


INFO-132511
Objets Connectés et
Programmation Microcontrôleurs

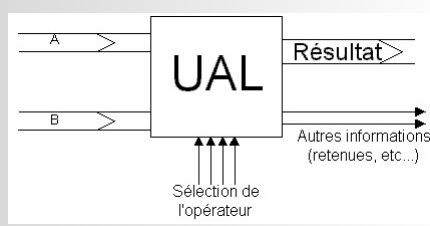
Cours #07
 Logique séquentielle
 Compteurs, registres à décalage
 Bernard Besserer

1




Circuits Logiques : pourquoi s'y intéresser ?

- Les opérateurs (et les circuits) de logique combinatoire sont la base de la logique séquentielle, puis de l'ALU
- Quelquefois, des circuits numériques (donc de la logique combinatoire et/ou de la logique séquentielles) sont inévitables... par exemple pour « économiser » des I/O en faisant du multiplexage



2



Concepts de base : logique combinatoire

- Les bases du système binaire (considérées comme acquises)
 - constantes logiques : true (1), false (0)
 - Numération binaire
 - Opérateurs logiques de base: ET, OU, NON, XOR et algebre de Boole
- Système logique combinatoire :
 - La sortie dépend directement du niveau logique des entrées
 - La sortie ne dépend PAS de l'histoire du système (pas de mémorisation)


Exprimer une équation logique :

Un master dans une université propose des UE d'enseignement dites "majeures" (correspondant à la discipline) nommées X, Y et Z, ainsi que des UE dites "mineures" nommées A et B. L'étudiant obtient son diplôme (D) si :

il obtient toutes les UE dites majeures de X à Z ou il obtient 2 UE majeures de son choix et les 2 UE mineures

$D = \mathbf{X.Y.Z + X.Y.A.B + X.Z.A.B + Z.Y.A.B}$

3



Logique combinatoire, coté systèmes















False -> etat bas (*low*)
état 0 : 0V, GND

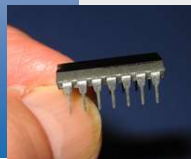
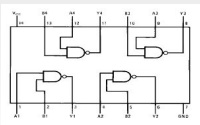
True -> etat haut (*high*)
état 1 : 5V, 3.3V, Vcc, +V

On sait créer des circuits :


- électromécaniques
- à base de transistors
- circuit intégrés

qui « concrétisent » les opérateurs logiques


Porte OUI (YES)			<table border="1" style="font-size: 0.8em;"> <tr><th>entrée</th><th>sortie</th></tr> <tr><td>0</td><td>0</td></tr> <tr><td>1</td><td>1</td></tr> </table>	entrée	sortie	0	0	1	1				
entrée	sortie												
0	0												
1	1												
Porte NON (NO)			<table border="1" style="font-size: 0.8em;"> <tr><th>entrée</th><th>sortie</th></tr> <tr><td>0</td><td>1</td></tr> <tr><td>1</td><td>0</td></tr> </table>	entrée	sortie	0	1	1	0				
entrée	sortie												
0	1												
1	0												
Porte ET (AND)			<table border="1" style="font-size: 0.8em;"> <tr><th>entrées</th><th>sortie</th></tr> <tr><td>0 0</td><td>0</td></tr> <tr><td>0 1</td><td>0</td></tr> <tr><td>1 0</td><td>0</td></tr> <tr><td>1 1</td><td>1</td></tr> </table>	entrées	sortie	0 0	0	0 1	0	1 0	0	1 1	1
entrées	sortie												
0 0	0												
0 1	0												
1 0	0												
1 1	1												
Porte OU (OR)			<table border="1" style="font-size: 0.8em;"> <tr><th>entrées</th><th>sortie</th></tr> <tr><td>0 0</td><td>0</td></tr> <tr><td>0 1</td><td>1</td></tr> <tr><td>1 0</td><td>1</td></tr> <tr><td>1 1</td><td>1</td></tr> </table>	entrées	sortie	0 0	0	0 1	1	1 0	1	1 1	1
entrées	sortie												
0 0	0												
0 1	1												
1 0	1												
1 1	1												
Porte OU exclusif (XOR)			<table border="1" style="font-size: 0.8em;"> <tr><th>entrées</th><th>sortie</th></tr> <tr><td>0 0</td><td>0</td></tr> <tr><td>0 1</td><td>1</td></tr> <tr><td>1 0</td><td>1</td></tr> <tr><td>1 1</td><td>0</td></tr> </table>	entrées	sortie	0 0	0	0 1	1	1 0	1	1 1	0
entrées	sortie												
0 0	0												
0 1	1												
1 0	1												
1 1	0												
Porte NON-ET (NAND)			<table border="1" style="font-size: 0.8em;"> <tr><th>entrées</th><th>sortie</th></tr> <tr><td>0 0</td><td>1</td></tr> <tr><td>0 1</td><td>1</td></tr> <tr><td>1 0</td><td>1</td></tr> <tr><td>1 1</td><td>0</td></tr> </table>	entrées	sortie	0 0	1	0 1	1	1 0	1	1 1	0
entrées	sortie												
0 0	1												
0 1	1												
1 0	1												
1 1	0												
Porte NON-OU (NOR)			<table border="1" style="font-size: 0.8em;"> <tr><th>entrées</th><th>sortie</th></tr> <tr><td>0 0</td><td>1</td></tr> <tr><td>0 1</td><td>0</td></tr> <tr><td>1 0</td><td>0</td></tr> <tr><td>1 1</td><td>0</td></tr> </table>	entrées	sortie	0 0	1	0 1	0	1 0	0	1 1	0
entrées	sortie												
0 0	1												
0 1	0												
1 0	0												
1 1	0												

4



Algèbre de Boole




George Boole (1815 - 1864) est un logicien, mathématicien et philosophe britannique. Il est le créateur de la logique classique, fondée sur une structure algébrique définissant une sémantique, et que l'on appelle algèbre de Boole en son honneur.

- Avec l'algèbre de Boole, les propositions logiques sont indiquées par des symboles et peuvent être exécutées par des opérateurs mathématiques abstraits qui correspondent aux lois de la logique.
- Théorème de de Morgan
 - Application principale : Transformation d'un opérateur « ou » en opérateur « et » et inversement
$$\overline{a \cdot b} = \overline{a} + \overline{b}$$

$$\overline{a + b} = \overline{a} \cdot \overline{b}$$
- Valable pour N variables

5



Arithmétique binaire : Cellule d'additionneur

- Binaire : systèmes de numération à poids positionnel
- En décimal, somme de deux nombres décomposé au niveau des unités / dizaines / centaines / etc... avec gestion des retenues
- En binaire, idem : somme au niveau de chaque bit
 - 3 entrées : A, B, C_{in}
 - 2 sorties : Somme, C_{out}

Row	Inputs			Outputs		Comment
	A	B	C _{in}	C _{out}	Somme	
0	0	0	0	0	0	0 + 0 + 0 = 00 ₂
1	0	0	1	0	1	0 + 0 + 1 = 01 ₂
2	0	1	0	0	1	0 + 1 + 0 = 01 ₂
3	0	1	1	1	0	0 + 1 + 1 = 10 ₂
4	1	0	0	0	1	1 + 0 + 0 = 01 ₂
5	1	0	1	1	0	1 + 0 + 1 = 10 ₂
6	1	1	0	1	0	1 + 1 + 0 = 10 ₂
7	1	1	1	1	1	1 + 1 + 1 = 11 ₂

6

Aritmetique binaire : Cellule d'additionneur

■ Tableau de Karnaugh

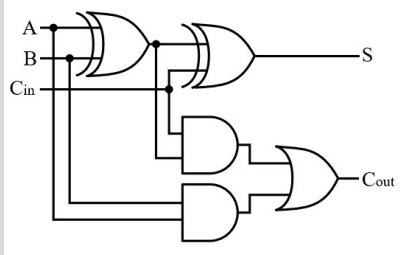
		C_{in}	
		0	1
AB	00		1
	01	1	
	11		1
	10	1	

		C_{in}	
		0	1
AB	00		
	01		1
	11	1	1
	10		1

■ Equation

■ Design de circuits électroniques :

- Cellules cascables
- Utilisation rationnelle de fonctions logiques (ensemble complet + théorème de DeMorgan)
-> réalisation de la cellule de l'additionneur avec des portes NAND uniquement
- Rationalisation + CAO/DAO = facilité de conception des circuits



7

Ce que l'on peut faire avec la logique combinatoire

■ On effectue des opérations logiques :

- Est-ce que tous les bits d'un mot sont à 0 ?
- Le nombre est-il positif ou négatif
- Inversion du 5eme bit du mot binaire : XOR
 - 10010011 XOR **00010000** → 10000011
- On peut additionner, soustraire (complément à 2)
- On peut comparer : if ($a > b$) → $(a - b) > 0$

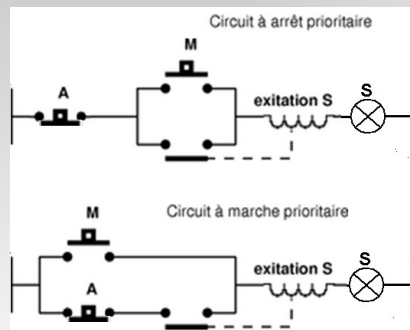
■ Comment réaliser (sans microcontrôleur) la fonction suivante :

- Un appui sur un bouton allume la lumière, un deuxième appui l'éteint...
- Même vecteur d'entrée (bouton au repos) mais 2 états de sortie différents (soit lumière allumé, soit lumière éteinte) : **pas réalisable en combinatoire**

8

Analogie électromécanique

- Il faut que le système dispose d'autres informations (une « mémoire », que l'on nomme aussi variable interne)
- On savait déjà le faire avec des systèmes électromécaniques :



$$S = \overline{A} \cdot (M + \overline{S})$$

$$S = M + (\overline{A} \cdot S)$$

- $S = f(..., S, ...)$: La sortie dépend de son propre état...

9

Introduction à la notion de mémoire

- Exemple de l'interrupteur bistable (~ bascule)
- Description du fonctionnement :
 - La sortie est à 1 si :
 - On actionne M ou si L est déjà à 1
 - La sortie est à 0 si :
 - On actionne A ou si L est déjà à 0

Lampe 2 boutons poussoirs: Arrêt (A), Marche (M)

Cahier des charges

M	A	L	
0	0	0	Aucun bouton enfoncé: lampe éteinte
1	0	1	Appui sur « M »: lampe s'allume
0	0	1	Relâche « M »: lampe reste allumée
0	1	0	Appui sur « A »: lampe s'éteint
0	0	0	Relâche « A »: lampe reste éteinte

Notation anglo-saxonne :
S (Set) pour la mise à 1 (marche)
R (Reset) pour la mise à 0 (arrêt)
La sortie est désignée par la lettre Q

10

Schéma des bascules

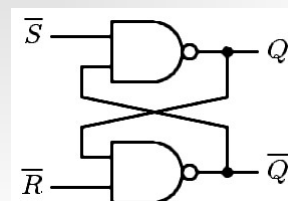
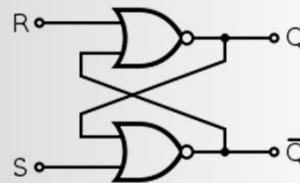
- A partir de :

$$S = \overline{A + (M + S)}$$

$$Q = \overline{R + (S + Q)}$$

$$S = \overline{\overline{M} \cdot (\overline{\overline{A}} \cdot S)}$$

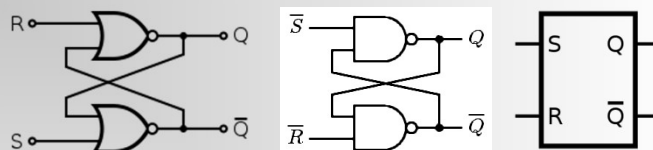
$$Q = \overline{\overline{S} \cdot (\overline{\overline{R}} \cdot Q)}$$



11

Schéma des bascules

- Briques de base de la logique séquentielle :
Bascule asynchrone = élément mémoire
(RS, flip-flop)



- Un système séquentiel peut donc être construit avec des briques de logique combinatoire (Formalisme de Moore / de Mealy)
- Autres bascules : bascule JK (synchrone, cadencé par une horloge), bascule D, bascule T, bascule maître-esclave, ...

12

Réalisation de systèmes séquentiels

- On peut représenter un système séquentiel par une suite d'états stables (les états internes)

$$L = 0 \quad L = 1$$

- Evolution du système : état précédent validé ET transition -> passage à l'état suivant
- On peut concrétiser avec des bascule. Le passage à l'état suivant met à 0 la bascule identifiant l'état précédent

13

Réalisation de systèmes séquentiels

$$L = 0 \quad L = 1$$

Attention : un système séquentiel peut aussi être bloqué ou bien instable (oscillateur) !!

Équation du circuit: $S = S + E$

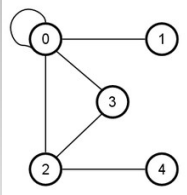
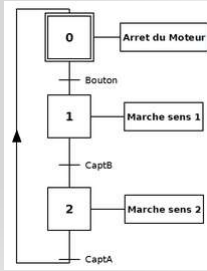
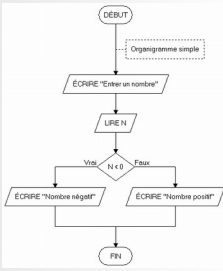
$S = 0$: Etat non définitif, évolution possible

$S = 1$: Etat définitif – la valeur de E n'a plus aucune influence... **RESET !**


Mémoire qui ne peut pas oublier !

14


Représentation des systèmes séquentiels

- Graphes, Réseaux de Petri
 
- Chronogramme (diagramme des temps)
 
- Graphe de fluence, graphe d'état, GRAFCET
 

Automate Programmable



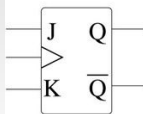
Microprocesseur



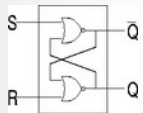
15

On fait le point :

- Logique combinatoire
 - On effectue des opérations logiques
 - On peut additionner, soustraire (comp. à 2), comparer, ...
- Bascules
 - Réalisé à partir de fonctions logiques combinatoires, c'est un élément mémoire
 - Bascule RS = élément mémoire asynchrone
 - Bascule JK = élément mémoire synchrone



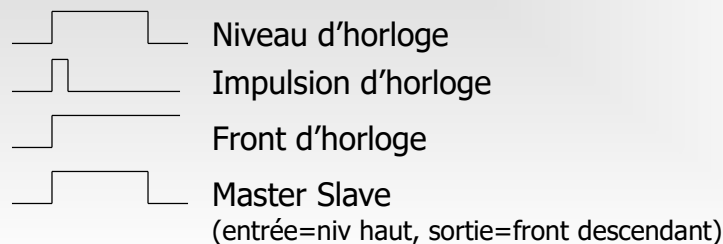
J	K	$Q_{(t+1)}$
0	0	$Q_{(t)}$ unchanged
0	1	0 reset
1	0	1 set
1	1	$\bar{Q}_{(t)}$ output inversion



16

Logique séquentielle : synchronisation

- **Système séquentiel asynchrone :**
le système évolue librement dès le changement d'une entrée...
- **Système séquentiel synchrone :**
Le système n'évolue qu'à des moments précis sous le contrôle d'un signal appelé *HORLOGE* (Clock). Permet de réaliser des systèmes qui seraient instables si asynchrone



17

Logique combinatoire + logique séquentielle

- Opérations logiques & arithmétiques
- Bascules asynchrones et synchrones (horloge)
- Avec des bascules on peut réaliser :
 - Des mémoires
 - Des compteurs (l'état d'une bascule change à chaque top)
 - Des registres à décalage
 - Et donc
 - Des mémoires « série » (Pile, File)
 - Un multiplicateur / diviseur en base 2

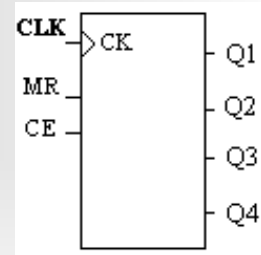


ALU
(Arithmetic
and Logic Unit)

18

Compteur

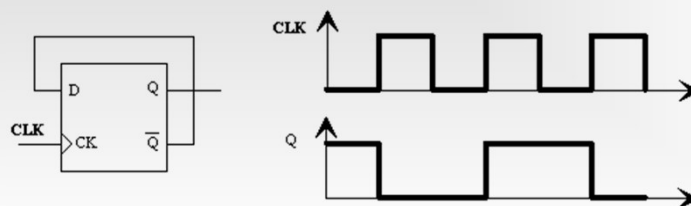
- **Un compteur est un circuit séquentiel qui permet de dénombrer des impulsions appliquées sur son entrée d' horloge et de restituer sur ses sorties les information de comptage sous forme binaire.**
- A chaque impulsion (front actif) l'état du compteur est modifié ; entre deux impulsions son état reste stable
- Les compteurs peuvent éventuellement disposer :
 - D'une broche de RAZ,
 - D'une broche de validation (Chip Enable),
 - D'une broche pour la mise à la valeur maximum (9 ou 15 selon de type),
 - D'une sélection de mode comptage/décomptage,
 - De deux horloges distinctes : l'une pour compter, l'autre pour décompter
 - De lignes de propagation de retenues pour les cascades



19

Compteur

- Le plus simple des compteurs (dit compteur asynchrone) est basé sur le diviseur de fréquence. Il s'agit d'une bascule T, la fréquence du signal d'horloge est divisée par deux.
- Rappel : la bascule T est une bascule JK avec $J=K=1$. On peut aussi réaliser facilement une bascule T avec une bascule D si l'on dispose de la sortie \bar{Q} .
- Un compteur asynchrone est simplement constitué de plusieurs diviseurs de fréquences (autant que de bits requis)



20

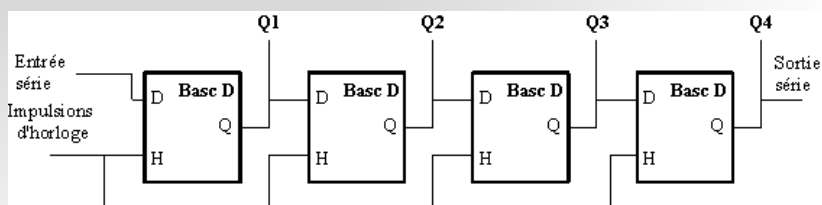
Registre à décalage

- Un registre à décalage est un registre (ensemble de cellules mémoires de 1 bit) de taille fixe dans lequel les bits sont **décalés** à chaque coup d'horloge
- Un registre à décalage est donc en général constitué d'un chaînage de bascule, la sortie d'une bascule étant reliée à l'entrée de la suivante. Il se décline en plusieurs variantes :
 - SIPO (Serial In - Parallel Out)
 - SISO (Serial In - Serial Out)
 - PISO (Parallel In - Serial Out)
 - PIPO (Parallel In - Parallel Out)

21

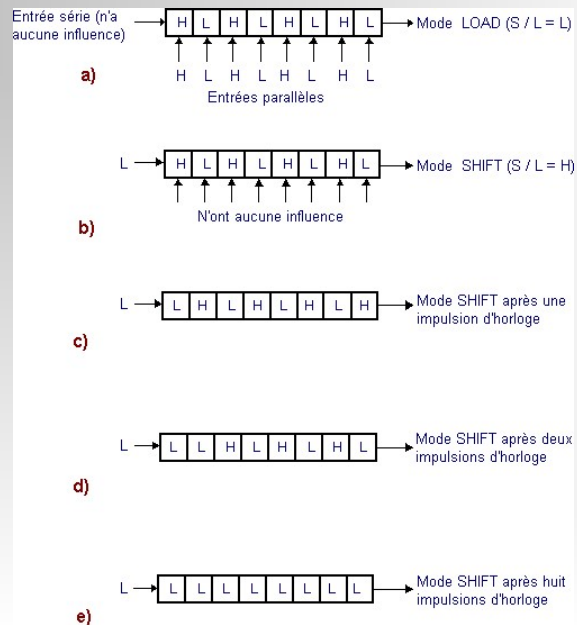
Registre à décalage

- Un registre à décalage peut être réalisé avec des bascules D
- Si la bascule dispose de lignes de SET et de RESET asynchrones, on peut precharger (load) le registre



22

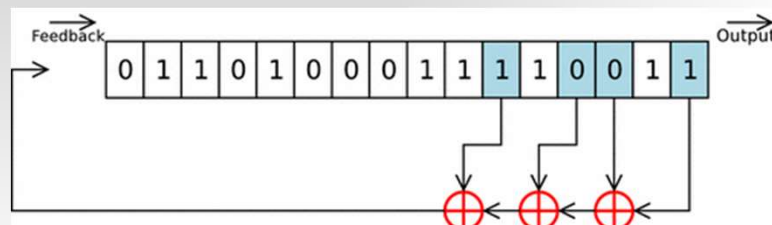
Fonctionnement d'un registre à décalage



23

Rebouclage de registre à décalage

- Le ou les bit(s) en sortie du registre subissent une série d'opérations pour être réinsérés dans le registre.
- Ce type de registre est utilisé en cryptographie ou pour le traitement du signal (filtrage), car un registre à décalage permet de calculer le résultat d'un polynôme binaire



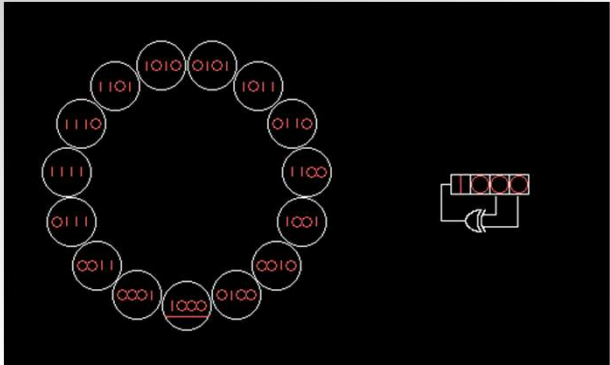
24

Génération de sequence pseudo-aléatoire

- Suites récurrentes linéaires :

$$u_{t+n} = \alpha_n u_t + \alpha_{n-1} u_{t+1} + \dots + \alpha_1 u_{t+n-1}$$
- Représentation polynomiale :

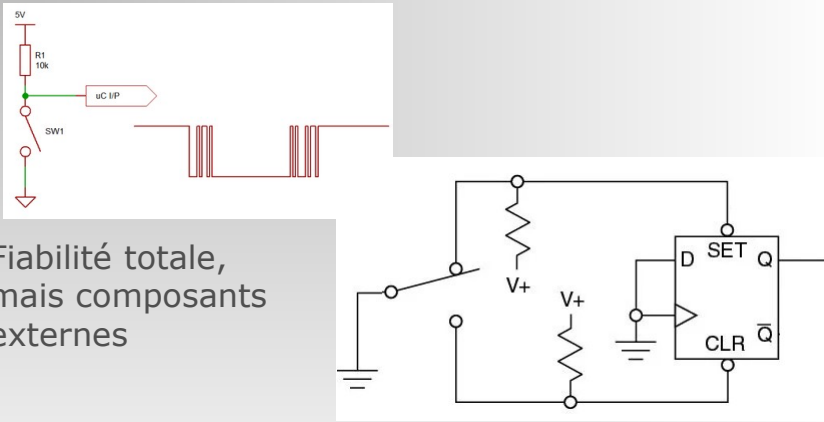
$$P(X) = 1 + \alpha_1 X + \dots + \alpha_n X^n$$



25

Utilisation avec les microcontrôleurs : bascule antirebonds

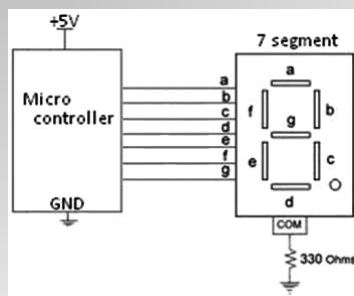
- Fiabilité totale, mais composants externes
- Alternatives :
 - Anti-rebond « software », moins fiable
 - Autres technologies : tactile capacitif



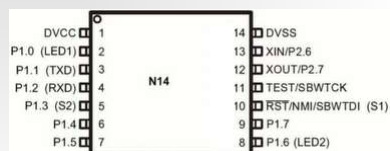
26

Utilisation avec les microcontrôleurs : Augmentation du nombres d'E/S

- Ex : Afficheur 7 segments



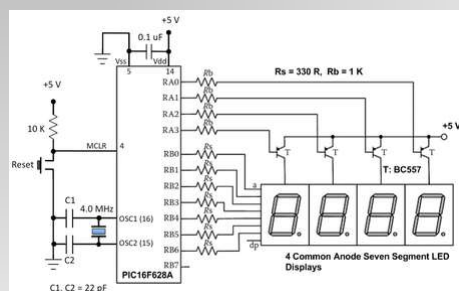
- Si affichage sur 4 digits ?
- Si microcontrôleur à empreinte réduite ?



27

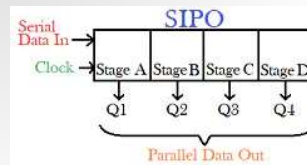
Commande d'afficheurs 7-segments : combien de sorties ?

- Si plusieurs afficheurs, multiplexage temporel



4 afficheurs :
 $7 + 4 = 11$
broches grâce au
multiplexage
temporel

- Augmentation du nombre d'E/S avec un registre à décalage
- Sortie Série, et « transformation » en parallèle.



28