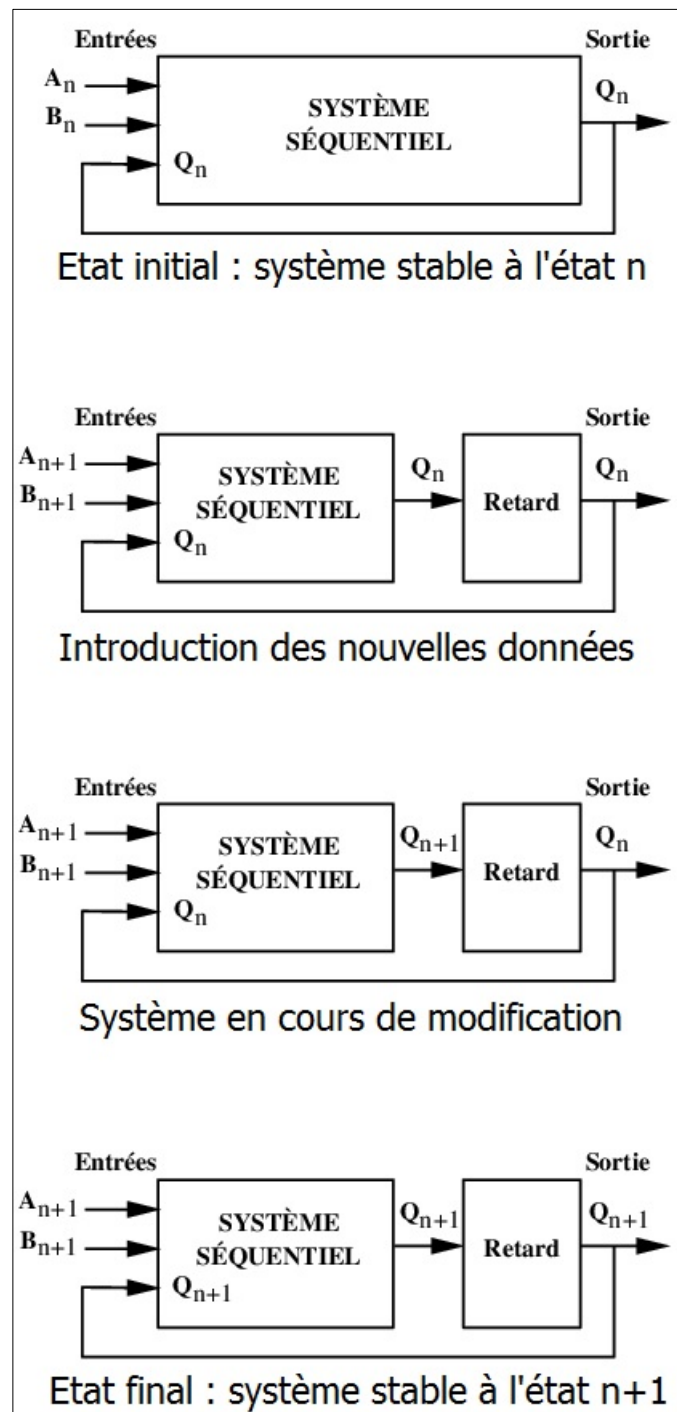


I – Définition :

Les circuits logiques combinatoires que nous avons vus précédemment sont des circuits idéaux (sans délais) et dépourvus de rétroaction (pas de retour des sorties dans les entrées).

Les circuits logiques séquentiels possèdent eux des rétroactions : la valeur de la sortie ne dépend alors plus uniquement des valeurs en entrée, mais aussi de la valeur antérieure de la sortie. La séquence des valeurs prises par la sortie va donc déterminer l'état du circuit.

Exemple de circuit séquentiel :

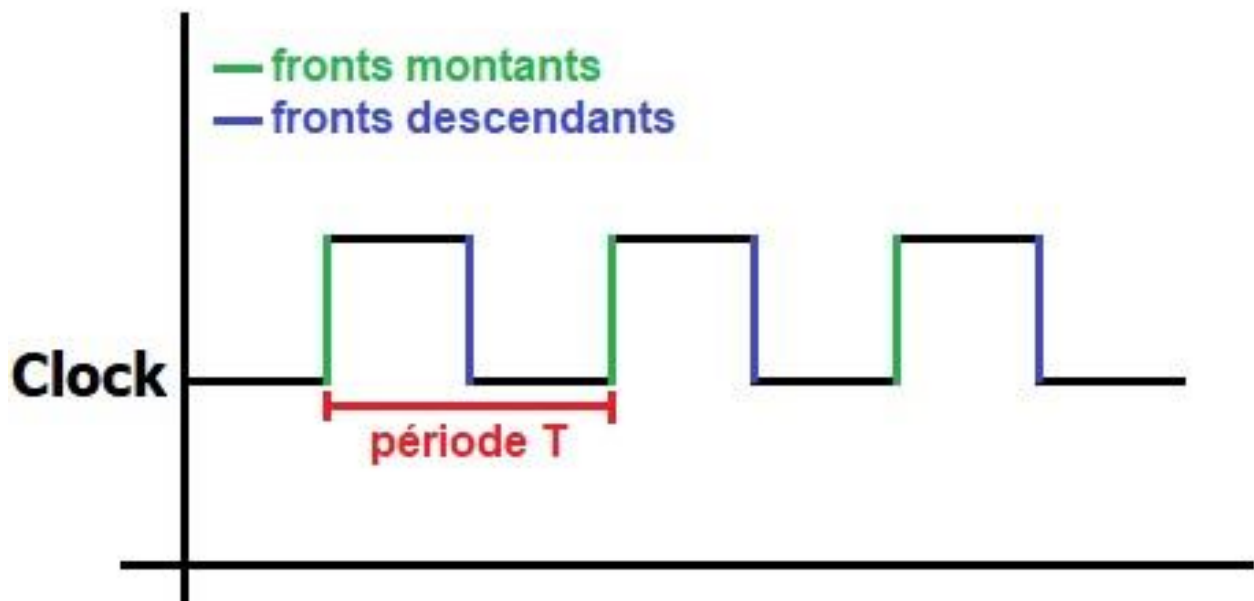


II – Les horloges :

Dans la plupart des systèmes logiques, l'ordre d'apparition des variables peut être crucial. Ainsi, pour respecter des relations de séquentialité importantes, on utilise des horloges pour assurer la synchronisation temporelle des variables logiques. On les utilise aussi pour déclencher des événements.

Une horloge est un système logique qui émet régulièrement une suite d'impulsions calibrées.

L'intervalle de temps entre 2 impulsions représente le temps de cycle ou période T de l'horloge. On définit la fréquence, c'est-à-dire le nombre d'impulsions par seconde, comme étant l'inverse de la période : $f=1/T$. Elle s'exprime en Hertz (Hz).



III - Les bascules :

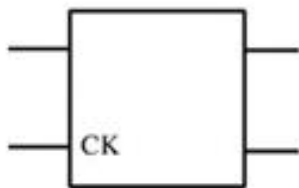
Ce sont les éléments de base de la logique séquentielle, en plus des portes logiques vues précédemment. Nous allons utiliser des bascules synchrones, c'est-à-dire déclenchées par un signal sur leur entrée CK.

1 – Types de synchronisation :

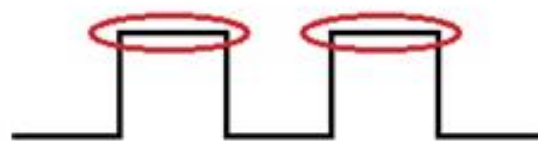
Il existe différentes manières de synchroniser des bascules sur leur entrée CK.

- La bascule peut être synchronisée selon l'état du signal sur l'entrée CK (**level-triggered**) : elle basculera sur l'état haut ou l'état bas de cette entrée :

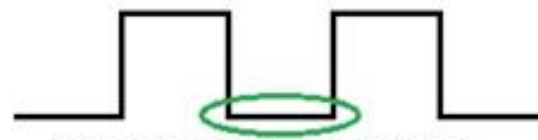
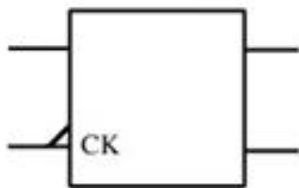
Type d'entrée



Type de synchronisation :



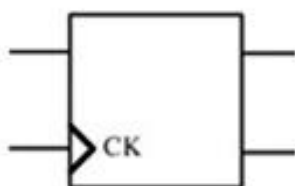
Synchronisation sur état haut



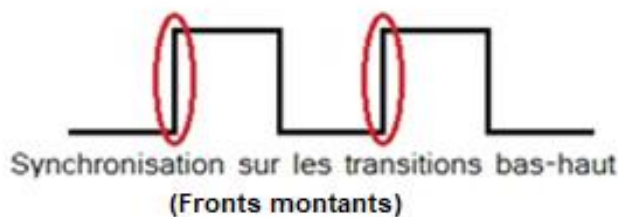
Synchronisation sur état bas

- La bascule peut être synchronisée sur les transitions d'état sur l'entrée CK (**edge-triggered**) : elle basculera sur le passage de l'état haut à l'état bas (quand le signal passe de 1 à 0 : front descendant) ou sur le passage de l'état bas à l'état haut (quand le signal passe de 0 à 1: front montant) :

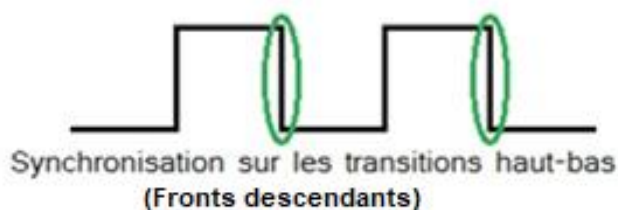
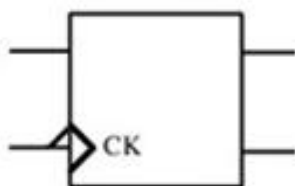
Type d'entrée



Type de synchronisation :



Synchronisation sur les transitions bas-haut
(Fronts montants)

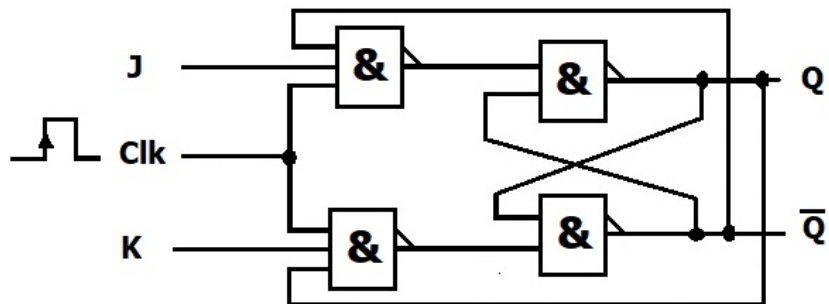


Synchronisation sur les transitions haut-bas
(Fronts descendants)

L'avantage des edge-triggered, c'est qu'elles sont synchronisées sur un événement qui est très bref, à la différence des level-triggered qui se basent sur un état qui dure une demi-période.

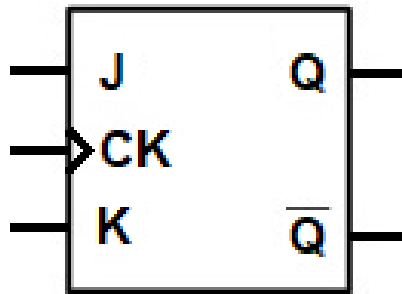
2 - La bascule JK :

Une des bascules fréquemment utilisées, notamment pour créer des circuits compteurs/décompteurs, est la bascule JK. Elle comporte deux entrées J et K en plus de l'entrée CK :



On a là une bascule JK synchrone.

On l'utilisera dans les circuits sous sa forme compacte (ici, une bascule JK synchrone sur fronts montants) :



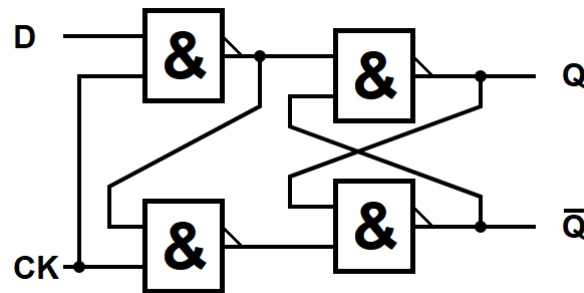
Sa table de vérité est :

J	K	Q_{n+1}
0	0	Q_n
0	1	0
1	0	1
1	1	\bar{Q}_n

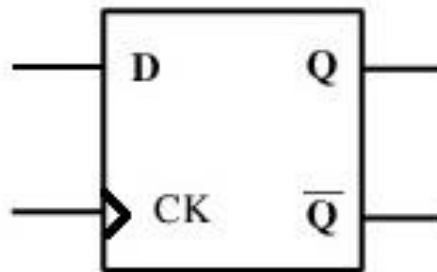
La bascule passera à l'état Q_{n+1} lors de la synchronisation de l'entrée CK (front montant ou front descendant pour une bascule en edge-triggered). Entre 2 synchronisations, elle ne changera pas d'état.

3 - La bascule D :

La seconde bascule la plus utilisée est la bascule D. Elle ne comporte qu'une entrée en plus de l'entrée CK :



On utilisera dans les circuits sa forme compacte (ici, une bascule D synchrone sur fronts montants) :



La table de vérité de la bascule D est très simple :

D	Q_{n+1}
0	0
1	1

La bascule D (Data) est une mémoire de 1 bit : quand elle se synchronise, la valeur en entrée en D sera répercutée sur la sortie Q où elle restera jusqu'à la prochaine synchronisation. Il y a mémorisation de la valeur jusqu'à la prochaine synchronisation.

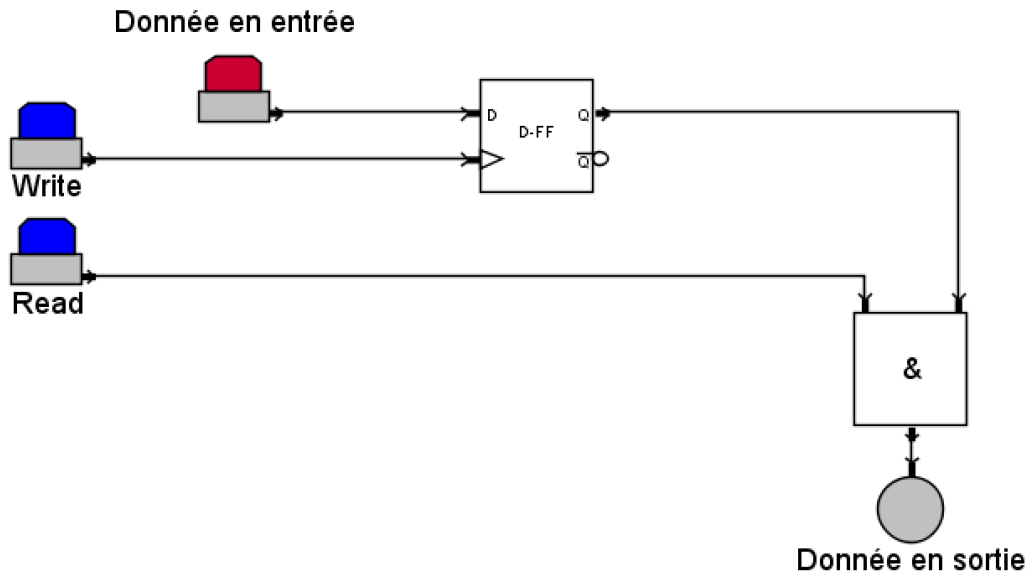
En utilisant plusieurs bascules D, on peut construire des mémoires de grande taille.

Exemple :

Voilà un circuit mémoire de 1 bit. Le bit Read permet de lire le contenu de la bascule. Le bit Write permet d'écrire la donnée en entrée dans la bascule.

Quand Write passe de 0 à 1, il y a un front montant qui entre dans l'entrée CK, ce qui provoque la mémorisation de la donnée en entrée en D sur la sortie Q.

Quand Read est à 1, le résultat de la porte ET est la valeur de Q, sinon, le résultat est 0.



4 - Entrées asynchrones :

Sur certaines bascules, il peut y avoir des entrées dites asynchrones, c'est-à-dire qu'elles agissent indépendamment de l'horloge.

Elles sont utilisées pour forcer la sortie de la bascule à une valeur précise.

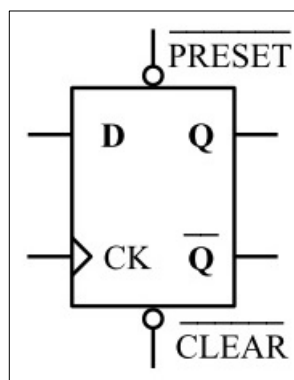
On trouve les entrées :

- RAZ, RESET ou CLEAR qui, lorsqu'elles sont activées, forcent la bascule à 0.
- RAU, SET ou PRESET qui, lorsqu'elles sont activées, forcent la bascule à 1.

Exemple :

Sur cette bascule D, on a 2 entrées asynchrones : CLEAR et PRESET . On constate que ce sont des entrées complémentées. Ainsi, Si l'entrée PRESET est à 0, alors la bascule passera à 1 et si l'entrée CLEAR est à 0 alors la bascule passera à 0.

Les entrées asynchrones permettent de placer facilement un ensemble de bascules dans un état donné et connu.



IV : Les registres :

Un registre est un ensemble de mémoires élémentaires comme les bascules D.
Un registre de n bits comporte n bascules.

Il existe 2 types de registres :

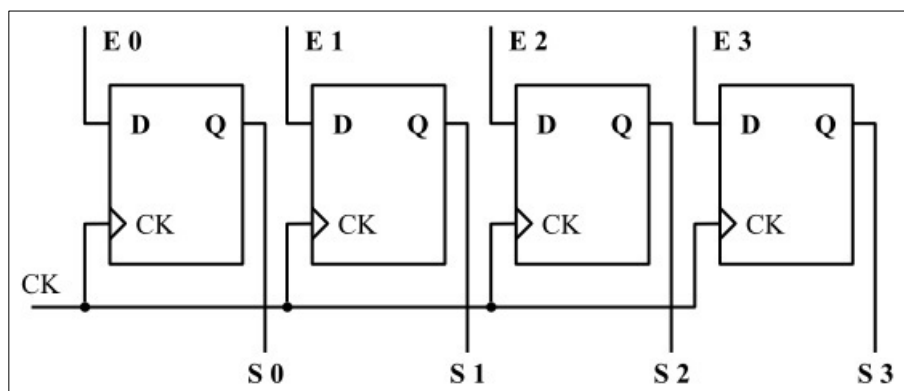
1 - Les registres à cellules non interconnectées :

Ils sont utilisés pour le stockage de données. L'action sur l'entrée CK se fait simultanément sur toutes les bascules du registre.

On les trouve notamment dans le processeur (voir cours d'assembleur en deuxième année).

Exemple :

Un registre mémoire de 4 bits :



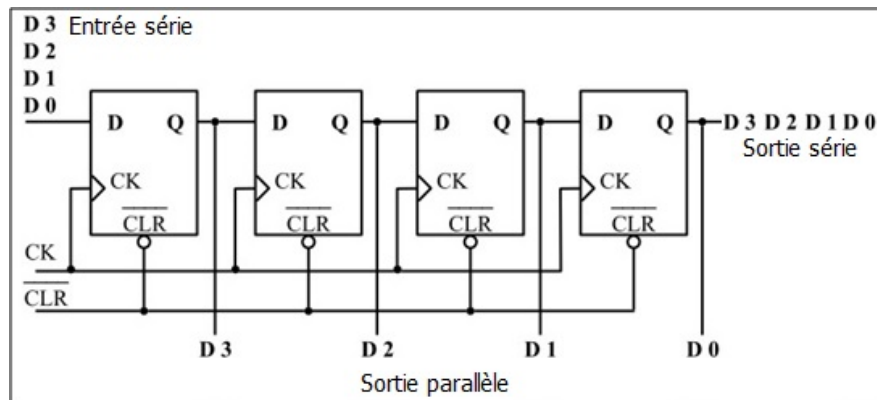
A l'arrivée d'un front montant, simultanément, sur les entrées CK, les données en E_n sont envoyées en S_n . Elles seront accessibles en S_n jusqu'à ce qu'un nouveau front montant se produise sur l'entrée CK.

2 – Les registres à cellules interconnectées :

Ce sont des registres permettant d'effectuer des décalages sur des données, ou de contrôler le mode de circulation des données.

Exemple :

Un registre à conversion série/parallèle (**Serial-In Parallel-Out**) et série/série (**Serial-In Serial-Out**) :



Les données D0, D1, D2 et D3 sont envoyées en série (à la suite les unes des autres) et ressortent

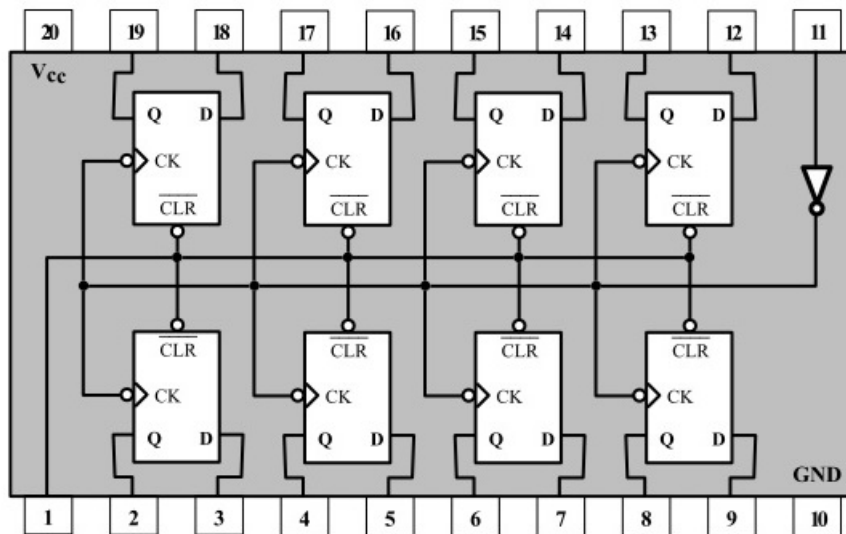
- En parallèle, chacune sur une sortie, après 4 actions sur CK (en bas du circuit)
- En série les unes à la suite des autres (en haut à droite du circuit)

On trouve des registres à écriture et lecture parallèle (PIPO), des registres à écriture parallèle et lecture série (PISO), des registres à écriture série et lecture parallèle (SIPO) et enfin des registres à écriture et lecture série (SISO).

Il existe aussi des registres permettant de décaler des données à gauche ou à droite afin d'effectuer des calculs mathématiques.

3 - Exemple de registre :

Voilà un exemple de circuit intégré contenant un registre 8 bits, le circuit 75273 :



On a différentes sortes de pattes :

- Les pattes en entrée, reliées aux entrées D des bascules : 3, 5, 7, 9, 12, 14, 16 et 18.
- Les pattes en sortie, reliées aux sorties Q des bascules : 2, 4, 6, 8, 13, 15, 17 et 19.
- La patte 11 correspond à l'horloge : synchronisation sur front montant.
- La patte 1 est reliée aux entrées asynchrones CLR des bascules : un 0 sur cette patte mettra toutes les bascules à 0.
- La patte 20 correspond à l'alimentation du circuit.
- La patte 10 correspond à la mise à la terre.

Ce type de circuit est envisageable pour des registres, qui sont de petite taille, mais ne peut être utilisé pour des mémoires de taille plus conséquente.

V - Organisation interne d'une mémoire :

La mémoire présentée sur la page suivante peut stocker 4 mots de 3 bits, soit 12 bits.

Elle possède 3 bits d'entrées de données I_0 , I_1 et I_2 et 3 bits de sortie de données O_0 , O_1 et O_2 .
On stockera et on lira les bits 3 par 3, c'est-à-dire par mot par mot.

On choisira le mot à lire ou à écrire avec les entrées de sélection A_0 et A_1 . On a donc un multiplexeur qui va sélectionner un des mots parmi les 4 possibles.

On a les combinaisons suivantes :

A_1	A_0	Mot sélectionné
0	0	Mot 0
0	1	Mot 1
1	0	Mot 2
1	1	Mot 3

On a aussi 3 bits d'entrée de commande qui vont influencer sur le fonctionnement du circuit :

- CS (Chip Select) : S'il est à 1, le circuit est activé.
- RD (ReaD) : S'il est à 0, les données I_0 , I_1 et I_2 sont écrites dans le mot sélectionné par A_0 et A_1 . S'il est à 1, le contenu du mot sélectionné par A_0 et A_1 est envoyé vers O_0 , O_1 et O_2 .
- OE (Output Enable): S'il est à 1, les sorties O_0 , O_1 et O_2 sont activées et reçoivent les valeurs contenues dans le mot sélectionné par A_0 et A_1 (si on est en mode lecture $RD=1$)

Ecriture :

Les valeurs d'entrée I_0 , I_1 et I_2 sont reliées aux entrées D de chaque mot : I_0 à la bascule correspondant au bit de poids faible du mot, I_1 à la bascule du milieu et I_2 à la bascule correspondant au bit de poids fort du mot.

Pour activer l'écriture dans les 3 bascules formant un mot, il faut envoyer un front montant sur leur entrée CK de manière à ce que les bits aux entrées D des bascules soient mémorisés.

On constate que les entrées CK sont reliées à la sortie d'une porte ET dans laquelle rentre :
 $CS \cdot \overline{RD}$ et un ET entre les valeurs de A_0 et A_1 correspondant au mot.

Par exemple, dans les entrées CK des bascules du mot 2 entrent :

$$CS \cdot \overline{RD} \cdot A_1 \cdot \overline{A_0}$$

Donc pour stocker les valeurs I_0 , I_1 et I_2 qui entrent dans le mot 2, il faut que :

- $CS=1$: on active la mémoire
- $RD=0$: on est en mode écriture
- $A_1=1$
- $A_0=0$

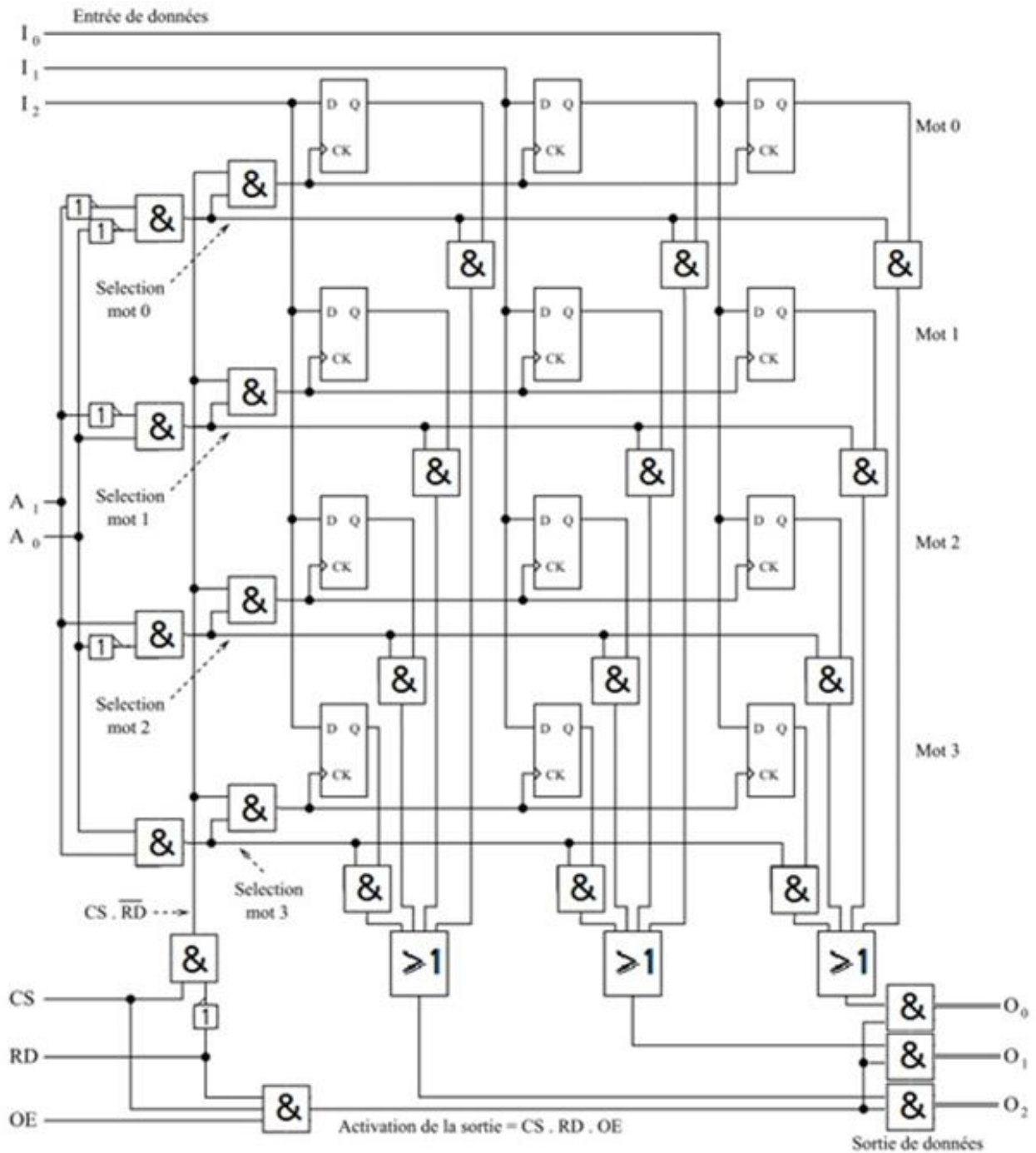
Lecture :

Les sorties Q des bascules sont reliées à une porte ET dans laquelle entre aussi la sortie d'une porte ET où entrent A_0 et A_1 selon les valeurs correspondant au mot sélectionné.

La sortie de cette porte ET est reliée à une porte OU dans laquelle entrent toutes les sorties des bascules de chaque mot correspondant au même bit (poids faible, bit du milieu ou poids fort). Il entrera dans cette porte OU trois zéros et le bit correspondant au mot sélectionné qui sortira donc cette porte. Cette sortie entre alors dans une porte ET avec $CS \cdot RD \cdot OE$. La sortie de cette porte ET est le bit lu dans la mémoire.

Par exemple, pour lire les valeurs stockées dans le mot 1, il faut que :

- $CS=1$: on active la mémoire
- $RD=1$: on est en mode lecture
- $OE=1$: on active la sortie
- $A_1=0$
- $A_0=1$



Ce circuit permet de stocker 12 bits avec un composant qui prendrait 14 pattes (les 11 énoncées précédemment plus 1 patte d'alimentation, 1 patte de mise à zéro et 1 patte de mise à la terre).

Soit beaucoup mieux que le circuit 75273.