sont FAUX, la sortie devient FAUX. Dans les autres cas, la sortie ne change pas. Les éléments C de Muller servent notamment à contrôler le flux des données. Par un agencement astucieux de plusieurs d'entre eux, ils permettent de gérer l'avancée de données le long d'une chaîne de passeurs électronique sans le besoin d'une horloge.

Notre groupe de recherche a récemment introduit un nouveau type de circuit *Rendez-vous*, nommé GasP *(voir l'encadré page 41)*. Le GasP est dérivé d'une famille plus ancienne de circuits mise au point par Charles Molnar, qui travailla aux Laboratoires *Sun Microsystems*. Ch. Molnar avait baptisé sa création asP* (pour protocole de pulsation asynchrone et symétrique). Nous y avons ajouté le *G*, pour obtenir l'onomatopée *(gasp)* qui, en anglais, traduit la surprise, surprise créée ici par la rapidité des nouveaux circuits. Nous avons établi que les modules GasP ont une rapidité et une efficacité énergétique aussi bonnes que les éléments C de Muller, qu'ils s'assemblent mieux avec les circuits ordinaires et qu'ils sont beaucoup plus faciles à utiliser dans des circuits complexes.

Les circuits *Arbitre* remplissent, quant à eux, une autre fonction essentielle pour les ordinateurs asynchrones. Un *Arbitre*, tel un agent de la circulation posté à une intersection qui décide quel véhicule passera le premier, orchestre le traitement des données. S'il n'y a qu'une requête, l'*Arbitre* donne immédiatement le feu vert à l'action correspondante, et met en attente une éventuelle seconde requête jusqu'à ce que la première soit terminée. Quand l'*Arbitre* reçoit deux requêtes quasi simultanément, il doit décider à laquelle accorder la priorité. Par exemple, quand deux processeurs réclament l'accès à une mémoire au même moment, le rôle de l'*Arbitre* est de n'accorder l'accès qu'à un processeur à la fois. L'*Arbitre* garantit que deux actions ne se déroulent jamais en même temps.

## L'âne de Buridan

Bien que les circuits *Arbitre* n'accèdent jamais à plus d'une requête à la fois, il n'existe aucun *Arbitre* qui tranche entre les diverses requêtes en un temps limité. Les *Arbitre* actuels prennent leurs décisions très rapidement en moyenne, généralement en quelques picosecondes (un millième de milliardième de seconde). Cependant, quand les demandes sont très rapprochées, les circuits mettent parfois deux fois plus longtemps à trancher, voire, dans quelques rares cas, dix fois plus.

Les difficultés auxquelles est confronté l'*Arbitre* pour prendre ses décisions rappellent celles de l'âne de Buridan. Selon cette parabole attribuée à Buridan, philosophe français du XIV$^{e}$ siècle, un âne affamé et assoiffé, se trouvant à égale distance d'un picotin d'avoine et d'un seau d'eau, mourrait de faim et de soif parce qu'il serait incapable de choisir entre l'un ou l'autre. Nous sommes confrontés tous les jours à ce type de dilemmes, par exemple lorsque deux

## Fonctionnement d'un circuit *Rendez-vous*

Les circuits *Rendez-vous* gèrent le flux des données à travers des circuits sans horloge centrale. Ici un pipeline électronique est contrôlé par une chaîne de circuits *Rendez-vous*, nommés éléments C de Muller. Chacun de ces éléments autorise le passage des données *(en orange)* seulement quand l'étage précédent est «plein» (les données sont prêtes à poursuivre leur chemin) et que l'étage suivant est «vide».

Chaque élément C de Muller a deux entrées et une sortie. La sortie devient FAUX quand les deux entrées sont FAUX, et redevient VRAI quand les deux entrées sont VRAI (dans le diagramme, les signaux VRAI sont en bleu, et les signaux FAUX en rouge). L'inverseur fait en sorte que les entrées initiales diffèrent de sorte qu'au départ, tous les étages sont vides.

Supposons que l'entrée de gauche soit VRAI et celle de droite FAUX *(a)*

Un changement de l'entrée de gauche de VRAI à FAUX *(b)* indique que l'étage de gauche est plein, c'est-à-dire que des données sont arrivées. Comme les deux entrées de l'élément sont maintenant toutes deux FAUX, l'élément bascule sa sortie sur FAUX. Cette modification a trois conséquences : des données sont transmises dans le pipeline pour poursuivre leur trajet (les portes s'ouvrent), un signal FAUX est renvoyé vers l'élément C de Muller précédent pour que l'étage soit vidé et un signal FAUX est expédié vers l'élément C de Muller suivant, de sorte que l'étage suivant est plein *(c)*.