C^3
Capteurs, contrôle, commande

Partie 3: systèmes à événements discrets, automatismes

E. Le Carpentier

ECN

EI1–2018/2019

Vidéoprojection disponible sur le serveur pédagogique

Rappels : « Signaux et systèmes »

Contexte

$$u(t) \longrightarrow \boxed{S} \longrightarrow y(t)$$

$$y = S(u)$$

L'entrée u et la sortie y peuvent prendre un continuum de valeurs.

Système statique Pour tout t, l'entrée à l'instant t impose la sortie à l'instant t :

$$y(t) = g(u(t))$$

Système dynamique Pour tout t, la sortie à l'instant t dépend de l'historique de l'entrée $\{u(\tau) \mid \tau \leqslant t\}$.
La représentation dans l'espace d'état est une modélisation de tels systèmes :

$$\begin{cases} y(t) = g(u(t), x(t)) \\ \dot{x}(t) = f(u(t), x(t)) \end{cases} \qquad \begin{cases} y[n] \quad\;\; = g(u[n], x[n]) \\ x[n+1] = f(u[n], x[n]) \end{cases}$$

Systèmes à événements discrets (SED)

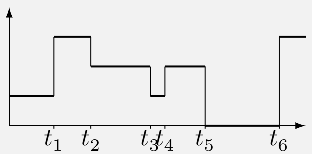
Nouveau contexte

$$u(t) \longrightarrow \boxed{S} \longrightarrow y(t)$$

$$y = S(u)$$

L'entrée u et la sortie y ne peuvent prendre qu'un nombre fini de valeurs.

Pourquoi l'appellation « SED » Si un signal ne prend qu'un nombre fini de valeurs, il ne peut changer de valeur continuellement, il est constant par morceaux.
Les sauts sont des événements discrets (on peut les compter).



Attention ! Les événements sont discrets, mais si les instants des événements :

- peuvent prendre un continuum de valeurs ($t_i \in \mathbb{R}$) : SED asynchrone ;
- sont alignés sur une grille d'échantillonnage ($t_i \in \mathbb{Z}\, T$) : SED synchrone

Notions d'alphabet

Alphabet d'entrée Ensemble des valeurs que peut prendre l'entrée.

Alphabet de sortie Ensemble des valeurs que peut prendre la sortie.

Elément de l'alphabet Lettre, caractère, symbole, étiquette.

Exemple en génomique On parcourt un chaîne d'ADN (suite de nucléotides A, T, G, C).
Le « temps » discret correspond à l'indice du nucléotide le long de la chaîne.
Cette chaîne excite un SED qui retourne 1 quand une certaine séquence (par exemple AATGGCA) est rencontrée, 0 sinon.

Alphabet d'entrée $\{A, T, G, C\}$
Alphabet de sortie $\{0, 1\}$

Notions d'alphabet

Niveau logique, binary digit, bit 0 ou 1

En automatismes Les entrées-sorties d'un SED sont :

- des interrupteurs, boutons-poussoirs, relais, ouverts (0) ou fermés (1)
- des capteurs tout (1) ou rien (0)

Exemple Alphabet d'entrée pour un système commandé par

- 2 boutons poussoirs : $\{00, 01, 10, 11\}$
- 1 basculeur 3 positions : $\{00, 01, 10\}$

Mais, en pratique, dans ce dernier cas, il est prudent de compléter l'alphabet avec l'étiquette 11 qui trahit un basculeur défectueux.

Remarque Si l'alphabet contient N éléments, numérotés de 0 à $N-1$, il peut être codé sur n bits, avec n tel que $2^n \geqslant N$.

SED combinatoires

Définition Pour tout t, l'entrée à l'instant t impose la sortie à l'instant t. Il existe une fonction g : alphabet d'entrée $\longrightarrow$ alphabet de sortie telle que :

$$y(t) = g(u(t))$$

Il n'y a pas d'ambiguïté à noter $y = g(u)$

Exemple : va et vient u_1 et u_2 positions des interrupteurs (0 : en haut ; 1 : en bas)
$y = 1$ si la lumière est allumée, 0 sinon

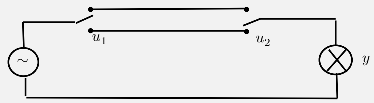


Table de vérité Décrit la fonction $g : \{00, 01, 10, 11\} \to \{0, 1\}$.

u_1	u_2	y
0	0	1
0	1	0
1	0	0
1	1	1

SED séquentiel

Définition Pour tout t, la sortie à l'instant t dépend de l'historique de l'entrée $\{u(\tau) \mid \tau \leqslant t\}$.
Il existe un *état* $x(t)$, avec son alphabet, tel qu'on puisse écrire :

$$y(t) = g(u(t), x(t))$$

Exemple On dispose de 2 boutons-poussoirs :

- Bouton 1 appuyé ($u_1 = 1$) pour allumer la lumière
- Bouton 2 appuyé ($u_2 = 1$) pour éteindre la lumière
- Relâcher les boutons n'a aucun effet

Table de vérité

$u_1(t)$	$u_2(t)$	$y(t + dt)$	
0	0	y(t)	mémorisation
0	1	0	extinction
1	0	1	allumage
1	1	?	que faire ?

Lorsque les 2 entrées sont à 0, la sortie peut être à 0 ou 1.

Schéma général (exemple d'une commande de volets roulants)

3 composants

- L'opérateur
- La partie opérative
- La partie commande

L'opérateur L'être humain ; sa fonction se réduit en général à enclencher la mise en route de l'automatisme (l'ouverture (Op, open) ou la fermeture des volets (Cl, close))
Soit c les consignes de l'opérateur : $c = (\mathrm{Op}, \mathrm{Cl}) \to \{0, 1\}^2$

Schéma général (exemple d'une commande de volets roulants)

La partie opérative Système à piloter (les volets roulants), muni :

- d'actionneurs (moteurs électriques) et de préactionneurs
(relais de commande du moteur, Up pour ouvrir, Dn pour fermer) ;
l'entrée de la partie opérative est donc $u = (\mathrm{Up}, \mathrm{Dn}) \to \{0,1\}^2$
- de capteurs binaires
(Top et Btm pour fins de course haut et bas) ;
la sortie de la partie opérative est donc $y = (\mathrm{Top}, \mathrm{Btm}) \to \{0,1\}^2$

Remarque C'est un SED, même si des variables internes évoluent continûment ; en supposant que le volet monte ou descend de $x = 0$ à $x = \ell$ à la vitesse V :

$$y(t) = \begin{cases} (1,0) & \text{si } x(t) = \ell \\ (0,1) & \text{si } x(t) = 0 \\ (0,0) & \text{sinon} \end{cases}$$

$$\dot{x}(t) = \begin{cases} +V & \text{si } 0 \leqslant x(t) < \ell \text{ et } u(t) = (1,0) \\ -V & \text{si } 0 < x(t) \leqslant \ell \text{ et } u(t) = (0,1) \\ 0 & \text{si } u(t) = (0,0) \end{cases}$$

Mais ce modèle de la partie opérative ne sera pas utilisé pour la piloter !

Schéma général (exemple d'une commande de volets roulants)

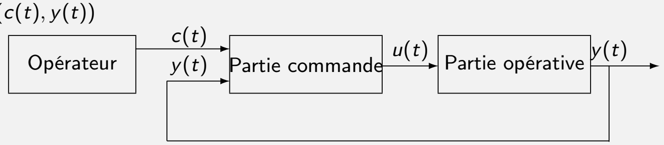
La partie commande C'est le contrôleur à réaliser.
C'est un SED qui utilise :

- la consigne $c = (\mathrm{Op}, \mathrm{Cl})$ de l'opérateur
- les informations $y = (\mathrm{Top}, \mathrm{Btm})$ retournées par les capteurs de la partie opérative

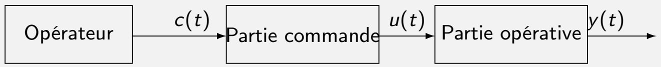
pour commander les préactionneurs $u = (\mathrm{Up}, \mathrm{Dn})$ de la partie opérative.
Son entrée est donc $e = (c, y) = (\mathrm{Op}, \mathrm{Cl}, \mathrm{Top}, \mathrm{Btm})$
Sa sortie est $u = (\mathrm{Up}, \mathrm{Dn})$

Schéma-bloc

Le plus général $e(t) = (c(t), y(t))$



Boucle ouverte $e(t) = c(t)$



Sans partie opérative

Par exemple, en génomique :

- t entier indice du nucléotide le long du brin d'ADN,
- $c(t) \in \{\mathrm{A}, \mathrm{T}, \mathrm{G}, \mathrm{C}\}$,
- $u(t) \in \{0,1\}$, à 1 quand une certaine séquence est détectée.

1re étape : écrire un cahier des charges

Exemple

- Un appui permanent sur le bouton-poussoir de fermeture (resp. ouverture) enclenche la fermeture (resp. ouverture) du volet
- Cependant, la fin de course haut (resp. bas) doit couper l'ouverture (resp. la fermeture) du volet.

Remarque Un cahier des charges est écrit en langage naturel. Il est donc incomplet, ambigu (voire incohérent).
Dans l'exemple, le cas où l'opérateur demande à la fois l'ouverture et la fermeture du volet est oublié.

2^{e} étape : concevoir et formaliser la partie commande

Conception La partie commande est-elle combinatoire ou séquentielle ? etc.

Formalisation Expression dans un langage descriptif : C, VHDL, schéma à contact, logigramme, grafcet…

3^{e} étape : réaliser l'automatisme

Circuits logiques élémentaires (portes, inverseurs…), par exemple :

Circuits TTL (Transistor Transistor Logic)
réalisés avec transistors bipolaires
alimentés sous 5 V

Circuits CMOS (Complementary Metal Oxide Semiconductor)
réalisés avec transistors à effet de champ
alimentés entre 4 et 15 V

Les entrées-sorties sont :

- soit au niveau logique 0 : 0 V
- soit au niveau logique 1 : tension d'alimentation

Automates programmables industriels, API (Programmable Logic Controller)

Micro-contrôleurs (cf. cours au choix SELEC)

Remarque Dans tous les cas, la tension de la technologie employée doit être compatible avec les pré-actionneurs et les capteurs de la partie opérative.

Réalisation d'un automatisme par conception d'une partie commande :

combinatoire si le cahier des charges le permet, traduit :

- en logigramme pour une réalisation avec circuits logiques ;
- en schéma à contacts pour une implantation sur API ;

séquentielle par une rétroaction sur l'entrée d'un système combinatoire, pour une réalisation avec circuits logiques ;

séquentielle traduit en langage grafcet, pour une implantation sur API ;

Définitions succinctes

Variable logique Variable pouvant prendre 2 valeurs (« vrai » ou « faux »)

- « faux » noté 0
- « vrai » noté 1

Algèbre de Boole Opérations et fonctions sur les variables logiques

Fonction logique $f : \{0,1\}^n \longrightarrow \{0,1\}^m$
En pratique, une fonction à valeur dans $\{0,1\}^m$ est décrite par m fonctions à valeur dans $\{0,1\}$

Remarque Dans une fonction logique, on appellera :

- Entrée : les antécédents
- Sortie : les images

Algèbre de Boole

Fonctions logiques fondamentales (à valeur dans $\{0,1\}$)

***n* entrées** 2^{2^n} fonctions $f(A_1,\dots,A_n)$ possibles

1 entrée 4 fonctions $f(A)$ possibles :

A	$f(A)=0$	$f(A)=1$	identité $f(A)=A$	négation $f(A)=\bar{A}$
0	0	1	0	1
1	0	1	1	0

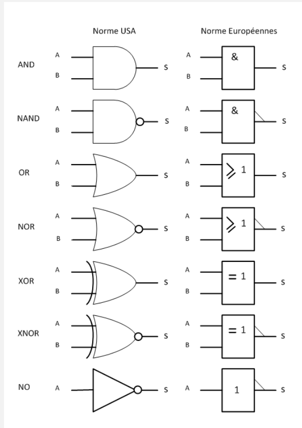
2 entrées 16 fonctions $f(A,B)$ possibles, dont :

A	B	et and $A.B$	non-et nand $\overline{A.B}$	ou or $A+B$	non-ou nor $\overline{A+B}$	ou excl. xor $A\oplus B$	non-ou excl. xnor $\overline{A\oplus B}$
0	0	0	1	0	1	0	1
0	1	0	1	1	0	1	0
1	0	0	1	1	0	1	0
1	1	1	0	1	0	0	1

Remarques
- et : produit logique (correspond au produit arithmétique)
- ou : somme logique (min(1, somme arithmétique))
- non-ou exclusif : coïncidence

Algèbre de Boole

Symbolisation



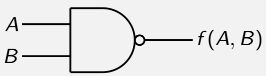
Le cercle (ou triangle) en sortie indique une négation.

Algèbre de Boole

Représentation d'une fonction logique

Il existe de multiples représentations *équivalentes* d'une fonction $f : (A,B) \mapsto f(A,B)$.

Graphique Par exemple, logigramme
Fonction non-et :



Tabulaire Par exemple, la table de vérité
Fonction non-et :

A	B	$f(A,B)$
0	0	1
0	1	1
1	0	1
1	1	0

Algébrique Une expression logique (nombreuses possibilités)
Fonction non-et :
$f(A,B)=\overline{A.B}$
$f(A,B)=\bar{A}+\bar{B}$

Algèbre de Boole

Lois fondamentales

Commutativité $A.B = B.A$
$A+B = B+A$

Associativité $A.(B.C) = (A.B).C = A.B.C$
$A+(B+C) = (A+B)+C = A+B+C$

Element neutre $A.1 = A$
$A+0 = A$

Element absorbant $A.0 = 0$
$A+1 = 1$

Element symétrique $A.\bar{A} = 0$
$A+\bar{A} = 1$

Involution $\bar{\bar{A}} = A$

Idempotence $A.A = A$
$A+A = A$

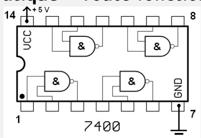
Distributivité $A.(B+C) = A.B+A.C$

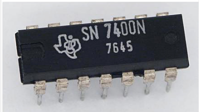
Loi de De Morgan $\overline{A.B} = \bar{A} + \bar{B}$
$\overline{A + B} = \bar{A}.\bar{B}$

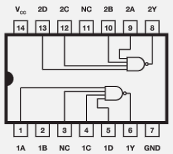
Se généralise par récurrence à une nombre quelconque de variables
$\overline{A.B.C.D} = \bar{A} + \bar{B} + \bar{C} + \bar{D}$
$\overline{A + B + C + D} = \bar{A}.\bar{B}.\bar{C}.\bar{D}$

Conséquence théorique Toute fonction logique peut s'écrire à l'aide de non-et (ou non-ou, ou non-ou exclusif) (l'inverseur est une porte non-et particulière, à une entrée).

Conséquence pratique Toute fonction logique peut être câblée à l'aide de portes non-et







7420
les entrées non utilisées doivent être mise à 1 (5V) ou reliées à une entrée utilisée.

Exercices

Exercice

A l'aide des règles de calcul booléen, simplifier les expressions logiques suivantes :

- $A + A.B$
- $A + \bar{A}.B$
- $A.(A + B)$
- $(A + B).(A + \bar{B})$

Objectif Réaliser la partie commande d'un automatisme.

Hypothèse On suppose dans cette partie que le cahier des charges entraîne une partie commande combinatoire $u(t) = g(e(t))$ qu'il faut déterminer.
On notera $u = g(e)$ pour simplifier.

Démarche
- déterminer les tables de vérité des fonctions logiques (une pour chaque bit de u) à partir du cahier des charges ;
- en déduire une expression simple de ces fonctions logiques ;
- les traduire :
  - en logigramme : très proche du câblage de composants logiques ;
  - en schéma à contact : en vue de la programmation d'un API ;
- implanter effectivement la partie commande.

Forme canonique disjonctive

Principe On obtient la forme canonique disjonctive à partir de la table de vérité de la façon suivante : c'est une somme logique de « mintermes »
- un minterme par ligne où la sortie est à 1
- chaque minterme est un produit des entrées (celles à 0 sont inversées)

Exemple $e = (A, B)$ a deux composantes binaires
$u = (U)$ a une composante binaire

Soit la table de vérité

A	B	U
0	0	1
0	1	1
1	0	0
1	1	1

La forme canonique disjonctive s'écrit : $U = \bar{A}.\bar{B} + \bar{A}.B + A\,B$

Remarque sur cet exemple l'expression logique se simplifie en $U = \bar{A} + B$
C'est l'implication logique $A \rightarrow B$

Obtention d'une expression logique à partir d'une table de vérité

Forme canonique conjonctive

Principe On obtient la forme canonique conjonctive à partir de la table de vérité de la façon suivante : c'est un produit logique de « maxtermes »

- un maxterme par ligne où la sortie est à 0
- chaque maxterme est une somme des entrées (celles à 1 sont inversées)

Exemple Soit la table de vérité de l'implication logique

A	B	U
0	0	1
0	1	1
1	0	0
1	1	1

La forme canonique conjonctive s'écrit : $U = \bar{A} + B$

Exercice

Ecrire les formes canoniques du XOR et du XNOR

Simplification par tableaux de Karnaugh

Principe

Objectif Définir une règle de simplification d'expression logique à faible nombre d'entrées (3, 4)

Méthode

- On réécrit le table de vérité sous forme de tableau

A B \ C	0	1
0 0	1	0
0 1	1	0
1 1	1	0
1 0	1	1

A B \ C D	0 0	0 1	1 1	1 0
0 0	1	0	1	0
0 1	1	0	1	1
1 1	1	0	1	1
1 0	1	1	1	0

  Les combinaisons (A, B) et (C, D) sont rangées en binaire réfléchi : 1 changement de bit à la fois
- regroupement des 1 adjacents en rectangles de cardinal 1, 2, 4 ou 8.
  ! bords gauche et droit adjacents, bords haut et bas adjacents
- l'expression logique est une somme de produits
  - un produit par regroupement
  - pour chaque regroupement, les entrées toujours à 1 sont conservées, celles à 0 sont inversées, les entrées qui varient disparaissent.

Exemple Le tableau de Karnaugh à 3 entrées ci-dessus donne $U = A.\bar{B} + \bar{C}$

Simplification par tableaux de Karnaugh

Remarques

- ne pas oublier de 1 isolés, même s'ils ne peuvent être regroupés
- penser à la structure cylindrique du tableau
- regroupements les plus grands possibles
- le moins possible de regroupements

Exercice

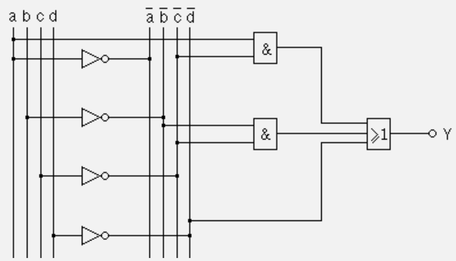
Ecrire une expression logique simplifiée correspondant au tableau de Karnaugh à 4 entrées de la page précédente

Transcription

Logigramme

principe C'est pratiquement le schéma de câblage à effectuer si on réalise la partie commande à l'aide de composants logiques.

exemple

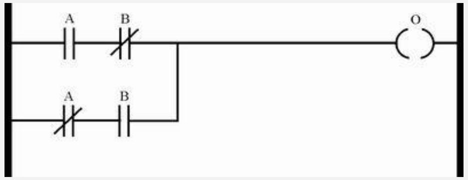


Ce logigramme correspond à la fonction logique $Y = a.\bar{c} + \bar{b}.\bar{c} + \bar{d}$

Schéma à contacts (ladder diagram)

principe Il représente la structure des premiers API.
On suppose qu'il y a une tension entre les extrémités du schéma.
Les entrées sont des contacts (une même entrée peut apparaître plusieurs fois).
Les sorties sont des bobines.

exemple



Ce schéma correspond à la fonction logique $O = A.\bar{B} + \bar{A}.B$,
c'est-à-dire $A \oplus B$

Cahier des charges

- Un appui permanent sur le bouton-poussoir de fermeture (resp. ouverture) enclenche la fermeture (resp. ouverture) du volet
- Cependant, la fin de course haut (resp. bas) doit couper l'ouverture (resp. la fermeture) du volet.
- Si l'opérateur demande à la fois l'ouverture et la fermeture, tout mouvement est interrompu.
- Si les 2 capteurs de fin de course sont actifs, cela indique une panne, tout mouvement est interrompu

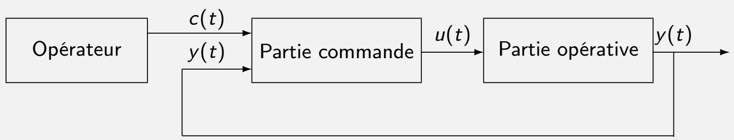
On note :

- $(\mathrm{Op}, \mathrm{Cl})$ les consignes de l'opérateur
- $(\mathrm{Top}, \mathrm{Btm})$ les capteurs de fin de course
- $(\mathrm{Up}, \mathrm{Dn})$ la commande de la partie opérative, émise par la partie commande.

Questions

- Ecrire une expression des fonctions logiques $\mathrm{Up}(\mathrm{Op}, \mathrm{Cl}, \mathrm{Top}, \mathrm{Btm})$ et $\mathrm{Dn}(\mathrm{Op}, \mathrm{Cl}, \mathrm{Top}, \mathrm{Btm})$
- Transcrire en logigramme et schéma à contact

Objectif On cherche à réaliser un automatisme



c'est-à-dire réaliser la « partie commande » qui,
à partir de $t \mapsto e(t)$ avec $e(t) = (c(t), y(t))$
(à partir de $t \mapsto e(t)$ avec $e(t) = c(t)$ en boucle ouverte),
génère $t \mapsto u(t)$ pour piloter la « partie opérative »
en fonction d'un cahier des charges.

Problème Si le cahier des charges conduit à la possibilité suivante :

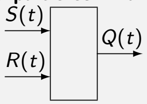
$$\exists (t_1, t_2) \quad e(t_1) = e(t_2) \text{ et } u(t_1) \neq u(t_2)$$

alors il n'existe pas de système combinatoire réalisant la « partie commande ».
Il faut un système séquentiel, c'est-à-dire tel que la sortie $u(t)$ dépend de l'historique $\{e(\tau) \mid \tau \leqslant t\}$.

Principe

Cahier des charges La partie opérative a une entrée binaire mais pas de capteurs (on réalise une commande en boucle ouverte).
L'opérateur dispose de 2 boutons-poussoirs, l'un pour mettre en marche la partie opérative, l'autre pour l'arrêter.

Solution pour la partie commande : la bascule RS (Reset, Set)



2 entrées binaires : $e(t) = (S(t), R(t))$
1 sortie binaire : $u(t) = (Q(t))$

$S(t)$	$R(t)$	$Q(t+dt)$	
0	0	$Q(t)$	Mémoire
0	1	0	Mise à 0
1	0	1	Mise à 1
1	1	1	Mise à 1 prioritaire

Remarque Il s'agit d'une bascule à enclenchement prioritaire.
Si on prend $Q(t+dt) = 0$ sur la dernière ligne, déclenchement prioritaire.

dt Retard symbolisant le temps de réponse de la bascule.

Table de vérité

Exercice

Remplir la table de vérité et le tableau de Karnaugh suivants.

$S(t)$	$R(t)$	$Q(t)$	$Q(t+dt)$
0	0	0	
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	

$S(t)$	$Q(t)$ / $R(t)$	0	1
0	0		
0	1		
1	1		
1	0		

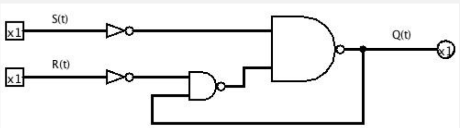
En déduire une expression simple de la fonction logique

$$Q(t+dt) = f(Q(t), R(t), S(t))$$

Vérifier que le logigramme page suivante convient.

Réalisation

Logigramme



Remarque Logigramme obtenu avec logiciel LOGISIM

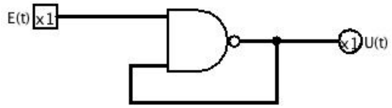
Remarque Un système séquentiel est obtenu par rétro-action (ici de la sortie).
Cette rétro-action, interne à la partie commande, ne doit pas être confondue avec la rétro-action dans le schéma global d'un automatisme en boucle fermée.

Remarque Dans le schéma ci-dessus, on a négligé le temps de propagation *dt*
C'est le principe de la logique *asynchrone*. Ceci ne peut fonctionner que parce que pour toute combinaison (S_0, R_0) de l'entrée, il existe une sortie *stable* Q_0 :

$$Q_0 = f(Q_0, (R_0, S_0))$$

Exercice rapide

Ce montage, d'entrée $E(t)$, de sortie $U(t)$, peut-il fonctionner ?



Remarque En logique *synchrone*, nous apporterons une modification qui permettra à ce montage de fonctionner.

Formalisation générale

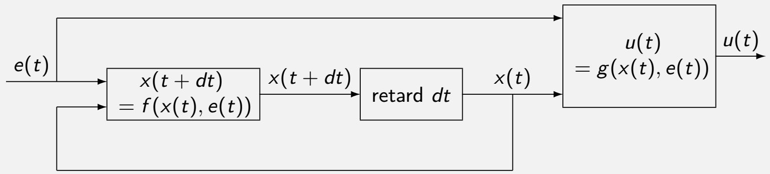
Représentation dans l'espace d'état : la machine de Mealy

Soit un SED d'entrée $e : t \mapsto e(t)$, de sortie $u : t \mapsto u(t)$.
On suppose qu'il existe une fonction $x : t \mapsto x(t)$ (l'état) à valeurs discrètes prises dans un alphabet, et deux fonctions logiques f et g, telles que :

$$\begin{cases} u(t) & = g(x(t), e(t)) \quad \text{fonction de sortie} \\ x(t+dt) & = f(x(t), e(t)) \quad \text{fonction de transition} \end{cases}$$

Formalisation générale

Schéma bloc de la machine de Mealy



Machine de Mealy

Remarque On peut souvent donner un nom symbolique aux valeurs de l'alphabet de l'état : « On », « Off », « marche », « arrêt », « veille »...

Câblage avec composants logiques Chaque liaison comprend en pratique plusieurs fils ; par exemple, si l'état peut prendre N valeurs différentes, il faut en sortie du bloc « transition » n fils avec $2^n \geqslant N$ pour *coder l'état*.

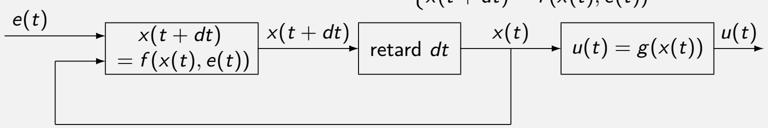
Retard dt Bloc symbolique, souvent non représenté ; pourtant, il existe un retard, *sans doute différent selon les composantes binaires codant l'état*.

Formalisation générale

Modèles simplifiés

Machine de Moore

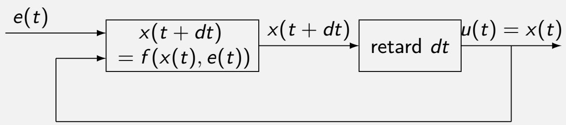
$$\begin{cases} u(t) & = g(x(t)) \\ x(t+dt) & = f(x(t), e(t)) \end{cases}$$



Machine de Moore

Retour de sortie

$$\begin{cases} u(t) & = x(t) \\ x(t+dt) & = f(x(t), e(t)) \end{cases}$$



Remarque : c'est le cas de la bascule RS (exemple introductif)

Formalisation générale

Vocabulaire

Etat stable La valeur de l'état x_0 est dite stable pour la valeur de l'entrée e_0 si :

$$x_0 = f(x_0, e_0)$$

Attention ! Une condition nécessaire pour qu'un automatisme séquentiel *asynchrone* puisse fonctionner est que, pour toute valeur d'entrée, il existe un état stable.

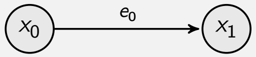
Représentation par graphe d'état

Principe

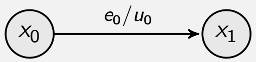
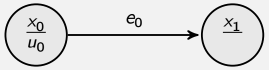
Objectif Décrire les fonctions de transition f et de sortie g.

Remarque préliminaire Ci-dessous, x_0 et x_1 désignent deux valeurs possibles de l'état,
e_0 une valeur possible de l'entrée,
u_0 une valeur possible de la sortie

Transition Le système, dans l'état x_0, soumis à l'entrée e_0, transite vers l'état x_1 : $x_1 = f(x_0, e_0)$



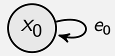
Sortie On complète par :

Machine de Mealy	Machine de Moore
Le système, dans l'état x_0, soumis à l'entrée e_0, produit la sortie u_0 : $u_0 = g(x_0, e_0)$ 	Le système, dans l'état x_0, produit la sortie u_0 : $u_0 = g(x_0)$ 

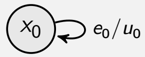
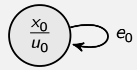
Représentation par graphe d'état

Cas d'un état stable

Transition Le système, dans l'état x_0, soumis à l'entrée e_0, reste dans l'état x_0 : $x_0 = f(x_0, e_0)$



Sortie On complète (rien de nouveau par rapport à la page précédente) par :

Machine de Mealy	Machine de Moore
Le système, dans l'état x_0, soumis à l'entrée e_0, produit la sortie u_0 : $u_0 = g(x_0, e_0)$ 	Le système, dans l'état x_0, produit la sortie u_0 : $u_0 = g(x_0)$ 

Représentation par graphe d'état

Construction générale

En général Plusieurs transitions (flèches) partent d'un état.
Plusieurs transitions arrivent à un état.

Propriétés Un graphe d'état doit être non contradictoire,
et devrait être complet (surtout pour une application critique).

non contradictoire on ne peut transiter d'un état vers plusieurs états
(f est bien une fonction) :
si $f(x_0, e_0)$ est défini, il est unique

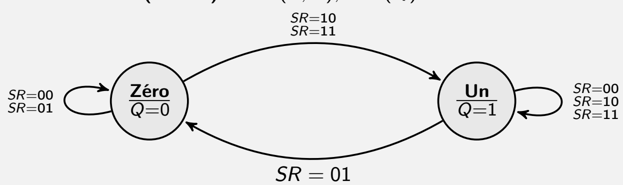
complet toutes les transitions possibles sont décrites :
pour tout (x_0, e_0) possible, $f(x_0, e_0)$ est défini

État initial (à la mise sous tension de l'automate) : cercle double.

Représentation par graphe d'état

Exemple

Graphe d'état de la bascule RS (Moore) $e = (S, R), u = (Q)$

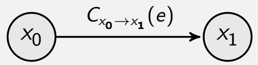


Pour simplifier, entre 2 états, une seule flèche est représentée pour toutes les valeurs possibles de l'entrée qui mènent de l'un à l'autre.

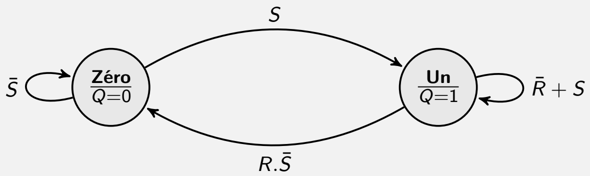
Remarque Ce graphe est complet et non contradictoire.

Représentation par condition sur les transitions

Principe Plutôt que d'énumérer toutes les valeurs d'entrée possibles permettant de transiter d'un état vers un autre état, on écrit la condition logique sur les entrées qui vaut 1 pour ces valeurs.
Le système, dans l'état x_0, soumis à une valeur e de l'entrée telle que $C_{x_0 \to x_1}(e) = 1$, transite vers l'état x_1



Graphe d'état de la bascule RS (Moore) $e = (S, R), u = (Q)$



Complet $\forall (x_0, e) \quad \exists x_1 \quad C_{x_0 \to x_1}(e) = 1$
Non contradictoire $\forall (x_0, e) \quad \forall (x_1, x_2) \quad x_1 \neq x_2 \Rightarrow C_{x_0 \to x_1}(e).C_{x_0 \to x_2}(e) = 0$

Objectif Réaliser la partie commande d'un automatisme.

Hypothèse On suppose dans cette partie que le cahier des charges entraîne une partie commande séquentielle à fonctions logiques f et g qu'il faut déterminer.

Démarche
- déterminer les états possibles à partir du cahier des charges ;
- déterminer le graphe d'état à partir du cahier des charges ;
- fusionner les états redondants si possible ;
- coder les états ;
- déterminer les tables de vérité des fonctions logiques f et g ;
- puis poursuivre comme en logique combinatoire (expression des fonctions logiques, câblage ou API ou micro-contrôleur. . .).

Remarque Si le graphe d'état n'est pas complet, le concepteur peut choisir :
- les transitions qui l'arrange pour minimiser le nombre d'états par fusion, ou obtenir les fonctions logiques les plus simples possibles ;
- réfléchir sur le caractère critique de ce manque, et compléter le graphe.

Chariot

Exercice

Un chariot se déplace vers la gauche (G) ou vers la droite (D).
A la mise sous tension de l'automate, le chariot est automatiquement déplacé vers la gauche (G) jusqu'à la fin de course gauche (A).
Ensuite :
Le bouton de mise en marche (M) enclenche le déplacement vers la droite.
La fin de course droite (B) enclenche le déplacement vers la gauche.
La fin de course gauche (A) interrompt le déplacement vers la gauche.

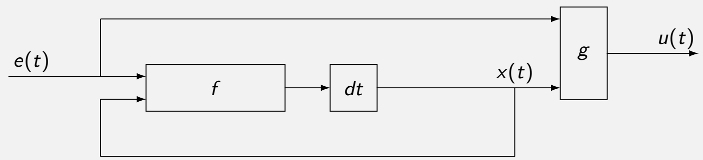
a) Bilan des entrées-sorties de l'opérateur, de la partie commande, de la partie opérative.

b) Graphe d'état complet non contradictoire d'une machine de Moore à 3 états
(« Veille », « Déplacement vers la droite », « Déplacement vers la gauche »).

c) Graphe d'état complet non contradictoire d'une machine de Mealy à 2 états
(« Pas vers la droite », « Vers la droite »).

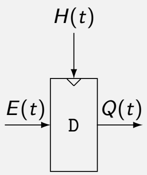
Critique de l'asynchrone

On rappelle le schéma général d'une partie commande séquentielle (machine de Mealy).



Les différents retards de propagation des composants du système combinatoire f peuvent conduire à des états transitoires indésirables rendant le système séquentiel imprédictible, voire instable.

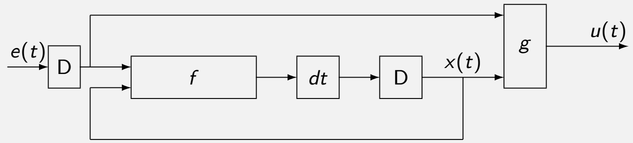
La bascule D



L'horloge H est un signal carré périodique de période T.
La sortie Q, initialement à 0, prend la valeur de l'entrée E aux fronts montants de H.
Une bascule D n'est autre qu'un échantillonneur-bloqueur.

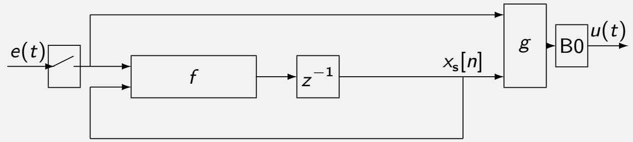
Fonctionnement synchrone : insertion de la bascule D

Insertion dans le schéma initial



Les 2 bascules sont synchrones (même horloge de période T).

Schéma équivalent Si $T > dt$:



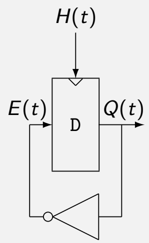
où $x_s[n]$ désigne l'état échantillonné : $x_s[n] = x(nT)$ et z^{-1} le retard d'un pas d'échantillonnage.
On obtient la représentation à temps discret :

$$\begin{cases} u_s[n] &= g(x_s[n], e_s[n]) \quad \text{fonction de sortie} \\ x_s[n+1] &= f(x_s[n], e_s[n]) \quad \text{fonction de transition} \end{cases}$$

Remarque sur la stabilité

Il n'est plus nécessaire qu'il existe un état stable pour toute entrée.
L'instabilité peut même être recherchée, par exemple pour créer un chenillard :

Bascule D câblée en diviseur de fréquence par 2

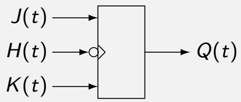


Q est inversée à chaque front montant de H.

- Si H est une horloge de période fixe T, Q est de période $2\,T$ (la fréquence est divisée par 2.
- Si on dispose d'un bouton-poussoir pour allumer et éteindre la lumière, on le relie à l'entrée horloge de ce montage. Le niveau logique est inversé à chaque impulsion.
- Ce montage peut servir à réaliser une bascule JK.

Des bascules aux compteurs et temporisateurs

Bascule JK (sur front descendant)



Si t est l'instant d'un front descendant de H :

$J(t)$	$K(t)$	$Q(t+dt)$	
0	0	$Q(t)$	Mémoire
0	1	0	Mise à 0
1	0	1	Mise à 1
1	1	$\bar{Q}(t)$	Inversion

Une bascule JK peut être munie d'une entrée de remise à zéro asynchrone (indépendante des fronts de H).
Elle peut servir à la réalisation de compteurs.

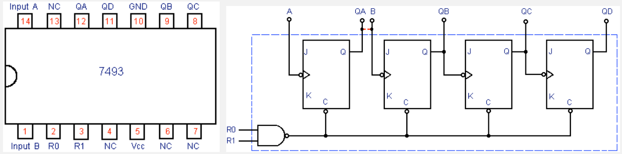
Des bascules aux compteurs et temporisateurs

Compteurs

Définition Un compteur binaire n bits est une système séquentiel dont les n sorties binaires Q_1, $Q_2, \ldots, Q_n$ codent le nombre de fronts montants (ou descendants) de l'entrée. Q_1 est le bit de poids faible.
A chaque front descendant de Q_i, il y a inversion de Q_{i+1}.
C'est un compteur modulo 2^n.

Exercice : chronogramme d'un compteur 2 bits

Exemple : compteur intégré 7493 réalisé en technologie TTL



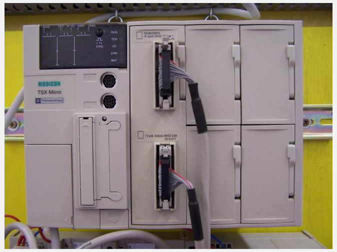
Temporisateur Ce n'est qu'un compteur d'impulsions de l'horloge d'un système séquentiel synchrone.

Généralités

Automate programmable industriel (API)

Définition « technologique » Un API est un dispositif :

- destiné à la commande séquentielle de processus industriels,
- électronique,
- équipées d'entrée-sortie logiques (Télémécanique TSXμ : 0-24 V),
- fonctionnant à une cadence fixée T (par exemple : 50 ms),
- *programmable*



Définition formelle Un API est une machine de Mealy synchrone

Généralités

Graphe de Commande Étape-Transition (GRAFCET)

Définition

- Langage graphique normalisé,
- destiné à formaliser le cahier des charges d'un automatisme séquentiel,
- comme le graphe d'état, mais plus intuitif.

Extrait de la norme IEC 60848 (2002) *Specification language for the functional description of the behavior of the sequential part of a control system.*

Application à la programmation d'un API

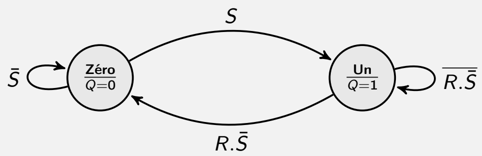
- On écrit un grafcet (algorithme en langage GRAFCET) en fonction du cahier des charges,
- puis on le transcrit dans le langage de programmation de l'API.

Remarque Le « Sequential Function Chart » (SFC) est un langage graphique de programmation d'API très proche du GRAFCET.

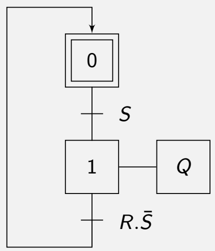
Généralités

Exemple introductif : grafcet d'une bascule RS

Rappel : graphe d'état $e = (S, R), u = (Q)$



grafcet

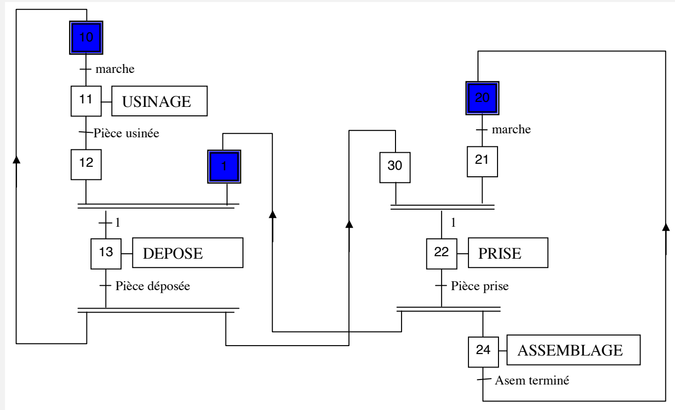


- Alternance étape-transition.
- Une étape est active ou inactive.
- A une transition est associée une réceptivité.
- A une étape peuvent être associées des actions.

Attention ! Ce grafcet est à *séquence unique*, une seule étape est active à la fois ; ce n'est pas général ⇒ Ne pas confondre « état de l'automate » et « étape du grafcet »

Généralités

Un exemple industriel plus complexe



Généralités

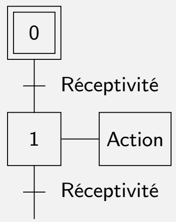
Etat de l'automate

L'état de l'automate formalisé par un grafcet contient :

- la liste des étapes actives et inactives,
- l'état de bascules, compteurs, temporisateurs...

Principes fondamentaux de parcours d'un grafcet

Etape



Un grafcet est constitué

- d'étapes,
- de transitions,
- d'arcs orientés.

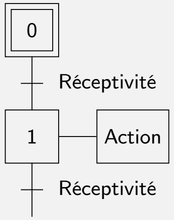
Une étape

- est représentée par un carré numéroté
  - numéro sans importance,
  - mais 2 étapes ne peuvent avoir le même numéro ;
- est représentée par un carré double si elle est *initiale* (active au démarrage de l'exécution) ;
- est active ou inactive ; à l'étape n°10 est associée la variable booléenne $X10$ valant 1 si l'étape est active, 0 sinon ;
- peut être assortie d'actions telles que
  - l'affectation d'une sortie de l'automate,
  - l'incrémentation d'un compteur...

  (une condition nécessaire à la réalisation de ces actions est l'activation de l'étape).

Principes fondamentaux de parcours d'un grafcet

Transition



Un grafcet est constitué

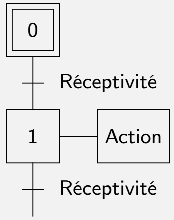
- d'étapes,
- de transitions,
- d'arcs orientés.

Une transition

- est représentée par un trait ;
- est assortie d'une réceptivité : fonction booléenne
  - des entrées de l'automate,
  - des variables d'activation des étapes (X10),
  - d'état de compteurs ou temporisateurs. . .
- Exemple : réceptivité $a.\uparrow b.\overline{X12}.[C1{=}10]$ vraie si :
  - l'entrée a de l'automate est à 1,
  - il y a front montant de l'entrée b,
  - l'étape 12 n'est pas active,
  - et le compteur $C1$ est à 10.
- Exemple : réceptivité $\underline{1}$ (ou =1) toujours vraie.

Principes fondamentaux de parcours d'un grafcet

Liaison



Un grafcet est constitué

- d'étapes,
- de transitions,
- d'arcs orientés.

Un arc relie transitions et étapes selon la règle fondamentale :

alternance étape-transition

Principes fondamentaux de parcours d'un grafcet

Règle fondamentale

Une transition est franchie si

- toutes les étapes immédiatement en amont sont actives,
- la réceptivité associée est vraie.

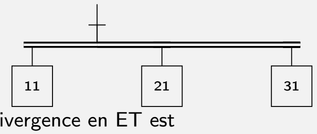
Et cela provoque

- la désactivation des étapes immédiatement en amont,
- l'activation des étapes immédiatement en aval.

Principes fondamentaux de parcours d'un grafcet

Séquences simultanées

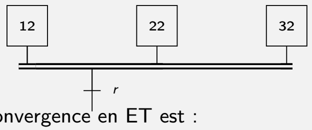
Divergence en ET



Une divergence en ET est

- précédée d'une transition,
- suivie d'étapes activées simultanément au franchissement de la transition.

Convergence en ET

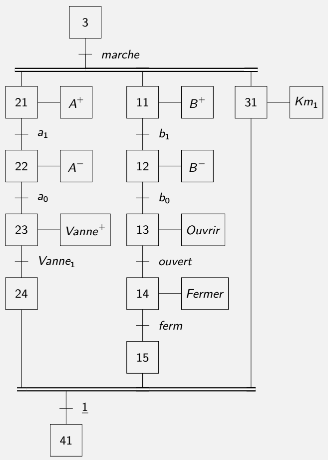


Une convergence en ET est :

- précédée d'étapes,
- suivie d'une transition franchie si
  - les étapes sont toutes actives,
  - la réceptivité r est vraie

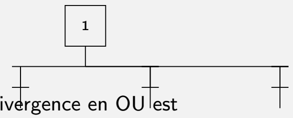
Pratique courante :
pas d'actions associées à ces étapes (étapes *d'attente*),
et réceptivité toujours vraie.

Séquences simultanées : utilisation typique



Sélection de séquence

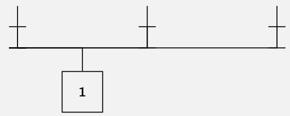
Divergence en OU



Une divergence en OU est

- précédée d'une étape,
- suivie de transitions.

Convergence en OU



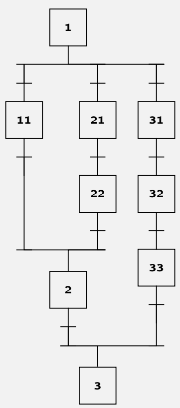
Une convergence en OU est :

- précédée de transitions,
- suivie d'une étape.

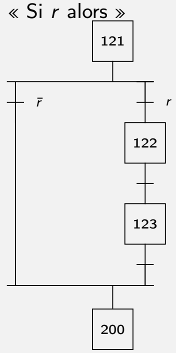
Attention !
Les séquences peuvent être simultanées.

Pratique courante :
Les réceptivités assurent l'exclusivité d'une séquence.

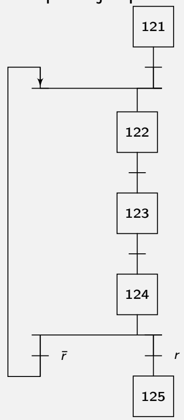
Sélection de séquence : utilisations typiques



Saut d'étapes
« Si r alors »



Reprise de séquence
« Répéter jusqu'à r »



Réceptivités associées à un temporisateur

Principe

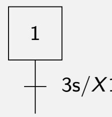
Soit B une variable booléenne.
$t_1/B/t_2$ désigne la variable booléenne dont, par rapport à B :

- les fronts montants sont retardés de t_1,
- les fronts descendants sont retardés de t_2

Un temps manquant est pris à 0.

Chronogramme

Exemples



L'étape 1 reste active 3 sec.

Actions continue, conditionnelle, mémorisée

	action	condition
1 — Action	continue	$X1$
1 — Action (c)	conditionnelle	$X1.c$
1 — S|Action 1 — Action :=1	mise à 1 mémorisée	$\uparrow X1$
1 — R|Action 1 — Action :=0	mise à 0 mémorisée	$\uparrow X1$

Actions sur événement

N'ont de sens que pour une action instantanée
(incrémentation d'un compteur, mise à 0, mise à 1. . .).

	condition
1 — $C1{:=}C1{+}1$	$\uparrow X1$
1 — $C1{:=}C1{+}1$	$\downarrow X1$
1 — $C1{:=}C1{+}1$ ($\uparrow c$)	$X1.\uparrow c$

Actions avec temporisation

Utilise les conventions sur les temporisateurs.

	action	condition
1 — Action ($3s/X1$) 1 — D|Action ($3s$)	retardée	$X1.(3s/X1)$
1 — Action ($\overline{3s/X1}$) 1 — L|Action ($3s$)	limitée	$X1.\overline{3s/X1}$

Cahier des charges
- Un appui, même court, sur le bouton-poussoir, enclenche successivement la fermeture, l'arrêt, l'ouverture, l'arrêt du volet.
- Cependant, la fin de course haut (resp. bas) doit couper l'ouverture (resp. la fermeture) du volet.

On note :
- (BP) les consignes de l'opérateur
- (Top, Btm) les capteurs de fin de course
- (Up, Dn) les actions de la partie opérative, pilotées par la partie commande.

Questions Ecrire un grafcet modélisant ce cahier des charges.

Exercice récapitulatif : commande de volet roulant

Cahier des charges

- Un appui, même court, sur le bouton-poussoir, enclenche l'ouverture ou la fermeture du volet. Un nouvel appui interrompt le mouvement. Un nouvel appui reprend le mouvement précédent.
- La fin de course haut (resp. bas) doit couper l'ouverture (resp. la fermeture) du volet.

On note :

- (BP) les consignes de l'opérateur
- (Top, Btm) les capteurs de fin de course
- (Up, Dn) les actions de la partie opérative, pilotées par la partie commande.

Questions Ecrire un grafcet modélisant ce cahier des charges.