PROCEDE DE GESTION DE DELAIS DANS UN COMMUTATEUR, COMMUTATEUR, PRODUIT PROGRAMME D'ORDINATEUR ET MOYEN DE STOCKAGE CORRESPONDANTS.

Il est proposé un procédé de gestion de délais dans un commutateur d'un réseau de communication de type synchrone. Le commutateur comprend au moins une paire de ports réseau (port d'entrée réseau, port de sortie réseau), permettant une connexion avec un autre commutateur via un lien réseau, et au moins une paire de ports applicatifs (port d'entrée applicatif, port de sortie applicatif), permettant une connexion avec une application via un lien applicatif. Le commutateur reçoit sur chacun des ports d'entrée des données à transférer sur un ou plusieurs des ports de sorties, les données étant réçues sur les ports d'entrée et émises sur les ports de sortie selon un rythme régulier du cycle réseau. Ce procédé comprend les étapes suivantes, effectuées par le commutateur : pour chacune des paires de ports, réseau ou applicatif, du commutateur, détermination (500) de la plus longue chaîne de commutateurs branchée sur la paire de ports ; détermination (501) de la plus longue chaîne de commutateurs passant par le commutateur, à partir des plus longues chaînes de commutateurs passant chacune par un autre commutateur ; détermination (503) d'un délai pour chacun des couples de ports comprenant un port d'entrée et un port de sortie du commutateur n'appartenant pas à la même paire de ports réseau, à partir des plus longues chaînes de commutateurs branchées chacune sur l'une des paires de ports du commutateur et de la plus longue chaîne de commutateurs du réseau.
Procédé de gestion de délais dans un commutateur, commutateur, produit programme d'ordinateur et moyen de stockage correspondants

1. **DOMAINE DE L'INVENTION**

Le domaine de l'invention est celui des réseaux de communication de type synchrone et plus particulièrement les nœuds de commutations, aussi appelés commutateurs par la suite (ou « switches » en anglais), qui constituent l'infrastructure de ces réseaux.

Plus précisément, l'invention concerne une technique de gestion de délais dans un commutateur d'un tel réseau.

Au sein d'un réseau de type synchrone, des applications communiquant les unes avec les autres sont connectées à différents commutateurs. On distingue deux types de synchronisation entre ces applications :

- soit une synchronisation entre une application productrice et ses différentes applications consommatrices,
- soit une synchronisation entre plusieurs applications productrices et leurs différentes applications consommatrices,

le but étant dans tous les cas qu'un ensemble de données générées à un premier instant par une ou plusieurs applications productrices soit reçu à un même second instant (la latence du réseau séparant le second instant du premier instant) par les différentes applications consommatrices.

En fonction de la position des applications les unes par rapport aux autres au sein du réseau, il va donc falloir gérer des délais afin d'assurer cette synchronisation, et ainsi assurer une latence du réseau fixe quels que soient le point d'entrée et le point de sortie du réseau.

2. **ARRIÈRE-PLAN TECHNOLOGIQUE**

Une première technique connue simple consiste à confier cette tâche de synchronisation aux applications. L'application productrice peut chercher à quantifier les distances entre elle et ses différentes applications consommatrices, puis en fonction de ces distances, envoie les données vers les différentes applications consommatrices avec des délais (aussi appelés décalages temporels) adaptés. L'inconvénient de cette
méthode est la duplication d’une même donnée sur le réseau à des instants différents : il en résulte une perte de bande passante.

Selon une deuxième technique connue, la gestion des délais peut aussi être faite au niveau de la réception (c'est-à-dire des applications consommatrices).

Mais ces première et deuxième techniques connues supposent que les applications ont été développées pour prendre en compte ces contraintes et que différentes applications synchrones entre elles doivent pouvoir communiquer pour s'échanger des données de contrôle afin de pouvoir déterminer les délais à appliquer. Or il peut être intéressant de synchroniser des applications qui ne sont pas développées dans ce but. Par exemple, un diaporama généré par un ordinateur PC (première application productrice) est affiché sur un téléviseur (première application consommatrice) pendant qu'un lecteur de CD (deuxième application génératrice) joue de la musique sur des enceintes (deuxième applications consommatrices).

Il semble donc plus avantageux de confier cette gestion de délais aux éléments constituant l’infrastructure du réseau, c'est-à-dire les commutateurs.

On peut citer ici le brevet EP 1289211A1 de Canon Inc (intitulé « Method for setting up an isochronous data stream connection, with the application of a predetermined, total isochronous delay on one or more routing paths »). Ce brevet Canon décrit une méthode permettant de synchroniser une application productrice et plusieurs applications consommatrices. L’application productrice et les applications consommatrices sont reliées entre elles par au moins un pont (« bridge » en anglais). Le but de cette méthode est d’avoir le même délai isochrone total entre l’application productrice et chacune des applications consommatrices, en utilisant des ponts dont le délai isochrone est paramétrable.

Un inconveniент de la technique de ce brevet Canon est qu’elle dépend des positions respectives des applications dans le réseau. En ce sens, elle peut être qualifiée de « dépendante des applications ».

En outre, ce brevet Canon ne décrit pas de méthode permettant de synchroniser plusieurs applications productrices (ainsi que leurs applications consommatrices respectives) entre elles. Il est possible de gérer plusieurs applications productrices mais de manière indépendante.
De plus, dans ce brevet Canon, le délai isochrone total est déterminé chaque fois qu’une application consommatrice est connectée à une application productrice, via le réseau. Ceci nécessite donc un gestionnaire de connexions spécifique.

On peut aussi citer le brevet US 7,185,090 de Sharp Laboratories of America Inc (intitulé « Multimedia Wired/Wireless content synchronization method »). Ce brevet Sharp propose une méthode pour synchroniser un flux de contenu multimédia envoyé vers une pluralité d’appareils consommateurs filaires ou sans-fil. Cette méthode propose de mémoriser (« bufférer ») le flux à la source, déterminer un délai et transmettre ce délai vers les appareils consommateurs. Ces derniers calculent un délai local permettant la synchronisation avec les autres appareils consommateurs. Puis le flux est transmis de la source vers les consommateurs.

Un inconvénient de la technique de ce brevet Sharp est qu’elle peut elle aussi être qualifiée de « dépendante des applications » (au sens précité).

En outre, ici aussi le cas de synchronisation entre différentes applications productrices (et leurs applications consommatrices respectives) n’est pas abordé. Il est possible de gérer plusieurs applications productrices de manière indépendante mais pas de considérer plusieurs applications productrices comme devant être synchronisées entre elles.

De plus, ce brevet Sharp nécessite d’obtenir de l’application productrice certaines caractéristiques comme la longueur des paquets transmis, le débit paquet minimal, le nombre de sauts (« hop » en anglais) radio… L’application productrice doit donc être développée dans ce sens et cette technique ne peut donc pas s’appliquer à n’importe quelle application.

3. **OBJECTIFS DE L’INVENTION**

L’invention, dans au moins un mode de réalisation, a notamment pour objectif de pallier ces différents inconvénients de l’état de la technique.

Plus précisément, dans au moins un mode de réalisation de l’invention, un objectif est de fournir une technique de gestion de délais dans les commutateurs d’un réseau de type synchrone, cette technique permettant une synchronisation entre une application productrice et ses applications consommatrices (c’est-à-dire la détermination d’un délai à appliquer sur des données reçues sur un port d’entrée d’un commutateur.
avant de les transférer vers un ou plusieurs ports de sortie de ce commutateur, dans le souci de permettre à plusieurs applications consommatrices de ces données de les rejoindre au même moment).

Au moins un mode de réalisation de l’invention a également pour objectif de fournir une telle technique permettant une synchronisation entre plusieurs applications productrices et leurs différentes applications consommatrices (c’est-à-dire la détermination d’un délai à appliquer sur des données reçues sur un port d’entrée d’un commutateur avant de les transférer vers un ou plusieurs ports de sortie de ce commutateur, dans le souci de permettre à plusieurs applications productrices de données d’être synchrones entre elles, c’est-à-dire que leurs applications consommatrices respectives joueront en même temps des données provenant de leur application productrice respective et générées au même moment).

Un autre objectif d’au moins un mode de réalisation de l’invention est de fournir une telle technique permettant à des applications non synchrones (c’est-à-dire qui n’auraient pas été développées dans ce but de pouvoir devenir synchrones) de le devenir.

Un objectif complémentaire d’au moins un mode de réalisation de l’invention est de fournir une telle technique qui ne nécessite la prise en compte des caractéristiques des applications à synchroniser ni de leur position respective dans le réseau.

4. **EXPOSÉ DE L’INVENTION**

Dans un mode de réalisation particulier de l’invention, il est proposé un procédé de gestion de délais dans un commutateur d’un réseau de communication de type synchrone comprenant une pluralité de commutateurs, ledit commutateur comprenant au moins une paire de ports réseau comprenant un port d’entrée réseau et un port de sortie réseau, permettant une connexion avec un autre commutateur via un lien réseau, et au moins une paire de ports applicatifs comprenant un port d’entrée applicatif et un port de sortie applicatif, permettant une connexion avec une application via un lien applicatif, ledit commutateur recevant sur chacun des ports d’entrée des données à transférer sur un ou plusieurs des ports de sorties, lesdites données étant reçues sur les ports d’entrée et émises sur les ports de sortie selon un rythme régulier dit cycle réseau. Ledit procédé comprend les étapes suivantes, effectuées par ledit commutateur :
- pour chacune des paires de ports, réseau ou applicatifs, du commutateur, détermination de la plus longue chaîne de commutateurs branchée sur ladite paire de ports ;
- détermination de la plus longue chaîne de commutateurs passant par le commutateur, à partir des plus longues chaînes de commutateurs déterminées pour les différentes paires de ports du commutateur ;
- détermination de la plus longue chaîne de commutateurs du réseau, à partir de la plus longue chaîne de commutateurs passant par le commutateur et des plus longues chaînes de commutateurs passant chacune par un autre commutateur ;
- détermination d’un délai pour chacun des couples de ports comprenant un port d’entrée et un port de sortie du commutateur n’appartenant pas une même paire de ports réseau, à partir des plus longues chaînes de commutateurs branchées chacune sur l’une des paires de ports du commutateur et de ladite plus longue chaîne de commutateurs du réseau.

Ainsi, dans ce mode de réalisation particulier, la technique proposée repose sur une approche tout à fait nouvelle et inventive consistant à gérer des délais dans les commutateurs, uniquement en fonction de la topologie de l’infrastructure, c'est-à-dire la topologie des commutateurs, sans tenir compte des caractéristiques des applications à synchroniser ni de leur position respective dans le réseau. Contrairement aux techniques connues précitées, cette technique n’est donc pas « dépendante des applications » (au sens précité).

La technique proposée n’a donc besoin d’être mise en œuvre qu’à chaque changement de la topologie de l’infrastructure (par ajout, suppression ou déplacement d’au moins un commutateur). En revanche, elle est indépendante des applications réellement connectées aux commutateurs et donc de tout ajout, suppression ou déplacement d’une application. En d’autres termes, elle permet une connexion d’application (qu’elle soit productrice de données ou consommatrice de données) à la volée.

Il est à noter que la mise en œuvre, par tous les commutateurs du réseau, de la technique proposée introduit une latence (décalage temporel) supplémentaire très faible au regard de la latence introduite par la traversée du réseau, et de toute façon non
gênante puisque toutes les applications sont doublement synchronisées entre elles : synchronisation entre chaque application productrice et ses applications consommatrices, et synchronisation entre plusieurs applications productrices et leurs applications consommatrices.

De façon avantageuse, dans ladite étape de détermination d’un délai pour chacun des couples de ports, on utilise la formule suivante : Délai[E,S] = M[S] - N[E], où

- Délai[E,S] est un délai, exprimé en nombre de cycles réseau, pour le couple de ports comprenant le port d’entrée E et le port de sortie S ;
- N[E] est le nombre de commutateurs composant la plus longue chaîne de commutateurs branchée sur la paire de ports, réseau ou applicatifs, à laquelle appartient ledit port d’entrée E ;
- M[S] est :
  * le nombre de commutateurs composant ladite plus longue chaîne de commutateurs du réseau, si le port de sortie S est un port de sortie applicatif ; ou
  * le nombre de commutateurs composant la plus longue des plus longues chaînes de commutateurs branchées chacune sur l’une des paires de ports réseau du commutateur autre que la paire de ports réseau à laquelle appartient ledit port de sortie S, si le port de sortie S est un port de sortie réseau.

Ainsi, le délai pour un couple de ports (E, S) est simple à calculer et dépend uniquement d’une caractéristique (N[E]) préalablement déterminée et associée au port d’entrée E et d’une caractéristique (M[S]) préalablement déterminée et associée au port de sortie S.

Avantageusement, ladite étape de détermination de la plus longue chaîne de commutateurs branchée sur chaque paire de ports comprend les étapes suivantes :

- pour chaque paire de ports p du commutateur, initialisation à zéro d’une variable N[p] correspondant au nombre de commutateurs composant la plus longue chaîne de commutateurs branchée sur ladite paire de ports p ;
- première détection permettant de détecter si une seule paire de ports réseau dudit commutateur est branchée au réseau ;
en cas de première détection positive, envoi sur le port de sortie de ladite seule paire de ports réseau branchée au réseau, d’un message d’un premier type contenant un paramètre à incrémenter k1 possédant la valeur zéro ;

deuxième détection, permettant de détecter la réception, sur un port d’entrée réseau du commutateur, d’un message du premier type contenant un paramètre à incrémenter k1 possédant une valeur x ;

à chaque deuxième détection positive sur un port d’entrée réseau donné p :

* extraction de la valeur x du paramètre à incrémenter k1 contenu dans le message du premier type reçu ;

* si la valeur courante de la variable N[p] associée à la paire de ports réseau p comprenant ledit port d’entrée réseau donné, est inférieure à la valeur x+1, alors remplacement de la valeur courante de la variable N[p] par la valeur x+1 ;

* envoi d’un message du premier type contenant un paramètre à incrémenter k1 possédant la valeur x+1, sur les ports de sortie réseau du commutateur à l’exception du port de sortie réseau compris dans la même paire de ports réseau que ledit port d’entrée réseau donné p.

Ainsi, on optimise le nombre de messages (du premier type) nécessaires pour la détermination de la plus longue chaîne de commutateurs branchée sur chaque paire de ports. En effet, ce nombre est minimisé du fait que seuls les commutateurs situés aux extrémités du réseau vont les initier. De plus, il n’y a pas de réponse attendue par le commutateur initiateur, donc pas de protocole à gérer. La connaissance de la plus longue chaîne de commutateurs reliée à chaque couple de ports se fait en traitant uniquement les valeurs du paramètre k1 reçues, dans les messages du premier type, sur le port d’entrée réseau de ce couple.

Selon une caractéristique avantageuse, chaque étape d’envoi d’un message du premier type est suivie d’une étape de déclenchement d’une première temporisation d’une première durée, et en ce que ladite étape de détermination de la plus longue chaîne de commutateurs branchée sur chaque paire de ports est terminée quand ladite première temporisation est expirée sans qu’un nouveau message du premier type ait été reçu.
De cette façon, on peut aisément fixer une durée maximale de cette phase de l’invention.

De façon avantageuse, ladite étape de détermination de la plus longue chaîne de commutateurs passant par le commutateur comprend les étapes suivantes :

- détermination des deux plus grandes valeurs parmi lesdites plus longues chaînes de commutateurs branchées chacune sur l’une des paires de ports réseau du commutateur ;
- addition desdites deux plus grandes valeurs, pour obtenir le nombre de commutateurs composant ladite plus longue chaîne de commutateurs passant par le commutateur.

Avantageusement, ladite étape de détermination de la plus longue chaîne de commutateurs du réseau comprend les étapes suivantes :

- initialisation à zéro d’une variable MaxPath correspondant au nombre de commutateurs composant la plus longue chaîne de commutateurs du réseau ;
- envoi sur tous les ports de sortie réseau du commutateur, d’un message d’un second type contenant un paramètre k2 possédant comme valeur le nombre de commutateurs composant ladite plus longue chaîne de commutateurs passant par le commutateur ;
- détection, permettant de déceler la réception, sur un port d’entrée réseau du commutateur, d’un message du second type contenant un paramètre k2 possédant une valeur x ;
- à chaque détection positive sur un port d’entrée réseau donné p :
  * extraction de la valeur x du paramètre k2 contenu dans le message du second type ;
  * si la valeur courante de la variable MaxPath est inférieure à la valeur x, alors remplacement de la valeur courante de la variable MaxPath par la valeur x ;
  * envoi d’un message du second type contenant un paramètre k2 possédant la valeur x, sur les ports de sortie réseau du commutateur à l’exception du port de sortie réseau compris dans la même paire de ports réseau que ledit port d’entrée réseau donné p.
Ainsi, on optimise le nombre de messages (du second type) nécessaires pour la détermination de la plus longue chaîne de commutateurs du réseau. En effet, comme pour les messages du premier type, il n’y a pas de réponse attendue pour chaque message du second type envoyé par un commutateur, donc pas de protocole à gérer. La connaissance de la plus longue chaîne de commutateurs du réseau se fait en traitant uniquement les valeurs du paramètre k2 reçues, dans les messages du second type, sur les ports d’entrée réseau du commutateur.

De façon avantageuse, chaque étape d’envoi d’un message du second type est suivie d’une étape de déclenchement d’une seconde temporisation d’une seconde durée, et ladite étape de détermination de la plus longue chaîne de commutateurs du réseau est terminée quand ladite seconde temporisation est expirée sans qu’un nouveau message du second type ait été reçu.

De cette façon, on peut aisément fixer une durée maximale de cette phase de l’invention.

Dans un autre mode de réalisation, l’invention concerne un produit programme d’ordinateur téléchargeable depuis un réseau de communication et/ou enregistré sur un support lisible par ordinateur et/ou exécutable par un processeur, ledit produit programme d’ordinateur comprenant des instructions de code de programme pour la mise en œuvre du procédé précité selon au moins un de ses modes de réalisation, lorsque ledit programme est exécuté sur un ordinateur.

Dans un autre mode de réalisation, l’invention concerne un moyen de stockage, éventuellement totalement ou partiellement amovible, lisible par un ordinateur, stockant un jeu d’instructions exécutables par ledit ordinateur pour mettre en œuvre le procédé précité selon au moins un de ses modes de réalisation.

Dans un autre mode de réalisation, l’invention concerne un commutateur d’un réseau de communication de type synchrone comprenant une pluralité de commutateurs, ledit commutateur comprenant au moins une paire de ports réseau comprenant un port d’entrée réseau et un port de sortie réseau, permettant une connexion avec un autre commutateur via un lien réseau, et au moins une paire de ports applicatifs comprenant un port d’entrée applicatif et un port de sortie applicatif, permettant une connexion avec une application via un lien applicatif, ledit commutateur recevant sur chacun des ports
d’entrée des données à transférer sur un ou plusieurs des ports de sorties, lesdites données étant reçues sur les ports d’entrée et émises sur les ports de sortie selon un rythme régulier dit cycle réseau. Ledit commutateur comprend :

- des moyens de détermination, pour chacune des paires de ports, réseau ou applicatifs, du commutateur, de la plus longue chaîne de commutateurs branchée sur ladite paire de ports ;

- des moyens de détermination de la plus longue chaîne de commutateurs passant par le commutateur, à partir des plus longues chaînes de commutateurs déterminées pour les différentes paires de ports du commutateur ;

- des moyens de détermination de la plus longue chaîne de commutateurs du réseau, à partir de la plus longue chaîne de commutateurs passant par le commutateur et des plus longues chaînes de commutateurs passant chacune par un autre commutateur ;

- des moyens de détermination d’un délai pour chacun des couples de ports comprenant un port d’entrée et un port de sortie du commutateur n’appartenant pas une même paire de ports réseau, à partir des plus longues chaînes de commutateurs branchées chacune sur l’une des paires de ports du commutateur et de ladite plus longue chaîne de commutateurs du réseau.


- Délai[E,S] est un délai, exprimé en nombre de cycles réseau, pour le couple de ports comprenant le port d’entrée E et le port de sortie S ;

- N[E] est le nombre de commutateurs composant la plus longue chaîne de commutateurs branchée sur la paire de ports, réseau ou applicatifs, à laquelle appartient ledit port d’entrée E ;

- M[S] est :
  * le nombre de commutateurs composant ladite plus longue chaîne de commutateurs du réseau, si le port de sortie S est un port de sortie applicatif ; ou

  * le nombre de commutateurs composant la plus longue des plus longues chaînes de commutateurs branchées chacune sur l’une des paires de ports
réseau du commutateur autre que la paire de ports réseau à laquelle appartient ledit port de sortie S, si le port de sortie S est un port de sortie réseau.

Avantageusement, lesdits moyens de détermination de la plus longue chaîne de commutateurs branchée sur chaque paire de ports comprennent:

- des moyens d’initialisation à zéro, pour chaque paire de ports p du commutateur, d’une variable N[p] correspondant au nombre de commutateurs composant la plus longue chaîne de commutateurs branchée sur ladite paire de ports p ;
- des premiers moyens de détection permettant de détecter si une seule paire de ports réseau dudit commutateur est branchée au réseau ;
- des moyens d’envoi, activés en cas de détection positive par lesdits premiers moyens de détection, permettant d’envoyer sur le port de sortie de ladite seule paire de ports réseau branchée au réseau, un message d’un premier type contenant un paramètre à incrémenter k1 possédant la valeur zéro ;
- des deuxième moyens de détection, permettant de détecter la réception, sur un port d’entrée réseau du commutateur, d’un message du premier type contenant un paramètre à incrémenter k1 possédant une valeur x ;
- les moyens suivants, activés à chaque détection positive par lesdits deuxième moyens de détection sur un port d’entrée réseau donné p :

* des moyens d’extraction de la valeur x du paramètre à incrémenter k1 contenu dans le message du premier type reçu ;
* des moyens de remplacement de la valeur courante de la variable N[p] par la valeur x+1, activés si la valeur courante de la variable N[p] associée à la paire de ports réseau p comprenant ledit port d’entrée réseau donné, est inférieure à la valeur x+1 ;
* des moyens d’envoi d’un message du premier type contenant un paramètre à incrémenter k1 possédant la valeur x+1, sur les ports de sortie réseau du commutateur à l’exception du port de sortie réseau compris dans la même paire de ports réseau que ledit port d’entrée réseau donné p.

Selon une caractéristique avantageuse, le commutateur comprend des moyens de déclenchement d’une première temporisation d’une première durée, et lesdits moyens de
détermination de la plus longue chaîne de commutateurs branchée sur chaque paire de ports sont désactivés quand ladite première temporisation est expirée sans qu’un nouveau message du premier type ait été reçu.

De façon avantageuse, lesdits moyens de détermination de la plus longue chaîne de commutateurs passant par le commutateur comprennent :

- des moyens de détermination des deux plus grandes valeurs parmi lesdites plus longues chaînes de commutateurs branchées chacune sur l’une des paires de ports réseau du commutateur ;
- des moyens d’addition desdites deux plus grandes valeurs, pour obtenir le nombre de commutateurs composant ladite plus longue chaîne de commutateurs passant par le commutateur.

Avantageusement, lesdits moyens de détermination de la plus longue chaîne de commutateurs du réseau comprennent :

- des moyens d’initialisation à zéro d’une variable MaxPath correspondant au nombre de commutateurs composant la plus longue chaîne de commutateurs du réseau ;
- des moyens d’envoi sur tous les ports de sortie réseau du commutateur, d’un message d’un second type contenant un paramètre k2 possédant comme valeur le nombre de commutateurs composant ladite plus longue chaîne de commutateurs passant par le commutateur ;
- des moyens de détection, permettant de détecter la réception, sur un port d’entrée réseau du commutateur, d’un message du second type contenant un paramètre k2 possédant une valeur x ;
- les moyens suivants, activés à chaque détection positive sur un port d’entrée réseau donné p :
  * des moyens d’extraction de la valeur x du paramètre k2 contenu dans le message du second type ;
  * des moyens de remplacement de la valeur courante de la variable MaxPath par la valeur x, activés si la valeur courante de la variable MaxPath est inférieure à la valeur x ;
* des moyens d’envoi d’un message du second type contenant un paramètre k2 possédant la valeur x, sur les ports de sortie réseau du commutateur à l’exception du port de sortie réseau compris dans la même paire de ports réseau que ledit port d’entrée réseau donné p.

De façon avantageuse, le commutateur comprend des moyens de déclenchement d’une seconde temporisation d’une seconde durée, et lesdits moyens de détermination de la plus longue chaîne de commutateurs du réseau sont désactivés quand ladite seconde temporisation est expirée sans qu’un nouveau message du second type ait été reçu.

5. **LISTE DES FIGURES**

D’autres caractéristiques et avantages de modes de réalisation de l’invention apparaîtront à la lecture de la description suivante, donnée à titre d’exemple indicatif et non limitatif (tous les modes de réalisation de l’invention ne sont pas limités aux caractéristiques et avantages des modes de réalisation décrits ci-après), et des dessins annexés, dans lesquels :

- la figure 1 décrit un exemple de réseau composé de commutateurs et d’applications productrices et consommatrices, dans lequel peut être mis en œuvre un mode de réalisation du procédé selon l’invention ;

- la figure 2 décrit un exemple de commutateur selon un mode de réalisation particulier de l’invention ;

- la figure 3 décrit un exemple de fonctionnement d’un lien réseau (connexion entre deux commutateurs) d’un point de vue temporel et format de données ;

- la figure 4 décrit un exemple de fonctionnement d’un lien applicatif (connexion entre un commutateur et une application) d’un point de vue temporel et format de données ;

- la figure 5 présente un organigramme d’un mode de réalisation particulier du procédé selon l’invention ;

- la figure 6 détaille un mode de réalisation particulier de l’étape 500 de la figure 5 (algorithme permettant de déterminer la plus longue chaîne de commutateurs branchée sur chaque paire de ports d’un commutateur) ;
- la figure 7 détaille un mode de réalisation particulier de l’étape 501 de la figure 5 (algorithme permettant de déterminer la plus longue chaîne de commutateurs passant par un commutateur) ;
- la figure 8 détaille un mode de réalisation particulier de l’étape 502 de la figure 5 (algorithme permettant de déterminer la plus longue chaîne de commutateurs du réseau) ;
- la figure 9 détaille un mode de réalisation particulier de l’étape 503 de la figure 5 (algorithme permettant la détermination de délais à appliquer aux données entre un port d’entrée et un port de sortie d’un commutateur) ;
- la figure 10 détaille un mode de réalisation particulier de l’étape 504 de la figure 5 (méthode d’application des délais) ;
- la figure 11 présente un exemple de règles de désignation des trois paires de ports réseau et de la paire de ports applicatif de chaque commutateur de la figure 1 ; et
- la figure 12 présente une table donnant les résultats des étapes 500 à 502 de la figure 5 appliquées au réseau de la figure 1, et avec les règles de désignation de la figure 11.

6. DESCRIPTION DÉTAILLÉE

Sur toutes les figures du présent document, les éléments et étapes identiques sont désignés par une même référence numérique.

Prenons l’exemple d’un système de communication par PLC (« Power Line Communication » en anglais ou « Communication sur Courant Porteur » en français) constitué de sept commutateurs (« switch » en anglais) selon un mode de réalisation de l’invention, implémentant la technique de gestion de délai selon un mode de réalisation de l’invention. Sur ces commutateurs, sont branchés des dispositifs applicatifs de deux types :

- applications productrices : dispositifs générant des données à une certaine fréquence propre à l’application ;
- applications consommatrices : dispositifs consommant les données générées et véhiculées au travers du réseau.
Cette invention est particulièrement utile dans le cadre d’applications dites synchrones : une application productrice génère des données à destination de plusieurs applications consommatrices qui doivent jouer la même donnée au même moment. Par exemple, une application productrice peut être un décodeur audio multicanal qui génère des échantillons audio vers plusieurs haut-parleurs (applications consommatrices) connectés au réseau qui doivent restituer les différents canaux au même moment.

Elle est aussi particulièrement utile dans le cadre d’applications devant être synchronisées les unes par rapport aux autres. Par exemple, dans le cadre d’un système home cinéma, un décodeur audio multicanal (première application productrice) génère des échantillons audio vers des haut-parleurs (premières applications consommatrices) connectés sur le réseau, tandis qu’un décodeur vidéo (deuxième application productrice) génère des échantillons vidéo à destination d’un téléviseur (deuxième application consommatrice) connecté lui aussi sur le réseau.

La **figure 1** décrit un réseau composé d’applications productrices et consommatrices et de commutateurs implémentant un mode de réalisation de l’invention. Ce réseau utilise les mécanismes du TDM (pour « Time Division Multiplex » en anglais ou « multiplexage par répartition dans le temps » en français).

L’infrastructure du réseau est composée de commutateurs 100. Dans l’exemple de la figure 1, il y a sept commutateurs 100 appelés : « TDM Switch 1 » à « TDM Switch 7 ». À chaque commutateur est connecté au moins un lien réseau 102. Un commutateur peut éventuellement être connecté à un lien applicatif 103. Les commutateurs du réseau sont reliés entre eux par le lien réseau 102. Ce lien réseau est bidirectionnel et permet l’échange de données entre les commutateurs. Son fonctionnement sera détaillé en relation avec la figure 3.

Des applications 101 sont branchées sur ce réseau. Il existe deux types d’applications :

- des applications productrices 101-a, qui génèrent des données (dans l’exemple de la figure 1, il y a deux applications productrices 101-a appelées : « Appli TX1 » et « Appli TX2 »), et
- des applications consommatrices 101-b, qui consomment les données précédemment générées (dans l’exemple de la figure 1, il y a cinq applications consommatrices 101-a : trois appelées « Appli RX1 » (qui consomment les données de « Appli TX1 ») et deux appelées « Appli RX2 » (qui consomment les données de « Appli TX2 »)).

Il est à noter qu’une application dite productrice peut aussi recevoir des données (par exemple de type données de contrôle) et qu’une application consommatrice peut aussi émettre des données (par exemple de type contrôle). Le point d’accès au réseau d’une application est un commutateur du réseau. Le lien utilisé est le lien applicatif 103. Ce lien est bidirectionnel. Son fonctionnement sera détaillé en relation avec la figure 4.

La figure 2 décrit l’architecture, selon un mode de réalisation particulier de l’invention, d’un commutateur du réseau 100 de la figure 1 dans le cas de trois paires de ports « réseau » (chaque paire comprenant un port d’entrée réseau et un port de sortie réseau) et une paire de ports « applicatifs » (comprenant un port d’entrée applicatif et un port de sortie applicatif).

Chaque commutateur 200 peut être connecté à un autre commutateur via un lien réseau connecté à un module d’entrée réseau 203, formant un port d’entrée réseau. Ce lien comprend :

- un signal $F_{Réseau}$ représentatif de la fréquence du réseau ;
- un signal Data sur lequel sont véhiculées les données du réseau.

Chaque module d’entrée réseau 203 (port d’entrée réseau) est en charge de récupérer les données issues du réseau et de les envoyer vers une mémoire FIFO 205.

Dans l’exemple de la figure 2, le commutateur comprend donc trois modules d’entrée réseau 203 (formant respectivement un « port d’entrée réseau 1 », un « port d’entrée réseau 2 » et un « port d’entrée réseau 3 »), reliés chacun à une mémoire FIFO 205.

Chaque commutateur 200 peut aussi être connecté à une application productrice via un lien applicatif connecté à un module d’entrée applicatif 204, formant un port d’entrée applicatif. Ce lien comprend

- un signal $F_{Appli}$ représentatif de la fréquence de l’application
- un signal Data sur lequel sont véhiculées les données de l’application.

Le module d’entrée applicatif (port d’entrée applicatif) 204 est en charge de récupérer les données issues de l’application productrice, de les placer dans des canaux virtuels puis de les envoyer vers une FIFO 205.

Les mémoires FIFO 205 (il y en a quatre dans l’exemple de la figure 2) sont utilisées par l’invention pour gérer le délai à appliquer sur les données d’une entrée en fonction de la sortie vers laquelle elles doivent être routées. Les canaux virtuels venant des modules d’entrée 203 (port d’entrée réseau) et 204 (port d’entrée applicatif) sont stockés dans les mémoires FIFO respectives via un pointeur d’écriture et ce suivant le cadencement du réseau (définissant le cycle réseau). Puis, en fonction des différents délais, ces canaux virtuels seront lus par le module de routage 206, au travers de différents liens 209 représentant différents pointeurs de lecture associés chacun à un port de sortie (réseau ou applicatif) du commutateur.

Le module de routage 206 est en charge du routage de données des entrées vers les sorties en fonction par exemple :

- d’une information de routage (adresse, numéro de port…) contenue dans le canal virtuel ;

- de tables de routage contenues dans ce module : une table indiquant un routage d’une entrée vers une sortie, d’un canal virtuel d’une entrée vers un canal virtuel d’une sortie… ;

- d’une méthode de routage propre au commutateur : dans l’exemple traité ici, le routage se fait d’un port d’entrée vers tous les ports de sortie : les commutateurs fonctionnent alors suivant une méthode dite de diffusion (« broadcast » en anglais).

En fonction du port de sortie, le module de routage 206 va lire les données contenues dans les mémoires FIFO via le pointeur de lecture correspondant à cette sortie. Puis il enverra les données ainsi lues vers le module de sortie réseau (port de sortie réseau) 207 ou le module de sortie applicatif (port de sortie applicatif) 208 considéré.
Chaque module de sortie réseau (port de sortie réseau) 207 est en charge d’émettre les canaux virtuels à destination d’un autre commutateur sur le lien réseau.

Le module de sortie applicatif (port de sortie applicatif) 208 est en charge d’extraire les données applicatives contenues dans les canaux virtuels et de les envoyer vers une application consommatrice à la fréquence $F_{\text{Appli}}$.

Il est à noter que les ports d’entrée et de sortie (qu’ils soient de type réseau ou applicatif) sont appariés. Ainsi le « port d’entrée réseau 1 » et le « port de sortie réseau 1 » forment une paire de ports réseau permettant de connecter, via un lien réseau, le commutateur auquel ils appartiennent à un autre commutateur, et permettant ainsi une communication bidirectionnelle entre ces deux équipements.

Il en va de même pour le lien applicatif à une nuance près. Si une application productrice est connectée au lien applicatif, c’est l’application productrice qui générera la fréquence applicable à destination du commutateur et cette fréquence sera utilisée pour les deux sens de communication. A l’inverse, si une application consommatrice est connectée au commutateur, c’est ce dernier qui générera la fréquence $F_{\text{Appli}}$.

Un module de synchronisation 201 est en charge de synchroniser le fonctionnement des différents modules afin d’assurer que le routage se fasse de manière synchrone dans tout le réseau. Ce module fonctionnera de deux manières différentes :

- en mode Maître : c’est le mode du commutateur en charge d’imposer la fréquence du réseau. En utilisant son oscillateur local, il va générer la fréquence du réseau $F_{\text{Réseau}}$. Cette fréquence sera véhiculée sur tous les ports de sortie du commutateur ; elle permet de définir le cycle réseau (cadencement du réseau) ;

- en mode Esclave : c’est le mode des autres commutateurs du réseau. Le module de synchronisation choisira le port d’entrée sur lequel le nœud maître est connecté (directement ou au travers une chaîne de commutateurs) puis utilisera la fréquence réseau de ce port pour générer en interne une fréquence la plus proche (au moyen d’une boucle à verrouillage de phase (ou « PLL ») par exemple).
Enfin un module de gestion des délais 202 est le module contenant le cœur de l’invention. C’est lui qui va être en charge de déterminer selon l’invention les différents délais. Il reçoit donc les canaux virtuels (de contrôle) contenant les données nécessaires à la détermination de ces délais. Puis, une fois la détermination effectuée, il est en charge de placer les pointeurs de lecture des différentes mémoires FIFO conformément aux délais.

La figure 3 décrit les mécanismes TDM utilisés dans le réseau. Considérons un commutateur 100 avec trois liens réseau.

L’ensemble de la bande passante disponible délivrée par le système est découpé en canaux virtuels (ou VC, pour « Virtual Channel » en anglais) synchrones dont le débit utile est caractérisé par la fréquence de cadencement des canaux virtuels, par exemple 8KHz, ainsi que par la taille des échantillons, par exemple 48 bits. Ainsi un canal virtuel a un débit constant de 384 Kbps (kilobit par seconde). À titre d’exemple, l’un de ces canaux virtuels, comprenant m bits, est représenté et référencé 303 sur la figure 3.

La séquence complète comprenant l’ensemble des canaux virtuels disponibles constitue un cycle complet de traitement de données synchrone, aussi appelé cycle SDPC (pour « Synchronous Data Processing Cycle » en anglais) ou encore cycle réseau (référencé 304). Cette séquence est représentée par une succession (référencée 302) de canaux virtuels VC (numérotés 1 à n dans cet exemple).

La durée de ce cycle SDPC, aussi appelée période TDM, est égale à 125μs dans le cas d’une fréquence de cadencement de 8KHz des canaux virtuels. Cette durée est comprise entre deux impulsions (ou « pulse » en anglais) 301 du signal 300 qui représente la période P_{Réseau} (donc aussi la fréquence réseau F_{Réseau}).

Sur chaque lien « réseau » du commutateur, une trame TDM composée de canaux virtuels VC tels que décrits précédemment sera envoyée à chaque cycle SDPC. Les canaux virtuels VC d’un port de sortie de type « réseau » seront remplis par le contenu des canaux virtuels VC des autres ports d’entrée de type « réseau » ainsi que des données venant des applications, en fonction des délais selon l’invention et de la politique de routage implantée dans le commutateur.
Les commutateurs étant synchronisés entre eux (même fréquence réseau \( F_{\text{Réseau}} \)), tous les ports de sortie de tous les commutateurs émettront en même temps une trame TDM composée de canaux virtuels VC.

La **figure 4** décrit les mécanismes utilisés sur l’interface entre une application et le commutateur.

Une application est caractérisée par la fréquence à laquelle elle produit ou consomme des données. Le signal 400 représente cette fréquence.

Le signal 300 est la fréquence réseau \( F_{\text{Réseau}} \) tel que décrit ci-dessus, en relation avec la figure 3.

Dans le cadre d’une application productrice, à chaque impulsion du signal 400 est associée une donnée. Durant un cycle réseau (cycle SDPC) (entre 2 impulsions du signal 300), le commutateur va recevoir un certain nombre de données de l’application productrice. Il va stocker ces données dans des canaux virtuels VC qui auront été réservés à cette application. Comme on le voit sur la figure 4, la donnée 1 (référencée 401) venant de l’application sera reçue par le commutateur et placée dans le premier canal virtuel VC (405).

A l’inverse, dans le cas d’une application consommatrice, le commutateur générera la fréquence applicative 400 à partir de son horloge interne et enverra vers l’application les données extraites des canaux virtuels VC reçus et selon les délais tels que décrits dans cette invention au rythme de cette fréquence applicative.

L’algorithme de la **figure 5** décrit le fonctionnement général d’un mode de réalisation particulier du procédé selon l’invention.

Durant la phase d’initialisation du commutateur, celui-ci cherche tout d’abord à déterminer la plus longue chaîne de commutateurs branchée sur chacune de ses paires de ports réseau, à l’étape 500.

A l’étape 501, le commutateur utilise les résultats précédents pour déterminer la plus longue chaîne de commutateurs le traversant.

A l’étape 502, le commutateur échange la valeur déterminée à l’étape 501 avec les autres commutateurs du réseau afin de déterminer la plus longue chaîne de commutateurs du réseau.
Puis, à l’étape 503, le commutateur détermine les délais à appliquer sur les données reçues sur les ports d’entrée, en fonction des paramètres déterminés dans les étapes précédentes et des différents ports de sortie.

Enfin, à l’étape 504, le commutateur applique ces délais en paramétrant les mémoires FIFO de ses ports d’entrée (paramétrage relatif au pointeur de lecture associé à chaque port de sortie du commutateur).

Toutes ces étapes sont décrites en détail dans les paragraphes suivants.

La figure 6 présente une description détaillée d’un mode de réalisation particulier de l’étape 500 de la figure 5, de détermination de la plus longue chaîne de commutateurs sur chaque paire de ports d’un commutateur.

L’étape 600 correspond au point d’entrée de la procédure.

A l’étape 601, le processus initialise à 0 un tableau de variables N[P], P correspondant au nombre de paires de ports du commutateur. La variable N attachée à une paire de ports p, aussi notée N[p], correspondra, quand cette procédure sera terminée, au nombre de commutateurs composant la plus grande chaîne de commutateurs branchée sur cette paire de ports.

A l’étape 602, l’invention va vérifier si le commutateur n’a qu’une seule paire de ports réseau de branchée sur le réseau. Il s’agit là de déterminer si le commutateur est à une des extrémités du réseau.

Si ce n’est pas le cas, le processus passe à l’étape 604. Si le commutateur est bien situé à l’une des extrémités du réseau, le processus passe à l’étape 603.

A l’étape 603, le processus va envoyer un message d’un premier type (appelé ci-après message PING) avec comme paramètre la valeur 0 (zéro) : PING(0). Pour ce faire, le processus va par exemple utiliser un canal virtuel VC de contrôle connu de tous les commutateurs du réseau. Ce canal virtuel VC peut être alloué dynamiquement par le commutateur maître et ce dernier peut en avertir les autres. Le canal virtuel VC de contrôle peut aussi être fixe dans la trame et connu de tous les commutateurs.

A l’étape 604, une temporisation (timeout) est armée. Elle permet de fixer un temps maximal de cette phase de l’invention au-delà de laquelle il est estimé qu’elle est terminée.
A l’étape 605, l’invention attend soit la fin de la temporisation, soit la réception d’un message PING(x) envoyé par un autre commutateur du réseau.

A l’étape 606, un test est effectué pour savoir si la sortie de l’étape précédente est liée à la fin de la temporisation. Si c’est le cas le processus va à l’étape 611, fin de l’algorithme. Dans le cas contraire, il s’agit donc d’une réception d’un message PING(x). L’invention passe à l’étape 607.

A l’étape 607, l’invention reçoit le message PING(x) et mémorise le numéro p du port d’entrée réseau sur lequel elle a reçu ce PING ainsi que la valeur du paramètre x contenu dans le message.

A l’étape 608, pour le port d’entrée réseau considéré (numéro p), l’invention compare le paramètre x avec la variable N[p] qui a été précédemment mémorisée. Dans le cas où N[p] est inférieur à x+1, l’invention passe à l’étape 609. Dans le cas contraire, on passe directement à l’étape 610.

A l’étape 609, on a donc reçu sur un port d’entrée réseau p, dans un message PING(x), une valeur x telle que x+1 est supérieure à la valeur mémorisée dans N[p]. L’invention va donc remplacer la valeur de N[p] par x+1.

A l’étape 610, l’invention va envoyer le message PING(x+1) sur les ports de sortie réseau, à l’exception de celui qui est associé au port d’entrée réseau sur lequel a été reçu le message PING(x) qui a déclenché le processus (à l’étape 607).

Puis l’invention retourne à l’étape 604 afin de continuer à recevoir les messages PING éventuellement envoyés par d’autres commutateurs.

La figure 7 décrit un mode de réalisation particulier de l’étape 501 de la figure 5, c’est-à-dire un exemple de méthode utilisée pour la détermination de la plus longue chaîne de commutateurs passant par le commutateur exécutant l’invention. Le résultat de cette détermination sera stocké dans la variable LocalMaxPath.

Le point d’entrée dans cette méthode se fait à l’étape 700.

L’étape 701 montre le calcul utilisé pour la détermination de la variable LocalMaxPath à partir des valeurs N[P] précédemment déterminées :

\[
\text{LocalMaxPath} = \text{Max1}(N[\text{tous les ports}]) + \text{Max2}(N[\text{tous les ports}])
\]

Max1(N[tous les ports]) va retourner la plus grande valeur de N[P].
Max2(N[tous les ports]) va retourner la deuxième plus grande valeur de N[P].
Dans le cas d’un commutateur à une seule paire de ports réseau, Max2(N[tous les ports]) retournera 0.

Une fois déterminée la valeur de la variable LocalMaxPath, l’invention passe à l’étape 702 représentant la fin de la méthode.

La figure 8 décrit un mode de réalisation particulier de l’étape 502 de la figure 5, c’est-à-dire un exemple de méthode utilisée pour la détermination de la plus longue chaîne de commutateurs du réseau. Le résultat de cette détermination sera stocké dans la variable MaxPath. Elle aboutira à la détermination d’une variable M[P] (une par port de sortie du commutateur) qui sera utilisé plus tard.

Le point d’entrée dans cette méthode se fait à l’étape 800. A l’étape 801, la variable MaxPath est initialisée à 0 (zéro).

A l’étape 802, l’invention envoie sur tous les ports de sortie réseau du commutateur, dans le canal virtuel VC de contrôle, un message d’un second type (appelé ci-après message SEND_MAXPATH) avec en paramètre la valeur de la variable LocalMaxPath déterminée précédemment, c’est-à-dire envoi du message SEND_MAXPATH(LocalMaxPath).

A l’étape 803, une temporisation (timeout) est armée. Elle permet de fixer un temps maximal de cette phase de l’invention au-delà de laquelle il est estimé qu’elle est terminée.

A l’étape 804, l’invention attend soit la fin de la temporisation, soit la réception d’un message SEND_MAXPATH(x) envoyé par un autre commutateur du réseau.

A l’étape 805, un test est effectué pour savoir si la sortie de l’étape précédente est liée à la fin de la temporisation. Si c’est le cas, le processus va à l’étape 810. Dans le cas contraire, il s’agit donc d’une réception d’un message SEND_MAXPATH(x). L’invention passe à l’étape 806.

A l’étape 806, l’invention reçoit le message SEND_MAXPATH(x) et mémorise la valeur du paramètre x contenu dans le message.
A l’étape 807, l’invention compare le paramètre x avec le paramètre MaxPath. Dans le cas où MaxPath est inférieur à x, l’invention passe à l’étape 808. Dans le cas contraire, on passe directement à l’étape 809.

A l’étape 808, on a donc reçu dans un message SEND_MAXPATH (x) une valeur x supérieure à celle mémorisée dans MaxPath. L’invention va donc remplacer la valeur de MaxPath par x.

A l’étape 809, l’invention va envoyer le message SEND_MAXPATH (x) sur les ports de sortie à l’exception de celui qui est associé au port d’entrée réseau sur lequel a été reçu le message SEND_MAXPATH (x) qui a déclenché le processus (à l’étape 806).

Puis l’invention retourne à l’étape 803 afin de continuer à recevoir les messages SEND_MAXPATH éventuellement envoyés par d’autres commutateurs.

A l’étape 810, la réception des messages SEND_MAXPATH est considérée comme terminée. Le calcul des variables M[P] va commencer et va dépendre de la nature de chaque port de sortie du commutateur (« réseau » ou « applicatif »).

A l’étape 811, l’invention va choisir, suivant la nature du port de sortie, l’un des deux chemins suivants : soit l’étape 812 si le port de sortie est de type « applicatif », soit l’étape 813 s’il est de type « réseau ».

A l’étape 812, M[p] prend la valeur MaxPath pour p représentant le numéro du port de sortie applicatif considéré.

A l’étape 813, M[p] prend la valeur maximale des variables N des paires de ports réseau autres que celle à laquelle appartient le port de sortie réseau p actuellement analysé.

L’étape 814 permet de vérifier s’il y a d’autres ports de sortie à analyser. Dans le cas positif, on retourne à l’étape 811. Sinon, l’analyse est terminée et le processus prend fin à l’étape 815.

La figure 9 détaille un mode de réalisation particulier de l’étape 503 de la figure 5, c’est-à-dire un exemple de processus de détermination d’un délai pour chaque couple de ports comprenant un port d’entrée et un port de sortie du commutateur n’appartenant pas une même paire de ports réseau. Chaque délai ainsi déterminé est un multiple du cycle SDPC.
Il est important de noter que le terme « couple de ports » est différent du terme « paire de ports ».

En effet, comme précédemment mentionné, une paire de ports est un ensemble d’un port de sortie et un port d’entrée d’un commutateur qui correspondent :

- soit à un même lien réseau (respectivement émission et réception de données du lien réseau), dans le cas d’une paire de ports réseau ;
- soit à un même lien applicatif (respectivement émission et réception de données du lien applicatif), dans le cas d’une paire de ports applicatifs.

En revanche, un couple de ports est un ensemble d’un port de sortie et un port d’entrée d’un commutateur, selon l’une quelconque de toutes les combinaisons possibles d’un des ports de sortie (qu’il soit de type réseau ou applicatif) et un des ports d’entrée (qu’il soit de type réseau ou applicatif) entre lesquels une transmission de données est possible, à l’exception de la combinaison d’un port de sortie réseau et un port d’entrée réseau correspondant à un même lien réseau. En effet, il est considéré qu’il n’y a pas de transmission de données entre le port d’entrée réseau et le port de sortie réseau d’une même paire de ports réseau (c’est-à-dire correspondant à un même lien réseau).

Le point d’entrée dans cette méthode se fait à l’étape 900.

À l’étape 901, l’invention va passer en revue les ports d’entrée un par un.

À l’étape 902, pour un port d’entrée choisi à l’étape précédente, l’invention va passer en revue les ports de sortie un par un.

Il est à noter que pour un port d’entrée de type « réseau », l’invention ne traitera pas le port de sortie correspondant. En revanche, pour un port d’entrée de type « applicatif », l’invention traitera également le port de sortie correspondant.

À l’étape 903, pour le port d’entrée choisi E et le port de sortie choisi S, l’invention calcule le délai selon la formule suivante :

\[ \text{Délai}[E,S] = M[S] - N[E] \]

À l’étape 904, l’invention va vérifier s’il y a un autre port de sortie à considérer. Si c’est le cas, on retourne à l’étape 903. Sinon on passe à l’étape 905.
A l’étape 905, l’invention va vérifier s’il y a un autre port d’entrée à considérer. Si c’est le cas, on retourne à l’étape 902. Sinon on passe à l’étape 906 qui correspond à la fin du processus.

La **figure 10** détaille un mode de réalisation particulier de l’étape 504 de la figure 5 (méthode d’application des délais), c'est-à-dire un exemple d’application des délais sur les données venant des ports d’entrée et transférées vers les ports de sortie d’un commutateur. Une gestion des délais par mémoires FIFO est décrite ci-dessous.

La FIFO 1000 est l’élément que l’on retrouve référencé 205 sur la figure 2. Il s’agit d’une mémoire à laquelle le port d’entrée accèdera en écriture à travers un pointeur d’écriture 1002 et à laquelle le module de routage accèdera en lecture à travers une série de pointeurs de lecture 1003, 1004, 1005, un par port de sortie (dans le cas de cette figure, il y a trois ports de sortie).

La mémoire est organisée en blocs 1001 représentant chacun un ensemble de canaux virtuels VC reçus durant un cycle SDPC. Un bloc 1001 a une taille correspondant au nombre maximal de canaux virtuels VC transmis pendant un cycle SDPC que multiplie le nombre d’octets nécessaire à stocker les informations d’un canal virtuel VC (identifiant du VC, données utiles du VC…).

Il est à noter que l’on pourrait utiliser une gestion plus dynamique des canaux virtuels VC consistant à stocker au fur et à mesure de leur arrivée les canaux virtuels VC (et les éventuelles informations associées à ces canaux virtuels VCs) afin d’économiser de la mémoire.

La mémoire totale allouée correspond à la taille d’un bloc que multiplie le délai maximum précédemment déterminé plus un bloc pour la réception.

Dans le cas d’une gestion plus dynamique, aucune mémoire ne sera allouée à l’avance mais cette allocation se fera au gré des réceptions.

Dans le cas d’une gestion non dynamique de la mémoire FIFO, la gestion des délais se fait donc par un positionnement du pointeur d’écriture sur le premier bloc 1001 et des pointeurs de lecture sur les blocs suivants en fonction du délai à appliquer. Puis à chaque cycle SDPC et avant l’écriture, chaque pointeur sera décrémenté d’un bloc. Si
avant le décès, le pointeur est positionné sur le premier bloc, il sera ensuite positionné sur le dernier bloc (buffer tournant).

Dans le cas d’une gestion plus dynamique, une liste chaînée de bloc sera gérée. A l’initialisation ou après chaque changement de topologie du réseau (ajout, retrait ou déplacement d’un commutateur), tous les pointeurs seront placés sur le premier élément (après la première réception). Après une attente d’un nombre de cycles SDPC correspondant au délai associé à chaque pointeur, la lecture sera validée et à chaque cycle, le pointeur sera décalé vers l’élément suivant. Un élément qui aura été traité par le pointeur le plus « retardé » sera ensuite libéré (« désalloué »).

La figure 11 présente un exemple de règles de désignation, sur une représentation graphique, des trois paires de ports réseau et de la paire de ports applicatif de chaque commutateur 100 de la figure 1.

Selon ces règles (arbitraires), on appelle :

- « paire de ports réseau 1 » la paire de ports réseau (référencée 1101) représentée graphiquement à gauche du commutateur 100 ;
- « paire de ports réseau 2 » la paire de ports réseau (référencée 1102) représentée graphiquement au-dessus du commutateur 100 ;
- « paire de ports réseau 3 » la paire de ports réseau (référencée 1103) représentée graphiquement à droite du commutateur 100 ;
- « paire de ports applicatif » la paire de ports applicatifs (référencée 1104) représentée graphiquement au-dessus ou au-dessous du commutateur 100.

La figure 12 présente une table donnant, pour chacun des commutateurs (colonne référencée 1201), les résultats des étapes 500 à 502 de la figure 5 appliquées au réseau de la figure 1, et avec les règles de désignation de la figure 11 :

- en colonnes référencées 1202 à 1205, les résultats de l’étape 500 (détermination des variables N[P]) ;
- en colonne référencée 1206, le résultat de l’étape 501 (détermination de la variable LocalMaxPath) ; et
- en colonne référencée 1207 à 1211, les résultats de l’étape 502 (détermination de la variable MaxPath et des variables M[P]) de la figure 5.
appliquées au réseau de la figure 1, et avec les règles de désignation de la figure 11.

On notera que les désignations « commutateur 1 » à « commutateur 7 » en colonne référencée 1201 de la figure 12 correspondent respectivement aux désignations « TDM Switch 1 » à « TDM Switch 7 » sur la figure 1.

On présente ci-dessous, à titre d’exemple, le résultat du calcul de délais (résultat de l’étape 503 de la figure 5) pour les commutateurs 1, 2, 3 et 7 (désignés par « TDM Switch 1 », « TDM Switch 2 », « TDM Switch 3 » et « TDM Switch 7 » respectivement sur la figure 1). Ces délais sont des nombres de cycles réseau.

### Combinateur 1 :

<table>
<thead>
<tr>
<th>Port d’Entrée</th>
<th>Port Résseau 1</th>
<th>Port Résseau 2</th>
<th>Port Résseau 3</th>
<th>Port Applicatif</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port Résseau 1</td>
<td>5</td>
<td>0</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>Port Résseau 2</td>
<td>5</td>
<td>0</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>Port Résseau 3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Port Applicatif</td>
<td>5</td>
<td>5</td>
<td>0</td>
<td>5</td>
</tr>
</tbody>
</table>

### Combinateur 2 :

<table>
<thead>
<tr>
<th>Port d’Entrée</th>
<th>Port Résseau 1</th>
<th>Port Résseau 2</th>
<th>Port Résseau 3</th>
<th>Port Applicatif</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port Résseau 1</td>
<td>3</td>
<td>0</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>Port Résseau 2</td>
<td>4</td>
<td>1</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>Port Résseau 3</td>
<td>0</td>
<td>4</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Port Applicatif</td>
<td>4</td>
<td>4</td>
<td>1</td>
<td>5</td>
</tr>
</tbody>
</table>

### Combinateur 3 :

<table>
<thead>
<tr>
<th>Port d’Entrée</th>
<th>Port Résseau 1</th>
<th>Port Résseau 2</th>
<th>Port Résseau 3</th>
<th>Port Applicatif</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port Résseau 1</td>
<td>0</td>
<td>1</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>Port Résseau 2</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>Port Résseau 3</td>
<td>2</td>
<td>1</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>Port Applicatif</td>
<td>3</td>
<td>2</td>
<td>3</td>
<td>5</td>
</tr>
</tbody>
</table>

### Combinateur 7 :

<table>
<thead>
<tr>
<th>Port d’Entrée</th>
<th>Port Résseau 1</th>
<th>Port Résseau 2</th>
<th>Port Résseau 3</th>
<th>Port Applicatif</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port Résseau 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Port Résseau 2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Port Résseau 3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Port Applicatif</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Port d'Entrée</td>
<td>Port Réseau 1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>--------------</td>
<td>--------------</td>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>Port Réseau 2</td>
<td>0</td>
<td>4</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>Port Réseau 3</td>
<td>0</td>
<td>4</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>Port Applicatif</td>
<td>0</td>
<td>4</td>
<td>4</td>
<td>5</td>
</tr>
</tbody>
</table>

Prenons l’exemple de la figure 1. En particulier, nous allons détailler le cas où les applications productrices TX1 et TX2 envoient des données qui doivent transiter vers le commutateur 4.

A un cycle SDPC N, les applications TX1 et TX 2 envoient en même temps un certain nombre d’échantillons vers respectivement les commutateurs 3 et 1.

Pour le commutateur 1, on voit qu’aucun retard n’est appliqué sur les données venant de l’application TX2 par le port applicatif et allant sur le port 3 vers le commutateur 2. Donc les données venant de TX2 transitent et sont reçues par le commutateur 2 au cycle réseau N+1.

Pour le commutateur 2, on voit aussi qu’aucun retard n’est appliqué pour les données (de TX2) arrivant par le port 1 et partant par le port 3 vers le commutateur 3. Donc les données (de TX2) transitent et sont reçues par le commutateur 3 au cycle réseau N+2.

Pour le commutateur 7, on voit aussi qu’aucun retard n’est appliqué pour les données arrivant par le port applicatif (il s’agit ici d’une application réceptrice RX2 qui enverrait des données de contrôle au cycle N) et partant par le port 1 vers le commutateur 3. Donc les données (de RX2) transitent et sont reçues par le commutateur 3 au cycle réseau N+1.

Pour le commutateur 3, on voit qu’un retard de 2 cycles est appliqué sur les données venant de l’application TX1 par le port applicatif à destination du port 2. On voit ensuite qu’un retard d’1 cycle est appliqué pour les données (de RX2) venant du commutateur 7 par le port 3 à destination du port 2. Puis on voit qu’aucun retard n’est appliqué pour les données (de TX2) arrivant par le port 1 à destination du port 2.

On voit donc que les données reçues par le commutateur 3 sur le port réseau 1, le port réseau 3 et le port applicatif (issues respectivement de TX2, RX2 et TX1) ont globalement un délai identique cumulé de N+2. Elles transiteront donc ensemble vers le commutateur 4 par le port 2 au cycle N+3.
On constate donc que dans une trame TDM durant un cycle M on ne trouve que des données générées au même cycle réseau C (C<M).

On notera que l’invention ne se limite pas à une implantation purement matérielle mais qu’elle peut aussi être mise en œuvre sous la forme d’une séquence d’instructions d’un programme informatique ou toute forme mixant une partie matérielle et une partie logicielle. Dans le cas où l’invention est implantée partiellement ou totalement sous forme logicielle, la séquence d’instructions correspondante pourra être stockée dans un moyen de stockage amovible (tel que par exemple une disquette, un CD-ROM ou un DVD-ROM) ou non, ce moyen de stockage étant lisible partiellement ou totalement par un ordinateur ou un microprocesseur.
REVENDICATIONS

1. Procédé de gestion de délais dans un commutateur d’un réseau de communication de type synchrone comprenant une pluralité de commutateurs, ledit commutateur comprenant au moins une paire de ports réseau comprenant un port d’entrée réseau et un port de sortie réseau, permettant une connexion avec un autre commutateur via un lien réseau, et au moins une paire de ports applicatifs comprenant un port d’entrée applicatif et un port de sortie applicatif, permettant une connexion avec une application via un lien applicatif, ledit commutateur recevant sur chacun des ports d’entrée des données à transférer sur un ou plusieurs des ports de sorties, lesdites données étant reçues sur les ports d’entrée et émises sur les ports de sortie selon un rythme régulier dit cycle réseau, procédé caractérisé en ce qu’il comprend les étapes suivantes, effectuées par ledit commutateur :
   - pour chacune des paires de ports, réseau ou applicatifs, du commutateur, détermination (500) de la plus longue chaîne de commutateurs branchée sur ladite paire de ports ;
   - détermination (501) de la plus longue chaîne de commutateurs passant par le commutateur, à partir des plus longues chaînes de commutateurs déterminées pour les différentes paires de ports du commutateur ;
   - détermination (502) de la plus longue chaîne de commutateurs du réseau, à partir de la plus longue chaîne de commutateurs passant par le commutateur et des plus longues chaînes de commutateurs passant chacune par un autre commutateur ;
   - détermination (503) d’un délai pour chacun des couples de ports comprenant un port d’entrée et un port de sortie du commutateur n’appartenant pas une même paire de ports réseau, à partir des plus longues chaînes de commutateurs branchées chacune sur l’une des paires de ports du commutateur et de ladite plus longue chaîne de commutateurs du réseau.

2. Procédé selon la revendication 1, caractérisé en ce que, dans ladite étape de détermination (503) d’un délai pour chacun des couples de ports, on utilise la formule suivante : Délai[E,S] = M[S] - N[E], où
   - Délai[E,S] est un délai, exprimé en nombre de cycles réseau, pour le couple de ports comprenant le port d’entrée E et le port de sortie S ;
- \( N[E] \) est le nombre de commutateurs composant la plus longue chaîne de comutateurs branchée sur la paire de ports, réseau ou applicatifs, à laquelle appartient ledit port d’entrée \( E \);

- \( M[S] \) est :

  * le nombre de commutateurs composant ladite plus longue chaîne de commutateurs du réseau, si le port de sortie \( S \) est un port de sortie applicatif ; ou

  * le nombre de commutateurs composant la plus longue des plus longues chaînes de commutateurs branchées chacune sur l’une des paires de ports réseau du commutateur autre que la paire de ports réseau à laquelle appartient ledit port de sortie \( S \), si le port de sortie \( S \) est un port de sortie réseau.

3. Procédé selon l’une quelconque des revendications 1 et 2, caractérisé en ce que ladite étape de détermination (500) de la plus longue chaîne de commutateurs branchée sur chaque paire de ports comprend les étapes suivantes :

- pour chaque paire de ports \( p \) du commutateur, initialisation à zéro d’une variable \( N[p] \) correspondant au nombre de commutateurs composant la plus longue chaîne de commutateurs branchée sur ladite paire de ports \( p \);

- première détection permettant de détecter si une seule paire de ports réseau dudit commutateur est branchée au réseau ;

- en cas de première détection positive, envoi sur le port de sortie de ladite seule paire de ports réseau branchée au réseau, d’un message d’un premier type contenant un paramètre à incrémenter \( k1 \) possédant la valeur zéro ;

- deuxième détection, permettant de détecter la réception, sur un port d’entrée réseau du commutateur, d’un message du premier type contenant un paramètre à incrémenter \( k1 \) possédant une valeur \( x \) ;

- à chaque deuxième détection positive sur un port d’entrée réseau donné \( p \):

  * extraction de la valeur \( x \) du paramètre à incrémenter \( k1 \) contenu dans le message du premier type reçu ;

  * si la valeur courante de la variable \( N[p] \) associée à la paire de ports réseau \( p \) comprenant ledit port d’entrée réseau donné, est inférieure à la valeur \( x+1 \),
alors remplacement de la valeur courante de la variable N[p] par la valeur x+1 ;
* envoi d'un message du premier type contenant un paramètre à incrémenter k1 possédant la valeur x+1, sur les ports de sortie réseau du commutateur à l'exception du port de sortie réseau compris dans la même paire de ports réseau que l'edit port d'entrée réseau donné p.

4. Procédé selon la revendication 3, caractérisé en ce que chaque étape d'envoi d'un message du premier type est suivie d'une étape de déclenchement d'une première temporisation d'une première durée, et en ce que ladite étape de détermination (500) de la plus longue chaîne de commutateurs branchée sur chaque paire de ports est terminée quand ladite première temporisation est expirée sans qu'un nouveau message du premier type ait été reçu.

5. Procédé selon l'une quelconque des revendications 1 à 4, caractérisé en ce que ladite étape de détermination (501) de la plus longue chaîne de commutateurs passant par le commutateur comprend les étapes suivantes :
- détermination des deux plus grandes valeurs parmi lesdites plus longues chaînes de commutateurs branchées chacune sur l'une des paires de ports réseau du commutateur ;
- addition desdites deux plus grandes valeurs, pour obtenir le nombre de commutateurs composant ladite plus longue chaîne de commutateurs passant par le commutateur.

6. Procédé selon l'une quelconque des revendications 1 à 5, caractérisé en ce que ladite étape de détermination (502) de la plus longue chaîne de commutateurs du réseau comprend les étapes suivantes :
- initialisation à zéro d'une variable MaxPath correspondant au nombre de commutateurs composant la plus longue chaîne de commutateurs du réseau ;
- envoi sur tous les ports de sortie réseau du commutateur, d'un message d'un second type contenant un paramètre k2 possédant comme valeur le nombre de commutateurs composant ladite plus longue chaîne de commutateurs passant par le commutateur ;
- détection, permettant de détecter la réception, sur un port d’entrée réseau du commutateur, d’un message du second type contenant un paramètre k2 possédant une valeur x ;
- à chaque détection positive sur un port d’entrée réseau donné p :
  * extraction de la valeur x du paramètre k2 contenu dans le message du second type ;
  * si la valeur courante de la variable MaxPath est inférieure à la valeur x, alors remplacement de la valeur courante de la variable MaxPath par la valeur x ;
  * envoi d’un message du second type contenant un paramètre k2 possédant la valeur x, sur les ports de sortie réseau du commutateur à l’exception du port de sortie réseau compris dans la même paire de ports réseau que ledit port d’entrée réseau donné p.

7. Procédé selon la revendication 6, caractérisé en ce que chaque étape d’envoi d’un message du second type est suivie d’une étape de déclenchement d’une seconde temporisation d’une seconde durée, et en ce que ladite étape de détermination (502) de la plus longue chaîne de commutateurs du réseau est terminée quand ladite seconde temporisation est expirée sans qu’un nouveau message du second type ait été reçu.

8. Produit programme d'ordinateur téléchargeable depuis un réseau de communication et/ou enregistré sur un support lisible par ordinateur et/ou exécutable par un processeur, caractérisé en ce qu’il comprend des instructions de code de programme pour la mise en œuvre du procédé selon au moins une des revendications 1 à 7, lorsque ledit programme est exécuté sur un ordinateur.

9. Moyen de stockage, éventuellement totalement ou partiellement amovible, lisible par un ordinateur, stockant un jeu d’instructions exécutables par ledit ordinateur pour mettre en œuvre le procédé selon au moins une des revendications 1 à 7.

10. Commutateur d’un réseau de communication de type synchrone comprenant une pluralité de commutateurs, ledit commutateur comprenant au moins une paire de ports réseau comprenant un port d’entrée réseau et un port de sortie réseau, permettant une connexion avec un autre commutateur via un lien réseau, et au moins une paire de ports applicatifs comprenant un port d’entrée applicatif et un port de sortie applicatif, permettant une connexion avec une application via un lien applicatif, ledit commutateur
recevant sur chacun des ports d’entrée des données à transférer sur un ou plusieurs des ports de sorties, lesdites données étant reçues sur les ports d’entrée et émises sur les ports de sortie selon un rythme régulier dit cycle réseau, commutateur caractérisé en ce qu’il comprend :

5 - des moyens de détermination, pour chacune des paires de ports, réseau ou applicatifs, du commutateur, de la plus longue chaîne de commutateurs branchée sur ladite paire de ports ;

- des moyens de détermination de la plus longue chaîne de commutateurs passant par le commutateur, à partir des plus longues chaînes de commutateurs déterminées pour les différentes paires de ports du commutateur ;

- des moyens de détermination de la plus longue chaîne de commutateurs du réseau, à partir de la plus longue chaîne de commutateurs passant par le commutateur et des plus longues chaînes de commutateurs passant chacune par un autre commutateur ;

10 - des moyens de détermination d’un délai pour chacun des couples de ports comprenant un port d’entrée et un port de sortie du commutateur n’appartenant pas une même paire de ports réseau, à partir des plus longues chaînes de commutateurs branchées chacune sur l’une des paires de ports du commutateur et de ladite plus longue chaîne de commutateurs du réseau.

11. Commutateur selon la revendication 10, caractérisé en ce que lesdits moyens de détermination d’un délai pour chacun des couples de ports utilisent la formule suivante :

Délai[E,S] = M[S] - N[E], où

- Délai[E,S] est un délai, exprimé en nombre de cycles réseau, pour le couple de ports comprenant le port d’entrée E et le port de sortie S ;

- N[E] est le nombre de commutateurs composant la plus longue chaîne de commutateurs branchée sur la paire de ports, réseau ou applicatifs, à laquelle appartient ledit port d’entrée E ;

- M[S] est :

* le nombre de commutateurs composant ladite plus longue chaîne de commutateurs du réseau, si le port de sortie S est un port de sortie applicatif ; ou
le nombre de commutateurs composant la plus longue des plus longues chaînes de commutateurs branchées chacune sur l’une des paires de ports réseau du commutateur autre que la paire de ports réseau à laquelle appartient ledit port de sortie S, si le port de sortie S est un port de sortie réseau.

12. Commutateur selon l’une quelconque des revendications 10 et 11, caractérisé en ce que lesdits moyens de détermination de la plus longue chaîne de commutateurs branchée sur chaque paire de ports comprennent :

- des moyens d’initialisation à zéro, pour chaque paire de ports p du commutateur, d’une variable N[p] correspondant au nombre de commutateurs composant la plus longue chaîne de commutateurs branchée sur ladite paire de ports p ;

- des premiers moyens de détection permettant de détecter si une seule paire de ports réseau dudit commutateur est branchée au réseau ;

- des moyens d’envoi, activés en cas de détection positive par lesdits premiers moyens de détection, permettant d’envoyer sur le port de sortie de ladite seule paire de ports réseau branchée au réseau, un message d’un premier type contenant un paramètre à incrémenter k1 possédant la valeur zéro ;

- des deuxième moyens de détection, permettant de détecter la réception, sur un port d’entrée réseau du commutateur, d’un message du premier type contenant un paramètre à incrémenter k1 possédant une valeur x ;

- les moyens suivants, activés à chaque détection positive par lesdits deuxième moyens de détection sur un port d’entrée réseau donné p :

  * des moyens d’extraction de la valeur x du paramètre à incrémenter k1 contenu dans le message du premier type reçu ;

  * des moyens de remplacement de la valeur courante de la variable N[p] par la valeur x+1, activés si la valeur courante de la variable N[p] associée à la paire de ports réseau p comprenant ledit port d’entrée réseau donné, est inférieure à la valeur x+1 ;

  * des moyens d’envoi d’un message du premier type contenant un paramètre à incrémenter k1 possédant la valeur x+1, sur les ports de sortie réseau du
commutateur à l’exception du port de sortie réseau compris dans la même paire de ports réseau que ledit port d’entrée réseau donné p.

13. Commutateur selon la revendication 12, caractérisé en ce qu’il comprend des moyens de déclenchement d’une première temporisation d’une première durée, et en ce que lesdits moyens de détermination de la plus longue chaîne de commutateurs branchée sur chaque paire de ports sont désactivés quand ladite première temporisation est expirée sans qu’un nouveau message du premier type ait été reçu.

14. Commutateur selon l’une quelconque des revendications 10 à 13, caractérisé en ce que lesdits moyens de détermination de la plus longue chaîne de commutateurs passant par le commutateur comprennent :

- des moyens de détermination des deux plus grandes valeurs parmi lesdites plus longues chaînes de commutateurs branchées chacune sur l’une des paires de ports réseau du commutateur ;

- des moyens d’addition desdites deux plus grandes valeurs, pour obtenir le nombre de commutateurs composant ladite plus longue chaîne de commutateurs passant par le commutateur.

15. Commutateur selon l’une quelconque des revendications 10 à 14, caractérisé en ce que lesdits moyens de détermination de la plus longue chaîne de commutateurs du réseau comprennent :

- des moyens d’initialisation à zéro d’une variable MaxPath correspondant au nombre de commutateurs composant la plus longue chaîne de commutateurs du réseau ;

- des moyens d’envoi sur tous les ports de sortie réseau du commutateur, d’un message d’un second type contenant un paramètre k2 possédant comme valeur le nombre de commutateurs composant ladite plus longue chaîne de commutateurs passant par le commutateur ;

- des moyens détection, permettant de détecter la réception, sur un port d’entrée réseau du commutateur, d’un message du second type contenant un paramètre k2 possédant une valeur x ;

- les moyens suivants, activés à chaque détection positive sur un port d’entrée réseau donné p ;
* des moyens d’extraction de la valeur x du paramètre k2 contenu dans le message du second type ;
* des moyens de remplacement de la valeur courante de la variable MaxPath par la valeur x, activés si la valeur courante de la variable MaxPath est inférieure à la valeur x ;
* des moyens d’envoi d’un message du second type contenant un paramètre k2 possédant la valeur x, sur les ports de sortie réseau du commutateur à l’exception du port de sortie réseau compris dans la même paire de ports réseau que l’édit port d’entrée réseau donné p.

16. Commutateur selon la revendication 15, caractérisé en ce qu’il comprend des moyens de déclenchement d’une seconde temporisation d’une seconde durée, et en ce que lesdits moyens de détermination de la plus longue chaîne de commutateurs du réseau sont désactivés quand ladite seconde temporisation est expirée sans qu’un nouveau message du second type ait été reçu.
Figure 1
Figure 4
Détermination de la plus longue chaîne de commutateurs pour chaque paire de ports du commutateur

Détermination de la plus longue chaîne de commutateurs passant par le commutateur

Détermination de la plus longue chaîne de commutateurs du réseau

Détermination des délais à appliquer sur chaque port d'entrée vers chaque port de sortie

Application de délais en vue du transfert de données des ports d'entrée vers les ports de sortie
Figure 6
Figure 7

LocalMaxPath = Max1(N[tous les ports]) + Max2(N[tous les ports])
Figure 8

Init

MaxPath = 0

Envoi SEND_MAXPATH(LocalMaxPath) sur les ports de sortie réseau

Timeout = SetupOK

Attente de réception d'un SEND_MAXPATH ou de la fin du timeout

Send_MAXPATH(x) reçu sur le port d'entrée p

MaxPath < x ?

MaxPath = x

Transfert SEND_MAXPATH(x) sur les autres ports de sortie

Analyse du 1er port de sortie P

P = Port de sortie applicatif ?

M[P] = MaxPath

M[P] = Max(N'autres)

Fin
Figure 9
Figure 10
Figure 11
<table>
<thead>
<tr>
<th>Commutateur</th>
<th>N[P]</th>
<th>Local Max Path</th>
<th>M[P]</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>7</td>
<td>0</td>
<td>5</td>
<td>5</td>
</tr>
</tbody>
</table>

Etape 500

<table>
<thead>
<tr>
<th>Paire de Ports Applicatifs</th>
<th>Paire de Ports Réseau 1</th>
<th>Paire de Ports Réseau 2</th>
<th>Paire de Ports Réseau 3</th>
<th>Max Path</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Etape 501

<table>
<thead>
<tr>
<th>Paire de Ports Applicatifs</th>
<th>Max Path</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>2</td>
<td>5</td>
</tr>
<tr>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>6</td>
<td>5</td>
</tr>
<tr>
<td>7</td>
<td>5</td>
</tr>
</tbody>
</table>

Etape 502

<table>
<thead>
<tr>
<th>Paire de Ports Applicatifs</th>
<th>Max Path</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>2</td>
<td>5</td>
</tr>
<tr>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>6</td>
<td>5</td>
</tr>
<tr>
<td>7</td>
<td>5</td>
</tr>
</tbody>
</table>

Figure 12
<table>
<thead>
<tr>
<th>Catégorie</th>
<th>Citation du document avec indication, en cas de besoin, des parties pertinentes</th>
<th>Revendication(s) concernée(s)</th>
<th>Classement attribué à l'invention par l'INPI</th>
</tr>
</thead>
</table>

**DOMAINES TECHNIQUES**

**RECHERCHES (IPC)**

- H04J
ANNEXE AU RAPPORT DE RECHERCHE PRÉLIMINAIRE
RELATIF A LA DEMANDE DE BREVET FRANÇAIS NO. FR 0760103 FA 701749


<table>
<thead>
<tr>
<th>Document brevet cité au rapport de recherche</th>
<th>Date de publication</th>
<th>Membre(s) de la famille de brevet(s)</th>
<th>Date de publication</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>US 2007153774 A1</td>
<td>05-07-2007</td>
</tr>
<tr>
<td>US 2006222018 A1</td>
<td>05-10-2006</td>
<td>AUCUN</td>
<td></td>
</tr>
<tr>
<td>EP 1289211 A</td>
<td>05-03-2003</td>
<td>DE 60201873 D1</td>
<td>16-12-2004</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DE 60201873 T2</td>
<td>20-10-2005</td>
</tr>
<tr>
<td></td>
<td></td>
<td>JP 3647429 B2</td>
<td>11-05-2005</td>
</tr>
<tr>
<td></td>
<td></td>
<td>JP 2003198584 A</td>
<td>11-07-2003</td>
</tr>
<tr>
<td></td>
<td></td>
<td>US 2003053466 A1</td>
<td>20-03-2003</td>
</tr>
</tbody>
</table>

Pour tout renseignement concernant cette annexe : voir Journal Officiel de l'Office européen des brevets, No.12/82