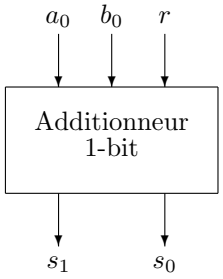
Option Informatique en Spé MP et MP*
TD : l'additionneur diviser pour régner

▶ On étudie dans cette partie deux modèles de circuits logiques *additionneurs n-bits* ; un tel circuit comporte $2n+1$ entrées : $a = (a_0, \ldots, a_{n-1})$ est le premier opérande, $b = (b_0, \ldots, b_{n-1})$ est le deuxième opérande et r est la retenue *entrante*. Il comporte $n+1$ sorties $s_0, \ldots, s_n$ et est entièrement décrit par l'équation

$$\sum_{0 \leqslant k < n} 2^k a_k + \sum_{0 \leqslant k < n} 2^k b_k + r = \sum_{0 \leqslant k \leqslant n} 2^k s_k$$

On note que s_n est la retenue *sortante*.

Question 1 • En utilisant des portes logiques élémentaires, construisez un circuit additionneur 1-bit. Vous commencerez par écrire les équations logiques exprimant s_0 et s_1 en fonction de a_0, b_0 et r.



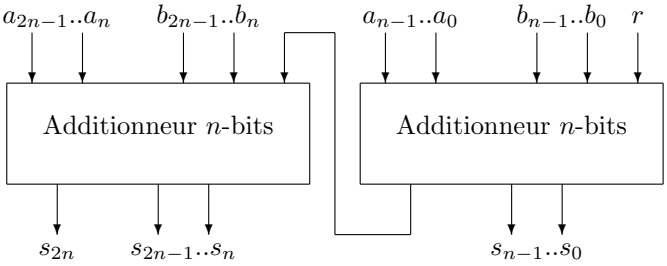
Question 2 • Expliquez comment assembler n additionneurs 1-bit pour obtenir un additionneur n-bits.

Question 3 • Déterminez le nombre $s(n)$ de portes logiques élémentaires nécessaires pour la réalisation de cet additionneur n-bits.

Question 4 • On suppose que le temps de propagation d'un signal logique à travers une porte logique élémentaire est une constante τ indépendante de la porte. Exprimez, en fonction de n et τ, le délai $t(n)$ qui s'écoule entre la présentation des données $a_0, \ldots, a_{n-1}, b_0, \ldots, b_{n-1}, r$ aux entrées du circuit additionneur n-bits que l'on vient de construire, et l'obtention des résultats $s_0, \ldots, s_n$ sur ses sorties.

▶ On se propose de construire un additionneur n-bits, fondé sur le principe *diviser pour régner*, pour diminuer le délai d'obtention du résultat. Bien entendu, comme on ne peut prétendre avoir à la fois le beurre et l'argent du beurre, la complexité du circuit (le nombre de portes logiques requises pour le réaliser) augmentera.

▶ L'étudiant Jean-Marcel MALHABILE propose le schéma suivant, pour réaliser un additionneur $2n$-bits à partir de deux additionneurs n-bits.



Question 5 • D'après vous, quel gain notre ami Jean-Marcel va-t-il retirer de cette savante construction ?

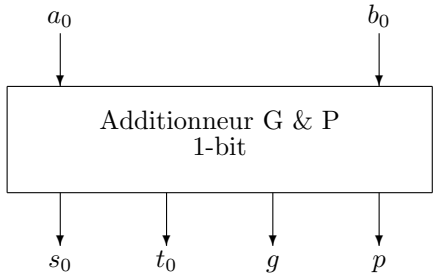
▶ On part alors de l'idée suivante : chaque additionneur n-bits va calculer en fonction de ses entrées $a = (a_{n-1}, \ldots, a_0)$ et $b = (b_{n-1}, \ldots, b_0)$ *deux* résultats : $s = (s_{n-1}, \ldots, s_0)$ qui correspond au cas où la retenue entrante est égale à 0, et $t = (t_{n-1}, \ldots, t_0)$ qui correspond au cas où la retenue entrante est égale à 1. Le circuit va calculer également deux indicateurs :

1. le bit de *génération* de retenue noté g, qui sera égal à 1 si et seulement si le calcul de la somme $a+b$ génère[1] une retenue ;
2. le bit de *propagation* de retenue noté p, qui sera égal à 1 si et seulement si le calcul de la somme $a+b+1$ génère une retenue, ce qui revient à dire que le calcul de la somme $a+b$ *propage* une éventuelle retenue entrante.

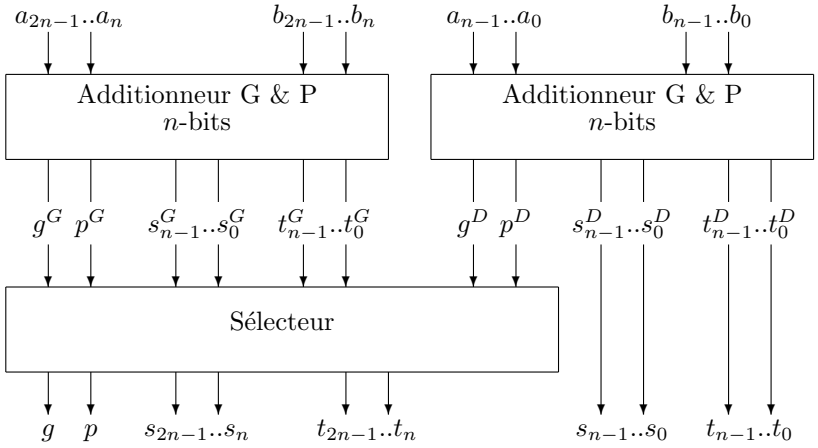
[1] Le verbe *générer* a été préféré à *produire* pour trois raisons : il est proche du terme anglo-saxon *generate* ; son initiale n'est pas la même que celle de *produire*, ce qui permet d'adopter des notations cohérentes avec la terminologie ; enfin, il est présent dans le *Petit Robert* auquel j'ai recours.

Nous dirons qu'un additionneur construit selon ce principe est un *additionneur à génération et propagation de retenue*, ou, en abrégé, *additionneur G & P*.

Question 6 • En utilisant des portes logiques élémentaires, construisez un circuit additionneur 1-bit avec génération et propagation de retenue. Vous commencerez par écrire les équations logiques exprimant s_0, t_0, g et p en fonction de a_0 et b_0.



▶ Le schéma d'ensemble qui suit décrit la réalisation de principe d'un additionneur $2n$-bits avec génération et propagation de retenue, à partir de deux additionneurs n-bits du même type et d'un *sélecteur* qui reste à décrire. Pour faciliter la lecture, les sorties des deux additionneurs n-bits sont marquées respectivement G (pour *gauche*) et D (pour *droite*).



Question 7 • Donnez des formules logiques exprimant g et p en fonction de g^G, p^G, g^D et p^D.

Question 8 • Donnez de même des formules logiques exprimant s_i et t_i en fonction de t_i^G, s_i^G, g^D et p^D, et ce pour $i \in [\![n, 2n-1]\!]$.

▶ On note $T(n)$ le délai d'obtention du résultat avec un additionneur n-bits *diviser pour régner* construit à partir de portes logiques élémentaires, et $S(n)$ le nombre de ces portes utilisées pour réaliser un tel additionneur.

Question 9 • Exprimez $T(2n)$ en fonction de $T(n)$; en déduire une expression simple de $T(2^n)$.

Question 10 • Présentez dans un tableau les valeurs de $t(2^n)$ et $T(2^n)$ pour $n \in [\![0, 6]\!]$ (on prendra $\tau = 1$ pour simplifier).

Question 11 • De la même façon, exprimez $S(2n)$ en fonction de $S(n)$, puis en déduire une expression simple de $S(2^n)$.

Question 12 • Présentez dans un tableau les valeurs de $s(2^n)$ et $S(2^n)$ pour $n \in [\![0, 6]\!]$.

▶ Une fonction f de $\mathcal{B}^n$ dans $\mathcal{B}$ est *complètement dépendante* si, pour tout indice $i \in [\![1, n]\!]$, il existe au moins un n-uplet $x = (x_1, \ldots, x_n)$ tel que

$$f(x_1, \ldots, x_{i-1}, x_i, x_{i+1}, \ldots, x_n) \neq f(x_1, \ldots, x_{i-1}, \overline{x_i}, x_{i+1}, \ldots, x_n)$$

On considère un circuit combinatoire construit à partir de portes logiques élémentaires, évaluant une telle fonction f complètement dépendante.

Question 13 • Montrez que le délai qui s'écoule entre la présentation des données aux entrées de ce circuit, et l'obtention du résultat à la sortie, est au moins égal à $\lceil \lg n \rceil \tau$.

Question 14 • Montrez que la fonction qui associe au $2n$-uplet

$$(a_0, \ldots, a_{n-1}, b_0, \ldots, b_{n-1})$$

le bit de poids 2^n de la représentation en base 2 de $\sum_{0 \leqslant i < n} 2^i(a_i + b_i)$ est complètement dépendante.

Question 15 • En déduire qu'un circuit additionneur n-bits, construit à partir de portes logiques élémentaires, et fondé sur la représentation des opérandes en numération de position en base 2, a un temps de calcul qui ne peut être négligeable devant $\ln n$.

FIN