Cours AU3 :

SYSTEMES LOGIQUES SEQUENTIELS

Octobre 2002

Systèmes séquentiel

I. Définition

II. Bascule

- **1.Bascule R S**
- **2. Bascule RST**
- **3. Maître esclave**
- **4.Bascule D MS**

II COMPTEUR - DECOMPTEUR

- **1. Compteur BCD**
- **2. Fréquencemètre**

III REGISTRE A DECALAGE

IV MEMOIRES

- **1. Principe**
- **2. Brochage**
- **3. Association de boîtiers mémoire:**

GRAFCET

I – introduction

II – définitions

III - exemple simple

IV - règles d'évolution

V - Configurations courantes

VI -Cas génériques

1 - priorité

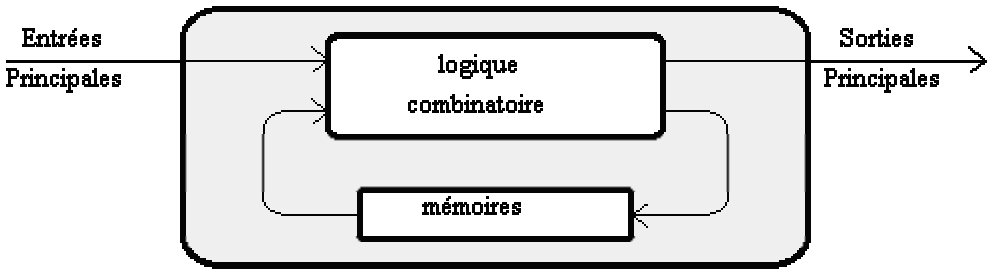
2 - travail à la chaîne

3 - ressource (ou sémaphore)

Systèmes séquentiel

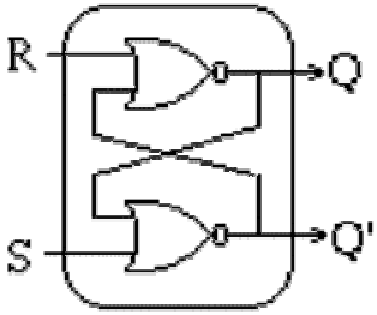
I. Définition

En combinatoire, pour chaque combinaison des entrées, il existe une et une seule combinaison des sorties. En séquentiel, l'état des sorties dépend en plus de l'histoire (de l'état précédent, qui lui aussi, dépend de l'état qui l'a précédé...)



II. Bascule

1.Bascule R S



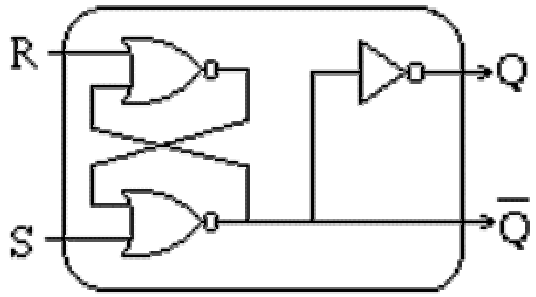
Ce composant est le composant de base du séquentiel. Analysons l'état des sorties dans différents états.

Dans cette table de vérité, on considérera un déroulement séquentiel : les combinaisons des entrées se suivent dans le même ordre

Rappel : la sortie d'une porte NOR ne vaut 1 que quand toutes ses entrées sont à 0.

S	R	Q	Q'	
1	0	1	0	Set (allumer)
0	0	1	0	Mémorisation
0	1	0	1	Reset (éteindre)
0	0	0	1	Mémorisation
1	1	0	0	Etat Interdit !

Si le dernier cas n'est pas utilisé (on ne demande pas simultanément d'allumer et d'éteindre), Q' vaut toujours l'opposé de Q, on l'appellera donc /Q (Q barre)



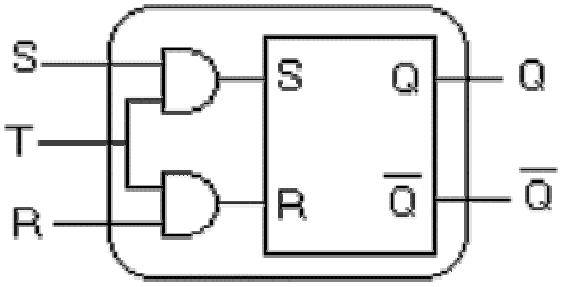
On réalise ici une bascule à enclenchement prioritaire (idem, excepté si S=R=1, Q et mis à 1). Ici, dans tous les cas, Q' est l'opposé de Q. Un bascule à priorité déclenchement aura également le même comportement qu'une RS, excepté dans l'état interdit où /Q vaudra 1

Application : **circuit anti rebond**

Un capteur ne peut pas passer de manière parfaite (sans aléa) de l'état 0 à l'état 1. On peut utiliser une bascule, qui mémorisera l'état stable précédent pendant l'état transitoire. (T1)

Exercice : quel est le comportement d'un même composant où l'on a remplacé les NOR par des NAND ? (réponse : bascule commandée par des niveaux 0)

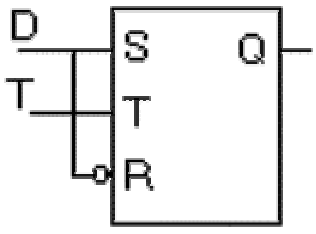
2. Bascule RST



T est l'entrée de validation : si T=1, les entrées R et S sont prises en compte, si T=0 elles ne le sont pas. Dans ce cas, la bascule n'est pas éteinte, elle reste "figée" dans le même état.

Souvent, la bascule comporte deux entrées supplémentaires : Preset (forçage à 1, quel que soit l'état de T) et Clear (forçage à 0), qui permettent de forcer la bascule même si T=0, utilisées généralement pour l'initialisation du composant.

Application : **bascule D** ou Latch ou mémoire : on possède une entrée D, reliée à S d'une RST, et à R par l'intermédiaire d'un inverseur. La sortie Q vaudra l'état lu et mémorisé lors du dernier T=1. C'est le composant de base d'une mémoire d'ordinateur : est mis à 1 ou 0 au moment voulu, figé le reste du temps.

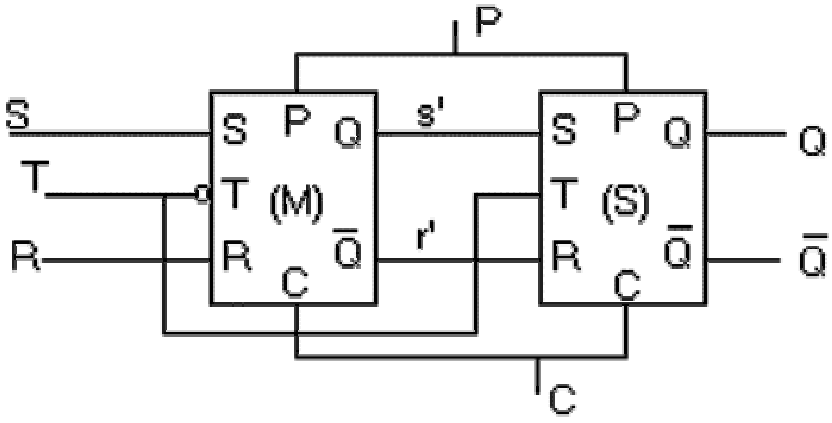


3. Maître esclave

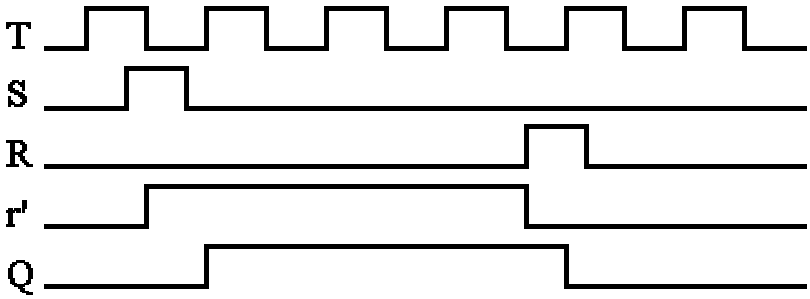
3.1: Fonctionnement

Deux bascules RST sont reliées en série. Une seule est validée à la fois (T inversé). Une entrée Preset permet le forçage à 1 de l'ensemble, une entrée Clear le forçage à 0 (indépendamment de l'état précédent et de T).

Analysons le fonctionnement de cette bascule:



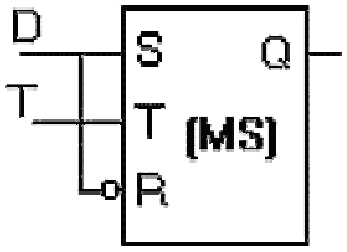
	maître (Master)	esclave (Slave)
Si T=0	information S/R transmise en s'r'	non transmis en Q (ancien Q)
Si T=1	R S en attente (ancien r's')	ancien r's' transmis en Q



On remarque donc que l'information est transmise au <u>prochain front montant</u> de l'horloge T.

4.Bascule D MS

Elle synchronise un signal extérieur sur un front d'horloge (à condition que le signal dure au moins une demi période).

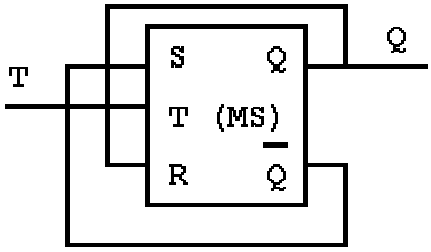


cas particulier : la bascule JK

Elle comporte en général plusieurs entrées de mise à 1 (J) et de mise à 0 (K). Dans le cas où l'on demande l'état impossible (J.K), la sortie est inversée à chaque front d'horloge. Dans les autres cas le fonctionnement est identique. De plus on peut avoir des bascules JK dont le basculement et commandé par un niveau (0 ou 1) ou par un front ("edge triggered").

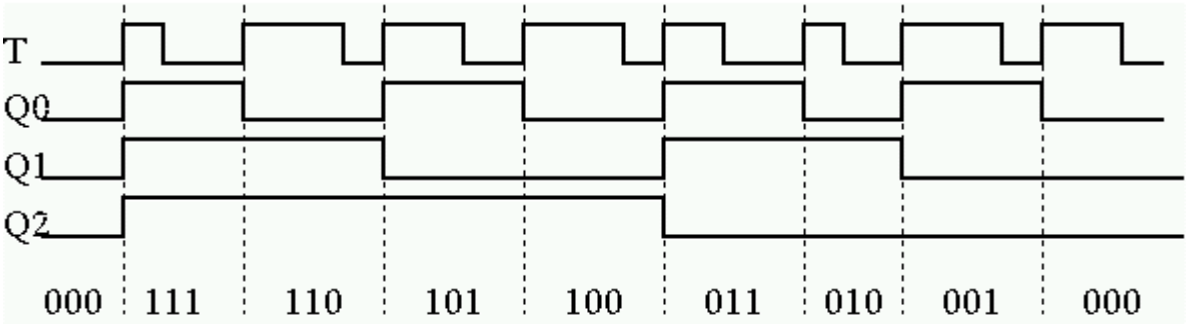
le diviseur de fréquence

A chaque front montant de T, la sortie Q change d'état. Si le signal T est carré, de fréquence F, alors Q sera carré de fréquence F/2. En disposant en série plusieurs diviseurs en cascade, on obtient un compteur ou décompteur binaire (même si T n'est pas régulier) :



II COMPTEUR - DECOMPTEUR

On dispose donc en série des diviseurs de fréquence (trois par exemple), à chaque front appliqué en entrée du premier diviseur, regardons l'état des sorties :



Ce système décompte les fronts (même si les signaux d'entrée ne sont pas régulièrement espacés). En inversant les sorties, on obtient un compteur. En reliant ensemble les Clear des différents étages, on peut remettre le compteur d'impulsions à 0. En général, on initialise le décompteur (par les P et C) au nombre à compter, et on attend la valeur 0. On peut remarquer le binaire se "crée" automatiquement : la base 2 est la mieux adaptée au comptage à l'aide de composants ToR.

Un compteur - décompteur comporte deux entrées, une de comptage (ajoute un) et l'autre de décomptage (soustrait un). Un compteur asynchrone (comme celui-ci) pose un petit problème : les bascules en série ont un temps de réponse qui fait que la nouvelle valeur se "propage" de gauche à droite, on aura donc pendant un très court instant une valeur de

sortie erronnée. Les compteurs synchrones résolvent ce problème (à l'aide de bascules JK, je ne donne pas le schéma, il faut bien que les éditeurs de livres scolaires aient encore quelque chose à vendre).

1) Définition :

Un compteur est un système logique dans le mot binaire en sortie se modifie chaque fois qu'une information est appliquée à son entrée. Il est constitué de n bascules ,il peut donc mémoriser des mots de n bits.
Il peut avoir au maximum 2^n combinaisons différentes. Un compteur binaire est appelé module N s'il peut compter jusqu'à (N-1) la $N^{ième}$ impulsion le remet obligatoirement à zéro.
Les compteurs binaires sont classés en deux catégories :
-Les compteurs asynchrones .
-Les compteurs synchrones.

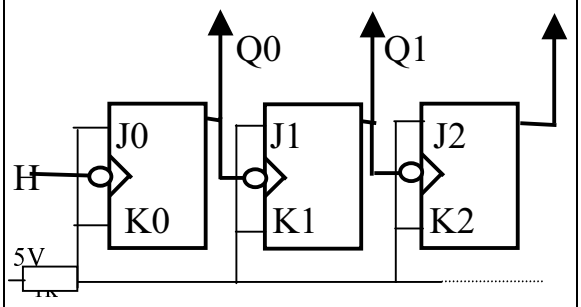
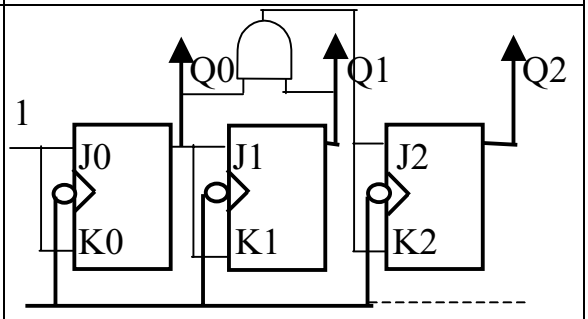
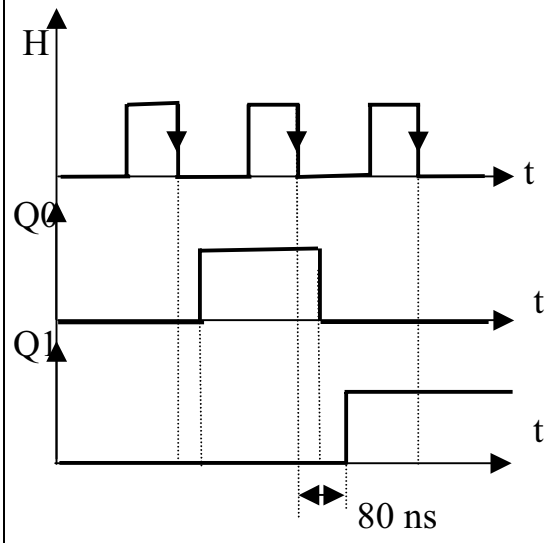
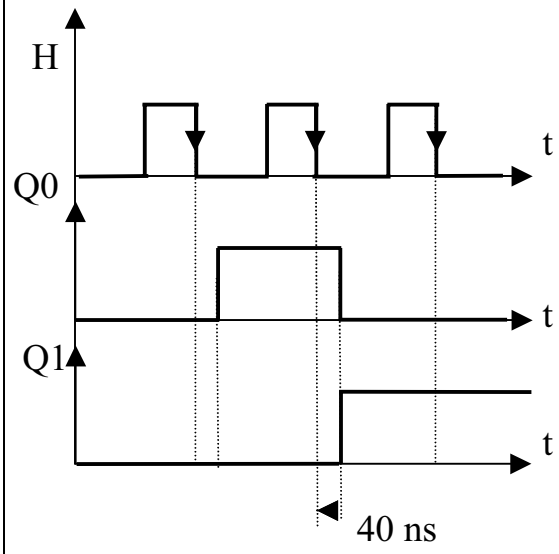
2) Compteurs asynchrones :

Un compteur asynchrones reçoit le signal de comptage H seulement sur l'entrée de la première bascule (premier étage de comptage). Pour les autres bascules constituant le compteur, le signal de comptage d'une bascule de rang (i) n'est autre que le signal de la sortie de la bascule de rang (i-1).
En pratique un compteur asynchrone est réaliser à partir des bascules JK en faisant J=k=1 ou à partir des bascules D .

3) Compteurs synchrones :

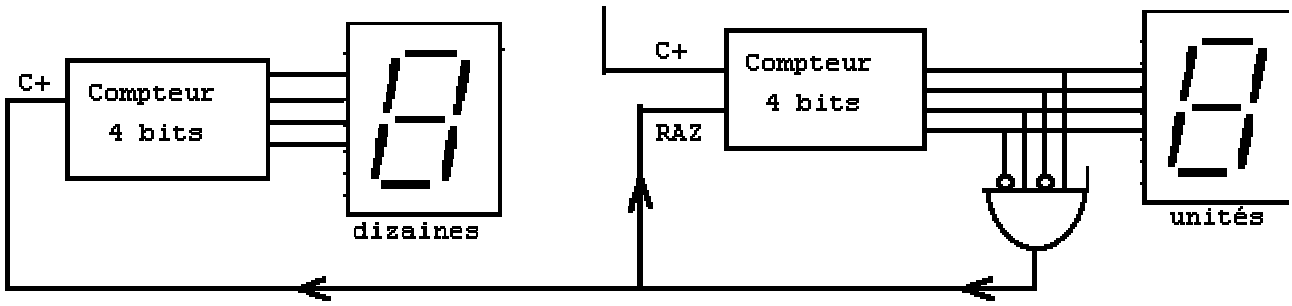
Un compteur synchrone est un compteur dans lequel une horloge commune est appliqué simultanément à toutes les entrées d'horloge des différentes bascules. Ceci supprime les problèmes dus aux temps de propagation des étages d'un compteur asynchrone.

4) Comparaison d'une structure asynchrone et d'une structure synchrone quand à la propagation d'un signal d'horloge :

STRUCTURE ASYNCHRONE	STRUCTURE SYNCHRONE
	
	
D'une bascule à l'autre les temps de propagation s'ajoutent.	Il n'y a pas d'accumulation de retard de propagation..

1. Compteur BCD

Une fois arrivé à la valeur 1010 (10 en binaire), on le remet à 0 et on lance un signal à la dizaine supérieure. Attention dans la pratique ce n'est pas aussi simple : si un composant va plus vite que l'autre, la remise à 0 peut se faire sur une période transitoire où l'on se trouvait à 1010. (C+=entrée de comptage)

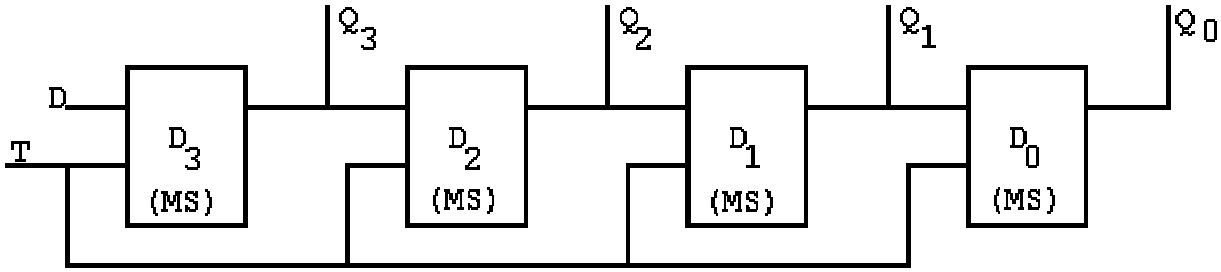


2. Fréquencemètre

Autre application du compteur. C'est un compteur d'impulsions pendant un temps donné (cas des fréquences élevées), ou alors on compte le temps pendant une période (fréquences faibles).

III REGISTRE A DECALAGE

De même en mettant en cascade des bascules D MS, on obtient un registre à décalage, que l'on peut initialiser en parallèle (indépendamment de T) par les broches Preset et Clear.

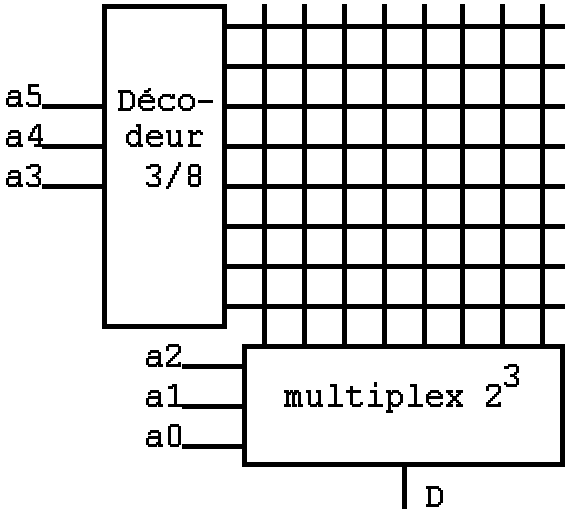


à chaque front de T la valeur Q_i est décalée en Q_{i-1}. Pour un bon fonctionnement, il faut que chaque D_i soit déclenché <u>après</u> D_{i-1}.

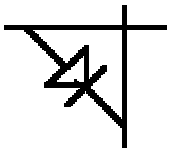
IV MEMOIRES

1. Principe

ROM : Read Only Memory : On a figé par construction le contenu des mémoires. En fait, pour un bit, une mémoire à 1 correspond à une liaison sur l'alimentation, un 0 à une liaison à la masse. Comment regrouper plusieurs bits ? Soit par exemple une mémoire de 64 valeurs binaires :



Les intersections sont : pour une valeur 0 : pas de liaison, pour une valeur 1 : une diode empêchant le courant de remonter de la colonne. :



En entrant une ADRESSE (numéro de mémoire, entre 0 et 63 ici) sous forme binaire, on obtient la donnée désirée (contenu de la mémoire). L'adresse se décompose en une partie haute (a3 à a5) déterminant la ligne mise à 1, et une partie basse (a0 à a2) déterminant la colonne connectée sur la sortie D

PROM : programmable une seule fois : liaison "fusible". On programme une Prom sous une tension supérieure à la tension de fonctionnement.

UVPROM : Prom reprogrammable après régénération sous ultra violets (20 mn).

EEPROM : Prom régénérable électriquement.

Application :Utilisation de ROM en combinatoire

Pour chaque état des entrées on mémorise la sortie (décomposition en mintermes). Par exemple, pour créer un générateur de caractères 8x8 pixels on préférera utiliser une ROM plutôt que de faire un circuit spécifique (pour 127 caractères, il faut 1 Ko).

RAM statique : garde la valeur tant qu'elle est alimentée. On utilise la même disposition que pour la ROM, mais à chaque intersection on place une bascule.

RAM dynamique : d'accès beaucoup plus rapide, mais il faut les régénérer (lire et réécrire) à intervalle régulier (plusieurs fois par seconde). On intègre actuellement plusieurs Mbit par composant.

2. Brochage

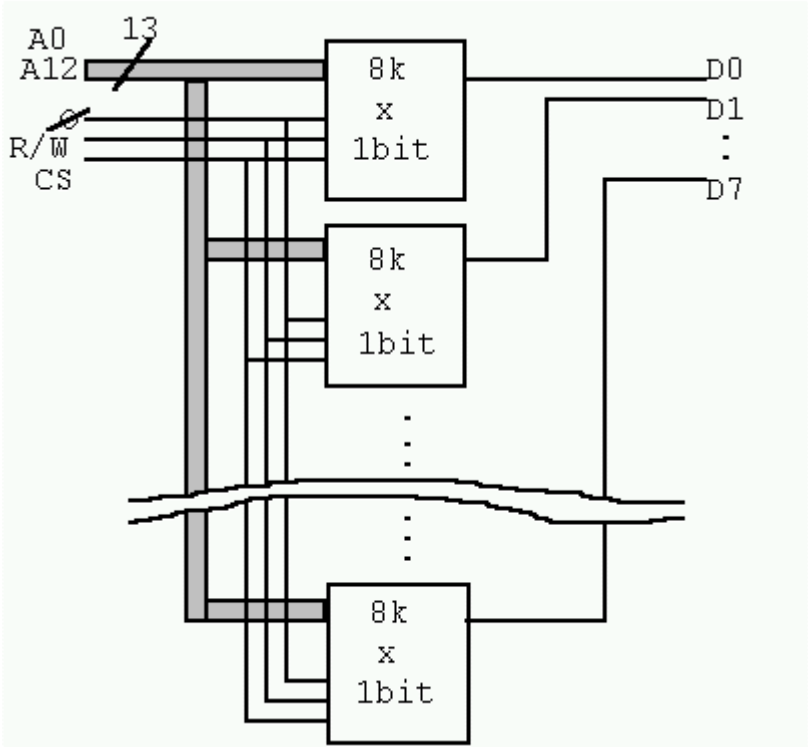
Un boîtier RAM comprend en général des entrées A_0 à A_n permettant de désigner la mémoire, R/W pour dire si lire ou écrire, et D_0 à D_m pour les données (entrée sortie) (ou D si c'est un boîtier de mémoires 1 bit). De plus, le composant ne fonctionnera que s'il est sélectionné (entrée CS : chip select). De plus, il faut entrer un signal de synchronisation (horloge) et évidement l'alimenter.

Chronogramme : en lecture, il faut donner l'adresse, CS, Read, on obtiendra le contenu D au prochain top d'horloge. En écriture, on donne l'adresse, CS et Write, puis la donnée au prochain top d'horloge.

3. Association de boîtiers mémoire:

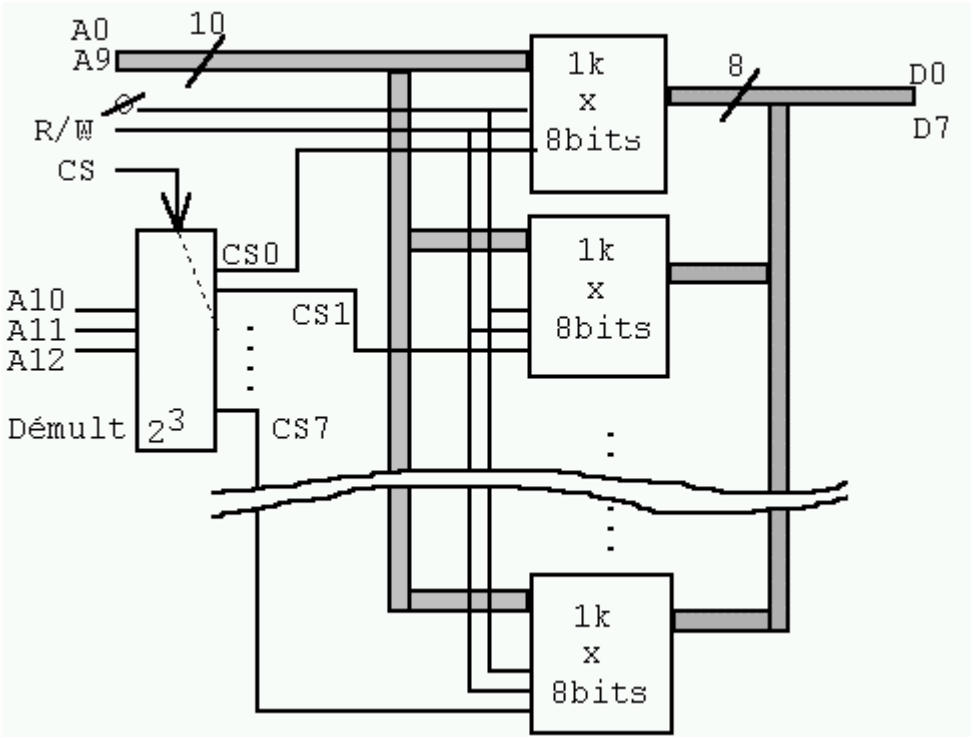
Comment utiliser 8 boîtiers de 8k x 1 bit pour créer une mémoire de 8k x 8 bits ?

En fait, on envoie les signaux de commande et l'adresse aux 8 boitiers. Ceux-ci, simultanément, traiteront les 8 bits du mot désiré. Les différents bits d'une même mémoire ne sont donc pas physiquement situés au même endroit.



Et avec 8 boîtiers de 1k x 8 bits ?

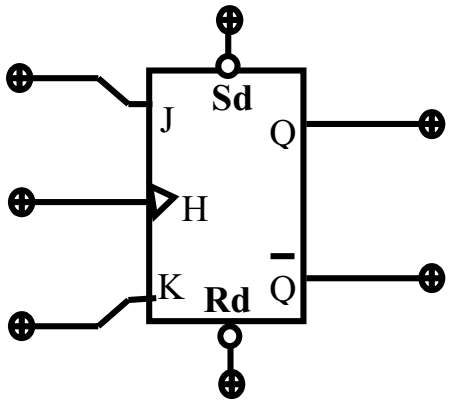
Ici, les 10 bits de poids faible de l'adresse désirée est transmise à tous les boitiers. Mais un seul est sélectionné, suivant les 3 bits de poids fort de l'adresse. Les 8 bits de données de tous les boitiers sont reliés ensemble, on est sûr qu'un seul sera sélectionné à la fois, via le démultiplexeur.



Les compteurs.

Travail demandé :

I/ Etude de la bascule JK :



1) Identifier les différentes entrées.

J=……

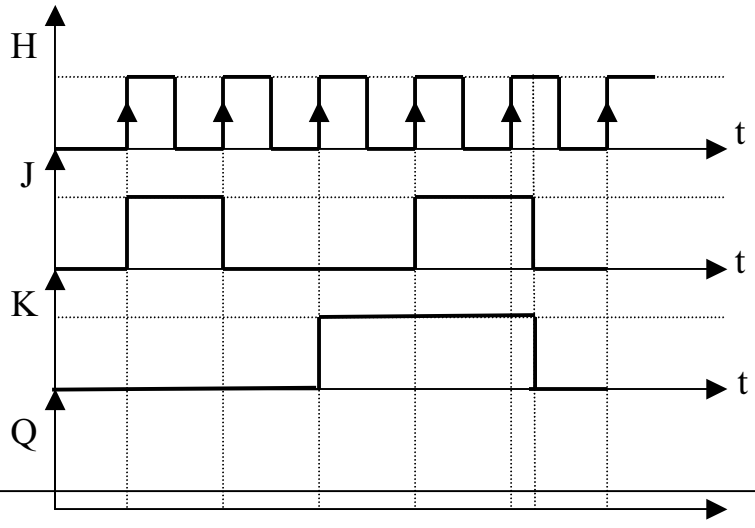
K=……..

H=……...

Sd=…….

Rd=…….

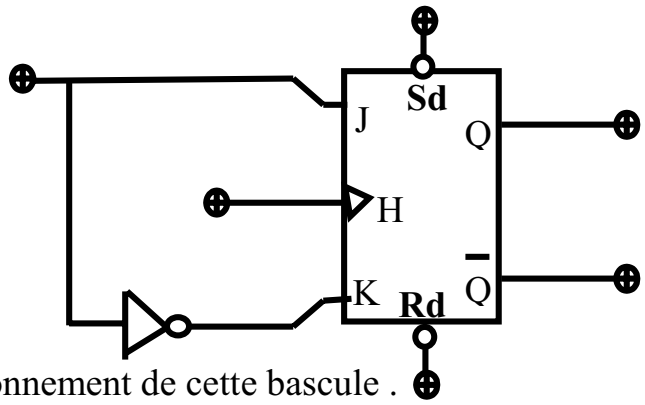
2)Compléter le chronogramme suivant :



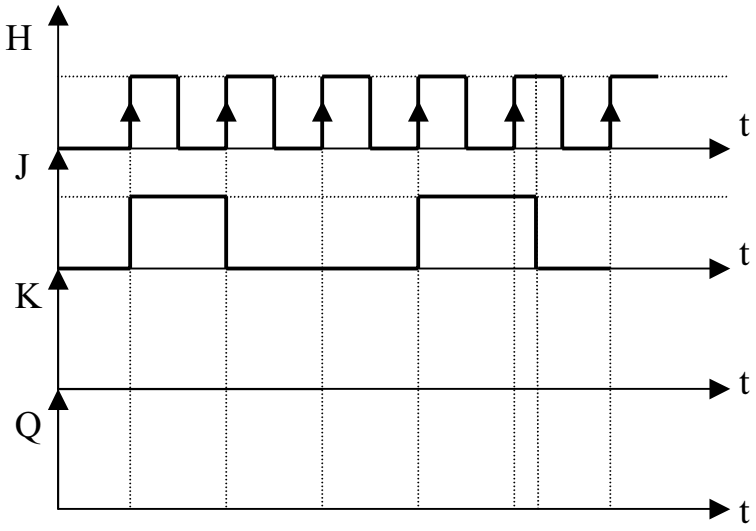
t

3)Soit J =$\overline{K}$:

3-1 Effectuer le schéma suivant.



3-2Simuler le fonctionnement de cette bascule .

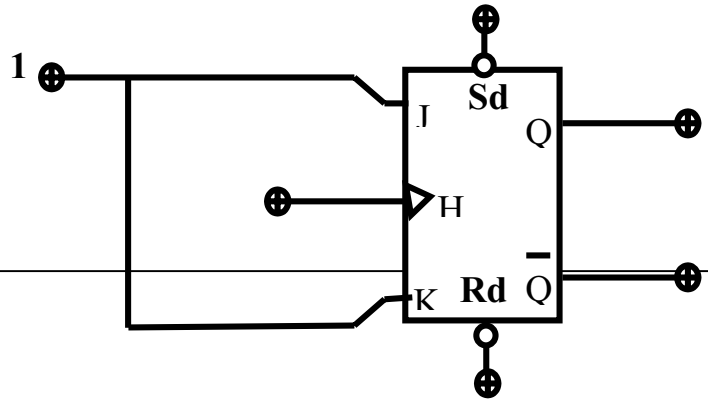


3-3Quel type de bascule s'agit –il ?.

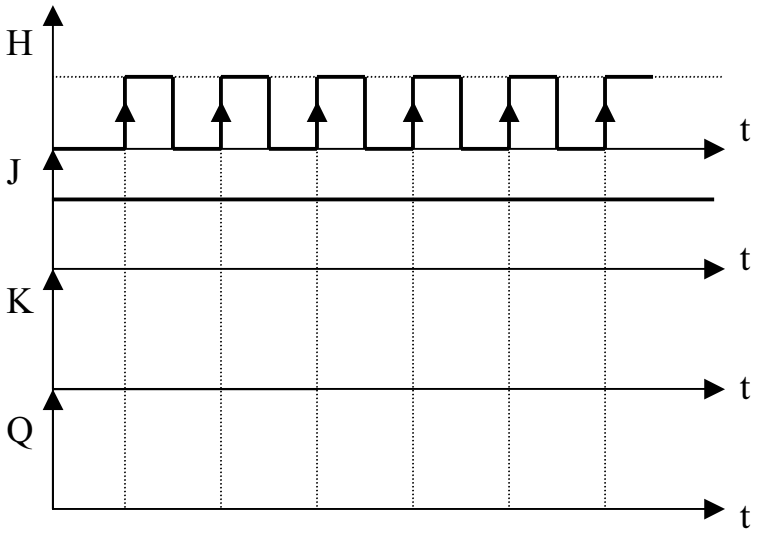
……
……..

4)Soit J=K=1 :

4-1Effectuer le schéma suivant.



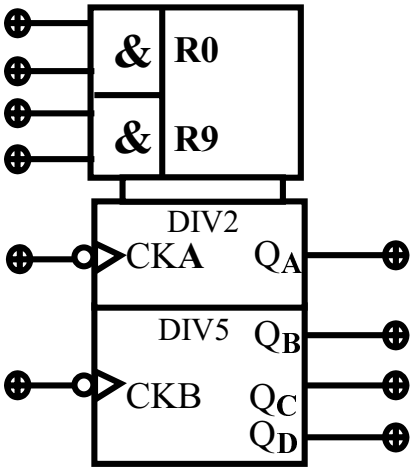
4-2 Simuler le fonctionnement de cette bascule .



4-3Quel type de bascule s'agit –il ?.

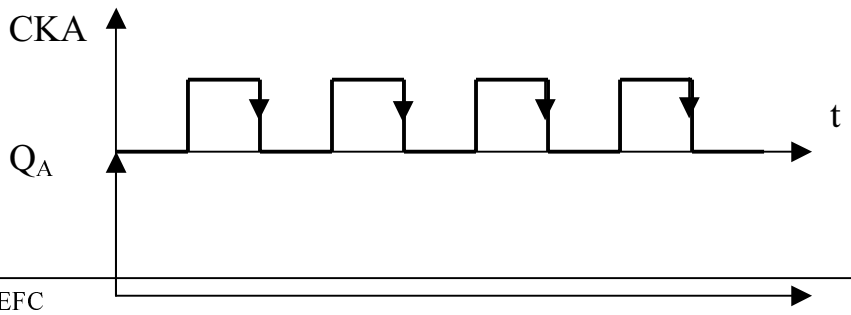
……
……

II/Analyse du compteur :



1) Etude du diviseur par deux :(DIV2)

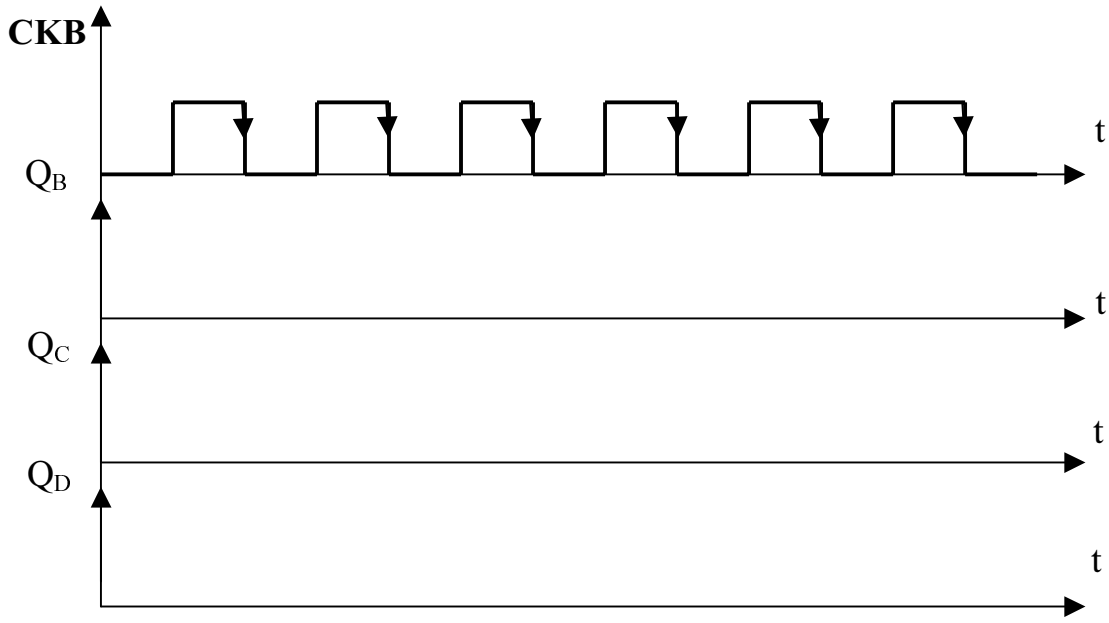
Compléter le chronogramme suivant :



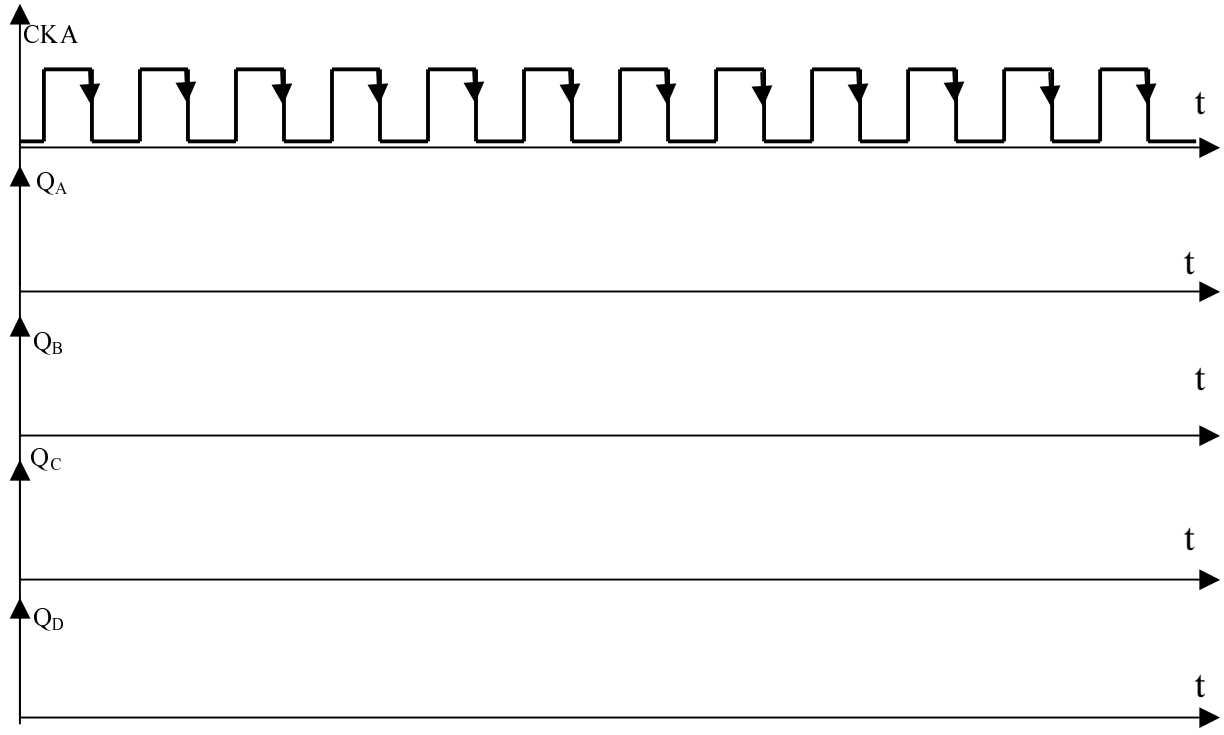
t

2) Etude du diviseur par 5 :(DIV5)

Compléter le chronogramme suivant :



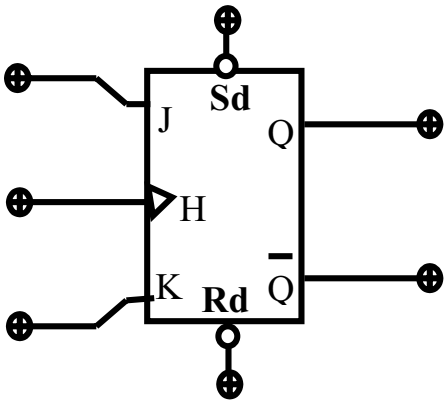
3)Faire le montage de ce circuit de telle sorte qu'on obtient un diviseur de fréquence par 10(modulo 10).



Correction : Les compteurs.

Travail demandé :

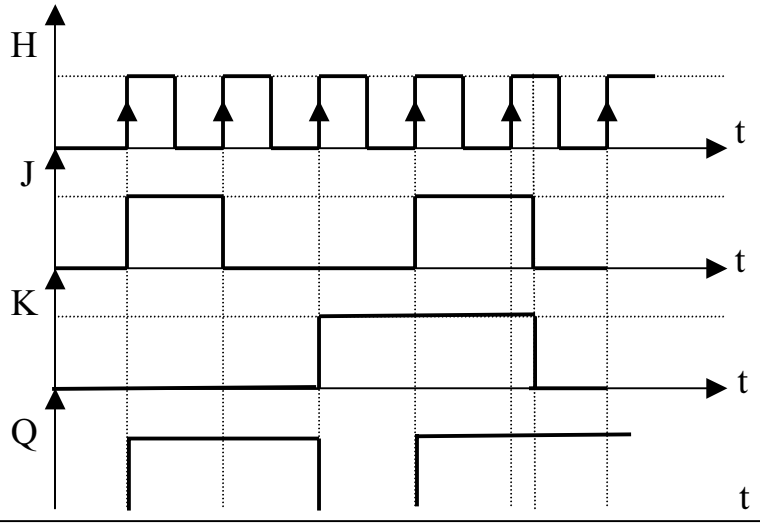
I/ Etude de la bascule JK :



3) Identifier les différentes entrées.

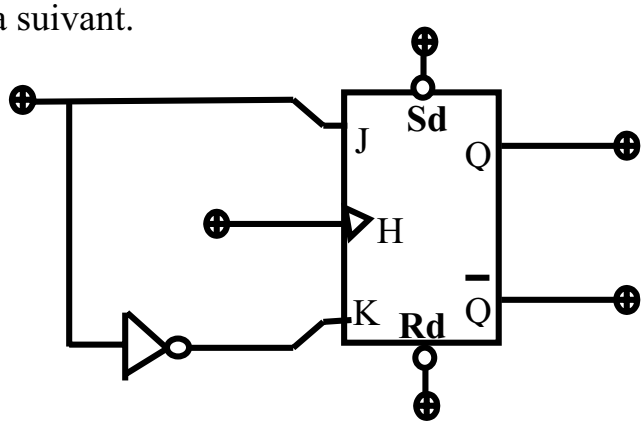
J =Entrée de mise à 1.
K=Entrée de mise à 0.
H=Entrée d'horloge.
Sd=Entrée asynchrone de forçage à 1.
Rd= Entrée asynchrone de forçage à 0.

2)Compléter le chronogramme suivant :

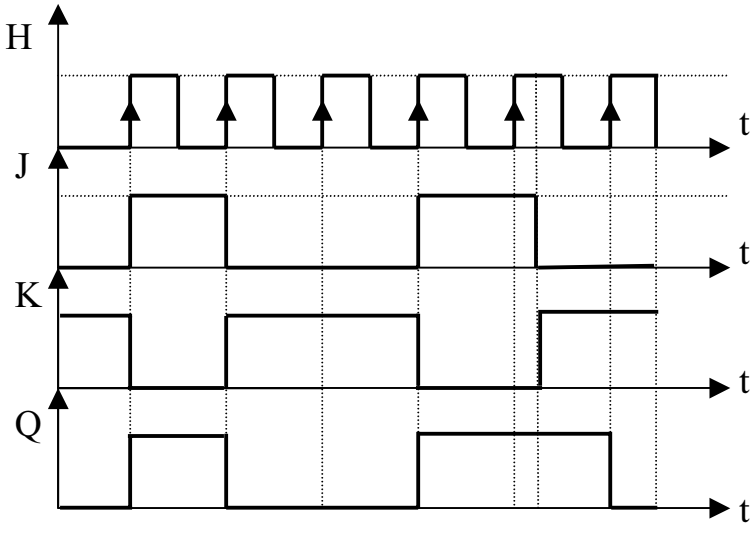


3)Soit J =$\overline{K}$:

3-1 Effectuer le schéma suivant.



3-2Simuler le fonctionnement de cette bascule .

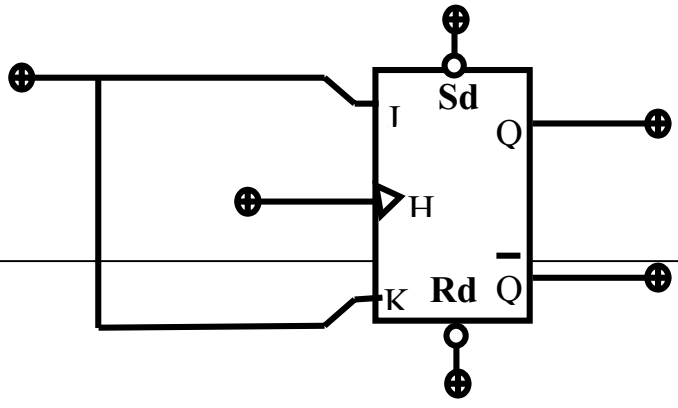


3-3Quel type de bascule s'agit –il ?.

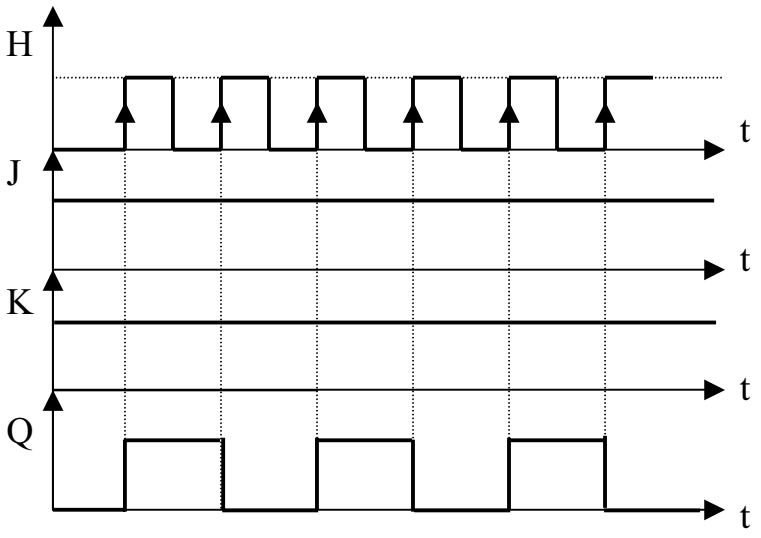
Il s'agit d'une bascule D : La sortie prend la même valeur que celle présente à l'entrée J quand le signal d'horloge effectue une transition.

4)Soit J=K=1 :

4-1Effectuer le schéma suivant.



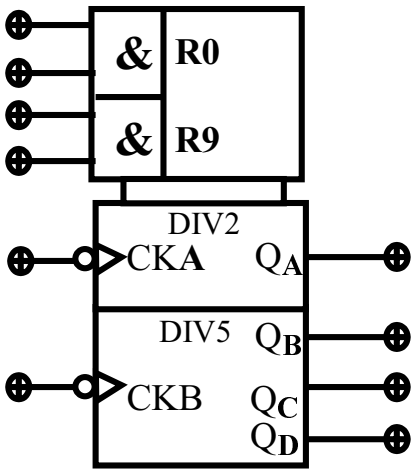
4-2Simuler le fonctionnement de cette bascule .



4-3Quel type de bascule s'agit –il ?.

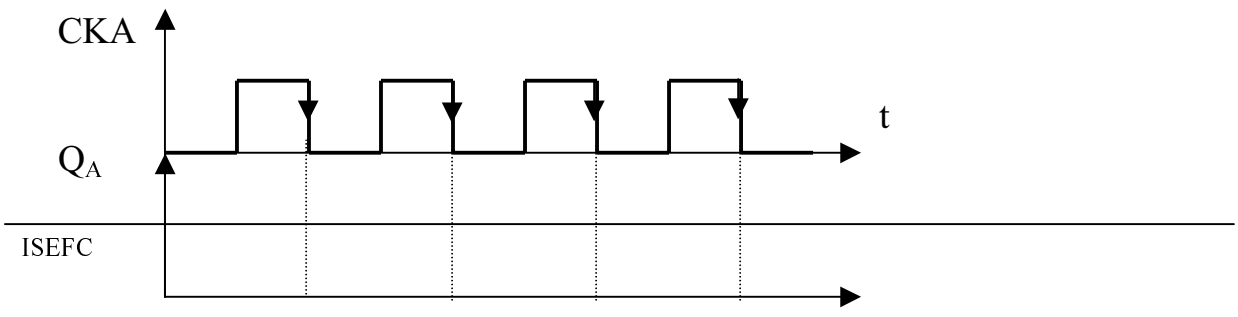
Il s'agit d'une bascule T : La sortie change d'état chaque fois que l'entrée d'horloge passe à l'état 1 et conserve son état le reste du temps.

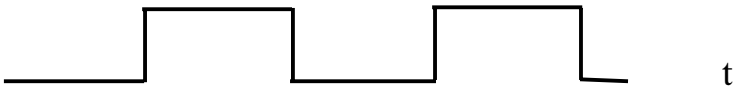
II/Analyse du compteur :



2) Etude du diviseur par deux :(DIV2)

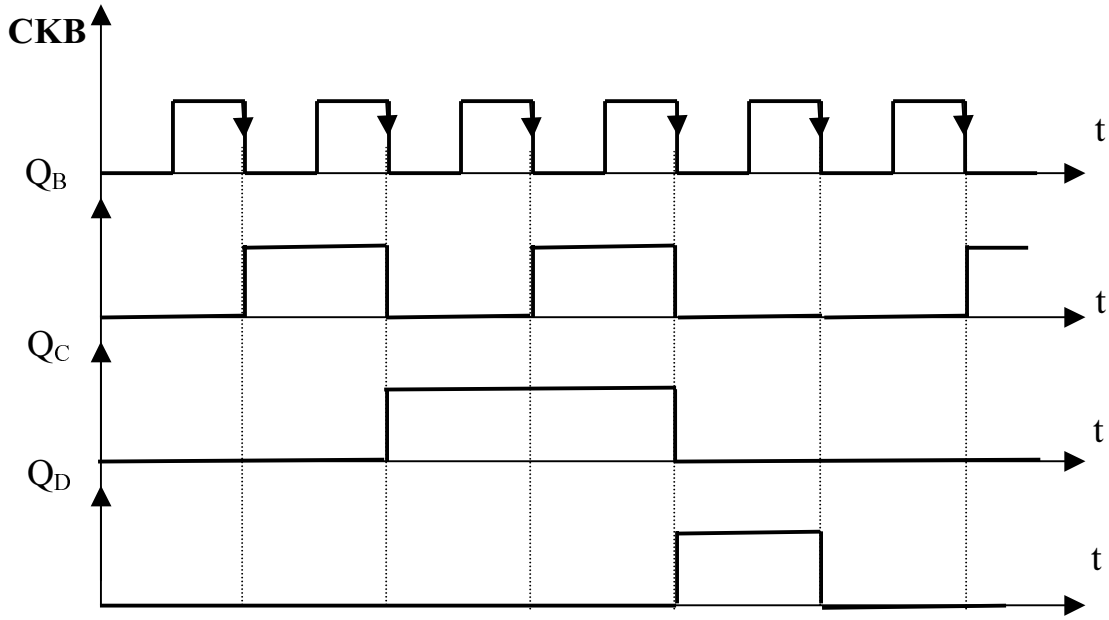
Compléter le chronogramme suivant :



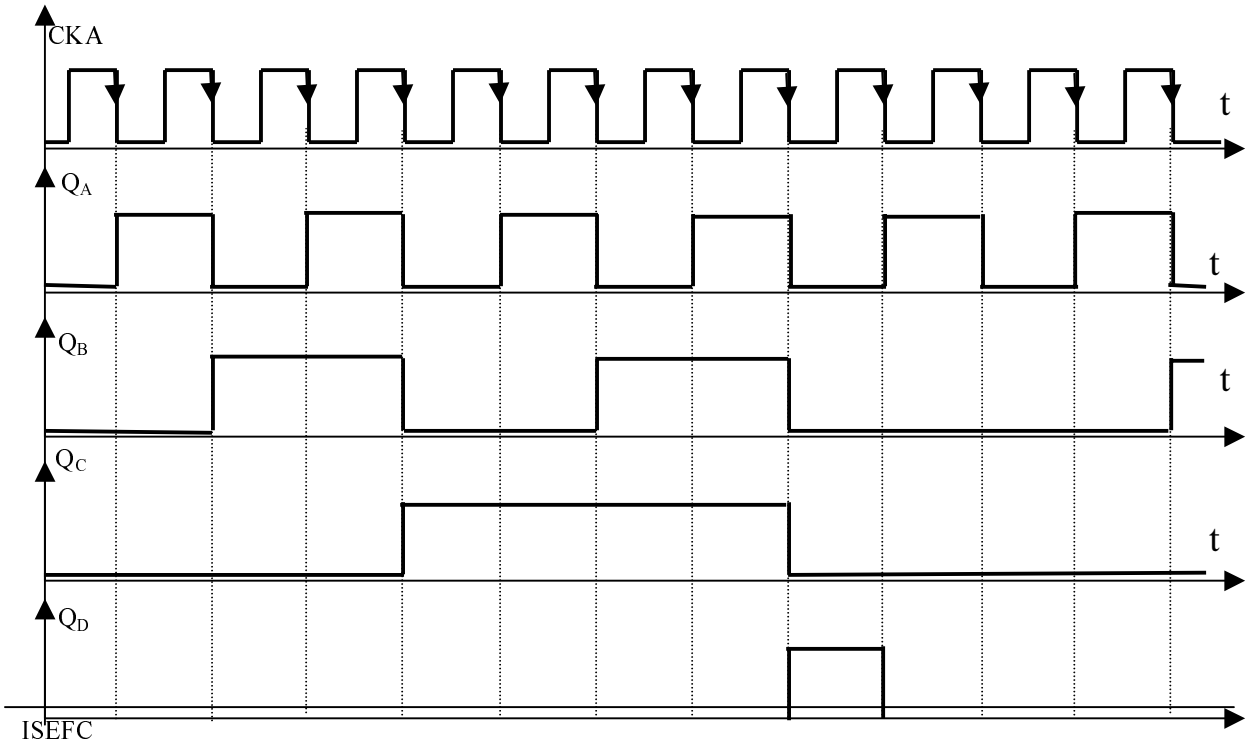


4) Etude du diviseur par 5 :(DIV5)

Compléter le chronogramme suivant :



3)Faire le montage de ce circuit de telle sorte qu'on obtient un diviseur de fréquence par 10(modulo 10).



t

4)Faire la synthèse d'un compteur asynchrone modulo 10 en utilisant des bascules JK.

$10 = 2^4$ $\Longrightarrow$ Donc 4 bascules.

Comptage : 0⟶► 9 La $10^{ème}$ impulsion remet le compteur à zéro.

Décimale	Binaire			
	Q_D	Q_C	Q_B	Q_A
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1
10=0	1	0	1	0

5)Faire le montage d'un système de comptage (modulo 100) en utilisant deux compteurs modulo 10 .

GRAFCET

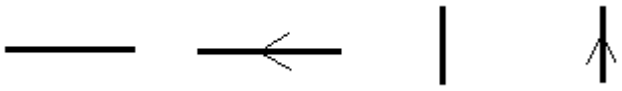
I - introduction

Le Grafcet est un outil graphique de définition pour l'automatisme séquentiel, en tout ou rien. Mais il est également utilisé dans beaucoup de cas combinatoires, dans le cas où il y a une séquence à respecter mais où l'état des capteurs suffirait pour résoudre le problème en combinatoire. Il utilise une représentation graphique. C'est un langage clair, strict mais sans ambiguïté, permettant par exemple au réalisateur de montrer au donneur d'ordre comment il a compris le cahier des charges. Langage universel, indépendant (dans un premier temps) de la réalisation pratique (peut se "câbler" par séquenceurs, être programmé sur automate voire sur ordinateur).

II - définitions

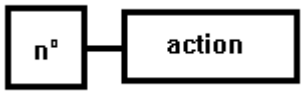
Un Grafcet est composé d'étapes, de transitions et de liaisons.

Une LIAISON est un arc orienté (ne peut être parcouru que dans un sens). A une extrémité d'une liaison il y a UNE (et une seule) étape, à l'autre UNE transition. On la représente par un trait plein rectiligne, vertical ou horizontal. Une verticale est parcourue de haut en bas, sinon il faut le préciser par une flèche. Une horizontale est parcourue de gauche à droite, sinon le préciser par une flèche.

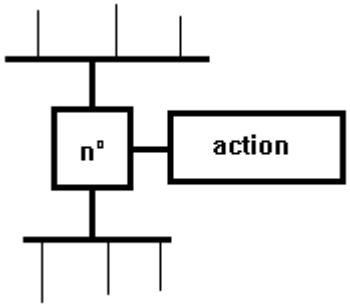


Une ETAPE correspond à une phase durant laquelle on effectue une ACTION pendant une certaine DUREE (même faible mais jamais nulle). L'action doit être stable, c'est à dire que l'on fait la même chose pendant toute la durée de l'étape, mais la notion d'action est assez large, en particulier composition de plusieurs actions, ou à l'opposé l'inaction (étape dite d'attente).

On représente chaque étape par un carré, l'action est représentée dans un rectangle à gauche, l'entrée se fait par le haut et la sortie par le bas. On numérote chaque étape par un entier positif, mais pas nécessairement croissant par pas de 1, il faut simplement que jamais deux étapes différentes n'aient le même numéro.



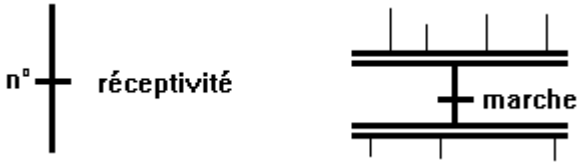
Si plusieurs liaisons arrivent sur une étape, pour plus de clarté on les fait arriver sur une barre horizontale, de même pour plusieurs liaisons partant de l'étape. Cette barre horizontale n'est pas une nouvelle entité du Grafcet, elle fait partie de l'étape, et ne représente qu'un "agrandissement" de la face supérieure (ou inférieure) de l'étape. On accepte de remplacer cette barre par un point si cela ne crée aucune ambiguïté.



Une étape est dite active lorsqu'elle correspond à une phase "en fonctionnement", c'est à dire qu'elle effectue l'action qui lui est associée. On représente quelquefois une étape active à un inótant donné en dessinant un point à l'intérieur.

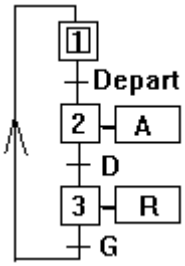
Une TRANSITION est une condition de passage d'une étape à une autre. Elle n'est que logique (dans son sens Vrai ou Faux), sans notion de durée. La condition est définie par une RECEPTIVITE qui est généralement une expression booléenne (c.à.d avec des ET et des OU) de l'état des CAPTEURS.

On représente une transition par un petit trait horizontal sur une liaison verticale. On note à droite la réceptivité, on peut noter à gauche un numéro de transition (entier positif, indépendant des numéros d'étapes). Dans le cas de plusieurs liaisons arrivant sur une transition, on les fait converger sur une grande double barre horizontale, qõi n'est qu'une représentation du dessus de la transition. De même pour plusieurs liaisons partant sous une transition.



III - exemple simple

Supposons un chariot pouvant avancer (A) ou reculer (R) sur un rail limité par deux capteurs G et D, Un cahier des charges pourrait être : Quand on appuie sur le bouton DEPART, on avance jusqu'en D, puis on revient. Ce C.d.C. est incomplet et imprécis. La réalisation du Grafcet permet de remarquer : Que fait-on avant l'appui de DEPART, jusqu'où revient-on, quelles sont les conditions initiales ? On réécrit le C.d.C. en 3 phases : Attendre jusqu'à l'appui de DEPART, avancer jusqu'en D, reculer jusqu'en G, attendre à nouveau DEPART et recommencer. On suppose le chariot initialement en G (sinon faire un cycle l'amenant en G).



IV - règles d'évolution

La modification de l'état de l'automatisme est appelée évolution, et est régie par 5 règles :

R1 : Les étapes INITIALES sont celles qui sont actives au début du fonctionnement. On les représente en doublant les côtés des symboles. On appelle début du fonctionnement le moment où le système n'a pas besoin de se souvenir de ce qui c'est passé auparavant (allumage du système, bouton "reset",...). Les étapes initiales sont souvent des étapes d'attente pour ne pas effectuer une action dangereuse par exemple à la fin d'une panne de secteur.

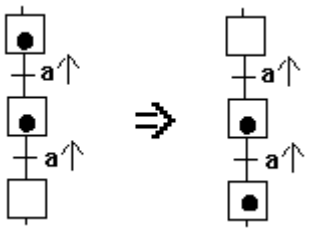
R2 : Une TRANSITION est soit validée, soit non validée (et pas à moitié validée). Elle est validée lorsque toutes les étapes immédiatement précédentes sont actives (toutes celles reliées directement à la double barre supérieure de la transition). Elle ne peut être FRANCHIE que lorsqu'elle est validée et que sa réceptivité est vraie. Elle est alors obligatoirement franchie.

R3 : Le FRANCHISSEMENT d'une transition entraîne l'activation de TOUTES les étapes immédiatement suivante et la désactivation de TOUTES les étapes immédiatement précédentes (TOUTES se limitant à 1 s'il n'y a pas de double barre).

R4 : Plusieurs transitions SIMULTANEMENT franchissables sont simultanément franchies (ou du moins toutes franchies dans un laps de temps négligeable pour le fonctionnement). La durée limite dépend du "temps de réponse" nécessaire à l'application (très différent entre un

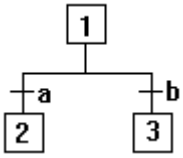
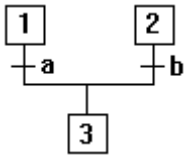
système de poursuite de missile et une ouverture de serre quand le soleil est suffisant).

R5 : Si une étape doit être à la fois activée et désactivée, elle RESTE active. Une temporisation ou un compteur actionnés par cette étape ne seraient pas réinitialisés. Cette règle est prévue pour lever toute ambiguïté dans certains cas particuliers qui pourraient arriver dans certains cas :

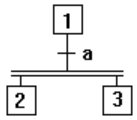
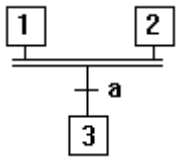


La partie COURS s'arrête ici. Toute autre règle que vous auriez pu entendre autre part ne fait pas partie du Grafcet. Il faudra TOUJOURS que votre Grafcet vérifie ce qui a été dit ci dessus (sinon ce n'est pas du Grafcet). Je tiens à préciser que le Grafcet devra être mis en oeuvre (câblé ou programmé) et donc une traduction de ce Grafcet en un schéma ou une suite d'instructions sera nécessaire. Le résultat de cette traduction, même s'il ressemble quelquefois à un Grafcet, ne peut pas imposer de nouvelles règles au Grafcet (qui dirait par exemple que le cas proposé après la règle 5 est interdit en Grafcet)

V - Configurations courantes

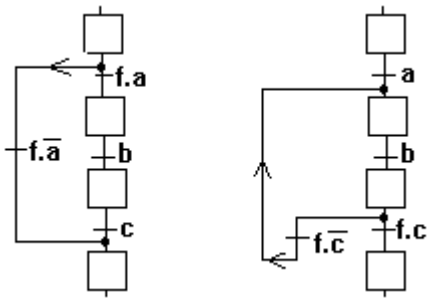
divergence en OU :  si 1 active et si a seul, alors désactivation de 1 et activation de 2, 3 inchangé. si a et b puis 1 active alors désactivation 1, activation 2 et 3 quel que soit leur état précédent. (règle 4)	Convergence en OU :  Si 1 active et a sans b, alors activation de 3 et désactivation de 1, 2 reste inchangé Si 1 et 2 et a et b alors 3 seule active

On appelle BARRE DE OU la barre symbolisant les entrées / sorties multiples d'étapes.

Divergence en ET :  si 1 active et si a, alors désactivation de 1 et activation de 2 et 3.	Convergence en ET :  Si 1 active seule et a alors aucun changement. Si 1 et 2 et a, alors activation de 3 et désactivation de 1 et 2.

On appelle couramment BARRE DE ET la double barre, mais attention ce n'est pas une entité à part mais une partie d'une transition.

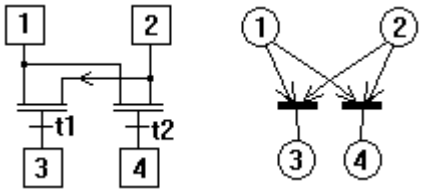
Détaillons également le saut avant (si a alors ...) et les boucles (répéter ... jusqu'à c). Ce sont les deux seules possibilités avec des OU: il ne peut y avoir de divergence en ou après une transition



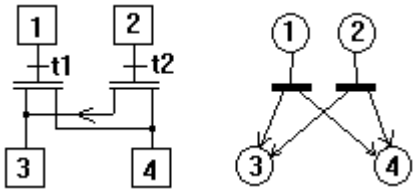
Passons maintenant à quelques problèmes plus complexes (tirés de "Comprendre et maîtriser le Grafcet, Blanchard, ed. Capadues"):

1- soient 4 étapes 1 à 4 et deux transitions de réceptivité t1 et t2. Construire la portion de Grafcet réalisant : Quand 1 ET 2 actifs alors si t1 passer en 3 (et désactiver 1 et 2), si t2 passer en 4 (et désactiver 1 et 2), sinon rester en 1 et 2

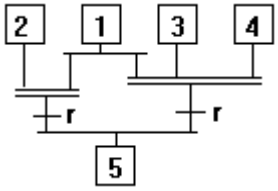
La solution ci-dessous est accompagnée d'une représentation de type "réseau de Petri" pour bien montrer où doivent se placer les convergences et divergences (à quoi doit être reliée 1?, à quoi doit être reliée t1? ...). En fait on trouve la solution facilement en analysant les cas d'évolution (quand franchit t'on t1 ?). Il faut souligner que l'ajout d'une étape intermédiaire n'est pas une bonne solution car tout passage d'une étape dure un laps de temps (donc discontinuité sur les sorties = aléa technologique)..



2 - Problème du même ordre : Quand (étape 1 et t1) OU (étape 2 et t2) alors passer en 3 ET 4:



3 - si {étape 1 et [étape 2 ou (étapes 3 et 4)]} et transition t alors activer l'étape 5 (et désactiver les autres).

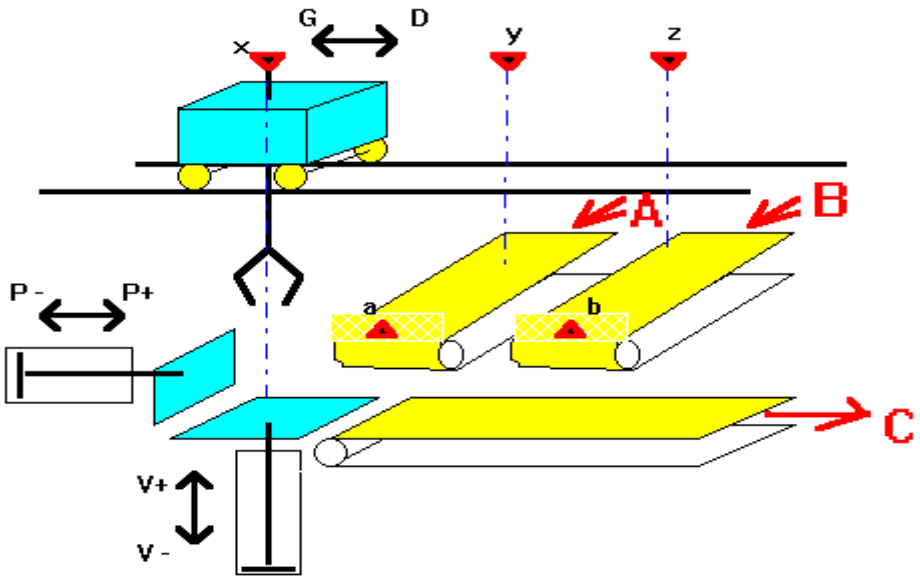


VI Cas génériques

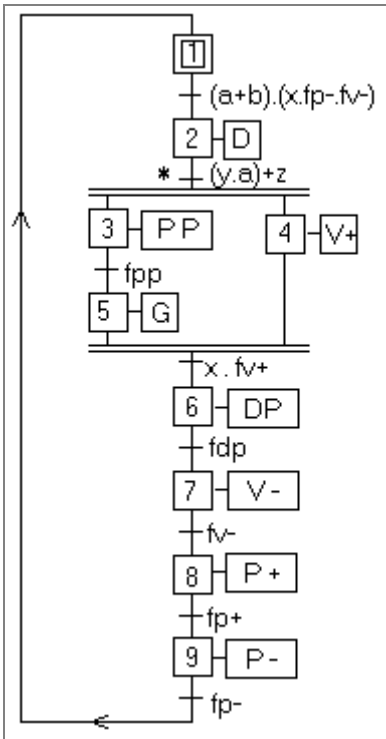
Nous traitons ici des exemples génériques, c'est à dire que les problèmes évoqués ici se posent assez souvent, et la méthode utilisée pour les résoudre pourra être réutilisée.

1 - priorité

Soit un chariot se déplaçant sur deux rails (action D vers la droite, G vers la gauche). Il comporte une pince pouvant prendre une pièce (PP, fin quand fpp) s'il se trouve sur le tapis A (capteur y) et qu'une pièce est présente (capteur a) (idem en z si b). Puis il retourne en x, pose la pièce (action DP, fin quand fdp) sur le plateaux supposé en position haute (fv+). Celui-ci descend (V-, jusqu'à fv-), un second vérin pousse la pièce (P+, fin quand fp+), puis le pousseur recule en fp-, le plateau remonte en fv+ Le tapis de sortie C est supposé toujours en mouvement. Les tapis A et B sont commandés par des systèmes non traités ici.



Effectuer d'abord un Grafcet linéaire comprenant une seule voie d'arrivée A. Puis l'améliorer en prévoyant les retours des actionneurs en temps masqué (attention toutefois de ne pas endommager le pousseur). Puis prévoir deux tapis d'alimentation A et B (en cas de pièces en a ET b, prendre celle en a). Puis prévoir une priorité tournante (en cas de conflit, prendre la voie qui n'a pas été servie la fois précédente) attention, si plusieurs pièces arrivent sur la même voie et aucune sur l'autre, ne pas bloquer le système. Puis modifier la règle de priorité en donnant en cas de conflit la priorité à celui qui n'en a pas profité lors du dernier conflit.

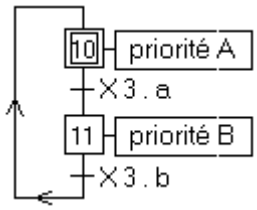


priorité voie A, retour V masqué

Pour gérer la priorité tournante, remplacer la réceptivité de la deuxième transition (notée *) par :

$y.a.(\overline{b}+X10)+z$

qui signifie : arrivé en y avec une pièce en a et soit pas de pièce en b, soit priorité. sinon on continue et quoi qu'il arrive on s'arrête en z (le chariot n'a pas de parachute), en rajoutant un second Grafcet définissant quelle voie est prioritaire :

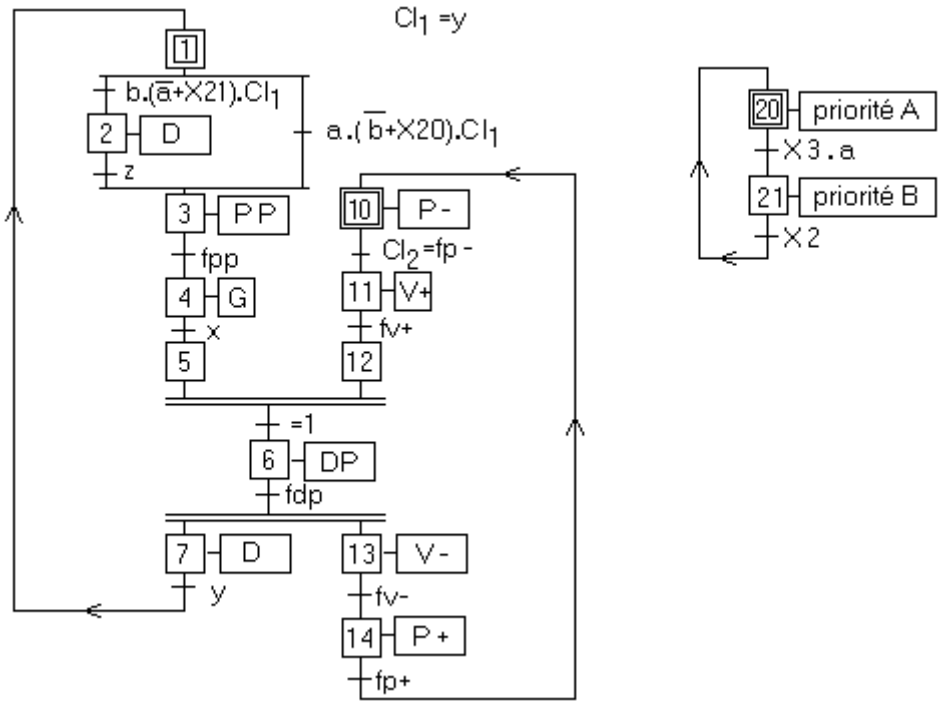


Chaque fois qu'une condition séquentielle (dépendant de ce qui s'est passé auparavant) intervient dans une réceptivité, il vaut mieux ne pas compliquer le Grafcet, mais "calculer" cette condition par un petit Grafcet annexe.

Améliorations :

a) permettre au chariot de rechercher une pièce dès qu'il a posé la précédente : séparer le problème en deux : chariot et partie basse.Prévoir deux Grafcet différents, pouvant évoluer simultanément, mais synchronisés pour le dépose de la pièce (par des Xi ou une ressource)

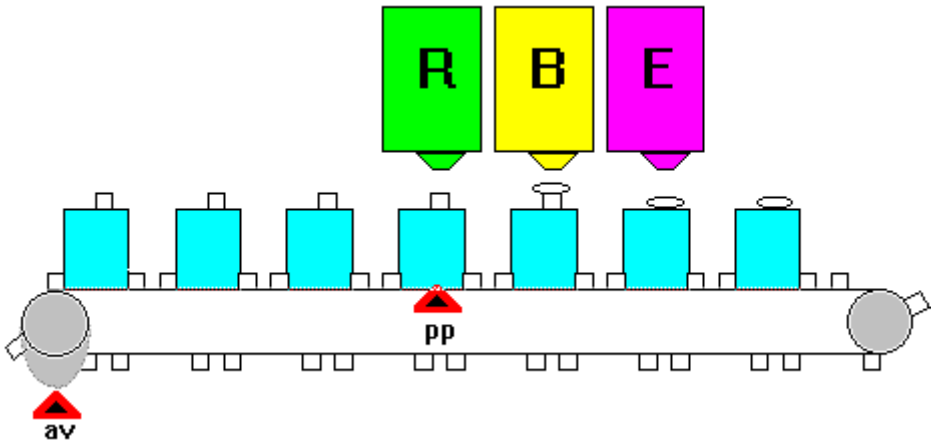
b) faire attendre le chariot en y plutôt qu'en x (pour améliorer le temps de réponse). Pour la partie basse, l'attente se fait plateau en haut, mais ce ne peut pas être l'état initial (il risque de descendre pendant la nuit). Prendre celà en compte :



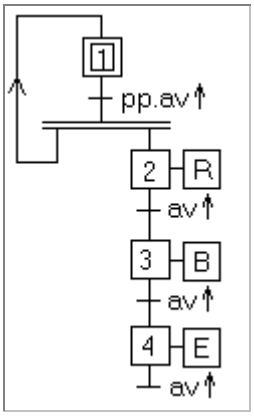
Les deux étapes initiales ne testent que leurs conditions initiales respectives (partie haute ou partie basse).

2 - travail à la chaîne

Soit une chaîne de remplissage de bidons d'huile. Un tapis roulant se déplaçant par saccades (cadencé par un système supposé externe à notre Grafcet, s'arrêtant à chaque nouvel appui de la came sur le capteur av) est alimenté manuellement (de temps en temps il manque des bidons). Trois postes sont prévus : remplissage (R), bouchage (B) et enfoncement (E).



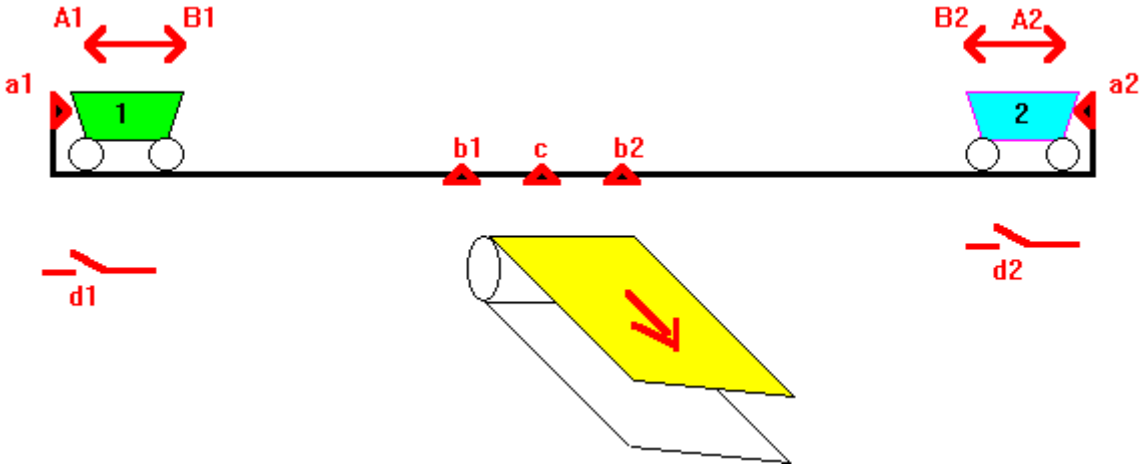
Un seul capteur détecte la présence d'un bidon en début de chaîne : pp. On désire faire les 3 opérations simultanément, sauf s'il n'y a pas de bidon sous le poste. S'il vous semble obligatoire de rajouter des capteurs, vous n'avez RIEN compris au Grafcet puisqu'il vous faut un système combinatoire (il vaut mieux alors câbler en combinatoire chaque poste : avance tapis ET présence bidon => effectuer l'action). On suppose que le tapis est vide lors de l'initialisation.



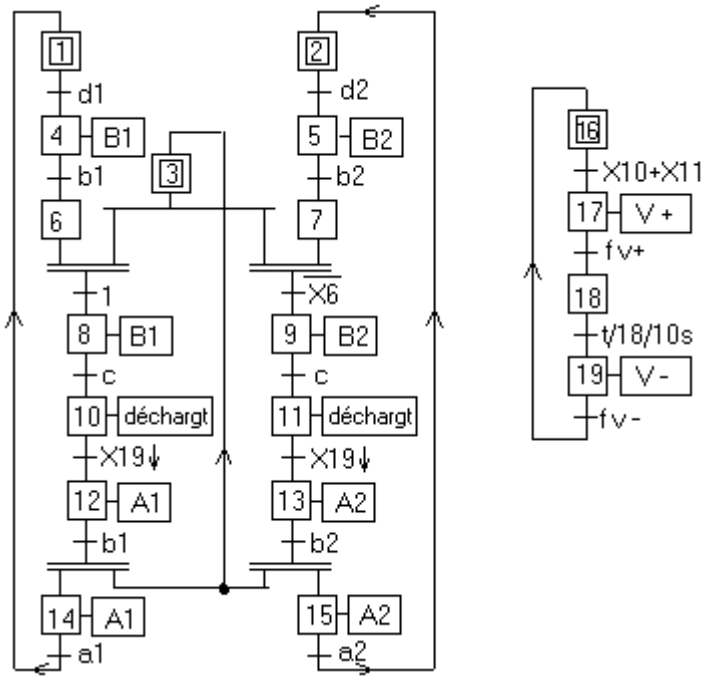
L'étape 1 est constamment active. La dernière transition est appelée "transition puits", mais il était possible de la relier à l'étape 1. En fonctionnement normal, toutes les étapes du Grafcet sont actives. Du point de vue commande, chaque opération comportera plusieurs étapes (R = descendre l'entonnoir, ouvrir le robinet,...) dont une seule sera active à la fois). Chaque activation représente un bidon dans le circuit.

Cette méthode utilise au mieux le séquencement du Grafcet, on peut maintenant rajouter des capteurs, mais qui n'auront pour fonction que de vérifier le bon fonctionnement du système. Dans tous les cas similaires, on utilisera cette démarche : faire le Grafcet pour une pièce seule, puis le modifier pour gérer l'ensemble des pièces, en vérifiant bien que jamais une même étape ne corresponde à 2 pièces, on décompose donc le système en tronçons et on ne laisse entrer dans un tronçon que s'il est libre. Exemples : atelier flexible (on suit la pièce pour chaque opération jusqu'au produit fini), montage (monter 2 pièces ensemble correspond à une convergence en ET : de 2 étapes actives on arrive à 1), chariots filo-guidés (si un tronçon est occupé, essayer de le contourner par une voie libre)...

3 - ressource (ou sémaphore)



Au fond du puits de mine ndeg. i, un mineur remplit un chariot X_i. Quand il est plein (le chariot), il (le mineur) appuie sur un bouton d_i. Immédiatement, le chariot se déplace dans la direction B_i jusqu'au poste de déchargement, composé d'un tapis roulant en mouvement continu, et d'un vérin V qui retourne la benne. Si le poste de déchargement est libre, le chariot avance jusqu'au capteur c, est déchargé puis s'en retourne en a_i. Si le poste est occupé, il attend son tour en b_i. Le poste de déchargement, commun à plusieurs voies, n'est utilisable que par une voie à la fois. On l'appelle une "ressource physique". Traiter le cas de 2 voies (pas nécessairement de la même longueur).



Supposer que la ressource est occupée en utilisant le capteur c est IDIOT : et s'il est entre b_i et c ? Et si le temps de freinage l'a arrêté juste à côté de c ? Il faut utiliser les facilités séquentielles du Grafcet autant que possible (ne tester un capteur que quand c'est nécessaire). Un capteur ne doit servir que

comme condition de passage d'une étape à une autre, mais pas pour vérifier un état du système qui découle du séquencement effectué (par exemple, une transition vérifie la présence d'une pièce, aucune action ne déplace la pièce puis on re-vérifie la présence : Ce n'est censé que si l'on prévoit dans le Grafcet ce qu'il faut faire si la pièce a disparu). Ici, on utilise donc une étape (la ressource), qui est active quand la ressource physique est disponible. Dès utilisation, on la désactive, pour la réactiver quand on libère la ressource physique.

On pouvait également résoudre le problème par des Grafcets séparés (un pour chaque chariot, un pour le déchargement) synchronisés par des Xi. La seule différence est que n'ayant plus de divergence sous l'étape 3, on risque d'oublier de traiter le cas d'arrivée simultanée en b1 et b2, cas arrivant assez rarement pour que l'on ne détecte pas le problème en phase d'essais, mais se produira de temps en temps en fonctionnement réel sans que l'on puisse reproduire le problème lorsqu'un spécialiste sera présent (seule solution : graphe des états accessibles).