyens (20, 200, 201) pour déterminer la valeur de n à partir des desdits premier et second paramètres de décodage et des valeurs de k bits consécutifs dudit train binaire, dits k premiers bits, k étant un entier prédéterminé supérieur ou égal à n, pour en déduire ledit préfixe (P2) et la valeur de m; et
- des moyens (20, 200, 202, 21) pour déterminer le suffixe (S2) à partir desdits premier et second paramètres de décodage et des valeurs d'au plus m bits consécutifs dudit train binaire qui suivent les n premier bits desdits k premiers bits.

6. Dispositif selon la revendication 5, caractérisé en ce lesdits moyens pour déterminer le suffixe (S2) comprennent un diviseur sur silicium (21).

7. Dispositif selon la revendication 6, caractérisé en ce que ledit dispositif de décodage (104) comprend en outre des moyens (22) pour concaténer ledit préfixe P2, ledit pivot P et ledit suffixe S2.

8. Dispositif selon la revendication 7, caractérisé en ce que ledit dispositif de décodage (104) comprend en outre des moyens (23) pour décaler le train binaire d'un nombre entiers de bits égal à m+n+1.

9. Produit programme d'ordinateur caractérisé en ce qu'il comprend des instructions de code de programme pour l'exécution des étapes du procédé selon l'une quelconque des revendications 1 à 4, lorsque ledit programme est exécuté sur un ordinateur.

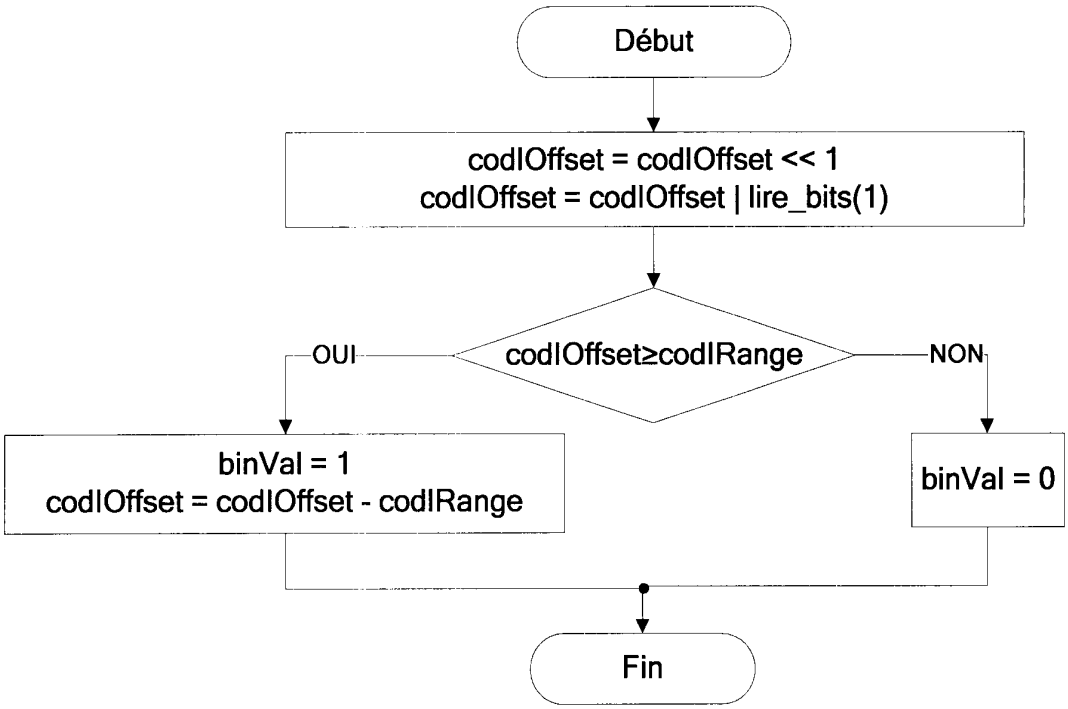


FIG.1 – Etat de l'art

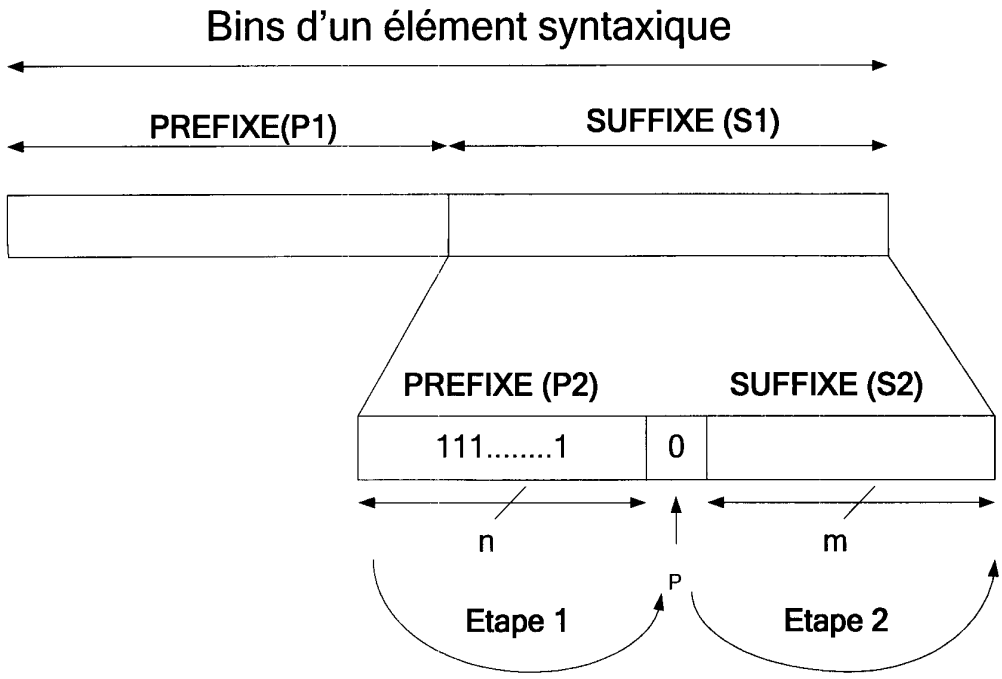


FIG.2

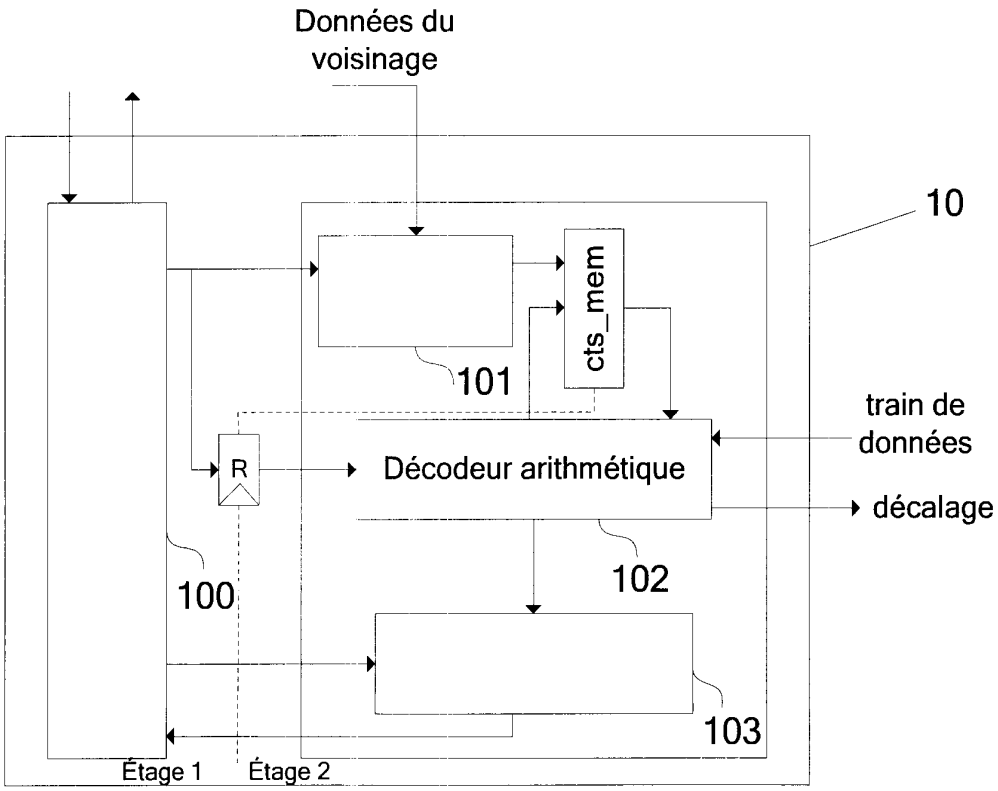

Données du voisinage
10
cts_mem
101
train de données
R
Décodeur arithmétique
décalage
100
102
103
Étage 1
Étage 2

FIG.3

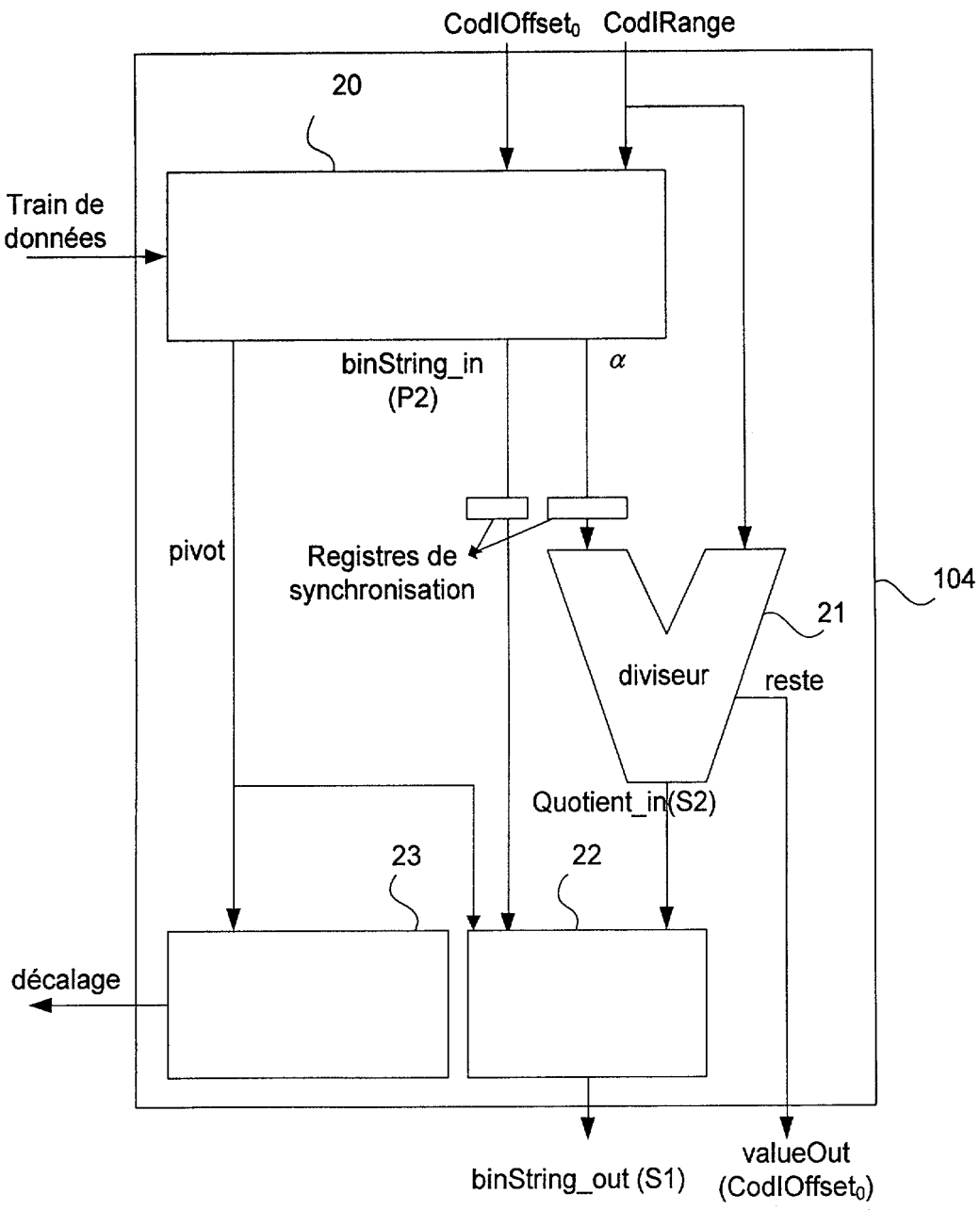

CodIOffset0
CodIRange
20
Train de données
binString_in (P2)
α
pivot
Registres de synchronisation
104
21
diviseur
reste
Quotient_in(S2)
23
22
décalage
binString_out (S1)
valueOut (CodIOffset0)

FIG.4

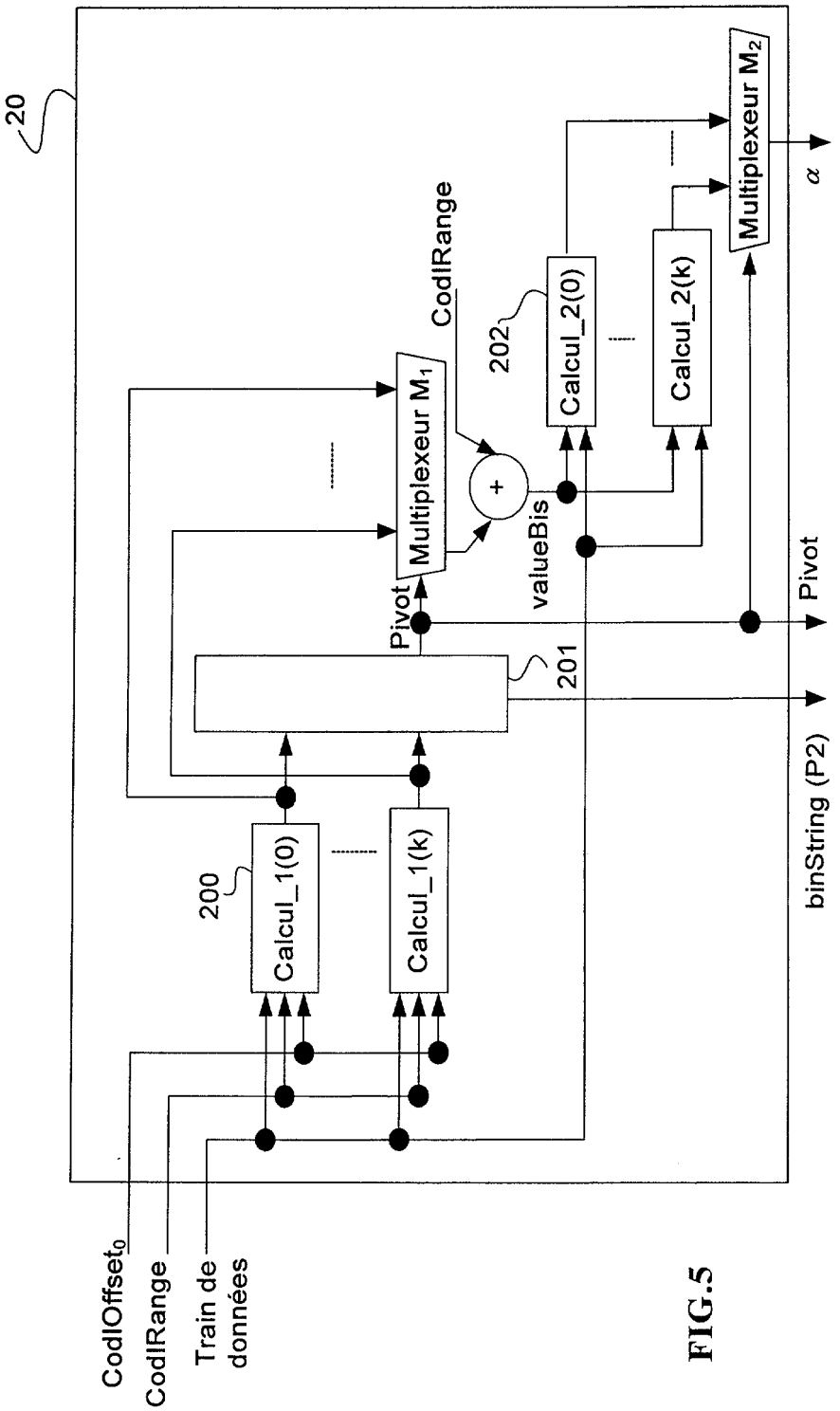

20
CodIOffset0
CodIRange
Train de données
200
Calcul_1(0)
Calcul_1(k)
201
Pivot
Multiplexeur M1
CodIRange
+
valueBis
202
Calcul_2(0)
Calcul_2(k)
Multiplexeur M2
α
Pivot
binString (P2)

FIG.5

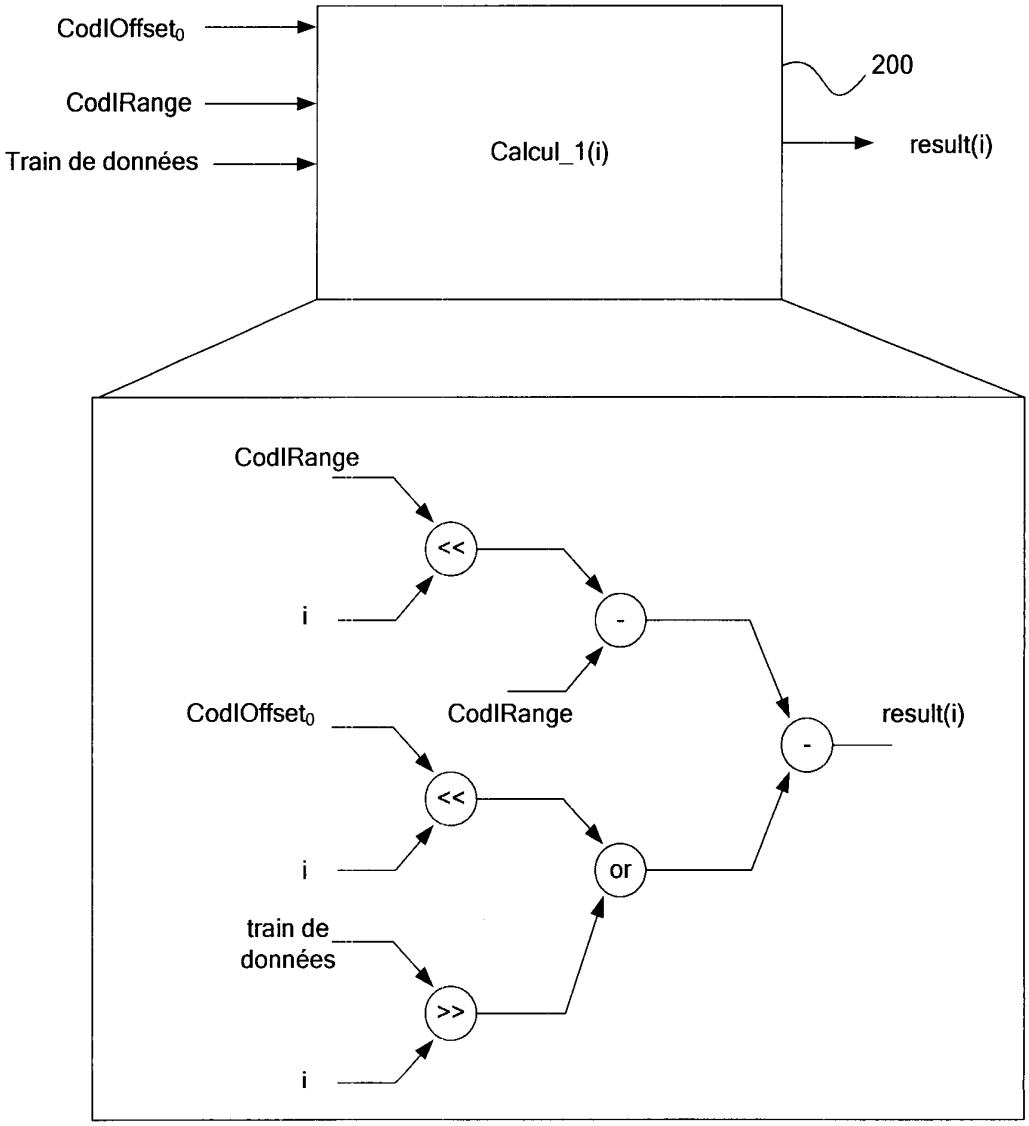

CodIOffset0
CodIRange
Train de données
Calcul_1(i)
200
result(i)
CodIRange
<<
i
-
CodIOffset0
CodIRange
result(i)
<<
-
i
or
train de
données
>>
i

FIG.6

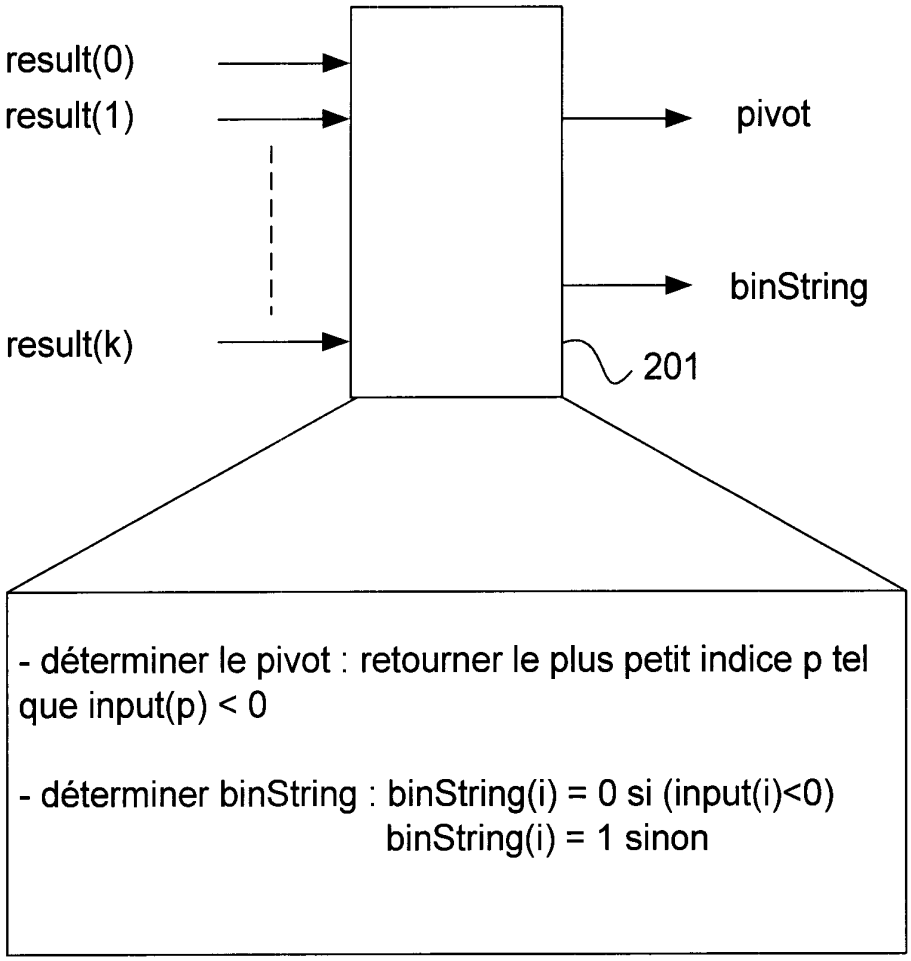

result(0)
result(1)
result(k)
pivot
binString
201
- déterminer le pivot : retourner le plus petit indice p tel que input(p) < 0
- déterminer binString : binString(i) = 0 si (input(i)<0)
binString(i) = 1 sinon

FIG.7

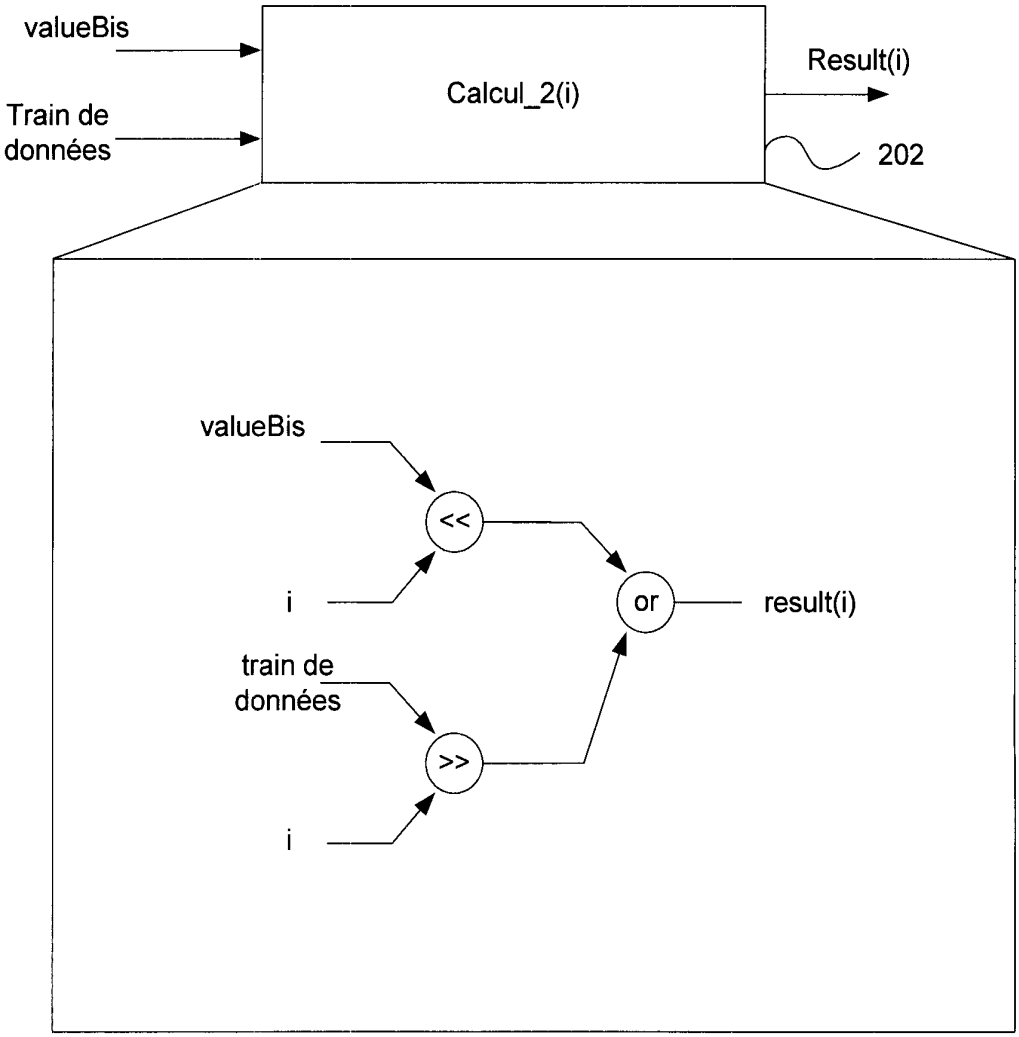

valueBis
Train de données
Calcul_2(i)
Result(i)
202
valueBis
<<
i
or
result(i)
train de données
>>
i

FIG.8

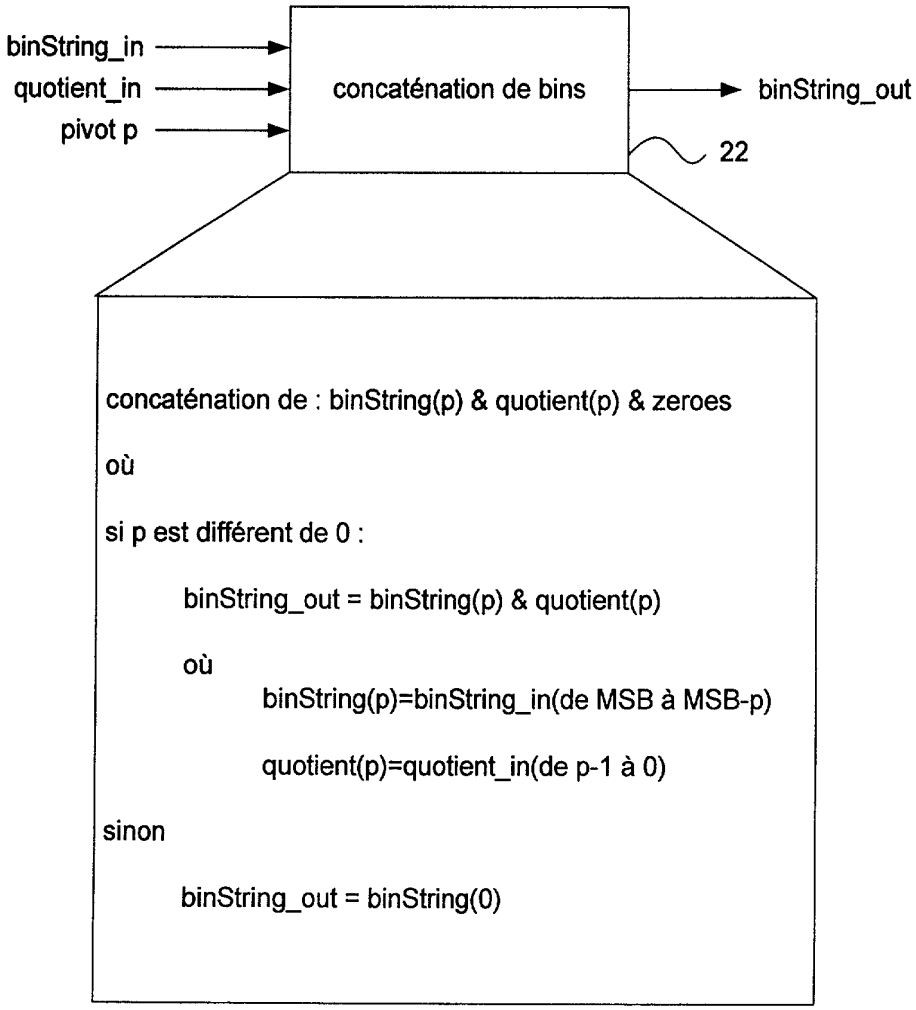

binString_in
quotient_in
pivot p
concaténation de bins
binString_out
22
concaténation de : binString(p) & quotient(p) & zeroes
où
si p est différent de 0 :
binString_out = binString(p) & quotient(p)
où
binString(p)=binString_in(de MSB à MSB-p)
quotient(p)=quotient_in(de p-1 à 0)
sinon
binString_out = binString(0)

FIG.9

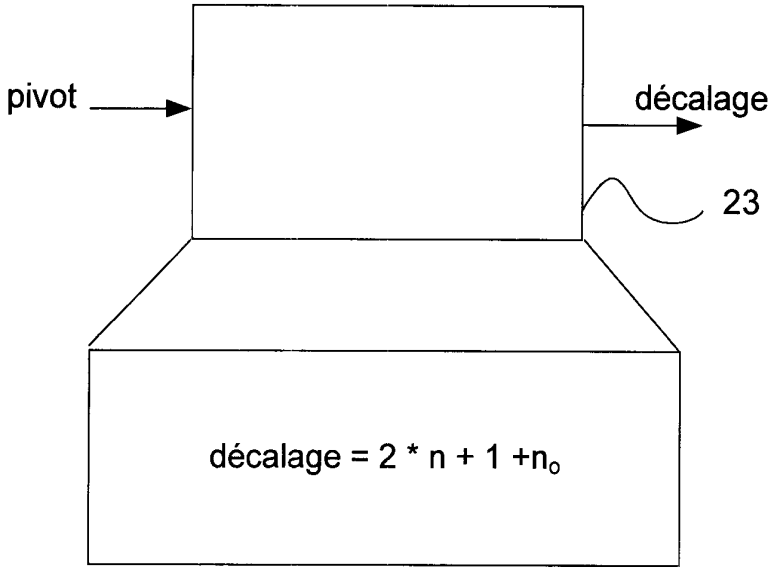


FIG.10

RÉPUBLIQUE FRANÇAISE

2900004

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 680917
FR 0603431

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	EP 1 624 580 A (SAMSUNG ELECTRONICS CO LTD [KR]) 8 février 2006 (2006-02-08) * page 2, colonnes 1-2 * * figure 6 * * page 6, alinéa 42-49 * * page 2, alinéa 11 * * figure 7 * * page 7, alinéa 71 *	1-3,5-9	
D,A	"Advanced video coding for generic audiovisual services" ITU-T STANDARD PRE-PUBLISHED (P), INTERNATIONAL TELECOMMUNICATION UNION, GENEVA,, CH, no. H264 3/5, 1 mars 2005 (2005-03-01), XP017401453 * page 242, alinéa 9.3.3.2.3; figure 9.5 *	1-9	
A	JIAN-WEN CHEN ET AL: "A Hardware Accelerator for Context-Based Adaptive Binary Arithmetic Decoding in H.264/AVC" CIRCUITS AND SYSTEMS, 2005. ISCAS 2005. IEEE INTERNATIONAL SYMPOSIUM ON KOBE, JAPAN 23-26 MAY 2005, PISCATAWAY, NJ, USA,IEEE, 23 mai 2005 (2005-05-23), pages 4525-4528, XP010817170 ISBN: 0-7803-8834-8 * abrégé *	1-9	DOMAINES TECHNIQUES RECHERCHÉS (IPC) H04N H03M
A	US 2004/240559 A1 (PRAKASAM RAMKUMAR [US] ET AL) 2 décembre 2004 (2004-12-02) * abrégé *	1-9	
A	US 2005/018774 A1 (WINGER LOWELL L [CA] ET AL) 27 janvier 2005 (2005-01-27) * abrégé *	1-9	

Date d'achèvement de la recherche	Examinateur
28 février 2007	Di Cagno, Gianluca

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 0603431 FA 680917

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 28-02-2007
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
EP 1624580 A	08-02-2006	JP 2006054877 A	23-02-2006
		KR 20060013021 A	09-02-2006
		US 2006028359 A1	09-02-2006
US 2004240559 A1	02-12-2004	US 2004268329 A1	30-12-2004
US 2005018774 A1	27-01-2005	AUCUN	

EPO FORM P0465