TD 5
Synthèse d'automates [1]

Méthodologie

Ex. 1 : Méthodologie de synthèse d'automates : reconnaisseur de séquences

Soit un circuit prenant en entrée une lettre dans l'ensemble $\{a, b, c\}$ et générant en sortie une lettre dans l'ensemble $\{o, n\}$ qui vaut o ssi on vient de reconnaître la séquence $a + bcc^*$ et n sinon. Noter que la séquence de caractères en entrée est infinie : l'automate ne s'arrête jamais (*i.e.* pas d'état puits).

Graphe d'états de l'automate

On spécifiera en général l'automate comme un automate de Moore (sortie liée aux états).

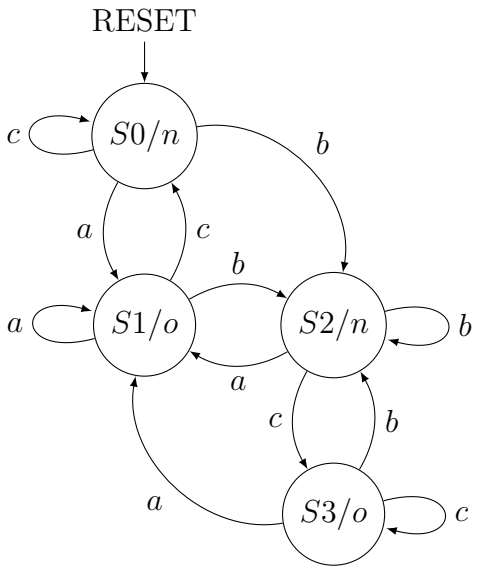


Table de transitions

Cette table est une représentation textuelle du graphe.

1. **Remarque : Automates en TL1 et en architecture**
Vus avez déjà vu une notion d'automate dans le cours de Théorie des Langages 1. Quel est le lien avec les automates vus ici ? S'agit-il de la même chose ? D'un autre formalisme ?

En TL1, l'objectif est de reconnaître des langages et les automates sont utilisés dans ce sens : un automate renvoie un booléen qui indique si le mot est dans le langage ou non.

En architecture, l'idée est de contrôler un système. Ce système n'a en général pas de durée de fonctionnement donc doit pouvoir recevoir des entrées indéfiniment et l'automate (de Moore) doit émettre des commandes pour le reste du sytème. Ainsi, la fonction de génération donne une valeur de sortie (une commande) pour chaque état, là ou en TL1 seul le dernier état avait une sortie booléenne (est-ce un état acceptant ?).

La correspondance entre les descriptions vues en TL1 et en architecture est la suivante : les états, l'état inital et les transitions ont le même sens dans les deux cadres ; le *vocabulaire* correspond aux combinaisons des signaux d'entrées et les *états acceptants* à la fonction de génération.

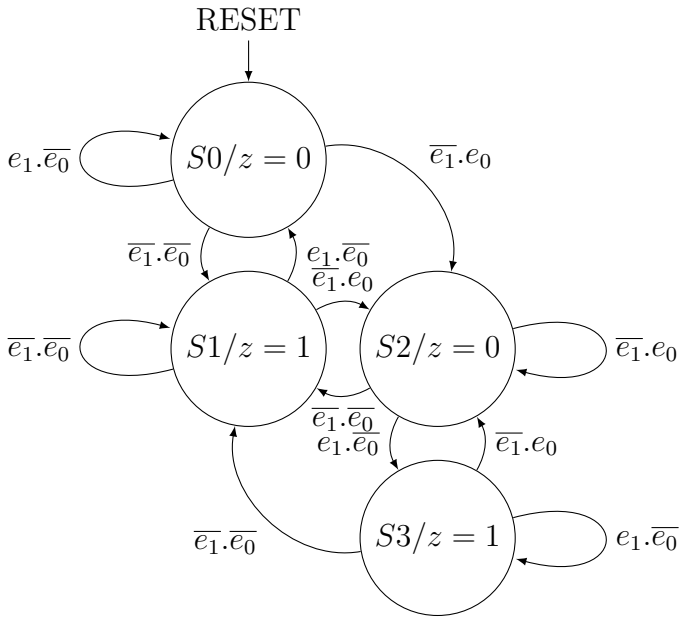
État courant	Sortie	Entrée	État futur
S0	n	a	S1
		b	S2
		c	S0
S1	o	a	S1
		b	S2
		c	S0
S2	n	a	S1
		b	S2
		c	S3
S3	o	a	S1
		b	S2
		c	S3

Codage des entrées et des sorties

On a besoin de 2 bits e_1 et e_0 pour coder les trois lettres en entrée : $a \equiv 00$, $b \equiv 01$ et $c \equiv 10$. Le code 11 est impossible. On a besoin de 1 bit z pour coder les deux lettres en sortie : $n \equiv 0$ et $o \equiv 1$.

Remarque : en général, le codage des entrées et des sorties d'un automate sur des variables booléennes est imposé par l'environnement, et cette étape ne sera pas nécessaire.

On peut réécrire le graphe d'états et la table de transitions de l'automate en utilisant ce codage des entrées et de la sortie :



État courant	Sortie	Entrée	État futur
S0	0	$\overline{e_1}.\overline{e_0}$	S1
		$\overline{e_1}.e_0$	S2
		$e_1.\overline{e_0}$	S0
S1	1	$\overline{e_1}.\overline{e_0}$	S1
		$\overline{e_1}.e_0$	S2
		$e_1.\overline{e_0}$	S0
S2	0	$\overline{e_1}.\overline{e_0}$	S1
		$\overline{e_1}.e_0$	S2
		$e_1.\overline{e_0}$	S3
S3	1	$\overline{e_1}.\overline{e_0}$	S1
		$\overline{e_1}.e_0$	S2
		$e_1.\overline{e_0}$	S3

Codage logarithmique des états

On choisit ici un codage logarithmique, c'est-à-dire avec le nombre minimum de variables booléennes (le codage 1 parmi n, qui utilise une variable par état, est présenté à la fin de l'exercice).

En codage logarithmique, pour coder les 4 états possibles, il faut au minimum 2 bits q_1 et q_0. Pour cet exercice, on peut choisir le codage : $S0 \equiv 00$, $S1 \equiv 01$, $S2 \equiv 10$ et $S3 \equiv 11$.

Expressions simplifiées des variables d'état et de la sortie de l'automate

Il s'agit de trouver les expressions des variables q_1 et q_0 à l'instant $t+1$, en fonction de q_1, q_0 et des entrées à l'instant t. Nous utilisons des bascules D : pour obtenir la valeur souhaitée à l'instant $t+1$ en sortie des bascules, il faut mettre à l'instant t cette valeur sur les entrées D des bascules. On cherchera donc les expressions simplifiées de D_1 et de D_0 en fonction de q_1, q_0 et des entrées e_1 et e_0.

e_1e_0 \ q_1q_0	**00**	**01**	**11**	**10**
00	0	0	0	0
01	1	1	1	1
11	Φ	Φ	Φ	Φ
10	0	0	1	1

Expression de D_1

e_1e_0 \ q_1q_0	**00**	**01**	**11**	**10**
00	1	1	1	1
01	0	0	0	0
11	Φ	Φ	Φ	Φ
10	0	0	1	1

Expression de D_0

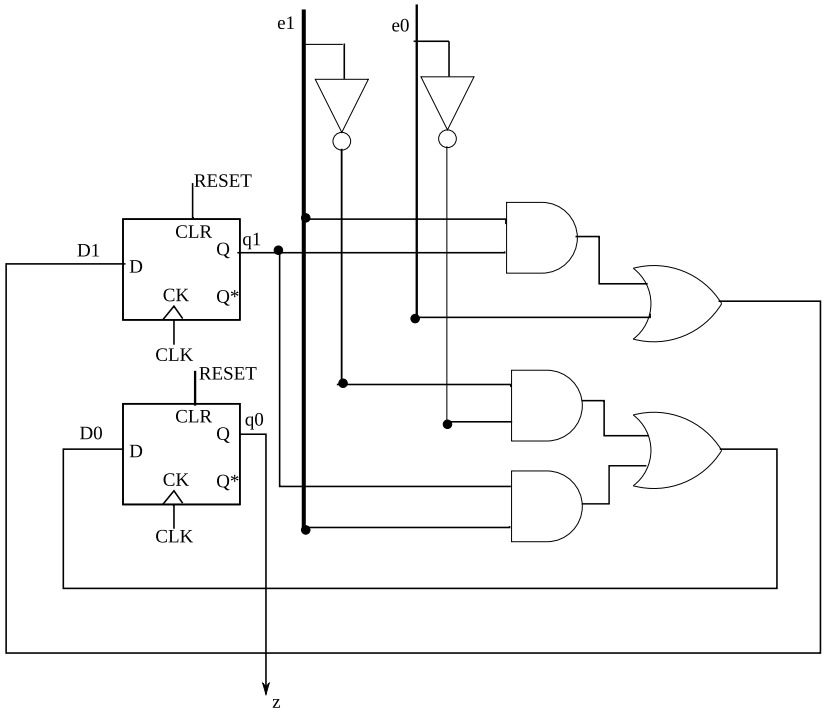
q_1 \ q_0	**0**	**1**
0	0	1
1	0	1

Expression de z

On a donc :

$$\begin{aligned} D_1 &= e_0 + e_1.q_1 \\ D_0 &= \overline{e_1}.\overline{e_0} + q_1.e_1 \\ z &= q_0 \end{aligned}$$

Schéma du circuit



Codage 1-parmi-n ou "one-hot"

Le principe de ce codage est d'associer une variable d'état à chaque état (donc une bascule par état). Une variable d'état (ou une bascule) à 1 signifie que l'état correspondant est actif ; quand la variable d'état est à 0, l'état est inactif.

On peut faire la synthèse du circuit à partir du graphe d'états ou de la table de transition.

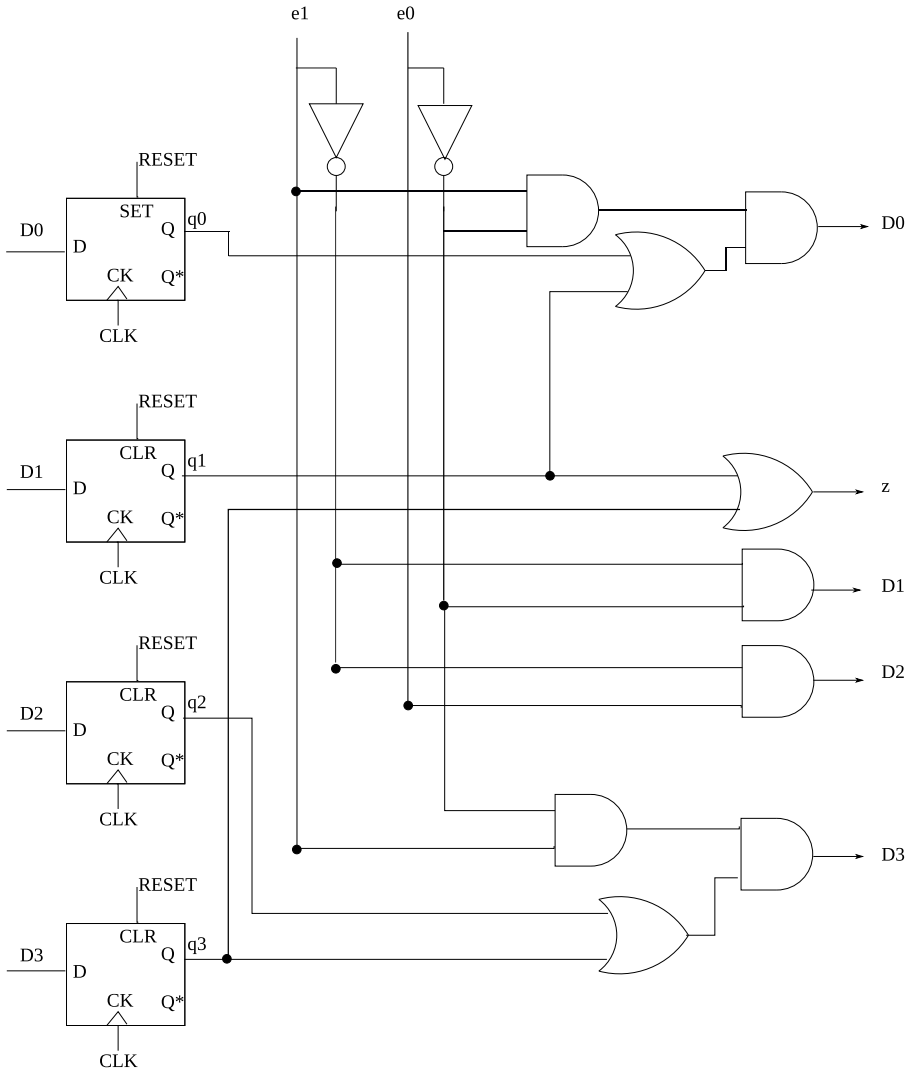
Pour notre exemple : il y a 4 états, donc 4 bascules (entrées D_0, D_1, D_2 et D_3, sorties Q_0, Q_1, Q_2 et Q_3 avec la bascule i correspondant à l'état S_i). A partir de la table de transition on calcule la table de vérité des entrées des bascules :

État courant	Sortie	Entrée	D0	D1	D2	D3
Q0	0	$\overline{e_1}.\overline{e_0}$	0	1	0	0
		$\overline{e_1}.e_0$	0	0	1	0
		$e_1.\overline{e_0}$	1	0	0	0
Q1	1	$\overline{e_1}.\overline{e_0}$	0	1	0	0
		$\overline{e_1}.e_0$	0	0	1	0
		$e_1.\overline{e_0}$	1	0	0	0
Q2	0	$\overline{e_1}.\overline{e_0}$	0	1	0	0
		$\overline{e_1}.e_0$	0	0	1	0
		$e_1.\overline{e_0}$	0	0	0	1
Q3	1	$\overline{e_1}.\overline{e_0}$	0	1	0	0
		$\overline{e_1}.e_0$	0	0	1	0
		$e_1.\overline{e_0}$	0	0	0	1

À partir de cette table, on obtient :

$$
\begin{aligned}
D_0 &= e_1.\overline{e_0}.(Q_0 + Q_1) \\
D_1 &= \overline{e_1}.\overline{e_0}.(Q_0 + Q_1 + Q_2 + Q_3) = \overline{e_1}.\overline{e_0} \\
D_2 &= \overline{e_1}.e_0.(Q_0 + Q_1 + Q_2 + Q_3) = \overline{e_1}.e_0 \\
D_3 &= e_1.\overline{e_0}.(Q_2 + Q_3) \\
z &= Q_1 + Q_3
\end{aligned}
$$

Le circuit correspondant est donné ci-dessous. Noter que la bascule associée à l'état initial Q_0 est initialisée à 1 (pour que cet état devienne actif à l'initialisation), toutes les autres sont initialisées à 0.



Ex. 2 : Détecteur de séquence $10^{+}11$

On désire faire un module matériel permettant de reconnaître la séquence $10^{+}11$ avec un automate de Moore synchrone.

Les signaux sont les suivants :

E fournit la donnée série (1 bit), c'est à dire la suite de bits qui constitue potentiellement la séquence ;

RESET force l'automate dans son état initial.

S est la sortie série qui passe à 1 quand la séquence $10^{+}11$ est détectée.

On détectera des séquences qui se recouvrent, par exemple 10011011, détectera 2 fois le motif $10^{+}11$.

L'entrée RESET n'est pas traitée comme une entrée ordinaire : elle attaque le reset des bascules D pour les remettre dans leur état initial. Il est en fait indispensable de faire ainsi car cela permet de forcer un état initial précis quel que soit l'état du circuit, même s'il ne représente pas un état de l'automate valide (par exemple, aucun registre à 1 dans un codage 1 parmi n), comme ce peut être le cas à la mise sous tension du circuit. En conséquence, le signal RESET n'apparaît pas dans les transitions de l'automate, ni dans les conditions de complétude et orthogonalité. On appelle un tel signal une « servitude », au même titre que l'horloge.

Question 1 Déterminer le graphe d'états, en précisant les conditions de transitions et les valeurs de sortie pour chaque état.

Question 2 Choisir un codage logarithmique des états de l'automate et construire la table de transition de l'automate (en notant q_i le $i_{ème}$ bit de la sortie de la bascule D qui représente l'état courant et d_i le $i_{ème}$ bit de l'entrée des bascules D qui représente l'état futur),

Question 3 Donner les expressions simplifiées des variables d'état de l'automate.

Question 4 Dessiner le schéma correspondant/

Question 5 Proposer une réalisation de cet automate en utilisant un codage 1 parmi n.

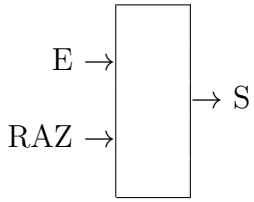
Ex. 3 : Décodeur « Non-Retour à Zéro Inversé »

On désire faire un module matériel permettant de décoder un signal NRZI avec un automate de Moore synchrone possédant 4 états, notés $\mathcal{A}$, $\mathcal{B}$, $\mathcal{C}$ et $\mathcal{D}$. Cet automate possède 2 entrées E et RAZ, et une unique sortie S. Les signaux sont les suivants :

E fournit la donnée série (1 bit), c'est à dire la suite de bits à décoder un par un ;

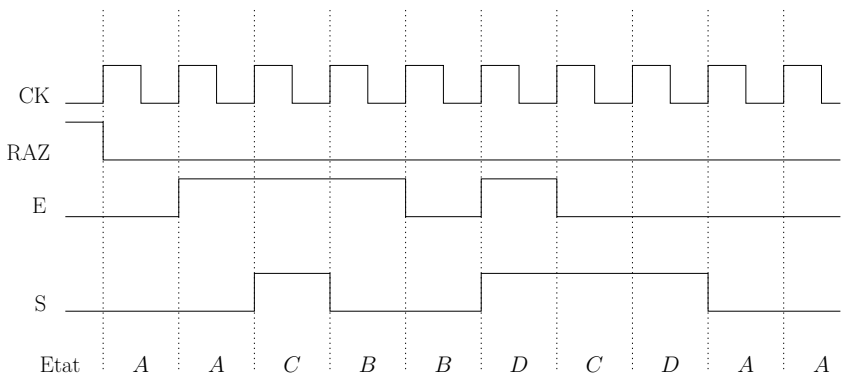
RAZ force l'automate dans son état initial, l'état $\mathcal{A}$, lorsqu'elle est à '1' ;

S est la sortie série.



L'entrée RAZ n'est pas traitée comme une entrée ordinaire : elle attaque le reset des bascules D pour les remettre dans leur état initial. Il est en fait indispensable de faire ainsi car cela permet de forcer un état initial précis quel que soit l'état du circuit, même s'il ne représente pas un état de l'automate valide (par exemple, aucun registre à 1 dans un codage 1 parmi n), comme ce peut être le cas à la mise sous tension du circuit. En conséquence, le signal RAZ n'apparaît pas dans les transitions de l'automate, ni dans les conditions de complétude et orthogonalité. On appelle un tel signal une « servitude », au même titre que l'horloge.

Le chronogramme suivant montre un exemple de décodage qui illustre également le fonctionnement de l'automate.



Question 1 En utilisant ce chronogramme, déterminer le graphe d'états, en précisant les conditions de transitions et les valeurs de sortie pour chaque état.

Question 2 Les états de l'automate sont codés ainsi, sur deux bits notés Q1 et Q0.

	Q1	Q0
$\mathcal{A}$	0	0
$\mathcal{B}$	0	1
$\mathcal{C}$	1	0
$\mathcal{D}$	1	1

Construire la table de transition de l'automate, en notant D1 et D0, les valeurs futures à écrire dans les bascules D mémorisant Q1 et Q0.

Question 3 Donner les expressions booléennes simplifiées de D1, D0 et S.

Pour aller plus loin...
Question 4 Proposer une réalisation de cet automate en utilisant un codage 1 parmi n.

Pour aller plus loin...
Question 5 Dessiner le schéma du circuit permettant de décoder le code NRZI.

Utilisation de cet automate : Le codage NRZI (Non-Retour à Zéro Inversé) consiste à changer le niveau du signal codé chaque fois que l'on code un 1 (si le signal de sortie vaut 0, alors il devra valoir 1 au cycle suivant, et inversement) et à conserver le niveau du signal codé chaque fois que l'on code un 0 (si le signal de sortie vaut 0 au cycle courant, alors il y reste au cycle suivant, *idem* pour 1). Le signal de sortie est initialement à 0. Dans cet exercice, on a proposé un module matériel permettant de décoder un signal NRZI avec un automate de Moore synchrone possédant 4 états.

Pour aller plus loin...
Ex. 4 : Gestion d'un feu tricolore

On cherche à modéliser le fonctionnement d'un feu tricolore composé d'un affichage à trois couleurs (Rouge, Orange, Vert) pour les véhicules et d'un affichage bicolore (Rouge, Vert) pour les piétons. Le feu fonctionne selon le principe très simplifié détaillé ci-dessous :

— le feu véhicules reste au vert tant qu'un piéton n'a pas appuyé sur le bouton d'appel ;

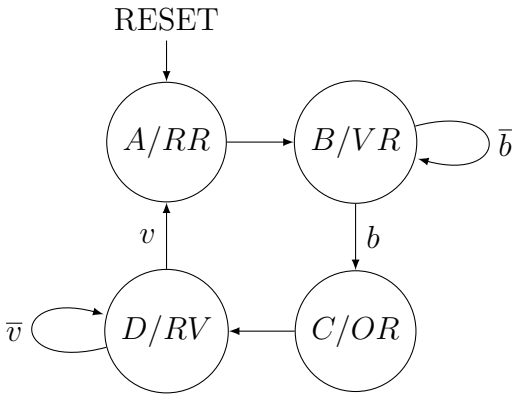
— le feu véhicules passe à l'orange dès qu'un piéton appuie sur le bouton d'appel ($b = 1$), puis après un certain temps il passe au rouge et le feu piétons passe au vert simultanément ;

— le feu piétons reste au vert tant qu'aucun véhicule n'est arrêté au feu ;

— lorsqu'un véhicule est détecté en attente devant le feu ($v = 1$), le feu piétons passe au rouge, puis après un certain temps, le feu véhicules passe au vert.

Pour simplifier, on considère que la notion de temps d'attente est manifestée par le passage d'un cycle (*i.e.* on ne prend pas en compte des durées d'attente différentes). Par convention, on décide que l'état initial est celui où les deux feux sont au rouge.

On donne un automate de Moore qui modélise le comportement de ce feu tricolore. Comme un énoncé en langue naturelle est toujours beaucoup plus imprécis qu'un modèle rigoureux, des choix d'interprétation ont été fait lors de la conception de cet automate : on considérera l'automate comme l'énoncé de référence.



Dans chaque état, la sortie est représentée par un couple de couleurs correspondant à l'affichage du feu véhicules suivi de celui du feu piétons.

Question 1 a) Combien y a t-il de combinaisons possibles et de combinaisons réelles pour les valeurs des feux piétons et véhicules ?
b) Peut-on établir un lien avec le nombre d'états de l'automate dans le cas général ?

Question 2 Choisir un codage logarithmique des états de l'automate et donner les expressions simplifiées des variables d'état de l'automate.

Question 3 Donner les expressions simplifiées des sorties de l'automate.

Pour aller plus loin...
Question 4 Dessiner le schéma du circuit pilotant le feu tricolore.

Pour aller plus loin...

Ex. 5 : Gestion de la température par hystérésis

On cherche à réaliser un système permettant d'assurer le maintien à une température quasi-constante d'un poulailler industriel, afin d'assurer aux volatiles une durée de vie compatible avec les normes européennes. Ce système repose sur la disponibilité d'un climatiseur pouvant produire du chaud, du froid, ou ne rien faire, en fonction de la température courante de l'entrepôt (qui dépend de la température externe et du degré d'agitation des gallinacés).

Le système est basé sur le principe de l'hystérésis, c'est-à-dire que le comportement lors de l'accroissement de la température est différent du comportement lorsque celle-ci baisse. Trois températures de référence sont nécessaires au fonctionnement du système : $T_{min} < T_{nom} < T_{max}$. La production du chaud ou du froid, indiquée respectivement par un signal C à 1 ou F à 1, se passe comme suit :

- lorsque la température T devient inférieure à T_{min}, il y a production de chaud : $C \leftarrow 1$;
- la production de chaud s'arrête, *i.e.* $C \leftarrow 0$, lorsque la température devient supérieure ou égale à la valeur nominale T_{nom} ;
- lorsque la température devient supérieure ou égale à T_{max}, il y a production de froid $F \leftarrow 1$;
- la production de froid cesse lorsque la température devient inférieure à la valeur nominale T_{nom}.

Question 1 Peut-on représenter les valeurs de C et F en fonction de la température sous la forme de tables de vérité ? Justifier.

Le système exploite un capteur de température qui fournit une information encodée sur 2 bits comme précisé ici :

b_1	b_0	Signification
0	0	$T < T_{min}$
0	1	$T_{min} \leq T < T_{nom}$
1	1	$T_{nom} \leq T < T_{max}$
1	0	$T_{max} \leq T$

On cherche à formaliser le comportement du système comme un automate de Moore, en faisant l'hypothèse que la période de l'horloge de l'automate est très inférieure au temps qu'il faut pour que la température T du poulailler passe de T_{min} à T_{max} ou inversement, à cause de l'inertie thermique.

Question 2 Préciser quelles sont les entrées et les sorties de l'automate.

Question 3 Donner le nombre d'états de l'automate et représenter le graphe de l'automate en y incluant les conditions de transitions sous forme *d'intervalle de températures* sur les arcs et les valeurs des sorties dans les états. Transformer ensuite les intervalles de températures en conditions booléennes.

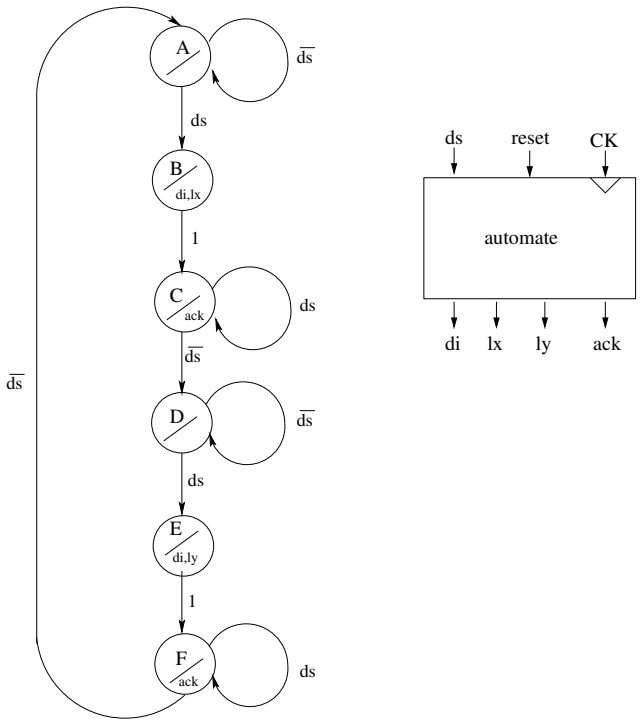
Question 4 Proposer un codage logarithmique des états de façon à ce que les expressions des sorties C et F soient simples et donner la table de transition qui précise les sorties et l'état futur en fonction des entrées et de l'état courant. On note Q_i les sorties du registre représentant l'état courant et D_i les entrées du registre, représentant l'état futur.

Question 5 Donner les expressions simplifiées de C, F et des D_i, et faites le schéma en portes correspondant.

Pour aller plus loin...

Ex. 6 : Automate de synchronisation

Soit un automate spécifié par le graphe ci-dessous. Outre l'entrée d'initialisation $RESET$, cet automate a une entrée : ds et 4 sorties di, ack, lx, ly. Il évolue au front montant de l'horloge CK.



Question 1 Construire la table de transition de cet automate.

Question 2 Proposer une réalisation de cet automate utilisant un codage 1 parmi n (one hot encoding) à partir de bascules D.

Question 3 On cherche à réaliser cet automate avec un codage logarithmique (avec 3 variables d'état, puisqu'il y a 6 états). Rechercher un codage permettant de minimiser le nombre total de monômes dans les expressions de ces variables d'état et des sorties, puis donner les équations.