Partie III) Logique séquentielle

On distingue deux types de systèmes logiques séquentiels :
- les circuits séquentiels asynchrones, dans lesquels les sorties évoluent dès qu'il y a un changement sur l'une des entrées ;
- les circuits séquentiels synchrones, dans lesquels les sorties ne peuvent évoluer que si un signal d'horloge est actif. Ce dernier peut être actif sur un niveau (0 ou 1) ou sur un front (montant ou descendant).

III.1) Bascules

Les bascules sont les circuits logiques de base de la logique séquentielle. Il existe des bascule asynchrones et des bascules synchrones.

III.1.1) Bascules asynchrones

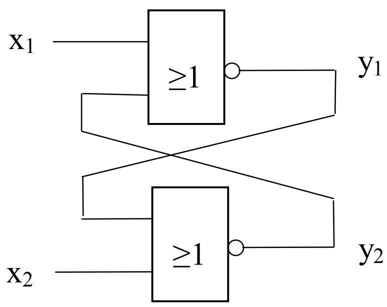
La bascule RS est la plus simple de toutes les bascules. Elle est à la base des autres : la bascule JK et la bascule D, entre autres.

a) Bascule RS

On peut réaliser une bascule RS au moyen de portes NON-OU ou de portes NON-ET.

Bascule RS à portes NON-OU

Etudions le circuit bouclé suivant, à 2 entrées et 2 sorties :



- Le niveau logique 1 étant l'élément absorbant du OU, si x_1=1 et x_2=1, on a forcément y_1=y_2=0.
- Si x_1=1 et x_2=0, on a y_1=0 et donc y_2=1.
- Pour les mêmes raisons, si x_1=0 et x_2=1, on a y_2=0 et donc y_1=1.
- Si x_1=0 et x_2=0, ces 2 entrées n'ont plus aucune influence sur les sorties ; ces dernières restent dans l'état dans lequel elles sont. On a en permanence :

$$y_1 = \overline{y_2} \text{ et } y_2 = \overline{y_1}$$

et on peut constater qu'il s'agit d'un état stable.

Ce dernier état correspond à une mémorisation. Il faut alors distinguer l'instant d'application des nouvelles entrées, qui sera indicé n+1, et l'instant précédent, indicé n. Pour formaliser cette mémorisation, on aura donc :

$$y_1(n+1) = y_1(n) \quad \text{et} \quad y_2(n+1) = y_2(n)$$

(avec $y_1 \neq y_2$).

Rassemblons ces résultats dans un tableau :

x_1	x_2	$y_1(n+1)$	$y_2(n+1)$
0	0	$y_1(n)$	$y_2(n)$
0	1	1	0
1	0	0	1
1	1	0	0

Si l'on considère la sortie y_1 comme la sortie principale, les entrées x_1 et x_2 ont des rôles bien distincts. Si on est en logique positive (dans laquelle le niveau actif est le niveau 1, ce qui est en général le cas), l'entrée x_1 provoque la mise à 0 de la sortie y_1 lorsqu'elle est active ; l'entrée x_2 provoque sa mise à 1. On appelle l'entrée x_1 "R" pour RESET, et l'entrée x_2 "S" pour SET.

On remarque également que les 2 sorties sont complémentaires pour les 2^{e} et 3^{e} lignes de la table de vérité (et donc pour la 1ère ligne également, si l'on y va à partir de la 2^{e} ou de la 3^{e}). On appellera donc la sortie principale Q et l'autre $\overline{Q}$, et on considèrera le 4^{e} cas comme un cas inutilisé (on dit également interdit).

On peut alors ré-écrire la table de vérité sous la forme suivante (sans indiquer la 2^{e} sortie, puisqu'on considère qu'il s'agit de la sortie complémentaire à la 1ère) :

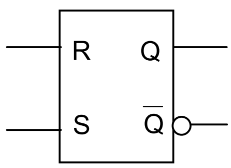
R	S	Q_{n+1}
0	0	Q_n
0	1	1
1	0	0
1	1	non utilisé (0)

On peut considérer cette bascule comme un élément de mémorisation de 1 bit.

Pour résumer, une bascule RS comporte :

- une entrée R (Reset) de mise à zéro,
- une entrée S (Set) de mise à un,
- une sortie Q et son inverse $\overline{Q}$.

Il s'agit d'un bloc fonctionnel logique à part entière ; on peut le symboliser par une boîte noire représentant ses entrées et ses sorties :



Pour déterminer la fonction Q_{n+1}, il faut considérer Q_n comme une des entrées, puisqu'il s'agit d'une variable pouvant prendre les valeurs 0 ou 1. On peut alors établir le tableau de Karnaugh, comme on l'a fait pour les circuits combinatoires :

Q_n \ RS	00	01	11	10
0	0	1	0	0
1	1	1	0	0

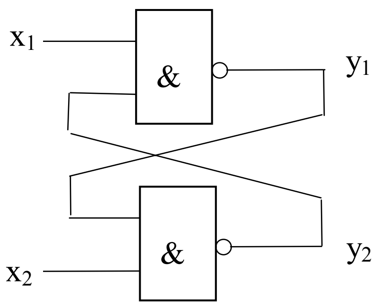
d'où

$$Q_{n+1} = \overline{R}.S + \overline{R}.Q_n$$

$$\leftrightarrow \qquad Q_{n+1} = \overline{R}(S + Q_n)$$

Bascule RS à portes NON-ET

Etudions maintenant le même circuit bouclé mais avec des portes NON-ET :



Le raisonnement à tenir est le même que pour la version à portes NON-OU, sauf que l'élément absorbant est ici le 0.

La table de vérité devient :

x_1	x_2	$y_1(n+1)$	$y_2(n+1)$
0	0	1	1
0	1	1	0
1	0	0	1
1	1	$y_1(n)$	$y_2(n)$

Si l'on considère la sortie y_1 comme la sortie principale Q, comme précédemment, la table de vérité devient :

x_1	x_2	Q_{n+1}
0	0	non utilisé (1)
0	1	1
1	0	0
1	1	Q_n

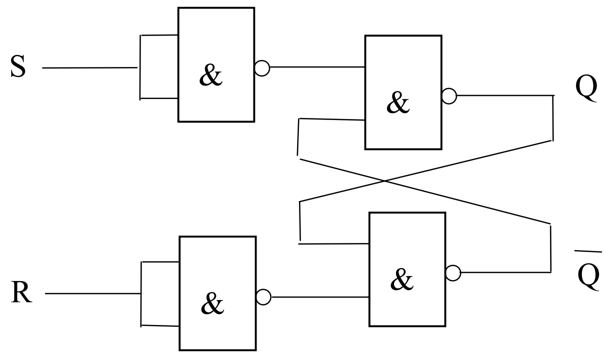
On voit que l'entrée x_1 n'a plus un rôle de mise à 0 mais un rôle de mise à 1 et x_2 un rôle de mise à 0. Pour que cette nouvelle table de vérité corresponde à celle de la bascule RS à portes NOR, il faut donc d'abord complémenter ces 2 entrées, ce qui donne :

$\overline{x_1}$	$\overline{x_2}$	Q_{n+1}
0	0	Q_n
0	1	0
1	0	1
1	1	1 (non utilisé)

Il reste à considérer que :

$$R = \overline{x_2} \quad \text{et} \quad S = \overline{x_1}$$

Le schéma et la table de vérité correspondante deviennent donc :



R	S	Q_{n+1}
0	0	Q_n
0	1	1
1	0	0
1	1	1 (non utilisé)

Un raisonnement similaire à la bascule RS à portes NON-OU mène au tableau de Karnaugh et à l'expression de la sortie Q suivants :

Q_n \ RS	00	01	11	10
0	0	1	1	0
1	1	1	1	0

$$Q_{n+1} = R + \overline{S}.Q_n$$

Aspect asynchrone

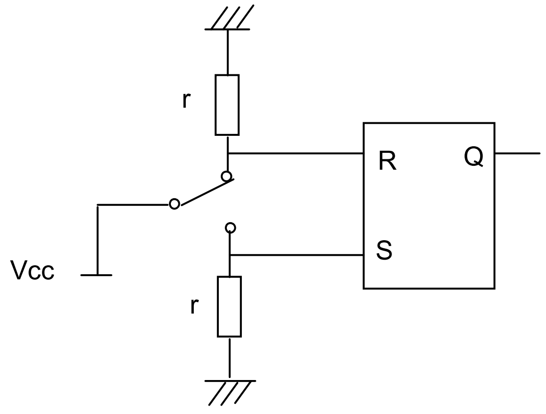
Les 2 bascules RS étudiées ci-dessus sont asynchrones, c'est à dire qu'il n'existe pas de signal par rapport auquel la mise à jour des sorties serait synchronisée ; dès qu'un changement apparaît en entrée, les sorties sont mises à jour immédiatement (plus exactement, après une durée très faible correspondant au temps de propagation des portes).

Exemple d'application : circuit anti-rebonds

Une des applications classiques de la bascule RS est le circuit anti-rebonds. Il s'agit de pallier à un inconvénient d'un commutateur mécanique dans une commande numérique. Le passage d'un niveau logique à l'autre (0 à 1 ou 1 à 0) ne se produit jamais "proprement" ; il y a toujours des rebonds mécaniques, qui font qu'il existe une alternance de niveaux 0 et 1 avant stabilisation (tout ceci se produisant sur quelques ms).

Une solution est donc d'utiliser une bascule RS, qui va se déclencher dès le premier changement d'état, et rester bloquée tant que la commande opposée n'aura pas été effectuée.

On peut avoir par exemple le schéma suivant :



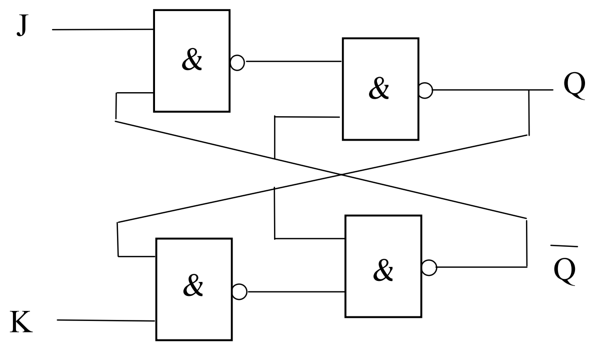
Quand le commutateur est en position haute, R=1 et S=0 donc Q=0. Quand il change de position, on passe par l'état R=S=0, donc on est dans l'état de mémorisation et les sorties ne changent pas. Quand il est en position basse, on a R=0 et S=1, donc Q=1.

On voit que la bascule ne passe jamais par un état indéterminé, et que s'il y a des rebonds, ils ne sont pas gênants puisqu'on reste dans l'état de mémorisation jusqu'à ce que l'autre entrée passe à 1.

b) Bascule JK

La bascule JK est dérivée d'une bascule RS (à portes NON-ET), avec J et K tels que:

$$R = J.\overline{Q} \qquad \text{et} \qquad S = K.Q$$



Par rapport à la bascule RS, l'intérêt est que l'état inutilisé devient utilisable. En effet, à la différence de la bascule RS, si J=K=1, les 2 sorties ne sont pas égales toutes les 2 à 1. Quand on arrive dans l'état J=K=1 à partir de l'un des 3 autres, on a donc forcément les 2 sorties complémentaires.

Par contre, quand au moins l'une des 2 entrées J et K est égale à 0, on retrouve le même comportement que la RS classique, car, par exemple pour K :

$$\overline{\overline{K.Q}.Q} = \overline{(\overline{K} + \overline{Q}).Q} = \overline{\overline{K}.Q + \overline{Q}.Q} = \overline{\overline{K}.Q}$$

(le fait que K soit combiné 2 fois de suite avec Q au moyen d'une porte NON-ET équivaut à une seule combinaison).

Le schéma est alors équivalent à celui de la bascule RS à portes NON-ET.

La table de vérité de la bascule JK est donc :

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

Le tableau de Karnaugh correspondant est :

Q_n \ JK	00	01	11	10
0	0	0	1	1
1	1	0	0	1

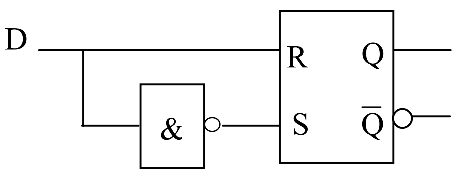
Son équation logique est donc :

$$Q_{n+1} = J.\overline{Q_n} + \overline{K}.Q_n$$

La combinaison d'entrée J=K=1 devient utilisable, mais elle pose un nouveau problème : puisque dans ce cas Q recopie $\overline{Q}$, cette recopie n'a aucune raison de ne se produire qu'une seule fois. En pratique, la sortie Q oscille entre 0 et 1, à la fréquence imposée par le temps de propagation des portes logiques, pendant tout le temps où cette combinaison est présente en entrée.

c) Bascule D

Supposons que l'on contraigne les entrée R et S de la bascule pour qu'elles soient toujours complémentaires, avec R=D et $S = \overline{D}$:



Seules 2 lignes de la table de vérité de la bascule RS sont alors utilisées :

R	S	Q_{n+1}
~~0~~	~~0~~	~~Q_n~~
0	1	0
1	0	1
~~1~~	~~1~~	~~non utilisé~~

On peut la simplifier de la manière suivante :

D	Q_{n+1}
0	0
1	1

On en déduit la relation entrée-sortie de cette bascule :

$$Q_{n+1} = D$$

Sous cette forme, la bascule D n'a pas grand intérêt puisqu'il s'agit d'un bloc fonctionnel qui ne fait que recopier son entrée, en permanence. On verra dans le paragraphe suivant que son utilité apparaît avec un signal supplémentaire dit "d'horloge" (version synchrone).

On peut également réaliser une bascule Dà partir d'une bascule JK, ce qui revient au même au niveau de son fonctionnement. Il suffit de poser $J = \overline{D}$ et K=D.

d) Bascule T

Si l'on relie ensemble les 2 entrées d'une bascule JK, on obtient ce que l'on appelle une bascule T. On limite alors son fonctionnement à 2 lignes de la table de vérité de la bascule JK (la 1ère et la 4e).

On a alors le fonctionnement suivant :

- si T=0, Q_{n+1}=Q_n : état de mémorisation ;
- si T=1, $Q_{n+1} = \overline{Q_n}$: changement d'état.

Ainsi, si T est un signal carré, Q sera également un signal carré, de fréquence 2 fois moindre, synchronisé sur T.

Son équation logique s'écrit :

$$Q_{n+1} = T.\overline{Q_n} + \overline{T}.Q_n = T \oplus Q_n$$

III.1.2) Bascules synchrones

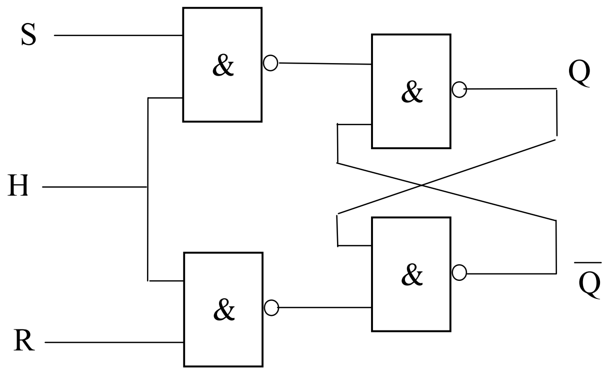
La synchronisation des circuits séquentiels peut se faire de trois façons différentes :

- sur niveau : les changements d'état ont lieu pour un niveau donné (0 ou 1) de l'horloge ;
- sur front : les bascules changent d'état uniquement sur un front montant ou descendant de l'horloge ;
- par impulsion : les bascules changent d'état après deux fronts successifs de l'horloge (front montant puis descendant ou vice versa).

a) Synchronisation sur niveau

Bascule RS synchrone (RSH)

Reprenons notre bascule RS à portes NAND, et conditionnons les 2 entrées à une 3e, appelée H :



Si H=0, cela est équivalent à avoir R=S=0. On est donc dans l'état de mémorisation, et ceci quels que soient les états de R et de S.

Par contre, si H=1, le fonctionnement revient à celui de la bascule RS à portes NON-ET, étudié précédemment.

L'entrée H permet donc de bloquer ou non le fonctionnement de la bascule RS. Il est utilisé comme signal de synchronisation. En général, il est périodique ; on l'appelle signal d'horloge. Il permet de n'autoriser les changements de la sortie qu'à des instants bien précis. On peut donc synchroniser le fonctionnement de cette bascule sur d'autres circuits.

Ici, cette synchronisation s'effectue sur un niveau.

Cette bascule est appelée également RSH ou RST.

Pour obtenir la table de vérité de cette bascule, il suffit de reprendre celle de la bascule RS à portes NON-ET :

$$Q_{n+1} = R + \overline{S}.Q_n$$

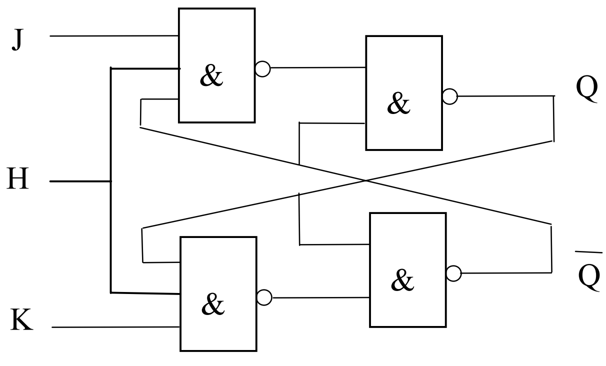
et de remplacer R par R.H et S pas S.H :

$$Q_{n+1} = R.H + \overline{S.H}.Q_n$$

Bascule JK synchrone (ou JKH)

Il s'agit de la bascule JK étudiée précédemment, à laquelle une entrée de synchronisation est ajoutée.

Le schéma du circuit correspondant est le suivant :



On utilise les symboles suivants :

Le symbole de gauche correspond à une synchronisation sur niveau haut (=1), celui de droite sur niveaubas (= 0).

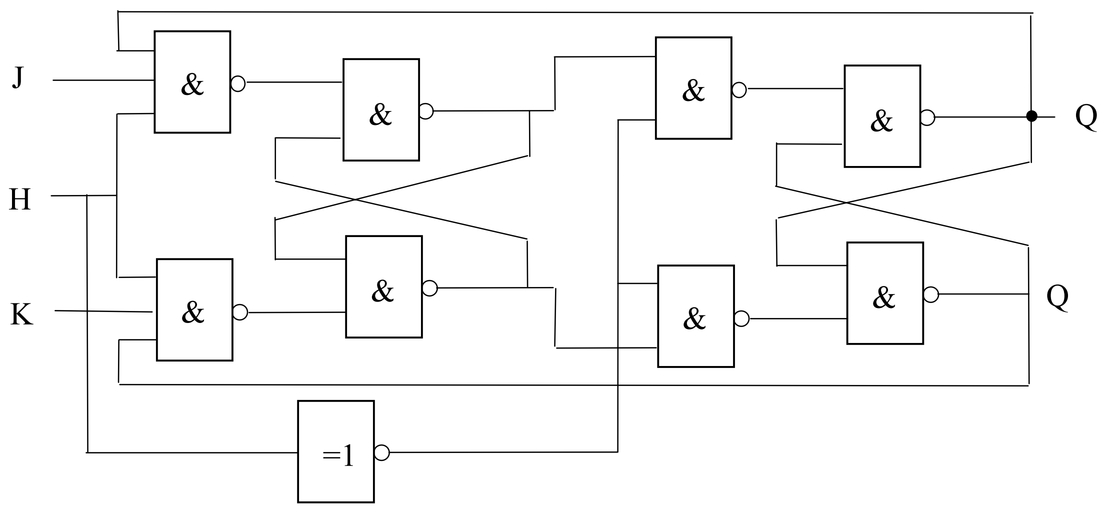
Son équation logique est :

$$Q_{n+1} = J.H.\overline{Q_n} + \overline{K.H}.Q_n$$

On retrouve la limitation de la bascule JK asynchrone pour H=J=K=1 : la sortie Q oscille entre 0 et 1 pendant toute la durée de l'état haut du signal d'horloge.

Pour y remédier, on peut utiliser une bascule JK Maître-Esclave. Elle consiste à utiliser 2 bascule JK en cascade, la 2e recevant comme signal d'horloge le signal d'horloge de la 1ère, complémenté. De plus, le bouclage des sorties de la 2e bascule (en partant de la gauche) s'effectue sur les portes d'entrée de la 1ère. Ainsi, le retour d'information ne s'effectue qu'après un niveau haut suivi par un niveau bas sur H. Cela revient à obtenir un déclenchement sur impulsion.

Le schéma de cette bascule est le suivant :

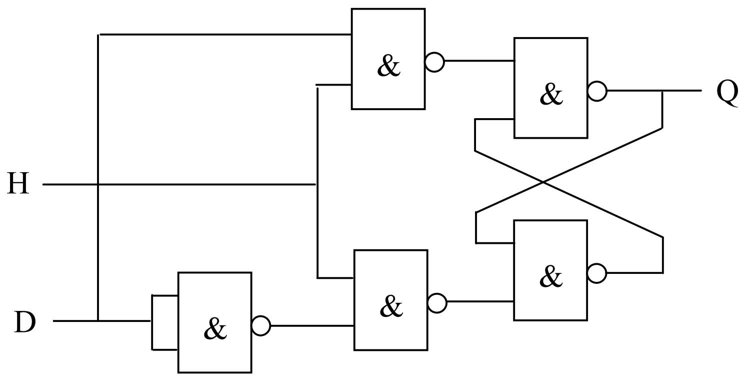


La bascule maître-esclave est donc une solution à l'oscillation existant pour la combinaison J=K=1. Une autre solution consiste à utiliser un déclenchement sur front (voir bascules synchrones, plus loin).

Bascule D à verrou (D latch)

On a vu qu'une bascule D recopiait sur sa sortie Q son entrée D. Avec une entrée supplémentaire d'horloge, on peut synchroniser la recopie sur cette dernière. On prend une bascule RSH, dans laquelle on complémente les entrées.

La bascule recopie l'entrée D en sortie Q quand l'horloge est active, c'est à dire sur niveau haut. Quand l'horloge est inactive, la bascule garde l'état précédent : $Q_{n+1} = Q_n$.



On utilise les schémas de blocs fonctionnels suivants, pour une synchronisation sur niveau haut (gauche) et niveau bas (droite) :

Pour la synchronisation sur niveau haut, l'équation logique associée est :

$$Q_{n+1} = D.H + \overline{H}.Q_n$$

Cette bascule est très utilisée dans les compteurs synchrones.

Bascule T synchrone

Une bascule T synchrone est une bascule T asynchrone possédant une entrée supplémentaire, selon le même principe que les bascules JK et D synchrones.
Son équation logique est donc :

$$Q_{n+1} = (T.H) \oplus Q_n$$

b) Synchronisation sur front

Les bascules synchrones sur front changent d'état uniquement sur un front du signal d'horloge ; en dehors de ces fronts, elle fonctionne en mémoire.
Ce front peut être montant ou descendant. On utilise respectivement les 2 symboles suivants :

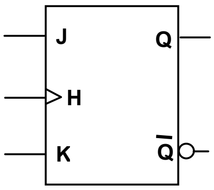
Ce mode de fonctionnement protège d'éventuels parasites sur les entrées car les entrées ne sont prises en compte que pendant la durée d'un front, qui est très courte.

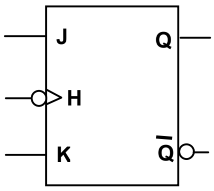
Bascule J K à déclenchement sur front (edge triggered)

La table de fonctionnement d'une bascule JK à déclenchement sur front est la suivante :

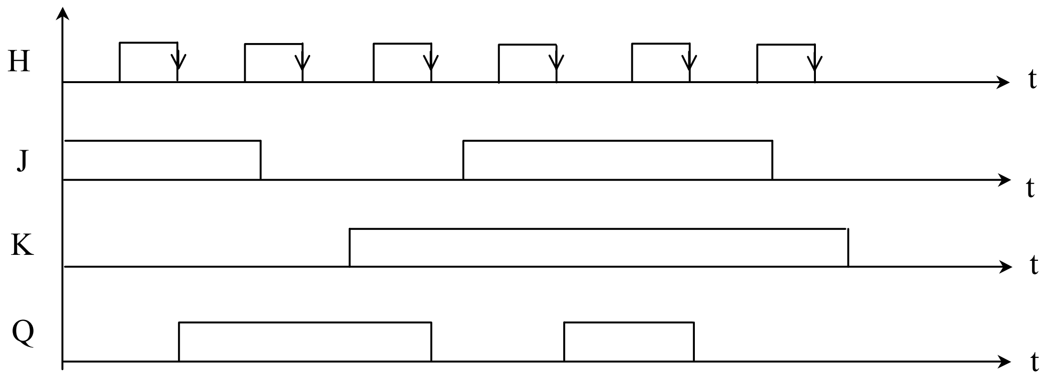
H	J	K	Q_{n+1}
↑ ou ↓	0	0	Q_n
↑ ou ↓	0	1	0
↑ ou ↓	1	0	1
↑ ou ↓	1	1	$\overline{Q}_n$

On utilise les mêmes symboles que pour les bascules à déclenchement sur niveau. Le cercle devant H indique un déclenchement sur front descendant, l'absence de cercle un déclenchement sur front montant :





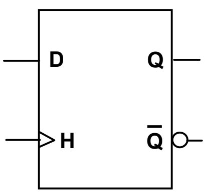
Le chronogramme suivant est un exemple de fonctionnement d'une bascule J K sur front descendant, avec au départ Q=0 :

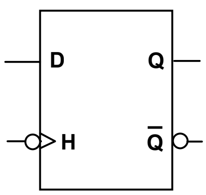


Bascule D à déclenchement sur front (edge triggered)

Dans une bascule D à déclenchement sur front, l'entrée D n'est prise en compte qu'au front montant (ou descendant) de l'horloge.

Les symboles de cette bascule à déclenchement sur front montant et sur front descendant sont respectivement :

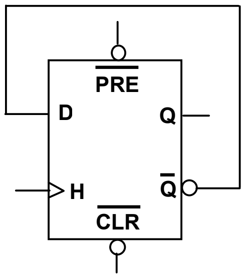




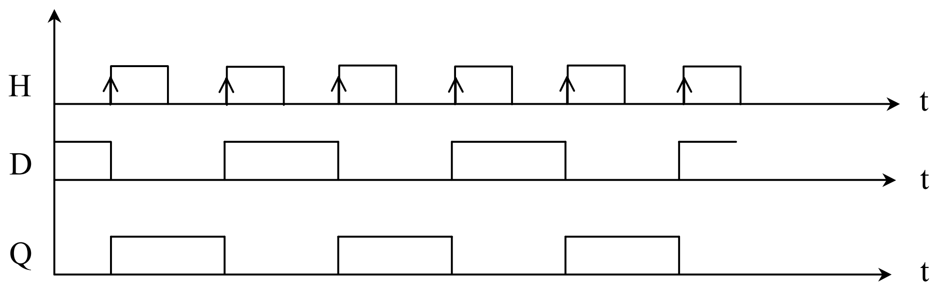
Quand il y a un front actif la sortie recopie l'entrée sinon la sortie garde son état précédent.

Application à la division de fréquence

Considérons le circuit suivant, dans lequel la sortie complémentée est rebouclée sur l'entrée D.



Supposons que Q est au départ à 0. Traçons le chronogramme de ces signaux pour 6 impulsions d'horloge :



Au départ, Q=0 donc $D = \overline{Q} = 1$. Au 1[er] front montant d'horloge, la sortie Q recopie la valeur de D (=1) et $\overline{Q}$ passe à 0. Au 2[e] front montant d'horloge, la sortie Q recopie de nouveau la valeur de D (=0) et $\overline{Q}$ passe à 1, et ainsi de suite.

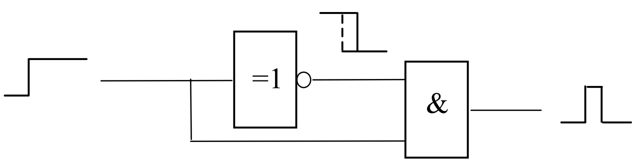
On constate que la sortie Q possède une fréquence 2 fois plus petite que celle de l'horloge. On a donc réalisé un diviseur de fréquence par 2.

Cette propriété est à la base des compteurs.

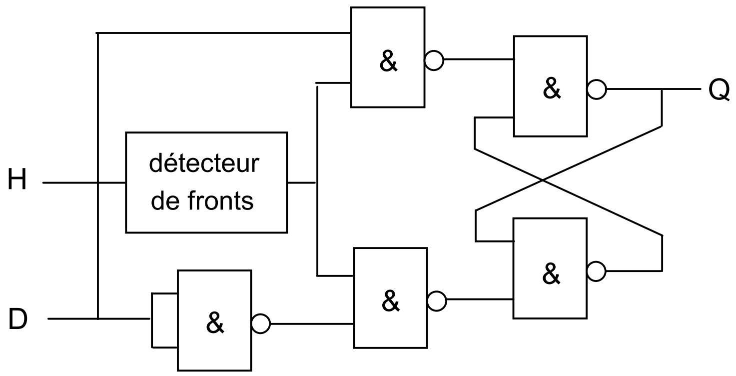
De même, si l'on place les entrée J et K d'une bascule JK à 1, la sortie changera d'état à chaque front actif d'horloge. Le signal en sortie aura une fréquence divisée par deux par rapport à celle de l'horloge.

Détection des fronts

Pour réaliser une synchronisation sur front, il faut utiliser un circuit détecteur de fronts. Le circuit suivant permet de générer une impulsion de courte durée, par exploitation du temps de propagation des portes logiques :



En insérant ce détecteur de fronts en entrée d'une bascule, on transforme une bascule asynchrone en bascule synchrone. Par exemple, pour une bascule D :



c) Entrées de forçage

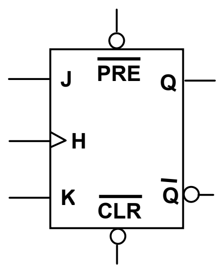
Les entrées de bascules que nous avons étudiées jusqu'ici sont appelées entrées synchrones car les changements des sorties qu'elles provoquent sont synchronisés sur un signal d'horloge H.

La plupart des bascules sont également munie d'entrées asynchrones appelées entrées de forçage ou de pré-positionnement, prioritaires et ne dépendant pas de l'horloge.

Il existe deux types de forçage :

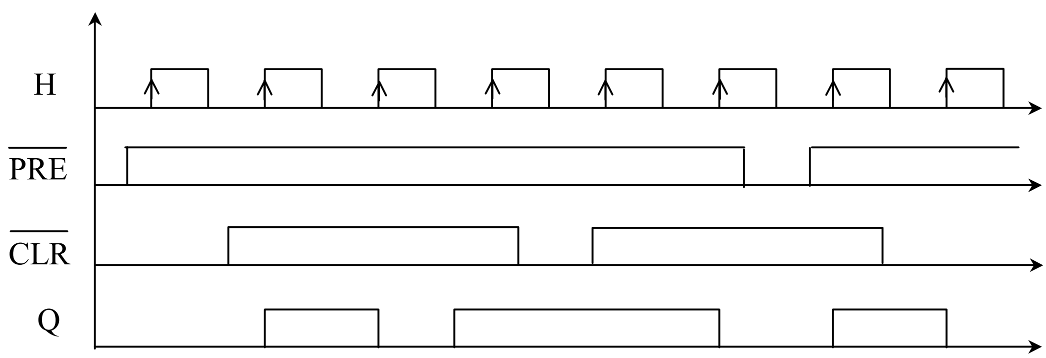
- mise à 1 : SET ou PRESET ou RAU (Remise A Un) ;
- mise à 0 : RESET ou CLEAR ou RAZ (Remise A Zéro).

Sur le schéma des bascules, elles sont en général situées en haut et en bas du bloc fonctionnel, comme le sur la figure suivante :



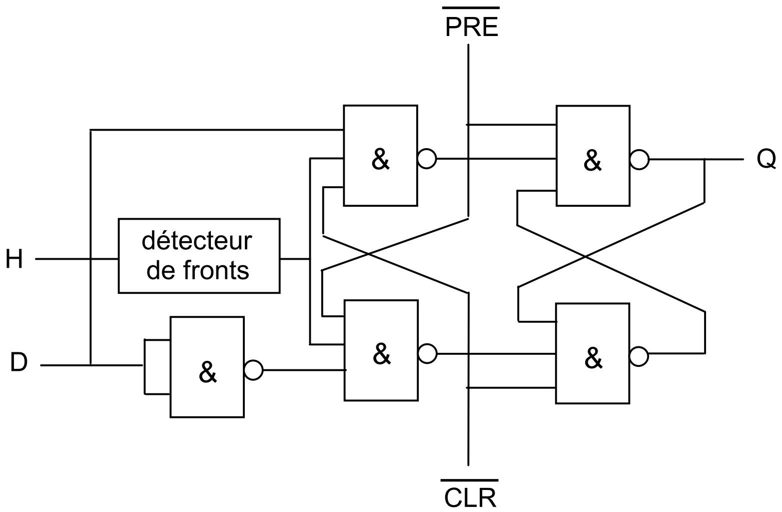
Elles sont généralement actives à l'état bas. Par exemple, un 0 sur l'entrée CLR provoque une remise à zéro de la bascule quelles que soient les valeurs des entrées synchrones. Ces deux entrées asynchrones ne peuvent être actives (=0) en même temps.

Voici un exemple de chronogramme de fonctionnement, pour J=K=1 :



Quand les entrées asynchrones sont inactives (=1), la bascule JK fonctionne en diviseur de fréquence par deux (J=K=1 : la sortie change d'état à chaque front actif de l'horloge).

Ces entrées de forçage sont simplement des entrées supplémentaires des portes situées en sortie. Par exemple, dans le cas d'une bascule D :



La barre sur le PRESET et sur le CLEAR indiquent que ces 2 entrées sont actives au niveau bas.

Quand ces 2 entrées sont au niveau 1, elles sont inactives puisque le 1 est l'élément neutre du ET.

Quand le PRESET est à 0 et le CLEAR à 1, un niveau 1 est imposé sur la sortie Q, et un niveau 0 sur la sortie $\overline{Q}$.

Quand le CLEAR est à 0 et le PRESET à 1, un niveau 0 est imposé sur la sortie Q, et un niveau 1 sur la sortie $\overline{Q}$.

Quand le CLEAR et PRESET sont tous les deux à 0, le fonctionnement de la bascule n'est plus correct puisqu'on a $Q = \overline{Q} = 1$..

d) Tables de transition

La table de transition donne les états dans lesquels doivent se trouver les entrées pour obtenir chacune des 4 transitions possibles de la sortie Q. Quand l'état de l'entrée considérée est indifférent (0 ou 1), on le remplace par une croix.

Bascule JK

Rappelons la table de vérité d'une bascule JK :

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

Par exemple, pour obtenir la transition 0→1 d'une sortie Q, il faut avoir

- J=K=1 qui inverse l'état de la bascule, ou
- J=1 et K=0 qui force la sortie de la bascule à 1.

Dans les 2 cas, il faut J=1, quel que soit l'état de l'entrée K. Pour cette transition, on doit donc mettre l'entrée J à 1, l'état de K étant indifférent.

De la même manière, on détermine la table de transition complète :

Q_n	Q_{n+1}	J	K
0	0	0	x
0	1	1	x
1	0	x	1
1	1	x	0

Par exemple, pour réaliser un diviseur de fréquence par 2, on veut que la sortie de la bascule change d'état à chaque front actif d'horloge. On cherche les valeurs de J et K pour lesquelles on à les transitions 0→1 et 1→0.

Dans les 2 lignes correspondantes de la table de transition, les 2 entrées ne sont jamais spécifiées simultanément ; il est donc possible de choisir, pour simplifier, l'égalité des 2 entrées :

$$J=K$$

Bascule D

Rappelons la table de vérité d'une bascule D :

D	Q_{n+1}
0	0
1	1

La détermination de sa table de transition utilise la même méthode que pour la bascule JK, mais elle est plus simple dans la mesure où il n'y a pas vraiment d'état de mémorisation. Il suffit donc de recopier les valeurs de la sortie Q_{n+1} pour obtenir les valeurs correspondantes de l'entrée :

Q_n	Q_{n+1}	D
0	0	0
0	1	1
1	0	0
1	1	1

Par exemple, pour obtenir un diviseur par 2, on veut que la sortie Q change d'état à chaque front actif d'horloge. Ceci correspond aux 2^e et 3^e lignes de la table de transition. Pour ces 2 lignes, on constate que l'on a $D = \overline{Q_n}$. Il suffit donc de relier la sortie $\overline{Q_n}$ à l'entrée D.