Architecture des ordinateurs

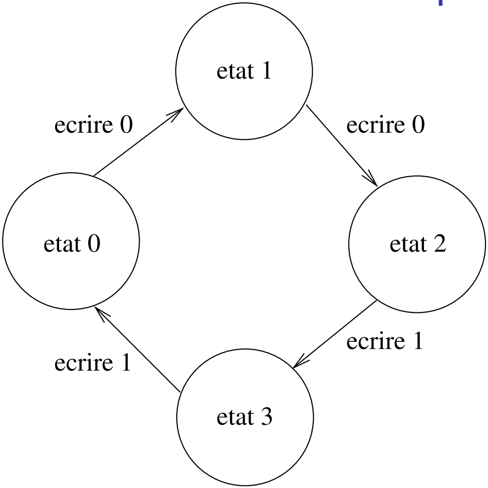
Automates

Automates

- circuits logiques combinatoires : coder une fonction
- circuits logiques séquentiels : mémoriser des valeurs
- les deux : fabriquer des automates

- automates finis déterministes : un nombre fini d'état, à partir d'un état donné, une transition ne mène qu'à un seul autre état possible.

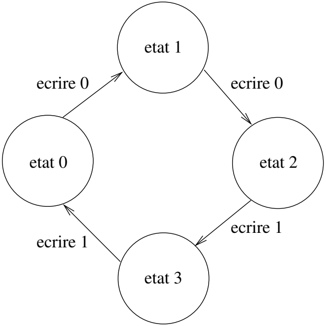
Exemple



- Un automate simple qui produit 0 0 1 1 0 0 1 1 0 0 1 1
- Un cercle : un état
- Une flèche : transition d'un état à un autre

Du graphe à la réalisation (1/2)

- Un registre pour stocker l'état (nombre binaire)
  $\rightarrow$ 4 états donc 2 bits : $e_1\ e_0$
- Un circuit logique combinatoire pour créer les transitions
- La sortie dépendant de l'état courant : out



état courant	e_1	e_0	out
0	0	0	0
1	0	1	0
2	1	0	1
3	1	1	1

$$out = e_1$$

Du graphe à la réalisation (2/2)

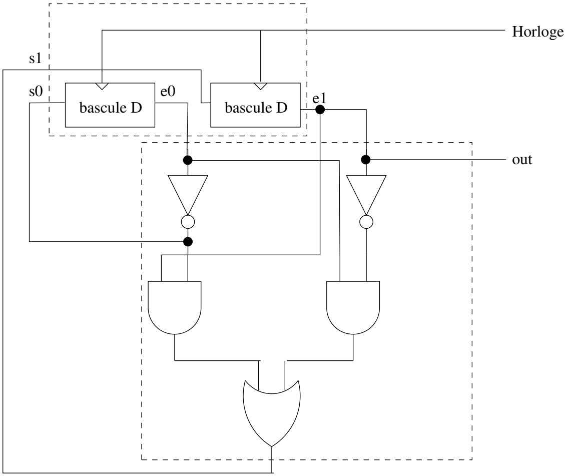
- Il faut calculer l'état suivant s_1 s_2 à partir de chaque état

état courant	état suivant	e_1	e_0	out	s_1	s_0
0	1	0	0	0	0	1
1	2	0	1	0	1	0
2	3	1	0	1	1	1
3	0	1	1	1	0	0

- De ces tables de vérité, nous déduisons que $s_1 = e_0.\overline{e_1} + \overline{e_0}.e_1$ et que $s_0 = \overline{e_0}$

Du graphe à la réalisation (2/2)

- $out = e_1$
- $s_1 = e_0.\overline{e_1} + \overline{e_0}.e_1$
- $s_0 = \overline{e_0}$



Réalisation d'un automate avec une entrée

- Créons une machine qui reçoit un nombre représenté en binaire, bit par bit de droite à gauche et multiplie ce nombre par 3.
- procédure : on multiplie un seul bit par 3, on donne le bit de résultat en sortie, et on garde la retenue pour l'ajouter à la multiplication du bit suivant
- Analyse des cas possibles :

$3 \times$ bit $+$ retenue	résultat	bit produit	retenue

Réalisation d'un automate avec une entrée

- Créons une machine qui reçoit un nombre représenté en binaire, bit par bit de droite à gauche et multiplie ce nombre par 3.
- procédure : on multiplie un seul bit par 3, on donne le bit de résultat en sortie, et on garde la retenue pour l'ajouter à la multiplication du bit suivant
- Analyse des cas possibles :

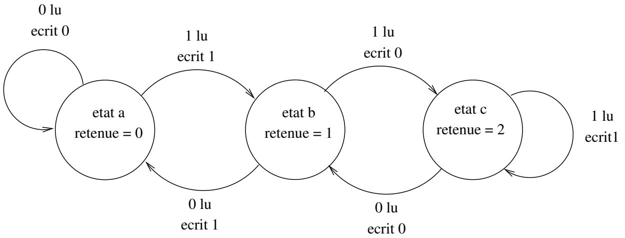
$3 \times \text{bit} + \text{retenue}$	résultat	bit produit	retenue
$3 \times 0 + 0$	0	0	0
$3 \times 1 + 0$	$3_{10} = 11_2$	1	1
$3 \times 0 + 1$	1	1	0
$3 \times 1 + 1$	$4_{10} = 100_2$	0	10
$3 \times 0 + 2$	$2_{10} = 10_2$	0	1
$3 \times 1 + 2$	$5_{10} = 101_2$	1	10

Création du graphe

$3 \times \text{bit} + \text{retenue}$	résultat	bit produit	retenue
$3 \times 0 + 0$	0	0	0
$3 \times 1 + 0$	$3_{10} = 11_2$	1	1
$3 \times 0 + 1$	1	1	0
$3 \times 1 + 1$	$4_{10} = 100_2$	0	10
$3 \times 0 + 2$	$2_{10} = 10_2$	0	1
$3 \times 1 + 2$	$5_{10} = 101_2$	1	10

- L'état de l'automate représente l'état de la retenue
- Une transition représente le bit lu en entrée et celui écrit en sortie
- Pour construire l'automate on se place dans l'état de départ (l'état a), puis on effectue son travail :
    1. lire le bit suivant du nombre à multiplier,
    2. choisir la transition qui y correspond depuis l'état courant,
    3. écrire la valeur indiquée sur la transition,
    4. l'état d'arrivée de la transition est le nouvel état courant,
    - recommencer à l'étape 1.

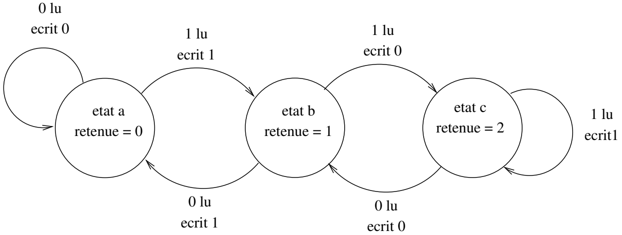
Vérification du graphe



- si l'automate reçoit en entrée les bits 110010
- qui correspondent au nombre $010011_2 = 19_{10}$
- il va passer par les états a, b, c, b, a, b, a
- et produire les bits 100111
- qui correspondent au nombre $111001_2 = 71_8 = 57_{10}$

état	a		b		c		b		a		b		a
bit lu		1		1		0		0		1		0	
bit écrit		1		0		0		1		1		1	

Construction de la table de vérité



état courant	e_1	e_0	i	etat suivant	n_1	n_0	o
0	0	0	0	0	0	0	0
0	0	0	1	1	0	1	1
1	0	1	0	0	0	0	1
1	0	1	1	2	1	0	0
2	1	0	0	1	0	1	0
2	1	0	1	2	1	0	1
	1	1	0	?	?	?	?
	1	1	1	?	?	?	?

Trouver les fonctions et les simplifier

état courant	e_1	e_0	i	etat suivant	n_1	n_0	o
0	0	0	0	0	0	0	0
0	0	0	1	1	0	1	1
1	0	1	0	0	0	0	1
1	0	1	1	2	1	0	0
2	1	0	0	1	0	1	0
2	1	0	1	2	1	0	1
	1	1	0	?	?	?	?
	1	1	1	?	?	?	?

- $n_1 = \overline{e_1}.e_0.i + e_1.\overline{e_0}.i$
- $n_0 = \overline{e_1}.\overline{e_0}.i + e1.\overline{e_0}.\bar{i}$
- $o = \overline{e_1}.\overline{e_0}.i + \overline{e1}.e_0.\bar{i} + e1.\overline{e_0}.i$

Trouver les fonctions et les simplifier

- $n_1 = \overline{e_1}.e_0.i + e_1.\overline{e_0}.i$
- $n_0 = \overline{e_1}.\overline{e_0}.i + e1.\overline{e_0}.\bar{i}$
- $o = \overline{e_1}.\overline{e_0}.i + \overline{e1}.e_0.\bar{i} + e1.\overline{e_0}.i$

	$e_1 e_0$	$e_1 \overline{e_0}$	$\overline{e_1}.\overline{e_0}$	$\overline{e_1} e_0$
i	?	V		V
$\bar{i}$	?			

donc $n_1 = e_1.i + e_0.i$.

Trouver les fonctions et les simplifier

- $n_1 = e_1.i + e_0.i$
- $n_0 = \overline{e_1}.\overline{e_0}.i + e1.\overline{e_0}.\bar{i}$
- $o = \overline{e_1}.\overline{e_0}.i + \overline{e1}.e_0.\bar{i} + e1.\overline{e_0}.i$

	$e_1 e_0$	$e_1 \overline{e_0}$	$\overline{e_1}.\overline{e_0}$	$\overline{e_1} e_0$
i	?		V	
$\bar{i}$	?	V		

donc $n_0 = e_1.\bar{i} + \overline{e_1}.\overline{e_0}.i$.

Trouver les fonctions et les simplifier

- $n_1 = e_1.i + e_0.i$
- $n_0 = e_1.\bar{i} + \overline{e_1}.\overline{e_0}.i$
- $o = \overline{e_1}.\overline{e_0}.i + \overline{e1}.e_0.\bar{i} + e1.\overline{e_0}.i$

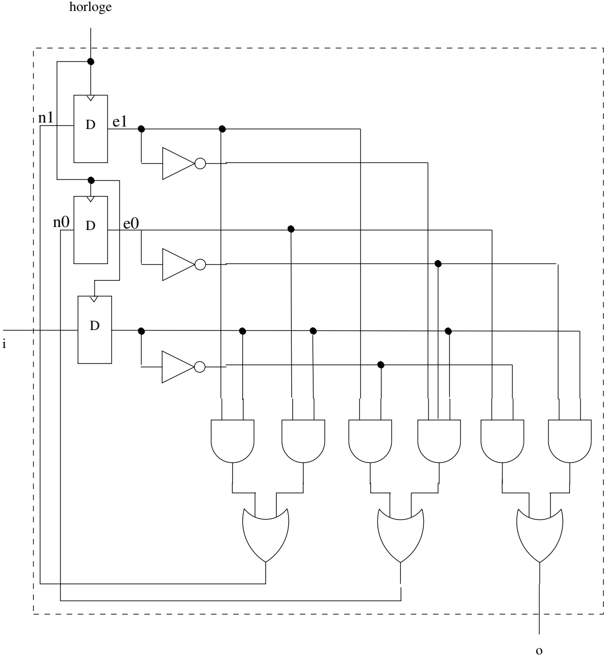
	$e_1 e_0$	$e_1 \overline{e_0}$	$\overline{e_1}.\overline{e_0}$	$\overline{e_1} e_0$
i	?	V	V	
$\bar{i}$	?			V

donc $o = \overline{e_0}.i + e_0.\bar{i}$.

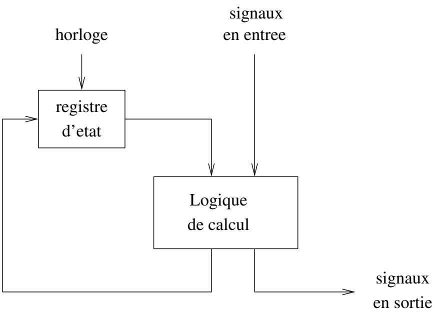
Trouver les fonctions et les simplifier

- $n_1 = e_1.i + e_0.i$
- $n_0 = e_1.\bar{i} + \overline{e_1}.\overline{e_0}.i$
- $o = \overline{e_0}.i + e_0.\bar{i}$

Réalisation de l’automate



Un automate général



- un registre contient l'état courant
- un bloc de logique calcule les signaux de sortie et l'état suivant à partir de l'état courant et des signaux en entrée
- une horloge assure la transition d'un état au suivant