Manuel de référence
Des modules
SIMPA et SIMPA Micropas

Date : 20.10.05

Référence : BLN48250.DOC

Révision : 16

Auteur : B.LOPEZ

FICHE DE MODIFICATION DOCS MI

Documentation concernée : Manuel de référence des Modules SIMPA et SIMPA Micropas
réf. : BLN48250.DOC

Date et demandeur de la (des) modification(s)	Type (corrective ou Evolutive) et nature de la modification(s) : (noter chapitre, paragraphes concernés)	Approbation de la (des) modification(s)	Mise en place de la (des) modification(s)	Indice
B.LOPEZ 14/02/95	Création			0
B.LOPEZ 17/10/05	Evolution § V.3 Description de l'effet d'un défaut matériel : X - arrêt moteur - arrêt séquence - coupure puissance	Nom : B.LOPEZ Date : 20/10/05 Oui ☒ Non ❑ motif du refus :	Personne chargée de la réalisation : N.ROUMEGOUX Date réalisation : 20/10/05	15

SOMMAIRE

AVANT PROPOS

Le présent document décrit les fonctionnalités communes à l'ensemble des produits SIMPA.
Pour chaque produit spécifique, l'utilisateur aura soin d'y associer comme complément indispensable le "Manuel d'utilisation du produit" traitant des particularités propres du module utilisé, notamment sa configuration et sa connectique spécifique.

Ce manuel décrit aussi bien les fonctionnalités des modules SIMPA en version pas entier ½ pas, qu'en version micropas (SIMPA micropas). Hormis les limites de paramètres et les interrogations des modules, les différences restent minimes. Elles sont précisées à chaque fois que cela est nécessaire.

I - INTRODUCTION

Les produits de la famille SIMPA développés par la société Midi Ingénierie sont des unités intelligentes destinées à la commande de moteurs pas à pas. Ces modules intègrent la fonction indexeur et pour la plupart l'électronique de puissance propre à la commande des moteurs pas à pas. Certaines cartes intègrent jusqu'à 4 modules SIMPA.

Chaque module SIMPA comprend : l'ensemble des fonctions et composants nécessaires au contrôle d'un axe moteur (pas à pas) une unité logique à microprocesseur et une unité de puissance à découpage de courant pouvant délivrer une intensité de 2A à 7A efficace ou plus par phase (3A à 10A crête) sous une tension d'alimentation de 15 à 100V suivant le type de carte.

Chaque module dispose d'une liaison série RS 232C lui permettant de communiquer avec un calculateur ou un automate. Celle-ci permet de définir à distance les paramètres de fonctionnement de chaque module et d'y mémoriser des séquences de mouvement prédéfinies. Grâce à ces séquences, les modules SIMPA peuvent fonctionner de façon autonome et réaliser de véritables petits automates.

Différentes options peuvent faciliter la mise en œuvre des modules SIMPA :

a) une face avant interactive

La face avant permet de visualiser sur un afficheur à 8 chiffres et de modifier au moyen de boutons poussoirs, les paramètres principaux du module.

Elle permet de réaliser des mouvements simples et d'exécuter les séquences prédéfinies, de connaître l'état d'entrées logiques.

b) un sélecteur de séquence

Le sélecteur de séquence, autorise une exploitation autonome du module sans liaison série en permettant de lancer n'importe quelle séquence pré-programmée. Le numéro de séquence à exécuter est défini grâce à deux roues codeuses.

c) des systèmes multiaxes

Les cartes SIMPA peuvent s'intégrer dans des châssis prévus à cet effet et dialoguer avec un calculateur au moyen d'une liaison série unique, réalisant ainsi un système de commande multi-axes.

d) une famille étendue

Différentes cartes fond de panier permettent une interconnexion simple entre une carte SIMPA (indexeur seul) et les cartes amplificateurs telles que MI452, MI454, MI904, MI907, MIP806, MIP909..., augmentant d'autant la famille des modules SIMPA.

I.1 - Mise en œuvre

La mise en œuvre des modules SIMPA peut se faire :

- à distance, grâce à une liaison série au standard RS232, unique pour un maximum de 64 modules. Les logiciels PCSIM et LIBSIM, implantables sur PC compatibles DOS, ont été développés pour faciliter le dialogue avec les modules SIMPA..
  Une version Windows de PCSIM ainsi que la DLL de gestion du protocole de dialogue avec les modules peuvent être livrées sur demande

I.2 - Modes de contrôle

Deux modes de contrôle des modules sont possibles :

- un mode direct accessible soit par l'intermédiaire de la face avant (accès local), soit par la ligne série. Il permet de réaliser les principaux mouvements de base.

- un mode séquence uniquement accessible à partir de la ligne série permet, après chargement des séquences, un fonctionnement autonome des modules. La synchronisation de plusieurs axes peut être réalisée grâce aux 8 entrées et 8 sorties logiques dont dispose chaque module pour communiquer avec l'environnement externe (automates, butées, capteurs...).

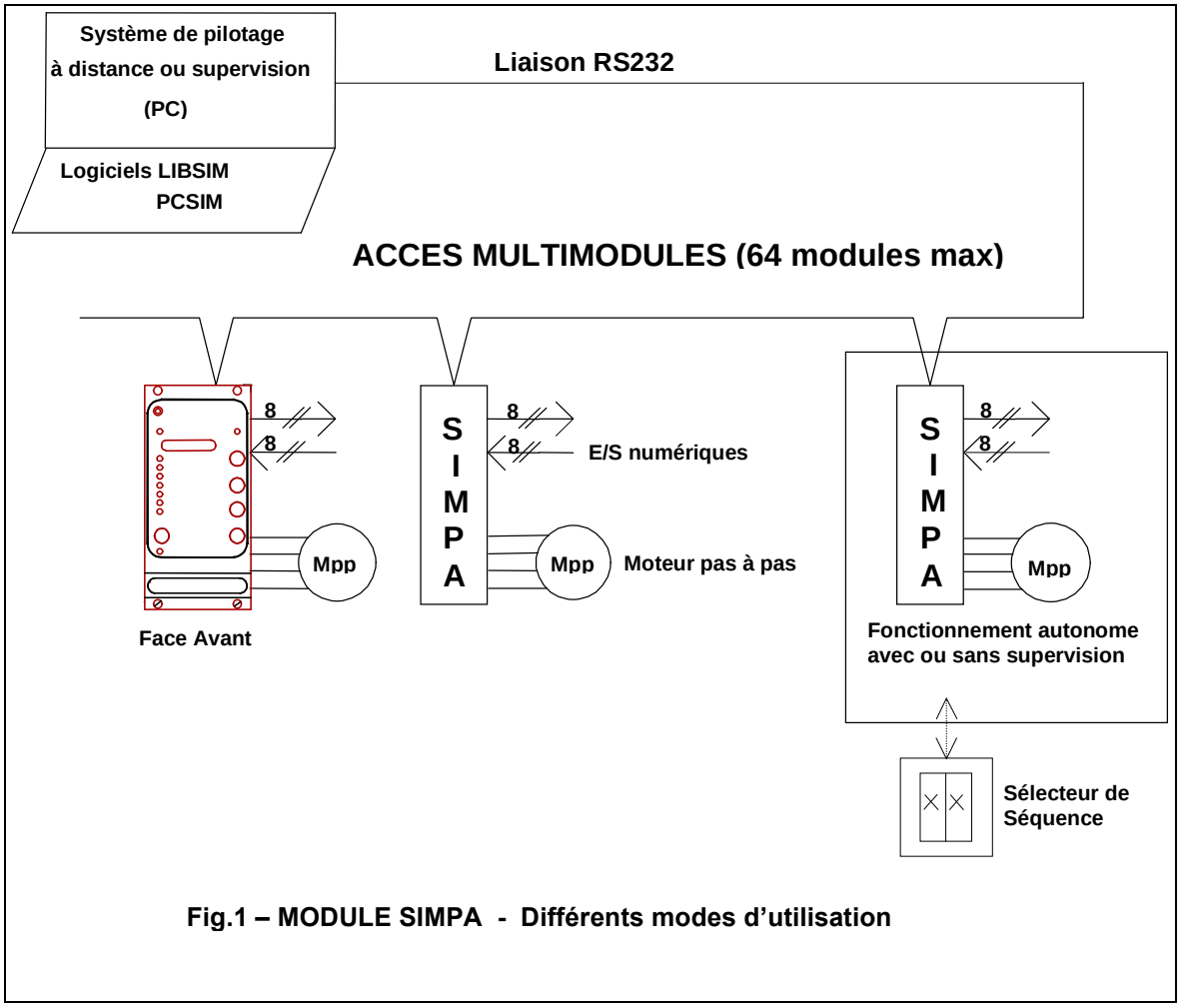


Fig.1 – MODULE SIMPA - Différents modes d'utilisation

I.3 - Fonctions

En mode direct, les principales fonctions accessibles sont :

- la définition des paramètres du mouvement à effectuer :
  * vitesse de démarrage (V_{min}) en pas/seconde (ou demi-pas/s),
  * vitesse de consigne ou palier (V_{max}) en pas/seconde (ou demi-pas/s),
  * durée des rampes d'accélération et de décélération en ms,
  * nombre de pas ou micropas à effectuer et le sens de rotation du moteur;
  * amplitude du courant dans le moteur.
- la mise sous tension du moteur et l'exécution des mouvements;
- le choix du fonctionnement avec ou sans butées ;
- le contrôle du fonctionnement et de l'état des modules.

En mode séquence, certaines fonctionnalités supplémentaires sont offertes :

- enchaînement de plusieurs mouvements,
- temporisation,
- gestion en temps réel de 8 sorties logiques,
- modification du déroulement de la séquence en temps réel en fonction de l'état de 8 entrées logiques,
- appel de sous-séquence, enchaînement de séquence, démarrage conditionnel ou non de séquences.

II - PRINCIPES DE LA COMMANDE DES MOTEURS PAS A PAS

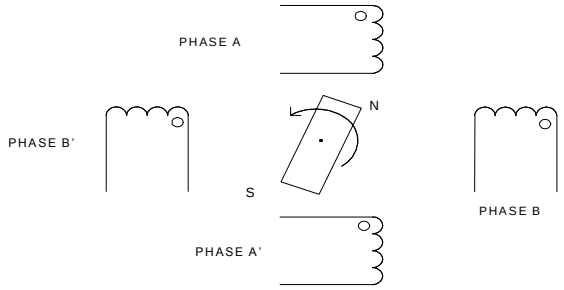
Le présent chapitre a pour but de rappeler les principes fondamentaux utilisés pour la commande des moteurs pas à pas.

Il n'a pas la prétention de constituer un cours sur le sujet, ni d'aborder l'ensemble des sujets relatifs au fonctionnement des moteurs.

Par contre, il montre simplement les principales règles à respecter pour obtenir les performances désirées.

II.1 - Les types de moteur

Le moteur pas à pas conçu pour développer un couple important à faible vitesse (voire à l'arrêt) est un transmetteur d'informations capable de contrôler position et vitesse par simple pilotage d'une fréquence.

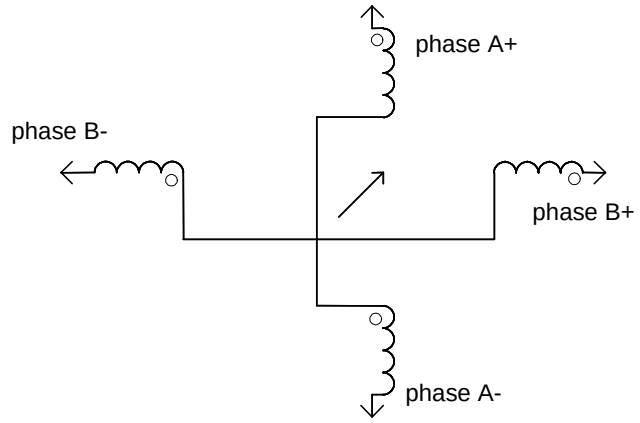


Le moteur est constitué d'un stator comportant un certain nombre de bobines et d'un rotor aimanté ou non.

Lorsque le rotor est passif (non aimanté) le moteur est dit à "réluctance variable". Lorsqu'il est aimanté, le moteur est dit à "aimant permanent". La combinaison des deux types conduit au moteur "hybride".

Le nombre de bobines du moteur ainsi que son type de branchement (4 fils, 6 fils, 8 fils) déterminent un nombre de phases qui, alimentées suivant un séquencement donné, créent un champ statorique tournant.

Par exemple : Branchement 4 fils, bobines en série :



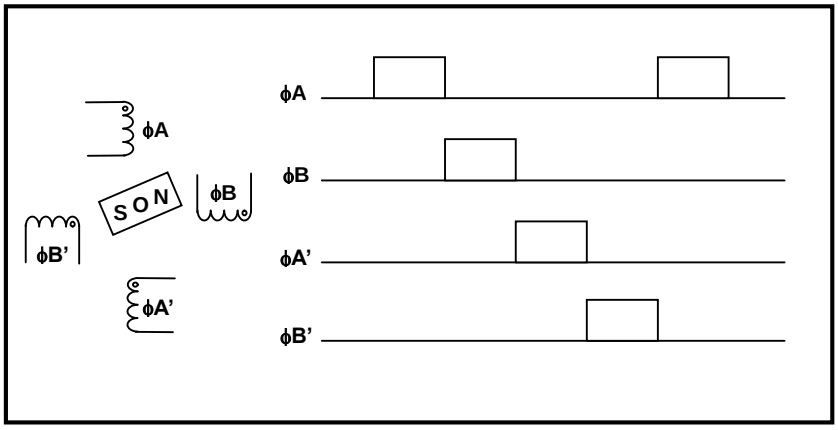
Le rotor s'oriente dans ce champ grâce au couple développé entre les deux champs (rotor aimanté) ou par le couple destiné à diminuer la réluctance (rotor passif).

Le moteur pas à pas est donc un moteur synchrone qui consomme une puissance électrique due au déphasage entre champ magnétique rotorique et statorique. Celui-ci est d'autant plus élevé que la charge mécanique augmente (charge utile + frottements). A cette puissance électrique doit être ajoutée la puissance consommée par effets joules. Au delà d'un déphasage électrique de $\pi/2$, le moteur décroche.

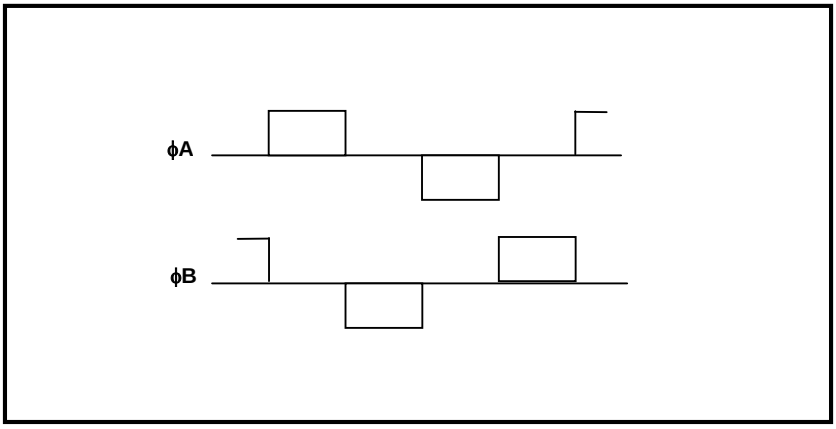
II.2 - Les modes de commutation

II.2.1 - Commutation phase par phase (1 phase ON)

En alimentant chaque phase du moteur l'une après l'autre, le rotor vient se placer successivement devant chaque bobine de phase.

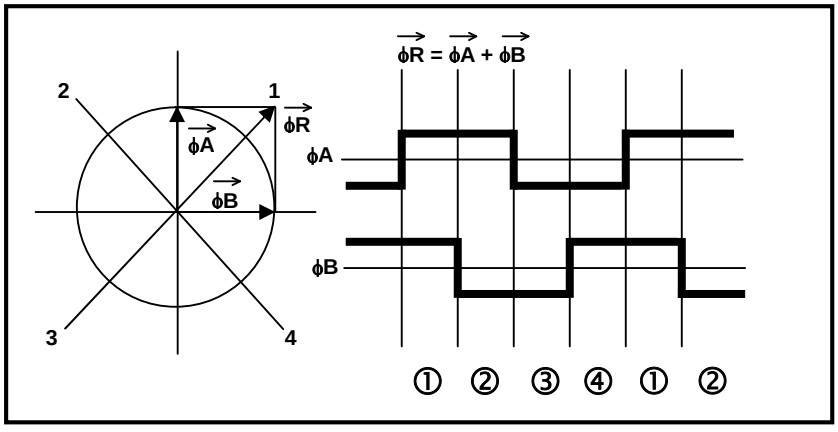


En plaçant les phases associées en parallèle ou en série et en jouant sur le sens du courant injecté dans les bobines, on obtient la commande bipolaire selon le chronogramme suivant :



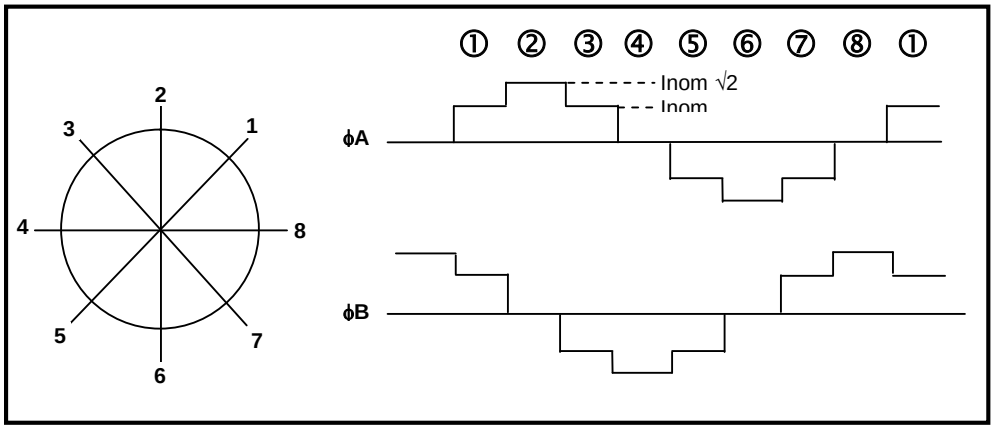
II.2.2 - Commutation 2 phases à la fois (2 phases ON)

En injectant simultanément le courant dans les 2 phases, le moteur s'aligne sur le champ résultant.



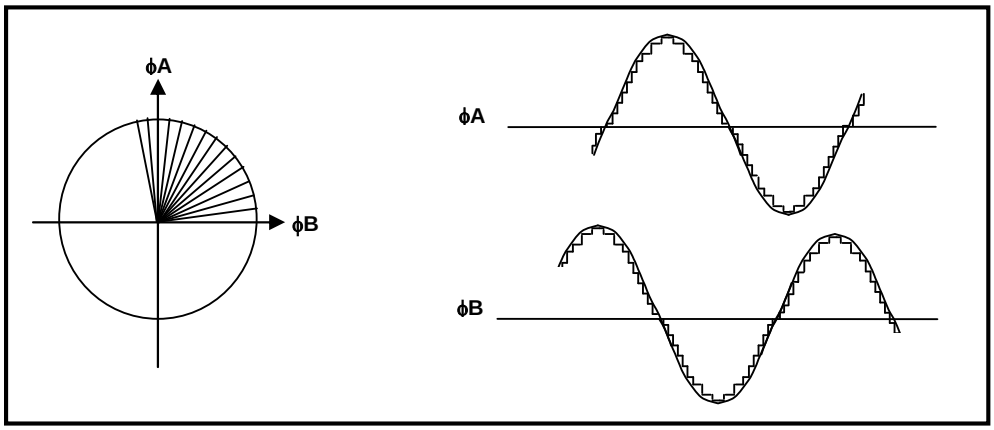
II.2.3 - Commutation en mode ½ pas

En combinant les 2 modes précédents, on obtient 8 positions angulaires par cycle. En modulant l'amplitude du courant par √2, on maintient l'amplitude du couple résultant constant.



II.2.4 - Commutation en mode micropas

En prolongeant le raisonnement précédant, on peut imaginer de créer autant de pas intermédiaires que désiré entre 2 pas "géométriques" du moteur. Il suffit que les champs ØA et ØB (et par conséquence les courants qui les produisent) soient respectivement de forme sinusoïdale et cosinusoïdale.

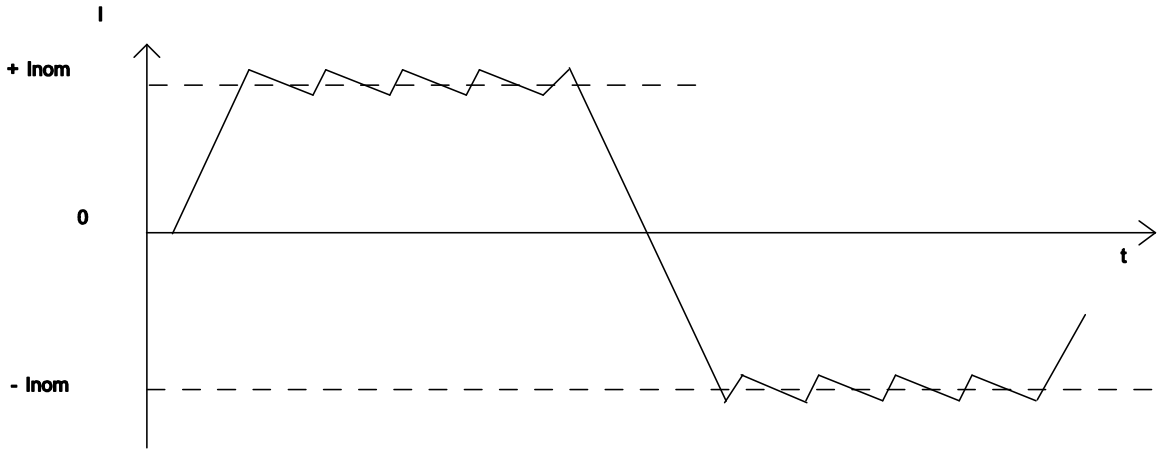


Le mouvement ainsi obtenu est d'autant plus régulier que le nombre de micropas est élevé.

II.3 - Asservissement du courant dans les bobines moteur

L'asservissement du courant dans chaque phase du moteur est obtenu par découpage de la tension d'alimentation aux bornes de l'inductance présentée par le bobinage.
La variation du courant est régie par la loi :

$$\supseteq V = L * \frac{dI}{dT}$$

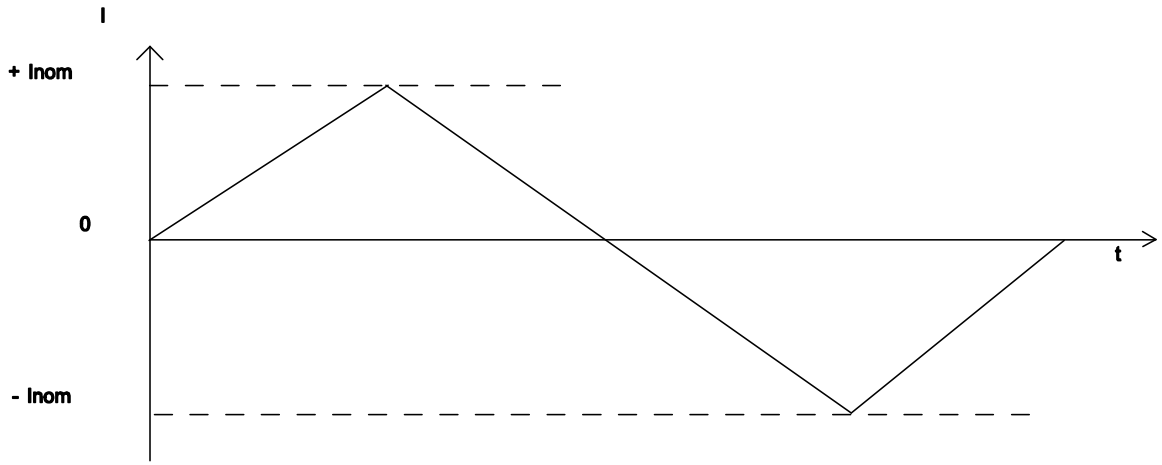


En imposant alternativement deux tensions V1 et –v2 aux bornes du bobinage, on peut maintenir le courant moyen à la valeur désirée simplement en contrôlant le rapport cyclique.

Pour un moteur donné le choix de I_{nom} détermine le couple maximal que peut fournir le moteur suivant la loi des ampères tour appliquée au bobinage.

$$\not\subset C = k_1 * I_{nom} * N = nombre\ de\ spires\ du\ bobinage$$

Le couple moteur étant proportionnel au courant dans les bobinages, ce dernier doit bénéficier d'un temps suffisant pour s'établir (v = Ldi/dt).
Dès que l'on n'atteint plus les paliers de régulation de courant, le couple que peut fournir le moteur diminue rapidement. La vitesse maximum qui pourra donc être obtenue correspond en première approximation à la vitesse pour laquelle la forme d'onde de courant devient triangulaire.



II.4 - Le choix d'un moteur

Connaissant le couple minimum nécessaire à l'application et la tension maximale de fonctionnement étant imposée, le choix du moteur découle des considérations suivantes :

Pour des moteurs possédant des performances de couple identiques mais dont les caractéristiques électriques diffèrent, on constate que les pertes Joules liées au moteur ($R * I_{nom}^2$) sont sensiblement constantes d'un moteur à l'autre alors que les pertes Joules liées à l'électronique de commande augmentent proportionnellement au courant délivré. Il convient donc de choisir le modèle de moteur qui nécessite le courant nominal le plus faible possible.

Pour cela, il convient de considérer que pour un courant variant de $-I_{nom}$ à $+I_{nom}$ (alimentation 2 phases on) à la vitesse maximum envisagée, la relation $\supset$ devient :

$$\subset \quad V_A \geq 2 * \frac{L * I_{nom}}{\Delta T} \quad \text{avec } \Delta T = \text{durée de 2 pas moteur à la vitesse maximum désirée Wmax}$$

$$\Delta T = \frac{2}{W\,max}$$

$$V_A \geq L * I_{nom} * W\,max$$

D'autre part, l'inductance d'une bobine est proportionnelle au carré du nombre de spires :

$$\subseteq L = k_2 * N^2$$

de $\not\subset, \subset, \subseteq$, il vient :

$$\in I_{nom} \geq \frac{k * W\ \max}{V_A} \quad \text{avec} \quad k = k_2 * \left(\frac{(C)}{k_1}\right)^2 = L_0 I_0{}^2$$

L_0 et I_0 self et courant nominal d'un des moteurs du type retenu.

Ainsi, à couple, tension et vitesse maximale de fonctionnement imposés, il convient de choisir le modèle de moteur qui a le courant nominal le plus faible possible et qui vérifie l'équation $\in$

Attention ! la démonstration ci-dessus est volontairement simplifiée pour mettre en relief l'importance du choix du moteur. Le résultat obtenu ne peut donner qu'une première idée du moteur à retenir.

III - FONCTIONNALITES

III.1 - Organisation des modules SIMPA

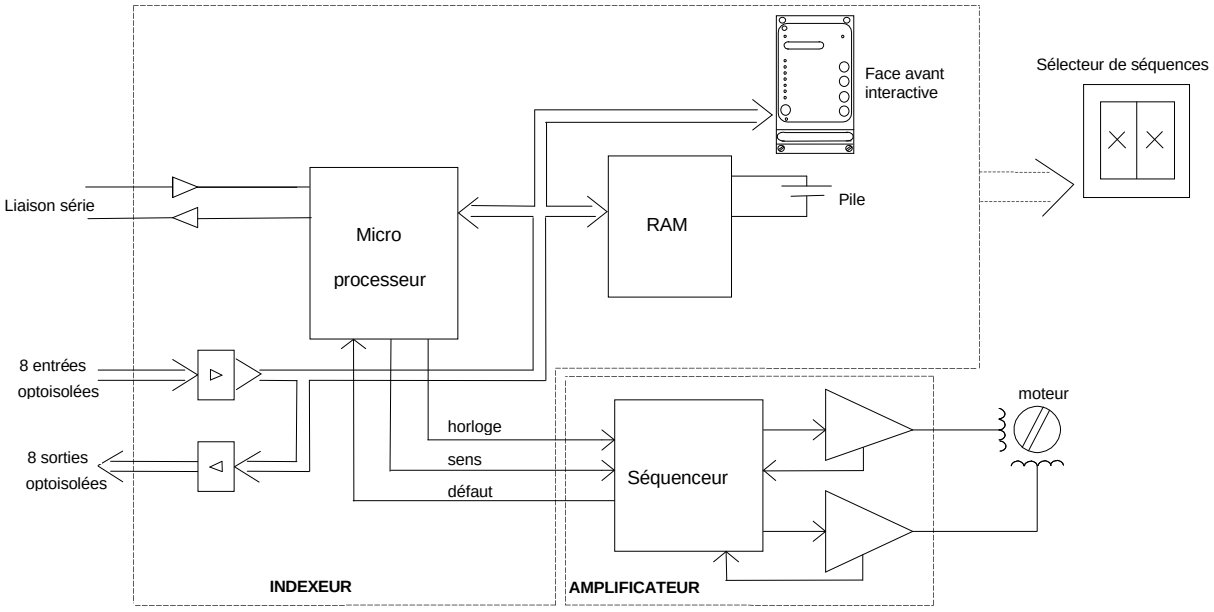
Les modules sont organisés pour :

- contrôler les mouvements d'un axe moteur pas à pas
- opérer de manière autonome,
- inter-réagir entre eux pour constituer des ensembles multi-axes,
- accepter une supervision extérieure et éventuellement être intégrés dans un système plus global,
- accepter un contrôle local directement par la face avant

Pour cela chaque module comprend :

- une unité à microprocesseur et une mémoire pour le stockage :
  * des paramètres définissant les mouvements,
  * des séquences pour fonctionnement autonome,
- les étages de puissance pour connexion directe sur les moteurs en 4, 6 ou 8 fils pour les modules intégrant l'amplificateur de puissance ;
- des entrées/sorties logiques pour communiquer avec l'environnement (butées, synchronisations, status et commandes diverses) ;
- une interface série pour accès à distance, commande et supervision multi-module.
- une interface pour accès local : face avant, sélecteur de séquence ...,

Synoptique d'un module SIMPA



III.2 - Indexeur, indexeur amplificateur

La famille des modules SIMPA comprend aussi bien des modules intégrant l'électronique de puissance (indexeurs amplificateurs) que des modules sans puissance (indexeurs seuls).

Ces derniers peuvent piloter à peu près tous les amplificateurs du commerce et particulièrement les modules MI et MIP développés par la société MIDI INGENIERIE.

Les modules MI sont des amplis micropas, les modules MIP des amplificateurs pas entier ½ pas. Les solutions indexeurs et amplificateurs séparés permettent des combinaisons de performances souvent mieux adaptées à un besoin particulier. Elles permettent aussi une localisation séparée entre la partie contrôle basse puissance (indexeur) et la partie puissance moteur (amplificateur). Un certain nombre de couples constituent des produits à part entière de la famille SIMPA.

ATTENTION ! Si la plupart des commandes définies plus loin sont utilisables sur tous les modules indexeurs amplificateurs, l'utilisation de certaines fonctions dépend de l'interconnexion entre l'indexeur SIMPA et l'ampli qui lui est associé dans le cas de 2 produits indépendants. Il convient notamment de faire particulièrement attention à la gestion du "standby" et des mises sous tension ou hors tension du moteur.

III.3 - Compteur de pas absolu

Chaque module SIMPA dispose d'un compteur absolu du nombre de pas ou micropas effectués. Suivant le sens de rotation, ce compteur compte ou décompte le nombre de pas (ou micropas en mode micropas) générés. Le compteur de pas absolu autorise des retours à la position d'origine 0 à partir de n'importe quelle position. Il peut être remis à zéro à n'importe quel instant (sous réserve que le moteur soit arrêté).

La dynamique du compteur de pas absolu est :

Version de base $-8\,388\,608 \leq CPa \leq +8\,388\,607$ (modulo 2^{24})
Version micropas $-2\,147\,483\,647 \leq CPa \leq +2\,147\,483\,647$ (modulo 2^{32})

III.4 - Paramètres de fonctionnement

III.4.1 - Paramètres définissant les mouvements moteur

Chaque mouvement est défini par cinq paramètres :

- la vitesse de démarrage : V_{min} de 20 à 20 000 pas/s (ou ½ pas/s) définit la vitesse initiale de tous les mouvements.
  Compte tenu des inerties, la vitesse du moteur ne peut prendre instantanément n'importe quelle valeur, il convient de définir une vitesse à partir de laquelle le mouvement du moteur est accéléré jusqu'à la vitesse désirée.

- la vitesse de palier (V_{max}) de 20 à 20 000 pas/s (ou ½ pas/s) définit la vitesse nominale d'exécution des mouvements;

- les durées (Ta et Td) des rampes d'accélération et décélération permettant de passer de V_{min} à V_{max} ou inversement (de 1 à 65 000 ms) ;

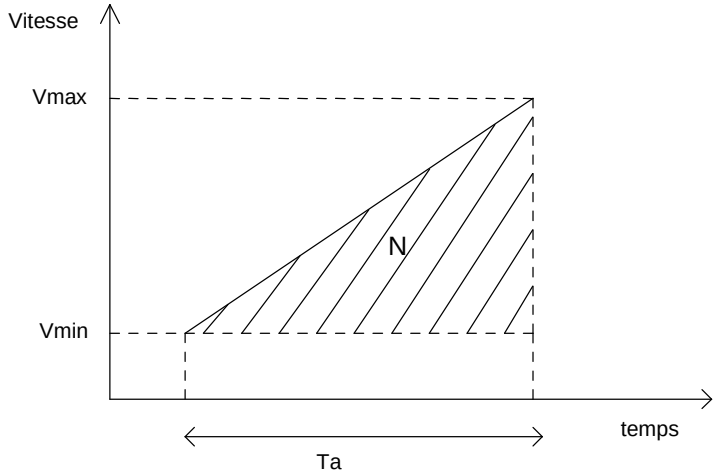
- la résolution μ en micropas par pas pour les cartes fonctionnant en mode micropas;

- le nombre de pas ou de micropas à effectuer N et le sens de rotation du moteur.

III.4.2 - Loi d'accélération/décélération

La plupart du temps, le principal effort à vaincre est lié à l'inertie du système. Une variation de vitesse linéaire (accélération constante) provoque un couple constant de réaction du système en mouvement.
Les modules SIMPA déterminent automatiquement les vitesses à générer durant les phases d'accélération ou de décélération de façon à maintenir l'accélération constante pendant les temps impartis Ta et Td pour passer de Vmin à Vmax ou respectivement de Vmax à Vmin. Ces profils de vitesse sont appelés respectivement loi d'accélération et loi de décélération.
Pour les modules pas entier demi-pas les temps Ta et Td sont obligatoirement égaux au seul temps de rampe définissable Tr.
Pour les modules micropas Ta et Td sont égaux par défaut mais peuvent être différentiés uniquement via la liaison série.

La loi d'accélération théorique étant une loi à accélération constante, elle est représentée par une droite sur un graphe vitesse en fonction du temps (les mêmes explications peuvent s'appliquer à la loi de décélération)..

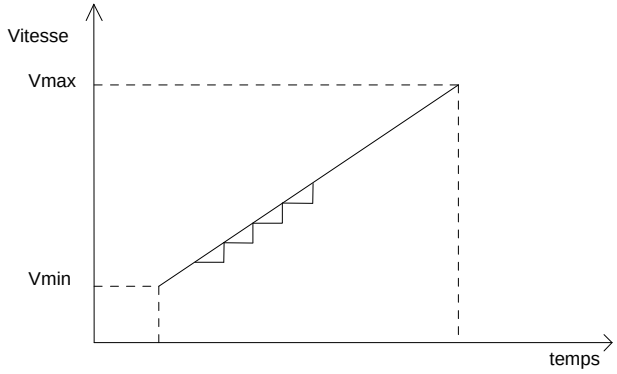


Les paramètres de la loi étant :

V_{min}	= vitesse de démarrage
V_{max}	= vitesse de palier
T_a	= temps de rampe
µ	= résolution en micropas/pas

Cette droite a pour pente l'accélération : $\frac{V_{\max} - V_{\min}}{T_a}$, le nombre total de pas ou micropas effectués pendant cette phase d'accélération est l'intégrale de la vitesse, soit : $N_a = \frac{(V\max + V\min)}{2} * T_a * \mu$

La rampe d'accélération pratique est la décomposition de la rampe théorique en au plus 250 paliers de durées quasi identiques (ceci n'est pas totalement vérifié pour des accélérations rapides et particulièrement sur les paliers à vitesse basse).



La vitesse en un point de la rampe à l'instant ti est donnée par la relation :

$$v_i = \frac{1}{p_i} = \frac{t_i}{T_a}(V_{max} - V_{min}) + V_{min} \quad (en\ p/s)$$

aux erreurs près liées à la discrétisation due aux calculs et à la résolution du générateur de micropas : 0,5 µs en version micropas, 1 µs en version pas entier ½ pas.

Un minimum de 1 pas ou micropas est effectué sur chaque palier, l'algorithme de calcul est tel que le calcul itératif sur le nombre de paliers s'arrête dès que la vitesse sur le palier en cours de calcul atteint la vitesse maximale.

Le calcul de la loi n'est possible que si les paramètres de mouvement respectent un certain nombre de spécifications liées à la taille mémoire réservée au stockage de la loi d'accélération, d'où les conditions imposées au paragraphe III.4.6.

III.4.3 - Mode pas entier, demi pas, micropas : unités

Le logiciel interne aux modules SIMPA a la capacité de gérer les modules selon 3 modes différents :

- mode pas entier,
- mode ½ pas,
- mode micropas.

Par contre l'électronique propre des modules est spécifique et ne peut pas gérer tous les modes.
Selon le type de module retenu, elle gère :

- soit le mode micropas pour les modules micropas,
- soit les modes pas entier ou demi-pas pour les modules pas entier demi-pas car dans ce cas la génération de l'horloge qui contrôle les mouvements est identique.

Nota : Les versions micropas peuvent évidemment fonctionner en pas entier ou en ½ pas en définissant respectivement une résolution de 1 ou 2 micropas.

1) Définition de la résolution

- Sur les modules de pas entier / ½ pas un cavalier ou un microcommutateur permet en général de choisir la résolution désirée entre les 2 seules possibles.

- Pour les modules micropas, la résolution en micropas par pas doit être définie par l'utilisateur et correspondre à la résolution sélectée sur l'ampli de puissance si celui-ci est indépendant de l'unité logique SIMPA (y compris éventuellement 1 micropas par pas ou 2 micropas par pas.

2) Définition des vitesses

- en mode pas entier les fréquences sont définies en pas par seconde (p/s)

- en mode demi-pas : en demi-pas par seconde (½ p/s) il y a donc un rapport 2 dans la vitesse finale du moteur pour les mêmes valeurs numériques saisies au regard du fonctionnement en pas entier.

- en mode micropas : les vitesses sont toujours définies en pas entier, de sorte qu'une modification de la résolution n'entraîne aucune modification des vitesses du moteur (sous réserve que les résolutions du module SIMPA et de l'ampli de puissance, si celui-ci est externe, soient identiques).

Remarque : Les valeurs retournées par le module correspondent aux vitesses réellement générées.
Elles peuvent sensiblement différer des valeurs programmées, compte tenu de la résolution en temps du module SIMPA : 1 µs version pas entier, 0,5 µs version micropas.

3) Définition des déplacements

Les déplacements sont toujours définis dans la résolution du module tel qu'il est utilisé :

- en pas entier pour les modules en mode pas entier,

- en ½ pas pour les modules en mode ½ pas,

- en micropas pour les modules en mode micropas.

Par exemple il faudra définir un déplacement de valeur 13 pour obtenir un mouvement de 1 pas entier avec un module micropas dont la résolution est de 13 micropas par pas.

III.4.4 - Contrôle du courant moteur

Le courant moteur maximum que peut délivrer un module est un des éléments qui distingue les différents modules de la famille SIMPA.

Le courant réellement appliqué au moteur peut cependant être modulé, indépendamment des configurations matérielles :

- par programmation du paramètre courant moteur pour les modules SIMPA qui offrent cette possibilité (voir le Manuel utilisateur du module utilisé);

- par passage en mode "standby" (ou repos): forçage, en dehors des mouvements, d'un courant de maintien plus faible que le courant nominal choisi.

- par passage en mode « Boost » (surcourant) : forçage d'un courant moteur plus élevé lors des phases d'accélération et décélération du moteur. Cette fonction n'est pas disponible sur tous les modules SIMPA.

* En version pas entier ½ pas

Le courant de standby est géré au moyen de la sortie logique 1.
L'état de repos (1 logique) correspond au courant de repos (standby). La gestion est automatique uniquement pour les mouvements directs. En séquence, l'utilisateur doit gérer la sortie logique 1 en fonction du courant qu'il désire appliquer au moteur. Le mode « boost » n'est jamais accessible.

* En version micropas

La gestion des courants standby et boost est totalement transparente et indépendante de la gestion des sorties logiques, y compris en séquence.
Cependant, l'utilisateur peut interdire séparément les modes boost et standby s'il le désire (se référer à la commande MS).

III.4.5 - Glissement toléré

En séquence, le compteur de pas absolu peut à tout instant être comparé à une valeur donnée. Si l'écart est supérieur à une valeur fixée la séquence peut être déroutée vers un traitement particulier.
Cette fonctionnalité, associée à la détection de butées ou de capteurs de position, permet de s'assurer du «non glissement » ou du « non décrochement » du moteur lors des mouvements.

III.4.6 - Limites et cohérence des paramètres de mouvement

III.4.6.1 – Version pas entier ½ pas

III.4.6.1.1 - Limites générales

* $20 \leq Vmin < Vmax \leq 20000$ (p/s ou ½ p/s)
* $\mu = 1$ uniquement (non programmable)
* $1 \leq TR \leq 63535$ ms
  $-999998 \leq Nrelatif \leq +999998$ (pas ou ½ pas)
  $-8388608 \leq Nabsolu \leq +8388607$ (pas ou ½ pas)

III.4.6.1.2 - Cohérence entre paramètres

$V_{max} * Tr < 65\ 10^{6}$

III.4.6.2 - Version micropas

III.4.6.2.1 - Limites générales

* $\frac{62}{\mu} \leq Vmin < Vmax \leq 20000$ (pas/s)

* $1 \leq \mu \leq 256$ pour les modules micropas sans puissance intégrée
(SIMPA micropas ou SIMPA micropas Face Avant,…)

μ : 1, 2, 4, 8, 16, 32 ou 64 pour les modules micropas avec puissance intégrée
(SIMPA micropas 1 axe, SIMPA micropas 1 axe Face Avant,
SIMPA micropas 4 fils ou SIMPA micropas 4 fils Face Avant,…)

* $1 \leq TR \leq 65535$ (ms)

* $-2147483647 \leq N_{relatif} \leq +2147483647$ (micropas)

* $-2147483647 \leq N_{absolu} \leq +2147483647$ (micropas)

III.4.6.2.2 - Cohérence entre paramètres

* $\frac{10^{+3}}{\mu * Tr\,(ms)} \leq V\min\,(p/s) \leq \frac{63{,}75\,10^{6}}{\mu * Tr\,(ms)}$
* $\mu * V\min \leq 20\,000\,(p/s)$
* $\mu * V\max \leq 1{,}28\,10^{6}\,(p/s)$

μ	X	Tr (ms)		
		μ Vmax ≤ 16 KHz	16 KHz < μ Vmax ≤ X	X < μ Vmax
1		$Tr < \frac{63{,}75\,10^{6}}{VMAX}$		
2 . 3	32 KHz	$Tr \leq \frac{63{,}75\,10^{6}}{\mu * Vmax}$	Tr ≤ 3984 ms	$Tr \leq \frac{127{,}50\,10^{6}}{\mu * Vmax}$
4 .. 7	64 KHz			$Tr \leq \frac{255\,10^{6}}{\mu * Vmax}$
8 .. 15	128 KHz			$Tr \leq \frac{510\,10^{6}}{\mu * Vmax}$
16 .. 31	256 KHz			$Tr \leq \frac{1{,}02\,10^{9}}{\mu * Vmax}$
32 .. 63	512 KHz			$Tr \leq \frac{2{,}04\,10^{9}}{\mu * Vmax}$
64 .. 256	1024 KHz			$Tr \leq \frac{4{,}08\,10^{9}}{\mu * Vmax}$

III.5 - Mouvements directs

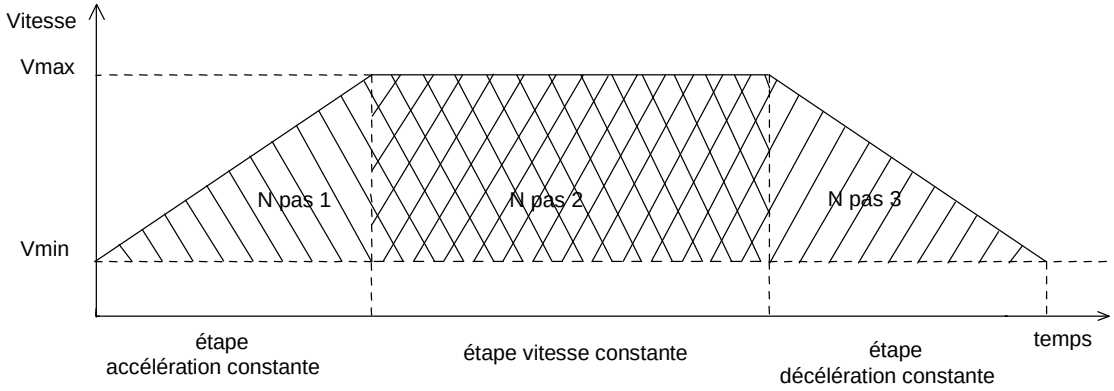
Ce sont des mouvements simples exécutés immédiatement à la réception de la commande.

III.5.1 - Description du mouvement de base

Les mouvements directs se décomposent toujours en trois étapes :

- étape à accélération constante au courant de boost (si le module le permet),
- étape à vitesse constante au courant nominal,
- étape à décélération constante au courant de boost (si le module le permet).

A la fin du mouvement le courant moteur est mis au niveau repos (standby).



N : nombre de pas ou micropas à effectuer.

Le module calcule automatiquement le nombre de pas ou micropas à effectuer au cours de chaque étape du mouvement en fonction des paramètres de mouvement.

N = Npas1 + Npas2 + Npas3

Si le nombre total de pas ou micropas imposé est insuffisant pour que le module effectue les étapes complètes d'accélération et décélération, les étapes d'accélération et de décélération sont tronquées et il n'y a pas de palier à vitesse constante.

Nota : en général Npas3 = Npas1 (sauf pour les modules micropas si les temps d'accélération : Ta et décélération : Td programmés sont différents).

III.5.2 - Types de mouvements

Les modules SIMPA peuvent gérer trois types de mouvements :

- Des mouvements relatifs : la consigne donnée définit le nombre de pas ou micropas à exécuter à partir de la position actuelle du moteur. Le sens de rotation est défini par le signe de la consigne.

- Des mouvements absolus : la consigne définit la nouvelle position à donner au moteur.
Le module effectue automatiquement le calcul du nombre de pas ou micropas à effectuer et en détermine le sens. Un retour à la position origine est directement réalisable sans consigne. Une nouvelle position origine peut être définie à tout instant, soit en lui substituant la position actuelle du moteur (cela revient à mettre à 0 le compteur de pas ou micropas absolus), soit en donnant une nouvelle valeur à la position courante.

- Des mouvements infinis dans un sens ou l'autre : après une phase d'accélération, le moteur reste à la vitesse de palier en attente d'un ordre d'arrêt.
Pour les versions micropas, la vitesse palier d'un mouvement « infini » peut être forcée à une valeur inférieure au paramètre de vitesse maximum défini, et changée en cours de mouvement sans obligation d'arrêt préalable du mouvement.

Tous ces mouvements peuvent être arrêtés par commande avant leur exécution complète, soit instantanément (attention au glissement du moteur), soit avec décélération.

L'activation du mode butée permet de limiter tout mouvement au moyen de 2 commutateurs extérieurs connectés aux entrées logiques E7 et E8 respectivement associés au sens **+** et **-** du mouvement moteur. L'activation d'une butée dans un sens qui la concerne provoque l'arrêt immédiat du moteur.

Remarque : En mode micropas, la phase de décélération peut être suivie d'un certain nombre de micropas effectués à vitesse minimum. Ce nombre ne peut excéder 1 pas entier moteur.

III.6 - Etat du module

- Registre d'état :

Ce registre permet de connaître l'état global du module notamment :

. un passage en phase 255	:	S
. certains défauts d'alimentation	:	W
. défaut hard du module (court-circuit ...)	:	X
. défaut sur les entrées logiques	:	E
. perte de la mémoire	:	M

- Paramètres :

La plupart des paramètres de fonctionnement du module peuvent être vérifiés au travers de la liaison série.

- Séquences :

Les phases des séquences peuvent être relues une à une.

III.7 - Séquences

Afin de réaliser des opérations plus complexes, les modules SIMPA peuvent mémoriser des successions d'actions. Ces suites d'actions élémentaires sont dénommées "séquences". Leur déroulement peut être conditionné à certains événements externes. Ce mode de fonctionnement transforme les modules SIMPA en véritables automates.

III.7.1 - Organisation des séquences

Les séquences sont constituées d'actions élémentaires appelées phases (mouvement relatif, mouvement absolu, attente ...). Chaque séquence est appelée par un numéro défini par l'opérateur lors de sa création. Les phases d'une séquence sont aussi repérées par un numéro qui leur est propre. Ces définitions sont biunivoques.

Un certain nombre de commandes permettent de préparer, sélecter, exécuter et relire les séquences.

Plusieurs séquences peuvent s'enchaîner automatiquement les unes aux autres avec ou sans condition.

III.7.2 - Préparation des séquences

La préparation des séquences s'effectue en deux étapes :

- La création de la séquence en réservant en mémoire la place nécessaire à sa mémorisation : numéro de séquence et nombre de phases prévues,

- l'écriture des phases proprement dites.

La commande d'effacement de séquence permet de libérer la place réservée à une séquence non utilisée.

Bien évidemment, chaque séquence peut être vérifiée en relisant ses phases une à une.
Mais ceci ne constitue pas un "éditeur" en soit. La description d'une séquence devra en général être préparée dans un fichier à l'aide de PCSIM ou de n'importe quel éditeur de texte, puis téléchargée dans le module.

L'écriture des séquences et de leurs phases peut se faire dans n'importe quel ordre, réécrire une phase de même numéro efface le contenu de la phase déjà écrite et la remplace par le nouveau contenu.

III.7.3 - Sélection et exécution des séquences

L'exécution des séquences par les modules SIMPA peut être lancée de plusieurs manières :

- exécution immédiate de la séquence sélectée par le calculateur hôte (éventuellement via PCSIM ou Libsim) ou au moyen de la face avant pour les modules qui en sont équipés ou de l'option sélecteur de séquence

- exécution différée de la séquence choisie à toute nouvelle mise sous tension (ou Reset)

Bien évidemment, l'opérateur est libre d'arrêter toute séquence en cours d'exécution ou d'en supprimer l'exécution automatique.

III.7.4 - Déroulement des séquences

Toute séquence débute à la phase n° 1 et se termine normalement à la dernière phase définie.

Les phases d'une séquence peuvent s'enchaîner, à partir de la phase 1, de deux manières distinctes :

- enchaînement naturel,
- enchaînement conditionnel.

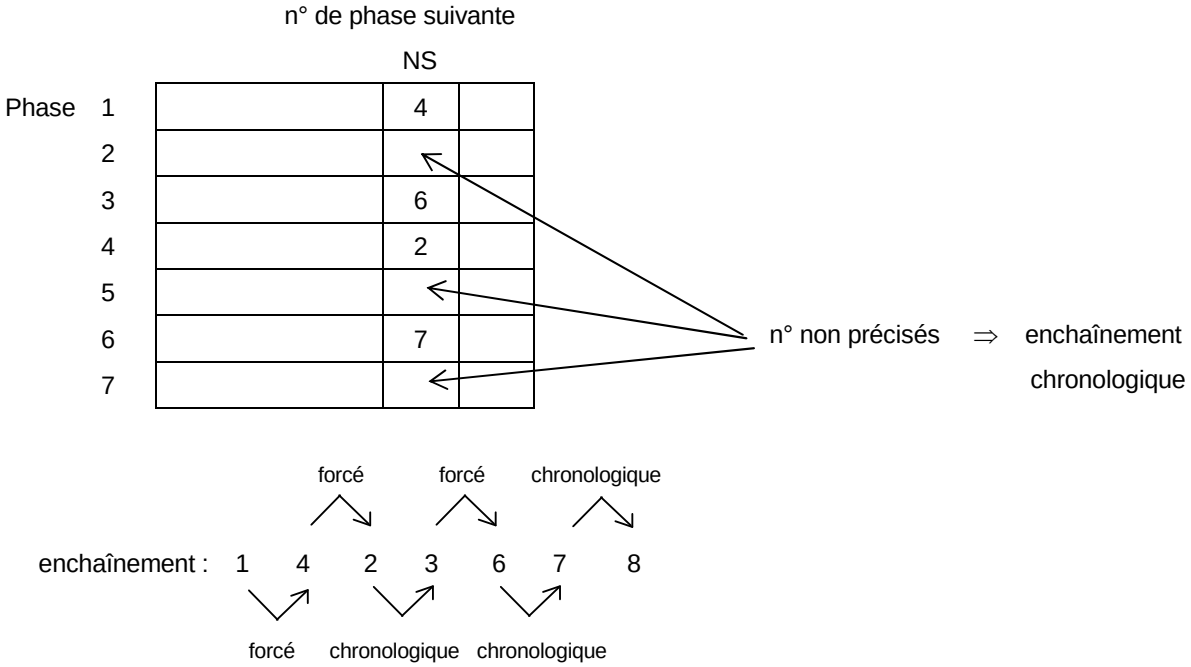
III.7.4.1 - Enchaînement naturel

Si rien de particulier n'est précisé, les phases s'enchaînent chronologiquement depuis la première jusqu'à la dernière dans l'ordre naturel 1, 2, 3.... n.

Cet ordre peut être forcé en spécifiant dans la définition de la phase le numéro de la phase suivante désirée (paramètre NS de la commande de définition des phases).

Si la phase suivante n'est pas définie ou n'existe pas, la séquence se termine sur cette phase lors de l'exécution, même si d'autres phases correctement définies lui succèdent.

Exemple :



III.7.4.2 - Enchaînement conditionnel

Les entrées logiques permettent d'intervenir sur le déroulement chronologique des phases d'une séquence.

Ces changements de phase peuvent intervenir :

- à la fin de la phase en cours,

- pendant le déroulement même de la phase en cours.

a) Enchaînement conditionnel à la fin de la phase en cours (type NE)

Le saut à une phase particulière différente de la phase suivante naturelle est lié à l'état d'une entrée logique particulière. A chacune des 8 entrées peut être associé un numéro de phase. Si plusieurs entrées sont sélectées et actives à la fin de la phase en cours, la phase suivante sera celle associée à la première (dans l'ordre chronologique 1, 2, 3.....8).

Pour une phase donnée, chaque entrée ne peut être associée qu'à une seule phase, mais un même numéro de phase peut être associé à plusieurs entrées permettant de réaliser des combinaisons de type "ou". Les combinaisons de type "et" pourront être mises en place par câblage ou par l'utilisation de phases chaînées conditionnellement (ou par la gestion de polarité des entrées, pour les versions micropas uniquement).

Les enchaînements conditionnels permettent de réaliser très facilement:

- des répétitions d'une ou de plusieurs phases jusqu'à ce qu'une entrée change d'état.

- des exécutions sélectives de phases.

En associant l'enchaînement conditionnel des phases à l'enchaînement des séquences décrit plus loin, on peut également construire des répétitions de séquences et des séquences conditionnelles.

b) Enchaînement conditionnel pendant le déroulement de la phase en cours (type NF)

Cet enchaînement ne diffère du précédent que par le fait que les entrées ne sont plus traitées sur leur état en fin de phase mais au moment de leur activation. La phase en cours est alors interrompue immédiatement et le branchement sur la phase associée à l'entrée qui a basculé est effectué.

Si l'entrée est déjà positionnée au début de l'exécution de la phase, le saut conditionnel a lieu immédiatement sans exécution de la phase (Attention aux sorties logiques, voir plus loin).

c) Entrée inactive, entrée active

Les entrées logiques des modules SIMPA sont soit optoisolées , soit rappelées à 5 V.
L'état inactif des entrées correspond à leur état de repos : absence de courant ou rappel au 5 V. Il est associé à l'état **logique** 1.
Les sauts conditionnels sont donc associés à l'état **logique** 0 correspondant normalement soit à l'activation de l'optocoupleur d'entrée, soit à la mise au 0V d'une entrée non isolée.

* Pour les versions micropas, il est possible de modifier la polarité des entrées logiques.

- **Globalement** avec les commandes de gestion des butées MB ou MN
  Le paramètre **L** par défaut associe l'état actif électrique au niveau **logique** 0
  Le paramètre **H** associe l'état actif électrique au niveau **logique** 1

- **Individuellement** en faisant précéder d'un signe ─ l'adresse de saut associée à l'entrée pour dérouter le cours de la phase sur l'état **logique inactif** (1).

III.7.5 - Phases et séquences réservées

III.7.5.1 - Phase d'arrêt : 54 (254)

L'accès à cette phase provoque l'arrêt de la séquence. La séquence est terminée. Si une séquence chaînée est définie, son exécution est alors lancée.

Dans l'exécution d'une séquence, la rencontre d'une phase non définie provoque automatiquement un branchement sur la phase d'arrêt. L'utilisateur peut utiliser cette phase pour terminer normalement une séquence en la nommant explicitement comme phase suivante dans une phase de la séquence (ou dans plusieurs si des fins de séquences conditionnelles sont souhaitées).

- en version pas entier ½ pas, le n° de phase d'arrêt est 54 et les sorties logiques sont forcées à l'état repos.

- en version micropas, le n° de phase d'arrêt est 254, les sorties logiques sont laissées en l'état.

III.7.5.2 - Phase d'interruption : 55 (255)

L'accès à la phase réservée d'interruption permet de terminer la séquence comme avec une phase d'arrêt et de faire en plus générer par le module un signal d'interruption ("BREAK" sur la ligne série) et d'enchaîner uniquement la séquence n° 99. Si la séquence 99 n'est pas définie, aucune autre séquence n'est lancée.

- en version pas entier ½ pas, le n° de phase d'interruption est 55

- en version micropas, le n° de phase d'interruption est 255

III.7.5.3 – Séquence 99

La fin d'exécution de la phase 255 provoque automatiquement l'enchaînement sur la séquence réservée n° 99.

Cette séquence peut être utilisée pour exécuter des mouvements de mise en sécurité par exemple retour à une position absolue, recherche de butée, dégagement, activation alarme...

Il n'est pas nécessaire de définir cette séquence si elle ne doit pas être utilisée, ou si aucune action n'est souhaitée.

III.7.6 - Enchaînement des séquences et sous séquences

III.7.6.1 - Enchaînement

Il est possible de chaîner plusieurs séquences en indiquant dans chaque séquence le numéro de la séquence suivante souhaitée. Si aucun numéro n'est indiqué, aucune séquence n'est chaînée.

Le numéro de la séquence suivante peut être spécifié dans n'importe quelle phase définie de la séquence (pas forcément la dernière), sous réserve que la phase qui la définit soit réellement exécutée, ce qui peut ne pas être le cas lorsque des phases conditionnelles sont mises en place comme décrit précédemment. Ceci permet aussi de choisir une séquence en fonction d'actions sur les entrées logiques.

Le numéro de séquence suivante est donné par le paramètre optionnel NL de la commande de définition des phases.

Lorsque plusieurs phases donnent des numéros de séquence suivante différents, c'est la séquence pointée par la dernière des phases exécutée avec un ordre NL qui sera chaînée.

Attention ! entre chaque séquence, le module SIMPA repasse par son état de repos : moteur arrêté en position standby et en version pas entier ½ pas les sorties logiques sont forcées au repos. Il ne peut pas y avoir de continuité de mouvement.

III.7.6.2 - Sous séquences

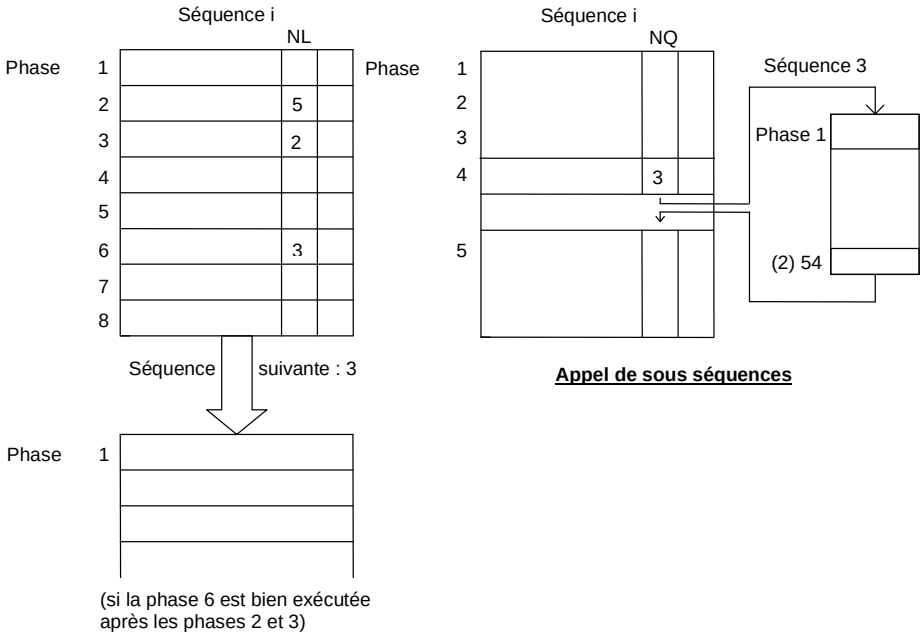
L'exécution d'une séquence peut être momentanément interrompue pour exécuter une sous séquence grâce à la commande NQ. Contrairement au chaînage, le module ne repasse pas par son état de repos, les sorties logiques sont laissées en l'état, le contrôle du moteur est entièrement assuré par les séquences.
Attention cependant aux discontinuités de mouvements possibles dues au temps de transfert entre séquences.

L'exécution de la sous séquence est lancée à la fin de l'exécution de la phase qui l'appelle.

La fin d'exécution de la sous séquence ramène l'exécution à la phase suivant chronologiquement (N° de phase +1) celle qui a appelé la sous séquence. Plusieurs phases d'une même séquence ou de séquences différentes peuvent appeler la même séquence. Une sous séquence ne diffère en rien d'une séquence classique, seule la fin naturelle de cette dernière ramène à la séquence d'appel. Une sous séquence peut parfaitement être exécutée seule comme toute séquence ordinaire sans être appelée par une autre séquence.

Attention ! L'accès à la phase "d'interruption" : (2)55 supprime le chaînage prévu ou le retour à la séquence d'appel puisqu'il chaîne obligatoirement la séquence 99. De plus 1 seul niveau de sous séquence est autorisé : une sous séquence n'a pas le droit d'en appeler une autre !

III.7.6.3 - Exemples



Appel de sous séquences

Enchaînement de séquences

III.7.7 - Description des séquences

Chaque phase d'une séquence est décrite par :

- le numéro de la séquence à laquelle elle se rapporte : ns.
  Toute phase est liée à une (et une seule) séquence.

- un numéro d'ordre de la phase : np.
  La première phase à être exécutée dans une séquence est la phase 1. Ensuite les phases s'enchaînent dans l'ordre chronologique avec possibilité de branchement inconditionnel ou conditionnel.

- la nature de la phase : Na

NA : phase d'accélération
ND : phase de décélération
NV : vitesse constante
NP : mouvement relatif (du même type que le mouvement direct relatif)
NX : mouvement absolu (du même type que le mouvement direct absolu)
NH : retour origine ("HOME")
NZ* : remise à zéro du compteur de pas absolu.
Détermination nouvelle référence pour mouvements absolus
NT* : puissance moteur ON
NU* : puissance moteur OFF
NW : attente
NC : modification consigne de vitesse palier V_{max}
NG : détection glissement
NY : attente avec TIME OUT d'une synchronisation par liaison série commande PG)
PV : programmation d'une variable
PI : modification de la valeur à l'initialisation d'une variable
PR : modification de la valeur courante d'une variable
PC : transfert du compteur de pas absolu dans une variable
PA : addition sur variable
PT : test sur variable

- la consigne : Cns (nombre de pas, position, vitesse palier, délai, n° de paramètre, …
  NV, NP, NX, NC, NW). Pour un déplacement relatif le sens du mouvement est défini par le signe de la consigne.

- et les paramètres suivants facultatifs :

NL-NQ : le numéro de séquence à exécuter en fin de la séquence en cours : NL sc, ou le numéro de la sous séquence à appeler en fin d'exécution de la phase : NQ ss

NE-NF : les correspondances entrées logiques <--> phases suivantes permettant de définir les branchements conditionnels sur état : NE ou sur transition : NF et les 8 adresses de saut X1 à X8 associés aux 8 entrées (valeur 0 pour les entrées non utilisées).

NS : le numéro : ns de la phase suivante à exécuter si aucun branchement conditionnel n'intervient. Par défaut, la phase suivante est la phase chronologique suivante (numéro phase en cours + 1).

NO : l'état des sorties logiques : out, au cours de la phase. Par défaut, l'état des sorties reste identique à ce qu'il était dans la phase précédemment exécutée. A la fin de la séquence les sorties sont mises au repos (état logique "1"), uniquement en version pas entier ½.

D'où la structure de la commande :

SP ns np Na [Cns] [NL sc] [NE X1 ... X8] [Ns ps] [NO out]
[NQ sq] [NF]

Nota : la mise entre crochet précise la nature optionnelle du paramètre.

Particularité de la version pas entier ½ pas :

- le sens du mouvement doit être défini par un paramètre indépendant mais obligatoire : S+ ou S- suivant le sens désiré d'un mouvement relatif (pour les natures n'impliquant pas une définition de sens, S+ ou S- peuvent être utilisés indifféremment).

- une valeur de consigne est obligatoire (par exemple 0), même pour les natures de phase : NH, NZ, NT et NU.

D'où la structure de commande :

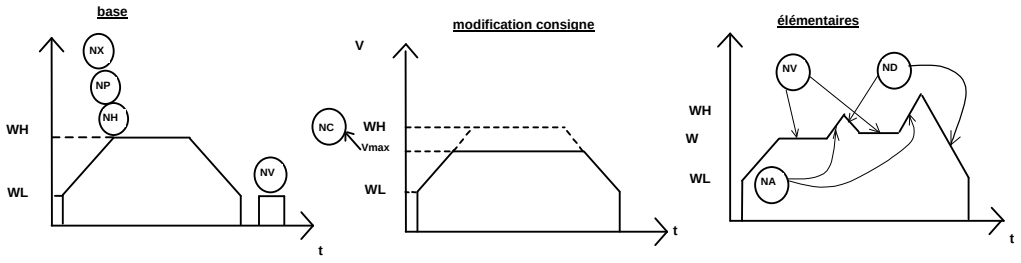
SP ns np S+ Na Cns [NL sc] [NE X1 ... X8] [Ns ps] [NO out]
S- [NQ sq] [NF]

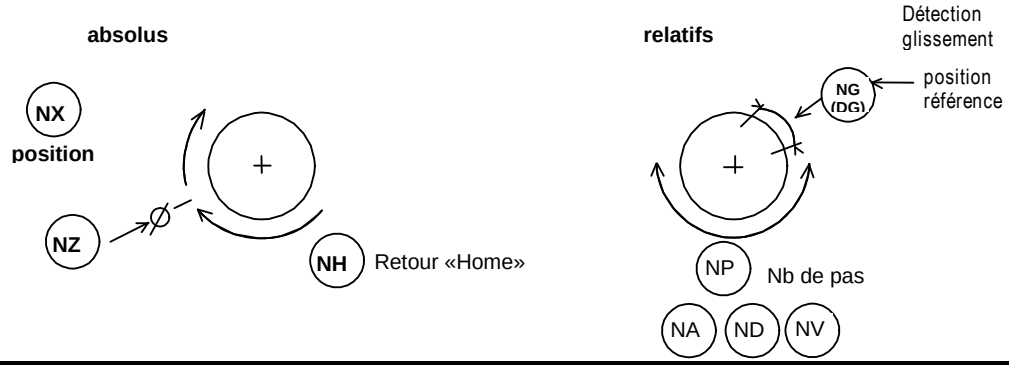
III.7.8 - Limites de fonctionnalité des séquences

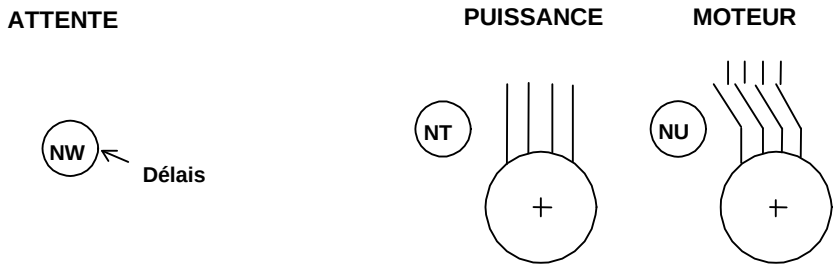
	Version pas entier ½ pas	Version micropas
Numéro de séquence	$1 \leq ns \leq 99$	$1 \leq ns \leq 120$
Nombre total de phases disponibles	413	384
N° de phase dans une séquence	$1 \leq np \leq 50$	$1 \leq np \leq 200$
Nombre de variables	aucune	32
Consigne déplacement relatif	$-999998 \leq dr \leq +999998$	$-2147483648 \leq dr \leq +2147483647$
absolu	$-8388606 \leq da \leq +8388606$	$-2147483648 \leq da \leq +2147483647$
temporisation	$1 \leq W \leq 65535$ ms	

III.7.9 - Résumé des natures de phases

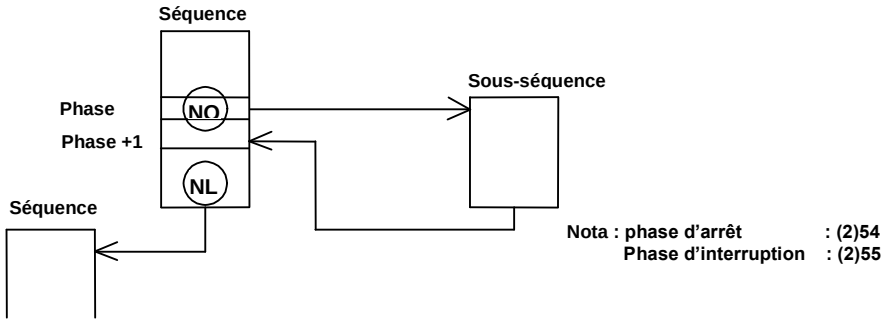
DEPLACEMENTS







ENCHAINEMENT DES SEQUENCES



III.8 - Utilisation des entrées logiques

III.8.1 - Généralités

Le fonctionnement en mode séquence du module SIMPA permet la gestion des 8 entrées logiques.

Ainsi qu'il a été présenté dans les paragraphes précédents, les entrées logiques permettent d'intervenir sur le déroulement chronologique des phases d'une séquence, soit de manière immédiate (transition), soit en fin de phase (niveau).

L'état actif normal d'une entrée est le niveau logique 0.

Pour les versions micropas, la polarité des entrées peut être modifiée globalement grâce aux commandes MB et MN.

Un câblage judicieux des entrées/sorties de différents modules permet une synchronisation des mouvements des différents axes (câblage à défaut par exemple, sur les cartes SIMPA 4 axes).

III.8.2 - Temps de prise en compte d'une entrée logique

Normalement les entrées logiques sont gérées en temps réel à la résolution du pas ou micropas moteur. Cependant, plus la vitesse du moteur est élevée, plus le temps disponible au traitement des entrées logiques diminue, le temps de réponse à ces dernières peut alors prendre plusieurs pas ou micropas moteur.

Compte tenu des remarques précédentes, le temps nécessaire à la prise en compte d'une entrée par le module peut atteindre 18 ms. Tant qu'il n'y a pas de dialogue sur la liaison série, ce temps reste inférieur à 10 ms, il convient de maintenir les entrées dans un état déterminé pour un temps supérieur à 20 ms si l'on ne veut pas risquer de ne pas voir la transition active de l'entrée.

REMARQUE IMPORTANTE : si ce temps de prise en compte s'applique bien à la gestion des entrées sur transition, il convient de faire attention au maintien de l'état d'une entrée, pendant au moins la durée de la phase, si l'on désire gérer les sauts conditionnels sur état en fin de phase.

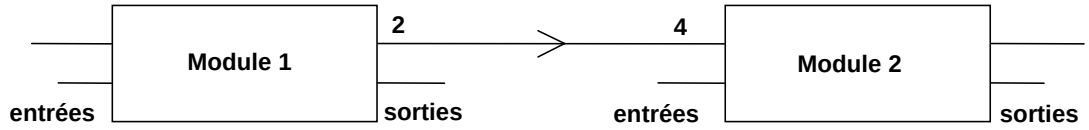
Exemple :

Soit un mouvement de 20 pas décrit à vitesse constante de 200 p/s, celui-ci dure donc $\frac{20}{200} = 0{,}1 \text{ seconde}$

Dans ce cas il convient de maintenir une entrée à l'état actif durant au moins 100 ms si on désire qu'elle soit prise en compte de manière certaine en fin de phase.

III.8.3 - Synchronisation de modules

En reliant la sortie d'un module à l'entrée d'un autre module, il est possible de lier l'action du second module à un événement du premier module.



Exemple :

Phase de génération de synchro du module 1 :

SP ns_1 np_1 + NW $_{t1}$ NO FD

Phase d'attente de synchro du module 2 :

SP ns_2 np_2 + NW t_2 NE 00 00 00 05 00 00 00 00 NS np_2

Le module 2 boucle sur la phase np2 en attendant l'activation de son entrée 4 pour exécuter la phase 5.
Le module 1 active sa sortie 2 pour un temps t_1 qui doit être supérieur à la durée t_2 de la phase d'attente du module 2.

III.8.4 - Défaut sur les entrées logiques

Une trop grande activité des entrées logiques due par exemple à des parasites, pourrait détruire les performances du module SIMPA (notamment au niveau de la liaison série). Aussi, au delà d'une certaine fréquence (≈ 10 KHz) les entrées logiques ne sont plus prises en compte, le moteur est remis au repos et toute séquence en cours interrompue, l'état du module retourne E et l'état des entrées est alors matérialisé par XX.

Seul un reset ou une remise sous tension du module permet de revenir au fonctionnement normal.

III.9 – Gestion des sorties logiques

III.9.1 – Généralités

Les modules SIMPA disposent de 8 sorties logiques indépendantes et totalement gérables par l'utilisateur. Ces sorties peuvent être contrôlées via la liaison série grâce à la commande « GL » ou dans les séquences grâce à la directive NO. Ces actions sont en général réalisées immédiatement (ou en début de phase dans les séquences).
Pour les modules SIMPA micropas il est cependant possible de conditionner la modification des sorties logiques au passage à une valeur définie du compteur de pas absolu (CPA) grâce à la commande GP.

III.9.2 – Gestion immédiate

Les nouveaux états des 8 sorties logiques sont définis par un octet en général donné sous la forme de 2 caractères hexadécimaux (0…9, A …F).
Pour les modules micropas, l'indépendance des 8 sorties est obtenue par l'utilisation d'un masque de 8 bits (2 caractères hexadécimaux) permettant de définir les seules sorties à modifier 0 : sortie logique non modifiable
1 : sortie logique à modifier

Les commandes ou directives s'écrivent :

GL OUT	ou	NO OUT	pour tous les modules
GL OUT [: MSQ]	ou	NO OUT [: MSQ]	pour les versions micropas uniquement

OUT est l'état désiré des 8 sorties et MSQ, le masque éventuel à appliquer. Le nouvel état des sorties logiques après exécution est :

$$S_t = (OUT * MSQ) + (S_{t-1} * \overline{MSQ})$$

Lorsque le masque n'est pas précisé (GL OUT ou NO OUT) toutes les sorties sont modifiées comme pour MSQ = 0FFh

III.9.3 – Gestion différée : uniquement pour les modules micropas

La gestion différée est mise en œuvre par la commande :
GL OUT : MSQ typ

Dans ce cas, la commande GL n'est pas exécutée immédiatement mais lors du passage du compteur de pas absolu à la valeur définie par la commande GP.

III.9.3.1 – Absolue, relative

Deux types de fonctionnement sont possibles selon le type (typ) donné dans la commande GL :

- Absolu : la position de modification est donnée directement dans la commande GP

- Relatif : la position de modification est relative à la position courante du compteur de pas absolu au moment de l'armement de la fonction par la commande GP. La position de modification est donnée par la valeur définie par GP additionnée à la valeur courante du compteur de pas absolu.

Dans tous les cas, il est possible de choisir non seulement la position d'activation mais aussi de conditionner l'activation au sens du mouvement en faisant précéder le type du signe correspondant au sens désiré. Si aucun signe n'est précisé, le sens est indifférent.

III.9.3.2 – Génération d'impulsion

La commande GP permet de définir une « largeur d'impulsion » en définissant une deuxième position (de même type) pour laquelle les sorties logiques seront complémentées (toujours au travers du masque si nécessaire).

En utilisant conjointement les commandes : GL OUT : MSQ typ et GL POS : DEL

On obtient au passage à la position du CPA définie par POS $S_t = (OUT * MSQ) + (S_{t-1} * \overline{MSQ})$

puis à la position définie par DEL $S_{t'} = (\overline{OUT} * MSQ) + (S_{t'-1} * \overline{MSQ})$

En fonctionnement de type absolu :

t est défini par $CPA_t = POS$ et
- $CPA_{t-1} < CPA_t$ si typ = +A
- $CPA_{t-1} > CPA_t$ si typ = - A
- ou typ = A

et t' défini de même par $CPA_{t'} = DEL$ et
- $CPA_{t'-1} < CPA_t$ si typ = +A
- $CPA_{t'-1} > CPA_t$ si typ = - A
- ou typ = A

En fonctionnement relatif, si la position du compteur absolu au moment de la commande GP est CPA_o

t est défini par $CPA_t = POS + CPA_o$ et
- $CPA_{t-1} < CPA_t$ si typ = +R
- $CPA_{t-1} > CPA_t$ si typ = -R
- ou typ = R

et t' défini de même par $CPA_{t'} = DEL + CPA_t$

Attention ! Le sens de transition du compteur de pas absolu pris en compte n'est pas lié au sens du mouvement lors de la commande GP notamment pour le fonctionnement relatif.

III.9.3.3 – Fonctionnements répétitifs

La commande GP peut être utilisée sans paramètre pour relancer exactement la même commande « GP » que la dernière exécutée.

De plus, il est possible d'activer un mode répétitif en ajoutant le type « M » à la commande GP qui a pour effet de réarmer automatiquement la fonction sortie différée à la fin de chaque exécution.

En type absolu, les sorties logiques seront activées (au travers du masque) à chaque passage à la position POS du CPA (éventuellement replacés à la position DEL) tant que la fonction gestion différée n'aura pas été arrêtée par la commande GPA.

En type relatif, les sorties sont activées tous les xpas (x = POS) sous réserve de ne pas modifier le sens de mouvement, ni la valeur du CPA par une remise à 0 (Commande DI) car la gestion de l'instant de commutation reste, malgré tout, absolue. Le module SIMPA détermine en effet la future position de commutation en additionnant la position courante à la valeur relative désirée (il n'y a pas de comptage de pas indépendant).

III.9.3.4 – Fonctionnement permanent

Normalement le fonctionnement des sorties logiques différées est effacé (désarmé) par un Reset Hard ou Soft (mise sous tension ou commande MR). Le mode P (permanent) défini dans la commande GP permet de réarmer la fonction sortie logique différée au Reset, quels que soient les types ou autres modes définis.
Il est donc possible, par exemple, de générer une synchro à chaque tour moteur sans avoir besoin de la redéfinir à chaque mise sous tension.
Le fonctionnement permanent est effacé par toute commande GP sauf si cette dernière contient le paramètre 'P'.

III.9.3.5 – Interaction avec les séquences

La gestion des sorties logiques différées est totalement indépendante de la gestion immédiate des sorties.

Aussi, est-il parfaitement possible de modifier directement l'état des sorties par une commande GL (sans type) ou une directive NO dans une phase, même lorsqu'une activation différée des sorties logiques a été lancée préalablement par la commande GP.

Trois cas de figures se présentent :

- l'activation des sorties par la commande différée est déjà réalisée, la commande immédiate modifie l'état des sorties à partir de l'état obtenu,

- l'activation différée n'a pas encore eu lieu, la commande immédiate impose son état, c'est cet état qui sera pris en compte comme état présent (S_{t-1}) lors de l'activation différée,

- l'activation différée est quasi simultanée avec la commande immédiate, il est alors possible que la commande immédiate ne soit pas réellement exécutée, si elle interrompt le processus de modification différée notamment pour une commande immédiate demandée par une séquence. Il est donc recommandé d'éviter une modification « simultanée » des sorties logiques et de prévoir des temps ou distances de non recouvrement pour éviter tout aléa.

III.10 – Variables (uniquement pour les versions micropas)

Les modules SIMPA en version micropas proposent l'utilisation de variables afin d'augmenter les performances des automatismes réalisables : le comptage de cycle, la calibration, le paramétrage sont autant de possibilité supplémentaires.

III.10.1 – Définition des variables

Une variable est une donnée modifiable à tout moment à partir du dialogue série ou dans le déroulement même des séquences.
Ces données sont stockées sur 4 octets signés, elles peuvent représenter des valeurs numérique de consignes ou des données binaires (ou hexa) comme un état de sortie logique et son masque.

III.10.2 – Ecriture des variables

Le logiciel SIMPA définit jusqu'à 32 variables, elles s'utilisent comme n'importe quelle donnée et sont représentées par les numéros de 1 à 32 précédés du caractère #

Les valeurs des variables peuvent être modifiées ou imposées par les commandes immédiates :

PG v_1, [v_2 …, [v_{10}]..]]
PD # n : v_n en décimal
PH # n : v_m en hexadécimal

ou dans les séquences par les natures de phase
PV # n : v_n , PC # n , PI # n , PR # n , PA # : v_m

Leurs valeurs peuvent être relues au moyen de la commande QN # n ou testées en séquence par une phase de nature PT # n.

III.10.3 – Utilisation des variables

Les variables permettent principalement de modifier en cours de séquence la valeur des consignes des mouvements. Cette fonctionnalité est particulièrement efficace dans l'utilisation des sous-séquences.

Au-delà des consignes, la valeur des sorties logiques peut aussi être placée en variable (cde No).

La valeur d'une variable peut être aussi utilisée comme compteur de cycle et permettre de réaliser un nombre de cycle déterminé : phase PT.

III.10.4 – Initialisation des variables, valeurs courantes

La valeur d'une variable définie par les commandes immédiates PG, PD ou PH est conservée d'une mise sous tension à l'autre ou après réinitialisation (MR).

Par contre, la valeur prise par une variable au cours du déroulement d'une séquence n'est pas conservée à l'initialisation. Seule la valeur définie par la dernière commande immédiate est restituée, ce qui permet de définir parfaitement les conditions initiales.

Lorsque la valeur courante et la valeur initiale diffèrent, la commande de relecture de variable donne les deux valeurs. Par défaut, la valeur d'init d'une variable est 0 (sortie usine ou commande MRZ).

III.11 - Etat à la mise sous tension

Les paramètres de définition des mouvements moteurs sont stockés dans une mémoire alimentée par pile.
Ils sont donc conservés même après coupure de l'alimentation des modules.

Cependant un contrôle de l'intégrité des informations stockées en mémoire est effectué à chaque mise sous tension.

Si un défaut mémoire est constaté (status positionné à M), alors les valeurs par défaut suivantes sont imposées (ce sont les valeurs sorties usine) :

V_{min} : 500 p/s (ou ½ pas/s) (version pas entier ½ pas) 75 p/s (version micropas)

V_{max} : 2 000 p/s (ou ½ pas/s). (version pas entier ½ pas) 1000 p/s (version micropas)

T_r : 1 000 ms (durée des rampes accélération et décélération) . 200 p/s (version micropas)

N : 0 (nombre de pas à effectuer).

μ : 1 (fonctionnement en pas entier).

Sens du déplacement : +

Glissement toléré (Nb de pas) : 10

Courant moteur : 0

Polarité des Entrées Logiques : actives à 0 électrique

Butées : inactives

Sorties : FF (force le courant de "standby" en version pas entier)

Valeur de compteur de pas absolu : 0

Les deux dernières valeurs sont imposées à chaque mise sous tension ou Reset.

En version micropas, la configuration sortie usine peut être forcée par la commande MRZ.

IV - INTERFACES OPERATEUR

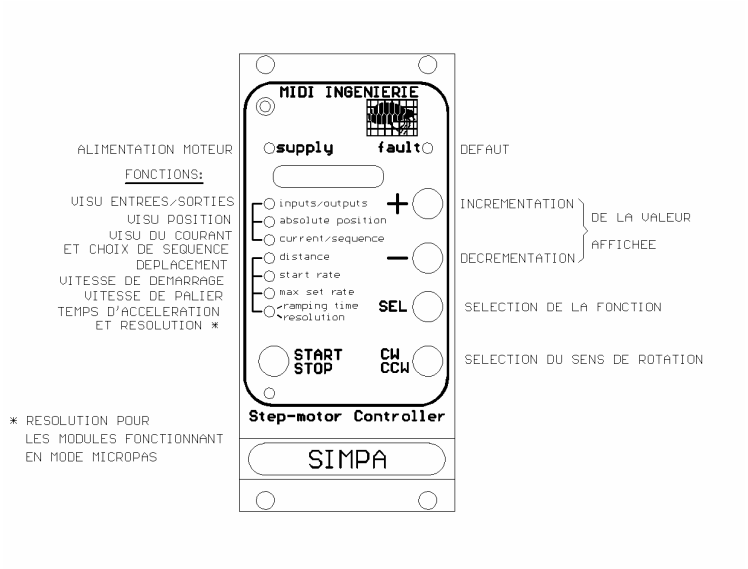
IV.1 - Face avant

Certains modules de la famille SIMPA peuvent être équipés en option d'une face avant, véritable interface opérateur.

Cette face avant offre à l'opérateur un accès simple aux fonctions du module pour une mise en œuvre rapide de mouvements moteur simples.

Toutes les fonctions restent accessibles par la liaison série, notamment le fonctionnement en séquence.

IV.1.1 - Face avant – Description



La face avant comprend cinq parties :

- en haut du module, un témoin d'alimentation et une visualisation des défauts,

- un afficheur 7 segments qui visualise les paramètres essentiels du module,

- une partie réservée à la sélection des paramètres d'environnement et d'état du module :
  * état des entrées/sorties logiques du module,
  * la valeur du compteur de pas absolu,
  * la liste des séquences existantes en mémoire module,
  * la valeur du courant nominal du moteur.

- une partie réservée à la saisie des paramètres de fonctionnement du module:
  * le nombre de pas à exécuter,
  * la vitesse de démarrage,
  * la vitesse de consigne,
  * la durée de la rampe d'accélération,
  * la résolution en micropas par pas.

- la partie basse est réservée au contrôle direct du moteur :
  * marche/arrêt,
  * sens du mouvement (CW/CCW),
  * un témoin de l'activité du moteur.

IV.1.2 - Sélection des paramètres à visualiser ou à modifier

A chaque paramètre sélectable correspond une led qui s'allume lorsque sa valeur est présente sur l'afficheur.

Pour les modules fonctionnant en mode micropas, les paramètres "temps de rampe" et "résolution" sont tous deux repérés par la même led. Le caractère le plus à gauche de l'afficheur précise la nature des paramètres.

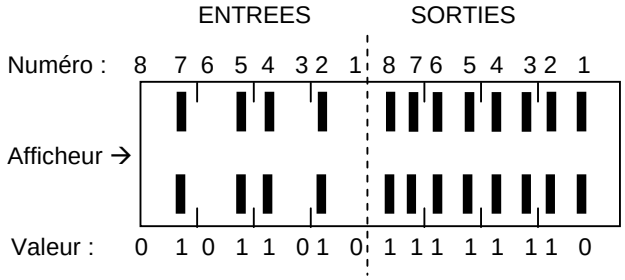
A : Temps de Rampe

U : Résolution

La sélection du paramètre à visualiser ou à modifier se fait par appuis successifs sur le bouton poussoir "SEL".

Les paramètres sont affichés avec les unités définies dans le chapitre fonctionnalités, les vitesses réellement générées peuvent différer légèrement de celles définies compte tenu de la résolution en temps des modules SIMPA.

Pour l'information Entrées/Sorties, chaque entrée ou sortie est représentée par un segment propre à l'afficheur : sortie à droite, entrée à gauche et 1 à 8 de droite vers la gauche. Le segment est présent lorsque la sortie ou l'entrée correspondante est active.



Sur l'exemple :

Motif entrée : 5Bh = 0 1 0 1 1 0 1 0
Motif sortie : FEh = 1 1 1 1 1 1 1 0

Pour les paramètres current/sequence, la partie gauche affiche « i »le prorata du courant nominal de la carte

appliquée au moteur, si ce dernier est réglable par logiciel $\left(\mathrm{Im} = \frac{Inom * i}{255} \right)$

La partie droite affiche la liste des séquences (défilé au moyen des poussoirs + ou -). Une séquence inexistante est précédée du signe -

Nota : lorsque le moteur est en marche, état matérialisé sur la face avant par la led située sous le bouton poussoir START/STOP, l'affichage est limité aux paramètres d'environnements seuls.

Etat des entrées et sorties logiques
Compteur de pas absolu
Numéro de séquence en cours d'exécution
Et en partie gauche la valeur du courant moteur

IV.1.3 - Modification de paramètres

Seuls les paramètres de fonctionnement sont modifiables.

Le principe de saisie décrit ci-dessous reste valable quel que soit le paramètre sélectionné. Seule la dynamique du paramètre, résumée dans le tableau ci-dessous, est différente.

Tableau 1 pour les modules fonctionnant en mode micropas

Nombre de pas programmable	: 1 à 9 999 999
Vitesse de démarrage	: (62/résolution) à vitesse palier (p/s)*
Vitesse de palier	: vitesse de démarrage à 20 000 p/s *
Temps de rampe	: 1 ms à 65535 ms *
Résolution	: 1 µpas/pas à 256 µpas/pas

Tableau 2 pour les modules pas entier et ½ pas

Nombre de pas programmable	: 1 à 2^{23}-1 (8388607)
Vitesse de démarrage	: 20 à (vitesse de palier -1) p/s (ou ½ pas/s)
Vitesse de palier	: (vitesse de démarrage + 1) à 20 000 (p/s ou ½ pas/s)
Temps de rampe	: 1 ms à 65535 ms *

* : Les paramètres Vitesses, Temps de rampe et Résolution doivent également respecter les relations du paragraphe III.4.6

Principe de modification

Les modifications des paramètres se font à l'aide des boutons poussoirs "+" et "-" (Bp+ et Bp-).

Un appui bref sur le Bp+ augmente d'une unité du paramètre sélectionné, inversement l'appui bref sur le Bp- provoque une diminution d'une unité.

Afin de permettre d'accéder rapidement à la valeur souhaitée, l'appui prolongé sur le Bp considéré (+ ou -) accélère l'incrémentation (Bp+) ou la décrémentation (Bp-) comme suit :

- l'évolution va se faire d'une unité par une unité jusqu'à ce que la valeur affichée soit multiple de 10, puis de 10 en 10, puis 100 par 100,...

L'auto-incrémentation (ou décrémentation) cesse dès que la valeur du paramètre atteint la limite maximale (ou minimale) autorisée.

Le relâchement du poussoir (+ ou -) pendant un temps court (≈ 1 s.) conserve le rang de l'incrémentation (décrémentation) (1, 10, 100, 1000,...). Cela autorise à reprendre la saisie sur le poussoir 'inverse' (changement +, - ou -, +).

Un relâchement prolongé provoquera le passage au rang 1 de l'incrémentation/décrémentation.

IV.1.4 - Contrôle du moteur

Sens de rotation (du prochain mouvement moteur)

Le bouton poussoir CW/CCW sélectionne, alternativement après appui, un des deux sens de rotation du moteur.

Le sens du moteur est symbolisé par la présence ou non d'un signe "-" lorsque le paramètre nombre de pas est affiché (CW : sens horaire " ", CCW : sens anti-horaire "-").

Remarque : l'appui sur le bouton poussoir sens, provoque automatiquement la sélection et la visualisation du paramètre Nombre de pas.

Marche/Arrêt

Un appui sur le bouton poussoir Start/Stop met le moteur en marche si le voyant correspondant est éteint, il l'arrêtera dans le cas contraire.

La commande Start/Stop fait exécuter au moteur le nombre de pas défini par le paramètre de distance en respectant les paramètres de vitesse et d'accélération.

L'arrêt du moteur par le poussoir start/stop est immédiat sans phase de décélération.

Attention ! Si le paramètre sélectionné lors de l'appui sur le bouton poussoir Start/Stop est un numéro de séquence existant, le module va alors exécuter la séquence indiquée et non pas un mouvement direct simple

Calcul d'une loi en cours

Chaque modification des paramètres vitesse et temps de rampe entraîne un calcul de loi avant le premier mouvement réalisé avec ces nouveaux paramètres.

Si le calcul de loi n'est pas terminé lors d'une demande de mouvement, cette dernière n'est exécutée qu'après la fin du calcul.

Le calcul de loi est symbolisé par le motif suivant sur l'afficheur : "--------"

Le temps nécessaire au calcul est au maximum de l'ordre de 4 secondes.

Le clignotement de la led "Absolute position" signale un dépassement de limite de la position absolue par rapport à la possibilité de l'afficheur (7 chiffres). La valeur n'est plus significative, mais la valeur interne du compteur reste correcte.

IV.1.5 - Fonctions étendues

Trois fonctions supplémentaires sont accessibles par la face avant :

- exécution d'un mouvement infini (GF)

- réinitialisation du compteur de pas absolu (DI)

- réinitialisation complète du module, analogue à la commande MR

Ces trois fonctions sont provoquées par un appui combiné du bouton poussoir + et du bouton poussoir – lorsque les paramètres suivants sont sélectionnés:

Entrées/Sorties logiques	: Mouvement infini	(GF)
Position absolue	: Raz du compteur de pas	(DI)
Séquence/courant	: Réinitialisation	(MR)

IV.1.6 –Réglage courant moteur

Pour les cartes dont le courant moteur est programmable, la commande GI Im ($0 \leq Im \leq 255$) permet d'en définir la valeur.

Le courant réellement appliqué au moteur est alors :

$$\text{Imoteur} = \text{Inom} * \frac{\text{Im}}{255} \quad (Aeff)$$

Attention ! La valeur Inom du courant nominal dépend du modèle de carte utilisé..

La valeur Im peut également être directement modifiée sur la carte, si celle-ci possède l'option face avant.

Ce réglage du courant n'est possible qu'à la mise sous tension selon les combinaisons des commutateurs suivantes :

Commutateur

Sw1.4	**Sw1.5**	**Sw1.1 à Sw1.3**	**Usage/Mode**
ON	ON	Motif adresse	Calculateur protocole Xon/Xoff
ON	OFF	Motif adresse	Calculateur protocole Ack/Nack
OFF	ON	Motif adresse	Console
OFF	OFF	Motif courant*	Face avant, Réglage du courant *Calculateur protocole Xon/Xoff imposé sur la liaison série *Adresse forcée à 0

Motif courant

Seules 8 valeurs de courant sont accessibles :

Sw1.3	Sw1.2	Sw1.1	Valeur Im	% du courant nolminal
ON	ON	ON	31	12 %
ON	ON	OFF	63	24 %
ON	OFF	ON	95	37 %
ON	OFF	OFF	127	50 %
OFF	ON	ON	159	62 %
OFF	ON	OFF	191	74 %
OFF	OFF	ON	223	87 %
OFF	OFF	OFF	255	100 %

IV.2 - Sélecteur de séquence

L'interface "sélecteur de séquence" permet de lancer l'exécution de séquences préenregistrées sans nécessiter de connexion à un calculateur pour les modules ne dispensant pas de face avant.

Cette option permet de sélecter, grâce à deux roues codeuses, une séquence mémorisée dans le module SIMPA (son numéro doit être compris entre 1 et 99).

L'exécution de la séquence sélectée est lancée dès que l'entrée logique E2 est active. Deux modes de fonctionnement sont possibles :

- en maintenant l'entrée E2 active en permanence, la séquence nommée par les roues codeuses est exécutée immédiatement. En modifiant la position des roues codeuses au cours de l'exécution de la séquence, on peut sélecter une séquence suivante différente *(Attention de ne pas modifier ce numéro juste en fin d'exécution sous peine de démarrer une séquence non désirée).*

- en n'activant l'entrée E2 que lorsqu'on désire exécuter la séquence, le module peut alors être synchronisé sur un événement externe (automate, pédale ...)

V - LIAISON CALCULATEUR ET COMMANDES

La liaison série présente sur chaque carte SIMPA permet le contrôle des modules par un calculateur hôte.
Une même liaison série permet de contrôler jusqu'à 64 modules d'un système multi-axes.

Le dialogue avec les modules SIMPA peut être grandement facilité par l'utilisation du logiciel PCSIM, véritable interface opérateur des modules SIMPA, ou grâce au Handler LIBSIM pour l'interface avec un programme écrit dans un langage évolué.

Ces deux logiciels disponibles uniquement pour les calculateurs compatibles PC, suppriment pour l'utilisateur tout l'aspect protocole lié au dialogue. Il ne reste plus à l'utilisateur qu'à donner le contenu de ses commandes.

Une version Windows de PCSIM peut être livrée sur demande avec la DLL de gestion du protocole de dialogue avec les modules SIMPA.

V.1 - Ligne et protocoles de communication

L'interface de communication de type RS232 (boucle de courant ou V24) est configurée comme ci-après :

- vitesse : 4 800 bauds (ou 9600 bauds uniquement pour la version micropas)
- données : 8 bits
- bits de stop : 1
- parité : pas de parité
- lignes utilisées : Rx et Tx

Les modules SIMPA peuvent être configurés dans l'un des deux modes de dialogue suivants : mode calculateur ou mode console.

Tous les modules placés sur une même ligne de communication doivent utiliser les mêmes modes, vitesses de transmission et protocoles.

V.1.1 - Mode calculateur

Il assure un contrôle global du message (nombre de caractères et "check sum").

Tout message reçu (ou émis) donne lieu à acquittement (caractère ACK: 06h) ou non acquittement (caractère NACK : 15h) suivant que la structure du message est correcte ou incorrecte (nombre de caractères et "check sum", mais pas de contrôle de syntaxe).

Les modules SIMPA ayant émis un message sur la ligne série considèrent l'absence d'acquittement dans un délai de 70 ms ou tout autre caractère que ACK comme un non acquittement.

Les non-acquittements provoquent de la part des modules SIMPA deux tentatives supplémentaires d'émission avant de positionner un statut d'erreur.

Les erreurs d'interprétation d'une commande ne sont signalées que lors du prochain adressage du module par le renvoi, à la place du caractère ACK concernant le message en cours, du caractère BEL (07h). L'émission d'un NACK pour la commande en cours reporte l'émission du BEL au prochain adressage du module.

Le protocole ACK/NACK peut être complété par un protocole supplémentaire permettant à chaque module :

- d'éviter de recevoir d'autres commandes tant que celles en cours ne sont pas totalement interprétées,
- de fournir un compte-rendu d'exécution.

Par la transmission après le caractère ACK ou BEL, du caractère XOFF (13h), le module concerné demande la suppression des émissions sur la ligne série, y compris de commandes destinées à d'autres modules. Le module concerné, après complète interprétation de la commande reçue, autorise la reprise du dialogue sur la ligne en émettant le caractère XON (1Ah) si la ou les commandes reçues ont été correctement interprétées ou le caractère XONERREUR (17h) si une commande n'a pas pu être exécutée (erreur de syntaxe). Dans le cas d'un message comportant plusieurs commandes, si une commande est détectée incorrecte, il y aura génération d'un XONERREUR, les commandes précédentes ayant été exécutées, les commandes suivantes étant perdues.

Gestion des erreurs en fonction du protocole retenu :

Protocole ACK/NACK : le caractère d'acquittement retourné par le module adressé (module 0 pour commande globale) dépend :

- de la syntaxe du message précédent,
- de la structure du message reçu.

		Structure du message reçu	
		Incorrecte	Correcte
Syntaxe du message précédent	incorrecte	NACK (15 h)	BEL (07 h)
	correcte	NACK (15 h)	ACK (06 h)

Protocole XON/XOFF

Le caractère d'acquittement retourné par le module adressé dépend uniquement de la structure du message reçu, le caractère de libération de ligne dépend de sa syntaxe

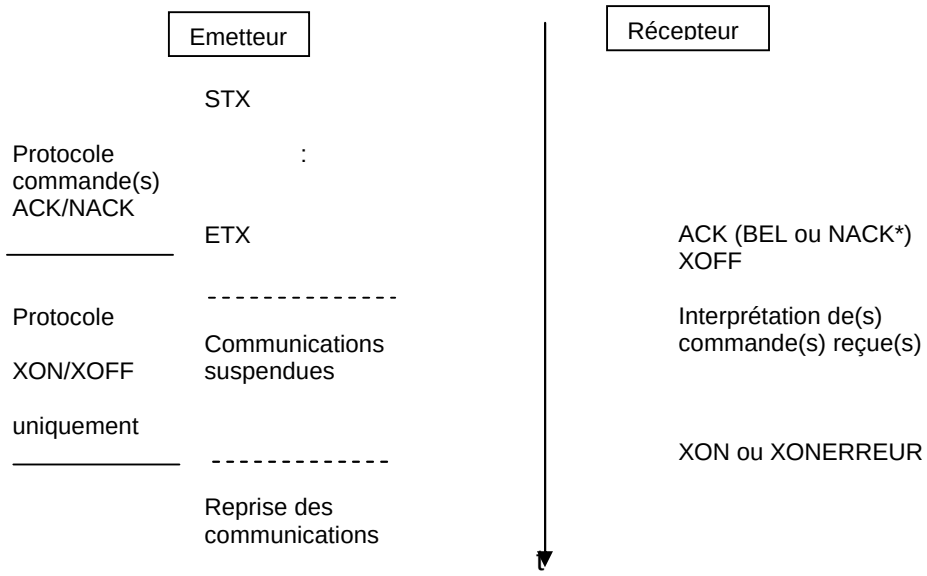
		Acquittement	Libération
Sructure du message reçu	correcte	ACK (06 h)	
	incorrecte	NACK (15 h)	
Syntaxe du message reçu	correcte		XON (1A h)
	incorrecte	BEL (07 h)*	XON ERREUR (17 h)

NOTA :

Dans le cas de commandes globales, seul le module 0 (obligatoire) assure le renvoi des caractères ACK, NACK, XOFF, XON et XONERREUR. Les autres modules mémorisent les erreurs éventuelles et dans ce cas transmettront le caractère BEL lors de leur prochain adressage individuel.

Lorsqu'un caractère BEL ou XONERREUR est retourné par un module, il est possible d'interroger le module (lecture code état) pour obtenir plus d'informations sur la cause de l'erreur.

Résumé des protocoles



* Dans ce cas le caractère XOFF est immédiatement suivi du caractère XON

V.1.2 - Mode console

Ce protocole a été développé pour faciliter le dialogue avec les modules à partir d'une simple console, sans calculateur, ou éventuellement à partir d'un automate.

Dans ce mode, le dialogue de type "écho" reporte les vérifications vers l'utilisateur qui doit s'assurer que les caractères retournés en écho par le module sont bien ceux souhaités. Le module retourne les caractères reçus à l'identique.

Dès réception du caractère CR (retour chariot) le module adressé retourne

CR	
LF	saut de ligne
>	prompt pour prochaine commande

Si une erreur est détectée par le module 0 dans le champ adresse, les caractères CR, LF et "prompt" retournés sont précédés de :

"_:?" erreur sur le champ adresse du message (renvoyé par le module 0).

Si une erreur est détectée sur les autres champs de la commande par le module adressé, la suite de caractères retournés CR LF est précédée de :

"_:!" erreur de syntaxe, de paramètre ou commande non autorisée.

Note : le symbole "_" matérialise l'espace (20 H)

V.2 - Syntaxe générale des commandes élémentaires

a) mode calculateur : protocole ACK/NACK avec ou sans protocole XON/XOFF

STX nc [@] cde1 [, cde2, cde3,...] CS ETX

STX et ETX	:	respectivement caractère de début (02h) et fin (03h) de message.
nc	:	nombre total sur 3 digits codés décimal des caractères transmis hormis STX, ETX, nc et CS. Le nombre de caractères doit être inférieur à 128 (nc ≤ 127)
[ ]	:	caractères optionnels.
@	:	adresse du module (00 à 63) auquel est destiné le message. Si l'adresse est omise, le message est destiné à l'ensemble des modules présents sur la ligne. Un module d'adresse 00 est indispensable dans toutes les configurations.
Cdei	:	libellé de la commande i avec ses paramètres.
,	:	séparation de commandes si plusieurs commandes sont regroupées dans un même message.
CS	:	contrôle de validité de type "check sum". Codé sur deux caractères ASCII représentant chacun un des deux digits hexadécimaux obtenus en faisant la somme arithmétique modulo 256 de tous les caractères du message sauf STX, nc, CS et ETX.

Les différents champs des commandes peuvent être séparés par un ou plusieurs espaces sauf entre le champ adresse (@) et la première commande (cde 1) où aucun espace n'est autorisé.

b) mode console : avec écho

[@] cde1 [, cde2, cde3, ...] CR

CR	:	retour chariot

Remarque : le nombre de caractères transmis entre 2 CR ne doit pas excéder 127 caractères !

V.3 - Etat du module et codes d'erreurs

Un registre d'état est positionné lors de l'interprétation de chaque commande reçue par un module.

Le registre peut être relu grâce à la demande : @RX (modules pas entier) ou @QX (modules micropas).
La réponse à cette demande est :

@EE_code

@ : adresse du module qui répond
EE : type de réponse : état du module
Code : caractère précisant l'état du module

* En l'absence de tout défaut le code N est retourné.
* En cas d'anomalie, un code d'erreur est retourné

Il existe deux niveaux d'erreurs :
Niveau 1 : erreurs liées à l'état du module ou erreurs générales.
Niveau 2 : erreurs spécifiques à chaque commande. Ces erreurs sont détaillées plus particulièrement avec chacune des commandes.

Erreurs de niveau 1 : le caractère retourné est alphabétique

A : commande non autorisée du fait de l'état du module (moteur en mouvement, mode séquence...).
B : arrêt immédiat du mouvement sur détection butées.
C : commande inconnue (erreur de syntaxe).
D : erreur de coordination de phase :
une commande phase hors de la commande SP
une commande non phase dans la commande SP
E : défaut sur entrées logiques (fréquence de basculement supérieure à 10 KHz).
I : en version micropas, commande GI interdite lorsque le module utilisé est un indexeur seul (le courant est alors imposé par l'amplificateur associé).
M : réinitialisation de la mémoire (suite à un défaut de sauvegarde).
ou en version micropas à une demande forcée de réinitialisation mémoire (commande MR0)
S : passage à la phase 255.
W : défaut de l'alimentation de puissance.
X : défaut matériel : provoque l'arrêt du mouvement en cours, la suppression de la puissance moteur et si une séquence est en cours d'exécution, arrête cette dernière.

Erreurs de niveau 2 : le caractère retourné est numérique.

0 : défaut lié au(x) paramètre(s) de la commande :
- absence de paramètre
- trop de paramètres
- syntaxe du paramètre

1 : premier paramètre hors limite (commandes DG, GI, DR, GA, RS, SE, SS, SA, SC, SD, WH, WL et WT) ou séquence inexistante (commande SP)
2 : second paramètre hors limite (commandes RS ou SN) ou phase inexistante (commande SP)
3 : la séquence précisée dans la commande n'existe pas (commande RS, SS, SA, SC, SD)
4 : phase inconnue (commande RS)
séquence déjà créée (commande SN)
défaut sur paramètre NE ou NF (commande SP)
5 : le nombre de phases restant disponible est insuffisant pour créer la séquence.

VI - DESCRIPTION DES COMMANDES

VI.1 - Conventions utilisées

Dans ce chapitre, chaque commande est décrite en utilisant un certain nombre de conventions pour exprimer :

- le type et la taille des paramètres,
- le type d'adressage (mono ou multi-modules),
- la validité de la commande en fonction de l'état du module ou de son type.

Un paramètre optionnel est représenté entre crochet '[' et ']'

Exemple :
GO [ddddddddd]

a) Type et taille des paramètres

Trois types sont différenciés :

- numériques : d
- numériques codés hexadécimal : h
- alphanumériques : a

La taille maximum est représentée par le nombre de caractères (d, h ou a) utilisés pour exprimer le paramètre.

Les espaces sont représentés par le caractère _.

Exemples :

- paramètre de déplacement absolu

  Pa = ±ddddddd

  signifie que la position absolue peut être donnée en utilisant au maximum 7 digits décimaux, plus un signe.

- paramètre définissant la gestion des entrées dans la définition d'une phase :

  Ne : aa_dd_dd_dd_dd_dd_dd_dd_dd_

  signifie que le paramètre Ne doit être constitué de deux caractères alphanumériques suivi de 8 groupes de trois caractères (un espace et deux digits décimaux).

Nota : Pour les valeurs numériques, les 0 non significatifs ainsi que le signe "+" peuvent être omis ou remplacés par des espaces.

Exemple : 00340 = __340 = 340

b) Type d'adressage

Le paramètre adresse du module est représenté par :

- @ si la commande ne peut être adressée qu'à un seul module à la fois.

- [@] si la commande peut être adressée à un ou à l'ensemble des modules.

- rien pour les commandes destinées à l'ensemble des modules.

c) Validité de la commande en fonction de l'état du module ou de son type

Si le caractère « * » précède une commande celle-ci est utilisable quel que soit l'état du module, dans le cas contraire le module doit être hors séquence et le moteur arrêté.

VI.2 - Liste des commandes

Les commandes peuvent être réparties en cinq groupes :

- initialisations paramètres

- lectures état module,

- déplacements (préparation et exécution),

- séquence (préparation et exécution),

- variable

Le tableau ci-dessous donne pour chaque groupe les mnémoniques des commandes.

Initialisation					Etat module		Déplacements				Séquences				Variable	
Paramètres		Courant moteur		Sorties logiques		Etat module		Préparation		Exécution		Préparation		Exécution			
p	µ	p	µ	p	µ	p	µ	p	µ	p	µ	p	µ	p	µ	p	µ
MB		GI		GL		RD	QD	D+		GA		SA		SS			PG
MN		GM			GP			D-		GE		SC					PD
MR		GR					QG	DR		GF		SD					PH
	MS					RL	QL			GH		SE					PC
WH						RP	QP		DG			SN					
WL						RS	QS		DI		GO	SP					
WT						RV	QV		DP		GS	SR					
	WN					RX	QX										

Une liste des commandes SIMPA, fournie en fin de document , donne pour chaque mnémonique, le ou les paramètres associés ainsi qu'une définition sommaire de l'action réalisée.

Les commandes spécifiques aux modules pas entiers / demi-pas sont repérées par l'indice p.

Les commandes spécifiques aux modules micropas sont repérées par l'indice µ.

VI.3 - Glossaire des commandes

(D+)$_p$: Sens horaire (commande réservée au module pas entier ½ pas) **(D+)$_p$**

Syntaxe : [@]D+

Paramètres : aucun.

Description : programmation du sens des prochains mouvements moteur de type relatif.

Cette commande sélectionne le sens horaire de rotation du moteur (en accord avec le câblage des phases moteur).

Pas de code d'erreur spécifique positionné.

(D-)$_p$: Sens anti-horaire (commande réservée au module pas entier ½ pas) **(D-)$_p$**

Syntaxe : [@]D-

Paramètres : aucun.

Description : programmation du sens des prochains mouvements moteur de type relatif.

Cette commande sélectionne le sens anti-horaire de rotation du moteur (en accord avec le câblage des phases moteur).

Pas de code d'erreur spécifique positionné.

DG : Tolérance au glissement **DG**

Syntaxe : [@] DG_np

Paramètres : np = ddd : nombre de pas (ou ½ pas ou micropas) définissant le glissement maximum toléré.

Numérique
Non signé (valeur absolue)
Limite : np ≤ 255

Description : fixe le seuil de glissement pris en compte par les phases de détection de glissement (NG).

Par défaut, le seuil est initialisé à la valeur 10.

A l'intérieur d'une séquence la valeur du seuil reste figée. Elle ne peut être modifiée qu'entre l'exécution de deux séquences.

La valeur du glissement peut être relue dans les commandes $(RL)_p$. ou $(QL)_\mu$

Code d'erreur positionné

0 : paramètre absent ou syntaxe incorrecte.

1 : valeur hors limite.

DI : Initialisation de la position absolue **DI**

Syntaxe : [@]DI

Paramètres : aucun

Description : remise à zéro du compteur de pas absolu.

La position actuelle du moteur devient donc la position de référence absolue des prochains mouvements : ORIGINE

NOTA : Une mise sous tension ou un MR provoque également cette initialisation (à la position courante du moteur).

Pas de code d'erreur spécifique positionné.

Attention ! Cette commande n'est pas acceptée lorsque le moteur est en mouvement ou si une séquence est en cours d'exécution.

DP : Position de référence **DP**

Syntaxe : [@]DP_Pr

Paramètres : Pr = ±ddddddddd : position de référence désirée

Numérique,
Limite : -8388606 ≤ Pr ≤ +8388606 (version pas entier ½ pas)
-2147483647 ≤ Pr ≤ +2147483647 (version micropas)

Description : Programmation d'une nouvelle position de référence pour tous les prochains mouvements de type absolu.

Cette commande permet de définir sans mouvement une position de référence différente de la position actuelle du moteur. Pr donne la position absolue (dans l'ancien repère) de la nouvelle référence.

La position est définie en pas ou ½ pas pour les modules pas entier / ½ pas, et en micropas pour les modules en mode micropas.

Code d'erreur positionné

Code 0 dans les cas suivants :

- la commande DP n'a pas de paramètre,
- la consigne fournie ne correspond pas à une valeur numérique.
- la valeur absolue du paramètre fourni est supérieure à 2^{31}-1 (ou 2^{23}-1)

|Pr| > 2^{31}-1 (version micropas)

|Pr| > 2^{23}-1 (version pas entier ou ½ pas)

Note

La commande DP 0 est équivalente à la commande DI.

(DR) $_{p}$: Déplacement relatif (commande réservée au module pas entier ½ pas) **(DR)** $_{p}$

Syntaxe : [@]DR_n

Paramètres : n = ddddddd: nombre de pas ou micropas du déplacement.

Numérique,
Non signé,
unité : pas, ½ pas ou micropas suivant la résolution du module
6 digits maximum,
Limite n ≤ 999998

Description : Programmation du nombre de pas ou ½ pas des prochains mouvements moteur de type relatif (commande GO).

Cette valeur est sauvegardée lors des coupures alimentation, le module conserve donc la dernière consigne lors d'une remise sous-tension.

Exemple : programmation de la consigne pour un mouvement relatif de 1890 pas :

DR 1890

Code d'erreur positionné

Code d'erreur = 0 si :

- la commande DR n'a pas de paramètre,
- le paramètre fourni ne correspond pas à une valeur numérique,
- le paramètre fourni est signé.

Code d'erreur = 1 si n > 999998

GA : Mouvement absolu **GA**

Syntaxe : [@]GA_Pa

Paramètres : Pa = ±ddddddddddd : position absolue désirée

Numérique,
Unité : pas, ½ pas ou micropas suivant la résolution du module,
10 digits maximum,
Limites : -8388606 ≤ Pa ≤ +8388606 (version pas entier ½ pas)
-2147483647 ≤ Pa ≤ +2147483647 (version micropas)

Description : exécution immédiate d'un mouvement de type absolu.

Le module positionne le moteur à la position absolue définie par la commande.

La position est définie en pas ou ½ pas pour les modules pas entier / ½ pas, et en micropas pour les modules en mode micropas.

S'il s'agit d'un premier mouvement après modification d'un des paramètres de vitesse , le module finit éventuellement le calcul de la loi d'accélération avant d'effectuer le mouvement.

Si le module se trouve déjà à la position requise, il n'y a pas de mouvement.

Avec un module µpas, la valeur donnée peut être remplacée par une variable : GA : # n

Exemple : exécution d'un mouvement jusqu'à la position absolue + 14576 :

GA 14576 (le signe + est optionnel).

Code d'erreur positionné

Code 0 dans les cas suivants :

- la commande GA n'a pas de paramètre,
- la consigne fournie ne correspond pas à une valeur numérique.
- $|Pa| > 2^{31}-1$ (version micropas)

Code 1 dans le cas suivant :

$|Pa| > 2^{23}-1$ (version pas entier ou ½ pas)

Note

La commande GA 0 est équivalente à la commande GH.

* GE : Arrêt avec décélération * GE

Syntaxe : [@] GE

Paramètres : Aucun

Description : le mouvement en cours (direct ou séquence) est décéléré jusqu'à Vmin puis arrêté.

En version pas entier ½ pas uniquement, les sorties logiques sont désactivées (« 1 » logique (FFh)) après l'arrêt.

Les prochains mouvements moteurs seront effectués à partir de Vmin.

La commande GE termine la séquence éventuellement en cours.

Pas de code d'erreur spécifique positionné.

Note :

- Si le mouvement en cours se fait à vitesse v = Vmin, l'arrêt est immédiat.

GF : Mouvement infini **GF**

Syntaxe : [@]GF [±] [v]

Paramètres : Mode pas entier ½ pas : Aucun
Mode micropas : v = ddddd nouvelle valeur de vitesse
Numérique
Unité : p/s
Limites 0 < v < 20000
Le sens du mouvement est défini par le signe de la consigne si celui-ci est explicite : + ou -

Description :

Lancement d'un mouvement accéléré (ou décéléré) jusqu'à la vitesse v puis maintien de la vitesse atteinte jusqu'à une commande d'arrêt ou d'initialisation (GS, GE, MR).
Le mouvement est exécuté dans le sens donné par le signe s'il est précisé. Dans le cas contraire, le sens du dernier mouvement est maintenu.
De même, si le mouvement est déjà en cours (version micropas uniquement) le sens est maintenu.
Si la vitesse n'est pas précisée (notamment pour les versions pas entiers), la vitesse du mouvement est Vmax (définie par WH) si la valeur donnée est 0, le mouvement s'effectue à vitesse minimum Vmin (définie par WL).
S'il s'agit d'un premier mouvement après modification d'un des paramètres de vitesse, le module termine d'abord, si besoin est, le calcul de la loi d'accélération avant d'effectuer le mouvement.

Nota : Seuls les mouvements initialisés par GF peuvent être modifiés par une nouvelle commande GF en version micropas, sinon la commande est refusée.

Seules les commandes GS ou GE, ainsi que MR arrêtent un mouvement infini.

Code d'erreur positionnée :
1 paramètre est incorrect

GH : Retour position origine : HOME **GH**

Syntaxe : [@]GH

Paramètres : aucun.

Description : exécution immédiate d'un mouvement de type 'Retour Origine'. Le module exécute un mouvement jusqu'à amener le compteur de pas absolu (ou micropas) à la valeur zéro.

La position origine peut être :

- la position du moteur à la mise sous tension ou après un RESET,
- la position du moteur au moment de l'exécution de la commande DI ou celle définie indirectement par la commande DP.

S'il s'agit d'un premier mouvement après modification d'un des paramètres de vitesse, le module termine d'abord, si besoin est, le calcul de la loi d'accélération avant d'effectuer le mouvement.

Si le module se trouve déjà en position origine, il n'y a pas de mouvement.

Pas de code d'erreur spécifique positionné.

Note : La commande GH est équivalente à la commande GA0

GI : Courant moteur **GI**

Syntaxe : [@]GI_Im

Paramètres : Im = ddd : amplitude du courant moteur

Numérique,
Non signé,
3 digits maximum,
Limite $0 \leq Im \leq 255$

$$Imoteur = \frac{Inom \times Im}{255}$$

Inom : courant nominal délivré par le module.

Description : Programmation de la valeur du courant appliqué au moteur (commande sans action sur certains modules SIMPA : se référer à la documentation spécifique).

Exemple : Pour un module SIMPA délivrant un courant maximum de 7A, la commande GI 128 permet de positionner la valeur de courant appliqué au moteur à 3,5 A.

Le paramètre GI est sauvegardé lors des coupures d'alimentation, la valeur par défaut à la première mise sous tension ou en sortie usine est cependant : .0 soit aucun courant délivré par la carte

Code d'erreur positionné

Code d'erreur = 0 si

- la commande GI n'a pas de paramètre,
- le paramètre fourni ne correspond pas à une valeur numérique,
- le paramètre fourni est signé négativement.

Code d'erreur = 1 si Im > 255

En version micropas :

Code d'erreur = I (i) si la commande GI est adressée à un module indexeur seul (non amplificateur), n'autorisant pas la programmation du courant !

* GL : Positionnement des sorties logiques * GL

Syntaxe : [@]GL_Out [: MSQ] [_t]

Paramètres :

	Out =	hh	Motif hexadécimal à présenter sur les sorties logiques
en mode µpas uniquement !!!	MSQ =	hh	Masque de modification : seuls les bits correspondant aux bits à 1 du masque sont modifiés réellement.
	t :	type	- si non précisé, positionnement immédiat
		+A / -A / A	- positionne au passage croissant (+), décroissant (-) ou indifféremment du CPA à une position absolue définie par la commande GP.
		+R / -R / R	- identique pour un déplacement relatif à la position courante du compteur de pas absolu

Description :

Lorsque le type n'est pas précisé, cette commande permet de contrôler globalement ou partiellement (avec masque) les sorties logiques du module et donc d'activer par exemple certains actuateurs parallèlement au déroulement d'une séquence.
Avec type, elle permet de conditionner l'état des sorties logiques à la position du moteur via le compteur de pas absolu. Il est donc possible de réaliser ainsi des impulsions de synchronisation liées à la position instantanée du moteur.

Voir la commande GP pour la définition des positions et largeur d'impulsion, lancement et arrêt de la fonction.

Attention :

La réponse à cette commande, peut être perturbée par le déroulement d'une séquence ou vice et versa.

En version pas entier ½ pas les sorties sont remises à FF en fin de mouvement ou de séquence, et le bit 1 des sorties logiques contrôle l'activation du standby, il convient donc de le positionner à 0 lors des mouvements sous peine de faire fonctionner le moteur en standby.

Motif	Sortie active (à 0)
FF	Aucune : Courant de Standby dans le moteur
FE	1 : Courant nominal (version pas entier ½ pas)
FD	2 : Standby
FB	3 : Standby
F7	4 : Standby
EF	5 : Standby
DF	6 : Standby
BF	7 : Standby
7F	8 : Standby
00	Toutes : Courant nominal (version pas entier ½ pas)

Code d'erreur positionné :

Code 0 dans les cas suivants :
- la commande GL n'a pas de paramètre,
- le motif fourni ne correspond pas à une valeur numérique codée hexadécimale.

Code 1 dans les cas suivants :
- si le paramètre fourni est supérieur à 255 (oFFh)
- défaut sur l'un des paramètres optionnels

* GM : Puissance moteur * GM

Syntaxe : [@]GM

Paramètres : aucun

Description : activation de la puissance moteur : le courant injecté dans le moteur est :

- le courant de Standby lorsqu'aucune commande de mouvement n'est en cours d'éxécution.

- le courant nominal si un mouvement est en cours (éventuellement programmable par la commande GI si le module accepte la programmation du courant).

Remarque : en version pas entier ½ pas , le standby est contrôlé par la sortie logique 1 qui peut donc modifier les états précédemment décrits.

Rappel :
La mise en puissance du moteur est implicite à chaque mouvement de type direct (GO GF GA GH) ainsi qu'au démarrage d'une séquence.

Attention ! Dans le cas d'un module SIMPA indexeur associé à un amplificateur séparé, cette commande peut être sans effet (conférer les explications données avec la commande GR).

GO : mouvement relatif **GO**

Syntaxe : [@]GO_[±][n]

Paramètres :

mode pas entier ½ pas : aucun

mode micropas : n = ddddddddd valeur de déplacement
Numérique
Unité : pas, ½ pas ou micropas suivant la résolution du module
10 digits maximum
Limite |n| ≤ 2147483647
Le sens du mouvement est défini par le signe de la consigne, si celui-ci est explicite : + ou –
Si le signe n'est pas donné, le sens du mouvement est celui du dernier mouvement relatif exécuté.

Description : exécution d'un mouvement de n pas ou micropas et dans le sens explicitement fourni dans la commande.

Le déplacement est défini [en pas ou demi-pas pour les modules pas entier/demi-pas, et] en micropas pour les modules en mode micropas.

S'il n'y a pas de paramètre : le mouvement est identique au dernier mouvement relatif effectué par le module.

Si le paramètre est un signe : le mouvement s'effectue dans le sens précisé par la commande avec un nombre de micropas identique au dernier mouvement relatif exécuté.

Pour les modules pas entier/demi-pas, le déplacement est défini par les commandes DR, D+ ou D-.

Pour les modules micropas, la valeur peut être remplacée par une variable : GO # n.

Les sens et consignes sont sauvegardés lors des coupures d'alimentation, le module conserve donc les derniers paramètres lors d'une remise sous tension.

Code d'erreur positionné :

0 : - Le paramètre "n" n'est pas un paramètre numérique

1 : - Le paramètre est hors limite (> à 2^{31}-1)

Commande réservée aux modules micropas

(GP)$_\mu$: Définition de la position de commutation des sorties logiques différées et armement **(GP)**$_\mu$

Syntaxe : [@] GP [p] : [d] [M] [P] [A]

Paramètres :

p = ± ddddddddd position d'activation relative ou absolue selon le type donné dans la commande GL
d = ± ddddddddd position de désactivation (éventuelle)
p et d : numériques, unité : micropas

M : fonctionnement répétitif : modifie les sorties à chaque passage à la position définie par p dans le sens défini par le type donné par la commande GL..

P : maintient le fonctionnement et réarme la commande des sorties différées à chaque reset ou mise sous tension.

A : arrêt du fonctionnement des sorties logiques différées.

Description :

Cette commande arme la fonction de modification des sorties logiques différées en précisant la position relative ou absolue du moteur à laquelle la modification doit être effectuée, et éventuellement la position de désactivation (le sens de transition est précisé dans la commande GL, un type A ou R doit préalablement être défini). Le paramètre d doit être donné pour préciser le fonctionnement impulsionnel si il est désiré. Lorsqu'aucun paramètre n'est donné, la fonction est réarmée avec les derniers paramètres utilisés (p et éventuellement d) mais la présence du seul paramètre p arrête le fonctionnement impulsionnel. Si le caractère permanent est demandé, il doit systématiquement être donné dans toutes les commandes GP.

Nota : pas de commande équivalente pour les modules pas entier ½ pas.

* GR : Coupure puissance moteur * GR

Syntaxe : [@]GR

Paramètres : aucun.

Description : suppression de la puissance moteur, le courant moteur est ramené à 0.

Pas de code d'erreur spécifique positionné.

Toutes les commandes de déplacement (GO, GA, GF, GH) et le lancement d'une séquence provoquent automatiquement la mise sous tension du moteur. Il est possible de couper et remettre cette puissance en cours de mouvement.

Attention !

- Dans le cas d'un module SIMPA indexeur, version micropas, associé à un amplificateur séparé, cette commande active simultanément les sorties boost et standby. Si celles-ci sont correctement reliées aux entrées correspondantes des modules MI ou MIP, leur activation simultanée provoque effectivement la mise hors puissance du moteur.

- Dans le cas d'autres modules du commerce, il conviendra probablement de gérer la puissance moteur à l'aide d'une sortie logique. Ceci est également valable lorsqu'on utilise un indexeur SIMPA en version pas entier ½ pas.

* GS : Stop * GS

Syntaxe : [@]GS

Paramètres : aucun.

Description : arrêt immédiat d'un mouvement ou d'une séquence en cours.

Les prochains mouvements seront effectués à partir de Vmin.

La commande GS termine la séquence éventuellement en cours.

En version pas entier ½ pas, les sorties logiques sont positionnées à "1" logique (FFh).

Pas de code d'erreur spécifique positionné.

Remarque : l'arrêt s'effectue sans décélération, il peut donc provoquer un glissement ou décrochement du moteur

MB : Autorisation du mode butée, Polarité des entrées logiques **MB**

Syntaxe : [@] MB [L/H]

Paramètres :

mode pas entier ½ pas : aucun

mode micropas : Optionnel codage de la polarité des Entrées logiques
H : une entrée excitée impose l'état logique 1 (9)
L : une entrée excitée impose l'état logique 0

Description : Cette commande autorise le module à gérer le mode Butée et peut définir également, pour les modules micropas, la polarité des entrées logiques.
Si aucun paramètre n'est donné, la polarité en cours des entrées logiques est conservée.
Une entrée est considérée active lorsque son état logique vaut 0.

Dans le mode de fonctionnement butée, deux entrées logiques (E7 et E8) sont utilisées en tant que détection de butée.
L'activation de l'entrée logique E7 (ou Butée+) lors d'un mouvement de sens horaire, provoque l'arrêt immédiat du mouvement, ou de la séquence réalisant ce mouvement. Cette entrée n'a pas d'action sur les mouvements en sens anti-horaire. De même, l'activation de l'entrée logique E8 (ou Butée-) lors d'un mouvement de sens anti-horaire provoquera son arrêt immédiat (mouvement ou séquence).
La configuration du mode Butée est sauvegardée pendant les coupures d'alimentation.

Code d'erreur positionné :

0 : Le paramètre fourni est différent du caractère de signe "H" ou "L" unique.

Note : La sélection de la polarité peut se faire sans mettre en œuvre le mode butée par la commande MN.
La valeur du codage des entrées logiques est par défaut : L (en sortie usine)
La valeur retournée par les commandes de relecture est toujours l'état logique.
Les configurations Mode Butée et Polarité peuvent être relues par la commande QL (µ)

MN : Inhibition du mode Butée, Polarité des entrées logiques **MN**

Syntaxe : [@] MN [L/H]

Paramètres :

mode pas entier ½ pas : aucun

mode micropas : Optionnel : codage de la polarité des entrées logiques
H : une entrée excitée impose l'état logique 1 (9)
L : une entrée excitée impose l'état logique 0

Description : Cette commande inhibe le mode de fonctionnement Butée (référence Commande MB). Cet état inactif des Butées est le mode par défaut du module en sortie usine.
Aucune entrée logique n'est affectée à une fonction spécifique.
Une entrée est considérée active lorsque son état logique vaut 0.

Code d'erreur positionné :

0 : Le paramètre fourni est différent du caractère « H » ou « L » unique.

9 une entrée optoisolée est excitée à l'état conducteur, une entrée non isolée pour une tension nulle.

* MR : Reset général du module * MR

Syntaxe : [@] MR [m]

Paramètres : pas entier, ½ pas : aucun
micropas m = a : remise en configuration sortie usine
(seule valeur admise m = Z)

Description : cette commande est équivalente à une remise sous-tension du module:

- un mouvement en cours est interrompu,
- le compteur de pas absolu est forcé à 0,
- les sorties sont forcées à l'état inactif : "1" logique (FFh),
- si une séquence de démarrage automatique est sélectée celle-ci est exécutée

Lorsque le paramètre m est donné (valeur Z), la configuration mémoire est effacée et remplacée par la configuration sortie usine (voir détail "Etat à la Mise Sous Tension").
De ce fait, toutes les séquences sont effacées et la notion de séquence de démarrage n'existe plus.

(commande accessible uniquement pour les modules micropas)

(MS)µ : Gestion des courants de repos et surcourant (MS)µ

Syntaxe : [@] MS m

Paramètres : m = a mode de gestion
valeurs admises N,S ou B

Description : Définition du mode de gestion des courants de repos et surcourant

* m = B l'ensemble des 3 états est autorisé
  - Courant de repos (standby) lorsque le moteur est arrêté
  - Surcourant (Boost) lors des phases d'accélération et de décélération
  - Nominal dans les mouvements

* m = S le mode de surcourant n'est plus utilisé, les phases d'accélération et de décélération se font au courant nominal.

* m = N quel que soit le mouvement ou non du moteur, le courant dans ce dernier est maintenu à sa valeur nominale.

Attention !

- Tous les modules SIMPA ne sont pas aptes à gérer le mode surcourant. Dans ce cas la commande MS B équivaut à la commande MS S
- Pour les modules indexeurs pilotant un amplificateur séparé, cette commande n'est pleinement efficace que si les connexions entre les sorties Boost et Standby de la carte indexeur sont correctement reliées aux entrées correspondantes de l'amplificateur.

Code d'erreur positionné :

Code 0 : La commandce MS n'a pas de paramètre
Le code fourni est différent des caractères N, S ou B

(commande accessible uniquement pour les modules micropas)

(*PD)µ : Valeur décimale d'une variable (*PD)µ

Syntaxe : [@] PD # n : v

Paramètres : n = dd : numéro de la variable à modifier $1 \leq n \leq 32$
v = ± ddddddddd : valeur à donner à la variable
Numérique
10 digits maximum
$-214783647 \leq V \leq +2147483647$

Description : donne la valeur v à la variable **#** n (valeur courante et valeur d'init)

Code d'erreur positionné :

Code 0 : Numéro de variable erroné
Valeur non numérique

Code 1 : $|v| > 2^{31} - 1$

(commande accessible uniquement pour les modules micropas)

(*PG)µ : Définition globale des variables, synchro (*PG)µ

Syntaxe : [@] PG [v_1 : [v_2 : [……. : [v_{10}]….]]]

Paramètres : vi = ± ddddddddd : valeur à donner à la variable # i
Numérique
10 digits maximum
$-2147483647 \leq vi \leq +2147483647$

Description : donne aux variables 1 à 32 respectivement les valeurs v_1, v_2 à v_{32} Si le nombre des valeurs données est inférieur à 32, seul les premières variables sont modifiées.
Cette commande est aussi utilisée comme synchro pour les phases de nature NY et dans ce cas ne comporte éventuellement aucun paramètre.

Code d'erreur positionné :

Code 0 : Valeur non numérique

Code 1 : $|v_U| > 2^{31} - 1$

(commande accessible uniquement pour les modules micropas)

(*PH)µ : Valeur hexadécimale d'une variable **(*PH)µ**

Syntaxe : [@] PH # n : v

Paramètres : n = dd : numéro de la variable à modifier $1 \leq n \leq 32$
v = hhhh : valeur hexadécimale à donner à la variable
Numérique
8 digits maximums
$0 \leq v \leq$ oFFFFFFFFh

Description : donne la valeur v à la variable **#** n (valeur courante et valeur d'init)

Remarque : les variables sont codées sur 31 bits + signe aussi la valeur 10000001h correspond à –1 en décimal et oFFFFFFFFh à $-2^{31}+1$

Code d'erreur positionné :

Code 0 : Numéro de variable erroné
Valeur non numérique

Code 1 : $v \geq$ 100000000h

(commande accessible uniquement pour les modules micropas)

(*QD)µ : Suivi des mouvements et séquences (*QD)µ

Syntaxe : @QD

Paramètres : aucun

Format de la réponse :

@ED_Séquence_Phase_Sens_Nature_Position_Entrées_Sorties_Contrôle_Séquence Suivante_Roue Codeuse
@ED_ddd_ddd_±_aa_±ddddddddddd_hh_hh_aa_ddd_dd

Soit un total de 45 caractères au maximum

- Seuls les digits utiles sont retournés

Description : demande de relecture de l'ensemble des paramètres de suivi de mouvement/séquence.

En cours d'exécution de mouvement immédiat ou de séquence, il est possible de suivre le fonctionnement en recherchant certains paramètres qui sont :

@	: Adresse du module interrogé
Séquence	: Numéro de la séquence en cours d'exécution (ou de la dernière séquence exécutée). Le n°0 signale que le module exécute un mouvement immédiat.
Phase	: Le numéro de la phase en cours d'exécution (ou n° de la dernière phase exécutée).
Sens	: Sens du mouvement (ou du dernier mouvement effectué).
Nature	: Nature du mouvement ou de la phase
Position	: Valeur du compteur de pas absolus.
Entrées	: Etat des entrées, en cas de détection de défauts sur les entrées logiques, la valeur retournée est XX (cf III.8.4).
Sorties	: Etat des sorties.
Contrôle	: Lx : mode local ou direct xF : énergie off Sx : mode séquence xO : énergie on Ux : mode sous séquence
Séquence chaînée	: N° de la prochaine séquence (valable si séquence en cours). (0 pour aucune)
Roue Codeuse	: N° de séquence (≤ 99) lu sur l'interface Sélecteur de séquence. (0 pour aucune)

Contrainte
Cette commande doit être impérativement adressée à un module unique.

Précision sur la lecture du compteur de pas absolu (CPA) ou micropas

Pendant la lecture par la commande QD, du compteur de pas absolu, celui-ci peut être modifié par le déroulement normal des pas (particulièrement à vitesse élevée), ce qui peut introduire des résultats aberrants. Pour éviter ces derniers, le module effectue jusqu'à 10 essais de lecture. Si malgré tout, il n'obtient pas de lecture sans interruption, due au mouvement, la valeur retournée par le module sera précédée par le symbole "!" de façon à préciser la nature incertaine de la valeur donnée.

Ce phénomène peut se produire quelle que soit la nature de la phase en cours.

Nota : Commande équivalente pour les modules pas entier / ½ pas : RD.

(commande accessible uniquement pour les modules micropas)

(*QG)µ : Suivi des séquences et phases (*QG)µ

Syntaxe : @QG

Paramètres : aucun

Format de la réponse :

@EG_Séquence_Phase_Sens_Contrôle

@EG_ddd_ddd_±_aa

Soit un total de 17 caractères maximums, seuls les digits significatifs sont émis.

Description : demande de relecture des n° de séquence et phase en cours, ainsi que d'une information de contrôle.

Séquence : N° de la séquence en cours d'exécution (ou n° de la dernière séquence exécutée). En cas de mouvement direct, la valeur retournée est : 0

Phase : N° de la phase en cours d'exécution (ou n° de la dernière phase exécutée)
Le n°254 est donné pour la phase d'arrêt
Le n°255 est donné pour la phase d'interruption

Sens : Sens du mouvement (ou du dernier mouvement réalisé)

Contrôle : Deux caractères précisant :

1) le mode local : **L** ou le mode séquence : **S** ou le mode séquence : **U**

2) Energie off : **F** ou Energie on sur le moteur : **O**

Nota : Cette commande n'a pas d'équivalent en version pas entier / ½ pas .

(commande accessible uniquement pour les modules micropas)

(*QL)µ : Relecture des paramètres locaux (*QL)µ

Syntaxe : @QL

Paramètres : aucun

Format de la réponse :

@EL WL:Vmin WH:Vmax WT:ta[:td] WN:résolution DR:±consigne GI:courant DG:glissement MD :Mode ButéePolarité

@EL_WL:ddddd_WH:ddddd_WT:ddddd_[:ddddd]_WN:ddd DR:±ddddddddddd_GI:ddd_DG:ddd_MD :da_aa_a

Description :

Cette commande permet à l'utilisateur de rechercher les paramètres suivants :

@	: adresse du module interrogé	
Vmin	: vitesse minimum (paramètre de la commande WL)*	
Vmax	: vitesse maximum (paramètre de la commande WH)*	
ta	: durée de l'accélération	
td	: durée de la décélération (si différent de ta)	
résolution	: résolution programmée (paramètre de la commande WN)*	
consigne	: consigne du mouvement relatif, mémorisée lors de la commande GO ±n	
courant	: consigne du courant moteur (paramètre de la commande GI)	
glissement	: valeur du glissement maximum toléré (paramètre de la commande DG)	
mode	: 0 : Mode micropas 1 : Mode pas entier 2 : Mode demi-pas	B : gestion du surcourant et du standby N : forçage du courant nominal S : gestion du standby
Butée	: MN ou MB	
Polarité	: L ou H	

Contrainte :

Cette commande doit être impérativement adressée à un module unique dans une configuration muti-axes.

*Note :

La relecture des paramètres vitesse peut donner des valeurs différentes de celles programmées.
Les valeurs retournées sont celles réellement générées par le module compte tenu de la quantification liée au générateur de fréquence et à la résolution du calcul de loi d'accélération.

Nota : Commande équivalente pour les modules pas entier / ½ pas : RL.

(commande accessible uniquement pour les modules micropas)

(*QN)µ : Relecture d'une variable **(*QN)µ**

Syntaxe : @QN # n

Paramètres : n = dd numéro de la variable à lire ($1 \leq n \leq 32$)

Format de la réponse :

@EN_# n : v [,(V)]

@EN_# dd = ±ddddddddddd, (±ddddddddddd)

Soit un total de 33 caractères maximums.

Description : demande de relecture d'une variable

v = valeur courante de la variable
V = valeur d'init de la variable (si différent de v)

Code d'erreur positionné :

0 : n° de variable erroné, absence de numéro de variable, …

Note :

L'ajout du suffixe '-H' à la commande permet une relecture en hexadécimale de la variable, dans ce cas le format de la réponse est :
@EN_dd : Hhhhhhhhh, (Hhhhhhhh)
avec hhhhhhhh : codage hexadécimal

(commande accessible uniquement pour les modules micropas)

(*QP)µ : Lecture de position (*QP)µ

Syntaxe : @QP

Paramètres : aucun

Format de la réponse :

@EP Position_absolue Entrées_logiques Sorties_logiques

@EP_±ddddddddddd_hh_hh @EP!±ddddddddddd_hh_hh

Soit un total de 22 caractères maximums (seuls les digits significatifs sont émis).

Pour des vitesses moteur élevées il est possible que le logiciel n'arrive pas à lire une position stable du compteur de pas. La valeur retournée peut alors être aberrante. Le caractère "!" permet d'indiquer cet aléa possible (cf. commande QD).

Nota : Commande équivalente pour les modules pas entier / ½ pas : RP.

(commande accessible uniquement pour les modules micropas)

(*QS)µ : Lecture de phase (*QS)µ

Syntaxe : @QS_Séquence_Phase

Paramètres :

Séquence et phase : ddd
Numériques non signés
3 digits maximums
Limites 1 ≤ Phase ≤ 200
Limites 1 ≤ Séquence ≤ 100

Séquence : numéro de la séquence contenant la phase à relire

Phase : numéro de la phase dans la séquence

Format de la réponse : (les 0 à gauche, non significatifs, sont omis)

@ES_Séquence_Phase_Nature_Consigne_NS_Phase_Suivante
_NO_Sortie Logique
_NL_Séquence Suivante
_NQ_Sous_Séquence
_NE_EL1_EL2_EL3_EL4_EL5_EL6_EL7_EL8
_NF_EL1_EL2_EL3_EL4_EL5_EL6_EL7_EL8

@ES_ddd_ddd_aa_±ddddddddd_NS_ddd
_NO_hh : hh
_NL_ddd
_NQ_ddd
_NE_ddd_ddd_ddd_ddd_ddd_ddd_ddd_ddd
_NF_ddd_ddd_ddd_ddd_ddd_ddd_ddd_ddd

Les informations NE/NF NL/NQ et NO ne sont retournées dans cet ordre que si elles ont été définies dans la phase.
Les informations NL et NQ ainsi que NF et NE sont exclusives.

Code d'erreur positionné

0 : Défaut sur les paramètres de la commande :
- il n'y a pas le nombre exact de paramètres (2)
- les paramètres sont non-numériques ou signés *

1 : N° de séquence hors limite : = 0 ou > 100
2 : N° de phase hors limite : = 0 ou > 200
3 : Séquence inexistante
4 : Phase inexistante dans la séquence (n° de la phase > au ns de la commande SN)

* Seul le signe '' - '' provoque une erreur de type paramètre signé.

Note :
- La relecture d'une phase non définie donne :
@ES Séquence Phase XX NS_0

Nota : Commande équivalente pour les modules pas entier / ½ pas : RS.

(commande accessible uniquement pour les modules micropas)

(*QV)µ : Demande de lecture des numéros de version et indice du logiciel **(*QV)µ**

Syntaxe : @QV

Paramètres : aucun

Format de la réponse :

@EV VR code

@EV_hh_ddddd

V : numéro de version
R : numéro d'indice
Code : information réservée pour usage interne à Midi Ingénierie

Nota : Commande équivalente pour les modules pas entier / ½ pas : RV.

(commande accessible uniquement pour les modules micropas)

(*QX)µ : Demande d'état/Acquittement des interruptions (*QX)µ

Syntaxe : @QX

Paramètres : aucun

Description : lorsque le module détecte une anomalie sur une commande, il positionne un code d'erreur.

Ce code est lu au moyen de la commande QX. Le module adressé répond le message suivant :

@EE_Code_état

@ : adresse du module sur 2 digits
EE : générique de la réponse
Code_état : Code d'état du module

Remarque :

Le fait de lire ce code par la commande QX provoque une remise à zéro du code d'erreur, un retour à l'état nominal du module : Code_état = N

En cas d'anomalie permanente (exemple : défaut sur l'alimentation : W), le code d'erreur peut persister tant que l'origine du défaut détectée n'a pas été supprimée.

Exemple

00QX…00EEN

00GI -10 ; commande erronée car valeur signée

00QX…00EE_0 ; retourne le code d'erreur 0 et l'efface

00QX…00EE_N ; 'pas d'erreur'

Contrainte

Cette commande doit impérativement être adressée à un module unique

Nota : Commande équivalente pour les modules pas entier / ½ pas : RX.

(commande réservée aux modules pas entier ½ pas)

(*RD)p : Suivi des mouvements et séquences (*RD)p

Syntaxe : @RD

Paramètres : aucun

Format de la réponse :

@ED_Séquence_Phase_Sens_Nature_Position_Entrées_Sorties_Contrôle_Séquence Suivante_Roue Codeuse

@ED_dd_dd_± aa_±ddddddd_hh_hh_aa_ddd_dd

39 caractères sont systématiquement retournés.

Description : demande de relecture de l'ensemble des paramètres de suivi de mouvement / séquence.

En cours d'exécution de mouvement immédiat ou de séquence, il est possible de suivre le fonctionnement en recherchant certains paramètres qui sont :

@	: Adresse du module interrogé
Séquence	: Numéro de la séquence en cours d'exécution (ou de la dernière séquence exécutée). Le n°0 signale que le module exécute un mouvement immédiat
Phase	: Le numéro de la phase en cours d'exécution (ou n° de la dernière phase exécutée).
Sens	: Sens du mouvement (ou du dernier mouvement effectué).
Nature	: Nature du mouvement ou de la phase
Position	: Valeur du compteur de pas absolus.
Entrées	: Etat des entrées, en cas de détection de défauts sur les entrées logiques, la .valeur retournée est XX (cf III.8.4).
Sorties	: Etat des sorties.
Contrôle	: Lx : mode local ou direct xF : énergie off Sx : mode séquence xO : énergie on Ux : mode sous séquence
Séquence chaînée	: N° de la prochaine séquence (valable si séquence en cours). (0 pour aucune)
Roue Codeuse	: N° de séquence (≤ 99) lu sur l'interface Sélecteur de séquence. (0 pour aucune)

Contrainte
Cette commande doit être impérativement adressée à un module unique.

Précision sur la lecture du compteur de pas absolu (CPA) ou micropas

Pendant la lecture par la commande RD, du compteur de pas absolu, celui-ci peut être modifié par le déroulement normal des pas (particulièrement à vitesse élevée), ce qui peut introduire des résultats aberrants. Pour éviter ces derniers, le module effectue jusqu'à 10 essais de lecture. Si malgré tout, il n'obtient pas de lecture sans interruption, due au mouvement, la valeur retournée par le module sera précédée par le symbole "!" de façon à préciser la nature incertaine de la valeur donnée.

Ce phénomène peut se produire quelle que soit la nature de la phase en cours.

Nota : Commande équivalente pour les modules micropas : QD.

(commande réservée aux modules pas entier ½ pas)

(*RL)p : Relecture des paramètres locaux (*RL)p

Syntaxe : @RL

Paramètres : aucun

Format de la réponse :

@EL Vmin Vmax Tr Consigne Courant Mode gL

@EL_ddddd_ddddd_ddddd_±dddddd_ddd_d_ddd

Soit un total de 41 caractères.

Description : demande de lecture des paramètres locaux pas entier ½ pas

Cette commande permet à l'utilisateur de rechercher les paramètres suivants :

@	: Adresse du module interrogé
Vmin	: Vitesse minimum (paramètre de la commande WL)*
Vmax	: Vitesse de consigne (paramètre de la commande WH)*
Tr	: Temps de rampe (paramètre de la commande WT)
Sens	: Sens du mouvement relatif (positionné par D- ou D+)
Consigne	: Consigne du mouvement relatif (paramètre de la commande DR)
Courant	: Consigne du courant moteur (possibilité de programmation du couple)
Mode	: information sur le fonctionnement en pas ou ½ pas (2 : ½ pas, 1 : pas entier 0 : micropas)
gL	: valeur du glissement maximum toléré fixé par la commande DG (par défaut 10)

Contrainte
Cette commande doit être impérativement adressée à un module unique dans une configuration multiaxes.

*Note

La relecture des paramètres vitesse peut donner des valeurs différentes de celles programmées. Cela est dû à la résolution du calcul dans le traitement effectué par le module.

Nota : Commande équivalente pour les modules micropas : QL.

(commande réservée aux modules pas entier ½ pas)

(*RP)p : Lecture de position **(*RP)p**

Syntaxe : @RP

Paramètres : aucun

Format de la réponse :

@EP Position_absolue Entrées_logiques Sorties_logiques

@EP_±ddddddd_hh_hh @EP!±ddddddd_hh_hh

Soit un total de 19 caractères maximums

Description : demande de relecture position entrées/sorties

Pour des vitesses moteur élevées, il est possible que le logiciel n'arrive pas à lire une position stable du compteur de pas. La valeur retournée peut alors être aberrante. Le caractère « ! » permet d'indiquer cet aléa possible (cf. commande RD).

Nota : Commande équivalente pour les modules micropas : QP.

(commande réservée aux modules pas entier ½ pas)

(*RS)p : Lecture de phase **(*RS)p**

Syntaxe : @RS_ns_np

Paramètres :

ns = dd numéro de la séquence contenant la phase.
np = dd numéro de la phase dans la séquence.

Description : demande de lecture d'une phase.

Cette commande permet à l'utilisateur de relire une phase précédemment programmée.

Les paramètres retournés par le module correspondent exactement à la commande Sp (définition de la phase).

Format de la réponse :

```
                                        NE                              NQxx
@ES_Ns_Np_Se_Na_Consigne_NF_EI.. E8_NS_ps_NO_xx_NLxx
```

Tous ces paramètres ont même format et signification que la commande SP.

Remarque

Si la phase n'a pas été initialisée par une commande SP, la réponse est : @ES_X

Quand un paramètre n'a pas été défini par la commande SP il est positionné à XX dans la réponse à la commande RS. Ainsi un retour avec RS... NOXX précise que le paramètre n'a pas été défini par SP.

Contrainte

Cette commande doit être impérativement adressée à un module unique dans une configuration multiaxes.

Cette commande n'est acceptée par le module que lorsqu'il se trouve hors séquence.

Code d'erreur positionné

0 : Défaut sur les paramètres de la commande.
Il n'y a pas le nombre exact de paramètres (2).
Les paramètres sont non-numériques, ou signés.

1 : Le N° de Séquence fourni est hors limite : = 0 ou > 99

2 : Le nombre de phases est hors limite : = 0 ou > 50

3 : La séquence précisée n'existe pas.

Attention !

Le contrôle du signe n'est effectué que sur le premier paramètre de la commande.

Nota : Commande équivalente pour les modules micropas : QS.

(commande réservée aux modules pas entier ½ pas)

(*RV)p : Demande de lecture des numéros de version et indice du logiciel **(*RV)p**

Syntaxe : @RV

Paramètres : aucun

Format de la réponse :

@EV VR code

@EV_hh_ddddd

V : numéro de version
R : numéro d'indice
Code : information réservée pour usage interne à Midi Ingénierie

Nota : Commande équivalente pour les modules micropas : QV.

(commande réservée aux modules pas entier ½ pas)

(*RX)p : Demande d'état/Acquittement des interruptions **(*RX)p**

Syntaxe : @RX

Paramètres : aucun

Description : lorsque le module détecte une anomalie sur une commande, il positionne un code d'erreur.

Ce code est lu au moyen de la commande RX. Le module adressé répond le message suivant :

@EE_Code_état

@ : adresse du module sur 2 digits
EE : générique de la réponse
Code_état : Code d'état du module

Remarque :

Le fait de lire ce code par la commande RX provoque une remise à zéro du code d'erreur, un retour à l'état nominal du module : Code_état = N

En cas d'anomalie permanente (exemple : défaut sur l'alimentation : W), le code d'erreur peut persister tant que l'origine du défaut détecté n'a pas été supprimée.

Exemple

00RX…00EEN

GI -10 ; commande erronée car valeur signée

00RX…00EE_0 ; retourne le code d'erreur 0 et l'efface

00RX…00EE_N ; 'pas d'erreur'

Contrainte

Cette commande doit impérativement être adressée à un module unique

Nota : Commande équivalente pour les modules micropas : QX.

SD : Séquence de démarrage **SD**

Syntaxe : [@]SD_ns

Paramètres : ns = ddd : n° de la séquence à démarrer à l'initialisation (mise sous tension ou commande MR)

Numérique,
Non signé,

Limite du paramètre :	1 ≤ N° de séquence ≤ 99	version pas entier ½ pas
	1 ≤ N° de séquence ≤ 100	version micropas

Description : sélection de la séquence à exécuter automatiquement lors des prochaines remises sous tension ou initialisation (commande : MR).

Le module peut exécuter automatiquement (sans action de l'utilisateur) une séquence lors des mises sous tension.

Le choix de la séquence de démarrage se fait par la commande SD.

Le paramètre N° de Séquence doit correspondre à une séquence connue du module.

Exemple : pour qu'à chaque remise sous tension le module exécute la séquence n° 11, il faut exécuter la commande : SD_11.

Code d'erreur positionné

Le module positionne les codes suivants :

0 : Défaut sur les paramètres.
Format incorrect.
Nombre de paramètre incorrect.

1 : Le N° de séquence fourni est hors limite.

3 : Le N° de séquence ne correspond pas à une séquence connue du module.

Contrainte

La séquence doit être créée avant de la sélectionner.

SE : Effacement de séquence SE

Syntaxe : [@]SE_ns

Paramètres : ns = ddd : n° de la séquence à effacer

Numérique,
Non signé,

Limite du paramètre :	1 ≤ N° de séquence ≤ 99	version pas entier ½ pas
	1 ≤ N° de séquence ≤ 100	version micropas

Description : effacement de la séquence ayant pour numéro le paramètre N° de Séquence.

Tout l'espace mémoire réservé précédemment pour décrire les phases de cette séquence redevient disponible pour la définition d'autres séquences.

Le paramètre N° de Séquence doit correspondre à un numéro de séquence déjà existant.

Remarque : si le paramètre N° de séquence est 0 toutes les séquences du module sont effacées.

Exemple : Effacement de la séquence ayant pour numéro 12 : SE_12.

Effacement de toutes les séquences : SE_0.

Code d'erreur positionné

0 : Défaut sur le paramètre.
Paramètre non numérique.
Le nombre de paramètre de la commande n'est pas respecté.

1 : Le paramètre est hors limite (> 200 ou 99)

Note : L'effacement d'une séquence inexistante ne provoque pas d'erreur !

SN : Création d'une séquence **SN**

Syntaxe : [@]SN_ns_np

Paramètres : ns = ddd : numéro de la séquence à créer
np = ddd : nombre de phases réservées pour cette séquence.

Numériques,
Non signés,
3 digits maximum,
limites : 1 ≤ N° de Séquence ≤ 99 } version pas entier ½ pas
1 ≤ Nombre de phase ≤ 50 }

1 ≤ N° de Séquence ≤ 100 } version micropas
1 ≤ Nombre de phase ≤ 200 }

Description : cette commande permet de réserver dans la mémoire du module la place nécessaire à la définition d'une séquence.

- Le paramètre Numéro de Séquence permet d'identifier la séquence à créer. Toute opération ultérieure sur cette séquence se référera à ce Numéro de Séquence.

- Le paramètre Nombre de phases permet au module de réserver uniquement l'espace mémoire nécessaire à la séquence.

Exemple : Création de la séquence n° 2 pouvant contenir jusqu'à 30 phases : SN_2_30.

Code d'erreur positionné

0 : Défaut sur les paramètres de la commande.
Il n'y a pas le nombre exact de paramètres (2).
Les paramètres sont non-numériques, ou signés.

1 : Le N° de Séquence fourni est hors limite

2 : Le nombre de phases est hors limite

4 : Le numéro de séquence fourni dans la commande est déjà affecté à une séquence du module.

5 : Il ne reste plus assez de phases disponibles dans le module pour créer cette séquence.

Cette erreur peut survenir si l'on crée des séquences ayant un nombre important de phases.

Actuellement le nombre de phases total est limité à ***414 en pas entier ½ pas***
384 en micropas

Contrainte

Il est impossible d'augmenter le nombre de phases d'une séquence déjà existante.
Il faut l'effacer et la recréer avec le bon nombre de phases.
On ne peut pas créer une séquence déjà existante (code erreur 4).

Attention !!!

Le contrôle du signe n'est effectué que sur le premier paramètre de la commande.

SP : Définition de la phase **SP**

Syntaxe :
Version pas entier ½ pas :
[@]SP_ns_np_Ss_Na_Cns[_NL_sc ou _NQ_sq][_Ne_X1_... X8][_NS_ps][_NO_out])

Version micropas
[@]SP_ns_np_Na_Cns[_NL_sc_NQ_sq][_Ne_X1_... X8][_NS_ps] NS@1[:@2][_NO_out][:msq]]

Description : définition de l'ensemble de paramètres de description d'une phase.

Paramètres :

a) n° de séquence
* ns = ddd : numéro de la séquence à laquelle appartient la phase

b) n° de phase
* np = ddd : numéro de la phase dans la séquence

c) Signe version pas entier ½ pas uniquement:

*Ss = S+ ou S- : sens de mouvement

Cette information est alors obligatoire, elle précise le sens du mouvement du moteur.

. S+ : sens horaire (sens +)
. S- : sens anti-horaire (sens -)

Remarque :

L'information S+ ou S- n'est pas toujours utilisée par le module, en particulier lors des phases spéciales 'retour origine' et 'mouvement absolu' : le module ne tient pas compte de cette information, mais déduit le sens du mouvement de la position actuelle du compteur et de la consigne du mouvement.

d) Nature de phase
* Na = aa : nature de la phase

permet de spécifier le type de phase à exécuter :

NP : phase mouvement Relatif
NH : phase retour Origine
NX : phase mouvement Absolu
NV : phase à vitesse constante
NA : phase accélération
ND : phase décélération
NW : phase attente
NZ : phase Remise à Zéro du Compteur de Pas Absolu
NG : phase détection glissement
NC : modification de la consigne de vitesse palier
NT : phase activation de la Puissance Moteur
NU : phase suppression de la Puissance Moteur
NY : synchronisation
PV : modification de variable
PC : transfert du CPA dans une variable
PT : test d'une variable
PR : transfert valeur d'init en valeur courante
PI : transfert valeur courante en valeur d'init
PA : addition sur variables

} uniquement pour les modules micropas

- NP, NX : Mouvements relatifs et absolus. La consigne fixe respectivement le nombre de pas à réaliser ou la nouvelle position absolue à atteindre.
Comme pour les mouvements immédiats, ceux-ci sont décomposés en 3 étapes : accélération, palier à vitesse constante, décélération.

- NH : Mouvement de la position actuelle à la position origine. Comme pour NP et NX, le mouvement est également décomposé en trois étapes : accélération, palier à vitesse constante, décélération.

- NV : Mouvement élémentaire. La consigne fixe le nombre de pas à décrire durant la phase.
La vitesse de la phase est la vitesse Vmin, sauf lorsqu'elle s'enchaîne avec des phases de nature NA ou ND dans ce cas la vitesse correspond à la vitesse en fin de la phase précédente.

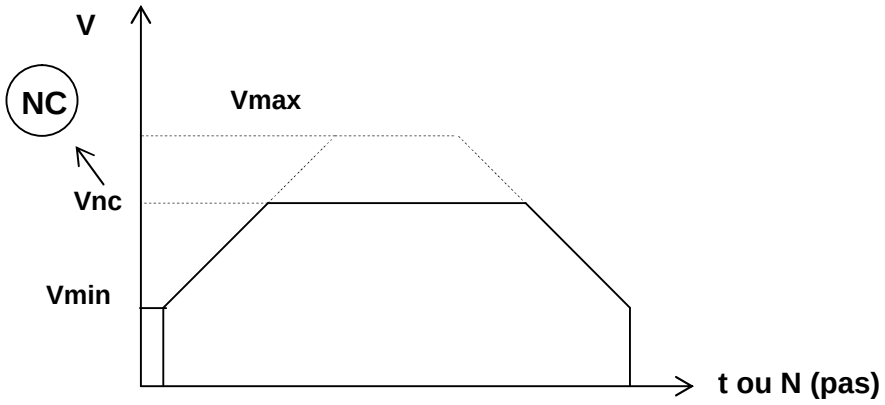
- NA, ND : Mouvements relatifs accélérés ou décélérés. La consigne fixe le nombre de pas (ou micropas à réaliser) en accélérant ou décélérant selon la loi de vitesse en mémoire dans le module. La variation de vitesse s'effectue à partir de la vitesse finale de la phase précédente. La vitesse reste bornée dans tous les cas aux vitesses définies par ωL et ωH.
Lorsqu'une vitesse limite est atteinte, le mouvement continue à vitesse constante jusqu'à la réalisation complète du nombre de micropas défini par la consigne.
Si une entrée logique détectée sur front est activée, la vitesse moteur est maintenue constante sur éventuellement quelques pas (micropas supplémentaires) jusqu'à chargement complet de la phase suivante, puis l'exécution de celle-ci est lancée. De même si le chargement en mémoire pour exécution de la phase suivante n'est pas terminé avant l'accomplissement complet du nombre de micropas défini par la consigne, le mouvement moteur est maintenu à vitesse constante jusqu'à la fin du chargement.
De plus, à vitesse élevée, le comptage des micropas s'effectue par paquets pour des raisons liées au temps de traitement microprocesseur, aussi le nombre de micropas réalisé réellement pourra être légèrement supérieur compte tenu de la résolution des « paquets ». En aucun cas, un paquet ne peut excéder un pas moteur, aussi si les consignes des phases NA, NV, ND correspondent à des nombres de pas entier (multiples de la résolution programmée) aucune erreur de positionnement ne sera ajoutée par cette gestion en « paquets », dans le cas contraire des micropas supplémentaires sont générés

- NW : Phase d'attente permettant de générer un délai en milliseconde.
Cette phase ne provoque aucun mouvement moteur mais les entrées logiques (NE/NF) continuent d'être gérées, permettant ainsi l'attente de l'activation d'une entrée particulière.

- NZ : Remise à zéro du compteur de pas absolu, la position courante devient la position d'origine des prochains déplacements absolus.
Pour les versions micropas, une valeur de consigne différente de zéro peut être donnée, forçant alors un délai d'attente de la valeur donnée exprimée en ms (1 ms par défaut).

- NG : Comparaison de la valeur du compteur de pas absolu avec la valeur de consigne. Si l'écart est supérieur au seuil fixé par la commande DG, la séquence est déroutée sur la phase précisée par la sous-commande NS (obligatoire). Sinon, branchement sur la phase naturelle suivante.
Cette phase ne provoque aucun déplacement moteur. Il n'y a aucune gestion des entrées logiques possibles.
Pour les versions micropas, le numéro de phase à enchaîner peut être différent de la phase naturelle suivante en précisant une deuxième adresse $@_2$ dans la directive NS @[:$@_2$]

- NC : Offre la possibilité d'effectuer les mouvements points à points suivants (dans la séquence ou sous-séquence) à vitesse réduite par modification de la vitesse de palier (V_{max}) tout en conservant la loi d'accélération initiale.



La nouvelle consigne de vitesse Vnc permet de tronquer provisoirement la loi d'accélération initiale.

Une valeur de consigne = 0 permet de rétablir la valeur V_{max} initiale pour les mouvements suivants.
La vitesse initiale V_{max} est automatiquement rétablie en fin de séquence.

Une valeur Vnc inférieure à Vmin impose un fonctionnement uniquement à vitesse min.
Une valeur Vnc supérieure à Vmax restitue le fonctionnement avec la loi de vitesse initiale.

La valeur de consigne Vnc est conservée lors de l'appel d'une sous-séquence et au retour à la séquence appelante.

- NT, NU : Etablissement ou coupure de la puissance moteur. La puissance moteur est implicitement établie en début de séquence comme pour les commandes GA, GH, GM, GO. Elle est supprimée par les commandes GR et MR.

Pour les versions micropas, une valeur de consigne différente de 0 peut être donnée forçant alors une attente de valeur donnée par la consigne exprimée en ms (par défaut 1 ms).

Les natures qui suivent sont spécifiques à la version micropas et n'autorise pas la gestion des entrées logiques, des sous-séquences et des séquences chaînées.

- (NY)µ : Attente Synchro. liaison série (uniquement pour les modules micropas). Cette phase se comporte comme une phase d'attente NW. Elle permet d'attendre la réception d'une commande PG, avec ou sans paramètre, avant de reprendre le cours normal de la séquence. La valeur donnée dans la consigne permet de limiter l'attente et de sauter à la phase définie par NS, en cas de dépassement du temps alloué (en ms).

Exemple : NY time out NS @1.[:@2.]

A la réception de la commande PG, la séquence continue à la phase naturelle suivante (n°phase +1) ou à la phase d'adresse @2 si elle est précisée. Si la commande PG n'est pas reçue avant l'expiration du temps time-out exprimé en ms, la séquence est déroutée sur la phase d'adresse @1.

Remarque : le test des entrées logiques n'est pas pris en compte dans ce type de phase.

- $(PV)_\mu$: Modification de variable (valeur courante)

Forme n°1 : PV # n : v v = ± ddddddddd
la valeur courante de la variable n prend la valeur v

Forme n°2 : PV # n : # m
la valeur courante de la variable n prend la valeur de la variable m

- $(Pi)_\mu$: Modification de la valeur d'initialisation d'une variable Pi # n
La valeur 'courante' de la variable n devient celle qui sera utilisée au MR ou Reset pour initialiser la variable n.

- $(PR)_\mu$: Réaffectation de la valeur d'une variable PR # n
La valeur d'initialisation de la variable n devient la valeur courante.

- $(PC)_\mu$: Recopie du compteur de pas absolu dans une variable PC # n
La valeur instantanée du compteur de pas absolu est recopié dans la variable n

- $(PA)_\mu$: Addition sur variable (valeur courante)

Forme n°1 : PA # n : v v = ± ddddddddd
la valeur v est additionné à la valeur courante de la variable n : # n = # n + v

Forme n°2 : PA # n : # m
la valeur courante de la variable m est additionné à la valeur courante de la variable n : # n = # n + # m

- $(PT)_\mu$: Test sur variable

La nature PT permet de conditionner le déroulement de la séquence à la valeur courante d'une variable ou de gérer un "compteur de boucle".
Dans tous les cas, il convient d'associer à la nature PT la directive NS dans sa forme étendue : NS @1 [: @2 [: @3]] qui permet de décrire jusqu'à 3 numéros de phase suivante.

Forme n°1 : PT # n : v NS @1 [: @2 [: @3]] v = ± ddddddddd
Compare la valeur courante de la variable n à la valeur v et sélectionne en conséquence la phase suivante à exécuter.

@1	si	# n = v (égalité)
@2	si	# n < v (infériorité)
@3	si	# n > v (supériorité)

Note : si les adresses @2 et @3 ne sont pas précisées par la directive NS elles sont remplacées par le numéro de la phase naturelle suivante (phase +1).

La variable n n'est pas modifiée

Forme n°2 : PT # n : # m NS @1 [: @2 [: @3]]
Compare la valeur courante de la variable n à la valeur courante de la variable m et sélectionne en conséquence la phase suivante à exécuter.

@1	si	# n = # m	(égalité)
@2	si	# n < # m	(infériorité)
@3	si	# n > # m	(supériorité)

Note : si les adresses @2 et @3 ne sont pas précisées par la directive NS elles sont remplacées par le numéro de la phase naturelle suivante (phase +1).

Les variables n et m ne sont pas modifiées.

Forme n°3 : (gestion compteur) PT # n NS @1 [: @2]
La valeur courante de la variable n est décrémentée de 1 : (# n = # n – 1)
Elle est alors comparée à la valeur 0 et la phase suivante est sélectionnée en conséquence.

@1	si	# n = 0	(# n après décrémentation)
@2	si	# n ≠ 0	

Note : l'absence de l'adresse @2 dans la directive NS entraînera la sélection de la phase naturelle suivante (phase +1)

! La variable # n est modifiée

e) Consigne

* Cns = [±]ddddddddd : consigne

Cette consigne peut être :

- un nombre de pas relatif dans les phases :

NV = vitesse constante sur Cns pas, ½ pas ou micropas (selon la résolution du module)
NP = mouvement de Cns pas, ½ pas ou micropas (selon la résolution du module)

Dans la version pas entier ½ pas, le déplacement doit être compris entre 1 et 999999.
Le sens du mouvement est défini par la commande S+/S-

Dans la version micropas, le sens du mouvement est déterminé par le signe de la consigne sa valeur est bornée
-2147483647 ≤ Cns ≤ 2147483647

- une position absolue

Le paramètre représente la position que doit atteindre le moteur pour les phases NX ou une position de référence pour les phases NG (en pas, ½ pas ou micropas, selon la résolution du module).

Format numérique signé ou non	-8388606 ≤ Cns ≤ 8388605	version pas entier ½ pas
	-2147483647 ≤ Cns ≤ 2147483647	version micropass

- un délai

Pour la phase NW (attente) et en version micropas, pour les phases NZ, NU, NT, NY; le délai est donné en ms (1 à 65535 ms).

- une vitesse

En pas/s (ou ½ pas/s) pour les phases NC

- forcée à 000, pour la version pas entier ½ pas

Dans les phases NU, NT, NZ et NH.

Les phases NU, NT et NZ n'effectuent aucun mouvement. Pour la phase NH le module recherche le nombre de pas et le sens du mouvement avant de l'exécuter.

- Consigne variable :

En version micropas, la valeur de la consigne peut être donnée par la valeur d'une variable en donnant son numéro n précédé de # : # n.

Attention ! Une consigne est toujours obligatoire pour la version pas entier ½ pas

f) Séquence chaînée (Directive NL)
* sc = ddd : numéro de séquence chaînée (optionnel)

Cette information précise le numéro de la nouvelle séquence à exécuter dès que la séquence en cours sera terminée.
Par défaut, le numéro de la séquence à chaîner déjà défini n'est pas modifié s'il n'est pas précisé par la phase en cours et par voie de conséquence, il n'y a pas de chaînage de séquence tant que le module ne rencontre pas cette donnée dans la description d'une phase qu'il exécute.
Pour modifier ou définir un numéro de séquence à chaîner, la phase contenant l'ordre NL doit obligatoirement être exécutée (pas seulement définie).

g) Sous-séquence (Directive NQ)
*sq = ddd : numéro de sous-séquence (optionnel)

Lorsque l'exécution de la phase en cours est terminée, le module interrompt le mouvement moteur, recherche et charge la sous-séquence Sc, l'exécute entièrement et reprend l'exécution de la séquence initiale à partir de la phase naturelle suivant la phase ayant provoqué le branchement à la sous-séquence.

Une sous-séquence se différencie d'une séquence uniquement par le fait :

- qu'elle ne peut appeler d'autres sous-séquences (un seul niveau d'appel),
- qu'elle ne peut enchaîner de séquence (NL).

Les paramètres NE/NF et NS ne sont pas pris en compte lors de l'exécution d'une phase d'appel de sous-séquence (pas d'appel conditionnel).

Les directives NL et NQ sont exclusives.

h) Test des entrées logiques (Directives NE et NF)

*Ne : aa_ddd_ddd_ddd_ddd_ddd_ddd_ddd_ddd : sélection des entrées conditionnelles	(optionnel)

NE X_1 X_2 X_3 X_4 X_5 X_6 X_7 X_8 : gestion sur état
ou NF X_1 X_2 X_3 X_4 X_5 X_6 X_7 X_8 : gestion en cours de phase

Par défaut, il n'y a pas de gestion des entrées logiques

Le code NE impose la prise en compte des entrées logiques en fin de phase uniquement.
Seul compte l'état (actif/inactif) des entrées en fin de phase, toute variation (actif <--> inactif --> actif) pendant le déroulement de la phase est ignorée.

Le code NF impose un traitement immédiat des entrées logiques dès l'activation d'une entrée en cours de phase ou en début de phase si l'entrée est déjà active.

Les paramètres X_1 X_2 X_3 X_4 X_5 X_6 X_7 X_8 précisent les huit numéros de phases conditionnelles rattachées aux huit entrées logiques, respectivement E1, E2, E3, E4, E5, E6, E7, et E8.
Ces huit paramètres doivent obligatoirement être donnés tous les huit dès que l'on fait appel aux sous-commandes NE ou NF.

Xi précise le numéro de phase à exécuter si l'entrée correspondante est active. Si Xi vaut 0 l'entrée associée n'est pas prise en compte.

Pour les versions micropas, il est possible de choisir individuellement l'état actif de chaque entrée en faisant précéder du signe – l'adresse du saut associé à une entrée, c'est l'état logique 1 qui devient actif au lieu de l'activité nominale à l'état logique 0.
Cette modification du niveau logique actif de l'entrée n'est valable que pour la phase où elle est précisée et elle se combine avec la polarité globale définie par les commandes MB et MN.

Dans le cas de l'activation simultanée de plusieurs entrées, la scrutation des entrées est effectuée dans l'ordre croissant.

Exemple

1 * aiguiller la séquence 1 sur la phase n° 7 dès l'apparition de l'entrée logique n° 3 lors de l'exécution de la phase 2 :

(E1)(E2)(E3)(E4)(E5)(E6)(E7)(E8)

SP_1_2 ... NF_00__00__07__00__00__00__00__00

La valeur 00 indique que l'entrée logique correspondante n'est pas prise en compte.

2 * aiguiller la séquence 3 sur la phase n° 5 si l'entrée logique n° 5 est présente en fin de phase 4.

SP_3_4 ... NE_00_00_00_00_05_00_00_00_00.....

i) Phase suivante

* ps = dd : n° de la phase suivante	(optionnel)

Permet de forcer l'ordre de succession des phases en imposant un numéro de phase suivant, différent de l'ordre chronologique.
Cette information est optionnelle : par défaut la phase naturelle suivante est le numéro de la phase en cours + 1.

Exemple : Provoquer un saut de la phase n° 3 à la phase n° 5.

SP_1_3..... NS_05

Remarque 1

En combinant les codes NE/NF et NS, il est possible de répéter une phase jusqu'à ce qu'une condition logique soit active.

SP_1_2 ... NE_03_00_00_00_00_00_00_00_NS_02.

Cette phase reboucle sur elle-même (NS02) tant que la condition logique E1 est inactive (en fin de phase).

Elle passe ensuite à la phase n° 3.

Remarque 2

Il est possible de contrôler dans la même phase plusieurs entrées logiques mais, dans l'hypothèse où celles-ci sont positionnées simultanément, le module traitera uniquement l'entrée dont le numéro est le plus faible.

SP_1_2 ... NE_00_00_05_00_17_00_07_00_NS_02

Si les entrées 5 et 7 sont actives en fin de phase 2, la séquence continuera en phase 17. Avec une commande de type NF, le saut dépend de la réelle "simultanéité" de l'activation de deux entrées. Il revient à l'utilisateur de bien maîtriser le timing de ses entrées quand il utilise la commande NF s'il ne veut pas être confronté à des aléas parfois difficilement maîtrisables.

j) Sorties logiques (Directives NO)

*out = hh : état à placer sur les sorties logiques msq = hh : masque des sorties modifiables	} (optionnel)

Cette dernière partie de la commande permet de configurer l'état des sorties logiques du module durant la phase décrite.
Par défaut les sorties logiques restent dans l'état donné par la phase précédemment exécutée.
L'état des sorties prend la valeur $S = OUT * msq + S_{-1}\,\overline{msq}$ où S_{-1} représente l'état préalable des sorties.
Si le masque n'est pas précisé, l'ensemble des 8 bits de sortie est modifié.
L'état et le masque peuvent être précisés au moyen d'une seule variable.
Les 2 digits de poids faible hexadécimaux donne la valeur de sortie OUT, les deux digits de rang supérieur, la valeur du masque si ils sont différents de 0.

n = h h h h

msq ↵ ↳ OUT

Attention! ni la notion de masque, ni la notion de variable ne sont utilisables dans les modules pas entiers ½ pas.

Exemple

NO_A5 : place les sorties logiques dans l'état :

1010 0101 = (A5H)
S8.....S1

***Restrictions* : si cette information existe dans la commande de définition de la phase, elle doit apparaître obligatoirement en fin de commande / SP.**

Les cartes pas entier ½ pas ne reconnaissent pas la notion de masque, l'utilisation d'un masque provoque une erreur de syntaxe.

k) Codes d'erreurs

Le module effectue plusieurs contrôles sur la cohérence et sur la syntaxe de la commande SP et positionne les codes suivants :

0 : défaut sur les paramètres de la phase :
les paramètres ne sont pas numériques.

1 : paramètre n° de séquence incorrect.
Le numéro précisé ne correspond pas à une séquence déjà existante ou le numéro de séquence est hors limite

2 : paramètre n° de phase incorrect.
Le numéro de la phase est supérieur au nombre de phases réservées pour la séquence (défini par la commande SN).

4 : paramètre entrées logiques incorrect.
Les huit numéros de phase conditionnelle ne sont pas tous numériques.

Attention !!!

Le contrôle du signe n'est effectué que sur le premier paramètre de la commande.

SR : Suppression de la séquence de démarrage **SR**

Syntaxe : [@]SR

Paramètres : aucun

Description : suppression du démarrage automatique de séquence sélectionnée par la commande SD.

Cette commande efface uniquement l'information qui permettait au module de trouver la séquence à exécuter à la mise sous tension.

La séquence elle-même n'est pas effacée.

Pas de paramètre.

Code d'erreur positionné :

0 : Présence d'un paramètre

SS : Lancement d'une séquence **SS**

Syntaxe : [@]SS_ns

Paramètres : ns = ddd : n° de la séquence à lancer

Numérique,
Non signé,
Limite du paramètre : 1 ≤ N° de séquence ≤ 99 version pas entier ½ pas
1 ≤ N° de séquence ≤ 100 version micropas

Description : exécution immédiate de la séquence demandée.

S'il s'agit de la première exécution de la séquence après modifications des paramètres vitesse, le module finit, si nécessaire, le calcul de la loi d'accélération puis exécute la séquence.

Le paramètre N° de Séquence doit correspondre à une séquence connue du module.

Exemple : Exécuter directement la séquence 15 :

SS_15.

Code d'erreur positionné

Le module positionne les codes suivants :

0 : Défaut sur le paramètre
Format incorrect
Nombre de paramètre incorrect

1 : Paramètre N° de Séquence hors limite

3 : Le n° de Séquence fourni ne correspond pas à une séquence connue du module.

WH : vitesse de consigne **WH**

Syntaxe : [@]WH_w

Paramètres : w = ddddd : vitesse de consigne en p/s ou en ½ p/s

Numérique,
Non signé,
5 digits maximums.
Limites absolues du paramètre : $1 < w \leq 20\,000$

Description : programmation de la vitesse de consigne de tous les prochains mouvements moteur même après réinitialisation.

Cette commande permet de donner au paramètre Vmax la valeur w.

Vmax correspond à la vitesse maximum que pourra prendre le moteur, elle ne dépend pas de la résolution en micropas par pas choisie.

Cette valeur w est donnée en pas/s pour les modes pas entier ou micropas, en ½ p/s pour le mode ½ pas.

Les limites absolues de ce paramètre sont 1 pas/s et 20000 pas/s.

Exemple : soit à fixer la vitesse de consigne à 13500 p/s

WH 13500

Code d'erreur positionné

Le module positionne les codes d'erreur suivants :

0 : La commande WH n'a pas de paramètre.
Le paramètre fourni ne correspond pas à une valeur numérique.
Le paramètre fourni est signé.

1 : La cohérence entre les paramètres V_{min}, V_{max}, T_r et µ n'est pas respectée.
(cf. § III.4.6.)

Remarque :

Le paramètre est mémorisé non pas sous la forme fréquence (pas/s) mais en format période en 0,5 µs ($2 * 10^6$/fréquence).

Une conversion Fréquence_Période est donc assurée avant la mise à jour du paramètre.

La relecture du paramètre par les commandes QL ou RL effectue l'opération inverse (conversion Période_Fréquence).

Du fait des arrondis de calcul (par troncature), le résultat de la relecture peut donc être différent de la programmation mais correspond à la fréquence réellement générée.

WL : vitesse de démarrage **WL**

Syntaxe : [@]WL_w

Paramètres : w = ddddd : vitesse de démarrage en p/s ou ½ p/s

Numérique,
Non signé
5 digits maximums
Limites absolues du paramètre : $1 \leq w < 20\,000$

Description : programmation de la vitesse de démarrage de tous les prochains mouvements moteur, même après réinitialisation.

Cette commande permet d'initialiser le paramètre Vmin avec la valeur de w.

Elle ne dépend pas de la résolution en micropas choisie.

Cette valeur w est donnée en p/s pour les modes pas entier ou micropas, en ½ p/s pour le mode ½ pas.

Vmin correspond également à la vitesse minimum du moteur.

Exemple : soit à fixer la vitesse de démarrage à 250 p/s : WL 250

Code d'erreur positionné

Le module positionne les codes d'erreurs suivants :

0 : La commande WL n'a pas de paramètre.
Le paramètre fourni ne correspond pas à une valeur numérique.
Le paramètre fourni est signé.

1 : La cohérence entre les paramètres V_{min}, V_{max}, T_r et µ n'est pas respectée.
(cf. § III.4.6.)

(commande réservée aux modules micropas)

(WN)µ : Résolution en micropas **(WN)µ**

Syntaxe : [@]WN_µ

Paramètres : µ = ddd : résolution en micropas par pas

Numérique,
Non signé,
3 digits maximums
Limites : $1 \leq \mu \leq 256$ pour les modules micropas sans puissance intégrée
(SIMPA micropas ou SIMPA micropas Face Avant,…)
µ : 1, 2, 4, 8, 16, 32 ou 64 pour les modules micropas avec puissance intégrée
(SIMPA micropas 1 axe, SIMPA micropas 1 axe Face Avant,
SIMPA micropas 4 fils ou SIMPA micropas 4 fils Face Avant,…)

Description : définition de la résolution du driver en micropas par pas. Les vitesses maximums et minimums, ainsi que le temps de rampe ne sont pas modifiés.

On peut donc modifier la résolution sans que la vitesse finale du moteur ne soit modifiée.

Cependant la vitesse de génération des micropas, quant à elle, est multipliée par µ.

Dés lors, compte tenu de la quantification liée à la résolution du générateur, les vitesses réellement générées (retournées par la commande QL) peuvent légèrement évoluer en fonction de la résolution choisie.

Exemple : soit fixer la résolution du driver à 32 micropas par pas : WN 32

Code d'erreur positionné

0 : La commande WN n'a pas de paramètre.
Le paramètre fourni ne correspond pas à une valeur numérique.
Le paramètre fourni est signé.

1 : La valeur donnée du paramètre est hors limite :

La cohérence entre les paramètres V_{min}, V_{max}, T_r et µ n'est pas respectée (cf. § III.4.6.)

Cette commande n'existe pas en version pas entier ½ pas

Note :
- Dans le cas d'un couple indexeur micropas / amplificateur , il faut que la résolution programmée par WN corresponde à la résolution sélectée (roue codeuse, cavaliers …) sur l'amplificateur.

WT : temps de rampe **WT**

Syntaxe [@]WT_ta [:td]

Paramètres : ta = ddddd : durée de la rampe d'accélération en ms.
td = ddddd : durée de la rampe de décélération (uniquement pour version micropas)
Numériques,
Non signés,
5 digits maximums
Limites : $1 < t_{ms} < 65535$ ms

Description : programmation des temps d'accélération ta et de décélération Td de tous les prochains mouvements moteur.

Ta = ta Td = td

Ta correspond au temps imposé pour passer la vitesse moteur de Vmin à Vmax et inversement Td au temps demandé pour revenir de Vmax à Vmin.

Lorsque la valeur td est omise, les temps d'accélération et décélération sont forcés à la même valeur
Ta = Td = ta

Les modules pas entier ½ pas n'admettent pas de paramètres td, les temps d'accélération et de décélération sont donc forcément égaux.

Exemple : programmation du temps de rampe avec t = 500 ms :

WT 500.

Code d'erreur positionné

Le module positionne les codes d'erreurs suivants :

0 : La commande WT n'a pas de paramètre.
Le paramètre fourni ne correspond pas à une valeur numérique.
Le paramètre fourni est signé.

1 : La cohérence entre les paramètres V_{min}, V_{max}, T_r et μ n'est pas respectée.
(cf. § III.4.6).

VII - GESTION PAR INTERRUPTIONS

Les modules SIMPA disposent d'un protocole interruption spécifique permettant la transmission rapide d'une information vers le calculateur hôte sans que celui-ci soit obligé de scruter en permanence l'ensemble des modules qu'il contrôle.

VII.1 - Généralités

Tout module peut émettre à tout moment une demande de service (interruption) vers le calculateur à la seule condition qu'il y ait été préalablement autorisé par une autorisation générale ou une autorisation par groupe (voir paragraphe VII.4 ci-après).

Trois événements sont susceptibles de provoquer la génération d'une interruption de la part d'un module :

- absence d'alimentation de puissance,
- défaut moteur (court-circuit, circuit ouvert...),
- accès à la phase 255 (55 en version pas entier ½ pas)

Ce dernier événement permet à l'utilisateur d'être immédiatement informé d'un enchaînement particulier de phases et donc de détecter des configurations spécifiques qu'il aura prévues (exemple : détection de butées, absence repère, etc.)

VII.2 - Support matériel

Les interruptions utilisent la liaison série RS232 comme support de transmission. Elles sont matérialisées par la génération d'un "BREAK" sur la ligne série (mise à 0 de la liaison pendant un temps supérieur à la durée d'émission d'un caractère).

VII.3 - Protocole

La gestion des interruptions obéit à un protocole totalement indépendant du protocole normal de communication avec les modules. Tous les adressages se font grâce à un seul caractère afin de minimiser les temps de réponse.

La réception de tout autre caractère que ceux de ce protocole par un module, le replace dans le mode de dialogue classique et arrête la gestion du processus d'interruption.

La gestion des interruptions comporte 3 phases :

- une phase d'autorisation des interruptions qui permet aux modules de faire connaître leurs demandes éventuelles,

- une phase de scrutation qui permet de localiser les modules interrupteurs,

- une phase d'acquittement qui permet d'indiquer au module interrupteur que sa demande a été prise en considération.

VII.4 - Autorisation des interruptions

Autorisation générale :

Caractère : 1Bh (00011011 b)

Tous les modules connectés à la liaison série sont autorisés à émettre une interruption (BREAK).

Autorisation par groupe

Il est possible de n'autoriser que certains modules à émettre une interruption.

La commande à la forme :

Caractères : C0h à FFh (11mmmmmm b)

Les bits 0 à 6 (masque d'autorisation) permettent de spécifier les modules autorisés à transmettre une interruption.

Tout module dont l'adresse est tel que : (adresse) ET (masque) = masque est autorisé à émettre une interruption ET logique sur 6 bits

Les autres modules ne peuvent pas émettre d'interruption.

Exemple : commande 1 1 0 0 0 1 1 0 (C6h)
masque 0 0 0 1 1 0
modules x x x 0 x x pas autorisés
x x x x 0 x pas autorisés
x x x 1 1 x autorisés

VII.5 - Scrutation des interruptions

Une interruption arrivant au calculateur lui signale qu'un module a émis une interruption mais ne permet pas de connaître le module générateur. L'autorisation par groupe ne permet de spécifier que des groupes de deux modules au minimum.

Exemple : masque 0 1 1 1 1 1
modules du groupe { 0 1 1 1 1 1 (31)
1 1 1 1 1 1 (63)

La réception d'un break indique qu'un ou plusieurs modules demandent un service.

Après une autorisation générale, des autorisations successives de groupe permettent de déterminer plus rapidement que par des interrogations individuelles le ou les modules en cause (approche dichotomique).

Seules six autorisations de groupe sont nécessaires pour déterminer un module interrupteur quel que soit son numéro parmi 63.

Pour permettre la recherche dichotomique, l'état des interruptions est figé pendant toute la phase de scrutation. Ainsi, les modules ne peuvent émettre une interruption vers le calculateur pendant les autorisations partielles que s'il sont adressés et que s'ils ont présenté leur interruption lors de la dernière autorisation générale.

Ce protocole permet en outre de gérer une certaine forme de chronologie dans les interruptions puisqu'il faut autoriser à nouveau l'ensemble des interruptions pour pouvoir prendre en compte de nouvelles demandes.

Remarque : le nombre de scrutations nécessaires diminue avec le nombre de modules interconnectés à condition de leur affecter des numéros successifs à partir de 0.

Le nombre de modules connectés : N

Le nombre de scrutations nécessaires : n

$2^{n-1} < N < 2^n$

Exemple :

Les modules 9(Ø9h) , 45(2Bh) et 6(Ø6h) passent successivement en phase 55.

Evolution des dialogues entre le calculateur hôte et les modules SIMPA

→	Emission autorisation générale (1Bh) vers tous les modules
	←——— le module 9 passe en phase 55
	←——— le module 45 passe en phase 55 break masqué par le break précédent
→	Première scrutation (C1h). **Inhibe tout envoi de break** par tout autre module hors 9 et 45 masque 000001
	← break module 9 et 45 le module 6 passe en phase 55 (envoi break inhibé)
→	Deuxième scrutation (C3h) masque 000011 and (001001 OR 101101) ≠ masque Pas de break
→	Troisième scrutation (C5h) masque 000101 ← break module 45 (45 = 101101)
→	Quatrième scrutation (CDh) masque 001101 ← break module 45
→	Cinquième scrutation (DDh) masque 011101 Pas de break
→	Sixième scrutation (EDh) masque 101101 ← break module 45 101101

Temps

A ce stade, on remarque que bien que le module 9 soit signalé avant le module 45, la recherche dichotomique détecte le module 45 en premier.

La recherche des autres modules n'est poursuivie qu'après acquittement du module 45.

VII.6 - Acquittement des interruptions

Tout module détecté en interruption doit être acquitté. Cet acquittement peut se faire de deux façons :

a) par commande d'acquittement : 80h à BFh (10aaaaaa b)

aaaaaa : adresse binaire du module à acquitter

Exemple : pour acquitter le module n° 45 (101101) envoi de la commande 10101101 soit (ADh)

De plus, cette commande force les autres modules interrupteurs avant la première scrutation (module 9 dans l'exemple) à confirmer leur demande de service.

C'est un moyen rapide pour :

- acquitter l'interruption sur un premier module (45 dans l'exemple) ;

- déterminer si d'autres modules sont en demande de service simultanément (module 9 dans l'exemple).

b) par interrogation de l'état du module

Une interrogation normale permet de récupérer la cause de l'interruption (voir commande RX ou QX) tout en l'acquittant.

Exemple : pour le module 45

Commande : 45RX
en retour : 45EE_S

Remarque : Dans tous les cas, l'état du module (S) est maintenu jusqu'à la lecture par la commande RX ou QX, sauf écrasement par un autre défaut même si l'acquittement est effectué par la commande d'acquittement directe.

La deuxième méthode ne permet pas de figer l'état des demandes à un instant donné, puisqu'elle rompt le protocole interruption en revenant au dialogue normal.

VII.7 - Poursuite des scrutations

La première série de scrutation a permis d'isoler le module 45.

L'acquittement de ce module par la commande d'acquittement signale qu'il existe d'autres modules en interruption.

Il est donc possible de relancer une série de six scrutations pour isoler un deuxième module.

→	Acquittement du module 45 (ADh)
	← il existe d'autres modules en défaut (présence break)
→	Première scrutation : C1h
←	break (du module 9)
→	Deuxième scrutation : C3h
→	Troisième scrutation : C5h
→	Quatrième scrutation : C9h
	← break (du module 9)
→	Cinquième scrutation : D9h
→	Sixième scrutation : E9h
	Module 9 détecté en défaut
→	Acquittement module 9

L'acquittement du module n° 9 révèle qu'il n'y a plus de module en interruption (absence de break en réponse).

A ce stade, tous les modules interrupteurs avant la première scrutation ont été isolés et acquittés (45 et 9).

L'utilisateur peut alors intervenir sur ces seuls modules s'il le désire (recherche cause d'erreur par exemple).

Pour localiser le module 6, (la demande d'interruption est intervenue après la première scrutation) il suffit d'autoriser à nouveau les modules de façon générale.

→	autorisation générale
	← break (du module 6)
→	Six scrutations pour isoler le module 6
→	acquittement module 6

Remarque :

Une série de six scrutations qui n'entraîne aucune réception de break signifie que la demande de service provient du module 0.

Il appartient à l'application d'assurer la gestion des interruptions (détection, recherche module en cause, acquittement).

Ces opérations répétitives sont entièrement prises en charge avec toutes les autres fonctions de mise en œuvre des modules par le logiciel LIBSIM développé par MIDI INGENIERIE pour faciliter la télégestion des modules à partir d'un PC ou compatible.

Algorithme de scrutation des interruptions

Exemple :

Détection de l'interruption du module n° 45 (20h)

Autorisation générale de break
Cde : 1 Bh

.....

scrutation niveau 1
Cde : 11xxxxx1b

.....

scrutation niveau 2
Cde : 1100001xb

.....

scrutation niveau 3
Cde : 110001xxxb

.....

scrutation niveau 4
Cde : 11001xxxb

.....

scrutation niveau 5
Cde : 1101xxxxb

.....

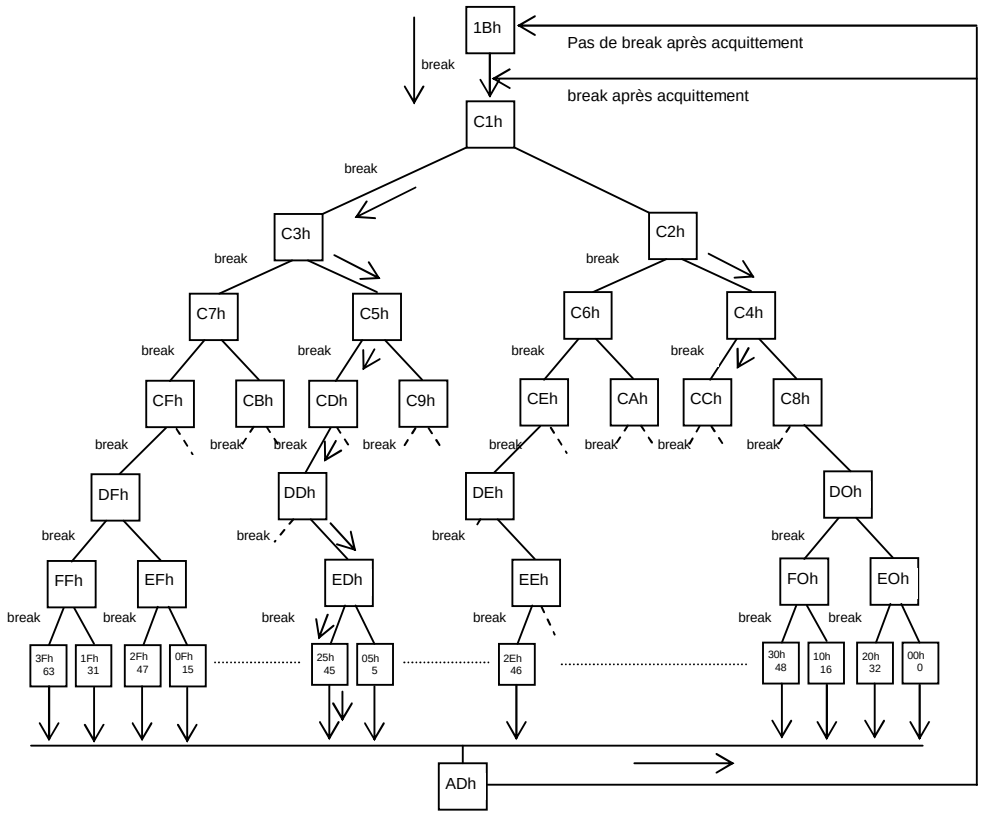
scrutation niveau 6
Cde : 111xxxxxb

.....

module détecté

.....

acquittement du module détecté

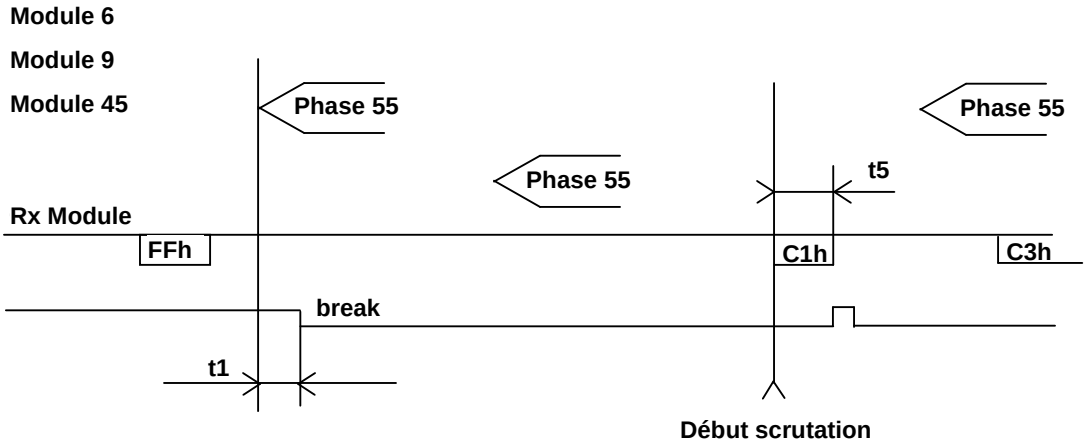


VII.8 - Timing des interruptions

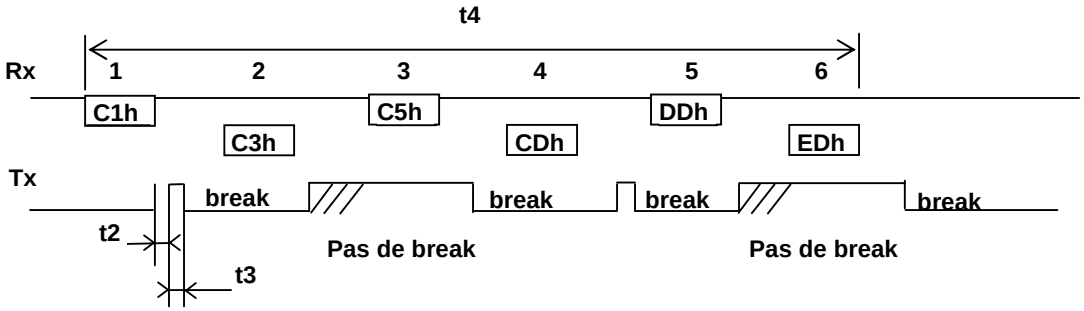
Des considérations de temps sont à prendre en compte lors du traitement des interruptions.

	à 4800 bauds minimum	à 9600 bauds	unité
Délai de réponse des modules à l'autorisation générale	3	1,5	ms
Délai de reconnaissance du break pour l'UART du PC	2,5	1,25	ms
Durée minimum d'un cycle de 6 scrutations	31	17	ms
Durée d'émission d'un caractère à 4800 bauds	2,05	1,03	ms

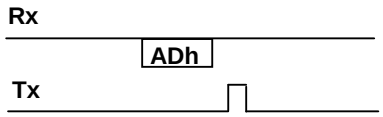
Chronogramme :



Première série de 6 scrutations



Acquittement module 45



		à 4800 bauds	à 9600 bauds	unité
Délai d'émission	t1	< 20 µs		µs
Délai de relâchement de la ligne Tx	t2	< 30 µs		µs
Durée d'émission du break en interrogation	t3	= 500 µs		µs
Temps de scrutation pour 63 modules	t4	31	17	ms
Durée d'un caractère	t5	2,05	1,3	ms

REMARQUE

- L'acquittement des axes aurait pu être obtenu par envoi des commandes (à l'aide du protocole standard).

Commande	45RX		pour l'axe 45
Réponse		45EE_S	
Commande	09RX		pour l'axe 09
Réponse		09EE_S	

- Bien que l'axe 9 ait demandé une interruption avant l'axe 45 c'est ce dernier qui est reconnu en premier par dichotomie.

Par contre, si le début de la scrutation intervient avant la demande de l'axe 6, pour localiser cet axe il sera nécessaire de refaire une autorisation générale suivie d'une nouvelle scrutation.

LISTE DES COMMANDES SIMPA (Version pas entier ½ pas)

* [@]MR		Reset général du module, sorties à FF (implicite à mise sous tension)
* [@]MB		Sélection du mode de fonctionnement Butée
* [@]MN		Annulation du mode de fonctionnement Butée
[@]WH	V_{max}	Vitesse de consigne (20 à 20000 p/s) palier
[@]WL	V_{min}	Vitesse de démarrage (20 à 20000 p/s)
[@]WT		Durée rampe accélération en ms (1 à $65x10^6$/Vmax)
[@]D+		Sens horaire pour prochains déplacements relatifs
[@]D-		Sens anti-horaire pour prochains déplacements relatifs
[@]DG	n	n pas (ou demi pas) tolérés en glissement (<256 pas)
[@]DI		Référence 0 des prochains déplacements absolus
[@]DP	Pr	Référence Pr pour les prochains mouvements absolus
[@]DR	n	nb pas prochains déplacements relatifs(0 à 999998)
[@]GA	Pa	Exécution d'un mouvement absolu, mode direct (Pa) (-8388606 < Pa < 8388606)
* [@]GE		Décélération puis arrêt d'un mouvement ou d'une séquence
[@]GF		Exécution d'un mouvement continu
[@]GH		Retour origine
[@]GI	Im	Imoteur = Inom * Im/255
* [@]GL	Out	Positionnement des sorties logiques (Out : en hexa)
* [@]GM		Puissance moteur (implicite avec GO, GA, GH et exécution séquence)
[@]GO		Exécution d'un mouvement relatif en mode direct (D+/- n pas)
* [@]GR		Coupure puissance moteur
* [@]GS		Stop (arrêt immédiat d'un mouvement ou d'une séquence)
[@]SA	ns	Exécution séquence si condition E2 (front)
[@]SC	ns	Sélection séquence de démarrage conditionnelle (E2 après RESET)
[@]SD	ns	Sélection séquence de démarrage (exécutée sur RESET)
[@]SE	ns	Effacement séquence (ns = 0 : effacement de toutes les séquences)
[@]SN	ns np	Création séquence ns (1 à 99) de np phases (1 à 50)
[@]SP	ns np S+/-/I Na Cns [NL sc ou NQ ss] [Nex1x2...x8] [NS ps] [NO out]	
		Définition phase np séquence ns
		Na : Nature
		NA accélération — NT puissance moteur ON
		NC modification vitesse palier — NU puissance moteur OFF
		ND décélération — NV vitesse constante
		NG détection glissement — NW attente
		NH retour origine (HOME) — NX mouvement absolu
		NP mouvement relatif — NZ RAZ position absolue
		Cns : consigne (nombre de pas, position, vitesse palier ou délai, suivant nature)
		NL sc : numéro séquence choisie pour séquence suivante
		NQ ss : appel sous-séquence (NL et NQ sont exclusifs)
		Ne : NE ou NF saut de phase sur état ou front entrées logiques
		NS ps : phase suivante
		NO out : positionnement sorties logiques
[@]SR		Suppression de la sélection séquence de démarrage
[@]SS	ns	Exécution de la séquence ns
* @ RD		Suivi des séquences et mouvements
* @ RL		Lecture paramètres locaux
* @ RP		Lecture compteur de pas, états E/S
* @ RS	ns np	Lecture de la phase np de la séquence ns
* @ RV		Lecture version et indice du logiciel
* @ RX		Lecture code état module

@	adresse du module	*	commande utilisable quelque soit l'état du module
[@]	commandes multimodules	[Cmde]	commande optionnelle

LISTE DES COMMANDES SIMPA (Version micropas)

*	[@]MR		Reset général du module, sorties à FF (implicite à mise sous tension)
*	[@]MB	[H]/[L]	Sélection du mode de fonctionnement Butée
*	[@]MN	[H]/[L]	Annulation du mode de fonctionnement Butée
	[@]MS	[N][S][B]	Mode de gestion du courant N :nom/S :Nom + Stb/B :Nom + Stb + Boost
	PG		Gestion variables, valeur à initialiser
	PD		Gestion variables, valeur d'initialisation décimale
	PH		Gestion variables, valeur d'initialisation héxadécimale
	[@]WH	V_{max}	Vitesse de consigne (20 à 20000 p/s) palier
	[@]WL	V_{min}	Vitesse de démarrage (20 à 20000 p/s)
	[@]WT	t_r	Durée rampe accélération en ms (1 à $65x10^6$/Vmax)
	[@]WN	µ	Résolution en micropas par pas (uniquement en mode micropas)
	[@]DG	n	n micropas tolérés en glissement (<256 pas)
	[@]DI		Référence 0 des prochains déplacements absolus
	[@]DP	Pr	Référence Pr pour les prochains mouvements absolus
	[@]GA	Pa	Exécution d'un mouvement absolu ($-2^{31}-1 < Pa < 2^{31}-1$)
*	[@]GE		Décélération puis arrêt d'un mouvement ou d'une séquence
	[@]GF	[+]/[-]	Exécution d'un mouvement continu
	[@]GH		Retour origine
	[@]GI	Im	Imoteur = Inom * Im/255
*	[@]GL	Out	Positionnement des sorties logiques (Out : en hexa)
*	[@]GM		Puissance moteur (implicite avec GO, GA, GH et exécution séquence)
	[@]GO	[±]/[n]	Exécution d'un mouvement relatif (± n micropas < $2^{31}-1$)
*	[@]GR		Coupure puissance moteur
*	[@]GS		Stop (arrêt immédiat d'un mouvement ou d'une séquence)
*	[@]GP	P[I]	Type : gestion Sortie Logique déportée
	[@]SD	ns	Sélection séquence de démarrage (exécutée sur RESET)
	[@]SE	ns	Effacement séquence (ns = 0 : effacement de toutes les séquences)
	[@]SN	ns np	Création séquence ns (1 à 200) de np phases (1 à 200)
	[@]SP	ns np Na Cns [NL sc ou NQ ss] [Nex1x2...x8] [NS ps] [NO out]	
			Définition phase np séquence ns
			Na : Nature
			NC modification vitesse palier
			NG détection glissement — NV vitesse constante
			NH retour origine (HOME) — NW attente
			NP mouvement relatif — NX mouvement absolu
			NT puissance moteur ON — NZ RAZ position absolue
			NU puissance moteur OFF — PV modification de variable
			PI modification valeur d'initialisation d'une variable
			PR modification de la courante d'une variable en valeur d'init
			PA addition sur variable — PC transfert CP1 variable
			PT test variable
			Cns : consigne (nombre de micropas, position, vitesse palier ou délai, suivant nature)
			NL sc : numéro séquence choisie pour séquence suivante
			NQ ss : appel sous-séquence (NL et NQ sont exclusifs)
			Ne : NE ou NF saut de phase sur état ou front entrées logiques
			NS ps : phase suivante
			NO out : positionnement sorties logiques
	[@]SR		Suppression de la sélection séquence de démarrage
	[@]SS	ns	Exécution de la séquence ns
*	@ QD		Suivi des séquences et mouvements
*	@ QL		Lecture paramètres locaux
*	@ QP		Lecture compteur de micropas, états E/S
*	@ QS	ns np	Lecture de la phase np de la séquence ns
*	@ QV		Lecture version et indice du logiciel
*	@ QX		Lecture code état module
*	@ QC		Relecture paramètre — sortie logique déportée (GL/GP)
*	@ QN	#n[H]	Relecture variable

@	adresse du module	*	commande utilisable quel que soit l'état du module
[@]	commandes multimodules	[Cmde]	commande optionnelle