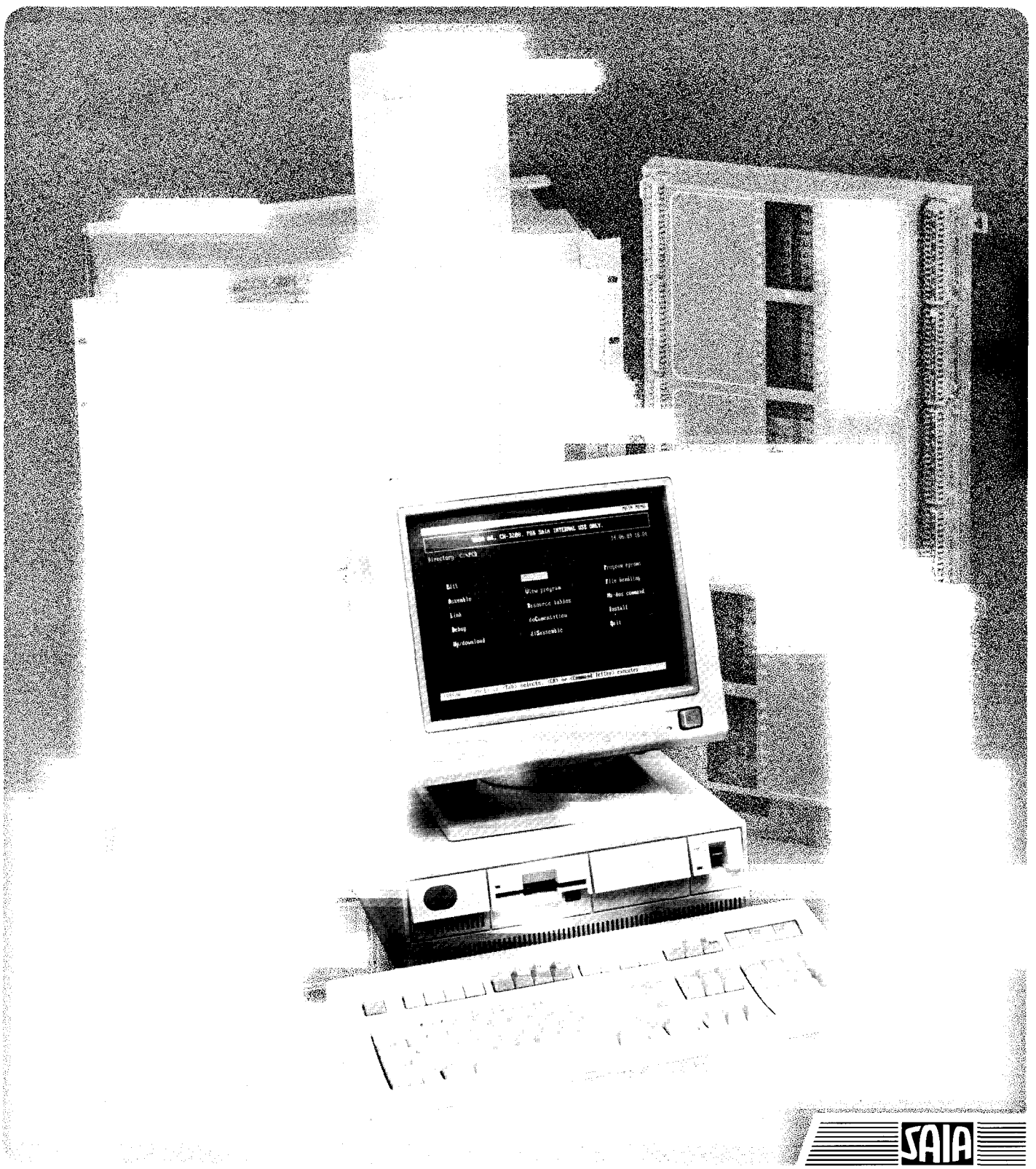
SAIA®PLC
Automates programmables

Manuel du logiciel niveau 1 H



Edition française 26/721 F1

Carrard P-A 304

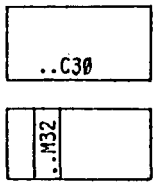
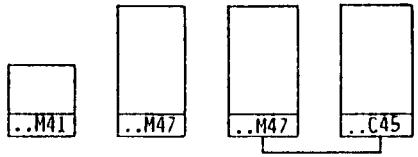
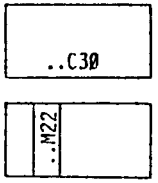
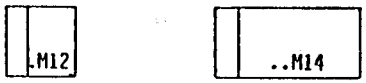
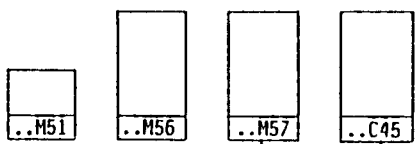
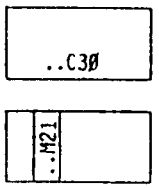
SOFTWARE NIVEAU 1H

TABLE DES MATIERES

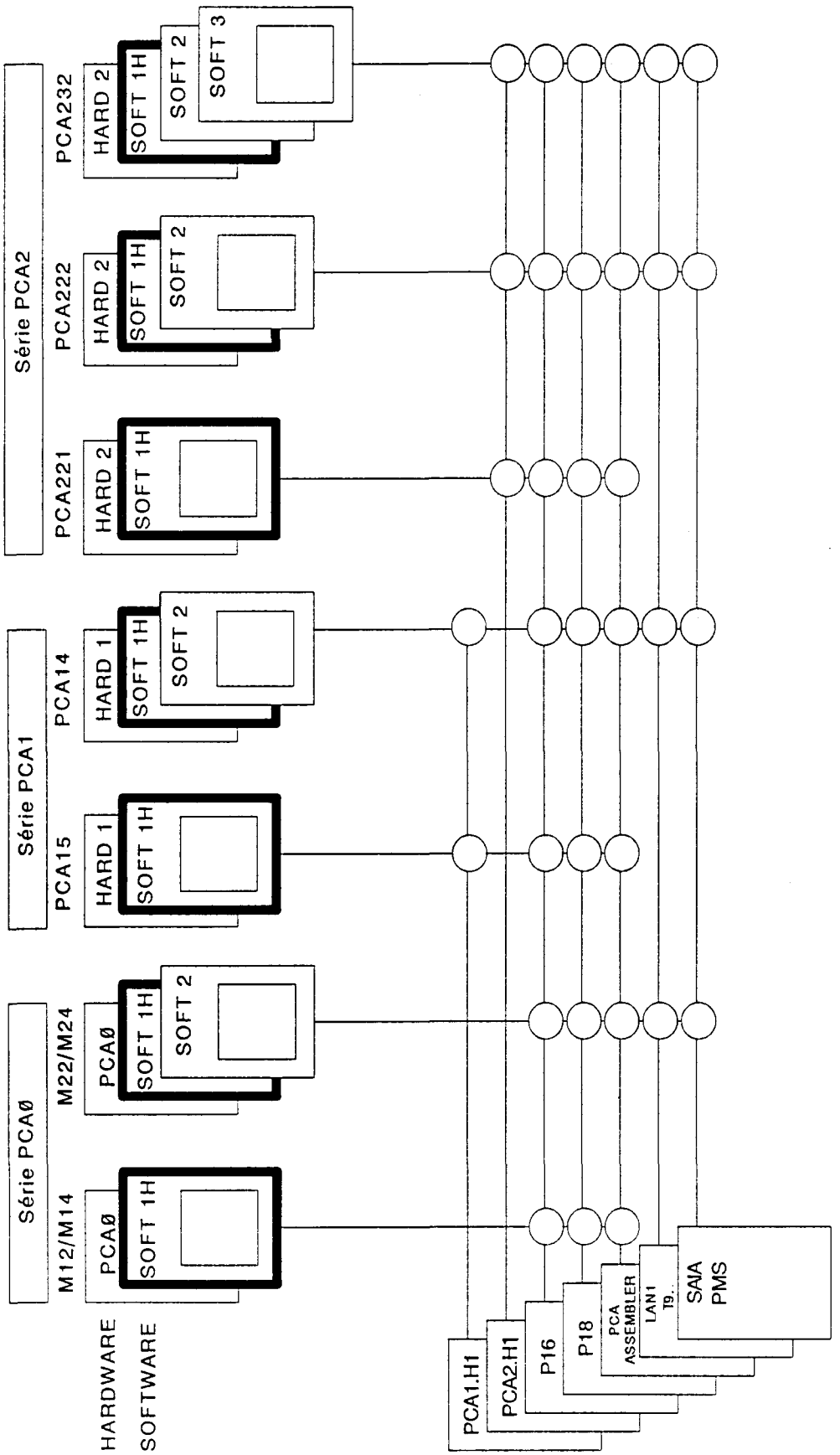
PARTIE D INTRODUCTION

Vue d'ensemble des automates programmables SAIA°PLC
Vue d'ensemble des manuels
Vue d'ensemble des structures de registres de la famille PCA
Jeu d'instructions du SAIA°PLC niveau de logiciel 1
Instructions supplémentaires pour niveau de logiciel 1H

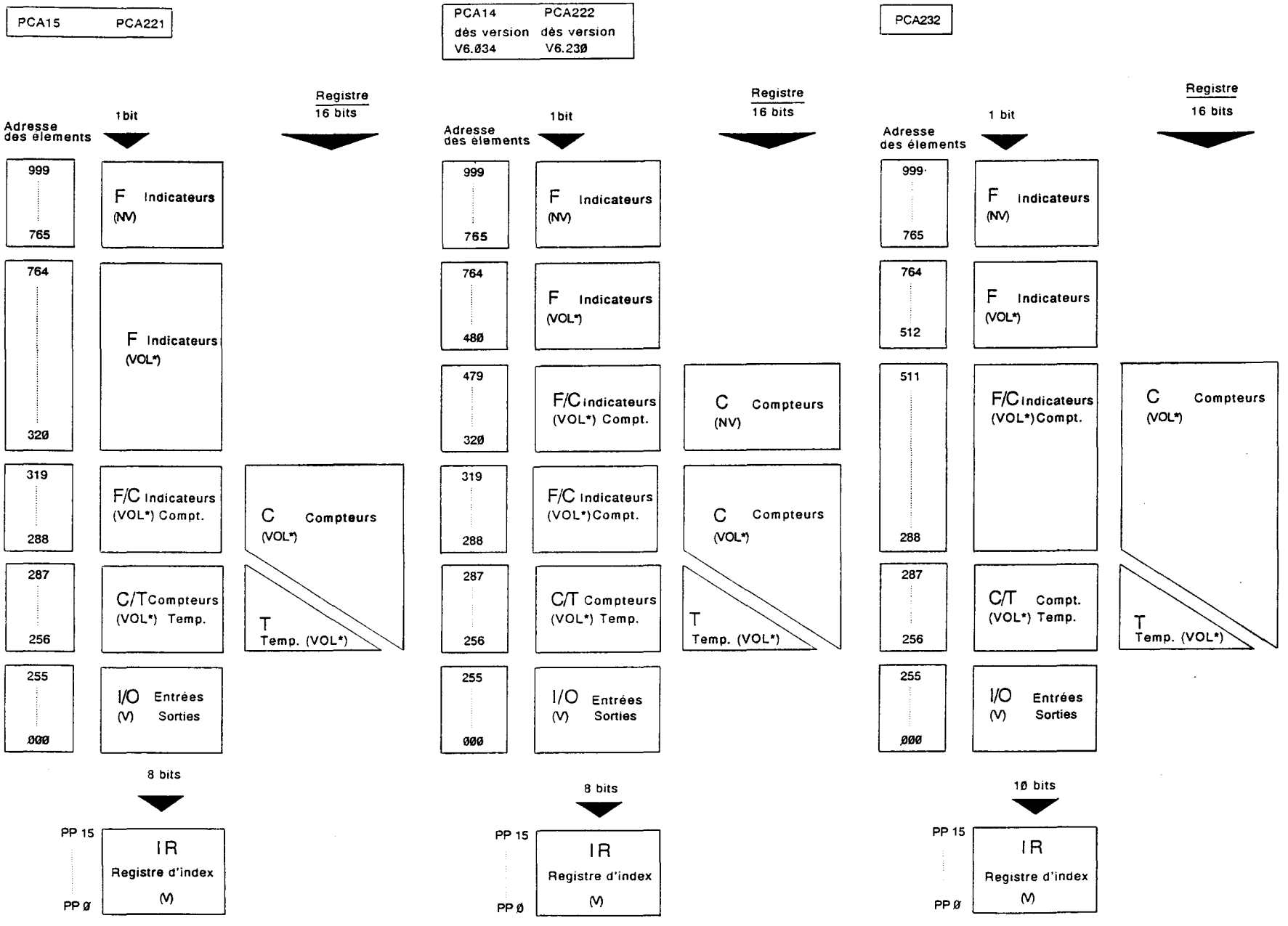
Système PCA

	Série PCAØ	Série PCA1	Série PCA2
Niveau du logiciel 3 Niveau du logiciel 2 + 32 instructions de mots pour - arithmétique, ±9 digits - transfert de données - registre de mots			PCA232 Mémoire utilisateur 8K lignes de programme + 8K caractères de texte + 8K byte de données  256 ou 512 E/S
Niveau du logiciel 2 Niveau du logiciel 1H + interface de données sérielles + horodateur + registre de données + instructions paramètriques (soft-interrupt, FIFO, PID)	Exécutions standard et OEM PCAØ.M22 — PCAØ.M24  max. 32 E/S — max. 64 E/S Mémoire utilisateur max. 4K lignes de programme max. 4K caractères de texte/données	PCA14 PCA141 — PCA147 — PCA147 + ..C45  36/56 — 64(112) — 128(224) E/S Mémoire utilisateur max. 8K lignes de programme max. 8K caractères de texte/données	PCA222  256 ou 512 E/S Mémoire utilisateur max. 8K lignes de programme max. 8K caractères de texte/données
Niveau du logiciel 1H 32 instructions de base pour - fonctions de temporisation et de comptage - programmes parallèles et sous-programme - indexation etc. 20 instructions addition. pour - arithmétique - transfert de données - check-sum	Exéxutions standard PCAØ.M12 — PCAØ.M14  24/32 E/S — 48/64 E/S Mémoire utilisateur max. 4K lignes de programme	PCA15 PCA151 — PCA156 — PCA157 + ..C45  36/56 — 64(112) — 128(224) E/S Mémoire utilisateur max. 4K lignes de programme	PCA221  256 ou 512 E/S Mémoire utilisateur max. 8K lignes de programme

Vue d'ensemble des manuels

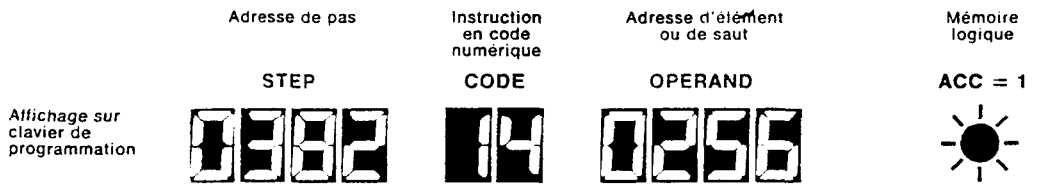


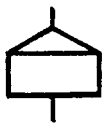
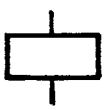
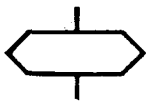
Vue d'ensemble des structures de registres de la famille PCA



*) Volatil, commutable "non-volatil" avec le pont NVOL.

Jeu d'instructions du SAIA°PLC niveau de logiciel ①



		Code numérique	Code mnémonique	Instruction en anglais	Description
	**Instructions de combinaison logique**	Ø1 Ø2	STH STL	Start High Start Low	Début d'une combinaison logique avec élément scruté sur High / Low
		Ø3 Ø4	ANH ANL	AND High AND Low	Fonction ET entre l'accumulateur et l'élément scruté sur High / Low
		Ø5 Ø6	ORH ORL	OR High OR Low	Fonction OU entre l'accumulateur et l'élément suivant scruté sur High / Low
		Ø7	XOR	Exclusive OR	Combinaison OU exclusif avec l'élément
		Ø8	NEG	Negate Accu	Complémente l'accu (résultat de la comb.)
		Ø9	DYN	Dynamic Control	Dynamise la combinaison logique (l'accu n'est influence qu'au 1er cycle)
	**Instructions de commutation**	1Ø	OUT	Set Output with Status of Accu	Positionne la sortie ou l'indicateur avec l'état de l'accu
		11	SEO	Set Output	Enclenche la sortie ou l'indic. avec mémorisat.
		12	REO	Reset Output	Déclenche la sortie ou l'indicateur
		13	COO	Complement Output	Scrute la sortie ou l'indicateur et le positionne dans l'état contraire
	Instructions de temporisation et de comptage	14*	STR*	Set Timer	Positionne le temporisateur sur la valeur choisie et le fait partir
		15*	SCR*	Set Counter	Positionne le compteur sur la valeur choisie
		17	INC	Increment Counter	Augmente / Diminue la position du compteur de 1
		18	DEC	Decrement Counter	
	**Instructions de saut**	2Ø	JMP	Unconditional Jump	Saute à une adresse de pas, sans condition
		21	JIO	Jump if Accu is One	Saute si accu = 1 / accu = Ø sur adresse de pas
		22	JIZ	Jump if Accu is Zero	
		23	JMS	Jump to Subroutine	Saute dans un sous-programme
		24	RET	Return from Subrout.	Revient d'un sous-programme
	Instructions d'attente	25	WIH	Wait if High	Attend aussi longtemps que High / Low l'élément est sur
		26	WIL	Wait if Low	
	**Instructions auxiliaires**	ØØ	NOP	No Operation	Sans effet
		19	SEA	Set Accu	Positionne l'accu = 1
		16	SEI	Set Index	Pos. le registre d'index sur la valeur choisie
		27	INI	Increment Index	Augmente / Diminue le registre d'index de 1
		28	DEI	Decrement Index	
		29**	PAS**	Program Assignment	Assigne un programme parallèle
		3Ø	DOP	Display Operand	Affiche un opérande choisi
		31	DTC	Display Timer or Counter	Affiche la position d'un temporisateur ou d'un compteur

* Instruction à 2 lignes (la deuxième contient la valeur choisie)

** Instruction à 2 lignes (la deuxième contient l'adresse de début du programme)

Instructions supplémentaires pour niveau de logiciel 1H

	Mnemo code	Num. code	Instruction en anglais	Description
Instructions de transfert	STR	14	Set Timer	
	SCR	15	Set Counter	
		19	2ème ligne	Lire 5 x 4 bits BCD
		20		Sortir 5 x 4 bits BCD
		21		Sortir 8 bits binaires
		22		Sortir 12 bits binaires
		23		Sortir 16 bits binaires
		24		Lire 8 bits binaires
		25		Lire 12 bits binaires
		26		Lire 16 bits binaires
		31		Trans. compteur -->compteur ou reg. d'index -->compteur
Instructions arithmétiques	SCR	15	Set Counter	
		27	2ème ligne	Additionne +
		28		Soustrais -
		29		Multiplie x
		30		Divise :
Instructions d'indexage	SEI	16	Set Index	Positionne registre d'index sur la valeur choisie
	INI	27	Increment-Index	Augmente le registre
	DEI	28	Decrement-Index	Diminue d'index de 1

	Mnemo code	Num. code	Operand	Description
Instructions spéciales (à deux ligne)	PAS	29	18	Modification du nombre de programmes parallèles actifs
	PAS	29	30...38	Check sum

Tel qu'on le voit dans les quelques pages précédentes, les instructions sont divisées en quatre parties distinctes toute compatibles vers le haut.

Les 32 instructions de base appartiennent au niveau logiciel (1), compatibles avec les systèmes PCA13 et PCA21Ø.

Aujourd'hui le logiciel de base s'appelle "Niveau software (1H)".
A ce paquet de base est ajouté 2Ø instructions supplémentaire déjà disponible sur les systèmes PCAØ, PCA15 et PCA221.

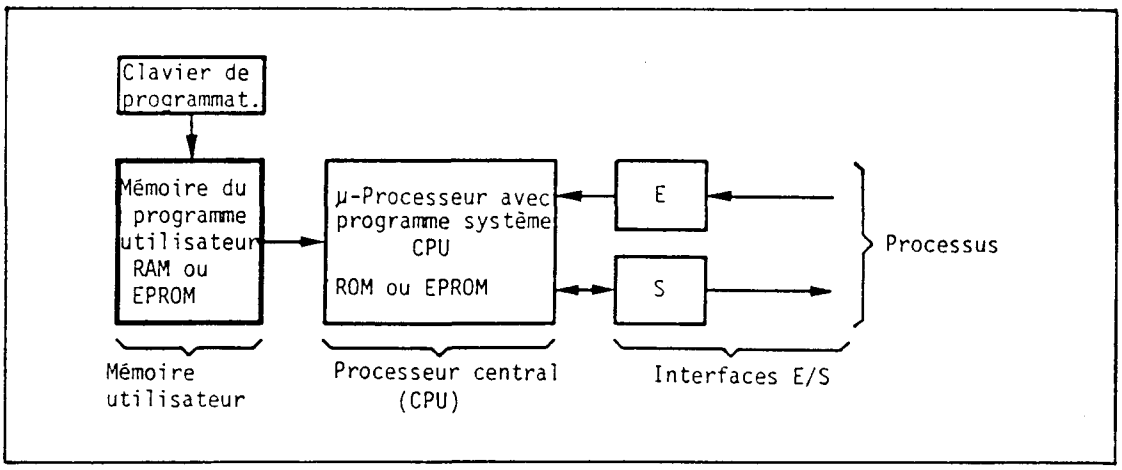
Ce niveau d'intelligence, le niveau 1H est le thème de ce manuel.

Avec seulement 5 instructions il est possible de réaliser un simple programme de scrutation mais en exploitant complètement les possibilités du niveau soft (1H) on accède aux opérations arithmétiques sur les registres de comptage, aux fonctions de temporisation, aux techniques des programmes parallèles et des sous-routines.

Les niveaux d'intélligence supérieurs propre aux systèmes PCA14, PCA222 et PCA232 contiennent tous ces fonctions de base, assurant ainsi la compatibilité vers le haut de la gamme SAIA°PLC.

PARTIE D Introduction

D 1 Généralités



Comme déjà décrit dans l'introduction de la partie "matériel", les caractéristiques du CPU sont déterminées par le logiciel du système µP. Le programme système détermine rigoureusement toutes les caractéristiques du PLC. L'utilisateur n'a pas accès au programme système. Le programme utilisateur permet l'adaptation individuelle aux différentes conditions du processus.

Le programme utilisateur sera écrit dans le langage SAIA°PLC orienté aux problèmes spécifiques et sera mémorisé dans la mémoire utilisateur. Pour la programmation, différentes unitées de programmation sont à disposition.

Le CPU (ou processeur) est à même de "lire" le programme utilisateur et d'éxecuter les instructions contenues, comme p.ex. scrutation des états d'entrée, combinaison et transmission des résultats aux sorties, contrôlant ainsi le processus à commander.

Suivant les besoins ou les préferences du programmeur, le programme sera developpé à partir de l'une ou l'autre description de programme:

- Schéma électrique á contact (schéma électrique de principe)
- Schéma logique (schéma des fonctions logiques)
- Plan de déroulement (diagramme de flux)
- Plan de fonction selon DIN 44719 ou Grafcet (contrôle de pas)

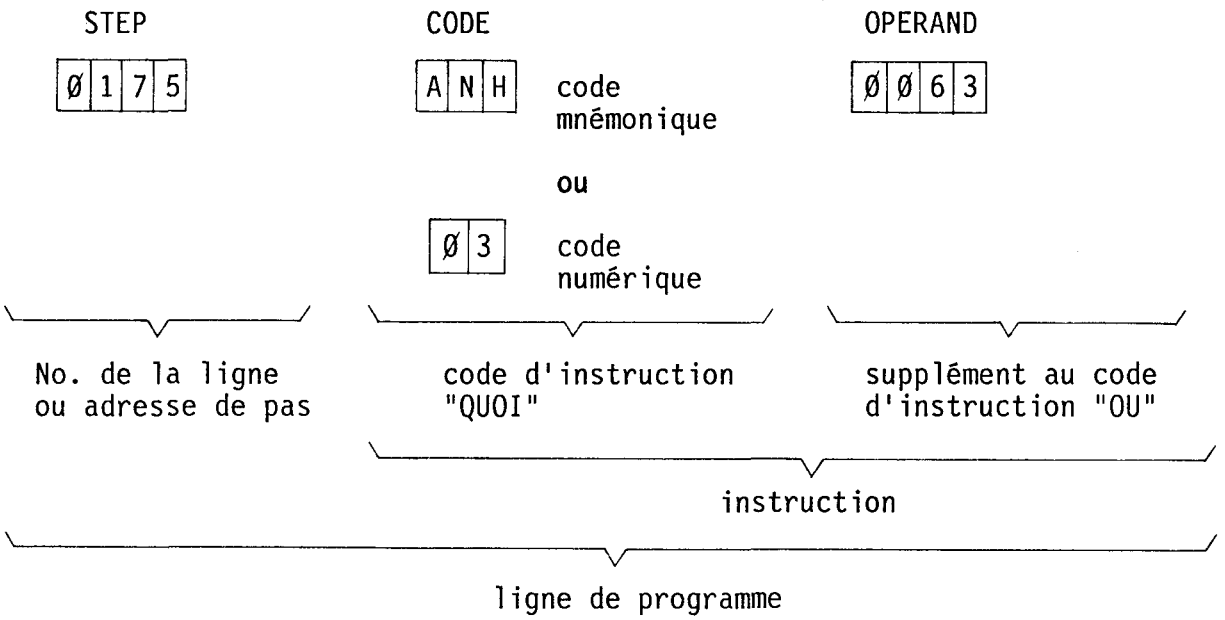
Les différentes méthodes de programmation peuvent aussi être combinées entre elles.

D 2 La ligne de programme

Chaque instruction du programme utilisateur est constituée d'une ligne de programme (dans certains cas 2 ou 1Ø lignes). La ligne contient en plus de son numéro ou adresse de pas (STEP), le code d'instruction (CODE) et l'opérand (OPERAND). Le code d'instruction définit "QUOI" instruction, l'opérand "OU" l'action est à exécuter.

Chaque ligne de programme contient 16 bit. 1K ≙ 1Ø24 lignes de programme qui peuvent être mémorisées dans une mémoire utilisateur de 16K bit (p.ex. dans une EPROM 2716).

Structure d'une ligne de programme resp. d'instruction de programme:



STEP La localisation de l'instruction dans la mémoire utilisateur est définie par le numéro de ligne. Numérotation décimale de Ø...2Ø47 (2K) ou 4Ø95 (4K).

CODE Selon l'unité de programmation le code d'instruction est introduit comme code mnémonique à 3 caractères ou code numérique de Ø...31. Le code mnémonique est une abréviation de l'expression anglaise pour l'instruction correspondante. Il est pour cela facile à rappeler et internationalement compréhensible.

OPERAND L'adresse d'un élément (entrée, sortie, temporisateur, compteur ou indicateur) ou l'adresse de déstination lors d'instructions de saut.

Les instructions de temporisation et de comptage sont constituées de 2 lignes de programme. La valeur du temporisateur ou compteur correspondant est affichée dans le champs d'opérand de la 2ème ligne.

D 3 Les opérands

Comme déjà défini auparavant, les opérands sont des adresses d'élément ou adresses de pas (numéros de ligne).

D 3.1 Adresses d'éléments

Tous les éléments adressables (entrées, sorties, compteurs, temporisateurs, indicateurs volatils et non-volatils) sont numérotés en décimales de Ø...999.

- Les entrées et les sorties sous forme de modules d'interface, sont adressées par leurs emplacements sur le module de base (PCAØ et PCA1) ou par un commutateurs DIL (PCA2).

  La plage d'adresse est assignée à loisir mais une adresse ne peut être qu'utilisée soit comme entrée ou comme sortie. (A l'exception du boîtier d'extension PCA2.C3Ø, qui permet d'atteindre 256 E + 256 S = 512 E + S).

  L'état des signaux d'entrées ne peut qu'être scruté.

  Les sorties peuvent être positionnées (mises) et repositionnées (remises) et de même leur état peut être scruté. Exception faite pour certain modules.

- Temporisateurs et compteurs sont des registres programmables. Côté hardware ils se trouvent dans le module CPU. Ils servent alternativement de temporisateur ou de compteur.

- Les indicateurs volatils et non-volatils sont des registres mémoires d'un bit qui peuvent être traités comme les sorties, c.-à-d. qu'ils peuvent être positionnés, remis et leur état peut être scruté. Les indicateurs volatils et non-volatils s'apprêtent donc parfaitement pour la mémorisation d'informations.

  Côté matériel, ils se trouvent sur le module CPU sous forme d'une mémoire RAM séparée. Pour que les 235 indicateurs non-volatils gardent leurs valeurs en cas de coupure de tension, ils sont munis d'une batterie tampon (la coupure de tension peut durer plus d'un mois). Les 477 indicateurs volatils sont aussi équipés de batteries-tampon, mais lors de la mise en service du PLC, ils sont remis à l'état "L". Ils réagissent comme indicateurs remis à la chute de tension.

  Par déplacement d'un pont les 712 indicateurs et tous les temporisateurs et compteurs peuvent être rendu non-volatils.

D 3.2 Adresses de pas

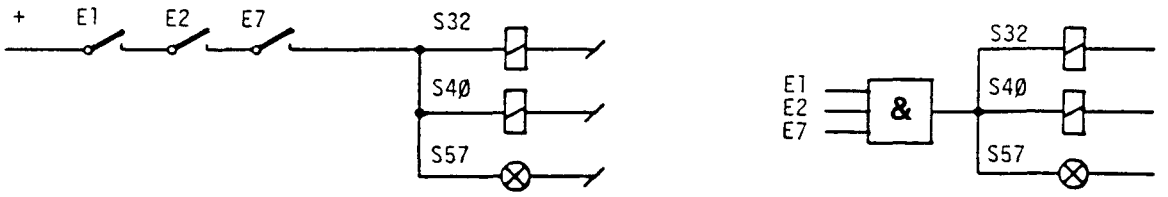
Le second type d'opérand est l'adresse de pas ou numéro de ligne. Celui-ci est rendu nécessaire par l'utilisation des instructions de saut. De ce fait on détermine à quelle adresse de destination du programme utilisateur un saut doit être effectué. Pour couvrir toute la mémoire utilisateur, toutes les adresses de pas de Ø...2Ø47 peuvent être utilisées en tant qu'opérand.

Dès le niveau (1H), le domaine adressable s'étend jusqu'à 8K. Ainsi, avec ce CPU d'intelligence supérieure il est possible de faire un saut jusqu'à l'adresse 8191 (instruction de deux lignes).

D 4 Quelques définitions

D 4.1 La ligne de combinaison

Exemple d'après schéma à contact ou schéma logique:

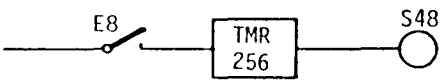


Programme

```
        .
        .
OUT 51
STH  1   ┐
ANH  2   │
ANH  7   │  Ligne de combinaison
OUT 32   │
OUT 40   │
OUT 57   ┘
STH  9
        .
        .
```

Une ligne de combinaison est une partie d'un programme et contient plusieurs lignes d'instruction. Elle est une combinaison d'éléments complète pour soi-même. Elle commence normalement avec une instruction de départ (STH ou STL). La combinaison est interprètée avec succès si le résultat final est "1", c.-à-d. si, dans notre exemple les sorties S32, S40 et S57 sont activées (positionnées).

Exemple avec temporisateur:



Programme

```
        .
        .
STH   8   ┐
STR 256   │  1ère ligne de
 00  50   ┘  combinaison
STH 256   ┐  2ème ligne de
OUT  48   ┘  combinaison
        .
        .
```

Comme le démontre le programme ci-dessus, la fonction comprend déjà 2 lignes de combinaison. A la 1ère ligne de combinaison l'entrée E8 est scrutée, puis selon son état logique le temporisateur est positionné (comme un relais de temporisation d'un circuit de commande. A la 2ème ligne de combinaison l'état de signalisation du temporisateur est transmis à la sortie S48 (comme par un circuit de charge d'un relais de temporisation).

D 4.2 L'accumulateur = ACCU

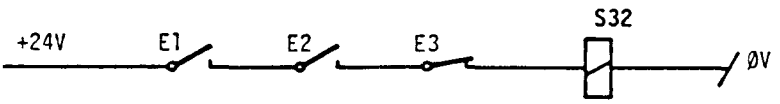
Cette mémoire se trouve sur le CPU et est constituée d'un registre 1 bit, qui peut prendre les états logiques 0 ou 1.

Pour exécuter une combinaison complète jusqu'à l'instruction suivante d'action, chaque résultat intermédiaire doit être mémorisé dans l'ACCU. Le résultat final de la combinaison est ensuite mémorisé dans l'ACCU (Ø ou 1).
Selon ce résultat final l'élément correspondant (p.ex. une sortie) n'est pas activé (ACCU = Ø) ou activé (ACCU = 1).

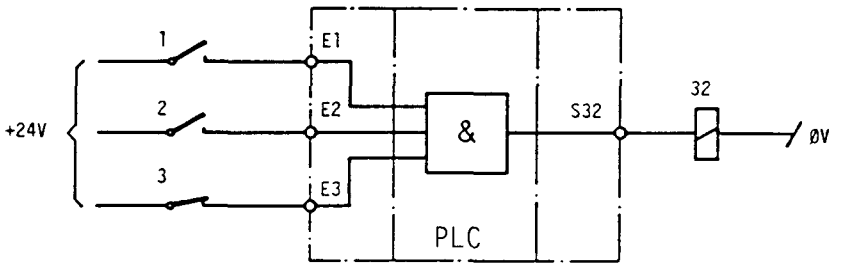
Le résultat mémorisé dans l'ACCU peut aussi servir pour sauter les instructions suivantes (voir par ex. instructions de saut).

D 4.3 Contact d'ouverture/contact de fermeture ou high/low

Une combinaison de contact comme le démontre le schéma ci-dessous, est exécuté comme câblage exactement d'après les symboles de contact:

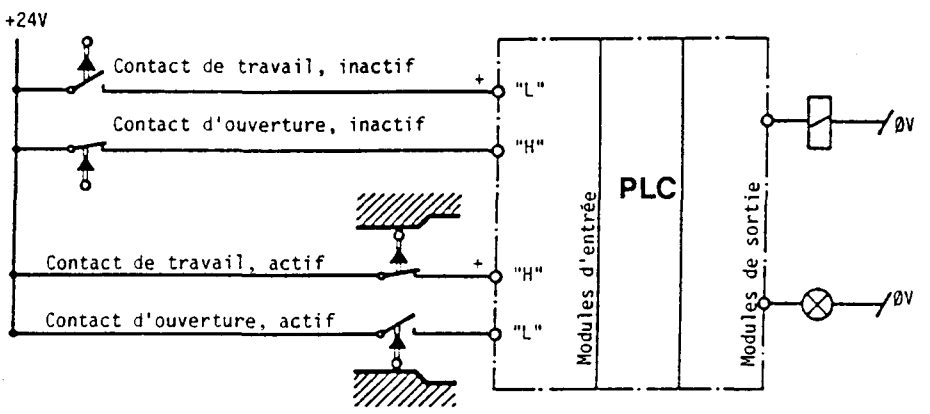


Pour résoudre ce problème à l'aide d'un PLC il faut se servir d'une entrée par contact. La combinaison est réalisée par le processeur du PLC et non par câblage. La présentation de la combinaison ci-dessus réalisable avec PLC est montré par la figure suivante.



Un PLC ne peut distinguer si le contact est un contact de fermeture ou d'ouverture reliés à ses entrées. Par contre, il distingue le signal HIGH (H) du signal LOW (L) présent á son entrée. Ces niveaux de signalisation sont exactement définis côté hardware pour chaque module d'entrée. Pour une entrée 24V en logique positive (standard) on peut dire en simplifiant:

- +24V à l'entrée ----------> état de signalisation "H" HIGH = HAUT
- ØV à l'entrée ----------> état de signalisation "L" LOW = BAS



La programmation du PLC peut-être faite selon une méthode souvent utilisée, le schéma à contact, en respectant les règles suivantes. Pour logique positive, c.-à-d. contact relié au +24V:

- Si un contact de fermeture sert de base de combinaison, le contact en question doit être scruté ou combiné d'après high (cas normal).

- Si un contact d'ouverture sert de base de combinaison, le contact en question doit être scruté ou combiné d'après low.

Ceci est valable pour contacts mécaniques comme pour détecteurs de proximité et cellules photo-électriques.

On a pour cela tendance à n'utiliser que des contacts de fermeture. Ceci est correct pour 90% des cas. Pour des raisons de sécurité il est en quelques cas absolument indispensable d'utiliser des contacts d'ouverture (voir aussi la partie EM1 de la bibliothèque de logiciel). En simplifiant on peut dire:

- Une procédure est initialisée par enclenchement de tension et arrêtée par déclenchement.

Respectant les réflexions précedentes, les instructions de scrutation et de combinaison apparaîssent toujours en paires:

STH	STL
ANH	ANL
ORH	ORL
(WIH)	(WIL)
(JIO)	(JIZ)

Ceci signifie que les signaux d'entrée peuvent être scrutés d'après "H" ou "L". Si le mode de scrutation correspond à l'état réel de signalisation d'entrée, le résultat est reconnu comme "1" logique de l'ACCU et le processus continue.

Les états de signalisation sont valables non seulement pour des entrées, mais aussi pour tous les élements adressables:

Entrée	"H":	+24V présent (standard, logique positive)
	"L":	0V présent (standard, logique positive)
Sorties	"H":	Sorties positionnées = transistor débloqué
	"L":	Sorties remises = transistor bloqué
Indicateur volatil/	"H":	Positionné
non-volatil	"L":	Non positionné ou remis
Temporisateur	"H":	Positionné et se déroulant
	"L":	Remis ou arrivé à la fin de temporisation
Compteur	"H":	Compteur positionné et contenu > 0
	"L":	Compteur non positionné ou contenu = 0

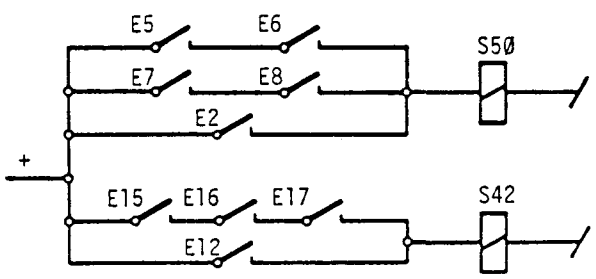
D 5 Méthodes de programmation

L'automate programmable SAIA°PLC offre une grande fléxibilité au développement et à l'adaptation d'un programme. Tout comme pour la commande traditionnelle, le constructeur doit établir tout d'abord un cahier des charges clair du problème de commande. Suivant le processus et ses préférences le constructeur utilisera différentes méthodes de description.

Grâce au jeux d'instruction très large du SAIA°PLC, les descriptions les plus différentes sont utilisables comme base d'un programme PLC.

D 5.1 Programmation d'après schéma de contact (schéma electrique de principe)

Plan:



Programme (liste d'instruction):

ADDR	NC	MNC	OPRD
10	01	STH	5
11	03	ANH	6
12	05	ORH	7
13	03	ANH	8
14	05	ORH	2
15	10	OUT	50
16	01	STH	15
17	03	ANH	16
18	03	ANH	17
19	05	ORH	12
20	10	OUT	42
21	20	JMP	10

Caractéristiques:

----> Instructions nécessaires STH, STL, ANH, ANL, ORH, ORL, OUT.

- La combinaison est représentée par des symboles de contact. Les sorties utilisent les résultats des combinaisons précédentes ET/OU.
- Toutes les parties de programme sont parcourues cycliquement. Un tel programme est appelé boucle de programme pure, car aucune instruction d'attente ni instruction de saut conditionnel n'est contenue.

Avantages:

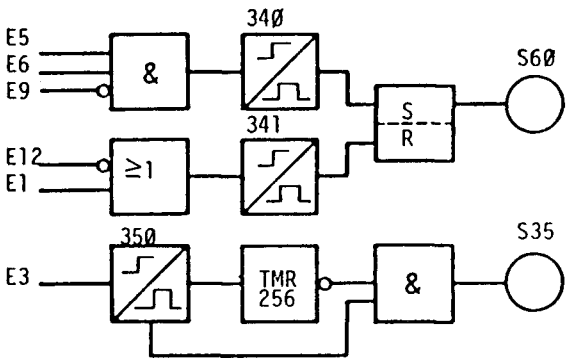
- Programmation simple avec seulement 7 instructions, pour amateur des symboles de contact.
- Bonne application pour contrôle.
- Différentes lignes de combinaison peuvent être alignées à désir.

Désavantages:

- Pas pratique pour processus séquentiel (contrôle séquentiel), parce que les parties non actives doivent être bloquées.
- Les contrôles plus complèxes deviennent difficiles et longs à l'interprétation des symboles de contact.
- Jeu d'instruction limité.
- Temps de réaction long lors de programmes longs.

D 5.2 Programmation d'après symboles de fonction (schéma des fonctions logiques)

Plan:



Programme (liste d'instruction):

ADDR	NC	MNC	OPRD	
30	01	STH	5	
31	03	ANH	6	
32	04	ANL	9	
33	09	DYN	340	
34	11	SEO	60	RS
35	02	STL	12	
36	05	ORH	1	
37	09	DYN	341	
38	12	REO	60	
39	01	STH	3	
40	09	DYN	350	
41	14	STR	256	
42	00	00	20	Timer
43	02	STL	256	
44	03	ANH	350	
45	10	OUT	35	
46	20	JMP	30	

Caractéristiques:

-----> Instructions supplémentaires SEO, REO, COO, STR, SCR, NEG, DYN etc.

- La combinaison est éffectuée au travers de symboles logiques orientés d'après leurs fonctions, ceci permettant de façon similaire l'interprétation du flux des signaux comme avec les symboles de contact.

- Toutes les parties du programme sont parcourues cycliquement en permanence (boucle de programme pure).

Avantages:

- Les sorties peuvent être positionnées avec mémorisation ou non.

- Très pratique pour contrôle et programme logique pur.

- Grâce á l'utilisation d'une grande partie du jeu d'instruction les problèmes les plus complèxes restent facile à interpréter.

- L'alignement de différentes lignes de combinaison laisse une certaine liberté.

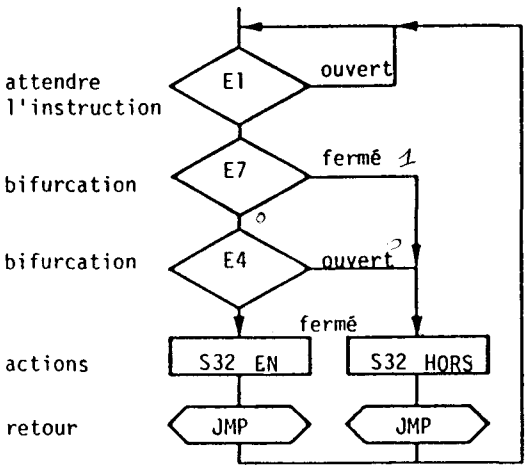
Désavantages:

- Pas pratique pour processus séquentiel (contrôle séquentiel), parce que les parties non actives doivent être bloquées.

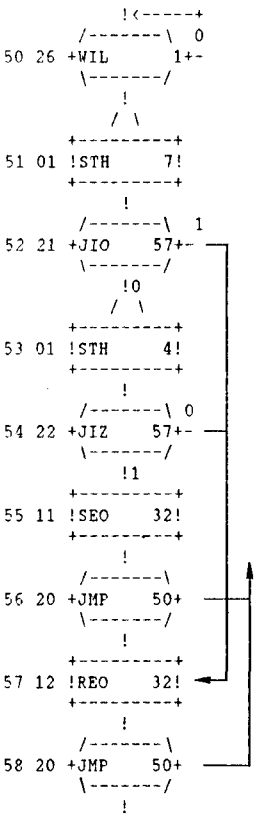
- Temps de réaction long lors de programmes longs.

D 5.3 Programmation d'après diagramme de flux (plan de déroulement)

Plan:



Programme sous forme de diagramme de flux:



Programme sous forme de liste d'instruction:

ADDR	NC	MNC	OPRD
50	26	WIL	1
51	01	STH	7
52	21	JIO	57
53	01	STH	4
54	22	JIZ	57
55	11	SEO	32
56	20	JMP	50
57	12	REO	32
58	20	JMP	50

Caractéristiques:

------> Instruction spéciales WIH, WIL, JIO, JIZ

- Il est possible de programmer des boucles d'attente, qui retiennent le processeur dans cette boucle jusqu'au moment où les conditions de continuation sont remplies.
- Bifurcations par sauts conditionnels sont possibles.
- Très pratique pour processus séquentiel.

Avantages:

- Programmation simple, directement d'après plan de déroulement du processus séquentiel.
- Cette méthode de programmation est très bien comprise par les ingénieurs de processus.
- Grâce aux boucles d'attente le programme ne se déroule qu'à la vitesse du processus même. Il en résulte une mise en service et la recherche d'erreurs rapide.
- Temps de réaction extrêmement courts. Ils ne sont pratiquement défini que par les délais des interfaces d'entrée.

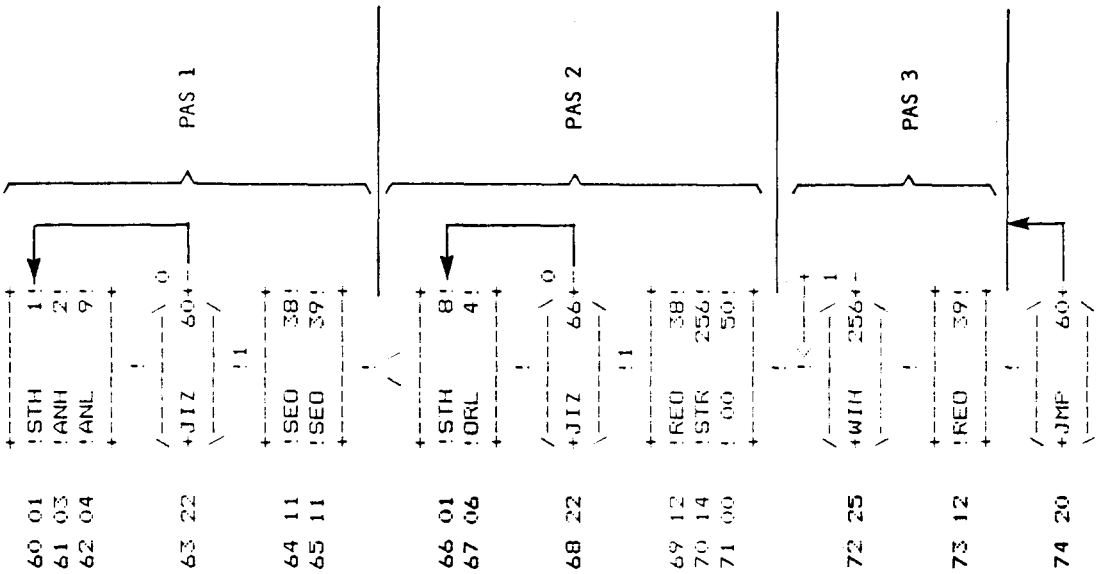
Désavantages:

- Les fonctions de contrôle sont à effectuer séparément dans des programmes parallèles.

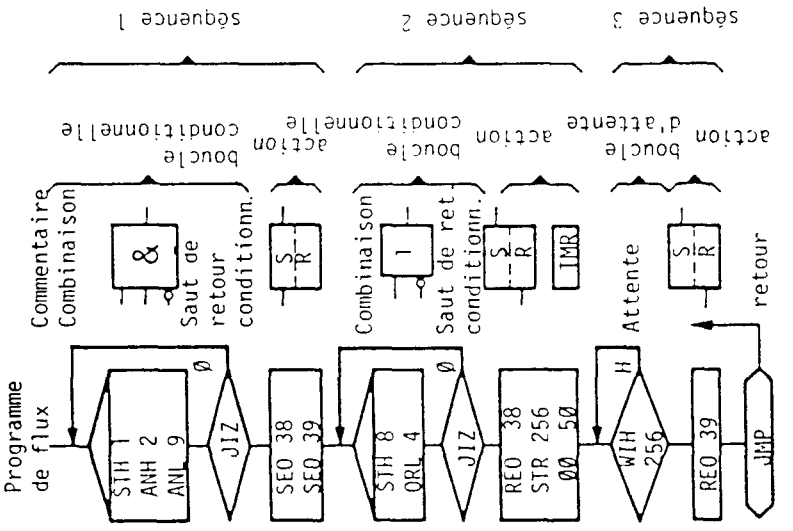
D 5.4 Programmation combinée d'après diagramme de flux/symboles logiques resp. plan de fonctions selon Grafcet

Le jeu d'instruction universel du SAIA°PLC permet, par application combinée des instructions, l'établissement de pas de programme très pratique.

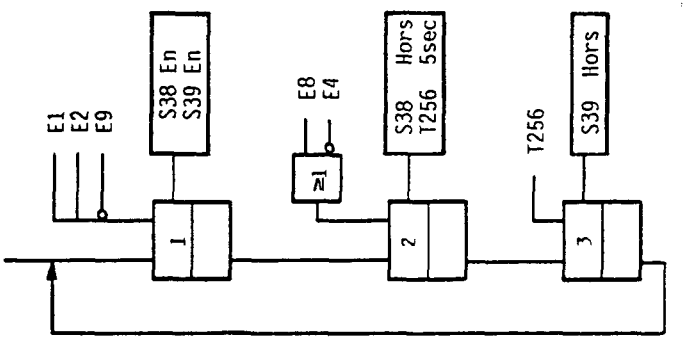
Documentation d'après diagramme de flux



Plan d'après diagramme de flux



Plan de fonction selon DIN ou Grafcet

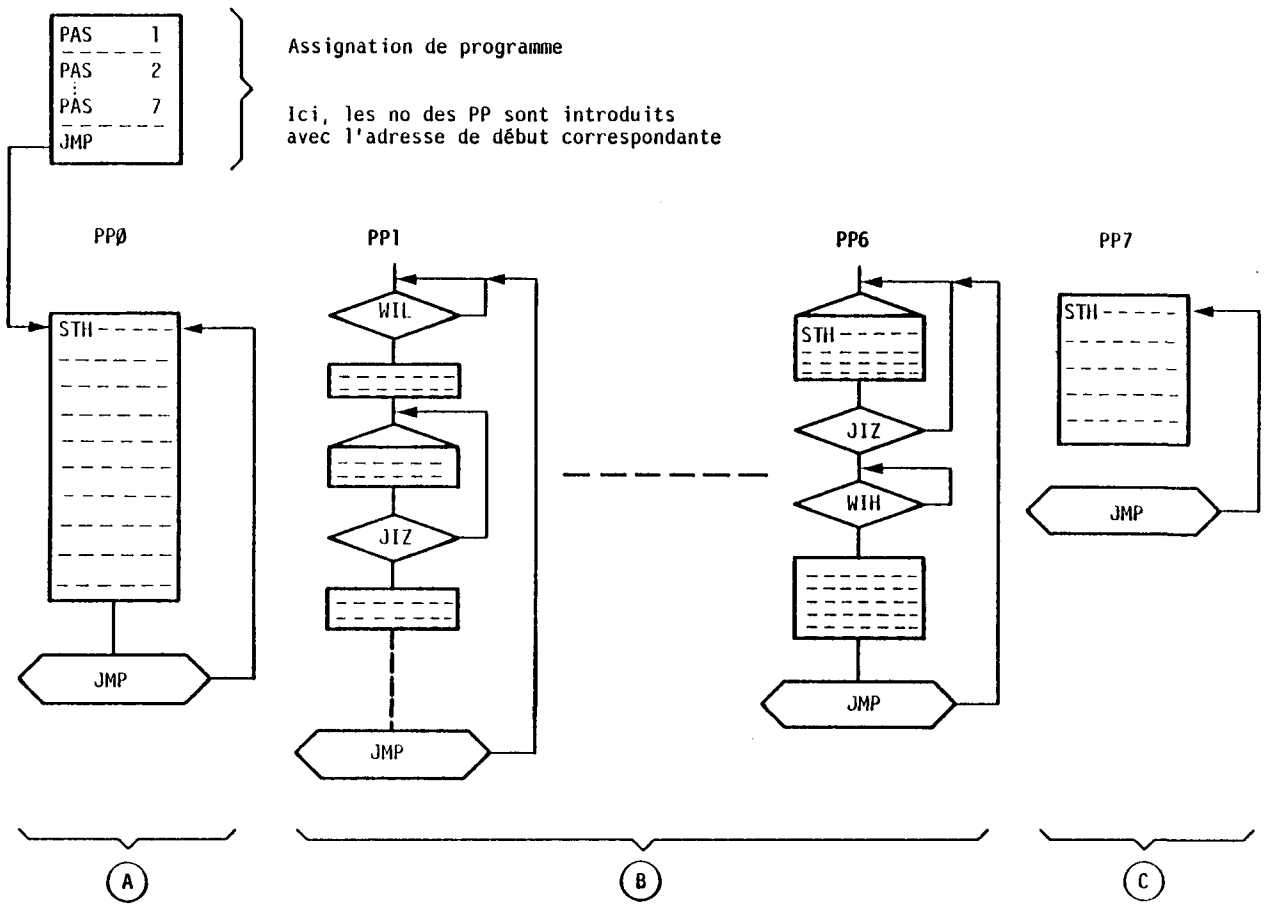


Caractéristiques: Les instructions logiques permettent une plus grande fléxibilté.

D 5.5 Programmation avec programmes parallèles

Des conditions optimales sont obtenues par l'utilisation de programmes parallèles. Jusqu'à 16 programmes parallèles, indépendants les uns des autres et se déroulant asynchronement, peuvent être utilisés. Ceci signifie que différents processus séquentiels d'une machine (p.ex. automate de montage) sont déposés dans différents programmes selon diagramme de flux et que ceux-ci sont parcourus en pas à pas en accord avec l'avance du processus.
Les contrôles et les fonctions en activité permanente sont introduites dans un programme parallèle cyclique.

Structure de programme:



A : PPØ (programme parallèle Ø) est une boucle de programme cyclique (sans boucle d'attente) avec contrôle et fonctions en activité permanente.

B : PP1 jusqu'à (p.ex.) PP6 sont des programmes parallèles séquentiels du diagramme de flux pour différentes fonctions se déroulant asynchrone.

C : PP7 est une boucle de programme cyclique brève, contenant quelques fonctions, qui demandent un temps de réaction très court.

Caractéristiques:

-----> Instruction PAS pour assignation de programme.
Autres caractéristiques voir structure de programme.

Avantages:

- Les programmes séquentiels (avec boucles d'attente) et fonctions en activité permanente (contrôles, logique pure etc.) peuvent être programmés séparément et selon les désirs du programmeur.

- Les programmes parallèles nécessaire sont assignés (max. 16).

- Les programmes parallèles peuvent être bloqués et débloqués à tout moment par l'interrupt du logiciel correspondant (instruction PAS 18).

Désavantages:

- Pour obtenir des temps de réaction rapides pour les fonctions en activité permanente, celles-ci doivent être traitées dans un programme court séparé.

D 5.6 Programmation avec sous-programmes

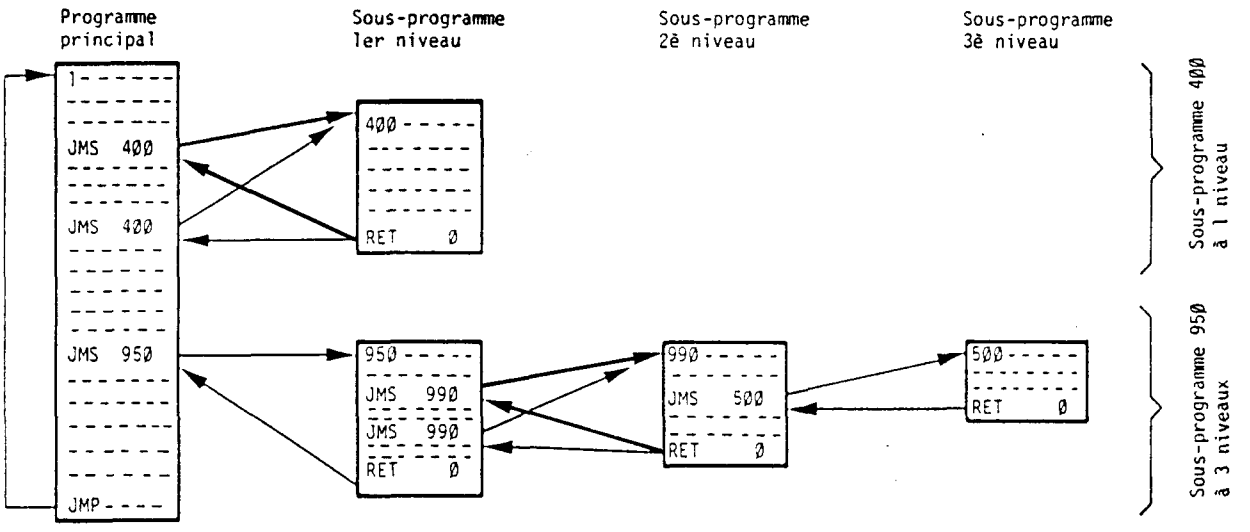
-----> Instructions nécessaires JMS et RET.

Les parties de programme á répétition fréquente sont avantageusement programmées sous forme de sous-programmes.

Des sous-programmes peuvent être utilisés avec toutes les techniques de programmation décrites auparavant, c.-à-d. schéma à contacts, schéma de fonction, diagramme de flux ou sous forme de programmes parallèles.

Les sous-programmes économisent l'espace de mémoire ainsi que le temps de programmation. Par la technique des sous-programmes les programmes sont divisés en blocs de fonctions faciles à interpréter (programmation structurée).

Le nombre de sous-programmes adressables est illimité, avec jusqu'à trois niveaux accessibles. Il faut éviter que le même sous-programme soit appelé par différents programmes parallèles.



D 5.7 Indéxage d'adresses (traitement en série)

-----> Instructions SEI, INI, DEI

Le traitement de plusieurs éléments (entrées, sorties, indicateurs ou temporisateurs) (p.ex. fonctions de contrôle ou régistres à décalage) donne normalement des programmes longs de combinaison peu profondes.
La remise de tous les indicateurs non-volatils par l'action d'une touche nécessiterait 236 lignes d'instruction. Le même problème est resolu avec uniquement 5 lignes de programme par indéxage d'adresse.

Sans indéxage:

```
┌─►STH     1
│  REO   765
│  REO   766
│  REO   767
│  REO   768
│  REO   769
│  REO   77Ø
│    :
│    :
│  REO   997
│  REO   998
│  REO   999
└──JMP
```

Avec indéxage:

```
┌──►SEI    Ø
│  ┌►STH    1
│  │ REO 1765
│  │ INI  234
│  └─JIO
└────JMP
```

La boucle d'indéxage est parcourue 1 + 234 fois, l'adresse d'élément précédée de 1ØØØ) incrémentée de 1 à chaque tour.

Notes:

PARTIE E JEU D'INSTRUCTIONS ET PROGRAMMATION

E 1 Instructions de scrutation et de combinaison
- STH, STL
- ANH, ANL
- ORH, ORL
- XOR
- NEG
- DYN

E 2 Instructions de commutation
- OUT
- SEO, REO
- COO

E 3 Instructions de temporisation et de comptage
E 3.1 Instructions de transfert et d'opérations arithmétiques
E 3.2 Temporisateurs et compteurs
- STR
- SCR
- INC, DEC

E 3.3 Instructions de transfert de bits BCD et binaires
E 3.4 Introduction externes de valeurs BCD
E 3.5 Fonctions additionnels STR, SCR
E 3.6 Codes pour les opérations arithmétiques
E 3.7 Resumé des instructions pour les registres de temporisation et de comptage

E 4 Instructions de saut et d'attente
JMP saut inconditionnel
JIO, JIZ saut conditionnel
JMS, RET saut et retour d'un sous-programme
Fonctions additionnelles JMP, JIO,
JIZ, JMS pour adresses 2048...8191
WIH, WIL instructions d'attente

E 5 Instructions auxiliaires
- NOP No operation
- SEA Positionner l'accumulateur à 1

E 6 Indexation
- SEI, INI, DEI
Indexage, fonctions de base
- SEI, INI, DEI
Indexage, fonctions additionnelles

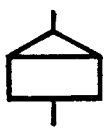
E 7 Instructions PAS
- PAS Assignation de programmes parallèles
- PAS 18 Limitation du nombre de programme parallèles travaillés
- PAS 30...38 "Check-Sum" du programme utilisateur et système

E 8 Instructions d'affichage
- DOP Affichage d'un opérand
- DTC Affichage de la valeur de temporisation ou de comptage

PARTIE E Jeu d'instructions et programmation

E 1 Instructions de scrutation et de combinaisons

	Adresse de pas	Instruction en code numérique	Adresse d'élément ou de saut	Mémoire logique
	STEP	CODE	OPERAND	ACC = 1
Affichage sur clavier de programmation	0382	14	0256	☀



	Code numérique	Code mnémonique	Instruction en anglais		Description	
Instructions de combinaison logique	Ø1	STH	Start High	\	Début d'une combinaison logique	/ High
	Ø2	STL	Start Low	/	avec élément scruté sur	\ Low
	Ø3	ANH	AND High	\	Fonction ET entre l'accumulateur	/ High
	Ø4	ANL	AND Low	/	et l'élément scruté sur	\ Low
	Ø5	ORH	OR High	\	Fonction OU entre l'accumulateur	/ High
	Ø6	ORL	OR Low	/	et l'élément suivant scruté sur	\ Low
	Ø7	XOR	Exclusive OR		Combinaison OU exclusif avec l'élément	
	Ø8	NEG	Negate Accu		Complémente l'accu (résultat de la comb.)	
	Ø9	DYN	Dynamic Control		Dynamise la combinaison logique (l'accu n'est influence qu'au 1er cycle)	

STH
STL

Instruction de départ et de scrutation

STH: Start High ---> Départ avec scrutation de l'élément d'après "H"
STL: Start Low ---> Départ avec scrutation de l'élément d'après "L"

Format de l'instruction:

Code d'instruction		Opérand	
Code mnémonique	Code numérique	Description	Plage
STH	Ø1	Adresse d'élément	Ø...999 (i)
STL	Ø2	Adresse d'élément	Ø...999 (i)

(i) = indéxable

Chaque ligne de combinaison débute avec une instruction de départ

Les résultats de combinaison précédentes sont éffacés par l'instruction de départ. En même temps l'état de signalisation de l'élément adressé E, S, T, C, I est scruté d'après "H" ou "L" et le résultat est mémorisé dans l'ACCU.

STH: Départ de combinaison et scrutation de l'élément adressé d'après "H"
au signal = H ---> ACCU = 1
au signal = L ---> ACCU = Ø

STL: Départ de combinaison et scrutation de l'élément adressé d'après "L"
au signal = L ---> ACCU = 1
au signal = H ---> ACCU = Ø

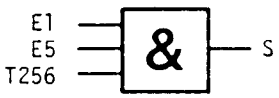
Table de vérité:

Instruction	Etat logique de l'élément	Contenu de l'ACCU après exécution de l'instruction
STH	H L	1 Ø
STL	L H	1 Ø

En général: Si l'état de l'élément correspond à l'état scruté, le résultat sera 1 dans l'ACCU.

Chapitre F
Tous les exemples

S = E1 · E5 · T256



ANH
ANL

Combinaisons ET

ANH: AND High ---> Combinaison et de l'ACCU avec scrutation de l'élément d'après "H"

ANL: AND Low ---> Combinaison et de l'ACCU avec scrutation de l'élément d'après "L"

Format de l'instruction:

Code d'instruction		Opérand	
Code mnémonique	Code numérique	Description	Plage
ANH	Ø3	Adresse d'élément	Ø...999 (i)
ANL	Ø4	Adresse d'élément	Ø...999 (i)

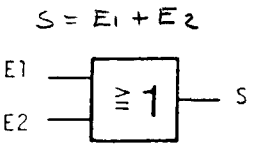
(i) = indéxable

L'état logique de l'élément adressé est scruté par l'instruction ET, et le résultat logique de la scrutation est combiné avec le contenu de l'ACCU.

Table de combinaison:

Instruction	Etat logique de l'élément	Contenu de l'ACCU avant exécution de l'instruction	Contenu de l'ACCU après exécution de l'instruction
ANH	H H	1 Ø	1 Ø
	L L	1 Ø	Ø Ø
ANL	L L	1 Ø	1 Ø
	H H	1 Ø	Ø Ø

Chapitre F
Exemples: 1, 2 et 4



ORH
ORL Combinaison OU

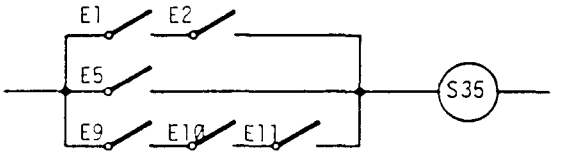
ORH: OR High ---> Combinaison OU de l'ACCU avec l'élément scruté d'après "H"
ORL: OR Low ---> Combinaison OU de l'ACCU avec l'élément scruté d'après "L"

Format de l'instruction:

Code d'instruction		Opérand	
Code mnémonique	Code numérique	Description	Plage
ORH	Ø5	Adresse d'élément	Ø...999 (i)
ORL	Ø6	Adresse d'élément	Ø...999 (i)

(i) = indéxable

Des éléments particuliers ou des combinaisons partielles sont éffectuées en parallèle par l'instruction OU (ORH/ORL). La scrutation H/L se refère toujours au premier élément de la combinaison partielle.




```
STH  1   } 1ère combinaison partielle
ANH  2   }
ORH  5   } 2ème combinaison partielle
ORH  9   }
ANH  1Ø  } 3ème combinaison partielle
ANH  11  }
OUT  35
```

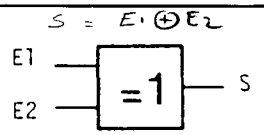
La combinaison entière est initialisée par l'instruction de départ (STH ou STL). Toute combinaison parallèle suivante est initialisée par l'instruction OU (ORH ou ORL).

Les états logiques des combinaison parallèles suivantes n'ont plus d'influence sur le résultat de la combinaison entière lorsqu'une des combinaisons partielles est remplie.
L'exemple ci-dessus montre que l'instruction OU est prioritaire à l'instruction ET.

Table de combinaison:

Instruction	Résultat de la combinaison partielle		ACCU, après exécution de l'instruction
	première	suivante	
OR	1 1	1 Ø	1 1
	Ø Ø	1 Ø	1 Ø

Chapitre F
Exemples: 1, 2



XOR Combinaison OU exclusif

XOR: Exclusive OR ---> combinaison OU exclusif de l'ACCU avec l'élément adressé

Format de l'instruction:

Code d'instruction		Opérand	
Code mnémonique	Code numérique	Description	Plage
XOR	Ø7	Adresse d'élément à comparer	Ø...999 (i)

(i) = indéxable

L'instruction XOR permet la comparaison de l'état de signalisation de deux éléments (ou l'ACCU avec un élément). S'ils coïncident, le contenu de l'ACCU est Ø; s'ils ne coïncident pas, le contenu de l'ACCU est 1. La comparaison de l'élément adressé, expliqué plus précisement, est éffectuée avec le contenu de l'ACCU selon la table de combinaison.

Table de combinaison:

Instruction	Etat de signalisation	Contenu de l'ACCU avant exécution de l'instruction	Contenu de l'ACCU après exécution de l'instruction
XOR	H	1	Ø
	H	Ø	1
	L	1	1
	L	Ø	Ø

Chapitre F
Exemple 3

NEG Negation de l'ACCU

NEG: Negate ACCU ---> complémente le contenu de l'ACCU

Format de l'instruction:

Code d'instruction		Opérand
Code mnémonique	Code numérique	
NEG	Ø8	Ø

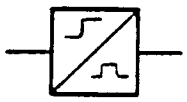
Table de vérité:

Instruction	Contenu de l'ACCU	
	avant exécution de l'instr.	après
NEG	1	Ø
	Ø	1

NEG complémente le contenu de l'ACCU.

Chapitre F
Exemple 3

M



DYN Contrôle dynamique

DYN: Dynamic control ---> déclenchement par le front ou dynamisation d'une signalisation ou d'une combinaison

Format de l'instruction:

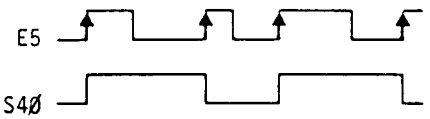
Code d'instruction		Opérand	
Code mnémonique	Code numérique	Description	Plage
DYN	Ø9	Adresse de l'indicateur de flanc	288...999 (i)

(i) = indéxable

Cette instruction provoque le déclenchement d'une impulsion de commande pour un élément donné, lorsqu'une combinaison a été préalablement réalisée. Lorsque le signal reste présent très longtemps, seul le front ascendant sera considéré. Autrement dit: lorsque le contenu de l'accumulateur change de Ø à 1, ce nouvel état ne persiste, après une instruction DYN que pendant le premier cycle du programme. Pour les cycles suivants, à cet endroit, l'accumulateur sera Ø.

Une instruction DYN nécessite toujours l'affectation d'un indicateur pour mémoriser le résultat de la combinaison. Aux cycles suivants si l'indicateur et positionné, les actions correspondantes ne sont plus exécutées (le contenu de l'accumulateur reste zéro même si la combinaison reste réalisée). De ce fait seul le front ascendant est effectif.
La fonction de l'instruction DYN est aisément démontrée par l'exemple d'un commutateur pas à pas avec en alternative le programme correspondant.

Commutateur pas à pas



Avec l'instruction DYN:

```
STH      5
DYN      5ØØ
COO      4Ø
```

Programme sans le DYN:

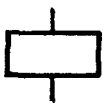
```
STH      5
ANL      5ØØ
SEO      5ØØ
COO      4Ø
STL      5
REO      5ØØ
```

DYN un signal d'entrée seulement.

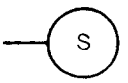
Chapitre F
Exemples: 6, 8, 9

E 2 Instructions de commutation

	Adresse de pas	Instruction en code numérique	Adresse d'élément ou de saut	Mémoire logique
	STEP	CODE	OPERAND	ACC = 1
Affichage sur clavier de programmation	0382	14	0256	

		Code numérique	Code mnémonique	Instruction en anglais	Description
	**Instructions de commutation**	1Ø	OUT	Set Output with Status of Accu	Positionne la sortie ou l'indicateur avec l'état de l'accu
		11	SEO	Set Output	Enclenche la sortie ou l'indic. avec mémorisat.
		12	REO	Reset Output	Déclenche la sortie ou l'indicateur
		13	COO	Complement Output	Scrute la sortie ou l'indicateur et le positionne dans l'état contraire

OUT Positionner la sortie selon le contenu de l'ACCU



OUT: Set output with status of ACCU
---> positionner la sortie ou l'indicateur selon contenu de l'ACCU

Format de l'instruction:

Code d'instruction		Opérand	
Code mnémonique	Code numérique	Description	Plage
OUT	1Ø	Adresse de la sortie ou de l'indicateur	Ø...255 (i) 288...999 (i)

(i) = indéxable

Avec l'instruction OUT le résultat (variable) de la combinaison est transmis directement à une sortie ou à un indicateur.

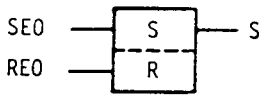
Table de vérité:

Instruction	Contenu de l'ACCU	Sortie ou indicateur
OUT	1 Ø	H L

Chapitre F
Exemples: 1, 2, 3, 5

SEO
REO

Positionner ou remettre la sortie



SEO: Set output ---> positionner la sortie
REO: Reset output ---> remettre la sortie

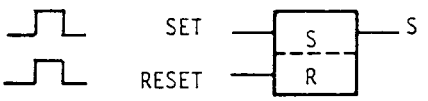
Format de l'instruction:

Code d'instruction		Opérand	
Code mnémonique	Code numérique	Description	Plage
SEO	11	Adresse de la sortie ou de l'indicateur	Ø...255* (i) 288...999 (i)
REO	12	Adresse de la sortie ou de l'indicateur	Ø...255* (i) 288...999 (i)

(i) = indéxable

* Effet de ces instructions sur temporisateurs et compteurs voir STR/SCR.

La fonction des instructions SEO et REO correspond à celle d'un flip-flop RS.



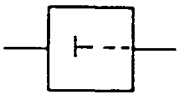
Bascule RS : toujours précédée de DYN

Une sortie ou un indicateur positionnée par l'instruction SEO reste positionnée (H) jusqu'à la remise par l'instruction REO. Les deux cas exigent que la combinaison précedente ait été remplie (ACCU = 1).

A noter les priorités de positionnement resp. remise lors de l'utilisation de SEO et REO dans une boucle de programme (plan logique).

Chapitre F
Exemple: 4, 5, 1Ø, 11 et 12

COO Complémenter la sortie



COO: Complement output ---> scruter la sortie et la complémenter

Format de l'instruction:

Code d'instruction		Opérand	
Code mnémonique	Code numérique	Description	Plage
COO	13	Adresse de la sortie ou de l'indicateur	Ø...255 (i) 288...999 (i)

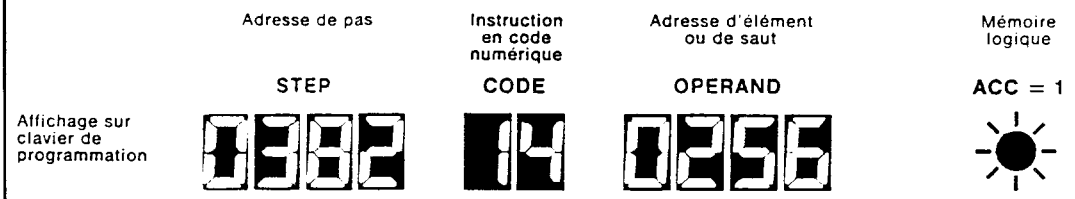
(i) = indéxable

L'état de signalisation de la sortie ou de l'indicateur est scruté (H/L) et positionné dans l'état inverse. COO est effectué que si ACCU = 1.

Chapitre F
Exemple 6

E 3 Instructions de temporisation et de comptage

E 3.1 Instructions de transfert et d'opérations arithmétiques



	Code mnemo.	Code num.	Opérand	Description
1ère ligne	STR/ SCR	14/ 15	256...319	Adresse de registre
2ème ligne Instructions de transfert		16 17 18	Adresse la plus haute de 8 éléments consécutifs	2 x 4 bits BCD (x 1) 2 x 4 bits BCD (x 10) 2 x 4 bits BCD (x 100)
E ⟶ T/C T/C ⟶ E		19 20 21 22 23 24 25 26	Adresse la plus haute de la groupe des élements	Lecture de 5x4 bits BCD Sortie de 5x4 bits BCD Sortie de 8 bits binaires Sortie de 12 bits binaires Sortie de 16 bits binaires Lecture de 8 bits binaires Lecture de 12 bits binaires Lecture de 16 bits binaires
Instructions arithmétiques ⊕ ⊖ ⊗ ÷		27 28 29 30	xxx xxx xxx xxx	Additionne Soustrais Multiplie Divise } avec une constante (0...255) ou avec le contenu d'un T/C (256...319)[1]
Instructions de transfert C ⟶ IR C ⟶ C		31	iii	iii = 0: Le compteur est chargé avec le contenu du registre d'index iii = 256...319[1]: Le compteur est chargé avec le contenu d'un autre T/C

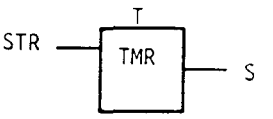
1) Le domaine d'emploi des compteurs est dépendant du Hardware et du Firmware (voir vue d'ensemble page III).

E 3.2 Temporisateurs et compteurs

Pour ces fonctions 32 registres programmables comme temporisateurs ou compteurs sont à disposition. Une même adresse de registre peut servir plusieurs fois dans le programme pour des fonctions differentes (temporisateur ou compteur) si la fonction précédente n'est plus active.

Ces instructions sont des d'instructions à deux lignes dont la deuxième ligne contient la valeur de temporisation ou de comptage.

L'instruction DTC (voir plus loin) permet l'affichage du contenu des registres sur le clavier de programmation ou sur le module d'affichage.



STR Positionner le temporisateur

STR: Set Timer ---> positionne le temporisateur sur la valeur préséléctionnée et l'initialise

Format de l'instruction (instruction à deux lignes):

Code d'instruction		Opérand		
Code mnémonique	Code numérique	Description	Plage	
STR	14	Adresse du temporisateur	256...287 (i)	1ère ligne
---	ØØ	Temps en 1/1Øs	Ø ...2Ø47	2ème ligne

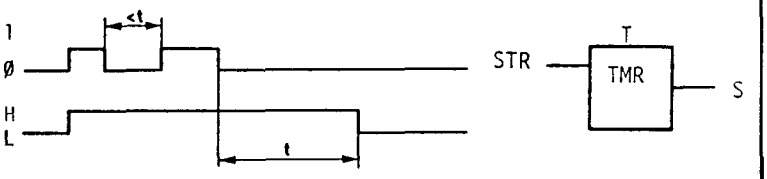
(i) = indéxable

Les temporisateurs peuvent être positionnés avec une valeur de temporisation appropriée. Le positionnement commute l'état logique de "L" à "H" et le temps commence immédiatement à s'écouler.

Si pendant le déroulement, le temporisateur est positionné à nouveau (STR), il recommence avec le nouveau temps. La fonction de base correspond au déclenchement retardé.

Instruction de positionnement (STR) avec ACCU

Temporisateur (S), état du signal



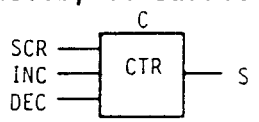
STR n'est exécuté que si l'ACCU = 1.

Chapitre F
Exemples: 7, 9, 1Ø, 12

L'utilisation de l'instruction REO (p.ex. REO 26Ø) permet d'interrompre le déroulement du temporisateur jusqu'à la libération par SEO. Alors, le reste de temps se déroule.

Notes:
- La base de temps est Ø,1s à la livraison. Par déplacement d'un pont la base de temps peut être diminuée à Ø,Ø1s pour obtenir une temporisation plus fine.
- La précision relative (précision de repétition) est de +Ø/-1 digit. Par exemple: Pour 3s au rythme de Ø,1s, le temps sera compris entre 2,9Ø à 3,ØØs. La précision absolue dépend du quartz de système.
- La modification de la base de temps à Ø,Ø1s ne rend pas le processeur plus rapide, mais le temps de cycle est prolongé d'approximativement 2Ø%. Ainsi, la précision en est améliorée 3s --> 2,99 à 3,ØØs.
- Pour rendre les compteurs et temporisateurs non volatils, il suffit d'enclencher le pont NVOL sur le module CPU.

SCR — Positionner le compteur



SCR: Set Counter ---> Positionne le compteur sur la valeur présélectionnée

Format de l'instruction (instruction à deux lignes):

Code d'instruction		Opérand		
Code mnémonique	Code numérique	Description	Plage	
SCR	15	Adresse du compteur	256...319*(i)	1ère ligne
---	ØØ	Valeur de comptage	Ø...2Ø47	2ème ligne

(i) = indéxable

*) Le nombre de registres à disposition 64, 228 respectivement 256 dépent du type de CPU utilisé. Voir vue d'ensemble des registres en page III.

Les compteurs peuvent être positionnés avec une valeur de comptage appropriée. Le positionnement commute l'état de signalisation de "L" à "H". L'état de signalisation se commute sur "L" au passage du contenu de ØØØ1 à ØØØØ, lorsque le compteur est décrementé. Le compteur ne peut pas prendre des valeurs négatives. Elles doivent être mémorisées par un second compteur.

Valeur de comptage (contenu du régistre)	Etat logique du compteur
ØØØØ	L
≥ØØØ1	H

SCR n'est exécuté que si l'ACCU = 1.

Chapitre F
Exemples: 8, 18

*) Le domaine d'emploi des compteurs est dépendant du Hardware et du Firmware (voir vue d'ensemble page III).

Extensions des valeurs de la 2ème ligne des temporisateurs et compteurs

La valeur de temporisation ou de comptage est déterminée par la deuxième ligne des instructions STR et SCR. A part le code de base ØØ de cette deuxième ligne d'instruction, l'utilisation des codes supérieurs permet l'extension de la plage resp. l'introduction externe de valeurs.

Extension de la plage

Avec le code ØØ, la valeur max. qui peut être introduite dans l'opérand est 2Ø47 ou 2Ø4,7s. Les codes Ø1 à 15 de la table permettent la présélection de valeurs plus grandes.

Code de 2ème ligne	Valeur totale = valeur de l'opérand
ØØ	+ Ø
Ø1	+ 2Ø48
Ø2	+ 4Ø96
Ø3	+ 6144
Ø4	+ 8192
Ø5	+ 1Ø24Ø
Ø6	+ 12288
Ø7	+ 14336
Ø8	+ 16382
Ø9	+ 18432
1Ø	+ 2Ø48Ø
11	+ 22528
12	+ 24576
13	+ 26624
14	+ 28672
15	+ 3Ø72Ø

Les valeurs indiquées sont directement applicable aux compteurs.

Pour temporisateur - Avec base de temps 1/1Ø (x Ø,1s) (Pont fermé, état à la livraison)
- Avec base de temps 1/1ØØ (x Ø,Ø1s) (Pont ouvert)

Exemple: Un temps de 1Ømin. doit être programmé (= 6ØØs).

Solution: Le code Ø2 est à choisir car 6ØØØ se trouve entre 4Ø96 et 6144. L'opérand correspondant sera 6ØØØ-4Ø96=19Ø4
Il est recommandé de combiner un temporisateur avec un compteur lors de l'application fréquente de temps longs.

E 3.3 Instructions de transfert de bits BCD et binaires (Vue d'ensemble)

Code d'instruction		Opérand		
Code mnémonique	Code numérique	Description	Plage	
SCR/STR	15/14	Adresse du registre T/C	256...319* (i)	(1ère ligne)
---	CODE	Adresse la plus haut du groupe de N éléments consécutifs	Elément (E, S, indicateur)	(2ème ligne)

(i) = indéxable

CODE	N élements	Description du code
16	8 élements	Lecture de 2 x 4 bits BCD x 1
17	8 élements	Lecture de 2 x 4 bits BCD x 1Ø
18	8 élements	Lecture de 2 x 4 bits BCD x 1ØØ
19	2Ø élements	Lecture de 5 x 4 bits BCD
2Ø	2Ø élements	Sortie de 5 x 4 bits BCD
21	8 élements	Sortie de 8 bits binaires
22	12 élements	Sortie de 12 bits binaires
23	16 élements	Sortie de 16 bits binaires
24	8 élements	Lecture de 8 bits binaires
25	12 élements	Lecture de 12 bits binaires
26	16 élements	Lecture de 16 bits binaires

*) Le domaine d'emploi des compteurs est dépendant du Hardware et du Firmware (voir vue d'ensemble page III).

E 3.4 Introduction externe de valeurs sous forme BCD (2x4 bits)

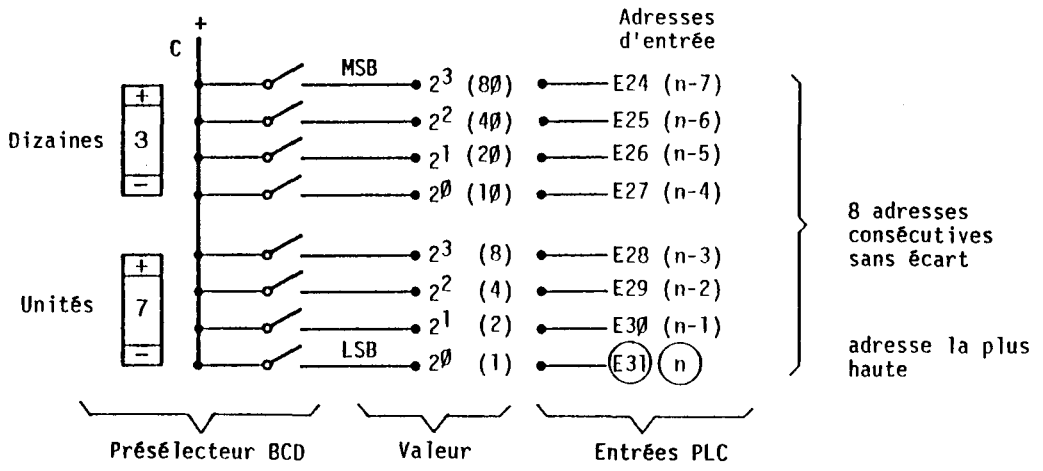
Des valeurs de temporisation ou de comptage peuvent être introduites de l'externe sous forme de code BCD à 2 digit (ØØ...99). Cette valeur est multiplié par 1, 1Ø ou 1ØØ selon le code 16, 17 ou 18 de la 2ème ligne d'instruction de programme.

Format d'instruction:

Multiplicateur	Code de la 2ème ligne	Opérand
1 x 1Ø x 1ØØ x	16 17 18	Adresse la plus élevée (n) de 8 entrées ou indicateurs consécutifs

La valeur BCD à 2 digit, consituée de 8 adresses consécutives (= 1 byte), peut être introduite par des entrées, des indicateurs ev. aussi des sorties lisibles.

Exemple d'un présélecteur BCD à 2 digits relié à 8 entrées:



En résumé ce qui donne:

		Code	Opérand	
- Compteur pour	1... 99	16	31	2ème ligne
Compteur pour	1Ø... 99Ø	17	31	
Compteur pour	1ØØ...99ØØ	18	31	
- Tempo * pour	Ø,1...9,9s	16	31	
Tempo * pour	1...99s	17	31	
Tempo * pour	1Ø...99Øs	18	31	

*) Pour la base de temps standard Ø,1s.

Chapitre F
Exemples: 9 et 18

E 3.5 Fonctions additionnelles

STR SCR	Les instructions élargies des compteurs et des temporisateurs pour le transfert des données et l'arithmétique

E 3.5.1 Transfert des données et report

a) Instruction de lecture des élements (E, S, indicateurs) comme valeur de temporisateurs (T) ou compteurs (C). L'opérand de la 2ème ligne indique l'adresse la plus élevée de l'élément du groupe (LSB).

Codes	(2ème ligne)
19	5x4 bits BCD (5 décades)
24	8 bits binaires
25	12 bits binaires
26	16 bits binaires

Exemple:

```
STR (14)  256
     24   431
```

24 = 8 bits binaires
431 = 8 indicateurs, 424...431

b) Instruction de sortie des registres de temporisation (T) ou comptage (C) sur des éléments (S ou indicateurs). L'opérand de la 2ème ligne indique l'adresse la plus élevée de l'élément du groupe (LSB).

Codes	(2ème ligne)
2Ø	5x4 bits BCD (5 décades)
21	8 bits binaires
22	12 bits binaires
23	16 bits binaires

Exemple:

```
SCR (15)  3ØØ
    21     75
```

21 = 8 bits, binaires
75 = 8 sorties, 68...75

c) Charge d'un temporisateur (T) ou d'un compteur (C) avec le contenu du registre d'index (IR) ou la valeur d'un autre T/C.

```
SCR (15)  ccc
    31    iii
```

31 = CODE

iii = Ø — Le compteur ccc est chargé avec le contenu du registre d'index (du programme parallèle correspondant).

iii = 256...287 — Le compteur ccc est chargé avec le contenu du T/C iii adressé.

iii = 288...319* — Le compteur ccc est chargé avec le contenu du C iii adressé.

E 3.5.2 Opérations arithmétiques

Les opérations sont exécutées entre deux registres de comptage, le résultat de l'opération étant disponible dans le registre de la première ligne (ccc). Si la capacité du registre est dépassé (65'535), l'ACCU indique Ø. La 2ème ligne peut également contenir une constante de valeur ≤255.

SCR (15) ou STR (14)

```
SCR (15)  ccc
CODE      xxx
```

ccc = adresse du 1ère T/C
xxx = adresse du 2ème T/C ou valeur d'une constante ≤255
CODE = 27...3Ø, voir page suivante

Remarque: Toutes les adresses d'éléments peuvent être indexées.
Les instructions STR et SCR sont exécutées uniquement si l'ACCU = 1.

*) Le nombre de registres à disposition 64, 228, resp. 256 dépent du type de CPU utilisé. Voir vue d'ensemble des registres en page III.

E 3.6 Codes pour les opérations arithmétiques

SCR (15)	ccc
CODE	xxx

27 Addition des contenus de ccc et de xxx, le résultat étant mémorisé en ccc. Si la valeur 65'535 est dépassée, ACCU = Ø.

Contenu ccc ⊕ contenu xxx ---> contenu ccc

28 Soustraction du contenu xxx de celui de ccc, le résultat étant mémorisé en ccc. Si le résultat est négatif, ACCU = Ø.

Contenu ccc ⊖ contenu xxx ---> contenu ccc

29 Multiplication des contenus de ccc et de xxx, le résultat étant mémorisé en ccc. Si la valeur 65'535 est dépassée, ACCU = Ø.

Contenu ccc ⊗ contenu xxx ---> contenu ccc

3Ø Division du contenu de ccc par le contenu de xxx, le résultat étant mémorisé en ccc. Si le contenu de xxx = Ø, le contenu de ccc reste inchangé et ACCU = Ø. Un reste éventuel est perdu à la division.

Contenu ccc ÷ contenu xxx ---> contenu ccc

Exemples:

- Avant l'addition — C256 = 3Ø, C26Ø = 54

  Opération

SCR(15)	256
27	26Ø

  Après — C256 = 84, C26Ø = 54, ACCU = 1

- Avant la soustraction — C258 = 124, C274 = 146 [1]

  Opération

SCR(15)	258
28	274

  Après — C258 = 65'514 [1], C274 = 146, ACCU = Ø

  [1] 124 - 146 = -22
  ---> 65'536 - 22 = 65'514

- Avant la multiplication — C26Ø = 12, C282 = 6

  Opération

SCR(15)	26Ø
29	282

  Après — C26Ø = 72, C282 = 6, ACCU = 1

- Avant la division — C31Ø = 1942, constante = 23

  Opération

SCR(15)	31Ø
3Ø	23

  Après — C31Ø = 84, ACCU = 1, le reste (1Ø) n'est pas mémorisé.

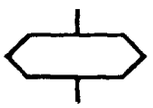
E 3.7 Résumé des instructions pour les registres de temporisation et de comptage

	Code mnémo.	Code numérique	Opérand	Signification
1ère ligne	STR SCR	14 15	256...319 [1]	Adresse du registre
2ème ligne	constantes en compteur/temporisateur	ØØ Ø1 Ø2 Ø3 Ø4 Ø5 Ø6 Ø7 Ø8 Ø9 1Ø 11 12 13 14 15	xxxx xxxx	Valeur de l'opérand + Ø . + 2Ø48 . + 4Ø96 . + 6144 . + 8192 . + 1Ø24Ø . + 12288 . + 14336 . + 16348 . + 18432 . + 2Ø48Ø . + 22528 . + 24576 . + 26624 . + 28672 Valeur de l'opérand + 3Ø72Ø
	transfert des données	16 17 18	} adresse la plus haute du groupe de 8 éléments consécutifs	2 x 4 bits BCD x 1 2 x 4 bits BCD x 1Ø 2 x 4 bits BCD x 1ØØ
	transfert des données	19 2Ø 21 22 23 24 25 26	} adresse la plus haute du groupe d'éléments	Lecture pour 5 x 4 bits BCD Sortie pour 5 x 4 bits BCD Sortie pour 8 bits binaires Sortie pour 12 bits binaires Sortie pour 16 bits binaires Lecture pour 8 bits binaires Lecture pour 12 bits binaires Lecture pour 16 bits binaires
	arithmétique	27 28 29	xxx xxx xxx	Addition Soustraction Multiplication Division } avec une constante (Ø...255) ou avec le contenu d'un T/C (256...319) [1]
	transfert IR→C / C→C	31	iii	iii = Ø le compteur est chargé avec le contenu du registre d'index iii = 256...319 [1] le compteur de la 1ère ligne est chargé avec le contenu du T/C de la 2ème ligne

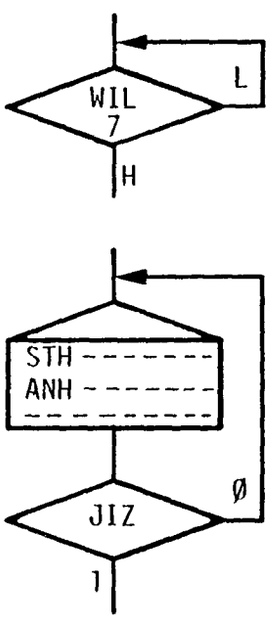
1) Le domaine d'emploi des compteurs est dépendant du Hardware et du Firmware (voir vue d'ensemble page III).

E 4 Instructions de saut et d'attente

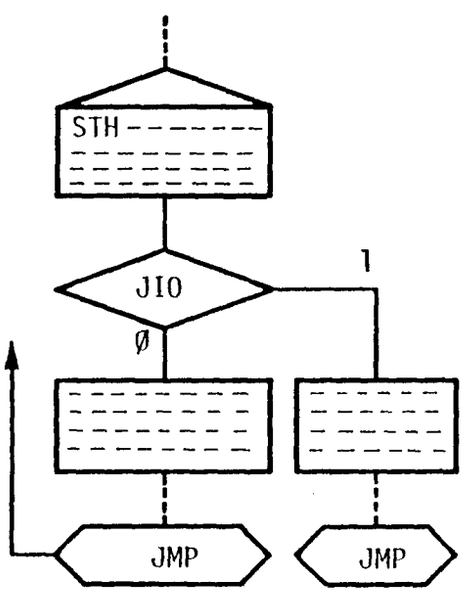
	Adresse de pas	Instruction en code numérique	Adresse d'élément ou de saut	Mémoire logique
	STEP	CODE	OPERAND	ACC = 1
Affichage sur clavier de programmation	0382	14	0256	

		Code numérique	Code mnémonique	Instruction en anglais	Description
	Instructions de saut	2Ø	JMP	Unconditional Jump	Saute à une adresse de pas, sans condition
		21	JIO	Jump if Accu is One	Saute si accu = 1 sur adresse de pas
		22	JIZ	Jump if Accu is Zero	Saute si accu = Ø sur adresse de pas
		23	JMS	Jump to Subroutine	Saute dans un sous-programme
		24	RET	Return from Subrout.	Revient d'un sous-programme
	Instructions d'attente	25	WIH	Wait if High	Attend aussi longtemps que l'élément est sur High
		26	WIL	Wait if Low	Attend aussi longtemps que l'élément est sur Low

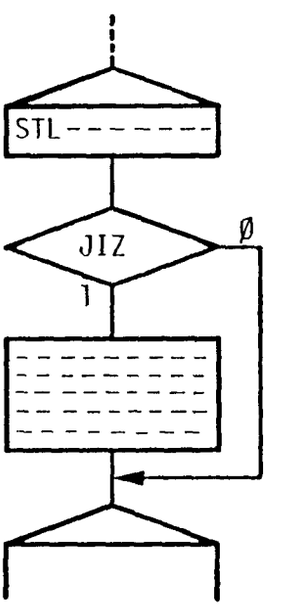
Les instructions de saut et d'attente permettent non seulement la réalisation de boucles d'attente (selon diagramme de flux) mais aussi des bifurcations et sauts pour ignorer des parties de programmes (selon plan logique).
Des instructions spéciales permettent de sauter dans les sous-programmes.



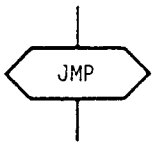
Boucles d'attente



Bifurcations



Sauts afin d'ignorer des parties de programmes



JMP | Saut inconditionnel

JMP: Jump ---> saut de programme inconditionnel (independant de l'ACCU).

Format de l'instruction:

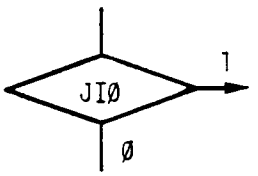
Code d'instruction		Opérand		
Code mnémonique	Code numérique	Description	Plage	
JMP	2Ø	Adresse de pas	1...2Ø47	
Pour un saut à une adresse de pas >2Ø47, il faut utiliser une seconde ligne d'instruction:				
JMP	2Ø	Toujours Ø	ØØØØ	1ère ligne
---	ØØ	Adresse de pas	Ø...8191	2ème ligne

Tout programme doit être terminé par un saut inconditionnel.
JMP positionne l'ACCU à 1.

Chapitre F
Tous les Exemples

JIO
JIZ

Saut conditionnel



JIO: Jump if ACCU is One ---> saute si ACCU = 1
JIZ: Jump if ACCU is Zero ---> saute si ACCU = Ø

Format de l'instruction:

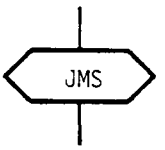
Code d'instruction		Opérand		
Code mnémonique	Code numérique	Description	Plage	
JIO	21	Adresse de pas	1...2Ø47	
JIZ	22	Adresse de pas	1...2Ø47	
Pour un saut conditionnel à une adresse de pas supérieure de 2Ø47 il faut utiliser une seconde ligne d'instruction:				
JIO	21	Toujours Ø	ØØØØ	1ère ligne
---	ØØ	Adresse de pas	Ø...8191	2ème ligne
JIZ	22	Toujours Ø	ØØØØ	1ère ligne
---	ØØ	Adresse de pas	Ø...8191	2ème ligne

Selon l'état de l'ACCU, c.-à-d. selon le résultat de la scrutation ou de la combinaison précedante, les sauts seront effectués ou non. Si la condition de saut n'est pas remplie le programme continu de façon linéaire. Les instructions de saut positionnent l'ACCU à 1, que le saut soit effectué ou non.

Chapitre F
Exemples: 11, 15, 16

JMS
RET

Saut dans et retour d'un sous-programme (sous-routine)



JMS: Jump to Subroutine ---> Saut dans un sous-programme
RET: Return from Subroutine ---> Retour d'un sous-programme
= fin du sous-programme

Les deux instructions sont inconditionnelles, c.-à-d. qu'elles sont exécutées indépendant de l'état de l'ACCU.

Format de l'instruction:

Code d'instruction		Opérand		
Code mnémonique	Code numérique	Description	Plage	
JMS	23	Adresse de début du sous-programme	1...2Ø47	
Pour les sous routines qui débutent à une adresse supérieure à 2Ø47, il faut utiliser cette instruction à deux lignes:				
JMS	23	Toujours Ø	ØØØØ	1ère ligne
---	ØØ	Adresse de début du sous-programme	Ø...8191	2ème ligne
RET	24	Toujours Ø	ØØØØ	

Une description de la technique de sous-programme se trouve dans la partie D 5.6.

L'adresse de pas suivante du programme principal est mémorisée au moment du saut dans le sous-programme. A l'instruction RET le programme retourne automatiquement à l'adresse correcte. Pour cela l'opérand de RET ne contient que Ø.

Il est possible de sauter du 1er dans un 2ème et de ce 2ème dans un 3ème sous-programme (3 niveaux de sous-programmes). Les retours se font en sens inverse jusqu'au programme principal.

JMS et RET positionnent l'ACCU à 1.

L'opérand ØØØØ est réservé pour des sauts de lignes supérieures à 2Ø47.

Chapitre F
Exemples: 12, 18

JMP, JIO, JIZ, JMS Fonctions additionnelles pour adresses 2Ø48...8191

S'il faut programmer de grands sauts dont l'adresse de destination se situe dans la deuxième moitié de la mémoire utilisateur, c.-à-d. de 2Ø48 à 8191, les instructions de saut doivent être réalisées sur deux lignes.

a) Instructions de saut avec l'opérand compris entre 1 et 2Ø47
(l'opérand ØØØØ n'est pas admis, voir b).

Exemple: saut conditionnel avec adresse de destination 1845

soit

```
JIO(21)  1845
```

sur une ligne comme jusqu'à maintenant

ou

```
JIO(21)     Ø
   ØØ    1845
```

sur deux lignes, avec Ø pour l'opérand de la première ligne.

b) Instructions de saut avec l'opérand compris entre 2Ø48 et 8191

Exemple: saut dans le sous-programme 328Ø

A la programmation:

```
5Ø1   JMS(23)     Ø    E
5Ø2      ØØ    328Ø    E
```

Lors du contrôle:

```
5Ø1   JMS(23)     Ø    +
5Ø2      Ø1    1232    C   ---> convertis
5Ø2      EE    328Ø    +
```

La valeur Ø1 1232, comme elle figure dans la mémoire utilisateur, correspond à l'adresse de saut 328Ø. Ø1 représente le multiple de 2Ø48 et 1232 le reste (1 x 2Ø48 + 1232 = 328Ø).

Avec la touche C on affiche l'adresse de saut réelle, en ayant comme CODE les signes EE (valables pour l'appareil de programmation ..PØ5).

Lors d'une instruction de saut avec Ø comme opérand, la deuxième ligne est automatiquement lue comme adresse de destination. Un saut vers l'adresse de destination Ø contient par conséquent toujours deux lignes:

```
JMP(2Ø)    Ø
   ØØ      Ø
```

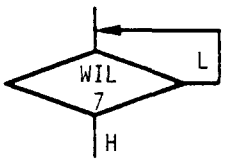
c) Un exemple pratique sur la base d'un clignoteur dans un sous-programme.

Programme principal

```
►5ØØ   26  WIL     1
 5Ø1   23  JMS     Ø
 5Ø2       ØØ   35ØØ  ⇒
└5Ø3   2Ø  JMP   5ØØ
```

Sous-programme

```
⇒ 35ØØ  Ø2  STL   256
  35Ø1  14  STR   256
  35Ø2      ØØ    2
  35Ø3  13  COO   34
  35Ø4  24  RET   Ø   ⇒
```



WIH
WIL Instructions d'attente

WIH: Wait if High ---> attends si l'état de signalisation d'élément = "H"
WIL: Wait if Low ---> attends si l'état de signalisation d'élément = "L"

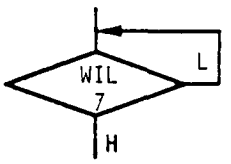
Format de l'instruction:

Code d'instruction		Opérand	
Code mnémonique	Code numérique	Description	Plage
WIH	25	Adresse d'élément	Ø...999 (i)
WIL	26	Adresse d'élément	Ø...999 (i)

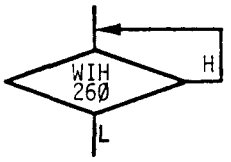
(i) = indéxable

L'instruction d'attente est utilisée uniquement avec le diagramme de flux. Le processeur attend dans la boucle d'attente jusqu'à ce que l'élément adressé ait pris l'état de signalisation nécessaire à la suite de programme.

par exemple attendre, **pendant** que l'entrée est "L".



par exemple attendre, **pendant** que le temps se déroule,c'est-à-dire que l'élement (temporisateur) est "H".

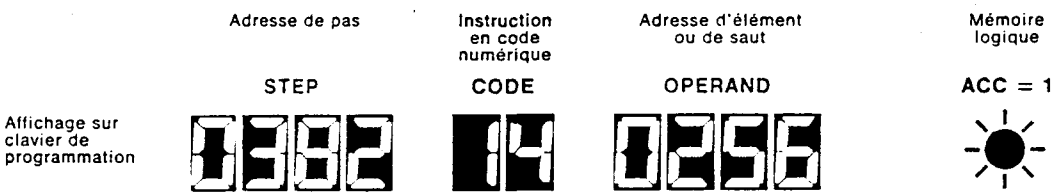


"Wait" permet de façon similaire à "Start" l'initialisation d'une partie de programme, car dans les deux cas un élément est scruté d'après son état de signalisation. WIH et WIL positionnent l'ACCU à 1, c.-à-d. la suite des actions programmées n'est effectué qu'après la sortie de la boucle d'attente.

Chapitre F
Exemples: 1Ø, 12, 16

E 5 Instructions auxiliaires

Indexage (voir E 6), instructions de PAS additionnelles (voir E 7), instructions d'affichage (voir E 8).



	Code Num.	Code Mnémo	Instruction en anglais	Description
Instructions auxiliaires (chapitre E5)	ØØ	NOP	No Operation	Sans effet
	19	SEA	Set ACCU	Positionne l'ACCU = 1
Indexage (chapitre E6)	16	SEI	Set Index	Pos. le registre d'index sur la valeur choisie
	27	INI	Increment Index	Augmente le registre d'index de 1
	28	DEI	Decrement Index	Diminue le registre d'index de 1
Instructions de PAS (chapitre E7)	29	PAS	Ø...15	Assigne un programme parallèle
	29	PAS	18	Modification de programmes parallèles
	29	PAS	3Ø...38	Check-Sum du système resp. mémoire utilisateur
Instructions d'affichage (chapitre E8)	3Ø	DOP	Display Operand	Affichage du chiffre de l'opérand
	31	DTC	Display Timer or Counter	Affichage du contenu d'un temporisateur ou d'un compteur

NOP No operation

NOP: No Operation ---> Pas d'opération

Format de l'instruction:

Code d'instruction		Opérand
Code mnémonique	Code numérique	
NOP	ØØ	Ø

Cette instruction est exécuté par le processeur, mais n'effectue aucune fonction. Elle sert à la reservation d'espace pour l'extension de programme ou au remplissage des trous entre les programmes.

Les lignes de programmes superflues après modifications peuvent être remplies (effacées) par des NOP. Plusieurs NOP consécutifs sont obtenus par l'actionnement repétitif de la touche "Enter" du clavier de programmation en mode PROG.

SEA Positionner l'ACCU à 1

SEA: Set Accumulator = 1 ---> Positionner l'ACCU à 1

Format de l'instruction:

Code d'instruction		Opérand
Code mnémonique	Code numérique	
SEA	19	Ø

L'instruction SEA positionne inconditionnellement l'ACCU à 1. SEA sera utilisé avant les instructions qui ne sont exécutées que si l'ACCU est égal à 1. Par ex. avant DTC.

Chapitre F
Exemple 8

E 6 Indexation

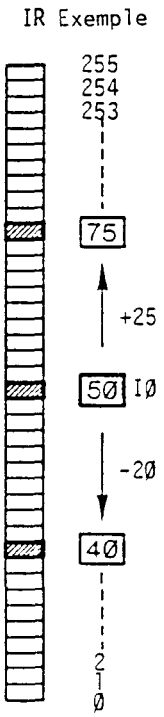
SEI
INI
DEI Adresses d'élément - indexage (traitement de séries)

SEI: Set Index ---> positionne le registre index avec valeur de départ selon l'opérand
INI: Increment Index ---> incrémente le registre index de 1 jusqu'à la valeur finale la plus haute selon l'opérand
DEI: Decrement Index ---> décremente le registre index de 1 jusqu'à la valeur finale la plus basse selon l'opérand

Il est souvent nécessaire de traiter une série d'entrées, de sorties, d'indicateurs, de temporisateurs ou de compteurs de la même façon (p.ex. repositionner tous les indicateurs non-volatils selon D 5.7). Dans ce cas l'indéxation d'adresses permet une réduction considérable de la longueur des programmes.

Le registre index IR est un registre similaire au registre de comptage et possède une capacité comprise entre Ø et 255*. Les trois instructions permettent le positionnement resp. l'altération du contenu du registre jusqu'à la valeur limite désirée.

Exemple



INI 75 Incrémente le registre index de 1 jusqu'à la valeur finale la plus haute 75. L'ACCU est alors Ø.

SEI 5Ø Positionne le registre index sur la valeur initiale IØ = 5Ø et l'ACCU à 1.

DEI 4Ø Décrémente le registre index de 1 jusqu'à la valeur finale la plus basse 4Ø et remet l'ACCU à Ø.

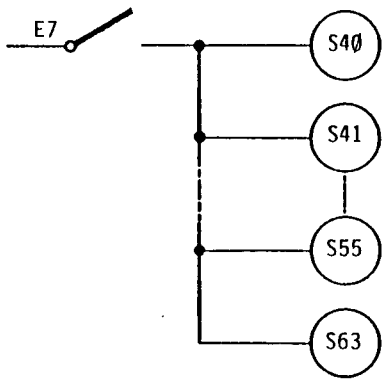
---

*) Chaque programme parallèle dispose d'un registre index particulier. Au total 16 registres sont à disposition. Le contenu de ces registres est perdu lors d'une chute de tension.

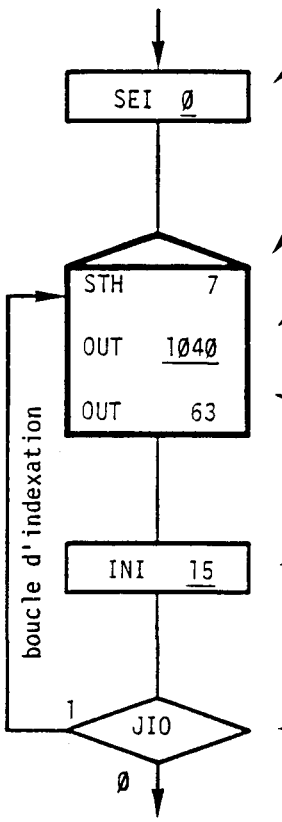
Le module processeur du PCA232 a un registre d'indexation d'une capacité de 1K (Ø...1Ø23). Voir exemple d) à la fin de ce chapitre.

Tous les éléments-opérand de la partie E de ce manuel désignés ayant un "(i)" peuvent être indexés. Le processeur additionne à l'adresse d'élément de l'opérand la valeur actuelle du registre index.

Par exemple effectuer ce qui suit:



Lors de la fermeture de E7, les sorties S4Ø à S55 ainsi que la sortie S63 doivent être activées.



Le registre index est positionné sur la valeur initiale Ø.

Scrutation de l'entrée E7.

La fonction d'indexation à partir de la sortie S4Ø sera influencée par l'introduction du nombre 1ØØØ. L'adresse S4Ø sera positionnée lors de la première boucle, car le registre d'index est égal à Ø à ce passage (SEI Ø). Pour les boucles suivantes, le contenu du registre d'index sera incrémenté de 1 par l'instruction INI, ce qui aura pour conséquence de positionner les sorties S41, 42, 43 à 55.

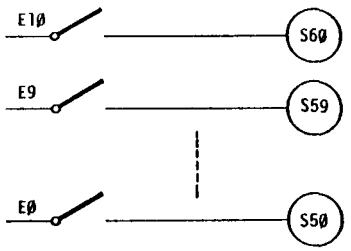
La sortie S63 est adressée directement (sans 1ØØØ), le registre d'index n'a pas d'influence pour elle.

INI incrémente le registre d'index de 1 à chaque boucle jusqu'à la valeur de 15. L'ACCU est égal à 1 après exécution de l'instruction INI 15 si la valeur du registre d'index est < 15.

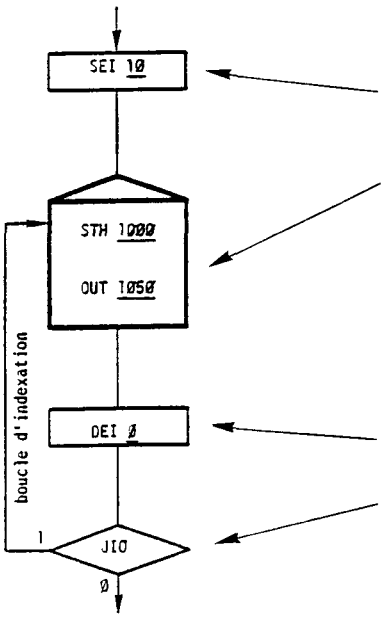
Le saut conditionnel est exécuté jusqu'à ce que la boucle ait été parcourue 16 fois et jusqu'à ce que toutes les 16 sorties S4Ø à S55 soient traitées.

Chapitre F
Exemples: 15, 16, 17

Deuxième exemple:



Les entrées E1Ø à EØ doivent activer les sorties S6Ø à S5Ø. Elles doivent être traitées en ordre descendant par DEI.



Le registre index est positionné sur la valeur initiale 1Ø (1Ø - Ø = 1Ø ou 6Ø - 5Ø).

Les entrées comme les sorties doivent être indexées, il faut donc additionner le nombre 1ØØØ. Les adresses les plus basses sont introduites, car le contenu du registre d'index 1Ø est additionné à l'adresse la plus basse. Ainsi E1Ø et S6Ø sont traitées en premier, puis E9 et S59 jusqu'à EØ et S5Ø.

DEI décrémente la valeur du registre d'index de 1 à chaque boucle jusqu'à la valeur finale Ø. Lorsque le registre d'index atteint la valeur Ø = DEI Ø, l'ACCU prends également Ø ce qui provoque la sortie de la boucle d'indexation.

Formats d'instruction:

Code d'instruction		Opérand	
Code mnémonique	Code numérique	Description	Plage
SEI	16	Valeur initiale du registre index (IØ)	IØ = Ø...255
INI	27	Valeur finale sup. pour indexation incr.	Ø...255 1)
DEI	28	Valeur finale inf. pour indexation decr.	Ø...255 2)

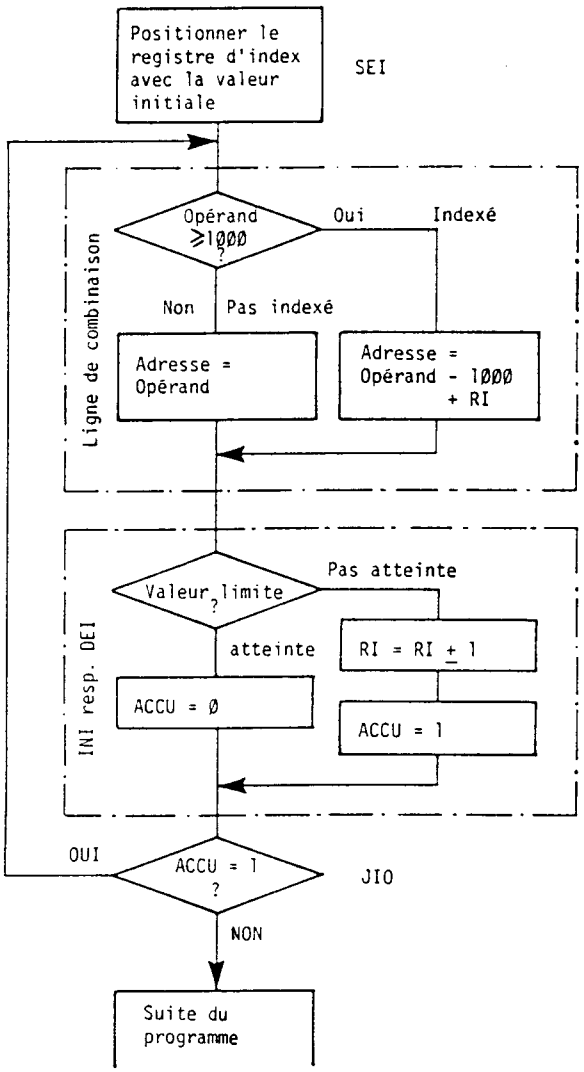
1) La valeur maximale 255 est atteinte, provoquée par l'instruction INI, lors du prochain passage par une instruction INI, la valeur du registre d'index sera alors Ø (...253,254,255, Ø,1,2...).
2) La valeur minimale Ø est atteinte, provoquée par l'instruction DEI, lors du prochain passage par une instruction DEI, la valeur du registre d'index sera alors 255 (...2,1,Ø,255, 254,253...)

Table de vérité:

Instruction	Influence sur l'ACCU
SEI	L'instruction positionne l'ACCU = 1
INI	Si la valeur de registre ≠ ou > opérand ---> ACCU = 1 Si la valeur de registre = opérand ---> ACCU = Ø
DEI	Si la valeur de registre ≠ opérand ---> ACCU = 1 Si la valeur de registre = opérand ---> ACCU = Ø

Résumé de l'indexation

Schéma d'indexation:



a) Indexation avec INI

- En général le registre est positionné à la valeur Ø par SEI.
- Les adresses les plus basses (+1ØØØ) des éléments à indexer sont introduites.
- La valeur finale pour INI résulte de la différence de l'adresse la plus haute moins la plus basse à indexer.

b) Indexation avec DEI

- En général le registre est positionné à la valeur X par SEI. X = différence entre l'adresse la plus haute moins la plus basse à indexer.
- Les adresses les plus basses (+1ØØØ) des éléments à indexer sont introduites.
- En général Ø est introduit pour la valeur finale de DEI.

Autres possibilités de programmation en technique d'indexation

Ce que l'on appelle le traitement en série par indexation est décrit dans cette partie. Il est aussi possible d'utiliser le registre d'index de façon que par ex. des sous-programmes activent d'autres éléments selon le contenu du registre d'index.

Exemple:

Fermeture de E1 fait clignoter S45, fermeture de E2 fait clignoter S5Ø.

Programme principal

```
********** PROGRAMME PRINCIPAL
ADDR  NC  MNC  OPRD
700   01  STH     1
701   22  JIZ   704   ->
702   16  SEI     0
703   23  JMS   710   =>
704   01  STH     2
705   04  ANL     1
706   22  JIZ   700   ->
707   16  SEI     5
708   23  JMS   710   =>
709   20  JMP   700   ->
```

Sous-programme

```
========== SOUS-PROGRAMME
ADDR  NC  MNC  OPRD
710   02  STL   256
711   14  STR   256
712   00  00      3
713   13  COO  1045
714   24  RET     0   ->
```

Le même SPRGM peut ainsi modifier les états de sorties ou d'entrées différentes.

SEI, INI, DEI Fonctions additionnelles

SEI(16) iii Positionner le registre d'index

a) iii = Ø...255
   Le registre d'index est chargé avec la valeur fixe iii de l'opérand.

b) iii = 256...319* (pour le PCA232: 256...511)
   Le registre d'index est chargé avec le contenu du registre T/C correspondant.

INI(27) iii augmenter le registre d'index de 1 (incrémenter)
DEI(28) iii diminuer le registre d'index de 1 (décrementer)

a) iii = Ø...255
   Le registre d'index est incrémenté ou décrémenté jusqu'à la valeur fixe iii introduite.

b) iii = 256...319* (pour le PCA232: 256...511)
   La valeur à considérer se trouve dans le registre T/C correspondant.

Remarques:
- Tous les PCA possèdent 16 registres d'index, un pour chacun des programmes parallèles.
- La capacité max. d'un registre d'index est 255. Si une valeur est transférée d'un compteur, celle-ci ne doit pas dépasser la valeur de 255. La capacité max. d'un registre d'index du PCA232 est 1Ø23 (voir exemple d).

Exemples généraux:

a) C267 = 1Ø2
   Après l'instruction SEI(16) 267, le contenu du registre d'index est également 1Ø2.

b) C256 = 44
   Après l'instruction INI(27) 256, la valeur limite à incrémenter est également 44.

c) C26Ø = 1ØØ, IR = 4
   Après l'instruction SEI(16) 1256, le registre d'index prend la valeur 1ØØ (indexation double).

d) Une valeur supérieur > 255 doit être chargé dans le registre d'index du PCA232. Ceci est réalisé indirectement par un compteur intermédiaire.

```
SCR  28Ø  } La valeur 8ØØ
 ØØ  8ØØ  } sera chargé dans
SEI  28Ø  } le registre d'index
```

*) Le domaine d'emploi des compteurs est dépendant du Hardware et du Firmware (voir vue d'ensemble page III).

E 7 Instructions PAS

PAS Ø ... PAS 15 Assignation de programmes parallèles (PP)

PAS: Program Assignment ---> Assignation de programme parallèle

Format de l'instruction (deux lignes):

Code d'instruction		Opérand		
Code mnémonique	Code numérique	Description	Plage	
PAS	29	Numéro de programme consécutif de	Ø...15	1ère ligne
---	ØØ	Adresse de départ du programme	Ø...819Ø	2ème ligne

Il faut "informer" dès le début du programme le CPU, si plusieurs programmes (max. 16) doivent se dérouler parallèlement. Ceci est éffectué en assignant les adresses de tous les programmes parallèles par l'instruction à deux lignes PAS.

L'assignement des programmes parallèles dans la partie d'assignation devrait être complète (tous les numéros) en ordre ascendant à partir du programme no. 1. Cette partie d'assignation ne doit être parcourue qu'une seule fois juste après la mise en service du PCA.
Les programmes parallèles peuvent être réassigné avec l'instruction PAS Ø...15. A n'importe quel endroit du programme il est possible de, par exemple de réassigner le PP3 de l'adresse de début 3ØØ à l'adresse de début 4ØØ.

Le programme parallèle no. Ø ne doit pas être assigné. Il est activé directement à la suite de l'assignation de tous les programmes parallèles.

La structure des programmes parallèles ressort de la partie D 5.5. Les PP particuliers sont exécutés d'une façon semblable au "Time sharing". Le processeur change d'un PP à l'autre selon des conditions très précises.

Conditions nécessaire au changement de programme parallèle

Chacune des instructions suivantes provoquent le changement:

- WIH, WIL (pour autant que la condition d'attente soie remplie)
- JMP, JIO, JIZ, JMS, RET
- Chaque deuxième ou troisième instruction STH ou STL

PAS Ø...15 est indépendant de la valeur de l'ACCU ainsi donc l'instruction PAS Ø...15 est toujours exécutée et ne change pas la valeur de l'ACCU.

Chapitre F
Exemples: 13, 18

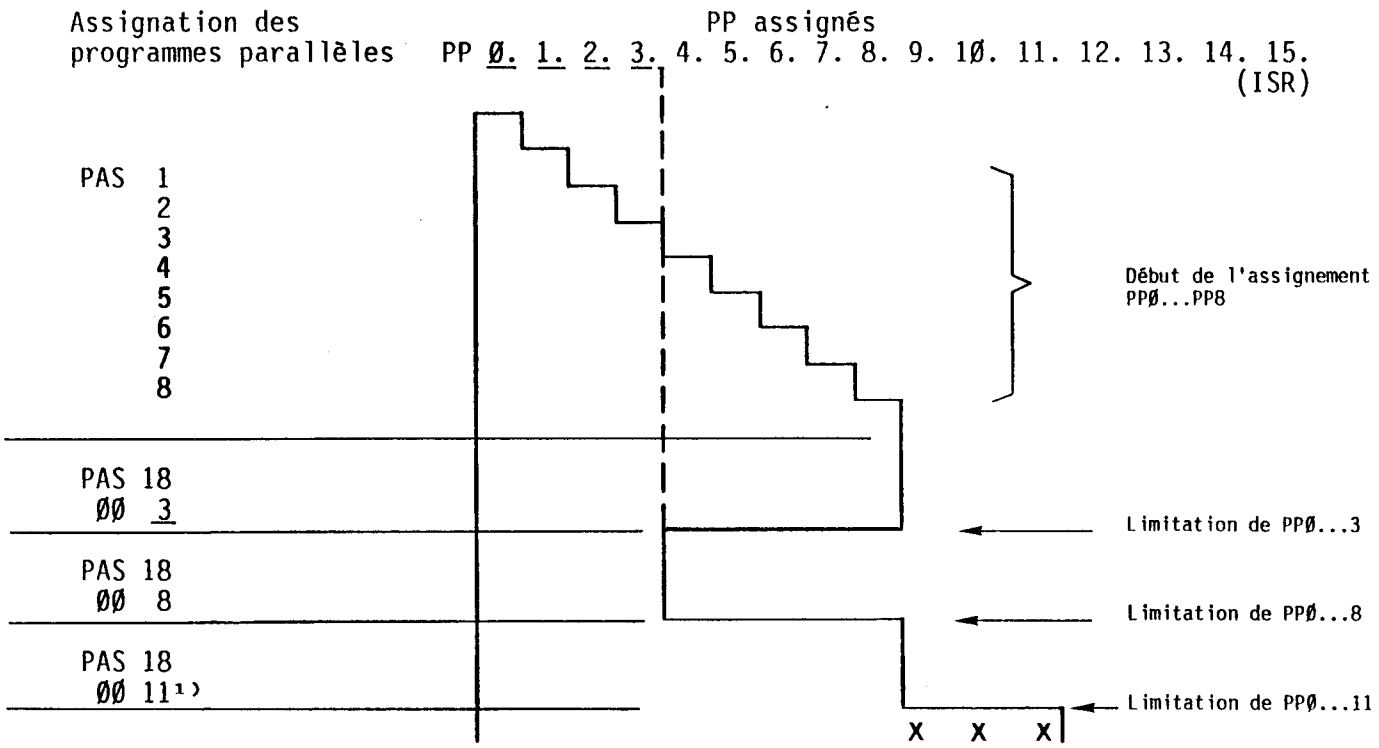
PAS 18 Limitation du nombre de programme parallèles travaillés

Tous les SAIA°PLC permettent la gestion simultanée de 16 programmes parallèles (PP). Jusqu'à présent, si un de ces PP n'était plus utilisé, il était désactivé à l'aide d'une réassignation sur une boucle de programme vide. Cette méthode ne changeait donc pas le nombre des PP, par conséquent pas ou peu de gain de temps.

Après l'assignation des PP par PAS Ø...15 max., dans n'importe quel programme parallèle, à n'importe quel endroit et aussi souvent qu'il le faut, le nombre de parogrammes parallèles actifs peut être limité par le haut avec l'instruction PAS 18.

Code d'instruction		Opérand		
Code mnémonique	Code numérique	Description	Plage	
PAS	29	toujours 18	- - - -	1ère ligne
---	ØØ	Limite du PP depuis le haut	PP1...15	2ème ligne

L'instruction PAS 18 est toujours exécutée (indépendament de l'état de l'ACCU) et l'état de l'ACCU n'est pas modifié.



1) Lorsque dans la 2ème ligne du PAS 18 un nombre plus grand que celui du dernier programme assigné est introduit, il n'en résultera pas d'erreur, mais une perte de temps dans l'exécution, car les PP 9...11 sont tout de même contrôlés par le programme système.

PAS 3Ø
PAS 31...38 "Check-Sum" du programme utilisateur et système

La fonction "Check-Sum" effectue une sommation du contenu de la mémoire du programme système (PAS 3Ø) ou du programme utilisateur (PAS 31...38). On peut ainsi s'assurer que, dans les mémoires vérifiées, il n'y a pas eu de modifications du contenu.

Après execution de l'instruction, il vient:
ACCU = 1 si la valeur de référence correspond à la somme de contrôle,
ACCU = Ø si la valeur de référence ne correspond pas à la somme de contrôle.

Les instructions PAS 3Ø...38 sont toujours exécutées, indépendament de l'état de l'ACCU. S'il y a eu une modification du contenu des mémoires, l'utilisateur peut programmer les mesures qui lui paraissent nécessaires: enclenchement d'une alarme, remise à zéro du Watchdog, etc.

Sommation de contrôle du programme système

Code mnémonique	Code numérique	Opérand	
PAS	29	Toujours 3Ø	1ère ligne
---	ØØ	Toujours Ø	2ème ligne

Sommation de contrôle du programme utilisateur

Code d'instruction		Opérand		
Code mnémonique	Code numérique	Description	Plage	
PAS	29	Séquence du programme 1er K...8ème K*	31...38*	1ère ligne
---	XX	Valeur de référence	xxxx	2ème ligne

On obtient la valeur de référence pour le programme utilisateur en exécutant l'instruction PAS correspondante en mode de fonctionnement STEP. Le PCA affiche pour quelques secondes la valeur de référence sur l'écran de l'appareil de programmation. En mode de fonctionnement PROG, on peut ensuite introduire cette valeur de référence dans la deuxième ligne.

Attention: Le temps d'exécution de ces instructions est relativement important:
PAS 3Ø ≈ 28,Øms, PAS 31...38 ≈ 8,3ms
(PCA232: PAS 30 ≙ 13,6ms PAS 31...38 ≙ 9,5ms)
"Check-Sum" ne doit être exécuté que lorsque le déroulement du processus à surveiller le permet, c.-à-d. à la mise en route de l'installation, à la fin du déroulement d'un cycle, etc.

*) La sommation de contrôle est effectuée séparement pour chaque bloc de "K" lignes de programme (voir exemples à la page suivante).

Il est recommandé de n'introduire cette instruction dans le programme utilisateur que lorsque celui-ci est développé et testé. Chaque modification du programme, que ce soit une extension ou un racourcissement, conduit à une modification de la somme de contrôle et demande d'une modification de la valeur de référence.

Exemple: A l'enclenchement tester un programme utilisateur de 2K.

```
    20    PAS     31   } "Check-Sum"
    21     Ø9    825   } 1er K
  ←22    JIZ     35
    23    PAS     32   } "Check-Sum"
    24     Ø7   154Ø   } 2ème K
  ←25    JIZ     35

    3Ø  ┌→COO    255   } Programme de travail
    31  └─JMP     3Ø   } avec contrôle
                       Watchdog

  ┌→35    SEO     15   } Positionner la sortie
  └─36    JMP     35   } d'alerte à l'extérieur
                       du programme de
                       travail
```

Marche à suivre:

- Après mise au point définitive du programme, sélectionner le mode STEP
- Choisir l'adresse 23:
  [A] 23 [+]
  --> La valeur de la somme pour le 2ème K de programme apparaît pendant 2Ø sec.
- Introduire la valeur de référence au mode d'exploitation PROG:
  [A] 24 val. de référence [+]
- Procédure identique pour PAS 31

E 8 Instructions d'affichage

DOP Affichage d'un opérand

DOP : Display Operand - - - > Affichage du contenu de l'opérand

Format de l'instruction:

Code d'instruction		Opérand	
Code mnémonique	Code numérique	Description	Plage
DOP	3Ø	Affichage de chiffre quelconque	Ø...2Ø47

DOP est une instruction auxiliaire utilisée avant tout lors de la mise en service et pour la détection d'erreurs. Il suffit de programmer de façon qu'à la détection d'un état particulier ou d'une erreur de processus, un chiffre d'identification soit affiché. Cet affichage a lieu en mode de fonctionnement "RUN" dans le champ opérand du clavier de programmation ou sur un module d'affichage.
Chaque affichage reste pour 1s. Si l'affichage doit rester plus longtemps, il faut exécuter l'instruction DOP au min. une fois par seconde (de préférence dans une boucle de programme).
DOP n'est exécuté que si la combinaison n'a pas été remplie (ACCU = Ø).

Chapitre F
Exemple 14

DTC Affichage de la valeur de temporisation ou de comptage

DTC: Display Timer or Counter ---> Affichage de la valeur de temporisation ou de comptage

Format de l'instruction:

Code d'instruction		Opérand	
Code mnémonique	Code numérique	Description	Plage
DTC	31	Adresse du temporisateur ou compteur	256...319* (i)

(i) = indéxable

DTC est également une instruction auxiliaire très utile lors de la mise en service et pour la détection d'erreurs.
DTC permet en mode de fonctionnement "RUN" d'effectuer l'affichage du déroulement d'un temporisateur ou d'une valeur de comptage dans le champ opérand du clavier de programmation ou sur le module d'affichage (max. 9999).
Chaque affichage reste pour 1s. Si l'affichage doit rester plus longtemps, il faut exécuter l'instruction DTC au min. une fois par seconde (de préférence dans une boucle de programme).
DTC n'est exécuté que si l'ACCU = 1.

*) Structure des registres voir page III.

Chapitre F
Exemples: 8, 18

PARTIE F EXEMPLES DE PROGRAMMATION

Exemple 1 ET/OU selon schéma de contact
2 ET/OU selon symbole de fonctions logiques
3 OU exclusif
4 Combinaison selon symboles de fonctions logiques
5 Contact START/STOP avec auto-maintien
6 Réducteur d'impulsion
7 Retard au déclenchement
8 Comptage / Décomptage
9 Impulsion calibrée à l'enclenchement avec introduction externe de temps en code BCD
10 Retard à l'enclenchement selon diagramme de flux
11 Combinaison ET selon diagramme de flux
12 Contrôle de déroulement avec ou sans sous-programme
13 Utilisation de programmes parallèles
14 Circuit de surveillance avec affichage d'erreurs sur le clavier de programmation
15 Commutation de plusieurs sorties en séries
16 Enclenchements et déclenchements décalés de plusieurs sorties
17 Petit circuit de surveillance
18 Programme d'un commutateur

Exemples de programmation avec modules analogiques

19 Sortie d'une tension analogique à partir de 8 resp. 12 entrées
20 Sortie d'une tension en dents de scie
21 Lecture d'une valeur BCD du module PCA1.F12 et transfert sur une sortie analogique du module PCA1.W12 (8 bits) resp. du module W32 (12 bits)
22 Introduction par lecture d'une tension analogique dans un registre compteur et affichage de la valeur binaire avec DTC
23 Régulateurs à 2 et 3 points (avec le module analogique PCA1.W1..resp. W3..)

Exemple pratique

24 Installation automatique de perçage
24.1 Dimensionnement du PLC
24.1.1 Fonctions
24.1.2 Nombre d'E/S
24.1.3 Types d'E/S
24.1.4 Capacité de mémoire
24.1.5 Affichage

24.2 Programmation
24.2.1 Structure du programme
24.2.2 Plan de déroulement selon DIN
24.2.3 Programmation

Schéma pour la résolution d'un problème de contrôle par utilisation d'un PLC

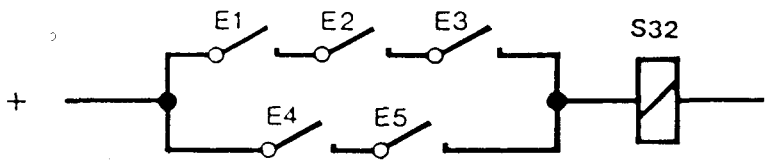
PARTIE F Exemples de programmation

Les instructions de niveau 1H definies dans la partie E sont mises en relation dans la collection d'exemples qui suit. Il s'agit de préférence d'exemples typiques qui servent à éclaircir le fonctionnement et l'usage de ces instructions particulières. Le dernier exemple est un programme complet tiré de l'usage pratique.
L'adressage des E/S a été choisi de façon à ce que tous les exemples pour un PCA1 resp. PCA2 puissent être testé à l'aide de l'unité de simulation d'entrées PCA2.S1Ø. L'adressage est donc:

E = Ø...31
S = 32...63

Exemple 1: ET/OU selon schéma de contact

1) Branchement parallèle



Adresse de pas (STEP)	Instruction Description	CODE mnémonique (numérique)	Influence sur element ou adresse de pas	resp. OPERAND
1Ø	Départ et scrutation	STH (Ø1)	E1	1
11	Combinaison ET	ANH (Ø3)	E2	2
12	Combinaison ET	ANH (Ø3)	E3	3
13	Combinaison OU	ORH (Ø5)	E4	4
14	Combinaison ET	ANH (Ø3)	E5	5
15	Transmission du résultat	OUT (1Ø)	S32	32
16	Saut au début	JMP (2Ø)	STEP 1Ø	1Ø

PAS QUOI OU

1	2	3	4

STEP

1	2

CODE

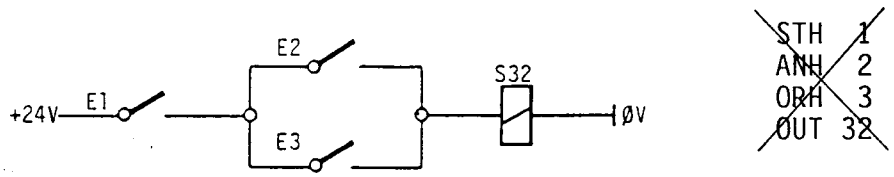
1	2	3	4

OPERAND

2) OU est prioritaire sur ET

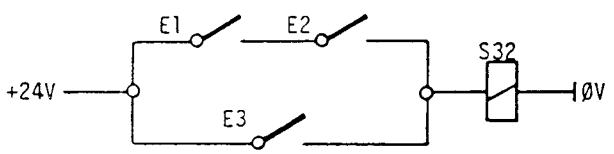
A noter la particularité lors de combinaison des fonctions OU et ET, expliqué par l'exemple suivant:

La disposition suivante de contact est à programmer:



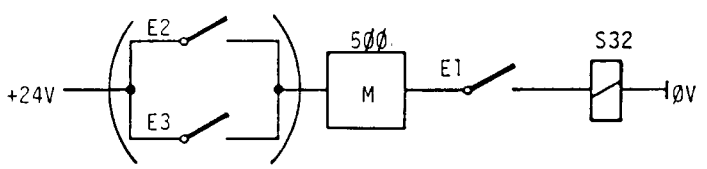
~~STH 1~~
~~ANH 2~~
~~ORH 3~~
~~OUT 32~~

On ne peut pas programmer selon le schéma à contact ci-dessus, car la fonction OU correspond par définition á un contact parallèle de la branche de combinaison suivante. Ceci corresponderait au schéma suivant:



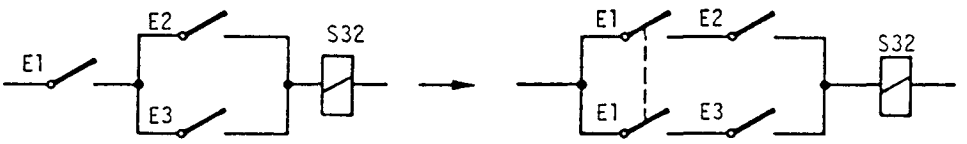
2 alternatives de programmation pour ce schéma sont à disposition:

- Par mémorisation intermédiaire du résultat OU




```
ADDR NC   MNC   OPRD
  40 01   STH      2   SCRUTATION DE E2 D'APRES "H"
  41 05   ORH      3   COMBINAISON DE E3, SCRUTEE D'APRES "H"
  42 10   OUT    500   MEMORISATION DU RESULTAT INTERMED. SUR UN INDICATEUR
--------------------
  43 01   STH    500   NOUVELLE LIGNE AVEC SCRUTATION DE L'INDICATEUR
  44 03   ANH      1   COMBINAISON ET AVEC E1, SCRUTEE D'APRES "H"
  45 10   OUT     32   TRANSFERT DU RESULTAT A S32
  46 20   JMP     40 ->
```

- Par modification du schéma à contact pour obtenir des contacts parallèles directement programmable avec uniquement des branches parallèles.



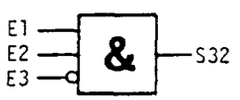
Le second contact E1 ne sert qu'à la programmation. En réalité E1 n'existe qu'une seule fois, mais il est scruté deux fois par le programme.

```
ADDR NC   MNC   OPRD
  50 01   STH      1   INITIALISATION DE LA 1ERE BRANCHE DE COMBINAISON
  51 03   ANH      2
  52 05   ORH      1   COMBINAISON ET INITIALISATION DE LA 2EME BRANCHE DE
  53 03   ANH      3                                          COMBINAISON
  54 10   OUT     32
  55 20   JMP     50 ->
```

Exemple 2: ET/OU selon symboles de fonctions logiques

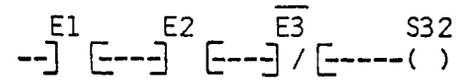
1) Fonction ET

Symboles de fonctions logiques:



Le symbole logique permet la représentation claire des combinaisons logiques et des états de signalisation.

Schéma à contacts:

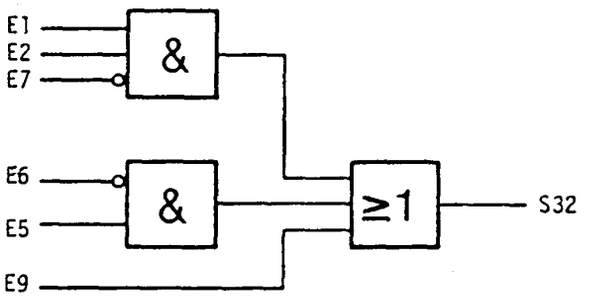


La représentation par schéma de contact de la fonction ci-dessus est problématique. Afin d'éviter les confusions lors de l'usage pratique on se sert souvent de contacts symboliques. La combinaison est réalisée (S32 active) si E1 et E2 = H et si E3 = L.

```
ADDR NC   MNC   OPRD
  60 01   STH      1   SCRUTATION DE E1 D'APRES "H"
  61 03   ANH      2   COMBINAISON ET REMPLIE (ACCU = 1) SI E2 = "H"
  62 04   ANL      3   COMBINAISON ET REMPLIE (ACCU = 1) SI E3 = "L"
  63 10   OUT     32
  64 20   JMP     60 ->
```

Lors de la programmation selon symboles logiques les éléments non inversés sont scrutés et combinés par STH, ANH, ORH, les éléments inversés par STL, ANL, ORL.

2) ET/OU combiné

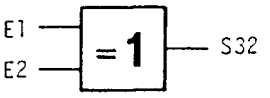



```
ADDR NC   MNC   OPRD
  70 01   STH      1
  71 03   ANH      2
  72 04   ANL      7
  73 06   ORL      6   NOUVELLE BRANCHE PARALLELE DE COMBINAISON
  74 03   ANH      5
  75 05   ORH      9   NOUVELLE BRANCHE PARALLELE DE COMBINAISON
  76 10   OUT     32
  77 20   JMP     70 ->
```

Exemple 3: Combinaison OU exclusif

1) Comparaison d'états logiques différents

Deux entrées sont à comparer. La sortie doit être = L, si les deux entrées ont le même état logique. La sortie doit être = H (active), s'ils différent.

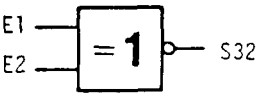



```
ADDR NC   MNC   OPRD
  80 01   STH      1   SCRUTATION DE E1
  81 07   XOR      2   COMBINAISON OU EXCLUSIF AVEC E2
  82 10   OUT     32
  83 20   JMP     80 ->
```

2) Comparaison de mêmes états logiques

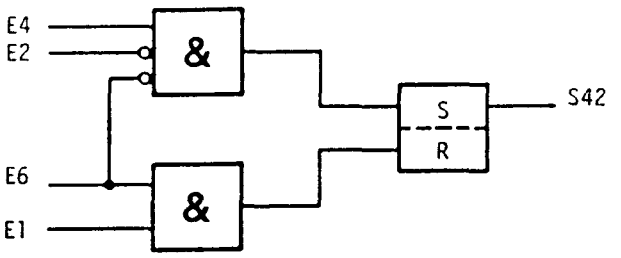
Comme exemple 1) mais

avec mêmes états logiques aux entrées ----> sortie = H (active)
avec différents états logiques aux entrées ----> sortie = L




```
ADDR NC   MNC   OPRD
  85 01   STH      1
  86 07   XOR      2   COMBINAISON OU EXCLUSIF
  87 08   NEG      0   NEGATION DU CONTENU DE L'ACCU
  88 10   OUT     32
  89 20   JMP     85 ->
```

Exemple 4: Combinaison selon symboles de fonctions logiques



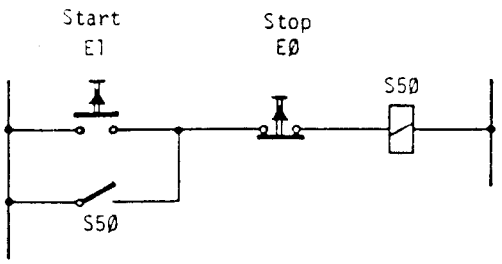

```
ADDR  NC   MNC   OPRD
  90  01   STH      4   SCRUTATION DE E4
  91  04   ANL      2   COMBINAISON ET, SCRUTEE D'APRES "L"
  92  04   ANL      6   COMBINAISON ET, SCRUTEE D'APRES "L"
  93  11   SEO     42   SI LA COMBINAISON EST REMPLIE, POSITIONNER SORTIE S42
--------------------
  94  01   STH      6   NOUVELLE COMBINAISON, SCRUTATION DE E6
  95  03   ANH      1   COMBINAISON ET
  96  12   REO     42   SI LA COMBINAISON EST REMPLIE, REMETTRE SORTIE S42
  97  20   JMP     90-->
```

Si les deux combinaisons seraient remplies, la sortie serait positionnée à l'adresse de pas 93 et remise à l'adresse de pas 96, à chaque parcours du programme, ce qui ferait osciller inadmissiblement la sortie. Ceci est évité en verrouillant avec E6.

Exemple 5: Contact start/stop avec auto-maintien

1) Selon schéma de contact

L'exemple classique qui suit est connu de la technique des contacteurs:

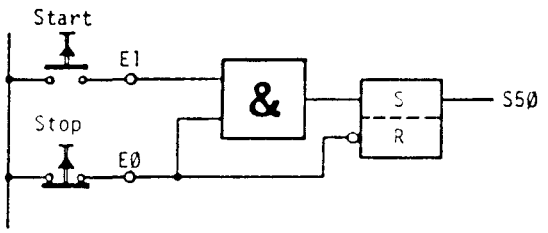


Programme:

```
STH     1
ORH     5Ø   } Start
OUT     4ØØ
STH     4ØØ
ANH     Ø    } Stop
OUT     5Ø
```

La sortie S5Ø peut être directement combinée, lorsque du coté "Hardware" ceci est possible (pas de rack C3Ø ou module B9Ø). Comme le démontre le programme, le contact d'ouverture EØ est combiné d'après "H", car l'activation de la S5Ø n'est possible que si EØ est fermée.
Cette programmation est de même sûre lors de rupture de fil. Une rupture de fil dans les lignes EØ, E1 ou S5Ø provoque toujours une interruption de S5Ø.

2) Selon symboles de fonctions logiques

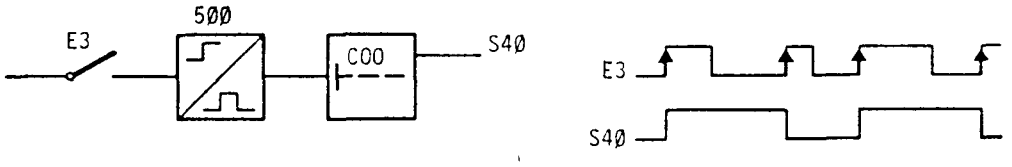


Programme:

```
STH     1
ANH     Ø    } Start
SEO     5Ø
STL     Ø    } Stop
REO     5Ø
```

Tout comme l'exemple 4, l'instruction de positionnement n'est active que si EØ = "H". L'instruction de remise est prioritaire lors de l'actionnement simultané des deux touches à cause de la combinaison "ET".
Cette programmation représentée ci-dessus est également sûre lors de rupture de fil.

Exemple 6: Réducteur d'impulsion (commutateur pas à pas)

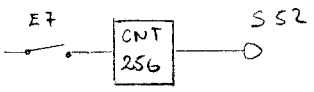


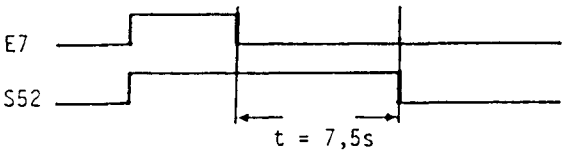

```
ADDR NC   MNC   OPRD
 100 01   STH      3 ┐ SCRUTATION DE E3
 101 09   DYN    500 │ DECL. PAR LE FLANC. MEMORISATION SUR L'INDIC. 500
 102 13   C00     40 │ SCRUTATION DE S40 ET COMPLEMENTATION DE SON CONTENU
 103 20   JMP    100 ┘
```

Sans instruction DYN, avec le contact E3 fermé, la sortie 4Ø serait complémenté à chaque parcours du programme, à peu près 3ØØØ fois par seconde dans cette petite boucle.

Lorsque le DYN est utilisé, à la fermeture de E3, la sortie 4Ø sera complémentée seulement au premier parcours du programme. Il n'y aura aucune influence lors des parcours suivants du programme sauf si E3 reçoit un nouveau front montant (fermeture du contact).

Exemple 7: Retard au déclenchement



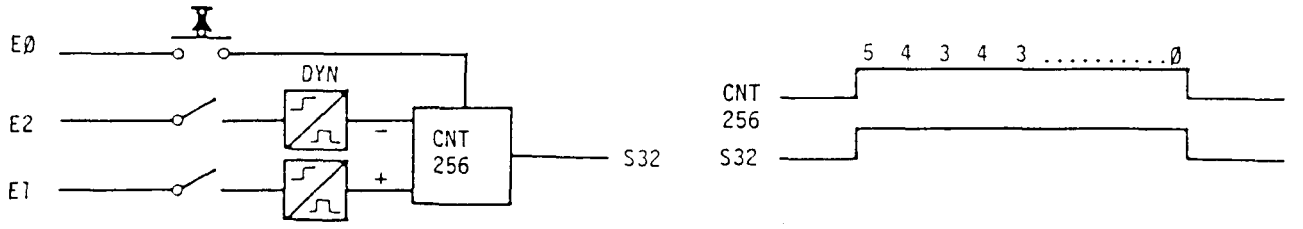



```
ADDR NC   MNC   OPRD
 104 01   STH      7   SCRUTATION DE E7
 105 14   STR    256   POSITIONNER LE TEMPORISATEUR    ) INSTRUCTION
 106 00    00     75   INTRODUCTION DU TEMPS EN 1/10s ) A 2 LIGNES
--------------------
 107 01   STH    256   SCRUTATION DU TEMPORISATEUR
 108 10   OUT     52   TRANSFERT SUR S52
 109 20   JMP    104 ->
```

A la fermeture de E7 le temporisateur est positionné et son état logique devient "H". Le temps ne commence à se dérouler qu'à l'ouverture de E7. (Plus précisement: le temps commence tout de suite à se dérouler mais le temporisateur est repositionné et recommence à temporiser après quelques 1ØØµs. Ceci à chaque parcours de programme, aussi longtemps que E7 reste fermée, c.-à-d. jusqu'à l'ouverture de E7).

Si E7 est refermée pendant que le temps se déroule, le temporisateur est repositionné et recommence sa temporisation.

Exemple 8: Comptage/décomptage
(avec affichage de la valeur de comptage sur le clavier de programmation ou sur un module d'affichage)

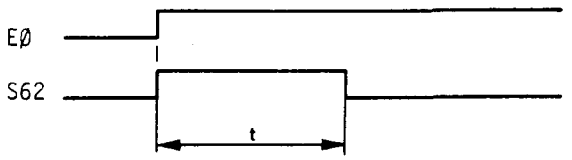



```
ADDR NC   MNC   OPRD
 110 01   STH      0   SCRUTATION DE E0
 111 15   SCR    256   POSITIONNER LE COMPTEUR    ) INSTRUCTION
 112 00    00      5   VALEUR DE COMPTAGE         ) A 2 LIGNES
--------------------
 113 01   STH      1   SCRUTATION DE E1
 114 09   DYN    500   DECL. PAR LE FRONT, MEMORISATION SUR L'IND. 500
 115 17   INC    256   +1
--------------------
 116 01   STH      2   SCRUTATION DE E2
 117 09   DYN    501   DECL. PAR LE FRONT, MEMORISATION SUR L'IND. 501
 118 18   DEC    256   -1
--------------------
 119 01   STH    256   SCRUTATION DU COMPTEUR (H SI LONGTEMPS QUE LE CONTENU 0)
 120 10   OUT     32   TRANSFERT  DE L'ETAT LOGIQUE DU COMPTEUR SUR S32
--------------------
 121 19   SEA      0   POSITIONNER L'ACCU = 1
 122 31   DTC    256   AFFICHAGE DE LA VALEUR DE COMPTAGE DANS LE CHAMP
 123 20   JMP    110 ->                                          OPERAND
```

L'exécution de DTC requiert ACCU = 1, sinon l'affichage de la valeur de comptage n'est pas exécuté.
Soit (comme dans l'exemple) SEA précède DTC ou l'instruction DTC est placée au début de la boucle de programme, car l'ACCU est toujours 1 après l'exécution de l'instruction JMP.

Exemple 9: Impulsion calibrée à l'enclenchement avec introduction externe de temps en code BCD

Plage de temps 1...99s codé externe par deux commutateurs BCD. Entrées E24...31 des commutateurs BCD.



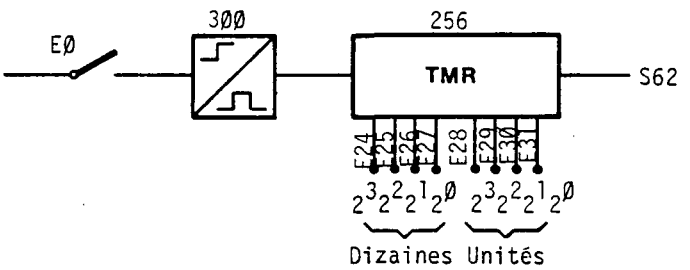


Table BCD

Signaux binaires aux 4 entrées (commutateur BCD)				
E28	E29	E3Ø	E31	Valeur décimale
$2^3 = 8$	$2^2 = 4$	$2^1 = 2$	$2^0 = 1$	
L	L	L	L	Ø
L	L	L	H	1
L	L	H	L	2
L	L	H	H	3
L	H	L	L	4
L	H	L	H	5
L	H	H	L	6
L	H	H	H	7
H	L	L	L	8
H	L	L	H	9

```
ADDR NC   MNC   OPRD
 130 01   STH     0   SCRUTATION DE E0
 131 09   DYN   300   DECL. PAR LE FRONT: TEMP. EST POSITIONNE QU'AU 1ER CYCLE
 132 14   STR   256   POSITIONNER TEMPORISATEUR
 133 17    17    31   VALEUR EXTERNE * 10 * 1/10S; ADRESSES N SUR E31
------------------
 134 01   STH   256   SCRUTATION TEMPORISATEUR
 135 10   OUT    62   TRANSFERT SUR S62
 136 20   JMP   130->
```

Exemple 1Ø: Retard à l'enclenchement selon diagramme de flux

Diagramme de flux:

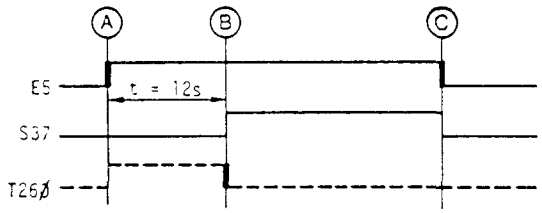
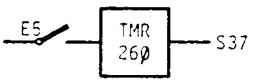
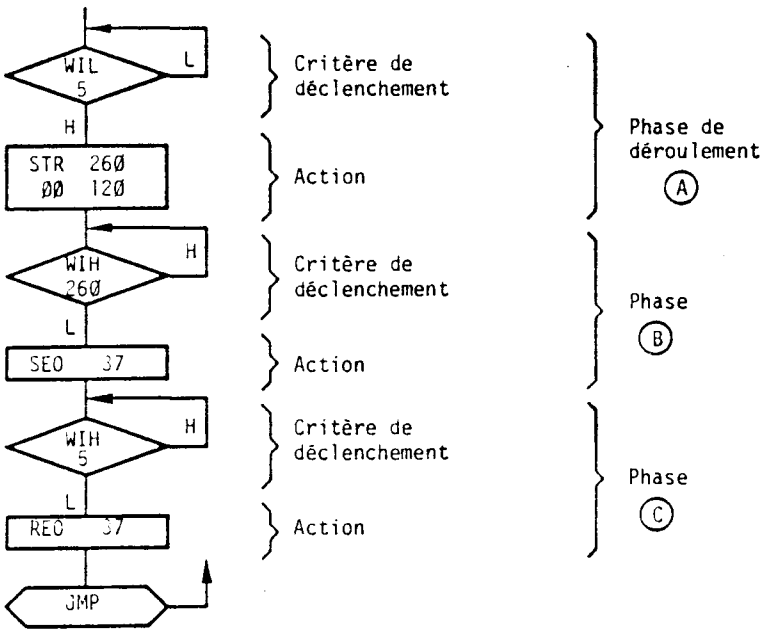


Schéma à contacts:



= Critère de déclenchement

Le diagramme séquentiel en temps peut être divisé en différentes phases de déroulement. Chaque phase dispose d'un critère correspondant de déclenchement. Chaque boucle d'attente attend jusqu'à ce que la condition correspondante à l'exécution de la fonction suivante soit remplie.

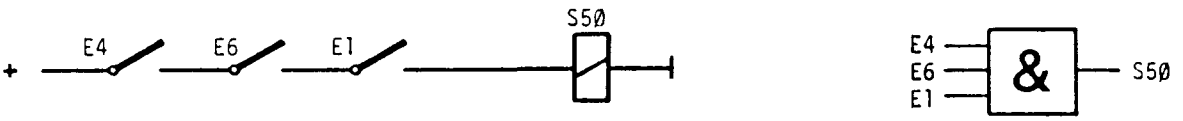


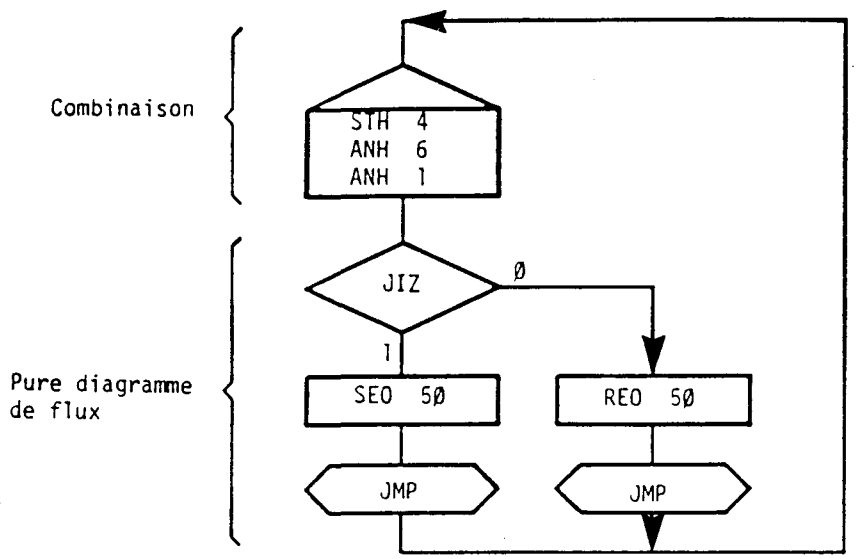

```
ADDR  NC  MNC   OPRD
140   26  WIL      5   CRITERE DE DECLENCHEMENT    ) PHASE DE DEROULEMENT
141   14  STR    26Ø   ACTION                      ) A
142   ØØ  ØØ     12Ø
143   25  WIH    26Ø   CRITERE DE DECLENCHEMENT    ) PHASE
144   11  SEO     37   ACTION                      ) B
145   25  WIH      5   CRITERE DE DECLENCHEMENT    ) PHASE
146   12  REO     37   ACTION                      ) C
147   20  JMP    14Ø->
```

Exemple 11: Combinaison ET selon diagramme de flux

Ensemble de combinaisons et de diagramme de flux (sans boucle d'attente)

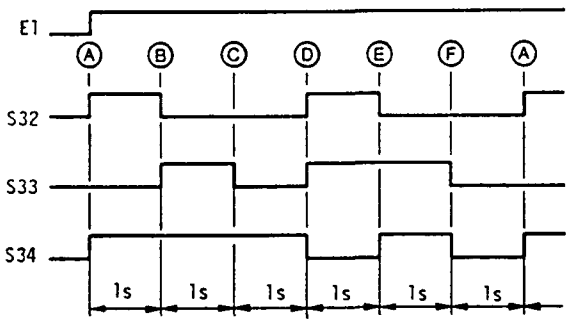
Basé sur schéma à contacts resp. de fonctions logiques:

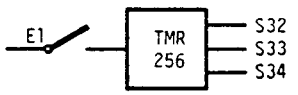


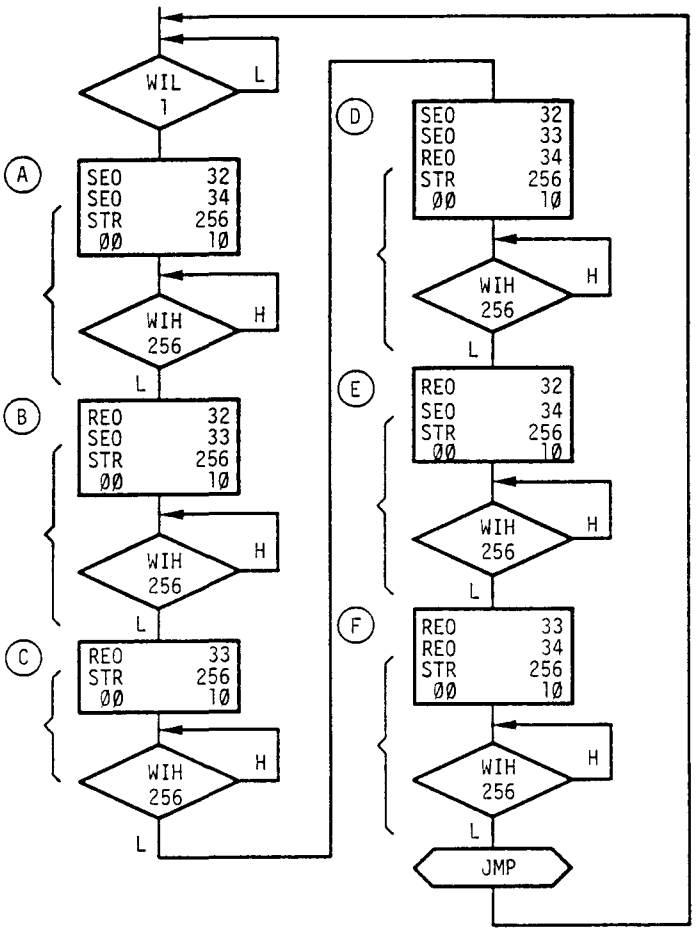



```
ADDR NC   MNC   OPRD
 150 01   STH     4   SCRUTATION DE E4
 151 03   ANH     6   COMBINAISON - ET
 152 03   ANH     1   COMBINAISON - ET
 153 22   JIZ   156-> SI 0, SAUTE A L'ADRESSE DE PAS 156
 154 11   SEO    50  SI 1, POSITIONNE LA SORTIE 50
 155 20   JMP   150-> SAUT DE RETOUR AU DEBUT DU PROGRAMME
 156 12   REO    50  REMET LA SORTIE 50
 157 20   JMP   150-> SAUT DE RETOUR AU DEBUT DU PROGRAMME
```

Exemple 12: Contrôle de déroulement avec ou sans sous-programme (subroutine)

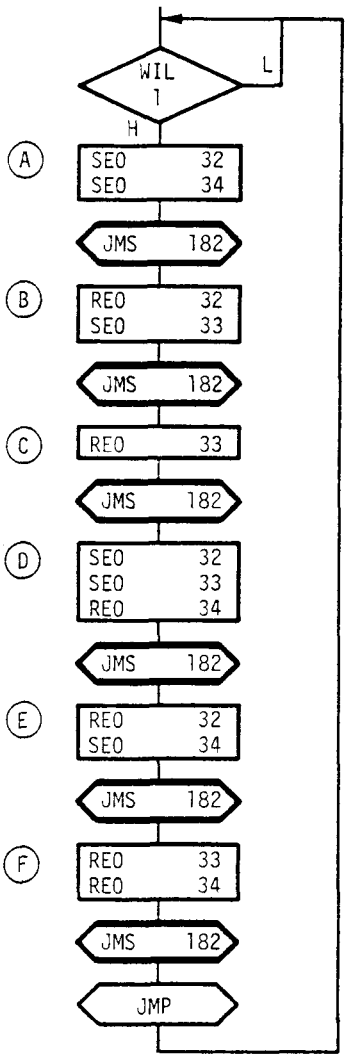






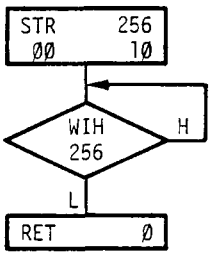
Le diagramme de flux démontre le déroulement du programme sans sous-programme. Les parties de programme désignés par une paranthèse sont répétées 6 fois et s'adaptent ainsi parfaitement à la programmation en sous-programme.

Programme principal




```
******************** PROGRAMME PRINCIPAL
ADDR NC   MNC   OPRD
 160 26   WIL      1   ATTEND, SI E1 OUVERT
 161 11   SEO     32
 162 11   SEO     34
 163 23   JMS    182=>  SAUTE AU SOUS-PROGRAMME 182
 164 12   REO     32
 165 11   SEO     33
 166 23   JMS    182=>  SAUTE AU SOUS-PROGRAMME 182
 167 12   REO     33
 168 23   JMS    182=>  SAUTE AU SOUS-PROGRAMME 182
 169 11   SEO     32
 170 11   SEO     33
 171 12   REO     34
 172 23   JMS    182=>  SAUTE AU SOUS-PROGRAMME 182
 173 12   REO     32
 174 11   SEO     34
 175 23   JMS    182=>  SAUTE AU SOUS-PROGRAMME 182
 176 12   REO     33
 177 12   REO     34
 178 23   JMS    182=>  SAUTE AU SOUS-PROGRAMME 182
 179 20   JMP    160->
```

Sous-programme "182"



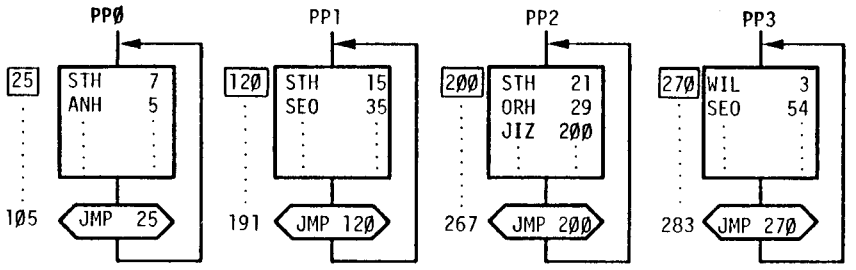

```
==================== SOUS-PROGRAMME 182 (ATTEND 1S)
 182 14   STR    256
 183 00    00     10
 184 25   WIH    256
 185 24   RET      0->
```

Exemple 13: Utilisation de programmes parallèles

Le programme de commande d'une installation avec 2 manipulateurs et une table rotative est divisé en 4 programmes parallèles.

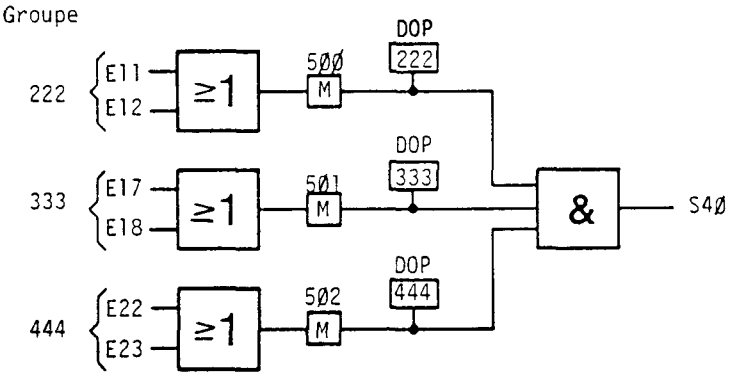
Progr. parallèle Ø	Surveillance	Adresse de pas de début 25
Progr. parallèle 1	Manipulateur 1	Adresse de pas de début 12Ø
Progr. parallèle 2	Manipulateur 2	Adresse de pas de début 2ØØ
Progr. parallèle 3	Table rotative	Adresse de pas de début 27Ø

Programme parallèle




```
******************** ASSIGNATION DES PROGRAMMES PARALLELES
ADDR  NC   MNC   OPRD
   0  29   PAS      1   ASSIGNATION PP1
   1  00    00    120
   2  29   PAS      2   ASSIGNATION PP2
   3  00    00    200
   4  29   PAS      3   ASSIGNATION PP3
   5  00    00    270
   6  20   JMP     25 -> PPO EST ACTIVE DIRECTEMENT

+++++++++++++++++++ PROGRAMME PARALLELE 0 (SURVEILLANCE)
 ┌► 25  01   STH      7
 │   .   .    .       .
 │   .   .    .       .
 │   .   .    .       .
 └─105  20   JMP     25 ->

+++++++++++++++++++ PROGRAMME PARALLELE 1 (MANIPULATEUR 1)
 ┌►120  01   STH     15
 │   .   .    .       .
 │   .   .    .       .
 │   .   .    .       .
 └─191  20   JMP    120 ->

+++++++++++++++++++ PROGRAMME PARALLELE 2 (MANIPULATEUR 2)
 ┌►200  01   STH     21
 │   .   .    .       .
 │   .   .    .       .
 │   .   .    .       .
 └─267  20   JMP    200 ->

+++++++++++++++++++ PROGRAMME PARALLELE 3 (TABLE ROTATIVE)
 ┌►270  01   STH      3
 │   .   .    .       .
 │   .   .    .       .
 │   .   .    .       .
 └─283  20   JMP    270 ->
```

Exemple 14: Circuit de surveillance avec affichage d'erreurs sur le clavier de programmation



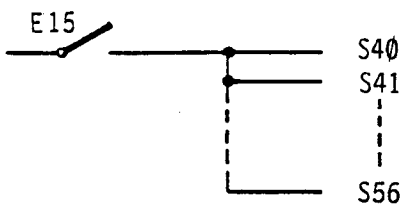
Les entrées des groupes 222, 333, 444 sont surveillées. Si une des conditions n'est pas remplie, la lampe à la sortie 4Ø s'éteint. Le groupe défectueux est alors affiché sur le clavier de programmation.

```
ADDR  NC   MNC   OPRD
 300  01   STH     11
 301  05   ORH     12
 302  10   OUT    500
 303  30   DOP    222   AFFICHAGE SI COMBINAISON PAS REALISEE (ACCU = 0)
--------------------
 304  01   STH     17
 305  05   ORH     18
 306  10   OUT    501
 307  30   DOP    333   AFFICHAGE SI COMBINAISON PAS REALISEE
--------------------
 308  01   STH     22
 309  05   ORH     23
 310  10   OUT    502
 311  30   DOP    444   AFFICHAGE SI COMBINAISON PAS REALISEE
--------------------
 312  01   STH    500
 313  03   ANH    501
 314  03   ANH    502
 315  10   OUT     40
 316  20   JMP    300->
```

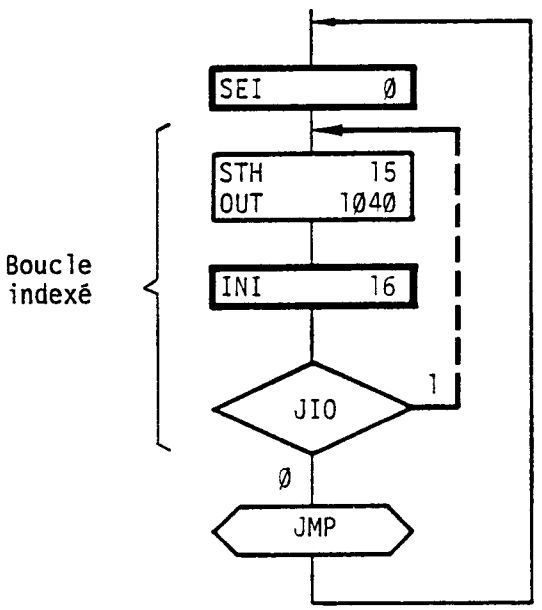
Exemple 15: Commutation de plusieurs sorties en séries

1er exemple avec indexation d'adresses.

Les sorties 4Ø...56 sont positionnées resp. remises par l'entrée 15.



Nombre de pas d'indexage = 56 - 4Ø = 16




```
ADDR NC   MNC   OPRD
 350 16   SEI      0   POSITIONNER REGISTRE D'INDEX SUR VALEUR DE DEPART 0
 351 01   STH     15   SCRUTATION DE L'ENTREE 15
 352 10   OUT   1040   POSITIONNER LA SORTIE INDEXEE (40 + 1000 = 1040)
 353 27   INI     16   INCREMENTER REGISTRE D'INDEX JUSQU'A LA VALEUR FINAL 16
 354 21   JIO   351->  REPETER L'INDEXATION JUSQU'AU TRAITEMENT DE TOUTES
 355 20   JMP   350->                              LES SORTIES S40...S56
```

autre sol.

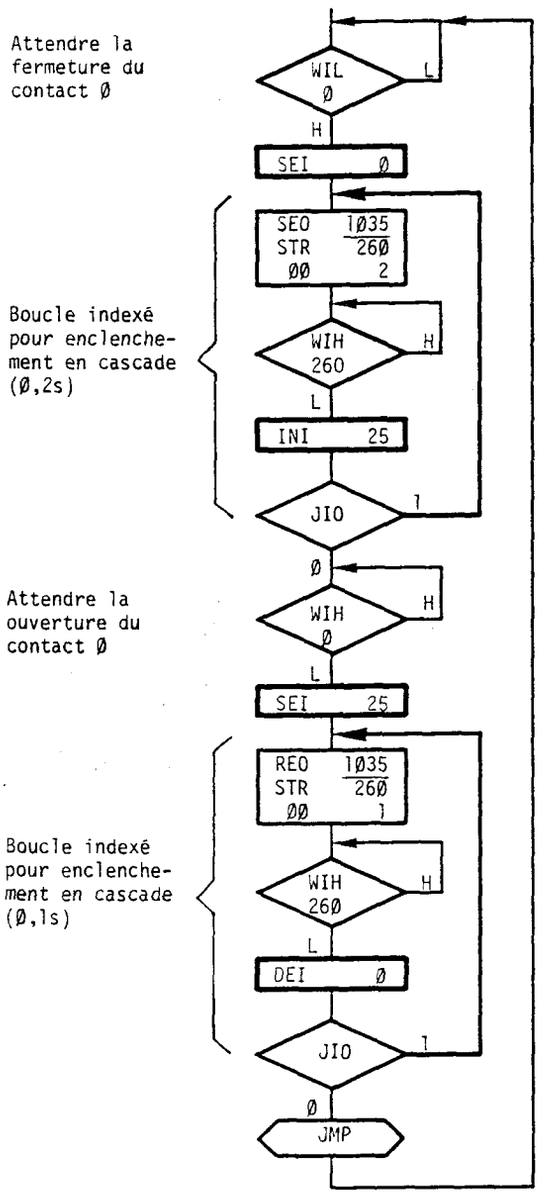
350	SEI	16
351	STH	15
	OUT	1040
	DEI	0
	JIO	351
	JMP	350

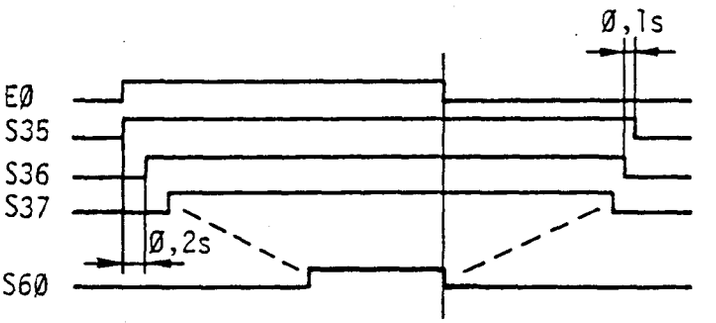
Exemple 16: Enclenchements et déclenchements décalés de plusieurs sorties

2ème exemple avec indexation d'adresses.

Si l'entrée Ø est fermée, les sorties 35...6Ø doivent être enclenchées une après l'autres au rythme de Ø,2s.

A l'ouverture de l'entrée Ø, les sorties 6Ø...35 doivent être déclenchées l'une après l'autre au rythme de Ø,1s.

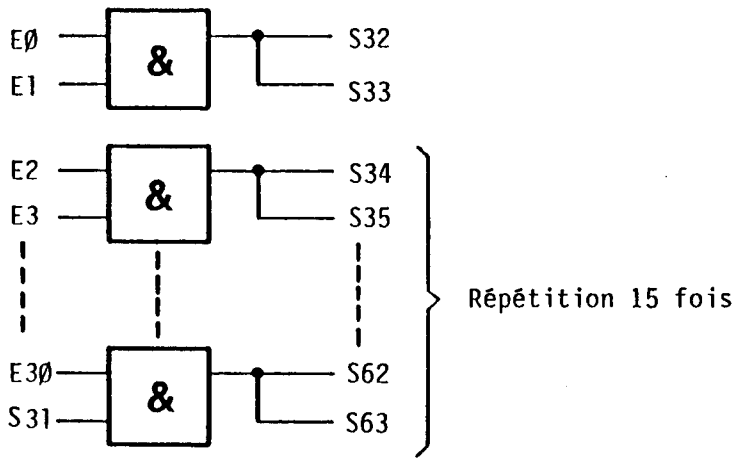





```
ADDR NC   MNC  OPRD
-------------------  SEQUENCE D'ENCLENCHEMENT
 360 26   WIL     0
 361 16   SEI     0
 362 11   SEO  1035   35 + 1000 = 1035
 363 14   STR   260
 364 00   00      2
 365 25   WIH   260
 366 27   INI    25   60 -   35 =   25
 367 21   JIO   362->
-------------------  SEQUENCE DE DECLENCHEMENT
 368 25   WIH     0
 369 16   SEI    25   60 -   35 =   25
 370 12   REO  1035   35 + 1000 = 1035
 371 14   STR   260
 372 00   00      1
 373 25   WIH   260
 374 28   DEI     0
 375 21   JIO   370->
 376 20   JMP   360->
```

Exemple 17: Petit circuit de surveillance

3ème exemple avec indexation d'adresse



Programme principal

STH	Ø
ANH	1
OUT	32
OUT	33

Dans cet exemple le programme principal est entièrement indexé.

```
ADDR NC   MNC   OPRD
 380 16   SEI     0
 381 01   STH   1000
 382 03   ANH   1001
 383 10   OUT   1032
 384 10   OUT   1033
 385 27   INI     30   ) INCREMENTATION DOUBLE, PARCE QUE L'INDEX
 386 27   INI     30   ) EST A INCREMENTER DE 2 PAS (62 - 32 = 30)
 387 21   JIO    381->
 388 20   JMP    380->
```

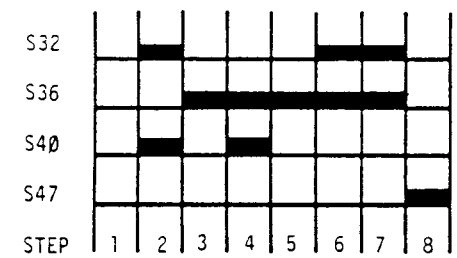
Exemple 18: Programme d'un commutateur

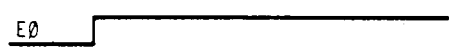
- avec vitesse variable
- avec mode de fonctionnement MAN/AUTO
- avec affichage du pas

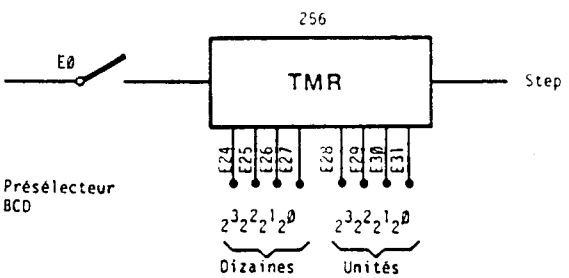
Enoncé: En fermant le commutateur EØ le programme suit les états indiqués sur le diagramme ci-dessous. Le temps entre les "PAS" est sélectionné par le codeur BCD E24...31, et il est règlable de Ø,1 à 9,9 s. Si le commutateur E4 est fermé, le poussoir E7 permet le défilement du programme en PAS à PAS. Le numéro de PAS est indiqué sur l'afficheur d'opérandes.

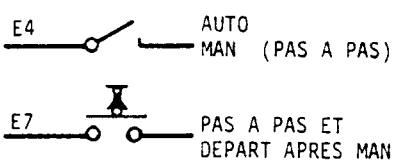
Solution: Le déroulement du programme est contenu dans les adresses 4Ø2...425. La sous-routine 43Ø contient la commande de PAS à PAS ainsi que la marche séquentielle automatique. Le compteur 28Ø sert au comptage des PAS. Pour que l'afficheur puisse travailler simultanément, on a constitué le programme parallèle P1 en 45Ø.

Diagramme:









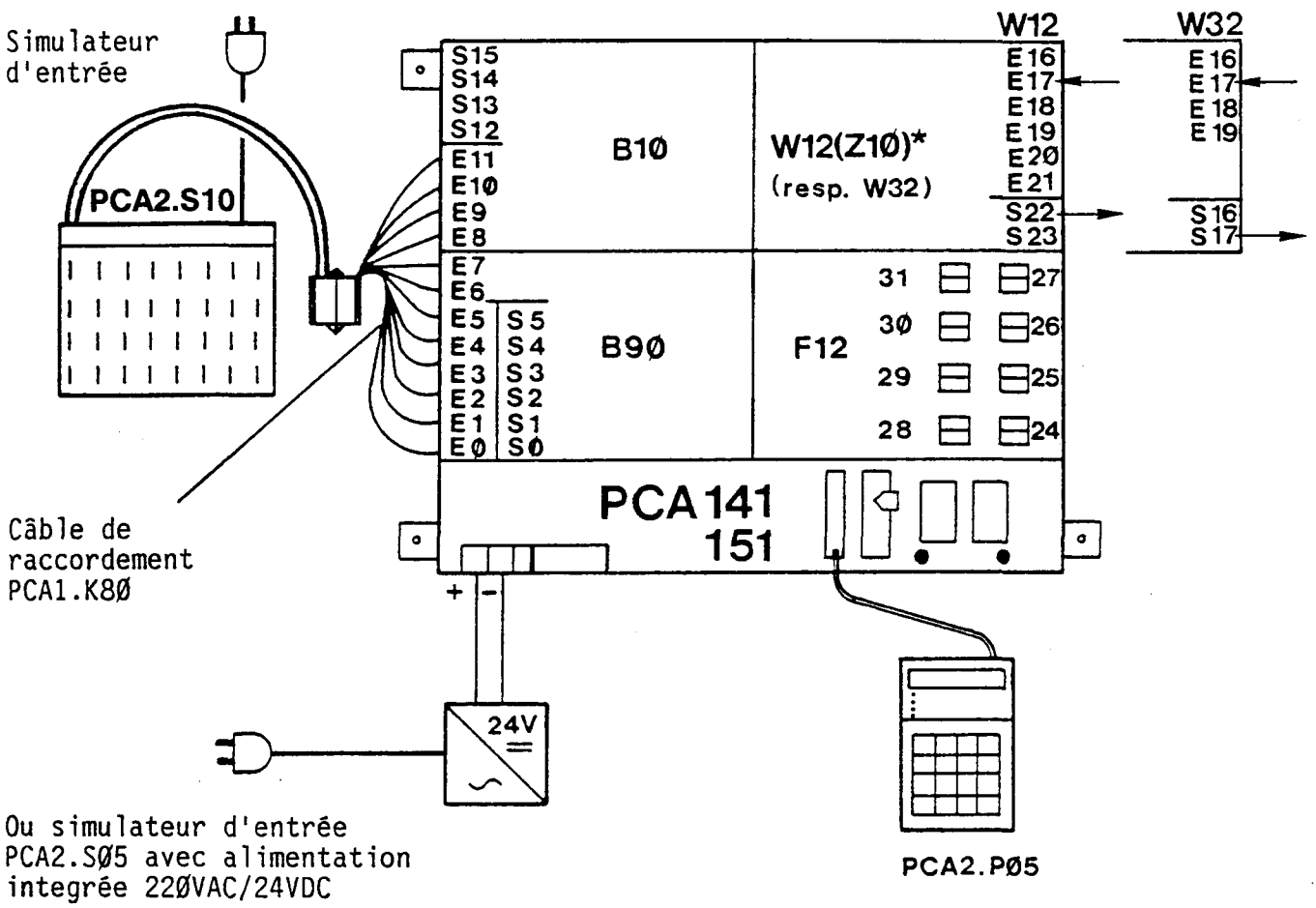

```
******************** ASSIGNATION DU:
 400 29  PAS      1   PROGRAMME PARALLELE 1
 401 00   00    450
******************** PROGRAMME PRINCIPAL PPO
 402 26  WIL      0   DEPART DU DEROULEMENT DU PROGRAMME
 403 15  SCR    280   COMPTEUR DE PAS
 404 00   00      1
-------------------- STEP 1
 405 23  JMS    430=>  SAUTE AU SOUS-PROGRAMME 430
-------------------- STEP 2
 406 11  SEO     32
 407 11  SEO     40
 408 23  JMS    430=>
-------------------- STEP 3
 409 12  REO     32
 410 12  REO     40
 411 11  SEO     36
 412 23  JMS    430=>
-------------------- STEP 4
 413 11  SEO     40
 414 23  JMS    430=>
-------------------- STEP 5
 415 12  REO     40
 416 23  JMS    430=>
-------------------- STEP 6
 417 11  SEO     32
 418 23  JMS    430=>
-------------------- STEP 7
 419 23  JMS    430=>
-------------------- STEP 8
 420 12  REO     32
 421 12  REO     36
 422 11  SEO     47
 423 23  JMS    430=>
--------------------
 424 12  REO     47
 425 20  JMS    402-> DEBUT

 ================== SOUS-PROGRAMME 430
 430 02  STL      4  MAN / AUTO
 431 22  JIZ    437->
 -------------------- TEMPORISATEUR VARIABLE (AUTO)
  432 14  STR    256
  433 16   16     31
  434 25  WIH    256
  435 17  INC    280
  436 24  RET      0->
 -------------------- OPERATION MAN
 437 26  WIL      7
  438 25  WIH      7
  439 17  INC    280
  440 24  RET      0->

 ++++++++++++++++++ PP1 : AFFICHAGE
 450 31  DTC    280
 451 20  JMP    450->
```

Exemples de programmation avec modules analogiques

Les adresses E/S ont été sélectionnées de manière que tous les exemples puissent être éxécutés avec la même configuration.

Tous les exemples suivants peuvent être programmés avec le module 8 bits PCA1.W12 et également le module 12 bits PCA1.W32.

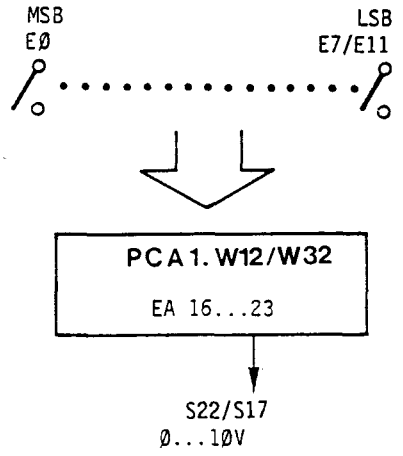
Cette configuration se présente comme suit:



*) Le module PCA1.W12 Z1Ø a une tension d'entrée de Ø...1ØV (au lieu de Ø...5V).

Exemple 19: Sortie d'une tension analogique à partir de 8 resp. 12 entrées

La valeur binaire, formée par les 12 entrées EØ...E7/E11, est à sortir sur la sortie analogique du canal S22 respectivement S17.



Avec module PCA1.W12 (8 bits resp. 7 bits)

```
4Ø   DTC   3Ø1    Affichage de la valeur binaire
     SCR   3Ø1    Entrée Ø à 7 sur compteur 3Ø1
      24    7
     SCR   3Ø1    Mettre le niveau du canal
      21    23    de sortie S22 en rapport
     SEO    23    du contenu du compteur
     ORH    22    C3Ø1
     REO    23
     JMP    4Ø
```

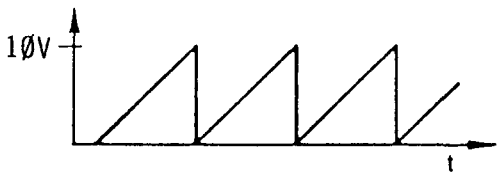
Avec module PCA1.W32 (12 bits)

```
5Ø   DTC   3Ø1    Affichage de C3Ø1
     SCR   3Ø1    Transfert des entrées
      25    11    sur le compteur C3Ø1
1)   JMS   7ØØ    --> SR pour "la sortie de
     JMP   5Ø     la valeur analogique"
```

1) Sous-routine voir "manuel Hardware PCA1" sous Module PCA1.W32 ou W2..

Exemple 2Ø: Sortie d'une tension en dents de scie

Il s'agit de générer une tension en dents de scie en incrémentant une valeur binaire à 8 bits respectivement 12 bits.



Pour le module W12 t est donné par le timer 256 et pour le module W32, t (pour 4Ø96 passages) est dépendant du sous-programme choisi et du nombre de programmes parallèles assignés.

Avec module PCA1.W12 (8 resp. 7 bits)

```
6Ø   SEI     7      }
  ┌─►COO   15ØØ      }
  │  STH   15ØØ      } Incrémentation de
  │ ┌JIO     66      } la valeur
  │ │DEI      Ø      }
  │ └─JIO     61
  │  ►SCR    3Ø1     } Charger la valeur binaire
  │   24     5Ø7     } dans le compteur C3Ø1
  │  SCR     3Ø1     }
  │   21      23     } Transfert du contenu de
  │  SEO      23     } C3Ø1 sur le canal de
  │  ORH      22     } sortie analogique A22
  │  REO      23     }
  │  STR     256     }
  │   ØØ       4     } Attendre Ø,Ø4s
  │  WIH     256     }
  └──JMP      6Ø
```

Avec module PCA1.W32 (12 bits)

```
8Ø   SEI     11      }
  ┌─►COO   15ØØ      }
  │  STH   15ØØ      } Incrémenter le
  │ ┌JIO     86      } compteur binaire
  │ │DEI      Ø      }
  │ └─JIO     81      }
  │  ►SCR    3Ø1     } Charger la valeur
  │   25     511     } binaire sur C3Ø1
  │ 1) JMS   7ØØ     --> SR pour la "sortie
  └──JMP      8Ø     de la valeur analogique"
                     (aussi IMS 6ØØ ou 65Ø)
```

1) Pour les sous-programmes voir "manuel Hardware PCA1" (module PCA1.W32 ou W2..).

Exemple 21: Lecture de valeurs BCD du module PCA1.F12 et transfert sur une sortie analogique du module PCA1.W12 (8 bits) resp. W32 (12 bits)

La valeur BCD à 2 chiffres du canal S24 (F12) doit être transférée toutes les 2s sur le canal S22 respectivement S17 du module analogique. La valeur binaire correspondante doit être affichée dans le champ opérand du clavier de programmation.

5,ØV ≙ valeur binaire 128 (resp. 2Ø48)
| 5 | Ø | ≙ 1Ø,ØV ≙ valeur binaire 256 (resp. 4Ø96)

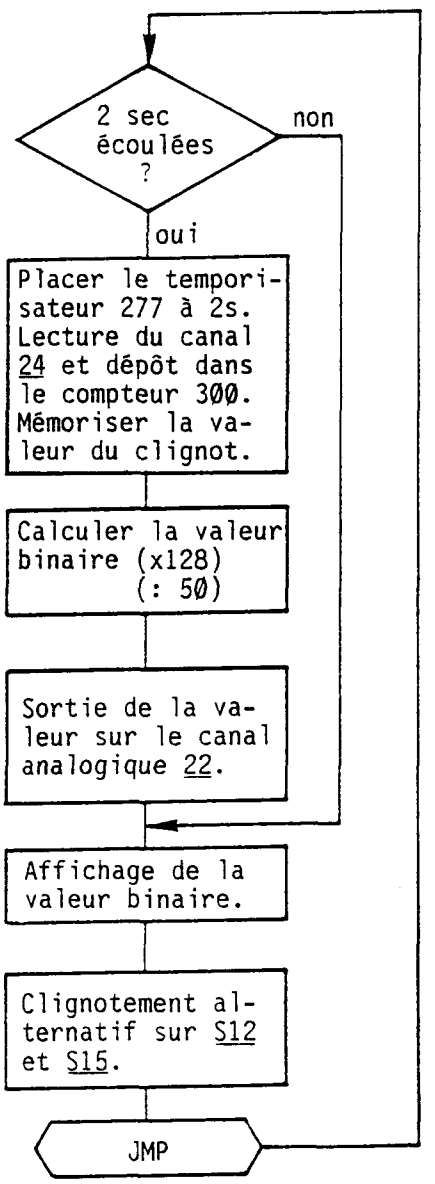
Les sorties S5 et S12 doivent clignoter alternativement selon la valeur BCD présélectionnée (en tant que 1/1Øs).

Programme
pour PCA1.W12 (8 resp. 7 bits)

```
2ØØ  Ø1   STH    277
2Ø1  21   JIO    227

2Ø4  14   STR    277
2Ø5  ØØ    ØØ     2Ø
2Ø6  11   SEO     24
2Ø7  15   SCR    3Ø1
2Ø8  16    16     31
2Ø9  12   REO     24
21Ø  15   SCR    3Ø2
211  31    31    3Ø1

214  15   SCR    3Ø1
215  29    29    128
216  15   SCR    3Ø1
217  3Ø    3Ø     5Ø

22Ø  15   SCR    3Ø1
221  21    21     23
222  11   SEO     23
223  Ø5   ORH     22
224  12   REO     23

227  31   DTC    3Ø1

23Ø  Ø2   STL    278
231  14   STR    278
232  31    31    3Ø2
233  13   COO     12
234  Ø2   STL     12
235  1Ø   OUT      5
236  2Ø   JMP    2ØØ
```

*

Schéma fonctionnel

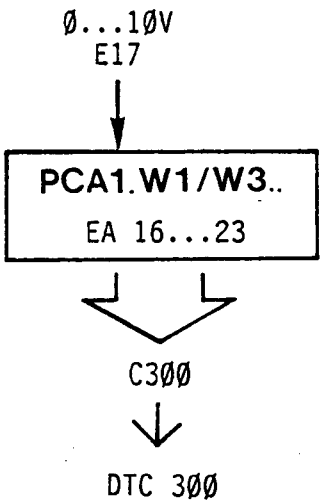


---

*) Pour PCA1.W32 (12 bits)

```
214   15   SCR    3Ø1   } Valeur BCD x 41
215   29    29     41   } (1ØV ≙ 1ØØ ≙ 4Ø96 resp. 41ØØ)
216   23   JMS    7ØØ   -->Pour les sous-programmes voir module Hardware
                          PCA1.W32 ou W2..
```

Exemple 22: Introduction par lecture d'une tension analogique dans un registre compteur et affichage de la valeur binaire avec DTC



Avec module PCA1.W1.. (8 bits)

```
9Ø   OUT   17   } Lire la valeur analogique
     SCR   3ØØ   et la placer dans le C3ØØ
      24    23  }
     DTC   3ØØ   Afficher la valeur binaire
     JMP    9Ø
```

Avec module PCA1.W3.. (12 bits)

```
1ØØ  JMS   5ØØ  ⇒ Lecture de la valeur ana-
                   logique et la placer dans
                   le C3ØØ *)
     DTC   3ØØ   Afficher la valeur binaire
     JMP   1ØØ
```

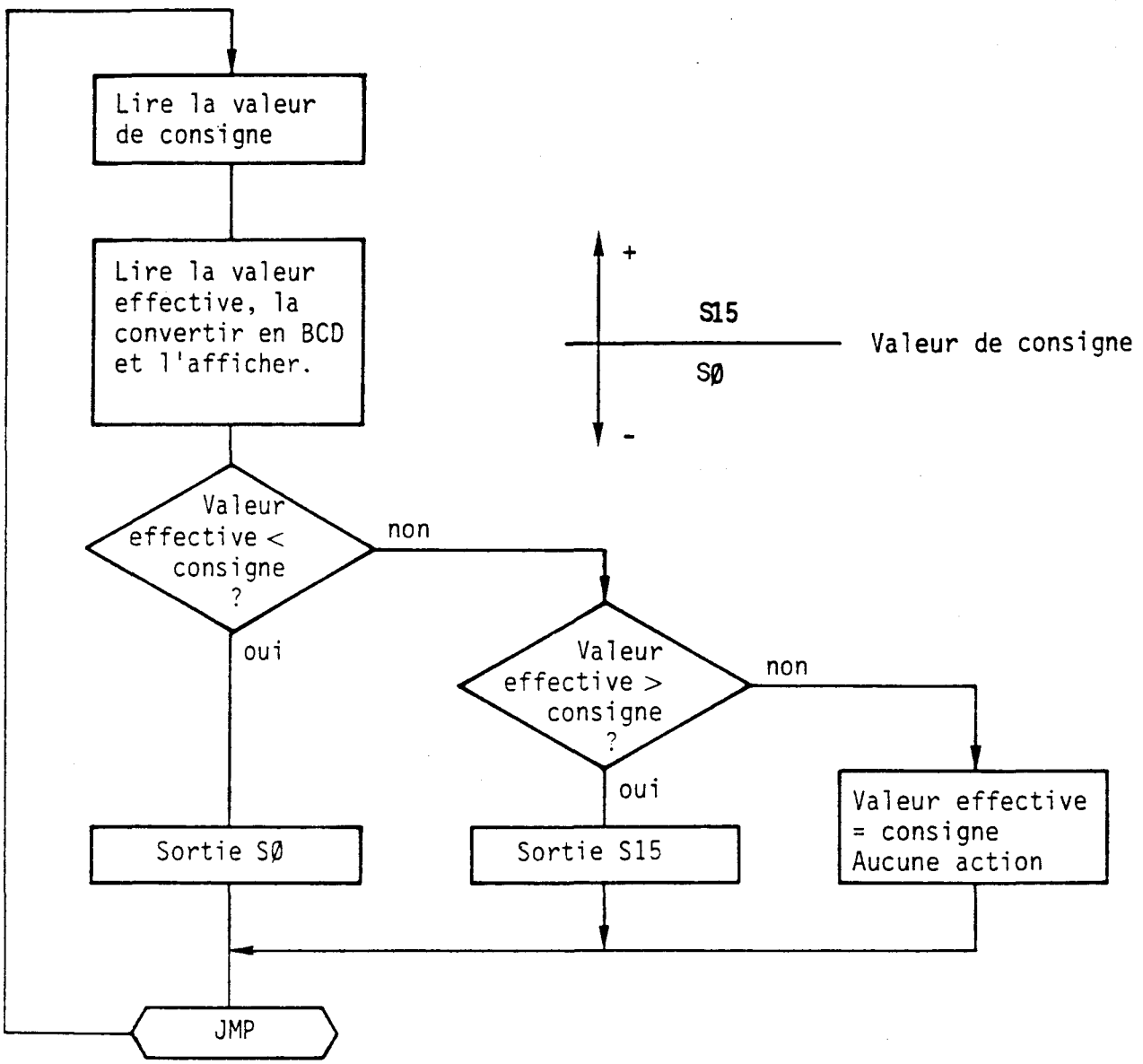
*) Pour les sous-programmes voir "manuel Hardware PCA1" (module PCA1.W32).

Exemple 23: Régulateurs 2 et 3 points (avec module analogique PCA1.W1.. resp. W3..)

Problème 23a: Régulateur 2 points

Une valeur limite doit être présélectionnée sur le codeur BCD à 2 chiffres du canal S24. La valeur effective est entrée par le canal 17 du module analogique. Si celle-ci est inférieure à la valeur de consigne, la sortie SØ est activée. Par contre si elle est supérieure à la consigne, c'est la sortie S15. La valeur effective doit être affichée dans le champ opérand en tant que valeur BCD (ØØ...99 ≙ Ø...9,9V).

Schéma fonctionnel



Solution 23a:

Pour PCA1.W1.. (8 bits)

```
400   11   SEO    24
401   15   SCR    310
402   16    16    31
403   12   REO    24

406   10   OUT    17
407   15   SCR    312
408   24    24    23

*411  15   SCR    312
412   29    29    50
413   15   SCR    312
414   30    30    128
415   31   DTC    312

418   15   SCR    310
419   28    28    312
420   22   JIZ    430   ;
421   01   STH    310
422   21   JIO    435   ;

425   12   REO    0
426   12   REO    15
427   20   JMP    400

430   12   REO    0
431   11   SEO    15
432   20   JMP    400

435   12   REO    15
436   11   SEO    0
437   20   JMP    400
```

Valeur de consigne en BCD dans le compteur C310

Valeur effective en binaire dans C312

Conversion de la valeur effective en BCD (x50, :128) et affichage

Comparaison de la valeur effective avec la consigne

Valeur effective > consigne

Valeur effective < consigne

Valeur effective = consigne: aucune action

Valeur effective > consigne: sortie S15

Valeur effective < consigne: sortie S0

---

*) Pour module PCA1.W3..

```
406   15   SCR    310
407   29    29    41

411   23   JMS    500   =>
412   15   SCR    312
413   31    31    300
414   31   DTC    312
```

Conversion BCD ---> binaire
BCD * 41 (10V ≙ 100 ≙ 4096 resp. 4100)

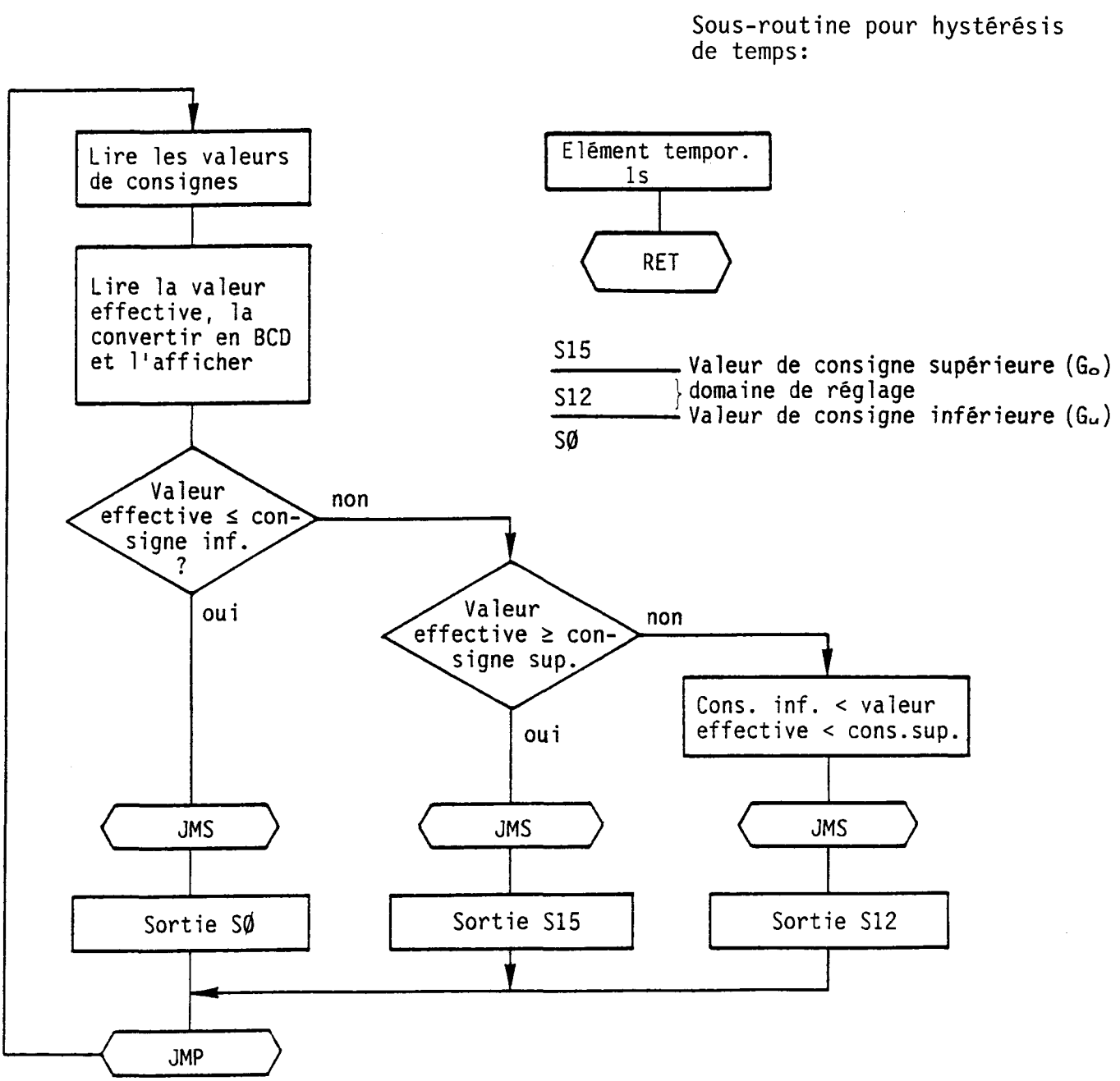
transfert valeur effective en binaire C300

Copier C300 dans C312
Afficher C312
c'est à dire la valeur effective

Problème supplémentaire 23b: Régulateur 3 points avec hystérésis de temps

Les valeurs de consignes inférieures et supérieures d'un régulateur 3 points doivent être entrées par les codeurs BCD à 2 chiffres des canaux S24 et S27. La valeur effective est lue sur le canal analogique E17. Si celle-ci est inférieure à la valeur de consigne inférieure, la sortie SØ doit être activée. Par contre si elle est supérieure à la valeur de consigne supérieure, c'est la sortie S15 qui sera activée. Dans le cas où elle se situe entre les consignes, c'est la sortie S12 qui est activée. Une temporisation de 1 sec servant d'hystérésis de temps est introduite entre chaque lecture. La valeur effective doit être affichée dans le champ opérand en format BCD.

Schéma fonctionnel



Solution 23b

Pour PCA1.W1.. (8 bits)

```
      300   11   SEO    24
      301   15   SCR   310   } Lire la valeur de consigne inférieure en
      302   16    16    31   } BCD dans le compteur C310
      303   12   REO    24

      305   11   SEO    27
      306   15   SCR   311   } Lire la valeur de consigne supérieure en
      307   16    16    31   } BCD dans le compteur C311
      308   12   REO    27

      310   10   OUT    17
      311   15   SCR   312   } Lire la valeur effective en binaire
      312   24    24    23   } dans le compteur C312

**    314   15   SCR   312
      315   29    29    50
      316   15   SCR   312   } Conversion de la valeur effective en
      317   30    30   128   } BCD (x50; :128) et affichage
      318   31   DTC   312

      321   15   SCR   310
      322   28    28   312   } Valeur effective ≤ consigne inférieure?
      323   22   JIZ   331
      324   23   JMS   350   ==> Hystérésis de temps
      325   12   REO    15
      326   12   REO    12
      327  111   SEO     0   ;  Sortie S0
      328   20   JMP   300

      331   15   SCR   312
      332   28    28   311   } Valeur effective ≥ consigne supérieure?
      333   22   JIZ   340
      334   23   JMS   350   ==> Hystérésis de temps
      335   12   REO     0
      336   12   REO    12
      337   11   SEO    15
      338   20   JMP   300   ;  Sortie S15

      340   23   JMS   350   ==> Hystérésis de temps
      341   12   REO     0
      342   12   REO    15
      343   11   SEO    12   ;  Sortie S12 (consigne inf. < valeur effective < consigne sup.)
      344   20   JMP   300
```

Sous-routine "hystérésis de temps":

```
      350   14   STR    284
      351   00    00     10
      352   25   WIH    284 *
      353   24   RET     0
```

*) Si l'hystérésis de temps est supérieure à une seconde, la valeur effective n'est pas constamment affichée à cause de l'instruction d'attente.

**) Pour le module PCA1.W3.. il faut utiliser la même sous-routine que dans l'exemple précedent les adresses 406...414 doivent être adaptées aux adresses 310...318.

Exemple 24: Exemple pratique d'une installation automatique de perçage

Enoncé

Une installation automatique de perçage est déterminée par le plan mécanique et par le diagramme de pas. Une partie de la zone utilisateur est également définie. Son confort dépend par contre des possibilités du PLC choisi.

Description générale des fonctions (voir schéma de la page suivante)

Transporter automatiquement des rondelles, les percer excentriquement et les éjecter.

PAS 1: Pousser les rondelles du magazin par le piston A en position de perçage et les fixer.
PAS 2: Moteur "ON" et abaisser la perçeuse.
PAS 3: Après approx. 4s elever la perçeuse, pour enlever les copeaux du perçage.
PAS 4: Abaisser à nouveau la perçeuse jusqu'à ce que la profondeur de perçage soit atteinte.
PAS 5: Elever la perçeuse.
PAS 6: Moteur "OFF" et reculer le piston de chargement A.
PAS 7: Avancer le piston éjecteur C.
PAS 8: Reculer le piston éjecteur C.
Recommencer au PAS 1.

Détecteurs

Toutes les positions finales des pistons sont reportées par détecteurs. Pour des raisons de securité contre rupture de fil on utilisera des sectionneurs pour de telles fonctions d'arrêt. Nous avons choisi des contacts de travail pour faciliter la simulation de ces fonctions.

Le niveau minimum du magazin est également contrôlé par des détecteurs. Le foret doit être contrôlé afin de détecter une rupture avant chaque cycle et, si necéssaire, l'appareil doit être arrêté.

Zone utilisateur

En plus des fonctions "Initialisation" et "Arrêt du cycle" une remise du programme en position initiale ("Reset du programme") doit être possible. Une action simultanée des touches "Stop" et "Reset" est nécessaire ceci pour éviter une remise involontaire.

L'arrêt d'urgence se fait du coté "Hardware" par la touche champignon directement sur le contacteur principal (prescription de securité).

Une présélection d'unité (1ØØ...1Ø'ØØØ) par ordre doit être possible par des présélecteurs BCD. L'introduction resp. la réintroduction du nombre présélectionné d'unités doit se faire par un bouton poussoir.

Un traitement pas à pas du programme est à prévoir pour la mise en service ainsi que pour le service.

L'affichage ne sert non seulement à afficher le nombre de pièces restantes mais aussi en cas d'erreur, d'afficher le pas dans lequel la machine est arrêtée resp. le no de la fonction attendue.

Les diverses fonctions sont à signaler par des lampes selon le dessin ci-dessous.

Dessin représentant la machine au PAS 2

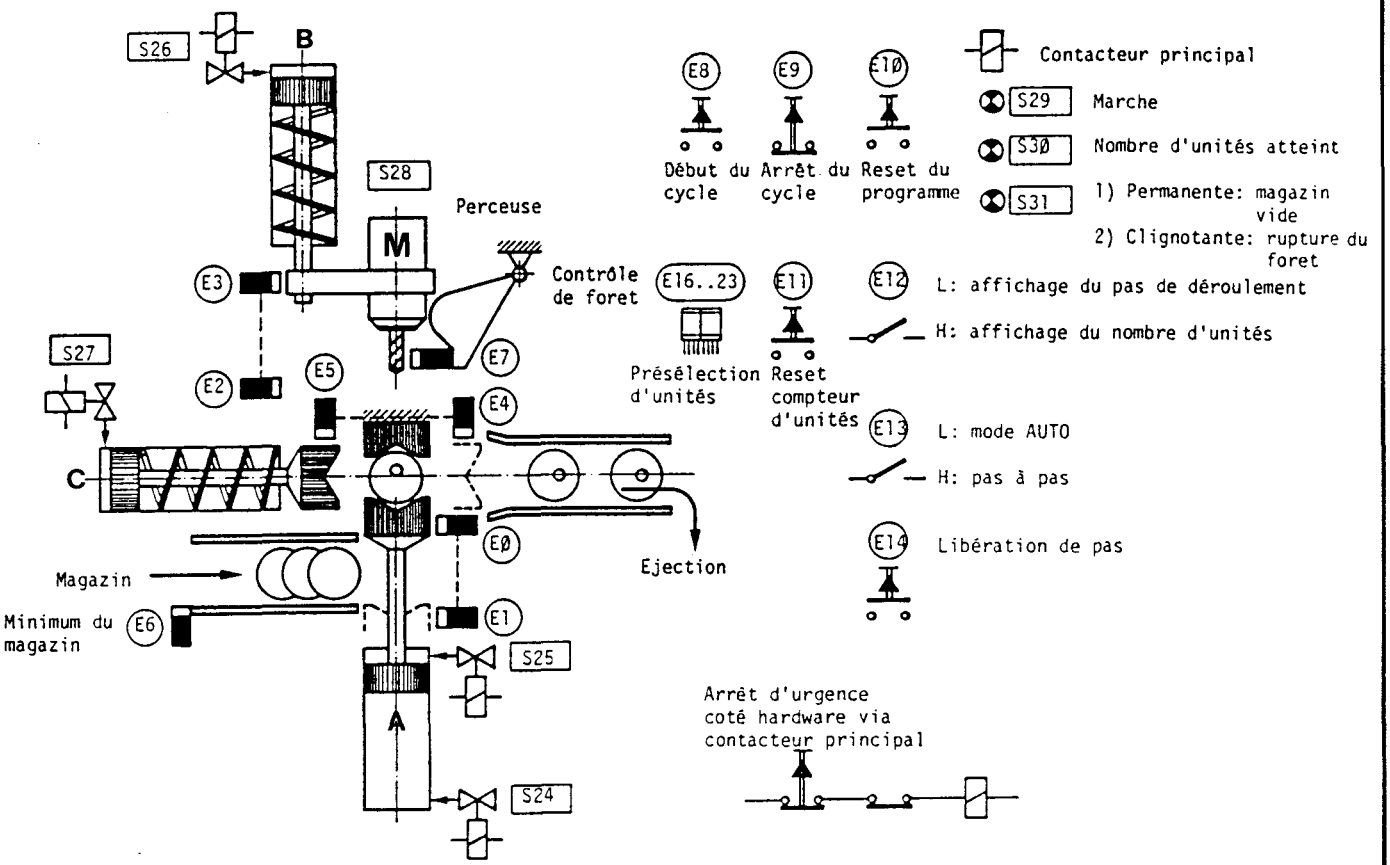
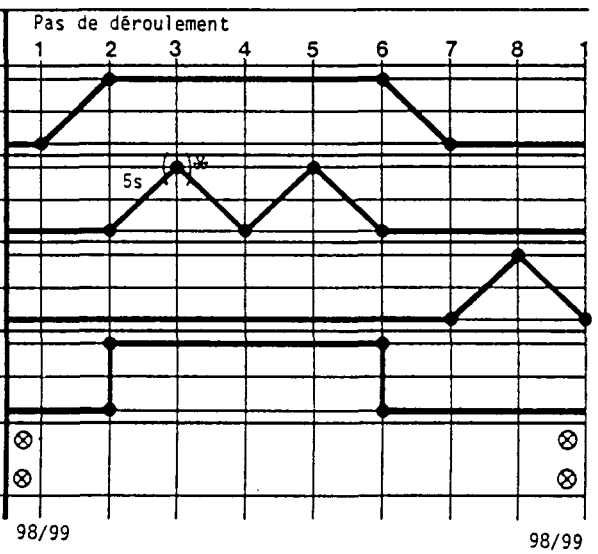


Diagramme représentant les divers pas

Action	Piston		Sortie	Signal
Introduire une pièce	A	avancer	S24	E0
		reculer	S25	E1
Abaisser la perceuse	B	en bas	S26	E2
		en haut	-	E3
Ejecter la pièce	C	avancer	S27	E4
		reculer	-	E5
Moteur de perceuse	M	ON	S28	-
		OFF	-	-
Contrôle du foret				E7
Contrôle du magazin				E6



* Position final E2 atteinte ou en altérnative percer pour 5s.

24.1 Dimensionnement du PLC

24.1.1 Fonctions

Pour réaliser les fonctions demandés, il faut à part des combinaisons logiques, le contrôle du déroulement ainsi que des fonctions de temporisation et de comptage. En plus il est nécessaire d'afficher le contenu des compteurs. Tout SAIA°PLC peut sans autre remplir ces devoirs.

24.1.2 Nombre d'E/S

Le petit nombre d'E/S permet l'utilisation d'un PCA1. Les E/S sont à définir sur le formulaire prévu à cet effet. A noter que les E/S sont à diviser en groupe de 8 ou de 4 E/S si nécessaire.

L'affichage du contrôle du magazin resp. du foret manque dans le cas présent. Solution adoptée: Vue que ces affichages arriveront rarement, nous nous servirons de la même sortie. Les deux affichages vont différer entre un clignotement et une illumination permanente. Ainsi la capacité d'un PCA151 est suffisante.

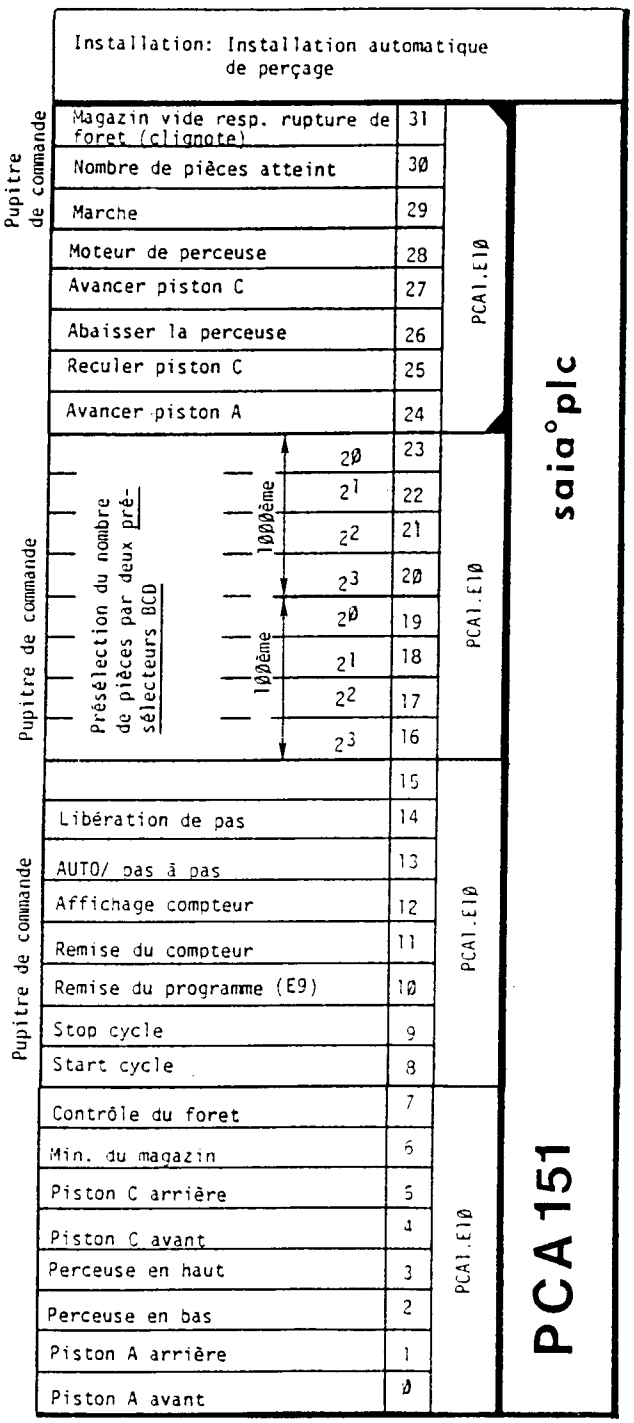
24.1.3 Types d'E/S

Nous choisissions les modules E/S 24VDC, sans séparation galvanique, avec transistor de sorties et nous choisissons les bobines magnétiques et les lampes correspondantes.

24.1.4 Capacité de mémoire

Vu la complexité peu importante de ce programme nous pouvons calculer avec un facteur de complexité de 5 approx. Si nous calculons 1E pour les 8E du présélecteur BCD le résultat sera:
24 E+S x facteur de complèxité 5
= 12Ø places de mémorisation.
Une mémoire de 1K suffira largement.

Nous utiliserons un interface externe avec relais pour le moteur 22ØVAC de la perceuse.
Du côté des entrées, le 24VDC offre l'avantage de pouvoir utiliser, comme source d'information, des détecteurs de proximité.



Formulaire d'attribution des E/S

24.1.5 Affichage

L'utilisation du module PCA1.D11 permet les affichages nécessaires sans perte d'E/S.

24.2 Programmation

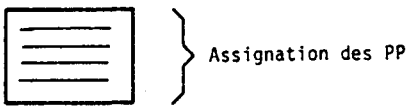
Cette description prosaique accompagnée des dessins suffira aux utilisateurs familiers des SAIA°PLC pour la programmation directe resp. pour établir le diagramme de flux.

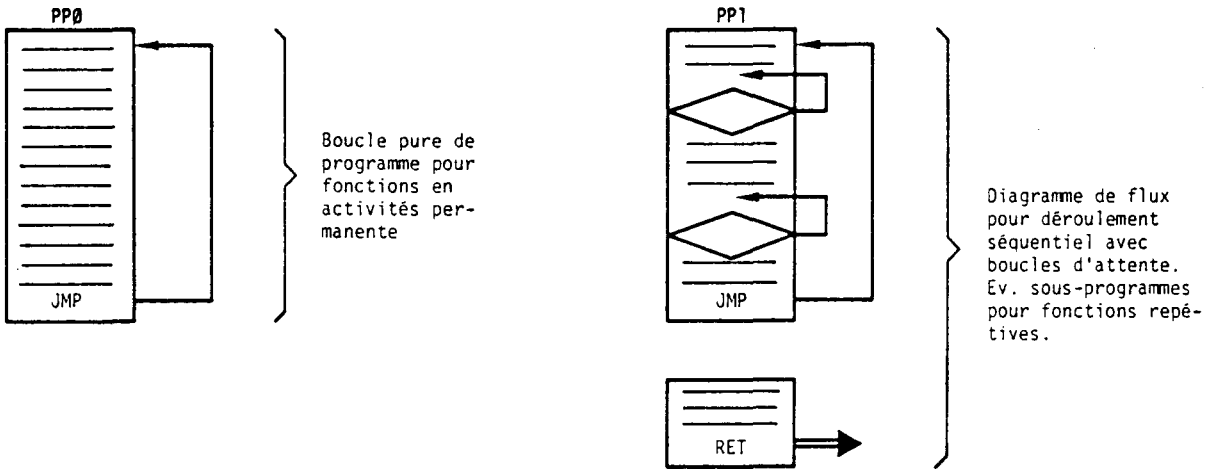
Le procédé suivant est conseillé aux débutants:

24.2.1 Structure du programme

Il s'agit d'un déroulement séquentiel, qui s'apprête parfaitement à la programmation selon diagramme de flux. Pour les fonctions en activité permanente comme Start/Stop, affichages etc., une boucle de programme cyclique contiendra ces diverses fonctions.

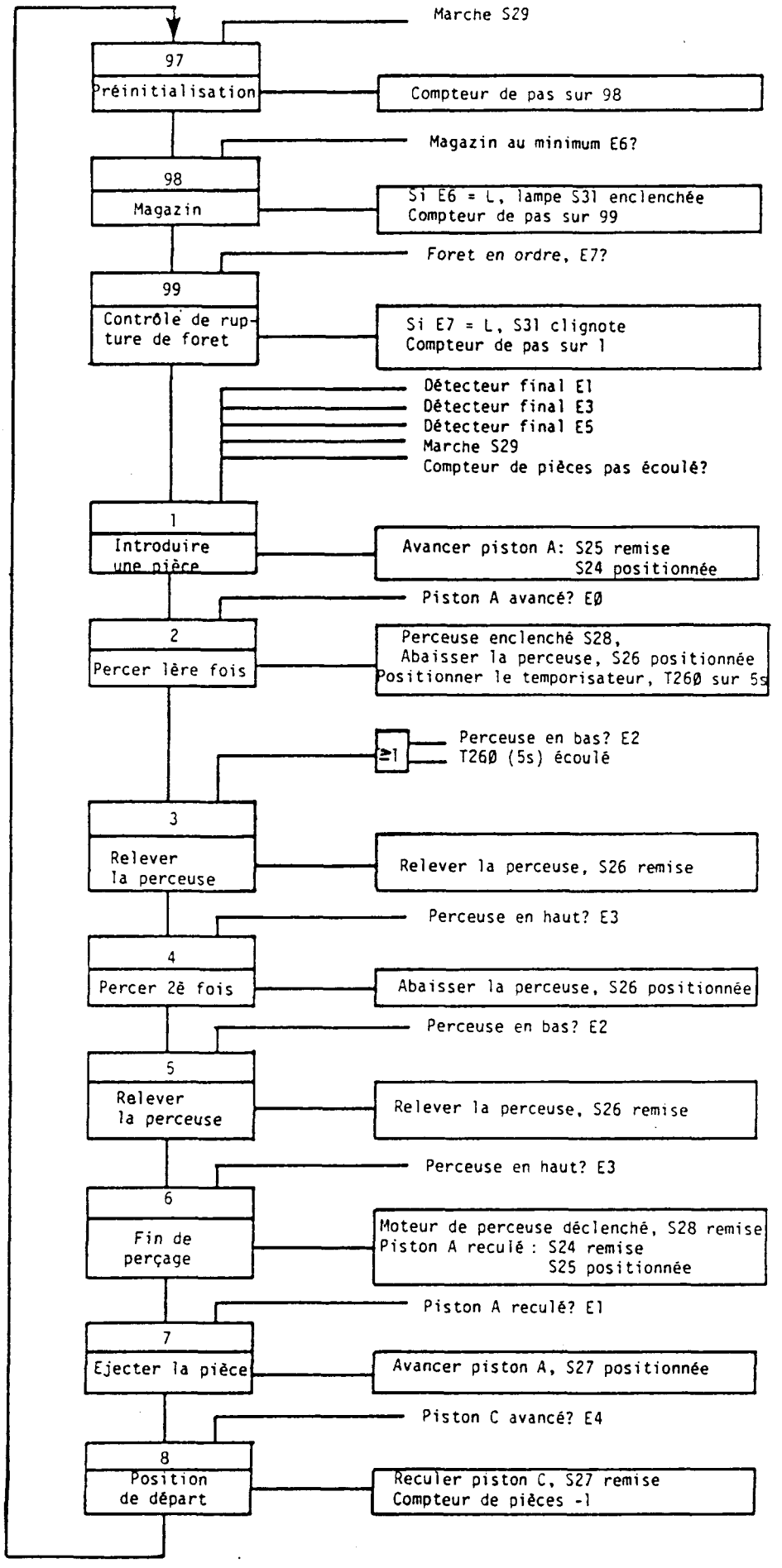
La structure suivante en résulte:





24.2.2 Plan de déroulement selon DIN

Si l'on est familier avec cette façon de présentation, la description prosaique et le diagramme représentant les divers pas peuvent d'abord être représenté de cette façon. On notera que quelques actions préliminaires, necéssaire pour le contrôle de différentes fonctions, doivent précéder l'action de déroulement 1. Ces actions préliminaires vont contenir les fonctions á exécuter avant chaque parcours de cycle. Les fonctions en activité permanente sont à programmer dans le programme cyclique PPØ.


Marche S29
97
Préinitialisation
Compteur de pas sur 98
Magazin au minimum E6?
98
Magazin
Si E6 = L, lampe S31 enclenchée
Compteur de pas sur 99
Foret en ordre, E7?
99
Contrôle de rupture de foret
Si E7 = L, S31 clignote
Compteur de pas sur 1
Détecteur final E1
Détecteur final E3
Détecteur final E5
Marche S29
Compteur de pièces pas écoulé?
1
Introduire une pièce
Avancer piston A: S25 remise
S24 positionnée
Piston A avancé? EØ
2
Percer 1ère fois
Perceuse enclenché S28,
Abaisser la perceuse, S26 positionnée
Positionner le temporisateur, T26Ø sur 5s
≥1
Perceuse en bas? E2
T26Ø (5s) écoulé
3
Relever la perceuse
Relever la perceuse, S26 remise
Perceuse en haut? E3
4
Percer 2è fois
Abaisser la perceuse, S26 positionnée
Perceuse en bas? E2
5
Relever la perceuse
Relever la perceuse, S26 remise
Perceuse en haut? E3
6
Fin de perçage
Moteur de perceuse déclenché, S28 remise
Piston A reculé : S24 remise
S25 positionnée
Piston A reculé? E1
7
Ejecter la pièce
Avancer piston A, S27 positionnée
Piston C avancé? E4
8
Position de départ
Reculer piston C, S27 remise
Compteur de pièces -1

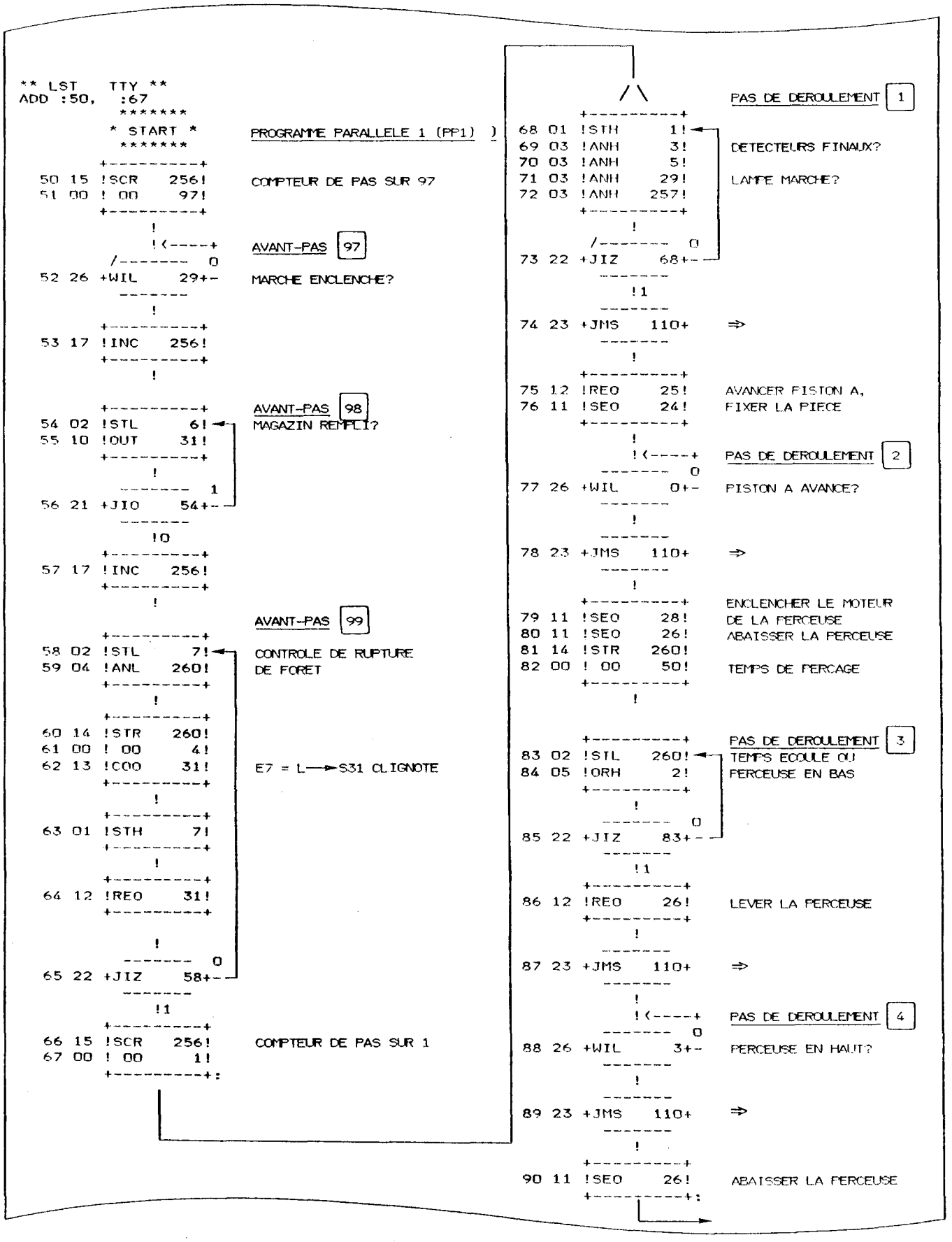
24.2.3 Programmation

Il est maintenant facile à établir le programme selon le plan de déroulement DIN. Les fonctions du mode pas à pas et du compteur de pas de programme sont à programmer dans un sous-programme parce qu'ils se répètent à chaque pas de programme. A noter que tout d'abord les fonctions dangereuses sont á remettre, avant d'attendre la touche "libération de pas", c.-à-d. d'abord tous les REO, puis les JMS, puis les SEO.

Nous trouverons dans la boucle de programme cyclique, comme mentionné, les fonctions en activité permanente. La dernière partie contenant le reset du programme est spécialement interessante. Pour réinitialiser le programme de déroulement toutes les sorties importantes sont remises par indexation puis le PP1 est réassigné à son adresse d'initialisation.

```
LISTING AVEC COMMENTAIRES A L'AIDE DU PCA-ASSEMBLER

********************  ASSIGNATION DES PP
ADDR NC   MNC   OPRD
   0 00   NOP      0
   1 29   PAS      1
   2 00    00     50
   3 20   JMP     10->
 ++++++++++++++++++++ PROGRAMME PARALLELE 0 (PPO)
->10 01   STH      8 } CYCLE START/STOP
  11 03   ANH      9 }
  12 11   SEO     29  LAMPE "MARCHE" ALLUMEE
  13 02   STL      9
  14 12   REO     29  LAMPE "MARCHE" ETEINTE
 -------------------- POSITIONNER LE COMPTEUR DE PIECES
  17 01   STH     11
  18 09   DYN    300
  19 15   SCR    257
  20 18    18     23
  21 02   STL    257
  22 10   OUT     30  LAMPE "NOMBRE D'UNITES ATTEINT"
 -------------------- AFFICHAGE PRESELECTION
  25 01   STH     12
  26 31   DTC    257  AFFICHAGE COMPTEUR DE PIECES
  27 08   NEG      0
  28 31   DTC    256  AFFICHAGE COMPTEUR DE PAS
 -------------------- PROGRAMME RESET (RE-ASSIGNATION)
  31 01   STH     10 }
  32 04   ANL      9 } REMISE?
  33 09   DYN    301 }
<-34 22   JIZ     10->
  35 16   SEI      0
->36 12   REO   1024  REMISE DES SORTIES
  37 27   INI      7
  38 21   JIO     36->
  39 29   PAS      1 } RE-ASSIGNATION DE PP1 ET DONC
  40 00    00     50 } RESET DU PROGRAMME
  41 20   JMP     10->
```

Représentation en diagramme de flux du PP1 au moyen du PCA-Assembler



```
                  <-----+  PAS DE DEROULEMENT [5]
               /-------\ 0
 91  26  +WIL       2+-   PERCEUSE EN BAS?
          -------/
              !
          +----------+
 92  12   !REO     26!    LEVER LA PERCEUSE
          +----------+
              !
           /-------\
 93  23   +JMS    110+     =>
           \-------/
              !
              !<-----+  PAS DE DEROULEMENT [6]
           /-------\ 0
 94  26   +WIL       3+-   PERCEUSE EN HAUT?
           \-------/
              !
          +----------+
 95  12   !REO     28!    DECLENCHER LE MOTEUR
          +----------+    DE LA PERCEUSE
              !
           /-------\
 96  23   +JMS    110+     =>
           \-------/
              !
          +----------+
 97  12   !REO     24!    RECULER PISTON A
 98  11   !SEO     25!
          +----------+
              !
              !<-----+  PAS DE DEROULEMENT [7]
           /-------\ 0
 99  26   +WIL       1+-   PISTON A RECULE?
           \-------/
              !
           /-------\
100  23   +JMS    110+     =>
           \-------/
              !
          +----------+
101  11   !SEO     27!    EJECTER LA PIECE
          +----------+
              !
              !<-----+  PAS DE DEROULEMENT [8]
           /-------\ 0
102  26   +WIL       4+-   PISTON C AVANCE?
           \-------/
              !
          +----------+
103  12   !REO     27!    RECULER PISTON C
          +----------+
              !
           /-------\
104  23   +JMS    110+     =>
           \-------/
              !
          +----------+
105  18   !DEC    257!    COMPTEUR DE PIECES -1
          +----------+
              !
           /-------
106  20   +JMP     50+ --+
           \-------/      |
                          |
```

```
ADD :110,   :114
            *******
=>          * START *   SOUS-PROGRAMME POUR PAS
            *******             A PAS
              /
          +----------+
110  01   !STH     13!    DETECTEUR PAS A PAS
          +----------+           ENCLENCHE?
              !
           /-------\ 0
111  22   +JIZ    113+-- SI NON, SAUT
           \-------/   |
              !1       |
              !<-----+ |
           /-------\ 0 |
112  26   +WIL     14+- | SI OUI, ATTENDRE LA
           \-------/   | LIBERATION DE PAS
              !        |
          +----------+ |
113  17   !INC    256!<-- COMPTEUR DE PAS +1
          +----------+
              !
           /-------\
114  24   +RET     0+
           \-------/
```

REFERENCES DE QUELQUES ELEMENTS ET DE
L'ADRESSE DE SAUT AU SOUS-PROGRAMME 110

COMPTEUR DE PAS

```
ADD :0,   :114,  OPERAND :256

  28  31   DTC   256
  50  15   SCR   256
  53  17   INC   256
  57  17   INC   256
  66  15   SCR   256
 113  17   INC   256
```

COMPTEUR DE PIECES

```
ADD :0,   :114,  OPERAND :257

  19  15   SCR   257
  21  02   STL   257
  26  31   DTC   257
  72  03   ANH   257
 105  18   DEC   257
```

SOUS-PROGRAMME 110

```
ADD :0,   :114,  OPERAND :110

  74  23   JMS   110
  78  23   JMS   110
  87  23   JMS   110
  89  23   JMS   110
  93  23   JMS   110
  96  23   JMS   110
 100  23   JMS   110
 104  23   JMS   110
```

LAMPE MARCHE

```
ADD :0,   :114,  OPERAND :29

  12  11   SEO    29
  14  12   REO    29
  52  26   WIL    29
  71  03   ANH    29
```

DETECTEUR FINAL HAUT DE LA PERCEUSE

```
ADD :0,   :114,  OPERAND :3

  69  03   ANH     3
  88  26   WIL     3
  94  26   WIL     3
```

Schéma pour la résolution d'un problème de contrôle par utilisation d'un PLC

Etablir le cahier de charges pour le processus entier (ev. avec esquisses, description de déroulement, division en blocs de processus etc.).

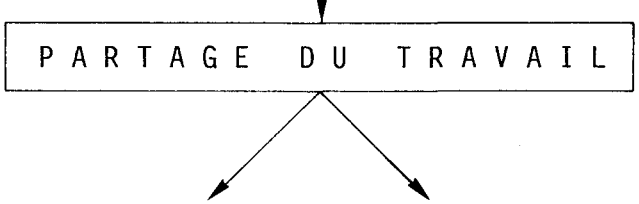
Schéma général de l'installation contenant les solutions mécaniques/par air comprimé/hydraulique ou electriques.

Dimensionnement du PLC

- Définition du nombre d'E+S (en respectant les possibilités de MUX)
- Définition des signaux d'E/S (=/≈, tension, courant)
- Définition du type de PLC (PCA1 ≤112 E+S, PCA2 >96 E+S), ev. division du processus en plusieurs PLC, hiérarchie.
- Définition des modules (modularité 8/16/32) en respectant une reserve de 1Ø - 2Ø% E+S.
- Définition de l'extension de mémoire selon les bases suivantes
  - Contrôle simple — Capacité de mémoire = approx. 5 x nombre d'E+S
  - Complexité moyenne — Capacité de mémoire = approx. 1Ø x nombre d'E+S
  - Grande complexité — Capacité de mémoire = approx. 2Ø x nombre d'E+S
- Contrôler si les fonctions standard du SAIA°PLC suffisent pour la programmation. Prévoir les types PCA14 ou PCA23, si des fonctions arithmétiques universelles ou des possibilités de protocoles volumineux sont désirées.
- Définition du type définitif de mémoire (standard = EPROM).

Commande du PLC

Définition exacte de l'équipement du PLC, y compris les accessoires comme câble, interface externe, module d'affichage, accessoires de programmation et de simulation.



PARTAGE DU TRAVAIL

Hardware

- Terminer les détails de construction
- Fabrication des pièces mécaniques

Après livraison du PLC commandé:
- Montage du PLC dans l'installation
- Câblage des E/S
- Contrôle du câblage des E/S par affichage LED et en mode de fonctionnement "MAN"

Programmation

- Designer les E/S
- Définition de la structure de programme et du mode de programmation des parties de programme correspondantes (modules de programme!)
- Programmation sur papier (en respectant la détection d'erreur par DOP, DTC et Watchdog)

Si une unité de programmation est déjà disponible ou après livraison du PLC commandé:
- Introduction du programme (Introduire des NOP comme réserve et modifications)
- Tester les parties de programme à l'aide du matériel de simulation
- Correction des erreurs
- Etablir la documentation provisoire
- Recopier le programme sur EPROM pour des raisons de securité

Mise en service

- Si le test du programme simulé ainsi que le contrôle des E/S en mode "MAN" a été couronné de succès ---> Enclencher (ev. débrancher les combinaisons dangereuses, ne pas actionner les touches d'initialisation).
- Si une possibilité pas à pas a été prévue (à ne confondre avec le mode de fonctionnement STEP), d'abord contrôler le processus en pas à pas dans cette position.
- Tester le mode automatique (ev. aussi les états dangereux de fonctionnement, comme rupture de fil, actionnement simultané de plusieurs touches ou chute de tension resp. processeur défectueux).
- Ev. correction du programme sur RAM.
- Si en ordre, copier le programme en 2 jeux sur EPROM
  1 jeu de service, 1 jeu de programme de sécurité.
- Actualiser la documentation du programme et commenter le programme de façon intelligible pour d'autres personnes.
- Etablir un manuel "recherche d'erreurs" pour le personnel d'entretien.

F I N

Vue d'ensemble des instructions niveau logiciel 1H

Instruction	Chapitre	Page
ANH	E1	3E
ANL	E1	3E
COO	E2	11E
DEC	E3	16E
DEI	E6	30E, 34E
DOP	E8	39E
DTC	E8	40E
DYN	E1	7E
INC	E3	16E
INI	E6	30E, 34E
JIO	E4	24E, 26E
JIZ	E4	24E, 26E
JMS	E4	25E, 26E
JMP	E4	23E, 26E
NEG	E1	6E
NOP	E5	29E
ORH	E1	4E
ORL	E1	4E
OUT	E2	9E
PAS Ø...15	E7	35E
PAS 18	E7	36E
PAS 3Ø	E7	37E
PAS 31...38	E7	37E
REO	E2	10E
RET	E4	25E
SCR	E3	14E, 19E
SEA	E5	29E
SEI	E6	30E, 34E
SEO	E2	10E
STH	E1	2E
STL	E1	2E
STR	E3	13E, 19E
WIH	E4	27E
WIL	E4	27E
XOR	E1	5E

SAIA SA

Electronique Industrielle et Composants
3280 Morat/Suisse

Centrale	Téléphone	037 727111
	Téléfax	037 71 44 43
	Télex	942127
Vente Suisse	Téléphone	037 727727
	Téléfax	037 71 19 83

Nos représentations nationales

Belgique
Landis & Gyr Belge SA, Dépt. Industrie
Avenue des Anciens Combattants 190, B-1140 Bruxelles
☎ 02 244 02 11, Tx 65930, Fax 02 242 88 31

Danmark
Skandia-Havemann
Vallensbækvej 46, DK-2625 Vallensbæk
☎ 02 643333, Tx 33383, Fax 02 642245

Deutschland
SAIA GmbH
Flinschstrasse 67, D-6000 Frankfurt 60
☎ 069 420993-0, Ttx 6999375, Fax 069 425654

España
Landis & Gyr BC SA
Batalla del Salado 25, Apartado 575, 28045 Madrid
☎ 91 467 19 00, Tx 22976, Fax 91 239 44 79

France
SAIA Sàrl
10, Blvd. Louise Michel, F-92230 Gennevilliers
☎ 1 40860345, Tx 613189, Fax 1 47914013

Great Britain
A.S.A.P. Ltd.
Unit 15D, Compton Place, Surrey Avenue, Camberley, Surrey GU15 3DX
☎ 0276 691580, Fax 0276 691581

Italia
SAIA S.r.l.
Via Cadamosto 3
20094 Corsico MI

Nederland
Landis & Gyr BV, Div. Electrowater
Kampenringweg 45, Postbus 444, NL-2800 AK-Gouda
☎ 01820 65683, Tx 20657, Fax 01820 32437

Norge
Malthe Winje & Co A/S
Cort Adelersgt. 14, Postboks 2440, Solli, N-0202 Oslo 2
☎ 02 558640, Tx 19629, Fax 02 552211

Österreich
COMECON
Landis & Gyr Gesellschaft m.b.H
Breitenfurterstrasse 148, Postfach 9, A-1230 Wien
☎ 0222 842626-0, Tx 132706, Fax 0222 842626313

Portugal
Infocontrol Electronica e Automatismo LDA.
Av. da Igreja No. 68–1° Esq., P-1700 Lisboa
☎ 01 775161-65, Tx 63454, Fax 01 775687

Suomi
Finland
OY Landis & Gyr AB
SF-02430 Masala
☎ 8 029731, Tx 1001153, Fax 8 029755 31

Sverige
Beving Elektronik AB
St. Eriksgatan 113a, Box 21104, S-10031 Stockholm
☎ 08 151780, Tx 10040, Fax 08 336863

USA
After sales services: Maxmar Controls Inc.
99 Castleton Street, Pleasantville, New York 10570-3403
☎ 914 747 3540, Fax 914 747 3567

Australia
Landis & Gyr (Australia) Pty Ltd
411 Ferntree Gully Road, P.O. Box 202, Mount Waverley, Vic. 3149
☎ 3 544-2322, Tx 32244, Fax 3 543 74 96

Argentina
Electromedidor S.A.I. y C.
Defensa 320, RA-1065 Buenos Aires
☎ 1 337125, Tx 23377, Fax 1 331 9582

Imprimé en Suisse 26/721 F1 11.89 G02