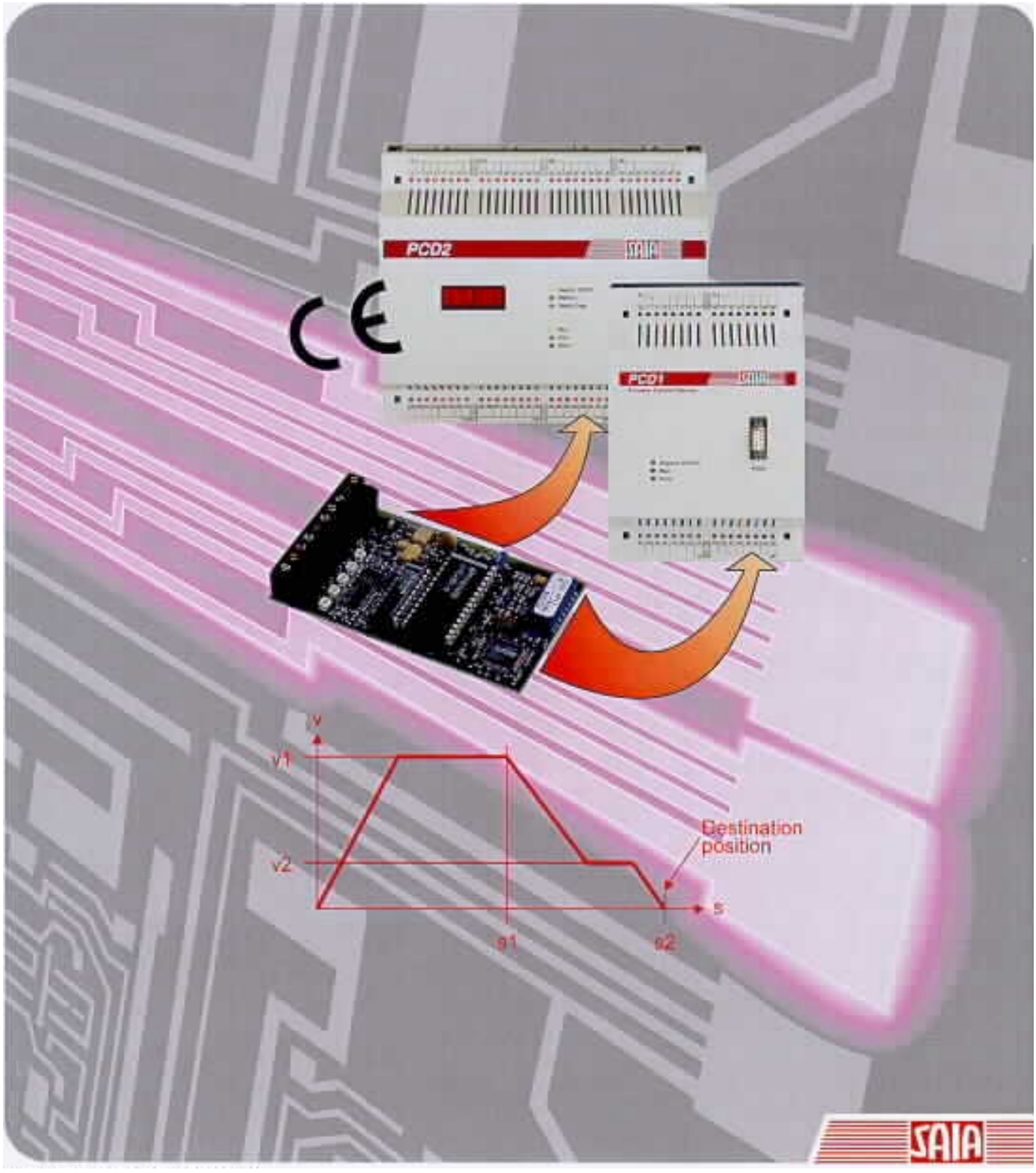
SAIA-Burgess Electronics

SWITCHES • MOTORS • CONTROLLERS

SAIA® PCD

Process Control Devices

PCD2.H31x
Module de positionnement pour servo-entraînements



Edition française 26/762 F2

SAIA-Burgess Electronics SA
Rue de la Gare 18
CH-3280 Morat (Suisse)
http;//www.saia-burgess.com

DIV: Electronic Controllers Téléphone 026 / 672 72 72
Télécopieur 026 / 672 74 99

Sociétés SAIA-Burgess

Suisse
SAIA-Burgess Electronics SA
Rue de Fribourg 33
CH-3280 Morat
☎ 026 672 77 77, Fax 026 670 19 83

France
SAIA-Burgess Electronics Sàrl.
10, Bld. Louise Michel
F-92230 Gennevilliers
☎ 01 46 88 07 70, Fax 01 46 88 07 99

Allemagne
SAIA-Burgess Electronics GmbH
Daimlerstrasse 1k
D-63303 Dreieich
☎ 06103 89 060, Fax 06103 89 06 66

Pays-Bas
SAIA-Burgess Electronics B.V.
Hanzeweg 12c
NL-2803 MC Gouda
☎ 0182 54 31 54, Fax 0182 54 31 51

Autriche
SAIA-Burgess Electronics Ges.m.b.H.
Schallmooser Hauptstrasse 38
A-5020 Salzburg
☎ 0662 88 49 10, Fax 0662 88 49 10 11

Belgique
SAIA-Burgess Electronics Belgium
Avenue Roi Albert 1er, 50
B-1780 Wemmel
☎ 02 456 06 20, Fax 02 460 50 44

Italie
SAIA-Burgess Electronics S.r.l.
Via Cadamosto 3
I-20094 Corsico MI
☎ 02 48 69 21, Fax 02 48 60 06 92

Hongrie
SAIA-Burgess Electronics Automation Kft.
Liget utca 1.
H-2040 Budaörs
☎ 23 501 170, Fax 23 501 180

Représentations

Grande-Bretagne
Canham Controls Ltd.
25 Fenlake Business Centre, Fengate
Peterborough PE1 5BQ UK
☎ 01733 89 44 89, Fax 01733 89 44 88

Portugal
INFOCONTROL Electronica e Automatismo LDA.
Praceta Cesário Verde, No 10 s/cv, Massamá
P-2745 Queluz
☎ 21 430 08 24, Fax 21 430 08 04

Danemark
Malthe Winje Automation AS
Håndværkerbyen 57 B
DK-2670 Greve
☎ 70 20 52 01, Fax 70 20 52 02

Espagne
Tecnosistemas Medioambientales, S.L.
Poligono Industrial El Cabril, 9
E-28864 Ajalvir, Madrid
☎ 91 884 47 93, Fax 91 884 40 72

Norvège
Malthe Winje Automasjon AS
Haukelivn 48
N-1415 Oppegård
☎ 66 99 61 00, Fax 66 99 61 01

Tchéquie
ICS Industrie Control Service, s.r.o.
Modranská 43
CZ-14700 Praha 4
☎ 2 44 06 22 79, Fax 2 44 46 08 57

Suède
Malthe Winje Automation AB
Truckvägen 14A
S-194 52 Upplands Våsby
☎ 08 795 59 10, Fax 08 795 59 20

Pologne
SABUR Ltd.
ul. Druzynowa 3A
PL-02-590 Warszawa
☎ 22 844 63 70, Fax 22 844 75 20

Suomi/ Finlande
ENERGEL OY
Atomitie 1
FIN-00370 Helsinki
☎ 09 586 2066, Fax 09 586 2046

Australie
Siemens Building Technologies Pty. Ltd.
Landis & Staefa Division
411 Ferntree Gully Road
AUS-Mount Waverley, 3149 Victoria
☎ 3 9544 2322, Fax 3 9543 8106

Argentine
MURTEN S.r.l.
Av. del Libertador 184, 4° “A”
RA-1001 Buenos Aires
☎ 054 11 4312 0172, Fax 054 11 4312 0172

Service après-vente

USA
SAIA-Burgess Electronics Inc.
1335 Barclay Boulevard
Buffalo Grove, IL 60089, USA
☎ 847 215 96 00, Fax 847 215 96 06

Issue : 22.11.2000

SAIA® Process Control Devices

Module de positionnement pour servo-entraînements

PCD2.H31x

Etition 26/762 F2 – 12.99

Sous réserve de modifications

Mise à jour

Manuel : Module de positionnement pour servo-entraînements PCD2.H31x - édition F2

Date	Chapitre	Page	Description
12.05.2000	Annexe A	A-36	ResStatRg : Diverses corrections

Table des matières

		Page
1.	**Présentation**	
1.1	Généralités	1-1
1.2	Fonctionnement et application	1-2
1.3	Points forts	1-3
1.4	Domaines d'utilisation	1-4
1.5	Programmation	1-5
1.6	Modes d'exploitation	1-6
1.7	Mise en service	1-7
2.	**Caractéristiques techniques**	
2.1	Caractéristiques matérielles	2-1
2.2	Caractéristiques électriques	2-4
2.3	Caractéristiques de fonctionnement	2-5
3.	**Constitution**	
4.	**Repérage des bornes et signification des voyants de signalisation d'état**	
5.	**Description fonctionnelle**	
5.1	Modes d'exploitation	5-1
5.2	Générateur de profil de vitesse	5-2
5.3	Régulateur PID	5-4
5.4	Capteur de position et circuit d'entrées	5-5
5.5	Convertisseur numérique-analogique	5-8
5.6	Complément d'information sur la fonction Retour à l'origine (bloc de fonctions « HOME »)	5-9

Page

6. Prise de contact

6.1 Bien démarrer avec la programmation en liste d'instructions (IL) 6-2

6.1.1 Exemple simple de programme édité en IL avec boucles d'attente 6-3
6.1.2 Exemple simple de programme édité en GRAFTEC 6-6
6.1.3 Programme simple de mise en service 6-11

6.2 Bien démarrer avec la programmation en FUPLA 6-13

7. Programmation

7.1 Programmer en liste d'instructions IL avec les blocs de fonctions « FB » 7-2

7.1.1 Constitution de la fourniture et installation des des blocs de fonctions 7-2
7.1.2 Description des blocs de fonctions 7-5

7.2 Programmer en FUPLA avec les boîtes de fonctions « FBoxes » 7-9
7.3 Programmer en GRAFTEC avec les boîtes de fonctions 7-10

8. Typologie des erreurs et diagnostic

8.1 Erreurs de définition 8-1
8.2 Erreurs de traitement 8-2

8.2.1 Paramétrage incorrect 8-2
8.2.2 Prise d'origine incorrecte 8-3

Page

9. Installation et mise en service

9.1 Introduction 9-1
9.2 Contrôle de l'installation et du câblage 9-2
9.3 Mise en service de l'entraînement sans module de positionnement 9-3
9.4 Mise en service de l'entraînement avec module de positionnement 9-4

9.4.1 Mise sous tension des alimentations 9-4
9.4.2 Édition d'un programme utilisateur simple 9-5
9.4.3 Définition des caractéristiques techniques de la machine 9-6
9.4.4 Contrôle du sens de rotation et du calcul de déplacement (codeur) 9-8
9.4.5 Vérification et réglage du régulateur PID 9-10
9.4.6 Effet des réglages PID sur le comportement du régulateur 9-14
9.4.7 Programme de mise en service 9-15

10. Outil de mise en service (PCD8.H31)

11. Sécurité

12. Exemples d'application

12.1 1er exemple : positionnement d'1 axe avec retour à l'origine 12-1
12.2 2ème exemple : positionnement d'1 axe, à 2 vitesses 12-9

Page

Annexe A Récapitulatif des blocs de fonctions et des instructions PCD2.H31x en langage IL

INIT	Initialisation du module PCD2.H31x	A-1
HOME	Initialisation de la position d'origine	A-3
EXEC	Exécution d'une instruction du PCD2.H31x	A-5
StartMot	Démarrage du déplacement	A-7
StopUrg	Arrêt d'urgence du déplacement	A-8
Stop	Arrêt du déplacement	A-9
MotOff	Désactivation de la régulation PID	A-10
RdActPos	Lecture de la position réelle	A-11
RdActVel	Lecture de la vitesse réelle	A-12
RdIntSum	Lecture de la somme d'intégration	A-13
RdIndexRg	Lecture du registre d'index	A-14
RdStatRg	Lecture du registre d'état	A-15
RdTargPos	Lecture de la position cible	A-16
RdTargVel	Lecture de la vitesse cible	A-17
GoForw	Avance	A-18
GoBackw	Recul	A-19
SgStpFor	Avance d'un pas	A-20
SgStpBak	Recul d'un pas	A-21
LdDestAbs	Chargement de la position cible absolue	A-22
LdDestRel	Chargement de la position cible relative	A-23
LdVelAbs	Chargement de la vitesse absolue	A-24
LdVelRel	Chargement de la vitesse relative	A-25
LdAccAbs	Chargement de l'accélération absolue	A-26
LdAccRel	Chargement de l'accélération relative	A-27
LdPropG	Chargement du gain proportionnel	A-28
LdIntG	Chargement du gain intégral	A-29
LdDerG	Chargement du gain dérivé	A-30
LdSampInt	Chargement du temps d'échantillonnage	A-31
LdIntLim	Chargement de la limite d'intégration	A-32
ActRegFact	Activation de la régulation PID	A-33
LdBrkPtAbs	Chargement d'un point d'arrêt absolu	A-34
LdBrkPtRel	Chargement d'un point d'arrêt relatif	A-35
ResStatRg	Remise à 0 du registre d'état	A-36
SetIdxPos	Définition de la position d'index	A-37
SetZero	Définition de la position 0	A-38
MotConf	Configuration du déplacement	A-39
SetPosTol	Définition de la tolérance de position	A-40

Annexe B Récapitulatif des boîtes de fonctions et des instructions PCD2.H31x en langage FUPLA

en préparation

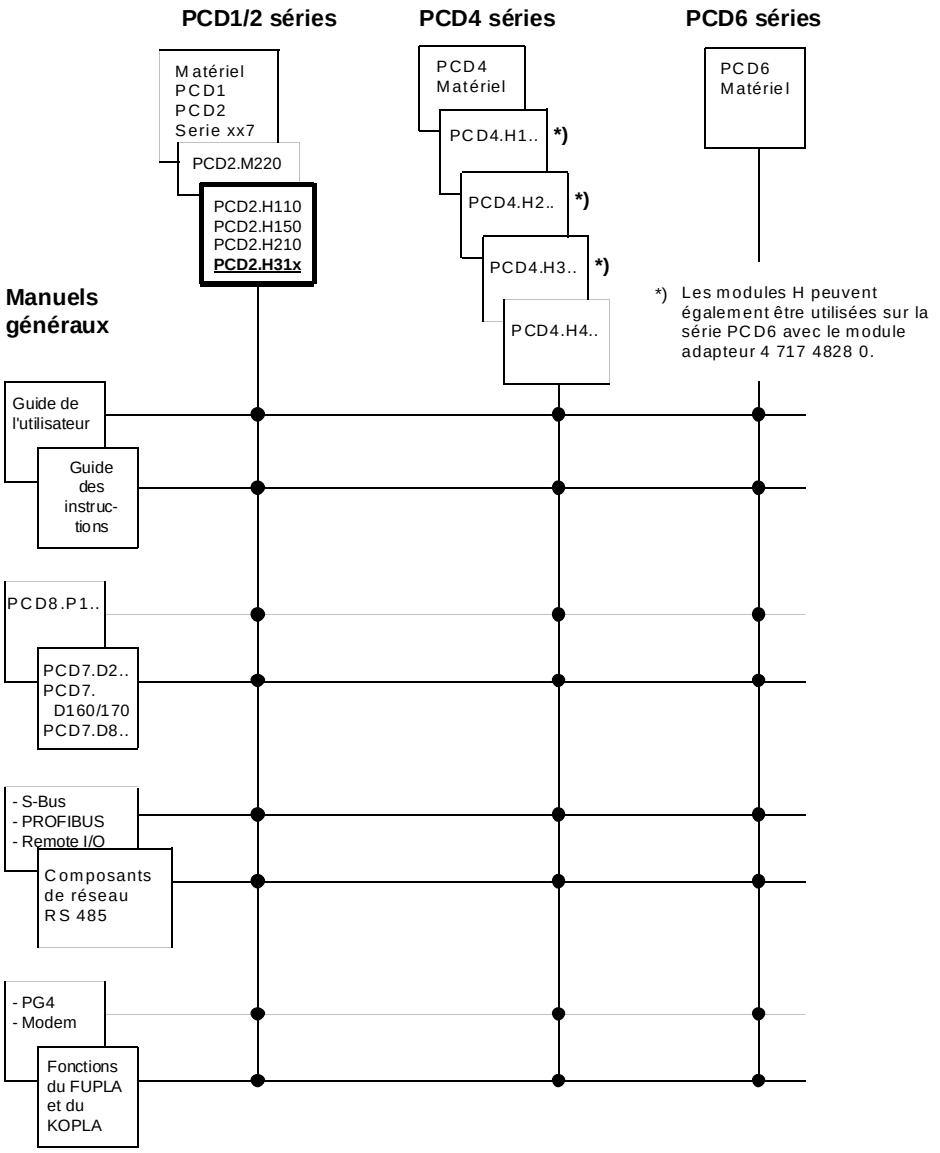
Avis aux lecteurs :

De nombreux manuels techniques précis et détaillés ont été élaborés par SAIA-Burgess Electronics SA afin de faciliter l'installation et l'exploitation de ses automates programmables ; ils s'adressent à un personnel qualifié ayant suivi au préalable nos stages de formation.

Pour optimiser les performances des appareils de commande de processus SAIA® PCD, nous vous conseillons de respecter scrupuleusement les consignes de montage, de câblage, de programmation et de mise en service figurant dans ces manuels. Cette démarche rigoureuse vous donnera l'assurance d'une satisfaction totale.

Toutefois, si vous souhaitez formuler des propositions ou des commentaires visant à améliorer la qualité et le contenu de nos documentations, nous vous serions reconnaissants de compléter le formulaire situé en dernière page de cette notice.

Vue d'ensemble de la gamme et de la documentation PCD



Fiabilité et sécurité des automates programmables

Soucieux d'offrir à sa clientèle des automates programmables fiables et sûrs, SAIA-Burgess Electronics SA apporte le plus grand soin à la conception, au développement et à la fabrication de ses produits.

Parmi ces mesures, citons :

- Technologie de pointe,
- Conformité aux normes,
- Certification ISO 9001,
- Agrément de nombreux organismes internationaux (Germanischer Lloyd, UL, Det Norske Veritas, marquage CE...),
- Choix de composants de haute qualité,
- Contrôles qualité aux différents stades de fabrication,
- Essais en conditions réelles de fonctionnement,
- Déverminage à 85 °C pendant 48 heures.

Malgré l'excellence et le grand soin apporté à sa production, SAIA-Burgess Electronics SA ne saurait être tenu responsable des défaillances naturelles d'un composant. A cet égard, les « Conditions générales de vente » exposent clairement les limites de garantie offertes par SAIA-Burgess Electronics SA.

Le responsable de production doit également s'assurer de la fiabilité de son installation ; il lui incombe en effet de se conformer aux spécifications techniques de l'automate sans jamais le soumettre à des conditions extrêmes d'utilisation (respect de la plage de températures, protection contre les surtensions, immunité aux parasites et tenue aux chocs).

Il lui faut en outre veiller à l'application de toutes les règles de sécurité en vigueur afin de garantir qu'aucun produit défectueux ne risque de porter atteinte à la sécurité des biens et des personnes. Tout défaut générateur de danger doit donner lieu à des mesures complémentaires visant à l'identifier et à en prévenir les conséquences. Ainsi les sorties directement liées à la sécurité de fonctionnement du matériel doivent être raccordées aux entrées et surveillées par logiciel. Il convient enfin de faire systématiquement appel aux fonctions de diagnostic du PCD (chien de garde, blocs d'organisation des exceptions « XOB », instructions de test ou de recherche d'erreurs).

Exploitée dans les règles de l'art, la gamme SAIA® PCD intègre des constituants d'automatismes modernes, alliant sécurité et haute fiabilité, et capables d'assurer pendant des années les fonctions de contrôle-commande, de régulation et de surveillance de votre équipement.

1. Présentation

1.1 Généralités

Le PCD2.H31x est un module d'E/S intelligent de la gamme PCD2. Il sert au positionnement d'un axe indépendant muni d'un entraînement à vitesse de rotation variable (servomoteur). Ce dernier peut être un moteur à courant continu ou à courant alternatif réglable, capable de déterminer le nombre de tours et la position via un étage de puissance et un codeur incrémental.

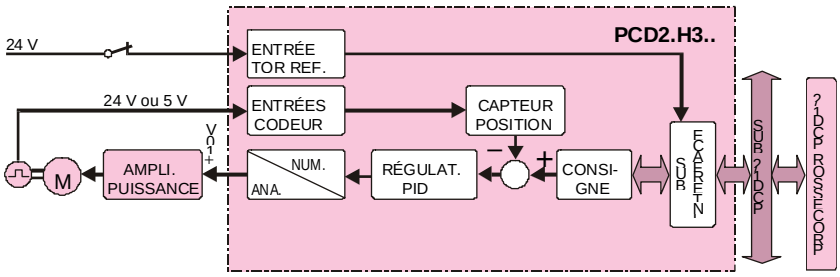
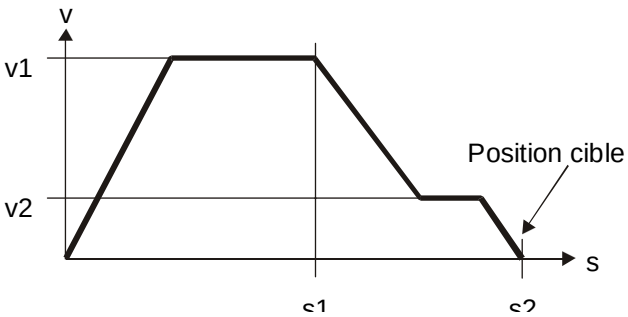
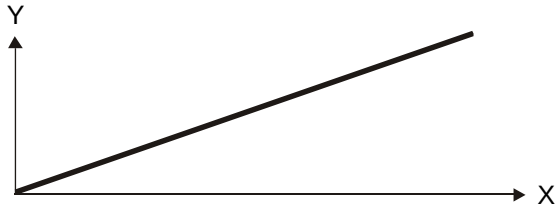


Schéma synoptique d'un servo-entraînement pour 1 axe



Mode régulation de position avec accélération et décélération constantes, palier de vitesse et approche à vitesse lente de la cible



Quasi-synchronisation de 2 axes

1.2 Fonctionnement et application

Le module PCD2.H31x permet de positionner un axe indépendant avec régulation de vitesse de servomoteurs à courant continu ou à courant alternatif. L'entraînement doit alors être équipé d'un étage de puissance et d'un codeur incrémental pour enregistrer la position ou la vitesse.

Le PCD2.H31x se raccorde au bus E/S de chaque automate de base PCD1 ou PCD2. Le PCD1 peut ainsi comporter jusqu'à 4 modules de positionnement (soit 4 axes) et le PCD2 jusqu'à 8 modules de positionnement (soit 8 axes).

Chaque axe fonctionne de manière autonome, grâce à son microprocesseur intégré qui, en régulation PID, maîtrise chaque mouvement indépendant dicté par les paramètres du programme utilisateur (vitesse, accélération et position cible). On peut relier par programme plusieurs axes point à point, en quasi-synchronisation, pour effectuer ainsi un déplacement linéaire sur des mécaniques cartésiennes.

De par leur consommation de courant relativement élevée, les modules PCD2.H31x doivent impérativement être enfichés dans l'automate de base PCD2.M1xx et **non** dans l'extension PCD2.C1xx.

1.3 Points forts

- Module idéal pour machines compactes et économiques
- Régulation PID de la position et de la vitesse de rotation du servo-entraînement
- Possibilité de modification de la vitesse, de la position cible et des fonctions PID en cours de mouvement
- Sortie analogique ±10 V, résolution 12 bits (dont bit de signe) pour la commande de l'étage de puissance du moteur
- Entrée TOR destinée au contact de référence (sur PCD2.H310 seulement)
- Entrées de signaux codeur 24 VCC (logique positive) ou 5 V en RS 422 (Antivalent Line Driver)
- Les autres entrées et sorties de l'axe (fins de course, contact de référence du PCD2.H311, signal de validation « Enable » de l'entraînement…) doivent être surveillées et pilotées par un module d'E/S standard (PCD2.B100, PCD2.E110/E111 ou PCD2.A410).

1.4 Domaines d'utilisation

- Robot de manutention
- Automates de positionnement et de montage
- Automates de palettisation
- Machines d'emballage
- Machines destinées au travail de la tôle
- Perceuses automatiques
- et bien d'autres…

1.5 Programmation

Les blocs de fonctions « FB » ou boîtes de fonctions « FBox » permettent de ne saisir que les paramètres indispensables au déplacement. Chaque bloc et ses instructions sont décrits en détail dans ce manuel et complétés d'exemples pratiques. La programmation fait appel au logiciel PG4 (à partir de la version V2.0.70), en liste d'instructions IL ou en langage graphique FUPLA.

Commandes d'initialisation

INIT Ce bloc de fonctions comporte 11 paramètres.

Cf. § 7.1.2 et annexe A (pages A1 et A2)

Commandes d'exécution

EXEC Ce bloc de fonctions, composé de 3 paramètres, permet d'exécuter plus de 30 instructions différentes.

Cf. § 7.1.2 et annexe A (pages A5 à A40)

Commandes de mise en service

HOME Ce bloc de fonctions, composé de 7 paramètres, recherche automatiquement le contact de référence : il assure la fonction de Retour à l'origine.

Cf. § 7.1.2 et annexe A (pages A3 et A4)

Typologie des erreurs et diagnostic

Détection des paramètres incorrects (registre de diagnostic).
Surveillance de la fonction Retour à l'origine (Home) par temporisation.

Cf. chapitre 8

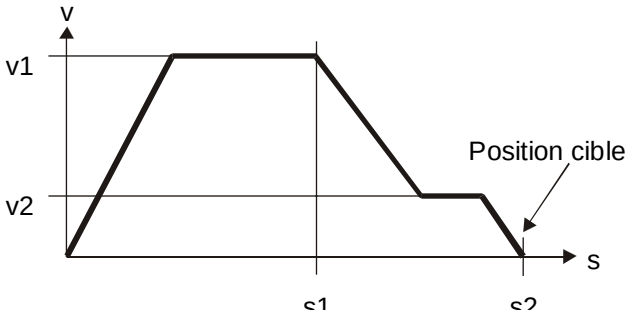
1.6 Modes d'exploitation

Le positionnement de servo-entraînement est en général régi par deux modes d'exploitation :

- régulation de position,
- régulation de vitesse de rotation.

Le mode régulation de position permet un déplacement régulé vers une position cible donnée, après réglage des paramètres (actions PID, accélération, vitesse, etc.). La vitesse, les coefficients PID et la position cible sont modifiables en cours de mouvement.

En mode régulation de vitesse de rotation, on accélère jusqu'à la vitesse de consigne, à l'accélération donnée. Le déplacement régulé se poursuit à cette vitesse jusqu'à l'intervention d'une commande d'arrêt. La vitesse de consigne est modifiable en cours de mouvement.



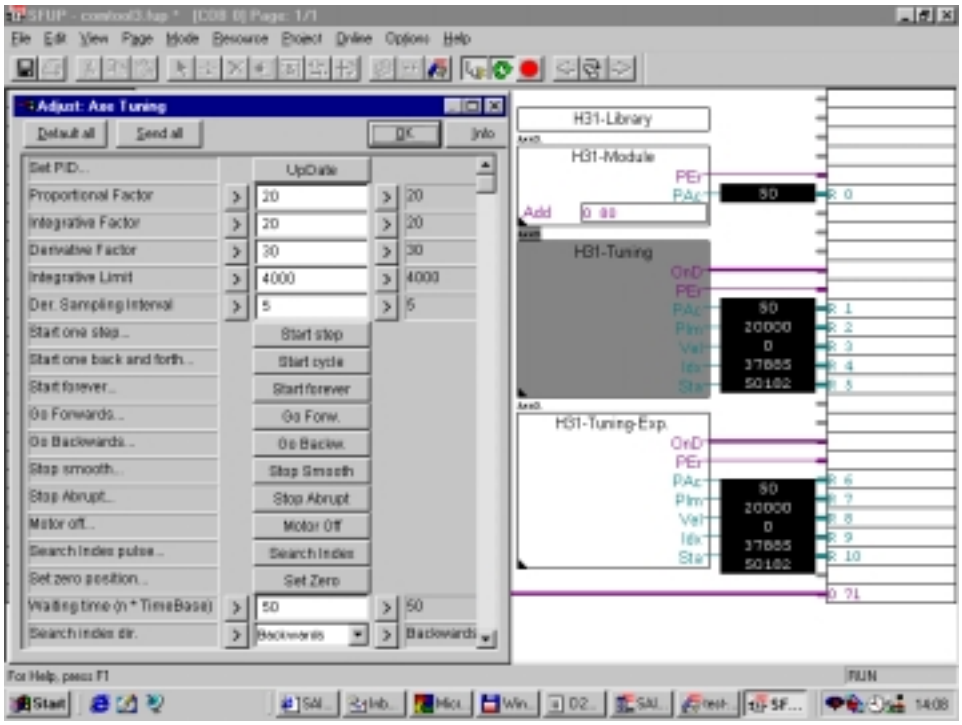
Mode régulation de position avec accélération et décélération constantes, palier de vitesse et approche à vitesse lente de la cible

1.7 Mise en service

La mise en service d'une installation fait appel à un outil de programmation simple et convivial, fonctionnant sous logiciel FUPLA. Il permet d'effectuer aisément tous les tests nécessaires à la mise en route et de vérifier les réglages en ligne.

Il est fourni sur disquette référencée PCD8.H31.

Cf. chapitre 10 pour le détail.



Notes personnelles :

2. Caractéristiques techniques

2.1 Caractéristiques matérielles

Entrées TOR du module PCD2.H310

Nombre d'entrées	3 entrées codeur (A, B, IN) 1 entrée référence (REF)
Tension nominale d'entrée Niveau bas : Niveau haut :	24 V typique 0 à +4 V +15 à +30 V
Fonctionnement exclusivement en logique positive	
Courant d'entrée à 24 VCC	6 mA typique
Type de circuit	Sans séparation galvanique
Temps de réponse	30 µs

Entrées TOR du module PCD2.H311

Nombre d'entrées	6 entrées codeur (A, /A, B, /B, IN, /IN) Pas d'entrée référence
Tension nominale d'entrée	5 V typique
Niveaux signaux logiques	Entrées différentielles en RS 422
Hystérésis	200 mV maxi
Résistance de terminaison	150 Ω

Sortie analogique de régulation sur PCD2.H310 et PCD2.H311

Résolution	12 bits (dont signe)
Protection contre les courts-circuits	Oui
Séparation galvanique	Non
Tension de sortie *)	±10 V, précision de réglage : ±5 mV
Logique	Positive (commutation du « + »)
Impédance de charge minimum	3 kΩ

*) Cette tension étant préréglée en usine, il est vivement déconseillé de toucher au potentiomètre correspondant.

Alimentation 5 V du codeur 5 V sur PCD2.H311

Sortie 5 V	Alimentation 5 V du codeur
Protection contre les courts-circuits	Oui
Séparation galvanique	Non
Tension de sortie	5 V
Courant de charge maximum	300 mA
Courant de court-circuit (alimente aussi le bus 5 V du PCD1/2)	400 mA

Conditions de fonctionnement

Température ambiante	Service :	0 à +50 °C (sans ventilation forcée)
	Stockage :	–20 à +85 °C
Immunité aux parasites		Marquage CE selon EN 50 081-1 et EN 50 082-2

Voyants de signalisation d'état

Nombre de voyants : 5

Voyant		Fonction
« A »		Etat entrée codeur « A »
« B »		Etat entrée codeur « B »
« IN »		Etat entrée index codeur
« Ref »	(sur H310)	Etat contact de référence
« Pw 5V »	(sur H311)	Alimentation codeur 5 V
« Power »		Présence du ±15 V

Contrôle par interrogation logicielle

Input Power (adresse 08)	Surveillance des alimentations
Input Ref (adresse 11)	Interrogation de l'état logique du contact de référence (sur H310)
Input Pw5V (adresse 11)	Surveillance de l'alimentation 5 V (sur H311)
Input Version (adresse 12)	Version du module (H310 ou H311)

Généralités

Processeur	LM 628
Programmation	Par programme utilisateur PCD (PG4), assisté d'une bibliothèque de blocs et boîtes de fonctions.

Références de commande

PCD2.H310	1 axe pour codeur 24 VCC
PCD2.H311	1 axe pour codeur 5 V en RS 422
PCD9.H31E	Bibliothèque de blocs de fonctions (FB)
PCD8.H31	Outil de mise en service sous FUPLA

2.2 Caractéristiques électriques

Consommation de courant interne (sans codeur)

+5 V	125 mA typique, 150 mA maxi
Uext	10 mA typique, 15 mA maxi

Alimentation externe

Bornes « + » et « – »	24 V (19 à 32 V) lissée, ondulation maxi 10 %
Entrées TOR	4 (H310) ou 6 (H311) Cf. § 2.1
Sortie analogique	1 Cf. § 2.1

Attention :

Le courant maxi fourni au bus 5 V est de 1600 mA pour un PCD2 et de 750 mA pour un PCD1.

Il incombe donc à l'utilisateur de s'assurer que le courant total consommé par **toutes** les cartes enfichées dans le PCD2/PCD1 **et** dans l'extension C100 ou C150 ne dépasse pas cette limite (le courant fourni au codeur 5 V faisant partie de ces 1600 mA).

Si votre installation met en œuvre un boîtier d'extension, il convient de veiller à ne placer les modules PCD2.H31x **que** dans l'automate de base et de réserver à l'extension des modules d'E/S standards ; à défaut, la chute de potentiel sur le câble de liaison risque d'atteindre des valeurs trop élevées.

2.3 Caractéristiques de fonctionnement

Nombre de systèmes	1
Fréquence de comptage	jusqu'à 50 kHz

Paramètres de mouvement

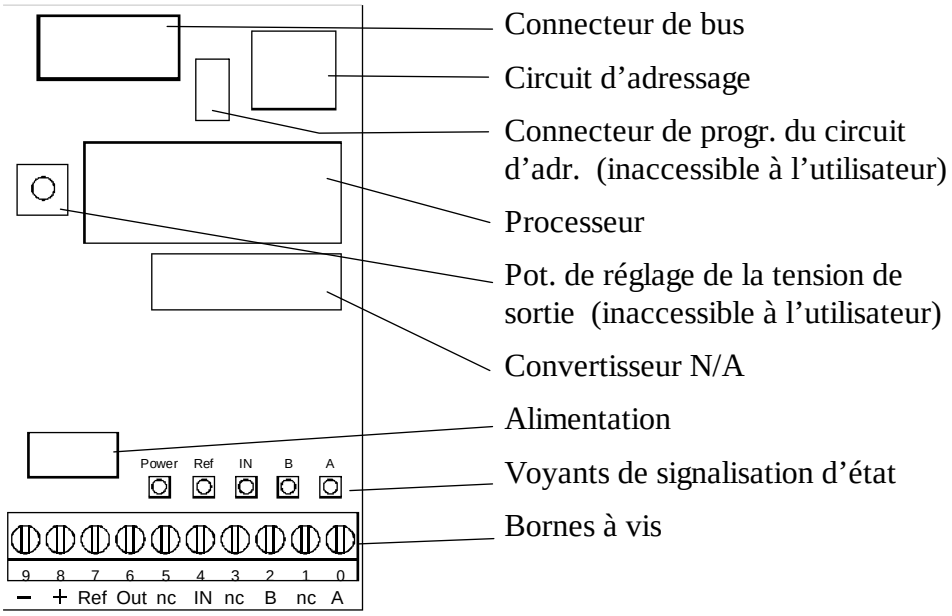
Des registres de 31 bits sont utilisés pour la position cible, la vitesse et l'accélération (plage $\pm 2^{30}$).

Position	Résolution programmable (en fonction des données mécaniques)
Vitesse	Résolution programmable (en fonction des données mécaniques)
Accélération	Résolution programmable (en fonction des données mécaniques)
Régulation PID	Temps d'échantillonnage 341 µs Actions Proportionnelle, Intégrale et Dérivée programmables (le temps d'échantillonnage de la dérivée est réglable indépendamment).

Notes personnelles :

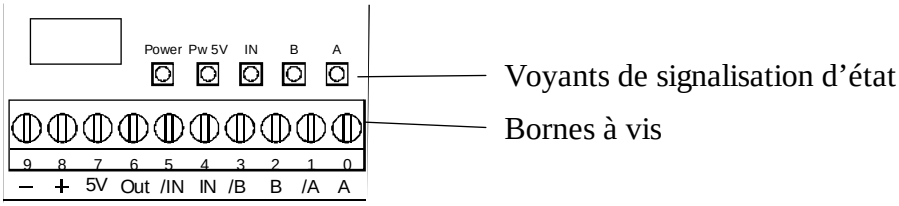
3. Constitution

Module PCD2.H310 équipé (codeur 24 V)



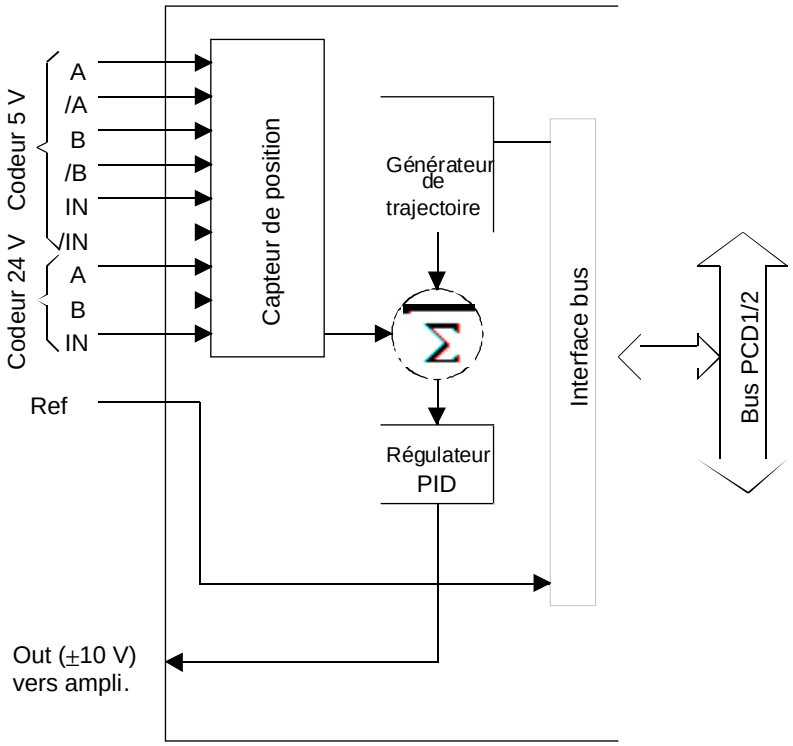
– et +	Alimentation externe V_{ext}
Ref	Entrée TOR du contact de référence
Out	Sortie analogique du régulateur PID
A, **B**, **IN**	3 entrées codeur
nc	Bornes inutilisées (non connectées)

Module PCD2.H311 équipé (codeur 5 V)



– et +	Alimentation externe V_{ext}
5V	Sortie 5 V d'alimentation codeur (300 mA maxi)
Out	Sortie analogique du régulateur PID
A, **B**, **IN**	3 entrées codeur non barrées
/A, **/B**, **/IN**	3 entrées codeur barrées

Schéma synoptique

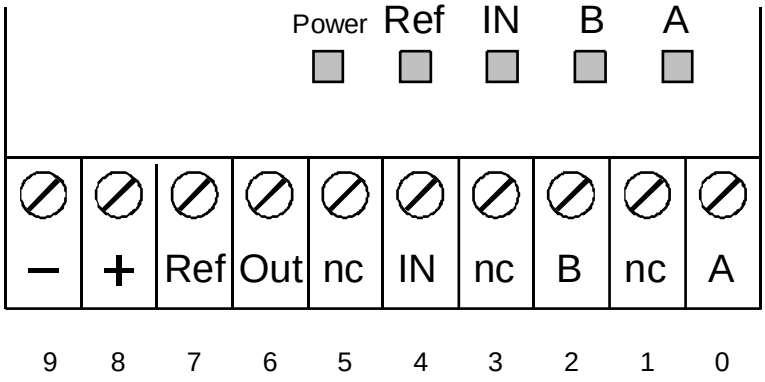


Les autres entrées et sorties de l'axe (fins de course, contact de référence du PCD2.H311, signal de validation « Enable » de l'entraînement…) doivent être surveillées et pilotées par un module d'E/S standard (PCD2.B100, PCD2.E110/E111 ou PCD2.A410).

4. Repérage des bornes et signification des voyants de signalisation d'état

Bornes à vis du PCD2.H310 (codeur 24 V)

La figure ci-dessous illustre le marquage des bornes sur le circuit imprimé. Le connecteur d'E/S est numéroté de 0 à 9 (de droite à gauche).



Entrées **4**

Borne 0 =	A :	Entrée codeur « A »
Borne 1 =	nc :	Inutilisée *)
Borne 2 =	B :	Entrée codeur « B »
Borne 3 =	nc :	Inutilisée *)
Borne 4 =	IN :	Entrée codeur « IN »
Borne 5 =	nc :	Inutilisée *)
Borne 7 =	Ref :	Entrée TOR du contact de référence

Sortie **1**

Borne 6 =	Out :	Sortie analogique du régulateur PID

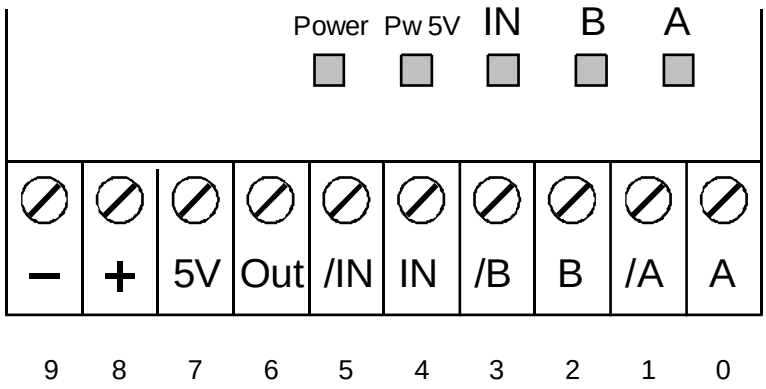
Alimentation

Borne 8 =	+	+ 24 VCC
Borne 9 =	–	Masse

*) **Important :** Ne raccordez aucun câble à ces bornes.

Bornes à vis du PCD2.H311 (codeur 5 V)

La figure ci-dessous illustre le marquage des bornes sur le circuit imprimé. Le connecteur d'E/S est numéroté de 0 à 9 (de droite à gauche).



Entrées **6**

Borne 0 =	A :	Entrée codeur « A »
Borne 1 =	/A :	Entrée codeur « /A »
Borne 2 =	B :	Entrée codeur « B »
Borne 3 =	/B :	Entrée codeur « /B »
Borne 4 =	IN :	Entrée codeur « IN »
Borne 5 =	/IN :	Entrée codeur « /IN »

Sorties **2**

Borne 6 =	Out :	Sortie analogique du régulateur PID
Borne 7 =	5 V :	Sortie 5 V d'alimentation codeur (300 mA maxi)

Alimentation

Borne 8 =	+	+ 24 VCC
Borne 9 =	–	Masse

Signification des voyants de signalisation d'état

Ref (réservé au H310) indique l'état logique du contact de référence ; il s'allume en cas de demande de référence ou en l'absence de câblage du contact.

Pw5V (réservé au H311) signale l'alimentation du codeur raccordé au module ; il s'allume pour indiquer le bon fonctionnement du 5 V (absence de court-circuit).

Power signale la présence du ±15 V ; il s'allume pour indiquer le bon fonctionnement des deux alimentations 15 V.

A et **B** renseignent sur les entrées codeur du même nom ; ils s'allument pour indiquer l'état logique Haut.

IN représente l'entrée d'index du codeur ; il s'allume pour signaler l'état actif (logique négative pour le H310).

Attention : Ce module intègre des composants particulièrement sensibles aux décharges électrostatiques.

Notes personnelles :

5. Description fonctionnelle

5.1 Modes d'exploitation

On distingue principalement deux modes d'exploitation :

- **Régulation de position**
- **Régulation de vitesse de rotation**

Régulation de position (lancée par l'instruction « StartMot »)

La régulation de position se déroule en trois étapes :

1) Saisie de la position et paramétrage du profil de vitesse,
2) Lancement du positionnement,
3) Attente du signal « **atteinte de la position cible »**.

Dans ce mode, après saisie des divers paramètres de mouvement (fonctions PID, accélération, vitesse, etc.), on effectue un déplacement régulé vers une position cible donnée. Vitesse, réglages PID et cible sont modifiables en cours de déplacement.

Régulation de vitesse de rotation (lancée par l'instruction « GoForw » ou « GoBackw »)

La régulation de vitesse de rotation se déroule en trois étapes :

1) Paramétrage du profil de vitesse,
2) Lancement du déplacement,
3) Arrêt du déplacement par une instruction **« Stop »**.

Dans ce mode, on accélère jusqu'à la vitesse de consigne, à l'accélération donnée. Le déplacement régulé se poursuit à cette vitesse jusqu'à l'intervention d'une commande d'arrêt. La vitesse de consigne est modifiable en cours de mouvement.

Les principaux maillons de la chaîne de régulation

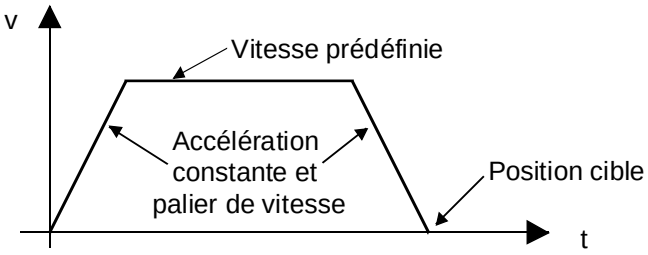
Le module de positionnement PCD2.H31x est constitué de cinq grandes unités fonctionnelles (Cf. synoptique de la page 3-2) :

- Un générateur de profil de vitesse,
- Un régulateur PID,
- Un capteur de position et un circuit d'entrées,
- Une interface de bus (CPLD) au bus PCD1/PCD2,
- Un convertisseur N/A destiné à la sortie analogique de régulation.

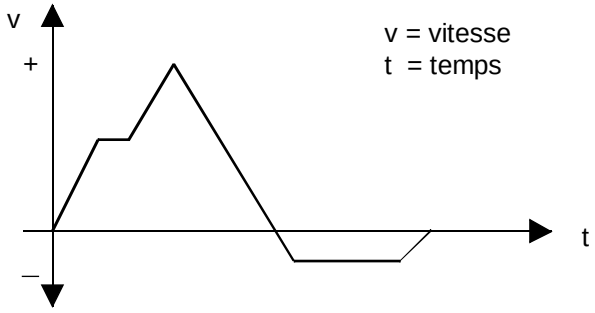
5.2 Générateur de profil de vitesse

En modes régulation de position et de vitesse de rotation, le générateur de profil calcule la vitesse théorique en fonction des paramètres d'accélération et vitesse prédéfinis et du temps.

En régulation de position, l'écart entre position cible et position réelle est transmis au régulateur PID en cours de mouvement, ce qui garantit l'extrême précision du positionnement.



Profil de vitesse type



Exemple de profil de vitesse avec modification de la vitesse de consigne et de la position cible en cours de mouvement

Vitesse et position cible peuvent être modifiées en n'importe quel point de la trajectoire ; dans ce cas, le régulateur accélère ou freine, à l'accélération donnée. Les rampes d'accélération et de freinage sont symétriques.

En régulation de vitesse de rotation, le régulateur accélère pour atteindre la vitesse de consigne paramétrée par l'utilisateur, puis le déplacement se poursuit à une vitesse constante jusqu'à réception d'une commande d'arrêt.

Principe de fonctionnement de la régulation de vitesse :

La position cible est constamment augmentée (en fonction de la vitesse de consigne). L'écart entre la cible et la position réelle, enregistré par le codeur, est transmis au régulateur PID qui compense immédiatement ces variations de vitesse (dues à des perturbations d'origines diverses) par une augmentation ou une diminution de sa sortie.

Si le moteur ne réussit pas atteindre la vitesse de consigne (en raison, par exemple, d'un blocage du rotor), l'écart entre la cible et la position réelle est considérable ; cela provoque l'apparition d'un message d'erreur de position, qui peut déclencher une interruption ou un arrêt automatique du moteur. L'erreur de position maximale admissible est programmable.

5.3 Régulateur PID

Le régulateur PID aide le moteur à approcher la position cible avec une précision maximale et à s'y maintenir ; il reste activé jusqu'à réception d'une commande d'arrêt.

Le régulateur PID exécute l'algorithme ci-dessous :

$$U(n) = kp * e(n) + ki * \sum_{N=0}^{n} e(n) + kd * [e(n') - e(n'-1)]$$

Avec :
- U(n) → Sortie de régulation du moteur
- e(n) → Erreur de position au $n^{ième}$ échantillonnage
- n → Échantillonnage de l'action intégrale
- n' → Échantillonnage de l'action dérivée
- kp → Action proportionnelle
- ki → Action intégrale
- kd → Action dérivée

Paramètres programmables :

- Termes kp, ki, kd
- Temps d'échantillonnage de la dérivée
- Limite d'intégration (IL) de l'intégrale

Les **coefficients PID kp, ki** et **kd** sont modifiables en cours de mouvement.

Le **temps d'échantillonnage de l'action Proportionnelle** et de **l'action Intégrale** est de 341µs ; la valeur de sortie du régulateur est donc rafraîchie toutes les 341 µs.

Le **temps d'échantillonnage de l'action Dérivée** est réglable par pas de 341 µs (soit 256 ∗ 341 µs maxi). Á basses vitesses, il convient d'augmenter le temps d'échantillonnage.

Limite d'intégration IL
Cette limite s'applique à la formule :

$$ki * \sum_{N=0}^{n} e(n)$$

5.4 Capteur de position et circuit d'entrées

Enregistrement de la position et de la vitesse de rotation

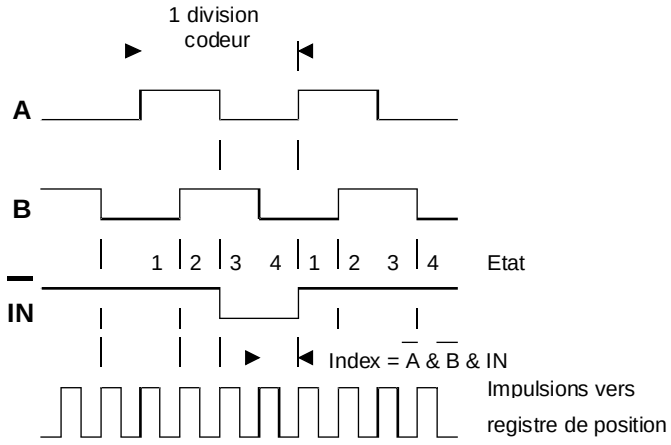
La position précise et/ou la vitesse de rotation du moteur sont saisies par un codeur incrémental. Il est possible de raccorder les signaux codeur suivants :

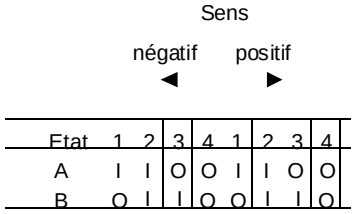
PCD2.H310 : Bornes A, B et IN
Signaux 24 V en logique positive

PCD2.H311 : Bornes A et /A, B et /B, IN et /IN
Entrées différentielles 5 V en RS 422

- Entrées A, B et /IN :

Diagramme d'état des signaux A, B et /IN au niveau du capteur :





- Entrées A et B :

Chaque changement d'état des deux signaux A et B délivrés par le codeur (passage de 0 à 1 et de 1 à 0) incrémente ou décrémente le registre de position interne d'une unité ; cela permet d'obtenir une résolution quatre fois plus fine que celle du codeur. Par conséquent, suivant ce principe, il faut aussi multiplier par 4 la saisie de la position cible.

Pour le capteur de position, les signaux doivent exactement suivre la séquence illustrée ci-dessus.

- Entrée IN :

Sur le H310 (codeur 24 V), l'entrée IN peut servir d'entrée d'index (top « zéro » en provenance du codeur) ou d'entrée de référence.

- Entrée d'index :

  A chaque fois que les 3 signaux codeur A, B et IN sont à 0 et que l'instruction « SetIdxPos » (définition de la position d'index) est exécutée, la position absolue du déplacement est écrite dans le registre de position d'index.

- Entrée de référence :

  Il est possible de raccorder un contact de référence afin, par exemple, de définir la position 0.

Raccordement de signaux codeur 5 V

On utilise dans ce cas un PCD2.H311, raccordé obligatoirement au codeur par un câble blindé :

- de longueur maxi 20 m
- de section maxi 0,25 mm^2

Exemples : câble PCD2.K271 ou PCD2.K273

Contact de référence

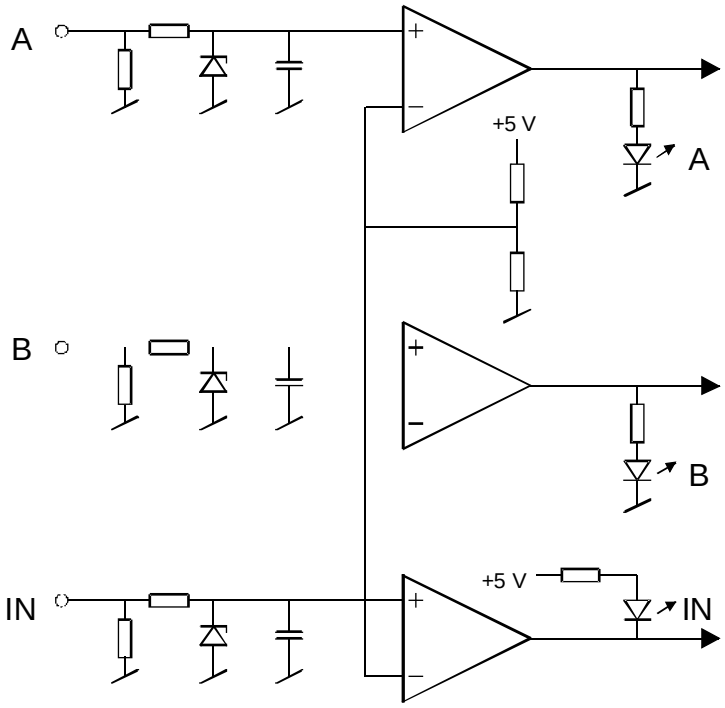
Le PCD2.H310 est le seul à posséder une entrée « Ref » (24 VCC en logique positive). Le signal de ce contact étant directement transmis au bus PCD2, cette entrée doit être sous contrôle du programme utilisateur pour déclencher l'action qui s'impose.

Sur le H311, il faut relier le contact de référence à un module d'E/S TOR standard.

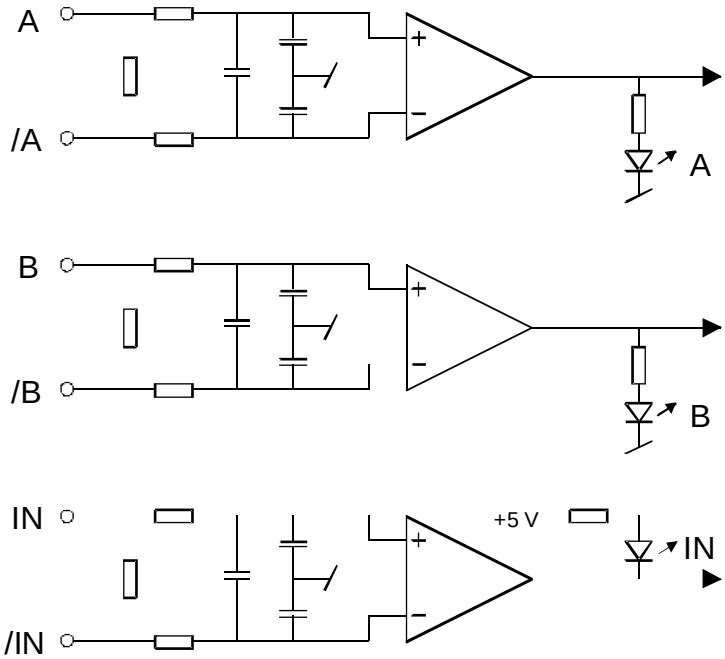
Dans le bloc de retour à l'origine « Home », le contact de référence est associé à des fins de course qui, sur les deux modules (H310 et H311), doivent être reliés à un module d'E/S TOR.

Raccordement des entrées

PCD2.H310



PCD2.H311

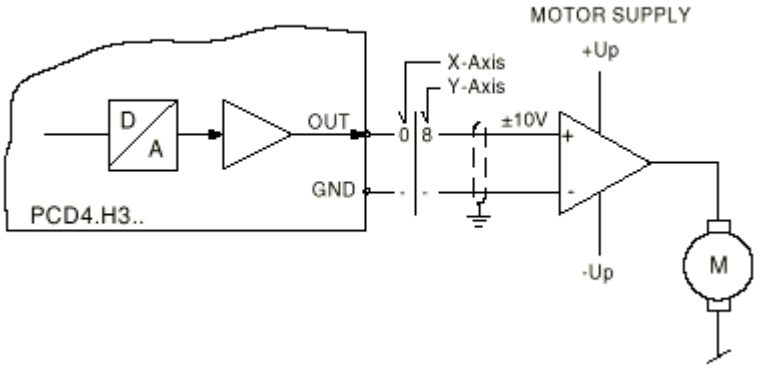


5.5 Convertisseur numérique-analogique

Les modules PCD2.H310 et H311 possèdent tous deux une sortie analogique de régulation du moteur.

Ils intègrent un convertisseur N/A de résolution 12 bits.

Raccordement de la sortie analogique :



5.6 Complément d'information sur la fonction Retour à l'origine (bloc de fonctions « Home »)

La prise d'origine peut être exécutée de façon indépendante à l'aide du bloc Home ; sept paramètres en définissent le parcours (Cf. exemple d'utilisation au § 12.1).

Il faut d'abord initialiser l'axe à référencer avec le bloc Init. Pour exploiter au mieux le bloc Home, l'axe et le contact de référence doivent être situés entre « LS1 » et « LS2 » (Cf. schéma de la page suivante).

La prise d'origine se déroule comme suit :

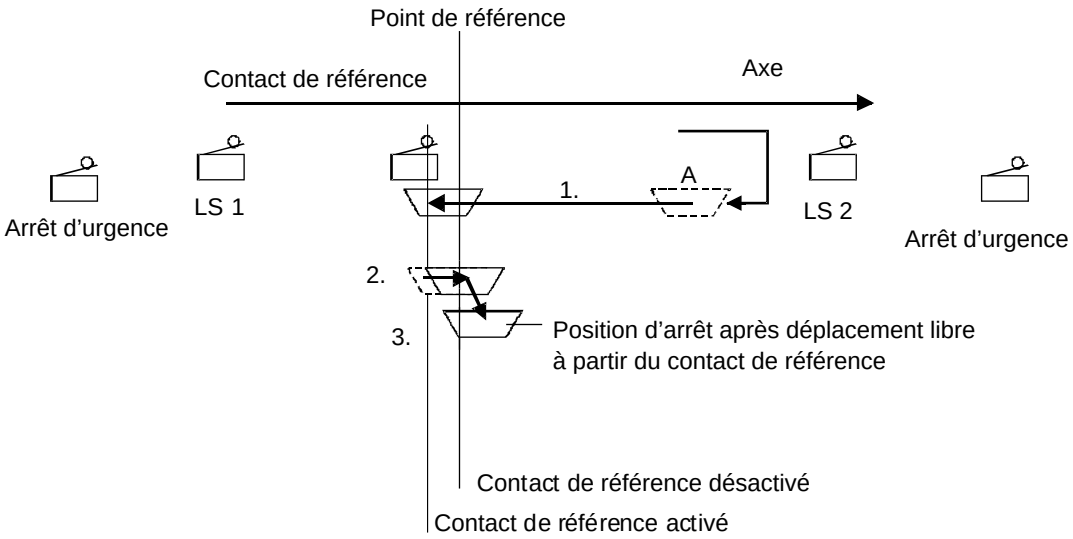
1) La recherche du contact de référence s'effectue à la vitesse Vmax du paramètre 5, dans le sens indiqué par le paramètre 2. Si, à défaut de contact de référence, l'axe rencontre un fin de course, le sens de la recherche est inversé.

2) L'entrée TOR à laquelle est raccordé le contact de référence est définie par le paramètre 7. Pour un PCD2.H310, son adresse correspond à l'adresse de base du module majorée de 11 ; pour un H311, elle est indifférente (module d'entrée TOR standard).

3) Lorsque le contact de référence est trouvé, le déplacement libre commence, dans le sens donné par le paramètre 3 et à la vitesse Vmin du paramètre 4.

4) Le contact de référence désactivé, l'axe se déplace jusqu'au signal d'index codeur suivant, puis s'arrête. Cette position devient alors la position 0.

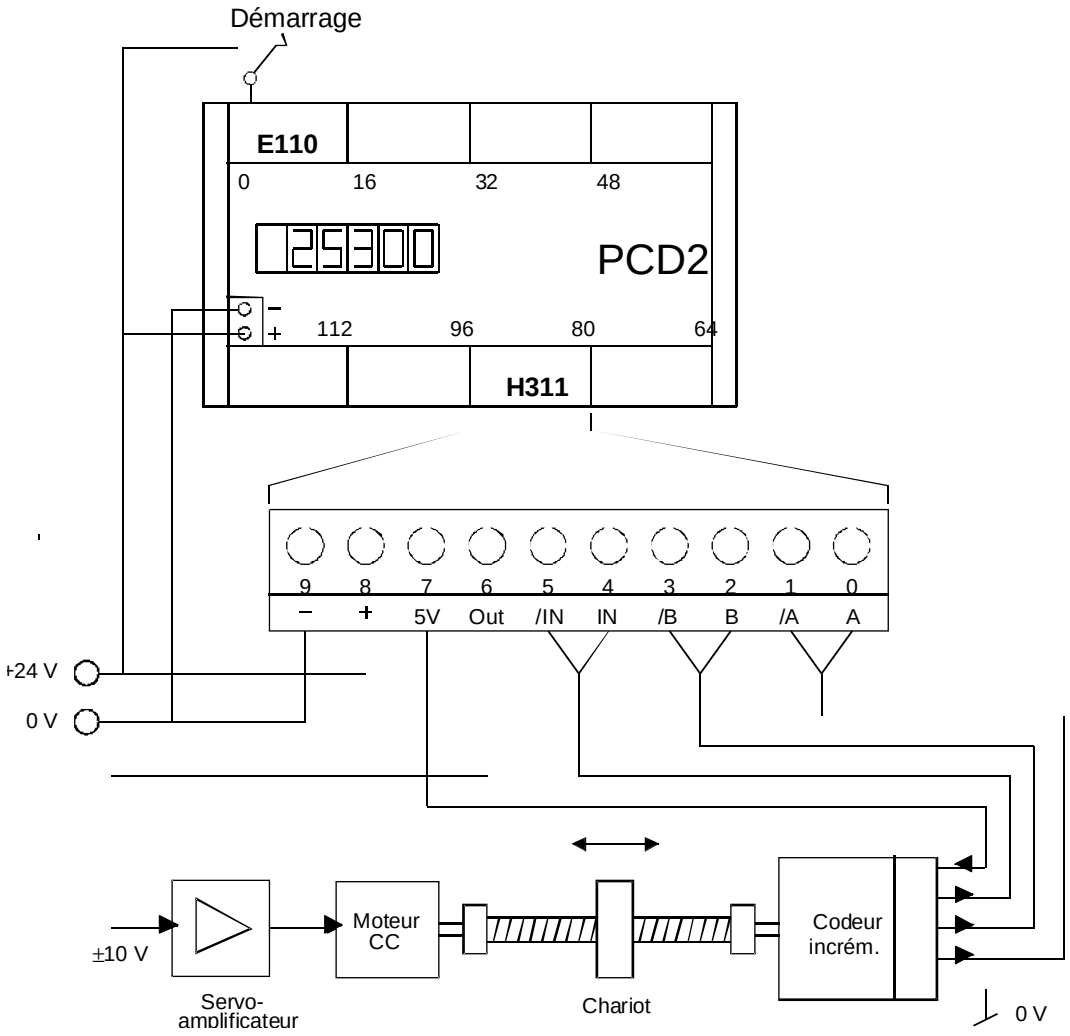
5) Le module est configuré avec les paramètres initiaux du bloc Init et le traitement du bloc Home s'interrompt.



Quelques précisions :

- Sur un même PCD, il est possible de référencer plusieurs modules à la fois.
- Pour démarrer la prise d'origine, l'indicateur « fEndHome_x » doit être à l'état bas.
- Les indicateurs « fLS1_x » et « fLS2_x » représentent les fins de course. Pour connaître leur état logique, il faut interroger les deux entrées TOR auxquelles ils sont connectés.
- Pour ces interrupteurs, il convient d'utiliser des contacts repos.
- Durant le retour à l'origine, les fins de course sont désactivés par l'électronique interne ; ils ne servent qu'à inverser le sens du déplacement.
- L'absence de contact de référence entraîne le positionnement de l'indicateur d'erreur « fHomeErr_x » (x étant le n° de module) et l'interruption du bloc Home.
- Au terme de l'exécution (correcte ou incorrecte) du bloc Home, l'indicateur « fEndHome_x » est positionné automatiquement.
- Si le traitement du bloc Home ne prend normalement fin qu'après exécution correcte du retour à l'origine ou sur détection d'une erreur, il est aussi possible d'y mettre un terme à l'aide d'une temporisation (paramètre 6) fixant le délai d'abandon (en secondes) du bloc Home. Il y a alors positionnement de l'indicateur « fHomeErr_x » et chargement du registre de diagnostic « rDiag » avec le code 6 (troisième octet). Le contrôle du paramétrage s'effectue comme exposé au chapitre 8.

6. Prise de contact



Configuration minimale requise pour piloter un axe par PCD2.H310 ou H311 (sans contact de référence, ni fins de course) :

- PCD1 ou PCD2, équipé d'au moins 1 module PCD2.H310/H311
  + afficheur F510/F530
  1 module PCD2.E110
- Mécanique de positionnement : moteur CC, arbre et codeur incrémental
- Moteur CC et réducteurs : env. 500 tours/min sous 10 VCC
  Pas de la vis : 1 mm/tour
  Codeur incrémental : 1000 points/tour
- Servo-amplificateur à 4 quadrants
  ex. : Ampli. op. LM 12 (National Semiconductor)
  Pour plus d'informations, consultez le site Internet :
  http://www.national.com/pf/LM/LM12.html
- PG4 (à partir de la version V2.0.70) et bibliothèque de blocs de fonctions PCD9.H31E

6.1 Bien démarrer avec la programmation en liste d'instructions (IL)

Pour simplifier au maximum la mise en service d'une commande d'axe régulée, nous vous proposons le petit programme suivant.

Un programme utilisateur bien écrit doit normalement être exempt de boucles d'attente. Ce n'est toutefois pas la démarche retenue pour notre premier exemple, qui vise à illustrer les principales instructions d'un PCD2.H31x. Dans la pratique, il convient de toujours choisir une structure GRAFTEC ou, à l'avenir, une structure FUPLA pour ce type de programme (Cf. § 6.1.2 et chapitre 10).

Exemple de tâche :

Après activation de l'entrée de démarrage « Start » du PCD2, le chariot se déplace dans un sens, marque une pause, puis revient à son point de départ. Au démarrage, il est censé se situer approximativement au milieu de l'axe.

Cet exemple entend décrire le principe de fonctionnement du positionnement et son programme utilisateur. Précisons que le programme débute par les directives assembleur « $include » et « $group ».

La signification et la définition de chacun des paramètres imposent une connaissance des automatismes mis en œuvre (Cf. chapitre 9).

Le premier exemple, avec boucles d'attente, s'intitule « intro-1.src » (Cf. § 6.1.1) ; le même exemple, cette fois édité en GRAFTEC, est renommé « intro-2.sfc » (Cf. § 6.1.2).

Les blocs de fonctions (IL sous PG4 à partir de la version V2.0.70) figurent sur la disquette PCD9.H31E. Pour les installer sur PC, suivez les consignes du chapitre 7 et consultez le fichier « README.TXT » (également sur la disquette).

Le nombre de modules (1) et l'adresse du PCD2.H311 (80) doivent être saisis dans le fichier D2H310_B.MBA, à savoir :

```
NbrModules   EQU   1     ; Nb. de modules H31x utilisés (0 à 16)

BA_1         EQU   80    ; Adresse de base du module n° 1
```

Ce fichier sur disquette devant résider dans le répertoire du projet affecté à l'exemple, il faut le copier dans le répertoire du projet en cours.

6.1.1 Exemple simple de programme édité en liste d'instructions avec boucles d'attente : « intro-1.src »

```
$include D2H310_B.equ
$group H310

        xob    16

        LD   R 1000
               4.0          ;  Données mécaniques
        LD   R 1001
               8000         ;  Vitesse absolue initiale
        LD   R 1002
               10000        ;  Accélération absolue initiale

        CFB    Init         ; Initialisation du module
             K 1            ;  Numéro de module
               250          ;  Proportionnelle (régulateur PID)
               0            ;  Intégrale (régulateur PID)
               0            ;  Dérivée (régulateur PID)
               4000         ;  Limite d'intégration
               5            ;  Temps d'échantillonnage de la dérivée
               500          ;  Tolérance de position
               0            ;  Comportement en cas d'erreur de pos.
             R 1000         ;  Registre des données mécaniques
             R 1001         ;  Registre de la vitesse initiale
             R 1002         ;  Registre de l'accélération initiale

        exob
; ---------------------------------------------------------------

        cob    0
               0

start:  sth  i 0

        CFB    Exec         ; Exécution du bloc de fonctions
             K 1            ;  Numéro de module
               RdActPos     ;  Instruction : lecture position réelle
             R 90           ;  Registre de la position réelle
        DSP  R 90           ; Affichage du contenu du registre

        jr   l start        ; Absence de signal « Start » : attente

        LD   R 100          ; Chargement de la position
               20000        ;  cible absolue

        CFB    Exec         ; Exécution du bloc de fonctions
             K 1            ;  Numéro de module
               LdDestAbs    ;  Instruction : charg. pos. cible abs.
             R 100          ;  Registre de la position cible absolue

        CFB    Exec         ; Exécution du bloc de fonctions
             K 1            ;  Numéro de module
               StartMot     ;  Instruction : démarrage déplacement
               rNotUsed     ;  Registre fictif
```

```
pos1:   CFB    Exec       ; Exécution du bloc de fonctions
             K 1          ;  Numéro de module
               RdActPos   ;  Instruction : lecture position réelle
             R 90         ;  Registre de la position réelle
        DSP  R 90         ; Affichage du contenu du registre

        CFB    Exec       ; Exécution du bloc de fonctions
             K 1          ;  Numéro de module
               RdStatRg   ;  Instruction : lecture registre d'état
             R 0          ;  Valeur du registre d'état

        STH    fOnDest_1  ; Position atteinte ?
        jr   l pos1       ; Dans la négative, attente (boucle)

        ld   t 0          ; Chargement d'une pause
               50         ;  de 5 secondes dans temporisateur
pause1: sth  t 0          ; Pause écoulée ?
        jr   h pause1     ; Dans la négative, attente (boucle)

        LD   R 100        ; Chargement de la position
               0          ;  cible absolue

        CFB    Exec       ; Exécution du bloc de fonctions
             K 1          ;  Numéro de module
               LdDestAbs  ;  Instruction : charg. pos. cible abs.
             R 100        ;  Registre de la position cible absolue

        CFB    Exec       ; Exécution du bloc de fonctions
             K 1          ;  Numéro de module
               StartMot   ;  Instruction : démarrage déplacement
               rNotUsed   ;  Registre fictif

pos2:   CFB    Exec       ; Exécution du bloc de fonctions
             K 1          ;  Numéro de module
               RdActPos   ;  Instruction : lecture position réelle
             R 90         ;  Registre de la position réelle
        DSP  R 90         ; Affichage du contenu du registre

        CFB    Exec       ; Exécution du bloc de fonctions
             K 1          ;  Numéro de module
               RdStatRg   ;  Instruction : lecture registre d'état
             R 0          ;  Valeur du registre d'état

        STH    fOnDest_1  ; Position atteinte ?
        jr   l pos2       ; Dans la négative, attente (boucle)

        ld   t 0          ; Chargement d'une pause
               50         ;  de 5 secondes dans temporisateur
pause2: sth  t 0          ; Pause écoulée ?
        jr   h pause2     ; Dans la négative, attente (boucle)

        ecob

$endgroup
```

Description du programme :

La directive « $include » intègre automatiquement le fichier D2H310_B.equ, qui contient à son tour le fichier D2H310_B.mba indiquant le nombre de modules H31x et leur adresse de base.

La directive « $group H310 » stipule que le code programme se déroulant jusqu'à la directive « $endgroup » appartient au PCD2.H31x.

L'initialisation du module a lieu dans le bloc de démarrage à froid XOB 16. Avant de traiter le bloc de fonctions Init, il faut charger les 3 registres contenant les données mécaniques, la vitesse et l'accélération. Le choix de chacun de ces paramètres est expliqué au chapitre 9 « Installation et mise en service ».

Le bloc Init est ensuite appelé ; le choix de ses 11 paramètres est également décrit au chapitre 9.

Le programme de déplacement effectif se déroule dans le COB 0.

Le programme attend le signal de démarrage de l'entrée 0 du PCD. Pour permettre l'enregistrement de la position en cours (même durant cette phase) et son affichage, l'instruction « RdActPos » est insérée à la boucle d'attente ; si la condition de démarrage n'est pas remplie, la position sera donc lue en continu.

La position cible absolue est chargée dans le registre R 100 du PCD, puis transférée au module par l'instruction « LdDestAbs ».

La commande « StartMot » lance le déplacement (course de 20 mm).

Dans une boucle de programme, la position est lue en permanence par l'instruction « RdActPos » et affichée via le registre R 90 du PCD. Le déplacement obéit aux paramètres choisis dans le bloc de fonctions Init : il s'effectue dans des conditions optimales, piloté par le module lui-même, sans intervention du programme utilisateur, jusqu'à la cible. Pour permettre au programme de se dérouler correctement, il faut savoir à quel moment le déplacement est terminé. C'est le rôle de l'indicateur « fOnDest_x » (soit fOnDest_1 pour le module n° 1 de notre exemple), qui doit au préalable être activé par une instruction de lecture du registre d'état « RdStatRg ».

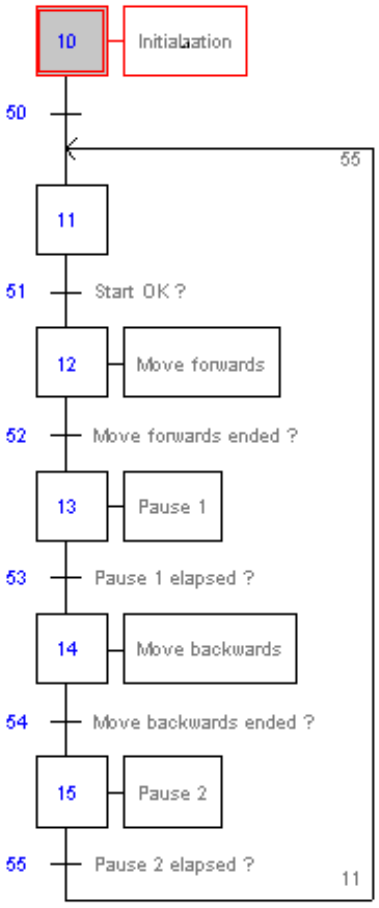
Cette partie de programme chargée de lire ou d'afficher la position en cours et d'activer ou d'interroger fOnDest_x boucle en continu tant que la position cible n'est pas atteinte.

Une fois la cible atteinte, le programme marque une pause (5 secondes dans cet exemple), puis la nouvelle position cible (0) est chargée et le chariot regagne son point de départ.

Pour connaître la position en cours réelle, même durant les pauses (stabilisation de la position finale), il faut également incorporer les instructions de lecture « RdActPos » et d'affichage « DSP R 90 » aux boucles d'attente de ces pauses.

6.1.2 Exemple simple de programme édité en GRAFTEC : « intro-2.sfc »

Il s'agit du même exemple que celui du § 6.1.1, mais cette fois en GRAFTEC, sans saut de programme ni boucle d'attente. Ses différentes étapes et transitions ont été éditées en langage IL.



Code programme de « intro-2.sfc »

(Pour obtenir cette représentation, le fichier « intro-2.sfc » doit être renommé « intro-2.src »).

```
  SB     0
;-------------------------------
  IST    10          ; Initialisation
         O 50

$include D2H310_B.equ
$group H310

  LD   R 1000
         4.00        ;  Données mécaniques
  LD   R 1001
         8000        ;  Vitesse absolue initiale
  LD   R 1002
         10000       ;  Accélération absolue initiale

  CFB    Init        ; Initialisation du module
       K 1           ;  Numéro de module
         250         ;  Proportionnelle (régulateur PID)
         0           ;  Intégrale (régulateur PID)
         0           ;  Dérivée (régulateur PID)
         4000        ;  Limite d'intégration
         5           ;  Temps d'échantillonnage de la dérivée
         500         ;  Tolérance de position
         0           ;  Comportement en cas d'erreur de position
       R 1000        ;  Registre des données mécaniques
       R 1001        ;  Registre de la vitesse initiale
       R 1002        ;  Registre de l'accélération initiale
  EST    ;10

;-------------------------------
  ST     11
       I 50
       I 55          ;  Pause n° 2 écoulée ?
       O 51          ;  Démarrage OK ?
  EST    ;11

;-------------------------------
  ST     12          ; Avance
       I 51          ;  Démarrage OK ?
       O 52          ;  Avance terminée ?

  LD   R 100         ; Chargement de la position
         20000       ;  cible absolue

  CFB    Exec        ; Exécution du bloc de fonctions
       K 1           ;  Numéro de module
         LdDestAbs   ;  Instruction : charg. pos. cible absolue
       R 100         ;  Registre de la position cible absolue

  CFB    Exec        ; Exécution du bloc de fonctions
       K 1           ;  Numéro de module
         StartMot    ;  Instruction : démarrage déplacement
         rNotUsed    ;  Registre fictif
   EST   ;12
```

```
;--------------------------------
  ST     13          ; Pause n° 1
       I 52          ;  Avance terminée ?
       O 53          ;  Pause n° 1 écoulée ?

  ld   t 0
         50
   EST   ;13

;--------------------------------
  ST     14          ; Recul
       I 53          ;  Pause n° 1 écoulée ?
       O 54          ;  Recul terminé ?

  LD   R 100         ; Chargement de la position
         0           ;  cible absolue

  CFB    Exec        ; Exécution du bloc de fonctions
       K 1           ;  Numéro de module
         LdDestAbs   ;  Instruction : charg. pos. cible absolue
       R 100         ;  Registre de la position cible absolue

  CFB    Exec        ; Exécution du bloc de fonctions
       K 1           ;  Numéro de module
         StartMot    ;  Instruction : démarrage déplacement
         rNotUsed    ;  Registre fictif
  EST    ;14

;--------------------------------
  ST     15          ; Pause n° 2
       I 54          ;  Recul terminé ?
       O 55          ;  Pause n° 2 écoulée ?

  ld   t 0
         50
  EST    ;15

;--------------------------------
  TR     50
       I 10          ; Initialisation
       O 11
  ETR    ;50

;--------------------------------
  TR     51          ; Démarrage OK ?
       I 11
       O 12          ; Avance

  CFB    Exec        ; Exécution du bloc de fonctions
       K 1           ;  Numéro de module
         RdActPos    ;  Instruction : lecture position réelle
       R 90          ;  Registre de la position réelle
  DSP    R 90        ; Affichage du contenu du registre

  sth  i 0

  ETR    ;51
```

```
;--------------------------------
  TR     52          ; Avance terminée ?
       I 12          ;  Avance
       O 13          ;  Pause n° 1

  CFB    Exec        ; Exécution du bloc de fonctions
       K 1           ;  Numéro de module
         RdActPos    ;  Instruction : lecture position réelle
       R 90          ;  Registre de la position réelle
  DSP  R 90          ; Affichage du contenu du registre

  CFB    Exec        ; Exécution du bloc de fonctions
       K 1           ;  Numéro de module
         RdStatRg    ;  Instruction : lecture registre d'état
       R 0           ;  Valeur du registre d'état

  STH    fOnDest_1   ; Position atteinte ?
  ETR    ;52

;--------------------------------
  TR     53          ; Pause n° 1 écoulée ?
       I 13          ;  Pause n° 1
       O 14          ;  Recul

  stl  t 0
  ETR    ;53

;--------------------------------
  TR     54          ; Recul terminé ?
       I 14          ;  Recul
       O 15          ;  Pause n° 2

  CFB    Exec        ; Exécution du bloc de fonctions
       K 1           ;  Numéro de module
         RdActPos    ;  Instruction : lecture position réelle
       R 90          ;  Registre de la position réelle
  DSP  R 90          ; Affichage du contenu du registre

  CFB    Exec        ; Exécution du bloc de fonctions
       K 1           ;  Numéro de module
         RdStatRg    ;  Instruction : lecture registre d'état
       R 0           ;  Valeur du registre d'état

  STH    fOnDest_1   ; Position atteinte ?
  ETR    ;54

;--------------------------------
  TR     55          ; Pause n° 2 écoulée ?
       I 15          ;  Pause n° 2
       O 11

  stl  t 0

$endgroup
  ETR    ;55

  ESB    ;0
```

Remarques sur le programme

La connaissance du PG4 en général et la maîtrise du GRAFTEC en particulier sont des prérequis.

Au cours de l'assemblage, le bloc séquentiel SB 0 est automatiquement appelé à partir d'un COB.

Le déroulement du programme GRAFTEC est visualisable en ligne.

L'initialisation du H310 a lieu en IST 10. Dans ce cas de figure, cette étape n'est traitée qu'au premier appel du bloc séquentiel SB, comme pour l'XOB 16. Cette démarche est logique si l'initialisation du H310 s'effectue dans l'IST du SB gérant ce module de façon que l'ensemble du programme soit regroupé en un seul endroit. On préférera l'XOB 16 pour toute initialisation s'appliquant au PCD complet.

Les étapes ST 12 et ST 14 assurent le chargement de la position cible absolue dans le H310, par l'intermédiaire du registre R 100 du PCD, puis le lancement du déplacement.

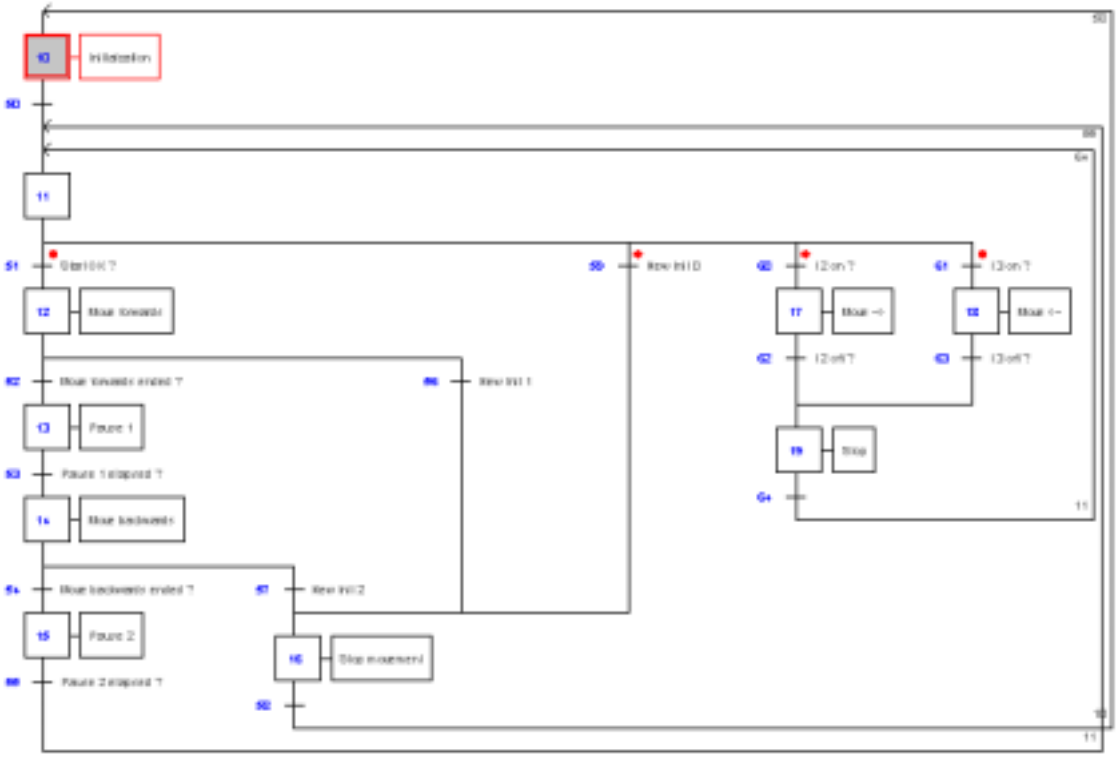
Les transitions TR 52 et TR 54 permettent de savoir, par interrogation logicielle, si le déplacement est terminé pour autoriser la poursuite du programme. Ce déplacement est sous contrôle direct du module. L'interrogation (STH fOnDest_1) est précédée de la lecture et de l'affichage de la position en cours, suivis du rafraîchissement de l'indicateur « fOnDest_1 » par l'instruction « RdStatRg ». Conformément aux règles du GRAFTEC, à chaque transition non satisfaite (la position cible n'étant pas encore atteinte, par exemple), le programme revient au COB appelant et poursuit son traitement puis, au cours du cycle de programme suivant, retraite **dans sa totalité** la transition non satisfaite. Ce principe garantit à chaque fois la lecture et l'affichage automatiques de la position ainsi que le rafraîchissement de « fOnDest_1 ».

Pour parfaire cette programmation, il faudrait que la position, dans une configuration d'essai, soit également lue et affichée durant les pauses, de façon à pouvoir visualiser l'état du chariot même en phase de stabilisation.

6.1.3 Programme simple de mise en service *)

Ce petit programme d'essai, dénommé « test-par.sfc » et inspiré du précédent « intro-2.sfc » (Cf. § 6.1.2), permet de tester les différents paramétrages afin d'optimiser le déplacement.

Fonction :



I 0 : Démarrage de l'avance et du recul (idem « intro-2.sfc »)

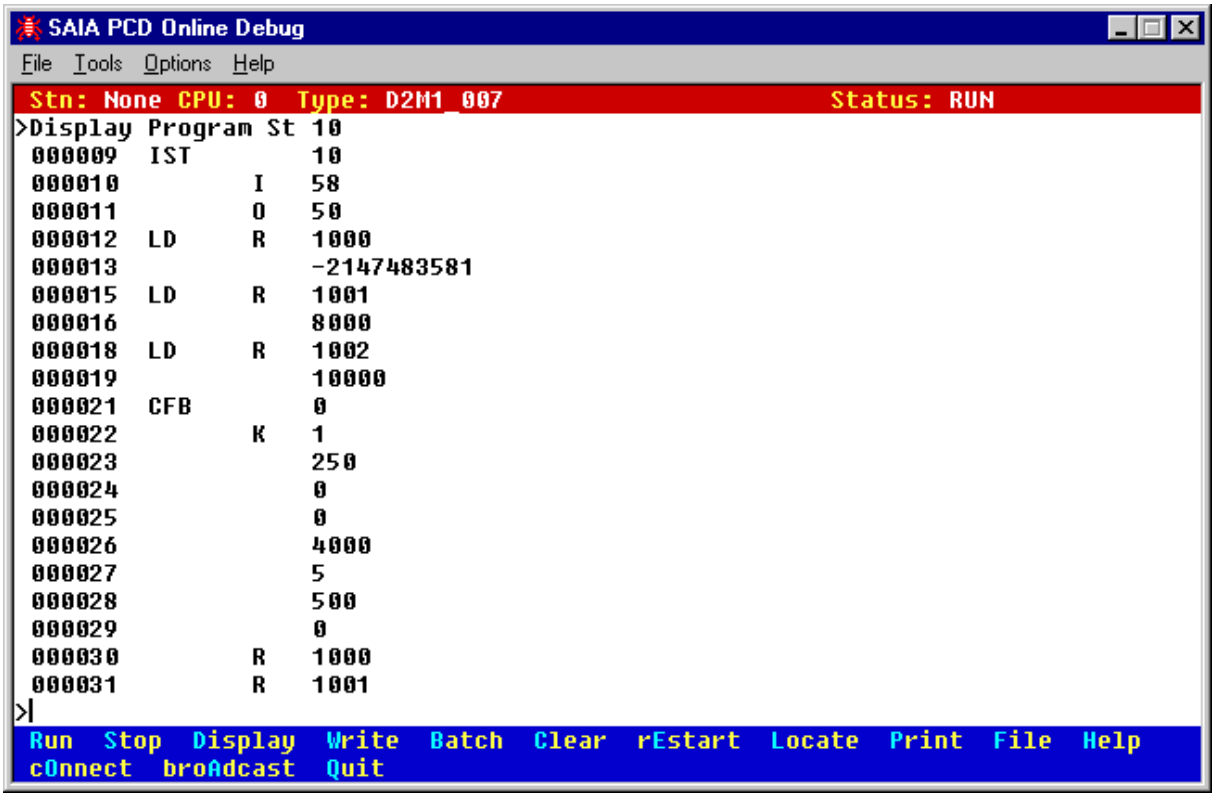
I 1 : Adoption des nouveaux paramètres, modifiés en ligne dans le débogueur. En activant I 1, le déplacement peut aussi s'arrêter brusquement.

I 2 : Partant de sa position d'arrêt, le chariot peut avancer selon les paramètres choisis, tant que I 2 est activée.

I 3 : Partant de sa position d'arrêt, le chariot peut reculer selon les paramètres choisis, tant que I 3 est activée.

Rappelons que les paramètres sont modifiables en ligne dans le débogueur. Pour repérer leurs adresses absolues, l'étape initiale IST 10 doit être affichée, voire imprimée par la commande :
<Display> <Program> <Step> <10> <CR>.

*) Un programme de mise en service sous FUPLA, plus convivial, vous est proposé au chapitre 10.




```
LD    R 1000
        4.0         ; Données mécaniques
LD    R 1001
        8000        ; Vitesse absolue initiale
LD    R 1002
        10000       ; Accélération absolue initiale

CFB     Init        ; Initialisation du module
      K 1           ;  Numéro de module
        250         ;  Proportionnelle (régulateur PID)
        0           ;  Intégrale (régulateur PID)
        0           ;  Dérivée (régulateur PID)
        4000        ;  Limite d'intégration
        5           ;  Temps d'échantillonnage de la dérivée
        500         ;  Tolérance de position
        0           ;  Comportement en cas d'erreur de position
      R 1000        ;  Registre des données mécaniques
      R 1001        ;  Registre de la vitesse initiale
      R 1002        ;  Registre de l'accélération initiale
EST     ;10
```

Par exemple, si l'on doit porter à 30 000 l'accélération absolue initiale de 10 000, il faut modifier la ligne de programme 19 par la commande :

<Write> <Program> <19> <CR> <30000> <CR> <Esc>

Déplacement à l'arrêt : activez puis désactivez l'entrée I 1 du PCD. Cela signifie que les modifications de paramètres sont adoptées. En activant I 0, le nouveau comportement peut être testé.

6.2 Bien démarrer avec la programmation en FUPLA

En préparation

Notes personnelles :

7. Programmation

L'automate PCD est programmé pour exploiter les modules de positionnement et de comptage PCD2.H.., par l'intermédiaire du programme utilisateur PCD associé au logiciel de programmation PG4 (à partir de la version V2.0.70).

(Pour les applications mettant en œuvre l'ancien outil de programmation PG3, il convient d'utiliser les blocs de fonctions du module PCD4.H3..).

La programmation s'effectue en langage IL (listes d'instructions) à l'aide de blocs de fonctions (abrégés « FB ») ou en FUPLA à l'aide de boîtes de fonctions (« FBox »).

Les blocs de fonctions sont regroupés sur une disquette référencée PCD9.H31E.

La commande d'axe faisant toujours appel à des tâches séquentielles, il est préférable d'écrire les programmes utilisateur en GRAFTEC, chaque étape et transition étant à leur tour éditées en langage IL (blocs de fonctions) ou en FUPLA (boîtes de fonction).

Le BLOCTEC et le FUPLA sont deux autres possibilités de programmation.

7.1 Programmer en liste d'instructions IL avec les blocs de fonctions « FB »

7.1.1 Constitution de la fourniture et installation des « FB »

Référence de commande de la disquette : PCD9.H31E.

Cette disquette est constituée de 6 répertoires :

- APPSDIR aide complète « helps »
- FB les fichiers .SRC et .EQU du H31x
- FBOX les boîtes de fonctions (FBoxes) du H31x
- PG3_FB tous les fichiers de blocs de fonctions du PG3
- PG4_FB les exemples et le fichier .MBA
- Readme les informations générales

L'ensemble est exploitable sous logiciel de programmation SAIA PG4, à partir de la version V2.0.70. Pour toutes les autres versions du PG4, consultez le fichier « Readme ». Cette fourniture comporte également les blocs de fonctions utilisables avec le PG3 (Cf. fichier « Readme »).

Les boîtes de fonction du FUPLA ne sont pas encore disponibles.

Procédure d'installation sous PG4

Pour simplifier votre tâche, utilisez le programme « Setup Extra Files » du PG4 :

1) Introduisez la disquette PCD9.H31E dans le lecteur A.
2) Cliquez sur le bouton « Start », pointez sur « Programs » et choisissez « SAIA PG4 », puis « Setup Extra Files ».

   Les blocs de fonctions « FB » et le fichier d'aide « Help » s'installent dans le répertoire « PG4 » du disque dur.

L'installation concerne les fichiers :

- D2H310_B.SRC Code source des FB (accessible en lecture seule)
- D2H310_B.EQU Définition des FB (accessible en lecture seule)

  Ces 2 fichiers, situés sous A:\FB, sont copiés dans le sous-répertoire …\PG4\FB du PG4.

- FB_LIB.HLP Bibliothèque de blocs de fonctions
- D2H310_B.HLP Aide des blocs de fonctions (help)

  Ces 2 fichiers, situés sous A:\APPSDIR, sont copiés dans le répertoire …\PG4 du PG4.

Le fichier **D2H310_B.MBA** de définition des adresses de base du module (l'extension « .MBA » signifiant *Module Base Addresses*) doit être copié **manuellement** du répertoire PG4_FB de la disquette dans le répertoire du projet en cours.

Le fichier **D2H310_B.MBA**, indispensable à l'utilisateur, est représenté ci-dessous :

```
;
; Fichier modifiable par l'utilisateur
;
; Adresses de base définies par l'utilisateur
; -----------------------------------
$group H310
NbrModules      EQU     1       ; Nb de PCD2.H310 utilisés (0 à 16)

;
; Adresses de base des modules(ne définir que les modules utilisés)

BA_1            EQU    32       ; Adresse de base du module 1
BA_2            EQU     0       ; Adresse de base du module 2
BA_3            EQU     0       ; Adresse de base du module 3
BA_4            EQU     0       ; Adresse de base du module 4
BA_5            EQU     0       ; Adresse de base du module 5
BA_6            EQU     0       ; Adresse de base du module 6
BA_7            EQU     0       ; Adresse de base du module 7
BA_8            EQU     0       ; Adresse de base du module 8
BA_9            EQU     0       ; Adresse de base du module 9
BA_10           EQU     0       ; Adresse de base du module 10
BA_11           EQU     0       ; Adresse de base du module 11
BA_12           EQU     0       ; Adresse de base du module 12
BA_13           EQU     0       ; Adresse de base du module 13
BA_14           EQU     0       ; Adresse de base du module 14
BA_15           EQU     0       ; Adresse de base du module 15
BA_16           EQU     0       ; Adresse de base du module 16
$endgroup
```

Ce fichier permet de saisir le nombre de PCD2.H31x installés ainsi que leur adresse de base. Ne figurant pas dans le Gestionnaire de projet, il faut passer par un éditeur de texte (par ex., SEDIT32) pour le modifier.

Numérotez les modules à la suite, en partant de « BA_1 ». Par exemple, si votre projet compte trois H310, vous aurez « BA_1 », « BA_2 » et « BA_3 ». L'affectation des connecteurs est laissée à votre convenance, comme l'illustre l'exemple ci-dessous :

```
NbrModules      EQU     3       ; Nb de PCD2.H310 utilisés (0 à 16)

;
; Adresses de base des modules(ne définir que les modules utilisés)

BA_1            EQU    64       ; Adresse de base du module 1
BA_2            EQU   208       ; Adresse de base du module 2
BA_3            EQU   112       ; Adresse de base du module 3
BA_4            EQU     0       ; Adresse de base du module 4
BA_5            EQU     0       ; Adresse de base du module 5
```

Les adresses de base des registres, indicateurs et blocs de fonctions sont automatiquement affectées ; elles apparaissent dans la liste des ressources, qui peut être consultée par les commandes « View → Resource List » du Gestionnaire de projet.

Le projet créé doit s'intituler « TEST-H3 » et le programme utilisateur « move-01.sfc ». Les fichiers sont organisés de la manière suivante :

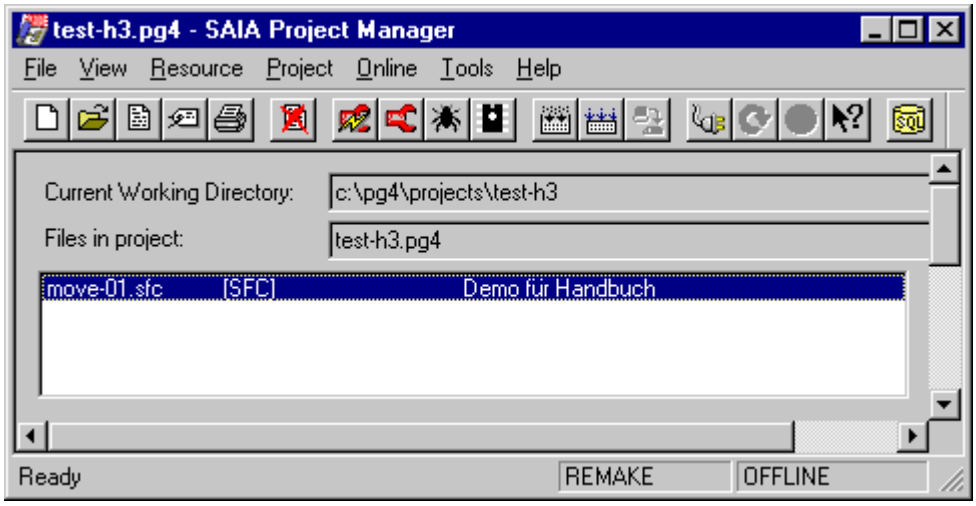
```
C:\PG4 \FB          \D2H310_b.equ
                    \D2H310_b.src
       \...
       \FBOX        \...
       \GALEP3      \...
       \PROJECTS    \FUP_E            (Exemple démo PG4)
                    \GRAF_E           (Exemple démo PG4)
                    \TEST-H3     \D2H310_b.mba
                                 \move-01.sfc
       \...
       \D2H310_b.hlp
```

Le programme utilisateur du H310 se présente comme suit :

```
$include D2H310_b.equ
$group H310

XOB     16


PCD-Code


ecob
$endgroup
```

Si le programme est édité en GRAFTEC, les directives assembleur « $include » et « $group » sont d'ordinaire placées dans la première étape (ST), qui constitue normalement l'étape initiale (IST).
La directive « $endgroup » vient clore la dernière transition (TR).

Une fois l'installation correctement effectuée, le programme utilisateur édité et tous les paramètres définis, le programme peut être traité et téléchargé dans le PCD par les commandes « Project → Build » du Gestionnaire de projet.



7.1.2 Description des blocs de fonctions

La bibliothèque est principalement constituée de 3 blocs de fonctions :

- INIT Initialisation Bloc de 11 paramètres
- EXEC Exécution Bloc de 3 paramètres
- HOME Retour à l'origine Bloc de 7 paramètres

L'appel du bloc INIT revêt toujours la forme suivante :
(Les valeurs ci-dessous sont données seulement à titre d'exemple.)

```
CFB       init      ; Initialisation d'un module PCD2.H31x
        k 1         ; Param.1 : n° de module (k 1 à k 16)
          250       ; Param.2 : fonction proportionnelle
          150       ; Param.3 : fonction intégrée
          10        ; Param.4 : fonction dérivée
          4000      ; Param.5 : limite d'intégration
          5         ; Param.6 : tps d'échantillon. dérivée
          500       ; Param.7 : tolérance de position
          0         ; Param.8 : comportement si erreur pos.
        r 1000      ; Param.9 : données mécaniques *)
        r 1001      ; Param.10: vitesse *)
        r 1002      ; Param.11: accélération *)
```

*) Avant tout traitement du bloc Init, il convient de charger les registres des paramètres 9, 10 et 11 avec les valeurs correctes.

Dans la majorité des cas, le bloc EXEC revêt la forme suivante :

```
CFB       exec
        k 1         ; Param.1 : n° de module (k 1 à k 16)
          LdDestRel ; Param.2 : instruction (fonction)
        r 777       ; Param.3 : valeur (dans reg. source)

CFB       exec
        k 1         ; Param.1 : n° de module.(k 1 à k 16)
          start     ; Param.2 : instruction (fonction)
          rNotUsed  ; Param.3 : inutilisé

CFB       exec
        k 1         ; Param.1 : n° de module (k 1 à k 16)
          RdActPos  ; Param.2 : instruction (fonction)
        r 1000      ; Param.3 : valeur (dans reg. cible)
```

Il importe de toujours définir ces trois paramètres, même s'il en suffit de deux ; le troisième est alors déclaré inutilisé (« rNotUsed ») ou renvoie à un registre fictif.

Toutes les instructions du bloc Exec sont énumérées en page suivante et étudiées dans le détail en Annexe A.

Instructions du PCD2.H31x (paramètre 2 du bloc Exec) :

N°	Code	Instruction	Détail en page
01	StartMot	Démarrage du déplacement	A-7
02	StopUrg	Arrêt d'urgence du déplacement	A-8
03	Stop	Arrêt du déplacement	A-9
04	MotOff	Désactivation de la régulation PID	A-10
05	RdActPos	Lecture de la position réelle	A-11
06	RdActVel	Lecture de la vitesse réelle	A-12
07	RdIntSum	Lecture de la somme d'intégration	A-13
08	RdIndexRg	Lecture du registre d'index	A-14
09	RdStatRg	Lecture du registre d'état	A-15
10	RdTargPos	Lecture de la position cible	A-16
11	RdTargVel	Lecture de la vitesse cible	A-17
12	GoForw	Avance	A-18
13	GoBackw	Recul	A-19
14	SgStpFor	Avance d'un pas	A-20
15	SgStpBak	Recul d'un pas	A-21
16	LdDestAbs	Chargement de la position cible absolue	A-22
17	LdDestRel	Chargement de la position cible relative	A-23
18	LdVelAbs	Chargement de la vitesse absolue	A-24
19	LdVelRel	Chargement de la vitesse relative	A-25
20	LdAccAbs	Chargement de l'accélération absolue	A-26
21	LdAccRel	Chargement de l'accélération relative	A-27
22	LdPropG	Chargement du gain proportionnel	A-28
23	LdIntG	Chargement du gain intégral	A-29
24	LdDerG	Chargement du gain dérivé	A-30
25	LdSampInt	Chargement du temps d'échantillonnage de la dérivée	A-31
26	LdIntLim	Chargement de la limite d'intégration	A-32
27	ActRegFact	Activation de la régulation PID	A-33
28	LdBrkPtAbs	Chargement d'un point d'arrêt absolu	A-34
29	LdBrkPtRel	Chargement d'un point d'arrêt relatif	A-35
30	ResStatRg	Remise à 0 du registre d'état	A-36
31	SetIdxPos	Définition de la position d'index	A-37
32	SetZero	Définition de la position 0	A-38
33	MotConf	Configuration du déplacement	A-39
34	SetPosTol	Définition de la tolérance de position	A-40

Le numéro de la première colonne représente la valeur absolue (0 à 34) du 2ème paramètre du bloc Exec ; il permet d'identifier la fonction mise en œuvre lorsque l'on visualise le programme utilisateur dans le débogueur.

L'appel du bloc HOME revêt toujours la forme suivante :
(Les valeurs ci-dessous sont données seulement à titre d'exemple.)

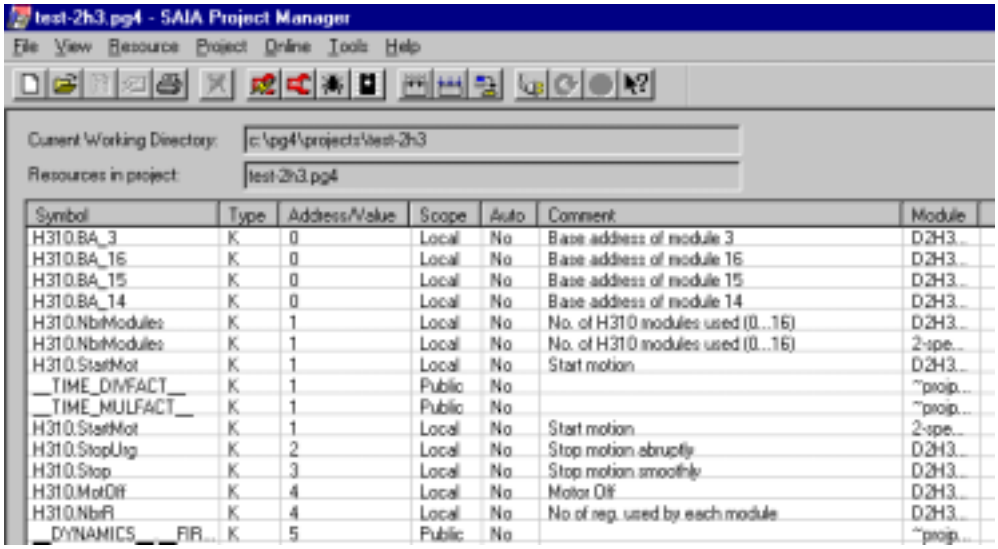
```
CFB       home     ; Initialisation de la position d'origine
        k 1        ; Param.1 : n° de module (k 1 à k 16)
          1        ; Param.2 : sens de la recherche
          0        ; Param.3 : sens du déplacement libre
        r 990      ; Param.4 : vitesse mini
        r 991      ; Param.5 : vitesse maxi
          1000     ; Param.6 : temporisation
        i 7        ; Param.7 : entrée de référence
```

Indicateurs accessibles à l'utilisateur :

Indicateur	**Description**
fHomeErr_x	A l'état haut en cas de prise d'origine incorrecte (échéance tempo, position d'origine non trouvée)
fLS1_x	A l'état haut sur atteinte du fin de course LS1
fLS2_x	A l'état haut sur atteinte du fin de course LS2
fEndHome_x	Toujours à l'état haut, sauf durant la prise d'origine
fBrkPt_x	A l'état haut sur atteinte du point d'arrêt
fOnDest_x	A l'état haut sur atteinte de la position cible
fPosErr_x	A l'état haut, en cas de dépassement de l'erreur de position maximale autorisée
fPar_Err	Erreur de paramétrage (hors bornes)
fTimeout	Problème matériel de lecture/écriture
Ref_1	Entrée de référence

(« _x » correspondant au numéro de module.)

Les adresses effectives de ces indicateurs figurent dans la liste complète et détaillée des ressources, accessible par les commandes « View → Resource List » du Gestionnaire de projet :



Le fichier « project .map » fournit en clair une liste des adresses absolues, à des fins de débogage :

```
SAIA PCD LINKER SP 2.0.83 FILE: test-2h3.pcd
LINKED: 06/30/99 10:57 PAGE 3

RÉSERVÉ Á l'USAGE DE SAIA

SYMBOLE           TYPE VALEUR        DÉFINI     RÉFÉRENCÉ

H310.Exec         FB   1             D2H310_B   2 vitesses
H310.fBrkPt_1     F    7517          D2H310_B
H310.fEndHome_1   F    7516          D2H310_B
H310.fHomeErr_1   F    7513          D2H310_B
H310.Flag_base    F    7502 à 7528   D2H310_B   2 vitesses
H310.fLS1_1       F    7514          D2H310_B
H310.fLS2_1       F    7515          D2H310_B
H310.fOnDest_1    F    7518          D2H310_B
H310.fPar_Err     F    7502          D2H310_B
H310.fPosErr_1    F    7528          D2H310_B
H310.fTimeout     F    7512          D2H310_B
H310.Home         FB   2             D2H310_B
H310.Init         FB   0             D2H310_B   2 vitesses
H310.rDiag        R    3500          D2H310_B
H310.Reg_base     R    3500 à 3524   D2H310_B   2 vitesses
H310.rIniVel_1    R    3524          D2H310_B
H310.rMecFac_1    R    3522          D2H310_B
H310.rSampInt_1   R    3523          D2H310_B

Édition de liens terminée.  0 erreur,  0 avertissement.
```

Affectation des adresses sur le bus

Chaque module occupe 16 adresses en entrée (accessibles en lecture) et 16 adresses en sortie (accessibles en écriture).

Bit n°	Entrées (lecture)	Sorties (écriture)
0	Données du bus (poids faible)	Données du bus (poids faible)
1	Données du bus	Données du bus
2	Données du bus	Données du bus
3	Données du bus	Données du bus
4	Données du bus	Données du bus
5	Données du bus	Données du bus
6	Données du bus	Données du bus
7	Données du bus (poids fort)	Données du bus (poids fort)
8	Alimentation *)	Ecriture (WR LM628)
9	-	Lecture (RD LM628)
10	-	Choix du port (PS LM628)
11	Contact réf. **) / Pw5V ***)	-
12	Version ****)	-
13 - 15	-	-

*) Surveillance du ±15 V (haut : correct ; bas : en défaut).
**) Contact de référence sur H310 (haut : activé ; bas : désactivé).
***) Surveillance du 5 V sur H311 (haut : correct ; bas : en défaut).
****) Version du module (haut : H310 ; bas : H311).

7.2 Programmer en FUPLA avec les boîtes de fonctions « FBoxes »

En préparation

7.3 Programmer en GRAFTEC avec les boîtes de fonctions

En préparation

8. Typologie des erreurs et diagnostic

8.1 Erreurs de définition

Les erreurs de définition du fichier D2H310_B.MBA sont repérées au cours de l'assemblage :

- Nombre de modules (« NbrModules ») inférieur à 1 → pas d'assemblage du code et affichage de l'avertissement suivant dans la fenêtre « Make » :

"Remark : No H310 used (NbrModules = 0 in D2H310_B.MBA)"

- Nombre de modules (« NbrModules ») supérieur à 16 → pas d'assemblage du code et affichage de l'avertissement suivant dans la fenêtre « Make » :

"Error : more than 16 Modules H310 defined (NbrModules = 0...16)"

- Écriture erronée d'une instruction du bloc de fonctions Exec (par ex., « RdIdenti » au lieu de « RdIdent ») → signalement d'une erreur par l'assembleur :

"Symbol not defined 'H310.RdIdenti'"
(l'expression « H310 » étant générée par la directive « $group H310 »)

- Absence de directive assembleur « $group H310 » → signalement d'une erreur par l'assembleur, pour chaque instruction et registre/indicateur utilisés dans le programme :

"Symbol not defined"

8.2 Erreurs de traitement

8.2.1 Paramétrage incorrect

Dans le bloc de fonctions Exec, seul le code instruction (paramètre 2) est vérifié. A l'inverse, les paramètres 1 (n° de module) et 3 (registre source/cible) sont exclus du contrôle pour minimiser le temps d'exécution.

Dans les blocs de fonctions Init et Home, tous les paramètres sont vérifiés pour s'assurer qu'ils sont bien dans la plage autorisée. Si un paramètre est hors bornes, il est forcé à sa valeur minimale, l'indicateur d'erreur « fPar_Err » est positionné et le registre de diagnostic « rDiag » est chargé avec le code d'erreur correspondant.

L'indicateur fPar_Err n'est pas remis à 0 dans les blocs de fonctions, mais dans le bloc XOB 16 ou dans l'étape Init.

Codage de l'erreur :

rDiag bit 31 . . . . . 24 23 . . . . . 16 15 . . . . . 8 7 0
\ Réservés / \ N° bloc / \N° param./ \N° module/
(avec Init = bloc 1,)
Exec = bloc 2,
Home = bloc 3)

Exemple : Si le temps d'échantillonnage (paramètre 6) du bloc Init est hors bornes (c'est-à-dire supérieur à 255), rDiag prend la valeur hexadécimale 00 01 06 02.

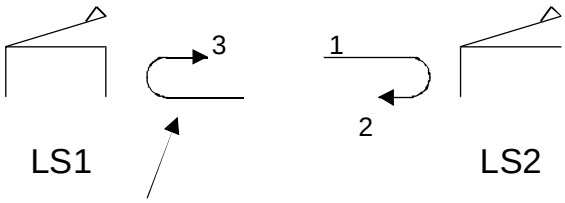
Ce registre de diagnostic étant écrasé à chaque paramètre incorrect, il contient toujours la dernière erreur. Il doit donc être lu dès que l'indicateur fPar_Err signale une erreur de plage. Les adresses absolues de rDiag et fPar_Err sont consultables dans le fichier « project.MAP » (Cf. § 7.1.2, page 7-8). Cela permet, à la mise en service, de localiser une erreur à l'aide du débogueur :

- Exécuter le programme jusqu'à ce que fPar_Err passe à l'état haut,
- Afficher rDiag en hexa,
- Effacer fPar_Err.

8.2.2 Prise d'origine incorrecte

Si l'axe ne rencontre pas de référence (en raison, par exemple, d'un défaut du contact de référence), l'indicateur d'erreur « fHomeErr_x » est positionné, le déplacement s'arrête, la position absolue est mise à 0 et le traitement du bloc Home s'interrompt.

Absence ou défaut de câblage du contact de référence :



- Positionnement de l'indicateur fHomeErr_x
- Mise à 0 de la position absolue

Si le bloc de fonctions Home interrompt son traitement sur expiration de sa temporisation (paramètre 6), le registre de diagnostic rDiag est chargé avec le code 6.

L'indicateur fHomeErr_x est défini pour chaque module (_x représentant le n° de module) et remis à 0 au lancement du bloc Home. Il doit être interrogé à chaque appel du bloc Home pour s'assurer que l'axe est correctement référencé.

Exemple :

```
CFB     Home            ; retour à l'origine de l'axe 3
      k 3               ;  n° de module
        0               ;  sens de la recherche
      r 1010            ;  vitesse mini
      r 1011            ;  vitesse maxi
        50              ;  tempo
      i 64              ;  entrée de référence

STH     fHomeErr_3      ; Interrogation de l'indicateur
                        ;  d'erreur de l'axe 3
CFB   h Errorhandl      ; Appel de blocs (spécifiques à
                        ;  l'utilisateur), si fHomeErr_3 est
                        ;  haut

CFB      Exec           ; Déplacement 1
```

Notes personnelles :

9. Installation et mise en service

9.1 Introduction

Ce chapitre décrit les différentes étapes de mise en service d'un servo-entraînement piloté par PCD2.H31x. Pour garantir une exploitation « zéro défaut » du module H31x, il importe de procéder à cette mise en service dans une même séquence.

Critères de choix du servo-entraînement couplé au PCD2.H31x

Un système d'entraînement est toujours constitué des éléments suivants :

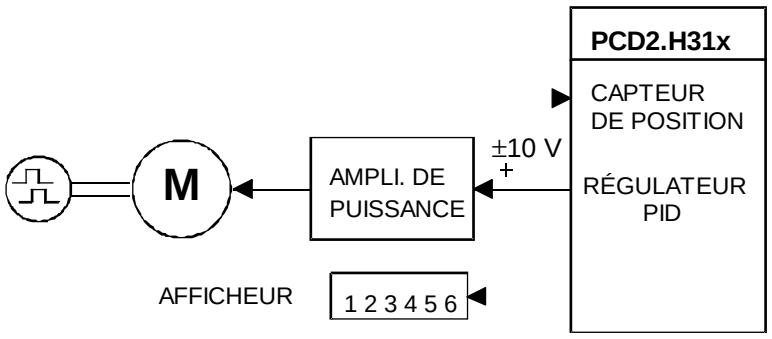
- Une électronique de commande pour programmer le mouvement (position, vitesse et accélération) et réguler le positionnement ; c'est le rôle du PCD1/PCD2 équipé du module H3.
- Un servo-amplificateur pour piloter le servomoteur.
- Un servomoteur assurant la conversion des signaux électriques en énergie mécanique.
- Un capteur de position, sous forme de codeur incrémental.
- Un entraînement mécanique.

Choix du servo-amplificateur et du servomoteur :

Quel que soit le servo-entraînement CC ou CA mis en œuvre, il convient de respecter tout particulièrement les points suivants :

- Amplificateur et moteur doivent être parfaitement adaptés (puissance, tension et intensité).
- Pour garantir la précision de la régulation de position et de vitesse, il faut avoir recours à un étage de puissance à quatre quadrants avec régulateur de vitesse intégré.
- La vitesse de l'amplificateur et du moteur doit couvrir la plus grande plage de régulation possible de façon à pouvoir appliquer le couple nécessaire, même à basses vitesses.
- Le H3 délivre un signal analogique ±10 V donnant la consigne de vitesse.
- Pour enregistrer la position, le H3 nécessite un codeur incrémental fournissant au moins deux signaux rectangulaires en quadrature de phase.

Synoptique d'un système d'entraînement avec PCD2.H31x



9.2. Contrôle de l'installation et du câblage

L'installation du PCD2 impose de respecter quelques règles et précautions ; la mise sous tension de l'appareil doit obligatoirement être précédée d'un contrôle visuel de l'installation et du câblage :

- Le PCD1/PCD2 a-t-il été endommagé au cours du transport ou du montage ?
- Le module H3 est-il enfiché dans l'emplacement prévu sur le bus PCD1/PCD2 ? Est-il correctement câblé sur le module de bus correspondant ?
- **Câblage et raccordement de l'alimentation**
  Vérifiez que la tension d'alimentation du H3 (bornes « + » et « – ») est de type lissée +24 VCC (19 à 32 V, ondulation maxi 10 %).
- **Mise à la terre**
  Une exploitation sans défaut exige une mise à la terre parfaite pour éliminer les tensions parasites externes : le PCD1/PCD2 doit être relié à la borne « GND » (24 VCC –) par un conducteur de grosse section sur le profilé de masse de l'armoire électrique. Il convient en outre de s'assurer que toutes les lignes de terre sont exemptes de boucles.
- **Pose des câbles**
  La réglementation en vigueur exige de dissocier câbles haute tension et câbles de commande.
- **Raccordement de la sortie régulation (±10 V analogique)**
  Vérifiez les raccordements décrits au § 5.5. Á noter que le câble doit être blindé.
- Le signal de validation **« Enable »** de la commande est déclenché par une sortie TOR standard du PCD1/PCD2.
- **Raccordement du codeur**
  Le codeur 5 V réclame une attention particulière :
  - Le câble doit être blindé et les lignes constituées de paires torsadées.
  - Longueur de câble maxi : 20 m ; section de conducteur mini : 0,25 mm².

    De plus, vérifiez le montage correct du codeur (serrage de l'accouplement), son type et ses caractéristiques (nombre d'impulsions/tour).
- Raccordez directement le contact de référence **Ref** (H310).
- Raccordez les fins de course **LS1/LS2** et le contact de référence **Ref** (H311) aux entrées TOR standards du PCD1/PCD2.

9.3 Mise en service de l'entraînement sans module de positionnement

La première étape consiste à mettre en service l'entraînement seul (étage de puissance + moteur), sans la commande d'axe.

Au préalable, quelques précautions s'imposent :

- Déconnecter la sortie régulation PID ±10 V du H31x (bornes 6 « Out » et « – ») de l'ampli de puissance.
- Mettre hors tension le PCD1/PCD2 et le H31x. Si cela est impossible, exécutez d'abord l'étape du § 9.4.1 (mise sous tension des alimentations).
- Régler les fins de course d'arrêt d'urgence de l'entraînement pour éviter tout risque d'accident en cas de déplacement incontrôlé de l'axe.

Vous pouvez maintenant procéder à la mise en service de l'étage de puissance et du moteur en suivant les consignes du fabricant.

9.4 Mise en service de l'entraînement avec module de positionnement

9.4.1 Mise sous tension des alimentations

Cette étape peut aussi s'effectuer avant la mise en service de l'étage de puissance et du moteur. Le cas échéant, veillez impérativement à ce que l'axe ne puisse pas exécuter de mouvement incontrôlé (étage de puissance hors tension, sortie régulation coupée).

A la première mise sous tension du PCD1/PCD2 et du H31x (l'appareil de programmation étant déconnecté), repérez les différents voyants de contrôle et leur signification.

Sur le PCD1 :

Désignation	Couleur	Signification
24 VDC	jaune	Voyant allumé → Présence tension d'alimentation
Run	jaune	Voyant éteint → L'UC n'est pas en mode « Run » (aucun programme n'ayant été chargé).
Error	jaune	Voyant éteint

Sur le PCD2 :

Désignation	Couleur	Signification
24 VDC	jaune	Voyant allumé → Présence tension d'alimentation
Battery	rouge	Voyant éteint → Absence de défaut batterie
Watch Dog	jaune	Voyant éteint → Chien de garde inactif (aucun programme n'ayant été chargé).
Run	jaune	Voyant éteint → L'UC n'est pas en mode « Run » (aucun programme n'ayant été chargé).
Halt	rouge	Voyant allumé → UC à l'arrêt (en l'absence de programme)
Error	jaune	Voyant éteint

Sur le PCD2.H310 :

Désignation	Couleur	Signification
Power	jaune	Voyant allumé → Présence du ±15 V
Ref	jaune	Contact de référence

Sur le PCD2.H311 :

Désignation	Couleur	Signification
Power	jaune	Voyant allumé → Présence du ±15 V
Pw 5V	jaune	Voyant allumé → Alimentation 5 V du codeur

9.4.2 Édition d'un programme utilisateur simple

Pour mettre en service un entraînement équipé du H31x, il faut charger un programme utilisateur élémentaire dans le PCD1/PCD2 de façon à effectuer tous les contrôles et réglages indispensables : le petit programme décrit au § 6.1.3 peut servir aux premiers essais.

Par ailleurs, un outil de mise en service complet, intégrant fins de course et contact de référence, est étudié au chapitre 10.

Six étapes sont nécessaires à l'élaboration de ce programme utilisateur :

- Installation des fichiers de la disquette PCD9.H31E (Cf. § 7.1.1).
- Ouverture d'un nouveau projet.
- Copie dans le fichier D2H310_B.MBA.
- Saisie du nombre de H31x et de leur adresse de base dans le fichier D2H310_B.MBA.
- Définition des caractéristiques mécaniques de la machine.
- Paramétrage (bloc de fonctions Init et déplacement aller-retour) et chargement du programme de mise en service.

Lecture de l'indicateur d'erreur de paramétrage « fPar_Err »

Il est vivement conseillé d'interroger cet indicateur pour effectuer la mise en service. Précisons qu'il n'est positionné qu'une fois, quel que soit le nombre d'axes mis en œuvre.

Il est ensuite possible de consulter le registre de diagnostic « rDiag » du PCD pour connaître l'origine de l'erreur (Cf. § 8.2.1).

Il appartient à l'utilisateur de remettre l'indicateur fPar_Err à 0.

9.4.3 Définition des caractéristiques techniques de la machine

Avant la première mise en service, il convient de définir les paramètres du bloc Init ; certaines valeurs sont imposées par la machine, d'autres sont préconisées à titre d'essai.

Tolérance de position → paramètre 7 du bloc Init, Cf. pages A-1/2

Valeur conseillée : 1 tour codeur

Pour un codeur de 500 imp./tour, ce paramètre vaut 2000, soit 4 ∗ 500 imp./tour (prise en compte des fronts d'impulsion du codeur).

Régulation PID → paramètres 2 à 6 du bloc Init, Cf. pages A-1/2

Valeurs conseillées :

Proportionnelle (paramètre 2) :	10
Intégrale (paramètre 3) :	0
Dérivée (paramètre 4) :	0
Limite d'intégration (paramètre 5) :	30000
Tps d'échantillonnage dérivée (paramètre 6) :	15 (= 5,61 ms)

Données mécaniques → paramètre 9 du bloc Init (en calcul flottant)

Ce paramètre est dicté par la résolution du codeur, le pas de la vis et les réducteurs.

> Important : l'unité de mesure choisie ici (m, cm, mm, 1/10 mm, 1/100 mm, µm) doit également s'utiliser pour la vitesse (paramètre 10), l'accélération (paramètre 11) et toutes les valeurs concernant le déplacement, le point d'arrêt, la vitesse et l'accélération, tout au long du programme utilisateur de cet axe.

Supposons $k = \frac{4 \times In}{s}$ [impulsions/unité de mesure]

Avec In : nb d'impulsions/tour (résolution du codeur)

s : course/tour (pas de la vis et réducteurs)

Exemple : In = 1000 impulsions/tour
s = 2 mm (pas de la vis, sans réducteur)

$$k = \frac{4 \times 1000}{2} = 2000 \text{ impulsions/mm} = 2 \text{ impulsions/}\mu\text{m}$$

Si l'unité de mesure retenue est 1 mm, le paramètre 9 du bloc Init doit valoir 2000,0 (en virgule flottante). De même, à une unité de 1 µm correspond la valeur flottante 2,0.

Vitesse → paramètre 10 du bloc Init

Á la première mise en service, il est conseillé d'adopter une vitesse basse. Sa valeur doit être saisie au format entier, dans la même unité de mesure que celle des données mécaniques (paramètre 9).

Valeur conseillée : 0,05 m/s soit 50 mm/s ou 50000 µm/s

Assurez-vous que l'entraînement mis en œuvre est effectivement capable d'atteindre cette vitesse ; sinon, le régulateur détecte en permanence des écarts par rapport à la vitesse cible, ce qui aboutit à un cumul d'erreurs provoquant une rampe de freinage incorrecte ou un arrêt brusque du déplacement.

Accélération → paramètre 11 du bloc Init

Á la première mise en service, il est conseillé de choisir une accélération faible. Sa valeur doit être saisie au format entier, dans la même unité de mesure que celle des données mécaniques (paramètre 9).

Valeur conseillée : 0,01 m/s^2 soit 10 mm/s^2 ou 10000 µm/s^2

Avant de lancer le mouvement, vérifiez le fonctionnement du codeur en déplaçant l'axe manuellement, l'amplificateur étant hors tension. Lisez aussi le § 9.4.5.

9.4.4 Contrôle du sens de rotation et du calcul de déplacement (codeur)

Ce contrôle impose quelques précautions :

- Déconnecter la sortie régulation ±10 V du H31x (bornes 6 et masse) de l'étage de puissance.
- Mettre le PCD1/PCD2 sous tension, le programme de mise en service étant chargé : l'automate est opérationnel (mode « Run »).
- Raccorder l'appareil de programmation et démarrer le logiciel de mise en service. Effectuer tous les paramétrages du menu « Configure », conformément aux définitions du fichier D2H310_B.MBA.

  Les essais qui suivent s'accompagnent d'un affichage de la position réelle.

Contrôle du sens de rotation :

1) Faites tourner l'arbre dans le sens positif : la position réelle augmente.

2) Faites tourner l'arbre dans le sens négatif : la position réelle diminue.

Si possible, la rotation de l'arbre peut être manuelle. Á défaut, appliquez une consigne (source de tension de ±0.5 à 10 V) à l'étage de puissance.

Définition des sens positif et négatif :

- Il y a déplacement dans le sens positif lorsqu'on applique une tension de consigne positive (0 à +10 V) à l'étage de puissance.
- Il y a déplacement dans le sens négatif lorsqu'on applique une tension de consigne négative (0 à –10 V) à l'étage de puissance.
- En cas de comportement inverse, il faut intervertir les deux signaux de phase A et B du codeur.

Contrôle du calcul de déplacement :

Effectuez une rotation d'un tour et regardez l'affichage de la position réelle : celui-ci doit indiquer une valeur de déplacement correspondant aux données mécaniques « k » (paramètre 9 du bloc Init). En cas d'erreur, effectuez un nouveau contrôle du calcul et de la saisie du paramètre k.

Contrôle du signal d'index IN :

Ce contrôle n'a d'utilité que si le signal d'index (encore appelé « top 0 ») est traité par le codeur pour donner la position 0.

Méthode :

(Toutes les actions ci-après sont effectuées manuellement, à l'aide de commandes du programme de mise en service.)

1) Regardez l'affichage du registre de position d'index : on peut y lire une valeur quelconque.
2) Exécutez l'instruction « SetIdxPos » ; lorsque l'impulsion d'index suivante est enregistrée, la position réelle est écrite dans le registre de position d'index.
3) Tournez l'axe jusqu'à ce que la position réelle soit écrite dans le registre de position d'index. Notez qu'il ne doit y avoir qu'une écriture de position par tour codeur.
4) Exécutez de nouveau l'instruction « SetIdxPos » et vérifiez qu'un tour codeur n'engendre qu'une écriture de la position en cours dans le registre.

Si vous obtenez un autre résultat, cela peut être dû :

- à un défaut codeur,
- aux signaux codeur A, B et IN, qui ne sont pas dans l'ordre voulu à l'entrée du capteur de position du H31x.

Si vous ignorez l'ordre de ces signaux, utilisez un oscilloscope pour le déterminer.

9.4.5 Vérification et réglage du régulateur PID

Conditions préalables :

- Le régulateur étant hors tension, fermer la boucle de régulation en reconnectant le câble de consigne de l'étage de puissance aux bornes 6 et masse du H31x.
- Mettre le régulateur sous tension.
  Attention ! activez l'arrêt d'urgence pour éviter tout déplacement incontrôlé de l'axe.
- Raccorder l'appareil de programmation et démarrer le logiciel de mise en service.

Méthode :

1) Chargez une position cible 0 et exécutez l'instruction de démarrage : le régulateur du H31x est activé et maintient l'axe en position.

2) Chargez une position cible située dans la plage de déplacement autorisée et démarrez le mouvement : l'axe se déplace vers cette position cible.

Comportements incorrects :

- L'axe se déplace à la vitesse maxi.

Explication	• Le circuit de régulation de position (H31x) ou le circuit de régulation de vitesse (étage de puissance) est mal polarisé. Résultat : lorsque la tension de consigne est positive (= position cible positive), le déplacement s'effectue dans un sens négatif. • La sortie analogique du H31x est en défaut : elle fournit une tension constante maxi d'environ +12 V ou –12 V à la sortie du régulateur.
Solution	• Vérifiez le sens de rotation, recommencez les contrôles du § 9.4.4 et modifiez la polarisation en conséquence. • Remplacez le module H31x.

- L'axe se déplace toujours à basse vitesse (dérive).

Explication	• La consigne n'atteint pas l'étage de puissance ; la régulation n'est pas activée sur le H31x.
Solution	• Vérifiez le câblage. • Assurez-vous qu'après l'instruction de démarrage « StartMot », le générateur de profil du H31x fournit bien une vitesse cible (instruction « RdTargVel »), ce qui signifie que le régulateur fonctionne correctement.

- L'axe démarre, effectue un bref déplacement, puis s'arrête.

Explication	• La fonction de surveillance de l'erreur de position est intervenue pour arrêter l'entraînement : le moteur est incapable de suivre le mouvement qui lui est imposé.
Solution	• Remédiez aux éventuels problèmes mécaniques. • Réglez vitesse et accélération. • Augmentez la valeur maxi de l'erreur de position.

- L'axe se déplace par à-coups vers une position cible incorrecte.

Explication	• La liaison mécanique codeur-moteur présente des défauts (desserrage de l'accouplement).
Solution	• Corrigez les éventuels problèmes mécaniques.

Á l'approche de la position cible, vous constaterez la persistance d'une erreur de position. Cela s'explique par le fait que dans une régulation Proportionnelle pure (les actions Intégrale et Dérivée étant nulles), la mesure ne rejoint jamais la consigne : il subsiste toujours un écart de réglage résiduel.

L'étape suivante consiste donc à ajuster les coefficients I et D pour obtenir une régulation optimale offrant le meilleur compromis précision-stabilité.

Les réglages présentés dans les pages suivantes sont le fruit de l'expérience issue de l'étude et de l'expérimentation d'applications concrètes.

Réglage du gain proportionnel « kp » :

1) Optez pour un gain proportionnel de faible valeur (l'expérience recommande 10). Les fonctions Intégrale et Dérivée étant nulles, vous n'utilisez que la régulation Proportionnelle.
2) Déplacez l'axe lentement (approchez la position cible ou utilisez les fonctions « GoForw » et « GoBackw »).
3) Augmentez progressivement kp jusqu'à ce que le circuit de régulation commence à osciller, puis diminuez kp d'environ 30 % et chargez-le ; vous obtenez le réglage de l'action Proportionnelle, qui ne doit pas être modifié pour l'instant.

Réglage de l'intégrale « ki » :

Augmentez progressivement ki jusqu'à obtenir le temps d'établissement voulu pour l'erreur de position : vous réglez ainsi l'action Intégrale, qui ne doit pas être modifiée pour l'instant.

Si, en approchant la cible, vous observez un dépassement excessif de la consigne, plusieurs causes sont envisageables :

- La valeur choisie pour l'accélération ou le freinage est trop élevée ; en conséquence, le moteur ne peut pas suivre la consigne et il vous faut diminuer l'accélération.
- Le même phénomène peut également se produire si la vitesse choisie est trop élevée.
- La valeur de ki est trop élevée.
- L'expérience montre qu'une augmentation de l'intégrale associée à une dérivée nulle accroît la tendance du régulateur à osciller : il faut donc augmenter la dérivée parallèlement à l'intégrale pour réduire le dépassement.

La lecture de la somme d'intégration (instruction « RdIntSum ») est un bon moyen de surveiller ce comportement : si celle-ci atteint une valeur élevée en cours de mouvement, il y a risque de dépassement.

Solution : - Diminuez ces paramètres.
- Limitez la somme d'intégration (instruction « LdIntLim »).

Réglage de la dérivée « kd » :

Augmentez progressivement kd jusqu'à obtenir le dépassement ou le temps d'établissement voulu pour l'erreur de position : vous réglez ainsi l'action Dérivée du régulateur.

Paramétrage du temps d'échantillonnage (instruction « LdSampInt ») de la dérivée :

Á basses vitesses, choisissez un temps d'échantillonnage plutôt élevé. L'expérience prouve qu'il varie généralement de 2 ms à 9 ms, ce qui donne les valeurs 5 et 25 dans le registre de l'instruction « LdSampInt ».

Optimisation des réglages :

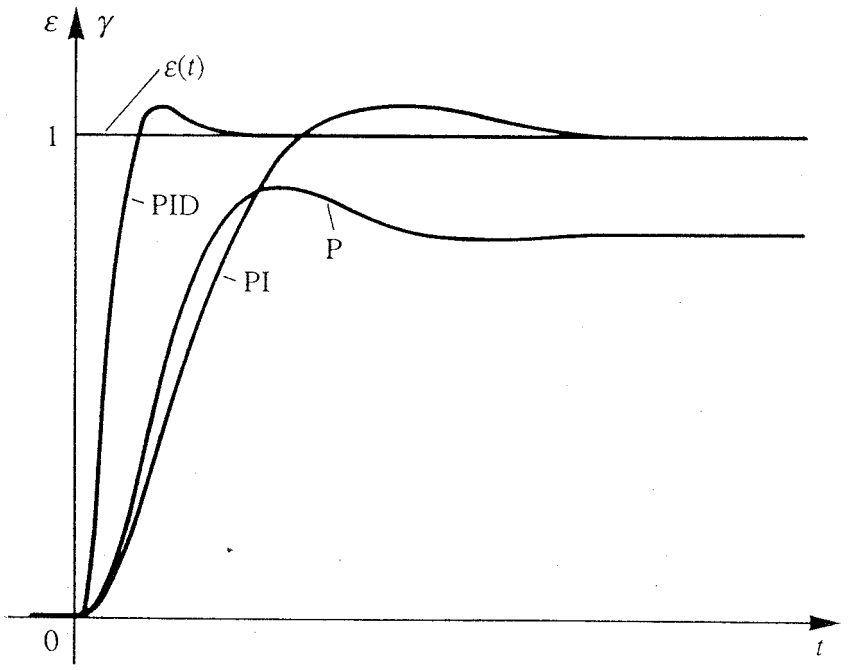
Si, au terme de cette démarche, la régulation ne donne pas satisfaction, reprenez et affinez le dosage de chaque coefficient P, I, D jusqu'à obtenir un réglage optimal.

9.4.6 Effet des réglages PID sur le comportement du régulateur

L'augmentation du **gain proportionnel kp** accroît les risques de dépassement du régulateur, mais réduit également l'écart de réglage en régime permanent tout en améliorant la robustesse du régulateur.

L'augmentation de l'**intégrale ki** accroît la tendance du régulateur au dépassement, mais accélère en parallèle l'établissement de l'erreur de position.

Un bon réglage de la **dérivée kd** a un effet stabilisateur et garantit un dépassement et un temps d'établissement moindres. À l'inverse, un dosage trop important de kd peut compromettre la stabilité du système.



9.4.7 Programme de mise en service

Un programme graphique de mise en service sous FUPLA, d'utilisation facile et conviviale, est étudié au chapitre 10.

Voici un exemple extrêmement simple de déplacement aller et retour avec affichage de la position réelle, sur un axe. Son paramétrage reprend les valeurs préconisées dans les paragraphes précédents.
(Pour plus d'informations, reportez-vous au § 6.1.1).

```
$include D2H310_B.equ
$group H310

            xob     16

            LD    R 1000
                    2.0         ;  Données mécaniques
            LD    R 1001
                    50000       ;  Vitesse absolue initiale
            LD    R 1002
                    10000       ;  Accélération absolue initiale

            CFB     Init        ; Initialisation du module
                  K 1           ;  Numéro de module
                    10          ;  Proportionnelle (régulateur PID)
                    0           ;  Intégrale (régulateur PID)
                    0           ;  Dérivée (régulateur PID)
                    30000       ;  Limite d'intégration
                    15          ;  Temps d'échantillonnage de la dérivée
                    2000        ;  Tolérance de position
                    1           ;  Comportement en cas d'erreur de pos.
                  R 1000        ;  Registre des données mécaniques
                  R 1001        ;  Registre de la vitesse initiale
                  R 1002        ;  Registre de l'accélération initiale

            exob
;           ------------------------------------------------------
;
            cob     0
                    0

start:      sth   i 0           ; Démarrage OK ?

            CFB     Exec        ; Exécution du bloc de fonctions
                  K 1           ;  N° de module
                    RdActPos    ;  Instruction: lecture position réelle
                  R 90          ;  Registre de résultat
            DSP   R 90          ; Affichage du contenu du registre

            jr    l start       ; Démarrage incorrect, attente

            LD    R 100         ; Chargement de la valeur
                    10000       ;  absolue de la position cible

            CFB     Exec        ; Exécution du bloc de fonctions
                  K 1           ;  N° de module

                    LdDestAbs   ;  Instruction: Charg. pos. cible abs.
                  R 100         ;  Registre de la position cible absolue
```

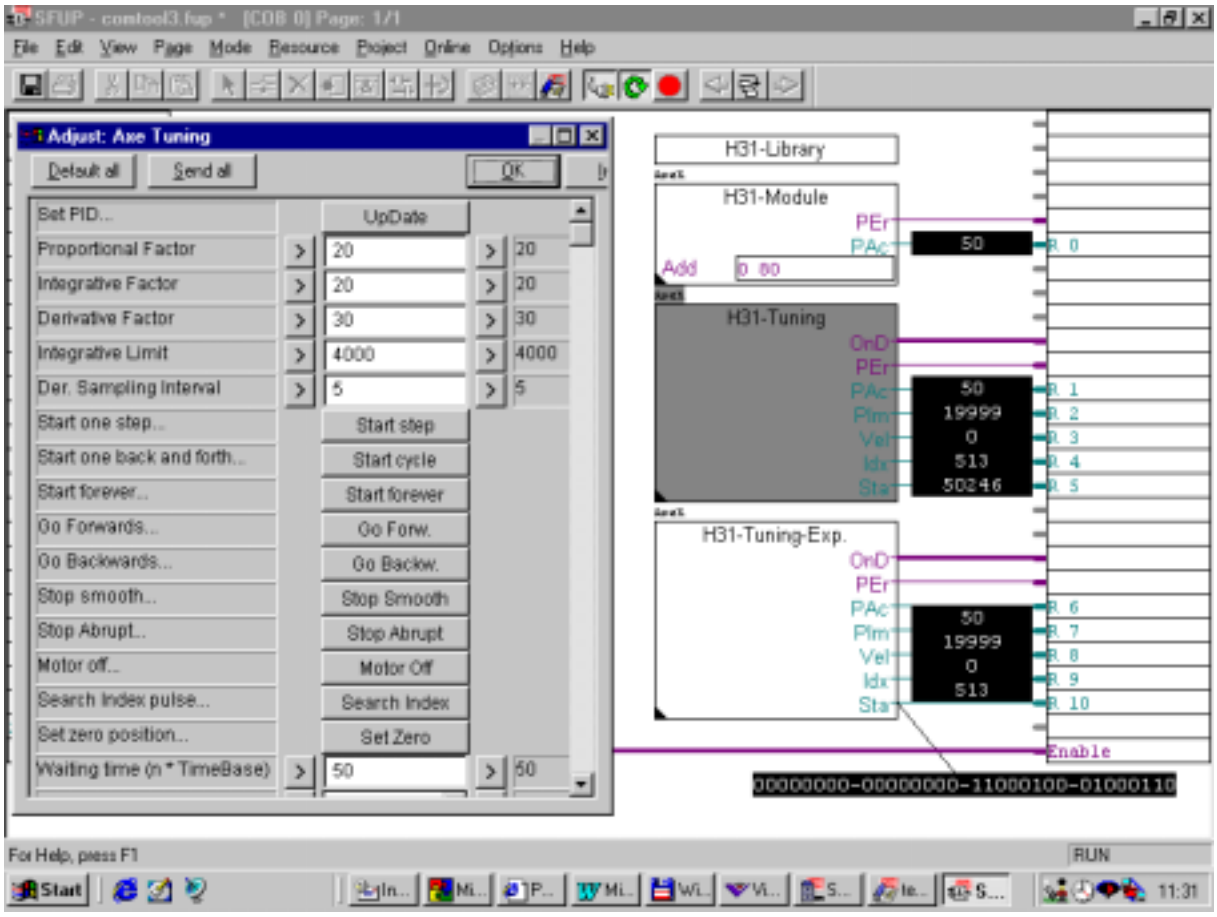
```
pos1:       CFB     Exec        ; Exécution du bloc de fonctions
                  K 1           ;  N° de module
                    StartMot    ;  Instruction: démarrage déplacement
                    rNotUsed    ;  Registre fictif

            CFB     Exec        ; Exécution du bloc de fonctions
                  K 1           ;  N° de module
                    RdActPos    ;  Instruction: lecture position réelle
                  R 90          ;  Registre de la position réelle
            DSP   R 90          ; Affichage du contenu du registre

            CFB     Exec        ; Exécution du bloc de fonctions
                  K 1           ;  N° de module
                    RdStatRg    ;  Instruction: lecture registre d'état
                  R 0           ;  Valeur du registre d'état

            STH     fOnDest_1   ; Position atteinte ?
            jr    l pos1        ; dans la négative, attente

            ld    t 0
                    50
pause1:     sth   t 0
            jr    h pause1

            LD    R 100         ; Chargement de la valeur
                    0           ;  absolue de la position cible

            CFB     Exec        ; Exécution du bloc de fonctions
                  K 1           ;  N° de module
                    LdDestAbs   ;  Instruction: charg. pos. cible abs.
                  R 100         ;  Registre de la position cible absolue

            CFB     Exec        ; Exécution du bloc de fonctions
                  K 1           ;  N° de module
                    StartMot    ;  Instruction: démarrage déplacement
                    rNotUsed    ;  Registre fictif

pos2:       CFB     Exec        ; Exécution du bloc de fonctions
                  K 1           ;  N° de module
                    RdActPos    ;  Instruction: lecture position réelle
                  R 90          ;  Registre de la position réelle
            DSP   R 90          ; Affichage du contenu du registre

            CFB     Exec        ; Exécution du bloc de fonctions
                  K 1           ;  N° de module
                    RdStatRg    ;  Instruction: lecture registre d'état
                  R 0           ;  Valeur du registre d'état

            STH     fOnDest_1   ; Position atteinte ?
            jr    l pos2        ; dans la négative, attente

            ld    t 0
                    50
pause2:     sth   t 0
            jr    h pause2

            ecob

$endgroup
```

10. Outil de mise en service

La disquette PCD8.H31 contient un outil de programmation, « comtool-fup », destiné à la mise en service du PCD2.H31x. Basé sur le logiciel FUPLA du PG4 (version V2.0.70 et ultérieure), il offre un bon confort d'utilisation avec 4 boîtes de fonctions :

1) Une boîte d'organisation (« H31-Library »),
2) Une boîte d'initialisation du PCD2.H31x (« H31-Module »),
3) Une boîte de paramétrage d'un programme simple de mouvement (« H31-Tuning ») permettant de modifier la totalité des réglages PID (pour optimiser le comportement du régulateur) et des paramètres du déplacement,
4) Une boîte de création et de modification en ligne d'un programme de mouvement, constitué de 4 séquences au choix (« H31-Tuning-Exp. »).

Une fois optimisés, les réglages PID sont transférés dans le programme utilisateur définitif (bloc de fonctions Init et, éventuellement, un bloc Exec supplémentaire pour régler chaque paramètre PID).



Écran de l'outil de mise en service : fenêtre de paramétrage, boîte de fonctions et programme de mouvement simple

Les paramètres PID sont définis un à un, transférés individuellement dans le programme utilisateur, puis tous activés en cliquant sur le bouton « UpDate ».

Les trois boutons « Start step », « Start cycle » et « Start forever » permettent de lancer respectivement :

- un déplacement unique,
- un aller-retour,
- un aller-retour continu.

Le paramètre « Waiting time » définit une pause après chaque séquence du programme.

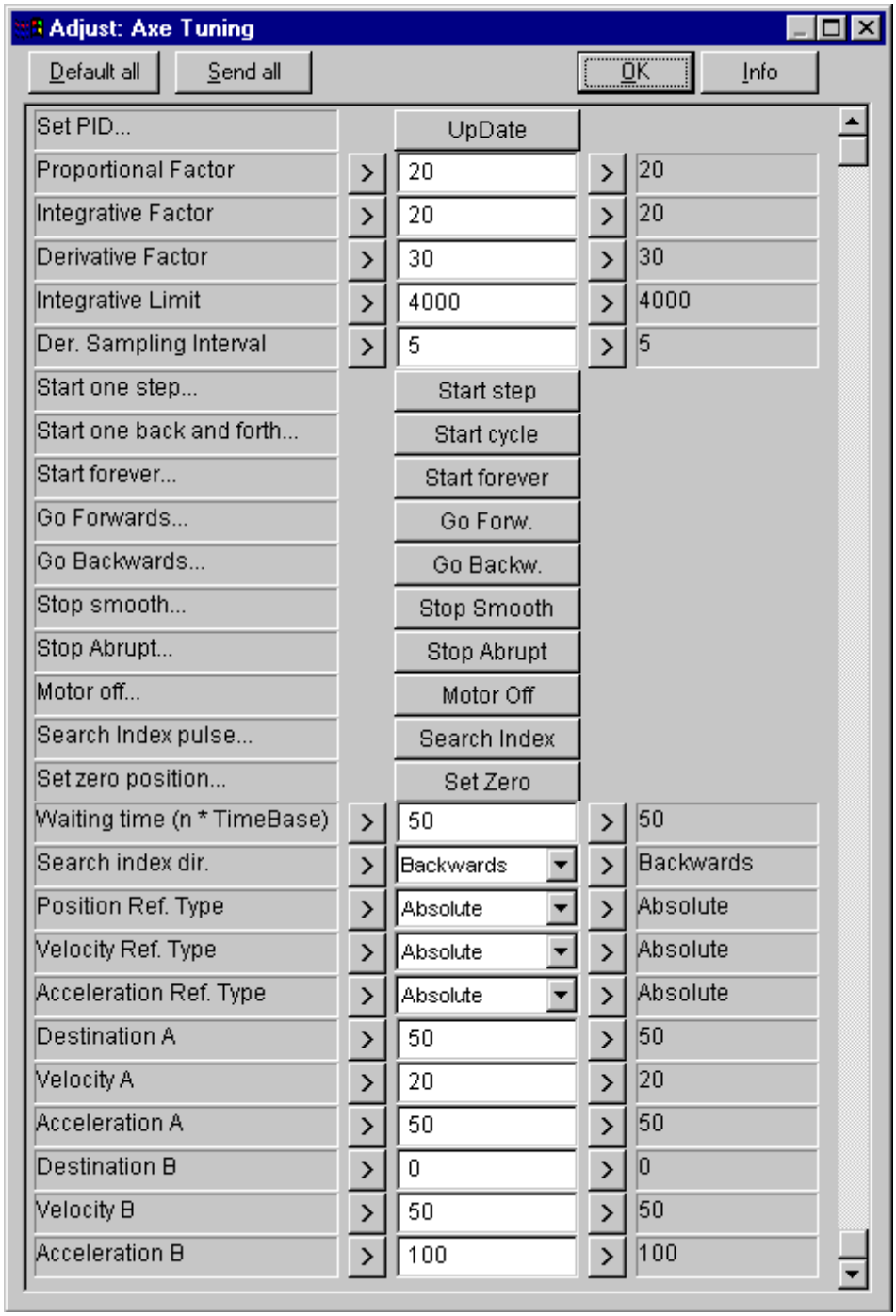
Quatre autres boutons déclenchent :

- une avance (« Go Forw. ») ou un recul (« Go Backw. »),
- un arrêt en douceur (« Stop Smooth ») ou brutal (« Stop Abrupt »).

Le bouton « Set Zero » permet de déclarer n'importe quelle position comme étant la nouvelle position 0.

Enfin, il est aussi possible de désactiver la commande moteur (« Motor Off ») ou de chercher la position d'index suivante (« Search Index »).

Fenêtre de paramétrage de la boîte de fonctions décrivant un simple aller et retour du chariot (grandeur réelle)

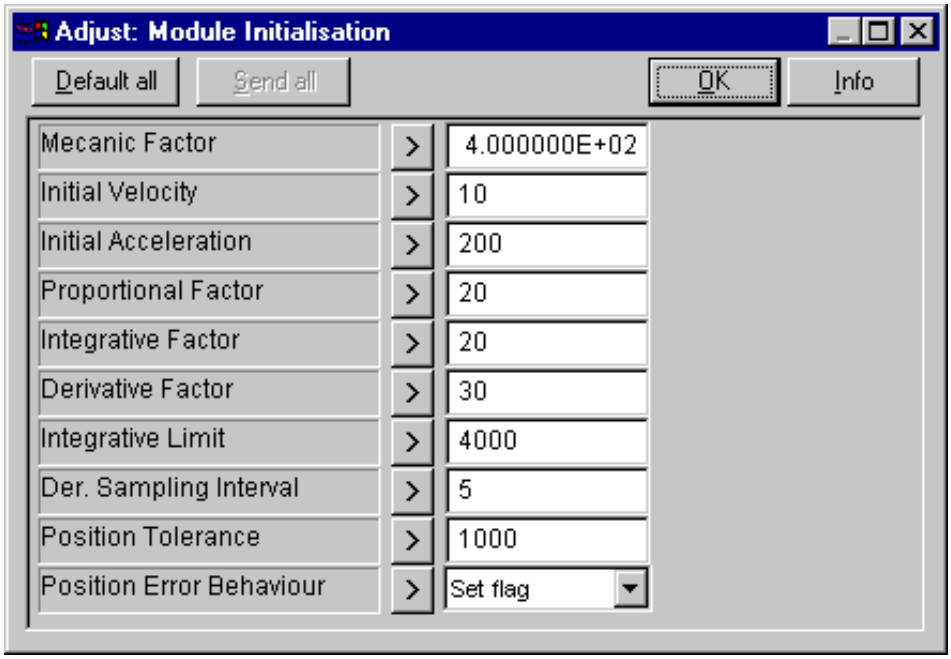


Comme sur l'écran précédent, les paramètres PID sont définis un à un, puis transférés individuellement dans le programme utilisateur avant d'être tous activés par « UpDate ».

Les paramètres du bas de l'écran, relatifs au programme de déplacement, sont transférés immédiatement après validation.

Fenêtre de paramétrage de la boîte d'initialisation du module

Ces paramètres ne sont transférés qu'après un téléchargement. En conséquence, le paramétrage doit avoir lieu **avant** compilation et **avant** téléchargement dans le PCD.

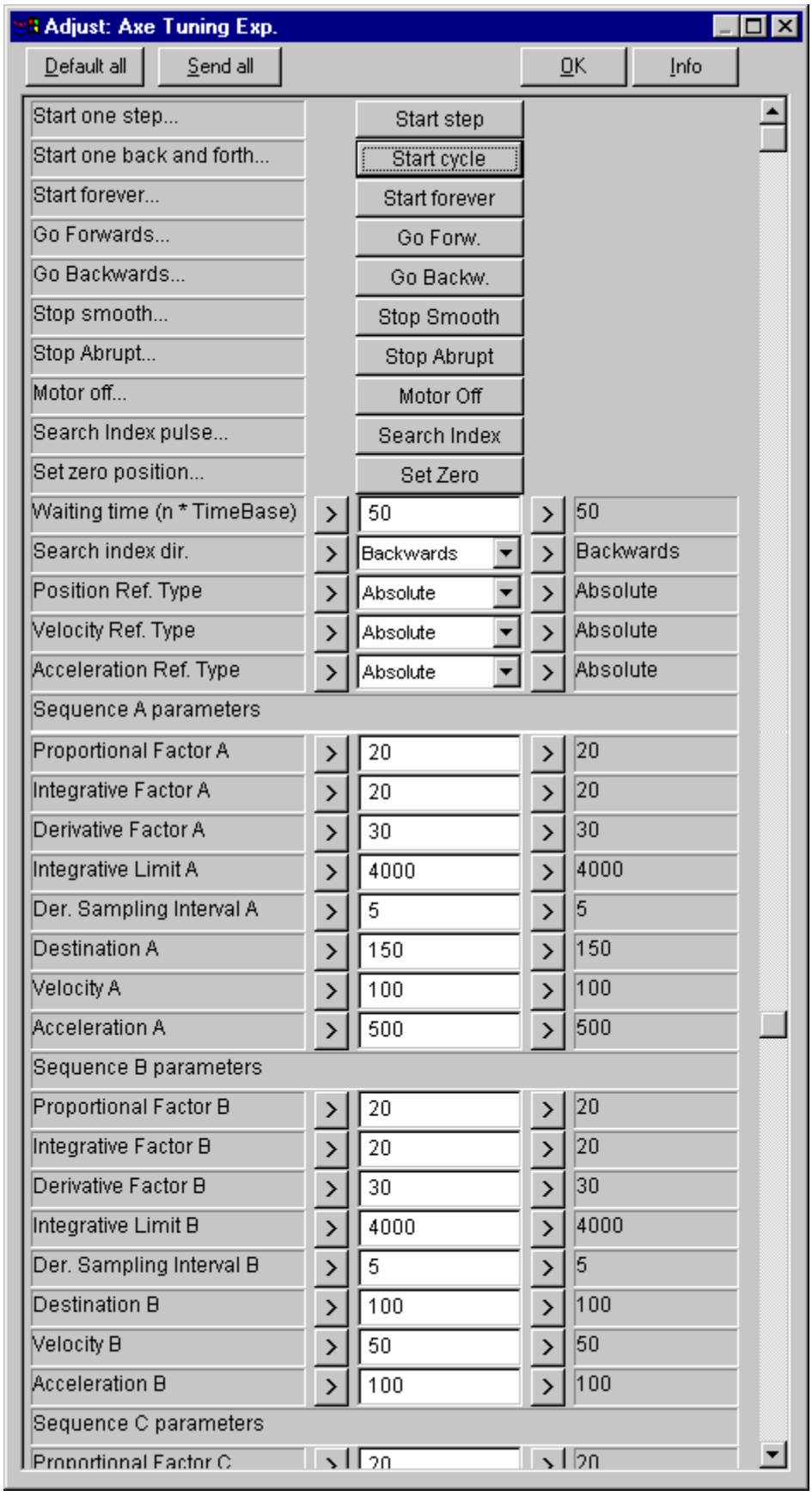


La 4ème boîte de fonctions permet de créer un programme de mouvement en 4 séquences au choix, chacune d'elle possédant des paramètres réglables et modifiables individuellement en ligne.

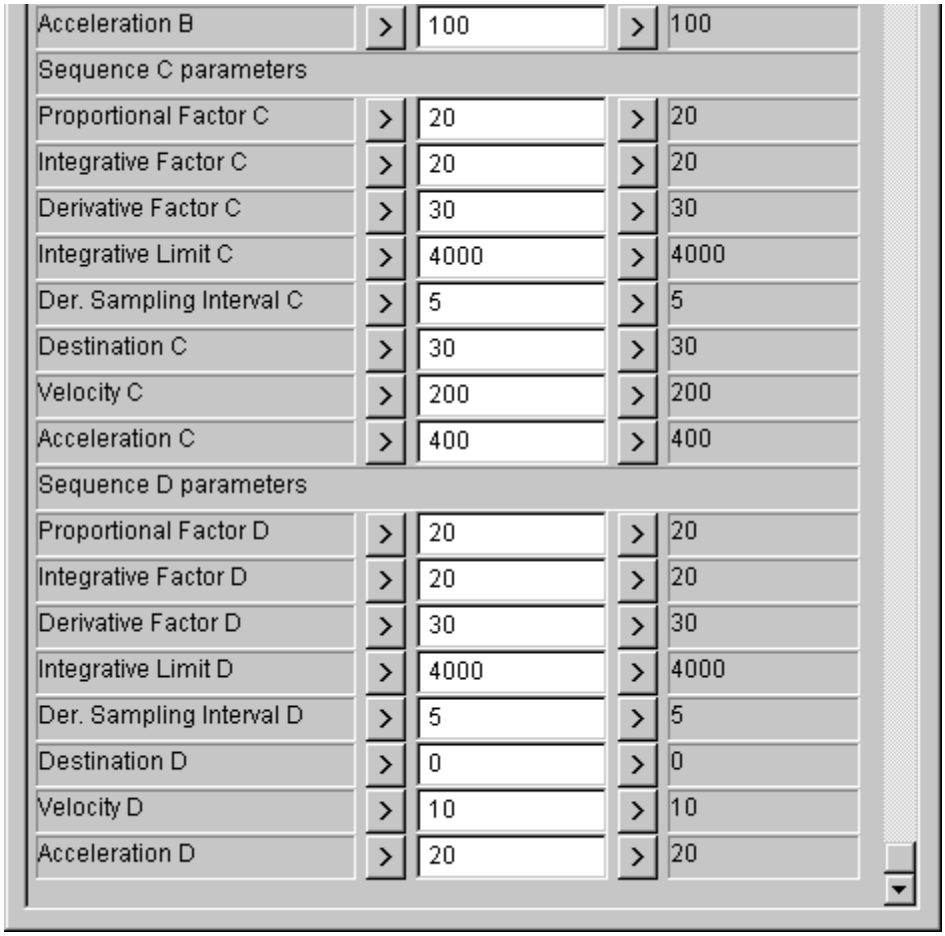
Le principe est le même que celui du programme simple d'aller-retour :

- « Start step » lance chaque séquence l'une après l'autre,
- « Start cycle » démarre une seule séquence complète,
- « Start forever » déroule les 4 séquences indéfiniment jusqu'à interruption par une commande « Stop ... » ou « Motor Off »,
- « Waiting time » définit une pause à l'issue de chaque séquence,
- « Go Forw. » et « Go Backw. » permettent d'avancer ou de reculer,
- « Stop Smooth » et « Stop Abrupt » arrêtent le déplacement en douceur ou brusquement,
- « Set Zero » déclare une position quelconque comme étant la nouvelle position 0,
- « Motor Off » désactive la commande moteur,
- « Search Index » recherche la position d'index suivante, vers l'avant « Search index dir. → Forwards » ou l'arrière « Backwards ».
- Les paramètres de position, vitesse et accélération peuvent être définis en valeur absolue (« Absolute ») ou relative (« Relative »).

Fenêtre de paramétrage de la boîte de fonctions des 4 séquences de programme (Notons que les valeurs ci-dessous sont données à titre d'exemple, mais conviennent parfaitement au modèle de démonstration V-PCX 24 du chapitre 12.) :



Fenêtre de paramétrage (suite)



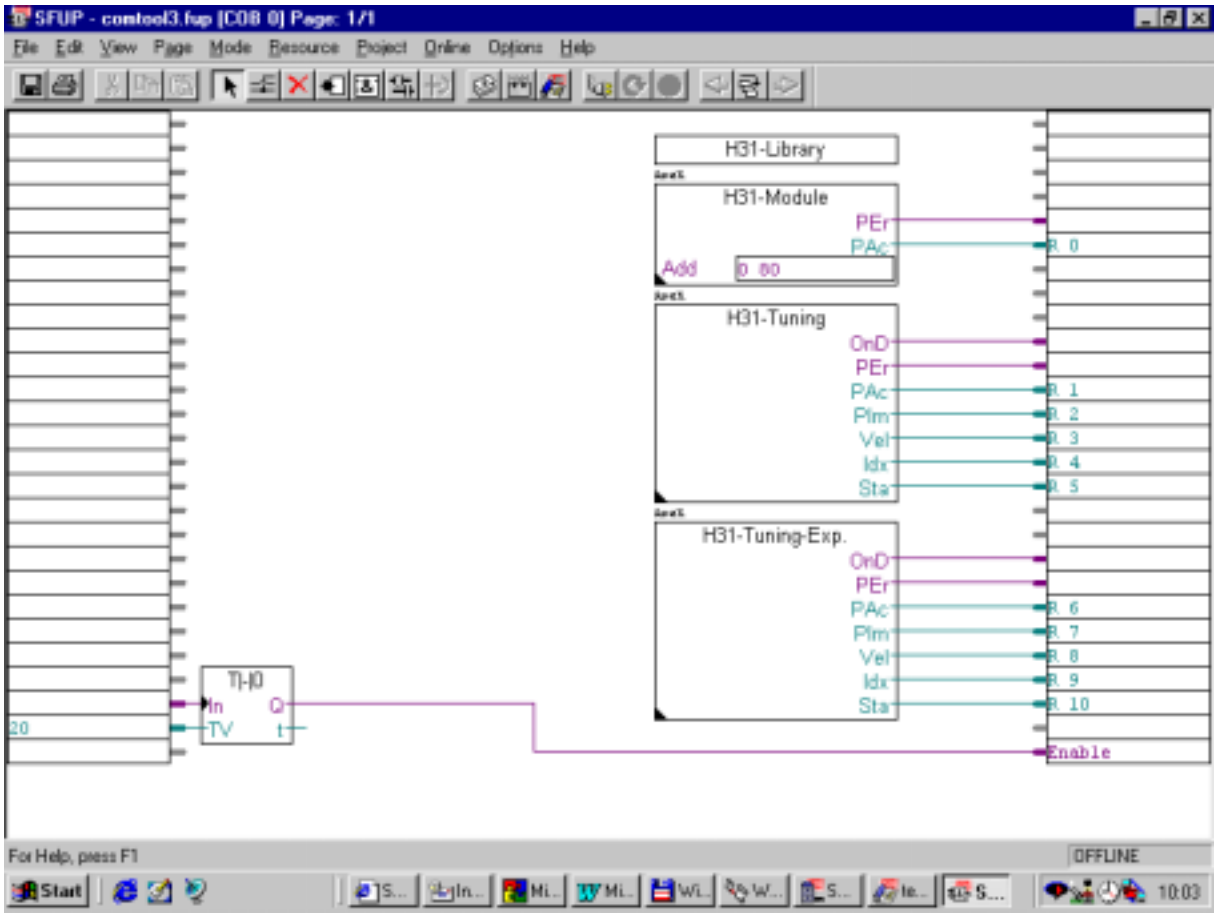
Acceleration B	>	100	>	100
Sequence C parameters				
Proportional Factor C	>	20	>	20
Integrative Factor C	>	20	>	20
Derivative Factor C	>	30	>	30
Integrative Limit C	>	4000	>	4000
Der. Sampling Interval C	>	5	>	5
Destination C	>	30	>	30
Velocity C	>	200	>	200
Acceleration C	>	400	>	400
Sequence D parameters				
Proportional Factor D	>	20	>	20
Integrative Factor D	>	20	>	20
Derivative Factor D	>	30	>	30
Integrative Limit D	>	4000	>	4000
Der. Sampling Interval D	>	5	>	5
Destination D	>	0	>	0
Velocity D	>	10	>	10
Acceleration D	>	20	>	20

Pour parfaire la mise en service, il est bon d'utiliser un oscilloscope à mémoire permettant d'afficher la tension de commande ±10 V ; c'est en effet le seul moyen d'observer le comportement du régulateur sous diverses charges et d'optimiser ses réglages.

Or, à ce jour, l'outil de mise en service ne dispose malheureusement pas de cet affichage.

Description des boîtes de fonctions de l'outil de mise en service

L'outil de mise en service se présente comme suit :



La temporisation, dans la marge inférieure, n'apparaît pas dans l'outil original. Il est conseillé de ne pas déclencher la commande moteur tant que l'autodiagnostic de l'unité centrale n'est pas terminé, ce qui évite tout saut incontrôlé du chariot à la mise sous tension du régulateur.

Boîte de fonctions H31-Library

Il s'agit d'une boîte d'organisation qui doit obligatoirement figurer en tête de page ; elle ne possède ni entrée-sortie, ni fenêtre de paramétrage.

Boîte de fonctions H31-Module

Elle assure l'initialisation du PCD2.H31x, dont l'adresse de base (par exemple O 80) doit être écrite dans le cadre « Add ».

Sorties	Format	Fonction
PEr	binaire	Erreur de position
PAc	entier	Position réelle

Fenêtre de paramétrage Cf . page 10-4

Boîte de fonctions H31-Tuning

Elle définit un programme simple de mouvement aller-retour destiné à optimiser les réglages.

Sorties	Format	Fonction
OnD	binaire	Position cible atteinte
PEr	binaire	Erreur de position
PAc	entier	Position réelle (en unité de longueur)
PIm	entier	Position réelle (en nombre d'impulsions) *)
Vel	entier	Vitesse réelle
IdX	entier	Registre d'index
Sta	entier	Registre d'état

Fenêtre de paramétrage Cf. page 10-3

Boîte de fonctions H31-Tuning-Exp.

Elle définit un programme de mouvement constitué de 4 séquences, dont chaque paramètre est modifiable.

Sorties	Format	Fonction
OnD	binaire	Position cible atteinte
PEr	binaire	Erreur de position
PAc	entier	Position réelle (en unité de longueur)
PIm	entier	Position réelle (en nombre d'impulsions) *)
Vel	entier	Vitesse réelle
IdX	entier	Registre d'index
Sta	entier	Registre d'état

Fenêtre de paramétrage Cf. pages 10-6 et 10-7

*) La sortie PIm donne le résultat de la multiplication de la sortie PAc par les données mécaniques (paramètre 9 du bloc Init).

Notes personnelles :

11. Sécurité

Si vous pilotez un entraînement avec un PCD2.H31x, certains points méritent une attention particulière pour sécuriser l'installation :

Mise sous tension de l'entraînement

Dès la mise sous tension du PCD, il ne faut pas plus de 2 secondes pour initialiser l'ensemble des modules d'E/S, passer l'unité centrale en mode « RUN » et être ainsi en mesure de traiter le programme utilisateur.

Dans ce laps de temps, la sortie analogique de régulation du H31x peut prendre n'importe quelle valeur entre –10 V et +l0 V. Il est donc impératif que la partie puissance de l'entraînement soit activée ou déclenchée par le programme utilisateur au moyen d'une sortie TOR ; cela garantit que l'entraînement reste toujours sous contrôle durant cette période.

Surveillance de l'erreur de position (par l'indicateur « fPosErr_x »)

Le dépassement de l'erreur de position autorisée signale des problèmes graves ; il faut donc toujours le surveiller.

Plusieurs origines peuvent être mises en cause :

1) Un défaut de raccordement sur le H31x
   Exemples : - liaison mécanique desserrée
   - non-concordance du sens de rotation moteur-codeur incrémental.
2) De mauvais réglages PID.
3) Un dimensionnement insuffisant de l'entraînement (ampli et/ou moteur).
4) Un mauvais paramétrage du déplacement : le servo-amplificateur ou le moteur est alors incapable de suivre l'accélération ou la vitesse imposée par le H31x.
5) Un rotor bloqué (problèmes mécaniques du servomoteur).
6) Une erreur matérielle sur le H31x (par ex., sortie analogique de régulation en défaut).
7) Une erreur matérielle sur le servo-amplificateur.

Si le dépassement de l'erreur de position trouve son origine dans les points 1 à 4, il est généralement décelé à la mise en service et peut être rectifié immédiatement.

S'il a pour origine les points 5 à 7, les problèmes peuvent aussi se manifester après la mise en service. Pour éviter d'endommager la machine, quelques préalables s'imposent :

Surveillez en permanence l'indicateur correspondant « fPosErr_x » dans le programme utilisateur ; en cas d'erreur, déconnectez l'entraînement au moyen d'une sortie TOR.

Il existe plusieurs façons de déconnecter l'entraînement ; pour certaines machines, il suffit de désactiver le régulateur de la partie puissance. Pour d'autres, il faut réduire la vitesse de l'entraînement aussi vite que possible par freinage (en court-circuitant l'entrée de consigne au niveau de la partie puissance) et couper l'alimentation dans un délai de quelques millisecondes. Toutefois, force est de constater que la possibilité ou l'obligation de déconnecter l'entraînement est fonction de la machine et doit être décidée au cas par cas.

Fins de course

Ils peuvent être exploités de deux manières différentes :

- associés à la fonction de prise d'origine « Home » pour rechercher la position 0 à la mise sous tension de l'installation,
- comme messages d'alarme dans le programme utilisateur pour indiquer un dépassement de position (à condition d'être correctement positionnés et programmés).

Fins de course de sécurité

Ces fins de course d'arrêt d'urgence doivent toujours être directement raccordés à l'entraînement. (Coupez l'alimentation de l'entraînement.)

Chien de garde

Activez toujours le chien de garde du PCD et utilisez le contact du chien de garde pour déconnecter l'entraînement directement.

Blocs d'exception XOB d'erreurs matérielles PCD

Programmez les XOB 0 à 5 et, au besoin, déconnectez l'entraînement directement par une sortie TOR.

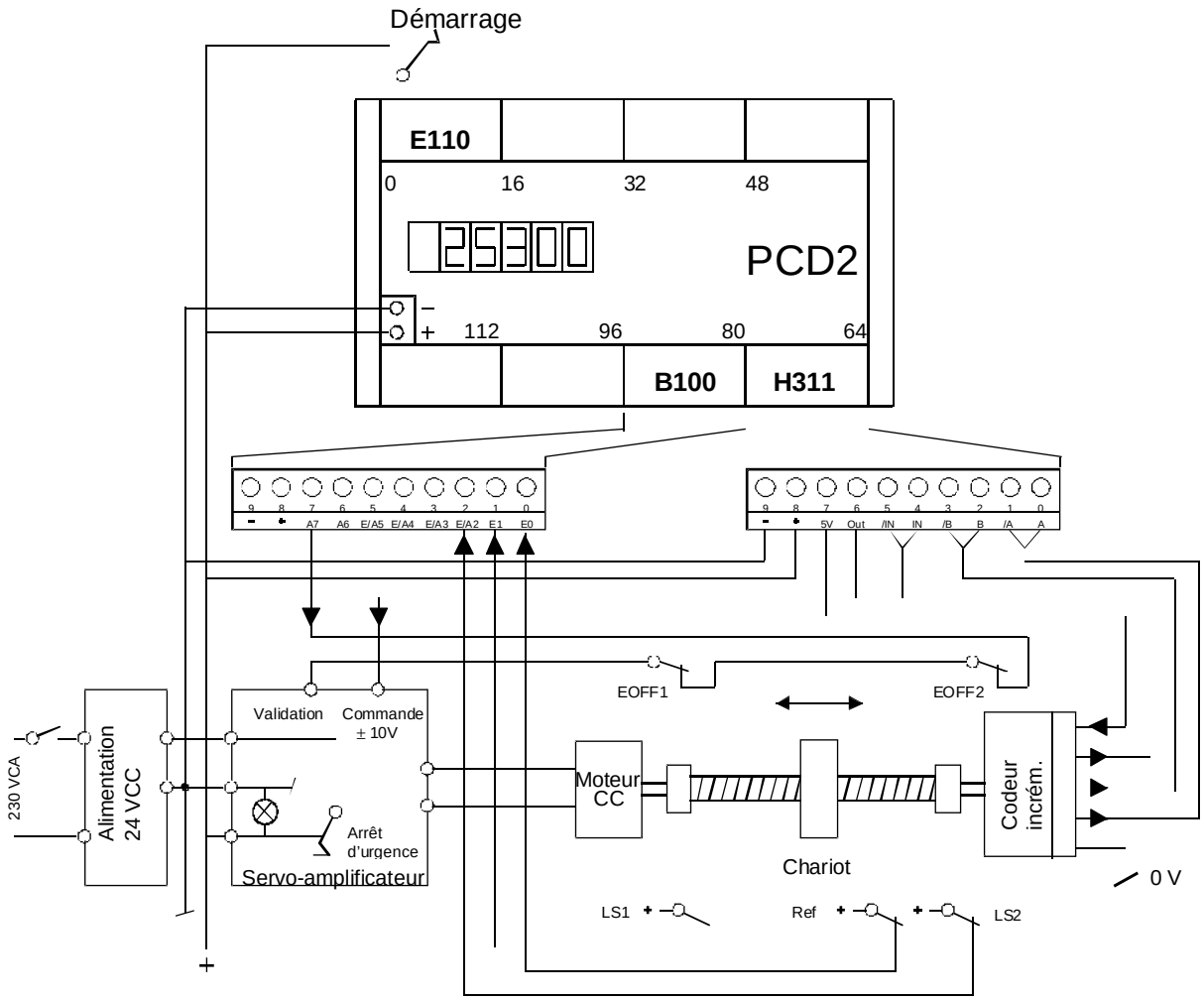
12. Exemples d'application

12.1 1er exemple : positionnement d'1 axe avec retour à l'origine

Le déplacement effectué dans cet exemple est identique à celui du chapitre 6 (simple aller et retour du chariot), à la différence près qu'il met en œuvre le contact de référence « Ref » et les deux fins de course « LS1 » et « LS2 », situés à chaque extrémité du parcours de retour à l'origine automatique.

Le matériel, basé sur le modèle de démonstration V-PCX 24, comprend les éléments suivants : chariot avec moteur CC, contact de référence, fins de course et fins de course d'extrémité, servo-amplificateur avec interrupteur d'urgence, alimentation, automate PCD2 et afficheur 6 chiffres.

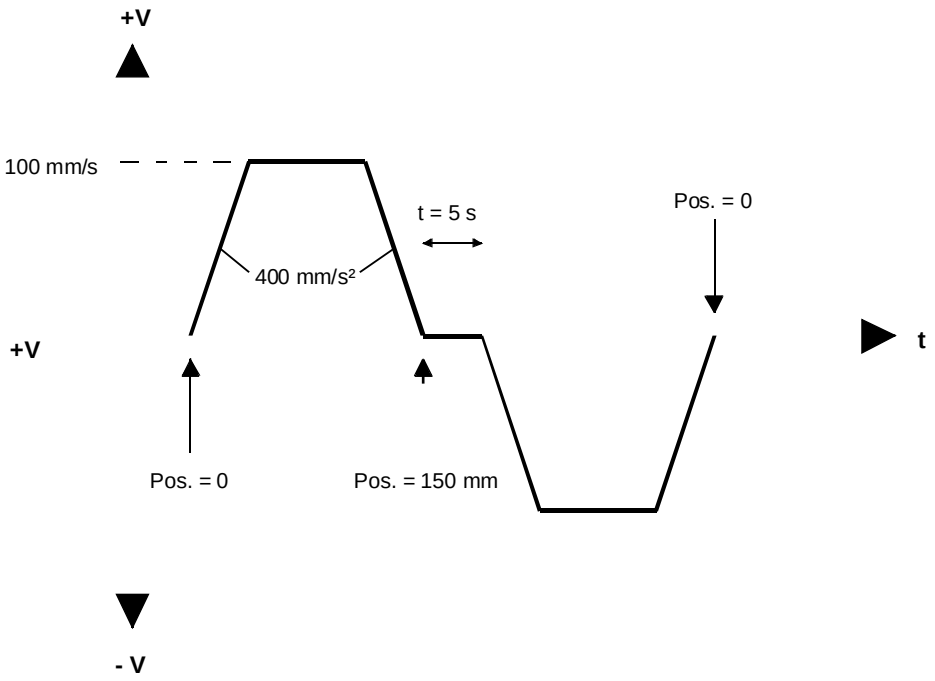
Équipé et câblé, le modèle se présente comme suit :



Caractéristiques techniques du modèle :

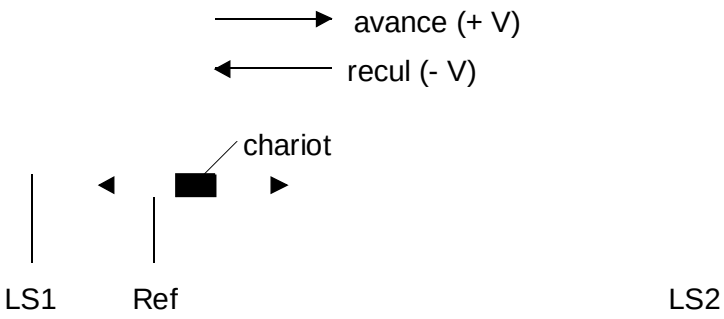
Codeur rotatif incrémental :	500 impulsions/tour
Pas de la vis :	2 mm
Vitesse maxi :	100 mm/s
Accélération maxi admissible :	50 mm/s^2

La prise d'origine, exécutée immédiatement après la mise sous tension, est suivie du profil de mouvement ci-après :



Prise d'origine

Avant d'étudier le programme utilisateur, il convient de s'attarder un instant sur cette fonction.



La disposition des fins de course et du contact de référence sur le modèle est grosso modo conforme à ce schéma. Durant l'exécution normale du programme, le chariot est dans la position illustrée ci-dessus.

En temps normal, le chariot est censé figurer entre « Ref » et « LS2 ». Pour revenir à l'origine, il lui faut se déplacer dans le sens du contact de référence, c'est-à-dire reculer. En atteignant ce dernier, il doit avancer librement jusqu'à ce qu'il arrive au signal d'index suivant du codeur incrémental ; c'est la preuve que la position de référence a été trouvée et que le programme de déplacement peut démarrer.

Dans ce cas, le paramétrage du bloc Retour à l'origine est le suivant :

```
LD   R 1010      ; registre PCD contenant la vitesse de
        500      ;  déplacement libre (lente)

LD   R 1011      ; registre PCD contenant la vitesse de
       5000      ;  recherche du contact de réf. (rapide)

CFB    Home      ; Appel du bloc de fonctions « Home »
     K 1         ;  Param.1 : n° de module
       0         ;  Param.2 : recul vers contact de réf.
       1         ;  Param.3 : avance libre
     R 1010      ;  Param.4 : vitesse avance libre
     R 1011      ;  Param.5 : vitesse recul vers contact
       30        ;  Param.6 : temporisation = 30 s
     I 64        ;  Param.7 : adresse entrée référence
```

Comportement du chariot en fonction de sa position de départ :

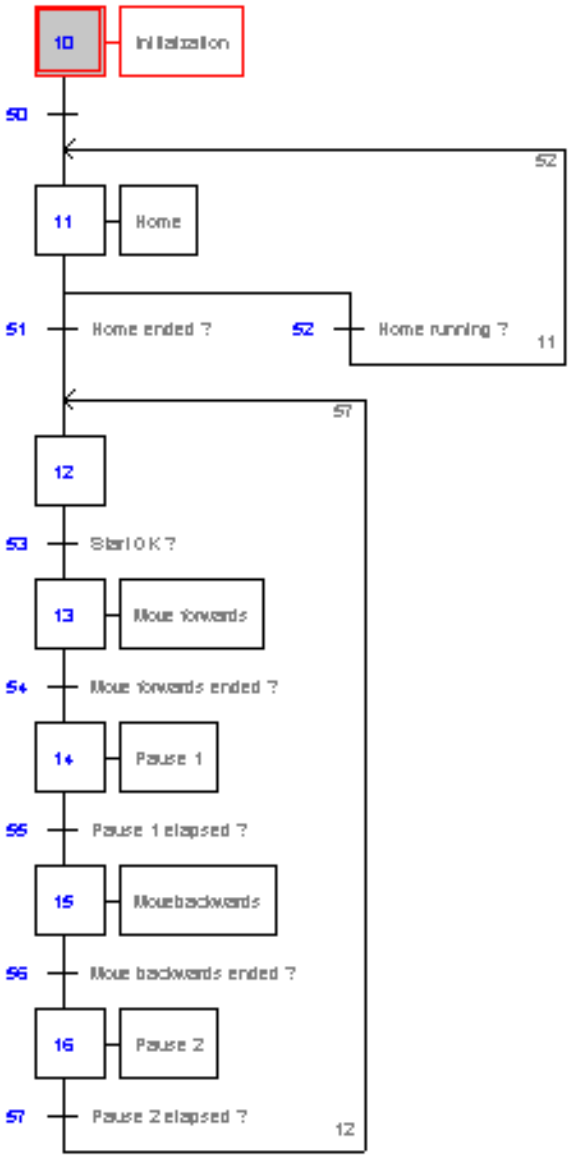
a) Chariot situé entre Ref et LS2 (position normale de départ) :
le chariot se déplace comme indiqué ci-dessus. S'il ne rencontre pas de référence à l'échéance de la temporisation, la fonction de retour à l'origine s'interrompt. Si Ref est activé à la mise sous tension (chariot situé au niveau du contact), le mouvement se limite à un déplacement libre jusqu'au signal d'index codeur suivant.

b) Chariot situé entre LS1 et Ref :
le chariot commence par reculer jusqu'à LS1, puis avance vers Ref. Il passe sur Ref à la vitesse du déplacement libre et poursuit sa course jusqu'à atteindre le signal d'index codeur suivant : la position d'origine a donc été atteinte et le programme de déplacement peut commencer.

c) Chariot situé à droite de LS2 :
la procédure est identique à celle du point a). Le chariot passe sur LS2, sans conséquence sur la prise d'origine.

d) Chariot situé à gauche de LS1 :
le chariot recule en direction du fin de course d'extrémité, sans trouver la position d'origine. S'il ne rencontre pas ce fin de course à l'expiration de la temporisation, la fonction de retour à l'origine est interrompue.

Programme utilisateur en GRAFTEC (« int-home.sfc »)



L'initialisation du module a lieu dans l'étape initiale IST. Elle est suivie de la prise d'origine (ST 1), qui se déroulera en continu jusqu'au terme de la fonction, signalé par l'indicateur « fEndHome_x ». Avant d'appeler le bloc Home, l'état des fins de course LS1 et LS2 (câblés aux entrées TOR du PCD) doit être lu dans les indicateurs « fLS1_x » et « fLS2_x ».

Á l'issue de la prise d'origine, le programme passe à la transition TR 53, qui correspond à l'attente de la condition de démarrage du programme de déplacement proprement dit.

Reportez-vous au § 6.1.2 pour la description du programme de déplacement.

Code programme de « int-home.sfc »

(Pour obtenir cette représentation, le fichier « int-home.sfc » doit être renommé « int-home.src ».)

```
  SB     0
;-------------------------------
  IST    10        ; Initialisation du module
         O 50

$include D2H310_B.equ
$group H310

  LD   R 1000
         1.00      ;  Données mécaniques

  LD   R 1001
         100000    ;  Vitesse initiale absolue

  LD   R 1002
         400000    ;  Accélération initiale absolue

  CFB    Init      ; Initialisation du module
       K 1         ;  N° de module
         250       ;  Proportionnelle (régulateur PID)
         0         ;  Intégrale (régulateur PID)
         60        ;  Dérivée (régulateur PID)
         4000      ;  Limite d'intégration
         5         ;  Tps d'échantillonnage de la dérivée
         500       ;  Tolérance de position
         0         ;  Comportement en cas d'erreur de pos.
         R 1000    ;  Registre des données mécaniques
         R 1001    ;  Registre de la vitesse initiale
         R 1002    ;  Registre de l'accélération initiale

  ld   t 0         ; Temporisation de validation « Enable »
         20        ;  2 secondes
  EST    ;10
```



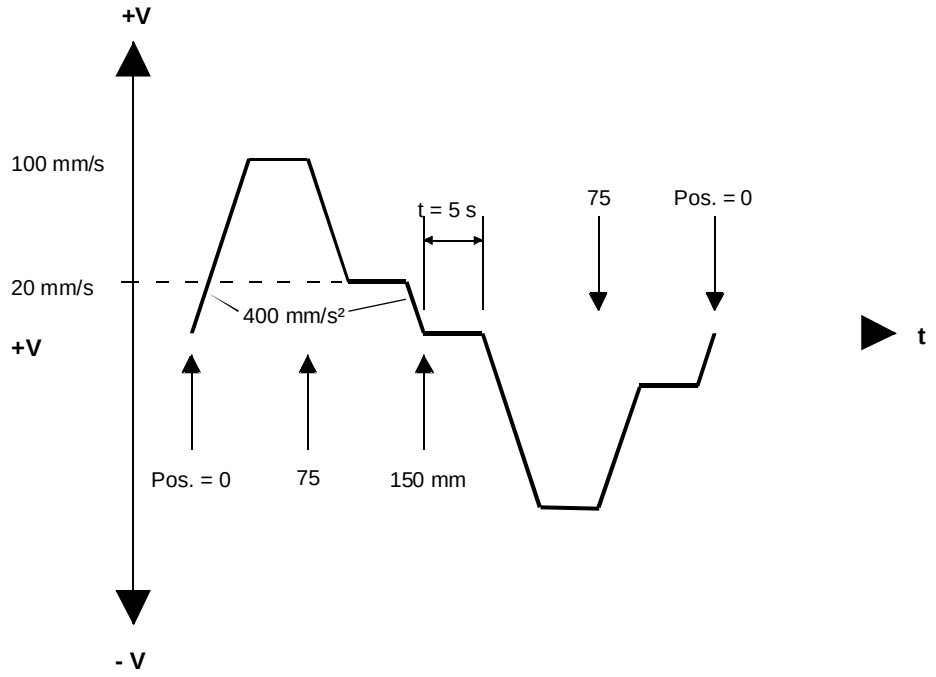
```
;-------------------------------
  ST     11        ; Retour à l'origine
         I 50
         I 52      ;  Retour à l'origine en cours ?
         O 51      ;  Retour à l'origine terminé ?
         O 52      ;  Retour à l'origine en cours ?

  set  o 71        ; Validation

  res    fEndHome_1

  stl  i 65
  out    fLS1_1

  stl  i 66
  out    fLS2_1

  ld   r 1010
         1000
  ld   r 1011
         10000

  cfb    home
       k 1         ;  N° de module
         0         ;  Sens de recherche
         1         ;  Sens du déplacement libre
       r 1010      ;  Vitesse mini
       r 1011      ;  Vitesse maxi
         50        ;  Tempo
       i 64        ;  Entrée de référence
  EST    ;11

;-------------------------------
  ST     12
         I 51      ;  Retour à l'origine terminé ?
         I 57      ;  Pause n° 2 écoulée ?
         O 53      ;  Démarrage OK ?
  EST    ;12

;-------------------------------
  ST     13        ; Avance
         I 53      ;  Démarrage OK ?
         O 54      ;  Avance terminée ?

  LD   R 100       ; Chargement de la valeur absolue
         60000     ;  de la position cible

  CFB    Exec      ; Exécution du bloc de fonctions
       K 1         ;  N° de module
         LdDestAbs ;  Instruction : charg. pos. cible abs.
       R 100       ;  Registre de la position cible absolue

  CFB    Exec      ; Exécution du bloc de fonctions
       K 1         ;  N° de module
         StartMot  ;  Instruction : démarrage déplacement
         rNotUsed  ;  Registre fictif
  EST    ;13

;-------------------------------
  ST     14        ; Pause n° 1
         I 54      ;  Avance terminée ?
         O 55      ;  Pause n° 1 écoulée ?

  ld   t 0
         50
  EST    ;14
```

```
;-------------------------------
ST     15          ; Recul
       I 55        ;  Pause n° 1 écoulée ?
       O 56        ;  Recul terminé ?

  LD   R 100       ; Chargement de la valeur absolue
         0         ;  de la position cible

  CFB    Exec      ; Exécution du bloc de fonctions
       K 1         ;  N° de module
         LdDestAbs ;  Instruction : charg. pos. cible abs.
       R 100       ;  Registre de la position cible absolue

  CFB    Exec      ; Exécution du bloc de fonctions
       K 1         ;  N° de module
         StartMot  ;  Instruction : démarrage déplacement
         rNotUsed  ;  Registre fictif
  EST    ;15

;-------------------------------
ST     16          ; Pause n° 2
       I 56        ;  Recul terminé ?
       O 57        ;  Pause n° 2 écoulée?

  ld   t 0
         50
  EST    ;16

;-------------------------------
TR     50
       I 10        ;  Initialisation
       O 11        ;  Retour à l'origine

  stl  t 0
  ETR    ;50

;-------------------------------
  TR     51        ; Retour à l'origine terminé ?
         I 11      ;  Retour à l'origine
         O 12

  CFB    Exec      ; Exécution du bloc de fonctions
       K 1         ;  N° de module
         RdActPos  ;  Instruction : lecture position réelle
       R 90        ;  Registre de la position réelle

  DSP  R 90        ; Affichage du contenu du registre

  sth    fEndHome_1
  ETR    ;51

;-------------------------------
  TR     52        ; Retour à l'origine en cours ?
         I 11      ;  Retour à l'origine
         O 11      ;  Retour à l'origine

  CFB    Exec      ; Exécution du bloc de fonctions
       K 1         ;  N° de module
         RdActPos  ;  Instruction : lecture position réelle
       R 90        ;  Registre de la position réelle

  DSP    R 90      ; Affichage du contenu du registre

  stl    fEndHome_1
  ETR    ;52
```

```
;-------------------------------
  TR     53        ; Démarrage OK ?
         I 12
         O 13      ;  Avance

  CFB    Exec      ; Exécution du bloc de fonctions
       K 1         ;  N° de module
         RdActPos  ;  Instruction : lecture position réelle
       R 90        ;  Registre de la position réelle
  DSP  R 90        ; Affichage du contenu du registre

  sth  i 0
  ETR    ;53

;-------------------------------
  TR     54        ; Avance terminée ?
         I 13      ;  Avance
         O 14      ;  Pause n° 1

  CFB    Exec      ; Exécution du bloc de fonctions
       K 1         ;  N° de module
         RdActPos  ;  Instruction : lecture position réelle
       R 90        ;  Registre de la position réelle
  DSP  R 90        ; Affichage du contenu du registre

  CFB    Exec      ; Exécution du bloc de fonctions
       K 1         ;  N° de module
         RdStatRg  ;  Instruction : lecture registre d'état
       R 0         ;  Valeur du registre d'état

  STH    fOnDest_1 ; Position atteinte ?
  ETR    ;54

;-------------------------------
  TR     55        ; Pause n° 1 écoulée ?
         I 14      ;  Pause n° 1
         O 15      ;  Recul
  stl  t 0
  ETR    ;55

;-------------------------------
  TR     56        ; Recul terminé ?
         I 15      ;  Recul
         O 16      ;  Pause n° 2

  CFB    Exec      ; Exécution du bloc de fonctions
       K 1         ;  N° de module
         RdActPos  ;  Instruction : lecture position réelle
       R 90        ;  Registre de la position réelle
  DSP  R 90        ; Affichage du contenu du registre

  CFB    Exec      ; Exécution du bloc de fonctions
       K 1         ;  N° de module
         RdStatRg  ;  Instruction : lecture registre d'état
       R 0         ;  Valeur du registre d'état

  STH    fOnDest_1 ; Position atteinte ?
  ETR    ;56

;-------------------------------
  TR     57        ; Pause n° 2 écoulée ?
         I 16      ;  Pause n° 2
         O 12
  stl  t 0
  $endgroup
  ETR    ;57

  ESB    ;0
```

12.2 2ème exemple : positionnement d'1 axe, à 2 vitesses

Cet exemple porte encore sur un chariot effectuant un aller et retour, mais cette fois, il met en œuvre deux vitesses (une vitesse rapide suivie d'une vitesse lente), dans les deux sens du déplacement (avance, puis recul).
Il a surtout pour but d'étudier le principe de modification des paramètres en cours de mouvement. Á noter que pour éviter une surcharge du programme, il ne comporte pas de retour à l'origine (absence de bloc Home).



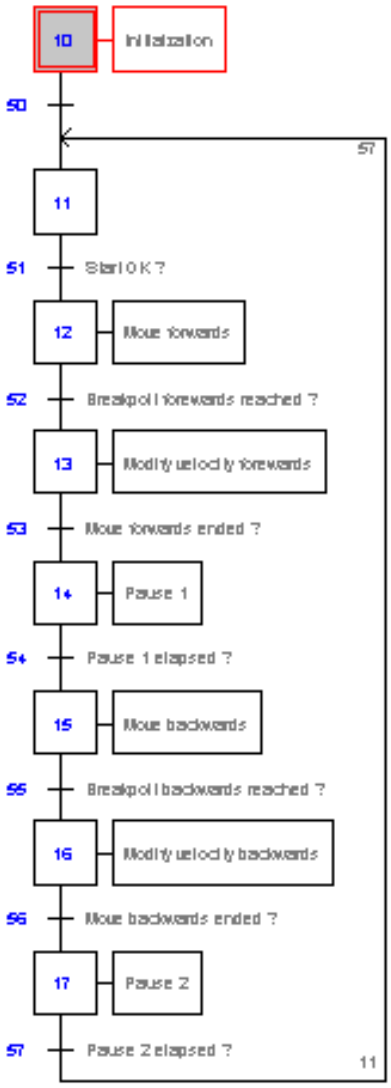
Les deux exemples précédents (Cf. § 6.1.2 et 11.1) reposaient sur des paramètres de vitesse, d'accélération et de régulation PID fixes. En fait, ces paramétrages n'étaient définis qu'une fois dans le bloc d'initialisation Init et restaient identiques tout au long du programme. Les seules modifications concernaient les positions relative et absolue du déplacement du chariot.

Dans cet exemple, la vitesse est modifiée en cours de mouvement (Cf. schéma ci-dessus). On utilise pour cela un point d'arrêt ; en effet, les étapes ST 12 et ST 15 contiennent non seulement les définitions de la position cible suivante, mais aussi celles d'un point d'arrêt, dont la position correspond en fait au point de basculement grande vitesse → petite vitesse.

Aux transitions TR 52 et TR 55, l'arrivée sur ce point d'arrêt est déterminée par l'interrogation de l'indicateur « fBrkPt_x ». Puis, en ST 13 et en ST 16, la nouvelle vitesse est chargée et activée dans chaque cas par un bloc de démarrage « StartMot ».

La modification des autres paramètres, en cours de mouvement, suit le même principe.

Programme utilisateur en GRAFTEC (« 2-speed.sfc »)



L'initialisation a lieu dans l'étape initiale IST.

Les étapes ST 12 et ST 15 contiennent la définition de la position cible suivante et du point d'arrêt marquant le changement de vitesse, le tout étant activé par la condition de démarrage « StartMot ».

Les transitions TR 52 et TR 55 comportent une pause jusqu'à ce que le point d'arrêt en question soit atteint, à des fins de contrôle du programme utilisateur. La bonne exécution du changement de vitesse est assurée par le module lui-même et non par le programme utilisateur.

En ST 13 et ST 16, la nouvelle vitesse (lente) est définie et lancée par Start.

Les transitions TR 53 et TR 56 comportent une pause jusqu'à ce que la position cible soit atteinte, à des fins de contrôle du programme utilisateur. L'arrêt correct du déplacement est assuré par le module lui-même et non par le programme utilisateur.

La plupart des transitions s'accompagnent d'une lecture et d'un affichage de la position réelle. Dans la pratique, ces parties de programme peuvent être facilement omises ou n'être exécutées que lorsqu'elles se justifient réellement.

Code programme de « 2-speed.sfc »

(Pour obtenir cette représentation, le fichier « int-home.sfc » doit être renommé « 2-speed.src ».)

```
  SB     0
;-------------------------------
  IST    10        ; Initialisation
         O 50

$include D2H310_B.equ
$group H310

  LD   R 1000
         1.00      ;  Données mécaniques

  LD   R 1001
         100000    ;  Vitesse initiale absolue

  LD   R 1002
         400000    ;  Accélération initiale absolue

  CFB    Init      ; Initialisation du module
       K 1         ;  N° de module
         250       ;  Proportionnelle (régulateur PID)
         0         ;  Intégrale (régulateur PID)
         60        ;  Dérivée (régulateur PID)
         4000      ;  Limite d'intégration
         5         ;  Temps d'échantillonnage de la dérivée
         500       ;  Tolérance de position
         0         ;  Comportement en cas d'erreur de pos.
       R 1000      ;  Registre des données mécaniques
       R 1001      ;  Registre de la vitesse initiale
       R 1002      ;  Registre de l'accélération initiale

  ld   t 0         ; Temporisation de validation « Enable »
         20        ;  2 secondes

  EST    ;10

;-------------------------------
  ST     11
         I 50
         I 57      ;  Pause n° 2 écoulée ?
         O 51      ;  Démarrage OK ?

  set    o 71      ; Validation

  EST    ;11
```

```
;-------------------------------
  ST     12         ; Avance
         I 51       ;  Démarrage OK ?
         O 52       ;  Point d'arrêt AV atteint ?

  LD   R 100        ; Chargement de la valeur absolue
         60000      ;  de la position cible

  LD   R 101        ; Chargement de la valeur absolue
         30000      ;  du point d'arrêt

  LD   R 102        ; Chargement de la
         100000     ;  vitesse absolue

  CFB    Exec       ; Exécution du bloc de fonctions
       K 1          ;  N° de module
         LdDestAbs  ;  Instruction : charg. pos. cible abs.
       R 100        ;  Registre de la position cible absolue

  CFB    Exec       ; Exécution du bloc de fonctions
       K 1          ;  N° de module
         LdBrkPtAbs;  Instruction : charg. pt d'arrêt abs.
       R 101        ;  Registre du point d'arrêt absolu

  CFB    Exec       ; Exécution du bloc de fonctions
       K 1          ;  N° de module
         LdVelAbs   ;  Instruction : charg. vitesse absolue
       R 102        ;  Vitesse

  CFB    Exec       ; Exécution du bloc de fonctions
       K 1          ;  N° de module
         StartMot   ;  Instruction : démarrage déplacement
         rNotUsed   ;  Registre fictif

  EST    ;12

;-------------------------------
  ST     13         ; Changement de vitesse AV
         I 52       ;  Point d'arrêt AV atteint ?
         O 53       ;  Avance terminée ?

  LD   R 102        ; Chargement de la
         20000      ;  vitesse absolue

  CFB    Exec       ; Exécution du bloc de fonctions
       K 1          ;  N° de module
         LdVelAbs   ;  Instruction : charg. vitesse absolue
       R 102        ;  Vitesse

  CFB    Exec       ; Exécution du bloc de fonctions
       K 1          ;  N° de module
         StartMot   ;  Instruction : démarrage déplacement
         rNotUsed   ;  Registre fictif

  EST    ;13

;-------------------------------
  ST     14         ; Pause n° 1
         I 53       ;  Avance terminée ?
         O 54       ;  Pause n° 1 écoulée ?

  ld   t 0
         50

  EST    ;14
```

```
;-------------------------------
  ST     15        ; Recul
         I 54      ;  Pause n° 1 écoulée ?
         O 55      ;  Point d'arrêt AR atteint ?

  LD   R 100       ; Chargement de la valeur absolue
         0         ;  de la position cible

  LD   R 101       ; Chargement de la valeur absolue
         30000     ;  du point d'arrêt

  LD   R 102       ; Chargement de la
         100000    ;  vitesse absolue

  CFB    Exec      ; Exécution du bloc de fonctions
       K 1         ;  N° de module
         LdDestAbs ;  Instruction : charg. pos. cible abs.
       R 100       ;  Registre de la position cible absolue

  CFB    Exec      ; Exécution du bloc de fonctions
       K 1         ;  N° de module
         LdBrkPtAbs;  Instruction : charg. pt d'arrêt abs.
       R 101       ;  Registre du point d'arrêt absolu

  CFB    Exec      ; Exécution du bloc de fonctions
       K 1         ;  N° de module
         LdVelAbs  ;  Instruction : charg. vitesse absolue
       R 102       ;  Vitesse

  CFB    Exec      ; Exécution du bloc de fonctions
       K 1         ;  N° de module
         StartMot  ;  Instruction : démarrage déplacement
         rNotUsed  ;  Registre fictif

  EST    ;15

;-------------------------------
  ST     16        ; Changement de vitesse AR
         I 55      ;  Point d'arrêt AR atteint ?
         O 56      ;  Recul terminé ?

  LD   R 102       ; Chargement de la
         20000     ;  vitesse absolue

  CFB    Exec      ; Exécution du bloc de fonctions
       K 1         ;  N° de module
         LdVelAbs  ;  Instruction : charg. vitesse absolue
       R 102       ;  Vitesse

  CFB    Exec      ; Exécution du bloc de fonctions
       K 1         ;  N° de module
         StartMot  ;  Instruction : démarrage déplacement
         rNotUsed  ;  Registre fictif

  EST    ;16

;-------------------------------
  ST     17        ; Pause n° 2
         I 56      ;  Recul terminé ?
         O 57      ;  Pause n° 2 écoulée ?
  ld   t 0
         50

  EST    ;17
```

```
;-------------------------------
  TR     50
         I 10      ; Initialisation
         O 11
  ETR    ;50

;-------------------------------
  TR     51        ; Démarrage OK ?
         I 11
         O 12      ; Avance

  CFB    Exec      ; Exécution du bloc de fonctions
       K 1         ;  N° de module
         RdActPos  ;  Instruction : lecture position réelle
       R 90        ;  Registre de la position réelle

  DSP  R 90        ; Affichage du contenu du registre

  sth  i 0

  ETR    ;51

;-------------------------------
  TR     52        ; Point d'arrêt AV atteint ?
         I 12      ;  Avance
         O 13      ;  Changement de vitesse AV

  CFB    Exec      ; Exécution du bloc de fonctions
       K 1         ;  N° de module
         RdActPos  ;  Instruction : lecture position réelle
       R 90        ;  Registre de la position réelle

  DSP  R 90        ; Affichage du contenu du registre

  CFB    Exec      ; Exécution du bloc de fonctions
       K 1         ;  N° de module
         RdStatRg  ;  Instruction : lecture registre d'état
       R 0         ;  Valeur du registre d'état

  STH    fBrkPt_1  ; Point d'arrêt atteint ?

  ETR    ;52

;-------------------------------
  TR     53        ; Avance terminée ?
         I 13      ;  Changement de vitesse AV
         O 14      ;  Pause n° 1

  CFB    Exec      ; Exécution du bloc de fonctions
       K 1         ;  N° de module
         RdActPos  ;  Instruction : lecture position réelle
       R 90        ;  Registre de la position réelle

  DSP  R 90        ; Affichage du contenu du registre

  CFB    Exec      ; Exécution du bloc de fonctions
       K 1         ;  N° de module
         RdStatRg  ;  Instruction : lecture registre d'état
       R 0         ;  Valeur du registre d'état

  STH    fOnDest_1 ; Position atteinte ?

  ETR    ;53
```

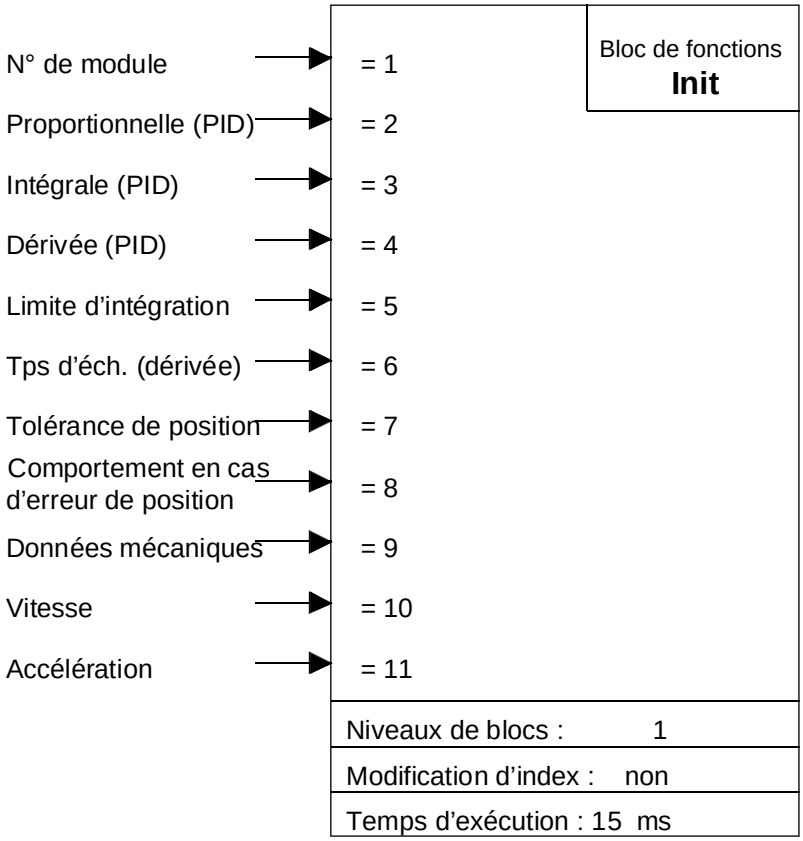
```
;-------------------------------
  TR     54        ; Pause n° 1 écoulée ?
         I 14      ;  Pause n° 1
         O 15      ;  Recul

  stl  t 0

  ETR    ;54

;-------------------------------
  TR     55        ; Point d'arrêt AR atteint ?
         I 15      ;  Recul
         O 16      ;  Changement de vitesse AR

  CFB    Exec      ; Exécution du bloc de fonctions
       K 1         ;  N° de module
         RdActPos  ;  Instruction : lecture position réelle
       R 90        ;  Registre de la position réelle

  DSP  R 90        ; Affichage du contenu du registre

  CFB    Exec      ; Exécution du bloc de fonctions
       K 1         ;  N° de module
         RdStatRg  ;  Instruction : lecture registre d'état
       R 0         ;  Valeur du registre d'état

  STH    fBrkPt_1  ; Point d'arrêt atteint ?

  ETR    ;55

;-------------------------------
  TR     56        ; Recul terminé ?
         I 16      ;  Changement de vitesse AR
         O 17      ;  Pause n° 2

  CFB    Exec      ; Exécution du bloc de fonctions
       K 1         ;  N° de module
         RdActPos  ;  Instruction : lecture position réelle
       R 90        ;  Registre de la position réelle

  DSP  R 90        ; Affichage du contenu du registre

  CFB    Exec      ; Exécution du bloc de fonctions
       K 1         ;  N° de module
         RdStatRg  ;  Instruction : lecture registre d'état
       R 0         ;  Valeur du registre d'état

  STH    fOnDest_1 ; Position atteinte ?

  ETR    ;56

;-------------------------------
  TR     57        ; Pause n° 2 écoulée ?
         I 17      ;  Pause n° 2
         O 11

  stl    t 0

$endgroup

  ETR    ;57

  ESB    ;0
```

Notes personnelles :

Annexe A Récapitulatif des blocs de fonctions et des instructions PCD2.H31x en langage IL

Bloc de fonctions « INIT »

INIT Initialisation du module PCD2.H31x



Description :

Ce bloc définit le paramétrage du module PCD2.H31x et lit son adresse de base dans le fichier D2H310_B.MBA.

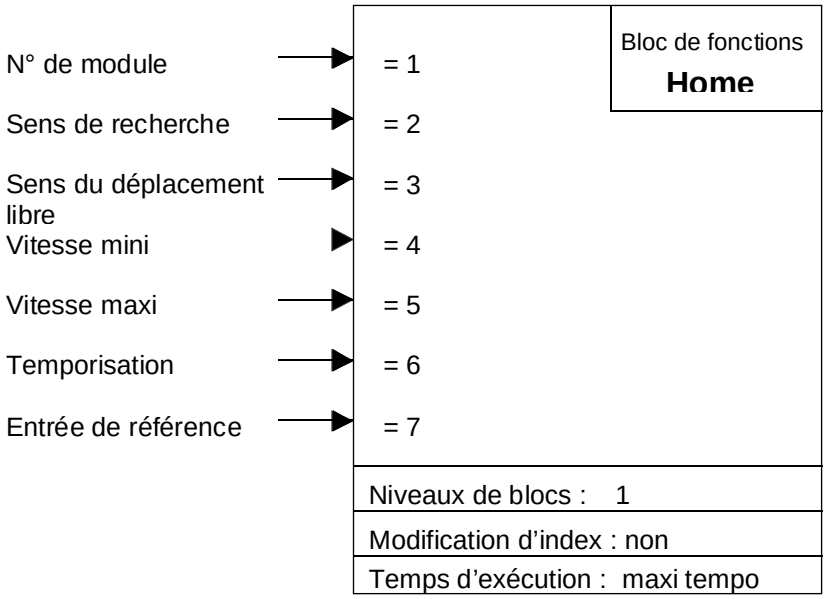
Il est constitué de 11 paramètres : le 1er s'exprime sous forme de constante K, les 9ème, 10ème et 11ème sont des adresses (absolues ou symboliques) de registres PCD ; tous les autres sont au format entier.

Le 9ème paramètre représente les caractéristiques techniques de la machine (paramètres du codeur et données mécaniques du système), saisies en virgule flottante (par exemple, 4 donnant 4.0 ou 4E1).

Par.	Désignation/Fonction	Type	Format	Plage	Remarques
= 1	N° de module	K	K n	K 1 à K 16	
= 2	Action proportionnelle		Entier	0 à 32 767	Fonction Proportionnelle du régulateur PID
= 3	Action intégrale		Entier	0 à 32 767	Fonction Intégrale du régulateur PID
= 4	Action dérivée		Entier	0 à 32 767	Fonction Dérivée du régulateur PID
= 5	Limite d'intégration		Entier	0 à 32 767	Limite d'influence de l'Intégrale sur le système
= 6	Temps d'échantillonnage de la dérivée		Entier	0 à 255	Période d'échantillonnage de la Dérivée
= 7	Tolérance de position		Entier	0 à 32 767	Erreur de position maximale comparée à la valeur calculée
= 8	Comportement en cas d'erreur de position		Entier	0 ou 1	Réponse du système en cas de dépassement de la tolérance de position : 0 = Indicateur erreur à 1, 1 = Arrêt déplacement.
= 9	Données mécaniques	R	Flottant	0 à $9.223371 * 10^{18}$	Paramètres du codeur et du système
= 10	Vitesse	R	Entier	--	Vitesse initiale, dictée par les données mécaniques
= 11	Accélération	R	Entier	--	Accélération initiale, dictée par les données mécaniques

Bloc de fonctions « HOME »

HOME Initialisation de la position d'origine



Description :

Ce bloc définit le paramétrage de la fonction Retour à l'origine du PCD2.H31x.

Il est constitué de 7 paramètres : le 1er s'exprime sous forme de constante K et le 7ème est une entrée notée « I » ; tous les autres sont au format entier.

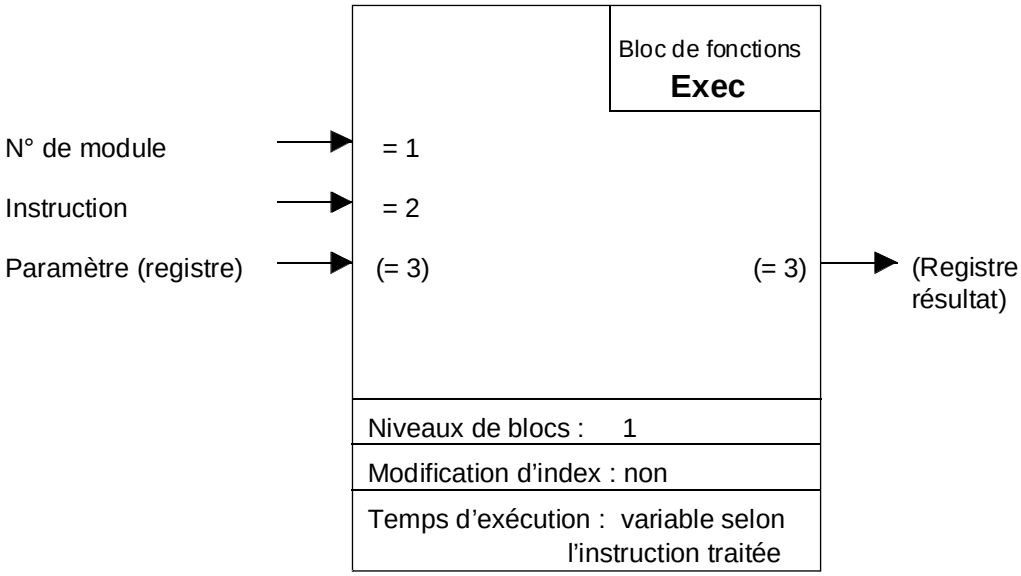
Notons que le PCD2.H310 possède une entrée de référence « Ref », dont l'adresse équivaut à l'adresse de base du module majorée de 11 : ainsi, pour un H310 portant l'adresse 64, l'entrée Ref se situe à l'adresse I 75. Á l'inverse, le PCD2.H311 n'a pas d'entrée Ref ; dans ce cas, n'importe quelle entrée PCD convient.

Important : il faut remettre à 0 l'indicateur « fEndHome » avant d'exécuter le bloc Home ; sinon, la fonction ne peut pas démarrer. Rappelons que **fEndHome** passe automatiquement à 1 au terme de la prise d'origine.

Par.	Désignation/Fonction	Type	Format	Plage	Remarques
= 1	N° de module	K	K n	K 1 à K 16	
= 2	Sens de recherche		Entier	0 ou 1	Sens du déplacement pour atteindre le contact de référence : 0 = recul 1 = avance
= 3	Sens du déplacement libre après atteinte du contact de référence		Entier	0 ou 1	Sens du déplacement libre, à partir du contact de référence : 0 = recul 1 = avance
= 4	Vitesse minimale	R	Entier	--	Vitesse du déplacement libre (à partir du contact de référence)
= 5	Vitesse maximale	R	Entier	--	Vitesse de recherche du contact de référence
= 6	Temporisation		Entier	0 à 65 535 [s]	Durée d'exécution de la prise d'origine (en secondes)
= 7	Entrée de référence	I	Entier	--	Entrée à laquelle est raccordé le contact de référence

Bloc de fonctions « EXEC »

EXEC Exécution d'une instruction du PCD2.H31x



Description :

Ce bloc envoie des commandes au PCD2.H31x.

Il est constitué de 3 paramètres : le 1er, saisi sous forme de constante K1 à K16, représente le numéro du module dont l'adresse de base est donnée par le fichier D2H310_B.MBA. Précisons que les blocs de fonctions peuvent traiter un maximum de 16 modules PCD2.H31x par PCD.

Le 2ème paramètre indique l'instruction mise en œuvre ; chacune d'elle est décrite dans les pages qui suivent.

Le 3ème paramètre donne le registre dans lequel sont transférés les paramètres de l'instruction (par exemple, la valeur de l'accélération pour l'instruction « LdAccAbs »). Si l'instruction n'a pas de paramètre (comme c'est le cas de l'instruction « Start »), on peut charger n'importe quel registre vide ou fictif (« rNotUsed ») du PCD.

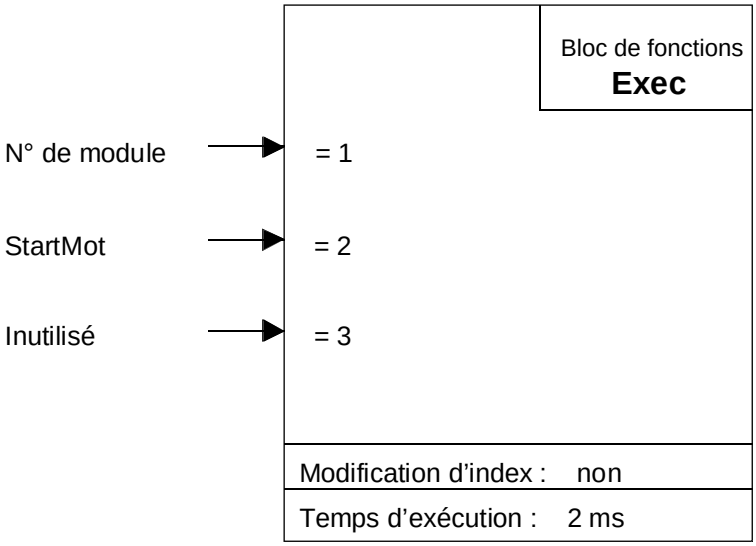
Liste des instructions PCD2.H31x (bloc EXEC)

N°	Code	Instruction	Détail en page
01	StartMot	Démarrage du déplacement	A-7
02	StopUrg	Arrêt d'urgence du déplacement	A-8
03	Stop	Arrêt du déplacement	A-9
04	MotOff	Désactivation de la régulation PID	A-10
05	RdActPos	Lecture de la position réelle	A-11
06	RdActVel	Lecture de la vitesse réelle	A-12
07	RdIntSum	Lecture de la somme d'intégration	A-13
08	RdIndexRg	Lecture du registre d'index	A-14
09	RdStatRg	Lecture du registre d'état	A-15
10	RdTargPos	Lecture de la position cible	A-16
11	RdTargVel	Lecture de la vitesse cible	A-17
12	GoForw	Avance	A-18
13	GoBackw	Recul	A-19
14	SgStpFor	Avance d'un pas	A-20
15	SgStpBak	Recul d'un pas	A-21
16	LdDestAbs	Chargement de la position cible absolue	A-22
17	LdDestRel	Chargement de la position cible relative	A-23
18	LdVelAbs	Chargement de la vitesse absolue	A-24
19	LdVelRel	Chargement de la vitesse relative	A-25
20	LdAccAbs	Chargement de l'accélération absolue	A-26
21	LdAccRel	Chargement de l'accélération relative	A-27
22	LdPropG	Chargement du gain proportionnel	A-28
23	LdIntG	Chargement du gain intégral	A-29
24	LdDerG	Chargement du gain dérivé	A-30
25	LdSampInt	Chargement du temps d'échantillonnage de la dérivée	A-31
26	LdIntLim	Chargement de la limite d'intégration	A-32
27	ActRegFact	Activation de la régulation PID	A-33
28	LdBrkPtAbs	Chargement d'un point d'arrêt absolu	A-34
29	LdBrkPtRel	Chargement d'un point d'arrêt relatif	A-35
30	ResStatRg	Remise à 0 du registre d'état	A-36
31	SetIdxPos	Définition de la position d'index	A-37
32	SetZero	Définition de la position 0	A-38
33	MotConf	Configuration du déplacement	A-39
34	SetPosTol	Définition de la tolérance de position	A-40

Le numéro de la première colonne représente la valeur absolue (0 à 34) du 2ème paramètre du bloc Exec ; il permet d'interpréter la fonction du bloc Exec lorsque l'on suit et analyse le programme utilisateur dans le débogueur.

StartMot Démarrage du déplacement

[01]



Description :

Cette instruction lance le déplacement, aux vitesse et accélération données, vers la position cible prédéfinie. Elle permet en outre de sélectionner le mode Régulation de position.

Dans l'octet d'état, « StartMot » remet à 0 les bits :

6 : Atteinte du point d'arrêt,

5 : Dépassement de l'erreur de position.

L'instruction « ResStatRg » (Cf. page A-36) remet à 0 tout ou partie des bits d'état. L'octet d'état est constitué des 8 premiers bits du registre d'état ; ces 8 bits sont remis à 0 en même temps que l'octet d'état.

StartMot remet également à 0 les bits 10 à 15 du registre d'état, dont le contenu peut être lu par l'instruction « RdStatRg » (Cf. page A-15).

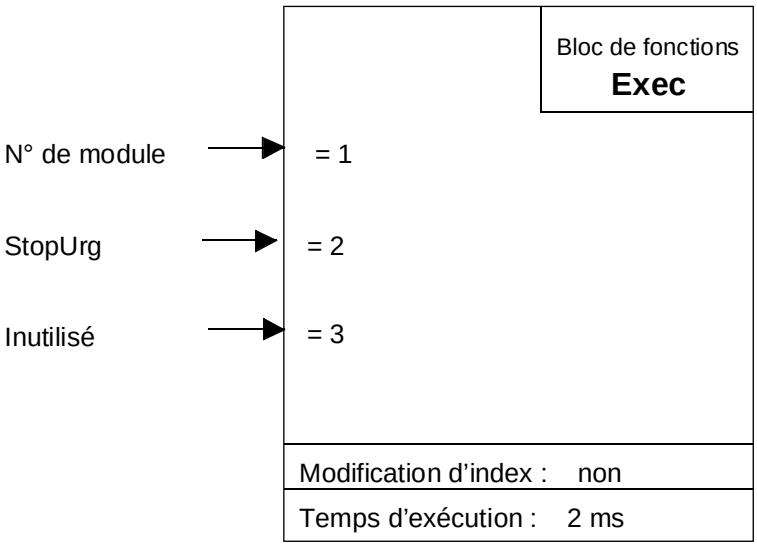
Notons que les indicateurs « fOnDest_x », « fBrkPt_x » et « fPosErr_x » ne sont rafraîchis qu'après un démarrage du déplacement par StartMot, suivi d'une lecture du registre d'état par RdStatRg.

Description des E/S intervenant dans l'instruction :

Param.	Désignation/Fonction	Type	Format	Plage	Remarques
= 1	N° de module	K		1 à 16	
= 2	Instruction : StartMot				
= 3	Registre vide du PCD ou « rNotUsed »	R			

StopUrg Arrêt d'urgence du déplacement

[02]



Description :

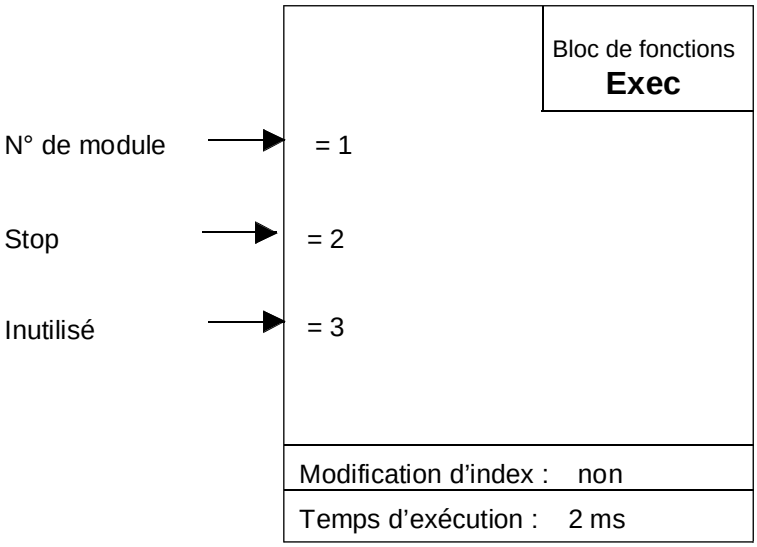
Cette instruction arrête brutalement le mouvement, sans rampe de freinage ; elle s'applique indistinctement aux deux modes de régulation, position et vitesse de rotation.

Description des E/S intervenant dans l'instruction :

Param.	Désignation/Fonction	Type	Format	Plage	Remarques
= 1	N° de module	K		1 à 16	
= 2	Instruction : StopUrg				
= 3	Registre vide du PCD ou « rNotUsed »	R			

Stop Arrêt du déplacement

[03]



Description :

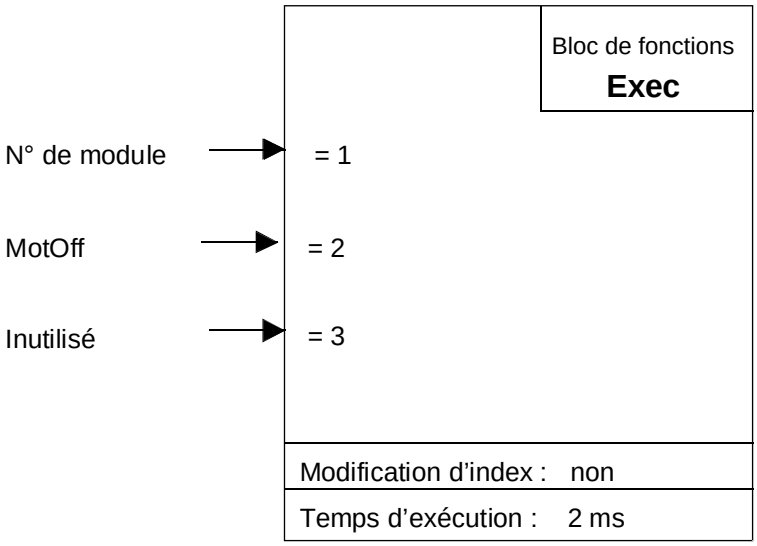
Cette instruction arrête normalement le mouvement, avec rampe de freinage ; elle s'applique indistinctement aux deux modes de régulation, position et vitesse de rotation.

Description des E/S intervenant dans l'instruction :

Par.	Désignation/Fonction	Type	Format	Plage	Remarques
= 1	N° de module	K		1 à 16	
= 2	Instruction : Stop				
= 3	Registre vide du PCD ou « rNotUsed »	R			

MotOff Désactivation de la régulation PID

[04]



Description :

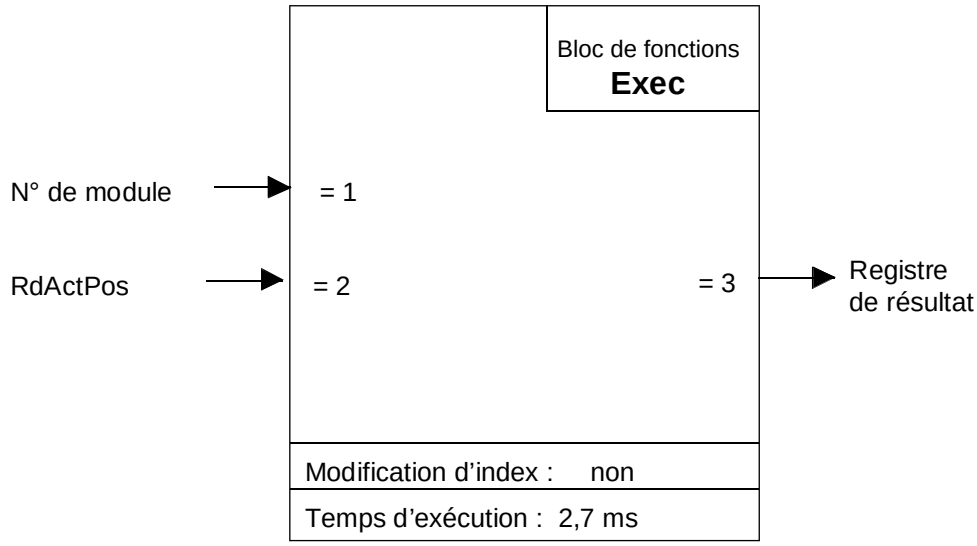
Cette instruction désactive la régulation PID agissant sur le moteur. Elle a le même effet qu'une coupure d'alimentation moteur.

Description des E/S intervenant dans l'instruction :

Param.	Désignation/Fonction	Type	Format	Plage	Remarques
= 1	N° de module	K		1 à 16	
= 2	Instruction : MotOff				
= 3	Registre vide du PCD ou « rNotUsed »	R			

RdActPos Lecture de la position réelle

[05]



Description :

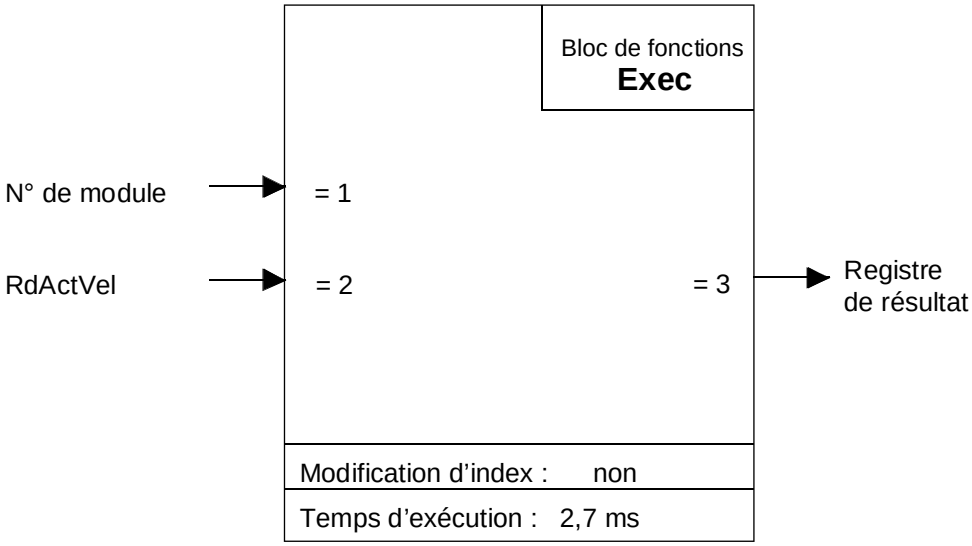
Cette instruction lit la position réelle du système. Elle doit être appelée à chaque fois qu'une lecture de position s'impose.

Description des E/S intervenant dans l'instruction :

Param.	Désignation/Fonction	Type	Format	Plage	Remarques
= 1	N° de module	K		1 à 16	
= 2	Instruction : RdActPos				
= 3	Registre PCD contenant la position réelle	R	Entier	31 bits	-2^{30} à $+(2^{30}-1)$

RdActVel Lecture de la vitesse réelle

[06]



Description :

Cette instruction permet de lire la vitesse réelle de déplacement ; elle doit être exécutée à chaque nouvelle lecture de vitesse.

Le format du processeur de vitesse est de 14 bits pour la partie entière et de 16 bits pour la décimale. Or, à l'heure actuelle, le processeur ne traite que la partie entière. Cela signifie qu'à basses vitesses les valeurs obtenues ne peuvent être comparées au résultat de l'instruction « RdTargVel » (Cf. A-17) qui, elle, contient la partie fractionnaire.

Si la valeur des données mécaniques est inférieure à 1, les basses vitesses sont difficiles à lire, la partie décimale de la vitesse n'étant pas disponible.

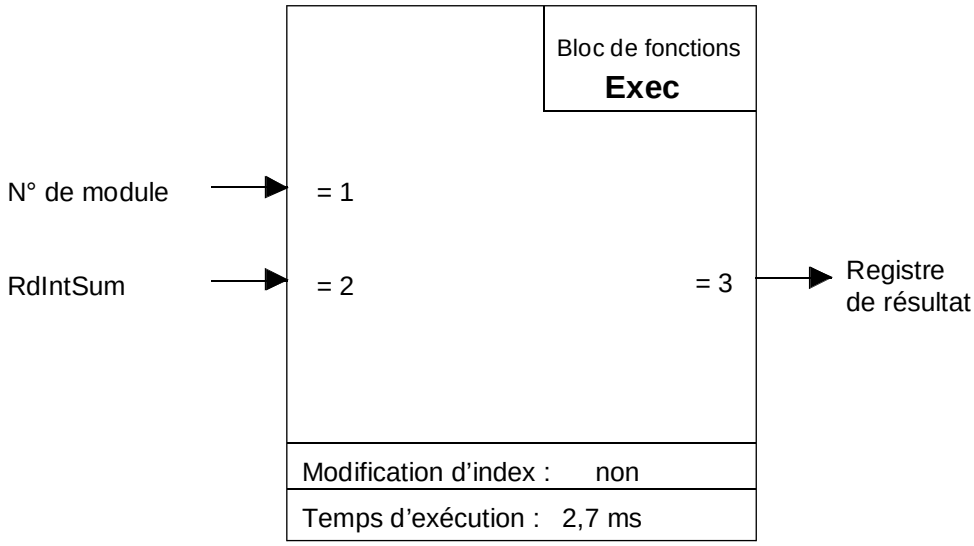
Description des E/S intervenant dans l'instruction :

Param.	Désignation/Fonction	Type	Format	Plage	Remarques
= 1	N° de module	K		1 à 16	
= 2	Instruction : RdActVel				
= 3	Registre PCD contenant la vitesse réelle	R	Entier	30 bits *)	Dépend des données mécaniques.

*) Résolution 14 bits

RdIntSum Lecture de la somme d'intégration

[07]



Description :

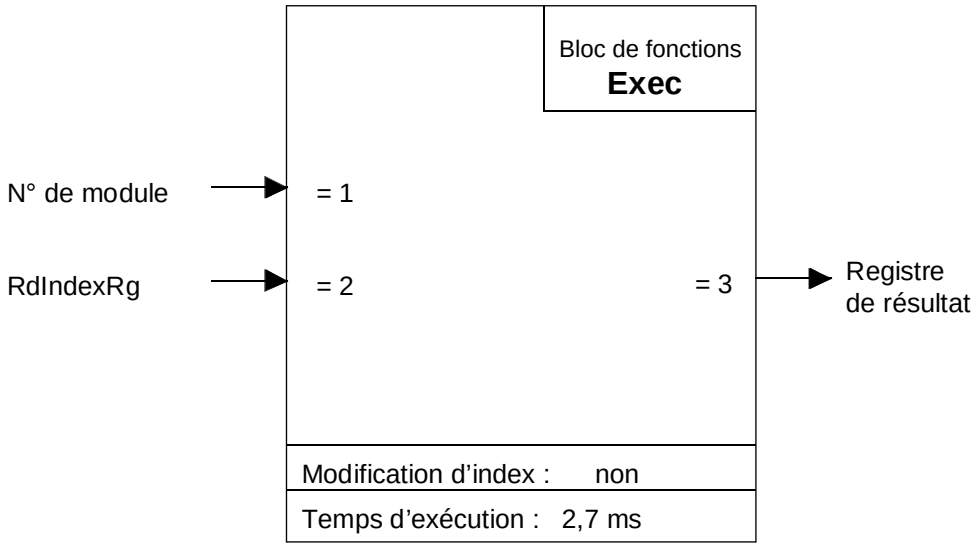
Cette instruction permet de lire la somme d'intégration de l'Intégrale du régulateur PID ; elle doit être appelée à chaque fois qu'une lecture de la somme d'intégration est nécessaire.

Description des E/S intervenant dans l'instruction :

Param.	Désignation/Fonction	Type	Format	Plage	Remarques
= 1	N° de module	K		1 à 16	
= 2	Instruction : RdIntSum				
= 3	Registre PCD contenant la somme d'intégration	R	Entier	15 bits	0 à 32 767

RdIndexRg Lecture du registre d'index

[08]



Description :

Cette instruction lit le registre d'index (compteur des tours du codeur) du H31x. (Á ne pas confondre avec le registre d'index de l'unité centrale du PCD.)

Si une instruction « SetIdxPos » (Cf. page A-37) a été exécutée, la position absolue sera enregistrée à la prochaine impulsion d'index codeur.

La lecture de ce registre peut faire partie du programme de contrôle d'erreur. La nouvelle position d'index moins la précédente, divisée par la résolution du codeur (division du codeur par 4), doit toujours donner une valeur entière.

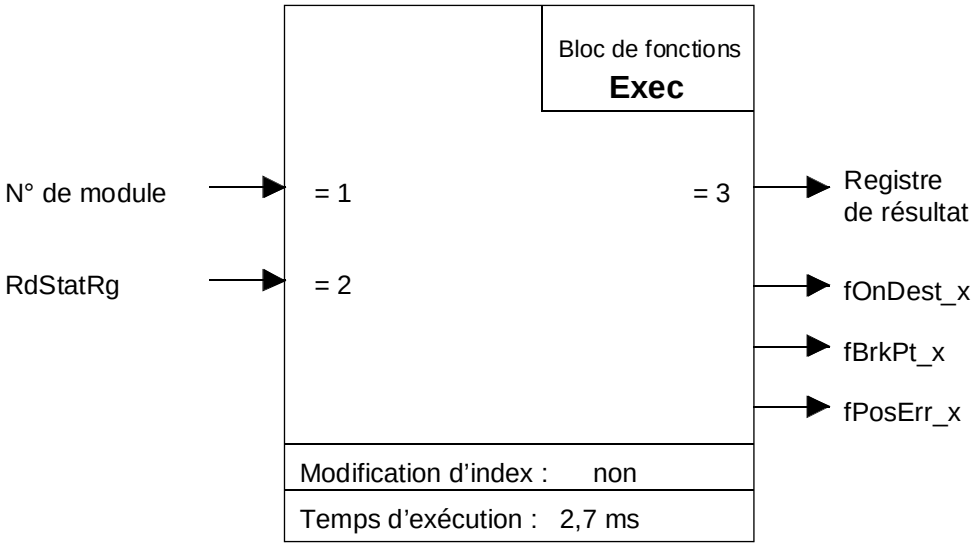
RdIndexRg doit être appelée à chaque fois qu'une lecture du registre d'index est nécessaire.

Description des E/S intervenant dans l'instruction :

Param.	Désignation/Fonction	Type	Format	Plage	Remarques
= 1	N° de module	K		1 à 16	
= 2	Instruction : RdIndexRg				
= 3	Registre PCD contenant la valeur du registre d'index	R	Entier	31 bits	-2^{30} à $+(2^{30}-1)$

RdStatRg Lecture du registre d'état

[09]



Description :

Cette instruction permet de lire le registre d'état ; elle doit être exécutée **avant** de pouvoir prendre en compte les indicateurs « fOnDest_x », « fPosErr_x » et « fBrkPt_x ».

La lecture de ce registre donne accès aux paramètres suivants :

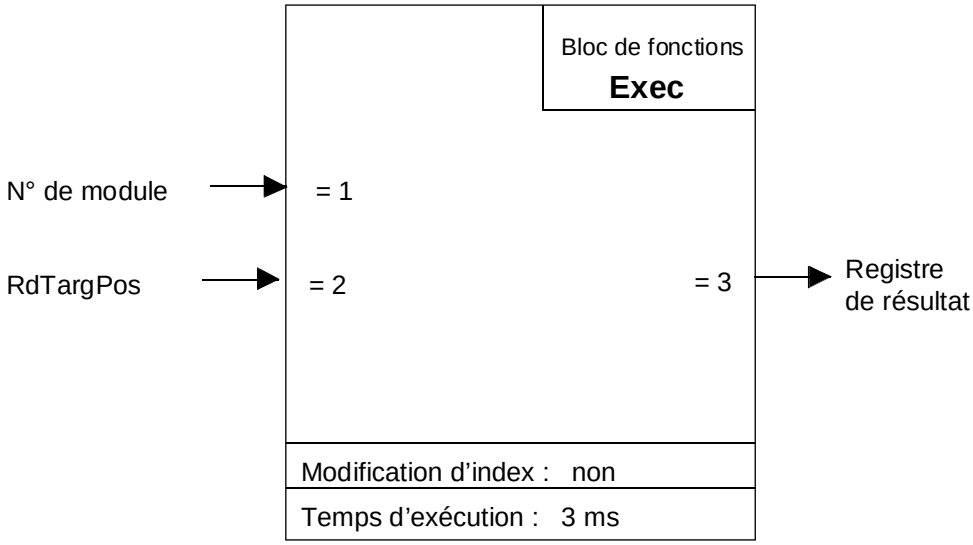
bit 15 : Interruption du processeur
bit 14 : Chargement de l'accélération
bit 13 : Mise à jour des paramètres du régulateur
bit 12 : Sens de déplacement AV
bit 11 : Mode vitesse
bit 10 : Atteinte de la cible
bit 9 : Désactivation en cas de dépassement de l'erreur de position
bit 8 : Réservé à un usage interne
bit 7 : Coupure moteur
bit 6 : Atteinte du point d'arrêt
bit 5 : Dépassement de l'erreur de position
bit 4 : Rebouclage
bit 3 : Acquisition d'une impulsion d'index
bit 2 : Fin du déplacement
bit 1 : Réservé à un usage interne
bit 0 : Acquisition de l'impulsion d'index suivante

RdStatReg doit être appelée à chaque fois qu'une lecture du registre d'état est nécessaire.

Param.	Désignation/Fonction	Type	Format	Plage	Remarques
= 1	N° de module	K		1 à 16	
= 2	Instruction : RdStatRg				
= 3	Registre PCD contenant la valeur du registre d'état	R	Entier	31 bits	-2^{30} à $+(2^{30}-1)$
fOnDest_x	Indicateur « atteinte cible »	F			
fBrkPt_x	Indic. « atteinte pt d'arrêt »	F			
fPosErr_x	Indicateur « dépassement erreur de position »	F			

RdTargPos Lecture de la position cible

[10]

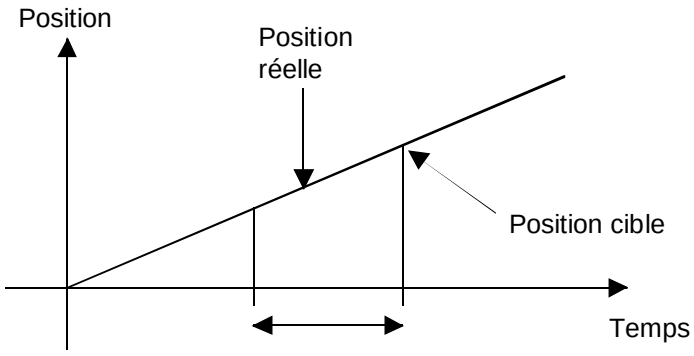


Description :

Cette instruction permet de lire la position cible en cours du déplacement réel.

Description des E/S intervenant dans l'instruction :

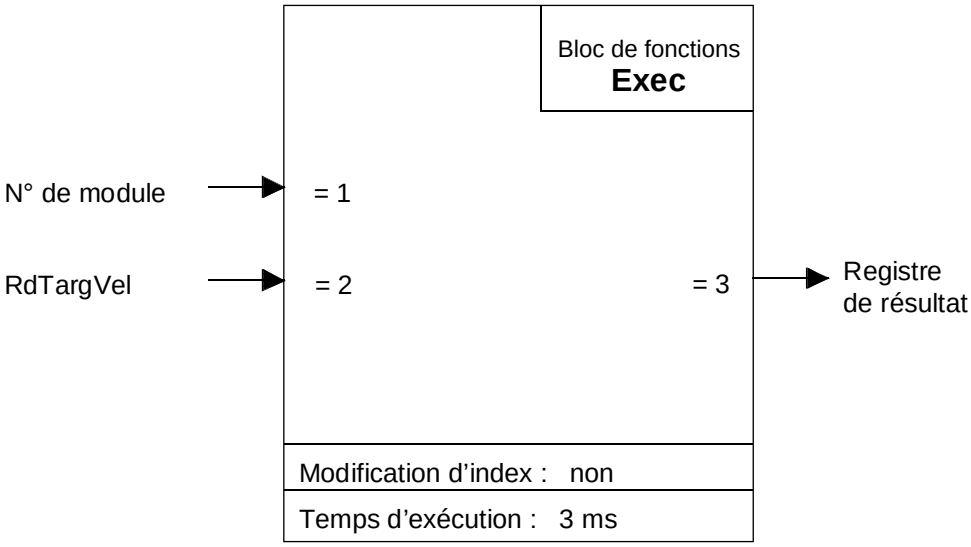
Param.	Désignation/Fonction	Type	Format	Plage	Remarques
= 1	N° de module	K		1 à 16	
= 2	Instruction : RdTargPos				
= 3	Registre PCD contenant la position cible	R	Entier	31 bits	-2^{30} à $+(2^{30}-1)$



Représentation de la position cible calculée en unités du temps d'échantillonnage (341 µs).

RdTargVel Lecture de la vitesse cible

[11]



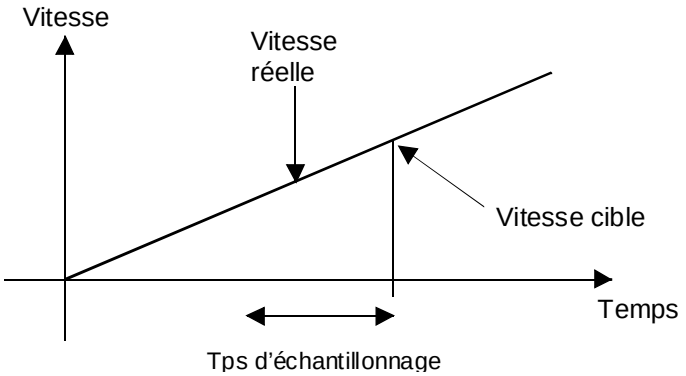
Description :

Cette instruction permet de lire la vitesse cible du système. La lecture correspond à la valeur qui doit être atteinte au cours du prochain cycle d'échantillonnage.

RdTargVel doit être appelée à chaque fois qu'une lecture de la vitesse cible s'impose.

Description des E/S intervenant dans l'instruction :

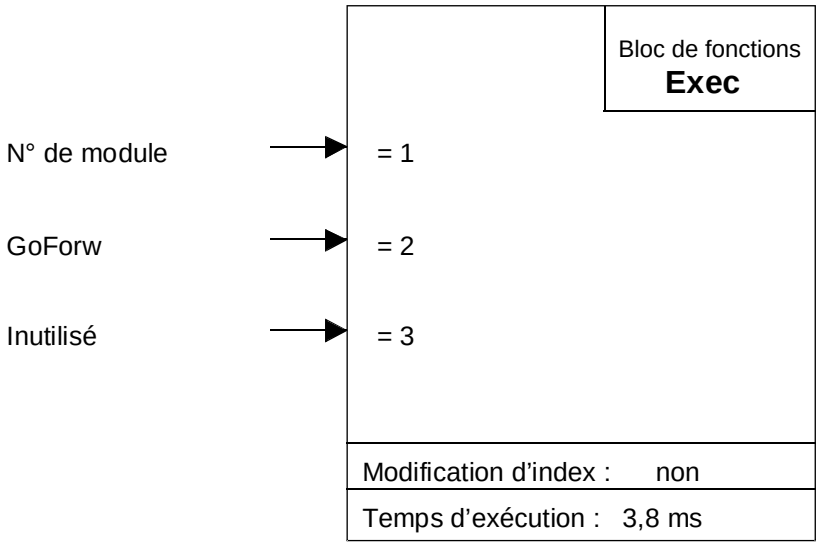
Param.	Désignation/Fonction	Type	Format	Plage	Remarques
= 1	N° de module	K		1 à 16	
= 2	Instruction : RdTargVel				
= 3	Registre PCD contenant la vitesse cible	R	Entier	30 bits	Dépend des données mécaniques.



Représentation de la vitesse cible calculée en unités du temps d'échantillonnage (341 µs)

GoForw Avance

[12]



Description :

Cette instruction lance un déplacement AV, aux vitesse et accélération prédéfinies, mais sans position cible.

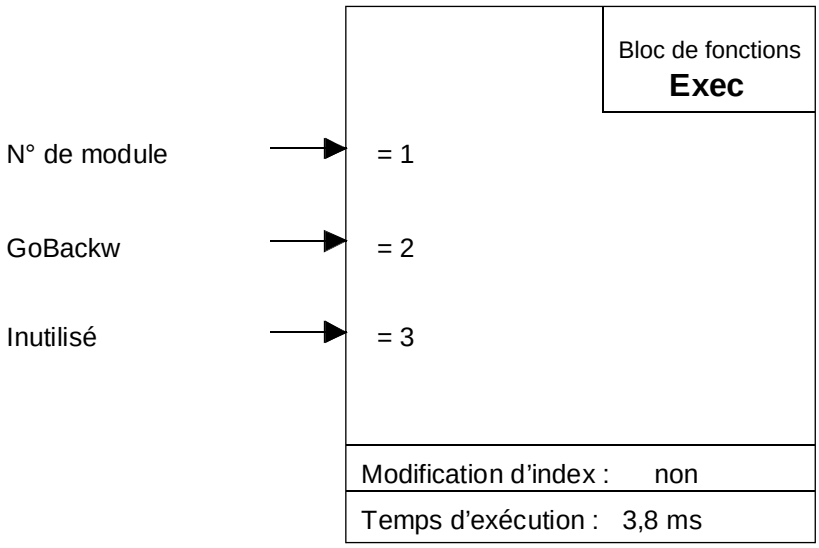
Elle permet en outre de sélectionner le mode Régulation de vitesse de rotation.

Description des E/S intervenant dans l'instruction :

Param.	Désignation/Fonction	Type	Format	Plage	Remarques
= 1	N° de module	K		1 à 16	
= 2	Instruction : GoForw				
= 3	Registre vide du PCD ou « rNotUsed »	R			

GoBackw Recul

[13]



Description :

Cette instruction lance un déplacement AR, aux vitesse et accélération prédéfinies, mais sans position cible.

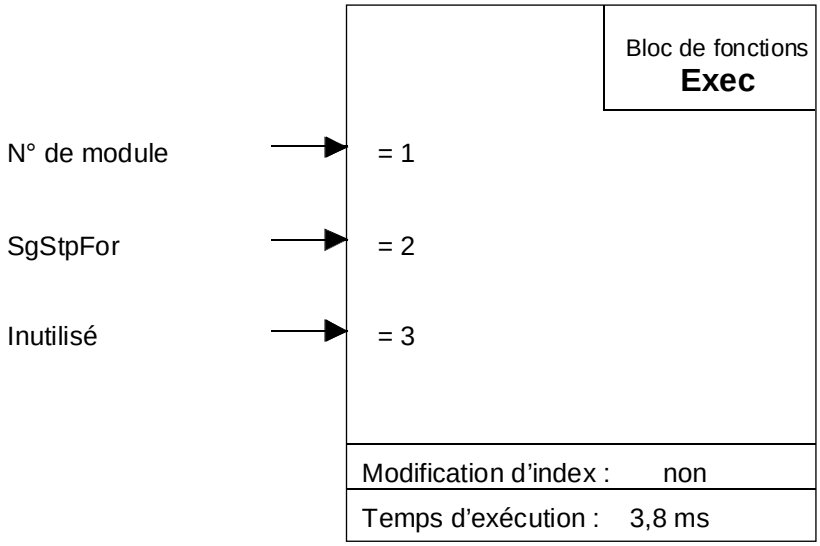
Elle permet en outre de sélectionner le mode Régulation de vitesse de rotation.

Description des E/S intervenant dans l'instruction :

Param.	Désignation/Fonction	Type	Format	Plage	Remarques
= 1	N° de module	K		1 à 16	
= 2	Instruction : GoBackw				
= 3	Registre vide du PCD ou « rNotUsed »	R			

SgStpFor Avance d'un pas

[14]



Description :

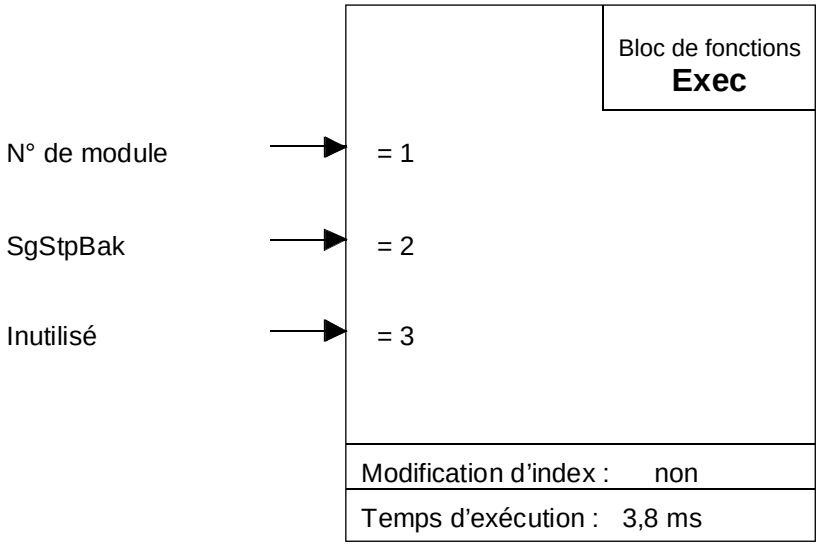
Cette instruction lance un déplacement AV équivalant à un incrément de la résolution du codeur, aux vitesse et accélération prédéfinies.

Description des E/S intervenant dans l'instruction :

Param.	Désignation/Fonction	Type	Format	Plage	Remarques
= 1	N° de module	K		1 à 16	
= 2	Instruction : SgStpFor				
= 3	Registre vide du PCD ou « rNotUsed »	R			

SgStpBak Recul d'un pas

[15]



Description :

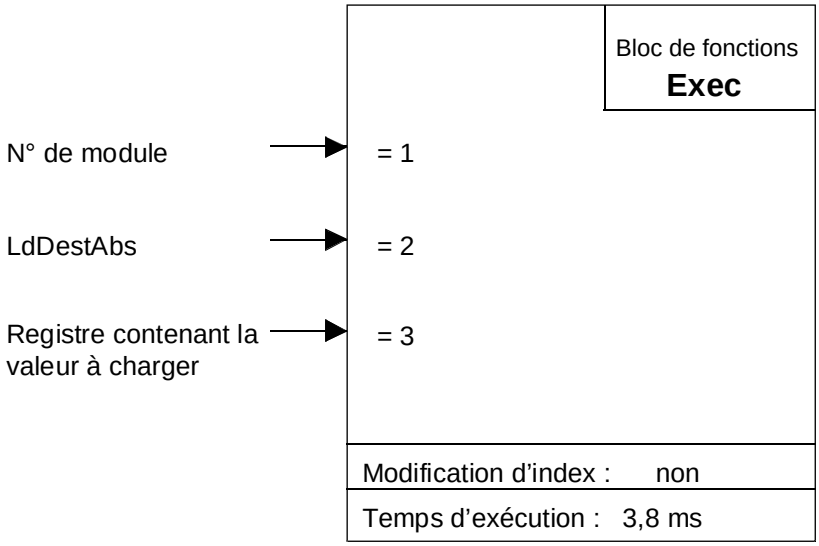
Cette instruction lance un déplacement AR équivalant à un incrément de la résolution du codeur, aux vitesse et accélération prédéfinies.

Description des E/S intervenant dans l'instruction :

Param.	Désignation/Fonction	Type	Format	Plage	Remarques
= 1	N° de module	K		1 à 16	
= 2	Instruction : SgStpBak				
= 3	Registre vide du PCD ou « rNotUsed »	R			

LdDestAbs Chargement de la position cible absolue

[16]



Description :

Cette instruction charge la position cible absolue ; elle doit être suivie d'une instruction « StartMot » (Cf . page A-7) pour lancer la fonction.

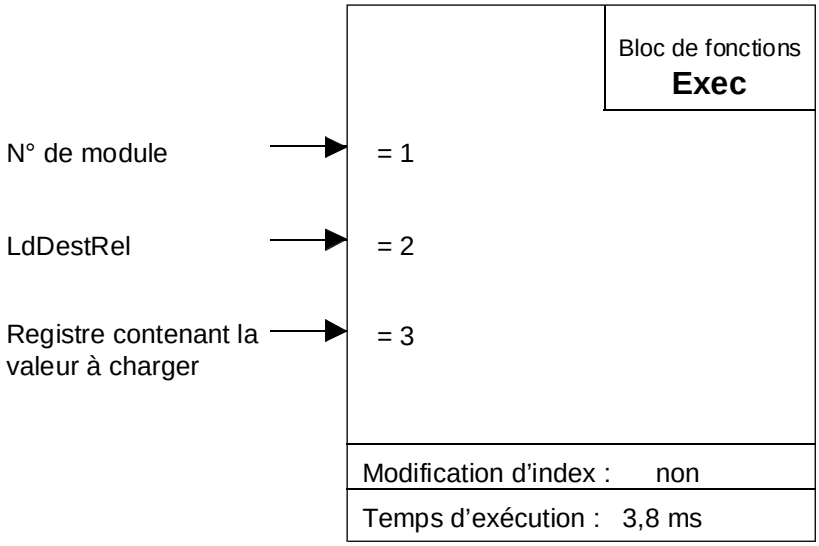
Nota : la position cible absolue se rapporte toujours à la position 0.

Description des E/S intervenant dans l'instruction :

Param.	Désignation/Fonction	Type	Format	Plage	Remarques
= 1	N° de module	K		1 à 16	
= 2	Instruction : LdDestAbs				
= 3	Registre PCD contenant la valeur à charger	R	Entier	31 bits	-2^{30} à $+(2^{30}-1)$
fOnDest_x	Indicateur « atteinte cible »	F			

LdDestRel Chargement de la position cible relative

[17]



Description :

Cette instruction charge la position cible relative ; elle doit être suivie d'une instruction « StartMot » (Cf . page A-7) pour lancer la fonction.

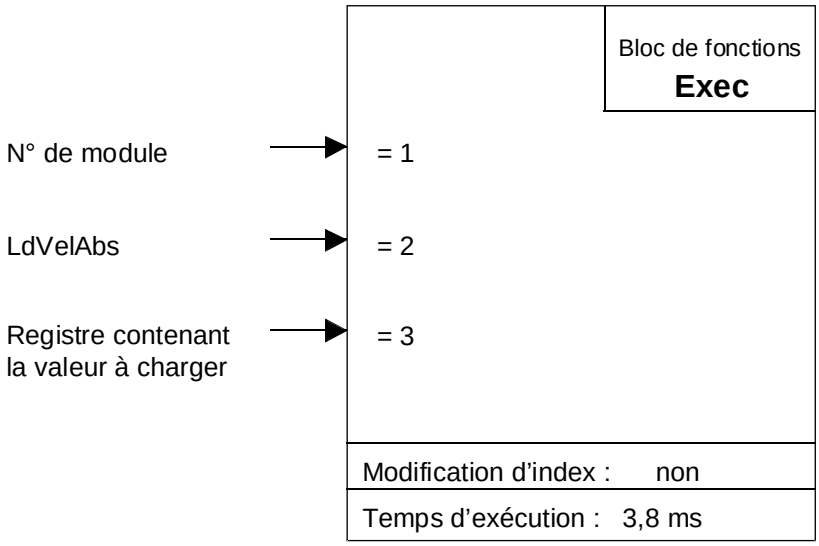
Nota : la position cible relative se rapporte toujours à la position réelle.

Description des E/S intervenant dans l'instruction :

Param.	Désignation/Fonction	Type	Format	Plage	Remarques
= 1	N° de module	K		1 à 16	
= 2	Instruction : LdDestRel				
= 3	Registre PCD contenant la valeur à charger	R	Entier	31 bits	-2^{30} à $+(2^{30}-1)$
fOnDest_x	Indicateur « atteinte cible »	F			

LdVelAbs Chargement de la vitesse absolue

[18]



Description :

Cette instruction permet de charger ou de modifier la vitesse absolue ; elle doit être suivie d'une instruction « StartMot » (Cf . page A-7) pour lancer la fonction.

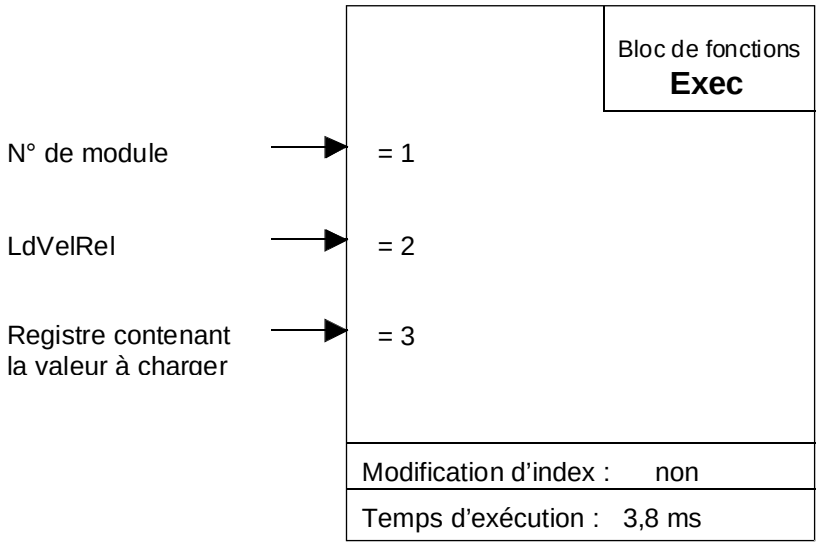
Nota : la vitesse absolue se rapporte toujours à la vitesse nulle.

Description des E/S intervenant dans l'instruction :

Param.	Désignation/Fonction	Type	Format	Plage	Remarques
= 1	N° de module	K		1 à 16	
= 2	Instruction : LdVelAbs				
= 3	Registre PCD contenant la valeur à charger	R	Entier	30 bits	Dépend des données mécaniques.

LdVelRel Chargement de la vitesse relative

[19]



Description :

Cette instruction permet de charger ou de modifier la vitesse relative ; elle doit être suivie d'une instruction « StartMot » (Cf . page A-7) pour lancer la fonction.

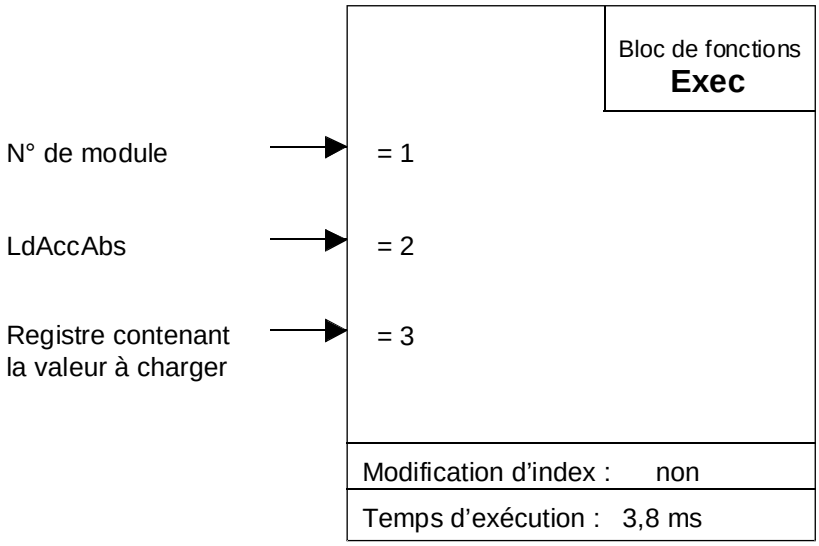
Nota : la vitesse relative se rapporte toujours à la vitesse réelle.

Description des E/S intervenant dans l'instruction :

Param.	Désignation/Fonction	Type	Format	Plage	Remarques
= 1	N° de module	K		1 à 16	
= 2	Instruction : LdVelRel				
= 3	Registre PCD contenant la valeur à charger	R	Entier	30 bits	Dépend des données mécaniques.

LdAccAbs Chargement de l'accélération absolue

[20]



Description :

Cette instruction permet de charger ou de modifier l'accélération absolue ; elle doit être suivie d'une instruction « StartMot » (Cf . page A-7) pour lancer la fonction.

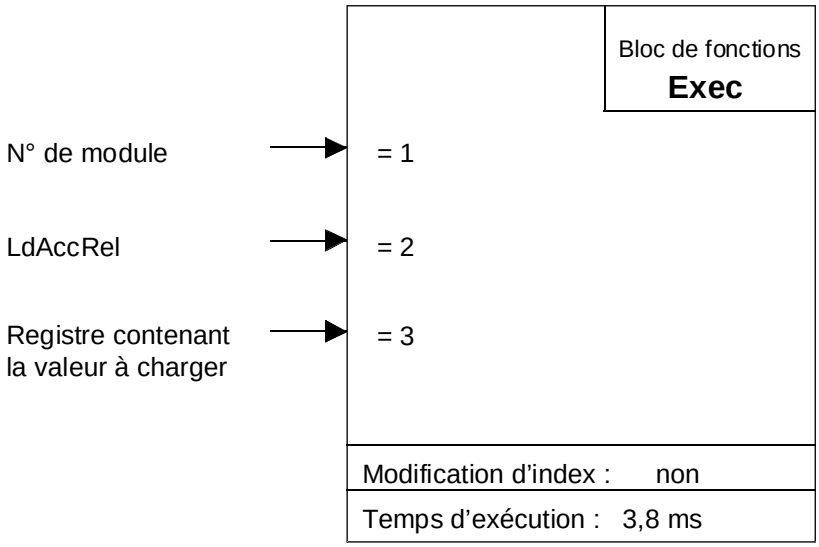
Nota : l'accélération absolue se rapporte toujours à la vitesse ou à l'accélération nulle.

Description des E/S intervenant dans l'instruction :

Param.	Désignation/Fonction	Type	Format	Plage	Remarques
= 1	N° de module	K		1 à 16	
= 2	Instruction : LdAccAbs				
= 3	Registre PCD contenant la valeur à charger	R	Entier	30 bits	Dépend des données mécaniques.

LdAccRel Chargement de l'accélération relative

[21]



Description :

Cette instruction permet de charger ou de modifier l'accélération relative ; elle doit être suivie d'une instruction « StartMot » (Cf . page A-7) pour lancer la fonction.

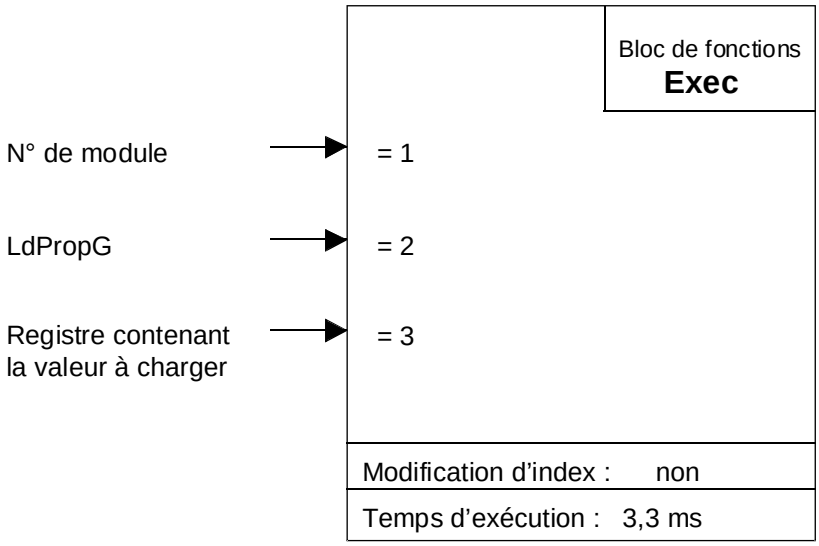
Nota : l'accélération relative se rapporte toujours à l'accélération réelle.

Description des E/S intervenant dans l'instruction :

Param.	Désignation/Fonction	Type	Format	Plage	Remarques
= 1	N° de module	K		1 à 16	
= 2	Instruction : LdAccRel				
= 3	Registre PCD contenant la valeur à charger	R	Entier	30 bits	Dépend des données mécaniques.

LdPropG Chargement du gain proportionnel

[22]



Description :

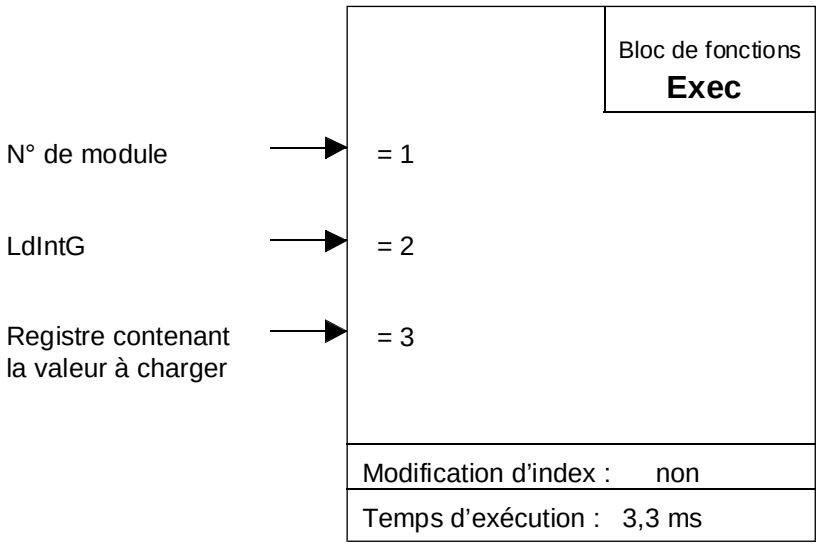
Cette instruction permet de charger ou de modifier l'action Proportionnelle du régulateur PID ; elle doit être suivie de l'instruction « ActRegFact » (Cf. page A-33) pour lancer la fonction.

Description des E/S intervenant dans l'instruction :

Param.	Désignation/Fonction	Type	Format	Plage	Remarques
= 1	N° de module	K		1 à 16	
= 2	Instruction : LdPropG				
= 3	Registre PCD contenant la valeur à charger	R	Entier	15 bits	0 à 32 767

LdIntG Chargement du gain intégral

[23]



Description :

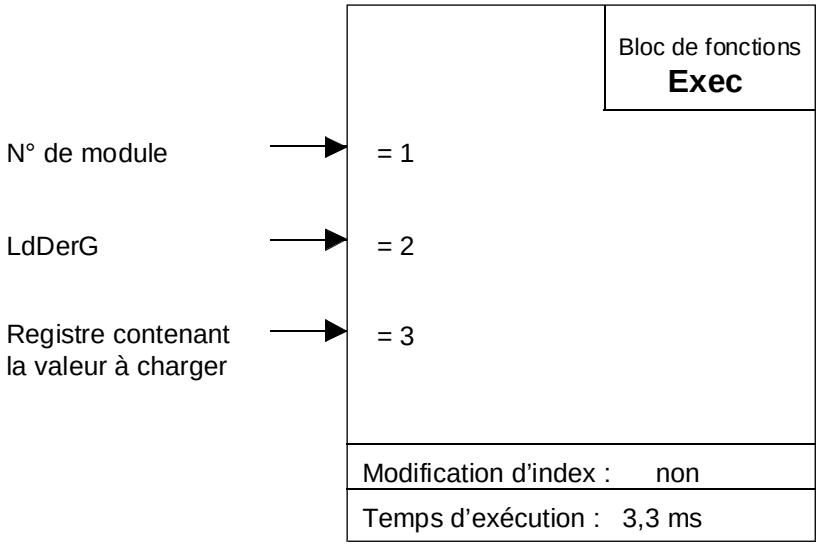
Cette instruction permet de charger ou de modifier l'action Intégrale du régulateur PID ; elle doit être suivie de l'instruction « ActRegFact » (Cf. page A-33) pour lancer la fonction.

Description des E/S intervenant dans l'instruction :

Param.	Désignation/Fonction	Type	Format	Plage	Remarques
= 1	N° de module	K		1 à 16	
= 2	Instruction : LdIntG				
= 3	Registre PCD contenant la valeur à charger	R	Entier	15 bits	0 à 32 767

LdDerG Chargement du gain dérivé

[24]



Description :

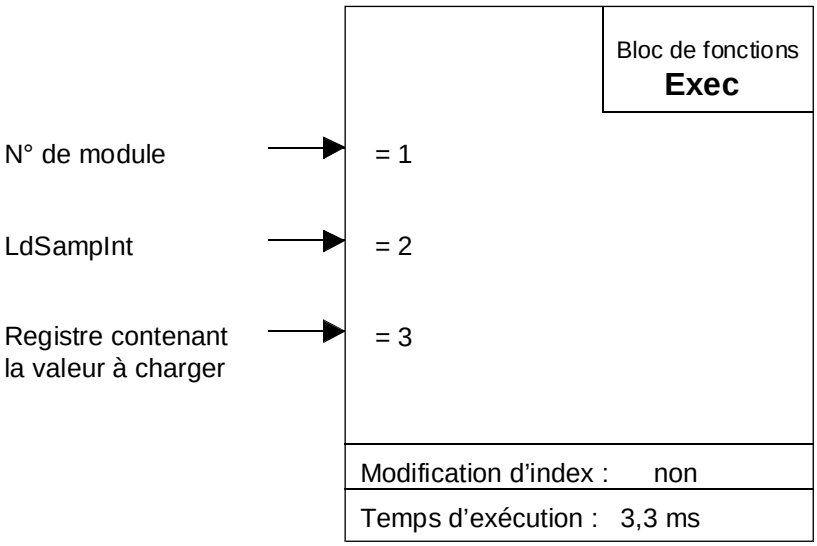
Cette instruction permet de charger ou de modifier l'action Dérivée du régulateur PID ; elle doit être suivie de l'instruction « ActRegFact » (Cf. page A-33) pour lancer la fonction.

Description des E/S intervenant dans l'instruction :

Param.	Désignation/Fonction	Type	Format	Plage	Remarques
= 1	N° de module	K		1 à 16	
= 2	Instruction : LdDerG				
= 3	Registre PCD contenant la valeur à charger	R	Entier	15 bits	0 à 32 767

LdSampInt Chargement du temps d'échantillonnage

[25]



Description :

Cette instruction permet de charger ou de modifier le temps d'échantillonnage du régulateur PID ; elle doit être suivie d'une instruction « ActRegFact » (Cf. page A-33) pour lancer la fonction.

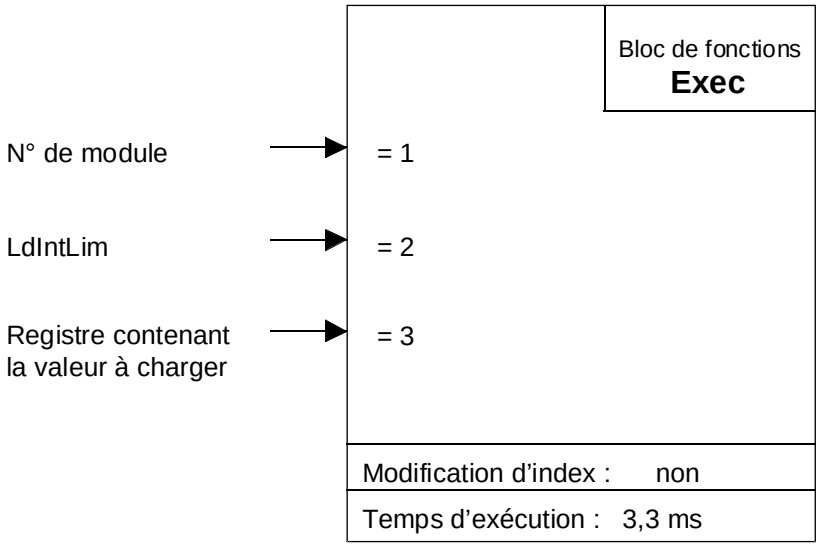
Le temps d'échantillonnage original (341 µs), adapté aux paramètres P et I de la régulation PID, est d'ordinaire trop faible pour la Dérivée ; celle-ci travaille sur l'écart que présente le signal d'erreur entre deux signaux d'échantillonnage. C'est la raison pour laquelle une valeur de 341 µs est insuffisante pour marquer une réelle différence. L'efficacité habituelle de réglage du temps d'échantillonnage se situe de 2 à 9 ms (soit 5 à 25 dans le registre).

Description des E/S intervenant dans l'instruction :

Param.	Désignation/Fonction	Type	Format	Plage	Remarques
= 1	N° de module	K		1 à 16	
= 2	Instruction : LdSampInt				
= 3	Registre PCD contenant la valeur à charger	R	Entier	8 bits	0 à 255

LdIntLim Chargement de la limite d'intégration

[26]



Description :

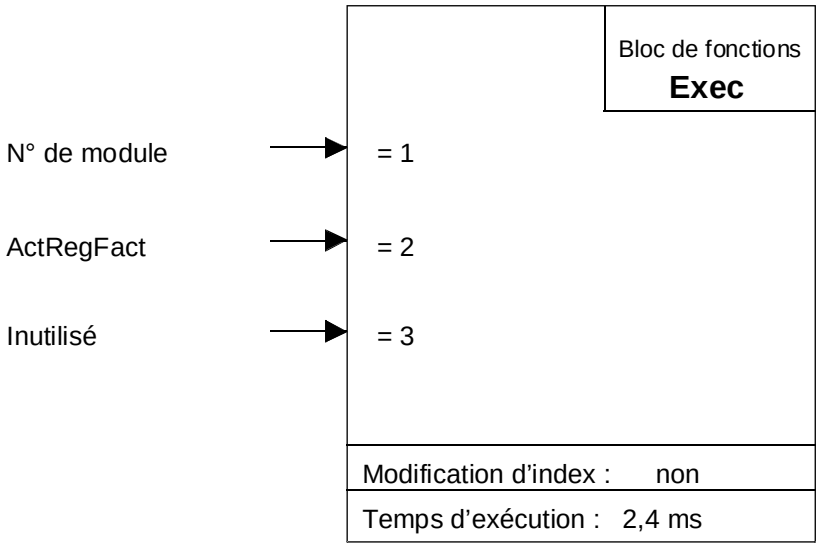
Cette instruction permet de charger ou de modifier la limite d'intégration ; elle doit être suivie de l'instruction « ActRegFact » (Cf. page A-33) pour lancer la fonction.

Description des E/S intervenant dans l'instruction :

Param.	Désignation/Fonction	Type	Format	Plage	Remarques
= 1	N° de module	K		1 à 16	
= 2	Instruction : LdIntLim				
= 3	Registre PCD contenant la valeur à charger	R	Entier	15 bits	0 à 32 767

ActRegFact Activation de la régulation PID

[27]



Description :

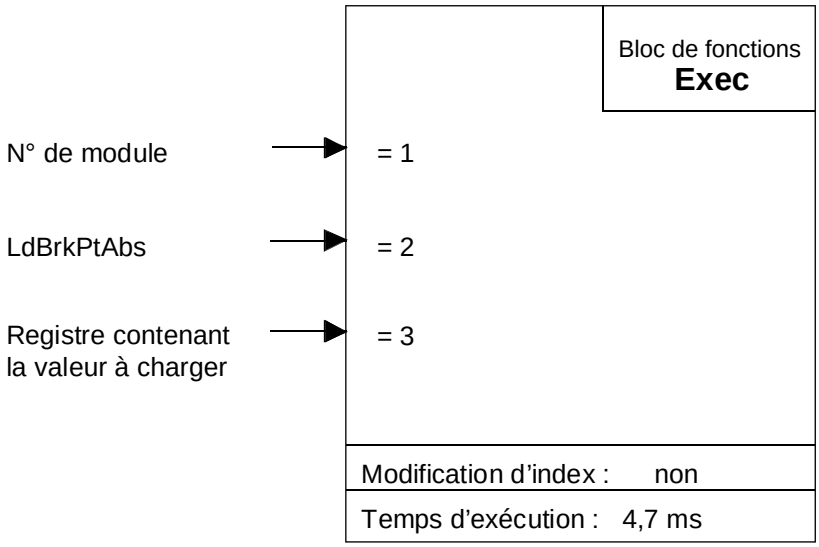
Cette instruction active l'ensemble des trois fonctions PID ; elle peut aussi n'en activer qu'une.

Description des E/S intervenant dans l'instruction :

Param.	Désignation/Fonction	Type	Format	Plage	Remarques
= 1	N° de module	K		1 à 16	
= 2	Instruction : ActRegFact				
= 3	Registre vide du PCD ou « rNotUsed »	R			

LdBrkPtAbs Chargement d'un point d'arrêt absolu

[28]



Description :

Cette instruction permet de charger un point d'arrêt absolu pour signaler une position intermédiaire.

Nota : la position absolue se rapporte toujours à la position 0.

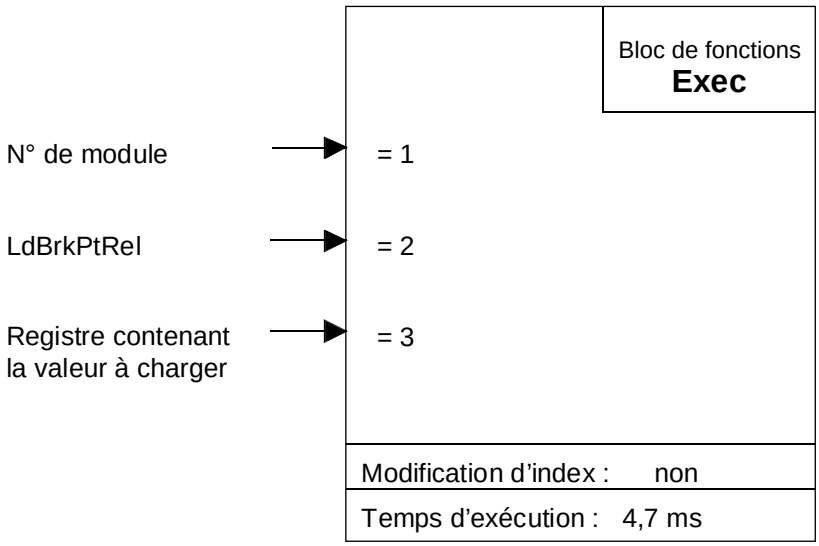
Si ce point d'arrêt est atteint, l'indicateur correspondant « fBrkPt_x » passe à 1 ; ce dernier n'est toutefois positionné qu'après traitement de l'instruction « RdStatRg » (Cf. page A-15).

Description des E/S intervenant dans l'instruction :

Param.	Désignation/Fonction	Type	Format	Plage	Remarques
= 1	N° de module	K		1 à 16	
= 2	Instruction : LdBrkPtAbs				
= 3	Registre PCD contenant la valeur à charger	R	Entier	31 bits	-2^{30} à $+(2^{30}-1)$
fBrkPt_x	Indicateur « atteinte point d'arrêt »	F			Remis à 0 par l'instruction « StartMot » ou « ResStatRg ».

LdBrkPtRel Chargement d'un point d'arrêt relatif

[29]



Description :

Cette instruction permet de charger un point d'arrêt relatif pour signaler une position intermédiaire.

Nota : la position relative se rapporte toujours à la position réelle.

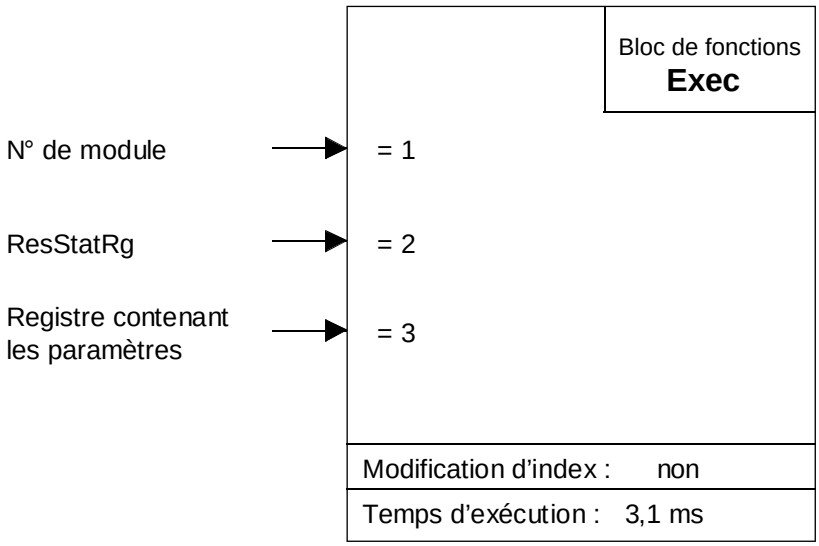
Si ce point d'arrêt est atteint, l'indicateur correspondant « fBrkPt_x » passe à 1 ; ce dernier n'est toutefois positionné qu'après traitement de l'instruction « RdStatRg » (Cf. page A-15).

Description des E/S intervenant dans l'instruction :

Param.	Désignation/Fonction	Type	Format	Plage	Remarques
= 1	N° de module	K		1 à 16	
= 2	Instruction : LdBrkPtRel				
= 3	Registre PCD contenant la valeur à charger	R	Entier	31 bits	-2^{30} à $+(2^{30}-1)$
fBrkPt_x	Indicateur « atteinte point d'arrêt »	F			Remis à 0 par l'instruction « StartMot » ou « ResStatRg ».

ResStatRg Remise à 0 du registre d'état

[30]



Description :

Cette instruction remet à 0 tout ou partie des paramètres du registre d'état.

Paramètres du registre d'état :

bit 7 :	Coupure moteur	128
bit 6 :	Atteinte point d'arrêt	64
bit 5 :	Dépassement erreur de position	32
bit 4 :	Rebouclage	16
bit 3 :	Acquisition impulsion d'index	8
bit 2 :	Fin déplacement	4
bit 1 :	Réservé	2
bit 0 :	Réservé	1

		255

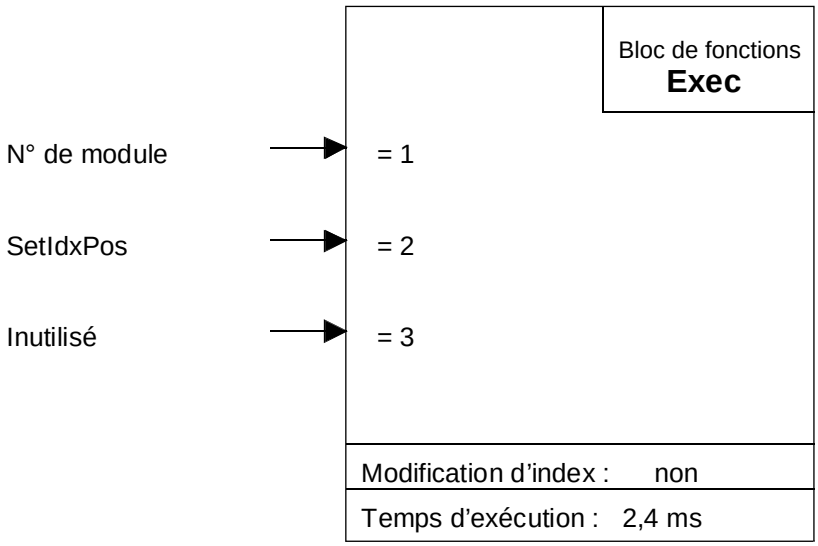
Tout bit positionné sera remis avec la valeur 0 dans le registre du paramètre 3.

Description des E/S intervenant dans l'instruction :

Param.	Désignation/Fonction	Type	Format	Plage	Remarques
= 1	N° de module	K		1 à 16	
= 2	Instruction : ResStatRg				
= 3	Registre PCD contenant l'information de remise à 0	R	Entier	8 bits	0 à 255 255 = tous les bits
fBrkPt_x	Indic. « atteinte pt d'arrêt »	F			
fOnDest_x	Indic. « atteinte cible »	F			
fPosErr_x	Indicateur « dépassement erreur de position »	F			

SetIdxPos Définition de la position d'index

[31]



Description :

Cette instruction charge la position d'index absolue, sur arrivée de l'impulsion codeur suivante, dans le registre d'index du H31x. Cette position est adoptée quand les trois entrées codeur « A », « B » et « IN » (index) sont à 0.

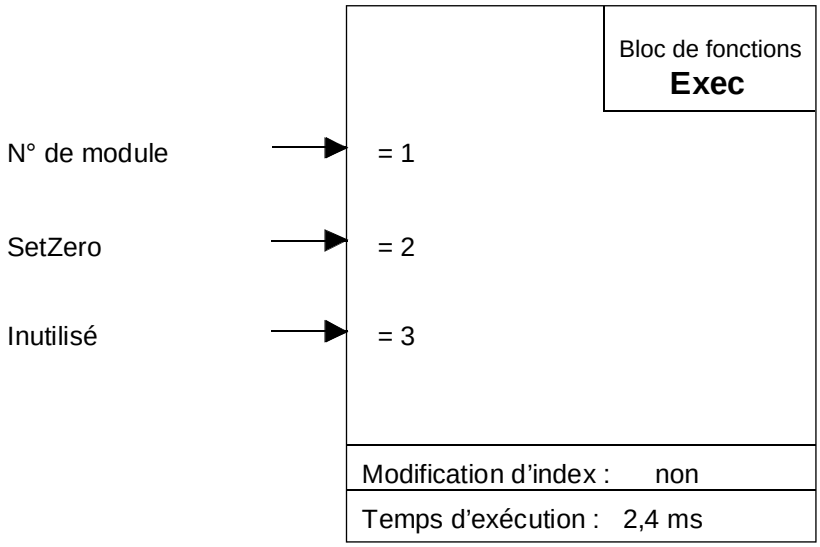
Le bloc Home utilise « SetIdxPos » pour simplifier la définition de la fonction Retour à l'origine.

Description des E/S intervenant dans l'instruction :

Param.	Désignation/Fonction	Type	Format	Plage	Remarques
= 1	N° de module	K		1 à 16	
= 2	Instruction : SetIdxPos				
= 3	Registre vide du PCD ou « rNotUsed »	R			

SetZero Définition de la position 0

[32]



Description :

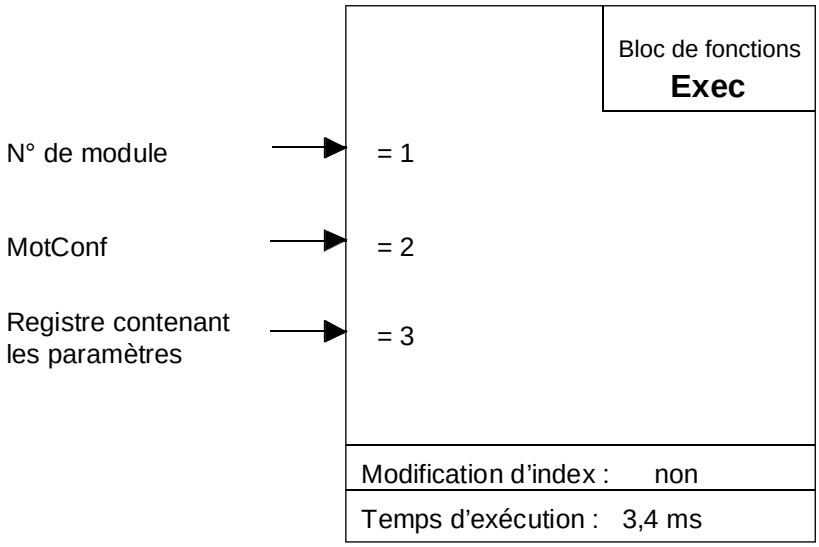
Cette instruction permet de déclarer une position quelconque comme étant la position 0, celle-ci devenant alors la nouvelle position d'origine.

Description des E/S intervenant dans l'instruction :

Param.	Désignation/Fonction	Type	Format	Plage	Remarques
= 1	N° de module	K		1 à 16	
= 2	Instruction : SetZero				
= 3	Registre vide du PCD ou « rNotUsed »	R			

MotConf Configuration du déplacement (choix du mode de régulation)

[33]



Description :

Cette instruction configure le déplacement sur 2 bits :

bit 0 : bas = Régulation de vitesse de rotation
haut = Régulation de position

bit 1 : (réservé à la régulation de position)
bas = Recul
haut = Avance

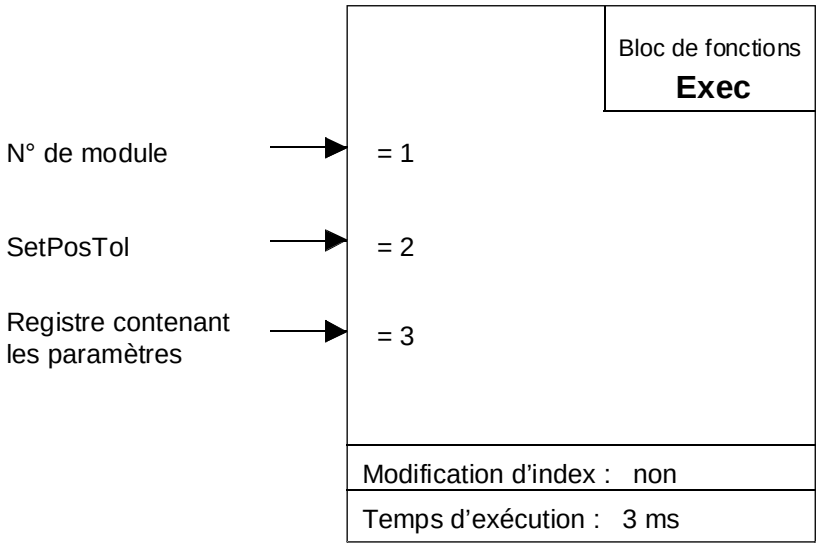
Le mode standard est la régulation de position.

Description des E/S intervenant dans l'instruction :

Param.	Désignation/Fonction	Type	Format	Plage	Remarques
= 1	N° de module	K		1 à 16	
= 2	Instruction : MotConf				
= 3	Registre PCD contenant la configuration	R	Entier	2 bits	0 à 3

SetPosTol Définition de la tolérance de position

[34]



Description :

Cette instruction définit l'erreur de position maximale admissible entre la position réelle et la position cible, calculée par le processeur à chaque période d'échantillonnage.

En cas de dépassement de cette valeur, le comportement du système est dicté par le paramètre 8 du bloc Init : arrêt du moteur ou simple positionnement de l'indicateur d'erreur « fPosErr_x ».

Ce dernier ne peut être pris en compte sans avoir exécuté au préalable l'instruction « RdStatRg » (Cf. page A-15) ; il est remis à 0 à l'instruction « StartMot » suivante (Cf. page A-7).

Description des E/S intervenant dans l'instruction :

Param.	Désignation/Fonction	Type	Format	Plage	Remarques
= 1	N° de module	K		1 à 16	
= 2	Instruction : SetPosTol				
= 3	Registre PCD contenant la tolérance de position	R	Entier	15 bits	0 à 32 767
fPosErr_x	Indicateur « dépassement erreur de position »				

Annexe B Récapitulatif des boîtes de fonctions et des instructions PCD2.H31x en langage FUPLA

(En préparation)

Notes personnelles :

Vos coordonnées :

Société :
Service :
Nom :
Adresse :

Téléphone :

Date :

A renvoyer à :

SAIA-Burgess Electronics SA
Rue de la Gare 18
CH-3280 Morat (Suisse)
http://www.saia-burgess.com

DIV. : Electronic Controllers

Manuel PCD2.H31x

Vos commentaires sont les bienvenus pour améliorer la qualité et le contenu de cette documentation SAIA® PCD. Nous vous remercions par avance de votre collaboration.

Vos commentaires :