GFK-0840A-F

GE Fanuc Manual Series 90-30

Module de commande d'axe APM pour API 90-30
Manuel utillsateur

GE Fanuc Automation

Automates Programmables Industriels

Module de commande d'axe APM pour API 90–30

Manuel utilisateur

GFK–0840A–FR *Mars 1994*

Utilisation de l'expression "Attention danger" et des termes "Attention" et "Remarque" dans ce document

Attention danger

L'expression "Attention danger" est utilisée pour mettre en évidence des risques de blessures dues aux tensions, aux courants, aux températures ou à d'autres grandeurs physiques.

Toutes les situations où un manque d'attention peut être source de blessures physiques ou de dommages pour l'équipement sont repérées par cette expression.

Attention

Le terme "Attention" est associé aux situations où un manque d'attention risque de conduire à des dégâts matériels.

Remarque

Les "Remarques" ont pour but d'attirer votre attention sur des informations particulièrement utiles à la compréhension et à la mise en oeuvre de l'équipement.

Ce document est basé sur des informations disponibles au moment de sa publication. Malgré nos efforts de précision, nous ne pouvons prétendre couvrir tous les détails et toutes les variations matérielles ou logicielles possibles, ni aborder tous les cas de figure de l'installation, du fonctionnement ou de la maintenance. Les caractéristiques décrites dans ce document peuvent être absentes de certains systèmes matériels ou logiciels. GE Fanuc Automation ne s'engage pas à avertir les possesseurs de ce document d'éventuelles modifications ultérieures.

GE Fanuc Automation ne fournit aucune garantie explicite, implicite ou statutaire, et décline toute responsabilité quant à la précision, à l'utilité, et au caractère complet ou suffisant des informations contenues dans ce document. GE Fanuc Automation ne donne aucune garantie de qualité marchande et d'aptitude à une utilisation donnée.

Les marques suivantes sont des marques déposées de GE Fanuc Automation North America, Inc. :

Alarm Master	CIMSTAR	Helpmate	PROMACRO	Series Six
CIMPLICITY	GEnet	Logicmaster	Series One	Series 90
CIMPLICITY 90-ADS	Genius	Modelmaster	Series Three	VuMaster
CIMPLICITY PowerTRAC	Genius PowerTRAC	ProLoop	Series Five	Workmaster

1. CONTENU DE CE MANUEL

Chapitre 1. **Présentation :** Ce chapitre présente le matériel et les logiciels utilisés pour configurer et exploiter un servomécanisme.

Chapitre 2. **Installation du module APM30 :** Ce chapitre décrit toutes les interfaces utilisateur du module APM30, ainsi que l'installation du module sur la platine d'un API Série 90–30.

Chapitre 3. **Configuration du module APM30 :** Ce chapitre explique comment configurer le module APM30 en utilisant le progiciel de configuration Logicmaster.

Chapitre 4. **Transferts automatiques de données :** Ce chapitre décrit les données %I, %AI, %Q et %AQ transférées entre l'UC de l'API et le module APM30 à chaque cycle.

Chapitre 5. **Procédures de démarrage d'un servomécanisme :** Ce chapitre explique les procédures à suivre pour mettre en oeuvre un servomécanisme.

Chapitre 6. **Commandes de mouvement du module APM :** Ce chapitre fournit des exemples pratiques illustrant la programmation de différents types de mouvements, de sauts, etc.

Annexe A. **Codes d'erreur**

Annexe B. **Téléchargement de paramètres avec COMM_REQ**

Annexe C. **Spécifications**

Annexe D. **Informations de commande**

2. AUTRES MANUELS À CONSULTER

GFK–0664 Series 90 PLC Axis Positioning Module (APM30) Programmer's Manual

GFK–0781 Series 90–30 PLC Axis Positioning Module (APM30) – Follower Mode User's Manual

3. VOS REMARQUES ET SUGGESTIONS SONT LES BIENVENUES

GE Fanuc Automation s'efforce d'éditer des documentations techniques de qualité. Après avoir utilisé ce manuel, merci de consacrer quelques instants à la page suivante, "Page de remarques", pour la compléter et nous la renvoyer.

Page laissée blanche intentionnellement

Page de remarques

GFK–0840A–FR	***Manuel utilisateur du module de commande d'axe APM pour API 90–30***

Cochez votre fonction principale SVP

- ☐ Concepteur système
- ☐ Distributeur
- ☐ Intégrateur système
- ☐ Installateur
- ☐ Programmeur
- ☐ Responsable de maintenance
- ☐ Opérateur
- ☐ Autre (à préciser ci–dessous)

Si vous désirez une réponse personnelle, indiquez votre adresse postale complète :

SOCIETE : . . . . . NOM : . . . . . .

ADRESSE : . . . . . .

. . . . . PAYS : . . . . . .

Remettez cet imprimé directement à votre correspondant GE Fanuc ou envoyez–le à :

GE Fanuc Automation France
45, rue du Bois Chaland
CE 2904 – Lisses
91029 EVRY Cedex

Toutes vos remarques seront étudiées par du personnel qualifié.

REMARQUES

Si besoin, utilisez le verso de cette page.

Page de remarques

Sommaire

Page

CHAPITRE 1 – PRÉSENTATION

CHAPITRE 2 – INSTALLATION DU MODULE APM

Sommaire

Sommaire

Page laissée blanche intentionnellement

Tableaux

Figures

Page

Chapitre 1 Présentation

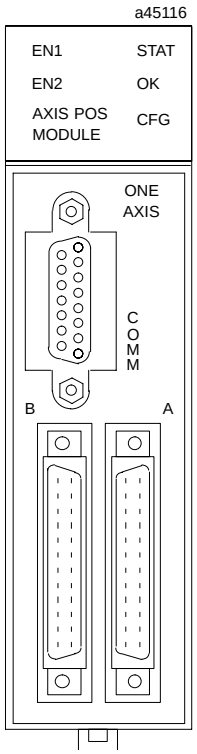
Le module APM est un module de commande d'axe, pour 1 ou 2 axes, de haute performance et facile à utiliser. Il est fortement intégré avec les fonctions de résolution logique et de communication de l'API Série 90–30.

Le module APM fonctionne en deux configurations de boucle de contrôle primaire :

- Mode Standard (axes indépendants)
- Mode Suiveur

Ce manuel décrit le module APM en configuration de boucle de contrôle *Standard*. Pour obtenir des informations sur le fonctionnement du module APM en mode de boucle de contrôle Suiveur, reportez–vous au document *GFK–0781 Series 90–30 PLC Axis Positioning Module (APM30) – Follower Mode User's Manual.*

1. CARACTÉRISTIQUES DU MODULE APM



1.1. HAUTE PERFORMANCE

- Mise à jour rapide de boucle d'asservissement (1 ms pour APU301, 2 ms pour APU302).
- Temps de traitement de bloc inférieur à 5 ms.
- Anticipation de vitesse et intégration d'erreur de position pour augmenter la précision du suivi.
- Unités de programmation de haute résolution.
  Position : de –8 388 608 à +8 388 607 unités utilisateur
  Vitesse : de 0 à 8 388 607 unités utilisateur/s
  Accélération : de 0 à 134 217 727 unités utilisateur/s/s

1.2. FACILITÉ D'UTILISATION

- Jeu d'instructions simple et puissant pour les programmes de mouvement.
- Programmes de mouvement pour 1 ou 2 axes avec synchronisation de début de bloc.

- Le programme peut être un court programme de mouvement créé dans le logiciel de configuration Logicmaster 90.
- Mémoire non–volatile pour le stockage de 10 programmes et 40 sous–programmes.
- Ecran d'état dans le logiciel Motion Programmer pour la surveillance des servocommandes.
- Mise à l'échelle par l'utilisateur des unités de programmation (unités utilisateur).
- Programmation générique avec des paramètres de commande utilisés comme opérandes pour les commandes d'accélération, de vitesse, de mouvement et d'arrêt momentané.
- Configuration avec le logiciel Logicmaster 90.
- Transfert de données automatique entre les tables de l'API et l'APM sans programmation utilisateur.
- Connexion des E/S aisée grâce aux câbles d'usine et aux borniers standard, ainsi qu'à un port série pour la connexion des consoles de programmation.

1.3. E/S POLYVALENTES

- Sortie analogique de commande de vitesse Ç 10 Volts.
- Entrée analogique 12 bits plus signe.
- Entrées de commutateurs de position initiale et de dépassement.
- Entrée d'échantillonnage pour l'acquisition de position.
- Entrées (8) et sorties (4) de commande définissables par l'utilisateur.
- Retour de codeur (jusqu'à 250 kHz par voie).

2. TABLEAU DE COMPATIBILITÉ DES MICROPROGRAMMES

Numéro de révision du microprogramme	Référence produit	Capacité d'axes/de commande	Documentation APM	Version de Logicmaster 90	Logiciel Motion Programmer	
					Version	Document
1.50	IC693APU302A–B	2 axes Suiveurs	GFK–0781A	3.50 et ultérieures	1.50 et ultérieures	GFK–0664
2.01, 2.02	IC693APU302C–D	2 axes Indépendants/ Suiveurs	GFK–0840A GFK–0781A	4.01 et ultérieures	1.50 et ultérieures	GFK–0664A
2.02	IC693APU301E	1 axe Indépendant/ Suiveur	GFK–0840A GFK–0781A	4.01 et ultérieures	1.50 et ultérieures	GFK–0664A

Remarques relatives à l'écran de la version 1.01 du logiciel Motion Programmer pour les révisions 1.50 et ultérieures du microprogramme de l'APM :

- Affiche uniquement les données de l'axe 1 pour un APM à 2 axes.
- N'affiche pas correctement les données transmises par un APM en mode Suiveur.
- N'affiche pas correctement les données relatives à une erreur de position signalée par l'APM mono–axe en mode Standard.

3. PRÉSENTATION DU FONCTIONNEMENT DU MODULE APM

Le module APM est un module optionnel de contrôle de mouvement intelligent et entièrement programmable pour l'automate programmable industriel (API) Série 90–30. Le module APM permet à l'utilisateur d'un API de combiner un contrôle de mouvement de haute performance et les fonctions de résolution logique de l'API en un seul système intégré. La schéma ci–dessous présente le matériel et les logiciels nécessaires pour configurer et utiliser un servomécanisme. Ce paragraphe décrit brièvement chacune des composantes du système pour fournir une vue d'ensemble du fonctionnement du système.

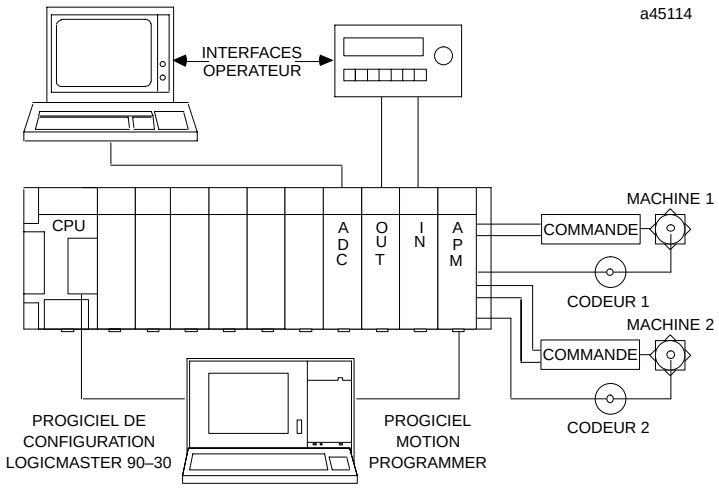


Figure 1-1. Matériel et logiciels utilisés pour configurer, programmer et exploiter un servomécanisme de module APM

Le système présenté ci–dessus peut être réparti en quatre catégories.

1. Le module APM et l'API Série 90–30
2. Le module APM et le servomécanisme
3. Les progiciels de programmation de mouvement
4. Les interfaces opérateur

La connaissance des termes suivants, relatifs au module APM, vous sera très utile.

Vitesse commandée (Commanded Velocity) : vitesse commandée par le générateur de trajectoire interne du module APM.

Position commandée (Commanded Position) : position commandée par le générateur de trajectoire interne du module APM.

Vitesse réelle (Actual Velocity) : vitesse de l'axe indiquée par le retour.

Position réelle (Actual Position) : position de l'axe indiquée par le retour.

Erreur de position (Position Error) : différence entre la position commandée et la position réelle.

Constante de temps de boucle de position (Position Loop Time Constant) : valeur configurable déterminant le temps de réponse de la boucle de position fermée.

3.1. MODULE APM ET API SÉRIE 90–30

Le module APM et l'API Série 90–30 fonctionnent en synergie pour constituer un système de contrôle de mouvement intégré. Le module APM communique avec l'API par l'interface du fond de bac. A chaque cycle de l'API, le module APM transfère des données telles que la *Vitesse commandée* et la *Position réelle* à l'API sous forme de données %I et %AI. De même, à chaque cycle, l'API transfère des données %Q et %AQ vers le module APM. Les données %Q et %AQ sont utilisées pour contrôler le module APM. Les bits %Q correspondent à des fonctions telles que la mise en mouvement, l'arrêt du mouvement et la mise à zéro des indicateurs d'échantillonnage. Les commandes %AQ correspondent à des fonctions telles que l'initialisation de la position et le chargement de registres de paramètres.

Vous pouvez utiliser les données de référence %I, %AI, %Q et %AQ pour transmettre des paramètres au module APM. Vous pouvez également utiliser des blocs fonctionnels COMM_REQ. Vous trouverez des détails sur le téléchargement de blocs de paramètres au chapitre 6, Contrôle du mouvement du module APM.

L'API permet également de configurer aisément le module APM en utilisant le logiciel de configuration Logicmaster 90–30. Comme les autres modules de l'API, le module APM est affecté à un emplacement et à un bac particuliers. D'autre part, vous devez entrer plusieurs autres types de données de configuration telles que :

- Les adresses d'E/S qui serviront aux transferts UC–APM,
- La configuration du port série utilisé pour connecter la console de programmation,
- Les données de configuration du module APM,
- Un programme court (20 lignes au maximum) créé avec l'éditeur Programme Zéro.

Les données de configuration du module APM sont décrites au chapitre 3, Configuration du module APM.

3.2. MODULE APM ET SERVOMÉCANISME

Le module APM communique avec le servomoteur et le dispositif de régulation de position par les connecteurs A et B. Il contrôle un axe en transmettant une tension analogique à une servocommande. Il interprète les déplacements de la servocommande, ou axe, en analysant les signaux d'un dispositif de régulation de position (codeur).

L'erreur de position est l'un des facteurs de contrôle de la fermeture de la boucle de position. Le module APM produit une sortie de commande de vitesse proportionnelle à l'erreur de position.

Le module APM comporte également des entrées de commutateur de position initiale, des entrées de commutateur de dépassement de limites, et d'autres entrées et sorties de contrôle.

3.3. PROGICIELS DE PROGRAMMATION DE MOUVEMENT

Les programmes de mouvement sont généralement créés avec le progiciel Motion Programmer, mais vous pouvez créer un programme court (éditeur Programme Zéro) en utilisant le logiciel de configuration Logicmaster 90–30 décrit plus haut.

Logiciel Motion Programmer. Le module APM comporte un port SNP configurable qui lui permet de communiquer avec le logiciel Motion Programmer. Le logiciel fonctionne en grande partie comme le progiciel Logicmaster 90. Il offre la possibilité d'écrire des programmes de mouvement avec des instructions en anglais, de les stocker sur disque et de les télécharger dans le module APM. Un écran d'état fournit la surveillance du servomécanisme. Vous trouverez à la fin de ce chapitre une présentation des fonctions du logiciel Motion Programmer. Pour plus d'informations sur le logiciel Motion Programmer, reportez–vous au document *GFK–0664 Series 90 PLC Axis Positioning Module (APM30) Programmer's Manual*.

Editeur Programme Zéro. Le logiciel de configuration Logicmaster 90 contient un éditeur appelé Programme Zéro qui permet de créer un programme court (au maximum 20 commandes). Les commandes utilisées sont en anglais, dans un format identique à celui des commandes du logiciel Motion Programmer. Aucune programmation en langage relais n'est nécessaire. L'éditeur Programme Zéro est décrit en détail au chapitre 3, *Configuration du module APM*.

3.4. INTERFACES OPÉRATEUR

Les interfaces opérateur permettent à l'opérateur de contrôler et de surveiller le servomécanisme par un panneau de contrôle ou un affichage sur écran. Ces interfaces communiquent avec l'API par des modules d'E/S logiques ou un module d'API Série 90 intelligent tel qu'un CIMPLICITY 90–ADS ou un module coprocesseur programmable (PCM).

Les données opérateur sont transférées automatiquement entre l'API et le module APM par les références %I, %AI, %Q et %AQ spécifiées lors de la configuration du module. Ce transfert de données automatique fournit une interface souple et simple pour de nombreuses interfaces opérateur, qui s'ajoutent au logiciel Motion Programmer.

4. PRÉSENTATION DU LOGICIEL MOTION PROGRAMMER

Ce paragraphe fournit un récapitulatif de la programmation avec le logiciel Motion Programmer. Pour une description complète de la programmation du module APM, reportez–vous au document *GFK–0664 Series 90 PLC Axis Positioning Module (APM30) Programmer's Manual*.

Le logiciel Motion Programmer est un puissant outil utilisant des instructions en anglais et destiné à la programmation du module APM. Vous pouvez l'exécuter de façon autonome ou à partir du menu de démarrage Logicmaster. Pour développer des programmes et des sous–programmes, vous devez sélectionner des commandes en utilisant des touches de fonction et en entrant des valeurs dans les champs d'opérandes pour compléter les instructions.

4.1. FONCTIONS DU LOGICIEL MOTION PROGRAMMER

- Création et édition de programmes et de sous–programmes
- Configuration des ports de la console de programmation
- Chargement et transfert de fichiers de programme entre la console de programmation et le module APM
- Impression des programmes de mouvement, des sous–programmes, et des informations de configuration

4.2. FONCTIONS DE L'ÉDITEUR DE PROGRAMMES

- Insertion de lignes de programme
- Edition de lignes de programme
- Suppression de lignes de programme
- Vérification de la syntaxe du programme
- Renumérotation des blocs
- Fonction GOTO

Le schéma ci–dessous présente l'écran de l'éditeur et des exemples d'instructions.

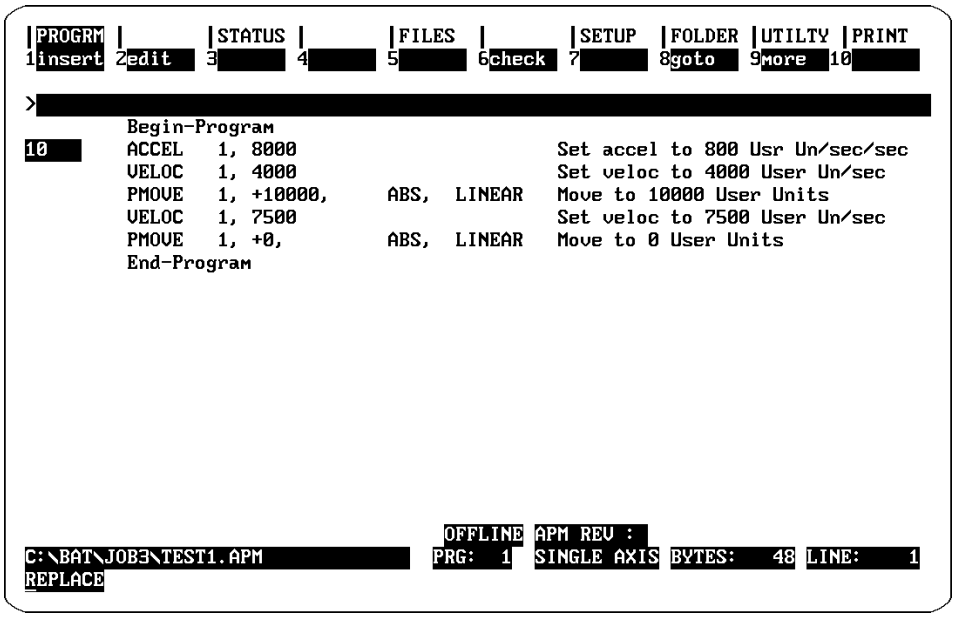


Figure 1-2. Exemple d'écran de l'éditeur du logiciel Motion Programmer

4.3. FORMAT DES INSTRUCTIONS DU LOGICIEL MOTION PROGRAMMER

Une instruction de programme ou de sous–programme se compose d'une commande et de données associées décrivant la commande. La commande et ses données doivent être saisies dans des zones particulières de l'écran de l'éditeur de programmes appelées champs. Ces champs sont, de gauche à droite :

Numéro de bloc. Ce champ est *optionnel*. Il permet la saisie d'un nombre repérant le début d'un bloc d'instructions de programme. Il est utilisé comme destination par la commande *Jump* et fournit un point de synchronisation pour les programmes à deux axes.

Spécification de bloc de synchronisation. Ce champ optionnel apparaît uniquement pendant l'édition de programmes multi–axes. Un "X" indique un bloc de synchronisation.

Nom de la commande. Ce champ est *obligatoire*. Il contient une désignation de la commande (en anglais). Les commandes sont sélectionnées à partir des touches de fonction correspondant à celles situées en haut de l'écran de l'éditeur.

Opérande. Tous les champs d'opérandes affichés sont *obligatoires*. Lorsque vous saisissez un nom de commande, le logiciel affiche les champs d'opérandes correspondants avec leurs valeurs par défaut. Le nombre de champs d'opérandes affichés varie de un à quatre en fonction du nom de la commande. Le champ d'opérande 1 est réservé au numéro d'axe. Le contenu des trois autres champs dépend de la commande.

Commentaire. Ce champ est *optionnel*. Il permet la saisie d'un commentaire d'environ 30 caractères pour chaque instruction du programme. Vous pouvez également ajouter un commentaire d'une longueur maximale de 80 caractères sur une ligne distincte.

Tableau 1-1. Commandes du logiciel Motion Programmer

Nom de la commande	Opérande 1	Opérande 2	Opérande 3	Opérande 4
Accélération (ACCEL)	N° axe (1, 2)	Unités utilisateur/s/s (1 – 134217727) ou numéro de paramètre (0 – 255)	n/a	n/a
Appel sous–programme (CALL)	N° axe (1, 2)	Numéro de sous–programme (1–40)	n/a	n/a
Mouvement continu (CMOVE)	N° axe (1, 2)	Unités utilisateur (–8388608...+8388607)* ou numéro de paramètre (0 – 255)	Mode de positionnement (ABS, INCR)	Mode d'accélération (LINEAR, SCURVE)
Saut (JUMP)	n/a	Numéro de CTL (CTL01–CTL12, inconditionnel)	Numéro de bloc (1–65535)	n/a
Arrêt momentané (DWELL)	N° axe (1, 2)	Millisecondes (0–60000) ou numéro de paramètre (0 – 255)	n/a	n/a
Chargement paramètre (LOAD)	n/a	Numéro de paramètre (0 – 255)**	Valeur de paramètre (32 bits maximum)	n/a
Aucune action (NULL)	n/a	n/a	n/a	n/a
Mouvement de positionnement (PMOVE)	N° axe (1, 2)	Unités utilisateur (–8388608...+8388607)* ou numéro de paramètre (0 – 255)	Mode de positionnement (ABS, INCR)	Mode d'accélération (LINEAR, SCURVE)
Vitesse (VELOC)	N° axe (1, 2)	Unités utilisateur/s (1 – 8388607) ou numéro de paramètre (0 – 255)	n/a	n/a
Attente (WAIT)	N° axe (1, 2)	Numéro de CTL (CTL01 – CTL12)	n/a	n/a

* Pour les mouvements absolus (ABS), vous devez spécifier une position comprise entre la fin de course **ET** les limites de comptage. Faute de quoi, une erreur se produira lors de l'exécution de la commande de mouvement.

** Les paramètres 246 à 255 sont réservés à une utilisation spéciale. La fonction de chargement de paramètres ne permet pas de charger de données dans ces paramètres, mais le module APM peut écraser leur valeur avec ses propres données.

Numéro de paramètre	Fonction spéciale
246 – 253	Réservé à une utilisation future
254	Stocke la valeur de la *position d'échantillonnage* de l'axe 2 (302 uniquement, unités utilisateur)
255	Stocke la valeur de la *position d'échantillonnage* de l'axe 1 (unités utilisateur)

Page laissée blanche intentionnellement

Chapitre 2 Installation du module APM

Ce chapitre décrit les modèles à 1 et 2 axes du module APM et leur installation sur la platine de l'API Série 90–30. Il comprend les paragraphes suivants :

- Description des modules APM à 1 et 2 axes
- Installation du module APM

1. DESCRIPTION DU MODULE APM

Ce paragraphe décrit les interfaces utilisateur des modules APM à 1 et 2 axes en mode standard.

1.1. VOYANTS

Le module APM comporte 5 voyants fournissant des indications sur son état. Ces voyants sont :

Status. Normalement allumé. Clignote pour indiquer une erreur de fonctionnement. Clignote lentement (4 fois/s) pour les erreurs d'état. Clignote rapidement (8 fois/s) pour les erreurs entraînant un arrêt de la servocommande.

OK. Le voyant OK du module APM indique l'état courant de la carte du module.

- Allumé : lorsque le voyant est allumé, le module APM fonctionne correctement. Ce voyant devrait normalement toujours être allumé.
- Eteint : lorsque le voyant est éteint, le module APM ne fonctionne pas. Ceci résulte d'un mauvais fonctionnement matériel ou logiciel.

CFG. Ce voyant est allumé lorsque le module a reçu une configuration valide de l'API. Il clignote lentement (4 fois/s) pendant l'exécution de la fonction d'enregistrement du programme de mouvement. Il clignote rapidement (8 fois/s) pendant l'opération d'écriture de la RAM utilisateur en EEPROM.

EN1. Lorsque ce voyant est allumé, le servomoteur de l'axe 1 est activé.

EN2. Lorsque ce voyant est allumé, le servomoteur de l'axe 2 est activé. Sur un APM mono–axe, ce voyant est allumé uniquement lorsqu'une commande *Forçage N/A immédiat* est utilisée sur la sortie analogique du connecteur B.

1.2. CONNECTEUR DE COMMUNICATION SÉRIE

La façade du module APM comporte un connecteur femelle de type D 15 broches pour les communications série. Ce connecteur permet de relier l'ordinateur exécutant le logiciel Motion Programmer au module APM. Le port du module APM utilise le protocole SNP et est compatible RS–485. Vous pouvez sélectionner un débit compris entre 300 et 19200 bauds.

La connexion entre l'ordinateur de programmation et le module APM est généralement réalisée entre le port RS–232 de l'ordinateur et le connecteur de communication série par un convertisseur RS–232 vers RS–485/RS–422. GE Fanuc propose pour cela un kit miniconvertisseur (IC690ACC901) comprenant un convertisseur et un câble de 2 m.

Le port est configuré avec le progiciel de configuration Logicmaster.

Vous trouverez ci–dessous les définitions des broches du port série du module APM.

Tableau 2-1. Définition des broches du connecteur de communication série

Broche	Signal	Description	Broche	Signal	Description
1	Blindage	Blindage	9	RT	Terminaison 120 Ohm pour RXD (A)
2	DCD (A)	Détection de porteuse	10	RD (A)	Réception de données
3	DCD (B)	Détection de porteuse	11	RD (B)	Réception de données
4	ATCH	Entrée d'attache de la miniconsole HHP	12	SD (A)	Emission de données
5	+5 V	Alimentation +5 V du convertisseur HHP/422–232	13	SD (B)	Emission de données
6	RTS (A)	Demande pour émettre	14	RTS (B)	Demande pour émettre
7	0 V	Commun de signal 0 V	15	CTS (A)	Prêt à émettre
8	CTS (B)	Prêt à émettre			

Connexion multipoint :

Vous pouvez connecter les modules APM en mode multipoint (voir exemple dans la figure ci–dessous). Chaque module APM présent dans le système nécessite un câble.

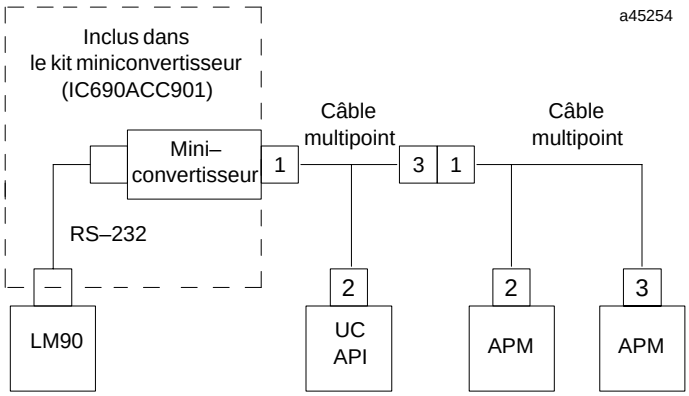


Figure 2-1. Connexion de modules APM en configuration multipoint

Le câble multipoint doit être conforme au schéma suivant.

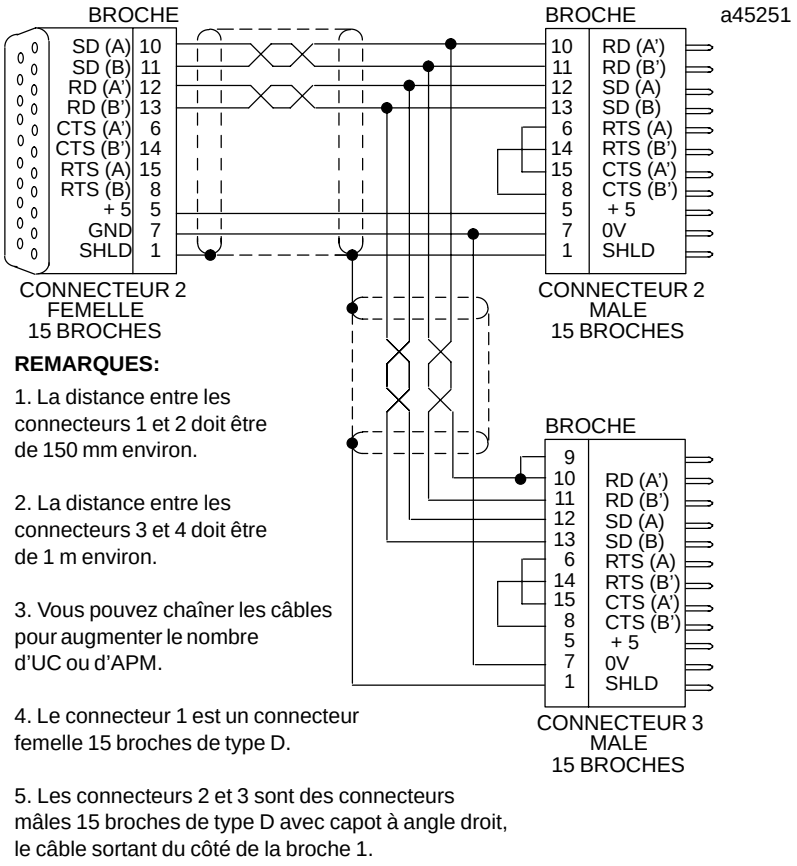


Figure 2-2. Câble multipoint pour le module APM

1.3. CONNECTEURS D'E/S

La façade du module APM comporte deux connecteurs mâles 24 broches haute densité pour la connexion des servocommandes. Le connecteur A contient des connexions pour l'axe 1. Le connecteur B contient des connexions pour l'axe 2 ainsi que des connexions générales, dont l'entrée analogique et les sorties de commande. Les tableaux suivants fournissent une description succincte de la fonction des connexions d'E/S.

1.3.1. Câble d'E/S et bornier

Les connecteurs haute densité sont utilisés sur le module APM afin de permettre un grand nombre de connexions d'E/S dans les limites physiques du module APM. Afin de faciliter le câblage vers la servocommande et la machine, tous les connecteurs haute densité sont généralement connectés par un câble de faible longueur à un bornier. Reportez–vous à la figure ci–dessous, aux tableaux 2–2, 2–3, 2–4 et 2–5, et aux annexes B et D.

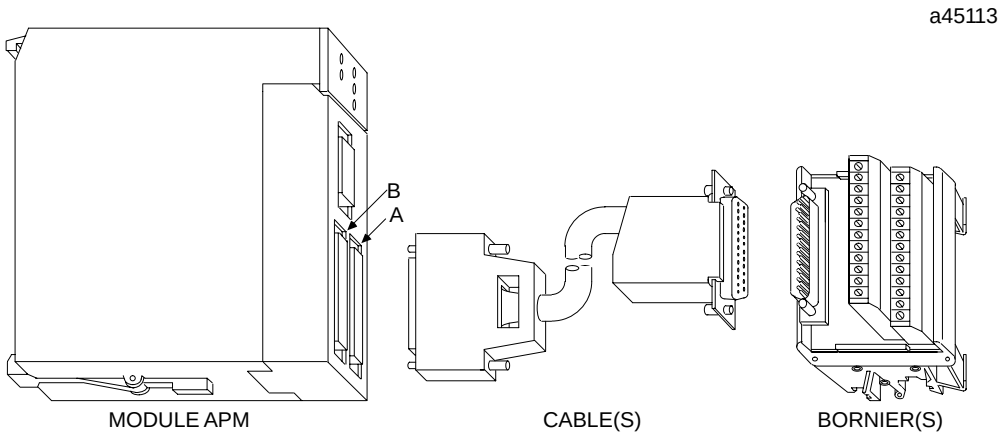


Figure 2-3. Câble d'E/S et bornier

1.3.2. Connexions du câble d'E/S du module APM mono–axe (IC693APU301)

Les tableaux 2–2 et 2–3 définissent les connexions des câbles pour le module APM mono–axe.

Tableau 2-2. Connexions des câbles pour le connecteur d'E/S A de la façade (APM mono–axe)

Numéros des broches du connecteur A du module d'E/S	Numéros des bornes du bornier	Description
A1	12	0 V
B1	24	Réservé
A2	11	Réservé
B2	23	Entrée CTL02 (+)
A3	10	Entrée CTL02 (–)
B3	22	Echantillonnage 1 (+) / Entrée CTL01 (+)
A4	9	Echantillonnage 1 (–) / Entrée CTL01 (–)
B4	15	Commun des entrées CTL03, 05, 06
A5	2	Dépassement 1 (direction –) / Entrée CTL06
B5	14	Dépassement 1 (direction +) / Entrée CTL05
A6	1	Position initiale 1 / Entrée CTL03
B6	16	Sortie d'activation du relais 1 (–)
A7	3	Sortie d'activation du relais 1 (+)
B7	17	Commun de la sortie analogique 1 (commande de vitesse) *
A8	4	Sortie analogique 1 (commande de vitesse) *
B8	21	0 V
A9	8	Alimentation codeur +5 V
B9	20	Voie codeur Z1+
A10	7	Voie codeur Z1–
B10	19	Voie codeur B1+
A11	6	Voie codeur B1–
B11	18	Voie codeur A1+
A12	5	Voie codeur A1–
B12	13	Blindage
	25	Pas de connexion

* Sortie protégée par fusible – l'application d'une source externe à cette sortie peut faire fondre le fusible.

Tableau 2-3. Connexions des câbles pour le connecteur d'E/S B de la façade (APM mono–axe)

Numéros des broches du connecteur B du module d'E/S	Numéros des bornes du bornier	Description
A1	12	0 V
B1	24	Entrée analogique (+)
A2	11	Entrée analogique (–)
B2	23	Sortie CTL12 *
A3	10	Sortie CTL11 *
B3	22	Sortie CTL10 *
A4	9	Sortie CTL09 *
B4	15	Commun des entrées CTL04, 07, 08
A5	2	Entrée CTL08
B5	14	Entrée CTL07
A6	1	Entrée CTL04
B6	16	Sortie d'activation du relais 2 (–)
A7	3	Sortie d'activation du relais 2 (+)
B7	17	Commun de la sortie analogique 2 *
A8	4	Sortie analogique 2 *
B8	21	0 V
A9	8	Sortie +5 V
B9	20	Réservé
A10	7	Réservé
B10	19	Réservé
A11	6	Réservé
B11	18	Réservé
A12	5	Réservé
B12	13	Blindage
	25	Pas de connexion

* Sortie protégée par fusible – l'application d'une source externe à cette sortie peut faire fondre le fusible.

1.3.3. Connexions du câble d'E/S pour le module APM à deux axes (IC693APU302)

Les tableaux 2–4 et 2–5 définissent les connexions des câbles pour le module APM à deux axes

Tableau 2-4. Connexions des câbles pour le connecteur d'E/S A de la façade (APM 2 axes)

Numéros des broches du connecteur A du module d'E/S	Numéros des bornes du bornier	Description
A1	12	0 V
B1	24	Réservé
A2	11	Réservé
B2	23	Echantillonnage 2 (+) / Entrée CTL02 (+)
A3	10	Echantillonnage 2 (–) / Entrée CTL02 (–)
B3	22	Echantillonnage 1 (+) / Entrée CTL01 (+)
A4	9	Echantillonnage 1 (–) / Entrée CTL01 (–)
B4	15	Commun des entrées CTL03, 05, 06
A5	2	Dépassement 1 (direction –) / Entrée CTL06
B5	14	Dépassement 1 (direction +) / Entrée CTL05
A6	1	Position initiale 1 / Entrée CTL03
B6	16	Sortie d'activation du relais 1 (–)
A7	3	Sortie d'activation du relais 1 (+)
B7	17	Commun de la sortie analogique 1 (commande de vitesse)*
A8	4	Sortie analogique 1 (commande de vitesse)*
B8	21	0 V
A9	8	Alimentation codeur +5 V
B9	20	Voie codeur Z1+
A10	7	Voie codeur Z1–
B10	19	Voie codeur B1+
A11	6	Voie codeur B1–
B11	18	Voie codeur A1+
A12	5	Voie codeur A1–
B12	13	Blindage
	25	Pas de connexion

* Sortie protégée par fusible – l'application d'une source externe à cette sortie peut faire fondre le fusible.

Tableau 2-5. Connexions des câbles pour le connecteur d'E/S B de la façade (APM 2 axes)

Numéros des broches du connecteur B du module d'E/S	Numéros des bornes du bornier	Description
A1	12	0 V
B1	24	Entrée analogique (+)
A2	11	Entrée analogique (–)
B2	23	Sortie CTL12 *
A3	10	Sortie CTL11 *
B3	22	Sortie CTL10 *
A4	9	Sortie CTL09 *
B4	15	Commun des entrées CTL04, 07, 08
A5	2	Dépassement 2 (direction –) / Entrée CTL08
B5	14	Dépassement 2 (direction +) / Entrée CTL07
A6	1	Position initiale 2 / Entrée CTL04
B6	16	Sortie d'activation du relais 2 (–)
A7	3	Sortie d'activation du relais 2 (+)
B7	17	Commun de la sortie analogique 2 *
A8	4	Sortie analogique 2 *
B8	21	0 V
A9	8	Sortie +5 V
B9	20	Voie codeur Z2 +
A10	7	Voie codeur Z2 –
B10	19	Voie codeur B2 +
A11	6	Voie codeur B2 –
B11	18	Voie codeur A2 +
A12	5	Voie codeur A2 –
B12	13	Blindage
	25	Pas de connexion

* Sortie protégée par fusible – l'application d'une source externe à cette sortie peut faire fondre le fusible.

1.4. SCHÉMAS FONCTIONNELS DE CONNEXION

Les schémas ci–dessous présentent les connexions des E/S des modules APM (1 et 2 axes) à un cicruit de commande et une machine dans une application caractéristique. Vous devez utiliser le câble blindé de la façon indiquée.

1.4.1. Schémas fonctionnels de connexion pour le module APM mono–axe

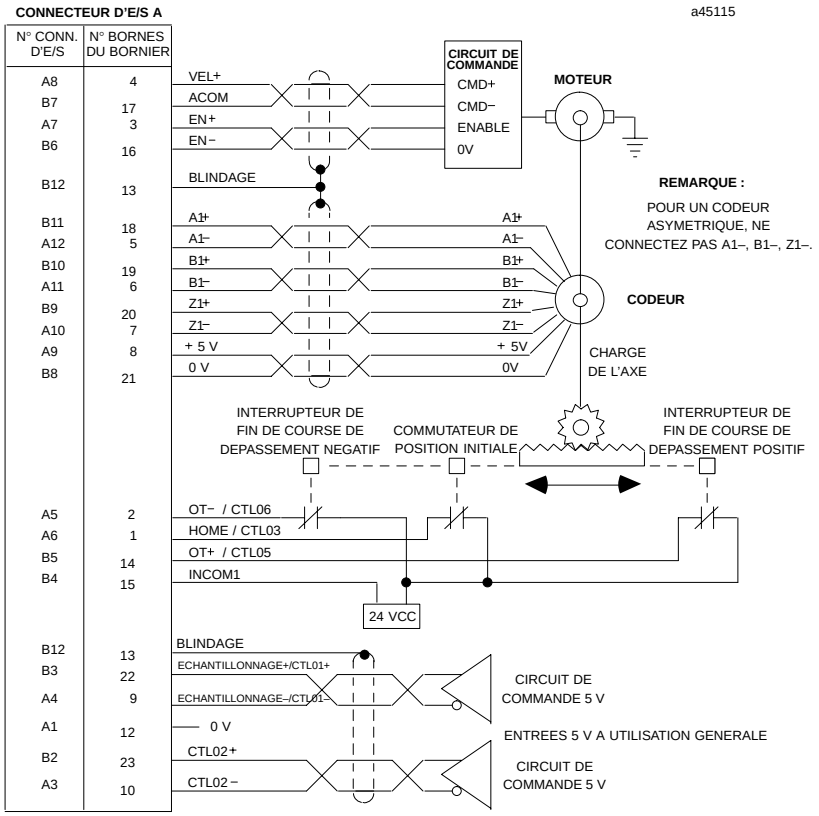


Figure 2-4. Schéma fonctionnel du connecteur d'E/S A du module APM mono–axe

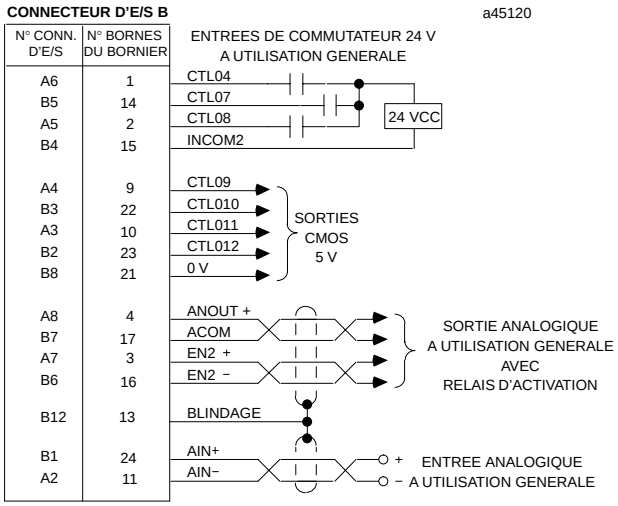


Figure 2-5. Schéma fonctionnel du connecteur d'E/S B du module APM mono–axe

1.4.2. Schémas fonctionnels de connexion pour le module APM à deux axes

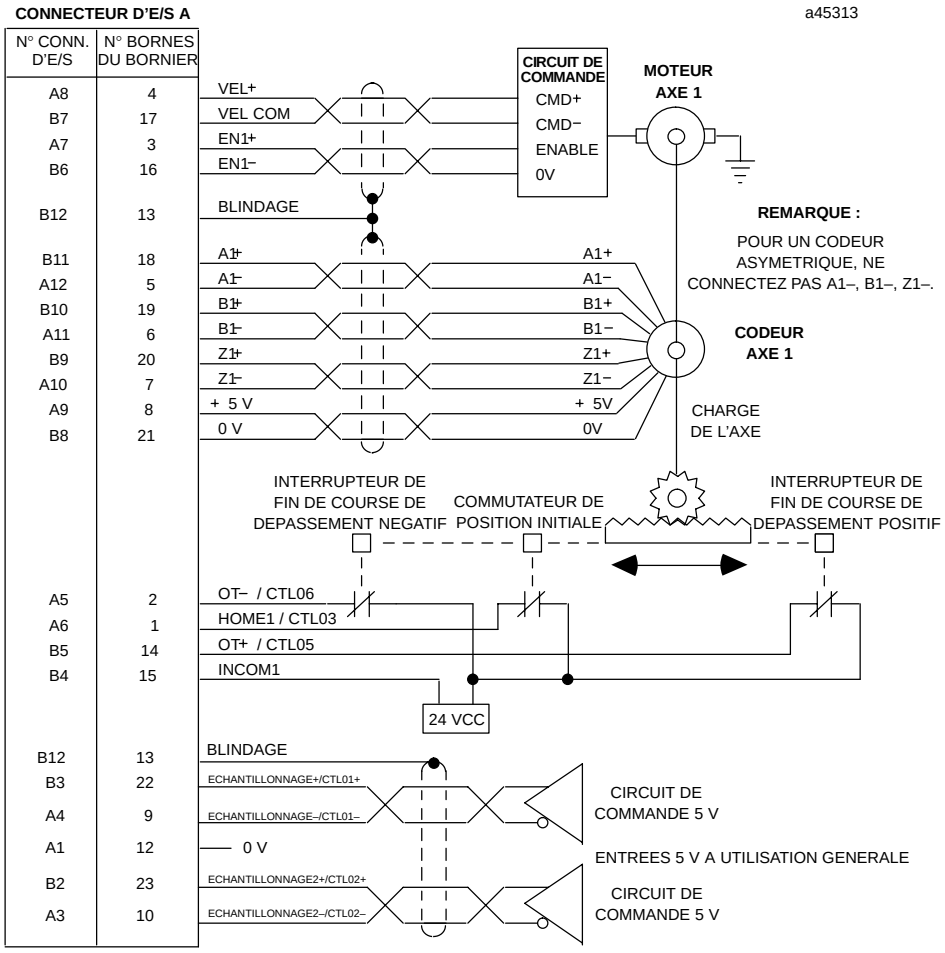


Figure 2-6. Schéma fonctionnel du connecteur d'E/S A du module APM à deux axes

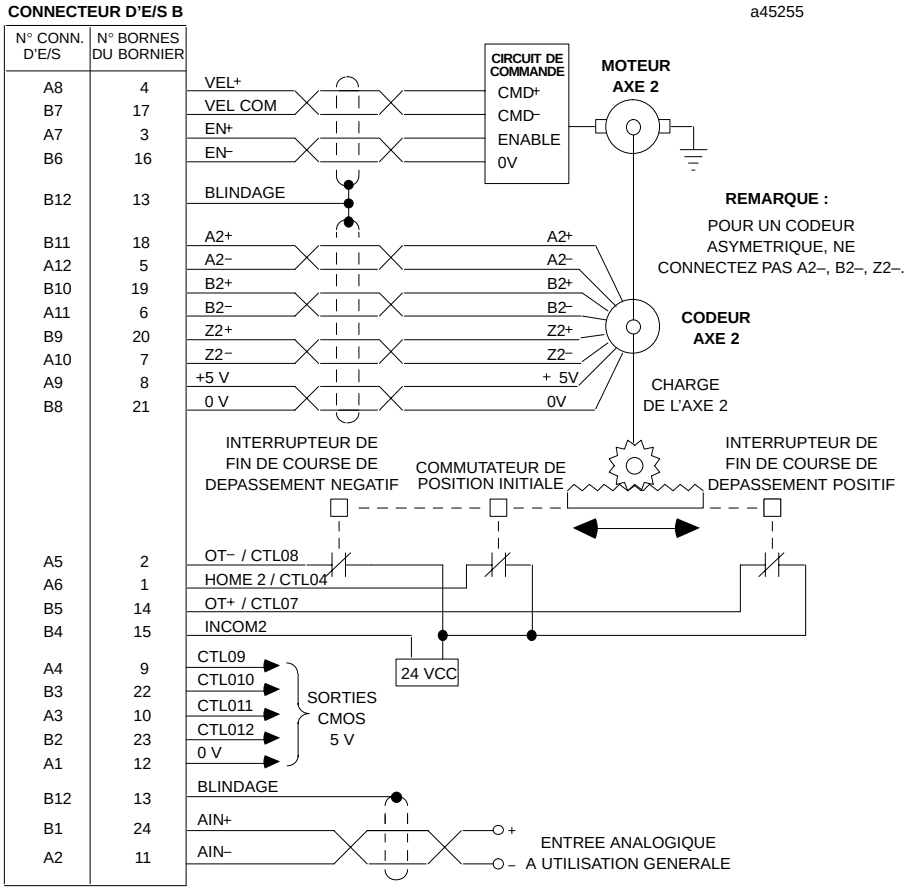


Figure 2-7. Schéma fonctionnel du connecteur d'E/S B du module APM à deux axes

2. INSTALLATION DU MODULE APM

Le module APM peut fonctionner dans toutes les platines d'UC ou d'extension de l'API Série 90–30. Pour connaître le nombre maximum de modules APM que vous pouvez installer par platine et par système, reportez–vous à l'annexe C, Spécifications.

Les fichiers de configuration créés par le logiciel de configuration Logicmaster 90 doivent correspondre à la configuration physique des modules.

Pour installer le module APM sur la platine, suivez la procédure suivante :

1. Utilisez le logiciel Logicmaster 90 ou la miniconsole de programmation pour arrêter l'API. Vous empêcherez ainsi le programme d'application local éventuellement présent de lancer une commande pouvant affecter le fonctionnement du module.
2. Mettez le système d'API Série 90–30 hors tension.
3. Alignez le module avec l'emplacement et le connecteur choisis. Inclinez le module vers l'avant de façon que le crochet arrière s'engage dans l'emplacement de la platine.
4. Poussez le module vers le bas jusqu'à ce que les connecteurs s'accouplent et que le levier de blocage en bas du module signale par un déclic qu'il est en place dans le cran de la platine.
5. Reportez–vous aux figures 2–4 et 2–5, ou 2–6 et 2–7, et aux tableaux 2–2 et 2–3, ou 2–4 et 2–5, pour les exigences de câblage.
6. Mettez le bac de l'API sous tension. Le voyant d'état (Status) du module APM s'allume lorsque les tests de mise sous tension du module sont terminés.
7. Répétez cette procédure pour chaque module APM.
8. Configurez le(s) module(s) APM de la façon décrite au chapitre 3.

Page laissée blanche intentionnellement

Chapitre 3 Configuration du module APM

La configuration du module APM est réalisée avec le logiciel de configuration Logicmaster 90–30. Cette configuration est une procédure à deux étapes :

- Configuration du bac d'E/S
- Configuration du module

1. CONFIGURATION DU BAC D'E/S

La configuration du module APM avec le logiciel Logicmaster 90–30 est identique à celle des autres modules de l'API Série 90–30. Le logiciel permet de définir le type et l'emplacement de tous les modules présents dans les bacs de l'API. Il suffit pour cela de compléter des écrans de configuration représentant les modules dans une platine et de sauvegarder les informations dans un fichier de configuration, qui sera ensuite transféré dans l'UC.

Après avoir défini une platine et un emplacement pour l'APM, vous pouvez continuer la deuxième partie de la configuration de l'APM, Configuration du module.

2. CONFIGURATION DU MODULE

Ce paragraphe est divisé en trois parties :

- Réglage des paramètres de configuration
- Paramètres de configuration essentiels
- Remarques importantes relatives à la configuration

2.1. RÉGLAGE DES PARAMÈTRES DE CONFIGURATION

Comme pour la configuration du bac d'E/S, la configuration du module est réalisée en complétant des écrans dans le logiciel de configuration Logicmaster 90–30.

Les données de configuration du module APM sont de quatre types :

- Données de configuration du module
- Données de configuration du port de la console de programmation
- Données de configuration des axes
- Editeur Programme Zéro

2.1.1. Données de configuration du module

Au cours de chaque cycle d'UC, le module APM et l'UC échangent un certain nombre de données. Les données d'interface APM–UC font référence aux emplacements de départ pour les transferts automatiques. Le tableau 3–1 décrit les paramètres de configuration des données de configuration du module.

Tableau 3-1. Données de configuration du module

Paramètre de configuration	Description	Valeurs	Valeur par défaut	Unités
Adresse de référence	Adresse de départ pour la classe d'implantation %I (32 bits)	En fonction de l'UC	%I00001 ou référence supérieure suivante	n/a
Adresse de référence	Adresse de départ pour la classe d'implantation %Q (32 bits)	En fonction de l'UC	%Q00001 ou référence supérieure suivante	n/a
Adresse de référence	Adresse de départ pour la classe d'implantation %AI (15 mots pour 1 axe, 28 pour 2 axes)	En fonction de l'UC	%AI00001 ou référence supérieure suivante	n/a
Adresse de référence	Adresse de départ pour la classe d'implantation %AQ (6 mots)	En fonction de l'UC	%AQ00001 ou référence supérieure suivante	n/a
%AI Err Pos	Sur les APM mono–axe, ce paramètre ajoute l'erreur de position aux données %AI	DEVALIDE/ VALIDE	DEVALIDE	n/a
Type fdback	Type de retour[1]	CODEUR/ LINEAI RESOLVR	CODEUR	n/a
Boucle Ctl	Type de boucle de contrôle	STANDARD/ SPECIAL[2]/ SUIVEUR[3]	STANDARD	n/a

1 Seul le type de retour CODEUR est actuellement mis en oeuvre dans le module APM.
2 Pour des applications spéciales uniquement.
3 Pour obtenir des informations sur le type de boucle de contrôle Suiveur, reportez–vous au document *GFK–0781 Series 90–30 PLC Axis Positioning Module (APM30) – Follower Mode User's Manual*

2.1.2. Données de configuration du port de communication série

Vous pouvez programmer le module APM en utilisant le logiciel Motion Programmer. L'ordinateur exécutant le logiciel Motion Programmer doit être connecté au port de communication série (qui supporte le protocole SNP) sur le plastron du module APM.

Le port de communication série doit être correctement configuré pour communiquer avec le logiciel Motion Programmer. Vérifiez que les paramètres de configurations du logiciel Motion Programmer et du port de communication série correspondent. Le tableau 3–2 décrit les données de configuration du port de communication série.

Tableau 3-2. Données de configuration du port de communication série

Paramètre de configuration	Description	Valeurs	Valeur par défaut	Unités
Vitesse, Bd	Débit du port SNP	300, 600, 1200, 2400, 4800, 9600, 19200	19200	n/a
Parité	Parité	PAIR, IMPA, SANS	IMPA	n/a
Bits Stop	Nombre de bits de stop	1 ou 2	1	n/a
Bits donn.	Nombre de bits de données	7 ou 8	8	n/a
Modem TT	Temps de retournement du modem	De 0 à 2550, en multiples de 10 millisecondes	0	ms
Temps idle	Temps d'inactivité maximum de la liaison	De 1 à 60	10	s
ID SNP	ID SNP	6 car. composés de A–F et 0–9. Le premier car. doit être A–F	A00001	n/a

2.1.3. Données de configuration des axes

Les données de configuration des axes du module APM se composent de valeurs de base attribuées aux paramètres de configuration et utilisées par un ou plusieurs programmes de mouvement. Les valeurs de ces paramètres étant généralement identiques, elles ne sont pas incluses dans le programme de mouvement. Vous trouverez ici une définition et une description succincte des paramètres de configuration. Voir tableau 3–3.

Tableau 3-3. Données de configuration des axes

Paramètre de configuration	Description	Valeurs	Valeur par défaut	Unités[1]
Unités Util	Valeur en unités utilisateurs	De 1 à 65535	1	n/a
Comptages	Comptages de retour	De 1 à 65535	1	n/a
Fdc Limite	Activation/désactivationd'interrupteur de fin de course de dépassement	VALIDE/DEVALIDE	VALIDE	n/a
Fdc Pos	Fin de course positive	De –8388608 à +8388607	+8388607	Unités utilisateur
Fdc Neg	Fin de course négative	De –8388608 à +8388607	– 8388608	Unités utilisateur
Lim Err Pos[2]	Limite d'erreur de position	De 256 à 60000	+ 4096	Unités utilisateur
Zone En Pos	En zone	De 0 à 2000	10	Unités utilisateur
Anticip vit	Anticipation de vitesse	De 0 à 100	0	%
CT Intgr	Constante de temps d'intégration	0 et de 10 à 65535	0	ms
Mode Intgr	Mode d'intégration	SANS/CONTINU/EN ZONE	SANS	n/a
Comp Retour	Compensation d'inversion	De 0 à 255	0	Unités utilisateur
Disdly	Temps de désactivation du circuit de commande	De 0 à 65535	100	ms
TC Bouc Pos[2]	Constante de temps de boucle de position	0 et de 5 à 10000	1000	ms
Vit à 10 V[2]	Vitesse pour une sortie de 10 V	De 400 à 1000000	+4000	Unités utilisateur/s
Vit Jog	Vitesse de mouvement	De 1 à 8388607	+1000	Unités utilisateur/s
Acc Jog	Accélération de mouvement	De 1 à 134217727	+10000	Unités utilisateur/s/s
Mod Acc Jog	Mode d'accélération de mouvement	LINEAI/COURBE	LINEAI	n/a
Limite Hte	Limite de comptage supérieure	De –8388608 à +8388607	+8388607	Unités utilisateur
Limite Bas	Limite de comptage inférieure	De –8388608 à +8388607	–8388608	Unités utilisateur
Positn Orig	Position initiale	De –8388608 à +8388607	0	Unités utilisateur
Décal. Orig.	Valeur de décalage de position initiale	De –32768 à +32767	0	Unités utilisateur
Vit Rech Or	Vitesse de recherche de position initiale	De 1 à 8388607	+2000	Unités utilisateur/s
Vit Fin Or	Vitesse finale de position initiale	De 1 à 8388607	+500	Unités utilisateur/s
Mode origin	Mode de retour à la position initiale	ORIGIN/SENS+/SENS–	ORIGIN	n/a

1 La mesure unité "unités utilisateur" est le rapport Unités Util/comptages.
2 Le paramètre de configuration dépendant de la mise à l'échelle, la plage de valeurs dépend de l'échelle de l'utilisateur.

Unités Util. Le rapport Unités utilisateur/comptages définit le nombre d'unités de programmation pour chaque comptage de retour. Ceci permet à l'utilisateur de programmer l'APM en unités adaptées à l'application. La plage des unités utilisateurs et des comptages va de 1 à 65535. Le rapport des unités utilisateur aux comptages doit être dans la plage 8:1 – 1:32. Par exemple, si 1000 pouces de déplacement correspondent à 8000 comptages de retour, un rapport unités utilisateur/comptages de 1000:8000 définit l'unité utilisateur à 0,001 pouce. Rapport par défaut 1:1.

Fdc Limit. Indique si l'APM utilise les entrées d'interrupteur de fin de course de dépassement matérielles. Si les interrupteurs de fin de course sont désactivés (DEVALIDE), les entrées d'interrupteur de fin de course peuvent être utilisées en tant qu'entrées d'utilisation générale. S'ils sont activés (VALIDE), une tension de 24 Vcc doit être appliquée aux deux entrées pour que l'APM puisse fonctionner. Si ce n'est pas le cas, chaque activation de la commande entraîne une erreur d'entrée d'interrupteur de fin de course. Vous pouvez utiliser les bits %Q de mouvement et d'effacement d'erreur simultanément pour éloigner l'APM de la position qui déclenche l'interrupteur de fin de course. Valeur par défaut : VALIDE.

Fdc Pos. Fin de course logicielle positive (unités utilisateur). Si vous programmez le module APM pour aller jusqu'à une position supérieure à la fin de course logicielle positive, une erreur se produit et l'APM empêche le mouvement de l'axe. Valeur par défaut : +8388607

Fdc Neg. Fin de course logicielle négative (unités utilisateur). Si vous programmez l'APM pour aller jusqu'à une position inférieure à la fin de course logicielle négative, une erreur se produit et le module APM empêche le mouvement de l'axe. Valeur par défaut : –8388608

Lim Err Pos. Limite d'erreur de position (unités utilisateur). Erreur de position maximale *(Position commandée – Position réelle*) autorisée lorsque l'APM contrôle une servocommande. Ce paramètre doit normalement être défini à une valeur de 10 à 20 % supérieure à la plus grande erreur de position rencontrée pendant le fonctionnement normal de la servocommande. Valeur par défaut : 4096. La formule de plage pour la limite d'erreur de position est la suivante :

256 * (unités utilisateur/comptages) ≤ Limite d'erreur de position ≤ 60000 * (unités utilisateurs/comptages)

Si vous n'utilisez pas l'anticipation de vitesse, vous pouvez définir la limite d'erreur de position à une valeur environ 20 % supérieure à l'erreur de position nécessaire pour produire une commande de 10 volts. *L'erreur de position* (unités utilisateur) nécessaire pour produire une commande de 10 V avec une anticipation de vitesse de 0 % est :

$$\text{Erreur de position (unités util)} = \frac{\text{Constante de temps de boucle de position (ms) x Vitesse à 10 V de la servocommande (unités util/s)}}{1000}$$

Zone En Pos. En Zone (unités utilisateur). Lorsque l'ampleur de l'erreur de position de la servocommande est inférieure ou égale à cette valeur et qu'aucun mouvement (Jog) ni mouvement à vitesse constante (Move at Velocity) n'est commandé, le bit d'état *En Zone* est mis à 1. "En Zone" détermine également l'erreur de position à partir de laquelle les *Pmove* sont considérés comme terminés. Valeur par défaut : 10

Anticip vit. Gain d'anticipation de vitesse (pourcentage). Pourcentage de la vitesse commandée qui est ajouté à la sortie de commande de vitesse du module APM. Si vous augmentez l'anticipation de vitesse, la servocommande fonctionne avec de meilleurs temps de réponse et une erreur de position inférieure. Les valeurs d'anticipation de vitesse optimales sont comprises entre 80 et 90 %. **Vous devez définir la valeur de "Vit à 10 V" correctement pour que l'anticipation de vitesse fonctionne correctement. Valeur par défaut : 0**

CT Intgr. Constante de temps d'intégration (ms). Constante de temps utilisée pour l'intégrateur d'erreur de position (en millisecondes). Cette valeur indique le temps qui sera nécessaire pour supprimer 63 % de l'erreur de position. Par exemple, si la constante de temps d'intégration vaut 1000, c'est–à–dire 1 seconde, l'erreur de position sera réduite à 37% de sa valeur initiale après 1 seconde. Une valeur de zéro désactive l'intégrateur. La constante de temps d'intégration doit être de 5 à 10 fois supérieure à la constante de temps de boucle de position pour empêcher l'instabilité et l'oscillation. Valeur par défaut : 0

Mode Intgr. Mode d'intégration. Mode de fonctionnement de l'intégrateur d'erreur de position. SANS signifie que l'intégrateur n'est pas utilisé. CONTINU signifie que l'intégrateur fonctionne en permanence, même pendant le déplacement de la servocommande. EN ZONE signifie que l'intégrateur fonctionne uniquement lorsque le bit d'état *En Zone* est à 1. Valeur par défaut : SANS

Comp Retour. Compensation d'inversion (unités utilisateur). Facteur de compensation qui permet à la servocommande d'inverser le sens tout en continuant à fournir un positionnement correct dans les systèmes ayant du jeu. Valeur par défaut : 0.

Disdly (ms). Temps de désactivation du circuit de commande (millisecondes). Retard entre la commande de vitesse nulle et la mise à 0 de la sortie d'activation du circuit de commande. Le temps de désactivation se produit lorsque le bit %Q d'activation du circuit de commande est à 0 ou que certaines conditions d'erreur apparaissent. Le temps de désactivation ne doit pas être supérieur au temps de décélération de la servocommande partant de sa vitesse maximale. Valeur par défaut : 100

TC Bouc Pos. Constante de temps de boucle de position (millisecondes). Constante de temps de boucle de position souhaité pour la servocommande. Plus cette valeur est faible, plus le système répond rapidement. Les valeurs trop faibles entraînent une instabilité et une oscillation du système. Pour un suivi précis du profil de vitesse commandée, "TC Bouc Pos" doit être compris entre 1/4 et 1/2 du temps de décélération MINIMUM du système. Si vous réglez la constante de temps de boucle de position à 0, l'APM est en mode de "boucle ouverte" et utilise uniquement l'anticipation pour produire la sortie analogique de commande de vitesse. **La constante de temps de boucle de position ne sera pas précise si la valeur "Vit à 10 V" n'est pas correctement réglée.** Valeur par défaut : 1000

Vit à 10 V. Vitesse de la servocommande réelle (unités utilisateur/seconde) pour une sortie de commande de vitesse de l'APM de 10 V. **Cette valeur doit être définie correctement pour que les facteurs "TC Bouc Pos" et "Anticip vit" soient précis.** Vous pouvez utiliser la commande immédiate %AQ *Forçage sortie N/A* et le mot d'état %AI *Vitesse réelle* de l'APM pour déterminer la valeur de configuration correcte. La formule de plage de la vitesse à 10 Volts, qui est affectée par la mise à l'échelle, est :

400 * (unités utilisateurs/comptages) ≤ Vitesse à 10 Volts ≤ 1000000 * (unités utilisateurs/comptages)

Valeur par défaut : 4000

Vit Jog. Vitesse de mouvement (unités utilisateur/seconde). Vitesse à laquelle la servocommande se déplace pendant une opération de mouvement (*Jog*). Valeur par défaut : 1000

Acc Jog. Taux d'accélération de mouvement (unités utilisateur/seconde/seconde). Taux d'accélération utilisé pendant les opérations de mouvement (*Jog*), recherche position initiale (*Find Home*), mouvement à vitesse constante (*Move at Velocity*) et interruption (*Abort*). L'accélération de mouvement est utilisée lorsqu'aucune accélération n'a été programmée. Valeur par défaut : 10000

Mod Acc Jog. Mode d'accélération de mouvement (LINEAI ou COURBE). Mode d'accélération pour les opérations de mouvement (*Jog*), recherche position initiale (*Find Home*), mouvement à vitesse constante (*Move at Velocity*) et interruption (*Abort*). LINEAI entraîne une modification linéaire de la vitesse dans le temps. COURBE entraîne une modification de la vitesse commandée moins forte que le mode linéaire au début et à la fin des intervalles d'accélération. Valeur par défaut : LINEAI

Limite Hte. Limite de comptage supérieure (unités utilisateur). Plus haute position possible pour le déplacement d'un axe avant "basculement" sur la limite de comptage inférieure. Les limites de comptages sont utilisées pour les applications rotatives. Valeur par défaut : 8388607

Limite Bas. Limite de comptage inférieure (unités utilisateur). Plus faible position possible pour le déplacement d'un axe avant "basculement" sur la limite de comptage supérieure. Les limites de comptages sont utilisées pour les applications rotatives. Valeur par défaut : –8388608

Positn Orig. Position initiale (unités utilisateur). Valeur affectée à la position réelle à la fin d'un cycle de recherche de position initiale. Valeur par défaut : 0

Décal. Orig. Décalage de position initiale (unités utilisateur). Décalage du point d'arrêt final de la servocommande à la fin d'un cycle de recherche de position initiale. Les décalages de position initiale règlent le point d'arrêt final de la servocommande par rapport au marqueur du codeur. Valeur par défaut : 0

Vit Rech Or. Vitesse de recherche de position initiale (unités utilisateur/seconde). Vitesse à laquelle la servocommande recherche la transition initiale du commutateur de position initiale au cours du cycle de recherche de position initiale. Si nécessaire, vous pouvez régler la vitesse de recherche de position initiale à une valeur élevée pour permettre à la servocommande de repérer rapidement le commutateur de position initiale. Valeur par défaut : 2000

Vit Fin Or. Vitesse finale de position initiale (unités utilisateur/seconde). Vitesse à laquelle la servocommande recherche la transition finale du commutateur de position initiale et l'impulsion du marqueur de codeur à la fin d'un cycle de recherche de position initiale. La vitesse finale de position initiale doit être suffisamment faible pour permettre un temps de 10 ms (temps de filtre) entre la transition finale du commutateur de position initiale et l'impulsion du marqueur de codeur. Valeur par défaut : 500

Mode origin. Mode de recherche de position initiale. Méthode utilisée pour rechercher la position initiale au cours d'un cycle de recherche de position initiale. ORIGIN indique la surveillance d'un commutateur de position initiale. SENS+ et SENS– spécifient un mouvement direct positif ou négatif vers le marqueur suivant à la vitesse finale de recherche de position initiale. Valeur par défaut : ORIGIN

2.1.4. Editeur Programme Zéro

L'éditeur Programme Zéro permet d'écrire un petit programme de mouvement (au maximum 20 commandes) téléchargé à chaque initialisation du module APM. Pour écrire ce programme, vous devez entrer, dans les écrans de configuration du module (du quatrième au sixième), des commandes de mouvement en anglais dans un format identique à celui des commandes du logiciel Motion Programmer. Pour un module à 2 axes, l'APM détermine s'il s'agit d'un programme à un axe, à deux axes ou multi–axe en fonction des axes utilisés.

Pour entrer les commandes, vous devez utiliser les touches de fonction (F1–F9). Chaque commande est associée à un champ de données permettant d'entrer un entier signé ou le numéro d'un paramètre de l'APM, en fonction de la commande configurée.

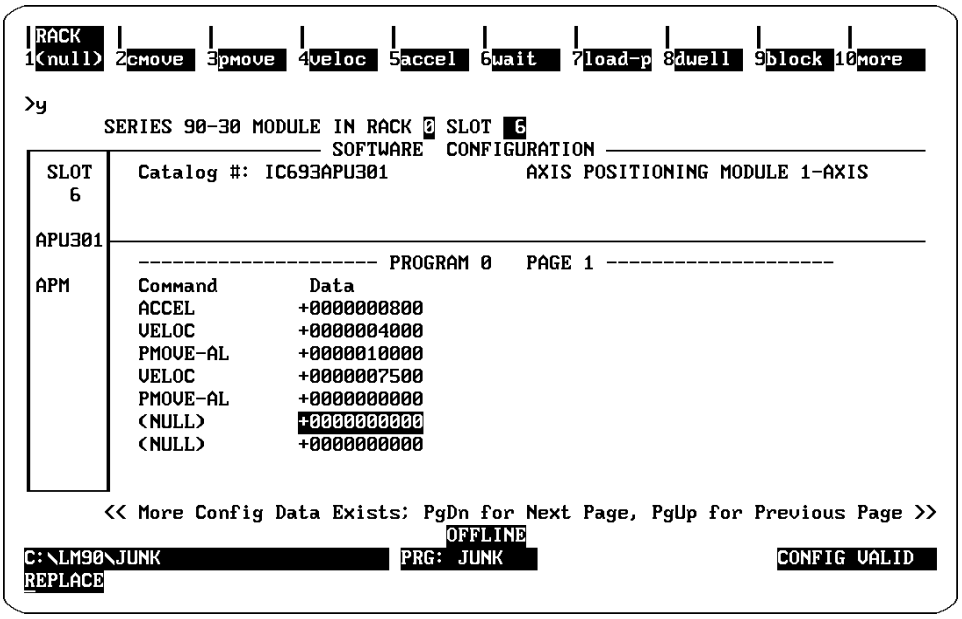


Figure 3-1. Exemple d'écran de l'éditeur Programme Zéro

2.1.4.1. Format des instructions

Les instructions créées avec l'éditeur Programme Zéro se composent d'une commande et de données associées décrivant la commande. Vous devez entrer la commande et les données dans des zones de l'écran de l'éditeur de programmes appelés champs. Vous trouverez ci–dessous une description de ces champs :

Champ du nom de la commande. Désignation de la commande (en anglais). Pour entrer une commande, placez le curseur sur un champ de nom de commande et appuyez sur la touche de fonction appropriée (F1–F9). La plupart des commandes sont réunies dans des groupes tels que CMOVE, PMOVE, VELOC, ACCEL, etc. Vous pouvez afficher les variantes de ces commandes en utilisant la touche Tab après avoir appuyé sur la touche de fonction d'un groupe particulier.

Par exemple, pour programmer une commande CMOVE–IS–P (mouvement continu, incrémentiel, courbe en S, avec paramètre), placez–vous d'abord sur un champ de commande et appuyez sur la touche de fonction *CMOVE*. Le logiciel affiche alors CMOVE–AL dans le champ du nom de commande de l'écran. Appuyez sur la touche Tab pour afficher les différentes variantes de la commande.

Pour programmer une commande JUMP, appuyez sur F10 (MORE) et appuyez sur F1 (JUMP).

Champ d'opérande. Chaque commande est associée à un champ de données. Dans ce champ, vous pouvez entrer un entier signé ou le numéro d'un paramètre (0– 255), en fonction de la commande configurée. Vous pouvez charger des données dans les paramètres 1–20 en utilisant la commande LOAD–P.

Tableau 3-4. Commande des programmes de mouvement de l'éditeur Programme Zéro

Nom de la commande	Définition	Plage	Valeur par défaut
(NULL)	Aucune action	0	0
BLOCK	Définition d'un numéro de bloc	De 1 à 65535	1
BLOCK–SYNC	Définition d'un numéro de bloc synchrone	De 1 à 65535	1
CMOVE–AL	Mouvement continu, absolu, linéaire	De –8388608 à +8388607	1
CMOVE–AL–P	Mouvement continu, absolu, linéaire, utilise la donnée du paramètre	De 0 à 255	1
CMOVE–AS	Mouvement continu, absolu, courbe en S	De –8388608 à +8388607	1
CMOVE–AS–P	Mouvement continu, absolu, courbe en S, utilise la donnée du paramètre	De 0 à 255	1
CMOVE–IL	Mouvement continu, incrémentiel, linéaire	De –8388608 à +8388607	1
CMOVE–IL–P	Mouvement continu, incrémentiel, linéaire, utilise la donnée du paramètre	De 0 à 255	1
CMOVE–IS	Mouvement continu, incrémentiel, courbe en S	De –8388608 à +8388607	1
CMOVE–IS–P	Mouvement continu, incrémentiel, courbe en S, utilise la donnée du paramètre	De 0 à 255	1
PMOVE–AL	Mouvement de positionnement, absolu, linéaire	De –8388608 à +8388607	1
PMOVE–AL–P	Mouvement de positionnement, absolu, linéaire, utilise la donnée du paramètre	De 0 à 255	1
PMOVE–AS	Mouvement de positionnement, absolu, courbe en S	De –8388608 à +8388607	1
PMOVE–AS–P	Mouvement de positionnement, absolu, courbe en S, utilise la donnée du paramètre	De 0 à 255	1
PMOVE–IL	Mouvement de positionnement, incrémentiel, linéaire	De –8388608 à +8388607	1
PMOVE–IL–P	Mouvement de positionnement, incrémentiel, linéaire, utilise la donnée du paramètre	De 0 à 255	1
PMOVE–IS	Mouvement de positionnement, incrémentiel, courbe en S	De –8388608 à +8388607	1
PMOVE–IS–P	Mouvement de positionnement, incrémentiel, courbe en S, utilise la donnée du paramètre	De 0 à 255	1
VELOC	Réglage de la vitesse	De 1 à 8388607	2000
VELOC–P	Réglage de la vitesse avec la donnée du paramètre	De 0 à 255	1
ACCEL	Réglage de l'accélération	De 1 à 134217727	5000
ACCEL–P	Réglage de l'accélération avec la donnée du paramètre	De 0 à 255	1
WAIT	Attendre mise à 1 du bit CTL XX avant déplacement	De 1 à 12	1
LOAD–P01 . LOAD–P20	Chargement du numéro de registre du paramètre du module APM	De –8388608 à +8388607	0
DWELL	Attendre X millisecondes	de 0 à 60000	0
DWELL–P	Attendre X millisecondes, X est la valeur du paramètre	De 0 à 255	1
JUMP–UNCOND	Continuer exécution du programme au numéro de bloc donné	De 1 à 65535	1
JUMP–CTL01 . JUMP–CTL12	Si le bit CTL donné passe à 1 au cours de l'exécution du bloc actuel, transférer l'exécution du programme au numéro de bloc donné	De 1 à 65535	1

2.1.4.2. Description des commandes de mouvement de l'éditeur Programme Zéro

Vous trouverez ci–dessous une description succincte de chaque commande. Pour une description complète de la programmation de mouvement sur le module APM, reportez–vous au document *GFK–0664 Series 90 PLC Axis Positioning Module (APM30) Programmer's Manual.*

Accélération (ACCEL). Cette commande est utilisée pour spécifier les taux d'accélération et de décélération de l'axe pour les mouvements ultérieurs. Une fois rencontré, le taux spécifié reste valide jusqu'à ce qu'il soit écrasé par une nouvelle commande d'accélération.

Bloc (BLOCK). Les blocs sont utilisés pour surveiller et synchroniser l'exécution des programmes, mettre fin aux tests de saut, et en tant que destination pour les sauts. Il s'agit vraiment d'une commande de l'éditeur Programme Zéro.

Mouvement continu (CMOVE). Cette commande est utilisée lorsqu'il n'est pas nécessaire que l'axe se trouve dans la zone configurée (*In Position Zone)* pour traiter la commande suivante. Si aucune accélération ou vitesse n'a été spécifiée précédemment dans le programme, le module utilise l'accélération et/ou la vitesse de mouvement (*Jog Acceleration* et/ou *Jog Velocity*) configurées.

Arrêt momentané (DWELL). Cette commande interrompt le mouvement pour la durée spécifiée (en millisecondes) avant le début de la commande suivante.

Saut (JUMP). Cette commande est utilisée pour faire un saut vers un autre emplacement du programme. Ce saut est conditionné par certains états spécifiés des entrées (CTL 1–8) et des sorties %Q (CTL 9–12) de contrôle. Le saut se produit lorsque les tests de conditions sont vrais (1 logique). Vous pouvez également sélectionner un saut inconditionnel. Le saut peut avoir une destination située en avant ou en arrière dans le programme. Le programme teste la condition de saut dès la fin du déplacement qui précède la commande *Jump*. Vous pouvez utiliser un maximum de 220 sauts pour l'ensemble des programmes et sous–programmes.

Après autorisation du début du test de la condition, le test se produit une fois par milliseconde jusqu'à ce que le système rencontre un numéro de bloc ou une autre commande de saut. Ceci permet un test continu pendant un déplacement ou une série de déplacements si la commande *Jump* est située, dans le bloc, avant les déplacements. Si un saut se produit pendant un mouvement, la suite du mouvement est annulée et la commande située à l'emplacement de destination est immédiatement effective. Les destinations de saut sont limitées au programme contenant la commande *Jump*.

Chargement paramètre (LOAD). Cette commande initialise ou modifie la valeur d'un paramètre du module APM. La nouvelle valeur devient effective dès que le système la rencontre dans le programme. Avec un programme créé avec l'éditeur Programme Zéro, vous pouvez charger les paramètres 1 à 20.

Mouvement de positionnement (PMOVE). Cette commande est utilisée lorsqu'il est nécessaire que l'axe se trouve dans la zone configurée (*In Position Zone)* pour traiter la commande suivante. Si aucune accélération ou vitesse n'a été spécifiée dans un programme de mouvement, le module utilise l'accélération et/ou la vitesse de mouvement (*Jog Acceleration* et/ou *Jog Velocity*) configurées.

Vitesse (VELOC). Cette commande spécifie la vitesse de mouvement de l'axe. Une fois rencontrée, cette commande reste active jusqu'à ce qu'elle soit écrasée par une nouvelle commande de vitesse.

Attente (WAIT). Cette commande synchronise le début du mouvement de l'axe avec une entrée externe ou un événement reporté dans CTL 1–12. Le début du mouvement est interrompu jusqu'à ce que le bit surveillé devienne vrai.

2.2. PARAMÈTRES DE CONFIGURATION ESSENTIELS

La configuration correcte du module APM, exige un réglage correct de nombreux paramètres. Ce paragraphe explique comment ces paramètres affectent le fonctionnement du module APM et comment les différents paramètres sont liés les uns aux autres. Bien que tous les paramètres soient importants, certains sont absolument essentiels à un fonctionnement correct.

2.2.1. Vitesse à 10 Volts

Du réglage de cette valeur dépend le fonctionnement de toutes les fonctions des modules APM et des servocommandes. Cette valeur doit correspondre à la vitesse de l'axe lorsque l'APM envoie une sortie de 10 volts à la servocommande. Reportez–vous au chapitre 5, Procédures de démarrage d'un servomécanisme, pour déterminer la valeur correcte. La plage du paramètre "Vitesse à 10 Volts" dépend de l'échelle de l'utilisateur. La formule est la suivante :

400 * (unités utilisateur/comptages) ≤ Vitesse à 10 Volts ≤ 1000000 * (unités utilisateur/comptages)

2.2.2. Constante de temps de boucle de position

Plus la valeur de la constante de temps de boucle de position est faible, plus l'axe répond rapidement. Cependant, si cette valeur est trop faible, le système risque de devenir instable, voire d'osciller. Au cours d'une décélération, en particulier pour les vitesses élevées, le module APM peut demander à une servocommande de s'arrêter à un certain point trop rapidement pour que la servocommande puisse répondre. Il y aurait alors un dépassement. Vous devez configurer le module APM de façon à EVITER TOUT RISQUE de dépassement.

Pour un suivi précis du profil de vitesse commandée, "TC Bouc Pos" doit être compris entre 1/4 et 1/2 du temps de décélération MINIMUM du système.

Si vous êtes habitués aux bandes passantes de servocommande exprimées en rad/s :

Bande passante (rad/s) = 1000 / Constante de temps de boucle de position (ms)

Si vous êtes habitués aux gains de servocommande exprimés en ipm/mil :

Gain (ipm/mil) = 60 / Constante de temps de boucle de position (ms)

Gain (ipm/mil)	Bande passante (rad/s)	Constante de temps de boucle de position (ms)
0,5	8,5	120
0,75	12,5	80
1	16,6	60
1,5	25,1	40
2	33,4	30
2,5	41,8	24
3	50	20

2.2.2.1. Exemple

Avec une constante de temps de boucle de position de 1 seconde, la position réelle doit être située une seconde après la position commandée (en supposant 0 % d'anticipation de vitesse, une constante de temps d'intégration nulle, et un rapport unités utilisateur/comptages de 1:1, qui sont les valeurs par défaut). La vitesse étant la distance parcourue en une seconde, l'erreur de position sera égale à n'importe quelle vitesse commandée dans les conditions de cet exemple.

2.2.2.2. Mode de boucle ouverte

Pour les applications ne nécessitant pas de retour ou utilisant des systèmes de positionnement très rudimentaires, vous pouvez utiliser le mode de boucle ouverte. Pour sélectionner ce mode, vous devez régler à zéro la constante de temps de boucle de position, ce qui indique que la boucle de positionnement n'est jamais fermée. Notez que, en mode de boucle fermée, la seule façon de créer un mouvement est de programmer une anticipation de vitesse non nulle. Le mode de boucle ouverte ignore tous les retours. L'erreur de position étant basée sur le retour, elle n'est plus utilisée pour générer des mouvements.

2.2.3. Unités utilisateur et comptages

Le module APM comporte une très puissante fonction de mise à l'échelle. Vous pouvez configurer le rapport unités utilisateur/comptages de façon à permettre la programmation en comptages différents des comptages par défaut. Pour donner un exemple simplifié, supposons qu'une application de retour de codeur utilise un codeur produisant 1000 comptages par rotation (250 lignes) et commandé par une machine produisant un pouce par rotation. L'unité par défaut serait d'un pouce par comptage. Cependant, si vous souhaitez écrire des programmes et utiliser le module APM avec des unités métriques, vous pouvez configurer un rapport de 2540 unités utilisateur pour 1000 comptages. Avec ce rapport, un mouvement de 254 unités dans l'un ou l'autre sens produit 100 comptages. L'unité serait de 0,01 mm par unité utilisateur.

La plage des unités utilisateur et des comptages est de 1 à 65535. Cependant, le rapport Unités utilisateur/comptages doit être compris entre 8:1 et 1:32. Vous trouverez ci–dessous un exemple plus détaillé.

Exemple :

Une machine comporte un moteur avec un codeur de 2000 lignes connecté, avec une réduction par engrenage de 20:1, à un engrenage droit de 14,336 pouces de diamètre. Le programmeur souhaite programmer une résolution de 0,01 pouce.

Données :

Engrenage droit de 14,336 pouces de diamètre

Codeur de 2000 lignes

Réduction par engrenage de 20:1

Unité de programmation souhaitée : 0,01 pouce

Premièrement, déterminez le nombre de comptages par rotation.

2000 lignes rotation * 4 * 20 = 160000 comptages par rotation

Le 4 provient du codeur générant 4 comptages par ligne

Le 20 provient de la réduction par engrenage

Ensuite, déterminez le nombre d'unités utilisateur par rotation en utilisant une résolution de 0,01 pouce.

14,336 pouces * π = 45,0378 pouces par rotation

45,0378 pouces par rotation / 0,01 pouce par unité utilisateur = 4503,78 unités utilisateur par rotation

Ce rapport unités utilisateur/comptages serait de 4503/160000 ou 0,02815, c'est–à–dire environ 1:36. Ce rapport est trop petit. Il n'est donc pas valide. Modifier le nombre de comptages par rotation nécessiterait un nouveau matériel. Nous devons donc régler la résolution. Supposons que la nouvelle unité utilisateur soit 0,001 pouce.

Déterminez les nouvelles unités utilisateur par rotation en utilisant une résolution de 0,001 pouce.

14,336 pouces * π = 45,0378 pouces par rotation

45,0378 pouces par rotation / 0,001 pouce par unité utilisateur = 45037,8 unités utilisateur par rotation

Ce rapport unités utilisateur/comptages serait de 45037/160000 ou 0,2815, c'est–à–dire environ 1:3,6. Nous devons donc utiliser une valeur de 2815 pour les unités utilisateur et de 10000 pour les comptages pour obtenir une résolution de 0,001.

2.2.4. Mode d'accélération de mouvement

Le module APM supporte deux types de mouvement : linéaire et courbe en S. Le mouvement linéaire est composé d'accélérations et de décélérations linéaires constantes jusqu'aux vitesses spécifiées. Un graphique présentant la vitesse en fonction du temps comporterait des droites. Les mouvements en courbe en S utilisent une accélération variable. Une accélération peut avoir une valeur peu élevée au départ et croître jusqu'à l'accélération spécifiée pour ensuite décroître jusqu'à à zéro lorsque la vitesse spécifiée est atteinte. Les mouvements en courbe en S nécessitent un temps et une distance doubles par rapport aux mouvements linéaires, mais ils diminuent les secousses subies par les servocommandes au démarrage et à l'arrêt tout en permettant des accélérations élevées.

Le mode d'accélération de chaque mouvement programmé, excepté la condition d'arrêt de saut décrite au chapitre 6 (Mouvement programmé), est spécifié dans la commande de mouvement. Mouvement (*Jog*), recherche de la position initiale (*Find Home*), interruption (*Abort*), et mouvement à vitesse constante (*Move at Velocity*) utilisent le mode d'accélération de mouvement configuré.

2.3. REMARQUES IMPORTANTES RELATIVES À LA CONFIGURATION

L'intégrateur est utilisé pour supprimer toutes les erreurs de position et peut être utilisé dans deux modes : EN ZONE et CONTINU. La constante de temps d'intégration est la durée nécessaire pour supprimer 63% de l'erreur de position. En mode EN ZONE, à la fin d'un mouvement, lorsque la position réelle est proche de la position commandée et que la servocommande est sur le point de s'arrêter, l'axe peut s'arrêter un peu plus tôt si la servocommande ne peut supprimer toute l'erreur. Pour augmenter la précision, l'intégrateur peut alors supprimer toute l'erreur. En mode CONTINU, l'intégrateur essaie en permanence de supprimer toute l'erreur de position, même pendant un mouvement. Des vitesses évoluant rapidement peuvent entraîner une sur– ou une sous–compensation de l'intégrateur. Le mode CONTINU est déconseillé pour les applications de positionnement. La constante de temps d'intégration doit être de cinq à dix fois supérieure à la c*onstante de temps de boucle de position* pour éviter tout risque d'oscillation.

Les limites de fin de course logicielles, positive et négative, sont utilisées pour limiter les mouvements commandés. Le module APM n'exécute pas les mouvements programmés qui utilisent une position commandée supérieure ou égale à la limite de fin de course. Si le bit %I *Position valide* est à 1, le mouvement cesse dès la limite de fin de course atteinte. Si ce bit est à 0, les mouvements et les commandes de mouvement à vitesse constante ignorent les limites de fin de course. Ceci permet de réaliser des mouvements au–delà des fins de course lors de la configuration d'un système.

Les limites de comptage inférieure et supérieure peuvent être utilisées pour les mouvements de type rotatif dans lesquels une servocommande peut se déplacer indéfiniment dans l'un ou l'autre sens. Lorsqu'une limite de comptage est atteinte, la position reportée "bascule" sur la limite opposée où elle peut continuer à évoluer. Si les limites de comptages sont définies comme étant égales ou inférieures aux limites de fin de course, celles–ci ne seront jamais atteintes et le mouvement ne s'arrêtera jamais. Les modules APM mono–axe avec une version de microprogramme 1.xx n'utilisent pas les limites de comptage. Vous devez donc définir celles–ci à zéro pour configurer le module APM.

Les interrupteurs de fin de course de dépassement sont des commutateurs externes au système. Leur état est transmis dans les bits CTL05 – CTL08. Vous pouvez configurer le module APM de façon à utiliser ces bits CTL en tant qu'interrupteurs de fin de course de dépassement. Si les commutateurs sont validés, le module APM commande une interruption immédiate de tous les mouvements sur l'axe chaque fois que le circuit de commande est activé et qu'un bit de dépassement est à 0. Dans ce cas, le système utilise simultanément les bits de mouvement (*Jog*) et d'effacement d'erreur (*Clear Error*) pour supprimer le dépassement de la servocommande.

Le module APM comporte une fonction de compensation d'inversion qui permet un positionnement précis sur les systèmes ayant du jeu. Le jeu est le petit déplacement que doit effectuer une servocommande avant que le système ne commence à se déplacer lorsqu'il change de direction. Par exemple, prenons le cas du verrou d'une porte à pêne fermant. Supposons que la servocommande contrôle la clé dans le verrou et que le retour indique les mouvements du pêne. Lorsque la servocommande tourne la clé dans le sens inverse des aiguilles d'une montre, le pêne se déplace vers la gauche. Cependant, si la servocommande tourne la clé dans le sens des aiguilles d'une montre, le pêne ne se déplace pas tant que la clé n'a pas atteint un certain point. La fonction de compensation d'inversion ajoute le mouvement nécessaire pour déplacer la servocommande jusqu'au point où le mouvement commence sur le dispositif de régulation. Le module APM supprime la compensation lorsqu'un mouvement dans le sens négatif est commandé et ajoute la compensation avant les mouvements dans le sens positif.

Le temps de désactivation du circuit de commande spécifie le temps que le module APM doit attendre, après une commande de vitesse nulle, avant de désactiver la sortie d'activation du circuit. L'ouverture du relais d'activation du circuit de commande fait que le module APM cesse de commander la servocommande. Ce relais doit donc parfois rester fermé. Imaginons par exemple que le mouvement entraîne la servocommande jusqu'à une limite de fin de course ; si le relais d'activation du circuit de commande était immédiatement désactivé à cause de l'erreur, la servocommande continuerait sur sa lancée jusqu'à atteindre un arrêt. Pour permettre au module APM de commander et de contrôler un arrêt rapide, le temps de désactivation doit être supérieur au temps de décélération de la servocommande à sa vitesse maximale.

Chapitre 4 Transferts automatiques de données

Ce chapitre définit les données qui sont transférées automatiquement lors de chaque cycle, sans programmation utilisateur, entre l'UC et le module APM. Ces données sont classées dans les catégories suivantes :

- **Données d'état en entrée (transférées du module APM vers l'UC)**
  - Bits d'état : 32 bits de données (%I)
  - Mots d'état : 15 mots de données (%AI) pour un module APM mono–axe
    28 mots de données (%AI) pour un module APM à deux axes
- **Données de commande en sortie (transférées de l'UC vers le module APM)**
  - Commandes logiques : 32 bits de données (%Q)
  - Commandes immédiates : 6 mots de données (%AQ)

1. BITS D'ÉTAT %I

Les bits d'état %I suivants sont transférés automatiquement du module APM vers l'UC lors de chaque cycle. Les adresses réelles des bits d'état dépendent de l'adresse de départ configurée pour les références %I (voir tableau 3–1, Données de configuration du module). Les numéros de bits listés dans le tableau suivant sont des décalages par rapport à l'adresse de départ.

Tableau 4-1. Bits d'état %I du module APM mono–axe (IC693APU301)

Bit *	Description	Bit *	Description
00	Axe prêt Axe 1	16	Etat de l'entrée CTL01 de la façade (état d'échantillonnage)
01	Position valide Axe 1	17	Etat de l'entrée CTL02 de la façade
02	Circuit de commande activé Axe 1	18	Etat de l'entrée CTL03 de la façade (commutateur position initiale)
03	Programme actif Axe 1	19	Etat de l'entrée CTL04 de la façade
04	En mouvement Axe 1	20	Etat de l'entrée CTL05 de la façade (dépassement positif)
05	En zone Axe 1	21	Etat de l'entrée CTL06 de la façade (dépassement négatif)
06	Indicateur d'échantillonnage de position Axe 1	22	Etat de l'entrée CTL07 de la façade
07	Erreur hors limite Axe 1	23	Etat de l'entrée CTL08 de la façade
08	Réservé	24	Réservé
09	Réservé	25	Réservé
10	Réservé	26	Réservé
11	Réservé	27	Réservé
12	Réservé	28	Réservé
13	Réservé	29	Réservé
14	Réservé	30	Contrôle par l'API actif
15	Réservé	31	Erreur

* Les numéros de bits représentent un décalage par rapport à l'adresse de départ des références %I.

Tableau 4-2. Bits d'état %I du module APM à deux axes (IC693APU302)

Bit *	Description	Bit *	Description
00	Axe prêt Axe 1	16	Etat de l'entrée CTL01 de la façade (état échantillonnage 1)
01	Position valide Axe 1	17	Etat de l'entrée CTL02 de la façade (état échantillonnage 2)
02	Circuit de commande activé Axe 1	18	Etat de l'entrée CTL03 de la façade (commutateur position initiale 1)
03	Programme actif Axe 1	19	Etat de l'entrée CTL04 de la façade (commutateur position initiale 2)
04	En mouvement Axe 1	20	Etat de l'entrée CTL05 de la façade (dépassement positif 1)
05	En zone Axe 1	21	Etat de l'entrée CTL06 de la façade (dépassement négatif 1)
06	Indicateur d'échantillonnage de position Axe 1	22	Etat de l'entrée CTL07 de la façade (dépassement positif 2)
07	Erreur hors limite Axe 1	23	Etat de l'entrée CTL08 de la façade (dépassement négatif 2)
08	Axe prêt Axe 2	24	Réservé
09	Position valide Axe 2	25	Réservé
10	Circuit de commande activé Axe 2	26	Réservé
11	Programme actif Axe 2	27	Réservé
12	En mouvement Axe 2	28	Réservé
13	En zone Axe 2	29	Réservé
14	Indicateur d'échantillonnage de position Axe 2	30	Contrôle par l'API actif
15	Erreur hors limite Axe 2	31	Erreur

* Les numéros de bits représentent un décalage par rapport à l'adresse de départ des références %I.

Axe prêt (Axis Enabled). Le bit d'état *Axe prêt* est à 1 lorsque l'APM est prêt à recevoir des commandes et à contrôler un servomécanisme. Il est mis à 0 lorsqu'une condition d'erreur arrête le servomécanisme.

Circuit de commande activé (Drive Enabled). Le bit d'état *Circuit de commande activé* indique l'état de la commande logique *Activation circuit de commande* et du contact de relais fournis par l'APM. Le bit d'état *Circuit de commande activé* est à 1 pour indiquer que le contact de relais est à l'état fermé. Il est mis à zéro après un démarrage ou la détection d'une condition d'erreur arrêtant le servomécanisme.

Erreur (Error). Ce bit d'état est mis à 1 lorsque l'APM détecte une erreur. Le mot d'état %AI *Code d'état* identifie alors la condition d'erreur. *Effacement erreur* est la seule commande qui peut remettre à zéro le bit d'état *Erreur* et le mot *Code d'état* associé. Cependant, elle ne peut remettre à 0 le bit d'état *Erreur* si la condition qui a entraîné l'erreur est toujours présente.

Etat des entrées du plastron CTL01–08. Ces entrées indiquent toujours l'état des capteurs externes connectés aux bornes CTL01–08 du plastron de l'APM. L'APM peut tester ces entrées (ainsi que CTL09–CTL12 dans la table d'API %Q) lors de l'exécution de commandes d'attente et de saut conditionnel.

Plusieurs entrées peuvent avoir des utilisations différentes :

CTL01	Entrée d'échantillonnage axe 1	CTL02	Entrée échantillonnage 1 axe 2
CTL03	Entrée commutateur de position initiale axe 1	CTL04	Entrée commutateur position initiale axe 2
CTL05	Entrée commutateur dépassement (+) axe 1	CTL07	Entrée commutateur dépassement (+) axe 2
CTL06	Entrée commutateur dépassement (–) axe 1	CTL08	Entrée commutateur dépassement (–) axe 2

Erreur hors limite ("In Error" Limit). Le bit d'état *Erreur hors limite* est à 1 lorsque la valeur absolue de l'erreur de position excède la valeur d'erreur de position (*Position Error Limit*) configurée. Lorsqu'il est à 1, la vitesse et la position commandées sont "gelées" afin de permettre à l'axe de combler son retard par rapport à la position commandée.

En zone (In Zone). Le bit d'état *En zone* indique que l'erreur de position est inférieure ou égale à la valeur *Zone En Pos* configurée. Cette condition apparaît à la fin de chaque commande de *Mouvement de positionnement* et chaque fois que les commandes d'axe sont bloquées et que la position réelle parvient à égalité avec la position commandée (par exemple pour l'*Arrêt momentané*, la *pause*, ou un pourcentage de forçage de vitesse d'avance nul).

En mouvement (Moving). Le bit d'état *En mouvement* est à 1 lorsque la vitesse commandée est non nulle. Il est à zéro dans tous les autres cas. Les commandes *Pmove*, *Cmove*, *Mouvement positif*, *Mouvement négatif* et *Mouvement à vitesse constante* entraînent la mise à 1 du bit *En mouvement*. La commande *Forçage sortie N/A* ne met pas à 1 le bit *En mouvement*.

Contrôle par l'API actif (PLC Control Active). N'est pas actuellement mis en oeuvre. Le bit d'état *Contrôle par l'API actif* est normalement à 1, ce qui indique que les commandes logiques %Q ou immédiates %AQ de l'API contrôlent l'APM. Le bit *Contrôle par l'API actif* est mis à 0 uniquement lorsque vous utilisez l'écran d'état (Status) du logiciel Motion Programmer à la place de l'API pour contrôler l'APM.

Indicateur d'échantillonnage de position (Position Strobe). Le bit d'état *Indicateur d'échantillonnage de position* indique que l'entrée d'échantillonnage du connecteur d'E/S a acquis une position d'axe, indiquée par le mot d'état %AI *Position d'échantillonnage*. La donnée est conservée dans le mot d'état *Position d'échantillonnage* jusqu'à ce que le bit *Indicateur d'échantillonnage de position* soit mis à 0 par le bit %Q *Remise à zéro de l'indicateur d'échantillonnage*. Une fois le bit *Indicateur d'échantillonnage de position* à 0, vous pouvez acquérir une nouvelle donnée en utilisant une nouvelle entrée d'échantillonnage.

Position valide (Position Valid). Le bit d'état *Position valide* indique que la valeur contenue dans le mot d'état %AI *Position réelle* a été initialisée par une commande *Réglage position* ou par une exécution complète d'un cycle de *recherche position initiale*. L'exécution d'un programme de mouvement nécessite que le bit *Position valide* soit à 1.

Programme actif (Program Active). Le bit d'état *Programme actif* d'un axe indique qu'un programme de mouvement (0–10) s'exécute sur l'axe. Sur un APM à deux axes, l'exécution d'un programme multi–axe met à 1 les deux bits *Programme actif*.

2. MOTS D'ÉTAT %AI

Les mots d'état %AI suivants sont transférés automatiquement de l'APM à l'UC lors de chaque cycle.

Les adresses réelles des mots d'état dépendent de l'adresse de départ configurée pour les références %AI (voir tableau 3–1, Données de configuration du module). Les numéros de mots listés dans le tableau suivant sont des décalages par rapport à l'adresse de départ.

Tableau 4-3. Mots d'état %AI du module APM mono–axe (IC693APU301)

Mot*	Description	
000	Code d'état	
001	Numéro de bloc de commande	Axe 1
002–003	Position commandée	Axe 1
004–005	Position réelle	Axe 1
006–007	Position d'échantillonnage	Axe 1
008–009	Vitesse commandée	Axe 1
010–011	Vitesse réelle	Axe 1
012–013	Erreur de position**	Axe 1
014	Valeur d'entrée analogique**	

* Les numéros de mots représentent un décalage par rapport à l'adresse de départ des références %AI.

** Pour assurer la compatibilité avec les versions précédentes de ce produit, vous pouvez configurer la valeur d'entrée analogique avec le mot %AI 012, "Erreur de position" étant désactivé.

Tableau 4-4. Mots d'état %AI du module APM à deux axes (IC693APU302)

Mot*	Description	
000	Code d'état	
001	Numéro de bloc de commande	Axe 1
002–003	Position commandée	Axe 1
004–005	Position réelle	Axe 1
006–007	Position d'échantillonnage	Axe 1
008–009	Vitesse commandée	Axe 1
010–011	Vitesse réelle	Axe 1
012–013	Erreur de position	Axe 1
014	Valeur d'entrée analogique	
015	Numéro de bloc de commande	Axe 2
016–017	Position commandée	Axe 2
018–019	Position réelle	Axe 2
020–021	Position d'échantillonnage	Axe 2
022–023	Vitesse commandée	Axe 2
024–025	Vitesse réelle	Axe 2
026–027	Erreur de position	Axe 2

* Les numéros de mots représentent un décalage par rapport à l'adresse de départ des références %AI.

Position réelle (Actual Position). La *Position réelle* (unités utilisateur) est une valeur gérée par l'APM qui représente la position physique de l'axe. Elle est définie à une valeur initiale par la commande *Réglage position* ou à la valeur de position initiale par le cycle de *recherche position initiale*. Elle est mise à jour par le mouvement des dispositifs de régulation.

Si la valeur de la *position réelle* dépasse l'une des limites de comptage, elle boucle sur l'autre limite et continue le comptage dans le sens du mouvement de l'axe.

Vitesse réelle (Actual Velocity). La *Vitesse réelle* (unités utilisateur/s) est une valeur gérée par l'APM qui est calculée à partir du dispositif de régulation. Elle représente donc la vitesse du mouvement de l'axe.

Entrée analogique (Analog Input). *Entrée analogique* renvoie la valeur numérique représentant la tension appliquée à la borne d'entrée analogique. Une tension de +10 V est indiquée par +32 000, et une tension de –10 V est indiquée par –32 000.

Numéro de bloc de commande (Command Block Number). Le *Numéro de bloc de commande* indique le numéro du bloc de la commande en cours d'exécution dans le programme ou le sous–programme actif. Il est modifié au début de chaque nouveau bloc au fur et à mesure de l'exécution des commandes du programme. Il identifie donc en permanence la partie en cours d'exécution du programme.

Position commandée (Commanded Position). La *Position commandée* (unités utilisateur) représente à tout instant la position où l'axe doit se trouver. La différence entre la *position commandée* et la *position réelle* est l'*erreur de position*, qui génère une commande de vitesse pour contrôler l'axe. Le taux d'incrémentation/décrémentation de la *position commandée* détermine la vitesse du mouvement de l'axe.

Si la p*osition commandée* excède l'une des limites de comptage, elle boucle sur l'autre limite et continue dans le sens du mouvement de l'axe.

Erreur de position (Position Error). L'*Erreur de position* (unités utilisateur) est égale à tout instant à la différence entre la *position commandée* et la *position réelle*.

Vitesse commandée (Commanded Velocity). La *Vitesse commandée* (unités utilisateur) est une valeur générée par l'APM qui indique la commande de vitesse instantanée qui produit le mouvement de l'axe. Elle augmente au taux d'accélération au début d'un mouvement pour se stabiliser ensuite à la vitesse programmée.

Code d'état (Status Code). Le *Code d'état* indique l'état de fonctionnement du module. Lorsque le bit d'état *Erreur* est à 1, ce mot contient le numéro du code d'erreur correspondant à la condition qui a entraîné l'erreur.

Reportez–vous à l'annexe A, Codes d'erreur, pour une liste des codes d'erreur APM.

Position d'échantillonnage (Strobe Position). La *Position d'échantillonnage* (unités utilisateur) contient la position de l'axe au moment où une entrée d'échantillonnage se présente. Lorsqu'une entrée d'échantillonnage se présente, *l'indicateur d'échantillonnage de position* est mis à 1 pour indiquer à l'API qu'une nouvelle donnée d'échantillonnage est disponible dans le mot d'état *Position d'échantillonnage*. Après avoir accusé réception, l'API met à 1 la commande *remise à zéro de l'indicateur d'échantillonnage* pour remettre à 0 *l'indicateur d'échantillonnage de position*.

La valeur de *Position d'échantillonnage* est conservée et n'est pas modifiée par de nouvelles entrées d'échantillonnage jusqu'à remise à 0 de *l'indicateur d'échantillonnage de position*. Si la commande *Remise à 0 de l'indicateur d'échantillonnage* est maintenue à 1 (ce qui maintient *l'indicateur d'échantillonnage de position* à 0), chaque nouvelle entrée d'échantillonnage entraîne l'acquisition de la position de l'axe dans le mot d'état *Position d'échantillonnage*.

La *Position d'échantillonnage* de l'axe 1 est placée dans le paramètre 255. Dans module APM à 2 axes, la *Position d'échantillonnage* de l'axe 2 est placée dans le paramètre 254.

3. COMMANDES LOGIQUES %Q

Les sorties %Q suivantes représentent des commandes logiques automatiquement transmises par l'UC à l'APM lors de chaque cycle. Pour exécuter une commande, il suffit de mettre à 1 le bit de sortie correspondant.

Les adresses réelles des bits de commandes logiques dépendent de l'adresse de départ configurée pour les références %Q (voir tableau 3–1, Données de configuration du module). Les numéros de bits listés dans le tableau suivant sont des décalages par rapport à l'adresse de départ.

Tableau 4-5. Commandes logiques %Q du module APM mono–axe (IC693APU301)

Bit *	Description		Bit *	Description
00	Interruption de tous les mouvements	Axe 1	16	Contrôle de sortie CTL09
01	Pause	Axe 1	17	Contrôle de sortie CTL10
02	Activation circuit de commande	Axe 1	18	Contrôle de sortie CTL11
03	Recherche position initiale	Axe 1	19	Contrôle de sortie CTL12
04	Mouvement positif (Jog Plus)	Axe 1	20	Exécution programme de mouvement 0
05	Mouvement négatif (Jog Minus)	Axe 1	21	Exécution programme de mouvement 1
06	Remise à zéro de l'indicateur d'échantillonnage	Axe 1	22	Exécution programme de mouvement 2
07	Réservé		23	Exécution programme de mouvement 3
08	Réservé		24	Exécution programme de mouvement 4
09	Réservé		25	Exécution programme de mouvement 5
10	Réservé		26	Exécution programme de mouvement 6
11	Réservé		27	Exécution programme de mouvement 7
12	Réservé		28	Exécution programme de mouvement 8
13	Réservé		29	Exécution programme de mouvement 9
14	Réservé		30	Exécution programme de mouvement 10
15	Réservé		31	Effacement erreur

* Les numéros de bit représentent un décalage par rapport à l'adresse de départ des références %Q.

Tableau 4-6. Commandes logiques %Q du module APM à deux axes (IC693APU302)

Bit *	Description		Bit *	Description
00	Interruption de tous les mouvements	Axe 1	16	Contrôle de sortie CTL09
01	Pause	Axe 1	17	Contrôle de sortie CTL10
02	Activation circuit de commande	Axe 1	18	Contrôle de sortie CTL11
03	Recherche position initiale	Axe 1	19	Contrôle de sortie CTL12
04	Mouvement positif (Jog Plus)	Axe 1	20	Exécution programme de mouvement 0
05	Mouvement négatif (Jog Minus)	Axe 1	21	Exécution programme de mouvement 1
06	Remise à zéro de l'indicateur d'échantillonnage	Axe 1	22	Exécution programme de mouvement 2
07	Réservé		23	Exécution programme de mouvement 3
08	Interruption de tous les mouvements	Axe 2	24	Exécution programme de mouvement 4
09	Pause	Axe 2	25	Exécution programme de mouvement 5
10	Activation circuit de commande	Axe 2	26	Exécution programme de mouvement 6
11	Recherche position initiale	Axe 2	27	Exécution programme de mouvement 7
12	Mouvement positif (Jog Plus)	Axe 2	28	Exécution programme de mouvement 8
13	Mouvement négatif (Jog Minus)	Axe 2	29	Exécution programme de mouvement 9
14	Remise à zéro de l'indicateur d'échantillonnage	Axe 2	30	Exécution programme de mouvement 10
15	Réservé		31	Effacement erreur

* Les numéros de bit représentent un décalage par rapport à l'adresse de départ des références %Q.

Interruption de tous les mouvements (Abort All Moves). Cette commande bloque tous les mouvements en cours à leur taux d'**accélération de mouvement** actif. Toutes les commandes programmées ou immédiates en attente sont annulées et ne peuvent être mises en oeuvre. La condition d'interruption est active tant que ce signal est présent. Si un mouvement était en cours lors de la réception de la commande, le bit d'état *En mouvement* reste à 1 et le bit d'état *En Zone* reste à 0 jusqu'à ce que la vitesse commandée atteigne zéro et que la condition *En Zone* soit atteinte.

Effacement erreur (Clear Error). Si une condition d'erreur est signalée, cette commande permet de mettre à 0 le bit d'état *Erreur* et le mot *Code d'état* associé. Les conditions d'erreurs qui sont toujours présentes (par exemple une erreur d'interruption *Fin de course*) ne sont pas effacées. Vous devez les effacer avec les actions de correction appropriées.

Contrôles de sortie CTL09–CTL12. Ces bits de commande contrôlent les sorties numériques CMOS sur le connecteur d'E/S B du plastron. L'APM peut également les tester pendant l'exécution d'une commande d'attente (*Wait*) ou de saut conditionnel (*Jump*).

Activation circuit de commande (Enable Drive). Si les bits d'état *Erreur* et *Circuit de commande activé* sont à 0, cette commande entraîne la fermeture du contact de relais d'activation du circuit de commande (ce qui active le circuit de commande) et la mise à 1 du bit *Circuit de commande activé* ; dans tous les autres cas, cette commande n'a aucun effet. Si le bit *Circuit de commande activé* est à 1, les fonctions de génération de trajectoire et de contrôle de position sont activées et les commandes de mouvement du programme peuvent être exécutées. La commande *Activation circuit de commande* doit être maintenue à 1 pour permettre un mouvement normal du servomécanisme (excepté lors de l'utilisation de commandes *Jog*).

Exécution programme de mouvement (Execute Motion Program) 0–10. Ces commandes sont utilisées pour sélectionner un programme enregistré et l'exécuter immédiatement. Chaque commande utilise une action ponctuelle ; un bit de commande doit passer de OFF à ON chaque fois que le programme doit être exécuté. Les programmes peuvent être temporairement interrompus par une commande *Pause*.

Un seul programme peut être exécuté sur un axe à un instant donné ; le bit d'état %I *Programme actif* doit être à zéro pour que l'exécution du programme de mouvement puisse commencer. Un programme de mouvement multi–axe utilise à la fois l'axe 1 et l'axe 2 ; les deux bits %I *Programme actif* doivent donc être à zéro pour qu'un programme de mouvement multi–axe puisse commencer.

Pause (début) (Feedhold (On Transition)). Cette commande arrête tout programme de mouvement en cours à son taux d'accélération actif. Une fois le mouvement arrêté, le bit d'état *En mouvement* est mis à 0 et le bit d'état *En Zone* est mis à 1 lorsque la condition "En Zone" est atteinte. Les commandes de mouvement *Jog* sont autorisées pendant une pause. En cas d'activation de la commande, le mouvement du programme s'interrompt, même si le bit de la commande repasse à zéro avant l'arrêt du mouvement.

Pause (fin) (Feedhold (Off Transition)). Cette commande entraîne la reprise de tout mouvement programmé arrêté par la commande *Pause (début)*, avec les taux d'accélération et de vitesse programmés. Les mouvements supplémentaires du programme sont ensuite traités et l'exécution normale du programme continue.

Si un mouvement (*Jog*) s'est produit pendant l'activation de la pause, la commande de mouvement reprendra à partir de la position atteinte après ce mouvement. Le mouvement se termine à la vitesse programmée correcte et continue jusqu'à la position programmée initialement, comme si aucun mouvement *Jog* ne s'était produit.

Recherche position initiale (Find Home). Cette commande est utilisée pour que l'APM établisse la position initiale (*Home position*) des systèmes utilisant un dispositif de régulation incrémentiel fournissant également une impulsion de marquage. Une transition du commutateur de limite de position initiale en provenance du connecteur d'E/S indique approximativement l'emplacement de référence de la position initiale ; le prochain marqueur rencontré pendant un déplacement dans le sens négatif indique l'emplacement exact. Le décalage de position initiale (*Home Offset*) configuré définit l'emplacement de la position initiale (*Home position*) comme étant la distance de décalage par rapport au marqueur de position initiale.

Reportez–vous au chapitre 6, **Contrôle du mouvement de l'APM** pour une description du cycle de recherche de position initiale.

Mouvement négatif (Jog Minus). Lorsque ce bit de commande est à 1, l'axe se déplace dans le sens négatif à la vitesse (*Jog Velocity*) et à l'accélération (*Jog Acceleration*) de mouvement configurées. Le mouvement continue tant que la commande *Mouvement négatif* est active et que la fin de course négative (*Negative End Of Travel)* configurée n'est pas rencontrée. Associée au bit *Effacement erreur,* la commande *Mouvement négatif*, permet de s'éloigner de la position qui ouvre le commutateur de limite de fin de course positive.

Mouvement positif (Jog Plus). Lorsque ce bit de commande est à 1, l'axe se déplace dans le sens positif à la vitesse (*Jog Velocity*) et à l'accélération (*Jog Acceleration*) de mouvement configurées. Le mouvement continue tant que la commande *Mouvement positif* est active et que la limite de fin de course positive n'est pas rencontrée. Associée au bit *Effacement erreur,* la commande *Mouvement positif*, permet de s'éloigner de la position qui ouvre le commutateur de limite de fin de course négative.

Remise à zéro de l'indicateur d'échantillonnage (**Reset Strobe Flag).** Le bit d'état %I *Indicateur d'échantillonnage* informe l'API qu'une entrée d'échantillonnage a acquis une position d'axe, enregistrée dans le mot d'état *Position d'échantillonnage*. Lorsque l'API accuse réception de cette donnée, il peut utiliser le bit %Q de *Remise à zéro de l'indicateur d'échantillonnage* pour remettre à 0 le bit d'état *Indicateur d'échantillonnage*. Si le bit *Indicateur d'échantillonnage* est à 1, les entrées d'échantillonnage suivantes n'entraînent pas l'acquisition de nouvelles données. L'indicateur doit être effacé pour qu'une nouvelle position d'échantillonnage soit enregistrée. Le bit *Indicateur d'échantillonnage* est maintenu à 0 tant que le bit de commande %Q *Remise à zéro de l'indicateur d'échantillonnage* est à 1. Dans cette condition, le mot d'état *Position d'échantillonnage* contient la dernière position d'entrée d'échantillonnage, bien que l'API ne puisse utiliser l'indicateur pour signaler la présence d'une nouvelle donnée.

4. COMMANDES IMMÉDIATES %AQ

L'UC envoie automatiquement 6 mots de données %AQ à l'APM lors de chaque cycle. Ces mots sont utilisés pour transmettre des commandes immédiates de l'API à l'APM. Les trois premiers mots, décalages 0 à 2, sont dédiés à l'axe 1 de l'APM. Les trois mots suivants, décalages 3 à 5, sont dédiés à l'axe 2 de l'APM. Il est donc possible de transmettre une commande à chaque axe de l'APM lors de chaque cycle.

La commande *Chargement immédiat de paramètre*, qui est indépendante de l'axe, est la seule exception. Vous pouvez transmettre cette commande en utilisant aussi bien le premier que le deuxième jeu de trois mots. Vous pouvez également transmettre deux commandes *Chargement immédiat de paramètre* au cours du même cycle (l'une avec les trois premiers mots %AQ, l'autre avec les trois mots %AQ suivants).

Pour un module APM mono–axe, les seules commandes effectives du deuxième jeu de mots %AQ sont *Forçage sortie N/A* (sortie analogique 2) et *Chargement immédiat de paramètre*.

Bien que les commandes soient transmises lors de chaque cycle, l'APM ne répond à une commande que si elle est différente de la commande du cycle précédent. Si l'un des six octets de données est modifié, l'APM accepte les données en tant que nouvelle commande et répond de la façon appropriée.

Les commandes immédiates ont un format de 6 octets, comme indiqué ci–dessous. Les adresses réelles des mots de commandes immédiates dépendent de l'adresse de départ configurée pour les références %AQ (voir tableau 3–1, Données de configuration du module). Les numéros de mots listés dans le tableau suivant sont des décalages par rapport à l'adresse de départ.

Les décalages des mots sont indiqués dans l'ordre inverse et en hexadécimal afin de simplifier l'entrée des données. L'exemple suivant transmet la commande *Réglage position* à l'axe 1. Le premier mot, le mot 0, contient le numéro réel de la commande. Pour la commande *Réglage position*, ce numéro est 0023h. Le deuxième et le troisième mots contiennent les données de la commande *Réglage position*, c'est–à–dire une position. Le deuxième mot, le mot 1, est le mot de poids faible de la position et le troisième mot, le mot 2, est le mot de poids fort. Pour régler une position de 3400250, convertissez cette valeur en hexadécimal. Vous devez obtenir 0033E23A. Pour cette valeur, 0033 est le mot de poids fort et E23A est le mot de poids faible. Les données à transmettre à l'APM sont :

Mot 2	Mot 1	Mot 0	Commande
0033	E23A	0023	Réglage position 3400250

Tableau 4-7. Commandes immédiates avec le format de 6 octets (Module APM à 1 ou 2 axes)

Mots 2, 5		Mots 1, 4		Mots 0, 3		
Octet 5	**Octet 4**	**Octet 3**	**Octet 2**	**Octet 1**	**Octet 0**	**Définition de la commande immédiate**
xx	xx	xx	xx	00	00h	**Null**
xx	xx	xx	Taux	00	20h	**Forçage taux** Taux = de 0 à 120 %
xx	xx	xx	Incr.	00	21h	**Incrément de position** Incr. = de –128 à +127 unités utilisateur
Vitesse				00	22h	**Mouvement à vitesse constante** Vit. = de –8388608 à +8388607 unités utilisateur/s
Position				00	23h	**Réglage position** Pos. = de –8388608 à +8388607 unités utilisateur
xx	xx	Sortie N/A		00	24h	**Forçage sortie N/A** Sortie N/A = –32000 à +32000
Vitesse				00	28h	**Vitesse de mouvement (Jog)** Vit. = de –8388608 à +8388607 unités utilisateur/s
Accélération				00	29h	**Accélération de mouvement (Jog)** Acc. = de –8388608 à +8388607 unités utilisateur/s/s
xx	xx	Constante de temps		00	2Ah	**Constante de temps de boucle de position** Constante de temps = 0 et de 5 à 10000
xx	xx	xx	Anticip. Vit.	00	2Bh	**Anticipation de vitesse** Anticip. Vit. = de 0 à 100 %
xx	xx	Constante de temps d'intégration		00	2Ch	**Constante de temps d'intégration** Constante de temps d'intégration = 0 et de 10 à 65535 ms
Données de paramètre				N° param.	50h	**Chargement immédiat de paramètre** N° param. = de 0 à 255 Données de paramètre = la plage dépend de l'utilisation du paramètre.

xx = sans intérêt ici.

Null. Commande immédiate %AQ par défaut. Les mots %AQ étant transférés à chaque cycle de l'API, vous devez toujours entrer une commande *Null* pour éviter l'exécution non prévue d'une autre commande immédiate.

Forçage sortie N/A (Force D/A Output). Cette commande force les sorties Numériques/Analogiques de commande de vitesse (*Velocity*) du connecteur d'E/S à un niveau de sortie constant. La tension de sortie est générée uniquement pendant le temps où le bit *Circuit de commande activé* est à 1 et où la commande *Forçage sortie N/A* est active. La commande *Forçage sortie N/A* règle la sortie dans une plage –10/+10 volts. Une commande de +32000 produit une sortie de +10 V et une commande de –32000 produit une sortie de –10 V.

Pour que la commande *Forçage sortie N/A* fonctionne, le bit %Q *Circuit de commande activé* doit être actif sans autre mouvement commandé. Cette commande est la seule commande immédiate %AQ continue. Elle doit rester dans les données %AQ pour que la commande continue de fonctionner. Toute autre commande immédiate %AQ interrompt Forçage N/A.

Constante de temps d'intégration (Integrator Time Constant). (millisecondes) Cette commande définit la *constante de temps d'intégration* pour l'intégrateur d'erreur de position. La constante de temps indique le nombre de millisecondes nécessaires pour supprimer 63 % de l'*Erreur de position*. La constante de temps doit être entre cinq et dix fois supérieure à la *Constante de temps de boucle de position* pour éviter l'instabilité et l'oscillation.

Chargement immédiat paramètre (Load Parameter Immediate). Cette commande est exécutée à partir de l'API pour modifier immédiatement la valeur d'un paramètre APM. Les paramètres sont utilisés uniquement par les programmes de mouvement. Chaque modification de paramètre nécessite une commande.

Mouvement à vitesse constante (Move At Velocity). (unités utilisateur/s) Cette commande est exécutée à partir de l'API pour obtenir un mouvement de l'axe à une vitesse constante. Les commandes *Mouvement à vitesse constante* utilisent le taux *d'accélération de mouvement* (*Jog Acceleration*) configuré. Les données de position d'axe basculent de +8388607 à –8388608 (et inversement) lorsque cette valeur est atteinte en cours de mouvement.

Incrément de position (Position Increment). (unités utilisateur) Cette commande ajoute un décalage compris entre –128 et +127 unités utilisateur à la *position réelle*. La *Position réelle* est modifiée par la valeur de l'incrément ; l'APM tente donc de se déplacer pour prendre en compte la modification de position.

Constante de temps de boucle de position (Position Loop Time Constant). (millisecondes) Cette commande permet de donner à la constante de temps de boucle de position du servomécanisme une valeur différente de la valeur configurée. Plus la valeur est faible, plus le système répond rapidement. Les valeurs trop faibles entraînent une instabilité et une oscillation du système. Pour un suivi efficace du profil de vitesse commandée, la *constante de temps de boucle de position* doit être comprise entre 1/4 et 1/2 du temps de décélération MINIMUM du système.

Forçage taux (Rate Override). Cette commande modifie immédiatement le taux de forçage de vitesse d'avance (généralement appelé Manual Feedrate Override ou MFO). La nouvelle valeur est effective dès réception par l'APM. Elle est enregistrée et reste effective jusqu'à ce qu'elle soit écrasée par une valeur différente. Le forçage de taux n'a aucun effet sur les mouvements non programmés. Le taux est réglé à 100 % à chaque lancement de programme.

Réglage position (Set Position). (unités utilisateur) Cette commande modifie la valeur du registre de position de l'axe sans déplacer l'axe. Les valeurs *Position commandée* et *Position réelle* sont toutes deux modifiées de façon qu'aucune commande de mouvement ne soit générée. La *Position réelle* est définie à la valeur indiquée et la *Position commandée* est définie à cette valeur + *Erreur de position*. Vous ne pouvez pas exécuter de commande *Réglage position* lorsque le bit %I *En mouvement* ou le bit %I *Programme actif* est à 1.

Anticipation de vitesse (Velocity Feedforward). Cette commande définit le gain d'anticipation de vitesse (*Velocity Feedforward*). Ce gain est le pourcentage de la *vitesse commandée* qui est ajouté à la sortie de commande de vitesse de l'APM. L'augmentation d'*Anticipation de vitesse* permet au servomécanisme de fonctionner avec une réponse plus rapide et une erreur de position plus faible. Les valeurs optimales d'*Anticipation de vitesse* sont comprises entre 80 et 90 %. Vous devez régler la valeur "Vit à 10 V" correctement pour un fonctionnement correct du facteur de gain *Anticipation de vitesse*.

Page laissée blanche intentionnellement

Chapitre 5 *Procédures de démarrage d'un servomécanisme*

Pour le démarrage de votre servomécanisme, suivez la procédure suivante :

1. Connectez le moteur à l'amplificateur de système asservi en suivant les recommandations du fabricant.
2. Connectez le relais **d'activation du circuit de commande** de l'APM et les sorties de **commande de vitesse** à l'amplificateur de système asservi. Connectez le dispositif de régulation de position (codeur) aux entrées du codeur de l'APM.
3. Si vous utilisez des interrupteurs de **fin de course de dépassement** (24 Vcc), branchez–les sur les entrées correctes de l'APM. Si vous n'en utilisez pas, vous devez désactiver les limites de dépassement avec le logiciel de configuration Logicmaster 90. Si vous utilisez un commutateur de **position initiale** (24 Vcc), branchez–le sur les entrées correctes de l'APM. Le commutateur de **position initiale** doit être branché de façon à être TOUJOURS ACTIF (fermé) lorsque l'axe est sur le côté négatif de la position initiale et TOUJOURS INACTIF (ouvert) lorsque l'axe est sur le côté positif de la position initiale.
4. Utilisez le logiciel de configuration Logicmaster 90 pour définir les facteurs de mise à l'échelle utilisateur et autres paramètres configurables. Enregistrez la configuration dans l'API.
5. Effacez le programme de l'API, mettez à zéro tous les bits %Q de l'APM et placez l'API en mode RUN. Surveillez les bits %I **CTL03, 4 (position initiale)**, **CTL05, 7 (dépassement positif)** et **CTL06, 8 (dépassement négatif)** et vérifiez que chaque bit répond au commutateur correct.
6. Mettez à un le bit %Q *Activation circuit de commande* et placez le code de la commande *Forçage sortie N/A* 0 dans la table %AQ. Vérifiez que l'amplificateur de système asservi est activé. Si l'axe du moteur se déplace, réglez le décalage de l'amplificateur zéro jusqu'à ce que le moteur cesse tout mouvement. Remarque : vous devez maintenir le bit %Q *Activation circuit de commande* à 1 pour que la commande *Forçage sortie N/A* fonctionne.
7. Transmettez le code de la commande *Forçage sortie N/A* +3200 (+1 V). Vérifiez que l'axe du moteur se déplace dans le sens souhaité (POSITIF) et que la *Vitesse réelle* reportée dans la table %AI de l'APM est POSITIVE. Si l'axe du moteur se déplace dans le mauvais sens, consultez les instructions des fabricants pour connaître les actions de correction. Si l'axe se déplace dans le sens POSITIF et si l'APM signale une *Vitesse réelle* NEGATIVE, vous devez permuter les entrées des voies A et B.
8. Notez la vitesse réelle du moteur reportée par l'APM avec une commande de vitesse de 1 Volt. Multipliez cette vitesse par 10 et mettez à jour l'entrée *Vit à 10 V* dans la configuration de l'APM. Placez une valeur initiale élevée (généralement entre 100 et 1000 ms) dans l'entrée de configuration *TC Bouc Pos*.
9. Mettez à 1 le bit %Q de mouvement positif. Vérifiez que le sens du mouvement est correct et que la *Vitesse réelle* reportée par l'APM dans la table %AI correspond à la vitesse de mouvement configurée.
10. Le bit %Q *Circuit de commande activé* à 1, sans mouvement commandé pour la servocommande, réglez le décalage du servomoteur à zéro pour une erreur de position nulle (*Position commandée* APM = *Position réelle* APM). Pendant ce processus, l'intégrateur doit être désactivé.

11. Contrôlez la bonne exécution du cycle de *Recherche position initiale* en activant momentanément le bit %Q *Recherche position initiale* (vous devez également maintenir à 1 le bit %Q *Circuit de commande activé*). L'axe doit se déplacer en direction du commutateur de position initiale à la vitesse de *Recherche position initiale* configurée, puis rechercher le marqueur du codeur à la vitesse finale de position initiale configurée. Si nécessaire, réglez les vitesses configurées et l'emplacement du **commutateur de position initiale** pour un fonctionnement stable. La transition finale du **commutateur de position initiale** DOIT se produire au minimum 10 ms avant que l'impulsion du marqueur du codeur ne soit rencontrée. Vous pouvez régler la *position initiale* en modifiant la valeur du décalage de position initiale avec le logiciel de configuration Logicmaster 90.

12. Surveillez la performance de la servocommande et utilisez les bits %Q *Mouvement positif* (*Jog Plus*) et *Mouvement négatif* (*Jog Minus*) pour déplacer l'axe du servomoteur dans chaque sens. Vous pouvez modifier temporairement la *constante de temps de boucle de position* en plaçant le code de commande correct dans la table %AQ. Pour la plupart des systèmes, vous pouvez réduire la *constante de temps de boucle de position* jusqu'à l'apparition d'une certaine instabilité de la servocommande, puis l'augmenter d'environ 50 % de sa valeur. Après avoir déterminé la valeur correcte de la constante de temps, vous devez mettre à jour la configuration de l'APM en utilisant le logiciel de configuration Logicmaster 90. Pour une réponse optimale de la servocommande, vous pouvez également régler l'*anticipation de vitesse* à une valeur non nulle (généralement entre 80 et 90 %).

Remarque

Pour un fonctionnement correct de la servocommande, vous DEVEZ régler l'entrée de configuration *Vit à 10 V* sur la vitesse réelle de la servocommande (en unités utilisateur/s) générée par une commande de 10 V.

Chapitre 6 Contrôle de mouvement de l'APM

Ce chapitre fournit des informations pratiques sur le contrôle de mouvement de l'APM et plusieurs exemples. Les principaux sujets abordés sont :

- Mouvement non programmé
- Mouvement programmé

1. MOUVEMENT NON PROGRAMMÉ

Le module APM peut générer cinq types de mouvement d'axe sans nécessiter de programme de mouvement.

- *Recherche position initiale* et *Mouvement* (*Jog*) utilisent des bits %Q pour commander le mouvement.
- *Mouvement à vitesse constante*, *Forçage sortie N/A* et *Incrément de position* utilisent des commandes immédiates %AQ.

Pendant les mouvements *Jog*, *Recherche position initiale*, *Mouvement à vitesse constante* et *Forçage sortie N/A*, tous les autres mouvements commandés, programmés ou non, génèrent une erreur. La seule exception est *Incrément de position*, qui peut être utilisé à tout moment. Reportez–vous à la description du mouvement *Incrément de position* pour plus de détails.

1.1. CYCLE DE POSITION INITIALE APM

Un cycle de position initiale établit la position initiale des systèmes utilisant un dispositif de régulation incrémentiel fournissant également une impulsion de marquage. Le décalage de position initiale configuré définit l'emplacement de la position initiale comme étant la distance de décalage par rapport au marqueur de position initiale.

Le bit %I *Circuit de commande activé* doit être à 1 pendant un cycle de position initiale entier. Cependant, il n'est pas nécessaire que le bit %Q *Recherche position initiale* reste à 1 pendant tout le cycle ; il peut être ponctuel. Notez que l'activation du bit *Recherche position initiale* désactive immédiatement le bit %I *Position valide* jusqu'à la fin du cycle de position initiale. Le bit %Q *Interruption de tous les mouvements* interrompt les cycles de position initiale, mais le bit *Position valide* ne le relance pas. Vous ne pouvez exécuter aucun programme de mouvement, excepté si le bit *Position valide* est à 1.

1.1.1. Mode ORIGIN

Si le mode de Recherche position initiale est configuré à ORIGIN, l'entrée **Commutateur de position initiale** du connecteur d'E/S donne une indication approximative de l'emplacement de référence de la position initiale. Le prochain marqueur rencontré lors du déplacement dans le sens négatif indique l'emplacement exact. Si le **Commutateur de position initiale** est ouvert, la servocommande est sur le côté positif du **Commutateur de position initiale**. Une transition de 0 à 1 de la commande *Recherche position initiale* produit le cycle de position initiale suivant. Sauf spécification contraire, l'accélération est l'*accélération de mouvement* (*Jog Acceleration*) en cours, dans le *Mode d'accélération de mouvement* configuré. S'il est lancé à partir du côté positif du **Commutateur de position initiale**, le cycle commence à l'étape 1 ; sinon, le cycle commence à l'étape 3 :

1. L'axe se déplace dans le sens négatif à la *Vitesse de recherche de position initiale* jusqu'à ce que le **Commutateur de position initiale** se ferme.
2. L'axe ralentit et s'arrête.
3. L'axe accélère ensuite dans le sens positif et se déplace à la *Vitesse de recherche de position initiale* jusqu'à ce que le **Commutateur de position initiale** s'ouvre.
4. L'axe ralentit et s'arrête.
5. L'axe accélère dans le sens négatif et se déplace à la *Vitesse de recherche de position initiale* configurée jusqu'à ce que le **Commutateur de position initiale** se ferme.
6. L'APM continue son mouvement négatif à la vitesse finale de position initiale configurée jusqu'à la détection d'une impulsion de marquage. Le marquage établit l'emplacement de référence de la position initiale.
7. L'axe ralentit et s'arrête.
8. L'axe se déplace, à la vitesse de mouvement configurée, du nombre d'unités utilisateur spécifié par la valeur du *Décalage de position initiale* en partant de l'emplacement de référence de la position initiale.
9. L'axe ralentit et s'arrête.
10. Un *Réglage position* interne règle les positions *commandée* et *réelle* à la valeur de position initiale configurée. Enfin, l'APM met le bit %I *Position valide* à 1 pour indiquer la fin du cycle de position initiale.

1.1.2. Modes Sens+ et Sens–

Si le mode de Recherche position initiale est configuré à Sens+ ou Sens–, le système utilise la première impulsion de marquage rencontrée pendant le mouvement dans le sens approprié (positif pour Sens+, négatif pour Sens–) pour établir l'emplacement exact. Une transition de 0 à 1 du bit %Q Recherche position initiale entraîne les opérations suivantes :

1. L'axe accélère au taux d'accélération de mouvement et se déplace à la *Vitesse finale de position initiale* configurée (sens positif pour Sens+, négatif pour Sens–) jusqu'à la détection d'une impulsion de marquage. Cette impulsion de marquage établit l'emplacement de référence de la position initiale.
2. L'axe s'arrête au taux d'*Accélération de mouvement*, avec le *Mode d'accélération de mouvement* configuré.
3. L'axe se déplace, à la *Vitesse de mouvement* (*Jog Velocity*) configurée, avec le taux d'*Accélération de mouvement* (*Jog Acceleration*) et le *Mode d'accélération de mouvement* configurés, du nombre d'unités utilisateur spécifié par la valeur du *Décalage de position initiale* en partant de l'emplacement de référence de la position initiale.
4. L'axe s'arrête au taux d'*Accélération de mouvement* (*Jog Acceleration*) configuré, avec le *Mode d'accélération de mouvement* configuré.
5. Un *Réglage position* interne règle les positions *commandée* et *réelle* à la valeur de position initiale configurée ; l'APM met le bit %I *Position valide* à 1 pour indiquer la fin du cycle de position initiale.

1.2. MOUVEMENT JOG AVEC L'APM

L'APM vous permet de configurer la *Vitesse de mouvement* (*Jog Velocity*), l'*Accélération de mouvement* (*Jog Acceleration*) et le *Mode d'accélération de mouvement* (*Jog*). Ces valeurs sont utilisées à chaque activation du bit %Q *Mouvement positif* (*Jog Plus*) ou *Mouvement négatif* (*Jog Minus*). Notez qu'aucun des deux bits ne génère de mouvement. Elles sont également utilisées lors des cycles de *Recherche position initiale* et lors de l'exécution d'une commande immédiate *Mouvement à vitesse constante*. Les mouvements programmés utilisent la *Vitesse de mouvement* et l'*Accélération de mouvement* comme valeurs par défaut.

Vous pouvez exécuter un mouvement (*Jog*) lorsqu'aucun autre mouvement n'est commandé, ou lorsqu'un mouvement programmé est en pause. L'état du bit %Q *Activation circuit de commande* n'a pas d'influence sur le mouvement (Jog). L'activation d'un bit %Q *Mouvement* (Jog) ferme automatiquement le relais d'activation et active le bit %I *Circuit de commande activé*. Lorsqu'un **Interrupteur de fin de course de dépassement** est désactivé, vous pouvez utiliser *Mouvement* (Jog) et *Effacement erreur* simultanément pour aller vers une position correcte. *Mouvement positif* ne fonctionne pas quand le **Commutateur de fin de course positive** est ouvert ; *Mouvement négatif* ne fonctionne pas quand le **Commutateur de fin de course négative** est ouvert.

1.3. COMMANDE MOUVEMENT À VITESSE CONSTANTE

Pour générer une commande *Mouvement à vitesse constante*, vous devez placer la valeur 22h dans le premier mot des données %AQ affectées à un axe alors que le bit %I *Circuit de commande activé* est à 1. Les deuxième et troisième mots représentent une vitesse sur 32 bits, signée, à laquelle l'APM essaiera de déplacer l'axe. Notez que le troisième mot est le mot de poids fort de la vitesse. Après avoir transmis la commande, vous pouvez effacer les données %AQ en envoyant une commande NULL ou les modifier en fonction de vos besoins.

La liste des commandes immédiates %AQ présente les mots dans l'ordre inverse pour faciliter la compréhension. Par exemple, pour commander une vitesse de 512 unités utilisateur par seconde dans un APM configuré avec des données %AQ commençant à %AQ1, vous devez utiliser les valeurs suivantes : 0022h (34 en décimal) dans %AQ1, 0200h (512 en décimal) dans %AQ2 et 0 dans %AQ3. Lorsque l'APM reçoit ces valeurs, si le bit %I *Circuit de commande activé* est à 1, si le bit %Q *Interruption de tous les mouvements* est à 0 et si aucun autre mouvement n'est commandé, l'axe doit se déplacer à 512 unités utilisateur par seconde dans le sens positif.

Si le bit %I *Circuit de commande activé* n'est pas à 1 lorsque l'APM reçoit la commande immédiate, une erreur se produit. De même, si une commande *Mouvement à vitesse constante* est déjà présente dans les données %AQ, la valeur de la vitesse doit être modifiée lorsque le bit *Circuit de commande activé* est à 1 pour que l'APM accepte la modification. L'APM détecte une commande *Mouvement à vitesse constante*, comme toutes les commandes immédiates %AQ, lorsque les valeurs %AQ changent.

Lorsque l'APM exécute un *Mouvement à vitesse constante*, il ignore les *Limites de fin de course logicielles*. Les limites de dépassement matérielles doivent être à 1 si elles sont validées. La limitation du mouvement de l'APM pendant un mouvement à vitesse constante dépend de l'opérateur, pas de l'APM.

Il existe deux façons d'interrompre un *Mouvement à vitesse constante* sans entraîner d'erreur : par une commande *Mouvement à vitesse constante* avec une vitesse nulle ou par l'activation pendant au moins un cycle d'API du bit %Q *Interruption de tous les mouvements*.

1.4. COMMANDE FORÇAGE N/A

Pour obtenir une tension de sortie spécifique de l'APM vers la servocommande, utilisez la commande immédiate *Forçage N/A*. Elle fait correspondre des valeurs comprises entre –32000 et +32000 avec des tensions comprises entre –10 V et +10 V. Une valeur de +3200 correspond ainsi à 1 V. Le bit %I *Circuit de commande activé* doit être à 1 pour que *Forçage N/A* fonctionne. Le deuxième axe d'un APM mono–axe fait exception à cette règle. Du fait qu'un APM mono–axe ne comporte pas de bit *Activation circuit de commande* pour le deuxième axe, il suffit de placer les valeurs dans les données %AQ et le deuxième axe est automatiquement activé. La désactivation du bit %Q *Activation circuit de commande* est une manière d'interrompre une commande *Forçage N/A*.

La commande *Forçage N/A* est la seule commande qui doive être maintenue dans les données %AQ pour fonctionner. Autrement dit, les données %AQ doivent continuer à présenter la commande *Forçage N/A* et les données à l'APM. Si une autre commande immédiate est transmise à l'APM, *Forçage N/A* s'arrête. Les commandes ponctuelles fonctionnent uniquement pendant le cycle où elles apparaissent.

1.5. COMMANDE INCRÉMENT DE POSITION

Pour générer de petites corrections entre la servocommande réelle et le suivi de l'APM, vous pouvez utiliser la commande *Incrément de position* de façon à injecter un nombre spécifique d'unités utilisateur dans l'*Erreur de position*. Le résultat de la mise à 1 du bit %I *Circuit de commande activé*, particulièrement visible sur un système stationnaire, est le fait que l'APM commande à la servocommande de se déplacer de façon à compenser l'incrément commandé. Si le bit %I *Circuit de commande activé* est à zéro, la valeur de l'*Incrément de position* est ajoutée à la fois à la *position commandée* et à la *position réelle*. L'utilisation d'*Incrément de position* ne devrait pas être nécessaire après la configuration d'un système. Il ne s'agit pas d'un mouvement continu, mais uniquement d'un mouvement ponctuel. Vous pouvez utiliser l'*Incrément de position* à tout moment, excepté simultanément avec la commande *Forçage N/A*. En effet, cette commande doit apparaître de façon continue dans les données %AQ.

1.6. AUTRES REMARQUES

Autres remarques relatives à l'utilisation d'un mouvement non programmé :

- Si le bit *Interruption de tous les mouvements* est à 1, aucun mouvement ne peut commencer.
- La mise à 1 du bit *Interruption de tous les mouvements* interrompt immédiatement tous les mouvements non programmés en cours à leur *Accélération de mouvement* (*Jog Acceleration*) courante.
- L'exécution de la commande *Réglage position* pendant un mouvement non programmé entraîne une erreur d'état.
- La mise à 0 du bit *Activation circuit de commande* pendant l'exécution d'un cycle de position initiale ou d'un *Mouvement à vitesse constante* entraîne une erreur d'arrêt.
- Le bit *Pause* n'a aucun effet sur les mouvements non programmés.
- La commande *Forçage taux* n'a aucun effet sur les mouvements non programmés.
- La modification de la *Vitesse de mouvement* (*Jog Velocity*) ou de l'*Accélération de mouvement* (*Jog Acceleration*) n'a aucun effet sur les mouvement non programmés en cours.

2. MOUVEMENT PROGRAMMÉ

L'APM exécute les commandes de mouvement du programme de façon séquentielle, bloc après bloc, dès qu'un programme a été sélectionné. Les commandes du programme peuvent être classées de la façon suivante :

Commandes de type 1

- Appel de sous–programme (CALL)
- Saut (JUMP)

Commandes de type 2

- N° de bloc (BLOCK)
- Aucune action (NULL)
- Accélération (ACCEL)
- Vitesse (VELOC)

Commandes de type 3

- Mouvement de positionnement (PMOVE)
- Mouvement continu (CMOVE)
- Arrêt momentané (DWELL)
- Attente (WAIT)
- Fin de programme (END)

Les commandes de type 1 peuvent rediriger le chemin d'exécution du programme, sans affecter directement la position. CALL exécute un sous–programme avant de rendre l'exécution à la commande suivante. JUMP continue l'exécution à un autre emplacement ou teste des bits CTL et, en fonction de la condition du bit, modifie ou non le chemin du programme.

Les commandes de type 2 n'affectent pas non plus la position. La commande BLOCK fournit une identification ou étiquette pour la commande de type 3 suivante. Si le système ne trouve aucun numéro de bloc dans le bloc courant, il utilise le numéro de bloc précédent. Les commandes de vitesse et d'accélération spécifient la vitesse et l'accélération du mouvement.

Les commandes de type 3 lancent ou interrompent un mouvement, affectant ainsi le contrôle de la position. Les mouvements de positionnement et continus commandent le mouvement ; les commandes d'arrêt momentané, d'attente et de fin de programme interrompent le mouvement.

Un bloc est constitué d'une seule commande de type 3 précédée d'un nombre et d'une combinaison quelconque de commandes de types 1 et 2. Les commandes de type 2 sont optionnelles ; un bloc peut ne contenir qu'une commande de type 3. Les commandes de type 2, et les sauts conditionnels, ne prennent pas effet tant que la commande de type 3 suivante n'est pas exécutée.

Lorsque l'APM exécute un bloc, le bloc de programme suivant est traité dans une zone de commande tampon afin de minimiser le délai entre deux blocs. Les paramètres utilisés dans un mouvement doivent donc être chargés avant la fin de l'exécution du mouvement commandé deux blocs avant.

Lorsqu'un APM à deux axes exécute un programme à deux axes, chaque axe scrute indépendamment de l'autre les commandes du programme et exécute uniquement les commandes qui lui sont destinées. Remarquez que certaines commandes ne spécifient aucun axe (BLOCK, JUMP, CALL et END) et s'appliquent donc aux deux axes.

Un programme à deux axes peut contenir des commandes de numéros de bloc de synchronisation permettant de synchroniser deux mouvements à des points spécifiés. Lorsque le premier axe atteint le bloc de synchronisation, il attend pour l'exécuter que l'autre axe ait également atteint le bloc de synchronisation. Reportez–vous à l'exemple 18 pour une illustration de ce type d'opération.

Vous trouverez ci–dessous une description de nombreux aspects des mouvements programmés.

2.1. PRÉREQUIS D'UN PROGRAMME DE MOUVEMENT

Pour qu'un programme de mouvement puisse être exécuté, les conditions suivantes doivent être satisfaites (pour un programme multi–axe, les deux axes, 1 et 2, doivent respecter ces conditions) :

- Le bit *Circuit de commande activé* doit être à 1
- Le bit *Position valide* doit être à 1
- Le bit *En mouvement* doit être à 0
- Le bit *Programme Actif* doit être à 0
- Le bit *Interruption de tous les mouvements* doit être à 0
- L'axe doit se trouver entre les limites de fin de course configurées
- Les interrupteurs de dépassement de fin de course doivent être activés s'ils sont validés
- La commande Forçage N/A ne doit pas être active
- Le programme à exécuter doit être valide et enregistré dans l'APM
- Les paramètres des deux premiers blocs doivent déjà se trouver dans l'APM

2.2. CONDITIONS QUI INTERROMPENT UN PROGRAMME DE MOUVEMENT

Les conditions suivantes interrompent immédiatement les programmes de mouvement :

- Le bit *Interruption de tous les mouvements* passe à 1
- Le bit *Circuit de commande activé* passe à 0
- Un **Interrupteur de dépassement de fin de course** passe à 0 alors qu'il est validé
- Le mouvement programmé suivant, PMOVE ou CMOVE, dépasse une *Limite de fin de course logicielle*
- Une erreur d'arrêt se produit

2.3. PARAMÈTRES DES MOUVEMENTS PROGRAMMÉS

Les mouvements programmés ont trois paramètres :

1. La *distance* de déplacement ou la *position* à atteindre,
2. Le *type de référence de positionnement* à utiliser pour le mouvement
3. Le *type de mouvement* à utiliser pendant l'exécution du mouvement.

2.3.1. Types de références de positionnement

Pour le type de référence à utiliser pour le mouvement, les choix possibles sont ABS (absolu) et INC (incrémentiel). Cette référence détermine comment sera interprété le premier paramètre, c'est–à–dire la distance de déplacement ou la position à atteindre.

2.3.1.1. Positionnement absolu

Dans un mouvement à positionnement absolu, le premier paramètre est la position à atteindre. Exemple de mouvement à positionnement absolu :

```
PMOVE 5000, ABS, LINEAR
```

Ce mouvement déplace l'axe à partir de sa position actuelle, quelle qu'elle soit, jusqu'à la position 5000. La distance réellement parcourue dépend donc de la position de l'axe au début du mouvement. Si sa position initiale est 0, l'axe se déplace de 5000 unités utilisateur dans le sens positif. Si la position initiale est 10000, l'axe se déplace de 5000 unités utilisateur dans le sens négatif. Si la position initiale est 5000, l'APM ne génère aucun mouvement.

2.3.1.2. Positionnement incrémentiel

Dans un déplacement incrémentiel, le premier paramètre est interprété en tant que distance à parcourir à partir de la position de l'axe au début du mouvement. L'APM traduit les distances des mouvements incrémentiels en positions de mouvement absolu pour empêcher l'accumulation des erreurs. Exemple de mouvement à positionnement incrémentiel :

```
PMOVE 5000, INC, LINEAR
```

Ce mouvement incrémentiel déplace l'axe de sa position actuelle jusqu'à une position supérieure de 5000 unités utilisateur. Avec un mouvement incrémentiel, le premier paramètre spécifie le nombre d'unités utilisateur que parcourt l'axe.

2.3.2. Types de mouvement

Les choix possibles pour le dernier paramètre, qui spécifie le type de mouvement à utiliser pendant l'exécution du mouvement, sont linéaire (LINEAR) et courbe en S (SCURVE).

2.3.2.1. Mouvement linéaire

Vous trouverez ci–après un exemple de profil de mouvement linéaire où la vitesse est présentée en fonction du temps. Les lignes droites du graphique indiquent qu'un mouvement linéaire utilise des accélérations constantes. L'aire située sous le graphique indique la distance parcourue.

```
ACCEL     1000
VELOC     2000
PMOVE     6000, INC, LINEAR
```

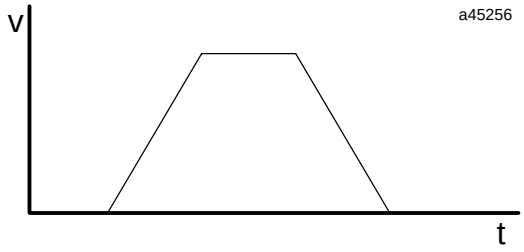


Figure 6-1. Exemple de mouvement linéaire

2.3.2.2. Mouvement en courbe en S

Vous trouverez ci–dessous un exemple de mouvement en courbe en S, où la vitesse est également présentée en fonction du temps. Les lignes courbes de ce graphique indiquent que l'accélération n'est pas constante. Au début du déplacement, l'accélération est très faible et augmente jusqu'à atteindre l'accélération programmée. Il s'agit du point central de l'accélération. Puis l'accélération décroît jusqu'à zéro ; l'axe a atteint la vitesse programmée. Un mouvement en courbe en S nécessite, pour une même accélération, deux fois plus de temps, et donc de distance, pour accélérer et décélérer qu'un mouvement linéaire. L'aire située sous le graphique indique également la distance parcourue.

```
ACCEL     2000
VELOC     2000
PMOVE     8000, INC, SCURVE
```

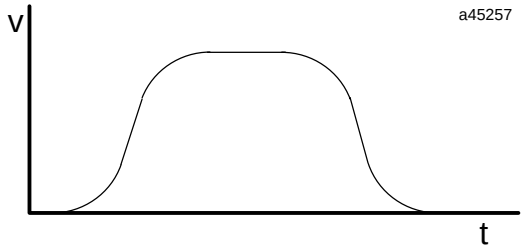


Figure 6-2. Exemple de mouvement en courbe en S

2.4. TYPES DE COMMANDES DE MOUVEMENT PROGRAMMÉ

2.4.1. Mouvement de positionnement (PMOVE)

Un mouvement PMOVE utilise la vitesse et l'accélération les plus récemment programmées. Si aucune commande VELOC n'a été rencontrée dans le programme de mouvement, l'APM utilise la *Vitesse de mouvement* (*Jog Velocity*) comme valeur par défaut. Si aucune commande ACCEL n'a été rencontrée dans le programme de mouvement, l'APM utilise l'*Accélération de mouvement* (*Jog Acceleration*) comme valeur par défaut.

Le mouvement PMOVE arrête toujours l'axe lorsqu'il est terminé pour permettre la mise à 1 du bit %I *EN ZONE*.

2.4.2. Mouvement continu (CMOVE)

Un mouvement CMOVE utilise la vitesse et l'accélération les plus récemment programmées. Si aucune commande VELOC n'a été rencontrée dans le programme de mouvement, l'APM utilise la *Vitesse de mouvement* (*Jog Velocity*) comme valeur par défaut. Si aucune commande ACCEL n'a été rencontrée dans le programme de mouvement, l'APM utilise l'*Accélération de mouvement* (*Jog Acceleration*) comme valeur par défaut.

Le mouvement CMOVE ne s'arrête pas après la fin du mouvement, excepté s'il est suivi d'une commande DWELL ou WAIT, si la vitesse programmée suivante est nulle ou s'il constitue la dernière commande du programme. Il n'attend pas que la position soit EN ZONE avant de passer au mouvement suivant. En fait, le mouvement CMOVE tente de se terminer avec la vitesse spécifiée pour la commande de mouvement suivante.

2.5. MOUVEMENTS PROGRAMMÉS

En combinant les mouvements CMOVE et PMOVE, les mouvements absolus et incrémentiels, et les mouvements linéaire et courbes en S, vous pouvez générer pratiquement n'importe quel profil de mouvement. Les exemples suivants présentent des profils de mouvement simples, ainsi que quelques cas de programmation incorrecte du mouvement.

Exemple 01 : Combinaison de mouvements PMOVE et CMOVE

Cet exemple présente une combinaison simple de mouvements PMOVE et CMOVE formant des profils de mouvement.

```
ACCEL     1000
VELOC     2000
PMOVE     5000, ABS, LINEAR
VELOC     1200
PMOVE     10000, ABS, SCURVE
ACCEL     1500
VELOC     2800
CMOVE     6000, INC, LINEAR
VELOC     1200
CMOVE     23000, ABS, SCURVE
ACCEL     1000
VELOC     2800
PMOVE     5000, INC, LINEAR
```

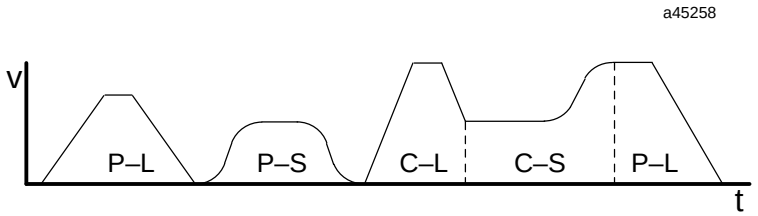


Figure 6-3. Combinaison de mouvements PMOVE et CMOVE

Les types de mouvement sont indiqués sous les mouvements correspondants ; par exemple, P–L indique un mouvement PMOVE linéaire.

Le premier PMOVE accélère jusqu'à la vitesse programmée, se déplace d'une certaine distance et décélère jusqu'à s'arrêter. Tous les mouvements s'arrêtent après un PMOVE. Lorsque le premier mouvement s'arrête, il est à la distance programmée.

Le deuxième mouvement est un mouvement PMOVE en courbe en S. Comme le premier, il accélère jusqu'à la vitesse programmée, se déplace pendant un certain temps, et décélère jusqu'à la vitesse nulle (il s'agit d'un PMOVE).

Le mouvement suivant est un mouvement CMOVE linéaire. Il accélère jusqu'à la vitesse programmée, se déplace pendant un certain temps, et décélère ensuite jusqu'à une vitesse inférieure en utilisant une accélération linéaire. Lorsqu'un mouvement CMOVE se termine, il est à la position programmée pour le mouvement qui vient de se terminer et à la vitesse du mouvement suivant. Donc, lorsque le quatrième mouvement commence, il est déjà à la vitesse programmée.

Le quatrième mouvement est un mouvement CMOVE. Donc, lorsqu'il approche de sa position finale, il accélère pour se terminer à la vitesse du cinquième mouvement. Le graphique indique que l'accélération du quatrième mouvement est en courbe en S.

Finalement, le cinquième mouvement commence et utilise la vitesse programmée un certain temps avant de décélérer jusqu'à la vitesse nulle. Tout mouvement consécutif au cinquième mouvement doit commencer à la vitesse nulle puisqu'il s'agit d'un mouvement PMOVE.

Exemple 02 : Modification du mode d'accélération pendant un profil

L'exemple suivant explique comment utiliser une accélération différente, et un mode d'accélération différent, dans un profil utilisant des mouvements CMOVE. Le premier CMOVE accélère de façon linéaire jusqu'à la vitesse programmée. La vitesse du deuxième CMOVE étant identique à celle du premier, le premier mouvement CMOVE se termine sans modification de vitesse. L'accélération du deuxième mouvement est en courbe en S pour décélérer jusqu'à la vitesse nulle.

```
ACCEL     2000
VELOC     6000
CMOVE     13000, ABS, LINEAR
ACCEL     4000
CMOVE     15000, INC, SCURVE
```

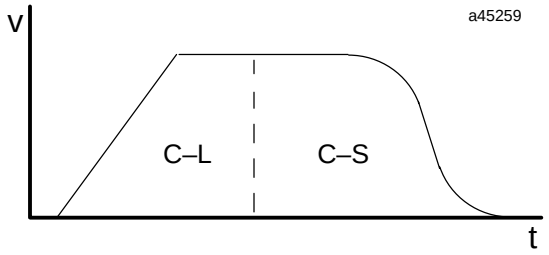


Figure 6-4. Modification du mode d'accélération pendant un profil

Exemple 03 : Distance insuffisante pour atteindre la vitesse programmée

Vous pouvez programmer des mouvements CMOVE et PMOVE avec des distances insuffisantes pour atteindre les vitesses programmées. Le graphique suivant présente un mouvement CMOVE qui n'atteint pas la vitesse programmée. Sachant cela, l'APM commence la décélération pour permettre à l'axe de se trouver à la vitesse du mouvement suivant. Il s'agit généralement d'une situation involontaire due à une erreur de programmation.

```
ACCEL     2000
VELOC     8000
CMOVE     7000, INC, LINEAR
ACCEL     10000
VELOC     2000
CMOVE     4400, INC, LINEAR
```

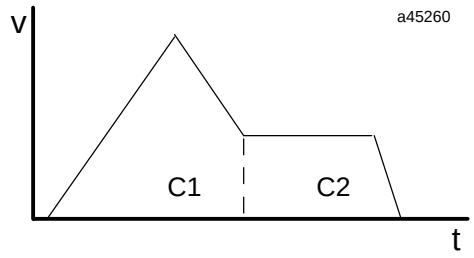


Figure 6-5. Distance insuffisante pour atteindre la vitesse programmée

Exemple 04 : Arrêt brutal de l'APM (distance insuffisante/vitesse)

L'arrêt brutal de l'APM à partir d'une vitesse élevée lorsque la distance est insuffisante est une erreur de programmation relativement grave. Dans l'exemple suivant, le premier CMOVE accélère jusqu'à une vitesse élevée. Le deuxième CMOVE a une vitesse identique. Cependant, la distance spécifiée pour ce CMOVE est très courte. L'axe se déplace donc à une vitesse élevée et doit s'arrêter sur une distance très courte. Si la décélération (ACCEL) programmée n'est pas suffisante, le profil suivant risque de se produire : pour ne pas dépasser la position finale, l'APM commande instantanément une vitesse nulle. La secousse provoquée est très dangereuse et peut endommager le système. L'axe risque également de dépasser la position programmée et de provoquer des dégâts.

```
ACCEL     500
VELOC     3000
CMOVE     9000, ABS, LINEAR
ACCEL     600
CMOVE     4800, INC, LINEAR
```

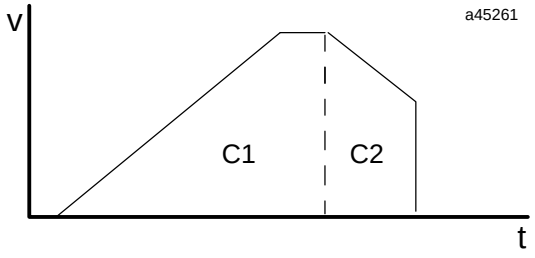


Figure 6-6. Arrêt brutal de l'APM (distance insuffisante/vitesse)

2.6. COMMANDE DWELL

La commande DWELL permet de ne générer aucun mouvement sur un axe pendant une période spécifiée. Placée après un mouvement CMOVE, la commande DWELL fait fonctionner le CMOVE à la manière d'un PMOVE, même si la durée spécifiée pour l'arrêt momentané est nulle.

Exemple 05 : DWELL

Un profil de mouvement simple (déplacement jusqu'à un point spécifique, attente et retour au point de départ) pourrait utiliser le programme et le profil de vitesse suivants.

```
ACCEL    30000
VELOC    15000
PMOVE    120000, ABS, LINEAR
DWELL    4000
PMOVE    0, ABS, LINEAR
```

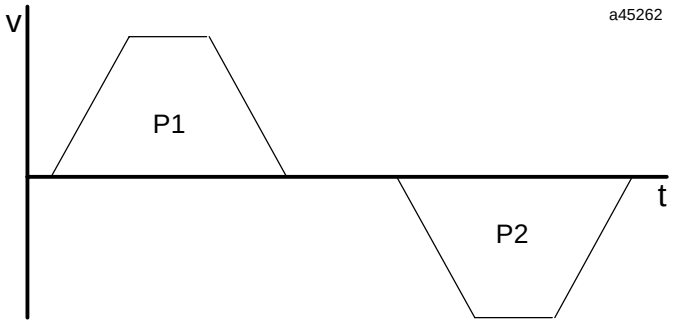


Figure 6-7. DWELL

2.7. COMMANDE WAIT

La commande WAIT est identique à la commande DWELL mais, au lieu de ne générer aucun mouvement pendant une période de temps spécifiée, elle interrompt le mouvement du programme et surveille un bit CTL jusqu'à ce qu'il s'active. Le mouvement s'interrompt donc chaque fois qu'une commande WAIT est rencontrée, même si le bit CTL est activé avant la rencontre du WAIT. Chacun des douze bits CTL peut servir à reprendre le programme.

Si, dans l'exemple précédent, nous remplaçons le DWELL par un WAIT, le profil de mouvement reste identique, excepté le fait que le deuxième mouvement PMOVE ne commence pas avant l'activation du bit CTL. Si le bit CTL est déjà activé lorsque le programme atteint le WAIT, le deuxième PMOVE commence immédiatement après la fin du premier PMOVE.

De plus, les mouvements CMOVE et PMOVE génèrent le même profil de vitesse. La commande WAIT interrompt le mouvement, que la commande précédente soit un CMOVE ou un PMOVE.

2.8. SOUS–PROGRAMMES

L'APM peut stocker jusqu'à dix programmes et quarante sous–programmes distincts. Les sous–programmes sont indépendants des axes. Autrement dit, tout programme utilisant un axe peut appeler n'importe quel sous–programme. Sur les APM à deux axes, les deux axes peuvent appeler le même sous–programme simultanément. La commande utilisée pour exécuter un sous–programme est la commande CALL, le numéro du sous–programme étant spécifié comme argument. L'exécution du programme continue au début du sous–programme. Elle reprend après la fin du sous–programme, avec la commande qui suit le CALL. Vous pouvez appeler un sous–programme à partir de n'importe quel autre sous–programme mais, pour appeler de nouveau un certain sous–programme sur le même axe, vous devez attendre qu'il soit terminé. La récursivité n'est pas autorisée.

2.9. NUMÉROS DE BLOCS ET SAUTS

Les numéros de blocs sont utilisés comme points de référence dans les programmes de mouvement. Ils servent également à contrôler les tests de saut. Un mot de données %AI affiche le numéro de bloc courant. Vous pouvez le surveiller pour garantir une exécution correcte du programme ou pour déterminer quand des événements doivent se produire. Les numéros de blocs peuvent également servir de destination pour les commandes JUMP.

2.10. SAUTS INCONDITIONNELS

Il existe deux types de sauts : les sauts conditionnels et les sauts inconditionnels. Une commande de saut inconditionnelle indique simplement à l'APM de continuer l'exécution du programme à partir du numéro de bloc destination. Exemple de saut inconditionnel :

Exemple 06 : Saut inconditionnel

Le programme exécute un PMOVE, fait une pause pendant 2 secondes, puis saute de façon inconditionnelle vers le bloc 1 au début du programme. La commande PMOVE sera donc répétée jusqu'à ce qu'une *Limite de fin de course* ou un **Interrupteur de dépassement de fin de course** soit atteint.

```
ACCEL     10000
VELOC     30000
BLOCK     1
PMOVE     200000
DWELL     2000
JUMP      UNCOND, 1
```

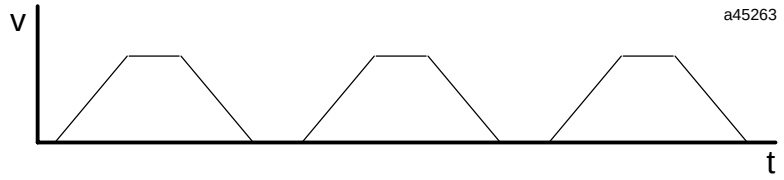


Figure 6-8. Saut inconditionnel

2.11. SAUTS CONDITIONNELS

Un saut conditionnel est une commande JUMP pour laquelle vous spécifiez un bit CTL. Les sauts conditionnels sont des commandes de type 1 dans la mesure où ils affectent le chemin d'exécution du programme. Ce sont également des commandes de type 2 puisqu'elles ne prennent effet que lors de l'exécution d'une commande de type 3 suivant la commande JUMP. Lorsqu'une commande de saut conditionnel est exécutée, l'APM examine le bit CTL spécifié. S'il est à 1, l'exécution du programme continue au numéro de bloc destination. Si le bit est à 0, le programme se poursuit par l'exécution de la commande suivant le JUMP. Notez que la commande de type 3 qui suit le saut conditionnel et la destination de saut affectent le comportement du saut.

2.11.1. Tests de saut

Les sauts conditionnels réalisent un test de saut. Si le bit CTL est à 1, le saut est exécuté immédiatement. Si le bit CTL est à 0, l'APM surveille le bit CTL et conserve la destination de saut en mémoire. Cette surveillance du bit CTL est appelée test de saut. Si le bit CTL passe à 1 au cours du test avant qu'une commande BLOCK, une autre commande JUMP, ou une commande CALL ne soit rencontrée, le saut est exécuté. Ces commandes (BLOCK, JUMP et CALL) mettent fin au test de saut.

Exemple 07 : Test de saut

Etudions les deux segments de programme suivants. Dans le programme de gauche, le mouvement vers la position 2000 se termine avant que le test de saut ne commence. La commande BLOCK qui suit immédiatement la commande de saut met fin au test de saut. Le temps pendant lequel l'APM surveille le bit CTL est donc très court. Dans le programme de droite, la commande JUMP est rencontrée avant la commande de mouvement. Le test de saut commence avant le mouvement et continue tant que le mouvement n'est pas terminé. Si le bit CTL passe à 1 pendant le mouvement, le saut est exécuté. Après le saut, la commande BLOCK met fin au test de saut et l'exécution du programme continue normalement. Le test de saut aurait continué pendant les mouvements suivants éventuellement rencontrés avant la commande BLOCK.

```
ACCEL   5000              ACCEL   5000
VELOC   1000              VELOC   1000
BLOCK   1                 BLOCK   1
CMOVE   2000, ABS, LINEAR JUMP    CTL01, 3
JUMP    CTL01, 3          CMOVE   2000,    ABS, LINEAR
BLOCK   2                 BLOCK   2
```

2.11.2. Interruption normale avant un JUMP

Les commandes de saut conditionnel sont identiques aux commandes de type 2 dans la mesure où elles ne sont pas exécutées avant l'exécution de la commande de type 3 située immédiatement après. Si cette commande de type 3 est une commande qui normalement arrête le mouvement, l'exécution du JUMP interrompt le mouvement. Les commandes de type 3 qui arrêtent le mouvement sont les commandes de pause, d'attente, de fin de programme ainsi que les mouvements dans le sens opposé.

Bien que le bit CTL puisse être à 1 avant l'exécution du bloc contenant le saut conditionnel et la commande de type 3, le mouvement de l'axe s'interrompt avant que l'exécution ne continue à la destination de saut. Cette interruption n'est pas un arrêt après saut, décrit plus loin.

Exemple 08 : Interruption normale avant un JUMP

L'exemple suivant contient un saut suivi d'une commande DWELL. L'APM, qui fait un traitement anticipé, sait qu'il doit s'arrêter après la commande CMOVE. Il s'arrête donc avant que le DWELL ne soit exécuté. Le test de saut ne commençant pas avant l'exécution du DWELL, il commence uniquement après l'arrêt du mouvement. Il se termine lorsque le CMOVE suivant commence, à cause de la commande BLOCK qui lui est associée. Les lignes en pointillés du profil de vitesse indiquent le début et la fin du test de saut. Le bit CTL03 ne passe pas à 1 pendant l'exécution du programme.

```
BLOCK   1
ACCEL   5000
VELOC   10000
CMOVE   60000, INC, LINEAR
BLOCK   2
JUMP    CTL03, 4
DWELL   4000
BLOCK   3
ACCEL   10000
VELOC   5000
CMOVE   15000, INC, LINEAR
BLOCK   4
NULL
```

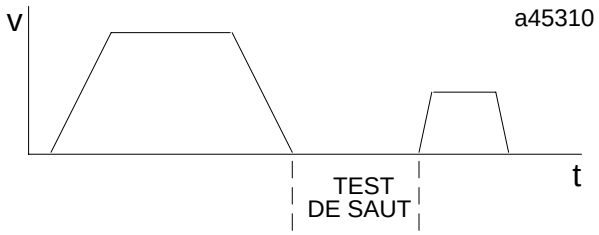


Figure 6-9. Interruption normale avant un JUMP

2.11.3. Saut sans interruption

Si la commande de type 3 qui suit un saut conditionnel est un CMOVE et si la commande de type 3 de la destination est une commande de mouvement avec une distance suffisante pour ralentir jusqu'à la vitesse nulle à la fin du mouvement, le saut est exécuté sans interruption. C'est le seul moyen de maintenir le mouvement lorsqu'un saut est exécuté.

Exemple 09 : JUMP sans interruption

Vous trouverez ci–dessous un exemple simple de saut conditionnel d'une commande CMOVE à une autre. Pendant le test de saut du bit CTL03, le premier CMOVE accélère jusqu'à la vitesse programmée. Avant la ligne pointillée, le bit CTL03 est à 0. Il est à 1 ensuite. L'exécution du programme est immédiatement transférée au bloc 3 et le deuxième CMOVE commence. La vitesse définie dans le bloc de destination étant différente, l'APM modifie cette vitesse avec l'accélération programmée dans le bloc de destination. Finalement, à la fin du deuxième mouvement, la vitesse est réduite à zéro et le programme se termine.

```
BLOCK    1
ACCEL    2000
VELOC    10000
JUMP     CTL03, 3
CMOVE    120000, INC, LINEAR
BLOCK    3
ACCEL    20000
VELOC    5000
CMOVE    15000, INC, LINEAR
```

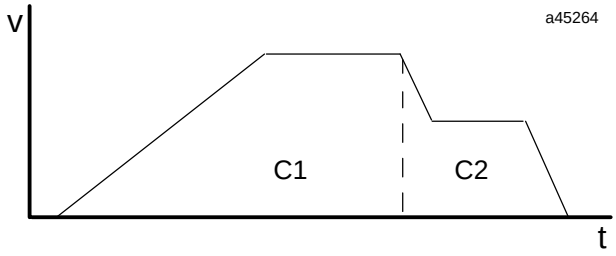


Figure 6-10. JUMP sans interruption

2.11.4. Arrêt après saut

Un arrêt après saut est un arrêt causé par un saut. Lorsqu'un arrêt après saut se produit, l'APM utilise l'*Accélération de mouvement* (*Jog Acceleration*) et le *Mode d'accélération de mouvement* (*Jog*) à la place de toute accélération programmée. Notez que les mouvements en courbe en S atteignent une vitesse constante avant d'utiliser l'*Accélération de mouvement* (*Jog Acceleration*) et de commencer à décélérer (voir exemples de sauts en courbe en S pour plus de détails). L'*Accélération de mouvement* (*Jog Acceleration*) est utilisée parce qu'un arrêt après saut pourrait indiquer un problème. L'*Accélération de mouvement* (*Jog Acceleration*) courante, que vous pouvez modifier avec une commande immédiate, fournit plus de souplesse que l'accélération programmée. Il existe deux façons de générer un arrêt après saut, décrites ci–dessous.

Une commande JUMP suivie d'un PMOVE génère un arrêt après saut si le saut est réalisé. Le PMOVE arrêtant toujours l'axe avant de permettre le début du mouvement suivant, tout saut exécuté pendant un PMOVE génère un arrêt après saut.

Lorsqu'un saut conditionnel se produit, un arrêt ou un changement de direction du mouvement en cours génère un arrêt après saut. Cette condition est appelée arrêt après saut et l'accélération programmée est ignorée.

Cependant, lorsqu'un saut se produit pendant un CMOVE, aucun arrêt après saut ne se produit si le mouvement programmé à la destination de saut est un PMOVE ou CMOVE dans le même sens.

Pendant un mouvement en courbe en S, un arrêt après saut a l'un des effets suivants : si le saut se produit après le point central de l'accélération ou de la décélération, l'accélération/décélération se termine avant le début de l'arrêt après saut. Si le saut se produit avant le point central de l'accélération/décélération, le profil commence immédiatement à se stabiliser. Lorsque l'accélération/décélération est nulle, l'arrêt après saut commence (voir exemples de sauts en courbe en S).

Exemple 10 : Arrêt après saut

L'exemple suivant présente un saut conditionnel avec un arrêt après saut. Une amélioration de l'exemple 5, DWELL, consisterait à surveiller un bit CTL externe qui pourrait indiquer un problème avec le mouvement positif. Si le bit CTL ne passe pas à 1, le profil du programme suivant sera identique au profil présenté dans l'exemple DWELL. Si le bit CTL passe à 1 pendant le premier PMOVE ou DWELL, le mouvement inverse commence immédiatement.

Le profil suivant apparaît si le bit CTL passe à 1 pendant le premier PMOVE, au niveau de la ligne pointillée, et si l'*Accélération de mouvement* (*Jog Acceleration*) est de 75000. Le premier mouvement se terminant rapidement grâce au bit CTL et à une accélération plus élevée (l'*Accélération de mouvement* (*Jog Acceleration*) par rapport à l'accélération programmée), le deuxième mouvement doit parcourir une distance inférieure pour revenir à la position 0 par rapport à l'exemple DWELL. Notez que, comme le mouvement programmé à la destination est dans le sens opposé à celui du mouvement initial, le profil serait identique si les mouvements étaient des CMOVE au lieu de PMOVE.

```
ACCEL    30000
VELOC    15000
BLOCK    1
JUMP     CTL09, 2
PMOVE    120000, ABS, LINEAR
DWELL    4000
BLOCK    2
PMOVE    0, ABS, LINEAR
```

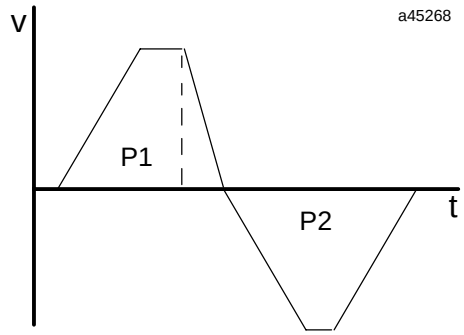


Figure 6-11. Arrêt après saut

Exemple 11 : JUMP suivi de PMOVE

Dans cet exemple, la commande suivant le JUMP est un PMOVE dans le même sens. Le profil de vitesse ci–dessous présente l'accélération et le mouvement du premier CMOVE ainsi que la décélération jusqu'à la vitesse du PMOVE. Le bit CTL01, à 0 lorsque le PMOVE commence, passe à 1 au niveau de la ligne pointillée. Le mouvement s'arrête après un PMOVE, même si un saut conditionnel fait passer le chemin d'exécution du programme à un autre bloc. Le bit CTL01 déclenche dont une décélération jusqu'à zéro avant le début du CMOVE final.

```
BLOCK   1
ACCEL   2000
VELOC   8000
CMOVE   76000, INC, LINEAR
BLOCK   2
ACCEL   1000
VELOC   4000
JUMP    CTL01, 3
PMOVE   50000, INC, LINEAR
BLOCK   3
ACCEL   6000
VELOC   6000
CMOVE   36000, INC, LINEAR
```

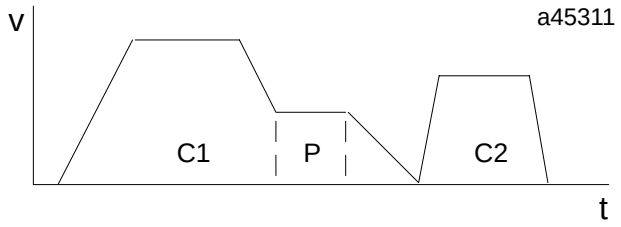


Figure 6-12. JUMP suivi de PMOVE

2.12. SAUTS EN COURBE EN S

Les sauts qui se produisent pendant des mouvements linéaires et des mouvements en courbe en S à des vitesses constantes déclenchent immédiatement l'accélération ou la décélération vers une nouvelle vitesse. Cependant, les sauts qui se produisent pendant une accélération ou une décélération en courbe en S nécessitent des règles différentes afin de conserver un profil de courbe en S. L'effet d'un saut se produisant pendant un mouvement en courbe en S et modifiant la vitesse dépend de la position du saut par rapport au point central, point où l'ampleur est la plus élevée, de l'accélération ou de la décélération, et de la différence entre la vitesse de la destination de saut et la vitesse en cours.

Si le saut se produit après le point central de modification de vitesse, la modification continue normalement jusqu'à ce qu'une vitesse constante soit atteinte ; la vitesse sera ensuite modifiée jusqu'à la nouvelle vitesse en utilisant le mode d'accélération du mouvement de la destination de saut.

Exemple 12 : SCURVE – Saut après le point central de l'accélération ou de la décélération

Dans l'exemple suivant, un saut se produit pendant la phase finale de décélération, au niveau de la ligne pointillée. La décélération continue jusqu'à ce que la vitesse constante soit atteinte, puis l'accélération vers la vitesse supérieure commence.

```
ACCEL     50000
VELOC     100000
BLOCK     1
JUMP      CTL01, 3
CMOVE     500000, ABS, SCURVE
BLOCK     2
VELOC     60000
CMOVE     –500000, INC, SCURVE
BLOCK     3
VELOC     85000
ACCEL     100000
CMOVE     250000, INC, SCURVE
```

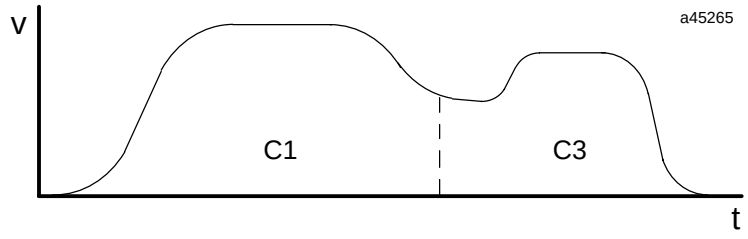


Figure 6-13. Saut après le point central de l'accélération ou de la décélération

Si un saut se produit avant le point central d'accélération ou de décélération, le résultat dépend de la différence entre la vitesse de la destination de saut et la vitesse avant le saut. Dans le cas d'une accélération avec une nouvelle vitesse inférieure à la précédente, ou dans le cas d'une décélération avec une nouvelle vitesse supérieure à la précédente, l'APM commence immédiatement à réduire l'accélération ou la décélération à zéro ; une fois le zéro atteint, l'APM utilise l'accélération et la vitesse de la destination de saut et passe à la nouvelle vitesse.

Exemple 13 : SCURVE – Saut avant le point central d'accélération ou de décélération

Dans l'exemple suivant, un saut se produit au niveau de la première ligne pointillée pendant l'accélération du premier CMOVE. La vitesse de la destination de saut étant inférieure à la vitesse du premier CMOVE, l'APM réduit l'accélération à zéro. La vitesse constante (accélération nulle) est atteinte au niveau de la deuxième ligne pointillée. A ce moment, l'APM commence à décélérer vers la nouvelle vitesse en utilisant l'accélération de la destination de saut. Finalement, le deuxième CMOVE se termine.

```
ACCEL     1000
VELOC     50000
BLOCK     1
JUMP      CTL01, 3
CMOVE     50000, INC, SCURVE
BLOCK     3
VELOC     5000
ACCEL     10000
CMOVE     15000, INC, SCURVE
```

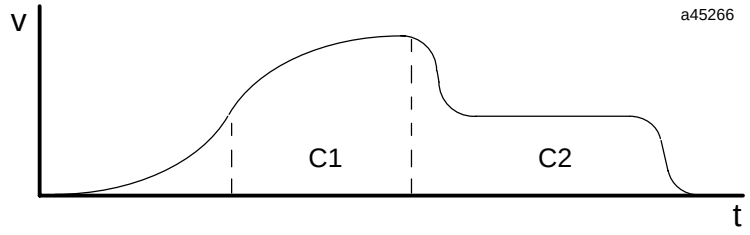


Figure 6-14. Saut avant le point central d'accélération ou de décélération

Dans le deuxième cas, le saut se produit pendant une accélération avec une vitesse de destination supérieure à la vitesse d'origine, ou pendant une décélération avec une vitesse de destination inférieure à la vitesse d'origine. Dans ce cas, l'APM continue avec l'accélération ou la décélération du premier mouvement. Cette accélération ou décélération est maintenue, comme pour une accélération linéaire, jusqu'à ce que l'axe approche de la nouvelle vitesse. La courbe en S normale est ensuite utilisée pour réduire l'accélération ou décélération à zéro.

Exemple 14 : SCURVE – Saut vers une vitesse supérieure pendant une accélération ou une vitesse inférieure pendant une décélération

Dans cet exemple, une commande JUMP est déclenchée pendant la phase initiale d'accélération (avant la première ligne pointillée) et la vitesse de la destination de saut est supérieure à la vitesse du mouvement en cours. La première ligne pointillée indique l'accélération maximale du premier CMOVE. Cette valeur est maintenue tandis que l'axe continue à accélérer, puis décélère jusqu'à ce qu'il revienne à la vitesse constante. La vitesse constante, deuxième ligne pointillée, indique le début du deuxième CMOVE. Ce mouvement continue jusqu'à atteindre une vitesse nulle à la fin du programme.

```
ACCEL     50000
VELOC     30000
BLOCK     1
JUMP      CTL02, 2
CMOVE     150000, INC, SCURVE
BLOCK     2
VELOC     90000
ACCEL     25000
CMOVE     500000, INC, SCURVE
```

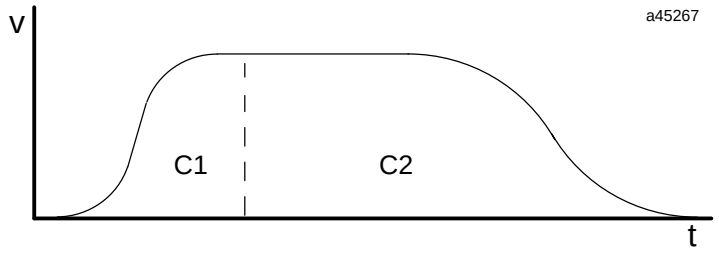


Figure 6-15. Saut vers une vitesse supérieure pendant une accélération ou une vitesse inférieure pendant une décélération

2.13. AUTRES REMARQUES RELATIVES AU MOUVEMENT PROGRAMMÉ

La durée maximale d'une accélération ou d'une décélération programmée est de 65,535 secondes. Si le temps d'accélération ou de décélération calculé est supérieur à cette durée, l'APM calcule l'accélération à utiliser pour une durée de 65,535 secondes. Pour obtenir des durées d'accélération supérieures, vous devez utiliser plusieurs CMOVE avec des vitesses croissantes ou décroissantes.

Exemple 15 : Durée d'accélération maximale

Les deux programmes suivants présentent un problème avec un temps d'accélération très long et une solution. Dans le premier cas, 120 secondes, deux minutes, sont nécessaires pour atteindre l'accélération programmée. Ce temps étant supérieur aux 65,535 secondes autorisées, l'APM calcule qu'une accélération de 184 permettrait d'atteindre la vitesse de 12000 en 65,535 secondes. Le profil de vitesse de gauche présente l'accélération de 184 légèrement supérieure utilisée. Il présente également une ligne pointillée indiquant l'accélération programmée jusqu'à la vitesse constante.

Une solution pour obtenir une accélération faible pendant de longues périodes consiste à diviser le mouvement en mouvements distincts avec des temps d'accélération individuels de 65,535 secondes. Cette méthode nécessite certains calculs. Chaque accélération ou décélération doit être divisée en mouvements distincts avec des temps d'accélération inférieurs à 65,535 secondes. Ainsi, pour autoriser une accélération de 100 pendant l'accélération et la décélération, trois mouvements seront nécessaires.

Le second programme, présenté à droite, montre comment le premier programme peut être divisé en trois parties. On obtient après calcul que la distance au point central de chaque accélération, lorsque la vitesse est de 6000, est de 180000, un quart de la distance nécessaire pour accélérer jusqu'à 12000. Un CMOVE initial utilise cette distance. Le CMOVE suivant accélère ensuite à sa vitesse avec le même taux d'accélération. Le PMOVE final est à la distance de point central, 180000 unités utilisateur, de la position finale. Le deuxième CMOVE décélère automatiquement à la vitesse du PMOVE lorsqu'il approche de sa position finale. Les lignes pointillées indiquent le début et la fin du deuxième CMOVE.

```
ACCEL    100
VELOC    12000
PMOVE    1500000, INC, LINEAR
```

```
ACCEL    100
VELOC    6000
CMOVE    180000, INC, LINEAR
VELOC    12000
CMOVE    1,140000, INC, LINEAR
VELOC    12000
PMOVE    180000, INC, LINEAR
```

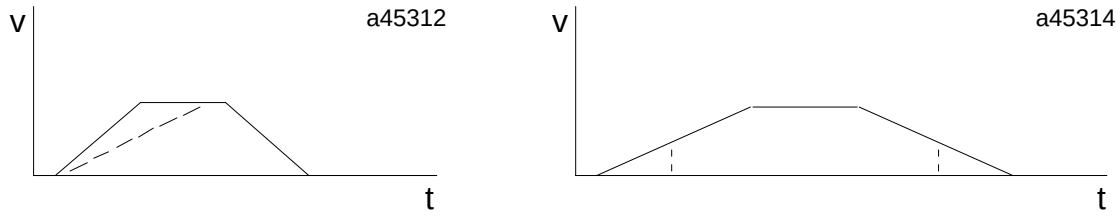


Figure 6-16. Durée d'accélération maximale

2.14. PAUSE AVEC L'APM

La pause (*Feedhold*) permet d'interrompre l'exécution sans mettre fin au programme, souvent pour étudier certains aspects du système. Tous les mouvements d'axes s'arrêtent alors à l'accélération programmée. Après la pause, l'exécution du programme reprend à l'accélération et à la vitesse programmées.

Pour commander une pause, vous devez mettre à 1 le bit %Q *Pause*. La pause cesse lorsque ce bit repasse à 0. L'activation du bit %Q *Interruption de tous les mouvements*, ou une erreur dont l'effet normal est d'entraîner une erreur d'arrêt, met fin à la fois à la pause et au programme. Pendant la pause, l'exécution de mouvements positifs (*Jog Plus*) et négatifs (*Jog Minus*) est autorisée, à l'exclusion de tout autre mouvement. Lorsque la pause est terminée et que l'exécution du programme reprend, l'APM se déplace à nouveau vers sa position initiale.

Exemple 16 : Pause

L'exemple suivant présente un profil de mouvement au moment de l'application de la pause. Le mouvement accélère jusqu'à la vitesse programmée, au taux d'accélération programmé. La pause est appliquée au niveau de la ligne pointillée, donc la vitesse décroît jusqu'à zéro à l'accélération programmée. L'opérateur commande ensuite un mouvement (Jog) avec le bit %Q *Mouvement négatif* (*Jog Minus*), ce qui est clairement indiqué par la *Vitesse de mouvement* (*Jog Velocity*) négative. Notez que l'accélération utilisée pendant le mouvement (Jog) est l'*Accélération de mouvement* (*Jog Acceleration*) courante, différente de l'accélération programmée. Notez également, que la pause doit être appliquée pendant toute la durée du mouvement (Jog). Après la fin du mouvement (Jog), la pause cesse et le programme reprend jusqu'à la fin.

```
ACCEL      1000
VELOC      2000
PMOVE      12000, INC, LINEAR
```

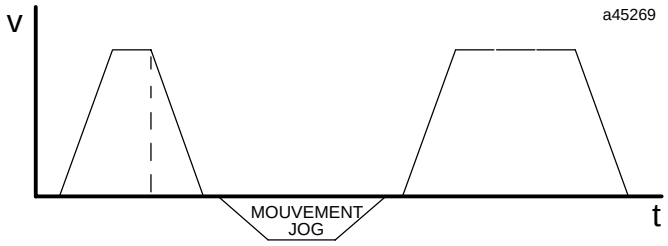


Figure 6-17. Pause

2.15. FORÇAGE DE VITESSE D'AVANCE

Certaines applications nécessitent une petite modification de la vitesse programmée pour gérer des modifications extérieures. La commande *Taux de forçage de vitesse d'avance* permet de modifier une vitesse programmée pendant l'exécution d'un programme. Lorsqu'un programme commence, le taux de forçage est réglé par défaut à 100 %. Les modifications de la vitesse d'avance effectuées avant l'activation du bit d'exécution de programme seront donc ignorées, mais une vitesse d'avance commandée pendant le même cycle que l'exécution du programme sera prise en compte.

Vous pouvez affecter à la vitesse d'avance un pourcentage compris entre 0 et 120 %. Lorsque vous utilisez une commande *Taux de forçage de vitesse d'avance*, l'APM multiplie de façon interne le pourcentage de vitesse d'avance par la vitesse programmée pour obtenir une nouvelle vitesse. Si l'axe est en mouvement, l'APM utilise le *Mode d'accélération de mouvement* (*Jog*) du mouvement en cours pour donner à la vitesse sa nouvelle valeur. Toutes les vitesses de mouvement futures seront affectés par le forçage de la vitesse d'avance. Notez que si vous appliquez une vitesse d'avance de 0 %, l'APM ne générera aucun mouvement jusqu'à ce que vous commandiez une nouvelle vitesse d'avance. Notez également que le bit %I *En mouvement* reste à 1 lorsque la vitesse d'avance est de 0 %.

Le *Taux de forçage de vitesse d'avance* n'a pas d'effet sur les mouvements non programmés tels que *Mouvement* (*Jog*), *Recherche position initiale* ou *Mouvement à vitesse constante.*

Exemple 17 : Forçage de vitesse d'avance

Pendant l'exécution de ce programme, nous commandons des modifications de vitesse d'avance de +/–10 %. Les lignes pointillées indiquent –10 %, les lignes de grands tirets indiquent +10 %.

```
ACCEL       1000
VELOC       6000
PMOVE       110000, INC, LINEAR
```

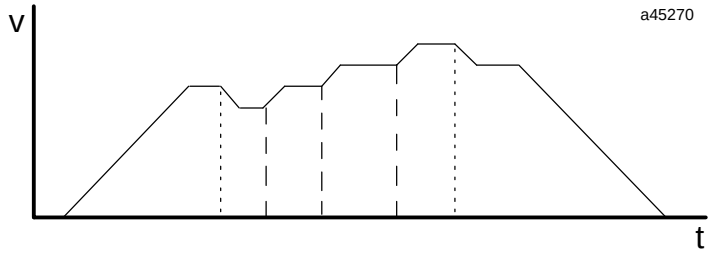


Figure 6-18. Forçage de vitesse d'avance

2.16. PROGRAMMATION MULTI–AXE

Dans un programme multi–axe, les blocs de synchronisation vous permettent de synchroniser les commandes de mouvement d'axe à des positions où la synchronisation est essentielle.

Exemple 18 : Programmation multi–axe

Cet exemple suppose que l'axe 1 contrôle le mouvement vertical et que l'axe 2 contrôle le mouvement horizontal. L'objectif est de déplacer un élément matériel d'un point A vers un point C aussi rapidement que possible tout en évitant l'obstacle qui empêche un mouvement direct entre A et C.

La solution la plus simple consiste à exécuter un mouvement du point A vers un point B, puis un mouvement de ce point B vers le point C. Cependant cette séquence fait perdre du temps. Une meilleure solution consisterait à commencer le mouvement horizontal avant d'atteindre le point B. Il a été déterminé que, lorsque l'axe 1 a atteint une position de 30000 unités utilisateur, l'axe 2 peut commencer son mouvement tout en évitant l'obstacle. Le segment de programme utilisé pourrait être le suivant :

```
BLOCK 10          CMOVE, 30000, INC, AXIS 1
BLOCK 20 [SYNC]   PMOVE, 50000, INC, AXIS 1
                  PMOVE, 150000, INC, AXIS 2
```

Lorsque ce programme est exécuté, l'axe 1 commence immédiatement son mouvement de 30000 unités. L'axe 2 ignore la première commande, puisqu'elle s'applique uniquement à l'axe 1, et se place sur le bloc de synchronisation. L'axe 2 attend que l'axe 1 atteigne le bloc de synchronisation avant de continuer l'exécution du programme. Lorsque l'axe 1 atteint la marque des 30000 unités, il commence le PMOVE de 50000 unités à partir du bloc de synchronisation sans interruption (le premier mouvement était un CMOVE). L'axe 1 ayant atteint le bloc de synchronisation, l'axe 2 commence son mouvement de 150000 unités. Le profil de position de la figure 6–19 montre que l'axe 1 termine son mouvement avant de s'arrêter à la fin du PMOVE. Lorsque l'axe 2 atteint le point C, il s'arrête également.

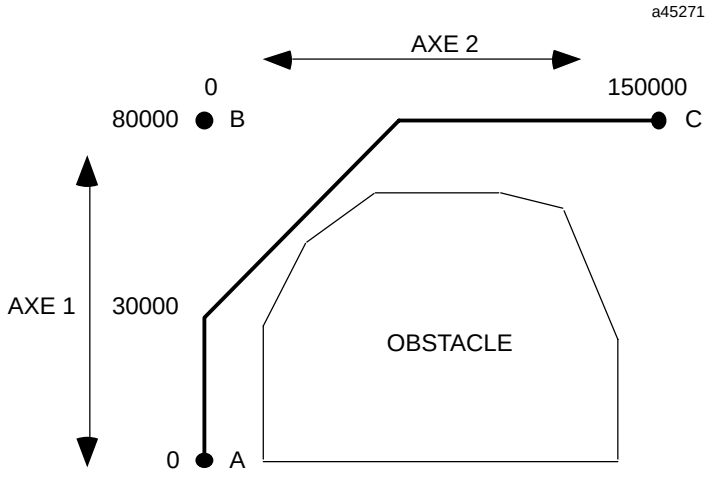


Figure 6-19. Programmation multi–axe

Si ce segment de programme n'est pas situé au début d'un programme et si, pour une raison quelconque, l'axe 2 n'a pas encore atteint le bloc 20 lorsque l'axe 1 s'est déplacé de 30000 comptages, une erreur se produit. L'axe 1 continue jusqu'à 80000 comptages, et l'APM signale dans le code d'état une erreur de bloc de synchronisation pendant un CMOVE.

S'il est impératif que les axes se synchronisent au bloc 20, modifiez le bloc 10 en PMOVE. L'axe 1 s'arrêtera cependant momentanément à 30000 comptages.

2.17. PARAMÈTRES DE L'APM

L'APM conserve en mémoire 256 paramètres (de 0 à 255) de type double mot. Vous pouvez utiliser ces valeurs en tant que paramètres dans des commandes de mouvement ACCEL, VELOC, DWELL, PMOVE et CMOVE. Notez que les limites de plage s'appliquent également et que des erreurs risquent de se produire si un paramètre contient une valeur hors plage. Les dix derniers paramètres sont particuliers. L'APM peut les utiliser pour charger des données, ce qui risque d'écraser des données utilisateur. Le tableau suivant décrit les fonctions de ces paramètres.

Paramètre	Fonction particulière
246 – 253	Réservé
254	Stocke la valeur *Position d'échantillonnage* de l'axe 2 (302 uniquement)
255	Stocke la valeur *Position d'échantillonnage* de l'axe 1

Tous les paramètres sont remis à zéro après un redémarrage ou après l'enregistrement d'une configuration d'APM par l'API. Trois moyens vous sont offerts pour définir les paramètres : la commande de programme de mouvement LOAD, la commande immédiate *Chargement immédiat de paramètre*, et le bloc fonctionnel COMM_REQ, dans l'API. L'annexe B contient une description du bloc fonctionnel COMM_REQ. L'affectation d'une valeur à un paramètre écrase sa valeur précédente. Vous pouvez modifier la valeur des paramètres pendant l'exécution d'un programme, mais la modification doit intervenir avant que l'APM ne commence à exécuter le bloc précédant le bloc qui utilise le paramètre.

Page laissée blanche intentionnellement

Annexe A Codes d'erreur

1. GÉNÉRALITÉS

Le mot *Code d'état* des mots d'état %AI contient un code décrivant l'erreur indiquée lorsque le bit d'état *Erreur* est à 1. Il existe trois catégories d'erreurs signalées par le *Code d'état*.

- Erreurs de programmation non bloquantes
- Erreurs de programmation bloquantes arrêtant le servomécanisme
- Erreurs matérielles (décodeur désynchronisé, commutateurs "Run" ouverts, perte de communication avec la console de programmation, etc.)

Remarque

Le voyant d'état du plastron du module clignote lentement (4 fois/s) pour les erreurs d'état uniquement et rapidement (8 fois/s) pour les erreurs qui entraînent l'arrêt du servomécanisme.

Les codes d'erreur sont placés dans le mot %AI *Code d'état*, dont le format est décrit ci–dessous :

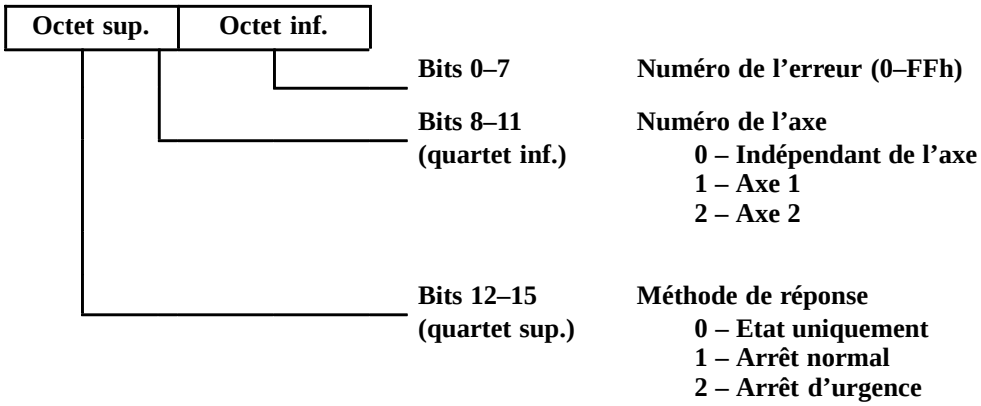


Figure A-1. Organisation du Code d'état

2. MÉTHODES DE RÉPONSE

1. **Erreurs d'état uniquement :** mise à 1 de l'indicateur d'erreur et mise à jour du code d'état, sans affecter le mouvement.

2. **Erreurs d'arrêt normal :** interruption interne de tout mouvement en cours. Les bits %I *Circuit de commande activé* et *Axe prêt* sont mis à 0 après le délai de désactivation de circuit de commande (*Drive Disable Delay)* configuré.

3. **Erreur d'arrêt d'urgence :** interruption instantanée de tout mouvement par mise à zéro de la tension de sortie analogique. Les bits %I *Circuit de commande activé* et *Axe prêt* sont mis à 0 après le *Temps de désactivation du circuit de commande (Disdly)* configuré.

Tableau A-1. Codes d'erreur du mot d'état

Numéro de l'erreur (hexadécimal)	Réponse	Description
0	Aucune	Aucune erreur
Erreurs de configuration		
2	Etat uniquement	Donnée mise à l'échelle trop importante
3	Etat uniquement	Position initiale > fin de course positive, utilisation de fin de course positive
4	Etat uniquement	Position initiale < fin de course négative, utilisation de fin de course négative
Erreurs de paramètre de configuration		
10	Etat uniquement	Constante de temps de boucle de position trop grande
11	Etat uniquement	Constante de temps de boucle de position trop petite
12	Etat uniquement	Dépassement de calcul pour la constante de temps de boucle de position
1E	Etat uniquement	Commande immédiate Vitesse de mouvement (Jog) hors plage
1F	Etat uniquement	Commande immédiate Accélération de mouvement (Jog) hors plage
Erreurs de configuration		
20	Etat uniquement	Accélération du programme hors plage
21	Etat uniquement	Accélération du programme trop faible, utilisation de 32 comptages/s/s
22	Etat uniquement	Vitesse mise à l'échelle supérieure à 1 million de comptages/s
23	Etat uniquement	Vitesse du programme nulle, utilisation de 1 comptage par seconde
24	Arrêt normal	Position du programme trop grande
25	Arrêt normal	Destination de saut inconditionnel non trouvée
26	Arrêt normal	Erreur de masque de saut
27	Arrêt normal	Erreur de masque d'attente
28	Arrêt normal	Paramètre de position trop grand
29	Etat uniquement	Temps de pause supérieur à 60 secondes, utilisation de 5 secondes
Erreurs d'incrément de position		
2C	Etat uniquement	Incrément de position hors plage
Erreurs de retour à la position initiale		
30	Etat uniquement	Retour à la position initiale avec circuit de commande désactivé
31	Etat uniquement	Retour à la position initiale pendant programme sélectionné
32	Etat uniquement	Retour à la position initiale pendant forçage N/A
33	Etat uniquement	Retour à la position initiale pendant mouvement (Jog)
34	Etat uniquement	Retour à la position initiale pendant mouvement à vitesse constante
36	Etat uniquement	Retour à la position initiale avec bit d'interruption de tous les mouvements à "1"
Erreurs de mouvement à vitesse constante		
39	Etat uniquement	Mouvement à vitesse constante avec circuit de commande désactivé
3A	Etat uniquement	Mouvement à vitesse constante pendant programme sélectionné
3B	Etat uniquement	Mouvement à vitesse constante pendant cycle de retour à la position initiale
3C	Etat uniquement	Mouvement à vitesse constante pendant mouvement (Jog)
3D	Etat uniquement	Mouvement à vitesse constante avec bit Interruption de tous les mouvements à "1"
3E	Etat uniquement	Donnée de mouvement à vitesse constante supérieure à 8 388 607 uu/s
3F	Etat uniquement	Donnée de mouvement à vitesse constante supérieure 1 million comptages/s

Tableau A. Codes d'erreur du mot d'état (suite)

Numéro de l'erreur (hexadécimal)	Réponse	Description
Erreurs de mouvement (Jog)		
40	Etat uniquement	Mouvement (Jog) pendant retour à la position initiale
41	Etat uniquement	Mouvement (Jog) pendant mouvement à vitesse constante
42	Etat uniquement	Mouvement (Jog) pendant forçage N/A
43	Etat uniquement	Mouvement (Jog) pendant programme sélectionné sans pause
Erreurs de forçage N/A		
47	Etat uniquement	Forçage N/A pendant mouvement (Jog)
48	Etat uniquement	Forçage N/A pendant mouvement à vitesse constante
49	Etat uniquement	Forçage N/A pendant programme sélectionné
Erreur de réglage position		
50	Etat uniquement	Réglage position pendant programme sélectionné
51	Etat uniquement	Donnée de réglage position hors plage
52	Etat uniquement	Réglage position sans erreur en limite
Erreurs de limites de fin de course et de comptage		
56	Etat uniquement	Position commandée supérieure à fin de course positive ou limite de comptage supérieure
57	Etat uniquement	Position commandée inférieure à fin de course négative ou limite de comptage inférieure
Erreurs de circuit de commande désactivé		
5B	Arrêt normal	Circuit de commande désactivé pendant mouvement
5C	Arrêt normal	Circuit de commande pendant programme actif
Erreurs logicielles		
5F	Etat uniquement	Erreur logicielle (appeler service client GE Fanuc)
Erreurs de programme et de sous–programme		
61	Arrêt normal	Sous–programme absent de la liste
62	Arrêt normal	Erreur d'appel (sous–programme déjà actif)
63	Arrêt normal	Commande de fin de sous–programme rencontrée dans programme
64	Arrêt normal	Commande de fin de programme rencontrée dans sous–programme
Erreurs d'exécution de programme		
70	Etat uniquement	Demande de programme 0 avec d'autres programmes actifs
71	Etat uniquement	Trop de demandes de programmes au cours du même cycle API
72	Etat uniquement	Demande de programme 1–10 avec programme multi–axe actif
73	Etat uniquement	Deux demandes de programmes au cours du même cycle avec programme actif
74	Etat uniquement	Deux demandes de programmes pour le même axe, programme de plus petit numéro exécuté
75	Etat uniquement	Demande de programme vide ou non valide
Erreurs de condition d'exécution de programme		
80	Etat uniquement	Exécution de programme pendant cycle de retour à la position initiale
81	Etat uniquement	Exécution de programme pendant mouvement (Jog)
82	Etat uniquement	Exécution de programme pendant mouvement à vitesse constante
83	Etat uniquement	Exécution de programme pendant forçage N/A
84	Etat uniquement	Exécution de programme pendant programme sélectionné
85	Etat uniquement	Exécution de programme avec bit Interruption de tous les programmes à "1"
86	Etat uniquement	Exécution de programme avec Position valide à "0"
87	Etat uniquement	Exécution de programme avec Circuit de commande activé à "0"
Erreurs de bloc de synchronisation de programme		
8C	Etat uniquement	Erreur de bloc de sync pendant CMOVE
8D	Etat uniquement	Erreur de bloc de sync pendant Jump
Erreurs EEPROM		
90	Etat uniquement	Erreur de programmation de la mémoire flash EEPROM

Tableau A. Codes d'erreur du mot d'état (suite)

Numéro de l'erreur (hexadécimal)	Réponse	Description
Erreurs d'interrupteur de fin de course matérielle		
A0	Erreur d'urgence	Erreur d'interrupteur de fin de course (+)
A1	Erreur d'urgence	Erreur d'interrupteur de fin de course (–)
Erreurs matérielles		
A8	Erreur d'urgence	Perte de synchronisation
A9	Erreur d'urgence	Perte de quadrature du codeur
AA	Arrêt normal	Défaut d'entrée analogique
Erreurs particulières		
E0	Etat uniquement	Erreur de type de boucle personnalisée
EF	Etat uniquement	Erreur de nombre d'axes dans le microprogramme (microprogramme mono–axe dans module à deux axes, microprogramme à deux axes dans module mono–axe

Annexe B

Téléchargement de paramètres avec COMM_REQ

Cette annexe décrit une méthode permettant de charger la mémoire des paramètres de l'APM à partir de l'API en utilisant un bloc fonctionnel COMM_REQ avec le code de commande E501h. Ce bloc fonctionnel peut transmettre jusqu'à 16 valeurs de paramètres APM à la fois. Vous devez définir la longueur totale des données de COMM_REQ à 68 octets (34 mots), organisés de la façon suivante :

Décalage de mot	Décalage d'octet	Donnée
0	0 – 1	Numéro de paramètre de départ (0 – 255)
1	2 – 3	Nombre de paramètres à charger
2 – 3	4 – 7	Donnée du premier paramètre (4 octets)
4 – 5	8 – 11	Donnée du deuxième paramètre (4 octets)
...	...	...
32 – 33	64 – 67	Donnée du seizième paramètre (4 octets)

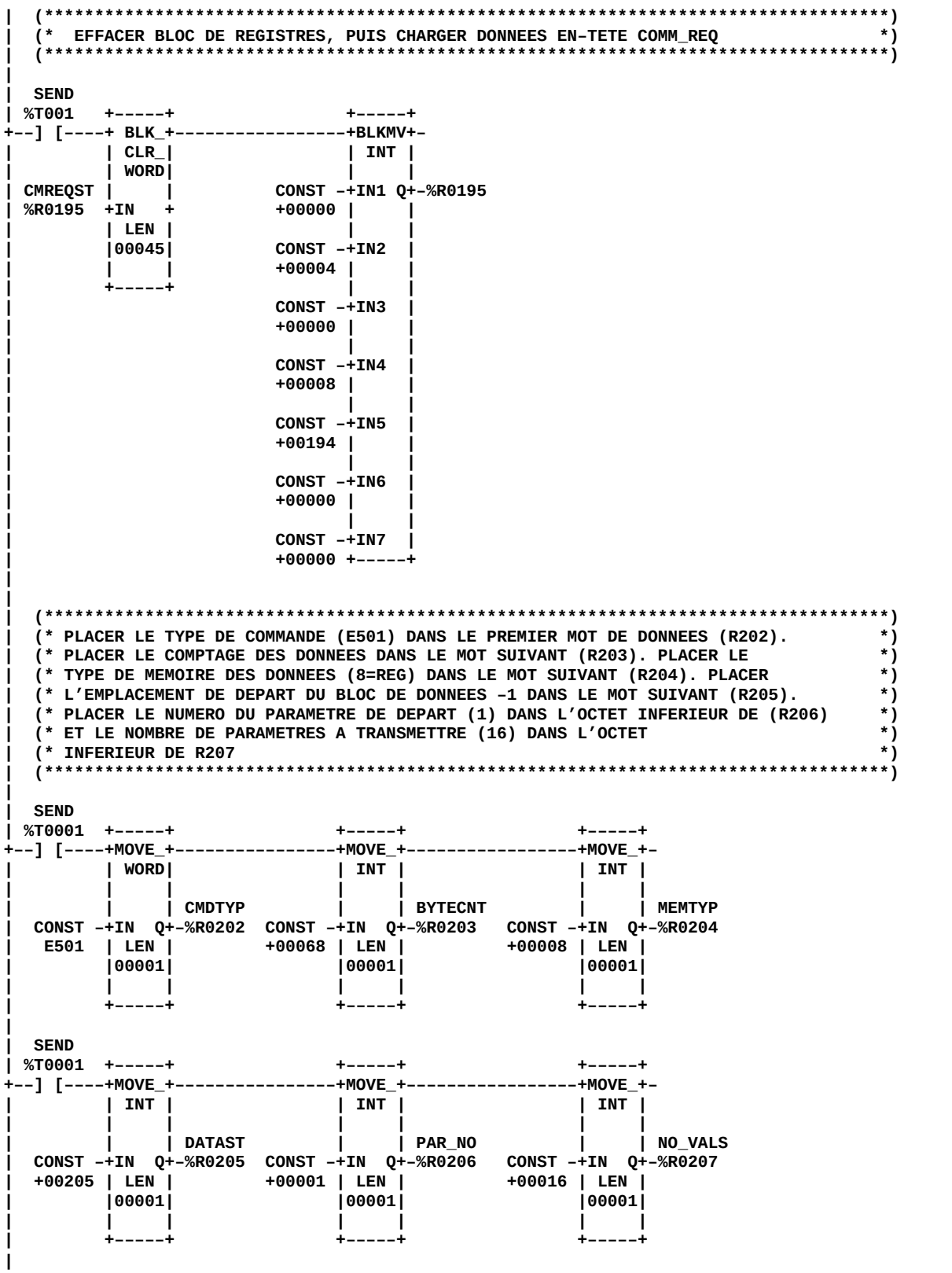
La mémoire des paramètres sera chargée avec le nombre de paramètres spécifié dans le décalage de mot 1. Cependant, le bloc de données de 68 octets doit toujours être initialisé dans l'API. Si le dernier paramètre à charger est supérieur à 255, la commande COMM_REQ sera rejetée. Le segment de programme d'API suivant présente un exemple de téléchargement de bloc de paramètres.

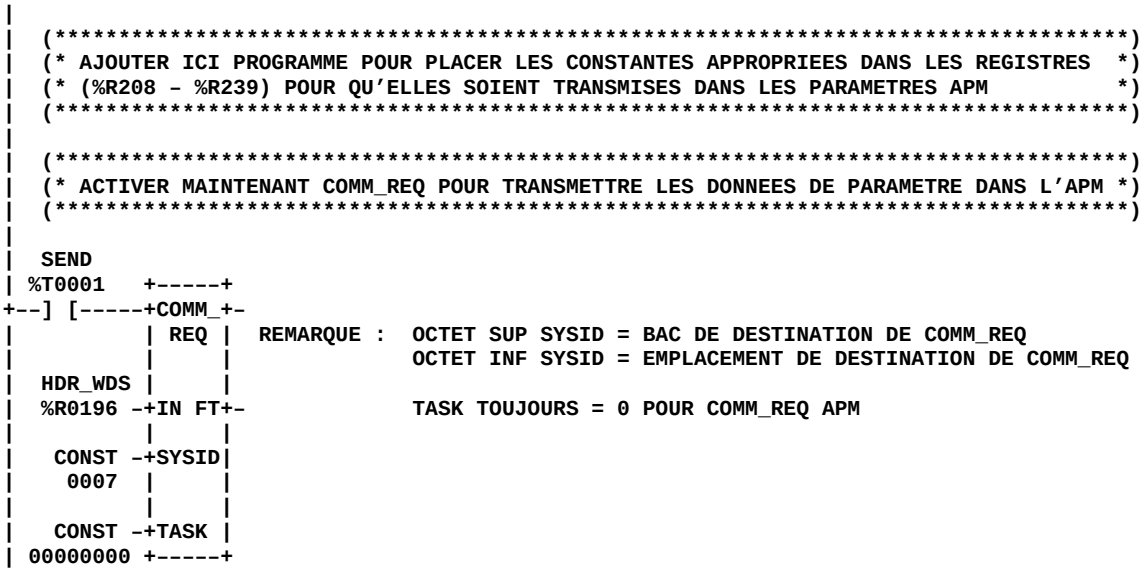
```
REFERENCE     MNEMONIQUE     DESCRIPTION DE LA REFERENCE
--------      ----------     ---------------------------------

%R0195          CMREQST      MOT D'ETAT COMM_REQ (mis a jour par COM_REQ)
%R0196          HDR_WDS      LONGUEUR EN-TETE COMM_REQ EN MOTS (TOUJOURS 4)
%R0197          NO_WAIT      PAS D'ATTENTE (TOUJOURS 0)
%R0198          STMEMTP      TYPE DE MEMOIRE D'ETAT (8=REG)
%R0199          STLOCM1      EMPLACEMENT MOT D'ETAT MOINS 1 (194 = %R0195)
%R0200          NO_USE1      INUTILISE (MIS A ZERO PAR BLKMV)
%R0201          NO_USE2      INUTILISE (MIS A ZERO PAR BLKMV)

%R0202          CMDTYP       TYPE DE COMMANDE (E501 POUR APM)
%R0203          BYTECNT      COMPTAGE DE DONNEES (OCTET)
%R0204          MEMTYP       TYPE DE DONNEES DE LA MEMOIRE (8=REG)
%R0205          DATAST       DEBUT BLOC DE DONNEES -1 (205 = %R0206)
%R0206          PAR_NO       NUMERO DE PARAMETRE DE DEPART
%R0207          NO_VALS      NOMBRE DE PARAMETRE A TRANSMETTRE

%R0208-%R0239   PAR_DAT      Donnees pour 16 parametres (32 mots)
```


(***)
(* EFFACER BLOC DE REGISTRES, PUIS CHARGER DONNEES EN-TETE COMM_REQ *)
(***)
SEND
%T001
BLK
CLR_WORD
CMREQST %R0195
IN
LEN 00045
BLKMV
INT
CONST +00000 IN1 %R0195
CONST +00004 IN2
CONST +00000 IN3
CONST +00008 IN4
CONST +00194 IN5
CONST +00000 IN6
CONST +00000 IN7
(***)
(* PLACER LE TYPE DE COMMANDE (E501) DANS LE PREMIER MOT DE DONNEES (R202). *)
(* PLACER LE COMPTAGE DES DONNEES DANS LE MOT SUIVANT (R203). PLACER LE *)
(* TYPE DE MEMOIRE DES DONNEES (8=REG) DANS LE MOT SUIVANT (R204). PLACER *)
(* L'EMPLACEMENT DE DEPART DU BLOC DE DONNEES -1 DANS LE MOT SUIVANT (R205). *)
(* PLACER LE NUMERO DU PARAMETRE DE DEPART (1) DANS L'OCTET INFERIEUR DE (R206) *)
(* ET LE NOMBRE DE PARAMETRES A TRANSMETTRE (16) DANS L'OCTET *)
(* INFERIEUR DE R207 *)
(***)
SEND
%T0001
MOVE
WORD
CONST E501 IN
LEN 00001
CMDTYP %R0202
CONST +00068
BYTECNT %R0203
CONST +00008
MEMTYP %R0204
SEND
%T0001
MOVE
INT
CONST +00205 IN
LEN 00001
DATAST %R0205
CONST +00001
PAR_NO %R0206
CONST +00016
NO_VALS %R0207


(***)
(* AJOUTER ICI PROGRAMME POUR PLACER LES CONSTANTES APPROPRIEES DANS LES REGISTRES *)
(* (%R208 – %R239) POUR QU'ELLES SOIENT TRANSMISES DANS LES PARAMETRES APM *)
(***)
(***)
(* ACTIVER MAINTENANT COMM_REQ POUR TRANSMETTRE LES DONNEES DE PARAMETRE DANS L'APM *)
(***)
SEND
%T0001
COMM_
REQ
HDR_WDS
%R0196
IN
FT
CONST
SYSID
0007
CONST
TASK
00000000
REMARQUE :
OCTET SUP SYSID = BAC DE DESTINATION DE COMM_REQ
OCTET INF SYSID = EMPLACEMENT DE DESTINATION DE COMM_REQ
TASK TOUJOURS = 0 POUR COMM_REQ APM

Page laissée blanche intentionnellement

Annexe C Spécifications

Cette annexe fournit les spécifications du module APM et des connexions d'E/S.

1. SPÉCIFICATIONS DU MODULE

Température de fonctionnement :	de 0 à 55 °C (en entrée)
Température de stockage :	de –40 à +85 °C
Humidité :	de 5 à 95 % (sans condensation)
Tension d'alimentation :	5 Vcc sur le fond de bac
Courant d'alimentation :	800 mA + (1,4 x courant absorbé par le codeur) + (95 mA pour le miniconvertisseur SNP, si utilisé)
Nombre maximum de modules par système :	API modèles 311, 313, 321, 323 : 3 APU301 dans la platine d'UC uniquement (nombre limité par l'alimentation) 2 APU302 dans la platine d'UC uniquement (nombre limité par les données %AI) API modèle 331 : 8 APU301 ou 4 APU302 dans les platines d'UC et d'extension (nombre limité par les données %AI) Maximum de 3 par platine (nombre limité par l'alimentation) API modèle 341 : 14 APU301 ou APU302 (nombre limité par l'alimentation) Maximum de 2 dans la platine d'UC et 3 dans chaque platine d'extension

2. SPÉCIFICATIONS DES E/S

Vous trouverez ci–dessous les spécifications et les circuits des connexions d'E/S :

2.1. COMMANDE DE VITESSE

Sortie du convertisseur N/A avec les caractéristiques suivantes :

- Résolution : 13 bits, signe compris
- Linéarité : 0,02 % de sortie pleine échelle
- Tension de décalage : ± 500 µV au maximum
- Sortie maximale : ± 10 V , ± 0,3 V
- Résistance de charge minimale : 2000 Ω
- Tension entre le commun analogique et la terre : ± 1 V

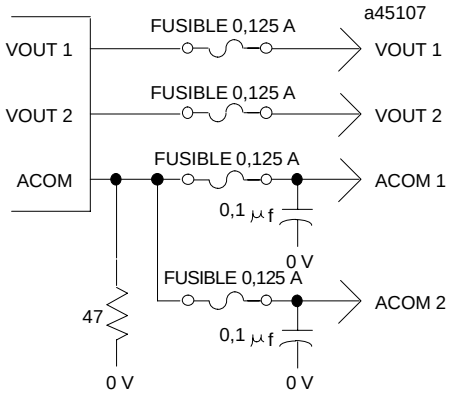


Figure C-1. Circuit de commande de vitesse

2.2. SORTIE DU RELAIS D'ACTIVATION

Contact de relais à semi–conducteurs CC normalement ouvert ; contacts prévus pour 30 V, 100 mA CC. Charge résistive uniquement. La fuite de courant à l'état bloqué est de 10 µA au maximum.

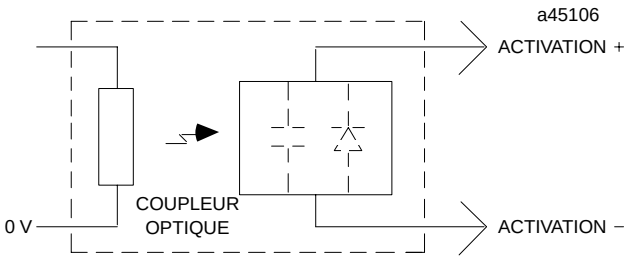
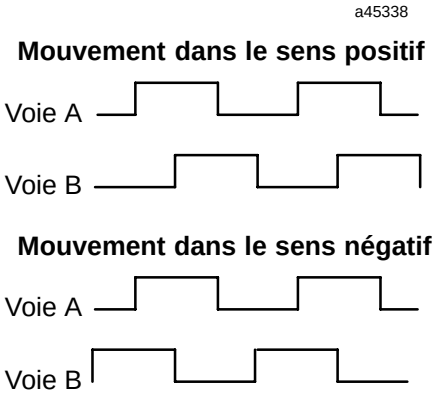


Figure C-2. Circuit de sortie du relais d'activation

2.3. ENTRÉES DU CODEUR

- Type d'entrée : 5 V référencée ou non référencée
- Impédance d'entrée : 4000 Ω (mode commun)
- Seuil d'entrée : non référencée : ±1 V nominal (± 0,4 V), référencée : +0,5 V nominal (± 0,4 V)
- Plage du mode commun d'entrée : ± 15 V
- Tension d'entrée non référencée : + 15 V maximum.
- Fréquence d'entrée maximale : 250 kHz/voie (fréquence de comptage X4 = 1000 kHz)
- Filtre d'entrée : rejette les impulsions de bruit inférieures à 1 microseconde.
- Alimentation du codeur : une alimentation +5 V à limitation de courant non isolée est disponible sur les connecteurs d'E/S de la façade de l'APM pour un ou plusieurs codeurs. La charge maximale doit être limitée à 500 mA à 40 °C, 300 mA à 55 °C.
- Tolérance de quadrature : 90 degrés ± 45 degrés.
- Détection d'erreur de quadrature : les transitions simultanées des entrées des voies A et B sont interprétées comme une perte de quadrature.
- Voie Z (fonctionnement du marqueur : un front montant sur la voie du marqueur sera utilisé pour verrouiller la position en cours du codeur. La largeur d'impulsion minimale est de 4 µs.)
- Direction de déplacement du codeur : si la voie A est en avance sur la voie B, le mouvement est dans le sens positif :



2.4. ENTRÉES D'ÉCHANTILLONNAGE

- Type d'entrée : 5 V référencée ou non référencée
- Impédance d'entrée : 4000 Ohms (mode commun)
- Seuil d'entrée : non référencée : +1 V nominal (± 0,4 V), référencée : +0,5 V nominal (± 0,4 V)
- Plage du mode commun d'entrée : ± 15 V
- Tension d'entrée non référencée : + 15 V maximum
- Largeur d'impulsion minimale nécessaire : 3 µs
- Une impulsion sur l'entrée d'échantillonnage entraîne le report de la plus récente valeur de *Position actuelle* dans la partie "Position d'échantillonnage" des données %AI.
- Retard d'acquisition de position :
  de 0 à 1 ms (APU301)

  de 0 à 2 ms (APU302).

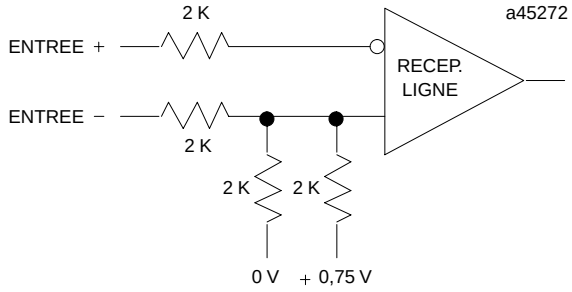


Figure C-3. Circuit des entrées du codeur et d'échantillonnage

2.5. ALIMENTATION DU CODEUR

- Alimentation du codeur : une alimentation +5 V à limitation de courant non isolée est disponible sur les connecteurs d'E/S de la façade de l'APM pour un ou plusieurs codeurs. La charge maximale doit être limitée à 500 mA à 40 °C, 300 mA à 55 °C.

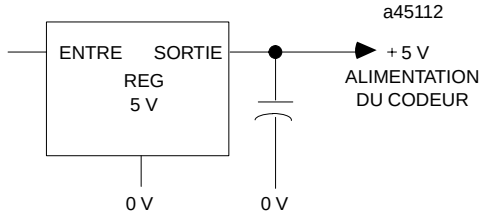


Figure C-4. Circuit d'alimentation du codeur

2.6. ENTRÉES NUMÉRIQUES À UTILISATION GÉNÉRALE

- Commutateur de position initiale axe 1, CTL03
- Commutateur de dépassement positif axe 1, CTL05
- Commutateur de dépassement négatif axe 1, CTL06
- Commutateur de position initiale axe 2 , CTL04
- Commutateur de dépassement positif axe 2, CTL07
- Commutateur de dépassement négatif axe 2, CTL08

Isolation optique avec les spécifications suivantes :

- Source/puits CC à isolation optique (coupleur d'entrée bidirectionnel)
- Seuil d'activation d'entrée : de 18 à 30 V
- Seuil de désactivation d'entrée : de 0 à 4 V
- Résistance d'entrée : 5400 Ω ± 10 %
- Filtre d'entrée : 5 ms nominal
- Tension d'isolation : perturbation de crête de 1500 V

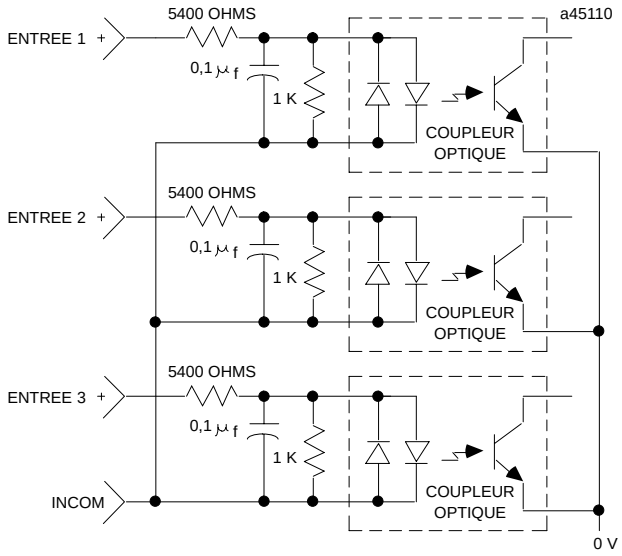


Figure C-5. Circuit des entrées à utilisation générale

2.7. SORTIES À UTILISATION GÉNÉRALE

- CTL09
- CTL10
- CTL11
- CTL12

5 V de bas niveau, 5 V d'alimentation générés par l'APM

- Le circuit d'attaque est un tampon CMOS avec une impédance de sortie de 20 ohms
- Chaque sortie est protégée par un fusible 1/8 A soudé (non remplaçable par l'utilisateur)
- Possibilité de commander des voyants d'entrée logique ou des indicateurs
- A ne pas utiliser pour commander des charges inductives, y compris les bobines de relais

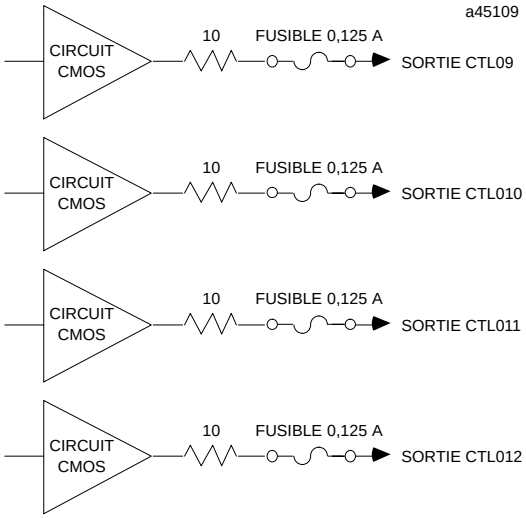


Figure C-6. Circuit des sorties à utilisation générale

2.8. ENTRÉES ANALOGIQUES

Convertisseur A/N 12 bits plus le signe avec les spécifications suivantes :

- Plage d'entrée : ± 10 V
- Impédance d'entrée : 50 kΩ
- Plage de mode commun : ± 20 V
- Résolution : 12 bits plus le bit de signe
- Linéarité : < 1 bit de poids faible
- Précision : 2 % en lecture ± 4 bits de poids faible
- Facteur d'échelle : +10 V = 32000, –10 V = –32000
- Fréquence de rafraîchissement : 16 ms (temps de scrutation de l'API non compris)

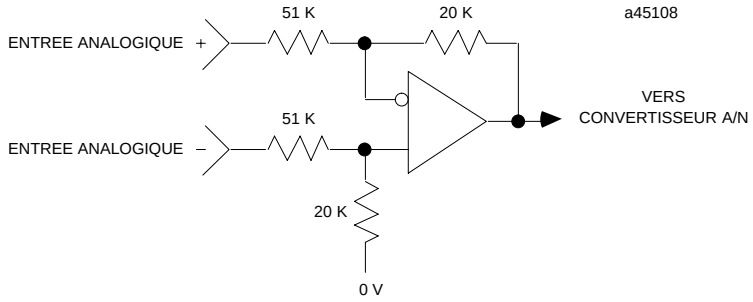


Figure C-7. Circuit d'entrée analogique

2.9. SPÉCIFICATIONS DU CONNECTEUR DE CÂBLE D'E/S

Vous pouvez réduire la longueur du câble reliant le connecteur d'E/S à un bornier externe en fonction des exigences de votre installation. Reportez–vous au tableau C–1 pour connaître les correspondances entre les fils du câble et les broches du connecteur. Reportez–vous également aux figures 2–4, 2–5, 2–6, 2–7 et aux tableaux 2–2, 2–3, 2–4, 2–5 pour les exigences de câblage particulières.

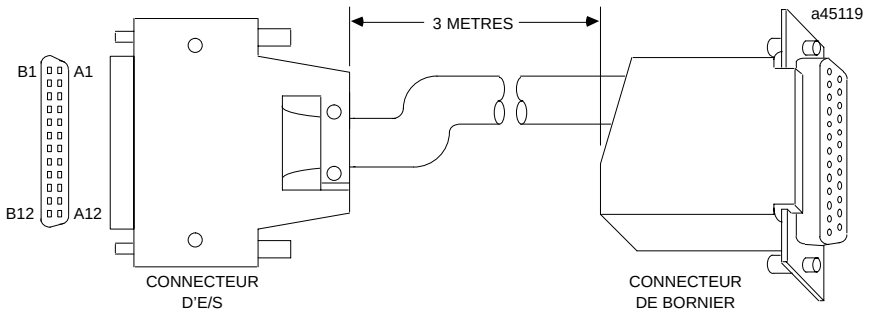


Figure C-8. Spécifications du câble du connecteur d'E/S

Tableau C-1. Codage des fils du câble d'E/S

Numéros des broches du connecteur d'E/S	Codes de couleur des fils du câble	Numéros des broches du connecteur 25 broches *
Pas de connexion	Fil 1 paire 1 (noir)	25
A1	Fil 2 paire 1 (rouge)	12
B1	Fil 1 paire 2 (noir)	24
A2	Fil 2 paire 2 (orange)	11
B2	Fil 1 paire 3 (noir)	23
A3	Fil 2 paire 3 (marron)	10
B3	Fil 1 paire 4 (noir)	22
A4	Fil 2 paire 4 (blanc)	9
B4	Fil 1 paire 5 (marron)	15
A5	Fil 2 paire 5 (blanc)	2
B5	Fil 1 paire 6 (bleu)	14
A6	Fil 2 paire 6 (blanc)	1
B6	Fil 1 paire 7 (rouge)	16
A7	Fil 2 paire 7 (blanc)	3
B7	Fil 1 paire 8 (vert)	17
A8	Fil 2 paire 8 (blanc)	4
B8	Fil 1 paire 9 (violet)	21
A9	Fil 2 paire 9 (blanc)	8
B9	Fil 1 paire 10 (orange)	20
A10	Fil 2 paire 10 (blanc)	7
B10	Fil 1 paire 11 (jaune)	19
A11	Fil 2 paire 11 (blanc)	6
B11	Fil 1 paire 12 (gris)	18
A12	Fil 2 paire 12 (blanc)	5
B12	Fil de drain (blindage)	13

* Identique aux numéros des bornes du bornier.

GE Fanuc propose un câble d'E/S équipé composé d'un connecteur d'E/S 24 broches, d'un câble et d'un connecteur de bornier de type D 25 broches (référence IC693CBL311A). Le connecteur d'E/S 24 broches seul (adapté au connecteur d'E/S du plastron du module APM) est disponible sous la référence produit GE Fanuc IC693ACC317A. Vous trouverez ci–dessous les références produit Fujitsu des connecteurs de câble d'E/S 24 broches disponibles auprès de votre distributeur Fujitsu.

FCN–361J024–AU	Embase à oeillet soudée
FCN–360C024–B	Capot (pour embase ci–dessus)
FCN–363J024*	Embase pour fil à sertir
FCN–363J–AU*	Broche à sertir (pour embase ci–dessus, 24 sont nécessaires)
FCN–360C024–B*	Capot (pour embase ci–dessus)
FCN–367J024–AUF	Embase IDC (plate) – couvercle fermé
FCN–367J024–AUH	Embase IDC (plate) – couvercle ouvert

* Disponible sous la référence produit GE Fanuc IC693ACC317A.

Annexe D Informations de commande

Module de commande d'axe pour API Série 90–30	IC693APU301 (1 axe) IC693APU302 (2 axes)
Logiciel Motion Programmer	IC641SWP025
Manuel de programmation de l'APM	GFK–0664A
Manuel utilisateur de l'APM – Axes indépendants	GFK–0840B
Manuel utilisateur de l'APM – Axes suiveurs	GFK–0781A
Câble d'E/S (3 mètres)	IC693CBL311A (Comprend un connecteur d'E/S 24 broches, un câble, et un connecteur de bornier de type D 25 broches).
Bornier	Weidmuller RD25 910648 ou équivalent (doit être compatible avec le câble d'E/S IC693CBL311)
Connecteur d'E/S 24 broches (par 10)	IC693ACC317A
Kit miniconvertisseur de communications série	IC690ACC901

Page laissée blanche intentionnellement

Symboles

A

B

C

D

E

F

G

I

J

L

M

N

P

R

S

T

U

V

W