personnes arrivant ensemble devant une porte marquent une pause avant de décider qui passera en premier. Dans ces situations, l'ordre de passage n'a pas d'importance, la seule difficulté consiste à prendre la décision.

Le rôle de l'*Arbitre* est précisément de prendre cette décision. Un *Arbitre* a deux états stables correspondant aux deux choix. On peut se représenter ces deux états, l'un comme la France, l'autre comme l'Espagne. Chaque requête exprimée auprès d'un *Arbitre* pousse le circuit vers l'un ou l'autre de ces états stables, comme un grêlon, tombé au sommet des Pyrénées, pourrait rouler aussi bien vers la France que vers l'Espagne. Entre les deux états stables, il doit néanmoins y avoir une ligne métastable, équivalente à la ligne de crête des Pyrénées. Si le grêlon tombe précisément sur cette ligne de partage, il peut rester en équilibre quelques instants sur la crête avant de basculer vers l'Espagne ou

Processeurs asynchrones et relativité

Les questions soulevées par I. Sutherland et J. Ebergen, *a priori* de nature technologique, sont aussi le reflet de questions fondamentales sur le temps. Pour connaître le temps, il ne suffit malheureusement pas de le mesurer. La difficulté rencontrée pour synchroniser tous les calculs au sein d'une puce électronique résulte aussi de ce fait : le temps n'existe pas sous la forme d'une grandeur physique unique qu'il suffirait de mesurer à l'endroit où l'on se trouve. C'est cette constatation qui a conduit à l'introduction de la théorie de la relativité. Certes, le temps est obtenu à l'aide des indications des horloges, mais ce n'est pas suffisant. Comme le dit Einstein en 1905 : «Il nous faut garder à l'esprit que tous les jugements dans lesquels le temps joue un rôle sont toujours des événements simultanés. Lorsque, par exemple, je dis : «Tel train arrive ici à 7 heures», cela signifie à peu près : «Le passage de la petite aiguille de ma montre sur le 7 et l'arrivée du train sont des événements simultanés.» Einstein ajoute plus loin : «Cette définition ne suffit cependant plus dès lors qu'il s'agit de relier temporellement des séries d'événements qui se produisent à des endroits différents.» En fait, pour disposer d'un temps coordonné en des lieux différents, il faut synchroniser les horloges qui s'y trouvent.

Pour ce faire, il suffit d'échanger des signaux. Malheureusement, il existe une vitesse maximale à laquelle peuvent se propager toutes formes de signaux, fixée par la vitesse de la lumière dans le vide, de l'ordre de 300 000 kilomètres par seconde. Il en résulte que la simultanéité de deux événements se produisant en des lieux différents ne peut plus être définie de façon absolue, mais prend un caractère conventionnel. De ce point de vue, les concepteurs des microprocesseurs actuels sont confrontés au même problème que les ingénieurs de la fin du XIX^{e} siècle : ils voulaient accorder les horaires des trains, mais se heurtaient à la difficulté de synchroniser les horloges des gares sur un territoire large de plusieurs centaines de kilomètres. Einstein, alors employé du bureau des brevets à Berne, voyait passer des demandes de brevets pour des systèmes de synchronisation électromagnétique des horloges. La théorie de la relativité qui en a résulté permet aujourd'hui de disposer, sur toute la Terre, d'une référence de temps, et ainsi de systèmes de repérage, tel le GPS.

Le problème de synchronisation se pose également dans les processeurs, où les signaux électriques n'ont pas le temps de se propager d'un bout à l'autre de la puce entre deux bips de l'horloge. L'horloge d'un ordinateur à 2 gigahertz a une période de 0,5 nanoseconde (un demi milliardième de seconde), c'est-à-dire qu'en une période, les signaux au sein des puces ne parcourent que quelques millimètres, alors que la largeur des puces est couramment de l'ordre de deux centimètres. Tout comme la simultanéité, l'ordre dans le temps de deux événements n'est clairement défini que dans le cas où les événements se produisent au même endroit. Pour des événements *A* et *B* distants dans l'espace, l'ordre temporel «*A* avant *B*» n'est définissable que si un signal peut voyager de *A* vers *B*, car dans ce cas la réception du signal est toujours postérieure à son émission. On dit encore que *A* et *B* sont en relation causale. Comme les signaux se propagent à vitesse finie, tous les couples d'événements ne satisfont pas nécessairement cette condition. Les couples d'événements qui ne peuvent pas être reliés de cette façon ne sont pas ordonnés dans le temps, et ne sont donc pas liés causalement.

Ainsi, au sein d'une puce, il n'existe pas d'ordre temporel total pour tous les événements qui s'y produisent. Dans les circuits synchrones, l'horloge maîtresse distribue un temps périodique à tous les endroits de la puce, selon un arbre dont toutes les branches sont parcourues en des temps approximativement égaux. Les événements du calcul sont ainsi sélectionnés de façon telle que leur ordre temporel et, par conséquent, leur relation causale sont définis sans ambiguïté au niveau global.

En revanche, dans les circuits asynchrones, l'activité d'un opérateur logique est déclenchée par l'arrivée de données sur ses entrées. Cela ne permet de définir un ordre temporel que pour une partie des événements du calcul seulement. Un calcul dépend en général de l'ordre global dans lequel les opérations s'effectuent. Or, dans certains cas, l'ordre relatif où arrivent les données sur différents opérateurs peut dépendre des durées de propagation. Afin de maîtriser le calcul effectué, une méthode consiste à restreindre les calculs possibles et à concevoir la puce de façon telle que ne subsistent que des calculs indépendants des retards dus aux propagations (on qualifie alors le calcul de «DI», c'est-à-dire insensible au délai). Certes, les performances en vitesse dépendent de ces retards, mais le résultat du calcul, lui, n'en dépend pas. La condition DI conduit à un ensemble de contraintes logiques, dites règles de Udding, qui restreignent le comportement des opérateurs logiques élémentaires (comme le *Rendez-vous* ou l'*Arbitre*). En fait, ces règles traduisent de façon logique l'existence ou l'absence d'une relation causale physique entre les événements du calcul.

La théorie de la relativité et les circuits asynchrones semblent deux domaines bien éloignés. Pourtant, ils se rejoignent sur une même question centrale : celle de l'existence d'une référence de temps et d'un ordre temporel. Cette convergence permet de mieux cerner les problèmes liés au calcul asynchrone, et d'envisager de concevoir des puces encore plus performantes. En retour, la physique doit aussi y gagner : la forte miniaturisation des circuits logiques (l'épaisseur d'une grille de transistor se rapproche des distances interatomiques) ne rend-elle pas indispensable de maîtriser l'espace-temps jusqu'au domaine microscopique ?

Philippe MATHERAT, Laboratoire traitement et communication de l'information, CNRS, École nationale supérieure des télécommunications et Marc-Thierry JAEKEL, Laboratoire de physique théorique, CNRS, École normale supérieure

vers la France. De même, si deux requêtes arrivent sur un *Arbitre* à quelques picosecondes d'écart, le circuit peut faire une pause dans son état métastable avant d'atteindre un de ses états stables et de résoudre le dilemme.

Les concepteurs débutants d'*Arbitre* cherchent souvent à éviter ces délais prolongés occasionnels en développant des circuits complexes. Ils commettent fréquemment l'erreur d'introduire un circuit qui pousse l'*Arbitre* vers une décision particulière si jamais le circuit se trouve dans l'état métastable. Cela revient à dresser l'âne de Buridan à aller vers la gauche (ou vers la droite) lorsqu'il est confronté à un choix difficile. Cependant un tel dressage a pour seul effet de présenter à l'âne trois choix plutôt que deux : aller à gauche, aller à droite, ou s'apercevoir qu'il y a un choix difficile et aller à gauche. Même un âne dressé mourra de faim s'il est incapable de trancher entre les deux dernières possibilités. En d'autres termes, pour reprendre la métaphore géographique, vous pouvez déplacer la ligne de crêtes des Pyrénées, mais vous ne pouvez pas vous en débarrasser. Bien qu'il n'existe aucun moyen d'éliminer la métastabilité, des circuits *Arbitre* simples et bien conçus peuvent garantir des pauses très courtes dans tous les cas. Typiquement, les *Arbitre* actuels ont des pauses normales de 100 picosecondes, et, exceptionnellement (une fois sur dix heures de fonctionnement), de 400 picosecondes. La probabilité qu'une pause survienne diminue exponentiellement avec la durée de la pause : une pause de 800 picosecondes se produit moins d'une fois pour un milliard d'années de fonctionnement.

Une nécessaire rapidité

Notre équipe étudie surtout le développement de systèmes asynchrones rapides. Nous avons observé que la simplicité est souvent garante de rapidité. Nous voulions construire un pipeline à double sens de circulation, comportant deux courants de données opposés (comme deux chaînes de passeurs de seaux parallèles, mais acheminant l'eau en sens contraire). Nous voulions que les données des deux flux interagissent à chaque étape ; la principale difficulté était de faire en sorte que chaque élément de données allant dans un sens interagisse avec chaque élément de données allant dans l'autre sens. L'arbitrage s'est révélé un facteur essentiel. À chaque jonction entre étapes successives, un circuit *Arbitre* autorisait le passage d'un seul élément à la fois. Nous avons beaucoup appris grâce à ce projet à propos de la coordination et de l'arbitrage. Nous avons ensuite construit des puces expérimentales pour vérifier la validité et la fiabilité de nos circuits *Arbitre*.

Plus récemment, nous nous sommes intéressés à une structure nommée FLEET. Ce nom fait référence à la rapidité de la structure, mais aussi à l'ensemble des composants qui la constituent, autant de «navires» de la flotte (*fleet*, en anglais) voguant à leur propre allure. Chaque navire effectue sa tâche en même temps que les autres ; le système FLEET les contrôle en faisant circuler des données parmi eux par l'intermédiaire d'un réseau de commutateurs asynchrones. Les recherches menées pour le projet FLEET nous ont conduits à de nombreuses découvertes. Nous voulions de la vitesse, et c'est pourquoi nous avons créé les circuits GasP de base. Nous voulions aiguiller des données d'un pipeline à un autre, comme des voitures aux jonctions autoroutières, et cela nous a amenés à concevoir une famille de circuits GasP plus vaste, agissant comme un système dense de voies rapides. Ces circuits peuvent transporter les données environ deux fois plus vite que les systèmes cadencés. Pour évaluer la vitesse de nos réseaux de commutateurs, nous construisons souvent sur nos puces expérimentales des circuits en anneau où des données tournent. Nous mesurons le temps qu'il faut à une donnée pour effectuer un tour complet, ce qui est bien plus simple que de mesurer le temps – notablement plus court – nécessaire aux données pour avancer d'une étape.

Fonctionnement d'un module GasP

Les modules GasP servent également d'éléments *Rendez-vous* pour un pipeline asynchrone de données. Au centre de chaque module se trouve un circuit NAND, qui produit une sortie FAUX si les entrées sont toutes deux VRAI. Sinon le circuit NAND produit une sortie VRAI. Au départ, tous les signaux dans les fils de connexion des modules sont VRAI *(en bleu, a)*. L'arrivée de données dans le pipeline *(b)* change le signal d'arrivée en FAUX *(en rouge)*. Un inverseur bascule le signal sur VRAI et l'envoie au circuit NAND, qui change sa sortie en FAUX. Ceci entraîne l'ouverture de la porte, permettant aux données d'avancer dans le pipeline. La sortie FAUX ramène également le fil d'entrée à son état VRAI (c'est-à-dire vide) par l'intermédiaire d'un transistor, et met le fil de sortie dans l'état FAUX (c'est-à-dire plein) par le biais d'un inverseur et d'un transistor. La sortie de la porte NAND redevient alors VRAI *(c)*, ce qui ferme la porte. Pendant ce temps, le signal FAUX du fil de sortie déclenche le même processus dans le module GasP suivant.