ES102/PC2 : énoncé

1) Soustraction entière et entiers négatifs (~20')

1a) Calculer 8-6 sur 5 bits, d'abord directement, puis en recourant au complément.

1b) Quel est le code de -4 en tant qu'entier signé sur 5 bits ? Le vérifier avec les poids.

1c) Retrouver le code de l'entier signé +4 en calculant celui de -(-4).

1d) Trouver à la main le code de l'entier 235 sur 8 bits. Puis l'écrire en hexadécimal.

2) Représentation flottante (~20')

2a) Calculer la décomposition du nombre réel 1/6 en puissances négatives de 2.

2b) En déduire la représentation du réel 1/6 en format flottant 64 bits.

2c) Quel est l'ensemble des réels ayant la même représentation flottante sur 64 bits que 1/6 ?

3) Réduction de délai par manipulations algébriques (~1h10')

On désigne par *commutation* (ou *transition*) le changement de valeur d'un signal binaire. Le délai d'un circuit combinatoire se mesure par le temps maximal qui s'écoule entre des commutations en entrée et les commutations qui en résultent en sortie. Parmi les multiples scénarios possibles, c'est le plus lent qui définit le temps de réponse du circuit.

L'esprit général de l'exercice est de mettre les possibilités de manipulation algébrique des formules booléennes au service de la réduction du délai combinatoire, pour calculer plus vite.

On dispose de la *bibliothèque* suivante de portes logiques. Le délai de chacune apparaît sur son symbole. Il s'agit d'un délai typique normalisé par rapport à celui de l'inverseur, qui est actuellement de quelques picosecondes (10^{-12} s) sur les technologies les plus avancées.

En fait, le délai réel dépendra aussi de ce qui se trouve en sortie de la porte, plus précisément de la capacité à charger. Mais ce point sera secondaire dans la suite de l'exercice.

Les portes à 2 entrées les plus rapides de cette bibliothèque sont NAND et NOR (en réalité, la NOR est soit moins rapide soit moins compacte que la NAND, comme expliqué au CM4). Une porte AND présente le même délai qu'une porte NAND suivie d'un inverseur : c'est effectivement ainsi qu'elle est réalisée en pratique. Sa présence au sein de la bibliothèque n'est donc en fait qu'une commodité. De même pour la porte OR. Au contraire, XOR et XNOR présentent une rapidité inégalable par combinaison des autres portes.

3a) Essayons pour la porte XNOR. Commencer par établir et implanter directement sa FDM. Puis procéder par « jeu de bulles » pour exploiter les portes les plus rapides, en particulier la porte NAND, et retranscrire sous forme algébrique la transformation ainsi réalisée. Enfin, déterminer le délai de chacune des 2 réalisations.

Plus généralement, comment accélérer un circuit combinatoire trop lent pour l'application visée ? Pour cela, il est utile d'identifier son « chemin critique ». Un chemin est ici une liste de

connexions qui se suivent au sein du circuit, franchissant les portes dans le sens entrée-sortie. A chaque chemin est associé le total des délais des portes franchies. On considère alors l'ensemble de tous les chemins qui relient entrées et sorties du circuit considéré. Le chemin critique est celui qui présente le plus long délai. Il n'est pas forcément unique.

Remarque : sur un plan algorithmique, il s'agit de trouver le plus long chemin sur un graphe orienté acyclique (dont les nœuds sont les portes et les arêtes/arcs sont les connections). C'est un problème classique (très utile en ordonnancement, notamment), solutionnable en temps linéaire par rapport au nombre de nœuds et arêtes (contrairement à son équivalent sur graphe quelconque qui, lui, ne connaît pas de solution en temps polynomial).

3b) Identifier les chemins critiques des circuits proposés à la question précédente.

La mise en série de *n* FAs (*Full Adder*) réalise un additionneur numérique, dit à retenues propagées. Constitué de quelques portes et destiné à être abouté en de multiples exemplaires, le FA est qualifié de « cellule ». Notons τ_{xy} le temps de réponse de la cellule FA entre son entrée x = *a*, *b* ou *c[in]* (retenue entrante) et sa sortie y = *s* ou *c[out]* (retenue sortante).

3c) Rappeler les équations de la cellule FA, liant ses sorties *s* et *cout* à ses entrées *a*, *b* et *cin*.

3d) Que peut-on dire du chemin critique de l'additionneur à retenues propagées ?

3e) Du coup, dans le cadre d'une implantation optimisée de la cellule FA, quelle est la caractéristique à minimiser en priorité ?

3f) Dans un FA, pour quelles valeurs de *a* et *b* la sortie *cout* dépend-elle effectivement de *cin* ? Ainsi, pour quelles paires d'entiers A et B l'additionneur présente-t-il vraiment le délai imposé par son chemin critique ?

Le besoin de minimiser le délai entre une entrée *x* et une sortie *z* particulières d'une cellule est général. Traitons le comme tel. Soit donc une fonction booléenne f telle que z=f(x,Y) où Y est la liste des autres entrées. Cherchons donc un montage où *x* intervienne au plus près de la sortie *z*.

3g) Exploiter la formule d'expansion de Boole pour aboutir à une première solution générique.

3h) Il est possible de faire mieux si f(x,Y) est croissante par rapport à *x* : $\forall Y, f(0,Y) \Rightarrow f(1,Y)$. Sous cette condition, démontrer l'égalité $f(x,Y) = f(0,Y) + x \cdot f(1,Y)$ puis l'exploiter.

3i) Appliquer les techniques développées ci-dessus pour optimiser la cellule FA et ainsi rendre l'additionneur à retenues propagées le plus rapide possible.