Université Joseph Fourier – Grenoble 1

Architecture des ordinateurs

Exercices sur les circuits

Décembre 2010

I. BCD : Binary Coded Decimal ou décimal codé en binaire

Extrait de l'article BCD de wikipédia : « Le binary coded decimal (BCD), qui peut se traduire en français par décimal codé en binaire, est un système de numération utilisé en électronique et en informatique pour coder des nombres d'une façon relativement proche de la représentation humaine usuelle (en base 10). En BCD, les nombres sont représentés en chiffres décimaux et chacun de ces chiffres est codé en binaire sur quatre bits.
La plupart des ordinateurs stockent les données dans des octets d'une taille de 8 bits. Deux méthodes communes permettent d'enregistrer les chiffres BCD de 4 bits dans un tel octet:

1. ignorer les quatre bits supplémentaires de chaque octet et leur ajouter quatre bits identiques (0 ou 1 comme pour EBCDIC) - format étendu.
2. enregistrer deux chiffres par octet ce qui est appelé le « packed » BCD - format compacté.

Bien que le BCD gâche de l'espace, il permet d'avoir une correspondance immédiate avec les codes de caractères ASCII ou EBCDIC. Les grands nombres sont facilement affichés sur des afficheurs à 7 segments en séparant les entiers. Le BIOS des PC conserve, en général, la date et l'heure en format BCD, probablement pour des raisons historiques (cela évite une conversion du binaire à l'ASCII).
L'utilisation du format étendu pour stocker les dates « AAMMJJ » est en grande partie à l'origine du bogue de l'an 2000.
Si un chiffre nécessite quatre bits, alors trois chiffres en nécessitent 12. On a alors 10^3 combinaisons. Or 10 bits suffisent pour les exprimer toutes.
Aujourd'hui, ces représentations sont utilisés par la plupart des SGBD pour représenter le format DECIMAL ou NUMERIC qui permettent de stocker des nombres plus grands que les entiers et plus précis que les nombres à virgule flottante.»

Q1. « le BCD gâche de l'espace », dixit wikipédia. Évaluez la perte. Dire si un codage "BCM" (milliers codés en binaire, où les nombres seraient codés 3 chiffres par 3 chiffres en binaire) aurait les mêmes avantages et inconvénients que le BCD.

Q2. Donnez la table de vérité, une formule booléenne équivalente et un circuit correspondant pour réaliser la fonction f :[0..9] → [1..28], f(x)=3x+1, où x et f(x) sont codés en BCD.

II. Petits circuits

Donner les tables de vérité des circuits suivants :

- Majorité à 3 entrées, une sortie. S=Majorité(E_2, E_1, E_0) le bit S est le gagnant du vote à la majorité.
- Encodeur à 3 entrées, 2 sorties. La sortie donne l'indice de la première entrée à 1, ou 3 si aucune entrée n'est à 1. Discuter du cas où l'on peut garantir qu'il y a toujours une et une seule entrée à 1.
- Décodeur à 3 entrées : le circuit a 8 sorties dont une seule est à 1 (vrai), celle dont l'indice correspond au nombre binaire donné en entrée sur trois bits. Autrement dit : $S_i = ((E_2\, E_1\, E_0)_{base\ 2} = i)$

III. UAL : Unité Arithmétique et Logique

Q1. Rappeler le circuit d'un additionneur 1 bit, et de l'additionneur n bits.

Q2. À partir du circuit précédent construire un circuit faisant des soustractions sur n bits.

Q3. Donner le dessin d'un circuit d'UAL pouvant faire les 4 opérations suivantes : Addition/Soustraction/Et (bit à bit)/ Opérande Gauche, au choix selon une commande (F_1 F_0).

IV. Addition récursive de Von Neumann

L'addition récursive de Von Neumann repose sur 2 principes : 1 - anticiper les retenues possibles lors des calculs d'une addition en effectuant 2 fois plus de calculs à un moment donnée (un calcul avec une hypothétique retenue entrante à 0 et un calcul avec une hypothétique retenue entrante à 1) ; 2 - décomposer les calculs de manière dichotomique.

Q1. Cas de base. Pour la somme de 2 nombres (A, B) codés sur 1 bit, donner la table de vérité du circuit de base qui effectue les calculs en doubles : avec une hypothétique retenue entrante à 0 et avec une hypothétique retenue entrante à 1. En entrée, il doit y avoir 2 nombres (A, B) codés sur 1 bit (A_0, B_0), en sortie, il doit y avoir 2 résultats (R, S), codés sur 2 bits (R_1 R_0, S_1 S_0), et l'on doit avoir R=A+B, S=A+B+1.

À partir ce cette table de vérité, on suppose la définition d'un circuit $\mathbf{C}_1$: (A_0, B_0) => (R_1 R_0, S_1 S_0). Donner le circuit correspondant.

Q2. Premier niveau. A l'aide de multiplexeurs, assembler 2 circuits $\mathbf{C}_1$ obtenus à la question précédente pour obtenir un circuit $\mathbf{C}_2$ effectuant la somme de 2 nombres (A, B) codés sur 2 bits, en effectuant les calculs en doubles : avec une hypothétique retenue entrante à 0 et avec une hypothétique retenue entrante à 1. En entrée il doit y avoir 2 nombres (A, B) codés sur 2 bits (A_1 A_0, B_1 B_0), en sortie, il doit y avoir 2 résultats (R, S), chacun sur 3 bits (R_2 R_1 R_0, S_2 S_1 S_0), et l'on doit toujours avoir R=A+B, S=A+B+1.

Indications : Pour obtenir cet assemblage, utiliser un premier circuit $\mathbf{C_1}$ avec (A_0, B_0), et un second circuit $\mathbf{C_1}$ avec (A_1, B_1). Combiner les résultats de ces circuits avec des multiplexeurs pour obtenir (R_2 R_1 R_0, S_2 S_1 S_0). R_0 S_0 est obtenu à partir du premier circuit $\mathbf{C}_1$. Les deux autres sorties de ce premier circuit $\mathbf{C}_1$ servent à commander les multiplexeurs (ce sont les retenues entrantes effectives du second circuit, elles peuvent valoir toutes les deux 0, ou toutes les deux 1, ou être différentes). En entrées de ces multiplexeurs on trouve également les sorties du second circuit $\mathbf{C}_1$. (Pas de table de vérité à cette question, juste le circuit)

Q3. Généralisation. Comme dans la question précédente, à l'aide de multiplexeurs et de 2 circuits $\mathbf{C}_n$ effectuant la somme de 2 nombres (A, B) codés sur n bits et donnant 2 résultats (R,S), chacun sur n+1 bits tels que R=A+B, S=A+B+1, obtenir un circuit $\mathbf{C}_{2n}$ effectuant la somme de 2 nombres (A, B) codés sur 2n bits et donnant 2 résultats (R,S), chacun sur 2n+1 bits tels que R=A+B, S=A+B+1.

Q4. Complexité. D'après votre table de vérité (Q1), évaluer la complexité du circuit $\mathbf{C}_1$. À partir de Q2, évaluer la complexité du circuit $\mathbf{C}_2$. Enfin, évaluer la complexité du circuit $\mathbf{C}_{32}$ et la comparer avec la complexité du circuit d'addition classique.

V. Comparaison sur 8 bits

Q1. Donner la table de vérité de la fonction binaire de comparaison ($Sup_{\geq}(A,B)$) pour des entrées A, B sur 1 bit : $Sup_{\geq}(A,B) = A \geq B$.

Q2. A partir de ce qui précède, en supposant que vous disposez de circuits $Sup_{\geq,1bit}$ et en s'inspirant de l'un des algorithmes ci-après proposer un <u>circuit combinatoire</u> qui réalise la comparaison entre 2 nombres.

Algorithme de comparaison : pour A, B deux entiers représentés sur n=7 bits (A_n, ... A_0), (B_n, ... B_0)

Formulation récursive :

$Sup_{\geq}(A,B,n)$: si n=0 alors retourne $A_0 \geq B_0$
sinon si $A_n=B_n$ alors retourne $Sup_{\geq}(A,B,n-1)$
sinon retourne $A_n \geq B_n$.

Appel : $Sup_{\geq}(A,B,8)$.

Formulation itérative :

$Sup_{\geq}(A,B)$: i=7 ; tant que i>0 et $A_i=B_i$ faire : i ← i-1 ; fin tant que ; retourne $A_i \geq B_i$.

Q3. A partir de ce qui précède proposer un <u>circuit séquentiel</u> qui réalise la comparaison entre 2 nombres.

Q4. Comparer les 2 circuits obtenus.

VI. Circuits d'un petit automate

Un automate de Moore est donné par la modélisation de ses états, de ses entrées, de ses sorties et par les tables de vérité de ses fonctions de sortie et de transition :

Etat	Représentation binaire
Init	00
tmp	01
Final	10

Entrées	Rep. bin.
« 0 »	0
« 1 »	1

Sortie	Rep. bin.
Pas Ok	0
Ok	1

Fonction de sortie : Etat	
00	0
01	0
10	1

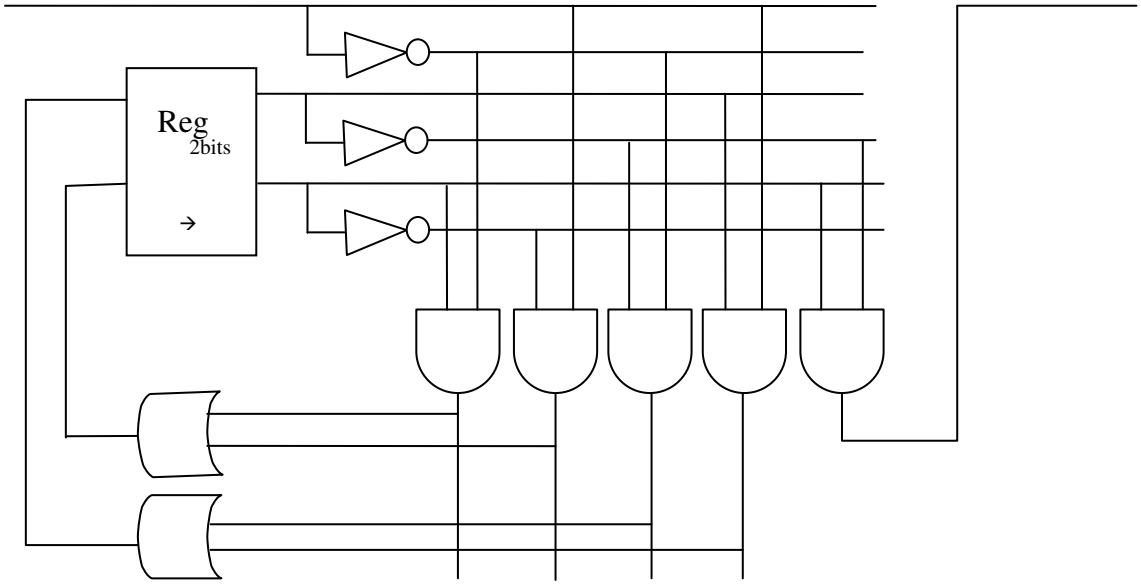
Transition : Etat	Entré	
00	0	00
00	1	01
01	0	01
01	1	10
10	0	10
10	1	10

Q1. Dessiner l'automate.

Q2. Dessiner un circuit qui réalise cet automate.

Q3. Dessiner le chronogramme qui correspond à la lecture de la suite d'entrées « 0 1 0 » (chaque valeur en entrée étant disponible sur une période d'horloge, l'automate étant dans l'état Init au départ).

VII. Circuit d'un petit automate



Q1. En supposant que le circuit ci-dessus représente un automate, identifier sur le schéma les différents éléments : état, prochain état, entrée, sortie, circuit de sortie et circuit de transition.
Q2. Donner les tables de vérité des deux circuits.
Q3. Dessiner l'automate correspondant.
Q4. Dessiner le chronogramme qui correspond à la lecture de la suite d'entrées « 0 1 0 » (chaque valeur en entrée étant disponible sur une période d'horloge, l'automate étant dans l'état 0 0 au départ).

VIII. Calcul de la puissance

Deux algorithmes sont proposés pour calculer R la puissance n d'un nombre X ($R \leftarrow X^n$).

Algorithme 1 :
R←1
Pour i allant de 1 à n faire :
  R←R*X

Algorithme 2 :
R←1
Tant que n != 0 faire :
  Si n est impair :
    R←R*X;N←N-1
  X←X²
  N←N/2

Q1. Choix de l'algorithme à implémenter. Choisir entre les deux algorithmes celui que vous comptez implémenter, donnez vos raisons.
Q2. Implémentation sous la forme d'un circuit à flot de donnée. Donner un circuit type « flot de données » réalisant l'exécution du calcul.
Q3. Implémentation sous la forme d'un circuit PC/PO. Donner le dessin de l'automate de la partie PC d'un circuit « PC/PO » réalisant le calcul.

IX. Devinette

L'algorithme suivant cherche à deviner un nombre « Mystère » entre 0 et 100 :

```
Début
Min ← 0
Max ← 100
Tant que Min ≠ Max :
      |     Somme ←Min + Max
      |     Si Somme < 2*Mystère :
      |           |    Min ← (Somme+1)/2
      |     sinon :
      |           |    Max ← Somme/2
```

Q1. Implémentation sous la forme d'un circuit à flot de donnée. Donner un circuit type « flot de données » réalisant l'exécution du calcul.
Q3. Implémentation sous la forme d'un circuit PC/PO (Partie Contrôle/Partie Opérative). Donner le dessin de l'automate de la partie PC (partie contrôle) d'un circuit PC/PO réalisant le calcul.