Logique binaire automate

(Automate de type PB)

Définition :

2- Approche structurée des systèmes automatisés
2.2 - Structuration en fonctions principales d'automatisme
2.2.1 - Fonctions et relations internes au système automatisé
- Fonction "Acquérir des informations"
- Fonction "Traiter les informations"
3- Représentation de l'information
3.2- Fonctions logiques
3.2.1 - Outils de description d'une fonction logique:
- table de vérité
- équations booléennes
- logigramme
3.2.2 - Théorème de de Morgan
3.2.3- Opérateurs logiques
3.2.4- Fonction mémoire
- concept d'état
- tables de vérité et équation(s) logique(s)
- fonction mémoire intégrée dans divers constituants.

Objectifs:

Vous devez être capable à la fin de ce T.P:
- De mettre en application les différents cas de logique binaire vus en cours.
- De mettre en évidence le fonctionnement d'un langage de programmation d'automate, la notion de boucle interne.
- De repositionner les différents éléments étudiés dans ce TP, et de les resituer par rapport à la structure générale d'un automatisme.

Moyens mis en oeuvre:

- un automate de type PB15
- un boîtier d'entrées,
- le lexique du PB15
- un schéma explicatif de la structure d'une page automate.

Pré-requis:

- Cours de logique binaire.
- Fonctions logiques.
- Structure générale d'un automatisme.
- Chronogrammes.

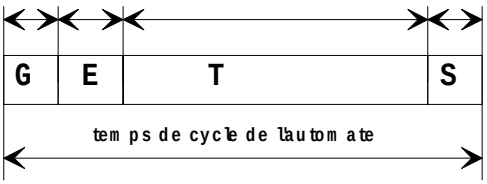
Le travail que vous effectuerez fera l'objet d'un compte-rendu.

Déroulement du TP

Présentation

L'automate que vous allez étudier est un automate de type PB qui a été réalisé conjointement par les sociétés Renault et Merlin-Gerin sous le nom d'April. En 1991 Merlin-Gerin a fusionné avec Télémécanique. Ce groupe est passé sous la direction de Schneider. Renault continue l'étude des automatismes par sa branche Renault-automation.

L'automate PB15 est le plus petit automate de la gamme PB, il est synchrone, il ne possède qu'une seule page de mémoire et possède 16 entrées sorties. Un automate est dit synchrone lorsque son fonctionnement est rythmé par une horloge. De plus cet automate a la particularité d'être synchrone en traitement et en entrées-sorties, c'est à dire que sa durée de cycle est constante.
Un cycle d'automate comprend la lecture des entrées, le traitement du programme, l'affectation des sorties et un temps supplémentaire dévolue à la gestion interne de l'automate (chien de garde et autres...).

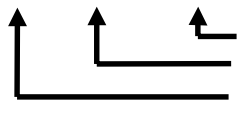


Le PB15 est un excellent outil d'apprentissage car il permet de traiter une multitude de cas des plus simples aux plus complexes. La mise en œuvre de la programmation du PB15 par pupitre est très facile d'accès et correspond aux règles de logique que vous connaissez.

Exemples :

☺ Programmation du calcul logique d'une fonction "ET" avec deux entrées 000 et 001 avec envoi du résultat sur la sortie 020 de l'automate.

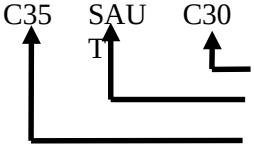
```
C30   SI    000   ← entrée
C31   SI    001   ← entrée
C32   ET    020   ← traitement et affectation de la sortie
C33   SAU   C30   ← saut à l'adresse initiale
      T           ← non opérande
```



variables traitées
fonctions ou opérandes
adresses du programme

☺ Programmation du calcul logique d'une fonction "ET" avec stockage du résultat sur un bit interne, puis utilisation de cette information dans le déroulement d'un programme :

```
C30   Sl     000    ← entrée
C31   Sl     001    ← entrée
C32   ET     A00    ← stockage sur bit interne
C33   Sl     A00    ← récupération de l'information
C34   ET     020    ← traitement et sortie
C35   SAUT   C30    ← saut à l'adresse initiale
```



variables traitées
fonctions ou opérateurs
adresses du programme

☺ Vous avez à votre disposition un automate muni d'un clavier. Sur ce clavier, chaque touche a une fonction qui lui est associée et une lettre ou un chiffre. L'automate reconnaît automatiquement s'il s'agit d'une lettre, d'un chiffre ou d'une commande en fonction du curseur sur l'écran.

Le début d'un programme, car ici il s'agit bien d'un programme, commence à l'adresse C30 pour finir à l'adresse 0E9F. **Il faut impérativement boucler un programme** pour que celui-ci soit cyclique, c'est à dire qu'il se déroule en continu. Pour en revenir à notre programme, on rajoutera, **TOUJOURS**, un saut arrière à la dernière ligne du programme pour qu'il puisse boucler sur lui même.

Exemple:

```
Début de programme   C30   SI     000   ←┐
                     C31   SI     001    │
                     C32   ET     A00    │ retour au début
                     C33   SI     A00    │ du programme.
                     C34   ET     020    │
Fin de programme     C35   SAUT   C30   ←┘
                                  ↖ adresse de retour
```

On peut voir à l'adresse C32 que la fonction logique "ET". Comme on travaille sur un programme et non pas sur des composants, on peut utiliser le nombre d'entrée que l'on veut. La fonction logique prendra en compte toutes les entrées immédiatement supérieures. Vous pouvez voir sur le lexique certaines fonctions utilisables pour des automates de type PB. Toutes les fonctions que vous connaissez sont réalisables par:

Fonctions logiques:

Dans ce premier cas d'application nous allons utiliser la notion de bit interne de la mémoire de l'automate. Il faut savoir que la mémoire d'un automate est agencée un peu à la manière d'une rue. A savoir que si vous souhaitez connaître le prix d'un kg de viande vous allez chez votre boucher à l'adresse de celui-ci dans la rue choisie pour y chercher l'information, de même si vous souhaitez connaître le prix d'un kg de pain vous allez chez votre boulanger à l'adresse de celui-ci dans la rue choisie pour y chercher l'information. Cet exemple simpliste a pour objet de vous sensibiliser à cette logique qui paraît relativement simple mais qui, appliquée à un automate peut sembler un peu plus complexe.

En effet un automate ne comprend que deux choses: présence ou absence de signal électrique, état 1 ou état 0. Cette information est alors repérée et stockée sous forme de bit. Suivant la position de cette information (voir TP sur la numération en deuxième série de TP) I'automate réagira ou ne réagira pas à l'ordre qui lui est donné. On va donc être amené à utiliser ce que l'on appelle une base de comptage. Il existe une multitude de base de comptage, la base 2 qui est utilisée en logique binaire, la base 8, qui est la base octale, la base 10 qui est la base la plus fréquemment utilisée et que l'on appelle la base décimale, la base 12 qui est la base phénicienne, ne vous êtes vous jamais demandé pourquoi on utilisait une douzaine d'oeufs ou douze mois par an, cette base n'est plus que très peu utilisée, et enfin la base 16 connue sous le nom de base hexadécimale. Cette base est très utilisée en informatique industrielle car c'est une base multiple de 2 et fera l'objet d'une étude ultérieure. Ce qu'il faut savoir sur la base hexadécimale c'est que l'on compte de 0 à 15 avant de changer de "seizaine". Comme on est limité par les chiffres, notre base de comptage n'utilise que les neuf chiffres dits "arabes" plus le 0, l'astuce consistera à mélanger les chiffres et les lettres. Ce qui nous donnera pour compter de 0 à 15, soient l6 valeurs : 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F, la dernière valeur F valant 15.

Remarque : la valeur immédiatement suivante sera $10_{(hex)}$ qui vaudra $16_{(10)}$

Tout ce préambule pour vous expliquer que la première adresse de l'automate se trouve en C00, que le premier bit mémoire trouvé s'appelle A00 et que le dernier bit mémoire de cette adresse s'appelle AOF. Ce qui donne 16 bits internes de mémoire à l'adresse C00. Vous pourrez retrouver ces variables sur le document : "structure d'une page mémoire d'un automate de type PB15".

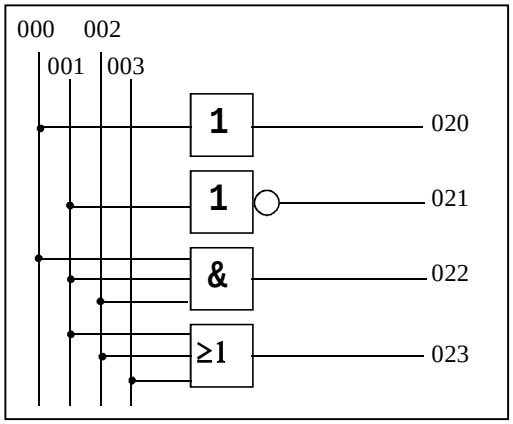
Remarque: le PB15 possède 4 adresses de mémoire C00, C01, C02, C03, soient 16 x 4 = 64 bits internes de mémoire.

Sl : condition d'entrée
Sl/: négation de la condition d'entrée
ET : "ET" logique
OU : "OU" logique

1 - Travail à réaliser :

Réalisez le programme de l'ensemble des fonctions de base correspondant au logigramme ci-contre en utilisant le lexique du PBl5, testez ce programme et reportez ce programme sur votre compte-rendu de TP.

Remarque : vous devez traiter l'ensemble des cas simultanément.



2 - Réalisez les fonctions suivantes à deux entrées :

Remarque : pour chacun des cas suivant vous réaliserer deux programmes, un programme sans bit interne et un programme utilisant des bits internes.

NOR
NAND
INHIBITION

3 - Réalisez la fonction suivante à deux entrées :

MODULO 2 (OU exclusif) $S \equiv a.\overline{b} + \overline{a}.b$

- 1er cas: affichez directement le résultat en sortie. Testez votre programme. Que se passe-t-il ?
- 2ème cas: stockez les résultats intermédiaires sur bit interne avant de les traiter pour envoyer le résultat en sortie. Testez votre programme. Que se passe-t-il ?
- Réalisez alors la fonction ET inclusif en vous appuyant sur la fonction Ou exclusif.

4 - Fonctions mémoires:

Vous allez travailler maintenant à la réalisation de 2 types de mémoires. Pour ce, 2 fonctions supplémentaires de l'automate doivent être utilisées:

MU, qui est l'abréviation de Mise à Un.
MZ, qui est l'abréviation de Mise à Zéro.

La mise inconditionnelle à 1 d'une sortie ou d'un bit interne se fait de la façon suivante: ADR MU 020 ou ADR MU A00 (ADR étant l'adresse du pas de programmation, par exemple C30 ou C44), le résultat sera la mise à 1 de 020 ou A00.

Problème : 000 étant l'entrée à mémoriser, 001 étant la remise à zéro de la mémoire et 020 étant l'indication de l'état de la sortie mémorisée, on vous demande de réaliser le programme d'une fonction mémoire à marche prioritaire, puis de réaliser une fonction mémoire à arrêt prioritaire.

Chaque programme devra être reporté sur votre compte rendu.

Structure d'une page mémoire d'un automate de type PB15

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0C00	A00	A01	A02	A03	A04	A05	A06	A07	A08	A09	A0A	A0B	A0C	A0D	A0E	A0F
0C01	A10	A11	A12	A13	A14	A15	A16	A17	A18	A19	A1A	A1B	A1C	A1D	A1E	A1F
0C02	A20	A21	A22	A23	A24	A25	A26	A27	A28	A29	A2A	A2B	A2C	A2D	A2E	A2F
0C03	A30	A31	A32	A33	A34	A35	A36	A37	A38	A39	A3A	A3B	A3C	A3D	A3E	A3F

0C1E	**décalage**
0C1F	**décalage opérations arithmétiques et logiques**
0C20	**adresse suivant un saut absolu**
0C21	
0C22	**registre de l'UC utilisé pour les codes DE, APL, IDL**
0C23	**registre de l'UC utilisé pour les codes DE, APL, IDL**
0C28	**registre index héxadécimal**
0C29	**registre index héxadécimal**
0C2A	**registre index héxadécimal**
0C2B	**registre index héxadécimal**
0C2C	**compteur ordinal**
0C2D	**N° de page des opérandes - Valeur de l'index**
0C2E	**indicateurs**
0C2F	**SAUT adresse du programme d'interruption**
0C30	**adresse du début du programme maître**
	programme
0EFF	**adresse de la fin du programme maître**