William Bolton

L'USINE NOUVELLE

AUTOMATES PROGRAMMABLES INDUSTRIELS

Traduction de Hervé Soulard

2e édition

Copyright © 2015 Dunod.

DUNOD

This sixth edition of the work entitled
PROGRAMMABLE LOGIC CONTROLLERS
[ISBN 9780128029299] by William BOLTON
is published with arrangement with ELSEVIER LIMITED
of The Boulevard, Langford Lane, Kidlington, Oxford, OX 1GB, UK.

Cet ouvrage est la traduction en langue française, par les Éditions Dunod,
de *Programmable Logic Controllers, sixth edition*
de William Bolton
Copyright © 2015, Elsevier Ltd. All rights reserved.

Illustration de couverture : © GraphicObsession - Fotolia.com

Le pictogramme qui figure ci-contre mérite une explication. Son objet est d'alerter le lecteur sur la menace que représente pour l'avenir de l'écrit, particulièrement dans le domaine de l'édition technique et universitaire, le développement massif du photocopillage.
Le Code de la propriété intellectuelle du 1er juillet 1992 interdit en effet expressément la photocopie à usage collectif sans autorisation des ayants droit. Or, cette pratique s'est généralisée dans les établissements d'enseignement supérieur, provoquant une baisse brutale des achats de livres et de revues, au point que la possibilité même pour les auteurs de créer des œuvres nouvelles et de les faire éditer correctement est aujourd'hui menacée.
Nous rappelons donc que toute reproduction, partielle ou totale, de la présente publication est interdite sans autorisation de l'auteur, de son éditeur ou du Centre français d'exploitation du droit de copie (CFC, 20, rue des Grands-Augustins, 75006 Paris).

© Dunod, Paris, 2010, 2015, pour la traduction française
5 rue Laromiguière
75005 Paris

ISBN 978-2-10-074219-6

Le Code de la propriété intellectuelle n'autorisant, aux termes de l'article L. 122-5, 2° et 3° a), d'une part, que les « copies ou reproductions strictement réservées à l'usage privé du copiste et non destinées à une utilisation collective » et, d'autre part, que les analyses et les courtes citations dans un but d'exemple et d'illustration, « toute représentation ou reproduction intégrale ou partielle faite sans le consentement de l'auteur ou de ses ayants droit ou ayants cause est illicite » (art. L. 122-4).
Cette représentation ou reproduction, par quelque procédé que ce soit, constituerait donc une contrefaçon sanctionnée par les articles L. 335-2 et suivants du Code de la propriété intellectuelle.

Copyright © 2015 Dunod.

TABLE DES MATIÈRES

Préface IX

1 • Automates programmables industriels 1

1.1 Systèmes de commande 1
1.2 Matériel 7
1.3 Architecture d'un API 9
1.4 Systèmes API 12
1.5 Programmes 17
1.6 En résumé 21
1.7 Problèmes 21
1.8 Recherches complémentaires 23

2 • Dispositifs d'entrées-sorties 25

2.1 Dispositifs d'entrées 25
2.2 Dispositifs de sorties 46
2.3 Exemples d'applications 55
2.4 En résumé 59
2.5 Problèmes 59
2.6 Recherches complémentaires 64

3 • Systèmes numériques 65

3.1 Binaire 66
3.2 Octal et hexadécimal 67
3.3 Décimal codé en binaire 68
3.4 Nombres en binaire, octal, hexadécimal et BCD 69
3.5 Arithmétique binaire 70
3.6 Données d'un API 74
3.7 Systèmes logiques combinatoires 74
3.8 Systèmes logiques séquentiels 76
3.9 En résumé 79
3.10 Problèmes 80
3.11 Recherches complémentaires 82

Copyright © 2015 Dunod.

© Dunod – La photocopie non autorisée est un délit.

4 • Traitement des entrées-sorties 83

4.1 Unités d'entrées-sorties 83
4.2 Traitement du signal 90
4.3 Connexions distantes 94
4.4 Réseaux 105
4.5 Exemples de systèmes commerciaux 110
4.6 Traitement des entrées 113
4.7 Adresses des entrées-sorties 115
4.8 En résumé 116
4.9 Problèmes 117
4.10 Recherches complémentaires 120

5 • Langage à contacts et diagrammes de schémas fonctionnels 121

5.1 Langage à contacts 122
5.2 Fonctions logiques 126
5.3 Verrouillage 134
5.4 Sorties multiples 135
5.5 Saisie des programmes 136
5.6 Diagrammes de schémas fonctionnels 137
5.7 Exemples de programmes 145
5.8 En résumé 149
5.9 Problèmes 149
5.10 Recherches complémentaires 158

6 • Programmation IL, SFC et ST 159

6.1 Listes d'instructions 159
6.2 Graphes de fonction séquentielle 168
6.3 Texte structuré 176
6.4 En résumé 182
6.5 Problèmes 182

7 • Relais internes 193

7.1 Relais internes 193
7.2 Programmes en langage à contacts 194
7.3 Relais sauvegardés par batterie 198
7.4 Monostable 199
7.5 Mise à un et mise à zéro 201
7.6 Relais de contrôle maître 206
7.7 En résumé 211
7.8 Problèmes 213

Copyright © 2015 Dunod.

8 • Sauts et appels 223

8.1 Sauts 223
8.2 Sous-programmes 226
8.3 En résumé 229
8.4 Problèmes 229
8.5 Recherches complémentaires 232

9 • Temporisateurs 233

9.1 Types de temporisateurs 233
9.2 Temporisateurs à l'enclenchement 235
9.3 Temporisateurs au déclenchement 239
9.4 Temporisateurs à impulsion 241
9.5 Temporisateurs rémanents 243
9.6 Exemples de programmes 243
9.7 En résumé 246
9.8 Problèmes 246
9.9 Recherches complémentaires 254

10 • Compteurs 255

10.1 Types de compteurs 255
10.2 Programmation 256
10.3 Compteurs progressifs-dégressifs 262
10.4 Temporisateurs et compteurs 262
10.5 Séquenceurs 264
10.6 En résumé 268
10.7 Problèmes 269
10.8 Recherches complémentaires 277

11 • Registres à décalage 279

11.1 Registres à décalage 279
11.2 Programmes en langage à contacts 280
11.3 En résumé 285
11.4 Problèmes 286
11.5 Recherches complémentaires 289

12 • Gestion des données 291

12.1 Registres et bits 291
12.2 Gestion de données 293
12.3 Fonctions arithmétiques 297
12.4 Contrôle en boucle fermée 298

Copyright © 2015 Dunod.

12.5 En résumé 302
12.6 Problèmes 302
12.7 Recherches complémentaires 306

13 • Conception des systèmes 307
13.1 Développement d'un programme 307
13.2 Systèmes sécurisés 311
13.3 Mise en service 316
13.4 Recherche des défauts 320
13.5 Documentation d'un système 325
13.6 En résumé 347
13.7 Problèmes 348
13.8 Recherches complémentaires 351

14 • Programmes 353
14.1 Régulation de température 353
14.2 Séquencement de vannes 359
14.3 Commande d'une bande transporteuse 369
14.4 Contrôle d'un processus 376
14.5 Exemple de sélection : distributeur de boissons 378
14.6 Exemple de comparaison de données : radiateur soufflant 379
14.7 Problèmes 382
14.8 Recherches complémentaires 386

Annexes

1 • Symboles 389

2 • Réponses 395

Index alphabétique 411

Copyright © 2015 Dunod.

PRÉFACE

Ces dernières années, les avancées technologiques ont conduit au développement des automates programmables industriels (API) et à une révolution importante dans l'automatique. Cet ouvrage, une introduction aux API, a pour objectif de faciliter le travail des ingénieurs praticiens qui font leurs premiers pas dans le domaine des automates programmables industriels. Il servira également de support de cours aux étudiants des écoles d'ingénieurs.

Ce livre s'intéresse aux problèmes liés à l'utilisation de nomenclatures et de programmes variés par les différents fabricants. Pour cela, il décrit les principes sous-jacents et les illustre par des exemples provenant de plusieurs fabricants. Voici les thèmes abordés :

- l'architecture de base des API et les caractéristiques des entrées-sorties souvent employées avec ces systèmes ;
- la présentation des systèmes de numération : décimal, binaire, octal, hexadécimal et BCD ;
- une introduction méthodique et minutieuse, avec de nombreuses illustrations, qui décrit la programmation des API, quel que soit le fabricant, et comment utiliser les relais, les temporisateurs, les compteurs, les registres à décalage, les séquenceurs et les fonctions de gestion des données ;
- un examen de la norme CEI 61131-3 et les méthodes de programmation, comme le langage à contacts (*Ladder*), les diagrammes de schémas fonctionnels, les listes d'instructions, le texte structuré et les graphes de fonction séquentielle ;
- de nombreux exemples, des questions à choix multiples et des problèmes pour aider le lecteur à développer les connaissances nécessaires à l'écriture de programmes pour des API (les réponses aux questions et aux problèmes sont données à la fin de l'ouvrage).

Connaissances requises

Le lecteur n'est pas supposé posséder une expérience en informatique. Toutefois, il est préférable qu'il ait des connaissances de base en électricité et en électronique.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

Changements par rapport à la précédente édition

Cette deuxième édition en français est la traduction de la sixième édition anglaise de *Programmable Logic Controllers* de William Bolton. Cette nouvelle édition actualisée a été complétée et enrichie de nouveaux cas d'étude. Dans la précédente édition ont été ajoutés des explications, des exemples et des problèmes, ainsi qu'un résumé des points essentiels à la fin de chaque chapitre. Dans cette nouvelle édition, le Chapitre 1 a été revu afin de comparer les systèmes de commande à relais, à microprocesseur et à API. Par ailleurs, les API commerciaux décrits ont été actualisés, et la présentation des méthodes de programmation des API établies par la norme CEI-61131 a été développée. Au Chapitre 2, nous avons développé le contenu sur les capteurs, avons réécrit la présentation des graphes de fonction séquentielle au Chapitre 6 afin de détailler plus en profondeur cette méthode. La partie du Chapitre 10 qui concerne le séquenceur a été révisée. Le principe du forçage expliqué au Chapitre 13 a été développé, et de nouveaux cas d'études ont été ajoutés au Chapitre 14.

Objectifs du livre

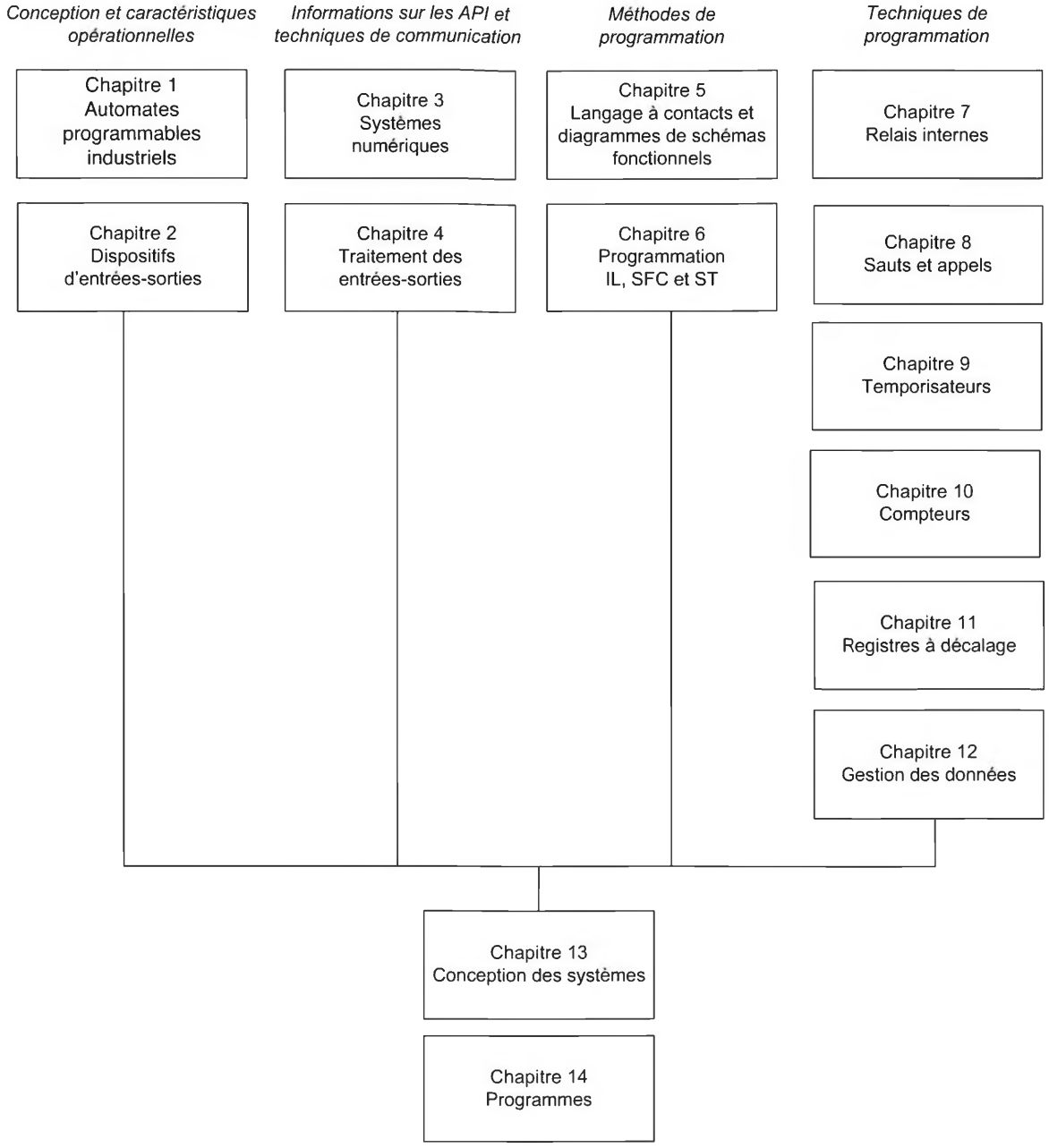
Cet ouvrage a pour objectif d'aider le lecteur à développer les capacités suivantes :

- identifier et expliquer les principales caractéristiques de conception, l'architecture interne et les principes de fonctionnement des automates programmables industriels ;
- utiliser des API de différentes tailles et de différents fabricants ;
- employer les dispositifs d'entrée et de sortie couramment utilisés avec les API, en tenant compte de leurs caractéristiques ;
- expliquer le traitement des entrées et des sorties par les API afin que les systèmes d'entrées-sorties soient correctement utilisés ;
- décrire les liens de communication impliqués dans les systèmes API, en reconnaissant les protocoles et les méthodes de mise en réseau ;
- développer des applications à partir de programmes à contacts fondés sur des relais, des temporisateurs, des compteurs, des registres à décalage, des séquenceurs et des fonctions de gestion des données ;
- identifier les problèmes de sécurité dans les systèmes API ;
- identifier les méthodes employées pour le diagnostic des défauts, les tests et le débogage.

Organisation du livre

La figure de la page suivante illustre la structure de cet ouvrage.

Copyright © 2015 Dunod.



Remerciements

Je remercie les nombreux relecteurs des différentes éditions de cet ouvrage pour leurs remarques et leurs commentaires.

– W. Bolton

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

Copyright © 2015 Dunod.

1 • AUTOMATES PROGRAMMABLES INDUSTRIELS

Ce chapitre constitue une introduction aux automates programmables industriels (API), ainsi qu'à leur fonction générale, leurs formes matérielles et leur architecture interne. Les API sont employés dans de nombreuses tâches d'automatisation, dans différents domaines, comme les processus de fabrication industriels. Cette vue d'ensemble sera complétée par des explications plus détaillées dans les chapitres suivants. Vous trouverez une description résumée des automates programmables industriels sur la page Wikipédia qui leur est consacrée.

1.1 Systèmes de commande

Quelles tâches un système de commande peut-il prendre en charge ? Il peut être chargé de contrôler une séquence d'événements, de maintenir constante une certaine variable ou de suivre un changement prévu. Par exemple, le système de commande d'une perceuse automatique (voir Figure 1.1a) peut démarrer la descente du foret lorsque la pièce est en position, démarrer le perçage lorsque le foret arrive sur la pièce, stopper le perçage lorsque la profondeur du trou souhaitée est atteinte, retirer le foret, arrêter la perceuse, puis attendre que la pièce suivante arrive en position avant de répéter le processus. Un autre système de commande (voir Figure 1.1b) pourrait être utilisé pour vérifier le nombre d'articles convoyés par une bande transporteuse et les diriger vers une caisse. Les entrées de ces systèmes de commande peuvent provenir d'interrupteurs, qui sont ouverts ou fermés. Par exemple, la présence de la pièce peut être signalée par son contact avec un interrupteur, qui se ferme, ou par d'autres capteurs, par exemple de température ou de débit. Le contrôleur peut être chargé de commander un moteur pour déplacer un objet vers un certain emplacement ou pour tourner une vanne, ou encore d'allumer ou d'éteindre un radiateur.

Quelles formes un système de commande peut-il prendre ? Pour la perceuse automatique, il est possible de câbler des circuits électriques dans lesquels l'ouverture ou la fermeture de l'interrupteur conduisent au démarrage de moteurs ou au déclenchement de vannes. En conséquence, les contacts d'un relais peuvent ainsi être fermés ou ouverts (voir Figure 1.2), ce qui met un moteur sous tension et déclenche la rotation du foret (voir Figure 1.3). Un autre interrupteur peut être utilisé pour

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

activer un relais et mettre sous tension une vanne pneumatique ou hydraulique, qui met sous pression un vérin afin de pousser la pièce dans la position requise. De tels circuits électriques sont spécifiques à la perceuse automatique. Pour contrôler le nombre d'articles placés dans un carton, nous pouvons également concevoir des circuits électriques qui mettent en œuvre des capteurs et des moteurs. Cependant, le circuit de chacun de ces systèmes de commande sera différent. Dans un système de commande « traditionnel », les règles de fonctionnement et les actions exécutées sont déterminées par le câblage. Lorsque les règles qui déterminent les actions de commande sont modifiées, le câblage doit également être revu.

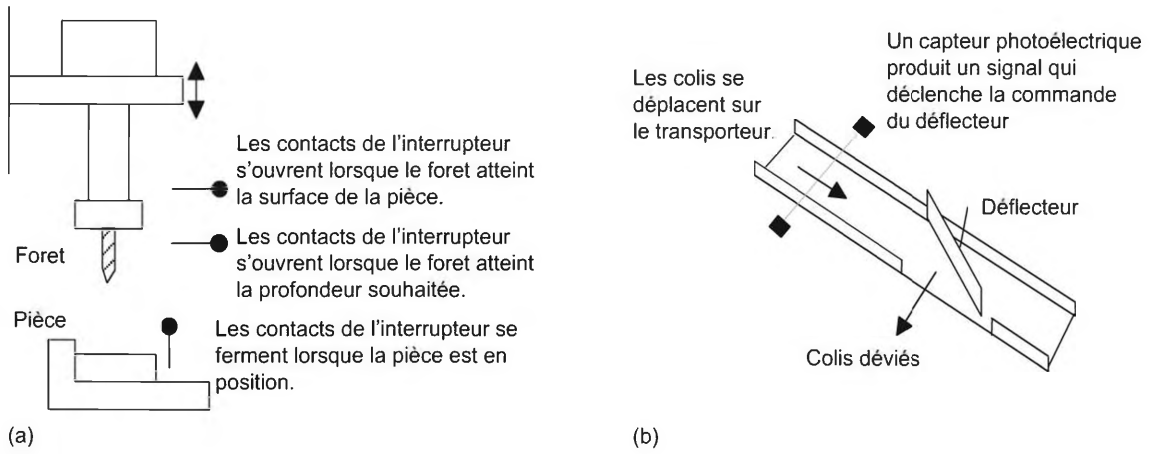


Figure 1.1 – Exemples de tâches de commande et de capteurs d'entrée : (a) une perceuse automatique et (b) un système de conditionnement.

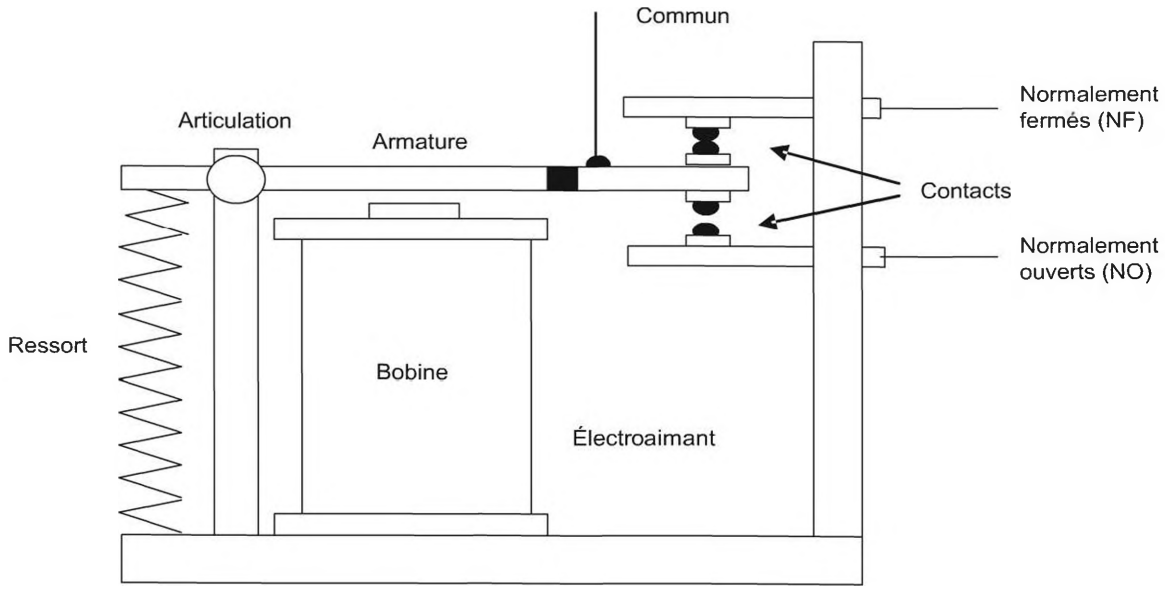


Figure 1.2 – Principe de base d'un relais.

Copyright © 2015 Dunod.

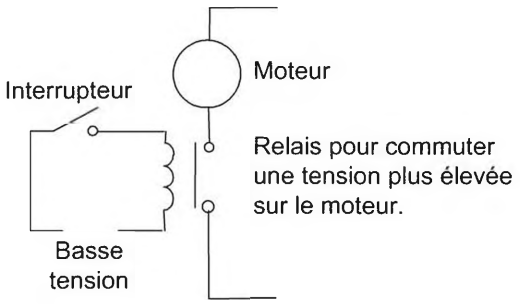


Figure 1.3 – Un circuit de commande.

1.1.1 Systèmes de commande à relais

Les systèmes de commande à relais sont des systèmes câblés. La Figure 1.2 illustre les constituants d'un relais simple. Lorsqu'un courant traverse la bobine du relais, les contacts normalement fermés (NF) s'ouvrent et les contacts normalement ouverts (NO) se ferment. Ces contacts peuvent servir à commander un système. Prenons par exemple un relais utilisé pour déclencher une vanne pneumatique ou hydraulique, ce qui conduit à l'application d'une pression à un piston pour déplacer une pièce. Il est possible de représenter ce fonctionnement par un schéma de commande. La Figure 1.4 donne les symboles standard pour la représentation des relais. Dans le schéma de commande illustré à la Figure 1.5, les lignes verticales représentent les sources d'alimentation tandis que les lignes horizontales relient les systèmes entre eux. La séquence des événements se lit en partant de la ligne horizontale supérieure et en allant vers le bas. Ainsi, dans la ligne supérieure de la Figure 1.5a, lorsque l'interrupteur Marche-arrêt est fermé, le relais est activé. Cela déclenche la fermeture des contacts représentés sur la seconde ligne et, par conséquent, l'alimentation de l'électrovanne. Un schéma plus classique est donné à la Figure 1.5b. Dans ce cas, le relais est activé par un bouton-poussoir normalement ouvert à appui momentané. Son action ferme deux jeux de contacts. Les contacts 1 mémorisent l'appui sur le bouton-poussoir afin que son relâchement ne coupe pas l'alimentation du relais. Les contacts 2 alimentent l'électrovanne. Le relais et, par conséquent, l'alimentation de l'électrovanne, sont coupés lors de l'appui sur le bouton-poussoir normalement fermé. Ces schémas ne représentent évidemment qu'une partie du système de commande car il faudrait d'autres lignes pour détecter le moment où l'électrovanne a déplacé la pièce de la distance requise et pour stopper son fonctionnement.

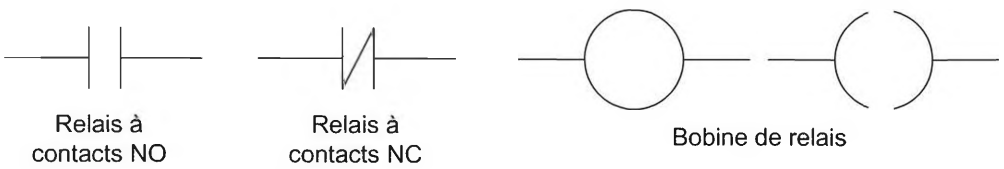


Figure 1.4 – Représentations symboliques d'un relais.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

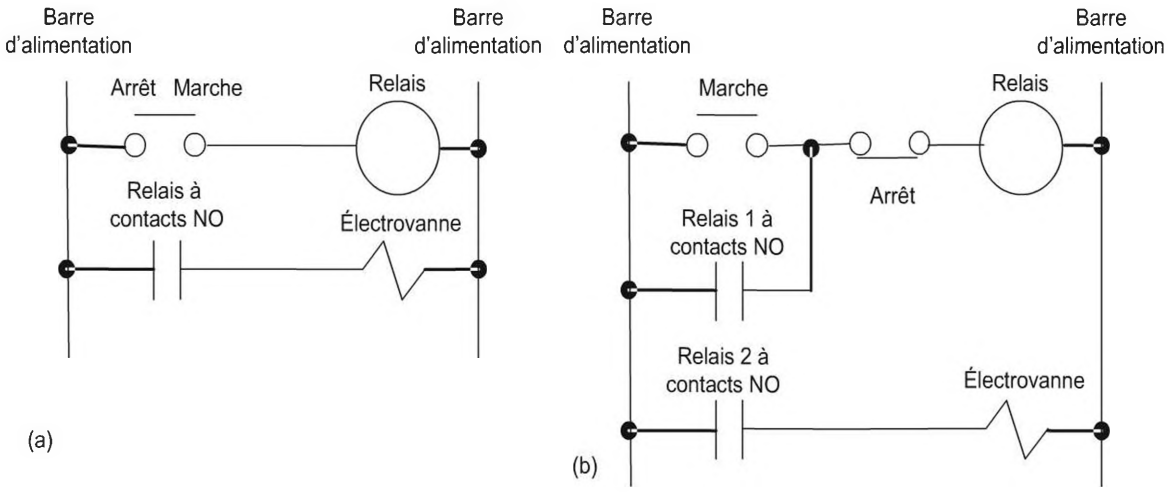


Figure 1.5 – Schémas de systèmes de commande à relais.

La Figure 1.6 montre un autre exemple de système de commande à relais. Lorsque le bouton-poussoir Marche est fermé, la bobine du relais est alimentée et mémorise l'actionnement du bouton afin que le relais reste activé jusqu'à l'appui sur le bouton Arrêt. Les contacts NO du relais se ferment et ses contacts NF s'ouvrent. La lampe verte s'allume, contrairement à la lampe rouge qui s'éteint. Suite à l'appui sur le bouton-poussoir Arrêt, le courant dans la bobine du relais est coupé. Les contacts NO s'ouvrent alors, tandis que les contacts NF se ferment ; la lampe verte s'éteint et la lampe rouge s'allume. L'étape suivante du schéma pourrait impliquer la mise en marche d'un moteur par les contacts NO. Dans ce cas, la lampe verte indiquerait que le moteur tourne et la lampe rouge qu'il est arrêté.

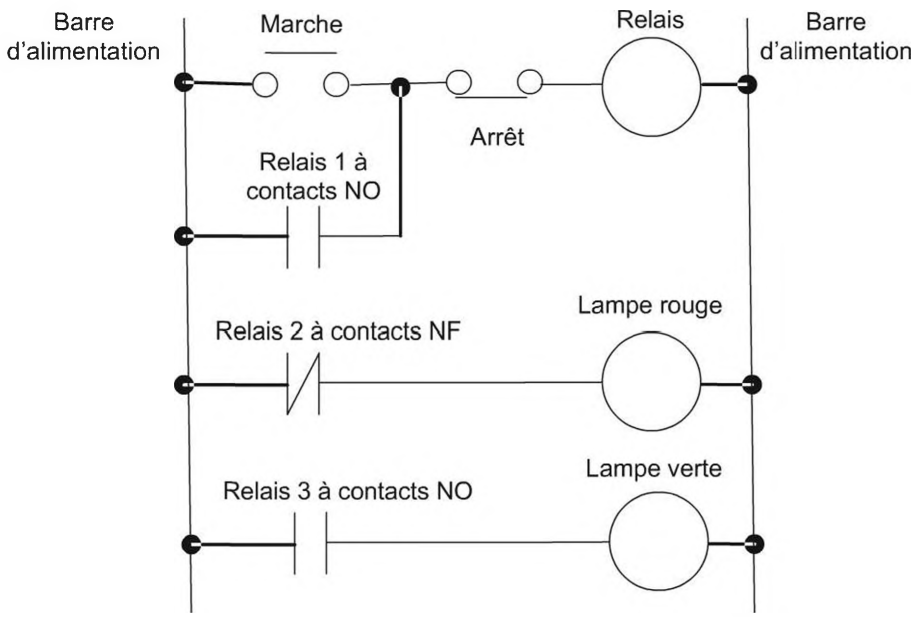


Figure 1.6 – Circuit à relais pour la commande de lampes rouge et verte.

Copyright © 2015 Dunod.

1.1.2 Systèmes de commande à microprocesseur

Au lieu de câbler un circuit de commande différent pour chaque situation, une approche permet d'employer le même système de base dans tous les cas. Elle se fonde sur un système à base de microprocesseur et sur un programme qui explique à ce microprocesseur comment réagir aux signaux d'entrée, par exemple issus d'interrupteurs, et comment produire les sorties requises, par exemple sur des moteurs ou des vannes. Ainsi, nous pourrions avoir un programme de la forme suivante :

```
Si l'interrupteur A se ferme
Activer la sortie du circuit du moteur
Si l'interrupteur B se ferme
Activer la sortie du circuit de la vanne
```

En modifiant les instructions du programme, nous pouvons conserver le même système à microprocesseur pour commander une grande diversité de cas.

Prenons l'exemple d'une machine à laver le linge moderne à base de microprocesseur. Ses entrées proviennent des cadrans de sélection du programme de lavage, d'un interrupteur qui indique la fermeture de la porte, d'un capteur qui détermine la température de l'eau et d'un interrupteur qui détecte le niveau d'eau. Le programme du microprocesseur utilise ces entrées pour positionner les sorties de manière à démarrer le moteur du tambour et contrôler sa vitesse, à ouvrir et à fermer les vannes d'eau chaude et d'eau froide, à mettre en marche la pompe de vidange, à commander la résistance de chauffage et à contrôler le verrou de la porte pour que la machine ne puisse pas être ouverte avant la fin du cycle.

1.1.3 Automates programmables industriels

Un *automate programmable industriel* (API) est une forme particulière de contrôleur à microprocesseur qui utilise une mémoire programmable pour stocker les instructions et qui implémente différentes fonctions, qu'elles soient logiques, de séquencement, de temporisation, de comptage ou arithmétiques, pour commander les machines et les processus (voir Figure 1.7). Il est conçu pour être exploité par des ingénieurs, dont les connaissances en informatique et langages de programmation peuvent être limitées. La création et la modification des programmes de l'API ne sont pas réservées aux seuls informaticiens. Les concepteurs de l'API l'ont préprogrammé pour que la saisie du programme de commande puisse se faire à l'aide d'un langage simple et intuitif (voir Chapitre 4). La programmation de l'API concerne principalement la mise en œuvre d'opérations logiques et de commutation, par exemple, si A *ou* B se produit, alors allumer C, ou si A *et* B se produisent, alors allumer D. Les dispositifs d'entrée, c'est-à-dire des capteurs, comme des interrupteurs, et les dispositifs de sortie, c'est-à-dire des moteurs, des vannes, etc., du système sont connectés à l'API. L'opérateur saisit une séquence d'instructions, le programme, dans la mémoire de l'API. L'automate surveille ensuite les entrées et les sorties conformément aux instructions du programme et met en œuvre les règles de commande définies.

Les API présentent un avantage majeur : le même automate de base peut être employé avec une grande diversité de systèmes de commande. Pour modifier un

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

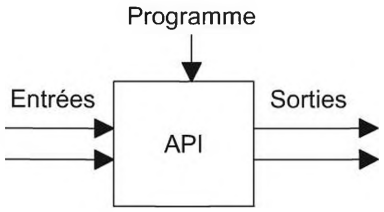


Figure 1.7 – Un automate programmable industriel.

système de commande et les règles appliquées, un opérateur doit simplement saisir une suite d'instructions différente. On obtient ainsi un système flexible et économique, utilisable avec des systèmes de commande dont la nature et la complexité peuvent varier énormément. En comparaison des systèmes à relais, les API :

- simplifient les modifications car elles sont mises en œuvre de façon logicielle et non pas par des solutions matérielles plus complexes ;
- peuvent être facilement étendus par l'ajout de nouveaux modules, alors que des changements matériels sont sinon requis ;
- sont plus robustes et plus fiables en raison d'un nombre de composants mécaniques moindre ;
- sont plus compacts ;
- exigent une maintenance moindre ;
- sont plus rapides.

Les API sont comparables aux ordinateurs. Toutefois, alors que les ordinateurs sont optimisés pour les tâches de calcul et d'affichage, les API le sont pour les tâches de commande et les environnements industriels. Voici ce qui différencie les API par rapport aux ordinateurs :

- Ils sont solides et conçus pour supporter les vibrations, les températures basses ou élevées, l'humidité et le bruit. *A contrario*, les ordinateurs personnels ne sont pas faits pour opérer dans des milieux hostiles.
- Les interfaces des entrées et des sorties sont intégrées à l'automate. Les API au format modulaire peuvent être facilement étendus pour recevoir un plus grand nombre d'entrées-sorties.
- Ils sont simples à programmer et leur langage de programmation d'apprentissage facile est principalement orienté sur les opérations logiques et de commutation. Ils sont par conséquent plus conviviaux.
- Ils sont moins adaptés au stockage à long terme et à l'analyse des données.
- Ils sont d'une fiabilité supérieure et moins sujets aux dysfonctionnements que les ordinateurs personnels.

Les premiers API ont été conçus en 1969 ; ils sont à présent largement utilisés. Ils prennent la forme de petites unités autonomes pour environ vingt entrées-sorties numériques ou de systèmes modulaires qui peuvent être employés pour des entrées-sorties très nombreuses, analogiques ou numériques, et qui disposent des

Copyright © 2015 Dunod.

modes de régulation PID (proportionnel, intégral et dérivé). On les retrouve dans l'automatisation des processus industriels, comme l'usinage, la manutention, l'assemblage automatique et le conditionnement. Cependant, dans le cas des tâches d'automatisation très simples, par exemple pour un lave-linge domestique, une autre solution moins coûteuse sera probablement retenue. Lorsque les contraintes sont fortes, par exemple dans le cas du contrôle du vol d'un avion, un ordinateur aura la préférence car il est capable d'effectuer des opérations mathématiques complexes avec une vitesse de traitement élevée.

1.2 Matériel

De manière générale, un API est structuré autour de plusieurs éléments de base que sont l'unité de traitement, la mémoire, l'unité d'alimentation, les interfaces d'entrées-sorties, l'interface de communication et le périphérique de programmation (voir Figure 1.8) :

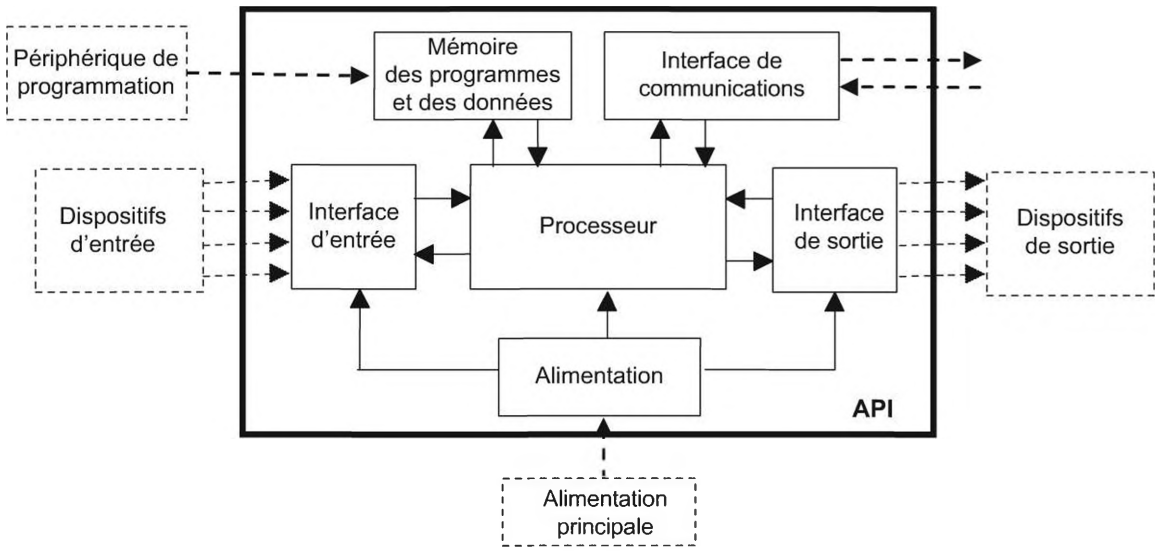


Figure 1.8 – Structure d'un API.

- Le *processeur* ou *unité centrale de traitement* (CPU, *Central Processing Unit*) contient le microprocesseur. Le CPU interprète les signaux d'entrée et effectue les actions de commande conformément au programme stocké en mémoire, en communiquant aux sorties les décisions sous forme de signaux d'action.
- L'*unité d'alimentation* est indispensable puisqu'elle convertit une tension alternative en une basse tension continue (5 V) nécessaire au processeur et aux modules d'entrées-sorties.
- Le *périphérique de programmation* est utilisé pour entrer le programme dans la mémoire du processeur. Ce programme est développé sur le périphérique, puis transféré dans la mémoire de l'API.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

- La *mémoire* contient le programme qui définit les actions de commande effectuées par le microprocesseur. Elle contient également les données qui proviennent des entrées en vue de leur traitement, ainsi que celles des sorties.
- Les *interfaces d'entrées-sorties* permettent au processeur de recevoir et d'envoyer des informations aux dispositifs extérieurs. Les entrées peuvent être des interrupteurs, comme dans le cas de la perceuse automatique (voir Figure 1.1a), ou d'autres capteurs, comme des cellules photoélectriques dans le cas du mécanisme de comptage (voir Figure 1.1b), des sondes de température, des débitmètres, etc. Les sorties peuvent être des bobines de moteur, des électrovannes, etc. Nous reviendrons sur les interfaces d'entrées-sorties au Chapitre 2. Les dispositifs d'entrées-sorties peuvent être classés en trois catégories, selon qu'ils produisent des signaux discrets, numériques ou analogiques (voir Figure 1.9). Les dispositifs qui génèrent des *signaux discrets* ou *numériques* sont ceux dont les sorties sont de type tout ou rien. Par conséquent, un interrupteur est un dispositif qui produit un signal discret : présence ou absence de tension. Les dispositifs *numériques* peuvent être vus comme des dispositifs discrets qui produisent une suite de signaux tout ou rien. Les dispositifs *analogiques* créent des signaux dont l'amplitude est proportionnelle à la grandeur de la variable surveillée. Par exemple, un capteur de température peut produire une tension proportionnelle à la température.

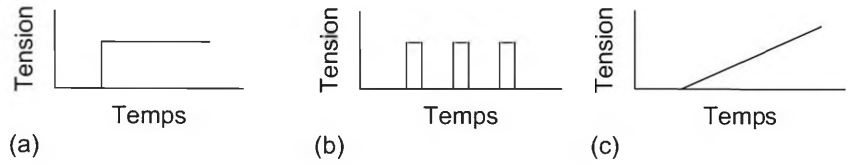


Figure 1.9 – Les signaux : (a) discrets, (b) numériques et (c) analogiques.

- L'*interface de communication* est utilisée pour recevoir et transmettre des données sur des réseaux de communication qui relient l'API à d'autres API distants (voir Figure 1.10). Elle est impliquée dans des opérations telles que la vérification d'un périphérique, l'acquisition de données, la synchronisation entres des applications et la gestion de la connexion.

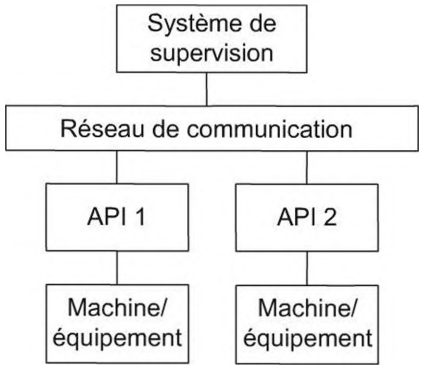


Figure 1.10 – Modèle de base des communications.

Copyright © 2015 Dunod.

1.3 Architecture d'un API

Un API est généralement constitué d'une unité centrale de traitement (CPU, *Central Processing Unit*) qui comprend le microprocesseur, la mémoire et les entrées-sorties du système. Il peut en réalité être vu comme une entité composée d'un grand nombre de relais, compteurs, temporisateurs et unités de stockage de données, même si ces éléments n'existent pas physiquement dans l'API et sont simulés de façon logicielle.

La capacité de stockage d'une unité de mémoire est déterminée par le nombre de mots qu'elle peut enregistrer. Si la taille d'une mémoire est de 256 mots, elle peut stocker $256 \times 8 = 2\,048$ bits, pour des mots de huit bits, et $256 \times 16 = 4\,096$ bits, pour des mots de seize bits. Le terme *octet* (*byte*) est utilisé pour désigner un mot de huit bits. La taille de la mémoire est souvent indiquée en fonction du nombre d'emplacements disponibles. 1 K représente le nombre 2^{10}, c'est-à-dire 1 024. Une mémoire de 4 K octets peut donc enregistrer 4 096 octets, et une mémoire de 50 K octets en stockera 51 200.

1.3.1 Unité d'entrées-sorties

L'unité d'entrées-sorties (E/S) d'un API apporte le circuit d'interface entre le système et le monde extérieur. Au travers de canaux d'entrées-sorties, elle permet d'établir des connexions avec des dispositifs d'entrée, comme des capteurs, et des dispositifs de sortie, comme des moteurs et des solénoïdes. C'est également par l'intermédiaire de cette unité que se fait la saisie des programmes depuis un terminal. Chaque point d'entrée-sortie dispose d'une adresse unique, que le CPU peut utiliser. Le principe est comparable à une rangée de maisons le long d'une rue ; le numéro 10 peut correspondre à la « maison » pour l'entrée d'un capteur particulier, tandis que le numéro 45 peut correspondre à la « maison » utilisé pour la sortie vers un moteur spécifique.

Puisque les canaux d'entrées-sorties mettent en place les fonctions d'isolation et de traitement des signaux, il est possible de connecter directement des capteurs et des actionneurs aux canaux, sans passer par un autre circuit d'interface (voir Figure 1.11). L'isolation électrique avec le monde extérieur est généralement réalisée par des *photocoupleurs* (également appelés *optocoupleurs*), dont le principe est illustré à la Figure 1.12. Lorsque la diode électroluminescente (LED, *Light Emitting Diode*) est traversée par une impulsion numérique, elle produit un rayonnement infrarouge. Ce rayonnement est détecté par le phototransistor, qui fait naître une tension dans son circuit. L'espace qui sépare la LED et le phototransistor crée une isolation électrique, mais une impulsion numérique dans le premier circuit permet néanmoins de produire une impulsion numérique dans le second circuit.

Le traitement du signal réalisé au niveau du canal d'entrée, avec l'isolation, permet de manipuler une grande diversité de signaux d'entrée. Il est converti en une tension compatible avec celle requise par le microprocesseur qui équipe l'API (voir Chapitre 3). Un API élaboré peut ainsi accéder à des entrées dont les signaux

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

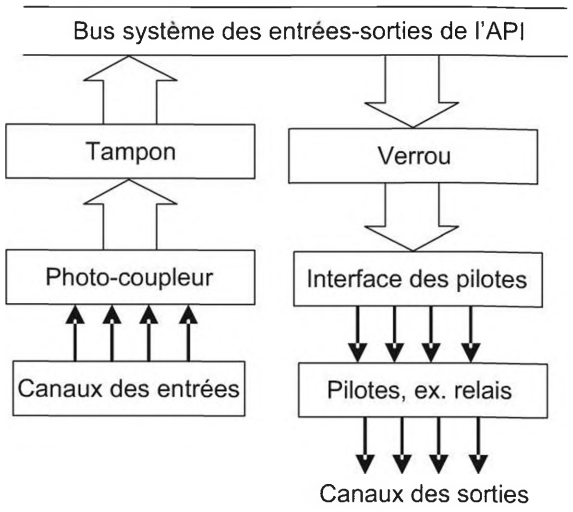


Figure 1.11 – Architecture des canaux d'entrées-sorties d'un API.

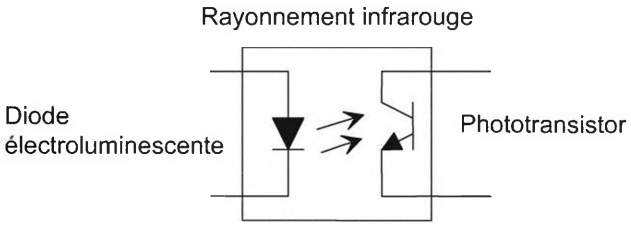


Figure 1.12 – Un photocoupleur.

numériques/discrets (c'est-à-dire tout ou rien) utilisent des tensions de 5 V, 24 V, 110 V et 240 V. Un API de base sera généralement en mesure d'utiliser une seule forme d'entrée, par exemple des signaux 24 V.

Les canaux de sortie permettent d'obtenir les sorties de l'API sous une forme adaptée à un connexion directe avec des circuits externes. Les sorties sont de type relais, transistor ou triac (voir Chapitre 3 pour plus de détails) :

- Avec le type *relais*, le signal issu d'une sortie de l'API est utilisé pour déclencher un relais et peut activer des courants de quelques ampères dans le circuit externe. Grâce au relais, non seulement des courants faibles peuvent commuter des courants forts, mais l'isolation entre l'API et le circuit externe est également assurée. En revanche, les relais sont relativement lents. Les sorties à relais sont adaptées à la commutation de courants alternatifs et continus. Ils peuvent supporter des surtensions transitoires et des courants de choc élevés.
- Avec le type *transistor*, la sortie se fonde sur un transistor pour commuter le courant dans le circuit externe. L'opération de commutation est donc extrêmement rapide. Toutefois, les sorties de ce type sont adaptées uniquement à la commutation d'un courant continu et sont détruites par les surintensités et les tensions inverses élevées. Pour leur protection, il faut utiliser un fusible ou un système électronique. Les photocoupleurs sont employés à des fins d'isolation.

Copyright © 2015 Dunod.

– Avec le type *triac*, et des photocoupleurs pour l'isolation, les sorties peuvent servir à contrôler les charges externes connectées à une alimentation en courant alternatif. Elles prennent en charge uniquement les courants alternatifs et sont très facilement détruites par les surintensités. De manière générale, les sorties de ce type sont toujours protégées par des fusibles.

Ainsi, après le traitement du signal par des relais, des transistors ou des triacs, le canal de sortie est capable de fournir un signal 24 V et 100 mA, une tension continue de 110 V et 1 A, ou 240 V, une tension alternative de 240 V et 1 A ou 2 A. Dans le cas d'un petit API, toutes les sorties seront d'un même type, par exemple 240 V alternatif et 1 A. En revanche, avec des API modulaires, il est possible de proposer un éventail de sorties en panachant les modules connectés.

1.3.2 Fourniture et absorption de courant

Les termes *fourniture* (*sourcing*) et *absorption* (*sinking*) décrivent la manière dont les appareils à courant continu sont connectés à un API. Dans le cas de la fourniture, en supposant le sens conventionnel du courant du plus vers le moins, un dispositif d'entrée reçoit le courant à partir du module d'entrée. Autrement dit, le module d'entrée est une source de courant et le fournit au dispositif d'entrée (voir Figure 1.13a). Dans le cas de l'absorption, un dispositif d'entrée fournit le courant au module d'entrée. Autrement dit, le module d'entrée reçoit et absorbe le courant (voir Figure 1.13b). Si le courant va du module de sortie vers une charge de sortie, le module de sortie fournit le courant (voir Figure 1.14a). En revanche, si le courant va de la charge de sortie vers le module de sortie, celui-ci absorbe le courant (voir Figure 1.14b).

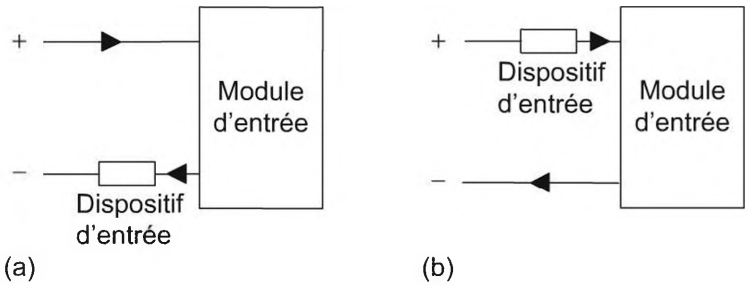


Figure 1.13 – Les entrées : (a) à fourniture et (b) à absorption de courant.

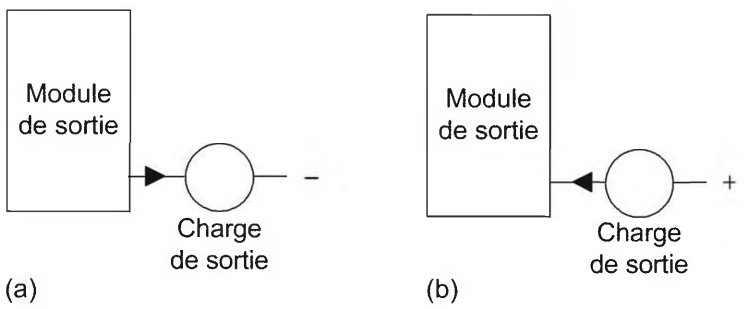


Figure 1.14 – Les sorties : (a) à fourniture et (b) à absorption de courant.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

Il est important de connaître le type de la sortie ou de l'entrée afin qu'elle soit branchée correctement à l'API. Ainsi, les capteurs avec sorties à fourniture de courant doivent être connectés aux entrées à absorption de courant de l'API, tandis que les capteurs avec sorties à absorption de courant doivent être connectés aux entrées à fourniture de courant. Si ces directives ne sont pas suivies, l'interface avec l'API ne fonctionnera pas et risque d'être endommagée.

1.4 Systèmes API

Les systèmes API sont principalement disponibles sous deux formes : en *boîtier unique* et en *version modulaire/rack*. Le modèle en boîtier unique, ou *coffret*, est souvent utilisé pour les petits automates programmables et correspond à un système compact complet équipé des unités d'alimentation, de processeur, de mémoire et d'entrées-sorties. Ces API peuvent disposer de six, huit, douze ou vingt-quatre entrées, de quatre, huit ou seize sorties et d'une mémoire permettant d'enregistrer entre 300 et 1 000 instructions. Par exemple, l'API compact TAR 116-6S de Toshiba dispose de huit entrées 120 V à courant alternatif, six sorties à relais et deux sorties à triax, alors que le modèle TDR140-6S plus important possède 24 entrées 24 V à courant continu, 14 sorties à relais et deux sorties à triac. Certains systèmes compacts peuvent être étendus afin d'offrir un plus grand nombre d'entrées et de sorties, simplement en connectant des modules d'extension.

La Figure 1.15 présente l'API compact CP1L d'Omron pour la commande de machines. Pour ce modèle précis, quatre tailles de processeur sont disponibles, chacune avec la possibilité de choisir entre des sorties à relais ou à transistor. La tension, la sortie et le nombre de points d'E/S peuvent être sélectionnés afin de répondre aux besoins. Le boîtier de base, selon le modèle choisi, dispose de 10, 14, 20 ou 30 entrées-sorties (E/S). La version à 10 E/S offre six entrées numériques et quatre sorties, la version à 14 E/S dispose de huit entrées numériques et six sorties, la version à 20 E/S propose douze entrées numériques et huit sorties, tandis que la version à 30 E/S offre dix-huit entrées numériques et douze sorties. Il est possible d'opter pour des sorties à relais, à transistor absorbeur de courant ou à transistor fournisseur de courant. Les modèles à 14, 20 et 30 E/S peuvent être étendus afin d'offrir un plus grand nombre d'entrées-sorties. Par exemple, l'extension du modèle 14 E/S permet d'obtenir 54 entrées-sorties.

La Figure 1.16 présente l'API compact FX3U de Mitsubishi. Les tableaux 1.1 et 1.2 détaillent les modèles de cette gamme de Mitsubishi.

Les systèmes qui offrent un plus grand nombre d'entrées et de sorties ont toutes les chances d'être modulaires et conçus pour s'intégrer dans des racks (voir Figure 1.17). Un système modulaire est constitué de modules séparés pour l'alimentation, le processeur, etc., souvent montés sur des rails dans une armoire métallique. Ce type de système peut être employé pour toutes les tailles d'automates programmables et les différentes unités fonctionnelles sont fournies sous forme de modules individuels qui se branchent à des prises sur un rack de base. Le choix des modules nécessaires à la réalisation d'un projet précis est décidé par

Copyright © 2015 Dunod.

Tableau 1.1 La gamme de produits MELSEC FX

Type	FX3S	FX3GE	FX3G	FX3U	FX3UC
Alimentation	100–240 V CA 24 V CC	100–240 V CA	100–240 V CA 24 V CC	100–240 V CA 24 V CC	24 V CC
Entrées	6–16	14–16	8–36	8–64	8–48
Sorties	4–14	10–14	8–24	8–64	8–48
Sorties numériques	Relais, transistor				Transistor
Durée d'un cycle par instruction logique	0,21 µs	0,21–0,42 µs	0,21–0,42 µs	0,065 µs	0,065 µs
Mémoire utilisateur	EEPROM 4 k pas (interne), cassettes EEPROM/EPROM (en option)	EEPROM 32 k pas (interne), cassettes EEPROM/EPROM (en option)	EEPROM 32 k pas (interne), cassettes EEPROM (en option)	64 k pas (standard), cassettes FLROM (en option)	64 k pas (standard), cassettes EEPROM (en option)
Dimensions en mm (L × H × P)	60–100 × 90 × 75	130–175 × 90 × 86	90–175 × 90 × 86	130–350 × 90 × 86	34–86 × 90 × 74

Reproduit avec l'autorisation de Mitsubishi Electric Europe.

Figure 1.15 – Le modèle CP1L d'Omron (reproduit avec l'autorisation d'Omron Industrial Automation).

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

Tableau 1.2 Unités principales du FX3U avec 16 entrées-sorties

Numéro de modèle	FX3U-16 MR/DS	FX3U-16 MR/ES	FX3U-16 MT/DSS	FX3U-16 MT/DS	FX3U-16 MT/ESS	FX3U-16 MT/ES
Élément mémorisé	S	S	S	S	S	S
Conformité	UL, cUL, CE (EMC)*					
Entrées-sorties intégrées	16	16	16	16	16	16
Alimentation	24 V CC	100–240 V CA	24 V CC	24 V CC	100–240 V CA	100–240 V CA
Entrées intégrées	8	8	8	8	8	8
Sorties intégrées	8	8	8	8	8	8
Type de sortie	Relais	Relais	Transistor (fourniture)	Transistor (absorption)	Transistor (fourniture)	Transistor (absorption)
Consommation (W)	25	30	25	25	30	30
Poids (kg)	0,60	0,60	0,60	0,60	0,60	0,60
Dimensions en mm (L × H × P)	130 × 90 × 86	130 × 90 × 86	130 × 90 × 86	130 × 90 × 86	130 × 90 × 86	130 × 90 × 86

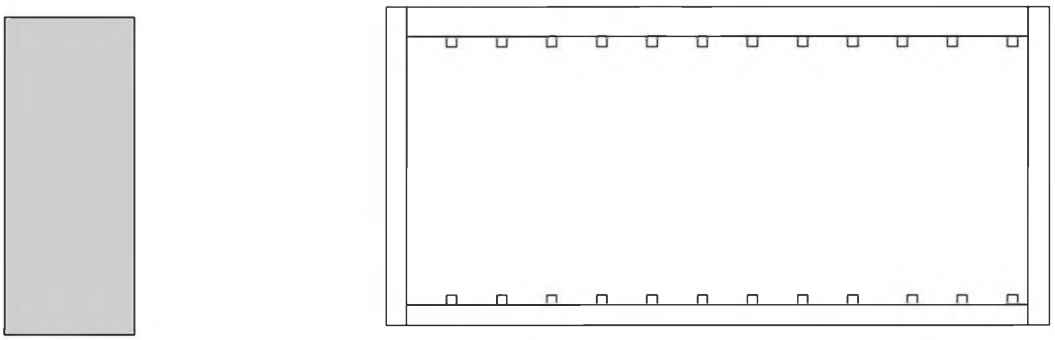
* : noms des organismes qui ont validé la conformité avec les certifications.

Reproduit avec l'autorisation de Mitsubishi Electric Europe.

Figure 1.16 – L'API compact FX3U de Mitsubishi (reproduit avec l'autorisation de Mitsubishi Electric Automation).

Copyright © 2015 Dunod.

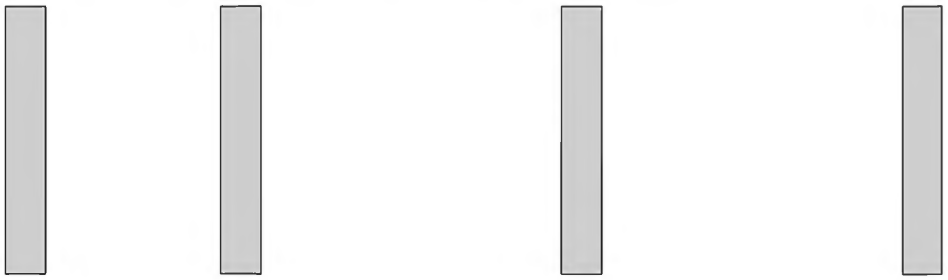
l'utilisateur et ces modules sont ensuite branchés dans le rack. Il est ainsi plus facile d'étendre le nombre d'entrées-sorties ou la quantité de mémoire. Pour cela, il suffit simplement d'ajouter des modules d'entrées-sorties ou des unités de mémoire supplémentaires. L'alimentation et les interfaces de données des modules dans le rack sont fournies par les conducteurs en cuivre sur le fond de panier. Lorsque les modules sont glissés dans le rack, ils s'insèrent dans des connecteurs du fond de panier.



Alimentation du système

La forme de base est un rack dans lequel les composants d'un API peuvent être enfichés, le fond de panier fournissant les connecteurs d'accès à l'alimentation et aux bus.

Quelques éléments à enficher dans le rack



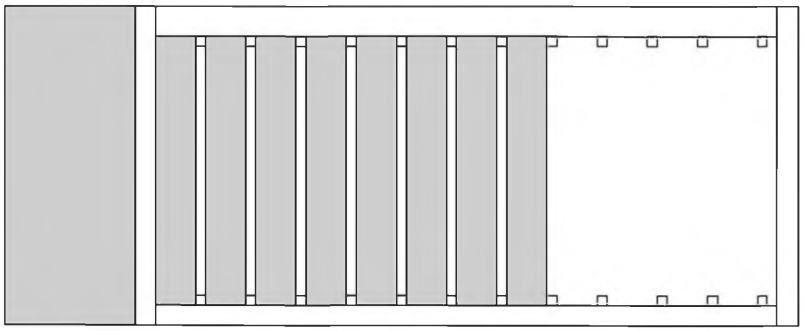
Module processeur

Module de communication avec les ordinateurs, les adaptateurs d'E/S et d'autres automates

Module adaptateur d'E/S pour connecter le fond de panier à un autre processeur distant

Modules d'E/S pour convertir les signaux d'entrée en niveaux du fond de panier et les signaux du fond de panier en niveaux de circuits de sortie

Exemple de système assemblé



Alimentation

Figure 1.17 – Une organisation possible d'un système modulaire, comme le modèle Allen-Bradley PLC-5 de Rockwell Automation.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

Le modèle Allen-Bradley PLC-5 de Rockwell Automation est un exemple de système modulaire. Il comprend au moins une alimentation, un module de commande programmable et des modules d'entrées-sorties. Plusieurs châssis sont disponibles pour la connexion des modules :

- *Châssis.* Certains châssis d'entrées-sorties 1771 sont conçus pour un montage en fond de panier, tandis que d'autres le sont pour un montage en rack et dans des tailles acceptant quatre, huit, douze ou seize modules d'entrées-sorties.
- *Modules de commande.* Les processeurs PLC-5 sont disponibles dans plusieurs variantes en termes de capacité d'entrées-sorties et de mémoire. Par exemple, le PLC-5/11 propose au maximum 512 entrées-sorties et une mémoire allant jusqu'à 8 000 mots (voir le Chapitre 3 pour une explication du terme « mot »), tandis que le PLC-5/20 dispose d'au plus 512 E/S et d'une mémoire de 16 000 mots. Ils peuvent être configurés pour différents réseaux de communication, comme le PLC-5/20C qui est compatible avec ControlNet et le PLC-5/20E qui utilise Ethernet. Il s'agit de modules mono-slot qui se placent dans l'emplacement de gauche d'un châssis d'entrées-sorties 1771.
- *Modules d'entrées-sorties.* Les modules d'entrées-sorties 1771 existent en versions huit, seize ou trente-deux entrées-sorties par module, pour un interfaçage avec des capteurs à courant alternatif ou continu et des actionneurs. Les modules d'entrées-sorties numériques s'interfacent avec des capteurs tout ou rien, comme des boutons-poussoirs et des interrupteurs de fin de course, et avec des actionneurs tout ou rien, comme des moteurs, des lampes témoin et des avertisseurs. Les modules d'entrées-sorties analogiques réalisent les conversions A/N et N/A, avec une résolution allant jusqu'à seize bits. Les entrées-sorties analogiques peuvent être configurées par l'utilisateur dans l'état défaut-réponse souhaité en cas de dysfonctionnement des communications. Cette caractéristique permet d'obtenir une réaction/réponse fiable en cas de panne, limite l'étendue des défauts et apporte une réponse prévisible aux défauts. Les modules d'entrées-sorties 1771 sont équipés d'un couplage optique et d'un circuit de filtrage pour la réduction du bruit. Les modules d'entrées-sorties numériques sont compatibles avec des courants alternatifs ou continus de 5 à 276 V. Par ailleurs, les modules de sorties à relais sont disponibles dans les plages 0 à 276 V CA et 0 à 175 V CC. Toute une gamme de niveaux de signaux analogiques peut être pris en charge, y compris les entrées et les sorties analogiques standard, de même que les entrées directes pour la mesure de température par thermocouple ou résistance. Le module d'entrées numériques 1771-1B propose ainsi huit entrées avec des tensions comprises dans la plage 10 à 27 V, le 1771-0VN avec 32 entrées numériques et des niveaux de tension situés entre 10 et 30 V, le module d'entrées analogiques 1771-NIV pour huit entrées en ±5 V et ±20 mA en courant continu, ainsi que le module de sortie analogique 1771-OFE2 pour quatre sorties dans la plage 4 à 20 mA. Un processeur PLC-5 peut communiquer avec les entrées-sorties par une liaison DeviceNet ou Universal Remote I/O.

Copyright © 2015 Dunod.

– *Modules de communication.* Des modules de communication peuvent être utilisés pour augmenter le nombre de ports de communication du processeur PLC-5 et aller au-delà de ceux fournis par un module de commande.

1.4.1 Sécurité

Puisque les API peuvent être connectés à des réseaux et qu'ils sont équipés de systèmes d'exploitation temps réel, la question de la sécurité se pose naturellement. En effet, ces réseaux peuvent être piratés et des informations peuvent alors tomber dans des mains non autorisées ou des virus peuvent être implantés. Il est également possible que des API soient attaqués lorsqu'un ordinateur avec lequel ils communiquent a lui-même été piraté.

1.5 Programmes

Les programmes utilisés avec les API peuvent être écrits dans différents formats. Pour que les ingénieurs ayant peu de connaissances en programmation puissent élaborer des programmes pour les API, le *langage à contacts* a été conçu. La plupart des fabricants d'automates ont adopté cette méthode d'écriture des programmes. Toutefois, puisque chacun a eu tendance à développer ses propres versions, une norme internationale a été établie pour le langage à contacts et, par voie de conséquence, pour toutes les méthodes de programmation employées avec les API. Cette norme, publiée en 1993 par la Commission électrotechnique internationale, est désignée sous la référence CEI 61131-3. La dernière version, qui date de 2013, est une extension qui reste compatible avec la version antérieure.

Les langages de programmation définis par la norme CEI 61131-3 sont le langage à contacts (LAD, *Ladder Diagram*), les listes d'instructions (IL, *Instruction List*), les graphes de fonction séquentielle (SFC, *Sequential Function Charts*), le texte structuré (ST, *Structured Text*) et les diagrammes de schémas fonctionnels (FBD, *Function Block Diagram*). La norme comprend une bibliothèque de fonctions préprogrammées et des blocs fonctionnels. On nomme fonction un calcul préprogrammé, par exemple pour donner la valeur moyenne de deux entrées. Un bloc fonctionnel évalue ses entrées et produit une valeur en sortie, par exemple un bloc de comptage qui compte les impulsions sur ses entrées et génère en sortie un signal lorsque le compteur a atteint une valeur précisée. Il s'agit de parties d'un programme de commande qui sont assemblées afin d'être utilisées en différents endroits du même programme ou dans différents programmes. La norme CEI donne une définition formelle de chaque paramètre d'entrée et de sortie afin que des blocs fonctionnels conçus par différents programmeurs puissent être facilement interconnectés. N'importe quel API conforme à la norme CEI prend en charge ces fonctions sous forme d'une bibliothèque, le code étant écrit dans la partie PROM d'une mémoire flash de l'appareil.

Parmi ces langages, deux sont textuels, le texte structuré et les listes d'instruction, et seront saisis sur l'appareil de programmation à partir d'un clavier, une ligne à la fois. Les autres, le langage à contacts, les graphes de fonction séquentielle et les

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

diagrammes de schémas fonctionnels, sont graphiques et un programme peut donc être développé à partir d'éléments graphiques sur l'écran de l'appareil de programmation.

1.5.1 La norme CEI 61131

La norme CEI 61131 couvre l'intégralité du cycle de vie des automates programmables industriels :

- Partie 1 : Définition générale de la terminologie et des concepts.
- Partie 2 : Exigences sur le matériel électronique et mécanique et tests de contrôle des API et des équipements associés.
- Partie 3 : Langages de programmation. Cinq langages ont été définis : langage à contacts (LD, *Ladder Diagram*), graphes de fonction séquentielle (SFC, *Sequential Function Charts*), diagrammes de schémas fonctionnels (FBD, *Function Block Diagram*), texte structuré (ST, *Structured Text*) et listes d'instructions (IL, *Instruction List*).
  1. Portée
  2. Références normatives
  3. Termes et définitions
  4. Modèles architecturaux
  5. Conformité
  6. Éléments communs
  7. Langages textuels (listes d'instructions et texte structuré)
  8. Langages graphiques (langage à contacts et diagrammes de schémas fonctionnels)

  Annexe A : Spécification formelle des éléments du langage

  Annexe B : Liste des modifications principales et des extensions de la troisième édition.
- Partie 4 : Conseils de sélection, d'installation et de maintenance des API.
- Partie 5 : Fonctions logicielles pour la communication avec d'autres appareils selon la norme MMS (*Manufacturing Messaging Specification*).
- Partie 6 : Communications via les fonctions logicielles de bus de terrain.
- Partie 7 : Programmation par logique floue.
- Partie 8 : Conseils d'implémentation des langages de programmation définis à la Partie 3.

La norme CEI 61131-6 traite des méthodes de programmation des API. Les origines du langage à contacts (voir Chapitre 5) se trouvent dans les schémas de câblage électrique pour les systèmes de commande à relais, comme l'illustre la Figure 1.4. Il a l'avantage d'être facile à comprendre par les personnes familières des schémas électriques. Il permet également au technicien de maintenance de repérer facilement les erreurs, car la plupart des appareils de programmation disposent d'un affichage animé qui représente l'état des contacts sur les lignes. Le langage à contacts permet de développer des programmes relativement volumineux, mais il s'avère moins pratique lorsqu'il faut employer des sous-routines ou

Copyright © 2015 Dunod.

des blocs de programme. D'autre part, les programmes qui impliquent un grand nombre de séquences risquent d'être difficiles à manipuler lorsque la commande d'une séquence est mélangée à l'application. Si des opérations arithmétiques simples sont réalisables avec le langage à contacts, les calculs plus complexes se révèlent plus difficiles à mettre en œuvre. Malgré ses inconvénients, le langage à contacts est largement utilisé car il est facile à employer et à comprendre. Les graphes de fonction séquentielle (voir Chapitre 6) ont le mérite d'afficher tous les états opérationnels d'un système, tous les changements d'état possibles et les conditions de ces changements. Ils conviennent mieux à l'affichage des séquences que le langage à contacts. Les diagrammes de schémas fonctionnels (voir Chapitre 5) possèdent les avantages d'un outil de programmation et facilitent l'emploi de blocs logiciels réutilisables, les portes logiques en étant un exemple. Le texte structuré (voir Chapitre 6) est un langage de programmation très proche du langage Pascal. Les listes d'instructions (voir Chapitre 6) ont une structure relativement simple et conviennent à l'écriture de petits programmes constitués de quelques points de décision et au nombre réduit de changements dans leur flux d'exécution. Elles ne facilitent pas la lecture du flux du programme.

La norme CEI 61131-5 concerne les communications dans les API (voir Figure 1.11) et s'intéresse donc aux outils d'échange de données et d'informations de contrôle entre les API interconnectés par un réseau. Elle stipule les informations d'état que chaque sous-système doit fournir dans un format standard pour simplifier les communications.

1.5.2 Programmer les API

Une unité de programmation peut être un appareil portatif, un terminal de bureau ou un ordinateur. Après que le programme a été conçu et testé sur l'unité de programmation, il est prêt à être transféré dans la mémoire de l'API.

- Un *appareil de programmation portatif* dispose généralement d'une quantité de mémoire suffisante pour conserver les programmes afin de les déplacer d'un endroit à un autre.
- Les *terminaux de bureau* sont généralement équipés d'un système d'affichage graphique, avec un clavier et un écran.
- Les *ordinateurs personnels* sont souvent employés pour la programmation des API. L'utilisation d'un ordinateur présente plusieurs avantages : le programme peut être stocké sur le disque dur ou sur un CD-ROM et les copies sont faciles à réaliser. L'ordinateur est relié à l'API par l'intermédiaire d'un câble Ethernet, RS-232, RS-485 ou RS-422.

Les fabricants d'API proposent des outils de développement pour leurs API. Par exemple, Mitsubishi fournit MELSOFT. Sa suite logicielle iQ Works est constituée de quatre outils MELSOFT qui permettent une programmation intuitive et la configuration d'un système iQ Platform, notamment la configuration du système et du réseau, la programmation des gammes Q et FX, le paramétrage d'un Q Motion Controller et de Servo, la conception d'une interface GOT1000. Des simulateurs et des outils de configuration supplémentaires complètent le système

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

de base et la programmation avec étiquettes a été généralisée. MELSOFT Navigator se trouve au cœur d'iQ Works et permet l'intégration des autres programmes de MELSOFT fournis avec iQ Works. Des fonctionnalités comme la conception d'une configuration système, la définition de jeux de paramètres, les libellés système et les lectures par lots aident à réduire les coûts d'exploitation (TCO, *Total Cost of Ownership*). MELSOFT GX Works 2 est l'outil de maintenance et de programmation des API. Il est compatible avec tous les contrôleurs MELSEC, des API compacts de la série MELSEC FX aux API modulaires, y compris le modèle MELSEC System Q, et utilise un environnement de type Windows. Il prend en charge les méthodes de programmation IL, LD et SFC (voir Chapitre 4). Vous pouvez basculer entre IL et LD au cours du développement. Vous pouvez également écrire vos propres blocs fonctionnels et de nombreux utilitaires permettent de configurer des fonctions spéciales. L'environnement de développement comprend des fonctions de diagnostic et des éditeurs puissants pour configurer les réseaux et le matériel MELSEC. Par ailleurs, des fonctions de tests et de surveillance facilitent la création d'applications rapides et efficaces. Vous disposez même d'une simulation hors ligne pour tous les types d'API. Cela permet de simuler les réponses des périphériques et des applications en vue de tests réels.

À l'instar de Mitsubishi, Siemens propose son environnement de développement nommé SIMATIC STEP 7. Il est parfaitement conforme à la norme internationale CEI 61131-3 pour les langages de programmation des API. Avec STEP 7, les programmeurs ont le choix entre différents langages de programmation. Outre LAD et FBD, STEP 7 Basis comprend également le langage IL. Des options supplémentaires pour d'autres langages de la CEI 61131-3, comme ST avec SIMATIC S7-SCL ou SFC avec SIMATIC S7-Graph, constituent des solutions efficaces pour décrire graphiquement des systèmes de commande séquentiels. L'environnement complet comprend des possibilités de diagnostic, des outils de diagnostic des processus, la simulation d'API, la maintenance à distance et la documentation des équipements. S7-PLCSIM est un complément de STEP 7 qui permet de simuler une plate-forme SIMATIC S7 et de tester un programme sur un PC. Vous pouvez ainsi évaluer et corriger le programme avant son installation sur le matériel physique. Grâce aux tests précoces, la qualité globale d'un projet peut être améliorée. L'installation et la mise en service sont alors plus rapides et moins coûteuses, car les défauts du programme sont détectés et corrigés très tôt au cours du développement.

Rockwell Automation fournit RSLogix pour les API PLC-5 d'Allen-Bradley. Les logiciels de programmation par langage à contacts conformes à la norme CEI-1131 de la famille RSLogix™ disposent d'éditeurs souples et conviviaux, d'une interface commune, d'outils de diagnostic et de dépannage, ainsi que de fonctionnalités puissantes qui permettent de gagner du temps. Cette famille de produits est conçue pour fonctionner sur les systèmes d'exploitation Windows de Microsoft. RSLogix™ 5 prend en charge la gamme d'automates programmables PLC-5 d'Allen-Bradley.

Copyright © 2015 Dunod.

1.6 En résumé

Un *automate programmable industriel* (API) est une forme particulière d'automate à base de microprocesseur qui se fonde sur une mémoire programmable pour enregistrer les instructions et mettre en œuvre des fonctions, qu'elles soient logiques, de séquencement, de temporisation, de comptage ou arithmétiques, pour contrôler des machines et des processus. Il est conçu pour être manipulé par des ingénieurs ayant, potentiellement, une connaissance limitée en informatique et en langages de programmation.

En général, un système API est constitué des composants fonctionnels de base suivants : une unité de traitement, de la mémoire, une unité d'alimentation, des interfaces d'entrées-sorties, une interface de communication et une unité de programmation. Pour fonctionner, le système API doit pouvoir accéder aux données à traiter et aux instructions, c'est-à-dire le programme, qui lui indiquent comment traiter les données. Ces informations sont enregistrées dans la mémoire de l'API afin d'y accéder au cours du traitement. Les canaux d'entrées-sorties assurent une fonction d'isolation et de traitement du signal pour que les capteurs et les actionneurs puissent être, le plus souvent, connectés directement aux entrées-sorties sans nécessiter d'autres circuits électroniques. Les sorties peuvent être à relais, à transistors ou à triacs. L'interface de communication est utilisée pour recevoir et transmettre des données sur les réseaux de communication qui relient des API distants. Les API existent principalement sous deux formes mécaniques : un boîtier unique et un système modulaire/rack.

Le document CEI 61131 normalise les API, la section 61131-3 définissant les langages de programmation : langage à contacts (LD, *Ladder Diagram*), listes d'instructions (IL, *Instruction List*), graphes de fonction séquentielle (SFC, *Sequential Function Chart*), texte structuré (ST, *Structured Text*) et diagrammes de schémas fonctionnels (FBD, *Function Block Diagram*).

1.7 Problèmes

Pour les questions 1 à 6, quatre réponses sont proposées : A, B, C et D. Choisissez la réponse correcte.

1. L'acronyme API signifie :
   A. Automate personnel industriel
   B. Automatisme programmable indépendant
   C. Automatisme personnel indépendant
   D. Automate programmable industriel
2. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Sur un API, un canal de sortie à transistor :
   (i) est utilisé uniquement pour la commutation de courant continu ;
   (ii) est isolé de la charge de sortie par un photocoupleur.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?

A. (i) V (ii) V
B. (i) V (ii) F
C. (i) F (ii) V
D. (i) F (ii) F

3. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Sur un API, un canal de sortie à relais :

(i) est utilisé uniquement pour la commutation de courant continu ;
(ii) peut supporter des surtensions temporaires.

Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?

A. (i) V (ii) V
B. (i) V (ii) F
C. (i) F (ii) V
D. (i) F (ii) F

4. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Sur un API, un canal de sortie à triac :

(i) est utilisé uniquement pour la commutation de courant alternatif ;
(ii) est isolé de la charge de sortie par un photocoupleur.

Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?

A. (i) V (ii) V
B. (i) V (ii) F
C. (i) F (ii) V
D. (i) F (ii) F

5. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Le terme fourniture peut être employé pour un appareil connecté à un API lorsque :

(i) le module d'entrée de l'API reçoit un courant à partir d'un appareil d'entrée ;
(ii) le module de sortie de l'API fournit un courant à la charge de sortie.

Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?

A. (i) V (ii) V
B. (i) V (ii) F
C. (i) F (ii) V
D. (i) F (ii) F

6. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Les photocoupleurs sont employés dans les unités d'entrées-sorties pour :

(i) servir de fusibles qui coupent le circuit en cas de tensions ou de courants élevés ;
(ii) isoler le processeur des tensions ou des courants élevés.

Copyright © 2015 Dunod.

Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?

A. (i) V (ii) V
B. (i) V (ii) F
C. (i) F (ii) V
D. (i) F (ii) F

7. Dessinez un schéma de principe qui indique en termes très généraux les principaux composants d'un API.
8. Donnez les caractéristiques des canaux de sorties de type relais, transistor et triac d'un API.
9. Combien de bits peut-on stocker dans une unité de mémoire de 2 K ?
10. Pour un modèle d'API, il est possible d'acheter plusieurs unités de traitement différentes. Une version dispose d'un module de dix entrées-sorties avec six entrées à courant continu et quatre sorties. Elle peut être achetée avec une alimentation en courant alternatif ou en courant continu. Les sorties peuvent être à relais ou à transistor, avec deux versions pour les sorties à transistor (en fourniture ou en absorption de courant). Expliquez les possibilités d'un tel API et la signification des différents types de sorties.

1.8 Recherches complémentaires

1. Sur Google, recherchez « automate programmable industriel » et examinez les formes et les spécifications des API disponibles chez différents fabricants. Recherchez ensuite un API adapté à certaines spécifications, par exemple six entrées à courant continu et six sorties à relais ou six sorties à transistor en absorption de courant, et un module adapté à cinq entrées en fourniture de courant continu, quatre entrées en absorption de courant continu et douze sorties à transistor en absorption de courant continu.
2. Recherchez la norme CEI 61131 et examinez les aspects couverts.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

Copyright © 2015 Dunod.

2 • DISPOSITIFS D'ENTRÉES-SORTIES

Ce chapitre présente brièvement les dispositifs d'entrées-sorties généralement employés avec les API. Pour les entrées, nous nous intéressons aux dispositifs numériques et analogiques, comme les interrupteurs mécaniques de détection de position, les interrupteurs de proximité, les interrupteurs photoélectriques, les encodeurs, les capteurs de température et de pression, les potentiomètres, les transformateurs différentiels à variation linéaire (LVDT, *Linear Variable Differential Transformer*), les extensomètres, les thermistances, les thermorésistances et les thermocouples. Les dispositifs de sorties examinés sont les relais, les contacteurs, les électrovannes et les moteurs.

2.1 Dispositifs d'entrées

Le terme *capteur* désigne un dispositif d'entrée qui fournit une sortie utilisable en réponse à une entrée physique donnée. Par exemple, un thermocouple est un capteur qui convertit une différence de température en une sortie électrique. Le terme *transducteur* désigne généralement un dispositif qui convertit un signal d'une certaine forme en une autre forme physique différente. Les capteurs sont donc souvent des transducteurs, mais c'est également le cas d'autres dispositifs, comme un moteur qui convertit une entrée électrique en une rotation.

Les capteurs qui produisent des sorties numériques ou discrètes, c'est-à-dire tout ou rien, peuvent être facilement connectés aux ports d'entrée des API. Un capteur analogique génère une sortie proportionnelle à la variable mesurée. Ces signaux analogiques doivent être convertis en signaux numériques avant qu'ils puissent être transmis aux ports d'entrée d'un API.

Voici quelques-uns des termes les plus utilisés pour définir les performances d'un capteur :

– La *précision* correspond à l'erreur maximale sur la valeur indiquée par le système ou l'élément de mesure. Par exemple, un capteur de température peut avoir une précision de ±0,1 °C. L'*erreur* d'une mesure correspond à la différence entre le résultat de la mesure et la valeur réelle de la quantité mesurée. Les erreurs peuvent se présenter de différentes manières. Une *erreur de non-linéarité* est due à l'hypothèse d'une relation linéaire entre l'entrée et la sortie sur la plage de travail ; autrement dit, un graphique de la sortie en fonction de

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

l'entrée est supposé donner une ligne droite. Cependant, peu de systèmes ou d'éléments présentent une relation véritablement linéaire et des erreurs surviennent donc en raison de cette hypothèse de linéarité (voir Figure 2.1a). Le terme *erreur d'hystérésis* (voir Figure 2.1b) est employé dans le cas d'une différence sur la valeur d'une même quantité mesurée selon que cette valeur a été atteinte par une augmentation ou une diminution continue. Ainsi, un thermomètre utilisé pour mesurer la même température d'un liquide peut donner une valeur différente selon que cette température mesurée est atteinte par réchauffement ou refroidissement du liquide.

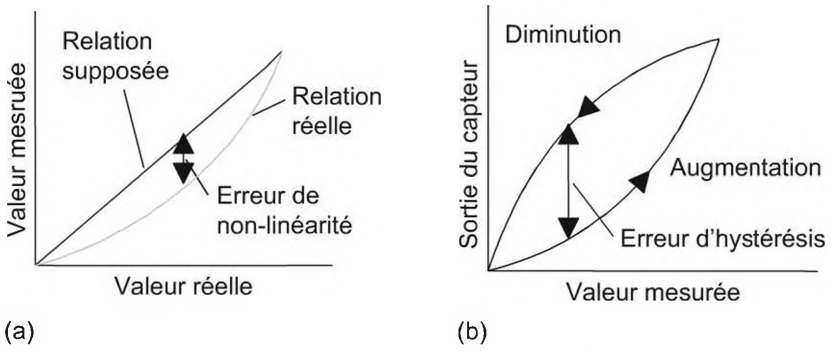


Figure 2.1 – Deux sources d'erreurs : (a) non-linéarité et (b) hystérésis.

- La *plage* d'une variable d'un système correspond aux limites de variation des entrées. Par exemple, un capteur de température à résistance peut avoir une plage de mesure allant de −200 à +800 °C.
- Lorsque la valeur d'entrée d'un capteur change, il lui faut un certain temps pour atteindre et conserver la valeur d'état stationnaire (voir Figure 2.2). Le *temps de réponse* correspond au temps écoulé après que l'entrée d'un système ou d'un élément a brutalement augmenté de zéro à une valeur constante, jusqu'au point où le système ou l'élément produit une sortie qui correspond à un pourcentage donné, par exemple 95 %, de la valeur d'entrée. Le *temps de montée* correspond au temps mis par la sortie pour passer d'un pourcentage donné à l'état stationnaire. Très souvent, le temps de montée fait référence au temps mis par la sortie pour passer de 10 % de la valeur d'état stationnaire à 90 ou 95 % de cette valeur. Le *temps d'établissement* est le temps mis par la sortie pour se fixer à un pourcentage, par exemple 2 %, de la valeur d'état stationnaire.
- La *sensibilité* indique la variation de la sortie d'un système d'instrumentation ou d'un élément lorsque la grandeur mesurée change d'une quantité donnée ; autrement dit, il s'agit du rapport entrée/sortie. Par exemple, un thermocouple pourrait avoir une sensibilité de 20 µV/°C, ce qui correspond à une sortie de 20 µV pour chaque changement de 1 °C en température.
- La *stabilité* d'un système concerne sa capacité à produire la même sortie lorsqu'il est utilisé pour mesurer une entrée constante pendant un certain temps. La *dérive* désigne la variation de la sortie au fil du temps. Elle peut être exprimée comme un pourcentage de la plage de sortie. L'expression *dérive à zéro* fait référence à la variation qui se produit sur la sortie lorsqu'il n'y a pas d'entrée.

Copyright © 2015 Dunod.

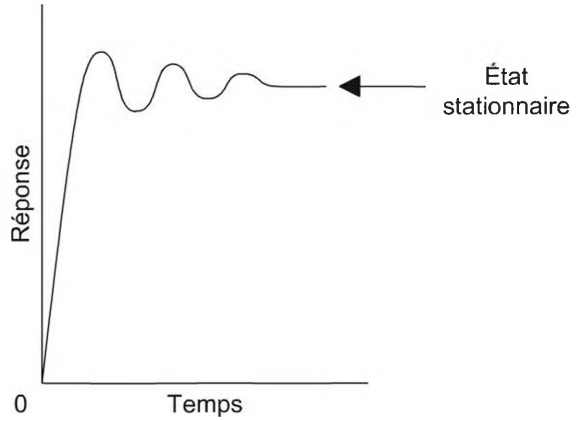


Figure 2.2 – Réponse d'un capteur ou d'un système de mesure à une entrée soudaine. Vous pouvez constater une telle réponse par lecture d'un ampèremètre lorsqu'un circuit électrique est brusquement mis sous tension.

- Le terme *répétabilité* fait référence à la capacité d'un système de mesure à donner la même valeur pour des mesures répétées de la même valeur d'une variable. L'absence de répétabilité peut être due à des fluctuations aléatoires de l'environnement, par exemple des modifications de température ou d'humidité. L'erreur due à la répétabilité s'exprime généralement comme un pourcentage de la plage de sortie. Par exemple, un capteur de pression peut avoir une répétabilité égale à ±0,1 % de la plage complète. Pour une plage de 20 kPa, cela donne une erreur de ±20 Pa.
- La *fiabilité* d'un système de mesure, ou d'un élément d'un tel système, se définit comme la probabilité qu'il fonctionne à un niveau de performance convenu pendant une durée précisée, sous des conditions environnementales définies. Le niveau de performance convenu peut correspondre à une précision particulière du système de mesure.

Pour illustrer l'emploi de ces termes dans une spécification, voici les caractéristiques du capteur de pression MPX100AP (voir Section 2.1.8 pour une description de ce capteur) :

Tension d'alimentation : 3 V (max 6 V)

Courant de sortie : 6 mA

Mesure de pleine échelle : 60 mV

Plage : 0 à 100 kPa

Sensibilité : 0,6 mV/kPa

Erreur de non-linéarité : ±0,05 % de la plage totale

Hystérésis de température : ±0,5 % de la plage totale

Résistance d'entrée : 400 à 550 Ω

Temps de réponse : 1 ms (10 % à 90 %)

Les sections suivantes décrivent quelques dispositifs d'entrée parmi les plus utilisés avec les API, ainsi que leurs capteurs.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

2.1.1 Interrupteurs mécaniques

Un interrupteur mécanique génère un ou plusieurs signaux tout ou rien en réponse à une entrée mécanique qui provoque l'ouverture ou la fermeture de l'interrupteur. Ce type d'interrupteur peut être utilisé pour indiquer la présence d'une pièce sur un chariot d'usinage ; la pièce entre en contact avec l'interrupteur et le ferme. L'absence de la pièce est indiquée par un interrupteur ouvert, tandis que sa présence est indiquée par un interrupteur fermé. Par conséquent, conformément au schéma de la Figure 2.3a, le signal d'entrée sur un seul canal d'entrée de l'API a les niveaux logiques suivants :

Pièce absente : 0

Pièce présente : 1

Le niveau 1 peut correspondre à une entrée de tension continue de 24 V, le niveau 0, à une entrée de 0 V.

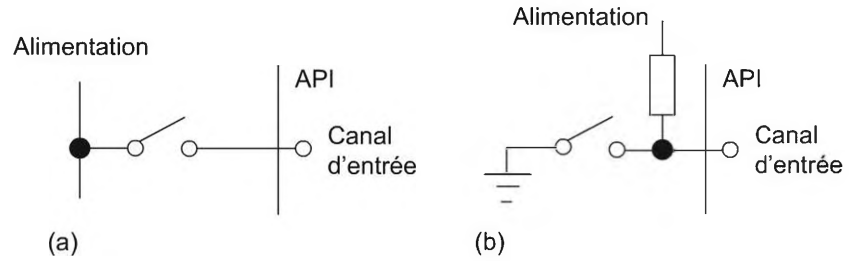


Figure 2.3 – Interrupteurs mécaniques.

Avec le circuit de la Figure 2.3b, la tension d'alimentation est appliquée à l'entrée de l'API lorsque l'interrupteur est ouvert ; lorsque l'interrupteur est fermé, la tension d'entrée tombe à la valeur basse. Dans ce cas, les niveaux logiques sont les suivants :

Pièce absente : 1

Pièce présente : 0

Les interrupteurs sont disponibles avec des contacts *normalement ouverts* (NO) ou *normalement fermés* (NF). Certains peuvent également être configurés dans l'un des deux modes en choisissant les contacts appropriés. Les contacts d'un interrupteur NO sont ouverts en l'absence d'entrée mécanique, qui est donc utilisée pour fermer l'interrupteur. Les contacts d'un interrupteur NF sont fermés en l'absence d'entrée mécanique, qui est donc utilisée pour ouvrir l'interrupteur. Les interrupteurs mécaniques sont définis par leur nombre d'ensembles de contacts (*poles*), c'est-à-dire le nombre de circuits séparés qui peuvent être activés lors d'une commutation, et par leur nombre de positions de contacts (*throw*), c'est-à-dire le nombre de contacts individuels dans chaque ensemble.

Les interrupteurs mécaniques souffrent d'un problème de *rebond*. Lorsque l'interrupteur est actionné, que ce soit pour l'ouvrir ou le fermer, un rebond se produit sur les contacts, qui ne sont pas alors clairement ouverts ou fermés. En raison de

Copyright © 2015 Dunod.

leur élasticité, les contacts rebondissent à la manière d'un ressort. Ce rebond peut produire des amplitudes qui changent les niveaux logiques pendant, potentiellement, plus de 20 ms et une seule action de l'interrupteur peut générer plusieurs signaux à la place du seul attendu. Il existe plusieurs manières d'éliminer ces signaux parasites. Une solution consiste à inclure dans le programme un retard d'approximativement 20 ms après la première transition détectée du signal, avant de lire les signaux suivants. Pour les interrupteurs unipolaires inverseurs (SPDT, *Single Pole/Double Throw*), il est possible d'utiliser deux portes logiques NON-ET (voir Chapitres 3 et 5), comme l'illustre la Figure 2.4a. Lorsque l'interrupteur est en position A, la sortie est au niveau logique 1. Lorsque l'interrupteur bascule en position B, la sortie passe au niveau logique 0 et reste verrouillée à ce niveau même en présence de rebonds. La Figure 2.4b montre une bascule D (voir Chapitre 3) qui permet de supprimer les rebonds sur un interrupteur unipolaire simple (SPST, *Single Pole/Single Throw*). La sortie de la bascule D ne change pas tant qu'un front d'horloge n'est pas appliqué. S'il est supérieur à la durée du rebond, l'effet du rebond disparaît en sortie.

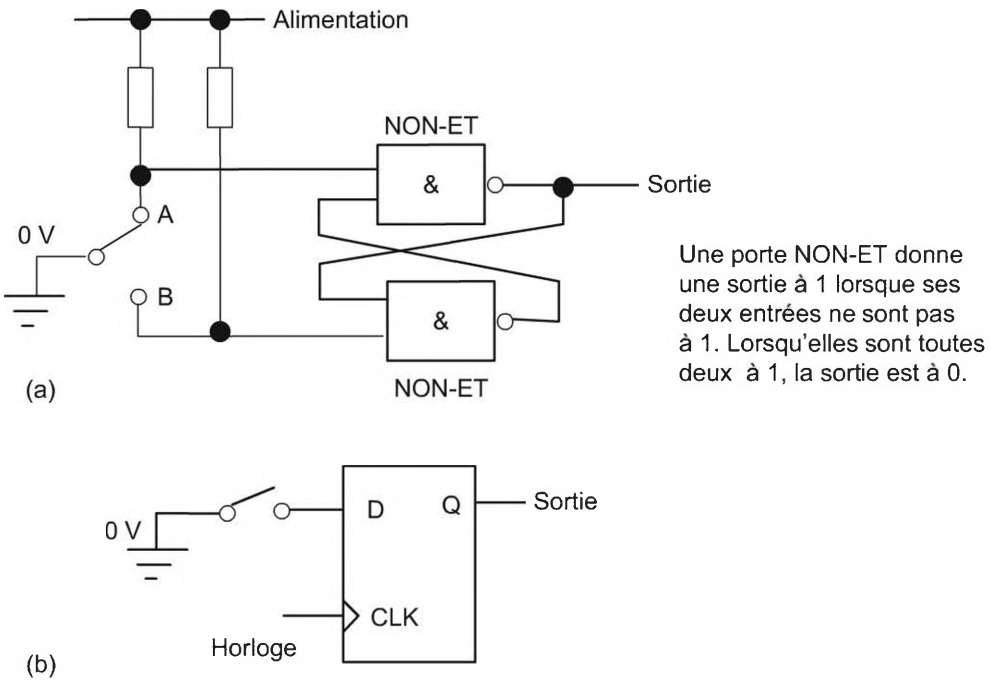


Figure 2.4 – (a) Des portes NON-ET pour annuler les rebonds sur un interrupteur SPDT et (b) une bascule D pour annuler les rebonds sur un interrupteur SPST.

Un *interrupteur de fin de course* est un interrupteur utilisé pour détecter la présence ou l'absence d'un objet, le passage d'une pièce en déplacement et l'arrivée d'un objet à la fin de son trajet. Cette dénomination leur a été donnée car ils ont été initialement employés pour déterminer la fin du déplacement d'un objet. S'ils sont très répandus, c'est en raison de leur robustesse, fiabilité et facilité d'installation. L'élément de base d'un tel organe est un interrupteur électrique intégré, qui est actionné au moyen d'un piston ; l'action de ce plongeur est commandé par la tête de l'actionneur, qui transfère la force et le mouvement extérieurs à l'interrup-

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

teur interne (voir Figure 2.5a). En fonction de l'objet et du mouvement à détecter, la tête de l'actionneur aura une forme particulière. L'une des plus répandues est constituée d'un galet, qui fait pivoter un levier afin d'activer les contacts de l'interrupteur, et d'un ressort de rappel (voir Figure 2.5b).

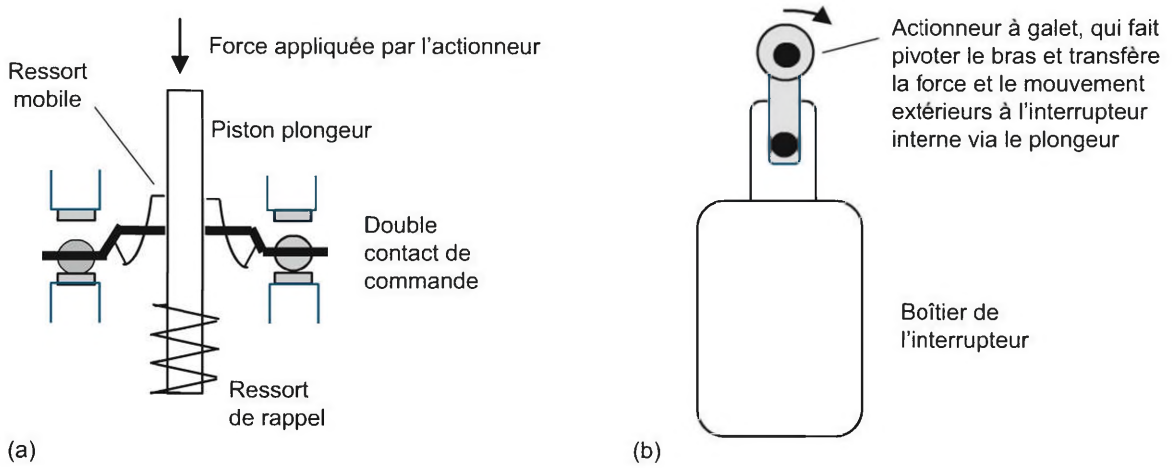


Figure 2.5 – Forme de base de (a) l'interrupteur interne et (b) d'un interrupteur de fin de course actionné par un galet.

Pour illustrer les différents types d'interrupteurs de fin de course disponibles sur le marché, voici quelques modèles proposés par le fabricant Rockwell Automation. La famille 801 comprend des interrupteurs de fin de course d'usage général adaptés à une grande diversité d'applications et proposés dans différentes configurations de contacts. Dans le cas d'un levier à galet, l'opération de contact peut être une action lente avec ressort de rappel, une action brusque avec ressort de rappel, un mécanisme à déclic avec maintien ou une action brusque avec maintien. Dans le cas d'une action brusque, le déplacement de l'actionneur crée un changement rapide de l'état des contacts lorsqu'il atteint sa position de déclenchement. Pour l'action lente, les contacts sont activés à une vitesse proportionnelle à celle du déplacement de l'actionneur. Pour le mécanisme à déclic, le déplacement du levier vers la droite active les contacts. Le levier est équipé d'un ressort de rappel, mais les contacts restent dans leur position jusqu'au prochain déplacement du levier à galet. Dans le cas d'un interrupteur à action brusque avec maintien, les contacts sont activés lorsque le levier est déplacé dans un sens et réinitialisés lorsqu'il est déplacé dans le sens opposé. L'angle de rotation du levier nécessaire à l'activation de l'interrupteur peut être sélectionné et va de quelques degrés jusqu'à environ 25.

Omron Industrial Automation propose également toute une gamme d'interrupteurs de fin de course. Par exemple, les modèles miniatures D4CC sont disponibles sous différentes formes : avec plongeur à goupille, plongeur à galet, plongeur à galet transversal, plongeur à galet haute sensibilité, plongeur à galet étanche, plongeur à galet transversal étanche, tige plastique et levier à galet central.

Illustrons l'usage des interrupteurs de fin de course en prenant l'exemple du déplacement d'un ascenseur entre les étages. Un tel interrupteur peut être placé au

Copyright © 2015 Dunod.

niveau du sol afin de détecter la présence de l'ascenseur par action sur la tête de l'interrupteur. Un signal peut alors être généré afin de démarrer ou d'arrêter le moteur de l'ascenseur.

Les *interrupteurs de niveau de liquide* contrôlent le niveau des liquides dans des réservoirs. Il s'agit principalement de flotteurs verticaux qui se déplacent en fonction du niveau du liquide et ce mouvement est utilisé pour actionner les contacts d'un interrupteur.

2.1.2 Interrupteurs de proximité

Les *interrupteurs de proximité* permettent de détecter la présence d'un objet sans entrer en contact avec lui. Il existe plusieurs formes d'interrupteurs de ce type, certains étant adaptés uniquement aux objets métalliques.

Un interrupteur de proximité à *courants de Foucault* est constitué d'une bobine alimentée par un courant alternatif constant et produit un champ magnétique alternatif constant. Lorsqu'un objet métallique approche de l'interrupteur, des courants de Foucault sont induits dans cet objet (voir Figure 2.6a). Le champ magnétique résultant de ces courants de Foucault induit une force électromotrice dans la bobine, ce qui oblige à modifier la tension dans la bobine pour que le courant qui la traverse reste constant. L'amplitude de la tension correspond donc à une mesure de proximité d'un objet métallique. La tension peut servir à activer un interrupteur électronique, souvent un transistor dont la sortie passe du niveau bas au niveau haut suite au changement de tension, afin de créer un dispositif tout ou rien. La plage de détection de tels objets est en générale située entre 0,5 et 20 mm.

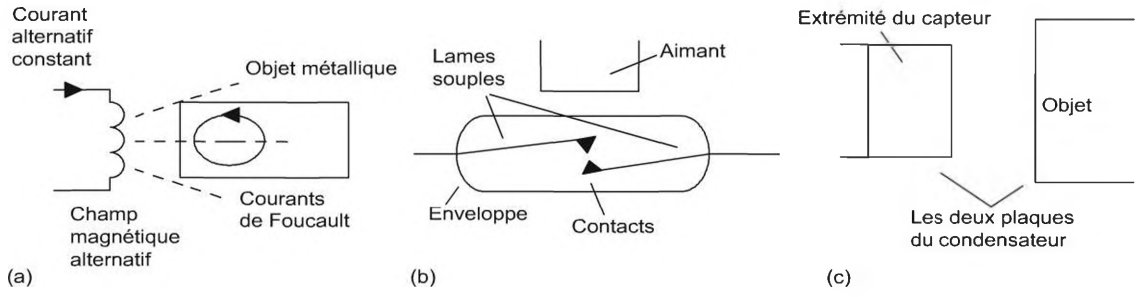


Figure 2.6 – Interrupteurs de proximité : (a) à courants de Foucault, (b) Reed et (c) capacitifs.

Les *interrupteurs Reed* sont constitués de deux lames souples en matériau ferromagnétique qui se chevauchent sans se toucher et qui sont scellées dans une enveloppe plastique ou en verre (voir Figure 2.6b). Lorsqu'un aimant ou une bobine alimentée approche de l'interrupteur, les lames sont magnétisées et s'attirent. Les contacts se ferment alors. En général, l'aimant ferme les contacts lorsqu'il se trouve à environ 1 mm de l'interrupteur. Ces interrupteurs sont très utilisés dans les systèmes d'alarme pour la détection de l'ouverture d'une porte. L'aimant est fixé sur la porte, tandis que l'interrupteur Reed est placé sur le cadre. Lorsque la porte s'ouvre, l'interrupteur s'ouvre.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

Un *interrupteur de proximité capacitif* peut être utilisé avec des objets métalliques ou non. La capacité électrique de deux plaques séparées d'une certaine distance dépend de cette distance ; plus la séparation est faible, plus la capacité est élevée. Le capteur de l'interrupteur de proximité capacitif correspond à l'une des plaques du condensateur, tandis que la seconde est représentée par l'objet métallique dont la proximité doit être détectée (voir Figure 2.6c). Cette proximité est donc détectée par un changement de capacité. Ce type de capteur est également capable de détecter les objets non métalliques, car la capacité d'un condensateur dépend du diélectrique entre ses plaques. Dans ce cas, les plaques sont représentées par le capteur et la terre, tandis que l'objet non métallique est le diélectrique. Le changement de capacité peut être utilisé pour activer un interrupteur électronique et créer ainsi un dispositif tout ou rien. Les interrupteurs de proximité capacitifs sont employés pour détecter des objets qui se trouvent entre 4 et 60 mm du capteur. Par exemple, un capteur de ce type pourrait être utilisé pour déterminer si une boîte en carton contient un gâteau lorsque ces boîtes se déplacent sur une bande transporteuse.

Prenons comme exemple le détecteur capacitif E2K-X d'Omron. Il peut être utilisé avec un large éventail d'objets métalliques ou non, par exemple en verre, en bois ou en plastique, à des distances situées entre 3 et 30 mm. Les détecteurs de proximité capacitifs trouvent également des applications en tant que capteurs tactiles dans les interfaces utilisateurs, par exemple les pavés tactiles des ordinateurs et les écrans tactiles des téléphones mobiles. Ces écrans sont constitués d'un isolant, comme du verre, revêtu d'un conducteur transparent. Puisque le corps humain est conducteur, son contact avec la surface de l'écran modifie la capacité de celui-ci (voir Wikipédia pour de plus amples informations).

Un *interrupteur de proximité inductif* est constitué d'une bobine enroulée autour d'un cœur métallique ferreux. Lorsqu'une extrémité de ce noyau est placée à proximité d'un objet métallique ferreux, la quantité de noyau métallique associé à la bobine est modifiée et, par conséquent, son inductance change. Cette variation peut être surveillée à l'aide d'un circuit résonnant, la présence d'un objet métallique ferreux modifiant le courant dans ce circuit. Le courant peut ensuite activer un interrupteur électronique et créer ainsi un dispositif tout ou rien. En général, la plage de détection des objets est située entre 2 et 15 mm. Ce type de capteur est employé pour détecter si les bouteilles qui se déplacent sur une bande transporteuse sont fermées par une capsule métallique. Par exemple, le capteur E2F d'Omron est capable de détecter des objets métalliques situés à une distance allant jusqu'à 8 mm.

2.1.3 Capteurs et interrupteurs photoélectriques

Les interrupteurs photoélectriques peuvent fonctionner selon un *principe transmissif*, dans lequel l'objet détecté coupe un faisceau lumineux, généralement un rayonnement infrarouge, et l'empêche d'atteindre le détecteur (voir Figure 2.7a). Par exemple, la Figure 2.7b, fondée sur un dispositif en U, montre un objet qui coupe le faisceau lumineux. Ils peuvent également fonctionner selon un *principe réflectif*, dans lequel l'objet détecté reflète un faisceau lumineux vers le détecteur (voir Figure 2.7c). Le capteur transmissif est utilisé pour compter les pièces transportées par un tapis roulant, qui coupent le faisceau lumineux. Le type réflectif peut détecter si le niveau de liquide dans un conteneur transparent est correct.

Copyright © 2015 Dunod.

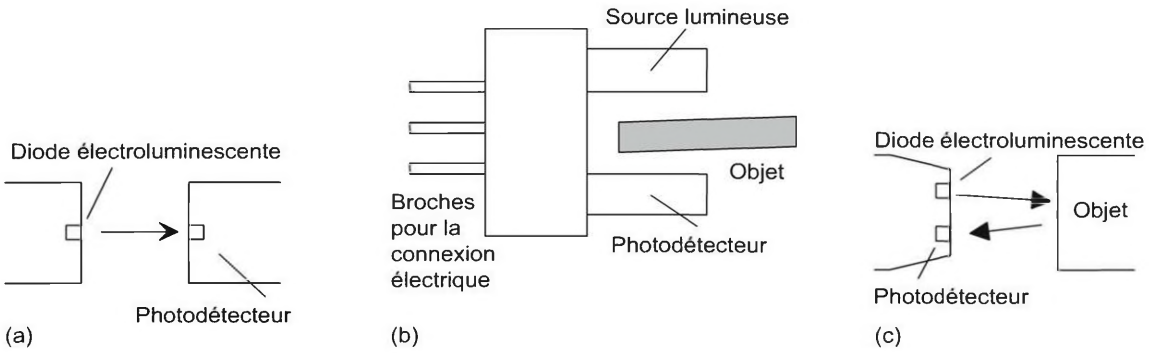


Figure 2.7 – Capteurs photoélectriques.

L'émetteur du rayonnement est généralement une *diode électroluminescente* (LED, *Light-Emitting Diode*). Le détecteur du rayonnement peut être un *phototransistor*, le plus souvent un couple de transistors, appelé *couple de Darlington*, pour augmenter la sensibilité. Selon le circuit employé, la sortie peut passer à l'état haut ou bas lorsque la lumière frappe le transistor. Ces capteurs sont utilisés pour détecter la présence d'objets à distance faible, généralement autour de 5 mm. La *photodiode* est un autre détecteur possible. Selon le circuit employé, la sortie peut passer à l'état haut ou bas lorsque la lumière frappe la diode. Il existe également la *cellule photoconductrice*, dont la résistance, souvent en sulfure de cadmium, dépend de l'intensité du rayon lumineux.

Dans ces capteurs, la lumière est convertie en changement de courant, de tension ou de résistance. Si la sortie doit être utilisée comme une mesure de l'intensité de la lumière, non simplement pour détecter la présence ou l'absence d'un objet sur le chemin du faisceau, le signal doit être amplifié et transformé par un convertisseur analogique-numérique. Une autre solution consiste à utiliser un convertisseur lumière-fréquence, qui transforme la lumière en un train d'impulsions dont la fréquence dépend de l'intensité lumineuse. Les capteurs intégrés, comme le circuit TSL220 de Texas Instrument, incluent le capteur de lumière et le convertisseur tension-fréquence (voir Figure 2.8).

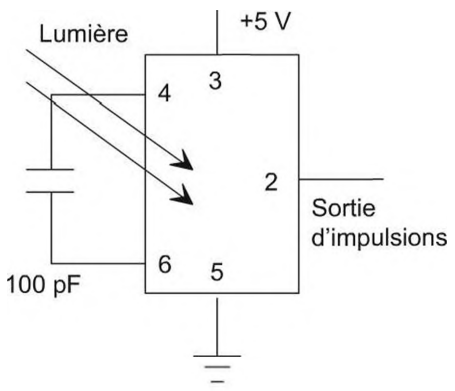


Figure 2.8 – Le TSL220.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

2.1.4 Encodeurs

Le terme *encodeur* est utilisé pour les dispositifs qui fournissent une sortie numérique fonction d'un déplacement angulaire ou linéaire. Un *encodeur incrémental* détecte les changements dans un déplacement angulaire ou linéaire à partir d'une position donnée ; un *encodeur absolu* fournit la position linéaire ou l'angle réels.

La Figure 2.9 illustre la forme de base d'un encodeur incrémental qui mesure un déplacement angulaire. Un rayon lumineux, sans doute issu d'une LED, passe au travers des encoches réalisées dans un disque et est détecté par un capteur lumineux, comme une photodiode ou un phototransistor. Lorsque le disque tourne, le rayon lumineux est alternativement transmis et stoppé. Un train d'impulsions se produit donc en sortie du capteur lumineux. Le nombre d'impulsions est proportionnel à l'angle de rotation du disque, la résolution étant proportionnelle au nombre d'encoches sur le disque. Avec 60 encoches, et puisqu'une révolution correspond à une rotation de 360°, un déplacement d'une encoche à la suivante correspond à une rotation de 6°. En décalant les encoches, il est possible d'en obtenir plus d'un millier par révolution et d'avoir ainsi une résolution beaucoup plus élevée.

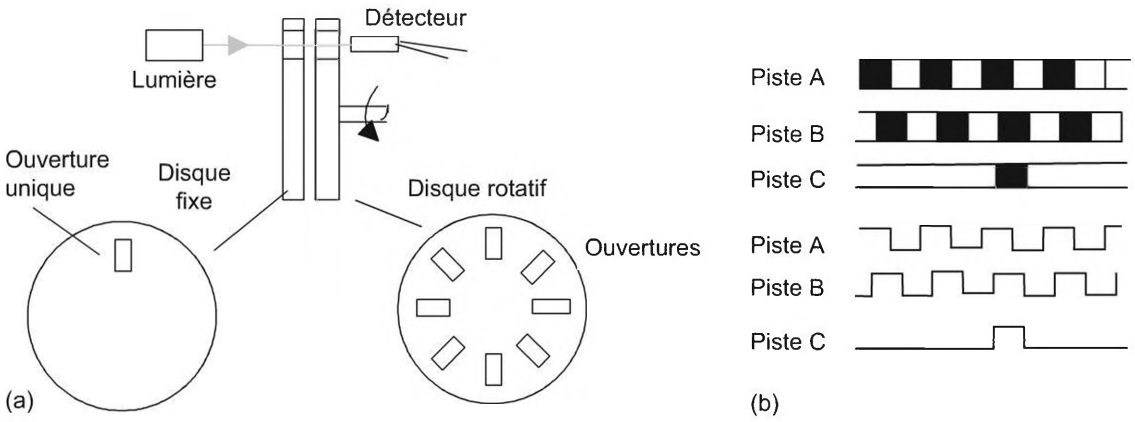


Figure 2.9 – (a) Un encodeur incrémental de base et (b) un système à trois pistes.

L'encodeur incrémental à une piste est très basique, sans possibilité de déterminer le sens de rotation. Quel que soit le sens de rotation, la sortie est identique. Par conséquent, les encodeurs disposent généralement de deux ou trois pistes avec des capteurs (voir Figure 2.9b). Dans un modèle à deux pistes, l'une est décalée d'un quart de cycle par rapport à l'autre. Ainsi, selon le sens de rotation, la sortie d'une piste va se trouver devant ou derrière celle de l'autre piste. Une troisième piste comprenant une seule ouverture est également ajoutée. Elle produit une impulsion par révolution et peut donc servir à compter le nombre de révolutions complètes.

L'encodeur absolu diffère de l'encodeur incrémental car il dispose de motifs d'encoches qui identifient de manière unique chaque position angulaire. Dans la version illustrée à la Figure 2.10, le disque rotatif comprend quatre cercles concentriques d'encoches et quatre capteurs pour détecter les faisceaux lumineux. Les encoches sont organisées de manière que la sortie séquentielle des capteurs soit un

Copyright © 2015 Dunod.

nombre binaire, chaque nombre correspondant à une position angulaire précise. Avec quatre pistes, le nombre est codé sur quatre bits et le nombre de positions détectables est égal à $2^4 = 16$, ce qui correspond à une résolution de 360/16 = 22,5°. En général, les encodeurs possèdent entre dix et douze pistes. Le nombre de bits de la valeur binaire est égal au nombre de pistes. Par conséquent, avec dix pistes, la valeur est codée sur dix bits et le nombre de positions détectables est égal à 2^{10}, c'est-à-dire 1 024, avec une résolution de 360/1 024 = 0,35°.

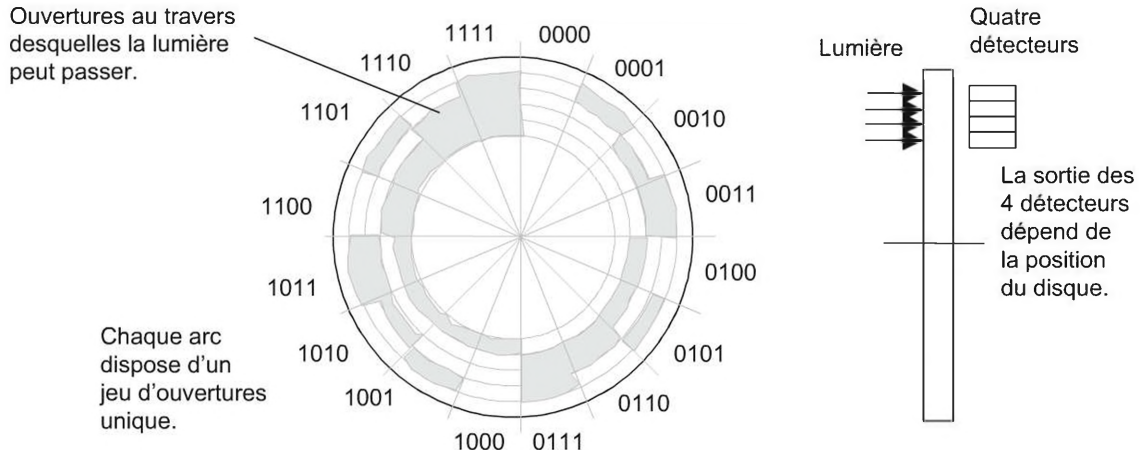


Figure 2.10 – Un encodeur absolu de base.

Si la Figure 2.10 illustre la forme normale d'un code binaire, en pratique, une version modifiée du code binaire, appelée *code de Gray* ou *code binaire réfléchi*, est généralement employée. Contrairement au code binaire normal, dans le code de Gray, un seul bit change lors du passage d'un nombre au suivant (voir Tableau 2.1). Ce code permet d'obtenir des données avec l'incertitude la plus faible, mais, puisque les systèmes utiliseront probablement le code binaire, un circuit de conversion du code Gray au code binaire doit être ajouté.

2.1.5 Capteurs de température

Un *élément bimétallique* est une forme simple de capteur de température, qui permet d'obtenir un signal tout ou rien lorsqu'une température précise est atteinte. Il est constitué de deux lames de métal différent, par exemple laiton et fer, soudées ensemble (voir Figure 2.11). Les deux métaux présentent des coefficients de dilatation différents. Par conséquent, lorsque la température du bilame augmente, les lames se courbent de manière que l'un des métaux puisse se dilater plus que l'autre. Le métal ayant le coefficient de dilatation le plus élevé se trouve à l'extérieur de la courbe. Lorsque les lames se refroidissent, l'effet de cintrage est inversé. Ce mouvement peut servir à établir ou à rompre un contact électrique et, par conséquent, à produire, à une température donnée, un courant tout au rien dans un circuit électrique. Si le dispositif n'est pas très précis, il est fréquemment employé dans les thermostats des systèmes de chauffage central en raison de sa simplicité et de sa robustesse.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

Tableau 2.1 Code binaire et code de Gray

Code binaire				Code de Gray			
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	1	1
0	0	1	1	0	0	1	0
0	1	0	0	0	1	1	0
0	1	0	1	0	1	1	1
0	1	1	0	0	1	0	1
0	1	1	1	0	1	0	0
1	0	0	0	1	1	0	0
1	0	0	1	1	1	0	1
1	0	1	0	1	1	1	1
1	0	1	1	1	1	1	0
1	1	0	0	1	0	1	0
1	1	0	1	1	0	1	1
1	1	1	0	1	0	0	1
1	1	1	1	1	0	0	0

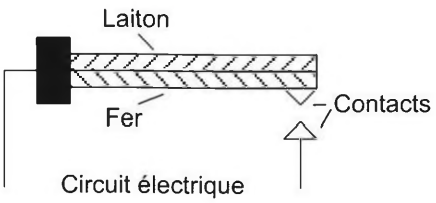


Figure 2.11 – Un bilame.

Le *thermomètre à résistance de platine* (RTD, *Resistive Temperature Detector*) est une autre forme de capteur de température. La résistance électrique des métaux ou des semi-conducteurs varie en fonction de la température. Le platine, le nickel ou les alliages de nickel font partie des métaux les plus employés. Ces détecteurs peuvent être utilisés dans une branche d'un pont de Wheatstone, la sortie du pont mesurant la température (voir Figure 2.12a). Dans un tel pont, la sortie est nulle

Copyright © 2015 Dunod.

lorsque les résistances des branches du pont sont telles que $P/Q = R/S$. Toute variation d'une résistance qui rompt cet équilibre conduit à une sortie non nulle. Si la résistance varie de manière linéaire avec la température sur une vaste gamme de températures, le changement de résistance par degré est relativement faible. Le problème d'un thermomètre à résistance réside dans les fils qui le connectent au pont. En effet, ces fils peuvent être relativement longs et donc présenter une résistance significative, qui varie avec la température. Pour contourner ce problème, une solution consiste à utiliser un circuit à trois fils (voir Figure 2.12b). Les modifications de la résistance des fils affectent deux branches du pont, ce qui permet de conserver son équilibre. Les détecteurs de ce type sont très stables et très précis, quoique onéreux. Ils sont disponibles sous forme d'éléments câblés dans des tubes céramiques ou d'éléments à couche mince déposés sur un substrat approprié.

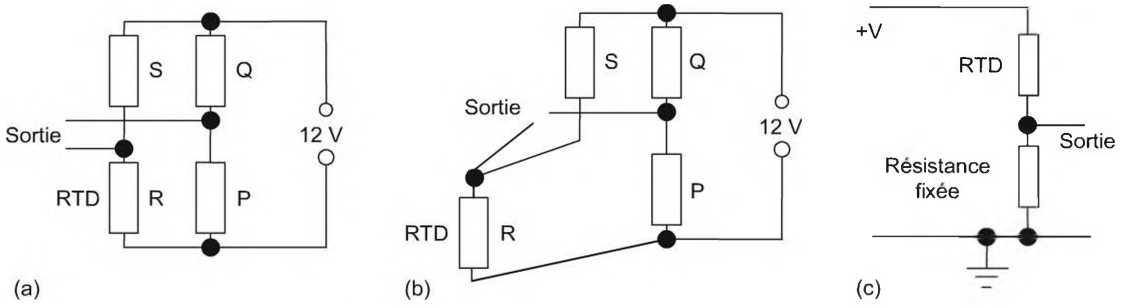


Figure 2.12 – (a) Un pont de Wheatstone, (b) un circuit à trois fils et (c) un circuit diviseur de tension.

Les semi-conducteurs, comme les thermistances (voir Figure 2.13), affichent des changements de résistance très importants selon la température. Cependant, ces changements ne sont pas linéaires. Les thermistances à coefficient de température négatif (CTN) voient leur résistance décroître lorsque la température augmente. À l'opposé, les thermistances à coefficient de température positif (CTP) voient leur résistance augmenter avec la température. Elles peuvent être utilisées dans un pont de Wheatstone, mais elles sont généralement employées dans un circuit diviseur de tension où la modification de la résistance de la thermistance change la tension appliquée à une résistance (voir Figure 2.12c). La sortie de ces deux types de circuit est un signal analogique qui mesure la température. Les thermistances ont l'avantage d'être peu coûteuses et de taille réduite, d'afficher de grandes variations de résistance et de réagir rapidement aux changements de température. En revanche, elles ont l'inconvénient de ne pas être linéaires et d'avoir des plages de température limitées.

Les *thermodiodes* et les *thermotransistors* peuvent servir de capteurs de température, car le débit des électrons et des trous dans les jonctions est affecté par la température. Les circuits intégrés peuvent se fonder sur un tel élément sensible à la température pour fournir une tension de sortie qui varie en fonction de la température. Le composant LM35 est largement utilisé. Il produit une sortie de 10 mV/°C lorsqu'il est alimenté en +5 V (voir Figure 2.14a). Un interrupteur de température numérique peut se baser sur un capteur analogique si la sortie analogique est envoyée à un amplificateur qui la compare à une valeur définie. La sortie de ce

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

comparateur est au niveau logique 1 lorsque la tension d'entrée est supérieure ou égale à la tension de référence, sinon, au niveau logique 0. Certains circuits intégrés, comme le LM3911N, combine un élément thermorésistant sensible à la température et un amplificateur opérationnel. Lorsque l'amplificateur est branché en comparateur (voir Figure 2.14b), la sortie bascule au moment où la température passe le seuil de référence ; on obtient ainsi un contrôleur de température tout ou rien. Ces capteurs de température ont l'avantage d'être bon marché et d'avoir une réponse raisonnablement linéaire. Toutefois, ils ont l'inconvénient de présenter une plage de température limitée.

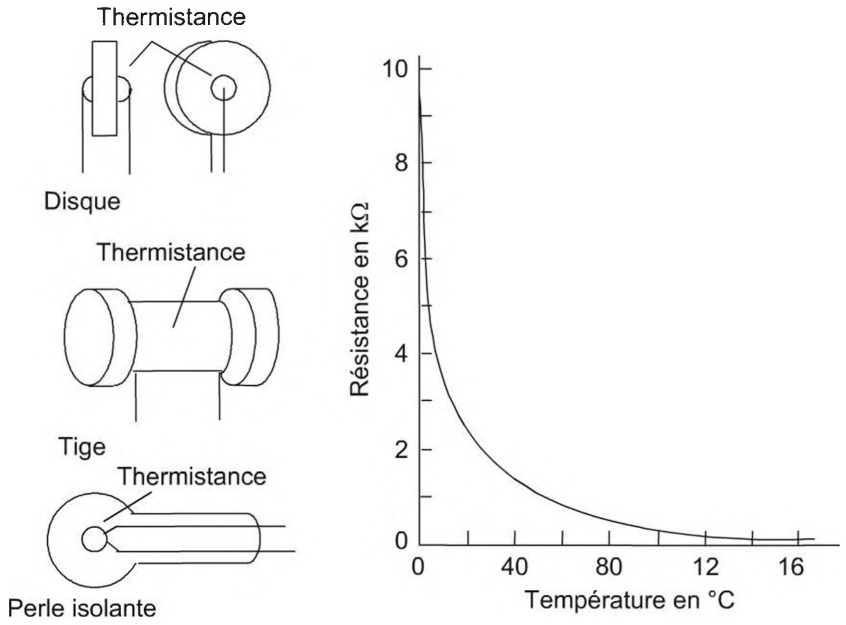


Figure 2.13 – Formes classiques des thermistances et variation de la résistance en fonction de la température pour une thermistance CTN.

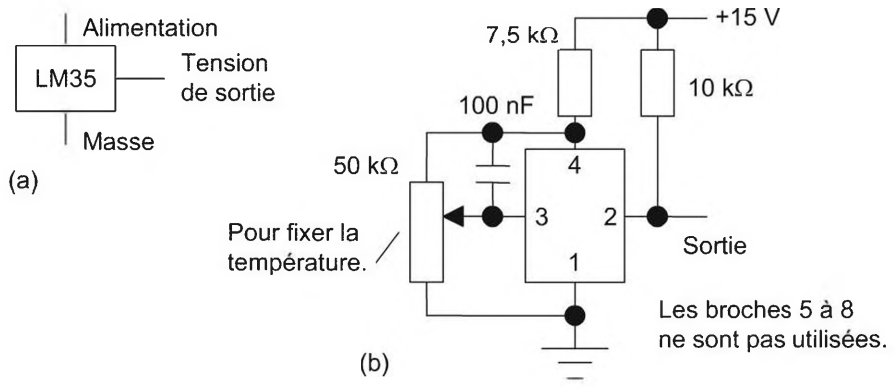


Figure 2.14 – (a) Le LM35 et (b) le LM3911N pour une commande tout ou rien.

Les *thermocouples* sont également très employés en capteurs de température. Un thermocouple est constitué de deux fils dissemblables, A et B, qui forment une jonction (voir Figure 2.15). Lorsque la jonction est chauffée afin d'atteindre une

Copyright © 2015 Dunod.

température supérieure à celle des autres jonctions du circuit, qui conservent une température froide constante, une force électromotrice dépendante de la température de la jonction chaude est produite. Les valeurs de force électromotrice d'un thermocouple sont données au Tableau 2.2, en supposant que la température de la jonction froide est égale à 0 °C. La tension du thermocouple est faible et doit être amplifiée avant d'être transmise au canal d'entrée analogique d'un API. Un circuit doit également compenser la température de la jonction froide, puisqu'elle sera rarement à 0 °C, mais plutôt à la température ambiante, et que cette température affecte la valeur de la force électromotrice. L'amplification et la compensation, ainsi que les filtres qui réduisent les effets des interférences du réseau électrique, sont souvent combinés dans une unité de traitement du signal. Les thermocouples ont l'avantage de pouvoir mesurer la température de pratiquement n'importe quel point, d'être solides et de pouvoir fonctionner sur une large gamme de températures. Ils ont l'inconvénient de donner une réponse non linéaire, avec de petits changements de force électromotrice pour chaque changement de degré, et d'exiger une compensation de température pour la jonction froide.

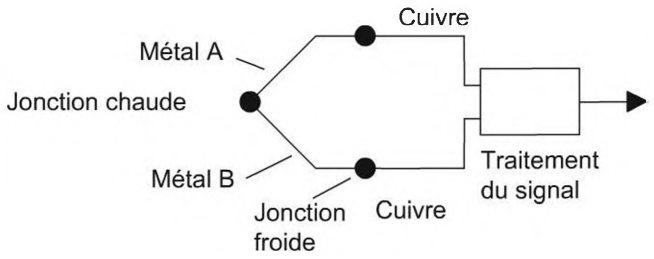


Figure 2.15 – Un thermocouple.

Tableau 2.2 Valeurs de force électromotrice pour quelques thermocouples

Référence	Matériaux	Plage (°C)	µV/°C
B	Platine, 30 % rhodium/platine, 6 % rhodium	0 à 1 800	3
E	Chromel/constantan	–200 à 1 000	63
J	Fer/constantan	–200 à 900	53
K	Chromel/alumel	–200 à 1 300	41
R	Nicrosil/nisil	–200 à 1 300	28
N	Platine/platine, 13 % rhodium	0 à 1 400	6
S	Platine/platine, 10 % rhodium	0 à 1 400	6
T	Cuivre/constantan	–200 à 400	43

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

2.1.6 Capteurs de position/déplacement

Un *capteur de position* mesure la distance entre un point de référence et l'emplacement actuel de la cible, tandis qu'un *capteur de déplacement* mesure la distance entre l'emplacement actuel de la cible et un emplacement précédemment enregistré.

Les *capteurs de position linéaires et angulaires résistifs* sont très utilisés et relativement bon marché. Ils sont également appelés *potentiomètres linéaires et rotatifs*. Une tension continue est appliquée sur la totalité de la piste résistive. La tension entre un curseur qui se déplace sur la piste et l'une de ses extrémités dépend de la position du curseur entre les deux extrémités de la piste résistive du potentiomètre (voir Figure 2.16). Le potentiomètre est donc un capteur de position linéaire ou angulaire analogique.

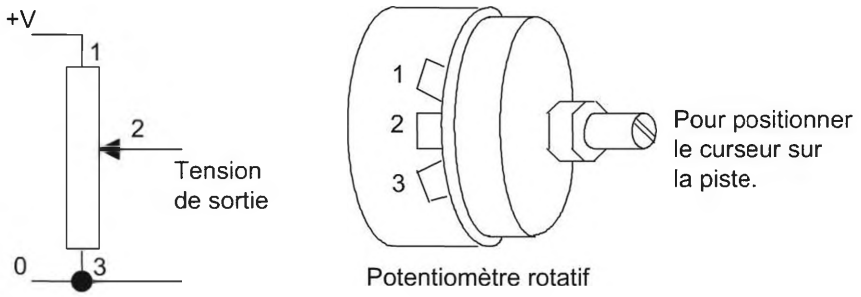


Figure 2.16 – Un potentiomètre.

Le *transformateur différentiel à variation linéaire* (LVDT, *Linear Variable Differential Transformer*) est un autre type de capteur de déplacement dont la tension de sortie est fonction de la position d'un noyau ferreux. Le LVDT est constitué de trois bobines placées symétriquement au travers desquelles le noyau ferreux se déplace (voir Figure 2.17). Lorsqu'un courant alternatif est appliqué à la bobine principale, des tensions alternatives, v_1 et v_2, sont induites dans les deux bobines secondaires. Les sorties de ces bobines sont reliées de manière à obtenir la différence entre les deux tensions, c'est-à-dire $v_1 - v_2$. Lorsque le noyau se trouve au milieu des bobines secondaires, les deux tensions alternatives induites sont égales et la tension de sortie est donc nulle. Lorsque le noyau est déplacé de sa position centrale, une plus grande partie se trouve dans l'une des bobines secondaires. Par conséquent, la valeur de la tension alternative induite dans l'une des bobines est supérieure à celle induite dans l'autre. La différence entre les deux tensions, c'est-à-dire la sortie, dépend ainsi de la position du noyau ferreux. La sortie du LVDT est une tension alternative qui est généralement convertie en une tension continue analogique et amplifiée avant d'attaquer le canal analogique d'un API.

Les *capteurs de déplacement capacitifs* sont essentiellement des condensateurs plans. La capacité change en fonction de l'écartement des plaques, de la zone de chevauchement des plaques ou du déplacement d'un élément diélectrique entre les plaques (voir Figure 2.18). Toutes ces solutions peuvent être utilisées pour obtenir des capteurs de déplacement linéaire. Le changement de capacité doit être converti en un signal électrique adéquat par traitement du signal.

Copyright © 2015 Dunod.

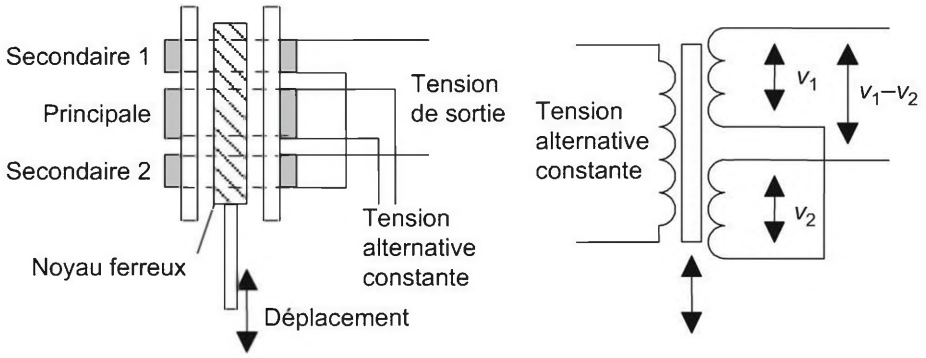


Figure 2.17 – Un transformateur différentiel à variation linéaire (LVDT).

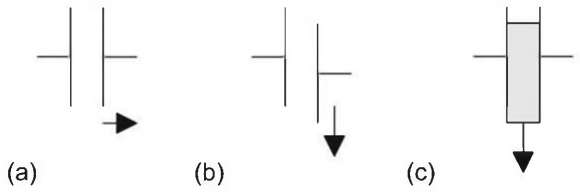


Figure 2.18 – Des capteurs capacitifs : (a) changer l'écartement des plaques, (b) changer la zone de chevauchements et (c) déplacer le diélectrique.

2.1.7 Extensomètres

Lorsqu'un fil ou une lame de semi-conducteur est étiré, sa résistance change. La modification fractionnaire de la résistance est proportionnelle à la modification fractionnaire de la longueur, c'est-à-dire de la déformation.

$$\frac{\Delta R}{R} = G \times \text{déformation}$$

ΔR correspond à la modification de la résistance d'un fil dont la résistance est R ; G est une constante appelée *coefficient de jauge*. Pour les métaux, le coefficient de jauge est environ de 2. Pour les semi-conducteurs, il vaut environ 100. Les extensomètres à résistance métallique prennent la forme d'une bobine plate afin d'obtenir une longueur de métal raisonnable sur une zone réduite. Ils sont souvent gravés à partir d'une feuille métallique (voir Figure 2.19a) fixée sur un film plastique mince de manière à pouvoir les coller sur des surfaces, à l'instar des timbres sur les enveloppes. La modification de la résistance de l'extensomètre, lorsqu'il est soumis à une déformation, est généralement convertie en une tension grâce à un pont de Wheatstone. Toutefois, la résistance de l'extensomètre change également avec la température et un système de compensation doit donc être ajouté pour que la sortie du pont soit une fonction de la seule déformation. Pour cela, il est possible de placer un extensomètre factice dans la branche opposée du pont, qui n'est pas sujet aux déformations seulement à la température (voir Figure 2.19b). Une autre solution classique se fonde sur quatre jauges actives placées dans les branches du pont. Un couple de jauges opposées est utilisé en extension, tandis

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

que l'autre couple l'est en compression. Cette méthode offre non seulement une compensation en température, mais permet une plus grande variation de la sortie lors d'une déformation. Le paragraphe suivant décrit des systèmes qui utilisent ce type de compensation.

En fixant des extensomètres sur d'autres dispositifs, les modifications provenant de la déformation de ces dispositifs peuvent être converties par les extensomètres en variations de tension. Par exemple, ils peuvent être fixés sur un porte-à-faux auquel des forces sont appliquées sur l'extrémité libre (voir Figure 2.19c). Les changements de tension, issus des extensomètres et du pont de Wheatstone, mesurent alors les forces exercées. Une autre possibilité consiste à fixer des extensomètres sur une membrane qui se déforme avec la pression (voir Figure 2.19d). La combinaison des extensomètres et du pont de Wheatstone permet de mesurer la pression appliquée.

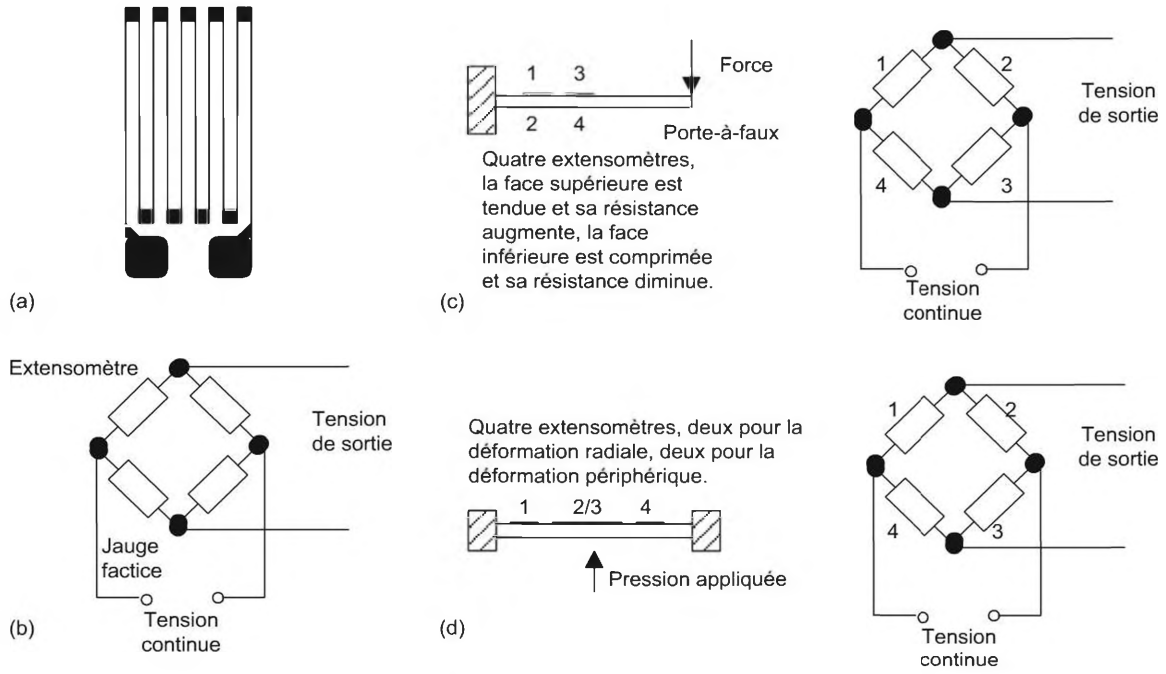


Figure 2.19 – (a) Un extensomètre à feuilles métalliques, (b) un pont de Wheatstone avec compensation des changements de température, (c) des extensomètres utilisés comme capteurs de force et (d) un capteur de pression.

2.1.8 Capteurs de pression

Les capteurs de pression peuvent être conçus pour générer des sorties proportionnelles à la *différence de pression* entre deux entrées. Si l'une des entrées reste ouverte, la jauge mesure la différence par rapport à la pression atmosphérique et la pression mesurée est appelée *pression manométrique*. Lorsque la mesure se fait par rapport au vide, il s'agit d'une *pression absolue*. Les capteurs de pression les plus utilisés sont les capteurs *à membrane* et *à soufflet*. Un capteur à membrane est constitué d'un disque mince de métal ou de plastique, fixé sur la périphérie. En cas de différence de pression entre les deux côtés de la membrane, son centre se

Copyright © 2015 Dunod.

déplace. La déviation est fonction de la différence de pression. Elle peut être détectée par des extensomètres fixés sur la membrane (voir Figure 2.19d), par un changement de capacité entre la membrane et une plaque positionnée parallèlement ou en utilisant la déviation pour appuyer sur un cristal piézo-électrique (voir Figure 2.20a).

Lorsqu'un cristal piézo-électrique est soumis à une pression, un déplacement relatif des charges positives et négatives se produit à l'intérieur du cristal et ses surfaces externes deviennent chargées. Une différence de potentiel apparaît alors. Le capteur MPX100AP de Motorola (Figure 2.20b) en est un exemple. Un vide d'air est créé sur l'un des côtés de la membrane et la variation de la membrane mesure la pression absolue appliquée sur l'autre côté. La sortie est une tension proportionnelle à la pression appliquée, avec une sensibilité de 0,6 mV/kPa. Il existe d'autres modèles dont un côté de la membrane est à l'air libre. Ils peuvent donc servir à mesurer une pression manométrique. Certains modèles permettent d'appliquer une pression sur les deux côtés de la membrane et donc de mesurer des pressions différentielles.

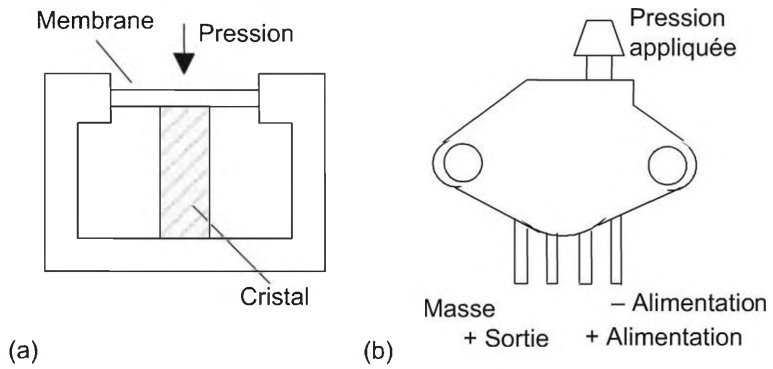


Figure 2.20 – (a) Un capteur de pression piézo-électrique et (b) le MPX100AP.

Certains capteurs de pression fonctionnent comme des interrupteurs tout ou rien pour une pression définie. Le modèle classique comprend une membrane ou un soufflet qui se déplace sous l'action de la pression et se comporte comme un interrupteur mécanique. La Figure 2.21 illustre deux possibilités. Les modèles à membrane sont moins sensibles que ceux à soufflet, mais ils acceptent des pressions plus élevées.

2.1.9 Détecteurs de niveau de liquide

Les capteurs de pression peuvent être utilisés pour surveiller le niveau de liquide dans un réservoir. La pression associée à la hauteur h du liquide au-dessus d'un certain niveau est égale à $h\rho g$, où ρ est la densité du liquide et g l'accélération liée à la pesanteur. Ainsi, une méthode couramment employée pour déterminer le niveau du liquide dans un réservoir consiste à mesurer la pression du liquide au-dessus d'un niveau donné (voir Figure 2.22).

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

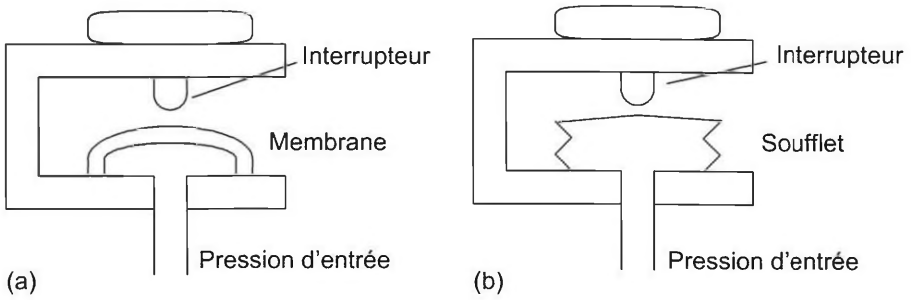


Figure 2.21 – Exemples d'interrupteurs à pression.

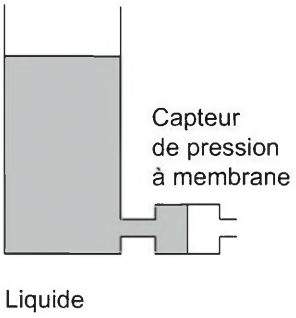


Figure 2.22 – Un capteur de niveau de liquide.

En général, le capteur doit simplement produire un signal lorsque le niveau dans le conteneur atteint un seuil fixé. Un interrupteur flottant utilisé dans ce but est constitué d'un interrupteur Reed et d'un flotteur, doté d'un aimant, qui se déplace dans un logement. Lorsque le flotteur monte ou descend, il ouvre ou ferme l'interrupteur Reed, qui est connecté à un circuit de commutation d'une tension.

2.1.10 Mesurer le débit d'un fluide

Pour mesurer le débit d'un fluide, on mesure souvent la différence de pression qui résulte du passage du fluide au travers d'un étranglement. La Figure 2.23 illustre la méthode la plus employée, le *débitmètre à diaphragme*. Lorsque le fluide passe au travers de l'orifice, la pression en A est plus élevée que celle en B. La différence de pression correspond à une mesure du débit. Elle peut être contrôlée par un manomètre à membrane afin de mesurer le débit.

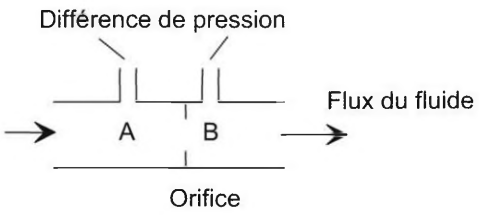


Figure 2.23 – Un débitmètre à diaphragme.

Copyright © 2015 Dunod.

2.1.11 Capteurs de proximité à ultrasons

Ces capteurs de proximité émettent des ondes ultrasonores (c'est-à-dire des ondes sonores à des fréquences supérieures aux fréquences audibles) en direction d'une cible et mesurent le temps nécessaire à leur retour. Plus l'objet est éloigné, plus le temps de retour des ondes est élevé. Ils sont utilisés pour des cibles situées à des distances allant de quelques centimètres à 1 m. Le modèle E4C-DS30 d'Omron est capable d'opérer dans la plage 50–300 mm. La version E4C-DS80 accepte une plage 70–800 mm, tandis que celle du E4C-DS100 est 90–1 000 mm. Dans l'idéal, l'objet visé doit avoir une surface plane et lisse afin d'obtenir une bonne réflexion. En effet, les surfaces incurvées ou irrégulières conduisent à une réflexion médiocre et le capteur doit alors être placé plus près des objets. Les matériaux mous, comme la mousse, réfléchissent mal les ondes sonores et ne seront pas détectés. Les applications des capteurs d'Omron sont nombreuses. Ils peuvent par exemple servir à détecter des bacs transparents, à inspecter les niveaux des réservoirs de solvants, à détecter le fléchissement d'une plaque, à détecter les pneus sur une bande transporteuse, à détecter l'affaissement de la bande de caoutchouc entre les rouleaux et à détecter l'emplacement des substrats en verre dans des cassettes.

2.1.12 Capteurs intelligents

Pour utiliser un capteur, il faut généralement ajouter un circuit de traitement du signal, qui, par exemple, l'amplifie et effectue une conversion analogique-numérique, de manière à obtenir un signal sous la forme appropriée, à tenir compte des non-linéarités et à le calibrer. Par ailleurs, nous devons également prendre en compte la dérive, c'est-à-dire la modification progressive des propriétés du capteur au fil du temps. Certains capteurs réunissent tous ces éléments dans un même boîtier : les *capteurs intelligents*.

Le terme capteur intelligent désigne donc un capteur qui réunit les circuits de tampon et de traitement nécessaires dans un même élément et qui offre des fonctions qui dépassent celles d'un capteur simple. Ces capteurs comprennent généralement des convertisseurs de données, un processeur et un micrologiciel, ainsi qu'une forme de mémoire morte effaçable électriquement et programmable (EEPROM, *Electrically-Erasable Programmable Read-Only Memory*) semblable à une EPROM. Le qualificatif *morte* est utilisé car la mémoire doit conserver certains paramètres même lorsque l'alimentation est coupée. Tous les éléments de ces capteurs intelligents sont placés sur une même puce de silicium. Puisqu'un tel capteur se fonde sur un processeur, il peut être programmé en fonction des besoins particuliers. Par exemple, il peut être programmé pour traiter des données d'entrée brutes, pour corriger certains aspects comme les non-linéarités et pour renvoyer ensuite les données traitées à une station de base. Il peut être programmé pour émettre un signal d'avertissement lorsque le paramètre mesuré atteint une valeur critique.

La norme IEEE 1451.4, qui concerne l'interface des capteurs et des actionneurs intelligents, se fonde sur un format de fiche technique électronique (TEDS) dont l'objectif est de faciliter la connexion des transducteurs analogiques existants à des

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

systèmes de mesure numériques. La norme impose qu'une EEPROM embarquée détienne et communique des données qui permettent d'offrir des possibilités plug-and-play. Elle contient les données d'identification du capteur, ainsi que ses caractéristiques. Elle peut également proposer un modèle de calibrage pour faciliter les interrogations numériques.

2.1.13 Gammes de capteurs

Pour vous donner une idée de la diversité des capteurs utilisés dans les systèmes de commande, voici une partie de la liste exhaustive des capteurs proposés par Rockwell Automation pour de telles applications :

- Les *capteurs d'état* communiquent des informations qui permettent le séquençage automatique d'un matériel. Dans cette catégorie, nous trouvons les capteurs de pression, les sondes de température, les capteurs de niveau, les interrupteurs de débit et les interrupteurs détecteurs de vitesse.
- Les *dispositifs de détection de présence* détectent la distance, l'absence ou la présence d'un objet. Ils s'agit de détecteurs de proximité capacitifs, de détecteurs de proximité inductifs, de détecteurs à ultrasons et de cellules photoélectriques. Ces dernières peuvent être employées pour la détection de matériaux transparents, comme les bouteilles en verre ou en plastique, et, en utilisant des filtres, un changement de couleur. Les capteurs à fibre optique, équipés d'un capteur lumineux à une extrémité du câble, peuvent être utilisés dans des applications de conditionnement qui imposent la détection d'objets de petite taille.
- Les *interrupteurs de fin de course* sont des dispositifs électromécaniques composés d'un actionneur couplé à un jeu de contacts. Lorsqu'un objet entre en contact avec l'actionneur, le dispositif actionne les contacts de façon à établir ou à interrompre une connexion électrique.
- Les *interrupteurs de sécurité* sont un moyen de protection d'une installation en coupant l'alimentation, en contrôlant l'accès du personnel et en empêchant le démarrage d'une machine lorsque la grille est ouverte.

2.2 Dispositifs de sorties

Les ports de sortie d'un API sont équipés de relais ou de photocoupleurs avec transistor ou triac, selon les dispositifs qui doivent être commandés. En général, le signal numérique issu d'un canal de sortie d'un API est utilisé pour commander un actionneur, qui à son tour contrôle un processus. Le terme *actionneur* désigne l'organe qui transforme le signal électrique en une action plus puissante, qui conduit ensuite au contrôle du processus. Les sections suivantes en présentent quelques exemples.

2.2.1 Relais

Un courant qui traverse un solénoïde produit un champ magnétique qui peut attirer des composants métalliques ferreux. Dans le cas du *relais*, cette attraction est

Copyright © 2015 Dunod.

utilisée pour actionner un interrupteur. Les relais peuvent donc servir à contrôler des courants ou des tensions de niveau plus élevé et, de plus, à isoler le courant de déclenchement de l'action de celui du contrôle. Pour un relais connecté à la sortie d'un API, l'activation de la sortie déclenche la production du champ magnétique dans le solénoïde, ce qui attire les contacts et ferme un ou plusieurs interrupteurs (voir Figure 2.24). De cette manière, des courants continus plus importants peuvent être commutés. Le relais est ainsi capable de mettre un moteur sous tension. Le solénoïde d'un relais peut être utilisé pour actionner plusieurs ensembles de contacts, chaque ensemble étant appelé *pôle*. Les contacts peuvent également être de type normalement ouverts (NO) ou normalement fermés (NF) – leur état en l'absence d'une entrée. Lors du choix d'un relais, vous devez donc décider du nombre de pôles nécessaires, des conditions de contact initiales, ainsi que de la tension et du courant nominaux.

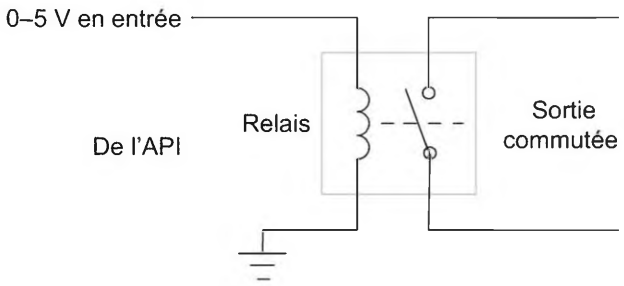


Figure 2.24 – Un relais utilisé en dispositif de sortie.

Dans un *relais à verrouillage*, les contacts restent ouverts ou fermés même après avoir coupé l'alimentation du solénoïde. Le terme *contacteur* est utilisé lorsque des courants de haute intensité sont commutés à partir de tensions élevées.

2.2.2 Vannes de commande directionnelle

Les solénoïdes sont aussi utilisés comme actionneurs dans les *vannes solénoïdes*. La vanne peut servir à contrôler le sens d'un flux d'air ou d'huile sous pression ou bien pour actionner d'autres dispositifs, comme un piston qui se déplace dans un cylindre. La Figure 2.25 illustre un tel système, une *vanne à tiroir*, qui permet de contrôler le mouvement d'un piston dans un cylindre. L'air sous pression ou le fluide hydraulique entre par l'orifice P, qui est relié à une pompe ou un compresseur, et l'orifice T est branché de manière à renvoyer le fluide hydraulique dans le réservoir d'alimentation ou, dans le cas d'un système pneumatique, pour laisser l'air s'échapper à l'exétrieur. Lorsque le solénoïde n'est pas alimenté (voir Figure 2.25a), le fluide ou l'air sous pression est envoyé sur la droite du piston et s'échappe par la gauche, ce qui conduit au déplacement du piston vers la gauche. Lorsque le solénoïde est alimenté, la vanne à tiroir bascule le fluide ou l'air sous pression vers la gauche du piston, avec un échappement vers la droite. Le piston se déplace alors vers la droite. Le mouvement du piston peut servir à pousser un déflecteur qui retirera des articles d'une bande transporteuse (voir Figure 1.1b) ou pour mettre en œuvre une autre forme de déplacement qui nécessite une certaine puissance.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

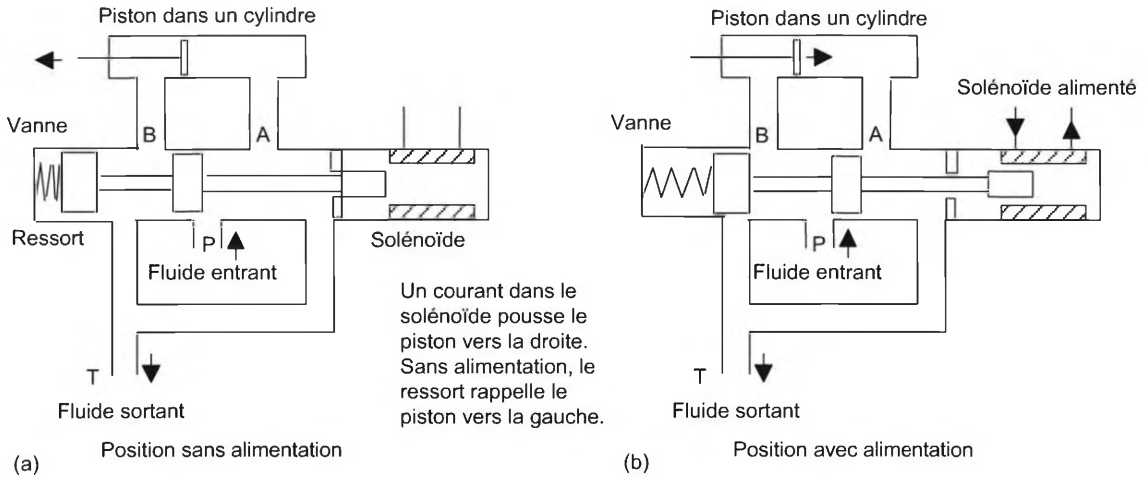


Figure 2.25 – Un exemple de vanne solénoïde.

Avec le modèle de vanne précédent, les deux positions de commande sont illustrées aux Figures 2.25a et 2.25b. Les vannes de commande directionnelle sont définies par leur nombre d'orifices et leur nombre de positions de commande. La vanne illustrée à la Figure 2.25 dispose de quatre orifices (A, B, P et T) et de deux positions de commande. Il s'agit donc d'une *vanne 4/2*. Pour dessiner des vannes, le symbole de base est un carré. Chaque carré décrit une position de commande. Par conséquent, le symbole de la vanne présentée à la Figure 2.25 comprend deux carrés (voir Figure 2.26a). Dans chaque carré, les positions sont ensuite décrites par des flèches pour indiquer le sens du flux ; une ligne terminée indique l'absence de chemin de flux (voir Figure 2.26b).

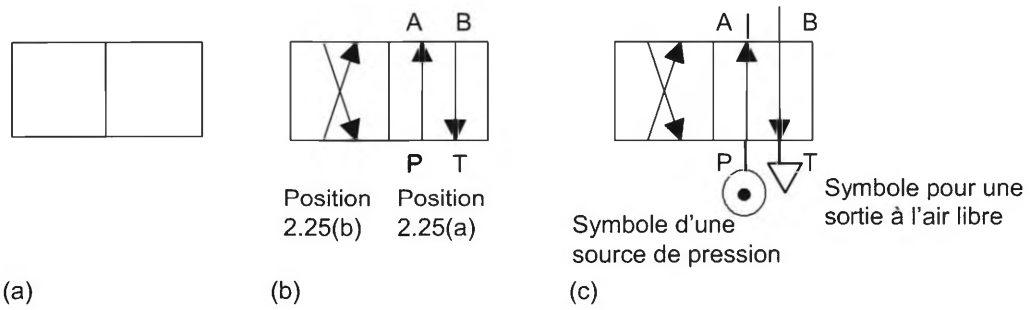
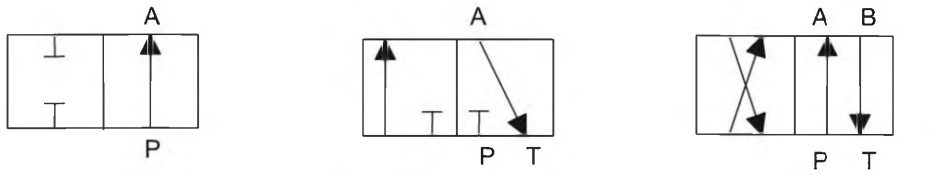


Figure 2.26 – (a) Le symbole de base d'une vanne à deux positions, (b) de la vanne 4/2 et (c) des connexions externes à la vanne 4/2. Le libellé P indique une connexion à une source de pression, tandis que T désigne un orifice d'échappement.

Les connexions des tuyaux d'une vanne, c'est-à-dire les orifices d'entrée et de sortie, sont représentées par des lignes tracées à l'extérieur de la boîte, uniquement sur celle qui représente la position inactive de la vanne. La Figure 2.26c montre ces connexions pour la vanne de la Figure 2.25. La Figure 2.27 présente d'autres exemples de vannes de commande directionnelle et leurs positions de commande.

Copyright © 2015 Dunod.



Vanne 2/2 : flux de P vers A, activer pour arrêt du flux.

Vanne 3/2 : pas de flux de P vers A et flux de A vers T, activer pour T fermé et flux de P vers A.

Vanne 4/2 : flux initial de P vers A et de B vers T, activer pour flux de P vers B et de A vers T.

Figure 2.27 – Des vannes de commande directionnelle.

Dans les schémas, les méthodes d'actionnement des vannes sont ajoutées au symbole. La Figure 2.28 en montre quelques exemples. La vanne de la Figure 2.25 est équipée d'un ressort pour activer une position et d'un solénoïde pour l'autre ; le symbole correspondant est donné à la Figure 2.28d.

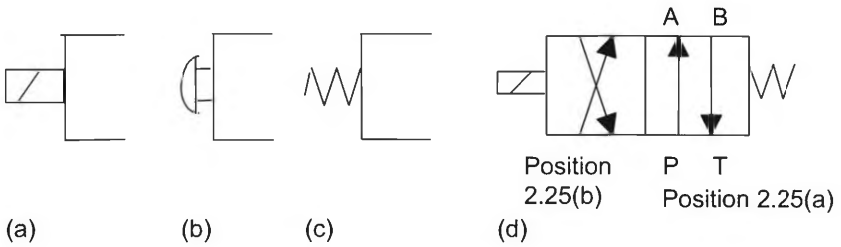


Figure 2.28 – Symboles d'actionnement : (a) à solénoïde, (b) à bouton-poussoir, (c) à ressort, (d) une vanne 4/2 à ressort et solénoïde.

Les vannes de commande directionnelle permettent de contrôler le sens de déplacement des pistons dans des cylindres, le mouvement des pistons étant utilisé pour mettre en œuvre les actions nécessaires. Dans un *vérin à simple effet* (voir Figure 2.29a), le fluide sous pression est appliqué sur un côté du piston de manière à créer un déplacement dans une direction. Le retour dans la direction opposée peut être assuré par un ressort interne. Dans un *vérin à double effet* (voir Figure 2.29b), le fluide sous pression permet le déplacement du piston dans les deux directions. La Figure 2.30 montre comment utiliser une vanne pour commander le sens de déplacement d'un piston dans un vérin à simple effet. La Figure 2.31 illustre l'utilisation de deux vannes pour commander un piston dans un vérin à double effet.

2.2.3 Moteurs

Un moteur à courant continu est constitué de bobines de fil montées dans des encoches sur un cylindre en matériau ferromagnétique, appelé *induit*. L'induit est monté sur des paliers et peut tourner. Il se trouve dans un champ magnétique produit par des aimants permanents ou par un courant traversant des *bobines d'induction*. Lorsqu'un courant traverse les bobines de l'induit, des forces agissent sur l'induit et déclenchent sa rotation. Des balais et un commutateur permettent

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

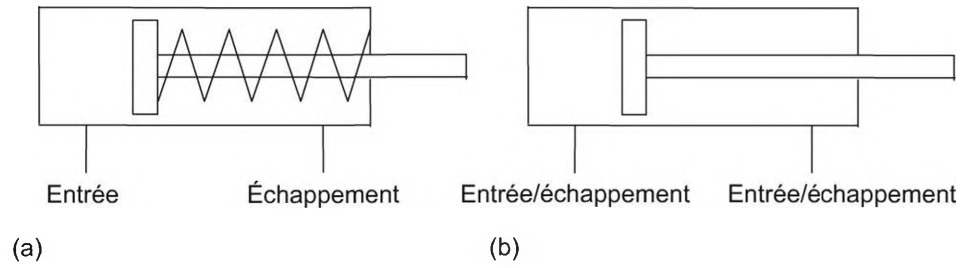


Figure 2.29 – Vérins (a) à simple effet et (b) à double effet.

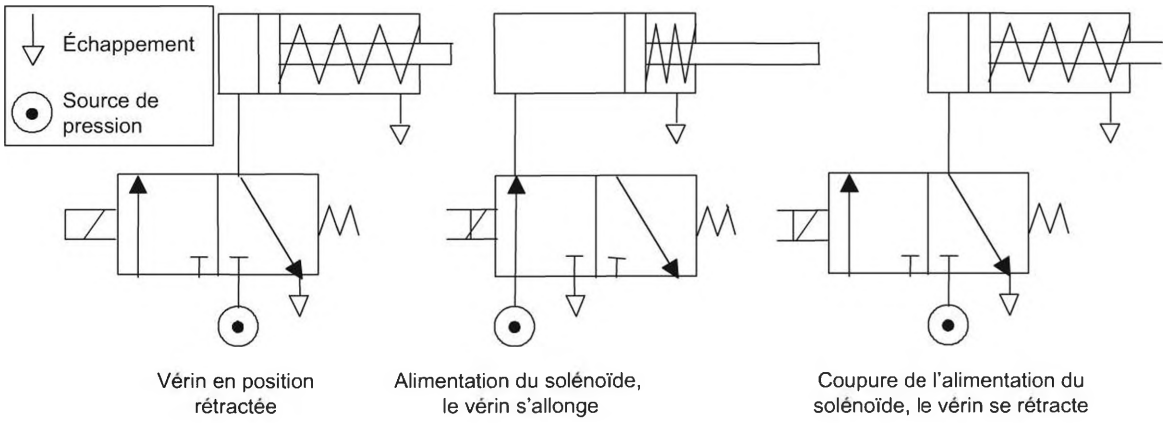


Figure 2.30 – Commande d'un vérin à simple effet.

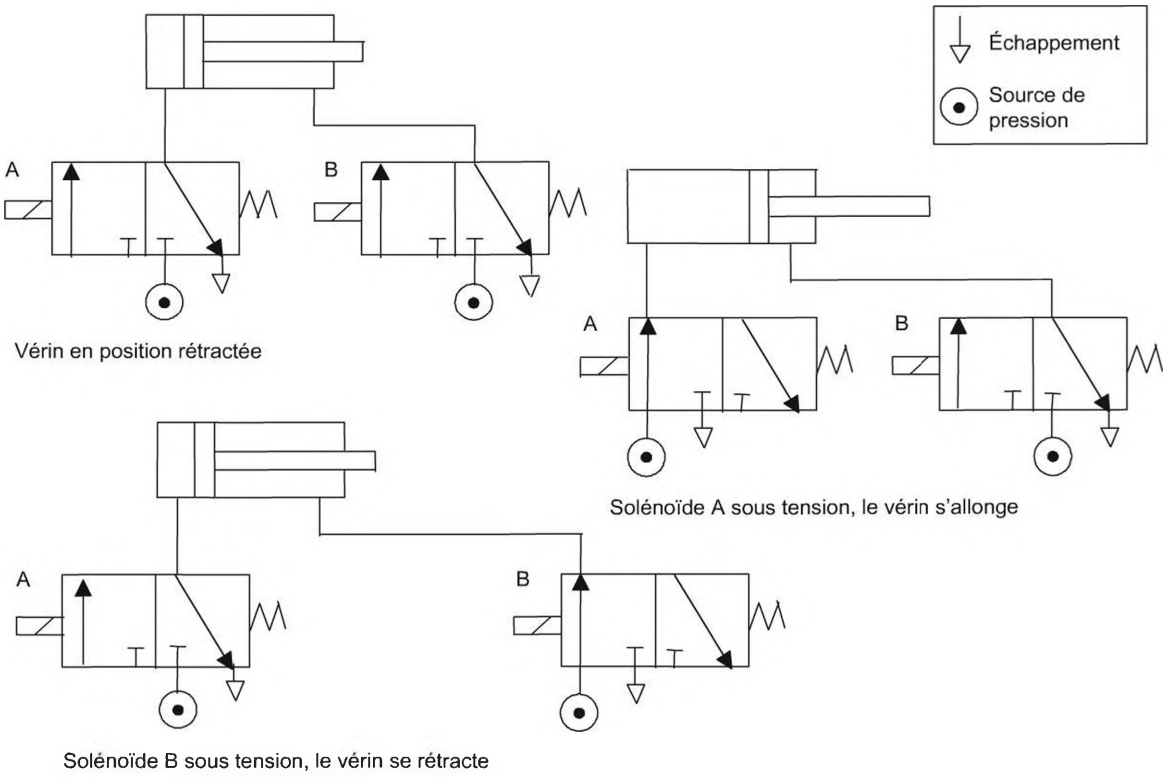


Figure 2.31 – Commande d'un vérin à double effet.

Copyright © 2015 Dunod.

d'inverser le courant dans les bobines à chaque demi rotation afin de maintenir la rotation de l'induit. La vitesse de rotation peut être modifiée en variant le courant dans les bobines de l'induit. Cependant, puisque les bobines sont généralement alimentées par des tensions fixes, le courant variable nécessaire est souvent obtenu par un circuit électronique. Il permet de contrôler la valeur moyenne de la tension, et donc du courant, en faisant varier la durée pendant laquelle la tension continue est fournie (voir Figure 2.32). Le terme *modulation de largeur d'impulsion* (MLI) désigne cette méthode car la largeur des impulsions de tension permet de contrôler la tension continue moyenne appliquée à l'induit. Un API peut donc réguler la vitesse de rotation d'un moteur en contrôlant le circuit électronique qui fixe la durée des impulsions.

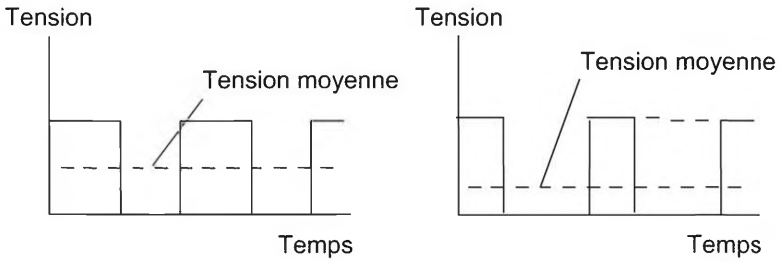


Figure 2.32 – Modulation de largeur d'impulsion.

De nombreux processus industriels demandent simplement à l'API de démarrer ou d'arrêter un moteur à courant continu. Pour cela, il est possible d'employer un relais. La Figure 2.33a illustre le principe de base. La diode sert à dissiper le courant induit par la force contre-électromotrice.

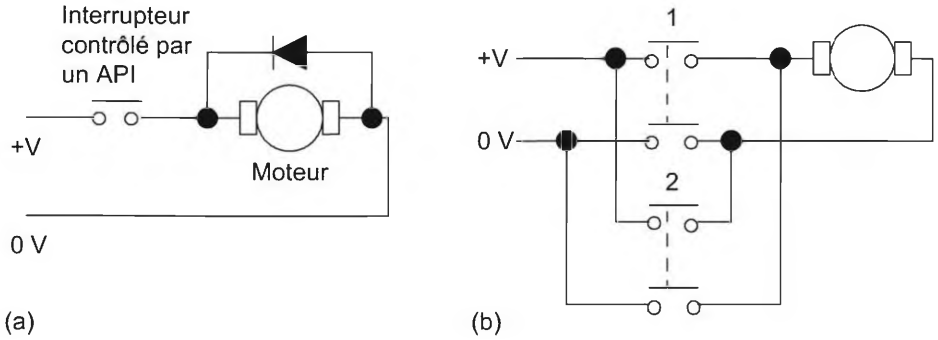


Figure 2.33 – Moteur à courant continu : (a) une commande tout ou rien et (b) une commande directionnelle.

L'API doit parfois inverser le sens de rotation du moteur. Pour cela, il est possible d'utiliser des relais qui inversent le sens du courant appliqué aux bobines de l'induit. La Figure 2.33b illustre le principe de base. Pour la rotation dans un sens, l'interrupteur 1 est fermé et l'interrupteur 2 est ouvert. Pour la rotation dans l'autre sens, l'interrupteur 1 est ouvert et l'interrupteur 2 est fermé.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

Il existe également des *moteurs à courant continu sans balais*. Ils utilisent des aimants permanents pour obtenir le champ magnétique, mais à la place de la rotation de l'induit provoquée par le champ magnétique de l'aimant, l'aimant permanent tourne dans la bobine fixe. Avec un moteur à courant continu classique, il est nécessaire d'utiliser un commutateur pour inverser le courant dans les bobines à chaque demi rotation de manière à poursuivre la rotation de l'induit dans le même sens. Avec les moteurs sans balais à aimants permanents, un circuit électronique permet d'inverser le courant. Le moteur peut être démarré et arrêté en contrôlant le courant dans la bobine fixe. L'inversion du sens de rotation du moteur est plus complexe, car inverser le courant n'est pas si simple, en raison du circuit électronique qui met en œuvre le commutateur. Une méthode consiste à incorporer des capteurs pour détecter l'emplacement des pôles nord et sud. Ces capteurs peuvent ensuite provoquer la commutation du courant dans la bobine au bon moment afin d'inverser les forces appliquées à l'aimant. La vitesse de rotation peut être commandée par une modulation de largeur d'impulsion, c'est-à-dire par contrôle de la valeur moyenne des impulsions de tension continue constante.

Si les moteurs à courant alternatif sont moins coûteux, plus solides et plus fiables que les moteurs à courant continu, le maintien d'une vitesse constante et le contrôle de cette vitesse sont généralement plus complexes. Par conséquent, les moteurs à courant continu, en particulier les moteurs sans balais à aimants permanents, sont les plus utilisés dans les processus de commande.

2.2.4 Moteurs pas à pas

Un *moteur pas à pas* produit une rotation par l'intermédiaire d'un mouvement angulaire, les *pas*, pour chaque impulsion numérique envoyée en entrée (voir Figure 2.34). Si une impulsion d'entrée produit une rotation de 1,8°, alors, vingt impulsions de ce type produisent une rotation de 36,0°. Pour obtenir une révolution complète de 360°, deux cents impulsions sont nécessaires. Le moteur peut donc être utilisé pour un positionnement angulaire précis.

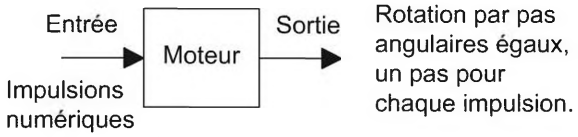


Figure 2.34 – Le moteur pas à pas.

Lorsqu'un moteur pas à pas est utilisé pour mettre en mouvement une courroie sans fin (voir Figure 2.35), il permet d'obtenir un positionnement linéaire précis. Ces moteurs sont employés dans les imprimantes d'ordinateurs, les robots, les machines-outils et divers instruments qui nécessitent un positionnement précis.

Il existe deux types de moteurs pas à pas de base : à *aimants permanents*, avec un rotor à aimants permanents, et à *réluctance variable*, avec un rotor de fer doux. Il existe également un type *hybride* qui combine ces deux technologies. Les modèles les plus courants sont ceux à aimants permanents.

Copyright © 2015 Dunod.

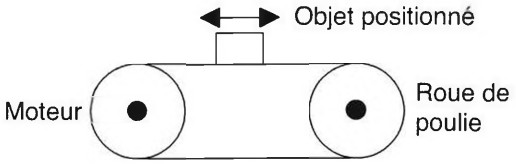


Figure 2.35 – Un positionnement linéaire.

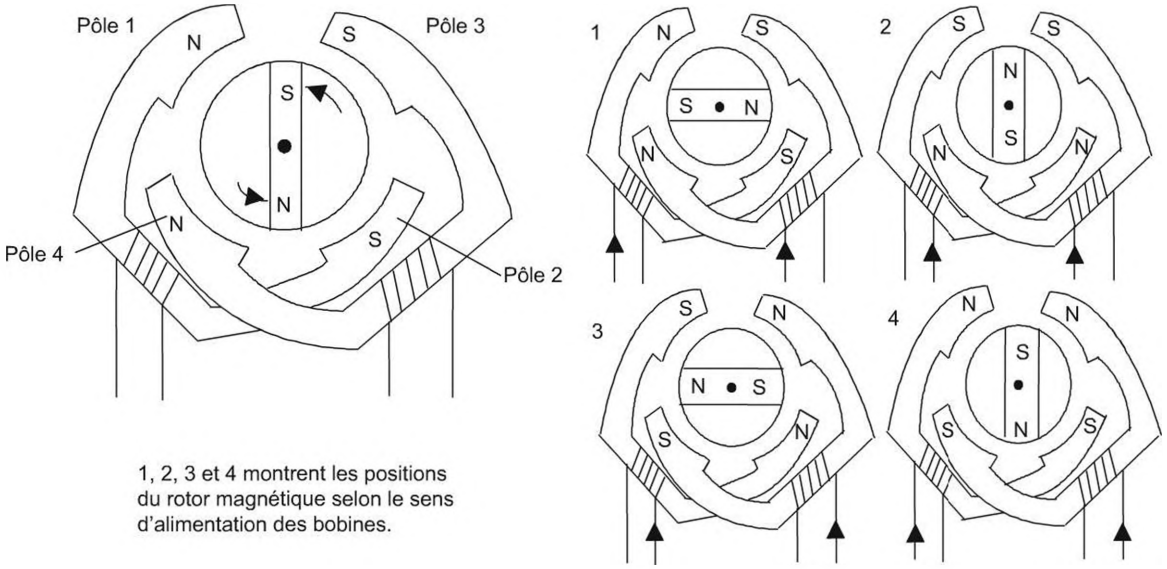


Figure 2.36 – Le principe de base d'un moteur pas à pas à aimants permanents (deux phases) avec des pas de 90°.

La Figure 2.36 présente les éléments de base d'un moteur à aimants permanents avec un stator à deux pôles. Chaque pôle est activé par un courant qui traverse l'enroulement inducteur approprié, les bobines étant disposées de manière à ce que des pôles opposés soient générés sur des bobines opposées. Une source d'alimentation continue fournit le courant aux enroulements par l'intermédiaire d'interrupteurs. Lorsque les courants dans les bobines sont tels que les pôles correspondent à ceux illustrés à la Figure 2.36, le rotor tourne de manière à s'aligner avec le couple de pôles suivants et s'arrête dans cette position. Il s'agit d'une rotation de 90°. Si le courant est ensuite commuté de manière à inverser les polarités, le rotor avance d'un pas pour s'aligner avec le couple de pôles suivant, à l'angle 180°, et s'arrête. Voici les polarités associées à chaque pas :

Pas	Pôle 1	Pôle 2	Pôle 3	Pôle 4
1	Nord	Sud	Sud	Nord
2	Sud	Nord	Sud	Nord
3	Sud	Nord	Nord	Sud
4	Nord	Sud	Nord	Sud
5	Répétition des phases 1 à 4			

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

Dans ce cas, le rotor peut prendre quatre positions : 0°, 90°, 180° et 270°.

La Figure 2.37 illustre le principe de base du moteur à réluctance variable. Le rotor est en fer souple et possède des dents, dont le nombre est inférieur au nombre de pôles sur le stator. Le stator est composé de couples de pôles, chaque couple est activé et transformé en un électroaimant par un courant qui traverse la bobine enroulée autour. Lorsqu'un couple de pôles est activé, le champ magnétique qui se produit attire le couple de dents du rotor le plus proche afin que les dents et les pôles s'alignent. Il s'agit de la position de *réluctance minimale*. En commutant séquentiellement le courant d'un couple de pôles au suivant, le rotor tourne pas à pas.

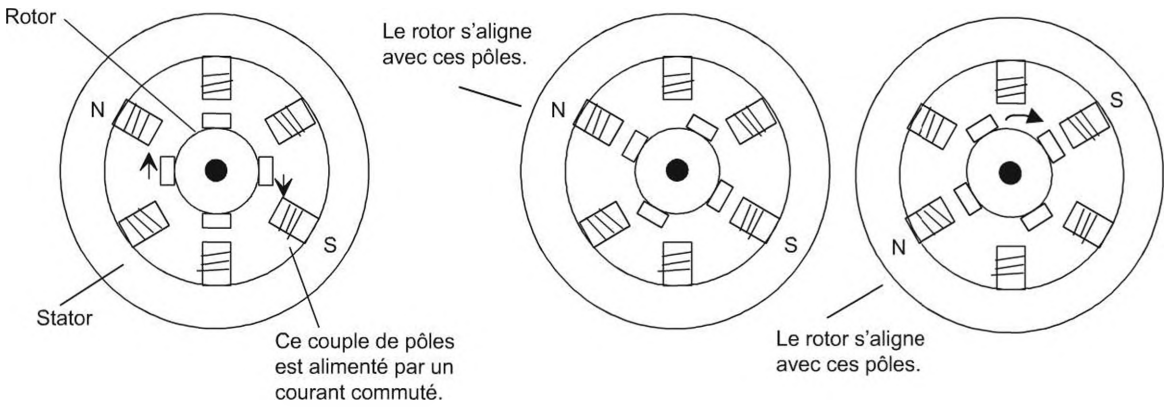


Figure 2.37 – Le principe du moteur pas à pas à reluctance variable.

Il existe également une version hybride du moteur pas à pas. Elle combine les caractéristiques des moteurs à aimants permanents et celles des moteurs à réluctance variable. Les moteurs pas à pas hybrides disposent d'un rotor à aimants permanents pris entre des disques dentés. Le rotor se place lui-même en position de réluctance minimale lorsqu'un couple de bobines de stator est alimenté.

Voici les termes fréquemment employés pour définir les moteurs pas à pas :

- *Phase*. Ce terme désigne le nombre d'enroulements indépendants sur le stator. Les moteurs à deux phases sont généralement employés dans les applications légères, les moteurs à trois phases sont généralement à réluctance variable et les moteurs à quatre phases sont utilisés dans les applications lourdes.
- *Angle de pas*. Il s'agit de l'angle de rotation du rotor à chaque impulsion envoyée sur les bobines du stator.
- *Couple statique*. Cette caractéristique correspond au couple maximal qui peut être appliqué à un moteur alimenté sans le faire quitter sa position de repos et provoquer une rotation du moyeu.
- *Couple d'accrochage*. Cette valeur correspond au couple maximal avec lequel un moteur démarrera, pour une fréquence de pas donnée, et se synchronisera sans perte de pas.
- *Couple de décrochage*. Il s'agit du couple maximal qui peut être appliqué à un moteur, fonctionnant à une fréquence de pas donnée, sans se désynchroniser.

Copyright © 2015 Dunod.

- *Taux d'accrochage*. Cette caractéristique correspond au taux de commutation maximal à laquelle un moteur chargé peut démarrer sans perte de pas.
- *Taux de décrochage*. Cette valeur indique la fréquence de commutation à laquelle un moteur chargé restera synchronisé lorsque cette fréquence est abaissée.
- *Plage de rotation*. Il s'agit de la plage de fréquence de pas entre l'accrochage et le décrochage dans laquelle le moteur reste synchronisé mais ne peut pas démarrer ou tourner en sens inverse

Pour que ce type de moteur puisse tourner pas à pas, il faut que chaque couple de bobines du stator soit alimenté dans un ordre précis avec une entrée de type train d'impulsions (voir Figure 2.38). Il existe des circuits de pilotage qui génèrent le séquencement approprié.

La Figure 2.39 illustre l'emploi du circuit SAA 1027 pour un moteur pas à pas unipolaire à quatre phases. Les moteurs sont dits *unipolaires* lorsque leur câblage permet au courant de circuler dans une seule direction au travers d'une borne du moteur ; ils sont dits *bipolaires* si le courant peut circuler dans un sens ou dans l'autre au travers d'une borne du moteur. Le moteur pas à pas pivote d'un pas chaque fois que l'entrée Trigger passe du niveau bas au niveau haut. Il tourne dans le sens des aiguilles d'une montre lorsque l'entrée Rotation est au niveau bas et dans le sens contraire lorsqu'elle est au niveau haut. Lorsque la broche Set est au niveau bas, les sorties sont réinitialisées. Dans un automatisme, ces impulsions d'entrée peuvent être fournies par un microprocesseur.

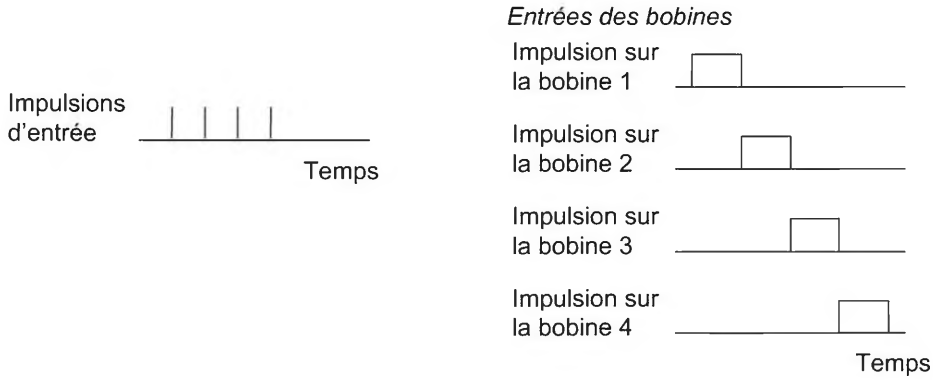


Figure 2.38 – Entrée et sortie du système de pilotage.

2.3 Exemples d'applications

Les sections suivantes décrivent quelques exemples d'automatismes qui illustrent l'utilisation de différents dispositifs d'entrées et de sorties.

2.3.1 Bande transporteuse

Prenons une bande transporteuse utilisée pour convoyer des marchandises depuis une chargeuse mécanique vers un atelier de conditionnement (voir Figure 2.40). Lorsqu'un article est placé sur la bande transporteuse, un interrupteur de contact

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

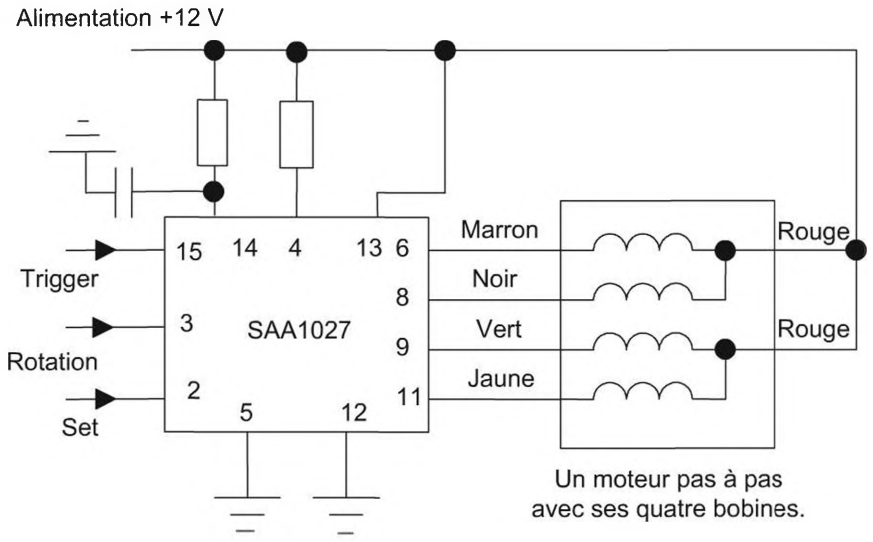


Figure 2.39 – Schéma d'un système de pilotage fondé sur le circuit SAA 1027.

signale sa présence et démarre le moteur du transporteur. Le moteur doit rester en action jusqu'à ce que l'article atteigne l'extrémité du transporteur et arrive dans l'atelier de conditionnement. À ce moment-là, un interrupteur est activé afin de stopper le moteur, qui reste à l'arrêt jusqu'à ce qu'un nouvel article soit chargé sur la bande transporteuse. Les entrées de l'API qui contrôle le transporteur comprennent les deux interrupteurs et la sortie, le moteur.

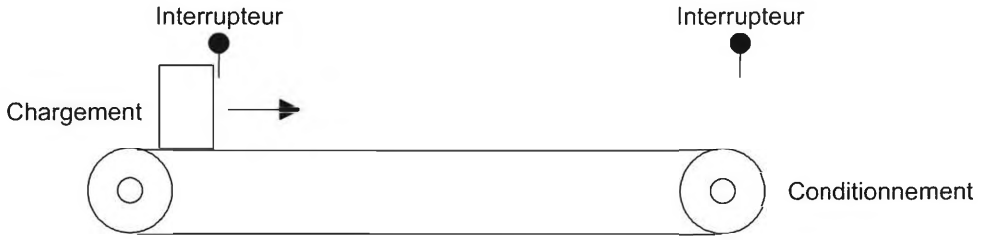


Figure 2.40 – Une bande transporteuse.

2.3.2 Monte-charge

Prenons un simple monte-charge qui sert à déplacer des articles d'un niveau à un autre. Il pourrait, par exemple, être utilisé pour monter des parpaings du rez-de-chaussée à la hauteur de travail des maçons. Le monte-charge se déplace vers le haut suite à l'appui sur un bouton-poussoir situé au rez-de-chaussée ou à un niveau supérieur pour aller plus haut. Dans les deux cas, une condition doit être satisfaite : la porte d'accès au monte-charge doit être fermée, comme indiqué par un interrupteur de fin de course. Le monte-charge se déplace vers le bas par un appui sur un bouton-poussoir situé au niveau supérieur ou inférieur. Dans les deux cas, l'interrupteur de fin de course doit indiquer que la porte d'accès est fermée. Dans cet exemple, les entrées de l'automate sont des signaux électriques tout ou rien provenant des boutons-poussoirs et des interrupteurs de fin de course. La sortie est le signal de déclenchement du moteur.

Copyright © 2015 Dunod.

2.3.3 Système de commande de robot

La Figure 2.41 illustre l'utilisation de vannes de commande directionnelle par un système de commande de robot. Lorsque l'entrée du solénoïde A de la vanne 1 est activée, le piston se déplace vers la droite et provoque la fermeture de la pince. Si le solénoïde B est sous tension alors que le solénoïde A n'est pas excité, le piston se déplace vers la gauche et la pince s'ouvre. Lorsque aucun des solénoïdes n'est sous tension, l'air n'arrive dans aucune des chambres du vérin et le piston conserve sa position. De la même manière, les entrées des solénoïdes de la vanne 2 permettent d'étendre ou de rétracter le bras. Les entrées des solénoïdes de la vanne 3 déplacent le bras vers le haut ou vers le bas. Celles des solénoïdes de la vanne 4 permettent de pivoter la base dans le sens horaire ou anti-horaire.

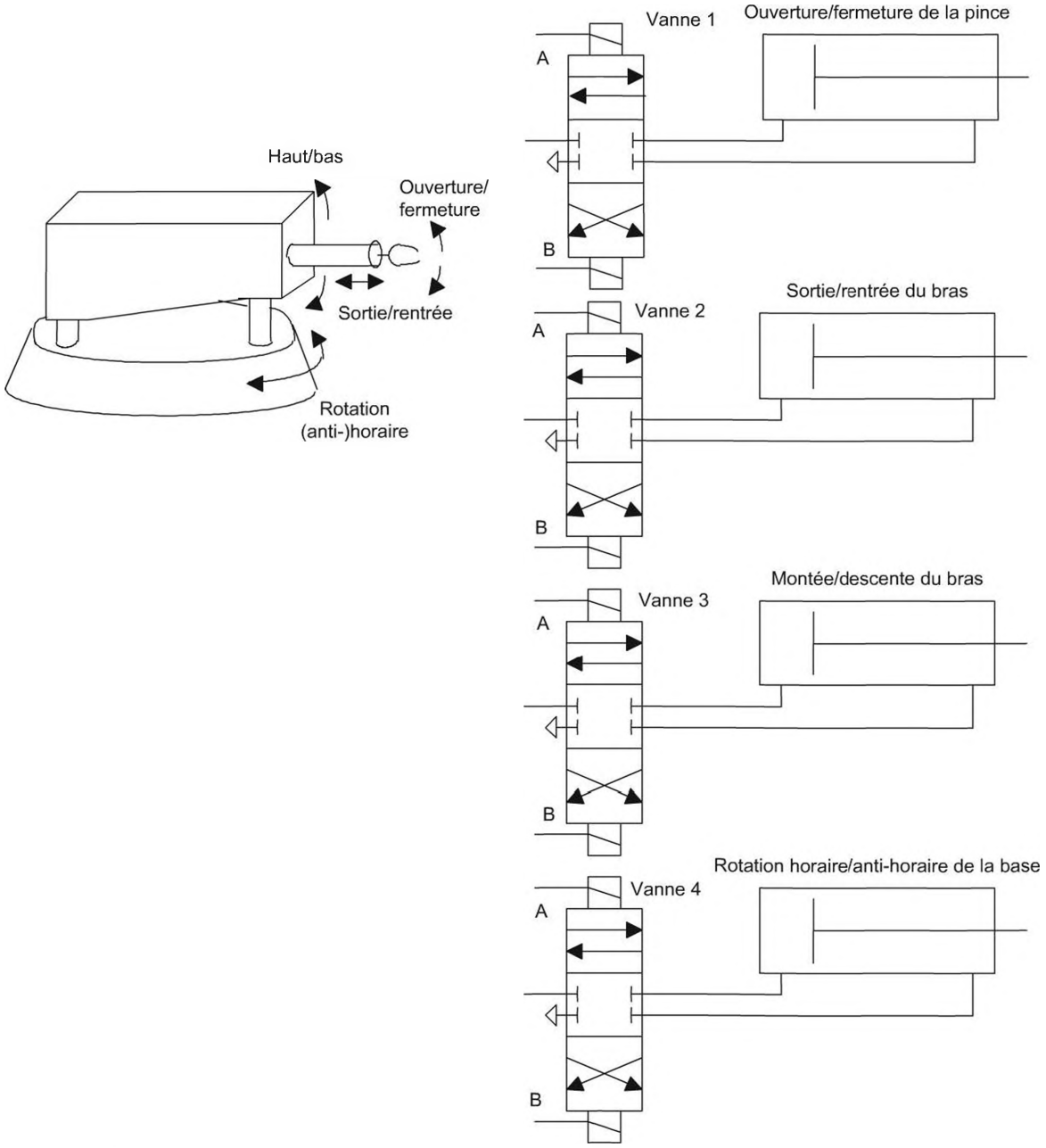


Figure 2.41 – Les commandes d'un robot.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

2.3.4 Surveillance du niveau d'un liquide

La Figure 2.42 illustre une méthode permettant d'obtenir un signal tout ou rien lorsqu'un liquide atteint un niveau critique dans un conteneur. Un flotteur magnétique – un anneau placé autour de la sonde – se déplace vers le bas lorsque le niveau du liquide baisse. Il ouvre un interrupteur Reed lorsque le niveau critique est atteint. L'interrupteur Reed est placé en série avec une résistance de 39 Ω afin qu'elle soit connectée en parallèle avec une résistance de 1 kΩ lorsque l'interrupteur est fermé. L'ouverture de l'interrupteur Reed fait donc passer la résistance du circuit de 37 Ω, environ, à 1 kΩ. Par simple traitement du signal, ce changement de résistance peut être transformé en un signal tout ou rien adapté.

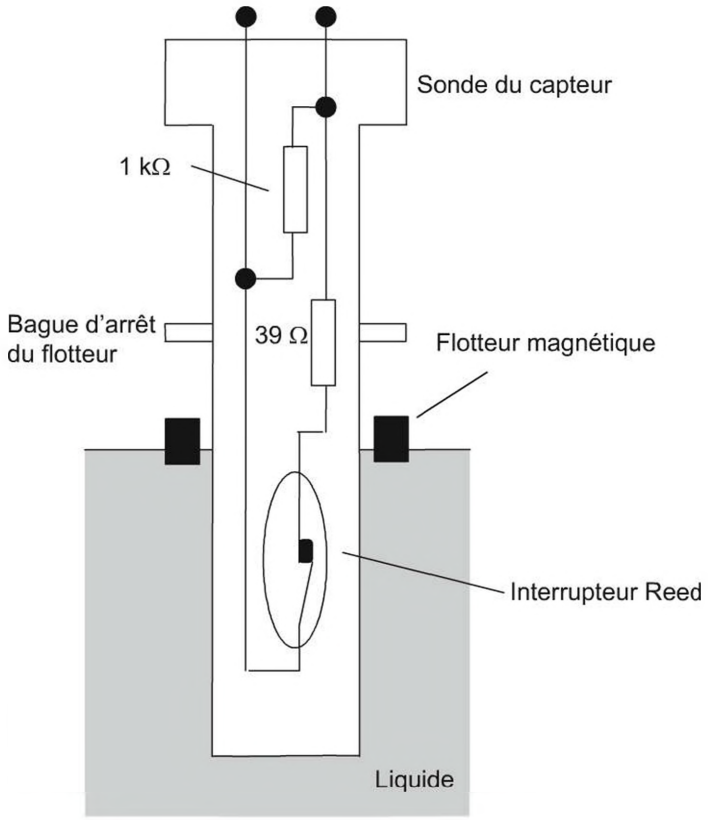


Figure 2.42 – Surveillance du niveau d'un liquide.

2.3.5 Colis sur une bande transporteuse

Dans certains cas, il faut vérifier si un article non transparent se trouve à un certain emplacement sur une bande transporteuse. Pour cela, il suffit d'utiliser un émetteur lumineux d'un côté de la bande et un capteur photoélectrique de l'autre côté. Lorsque le faisceau lumineux est interrompu, cela signifie qu'un colis se trouve à l'emplacement correspondant. Si l'article est transparent, par exemple une bouteille, le capteur photoélectrique doit être positionné de manière à être stimulé par la lumière réfléchie lorsque l'article se trouve à l'emplacement surveillé.

Copyright © 2015 Dunod.

2.4 En résumé

Un *capteur* est un dispositif d'entrée qui fournit une sortie utilisable en réponse à une entrée précisée. Un *transducteur* est généralement utilisé comme dispositif de conversion d'un signal d'une forme physique en une forme physique différente.

Les termes suivants sont généralement employés pour spécifier les performances des capteurs. La *précision* correspond à l'erreur maximale sur la valeur indiquée par le système ou l'élément de mesure. L'*erreur* est la différence entre le résultat de la mesure et la valeur réelle. Le terme *erreur de non-linéarité* décrit une erreur due à l'hypothèse d'une relation linéaire entre l'entrée et la sortie. Le terme *erreur d'hystérésis* est employé dans le cas d'une différence sur la valeur d'une même quantité mesurée selon que cette valeur a été atteinte par une augmentation ou une diminution progressive. La *plage* correspond aux limites de variation d'une entrée. Le *temps de réponse* correspond au temps écoulé après que l'entrée a brutalement augmenté de zéro à une valeur constante, jusqu'au point où elle atteint un pourcentage donné de la valeur d'état stationnaire. La *sensibilité* indique la variation de la sortie lorsque la grandeur mesurée change d'une quantité donnée. La *stabilité* d'un système concerne sa capacité à donner la même sortie pour une entrée donnée au cours d'une certaine durée. Le terme *répétabilité* fait référence à la capacité d'un système à donner la même valeur pour des mesures répétées de la même grandeur. La *fiabilité* d'un système se définit comme la probabilité qu'il fonctionne à un niveau de performance convenu.

Les capteurs les plus utilisés sont les interrupteurs mécaniques, les interrupteurs de proximité, qu'ils soient à courants de Foucault, de type Reed, capacitifs ou inductifs, les interrupteurs photoélectriques, qu'ils soient de type transmissif ou réflectif, les encodeurs, qui produisent une sortie numérique en fonction d'un déplacement angulaire ou linéaire, les encodeurs incrémentaux, qui mesurent des déplacements angulaires, les encodeurs absolus, dont la sortie binaire représente de manière unique chaque position angulaire, les capteurs de température, comme les bilames, les thermomètres à résistance de platine, les thermistances, les thermodiodes, les thermotransistors ou les thermocouples, les capteurs de position et de déplacement, comme les potentiomètres, les transformateurs différentiels à variation linéaire et les capteurs de déplacement capacitifs, les extensomètres, dont la résistance change en fonction de la déformation, les capteurs de pression, comme les capteurs à membrane, les détecteurs de niveau de liquide, qui utilisent des capteurs de pression ou des flotteurs, ainsi que les débitmètres, comme ceux à diaphragme.

Les dispositifs de sortie les plus utilisés sont les relais, les vannes de commande directionnelle à cylindre, les moteurs à courant continu et les moteurs pas à pas.

2.5 Problèmes

Pour les questions 1 à 14, quatre réponses sont proposées : A, B, C et D. Choisissez la réponse correcte.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

1. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Un interrupteur de proximité :
   (i) peut être utilisé pour détecter la présence d'une pièce en déplacement ;
   (ii) est activé par des contacts qui établissent ou coupent un circuit électrique.

   Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
   A. (i) V (ii) V
   B. (i) V (ii) F
   C. (i) F (ii) V
   D. (i) F (ii) F
2. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Une thermistance est un capteur de température dont les changements de résistance sont :
   (i) une fonction non linéaire de la température ;
   (ii) importants pour des modifications de température comparativement faibles.

   Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
   A. (i) V (ii) V
   B. (i) V (ii) F
   C. (i) F (ii) V
   D. (i) F (ii) F
3. Un capteur de pression à membrane est nécessaire pour mesurer la pression manométrique dans un système. Un tel capteur doit disposer d'une membrane avec :
   A. Une chambre sous vide sur l'un des côtés.
   B. Un côté ouvert à l'air libre.
   C. La pression appliquée sur les deux côtés.
   D. Une pression ajustable contrôlée appliquée sur l'un des côtés.
4. Lorsqu'il est soumis à une déformation de 0,001, un extensomètre à résistance électrique dont le facteur de jauge est de 2,0 et la résistance est de 100 Ω présente une variation de résistance de :
   A. 0,0002 Ω
   B. 0,002 Ω
   C. 0,02 Ω
   D. 0,2 Ω
5. Un encodeur rotatif incrémental produit une sortie qui correspond à une mesure directe :
   A. Du diamètre de l'axe.
   B. Du changement du diamètre de l'axe.
   C. Du changement de la position angulaire de l'axe.
   D. De la position angulaire absolue de l'axe.

Copyright © 2015 Dunod.

6. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Les dispositifs d'entrées qui produisent une entrée analogique correspondant à un déplacement comprennent :
   (i) un potentiomètre linéaire ;
   (ii) un transformateur différentiel à variation linéaire.

   Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
   A. (i) V (ii) V
   B. (i) V (ii) F
   C. (i) F (ii) V
   D. (i) F (ii) F

Les problèmes 7 et 8 font référence à la Figure 2.43, qui montre le symbole d'une vanne de commande directionnelle.

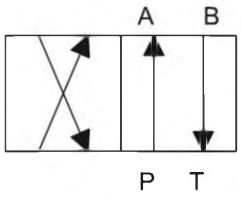


Figure 2.43 – Schéma pour les problèmes 7 et 8.

7. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). La vanne a :
   (i) quatre orifices ;
   (ii) deux positions.

   Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
   A. (i) V (ii) V
   B. (i) V (ii) F
   C. (i) F (ii) V
   D. (i) F (ii) F

8. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Dans les positions de commande :
   (i) A est connecté à T et P à B ;
   (ii) P est connecté à A et B à T.

   Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
   A. (i) V (ii) V
   B. (i) V (ii) F
   C. (i) F (ii) V
   D. (i) F (ii) F

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

9. Pour le montage de la Figure 2.44, indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F) :
   (i) lorsqu'un courant passe au travers du solénoïde, la tige sort ;
   (ii) lorsque le courant est coupé, la tige reste sortie.

   Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
   A. (i) V (ii) V
   B. (i) V (ii) F
   C. (i) F (ii) V
   D. (i) F (ii) F

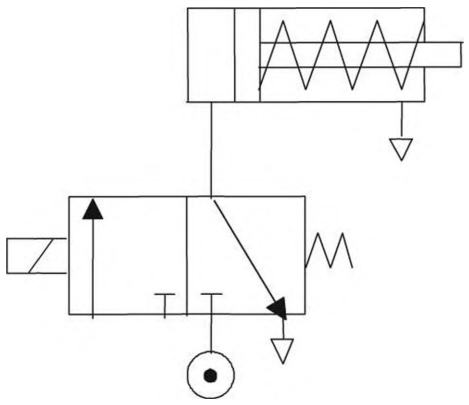


Figure 2.44 – Schéma pour le problème 9.

10. Pour le montage de la Figure 2.45, indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F) :
    (i) lorsque le solénoïde A est sous tension, la tige sort ;
    (ii) lorsque le solénoïde B est sous tension, la tige sort.

    Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
    A. (i) V (ii) V
    B. (i) V (ii) F
    C. (i) F (ii) V
    D. (i) F (ii) F

11. Pour les deux vannes 3/2 de la Figure 2.46, indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F) :
    (i) lorsque que le solénoïde de la vanne 1 est sous tension, A est ouvert ;
    (ii) lorsque que le solénoïde de la vanne 2 est sous tension, A est ouvert.

    Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
    A. (i) V (ii) V
    B. (i) V (ii) F
    C. (i) F (ii) V
    D. (i) F (ii) F

Copyright © 2015 Dunod.

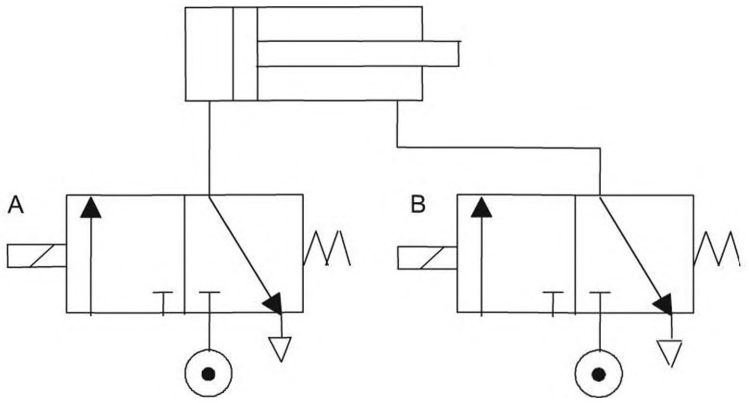


Figure 2.45 – Schéma pour le problème 10.

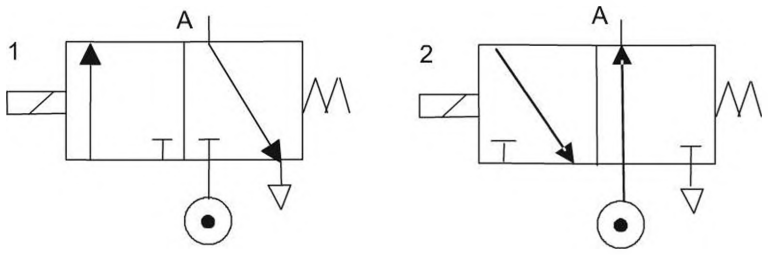


Figure 2.46 – Schéma pour le problème 11.

12. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Si l'angle de pas d'un moteur pas à pas est de 1,8°, cela signifie que :
    (i) chaque impulsion envoyée au moteur fait tourner son axe de 1,8° ;
    (ii) l'axe du moteur prend 1 s pour pivoter de 1,8°.

    Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
    A. (i) V (ii) V
    B. (i) V (ii) F
    C. (i) F (ii) V
    D. (i) F (ii) F

13. L'angle de pas d'un moteur pas à pas est de 7,5°. La fréquence de l'entrée numérique qui permet de produire une rotation de dix révolutions par seconde est de :
    A. 48 impulsions par seconde
    B. 75 impulsions par seconde
    C. 480 impulsions par seconde
    D. 750 impulsions par seconde

14. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Un interrupteur de proximité est nécessaire pour détecter la présence d'un objet non métallique. Les types d'interrupteurs adaptés sont :
    (i) les interrupteurs à courants de Foucault ;
    (ii) des interrupteurs capacitifs.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?

A. (i) V (ii) V
B. (i) V (ii) F
C. (i) F (ii) V
D. (i) F (ii) F

15. Expliquez le fonctionnement des dispositifs d'entrées suivants, en précisant la forme du signal perçu et la sortie : (a) interrupteur Reed, (b) encodeur rotatif incrémental, (c) interrupteur photoélectrique transmissif, (d) interrupteur de pression à membrane.
16. Expliquez comment le fonctionnement en tout ou rien et le sens d'un moteur à courant continu peuvent être commandés par des interrupteurs.
17. Expliquez le principe du moteur pas à pas et indiquez les différents types disponibles.
18. Choisissez les capteurs qui peuvent être utilisés dans les applications suivantes : (a) comptage des colis se déplaçant sur une bande transporteuse, (b) vérification du niveau de lait dans une bouteille en plastique se déplaçant sur une bande transporteuse, (c) détection de la sortie du piston d'un vérin jusqu'à un point précis, (d) détection de l'arrivée d'une plaque métallique à la position appropriée sous un outil.
19. Les caractéristiques suivantes font partie des spécifications d'un moteur pas à pas. Expliquez la signification des termes suivants : quatre phases, angle de pas de 7,5°, courant par phase de 130 mA, résistance par phase de 94 Ω, inductance par phase de 43 mH, circuit de pilotage approprié SAA1027.
20. Proposez une solution de contrôle d'un moyeu de manière à positionner un mécanisme à intervalles de 5°.
21. Des bouteilles opaques de différentes tailles se déplacent sur une bande transporteuse. Proposez une méthode qui permet (a) de détecter les différentes tailles et (b) de pousser les bouteilles hors de la bande.

2.6 Recherches complémentaires

1. Recherchez les spécifications de thermistances et sélectionnez un modèle qui permet de surveiller des températures d'environ 40 °C.
2. Recherchez les spécifications du capteur de pression MPX100AP et décrivez ses utilisations envisageables ainsi que ses possibilités.
3. Recherchez les spécifications du capteur de température intégré LM3911N et décrivez ses utilisations envisageables ainsi que ses possibilités.

Copyright © 2015 Dunod.

3 • SYSTÈMES NUMÉRIQUES

Les systèmes numériques manipulent des entrées, qui sont essentiellement des signaux tout ou rien, avec les deux niveaux de signaux représentés par 0 et 1, autrement dit des *valeurs binaires*. Le système de numération généralement employé pour les calculs est le *système décimal*. Il se fonde sur dix chiffres : 0, 1, 2, 3, 4, 5, 6, 7, 8, 9. Lorsqu'un nombre est représenté dans ce système, la position d'un chiffre dans le nombre indique le poids associé à ce chiffre. Ce poids augmente d'un facteur dix pour chaque position en partant de la droite :

	10^3	10^2	10^1	10^0
	Milliers	Centaines	Dizaines	Unités
Décimal	1000	100	10	1

Pour le nombre décimal 1 234, la valeur de la position du chiffre 1 est 10^3, celle du 2 est 10^2, celle du 3 est 10^1 et celle du 4 est 10^0. Toutefois, les calculs peuvent se faire dans n'importe quelle base. Le système décimal se révèle pratique, principalement en raison de nos dix doigts. Si nous n'avions eu que deux doigts, nous aurions probablement opté pour un système de comptage différent. Les ordinateurs et les API emploient un système de comptage en base deux, car il correspond à leur fonctionnement fondé sur les deux valeurs binaires des signaux tout ou rien. Toutefois, d'autres systèmes numériques sont également employés par les API. Par exemple, les adresses des entrées et des sorties sont souvent numérotées à l'aide du système octal, c'est-à-dire en base huit. Néanmoins, l'API lui-même manipule des nombres binaires. Dans ce chapitre, nous examinons les différents systèmes de numération.

Nous en profitons également pour introduire les systèmes logiques. Les *systèmes logiques combinatoires* prennent des entrées binaires et les combinent pour produire une sortie binaire. La relation entre les entrées et la sortie peut être décrite sous forme d'une *table de vérité*. Avec de tels systèmes, la sortie qui correspond à une combinaison particulière d'entrées est déterminée uniquement par l'état de ces entrées au moment examiné. Dans les *systèmes logiques séquentiels*, l'état de la sortie dépend non seulement des entrées actuelles mais également des entrées précédentes. Dans ce chapitre, nous présentons ces deux systèmes logiques.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

3.1 Binaire

Le *système binaire* se fonde uniquement sur deux chiffres : 0 et 1. Ils sont appelés *bits* (contraction des mots anglais « *bi*nary » et « digi*ts* »). Lorsqu'un nombre est représenté dans ce système, la position d'un chiffre dans le nombre indique le poids associé à ce chiffre. Le poids augmente d'un facteur deux pour chaque position en partant de la droite :

	2^3	2^2	2^1	2^0
	bit 3	bit 2	bit 1	bit 0
Binaire	1000	100	10	1

Le bit zéro est le *bit de poids faible* (LSB, *Least Significant Bit*), tandis que le bit ayant la plus grande valeur est le *bit de poids fort* (MSB, *Most Significant Bit*). Par exemple, pour le nombre binaire 1010, le LSB est 0, puisqu'il s'agit du bit situé à l'extrême droite, et le MSB est 1, puisqu'il s'agit du bit à l'extrême gauche.

La conversion d'un nombre binaire en un nombre décimal se fait en additionnant les puissances de deux indiquées par l'emplacement d'un chiffre dans le nombre global. Pour le nombre binaire 1010, nous avons 1 à l'emplacement de valeur 2^3, 0 à l'emplacement de valeur 2^2, 1 à l'emplacement de valeur 2^2 et 0 à l'emplacement de valeur 2^0. Voici donc sa conversion en un nombre décimal :

	2^3	2^2	2^1	2^0
Binaire	1	0	1	0
Décimal	$2^3 = 8$	0	$2^1 = 2$	0

L'équivalent décimal au nombre binaire 1010 est donc 8 + 2 = 10.

La conversion d'un nombre décimal en un nombre binaire se fait en recherchant les puissances de 2 appropriées. Pour cela, nous effectuons des divisions par 2 successives, en notant le reste de chaque division. Voici l'application de la méthode au nombre binaire 31 :

31 ÷ 2 = 15 et il reste 1 (cette opération donne le LSB)

15 ÷ 2 = 7 et il reste 1

7 ÷ 2 = 3 et il reste 1

3 ÷ 2 = 1 et il reste 1 (cette opération donne le MSB)

L'équivalent binaire au nombre décimal 31 est donc 1111. La première opération donne le LSB, car nous avons divisé 31 par 2, c'est-à-dire 2^1, et avons trouvé un reste de 1 pour le chiffre 2^0. La dernière opération donne le MSB, car 31 a été divisé quatre fois par 2, c'est-à-dire 2^4, et il reste 1.

Copyright © 2015 Dunod.

3.2 Octal et hexadécimal

Les ordinateurs utilisent les nombres binaires car les deux états représentés par 0 et 1 sont faciles à manipuler dans les circuits de commutation, où ils peuvent représenter les états marche et arrêt. Cependant, les nombres binaires présentent un problème : une petite valeur binaire nécessite un grand nombre de chiffres. Par exemple, le nombre décimal 9, qui utilise un seul chiffre, a besoin de quatre chiffres sous sa forme binaire 1001. Le nombre décimal 181, qui s'écrit avec trois chiffres, en demande huit sous sa forme binaire 10110101. C'est pour cette raison que des nombres octaux ou hexadécimaux sont parfois employés. Ils sont plus faciles à manipuler et se trouvent à « mi-chemin » entre les nombres décimaux utilisés par les hommes et les nombres binaires utilisés par les ordinateurs. Par exemple, Allen-Bradley utilise un système octal pour les adresses des entrées et des sorties dans ses API.

3.2.1 Système octal

Le *système octal* se fonde sur huit chiffres : 0, 1, 2, 3, 4, 5, 6, 7. Lorsqu'un nombre est représenté dans ce système, la position d'un chiffre dans le nombre indique le poids associé à ce chiffre. Le poids augmente d'un facteur huit pour chaque position en partant de la droite :

	8^3	8^2	8^1	8^0
Octal	1000	100	10	1

La conversion d'un nombre décimal en un nombre octal se fait par des divisions par 8 successives, en notant le reste de chaque division. Le nombre décimal 15 divisé par 8 donne 1 avec un reste de 7 et s'écrit donc 17 en octal. Pour convertir un nombre octal en décimal, nous multiplions chaque chiffre par la puissance de 8 qui correspond à sa position dans le nombre. Par exemple, le nombre octal 365 correspond à $3 \times 8^2 + 6 \times 8^1 + 5 \times 8^0 = 245$. Pour convertir du binaire vers l'octal, le nombre binaire est écrit sous forme de groupes de trois bits, en commençant par le bit de poids faible. Par exemple, le nombre binaire 11010110 est décomposé de la manière suivante :

11 010 110

Chaque groupe est ensuite remplacé par le chiffre correspondant entre 0 et 7. Par exemple, le nombre binaire 110 correspond à 6, le nombre 010, à 2 et le nombre 11, à 3. En octal, le nombre binaire précédent s'écrit donc 326. Voici un autre exemple pour le nombre binaire 100111010 :

100 111 010 (en binaire)

4 7 2 (en octal)

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

Pour la conversion octal vers binaire, chaque chiffre octal est converti en son équivalent sur trois bits. Ainsi, pour le nombre octal 21, 1 devient 001 et 2 devient 010 :

2 1 (en octal)

010 001 (en binaire)

Par conséquent, le nombre octal 21 s'écrit 010001 en binaire.

3.2.2 Système hexadécimal

Le *système hexadécimal* (*hexa*) se fonde sur seize chiffres et lettres : 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F. Lorsqu'un nombre est représenté dans ce système, la position d'un chiffre dans le nombre indique le poids associé à ce chiffre. Le poids augmente d'un facteur seize pour chaque position en partant de la droite :

	16^3	16^2	16^1	16^0
Hexa	1000	100	10	1

Le nombre décimal 15 correspond ainsi à F dans le système hexadécimal. Pour convertir les nombres décimaux en hexadécimal, nous procédons à des divisions par 16 successives et notons les restes. Ainsi, le nombre décimal 156 divisé par 16 donne 9 et un reste de 12, et s'écrit donc 9C en hexa. Pour passer de l'hexadécimal au décimal, nous multiplions les chiffres par la puissance 16 qui correspond à leur position dans le nombre. Ainsi, le nombre hexadécimal 12 devient $1 \times 16^1 + 2 \times 16^0 = 18$. Pour convertir les nombres binaires en nombres hexadécimaux, nous regroupons les chiffres binaires par groupes de quatre en commençant par le bit de poids faible. Par exemple, pour le nombre binaire 1110100110, nous avons :

11 1010 0110 (en binaire)

3 A 6 (en hexadécimal)

Pour la conversion hexadécimal vers binaire, chaque nombre hexadécimal est transformé en son équivalent sur quatre bits. Ainsi, pour le nombre 1D, nous avons 0001 pour le 1 et 1101 pour le D :

1 D (en hexadécimal)

0001 1101 (en binaire)

L'équivalent binaire au nombre hexadécimal 1D est donc 00011101.

3.3 Décimal codé en binaire

Puisque les hommes ont plutôt tendance à manipuler des nombres décimaux et les ordinateurs, des nombres binaires, le problème de conversion se pose en permanence. Malheureusement, il n'existe aucun lien simple entre l'emplacement des chiffres d'un nombre décimal et celui des chiffres d'un nombre binaire. Par conséquent, une autre méthode, le système *décimal codé en binaire* (BCD, *Binary Coded Decimal*), est souvent employée. Dans ce système, chaque chiffre décimal est codé

Copyright © 2015 Dunod.

séparément en binaire. Par exemple, pour le nombre décimal 15, le 5 est converti en chiffre binaire 0101 et le 1 est converti en 0001 :

1 5 (en décimal)

0001 0101 (en BCD)

Par conséquent, le nombre décimal 15 s'écrit 0001 0101 en BCD. Dans le système BCD, le plus grand nombre décimal que l'on peut représenter est 9, avec les quatre chiffres binaires 1001.

Pour convertir un nombre BCD en un nombre décimal, chaque groupe de quatre bits est converti séparément en la valeur décimale correspondante. Par exemple, le nombre BCD 0011 1001 donne 3 pour 0011 et 9 pour 1001, c'est-à-dire 39 en décimal.

0011 1001 (en BCD)

3 9 (en décimal)

Dans les API, les données numériques sont souvent saisies à l'aide d'interrupteurs rotatifs ou à molette, avec des valeurs dans la plage 0 à 9. Par conséquent, ils peuvent être équipés d'une rangée d'interrupteurs de ce type, l'un donnant, par exemple, les centaines, l'autre, les dizaines et l'autre encore, les unités. La sortie de chaque interrupteur est ensuite convertie indépendamment en binaire, pour obtenir un résultat global sous forme de nombre BCD. Certains API disposent d'une fonction de conversion des nombres BCD en nombres binaires, pour d'autres, elle doit être réalisée par programme.

3.4 Nombres en binaire, octal, hexadécimal et BCD

Le Tableau 3.1 présente des exemples de nombres en décimal, binaire, octal, hexadécimal et BCD.

Tableau 3.1 Exemples de nombres dans différents systèmes de numération

Décimal	Binaire	Octal	Hexadécimal	BCD
0	00000	0	0	0000 0000
1	00001	1	1	0000 0001
2	00010	2	2	0000 0010
3	00011	3	3	0000 0011
4	00100	4	4	0000 0100
5	00101	5	5	0000 0101
6	00110	6	6	0000 0110

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

Tableau 3.1 Exemples de nombres dans différents systèmes de numération *(suite)*

Décimal	Binaire	Octal	Hexadécimal	BCD
7	00111	7	7	0000 0111
8	01000	10	8	0000 1000
9	01001	11	9	0000 1001
10	01010	12	A	0001 0000
11	01011	13	B	0001 0001
12	01100	14	C	0001 0010
13	01101	15	D	0001 0011
14	01110	16	E	0001 0100
15	01111	17	F	0001 0101
16	10000	20	10	0001 0110
17	10001	21	11	0001 0111

3.5 Arithmétique binaire

L'addition de nombres binaires se fait conformément aux règles suivantes :

$0 + 0 = 0$

$0 + 1 = 1 + 0 = 1$

$1 + 1 = 10$

$1 + 1 + 1 = 11$

Prenons pour exemple l'addition des nombres binaires 01110 et 10011.

```
          01110
          10011
Somme    100001
```

Pour le bit 0 de la somme, nous avons $0 + 1 = 1$. Pour le bit 1 de la somme, $1 + 1 = 10$, ce qui donne 0 avec la retenue 1 dans la colonne suivante. Pour le bit 2 de la somme, $1 + 0 +$ la retenue $1 = 10$. Pour le bit 3 de la somme, $1 + 0 +$ la retenue $1 = 10$. La procédure se poursuit pour les différents bits et nous obtenons 100001 comme valeur finale.

La soustraction de nombres binaires se fait conformément aux règles suivantes :

$0 - 0 = 0$

Copyright © 2015 Dunod.

$1 - 0 = 1$

$1 - 1 = 0$

Pour le calcul de $0 - 1$, un 1 est pris dans la colonne de gauche suivante qui contient un 1. L'exemple suivant illustre cette méthode dans la soustraction de 01110 à 11011 :

	11011
	01110
Différence	01101

Pour le bit 0 de la différence, nous avons $1 - 0 = 1$. Pour le bit 1, $1 - 1 = 0$. Pour le bit 2, nous avons $0 - 1$. Nous prenons un 1 dans la colonne suivante et avons donc $10 - 1 = 1$. Pour le bit 3, nous avons $0 - 1$ (n'oubliez pas que nous avons pris le 1). Une fois encore, nous prenons le 1 dans la colonne suivante et avons $10 - 1 = 1$. Pour le bit 4, $0 - 0 = 0$ (nous avons pris le 1).

3.5.1 Nombres signés

Les nombres binaires donnés jusqu'à présent ne contiennent aucune indication permettant de savoir s'ils sont positifs ou négatifs : ils sont *non signés*. Puisqu'il est souvent nécessaire de gérer des nombres positifs et des nombres négatifs, il faut un système permettant de les différencier. Pour cela, il est possible d'ajouter un *bit de signe*. Lorsqu'un nombre est *signé*, son MSB indique son signe : un 0 correspond à un nombre positif, un 1, à un nombre négatif. Par conséquent, un nombre signé a le format suivant :

xxxx xxxx
↑
Bit de signe

Lorsqu'un nombre est positif, il s'écrit de manière normale, en le commençant par un 0. Le nombre binaire positif 10110 s'écrit donc 010110. Quant au nombre binaire négatif 10110, il s'écrit 110110. Toutefois, la manipulation par les ordinateurs des nombres négatifs écrits selon cette méthode n'est pas des plus simles.

3.5.2 Compléments à un et à deux

Pour écrire les nombres négatifs signés, une solution plus pratique se fonde sur le complément à deux. Un nombre binaire possède deux compléments, appelés *complément à un* et *complément à deux*. Le complément à un d'un nombre binaire s'obtient en remplaçant tous les 1 du nombre non signé par des 0 et tous les 0 par des 1. Par exemple, le complément à un du nombre binaire 101101 est 010010. Le complément à deux s'obtient à partir du complément à un, en ajoutant 1 au LSB. Ainsi, le complément à deux de 101101 est 010011.

Lorsque nous avons un nombre négatif, pour obtenir le complément à deux signé, nous prenons tout d'abord le complément à deux, puis le signons avec un 1. Examinons par exemple la représentation du nombre décimal −6 sous forme d'un

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

nombre en complément à deux signé sur huit bits. Nous commençons par écrire le nombre binaire +6, c'est-à-dire 0000110, puis nous prenons son complément à un, 1111001, ajoutons 1 pour obtenir le complément à deux, 1111010, et, enfin, le signons avec 1 (négatif). Le résultat de ces opérations successives est 11111010.

Nombre binaire non signé, sans le bit de signe	000 0110
Complément à un	111 1001
Addition avec 1	1
Complément à deux non signé	111 1010
Complément à deux signé	1111 1010

Le Tableau 3.2 recense les compléments à deux signés sur quatre bits de quelques nombres décimaux.

Tableau 3.2 Compléments à deux signés

Nombre décimal	Complément à deux signé
–5	1011
–4	1100
–3	1101
–2	1110
–1	1111

Lorsque le nombre est positif, nous signons le nombre binaire normal avec un 0. Autrement dit, seuls les nombres négatifs sont écrits dans leur complément à deux. En adoptant cette méthode d'écriture des nombres positifs et négatifs, l'addition des valeurs binaires signées équivalentes à +4 et à –4, c'est-à-dire 0000 0100 et 1111 1100, nous obtenons (1)0000 0000. Cela correspond à zéro car, avec le nombre de bits utilisés, (1) est ignoré.

La soustraction d'un nombre positif à un nombre positif peut être vue comme l'addition d'un nombre négatif à un nombre positif. Il suffit donc d'obtenir le complément à deux signé du nombre négatif et de l'ajouter au nombre positif signé. Ainsi, pour la soustraction du nombre décimal 6 au nombre décimal 4, nous pouvons effectuer l'opération (+4) + (–6). Nous ajoutons donc le nombre positif signé au complément à deux signé du nombre négatif.

Forme binaire de +4	0000 0100
(-6) sous forme de complément à deux signé	1111 1010
Somme	1111 1110

Le MSB, c'est-à-dire le signe, du résultat est 1 et la somme est donc négative. Il s'agit du complément à deux signé sur huit bits de –2.

Copyright © 2015 Dunod.

Pour ajouter deux nombres négatifs, il suffit d'obtenir le complément à deux signé de chaque nombre et de les ajouter. Dès qu'un nombre est négatif, nous utilisons le complément à deux signé, dès qu'il est positif, nous utilisons simplement le nombre signé.

3.5.3 Nombres à virgule flottante

Avant d'examiner les nombres à virgule flottante, étudions les *nombres à virgule fixe*. Dans les nombres à virgule fixe, l'emplacement de la virgule qui sépare la partie entière de la partie fractionnaire est figé. Par exemple, 15,3 est un nombre décimal à virgule fixe, 1010,1100 est un nombre binaire à virgule fixe et DE,2A est un nombre hexadécimal à virgule fixe. Avec le nombre binaire sur huit bits, nous avons quatre chiffres avant la virgule binaire et quatre chiffres après. Lorsque deux nombres binaires de ce type sont ajoutés par un système informatique, la procédure consiste à déterminer que la virgule fixe se trouve au même endroit dans les deux nombres. Dans ce cas, elle peut être ignorée au cours de l'addition, l'opération peut être effectuée, puis la virgule peut être insérée dans le résultat à sa position déterminée. Par exemple, supposons que nous souhaitions ajouter 0011,1010 et 0110,1000. Nous retirons la virgule binaire pour obtenir :

$$0011\ 1010 + 0110\ 1000 = 1010\ 0010$$

En remettant la virgule binaire à son emplacement, nous obtenons 1010,0010.

L'utilisation des virgules fixes pose des problèmes. En effet, si nous manipulons des nombres très grands ou très petits, nous risquons d'avoir un grand nombre de zéros entre les chiffres entiers et la virgule, comme dans 0,000 000 000 000 023. C'est pourquoi, la *notation scientifique* est employée avec ces nombres. Le nombre précédent peut s'écrire $0{,}23 \times 10^{-13}$ ou $2{,}3 \times 10^{-14}$ ou 23×10^{-15}. De même, le nombre binaire 0,0000 0111 0010 peut être présenté sous la forme 110010×2^{-12} (la puissance 12 est également au format binaire) ou $11001{,}0 \times 2^{-11}$ (la puissance 11 est au format binaire). Cette notation est dite à *virgule flottante*.

Un nombre à virgule flottante est donné sous la forme $m \times b^e$, où m est la *mantisse*, b, la *base*, et e, l'*exposant*. Pour les nombres binaires, puisque la base est 2, nous avons $m \times 2^e$. Si nous savons que nous manipulons des nombres binaires, il est inutile d'enregistrer la base avec le nombre. Par conséquent, un système informatique stocke donc la mantisse et l'exposant, en plus du signe qui indique si le nombre est positif ou négatif.

Puisqu'un nombre à virgule flottante peut être stocké de différentes manières, par exemple $0{,}1 \times 10^2$ et $0{,}01 \times 10^3$, ces nombres ont été normalisés pour les systèmes informatiques : ils sont tous enregistrés sous la forme $0{,}1 \times b^e$. Les nombres binaires sont donc de la forme $0{,}1 \times 2^e$. Le nombre binaire 0,00001001 devient ainsi $0{,}1001 \times 2^{-4}$. Pour tenir compte du signe d'un nombre binaire, nous ajoutons ensuite un bit de signe égal à 0 pour une valeur positive et 1 pour une valeur négative. Le nombre $0{,}1001 \times 2^{-4}$ devient donc $1{,}1001 \times 2^{-4}$ s'il est négatif, et $0{,}1001 \times 2^{-4}$ s'il est positif.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

Contrairement aux nombres à virgule fixe, il est impossible d'ajouter directement des nombres à virgule flottante dont les exposants diffèrent. Il faut commencer par transformer les nombres pour qu'ils aient le même exposant.

3.6 Données d'un API

La plupart des API travaillent avec des mots de seize bits, un *mot* représentant un groupe de bits d'information. Cela permet d'obtenir des nombres positifs dans l'intervalle 0 à +65 535, c'est-à-dire 1111 1111 1111 1111, ou des nombres négatifs dans l'intervalle −32 768 à +32 767, en utilisant le complément à deux et le MSB pour le signe. Ces nombres signés sont des *entiers* et le symbole INT (*integer*) est utilisé dans les programmes pour les entrées et les sorties sur seize bits. Le symbole SINT (*small integer*) désigne des *nombres entiers courts* pour lesquels seuls huit bits sont utilisés ; ces nombres sont donc dans l'intervalle −128 à +127. Le symbole DINT (*double integer*) est employé pour les *nombres entiers doubles* qui sont codés sur trente-deux bits et se trouvent dans l'intervalle -2^{31} à $+2^{31} - 1$. LINT (*long integer*) correspond aux *nombres entiers longs*, qui sont codés sur soixante-quatre bits et se trouvent dans l'intervalle -2^{63} à $+2^{63} - 1$. Lorsque les nombres ne sont pas signés, les symboles UINT, USINT, UDINT et ULINT sont employés pour les entiers, les entiers courts, les entiers doubles et les entiers longs.

Les fractions décimales sont appelées *nombres à virgule flottante* ou *nombres réels* et sont représentées par le symbole REAL pour les entrées et les sorties dans les programmes. Elles sont composées de deux mots de seize bits. Par conséquent, le nombre $1{,}234567 \times 10^{+3}$ peut être représenté par 1,234567E+03, où E signale que le nombre qui suit correspond à l'exposant. Le symbole LREAL est utilisé pour les *nombres réels longs* codés sur soixante-quatre bits.

Le symbole BOOL est employé pour les *données booléennes*, comme les données tout ou rien, c'est-à-dire 0 ou 1, qui sont donc représentées par un seul bit.

Les durées, comme la durée d'un processus, sont représentées selon la norme CEI à l'aide des symboles *d* (*day*) pour les jours, *h* pour les heures, *m* pour les minutes, *s* pour les secondes et *ms* pour les millisecondes. Par exemple, T#12d2h5s3ms ou TIME#12d2h5s correspond à 12 jours, 2 heures, 5 secondes et 3 millisecondes. Notez que le symbole # est utilisé pour indiquer que la quantité qui suit est numérique.

3.7 Systèmes logiques combinatoires

Prenons un système de « verrouillage » qui protège le fonctionnement d'une machine. Cette machine peut démarrer uniquement si deux conditions de sécurité suivantes sont remplies : la pièce est en position et le dispositif de sécurité est en place. Le positionnement de la pièce peut être considéré comme l'entrée A d'un système, tandis que la mise en place du dispositif de sécurité peut être considérée comme l'entrée B (voir Figure 3.1).

Copyright © 2015 Dunod.

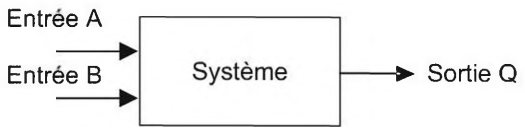


Figure 3.1 – Système de verrouillage d'une machine.

Pour que les conditions d'entrée puissent être exprimées sous forme binaire, nous imposons que chaque entrée n'ait que deux possibilités. Dans ce cas, si nous supposons que la consultation de chaque entrée ne peut donner qu'une réponse OUI ou NON, nous avons simplement deux conditions, que l'on peut exprimer par 1 pour OUI et 0 pour NON. Ainsi, l'entrée A peut être formulée « la pièce est-elle en position ? », avec pour réponse OUI ou NON. De la même manière, l'entrée B peut être formulée « le dispositif de sécurité est-il en place ? », avec pour réponse OUI ou NON. Pour ce système, nous voulons que la sortie ne soit pas nulle lorsque les entrées A et B sont à 1. Cette relation entre les entrées et la sortie peut être donnée par une *table de vérité*, qui montre toutes les combinaisons possibles des entrées, ainsi que celles qui donnent une sortie à 1, c'est-à-dire à OUI, ou à 0, c'est-à-dire à NON. Le Tableau 3.3 correspond à la table de vérité de notre exemple de système.

Tableau 3.3 Une table de vérité

Entrée A	Entrée B	Sortie Q
0	0	0
1	0	0
0	1	0
1	1	1

Chaque entrée peut prendre uniquement deux valeurs : 0 ou 1. Elles sont ainsi appelées *variables à deux états* ou *variables logiques*. Le système complet construit à partir d'une telle variable est appelé *système logique* ou *porte logique*. Si la sortie de ce système dépend uniquement de l'état actuel des entrées, comme dans le cas du verrouillage de la machine, il s'agit d'un *système logique combinatoire*.

Les systèmes logiques combinatoires utiles, que nous rencontrerons au Chapitre 5, sont les portes ET, OU, NON, NON-ET, NON-OU et OU exclusif. Le système de verrouillage de la machine est un exemple de porte ET dans lequel l'entrée A *et* l'entrée B doivent être à 1 pour que la sortie soit à 1.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

3.8 Systèmes logiques séquentiels

Dans un *système logique séquentiel*, la sortie actuelle dépend des états précédents et des états actuels des entrées. Il diffère donc d'un système logique combinatoire où la sortie dépend *uniquement* de l'état actuel des entrées. Un compteur binaire peut être vu comme un système logique séquentiel car la sortie binaire dépend de l'entrée actuelle et de la somme des entrées précédentes. En cela, il possède une « mémoire ».

La plupart des systèmes séquentiels se fondent sur un petit nombre de systèmes logiques séquentiels appelés *bistables*, car ils disposent de deux conditions stables et peuvent être basculés de l'une à l'autre en agissant sur d'autres entrées. Après que le circuit a basculé, il reste dans le nouvel état stable jusqu'à ce qu'une autre impulsion d'entrée le replace dans l'état d'origine. Les bistables sont donc des dispositifs mémoire, puisqu'ils peuvent « mémoriser » l'effet d'une entrée après qu'elle a été supprimée.

Un verrou et une bascule, ainsi appelée car, sur commande, le dispositif peut basculer d'un état à l'autre, sont des bistables. Un verrou est déclenché par le niveau de tension appliqué en entrée, sous condition qu'il ait été activé par l'entrée d'horloge au niveau, en général, haut. Les bascules sont des dispositifs qui changent d'état sur le front montant ou le front descendant d'une impulsion d'activation ou d'horloge.

3.8.1 Verrous

Le *verrou D à entrée d'horloge* possède une entrée de données D, des sorties Q et $\overline{Q}$, et une entrée d'activation/horloge CLK (voir Figure 3.2). Q est toujours le complément de $\overline{Q}$. L'état logique de la sortie Q suit tous les changements de l'état logique de l'entrée D tant que l'entrée d'horloge reste au niveau haut. Lorsque l'entrée d'horloge passe au niveau bas, la sortie Q conserve l'état logique que l'entrée D avait à ce moment-là, quels que soient les changements qui peuvent se produire par la suite sur l'entrée D. Lorsque l'entrée d'horloge repasse au niveau haut, la sortie Q suit à nouveau les changements de l'état logique de l'entrée D. Lorsque l'horloge est au niveau haut, le verrou est dit *transparent.*

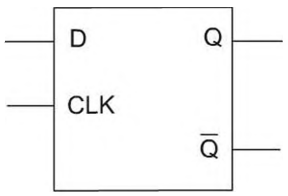


Figure 3.2 – Le verrou D à entrée d'horloge.

Il est possible de construire des tables de vérité pour les verrous, mais elles doivent prendre en compte les effets d'une impulsion sur l'entrée d'horloge. C'est pourquoi elles sont souvent appelées *tables de fonctionnement.* Le Tableau 3.4 donne la table de fonctionnement du verrou D à entrée d'horloge. Q^+ correspond à l'état de la sortie Q après qu'une impulsion d'horloge a déclenché l'entrée.

Copyright © 2015 Dunod.

Tableau 3.4 Table de fonctionnement du verrou D à entrée d'horloge

CLK	D	Q	Q^+	
0	0	0	0	Pas de changement en sortie. Elle conserve sa valeur précédente.
0	0	1	1	
0	1	0	0	
0	1	1	1	
1	0	0	0	La sortie prend une nouvelle valeur.
1	0	1	0	
1	1	0	1	
1	1	1	1	

Le *verrou RS à entrée d'horloge*, ou verrou RSH, possède deux entrées, R pour mettre la sortie à 0 et S pour la mettre à 1, des sorties Q et $\overline{Q}$, et une entrée d'activation/horloge CLK (voir Figure 3.3). Q est toujours le complément de $\overline{Q}$. Lorsque les entrées R et S sont au niveau bas, l'état logique des sorties ne change pas. Lorsque R est à 1 et S à 0, la sortie Q passe à 0, quelle que soit sa valeur précédente. Cette opération est une *mise à 0* (*reset*). Lorsque R est à 0 et S à 1, la sortie Q passe à 1, quelle que soit sa valeur précédente. Cette opération est une *mise à 1* (*set*). Lorsque R et S sont à 1, le fonctionnement du verrou est imprévisible et cette combinaison des entrées doit être interdite. Le Tableau 3.5 présente la table de fonctionnement de ce verrou.

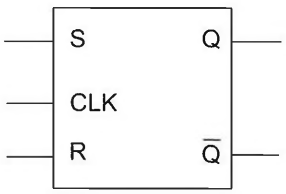


Figure 3.3 – Le verrou RS à entrée d'horloge.

3.8.2 Bascules

Une bascule JK possède deux entrées de données, J et K, une entrée d'horloge et deux sorties, Q et $\overline{Q}$ (voir Figure 3.4). La sortie d'une bascule JK change d'état lorsque l'entrée d'horloge change d'état, soit sur le front montant, soit sur le front descendant. Une bascule JK change toujours d'état lorsque J = K = 1. Le Tableau 3.6 présente sa table de fonctionnement.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

Tableau 3.5 Table de fonctionnement du verrou RS à entrée d'horloge

CLK	S	R	Q	Q^+	
0	0	0	0	0	Pas de changement, état maintenu.
0	0	0	1	1	
0	1	0	0	0	
0	1	0	1	1	
0	0	1	0	0	
0	0	1	1	1	
0	1	1	0	0	
0	1	1	1	1	
1	0	0	0	0	
1	0	0	1	1	
1	1	0	0	1	Mise à 1
1	1	0	1	1	
1	0	1	0	0	Mise à 0
1	0	1	1	0	
1	1	1	0	x	Indeterminée
1	1	1	1	x	

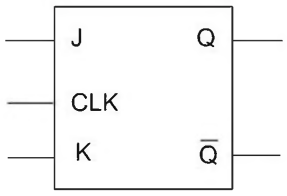


Figure 3.4 – La bascule JK.

Copyright © 2015 Dunod.

Tableau 3.6 Table de fonctionnement de la bascule JK

J	K	Q	Q^+	
0	0	0	0	Pas de changement, état maintenu.
0	0	1	1	
1	0	0	0	
1	0	1	1	
0	0	1	1	
1	0	0	1	Mise à 1
1	0	1	1	
0	1	0	0	Mise à 0
0	1	1	0	
1	1	0	1	Bascule
1	1	1	0	

3.9 En résumé

Le système de numération *décimal* utilise les dix chiffres 0, 1, 2, 3, 4, 5, 6, 7, 8, 9. Le système *binaire* se fonde sur deux chiffres uniquement : 0 et 1. Le système *octal* se fonde sur huit chiffres : 0, 1, 2, 3, 4, 5, 6, 7. Le système *hexadécimal* emploie seize chiffres : 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F. Dans le système *décimal codé en binaire* (BCD, *Binary Coded Decimal*), chaque chiffre décimal est codé individuellement en binaire.

Pour l'addition de deux nombres binaires, les règles sont $0 + 0 = 0$, $0 + 1 = 1$ et $1 + 1 = 10$. Pour la soustraction, elles sont $0 - 0 = 0$, $1 - 0 = 1$ et $1 - 1 = 0$. Les nombres binaires qui ne précisent pas s'ils sont positifs ou négatifs sont dits *non signés*. Lorsqu'un nombre est signé, le bit de poids fort précise son signe : 0 s'il est positif et 1 s'il est négatif. Un nombre binaire possède deux compléments. Le *complément à un* s'obtient en remplaçant tous les 1 du nombre non signé par des 0 et tous les 0 par des 1. Le *complément à deux* s'obtient à partir du complément à un en ajoutant 1 au bit de poids faible. Lorsque le nombre est négatif, le complément à deux est signé avec un 1.

Un nombre à virgule flottante est donné sous la forme $m \times b^e$, où m est la *mantisse*, b, la *base*, et e, l'*exposant*. Pour les nombres binaires, puisque la base est 2, nous avons $m \times 2^e$. Si nous savons que nous manipulons des nombres binaires, il est

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

inutile d'enregistrer la base avec le nombre. Par conséquent, un système informatique stocke la mantisse et l'exposant, en plus du signe qui indique si le nombre est positif ou négatif.

La plupart des API travaillent avec des mots de seize bits, un *mot* représentant un groupe de bits d'information. Les nombres signés sont des *entiers* et le symbole INT est utilisé dans les programmes avec les entrées et les sorties de ces mots de seize bits. Le symbole SINT désigne des *nombres entiers courts* pour lesquels seuls huit bits sont utilisés. Le symbole DINT est employé pour les *nombres entiers doubles* qui sont codés sur trente-deux bits. LINT correspond aux *nombres entiers longs*, qui sont codés sur soixante-quatre bits. Lorsque les nombres ne sont pas signés, les symboles UINT, USINT, UDINT et ULINT sont employés pour les entiers, les entiers courts, les entiers doubles et les entiers longs.

Un système dont les entrées et les sorties sont représentées par un 0 ou un 1 est un *système logique*. Lorsque la sortie d'un tel système dépend uniquement de l'état actuel des entrées, il s'agit d'un *système logique combinatoire*. La relation entre les entrées et la sortie est donnée par une table de vérité, qui recense toutes les combinaisons des entrées qui donnent une sortie à 1 et celles qui donnent une sortie à 0.

Dans un *système logique séquentiel*, la sortie actuelle dépend des entrées précédentes et des entrées actuelles. Ils sont principalement fondés sur des dispositifs appelés *bistables*, car ils possèdent deux états stables et peuvent être basculés de l'un à l'autre par des entrées appropriées. Après que le circuit a commuté, il reste dans le nouvel état stable, jusqu'à ce qu'une autre impulsion d'entrée le force à revenir dans l'état d'origine. Les verrous et les bascules sont des bistables. Un verrou est déclenché par le niveau de tension appliqué en entrée, sous condition qu'il ait été activé par l'entrée d'horloge au niveau, en général, haut. Les bascules sont des dispositifs qui changent d'état sur le front montant ou le front descendant d'une impulsion sur l'entrée d'activation/horloge. Il est possible de donner des tables de vérité pour les verrous, mais elles doivent prendre en compte les effets d'une impulsion sur l'entrée d'horloge. C'est pourquoi elles sont souvent appelées *tables de fonctionnement*.

3.10 Problèmes

1. Convertissez en décimal les nombres binaires suivants : (a) 000011, (b) 111111, (c) 001101.
2. Convertissez en binaire les nombres décimaux suivants : (a) 100, (b) 146, (c) 255.
3. Convertissez en décimal les nombres hexadécimaux suivants : (a) 9F, (b) D53, (c) 67C.
4. Convertissez en hexadécimal les nombres décimaux suivants : (a) 14, (b) 81, (c) 2562.

Copyright © 2015 Dunod.

5. Convertissez en binaire les nombres hexadécimaux suivants : (a) E, (b) 1D, (c) A65.
6. Convertissez en décimal les nombres octaux suivants : (a) 372, (b) 14, (c) 2540.
7. Convertissez en octal les nombres décimaux suivants : (a) 20, (b) 265, (c) 400.
8. Convertissez en binaire les nombres octaux suivants : (a) 270, (b) 102, (c) 673.
9. Convertissez en BCD les nombres décimaux suivants : (a) 20, (b) 35, (c) 92.
10. Donnez le complément à deux signé sur huit bits des nombres décimaux suivants : (a) −1, (b) −35, (c) −125.
11. Donnez l'équivalent décimal des nombres en complément à deux signé sur huit bits suivants : (a) 1111 0000, (b) 1100 1001, (c) 1101 1000.
12. Convertissez en nombres à virgule flottante normalisés les nombres binaires suivants : (a) 0011 0010, (b) 0000 1100, (c) 1000.0100.
13. Expliquez ce que sont les systèmes logiques combinatoires et les systèmes logiques séquentiels.
14. Dans les tables de vérité suivantes, quelles combinaisons des entrées produisent une sortie à 1 ?

(a)

Entrée A	Entrée B	Sortie Q
0	0	0
1	0	0
0	1	0
1	1	1

(b)

Entrée A	Entrée B	Sortie Q
0	0	0
1	1	1
0	1	1
1	1	1

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

(c)

Entrée A	Entrée B	Sortie Q
0	0	1
1	0	1
0	1	1
1	1	0

15. Dans un verrou D à entrée d'horloge, quelles entrées doivent changer pour que la sortie Q passe de 0 à 1 ?
16. Donnez la table de fonctionnement d'un verrou RS à entrée d'horloge lorsque l'entrée de S est également appliquée à R au travers d'une porte NON (elle convertit un 0 en un 1 et vice versa).

3.11 Recherches complémentaires

1. Recherchez les spécifications (a) d'une porte ET et (b) d'un verrou D transparent.

Copyright © 2015 Dunod.

4 • TRAITEMENT DES ENTRÉES-SORTIES

Ce chapitre revient sur le sujet du Chapitre 2, en s'intéressant au traitement des signaux issus des dispositifs d'entrées et de sorties. L'unité d'entrées-sorties apporte l'interface entre le contrôleur de l'API et le monde extérieur. Elle réalise un traitement pour que le signal ait la tension requise et met en place une protection contre les risques électriques, comme les surtensions. Ce chapitre s'articule autour des modules d'entrées-sorties classiques et, dans une installation où les capteurs peuvent être éloignés de l'API de traitement, à leurs liens de communication avec l'API.

4.1 Unités d'entrées-sorties

Les signaux d'entrée issus des capteurs et les sorties requises par les actionneurs peuvent être de trois types :

- *Analogiques*. Un signal dont l'amplitude est en rapport avec la taille de la quantité mesurée.
- *Discrets*. Principalement un signal tout ou rien.
- *Numériques*. Un train d'impulsions.

Le CPU a besoin de signaux numériques d'une certaine amplitude, en général entre 0 et 5 V. La sortie du CPU est un signal numérique dont le niveau est normalement entre 0 et 5 V. Par conséquent, il est souvent nécessaire de manipuler les signaux d'entrée et de sortie afin de leur donner la forme requise.

Les unités d'entrées-sorties des API sont conçues pour que différents signaux d'entrée puissent être convertis en signaux numériques 5 V et pour générer différents niveaux de sortie afin de commander les dispositifs externes. La facilité d'utilisation des API vient de cette faculté à prendre en charge une grande diversité d'entrées et de sorties. Les sections suivantes décrivent les circuits de base employés par les unités d'entrées-sorties. Dans le cas des systèmes modulaires, ils sont montés sur des cartes, qui peuvent être insérées dans les racks. Les caractéristiques des entrées-sorties de l'API peuvent ainsi évoluer par simple remplacement des cartes. Dans un API en boîtier autonome, le fabricant a inclus les unités d'entrées-sorties.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

4.1.1 Unités d'entrées

Les termes *fourniture* et *absorption* font référence à la manière dont les dispositifs à courant continu s'interfacent avec l'API (voir Section 1.3.2). Dans le cas d'une unité d'entrées à fourniture de courant, elle fournit le courant d'alimentation au dispositif d'entrée qui lui est connecté (voir Figure 4.1a). Dans le cas de l'absorption, le dispositif d'entrée fournit le courant à l'unité d'entrées (voir Figure 4.1b).

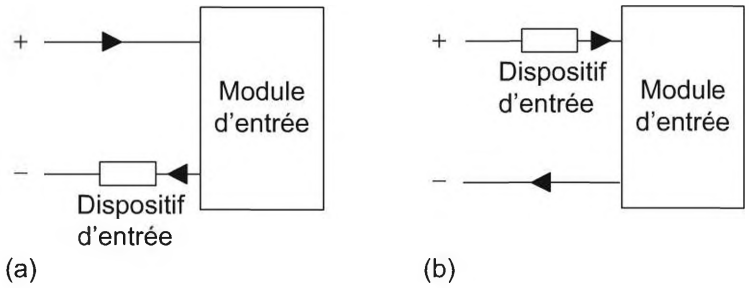


Figure 4.1 – Les entrées : (a) à fourniture et (b) à absorption de courant.

Les Figures 4.2 et 4.3 dévoilent les circuits de base utilisés pour les entrées à courant continu et à courant alternatif. Des photocoupleurs (voir Section 1.3.1) sont employés pour la protection. Dans le cas d'une unité d'entrée CA, un pont redresseur convertit le courant alternatif en un signal CC, qui est utilisé par le photocoupleur pour produire les signaux d'entrée envoyés au CPU de l'API. Des témoins lumineux sont associés à chaque entrée pour indiquer si le dispositif d'entrée fournit un signal.

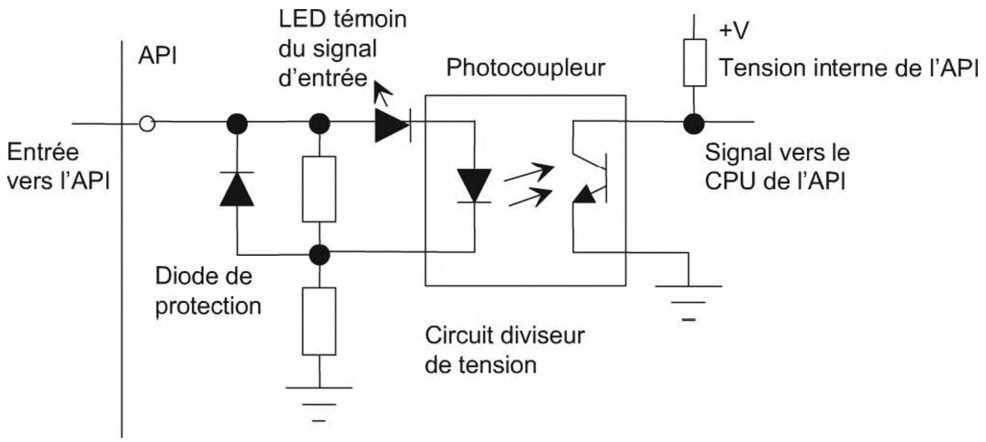


Figure 4.2 – Unité d'entrées à courant continu.

Lorsque des signaux analogiques sont envoyés à un API, le canal d'entrée doit les convertir en signaux numériques à l'aide d'un convertisseur analogique-numérique. Dans un système modulaire, il suffit d'insérer la carte d'entrées analogiques adaptée dans le rack. Pour ne pas avoir à ajouter une carte pour chaque entrée analogique, un multiplexage est généralement mis en œuvre (voir Figure 4.4).

Copyright © 2015 Dunod.

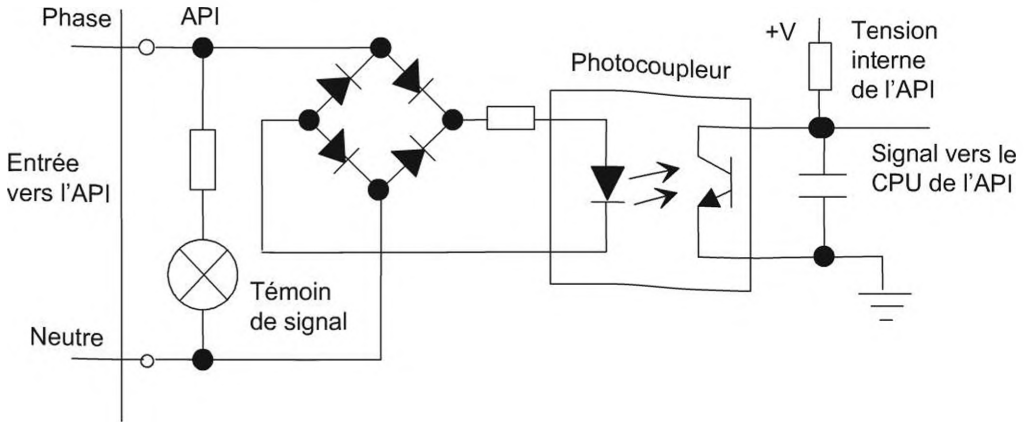


Figure 4.3 – Unité d'entrées à courant alternatif.

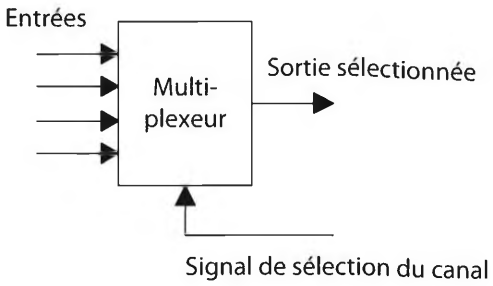


Figure 4.4 – Un multiplexeur.

Dans ce cas, plusieurs entrées analogiques sont connectées à la carte et des commutateurs électroniques permettent ensuite de sélectionner chaque entrée tour à tour. En général, les cartes disponibles proposent quatre, huit ou seize entrées analogiques.

La Figure 4.5a illustre le fonctionnement d'un convertisseur analogique-numérique (CAN). Un seul signal d'entrée analogique donne lieu à des signaux de sortie tout ou rien sur, par exemple, huit fils séparés. Les huit signaux constituent le *mot* numérique qui correspond au niveau du signal d'entrée analogique. Avec un tel convertisseur huit bits, il existe $2^8 = 256$ valeurs de signal possibles ; elles vont de 0000 0000 à 1111 1111, c'est-à-dire de 0 à 255. Le niveau du signal de sortie monte par palliers (voir Figure 4.5b). Les tensions analogiques qui produisent chaque sortie numérique sont appelées *niveaux de quantification*.

Pour que la sortie binaire change, la tension analogique doit changer conformément à la différence de tension analogique entre les niveaux successifs. Le terme *résolution* désigne le plus petit changement de tension analogique qui donne lieu à une modification d'un bit dans la sortie numérique. Avec un CAN huit bits, lorsque la variation du signal d'entrée analogique est dans la plage 0 à 10 V, une avancée d'un bit numérique implique un changement de 10/255 V sur l'entrée analogique, c'est-à-dire environ 0,04 V. Autrement dit, un changement de 0,03 V sur l'entrée analogique ne produit aucun changement sur la sortie numérique. Le nombre de bits de la sortie d'un convertisseur analogique-numérique détermine donc sa *résolution* et,

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

par conséquent, sa *précision*. Avec un CAN dix bits, il est possible d'obtenir $2^{10} = 1\,024$ valeurs numériques différentes et, pour une entrée analogique dans la plage 0 à 10 V, un pas d'un bit numérique implique un changement de 10/1 023 V, c'est-à-dire environ 0,01 V. Avec un CAN douze bits, le nombre de valeurs numériques possibles atteint $2^{12} = 4\,096$ et, pour une plage d'entrée analogique entre 0 et 10 V, un pas d'un bit correspond à un changement de 10/4 095 V, c'est-à-dire environ 2,4 mV. En général, la résolution d'un convertisseur analogique-numérique n-bits est de $1/(2^n - 1)$, que l'on approxime parfois à 2^{-n}.

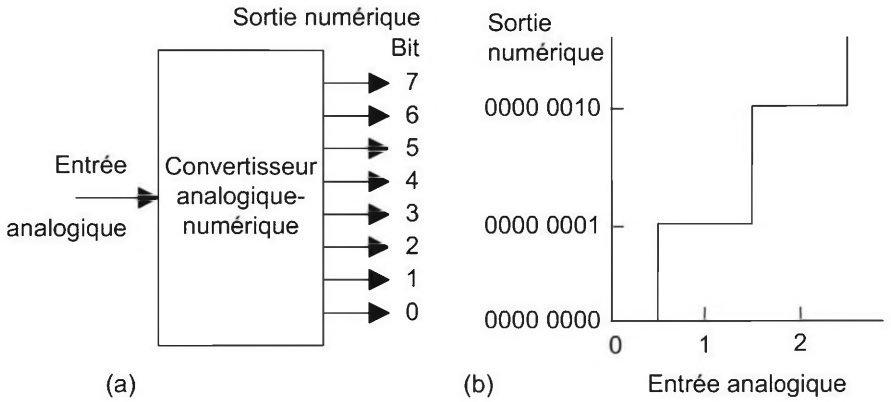


Figure 4.5 – (a) Fonctionnement d'un convertisseur analogique-numérique et (b) rapport entre l'entrée analogique et la sortie numérique.

Le tableau suivant illustre la conversion analogique-numérique d'une entrée analogique dans la plage 0 à 10 V à l'aide d'un convertisseur huit bits :

Entrée analogique (V)	Sortie numérique (V)
0,00	0000 0000
0,04	0000 0001
0,08	0000 0010
0,12	0000 0011
0,16	0000 0100
0,20	0000 0101
0,24	0000 0110
0,28	0000 0111
0,32	0000 1000
etc.	

Copyright © 2015 Dunod.

Pour mettre en pratique cette idée, prenons un thermocouple utilisé comme capteur avec un API et qui produit une sortie de 0,5 mV par °C. Quelle sera la précision avec laquelle l'API activera le dispositif de sortie si le thermocouple est connecté à une entrée analogique dont la plage de fonctionnement est de 0 à 10 V CC et si le convertisseur analogique-numérique est sur dix bits ? Avec un convertisseur dix bits, nous disposons de $2^{10} = 1\,024$ bits pour couvrir la plage 0 à 10 V. Par conséquent, chaque changement d'un bit correspond à un changement de 10/1 023 V, c'est-à-dire 10 mV. La précision de reconnaissance de l'entrée issue du thermocouple par l'API est donc de ±5 mV ou ±10 °C.

La conversion analogique-numérique prend du temps. Par ailleurs, en raison du multiplexeur, la carte d'entrées analogiques peut uniquement prendre des échantillons des signaux d'entrée. Dans la plupart des systèmes industriels, les signaux produits par les machines varient lentement et ce fonctionnement ne pose donc pas de problème. Les temps de conversion sont généralement de quelques millisecondes.

4.1.2 Unités de sorties

Lorsqu'une unité de sorties d'un API fournit le courant au dispositif de sortie (voir Figure 4.6a), il s'agit d'une unité à *fourniture* de courant. Lorsque le courant vient du dispositif de sortie, l'unité est dite à *absorption* de courant (voir Figure 4.6b). Très souvent, les unités d'entrées à absorption de courant sont employées comme interfaces avec du matériel électronique et les unités de sortie à fourniture de courant, comme interfaces avec des solénoïdes.

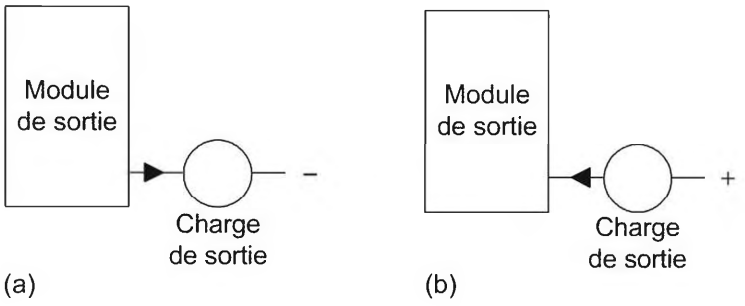


Figure 4.6 – Les sorties : (a) à fourniture et (b) à absorption de courant.

Les unités de sorties peuvent être à relais, à transistor ou à triac. La Figure 4.7 présente le circuit de base d'une unité de sorties à relais, la Figure 4.8, d'une unité de sorties à transistor et la Figure 4.9, d'une unité de sorties à triac.

Les sorties analogiques sont fréquemment utilisées et peuvent être fournies par des convertisseurs numérique-analogique (CNA) sur le canal de sortie. L'entrée du convertisseur est un ensemble de bits individuels. La Figure 4.10 illustre le fonctionnement de base d'un tel convertisseur.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

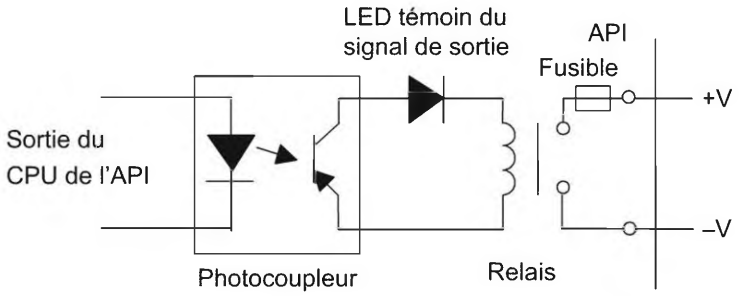


Figure 4.7 – Unité de sorties à relais.

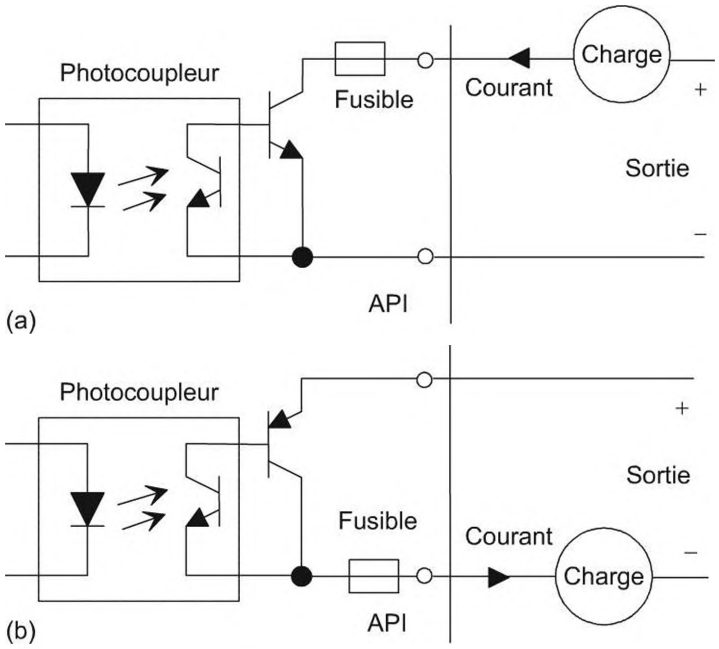


Figure 4.8 – Circuits de base d'une sortie à transistor : (a) à absorption et (b) à fourniture de courant.

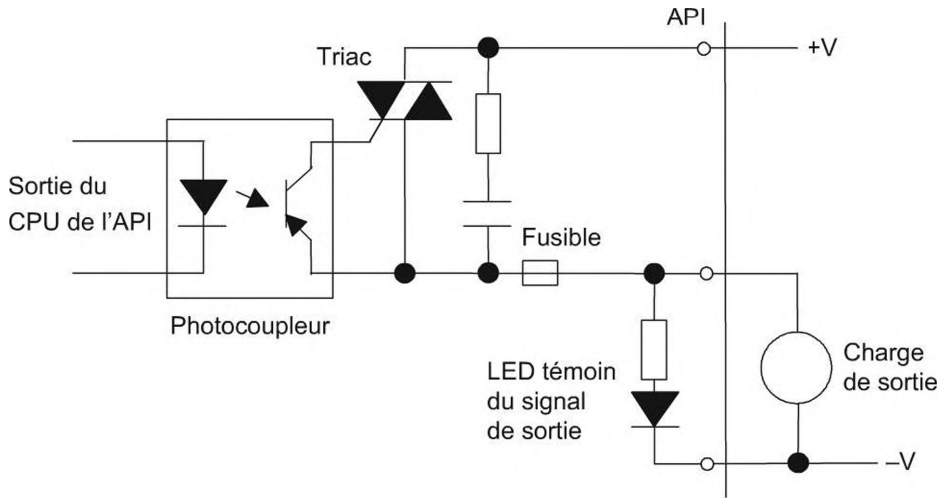


Figure 4.9 – Unité de sorties à triac.

Copyright © 2015 Dunod.

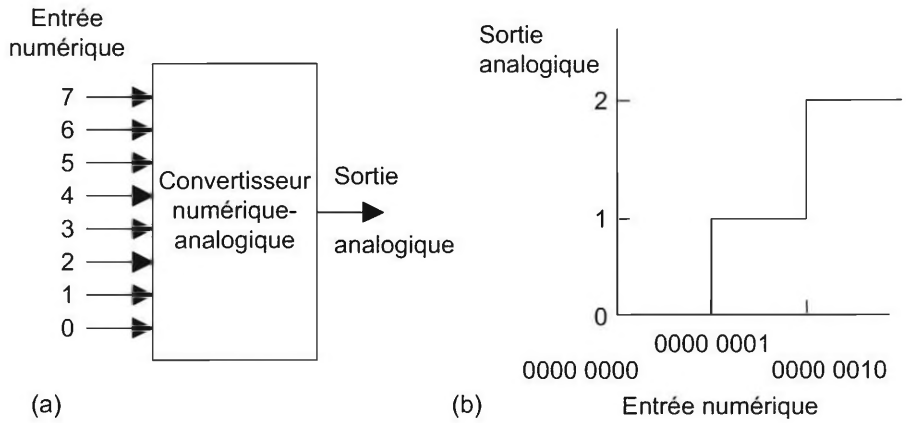


Figure 4.10 – (a) Fonctionnement d'un convertisseur numérique-analogique et (b) conversion numérique-analogique.

Un bit sur la ligne 0 produit une impulsion de sortie d'une certaine amplitude. Un bit sur la ligne 1 produit une impulsion de sortie dont l'amplitude est deux fois supérieure à celle de l'impulsion due à un bit sur la ligne 0. Un bit sur la ligne 2 produit une impulsion de sortie dont l'amplitude est deux fois supérieure à celle de l'impulsion due à un bit sur la ligne 1. Un bit sur la ligne 3 donne lieu à une impulsion de sortie dont l'amplitude est deux fois supérieure à celle de l'impulsion due à un bit sur la ligne 2. Etc. Toutes les sorties s'additionnent pour donner la version analogique de l'entrée numérique. Lorsque l'entrée numérique change, la sortie analogique change par palliers de tension associée à chaque bit. Par exemple, avec un convertisseur huit bits, la sortie est constituée de $2^8 = 256$ palliers. Supposons que la plage de sortie soit fixée à 10 V CC. Un bit produit une variation de 10/255 V, c'est-à-dire environ 0,04 V. Le tableau suivant illustre cette conversion :

Entrée numérique (V)	Sortie analogique (V)
00000000	0,00
00000001	0,04
00000010	0,08 + 0,00 = 0,08
00000011	0,08 + 0,04 = 0,12
00000100	0,16
00000101	0,016 + 0,00 + 0,04 = 0,20
00000110	0,016 + 0,08 = 0,24
00000111	0,016 + 0,08 + 0,04 = 0,28
00001000	0,32
etc.	

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

Les modules de sorties analogiques sont généralement fournis avec un certain nombre de sorties, par exemple quatre à 20 mA, de 0 à +5 V CC, et de 0 à +10 V CC, et la sortie appropriée est choisie par des interrupteurs. Les modules proposent généralement deux formes de sorties. Dans la première, toutes les sorties du module ont une alimentation commune, dans la seconde, chacune possède sa propre alimentation. La Figure 4.11 illustre les principes de base de ces deux variantes.

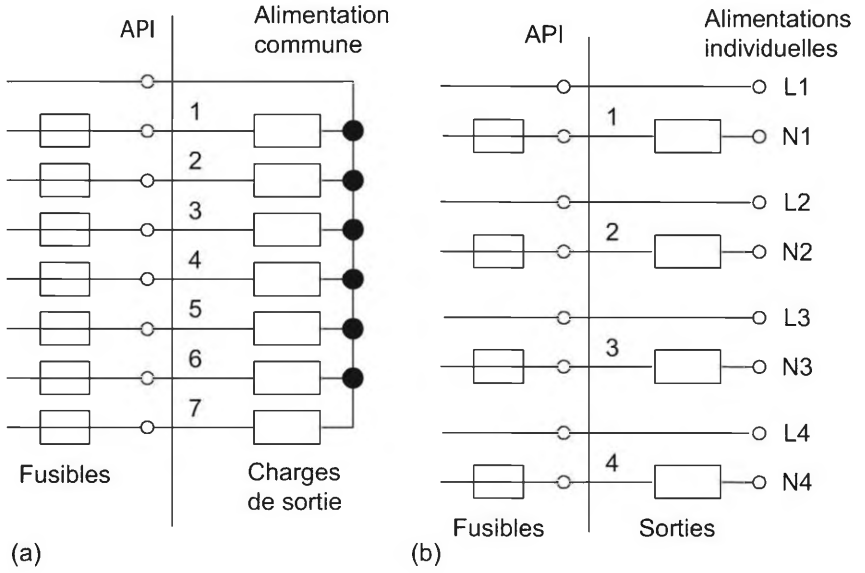


Figure 4.11 – Variantes de sorties : (a) alimentation commune et (b) alimentations individuelles.

4.2 Traitement du signal

Pour connecter des capteurs qui génèrent des signaux numériques ou discrets à une unité d'entrées, il faut vérifier que les niveaux de tension correspondent. Cependant, de nombreux capteurs génèrent des signaux analogiques. Pour prendre en charge la grande diversité de signaux analogiques que peuvent produire les capteurs, une multiplication des canaux d'entrées analogiques risque d'être nécessaire. Pour l'éviter, un traitement du signal externe est souvent réalisé, afin que les signaux analogiques se trouvent tous dans la même plage et qu'une forme standard de canal d'entrée analogique puisse être utilisée.

Une méthode standard (voir Figure 4.12) consiste à convertir les signaux analogiques en un courant dans la plage 4 à 20 mA, puis en une tension par son passage dans une résistance de 250 Ω de manière à obtenir un signal d'entrée entre 1 et 5 V. Par exemple, un capteur utilisé pour surveiller le niveau du liquide entre 0 et 1 m peut représenter le niveau 0 par un courant de 4 mA et le niveau 1 m, par un courant de 20 mA. Un courant de 4 mA est utilisé pour représenter la valeur basse d'une plage analogique de manière à pouvoir faire la différence entre un capteur qui indique zéro et un capteur qui ne fonctionne pas et qui ne produit donc aucune réponse. Dans ce cas, le courant est de 0 mA. Un courant de 4 mA est

Copyright © 2015 Dunod.

également souvent adapté au fonctionnement d'un capteur et permet d'éviter l'ajout d'une alimentation séparée.

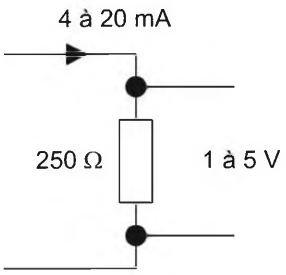


Figure 4.12 – Signal analogique standard.

4.2.1 Changer les niveaux de tension

Un diviseur de tension (voir Figure 4.13) permet de réduire la tension produite par un capteur jusqu'au niveau requis. La tension de sortie V_{out} est :

$$V_{out} = \frac{R_2}{R_1 + R_2} V_{in}$$

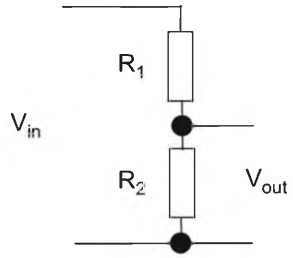


Figure 4.13 – Diviseur de tension.

Des amplificateurs peuvent également servir à augmenter le niveau de tension. La Figure 4.14 présente les circuits de base fondés sur un amplificateur opérationnel 741 monté (a) en amplificateur inverseur et (b) en amplificateur non-inverseur. Dans le montage en amplificateur inverseur, la tension de sortie V_{out} est :

$$V_{out} = -\frac{R_2}{R_1} V_{in}$$

Dans le cas de l'amplificateur non-inverseur, la formule de V_{out} devient :

$$V_{out} = \frac{R_1 + R_2}{R_1} V_{in}$$

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

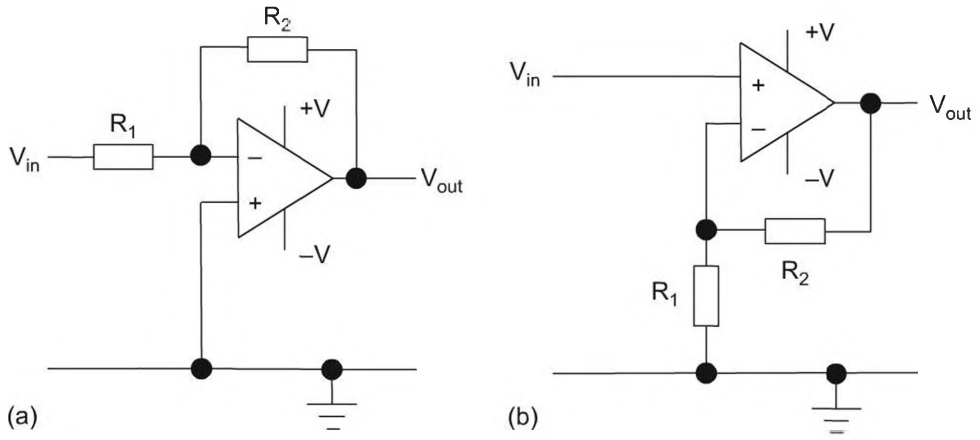


Figure 4.14 – Circuits à amplificateurs opérationnels.

Un amplificateur différentiel doit souvent être utilisé pour amplifier la différence entre deux tensions d'entrée. C'est notamment le cas lorsqu'un capteur, par exemple un extensomètre, est inséré dans un pont de Wheatstone et que la sortie correspond à la différence entre deux tensions, ou avec un thermocouple dont la différence de tension entre les jonctions chaude et froide doit être mesurée. La Figure 4.15 présente le schéma de base d'un amplificateur opérationnel employé de cette manière.

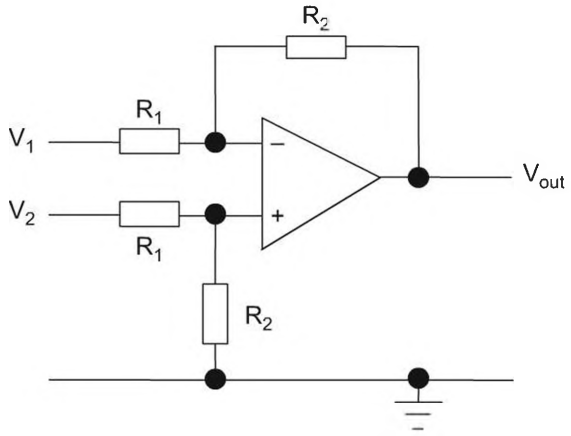


Figure 4.15 – Amplificateur différentiel.

La tension de sortie V_{out} est :

$$V_{out} = \frac{R_2}{R_1}(V_2 - V_1)$$

La Figure 4.16 illustre l'utilisation du traitement du signal à partir d'un schéma qui peut être utilisé avec un extensomètre. Le capteur est inséré dans un pont de Wheatstone et la différence de potentiel est amplifiée par un amplificateur différentiel, avant d'être transmise à un convertisseur analogique-numérique, qui fait partie du port d'entrée analogique de l'API.

Copyright © 2015 Dunod.

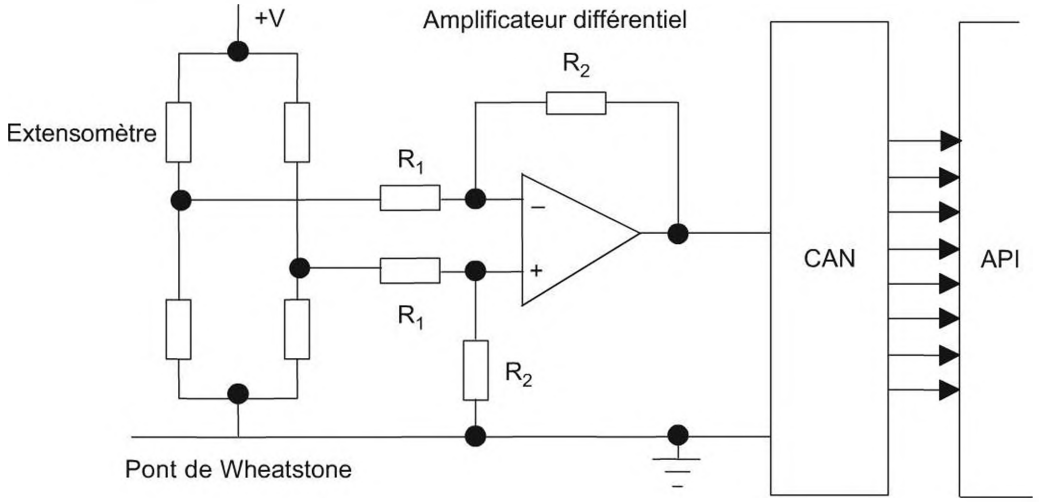


Figure 4.16 – Traitement du signal avec un extensomètre.

4.2.2 Comparateur à amplificateur opérationnel

La sortie d'un amplificateur opérationnel sature autour de ±12 V, une telle tension étant atteinte pour des entrées d'environ ±10 µV (voir Figure 4.17). C'est en raison de cette caractéristique que les amplificateurs opérationnels sont largement utilisés pour obtenir des signaux tout ou rien en fonction de la valeur relative de deux signaux d'entrée. Un signal est connecté à l'entrée non inverseuse, tandis que l'autre est connecté à l'entrée inverseuse. L'amplificateur opérationnel détermine si le niveau du signal sur l'entrée inverseuse est au-dessus ou en dessous de celui appliqué à l'entrée non inverseuse. Il fonctionne donc en comparateur. En échangeant la tension de référence et la tension d'entrée, la polarité de la sortie est modifiée. Ces comparateurs peuvent servir de base aux systèmes de commande tout ou rien. Nous pouvons appliquer une tension de référence et la comparer à celle fournie par un capteur de manière à obtenir une sortie tout ou rien qui bascule selon que la tension du capteur est supérieure ou inférieure à la tension de référence.

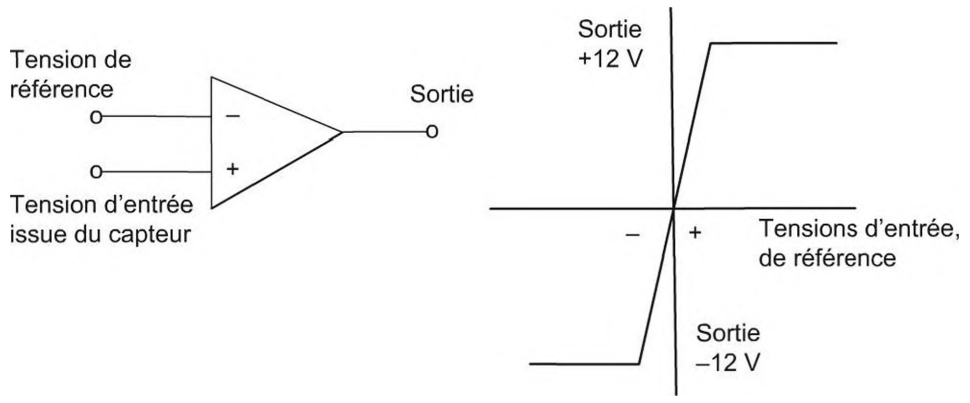


Figure 4.17 – Caractéristique d'un amplificateur opérationnel.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

4.2.3 Protection des sorties

Les sorties qui mettent en œuvre des bobines de fils, comme les solénoïdes et les moteurs, forment des inducteurs. Lorsqu'un courant les traverse, de tels inducteurs génèrent un champ magnétique qui, lorsque le courant est coupé, s'effondre et produit une force contre-électromotrice (FCEM) potentiellement importante. La méthode la plus simple pour se protéger contre cette FCEM consiste à placer une diode en parallèle avec la bobine. Cette diode est connectée de sorte que le courant ne la traverse pas lorsqu'il alimente la bobine. En revanche, elle court-circuite la bobine et supprime le courant lorsqu'il est produit par la FCEM. Une telle diode est souvent appelée *diode de roue libre*.

Certains dispositifs de sortie peuvent avoir besoin d'une résistance de limitation du courant. Par exemple, une LED accepte généralement un courant maximal situé entre 10 et 30 mA. Avec 20 mA, la baisse de tension dans la LED peut être de 2,1 V. Par conséquent, si la sortie applique à la LED une tension de 5 V, il faut l'abaisser de 2,9 V par une résistance en série. Cette résistance doit avoir une valeur de 2,9/0,020 = 145 Ω ; une résistance standard de 150 Ω peut donc être utilisée. Dans certaines LED, la résistance en série est déjà intégrée.

4.3 Connexions distantes

Lorsque de nombreuses entrées ou sorties sont très éloignées de l'API, il est possible de tirer des câbles entre chaque dispositif et l'API, mais une solution plus économique consiste à placer des modules d'entrées-sorties à côté des entrées et des sorties et à utiliser un câble monoconducteur qui les relie à l'API, sur une longue distance, à la place d'un câble multiconducteur qui serait nécessaire sans ces modules d'entrées-sorties distants (voir Figure 4.18).

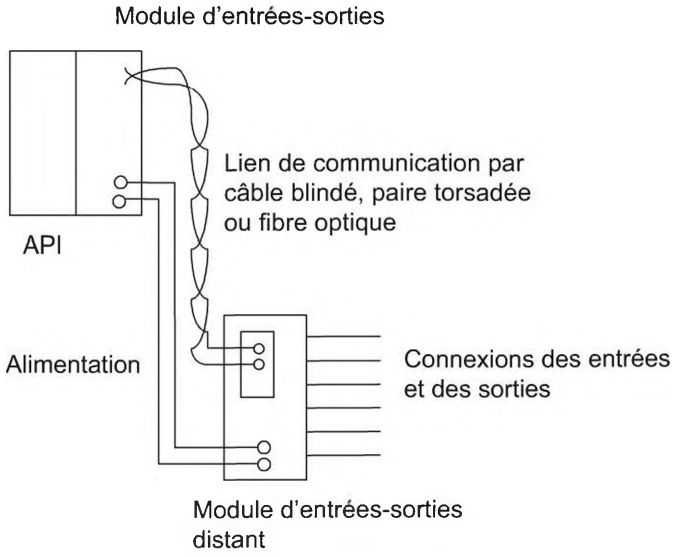


Figure 4.18 – Utilisation d'un module d'entrées-sorties distant.

Copyright © 2015 Dunod.

Dans certains cas, plusieurs API peuvent être reliés à un API maître, qui envoie et reçoit des données d'entrées-sorties des autres unités (voir Figure 4.19). Les API distants ne disposent pas du programme de commande car tout le traitement est réalisé sur l'API maître.

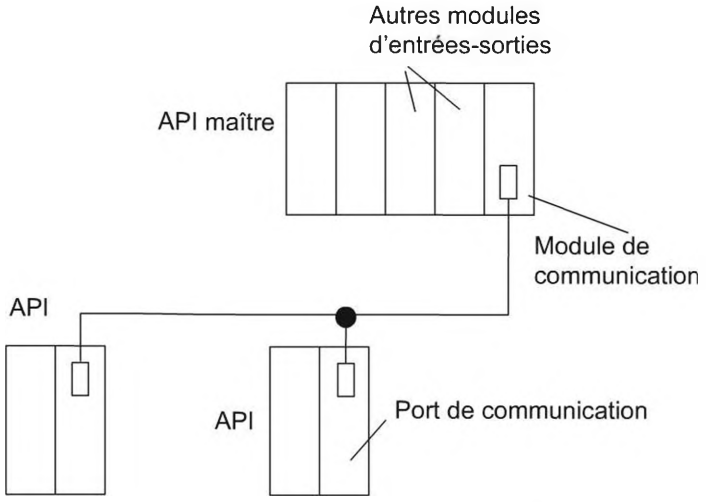


Figure 4.19 – Utilisation d'API pour des entrées-sorties distantes.

Les câbles employés pour l'échange des données entre les modules d'entrées-sorties distants et un API central, des API distants et un API maître sont généralement des *câbles à paires torsadées*, placés dans un tube métallique relié à la terre pour réduire les effets des « bruits » électriques. Un *câble coaxial* permet d'obtenir des débits plus élevés et n'a pas besoin d'un tube métallique de protection. Une *fibre optique* présente l'avantage d'offrir une plus grande résistance aux bruits, d'être de taille plus réduite et d'avoir une grande souplesse. C'est pourquoi elles sont de plus en plus utilisées.

4.3.1 Communications série et parallèle

Dans une *communication série*, les données sont transmises un bit à la fois (voir Figure 4.20a). Par conséquent, si un mot de huit bits doit être envoyé, les huit bits sont transmis un par un en série le long d'un câble. Cela signifie qu'un mot de données doit être décomposé en une suite de bits avant d'être transmis, pour ensuite être réassemblé après réception des bits. Dans une *communication parallèle*, tous les bits qui composent un mot sont transmis simultanément le long de câbles en parallèle (voir Figure 4.20b). Cela permet d'envoyer des données à très haute vitesse sur de courtes distances.

La communication série est utilisée pour transmettre les données sur de longues distances. Il est moins coûteux de tirer un câble monoconducteur sur une longue distance pour une communication série que d'utiliser un câble multiconducteur nécessaire à une communication parallèle. Dans un système à base d'API, une communication série peut être utilisée pour connecter un ordinateur, qui sert de

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

terminal de programmation, à l'API. Une communication parallèle peut être employée pour connecter des instruments de laboratoire au système. En interne, les API utilisent des communications parallèles en raison de leur vitesse. Des circuits émetteurs-récepteurs asynchrones universels (UART, *Universal Asynchronous Receivers/Transmitters*) sont utilisés au niveau des ports d'entrées-sorties pour convertir les signaux des communications série en signaux parallèles.

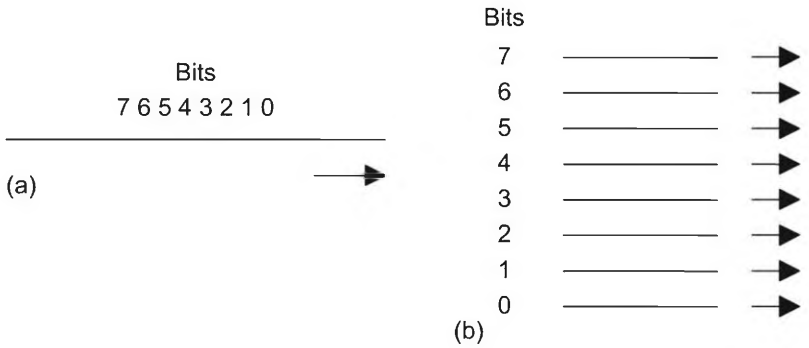


Figure 4.20 – (a) Communication série et (b) communication parallèle.

4.3.2 Normes de communication série

Pour que les communications série puissent se faire correctement, il est nécessaire de préciser les points suivants :

- Les niveaux de tension utilisés pour les signaux, c'est-à-dire quel niveau représente un 0 et lequel représente un 1.
- La signification des séquences de bits transmis et la manière de construire un message. N'oubliez pas que les différents mots sont envoyés sur un même câble et qu'il faut donc un moyen pour déterminer le début et la fin d'un mot, ainsi que le début du mot suivant.
- La vitesse à laquelle la séquence de bits est envoyée, c'est-à-dire le nombre de bits par seconde.
- La synchronisation des horloges à chaque extrémité. Ce point se révèle nécessaire si, par exemple, une certaine durée de l'impulsion transmise doit être reconnue par le récepteur comme un seul bit à la place de deux.
- Les protocoles, ou contrôles de flux, pour échanger des informations comme « en mesure de recevoir des données » ou « non prêt à recevoir des données ». Pour cela, deux fils de signalisation supplémentaires, appelés *fils de dialogue*, sont utilisés, l'un pour indiquer au récepteur que l'émetteur est prêt à envoyer des données et l'autre pour indiquer à l'émetteur que le récepteur est prêt à les recevoir.
- Le contrôle d'erreur pour qu'une séquence de bits puisse être vérifiée afin de détecter une corruption éventuelle des données au cours de leur transmission.

L'interface de communication série la plus utilisée se nomme *RS232*. Les connexions se font par des connecteurs de type DB à 25 broches (voir

Copyright © 2015 Dunod.

Figure 4.21), avec, même si ce n'est pas toujours le cas, une prise mâle sur le câble et une prise femelle sur le matériel. Toutes les broches ne sont pas forcément utilisées dans toutes les applications.

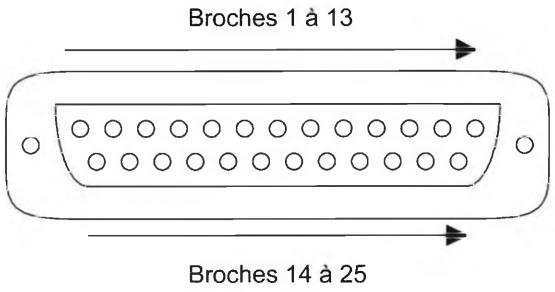


Figure 4.21 – Connecteur DB-25.

Les broches suivantes sont obligatoirement utilisées :

Broche 1 : connexion à la masse du châssis

Broche 2 : émission des données série (broche de sortie des données)

Broche 3 : réception des données série (broche d'entrée des données)

Broche 7 : masse électrique, qui sert de chemin commun de retour du signal

Lorsque l'interface est employée avec des ordinateurs, les broches suivantes sont généralement utilisées :

Broche 1 : connexion à la masse du châssis

Broche 2 : émission des données série (broche de sortie des données)

Broche 3 : réception des données série (broche d'entrée des données)

Broche 4 : demande d'émission (RTS, *Request To Send*)

Broche 5 : prêt à émettre (CTS, *Clear To Send*)

Broche 6 : poste de données prêt (DSR, *Data Set Ready*)

Broche 7 : masse électrique, qui sert de chemin commun de retour du signal

Broche 20 : terminal de données prêt (DTR, *Data Terminal Ready*)

Les signaux des broches 4, 5, 6 et 20 sont utilisés pour vérifier que l'extrémité de réception est prête à recevoir un signal, que l'extrémité d'émission est prête à envoyer et que les données sont prêtes à être envoyées. Dans la norme RS232, un bit 1 est représenté par une tension entre −25 et −5 V, habituellement −12 V, tandis qu'un bit 0 est représenté par une tension entre +5 et +25 V, habituellement +12 V.

Le *débit en bauds* décrit la vitesse de transmission et équivaut approximativement au nombre de bits transmis ou reçus par seconde. Cependant, tous les bits transmis ne sont pas utilisés pour les données. Certains indiquent le début et la fin d'un élément de données série et sont souvent appelés *fanions*. D'autres servent à vérifier si les données ont été corrompues pendant la transmission. La Figure 4.22 montre le type de signal qui peut être envoyé par une interface RS232. Le bit de

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

parité est ajouté pour détecter la corruption des données ; dans le cas d'une parité paire, un bit à 1 est ajouté de manière à obtenir un nombre pair de 1. Pour envoyer sept bits de données, il faut donc transmettre onze bits.

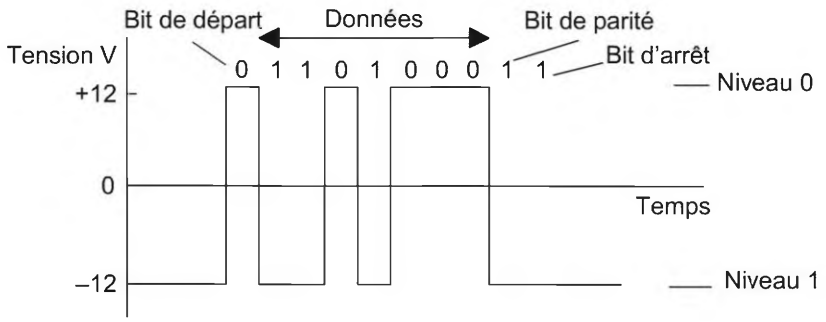


Figure 4.22 – Niveaux des signaux RS232.

D'autres normes, comme *RS422* et *RS423*, sont comparables à la norme RS232. RS232 présente des limites sur la longueur des câbles qui peuvent être utilisés, car le bruit empêche les transmissions à un débit élevé lorsque la longueur du câble dépasse 15 m. RS422 peut être employée pour des distances plus importantes. Cette technologie se fonde sur une méthode de transmission équilibrée. Elle utilise deux lignes pour la transmission, le signal transmis étant une différence de tension entre les deux lignes. Puisque le bruit affecte les deux lignes de la même manière, il n'a aucun effet sur le signal transmis. La Figure 4.23 montre comment, pour les interfaces RS232 et RS422, le débit de données atteignable sans que le bruit affecte la transmission dépend de la distance. Les lignes RS422 peuvent être utilisées sur des distances plus importantes que les lignes RS232.

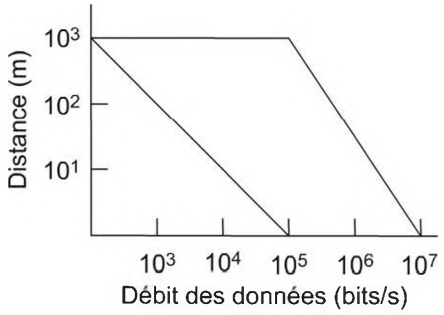


Figure 4.23 – Transmission selon les normes RS232 et RS422.

RS422 se fonde sur une interface équilibrée dans laquelle un signal à 1 correspond à une différence de tension entre deux lignes et un signal à 0 correspond à cette même différence de tension de polarité inverse. RS423 utilise une interface non équilibrée dans laquelle la différence entre la tension sur la ligne et une tension figée, en général la masse, permet de déterminer les signaux à 1 et à 0. Elle utilise une ligne pour les signaux transmis et une autre pour les signaux reçus. RS423 ne

Copyright © 2015 Dunod.

permet pas une transmission aussi rapide que RS422, mais les distances de transmission sont comparables. Certains dispositifs peuvent être configurés de manière à offrir une interface RS422 ou une interface RS423.

La *boucle de courant 20 mA* est une alternative aux interfaces RS422 et RS423. Il s'agit d'une norme antérieure, qui est toujours employée pour les communications série sur de longues distances, particulièrement dans les systèmes industriels où le chemin de communication peut subir les effets des bruits électriques (voir Figure 4.24). Cette méthode permet de transférer des données sur des distances de quelques kilomètres. Elle comprend un circuit, une boucle filaire, contenant une source de courant. Les données série sont transmises par commutation du courant, un 0 étant transmis par l'absence de courant et un 1, par un courant de 20 mA. Pour les communications bidirectionnelles, un couple de fils séparés est employé dans des boucles d'émission et de réception. Les données série sont codées avec un bit de départ, le nombre de bits de données adaptés et deux bits d'arrêt.

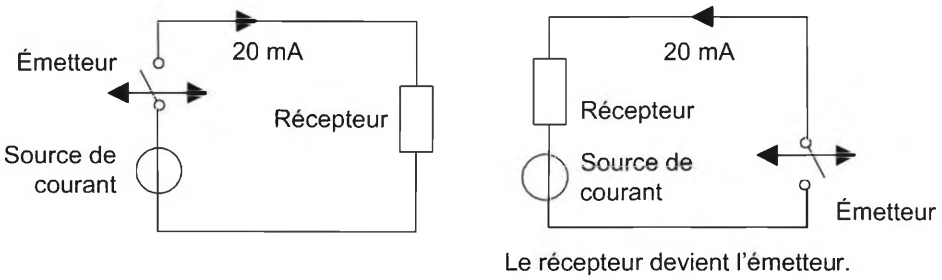


Figure 4.24 – Boucle 20 mA.

D'autres types de bus sont souvent employés dans certaines situations particulières. Il s'agit notamment du bus I^2C (*Inter-IC Communication*) conçu par Philips pour les communications entre les circuits intégrés ou les modules, du bus CAN (*Controller Area Network*) développé par Bosch pour les systèmes de gestion des moteurs de voitures, du bus USB (*Universal Serial Bus*) conçu pour connecter facilement les moniteurs, les imprimantes et d'autres périphériques aux ordinateurs et du bus Firewire développé par Apple Computers pour apporter des possibilités plug-and-play aux ordinateurs.

4.3.3 Normes de communication parallèle

L'interface standard la plus employée pour les communications parallèles est l'interface *IEEE-488*. Elle a été conçue à l'origine par Hewlett-Packard pour interconnecter ses ordinateurs et ses instruments. Si elle se nommait initialement *Hewlett-Packard Instrumentation Bus*, elle est à présent plus généralement appelée *General-Purpose Instrument Bus* (GPIB). Ce bus permet des interconnexions dans lesquelles les échanges de données en parallèle se font entre des écouteurs, des parleurs et des contrôleurs. Les *écouteurs* sont des appareils qui acceptent des données provenant du bus. Les *parleurs* placent des données sur le bus à la demande. Les *contrôleurs*

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

gèrent le flux des données sur le bus et apportent des fonctions de traitement. Un bus comprend vingt-quatre lignes, dont huit lignes bidirectionnelles servent au transport des données et des commandes entre les différents appareils connectés, cinq lignes sont utilisées pour les signaux de contrôle et d'état, trois participent au dialogue entre les appareils et huit sont des lignes de masse (voir Figure 4.25). Le *dialogue* (*handshaking*) désigne le transfert des informations de contrôle, comme les signaux DATA READY et INPUT ACKNOWLEDGED, entre deux appareils.

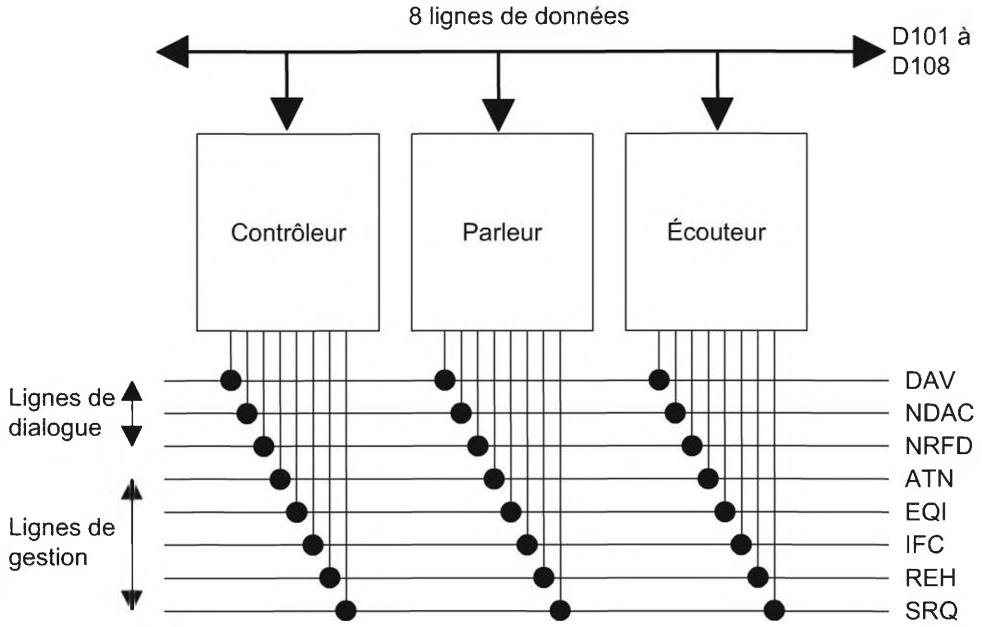


Figure 4.25 – Structure du bus IEEE-488.

Les commandes émises par le contrôleur sont signalées en plaçant la ligne ATN (ATTENTION) au niveau bas ; lorsqu'elle est au niveau haut, elle indique que les lignes de données convoient des données. Les commandes peuvent être dirigées vers des appareils individuels en plaçant des adresses sur les lignes de données. Chaque appareil connecté au bus possède sa propre adresse. Les adresses sont envoyées en parallèle sur les lignes de données sous forme d'un mot de sept bits, les cinq bits de poids faible précisant l'adresse de l'appareil et les deux autres apportant des informations de contrôle. Si tous ces bits sont à 0, les commandes sont envoyées à toutes les adresses. Si le bit 6 est à 1 et le bit 7 à 0, l'appareil adressé devient un écouteur. Si le bit 6 est à 0 et le bit 7 à 1, l'appareil devient un parleur.

Comme l'illustre le fonctionnement de la ligne ATN, chaque ligne de gestion possède son propre rôle dans le contrôle des informations. Les lignes de dialogue servent au contrôle du transfert des données. Elles permettent de garantir que le parleur ne parlera que si des écouteurs l'écoutent. Le Tableau 4.1 recense les fonctions de toutes les lignes et leurs numéros de broche dans le connecteur RIBBON à vingt-quatre broches.

Copyright © 2015 Dunod.

Tableau 4.1 Le bus IEEE-448

Broche	Groupe	Abréviation	Signal/fonction
1	Données	D101	Ligne de données 1.
2	Données	D102	Ligne de données 2.
3	Données	D103	Ligne de données 3.
4	Données	D104	Ligne de données 4.
5	Gestion	EOI	Fin de transmission (*End Or Identify*). Cette ligne est utilisée pour indiquer la fin d'une séquence de messages émise par un parleur ou elle est utilisée par le contrôleur pour demander à un appareil de s'identifier.
6	Dialogue	DAV	Validation des données (*Data Valid*). Lorsque cette ligne est au niveau bas, les informations présentes sur le bus de données sont valides et acceptables.
7	Dialogue	NRFD	Non prêt à recevoir des données (*Not Ready For Data*). Cette ligne est utilisée par les écouteurs, qui la mettent au niveau haut pour indiquer qu'ils sont prêts à accepter des données.
8	Dialogue	NDAC	Données non acceptées (*Not Data Accepted*). Cette ligne est utilisée par les écouteurs, qui la mettent au niveau haut pour indiquer que les données sont acceptées.
9	Gestion	IFC	Initialisation (*Interface Clear*). Cette ligne est utilisée par le contrôleur pour réinitialiser tous les appareils du système dans leur état de départ.
10	Gestion	SRQ	Demande de service (*Service Request*). Cette ligne est utilisée par les appareils pour signaler au contrôleur qu'ils requièrent l'attention.
11	Gestion	ATN	Attention. Cette ligne est utilisée par le contrôleur pour signaler qu'il va placer une commande sur les lignes de données.
12		SHIELD	Protection.
13	Données	D105	Ligne de données 5.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

Tableau 4.1 Le bus IEEE-448 *(suite)*

Broche	Groupe	Abréviation	Signal/fonction
14	Données	D106	Ligne de données 6.
15	Données	D107	Ligne de données 7.
16	Données	D108	Ligne de données 8.
17	Gestion	REN	Appareil sous contrôle du bus (*Remote Enable*). Cette ligne permet à un appareil sur le bus d'indiquer qu'il est configuré pour un contrôle à distance, non plus depuis son panneau de contrôle.
18		GND	Masse/commun.
19		GND	Masse/commun.
20		GND	Masse/commun.
21		GND	Masse/commun.
22		GND	Masse/commun.
23		GND	Masse/commun.
24		GND	Masse/commun.

La Figure 4.26 illustre le dialogue qui se produit lorsque des données sont placées sur les lignes de données. Initialement, la ligne DAV est au niveau haut, pour indiquer que le bus ne contient aucune donnée valide, tandis que NRFD et NDAC sont au niveau bas. Lorsqu'un mot de données est placé sur les lignes de données, NRFD passe au niveau haut, pour indiquer que tous les écouteurs sont prêts à accepter des données, et DAV passe au niveau bas, pour indiquer que de nouvelles données se trouvent sur le bus. Lorsqu'un appareil accepte un mot de données, il place la ligne NDAC au niveau haut, pour indiquer qu'il accepte les données, et la ligne NRFD au niveau bas, pour indiquer qu'il n'est plus prêt à accepter des données. Lorsque tous les écouteurs ont placé la ligne NDAC au niveau haut, le parleur passe la ligne DAV au niveau haut. Cela conduit à passer la ligne NDAC au niveau bas. L'intégralité de cette procédure est répétée lorsqu'un nouveau mot est placé sur les lignes de données.

4.3.4 Protocoles

Il est indispensable d'effectuer un contrôle sur le flux des données entre deux dispositifs afin de définir ce qui constitue le message et la manière de démarrer et d'arrêter la communication. Tous ces aspects forment un *protocole*.

Copyright © 2015 Dunod.

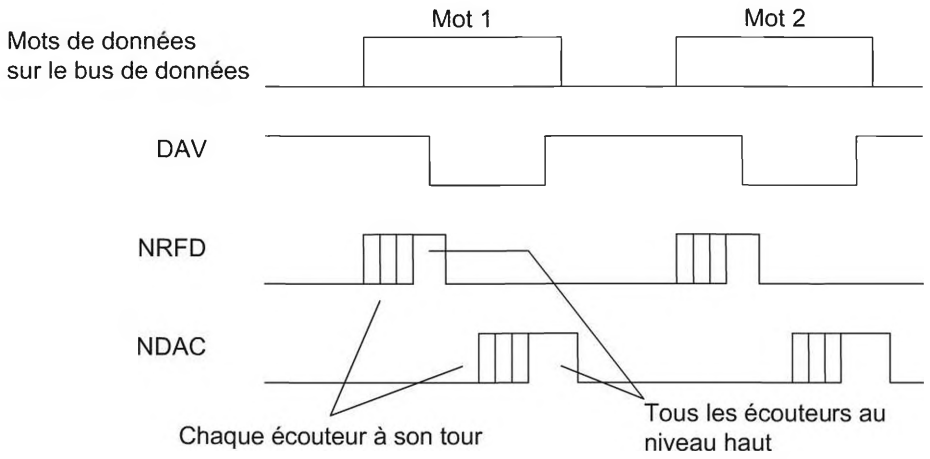


Figure 4.26 – Séquence de dialogue.

Un dispositif a besoin de pouvoir indiquer à un autre qu'il doit commencer ou arrêter d'envoyer des données. Cet échange peut se faire à l'aide des fils de dialogue qui connectent l'émetteur et le récepteur. Un signal sur l'un des fils indique au récepteur que l'émetteur est prêt à envoyer (RTS, *Ready To Send*), et un signal sur un autre fil indique que l'émetteur est prêt à recevoir (CTS, *Clear-To-Send*). Les lignes RTS et CTS existent sur les interfaces de communication série RS232.

Une autre méthode consiste à utiliser des caractères supplémentaires lors des transmissions. Dans le protocole ENQ/ACK, des paquets de données sont envoyés à un récepteur avec le caractère de requête ENQ. La réception de ce caractère signifie la fin du paquet de données. Lorsque le récepteur a traité ces données, il peut indiquer qu'il est prêt à recevoir un autre bloc de données en renvoyant un signal de confirmation (ACK). Dans le protocole XON/XOFF, l'appareil récepteur envoie un signal XOFF à l'appareil émetteur lorsqu'il souhaite stopper le flux des données. L'émetteur attend ensuite l'arrivée d'un signal XON pour reprendre la transmission.

Pour vérifier si des erreurs se sont introduites dans le message suite à sa transmission, une méthode consiste à effectuer un *contrôle de parité*. Il s'agit d'un bit supplémentaire ajouté au message pour faire en sorte que le nombre de bits à 1 contenus dans un élément de données soit toujours pair ou impair. Par exemple, 0100100 est pair car il contient un nombre pair de 1, tandis que 0110100 est impair car il contient un nombre impair de 1. Pour mettre en œuvre une parité impaire, nous ajoutons un bit 1 à la fin du premier nombre et un bit 0 à la fin du second, pour obtenir 01001001 et 01101000. Lorsqu'un message est envoyé avec un bit de parité impaire, le récepteur peut déterminer qu'il est corrompu si les bits reçus produisent une parité paire. Dans ce cas, il peut demander une nouvelle transmission du message.

Le bit de parité permet de détecter une erreur due au changement d'un seul bit 0 en un bit 1, ou inversement, mais est incapable de détecter deux erreurs de ce type, puisqu'elles ne changent pas la parité. Pour détecter de telles erreurs, des

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

méthodes plus sophistiquées doivent être employées. L'une d'elles consiste à stocker les mots de données dans un tableau de lignes et de colonnes. La parité de chaque ligne et de chaque colonne peut ensuite être vérifiée. Voici une illustration de ce principe pour sept mots et une parité paire.

		Bits de parité des lignes
Bits de parité des colonnes	00101010	1
↑	10010101	0
	10100000	0
Bloc de	01100011	0
données	11010101	1
	10010101	1
↓	00111100	0

Une autre méthode, appelée *contrôle de redondance cyclique* (CRC, *Cyclic Redundancy Check*), oblige à découper le message en blocs. Chaque bloc est ensuite traité comme un nombre binaire et est divisé par une valeur prédéterminée. Le reste de la division est envoyé comme nombre de contrôle des erreurs à la fin du message et permet de vérifier son exactitude.

4.3.5 Code ASCII

Le code ASCII (*American Standard Code for Information Interchange*) est le code le plus employé dans la transmission des caractères. Il s'agit d'un code sur sept bits qui permet d'obtenir 128 combinaisons de bits différentes afin de couvrir les caractères alphanumériques en minuscule et en majuscule, les caractères de ponctuation et trente-deux codes de contrôle. Le Tableau 4.2 présente les codes qui correspondent aux lettres majuscules. Parmi les codes de contrôle, nous trouvons SOH (*Start Of Heading*), c'est-à-dire le premier caractère de l'en-tête d'un message d'information, avec le code 000 0001, STX (*Start Of Text*), c'est-à-dire début du texte, avec le code 000 0010, ETX (*End Of Text*), c'est-à-dire fin du texte, avec le code 000 0011, et EOT (*End Of Transmission*), c'est-à-dire fin de transmission, avec le code 000 0011.

Tableau 4.2 Exemples de codes ASCII

	ASCII		ASCII		ASCII
A	100 0001	N	100 1110	0	011 0000
B	100 0010	O	100 1111	1	011 0001
C	100 0011	P	101 0000	2	011 0010
D	100 0100	Q	101 0001	3	011 0011
E	100 0101	R	101 0010	4	011 0100

Copyright © 2015 Dunod.

Tableau 4.2 Exemples de codes ASCII *(suite)*

	ASCII		ASCII		ASCII
F	100 0110	S	101 0011	5	011 0101
G	100 0111	T	101 0100	6	011 0110
H	100 1000	U	101 0101	7	011 0111
I	100 1001	V	101 0110	8	011 1000
J	100 1010	W	101 0111	9	011 1001
K	100 1011	X	101 1000		
L	100 1100	Y	101 1001		
M	100 1101	Z	101 1010		

4.4 Réseaux

L'utilisation croissante de l'automatique dans l'industrie a conduit à des besoins de communication et de contrôle au niveau de l'entreprise, avec des automates programmables, des ordinateurs, des machines numériques et des robots interconnectés. Le terme *réseau local* (LAN, *Local Area Network*) décrit un réseau de communication conçu pour relier les ordinateurs et leur périphériques à l'intérieur du même bâtiment ou du même site.

Les réseaux existent sous trois topologies de base. Dans un réseau en *étoile* (voir Figure 4.27a), les terminaux, appelés *esclaves*, sont tous reliés directement à un ordinateur central, appelé *hôte* ou *maître*. L'hôte comprend la mémoire, le traitement et le matériel de commutation qui permettent aux terminaux de communiquer. L'accès aux terminaux est décidé par l'hôte, qui demande à chacun s'il souhaite émettre ou recevoir. Dans un réseau en *bus* (voir Figure 4.27b), tous les terminaux sont reliés à un même câble et chaque émetteur/récepteur dispose ainsi d'un chemin direct avec chaque autre émetteur/récepteur du réseau. Des protocoles doivent être mis en place pour garantir qu'un seul terminal émet à la fois ; dans le cas contraire, une confusion peut se produire. Un terminal doit pouvoir détecter si un autre terminal émet avant qu'il ne puisse commencer à émettre lui-même. Dans un réseau en *anneau* (voir Figure 4.27c), un câble relie tous les terminaux dans un anneau. Une fois encore, des protocoles doivent être mis en place pour permettre les communications entre les différents terminaux sans que les messages ne se mélangent. Les réseaux en bus et en anneau sont également appelés réseau *pair à pair* car chaque terminal a le même statut. Ces systèmes permettent à de nombreuses stations d'utiliser le même réseau.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

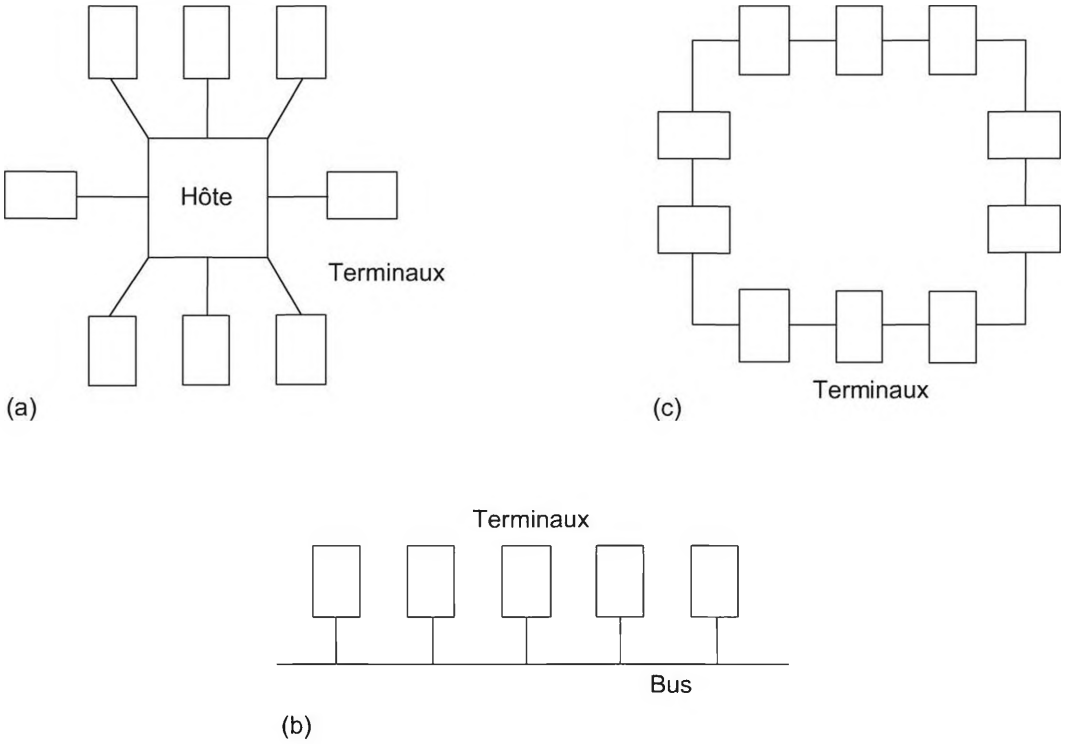


Figure 4.27 – Topologies de réseaux : (a) en étoile, (b) en bus et (c) en anneau.

Dans les réseaux en anneau, deux méthodes sont fréquemment employées pour éviter que deux terminaux n'émettent en même temps et ne sèment la confusion : passage de jeton et passage d'emplacement. Dans le *passage de jeton*, un motif de bit particulier, appelé *jeton*, circule dans le réseau. Lorsqu'un terminal souhaite émettre sur le réseau, il attend de recevoir le jeton, puis transmet ses données en leur associant le jeton. Si une autre station souhaite émettre, elle doit attendre qu'un récepteur libère le jeton en le retirant des données. Dans le *passage d'emplacement*, des emplacements vides circulent et les terminaux peuvent y déposer des données pour leur transmission.

Les réseaux en bus emploient généralement une méthode dans laquelle le système qui souhaite émettre écoute le réseau pour savoir si des messages sont en cours de transmission. Dans la négative, une station peut prendre le contrôle du réseau et envoyer son message. Cette méthode se nomme CSMA (*Carrier Sense Multiple Access*). Toutefois, il peut arriver que deux stations détectent simultanément que le réseau est libre et qu'elles prennent donc toutes deux son contrôle et envoient des messages. Cela déclenche une « collision » des données transmises et leur corruption. Lorsque cette situation est détectée, les deux terminaux cessent d'émettre et patientent pendant une durée aléatoire avant de tenter à nouveau d'émettre. Cette méthode se nomme CSMA/CD (*Carrier Sense Multiple Access with Collision Detection*).

Les fabricants d'API adoptent différentes topologies de réseaux et de méthodes de communication. Par exemple, Mitsubishi a choisi un réseau nommé MelsecNET, Allen-Bradley utilise Data Highway Plus, General Electric a opté pour GENET,

Copyright © 2015 Dunod.

Texas Instruments a retenu TIWAY et Siemens utilise PROFIBUS DP. À l'instar d'Allen-Bradley, la plupart emploient des systèmes pair à pair. Le réseau de Siemens, PROFIBUS DP, utilise une topologie en étoile, c'est-à-dire de type maître-esclave.

4.4.1 Systèmes distribués

Les API se placent souvent dans une hiérarchie de communication (voir Figure 4.28). Au niveau le plus bas se trouvent les dispositifs d'entrées et de sorties, tels que les capteurs et les moteurs, connectés au niveau suivant par l'intermédiaire des interfaces d'entrées-sorties. Le niveau suivant fait intervenir des automates, comme de petits API ou de petits ordinateurs, reliés par un réseau, avec, au niveau supérieur, des API plus importants et des ordinateurs mettant en œuvre un contrôle local. À leur tour, ils peuvent faire partie d'un réseau dans lequel se trouve un mainframe qui contrôle l'ensemble.

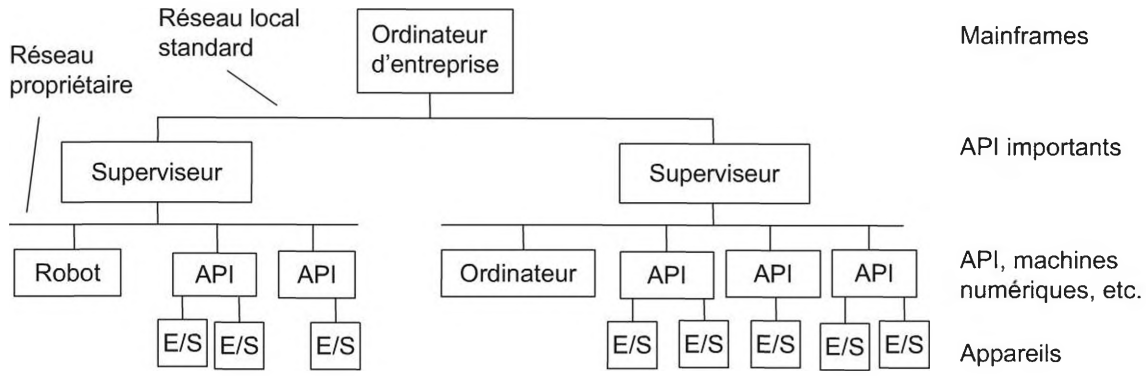


Figure 4.28 – Hiérarchie de communication.

Les systèmes qui permettent de commander et de surveiller des processus industriels sont de plus en plus utilisés. Cela implique un contrôle et la collecte de données. Le terme SCADA (*Supervisory Control And Data Acquisition*) est largement employé pour décrire un tel système de télésurveillance et d'acquisition de données.

4.4.2 Normes réseau

L'interconnexion de multiples dispositifs peut soulever des problèmes de compatibilité. Par exemple, ils peuvent opérer à des débits différents ou employer des protocoles différents. Pour faciliter les communications entre les appareils, l'ISO (*International Standards Organization*) a défini, en 1979, un modèle permettant de standardiser l'interconnexion des systèmes ouverts (OSI, *Open Systems Interconnection*) ; il se nomme *modèle OSI de l'ISO*. Un lien de communication entre des équipements numériques est défini par des normes physiques, électriques, de protocole et d'utilisateurs. Le modèle OSI les découpe en sept couches (voir Figure 4.29).

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

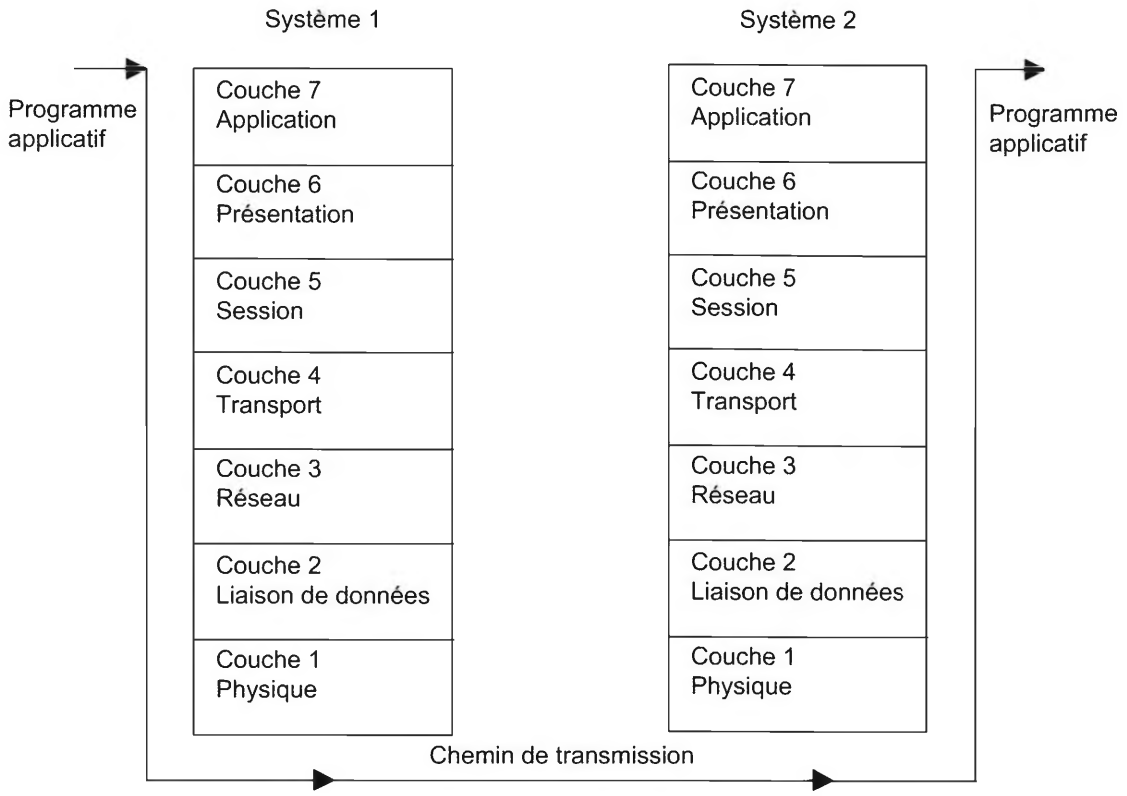


Figure 4.29 – Le modèle OSI de l'ISO.

Voici la fonction de chaque couche du modèle :

- *Couche 1 : physique.* Cette couche concerne le codage et la transmission physique des informations. Elle prend en charge la synchronisation des échanges de données et le transfert des bits de données entre des systèmes.
- *Couche 2 : liaisons de données.* Cette couche définit les protocoles d'émission et de réception des informations entre des systèmes connectés directement les uns aux autres. Elle assemble les bits issus de la couche physique sous forme de blocs et prend en charge leur transfert, le contrôle de l'ordre des blocs de données, ainsi que la détection et la correction des erreurs.
- *Couche 3 : réseau.* Cette couche définit la distribution des données entre les systèmes du réseau.
- *Couche 4 : transport.* Cette couche définit les protocoles d'envoi des messages d'une extrémité du réseau à une autre. Elle contrôle le flux des messages.
- *Couche 5 : session.* Cette couche apporte les fonctions qui permettent d'établir des communications entre des utilisateurs distants.
- *Couche 6 : présentation.* Cette couche garantit que les informations sont livrées sous une forme adaptée.
- *Couche 7 : application.* Cette couche prend en charge la liaison du programme utilisateur avec le système de communication et s'occupe de donner un sens aux informations transmises.

Copyright © 2015 Dunod.

Chaque couche est autonome et n'est concernée que par les interfaces des couches immédiatement supérieure et inférieure. Elle réalise ses opérations et transfère les résultats à la couche supérieure ou à la couche inférieure. Cela permet aux fabricants de concevoir des produits qui fonctionnent dans une couche particulière et qui s'interfacent avec le matériel des autres fabricants.

Pour illustrer le fonctionnement de chaque couche, faisons l'analogie avec le passage d'un appel téléphonique. Le médium physique correspond à la ligne téléphonique. La couche 1 s'assure que le signal vocal est converti en un signal électrique avant sa transmission et, à l'autre extrémité de la ligne, qu'il est de nouveau transformé en signal électrique. Elle définit également les types de connecteurs et les niveaux de signal requis. La couche 2 s'assure que les mots qui ne sont pas clairement reçus sont signalés à l'émetteur afin qu'il les transmette de nouveau. La couche 3 apporte le mécanisme de composition du numéro du correspondant afin d'établir la connexion entre l'émetteur et le récepteur. La couche 4 est utilisée pour garantir que les messages sont transmis sans perte. La couche 5 apporte les protocoles permettant de passer un appel entre des individus précis, par exemple pour qu'une personne dans un bureau réponde au téléphone. La couche 6 résout les problèmes de langue afin que les deux correspondants parlent la même langue. Elle fournit les procédures à adopter pour transmettre les éléments d'information, comme la quantité à commander suivie du numéro de référence du produit.

En 1980, l'IEEE (*Institute of Electronic and Electrical Engineers*) a démarré le Projet 802. Il s'agit un modèle qui adhère à la couche physique de l'OSI, mais qui sépare la couche de liaison de données en deux couches distinctes : la couche de contrôle d'accès au support (MAC, *Media Access Control*) et la couche de contrôle de la liaison logique (LLC, *Logical Link Control*). La couche MAC définit la méthode d'accès au médium de transmission. Elle consiste en un ensemble de normes de contrôle de l'accès au réseau et s'assure qu'un seul utilisateur peut transmettre à la fois. L'une des normes est l'IEEE 802.3 CSMA/CD (*Carrier Sense Multiple Access and Collision Detection*). Les stations doivent vérifier l'absence d'autres transmissions avant de pouvoir obtenir le contrôle du réseau et d'émettre. L'IEEE 802.4 Token Bus est une autre norme. Dans cette méthode, un motif de bits particulier, le jeton, circule sur le réseau et, lorsqu'une station veut émettre, elle attend de recevoir ce jeton et l'ajoute ensuite à la fin des données. La couche LLC est responsable de la fiabilité de la transmission des paquets au travers de la couche physique.

La partie 5 de la norme CEI 61131 pour les automates programmables industriels (voir Section 1.5.1) concerne les communications. Elle fixe les standards pour les fonctions de communication avec les API, qu'ils soient utilisés comme serveurs (fournissent des informations et répondent aux requêtes de services) ou comme clients (demandent des informations et initient des requêtes aux services). Pour que des API conformes à la CEI puissent échanger des informations et des signaux de contrôle, plusieurs blocs de communication standard sont définis :

– CONNECT pour établir un canal entre un API appelant et un API distant. Lors de l'invocation de celui-ci, l'adresse réseau complète de l'API doit être précisée.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

- STATUS et USTATUS pour fournir le service de lecture de l'état d'API distants afin de vérifier qu'ils sont opérationnels. Le bloc STATUS correspond à la demande des informations d'état d'un API distant, tandis que le bloc USTATUS permet à un API de recevoir ces informations à partir d'un API distant.
- READ, USEND et URCV pour la lecture de la valeur de variables sur des API distants. Le bloc READ interroge un API distant et lui demande les valeurs d'une ou plusieurs variables. Le bloc USEND est utilisé par un API pour transmettre des données à une instance particulière d'un bloc URCV présent sur un API distant.
- WRITE, SEND et RCV pour le contrôle des interactions entre des API. Le bloc WRITE écrit une ou plusieurs valeurs dans une ou plusieurs variables sur un API distant. Le bloc SEND correspond à une demande d'envoi de données à un API distant, tandis que le bloc RCV est employé pour leur réception.
- NOTIFY et ALARM pour qu'un API signale une condition d'alarme à des API distants. NOTIFY rapporte un message d'alarme, tandis que ALARM correspond à un message d'alarme signalé avec un accusé de réception de l'alarme.
- REMOTE_VAR pour obtenir l'adresse précise d'une variable nommée.

4.5 Exemples de systèmes commerciaux

Les sections suivantes décrivent quelques systèmes que vous pourriez rencontrer dans les installations impliquant des API.

4.5.1 MAP

En 1990, General Motors avait un problème dans l'automatisation de ses activités de fabrication : tous les systèmes de la société devaient être en mesure de communiquer les uns avec les autres. Elle a donc développé un système de communication standard pour les automatismes appelé MAP (*Manufacturing Automation Protocol*). Le système a été appliqué à toutes les machines de l'atelier, comme les robots, les API et les postes de soudage. Le Tableau 4.3 présente le modèle MAP et son lien avec le modèle de l'ISO. Pour que le matériel non-OSI puisse fonctionner dans le système MAP, des *passerelles* sont nécessaires. Il s'agit d'unités autonomes ou de cartes d'interface qui se connectent au dispositif afin que les messages issus d'un réseau ou d'un appareil non-OSI puissent être transmis vers d'autres systèmes au travers du bus à jeton large bande MAP.

La couche application prend en charge le protocole MMS (*Manufacturing Message Service*), qui définit les interactions entre les API et les machines numériques et les robots.

Au niveau de la liaison de données, des méthodes sont nécessaires pour garantir que seul l'utilisateur du réseau est en mesure de transmettre à un moment donné. Dans le cas de MAP, la méthode employée est le passage de jetons. Le terme *large bande* (*broadband*) est utilisé pour un réseau dans lequel les informations sont modulées sur une porteuse radioélectrique transmise au travers du câble coaxial.

Copyright © 2015 Dunod.

Tableau 4.3 MAP

Couche ISO		MAP
7	Application	Transfert de fichiers ISO, MMS, FTAM, CASE
6	Présentation	
5	Session	ISO Session Kernel
4	Transport	ISO Transport Class 4
3	Réseau	ISO Internet
2	Liaison de données	IEEE 802.2 Class 1, IEEE 802.4 Token Bus
1	Physique	IEEE 802.4 Broadband
	Transmission	Câble coaxial 10 mbps avec modulateur RF

Note : MMS = système de messagerie industrielle, FTAM = transfert de fichier, CASE = services applicatifs communs. Chacun de ces protocoles propose un ensemble de commandes qui seront comprises par les dispositifs et les logiciels utilisés.

Si MAP est peu utilisé, *Ethernet* est beaucoup plus répandu. Il s'agit d'un système de bus unique dans lequel le contrôle des accès se fait selon la méthode CSMA/CD. Il utilise un câble coaxial dont la longueur maximale est de 500 m. Jusqu'à 1 024 stations sont acceptées et des répéteurs, qui remettent en forme le signal, permettent d'étendre ses possibilités (voir Figure 4.30). Chaque station est connectée au bus au travers d'un émetteur-récepteur, qui se fixe sur le câble du bus. Le terme *prise vampire* désigne la pince de liaison au câble car les stations peuvent être connectées et déconnectées sans interrompre le fonctionnement du système.

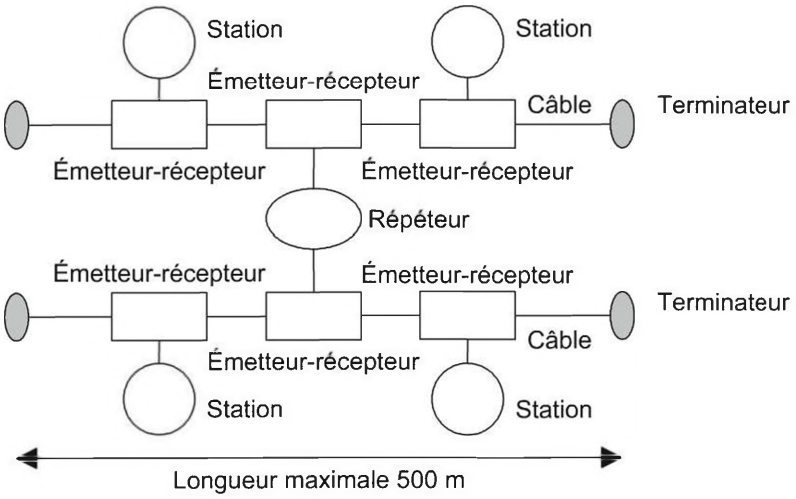


Figure 4.30 – Ethernet en bande de base avec des répéteurs.

Le terme *bande de base* est utilisé lorsque le signal est transmis sous forme d'une suite de niveaux de tension qui représentent directement les bits envoyés.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

4.5.2 Ethernet

Ethernet n'implique aucune station maître. Toutes les stations connectées ont le même statut et nous avons donc une communication de pair à pair. Une station qui souhaite envoyer un message sur le bus détermine si celui-ci est disponible et, dans l'affirmative, place la trame du message sur le bus. Il est possible que plusieurs stations détectent un bus inactif et tentent d'émettre. C'est pourquoi chaque émetteur surveille le bus au cours de la transmission et est capable de détecter lorsque le signal sur le bus ne correspond pas à sa propre sortie. Lorsqu'une telle « collision » est détectée, la transmission se poursuit pendant un court moment afin que les autres stations aient également l'opportunité de la détecter et tentent de retransmettre leur message ultérieurement. Chaque message comprend une suite de bits qui indiquent l'adresse de destination, l'adresse de source, les données à transmettre et une séquence de contrôle du message. Cette dernière contient le CRC (voir Section 4.3.4). Chaque station réceptrice vérifie l'adresse de destination de la trame pour savoir si elle lui est destinée. Dans l'affirmative, elle accepte le message. Ethernet est très utilisé dans les configurations où les API doivent communiquer avec des ordinateurs. Le modèle modulaire PLC-5 d'Allen-Bradley peut être configuré pour différents réseaux de communication en ajoutant les modules appropriés (voir Figure 1.17) ; un module prend en charge Ethernet. Ethernet est plus rapide que MAP, car la méthode de passage du jeton employée par MAP est plus lente que celle employée avec Ethernet. En plus d'offrir une compatibilité avec Ethernet, les fabricants d'API proposent généralement leurs propres réseaux.

4.5.3 ControlNet

Ce réseau est utilisé par Allen-Bradley. Les données sont placées sur le réseau sans aucune indication, comme celle du destinataire. Toutes les stations qui utilisent ces données peuvent ainsi les accepter simultanément. Cela permet de réduire le nombre de messages envoyés sur le réseau et donc d'augmenter sa rapidité. Les racks d'API et leurs données peuvent ainsi être partagés de manière égale entre plusieurs processeurs et non dédiés à un seul. L'accès au réseau est sous le contrôle d'un algorithme de minutage, appelé CTDMA (*Concurrent Time Domain Multiple Access*), qui détermine si un nœud a la possibilité de transmettre sur le réseau.

4.5.4 DeviceNet

Ce réseau se fonde sur le bus CAN (*Controller Area Network*), qui a été très utilisé dans les voitures (voir Section 4.3.2). Chaque dispositif du réseau se voit demandé d'émettre ou de recevoir une mise à jour de son état, chacun devant généralement répondre à son tour. Les dispositifs sont configurés pour envoyer automatiquement des messages à intervalles planifiés ou uniquement lorsque leur état change. DeviceNet constitue généralement un sous-réseau d'un API connecté à un réseau Ethernet ou ControlNet. Il est utilisé pour relier des dispositifs comme des capteurs, des démarreurs et des vannes pneumatiques.

Copyright © 2015 Dunod.

4.5.5 Allen-Bradley Data Highway

Le réseau *Allen-Bradley Data Highway* est un système pair à pair développé pour les API d'Allen-Bradley. Il utilise le passage de jeton pour contrôler la transmission des messages. Les adresses de chaque API sont configurées par des interrupteurs. La communication est établie par un seul message sur le réseau, en précisant les adresses d'émission et de réception, ainsi que la taille du bloc à transférer.

4.5.6 PROFIBUS

PROFIBUS (*Process Field Bus*) est un système développé en Allemagne que Siemens utilise avec ses API. PROFIBUS DP (*Decentralized Periphery*) est un bus de niveau appareil qui fonctionne habituellement avec un seul DP maître et plusieurs esclaves. Plusieurs de ces systèmes DP peuvent être installés sur un réseau PROFIBUS. Les transmissions se font par RS485 (comparable à RS422 ; voir Section 4.3.2) ou par fibre optique. Ce système est équivalent à DeviceNet. PROFIBUS PA (*Process Automation*) est une extension de PROFIBUS DP pour la transmission de données à partir de dispositifs tels que des capteurs et des actionneurs. Il est possible de connecter un PROFIBUS DP à un PROFIBUS PA en utilisant un coupleur DP/PA si le système peut fonctionner à 45,45 kbits/s. Dans le cas contraire, un lien DP/PA doit être utilisé pour passer du débit des données de PROFIBUS DP à celui de PROFIBUS PA.

4.5.7 Réseau industriel

Un réseau industriel utilise généralement différents systèmes réseau. Un Ethernet constitue la couche informations et sert à la collecte des données et à la maintenance des programmes. Le niveau inférieur peut être un réseau ControlNet, pour le traitement en temps réel des entrées-sorties, et le niveau le plus bas, un réseau DeviceNet pour gérer les capteurs et les pilotes. Les API prennent leurs instructions à partir de la couche Ethernet et exécutent leurs commandes au travers de la couche ControlNet.

4.6 Traitement des entrées

Un API fonctionne en continu, exécutant son programme et réagissant aux signaux d'entrée. Cette boucle de fonctionnement est appelée *cycle*. Les API peuvent être activés par chaque entrée examinée lorsqu'elle se présente dans le programme, son effet sur le programme déterminé et la sortie modifiée en conséquence. Ce mode de fonctionnement est appelé *actualisation continue*.

Dans une actualisation continue, du temps est passé à interroger chaque entrée. Par conséquent, le temps nécessaire à examiner plusieurs centaines de points d'entrées-sorties peut devenir relativement long. Afin de permettre une exécution plus rapide d'un programme, une zone particulière de la RAM sert de tampon entre la logique de commande et l'unité d'entrées-sorties. Chaque entrée-sortie possède une adresse dans cette zone de mémoire. Au début de chaque cycle du

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

programme, le CPU scrute toute les entrées et copie leur état aux adresses correspondantes dans la RAM. Pendant l'exécution du programme, les données d'entrées enregistrées sont lues à partir de la RAM et les opérations logiques sont réalisées. Les signaux de sortie résultants sont stockés dans la zone de mémoire réservée aux entrées-sorties. À la fin de chaque cycle du programme, toutes les sorties sont transférées depuis la RAM vers les canaux de sorties appropriés. Les sorties conservent ensuite leur état jusqu'à la prochaine actualisation. Cette solution est appelée *copie en masse des E/S*. Le fonctionnement de l'API peut se résumer de la manière suivante (voir Figure 4.31) :

1. Scruter toutes les entrées et les copier en RAM.
2. Lire, décoder et exécuter toutes les instructions du programme en séquence, en copiant le résultat des instructions en RAM.
3. Affecter toutes les sorties.
4. Répéter la procédure.

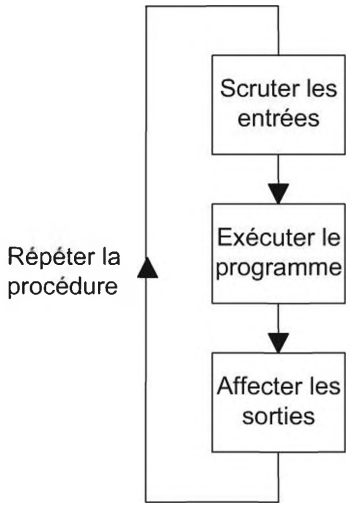


Figure 4.31 – Cycle de fonctionnement d'un API.

Si le temps mis pour effectuer l'intégralité d'un cycle d'analyse des entrées et d'affectation des sorties conformément aux instructions du programme, appelé *temps de cycle*, est relativement rapide, il n'est pas instantané. Autrement dit, toutes les entrées ne sont pas scrutées en même temps, mais des échantillons de leur état sont pris périodiquement. Un temps de cycle se situe entre 10 et 50 ms. Cela signifie que les entrées et les sorties sont actualisées toutes les 10 à 50 ms et qu'un retard de cet ordre de grandeur peut être constaté dans la réaction du système. Cela signifie également qu'un événement d'entrée très court apparaissant au mauvais moment du cycle peut être manqué. En général, une entrée doit être présente pendant une durée plus longue que le temps de cycle. Des modules spéciaux existent pour prendre en charge ces cas.

Prenons un API dont le temps de cycle est de 40 ms. Quelle est la fréquence maximale à laquelle les impulsions numériques peuvent être détectées ? Cette fréquence

Copyright © 2015 Dunod.

maximale est obtenue lorsque l'impulsion se produit chaque 40 ms, c'est-à-dire à une fréquence de 1/0,04 = 25 Hz.

Le temps de cycle ou de scrutation d'un API, c'est-à-dire son temps de réponse, est déterminé par les facteurs suivants :

1. Le CPU employé.
2. La taille du programme à analyser.
3. Le nombre d'entrées-sorties à gérer.
4. Les fonctions systèmes utilisées ; plus leur nombre est grand, plus le temps de scrutation est long.

Par exemple, l'API compact MELSEC FX3U de Mitsubishi (voir Section 1.4) affiche un temps de cycle de 0,065 µs par instruction logique. Par conséquent, plus le programme sera complexe, plus le temps de cycle sera long.

4.7 Adresses des entrées-sorties

L'API doit être en mesure d'identifier chaque entrée et chaque sortie. Pour cela, il alloue des adresses à chacune d'elles. Dans le cas d'un petit API, une adresse correspondra probablement à un nombre préfixé d'une lettre qui indique s'il s'agit d'une entrée ou d'une sortie. Dans le cas de l'API de Mitsubishi, nous avons des entrées avec les adresses X400, X401, X402, etc., et des sorties avec des adresses Y430, Y431, Y432, etc. La lettre X désigne donc une entrée, tandis que la lettre Y désigne une sortie. Toshiba utilise un système comparable.

Pour les API plus importants, qui disposent de plusieurs racks de canaux d'entrées et de sorties, chaque rack est numéroté. Dans le cas du modèle PLC-5 d'Allen-Bradley, le rack qui contient le processeur reçoit le numéro 0, tandis que les autres racks possèdent les adresses 1, 2, 3, etc., comme indiqué par des interrupteurs. Chaque rack peut avoir plusieurs modules, chacun gérant plusieurs entrées et/ou sorties. Les adresses ont ainsi le format illustré à la Figure 4.32. Par exemple, nous pouvons avoir une entrée dont l'adresse est I:012/03. Elle correspond à une entrée, sur le rack 01, le module 2 et le port 03.

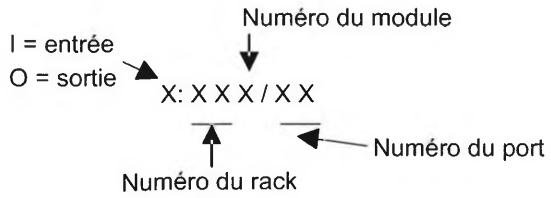


Figure 4.32 – Format d'une adresse dans l'automate PLC-5 d'Allen-Bradley.

Pour le modèle SIMATIC S7 de Siemens, les entrées et les sorties sont regroupées par huit. Chaque groupe est un *octet* et chaque entrée ou sortie d'un groupe est un *bit*. Les adresses des entrées et des sorties sont donc données avec un octet et un bit, ce qui constitue un numéro de module suivi d'un numéro de port, séparés par

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

un point (.). La Figure 4.33 illustre le principe. L'adresse I0.1 correspond donc à une entrée sur le bit 1 de l'octet 0, et Q2.0, à une sortie sur le bit 0 de l'octet 2.

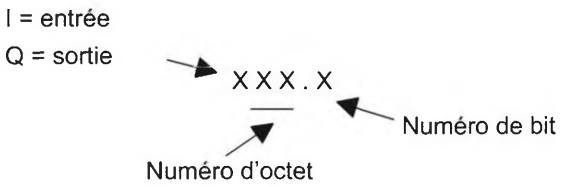


Figure 4.33 – Format d'une adresse dans l'automate SIMATIC S7 de Siemens.

L'automate GEM-80 utilise des adresses d'entrées-sorties fondées sur un numéro de module et un numéro de port dans ce module. La lettre A désigne une entrée, la lettre B, une sortie. Par conséquent, A3.02 correspond à une entrée sur le port 02 du module 3, et B5.12, à une sortie sur le port 12 du module 5.

Outre l'emploi des adresses pour identifier les entrées et les sorties, les API utilisent également leur système d'adressage pour désigner les dispositifs logiciels internes, comme les relais, les temporisateurs et les compteurs.

4.8 En résumé

Les unités d'entrées-sorties des API sont conçues pour que différents signaux d'entrée puissent être transformés en signaux numériques 5 V et pour proposer toute une gamme de signaux de sortie. Une unité d'entrées à fourniture de courant fournit le courant d'alimentation au dispositif d'entrée qui lui est connecté. Avec une unité d'entrées à absorption de courant, le dispositif d'entrée fournit le courant à l'unité d'entrées. Une unité de sorties à fourniture de courant fournit le courant au dispositif de sortie. Dans le cas d'une unité à absorption de courant, le dispositif de sortie produit le courant nécessaire à la sortie de l'API. Les unités de sorties peuvent être à relais, à transistor ou à triac.

Pour les entrées, un traitement du signal est généralement mis en œuvre de manière à convertir les signaux analogiques en un courant de 4 à 20 mA et, en le faisant passer dans une résistance de 250 Ω, en un signal d'entrée de 1 à 5 V. Pour cela, il est possible d'utiliser un diviseur de tension ou un amplificateur opérationnel. Un amplificateur opérationnel peut également servir à comparer deux signaux et à produire un signal tout ou rien en fonction de leur valeur relative.

Dans une communication série, les données sont transmises un bit à la fois. Dans une communication parallèle, les bits qui composent un mot de données sont séparés et transmis simultanément sur des câbles en parallèle. La norme de communication série la plus employée est la norme RS232 ; il existe également les normes RS422 et RS423. La boucle 20 mA peut être employée pour les communications série. La norme de communication parallèle la plus employée est la norme IEEE-488. Des protocoles sont nécessaires pour contrôler le flux de données entre des appareils. Le code le plus employé pour la transmission des caractères est le code ASCII.

Copyright © 2015 Dunod.

Un *réseau local* (LAN, *Local Area Network*) est un réseau de communication conçu pour relier des ordinateurs et leurs périphériques, à l'intérieur du même bâtiment ou du même site. Il existe trois topologies de réseau : en étoile, en bus et en anneau. Les API se placent souvent dans une hiérarchie de communication. Au niveau le plus bas se trouvent les dispositifs d'entrées et de sorties. Le niveau suivant fait intervenir de petits API ou de petits ordinateurs. Au niveau supérieur se placent les API et les ordinateurs plus importants. Le modèle OSI de l'ISO a été créé pour normaliser l'interconnexion des systèmes ouverts. Parmi les réseaux commerciaux, on trouve MAP, Ethernet, ControlNet, DeviceNet, Allen-Bradley Data Highway et PROFIBUS.

Un API fonctionne en continu, exécutant son programme et le mettant à jour. Pour cela, il réalise une copie en masse des entrées-sorties au cours de laquelle les entrées sont scrutées et copiées en RAM, puis elles sont lues et décodées, toutes les instructions du programme sont exécutées en séquence et les résultats sur les sorties sont copiés en RAM. Ensuite, toutes les sorties sont affectées avant que la procédure complète ne reprenne. Pour être en mesure d'identifier chaque entrée et chaque sortie, l'API alloue une adresse à chacune.

4.9 Problèmes

Pour les questions 1 à 15, quatre réponses sont proposées : A, B, C et D. Choisissez la réponse correcte.

1. Un CAN est utilisé pour échantillonner la tension de sortie d'un capteur de pression. Si la sortie du capteur est égale à 0 V lorsque la pression est égale à 0 kPa et à 10 V pour une pression de 10 kPa, le nombre minimal de bits requis pour convertir la sortie du capteur si son erreur ne dépasse pas 0,01 kPa est :
   A. Quatre
   B. Huit
   C. Dix
   D. Douze
2. Un CAN douze bits peut représenter les tensions analogiques sur son entrée avec :
   A. 12 nombres binaires différents
   B. 24 nombres binaires différents
   C. 144 nombres binaires différents
   D. 4 096 nombres binaires différents
3. Pour une entrée analogique dont la tension varie entre 0 et 10 V, le nombre minimal de bits du CAN pour noter un changement de 0,1 V est :
   A. Quatre bits
   B. Six bits
   C. Huit bits
   D. Douze bits

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

4. Un amplificateur opérationnel monté en inverseur a une résistance d'entrée de 10 kΩ et une résistance de contre-réaction de 100 kΩ. Le gain en boucle fermée de l'amplificateur est égal à :
   A. −100
   B. −10
   C. +10
   D. +100

Les problèmes 5 et 6 font référence à un amplificateur opérationnel dont le gain en boucle fermée est égal 100 et dont la résistance d'entrée est de 47 kΩ.

5. La résistance de contre-réaction de l'amplificateur opérationnel en montage inverseur doit être de :
   A. 4,65 kΩ
   B. 4,7 kΩ
   C. 465 kΩ
   D. 470 kΩ
6. La résistance de contre-réaction de l'amplificateur opérationnel en montage non-inverseur doit être de :
   A. 4,65 kΩ
   B. 4,7 kΩ
   C. 465 kΩ
   D. 470 kΩ
7. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Une interface de communication série :
   (i) implique que les données soient transmises et reçues un bit à la fois ;
   (ii) est plus rapide qu'une communication parallèle.

   Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
   A. (i) V (ii) V
   B. (i) V (ii) F
   C. (i) F (ii) V
   D. (i) F (ii) F
8. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). L'interface de communication RS232 :
   (i) est une interface série ;
   (ii) est généralement employée pour des distances allant jusqu'à 15 m.

   Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
   A. (i) V (ii) V
   B. (i) V (ii) F
   C. (i) F (ii) V
   D. (i) F (ii) F

Les problèmes 9 et 10 font référence à la transmission des données 1100001 sur une ligne de communication RS232 selon le format suivant :

Copyright © 2015 Dunod.

```
0110000111
X        YZ
```

9. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Les bits supplémentaires X et Z au début et à la fin sont utilisés :
   (i) pour vérifier si le message a été corrompu au cours de sa transmission ;
   (ii) pour indiquer où commencent et s'arrêtent les données.

   Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
   A. (i) V (ii) V
   B. (i) V (ii) F
   C. (i) F (ii) V
   D. (i) F (ii) F

10. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Le bit Y est :
   (i) le bit de parité qui indique une parité impaire ;
   (ii) le bit de parité qui indique une parité paire.

   Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
   A. (i) V (ii) V
   B. (i) V (ii) F
   C. (i) F (ii) V
   D. (i) F (ii) F

11. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Une interface de communication parallèle :
   (i) permet de transmettre des données sur de courtes distances à vitesse élevée ;
   (ii) est souvent à la norme IEEE-488.

   Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
   A. (i) V (ii) V
   B. (i) V (ii) F
   C. (i) F (ii) V
   D. (i) F (ii) F

12. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Pour les communications sur de longues distances, de l'ordre de 100 à 300 m, à des débits élevés :
   (i) il est possible d'utiliser une interface RS232 ;
   (ii) il est possible d'utiliser une boucle de courant de 20 mA.

   Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
   A. (i) V (ii) V
   B. (i) V (ii) F
   C. (i) F (ii) V
   D. (i) F (ii) F

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

13. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Dans le traitement des entrées-sorties, la copie en masse des entrées-sorties :
    (i) scrute toutes les entrées et copie leur état en RAM ;
    (ii) est plus rapide qu'une actualisation continue.

    Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
    A. (i) V (ii) V
    B. (i) V (ii) F
    C. (i) F (ii) V
    D. (i) F (ii) F
14. Le temps de cycle d'un API correspond au temps nécessaire à :
    A. Lire un signal d'entrée
    B. Lire tous les signaux d'entrée
    C. Comparer tous les signaux d'entrée au programme
    D. Lire toutes les entrées, exécuter le programme et affecter toutes les sorties

15. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Un API ayant un temps de cycle long est adapté :
    (i) aux entrées brèves ;
    (ii) aux entrées à haute fréquence.

    Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
    A. (i) V (ii) V
    B. (i) V (ii) F
    C. (i) F (ii) V
    D. (i) F (ii) F
16. Indiquez (a) le bit de parité impaire et (b) le bit de parité paire utilisé pour la transmission des données 1010100.
17. Expliquez le rôle du bit de parité.
18. Expliquez l'actualisation continue et la copie en masse des entrées-sorties utilisées pour le traitement des entrées-sorties.
19. Quelles résistances d'entrée et de contre-réaction doivent être utilisées avec un amplificateur opérationnel inverseur pour obtenir un gain de −100 ?
20. Comparez les topologies réseau en étoile, en bus et en anneau, ainsi que les méthodes employées pour éviter les problèmes de transmission des messages.
21. Quelles sont les fonctions de (a) PROFIBUS DP et PROFIBUS PA, (b) de ControlNet et de DeviceNet ?
22. Que signifie un réseau à passage de jeton ?

4.10 Recherches complémentaires

1. Recherchez les types de réseau avec lesquels les API d'un fabricant sont compatibles.

Copyright © 2015 Dunod.

5 • LANGAGE À CONTACTS ET DIAGRAMMES DE SCHÉMAS FONCTIONNELS

Dans les systèmes à base de microprocesseur, les programmes sont chargés sous forme de *code machine*, c'est-à-dire sous forme d'une suite de nombres binaires qui représentent les instructions. Pour faciliter l'écriture des programmes, il est possible d'utiliser un *langage assembleur*, qui se fonde sur des mnémoniques. Par exemple, le mnémonique LD désigne une opération de chargement des données indiquées après LD. Un programme informatique, appelé *assembleur*, convertit ensuite les mnémoniques en code machine. L'utilisation d'un *langage de haut niveau*, comme C, BASIC, Pascal, FORTRAN ou COBOL, facilite encore plus la programmation. Ces langages prédéfinissent des fonctions, qui sont représentées par des mots simples ou des symboles qui les décrivent. Par exemple, dans le langage C, le symbole & est utilisé pour l'opération ET logique. Cependant, l'utilisation de ces langages nécessite des connaissances en programmation, alors que les API sont destinés à des ingénieurs dont les compétences dans ce domaine peuvent être limitées. Par conséquent, le *langage à contacts* (LD, *Ladder Diagram*) a été développé pour l'écriture des programmes. Ils sont ensuite convertis en code machine par un logiciel et sont utilisés par le microprocesseur d'un API. Cette méthode d'écriture des programmes a été adoptée par la plupart des fabricants d'API, mais chacun a eu tendance à développer ses propres versions. C'est ainsi qu'une norme internationale a été établie pour le langage à contacts et, par voie de conséquence, pour toutes les méthodes de programmation employées avec les API. Cette norme, publiée en 1993 par la Commission électrotechnique internationale, est désignée sous la référence CEI 61131-3 (voir Section 1.5.1). Les *diagrammes de schémas fonctionnels* (FBD, *Function Block Diagram*) constituent une autre méthode de programmation.

Ce chapitre est une introduction à la programmation des API à l'aide du langage à contacts et des diagrammes de schémas fonctionnels. Nous nous intéressons aux techniques de base mises en œuvre dans le développement de programmes à l'aide de ces deux méthodes, de manière à représenter des opérations de commutation de base qui impliquent les fonctions logiques ET, OU, OU exclusif, NON-ET et NON-OU, ainsi que le verrouillage. Dans les chapitres ultérieurs, nous continuerons à utiliser le langage à contacts, mais avec d'autres éléments.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

5.1 Langage à contacts

Pour présenter le langage à contacts, prenons le schéma du circuit électrique illustré à la Figure 5.1a, qui permet de démarrer et d'arrêter un moteur électrique. Nous pouvons redessiner ce schéma de manière différente, en utilisant deux lignes verticales pour représenter les sources d'alimentation et en insérant les autres éléments du circuit entre ces deux lignes (voir Figure 5.1b). Les deux versions placent un interrupteur en série avec un moteur, qui est alimenté lorsque l'interrupteur est fermé. La Figure 5.1b présente un *schéma à contacts*.

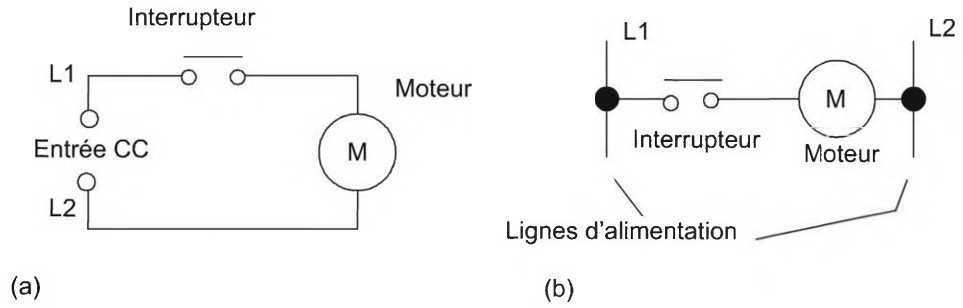


Figure 5.1 – Deux manières de dessiner le même circuit électrique.

Dans un tel schéma, l'alimentation est toujours représentée sous forme de lignes verticales, les autres éléments du circuit étant placés sur des lignes horizontales. Les lignes d'alimentation, ou *barres*, s'apparentent aux montants verticaux d'une échelle et les lignes horizontales, aux barreaux ; d'où le nom *Ladder* (échelle) en anglais. La ligne horizontale présente uniquement les parties de commande du circuit ; dans le cas illustré à la Figure 5.1, il s'agit de l'interrupteur en série avec le moteur. Les schémas électriques révèlent souvent l'emplacement physique relatif des composants du circuit et leur câblage réel. Dans un schéma à contacts, rien n'indique les emplacements physiques, l'intérêt étant porté sur la mise en œuvre de la commande.

La Figure 5.2 montre un exemple de schéma à contacts pour un circuit qui permet de démarrer et d'arrêter un moteur à partir de boutons-poussoirs. Dans l'état normal, le bouton-poussoir 1 est ouvert et le bouton-poussoir 2 est fermé. Lors d'un appui sur le bouton 1, le circuit du moteur est terminé et le moteur démarre. Par ailleurs, les contacts de maintien branchés en parallèle du moteur se ferment et le restent tant que le moteur fonctionne. Ainsi, lorsque le bouton 1 est relâché, les contacts de maintien conservent le circuit et donc l'alimentation du moteur. L'arrêt du moteur se fait par un appui sur le bouton 2. L'alimentation du moteur est alors coupée et les contacts de maintien s'ouvrent. Ainsi, lorsque le bouton 2 est relâché, le moteur n'est plus alimenté. Nous avons donc un moteur qui est démarré par un appui sur le bouton 1 et arrêté par un appui sur le bouton 2.

Copyright © 2015 Dunod.

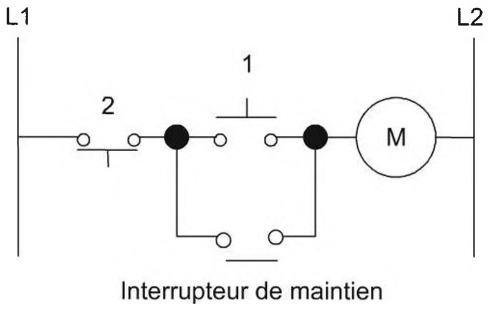


Figure 5.2 – Interrupteur marche-arrêt.

5.1.1 Programmation

La programmation des API se fonde très souvent sur les *schémas à contacts*. L'écriture d'un programme équivaut à tracer un circuit de commutation. Le schéma à contacts est constitué de deux lignes verticales, qui représentent les barres d'alimentation, les éléments du circuit étant connectés sous forme de lignes horizontales entre ces deux lignes verticales.

Pour le tracé d'un schéma à contacts, voici les conventions à respecter :

- Les lignes verticales du schéma représentent les barres d'alimentation, entre lesquelles des circuits sont connectés. Le flux du courant part de la ligne verticale de gauche et traverse une ligne horizontale.
- Chaque ligne horizontale définit une opération du processus de commande. La Figure 5.3 illustre l'analyse effectuée par l'API. La ligne supérieure est lue de gauche à droite. Ensuite, la deuxième ligne est lue de gauche à droite. La procédure se répète pour les lignes suivantes. Lorsque l'API est en mode exécution, il parcourt l'intégralité du programme jusqu'à la fin, la dernière ligne du programme étant clairement indiquée, puis reprend au début (voir Section 4.6). La procédure de parcours de toutes les lignes du programme est appelée *cycle*. La dernière ligne peut être indiquée par un bloc qui contient le mot END ou RET, pour retour, car le programme reprend au début. Le *temps d'analyse* dépend du nombre de pas du programme. S'il faut environ 1 ms pour 1 000 octets de programme, le temps d'analyse se trouve entre 10 et 50 ms.
- Chaque ligne doit commencer par une ou plusieurs entrées et doit se terminer par au moins une sortie. Le terme *entrée* correspond à une action de commande, comme fermer les contacts d'un interrupteur utilisé en entrée de l'API. Le terme *sortie* correspond à un dispositif, comme un moteur, connecté à la sortie d'un API. Les sorties ne sont pas affectées immédiatement au cours de l'analyse du programme. Les résultats des instructions sont placés en mémoire et toutes les sorties sont affectées simultanément à la fin de l'analyse du programme (voir Section 4.6).
- Les dispositifs électriques sont représentés dans leur condition normale. Ainsi, un interrupteur qui est normalement ouvert jusqu'à ce qu'un objet le ferme est représenté ouvert sur le schéma à contacts. Un interrupteur normalement fermé est représenté fermé.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

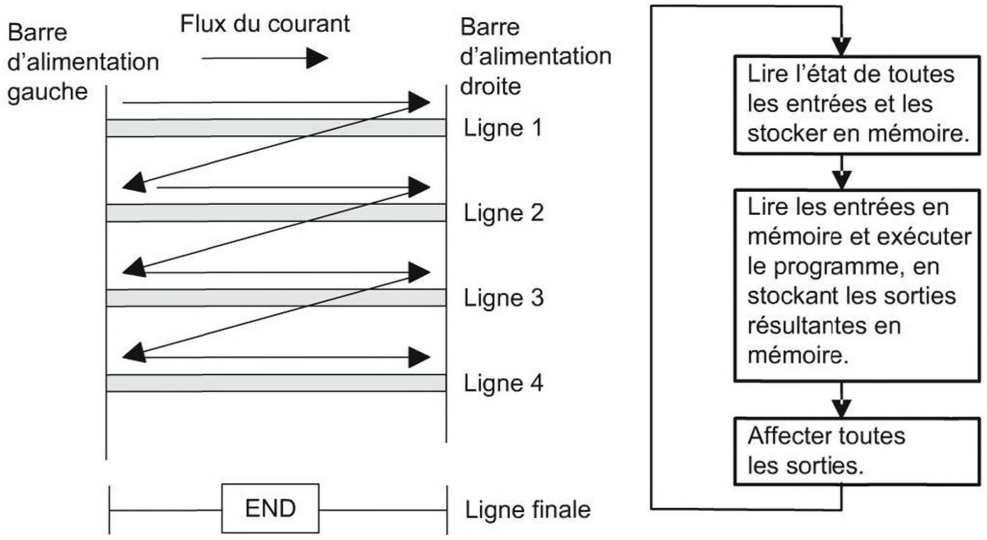


Figure 5.3 – Analyse du schéma à contacts.

- Un même dispositif peut apparaître sur plusieurs lignes du schéma. Par exemple, un relais peut commuter un ou plusieurs systèmes. Les mêmes lettres et/ou numéros sont utilisés comme libellés du dispositif dans chaque cas.
- Les entrées et les sorties sont toutes identifiées par leur adresse, dont le format dépend du fabricant de l'API. Il s'agit de l'adresse de l'entrée ou de la sortie dans la mémoire de l'API (voir Section 4.6).

La Figure 5.4 présente les symboles standard définis par la CEI 61131-3 et utilisés pour les dispositifs d'entrées et de sorties. Il existe une légère différence entre leurs représentations graphique et semi-graphique. La version semi-graphique est obtenue en utilisant les touches d'un clavier normal, tandis que la version graphiques est obtenue à l'aide d'outils de dessin. Vous remarquerez que les entrées sont représentées par différents symboles qui illustrent des contacts normalement ouverts ou normalement fermés. L'action de l'entrée équivaut à l'ouverture ou à la fermeture d'un interrupteur. Les bobines de sortie sont représentées par un seul type de symbole. Nous verrons d'autres symboles dans les chapitres ultérieurs. Le nom de la variable associée à l'adresse est affiché directement au-dessus du symbole (par exemple, X400 pour un interrupteur de mise en marche en entrée et Y430 pour le moteur 1 en sortie).

Pour illustrer le tracé des lignes horizontales d'un schéma à contacts, prenons le cas où la mise sous tension d'un dispositif de sortie, comme un moteur, est déclenchée par la fermeture d'un interrupteur de départ normalement ouvert. La Figure 5.5a présente le schéma à contacts. En commençant par l'entrée, nous avons le symbole du contact normalement ouvert `| |`. Il n'y a aucun autre dispositif d'entrée et la ligne se termine par la sortie, représentée par le symbole `( )`. Lorsque l'interrupteur est fermé, c'est-à-dire lorsqu'il y a une entrée, la sortie du moteur est activée. Tout pendant que l'entrée est présente sur les contacts, il existe une sortie. Si le contact avait été de type normalement fermé `|/|` (voir Figure 5.5b), la sortie aurait été présente pendant que l'interrupteur était ouvert. Dans ce cas, ce n'est qu'en absence d'entrée sur les contacts qu'il y a une sortie.

Copyright © 2015 Dunod.

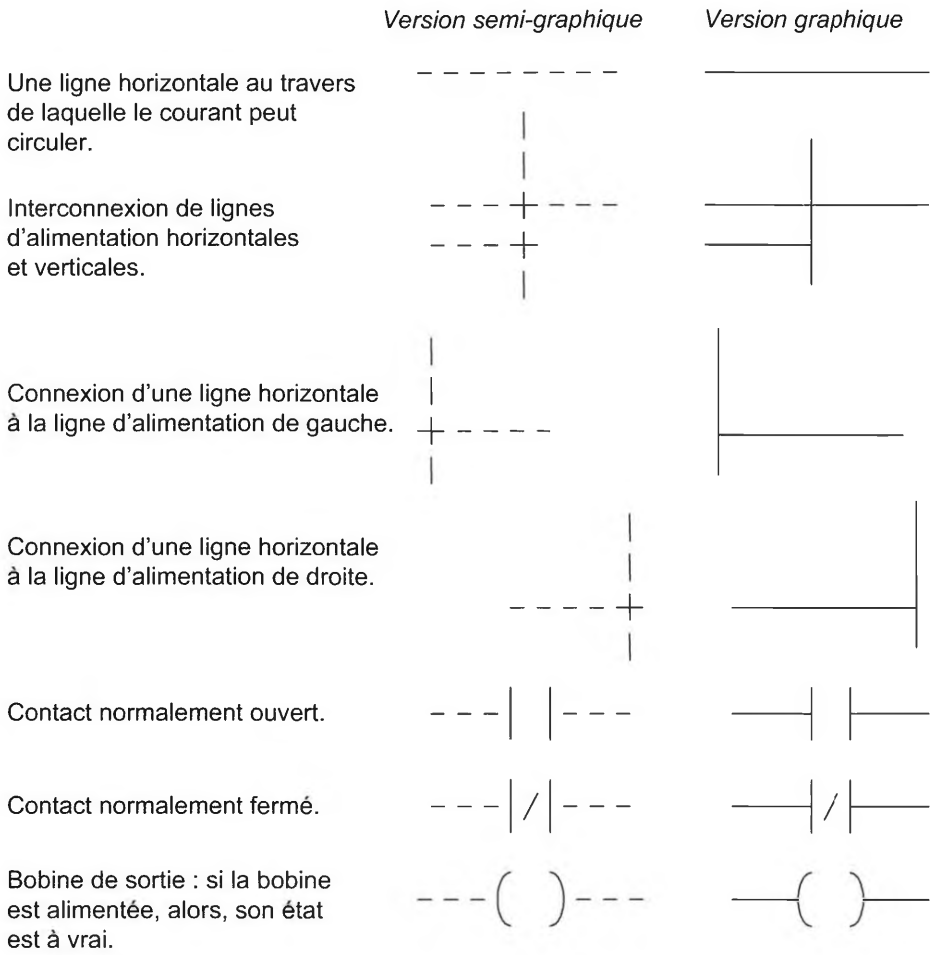


Figure 5.4 – Symboles de base.

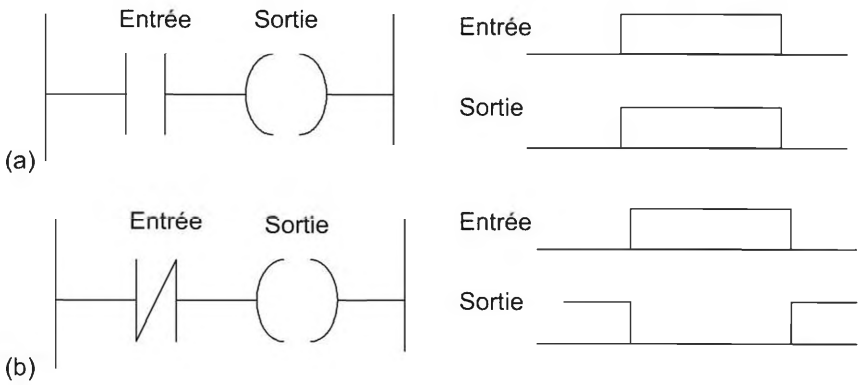


Figure 5.5 – Ligne d'un schéma à contacts.

Dans un schéma à contacts, le nom de la variable associée à chaque élément et son adresse sont ajoutés au symbole. Il est préférable de choisir un nom descriptif, par exemple *interrupteur de commande du moteur de la pompe*, à la place d'un simple *entrée*, et *moteur de la pompe*, à la place d'un simple *sortie*. La Figure 5.6 présente le schéma à contacts de la Figure 5.5a tel qu'il serait dessiné avec les adresses au format (a) Mitsubishi, (b) Siemens, (c) Allen-Bradley et (d) Telemecanique. La

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

Figure 5.6a montre ainsi que cette ligne du schéma à contacts a une entrée à l'adresse X400 et une sortie à l'adresse Y430. Lors de la connexion des entrées et des sorties à l'API, les dispositifs correspondants doivent être connectés aux ports possédant ces adresses.

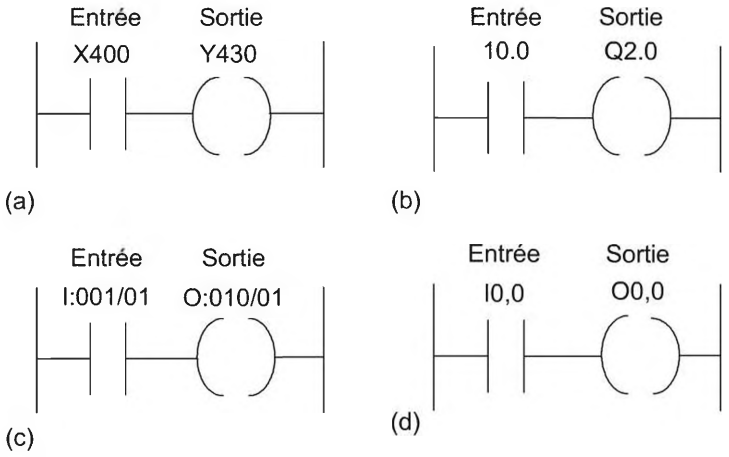


Figure 5.6 – Ajout des adresses : au format (a) Mitsubishi, (b) Siemens, (c) Allen-Bradley et (d) Telemecanique.

5.2 Fonctions logiques

Dans de nombreux cas, les actions doivent être déclenchées lorsqu'une certaine combinaison de conditions est vérifiée. Par exemple, pour une perceuse automatique (voir Figure 1.1a), le démarrage du moteur du foret peut dépendre d'interrupteurs de fin de course qui signalent la présence de la pièce et la proximité du foret avec la pièce. Une telle situation implique une fonction logique ET : la condition A *et* la condition B doivent être vérifiées pour qu'une sortie se produise. Cette section se focalise sur ces fonctions logiques.

5.2.1 ET

La Figure 5.7a illustre le cas où la sortie n'est pas activée tant que deux interrupteurs normalement ouverts ne sont pas tous les deux fermés. Puisque l'interrupteur A *et* l'interrupteur B doivent être fermés, nous avons un cas de logique ET (*AND*). Nous pouvons voir cela comme un système de commande avec deux entrées, A et B (Figure 5.7b). La sortie n'est présente que si les entrées A *et* B sont toutes deux présentes. Ainsi, si nous utilisons 1 pour indiquer la présence du signal et 0 pour présenter son absence, les entrées A *et* B doivent être à 1 pour que la sortie soit à 1. Une telle opération est mise en œuvre par une *porte logique* et la relation entre les entrées et les sorties d'une porte logique est présentée sous forme d'une *table de vérité*. Voici la table de vérité d'une porte ET :

Copyright © 2015 Dunod.

Entrées		Sortie
A	B	
0	0	0
0	1	0
1	0	0
1	1	1

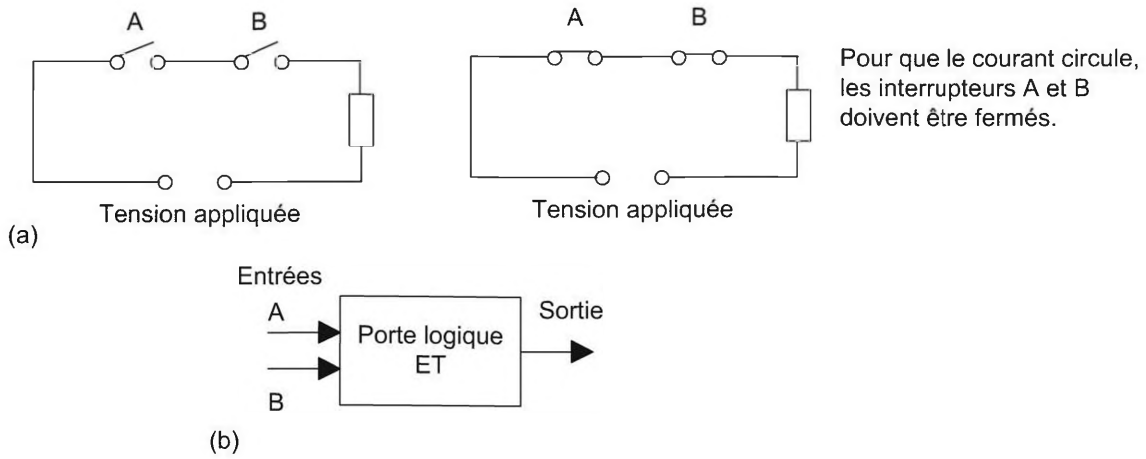


Figure 5.7 – (a) Un circuit ET et (b) une porte logique ET.

Une porte ET est par exemple utilisée dans le système de verrouillage d'une machine-outil pour qu'elle ne démarre que si le dispositif de sécurité est en place et si elle est sous tension.

La Figure 5.8a montre un système à porte ET sur un schéma à contacts. Le schéma commence par | |, un ensemble de contacts normalement ouverts intitulé Entrée A, pour représenter l'interrupteur A. Il est placé en série avec un autre symbole | |, également un ensemble de contacts normalement ouverts, intitulé Entrée B, pour représenter l'interrupteur B. La ligne se termine par ( ) pour représenter la sortie. Pour que la sortie soit active, les entrées A et B doivent se produire. Autrement dit, les contacts Entrée A et Entrée B doivent être fermés (voir Figure 5.8b). En général :

> « *Dans un schéma à contacts, les contacts placés sur une ligne horizontale, c'est-à-dire des contacts en série, représentent les opérations logiques ET.* »

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

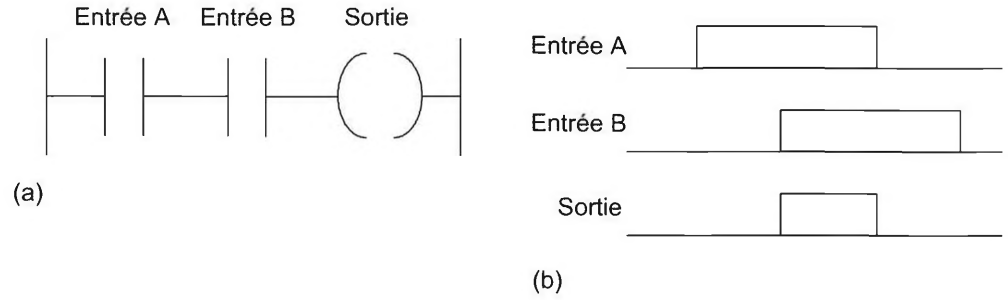


Figure 5.8 – Porte ET sur une ligne d'un schéma à contacts.

5.2.2 OU

La Figure 5.9a présente un circuit électrique dans lequel la sortie est alimentée lorsque l'interrupteur A *ou* l'interrupteur B, tous deux normalement ouverts, est fermé. Il s'agit d'une porte logique OU (*OR*) dans laquelle l'entrée A *ou* l'entrée B doit être présente pour que la sortie soit présente (voir Figure 5.9b). Voici la table de vérité correspondante :

Entrées		Sortie
A	B	
0	0	0
0	1	1
1	0	1
1	1	1

La Figure 5.10a montre un système à porte OU sur un schéma à contacts. La Figure 5.10b est une représentation équivalente du même système. Le schéma commence par | |, un contact normalement ouvert intitulé Entrée A, pour représenter l'interrupteur A. Il est placé en parallèle avec un autre symbole | |, également un contact normalement ouvert, intitulé Entrée B, pour représenter l'interrupteur B. Pour que la sortie soit active, Entrée A ou Entrée B doit être fermée (voir Figure 5.10c). La ligne se termine par ( ) pour représenter la sortie. En général :

> « *Des chemins alternatifs fournis par des lignes verticales à partir de la ligne principale d'un schéma contacts, c'est-à-dire des chemins en parallèle, représentent les opérations logiques OU.* »

Une porte OU est par exemple utilisée dans une bande transporteuse qui convoie des bouteilles en vue de leur mise en carton. Un déflecteur est activé pour dévier une bouteille vers un bac de rebut si son poids n'est pas dans un certain intervalle ou si la capsule n'est pas posée.

Copyright © 2015 Dunod.

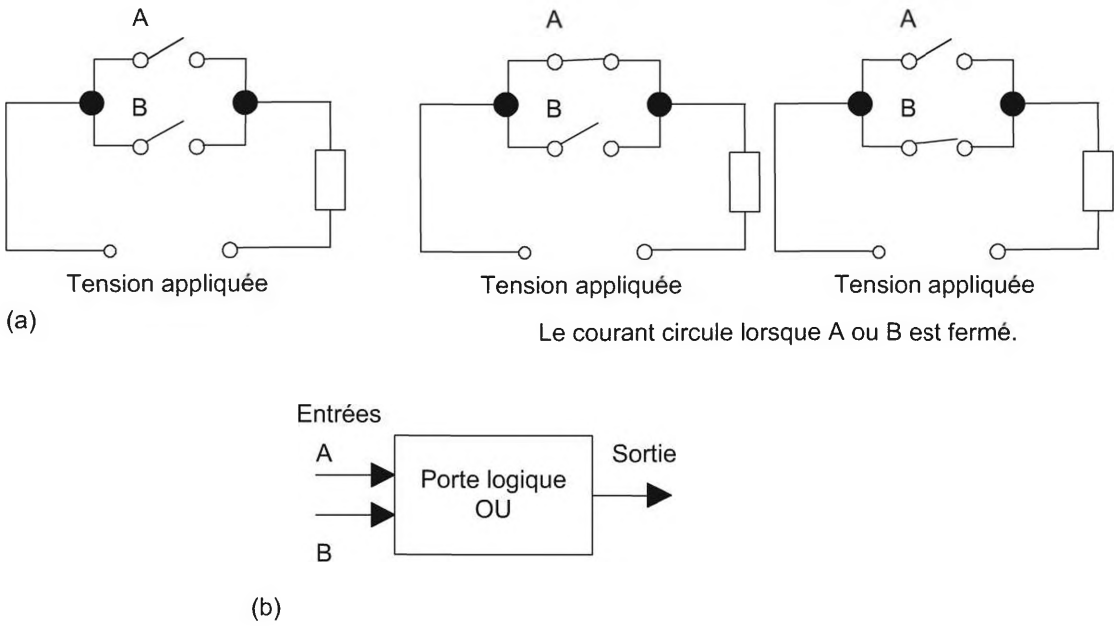


Figure 5.9 – (a) Un circuit OU et (b) une porte logique OU.

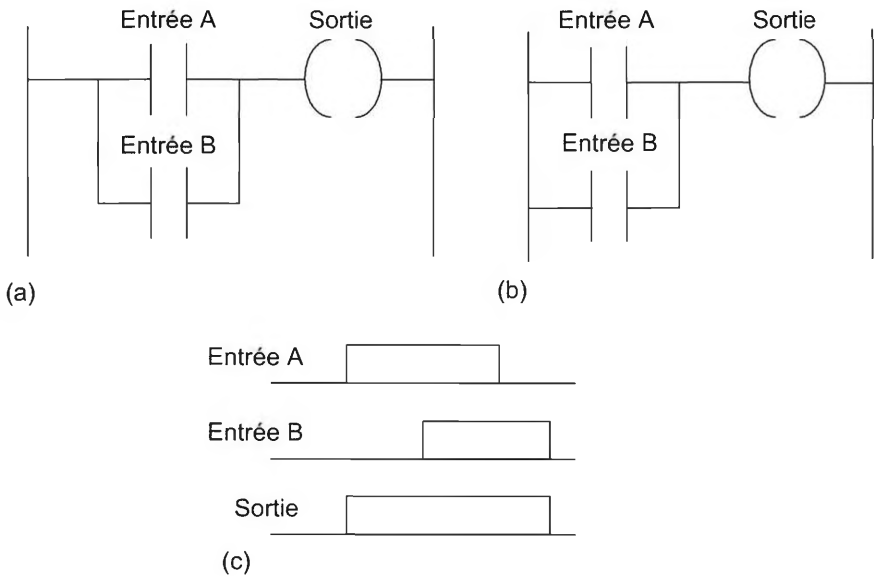


Figure 5.10 – Porte OU sur une ligne d'un schéma à contacts.

5.2.3 NON

La Figure 5.11a présente un circuit électrique commandé par un interrupteur normalement fermé. Lorsqu'il existe une entrée sur l'interrupteur, il s'ouvre et le courant ne circule plus dans le circuit. Il s'agit d'une porte logique NON (*NOT*) dans laquelle la sortie est présente lorsque l'entrée est absente, et vice versa (voir Figure 5.11b). C'est pourquoi cette porte est parfois appelée *inverseur*. Voici la table de vérité correspondante :

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

Entrée	Sortie
A	
0	1
1	0

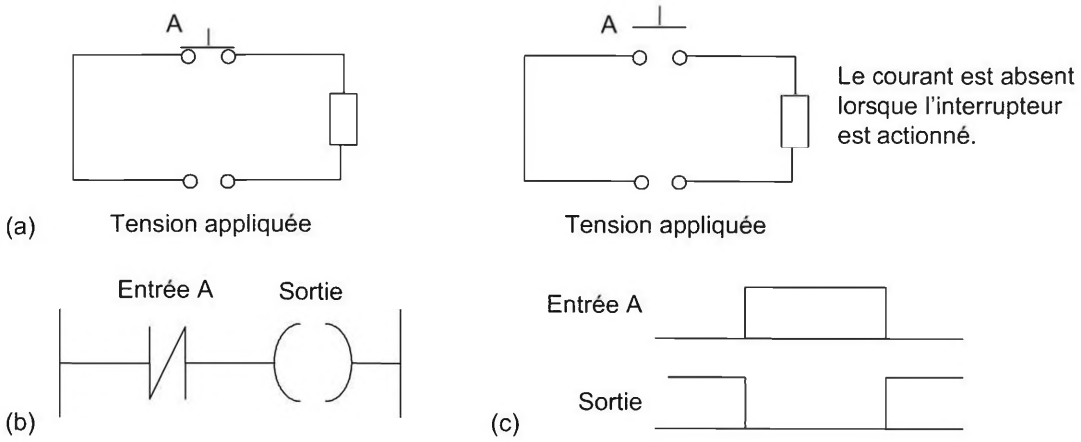


Figure 5.11 – (a) Un circuit NON, (b) une porte logique NON dans un schéma à contacts et (c) une sortie au niveau haut en cas d'absence d'entrée sur A.

La Figure 5.11b montre un système à porte NON sur un schéma à contacts. Le contact Entrée A est représenté normalement ouvert. Cette entrée se trouve en série avec la sortie ( ). Lorsque l'entrée sur A est absente, les contacts sont fermés et la sortie est présente. En cas d'entrée sur A, les contacts s'ouvrent et il n'y a pas de sortie.

Une porte NON est par exemple utilisée avec une lampe qui s'allume lorsqu'il fait nuit. Autrement dit, en cas d'absence de lumière sur le capteur de luminosité, la sortie est active.

5.2.4 NON-ET

Supposons que nous faisions suivre une porte ET d'une porte NON (voir Figure 5.12a). Cette porte NON a pour effet d'inverser la sortie de la porte ET. Une autre solution, qui produit exactement le même résultat, consiste à placer une porte NON sur chaque entrée et d'ajouter une porte OU (voir Figure 5.12b). Nous obtenons alors la même table de vérité suivante :

Copyright © 2015 Dunod.

Entrées		Sortie
A	B	
0	0	1
0	1	1
1	0	1
1	1	0

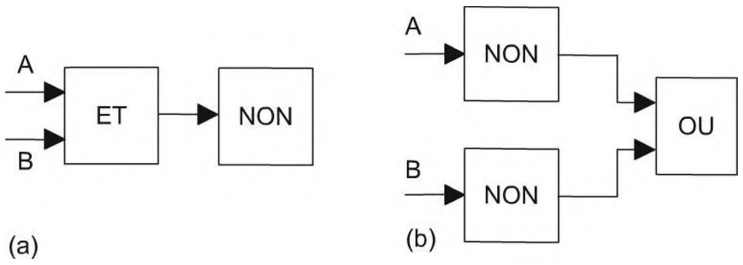


Figure 5.12 – Porte NON-ET.

Les entrées A et B doivent être à 0 pour que la sortie soit à 1. La sortie est présente lorsque l'entrée A et l'entrée B ne sont pas à 1. La combinaison de ces portes est appelée *porte NON-ET* (*NAND*).

La Figure 5.13 illustre un schéma à contacts qui donne une porte NON-ET. Lorsque Entrée A et Entrée B sont toutes deux à 1, la sortie est à 0. Lorsque Entrée A et Entrée B sont à 0 ou que l'une est à 0 et l'autre à 1, la sortie est à 1.

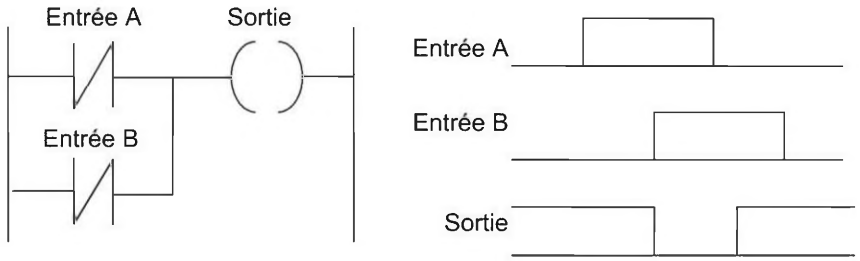


Figure 5.13 – Porte NON-ET sur une ligne d'un schéma à contacts.

Une porte NON-ET est par exemple utilisée lorsqu'une lampe témoin s'allume si, dans une machine-outil, le dispositif de sécurité et l'interrupteur de fin de course signalant la présence de la pièce ne sont pas activés.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

5.2.5 NON-OU

Supposons que nous faisions suivre une porte OU d'une porte NON (voir Figure 5.14a). Cette porte NON a pour effet d'inverser la sortie de la porte OU. Une autre solution, qui produit exactement le même résultat, consiste à placer une porte NON sur chaque entrée et d'ajouter une porte ET (voir Figure 5.14b). Nous obtenons alors la même table de vérité suivante :

Entrées		Sortie
A	B	
0	0	1
0	1	0
1	0	0
1	1	0

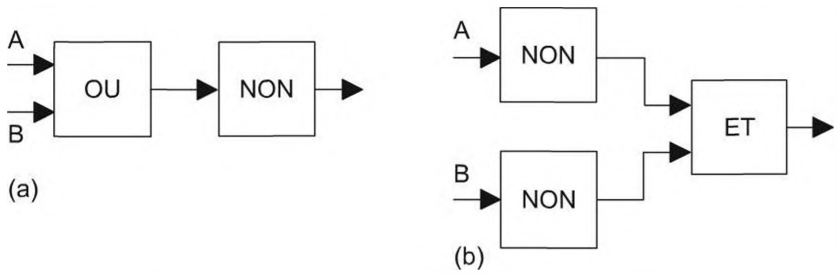


Figure 5.14 – Porte NON-OU.

La combinaison de ces portes OU et NON est appelée *porte NON-OU* (*NOR*). La sortie est présente lorsque ni l'entrée A ni l'entrée B ne sont à 1.

La Figure 5.15 illustre un schéma à contacts qui donne une porte NON-OU. Lorsque ni Entrée A ni Entrée B n'est activée, la sortie est à 1. Lorsque Entrée A ou Entrée B est à 1, la sortie est à 0.

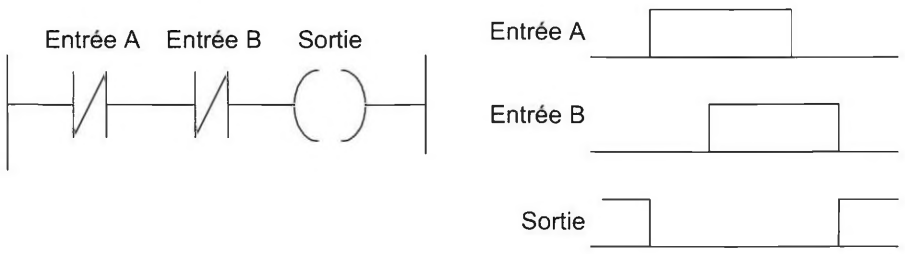


Figure 5.15 – Porte NON-OU sur une ligne d'un schéma à contacts.

Copyright © 2015 Dunod.

5.2.6 OU exclusif

La porte OU donne une sortie lorsque l'une de ses entrées ou les deux sont à 1. Toutefois, on a parfois besoin d'une porte qui donne une sortie lorsque l'une des entrées est à 1, non les deux. Voici la table de vérité correspondante :

Entrées		Sortie
A	B	
0	0	0
0	1	1
1	0	1
1	1	0

Une telle porte est appelée *OU exclusif* (*XOR*). Pour l'obtenir, il suffit d'utiliser des portes NON, ET et OU comme l'illustre la Figure 5.16.

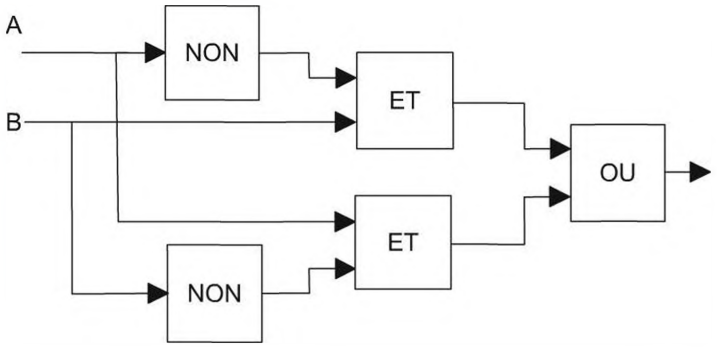


Figure 5.16 – Porte OU exclusif.

La Figure 5.17 montre un schéma à contacts qui représente une porte OU exclusif. Lorsque Entrée A et Entrée B ne sont pas activées, la sortie est absente. Lorsque seule Entrée A est activée, la branche supérieure produit une sortie à 1. Lorsque seule Entrée B est activée, la branche inférieure produit une sortie à 1. Lorsque Entrée A et Entrée B sont activées, il n'y a pas de sortie. Dans cet exemple de portes logiques, Entrée A et Entrée B possèdent deux ensembles de contacts dans les circuits, l'un étant normalement ouvert, l'autre, normalement fermé. Pour la programmation d'un API, chaque entrée est considérée comme disposant d'autant d'ensembles de contacts que nécessaire.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

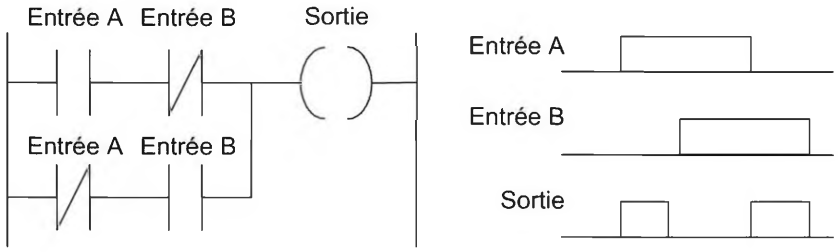


Figure 5.17 – Porte OU exclusif sur une ligne d'un schéma à contacts.

5.3 Verrouillage

Dans de nombreuses situations il est nécessaire de maintenir la sortie alimentée même lorsque l'entrée disparaît. C'est par exemple le cas d'un moteur démarré par l'appui sur un bouton-poussoir. Bien que les contacts de l'interrupteur ne restent pas fermés, le moteur doit continuer à fonctionner jusqu'à l'appui sur un bouton-poussoir d'arrêt. Un *circuit à verrouillage* est un circuit qui permet ce type de fonctionnement. Il s'agit d'un circuit de maintien car, après avoir été excité, il conserve cet état jusqu'à la réception d'une autre entrée.

La Figure 5.18 montre un exemple de circuit à verrouillage. Lorsque les contacts d'Entrée A sont fermés, une sortie est présente. Cependant, lorsque la sortie est présente, un autre ensemble de contacts associé à cette sortie se ferme. Ces contacts constituent une porte logique OU avec les contacts d'entrée. Par conséquent, même si Entrée A s'ouvre, le circuit maintient la sortie activée. La seule manière de libérer la sortie consiste à actionner le contact B normalement fermé.

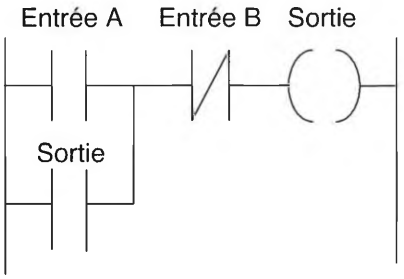


Figure 5.18 – Circuit à verrouillage.

Pour illustrer la mise en application d'un circuit à verrouillage, prenons un moteur commandé par des boutons-poussoirs de démarrage et d'arrêt. Par ailleurs, un témoin lumineux s'allume lorsque le moteur est sous tension, tandis qu'un autre s'allume lorsqu'il ne l'est pas. La Figure 5.19 présente un schéma à contacts avec des adresses au format Mitsubishi. X401 est fermé au démarrage du programme. Lorsque X400 est momentanément fermé, Y430 est excité et ses contacts se ferment. Cela conduit au verrouillage, ainsi qu'à la mise hors tension de Y431 et à la mise sous tension de Y432. Pour arrêter le moteur, X401 est appuyé et s'ouvre. Le contact Y430 s'ouvre dans la ligne supérieure et la troisième, mais se ferme dans la deuxième. Par conséquent, Y431 s'active, tandis que Y432 se désactive.

Copyright © 2015 Dunod.

Le verrouillage est souvent employé avec les démarrages afin que l'activation initiale d'une application soit verrouillée.

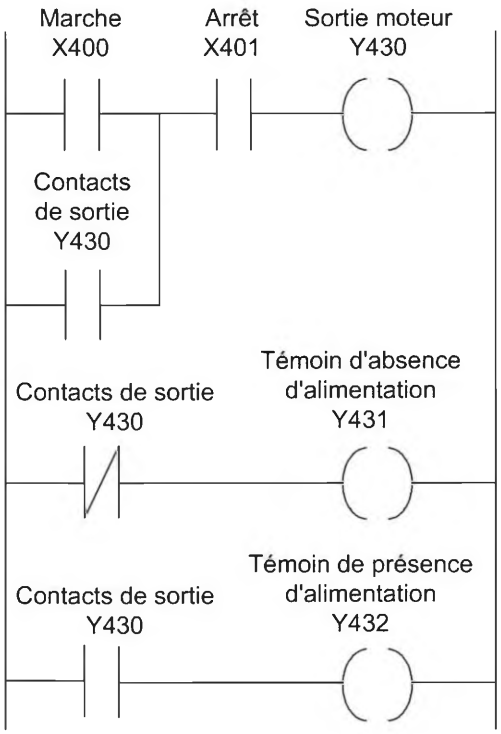


Figure 5.19 – Schéma à contacts d'un moteur marche/arrêt, avec témoins lumineux. Notez que le contact d'arrêt X401 est représenté ouvert. Si l'interrupteur d'arrêt utilisé est normalement fermé, X401 reçoit un signal de démarrage pour se fermer. Cela permet un fonctionnement plus sûr que de programmer X401 en normalement fermé.

5.4 Sorties multiples

Dans un schéma à contacts, plusieurs sorties peuvent être connectées à un contact. La Figure 5.20 présente un tel schéma avec deux bobines de sortie. Lorsque le contact d'entrée se ferme, les deux bobines produisent des sorties.

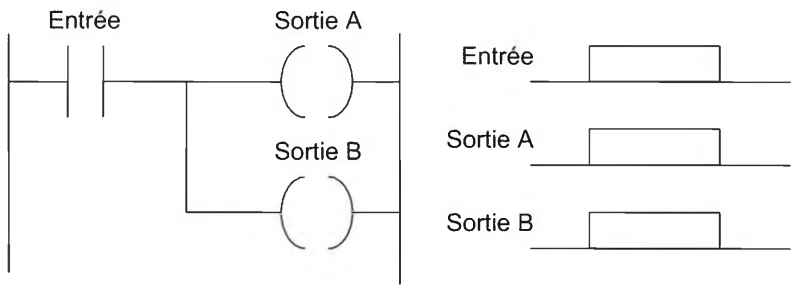


Figure 5.20 – Ligne d'un schéma à contacts avec deux sorties.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

Pour le schéma à contacts de la Figure 5.21, la sortie A se produit lorsque l'entrée A se produit. La sortie B se produit uniquement lorsque l'entrée A et l'entrée B sont présentes.

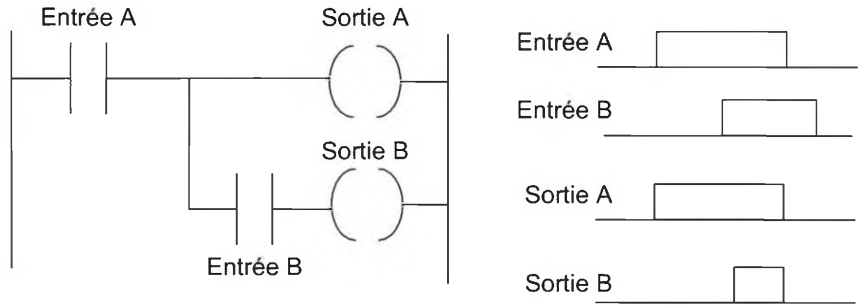


Figure 5.21 – Ligne d'un schéma à contacts avec deux entrées et deux sorties.

Ce type de disposition permet de produire des sorties en séquence, l'ordre étant celui dans lequel les contacts sont fermés. La Figure 5.22 illustre ce principe avec un programme, donné dans les notations de Mitsubishi et de Siemens. Les sorties A, B et C sont activées dans cet ordre lorsque les contacts sont fermés dans l'ordre A, B et C. Tant que l'entrée A n'est pas fermée, aucune des sorties ne peut être activée. Lorsque l'entrée A est fermée, la sortie A est activée. Ensuite, lorsque l'entrée B est fermée, la sortie B est activée. Enfin, lorsque l'entrée C est fermée, la sortie C est activée.

5.5 Saisie des programmes

Chaque ligne horizontale du schéma à contacts représente une instruction dans le programme exécuté par l'API. L'ensemble du schéma correspond au programme complet. Il existe plusieurs méthodes pour saisir le programme depuis un pupitre de programmation ou un ordinateur. Quelle que soit celle employée, le programme doit être placé dans la mémoire de l'API sous une forme compatible avec le microprocesseur. Cette forme est appelée *langage machine* et n'est rien d'autre qu'un code binaire, comme 0010100001110001.

5.5.1 Symboles du langage à contacts

Pour saisir le programme à partir d'un pupitre de programmation, il est possible d'utiliser un clavier dont les touches sont marquées des symboles qui correspondent aux différents éléments du schéma à contacts. En appuyant sur les touches appropriées, le schéma à contacts apparaît sur l'écran du pupitre de programmation. Le terminal traduit ensuite le programme tracé à l'écran en langage machine.

Les ordinateurs peuvent également être utilisés pour tracer les schémas à contacts. Pour cela, il faut commencer par installer le logiciel approprié, par exemple RSLogix de Rockwell Automation Inc. pour les API d'Allen-Bradley, MELSOFT-GX Developer pour les API de Mitsubishi et STEP 7-Micro/WIN V4 pour les

Copyright © 2015 Dunod.

API de Siemens. Les logiciels fonctionnent sous Windows et permettent de choisir les éléments dans des menus déroulants.

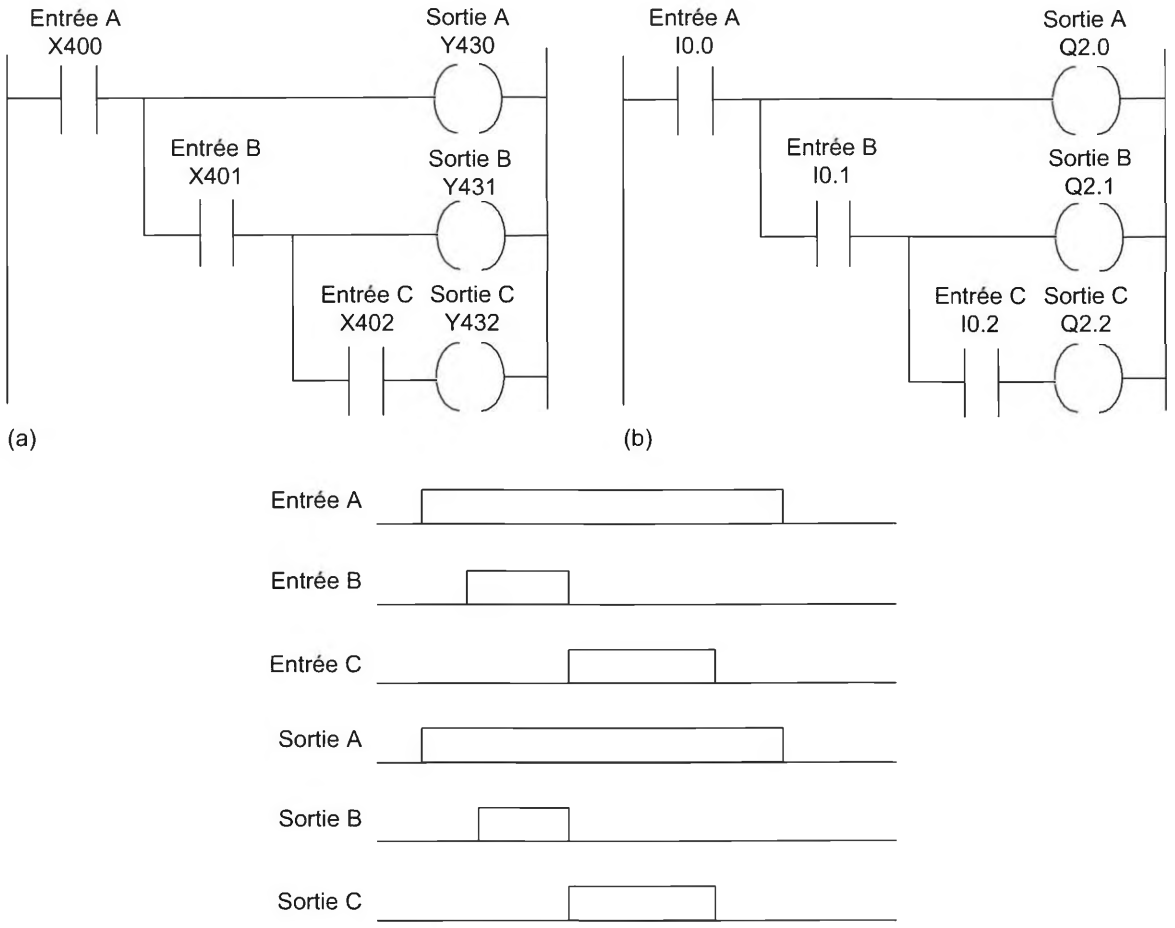


Figure 5.22 – Sorties ordonnées.

5.6 Diagrammes de schémas fonctionnels

Un *diagramme de schémas fonctionnels* (FBD, *Function Block Diagram*) est utilisé pour les programmes d'API décrits sous forme de blocs graphiques. Il s'agit d'un langage graphique pour représenter les signaux et les données passant au travers de blocs, qui sont des éléments logiciels réutilisables. Un *bloc fonctionnel* est une instruction du programme qui, lorsqu'elle est exécutée, produit une ou plusieurs valeurs de sortie. La Figure 5.23 montre la représentation d'un bloc. Le nom de la fonction est écrit dans la boîte.

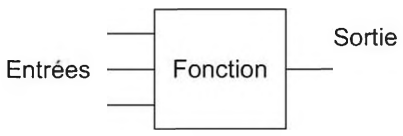


Figure 5.23 – Bloc fonctionnel.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

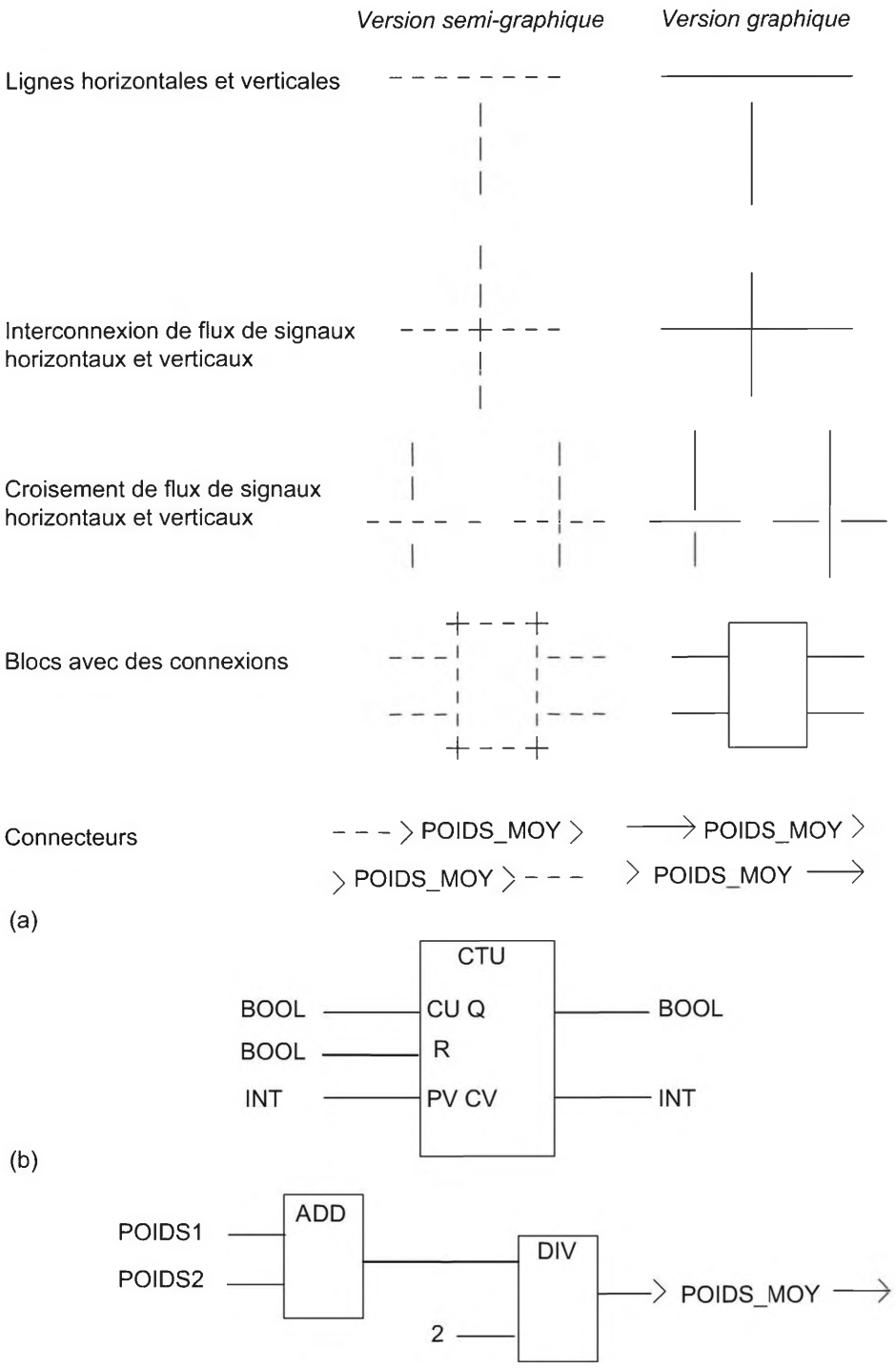


Figure 5.24 – Représentation d'un diagramme de schémas fonctionnels. (a) Un exemple de bloc fonctionnel standard : un compteur CTU qui produit une sortie sur Q lorsque le nombre d'impulsions sur l'entrée CU a atteint la valeur fixée par PV. À chaque impulsion d'entrée, la sortie CV est augmentée de 1. L'entrée R remet à zéro le compteur. Les libellés des entrées et des sorties indiquent le type du signal : BOOL pour booléen et INT pour entier. Consultez le Chapitre 10 pour l'utilisation d'un tel bloc. (b) Un exemple de bloc défini par l'utilisateur pour calculer la moyenne de deux poids. Il peut être réutilisé ensuite dans d'autres blocs fonctionnels.

Copyright © 2015 Dunod.

La norme CEI 61113-3 définit la manière de représenter de tels blocs (voir Figure 5.24). Un bloc fonctionnel est tracé sous forme d'une boîte rectangulaire, avec les entrées placées à gauche et les sorties, à droite. Le type du bloc est inscrit dans la boîte, par exemple ET, tandis que son nom dans le système apparaît au-dessus, Temporisateur1. Les noms des entrées des blocs fonctionnels sont placés à l'intérieur de la boîte aux points d'entrées-sorties appropriés. Des connexions inter-schémas sont utilisées lorsque le tracé des lignes graphiques serait complexe sans encombrer ou compliquer le schéma. La Figure 5.24b montre un exemple de diagramme de schémas fonctionnels.

Les blocs fonctionnels peuvent mettre en œuvre des fonctions standard, comme ceux qui correspondent aux portes logiques, aux compteurs et aux temporisateurs, ou des fonctions définies par l'utilisateur, comme un bloc d'obtention de la valeur moyenne des entrées.

5.6.1 Portes logiques

Les programmes impliquent souvent des portes logiques. Il existe deux formes de symbolisation des portes logiques, l'une provenant des États-Unis, l'autre étant un standard international (IEEE/ANSI) qui se fonde sur un rectangle dans lequel est écrit la fonction. Le 1 dans une boîte indique qu'il existe une sortie si l'entrée est à 1. La fonction OU est indiquée par ≥1 car la sortie est présente si une entrée est supérieure ou égale à 1. Une entrée inversée est représentée par un petit cercle sur l'entrée, tout comme une sortie inversée (voir Figure 5.25). La Figure 5.26 montre les symboles. Dans les diagrammes de schémas fonctionnels, les symboles IEEE/ANSI sont les plus souvent utilisés.

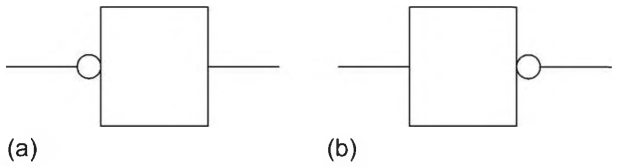


Figure 5.25 – (a) Entrée inversée et (b) sortie inversée.

La Figure 5.27 montre l'effet de ces blocs fonctionnels dans les programmes des API.

Pour illustrer la forme d'un tel diagramme et son lien avec un schéma à contacts, la Figure 5.28 prend l'exemple d'une porte OU. Lorsque l'entrée A ou l'entrée B est à 1, la sortie est présente.

La Figure 5.29 montre un schéma à contacts et le bloc fonctionnel équivalent selon la notation de Siemens. Le bloc = indique une sortie du système.

La Figure 5.30 présente un schéma à contacts qui met en œuvre une sortie dont les contacts servent d'entrée. Le diagramme de schémas fonctionnels équivalent inclut une boucle de retour.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

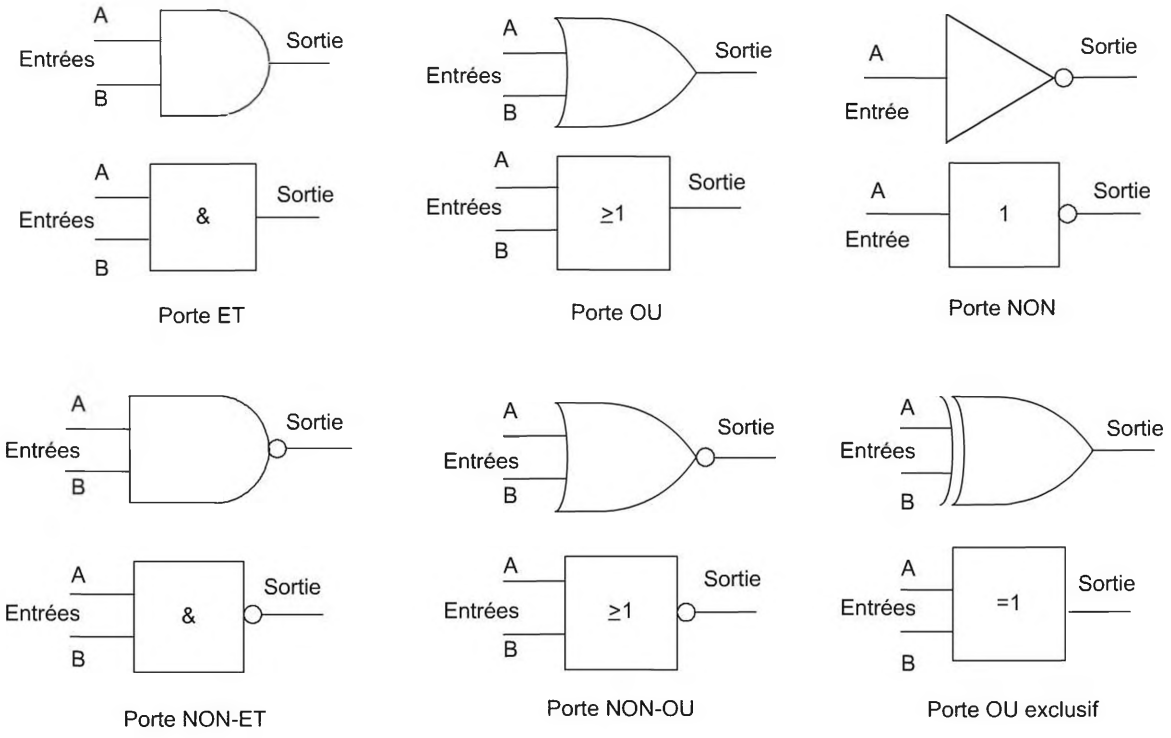


Figure 5.26 – Symboles des portes logiques.

Examinons la création d'un diagramme de schémas fonctionnels et d'un schéma à contacts pour une application dans laquelle une pompe doit être activée pour pomper le contenu d'un réservoir lorsque le bouton de mise en marche est fermé, lorsque le niveau du liquide dans le réservoir est en dessous du niveau requis et lorsque le réservoir contient le liquide. Nous faisons face à une logique ET entre l'entrée de l'interrupteur de démarrage et l'entrée d'un capteur qui est au niveau haut tant que le liquide dans le réservoir se trouve sous le niveau requis. Ces deux conditions se retrouvent également dans une logique ET avec un interrupteur qui signale la présence de liquide dans le réservoir. Supposons que cet interrupteur produise une entrée lorsque du liquide est présent. Le diagramme de schémas fonctionnels qui correspond à ce cas et le schéma à contacts équivalent sont donnés à la Figure 5.31.

5.6.2 Algèbre booléenne

Puisque nous utilisons un système de logique mathématique, il est possible de dériver des programmes en langage à contacts à partir des expressions booléennes. Dans l'algèbre booléenne, seuls deux chiffres sont utilisés : 0 et 1. Lorsque nous avons une opération ET entre les entrées A et B, nous pouvons écrire :

$$A \cdot B = Q$$

où Q représente la sortie. Ainsi, Q est égale à 1 uniquement lorsque A = 1 et B = 1.

Copyright © 2015 Dunod.

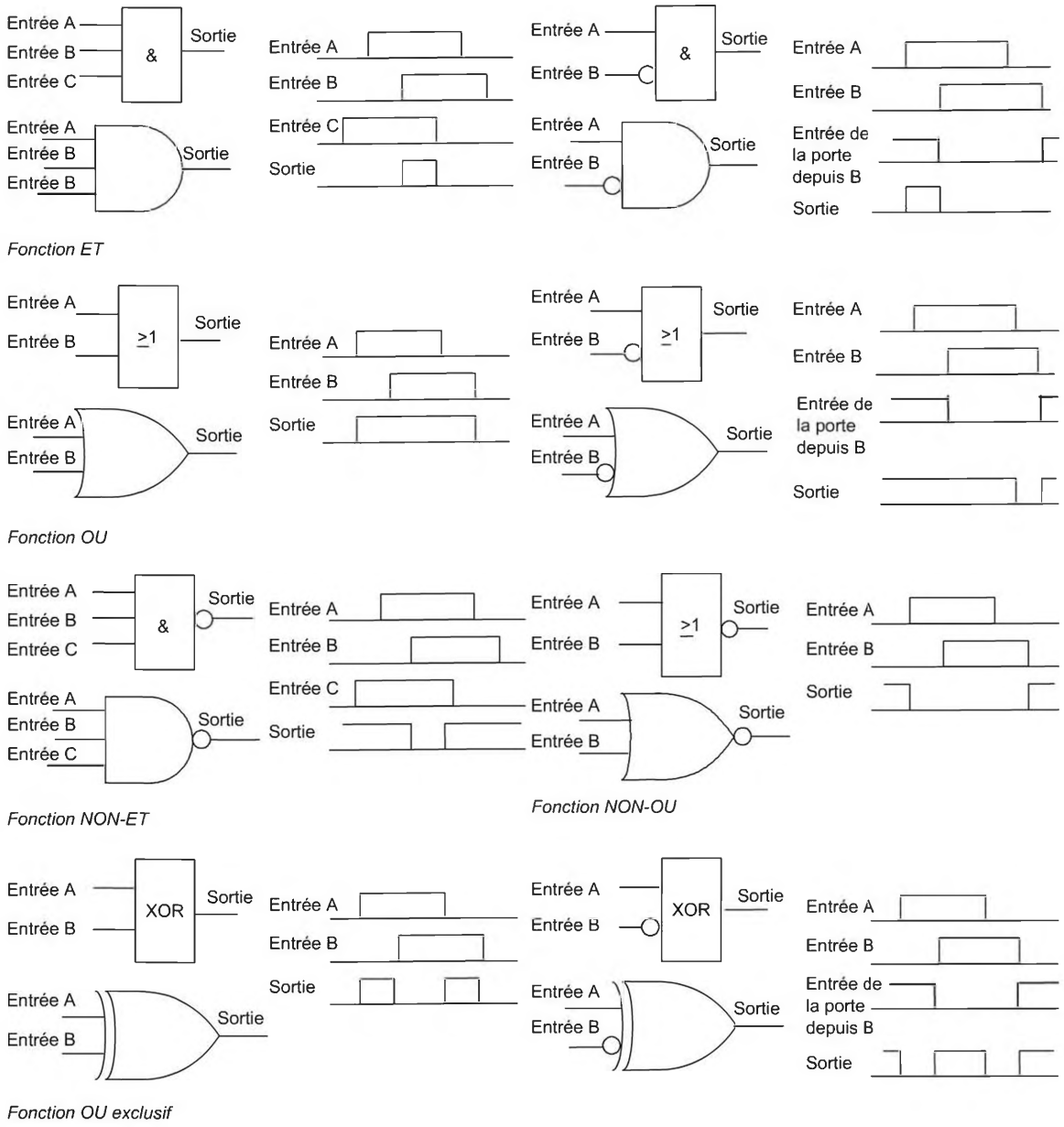


Figure 5.27 – Blocs fonctionnels des portes logiques.

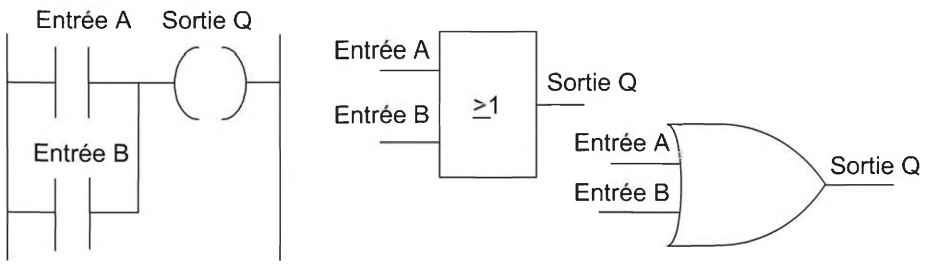


Figure 5.28 – Schéma à contacts et diagramme de schémas fonctionnels équivalent.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

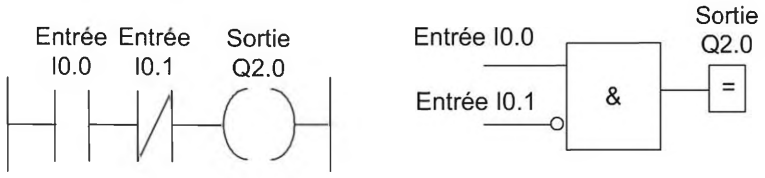


Figure 5.29 – Schéma à contacts et diagramme de schémas fonctionnels équivalent.

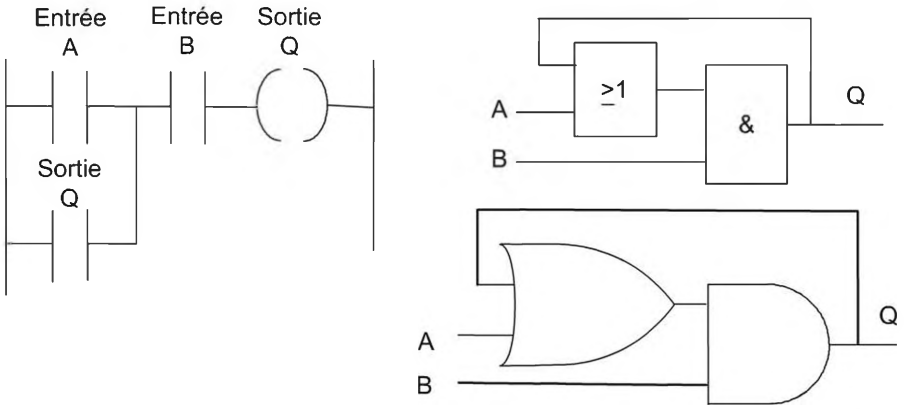


Figure 5.30 – Schéma à contacts et diagramme de schémas fonctionnels équivalent.

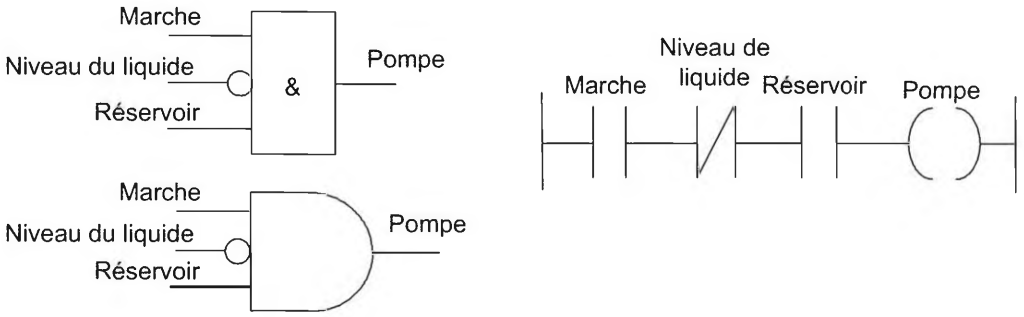


Figure 5.31 – Mise en œuvre de la pompe.

Une opération OU entre les entrées A et B s'écrit de la manière suivante :

$$A + B = Q$$

Dans ce cas, Q est égale à 1 uniquement lorsque A = 1 ou B = 1. Voici comment écrire l'opération NON sur l'entrée A :

$$\overline{A} = Q$$

Lorsque A n'est pas à 1, la sortie est présente.

Pour illustrer la manière de lier les expressions booléennes aux schémas à contacts, prenons l'expression suivante :

$$A + B \cdot C = Q$$

Copyright © 2015 Dunod.

Pour que la sortie Q soit présente, il faut que A ou B et C soient présentes. La Figure 5.32 montre le schéma à contacts et le diagramme de schémas fonctionnels correspondants. En utilisant la notation de Mitsubishi, l'expression précédente devient :

$$X400 + X401 \cdot X402 = Y430$$

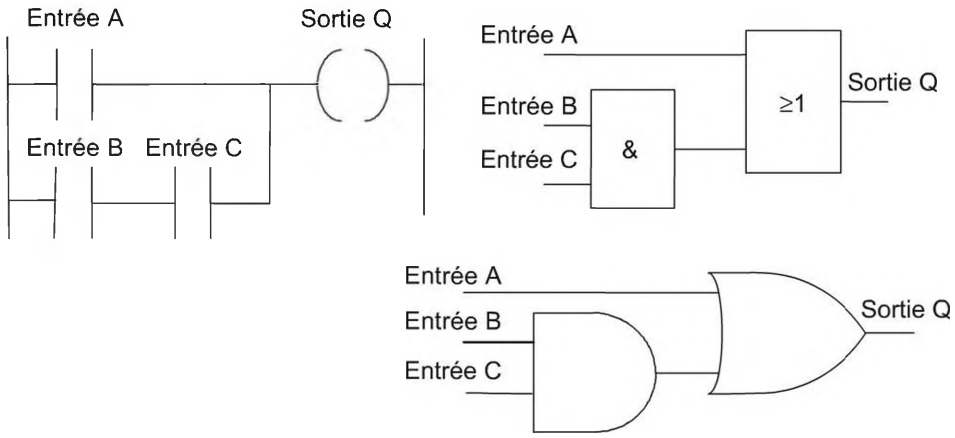


Figure 5.32 – Schéma à contacts et diagramme de schémas fonctionnels.

Voici sa version dans la notation de Siemens :

$$I0:0 + I0:1 \cdot I0:2 = Q2:0$$

Pour compléter notre illustration, prenons l'expression booléenne suivante :

$$A + \overline{B} = Q$$

La Figure 5.33 montre le schéma à contacts et le diagramme de schémas fonctionnels correspondants. Selon la notation de Mitsubishi, l'expression s'écrit de la manière suivante :

$$X400 + \overline{X401} = Y430$$

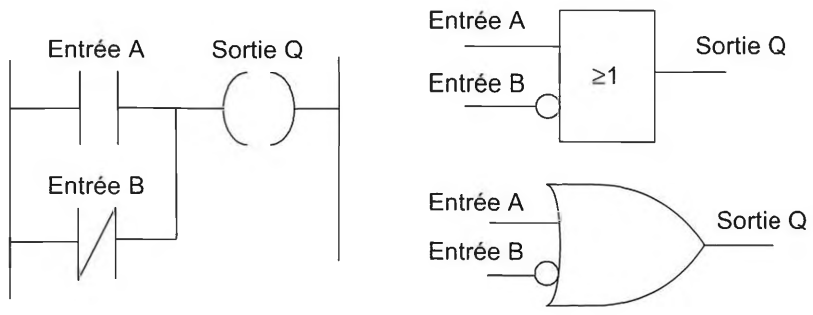


Figure 5.33 – Schéma à contacts et diagramme de schémas fonctionnels.

Voici la même expression selon la notation de Siemens :

$$I0:0 + \overline{I0:1} = Q2.0$$

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

Examinons à présent la porte OU exclusif constituée de portes NON, ET et OU (voir Figure 5.34).

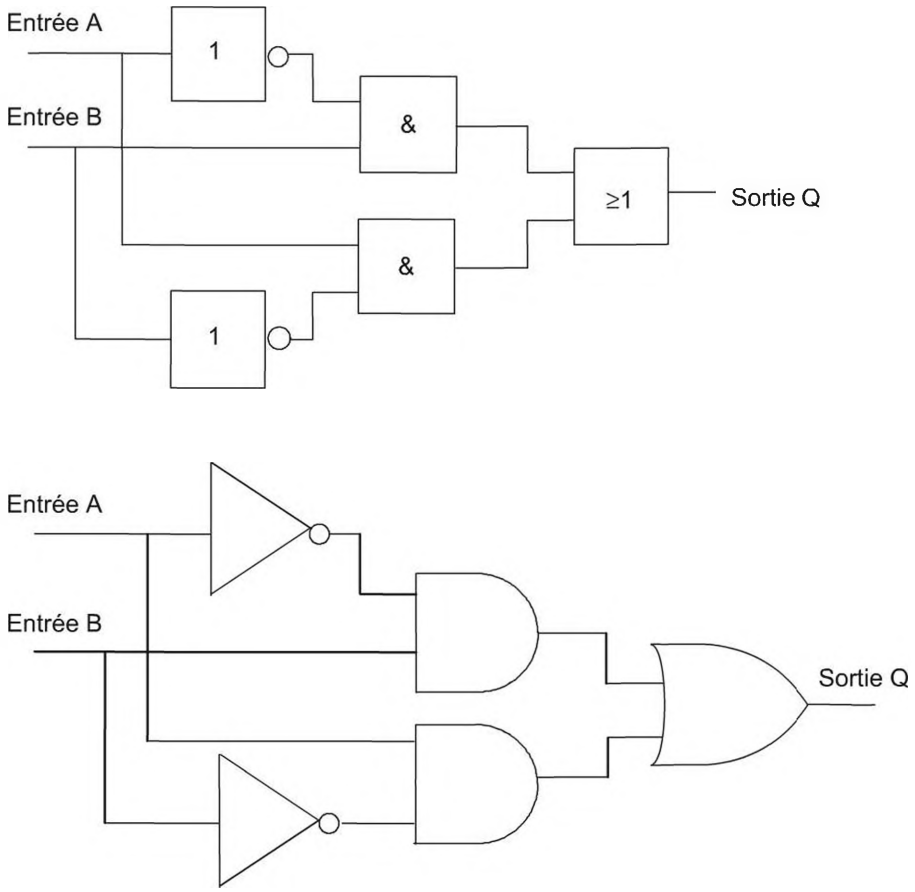


Figure 5.34 – Une porte OU exclusif.

Les entrées de la porte ET inférieure sont A et $\overline{B}$. Par conséquent, sa sortie est donnée par l'expression suivante :

$$A \cdot \overline{B} = Q$$

Les entrées de la porte ET supérieure sont $\overline{A}$ et B. Par conséquent, sa sortie est donnée par l'expression suivante :

$$\overline{A} \cdot B = Q$$

L'expression booléenne de la sortie de la porte OU est donc :

$$A \cdot \overline{B} + \overline{A} \cdot B = Q$$

Étudions un circuit logique avec de nombreuses entrées et sa représentation par une expression booléenne et une ligne d'un schéma à contacts (voir Figure 5.35).

La sortie de la porte ET supérieure répond à l'expression A · B. La sortie de la porte OU suivante est alors A · B + C. La sortie Q de la dernière porte ET est donnée par l'expression suivante :

$$(A \cdot B + C) \cdot \overline{D} \cdot E \cdot \overline{F} = Q$$

Copyright © 2015 Dunod.

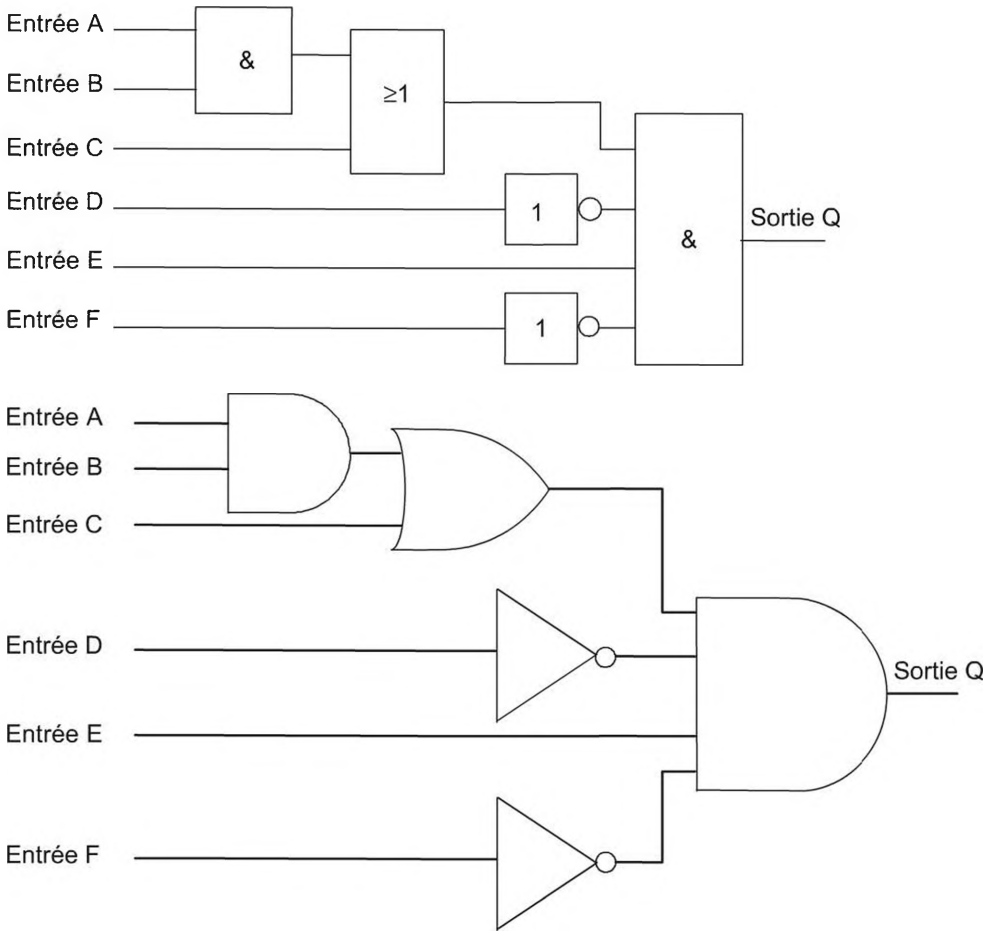


Figure 5.35 – Un circuit logique.

Le schéma à contacts qui correspond à cette expression est donné à la Figure 5.36.

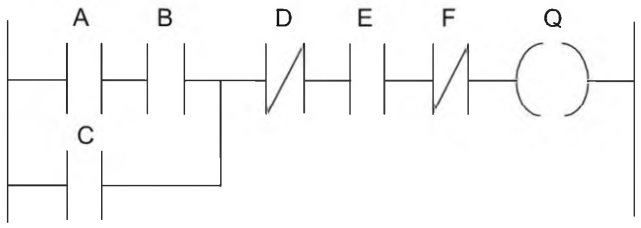


Figure 5.36 – Schéma à contacts pour le circuit logique de la Figure 5.35.

5.7 Exemples de programmes

Les tâches suivantes illustrent la mise en pratique des techniques de programmation décrites dans ce chapitre.

Un témoin lumineux doit être allumé lorsqu'une pompe est en marche et que la pression est satisfaisante, ou bien lorsque son interrupteur de test est fermé. Pour les entrées issues de la pompe et du capteur de pression, nous avons un cas de ET logique, puisque ces deux entrées doivent être actives pour avoir une sortie sur le

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

témoin lumineux. Cependant, nous avons également un OU logique avec l'interrupteur de test, car il doit produire une sortie d'activation du témoin lumineux, que la combinaison ET produise ou non un signal. La Figure 5.37 révèle le diagramme de schémas fonctionnels et le schéma à contacts correspondants. Dans le schéma à contacts, vous remarquerez que nous indiquons à l'API la fin du programme à l'aide d'une instruction END ou RET.

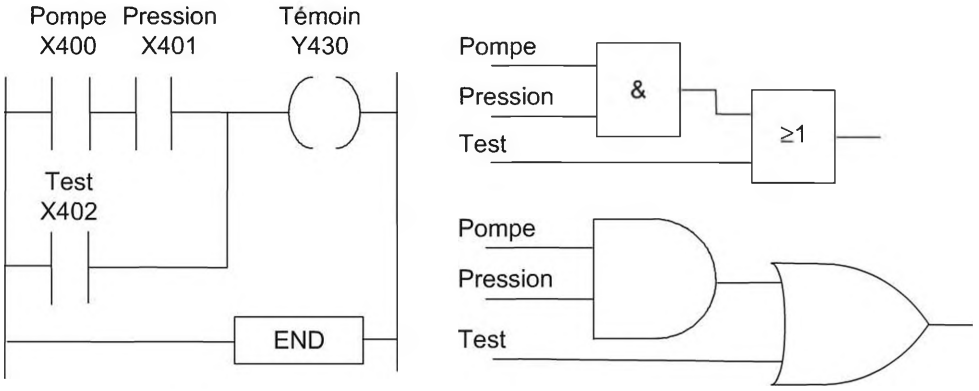


Figure 5.37 – Tâche pour le témoin lumineux.

Prenons un autre exemple, celui d'une vanne qui doit être actionnée pour soulever une charge lorsqu'une pompe est en fonctionnement et lorsque l'interrupteur de levage ou lorsqu'un interrupteur indiquant que la charge n'est pas encore soulevée et qu'elle se trouve en bas de son guide de levage est actionné. Nous avons un OU entre les deux interrupteurs et un ET entre les deux interrupteurs et la pompe. La Figure 5.38 propose un programme qui correspond à cette tâche.

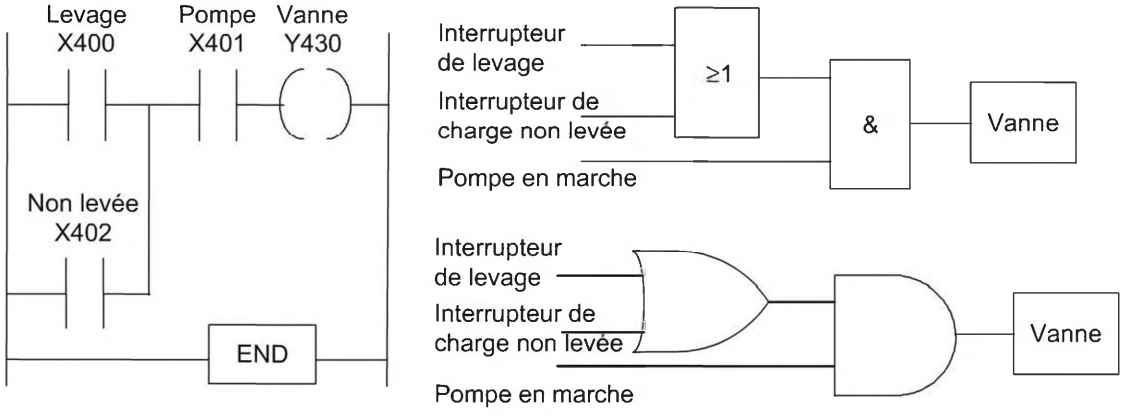


Figure 5.38 – Programme de commande d'une vanne.

Poursuivons nos exemples en examinant un système où la sortie doit être absente lorsque aucun des quatre capteurs ne produit une sortie. Dans le cas contraire, la sortie doit être présente. Pour écrire le programme qui correspond à ce système, nous pouvons utiliser des capteurs dont les contacts sont normalement fermés afin

Copyright © 2015 Dunod.

qu'une sortie soit présente. Lorsqu'une entrée est présente sur le capteur, les contacts s'ouvrent et la sortie disparaît. Nous avons ainsi un ET logique. La Figure 5.39 présente le diagramme de schémas fonctionnels et le schéma à contacts qui correspondent à ce système.

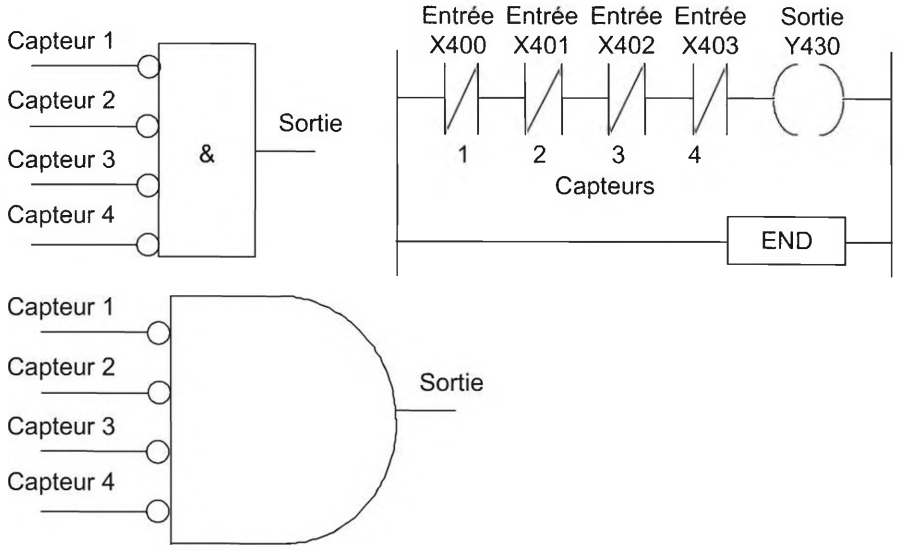


Figure 5.39 – Sortie arrêtée par activation d'un seul des quatre capteurs.

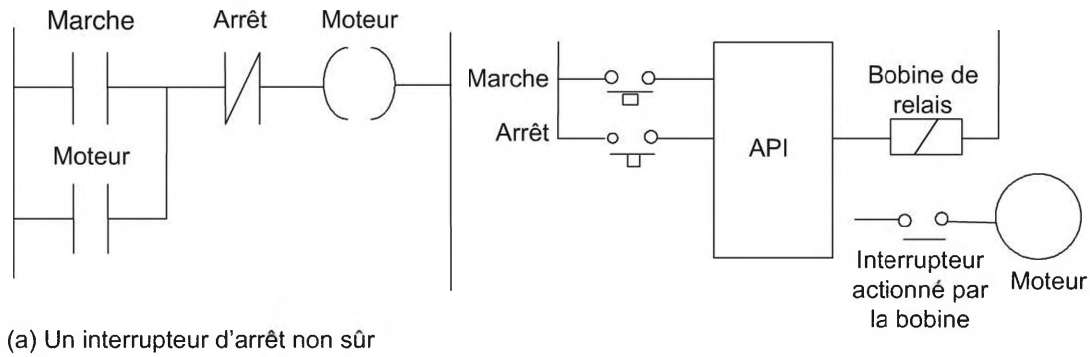
5.7.1 Positionnement des interrupteurs d'arrêt

Dans de nombreuses applications, le positionnement des interrupteurs d'arrêt doit être soigneusement étudié afin de garantir la sécurité du système. Un interrupteur d'arrêt ne donne pas un système fiable s'il est normalement fermé et s'il doit être ouvert pour déclencher l'action d'arrêt. En effet, si l'interrupteur ne fonctionne pas correctement et reste fermé, l'arrêt du système est impossible (voir Figure 5.40a). Dans le schéma à contacts, il est préférable que l'interrupteur d'arrêt soit programmé ouvert (voir Figure 5.33b) et d'utiliser un interrupteur d'arrêt normalement fermé et de l'actionner pour l'ouvrir. Un signal d'entrée ferme alors les contacts au démarrage du programme .

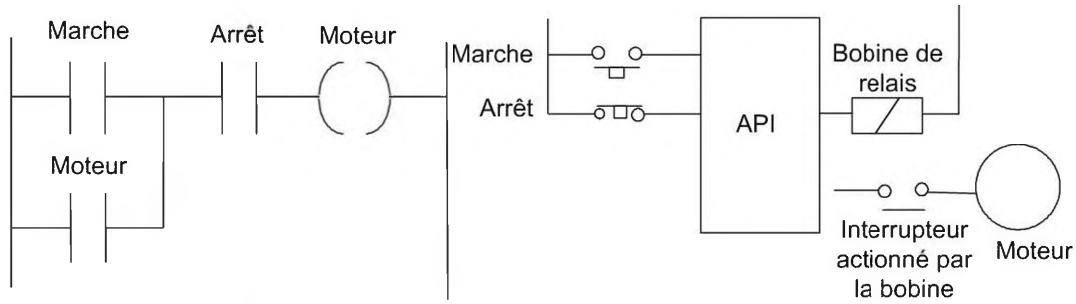
La Figure 5.41 montre où nous pouvons positionner de manière sûre un interrupteur d'arrêt d'urgence. S'il se trouve en entrée de l'API (voir Figure 5.41a), un dysfonctionnement de l'API risque d'empêcher l'arrêt du moteur. En revanche, s'il se trouve en sortie, son activation arrête le moteur et déverrouille l'interrupteur de mise en marche si le montage correspond à celui de la Figure 5.41b. Le moteur ne redémarrera pas tant que le bouton d'arrêt d'urgence sera relâché.

Vous devez toujours envisager la possibilité d'un dysfonctionnement de l'API. Les sorties doivent donc se placer dans un état de « sécurité intégrée » afin que les personnes travaillant dans l'usine ne risquent pas d'être blessées.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

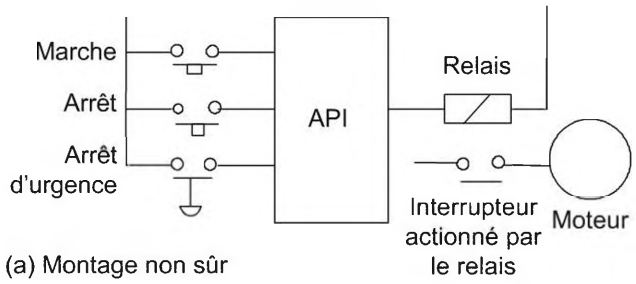


(a) Un interrupteur d'arrêt non sûr

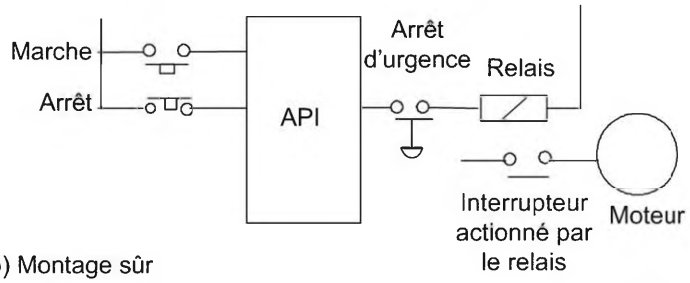


(b) Un interrupteur d'arrêt sûr

Figure 5.40 – Positionnement de l'interrupteur d'arrêt d'un moteur.



(a) Montage non sûr



(b) Montage sûr

Figure 5.41 – Positionnement d'un interrupteur d'arrêt d'urgence.

Copyright © 2015 Dunod.

5.8 En résumé

Le langage à contacts (LD, *Ladder Diagram*) est couramment employé pour programmer les API. Chaque ligne horizontale du schéma définit une opération du processus de commande et doit commencer par une ou plusieurs entrées et se terminer par au moins une sortie. Le programme est analysé ligne par ligne, chacune lue de gauche à droite, et se termine par la ligne END. Le cycle d'analyse du programme reprend indéfiniment. Les entrées et les sorties sont toutes identifiées par leur adresse. Les lignes horizontales du programme peuvent impliquer les opérations logiques ET, OU, NON, NON-ET, NON-OU et OU exclusif sur les entrées. Dans les programmes à verrouillage, suite à l'activation d'une entrée, la sortie verrouille l'entrée afin qu'elle puisse rester présente même lorsque l'entrée disparaît. Pour cela, la sortie dispose d'un ensemble de contacts de type relais, qui sont activés lorsque la sortie se produit et créent un OU avec l'entrée. Par ailleurs, une entrée peut être utilisée pour actionner plusieurs sorties.

Les diagrammes de schémas fonctionnels (FBD, *Function Block Diagrams*) peuvent également être employés pour programmer les API. Ils utilisent un langage graphique pour décrire les signaux et les flux de données au travers de blocs, qui constituent des éléments logiciels réutilisables. Les portes logiques sont des exemples de blocs fonctionnels. L'algèbre booléenne peut être utilisée pour décrire de tels programmes.

5.9 Problèmes

Pour les questions 1 à 19, quatre réponses sont proposées : A, B, C et D. Choisissez la réponse correcte.

1. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). La Figure 5.42 montre une ligne d'un schéma à contacts dans laquelle :
   (i) les contacts d'entrée sont normalement ouverts ;
   (ii) la sortie est présente lorsqu'il existe une entrée sur les contacts.

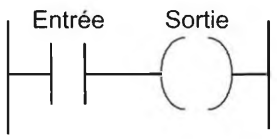


Figure 5.42 – Schéma à contacts pour le problème 1.

Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?

A. (i) V (ii) V
B. (i) V (ii) F
C. (i) F (ii) V
D. (i) F (ii) F

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

2. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). La Figure 5.43 montre une ligne d'un schéma à contacts dans laquelle :
   (i) les contacts d'entrée sont normalement ouverts ;
   (ii) la sortie est présente lorsqu'il existe une entrée sur les contacts.

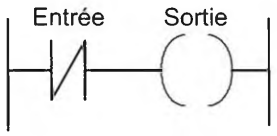


Figure 5.43 – Schéma à contacts pour le problème 2.

   Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
   A. (i) V (ii) V
   B. (i) V (ii) F
   C. (i) F (ii) V
   D. (i) F (ii) F

3. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). La Figure 5.44 montre une ligne d'un schéma à contacts dans laquelle :
   (i) la sortie est présente lorsque seuls les contacts de l'entrée 1 sont activés ;
   (ii) la sortie est présente lorsque seuls les contacts de l'entrée 2 sont activés.

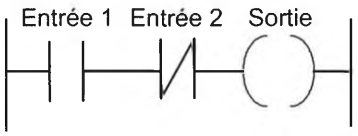


Figure 5.44 – Schéma à contacts pour le problème 3.

   Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
   A. (i) V (ii) V
   B. (i) V (ii) F
   C. (i) F (ii) V
   D. (i) F (ii) F

4. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). La Figure 5.45 montre une ligne d'un schéma à contacts dans laquelle une sortie est présente si :
   (i) les entrées 1 et 2 sont toutes deux activées ;
   (ii) l'une des entrées 1 et 2 est activée.

   Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
   A. (i) V (ii) V
   B. (i) V (ii) F

Copyright © 2015 Dunod.

C. (i) F (ii) V
D. (i) F (ii) F

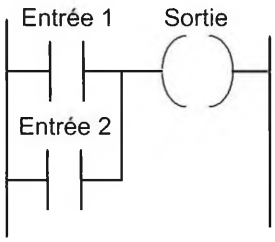


Figure 5.45 – Schéma à contacts pour le problème 4.

5. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). La Figure 5.46 montre une ligne d'un schéma à contacts dans laquelle une sortie est présente si :
   (i) les entrées 1 et 2 sont toutes deux activées ;
   (ii) l'une des entrées 1 et 2 est activée.

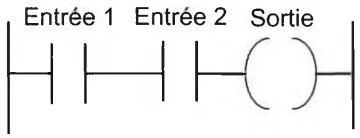


Figure 5.46 – Schéma à contacts pour le problème 5.

Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
A. (i) V (ii) V
B. (i) V (ii) F
C. (i) F (ii) V
D. (i) F (ii) F

6. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). La Figure 5.46 montre une ligne d'un schéma à contacts dans laquelle une sortie est présente si :
   (i) l'entrée 1 est momentanément activée avant de revenir dans son état normalement ouvert ;
   (ii) l'entrée 2 est activée.

   Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
   A. (i) V (ii) V
   B. (i) V (ii) F
   C. (i) F (ii) V
   D. (i) F (ii) F

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

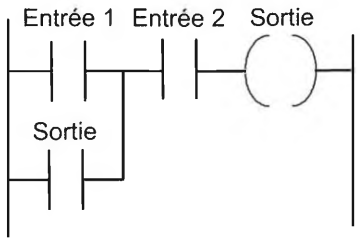


Figure 5.47 – Schéma à contacts pour le problème 6.

Les problèmes 7 à 10 font référence aux portes logiques suivantes :

A. ET
B. OU
C. NON-OU
D. NON-ET

7. Quelle porte logique est représentée par une ligne d'un schéma à contacts dans laquelle deux ensembles de contacts normalement ouverts sont en parallèle ?
8. Quelle porte logique est représentée par une ligne d'un schéma à contacts dans laquelle deux ensembles de contacts normalement fermés sont en parallèle ?
9. Quelle porte logique est représentée par une ligne d'un schéma à contacts dans laquelle deux ensembles de contacts normalement fermés sont en série ?
10. Quelle porte logique est représentée par une ligne d'un schéma à contacts dans laquelle deux ensembles de contacts normalement fermés sont en série ?

Les problèmes 11 à 14 concernent des expressions booléennes qui impliquent des entrées A et B.

A. Les entrées A et B, normalement inactives, sont en série.
B. Les entrées A et B, normalement inactives, sont en parallèle.
C. L'entrée A, normalement inactive, et l'entrée B, normalement active, sont en série.
D. Les entrées A et B, normalement actives, sont en parallèle.

11. Quel agencement des entrées est décrit par la relation booléenne $A \cdot B$?
12. Quel agencement des entrées est décrit par la relation booléenne $A + B$?
13. Quel agencement des entrées est décrit par la relation booléenne $\overline{A} + \overline{B}$?
14. Quel agencement des entrées est décrit par la relation booléenne $A \cdot \overline{B}$?
15. À quelle relation booléenne correspond l'agencement des entrées de la Figure 5.48 :

A. $A \cdot B \cdot C$
B. $(A + C) \cdot B$
C. $(A + B) \cdot C$
D. $A \cdot C + B$

Copyright © 2015 Dunod.

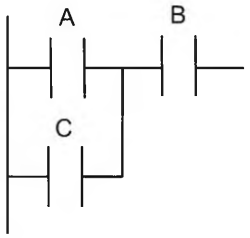


Figure 5.48 – Schéma pour le problème 15.

16. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Pour le diagramme de schémas fonctionnels de la Figure 5.49, une sortie est présente :
(i) lorsque A est à 1 ;
(ii) lorsque B est à 1.

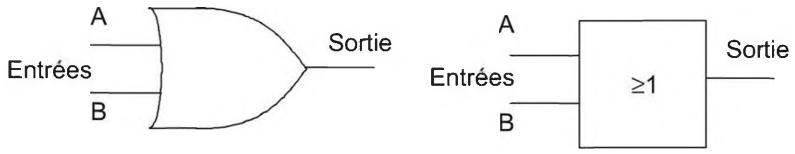


Figure 5.49 – Schéma pour le problème 16.

Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
A. (i) V (ii) V
B. (i) V (ii) F
C. (i) F (ii) V
D. (i) F (ii) F

17. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Pour le diagramme de schémas fonctionnels de la Figure 5.50, une sortie est présente :
(i) lorsque A est à 1 ;
(ii) lorsque B est à 1.

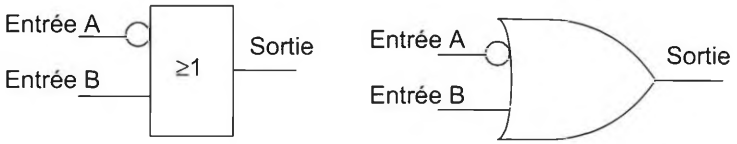


Figure 5.50 – Schéma pour le problème 17.

Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
A. (i) V (ii) V
B. (i) V (ii) F

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

C. (i) F (ii) V
D. (i) F (ii) F

18. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Pour le diagramme de schémas fonctionnels de la Figure 5.51, une sortie est présente :
(i) lorsque A est à 1, B est à 0 et C est à 0 ;
(ii) lorsque A est à 0, B est à 1 et C est à 1.

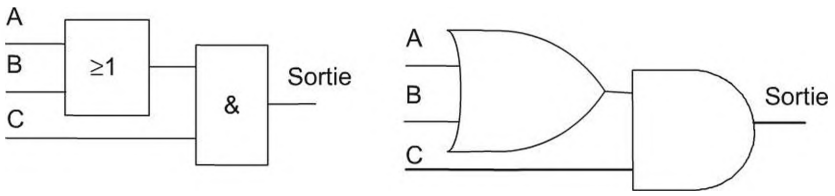


Figure 5.51 – Schéma pour le problème 18.

Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
A. (i) V (ii) V
B. (i) V (ii) F
C. (i) F (ii) V
D. (i) F (ii) F

19. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Pour le diagramme de schémas fonctionnels de la Figure 5.52, avec A étant une condition d'entrée stable et B, une entrée transitoire, une sortie est présente :
(i) lorsque A est à 1 et B est à 0 ;
(ii) lorsque A est à 0 et B est à 1.

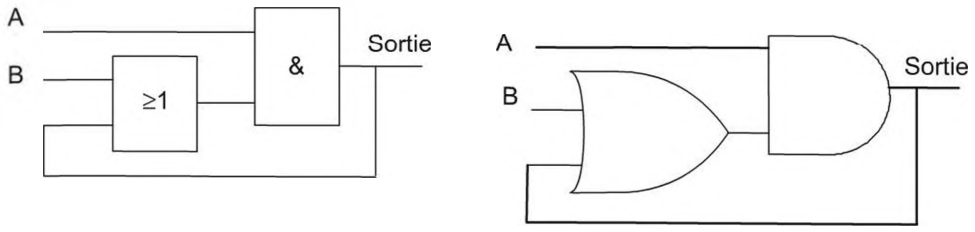


Figure 5.52 – Schéma pour le problème 19.

Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
A. (i) V (ii) V
B. (i) V (ii) F
C. (i) F (ii) V
D. (i) F (ii) F

Copyright © 2015 Dunod.

20. La Figure 5.53 montre un schéma à contacts. Quel est le diagramme de schémas fonctionnels équivalent ?

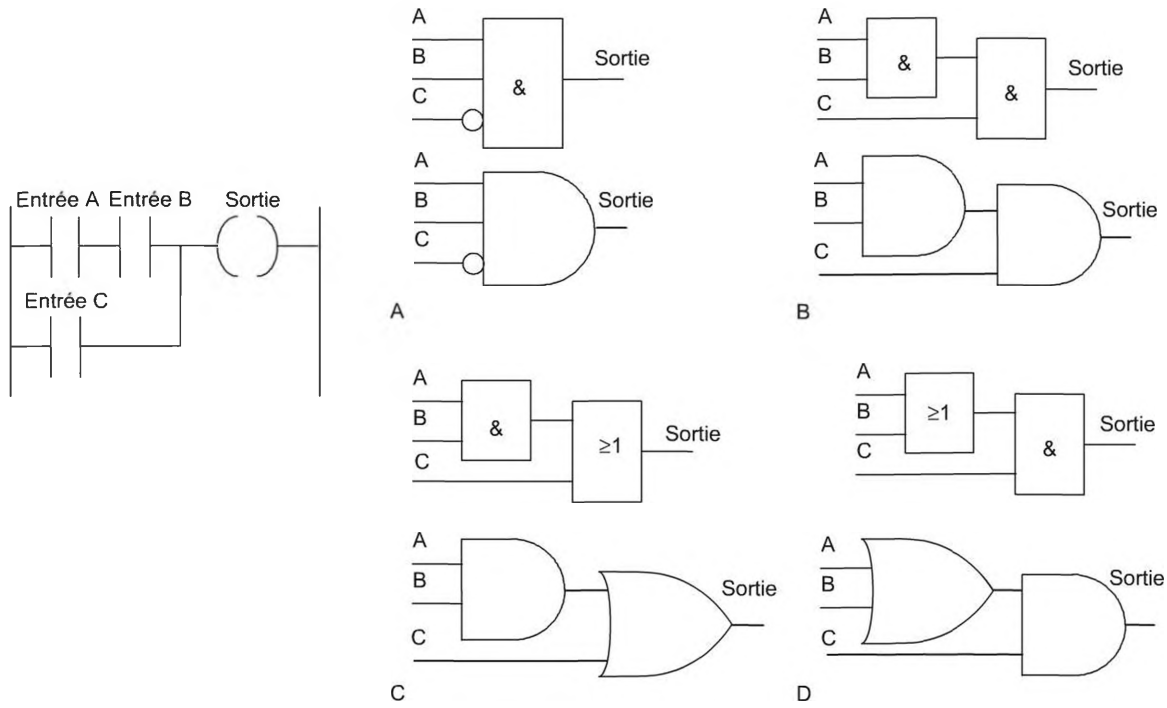


Figure 5.53 – Schéma pour le problème 20.

21. La Figure 5.54 montre un diagramme de schémas fonctionnels. Quel est le schéma à contacts équivalent ?
22. La Figure 5.55 montre un schéma à contacts. Quel chronogramme montre les signaux d'entrée et de sortie obtenus ?
23. La Figure 5.56 montre un schéma à contacts. Quel chronogramme montre les signaux d'entrée et de sortie obtenus ?
24. La Figure 5.57 montre un schéma à contacts. Quel chronogramme montre les signaux d'entrée et de sortie obtenus ?
25. Dessinez un schéma à contacts pour représenter les situations suivantes :
    (a) Deux interrupteurs normalement ouverts qui doivent être fermés pour démarrer un moteur.
    (b) L'un des deux interrupteurs normalement ouverts doit être fermé pour exciter une bobine et activer un actionneur.
    (c) Un moteur est mis en marche en appuyant sur bouton-poussoir de démarrage avec ressort de retour, et il reste en marche jusqu'à l'appui sur un bouton-poussoir d'arrêt avec ressort de retour.
    (d) Un témoin lumineux s'allume en cas d'entrée sur un capteur A ou un capteur B.
    (e) Un témoin lumineux s'allume en cas d'absence d'entrée sur un capteur.
    (f) Une électrovanne est activée si un capteur A produit une entrée.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

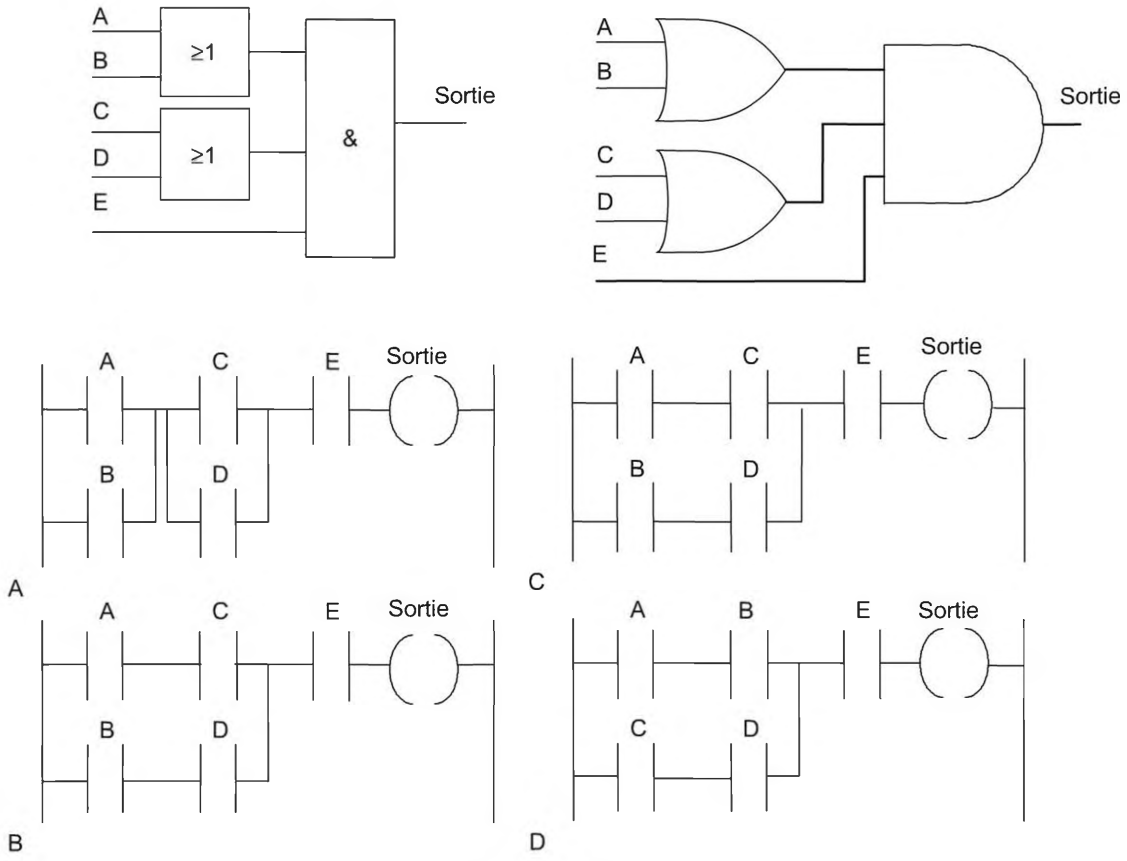


Figure 5.54 – Schéma pour le problème 21.

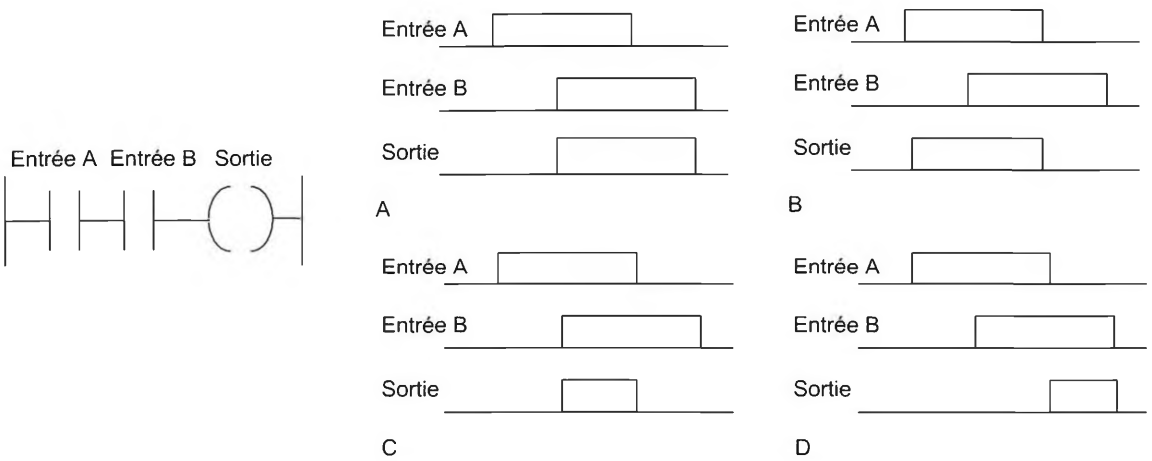


Figure 5.55 – Schéma pour le problème 22.

(g) Pour des questions de sécurité, les machines sont souvent configurées de manière à pouvoir être manipulées par un opérateur uniquement si deux interrupteurs sont pressés simultanément, l'un de la main droite, l'autre de la main gauche. Cela permet d'être certain que les deux mains de l'opérateur sont placées sur les interrupteurs, non dans la machine, lorsqu'elle est mise en marche. Dessinez le schéma à contacts qui correspond à ce type de contrainte.

Copyright © 2015 Dunod.

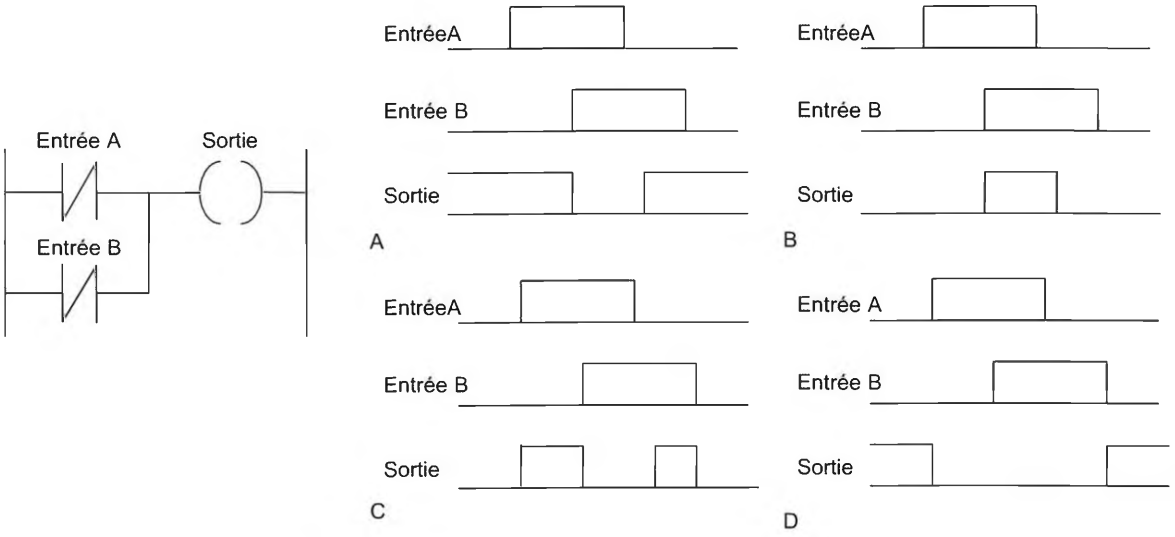


Figure 5.56 – Schéma pour le problème 23.

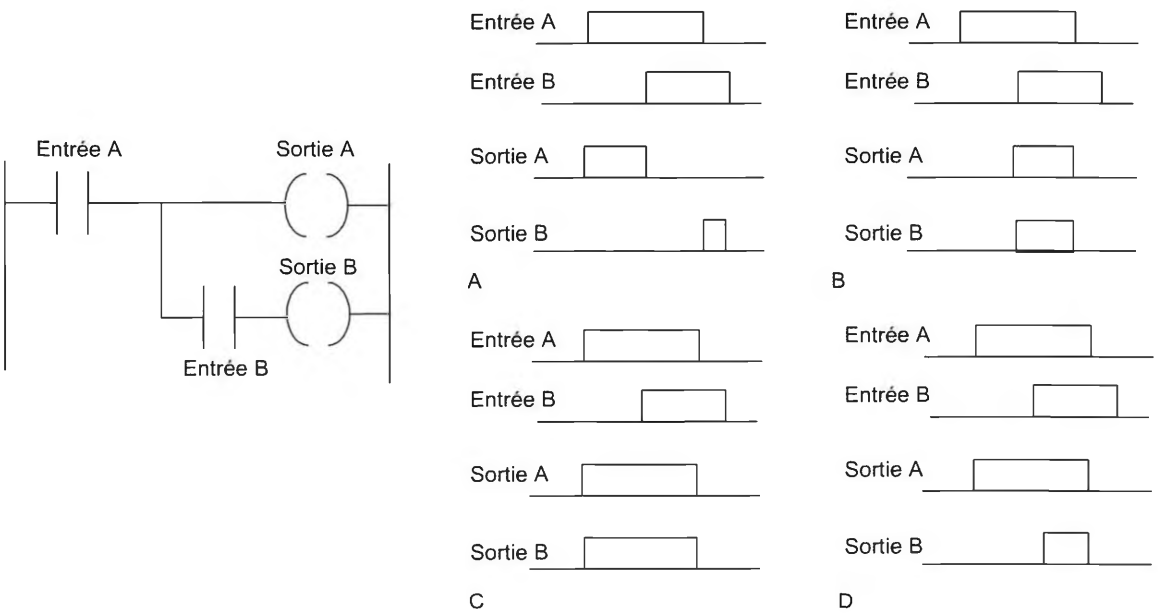


Figure 5.57 – Schéma pour le problème 24.

26. Dessinez un diagramme de schémas fonctionnels pour représenter les situations suivantes :
    (a) Un moteur est démarré lorsque l'interrupteur A ou l'interrupteur B est activé.
    (b) Un moteur est démarré lorsque deux interrupteurs normalement ouverts sont activés et le restent, même si le premier des deux interrupteurs est désactivé tandis que l'autre reste activé.
    (c) Une pompe est mise en marche si un interrupteur de démarrage est activé ou si un interrupteur de test est actionné.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

27. Tracez le schéma à contacts qui correspond aux expressions booléennes suivantes :
    (a) $Q = A + \overline{B}$
    (b) $Q = \overline{A} \cdot B \cdot C + D$
    (c) $Q = A \cdot B + C \cdot D$
28. Écrivez les équations booléennes qui représentent les cas suivants :
    (a) La sortie Q est présente lorsque l'une des entrées A et B est présente.
    (b) La sortie Q est présente lorsque les entrées A et B sont présentes et lorsque l'entrée C est absente.
    (c) La sortie Q est présente lorsque l'entrée A est présente et lorsque l'entrée B est absente.

5.10 Recherches complémentaires

1. Recherchez les spécifications d'un API et relevez les informations qui concernent (a) le temps d'analyse et (b) la programmation.

Copyright © 2015 Dunod.

6 • PROGRAMMATION IL, SFC ET ST

Ce chapitre poursuit sur la lancée du précédent en décrivant d'autres langages de programmation définis par la CEI 61131-3, c'est-à-dire les listes d'instructions (IL, *Instruction Lists*), les graphes de fonction séquentielle (SFC, *Sequential Function Charts*) et le texte structuré (ST, *Structured Text*).

6.1 Listes d'instructions

Les listes d'instructions constituent une méthode de programmation que l'on peut comparer à la saisie d'un schéma à contacts sous forme d'un texte. Un programme écrit selon cette méthode est constitué d'une suite d'instructions, chacune placée sur une nouvelle ligne. Chaque instruction est constituée d'un opérateur suivi d'un ou plusieurs opérandes, c'est-à-dire les objets de l'opérateur. Ainsi, nous pouvons avoir la ligne suivante :

```
LD A
```

Elle indique le chargement de l'opérande A ; LD est l'opérateur utilisé pour effectuer un chargement. Comparé au langage à contacts, un opérateur peut être vu comme un élément d'une ligne et LD équivaut à commencer une ligne par des contacts ouverts pour l'entrée A. Voici une autre instruction :

```
OUT Q
```

Elle indique qu'une sortie doit être produite sur Q.

Des codes mnémoniques sont utilisés pour les opérateurs, chaque code correspondant à un opérateur/élément d'un schéma à contacts. Ces codes diffèrent d'un fabricant à l'autre, mais la norme CEI 61131-3 a été établie et est à présent largement adoptée. Le Tableau 6.1 recense quelques codes utilisés par les fabricants et les instructions normalisées que nous rencontrerons dans ce chapitre (les chapitres ultérieurs emploieront d'autres codes pour d'autres fonctions).

Les listes d'instructions sont un langage textuel de bas niveau simple à implémenter et utilisé par nombre de fabricants d'API, principalement pour les modèles de petite et moyenne taille. Ce langage est particulièrement adapté aux programmes simples. Certains fabricants ne prennent pas en charge les listes d'instructions, mais proposent le texte structuré, un langage de plus haut niveau.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

Tableau 6.1 Codes mnémoniques des instructions

CEI 61131-3	Mitsubishi	OMRON	Siemens	Opération	Langage à contacts
LD	LD	LD	A	Charger l'opérande dans un registre de résultat.	Commencer une ligne avec des contacts ouverts.
LDN	LDI	LD NOT	AN	Charger l'opérande inversé dans un registre de résultat.	Commencer une ligne avec des contacts fermés.
AND	AND	AND	A	ET booléen.	Éléments en série avec des contacts ouverts.
ANDN	ANI	AND NOT	AN	ET booléen avec un opérande inversé.	Éléments en série avec des contacts fermés.
OR	OR	OR	O	OU booléen.	Éléments en parallèle avec des contacts ouverts.
ORN	ORI	OR NOT	ON	OU booléen avec un opérande inversé.	Éléments en parallèle avec des contacts fermés.
ST	OUT	OUT	=	Stocker un registre de résultats dans un opérande.	Une sortie

Voici une illustration de l'utilisation des opérateurs définis par la CEI 61131-3 :

```
LD      A       (*Charger A*)
AND     B       (*ET B*)
ST      Q       (*Stocker le résultat dans Q, c-à-d sortie sur Q*)
```

Sur la première ligne du programme, LD représente l'opérateur, A représente l'opérande et les mots placés en fin de lignes, entre parenthèses et précédés et suivis par un caractère *, sont des commentaires qui expliquent l'opération et ne font pas partie des instructions du programme de l'API. LD A est ainsi une instruction qui charge A dans le registre mémoire. Il peut ensuite être rappelé pour d'autres opérations. La ligne suivante du programme effectue une opération booléenne ET entre A et B. La dernière ligne place le résultat dans Q, c'est-à-dire sur la sortie Q.

Il est possible d'utiliser des libellés pour identifier différents points d'entrée dans un programme. Nous le verrons plus loin, ils sont utiles pour sauter à des endroits d'un programme. Un libellé se place avant l'instruction et en est séparé par des deux-points :

```
POMPE_OK:   LD      C       (*Charger C*)
```

Copyright © 2015 Dunod.

Une instruction dans le programme peut sauter à la ligne POMPE_OK lorsqu'une condition particulière est rencontrée.

Pour les opérateurs de la CEI 6113-1, l'ajout d'un N à la fin de leur mnémonique indique une valeur inversée. Par exemple :

```
LD      A       (*Charger A*)
ANDN    B       (*ET NON B*)
```

Ainsi, l'opérateur ANDN inverse la valeur des contacts et réalise un ET avec le résultat.

6.1.1 Schémas à contacts et listes d'instructions

Pour correspondre au début d'une ligne horizontale d'un schéma à contacts, nous devons, dans une liste d'instructions, employer un code « débuter une ligne ». Il peut s'agir de LD, ou peut-être A ou L, pour indiquer que la ligne commence par des contacts ouverts, ou de LDI, ou peut-être LDN, LD NOT, AN ou LN, pour indiquer qu'elle commence par des contacts fermés. Toutes les lignes doivent se terminer par une sortie ou un code de stockage du résultat. Il peut s'agir de OUT, ou peut-être de = ou ST. Nous allons voir comment des lignes individuelles d'un schéma à contacts sont saisies à l'aide des mnémoniques de Mitsubishi pour la porte ET illustrée à la Figure 6.1a.

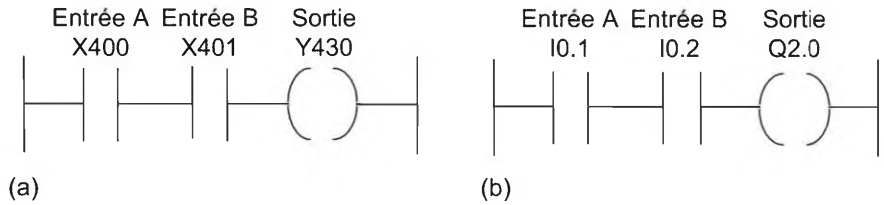


Figure 6.1 – Porte ET : (a) Mitsubishi et (b) Siemens.

La première instruction utilise l'opérateur LD car la ligne commence par des contacts ouverts. L'adresse de l'entrée étant X400 (voir Figure 6.1a), l'instruction est LD X400. Ensuite vient une autre entrée à contacts ouverts. La ligne suivante du programme est donc l'instruction AND avec l'adresse de l'entrée, c'est-à-dire AND X401. Puisque la ligne se termine par une sortie, l'instruction OUT est utilisée avec l'adresse de cette sortie, ce qui donne OUT Y430. La ligne du schéma à contacts correspond donc à la liste d'instructions suivante :

```
LD      X400
AND     X401
OUT     Y430
```

En utilisant la notation de Siemens (voir Figure 6.1b), nous avons :

```
A       I0.1
A       I0.2
=       Q2.0
```

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

Prenons un autre exemple, celui d'une porte OU. La Figure 6.2a présente cette porte avec la notation de Mitsubishi.

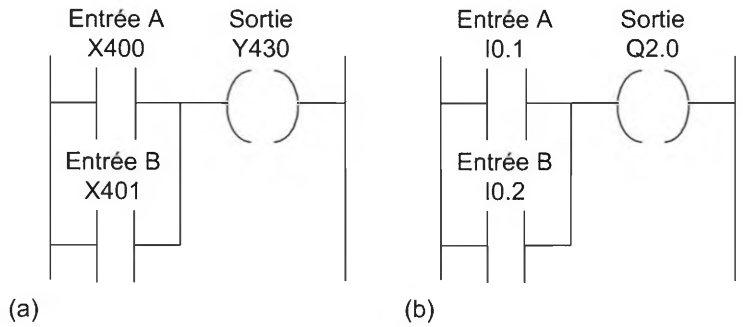


Figure 6.2 – Porte OU : (a) Mitsubishi et (b) Siemens.

Puisque la ligne commence par des contacts ouverts, l'instruction correspondante est LD X400. Ensuite vient l'ensemble de contacts X401 placé en parallèle, qui représente un OU. L'instruction suivante est donc OR X401. La dernière étape est représentée par la sortie, d'où OUT Y430. Voici la liste d'instructions :

```
LD      X400
OR      X401
OUT     Y430
```

La Figure 6.2b montre la porte OU avec la notation de Siemens. La liste d'instructions devient alors la suivante :

```
A       I0.1
O       I0.2
=       Q2.0
```

La Figure 6.3a présente le schéma à contacts d'une porte NON-OU dans la notation de Mitsubishi.

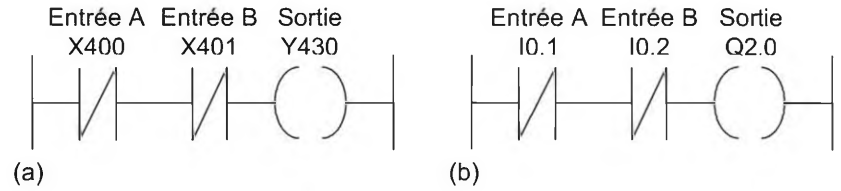


Figure 6.3 – Porte NON-OU : (a) Mitsubishi et (b) Siemens.

La ligne commence par des contacts normalement fermés, ce qui correspond à l'instruction LDI. Lorsqu'un I est ajouté à une instruction Mitsubishi, il indique une opération inverse. Viennent ensuite des contacts normalement fermés en série. L'instruction est donc ANI, le I étant à nouveau utilisé pour inverser l'instruction AND. I représente également l'instruction d'une porte NON. Voici la liste des instructions pour la porte NON-OU du schéma à contacts :

```
LDI     X400
ANI     X401
OUT     Y430
```

Copyright © 2015 Dunod.

La Figure 6.3b présente la même porte, mais selon la notation de Siemens. Vous remarquerez l'utilisation de N sur les instructions pour représenter la négation. Voici la liste d'instructions correspondante :

```
LN      I0.1
AN      I0.2
=       Q2.0
```

La Figure 6.4a illustre une porte NON-ET avec la notation de Mitsubishi.

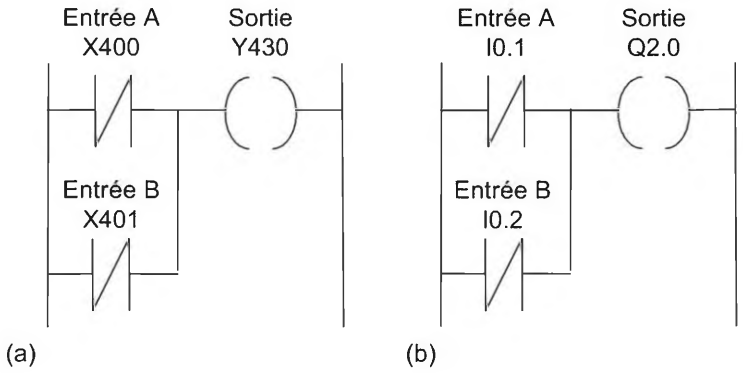


Figure 6.4 – Porte NON-ET : (a) Mitsubishi et (b) Siemens.

Elle commence par les contacts normalement fermés X400 et donc l'instruction LDI X400. L'instruction suivante, ORI X401, correspond aux contacts normalement fermés placés en parallèle. La dernière étape est la sortie, d'où OUT Y430. Voici la liste d'instructions complète :

```
LDI     X400
ORI     X401
OUT     Y430
```

La Figure 6.4b illustre la porte NON-ET avec la notation de Siemens, dont voici la liste d'instructions correspondante :

```
AN      I0.1
ON      I0.2
=       Q2.0
```

6.1.2 Codage des embranchements

La porte OU exclusif de la Figure 6.5 comprend deux branches en parallèle avec un ET dans chacune d'elles.

La Figure 6.5a montre la version Mitsubishi. Dans ce genre de situation, Mitsubishi utilise une instruction ORB pour indiquer un « OU entre des branches en parallèle ». La première instruction concerne le couple de contacts normalement ouverts X400. L'instruction suivante correspond aux contacts normalement fermés placés en série X401, d'où ANI X401. Après avoir lu les deux premières instructions, qui forment une première ligne, la troisième débute une nouvelle ligne. Il s'agit d'une nouvelle ligne car elle commence par LDI ; les nouvelles lignes commencent toutes par LD ou LDI. Toutefois, la première ligne n'a pas été

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

terminée par une sortie. L'API détermine donc que la deuxième ligne est en parallèle avec la première et lit les éléments indiqués jusqu'à atteindre l'instruction ORB. Le mnémonique ORB (OU entre des branches/blocs) indique à l'API qu'il doit effectuer un OU entre le résultat des deux premières instructions et le résultat de la nouvelle ligne, qui contient les troisième et quatrième instructions. La liste se termine par la sortie OUT Y430. Voici l'intégralité de la liste d'instructions correspondante :

```
LD      X400
ANI     X401
LDI     X400
AND     X401
ORB
OUT     Y430
```

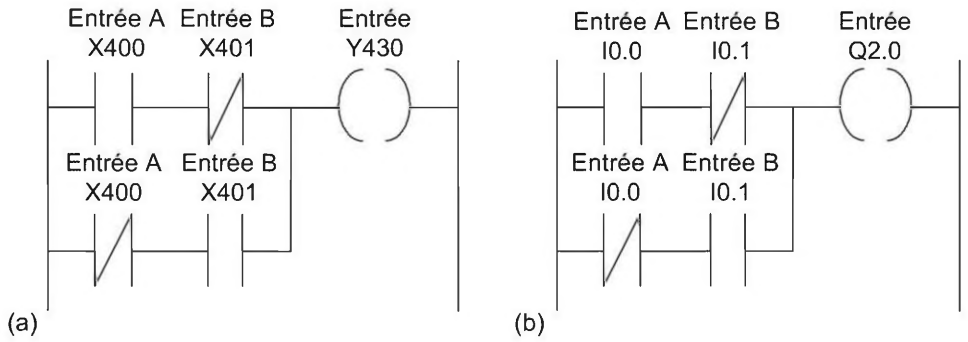


Figure 6.5 – Porte OU exclusif : (a) Mitsubishi et (b) Siemens.

La Figure 6.5b présente la version Siemens de la porte OU exclusif. Les parenthèses indiquent que certaines instructions forment un bloc. Elles jouent le même rôle que les parenthèses dans une équation mathématique. Par exemple, (2 + 3) / 4 signifie que 2 et 3 doivent être additionnés avant que le résultat ne soit divisé par 4. Ainsi, dans la liste d'instructions de Siemens, nous avons l'instruction A(à l'étape 0. La parenthèse est fermée à l'étape 3. Cela signifie que l'instruction A de l'étape 0 est appliquée uniquement après l'exécution des instructions des étapes 1 et 2.

Étape	Instruction	
0	A(	
1	A	I0.0
2	AN	I0.1
3	)	
4	O(	

Copyright © 2015 Dunod.

Étape	Instruction	
5	AN	I0.0
6	A	I0.1
7	)	
8	=	Q2.0

Dans la norme CEI 61131-3, les parenthèses servent le même objectif que dans l'exemple précédent pour la version Siemens, c'est-à-dire comme dans les expressions arithmétiques normales. Cela permet de différer l'exécution des instructions entre les parenthèses jusqu'à ce que la parenthèse soit fermée. Ainsi, la liste d'instructions CEI suivante :

```
LD X
ADD( B
MUL( C
ADD D
)
)
```

correspond à l'expression X + (B × (C + D)).

La Figure 6.6 présente un circuit effectuant un ET entre des blocs à deux branches. La version Mitsubishi est donnée à la Figure 6.6a. L'instruction utilisée ici est ANB. Voici la liste d'instructions correspondante :

Étape	Instruction	
0	LD	X400
1	OR	X402
2	LD	X401
3	OR	X403
4	ANB	
5	OUT	Y430

La Figure 6.6b présente le même circuit, mais dans la notation de Siemens. Le programme est écrit sous forme d'une liste d'instructions avec des parenthèses. L'instruction A de l'étape 0 est appliquée au résultat des étapes 1 et 2. L'instruction A de l'étape 4 est appliquée au résultat des étapes 5 et 6. Voici la liste d'instructions correspondante :

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

Étape	Instruction	
0	A(	
1	A	I0.0
2	O	I0.2
3	)	
4	A(	
5	A	I0.1
6	O	I0.3
7	)	
8	=	Q2.0

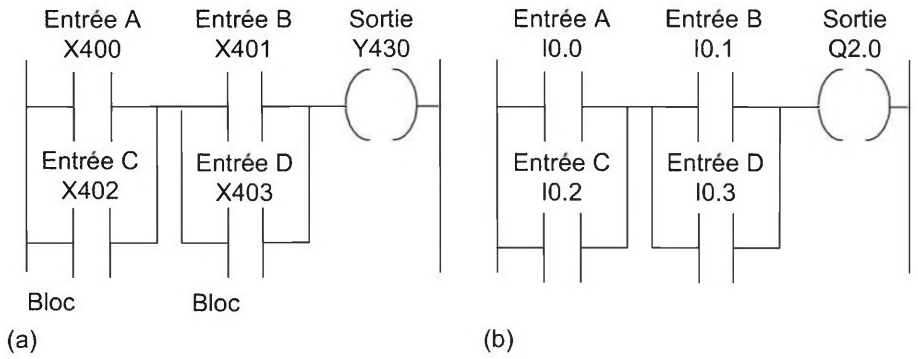


Figure 6.6 – ET entre deux branches : (a) Mitsubishi et (b) Siemens.

6.1.3 Plusieurs lignes horizontales

La Figure 6.7a présente un schéma à contacts, selon la notation de Mitsubishi, avec deux lignes horizontales. Lors du développement de la liste d'instructions, nous écrivons simplement les instructions qui correspondent à chaque ligne, tour à tour. Les instructions LD et LDI indiquent à l'API le début d'une nouvelle ligne. La liste des instructions est donc la suivante :

```
LD      X400
OUT     Y430
LDI     X400
OUT     Y431
```

Dans un tel système, lorsque X400 n'est pas activée, il existe une sortie pour Y431, mais non pour Y430. En revanche, lorsque X400 est activée, il existe une sortie pour Y430, mais non pour Y431.

Copyright © 2015 Dunod.

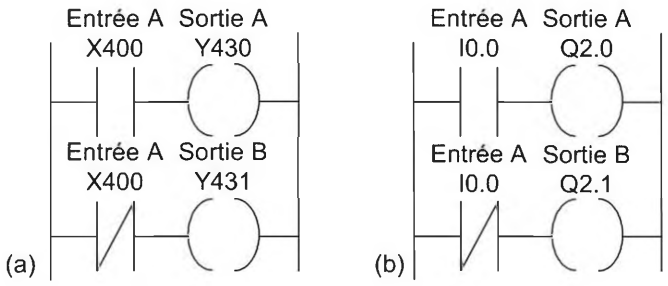


Figure 6.7 – Circuit à bascule : (a) Mitsubishi et (b) Siemens.

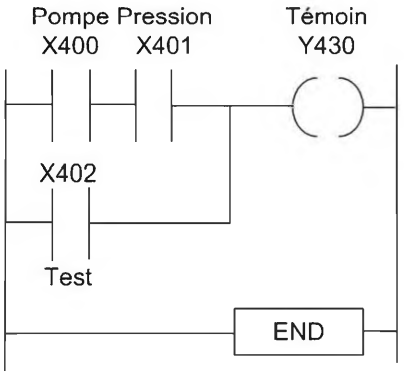
La Figure 6.7b présente le même circuit dans la notation de Siemens. L'instruction = indique la fin d'une ligne. Les instructions A et AN n'indiquent pas nécessairement le début d'une ligne horizontale, car la même instruction est utilisée pour les opérations ET et NON-ET. Voici la liste d'instructions obtenue :

```
A       I0.0
=       Q2.0
AN      I0.0
=       Q2.1
```

6.1.4 Exemples de programmes

Les tâches suivantes illustrent la mise en pratique des techniques de programmation données dans cette section et correspondent aux exemples de schémas à contacts et de diagrammes de schémas fonctionnels de la Section 5.7 (consultez cette section pour une explication des schémas à contacts ; nous présentons ici les listes d'instructions des programmes).

Un témoin lumineux doit être allumé lorsqu'une pompe est en marche et que la pression est satisfaisante, ou bien lorsque l'interrupteur de test est fermé. La Figure 6.8 montre le schéma à contacts et la liste d'instructions correspondante.



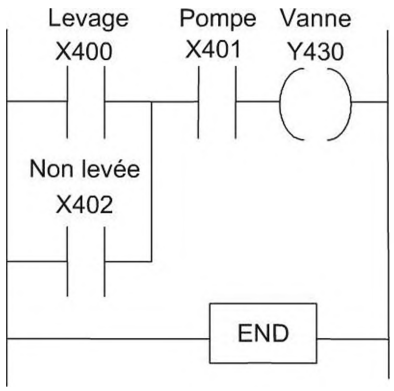

```
LD      X400
AND     X401
LD      X402
ORB
OUT     Y430
END
```

Figure 6.8 – Lampe témoin.

Une vanne doit être actionnée pour soulever une charge lorsqu'une pompe est en fonctionnement et lorsque l'interrupteur de levage ou lorsqu'un interrupteur indiquant que la charge n'est pas encore soulevée et qu'elle se trouve en bas de son

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

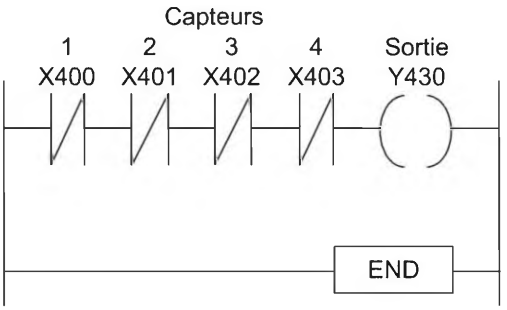
guide de levage est actionné. La Figure 6.9 présente le schéma à contacts et la liste d'instructions correspondante.




```
LD    X400
OR    X402
AND   X401
OUT   Y430
END
```

Figure 6.9 – Programme d'activation de la vanne.

Dans un système, la sortie doit être absente lorsque aucun des quatre capteurs ne produit une sortie. Dans le cas contraire, la sortie doit être présente. La Figure 6.10 donne le schéma à contacts et la liste d'instructions correspondante.




```
LDI   X400
ANI   X401
ANI   X402
ANI   X403
OUT   Y430
END
```

Figure 6.10 – Sortie arrêtée par activation d'un seul des quatre capteurs.

6.2 Graphes de fonction séquentielle

Si vous souhaitez décrire le fonctionnement rouge-vert d'un feu de circulation, une solution consiste à le représenter sous forme d'une suite de fonctions ou d'états, comme l'état du feu rouge et celui du feu vert, ainsi que des entrées et des sorties de chaque état. La Figure 6.11 illustre cela. L'état 0 possède une entrée qui est déclenchée après que le feu vert est resté allumé pendant une minute. Sa sortie est l'allumage du feu rouge, la condition de passage à partir du feu rouge étant une durée d'une minute. L'état 1 possède une entrée qui est déclenchée après que le feu rouge est resté allumé pendant une minute. Sa sortie est l'allumage du feu vert, la condition de passage à partir du feu vert étant une durée d'une minute. Lorsque le feu vert est resté allumé pendant une minute, il y a un retour vers l'état 0.

Le terme *graphe de fonction séquentielle* (SFC, *Sequential Function Chart*) correspond à une représentation graphique du fonctionnement d'un système afin de révéler

Copyright © 2015 Dunod.

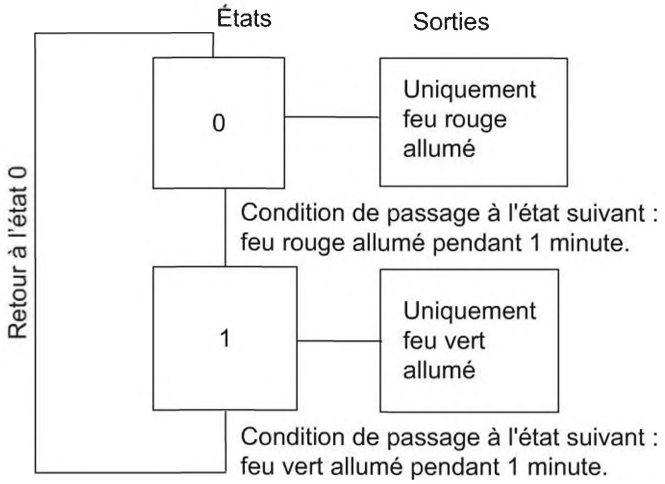


Figure 6.11 – Enchaînement pour un feu de circulation.

l'enchaînement des événements qui conduisent à ce fonctionnement. La Figure 6.11 illustre ce type de fonctionnement. Un SFC est composé des éléments suivants :

1. Le fonctionnement est décrit par une suite d'étapes ou d'états distincts reliés séquentiellement et représentés par des rectangles. Chaque rectangle correspond à un état particulier du système et où une action se produit. L'étape initiale d'un programme est représentée différemment des autres étapes, avec des doubles lignes. La Figure 6.12 montre une étape de départ et les étapes suivantes.

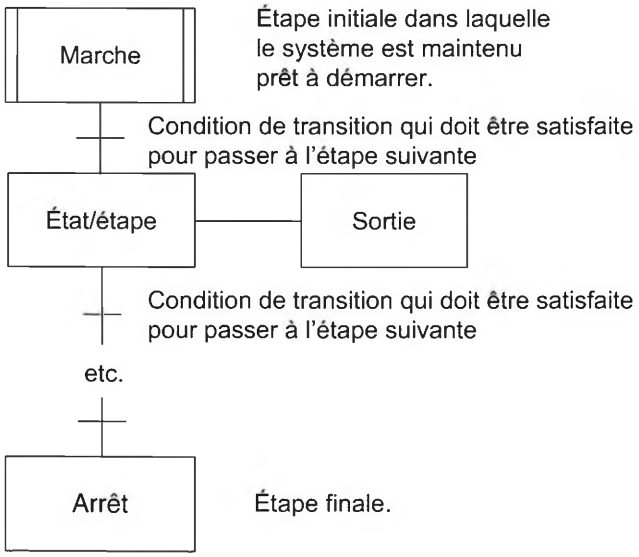


Figure 6.12 – Un état et sa transition.

2. Chaque ligne de connexion entre des états comprend une barre horizontale, qui représente la condition de transition qui doit être satisfaite pour que le système puisse passer d'un état au suivant. Deux états ne peuvent jamais être reliés directement ; ils doivent toujours être séparés par une transition. Deux

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

transitions ne peuvent jamais se trouver l'une derrière l'autre ; elles doivent toujours être séparées par une étape.

3. Le programme vérifie les conditions de passage à l'état suivant et, lorsqu'elles sont satisfaites, l'étape ou l'état suivant est activé.
4. Le processus se poursuit d'un état au suivant, jusqu'à ce que le cycle complet de la machine soit effectué.
5. Les sorties/actions d'un état/étape sont indiquées dans une boîte connectée horizontalement à l'état et sont réalisées lorsque celui-ci est atteint. Nous pouvons donc rencontrer des blocs comme Attendre une minute, Ouvrir la vanne 1 ou Fermer la vanne 2.

La Figure 6.13 présente une partie d'un graphe de fonction séquentielle, ainsi que le schéma à contacts équivalent. Le programme démarre lorsque Entrée 1 est validée, ce qui donne la sortie de l'étape 1. Lorsque cette sortie est terminée, la transition suivante a lieu et l'étape 2 se produit, avec la sortie 2 résultante. Lorsqu'elle est terminée, la condition de passage est satisfaite pour aller à l'étape Fin.

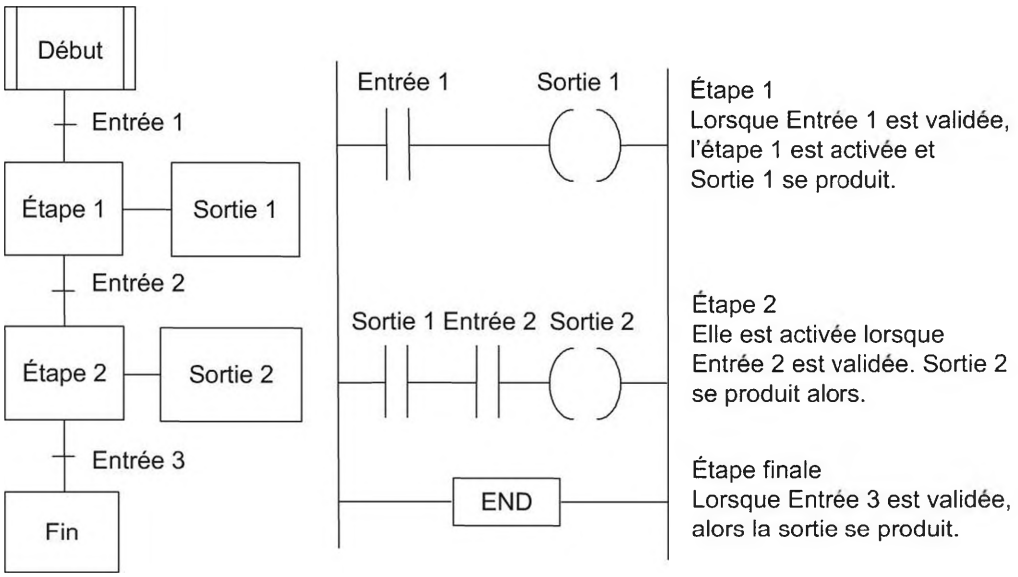


Figure 6.13 – Graphe de fonction séquentielle et schéma à contacts équivalent.

Pour illustrer les principes du SFC, examinons la partie du cycle de lavage d'un lave-linge domestique qui correspond au remplissage du tambour, à la mise sous tension de la résistance de chauffage lorsque le tambour est plein et au maintien du chauffage tant que la température réglée n'est pas atteinte. Ensuite, le tambour est mis en rotation pendant une durée déterminée. Nous avons une séquence d'états qui peut être représentée par le graphe de fonction séquentielle montré à la Figure 6.14.

La transition définit les conditions qui doivent être satisfaites pour que le programme puisse passer à l'étape suivante et doivent donc se produire entre chacune de ses étapes. Tant que les conditions ne sont pas remplies, le programme

Copyright © 2015 Dunod.

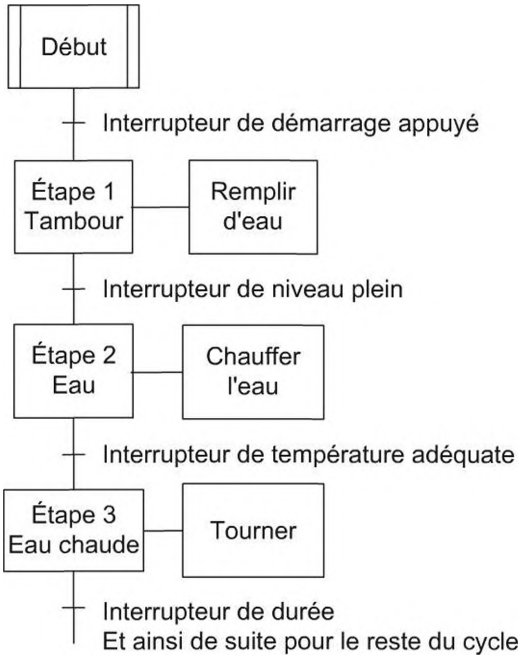


Figure 6.14 – Lave-linge.

poursuit son exécution de l'étape courante. Pour une transition, nous pouvons donc avoir la condition suivante :

```
Si vraie passer à l'étape suivante. Si fausse continuer l'étape précédente.
```

Voici d'autres possibilités :

```
Température > 50
Pompe = En marche
Vanne_Ouverte ET Compresseur_En_marche
Produit = 10
```

Elles doivent être satisfaites avant que le programme ne puisse passer à l'étape suivante. Lors de l'écriture du programme, une transition peut être donnée sous forme d'une expression booléenne en texte structuré (voir Section 6.3) afin de vérifier si une condition est vraie (1) ou fausse (0). Il est également possible d'appeler une sous-routine pour tester la condition requise et, lorsqu'elle est validée, de retourner une réponse à vrai (1), sinon à faux (0). À la fin de la sous-routine, une instruction EOT (*End of Transition*) permet de fixer l'état de la transition à la valeur booléenne déterminée par la sous-routine.

Des actions sont ajoutées aux étapes afin de préciser les différentes fonctions qu'elles doivent réaliser. Nous pouvons donc rencontrer la mise en marche d'un moteur, l'ouverture d'une vanne ou l'appel à une sous-routine.

Les graphes de fonction séquentielle représentent une technique graphique puissante pour décrire le comportement séquentiel d'un programme. Les langages graphiques sont utilisés depuis plusieurs années, notamment le GRAFCET. La norme CEI 61131-1, qui concerne les graphes de fonction séquentielle, emprunte énormément au GRAFCET.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

6.2.1 Divergence et convergence

La *divergence sélective* est illustrée à la Figure 6.15. Elle permet d'activer des états différents selon la condition de transition qui est validée. Des transitions sont attendues sous les lignes horizontales afin d'indiquer les conditions qui doivent être satisfaites avant de pouvoir aller de l'étape qui se trouve au-dessus de la ligne horizontale à celle qui se trouve en dessous.

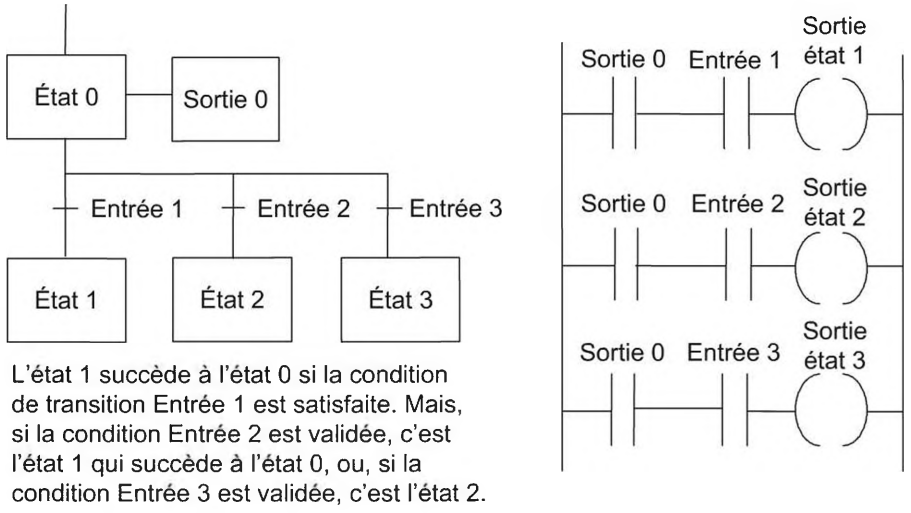


Figure 6.15 – Divergence sélective : l'état qui vient après l'état 0 dépend de la transition qui se produit (Entrée 1, Entrée 2 ou Entrée 3).

La *divergence parallèle* (voir Figure 6.16), représentée par deux lignes horizontales, permet à deux états différents, ou plus, d'être activés et d'être exécutés simultanément. Une transition est nécessaire à l'extérieur des deux lignes horizontales afin de préciser la condition qui doit être satisfaite pour que le groupe d'étapes qui viennent après soient activées simultanément.

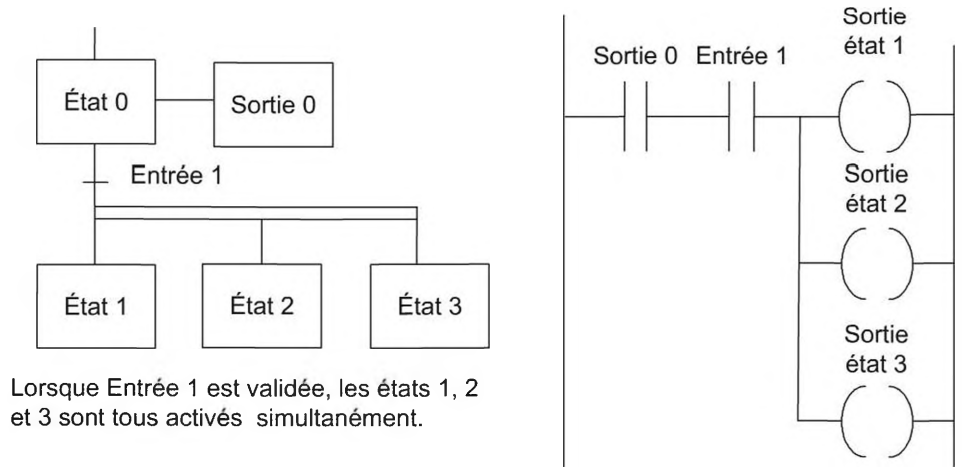


Figure 6.16 – Divergence parallèle : les états 1, 2 et 3 sont activés simultanément lorsque la condition de transition Entrée 1 est satisfaite.

Copyright © 2015 Dunod.

Les Figures 6.17 et 6.18 montrent comment représenter la convergence dans un graphe de fonction séquentielle. À la Figure 6.17, la séquence peut passer de l'état 2 à l'état 4 si la condition Entrée 4 est satisfaite, ou de l'état 3 à l'état 4 si la condition Entrée 5 est validée. Les transitions doivent être précisées au-dessus de la ligne horizontale afin de spécifier les conditions qui autorisent le passage depuis les étapes au-dessus de la ligne à celle en dessous. À la Figure 6.18, la séquence peut aller simultanément à l'état 3 depuis l'état 2 et l'état 4 lorsque la condition Entrée 4 est validée. La transition se trouve sous les deux lignes horizontales et indique les conditions qui doivent être satisfaites pour que le programme passe à l'étape suivante.

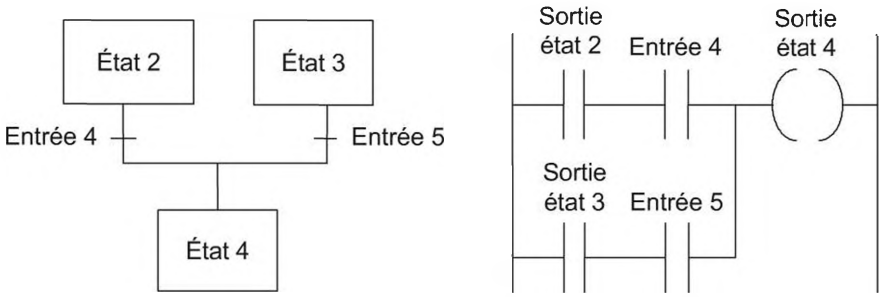


Figure 6.17 – Convergence : l'état 4 est activé lorsque Entrée 4 ou Entrée 5 est satisfaite.

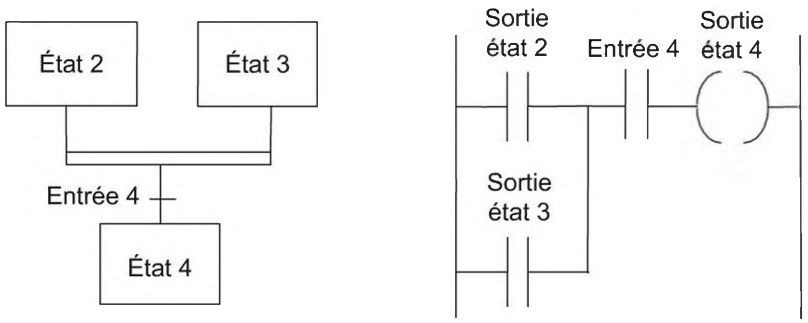


Figure 6.18 – Convergence simultanée : lorsque Entrée 4 est satisfaite, l'état 4 est activé à partir de l'état 2 ou de l'état 3.

Pour illustrer les divergences et les convergences, la Figure 6.19 présente une partie d'un programme donné sous forme d'un SFC et d'un schéma à contacts.

6.2.2 Actions

À chaque étape, une ou plusieurs actions doivent être effectuées. Ces actions, comme les sorties de l'exemple précédent, sont représentées par des rectangles reliés à l'état (voir Figure 6.20). Le comportement d'une action peut être décrit par un schéma à contacts, un diagramme de schémas fonctionnels, une liste d'instructions ou un texte structuré. Lorsqu'un schéma à contacts est utilisé, il est tracé à l'intérieur du rectangle de l'action. L'action représentée est ensuite effectuée lorsque l'état est activé. La Figure 6.21a illustre ce concept.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

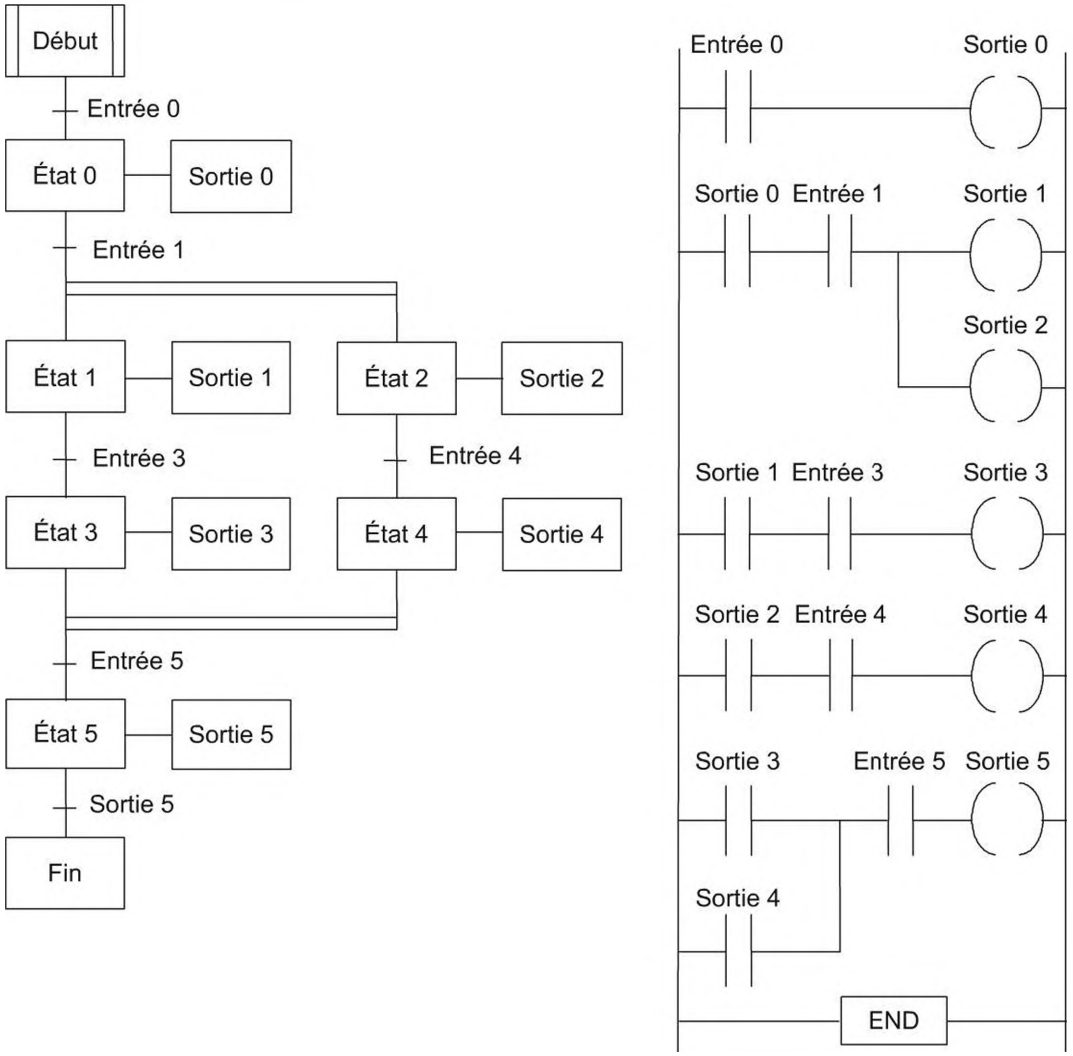


Figure 6.19 – Partie d'un SFC et le schéma à contacts équivalent.

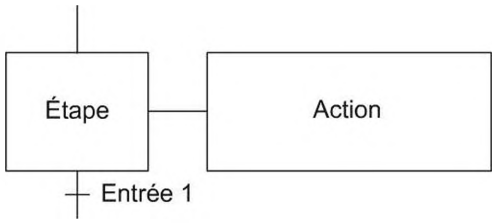


Figure 6.20 – Action ajoutée à une étape.

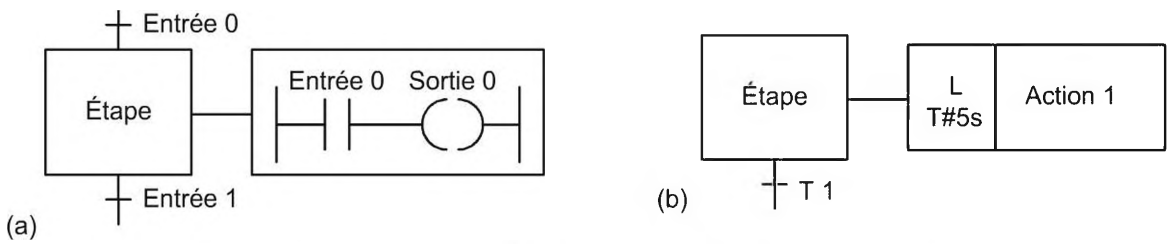


Figure 6.21 – (a) Action représentée par un schéma à contacts, (b) illustration d'un qualificatif utilisé avec une action, à durée limitée dans ce cas.

Copyright © 2015 Dunod.

Les rectangles d'action peuvent être précédés de qualificatifs qui précisent les conditions de l'action. En l'absence de qualificatif ou en présence du qualificatif N, l'action n'est pas mémorisée et est exécutée en continu pendant que l'état associé est actif. Elle s'arrête lorsqu'il est désactivé. Le qualificatif P est utilisé pour les actions impulsionnelles qui s'exécutent une seule fois lors de l'activation d'un état. Le qualificatif D concerne les actions retardées, qui ne démarrent qu'après un délai précisé et s'arrêtent lorsque l'étape est désactivée. Le qualificatif L indique des actions à durée limitée, qui démarrent lorsque l'étape est activée et se terminent après une durée indiquée (voir Figure 6.21b). Le Tableau 6.2 recense les qualificatifs d'action définis par la norme CEI 61131.

Tableau 6.2 Qualificatifs d'action

Qualificatif	Description
Aucun	Non mémorisée, par défaut, identique à N.
N	Non mémorisée, s'exécute tant que l'étape associée est active, puis s'arrête.
R	Réinitialise une action mémorisée.
S	Mémorisée. Reste active jusqu'à ce qu'une action de réinitialisation désactive cette action.
L	Action à durée limitée. Se termine après un temps donné, qui doit être précisé.
D	Action retardée. Démarre après un temps donné, qui doit être précisé.
P	Action impulsionnelle, qui se produit une fois lorsque l'étape est activée et une fois lorsqu'elle est désactivée.
P1	Action impulsionnelle qui se produit uniquement lorsqu'une étape est activée.
P0	Action impulsionnelle qui se produit uniquement lorsqu'une étape est désactivée.
SD	L'action démarre après un temps donné, même si l'étape associée est désactivée avant que cette durée soit écoulée.
DS	L'action démarre au bout d'un temps donné après que l'étape est activée et si elle est encore active. Elle s'arrête lorsque l'étape est désactivée.
SL	L'action démarre après que l'étape est activée. Elle est limitée dans le temps et s'exécute pendant la durée indiquée.

Lorsqu'une condition de transition est validée, qu'il reste une étape et que la suivante est démarrée, il est nécessaire d'examiner dans quel état l'action de cette

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

étape doit être laissé. Il sera peut-être nécessaire d'arrêter un dispositif activé par l'étape. Pour cela, il est possible d'écrire le programme de sorte que l'action de ce dispositif réalise cette opération. S'il faut revenir à l'état initial, il suffit de la programmer pour revenir à l'étape initiale.

6.2.3 Programmation d'un API

Les fabricants d'API fournissent l'outil qui permet de saisir des programmes destinés à leurs appareils. Voici par exemple la procédure générale d'utilisation du logiciel d'Allen-Bradley pour programmer un graphe de fonction séquentielle. Avec la barre d'outils SFC affichée à l'écran, il suffit de cliquer sur l'élément souhaité et de le faire glisser vers l'emplacement requis sur le graphe. L'élément qui correspond à une étape suivie d'une transition peut ainsi être ajouté de cette manière. Pour relier deux éléments, par exemple deux étapes, il suffit de cliquer sur la marque du premier élément puis sur celle du second. Un point de connexion valide est de couleur verte. Pour ajouter un branchement simultané au programme, comme un branchement sélectif, il suffit de cliquer sur le bouton correspondant de la barre d'outils et de faire glisser. La programmation d'une transition passe par un double-clic sur la zone de texte de l'icône de la transition ; si une expression booléenne est demandée, il suffit de la saisir. Pour appeler une sous-routine, cliquez du bouton droit sur l'icône de la transition, sélectionnez Set JSR puis la routine, et cliquez sur OK. Pour ajouter une action à une étape, il faut cliquer du bouton droit sur l'étape et choisir Add Action. Dans l'élément d'action, un clic sur le bouton affiche une série d'onglets qui permettent de préciser le qualificatif de l'action, par exemple L, l'ordre de l'action et une étiquette. Un clic du bouton droit sur l'entrée textuelle de l'élément d'action permet de choisir Set JSR et les sous-routines à ajouter.

6.3 Texte structuré

Le texte structuré est un langage de programmation qui ressemble énormément au langage Pascal. Les programmes sont écrits sous forme d'une suite d'instructions séparées par des points-virgules. Il s'agit d'instructions prédéfinies et de sous-routines qui permettent de modifier des variables, celles-ci étant des valeurs définies, des valeurs mémorisées de manière interne ou des entrées et des sorties.

Une instruction d'affectation permet de modifier la valeur d'une variable :

```
Lampe := InterrupteurA;
```

Cette instruction indique qu'une lampe, la variable, doit voir sa « valeur » modifiée, c'est-à-dire être allumée ou éteinte, lorsque l'interrupteur A change sa « valeur », c'est-à-dire marche ou arrêt. Voici le format général d'une instruction d'affectation :

```
X := Y;
```

Copyright © 2015 Dunod.

Y représente une expression qui produit une nouvelle valeur pour la variable X, tandis que := est le symbole d'affectation. La variable conserve la valeur qui lui a été affectée jusqu'à ce qu'une autre affectation la modifie. Voici un autre exemple :

```
Lampe := InterrupteurA OR InterrupteurB;
```

La lampe est allumée par l'interrupteur A ou par l'interrupteur B. La fonction ET existe également :

```
Marche := Vapeur AND Pompe;
```

La mise en marche se produit lorsque le détecteur de vapeur et la pompe sont actifs.

Tableau 6.3 Opérateurs utilisés dans un texte structuré

Opérateur	Description	Précédence
(...)	Expression parenthésée	La plus élevée
Function(...)	Liste des paramètres d'une fonction	
**	Élévation à une puissance	
-, NOT	Négation, NON booléen	
*, /, MOD	Multiplication, division, modulo	
+, -	Addition, soustraction	
<, >, <=, >=	Inférieur à, supérieur à, inférieur ou égal à, supérieur ou égal à	
=, <>	Égalité, inégalité	
AND, &	ET booléen	
XOR	Ou exclusif booléen	
OR	OU booléen	La plus basse

Le Tableau 6.3 recense quelques opérateurs, comme les opérateurs OR et AND des instructions précédentes, utilisés dans les programmes en texte structuré et leur priorité relative lors de l'évaluation d'une expression. Les parenthèses permettent de regrouper des expressions au sein d'autres expressions afin de fixer l'ordre d'évaluation. Par exemple :

```
EntreeA := 6;
EntreeB := 4;
EntreeC := 2;
SortieQ := EntreeA/3 + EntreeB/(3 - EntreeC);
```

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

Dans ce programme, l'expression (3 - EntreeC) est évaluée avant que sa valeur ne soit utilisée comme diviseur. La seconde partie de l'expression de SortieQ devient donc $4/(3-2) = 4$. Puisque la division est prioritaire sur l'addition, la première partie de l'expression de SortieQ, 6/3, est évaluée avant l'addition. Autrement dit, la valeur de SortieQ est $2 + 4 = 6$.

Le texte structuré n'est pas sensible à la casse. Les lettres majuscules ou minuscules peuvent donc être utilisées, par exemple pour faciliter la lecture. De même, les espaces ne sont pas obligatoires, mais elles permettent d'obtenir un code plus clair, tout comme l'indentation des lignes. Les identités des variables représentées directement commencent par le caractère % et sont suivies d'un code sur une ou deux lettres, qui indique si l'emplacement mémoire est associé à une entrée, une sortie ou à la mémoire interne, et s'il correspond à des bits, des octets ou des mots. Par exemple :

```
%IX100 (*Bit mémoire de l'entrée 100*)
%ID200 (*Mot mémoire de l'entrée 200*)
%QX100 (*Bit mémoire de la sortie 100*)
```

La première lettre est un I pour un emplacement mémoire d'entrée, Q pour une sortie ou M pour un emplacement mémoire interne. La deuxième lettre est X pour un bit, B pour un octet (huit bits), W pour un mot (seize bits), D pour un mot double (trente-deux bits) ou L pour un mot long (soixante-quatre bits).

L'opérateur AT permet de fixer l'emplacement mémoire d'une variable, par exemple :

```
Entree1 AT %IX100; (*Entree1 correspond au bit mémoire de l'entrée 100*)
```

6.3.1 Instructions conditionnelles

Avec l'instruction IF suivante, si la variable de température du fluide est active, c'est-à-dire à 1, les actions qui viennent après cette ligne dans le programme sont exécutées :

```
IF temp_fluide THEN
```

Avec l'instruction IF suivante, si la variable de température du fluide n'est pas à 1, les actions qui viennent après cette ligne dans le programme sont effectuées :

```
IF NOT temp_fluide THEN
```

Dans le cas suivant, les actions qui viennent après l'instruction IF dans le programme sont exécutées si la variable 1 de température du fluide est à 1 ou si la variable 2 de température du fluide est active, c'est-à-dire à 1 :

```
IF temp_fluide1 OR temp_fluide2 THEN
```

La combinaison IF ... THEN ... ELSE est utilisée pour exécuter des instructions sélectionnées lorsque certaines conditions se réalisent :

```
IF (Fin_de_course1 AND Piece_presente) THEN
    Porte1 := Ouvert;
    Porte2 := Ferme;
ELSE
```

Copyright © 2015 Dunod.

```
    Porte1 := Ferme;
    Porte2 := Ouvert;
END_IF;
```

Notez que la fin de l'instruction IF doit être indiquée. Voici un autre exemple avec des adresses d'API :

```
IF (I:000/00 = 1) THEN
    O:001/00 := 1;
ELSE
    O:000/01 := 0;
END_IF;
```

Dans ce cas, si l'entrée I:000/00 est à 1, la sortie O:001/00 est mise à 1 ; sinon elle est mise à 0.

L'instruction CASE est utilisée pour exécuter des instructions lorsqu'une valeur entière particulière est rencontrée, sinon d'autres instructions sont exécutées. Par exemple, pour un contrôle de température, nous pouvons avoir le code suivant :

```
CASE (Temperature) OF
    0 ... 40:        Interrupteur_four := Marche;
    40 ... 100:      Interrupteur_four := Arret;
ELSE
                     Interrupteur_four := Arret;
END_CASE;
```

À l'instar de toutes les instructions conditionnelles, la fin de l'instruction CASE doit être indiquée. Dans l'exemple suivant, les ventilateurs d'un moteur doivent tourner à des vitesses différentes en fonction de la position d'un interrupteur :

```
CASE Configuration_vitesse OF
    1:          vitesse := 5;
    2:          vitesse := 10;
    3:          vitesse := 15; ventilo1 := Marche;
    4:          vitesse := 20; ventilo2 := Marche;
ELSE
                vitesse := 0; defaut_vitesse := TRUE;
END_CASE
```

6.3.2 Instructions d'itération

Ces instructions permettent de répéter une ou plusieurs instructions un certain nombre de fois, en fonction de l'état d'une autre variable. La boucle FOR ... DO permet de répéter une suite d'instructions en fonction de la valeur d'une variable d'itération entière :

```
FOR Entree := 10 TO 0 BY -1
    DO
    Sortie := Entree;
END_FOR;
```

Dans cet exemple, la sortie diminue de 1 chaque fois que l'entrée, qui va de 10 à 0, diminue de 1.

La boucle WHILE ... DO permet d'exécuter une ou plusieurs instructions tant qu'une expression booléenne reste vraie :

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

```
SortieQ := 0;
WHILE EntreeA AND EntreeB
    DO
    SortieQ := SortieQ + 1;
END_WHILE;
```

La boucle REPEAT ... UNTIL permet d'exécuter une suite d'instructions et de la répéter tant qu'une expression booléenne reste vraie :

```
SortieQ := 0
REPEAT
    SortieQ := SortieQ + 1;
UNTIL (Entree1 = Arret) OR (SortieQ > 5)
END_REPEAT;
```

6.3.3 Écriture des programmes

Les programmes doivent commencer par définir les types qui représentent les données :

```
TYPE Moteur: (Arrete, EnFonction);
END_TYPE;

TYPE Vanne: (Ouverte, Fermee);
END_TYPE;

TYPE Pression: REAL; (*La pression est une valeur analogique*)
END_TYPE;
```

Ensuite viennent les variables, c'est-à-dire les signaux issus des capteurs et les signaux de sortie qui seront utilisés dans le programme :

```
VAR_IN (*Entrées*)
    Defaut_Pompe: BOOL; (*Le défaut de fonctionnement de la pompe*)
END_VAR;                (*est une variable booléenne.            *)

VAR_OUT (*Sorties*)
    Vitesse_Moteur: REAL;
END_VAR;

VAR_IN
    Valeur: INT; (*La valeur est un entier*)
END_VAR;

VAR
    Entree1 AT %IX100; (*Entree1 correspond au bit mémoire de l'entrée 100*)
END_VAR;
```

Des valeurs initiales doivent être données aux variables :

```
VAR
    Temp: REAL = 100; (*La valeur initiale est un nombre analogique 100*)
END_VAR;
```

Ce n'est qu'alors que vous pouvez ajouter les instructions.

Le code suivant est un exemple de bloc fonctionnel qui peut apparaître dans un programme plus vaste ; il vérifie des tensions :

Copyright © 2015 Dunod.

```
FUNCTION_BLOCK TEST_VOLTAGE
    VAR_INPUT
        VOLTS1, VOLTS2, VOLTS3
    END_VAR
    VAR_OUTPUT
        SURTENSION: BOOL;
    END_VAR
    IF VOLTS1 > 12 THEN
        SURTENSION := TRUE; RETURN;
    END_IF;
    IF VOLTS2 > 12 THEN
        SURTENSION := TRUE; RETURN;
    END_IF;
    IF VOLTS3 > 12 THEN
        SURTENSION := TRUE;
    END_IF;
END_FUNCTION_BLOCK;
```

Si le test des tensions 1, 2 et 3 montre que l'une d'elles est supérieure à 12, la sortie SURTENSION est fixée à TRUE et l'instruction RETURN est invoquée pour terminer l'exécution du bloc fonctionnel. Ailleurs dans le programme, lorsque SURTENSION est à TRUE, une certaine action sera réalisée.

6.3.4 Comparaison avec le langage à contacts

La Figure 6.22 montre un schéma à contacts et les expressions équivalentes en texte structuré. La Figure 6.23 présente un autre schéma à contacts, ainsi que ses équivalents sous forme de diagramme de schémas fonctionnels et de texte structuré.

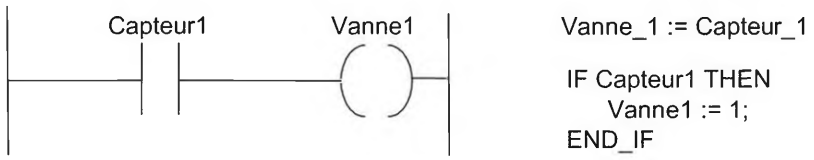


Figure 6.22 – Ligne d'un schéma à contacts et deux versions équivalentes en texte structuré.

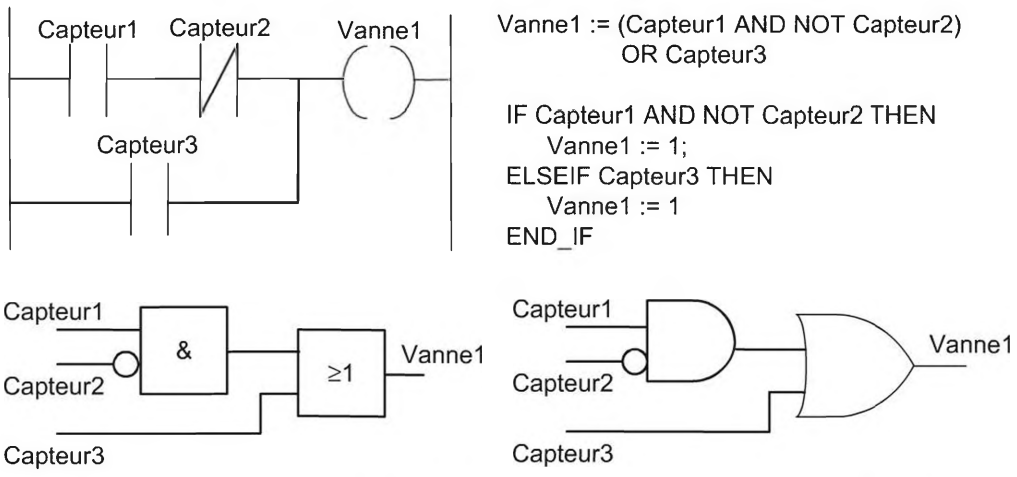


Figure 6.23 – Ligne d'un schéma à contacts, le diagramme de schémas fonctionnels équivalent et deux versions équivalentes en texte structuré.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

6.4 En résumé

Les listes d'instructions (IL, *Instruction List*) constituent une méthode de programmation que l'on peut comparer à la saisie d'un schéma à contacts sous forme d'un texte. Un programme écrit selon cette méthode est constitué d'une suite d'instructions, chacune sur une nouvelle ligne. Chaque instruction est constituée d'un opérateur suivi d'un ou plusieurs opérandes, c'est-à-dire les objets de l'opérateur. Des codes mnémoniques sont employés pour les opérateurs, chaque code correspondant à un opérateur/élément d'un schéma à contacts.

Un graphe de fonction séquentielle (SFC, *Sequential Function Chart*) correspond à une représentation graphique du fonctionnement d'un système afin de révéler l'enchaînement des événements qui conduisent à ce fonctionnement. Le fonctionnement est décrit par une suite d'étapes ou d'états distincts reliés séquentiellement et représentés par des rectangles. Chaque rectangle correspond à un état particulier du système contrôlé. Chaque ligne de connexion entre des états comprend une barre horizontale, qui représente la condition de transition qui doit être satisfaite pour que le système puisse passer d'un état au suivant. Lorsque la condition de passage à l'état suivant est satisfaite, l'étape ou l'état suivant du programme est activé. Le processus se poursuit d'un état au suivant, jusqu'à ce que le cycle complet de la machine soit effectué. Les sorties/actions d'un état sont indiquées dans une boîte connectée horizontalement à l'état et sont réalisées lorsque celui-ci est atteint.

Avec le texte structuré (ST, *Structured Text*), les programmes sont écrits sous forme d'une suite d'instructions séparées par des points-virgules. Il s'agit d'instructions prédéfinies et de sous-routines qui permettent de modifier des variables, celles-ci étant des valeurs définies, des valeurs mémorisées de manière interne ou des entrées et des sorties. Les instructions d'affectation permettent de modifier la valeur d'une variable, par exemple `X := Y`. Le texte structuré n'est pas sensible à la casse et les espaces ne sont pas obligatoires, mais elles permettent d'obtenir un code plus lisible. La combinaison IF ... THEN ... ELSE est utilisée pour exécuter des instructions sélectionnées lorsque certaines conditions se réalisent. L'instruction CASE est utilisée pour exécuter des instructions lorsqu'une valeur entière particulière est rencontrée, sinon d'autres instructions sont exécutées. La boucle FOR ... DO permet de répéter une suite d'instructions en fonction de la valeur d'une variable d'itération entière. La boucle WHILE ... DO permet d'exécuter une ou plusieurs instructions tant qu'une expression booléenne reste vraie. La boucle REPEAT ... UNTIL permet d'exécuter une suite d'instructions et de la répéter tant qu'une expression booléenne reste vraie.

6.5 Problèmes

Pour les questions 1 à 24, quatre réponses sont proposées : A, B, C et D. Choisissez la réponse correcte.

Copyright © 2015 Dunod.

1. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). La liste d'instructions suivante :

```
LD      X401
AND     X402
OUT     Y430
```

décrit une ligne d'un schéma à contacts dans laquelle il existe une sortie si :
(i) l'entrée X401 est activée et l'entrée X402 n'est pas activée ;
(ii) les entrées X401 et X402 sont toutes deux activées.

Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
A. (i) V (ii) V
B. (i) V (ii) F
C. (i) F (ii) V
D. (i) F (ii) F

2. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). La liste d'instructions suivante :

```
LD      X401
OR      X402
OUT     Y430
```

décrit une ligne d'un schéma à contacts dans laquelle il existe une sortie si :
(i) l'entrée X401 est activée et l'entrée X402 n'est pas activée ;
(ii) l'entrée X402 est activée et l'entrée X401 n'est pas activée.

Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
A. (i) V (ii) V
B. (i) V (ii) F
C. (i) F (ii) V
D. (i) F (ii) F

3. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). La liste d'instructions suivante :

```
LD      X401
ANI     X402
OUT     Y430
```

décrit une ligne d'un schéma à contacts dans laquelle il existe une sortie si :
(i) l'entrée X401 est activée et l'entrée X402 n'est pas activée ;
(ii) les entrées X401 et X402 sont toutes deux activées.

Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
A. (i) V (ii) V
B. (i) V (ii) F
C. (i) F (ii) V
D. (i) F (ii) F

4. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). La liste d'instructions suivante :

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

```
LDI     X401
ANI     X402
OUT     Y430
```

décrit une ligne d'un schéma à contacts dans laquelle il existe une sortie si :
(i) l'entrée X401 est activée et l'entrée X402 n'est pas activée ;
(ii) les entrées X401 et X402 sont toutes deux activées.

Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?

A. (i) V (ii) V
B. (i) V (ii) F
C. (i) F (ii) V
D. (i) F (ii) F

5. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). La liste d'instructions suivante :

```
LD      X401
OR      Y430
ANI     X402
OUT     Y430
```

décrit une ligne d'un schéma à contacts dans laquelle la sortie :
(i) est présente lorsque l'entrée X401 est momentanément activée ;
(ii) est absente lorsque X402 est activée.

Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?

A. (i) V (ii) V
B. (i) V (ii) F
C. (i) F (ii) V
D. (i) F (ii) F

6. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). La liste d'instructions suivante :

```
A       I0.1
A       I0.2
=       Q2.0
```

décrit une ligne d'un schéma à contacts dans laquelle il existe une sortie si :
(i) l'entrée I0.1 est activée et l'entrée I0.2 n'est pas activée ;
(ii) les entrées I0.1 et I0.2 sont toutes deux activées.

Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?

A. (i) V (ii) V
B. (i) V (ii) F
C. (i) F (ii) V
D. (i) F (ii) F

7. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). La liste d'instructions suivante :

```
A       I0.1
O       I0.2
=       Q2.0
```

Copyright © 2015 Dunod.

décrit une ligne d'un schéma à contacts dans laquelle il existe une sortie si :
(i) l'entrée I0.1 est activée et l'entrée I0.2 n'est pas activée ;
(ii) l'entrée I0.2 est activée et l'entrée I0.1 n'est pas activée.

Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?

A. (i) V (ii) V
B. (i) V (ii) F
C. (i) F (ii) V
D. (i) F (ii) F

8. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). La liste d'instructions suivante :

```
A       I0.1
AN      I0.2
=       Q2.0
```

décrit une ligne d'un schéma à contacts dans laquelle il existe une sortie si :
(i) l'entrée I0.1 est activée et l'entrée I0.2 n'est pas activée ;
(ii) les entrées I0.1 et I0.2 sont toutes deux activées.

Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?

A. (i) V (ii) V
B. (i) V (ii) F
C. (i) F (ii) V
D. (i) F (ii) F

9. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). La liste d'instructions suivante :

```
AN      I0.1
AN      I0.2
=       Q2.0
```

décrit une ligne d'un schéma à contacts dans laquelle il existe une sortie si :
(i) l'entrée I0.1 est activée et l'entrée I0.2 n'est pas activée ;
(ii) les entrées I0.1 et I0.2 sont toutes deux activées.

Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?

A. (i) V (ii) V
B. (i) V (ii) F
C. (i) F (ii) V
D. (i) F (ii) F

10. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). La liste d'instructions suivante :

```
A       I0.1
O       Q2.0
AN      I0.2
=       Q2.0
```

décrit une ligne d'un schéma à contacts dans laquelle la sortie :

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

(i) est présente lorsque l'entrée I0.1 est momentanément activée ;
(ii) est absente lorsque I0.2 est activée.

Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?

A. (i) V (ii) V
B. (i) V (ii) F
C. (i) F (ii) V
D. (i) F (ii) F

11. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). La liste d'instructions suivante :

```
LD      X401
OUT     Y430
LDI     X401
OUT     Y431
```

décrit un programme pour lequel :

(i) lorsque X401 est activée, une sortie existe sur Y430 mais non sur Y431 ;
(ii) lorsque X401 n'est pas activée, une sortie existe sur Y430 mais non sur Y431.

Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?

A. (i) V (ii) V
B. (i) V (ii) F
C. (i) F (ii) V
D. (i) F (ii) F

12. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). La liste d'instructions suivante :

```
LD      X400
OR      X401
OR      X402
AND     X403
OUT     Y431
```

décrit un programme pour lequel il existe une sortie sur Y431 lorsque :

(i) uniquement X400, X401 ou X402 est activée ;
(ii) uniquement X400 et X403 sont activées.

Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?

A. (i) V (ii) V
B. (i) V (ii) F
C. (i) F (ii) V
D. (i) F (ii) F

13. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). La liste d'instructions suivante :

```
LD      X400
AND     X401
OR      X402
OUT     Y430
```

Copyright © 2015 Dunod.

décrit un programme pour lequel il existe une sortie sur Y430 lorsque :

(i) uniquement X400 ou X402 est activée ;

(ii) uniquement X400 et X401 sont activées.

Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?

A. (i) V (ii) V
B. (i) V (ii) F
C. (i) F (ii) V
D. (i) F (ii) F

14. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Pour le graphe de fonction séquentielle de la Figure 6.24 :

(i) l'état 1 est activé lorsque la condition X1 est validée ;

(ii) la sortie 1 se produit lorsque la condition X2 est validée.

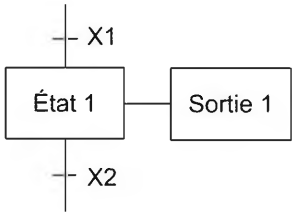


Figure 6.24 – Diagramme pour le problème 14.

Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?

A. (i) V (ii) V
B. (i) V (ii) F
C. (i) F (ii) V
D. (i) F (ii) F

15. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Pour le graphe de fonction séquentielle de la Figure 6.25, si l'état 1 est activé :

(i) l'état 2 est activé lorsque la condition X2 est validée ;

(ii) l'état 3 est activé lorsque la condition X3 est validée.

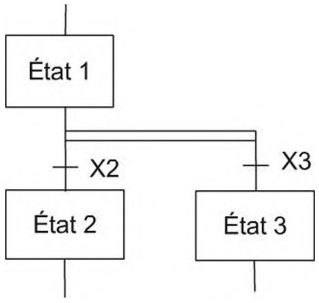


Figure 6.25 – Diagramme pour le problème 15.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?

A. (i) V (ii) V
B. (i) V (ii) F
C. (i) F (ii) V
D. (i) F (ii) F

16. Quel graphe de fonction séquentielle de la Figure 6.26b correspond au schéma à contacts de la Figure 6.26a ?

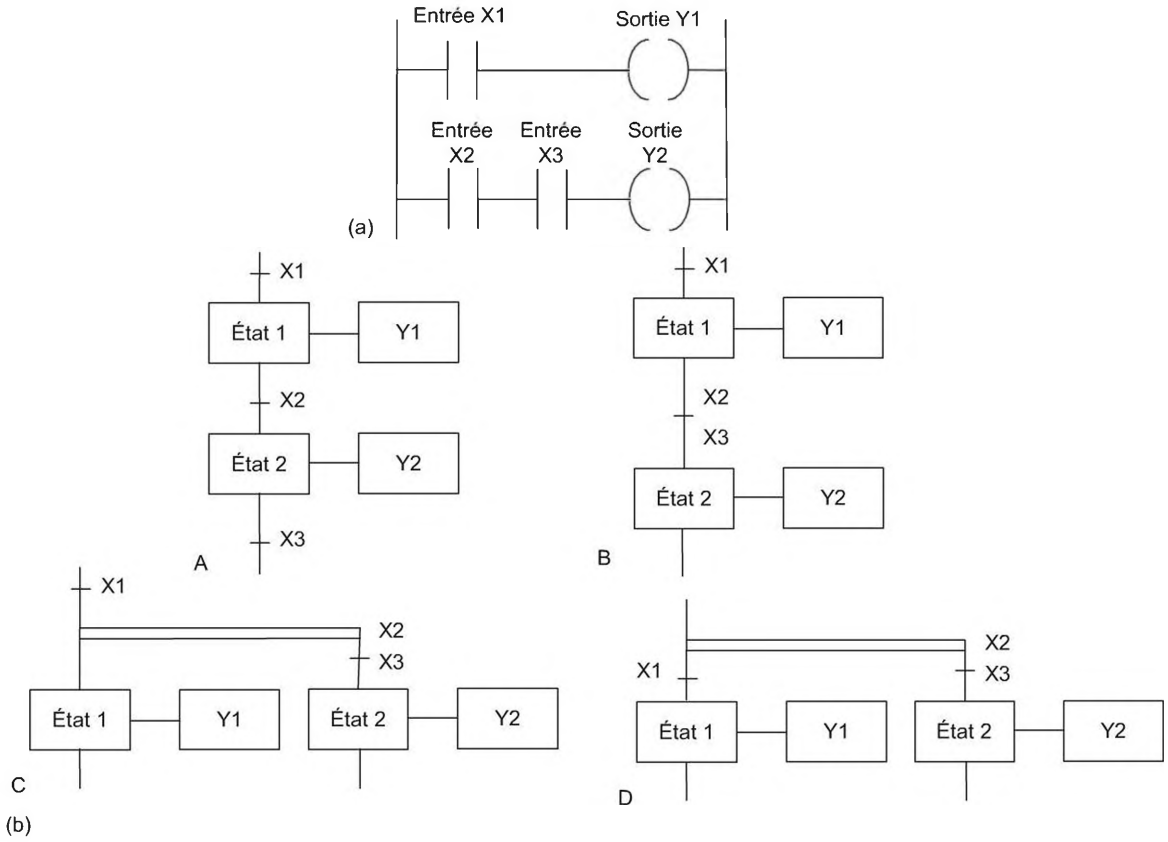


Figure 6.26 – Diagramme pour le problème 16.

Les problèmes 17, 18 et 19 font référence au graphe de fonction séquentielle de la Figure 6.27.

17. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Pour le graphe de fonction séquentielle donné à la Figure 6.27, la sortie Y2 se produit si la sortie Y1 s'est produite et si :
(i) les conditions X2 et X3 sont toutes deux validées ;
(ii) uniquement la condition X2 est validée.

Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?

A. (i) V (ii) V
B. (i) V (ii) F
C. (i) F (ii) V
D. (i) F (ii) F

Copyright © 2015 Dunod.

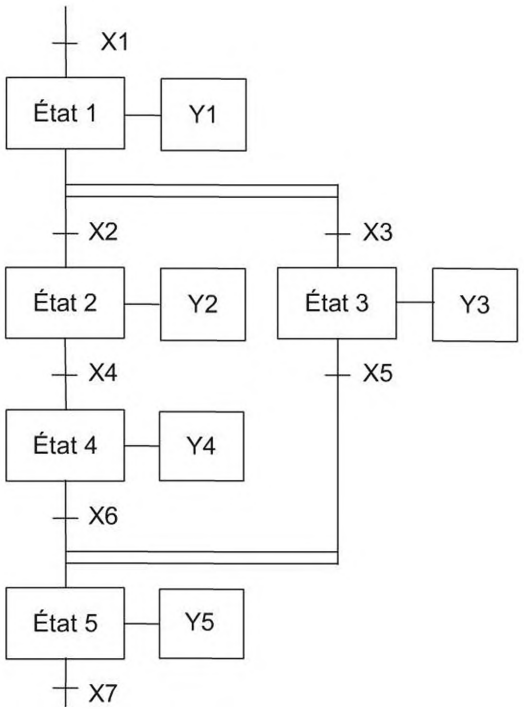


Figure 6.27 – Diagramme pour les problèmes 17, 18 et 19.

18. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Pour le graphe de fonction séquentielle donné à la Figure 6.27, la sortie Y4 se produit si la sortie Y2 s'est produite et si :
    (i) la sortie Y3 s'est produite ;
    (ii) la condition X4 est validée.

    Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
    A. (i) V (ii) V
    B. (i) V (ii) F
    C. (i) F (ii) V
    D. (i) F (ii) F

19. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Pour le graphe de fonction séquentielle donné à la Figure 6.27, la sortie Y5 se produit si :
    (i) la sortie Y4 s'est produite et la condition X6 est validée ;
    (ii) la sortie Y3 s'est produite et la condition X5 est validée.

    Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
    A. (i) V (ii) V
    B. (i) V (ii) F
    C. (i) F (ii) V
    D. (i) F (ii) F

20. Quel schéma à contacts de la Figure 6.29 correspond au graphe de fonction séquentielle de la Figure 6.28 ?

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

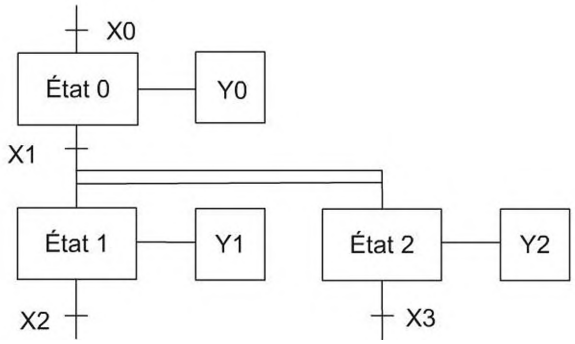


Figure 6.28 – Diagramme pour le problème 20.

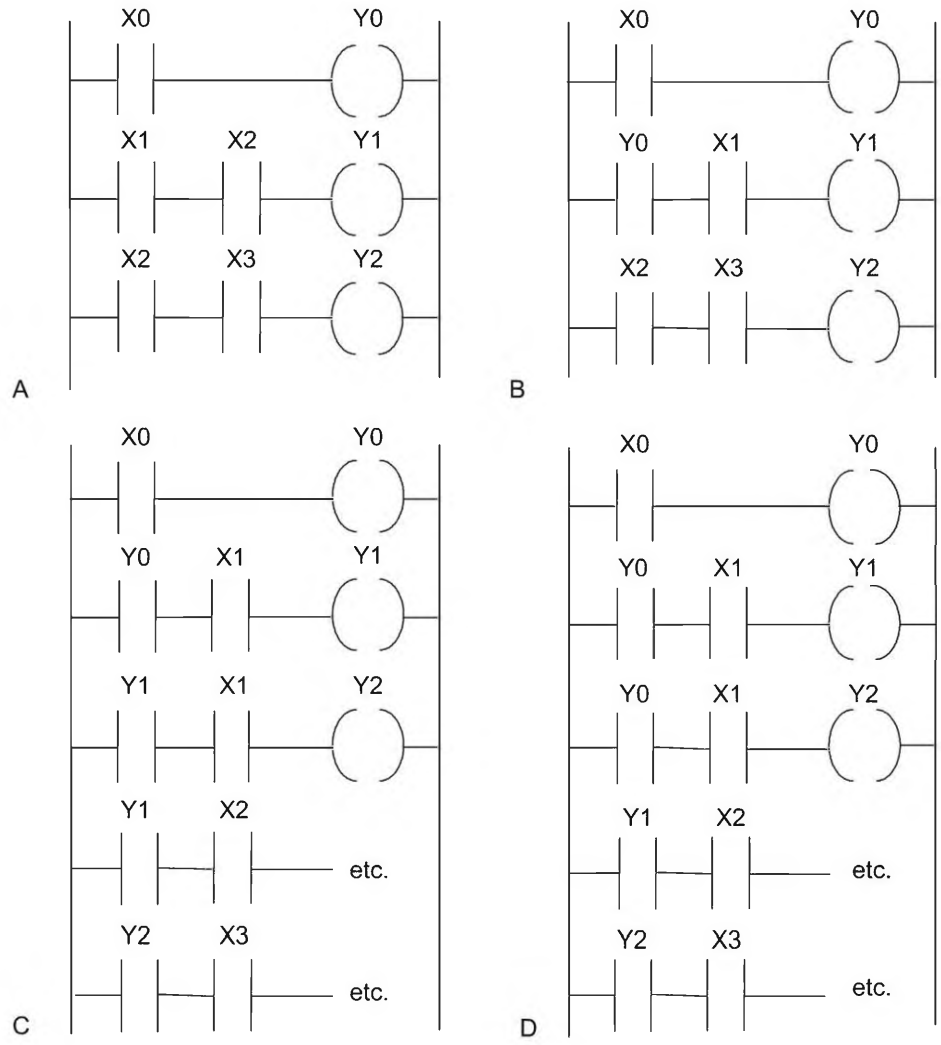


Figure 6.29 – Diagramme pour le problème 20.

21. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Pour le programme en texte structuré suivant :

```
VAR
    i: INT;
END_VAR;
i := 0;
```

Copyright © 2015 Dunod.

```
REPEAT
    i := i + 1;
UNTIL i = 5;
END_REPEAT;
```

(i) la variable i peut avoir uniquement les valeurs 0 ou 1 ;
(ii) chaque fois que le programme répète les instructions de la boucle, la valeur de la variable i augmente de 1.

Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?

A. (i) V (ii) V
B. (i) V (ii) F
C. (i) F (ii) V
D. (i) F (ii) F

22. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Pour le programme en texte structuré suivant :

```
IF Entree1 THEN
    Moteur := 1;
END_IF;
IF Entree2 THEN
    Moteur := 0;
END_IF;
```

(i) le moteur est démarré lorsque l'entrée 1 se produit ;
(ii) le moteur est arrêté lorsque l'entrée 2 se produit.

Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?

A. (i) V (ii) V
B. (i) V (ii) F
C. (i) F (ii) V
D. (i) F (ii) F

23. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Pour le programme en texte structuré suivant :

```
IF (Fin_de_course_1 AND Piece_presente) THEN
    Porte_1 := Ouverte;
    Porte_2 := Fermee;
ELSE
    Porte_1 := Fermee;
    Porte_2 := Ouverte;
END_IF;
```

(i) si seule la pièce est en place, la porte 1 est ouverte et la porte 2 est fermée ;
(ii) si seul l'interrupteur de fin de course est activé, la porte 1 est fermée et la porte 2 est ouverte.

Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?

A. (i) V (ii) V
B. (i) V (ii) F
C. (i) F (ii) V
D. (i) F (ii) F

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

24. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Pour le programme en texte structuré suivant :

```
VAR
    Mise_en_marche AT %IX120;
END_VAR
```

(i) `Mise_en_marche` correspond au bit mémoire de l'entrée 120 ;
(ii) `Mise_en_marche` a la valeur 120 bits.

Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?

A. (i) V (ii) V
B. (i) V (ii) F
C. (i) F (ii) V
D. (i) F (ii) F

25. Tracez un graphe de fonction séquentielle qui permet de suivre le fonctionnement d'un interrupteur de mise en marche afin de déclencher le remplissage d'un réservoir en ouvrant la vanne 1, jusqu'à ce qu'un interrupteur de niveau 1 soit activé. Ensuite, le réservoir est vidangé en ouvrant la vanne 2, jusqu'à ce qu'un interrupteur de niveau 2 soit activé. Puis, la procédure se répète.
26. Écrivez un programme en texte structuré pour le processus suivant : un réservoir se remplit lorsque la vanne 1 est ouverte, tant qu'un interrupteur de niveau 1 n'est pas activé et que la vanne de vidange est fermée.
27. Écrivez un programme en texte structuré pour fixer, à l'aide d'interrupteurs, la température d'une enceinte aux valeurs 40, 50, 60 et 70, et mettre en marche le ventilateur 1 lorsque la température est égale à 60 et le ventilateur 2, lorsqu'elle est égale à 70.

Copyright © 2015 Dunod.

7 • RELAIS INTERNES

En présentant les *relais internes*, ce chapitre se veut une continuité des précédents, qui traitent de la programmation. D'autres termes sont parfois employés pour décrire ces éléments, comme *relais auxiliaires*, *marqueurs*, *fanions*, *bobines* et *stockage binaire*. Ces éléments font partie des fonctionnalités spéciales intégrées aux API et largement utilisées dans les programmes. Un petit API peut disposer d'une centaine de relais internes, voire plus, certains étant sauvegardés par batterie et utilisés, par exemple, pour garantir l'arrêt propre d'un équipement en cas de panne d'alimentation. Les chapitres ultérieurs présenteront d'autres éléments intégrés courants.

7.1 Relais internes

Dans les API, certains éléments sont utilisés pour conserver des données, c'est-à-dire des bits, et se comportent comme des relais, capables d'être allumés ou éteints et d'allumer ou d'éteindre d'autres dispositifs. Voilà donc l'origine du terme *relais interne*. Ces relais internes n'existent pas en tant que dispositif de commutation réels. Il s'agit simplement de bits dans la mémoire de l'API qui se comportent comme des relais. Du point de vue programmation, ils peuvent être considérés de la même manière qu'un relais externe de sortie ou d'entrée. Ainsi, les entrées d'interrupteurs externes peuvent être utilisées pour produire une sortie sur un relais interne. Ensuite, les contacts du relais interne, conjointement à d'autres interrupteurs d'entrée externes, peuvent produire une autre sortie, comme activer un moteur. Nous avons ainsi le fonctionnement suivant (voir Figure 7.1) :

Sur une ligne du programme :

Les entrées sur des entrées externes activent la sortie du relais interne.

Sur une autre ligne du programme :

Suite à l'activation de la sortie du relais interne, les contacts du relais interne se ferment et commandent une autre sortie.

Pour utiliser un relais interne, il doit être activé par une ligne d'un programme et sa sortie doit ensuite servir à actionner les contacts d'un interrupteur sur une ou plusieurs autres lignes du programme. Les relais internes peuvent être programmés avec autant d'ensembles de contacts que nécessaire.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

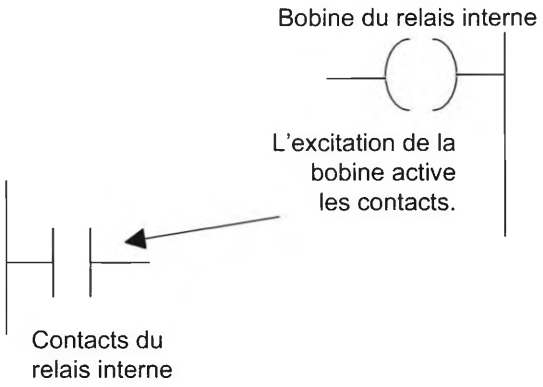


Figure 7.1 – Relais interne.

Pour faire la différence entre des sorties de relais internes et des sorties de relais externes, leurs adresses sont de type différent. Les fabricants ont tendance à employer des termes différents pour les relais internes et à exprimer leur adresse d'une autre manière. Par exemple, Mitsubishi emploie les termes *relais auxiliaires* ou *marqueurs* et la notation M100, M101, etc. Siemens utilise le terme *fanion* et la notation F0.0, F0.1, etc. Telemecanique a choisi le terme *bit* et la notation B0, B1, etc. Toshiba a opté pour *relais internes* et la notation R000, R001, etc. Allen-Bradley a retenu le terme *stockage binaire* et la notation B3/001, B3/002, etc. pour les API PLC-5.

7.2 Programmes en langage à contacts

Dans un schéma à contacts, la sortie d'un relais interne est représentée par le symbole d'un dispositif de sortie, ( ), et une adresse qui indique sa nature de relais interne. Par exemple, avec un API de Mitsubishi, nous pouvons avoir l'adresse M100, où M indique qu'il s'agit d'un relais interne, ou marqueur, à la place d'un dispositif externe. Les contacts du relais interne sont représentés par le symbole d'un dispositif d'entrée, | |, et prennent la même adresse que la sortie du relais interne, comme M100.

7.2.1 Programmes avec plusieurs conditions d'entrée

Afin d'illustrer les possibilités des relais internes, prenons le cas suivant. Un système doit être activé lorsque deux ensembles différents de conditions d'entrée sont validés. Nous pouvons programmer ce système simplement avec des portes logiques ET. Toutefois, si plusieurs entrées doivent être examinées pour que chaque condition d'entrée puissent être validées, il est sans doute plus simple d'employer un relais interne. Les premières conditions d'entrée sont ensuite utilisées pour produire une sortie sur un relais interne. Les contacts de ce relais font ensuite partie des conditions d'entrée de la seconde entrée.

La Figure 7.2 présente le schéma à contacts qui met en œuvre ce fonctionnement. Sur la première ligne, lorsque l'entrée E1 ou l'entrée E3 est fermée et lorsque l'entrée E2 est fermée, le relais interne RI1 est activé. Cela déclenche la fermeture

Copyright © 2015 Dunod.

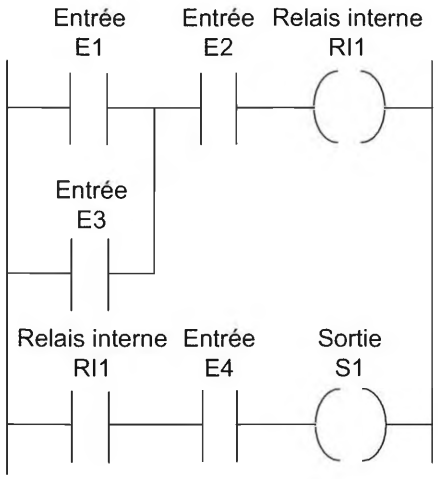


Figure 7.2 – Relais interne.

de ses contacts. Si l'entrée E4 est ensuite activée, il se produit une sortie sur S1. Ce type de fonctionnement peut se rencontrer dans le levage automatique d'une barrière lorsqu'une personne s'en approche d'un côté ou de l'autre. Les entrées E1 et E3 proviennent de capteurs photoélectriques qui détectent la présence d'une personne, qu'elle s'approche ou qu'elle s'éloigne de la barrière. E1 est activée pour l'un des côtés, E3, pour l'autre. L'entrée E2 est un interrupteur qui permet au système d'être fermé. Ainsi, lorsque E1 ou E3, et E2, sont activées, il existe une sortie sur le relais interne RI1 et ses contacts se ferment. Si l'entrée E4, sans doute un interrupteur de fin de course, détecte que la barrière est fermée, il est alors activé et fermé. Le résultat est une sortie pour S1, par exemple un moteur qui lève la barrière. Si l'interrupteur de fin de course détecte que la barrière est déjà ouverte, puisque la personne l'a traversée, il s'ouvre alors et la sortie S1 n'est plus activée. Un contrepoids se charge ensuite de fermer la barrière. Le relais interne a permis de lier deux parties du programme, l'une étant la détection de la présence d'une personne, l'autre, la détection de la barrière déjà levée ou abaissée. Les Figures 7.3a et 7.3b montrent, respectivement, comment représenter le schéma de la Figure 7.2 selon la notation de Mitsubishi et de Siemens.

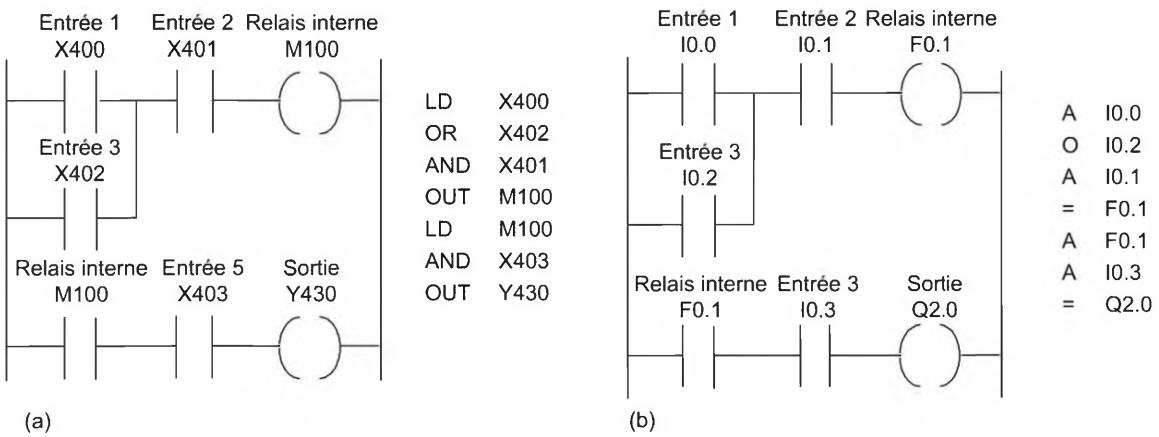


Figure 7.3 – La Figure 7.2 dans la notation de (a) Mitsubishi et de (b) Siemens.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

La Figure 7.4 est un autre exemple de schéma à contacts qui implique des relais internes. La sortie S1 est commandée par deux systèmes d'entrées. La première ligne comprend le relais interne RI1, qui est excité lorsque l'entrée E1 ou l'entrée E2 est activée. La deuxième ligne inclut le relais interne RI2, qui est excité lorsque les entrées E3 et E4 sont activées. La troisième ligne montre que la sortie S1 est présente lorsque le relais interne RI1 ou le relais interne RI2 est activé. Autrement dit, il existe une sortie sur le système lorsque l'un des deux ensembles de conditions d'entrées est validé.

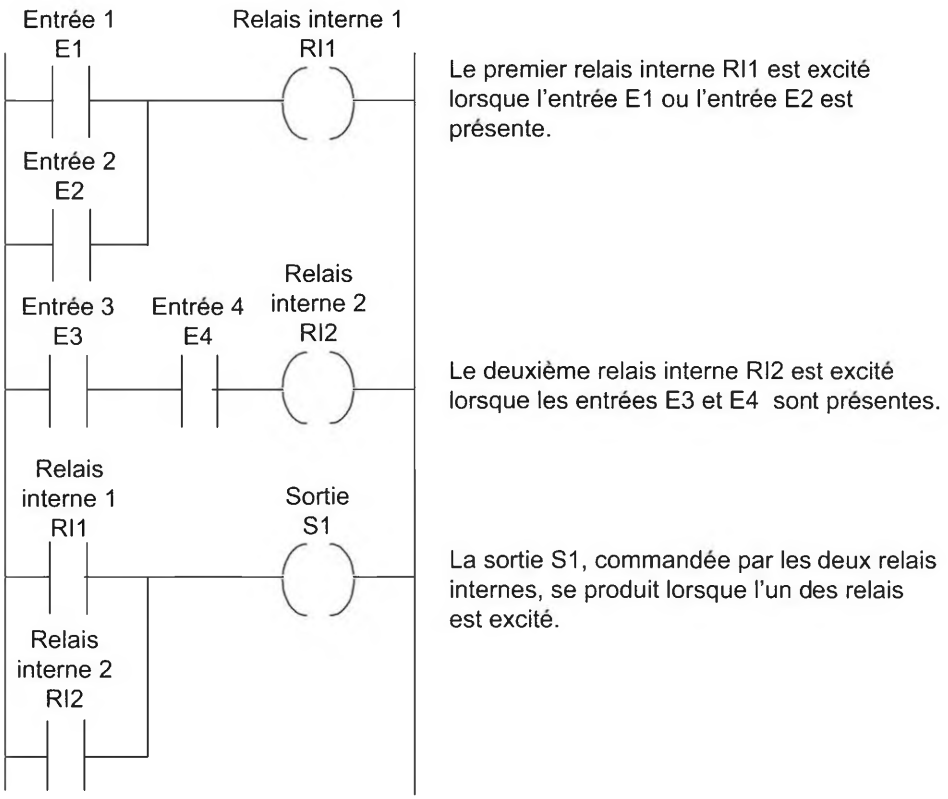


Figure 7.4 – Utilisation de deux relais internes.

7.2.2 Programmes à verrouillage

Les relais internes peuvent également être utilisés pour réinitialiser un circuit de verrouillage. La Figure 7.5 montre un exemple de schéma à contacts qui met en œuvre ce type d'utilisation.

Lorsque les contacts de l'entrée E1 se ferment momentanément, une sortie est présente sur S1. Les contacts de S1 se ferment donc et la sortie est maintenue même lorsque l'entrée E1 disparaît. Lorsque l'entrée E2 est fermée, le relais interne RI1 est excité et ses contacts, normalement fermés, s'ouvrent. La sortie S1 est alors désactivée et déverrouillée.

Examinons un cas qui requiert des circuits de verrouillage, car une machine automatique est démarrée ou arrêtée à l'aide de boutons-poussoirs. Un circuit à verrouillage applique ou coupe l'alimentation de la machine. Celle-ci possède plusieurs sorties, qui sont activées lorsque l'alimentation est appliquée, et désactivées lorsqu'elle est coupée. Il est possible de concevoir un schéma à contacts qui se

Copyright © 2015 Dunod.

fonde sur des commandes verrouillées pour chacune de ces sorties. Toutefois, une méthode plus simple consiste à utiliser un relais interne (voir Figure 7.6). La première ligne comprend le verrou qui maintient le relais interne RI1 excité lors d'une entrée momentanée sur le bouton de mise en marche. La deuxième ligne est alors mise sous tension. La troisième ligne est également mise sous tension et produit la sortie S2 si les contacts de l'entrée 2 sont fermés. La quatrième ligne est également mise sous tension et produit la sortie S3 si les contacts de l'entrée 3 sont fermés. Ainsi, toutes les sorties peuvent être activées par un appui sur le bouton de mise en marche. Toutes les sorties seront désactivées si l'interrupteur d'arrêt est ouvert. Par conséquent, toutes les sorties sont verrouillées par le relais interne RI1.

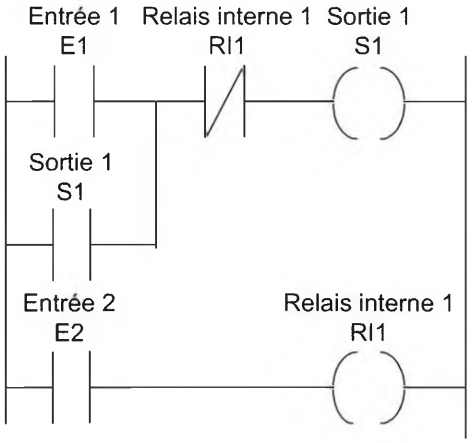


Figure 7.5 – Réinitialisation d'un circuit de verrouillage.

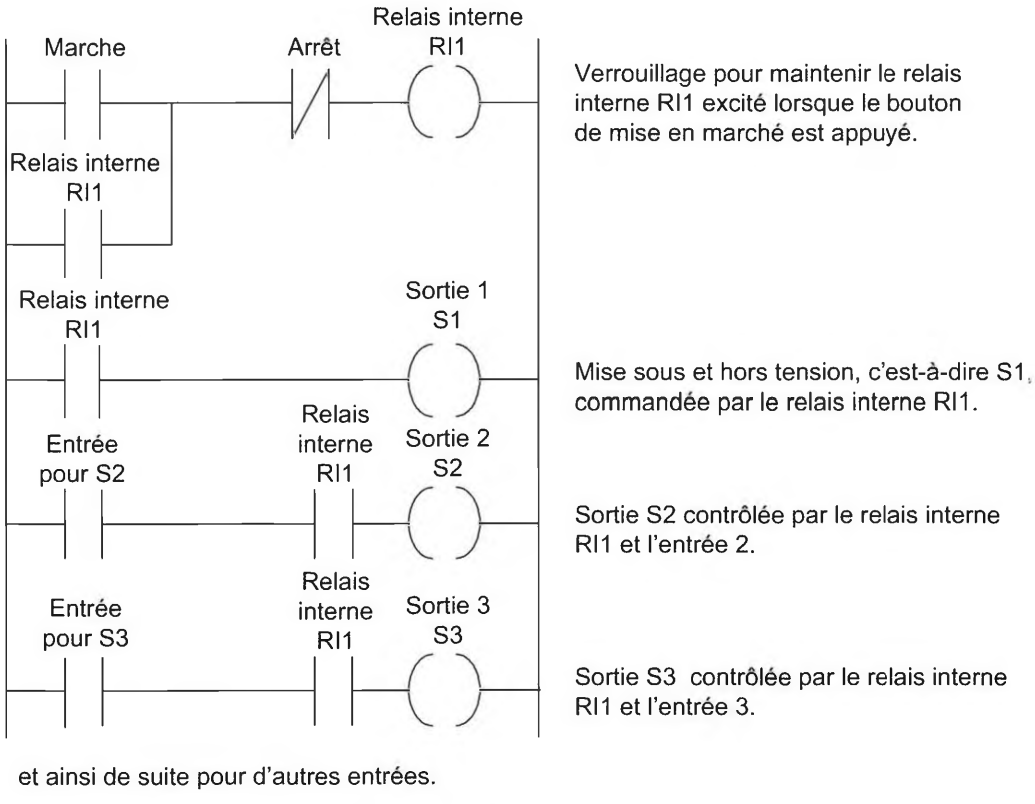
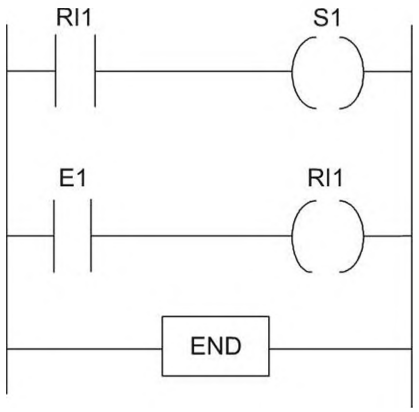


Figure 7.6 – Activation de plusieurs sorties.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

7.2.3 Temps de réponse

Le temps qui sépare l'arrivée d'une entrée et le changement résultant en sortie dépend de plusieurs facteurs, comme le temps de réponse électrique du circuit d'entrée, le temps de réponse mécanique du dispositif de sortie et le temps d'analyse du programme. Un schéma à contacts est lu de gauche à droite et de haut en bas. Par conséquent, si un dispositif de sortie, comme un relais interne, est activé lors d'un cycle d'analyse et si la sortie est utilisée sur une ligne précédente du programme, il faudra attendre l'analyse suivante pour qu'elle soit prise en compte. La Figure 7.7 illustre ce principe.



RI1 n'est pas excité suite à l'entrée sur E1 lors de la première analyse, il faut attendre la seconde analyse du programme.

Figure 7.7 – Décalage dans les réponses dû au temps d'analyse.

7.3 Relais sauvegardés par batterie

Si l'alimentation d'un API est coupée alors qu'il est en utilisation, tous les relais de sortie et tous les relais internes sont désactivés. Lorsque l'alimentation revient, les contacts associés à ces relais ne sont plus dans l'état où ils se trouvaient au moment de la coupure. Par conséquent, si l'API se trouvait au milieu d'une séquence d'actions de commande, il reprend en un point différent de la séquence. Pour résoudre ce problème, certains relais internes sont sauvegardés par batterie et peuvent ainsi être utilisés dans les circuits pour garantir un arrêt propre d'un équipement lors d'une panne de courant et lui permettre de redémarrer de manière appropriée. Ces relais sauvegardés par batterie conservent leur état d'activation même lorsque l'alimentation est coupée. Le relais est dit *à rémanence*.

Le terme *bobine mémoire rémanente* est parfois employé avec ce type d'éléments. La Figure 7.8a présente le symbole que la CEI 61131-3 définit pour ces éléments. Dans les API de Mitsubishi, les relais internes sauvegardés par batterie sont référencées par les adresses M300 à M377. D'autres fabricants emploient des adresses et des méthodes différentes pour obtenir la mémoire rémanente. Ainsi, le modèle PLC-5 d'Allen-Bradley se fonde sur des lignes avec et sans verrouillage. Si le relais est verrouillé, il le reste lorsque l'alimentation est coupée et il est déverrouillé lorsque le relais de déverrouillage est activé. Pour de plus amples informations

Copyright © 2015 Dunod.

concernant ces relais, consultez la Section 7.5 qui les présente dans le contexte des bobines d'enclenchement et de déclenchement.

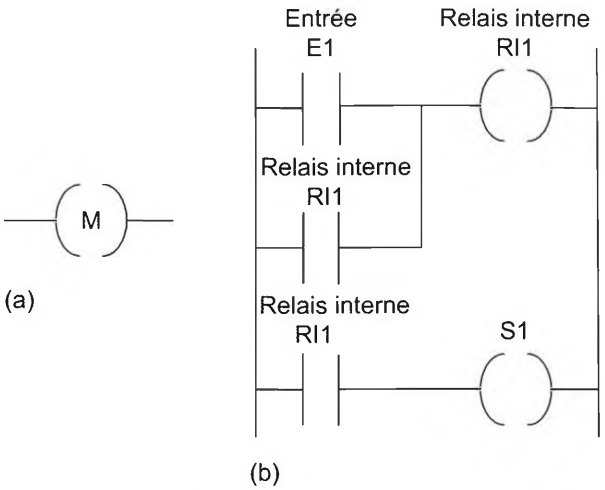


Figure 7.8 – (a) Bobine de mémoire rémanente et (b) schéma à relais sauvegardé par batterie.

La Figure 7.8b illustre l'utilisation d'un tel relais dans le schéma à contacts d'un système conçu pour faire face aux pannes d'alimentation. RI1 est un relais interne sauvegardé. Lorsque les contacts de l'entrée E1 se ferment, la sortie RI1 est excitée. Les contacts de RI1 se ferment alors, avec un verrouillage qui permet à RI1 de rester excité même si l'entrée E1 s'ouvre. Le résultat est une activation de la sortie S1. En cas de panne d'alimentation, RI1 reste excité. Ses contacts restent donc fermés et la sortie sur S1 est toujours présente.

7.4 Monostable

Certains fabricants d'API permettent de programmer un relais interne afin que ses contacts soient activés pendant un seul cycle, c'est-à-dire pendant une seule analyse du programme. Lorsqu'il est activé, il fournit ainsi une impulsion de durée fixe sur ses contacts. Cette fonction est souvent appelée *monostable*. Bien que certains API proposent cette fonction sous forme d'entité dans leurs programmes, elle peut facilement être mise en œuvre par un schéma à contacts de deux lignes (voir Figure 7.9).

À la Figure 7.9a, lorsque l'entrée de déclenchement est activée, elle produit une sortie de déclenchement sur la ligne 1. Sur la ligne 2, elle produit une sortie de contrôle de cycle sur un relais interne. Puisque la ligne 2 est traitée après la ligne 1, l'effet du contrôle de cycle n'est visible qu'au prochain cycle du programme de l'API, lorsqu'il ouvre les contacts du contrôle de cycle dans la ligne 1 et stoppe la sortie de déclenchement. La sortie de déclenchement reste au niveau bas malgré la présence de l'entrée de déclenchement. La sortie de déclenchement peut être renouvelée uniquement si elle est désactivée puis de nouveau activée.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

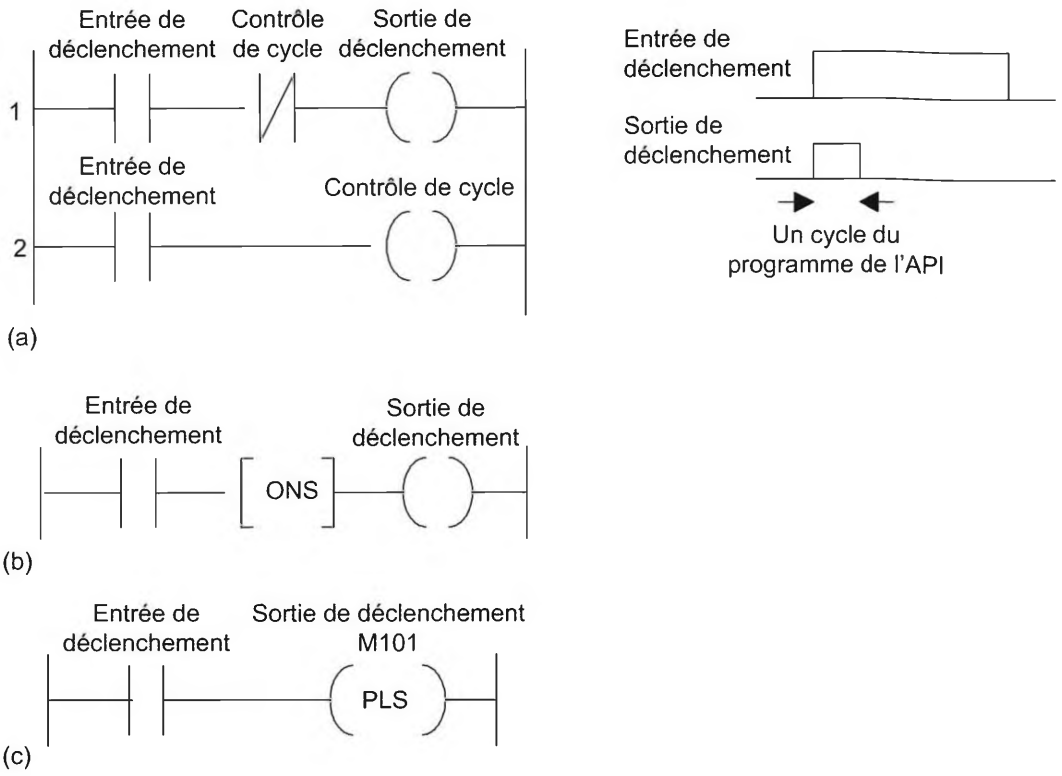


Figure 7.9 – Monostable (a) par programme, (b) comme fonction dans un API d'Allen-Bradley et (c) comme fonction dans un API de Mitsubishi.

Les Figures 7.9b et 7.9c présentent les fonctions intégrées dans les API d'Allen-Bradley et de Mitsubishi. Pour l'API de Mitsubishi (voir Figure 7.9c), le relais interne de sortie, disons M100, est activé lorsque les contacts de l'entrée de déclenchement, disons X400, se ferment. Dans des circonstances normales, M100 reste actif tant que les contacts X400 sont fermés. Toutefois, si M100 a été programmé pour un fonctionnement impulsionnel, il reste actif uniquement pendant une durée fixée (un cycle de programme). Ensuite, il redevient inactif, quel que soit l'état de X400. Les instructions correspondantes sont LD X400, PLS M100. Le fonctionnement impulsionnel précédent existe lorsque l'entrée passe du niveau bas au niveau haut, c'est-à-dire lors d'un front montant. Si, à la Figure 7.9c, l'entrée de déclenchement est normalement fermée au lieu d'être normalement ouverte, l'impulsion se produit lorsque l'entrée passe du niveau haut au niveau bas, c'est-à-dire lors d'un front descendant.

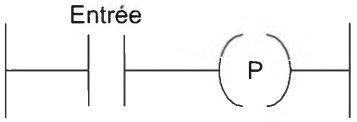
La norme CEI 61131-3 définit des symboles standard pour les bobines à détection de front montant (positif) et à détection de front descendant (négatif). Ils sont donnés à la Figure 7.10.

Avec la bobine à détection de front montant, si la tension d'alimentation passe du niveau bas au niveau haut, la sortie est activée pendant une évaluation de la ligne du schéma. Avec la bobine à détection de front descendant, si la tension d'alimentation passe du niveau haut au niveau bas, la sortie est activée pendant une évaluation de la ligne du schéma. Ainsi, pour la ligne du schéma à contacts de la

Copyright © 2015 Dunod.

Figure 7.11, lorsque l'entrée est inactive, la sortie est absente. Lorsque l'entrée passe au niveau haut, une sortie est présente sur la bobine. Toutefois, les cycles suivants du programme ne produisent pas de sortie sur la bobine même si l'entrée reste au niveau haut. La sortie sur la bobine existe uniquement lors de la première activation de l'entrée.

Figure 7.10 – (a) Bobine à détection de front montant et (b) bobine à détection de front descendant.



Évaluation	Entrée	Sortie P
1	Bas	Bas
2	Haut	Haut
3	Haut	Bas
4	Haut	Bas

Figure 7.11 – Ligne d'un schéma à contacts avec une bobine à détection de front montant.

7.5 Mise à un et mise à zéro

En général, les fabricants d'API proposent également une fonction de mise à un et de mise à zéro d'un relais interne. L'instruction de mise à un provoque l'enclenchement du relais, c'est-à-dire son verrouillage. Il reste dans cet état jusqu'à ce que l'instruction de mise à zéro soit reçue et provoque son déclenchement, c'est-à-dire son déverrouillage. Le terme *bascule* est souvent employé pour décrire ce fonctionnement. La Figure 7.12 montre les symboles définis par la CEI 61131-3 pour représenter ces bobines. La bobine d'enclenchement (SET) est activée lorsqu'elle est alimentée et reste active jusqu'à ce qu'elle soit mise à zéro. La bobine de déclenchement (RESET) est placée dans l'état inactif lorsqu'elle est alimentée et reste ainsi jusqu'à ce qu'elle soit mise à un.

La Figure 7.13 présente un schéma à contacts qui comprend une telle fonction. L'activation de la première entrée, X400, provoque l'activation de la sortie Y430 et sa mise à un, c'est-à-dire son verrouillage. Par conséquent, si la première entrée est désactivée, la sortie reste active. L'activation de la seconde entrée, X401, provoque la mise à zéro de la sortie Y430, c'est-à-dire sa désactivation et son déverrouillage. Par conséquent, la sortie Y430 reste active pendant le moment où X400 est momentanément activée et celui où X401 est momentanément activée. Entre les deux lignes qui incluent les opérations de mise à un et de mise à zéro, d'autres lignes peuvent procéder à d'autres opérations, avec la ligne de mise à un qui active une sortie au début de la séquence et la désactive à la fin.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

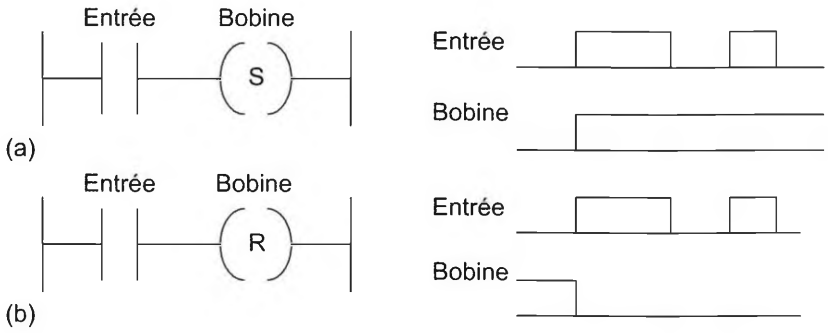


Figure 7.12 – Bobines (a) d'enclenchement et (b) de déclenchement.

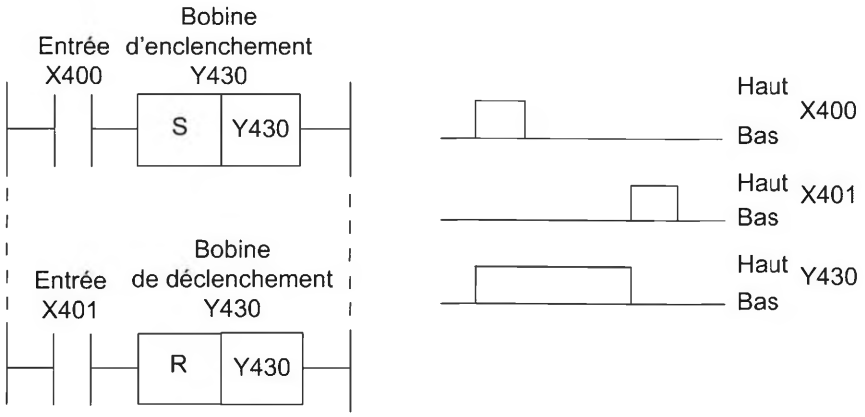


Figure 7.13 – Mise à un et mise à zéro.

Voici les instructions qui correspondent au schéma à contacts de la Figure 7.13 :

```
LD      X400
S       Y430
```

et :

```
LD      X401
R       Y430
```

La Figure 7.14 présente le schéma à contacts équivalent pour un API de Telemecanique. Les instructions deviennent alors les suivantes :

```
L       I0,0
S       00,0
L       I0,1
R       00,0
```

Pour un API d'Allen-Bradley, les termes utilisés sont *verrouillage* et *déverrouillage*. La Figure 7.15 montre le schéma à contacts correspondant.

Les symboles des bobines d'enclenchement et de déclenchement sont souvent combinés dans un symbole unique. La Figure 7.16 présente le schéma à contacts qui équivaut à la fonction mise à un/mise à zéro des figures précédentes pour un API de Siemens. Le terme *bloc bistable* remplace le bloc mise à un/mise à zéro et il est libellé SR, ou mémoire avec mise à zéro prioritaire, car la mise à zéro est prioritaire.

Copyright © 2015 Dunod.

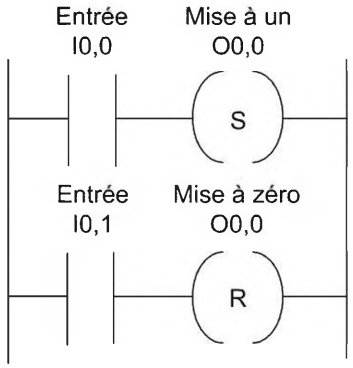


Figure 7.14 – Mise à un et mise à zéro (API de Telemecanique).

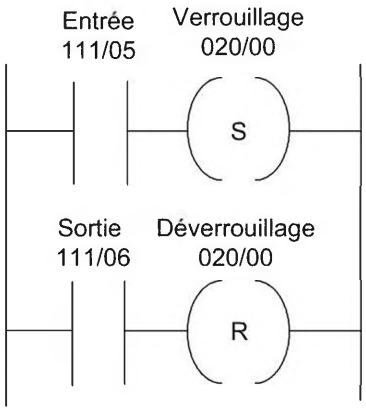


Figure 7.15 – Verrouillage et déverrouillage (API d'Allen-Bradley).

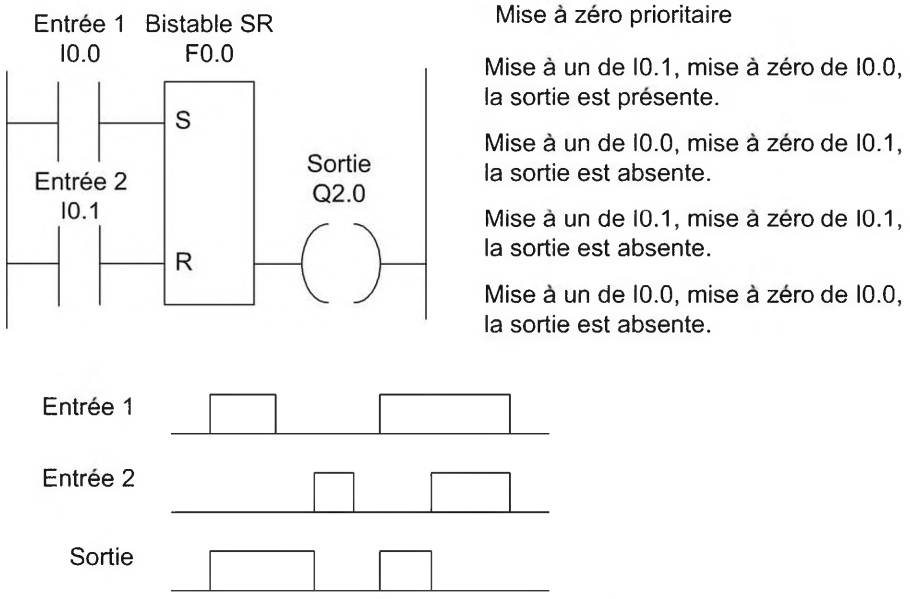


Figure 7.16 – Mise à un et mise à zéro, avec mise à zéro prioritaire (API de Siemens).

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

Avec la mise à un prioritaire (bloc bistable RS), le schéma devient celui de la Figure 7.17. Voici les instructions (F indique un relais interne) pour la mise à zéro prioritaire :

```
A       I0.0
S       F0.0
A       I0.1
R       F0.0
A       F0.0
=       Q2.0
```

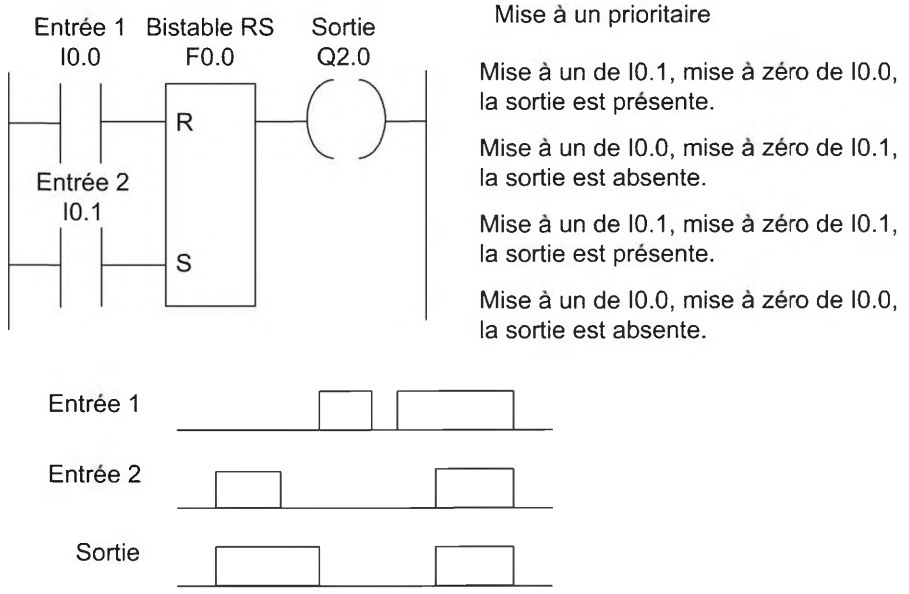


Figure 7.17 – Mise à un et mise à zéro, avec mise à un prioritaire (API de Siemens).

Toshiba emploie le terme *bascule*. La Figure 7.18 présente cette version du schéma à contacts.

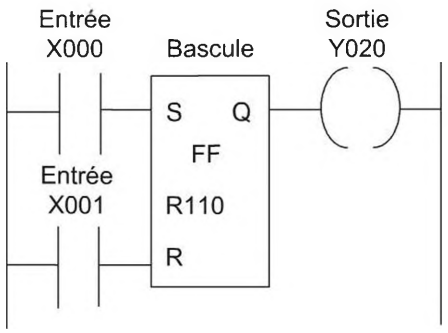


Figure 7.18 – Bascule (API de Toshiba).

La Figure 7.19 montre comment utiliser la fonction de mise à un et de mise à zéro pour obtenir le fonctionnement impulsionnel (monostable) décrit à la section

Copyright © 2015 Dunod.

précédente. La Figure 7.19a correspond à la version pour un API de Siemens (F désigne un relais interne) et la Figure 7.19b, à un API de Telemecanique (B désigne un relais interne). Dans les Figures 7.19a et 7.19b, une entrée (I0.0, I0,0) déclenche l'activation du relais interne (F0.0, B0) de la première ligne. Cela conduit à la mise à un du relais interne mise à un/mise à zéro de la deuxième ligne. Cette opération déclenche l'ouverture du relais interne (F0.1, B1) de la première ligne et, malgré la présence d'une entrée sur la première ligne, le relais interne (F0.0, B0) s'ouvre. Toutefois, puisque les lignes sont analysées de haut en bas, il faut un cycle complet avant que le relais interne de la première ligne ne s'ouvre. Une impulsion d'une durée d'un cycle est donc produite. Le système est mis à zéro lorsque l'entrée (I0.0, I0,0) disparaît.

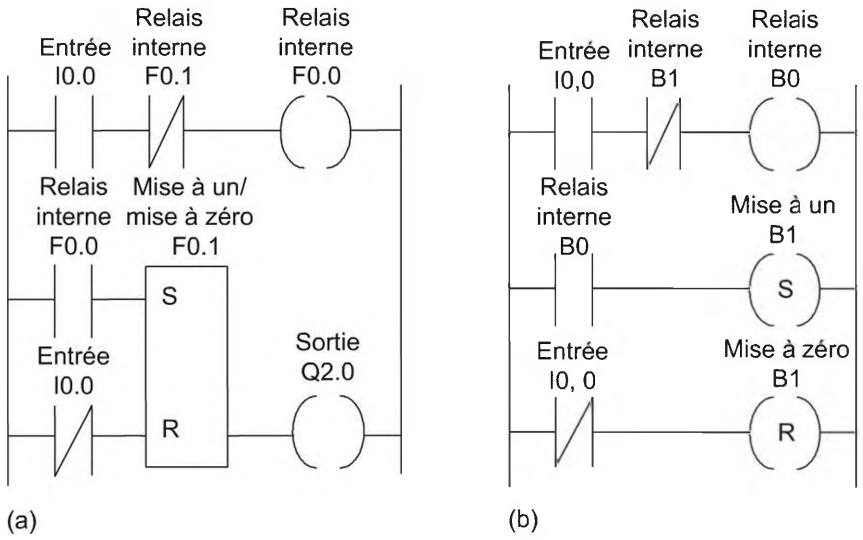


Figure 7.19 – Fonctionnement impulsionnel : (a) API de Siemens et (b) API de Telemecanique.

7.5.1 Exemples de programmes

Les éléments de base d'un programme simple pour un système d'alarme incendie sont présentés à la Figure 7.20. Trois détecteurs sont en entrée d'un bloc fonctionnel de mise à un/mise à zéro afin que l'activation de l'un d'eux mette à un l'alarme, qui reste dans cet état jusqu'à ce qu'elle soit arrêtée par une mise à zéro *via* un interrupteur.

La Figure 7.21 présente un autre exemple qui utilise des éléments de base d'un programme. Il pourrait être utilisé dans un système qui détecte le bon positionnement d'une pièce en vue de son usinage. Lorsque les contacts de mise en marche sont fermés, la sortie provoque le déplacement de la pièce. Il se poursuit jusqu'à ce qu'un faisceau lumineux soit interrompu. À ce moment-là, le déplacement de la pièce est mis à zéro et la sortie est désactivée. Un bouton d'arrêt permet de stopper le déplacement à tout moment.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

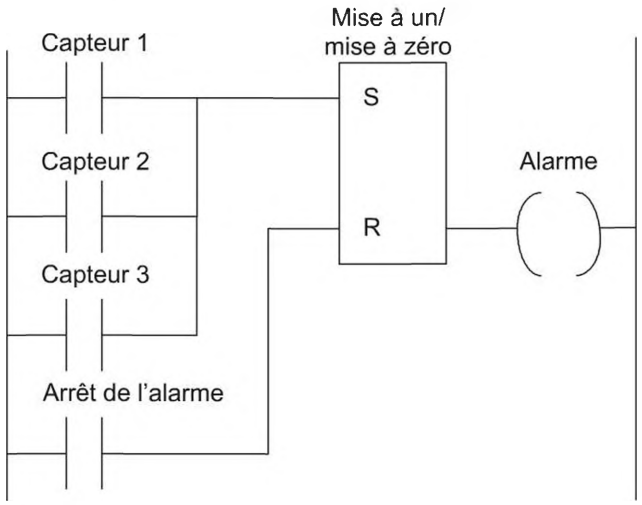


Figure 7.20 – Système d'alarme incendie.

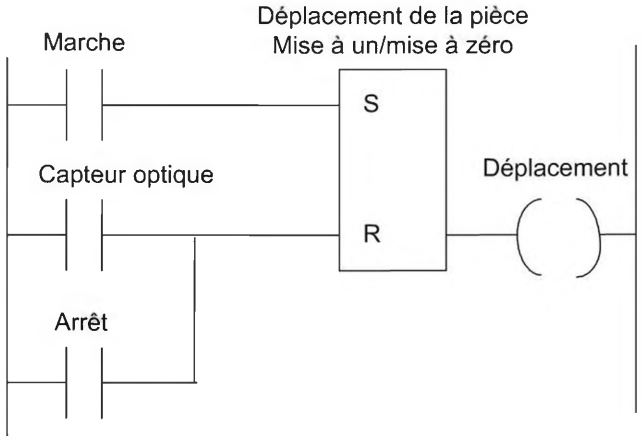


Figure 7.21 – Système de chargement.

7.6 Relais de contrôle maître

Lorsque le nombre de sorties à contrôler est important, il est parfois nécessaire que des sections entières d'un schéma à contacts soient activées ou désactivées en fonction de certaines conditions. Ce fonctionnement peut être obtenu en incluant les contacts du même relais interne dans chacune des lignes afin que sa commande les affecte toutes. Une autre solution se fonde sur le *relais de contrôle maître* (MCR, *Master Control Relay*). La Figure 7.22 illustre l'emploi d'un tel relais pour contrôler une section d'un schéma à contacts.

En l'absence d'entrée sur E1, le relais interne de sortie MC1 n'est pas excité et ses contacts sont donc ouverts. Cela signifie que toutes les lignes placées entre celle où il est actionné et celle sur laquelle se trouve son relais de remise à zéro, ou un autre relais de contrôle maître, sont désactivées. Si l'on suppose qu'il est conçu pour fonctionner à partir de sa propre ligne, nous pouvons imaginer qu'il est situé sur la ligne d'alimentation à l'emplacement indiqué. Les lignes 2 et 3 sont donc désactivées. Lorsque les contacts de l'entrée E1 se ferment, le relais maître MC1 est

Copyright © 2015 Dunod.

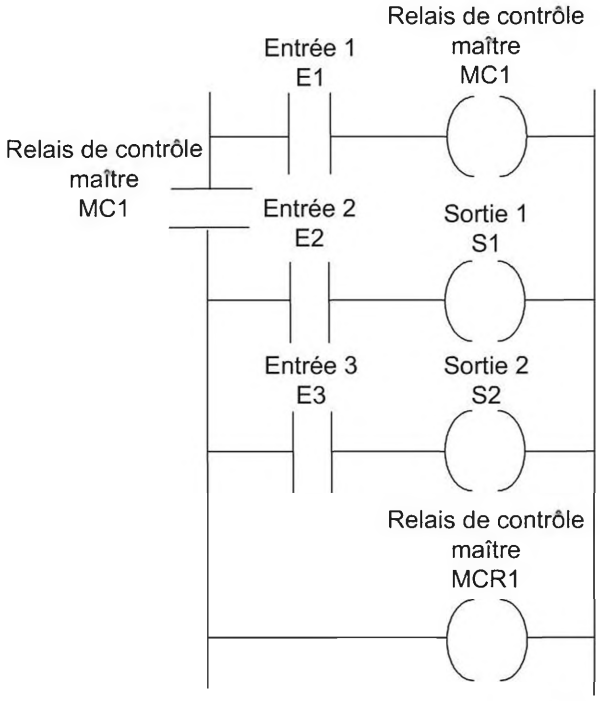


Figure 7.22 – Principe d'utilisation d'un relais de contrôle maître.

excité. Dans ce cas, toutes les lignes qui se trouvent entre sa ligne et celle sur laquelle se trouve MCR1 sont activées. Par conséquent, les sorties S1 et S2 ne peuvent pas être activées tant que le relais de contrôle maître n'est pas excité. Le relais MC1 affecte uniquement la section placée entre la ligne où il est actionné et la ligne sur laquelle se trouve MCR1.

Dans un API de Mitsubishi, un relais interne peut devenir un relais de contrôle maître par simple programmation. Voici l'instruction qui permet de programmer un relais interne M100 en relais de contrôle maître :

MC M100

Pour programmer la remise à zéro de ce relais, voici l'instruction à utiliser :

MCR M100

Les instructions suivantes correspondent au schéma à contacts de la Figure 7.23, qui équivaut à celui de la Figure 7.22 avec des adresses au format Mitsubishi :

```
LD      X400
OUT     M100
MC      M100
LD      X401
OUT     Y430
LD      X402
OUT     Y431
MC      M100
```

La Figure 7.24 illustre le format utilisé par Allen-Bradley. Pour terminer le contrôle d'un relais de contrôle maître, un second relais de contrôle maître est utilisé, sans qu'il dépende de contacts ou d'une logique. Il est programmé de manière inconditionnelle.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

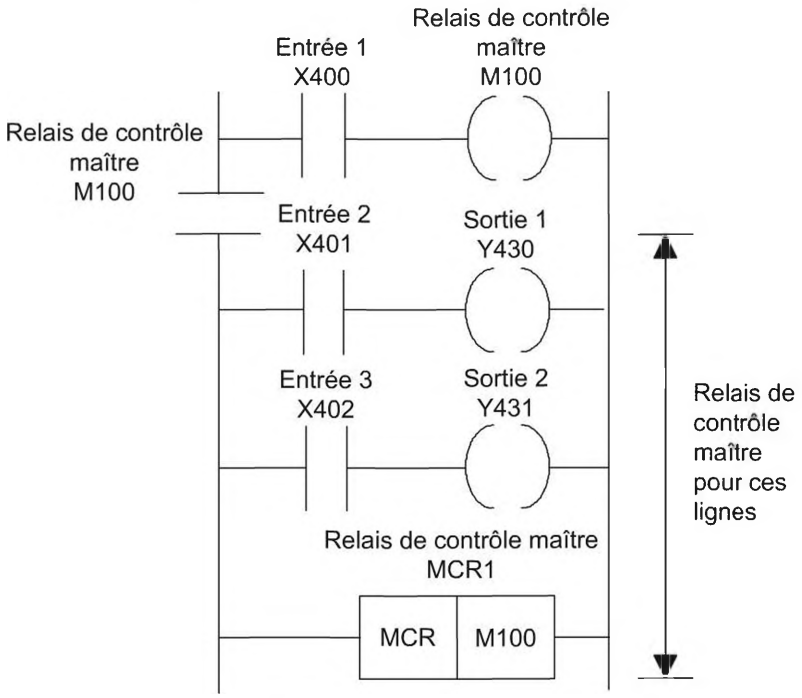


Figure 7.23 – Relais de contrôle maître avec un API de Mitsubishi.

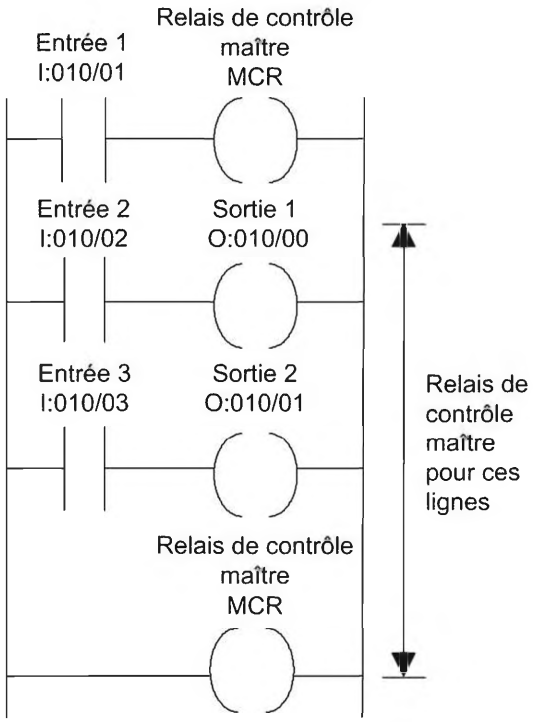


Figure 7.24 – Relais de contrôle maître avec un API d'Allen-Bradley.

La Figure 7.25 illustre la représentation des relais de contrôle maître, ou relais de masquage, dans le schéma à contacts pour les API de Siemens. La section dans laquelle un MCR doit opérer est définie par les fonctions d'activation et de désactivation du relais de masquage. Au sein de cette zone, le MCR est activé lorsque la bobine MCR> est activée, et désactivé lorsque la bobine MCR< est activée.

Copyright © 2015 Dunod.

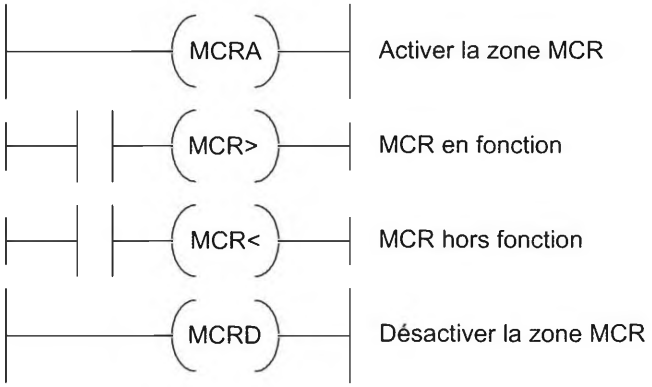


Figure 7.25 – Représentation des relais de contrôle maître selon Siemens.

Un programme peut employer plusieurs MCR, permettant à différentes sections d'un schéma à contacts d'être activées ou désactivées. La Figure 7.26 présente un schéma à contacts au format Mitsubishi qui met en œuvre deux MCR. Lorsque M100 est en fonction et M101, hors fonction, la séquence des lignes est 1, 3, 4, 6, etc. La fin de la section contrôlée par M100 est indiquée par l'autre relais de contrôle maître, M101. Lorsque M101 est activé et M100 désactivé, la séquence des lignes devient 2, 4, 5, 6, etc. La fin de cette section est indiquée par la présence de la mise à zéro. Cette mise à zéro est utilisée car la ligne n'est pas immédiatement suivie d'un autre MCR. Un tel schéma peut être utilisé pour activer un ensemble de lignes différent selon le type d'entrée qui se produit.

7.6.1 Exemple de programme

Dans cette section, nous étudions un programme qui illustre l'utilisation des relais de contrôle maître. Le programme est développé pour un système de vannes pneumatiques impliquant des vérins de manière à obtenir un ordre particulier de déplacement des pistons. Toutefois, nous commençons par examiner l'utilisation du verrouillage dans un tel système pour maintenir les actions.

Prenons un système pneumatique avec des vannes monosolénoïdes et deux verrins, A et B. Les interrupteurs de fin de course a–, a+, b– et b+ définissent les limites de mouvement de la tige du vérin (voir Figure 7.27), dont le déplacement doit se faire dans l'ordre A+, B+, A–, B–. La Figure 7.28 présente le schéma à contacts que nous pouvons utiliser.

L'électrovanne A+ est alimentée lorsque le bouton de mise en marche et l'interrupteur de fin de course b– sont fermés. Cela permet d'obtenir le verrouillage de A+ afin qu'elle reste excitée tant que les contacts normalement fermés du fin de course b+ ne sont pas activés. Lorsque l'interrupteur de fin de course a+ est activé, l'électrovanne B+ est alimentée. Cela permet d'obtenir le verrouillage de B+ afin qu'elle reste excitée tant que les contacts normalement fermés du fin de course a– ne sont pas activés. Lorsque le piston B sort, le fin de course b+ ouvre ses contacts normalement fermés et déverrouille l'électrovanne A+. Le piston A se rétracte alors. Lorsqu'il est rentré et a ouvert les contacts normalement fermés a–, l'électrovanne B+ est déverrouillée et le piston B se rétracte.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

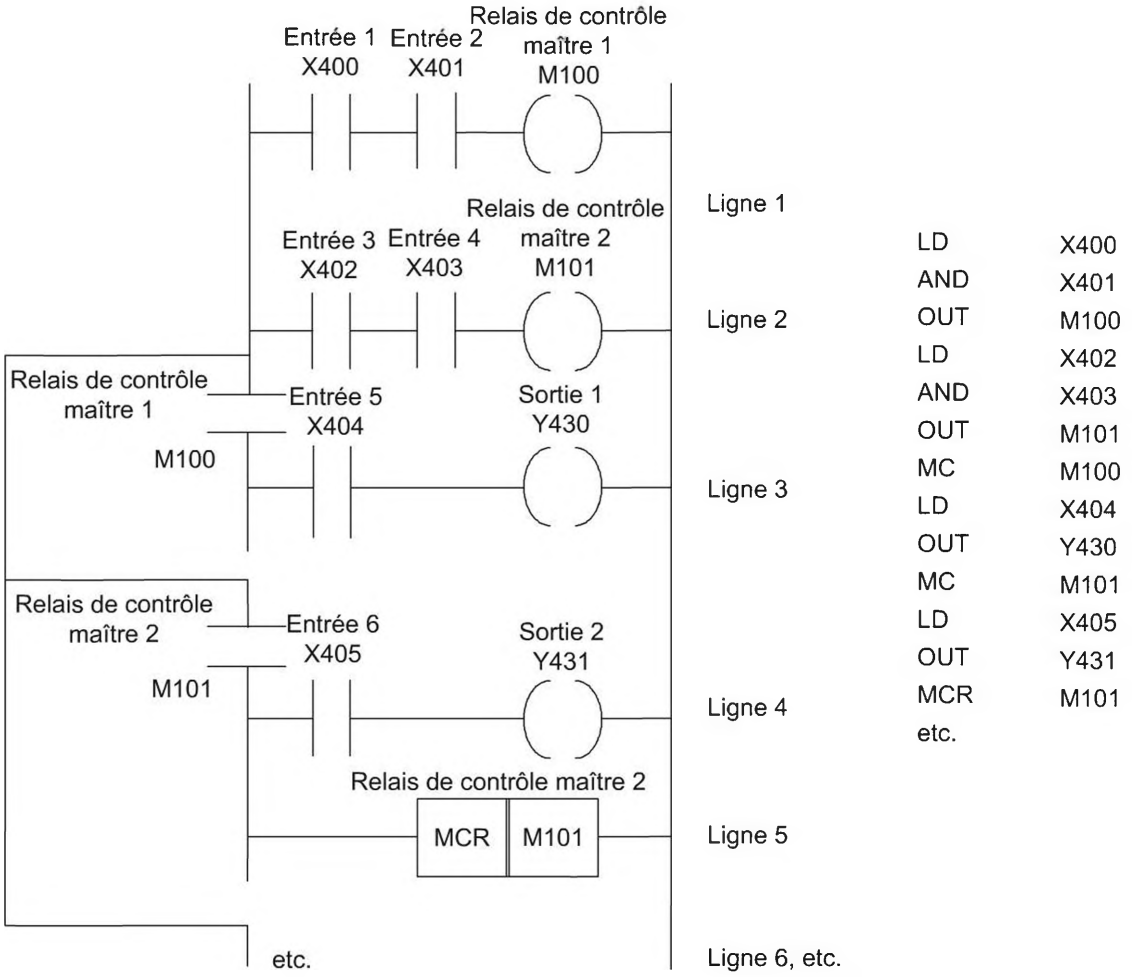


Figure 7.26 – Utilisation de plusieurs relais de contrôle maître.

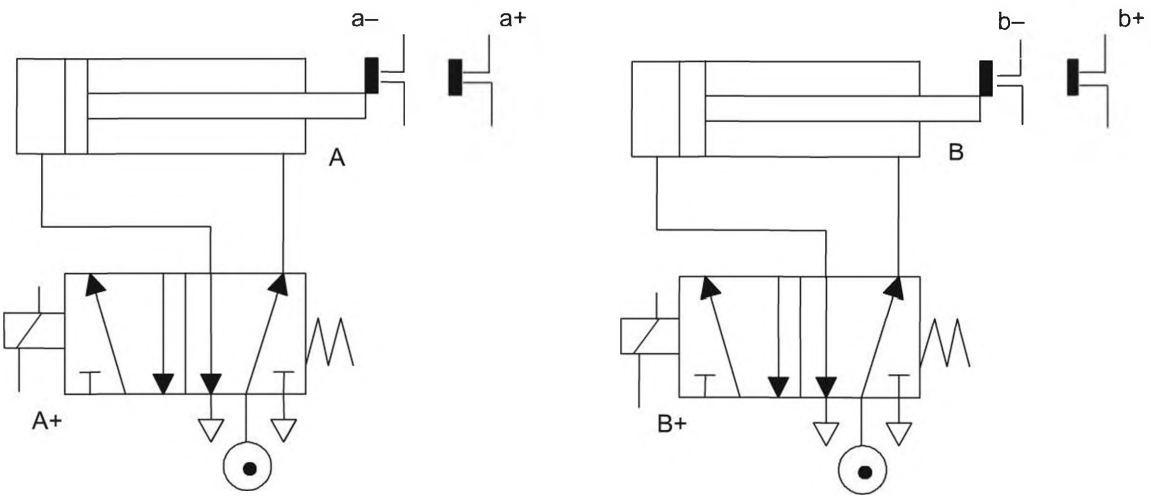


Figure 7.27 – Système de vannes.

Examinons à présent le schéma à contacts que nous pouvons utiliser avec les deux vannes monosolénoïdes de la Figure 7.27 pour obtenir, lorsque et uniquement lorsque l'interrupteur de mise en marche est momentanément activé, la séquence

Copyright © 2015 Dunod.

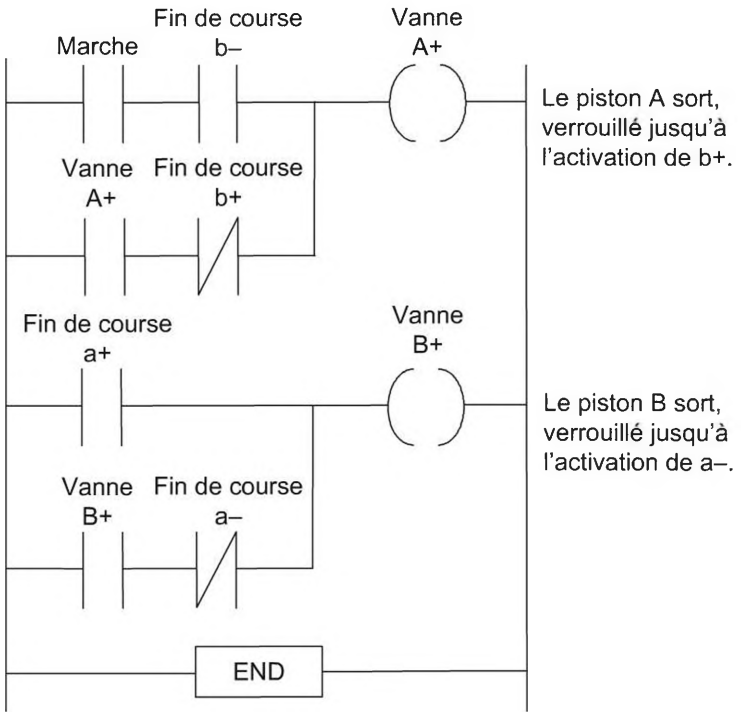


Figure 7.28 – Schéma à contacts.

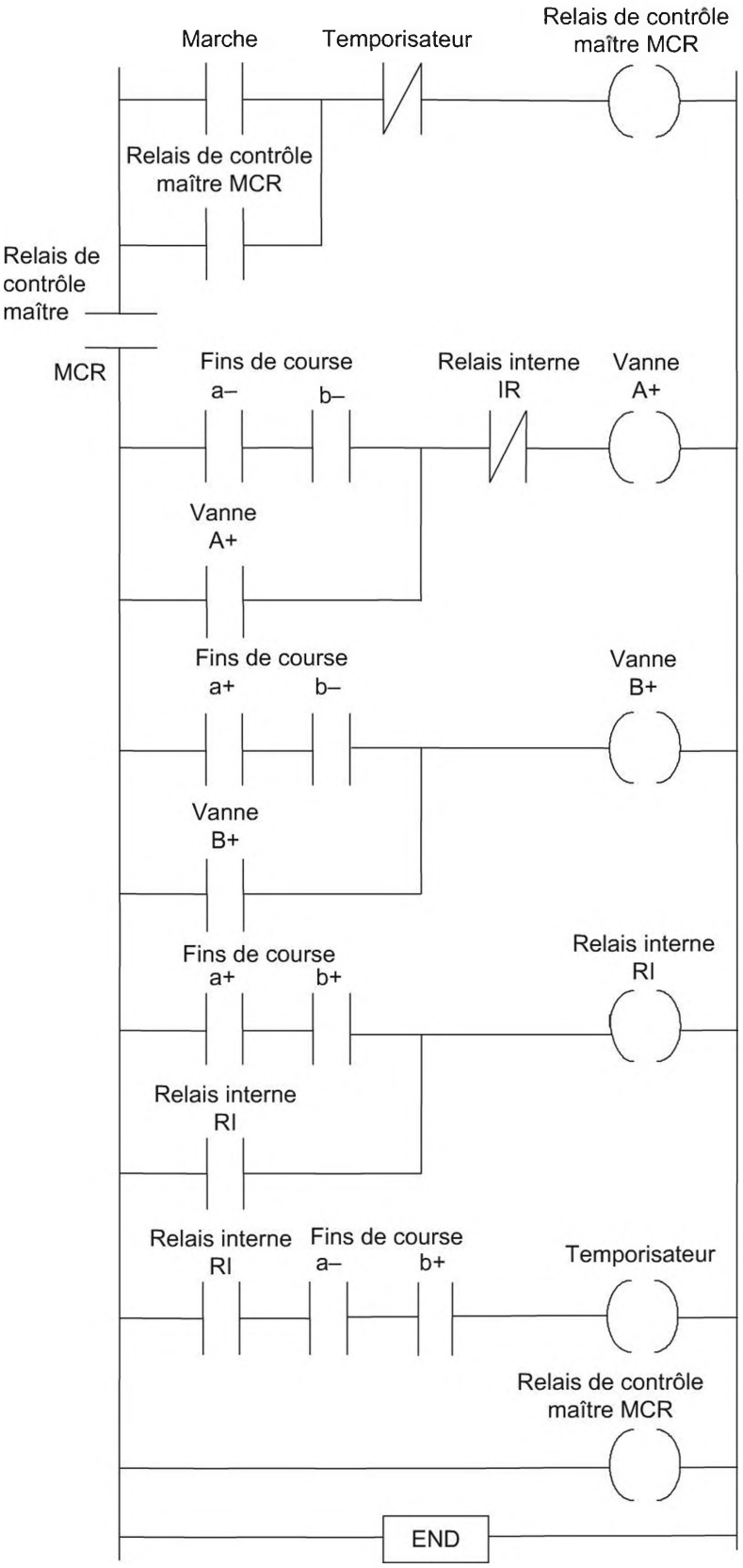
A+, B+, A–, une pause de 10 s, B– et l'arrêt en ce point jusqu'au nouvel appui sur l'interrupteur de mise en marche. La Figure 7.29 présente le programme correspondant fondé sur un relais de contrôle maître. Le MCR est activé par l'interrupteur de mise en marche et reste actif jusqu'à ce qu'il soit coupé par la ligne qui contient uniquement MCR. Pour de plus amples informations sur les temporisateurs, consultez le Chapitre 9.

7.7 En résumé

Dans les API, certains éléments sont utilisés pour conserver des données, c'est-à-dire des bits, et se comportent comme des relais, capables d'allumer ou d'éteindre d'autres dispositifs. Ils sont appelés relais internes, relais auxiliaires, marqueurs, fanions ou éléments de stockage binaire. Dans un schéma à contacts, la sortie d'un relais interne est représentée à l'aide du symbole d'un dispositif de sortie, ( ), et une adresse qui indique sa nature de relais interne. Les contacts du relais interne sont représentés par le symbole d'un dispositif d'entrée, | |, et prennent la même adresse que la sortie du relais interne. Les relais internes sauvegardés par batterie sont en mesure de conserver leur état même lorsque l'alimentation est coupée. Ils sont dits à rémanence.

Certains fabricants d'API permettent de programmer un relais interne afin que ses contacts soient activés pendant un seul cycle. Cette fonction est souvent appelée monostable. Les fabricants proposent également une fonction de mise à un et de mise à zéro d'un relais interne. Le terme bascule est souvent employé pour décrire ce fonctionnement.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.



L'interrupteur de mise en marche excite le relais de contrôle maître et verrouille la mise en marche. Quel que soit l'état des interrupteurs de fin de course, le système ne peut pas démarrer tant que le MCR n'est pas excité.

Le relais de contrôle maître place les lignes suivantes sous tension.

Les deux vérins étant rétractés, le piston A sort, verrouillant les interrupteurs de fin de course.

A étant sorti et B rétracté, le piston B sort, verrouillant les interrupteurs de fin de course.

A et B étant sortis, le relais interne est excité. Cela met l'électrovanne A+ hors tension et le piston A se rétracte.

Lorsque A est rétracté, le piston B sort toujours, activant le relais interne de déclenchement du temporisateur. Lorsque la temporisation est écoulée, le système est coupé et B se rétracte.

Figure 7.29 – Schéma à contacts.

Copyright © 2015 Dunod.

Un relais de contrôle maître est capable de mettre sous ou hors tension une section d'un schéma à contacts, jusqu'à ce qu'il soit remis à zéro.

7.8 Problèmes

Pour les questions 1 à 23, quatre réponses sont proposées : A, B, C et D. Choisissez la réponse correcte. Les problèmes 1 à 3 font référence à la Figure 7.30, qui présente un schéma à contacts comprenant un relais interne (RI1), deux entrées (E1 et E2) et une sortie (S1).

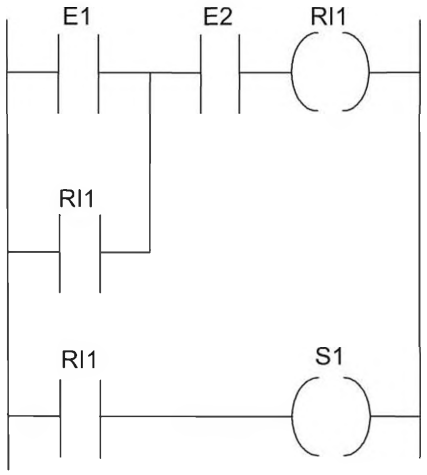


Figure 7.30 – Diagramme pour les problèmes 1, 2 et 3.

1. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Pour le schéma à contacts de la Figure 7.30, la sortie S1 est présente :
   (i) lorsqu'il existe une entrée seulement sur E1 ;
   (ii) lorsqu'il existe une entrée seulement sur E2 ;

   Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
   A. (i) V (ii) V
   B. (i) V (ii) F
   C. (i) F (ii) V
   D. (i) F (ii) F
2. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Pour le schéma à contacts de la Figure 7.30, la sortie S1 est présente :
   (i) lorsqu'il existe une entrée sur E2 et une entrée momentanée sur E1 ;
   (ii) lorsqu'il existe une entrée sur E1 ou sur E2.

   Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
   A. (i) V (ii) V
   B. (i) V (ii) F
   C. (i) F (ii) V
   D. (i) F (ii) F

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

3. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Pour le schéma à contacts de la Figure 7.30, les relais internes sont alimentés :
   (i) lorsqu'il existe une entrée uniquement sur E1 ;
   (ii) lorsqu'il existe une entrée sur E1 et sur E2.

   Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
   A. (i) V (ii) V
   B. (i) V (ii) F
   C. (i) F (ii) V
   D. (i) F (ii) F

Les problèmes 4 à 6 font référence à la Figure 7.31, qui présente un schéma à contacts comprenant deux relais internes (RI1 et RI2), quatre entrées (E1, E2, E3 et E4) et une sortie (S1).

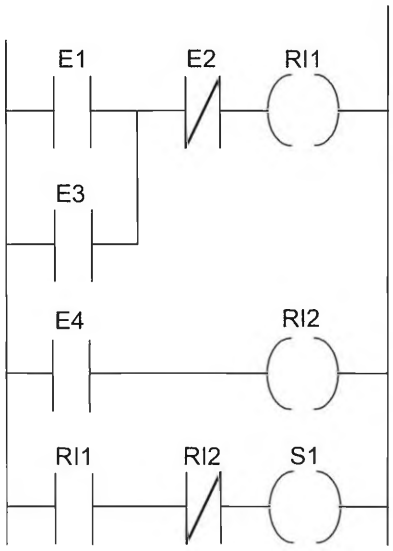


Figure 7.31 – Diagramme pour les problèmes 4, 5 et 6.

4. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Pour le schéma à contacts de la Figure 7.31, le relais interne RI1 est alimenté :
   (i) lorsqu'il existe une entrée sur E1 ;
   (ii) lorsqu'il existe une entrée sur E3.

   Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
   A. (i) V (ii) V
   B. (i) V (ii) F
   C. (i) F (ii) V
   D. (i) F (ii) F
5. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Pour le schéma à contacts de la Figure 7.31, le relais interne RI3 est alimenté :
   (i) lorsque le relais interne RI1 est excité ;
   (ii) lorsqu'il existe une entrée sur E4.

Copyright © 2015 Dunod.

Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?

A. (i) V (ii) V
B. (i) V (ii) F
C. (i) F (ii) V
D. (i) F (ii) F

6. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Pour le schéma à contacts de la Figure 7.31, la sortie S1 est présente :
   (i) lorsqu'il existe une entrée uniquement sur E1, E2 et E4 ;
   (ii) lorsqu'il existe une entrée uniquement sur E3 et E4.

   Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?

   A. (i) V (ii) V
   B. (i) V (ii) F
   C. (i) F (ii) V
   D. (i) F (ii) F

7. Quel programme de la Figure 7.32 permet d'obtenir une sortie sur S1 lorsqu'il existe une entrée uniquement sur E1 ?

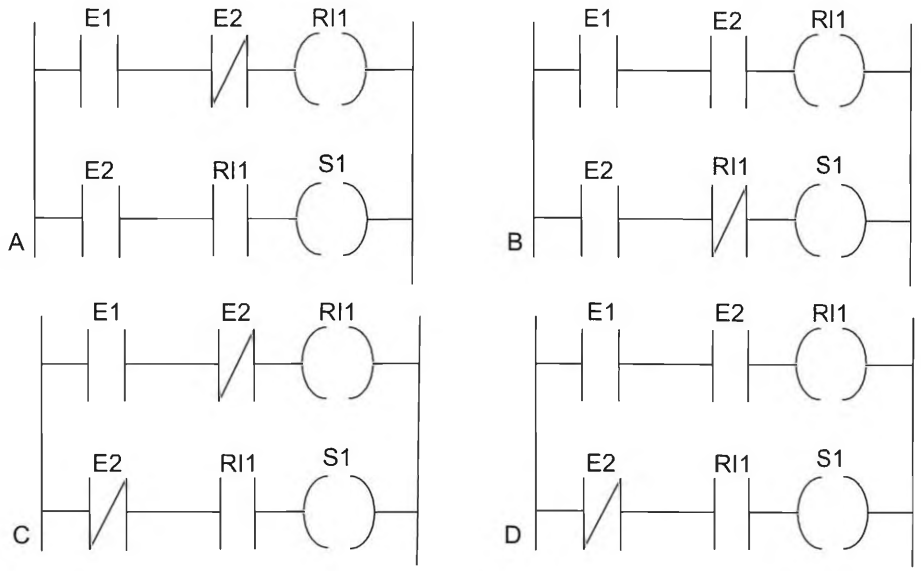


Figure 7.32 – Diagramme pour le problème 7.

8. Quel programme de la Figure 7.33 permet d'obtenir une sortie sur S1 au cours de la même analyse du programme lorsqu'il existe une entrée sur E1 ?

Les problèmes 9 et 10 font référence à la Figure 7.34, qui présente un schéma à contacts comprenant un relais sauvegardé par batterie (RI1), deux entrées (E1 et E2) et une sortie (S1).

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

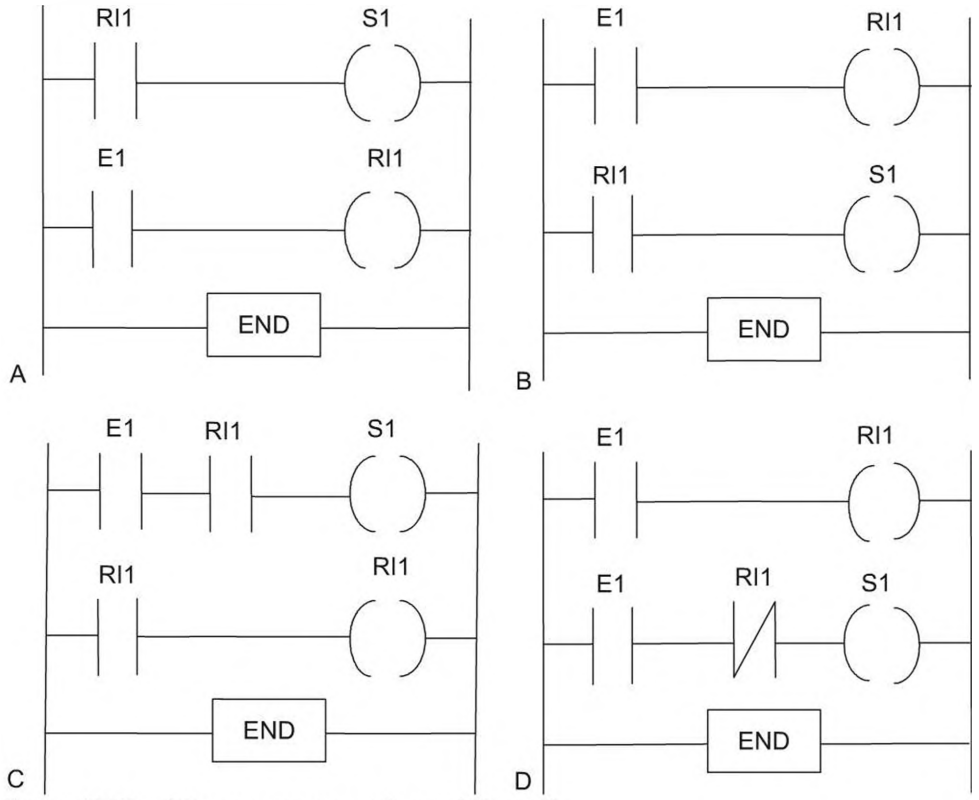


Figure 7.33 – Diagramme pour le problème 8.

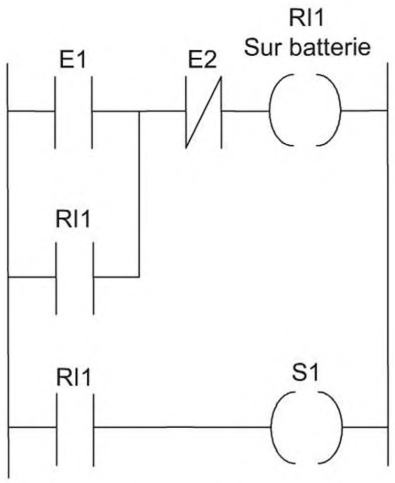


Figure 7.34 – Diagramme pour les problèmes 9 et 10.

9. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Pour le schéma à contacts de la Figure 7.34, la sortie S1 est présente :

   (i) lorsqu'il existe une entrée de courte durée sur E1 ;

   (ii) lorsqu'il n'existe pas d'entrée sur E2.

   Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?

   A. (i) V (ii) V

   B. (i) V (ii) F

Copyright © 2015 Dunod.

C. (i) F (ii) V
D. (i) F (ii) F

10. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Pour le schéma à contacts de la Figure 7.34 :
    (i) l'entrée E1 est verrouillée par le relais interne afin que RI1 reste excité même lorsque l'entrée sur E1 disparaît ;
    (ii) puisque le relais interne RI1 est sauvegardé par batterie, la sortie sur S1 reste présente même lorsque que l'alimentation est coupée, jusqu'à l'arrivée d'une entrée sur E2.

    Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
    A. (i) V (ii) V
    B. (i) V (ii) F
    C. (i) F (ii) V
    D. (i) F (ii) F

11. Lorsque les instructions LD X100, PLS M400 sont utilisées sur une ligne d'un schéma à contacts, le relais interne M400 :
    A. Reste actif même lorsque l'entrée sur X100 cesse
    B. Reste fermé à moins d'une entrée impulsionnelle sur X100
    C. Reste actif pendant un cycle du programme lorsque l'entrée sur X100 existe
    D. Reste fermé pendant un cycle du programme après une entrée sur X100

12. Lorsque les instructions LDI X100, PLS M400 sont utilisées sur une ligne d'un schéma à contacts, le relais interne M400 :
    A. Reste actif même lorsque l'entrée sur X100 cesse
    B. Reste actif lorsqu'il existe une entrée impulsionnelle sur X100
    C. Reste actif pendant un cycle du programme lorsque l'entrée sur X100 existe
    D. Reste actif pendant un cycle du programme après disparition de l'entrée sur X100

13. Un programme en langage à contacts pour un API de Mitsubishi contient les instructions LD X100, S M200, LD X101, R M200, suivies d'autres instructions pour les lignes suivantes. La séquence suivante se produit : une entrée sur l'entrée X100, l'entrée sur X100 cesse, un certain temps s'écoule, une entrée sur l'entrée X101, l'entrée sur X101 cesse, des entrées sur les lignes suivantes. Le relais interne M200 restera actif :
    A. Pendant un cycle du programme à partir du début de l'entrée sur X100
    B. Depuis le début de l'entrée sur X100 jusqu'au début de l'entrée sur X101
    C. Depuis le début de l'entrée sur X100 jusqu'à la fin de l'entrée sur X101
    D. Depuis la fin de l'entrée sur X100 jusqu'à la fin de l'entrée sur X101

14. Un programme en langage à contacts pour un API de Siemens contient les instructions A I0.0, S F0.0, A I0.1, R F0.0, A F0.0, = Q2.0, suivies d'autres instructions pour les lignes suivantes. La séquence suivante se produit : une entrée sur l'entrée I0.0, l'entrée sur I0.0 cesse, un certain temps s'écoule, une entrée sur l'entrée I0.1, l'entrée sur I0.1 cesse, des entrées sur les lignes suivantes. Le relais interne F0.0 restera actif :

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

A. Pendant un cycle du programme à partir du début de l'entrée sur I0.0
B. Depuis le début de l'entrée sur I0.0 jusqu'au début de l'entrée sur I0.1
C. Depuis le début de l'entrée sur I0.0 jusqu'à la fin de l'entrée sur I0.1
D. Depuis la fin de l'entrée sur I0.0 jusqu'à la fin de l'entrée sur I0.1

15. Un programme en langage à contacts pour un API de Telemecanique contient les instructions L I0,0, S O0,0, L I0,1, R O0,0, suivies d'autres instructions pour les lignes suivantes. La séquence suivante se produit : une entrée sur l'entrée I0,0, l'entrée sur I0,0 cesse, un certain temps s'écoule, une entrée sur l'entrée I0,1, l'entrée sur I0,1 cesse, des entrées sur les lignes suivantes. Le relais interne O0,0 restera actif :
    A. Pendant un cycle du programme à partir du début de l'entrée sur I0,0
    B. Depuis le début de l'entrée sur I0,0 jusqu'au début de l'entrée sur I0,1
    C. Depuis le début de l'entrée sur I0,0 jusqu'à la fin de l'entrée sur I0,1
    D. Depuis la fin de l'entrée sur I0,0 jusqu'à la fin de l'entrée sur I0,1
16. Une sortie sur Y430 doit être présente pendant un cycle après le début d'une entrée sur X100. Ce fonctionnement peut être obtenu par un programme en langage à contacts avec les instructions :
    A. LD X100, Y430
    B. LD X100, M100, LD M100, Y 430
    C. LD X100, PLS M100, LD M100, Y 430
    D. LD X400, PLS M100, LDI M100, Y430

Les problèmes 17 et 18 font référence à la Figure 7.35, qui présente deux versions du même schéma à contacts pour deux fabricants d'API. La Figure 7.35a utilise la notation de Siemens, avec I pour les entrées, F pour les relais internes et Q pour la sortie. La Figure 7.35b utilise la notation de Telemecanique, avec I pour les entrées et B pour les relais internes.

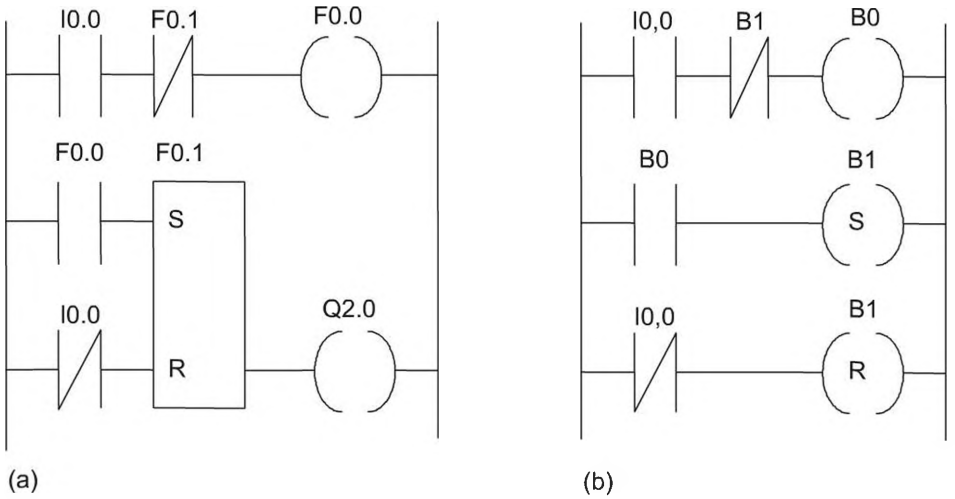


Figure 7.35 – Diagramme pour les problèmes 17 et 18.

17. Pour le schéma à contacts de la Figure 7.35a, lorsqu'une entrée est présente sur I0.0, la sortie Q2.0 :

Copyright © 2015 Dunod.

A. Devient active et le reste pendant un cycle
B. Devient active et le reste
C. Disparaît et reste inactive pendant un cycle
D. Disparaît et reste inactive

18. Pour le schéma à contacts de la Figure 7.35b, lorsqu'une entrée est présente sur I0,0, le relais interne B1 :
A. Devient actif et le reste pendant un cycle
B. Devient actif et le reste
C. Est coupé et reste inactif pendant un cycle
D. Est coupé et reste inactif

Les problèmes 19 et 20 font référence à la Figure 7.36, qui présente un programme en langage à contacts pour un API de Toshiba, avec trois entrées (X000, X001 et X002), une sortie (Y020) et une bascule (R110).

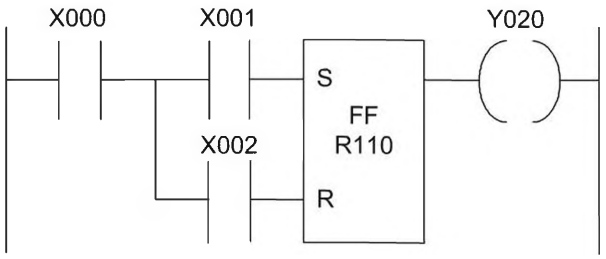


Figure 7.36 – Diagramme pour les problèmes 19 et 20.

19. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Pour que la sortie sur Y020 soit présente, il faut une entrée sur :
(i) X000 ;
(ii) X001.

Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
A. (i) V (ii) V
B. (i) V (ii) F
C. (i) F (ii) V
D. (i) F (ii) F

20. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Avec une entrée sur X000 :
(i) une entrée sur X001 provoque l'activation de la sortie ;
(ii) une entrée sur X002 provoque l'activation de la sortie.

Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
A. (i) V (ii) V
B. (i) V (ii) F
C. (i) F (ii) V
D. (i) F (ii) F

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

21. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Un relais de contrôle maître peut être utilisé pour :
    (i) activer une section d'un programme lorsque certains critères sont satisfaits ;
    (ii) désactiver une section d'un programme lorsque certains critères ne sont pas satisfaits.

    Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
    A. (i) V (ii) V
    B. (i) V (ii) F
    C. (i) F (ii) V
    D. (i) F (ii) F

Les problèmes 22 et 23 font référence à la Figure 7.37, qui présente un programme en langage à contacts pour un API d'Allen-Bradley.

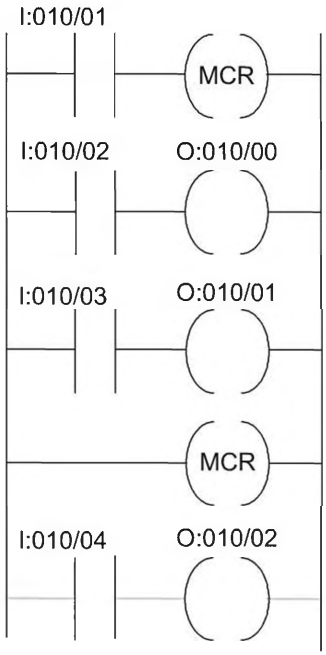


Figure 7.37 – Diagramme pour les problèmes 22 et 23.

22. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Lorsqu'il existe une entrée sur I:010/01 :
    (i) une entrée sur I:010/02 provoque une sortie sur O:010/00 ;
    (ii) une entrée sur I:010/03 provoque une sortie sur O:010/01.

    Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
    A. (i) V (ii) V
    B. (i) V (ii) F
    C. (i) F (ii) V
    D. (i) F (ii) F

Copyright © 2015 Dunod.

23. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Lorsqu'il n'existe pas d'entrée sur I:010/01 :
    (i) une entrée sur I:010/02 ne provoque aucune sortie sur O:010/00 ;
    (ii) une entrée sur I:010/04 ne provoque aucune sortie sur O:010/02.

    Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
    A. (i) V (ii) V
    B. (i) V (ii) F
    C. (i) F (ii) V
    D. (i) F (ii) F

24. Imaginez des programmes en langage à contacts qui peuvent être utilisés pour :
    (a) Maintenir une sortie active, même lorsque l'entrée cesse et que l'alimentation est coupée.
    (b) Activer une sortie pendant une durée d'un cycle suite à une entrée brève.
    (c) Mettre sous tension un ensemble de lignes.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

Copyright © 2015 Dunod.

8 • SAUTS ET APPELS

Ce chapitre se focalise sur l'instruction de saut, qui permet de passer par-dessus une partie d'un programme, et sur la manière d'invoquer des sous-programmes dans un programme en langage à contacts. Les sous-programmes sont des opérations souvent exécutées au sein d'un programme et qu'il est possible d'appeler de manière répétée et de réutiliser.

8.1 Sauts

Les automates programmables industriels possèdent souvent une fonction de *saut conditionnel*, que nous pouvons décrire de la manière suivante :

```
SI (une certaine condition se produit) ALORS
    exécuter ces instructions
SINON
    exécuter d'autres instructions
```

Grâce à cette fonction, les programmes peuvent être conçus de manière telle que lorsque certaines conditions sont remplies, certains événements se produisent, et lorsqu'elles ne sont pas remplies, d'autres événements se produisent. Par exemple, nous pourrions concevoir un système dans lequel un ventilateur est mis sous tension si la température dépasse 60 °C et dans lequel rien ne se passe si la température reste inférieure à ce seuil.

Ainsi, lorsque les conditions appropriées sont remplies, cette fonction permet de sauter par-dessus une partie d'un programme à contacts. La Figure 8.1 illustre ce concept de manière générale. Lorsqu'il existe une entrée sur E1, ses contacts se ferment et il existe une sortie sur le relais de saut. Cela conduit ensuite le programme à sauter jusqu'à la ligne dans laquelle se trouve la destination du saut. Les lignes intermédiaires sont donc ignorées. Dans cet exemple, lorsqu'une entrée est présente sur E1, le programme saute à la ligne 4 et poursuit son exécution sur les lignes 5, 6, etc. Lorsqu'il n'y a pas d'entrée sur E1, le relais de saut n'est pas activé et le programme continue avec les lignes 2, 3, etc.

La Figure 8.2a présente le programme à contacts précédent selon la notation de Mitsubishi. L'instruction de saut est indiquée par un saut conditionnel (CJP) et la destination est signalée par une fin de saut (EJP). Pour que le saut se produise, la condition est l'existence d'une entrée sur X400. Lorsque c'est le cas, les lignes qui comprennent les entrées X401 et X403 sont ignorées alors que le programme saute

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

aux lignes qui suivent l'instruction de fin de saut dont le numéro est identique à celui de l'instruction de début de saut — dans notre exemple, EJP 700.

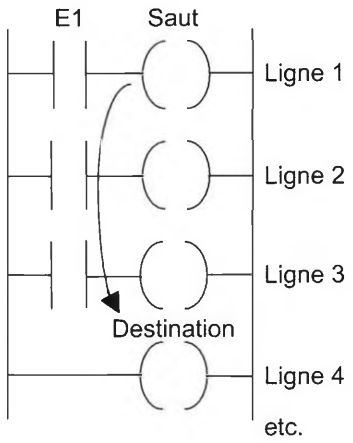


Figure 8.1 – Opération de saut.

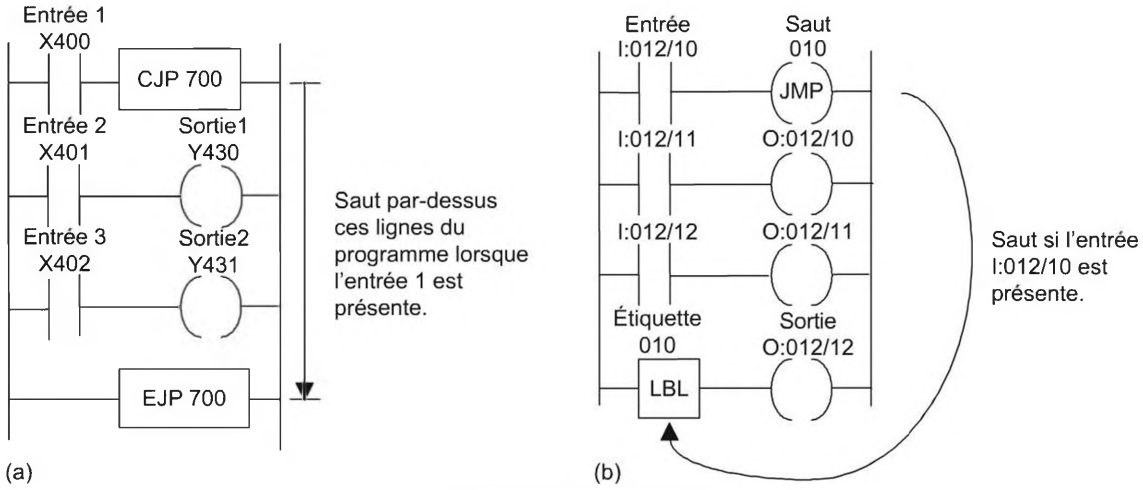


Figure 8.2 – Saut : (a) dans un programme Mitsubishi et (b) dans un programme Allen-Bradley.

Dans le format de l'API PLC-5 d'Allen-Bradley, la fonction est indiquée à l'aide d'une instruction de saut (JMP) et d'une instruction d'étiquette (LBL). L'instruction JMP doit préciser un numéro à trois chiffres dans l'intervalle 000 à 255 et l'instruction LBL associée doit préciser le même numéro. La Figure 8.2b montre un programme à contacts selon cette notation.

Dans un programme pour un API de Siemens, les sauts sont représentés conformément à la Figure 8.3. L'instruction de saut JMP est exécutée si l'entrée est à 1, tandis que l'instruction de saut JMPN est exécutée si l'entrée est à 0. La destination de ces deux instructions est le repère DEST.

Copyright © 2015 Dunod.

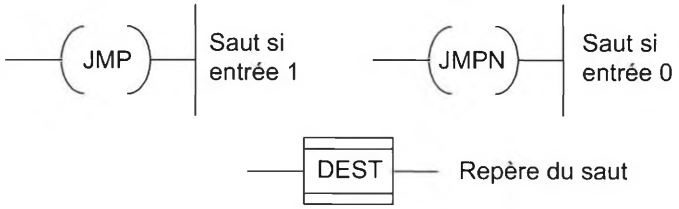


Figure 8.3 – Instructions de saut pour un API de Siemens.

8.1.1 Sauts imbriqués

Les sauts imbriqués sont possibles, comme, par exemple, dans le cas illustré à la Figure 8.4. Si la condition du saut 1 est vérifiée, le programme saute à la ligne 8. Si la condition n'est pas remplie, le programme poursuit à la ligne 3. Si la condition du saut 2 est vérifiée, le programme passe à la ligne 6. Si elle ne l'est pas, le programme poursuit sur les lignes suivantes.

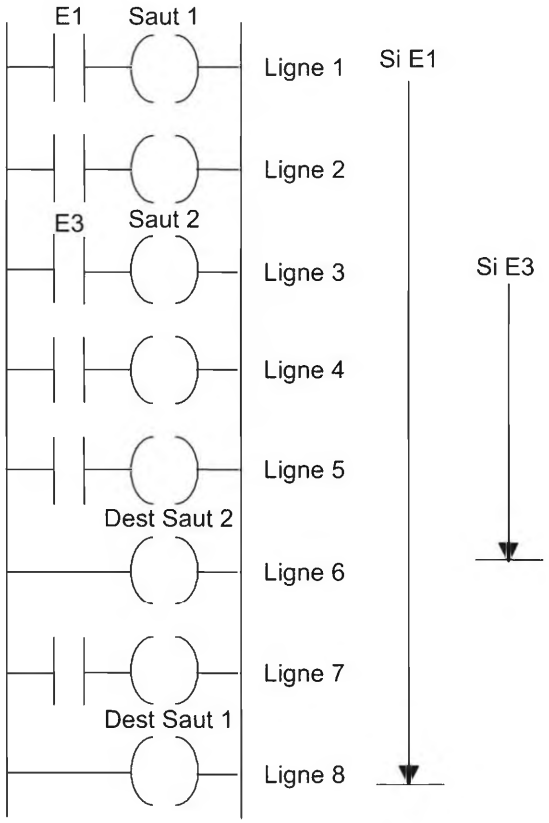


Figure 8.4 – Sauts imbriqués.

En cas d'entrée sur E1, les lignes exécutées sont les lignes 1, 8, etc. Si l'entrée sur E1 est absente mais que l'entrée sur E3 est présente, le programme passe par les lignes 1, 2, 6, 7, 8, etc. S'il n'existe aucune entrée sur E1 et E3, les lignes 1, 2, 3, 4, 5, 6, 7, 8, etc. sont exécutées. L'instruction de saut permet de sélectionner des groupes de lignes d'un programme en fonction de l'arrivée de certains conditions.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

8.2 Sous-programmes

Les *sous-programmes* sont de petits programmes qui réalisent des tâches particulières et qui peuvent être appelés depuis des programmes plus importants. L'utilisation des sous-programmes présente un avantage : ils peuvent être appelés de manière répétée pour réaliser des tâches spécifiques sans avoir à les récrire intégralement dans le programme principal. Par exemple, pour un programme Mitsubishi, nous pourrions avoir la situation illustrée à la Figure 8.5a. Lorsque l'entrée E1 se produit, le sous-programme P est appelé et exécuté. L'instruction SRET indique la fin du sous-programme et déclenche le retour au programme principal.

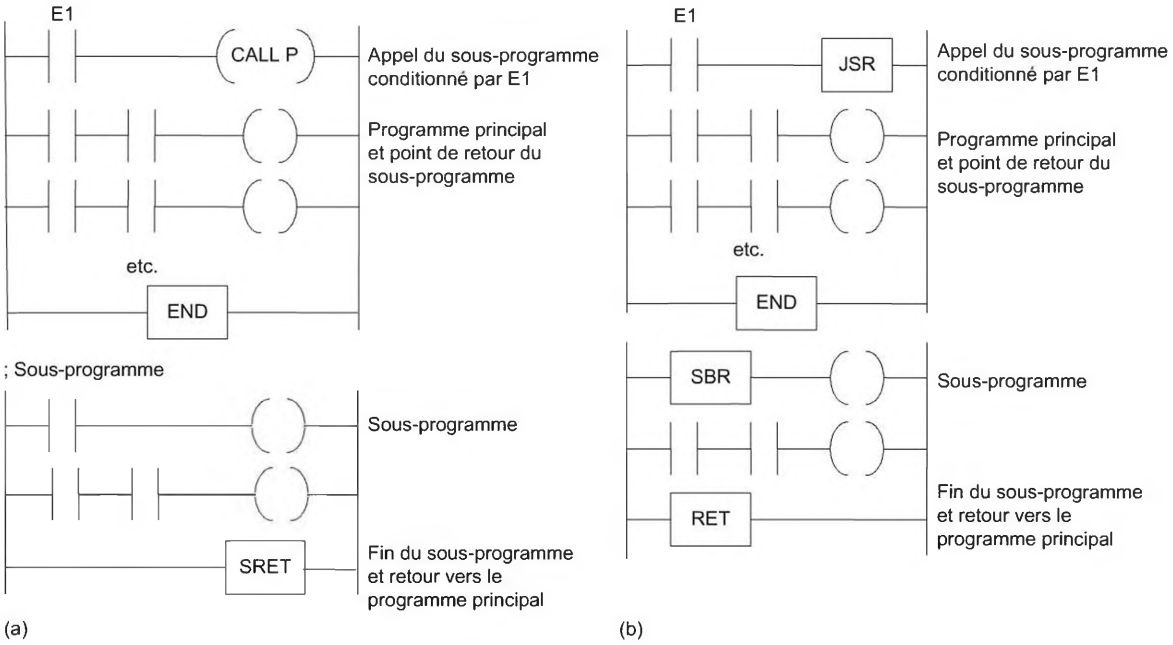


Figure 8.5 – (a) Appel de sous-programme dans un API de Mitsubishi et (b) saut vers un sous-programme dans un API d'Allen-Bradley.

Avec la syntaxe d'Allen-Bradley, les sous-programmes sont appelés à l'aide de l'instruction JSR. Le début du sous-programme est indiqué par SBR, tandis que sa fin et le retour au programme principal sont précisés par RET (voir Figure 8.5b). Les autres fabricants d'API adoptent un format comparable. Par exemple, l'appel d'un sous-programme peut se faire avec une instruction CALL, tandis que l'instruction RET indique le retour au programme principal.

8.2.1 Blocs fonctionnels

Il est possible d'utiliser une approche « bloc fonctionnel » dans les programmes, ce qui se révèle particulièrement utile lorsque les fonctions d'une bibliothèque de sous-programmes doivent être appelées. Un *bloc fonctionnel* est une partie d'un programme qui est préparée de manière à pouvoir être utilisée plusieurs fois dans d'autres portions du même programme ou dans des programmes différents. En utilisant les blocs, des programmes peuvent être construits à partir d'éléments plus

Copyright © 2015 Dunod.

petits et plus faciles à manipuler. Chaque bloc fonctionnel possède une entrée et une sortie pour sa connexion au programme principal, est capable de mémoriser des valeurs et contient un bout de programme qui s'exécute chaque fois que le bloc est utilisé, en traitant l'entrée pour produire la sortie. Les fabricants d'API proposent divers blocs fonctionnels que vous pouvez employer dans les programmes.

La Figure 8.6a montre la forme d'un bloc fonctionnel standard, comme un temporisateur à l'enclenchement (voir Chapitre 9). Lorsque l'entrée IN passe à 1, la sortie Q suit et reste à 1 pendant la durée fixée par l'entrée PT.

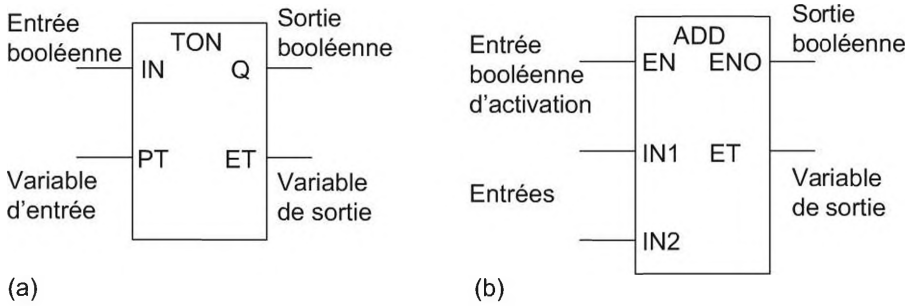


Figure 8.6 – Blocs fonctionnels.

Il est possible de contrôler un bloc fonctionnel (voir Figure 8.6b), par exemple un bloc fonctionnel additionnant deux entrées, en utilisant une entrée spéciale d'activation appelée EN (*enable*). Lorsque EN est à 1, la fonction du bloc est exécutée. Lorsqu'elle est à 0, le bloc est inopérant et n'affecte aucune valeur à la sortie. Ce type de bloc fonctionnel possède une sortie ENO qui est mise à un lorsque l'exécution de la fonction s'est terminée avec succès.

Si l'entrée EN du bloc est connectée directement à la barre d'alimentation de gauche, l'appel se fait sans condition et est donc toujours exécuté. Lorsqu'une opération logique précède EN, le bloc est appelé uniquement lorsque la condition est remplie. À la Figure 8.7, il s'agit de la fermeture des contacts de l'entrée 1. Plusieurs blocs peuvent être connectés en série, en reliant ENO (sortie d'activation) à l'entrée EN du bloc suivant.

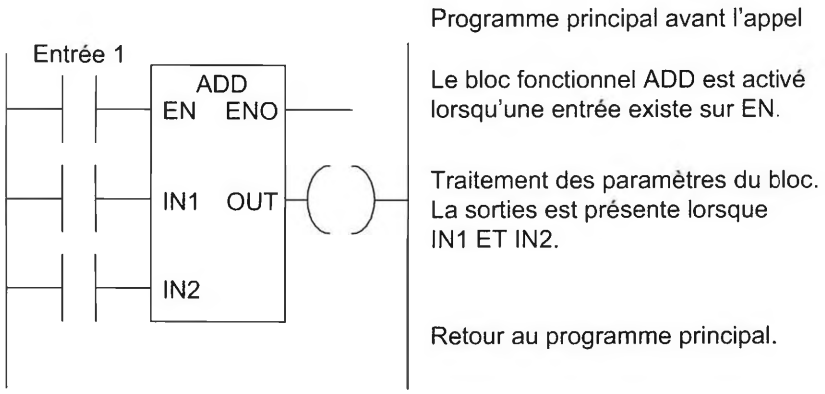


Figure 8.7 – Appel d'un bloc fonctionnel dans un API de Siemens.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

À titre d'illustration, prenons le bloc fonctionnel standard SR (un bistable, décrit à la Section 3.8) de la Figure 8.8a.

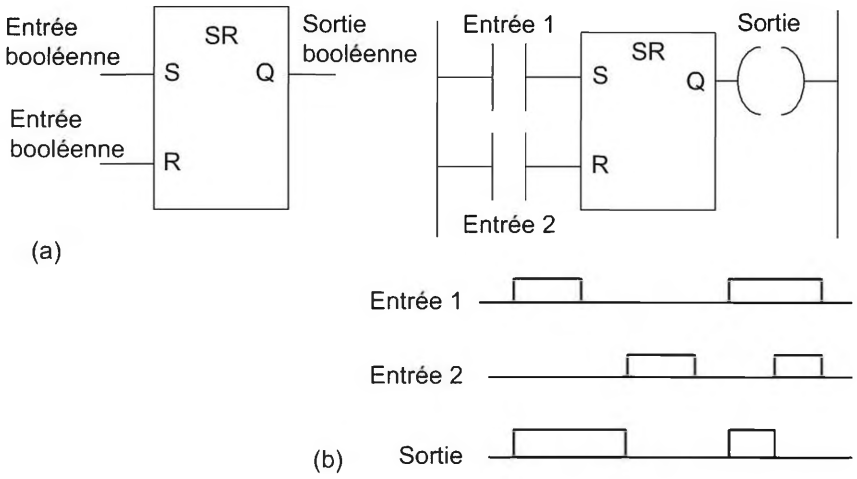


Figure 8.8 – (a) Symbole d'un bloc fonctionnel SR et (b) bloc SR dans un programme à contacts.

Lorsque l'entrée S est à 1 et que l'entrée R est à 0, la sortie Q est à 1 et le bloc « mémorise » cet état tant qu'il n'est pas remis à zéro. Si les entrées S et R sont toutes deux à 1, la sortie est à 1. La « mémoire » est mise à zéro lorsque l'entrée R est à 1 et l'entrée S est à 0. La Figure 8.8b présente l'utilisation d'un tel bloc dans un schéma à contacts.

La Figure 8.9 propose une autre illustration avec le bloc fonctionnel RS (un verrou bistable). La sortie est à 1 lorsque l'entrée S est à 1 ; elle est à 0 lorsque l'entrée R est à 1. Si les entrées S et R sont toutes deux à 1, la sortie est à 0.

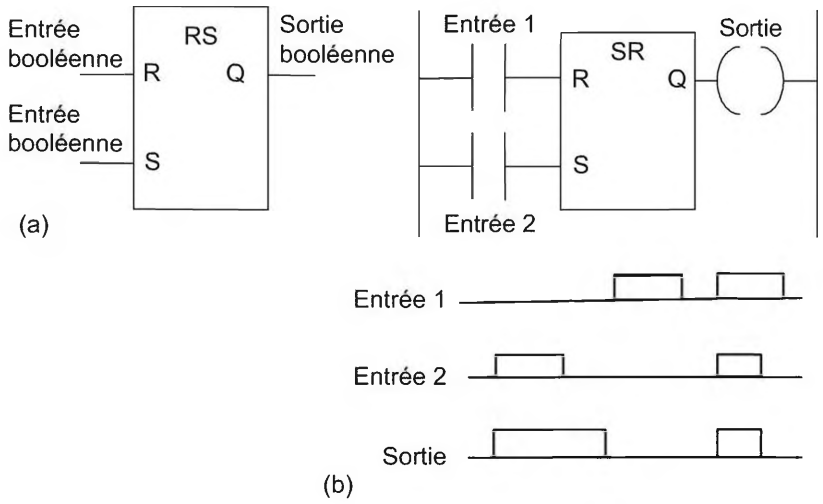


Figure 8.9 – (a) Symbole d'un bloc fonctionnel RS et (b) bloc RS dans un programme à contacts.

D'autres blocs fonctionnels couramment utilisés seront présentés dans les chapitres ultérieurs.

Copyright © 2015 Dunod.

8.3 En résumé

Les automates programmables industriels possèdent souvent une fonction de saut conditionnel, que nous pouvons décrire de la manière suivante : SI une certaine condition se produit ALORS exécuter ces instructions SINON exécuter d'autres instructions.

Les sous-programmes sont de petits programmes qui réalisent des tâches particulières et qui peuvent être appelés depuis des programmes plus importants. L'utilisation des sous-programmes présente un avantage : ils peuvent être appelés de manière répétitive pour réaliser des tâches spécifiques sans avoir à les récrire intégralement dans le programme principal. Il est possible d'utiliser une approche « bloc fonctionnel » dans les programmes, ce qui se révèle particulièrement utile lorsque les fonctions d'une bibliothèque de sous-programmes doivent être appelées. Un bloc fonctionnel est une partie d'un programme qui est préparée de manière à pouvoir être utilisée plusieurs fois dans d'autres portions du même programme ou dans des programmes différents. En utilisant des blocs, des programmes peuvent être construits à partir d'éléments plus petits et plus faciles à manipuler. Chaque bloc fonctionnel possède une entrée et une sortie pour sa connexion au programme principal, est capable de mémoriser des valeurs et contient un bout de programme qui s'exécute chaque fois que le bloc est utilisé, en traitant l'entrée pour produire la sortie. Les fabricants d'API proposent différents blocs fonctionnels, que vous pouvez employer dans les programmes.

8.4 Problèmes

Pour les questions 1 à 6, quatre réponses sont proposées : A, B, C et D. Choisissez la réponse correcte. Les problèmes 1 et 2 font référence à la Figure 8.10, qui présente un schéma à contacts avec quatre entrées (E1, E2, E3 et E4), quatre sorties (S1, S2, S3 et S4) et une instruction de saut.

1. D'après le schéma à contacts de la Figure 8.10, la sortie S1 est présente si :
   A. Seule l'entrée E1 est présente
   B. Les deux entrées E1 et E2 sont présentes
   C. L'entrée E1 est absente et l'entrée E2 est présente
   D. Les deux entrées E1 et E2 sont absentes
2. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Pour le schéma à contacts de la Figure 8.10, la présence de l'entrée E1 :
   (i) active la sortie S1 ;
   (ii) active la sortie S3.

   Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
   A. (i) V (ii) V
   B. (i) V (ii) F
   C. (i) F (ii) V
   D. (i) F (ii) F

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

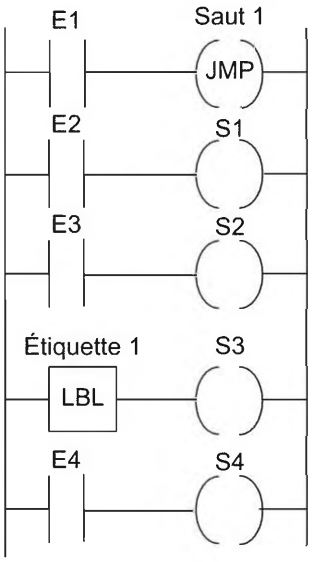


Figure 8.10 – Diagramme pour les problèmes 1 et 2.

Les problèmes 3 et 4 font référence à la Figure 8.11, qui présente un schéma à contacts avec trois entrées (E1, E2 et E3), trois sorties (S1, S2 et S3) et une instruction de saut vers un sous-programme.

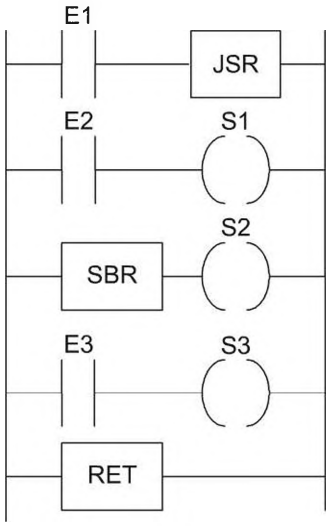


Figure 8.11 – Diagramme pour les problèmes 3 et 4.

3. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Pour le schéma à contacts de la Figure 8.11 :
   (i) après l'activation de l'entrée E1, la sortie S2 se produit ;
   (ii) après l'activation de la sortie S3, le programme attend l'activation de l'entrée E2 avant de poursuivre.

   Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?

Copyright © 2015 Dunod.

A. (i) V (ii) V
B. (i) V (ii) F
C. (i) F (ii) V
D. (i) F (ii) F

4. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Pour le schéma à contacts de la Figure 8.11 :
(i) lorsque l'entrée E2 est présente, les sorties S1 et S2 se produisent ;
(ii) lorsque l'entrée E3 est présente, la sortie S3 se produit.

Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
A. (i) V (ii) V
B. (i) V (ii) F
C. (i) F (ii) V
D. (i) F (ii) F

5. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Pour le programme donné à la Figure 8.12, il existe une sortie lorsque :
(i) l'entrée 1 est à 1 et l'entrée 2 est à 0 ;
(ii) les entrées 1 et 2 sont à 1.

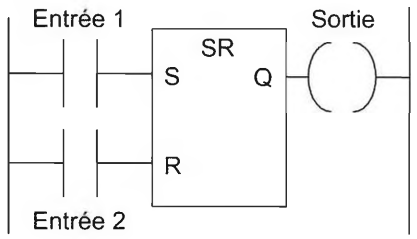


Figure 8.12 – Diagramme pour le problème 5.

Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
A. (i) V (ii) V
B. (i) V (ii) F
C. (i) F (ii) V
D. (i) F (ii) F

6. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Pour le programme donné à la Figure 8.13, il existe une sortie lorsque :
(i) l'entrée 1 est à 1, l'entrée 2 est à 0 et l'entrée 3 est à 1 ;
(ii) l'entrée 1 est à 0, l'entrée 2 est à 1 et l'entrée 3 est à 0.

Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
A. (i) V (ii) V
B. (i) V (ii) F
C. (i) F (ii) V
D. (i) F (ii) F

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

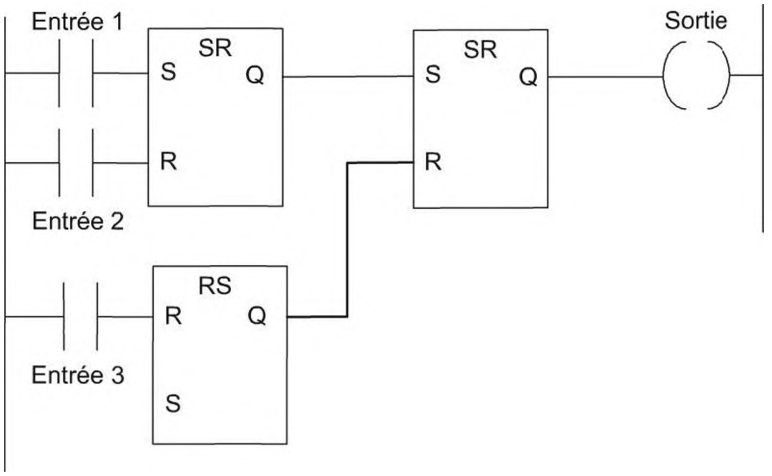


Figure 8.13 – Diagramme pour le problème 6.

7. Le programme d'un équipement de production doit répéter plusieurs fois les opérations suivantes : remplir une cuve, chauffer le liquide de la cuve et, lorsque le liquide est à la température adéquate, vider la cuve. Expliquez comment cette procédure peut être programmée avec un sous-programme.

8.5 Recherches complémentaires

1. Indiquez les blocs fonctionnels disponibles pour un modèle d'API.
2. Indiquez la méthode de programmation utilisée pour invoquer un sous-programme dans un modèle d'API.

Copyright © 2015 Dunod.

9 • TEMPORISATEURS

Pour nombre de tâches de commandes, il est nécessaire de contrôler le temps. Par exemple, un moteur ou une pompe peuvent avoir besoin de fonctionner pendant une durée précise ou d'être mis en marche après un certain temps. C'est pourquoi les API disposent de temporisateurs intégrés. Les temporisateurs comptent les secondes ou les fractions de seconde en utilisant l'horloge interne du CPU. Ce chapitre explique comment ces blocs fonctionnels de temporisation peuvent être programmés pour mener à bien certaines tâches de commande.

9.1 Types de temporisateurs

Les fabricants d'API ne s'accordent pas sur la programmation des temporisateurs ni sur la manière de les considérer. Une approche classique consiste à les voir comme des relais avec des bobines dont l'excitation conduit à la fermeture ou à l'ouverture des contacts après une durée déterminée. Le temporisateur est donc vu comme une sortie d'une ligne, avec une commande effectuée ailleurs par des contacts (voir Figure 9.1a). Cette approche est celle que nous employons principalement dans cet ouvrage. D'autres fabricants voient le temporisateur comme un bloc de retard qui, lorsqu'il est inséré dans une ligne, retarde les signaux d'entrée avant qu'ils n'atteignent la sortie (voir Figure 9.1b).

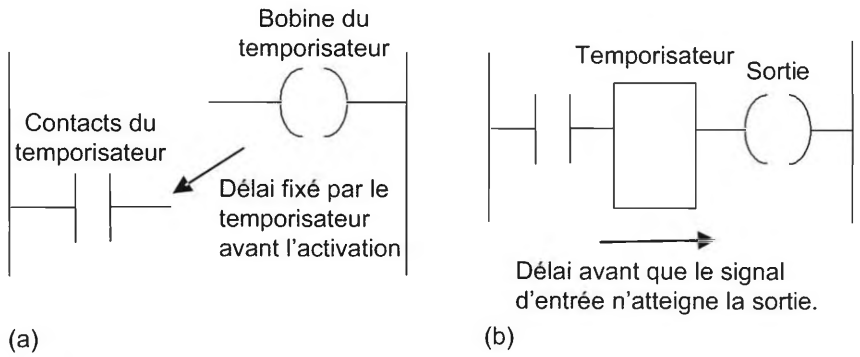


Figure 9.1 – Traitement des temporisateurs.

Dans les API, vous trouverez plusieurs variantes de temporisateurs : à l'enclenchement, au déclenchement et à impulsion. Les petits API ne proposeront sans doute

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

qu'une seule variante, les temporisateurs à l'enclenchement. La Figure 9.2 présente les symboles définis par la CEI. TON correspond aux temporisateurs à l'enclenchement, TOF, aux temporisateurs au déclenchement et TP, aux temporisateurs à impulsion. Les temporisateurs à l'enclenchement sont également représentés par T-0, tandis que ceux au déclenchement le sont par 0-T.

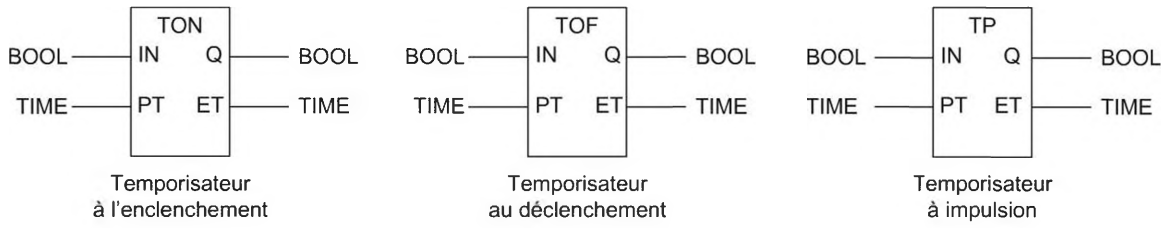


Figure 9.2 – Norme CEI 61131-1 : IN est l'entrée booléenne, Q est la sortie booléenne, ET est la sortie de temps écoulé, PT est l'entrée qui précise le retard ou la durée de l'impulsion.

Les *temporisateurs à l'enclenchement* (TON) s'activent après un certain retard (voir Figure 9.3a). Ainsi, lorsque l'entrée passe de 0 à 1, le temps écoulé ET commence à augmenter et, lorsqu'il atteint la durée fixée par l'entrée PT, la sortie passe à 1. Les *temporisateurs au déclenchement* (TOF) sont actifs pendant une durée fixée, avant de se désactiver (voir Figure 9.3b). La temporisation débute lorsque le signal d'entrée passe de 1 à 0. Le *temporisateur à impulsion* (TP) représente un autre type de temporisateur, qui produit une sortie à 1 pendant une durée fixée (voir Figure 9.3c). Elle débute lorsque l'entrée passe de 0 à 1, puis la sortie revient à 0 lorsque le temps indiqué par PT est écoulé.

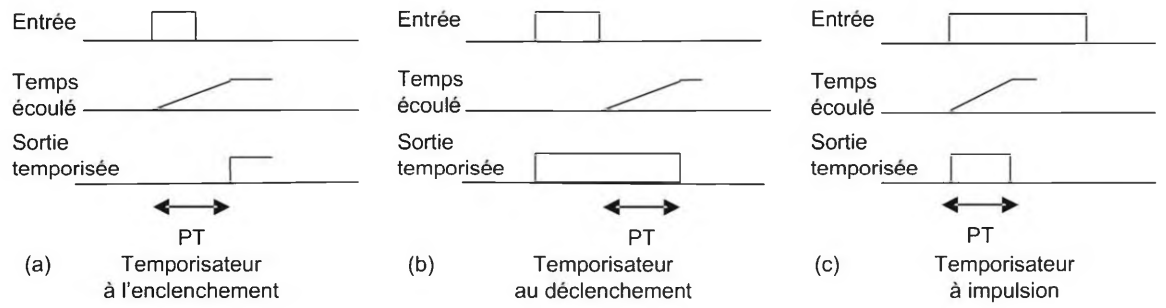


Figure 9.3 – Temporisateurs : (a) à l'enclenchement, (b) au déclenchement et (c) à impulsion.

La temporisation définie est appelée *présélection* et correspond à un multiple de la base de temps. Les bases de temps les plus fréquentes sont 10 ms, 100 ms, 1 s, 10 s et 100 s. Ainsi, avec une présélection de 5 et une base de temps de 100 ms, la temporisation est de 500 ms. Pour des raisons pratiques, nous utiliserons une base de temps de 1 s.

Copyright © 2015 Dunod.

9.2 Temporisateurs à l'enclenchement

En général, tous les API proposent des temporisateurs à l'enclenchement ; un petit API offrira sans doute uniquement ce type de temporisateur. La Figure 9.4 montre un schéma à contacts qui comprend un temporisateur à l'enclenchement. La Figure 9.4a utilise la notation de Mitsubishi. Le temporisateur est comparable à un relais avec une bobine qui est excitée lorsque l'entrée E1 se produit (ligne 1). Après un certain retard fixé, ses contacts sur la ligne 2 se ferment. Par conséquent, la sortie apparaît un certain temps après l'activation de l'entrée E1. La Figure 9.4b présente le temporisateur en tant que bloc de retard dans une ligne, non comme un relais. Cet exemple correspond à un API de Siemens. Lorsque le signal sur l'entrée de démarrage du temporisateur passe de 0 à 1, le temporisateur démarre et fonctionne pendant la durée programmée. Ensuite, sa sortie est envoyée à la bobine de sortie. La sortie TV peut être utilisée pour déterminer le temps restant à tout moment. Lorsque l'entrée de réinitialisation passe à 1, le temporisateur est remis à zéro, qu'il soit ou non en exécution. Il existe plusieurs méthodes pour préciser la valeur de présélection de la temporisation. Elle consiste souvent à saisir une commande de constante K, suivie de la durée donnée sous forme d'un multiple de la base de temps. Les Figures 9.4c, 9.4d et 9.4e montrent les schémas à contacts pour des API de Telemecanique, de Toshiba et d'Allen-Bradley. Le symbole employé par Allen-Bradley montre le type du temporisateur, son adresse et la base de temps, qui indique les incréments utilisés par le temporisateur pour atteindre la valeur de présélection, comme 0,001 s, 0,01 s, 0,1 s ou 1 s. La valeur de présélection (PRE) correspond au nombre d'incréments que le temporisateur doit effectuer pour atteindre la durée requise. L'accumulateur (ACC) précise le nombre d'incréments effectués par le temporisateur depuis qu'il est actif. Sa valeur est remise à zéro lorsque le temporisateur est mis à zéro (utile si le programme doit enregistrer la durée d'une opération). Les temporisateurs d'Allen-Bradley disposent de trois bits booléens pour la logique de commande. Le bit d'activation du temporisateur (EN) passe à 1 lorsque l'accumulateur du temporisateur augmente. Le bit de fin de la temporisation (DN) passe à 1 après que le délai fixé est écoulé. Le bit de temporisation en cours (TT) est à 1 lorsque l'accumulateur augmente et reste à 1 jusqu'à ce que l'accumulateur atteigne la valeur présélectionnée.

Tous les programmes illustrés à la Figure 9.4 activent le dispositif de sortie après un certain retard suite à l'activation de l'entrée.

9.2.1 Séquencement

Afin d'illustrer l'utilisation d'un temporisateur TON, examinons le schéma à contacts de la Figure 9.5. Lorsque l'entrée E1 est à 1, la sortie S1 est activée. Les contacts associés à cette sortie démarrent ensuite le temporisateur. Les contacts du temporisateur se fermeront une fois la durée de présélection écoulée, dans ce cas 5,5 s. Lorsque cela se produit, la sortie S2 est activée. En résumé, suite à l'activation de l'entrée E1, la sortie S1 est activée, puis, après un retard de 5,5 s, la sortie S2 est activée. Ce fonctionnement montre comment obtenir une chronologie des sorties. La Figure 9.5b présente la même opération dans le cas où le temporisateur induit un retard du signal. La Figure 9.5c donne le chronogramme correspondant.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

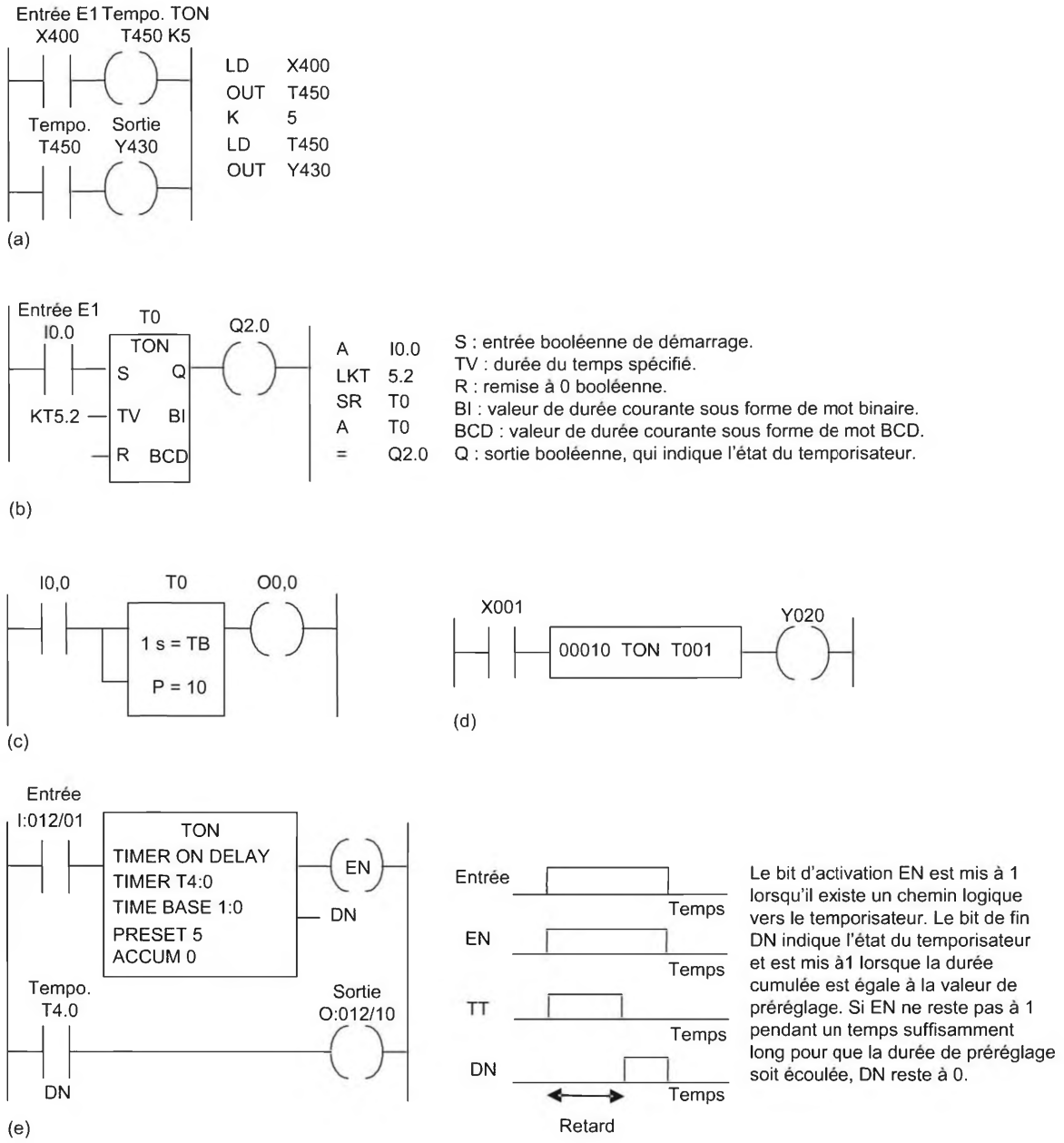


Figure 9.4 – Temporisateurs : (a) Mitsubishi, (b) Siemens, (c) Telemecanique, (d) Toshiba et (e) Allen-Bradley.

La Figure 9.6 montre deux manières d'utiliser des temporisateurs pour démarrer en séquence trois sorties, par exemple trois moteurs, après l'appui sur un bouton de mise en marche. En (a), les temporisateurs sont programmés comme des bobines, tandis qu'en (b), ils sont programmés comme des retards. Un appui sur le bouton-poussoir provoque une sortie sur le relais interne RI1. Cela verrouille l'entrée de démarrage et déclenche les deux temporisateurs, T1 et T2, ainsi que le moteur 1. Lorsque la durée de présélection du temporisateur T1 est écoulée, ses contacts se ferment et le moteur 2 démarre. Lorsque la durée de présélection du temporisateur T2 est écoulée, ses contacts se ferment et le moteur 3 démarre. Les trois moteurs sont stoppés en appuyant sur le bouton-poussoir d'arrêt. Puisque nous présentons un programme complet, l'instruction de fin n'a pas été oubliée.

Copyright © 2015 Dunod.

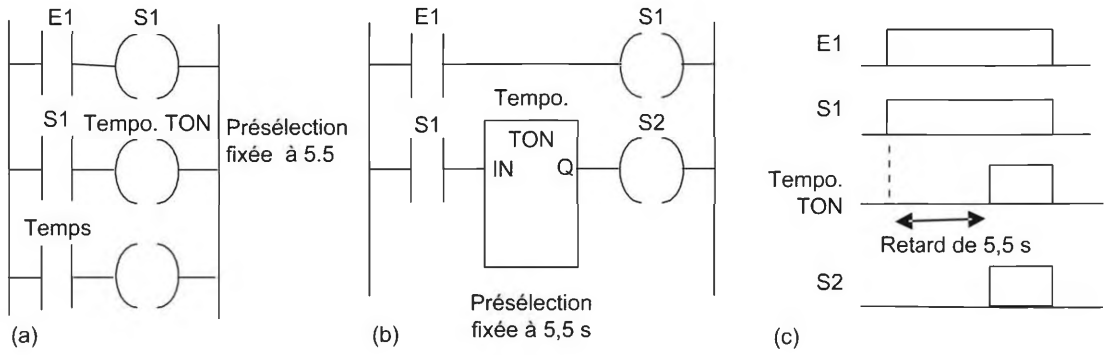


Figure 9.5 – Séquencement des sorties.

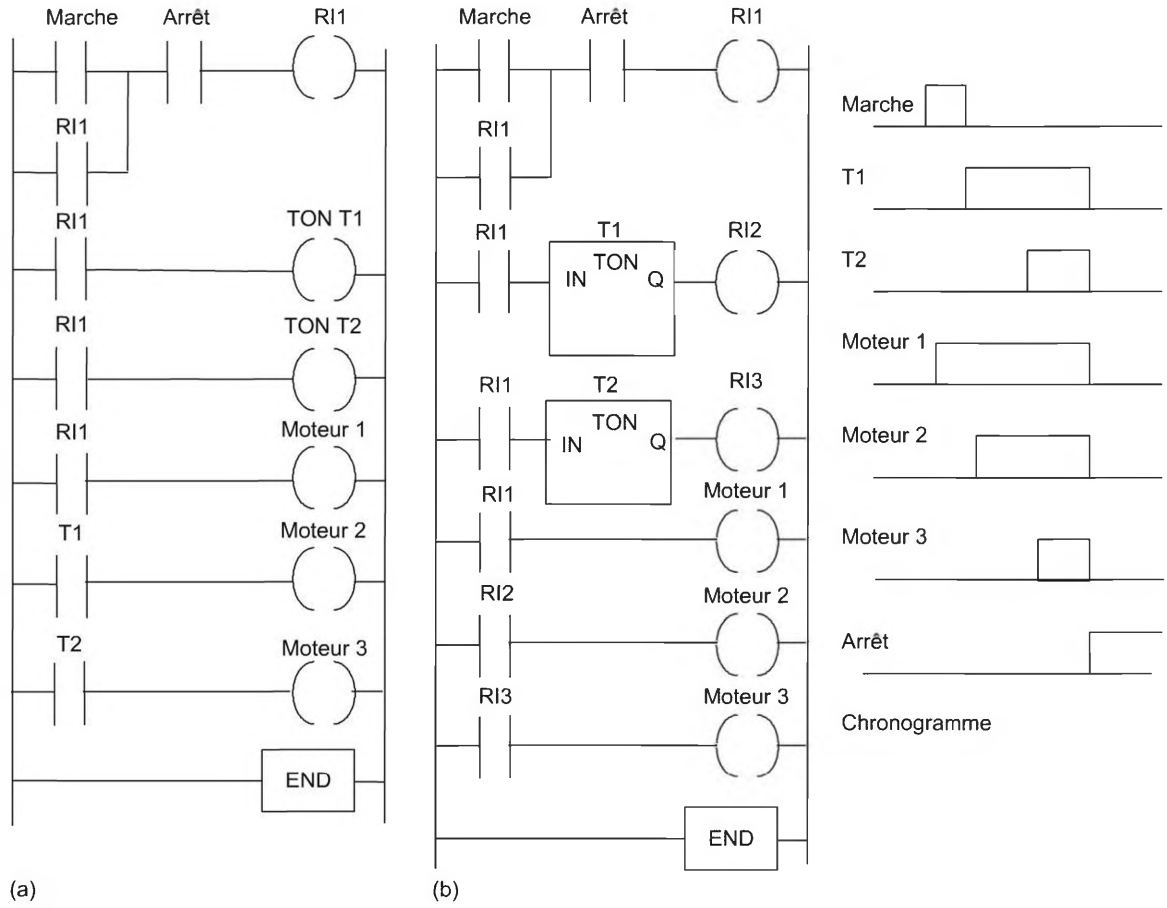


Figure 9.6 – Séquencement de moteurs.

9.2.2 Temporisateurs en cascade

Des temporisateurs peuvent être reliés les uns aux autres (mis en *cascade*) pour obtenir un retard plus long que ne le permet un seul temporisateur. La Figure 9.7a présente le schéma à contacts correspondant. Le temporisateur 1 peut être configuré avec un retard de 999 s. Il est démarré par la présence de l'entrée E1. Lorsque 999 s se sont écoulées, les contacts du temporisateur 1 se ferment. Cela déclenche le temporisateur 2, dont la présélection est fixée à 100 s. Lorsque cette durée est écoulée, les contacts du temporisateur 2 se ferment et la sortie S1 est activée. Ainsi,

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

la sortie se produit 1 099 s après l'activation de l'entrée E1. La Figure 9.7b montre la version Mitsubishi de ce schéma à contacts avec des temporisateurs TON, ainsi que les instructions du programme correspondantes.

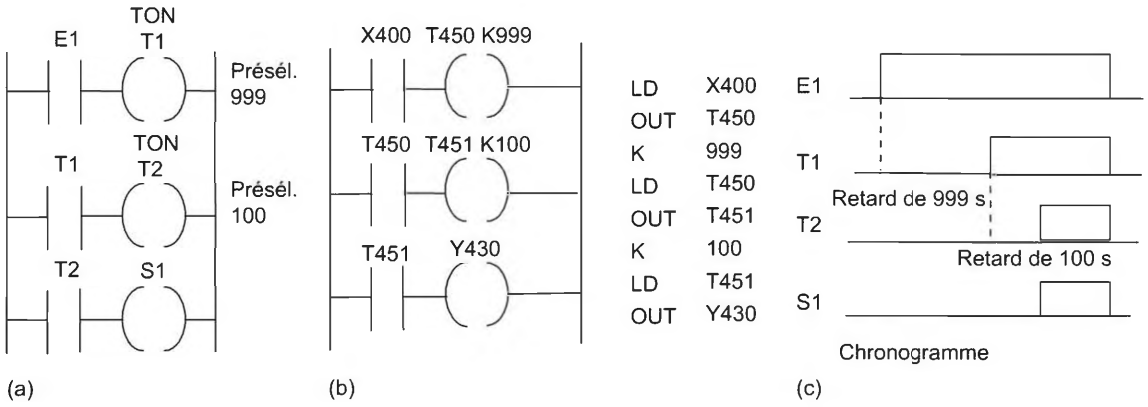


Figure 9.7 – Temporisateurs TON en cascade.

9.2.3 Temporisateur cyclique marche-arrêt

La Figure 9.8 illustre l'utilisation des TON pour obtenir un *temporisateur cyclique marche-arrêt*. Le temporisateur est conçu pour activer une sortie pendant 5 s, puis la désactiver pendant 5 s, puis l'activer pendant 5 s, puis la désactiver pendant 5 s, etc. Lorsque l'entrée E1 est présente et que ses contacts se ferment, le temporisateur 1 démarre. Il est configuré avec un retard de 5 s. Après 5 s, il active le temporisateur 2 et la sortie S1. Le temporisateur 2 est configuré avec un retard de 5 s. Après 5 s, les contacts du temporisateur 2, qui sont normalement fermés, s'ouvrent. Cela conduit à la désactivation du temporisateur 1 sur la première ligne, à l'ouverture de ses contacts sur la deuxième ligne et à la désactivation du temporisateur 2. Les contacts du temporisateur 2 reviennent donc à leur état normalement fermés et l'entrée sur E1 provoque la reprise du cycle.

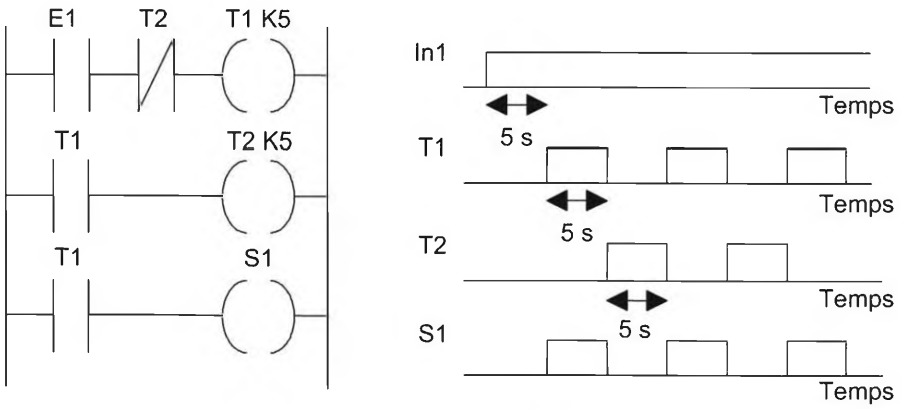


Figure 9.8 – Temporisateur cyclique marche-arrêt.

Copyright © 2015 Dunod.

La Figure 9.9 reprend le schéma à contacts précédent en considérant que le temporisateur est un retard non une bobine. Il correspond, par exemple, à un API de Siemens ou de Toshiba. Lorsque l'entrée E1 se ferme, le temporisateur T1 démarre. Une fois que son temps de présélection est écoulé, la sortie S1 est activée et le temporisateur T2 démarre. Après que son temps de présélection est écoulé, la sortie sur le relais interne RI1 est activée. Cela ouvre ses contacts et désactive la sortie sur S1. Le temporisateur T2 est alors arrêté. L'intégralité du cycle peut ensuite reprendre.

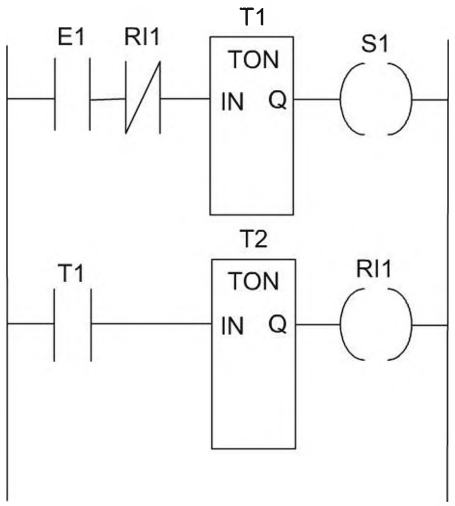


Figure 9.9 – Temporisateur cyclique marche-arrêt avec des TON.

9.3 Temporisateurs au déclenchement

La Figure 9.10 montre comment utiliser un temporisateur à l'enclenchement pour obtenir un *temporisateur au déclenchement* (TOF). Avec un tel schéma, lorsqu'une entrée momentanée est présente en E1, la sortie S1 et le temporisateur sont activés. Puisque l'entrée est verrouillée par les contacts de S1, la sortie reste active. Lorsque la durée de présélection du temporisateur est écoulée, ses contacts normalement fermés s'ouvrent et désactivent la sortie. Par conséquent, la sortie est tout d'abord active et le reste jusqu'à ce que la durée de temporisation soit écoulée.

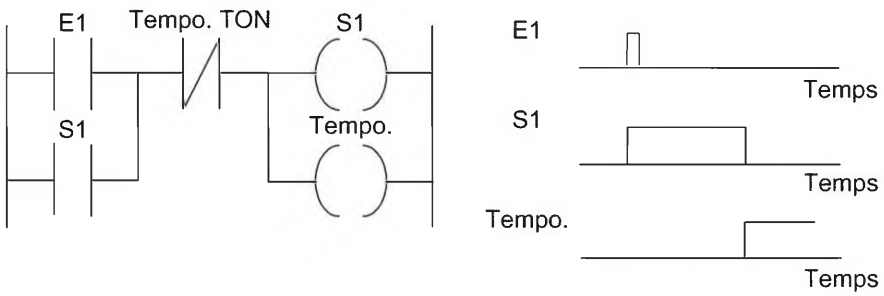


Figure 9.10 – Temporisateur au déclenchement à l'aide d'un TON.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

Dans certains API, les temporisateurs à l'enclenchement et les temporisateurs au déclenchement sont intégrés. Dans ce cas, il est inutile d'utiliser un TON pour obtenir un TOF. La Figure 9.11a illustre cette fonction pour un API d'Allen-Bradley, tandis que la Figure 9.11b correspond à un API de Siemens, avec le schéma à contacts et la liste d'instructions. Vous noterez que, pour l'API de Siemens, le temporisateur est un élément de retard dans la ligne, non un relais.

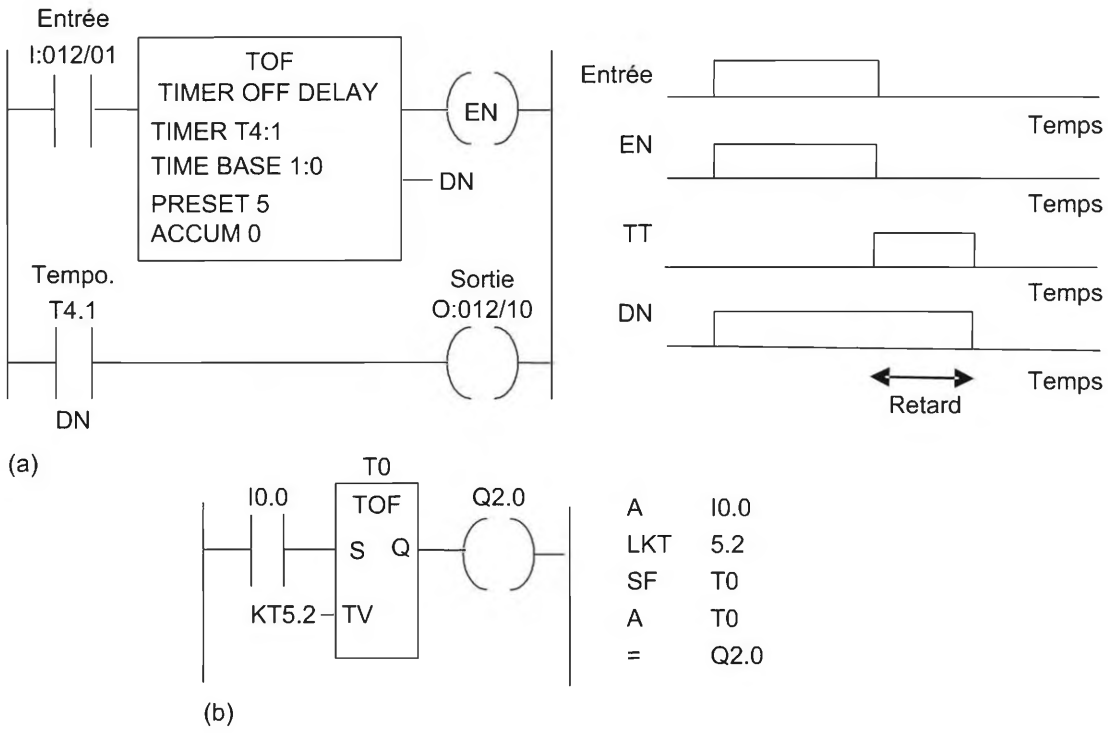


Figure 9.11 – Temporisateurs au déclenchement : versions (a) Allen-Bradley et (b) Siemens.

Afin d'illustrer l'utilisation d'un temporisateur au déclenchement, examinons le programme Allen-Bradley de la Figure 9.12. Le libellé TOF précise que le temporisateur est au déclenchement, non à l'enclenchement (TON). La base de temps est fixée à 1:0, c'est-à-dire 1 s. La présélection est 10, ce qui donne une temporisation de 10 s.

Sur la première ligne, la sortie du temporisateur est prise à partir des contacts EN (pour *enable*, c'est-à-dire activation). Cela signifie qu'il n'y a aucun retard entre l'arrivée du signal d'entrée sur I:012/01 et la sortie EN. Par conséquent, les contacts EN sur la ligne 2 se ferment immédiatement dès que l'entrée I:012/01 est activée. Lorsque l'entrée I:012/01 se produit, la sortie O:013/01 est immédiatement activée. Les contacts TT (pour *timer timing*, c'est-à-dire temporisation en cours) de la ligne 3 sont excités pendant le fonctionnement du temporisateur. Puisqu'il s'agit d'un temporisateur au déclenchement, il est actif pendant 10 s avant de devenir inactif. Les contacts TT se ferment donc pendant que le délai de 10 s s'écoule. La sortie O:012/02 est ainsi active pendant cette durée de 10 s. Les contacts DN (pour *done*, c'est-à-dire fin de la temporisation) normalement fermés

Copyright © 2015 Dunod.

s'ouvrent après les 10 s et la sortie O:013/03 est alors activée. Les contacts DN normalement ouverts se ferment après 10 s et la sortie O:013/04 est désactivée après ces 10 s.

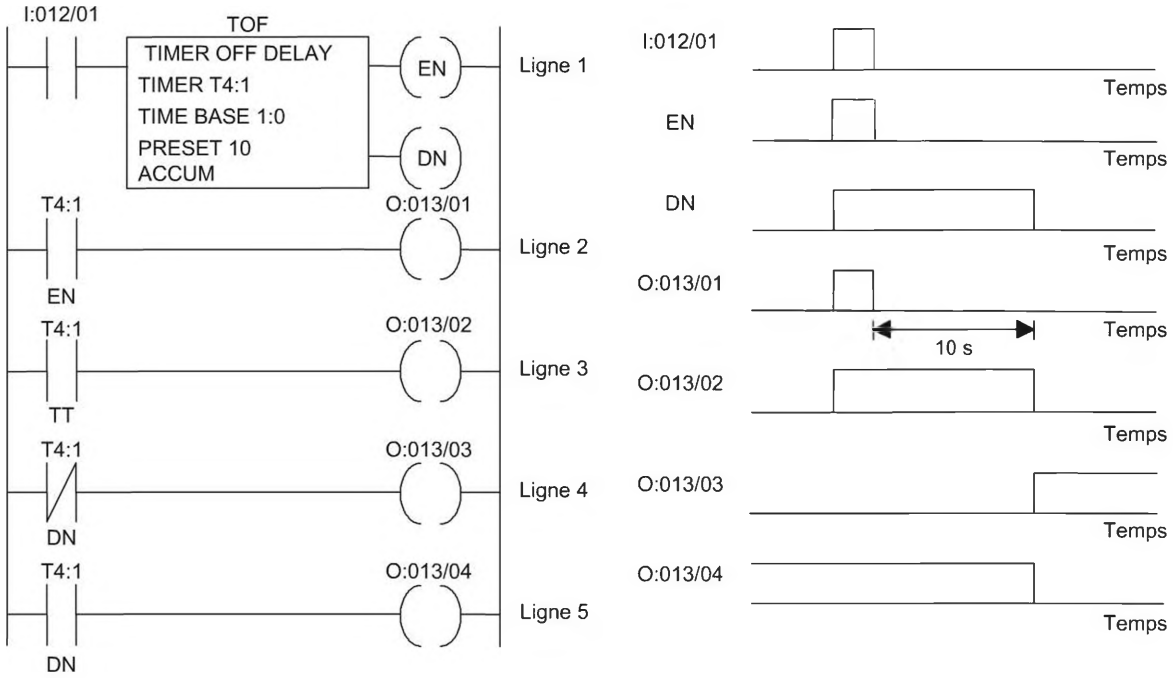


Figure 9.12 – Mise en application d'un temporisateur au déclenchement.

9.4 Temporisateurs à impulsion

Les *temporisateurs à impulsion* produisent une sortie de durée fixée suite à l'activation d'une entrée. La Figure 9.13a montre le schéma à contacts d'un système qui produit une sortie sur S1 pendant une durée déterminée lorsque l'entrée E1 est présente ; le temporisateur est vu comme une bobine. Il existe deux sorties pour l'entrée E1. En cas d'entrée sur E1, la sortie S1 est activée et le temporisateur démarre. Lorsque la durée présélectionnée est écoulée, les contacts du temporisateur se ferment, ce qui désactive la sortie. Par conséquent, la sortie reste active pendant la durée définie par le temporisateur.

La Figure 9.13b présente un schéma à contacts équivalent à celui de la Figure 9.13a, mais fondé sur un temporisateur qui retarde l'arrivée d'un signal sur la sortie.

La sortie du temporisateur à impulsion de la Figure 9.13 est activée pendant une durée prédéterminée, puis désactivée. La Figure 9.14 montre un autre temporisateur à impulsion qui active une sortie pendant une durée prédéterminée après que l'entrée a disparu. Pour cela, un temporisateur et deux relais internes sont employés. En cas d'entrée sur E1, le relais interne RI1 est excité. Le temporisateur ne démarre pas à ce moment-là car les contacts normalement fermés d'E1 sont ouverts. La fermeture des contacts de RI1 provoque l'excitation du relais interne RI2. Cependant, à ce stade, il n'y a aucune sortie sur S1 car, sur la ligne inférieure,

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

les contacts d'E1 sont ouverts. Lorsque l'entrée sur E1 cesse, les deux relais internes restent excités et le temporisateur est démarré. Après la durée fixée, les contacts normalement fermés du temporisateur s'ouvrent et désactivent RI2. Cela désactive ensuite RI1, ainsi que, sur la ligne suivante, la sortie S1. Par conséquent, la sortie est inactive pendant la durée de l'entrée, puis elle est activée pendant une durée prédéterminée.

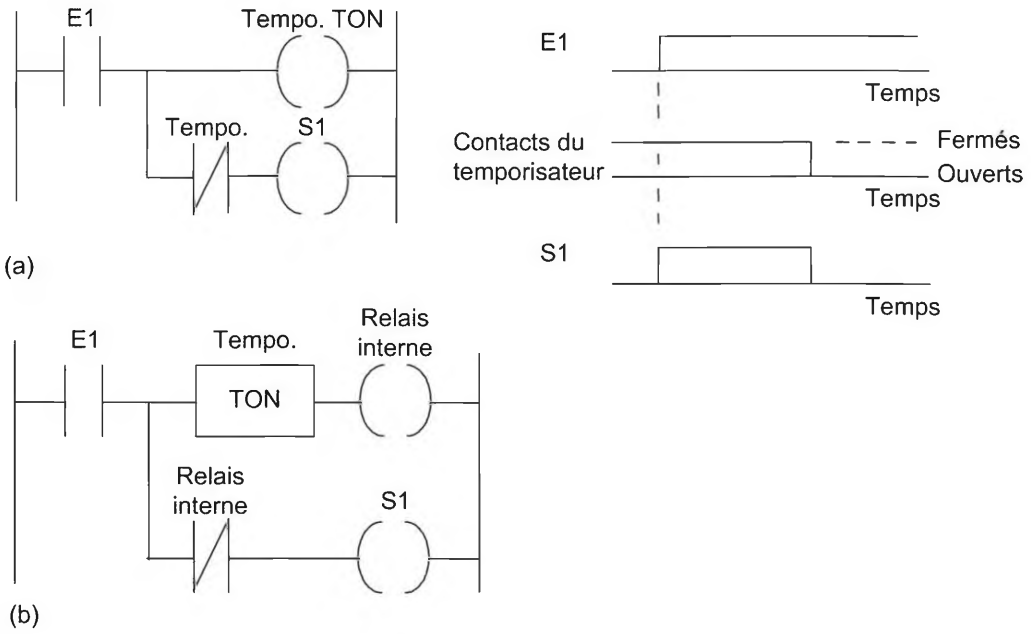


Figure 9.13 – Temporisateur à impulsion.

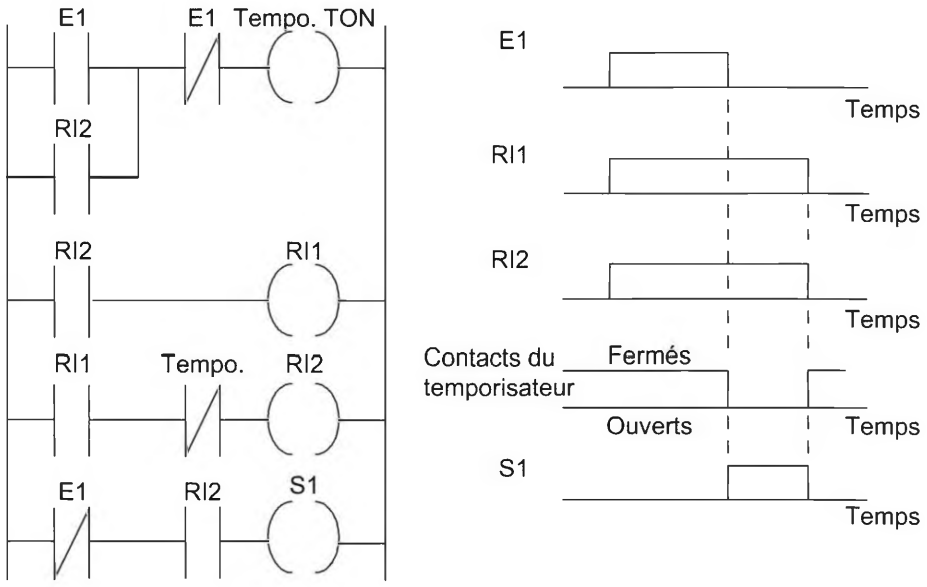


Figure 9.14 – Temporisateur à impulsion activé lorsque l'entrée cesse.

Copyright © 2015 Dunod.

9.5 Temporisateurs rémanents

Certains API disposent également de temporisateurs rémanents. Un *temporisateur rémanent* (RTO), tel que celui d'Allen-Bradley (voir Figure 9.15), peut être démarré et arrêté sans remettre à zéro la valeur cumulée. Il fonctionne de la même manière qu'un temporisateur TON, excepté que l'accumulateur n'est pas réinitialisé lorsque l'entrée d'activation du temporisateur est à 0 et continue à augmenter dès qu'elle est à 1. Lorsque l'accumulateur atteint la valeur de présélection, le bit de temporisation en cours passe à 0 et le bit de fin de temporisation, à 1. Il reste dans cet état jusqu'à la réception d'une instruction de remise à zéro.

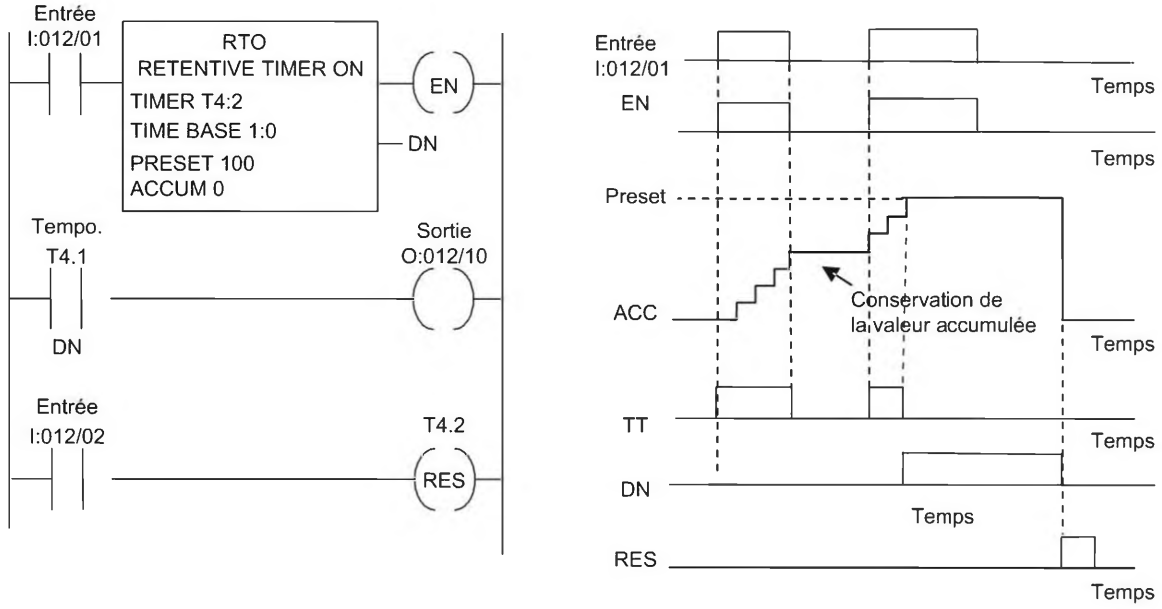


Figure 9.15 – Temporisateur rémanent d'Allan-Bradley.

9.6 Exemples de programmes

Examinons un programme qui permet de faire clignoter un témoin lumineux pendant qu'une sortie est activée (voir Figure 9.16). Les temporisateurs 0 et 1 sont configurés à 1 s. Lorsque la sortie est présente, le temporisateur 0 démarre et s'active après 1 s. Ses contacts se ferment alors, ce qui démarre le temporisateur 1. Il devient actif après 1 s, ce qui désactive le temporisateur 0. La lampe est allumée uniquement lorsque le temporisateur 0 est actif. Nous disposons ainsi d'un programme qui fait clignoter le témoin lumineux tant qu'une sortie est présente.

Pour illustrer la programmation avec des temporisateurs, étudions le séquencement d'un feu de circulation (selon le fonctionnement en vigueur au Royaume-Uni) afin d'obtenir l'allumage du feu rouge uniquement, du feu rouge et du feu orange, du feu vert, puis du feu orange, avec reprise du cycle. Un système simple peut se fonder sur une temporisation pour obtenir le séquencement, chacun des états possibles se produisant l'un à la suite de l'autre pendant une durée fixe. La Figure 9.17 présente le graphe de fonction séquentielle et le schéma à contacts qui permettent d'obtenir ce séquencement.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

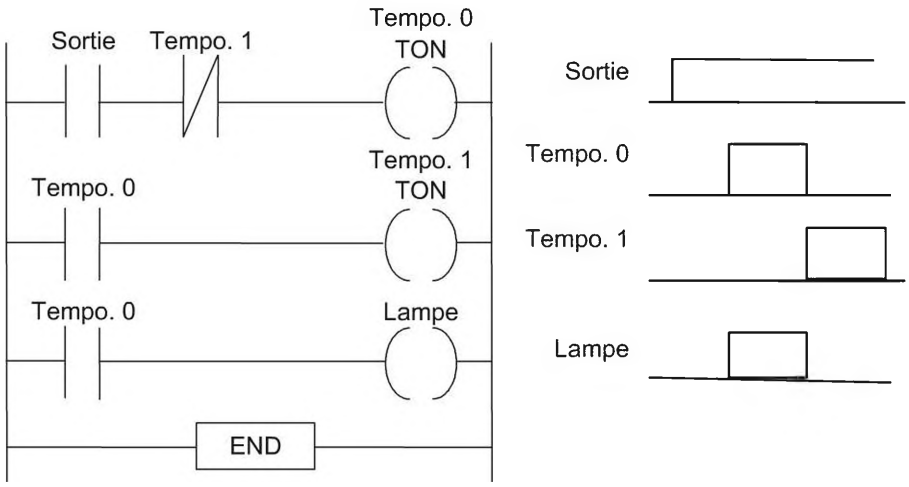


Figure 9.16 – Clignotement d'un témoin lumineux.

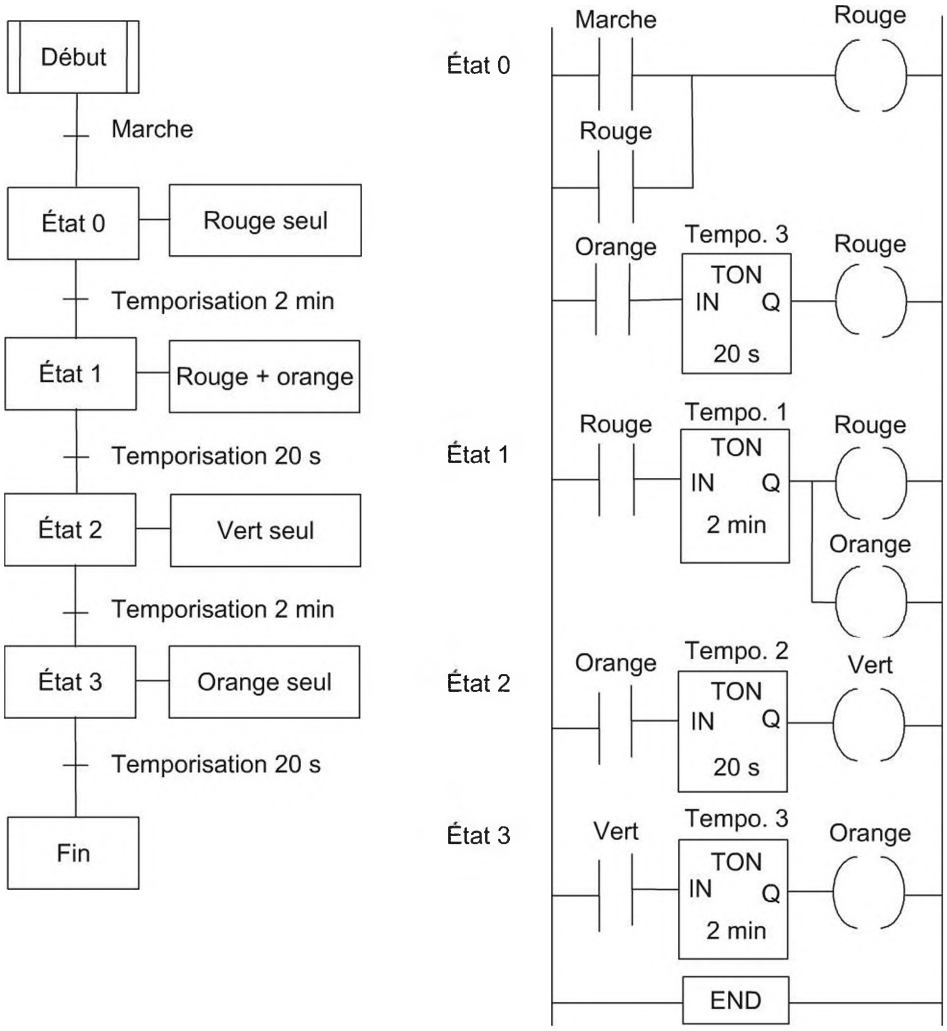


Figure 9.17 – Séquencement d'un feu de circulation.

Copyright © 2015 Dunod.

La Figure 9.18 illustre une autre utilisation d'un temporisateur. Il s'agit d'un programme qui initialement active le moteur 1 et son témoin lumineux L1, puis, après une durée fixée par le temporisateur, arrête le moteur 1 et le témoin 1, et active le moteur 2 et le témoin lumineux 2. Lorsque l'entrée est présente, le temporisateur démarre. Tant que la temporisation n'a pas expiré, les contacts fermés du temporisateur restent fermés pour que le moteur 1 et la lampe 1 soient activés ; les contacts ouverts des temporisateurs restent ouverts et le moteur 2 et la lampe 2 sont inactifs. Lorsque la temporisation a expiré, les contacts fermés du temporisateur s'ouvrent et désactivent le moteur 1 et la lampe 1 ; les contacts ouverts du temporisateur se ferment et activent le moteur 2 et la lampe 2.

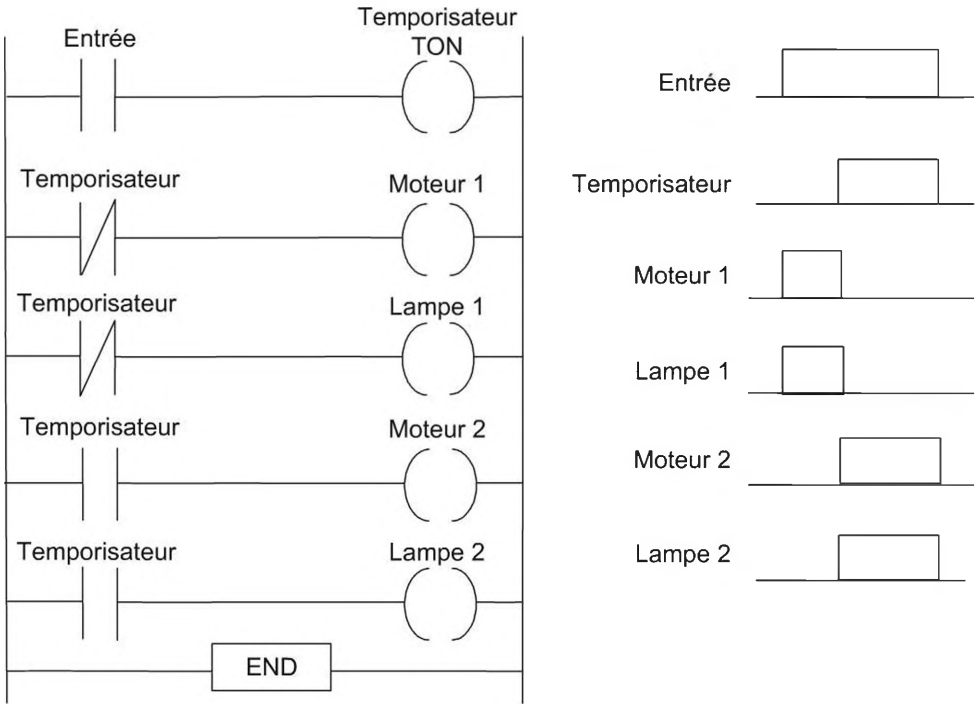


Figure 9.18 – Programme pour la séquence activer le moteur 1, le désactiver et activer le moteur 2.

L'exemple de la Figure 9.19 se fonde sur un temporisateur TON et un temporisateur TOF pour commander un moteur. La première ligne commence par un interrupteur de mise en marche, qui est verrouillé par le temporisateur TON afin que l'entrée reste active même lorsque l'interrupteur n'est plus activé. Une fois que la durée du temporisateur TON est écoulée, le moteur est démarré. Il est stoppé par une entrée sur l'interrupteur d'arrêt, mais il continue à fonctionner pendant la durée fixée par le temporisateur TOF, avant de s'arrêter totalement.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

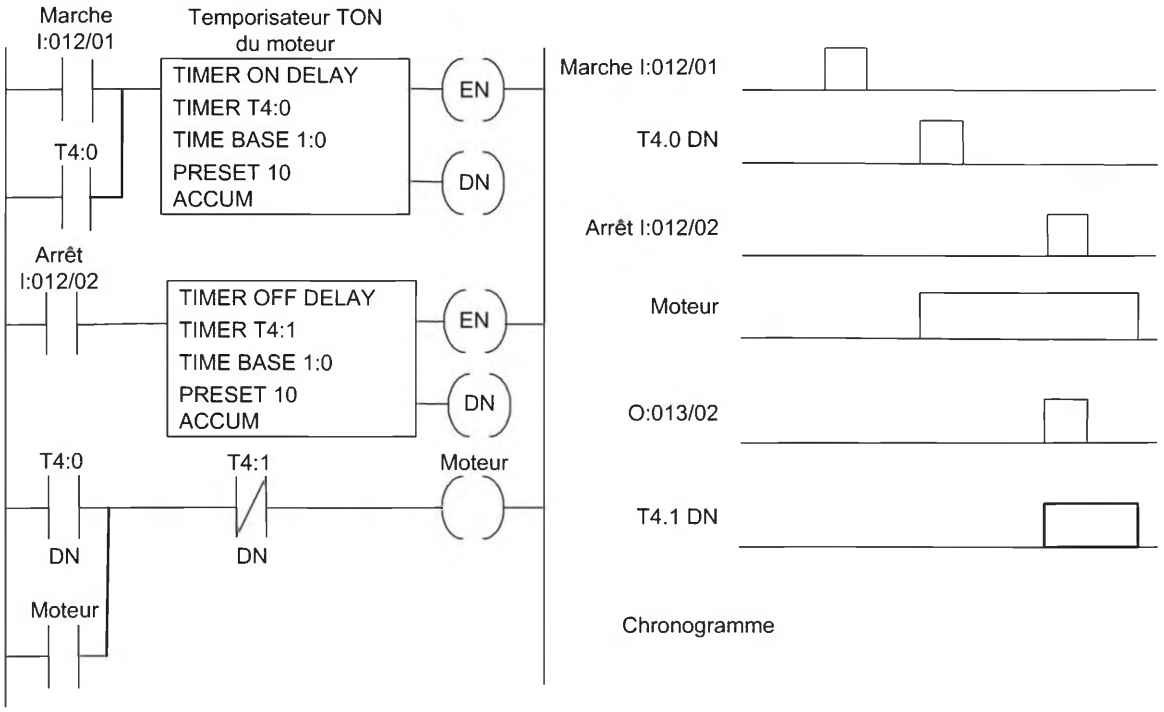


Figure 9.19 – Démarrage et arrêt retardés d'un moteur.

9.7 En résumé

Les temporisateurs sont généralement vus comme des relais avec des bobines dont l'excitation conduit à la fermeture ou à l'ouverture des contacts après une durée déterminée. Toutefois, d'autres fabricants voient le temporisateur comme un bloc de retard qui, lorsqu'il est inséré dans une ligne, retarde les signaux d'entrée avant qu'ils n'atteignent la sortie. Dans les API, il existe plusieurs variantes de temporisateurs : à l'enclenchement (TON), au déclenchement (TOF) et à impulsion (TP). Les petits API ne proposeront sans doute qu'une seule variante, les temporisateurs à l'enclenchement. Avec un temporisateur à l'enclenchement, après activation du temporisateur, les contacts se ferment lorsque la durée définie est écoulée. Avec un temporisateur au déclenchement, après activation du temporisateur, les contacts ne s'ouvrent pas jusqu'à expiration de la durée définie. Un temporisateur rémanent (RTO) peut démarrer et s'arrêter sans que la valeur du temps cumulé soit remise à zéro.

9.8 Problèmes

Pour les questions 1 à 21, quatre réponses sont proposées : A, B, C et D. Choisissez la réponse correcte. Les problèmes 1 à 3 font référence à la Figure 9.20, qui présente un schéma à contacts avec un temporisateur à l'enclenchement, une entrée (E1) et une sortie (S1).

Copyright © 2015 Dunod.

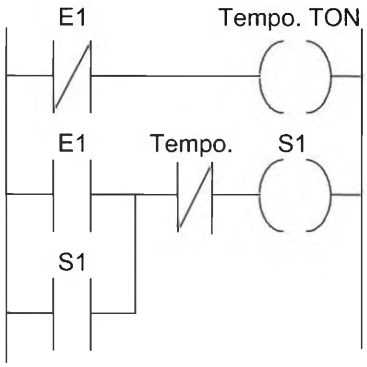


Figure 9.20 – Diagramme pour les problèmes 1 à 3.

1. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). À la Figure 9.20, lorsque l'entrée E1 est présente :
   (i) le temporisateur démarre ;
   (ii) il existe une sortie sur S1.

   Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
   A. (i) V (ii) V
   B. (i) V (ii) F
   C. (i) F (ii) V
   D. (i) F (ii) F

2. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Le temporisateur de la Figure 9.20 démarre lorsque :
   (i) la sortie est présente ;
   (ii) l'entrée cesse.

   Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
   A. (i) V (ii) V
   B. (i) V (ii) F
   C. (i) F (ii) V
   D. (i) F (ii) F

3. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). À la Figure 9.20, lorsque l'entrée E1 est présente, la sortie est :
   (i) activée pendant la durée définie sur le temporisateur ;
   (ii) désactivée pendant la durée définie sur le temporisateur.

   Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
   A. (i) V (ii) V
   B. (i) V (ii) F
   C. (i) F (ii) V
   D. (i) F (ii) F

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

Les problèmes 4 à 6 font référence à la Figure 9.21, qui présente deux versions d'un schéma à contacts avec deux entrées (E1 et E2), deux sorties (S1 et S2) et un temporisateur à l'enclenchement.

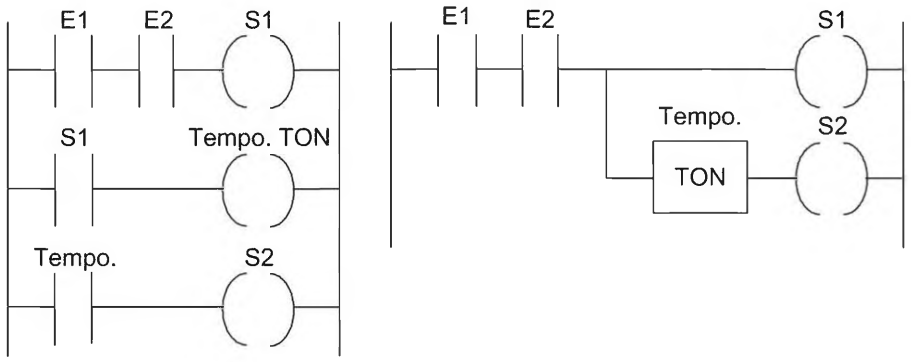


Figure 9.21 – Diagramme pour les problèmes 4 à 6.

4. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). À la Figure 9.21, lorsque seule l'entrée E1 est présente :
   (i) le temporisateur démarre ;
   (ii) il existe une sortie sur S2.

   Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
   A. (i) V (ii) V
   B. (i) V (ii) F
   C. (i) F (ii) V
   D. (i) F (ii) F

5. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). À la Figure 9.21, lorsque seule l'entrée E2 est présente :
   (i) le temporisateur démarre ;
   (ii) il existe une sortie sur S2.

   Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
   A. (i) V (ii) V
   B. (i) V (ii) F
   C. (i) F (ii) V
   D. (i) F (ii) F

6. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). À la Figure 9.21, lorsque l'entrée E1 est présente et l'entrée E2 est absente, il existe une sortie sur S2 qui :
   (i) se produit immédiatement ;
   (ii) cesse après la durée de la temporisation.

   Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?

Copyright © 2015 Dunod.

A. (i) V (ii) V
B. (i) V (ii) F
C. (i) F (ii) V
D. (i) F (ii) F

7. Voici une liste d'instructions pour un API de Mitsubishi :

```
LD      X400
OUT     T450
K       6
LD      T450
OUT     Y430
```

Une entrée sur X400 produit :

A. Une sortie active pendant 6 s puis inactive pendant 6 s
B. Une sortie active pendant 6 s
C. Une sortie active après 6 s
D. Une sortie inactive pendant 6 s puis active pendant 6 s

8. Voici une liste d'instructions pour un API de Telemecanique

```
L       I0,0
=       T0
L       T0
=       O0,0
```

Une entrée sur I0,0 produit :

A. Une sortie active pendant 6 s puis inactive pendant 6 s
B. Une sortie active pendant 6 s
C. Une sortie active après 6 s
D. Une sortie inactive pendant 6 s puis active pendant 6 s

Les problèmes 9 à 10 font référence à la liste d'instructions suivante pour un API de Mitsubishi :

```
LD      X400
OR      Y430
ANI     T450
OUT     Y430
LD      X401
OR      M100
AND     Y430
OUT     T450
K       10
OUT     M100.
```

9. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Lorsqu'une entrée est momentanément présente sur X400 :
(i) la sortie sur Y430 s'active ;
(ii) la sortie sur Y430 cesse après 10 s.

Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?

A. (i) V (ii) V
B. (i) V (ii) F
C. (i) F (ii) V
D. (i) F (ii) F

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

10. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). La sortie sur Y430 :
    (i) s'active lorsqu'il existe une entrée momentanée sur X401 ;
    (ii) cesse 10 s après une entrée sur X401.

    Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
    A. (i) V (ii) V
    B. (i) V (ii) F
    C. (i) F (ii) V
    D. (i) F (ii) F

Les problèmes 11 et 12 font référence à la Figure 9.22, qui présente un système avec une entrée (E1), un temporisateur à l'enclenchement et une sortie (S1). Le temporisateur est configuré avec une durée de 6 s. Le graphique montre la variation du signal d'entrée en fonction du temps.

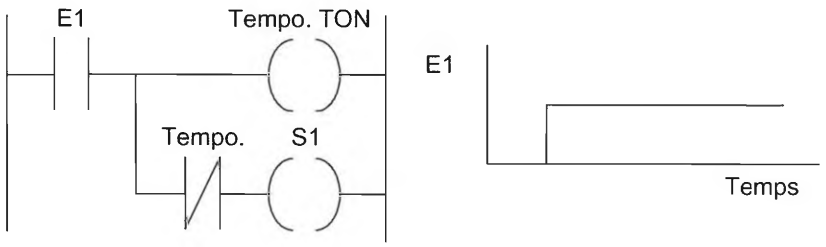


Figure 9.22 – Diagramme pour les problèmes 11 et 12.

11. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). À la Figure 9.22, la sortie sur S1 :
    (i) débute lorsque l'entrée débute ;
    (ii) cesse 6 s peu après le début de l'entrée.

    Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
    A. (i) V (ii) V
    B. (i) V (ii) F
    C. (i) F (ii) V
    D. (i) F (ii) F

12. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). À la Figure 9.22, les contacts du temporisateur :
    (i) restent fermés pendant 6 s après le début de l'entrée ;
    (ii) s'ouvrent 6 s après le début de l'entrée.

    Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
    A. (i) V (ii) V
    B. (i) V (ii) F
    C. (i) F (ii) V
    D. (i) F (ii) F

Copyright © 2015 Dunod.

Les problèmes 13 à 15 font référence à la Figure 9.23, qui présente un schéma à contacts pour un API de Toshiba avec des relais internes, représentés par la lettre R, et un temporisateur TON avec une présélection de 20 s.

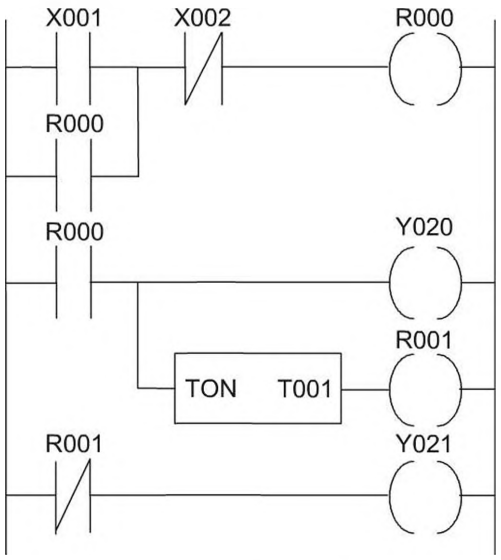


Figure 9.23 – Diagramme pour les problèmes 13 à 15.

13. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). À la Figure 9.23, le relais interne R000 est utilisé pour :
    (i) verrouiller l'entrée X001 ;
    (ii) démarrer le temporisateur T001.

    Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
    A. (i) V (ii) V
    B. (i) V (ii) F
    C. (i) F (ii) V
    D. (i) F (ii) F

14. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). À la Figure 9.23, en l'absence d'une entrée sur X002, la sortie Y020 :
    (i) est excitée lorsque l'entrée X001 est présente ;
    (ii) cesse lorsqu'il n'y a pas d'entrée sur X001.

    Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
    A. (i) V (ii) V
    B. (i) V (ii) F
    C. (i) F (ii) V
    D. (i) F (ii) F

15. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). À la Figure 9.23, en l'absence d'une entrée sur X002 :
    (i) la sortie Y021 est activée 20 s après une entrée sur X001 ;
    (ii) la sortie Y020 est désactivée 20 s après une entrée sur X001.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?

A. (i) V (ii) V
B. (i) V (ii) F
C. (i) F (ii) V
D. (i) F (ii) F

Les problèmes 16 à 19 font référence à la Figure 9.24, qui présente un programme Allen-Bradley, et à la Figure 9.25, qui montre plusieurs chronogrammes pour un signal d'entrée particulier sur I:012/01.

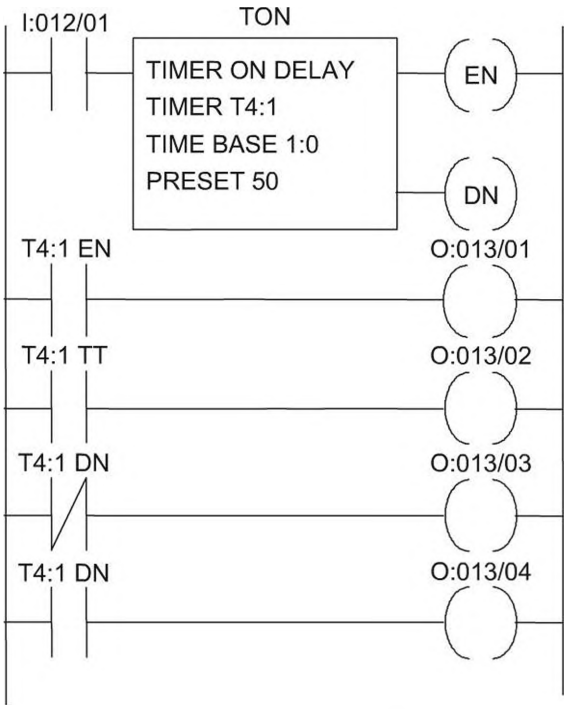


Figure 9.24 – Diagramme pour les problèmes 16 à 19.

16. Pour l'entrée de la Figure 9.24, quelle est la sortie de O:013/01 à la Figure 9.25 ?
17. Pour l'entrée de la Figure 9.24, quelle est la sortie de O:013/02 à la Figure 9.25 ?
18. Pour l'entrée de la Figure 9.24, quelle est la sortie de O:013/03 à la Figure 9.25 ?
19. Pour l'entrée de la Figure 9.24, quelle est la sortie de O:013/04 à la Figure 9.25 ?
20. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Dans le schéma à contacts de la Figure 9.26, le temporisateur est fixé à 5 s. Lorsqu'il existe une entrée sur E1 :
    (i) la sortie S1 est activée immédiatement ;
    (ii) la sortie S2 est activée après 5 s.

Copyright © 2015 Dunod.

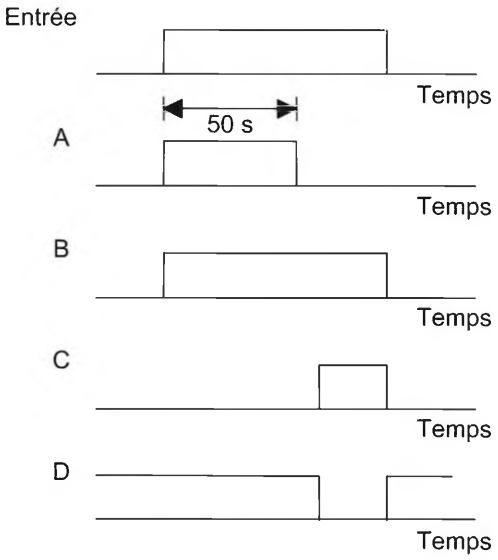


Figure 9.25 – Diagramme pour les problèmes 16 à 19.

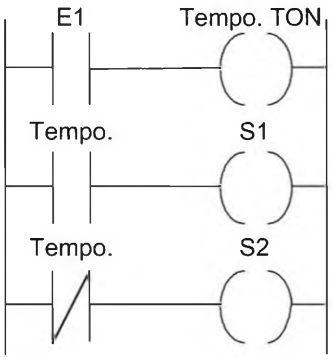


Figure 9.26 – Diagramme pour le problème 20.

Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?

A. (i) V (ii) V
B. (i) V (ii) F
C. (i) F (ii) V
D. (i) F (ii) F

21. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Dans le schéma à contacts de la Figure 9.27, les deux temporisateurs sont fixés à 10 s. Lorsqu'il existe une entrée sur E1 :

(i) la sortie S1 est activée 20 s après l'entrée ;
(ii) la sortie S1 est activée pendant une durée de 20 s.

Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?

A. (i) V (ii) V
B. (i) V (ii) F
C. (i) F (ii) V
D. (i) F (ii) F

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

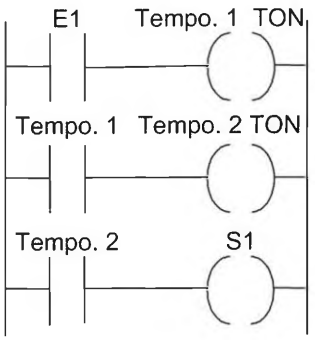


Figure 9.27 – Diagramme pour le problème 21.

22. Concevez les programmes en langage à contacts des systèmes qui mettent en œuvre les tâches suivantes :
    (a) Activer une sortie 5 s après arrivée d'une entrée et conserver la sortie active pendant la durée de cette entrée.
    (b) Activer une sortie pour la durée de l'entrée et la conserver active pendant 5 s supplémentaires.
    (c) Activer une entrée pendant 5 s après le début du signal d'entrée.
    (d) Démarrer une machine lorsque l'interrupteur B est fermé dans un intervalle de 0,5 s après la fermeture de l'interrupteur A ; dans le cas contraire, la machine n'est pas démarrée.

9.9 Recherches complémentaires

1. Recherchez les temporisateurs proposés par une gamme d'API.

Copyright © 2015 Dunod.

10 • COMPTEURS

Dans les API, les compteurs sont disponibles sous forme d'éléments intégrés et permettent de compter le nombre d'occurrences des signaux d'entrée. Cela peut servir à compter, par exemple, des articles lorsqu'ils passent sur une bande transporteuse, le nombre de révolutions d'un axe ou le nombre de personnes qui passent par une porte. Ce chapitre explique comment programmer ces compteurs.

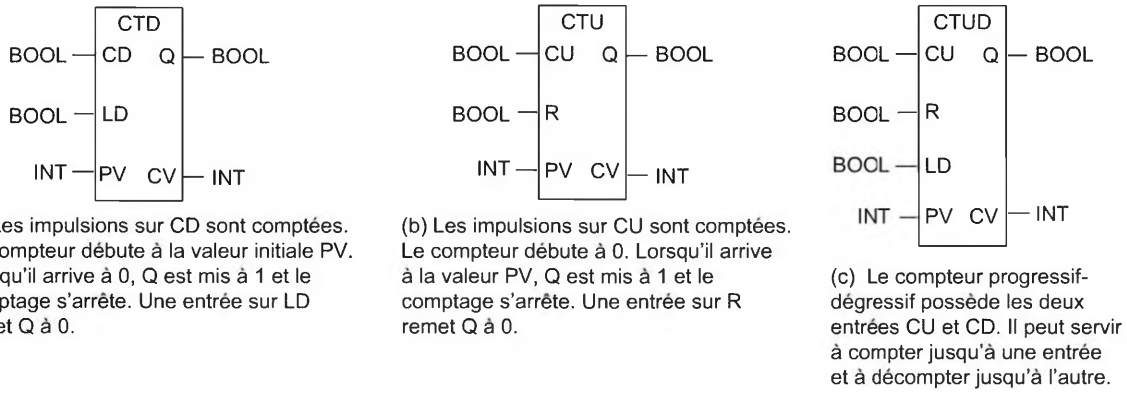
10.1 Types de compteurs

Un compteur est initialisé à une valeur de présélection et, lorsque ce nombre d'impulsions d'entrée a été reçu, il actionne ses contacts. Les contacts normalement ouverts sont fermés, les contacts normalement fermés sont ouverts.

Il existe deux types de compteurs : les compteurs dégressifs et les compteurs progressifs. Un *compteur dégressif*, ou compteur décrémental, décompte à partir de la valeur présélectionnée jusqu'à zéro. Autrement dit, des événements sont soustraits de la valeur fixée. Lorsque le compteur atteint zéro, ses contacts changent d'état. La majorité des API disposent de compteurs dégressifs. Un *compteur progressif*, ou compteur incrémental, compte à partir de zéro jusqu'à la valeur présélectionnée. Autrement dit, des événements sont ajoutés jusqu'à ce que le cumul atteigne la présélection. Lorsque le compteur atteint la valeur présélectionnée, ses contacts changent d'état. Certains API offrent dans une même fonction une possibilité de comptage progressif et dégressif. La Figure 10.1 présente les symboles des compteurs tels que définis par la CEI.

Les fabricants d'API traitent les compteurs de manière légèrement différente. Certains comptent de manière décrémentale (CTD) ou incrémentale (CTU), se remettent à zéro et sont traités comme des bobines de relais, c'est-à-dire une sortie sur une ligne. Dans ce cas, il est possible de considérer que les compteurs sont constitués de deux éléments de base : une bobine pour compter les impulsions d'entrée et une autre pour mettre à zéro le compteur, ses contacts étant utilisés sur d'autres lignes. La Figure 10.2a illustre cette façon de voir, choisie par Mitsubishi. D'autres fabricants traitent le compteur comme un bloc intermédiaire dans une ligne, à partir duquel des signaux émanent lorsque la valeur présélectionnée est atteinte. La Figure 10.2b illustre ce modèle, retenu par Siemens et d'autres.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

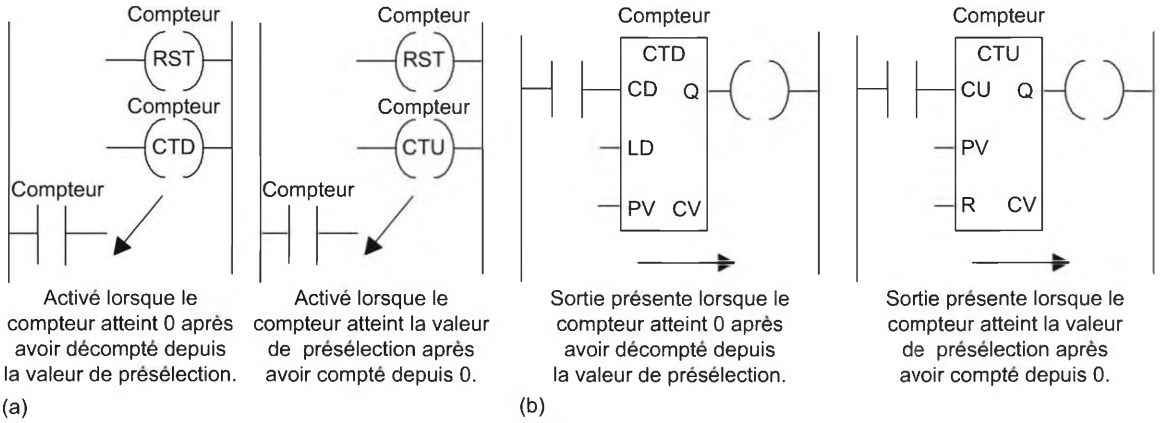


(a) Les impulsions sur CD sont comptées. Le compteur débute à la valeur initiale PV. Lorsqu'il arrive à 0, Q est mis à 1 et le comptage s'arrête. Une entrée sur LD remet Q à 0.

(b) Les impulsions sur CU sont comptées. Le compteur débute à 0. Lorsqu'il arrive à la valeur PV, Q est mis à 1 et le comptage s'arrête. Une entrée sur R remet Q à 0.

(c) Le compteur progressif-dégressif possède les deux entrées CU et CD. Il peut servir à compter jusqu'à une entrée et à décompter jusqu'à l'autre.

Figure 10.1 – Symboles de la CEI pour les compteurs : (a) compteur dégressif, (b) compteur progressif et (c) compteur progressif-dégressif.



Activé lorsque le compteur atteint 0 après avoir décompté depuis la valeur de présélection.

Activé lorsque le compteur atteint la valeur de présélection après avoir compté depuis 0.

(a)

Sortie présente lorsque le compteur atteint 0 après avoir décompté depuis la valeur de présélection.

Sortie présente lorsque le compteur atteint la valeur de présélection après avoir compté depuis 0.

(b)

Figure 10.2 – Différentes représentations d'un compteur. (a) Compteur sous forme de bobine avec des contacts dans une autre ligne, RST est la mise à zéro. (b) Représentation conforme à la CEI 61131-3 en tant qu'élément dans une ligne.

10.2 Programmation

La Figure 10.3 présente un circuit de comptage de base. Lors d'un front montant sur l'entrée E1, le compteur est mis à zéro. Lors d'un front montant sur l'entrée E2, le compteur se déclenche. S'il est configuré pour, par exemple, dix impulsions, alors, après dix impulsions sur l'entrée E2, c'est-à-dire dix transitions de 0 à 1, les contacts du compteur se ferment et la sortie S1 est activée. Si, au cours du comptage, une entrée est présente sur E1, le compteur est réinitialisé et le cycle recommence pour dix impulsions.

La Figure 10.4a montre le programme précédent et sa liste d'instructions pour un API de Mitsubishi et un compteur CTU. Les éléments de mise à zéro et de comptage sont combinés dans une même boîte qui recouvre les deux lignes. Vous pouvez considérer que le rectangle englobe les deux sorties ( ) du compteur de la Figure 10.3. La valeur du compteur est fixée par une instruction K dans le programme. La Figure 10.4b présente le même programme pour un API de Siemens.

Copyright © 2015 Dunod.

Dans ce schéma à contacts, le compteur est considéré comme un élément de retard sur la ligne de sortie (voir Figure 10.1b). Le compteur est remis à zéro par une entrée sur I0.1 et compte les impulsions présentes sur l'entrée I0.0. Le libellé CU indique qu'il s'agit d'un compteur progressif, tandis que CD désigne un compteur dégressif. La valeur présélectionnée du compteur est indiquée par LKC. La Figure 10.4c donne le programme pour un API de Toshiba.

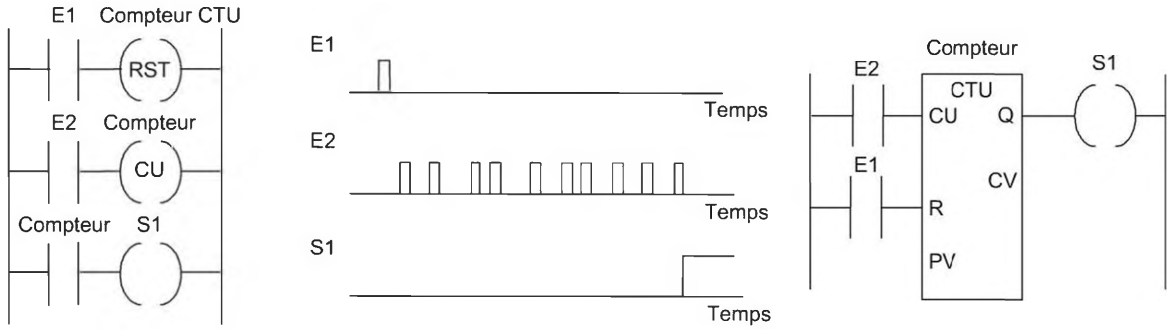


Figure 10.3 – Programme de comptage de base.

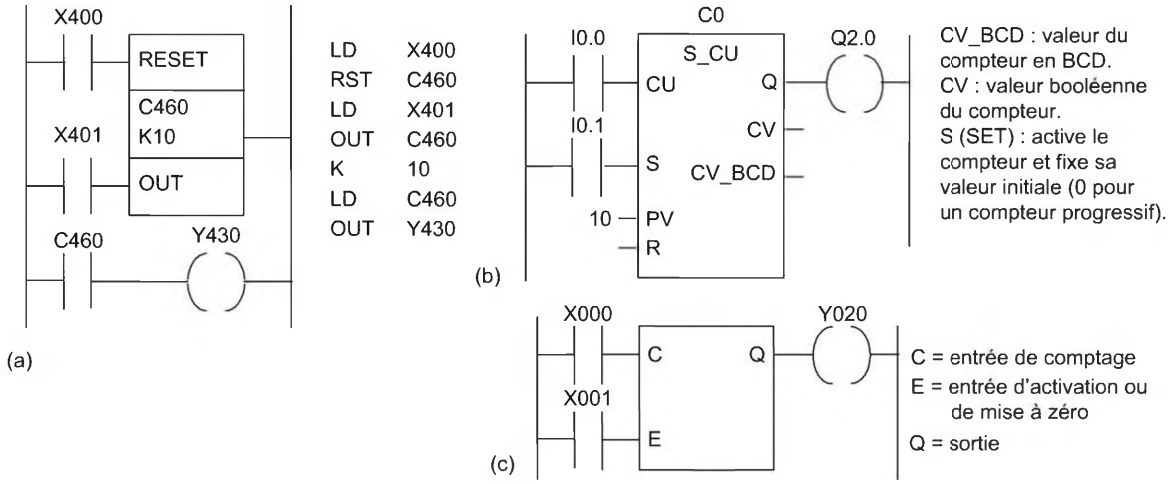


Figure 10.4 – (a) Programme pour un API de Mitsubishi, (b) de Siemens et (c) de Toshiba.

Les Figures 10.5a et 10.5b montrent le programme pour un API d'Allen-Bradley avec des compteurs progressifs et dégressifs. Les termes suivants sont associés à ce type de compteurs :

- La valeur de présélection (PRE) correspond à la valeur que le compteur doit cumuler avant que sa sortie ne soit mise à 1.
- La valeur cumulée (ACC) correspond au comptage du nombre de passages de 0 à 1 sur la ligne du compteur.
- Le bit d'activation du comptage (CU) est à 1 lorsque la ligne du compteur progressif est à 1, et à 0 lorsque la ligne est à 0.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

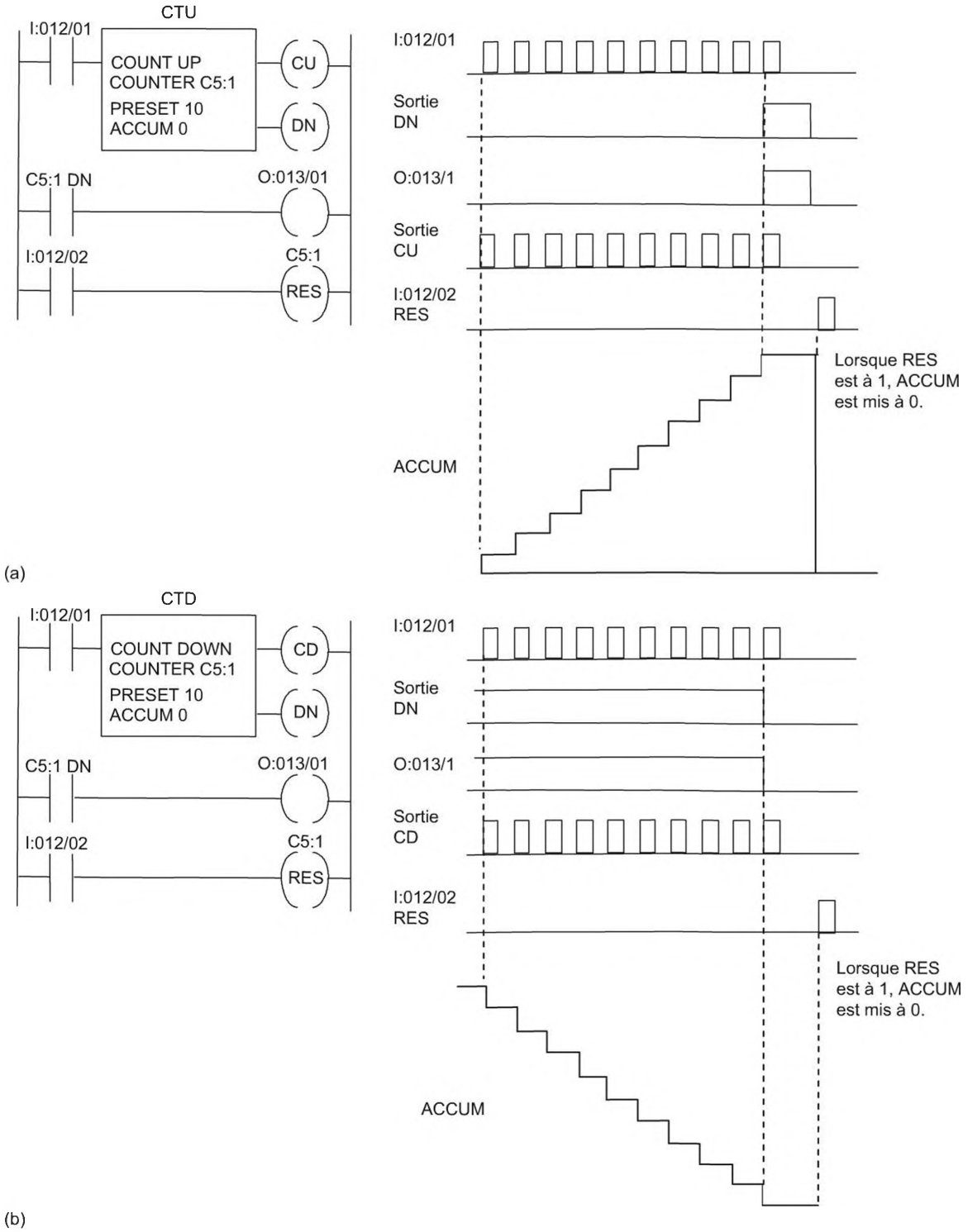


Figure 10.5 – Compteurs Allen-Bradley : (a) progressif et (b) dégressif.

- Le bit d'activation du décomptage (CD) est à 1 lorsque la ligne du compteur dégressif est à 1, et à 0 lorsque la ligne est à 0.
- Le bit de fin (DN) est à 1 pour les deux compteurs lorsque la valeur ACC est égale ou supérieure à la valeur PRE, et à 0 lorsqu'elle est inférieure.

Copyright © 2015 Dunod.

- Le bit de dépassement supérieur (OV) est à 1 lorsque le compteur progressif dépasse la valeur positive maximale.
- Le bit de dépassement inférieur (UN) est à 1 lorsque le compteur passe en dessous de la valeur négative maximale.
- La mise à zéro (RES) remet à zéro l'accumulateur du compteur. Tant que RES est à 1, ACC et tous les bits de sortie sont maintenus à 0. Lorsque RES est à 0, le compteur est en mesure de compter.

Pour garantir que les impulsions d'entrée d'un compteur soient de courte durée, vous pouvez utiliser le schéma à contacts de la Figure 10.6. Lorsque l'entrée sur E1 est présente, le relais interne RI1 est activé. Lors de l'analyse de la ligne suivante, c'est-à-dire un court moment après, le relais interne RI2 est activé. Lorsque RI2 est activé, il désactive l'entrée sur RI1. Ainsi, RI1 produit une impulsion de courte durée, qui sert ensuite d'entrée à un compteur.

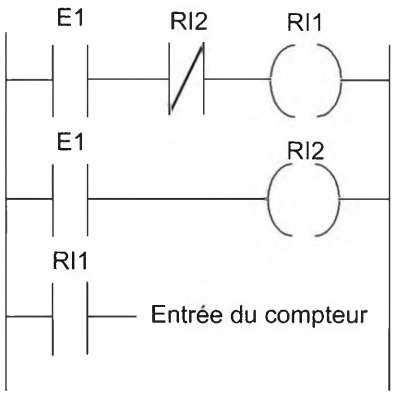


Figure 10.6 – Impulsions de courte durée pour un compteur.

10.2.1 Mise en application d'un compteur

Afin d'illustrer l'utilisation d'un compteur, examinons le traitement des articles convoyés par une bande transporteuse. Le passage d'un article en un point particulier est enregistré par l'interruption d'un faisceau lumineux sur une cellule photoélectrique. Après un nombre défini, un signal est envoyé pour informer que ce compte a été atteint et la bande transporteuse est arrêtée. La Figure 10.7a présente les éléments de base d'un programme Siemens qui met en œuvre ce fonctionnement. Un signal de mise à zéro provoque la réinitialisation du compteur et la reprise du comptage. Un signal d'initialisation permet d'activer le compteur. La Figure 10.7b présente les éléments de base du programme équivalent pour un API d'Allen-Bradley. Lorsque le compteur atteint la valeur présélectionnée, le bit de fin est mis à 1. La sortie O:013/01 se produit alors, les contacts correspondants s'ouvrent et la bande transporteuse est arrêtée.

Comme autre illustration de l'emploi d'un compteur, étudions la commande d'une machine qui doit diriger six boîtes de conserve vers un chemin pour leur emballage dans un carton et douze boîtes vers un autre chemin pour leur emballage dans un autre carton (voir Figure 10.8a). Un déflecteur est commandé par un

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

capteur photoélectrique, qui produit une sortie chaque fois qu'une boîte passe devant lui. Les impulsions produites par le capteur doivent donc être comptées et utilisées pour commander le déflecteur. La Figure 10.8b présente le schéma à contacts qui peut être utilisé, avec la notation de Mitsubishi.

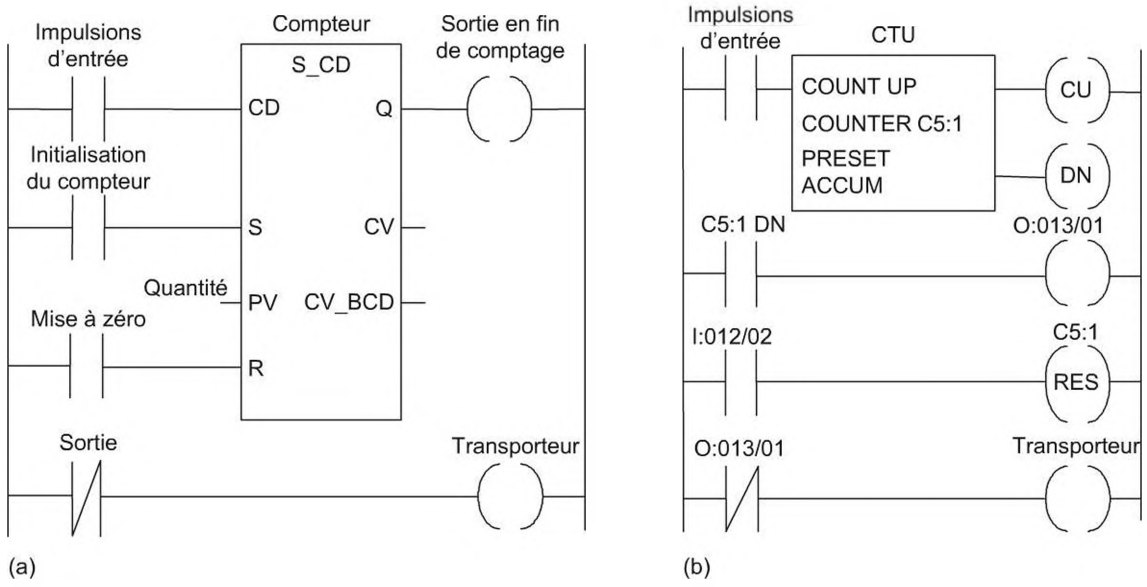


Figure 10.7 – Programmes de comptage (a) Siemens et (b) Allen-Bradley.

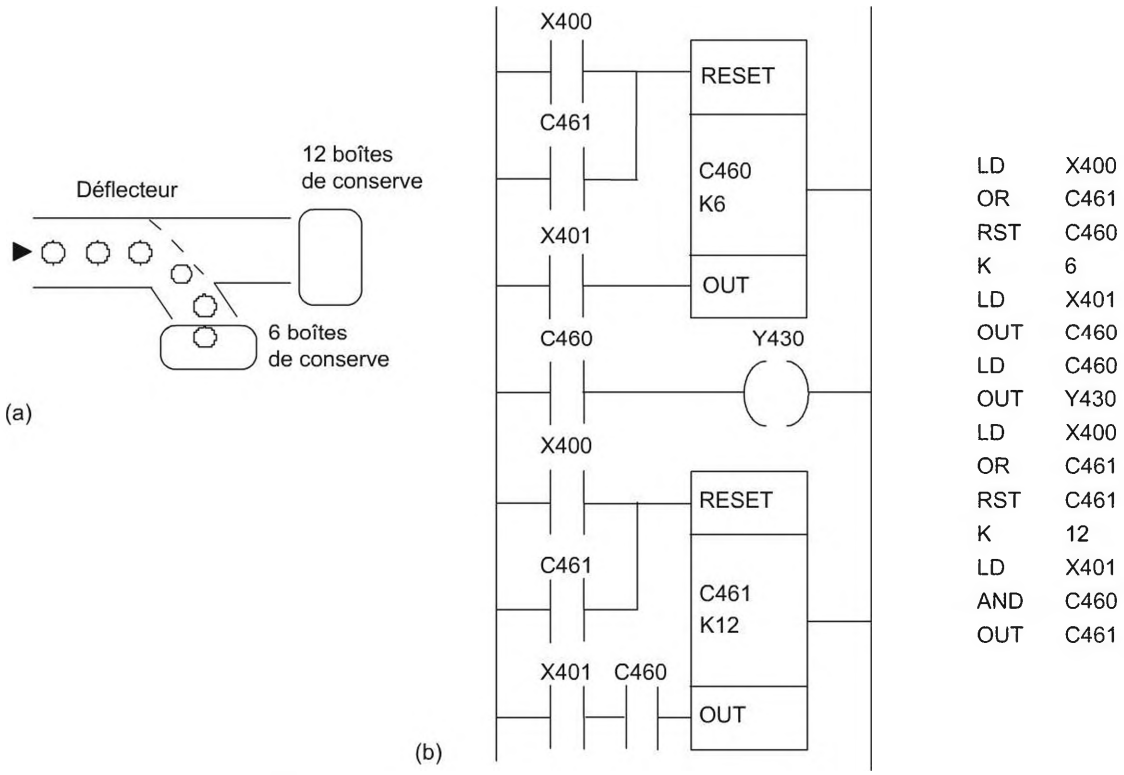


Figure 10.8 – (a) Tâche de comptage et (b) schéma à contacts correspondant.

Copyright © 2015 Dunod.

Une impulsion d'entrée sur X400 met à zéro les deux compteurs. L'entrée sur X400 pourrait être un bouton-poussoir qui permet de démarrer la bande transporteuse. L'entrée comptée est X401. Elle peut provenir d'un capteur photoélectrique qui détecte la présence des boîtes de conserve passant sur la bande transporteuse. C460 commence à compter après la fermeture momentanée de X400. Lorsque C460 a compté six articles, il ferme ses contacts et produit une sortie sur Y430. Il peut s'agir d'un solénoïde qui active un déflecteur afin de dévier les articles vers un carton ou un autre. Le déflecteur peut ainsi se trouver dans une position qui permet de dévier les six premières boîtes de conserve vers le carton de six boîtes. Ensuite, le déflecteur est déplacé pour que les boîtes se dirigent vers le carton de douze boîtes. Lorsque C460 arrête de compter, il ferme ses contacts et permet à C461 de débuter son travail. C461 compte douze impulsions sur X401, puis ferme ses contacts. Cela remet à zéro les deux compteurs et le processus se répète.

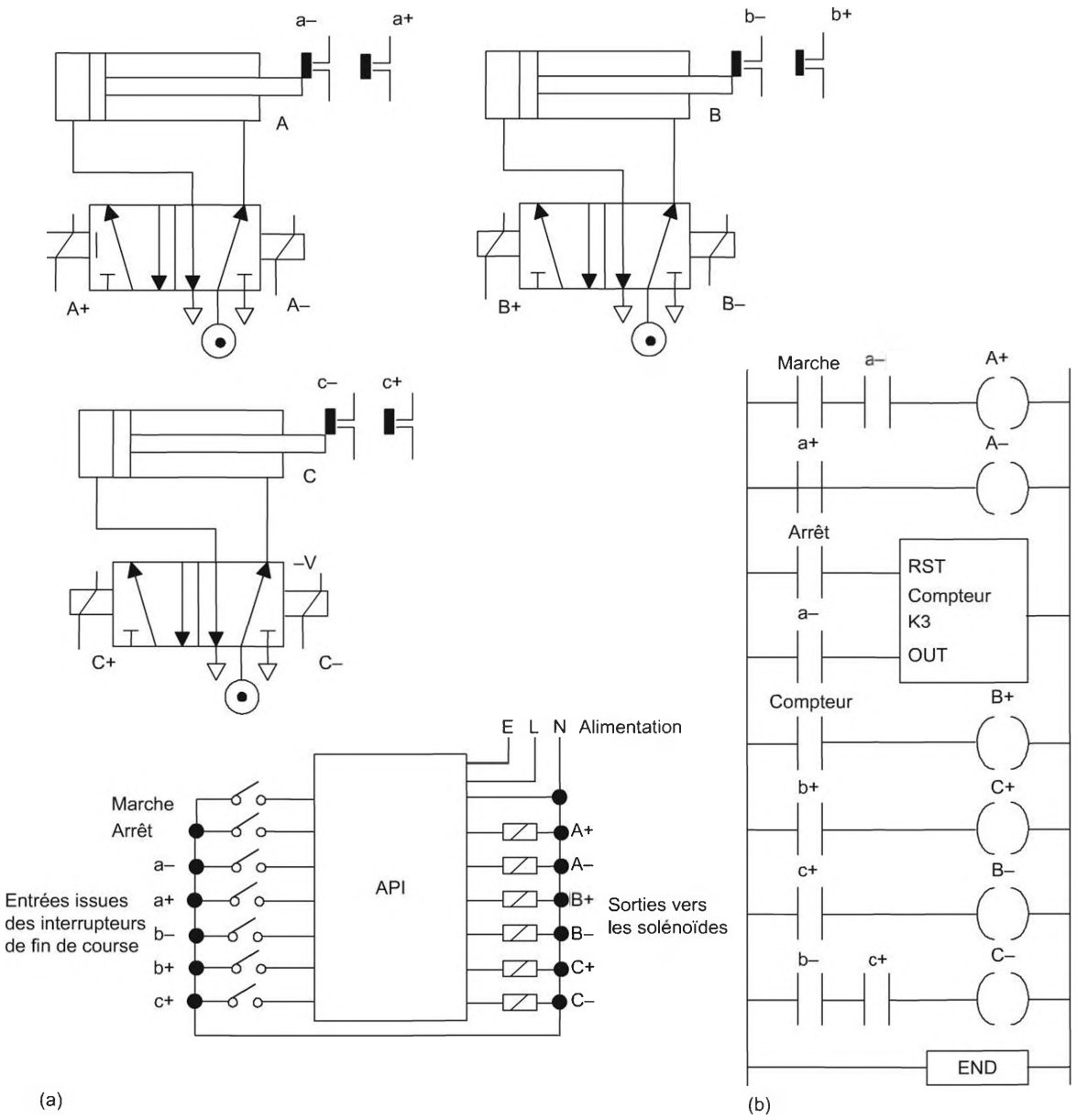


Figure 10.9 – (a) Système à trois vérins et (b) le programme.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

Les compteurs peuvent également servir à garantir le nombre de répétitions d'une partie d'une séquence. Cette fonction est illustrée par le programme suivant. Il met en œuvre un système à trois pistons commandés par des solénoïdes doubles (voir Figure 10.9a) de manière à obtenir la séquence A+, A–, A+, A–, A+, A–, B+, C+, B–, C–. La séquence A+, A– est répétée trois fois avant que la séquence B+, C+, B–, C– ne soit déclenchée. Nous pouvons employer un compteur pour assurer cette répétition. La Figure 10.9b présente un programme possible. Le compteur n'autorise B+ qu'après avoir reçu trois entrées qui correspondent à trois signaux a–.

10.3 Compteurs progressifs-dégressifs

Il est possible de programmer des compteurs progressifs et dégressifs ensemble. Les *compteurs progressifs-dégressifs* sont disponibles sous forme d'entités autonomes (voir Figure 10.1 pour le symbole de la CEI). Étudions le problème de comptage des articles qui arrivent sur une bande transporteuse et qui en sortent, ou bien le comptage des voitures qui pénètrent dans un parking et qui en sortent. Une sortie doit être activée lorsque le nombre d'articles ou de voitures qui entrent est supérieur au nombre d'articles ou de voitures qui sortent. Autrement dit, le nombre d'articles ou de voitures a atteint une valeur de « saturation ». La sortie peut commander l'allumage d'un panneau « Complet ». Supposons que nous utilisions le compteur progressif pour les articles qui entrent et le compteur dégressif pour ceux qui sortent. La Figure 10.10a montre la forme de base d'un schéma à contacts pour une telle application. Lorsqu'un article arrive, il produit une impulsion sur l'entrée E1, ce qui incrémente le compteur. Ainsi, chaque article qui arrive augmente de 1 la valeur cumulée. Lorsque un article sort, il produit une entrée sur E2, ce qui diminue de 1 le nombre. Ainsi, chaque article qui sort diminue de 1 la valeur cumulée. Lorsque la valeur cumulée atteint la valeur de présélection, la sortie S1 est activée. La Figure 10.10b montre la mise en œuvre du système précédent pour un API de Siemens, ainsi que la liste d'instructions correspondante. CU est l'entrée de comptage progressif, CD, l'entrée de comptage dégressif. R est la mise à zéro. La valeur de présélection est chargée *via* F0.0, qui est un relais interne.

La Figure 10.11 montre l'implémentation de ce programme dans un API d'Allen-Bradley.

10.4 Temporisateurs et compteurs

Un temporisateur type peut compter jusqu'à seize bits de données, ce qui correspond à 32 767 unités de temps. Par conséquent, si la base de temps est égale à 1 s, le temporisateur peut traiter une durée maximale de 546 minutes, ou 9,1 heures. Si la base de temps est égale à 0,1 s, la durée maximale devient 54,6 minutes, c'est-à-dire à peine une heure. En combinant un temporisateur et un compteur, il est possible d'obtenir des durées plus longues. La Figure 10.12 illustre ce principe avec un programme Allen-Bradley. Si le temporisateur est configuré avec une base

Copyright © 2015 Dunod.

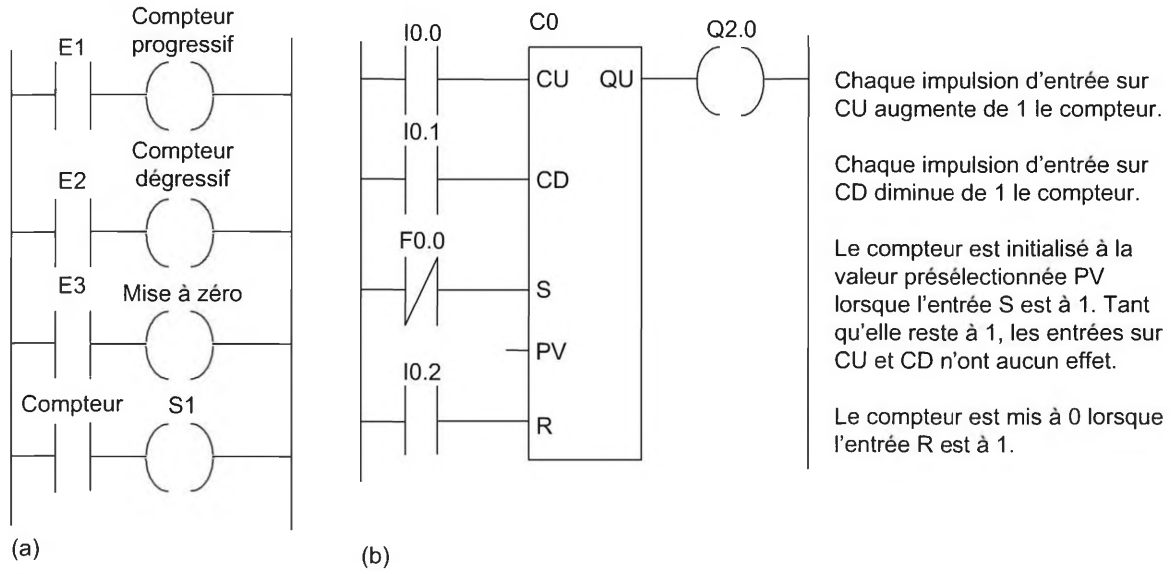


Figure 10.10 – (a) Compteurs progressif et dégressif et (b) programme Siemens.

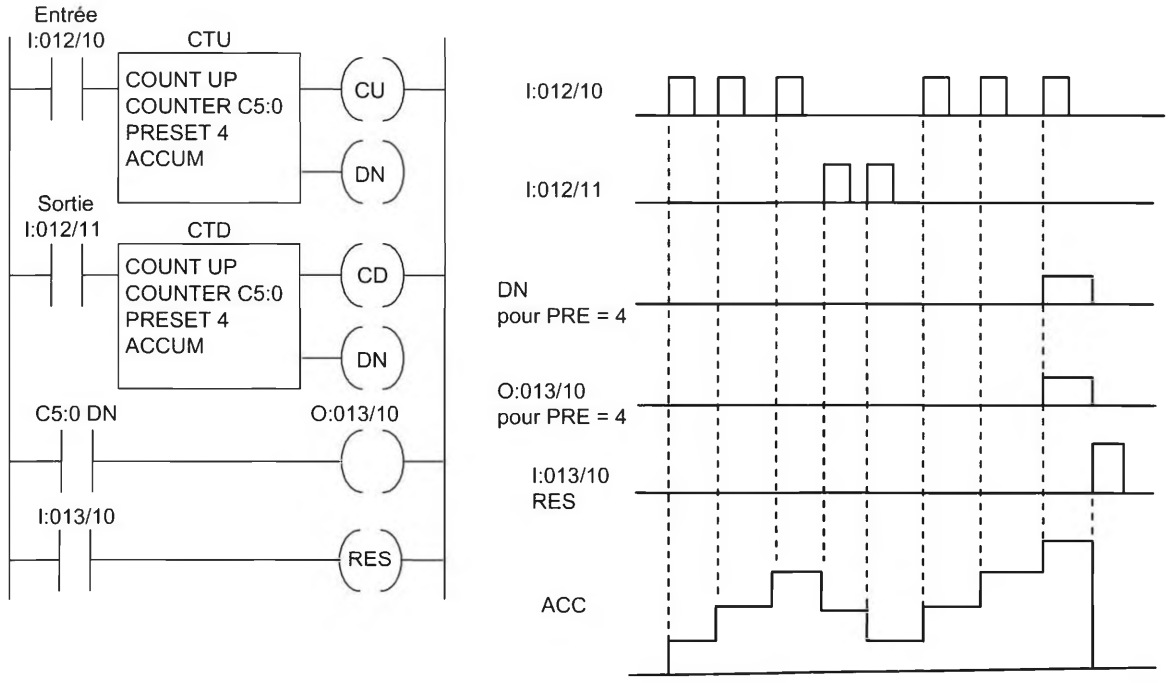


Figure 10.11 – Programme Allen-Bradley.

de temps de 1 s et une valeur de présélection de 3 600, il peut compter jusqu'à 1 heure. Lorsque l'entrée I:012/01 est activée, le temporisateur commence son chronométrage par incrément d'une seconde. Lorsque la durée atteint la valeur présélectionnée d'une heure, le bit DN est mis à 1 et le compteur est incrémenté de 1. Le passage à 1 du bit DN remet également à zéro le temporisateur, qui reprend son chronométrage. Lorsqu'il atteint à nouveau la durée présélectionnée, le bit DN passe à nouveau à 1 et le compteur est incrémenté de 1. Si le compteur est initialisé avec une valeur de présélection égale à 24, le bit DN du compteur est

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

mis à 1 lorsqu'il atteint 24 et la sortie O:013/01 est alors activée. Nous disposons donc d'un temporisateur capable de compter les secondes pendant une journée et d'activer un dispositif au bout de 24 heures.

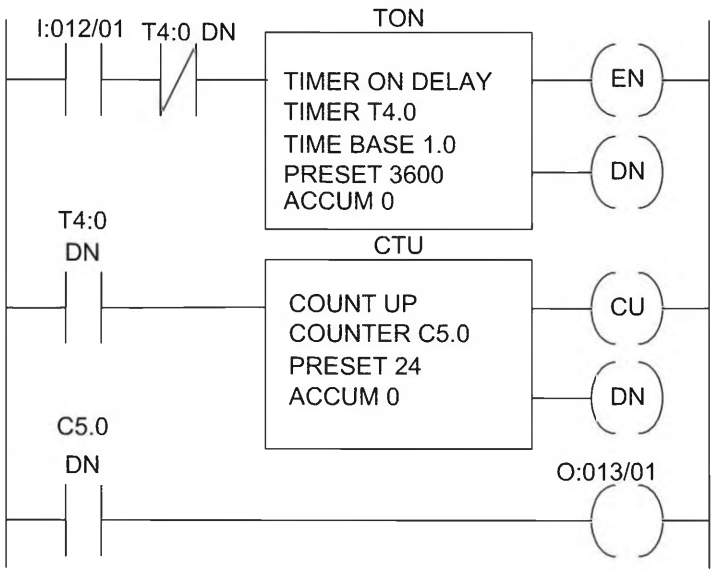


Figure 10.12 – Compteur utilisé pour étendre la plage d'un temporisateur.

10.5 Séquenceurs

Le *séquenceur à tambour* est une forme de compteur utilisé pour un contrôle séquentiel. Il remplace le séquenceur à tambour mécanique employé pour la commande de machines qui répétaient une séquence d'opérations. Ce séquenceur mécanique peut être constitué d'un tambour duquel dépassent des picots (voir Figure 10.13). Lorsque le cylindre tourne, les contacts alignés avec les picots sont fermés au moment où les picots les percutent et s'ouvrent lorsque les picots sont passés. Selon la Figure 10.13, au cours de la première étape, les contacts alignés avec le picot de la sortie 1 sont activés. Ensuite, à l'étape 2, les contacts qui correspondent au picot de la sortie 3 sont activés. Puis, à l'étape 3, les contacts associés au picot de la sortie 2 sont activés. Et ainsi de suite. Différentes sorties peuvent être commandées par des picots placés à différentes distances sur le tambour. Une autre forme de séquenceur utilise des cames sur un même axe. Le profil d'une came permet d'activer ou de désactiver des contacts.

Le séquenceur d'un API comprend un compteur maître dont les différentes présélections correspondent aux étapes. Ainsi, lorsque le compteur avance, chaque valeur présélectionnée atteinte peut servir à commander des sorties. Chaque étape de la séquence du compteur concerne une sortie ou un groupe de sorties. Les sorties sont des relais internes, qui sont à leur tour utilisés pour commander des dispositifs de sortie externes.

Prenons comme exemple le séquencement suivant. La sortie 1 doit être activée 5 s après le démarrage et doit le rester jusqu'aux 10 s. La sortie 2 doit être activée à

Copyright © 2015 Dunod.

10 s et le rester jusqu'aux 20 s. La sortie 3 doit être activée à 15 s et le rester jusqu'aux 25 s. Et ainsi de suite. Nous pouvons représenter ces contraintes par un diagramme qui montre le séquencement requis (voir Figure 10.14).

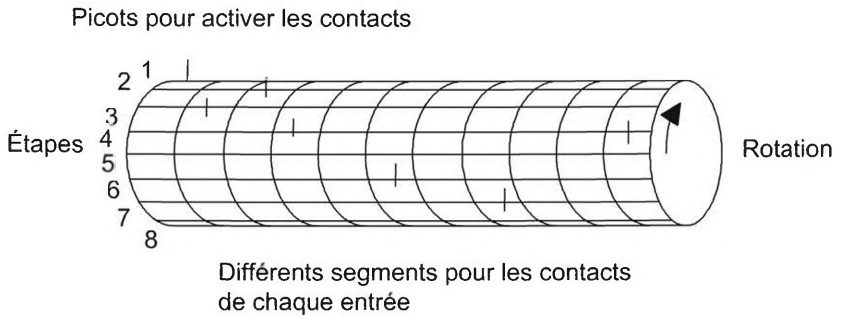


Figure 10.13 – Séquenceur à tambour.

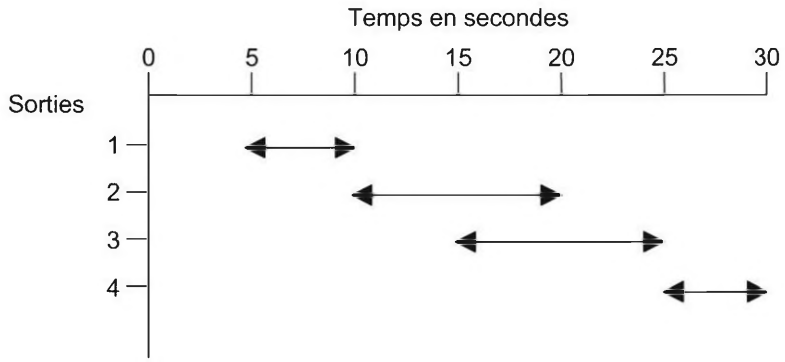


Figure 10.14 – Diagramme de temps.

Ce diagramme de temps peut être transformé en une contrainte de séquencement du tambour. Si chaque étape prend 5 s, nous obtenons le diagramme d'exigences donné au Tableau 10.1. Ainsi, à l'étape 1, la sortie 1 doit être activée et le rester jusqu'à l'étape 2. Ensuite, à l'étape 2, la sortie 2 doit être activée et le rester jusqu'à l'étape 4. À l'étape 3, la sortie 3 doit être activée et le rester jusqu'à l'étape 5. À l'étape 5, la sortie 4 doit être activée et le rester jusqu'à l'étape 6.

Tableau 10.1 Séquencement requis

Étape	Temps (s)	Sortie 1	Sortie 2	Sortie 3	Sortie 4
0	0	0	0	0	0
1	5	1	0	0	0
2	10	0	1	0	0
3	15	0	1	1	0

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

Tableau 10.1 Séquencement requis *(suite)*

Étape	Temps (s)	Sortie 1	Sortie 2	Sortie 3	Sortie 4
4	20	0	0	1	0
5	25	0	0	1	1
6	30	0	0	0	0

Pour les API de Toshiba, le séquenceur est configuré en activant le bloc fonctionnel STIZ (*Step Sequence Initialize*) R500 (voir Figure 10.15). Cela place le programme à l'étape 1 et active R501. Ce relais active ensuite la sortie Y020. L'étape suivante correspond à l'activation de R502. Ce relais active la sortie Y021 et un temporisateur à l'enclenchement pour que R503 ne soit pas activé tant que la temporisation n'est pas écoulée. Puis, R503 active Y022, ainsi que l'étape suivante de la séquence.

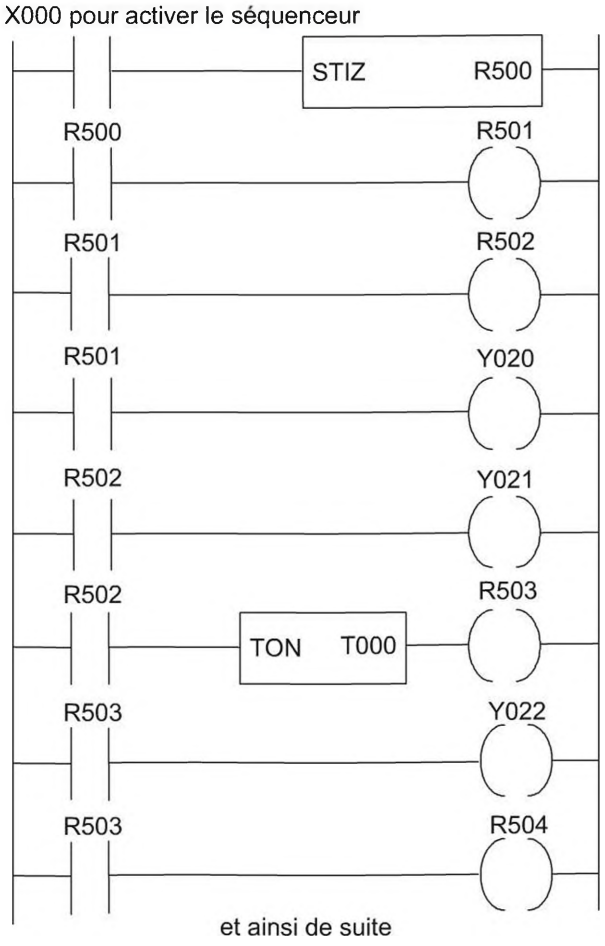


Figure 10.15 – Séquenceur avec un API de Toshiba.

Copyright © 2015 Dunod.

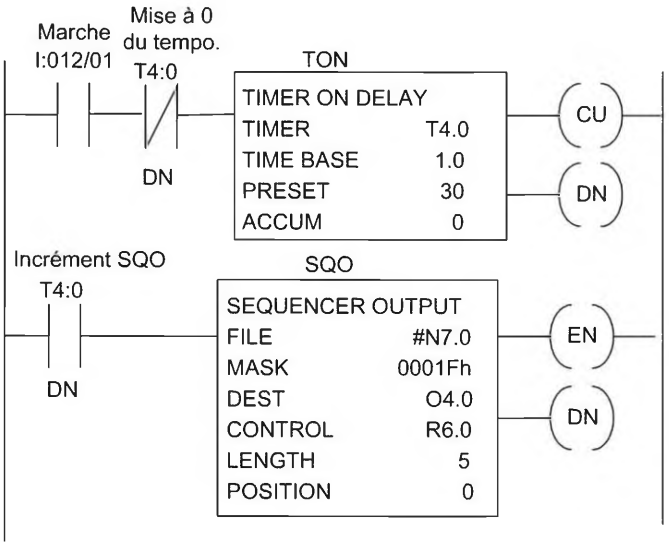
Avec un API d'Allen-Bradley, le séquenceur est programmé à l'aide d'une suite de mots binaires, qui indiquent les sorties requises. Allen-Bradley utilise le terme *sortie séquenceur* (SQO) pour l'instruction de sortie qui utilise un fichier ou un tableau pour commander différents dispositifs de sortie. Nous pouvons placer la suite de mots binaires suivante dans un fichier pour chaque étape du séquenceur en utilisant l'appareil de programmation. Pour six étapes et cinq sorties, à l'étape 0 toutes les sorties sont à 0. À l'étape 1, la sortie 1 sera à 1 et les autres à 0. À l'étape 2, la sortie 2 sera à 1 et les autres sorties à 0, etc.

Voici la terminologie employée avec le séquenceur :

- *File* pour l'adresse de départ des registres dans le fichier du séquenceur, l'indicateur de fichier marqué # étant utilisé pour cette adresse. Nous pouvons donc avoir #N7.0 comme fichier pour l'étape 0, et #N7.1 pour l'étape 1, #N7.2 pour l'étape 2, etc.
- *Mask* pour le motif de bits au travers duquel l'instruction du séquenceur déplace les données source vers l'adresse de destination. Un h est ajouté à la fin du paramètre pour indiquer que le masque est donné sous forme hexadécimale, tandis qu'un B indique une notation binaire. Sans autre indication, il s'agit d'une valeur décimale. Avec le masque 0001Fh, ou 0000000000011111 en binaire, seuls les cinq premiers bits du séquenceur sont passés à l'adresse de destination, tous les autres étant occultés, c'est-à-dire non transmis à la destination. Avec le masque 1110011011111110, les bits 1, 9, 12 et 13 sont à 0 et les bits correspondants dans le mot du séquenceur ne sont donc pas transmis à la destination. Ces motifs de bits permettent de masquer sélectivement des données provenant du fichier du séquenceur sur la sortie concernée.
- *Destination* est l'adresse du mot ou du fichier de sortie vers laquelle l'instruction déplace les données à partir de son fichier séquenceur. Ainsi, O4.0 indiquera la sortie vers laquelle la sortie de l'étape 1 est dirigée.
- *Control* précise l'adresse qui contient les paramètres des informations de contrôle pour l'instruction et des sorties discrètes pour indiquer les résultats et l'état de l'instruction du séquenceur. L'adresse se trouve dans la zone de contrôle R de la mémoire du processeur et le fichier par défaut est le 6.
- *Length* est le nombre d'étapes dans le fichier séquenceur, en commençant à l'étape 1. Puisqu'il existe une étape 0, le nombre total de mots dans le fichier correspond à la valeur de *length* plus 1.
- *Position* est l'emplacement du mot/étape dans le fichier séquenceur utilisé par l'instruction pour déplacer des données. Si cette valeur vaut 0, alors après le premier scan, elle est incrémentée de 1 et le fichier séquenceur est déplacé à ce numéro de position dans le fichier destination, en passant par le masque.

À titre d'illustration, la Figure 10.16 montre un simple schéma à contacts pour un API d'Allen-Bradley qui comprend un séquenceur. Le temporisateur est démarré par une entrée sur I:012/1 et sa présélection est fixée à 30 s. Il est remis à zéro par son bit DN, qui incrémente également l'instruction SQO pour le mot de sortie suivant. Ainsi, le séquenceur est incrémenté toutes les 30 s.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.



O:4.0	Destination		-	-	-	-	-	-	-	-	-	-	-	1	1	1	0	0	
			Sorties masquées											↑	↑	↑	↑	↑	
000Fh	Masque		0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	
#N7.0	Fichier	N7.0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Positions
		N7.1	0	0	1	0	1	0	1	0	1	0	0	0	0	0	1	1	0
		N7.2	0	0	1	0	1	0	1	0	0	0	1	1	0	0	0	1	1
		N7.3	0	1	0	1	0	1	0	1	1	0	1	1	1	1	0	0	2
		N7.4	1	1	0	1	0	1	0	1	0	1	0	0	0	0	1	1	3
		N7.5	1	1	0	0	1	0	0	1	1	1	0	1	0	0	0	0	4
		N7.6	1	0	0	0	1	0	0	1	0	1	0	1	1	0	0	0	5

Figure 10.16 – Séquenceur Allen-Bradley avec déplacement des données provenant d'un fichier vers une sortie. Le pointeur de position est à l'emplacement 2 pour produire la sortie 11100.

10.6 En résumé

Dans les API, les compteurs sont disponibles sous forme d'éléments intégrés et permettent de compter le nombre d'occurrences des signaux d'entrée. Un compteur dégressif, ou compteur décrémental, décompte à partir de la valeur présélectionnée jusqu'à zéro. Autrement dit, des événements sont soustraits de la valeur fixée. Lorsque le compteur atteint zéro, ses contacts changent d'état. Un compteur progressif, ou compteur incrémental, compte à partir de zéro jusqu'à la valeur présélectionnée. Autrement dit, des événements sont ajoutés jusqu'à ce que le cumul atteigne la présélection. Lorsque le compteur atteint la valeur présélectionnée, ses contacts changent d'état. Certains API offrent dans une même fonction une possibilité de comptage progressif et dégressif.

Copyright © 2015 Dunod.

Le séquenceur d'un API comprend un compteur maître dont les différentes présélections correspondent aux étapes. Ainsi, lorsque le compteur avance, chaque valeur présélectionnée atteinte peut être utilisée pour commander des sorties.

10.7 Problèmes

Pour les questions 1 à 19, quatre réponses sont proposées : A, B, C et D. Choisissez la réponse correcte. Les problèmes 1 à 3 font référence à la Figure 10.17, qui présente un schéma à contacts avec un compteur dégressif, deux entrées (E1 et E2) et une sortie (S1).

1. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Pour le schéma à contacts de la Figure 10.17, lorsque le compteur est initialisé à 5, la sortie S1 est présente :
   (i) chaque fois que E1 a été fermée cinq fois ;
   (ii) chaque fois que E2 a été fermée cinq fois.

   Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
   A. (i) V (ii) V
   B. (i) V (ii) F
   C. (i) F (ii) V
   D. (i) F (ii) F

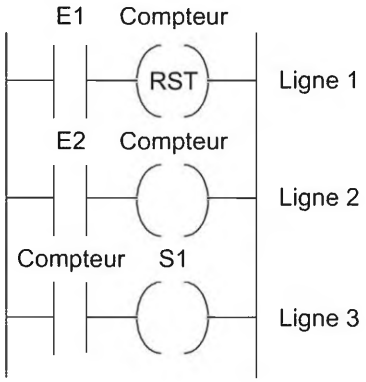


Figure 10.17 – Diagramme pour les problèmes 1 à 3.

2. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Pour le schéma à contacts de la Figure 10.17 :
   (i) la ligne 1 donne la condition nécessaire à la mise à zéro du compteur ;
   (ii) la ligne 2 donne la condition nécessaire à la génération des impulsions à compter.

   Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
   A. (i) V (ii) V
   B. (i) V (ii) F

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

C. (i) F (ii) V
D. (i) F (ii) F

3. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Pour le schéma à contacts de la Figure 10.17, lorsque l'entrée est présente sur E1 :
   (i) les contacts du compteur sur la troisième ligne se ferment ;
   (ii) le compteur est prêt à démarrer le comptage des impulsions sur E2.

   Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
   A. (i) V (ii) V
   B. (i) V (ii) F
   C. (i) F (ii) V
   D. (i) F (ii) F

Les problèmes 4 et 5 font référence à la liste d'instructions suivante, qui implique un compteur dégressif :

```
LD      X400
RST     C460
LD      X401
OUT     C460
K       5
LD      460
OUT     Y430
```

4. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Chaque fois qu'il existe une entrée sur X401 :
   (i) la valeur cumulée par le compteur diminue de 1 ;
   (ii) la sortie est activée.

   Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
   A. (i) V (ii) V
   B. (i) V (ii) F
   C. (i) F (ii) V
   D. (i) F (ii) F

5. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Lorsqu'il existe une entrée sur X400, le compteur :
   (i) se réinitialise à la valeur 5 ;
   (ii) commence à compter à partir de 0.

   Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
   A. (i) V (ii) V
   B. (i) V (ii) F
   C. (i) F (ii) V
   D. (i) F (ii) F

Les problèmes 6 et 7 font référence à la liste d'instructions suivante, qui implique un compteur C0 :

Copyright © 2015 Dunod.

```
A       I0.0
CD      C0
LKC     5
A       I0.1
R       C0
Q       2.00
```

6. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Chaque fois qu'il existe une entrée sur I0.0 :
   (i) la valeur cumulée par le compteur diminue de 1 ;
   (ii) la sortie est activée.

   Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
   A. (i) V (ii) V
   B. (i) V (ii) F
   C. (i) F (ii) V
   D. (i) F (ii) F

7. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Lorsqu'il existe une entrée sur I0.1, le compteur :
   (i) se réinitialise à la valeur 5 ;
   (ii) commence à compter à partir de 0.

   Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
   A. (i) V (ii) V
   B. (i) V (ii) F
   C. (i) F (ii) V
   D. (i) F (ii) F

Les problèmes 8 et 9 font référence à la Figure 10.18, qui montre un compteur dégressif C460 contrôlé par les entrées X400 et X401, et une sortie Y430.

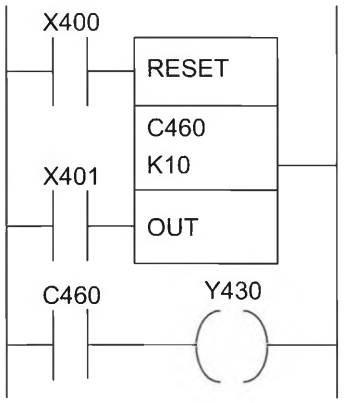


Figure 10.18 – Diagramme pour les problèmes 8 et 9.

8. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Lorsqu'il existe une entrée sur X400, le compteur :
   (i) se réinitialise à la valeur 0 ;
   (ii) commence à compter.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?

A. (i) V (ii) V
B. (i) V (ii) F
C. (i) F (ii) V
D. (i) F (ii) F

9. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Chaque fois qu'il existe une entrée sur X401, le compteur :
   (i) produit une sortie sur Y430 ;
   (ii) diminue la valeur cumulée de 1.

   Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?

   A. (i) V (ii) V
   B. (i) V (ii) F
   C. (i) F (ii) V
   D. (i) F (ii) F

Les problèmes 10 à 12 font référence à la Figure 10.19, qui montre un schéma à contacts avec un compteur (C460), deux entrées (X400 et X401), deux relais internes (M100 et M101) et une sortie (Y430).

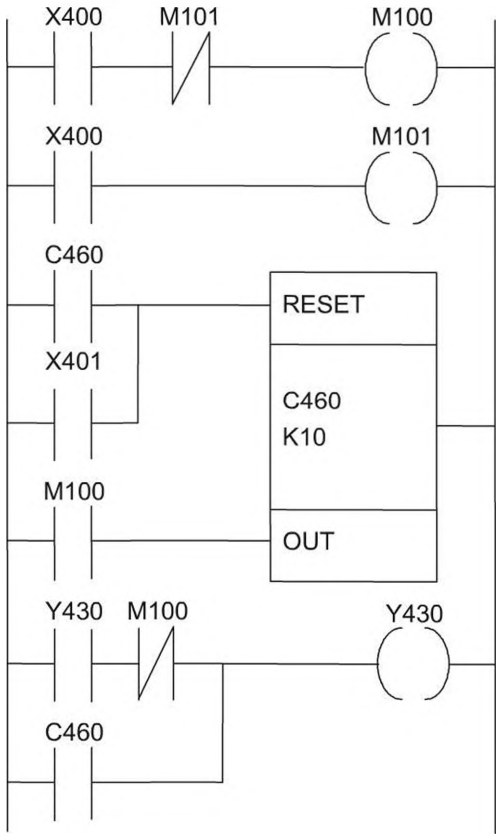


Figure 10.19 – Diagramme pour les problèmes 10 à 12.

Copyright © 2015 Dunod.

10. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). La sortie Y430 :
    (i) est activée par la dixième impulsion sur X400 ;
    (ii) est désactivée au début de la onzième impulsion sur X400.

    Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
    A. (i) V (ii) V
    B. (i) V (ii) F
    C. (i) F (ii) V
    D. (i) F (ii) F

11. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Lorsqu'il existe une entrée sur X400 :
    (i) le relais interne M100 est excité ;
    (ii) le relais interne M101 est excité.

    Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
    A. (i) V (ii) V
    B. (i) V (ii) F
    C. (i) F (ii) V
    D. (i) F (ii) F

12. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Il existe une sortie sur Y430 tant que :
    (i) les contacts C460 sont fermés ;
    (ii) Y430 produit une sortie et M100 est excité.

    Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
    A. (i) V (ii) V
    B. (i) V (ii) F
    C. (i) F (ii) V
    D. (i) F (ii) F

13. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). La Figure 10.20 présente un compteur selon le format de Siemens. Après dix entrées sur I0.0 :
    (i) la lampe s'allume ;
    (ii) le moteur démarre.

    Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
    A. (i) V (ii) V
    B. (i) V (ii) F
    C. (i) F (ii) V
    D. (i) F (ii) F

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

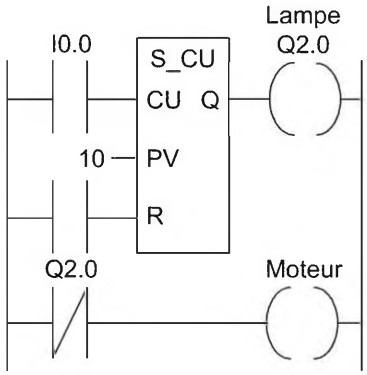


Figure 10.20 – Diagramme pour le problème 13.

Les problèmes 14 et 15 font référence à la Figure 10.21, qui montre un programme Siemens faisant intevenir un compteur progressif-dégressif.

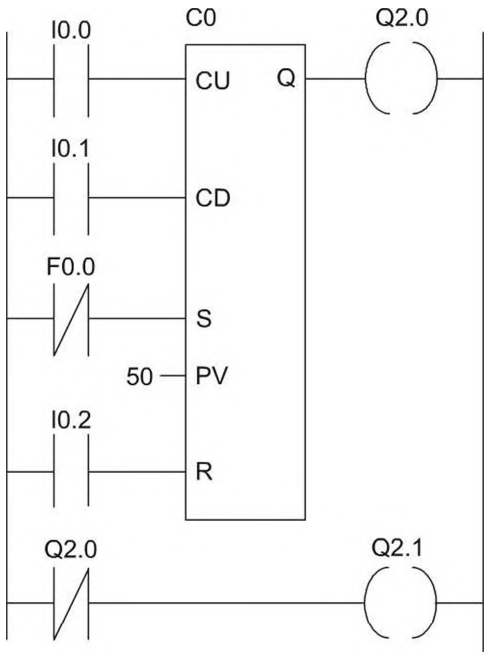


Figure 10.21 – Diagramme pour les problèmes 14 et 15.

14. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). À la Figure 10.21, lorsque la valeur du compteur est inférieure à 50 :

    (i) il existe une sortie sur Q2.0 ;

    (ii) il existe une sortie sur Q2.1.

    Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?

    A. (i) V (ii) V

    B. (i) V (ii) F

    C. (i) F (ii) V

    D. (i) F (ii) F

Copyright © 2015 Dunod.

15. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). À la Figure 10.21, lorsque la valeur du compteur atteint 50 :
    (i) il existe une sortie sur Q2.0 ;
    (ii) il existe une sortie sur Q2.1.

    Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
    A. (i) V (ii) V
    B. (i) V (ii) F
    C. (i) F (ii) V
    D. (i) F (ii) F

Les problèmes 16 et 17 font référence à la Figure 10.22, qui montre un programme Allen-Bradley avec un compteur progressif.

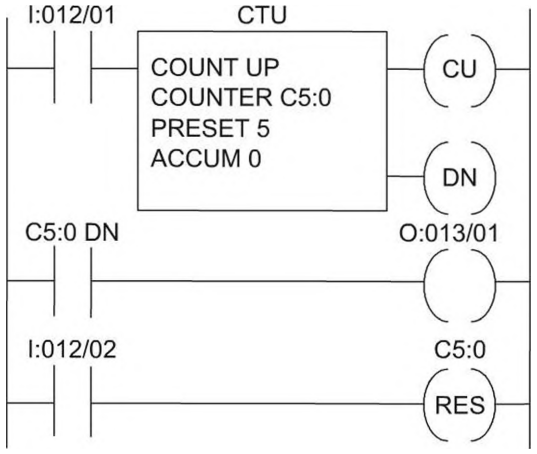


Figure 10.22 – Diagramme pour les problèmes 16 et 17.

16. Avec le programme de la Figure 10.22, le compteur est réinitialisé lorsque :
    A. Sa valeur atteint 5
    B. Sa valeur dépasse 5
    C. Il existe une entrée sur I:012/01
    D. Il existe une entrée sur I:012/02.

17. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Avec le programme de la Figure 10.22, il existe une sortie sur O:013/01 lorsque :
    (i) il existe une entrée sur I:012/01 ;
    (ii) il existe une sortie à partir du bit de fin de comptage DN.

    Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
    A. (i) V (ii) V
    B. (i) V (ii) F
    C. (i) F (ii) V
    D. (i) F (ii) F

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

Les problèmes 18 et 19 font référence à la Figure 10.23, qui montre un programme Allen-Bradley avec un compteur progressif.

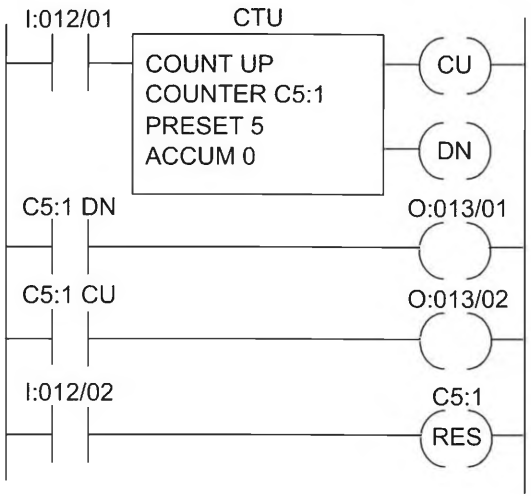


Figure 10.23 – Diagramme pour les problèmes 18 et 19.

18. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). À la Figure 10.23, lorsqu'il existe une seule impulsion d'entrée sur I:012/01 :
    (i) la sortie O:013/01 est activée ;
    (ii) la sortie O:013/02 est activée.

    Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
    A. (i) V (ii) V
    B. (i) V (ii) F
    C. (i) F (ii) V
    D. (i) F (ii) F

19. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). À la Figure 10.23, lors de la cinquième impulsion d'entrée sur I:012/01 :
    (i) la sortie O:013/01 est activée ;
    (ii) la sortie O:013/02 est activée.

    Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
    A. (i) V (ii) V
    B. (i) V (ii) F
    C. (i) F (ii) V
    D. (i) F (ii) F

20. Concevez les programmes en langage à contacts des systèmes qui mettent en œuvre les tâches suivantes :
    (a) Activer une sortie après qu'un capteur photoélectrique a produit dix signaux d'entrée impulsionnels qui correspondent à la détection du passage de dix objets devant le capteur.

Copyright © 2015 Dunod.

(b) Activer une sortie lorsque le nombre de personnes dans un magasin atteint 100, alors que des personnes entrent et sortent continuellement du magasin.
(c) Allumer un témoin lumineux rouge lorsque le compteur est inférieur à 5 et un témoin vert lorsqu'il est égal ou supérieur à 5.
(d) Compter dix objets qui passent sur une bande transporteuse et fermer un déflecteur lorsque ce nombre d'objets ont été déviés dans une chute, en ajoutant une temporisation de 5 s entre le dixième objet compté et la fermeture du déflecteur.
(e) Déterminer le nombre d'articles présents sur une bande transporteuse à un moment donné en comptant ceux qui arrivent sur la bande et ceux qui en sortent, et générer un signal de sortie lorsque ce nombre atteint 100.

10.8 Recherches complémentaires

1. Recherchez les compteurs proposés par une gamme d'API.
2. Sélectionnez, à partir des fiches techniques d'un fabricant, les capteurs et un API qui permettent de contrôler le comptage d'objets opaques convoyés par une bande transporteuse.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

Copyright © 2015 Dunod.

11 • REGISTRES À DÉCALAGE

Un *registre* est un dispositif électronique dans lequel des données sont mémorisées. Les relais internes (voir Chapitre 7) sont de tels dispositifs. Un *registre à décalage* est un regroupement de relais internes qui permet aux bits mémorisés d'être décalés d'un relais à l'autre. Ce chapitre s'intéresse aux registres à décalage et à leur utilisation pour le séquencement d'opérations ou pour le suivi d'articles particuliers dans un système de production

11.1 Registres à décalage

Un registre est constitué de plusieurs relais internes regroupés, normalement huit, seize ou trente-deux. Chaque relais interne est dans l'état ouvert ou fermé, ces deux états étant représentés par 0 et 1 ; ces chiffres binaires sont appelés *bits*. Si le registre est constitué de huit relais internes, nous pouvons ainsi mémoriser huit états 0/1. Un tel registre peut être représenté de la manière suivante :

1	2	3	4	5	6	7	8

Chacun des huit relais mémorise un signal tout ou rien et le registre peut, à un instant donné, être dans l'état suivant :

1	0	1	1	0	0	1	0

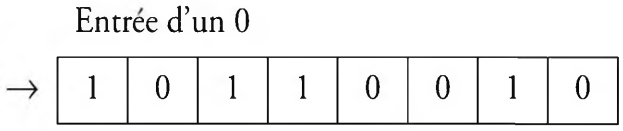
Autrement dit, le relais 1 est activé, le relais 2 est désactivé, le relais 3 est activé, le relais 4 est activé, le relais 5 est désactivé, etc. Un registre constitué de huit éléments est dit *registre huit bits*. Les registres peuvent servir à stocker des données qui proviennent d'entrées autres que de simples dispositifs tout ou rien comme les interrupteurs.

Dans un *registre à décalage*, il est possible de décaler les bits mémorisés. Les registres à décalage ont besoin de trois entrées : une pour charger des données dans le premier emplacement du registre, une pour déclencher le décalage des données d'un emplacement et une pour mettre à zéro ou effacer le registre. Afin d'illustrer cette idée, examinons un registre huit bits qui se trouve dans l'état initial suivant :

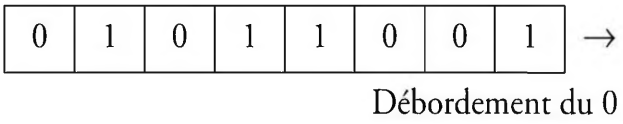
1	0	1	1	0	0	1	0

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

Supposons à présent que nous recevions le signal d'entrée 0, pour le premier relais interne.



Si nous recevons également le signal de décalage, le signal d'entrée arrive dans le premier emplacement du registre et tous les bits sont décalés d'un emplacement. Le dernier bit déborde et est perdu.



Les relais internes qui se trouvaient initialement dans l'état 1, 0, 1, 1, 0, 0, 1, 0 sont à présent dans l'état 0, 1, 0, 1, 1, 0, 0, 1.

Le regroupement des relais internes pour former un registre à décalage est réalisé automatiquement par l'API lorsqu'une fonction de registre à décalage est choisie. Pour un API de Mitsubishi, il faut employer le code de programmation SFT avec le numéro du relais interne qui constitue le premier bit du registre. Le bloc de relais qui commence à ce numéro initial est alors réservé au registre à décalage.

11.2 Programmes en langage à contacts

Étudions la représentation d'un registre à décalage quatre bits dans un schéma à contacts (voir Figure 11.1a). L'entrée E3 est utilisée pour réinitialiser le registre à décalage, c'est-à-dire mettre toutes ses valeurs à zéro. E1 constitue l'entrée dans le premier relais interne du registre. L'entrée E2 permet le décalage de l'état des relais internes. Chaque relais interne du registre, c'est-à-dire RI1, RI2, RI3 et RI4, est connecté à une sortie parmi S1, S2, S3 et S4.

Supposons que nous commencions par une entrée momentanée sur E3. Tous les relais internes sont alors mis à zéro et l'état des quatre relais internes RI1, RI2, RI3 et RI4 est donc 0, 0, 0, 0. Lorsque E1 est momentanément fermée, une entrée à 1 arrive dans le premier relais. L'état des relais internes RI1, RI2, RI3 et RI4 est donc à présent 1, 0, 0, 0. Les contacts de RI1 se ferment et nous obtenons donc une sortie sur S1. Si nous appliquons à présent une entrée momentanée sur E2, le 1 se décale du premier relais vers le deuxième. L'état des relais internes devient alors 0, 1, 0, 0. La sortie S1 est donc désactivée, tandis que la sortie S2 est activée. Si nous appliquons une autre entrée momentanée sur E2, nous décalons à nouveau l'état des relais d'un emplacement, pour obtenir 0, 0, 1, 0. Les sorties S1 et S2 sont donc désactivées, tandis que la sortie S3 est activée. Une nouvelle entrée momentanée sur E2 décale de nouveau l'état des relais d'un emplacement et nous

Copyright © 2015 Dunod.

obtenons 0, 0, 0, 1. Les sorties S1, S2 et S3 sont alors désactivées et la sortie S4 est activée. L'entrée momentanée suivante appliquée à E2 effectue un nouveau décalage et conduit à l'état 0, 0, 0, 0 ; le 1 a débordé et est perdu. Toutes les sorties sont à présent désactivées. La séquence des entrées sur E2 a donc généré un séquencement des sorties : S1, puis S2, puis S3, puis S4. La Figure 11.1b présente les signaux correspondants.

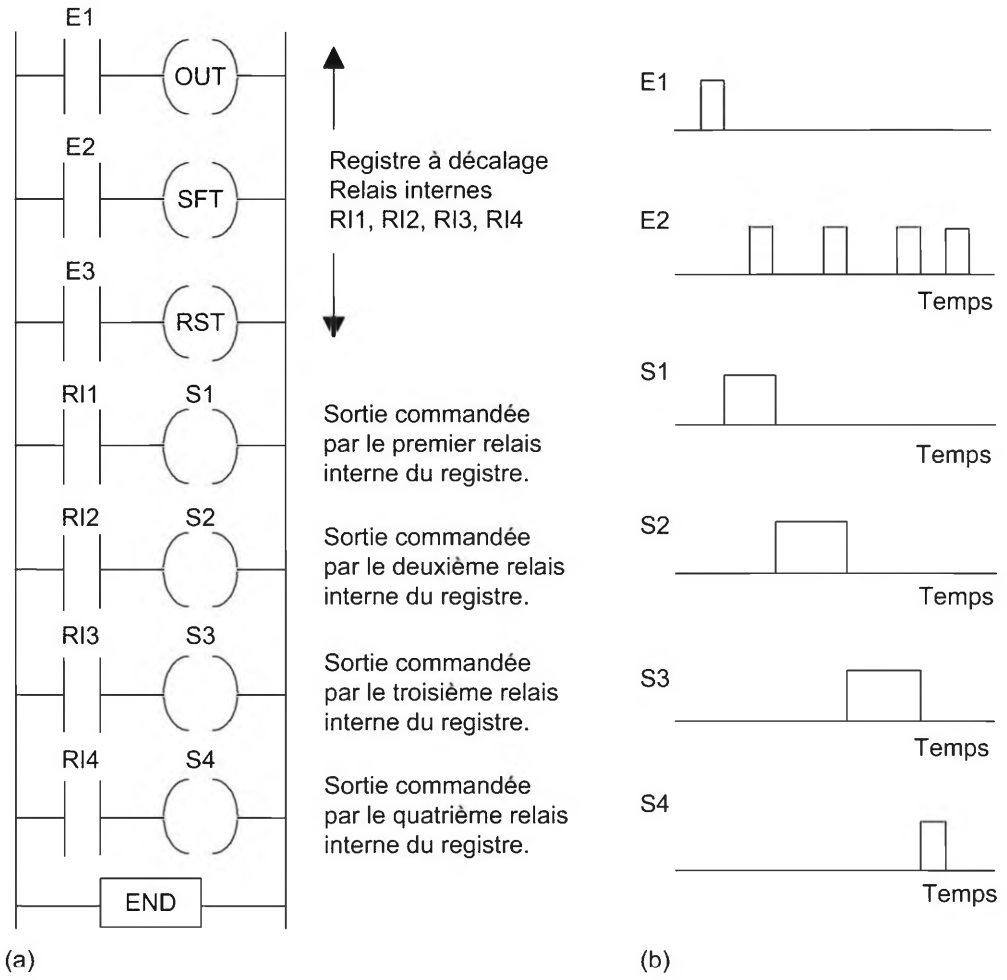


Figure 11.1 – Registre à décalage.

La Figure 11.2 donne la version Mitsubishi du schéma à contacts précédent et la liste d'instructions associée. À la place des trois sorties séparées pour la mise à zéro, la sortie et le décalage, le registre à décalage de Mitsubishi apparaît dans un programme sous forme d'un bloc fonctionnel unique. Dans ce cas, M140 correspond à l'adresse du premier relais du registre.

La Figure 11.3 présente le schéma à contacts d'un registre à décalage pour un API de Toshiba. R016 correspond à l'adresse du premier relais du registre. Le libellé (08) indique que le registre est constitué de huit relais. D correspond à l'entrée des données, S, à l'entrée de décalage, E, à l'entrée d'activation ou de mise à zéro et Q, à la sortie.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

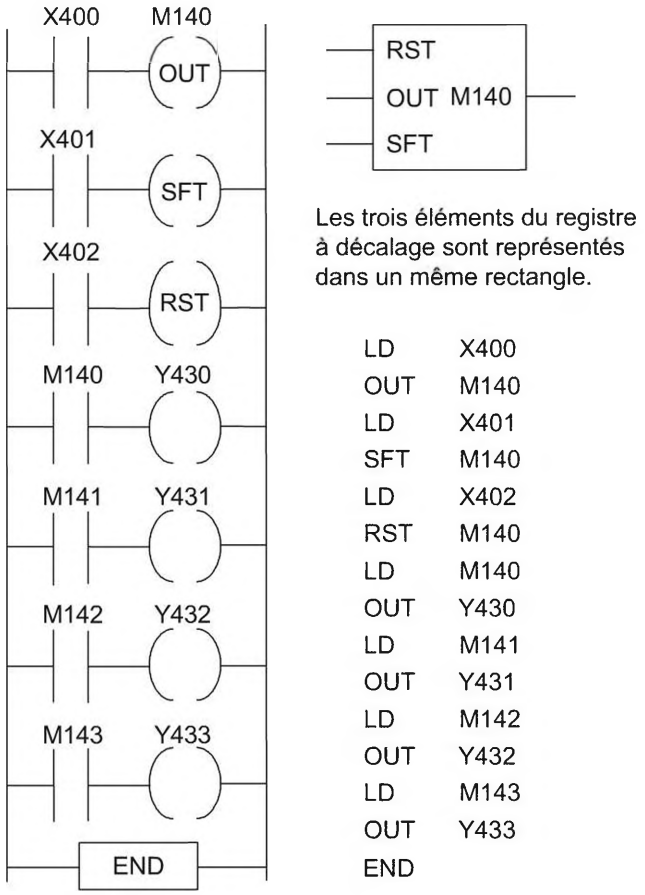


Figure 11.2 – Registre à décalage pour un API de Mitsubishi.

Le symbole du registre à décalage tel que défini par la CEI 61131-3 est illustré à la Figure 11.4. La valeur à décaler se trouve sur l'entrée IN et le nombre d'emplacements dont elle doit être décalée est indiqué par l'entrée N.

La Figure 11.5a présente le symbole d'un registre à décalage selon Siemens. Si l'entrée d'activation EN est à 1, le décalage se produit et ENO est mis à 1. Si EN est à 0, le décalage n'est pas réalisé et ENO est à 0. La fonction SHL_W décale vers la gauche, un bit à la fois, le contenu du mot sur l'entrée IN ; le nombre de décalages est précisé par l'entrée N. Le mot décalé de sortie se trouve sur OUT. La Figure 11.5b montre les symboles des registres à décalage pour les modèles PLC 5 et SLC 500 d'Allen-Bradley. FILE indique l'adresse du tableau des bits à décaler. CONTROL précise l'adresse des bits de contrôle, comme le bit 15 (EN) est à 1 lorsque l'instruction est activée, le bit 13 (DN) est à 1 lorsque les bits ont été décalés, le bit 11 (ER) est à 1 lorsque la longueur est négative et le bit 10 (UL) enregistre l'état du bit qui a débordé. BIT ADDRESS correspond à l'adresse des données à décaler. LENGTH précise le nombre de bits à décaler dans le registre.

Copyright © 2015 Dunod.

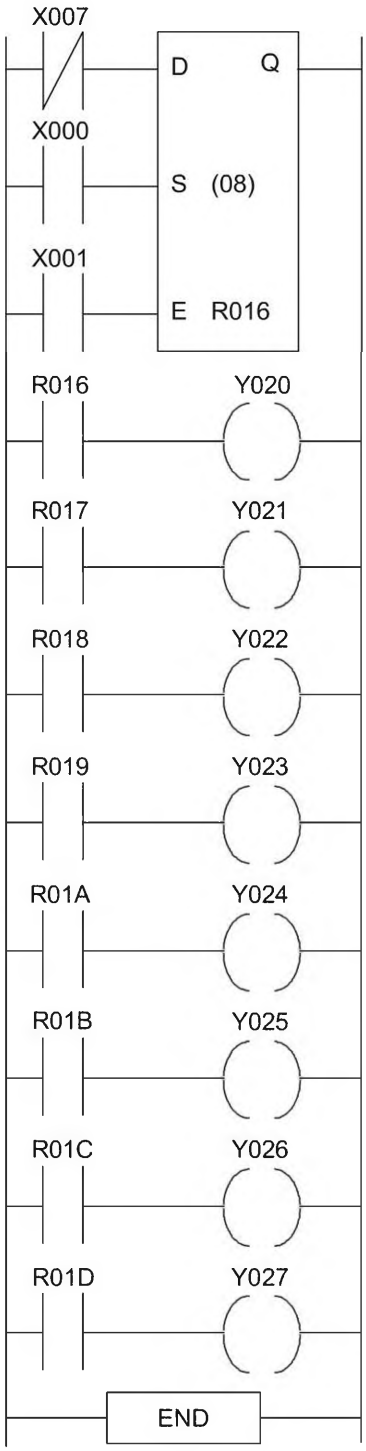


Figure 11.3 – Registre à décalage pour un API de Toshiba.

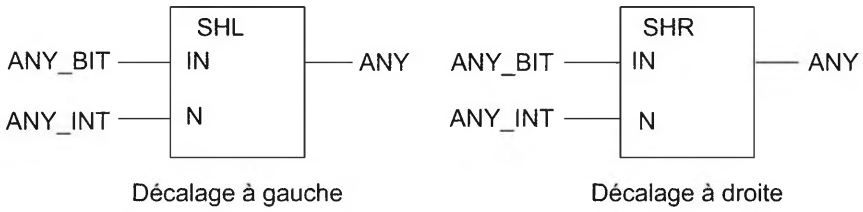


Figure 11.4 – Symboles du registre à décalage définis par la CEI 61131-3.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

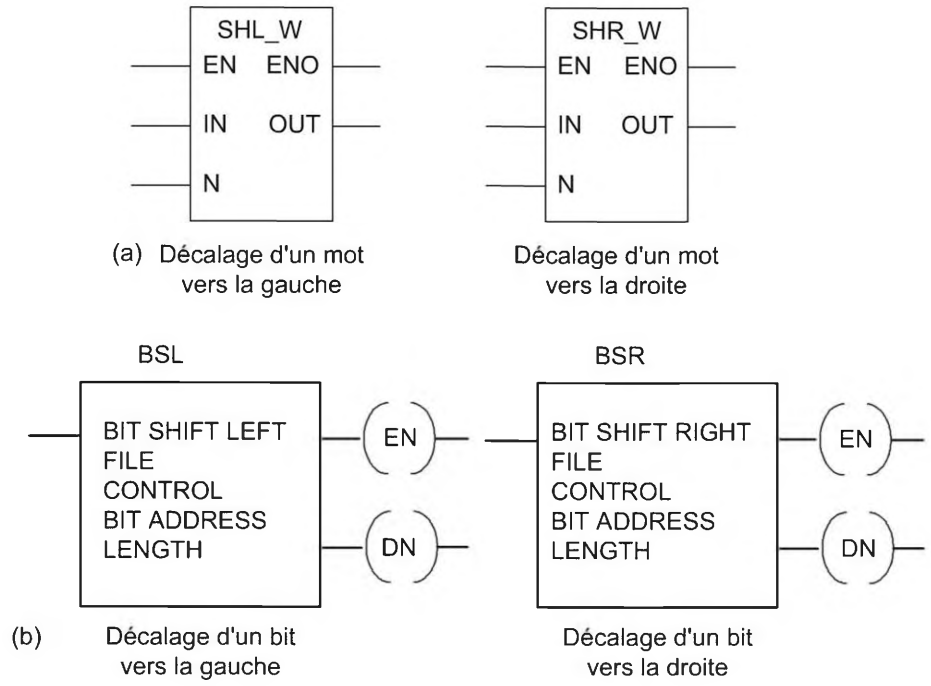


Figure 11.5 – Symboles du registre à décalage : (a) Siemens et (b) Allen-Bradley.

11.2.1 Mise en œuvre d'un séquencement

Examinons un système constitué de deux pistons à double solénoïde qui doit produire la séquence A+, B+, A–, B– (voir Figure 11.6a). Le programme correspondant, présenté à la Figure 11.6b, se fonde sur un registre à décalage.

11.2.2 Suivi d'articles

L'exemple précédent montre comment obtenir un séquencement à l'aide d'un registre à décalage, mais cet élément peut également servir au suivi d'articles. Par exemple, un capteur peut détecter les articles défectueux convoyés par une bande transporteuse et les suivre afin que, lorsqu'ils arrivent à l'endroit approprié, un mécanisme d'éjection soit activé et les retire de la bande transporteuse. La Figure 11.7 illustre ce système et un schéma à contacts possible.

Chaque fois qu'un article défectueux est détecté, une impulsion se produit sur l'entrée X400. Cela conduit à l'entrée d'un 1 sur le relais interne M140 du registre à décalage. Lorsque les articles se déplacent, qu'ils soient défectueux ou non, une impulsion d'entrée est présente sur X401. Le 1 se décale alors dans le registre. Lorsque ce 1 atteint le registre interne M144, il active la sortie Y430 et le mécanisme d'éjection retire l'article défectueux de la bande transporteuse. Lorsque l'article est retiré, une entrée sur X403 se produit. Elle remet à zéro le mécanisme afin que les articles suivants ne soient pas retirés de la bande transporteuse avant que le signal d'éjection n'atteigne M144. Pour cela, la sortie sur le relais interne M100 est activée, ce qui verrouille l'entrée X403 et désactive la sortie d'éjection Y430. Ce schéma présente uniquement les éléments de base du système. Un

Copyright © 2015 Dunod.

système réel comprendrait d'autres relais internes pour vérifier que le mécanisme d'éjection est désactivé lorsque des articles valides se déplacent sur la bande transporteuse et pour désactiver l'entrée sur X400 lors du décalage.

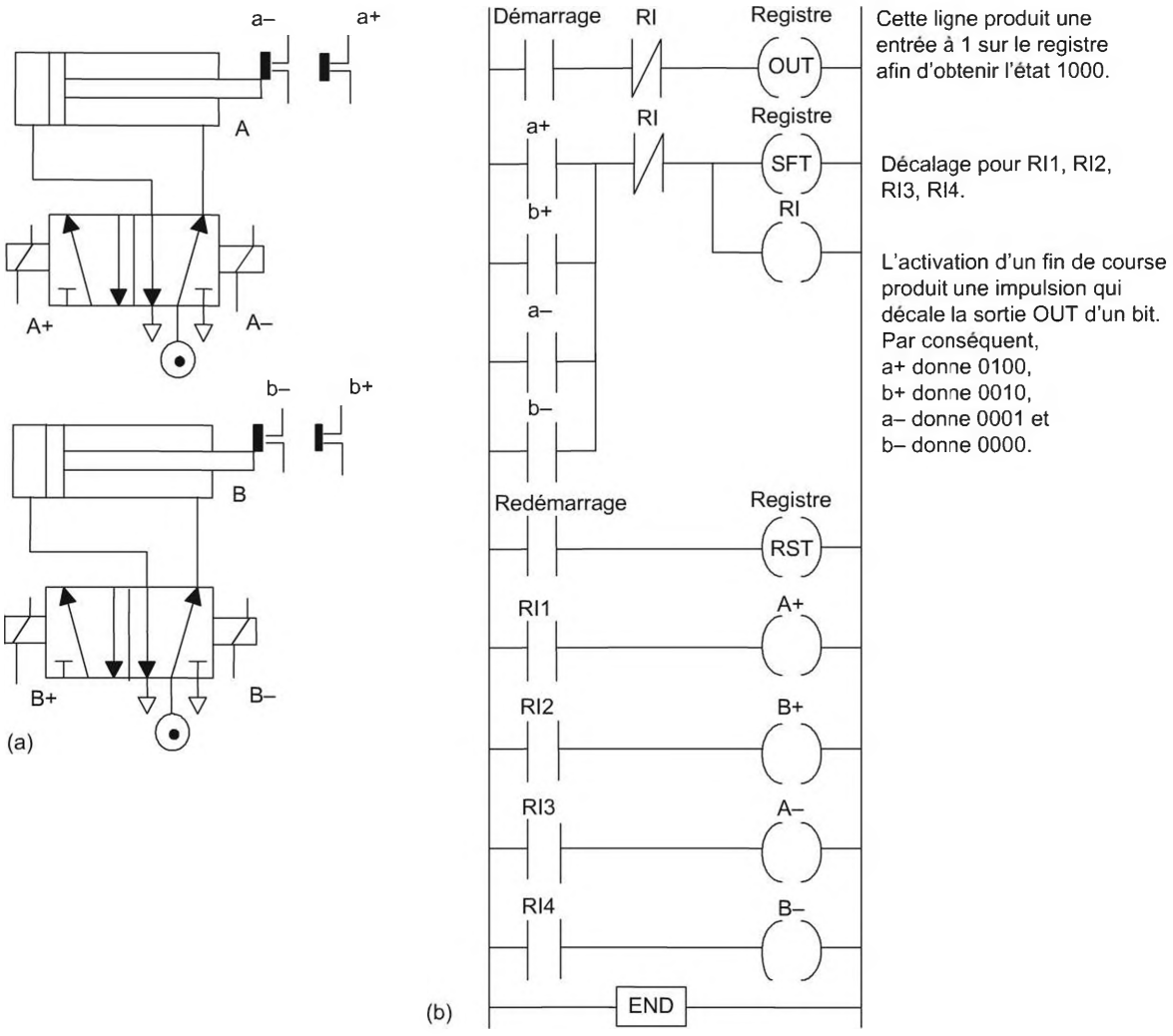


Figure 11.6 – Séquencement de pistons.

11.3 En résumé

Un *registre* est un dispositif électronique dans lequel des données sont mémorisées. Un *registre à décalage* est un regroupement de relais internes qui permet aux bits mémorisés d'être décalés d'un relais à l'autre. Dans un registre à décalage, il est possible de décaler les bits mémorisés. Les registres à décalage ont besoin de trois entrées : une pour charger des données dans le premier emplacement du registre, une pour déclencher le décalage des données d'un emplacement et une pour mettre à zéro ou effacer le registre. Le regroupement des relais internes pour former un registre à décalage est réalisé automatiquement par l'API lorsqu'une fonction de registre à décalage est choisie.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

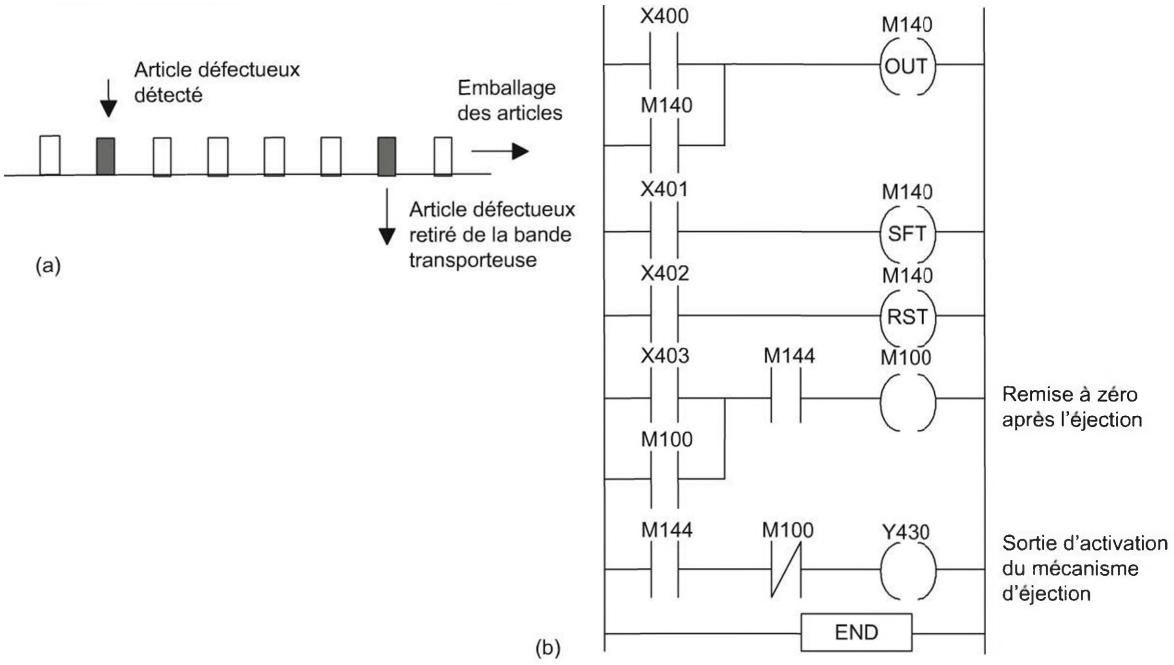


Figure 11.7 – Suivi des articles défectueux.

11.4 Problèmes

Pour les questions 1 à 9, quatre réponses sont proposées : A, B, C et D. Choisissez la réponse correcte. Les problèmes 1 à 5 concernent un registre à décalage quatre bits, avec les relais internes RI1, RI2, RI3 et RI4, qui a été réinitialisé dans l'état 0, 0, 0, 0.

1. Lorsqu'une impulsion d'entrée à 1 arrive sur la bobine OUT du registre à décalage, les relais internes du registre affichent :
   A. 0001
   B. 0010
   C. 0100
   D. 1000
2. Après une impulsion d'entrée à 1 sur la bobine OUT du registre à décalage, une impulsion d'entrée arrive sur SHIFT. Les relais internes affichent alors :
   A. 0001
   B. 0010
   C. 0100
   D. 1000
3. L'entrée sur la bobine OUT du registre à décalage étant toujours à 1, une impulsion d'entrée arrive sur SHIFT. Les relais internes affichent alors :
   A. 0011
   B. 0110
   C. 1100
   D. 0010

Copyright © 2015 Dunod.

4. L'entrée sur la bobine OUT du registre à décalage étant toujours à 1, deux impulsions d'entrée arrivent sur SHIFT. Les relais internes affichent alors :
   A. 0001
   B. 0010
   C. 1100
   D. 1110
5. L'entrée sur la bobine OUT du registre à décalage étant toujours à 1, une impulsion d'entrée arrive sur SHIFT, suivie d'une impulsion d'entrée sur RESET. Les relais internes affichent alors :
   A. 0000
   B. 0010
   C. 0100
   D. 1000

Les problèmes 6 à 9 font référence à la Figure 11.8, qui présente un registre à décalage quatre bits, avec les relais internes RI1, RI2, RI3 et RI4, trois entrées (E1, E2 et E3) et quatre sorties (S1, S2, S3 et S4).

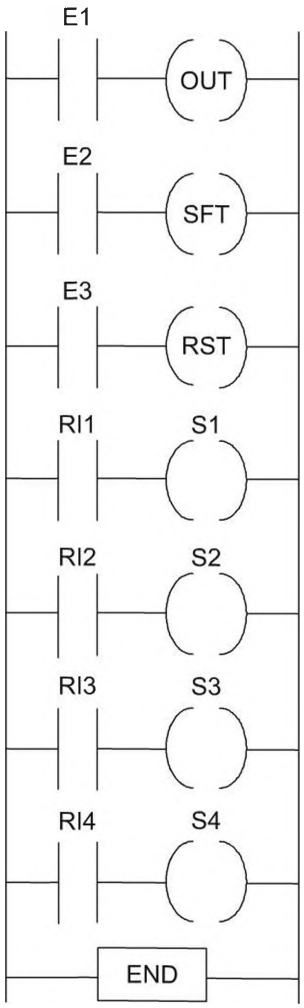


Figure 11.8 – Diagramme pour les problèmes 6 à 9.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

6. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Lorsqu'il existe une impulsion d'entrée sur E1 :
   (i) la sortie S1 est activée ;
   (ii) les contacts du relais interne RI1 se ferment.

   Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
   A. (i) V (ii) V
   B. (i) V (ii) F
   C. (i) F (ii) V
   D. (i) F (ii) F

7. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Lorsqu'il existe une impulsion d'entrée sur E1, suivie d'une impulsion d'entrée sur SFT :
   (i) la sortie S1 est activée ;
   (ii) la sortie S2 est activée.

   Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
   A. (i) V (ii) V
   B. (i) V (ii) F
   C. (i) F (ii) V
   D. (i) F (ii) F

8. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Pour obtenir l'activation en séquence des sorties S1, S2, S3 et S4 et leur maintien dans cet état, il faut sur les entrées :
   (i) une impulsion sur E1 suivie de trois impulsions sur SFT ;
   (ii) une entrée permanente sur E1 suivie de trois impulsions sur SFT.

   Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
   A. (i) V (ii) V
   B. (i) V (ii) F
   C. (i) F (ii) V
   D. (i) F (ii) F

9. État initial : S1 active, S2 inactive, S3 inactive, S4 inactive

   État suivant : S1 active, S2 inactive, S3 inactive, S4 inactive

   État suivant : S1 inactive, S2 active, S3 inactive, S4 inactive

   État suivant : S1 active, S2 inactive, S3 active, S4 inactive

   Pour obtenir la séquence précédente, les entrées requises sont :
   A. Une impulsion d'entrée sur E1 suivie d'une impulsion d'entrée sur E2
   B. Une impulsion d'entrée sur E1 suivie de deux impulsions d'entrée sur E2
   C. Une impulsion d'entrée sur E1 suivie d'une impulsion d'entrée sur E2, puis d'une impulsion d'entrée sur E1.
   D. Une impulsion d'entrée sur E1 suivie d'une impulsion d'entrée sur E2, puis d'une impulsion d'entrée sur E1 et E2.

Copyright © 2015 Dunod.

10. Concevez les programmes en langage à contacts des systèmes qui mettent en œuvre les tâches suivantes :
    (a) Obtenir une séquence de quatre sorties telle que la sortie 1 est activée et le reste lorsque le premier événement est détecté, la sortie 2 est activée et le reste lorsque le deuxième événement est détecté, la sortie 3 est activée et le reste lorsque le troisième événement est détecté, la sortie 4 est activée et le reste lorsque le quatrième événement est détecté, et toutes les sorties sont désactivées lorsqu'un signal d'entrée particulier se produit.
    (b) Commander un pistolet à peinture dans une cabine par laquelle passent des articles sur un convoyeur aérien. Le pistolet doit être activé lorsqu'un article se trouve devant lui et désactivé dans le cas contraire. Les articles sont suspendus au convoyeur aérien par des crochets ; tous les crochets ne transportent pas nécessairement un article.

11.5 Recherches complémentaires

1. Recherchez des informations sur les registres à décalage disponibles dans une gamme d'API d'un fabricant.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

Copyright © 2015 Dunod.

12 • GESTION DES DONNÉES

Les temporisateurs, les compteurs et les relais internes sont tous impliqués dans une gestion de bits individuels, c'est-à-dire des signaux tout ou rien uniques. Quant aux registres à décalage, ils gèrent plusieurs bits formés par un groupe de relais internes associés (voir Chapitre 11). Les opérations se font sur le bloc de bits du registre. Ce chapitre s'intéresse aux opérations de l'API qui concernent des blocs de données représentant une valeur ; ces blocs sont appelés *mots*. Un bloc de données est indispensable si l'on veut représenter des nombres à la place d'une simple entrée tout ou rien. La gestion des données comprend des opérations qui déplacent ou transfèrent des informations numériques depuis un mot de mémoire vers un autre enregistré à un emplacement différent, qui comparent des valeurs et qui réalisent des opérations arithmétiques simples. Par exemple, nous pouvons comparer une valeur numérique à une valeur de référence et déclencher une action si la valeur actuelle est inférieure à la valeur fixée. Ce chapitre constitue une introduction aux opérations de ce type.

12.1 Registres et bits

Les données sont mémorisées dans des registres (voir Section 8.1 pour une présentation des registres). Un API dispose de plusieurs registres. Chaque registre de données peut mémoriser un *mot binaire* constitué, en général, de huit ou seize bits. Le *système binaire* utilise uniquement les deux symboles 0 et 1 (voir Chapitre 3). Ainsi, le nombre sur quatre bits 1111 correspond à la représentation binaire du nombre décimal $2^0 + 2^1 + 2^2 + 2^3 = 1 + 2 + 4 + 8 = 15$. Un registre quatre bits peut contenir une valeur positive dans l'intervalle 0 à $2^0 + 2^1 + 2^2 + 2^3$ ou $2^4 - 1 = 15$. Un registre huit bits peut mémoriser une valeur positive entre 0 et $2^0 + 2^1 + 2^2 + 2^3 + 2^4 + 2^5 + 2^6 + 2^7$ ou $2^8 - 1$, c'est-à-dire 255. Un registre seize bits peut stocker une valeur positive entre 0 et $2^{16} - 1$, c'est-à-dire 65 535.

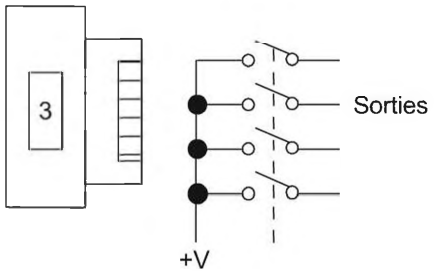
Un mot de seize bits peut donc être utilisé pour des nombres positifs dans l'intervalle 0 à +65 535. Si des nombres négatifs doivent être manipulés, le bit de poids fort indique le signe : 1 représente un nombre négatif, 0, un nombre positif. Le format employé avec les nombres négatifs est le *complément à deux*. Le complément à deux est une manière d'écrire les nombres négatifs afin que la somme des équivalents signés de +5 et de −5, par exemple, donne 0. Dans ce format, 1011

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

représente le nombre négatif −5 et 0101, le nombre positif +5 ; 1011 + 0101 = (1)0000, le (1) étant perdu dans le cas d'un nombre sur quatre bits. Pour plus de détails, consultez le Chapitre 3.

Très souvent, le format *décimal codé en binaire* (BCD, *Binary Coded Decimal*) est utilisé avec les API, notamment lorsqu'ils sont connectés à des dispositifs comme des afficheurs numériques. Avec le système binaire normal, il n'existe aucun rapport simple entre les différents chiffres d'un nombre décimal et son équivalent binaire. Il faut procéder à des opérations arithmétiques pour passer d'une base à l'autre. Dans le système BCD, chaque chiffre décimal est représenté par un nombre binaire sur quatre bits (il faut au moins quatre bits pour pouvoir représenter un chiffre décimal entre zéro et neuf, c'est-à-dire $2^n \geq 10$). Pour illustrer cette idée, prenons le nombre décimal 123. Le 3 est représenté par le nombre binaire sur quatre bits 0011, le 2, par le nombre 0010, et le 1, par 0001. Par conséquent, l'équivalent BCD du nombre 123 est 0001 0010 0011. Le système BCD se révèle très pratique avec les dispositifs externes qui se fondent sur un format décimal, comme les interrupteurs décimaux à molette et les afficheurs numériques. Dans ce cas, quatre bits sont utilisés pour représenter chaque chiffre décimal. Les API disposent souvent d'entrées et de sorties qu'il est possible de programmer de manière à convertir les nombres BCD fournis par les dispositifs externes en nombres binaires utilisés dans l'API, ainsi que pour convertir les nombres binaires utilisés en interne par l'API en nombres BCD destinés aux dispositifs de sorties externes (voir Section 12.3).

Les interrupteurs à molette sont largement employés pour la saisie manuelle de données BCD dans un API. Ils disposent de quatre contacts, qui peuvent être ouverts ou fermés pour coder les quatre bits qui représentent le chiffre décimal (voir Figure 12.1). Les contacts sont ouverts ou fermés en faisant tourner une molette à l'aide du pouce. En utilisant plusieurs interrupteurs de ce type, il est possible d'entrer des données au format BCD.



Position	Sorties
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

0 = interrupteur ouvert
1 = interrupteur fermé

Figure 12.1 – Interrupteur à molette.

Copyright © 2015 Dunod.

12.2 Gestion de données

Les sections suivantes décrivent quelques exemples d'instructions de gestion des données que proposent les API.

12.2.1 Déplacement de données

La Figure 12.2 présente une méthode courante pour déplacer des données d'un emplacement ou d'un registre à un autre. Elle utilise une ligne d'un schéma à contacts pour chaque opération de déplacement, dans les format employés par Mitsubishi, Allen-Bradley et Siemens. Pour la ligne donnée en exemple, lorsque l'entrée est présente sur | |, le déplacement se fait depuis l'adresse source indiquée vers l'adresse de destination précisée. Avec ces API, l'instruction type du schéma à contacts comprend l'instruction de traitement des données, l'adresse source (S), qui indique où sont prises les données, et l'adresse de destination (D), qui précise où doivent être déplacées les données. Dans l'approche employée par certains fabricants, comme Siemens, le déplacement des données est vu comme deux instructions séparées : le chargement des données à partir de la source dans un accumulateur, puis le transfert des données depuis l'accumulateur vers la destination. La Figure 12.2c présente le symbole Siemens de la fonction MOVE. Les données sont déplacées depuis l'entrée IN vers la sortie OUT lorsque l'entrée EN est activée.

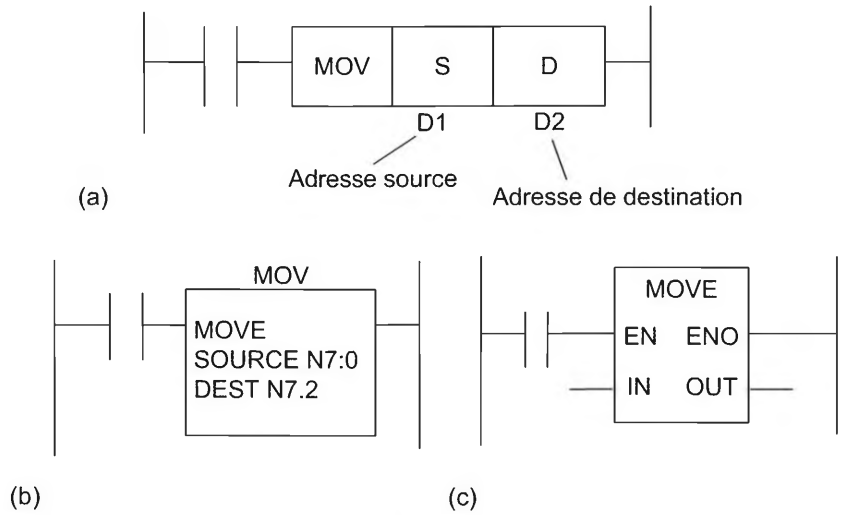


Figure 12.2 – Déplacement de données : (a) Mitsubishi, (b) Allen-Bradley et (c) Siemens.

Les transferts de données peuvent servir à déplacer la valeur présélectionnée vers un temporisateur ou un compteur, une durée ou une valeur de compteur vers un registre pour sa mémorisation, des données depuis une entrée vers un registre ou un registre vers une sortie. La Figure 12.3 présente une ligne, au format Allen-Bradley, qui peut être utilisée pour transférer le nombre stocké à l'adresse N7:0 vers la valeur de présélection du temporisateur T4:6 lorsque les conditions

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

d'entrée de cette ligne sont satisfaites. Le transfert de la valeur cumulée d'un compteur vers un registre aurait une adresse source de la forme C5:18.ACC et une adresse de destination de la forme N7:0. Le transfert d'une entrée vers un registre aurait une adresse source de la forme I:012 et une adresse de destination de la forme N7:0. Le transfert d'un registre vers une sortie aurait une adresse source de la forme N7:0 et une adresse de destination de la forme O:030.

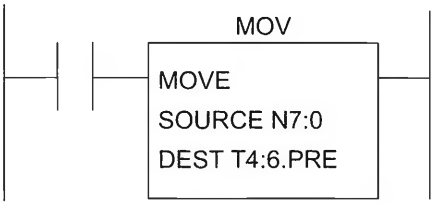


Figure 12.3 – Déplacement d'un nombre vers la présélection d'un temporisateur.

12.2.2 Comparaison de données

Grâce aux instructions de comparaison de données, l'API est capable de comparer deux valeurs. Il peut ainsi comparer une valeur numérique lue à partir d'un dispositif d'entrée et une deuxième valeur contenue dans un registre. Par exemple, nous pourrions souhaiter déclencher une action lorsque l'entrée issue d'un capteur de température produit une valeur numérique inférieure à une valeur mémorisée dans un registre de l'API. Les API disposent généralement des opérations de comparaison *inférieur à* (<, LT ou LES), *égal à* (=, ==, EQ ou EQU), *inférieur ou égal à* (≤, <=, LE ou LEQ), *supérieur à* (>, GT ou GRT), *supérieur ou égal à* (≥, >=, GE ou GEQ) et *différent de* (…, <>, NE ou NEQ). Les parenthèses indiquent les abréviations couramment employées dans les programmes. Voici un exemple dans un programme en texte structuré :

```
(*Vérifier que la pression P2 est inférieure à la pression P1*)
Output := P2 < P1;
```

Dans un schéma à contacts, l'instruction comprend généralement l'instruction de comparaison des données, l'adresse source (S), qui précise où est prise la valeur à comparer, et l'instruction de destination (D), qui précise où se trouve la donnée de comparaison. Les instructions de comparaison les plus employées correspondent aux termes donnés précédemment dans les parenthèses. La Figure 12.4 montre le format employé par trois fabricants pour la comparaison « supérieur à » ; des formats semblables sont utilisés pour les autres opérations de comparaison. À la Figure 12.4a, le format est celui de Mitsubishi, où S indique la source de la valeur à comparer et D, la destination ou la valeur de comparaison. Si la valeur de la source est supérieure à celle de la destination, la sortie est à 1. À la Figure 12.4, le format d'Allen-Bradley est présenté. Dans ce cas, la source de données à comparer est la valeur cumulée du temporisateur T4.0 et la valeur de comparaison est le nombre 400. La Figure 12.4c présente le format de Siemens. Les valeurs comparées sont les entrées IN1 et IN2, tandis que le résultat de la comparaison est la sortie, qui est à 1 si la comparaison réussit, sinon 0. R indique des nombres réels,

Copyright © 2015 Dunod.

c'est-à-dire des nombres à virgule flottante. I est utilisé pour des entiers, c'est-à-dire des nombres à virgule fixe sur seize bits. D correspond à des nombres à virgule fixe sur trente-deux bits. Les deux entrées doivent avoir le même type de données, par exemple REAL.

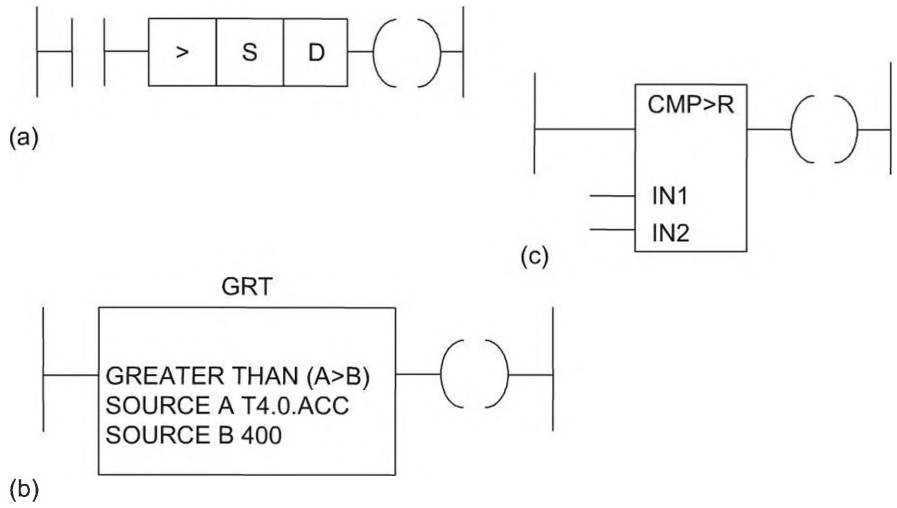


Figure 12.4 – Comparaison « supérieur à » : (a) Mitsubishi, (b) Allen-Bradley et (c) Siemens.

Pour illustrer l'utilisation de la comparaison, étudions un système dans lequel une alarme doit être déclenchée lorsqu'un capteur indique qu'une température a dépassé une certaine valeur, disons 100 °C. L'alarme doit rester en fonctionnement jusqu'à ce que la température redescende sous 90 °C. La Figure 12.5 présente le schéma à contacts que nous pouvons utiliser. Lorsque la température augmente pour devenir supérieure ou égale à 100 °C, l'élément de comparaison « supérieur » produit une sortie à 1 et active alors un relais interne. La sortie est ensuite activée. Elle verrouille l'élément de comparaison pour que la sortie reste active même lorsque la température repasse sous 100 °C. La sortie est désactivée uniquement lorsque l'élément de comparaison « inférieur à 90 °C » produit une sortie et met le relais interne à zéro.

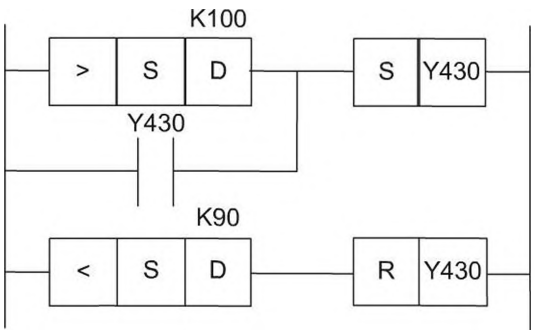


Figure 12.5 – Programme d'une alarme.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

Comme autre exemple de comparaison, prenons quatre sorties qui doivent être activées en séquence. La sortie 1 est activée lorsque l'interrupteur initial est fermé, suivie un peu plus tard de la sortie 2, puis de la sortie 3 et enfin de la sortie 4. Bien que ce système pourrait être mis en œuvre par trois temporisateurs, nous pouvons également utiliser un temporisateur avec des éléments « supérieur ou égal à ». La Figure 12.6 présente un schéma à contacts possible. Lorsque les contacts de X401 se ferment, la sortie Y430 est activée. Le temporisateur est également démarré. Lorsque la valeur cumulée du temporisateur atteint 5 s, l'élément de comparaison active Y431. Lorsque la valeur cumulée du temporisateur atteint 15 s, l'élément de comparaison active Y432. Lorsque la valeur cumulée du temporisateur atteint 25 s, ses contacts activent Y433.

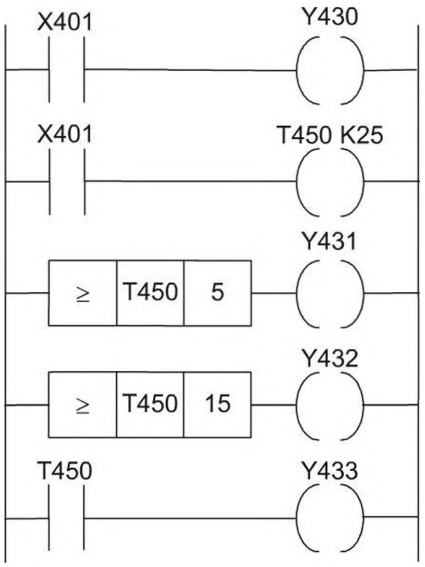
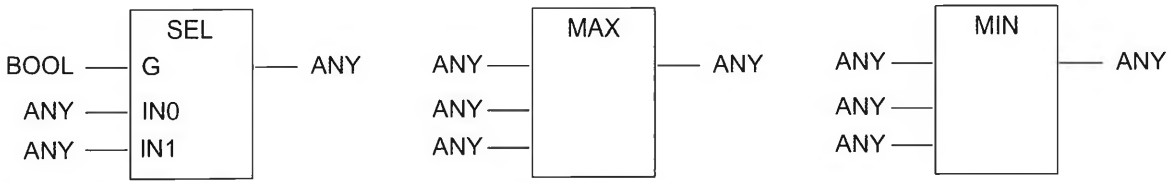


Figure 12.6 – Activation séquentielle.

12.2.3 Sélection de données

Les API proposent différents blocs de sélection des données. La Figure 12.7 montre les symboles correspondants définis par la CEI.



(a) Si G = 1, alors la sortie est IN1, sinon IN0.

(b) La sortie est la valeur maximale des entrées.

(c) La sortie est la valeur minimale des entrées.

Figure 12.7 – Symboles de la CEI : (a) sélection, (b) maximum et (c) minimum.

Copyright © 2015 Dunod.

12.3 Fonctions arithmétiques

La plupart des API disposent de fonctions de conversion BCD vers binaire ou entier et de binaire ou entier vers BCD. Elles se révèlent particulièrement utiles lorsque l'entrée provient d'un interrupteur à molette ou que la sortie est un afficheur décimal. La Figure 12.8a présente une instruction employée dans de tels cas. La Figure 12.8b présente la conversion BCD vers entier et entier vers BCD telle qu'on la rencontre dans les API de Siemens, tandis que la Figure 12.8c montre cette conversion BCD au format Allen-Bradley.

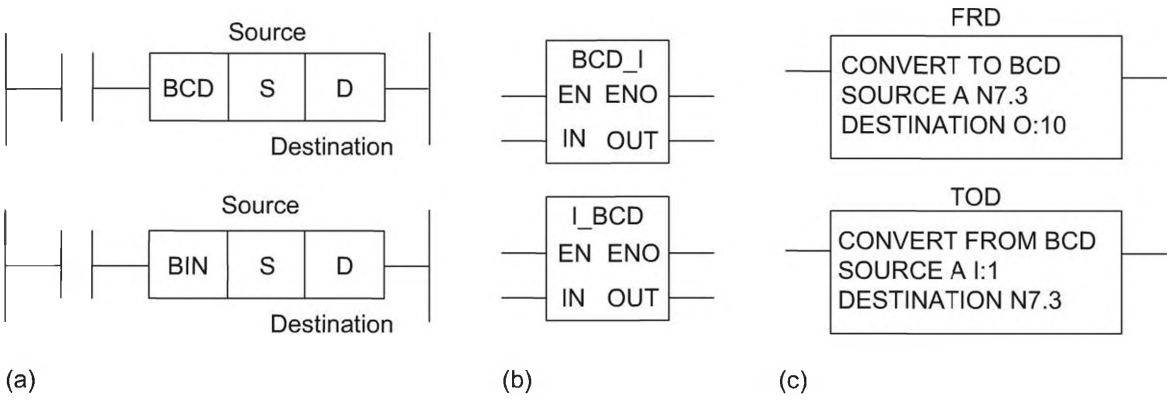


Figure 12.8 – Conversion BCD vers binaire et binaire vers BCD : (a) un format, (b) le format Siemens et (c) le format Allen-Bradley.

12.3.1 Opérations arithmétiques

Certains API sont uniquement capables de réaliser des opérations d'addition et de soustraction. D'autres disposent des quatre opérations arithmétiques de base (addition, soustraction, multiplication et division). D'autres encore offrent ces opérations ainsi que diverses fonctions, comme l'exponentielle. L'addition et la soustraction permettent de modifier la valeur des registres. Elle peuvent, par exemple, servir à ajuster la valeur lue d'un capteur, obtenir le résultat de la soustraction des valeurs de deux capteurs ou modifier les valeurs de présélection de temporisateurs et de compteurs. La multiplication peut servir à multiplier une valeur d'entrée avant, peut-être, de la combiner à une autre valeur par addition ou soustraction.

La manière de programmer les API pour ces opérations varie. Les modèles d'Allen-Bradley offrent les opérations d'addition (ADD), de soustraction (SUB), de division (DIV), de multiplication (MUL) et de racine carrée (SQR). La Figure 12.9a présente le format de l'addition, celui des autres fonctions arithmétiques étant comparable. La valeur présente sur la source A, c'est-à-dire en N7.1, est additionnée à la valeur présente sur la source B, c'est-à-dire en N7.3, et le résultat est placé dans la destination N7.5.

La Figure 12.9b illustre le format de base des instructions arithmétiques de Siemens. Les fonctions disponibles avec les entiers sont l'addition (ADD_1), la

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

soustraction (SUB_1), la multiplication (MUL_1) et la division (DIV_1), avec le quotient en résultat. Les fonctions arithmétiques sont exécutées lorsque l'entrée d'activation EN est à 1.

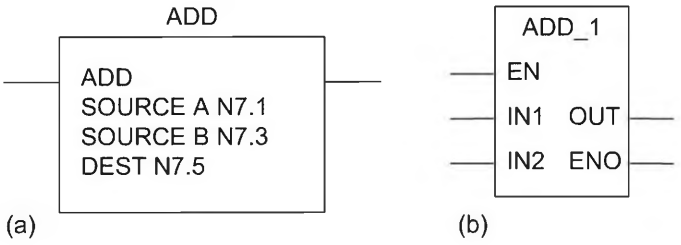


Figure 12.9 – Addition : (a) format Allen-Bradley et (b) format Siemens.

12.4 Contrôle en boucle fermée

La température d'une pièce peut être contrôlée en allumant un radiateur électrique. Le radiateur va chauffer la pièce jusqu'à la température maximale possible, en tenant compte de la rapidité à laquelle le radiateur chauffe la pièce et celle à laquelle il perd de la chaleur. Dans ce cas, il s'agit d'un *contrôle en boucle ouverte*, car aucune rétroaction sur le radiateur ne modifie la rapidité de chauffage de la pièce. Pour contrôler l'évolution de la température, nous avons besoin d'un thermostat qui allume le radiateur lorsque la température de la pièce est en dessous de la valeur requise et l'éteint lorsqu'elle est au-dessus. Dans ce système, il existe un retour d'informations sur la température. Il s'agit d'un *contrôle en boucle fermée*.

Un contrôle en boucle fermée d'une certaine variable, comme la température dans une pièce, est obtenu en comparant la valeur réelle de la variable à la valeur souhaitée, la consigne, et en produisant une sortie, comme allumer le radiateur, de manière à réduire la différence entre ces deux valeurs. La Figure 12.10 illustre cette idée au moyen d'un schéma fonctionnel. La valeur réelle de la variable est comparée à la valeur de référence et un signal qui représente la différence ou l'erreur est obtenu. Un contrôleur prend ensuite ce signal de différence et produit une sortie sur un actionneur de manière à corriger la divergence.

La Figure 12.11 illustre l'utilisation d'un API pour mettre en œuvre un contrôle en boucle fermée. Nous supposons que l'actionneur et les valeurs mesurées sont analogiques et qu'une conversion numérique doit donc être effectuée ; les unités de conversion analogique vers numérique et numérique vers analogique sont ainsi incluses.

12.4.1 Modes de contrôle

La réaction du contrôleur à un signal d'erreur peut se faire selon différentes méthodes :

– *Mode tout ou rien*. Dans ce cas, le contrôleur joue essentiellement le rôle d'un interrupteur qui fournit un signal tout ou rien selon la présence ou l'absence

Copyright © 2015 Dunod.

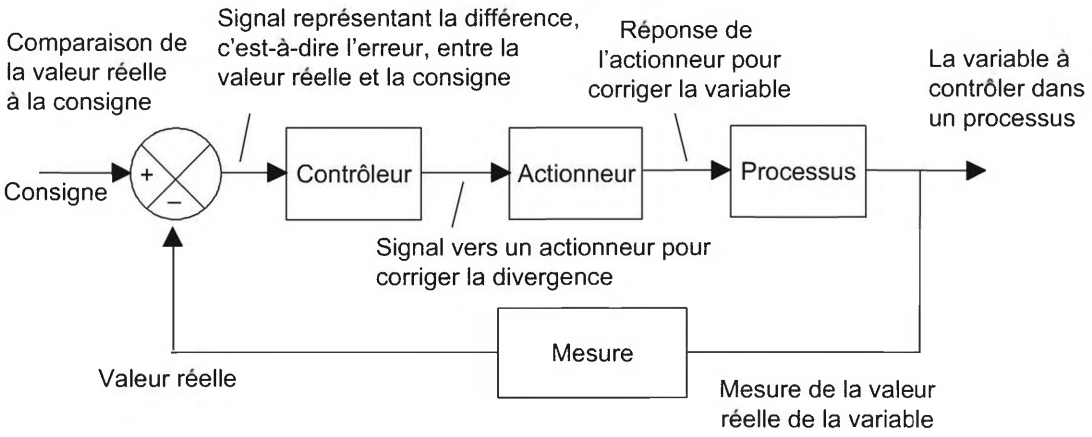


Figure 12.10 – Contrôle en boucle fermée.

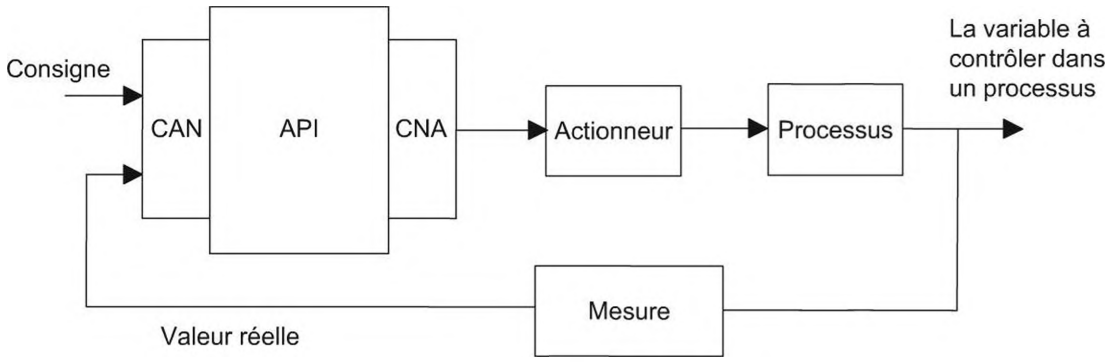


Figure 12.11 – API utilisé pour un contrôle en boucle fermée.

d'une erreur (voir Figure 12.12a). En cas d'erreur, le contrôleur produit une sortie constante, quelle que soit la taille de l'erreur. Un bilame est un exemple de contrôleur de ce type (voir Figure 2.11), souvent utilisé dans les systèmes de chauffage central des maisons. Il s'agit d'une forme d'interrupteur qui est éteint lorsque la température se trouve au-dessus de la température souhaitée et allumé lorsqu'elle se trouve en dessous. Puisque l'action de commande est discontinue et que des décalages temporels existent dans le système, la valeur de la variable contrôlée oscille autour de la consigne. Par exemple, lorsqu'un thermostat bilame est activé, le radiateur ne produit pas un effet immédiat sur la température et, lorsque la température a atteint la valeur souhaitée, la coupure du radiateur ne baisse pas immédiatement la température. Toutefois, un contrôle tout ou rien n'est pas trop mauvais si le système fait preuve d'une grande inertie, car l'effet des changements conduit à une modification lente de la variable. Une zone morte ou une hystérésis est souvent ajoutée pour que le contrôleur évite de réagir aux petites valeurs d'erreur et ne commute pas constamment lorsque la variable oscille autour de la consigne. La valeur d'activation de la variable est donc différente de sa valeur de désactivation (voir Figure 12.12b).

– *Mode proprotionnel.* Dans ce mode, le contrôleur envoie à l'actionneur une sortie proportionnelle à la différence entre la valeur réelle et la valeur de référence, c'est-à-dire de l'erreur (voir Figure 12.12c). Un tel contrôle ne peut être

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

réalisé qu'avec un API qui dispose de fonctions arithmétiques de base. La consigne et la valeur réelle sont probablement analogiques et doivent donc être converties en numérique. La valeur réelle est ensuite soustraite de la valeur de référence et la différence est multipliée par une constante, la constante de gain K_P, pour obtenir la sortie. Celle-ci, après conversion en analogique, correspond au signal de correction appliqué à l'actionneur :

$$\text{Sortie du contrôleur} = K_P \times \text{Erreur}$$

– *Mode intégral.* Dans ce cas, la sortie du contrôleur est proportionnelle à l'intégrale de l'erreur avec le temps, c'est-à-dire la zone sous le diagramme erreur-temps (voir Figure 12.12d) :

$$\text{Sortie du contrôleur} = K_I \times \text{Intégrale de l'erreur avec le temps}$$

– *Mode dérivé.* Dans ce mode, la sortie du contrôleur est proportionnelle à la vitesse de changement de l'erreur, c'est-à-dire à la pente dans le diagramme erreur-temps (voir Figure 12.12e) :

$$\text{Sortie du contrôleur} = K_D \times \text{Vitesse de changement d'erreur}$$

– *Combinaisons des modes.* En général, les trois modes (proportionnel, intégral et dérivé) sont combinés pour obtenir le *mode PID.*

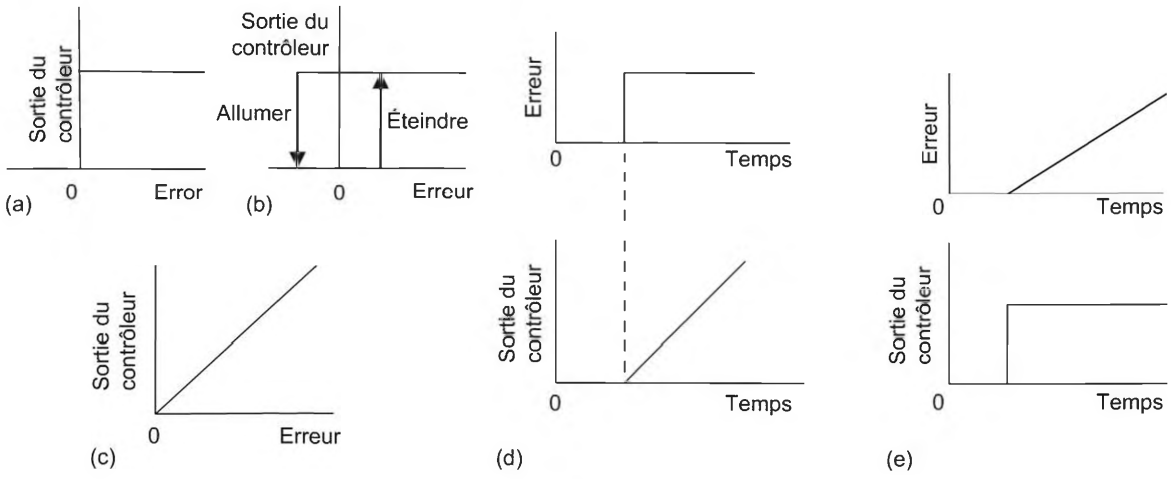


Figure 12.12 – Contrôle : (a) marche-arrêt, (b) marche-arrêt avec hystérésis, (c) proportionnel, (d) intégral et (e) dérivé.

Le contrôle proportionnel présente un inconvénient car, en raison des décalages temporels inhérents au système, le signal de correction appliqué à l'actionneur a tendance à faire osciller la variable autour de la consigne. Il faut donc un signal de correction qui diminue lorsque la variable s'approche de la valeur de référence.

Copyright © 2015 Dunod.

Pour cela, on utilise le *contrôle PID*. Le contrôleur produit un signal de correction qui est calculé à partir d'un élément proportionnel (le terme P), d'un élément lié aux valeurs précédentes de la variable (le terme I, pour intégrale) et d'un élément lié à la vitesse de changement de la variable (le terme D, pour dérivée).

L'*ajustement* consiste à déterminer les valeurs optimales de K_P, K_I et K_D pour un système particulier. La valeur K_D/K_P est appelée *temps de dérivation* T_D et la valeur K_P/K_I est appelée *temps d'intégration* T_I. En général, les termes K_P, T_D et T_I sont précisés.

12.4.2 Contrôle avec un API

La Figure 12.13 montre une ligne d'un schéma à contacts qui permet de mettre en œuvre un contrôle en deux étapes. La sortie est activée lorsque la source A, la température réelle, est inférieure à la source B, la température souhaitée, c'est-à-dire la consigne.

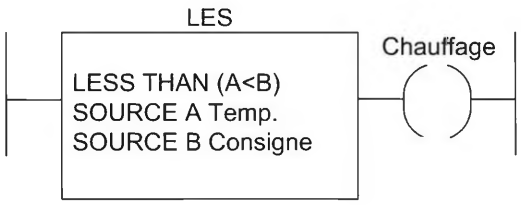
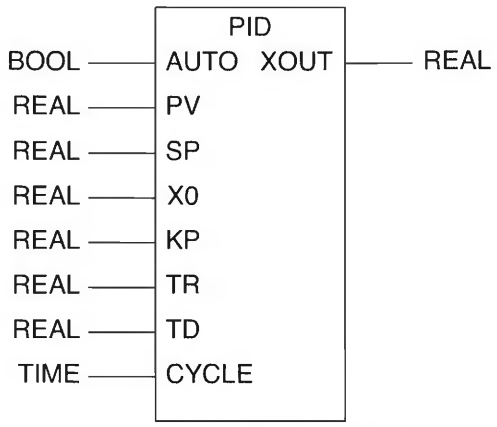


Figure 12.13 – Contrôle à deux étapes.

De nombreux API offrent les calculs PID sous forme d'un bloc fonctionnel standard. Il suffit alors de passer les paramètres souhaités, c'est-à-dire les valeurs de K_P, de K_I et de K_D, ainsi que l'emplacement des entrées-sorties, au travers du programme de l'API. La Figure 12.14 présente le symbole standard défini par la CEI 61131-3 pour la fonction de contrôle PID. Lorsque AUTO est à 1, le bloc calcule la valeur de sortie XOUT, qui permet à la variable de s'approcher de la consigne.



SP : valeur de consigne.
PV : variable à contrôler.
KP : constante de contrôle proportionnel.
TR : constante de temps d'intégration.
TD : constante de temps de dérivation.

REAL : nombres réels, c'est-à-dire des valeurs analogiques.
BOOL : booléens, c'est-à-dire des signaux tout ou rien.

Figure 12.14 – Symbole standard de la CEI 61131-3.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

12.5 En résumé

La gestion des données comprend des opérations qui déplacent ou transfèrent des informations numériques depuis un mot de mémoire vers un autre enregistré à un emplacement différent, qui comparent des valeurs et qui réalisent des opérations arithmétiques simples.

Le contrôle en boucle fermée d'une certaine variable est obtenu en comparant la valeur réelle de la variable à la valeur souhaitée, la consigne, et en produisant une sortie de manière à réduire la différence. Avec le contrôle proportionnel, le contrôleur envoie à l'actionneur une sortie proportionnelle à la différence entre la valeur réelle et la consigne, c'est-à-dire de l'erreur. Avec le contrôle PID, le contrôleur produit un signal de correction calculé à partir d'un élément proportionnel (le terme P), d'un élément lié aux valeurs précédentes de la variable (le terme I, pour intégrale) et d'un élément lié à la vitesse de changement de la variable (le terme D, pour dérivée).

12.6 Problèmes

Pour les questions 1 à 9, quatre réponses sont proposées : A, B, C et D. Choisissez la réponse correcte. Les problèmes 1 et 2 font référence à la Figure 12.15, qui présente deux formats employés pour les opérations de déplacement.

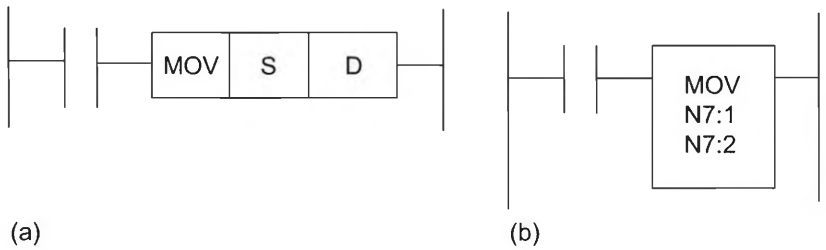


Figure 12.15 – Diagramme pour les problèmes 1 et 2.

1. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). À la Figure 12.15a, l'instruction du programme correspond à :
   (i) déplacer la valeur en S vers D, en laissant S vide ;
   (ii) copier la valeur en S et la placer dans D.

   Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
   A. (i) V (ii) V
   B. (i) V (ii) F
   C. (i) F (ii) V
   D. (i) F (ii) F
2. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). À la Figure 12.15b, l'instruction du programme correspond à :

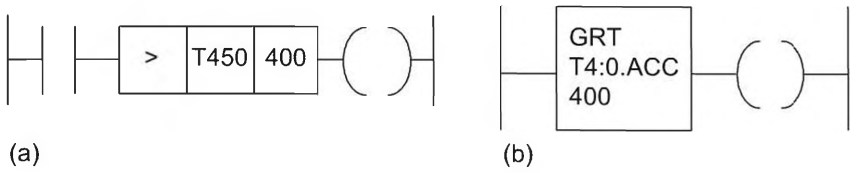
Copyright © 2015 Dunod.

(i) déplacer la valeur en N7:1 vers N7:2, en laissant N7:1 vide ;
(ii) copier la valeur en N7:1 et la placer dans N7:2.

Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?

A. (i) V (ii) V
B. (i) V (ii) F
C. (i) F (ii) V
D. (i) F (ii) F

Les problèmes 3 et 4 font référence à la Figure 12.16, qui présente deux versions d'un schéma à contacts avec une comparaison.



(a) (b)

Figure 12.16 – Diagramme pour les problèmes 3 et 4.

3. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). À la Figure 12.16a, l'instruction du programme produit une sortie :
   (i) lorsque la durée cumulée par le temporisateur T450 dépasse 400 ;
   (ii) jusqu'à ce que la durée cumulée par le temporisateur T450 atteigne 400.

   Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?

   A. (i) V (ii) V
   B. (i) V (ii) F
   C. (i) F (ii) V
   D. (i) F (ii) F

4. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). À la Figure 12.16b, l'instruction du programme produit une sortie :
   (i) lorsque la durée cumulée par le temporisateur T4:0 dépasse 400 ;
   (ii) jusqu'à ce que la durée cumulée par le temporisateur T4:0 atteigne 400.

   Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?

   A. (i) V (ii) V
   B. (i) V (ii) F
   C. (i) F (ii) V
   D. (i) F (ii) F

5. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). À la Figure 12.17, si les conditions d'entrée sont satisfaites, l'instruction du programme produit une sortie lorsque :
   (i) les données en N7:10 sont égales à celles en N7:20 ;
   (ii) les données en N7:10 sont inférieures à celles en N7:20.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

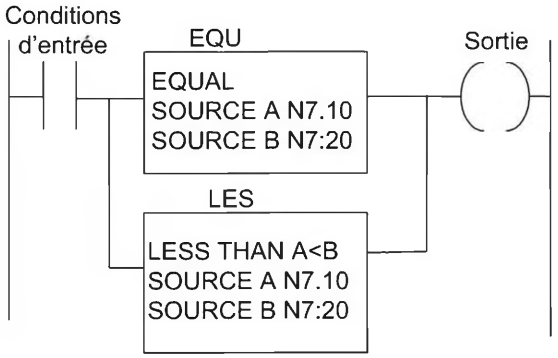


Figure 12.17 – Diagramme pour le problème 5.

Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?

A. (i) V (ii) V
B. (i) V (ii) F
C. (i) F (ii) V
D. (i) F (ii) F

6. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). À la Figure 12.18, si les conditions d'entrée sont satisfaites, l'instruction du programme produit une sortie lorsque :
(i) les données en N7:10 ne sont pas égales à celles en N7:20 ;
(ii) les données en N7:10 sont supérieures ou inférieures à celles en N7:20.

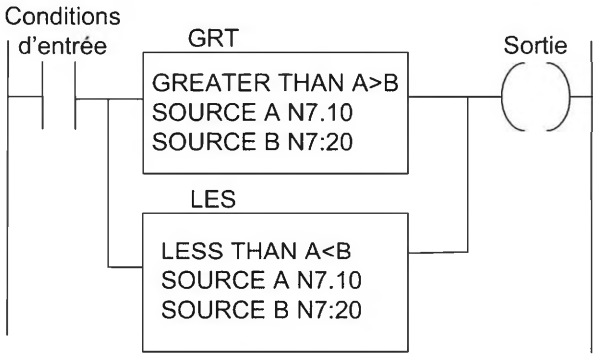


Figure 12.18 – Diagramme pour le problème 6.

Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?

A. (i) V (ii) V
B. (i) V (ii) F
C. (i) F (ii) V
D. (i) F (ii) F

7. À la Figure 12.19, si les conditions d'entrée sont satisfaites, l'instruction du programme calcule :

Copyright © 2015 Dunod.

A. La somme des données présentes sur les sources A et B.
B. Le produit des données présentes sur les sources A et B.
C. La différence entre les données présentes sur les sources A et B.
D. Le résultat de la division de la donnée présente sur la source A par celle présente en B

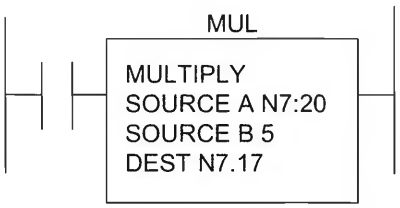


Figure 12.19 – Diagramme pour le problème 7.

8. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Pour le bloc fonctionnel Siemens de la Figure 12.20, la sortie est activée lorsque :
(i) les entrées IN1 et IN2 sont toutes deux le même nombre réel ;
(ii) l'entrée IN1 est un nombre réel supérieur à celui de l'entrée IN2.

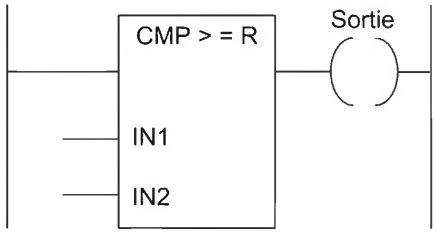


Figure 12.20 – Diagramme pour le problème 8.

Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
A. (i) V (ii) V
B. (i) V (ii) F
C. (i) F (ii) V
D. (i) F (ii) F

9. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Pour le bloc fonctionnel Siemens de la Figure 12.21, la sortie est activée lorsque :
(i) les entrées IN1 et IN2 sont toutes deux le même nombre réel ;
(ii) l'entrée IN1 est un nombre réel supérieur à celui de l'entrée IN2.

Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
A. (i) V (ii) V
B. (i) V (ii) F
C. (i) F (ii) V
D. (i) F (ii) F

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

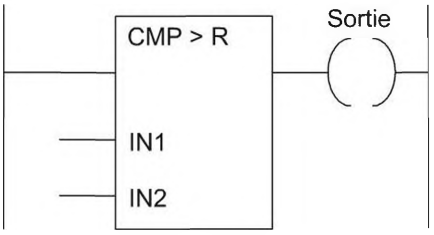


Figure 12.21 – Diagramme pour le problème 9.

10. Concevez les programmes en langage à contacts des systèmes qui mettent en œuvre les tâches suivantes :
    (a) Mettre en marche une pompe lorsque le niveau d'eau dans un réservoir dépasse 1,2 m et la couper lorsqu'il passe en dessous de 1,0 m.
    (b Mettre en marche une pompe, puis, 100 s plus tard, allumer une résistance de chauffage, puis, 30 s plus tard, mettre en marche le moteur de circulation.
    (c) Allumer un radiateur lorsque la température ambiante est inférieure à la température de référence.
    (d) Allumer une lampe lorsqu'une source de données est différente de 100.
11. Décrivez le fonctionnement d'un contrôleur tout ou rien et expliquez comment l'utiliser pour contrôler la température dans un système de chauffage central domestique.
12. Expliquez le principe d'un contrôleur proportionnel.

12.7 Recherches complémentaires

1. Déterminez les fonctions de gestion des données offertes par un modèle d'API.

Copyright © 2015 Dunod.

13 • CONCEPTION DES SYSTÈMES

Ce chapitre étudie la manière de concevoir les programmes et comment tester et trouver les défauts dans ces programmes et les systèmes à base d'API. Pour cela, les aspects matériels et logiciels doivent être pris en compte.

13.1 Développement d'un programme

Quel que soit le langage dans lequel un programme va être écrit, une approche systématique du problème permet d'augmenter les chances d'arriver à une solution de haute qualité en un temps aussi réduit que possible. La conception systématique se fonde sur les étapes suivantes :

1. La définition des besoins, en précisant les entrées et les sorties.
2. La définition de l'algorithme à utiliser. Un algorithme est une séquence pas à pas qui définit la méthode de résolution d'un problème. Il est souvent présenté sous forme d'un organigramme ou écrit dans un pseudo-code en utilisant les mots BEGIN, DO, END, IF-THEN-ELSE et WHILE-DO.
3. L'algorithme est ensuite converti en instructions, qui peuvent être saisies dans l'API. Puisque les programmes sont souvent très longs et peuvent se révéler difficiles à écrire sous forme d'un seul bloc et encore plus difficiles à déboguer et à maintenir ultérieurement, il est préférable de les décomposer en plusieurs niveaux de sections jusqu'à obtenir des blocs de taille gérable. Cette technique se nomme *conception descendante*.
4. Le programme est ensuite testé et débogué.
5. Le programme est documenté afin que les personnes qui l'utilisent ou qui doivent le modifier ultérieurement comprennent son fonctionnement.

13.1.1 Organigramme et pseudo-code

La Figure 13.1a présente les symboles utilisés dans les organigrammes. Chaque étape d'un algorithme est représentée par un ou plusieurs de ces symboles, qui sont reliés de manière à indiquer le flux du programme (voir Figure 13.1b). Le pseudo-code est une manière de décrire les étapes d'un algorithme de façon informelle.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

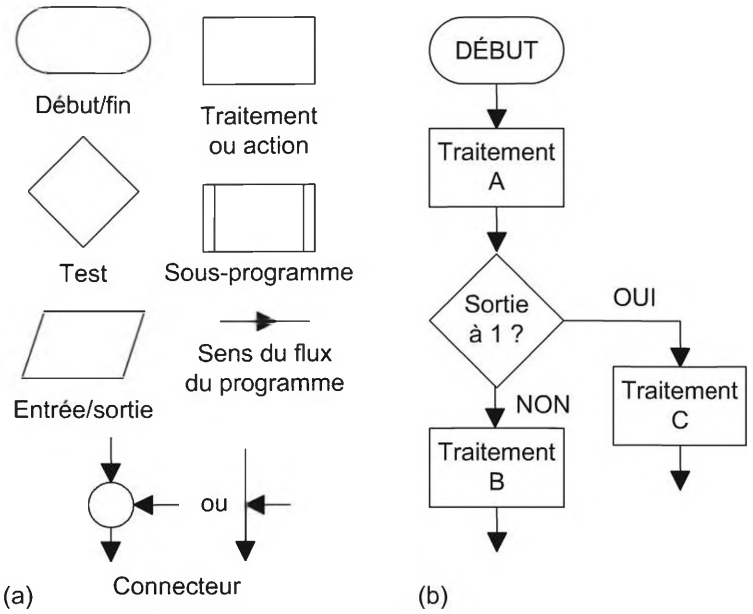


Figure 13.1 – (a) Symboles d'un organigramme et (b) exemple d'organigramme simple.

Voyons comment les fonctionnements suivants d'un programme peuvent être représentés par un organigramme et du pseudo-code, puis programmés à l'aide du langage à contacts et d'un graphe de fonction séquentielle :

– *Séquentiel.* Prenons une séquence dans laquelle l'événement A doit être suivi de l'événement B. La Figure 13.2a montre la représentation de ce fonctionnement séquentiel dans un organigramme. Voici le pseudo-code correspondant :

```
BEGIN A
    DO A
END A
BEGIN B
    DO B
END B
```

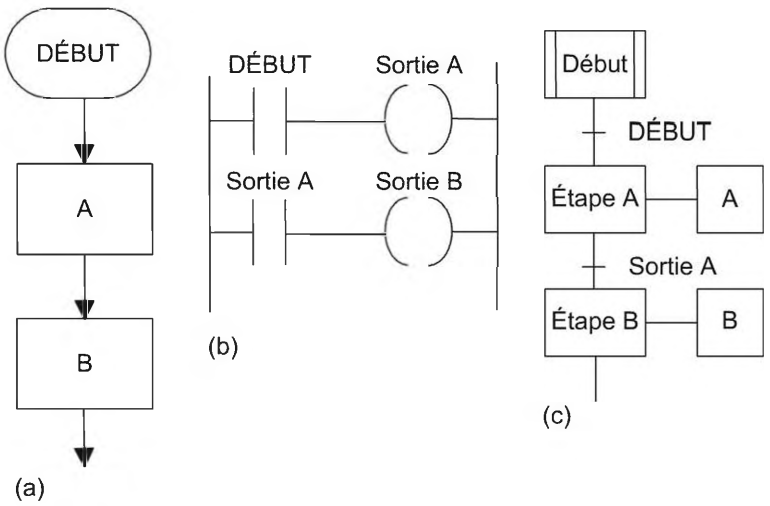


Figure 13.2 – Séquence.

Copyright © 2015 Dunod.

Une séquence peut être convertie en un schéma à contacts (voir Figure 13.2b). Lorsque l'entrée DÉBUT est présente, la sortie A est activée. Lorsque l'action A se produit, elle active le relais de la sortie A, qui active la sortie B. Le graphe de fonction séquentielle qui réalise une séquence est donné à la Figure 13.2c.

– *Conditionnel.* La Figure 13.3a montre l'organigramme d'un fonctionnement conditionnel. L'action A ou l'action B se produit si une condition X particulière vaut OUI ou NON. Le pseudo-code qui décrit le fonctionnement fait intervenir les mots IF-THEN-ELSE-ENDIF.

```
IF X
THEN
    BEGIN A
        DO A
    END A
ELSE
    BEGIN B
        DO B
    END B
ENDIF X
```

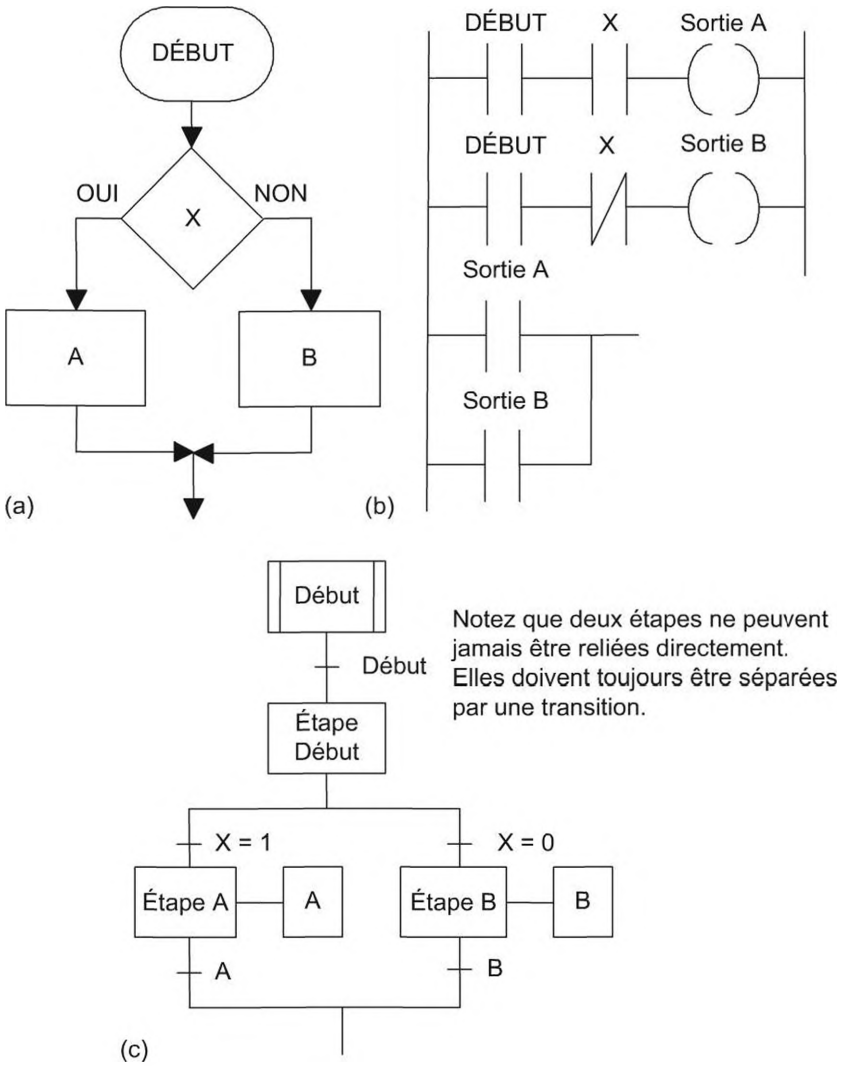


Figure 13.3 – Condition.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

Cette condition est représentée par le schéma à contacts de la Figure 13.3b. Lorsque l'entrée DÉBUT est présente, la sortie A est activée si une entrée est présente sur X, sinon la sortie B est activée. La Figure 13.3c donne le graphe de fonction séquentielle qui correspond à ce branchement.

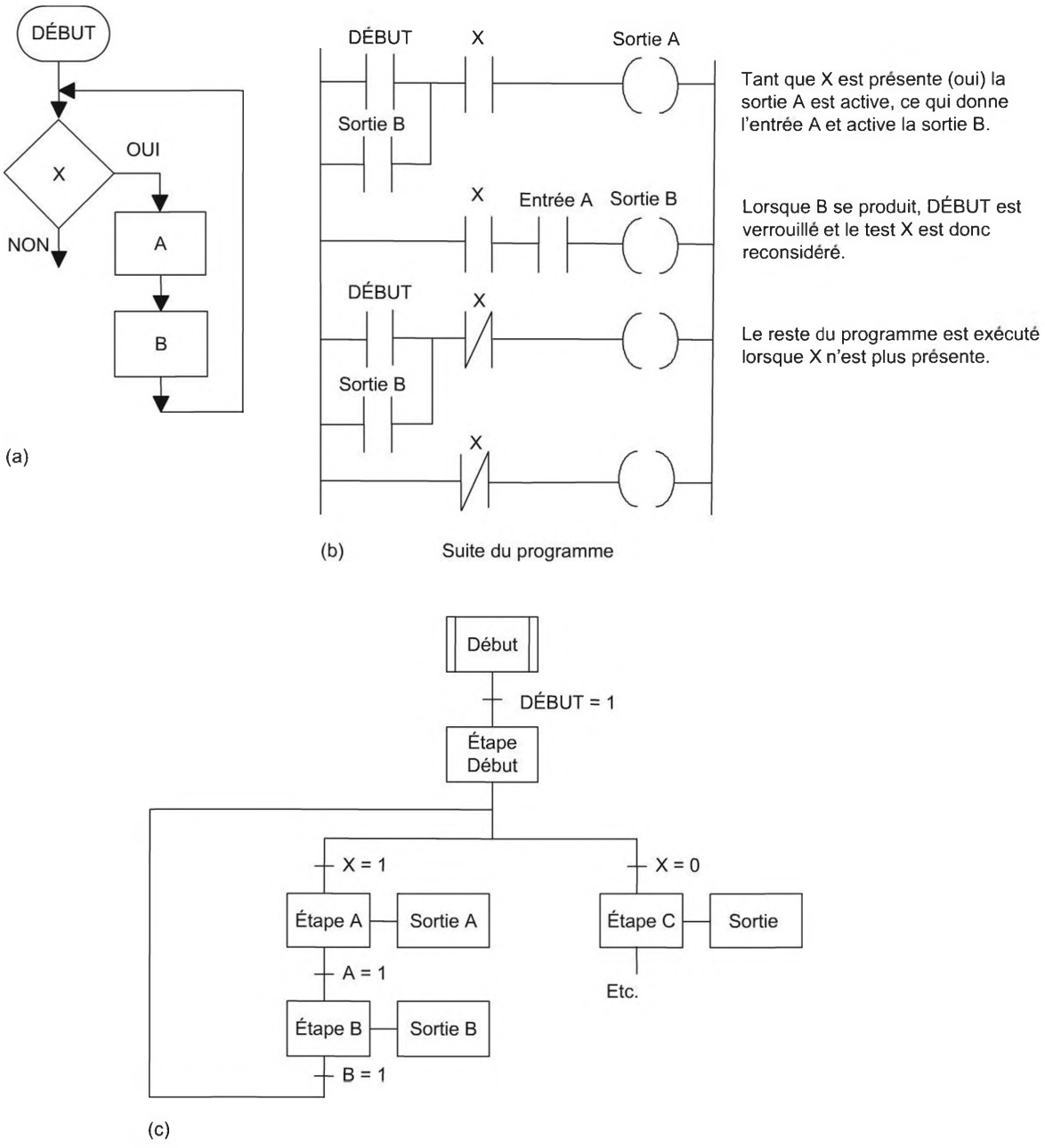


Figure 13.4 – Boucle.

– *Boucle.* Une boucle est une répétition d'une partie d'un programme qui se produit tant qu'une condition est satisfaite. La Figure 13.4a montre la représentation de cette répétition dans un organigramme. Tant que la condition X est remplie, la séquence action A suivie de l'action B se produit et se répète. Lorsque X n'est plus satisfaite, le programme se poursuit et la boucle sur les actions A et B s'interrompt. En pseudo-code, ce fonctionnement est représenté à l'aide des mots WHILE-DO-ENDWHILE :

Copyright © 2015 Dunod.

```
WHILE X
    BEGIN A
        DO A
    END A
    BEGIN B
        DO B
    END B
ENDWHILE X
```

La Figure 13.4b montre la mise en œuvre de cette idée dans un schéma à contacts fondé sur un relais interne. La Figure 13.4c présente le graphe de fonction séquentielle correspondant.

Lorsqu'une boucle doit être répétée un certain nombre de fois, il est possible d'utiliser un compteur. Il reçoit une impulsion d'entrée à chaque exécution de la boucle et désactive la séquence de la boucle lorsque le nombre d'exécutions requises est atteint (voir Figure 13.5).

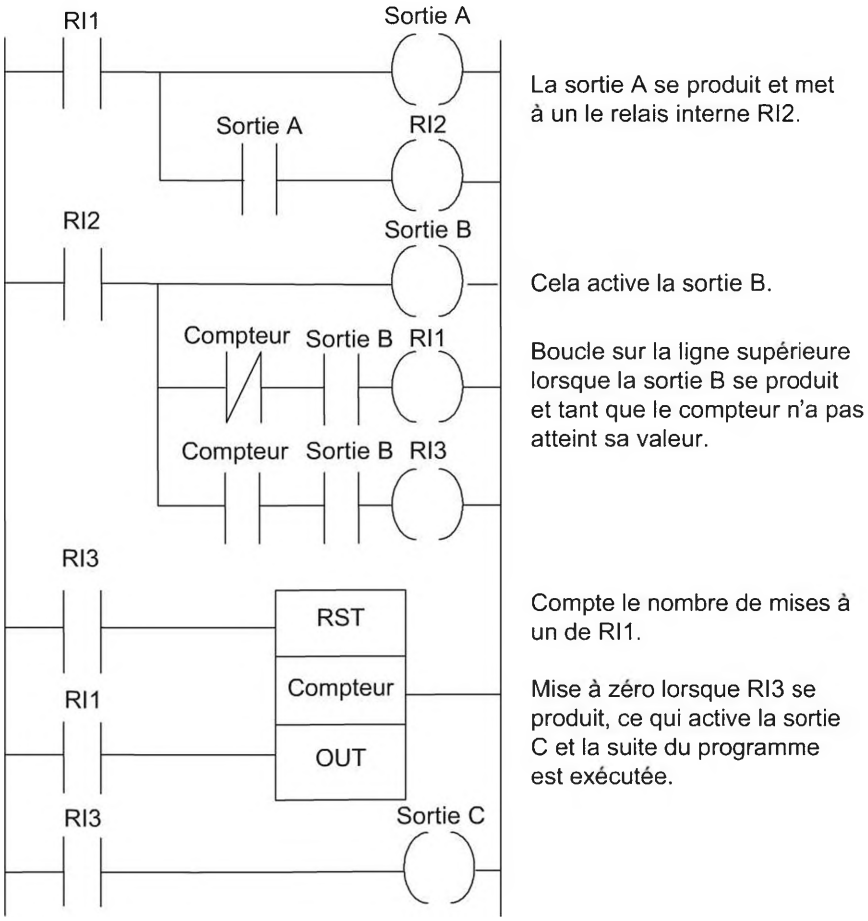


Figure 13.5 – Boucle avec compteur.

13.2 Systèmes sécurisés

La législation moderne en termes de sécurité demande aux employeurs de créer un lieu de travail sécurisé et exempt de risques sanitaires, de faire en sorte que les équipements sont sûrs et que des procédures de travail sécurisées soient établies et

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

suivies. Il est donc nécessaire d'estimer les risques du lieu de travail. Cela signifie rechercher les sources de danger, c'est-à-dire tout ce qui peut blesser, déterminer qui peut être blessé et comment, évaluer les risques qu'une personne soit blessée et si les précautions existantes sont adaptées ou si elles doivent être améliorées, rédiger et conserver les conclusions, et examiner et revoir les évaluations.

Par exemple, les problèmes d'arrêt d'urgence et d'accès à l'outil doivent être examinés, les risques évalués et les systèmes de sécurité conçus. Concernant les capots d'accès à l'outil, des contacts peuvent être utilisés de manière à stopper le système si les capots ne protègent pas correctement l'accès à l'outil.

La norme CEI 61508 traite de la sécurité fonctionnelle des systèmes électriques/électroniques et électroniques programmables (E/E/PE). Elle est composée de sept parties : Partie 1 : Prescriptions générales, Partie 2 : Prescriptions pour les systèmes E/E/PES relatifs à la sécurité, Partie 3 : Prescriptions concernant les logiciels, Partie 4 : Définitions et abréviations, Partie 5 : Exemple de méthodes de détermination des niveaux d'intégrité de sécurité, Partie 6 : Lignes directrices pour l'application des parties 2 et 3 et Partie 7 : Présentation de techniques et mesures. Pour obtenir une sécurité fonctionnelle d'une machine ou d'un équipement, le système de contrôle ou de protection relative à la sécurité doit fonctionner correctement et, lorsqu'une panne se produit, il doit faire en sorte que la machine ou l'équipement soit placé dans un état d'arrêt sûr.

13.2.1 Systèmes à API et sécurité

La sécurité doit être une priorité dans la conception d'un système fondé sur un API. Ainsi, les boutons d'arrêt d'urgence et les interrupteurs des dispositifs de sécurité doivent être câblés et ne pas dépendre du logiciel de l'API. Dans le cas d'un dysfonctionnement du bouton d'arrêt ou de l'API, le système est alors automatiquement sécurisé ; le système doit présenter une sécurité intrinsèque. Lors d'une panne, les sorties doivent passer en mode de sécurité intrinsèque afin que personne ne risque d'être blessé. Par exemple, les protections d'une machine ne doivent pas être ouvertes ou pouvoir être ouvertes lorsque l'API est en panne.

Dans un système à base d'API, un signal d'arrêt peut être fourni par un interrupteur (voir Figure 13.6). Cette mise en œuvre de l'arrêt d'urgence n'est pas sûre car, en cas de panne et d'impossibilité d'actionner l'interrupteur, aucun signal d'arrêt n'est fourni. La mise en marche est déclenchée par une fermeture momentanée du bouton-poussoir de démarrage et le relais interne de commande du moteur verrouille cette fermeture pour que la sortie reste active. Pour l'arrêt, nous devons ouvrir momentanément l'interrupteur d'arrêt ; cela déverrouille l'interrupteur de démarrage. Toutefois, si le bouton d'arrêt ne peut pas être actionné, nous sommes dans l'incapacité d'arrêter le système. Nous avons besoin d'un système qui pourra être stoppé en cas de dysfonctionnement du bouton d'arrêt.

Nous pouvons obtenir ce fonctionnement à l'aide du montage de la Figure 13.7. Le programme voit l'interrupteur d'arrêt avec des contacts ouverts. Cependant, puisque le bouton d'arrêt physique dispose de contacts normalement fermés, le

Copyright © 2015 Dunod.

programme signale la fermeture des contacts du programme. Un appui sur le bouton d'arrêt ouvre les contacts du programme et stoppe le système.

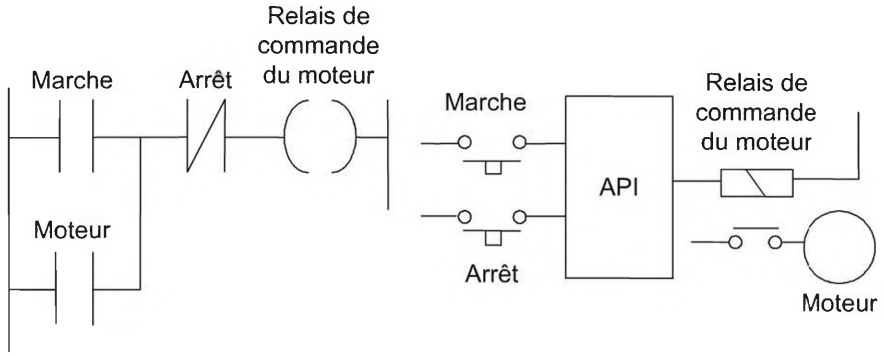


Figure 13.6 – Système d'arrêt non sûr.

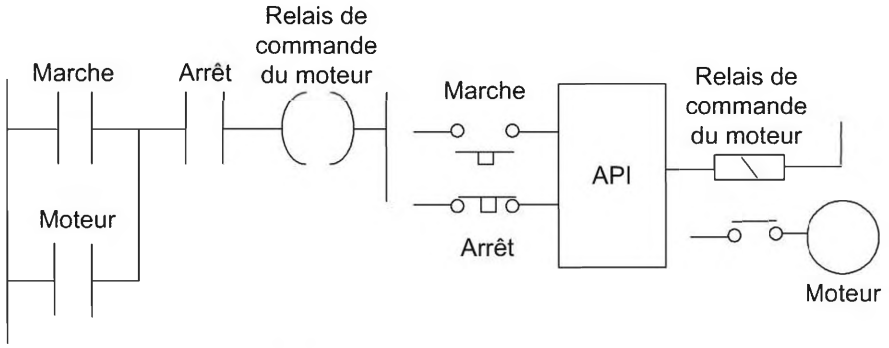


Figure 13.7 – Système d'arrêt plus sûr.

Un système d'arrêt d'urgence sûr doit être conçu de manière à fournir un signal d'arrêt en cas de défaut et d'impossibilité d'actionner l'interrupteur. Puisque l'API peut présenter un dysfonctionnement, l'arrêt d'urgence doit également fonctionner indépendamment de l'API. Si l'arrêt d'urgence se trouve en entrée de l'API, le système n'est pas sûr (voir Figure 13.8).

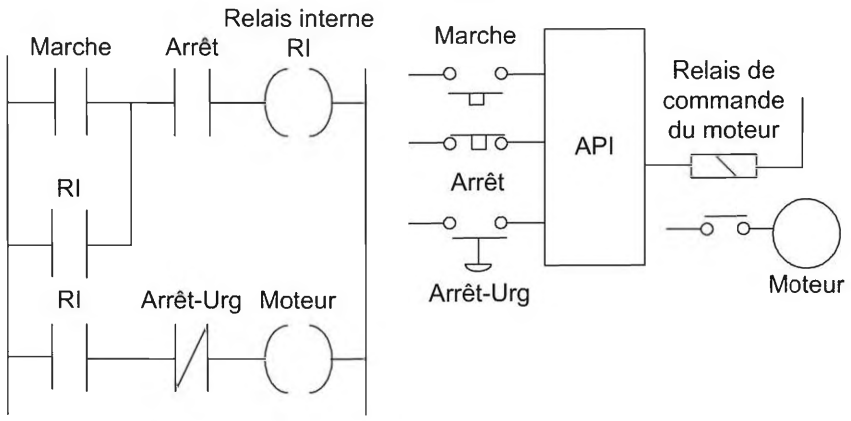


Figure 13.8 – Arrêt d'urgence non sûr.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

La Figure 13.9 présente un système plus sûr dans lequel le bouton d'arrêt d'urgence est câblé en sortie. L'appui sur ce bouton arrête le moteur en fonctionnement. Lorsque le bouton d'arrêt est relâché, le moteur ne peut pas redémarrer car les contacts du relais interne ont été déverrouillés.

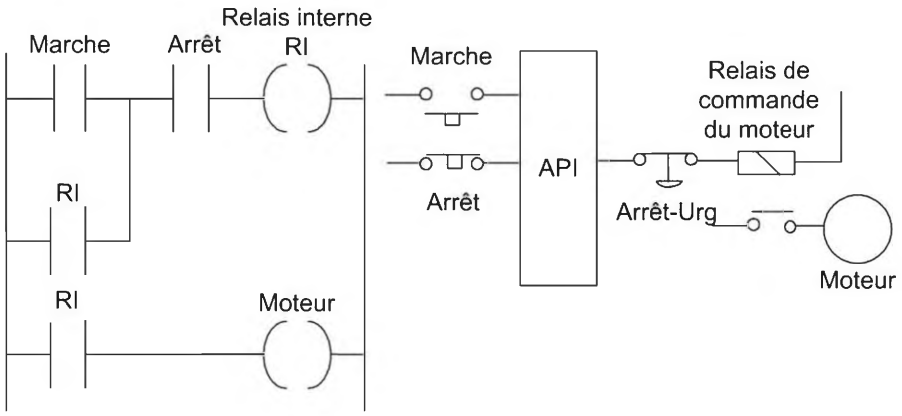


Figure 13.9 – Arrêt d'urgence plus sûr.

13.2.2 Relais d'arrêt d'urgence

Les *relais d'arrêt d'urgence* sont largement utilisés dans les mécanismes d'arrêt d'urgence, comme le modèle PNOZ p1p de Pilz GmbH & Co. Ce dispositif est équipé de LED pour indiquer l'état des circuits d'entrée et de sortie, du circuit de mise à zéro, de l'alimentation et des défauts. L'unité de base peut être connectée à un API *via* un module d'interface afin que l'automate puisse accéder aux informations d'état. Cette interface utilise des photocoupleurs pour isoler la sortie du relais d'arrêt d'urgence et l'entrée de l'API (voir Figure 1.12). Ainsi, bien que l'arrêt d'urgence soit indépendant de l'API, il peut lui fournir des signaux afin, par exemple, d'initier des procédures d'arrêt sécurisé. La Figure 13.10 illustre cette idée.

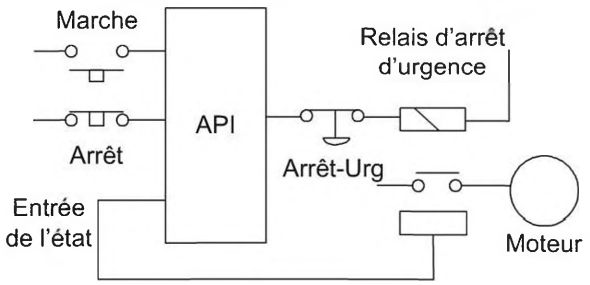


Figure 13.10 – Relais d'arrêt d'urgence qui indique son état.

L'utilisation d'un bouton d'arrêt d'urgence pour couper le circuit de commande du relais d'arrêt d'urgence, provoquant sa désactivation et la coupure de l'alimentation (voir Figure 13.11a), ne suffit pas. En effet, si les contacts du relais se collent ensemble, l'arrêt d'urgence ne fonctionne plus. Pour contourner ce problème, nous pouvons utiliser un fonctionnement en double voie dans lequel deux contacts normalement ouverts sont placés en série et sont coupés par la désactivation du relais (voir Figure 13.11b). La sécurité peut encore être améliorée

Copyright © 2015 Dunod.

en plaçant trois contacts en série. L'un d'eux possède des contacts normalement fermés, les autres, des contacts normalement ouverts. Ensuite, l'un des ensembles de contacts doit être éteint et les deux autres excités.

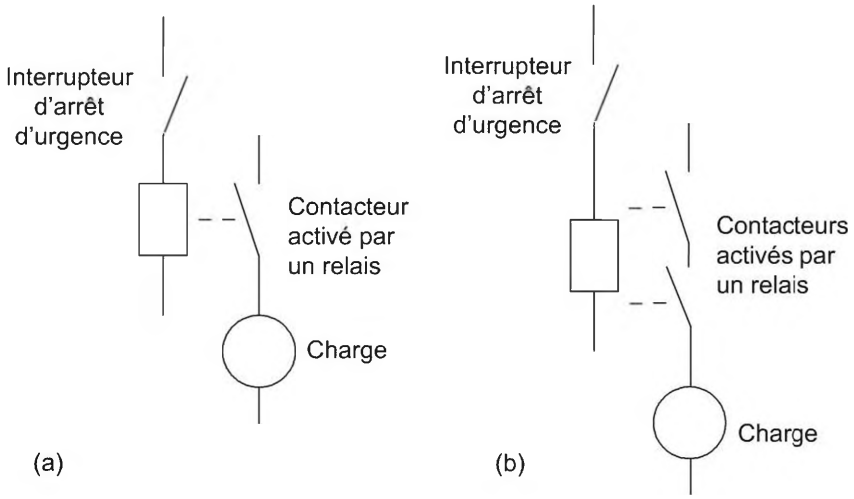


Figure 13.11 – Relais d'arrêt d'urgence : (a) monovoie et (b) double voie.

13.2.3 Fonctions de sécurité

Lors de la conception d'un système de commande, il est essentiel que l'opérateur ne puisse pas entrer en contact avec la machine pendant qu'elle fonctionne. Cela peut impliquer les éléments suivants :

- Une *commande à deux mains* pour que les deux mains de l'opérateur soient en permanence sur les interrupteurs et que la machine s'arrête si seul l'un des interrupteurs est actionné.
- Une *surveillance du capot de protection* pour empêcher l'accès à la machine pendant son fonctionnement. Pour cela, des dispositifs de sécurité peuvent être utilisés, par exemple des portes et des grilles. Des interrupteurs de fin de course positionnés sur les verrous de la porte ou de la grille peuvent détecter son ouverture et déclencher l'arrêt de la machine. Toutefois, l'opérateur peut facilement désactiver ces interrupteurs de fin de course, par exemple en plaçant un tournevis sur les contacts pour forcer le fonctionnement de la machine. Des systèmes de sécurité plus sophistiqués doivent donc être imaginés, comme des détecteurs de proximité et des verrous à clé.
- Des *barrières immatérielles* pour empêcher quiconque d'approcher de la machine. Une zone de danger, par exemple une machine d'emballage, peut être équipée de faisceaux infrarouges pour éviter que le personnel ne s'en approche trop près. Lorsqu'un faisceau lumineux est coupé, il déclenche immédiatement une commande d'arrêt sécurisé.
- Les *planchers sensibles de sécurité* constituent un autre moyen de détecter la présence d'une personne près d'une machine. Ils sont placés autour de la machine et lorsqu'une personne marche sur le plancher, un contact est fermé et déclenche l'arrêt de la machine.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

– Les *relais d'arrêt d'urgence* permettent d'arrêter la machine lorsqu'un événement d'urgence se produit (voir Section 13.2.2).

Un système de fonctionnement sécurisé pour une cellule de travail peut se fonder sur des systèmes d'entrée protégés, par exemple des capots qui activent des relais d'arrêt lorsqu'ils ne sont pas en place, des barrières immatérielles et des relais d'arrêt d'urgence.

13.2.4 API de sécurité

Les API de sécurité sont spécialement conçus pour assurer des fonctions de sécurité. Dans un tel automate, deux ou trois microprocesseurs mettent en œuvre la même logique, se consultent et produisent des sorties uniquement s'ils sont d'accord. L'une des méthodes employées se fonde sur un système double voie avec deux sous-systèmes identiques qui communiquent au travers d'une liaison à fibre optique. Les entrées des capteurs sont envoyées simultanément aux deux sous-systèmes. Pendant le fonctionnement de l'API, des données sont échangées entre les deux sous-systèmes *via* la fibre optique. Les deux sous-systèmes opèrent de manière synchrone avec le même programme, comparent les signaux d'entrées et de sorties, les résultats des opérations logiques, les compteurs, etc., et passent automatiquement en condition d'arrêt sécurisé en cas de divergence des sorties, de défauts internes ou de pannes. Pour les sorties numériques associées à la sécurité, les actionneurs sont activés ou désactivés à partir des deux sous-systèmes. Autrement dit, un seul sous-système peut arrêter l'équipement.

13.3 Mise en service

La mise en service d'un système à base d'API implique les étapes suivantes :

1. Vérifier que toutes les connexions par câble entre l'API et l'équipement contrôlé sont complètes, sûres, conformes à la spécification et aux normes locales.
2. Vérifier que l'alimentation correspond au paramètre de tension configuré sur l'API.
3. Vérifier que tous les dispositifs de protection sont configurés avec les réglages de déclenchement appropriés.
4. Vérifier que les boutons d'arrêt d'urgence fonctionnent.
5. Vérifier que tous les dispositifs d'entrées-sorties sont connectés aux points d'entrées-sorties corrects et produisent les signaux appropriés.
6. Charger et tester le logiciel.

13.3.1 Test des entrées et des sorties

Les dispositifs d'entrées, tels que les interrupteurs, peuvent être manipulés pour obtenir les conditions d'ouverture et de fermeture des contacts et constater l'activation des LED correspondantes sur le module d'entrée. Les LED doivent

Copyright © 2015 Dunod.

s'allumer lorsque les entrées sont fermées et s'éteindre lorsqu'elles sont ouvertes. Si une LED ne s'allume pas, le dispositif d'entrée peut ne pas fonctionner correctement, les connexions aux modules d'entrées peuvent être incorrectes, le dispositif d'entrée peut ne pas être correctement alimenté ou la LED ou le module d'entrée peut être défectueux. Pour les dispositifs de sorties qu'il est possible de démarrer de manière sécurisée, des boutons-poussoirs peuvent avoir été installés afin d'effectuer le test de chaque sortie.

Une autre méthode de test des entrées-sorties implique un *forçage*. Elle se fonde sur un logiciel, non une mise en marche ou un arrêt mécanique, avec des instructions affichées à l'écran pour activer ou désactiver des entrées-sorties. Le forçage écrase donc une condition d'entrée ou de sortie. Cela permet non seulement de tester le fonctionnement des entrées et des sorties, mais également de vérifier qu'elles sont correctement reliées à l'API.

Par exemple, avec le système S7-1200 de Siemens, il suffit de cliquer sur Force Table dans le menu Watch and Force pour saisir l'adresse de l'élément à forcer, puis la valeur à appliquer, TRUE ou FALSE. Dans le cas de l'API FX de Mitsubishi, la sélection de l'outil Forced I/O se fait en appuyant sur des touches. L'adresse de l'entrée ou de la sortie à forcer est ensuite saisie dans la fenêtre Set/Reset qui s'affiche, puis la valeur on ou off doit être précisée. L'état de la sortie ou de l'entrée est alors fixé à la valeur de forçage choisie et les résultats de cette action peuvent être observés. La Figure 13.12a illustre le forçage de la sortie Q0.1. Cette opération conduit à l'activation de la sortie Q0.3. La Figure 13.12b montre les effets du forçage de l'entrée I0.1. Pour mettre fin au forçage, il faut cliquer sur l'icône Stop Forcing ou utiliser l'instruction correspondante. La Figure 13.13 montre comment employer le forçage avec un programme SFC. Il est possible de forcer une transition, ce qui écrasera les conditions de la transition chaque fois que le programme arrivera à ce point. Lorsqu'une transition est forcée à l'état faux dans un branchement simultané (voir Figure 13.13a), le programme reste dans la branche correspondante tant que le forçage est actif, ce qui empêche d'atteindre la dernière étape de la branche. Une autre possibilité est de forcer un chemin simultané (voir Figure 13.13b). Cela empêche l'exécution d'un chemin et donc d'un ou plusieurs chemins d'un branchement simultané. Le forçage requiert une grande prudence et d'énormes précautions car des entrées et des sorties quelconques sont appliquées. Il est donc possible d'endommager les appareils ou de blesser des personnes. Par ailleurs, lorsque le forçage est arrêté, des éléments peuvent rester dans l'état qui avait été imposé.

13.3.2 Test logiciel

La plupart des API disposent d'une sorte de programme de test des logiciels. Il vérifie que les programmes installés utilisent les adresses correctes et affiche ou imprime une liste de tous les points d'entrées-sorties employés, des paramètres des compteurs et des temporisateurs, etc., ainsi que les erreurs détectées. Par exemple, un message pourrait signaler qu'une adresse de sortie est utilisée plusieurs fois dans le programme, qu'un temporisateur ou un compteur est employé sans valeur de présélection, qu'un compteur est utilisé sans réinitialisation, etc.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

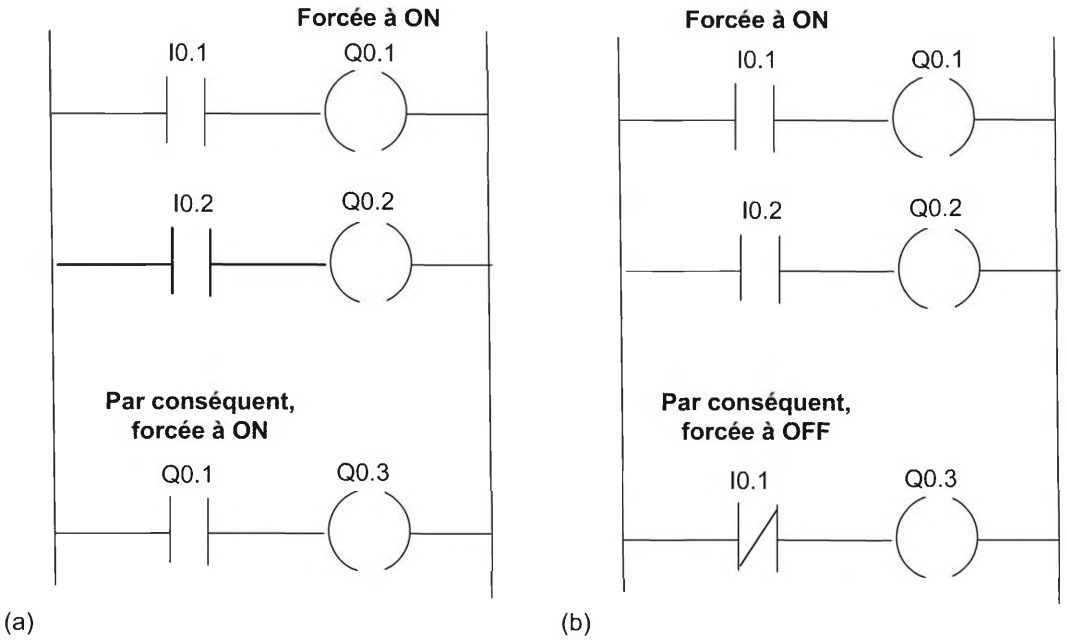


Figure 13.12 – Schéma à contacts pour le forçage (a) d'une sortie et (b) d'une entrée.

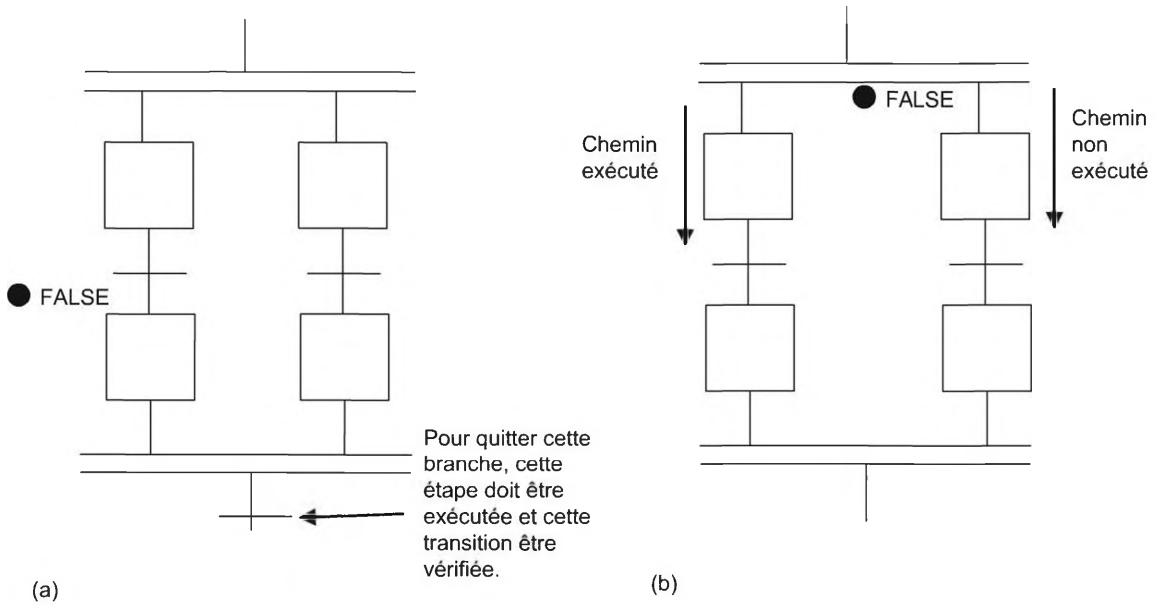


Figure 13.13 – Forçage d'une entrée.

13.3.3 Simulation

De nombreux API peuvent être utilisés avec une unité de simulation capable de lire et d'écrire des informations directement dans la mémoire d'entrées-sorties, et ainsi de simuler les actions des entrées et des sorties. Le programme installé peut donc être exécuté et les entrées-sorties simulées de manière à les vérifier, ainsi que les valeurs de présélection. Pour mener à bien ce type d'opération, le terminal doit être placé dans le mode approprié. Pour Mitsubishi, il s'agit du *mode monitoring*, pour Siemens, du *mode de test*, et pour Telemecanique, du *mode de débogage*.

Copyright © 2015 Dunod.

Pour un API de Mitsubishi en mode monitoring, la Figure 13.14 montre comment se présentent les entrées lorsqu'elles sont ouvertes et fermées et comment se présentent les sorties lorsqu'elles sont désactivées et activées. L'écran affiche une partie sélectionnée du schéma à contacts et le résultat de l'exécution du programme. Ainsi, à un certain stade du programme, l'écran peut apparaître comme l'illustre la Figure 13.15a. Pour la ligne 12, avec des entrées sur X400, X401 et X402, mais non sur M100, il n'existe aucune sortie sur Y430. Pour la ligne 13, les contacts du temporisateur T450 sont fermés, le bas de l'écran indiquant que la temporisation de T450 est écoulée. Puisque Y430 n'est pas excité, ses contacts Y430 sont ouverts et la sortie est absente sur Y431. Si nous forçons une entrée sur M100, l'affichage change (voir Figure 13.15b). À présent, les sorties Y430 et, par conséquent, Y431 sont activées.

Figure 13.14 – Symboles du mode de supervision.

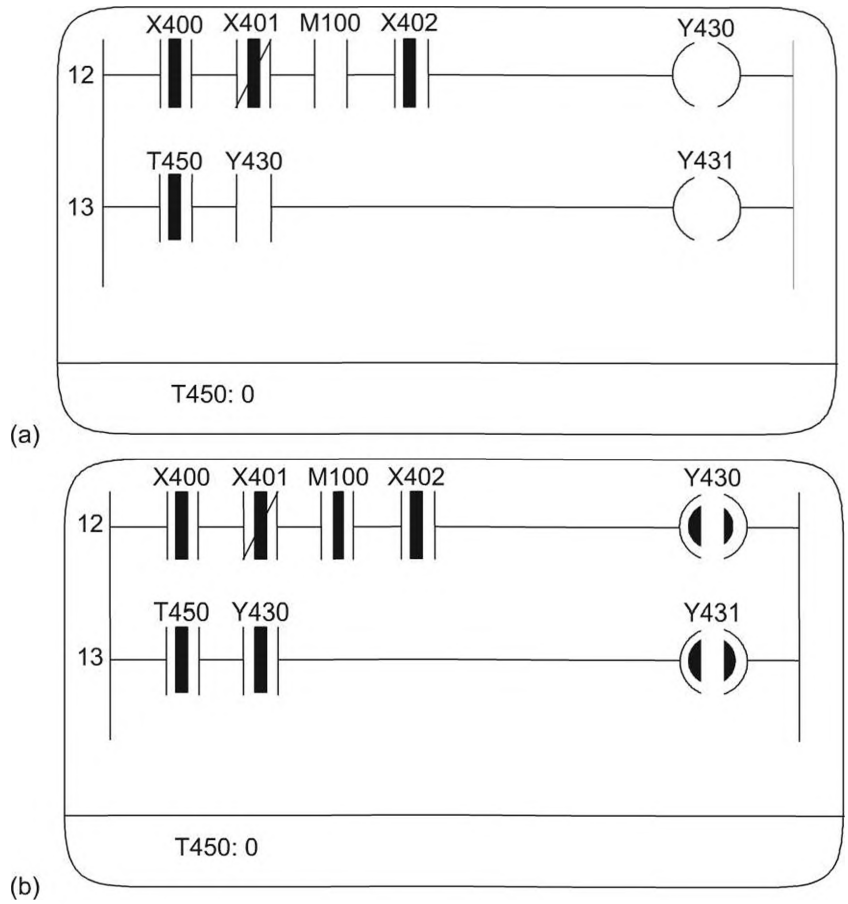


Figure 13.15 – Monitoring du programme à contacts.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

13.4 Recherche des défauts

Pour tout équipement commandé par un API, les défauts proviennent en grande majorité des capteurs, des actionneurs et du câblage, non de l'API lui-même. Parmi les dysfonctionnements possibles des API, les plus probables concernent les canaux d'entrées-sorties et l'alimentation, non le CPU.

Afin d'illustrer un défaut, prenons l'exemple d'un dispositif de sortie qui ne s'active pas, même si la LED de sortie indique le contraire. Si le contrôle de la sortie de l'API indique une tension normale, le défaut peut provenir d'un câble ou d'un dispositif. Si le contrôle de la tension au niveau du dispositif se révèle normal, le dispositif est à mettre en cause. Examinons également un dysfonctionnement des entrées. Il peut provenir d'un court-circuit ou d'un problème de masse sur une entrée. Pour isoler les défauts, une méthode consiste à déconnecter les entrées une à une, jusqu'à déterminer l'entrée défectueuse. Par ailleurs, le défaut peut être un arrêt total du système. Il peut provenir d'un problème d'alimentation, quelqu'un l'ayant coupée, ou du déclenchement d'un disjoncteur.

De nombreux API disposent de procédures d'analyse des défauts qui procèdent à des autotests et affichent des codes de défaut, parfois avec un court message qui peut être compris en recherchant le code dans une liste afin de connaître la source du défaut et les solutions de réparation. Par exemple, le code peut indiquer que le défaut vient d'un certain module et que la méthode de réparation est « Remplacer le module » ou, peut-être, « Couper l'alimentation et la rétablir ».

13.4.1 Techniques de détection des défauts

Voici quelques techniques classiques qui permettent de détecter les défauts :

– *Vérification du minutage*. Le terme *chien de garde* désigne une vérification du minutage réalisée par l'API afin de contrôler qu'une fonction a été menée à bien en un temps normal. Si la fonction a demandé un temps plus long que la normale, un défaut est supposé et le temporisateur du chien de garde se déclenche, activant une alarme et arrêtant sans doute l'API. Pendant les diagnostics internes effectués par l'API, le temporisateur du chien de garde sert à détecter les défauts. Il est initialisé à une durée légèrement plus longue que le temps de scrutation normale. Il est démarré au début de chaque analyse du programme et, si le temps de cycle est normal, il n'expire pas et est remis à zéro à la fin du cycle, paré pour le suivant. En revanche, si le temps de cycle est plus long qu'il ne devrait, la temporisation de chien de garde est écoulée avant la fin du cycle et signale un défaut dans le système.

  À l'intérieur d'un programme, des lignes à contacts supplémentaires sont souvent incluses pour démarrer un temporisateur au début d'une fonction. Si elle est terminée avant la fin de la temporisation, le programme se poursuit. Dans le cas contraire, une commande de saut passe à d'autres lignes, qui déclenchent une alarme et peuvent arrêter le système. La Figure 13.16 montre un exemple de temporisateur de chien de garde qui peut être employé avec le déplacement d'un piston dans un vérin. Lorsque l'interrupteur de mise en marche est fermé, le solénoïde d'une vanne est excité et déclenche le déplace-

Copyright © 2015 Dunod.

ment du piston. Il démarre également le temporisateur. Lorsque le piston est totalement sorti, il actionne un interrupteur de fin de course et stoppe le temporisateur. Si l'action du piston sur l'interrupteur se fait en un temps plus long que la valeur présélectionnée du temporisateur, l'alarme est déclenchée.

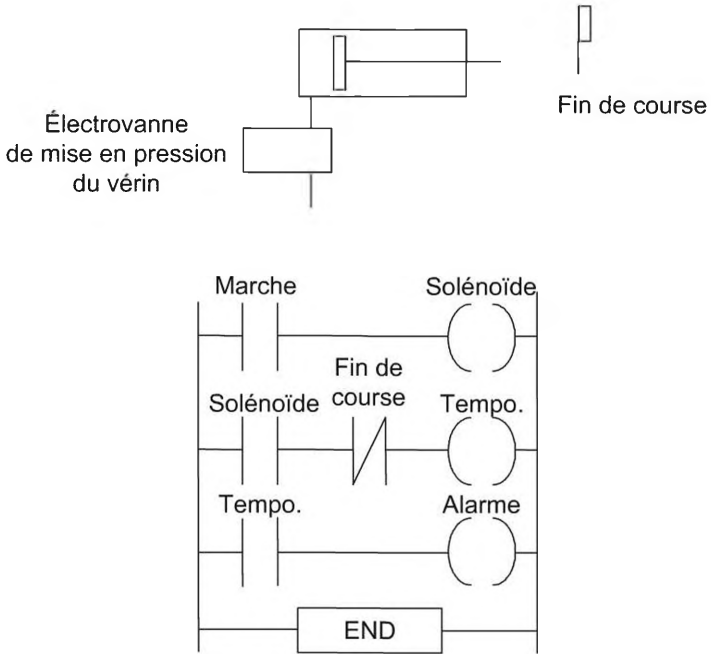


Figure 13.16 – Chien de garde.

– *Dernière sortie activée*. Cette technique se fonde sur l'utilisation de témoins d'état qui indiquent la dernière sortie ayant été activée au cours d'un processus qui s'est arrêté. Ces témoins sont intégrés au programme de manière à être allumés lorsque les sorties se produisent. Ils indiquent donc les sorties qui sont activées. Le programme est conçu pour éteindre les témoins d'état précédents et pour allumer un nouveau témoin d'état à chaque activation d'une nouvelle sortie. La Figure 13.17 illustre ce principe.

  Une telle technique peut se révéler lourde dans un grand système avec de nombreuses sorties. Dans ce cas, les sorties sont regroupées et un témoin d'état est utilisé pour chaque groupe. Un sélecteur permet ensuite de choisir chaque entrée dans chaque groupe afin de savoir si elle est active. La Figure 13.18 illustre cette idée.

  La Figure 13.19 illustre l'utilisation de cette technique pour indiquer la dernière action qui a eu lieu en présentant un programme qui peut être utilisé avec un système pneumatique pour actionner des vérins en séquence. Il indique à quel stade de la séquence un défaut, par exemple le blocage d'un piston, s'est produit. Il doit être ajouté au programme principal de commande des vérins. À chaque mouvement du vérin est associé une LED. Le dernier qui a été réalisé est indiqué par l'allumage de sa LED.

– *Réplication*. Lorsque la question de la sécurité est soulevée lors de l'apparition d'un défaut, des contrôles peuvent être constamment mis en œuvre pour détecter les dysfonctionnements. Une technique consiste à répliquer les vérifi-

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

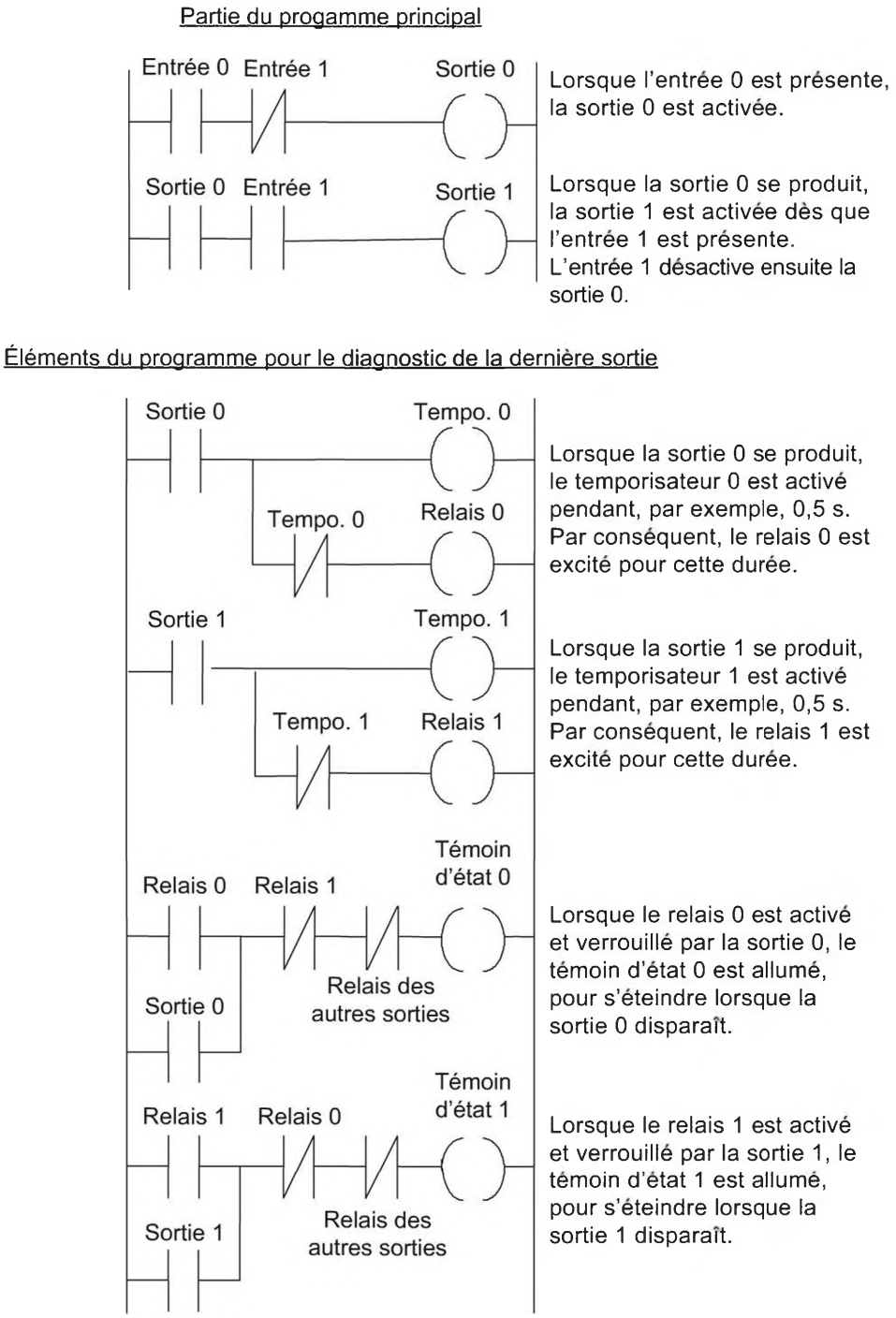


Figure 13.17 – Programme de diagnostic de la dernière sortie activée.

cations, ce qui implique la duplication, c'est-à-dire la redondance, du système API. Cela peut signifier que le système répète deux fois chaque opération et, s'il obtient le même résultat, suppose qu'il n'y a pas de défaut. Cette procédure permet de détecter les défauts temporaires. Une solution plus coûteuse consiste à dupliquer les systèmes API et à comparer les résultats qu'ils produisent. En l'absence de défaut, les deux résultats doivent être identiques. Un défaut apparaît sous forme d'une différence.

Copyright © 2015 Dunod.

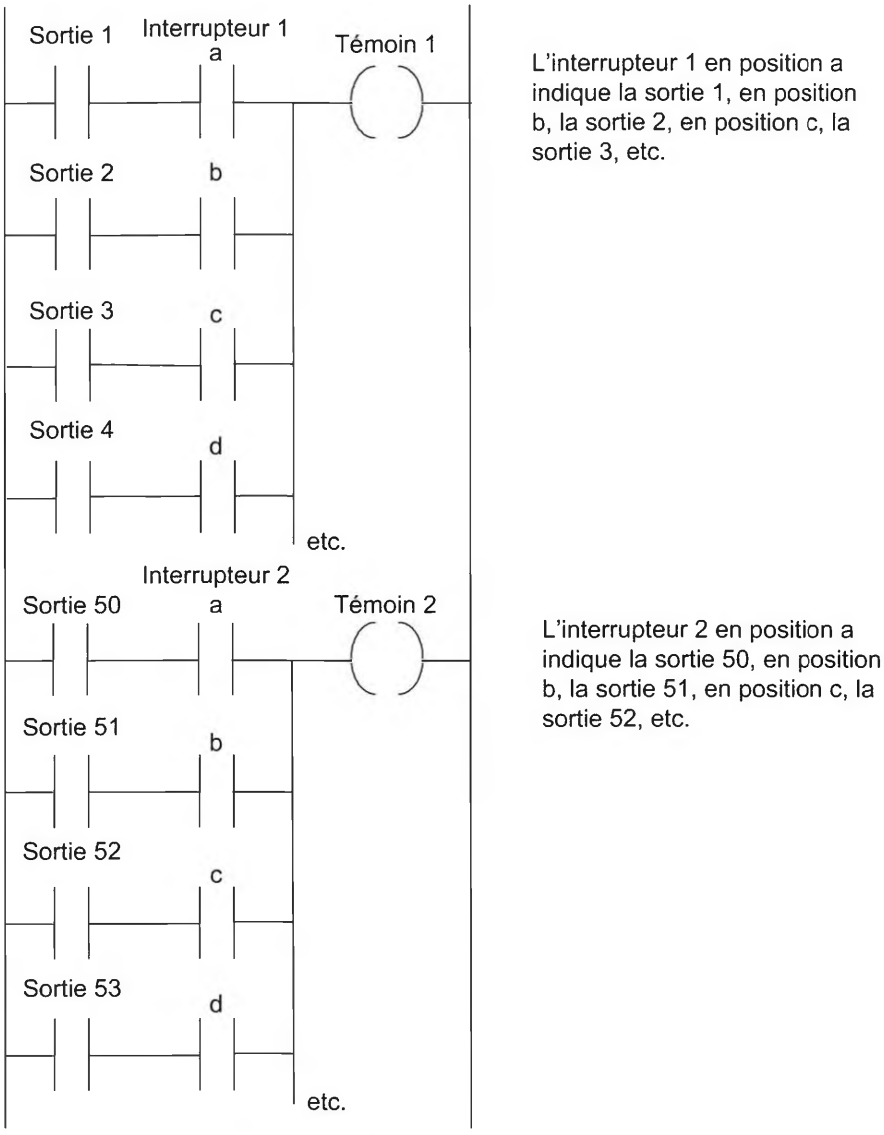


Figure 13.18 – Un seul témoin d'état pour un groupe de sorties.

– *Vérification d'une valeur attendue.* Des erreurs logicielles peuvent être détectées en vérifiant si une valeur attendue est obtenue lorsqu'une certaine entrée se produit. Si la valeur attendue n'est pas obtenue, un défaut est supposé.

13.4.2 Stockage du programme

Des programmes applicatifs peuvent être chargés dans la mémoire RAM sauvegardée par batterie d'un API. Un dysfonctionnement de la batterie peut conduire à la perte totale des programmes mémorisés. Une solution consiste à stocker les programmes des applications dans une EPROM, un type de mémoire qui n'est pas sujet aux coupures d'alimentation. Pour se protéger des défauts possibles de la mémoire et de la perte du programme stocké, une copie de chaque programme applicatif doit être conservée. Si le programme a été développé sur un ordinateur, la copie peut se trouver sur un CD-ROM ou un disque dur. Sinon, la sauvegarde peut être placée sur une cartouche EPROM. Le programme peut ainsi être de nouveau envoyé sur l'API sans avoir à le récrire.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

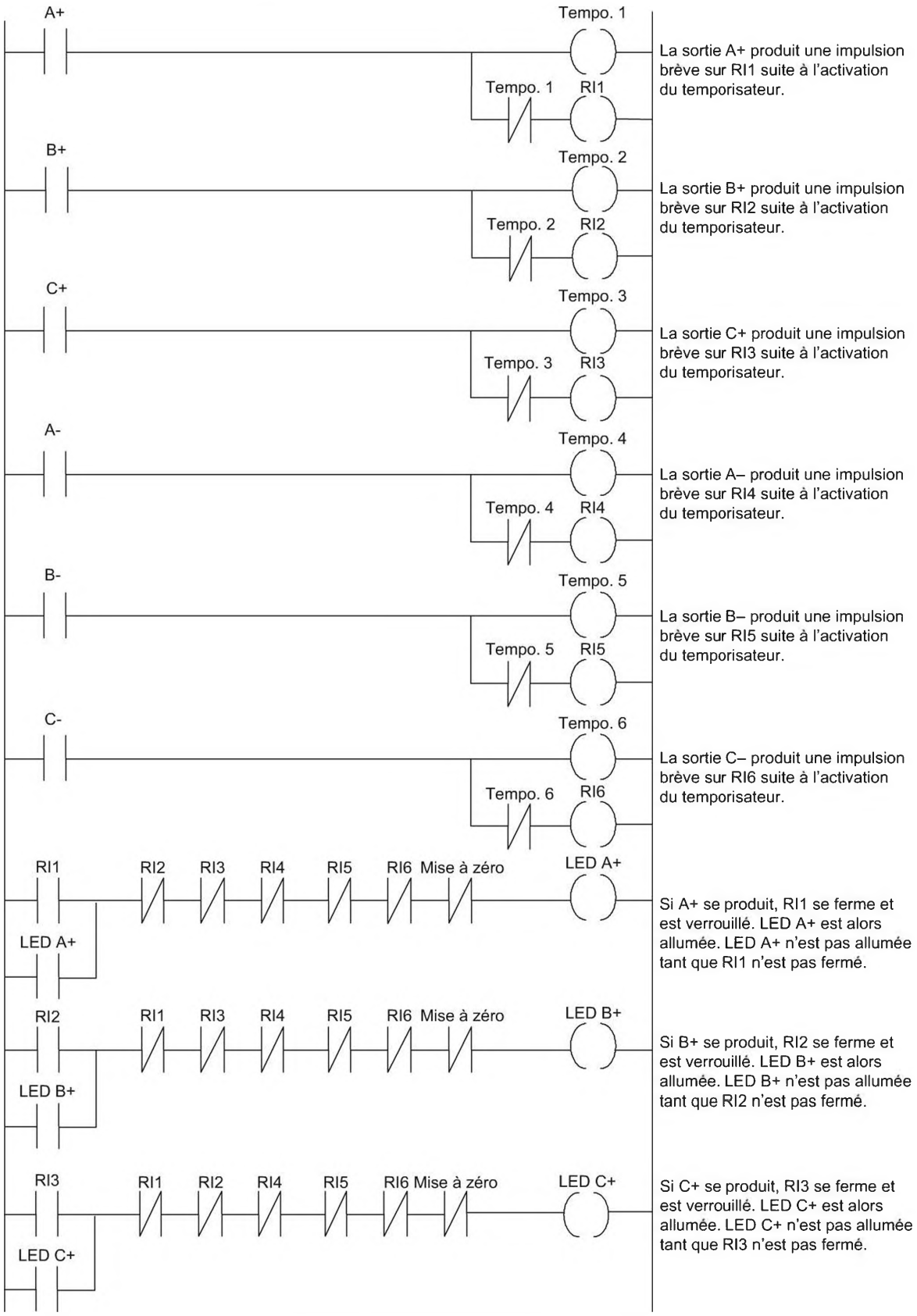


Figure 13.19 – Programme de diagnostic pour le dernier vérin actionné.

Copyright © 2015 Dunod.

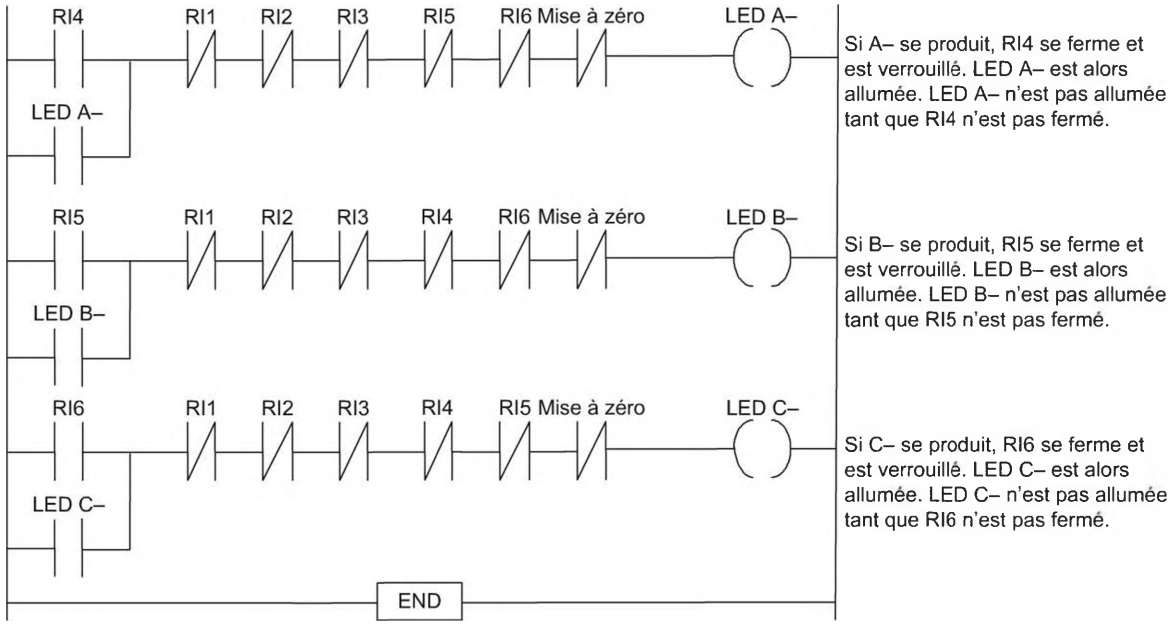


Figure 13.19 – Programme de diagnostic pour le dernier vérin actionné *(suite)*.

13.5 Documentation d'un système

La documentation constitue le principal guide employé quotidiennement par les utilisateurs, ainsi que pour le dépannage et la recherche des défauts. Elle doit donc être complète et sous une forme facile à suivre. La documentation d'installation d'un API doit comprendre les éléments suivants :

- une description de l'équipement ;
- une spécification des exigences de commande ;
- des détails sur l'automate programmable industriel ;
- des schémas de l'installation électrique ;
- la liste des connexions d'entrées et de sorties ;
- un programme applicatif, avec une description de ses opérations ;
- une sauvegarde du logiciel ;
- un manuel d'exploitation, avec les détails de toutes les procédures de mise en marche et d'arrêt, ainsi que les alarmes.

13.5.1 Exemple de programme industriel

L'exemple suivant montre un programme pour un équipement réel contrôlé par un API PLC5 d'Allen-Bradley ; je remercie Andrew Parr pour me l'avoir fourni. Il illustre la manière de documenter un fichier de programme dans un but de clarification et de description des procédures de sécurité et d'indication des défauts employées. Vous remarquerez que la barre d'alimentation droite a été omise, ce qu'autorise la norme CEI 61131-3.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

Le programme complet est constitué d'environ quarante fichiers. Chacun contrôle une partie du fonctionnement global et est séparé du suivant par un saut de page. Le fichier donné ci-après contrôle une scie à ruban pour la coupe de ballots. Il implique les commandes du moteur, des témoins lumineux et une courte séquence de transitions à état.

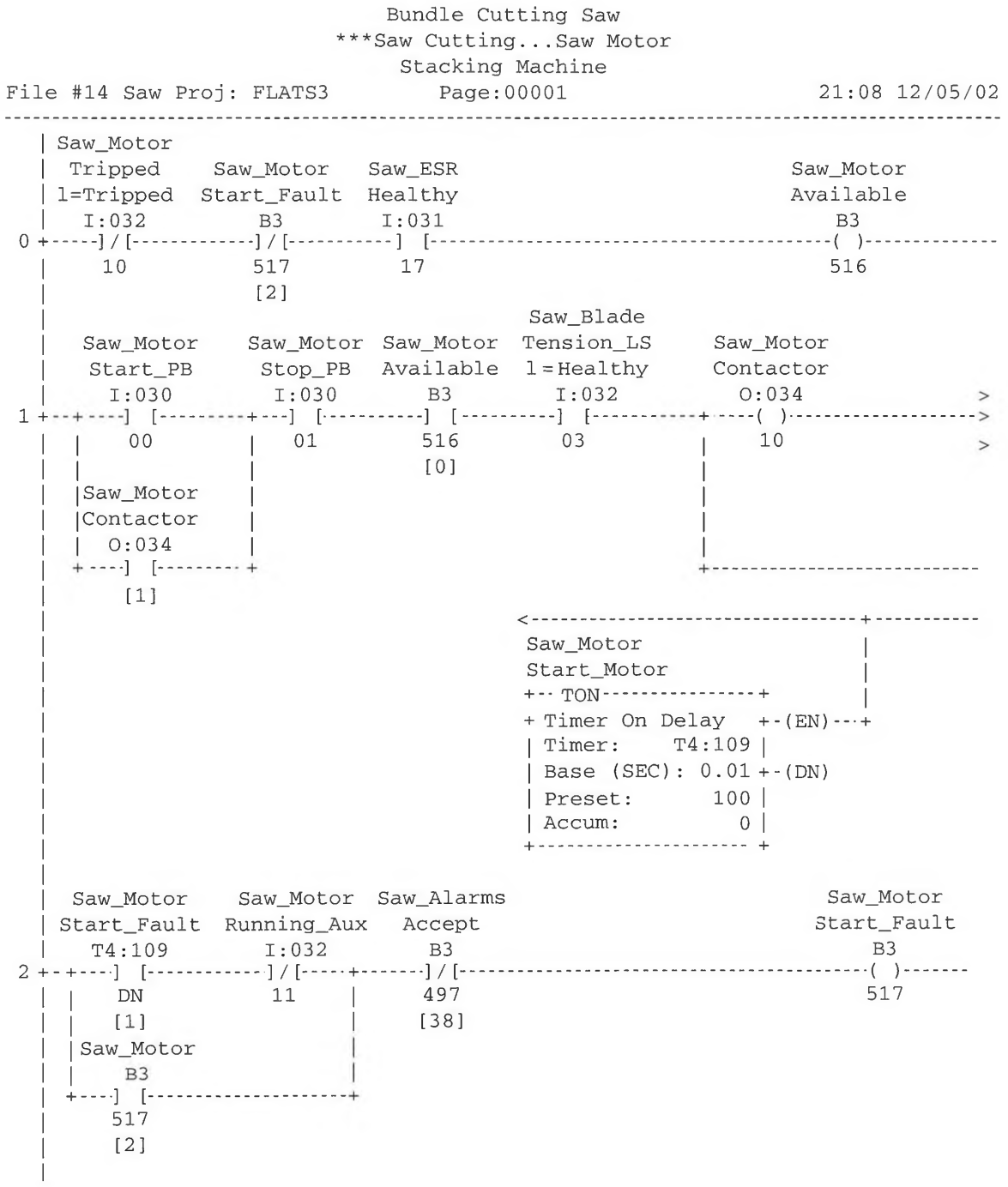
Vous remarquerez l'utilisation de références croisées entre les lignes, par exemple [38] sous B3/497 à la ligne 2. Elles sont utilisées pour indiquer le point d'origine de l'élément, par exemple à la ligne 38 pour B3/497 dans ce programme. Notez également que toutes les instructions sont accompagnées de description et que le fichier est décomposé en pages. Le logiciel permet d'aller directement à une fonction en consultant les intitulés des pages.

Toutes les lignes de démarrage du moteur fonctionnent de la même manière. L'API excite le contacteur et, une seconde plus tard, examine le retour du relais auxiliaire (libellé Aux dans le programme) pour savoir si le contacteur a été activé. Si une défaillance conduit à la désactivation du contacteur, par exemple une perte d'alimentation, un déclenchement ou une bobine de circuit ouvert, l'API signale le défaut et désactive la sortie du contacteur afin que la machine ne redémarre pas immédiatement lorsque le défaut est corrigé.

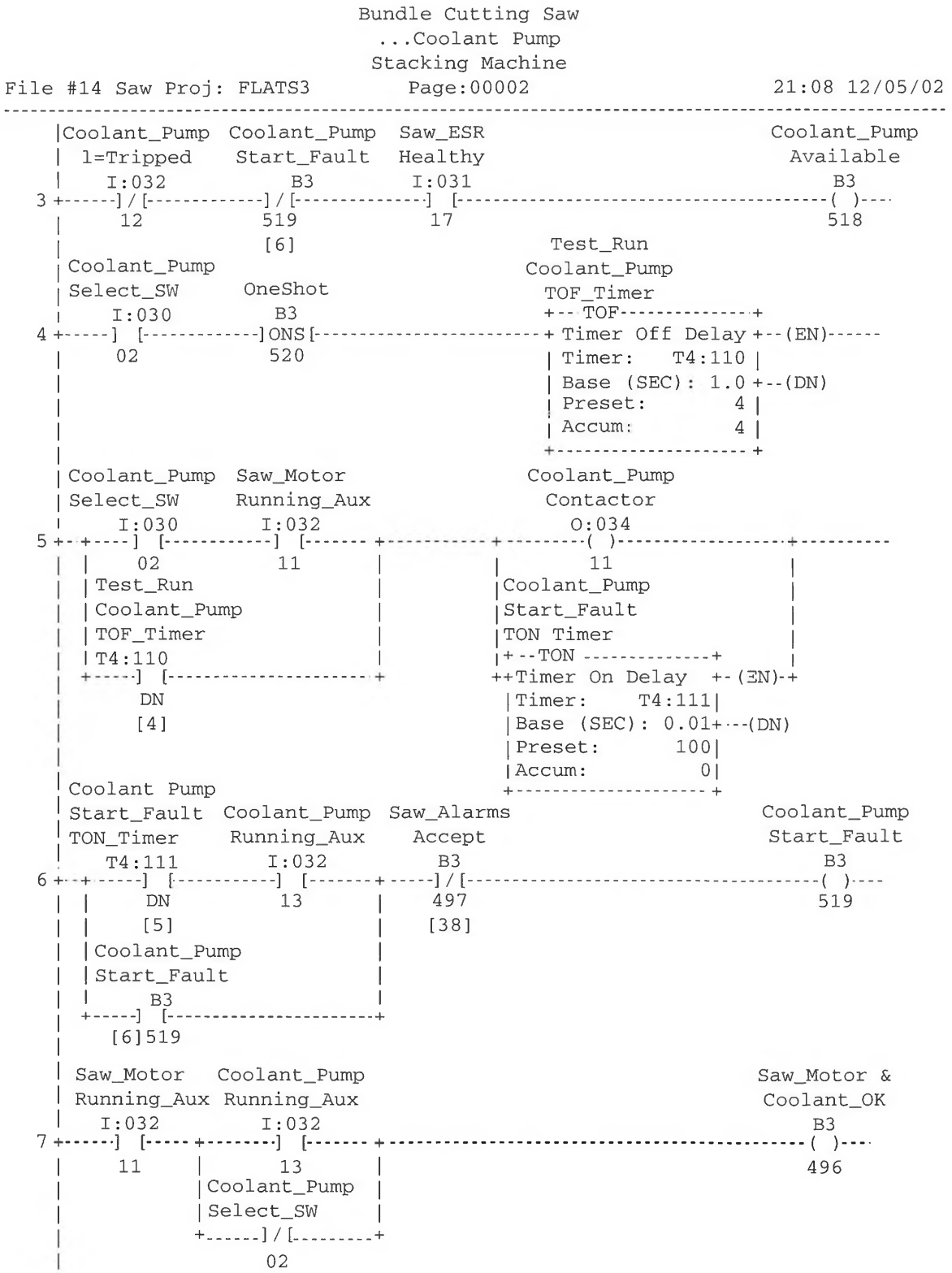
La scie se trouve normalement en position haute, loin du ballot. Pour commencer la coupe, le moteur doit être démarré et le bouton-poussoir du bas enfoncé (à la ligne 8). La scie descend par gravité, à une vitesse rapide ou lente, déterminée par des vannes hydrauliques. Pour relever la scie, une pompe hydraulique est mise en marche de manière à envoyer l'huile dans les vérins de la scie. La scie peut être relevée à tout moment, par exemple pour évacuer les copeaux, ce qui la place dans un état de pause. Sinon, la coupe se poursuit jusqu'à ce que la limite inférieure soit atteinte. La scie est alors relevée jusqu'à la limite supérieure, en attente du ballot suivant. La coupe peut être annulée en appuyant pendant deux secondes sur le bouton de relevage. Pendant la coupe, le ballot est maintenu par des solénoïdes de pinces.

Les trois dernières lignes du programme déterminent la longueur à couper. Deux cellules photoélectriques sont disposées à environ 20 mm de chaque côté d'un chariot mobile. Elles sont positionnées à la longueur souhaitée. L'opérateur déplace le ballot jusqu'à ce que la première soit cachée et la seconde dégagée. Elles contrôlent les témoins trop long/correct/trop court.

Copyright © 2015 Dunod.



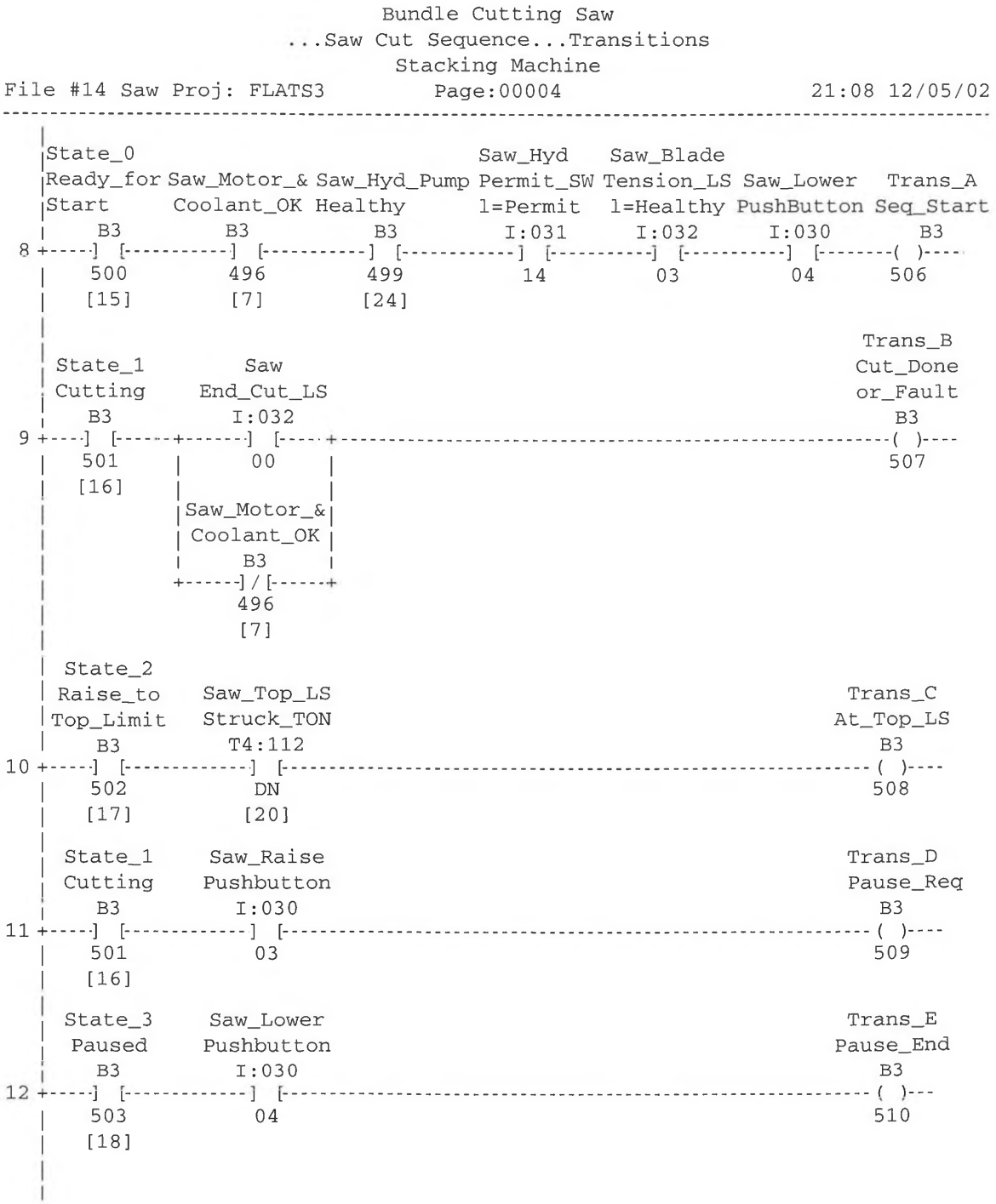
Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.



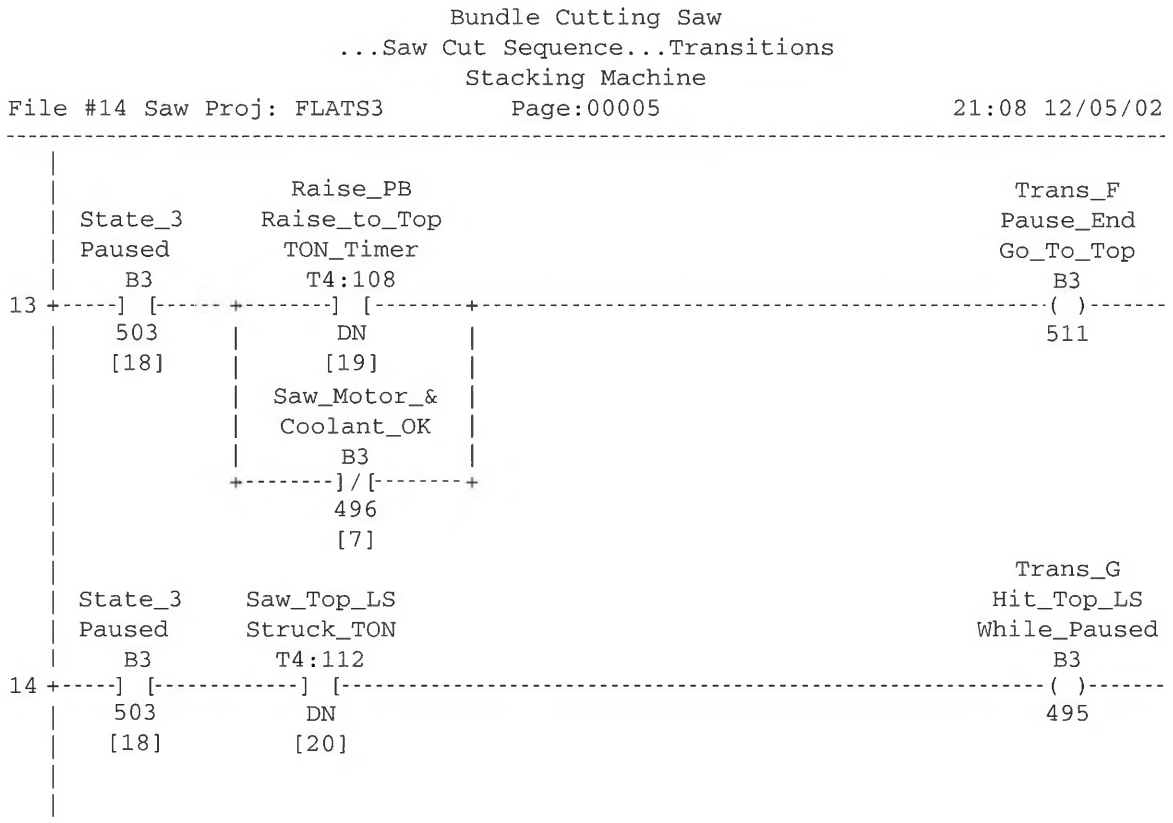
Copyright © 2015 Dunod.

```
                                Bundle Cutting Saw
                                 ...Coolant Pump
                                 Stacking Machine
File #14 Saw Proj: FLATS3          Page:00003                    21:08 12/05/02
--------------------------------------------------------------------------------
Blank page for future modification
```

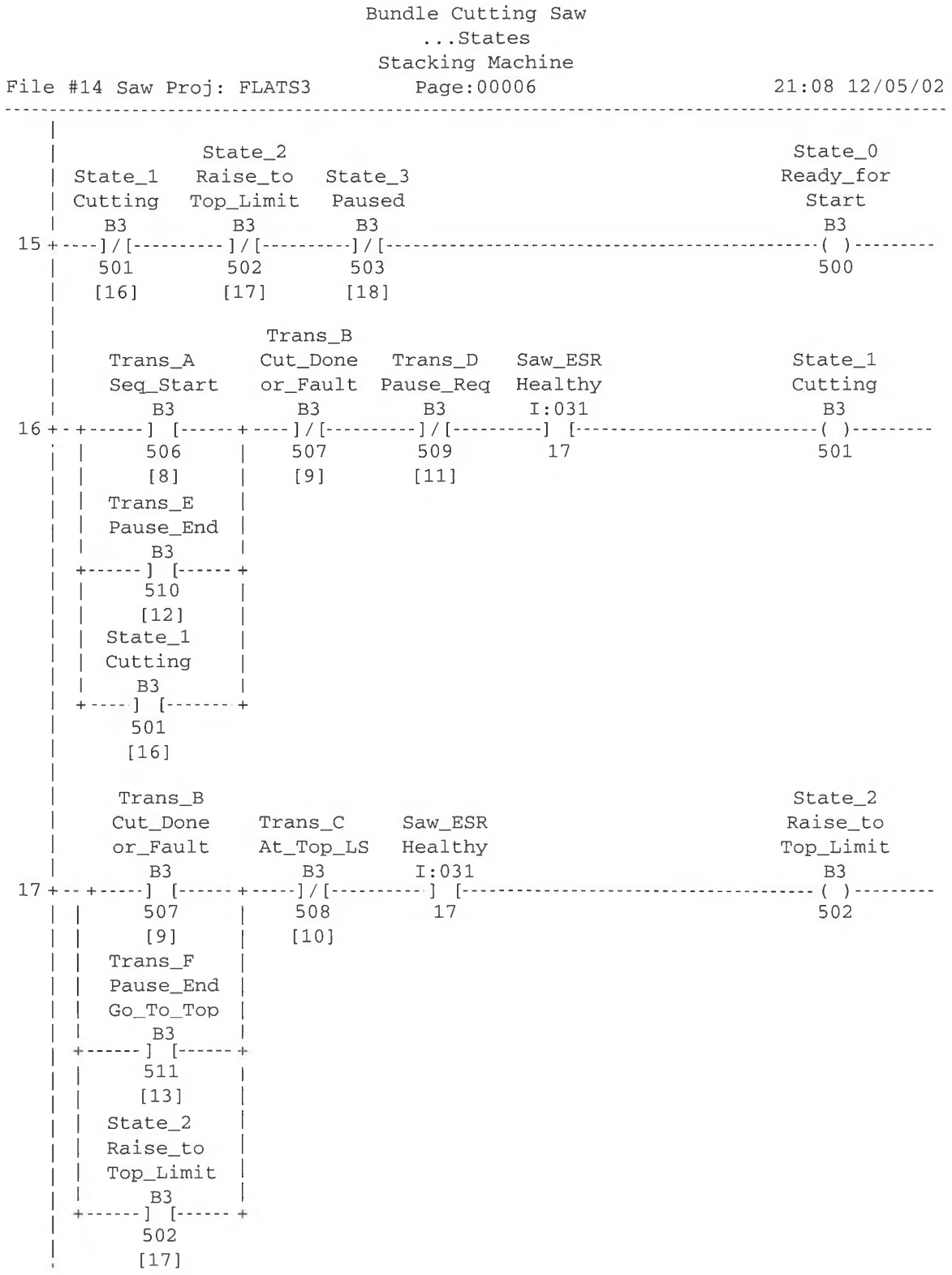
Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.



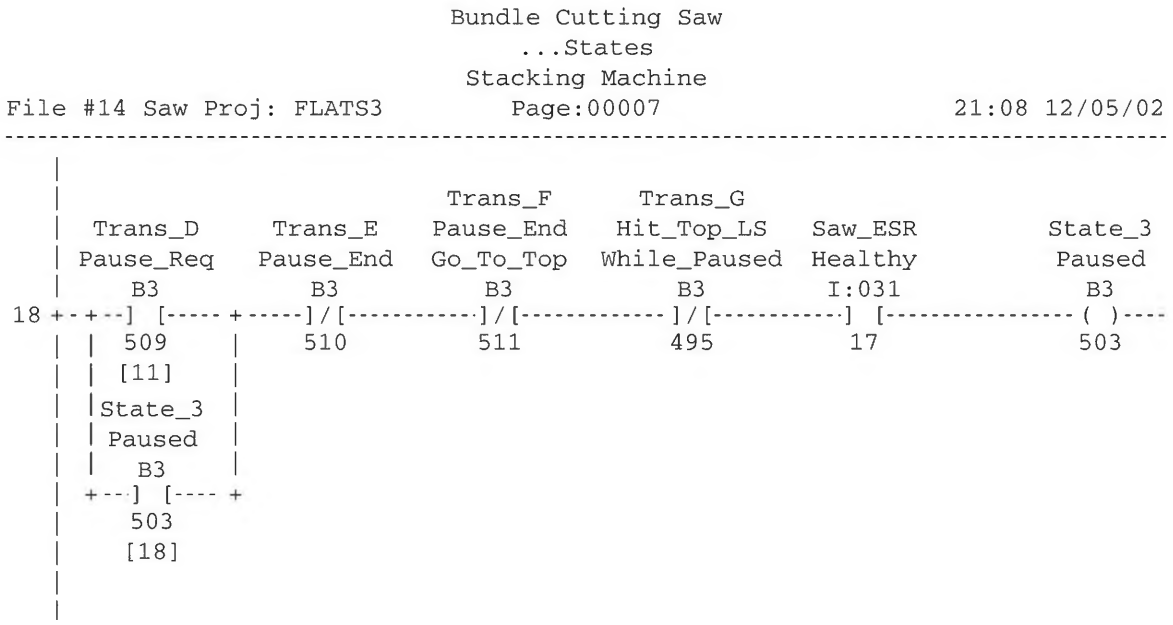
Copyright © 2015 Dunod.



Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.



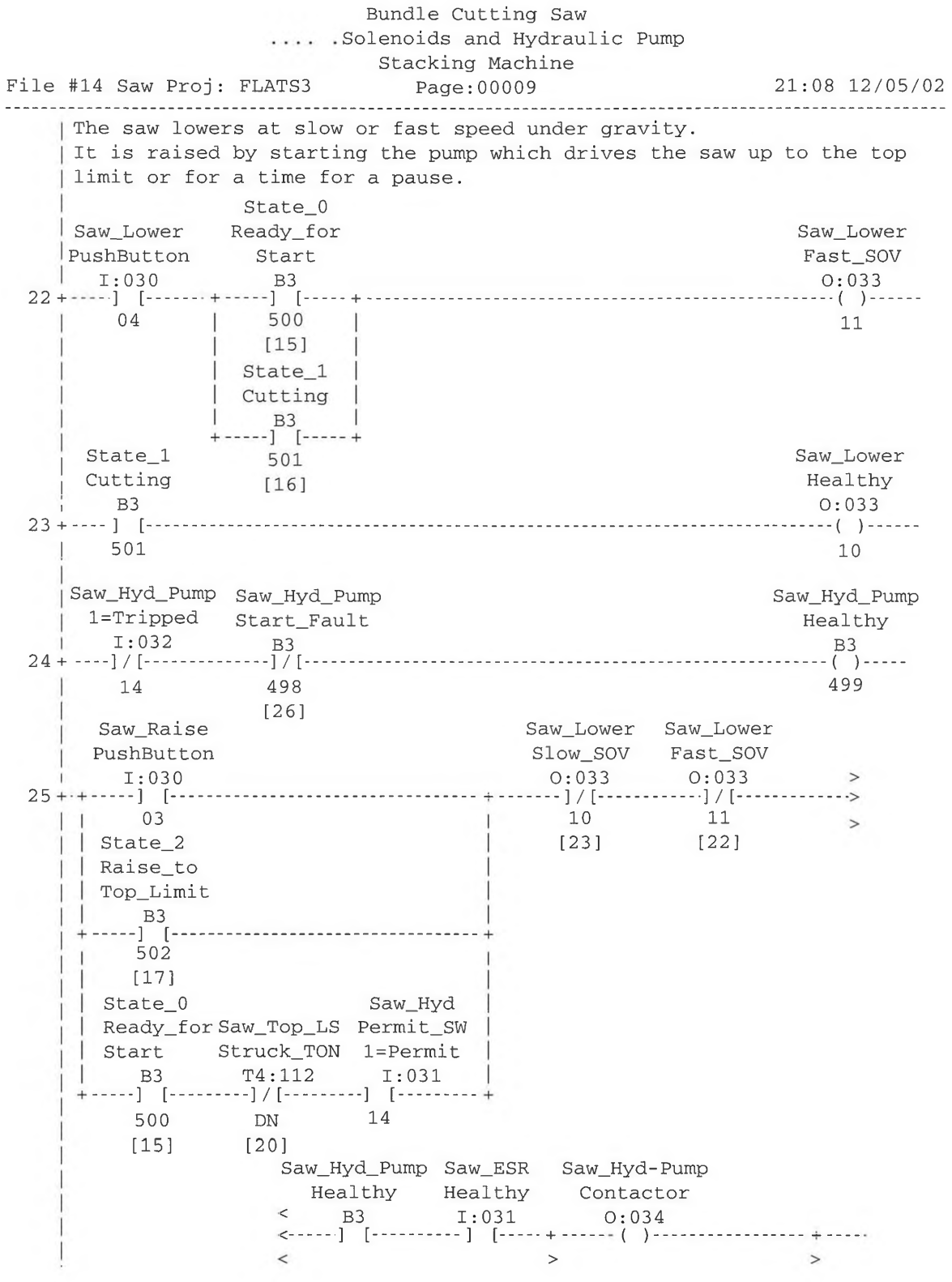
Copyright © 2015 Dunod.



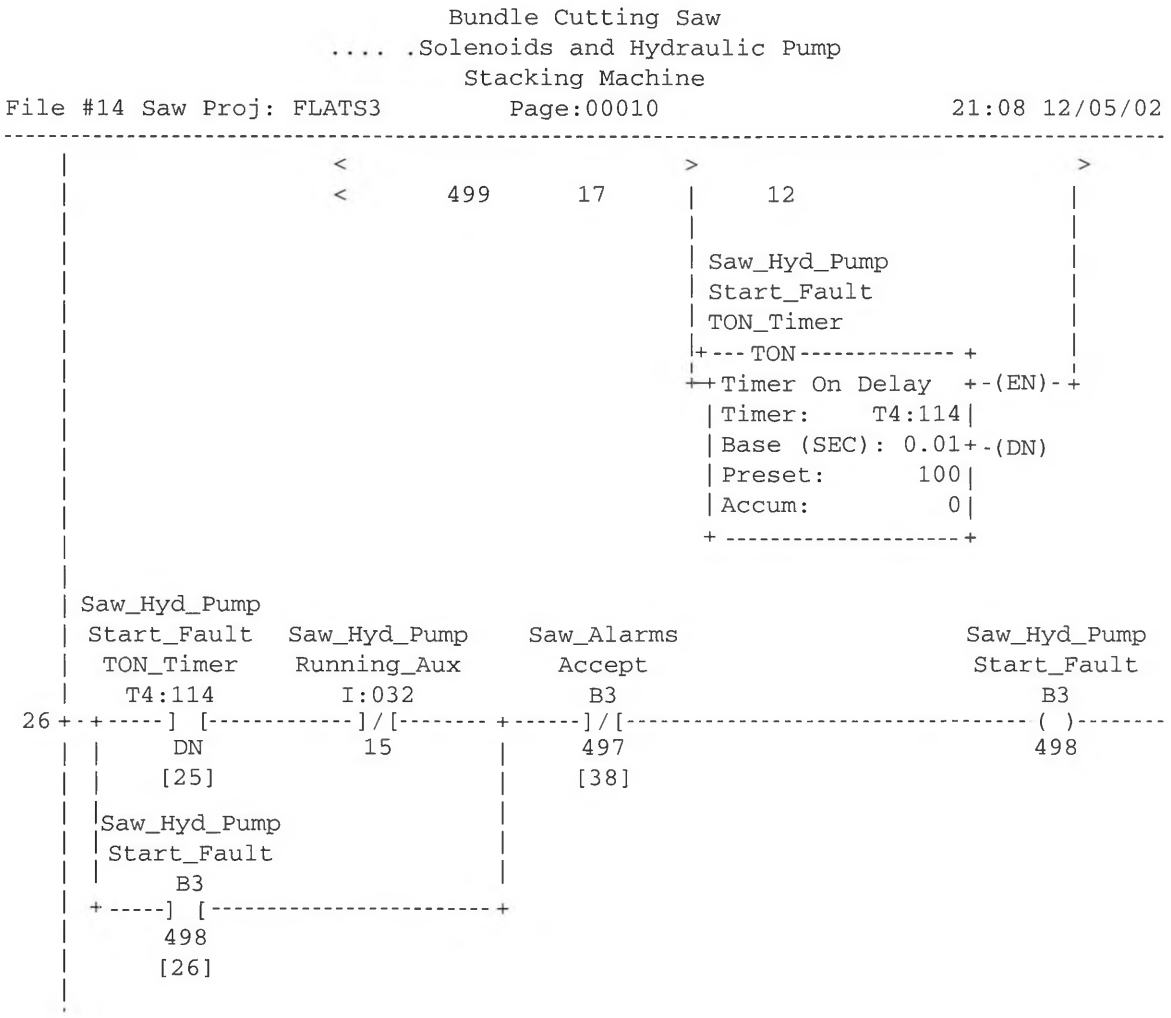
Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

```
                                Bundle Cutting Saw
                                   .... .Timers
                                 Stacking Machine
File #14 Saw Proj: FLATS3          Page:00008                          21:08 12/05/02
-------------------------------------------------------------------------------------------
    |
    |If Raise PB is pressed for more than 2 secs go right to top limit switch
    |
    |                                                    Raise_PB
    |    State_3    Saw_Raise                          Raise_to_Top
    |     Paused    PushButton                           TON_Timer
    |       B3         I:030                        +--TON-------------+
 19 +-------] [----------] [--------------------------+Timer On Delay  +--(EN)------
    |       503          03                         |Timer:     T4:108|
    |                                               |Base (SEC): 0.01+--(DN)
    |                                               |Preset:       200|
    |                                               |Accum:          0|
    |                                               +-----------------+
    |T4:112 ensures saw carriage goes past top limit to help avoid creeping
    |off the top position
    |
    |Saw_Top_LS                                      Saw_Top_LS
    | 1=Struck                                       Struck_TON
    |   I:032                                     +---TON----------------+
 20 +----] [-----------------------------------------+ Time On Delay    +--(EN)---+---
    |     01                                      | | Timer:     T4:112 |         |
    |                                             | | Base (SEC): 0.01 +---(DN)  |
    |                                             | | Preset:       100 |         |
    |                                             | | Accum:        101 |         |
    |                                             | +-----------------------+         |
    |                                             |Saw_Top_LS                         |
    |                                             |Struck_TOF                         |
    |                                             | 1=At_Top                          |
    |                                             |+-- TOF---------------+            |
    |                                             ++ Timer Off Delay  +---(EN)--+     |
    |                                              | Timer:     T4:113 |            |
    |                                              | Base (SEC): 0.01 +----(DN)      |
    |                                              | Preset:       300 |            |
    |                                              | Accum:          0 |            |
    |                                              +-----------------------+         |
    |
    |Permissive for bundle delivery/despatch
    |  Saw_Top_LS                                                          Saw_Not
    |   1=Struck                                                         Operating
    |     I:032                                                            B3
 21 +-+----] [--------+-------------------------------------------------------------( )----
    | |    01        |                                                              524
    | | Saw_Hyd      |
    | |Permit_SW     |
    | | 1=Permit     |
    | |   I:031      |
    | +----]/[-------+
    |      14
    '
```

Copyright © 2015 Dunod.



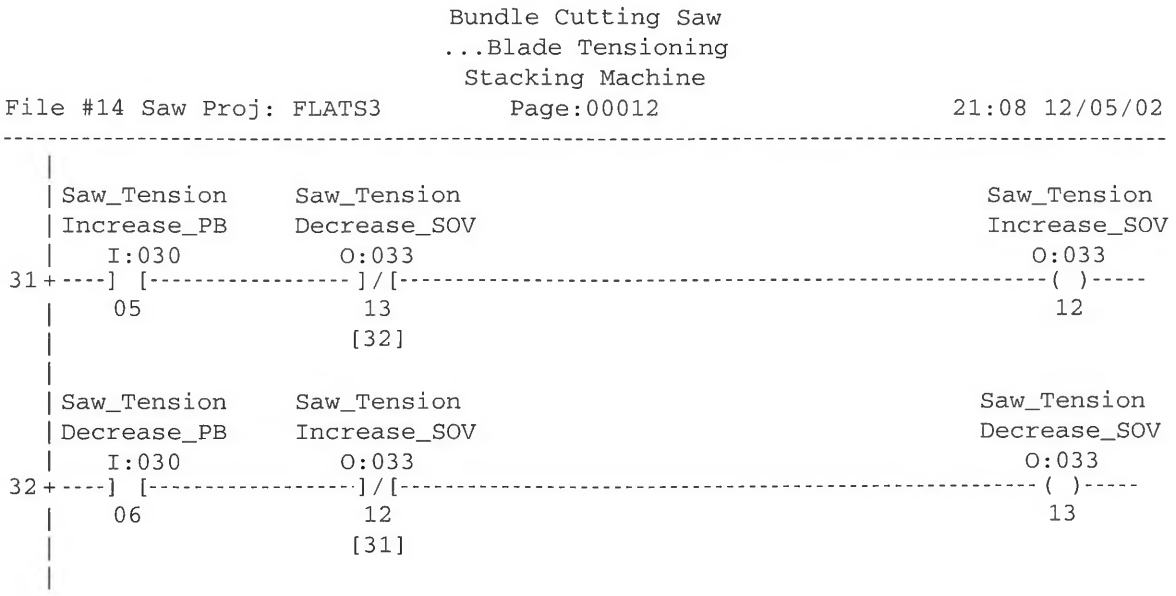
Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.



Copyright © 2015 Dunod.

```
                                  Bundle Cutting Saw
                                 ...Blade Tensioning
                                  Stacking Machine
File #14 Saw Proj: FLATS3           Page:00011                        21:08 12/05/02
-------------------------------------------------------------------------------------
    |Saw tension is changed via two hydraulic soleniods.
    |The TOF timer on the pump reduces start commands on the pump.
    |
    |Saw_Tension
    |Motor_Tripped  TensionPump   Saw_ESR                                Tension_Pump
    | 1=Tripped     Start_Fault   Healthy                                 Available
    |    I:032          B3         I:031                                     B3
 27 +-----]/[-----------]/[-------------] [-----------------------------------( )--------
    |     05           513           17                                      512
    |                 [30]
    |Saw_Tension     TensionPump                    TensionPump
    |Increase_PB      Available                     Run_Cmd_TOF
    |   I:030            B3                      +---TOF-----------+
 28 +-+--] [----------+-----] [--------------------+Timer Off Delay+--(EN)------
    | |  02          |    512                     |Timer:   T4:115|
    | |              |    [27]                    |Base (SEC): 1.0+--(DN)
    | | Saw_Tension  |                            |Preset:       5|
    | | Decrease_PB  |                            |Accum:        5|
    | |    I:030     |                            +-----------------+
    | +------] [-------+
    |       06
    | Tension_Pump                              Tension_Pump
    |  Run_Cmd_TOF                                Contactor
    |    T4:115                                     O:034
 29+ -------] [-------------------------------------+------( )-------------------------+-------
    |       DN                                      |     13                           |
    |                                               |Tension_Pump                      |
    |                                               | Start_Fault                      |
    |                                               |  TON Timer                       |
    |                                               |+--TON----------------+           |
    |                                               ++ Timer On Delay      +---(EN)---+
    |                                                | Timer:    T4:116    |
    |                                                | Base (SEC): 0.01    +--(DN)
    |                                                | Preset:       100   |
    |                                                | Accum:          0   |
    |                                                +---------------------+
    | Tension_Pump
    |  Start_Fault   Saw_Tension    Saw_Alarms                          Tension_Pump
    |   TON_Timer     Pump_Aux       Accept                             Start_Fault
    |    T4:116         I:032          B3                                  B3
 30+- +---] [-------------]/[-------+------]/[-------------------------------( )--------
    | |    DN             06       |     497                              513
    | |   [29]                     |     [38]
    | |Tension_Pump                |
    | |Start_Fault                 |
    | |     B3                     |
    | +-----] [--------------------- +
    |       513
    |       [6]
```

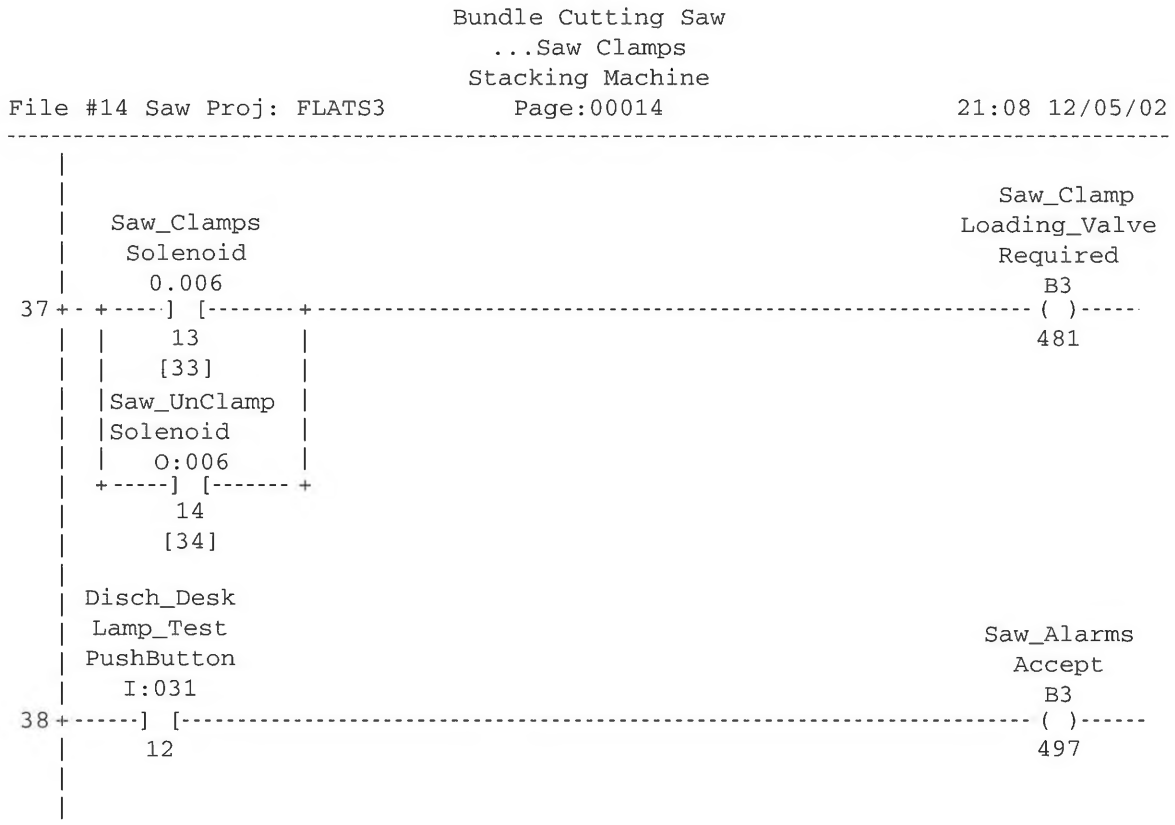
Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.



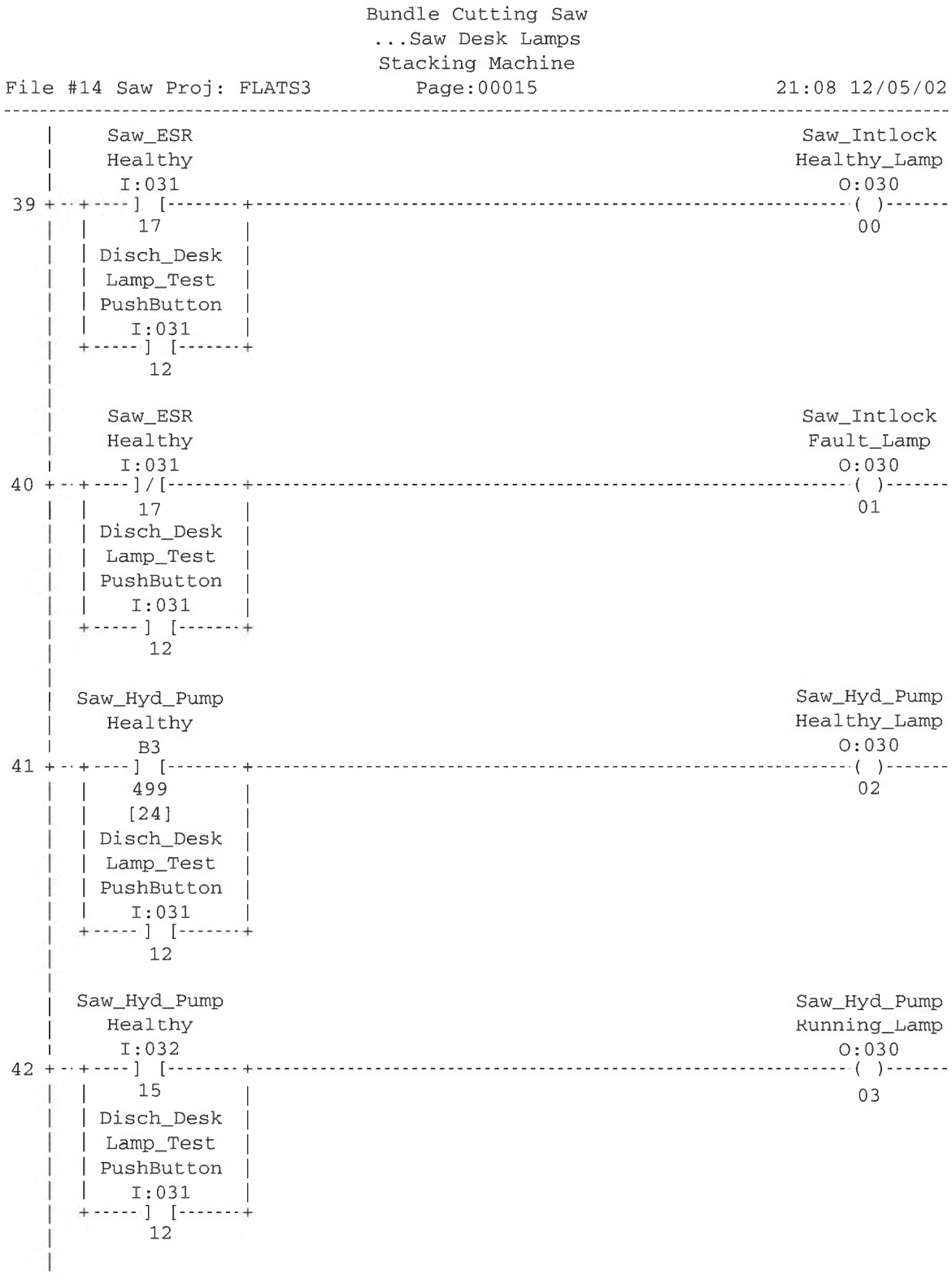
Copyright © 2015 Dunod.

```
                                  Bundle Cutting Saw
                                     ...Saw Clamps
                                   Stacking Machine
File #14 Saw Proj: FLATS3            Page:00013                          21:08 12/05/02
-------------------------------------------------------------------------------------------
     |T4:118 & 119 operate the clamp/unclamp solenoids for a fixed time.
     |
     |    Saw_Clamp  Saw_Unclamp  Saw_Unclamp       Saw_Clamp  Saw_Clamp
     |    PushButon   Solenoid    PushButton        TON_Timer  Solenoid
     |      I:034       O:006       I:034            T4:118     O:006
  33 +- +------] [------ +----]/[------------]/[---------+--------]/[---------( )--------+----
     |  |      00       |   14           01              |       DN         13          |
     |  |               |  [34]                          |      [33]                    |
     |  |  Saw_Clamp    |                                |                              |
     |  |   Solenoid    |                                |Saw_Clamp                     |
     |  |    O:006      |                                |TON_Timer                     |
     |  +-------] [-----+                                |+--TON------------+           |
     |         13                                        ++Timer On Delay +--(EN)-+
     |        [33]                                        |Timer:   T4:118|
     |                                                    |Base (SEC): 1.0+--(DN)
     |                                                    |Preset:       2|
     |                                                    |Accum:        0|
     |                                                    +-----------------+
     |
     |    Saw_UnClamp  Saw_Clamp   Saw_Clamp     Saw_UnClamp  Saw_UnClamp
     |    PushButton   Solenoid    PushButton    TON_Timer    Solenoid
     |      I:034       O:006       I:034          T4:119       O:006
  34 +- +-------] [------+----]/[------------]/[--------+--------]/[---------( )--------+---
     |  |       01      |   13           00              |       DN         14          |
     |  |               |  [33]                          |      [34]                    |
     |  | Saw_UnClamp   |                                |                              |
     |  |   Solenoid    |                                |Saw_UnClamp                   |
     |  |    O.006      |                                |TON_Timer                     |
     |  +------] [------+                                | +--TON-----------+           |
     |         14                                        ++Timer On Delay +---(EN)-+
     |        [34]                                        | Timer:   T4:119|
     |                                                    | Base (SEC): 1.0+--(DN)
     |                                                    | Preset:       2|
     |                                                    | Accum:        0|
     |                                                    +-----------------+
     |
     |  Saw_Clamp                                                  Saw_Clamps
     |  Solenoid                                                   Last_Clamped
     |    O.006                                                        B3
  35 +------] [-------------------------------------------------------------(L)----------
     |      13                                                        488
     |     [33]
     |
     | Saw_UnClamp                                                 Saw_Clamps
     | Solenoid                                                    Last_Clamped
     |    O.006                                                        B3
  36 +------] [-------------------------------------------------------------(U)---------
     |      14                                                        488
     |     [34]
     |
```

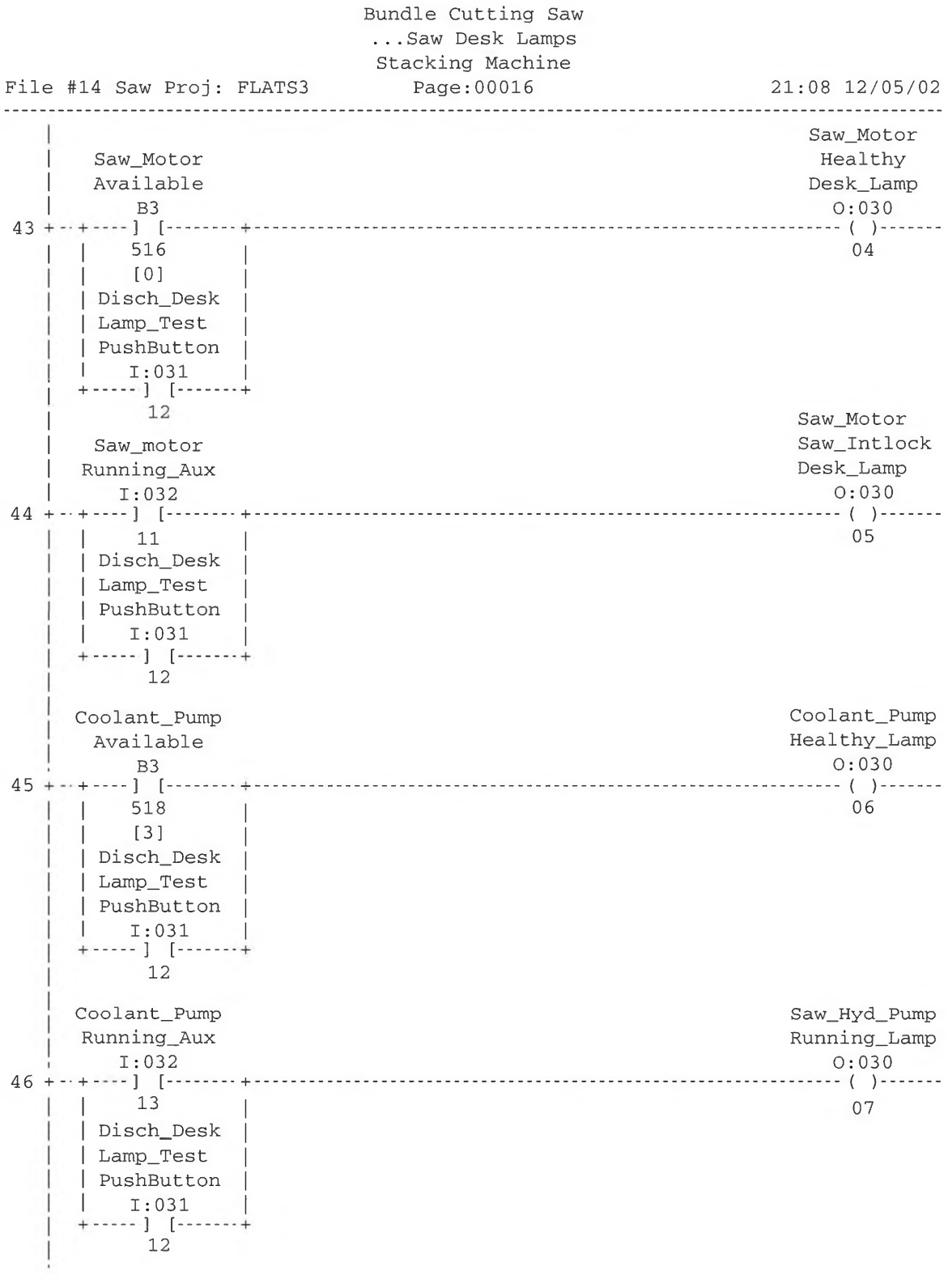
Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.



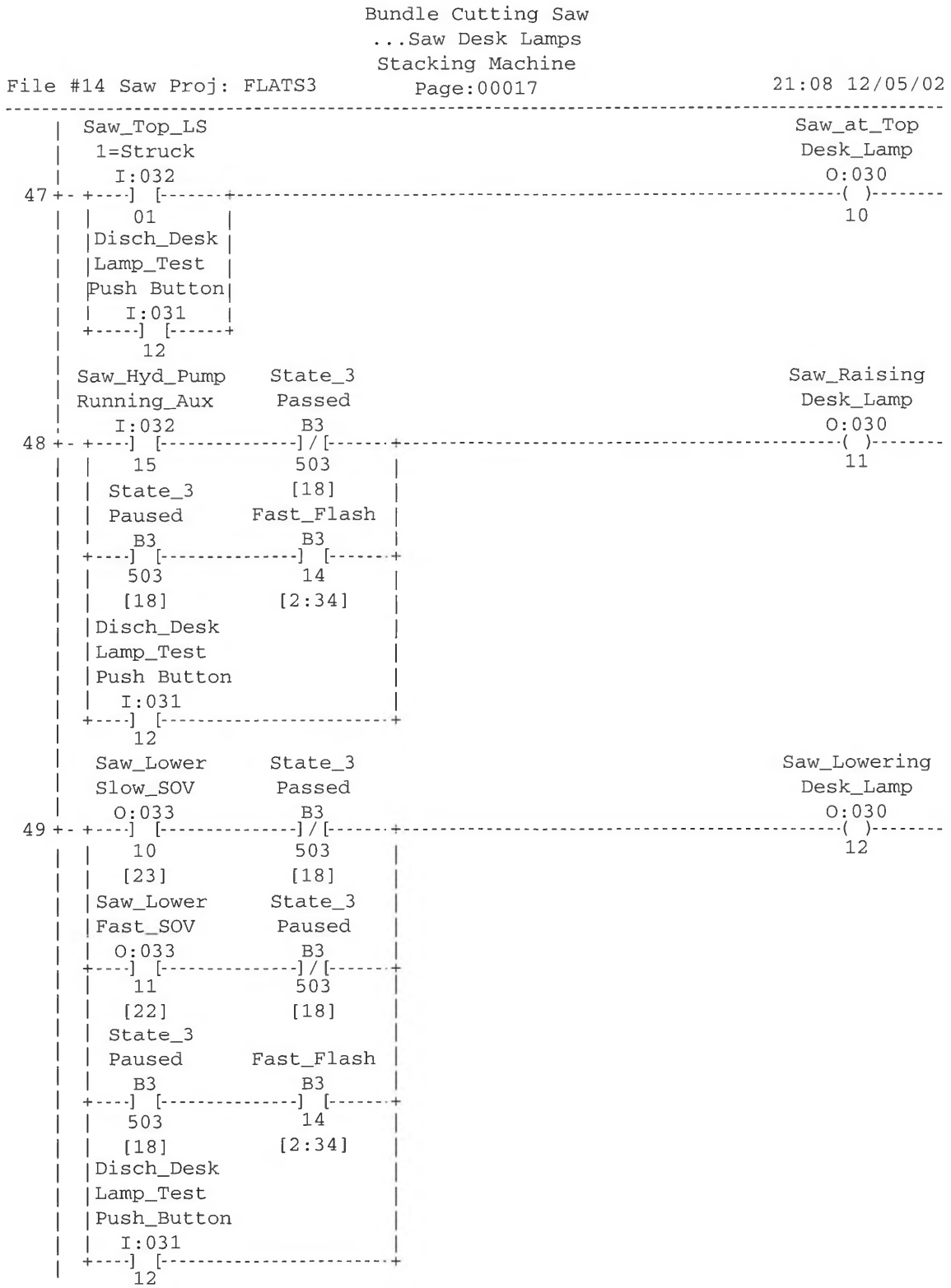
Copyright © 2015 Dunod.



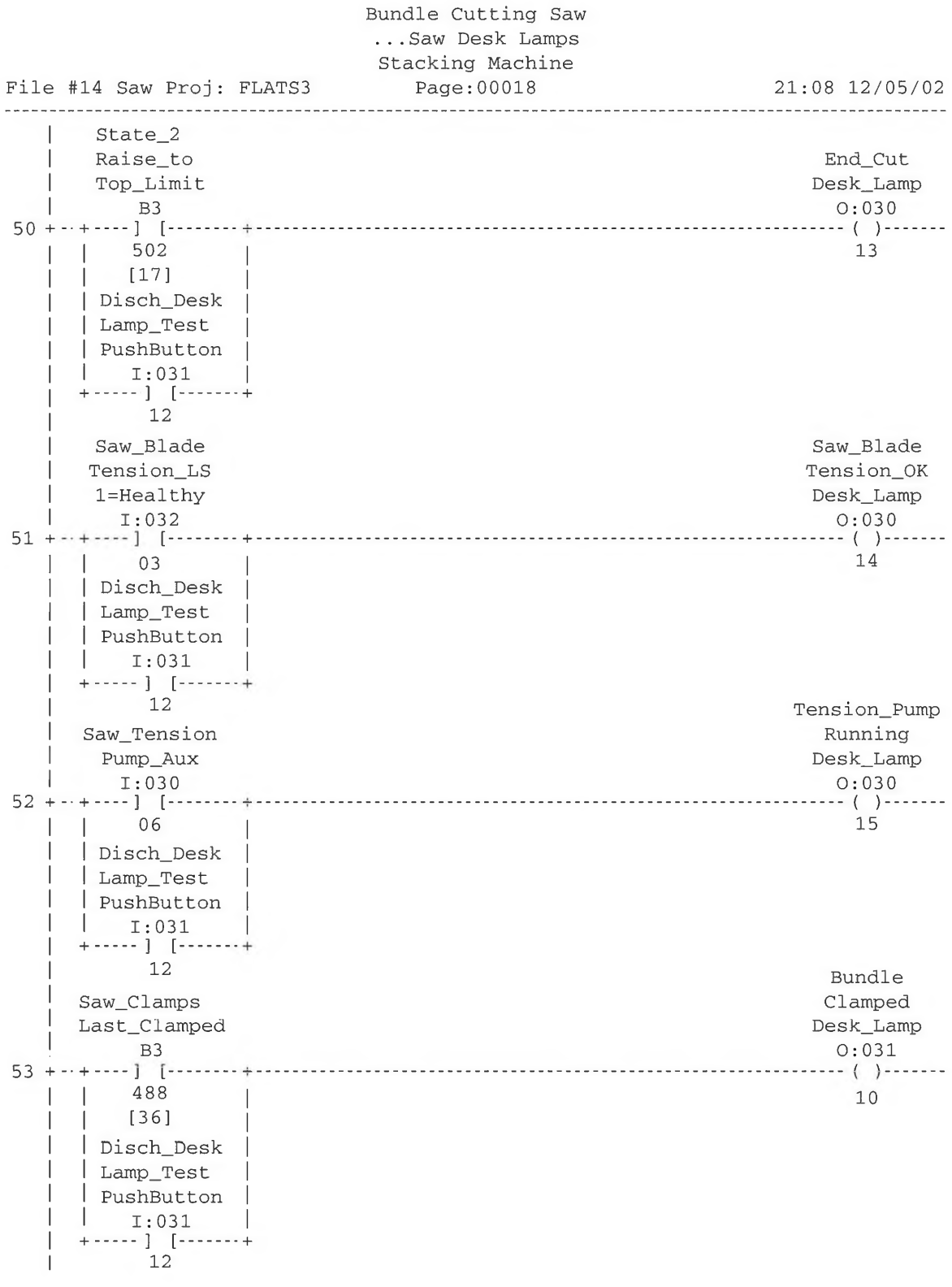
Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.



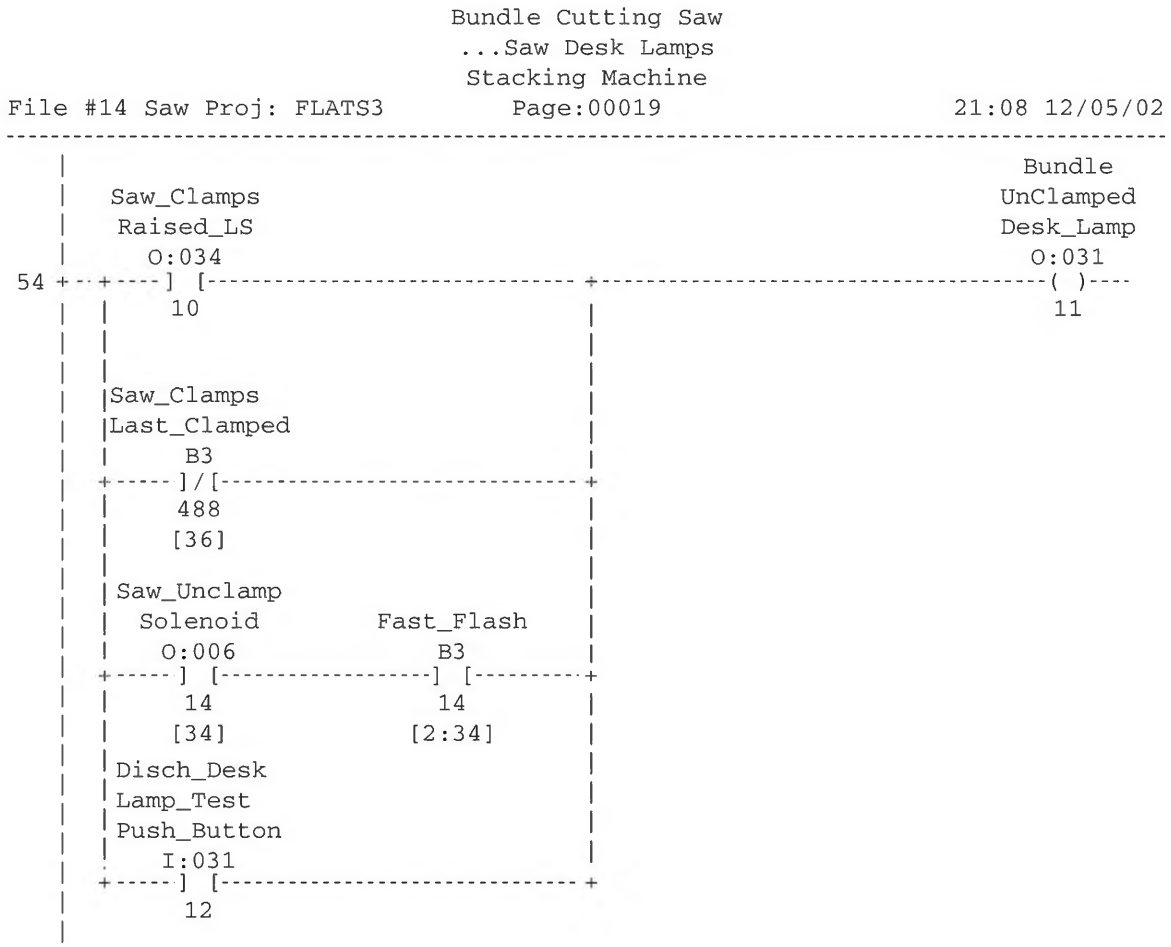
Copyright © 2015 Dunod.



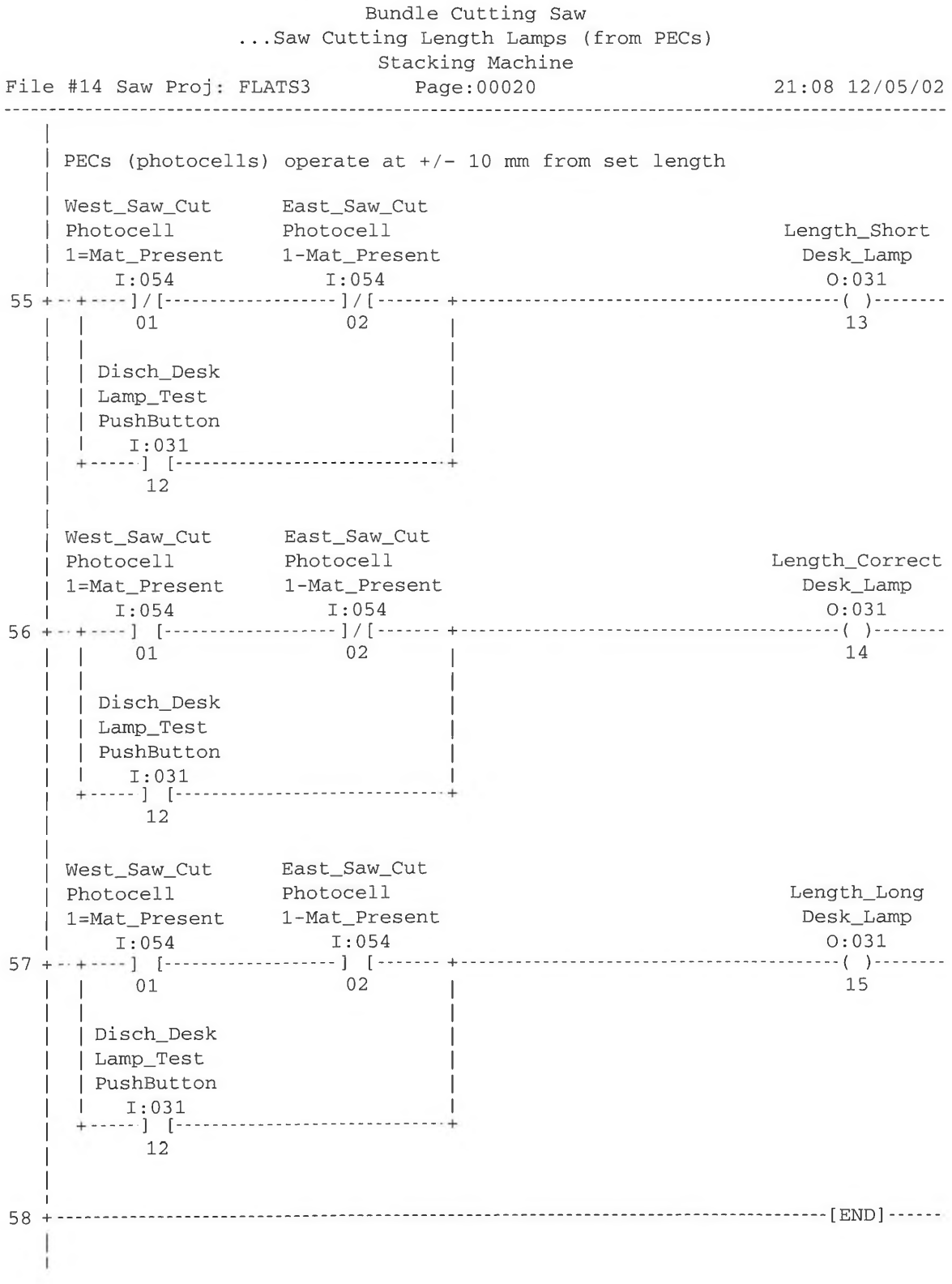
Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.



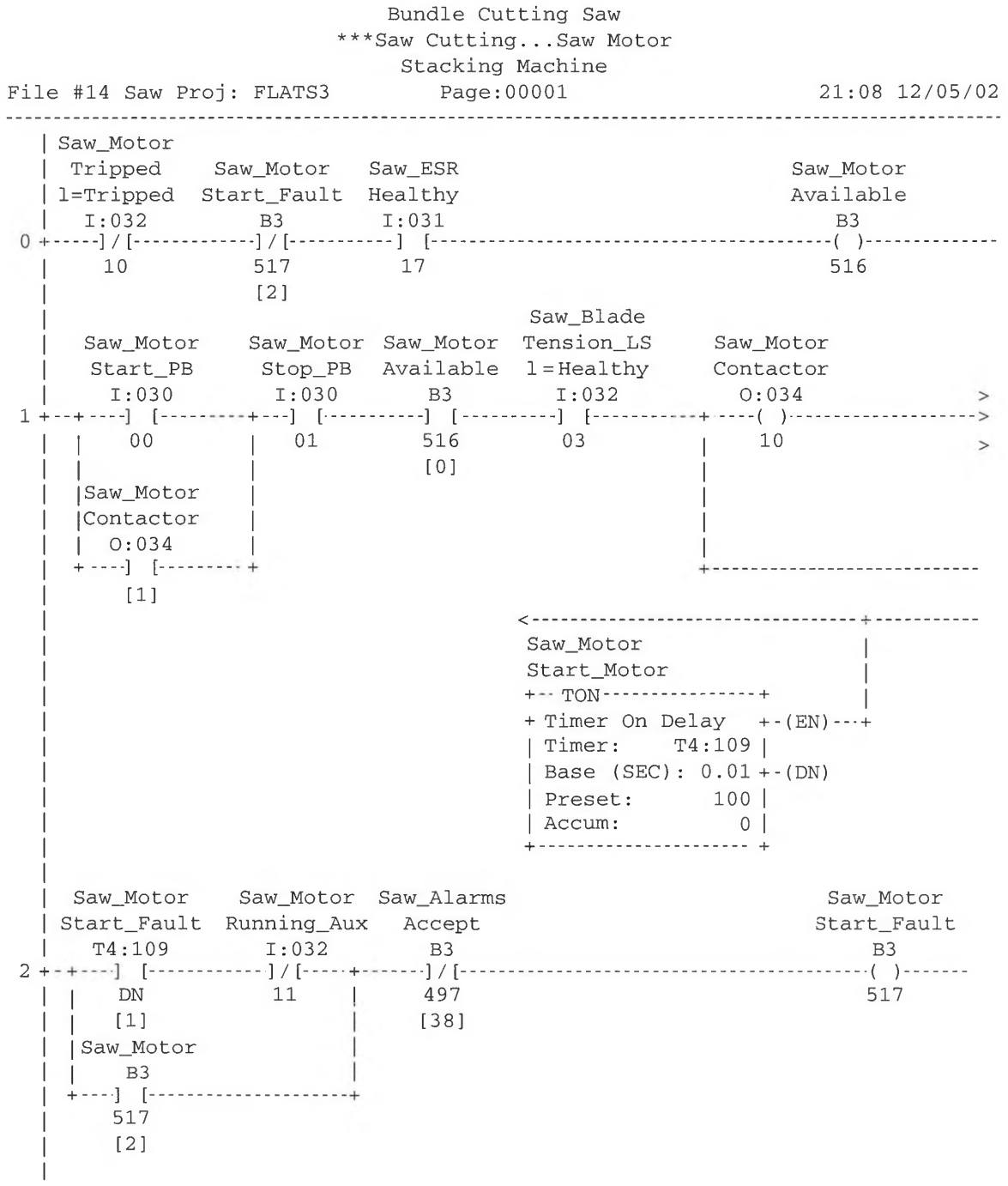
Copyright © 2015 Dunod.



Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.



Copyright © 2015 Dunod.



13.6 En résumé

Une approche systématique de l'écriture des programmes permet d'augmenter les chances d'arriver à des solutions de haute qualité en un temps aussi réduit que possible. Cette approche se fonde sur la définition des besoins, la définition de l'algorithme à utiliser, la conversion de l'algorithme en instructions pour l'API, le test et le débogage du programme, et la documentation du programme pour que les personnes qui l'utilisent comprennent son fonctionnement.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

La législation moderne en termes de sécurité demande aux employeurs de créer un lieu de travail sécurisé et exempt de risques sanitaires, de faire en sorte que les équipements soient sûrs et que des procédures de travail sécurisées soient établies et suivies. La norme CEI 61508 traite de la sécurité fonctionnelle des systèmes électriques/électroniques et électroniques programmables (E/E/PE). Les boutons d'arrêt d'urgence et les interrupteurs des dispositifs de sécurité doivent être câblés et ne pas dépendre du logiciel de l'API. Ainsi, en cas de dysfonctionnement du bouton d'arrêt ou de l'API, le système est automatiquement sécurisé. Le système doit présenter une sécurité intrinsèque. Cela peut impliquer une commande à deux mains, une surveillance du capot de protection, des barrières immatérielles, des planchers sensibles de sécurité et des relais d'arrêt d'urgence.

La mise en service d'un système à base d'API implique de vérifier que toutes les connexions par câble entre l'API et l'équipement contrôlé soient complètes, sûres, conformes à la spécification et aux normes locales, que l'alimentation corresponde au paramètre de tension configuré sur l'API, que tous les dispositifs de protection soient configurés avec les réglages de déclenchement appropriés, que les boutons d'arrêt d'urgence fonctionnent, que tous les dispositifs d'entrées-sorties soient connectés aux points d'entrées-sorties corrects et produisent les signaux appropriés, et que le logiciel ait été chargé et testé. Les techniques classiques de détection des défauts sont notamment la vérification du minutage par chien de garde, l'utilisation d'indicateurs montrant la dernière sortie activée, la réplication et la vérification des valeurs attendues.

La documentation d'une installation à base d'API doit comprendre une description de l'équipement, une spécification des exigences de commande, des détails sur l'automate programmable industriel, des schémas de l'installation électrique, la liste des connexions d'entrées et de sorties, un programme applicatif et un manuel d'exploitation, avec les détails de toutes les procédures de mise en marche et d'arrêt, ainsi que les alarmes.

13.7 Problèmes

Pour les questions 1 à 7, quatre réponses sont proposées : A, B, C et D. Choisissez la réponse correcte.

1. Les éléments du schéma à contacts de la Figure 13.20 peuvent être décrits par un algorithme de base du type :
   A. DO-THEN-DO-ENDDO
   B. IF-THEN-ELSE-ENDIF
   C. WHILE-DO-ENDWHILE
   D. Aucune des propositions A, B ou C
2. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Le terme *forçage*, lorsqu'il est appliqué à une entrée ou à une sortie d'un API, signifie utiliser un programme pour :

Copyright © 2015 Dunod.

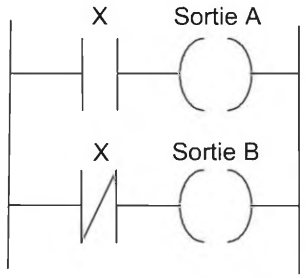


Figure 13.20 – Diagramme pour le problème 1.

(i) activer ou désactiver des entrées-sorties ;
(ii) vérifier que toutes les entrées-sorties produisent la réponse correcte lorsqu'elles sont sélectionnées.

Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?

A. (i) V (ii) V
B. (i) V (ii) F
C. (i) F (ii) V
D. (i) F (ii) F

3. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Dans le contexte des API, le terme *chien de garde* représente un mécanisme de vérification :

(i) d'absence de courants excessifs ;
(ii) d'exécution des fonctions dans des limites de temps définies.

Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?

A. (i) V (ii) V
B. (i) V (ii) F
C. (i) F (ii) V
D. (i) F (ii) F

4. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Lorsqu'un API se trouve en mode de monitoring, de test ou de débogage, il permet de :

(i) simuler le fonctionnement d'un programme ;
(ii) mener à bien un contrôle des défauts.

Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?

A. (i) V (ii) V
B. (i) V (ii) F
C. (i) F (ii) V
D. (i) F (ii) F

5. Lorsqu'un API se trouve en mode de monitoring, de test ou de débogage et lorsque le symbole illustré à la Figure 13.21 apparaît, cela signifie qu'une entrée :

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

A. Est défectueuse
B. Fonctionne correctement
C. Est active
D. Est inactive

Figure 13.21 – Diagramme pour le problème 5.

6. Indiquez si chacune des affirmations suivantes est vraie (V) ou fausse (F). Le dysfonctionnement d'un capteur d'entrée ou de son câblage, non d'une LED ou du canal d'entrée de l'API, est révélé par :
   (i) la LED d'entrée qui ne s'allume pas ;
   (ii) le forçage de cette entrée afin que la LED d'entrée s'allume.

   Parmi les propositions suivantes, laquelle décrit correctement ces deux affirmations ?
   A. (i) V (ii) V
   B. (i) V (ii) F
   C. (i) F (ii) V
   D. (i) F (ii) F
7. Un seul dispositif de sortie ne s'active pas alors que la LED de sortie est allumée. Le test de la tension en sortie révèle qu'elle est normale, mais le dispositif n'est pas alimenté. Le défaut vient :
   A. D'un problème de câblage
   B. D'un dispositif de sortie défectueux
   C. D'un problème dans l'API
   D. D'un problème dans le programme
8. Expliquez comment, en utilisant le forçage, la panne d'un capteur d'entrée ou de son câblage peut être détectée.
9. Donnez les causes possibles de l'arrêt complet d'une opération de commande et de l'API, avec un voyant d'alimentation éteint.
10. Donnez les causes possibles d'une LED de sortie allumée, mais avec le dispositif de sortie qui ne démarre pas.
11. Concevez un programme avec chien de garde pour arrêter une machine en cas d'existence d'un défaut dans l'un de ses systèmes de commande.
12. Concevez le programme d'un système pneumatique contrôlé par un API qui permet d'obtenir la séquence d'activation des vérins A+, B+, B–, A– et, en cas de défaut comme le blocage d'un vérin, qui affiche par une LED le point d'origine du défaut dans le cycle. Expliquez les actions de tous les éléments du système.

Copyright © 2015 Dunod.

13.8 Recherches complémentaires

1. Recherchez la norme de sécurité CEI 61508. Vous rechercherez également sur le Web des récapitulatifs de ses principales implications.
2. Recherchez les détails des systèmes de barrières immatérielles disponibles dans le commerce.
3. Recherchez les caractéristiques des relais de sécurité électronique disponibles dans le commerce.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

Copyright © 2015 Dunod.

14 • PROGRAMMES

Ce chapitre poursuit les exemples décrits dans les précédents et présente des programmes qui réalisent des tâches précises. Cela inclut la régulation de température et la commande de vannes pneumatiques.

14.1 Régulation de température

Prenons l'exemple d'un API, utilisé en tant que contrôleur tout ou rien d'un appareil de chauffage, pour la régulation de la température dans une pièce. L'appareil de chauffage doit être mis en marche lorsque la température tombe sous la valeur souhaitée et être arrêté lorsqu'elle l'atteint ou la dépasse. L'algorithme de base est le suivant :

```
IF la température est sous la valeur fixée
THEN
    DO allumer l'appareil de chauffage
ELSE
    DO éteindre l'appareil de chauffage
ENDIF
```

Pour produire l'entrée de l'API, nous avons besoin d'un dispositif qui ne génère aucune sortie lorsque la température se trouve sous la valeur souhaitée et qui génère une sortie lorsqu'elle est au-dessus, comme, par exemple, un thermostat bilame (voir Figure 14.1 et Section 2.1.5). Un bilame est constitué de deux lames de métal différent, de la même longueur et soudées ensemble. Les deux métaux présentent des coefficients de dilatation différents. Par conséquent, lorsque la température du bilame augmente, les lames se courbent de manière que l'un des métaux puisse se dilater plus que l'autre. Le métal ayant le coefficient de dilatation le plus élevé se trouve à l'extérieur de la courbe. Ce mouvement peut servir à établir ou à rompre un contact électrique et, par conséquent, peut fonctionner comme un interrupteur dépendant de la température. Si la température réelle se trouve en dessous de la valeur souhaitée, le bilame se courbe dans la position active et l'alimentation est appliquée ; la sortie est donc présente. La sortie du thermostat est de type tout ou rien et l'entrée de l'API est de type avec ou sans tension. La Figure 14.2 illustre la connexion de l'API et le schéma à contacts que nous pouvons utiliser.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

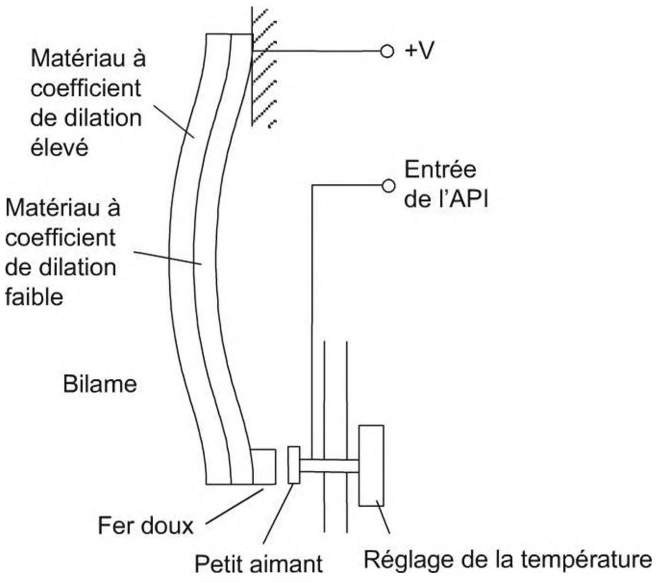


Figure 14.1 – Thermostat bilame connecté à l'API. Une tension est appliquée à l'entrée de l'API lorsque la température se trouve sous la valeur fixée. Aucune tension n'est appliquée lorsqu'elle se trouve au-dessus de cette valeur.

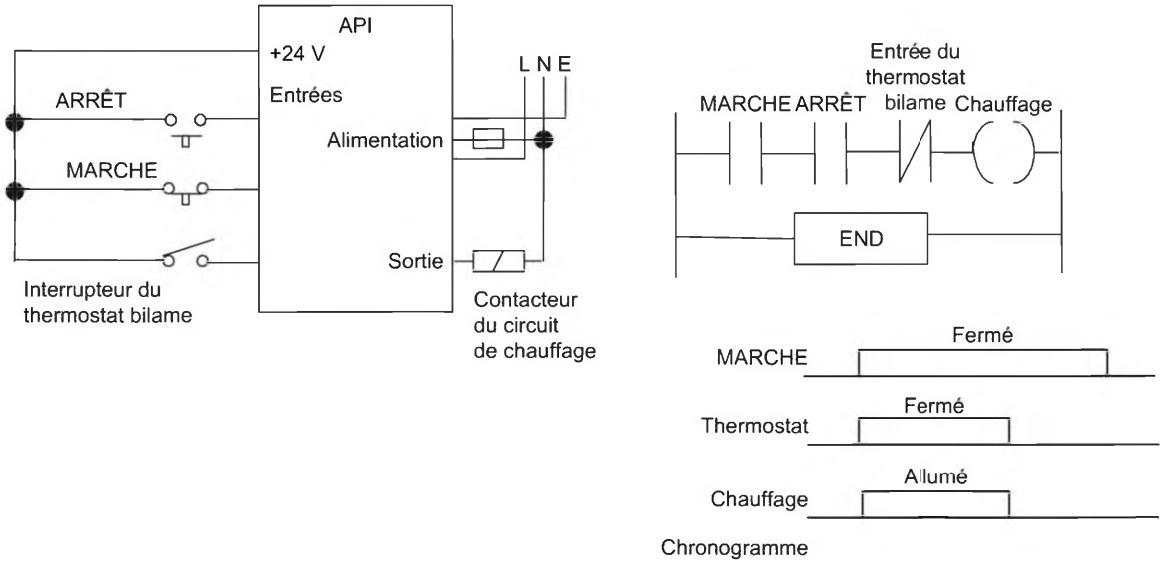


Figure 14.2 – Branchement de l'API et schéma à contacts impliquant le thermostat bilame.

À la place du thermostat bilame, nous pouvons également utiliser un thermocouple, une thermistance ou un circuit intégré (voir Section 2.1.5). Lorsqu'il est inséré dans le montage approprié, le capteur produit la tension adéquate en fonction de la température. Par exemple, la résistance d'une thermistance change en fonction de la température et peut être convertie en une tension à l'aide d'un circuit diviseur de tension (voir Figure 14.3). Supposons que nous utilisions une thermistance perle de verre de 4,7 kΩ. Elle possède une résistance de 4,7 kΩ à 25 °C, de 15,28 kΩ à 0 °C et de 0,33 kΩ à 100 °C. La résistance variable, qui

Copyright © 2015 Dunod.

permet de modifier la sensibilité du montage, peut avoir une plage allant de 0 à 10 kΩ. Cependant, puisqu'elle peut être réglée avec une résistance nulle, une résistance de protection doit être ajoutée afin d'éviter qu'un courant important ne traverse la thermistance, dont la puissance maximale acceptée est de 250 mW. Avec une alimentation de 6 V, une résistance variable nulle, une résistance de protection R et une température de 100 °C, le courant I qui traverse la thermistance est donné par $U = RI$, c'est-à-dire $6 = I(0 + R + 330)$, ou $I = 6/(R + 330)$. La puissance dissipée par la thermistance est $I^2 \times 330$. Si nous voulons que cette puissance soit bien inférieure à la puissance maximale supportée, par exemple 100 mW, alors, nous avons $0{,}100 = I^2 \times 330$. R doit donc être une résistance d'environ 15 Ω.

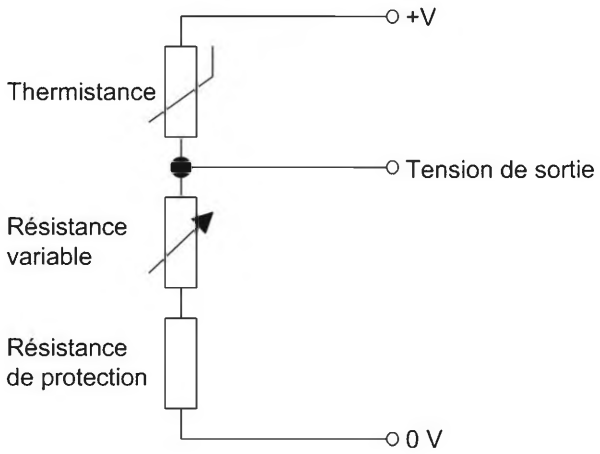


Figure 14.3 – Thermistance dans un circuit diviseur de tension.

En utilisant un amplificateur opérationnel, nous comparons la tension de sortie de la thermistance à celle qui correspond à la température souhaitée. Nous pouvons ainsi obtenir un signal de sortie au niveau haut lorsque la température se trouve au-dessus de la valeur requise et un signal au niveau bas lorsqu'elle se trouve en dessous, ou vice versa. Les amplificateurs opérationnels sont très souvent employés pour produire des signaux tout ou rien en fonction de la valeur relative de deux signaux d'entrée – une tension de référence et la tension d'un capteur (voir Section 4.2.2). L'un des signaux est envoyé à l'entrée non inverseuse, tandis que l'autre est envoyé à l'entrée inverseuse. L'amplificateur opérationnel détermine si la valeur du signal de l'entrée inverseuse se trouve au-dessus ou en dessous de celui de l'entrée non inverseuse. La Figure 14.4 montre la relation entre les entrées et la sortie de l'amplificateur opérationnel. En échangeant les tensions d'entrée et de référence, la polarité de la sortie est inversée.

Nous pouvons ainsi faire en sorte que, lorsque la température passe d'une valeur au-dessus de celle requise à en dessous, le signal de sortie passe du niveau haut au niveau bas. L'API peut ensuite être programmé pour activer une sortie lorsque l'entrée est au niveau bas et allumer l'appareil de chauffage ; lorsque l'entrée est au niveau haut, l'appareil de chauffage est coupé.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

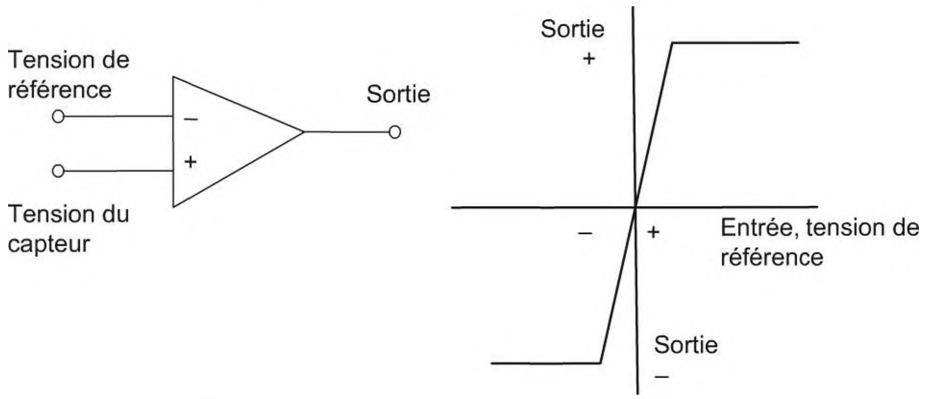


Figure 14.4 – Caractéristique d'un amplificateur opérationnel.

La Figure 14.5 présente le montage qui peut être utilisé et les éléments de base d'un schéma à contacts. L'entrée de l'amplificateur opérationnel est connectée au port d'entrée, dont les contacts sont normalement fermés. Lorsque l'entrée passe au niveau haut, les contacts s'ouvrent. La sortie vers l'appareil de chauffage est prise sur le port de sortie. Par conséquent, la sortie est présente lorsque l'entrée du capteur est au niveau bas et elle est absente lorsque l'entrée est au niveau haut.

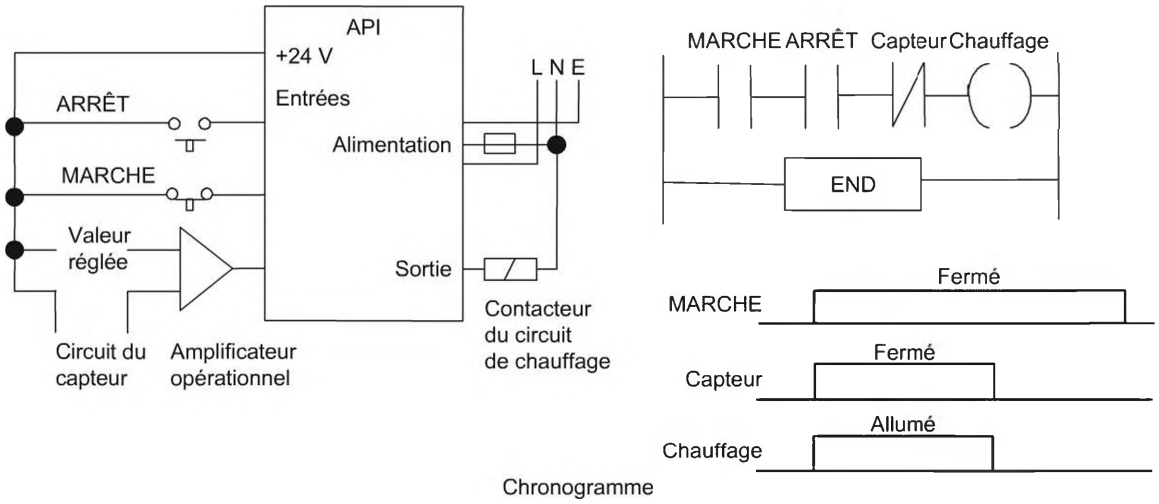


Figure 14.5 – Régulation de la température.

Étudions à présent la régulation de température dans un système plus complexe, un chauffage central domestique (voir Figure 14.6). La chaudière est sous contrôle thermostatique et fournit de l'eau chaude pour les radiateurs de la maison, ainsi que pour le ballon sur lequel sont branchés les robinets. Les moteurs des pompes doivent être mis en marche pour envoyer l'eau chaude depuis la chaudière vers le circuit des radiateurs et/ou le circuit des robinets d'eau chaude, selon la température indiquée par les capteurs situés dans la pièce et dans le ballon. L'intégralité du système est contrôlé par une horloge afin qu'il fonctionne uniquement à certaines heures de la journée. La Figure 14.7 montre une façon de le programmer.

Copyright © 2015 Dunod.

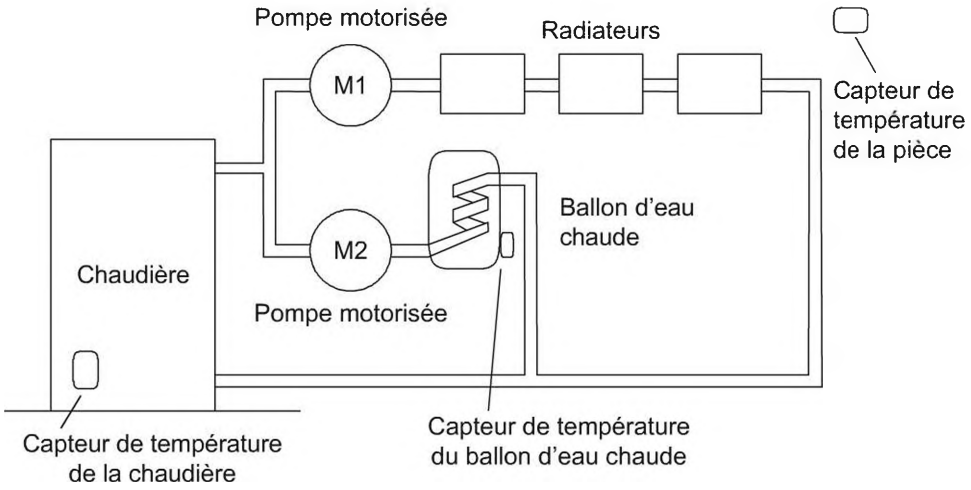


Figure 14.6 – Système de chauffage central.

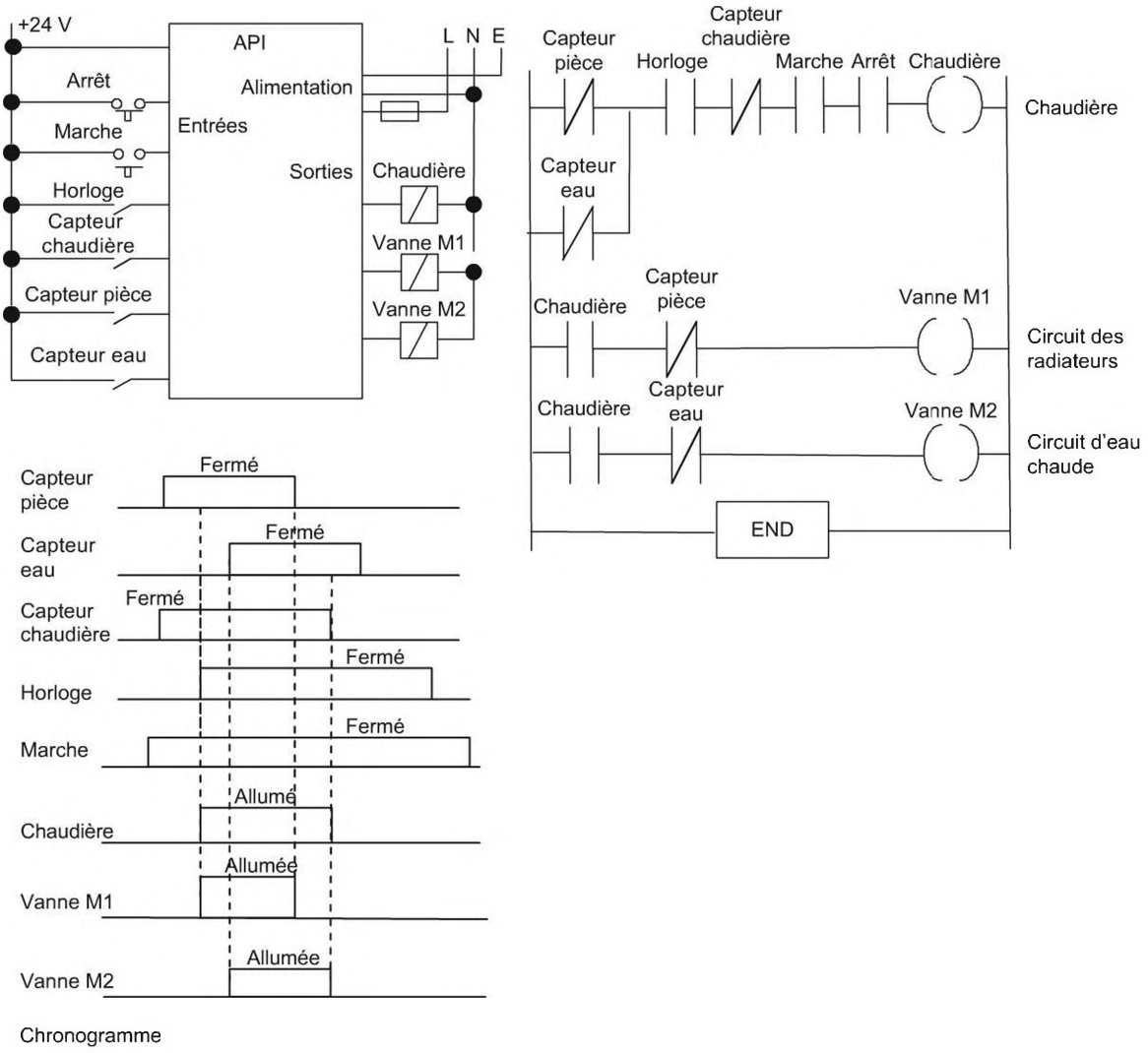


Figure 14.7 – Mise en œuvre du système de chauffage central.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

Les capteurs peuvent être connectés à des amplificateurs opérationnels afin qu'une sortie au niveau haut soit générée lorsque la température mesurée se trouve au-dessus de la valeur fixée et les appareils doivent être éteints lorsque la sortie de l'amplificateur opérationnel passe au niveau bas. Lorsque le système fonctionne, la chaudière est allumée si l'horloge et l'un des capteurs de température n'est pas déclenché. La vanne M1 est mise sous tension si la chaudière est en marche et si le capteur de température de la pièce est au niveau bas. La vanne M2 est mise sous tension si la chaudière est en marche et si le capteur de température de l'eau est au niveau bas.

Un simple système tout ou rien pose un problème. En effet, lorsque la température de la pièce approche de la valeur de référence, le capteur peut réagir à des changements de température très faibles et être continuellement activé ou désactivé. Ainsi, lorsque la température est à la valeur de référence, un léger courant d'air peut le déclencher. Ce problème peut être diminué si la chaudière est mise en marche à une température inférieure à celle à laquelle elle est coupée (voir Figure 14.8). Le terme *zone morte* est utilisé pour désigner les valeurs qui se trouvent entre le seuil de mise en marche et celui d'arrêt. Par exemple, si le thermostat est réglé sur 20 °C, la zone morte peut déclencher son activation lorsque la température tombe en dessous de 19,5 °C et sa désactivation lorsqu'elle est supérieure à 20,5 °C. Dans ce cas, la température doit varier d'un degré pour que l'API allume ou éteigne la chaudière. De petites variations ne provoquent pas un changement d'état du thermostat. Une zone morte large conduit à de grandes fluctuations de la température autour de la température de référence ; une zone morte étroite conduit à une fréquence de mise en marche-extinction plus élevée. Le thermostat bilame de la Figure 14.1 comprend un aimant permanent sur l'un des contacts de l'interrupteur et un petit morceau de fer doux sur l'autre. Cela a pour effet de produire une petite zone morte car, lorsque l'interrupteur est fermé, il faut une augmentation de température significative pour que l'élément bimétallique produise une force suffisante pour séparer les contacts.

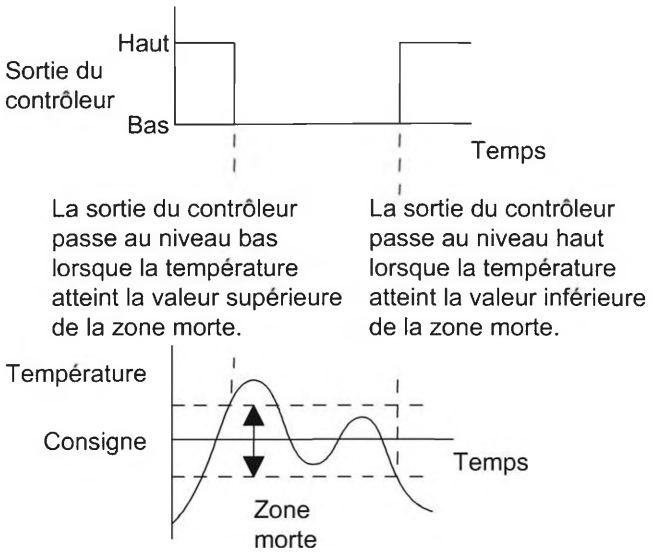


Figure 14.8 – Contrôleur tout ou rien avec une zone morte.

Copyright © 2015 Dunod.

Une autre solution consiste à utiliser deux entrées de l'API, l'une avec une température fixée à la valeur supérieure de la zone morte, l'autre, à la valeur inférieure de la zone morte. Dans ce cas, le programme de la Figure 14.5 est modifié pour obtenir celui de la Figure 14.9. Au démarrage du programme, le chauffage est allumé. Lorsque la température inférieure fixée est atteinte, le capteur inférieur est désactivé. En raison du verrouillage, l'appareil de chauffage reste en marche. Lorsque la température supérieure fixée est atteinte, le capteur supérieur est désactivé et le chauffage est coupé. Lorsque la température baisse, elle doit descendre jusqu'à la température inférieure fixée avant que l'appareil de chauffage ne soit remis en marche.

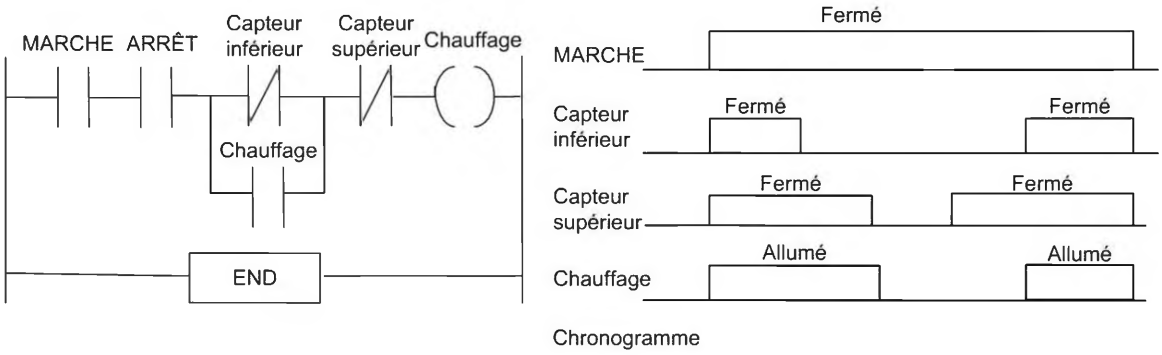


Figure 14.9 – Programme avec zone morte.

Jusqu'à présent, nous avons utilisé une régulation de température simple, de type marche/arrêt, avec un amplificateur comparateur qui produit une sortie à 1 lorsque la température se trouve au-dessus de la température fixée et une sortie à 0 lorsqu'elle est en dessous. La sortie du système de chauffage est donc en tout ou rien. Un système plus élaboré peut se fonder sur une régulation proportionnelle, la sortie vers le système de chauffage étant un signal proportionnel à la différence de température entre la valeur réelle et la valeur souhaitée. Le programme pourrait alors réaliser les tâches suivantes :

1. Lire l'entrée de la température réelle, après conversion analogique-numérique par un CAN.
2. Lire l'entrée de la température souhaitée.
3. Soustraire la température réelle de la température souhaitée.
4. Multiplier le résultat par la constante proportionnelle.
5. Utiliser le résultat pour contrôler la valeur de la sortie vers le système de chauffage.

14.2 Séquencement de vannes

Nous allons à présent examiner des tâches impliquant des vannes de commande directionnelle (voir Section 2.2.2 pour une introduction).

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

14.2.1 Mouvement cyclique

La première tâche consiste à obtenir un mouvement cyclique d'un piston dans un vérin. Il peut s'agir d'un positionnement périodique de pièces dans une machine-outil, avec un autre montage comparable, mais déphasé, pour extraire les pièces usinées. La Figure 14.10 présente un montage possible de l'électrovanne et du vérin, un schéma à contacts envisageable et un chronogramme de chaque sortie.

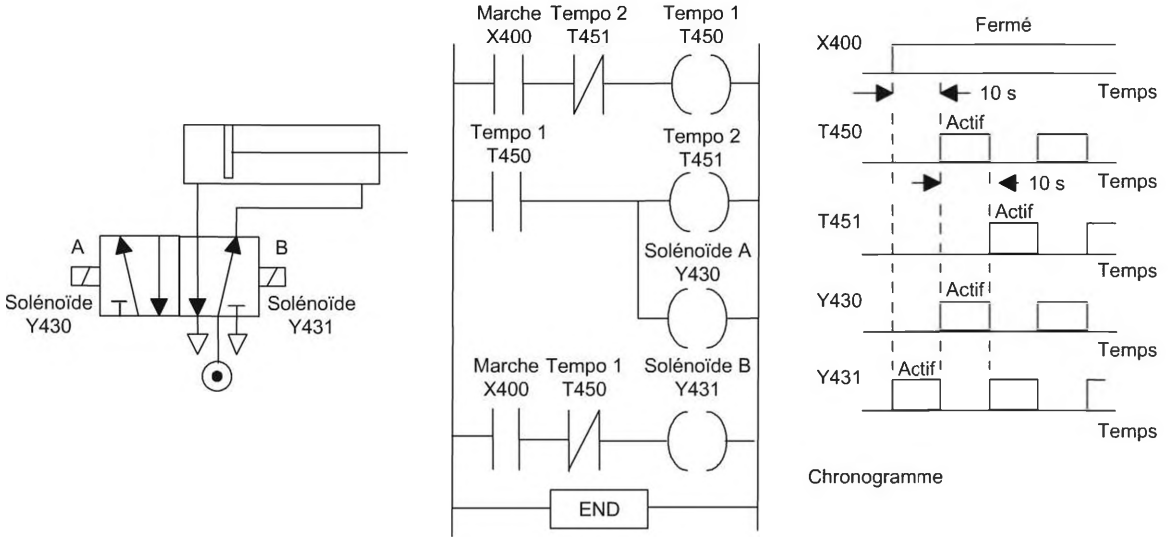


Figure 14.10 – Mouvement cyclique d'un piston.

Supposons que les deux temporisateurs soient réglés à 10 s. Lorsque les contacts de mise en marche X400 sont fermés, le temporisateur T450 démarre. Une sortie est également présente sur Y431. Il s'agit de l'un des solénoïdes utilisés pour actionner la vanne. Lorsqu'il est excité, la pression est appliquée à l'orifice droit du vérin et l'orifice gauche est connecté à l'échappement. Le piston se déplace donc vers la gauche. Après 10 s, les contacts normalement ouverts de T450 se ferment et les contacts normalement fermés de T450 s'ouvrent. Cela arrête la sortie Y431, démarre le temporisateur T451 et active la sortie Y430. Par conséquent, la pression est appliquée à l'orifice gauche du vérin et l'orifice droit est connecté à l'échappement. Le piston se déplace à présent vers la droite. Après 10 s, les contacts normalement fermés de T451 s'ouvrent. Cela déclenche la fermeture des contacts normalement fermés de T450 et l'excitation de Y431. Ainsi, la séquence se répète.

14.2.2 Séquencement

Examinons une autre tâche qui met en jeu les trois pistons A, B et C. Ils doivent être actionnés selon la séquence suivante : A vers la droite, A vers la gauche, B vers la droite, B vers la gauche, C vers la droite, C vers la gauche. Ce type de séquencement s'écrit souvent A+, A–, B+, B–, C+, C–. La Figure 14.11 illustre les vannes que nous pouvons utiliser ; les Figures 14.12 et 14.13 montrent les schémas à contacts correspondants, qui se fondent sur des temporisateurs. Une autre solution consiste à utiliser un registre à décalage.

Copyright © 2015 Dunod.

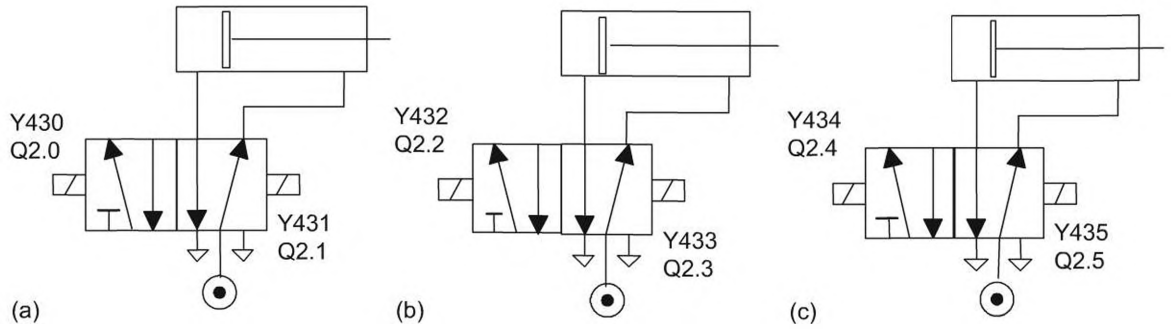


Figure 14.11 – Les vannes.

X400/I0.0 correspond à l'interrupteur de mise en marche. Lorsqu'il est fermé, une sortie est présente sur Y430/Q2.0 et le temporisateur T450/T0 démarre. L'interrupteur de mise en marche est verrouillé par la sortie. Le piston A se déplace vers la droite. Après la durée fixée, K = 4, les contacts normalement fermés du temporisateur T450/relais interne F0.0 s'ouvrent et les contacts normalement ouverts du temporisateur T450/relais interne F0.0 se ferment. Cela désactive Y430/Q2.0, excite Y431/Q2.1 et démarre le temporisateur T451/T1. Le piston A se déplace vers la gauche. Sur la ligne 2, les contacts du temporisateur T450/relais interne F0.0 sont verrouillés et la sortie reste active jusqu'à ce que la durée fixée soit écoulée. À ce moment-là, les contacts normalement fermés du temporisateur T451/relais interne F0.1 s'ouvrent et ses contacts normalement ouverts se ferment. Cela désactive Y431/Q2.1, excite Y432/Q2.2 et démarre le temporisateur T452/T2. Le piston B se déplace vers la droite. Chaque ligne successive excite le solénoïde suivant. Ainsi, chacune des sorties est activée en séquence.

Voici la liste d'instructions du programme précédent pour un API de Mitsubishi :

```
LD      X400    (*Interrupteur de mise en marche*)
OR      Y430
ANI     T450
ANI     Y431
ANI     Y432
ANI     Y433
ANI     Y434
ANI     Y435
OUT     Y430    (*Le piston A se déplace vers la droite*)
OUT     T450    (*Le temporisateur T450 démarre*)
LD      T450
OR      Y431
ANI     T451
OUT     Y431    (*Le piston A se déplace vers la gauche*)
OUT     T451    (*Le temporisateur T451 démarre*)
LD      T451
OR      Y432
ANI     T452
OUT     Y432    (*Le piston B se déplace vers la droite*)
OUT     T452    (*Le temporisateur T452 démarre*)
LD      T452
OR      Y433
ANI     T453
OUT     Y433    (*Le piston B se déplace vers la gauche*)
```

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

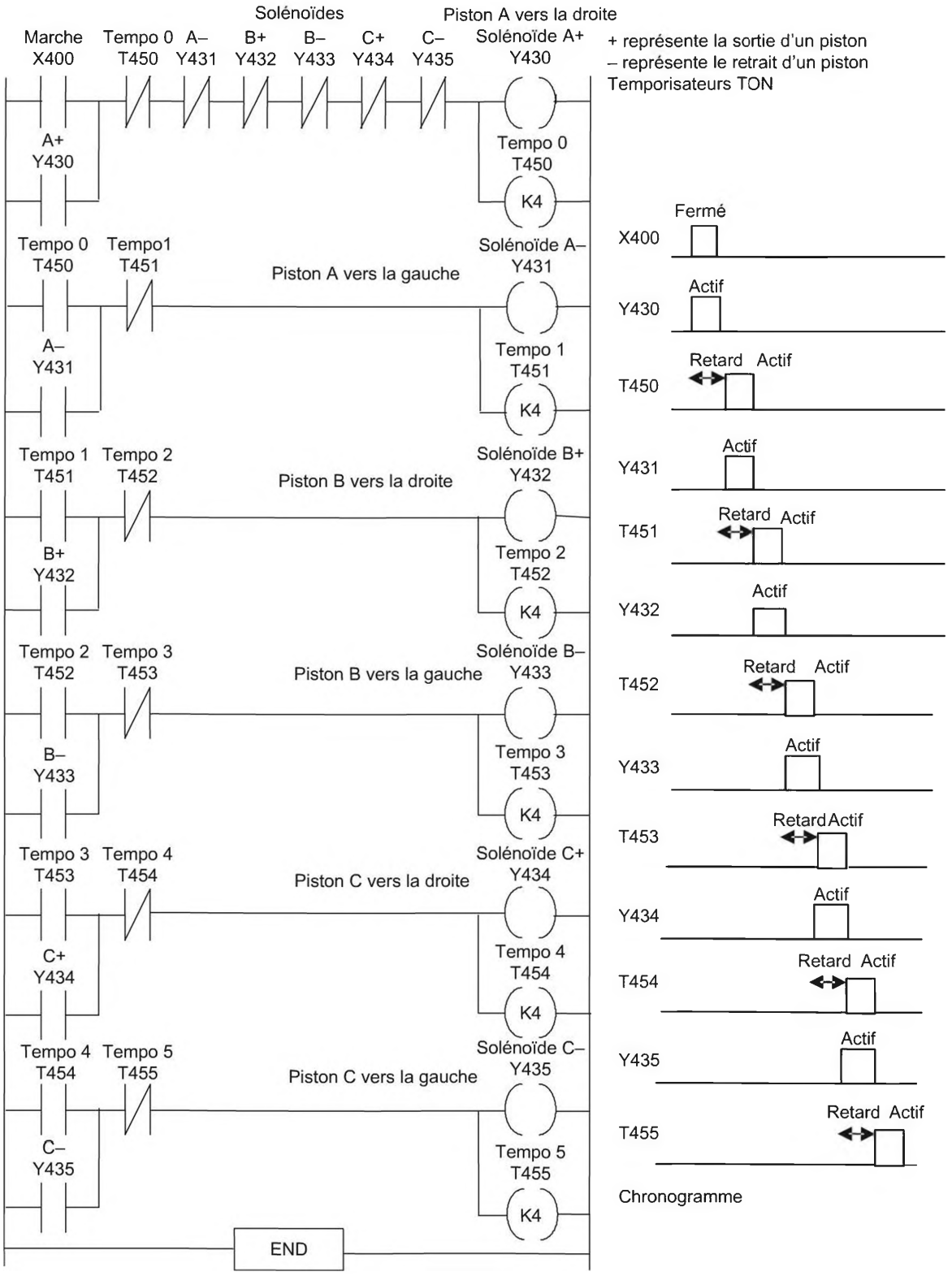


Figure 14.12 – Programme au format Miitsubishi.

Copyright © 2015 Dunod.

```
OUT     T453    (*Le temporisateur T453 démarre*)
LD      T453
OR      Y434
ANI     T454
OUT     Y434    (*Le piston C se déplace vers la droite*)
OUT     T454    (*Le temporisateur T454 démarre*)
LD      T454
OR      Y435
ANI     T455
OUT     Y435    (*Le piston C se déplace vers la gauche*)
OUT     T455    (*Le temporisateur T455 démarre*)
END
```

14.2.3 Séquencement avec un graphe de fonction séquentielle

Illustrons à présent l'utilisation d'un graphe de fonction séquentielle. Il décrit un programme qui met en œuvre une commande séquentielle de vannes et de vérins pneumatiques. Les pistons des vérins doivent être actionnés selon la séquence A+, A−, A+, B+ et ensuite A− et B− simultanément. Autrement dit, le piston A sort totalement, puis se rétracte, puis sort à nouveau, puis le piston B sort totalement et ensuite les deux pistons A et B se rétractent (voir Figure 14.14a). Le graphe de fonction séquentielle correspondant est donné à la Figure 14.14b.

14.2.4 Barrières d'un parking commandées par des vannes

Étudions l'emploi de vannes pneumatiques pour actionner les barrières d'un parking. La barrière d'entrée doit être ouverte lorsque la somme correcte a été insérée dans la borne de paiement ; la barrière de sortie s'ouvre lorsqu'une voiture est détectée à côté de cette barrière. La Figure 14.15 montre le type de système que nous pouvons utiliser. Les vannes qui actionnent les barrières disposent d'un solénoïde permettant de placer la barrière dans une position et d'un ressort de retour pour obtenir la seconde position. Lorsque le solénoïde n'est pas excité, la position de la barrière correspond à celle obtenue par le ressort. Les vannes sont utilisées pour déclencher le déplacement des pistons. Lorsque les pistons se déplacent vers le haut, la barrière pivote autour de son axe et se lève. Lorsqu'un piston se rétracte, sous l'action du ressort de retour, la barrière s'abaisse. Lorsqu'une barrière est fermée, elle actionne un interrupteur ; lorsqu'elle est ouverte, elle déclenche un autre interrupteur. Ces interrupteurs produisent des entrées qui indiquent la position de la barrière. Des capteurs indiquent si la somme correcte a été insérée dans la borne de paiement lorsqu'un véhicule veut entrer et détectent lorsqu'un véhicule est proche de la barrière de sortie.

La Figure 14.16 montre le schéma à contacts correspondant, pour un API de (a) Mitsubishi et de (b) Siemens. La sortie Y430/Q2.0 vers le solénoïde 1 qui permet de lever la barrière d'entrée se produit lorsque la sortie du capteur de la borne de paiement produit l'entrée X400/I0.0. Y430/Q2.0 est verrouillée et reste active jusqu'à l'ouverture du relais interne M100/F0.1. La sortie n'est pas activée si la barrière est en cours de fermeture et si la sortie Y431/Q2.1 vers le solénoïde 2 est activée. Le temporisateur T450/T1 permet de maintenir la barrière levée pendant 10 s. Il est démarré par l'entrée X402/I0.2 du capteur qui indique que la barrière

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

est levée. Une fois cette durée écoulée, la sortie Y431/Q2.1 est activée, le solénoïde 2 est excité et la barrière s'abaisse. La barrière de sortie est levée par la sortie Y432/Q2.2 vers le solénoïde 3 lorsqu'un capteur détecte la présence d'une voiture et produit l'entrée X401/I0.1. Lorsque la barrière est levée, le temporisateur T451/T2 permet de la maintenir ainsi pendant 10 s. Il est démarré par l'entrée X404/I0.4, issue d'un capteur qui indique que la barrière est levée. Lorsque les 10 s sont écoulées, la sortie Y433/Q2.3 est activé, le solénoïde 4 est excité et la barrière s'abaisse.

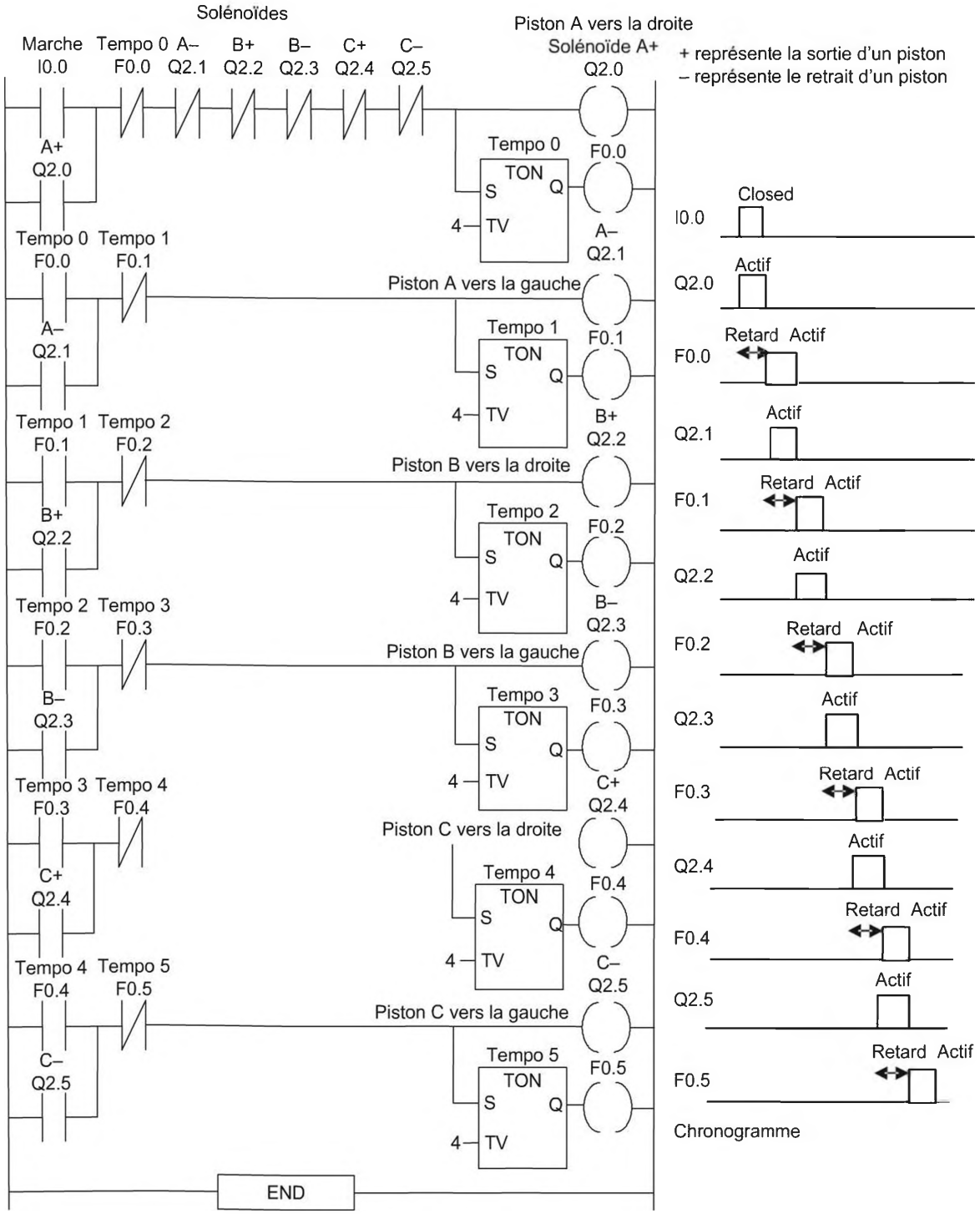


Figure 14.13 – Programme au format Siemens.

Copyright © 2015 Dunod.

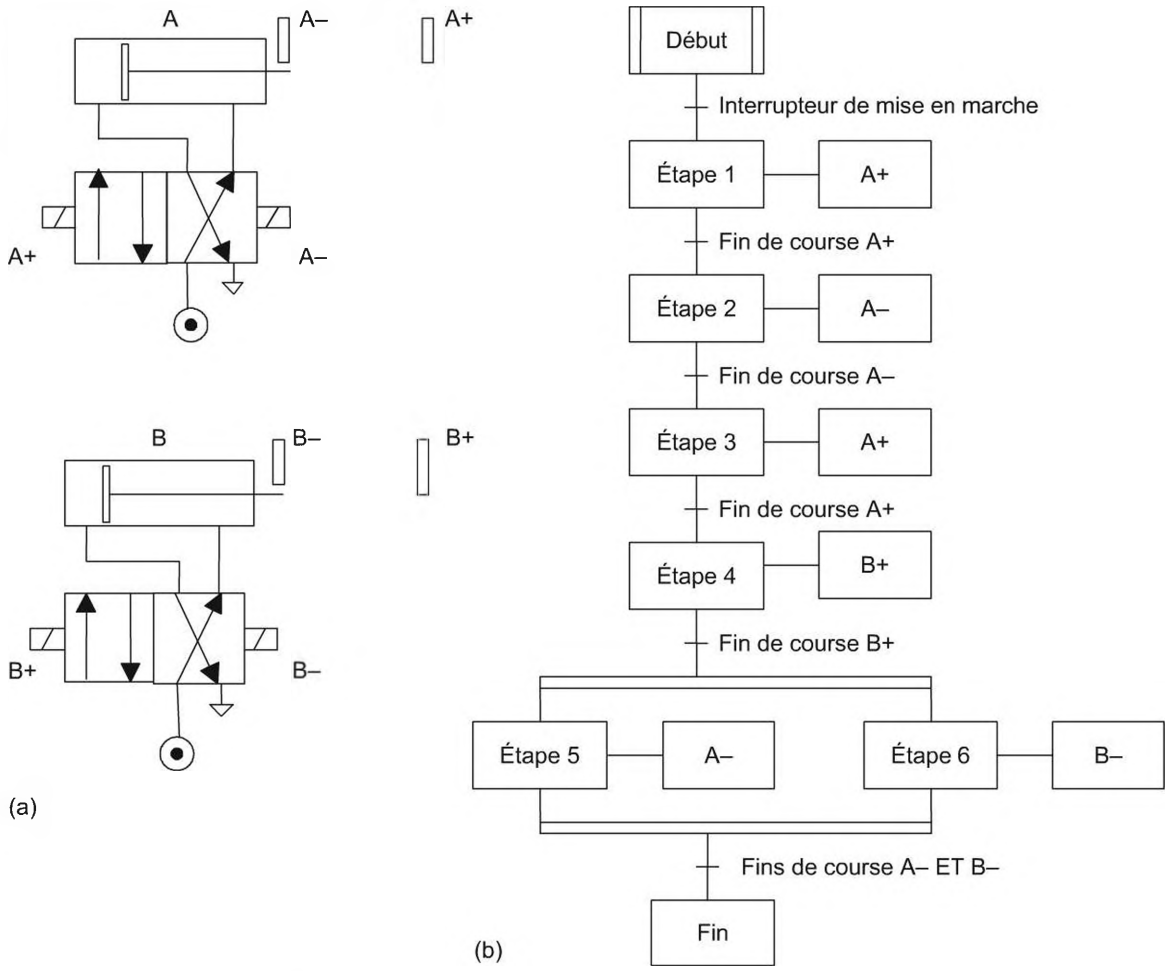


Figure 14.14 – Séquencement de pistons : A+, A-, A+, B+, puis, simultanément, A- et B-.

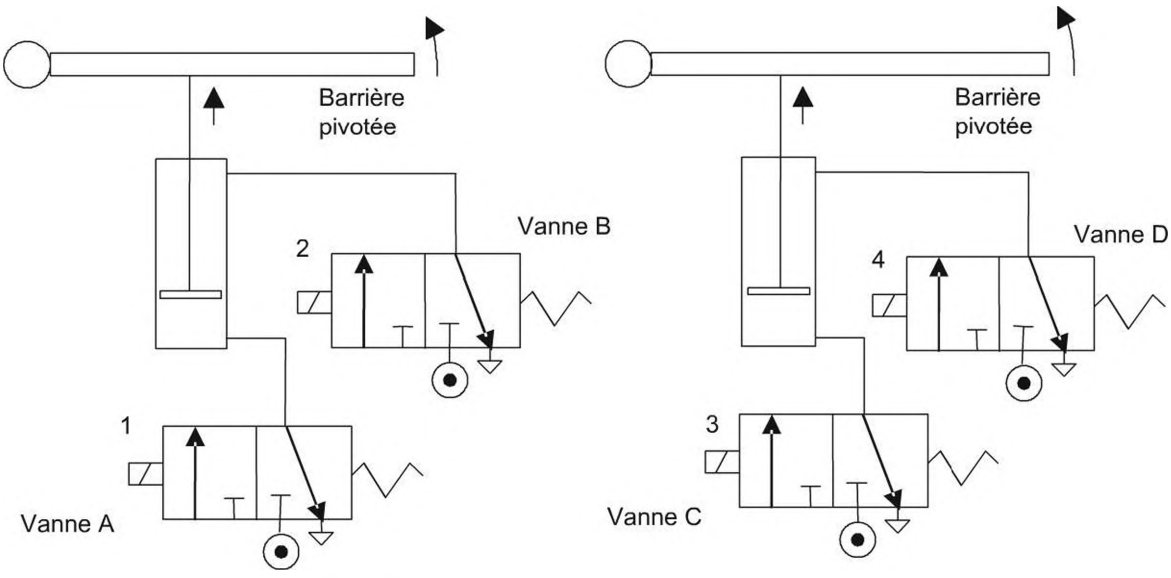
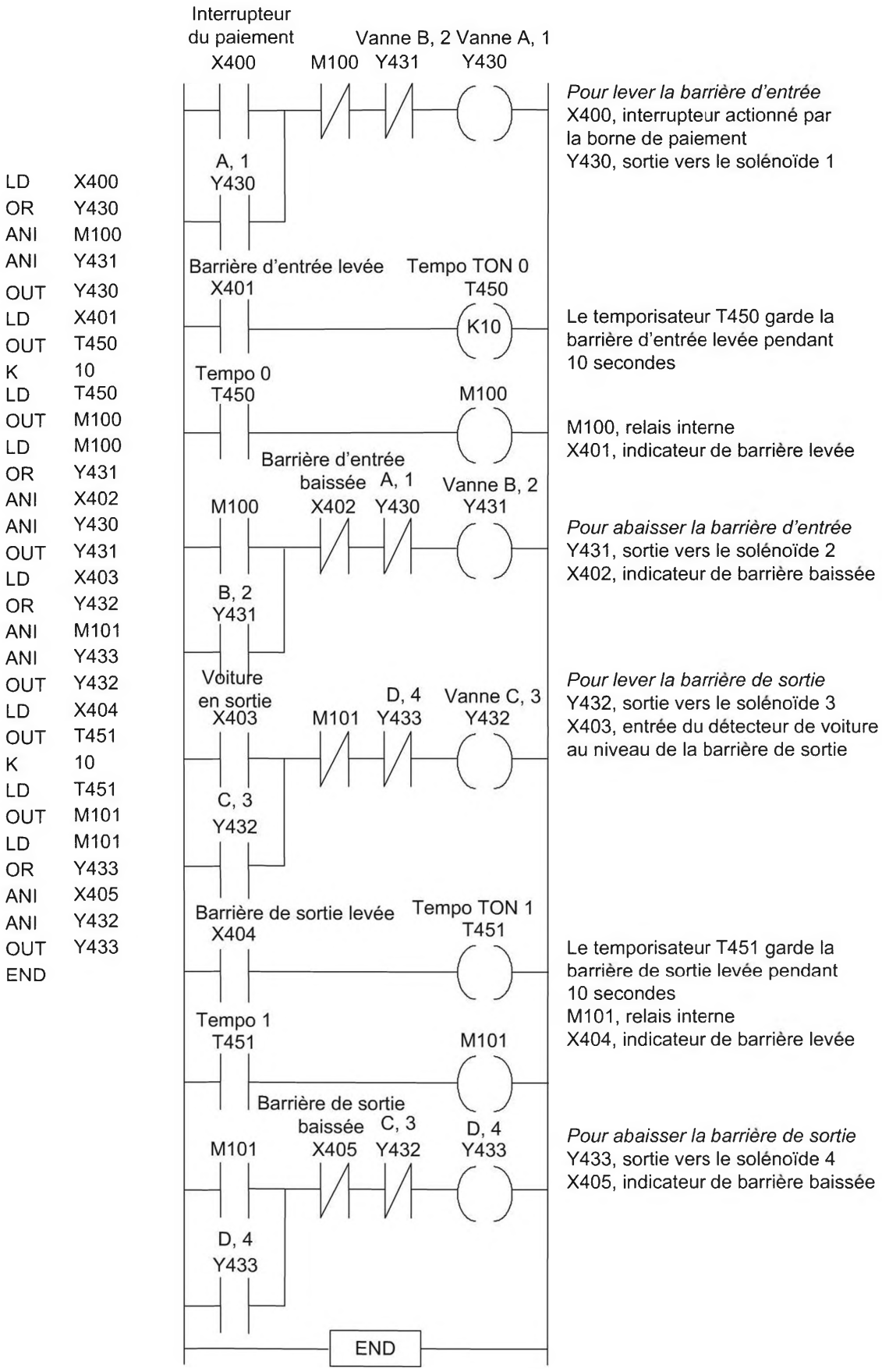


Figure 14.15 – Système à vannes et vérins.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.




```
LD    X400
OR    Y430
ANI   M100
ANI   Y431
OUT   Y430
LD    X401
OUT   T450
K     10
LD    T450
OUT   M100
LD    M100
OR    Y431
ANI   X402
ANI   Y430
OUT   Y431
LD    X403
OR    Y432
ANI   M101
ANI   Y433
OUT   Y432
LD    X404
OUT   T451
K     10
LD    T451
OUT   M101
LD    M101
OR    Y433
ANI   X405
ANI   Y432
OUT   Y433
END
```

Pour lever la barrière d'entrée
X400, interrupteur actionné par la borne de paiement
Y430, sortie vers le solénoïde 1

Le temporisateur T450 garde la barrière d'entrée levée pendant 10 secondes

M100, relais interne
X401, indicateur de barrière levée

Pour abaisser la barrière d'entrée
Y431, sortie vers le solénoïde 2
X402, indicateur de barrière baissée

Pour lever la barrière de sortie
Y432, sortie vers le solénoïde 3
X403, entrée du détecteur de voiture au niveau de la barrière de sortie

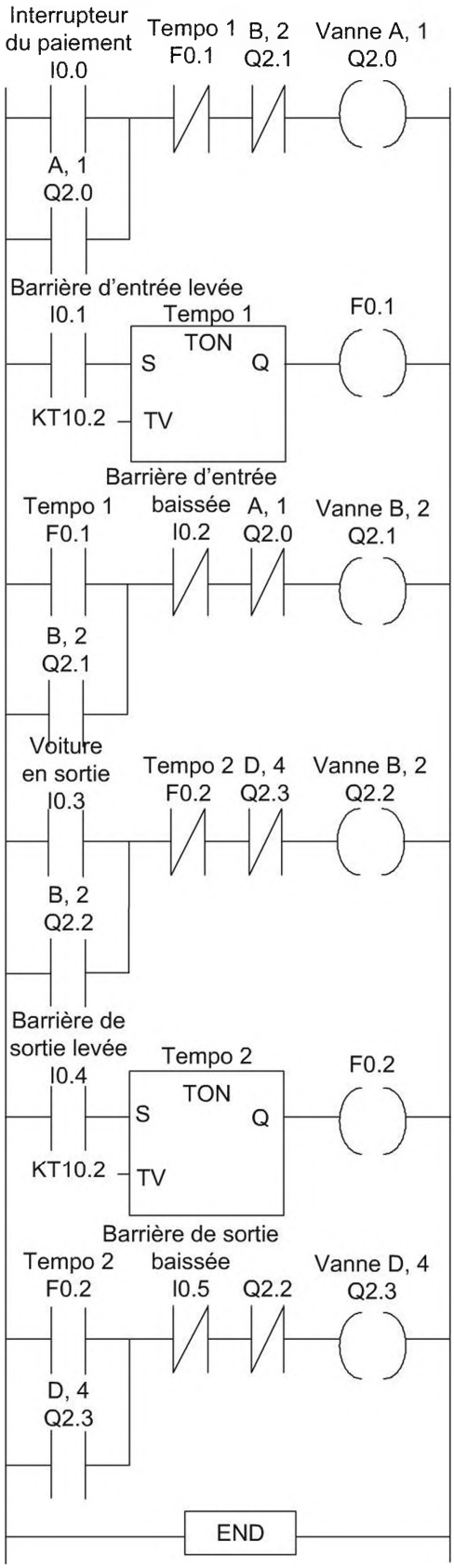
Le temporisateur T451 garde la barrière de sortie levée pendant 10 secondes
M101, relais interne
X404, indicateur de barrière levée

Pour abaisser la barrière de sortie
Y433, sortie vers le solénoïde 4
X405, indicateur de barrière baissée

Figure 14.16(a) – Programme de commande de la barrière, format Mitsubishi.

Copyright © 2015 Dunod.

```
A     I0.0
O     Q2.0
AN    F0.1
AN    Q2.1
=     Q2.0
A     I0.1
LKT   I0.2
SR    T0
A     T0
=     Q2.0
A     F0.1
O     Q2.1
AN    I0.2
AN    Q2.0
=     Q2.1
A     I0.3
O     Q2.2
AN    F0.2
AN    Q2.3
=     Q2.2
A     I0.4
LKT   I0.2
SR    T1
A     T1
=     F0.2
A     F0.2
O     Q2.3
AN    I0.5
AN    Q2.2
=     Q2.3
END
```



Pour lever la barrière d'entrée
I0.0, interrupteur actionné par la borne de paiement
Q2.0, sortie vers le solénoïde 1

Le temporisateur T1 garde la barrière d'entrée levée pendant 10 secondes
F0.1, relais interne
I0.1, indicateur de barrière levée

Pour abaisser la barrière d'entrée
Q2.1, sortie vers le solénoïde 2
I0.2, indicateur de barrière baissée

Pour lever la barrière de sortie
Q2.2, sortie vers le solénoïde 3
I0.3, entrée du détecteur de voiture au niveau de la barrière de sortie

Le temporisateur T2 garde la barrière de sortie levée pendant 10 secondes
F0.2, relais interne
I0.4, indicateur de barrière levée

Pour abaisser la barrière de sortie
Q2.3, sortie vers le solénoïde 4
I0.5, indicateur de barrière baissée

Figure 14.16(b) – Programme de commande de la barrière, format Siemens.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

Voici les entrées et les sorties du programme Mitsubishi :

Entrée		Sortie	
X400	Interrupteur actionné par le paiement.	Y430	Vanne A, solénoïde 1
X401	Entrée lorsque la barrière d'entrée est levée.	Y431	Vanne B, solénoïde 2
X402	Entrée lorsque la barrière d'entrée est baissée.	Y432	Vanne C, solénoïde 3
X403	Entrée lorsqu'une voiture se trouve au niveau de la barrière de sortie.	Y433	Vanne D, solénoïde 4
X404	Entrée lorsque la barrière de sortie est levée.		
X405	Entrée lorsque la barrière de sortie est baissée.		

Voici celles du programme Siemens :

Entrée		Sortie	
I0.0	Interrupteur actionné par le paiement.	Q2.0	Vanne A, solénoïde 1
I0.1	Entrée lorsque la barrière d'entrée est levée.	Q2.1	Vanne B, solénoïde 2
I0.2	Entrée lorsque la barrière d'entrée est baissée.	Q2.2	Vanne C, solénoïde 3
I0.3	Entrée lorsqu'une voiture se trouve au niveau de la barrière de sortie.	Q2.3	Vanne D, solénoïde 4
I0.4	Entrée lorsque la barrière de sortie est levée.		
I0.5	Entrée lorsque la barrière de sortie est baissée.		

Nous pouvons ajouter à ce programme un système qui contrôle le nombre de véhicules présents dans le parking. Un panneau « Disponible » s'allume lorsque le parking n'est pas plein et un panneau « Complet » s'allume lorsqu'il n'y a plus de places libres. Pour cela, il suffit d'utiliser un compteur progressif-dégressif. La Figure 14.17 présente le schéma à contacts possible pour un API de Siemens.

14.2.5 Réinitialisation contrôlée de vérins

Pendant le fonctionnement d'un système qui met en œuvre de nombreux vérins, il est possible que l'un des éléments tombe en panne et laisse des vérins dans des positions non sécurisées. Nous pouvons modifier le programme pour qu'une entrée de réinitialisation replace les pistons dans leur position initiale. Par exemple, avec trois vérins A, B et C et la contrainte de séquencement A+, B+, C+, C–, B–, A–, nous pouvons inclure une entrée RESET qui replace tous les pistons dans leur position rétractée, c'est-à-dire A–, B– et C–. La Figure 14.18 présente un tel programme.

En présence d'un défaut, une opération d'arrêt potentiellement plus sûre consiste à rétracter les pistons en respectant une séquence particulière. La Figure 14.19 montre un programme qui rétracte les pistons selon la séquence C–, B–, A–.

Copyright © 2015 Dunod.

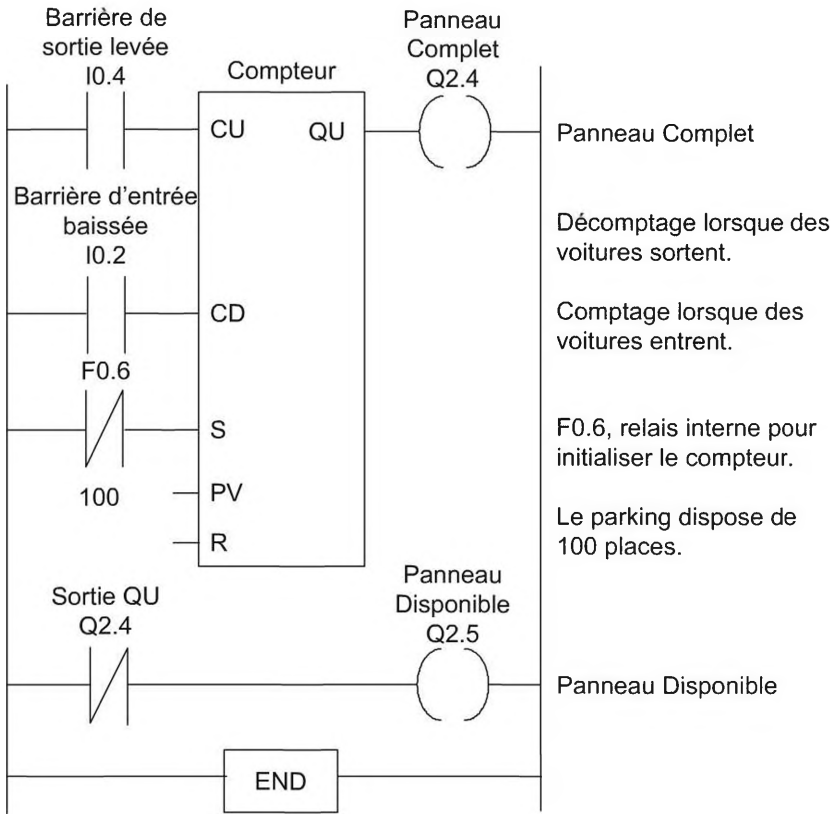


Figure 14.17 – Parking à voitures avec panneaux disponible/complet.

14.3 Commande d'une bande transporteuse

Étudions à présent un programme qui sert à compter le nombre d'articles placés sur une bande transporteuse et qui déclenche une alerte lorsque ce nombre atteint 100. Ce programme pourrait faire partie d'un programme plus important utilisé pour contrôler une unité de production. Un capteur de proximité permet de détecter la pose d'un article sur le convoyeur ; à chaque fois, un signal à 1 est produit. Le programme correspondant, au format Allen-Bradley, est proposé à la Figure 14.20.

Dans cette gestion de la bande transporteuse, il faut également s'assurer, qu'à tout moment, un trop grand nombre d'articles ne soient pas déposés sur le convoyeur. Pour cela, le programme peut imposer un délai entre la dépose d'un article sur la banque transporteuse et le suivant. La Figure 14.21 présente les éléments du programme qui répondent à cette spécification. Lorsqu'un article passe devant le détecteur de proximité, le retard à l'enclenchement est démarré et, lorsqu'il est écoulé, un autre article est autorisé à entrer sur la bande transporteuse.

14.3.1 Emballage de bouteilles

Considérons une ligne de production qui met en œuvre une bande transporteuse pour convoyer des bouteilles vers une unité d'emballage. Les bouteilles sont chargées sur le convoyeur, leur remplissage est vérifié, la capsule est posée et le nombre

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

correct de bouteilles (quatre) sont placées dans un carton. Les opérations de contrôle requises sont donc les suivantes. Si une bouteille n'est pas remplie, la bande transporteuse est stoppée. La machine de capsulage est activée lorsqu'une bouteille se trouve au bon emplacement, la bande transporteuse étant stoppée pendant cette opération. Quatre bouteilles sont comptées et la machine d'emballage est activée, la bande transporteuse étant stoppée si une autre bouteille arrive au point d'emballage à ce moment-là. Une alarme est déclenchée lorsque le convoyeur est arrêté.

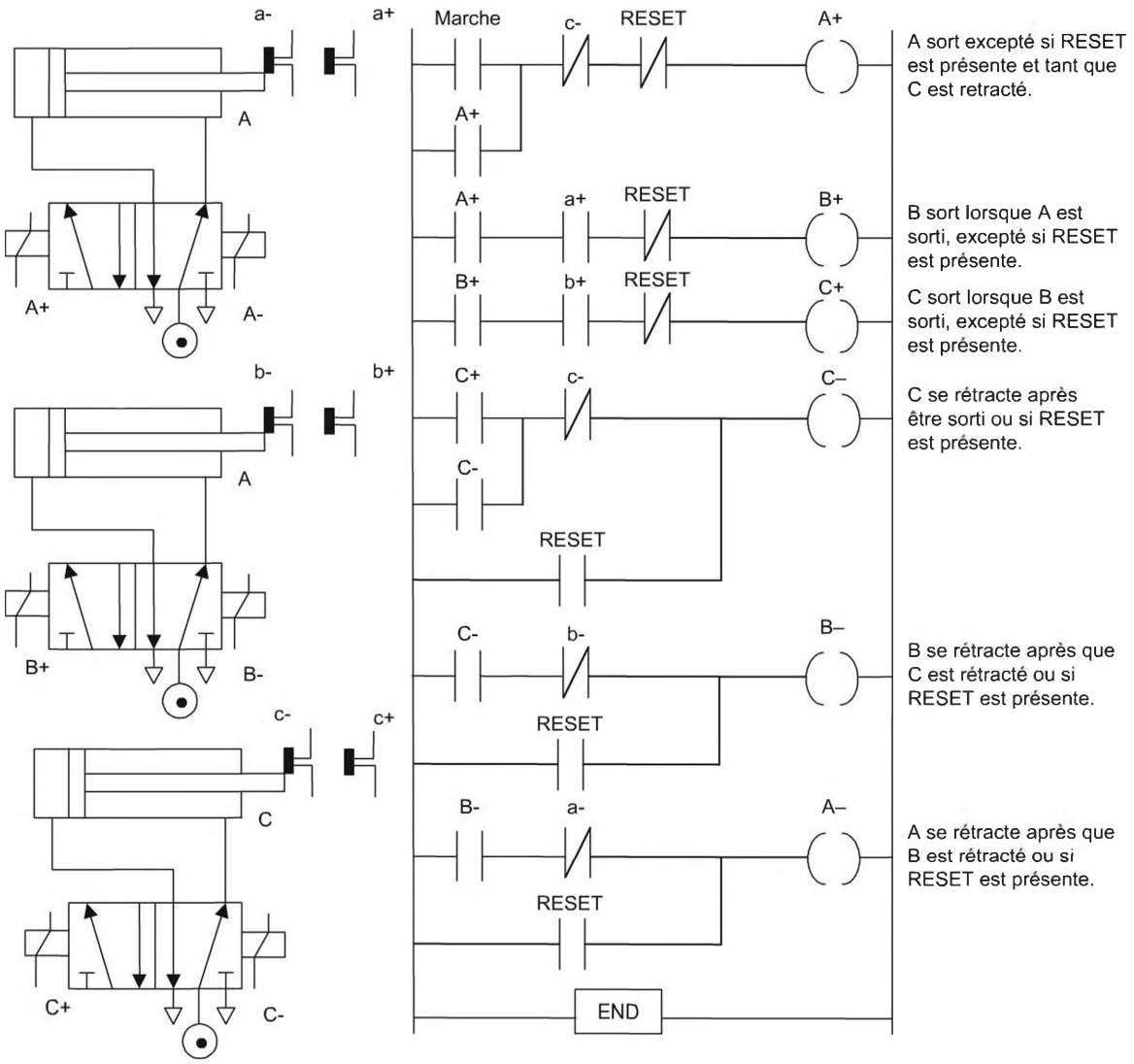


Figure 14.18 – Réinitialisation simultanée de tous les vérins.

Pour déterminer si une bouteille est pleine, nous pouvons utiliser un capteur photoélectrique qui actionne un interrupteur (entrée X402/I0.2). La présence d'une bouteille pour la machine de capsulage peut également être indiquée par un capteur photoélectrique (entrée X403/I0.3). L'entrée du compteur qui dénombre les quatre bouteilles peut également provenir d'un capteur photoélectrique (entrée X404/I0.4). Les autres entrées peuvent être des interrupteurs de mise en marche

Copyright © 2015 Dunod.

(entrée X400/I0.0) et d'arrêt (entrée X401/I0.1) du convoyeur et un signal (entrée X405/I0.5) fourni par la machine d'emballage lorsqu'elle est en action, a reçu quatre bouteilles et est indisponible. La Figure 14.22 propose un schéma à contacts correspondant au format Mitsubishi, tandis que la Figure 14.23 le donne au format Siemens.

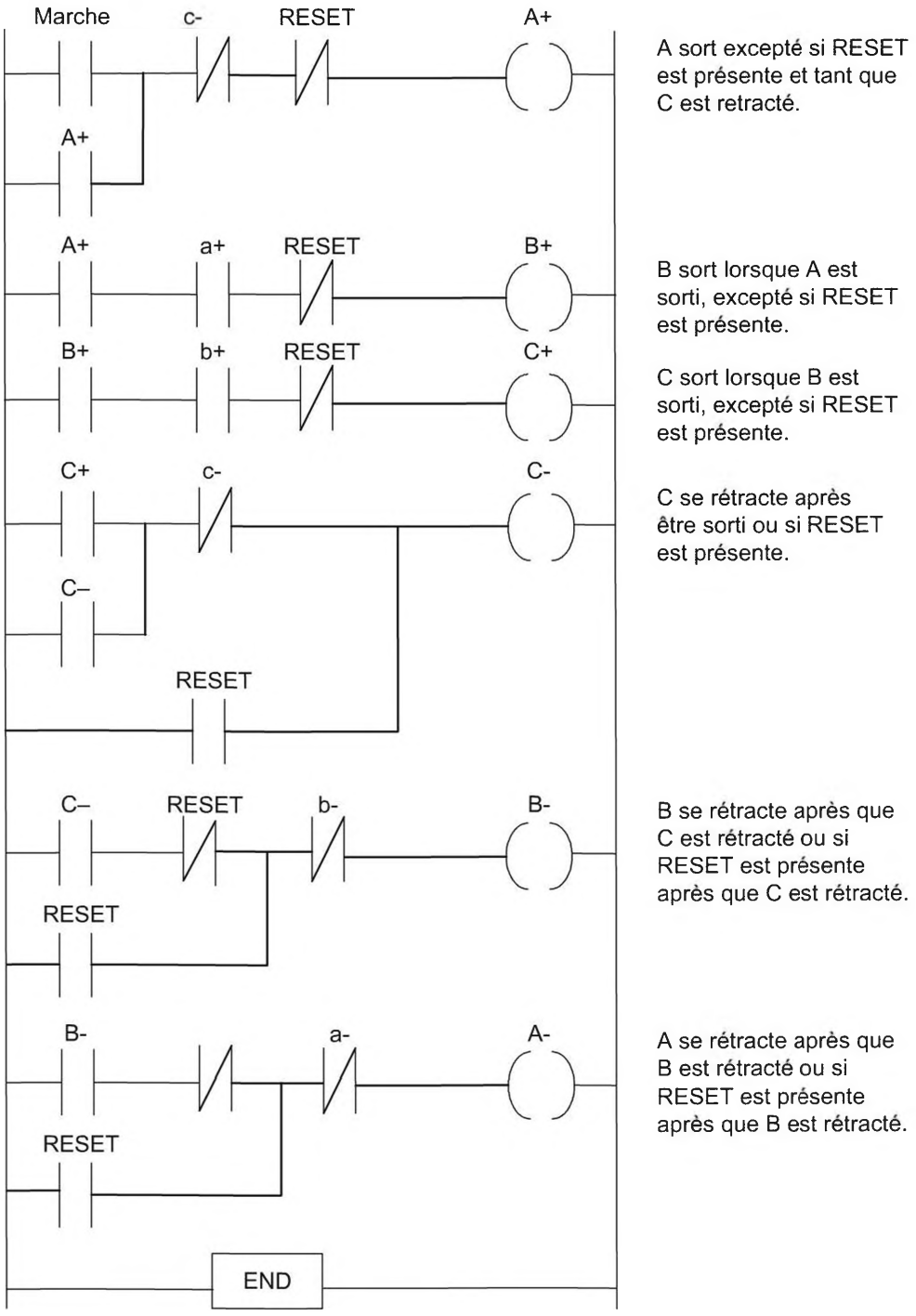


Figure 14.19 – Réinitialisation dans l'ordre C–, B–, A–.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

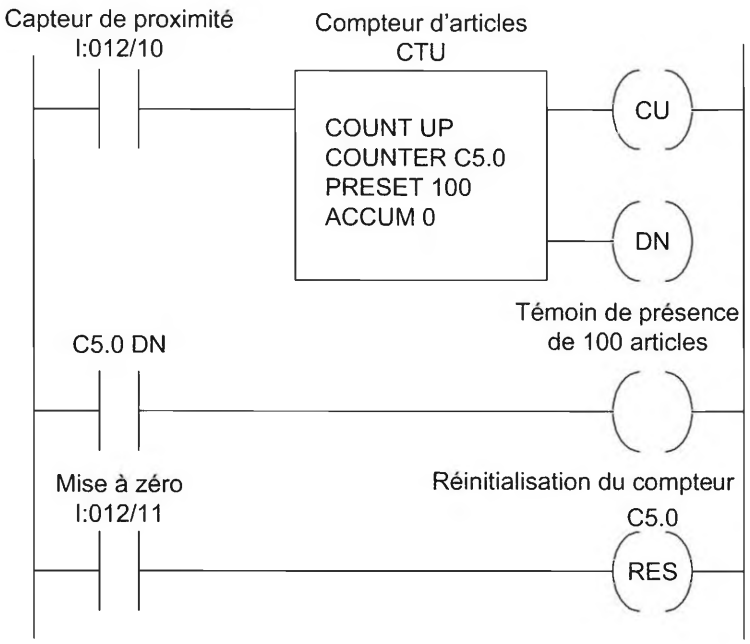


Figure 14.20 – Bande transporteuse avec comptage des articles.

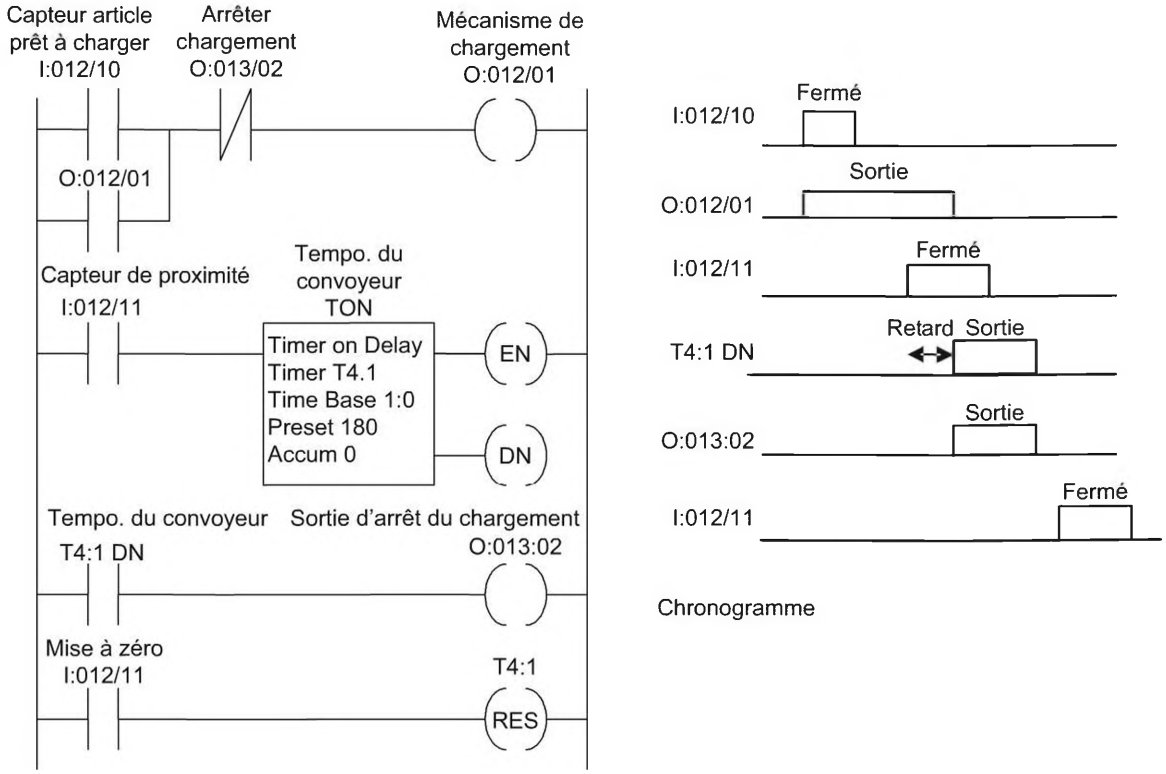
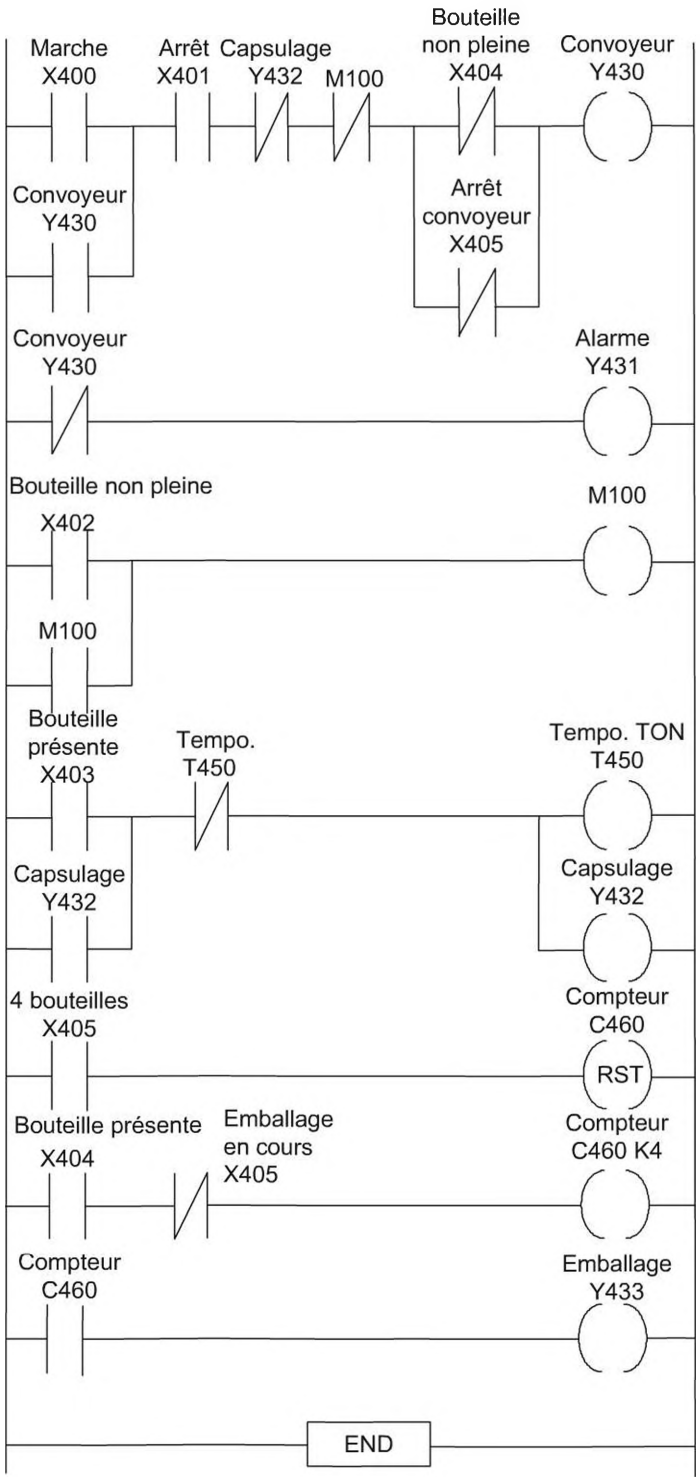


Figure 14.21 – Bande transporteuse avec délai.

Copyright © 2015 Dunod.



Y430, sortie vers le convoyeur.
X400, bouton de mise en marche.
X401, bouton d'arrêt, fermé de manière externe.
Le convoyeur est stoppé par l'activation de Y232, M100, X404 ou X405.

Y431, sortie vers l'alarme.
L'alarme est déclenchée lors de l'arrêt du convoyeur.

M100, relais interne.
Il est activé par la fermeture de X402 lorsqu'une bouteille n'est pas pleine.
Il arrête le convoyeur.

T450, temporisateur.
Il stoppe le convoyeur pendant la durée nécessaire au capsulage de la bouteille. Y432 active la machine de capsulage et arrête le convoyeur.

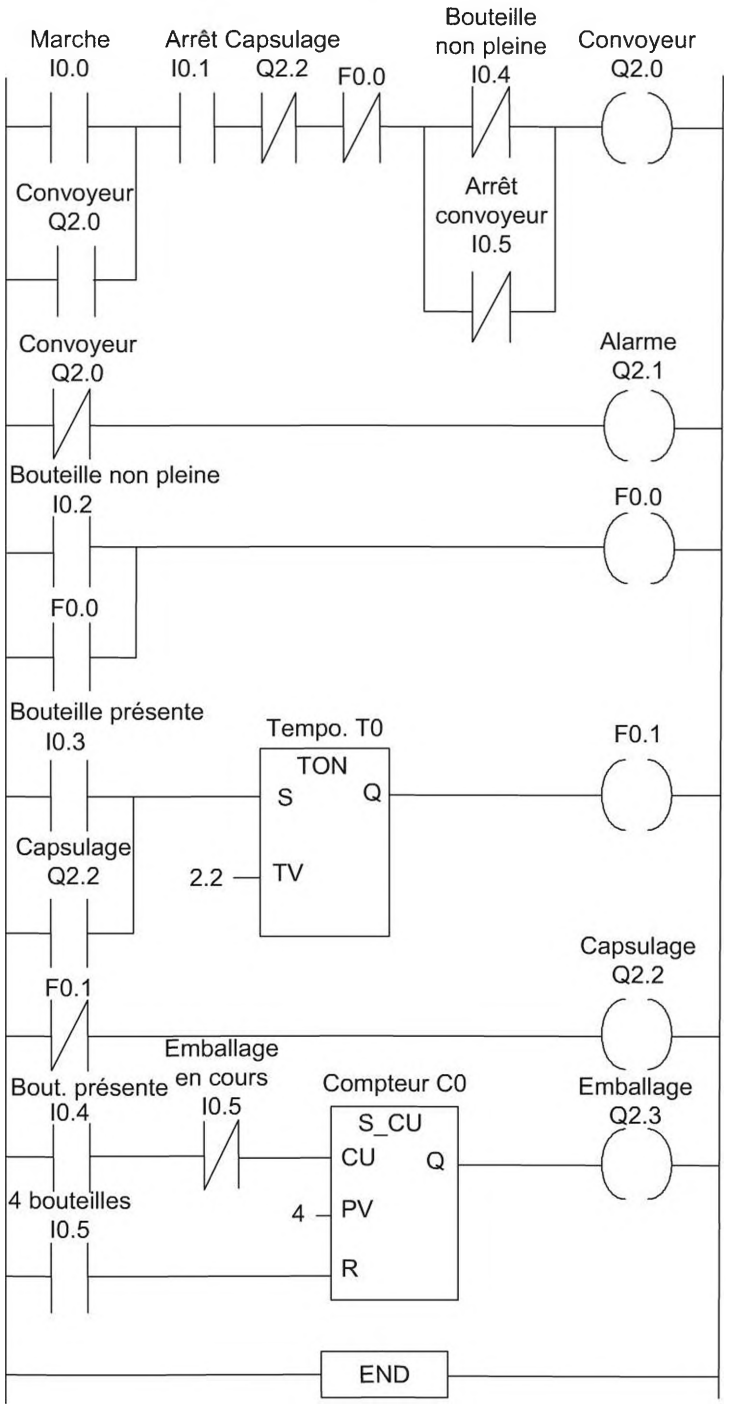
Réinitialisation du compteur lorsque la machine d'emballage dispose de quatre bouteilles.

X404, entrée lorsqu'une bouteille est détectée.
X405 s'ouvre lorsque l'emballage est en cours. Comptage de quatre bouteilles.

Y433 active la machine d'emballage lorsque C460 a compté quatre bouteilles.

Figure 14.22 – Programme d'emballage des bouteilles (version Mitsubishi).

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.



Q2.0, sortie vers le convoyeur.
I0.0, bouton de mise en marche.
I0.1, bouton d'arrêt, fermé de manière externe.
Le convoyeur est stoppé par l'activation de Q2.2, F0.0, I0.4 ou I0.5.

Q2.1, sortie vers l'alarme.
L'alarme est déclenchée lors de l'arrêt du convoyeur.

F0.0, relais interne.
Il est activé par la fermeture de I0.2 lorsqu'une bouteille n'est pas pleine.
Il arrête le convoyeur.

T1, temporisateur.
Il stoppe le convoyeur pendant la durée nécessaire au capsulage de la bouteille.

Q2.2 active la machine de capsulage et arrête le convoyeur.

I0.4, entrée lorsqu'une bouteille est détectée.
I0.5 s'ouvre lorsque l'emballage est en cours. Comptage de quatre bouteilles.

Q2.3 active la machine d'emballage lorsque le compteur a compté quatre bouteilles.

Figure 14.23 – Programme d'emballage des bouteilles (version Siemens).

Voici la liste d'instructions du programme pour un API de Mitsubishi :

```
LD      X400    (*Première ligne*)
OR      Y430
AN      X401
ANI     Y432
ANI     M100
LDI     X404
ORI     X405
ANB
```

Copyright © 2015 Dunod.

```
OUT     Y430
LDI     Y430    (*Deuxième ligne*)
OUT     Y431
LD      X402    (*Troisième ligne*)
OR      M100
OUT     M100
LD      X403    (*Quatrième ligne*)
OR      Y432
ANI     T450
OUT     T450
K       2       (*2 secondes pour le capsulage*)
OUT     Y432
LD      X405    (*Cinquième ligne*)
RST     C460
LD      X404    (*Sixième ligne*)
ANI     X405
OUT     C460
K       4       (*Compter quatre bouteilles*)
LD      C460    (*Septième ligne*)
OUT     Y433
END             (*Dernière ligne*)
```

Voici la liste d'instructions correspondante pour un API de Siemens :

```
A       I0.0    (*Première ligne*)
O       Q2.0
A       I0.1
AN      Q2.2
AN      F0.0
(AN     I0.4
ON      I0.5
)
=       Q2.0
AN      Q2.0    (*Deuxième ligne*)
=       Q2.1
A       I0.2    (*Troisième ligne*)
O       F0.0
=       F0.0
A       I0.3    (*Quatrième ligne*)
O       Q2.2
LKT     2.2     (*2 secondes pour le capsulage*)
SR      T0
A       T0
=       F0.1
AN      F0.1    (*Cinquième ligne*)
=       Q2.2
A       I0.4    (*Sixième ligne*)
AN      I0.5
CU      C0
LKC     4       (*Compter quatre bouteilles*)
A       I0.5
R       C0
=       Q2.3
END             (*Dernière ligne*)
```

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

14.4 Contrôle d'un processus

Dans cette section, nous illustrons l'utilisation d'un graphe de fonction séquentielle pour écrire un programme. Le processus (voir Figure 14.24a) concerne le remplissage de deux conteneurs par deux liquides. Lorsque les conteneurs sont pleins, ils sont vidés dans une chambre de mélange, d'où est ensuite extrait le mélange des liquides. Puis, le processus se répète. La Figure 14.24b montre le type de vanne que nous pouvons utiliser dans un tel processus. Lorsque son solénoïde est excité, la vanne laisse passer le liquide ; dans le cas contraire, un ressort de rappel la referme.

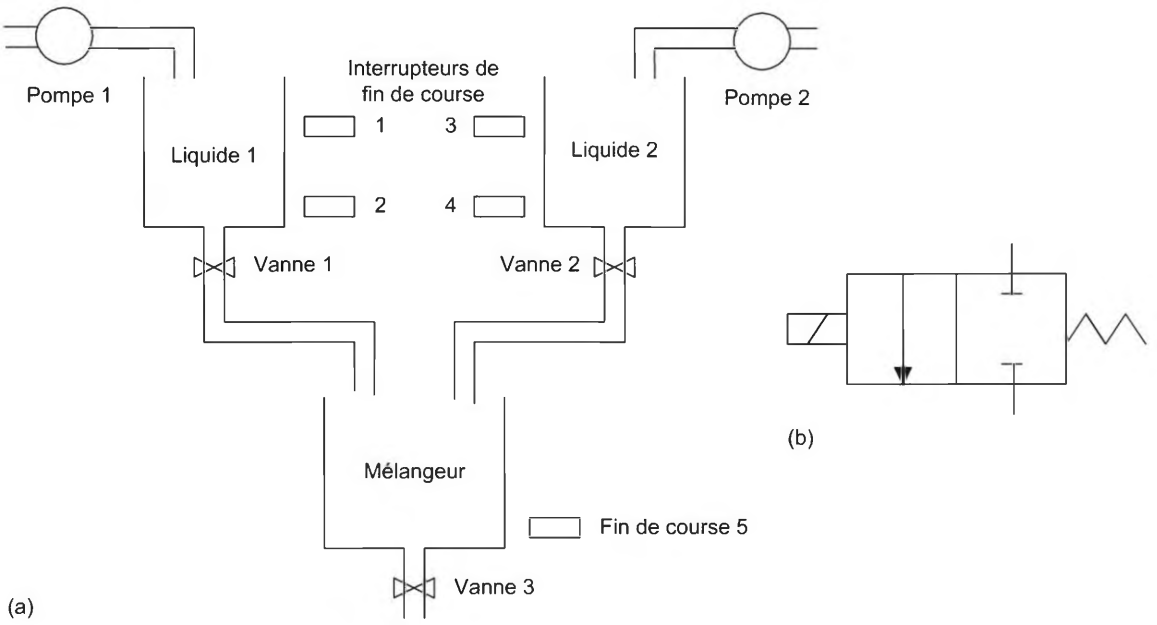


Figure 14.24 – (a) Opération de mélange et (b) une vanne.

La Figure 14.25 présente le graphe de fonction séquentielle correspondant. Lorsque le bouton de démarrage est activé, les étapes de remplissage 1 et 2 se produisent simultanément par mise en marche des pompes 1 et 2. Lorsque l'interrupteur de fin de course 1 est activé, le remplissage 1 cesse. De même, lorsque l'interrupteur de fin de course 3 est activé, le remplissage 2 s'arrête. Les conteneurs des liquides 1 et 2 sont alors pleins. Lorsque les deux interrupteurs de fin de course 1 et 3 sont activés, les conteneurs commencent à se vider par ouverture des vannes 1 et 2. Lorsque les fins de course 2 et 4 sont activés, les conteneurs sont vides. L'étape suivante, le mélange des liquides, se produit lorsque les interrupteurs de fin de course 2 et 4 sont tous deux activés. Après une durée de 100 s, le mélange cesse et le liquide obtenu est vidé grâce à la vanne 3. Lorsque le fin de course 5 est activé, le programme atteint la fin du cycle et l'intégralité du processus reprend.

Prenons comme autre exemple celui d'un processus de contrôle du liquide dans un réservoir de stockage qui peut être vidé à l'aide d'une vanne. Avant sa mise sous

Copyright © 2015 Dunod.

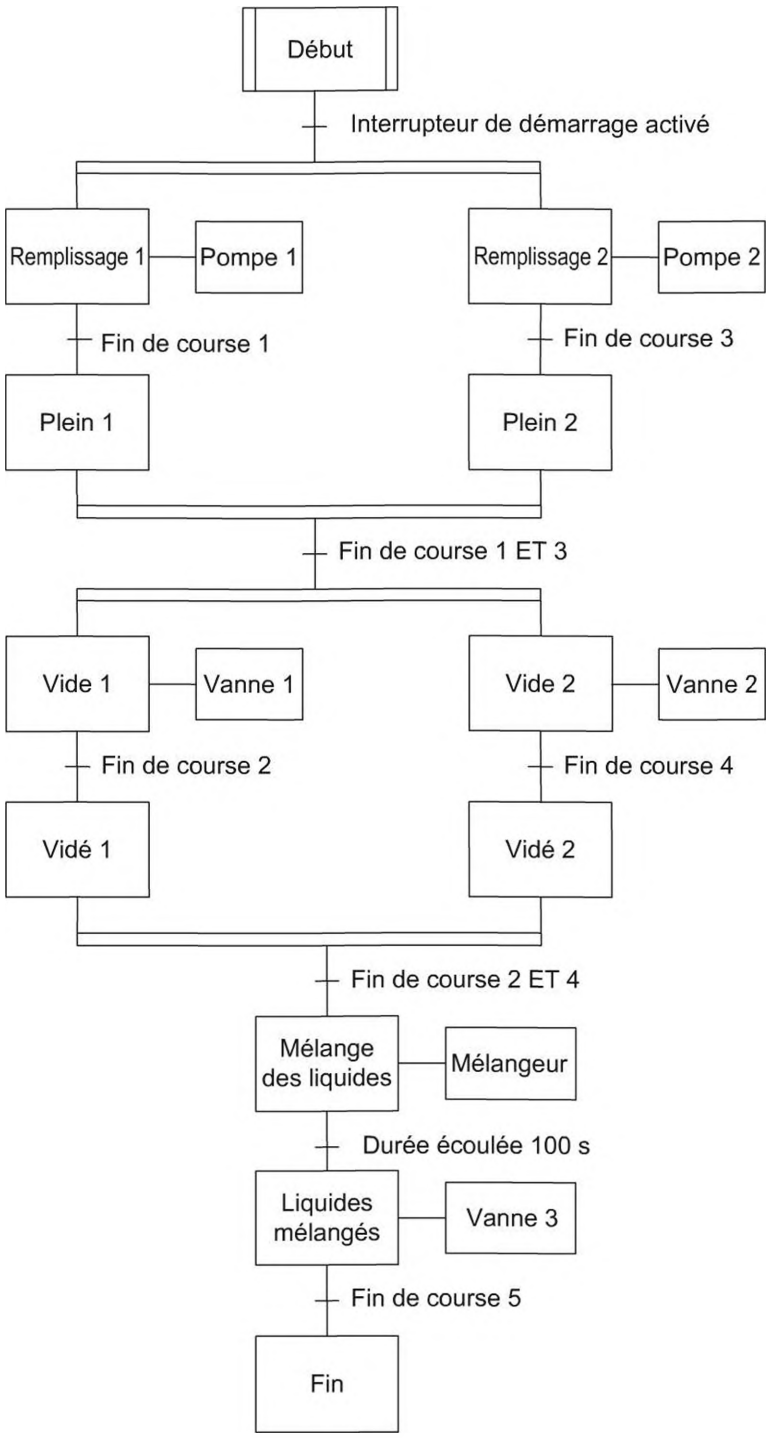


Figure 14.25 – Programme pour l'opération de mélange.

tension, la vanne placée sur le tuyau qui alimente le réservoir doit être fermée. Lorsque l'alimentation est appliquée, le réservoir peut être déjà plein ou vide. S'il est plein, il doit être vidé à l'aide de la vanne de sortie. Un réservoir vide doit être rempli grâce à l'ouverture de la vanne d'entrée. Lorsqu'il est plein, il doit être vidé. La Figure 14.26 montre le graphe de fonction séquentielle qui met en œuvre ce processus.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

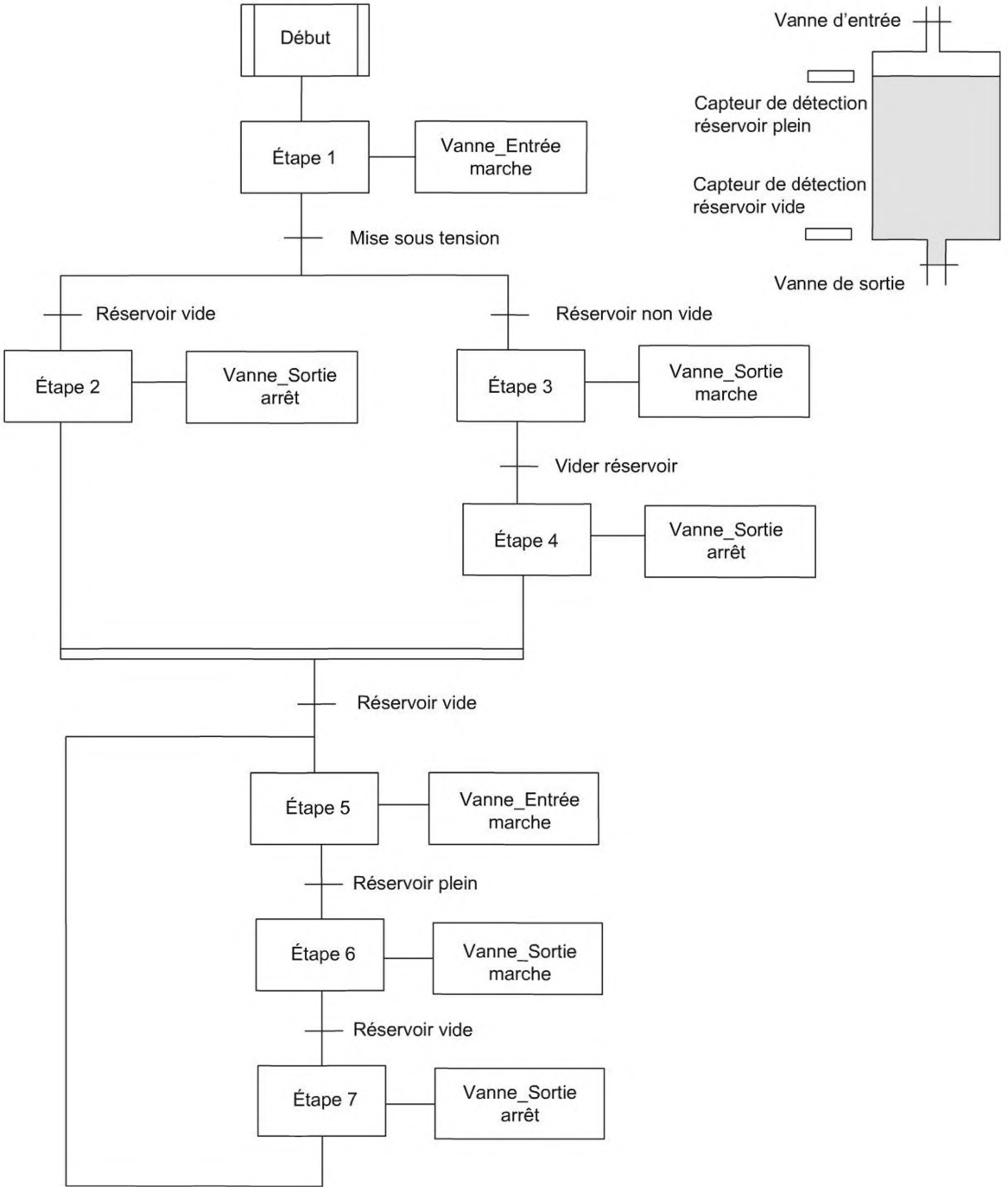


Figure 14.26 – Remplissage d'un réservoir de stockage.

14.5 Exemple de sélection : distributeur de boissons

Prenons un distributeur de boissons qui permet de choisir entre du thé et du café, avec ou sans lait, avec ou sans sucre, et qui prépare la boisson chaude sélectionnée après le paiement. La Figure 14.27 présente le diagramme de schémas fonctionnels correspondant. Le signal de sortie de la première porte OU indique si le client a choisi un thé ou un café. La sortie de la première porte ET signale si une boisson a été choisie et si une pièce a été insérée dans la machine. Elle est reliée à la

Copyright © 2015 Dunod.

dernière porte ET, qui produit une sortie pour la boisson et l'eau chaude. Le lait et le sucre sont des options qui peuvent être activées après le paiement. La Figure 14.28 donne le schéma à contacts pour ce distributeur de boissons.

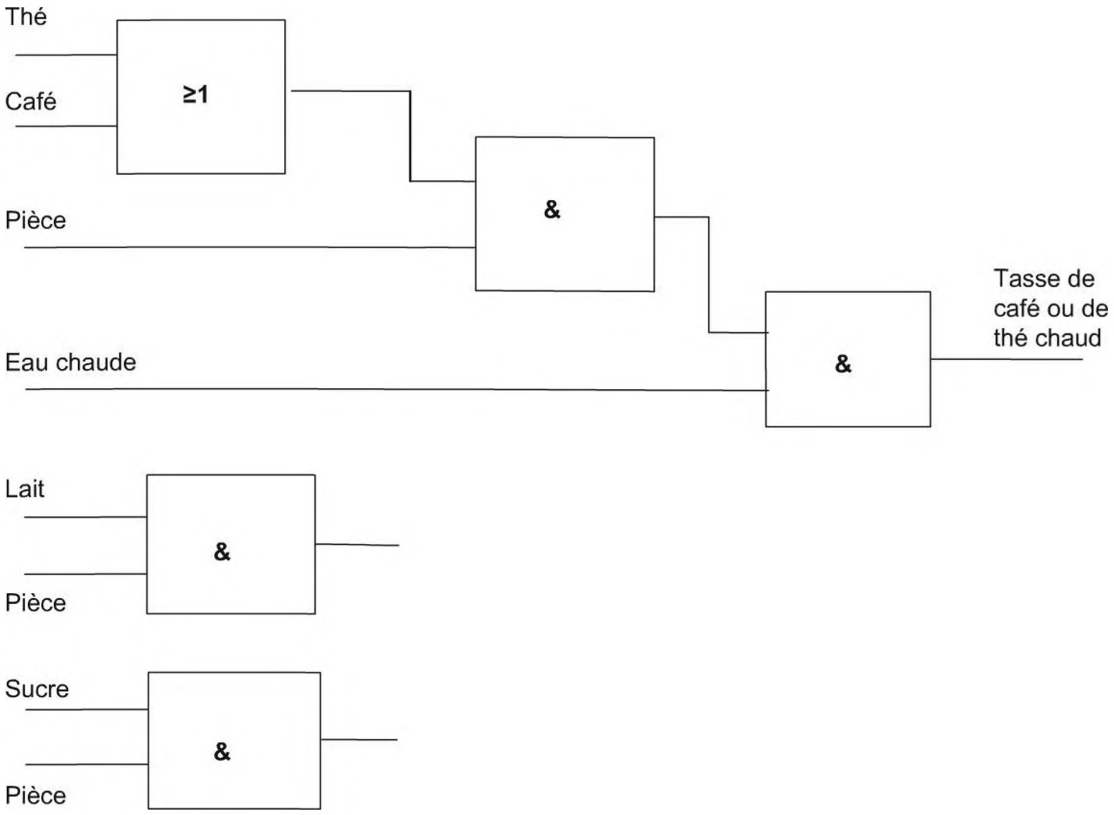


Figure 14.27 – Programme FBD pour un distributeur de boissons.

14.6 Exemple de comparaison de données : radiateur soufflant

Afin d"illustrer la comparaison de données à l'aide de l'opérateur supérieur à, prenons l'exemple simple d'un radiateur électrique qui doit commencer à chauffer lorsqu'il est allumé et, après dix secondes, démarrer un ventilateur, puis, après trente secondes, démarrer un autre ventilateur. La Figure 14.29 donne un schéma à contacts possible pour ce système, tandis que la Figure 14.30 propose le diagramme de schémas fonctionnels correspondant.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

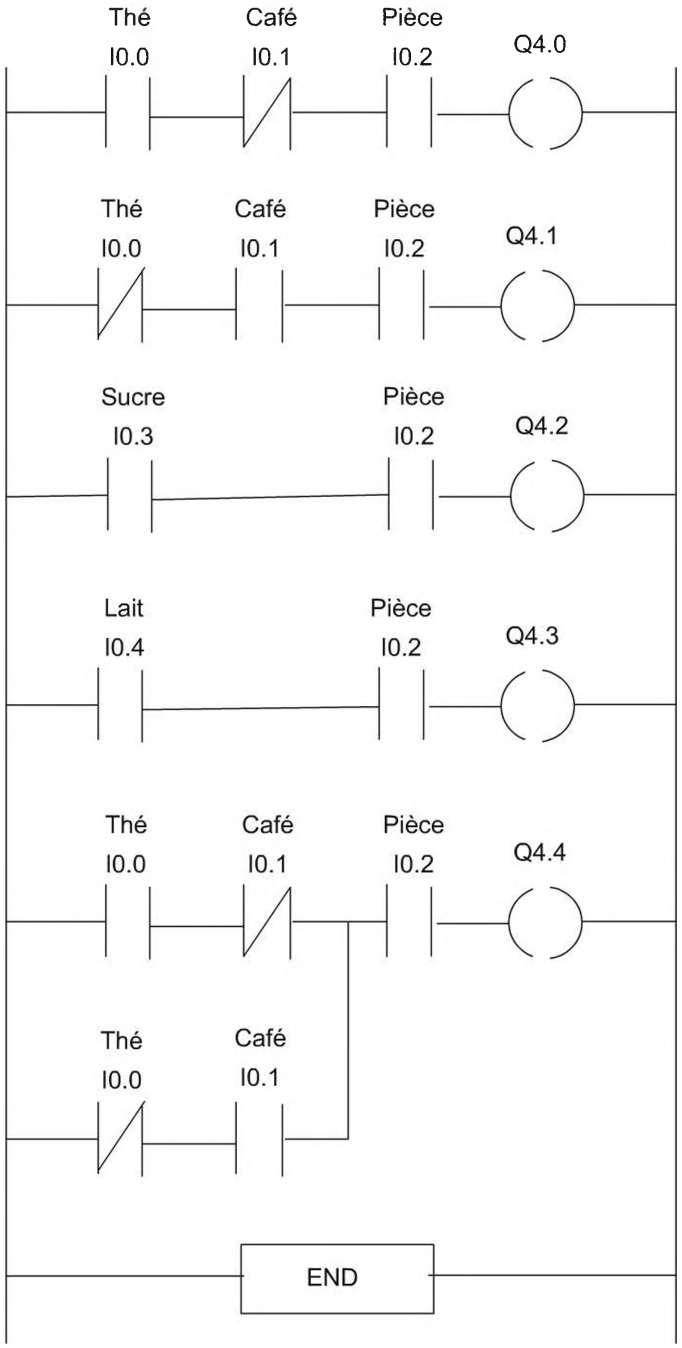


Figure 14.28 – Programme à contacts pour un distributeur de boissons.

Copyright © 2015 Dunod.

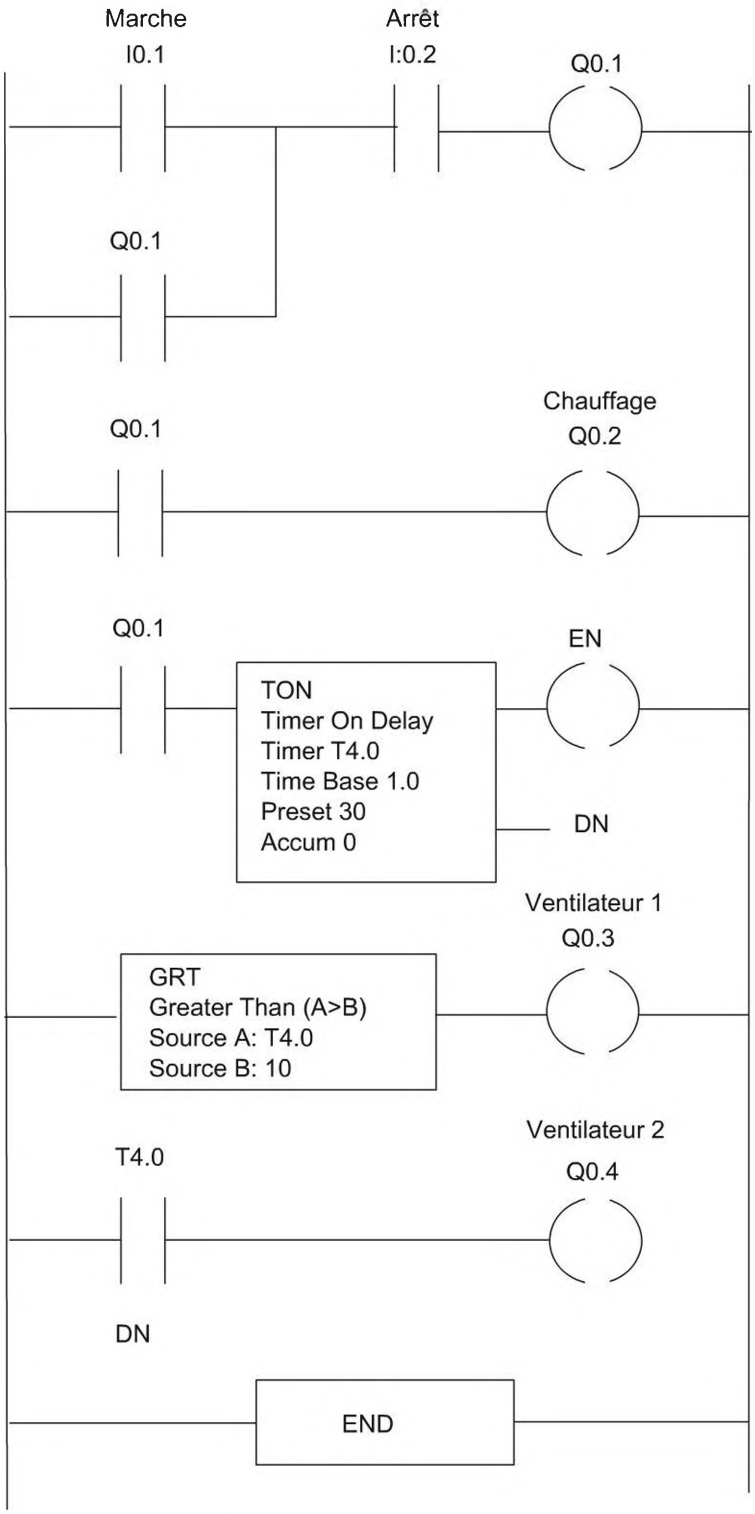


Figure 14.29 – Programme à contacts pour un radiateur soufflant.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

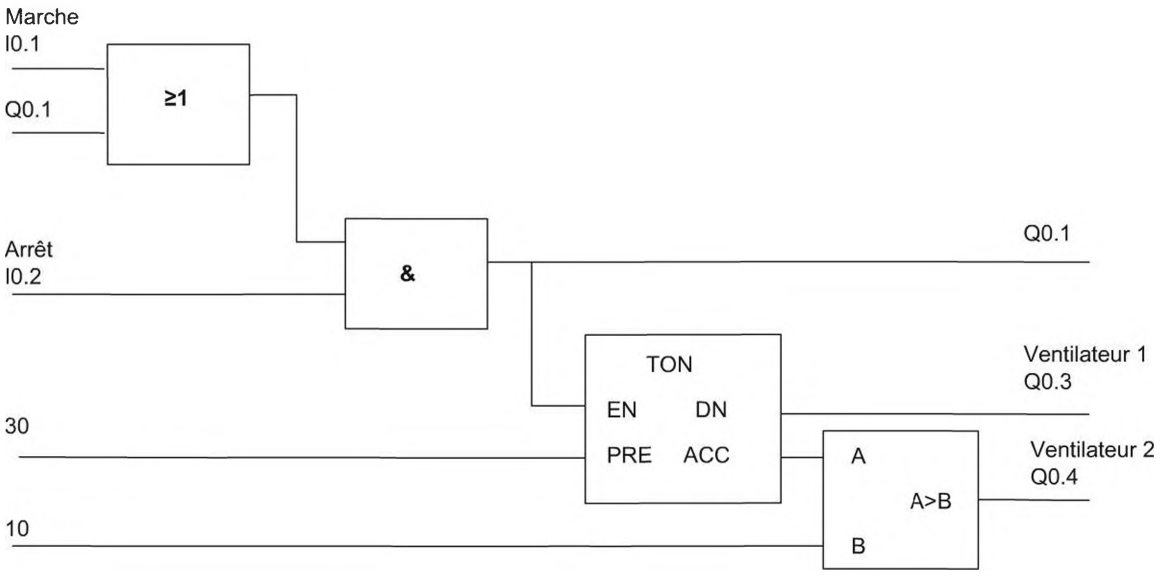


Figure 14.30 – Programme FBD pour un radiateur soufflant.

14.7 Problèmes

1. Ce problème est un élément essentiel d'un programme d'une machine à laver. Concevez un schéma à contacts pour activer une pompe pendant 100 s. Elle est ensuite désactivée et une résistance de chauffage est activée pendant 50 s. Ensuite, cette résistance est coupée et une pompe est activée pour vider l'eau.
2. Concevez un schéma à contacts qui peut être utilisé avec un vérin double effet commandé par une vanne solénoïde, c'est-à-dire un cylindre dont le piston peut être déplacé dans les deux directions au moyen du solénoïde. Le piston doit se déplacer vers la droite, rester dans cette position pendant 2 s, puis revenir vers la gauche.
3. Concevez un schéma à contacts qui permet de commander le perçage automatique de pièces conformément à la tâche simplifiée de la Figure 14.31. Le moteur du foret et la pompe d'alimentation en air des vannes pneumatiques doivent être démarrés. La pièce doit être serrée. Le foret doit ensuite être abaissé et le perçage doit aller jusqu'à la profondeur requise. Ensuite, le foret doit être retiré et la pièce libérée.
4. Quels sont les principes à respecter lors de l'installation d'un système d'arrêt d'urgence sécurisé avec un PLC ?
5. À la Figure 14.32a, les entrées issues des interrupteurs de fin de course et de l'interrupteur de mise en marche, ainsi que les sorties vers les solénoïdes des vannes, sont connectées à un API, dont le programme à contacts est donné à la Figure 14.32b. Indiquez le séquencement des vérins.
6. À la Figure 14.33a, les entrées issues des interrupteurs de fin de course et de l'interrupteur de mise en marche, ainsi que les sorties vers les solénoïdes des vannes, sont connectées à un API, dont le programme à contacts est donné à la Figure 14.33b. Indiquez le séquencement des vérins.

Copyright © 2015 Dunod.

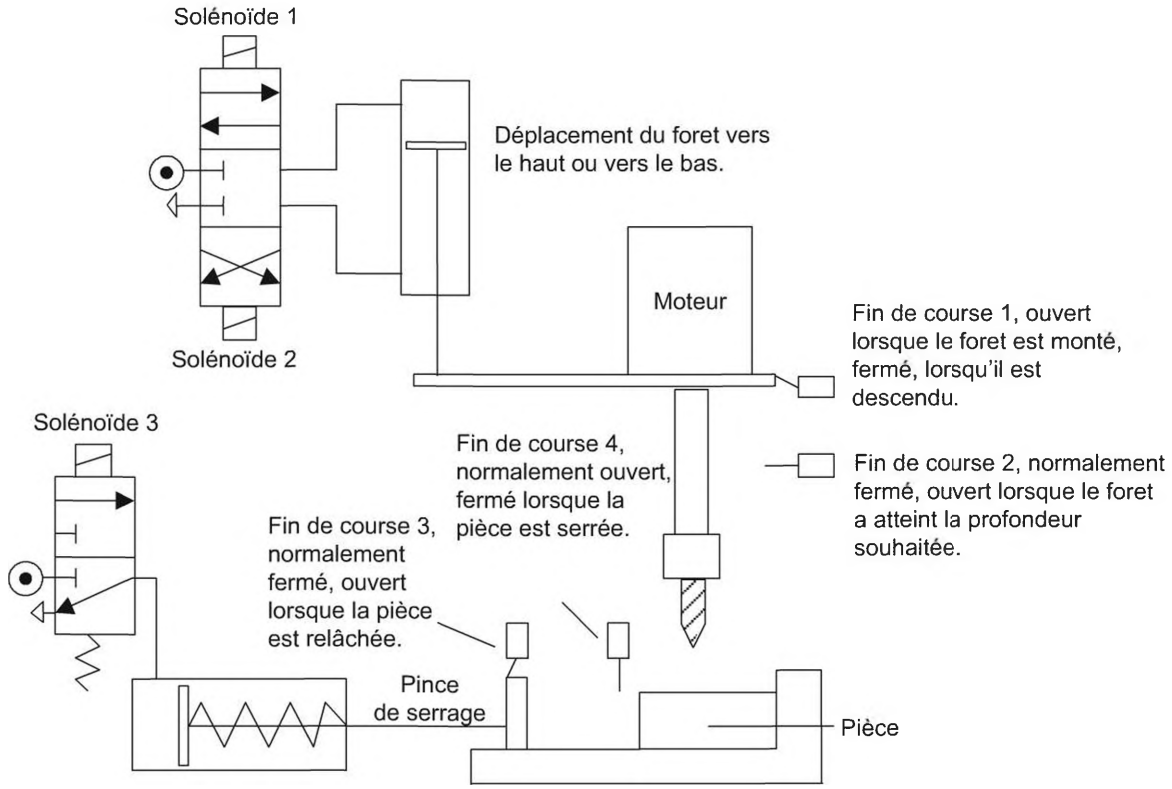


Figure 14.31 – Diagramme pour le problème 3.

7. La Figure 14.34 montre un schéma à contacts qui met en œuvre le compteur C460, les entrées X400 et X401, les relais internes M100 et M101, et la sortie Y430. X400 est l'interrupteur de mise en marche. Expliquez comment activer la sortie Y430.
8. Concevez un schéma à contacts qui démarre deux moteurs lorsque l'interrupteur de mise en marche est actionné, puis coupe un moteur après 200 s et l'autre moteur après 100 s supplémentaires. Lorsque les deux moteurs ont été arrêtés, un troisième moteur est démarré pendant 50 s. Le cycle reprend ensuite, à moins qu'un interrupteur d'arrêt n'ait été activé.
9. Concevez un schéma à contacts qui démarre un moteur lorsque l'interrupteur de mise en marche est momentanément actionné. Le moteur doit rester en fonctionnement pendant 50 s. Ensuite, un deuxième moteur doit être activé pendant 50 s. Un troisième moteur doit être activé 10 s avant que le deuxième moteur ne s'arrête et doit rester en fonctionnement pendant 50 s. Le cycle reprend ensuite, à moins qu'un interrupteur d'arrêt n'ait été activé.
10. Rédigez les spécifications d'un système de commande pour un ascenseur transportant des personnes. Il doit être actionné depuis le rez-de-chaussée et le premier étage du bâtiment. Proposez un schéma à contacts pour mettre en œuvre vos spécifications.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

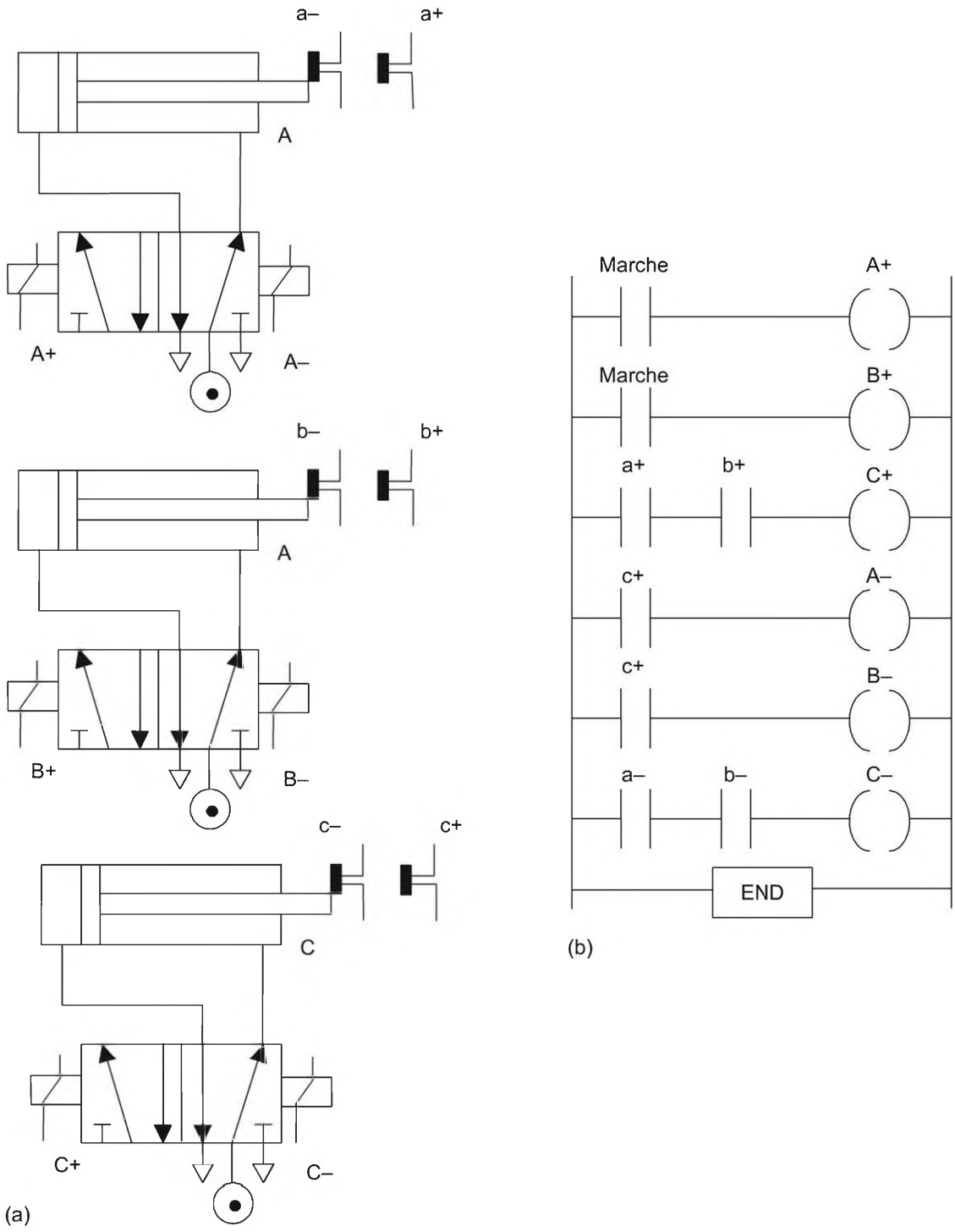


Figure 14.32 – Diagramme pour le problème 5.

Copyright © 2015 Dunod.

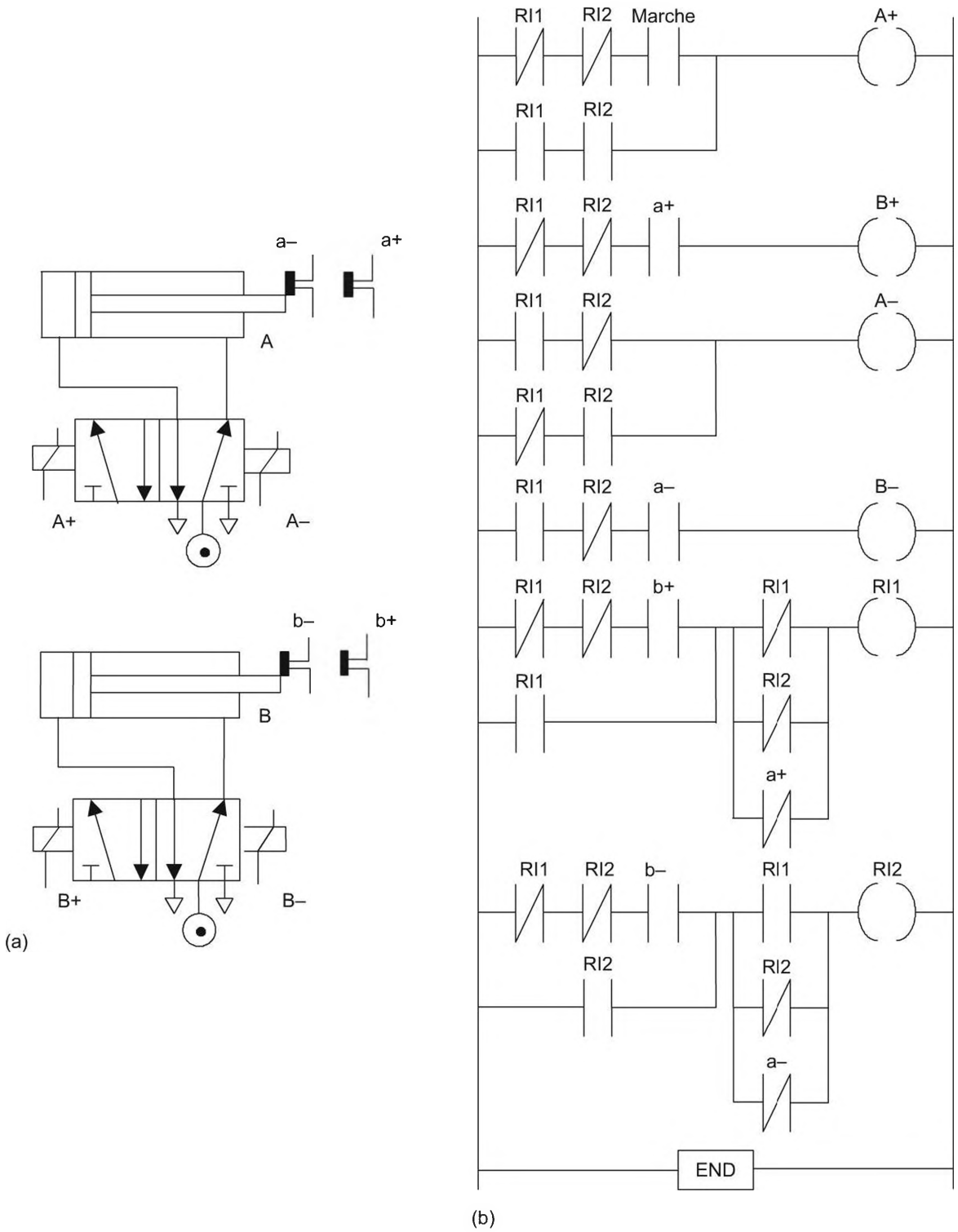


Figure 14.33 – Diagramme pour le problème 6.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

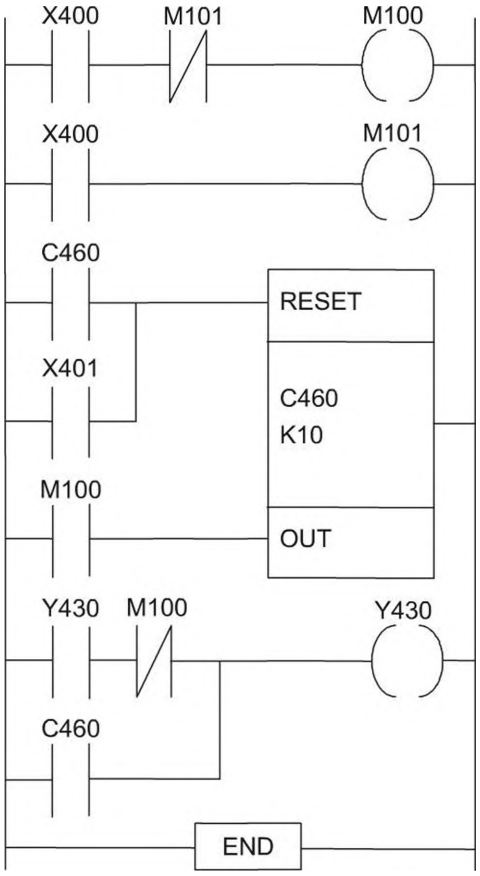


Figure 14.34 – Diagramme pour le problème 7.

14.8 Recherches complémentaires

1. Recherchez un API qui peut être utilisé pour (a) le système de chauffage central illustré à la Figure 14.7 et pour (b) le système d'emballage des bouteilles présenté aux Figures 14.22 et 14.23.
2. Recherchez les capteurs adaptés (a) au système de bande transporteuse décrit à la Figure 14.20 et (b) au système d'emballage des bouteilles décrit à la Figure 14.23.

Copyright © 2015 Dunod.

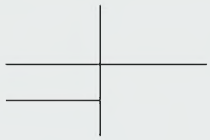
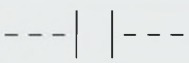
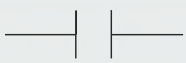
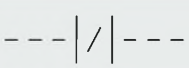
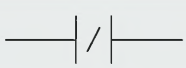
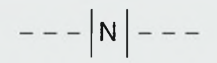
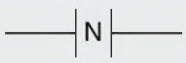
Annexes

Copyright © 2015 Dunod.

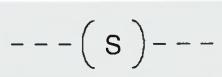
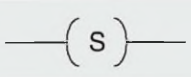
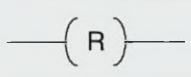
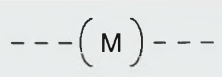
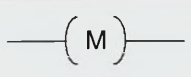
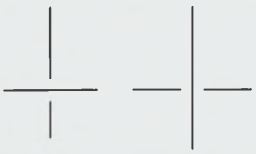
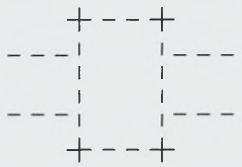
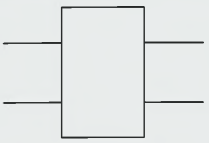
Copyright © 2015 Dunod.

ANNEXE 1 • SYMBOLES

A1.1 Programmes à contacts

	Semi-graphique	Graphique
Une ligne horizontale au travers de laquelle le courant peut circuler.		
Interconnexion de lignes d'alimentation horizontales et verticales.		
Connexion d'une ligne horizontale à la ligne d'alimentation de gauche.		
Connexion d'une ligne horizontale à la ligne d'alimentation de droite.		
Contact normalement ouvert.		
Contact normalement fermé.		
Contact à détection de front montant. Le courant passe lorsque la variable associée change du niveau 0 au niveau 1.		
Contact à détection de front descendant. Le courant passe lorsque la variable associée change du niveau 1 au niveau 0.		
Bobine de sortie : si la bobine est alimentée, alors, son état est à vrai.		

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

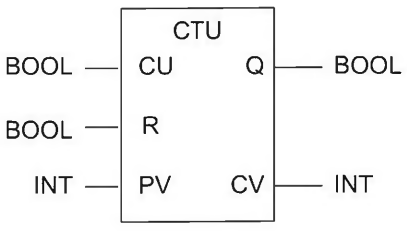
	Semi-graphique	Graphique
Bobine d'enclenchement.		
Bobine de déclenchement.		
Bobine mémoire rémanente. L'état de la variable associée est conservé lorsque l'alimentation de l'API est coupée.		
Lignes horizontales et verticales.		
Interconnexion de flux de signaux horizontaux et verticaux.		
Croisement de flux de signaux horizontaux et verticaux.		
Blocs avec des connexions.		
Connecteurs.	POIDS_MOY POIDS_MOY	POIDS_MOY POIDS_MOY

A1.2 Blocs fonctionnels

A1.2.1 Blocs les plus employés

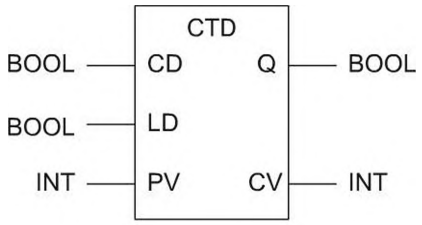
BOOL est un signal booléen, INT est un entier, REAL est un nombre à virgule flottante, ANY est toute forme de signal.

Un compteur progressif compte le nombre de fronts montants sur l'entrée CU. PV définit la valeur maximale du compteur. Chaque nouveau front montant sur CU incrémente CV de 1. La sortie Q est active après que la valeur maximale a été atteinte. R est la remise à zéro.

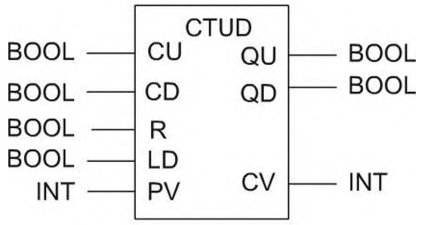


Copyright © 2015 Dunod.

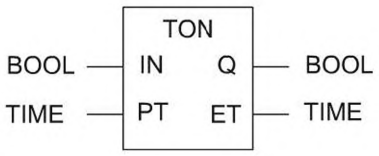
Un compteur dégressif compte le nombre de fronts montants sur l'entrée CU. PV définit la valeur de départ du compteur. Chaque nouveau front montant sur CU décrémente CV de 1. La sortie Q est active après que le compteur a atteint zéro.



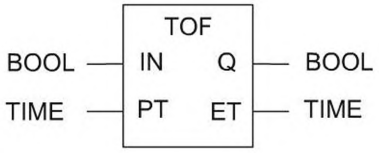
Un compteur progressif-dégressif peut servir à compter une entrée et à décompter l'autre.



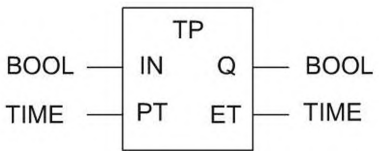
Temporisateur à l'enclenchement. Lorsque l'entrée IN passe à vrai, le temps écoulé ET commence à augmenter et, lorsqu'il atteint la durée fixée par l'entrée PT, la sortie Q passe à vrai.



Temporisateur au déclenchement. Lorsque l'entrée IN passe à vrai, la sortie Q suit et reste à vrai pendant la durée fixée, après quoi elle passe à faux.



Temporisateur à impulsion. Lorsque l'entrée IN passe à vrai, la sortie Q suit et reste à vrai pendant la durée de l'impulsion indiquée par l'entrée PT.



A1.3 Portes logiques

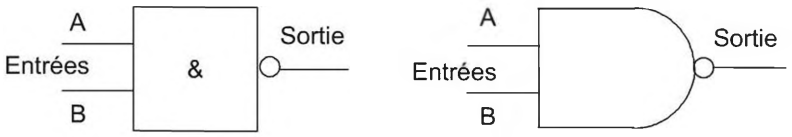
Porte ET

Porte OU

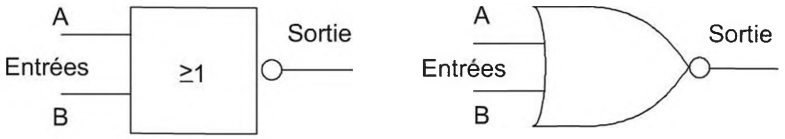
Porte NON

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

Porte NON-ET



Porte NON-OU



Porte OU exclusif

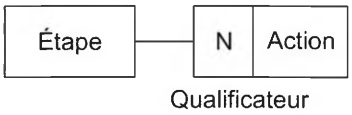
A1.4 Graphe de fonction séquentielle

Étape initiale. Elle représente l'étape qui sera activée lorsque l'API est mis sous tension.

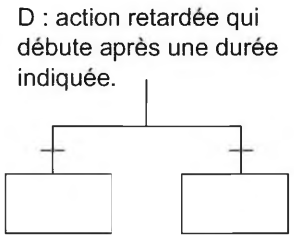
Condition de transition. Chaque transition doit avoir une condition. Si la transition doit toujours se produire, sa condition doit être TRUE.

Étape d'un programme.

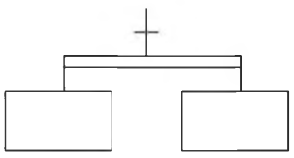
À chaque étape est associée une action. Une action décrit le comportement qui se produit lorsque l'étape est activée. Chaque action peut avoir un qualificateur : N indique que l'action est exécutée lorsque l'étape est active. Si aucun qualificateur n'est précisé, N est pris par défaut.



Divergence sélective.

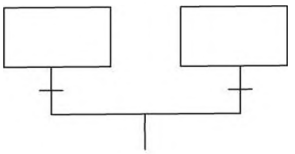


Divergence parallèle lorsque la transition se produit.

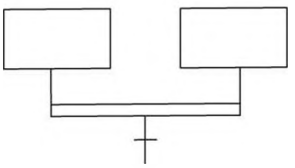


Copyright © 2015 Dunod.

Convergence lorsque les deux transitions se produisent.



Convergence simultanée.



A1.5 Liste d'instructions (symboles de la CEI 61131-3)

LD	Commencer une ligne par un contact ouvert.
LDN	Commencer une ligne par un contact fermé.
ST	Une sortie.
S	Mise à un.
R	Mise à zéro.
AND	ET booléen.
AND N	NON-ET booléen.
OR	OU booléen.
ORN	NON-OU booléen.
XOR	Ou exclusif booléen.
NOT	NON booléen.
ADD	Addition.
SUB	Soustraction.
MUL	Multiplication.
DIV	Division.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

A1.6 Texte structuré

X := Y	Y représente une expression qui produit une nouvelle valeur pour la variable X.

A1.6.1 Opérateurs

(...)	Expression parenthésée.
Function(...)	Liste des paramètres d'une fonction.
**	Élévation à une puissance.
–, NOT	Négation, NON booléen.
*, /, MOD	Multiplication, division, modulo.
+, –	Addition, soustraction.
<, >, <=, >=	Inférieur à, supérieur à, inférieur ou égal à, supérieur ou égal à.
=, <>	Égalité, inégalité.
AND, &	ET booléen.
XOR	Ou exclusif booléen.
OR	OU booléen.

A1.6.2 Instructions conditionnelles et d'itération

IF ... THEN ... ELSE	Elle est utilisée pour exécuter des instructions sélectionnées lorsque certaines conditions se réalisent.
FOR ... DO	Elle permet de répéter une suite d'instructions en fonction de la valeur d'une variable d'itération entière.
WHILE ... DO	Elle permet d'exécuter une ou plusieurs instructions tant qu'une expression booléenne reste vraie.
REPEAT ... UNTIL	Elle permet d'exécuter une ou plusieurs instructions et de la répéter tant qu'une expression booléenne reste vraie.

Copyright © 2015 Dunod.

ANNEXE 2 • RÉPONSES

A2.1 Chapitre 1

1. D.
2. A.
3. C.
4. A.
5. A.
6. C.
7. Voir Figure 1.4.
8. Voir Section 1.3.1.
9. $2 \times 1\,024$.
10. Voir Section 1.3.2 pour une explication des concepts de fourniture et d'absorption de courant, ainsi que Section 1.3.1 et Figure 1.15 pour les sorties à relais et à transistor.

A2.2 Chapitre 2

1. A.
2. A.
3. B.
4. D.
5. C.
6. A.
7. A.
8. A.
9. B.
10. B.
11. C.
12. B.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

13. C.
14. C.
15. Voir (a) Figure 2.6, (b) Section 2.1.4, (c) Section 2.1.3, (d) Section 2.1.8.
16. Voir Section 2.2.3.
17. Voir Section 2.2.4.
18. Par exemple, (a) système photoélectrique transmissif, (b) capteur de proximité capacitif, (c) interrupteur de fin de course mécanique, (d) capteur de proximité inductif.
19. Voir Section 2.2.4. Comportement des circuits *RL*.
20. Moteur pas à pas avec un pas de 5°.
21. (a) Système photoélectrique transmissif, (b) vérin actionné par une vanne de commande directionnelle.

A2.3 Chapitre 3

1. (a) 3, (b) 63, (c) 13.
2. (a) 110 0100, (b) 1001 0010, (c) 1111 1111.
3. (a) 159, (b) 3411, (c) 1660.
4. (a) E, (b) 51, (c) A02.
5. (a) 1110, (b) 11101, (c) 1010 0110 0101.
6. (a) 250, (b) 12, (c) 1376.
7. (a) 24, (b) 411, (c) 620.
8. (a) 010 111 000, (b) 001 000 010, (c) 110 111 011.
9. (a) 0010 0000, (b) 0011 0101, (c) 1001 0010.
10. (a) 1111 1111, (b) 1101 1101, (c) 1000 0011.
11. (a) −16, (b) −55, (c) −40.
12. (a) 0,110010 ∞ 2^{-3}, (b) 0,1100 ∞ 2^{-4}, (c) 0,1000 0100 ∞ 2^{4}.
13. Voir Sections 3.7 et 3.8.
14. (a) 1 ET 1, (b) 1 OU 1, (c) 1 ET NON 1.
15. D 1, CLK 1.
16. Il s'agit d'un verrou D, comme au Tableau 3.4.

A2.4 Chapitre 4

1. C.
2. D.
3. C.
4. B.

Copyright © 2015 Dunod.

5. B.
6. A.
7. B.
8. A.
9. C.
10. C.
11. A.
12. C.
13. A.
14. D.
15. D.
16. (a) 0, (b) 1.
17. Pour détecter la corruption d'un message.
18. Voir Section 4.5.
19. Entrée 1 kΩ, sortie 100 kΩ.
20. Voir Section 4.4.
21. Voir Sections (a) 4.5.5, (b) 4.5.2 et 4.5.3.
22. Voir Section 4.4.

A2.5 Chapitre 5

1. A.
2. D.
3. B.
4. B.
5. B.
6. B.
7. B.
8. D.
9. C.
10. A.
11. A.
12. B.
13. D.
14. C.
15. B.
16. A.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

17. C.
18. C.
19. D.
20. C.
21. A.
22. C.
23. A.
24. D.
25. Voir (a) Figure 5.8, (b) Figure 5.10, (c) Figure 5.19, (d) Figure 5.10, (e) Figure 5.11, (f) Figure 5.5a, (g) porte ET comme à la Figure 5.8.
26. (a) Porte OU comme à la Figure 5.28, (b) comme à la Figure 5.30, (c) porte OU comme à la Figure 5.28.
27. (a) Comme à la Figure 5.33, (b) voir Figure A2.1a, (c) voir Figure A2.1b.

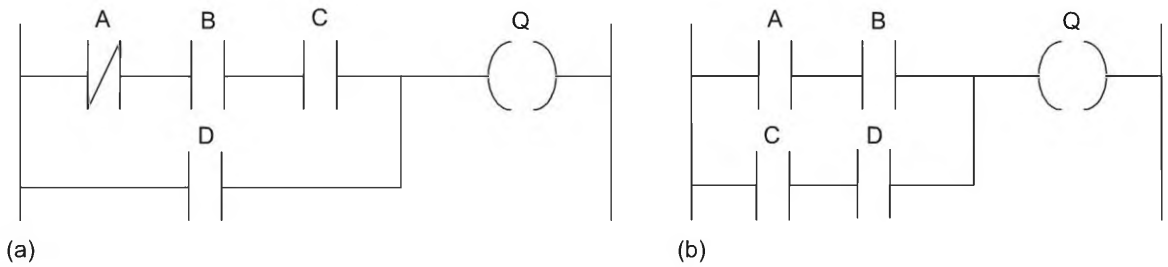


Figure A2.1 – Chapitre 5, problème 27.

28. (a) $Q = A + B$, (b) $Q = A \cdot B \cdot \overline{C}$, (c) $Q = A \cdot \overline{B}$.

A2.6 Chapitre 6

1. C.
2. A.
3. B.
4. D.
5. A.
6. C.
7. A.
8. B.
9. D.
10. A.
11. A.
12. C.

Copyright © 2015 Dunod.

13. D.
14. B.
15. A.
16. B.
17. A.
18. C.
19. A.
20. D.
21. C.
22. A.
23. C.
24. B.
25. Voir Figure A2.2.

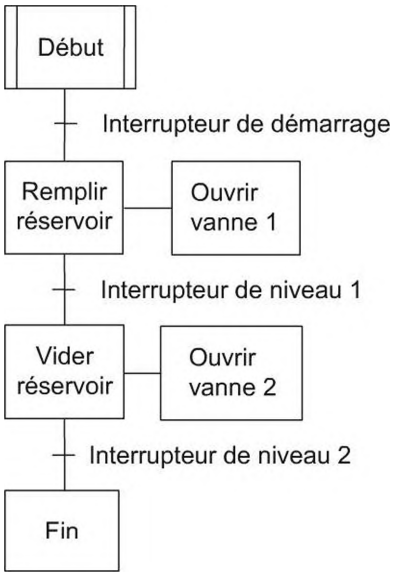


Figure A2.2 – Chapitre 6, problème 25.

26.
```
WHILE NOT (Interrupteur_Niveau1 AND Vanne_Vidange)
    Vanne1 := 1;
END_WHILE
```

27.
```
CASE Parametre_Temperature OF
    Interrupteur_Four := 1;
    1 : temp := 40;
    2 : temp := 50;
    3 : temp := 60; Ventilo1 := 1;
    4 : temp := 70; Ventilo2 := 1;
ELSE
    Interrupteur_Four := 0;
END_Case
```

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

A2.7 Chapitre 7

1. D.
2. B.
3. C.
4. A.
5. C.
6. C.
7. C.
8. B.
9. A.
10. A.
11. C.
12. D.
13. B.
14. B.
15. B.
16. C.
17. A.
18. A.
19. A.
20. B.
21. A.
22. A.
23. A.
24. Voir (a) Figure 7.8, (b) Figure 7.9 ou 7.10, (c) Figure 7.22.

A2.8 Chapitre 8

1. C
2. B
3. A
4. B
5. B
6. A
7. Appel de sous-programmes et retour. Les sous-programmes sont des blocs de code dans un programme ; voir Section 8.2.

Copyright © 2015 Dunod.

A2.9 Chapitre 9

1. C.
2. A.
3. D.
4. D.
5. D.
6. D.
7. C.
8. C.
9. B.
10. C.
11. A.
12. A.
13. A.
14. B.
15. D.
16. B.
17. A.
18. D.
19. C.
20. D.
21. B.
22. Voir (a) Figure 9.4, (b) Figure 9.10, (c) Figure 9.12, (d) Figure A2.3.

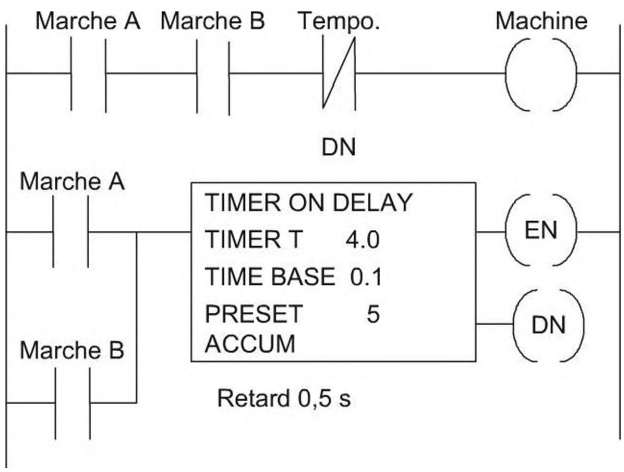


Figure A2.3 – Chapitre 9, problème 22d.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

A2.10 Chapitre 10

1. C.
2. A.
3. C.
4. B.
5. B.
6. B.
7. B.
8. D.
9. C.
10. A.
11. A.
12. B.
13. B.
14. C.
15. B.
16. D.
17. C.
18. C.
19. A.
20. Voir (a) Figure 10.4, (b) Figure 10.10, (c) Figure A2.4a, (d) Figure A2.4b, (e) semblable à la Figure 10.10 ou 10.11.

A2.11 Chapitre 11

1. D.
2. C.
3. C.
4. D.
5. A.
6. A.
7. C.
8. C.
9. D.
10. (a) Comme à la Figure 11.1/11.2, avec une entrée constante sur E1/X400, pour entrer un 1 à chaque décalage, (b) comme à la Figure 11.3 mais en remplaçant l'article défecteux par un crochet avec un article et l'article valide par un crochet sans article.

Copyright © 2015 Dunod.

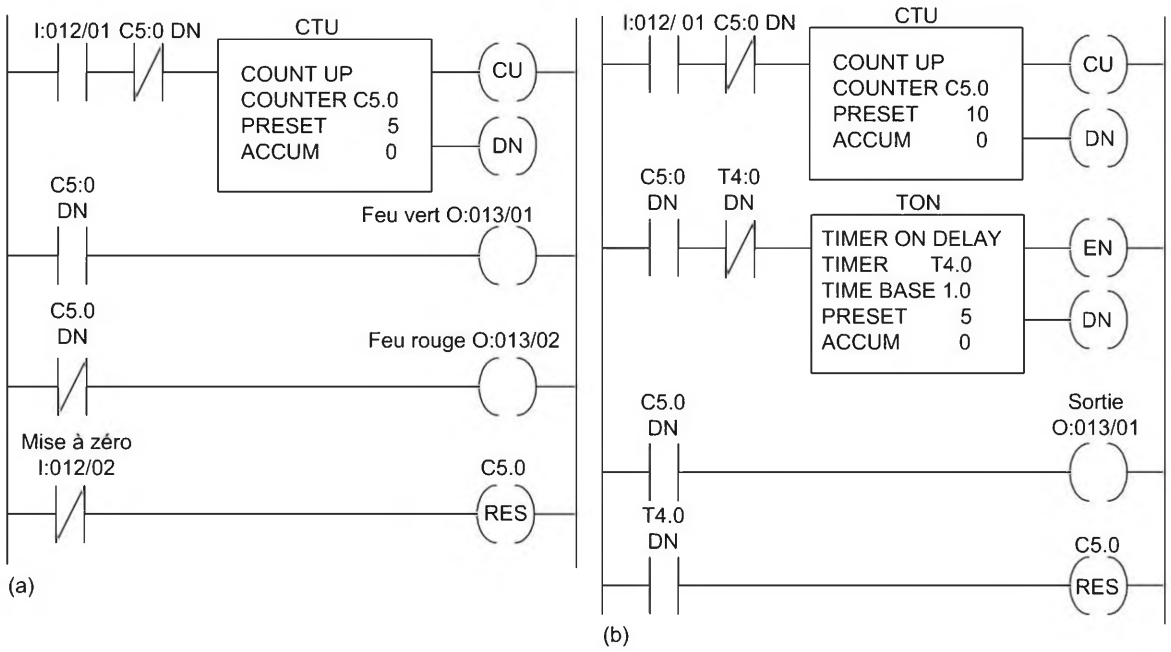


Figure A2.4 – Chapitre 10, (a) problème 20c, (b) problème 20d.

A2.12 Chapitre 12

1. C.
2. C.
3. B.
4. B.
5. A.
6. A.
7. B.
8. B.
9. C.
10. Semblable à (a) la Figure 12.5, (b) la Figure 12.6, (c) voir Figure A2.5a, (d) voir Figure A2.5b.
11. Voir Section 12.4.
12. Voir Section 12.4.

A2.13 Chapitre 13

1. B.
2. B.
3. C.
4. B.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

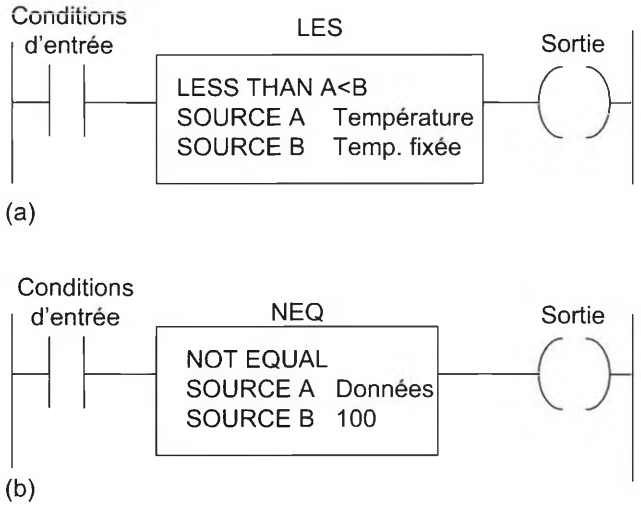


Figure A2.5 – Chapitre 12, (a) problème 10c, (b) problème 10d..

5. C.
6. A.
7. A.
8. Voir Section 13.3.1.
9. Panne d'alimentation, alimentation coupée, alimentation déclenchée.
10. Défaut du câblage, défaut du dispositif.
11. Voir Figure A2.6.
12. Voir Figure A2.7.

A2.14 Chapitre 14

1. Voir Figure A2.8.
2. Voir Figure A2.9.
3. Voir Figure A2.10 pour une réponse de base.
4. Bouton d'arrêt d'urgence câblé, indépendant du logiciel.
5. A+ et B+, C+, A– et B–, C–.
6. A+, B+, A-, B-, A+, A-.
7. M100 et M101 activés. Comptage de dix impulsions sur X401. Puis sortie.
8. Voir Figure A2.11.
9. Voir Figure A2.12.
10. Voici une spécification simple : l'ascenseur peut se déplacer uniquement lorsque les deux portes d'accès sont fermées et que la porte de l'ascenseur est fermée. L'ascenseur monte du rez-de-chaussée au premier étage lorsqu'il est appelé depuis le premier étage et va au rez-de-chaussée lorsqu'il est appelé depuis le rez-de-chaussée. À chaque niveau, les témoins lumineux indiquent

Copyright © 2015 Dunod.

l'étage auquel se trouve l'ascenseur. La Figure A2.13 propose un programme. Vous pouvez l'améliorer en ajoutant un temporisateur qui déclenche une alarme si le déplacement de l'ascenseur entre deux niveaux prend trop de temps.

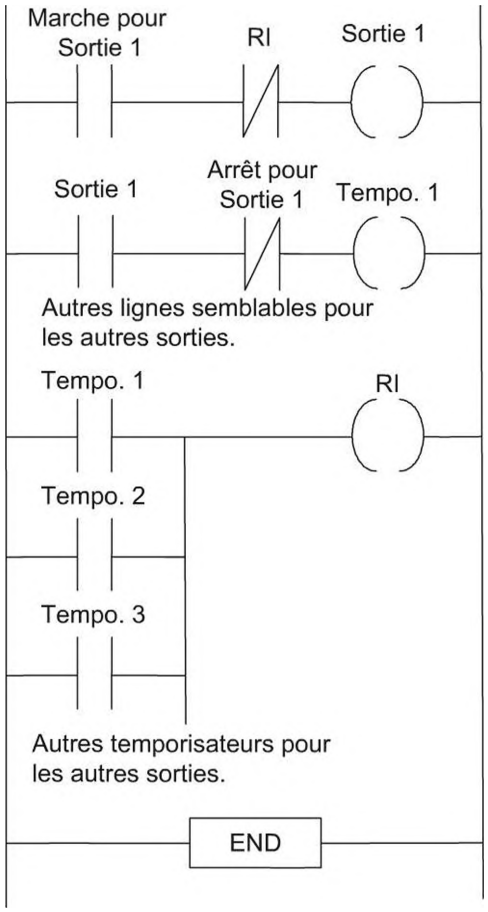


Figure A2.6 – Chapitre 13, problème 11.

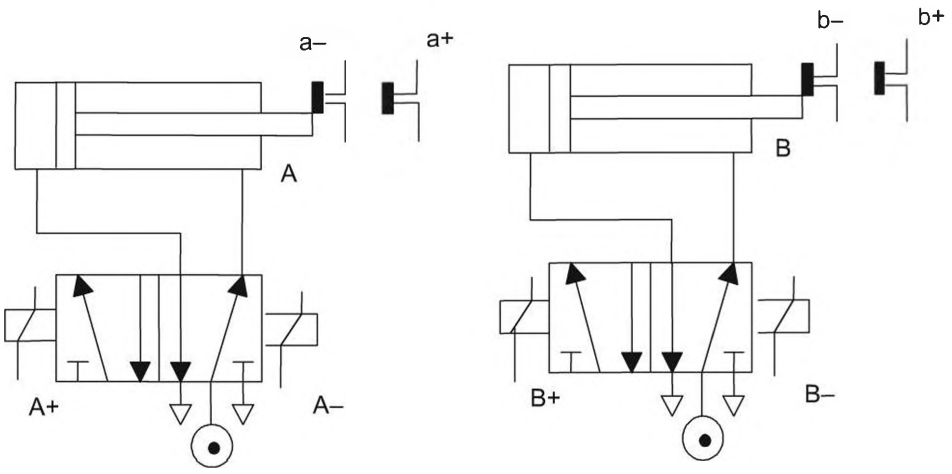
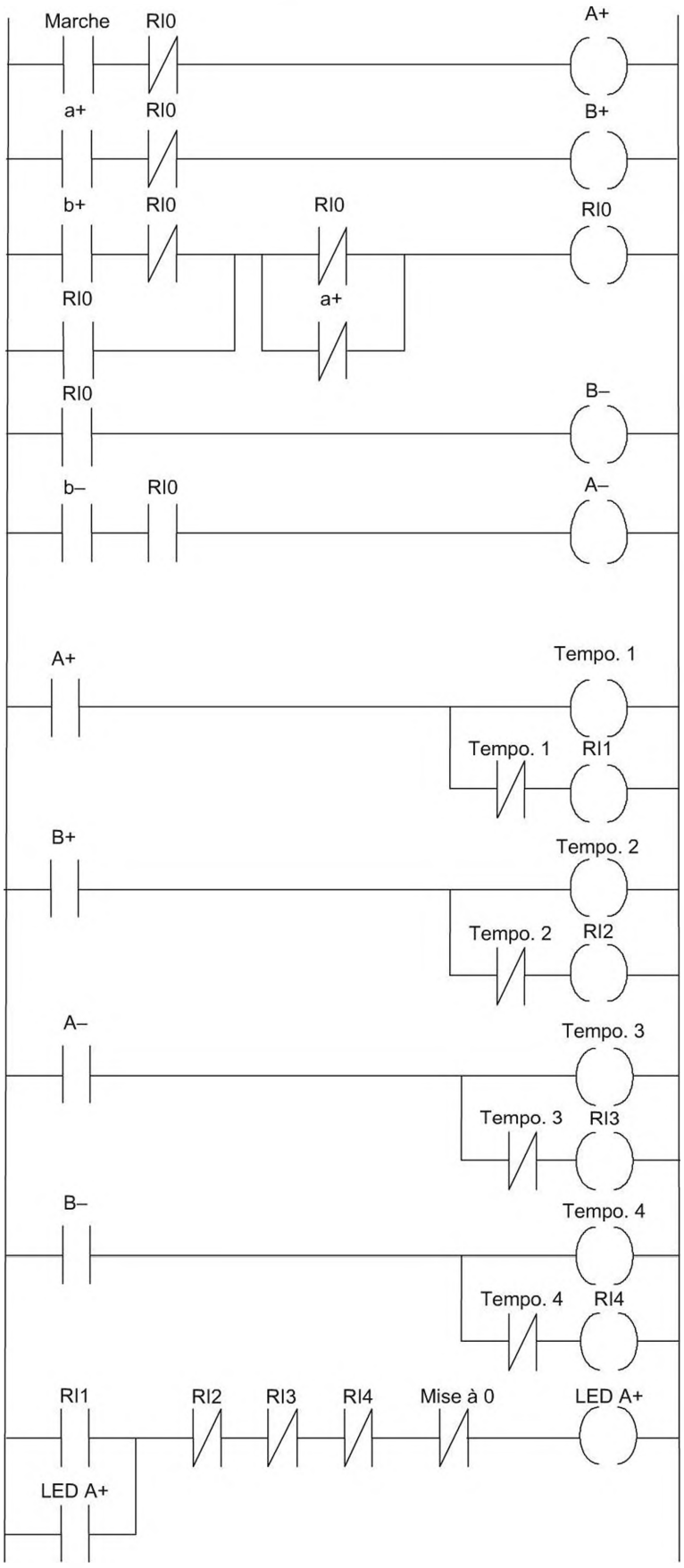


Figure A2.7 – Chapitre 13, problème 12.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.



Lorsque l'interrupteur de mise en marche est fermé, A+ est excité.

Lorsque a+ est activé, B+ est excité.

Lorsque b+ est activé, RI0 est excité et ses contacts se ferment dans cette ligne et les lignes suivantes, mais s'ouvrent dans les lignes précédentes. A+ et B+ ne sont plus excités.

B– est activé.

Lorsque b– est activé, A– est excité.

La partie ci-dessus du programme produit la séquence A+, B+, B–, A–, la partie suivante, les diagnostics.

La sortie A+ produit une impulsion brève sur RI1 suite à la mise à un du temporisateur.

La sortie B+ produit une impulsion brève sur RI2 suite à la mise à un du temporisateur.

La sortie A– produit une impulsion brève sur RI4 suite à la mise à un du temporisateur.

La sortie B– produit une impulsion brève sur RI5 suite à la mise à un du temporisateur.

Lorsque la sortie A+ se produit, RI1 se ferme et est verrouillé. LED A+ est allumée. LED A+ n'est pas allumée si RI1 n'est pas fermé.

Figure A2.7 – Chapitre 13, problème 12 *(suite)*.

Copyright © 2015 Dunod.

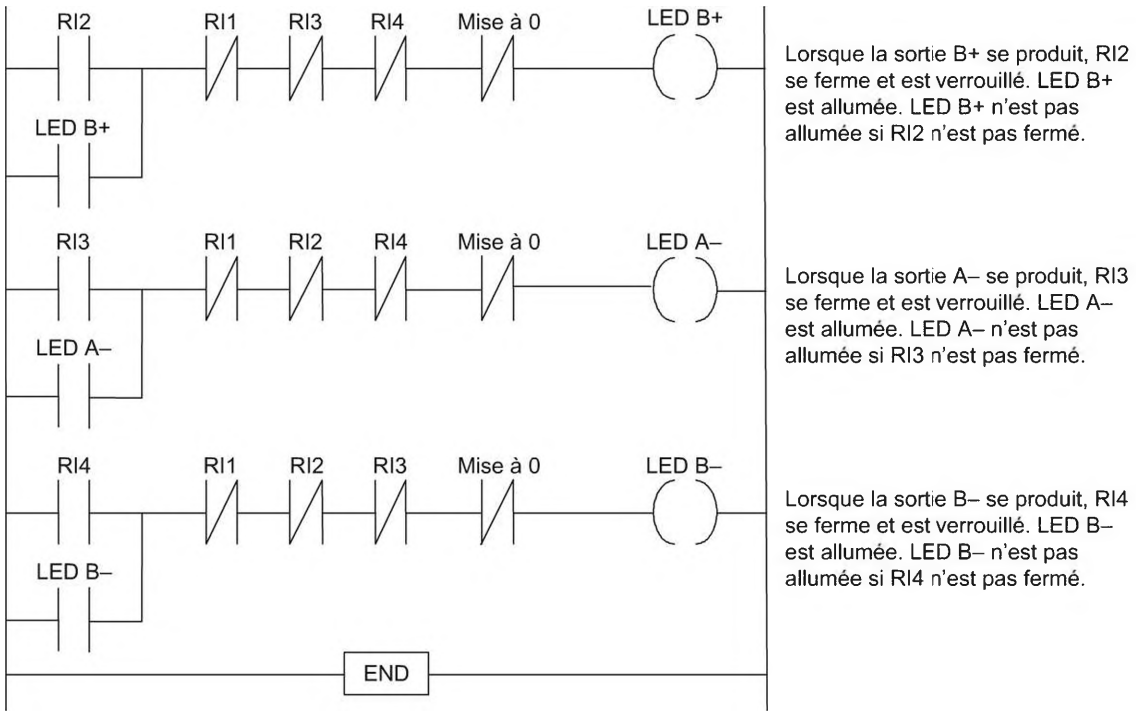


Figure A2.7 – Chapitre 13, problème 12 *(suite)*.

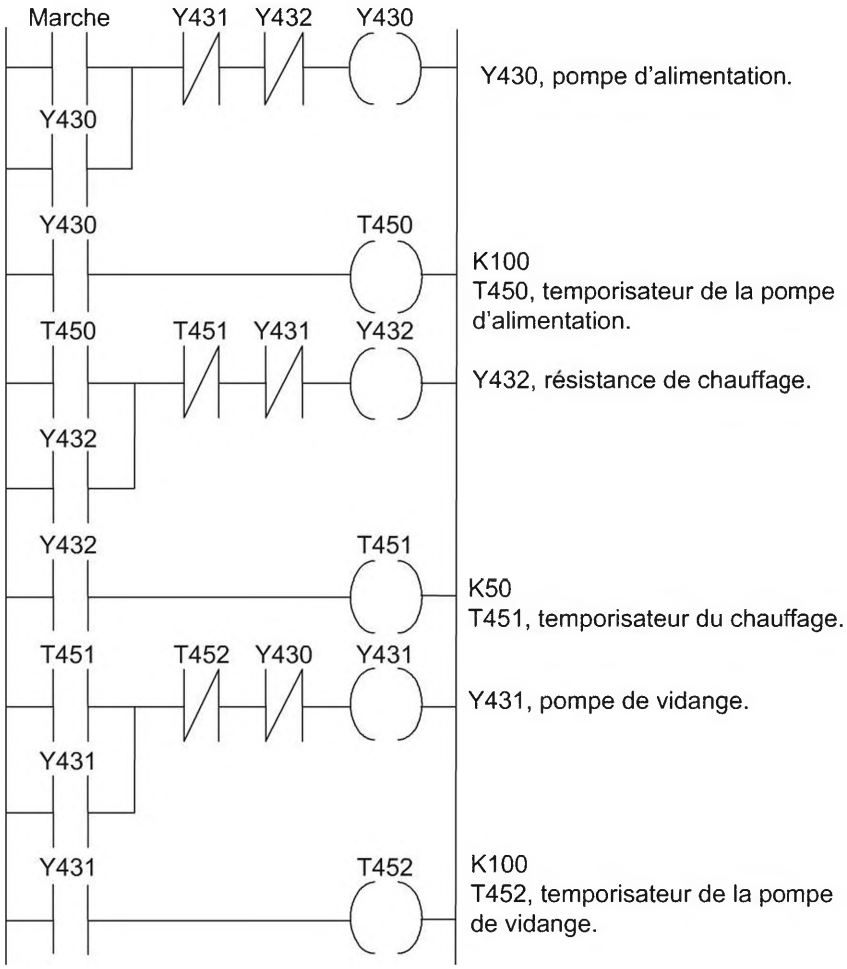


Figure A2.8 – Chapitre 14, problème 1.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

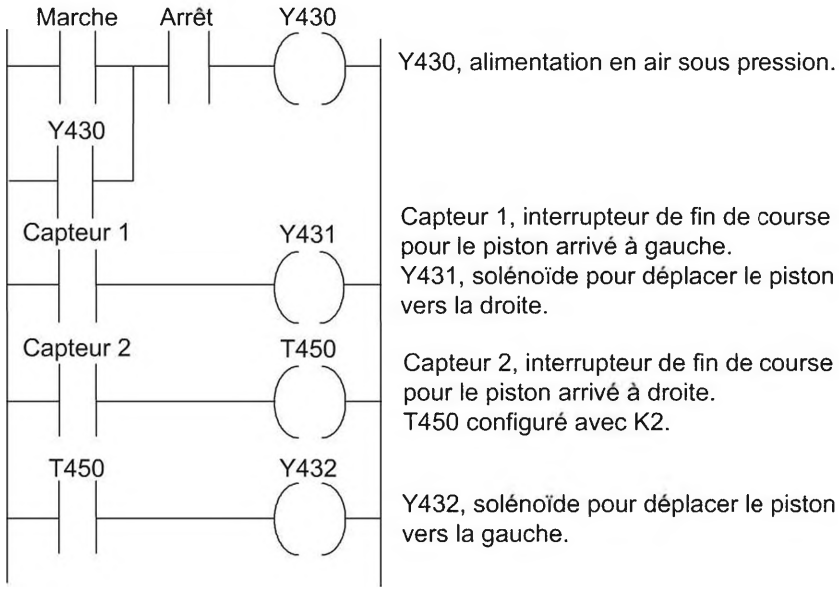


Figure A2.9 – Chapitre 14, problème 2.

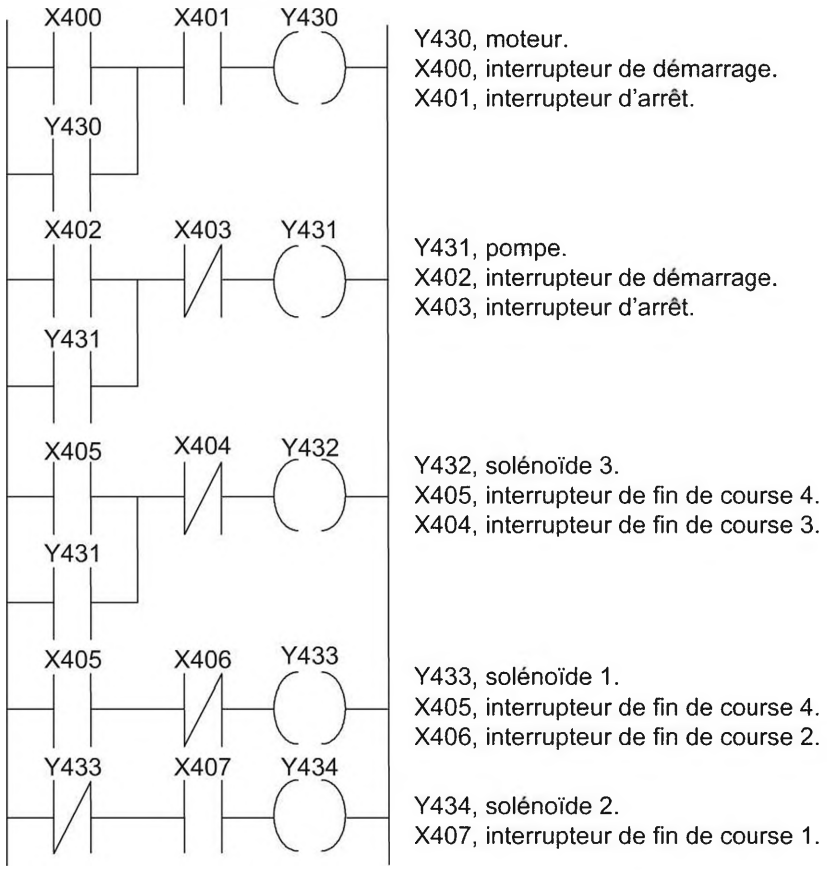
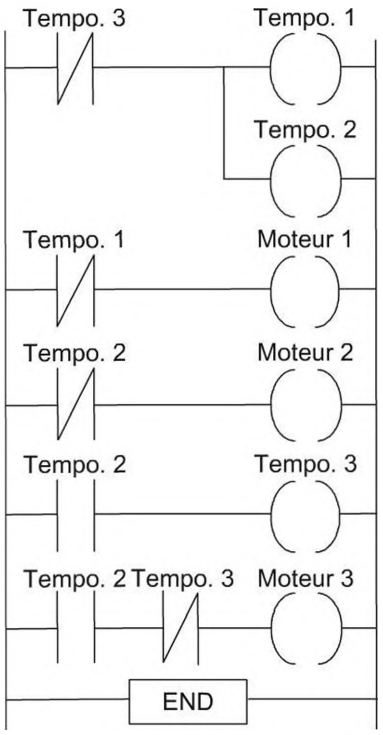


Figure A2.10 – Chapitre 14, problème 3.

Copyright © 2015 Dunod.



Les temporisateurs 1 et 2 sont démarrés lorsque le temporisateur 3 est inactif. Le temporisateur 1 est réglé sur 200 s, le temporisateur 2 sur 300 s et le temporisateur 3 sur 50 s.

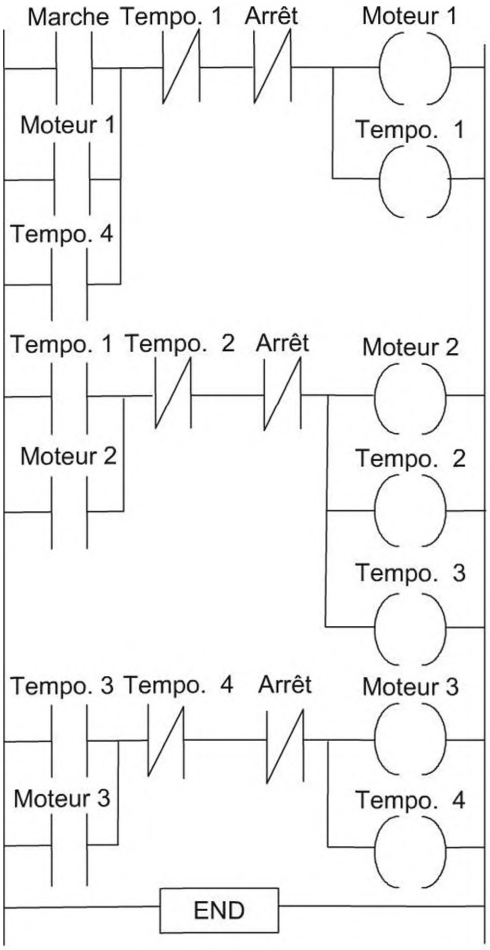
Le moteur 1 est arrêté après 200 s.

Le moteur 2 est arrêté après 300 s .

Le temporisateur 3 démarre lorsque le temporisateur 2 est désactivé.

Le moteur 3 démarre lorsque le temporisateur 2 est activé et le temporisateur 3, désactivé.

Figure A2.11 – Chapitre 14, problème 8.



Le temporisateur 1 est démarré lorsque l'interrupteur de mise en marche est activé. Le moteur 1 est activé jusqu'à ce que le temporisateur 1 s'arrête. Le temporisateur 1 est réglé sur 50 s.

Les temporisateurs 2 et 3 sont démarrés lorsque le temporisateur 1 s'arrête. Le moteur 2 est activé jusqu'à ce que le temporisateur 1 s'arrête. Le temporisateur 2 est réglé sur 50 s, le temporisateur 3, sur 40 s.

Le moteur 3 est activé lorsque le temporisateur 3 est actif et désactivé lorsque le temporisateur 4 s'arrête. Le temporisateur 4 est réglé sur 50 s.

Figure A2.12 – Chapitre 14, problème 9.

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

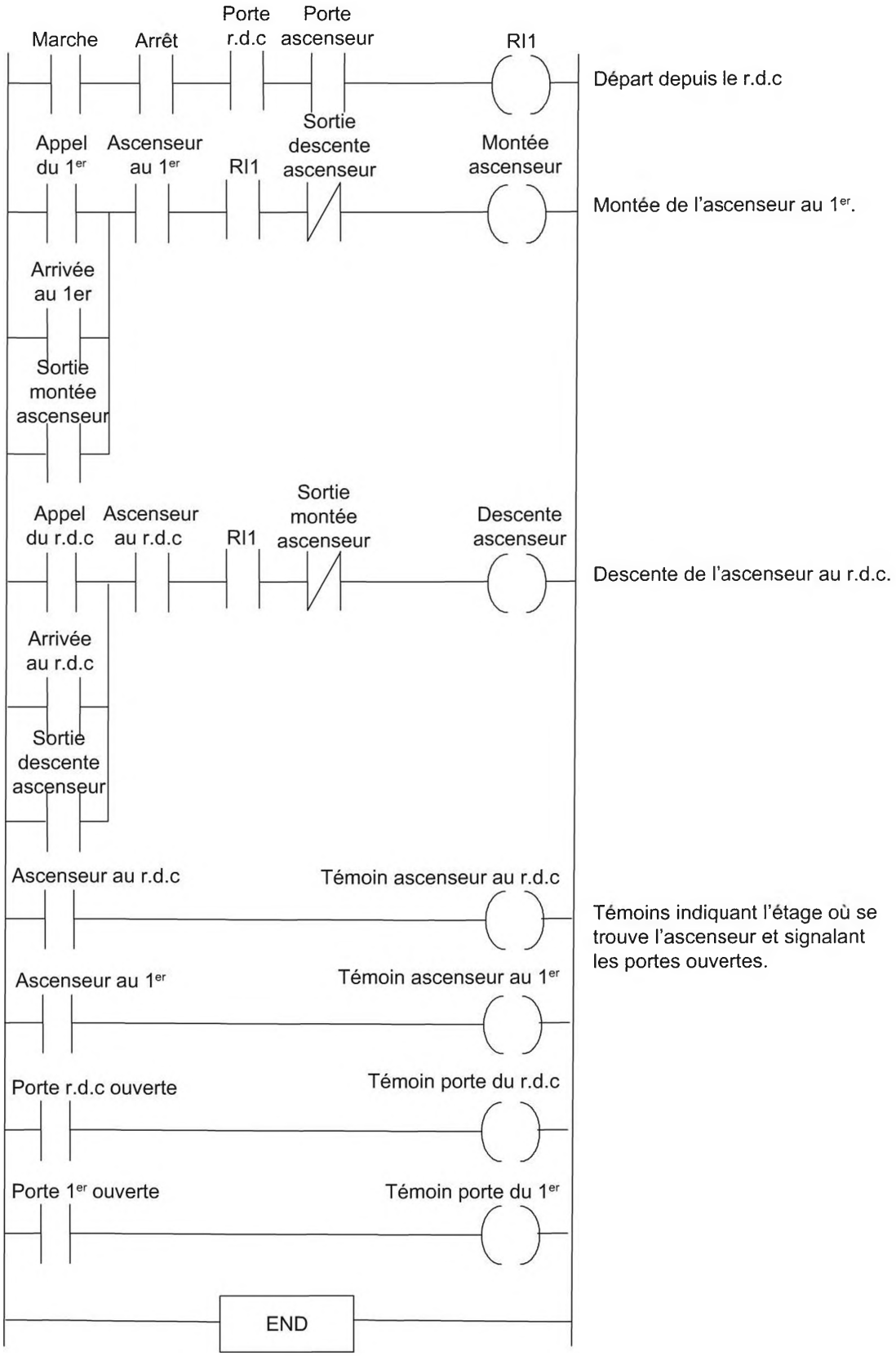


Figure A2.13 – Chapitre 14, problème 10.

Copyright © 2015 Dunod.

INDEX ALPHABÉTIQUE

Note : les numéros de page en italique font référence à des figures, ceux en gras, à des tableaux.

A

absorption de courant 11–12
  entrées 84, *84*
  sorties 87, *87*
accumulateurs, *Voir* compteurs
ACK, signal 103
actionneurs 46
  Voir aussi dispositifs de sorties
actions, dans un SFC 173, *174*
actualisation continue 113
addition
  de données dans des registres 297–298
  nombres binaires 70, 72
    à virgule flottante 73
adresses, bus 9–12
affectation, instruction (ST) 176
ajustement 301
ALARM, bloc 110
alarme, exemple de système *206*
alimentation
  commune 90, *90*
  individuelle 90, *90*
Allen-Bradley
  ControlNet, réseau 112
  Data Highway, réseau 106, 113
  PLC-5 *15*, 16, 112, 115
    exemple 325–346
Allen-Bradley, schémas à contacts
  addition, opération *298*
  comparaison supérieur à *295*
  compteurs *260*
    et temporisateurs 262–264, *264*
    progressifs-dégressifs *263*
  déplacement de données *293*
  notation 125, *126*
  opération monostable *200*
  registres à décalage *284*
  relais
    de contrôle maître 207
    sauvegardés par batterie 198
  sauts conditionnels *224*
  séquenceurs 267, *268*
  sous-programmes *226*, 226
  temporisateurs
    à l'enclenchement (TON) 235, *236*
    au déclenchement (TOF) 240, *240*, *241*
    rémanents (RTO) 243, *243*
  verrouillage-déverrouillage 202
amplificateurs
  différentiels 92, *92*
  inverseurs 91
  non inverseurs 91
amplification des tensions 91, *92*, *356*
  comparateurs à amplificateur opérationnel 93
analogiques
  dispositifs 8
  signaux, *Voir* signaux analogiques

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

analyse des schémas à contacts 123, *124*
angle de pas (moteurs pas à pas) 54
anneau, réseaux 105, *106*
API
  à propos de 5–7
  conception, *Voir* systèmes à API, conception
appareils de programmation 7, 19–20
  portatifs 19
Apple Firewire, bus 99
application, couche (modèle ISO/OSI) 108
applications
  des systèmes de commande 55–58
    Voir aussi programmes
  développement, *Voir* programmes
  stockage 323
architecture des API 9–12
  unité centrale de traitement (CPU) 7, 9–12, 83
arithmétique
  fonctions 297–298
    opérations 297–298
  sur les nombres binaires 70–74
    compléments à un et à deux 71–73
    nombres à virgule flottante 73–74
ASCII, codes 104
  exemple **104**
assembleur 121
ATN, ligne (IEEE-488) 100, **101**
automates programmables industriels, *Voir* API
autotest, *Voir* défauts, rechercher
auxiliaires, relais, *Voir* relais internes

B

bande de base 111
bandes transporteuses 55, *56*
  colis sur, détecter 58
  compteurs 259
  portes OU 128
  programme complet, exemple 369–375
    emballage de bouteilles 369–375
  suivi d'articles avec des registres à décalage 284–285, *286*
barrières
  de parking (exemple) 363–368
  immatérielles 315
bascules 76, 77, *78*, 201–205, *204*
  D 28–29, *29*
  JK, à entrée d'horloge 77, *78*
base (nombres à virgule flottante) 73
batterie
  dysfonctionnement 323
  relais sauvegardés par 198, *199*
BCD (décimal codé en binaire) 68–69, 292
  exemple de nombres **69**
bilame *36*
  thermostat 353
bimétallique, élément 35, *36*
binaire, système 65, 66, 291
  arithmétique 70–74
    compléments à un et à deux 71–73
    nombre signés et non signés 71
    nombres à virgule flottante 73–74
  conversion vers/depuis
    le système décimal 66
    le système hexadécimal 68
    le système octal 67
  exemple de nombres **69**
bistable 76
  bloc 202
bits 66, 115, 279, 291–292
  contrôle de parité 103
  d'activation
    du comptage (CU) 257
    du décomptage (CD) 258
  de dépassement
    inférieur (UN) 259
    supérieur (OV) 259

Copyright © 2015 Dunod.

de fin (DN) 258
de poids
faible (LSB) 66
fort (MSB) 66
de signe 71
jetons 106
mots 74, 85, 291
registres à décalage, *Voir* registres à décalage
blocs
de communication standard 109
fonctionnels 226–228, *227*
symboles 390–391
texte structuré (ST) 180–181
bobines
d'induction 49
de détection de front
descendant 200, *201*
montant 200, *201*
mémoire rémanente 198, *199*
mise à un et mise à zéro 201–205, *202*
opération monostable 204–205
programmes, exemples 205
Voir aussi bascules
Voir aussi relais internes
boîtier unique, systèmes en 12
adresses d'E/S 115
booléens
algèbre 140–145
BOOL, type 74
logique 126–133
ET logique 126–127
NON logique 129–130
NON-ET logique 130–131
NON-OU logique 132
OU exclusif logique 132, 133
OU logique 128
tables de vérité, *Voir* tables de vérité
texte structuré (ST) 394
type de données 74
Bosch, bus CAN 99
boucles de courant 20 mA 99, *99*
branchements
graphes de fonction séquentielle (SFC) 172–173
liste d'instructions (IL) 163–165
bus 9–12
communications
parallèles 99–102
série 95, 98
réseaux 105, 106, *106*
série universel (USB) 99

C

câble
à fibre optique 95
à paires torsadées 95
coaxial 95
CAN (*Controller Area Network*), bus 99
DeviceNet, réseau 112
CAN (convertisseur analogique-numérique) 85, *86*
capacité de stockage des unités de mémoire 9
capot, surveillance 315
capteurs 25–46
à fibre optique 46
d'état 46
de déplacement 40
capacitifs 40
de position 40
linéraire et angulaire résistifs 40
de pression 42–43
à membrane 42
à soufflet 42
détecteurs de niveau de liquide 31, 43
extensomètres 41–42
MX100AP 27
système de surveillance 58, *58*
de proximité à ultrasons 45
de température 35–39, 295
programme de régulation de température complet 353
définition 25

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

capteurs (suite)
détecteurs de niveau de liquide 31, 43
système de surveillance 58, *58*
extensomètres 41–42
amplificateurs différentiels 92, *93*
comme capteurs de force 42
gammes 46
intelligents 45
photoélectriques 32–33, *33*
réflectifs 32
transmissifs 32
réponse *27*
systèmes distribués 107
tension, changer 91–92
comparateurs à amplificateurs opérationnels 93
Voir aussi interrupteurs
caractères alphanumériques, codes, *Voir* ASCII, codes
Carrier Sense Multiple Access (CSMA) 106
Carrier Sense Multiple Access with Collision Detection (CSMA/CD) 106, 109
CASE, instruction (ST) 179
CEI 61131, norme 18
CEI 61131-1, norme
graphes de fonction séquentielle (SFC) 171
temporisateurs *234*
CEI 61131-3, norme 393
compteurs **265**
contrôle PID *301*
diagrammes de schémas fonctionnels 139
listes d'instructions 159, 165
opération monostable *200*
registres à décalage *283*
relais sauvegardés par batterie *199*
schémas à contacts 124
CEI 61508, norme 312
cellules
photoconductrices 33
photoélectriques 46
châssis 16
chauffage central (exemple) 356
chien de garde, temporisateurs 320
circuit
à trois fils 37, *37*
de commande 3
émetteur-récepteur asynchrone universel (UART) 96
intégré en capteur de température 354–355
CJP (saut conditionnel), *Voir* instructions de saut
CNA (convertisseur numérique-analogique) *89*
code machine 121
coefficient
de jauge 41
de température 35–37
colis sur bande transporteuse, détecter 58
collision des données 106
commande
à deux mains 315
directionnelle de moteurs 49–52, *51*
commande d'une vanne (exemple de programme)
liste d'instructions 167, *168*
programme complet 359–368
barrières d'un parking 363–368
mouvement cyclique 360
séquencement 360–363
vérin contrôlé, réinitialiser 368
relais de contrôle maître *210*
schémas à contacts *146*
communications
chemins, *Voir* bus
parallèles 95–96, 99–102, 116
contrôleurs 99
écouteurs 99
parleurs 99
série 95–99, 116
systèmes, *Voir* entrées-sorties, traitement
comparaison de données 294–296

Copyright © 2015 Dunod.

comparateurs à amplificateur opérationnel 93
complément
à deux 71–73, 291
à un 71–73
compteurs 255–269
dégressifs 255, **265**, 390–391
formes 255
progressifs-dégressifs 262, *263*, 390–391
séquenceurs 264–267
temporisateurs avec 262–264, *264*
conditionnelles, instructions
pseudocode 309, *309*
saut, *Voir* instructions de saut
texte structuré (ST) 178–179, 394
conducteur, corps humain 32
CONNECT, bloc 109
connecteurs DB-25 96, *97*
connexions distantes
codes ASCII 104
exemple **104**
protocoles 102–104
consoles 19
constantes (ST), définir 180
contacteurs 47
contacts
normalement fermés (NF) 47
normalement ouverts (NO) 47
control, séquenceur 267
contrôle
bus 9–12
d'accès au support (MAC), couche 109
de flux dans les communications série 96
de la liaison logique (LLC), couche 109
de parité 103
de redondance cyclique, codes CRC 104
directionnel, vannes 47–49, *49*
en boucle fermée 298–301, *299*
modes de contrôle 298–301
en deux étapes 301
contrôleurs
communications parallèles 99
systèmes distribués 107
Controller Area Network (CAN), bus 99
DeviceNet, réseau 112
ControlNet, réseau 112
convergence
graphes de fonction séquentielle (SFC) 172–173, *173*
simultanée (SFC) *173*
convertisseurs
analogique-numérique (CAN) 85, *86*
numérique-analogique (CNA) *89*
copie en masse des E/S 113–114
couches du modèle ISO/OSI 107–110
couple
d'accrochage (moteurs pas à pas) 54
de décrochage (moteurs pas à pas) 54
statique (moteurs pas à pas) 54
courant
absorption, *Voir* absorption de courant
alternatif
moteurs 52
unités d'entrées 84, *85*
continu
moteurs 49, *51*
unités d'entrées 84, *84*
fourniture, *Voir* fourniture de courant
CP1L, Omron *13*
CPU (*Central Processing Unit*) 7, 9–12, 83
cristaux piézo-électriques 43
CSMA (*Carrier Sense Multiple Access*) 106
CSMA/CD (*Carrier Sense Multiple Access with Collision Detection*) 106, 109

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

CTD, compteur dégressif, *Voir* compteurs dégressifs
CTN, semiconducteurs 37
CTS (*Clear To Send*), signal 103
CTU, compteur progressif, *Voir* compteurs progresifs
cycles, API 114
  opération monostable 199–201
  schémas à contacts 123
  temps de 114

D

D
  bascule 28–29, *29*
  verrous 76–77, *76*
D101 à D108, lignes (IEEE-488) **101**
Darlington, couple 33
Data Highway (Allen-Bradley), réseau 106, 113
DAV, ligne (IEEE-488) *100*, **101**, 102
DB-25, connecteur 96, *97*
débitmètres 44
débits
  communications série 97–98
  en baud 97–98
décimal codé en binaire (BCD) 68–69, 292
  exemple de nombres **69**
décimal, système 65, 292
  complément à deux **72**
  conversion vers/depuis
    le système BCD 69
    le système binaire 66
    le système hexadécimal 68
    le système octal 67
  nombres
    à virgule fixe 73–74
    à virgule flottante 73–74
    exemple **69**
défauts, rechercher 320–323
  programme, stockage 323
  techniques de détection 320–323
dépassement
  inférieur (UN), bit 259
  supérieur (OV), bit 259
dérivation, temps de 301
dérive 26
  à zéro 26
dernière sortie activée (technique de détection des défauts) 321, *322*
destination, séquenceur 267
détecteurs
  à ultrasons 46
  de niveau de liquide 31, 43
    système de surveillance 58, *58*
  de proximité
    capacitifs 46
    inductifs 46
détection de front, bobines de 200, *201*
DeviceNet, réseau 112
diagnostics, *Voir* défauts, rechercher
diagrammes de schémas fonctionnels (FBD) 137–145, *141*
  blocs de portes logiques *141*
  exemple de programmes 145–147
  symboles de base *138*
dialogue 100, *103*
DINT (entier double) 74
diodes
  de roue libre 94
  électroluminescentes (LED) 33
dispositifs
  de détection de présence 46
  de sorties 46–55
    moteurs, *Voir* moteurs
    relais, *Voir* relais
  numériques, définition 8
dispositifs d'entrées 25–46
  capteurs de position/déplacement 40
  capteurs de pression 42–43
    extensomètres 41–42
  capteurs de température 35–39, 295
    régulation, programme complet 353

Copyright © 2015 Dunod.

débitmètres 44
détecteurs de niveau de liquide 31, 43
système de surveillance 58, *58*
encodeurs 34–35
extensomètres 41–42
amplificateurs différentiels 92, *93*
interrupteurs de proximité 31–32
interrupteurs mécaniques 28–31, *28*
schéma à contacts (exemple) 122, *122*
test 316
distributeur de boissons 378
programme FBD *379*
schéma à contacts *380*
divergence
parallèle 172, *172*
sélective 172, *172*
diviseur de tension 91, *91*
division
opération 297–298
texte structuré (ST) 178
documentation
d'un programme (exemple) 325–346
des systèmes à API 325–346
données 74
binaires, *Voir i* bits
bus 9–12
collisions (dans les réseaux) 106
communications
parallèles 95–96, 99–102, 116
série 95–99, 116
Voir aussi entrées-sorties, traitement
contrôle de parité 103
gestion 291–302
comparaison de données 294–296
contrôle en boucle fermée 298–301, *299*
déplacement de données 293–294
fonctions arithmétiques 297–298
opérations 297–298
sélection de données 296
stockées, *Voir* relais internes ; registres à décalage
durées 74

E

E2K-X, détecteur capacitif d'Omron 32
E4C-DS100, capteur de proximité à ultrasons d'Omron 45
E4C-DS30, capteur de proximité à ultrasons d'Omron 45
E4C-DS80, capteur de proximité à ultrasons d'Omron 45
écouteurs (communications parallèles) 99
écrans tactiles 32
égal à, comparaison 294
emballage de bouteilles (exemple de programme) 369–375
emplacement, passage 106
encodeurs 34–35
à deux pistes 34
à trois pistes *34*, 34
absolus 34, *35*
incrémentaux 34, *34*
ENQ, signal 103
ENQ/ACK, protocole 103
entiers
courts (SINT) 74
doubles (DINT) 74
simples (INT) 74
entrées
fourniture et absorption de courant 11–12, *11*, 84, *84*
inversées, dans les diagrammes de schémas fonctionnels *139*
opérations arithmétiques sur 297–298
relais internes avec multiple conditions d'entrée 194–196
schémas à contacts 123, *136*

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

entrées (suite)
test (développement de programme) 316
entrées-sorties, adresses 115–116
entrées-sorties, traitement 83–116
connexions distantes 94–104, *94*, *95*
entrées 113–115
réseaux 105–110
normes 107–110
systèmes distribués 107
systèmes commerciaux, exemples 110–113
traitement du signal 90–94
comparateurs à amplificateurs opérationnels 93–94
niveaux de tension, changer 90–92, *93*
protection de la sortie 94
unités d'entrées-sorties 9–11, 12–17, 83–90
EOI, ligne (IEEE-488) **101**
EPROM (*Erasable and Programming Read-Only Memory*) 323
EPROM (*Erasable and Programming Read-Only Memory*)
capteurs intelligents 45
erreur
contrôle dans les communications 103
communications série 96
d'hystérésis 26, *26*
d'un capteur 25, *26*
de mesure (capteurs) 25
non-linéarité 25, *26*
signal, réagir à, *Voir* contrôle en boucle fermée
esclave, réseaux 105
ET
opération
algèbre 140
portes logiques 126–127, *127*, *140*
blocs fonctionnels *141*
liste d'instructions (IL) *161*, *166*
Ethernet 112
étoile, réseaux 105, *106*
exposant (nombres à virgule flottante) 73
extensomètres 41–42
à résistance métallique 41, *42*
amplificateurs différentiels 92, *93*

F

fanions (bits de communication série) 97–98
Voir aussi relais internes
FBD (*Function Block Diagram*), *Voir* diagrammes de schémas fonctionnels (FBD)
FEM, protection 94
feux de circulation, séquencement 168, *169*
temporisateurs 243, *244*
fiabilité, système de mesure 27
file, séquenceur 267
Firewire, bus 99
fonctions monostables 204–205, *205*
FOR...DO, itération (ST) 179, 394
forçage, test des entrées-sorties 317, *318*
formes d'automates 1
fourniture de courant 11–12
entrées 84, *84*
sorties 87, *87*
fréquence de commutation 358

G

gammes de capteurs 46
GEM-80, API 116
General Electric, réseau GENET 106
General-Purpose Instrument Bus (GPIB) 99
GENET, réseau (General Electric) 106
GND, ligne (IEEE-488) **102**

Copyright © 2015 Dunod.

GPIB (*General-Purpose Instrument Bus*) 99
Grafcet 171
graphes de fonction séquentielle (SFC) 168–175, 363, 392
  actions 173, *174*
  divergence et convergence 172–173
Gray, code 35, **36**

H

Hewlett-Packard Instrumentation Bus 99
hexadécimal, système 67–68
  exemple de nombres **69**
hiérarchies de communication 107, *107*
horloge (CPU) 9–12

I

I²C, bus 99
IEEE 1451.4, norme 45
IEEE 802.3, norme 109
IEEE 802.4, norme 109
IEEE Projet 802 109
IEEE-488, bus 99, *100*, **101**
IF, instruction (ST) 178, 394
IFC, ligne (IEEE-488) **101**
IL (*Instruction List*), *Voir* liste d'instructions (IL)
impulsions brèves 259
inducteurs, protection des sorties 94
induit (moteurs CC) 49
inférieur à, comparaison 294
inférieur ou égal à, comparaison 294
instructions
  de boucle
    pseudocode *310*, 310, *311*
    texte structuré (ST) 179–180
  de saut 223–225, *224*
    blocs fonctionnels 226–228
    sauts imbriqués 225, *225*
    sous-programmes 226–228
interfaces
  d'entrées-sorties des API 8
  de communication des API 8
  de sortie des systèmes API 8
  RS422 98
Inter-IC Communication (I²C), bus 99
interrupteurs
  à molette 292, *292*
  d'arrêt 312–314
    emplacement 147
  d'arrêt d'urgence 147, *148*, 314–315
    emplacement *148*
  de fin de course 29, 46
  de proximité 31–32
    à courant de Foucault 31, *31*
    capacitifs *31*, 32, *41*
    inductifs 32
  de sécurité 46
  intégrés 29, *30*
  marche-arrêt, schémas à contacts pour 122, *123*
  mécaniques 28–31, *28*
    schémas à contacts *122*, 122
  normalement fermés (NF) 28
    schémas à contacts 124
  normalement ouverts (NO) 28
    schémas à contacts 124
  Reed *31*, 31
  SPDT (*Single Pole/Double Throw*) 28–29
  Voir aussi systèmes logiques (portes logiques) ; photoélectriques, capteurs et interrupteurs
inverseurs, *Voir* NON, portes logiques
ISO/OSI, modèle 107, *108*
itération, instruction (ST) 179–180, 394

J

jeton, passage 106
JK, bascules 77, *78*

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

L

langages
  assembleur 121
  de haut niveau 121
  machine 136
large bande 110
lave-linge, SFC pour *171*
LD (*Ladder Diagram*), *Voir* programmes et schémas à contacts (LD)
length, séquenceur 267
levier à galet central 30
liaison de données, couche (modèle ISO/OSI) 108
lieu de travail sécurisé, *Voir* systèmes sécurisés
lignes
  de production, *Voir* bandes transporteuse
  Voir programmes et schémas à contacts (LD)
liste d'instructions (IL) 159–168
  codes de branchement 163–165
  multiples lignes 166
  programmes, exemples 167
LLC (*Logical Link Control*), couche 109
LM35, composant *38*
LM3911N, circuit *38*
logiciels pour API 19
  test et vérification 317
logique
  combinatoire 65, 74–75
  séquentielle 65, 76–79
LREAL (réels longs) 74
LSB (*Least Significant Bit*) 66
LVDT (*Linear Variable Differential Transformer*) 40, *41*

M

MAC (*Media Access Control*), couche 109
maître, réseaux 105
mantisse 73
Manufacturing Automation Protocol (MAP) 110–111, **111**, **175**
Manufacturing Message Service (MMS) 110
marqueurs, *Voir* relais internes
mask, séquenceur 267
matériel des API 7–8
mathématique, binaire, *Voir* binaire, système, arithmétique
MCR (*Master Control Relay*), *Voir* relais de contrôle maître (MCR)
MELSEC FX3U, API compact (Mitsubishi) *14*, 115
MelsecNET, réseau (Mitsubishi) 106
MELSOFT, langage 19
mémoire 9
  morte reprogrammable (EPROM) 323
    capteurs intelligents 45
  morte (ROM) 45
  pour les bits de données, *Voir* registres
  programme, stockage 323
  unité 8
    capacité de stockage 9
  vive (RAM), panne de batterie et 323
méthodes de programmation, *Voir* diagrammes de schémas fonctionnels (FBD) ; liste d'instructions (IL) ; programmes et schémas à contacts (LD) ; graphes de fonction séquentielle (SFC) ; texte structuré (ST)
mise à zéro
  bit (RES) 259
  bobine 201–205, *202*
    opération monostable 204–205
    programmes, exemples 205
    Voir aussi bascules
  des compteurs 255, 259
Mitsubishi
  MELSEC FX3U, API compact *14*, 115

Copyright © 2015 Dunod.

MelsecNET, réseau 106
MELSOFT, langage 19
mode monitoring 318
Mitsubishi, schémas à contacts
commande d'une bande transporteuse (exemple de programme) 374
comparaison supérieur à *295*
compteurs **265**
déplacement de données *293*
mnémoniques des instructions
portes ET *161*, *166*
portes NON-ET *163*
portes NON-OU *162*
portes OU *162*
portes OU exclusif *164*
notation 125
opération monostable *200*
registres à décalage 280, 281, *282*
relais
de contrôle maître 207
internes avec multiple conditions d'entrée *195*
sauvegardés par batterie 198
sauts conditionnels *224*
séquencement de vannes (exemple) 361–363
barrières d'un parking *366*
entrées-sorties 368
sous-programmes 226, *226*
temporisateurs à l'enclenchement (TON) 235, *236*
MLI (modulation de largeur d'impulsion) 51
MMS (*Manufacturing Message Service*) 110
mode
contrôle en boucle fermée
combinaison 300, *300*
dérivée 300, *300*
intégrale 300, *300*
proportionnel 299, *300*
tout ou rien 298, *300*
de débogage (Telemecanique) 318
de test (Siemens) 318
monitoring (Mitsubishi) 318
modulation de largeur d'impulsion (MLI) 51, *51*
modules
d'entrées-sorties 16
distantes 94–104, *94*, *95*
de commande 16
de communication 17
monostable 199–201
fonctions de mise à un et de mise à zéro 204–205
monte-charge 56
moteurs 49–52
bipolaires 55
CC sans balais 52
interrupteur d'arrêt, emplacements 147
pas à pas 52–55, *52*
à aimants permanents 52, *53*
à réluctance variable 52, *54*
hybrides 52
protection de la sortie 94
systèmes distribués 107
temporisateurs avec (exemples) 236, *237*, 245, *245*
unipolaires 55
Motorola MPX100AP, capteur 27, 43
mots (données binaires) 9, 74, 85, 291
Voir aussi bits
mouvement cyclique, piston d'un vérin (exemple) 360
MPX100AP, capteur (Motorola) 27, 43
MSB (*Most Significant Bit*) 66, 71
multiplexeurs 84–85, *85*
multiplication, opération 297–298

N

NDAC, ligne (IEEE-488) *100*, **101**, 102
niveaux
d'entrée 9, 83
de quantification 85
de sortie 11, 83
RS232, interface *98*

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

niveaux (suite)
de tension
changer 91–92
réduire 91–92, *91*
des signaux 9, 83
RS232, interface *98*
nombres
à virgule
fixe 73
flottante 73–74
binaires
compléments à un et à deux 71–73
négatifs 71
non signés 71, 74
positifs 71
signés 71, 74
réels longs (REAL) 74
NON
opération, algèbre 142
portes logiques 129–130, *130*, *140*
NON-ET, portes logiques 28–29, *29*, 130–131, *131*, *140*
liste d'instructions *163*
NON-OU, portes logiques 132, *132*, *140*
blocs fonctionnels *141*
liste d'instructions *162*
normalement fermés (NF)
contacts 47
interrupteurs 28
schémas à contacts 124
normalement ouverts (NO)
contacts 47
interrupteurs 28
schémas à contacts 124
notation scientifique 73
NOTIFY, bloc 110
NRFD, ligne (IEEE-488) *100*, **101**, 102
numériques
signaux, *Voir* signaux numériques
systèmes, *Voir* systèmes numériques

O

octal, système 67–68
exemple de nombres **69**
octets 115
Omron
capteurs de proximité à ultrasons 45
CP1L *13*
détecteur capacitif 32
Omron Industrial Automation 30
Open Systems Interconnection (OSI), *Voir* ISO/OSI, modèle
opérateurs (ST) 394
opérations arithmétiques 297–298
ordinateurs personnels
comparés aux API 6
programmer les API 19
organigrammes
développement de programmes 307–311, *308*
séquentiels 308, 376
orifice, débitmètres 44, *44*
OSI (*Open Systems Interconnection*), *Voir* ISO/OSI, modèle
OU
opération, algèbre 142
portes logiques 128–130, *129*, *140*
blocs fonctionnels *141*
liste d'instructions (IL) *162*
OU exclusif
opération, algèbre 144
portes logiques 133, *133*, *140*
blocs fonctionnels *141*
liste d'instructions (IL) 163–165

P

parleurs (communications parallèles) 99
pas (moteurs pas à pas) 52
pavés tactiles 32
phase (moteurs pas à pas) 54
Phillips I²C, bus 99

Copyright © 2015 Dunod.

photocoupleurs 9, *10*, 11, 84
photodiodes 33
photoélectriques, capteurs et interrupteurs 32–33, *33*
phototransistors 33
physique, couche (modèle ISO/OSI) 108
PID, contrôle 300, *301*
plage
  capteur 26
  de rotation (moteurs pas à pas) 55
planchers sensibles de sécurité 315
plongeurs
  à galet 30
    étanches 30
    haute sensibilité 30
    transversal 30
    transversal étanches 30
  à goupille 30
ponts de Wheatstone 36–37, *37*
  amplificateurs différentiels 92, *93*
  extensomètres 41, *42*
position, séquenceur 267
position/déplacement, capteurs 40
positionnement linéaire avec les moteurs pas à pas 52, *53*
potentiomètres 40, *40*
  linéaires 40
  rotatifs 40, *40*
précision d'un capteur 25, 85–87
  thermocouples 87
présentation, couche (modèle ISO/OSI) 108
pression
  absolue 42
  capteurs 42–43
  manométrique 42
prise vampire 111
processus de mélange (exemple de programme) 376
PROFIBUS DP, réseau (Siemens) 107
Profibus (*Process Field Bus*), système 113
programmes 353–377
  commande d'une bande transporteuse 369–375
    emballage de bouteilles 369–375
  commande de processus 376
  pour les API 17–20
  régulation de température 353
  saisie 136–137
    symboles d'un schéma à contacts 136
  séquencement de vannes 359–368
    barrières d'un parking 363–368
    mouvement cyclique 360
    vérin contrôlé, réinitialiser 368
  stocker 323
programmes et schémas à contacts (LD) 21, 122–126, *122*, *141*
  bobines mise à un et mise à zéro 201
  conventions 123–124
  entrées 123, *136*
  exemple de programmes 145–147
  graphes de fonction séquentielle (SFC) *170*, *174*
  lignes multiples 166
  liste d'instructions 161–163
  logique, *Voir* systèmes logiques (portes logiques)
  pseudocode, écrire et convertir
    boucles *310*, 310, *311*
    flux conditionnel 309
    séquences 308
  relais à verrouillage *134*
  relais de contrôle maître *207*, *211*, *212*
  relais internes 194–198
    conditions d'entrée multiples 194–196
    programmes à verrouillage 196–197
    temps de réponse 198
  saisie des programmes 136–137
  sauts, *Voir* instructions de saut
  sorties 123, 135–136, *135*

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

programmes et schémas à contacts (LD) (suite)
symboles 136
de base 124–126, *125*, 389
texte structuré (ST) contre 181
Voir aussi les sociétés (par exemple, Siemens) ; liste d'instructions (IL)
programmes, développer 307
défauts, rechercher 320–323
programme, stockage 323
techniques de détection 320–323
documentation 325–346
mise en service 316–319
simulation 318
test des entrées-sorties 316
vérification du logiciel 317
organigrammes et pseudocode 307–311
systèmes sécurisés 311–316
API de sécurité 316
fonctions de sécurité 315–316
relais d'arrêt d'urgence 314–315
progressifs, compteurs 255, **265**, 390–391
progressifs-dégressifs, compteurs 262, *263*, **265**, 390–391
Projet 802 (IEEE) 109
protocoles de communication 96, 102–104
réseaux 105, 107–110
pseudocode 307–311
puissance (nombres à virgule flottante) 73

R

radiateur soufflant 379
programme FBD *382*
schéma à contacts *381*
RAM (*Random Access Memory*) 323
RCV, bloc 110
READ, bloc 110
REAL, données 74
rebond, dans les interrupteurs mécaniques 28–29
réels, nombres 74
registres 279, 291–292
à décalage 279–285, *281*
schémas à contacts 280–285
séquencement, exemple d'application 284, *285*
suivi d'articles 284–285, *286*
données, déplacer entre 293–294
huit bits 279
régulation de température tout ou rien (exemple) 358
relais 46, *47*, 87
à rémanence 198
à verrouillage 47, 134
relais internes 196–197
d'arrêt d'urgence 314–315, 316
double voie 314–315
de base 1–2, *2*
de contrôle maître (MCR) 206–211, *207*
multiples 209
programmes, exemples 209–211
schémas à contacts *207*, *211*, *212*
externes 193–194
moteur, inverser la rotation 51
sauvegardés par batterie 198, *199*
sortie à 10, *88*
symboles 3–4, *3*
relais internes 193–194, *194*
mise à un et mise à zéro 201–205
opération monostable 199–201, *200*
mise à un et mise à zéro 204–205
programmes à contacts 194–198
conditions d'entrée multiples 194–196
programmes à verrouillage 196–197
temps de réponse 198
programmes, exemples 205
registres à décalage, *Voir* registres à décalage

Copyright © 2015 Dunod.

relais de contrôle maître 206–211
sauvegardés par batterie 198, *199*
Voir aussi bascules
REMOTE_VAR, bloc 110
REN, ligne (IEEE-488) **102**
REPEAT...UNTIL, itération (ST) 180, 394
répétabilité, système de mesure 27
réplication 321
réseau, couche (modèle ISO/OSI) 108
réseaux 105–110
en bus 105, *106*
hôte 105
industriels 113
locaux (LAN) 105
normes 107–110
pair à pair 105
systèmes distribués 107
réservoir de stockage 376, *378*
résolution (tension) 85–87
RETURN, instruction (ST) 181
Rockwell Automation
PLC-5 d'Allen-Bradley *15*, 16, 20, 113
RSLogix, langage 20
rotation du moteur, inverser 51
RS
bloc fonctionnel 228, *228*
verrous 77, *77*, **78**
RS232, interface 96
signaux RTS et CTS 103
RS422, interface 98, *98*
RS485, interface 113
RSLogix, langage 20
RTD (*Resistive Temperature Detector*) 36–37
RTO (*Retentive Timer On*), *Voir* temporisateurs rémanents (RTO)
RTS (*Ready To Send*), signal 103

S

S7-1200, Siemens 317
SAA 1027, circuit 55, *56*
sauts imbriqués 225, *225*
SCADA (*Supervisory Control And Data Acquisition*) 107
schémas à contacts
et schémas de circuits 122
versions graphique et semi graphique 124–126
sécurité 17
sélection, bloc fonctionnel 296
semiconducteurs 37
CTN 37, *38*
extensomètres 41–42
amplificateurs différentiels 92, *93*
SEND, bloc 110
sensibilité, système de mesure 26
séquencement
déplacement d'un piston (exemple) 360–363
programmer 308
registres à décalage 284, *285*
temporisateurs TON 235
séquenceurs 264–267
control 267
destination 267
file 267
length 267
mask 267
position 267
session, couche (modèle ISO/OSI) 108
SFC (*Sequential Function Chart*), *Voir* graphes de fonction séquentielle (SFC)
SFC (SIMATIC S7-Graph), langage 20
SHIELD, ligne (IEEE-488) **101**
Siemens
mode de test 318
PROFIBUS DP, réseau 107
Profibus (*Process Field Bus*), système 113
S7-1200 317
SIMATIC S5, adressage 115
SIMATIC S5-95F, API 316
SIMATIC STEP 7, langage 20

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

Siemens, schémas à contacts
addition, opération *298*
commande d'une bande transporteuse (exemple de programme) 374–375
comparaison supérieur à *295*
compteurs *260*, *263*, **265**
déplacement de données *293*
diagrammes de schémas fonctionnels *142*
fonctions de mise à un et de mise à zéro 202, 204–205, *205*
mnémoniques des instructions 161
portes ET *161*, *166*
portes NON-ET *163*
portes NON-OU *162*
portes OU *162*
portes OU exclusif *164*
notation 125, *126*
registres à décalage *284*
relais
de contrôle maître 208
internes avec multiple conditions d'entrée *195*
sauts conditionnels *225*
bloc de sous-programme *227*
séquencement de vannes (exemple) *364*
barrières d'un parking *367*
entrées-sorties 368
temporisateurs
à l'enclenchement (TON) 235, *236*
au déclenchement (TOF) 240, *240*
signaux
analogiques 83, 84–87
comme sortie 87, 90
comparateurs à amplificateurs opérationnels 92–93
niveaux de tension, changer 91–92, *93*
protection de la sortie 94
standard 90, *91*
traitement du signal 90–94
discrets, définition 83
numériques 10
capteurs 25
chemins, *Voir* bus
conversion vers/depuis l'analogique 85, *86*, *89*
définition 8, 83
niveaux 10, 83
SIMATIC S5, adressage (Siemens) 115
SIMATIC S5-95F, API (Siemens) 316
SIMATIC STEP 7, langage (Siemens) 20
simulation 318
SINT (entiers courts) 74
solénoïdes, comme actionneurs
protection de la sortie 94
relais 47
vannes de commande directionelle 47, *48*
sortie séquenceur (SQO) 267
sorties
fourniture et absorption de courant 11, *11*, 87, *87*
protection 94
relais internes avec circuit de verrouillage 194–196
schémas à contacts 123, 135–136, *135*
taille d'impulsion 89
tension des capteurs, changer 91–92
comparateurs à amplificateurs opérationnels 93
test (développement de programme) 316
sous-programmes 226–228
blocs fonctionnels 226–228
soustraction
de données dans des registres 297–298
nombres binaires 70, 72
SR, bloc fonctionnel 228, *228*
SRQ, ligne (IEEE-488) **101**

Copyright © 2015 Dunod.

ST (SIMATIC S7-SCL), langage 20
ST (*Structured Text*), *Voir* texte structuré (ST)
stabilité, système de mesure 26
STATUS, bloc 110
stockage
  d'un programme 323
  de bits, *Voir* mémoire ; relais internes ; registres à décalage
suivi d'articles avec des registres à décalage 284–285, *286*
supérieur à, comparaison 294
supérieur ou égal à, comparaison 294
symboles 389–394
synchronisation dans une communication série 96
systèmes
  à sécurité intrinsèque 312
  binaire, *Voir* binaire, système
  contrôlés par microprocesseur 5
    Voir aussi entrées d'un API
  d'E/S commerciaux, exemples 110–113
  de chargement (exemple) *206*
  de commande
    à relais, schéma 3–4, *4*
    application 55–58
    d'un robot 57, *57*
    traditionnels 2
  de numération, *Voir* binaire, système
  de surveillance
    capot de protection 315
    détecteurs de niveau de liquide 58, *58*
  de verrouillage d'une machine 74, *75*
  décimal, *Voir* décimal, système
  distribués 107
  en boîtier unique, adresses d'E/S 115
  hexadécimal, *Voir* hexadécimal, système
  logique
    combinatoire 65
    séquentielle 65, 76–79
  modulaires (rack) 12, 84–85
    adresses d'entrées-sorties 115
  numériques 65–79
  sécurisés 311–316
    API de sécurité 316
    fonctions de sécurité 315–316
    relais d'arrêt d'urgence 314–315
systèmes à API
  à propos de 12–20, 21
  conception 307–346
    Voir aussi programme, développer
  documentation 325–346
  mise en service 316–319
    simulation 318
    test des entrées-sorties 316
    vérification du logiciel 317
  sécurisés, *Voir* systèmes sécurisés
systèmes logiques (portes logiques) 65, 75, 126–133, 391–392
  algèbre booléenne 140–145
  combinatoires 65, 74–75
  diagrammes de schémas fonctionnels 139–140
  ET logique 126–127, *127*, *140*
    blocs fonctionnels *141*
    liste d'instructions (IL) *161*, *166*
  exemple de programmes 145–147
  NON logique 129–130, *130*, *140*
  NON-ET logique 28–29, *29*, 130–131, *131*, *140*
    liste d'instructions (IL) *163*
  NON-OU logique 132, *132*, *140*
    blocs fonctionnels *141*
    liste d'instructions (IL) *162*
  OU exclusif, logique 133, *133*, *140*
    blocs fonctionnels *141*
    liste d'instructions (IL) 163–165
  OU logique 128, *129*, *140*
    blocs fonctionnels *141*
    liste d'instructions (IL) *162*

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

OU logique (suite)
séquentiels 65, 76–79
symboles *140*

T

tableaux de registres, *Voir* RAM des données
tables de vérité 65, 75, 75
ET logique 126–127
NON logique 129
NON-ET logique 130
NON-OU logique 132
OU exclusif, logique 133
OU logique 128
verrous 76
tâches
adaptées aux systèmes de commande 1
d'automatisation 7
taille d'impulsion, sortie 89
taux
d'accrochage (moteurs pas à pas) 55
de décrochage (moteurs pas à pas) 55
Telemecanique
bobines mise à un et mise à zéro 202
mode de débogage 318
schémas à contacts
mise à un et mise à zéro 202, 204–205, *205*
notation 125, *126*
temporisateurs à l'enclenchement (TON) 235, *236*
téléphones mobiles 32
témoin
clignotant à base de temporisateurs (exemple) 243, *244*
de signalisation (exemple)
dernière sortie activée (technique de détection d'un défaut) 321, *323*
liste d'instructions 167, *167*
schémas à contacts *146*
temporisateurs 243, *245*
température, capteurs 35–39, 295
régulation, programme complet 353
temporisateurs 233–245
à impulsion (TP) 234, *234*, 241–242, 391
à l'enclenchement (TON) 234, *234*, 235–239, 391
créer 239, *239*
cycle marche-arrêt 238, *239*
en cascade 237–238
séquenceurs 235
au déclenchement (TOF) 234, *234*, 239–241, 391
avec des compteurs 262–264, *264*
chien de garde 320
cycliques marche-arrêt 238, *238*
en cascade *238*
programmes, exemples 243
rémanents (RTO) 243
types 233
temps
d'analyse, API 114
réponse d'un relais interne 198
d'établissement 26
d'intégration 301
de cycle des API 114
de montée, capteur 26
de réponse
avec des relais internes 198
capteur 26
test des interrupteurs 316
test (développement de programmes)
défauts, rechercher 320–323
programme, stockage 323
techniques de détection 320–323
entrées-sorties 316
vérification du logiciel 317
Texas Instruments
TIWAY, réseau 107
TSL220, capteur *33*

Copyright © 2015 Dunod.

texte structuré (ST) 176–181, 394
  instructions
    conditionnelles 178–179, 394
    d'itération 179–180, 394
  opérateurs 394
  programmes 180–181
  schémas à contacts contre 181
thermistances *38*, 354–355, *355*
  Voir aussi semiconducteurs
thermocouples 38–39, *39*, **39**, 354–355
  précision 87
thermodiodes 37–38
thermomètre à résistance de platine (RTD) 36–37
thermostats, *Voir* capteurs de température
thermotransistors 37–38
tige plastique 30
TIWAY, réseau (Texas Instruments) 107
TOF (*Timer Off Delay*), *Voir* temporisateurs au déclenchement (TOF)
TON (*Timer On Delay*), *Voir* temporisateurs à l'enclenchement (TON)
Toshiba, schémas à contacts
  compteurs **265**
  registres à décalage 281, *283*
  séquenceurs 266, *266*
  temporisateurs à l'enclenchement (TON) *236*
TP (*Pulse Timer*), *Voir* temporisateurs à impulsion (TP)
traitement du signal 90–94
  changer les niveaux de tension 91–92, *93*
  comparateurs à amplificateur opérationnel 93
  protection de la sortie 94
transducteurs, définition 25
transfert de données 293–294
transformateurs différentiels à variation linéaire (LVDT) 40, *41*
transistor, sortie à 10, *88*
transport, couche (modèle ISO/OSI) 108
triac, sortie à 11, *88*
TSL220, capteur (Texas Instruments) *33*
TYPE, instruction (ST) 180
types de données (ST), définir 180

U

UART (*Universal Asynchronous Receivers/Transmitter*) 96
UDINT, données 74
UINT, données 74
ULINT, données 74
ultrasons 45
unités
  centrales de traitement (CPU) 9–12, 83
  d'alimentation 7
  d'entrées-sorties 9–11, 12–17, 83–90
  de traitement 7
URCV, bloc 110
USB (*Universal Serial Bus*) 99
USEND, bloc 110
USINT, données 74
USTATUS, bloc 110

V

valeurs
  actuelles, compteurs 255, 256
  attendues, vérification 323
  cumulées (ACC) 257
  de présélection (PRE) 257
  initiales (ST), définir 180
  maximales, sélectionner *296*
  minimales, sélectionner *296*
vannes
  4/2 48, *48*
  à deux positions *48*
  à tiroir 47
  de commande directionelle 47–49, *49*

Copyright © 2015 Dunod.
© Dunod – La photocopie non autorisée est un délit.

variables
à deux états 75
logiques 75
texte structuré (ST), définir 180
vérification
de valeurs 323
du minutage 320
logiciel de l'API 317
vérins 49, *50*
double effet 49, *50*
simple effet 49, *50*
verrous 76–77
D 76–77, *76*
à entrée d'horloge 76–77, *76*
RS 77, *77*, **78**
à entrée d'horloge 77, *77*, **78**
transparents 76
virgule
fixe, nombres à 73–74
flottante, nombres à 73–74
vitesse de réponse, API 115

W

WHILE...DO, itération (ST) 179–180, 394
WRITE, bloc 110

X

XON/XOFF, protocole 103

Z

zone morte 358