

# Manuel d'utilisation du module MICROSIMPA 4 fils : MS41



Date : 03.04.08  
Référence : ms41\_v3\_um\_fr.pdf  
*Réf. MI* : BLN1570830.DOC  
Révision : 3  
Auteur : B.LOPEZ



## SOMMAIRE

<b>I – INTRODUCTION.....</b>	<b>4</b>
<b>I.1 – Description générale .....</b>	<b>5</b>
<b>I.2 – Fonctions .....</b>	<b>6</b>
<b>I.3 – Organisation des modules MICROSIMPA 4 fils .....</b>	<b>6</b>
<b>I.4 – Mise en œuvre .....</b>	<b>8</b>
<b>I.5 – Modes de contrôle.....</b>	<b>8</b>
<b>II – SPECIFICATIONS DU MODULE MICROSIMPA 4 FILS.....</b>	<b>9</b>
<b>II.1 – Caractéristiques générales.....</b>	<b>9</b>
<b>II.2 – Caractéristiques mécaniques.....</b>	<b>10</b>
<b>II.3 – Connectique.....</b>	<b>11</b>
<b>II.4 – Visualisation .....</b>	<b>13</b>
<b>III – MISE EN ŒUVRE DU MODULE MICROSIMPA 4 FILS .....</b>	<b>14</b>
<b>III.1 – Alimentation (J1) .....</b>	<b>14</b>
<b>III.2 – Gestion moteur (J1).....</b>	<b>14</b>
<i>III.2.1 – Connexion au moteur .....</i>	<i>14</i>
<i>III.2.2 - Réglage du courant .....</i>	<i>14</i>
<i>III.2.3 - Courant de repos et surcourant.....</i>	<i>15</i>
<i>III.2.4 - Gestion de la puissance moteur.....</i>	<i>15</i>
<i>III.2.5 – Résolution.....</i>	<i>15</i>
<b>III.3 – Entrées/sorties.....</b>	<b>15</b>
<i>III.3.1 - Les entrées/sorties logiques (J5, J6, J7, J8) .....</i>	<i>15</i>
<i>III.3.2 - Entrée STOP (J11).....</i>	<i>16</i>
<i>III.3.3 - Entrées Butées +/- (J3, J4).....</i>	<i>17</i>
<i>III.3.4 – Les tensions auxiliaires .....</i>	<i>17</i>
<i>III.3.5 – Entrées analogiques (J9).....</i>	<i>17</i>
<i>III.3.6 – Entrées codeur incrémental (J10).....</i>	<i>18</i>
<b>III.4 – Liaison série du module MICROSIMPA 4 fils .....</b>	<b>19</b>
<i>III.4.1 - Configuration par roues codeuses.....</i>	<i>19</i>
<i>III.4.2 – RS485 et retournement de ligne.....</i>	<i>19</i>
<i>III.4.3 - Configuration monoaxe .....</i>	<i>20</i>
<i>III.4.4 - Configuration multiaxe.....</i>	<i>20</i>
<b>IV – FONCTIONNALITES.....</b>	<b>21</b>
<b>IV.1 – Indexeur, indexeur amplificateur .....</b>	<b>21</b>
<b>IV.2 – Résolution et position : compteur de position absolue.....</b>	<b>21</b>
<b>IV.3 – Paramètres de fonctionnement.....</b>	<b>22</b>
<i>IV.3.1 – Paramètres définissant les mouvements moteur .....</i>	<i>22</i>
<i>IV.3.2 – Loi d'accélération/décélération .....</i>	<i>22</i>
<i>IV.3.3 – Limites et cohérence des paramètres de mouvement .....</i>	<i>23</i>
<i>IV.3.3.1 – Limites générales .....</i>	<i>23</i>
<i>IV.3.3.2 – Cohérence entre paramètres.....</i>	<i>24</i>
<i>IV.3.4 – Contrôle du courant moteur.....</i>	<i>24</i>
<b>IV.4 – Gestion des mouvements.....</b>	<b>25</b>
<i>IV.4.1 – Description du mouvement de base.....</i>	<i>25</i>
<i>IV.4.2 – Types de mouvements .....</i>	<i>25</i>
<b>IV.5 – Variables .....</b>	<b>26</b>
<i>IV.5.1 – Définition des variables .....</i>	<i>26</i>
<i>IV.5.2 – Manipulation des variables.....</i>	<i>26</i>
<i>IV.5.3 – Liste des variables système .....</i>	<i>27</i>

<b>IV.6 – Automatisation et séquences</b> .....	<b>29</b>
IV.6.1 – Organisation des séquences.....	29
IV.6.2 – Ecriture et mémorisation des séquences .....	29
IV.6.3 - Description des séquences.....	29
IV.6.4 – Sélection et exécution des séquences.....	30
IV.6.5 – Déroulement des séquences.....	30
IV.6.5.1 – Enchaînement naturel.....	31
IV.6.5.2 – Enchaînement conditionnel .....	32
IV.6.6 – Phases et séquences réservées .....	33
IV.6.6.1 – Phase d'arrêt : 254 .....	33
IV.6.6.2 - Phase d'interruption : 255.....	33
IV.6.6.3 – Séquence butée ou Séquence d'interruption : 99 .....	33
IV.6.7 – Enchaînement des séquences et sous séquences.....	33
IV.6.7.1 – Enchaînement .....	33
IV.6.7.2 – Sous séquences.....	34
IV.6.7.3 – Exemples .....	34
IV.6.8 – Résumé des natures de phases.....	35
<b>IV.7 – Utilisation des entrées logiques</b> .....	<b>36</b>
IV.7.1 – Généralités .....	36
IV.7.2 – Temps de prise en compte d'une entrée logique .....	36
IV.7.3 – Synchronisation de modules .....	36
<b>IV.8 – Gestion des sorties logiques</b> .....	<b>37</b>
IV.8.1 – Généralités.....	37
IV.8.2 – Gestion immédiate .....	37
IV.8.3 – Gestion différée.....	38
IV.8.3.1 – Absolue, relative .....	38
IV.8.3.2 – Génération d'impulsion .....	38
IV.8.3.3 – Fonctionnements répétitifs.....	39
IV.8.3.4 – Fonctionnement permanent.....	39
IV.8.3.5 – Interaction avec les séquences .....	40
<b>IV.9 – Etat du module et codes d'erreurs</b> .....	<b>40</b>
<b>V – DESCRIPTION DES COMMANDES</b> .....	<b>43</b>
<b>V.1 – Conventions utilisées</b> .....	<b>43</b>
<b>V.2 – Liste des commandes</b> .....	<b>44</b>
<b>V.3 – Glossaire des commandes</b> .....	<b>45</b>
<b>VI – ANNEXES</b> .....	<b>21</b>
<b>VI.1 – Schémas câbles d'interconnexion</b> .....	<b>75</b>
VI.1.1 – Câble liaison série PC RS232 en V24.....	75
VI.1.2 – Câble d'extension RS485.....	75
<b>VI.2 – Guide pratique d'utilisation de la carte MICROSIMPA 4 fils</b> .....	<b>76</b>
<b>VI.3 – Eléments livrés avec la carte MICROSIMPA 4 fils</b> .....	<b>81</b>
<b>VI.4 – Récapitulatif des commandes MICROSIMPA</b> .....	<b>82</b>

## **I – INTRODUCTION**

Les produits de la famille MICROSIMPA 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 MICROSIMPA.

Chaque module MICROSIMPA comprend : l'ensemble des fonctions et composants nécessaires au contrôle d'un axe moteur (pas à pas) une unité logique à microprocesseur et, suivant le cas, une unité de puissance à découpage de courant pouvant délivrer une intensité de 2A à 9A efficace ou plus par phase (3A à 13A crête) sous une tension d'alimentation de 12 à 160V suivant le type de carte.

Chaque module dispose d'une liaison série lui permettant de communiquer avec un ordinateur 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 MICROSIMPA peuvent fonctionner de façon autonome et réaliser de véritables petits automates.

A la différence des modules et cartes SIMPA Micropas basées avant tout sur une structure carte en rack au format 3U, les modules et cartes MICROSIMPA sont utilisables directement sans carte fond de panier ni carte de connexion complémentaire. Ils comprennent toute la connectique nécessaire à leur alimentation, liaison au moteur et aux différentes entrées/sorties. Ils possèdent en outre les éléments de fixation pour un montage en armoire ou boîtier électrique classique, voire une fixation par rail DIN normalisé.

Pour les particularités de la liaison série voir la note d'application "Liaison ordinateur : protocoles et syntaxe" (BLN1570828.DOC).

Pour les différences fonctionnelles entre les produits de la famille SIMPA voir la note d'application "Evolution des produits SIMPA, SIMPA Micropas et Microsimpa (BLN1570829.DOC).

Nos produits sont conçus pour fonctionner de manière fiable si ceux-ci sont installés et utilisés conformément au manuel utilisateur.

La maintenance du produit doit être exclusivement effectuée par Midi Ingénierie, sauf remplacement du fusible.

### **Précautions d'utilisation et de stockage**

- ✓ Ne pas toucher ou débrancher le produit lorsqu'il est sous tension.
- ✓ Attendre l'extinction complète des leds avant toute manipulation du produit.
- ✓ Ne pas brancher le produit lorsque l'alimentation est sous tension.
- ✓ Ne pas poser le produit sur un emplacement qui ne soit pas stable : le produit pourrait tomber et entraîner des blessures ou être endommagé.
- ✓ Respecter les consignes d'aération précisées dans le manuel utilisateur.
- ✓ Ne pas utiliser ou stocker le produit dans un endroit humide.
- ✓ Relier à priori la masse mécanique du produit à la masse de référence de la machine (terre) via la borne prévue à cet effet (voir § II.3 pour plus de détails).
- ✓ Réaliser un câblage soigneux de la carte.
- ✓ Utiliser des câbles blindés à la terre pour des liaisons d'alimentation et moteur supérieures à 0,3 m.

### **Protections internes**

Une protection en entrée par fusible protège l'alimentation amont de surconsommation éventuellement due à une défaillance du produit ou de l'élément qu'il pilote sous réserve d'un dimensionnement des conducteurs d'alimentation en accord avec la valeur de coupure de 5 A du fusible (voir § II.3 et III.1).

Des éléments de protection accessibles à l'utilisateur sont présents :

- ✓ Entrée arrêt d'urgence.

## **I.1 – Description générale**

Le module MICROSIMPA 4 fils est une carte de commande intelligente qui permet de piloter pratiquement tout type de moteur pas à pas en mode pas entier, demi pas ou micropas jusqu'à des courants de 3 Aeff par phase en mode bipolaire.

Le module MICROSIMPA 4 fils, sous forme d'un petit boîtier autonome, est directement dérivé des versions rackables au format 3U des cartes SIMPA micropas 4 fils.

Ce module intègre un ensemble de connecteurs qui facilitent la connexion des alimentations et des diverses entrées/sorties ainsi que des connecteurs nécessaires à l'interfaçage vers un PC via une liaison série standard et la connectique vers un moteur pas à pas 4 ou 8 fils.

Comme tous les produits de la famille SIMPA, le module MICROSIMPA 4 fils gère les rampes d'accélération indispensables à la réalisation des mouvements réels.

Il dispose de 8 entrées et 8 sorties logiques indépendantes opto-isolées et il offre en plus :

- ✓ 2 entrées analogiques,
- ✓ 1 entrée codeur incrémental,
- ✓ une gestion réelle en micropas de 1 à 64 micropas par pas jusqu'à 20Kpas/s,
- ✓ une gestion des vitesses en pas/seconde quelle que soit la résolution choisie (la vitesse moteur est indépendante de la résolution choisie),
- ✓ la gestion de 64 variables et de sorties de synchronisation programmables pour augmenter les performances et la souplesse des automatismes réalisables,
- ✓ une gestion des modes surcourant et repos totalement automatique, de façon indépendante des entrées/sorties logiques,

Grâce à son langage de programmation performant et commun à toute la famille SIMPA, le module MICROSIMPA 4 fils est un véritable automate temps réel. Toutes les entrées et sorties sont gérées à la vitesse du pas (ou micropas) d'horloge.

## **I.2 – Fonctions**

Les principales fonctions directement accessibles sont :

- ✓ la définition des paramètres du mouvement à effectuer :
  - vitesse de démarrage ( $V_{min}$ ) en pas/seconde,
  - vitesse de consigne ou palier ( $V_{max}$ ) en pas/seconde,
  - durée des rampes d'accélération et de décélération en ms,
  - 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 (lecture position, entrées, sorties,...).

Certaines fonctionnalités supplémentaires sont offertes pour le fonctionnement en séquence :

- ✓ enchaînement de plusieurs mouvements,
- ✓ temporisation,
- ✓ gestion en temps réel de 8 sorties logiques,
- ✓ gestion de 2 entrées analogiques,
- ✓ acquisition d'un codeur incrémental,
- ✓ modification du déroulement de la séquence en temps réel en fonction de l'état de 8 entrées logiques,
- ✓ appel de sous-séquences, enchaînement de séquences, démarrage conditionnel ou non de séquences,
- ✓ utilisation de variables pour réaliser des compteurs, mémoriser des positions...

## **I.3 – Organisation des modules MICROSIMPA 4 fils**

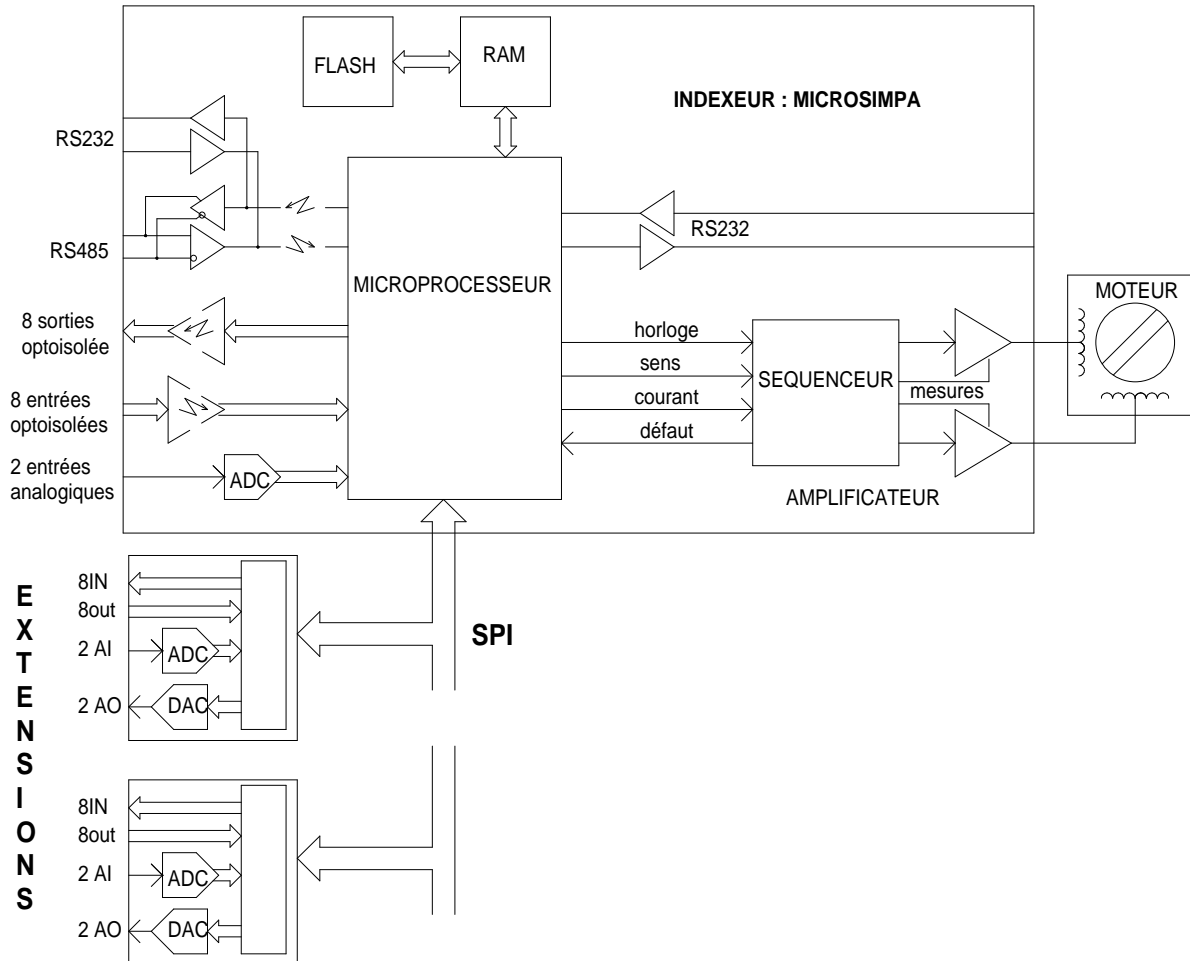
Les modules sont organisés pour :

- ✓ 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,

Pour cela chaque module comprend :

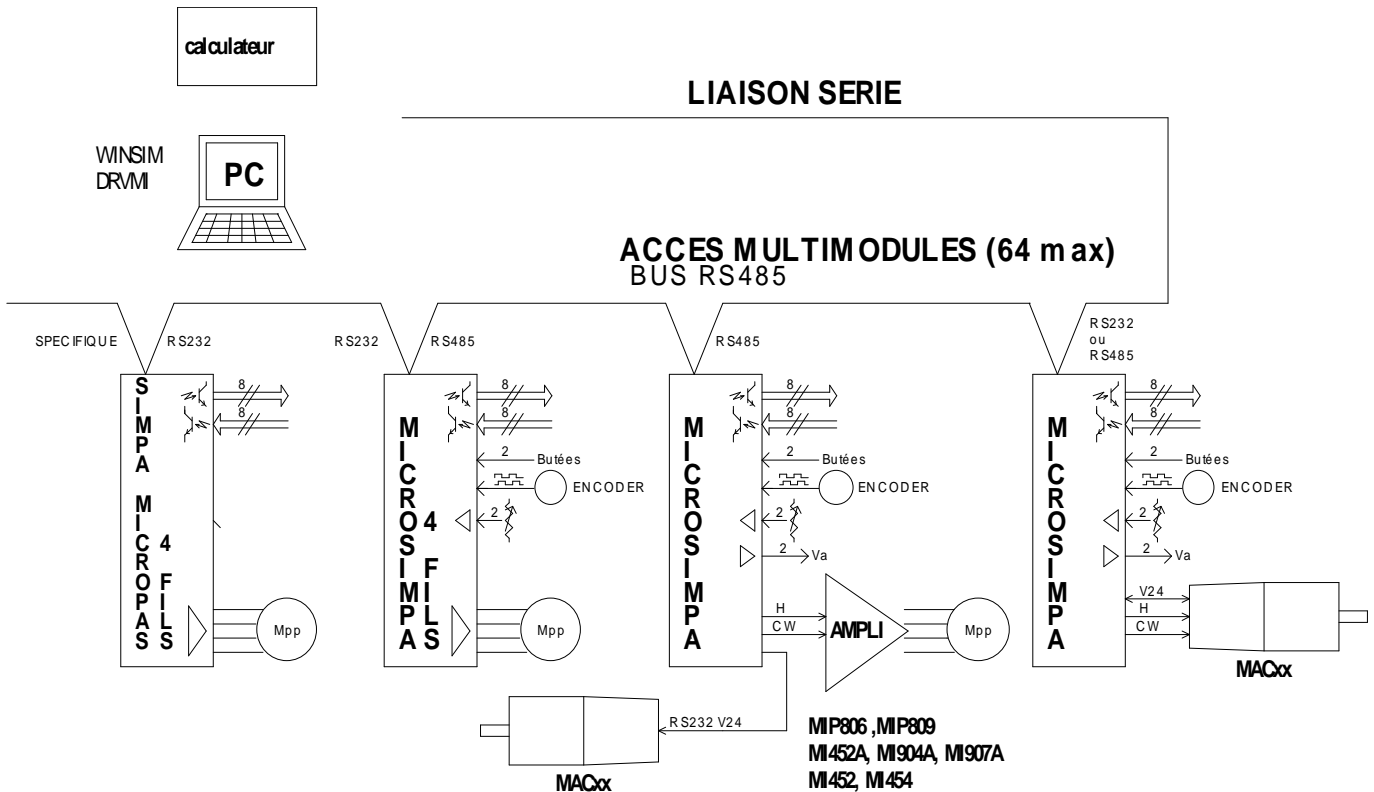
- ✓ une unité à microprocesseur et une mémoire pour le stockage :
  - des paramètres définissant les mouvements,
  - des séquences pour fonctionnement autonome,
- ✓ les étages de puissance pour connexion directe sur les moteurs en 4 ou 8 fils ;
- ✓ des entrées/sorties logiques pour communiquer avec l'environnement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

## Synoptique général d'un module MICROSIMPA 4 fils



### I.4 – Mise en œuvre

La mise en œuvre des modules MICROSIMPA peut se faire à distance, grâce à une liaison série au standard RS232 ou RS485, unique pour un maximum de 64 modules. Le logiciel WINSIM2 et la DLL de gestion du protocole de dialogue avec les modules (DRVSIM) peuvent être livrées sur demande afin de faciliter le dialogue avec les modules MICROSIMPA à partir d'un PC.



Les modules MICROSIMPA sont parfaitement compatibles avec l'ensemble des modules et cartes de la famille SIMPA et peuvent donc être utilisés conjointement sur la même liaison série.

### I.5 – Modes de contrôle

Comme tous les modules de la famille SIMPA, les modules MICROSIMPA peuvent être utilisés de deux manières différentes :

- En accès direct : Les commandes envoyées au module sont exécutées immédiatement. L'état du module est obtenu par relecture des paramètres et des variables d'état. Dans ce cas, l'automatisme, si nécessaire, doit être entièrement réalisé par le calculateur, les modules se comportent comme de simples actionneurs ou capteurs.
- En automate indépendant ou non. Les commandes à exécuter sont mémorisées dans le module lui-même sous forme de séquences. Le déroulement des séquences peut être contrôlé par différentes entrées logiques telles que des butées, la valeur de variables utilisées directement par les séquences. Grâce aux entrées et sorties logiques dont dispose chaque module, il est possible de synchroniser plusieurs axes entre eux, voir avec l'environnement des modules (automates, butées, capteurs, ...) et de piloter directement quelques actionneurs (électrovannes, voyants,...).



## II – SPECIFICATIONS DU MODULE MICROSIMPA 4 FILS

### II.1 – Caractéristiques générales

Dimensions	Compatible rail DIN	104 x 73 x 58,9 mm
Masse		460 g
Alimentation	Continue	12 V - 50 V <sub>DC</sub>
	Consommation (sans moteur ni actuateur)	< 0,2 A
Fusible	à souder	Retardé 5 A
Résolution	r	1, 2, 4, 8, 16, 32, 64 $\mu$ pas/pas
Vitesse		$1 \leq \omega \leq 20\,000$ pas/s*
Contrôle moteur	Moteur pas à pas 4 ou 8 fils	$\leq 3 A_{eff}$
2 entrées butées	alimentation capteur - pull up entrée 10 K $\Omega$ -	24V (<10mA) seuil TTL
8 entrées logiques	opto-isolées	4V $\leftrightarrow$ 30 V
8 sorties logiques	opto-isolées @ 1 mA @ 5 mA	< 0,6 V < 1,1 V I $\leq$ 80 mA
2 entrées analogiques	0-10V	Résolution 10 bits Précision $\pm 2$ LSB
1 entrée codeur incrémental	Diphasée	2 entrées différentielles 5V Seuil $\pm 200$ mV
Liaison série	interface RS232V24 et interface RS485	1 à 16 modules parallèles 9600, 19200, 38400, 115000 bauds 8 bits sans parité
Sortie 5V	5V <sub>aux</sub>	} P $\leq$ 1W Pour l'ensemble
Sortie 24V	24V <sub>aux</sub>	
Sortie 12V	+12V <sub>aux</sub>	
Extensions	SPI	Contrôle Modules d'extension d'entrées/sorties
Température de fonctionnement		0 - 50°C
Divers	gestion repos et surcourant	Indépendants des sorties logiques

\*suivant la résolution choisie.

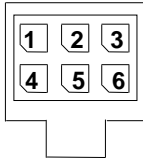


## II.3 – Connectique

### Alimentation et sortie moteur : J1 MOLEX MINIFIT (2x3)

Embase : 39-28-1063  
Fiche : 39-01-2060      Contacts : 39-00-0078

#### J1

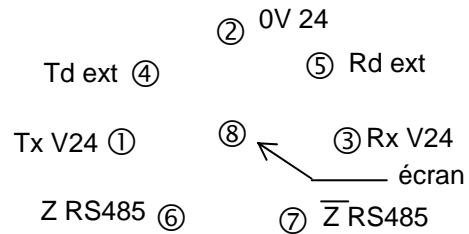


1 : Phase B+	2 : MASSE	3 : Phase A+
4 : Phase B-	5 : VALIM	6 : Phase A-

Cosse Faston : Masse méca

### ▪ Liaison série : J2 Connecteur DIN 45326 (8 pts)

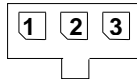
Embase : KRG81      Lumberg  
Fiche : SV81      Lumberg



### ▪ Liaison multiaxe : J15

Embase : 43650-0315  
Fiche : 43645-0300      Contacts : 430-30-007

#### J15



1 : 0V Liaison série	2 : Z RS485	3 : $\bar{Z}$ RS485
----------------------	-------------	---------------------

### ▪ Sorties logiques : J6, J5 MOLEX MICROFIT (2x4)

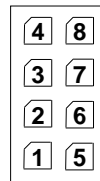
Embases : 43045-0812  
Fiche : 43025-0800      Contacts : 430-30-007

#### J6



4 : OUT1+	8 : OUT1-
3 : OUT2+	7 : OUT2-
2 : OUT3+	6 : OUT3-
1 : OUT4+	5 : OUT4-

#### J5

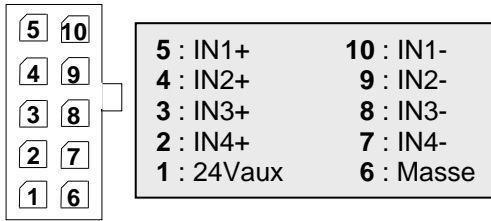


4 : OUT5+	8 : OUT5-
3 : OUT6+	7 : OUT6-
2 : OUT7+	6 : OUT7-
1 : OUT8+	5 : OUT8-

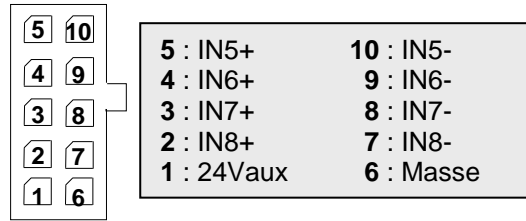
▪ **Entrées logiques : J8, J7 MOLEX MICROFIT (2x5)**

Embases : 43045-1012  
Fiche : 43025-1000    Contacts : 430-30-007

**J8**



**J7**



▪ **Arrêt d'urgence : STOP J11 MOLEX MICROFIT (1x2)**

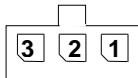
**J11**



▪ **Butées : J3, J4 MOLEX MICROFIT (1x3)**

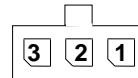
Embase : 43650-0315  
Fiche : 43645-0300    Contacts : 430-30-007

**J3**



3 : 24Vaux pour l'alimentation capteur  
2 : Entrées info capteur de butée sens anti-horaire (actif à 0) butée-  
1 : Masse capteur

**J4**



3 : 24Vaux pour l'alimentation capteur  
2 : entrées info capteur de butée sens horaire (actif à 0) butée+  
1 : Masse

Ces 2 butées sont connectées respectivement en entrées logiques 8 (butée-) et 7 (butée+) via les switches Cx2 et Cx1.

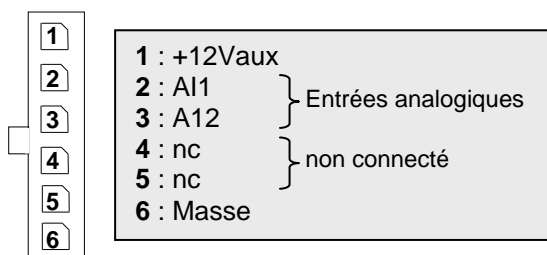


**Ne pas câbler les entrées 7 et 8 lorsque Cx1 et Cx2 sont sur ON.**

▪ **Entrées sorties analogiques : J9 MOLEX MICROFIT (1x6)**

Embase : 43650-0615  
Fiche : 43645-0600    Contacts : 430-30-007

**J9**

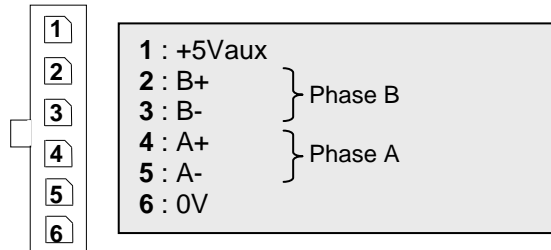


! Courant max fourni par +12Vaux :10mA

▪ **Entrées codeur incrémental : J10 MOLEX MICROFIT (1x6)**

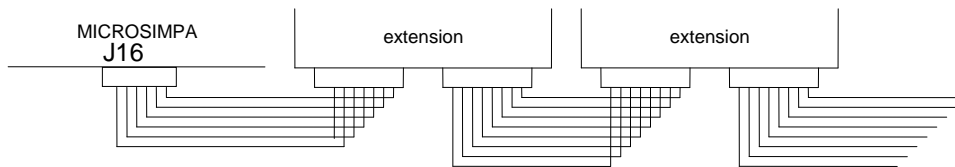
Embase : 43650-0615  
Fiche : 43645-0600    Contacts : 430-30-007

**J10**



▪ **Extension des entrées/sorties : HE14 6 points J16**

Les modules d'extension sont connectés en série par des câbles point à point, dont la longueur totale ne doit pas dépasser 50 cm (placer si possible les modules les uns à côté des autres). Cette connectique n'est pas décrite ; elle est exclusivement réservée à l'interconnexion des modules d'extension Midi Ingénierie.



**II.4 – Visualisation**

Cinq diodes électroluminescentes résument l'état du module MICROSIMPA :

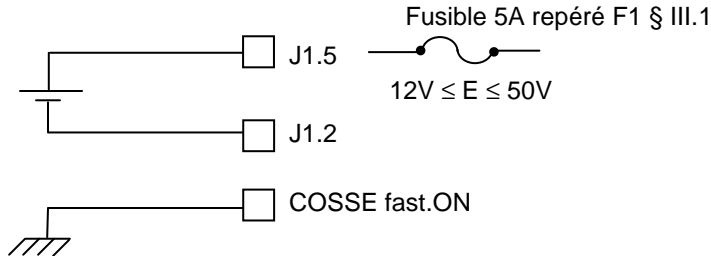
- ✓ la led verte "ON" est un témoin d'alimentation,
- ✓ la led jaune "Busy" matérialise l'activité du moteur : mouvement ou séquence en cours,
- ✓ la led rouge "Def" indique un défaut : surtension (U) ou sous-tension (u), même fugitive, d'alimentation (12 à 50 V<sub>DC</sub>), un défaut surcharge des tensions auxiliaires (O) ou autre défaut (X). Par ailleurs, cette led clignote à la mise sous tension, à l'initialisation et à la mise à zéro du module (MR), (voir chapitre IV.9).
- ✓ 2 leds rouges visualisent l'état des entrées butées, elles sont placées respectivement à côté des bornes de connexion des butées.

### III – MISE EN ŒUVRE DU MODULE MICROSIMPA 4 FILS

**Attention !** Les éléments de configuration qui ne sont pas décrits dans ce chapitre ne doivent pas être modifiés par l'utilisateur sous peine de destruction de la carte.

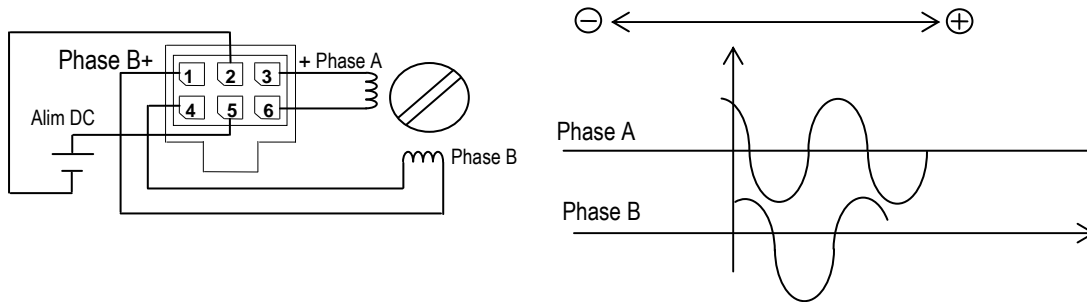
#### III.1 – Alimentation (J1)

L'alimentation est obligatoirement continue. Elle peut être comprise entre 12 et 50V. La borne de terre doit être connectée impérativement.



#### III.2 – Gestion moteur (J1)

##### III.2.1 – Connexion au moteur



##### III.2.2 - Réglage du courant

La valeur efficace du courant injecté dans les phases moteur peut être ajustée entre 0 et 3 Aeff grâce à la commande GI :  $I_{MOT} = \frac{3 \cdot Im}{255} A_{eff}$ .

La puissance réellement consommée par le moteur est égale aux pertes joules liées aux résistances des bobines moteur et à la puissance mécanique réellement fournie :

$$P_m = C\omega + 2 r_m I_{MOT}^2 + \text{pertes fer}$$

- C : le couple réellement fourni par le moteur
- $r_m$  : la résistance d'une bobine de phase moteur
- $I_{MOT}$  : le courant moteur injecté

Le courant d'alimentation du module peut être estimé à  $I_{alim} = I_o + \frac{P_M}{V_{alim} \cdot \eta}$

$\eta$  : rendement du moteur et du découpeur  $\approx 60\%$

$I_o$  : courant consommé par la carte sans moteur

}	$I_o \leq 50 \text{ mA @ } 24 \text{ V}$
}	$I_o \leq 25 \text{ mA @ } 45 \text{ V}$

### III.2.3 - Courant de repos et surcourant

Le module MICROSIMPA gère automatiquement la mise au courant de repos du moteur à chaque arrêt de mouvement que ce soit en mouvement direct ou en séquence.

Cette fonctionnalité peut être supprimée à l'aide de la commande MSN (rétablie par MSB ou MSS).

De même lors des phases d'accélération et décélération, le courant moteur peut être forcé à une valeur supérieure du courant sélectionné si la fonction surcourant (Boost) est sélectionnée.

Cette fonctionnalité peut être supprimée à l'aide des commandes MSN et MSS (rétablie par MSB).

Le courant de repos du moteur est  $I_{stb} = \frac{I_{MOT}}{2}$ .

Le courant de boost est  $I_{bst} = 1,3 * I_{MOT}$  (borné à 3 Aeff).

### III.2.4 - Gestion de la puissance moteur

La commande GR, la réinitialisation du module MICROSIMPA (commande MR ou mise sous tension) impose la coupure de la puissance moteur. Elle est réactive à chaque lancement de mouvement ou par la commande GM, sauf en cas de disjonction.

### III.2.5 – Résolution

La résolution moteur peut être choisie au mieux pour s'adapter à la résolution des mouvements à réaliser parmi les valeurs suivantes au moyen de la commande WN res.

Res = 1, 2, 4, 8, 16, 32 ou 64 micropas / pas

Soit pour un moteur à 200 pas/tour

Res = 200, 400, 800, 1600, 3200, 6400, 12800 positions / tours

Plus la résolution choisie sera élevée, plus régulier sera le mouvement moteur et moins les phénomènes de résonances liées au fonctionnement pas à pas seront sensibles.

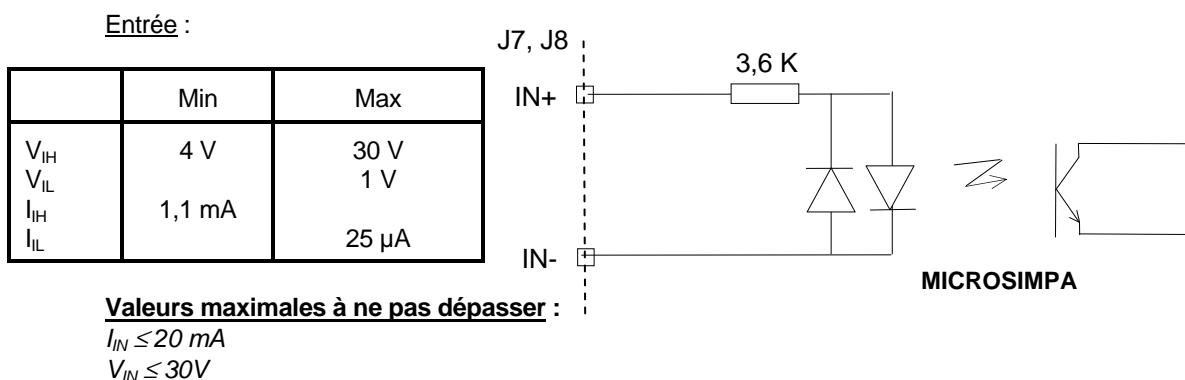
Dans la pratique, on considère que ces dernières deviennent négligeables au dessus de 8 à 10  $\mu$ pas/pas.

## III.3 – Entrées/sorties

### III.3.1 - Les entrées/sorties logiques (J5, J6, J7, J8)

Le module MICROSIMPA dispose de 8 entrées et 8 sorties logiques opto-isolées indépendantes. Elles sont repérées de IN1 à IN8 pour les entrées et OUT1 à OUT8 pour les sorties.

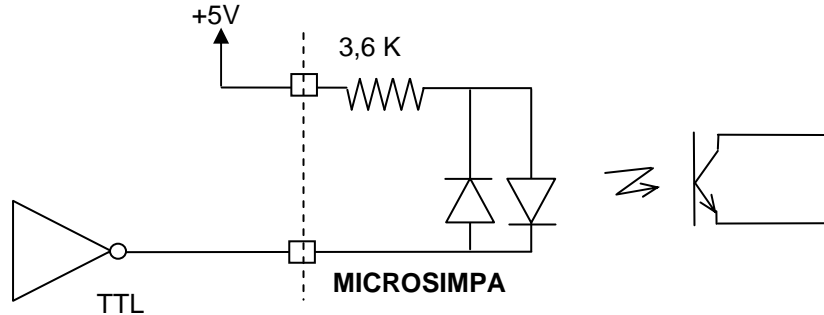
Chaque entrée ou sortie se comporte comme un élément de "boucle sèche" et possède donc un signal et un retour, d'où les broches IN+, IN-, OUT+ et OUT- définies dans le paragraphe connectique. Elles sont compatibles avec des logiques 5V classiques ou des logiques automates en 24V.



Ces entrées sont protégées contre les inversions de polarité.



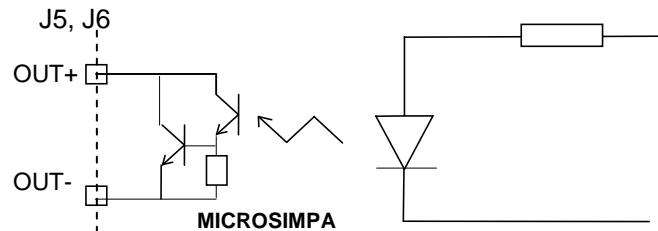
Pour fonctionner avec un niveau TTL, il convient de travailler en logique inversée comme présente sur le schéma ci-dessous car le niveau TTL ne permet pas de polariser correctement les optocoupleurs.



Les entrées In1 à In8 sont associées aux bits de la variable #IN (dans l'ordre de leur poids).

Sortie :

	Min	Max
$I_{OH}$		0,1 mA
$V_{OL}$		0,6V @1 mA 1,1V @5 mA 1,3V @50 mA 1,6V @80 mA



**Valeurs maximales à ne pas dépasser :**

$$V_o \leq 30V$$

$$I_o \leq 80 mA$$

Les sorties OUT1 à OUT8 sont associées aux bits de la variable #OUT (dans l'ordre de leur poids).

### III.3.2 - Entrée STOP (J11)

L'entrée STOP est une entrée opto-isolée ayant les mêmes caractéristiques que les entrées logiques décrites plus haut.

L'activation de l'entrée provoque l'arrêt de tout mouvement sans perte de la puissance de façon à éviter si possible le glissement.

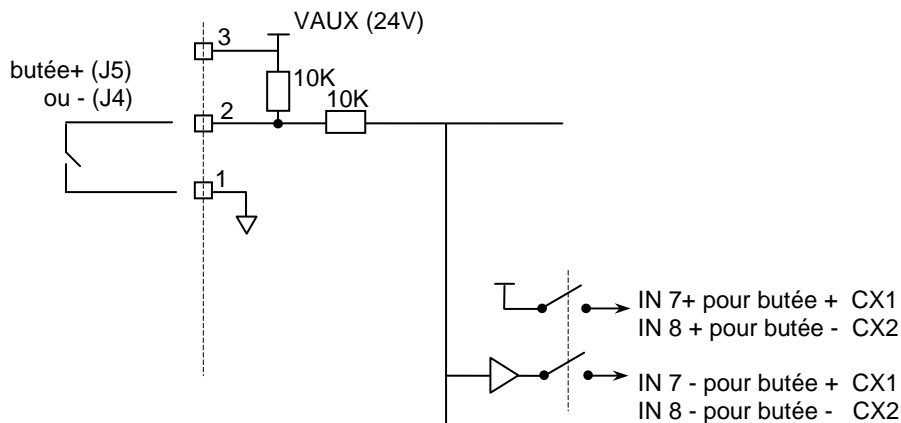
Cette entrée STOP est équivalente à la commande GS



### III.3.3 - Entrées Butées +/- (J3, J4)

Ces 2 entrées butée –(J3) et butée +(J4) sont associées aux entrées IN8 pour butée- et IN7 pour butée+ lorsque les microswitches CX2 et CX1 sont placés en position ON.  
(les entrées 7 et 8 peuvent être configurées en mode butée par la commande MB).

Deux leds rouges visualisent respectivement l'activation de ces deux signaux but+ et but-.



**Lorsque Cx1 et Cx2 sont sur On, ne pas câbler les entrées In7 et In8**

### III.3.4 – Les tensions auxiliaires

Le module Microsimpa 4 fils délivre trois niveaux de tension auxiliaire afin de faciliter l'interconnexion des différents périphériques.

L'alimentation +5V logique correspond à l'alimentation logique interne de la carte. Elle est disponible sur la broche 1 du connecteur J10 et ne doit être utilisée que pour alimenter un codeur sous réserve de ne pas dépasser 50 mA.

Le 12V délivré par la broche 1 du connecteur analogique J9 est conçu pour servir de référence pour un potentiomètre, il conviendra donc de ne pas dépasser 10 mA.

Le +24V est accessible sur la broche 1 des connecteurs d'entrées logiques (J7, J8) et des butées (J3, J4).

La puissance maximale délivrée par l'ensemble de ces trois tensions auxiliaires ne doit pas dépasser 1W. En cas de surcharge, le module suspend la génération de ces trois tensions et rentre en mode défaut surcharge (QX=0).

### III.3.5 – Entrées analogiques (J9)

#### Entrées analogiques AI1 J9.2, AI2 J9.3

Dynamique	$0 \leq V_{IN} \leq 10V$
Résolution	10 bits (9,76 mV)
Précision	2%
Impédance d'entrée	100 K $\Omega$
Bande passante	10 Hz

Valeur à ne pas dépasser  $\pm 20V$

L'entrée AI1 est associée à la variable # AI1, l'entrée AI2 à la variable # AI2 (exprimé directement en mV).

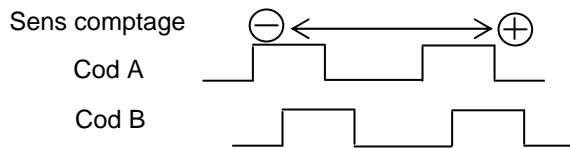
III.3.6 – Entrées codeur incrémental (J10)

Cette entrée est destinée à l'acquisition d'un codeur incrémental diphasé différentiel en logique 5V.

Elle est associée à la variable # codeur.

Etat	Vdif
1 logique	> 200 mV
0 logique	< 200 mV

Plage de mode commun à ne pas dépasser  $-14 \leq V_{CM} \leq +14V$



Fréquence maximum : 50KHz (200 000 transitions par seconde)

### III.4 – Liaison série du module MICROSIMPA 4 fils

La nouvelle génération des cartes des familles SIMPA dispose en standard de deux interfaces d'entrée série utilisant le même port de communication : une interface V24 pour les configurations monoaxes et une interface RS485 pour des configurations multiaxes.

En complément des interfaces vers le calculateur, le module MICROSIMPA dispose d'une recopie du dialogue série sur une interface V24 supplémentaire, ce qui permet d'associer sur une seule liaison calculateur un module MICROSIMPA et un module MAC.

#### III.4.1 - Configuration par roues codeuses

Le protocole et la vitesse de transmission sont fixés par les roues codeuses SW1 et SW2 accessibles sur le côté du module.

Adresse : SW1

Adresse	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Position	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

#### Baudrate et protocole SW2

Protocole		XON/XOFF	TERM	ACK/NACK	Réservé
Baudrate	9600 b	0	4	8	C
	19200 b	1	5	9	D
	38400 b	2	6	A	E
	115200 b	3	7	B	F

Les caractéristiques précises des protocoles utilisables sont décrites dans la note d'application "liaison calculateur : protocoles et syntaxe" (BLN1570828.DOC).

La mise en œuvre de ce dialogue série peut être grandement facilitée par l'utilisation du logiciel WINSIM2, véritable interface opérateur implantable sur tout type de PC.

Pour l'utilisateur qui souhaiterait dialoguer avec les cartes SIMPA à l'intérieur de ses propres programmes, Midi Ingénierie a développé une DLL de gestion du protocole de dialogue avec les modules pouvant être livrée sur demande.

Le protocole XON/XOFF est configuré à défaut, il doit être conservé sauf nécessité contraire.

#### III.4.2 – RS485 et retournement de ligne

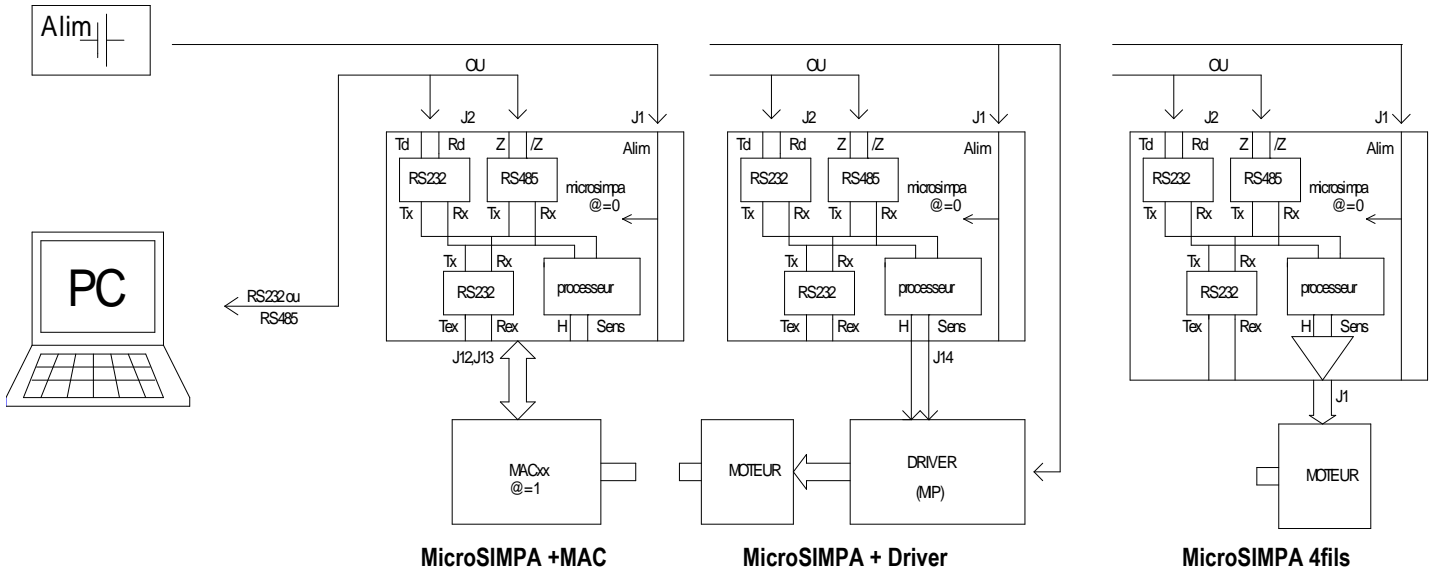
En liaison RS485, la même ligne sert de support à l'émission et à la réception des messages.

Pour éviter tout conflit de prise de ligne, le temps de retournement de ligne des modules MICROSIMPA peut être programmé pour s'adapter aux caractéristiques du driver de ligne de l'utilisateur.

A défaut, le temps entre réception et réponse est de 80 µs, il peut être modifié grâce à la commande :

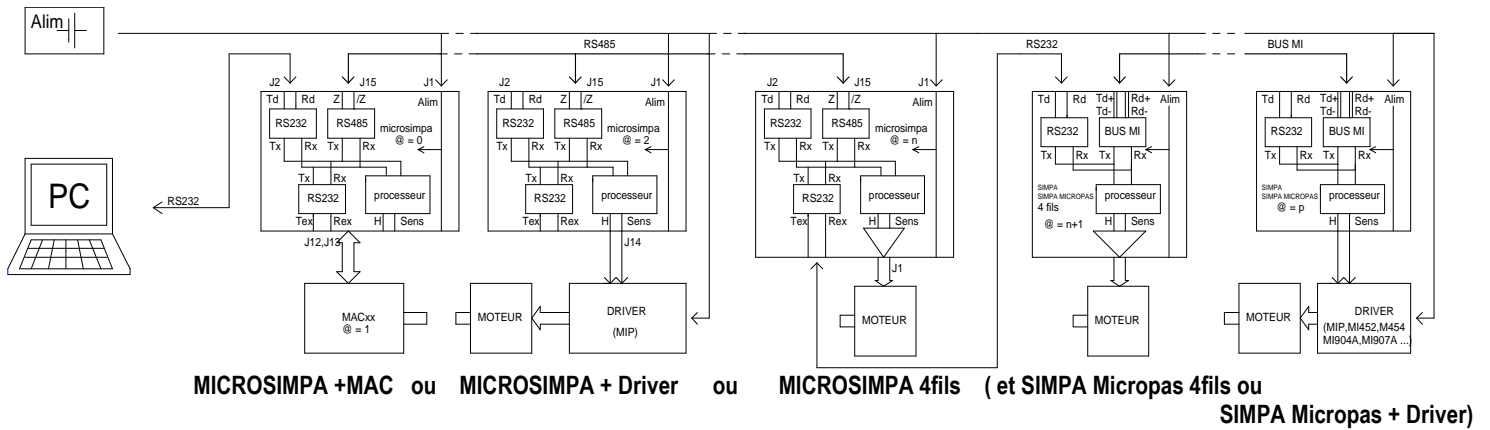
PO#TRS:=t où t est le délai de retournement exprimé en multiple de 80 µs avec  $1 \leq t \leq 100$

**III.4.3 - Configuration monoaxe**



**III.4.4 - Configuration multiaxe**

Le premier axe MICROSIMPA peut être utilisé en passerelle pour coupler à une liaison série standard V24 d'un PC au bus RS485 qui permet de placer plusieurs modules MICROSIMPA sur la même liaison série. Il est possible de lier l'ancien bus des cartes SIMPA et SIMPAMICROPAS à ce bus en utilisant la liaison série d'extension pour se connecter en V24 au premier module de la chaîne SIMPA conformément au synoptique suivant.



## IV – FONCTIONNALITES

### IV.1 – Indexeur, indexeur amplificateur

La famille des modules MICROSIMPA 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-xxx, MIP-xxx et MAC-xx développés par la société MIDI INGENIERIE.

Les modules MI-xxx sont des amplis micropas, les modules MIP-xxx des amplificateurs pas entier ½ pas. Les modules MAC-xx sont des modules autocommutés intégrant moteur et électronique. Il sont disponibles sous deux versions : une version standard directement pilotable par liaison série et une version dite "horloge & sens" prévue pour être couplée avec un module SIMPA ou MICROSIMPA pour bénéficier de la puissance de gestion d'automatisme de ce dernier. 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 MICROSIMPA.

**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 MICROSIMPA 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 ainsi qu'à la définition de la résolution moteur (en micropas par pas).**

### IV.2 – Résolution et position : compteur de position absolue

Chaque module MICROSIMPA gère la position du moteur qui lui est associé grâce à un compteur : CPA (compteur de position absolue). Cette position est exprimée dans l'unité d'échelle définie par la résolution angulaire du moteur :  $r_m$  (exprimé en pas / tour) et la résolution de l'amplificateur utilisé :  $\mu$  (exprimée en  $\mu$ pas / pas). La résolution globale  $r = \mu * r_m$  exprime donc un nombre de positions par tour.

Le compteur de position absolue compte ou décompte, suivant le sens de rotation, le nombre d'incrément de mouvements imposés au moteur.

Le compteur de position absolue 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 position absolue est :  $-2\ 147\ 483\ 647 \leq CPA \leq +2\ 147\ 483\ 647$  (modulo  $2^{32}$ )

#### 1) Définition de la résolution

La résolution en micropas par pas doit être définie par l'utilisateur et correspondre à la résolution sélectionnée sur l'ampli de puissance si celui-ci est indépendant de l'unité logique MICROSIMPA (y compris éventuellement 1 micropas par pas ou 2 micropas par pas pour les fonctionnements respectifs pas entier et 1/2 pas).

Pour les modules MICROSIMPA 4 fils et MICROSIMPA 4 axes 6 fils, seules les résolutions 1, 2, 4, 8, 16, 32 et 64  $\mu$ pas/pas sont accessibles

#### 2) Définition des vitesses

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 MICROSIMPA 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 MICROSIMPA.*

#### 3) Définition des déplacements

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

$$r = r_m * \mu$$

$r_m$  : résolution moteur en pas/tour

$\mu$  : résolution de l'amplificateur en  $\mu$ pas/pas défini avec le paramètre WN

Par exemple il faudra définir un déplacement de valeur 16 pour obtenir un mouvement de 1 pas entier avec un module micropas dont la résolution est de 16 micropas par pas et un déplacement de valeur 3200 pour réaliser un tour moteur pour un moteur à 200 pas/tour.

### IV.3 – Paramètres de fonctionnement

#### IV.3.1 – Paramètres définissant les mouvements moteur

Chaque mouvement est défini par les quatre paramètres de la loi WL, WH, WT et WN :

- ✓ La vitesse de démarrage :  $V_{min}$  de 1 à 20 000 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 1 à 20 000 pas/s définit la vitesse nominale d'exécution des mouvements.
- ✓ Les durées ( $T_a$  et  $T_d$ ) 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  $\mu$  en micropas par pas.

et par une expression du déplacement :

- ✓ La variation de position à réaliser en micropas et le sens de rotation du moteur :
- ✓ La position algébrique du moteur à obtenir en micropas

#### Mode direct Séquence

GO +/-ddd NP +/-ddd

GA +/-Pos NX +/-Pos

\*La valeur minimum de la vitesse dépend en fait de la résolution (voir § 4.3.3).

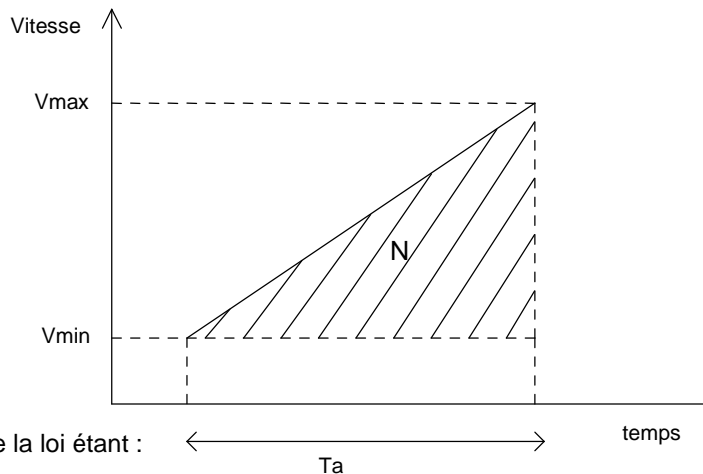
#### IV.3.2 – Loi d'accélération/décélération

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 MICROSIMPA 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  $T_a$  et  $T_d$  pour passer de  $V_{min}$  à  $V_{max}$  ou respectivement de  $V_{max}$  à  $V_{min}$ . Ces profils de vitesse sont appelés respectivement loi d'accélération et loi de décélération.

$T_a$  et  $T_d$  sont égaux par défaut mais peuvent être différenciés 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)..

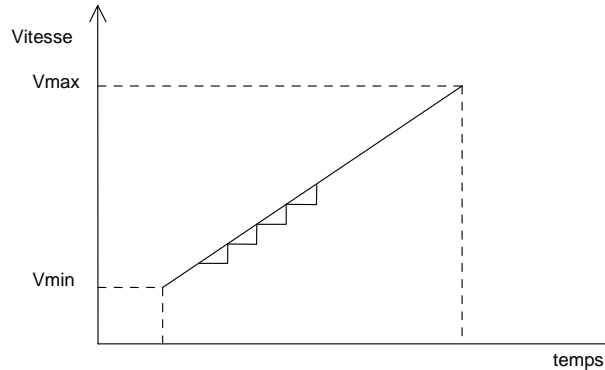


Les paramètres de la loi étant :

- $V_{min}$  = vitesse de démarrage (pas/s)
- $V_{max}$  = vitesse de palier (pas/s)
- $T_a$  = temps de rampe
- $\mu$  = résolution en micropas/pas

Cette droite a pour pente l'accélération :  $\frac{V_{\max} - V_{\min}}{T_a}$ , le nombre total d'incrément 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 t<sub>i</sub> est donnée par la relation :

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

aux erreurs près liées à la discrétisation due aux calculs et à la résolution du générateur de micropas : 0,5 μs.

Un minimum de 1 incrément de position 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 suivant.

### IV.3.3 – Limites et cohérence des paramètres de mouvement

#### IV.3.3.1 – Limites générales

\*  $\frac{62}{\mu} \leq V_{\min} < V_{\max} \leq \frac{2^n}{\mu} * 20000 \text{ (pas / s)}$  avec n tel que  $2^n \leq \mu < 2^{n+1}$

\*  $1 \leq \mu \leq 256$  pour les modules micropas sans puissance intégrée  
*MICROSIMPA*

μ : 1, 2, 4, 8, 16, 32 ou 64 pour les modules micropas avec puissance intégrée  
*MICROSIMPA 4 fils et MICROSIMPA 4 axes 6 fils*

\*  $1 \leq TR \leq 65535 \text{ (ms)}$  (s'applique donc à T<sub>a</sub> et T<sub>d</sub>)

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

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

#### IV.3.3.2 – Cohérence entre paramètres

En plus des limites absolues de paramètres de mouvements, les paramètres doivent respecter certaines cohérences afin de ne pas dépasser la dynamique de calcul du module.

Cette cohérence est systématiquement vérifiée par le module à chaque modification de paramètre et donne lieu au code d'erreur 1 en cas de non respect.

Les tableaux suivants donnent les relations de cohérence entre l'ensemble des paramètres. C'est avant tout la dynamique du paramètre TR (temps de rampe) qui est réduite suivant les valeurs données aux autres paramètres.

$$* \frac{10^{+3}}{\mu * Tr \text{ (ms)}} \leq V \text{ min (p/s)} \leq \frac{63,75 \cdot 10^6}{\mu * Tr \text{ (ms)}}$$

$$* \mu * V \text{ min} \leq 20\,000 \text{ (p/s)}$$

$$* \mu * V \text{ max} \leq 1,28 \cdot 10^6 \text{ (p/s)}$$

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

#### IV.3.4 – Contrôle du courant moteur

Le courant moteur maximum que peut délivrer un module est un des éléments qui distingue les différents produits des familles MICROSIMPA, SIMPA et MI.

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

- ✓ par programmation du paramètre courant moteur pour les modules MICROSIMPA avec puissance intégrée,
- ✓ 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.

*Remarque* : 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).



## IV.4 – Gestion des mouvements

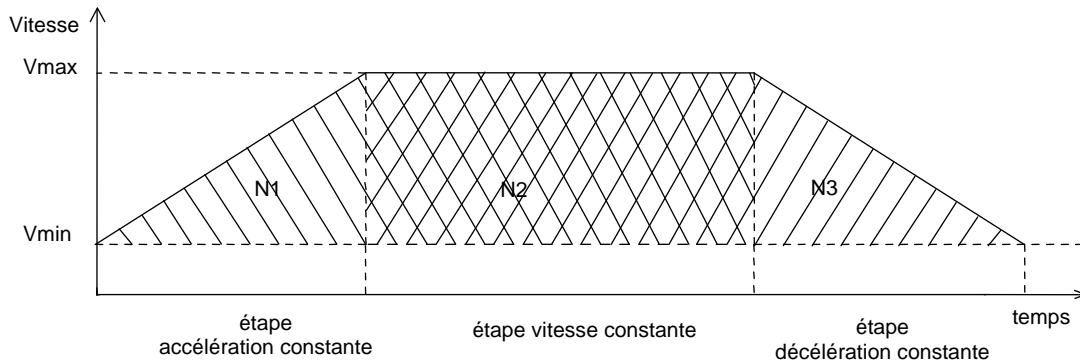
La gestion des mouvements constitue la fonction principale des modules MICROSIMPA. Les mouvements pouvant résulter de commandes exécutées immédiatement, ou de commande mémorisées dans les séquences.

### IV.4.1 – Description du mouvement de base

Sauf exception, tous les mouvements 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 : déplacement à effectuer.

Le module calcule automatiquement le déplacement à effectuer au cours de chaque étape du mouvement en fonction des paramètres de mouvement.

$$N = N1 + N2 + N3$$

Si le déplacement 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  $N3 = N1$  (sauf si les temps d'accélération :  $Ta$  et décélération :  $Td$  programmés sont différents).*

### IV.4.2 – Types de mouvements

Les modules MICROSIMPA gèrent trois types de mouvements :

- ✓ **Des mouvements relatifs** : la consigne donnée définit le déplacement 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 déplacement 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 position absolue), 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. 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 : 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.*

## IV.5 – Variables

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

### IV.5.1 – Définition des variables

Une variable est une donnée modifiable à tout moment à partir du dialogue série ou dans le déroulement même des séquences. On distinguera 2 types de variables, les variables utilisateur et les variables systèmes :

#### ✓ Les variables utilisateur

Ces variables sont totalement libres d'utilisation et peuvent être affectées à n'importe quelle grandeur au gré de l'utilisateur. Elles sont au nombre de 64 : 32 variables remises à zéro à chaque mise sous tension ou reset notées #1 à #32, et 32 variables dont les valeurs courantes restent mémorisées d'une mise sous tension à l'autre notées #M1 à #M32.

Ces données sont stockées sur 4 octets signés, la valeur des variables utilisateur doit donc être comprise entre  $-(2^{31}-1)$  et  $(2^{31}-1)$  elles peuvent représenter des valeurs numériques de consignes ou des données binaires (ou hexa) comme un état de sortie logique.

#### ✓ Les variables système

Si elles restent manipulables par l'utilisateur comme les précédentes, leurs affectations sont imposées par le module. Elles permettent d'accéder à certains paramètres internes du module. Les limites de ces variables dépendent évidemment de la nature des paramètres qu'elles représentent, elles sont systématiquement bornées par ces limites.

Les variables système sont représentées par un mnémonique précédé du signe # (voir le tableau du §IV.5.3).

### IV.5.2 – Manipulation des variables

#### ▪ Lecture écriture

Les valeurs des variables peuvent être modifiées, imposées ou relues par des commandes immédiates via la liaison série, la commande PO impose la valeur d'une variable.

PO#n\_:=\_v<sub>n</sub> (décimal)

PO#n\_:=\_Hv<sub>n</sub> (hexadécimal)

La commande QR#n retourne sur la liaison série la valeur courante d'une variable.

*Nota* : le caractère "\_" représente un espace obligatoire. La commande PO est implicite et peut donc être omise :

PO#n\_:=\_v<sub>n</sub> ⇔ #n\_:=\_v<sub>n</sub>

#### ▪ Accès bit à bit

L'ensemble des variables, et plus particulièrement les variables logiques peut être manipulé bit à bit #n.m = v avec v = 1 ou 0 et 1 ≤ m ≤ 32, permet de modifier seulement le bit m de la variable # n en lui donnant la valeur v. Attention, seuls les bits réellement utilisés sont modifiables ; la modification des autres bits est refusée par le module.

Ex :#OUT.1:=0 active la sortie 1 sans modifier les autres sorties.



**La lecture bit à bit en commande directe n'est accessible qu'à travers une relecture binaire.**

**Ex : QR #IN B**

#### ▪ Limites des variables

La dynamique des variables système dépend de leur utilisation, conformément au tableau donné ci-après. Lors des modifications de ces variables, les valeurs données sont automatiquement bornées aux limites supérieures et inférieures précisées dans le tableau. Ce mécanisme est valable aussi bien lors d'une écriture directe d'une variable que dans le cas du résultat d'une opération.

#### ▪ Opérations et séquences

Dans une séquence, les variables peuvent être utilisées directement comme valeurs de consignes, manipulées pour réaliser des opérations arithmétiques et logiques ou testées en association avec la directive NS.

#n\_:=\_#m [op.CTE]

#n\_?\_#m NS@1:@2:@3

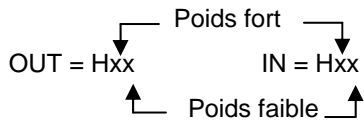
IV.5.3 – Liste des variables système
**Tableau récapitulatif des variables systèmes des modules MICROSIMPA**

Variable	Nom	Unité	Dynamique MICROSIMPA MS01	Dynamique MICROSIMPA 4 fils B157 MS41	Dynamique MICROSIMPA 4 axes 9160 MS64	Remarques
Position absolue	#CPA	Micropas	$\pm 2^{31} - 1$ ( $\pm 2147\ 483\ 647$ )			
Position codeur	#CODEUR	Pas codeur	$\pm 2^{31} - 1$ ( $\pm 2147\ 483\ 647$ )			
Tension d'alimentation	#VSUPPLY	1 m V	0..90 000	0..50 000	0.. 45000	Résolution : 100 mV
Tension auxiliaire	#VAUXP	1 mV			0..30000	Résolution : 200 mV
Température	#TEMP	°C	0..100			
Entrées logiques	#IN		1 octet (8 bits)			
	#INx		1 octet (8 bits)			x n° module extension ( $1 \leq x \leq 7$ )
Sorties logiques	#OUT		1 octet (8 bits)			
	#OUTx		1 octet (8 bits)			x n° module extension ( $1 \leq x \leq 7$ )
Entrées analogiques	#AI1 #AI2	1 mV	0..10000		0...10000	Résolution : 10 mV
	#AIAx #AIBx	1 mV	0..10000			Voies A et B du module d'extension n°x ( $1 \leq x \leq 7$ )
Sorties analogiques	#AO1 #AO2	1 mV	0..10000			Résolution 10 mV
	#AOAx #AOBx	1 mV	0..10000			Voies A et B du module d'extension n°x ( $1 \leq x \leq 7$ )

Toutes ces variables sont accessibles en lecture (QR) et écriture (PO).

L'écriture des variables #VSUPPLY, #TEMP, #VAUXP, #IN, #AI1 et #AI2 n'a bien évidemment aucun effet.

**Tableau de concordance entre les valeurs hexa. et les entrées/sorties**



		Entrées/sorties				Poids
Valeur Hexa	Valeur Binaire	IN/OUT 4 8	IN/OUT 3 7	IN/OUT 2 6	IN/OUT 1 5	Faible Fort
0	0000	Actif	Actif	Actif	Actif	
1	0001	Actif	Actif	Actif	Non actif	
2	0010	Actif	Actif	Non actif	Actif	
3	0011	Actif	Actif	Non actif	Non actif	
4	0100	Actif	Non actif	Actif	Actif	
5	0101	Actif	Non actif	Actif	Non actif	
6	0110	Actif	Non actif	Non actif	Actif	
7	0111	Actif	Non actif	Non actif	Non actif	
8	1000	Non actif	Actif	Actif	Actif	
9	1001	Non actif	Actif	Actif	Non actif	
A	1010	Non actif	Actif	Non actif	Actif	
B	1011	Non actif	Actif	Non actif	Non actif	
C	1100	Non actif	Non actif	Actif	Actif	
D	1101	Non actif	Non actif	Actif	Non actif	
E	1110	Non actif	Non actif	Non actif	Actif	
F	1111	Non actif	Non actif	Non actif	Non actif	

Ex.: la commande '00PO #OUT := h 7 A' Active les sorties Out8, Out3 et Out1 de l'axe 00 force ses autres sorties (Out7,6,5,4,2) dans l'état non-actif(na).



A quelques exceptions près, les commandes de phases sont exactement équivalentes aux commandes directement exécutées. Les fonctions (PO) permettent de manipuler les variables.

*	NA	: phase d'accélération	
*	ND	: phase de décélération	
*	NV	: vitesse constante	
	NP	: mouvement relatif	⇔ GO
	NX	: mouvement absolu	⇔ GA
	NH	: retour origine ("HOME")	⇔ GH
	NT	: puissance moteur ON	⇔ GM
	NU	: puissance moteur OFF	⇔ GR
*	NW	: attente	
*	NC	: modification consigne de vitesse palier $V_{max}$	
	PO	: Calcul et test sur les variables utilisateurs et les variables système.	

- ✓ la consigne : Cns (déplacement, position, vitesse palier, délai, n° de paramètre, ...)  
Pour un déplacement relatif le sens du mouvement est défini par le signe de la consigne.

- ✓ et les directives suivantes facultatives :

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 logiques restent en état.

D'où la structure de la commande :

SP 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.*

- \* Commande uniquement utilisables en séquence.

#### IV.6.4 – Sélection et exécution des séquences

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

- ✓ exécution immédiate de la séquence sélectionnée par le calculateur hôte (éventuellement via WINSIM ou DRVMI) (: commande SSn).
- ✓ exécution différée de la séquence choisie à toute nouvelle mise sous tension (ou Reset) (: commande SDn).

Bien évidemment, l'opérateur est libre d'arrêter toute séquence en cours d'exécution ou d'en supprimer l'exécution automatique (: commande GS, GE et SR).

#### IV.6.5 – 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.

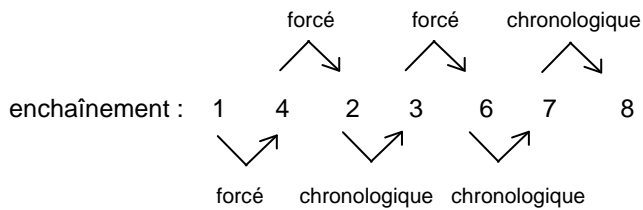
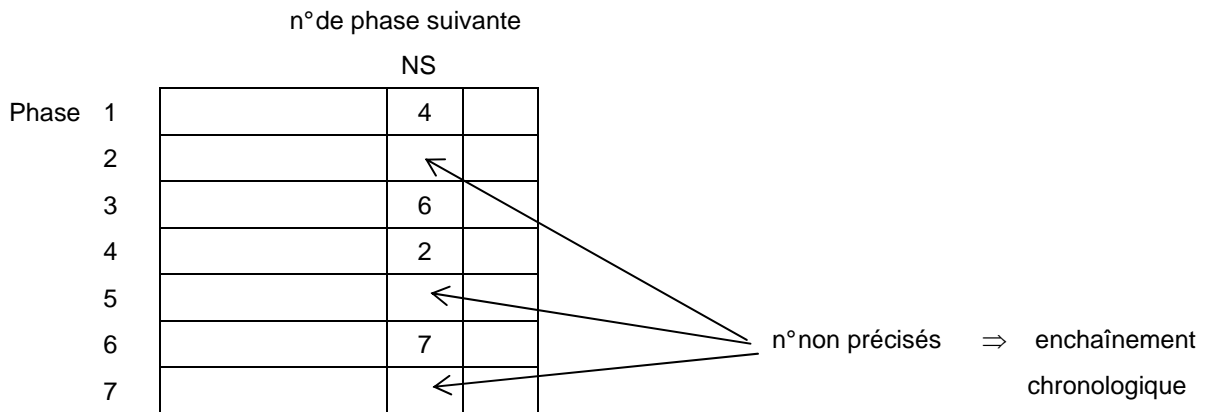
IV.6.5.1 – Enchaînement naturel

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 (directive NS).

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 :



#### IV.6.5.2 – Enchaînement conditionnel

Les entrées logiques et les variables 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 sélectionnées 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).

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/sorties inactive, entrée active

Les entrées /sorties logiques des modules MICROSIMPA et MICROSIMPA 4 fils sont opto-isolées. Celles des MICROSIMPA 4 axes 6 fils sont référencées à la masse électrique.

A défaut l'état inactif des entrées / sorties correspond à leur état de repos : absence de courant. Il est associé à l'état **logique 1**.

Les sauts conditionnels sur les entrées sont associés à l'état **logique 0** correspondant normalement à

- ✓ l'activation de l'optocoupleur d'entrée pour les modules MICROSIMPA et MICROSIMPA 4 fils
- ✓ la mise à la masse électrique pour la carte MICROSIMPA 4 axes 6 fils.

Il est possible de modifier la polarité des entrées logiques :

- ✓ **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)**.
- ✓ **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** (valeur à défaut)  
 Le paramètre **H** associe l'état actif électrique au niveau **logique 1**

##### d) Variables et conditions

La valeur instantanée d'une variable peut être comparée à une valeur constante ou à une autre variable. La séquence peut se dérouter vers 3 phases différentes suivant les valeurs respectives (=, < ou >).



#### IV.6.6 – Phases et séquences réservées

##### IV.6.6.1 – Phase d'arrêt : 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).

Le numéro de phase d'arrêt est 254, les sorties logiques sont laissées en l'état.

##### IV.6.6.2 - Phase d'interruption : 255

L'accès à la phase réservée d'interruption permet de terminer la séquence comme avec une phase d'arrêt et d'enchaîner uniquement la séquence d'interruption. Si la séquence d'interruption n'est pas définie, aucune autre séquence n'est lancée.

Le numéro de phase d'interruption est 255.

La séquence d'interruption est la séquence 99

##### IV.6.6.3 – Séquence butée ou Séquence d'interruption : 99

Lorsque la gestion des butées est autorisée (MB / MN), l'activation d'une entrée butée (but+=in7 / but=-in8) lors d'un mouvement moteur provoque l'arrêt du mouvement et en mode séquence, le passage à la séquence 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.

Les mouvements initiés par cette séquence 99 ne sont pas interrompus par le mode butée. Pour prendre en compte les butées + (in7) et -(in8), il suffit d'introduire le test des entrées (NE ou NF) lors des phases de déplacements (NH, NP, NX, NA, NV, ND) ou des phases de test (#IN.7?0 NS....., #IN.8?0 NS...)

*Nota* : L'activation d'une butée dans un mouvement hors séquence ne provoque pas le lancement de cette séquence mais uniquement l'arrêt du mouvement.

#### IV.6.7 – Enchaînement des séquences et sous séquences

##### IV.6.7.1 – Enchaînement

Il est possible de chaîner plusieurs séquences, grâce à la directive NL, 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 la directive optionnelle 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 une directive NL qui sera chaînée.

**Attention ! Entre chaque séquence, le module MICROSIMPA repasse par son état de repos : moteur arrêté sous tension (opt. standby). Il ne peut pas y avoir de continuité de mouvement. Les sorties, quant à elles ne sont pas modifiées.**

**Commande équivalente : GS, GM, #OUT:=#OUT**

### IV.6.7.2 – Sous séquences

L'exécution d'une séquence peut être momentanément interrompue pour exécuter une sous séquence grâce à la directive NQ. Comme lors du chaînage (NL), le module passe par son état de repos, les sorties logiques sont laissées en l'état.

Attention 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.

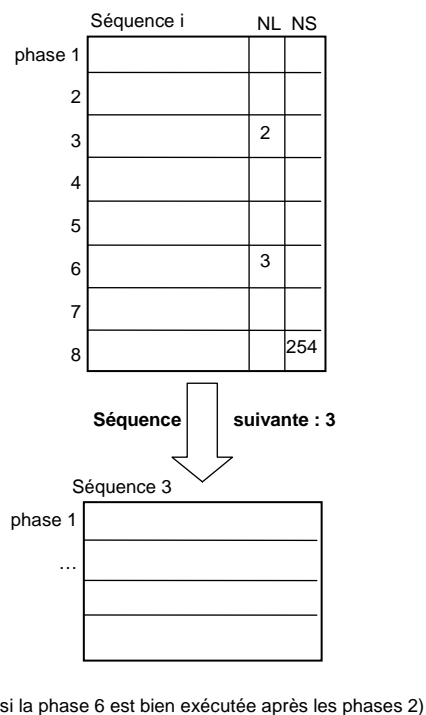
La fin de la sous séquence exécute la phase suivant chronologiquement celle qui a appelé la sous séquence (N° phase +1, ou phase @1 si spécifiée). Plusieurs phases d'une même séquence ou de séquences différentes peuvent appeler la même séquence. Une sous séquence ne diffère en rien d'une séquence classique, 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 : 255" supprime le chaînage prévu ou le retour à la séquence d'appel puisqu'il chaîne obligatoirement la séquence 99.**

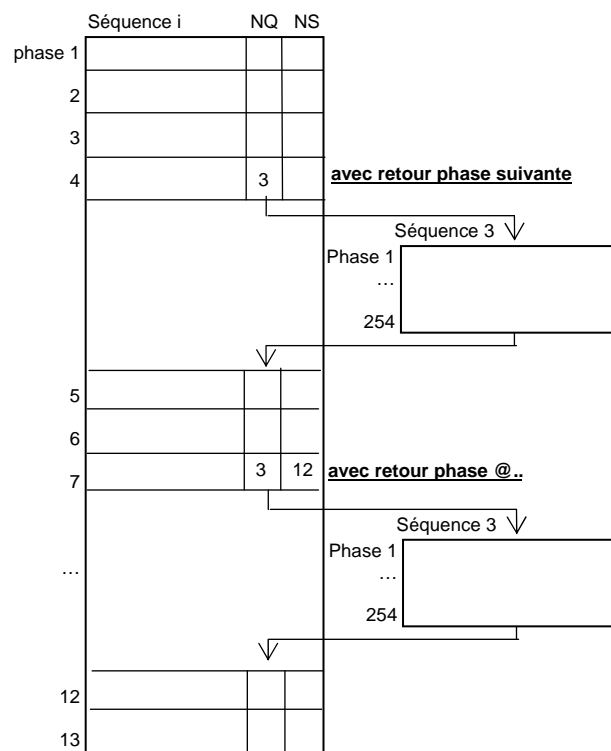
**De plus 1 seul niveau de sous séquence est autorisé : une sous séquence n'a pas le droit d'en appeler une autre !**

### IV.6.7.3 – Exemples

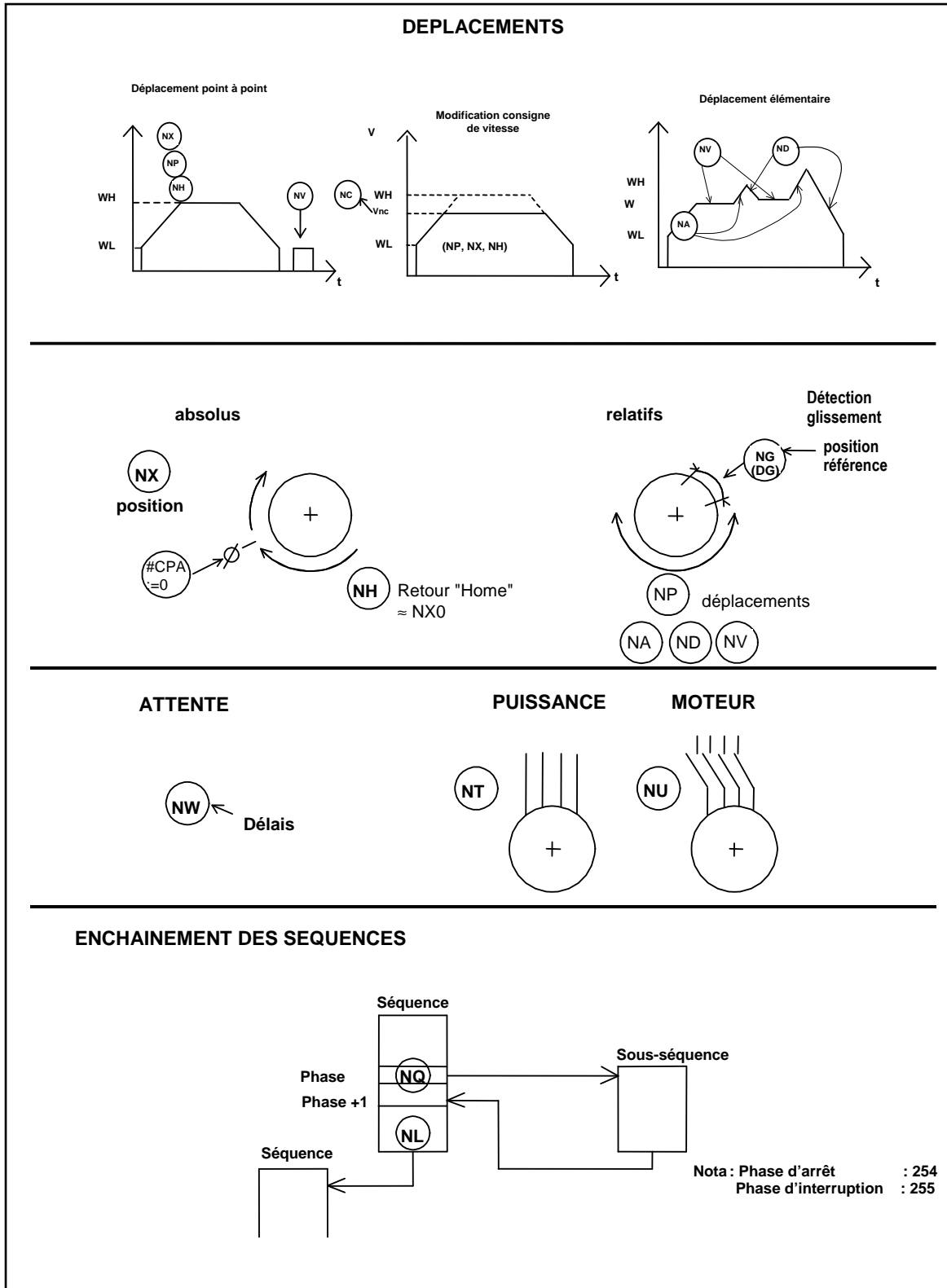
#### Enchaînement de séquences



#### Enchaînement de sous- séquences



IV.6.8 – Résumé des natures de phases





Exemple :

Phase de génération de synchro du module 1 :

SP np<sub>1</sub> + NW t<sub>1</sub> NO FD

Phase d'attente de synchro du module 2 :

SP np<sub>2</sub> + NW t<sub>2</sub> NE 00 00 00 05 00 00 00 00 NS np<sub>2</sub>

Le module 2 boucle sur la phase np<sub>2</sub> en attendant l'activation de son entrée 4 pour exécuter la phase 5.

Le module 1 active sa sortie 2 pour un temps t<sub>1</sub> qui doit être supérieur à la durée t<sub>2</sub> de la phase d'attente du module 2.

## **IV.8 – Gestion des sorties logiques**

### IV.8.1 – Généralités

Les modules MICROSIMPA disposent de 8 sorties logiques indépendantes et totalement gérables par l'utilisateur. Ces sorties peuvent être contrôlées via la liaison série ou dans les séquences grâce à la variable #OUT. Ces actions sont réalisées immédiatement (ou en début de phase dans les séquences).

Il est cependant possible de conditionner la modification des sorties logiques au passage à une valeur définie du compteur de position absolue (CPA) grâce aux commandes GL et GP.

### IV.8.2 – 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). en imposant la variable #OUT, l'indépendance des 8 sorties est obtenue en accédant à #OUT en bit à bit :

#OUT:=OUT impose l'état des 8 sorties logiques du module. Chacune des sorties logiques correspond à l'un des huit poids binaires exprimés par la valeur OUT. Pour modifier une seule sortie logique, il convient d'accéder au bit correspondant de la variable #OUT via la formulation #OUT.n:=v (n numéro de la sortie à modifier, v la valeur à donner 0 ou 1).

En séquence, la directive NO permet de modifier éventuellement plusieurs sorties à la fois au début de chaque phase.

NO OUT [ : MSQ]

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

$$S_i = (OUT * MSQ) + (S_{i-1} * \overline{MSQ})$$

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

IV.8.3 – Gestion différée

La gestion différée est mise en œuvre en utilisant les commandes :

GL OUT : MSQ typ  
et GP Pos [:DEL]

IV.8.3.1 – Absolue, relative

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

- ✓ Absolu (typ=A) : la position de modification : Pos est donnée directement dans la commande GP
- ✓ Relatif (typ=R) : la position de modification est relative à la position courante du compteur de position absolue au moment de l'armement de la fonction par la commande GP. La position de modification est donnée par la valeur Pos 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 (uniquement en fonctionnement absolu).

IV.8.3.2 – Génération d'impulsion

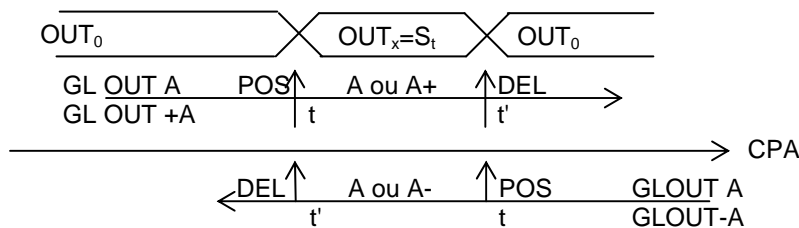
La commande GP permet de définir une "largeur d'impulsion" en définissant une deuxième position (de même type ) pour laquelle les sorties logiques seront complémentées (toujours au travers du masque si nécessaire).

En utilisant conjointement les commandes : GL OUT : MSQ typ et GP 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  $S_{t'} = (\overline{OUT} * MSQ) + (S_{t'-1} * \overline{MSQ})$

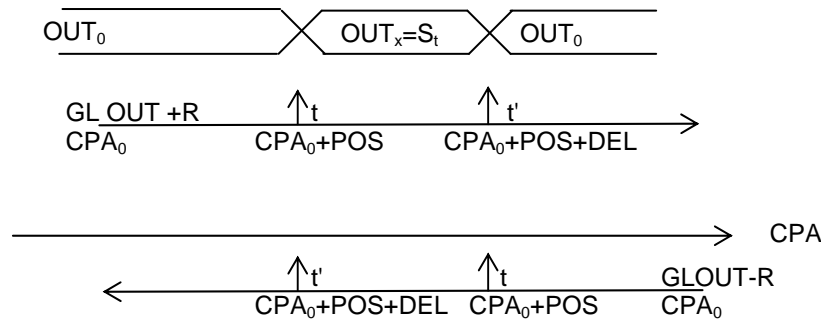
En fonctionnement de type absolu :



t est défini par  $CPA_t = POS$  et  $\begin{cases} CPA_{t-1} < CPA_t & \text{si typ} = +A \\ CPA_{t-1} > CPA_t & \text{si typ} = -A \\ \text{ou typ} = A \end{cases}$

et t' défini de même par  $CPA_{t'} = DEL$  et  $\begin{cases} CPA_{t'-1} < CPA_{t'} & \text{si typ} = +A \\ CPA_{t'-1} > CPA_{t'} & \text{si typ} = -A \end{cases}$

En fonctionnement relatif, si la position du compteur absolu au moment de la commande GP est CPA<sub>0</sub>,



t est défini par  $CPA_t = POS + CPA_0$  et  $\begin{cases} CPA_{t-1} < CPA_t & \text{si typ} = +R \\ CPA_{t-1} > CPA_t & \text{si typ} = -R \end{cases}$

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

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

#### IV.8.3.3 – 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 car la gestion de l'instant de commutation reste, malgré tout, absolue. Le module MICROSIMPA 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).

#### IV.8.3.4 – 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'.

#### IV.8.3.5 – 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 grâce à la variable #OUT 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 (St-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.

### **IV.9 – Etat du module et codes d'erreurs**

#### - Paramètres

La plupart des paramètres de fonctionnement du module peuvent être vérifiés et relus par l'intermédiaire de la liaison série.

#### - Etat à la mise sous tension

Les paramètres de définition des mouvements moteurs sont 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) :

$V_{\min}$	: 75 p/s
$V_{\max}$	: 1000 p/s
$T_r$	: 200 ms
N	: 0 (déplacement à effectuer).
$\mu$	: 1 (fonctionnement en pas entier).
Sens du déplacement	: +
Courant moteur	: 0
Polarité des Entrées Logiques	: actives à 0 électrique (L)
Butées	: inactives
Sorties	: FF

Valeur du compteur de position absolue: 0

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

*Nota : la configuration sortie usine peut être forcée par la commande MRZ.*



- Registre d'état

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 QX

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

- O : surcharge sur les tensions auxiliaires 5V, 24V, 12V fournies par les modules MICROSIMPA et MICROSIMPA 4 fils;  
pour le module MICROSIMPA 4 axes 6 fils : les tensions auxiliaires sont fournies par l'utilisateur, le défaut surcharge n'existe pas.
- u : sous-tension ou microcoupure de l'alimentation (passage sous 12Vdc)
- U : surtension de l'alimentation du module MICROSIMPA  
MICROSIMPA : 90V, MICROSIMPA 4 fils : 50V et MICROSIMPA 4 axes 6 fils : 45V
- X : défaut matériel autres  
module SM4F : surcourant moteur, courant de fuite moteur, dépassement de température  
module MICROSIMPA 4 axes 6 fils : défaut de couplage moteur, surcourant moteur et dépassement de température
- 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
- I : la commande GI est 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 suite à une demande forcée de réinitialisation mémoire (commande MRZ)
- S : passage à la phase 255.

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

- Comportement du module MICROSIMPA sur un défaut

- |                               |  |                    |
|-------------------------------|--|--------------------|
| 1. Lors d'un défaut de type : | –sous-tension ou microcoupure d'alimentation                     | QX ⇒ EE u          |
|                               | –surcharge sur les tensions auxiliaires                          | QX ⇒ EE O          |
| •                             | la puissance moteur est coupée                                   | ~GR                |
| •                             | le mouvement moteur et le déroulement des séquences sont arrêtés | ~GS                |
| •                             | les 8 sorties numériques sont maintenues inactives               | #out := hFF        |
| •                             | les sorties analogiques de la carte MICROSIMPA sont forcées à 0v | #AO1:= 0,#AO2 := 0 |
| •                             | Le led rouge clignote  |                    |

*Remarque: Lors d'un défaut de ce type, hormis la commande MR, seules les commandes de lecture peuvent être exécutées (QR, QX, QV, QD).*

- |                             |   |           |
|-----------------------------|---|-----------|
| 2. Lors d'un défaut de type | – surtension d'alimentation   | QX ⇒ EE U |
|                             | – défaut matériel   | QX ⇒ EE X |
| •                           | la puissance moteur est coupée  | ~GR       |
| •                           | le mouvement moteur et le déroulement des séquences sont arrêtés                | ~GS       |
| •                           | les 8 sorties numériques sont maintenues dans leurs états                       |           |
| •                           | les sorties analogiques de la carte MICROSIMPA sont maintenues dans leurs états |           |
| •                           | Le led rouge est On   |           |

*Remarque: Lors d'un défaut de ce type, il est possible de lancer tous les types commandes.*

*(lecture: QR, QX, QV, QD,... séquence: SE, SN, SP, SF, SS ou SD, mouvement: GF, GO, GA, GH, GE ou GS, puissance moteur: GM ou GR, paramétrage WN,WH,WL,WT, MS., ...)*

*La puissance moteur, le mouvement moteur et le déroulement des séquence ne sont alors bien évidemment pas pris en compte.*

- |   |   |                           |
|---|---|---------------------------|
| 3. Les cartes extensions type '8.In/Out 2.In/Out analog.' | ne réagissent qu'à un défaut sous-tension (<11v)                |                           |
| •   | les 8 sorties numériques sont maintenues inactives              | #out...=hFF               |
| •   | les sorties analogiques de la carte extension sont forcées à 0v | #AOA... := 0,#AOB... := 0 |
| •   | Le led rouge de la carte extension clignote                     |                           |

4. Pour acquitter un défaut :

1. Vérifier le rétablissement des conditions d'utilisation nominal
  - tension d'alimentation correcte : 12v-90v pour la carte MICROSIMPA  
12v-50v pour la carte MICROSIMPA 4Fils  
12v-45v pour la carte MICROSIMPA 4Axes
  - surcharge sur les tensions auxiliaires supprimée
  - suppression des courts-circuits moteurs, température correcte, ...
2. Envoyer la commande MR ou couper puis remettre l'alimentation.

## **V – DESCRIPTION DES COMMANDES**

Seuls sont présentés ici les contenus des commandes en l'absence de tout caractère de protocole, taille et Checksum. Seule l'adresse module est présentée pour préciser le caractère d'adressage unique ou non de la commande.

### **V.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 [ddddddd]

#### **a) Type et taille des paramètres**

Trois types sont différenciés :

- ✓ numériques : d
- ✓ numériques codés hexadécimal : h
- ✓ alphanumériques : a
- ✓ binaire : b

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

Les espaces sont représentés par le caractère '\_'.

Exemples :

- ✓ paramètre de déplacement absolu

Pa = ±dddddd

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\_

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 = + 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.

**V.2 – Liste des commandes**

Les commandes peuvent être réparties en cinq groupes :

- ✓ initialisations,
- ✓ lectures,
- ✓ déplacements,
- ✓ séquences (préparation et exécution),
- ✓ variables.

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

Initialisation			Lecture	Déplacement	Séquences	Variables
Paramètres	Courant moteur	Sorties logiques différées	Etat module	Exécution		
MB	*GI	GL	QC	GA	SS	PO
MN	GM	GP	QD	GE		
MR	GR			GF	SD	
MS			QL	GH	SE	
WH					SN	
WL			QR	GO	SP	
WT			QS	GS	SR	
WN			QV		SF	
			QX			

\*GI : pour module MICROSIMPA 4 fils et MICROSIMPA 4 axes 6 fils.

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

### V.3 – Glossaire des commandes

**GA** : Mouvement absolu

**GA**

Syntaxe :        [@]GA\_Pa (ou [@]GA\_#n)

Paramètres :    Pa = ±ddddddddd : position absolue désirée

Numérique,  
Unité : pas, ½ pas ou micropas suivant la résolution du module (WN),  
10 digits maximum,  
Limites :  $-2147483647 \leq Pa \leq +2147483647$

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

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

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

La consigne du mouvement peut être une variable : GA\_#n.

Exemple :        01GA 14576     ;Le moteur de l'axe 01 va à la position +14576  
                  ...                     ;(le signe + est optionnel)  
                  01GA -1520    ;puis il va à la position absolue -1520 (position en micropas)  
                  01GA #1        ;enfin il va à la position définie par la variable #1

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,
- ✓  $|Pa| > 2^{31}-1$ .

Note :            *La commande GA 0 est équivalente à la commande GH.*  
*La commande équivalente en séquence est NX.*

**\*GE** : Arrêt avec décélération

**\*GE**

Syntaxe :        [@] GE

Paramètres :    Aucun

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

Les prochains mouvements moteurs seront effectués à partir de Vmin.

La commande GE termine la séquence éventuellement en cours.

Exemple :        GF +200        ; sur tous les axes, le moteur tourne à +200 pas/s.  
                  ...  
                  GE                ;le moteur s'arrête en décélérant de Vmax à Vmin.

Code d'erreur    Pas de code d'erreur spécifique positionné.

Note :            *Si le mouvement en cours se fait à vitesse  $v = Vmin$ , l'arrêt est immédiat.*  
*La commande équivalente en séquence est ND.*

## GF : Mouvement à vitesse imposée

## GF

Syntaxe : [ @ ] GF \_ [ ± ] [ v ]

Paramètres : v = dddd 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 -

Ce paramètre ne peut pas être une variable.

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 le sens est maintenu. Si la vitesse n'est pas précisée, 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).

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

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

Exemple : GF +2000 ; Le moteur accélère jusqu'à +2 000 pas/s et se maintient à cette vitesse.

...

GF 1500 ; Le moteur décélère jusqu'à la vitesse de 1500 p/s en suivant la loi de décélération puis maintient le moteur à cette nouvelle vitesse dans le sens du mouvement initial.

Code d'erreur positionnée :

1 paramètre est incorrect

Note : La commande GF n'a pas d'équivalent direct en séquence, Elle correspond approximativement à NA +/-Inf.

## 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 position absolue 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 PO #CPA := ...

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

Exemple : 01GA 14576 ; Le moteur de l'axe 01 va à la position +14576.

⋮

01GH ; puis revient à la position Home (0)

Code d'erreur Pas de code d'erreur spécifique positionné.

Note : La commande GH est équivalente à la commande GA0  
La commande équivalente en séquence est NX 0.

**GI** : Courant moteur (uniquement pour les versions MICROSIMPA avec l'amplificateur intégré)

**GI**

Syntaxe :        [@]GI\_Im

Paramètres :    Im = ddd : amplitude du courant moteur

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

$$I_{\text{moteur}} = \frac{I_{\text{nom}} \times Im}{255}$$

Inom : courant nominal délivré par le module.

Ce paramètre Im ne peut pas être une variable.

Description :    Programmation de la valeur du courant appliqué au moteur.

Exemple :        00GI 128     ;Le courant moteur de l'axe 00 est positionné sur '  $I_{\text{nom}} \times 128 / 255$  '.

Pour un module **MICROSIMPA 4 fils** délivrant un courant Inom de 3Aeff, la commande GI 128 permet de positionner la valeur de courant appliqué au moteur à 1,5Aeff soit 2,1Acrête.

Pour un module **MICROSIMPA 4 axes 6 fils** délivrant un courant Inom de 2Aeff, la commande GI128 permet de positionner la valeur de courant appliquée au moteur à 1Aeff soit 1,4Acrête.

Pour le module MICROSIMPA, cette commande GI n'est pas accessible puisqu'il n'intègre pas d'amplificateur

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  $Im > 255$

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; GP** : Gestion des sorties logiques différées

**\*GL; GP**

**\*GL** : Positionnement des sorties logiques différées

**\*GL**

Syntaxe : [@]GL\_Out : MSQ \_type

Paramètres : Out = hh

Motif hexadécimal à présenter sur les sorties logiques

MSQ = hh

Masque de modification : seuls les bits correspondant aux bits à 1 du masque sont modifiés réellement.

type : +A / -A / A

- positionne au passage croissant (+), décroissant (-) ou indifféremment du CPA à une position absolue définie par la commande GP.

+R / -R

- identique pour un déplacement relatif à la position courante du compteur de position absolue.

Ces paramètres ne peuvent en aucun cas être des variables.

Description : Les sorties logiques des cartes MICROSIMPA permettent d'activer par exemple certains actionneurs parallèlement au déroulement d'une séquence ou d'un mouvement.

- ✓ Cette commande GL permet de contrôler globalement les sorties logiques du module ou partiellement avec le masque.  
L'état logique 1 correspond à une sortie non-actif, l'état logique 0 à une sortie Active.  
L'état des sorties logiques est par défaut non-actif à la mise sous tension et après une commande MR, soit l'équivalent d'un positionnement GL FF  
Lors d'un défaut de type sous-tension d'alimentation ou surcharge d'une alimentation auxiliaire, les sorties logiques sont forcées à l'état non-actif.  
Les sorties logiques sont par ailleurs accessibles grâce à la variable système #OUT
- ✓ Pour activer la gestion différée des entrées/sorties, la commande GL doit être associée à la commande GP.
- ✓ Pour modifier immédiatement les sorties logiques, utiliser plutôt la modification de la variable #OUT.

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

**GP** : Définition de la position de commutation des sorties logiques différées et armement

**GP**

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

Paramètres : p = ± dddddddddd position d'activation relative ou absolue selon le type donné dans la commande GL

d = ± dddddddddd position de désactivation (éventuelle)  
p et d : numériques, unité : incrément de position moteur

M : Fonctionnement répétitif : modifie les sorties à chaque passage aux positions définies par p et d 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.

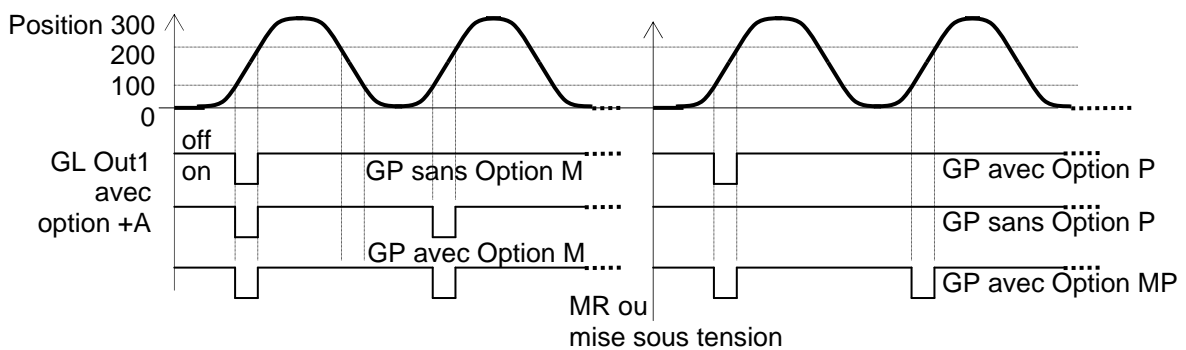
A : arrêt du fonctionnement des sorties logiques différées.

Ces paramètres ne peuvent en aucun cas être des variables.



**Description :** Cette commande GP 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 impulsif 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 impulsif. Si le caractère permanent est demandé, il doit systématiquement être donné dans toutes les commandes GP.

**Exemple :** 00GL00:01 +A ;La sortie Out1 sera activée au passage dans le sens + à la position absolue définie par la commande GP.  
 00GP100:200 P ;entre la position +100 et +200, la première fois et après chaque mise sous tension ou une commande MR  
 00QC ;réponse 00EC GL:00:01 +A GP:+100:+200 P  
 00GP100:200 M ;entre la position +100 et +200, à chaque fois mais pas après une nouvelle mise sous tension ou une commande MR  
 00QC ;réponse 00EC GL:00:01 +A GP:+100:+200 M  
 00GP100:200 MP ;...à chaque fois ETaprès une mise sous tension ou MR.



Voir détail de mise en oeuvre au § IV.8.3

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

**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 (voir les explications données avec la commande GR).**

**Note :** La commande équivalente en séquence est NT  
 La commande inverse (puissance off) est GR

**Code d'erreur :** Pas de code d'erreur spécifique positionné.

**GO** : Mouvement relatif

**GO**

Syntaxe :        [@]GO\_[±][n] (ou GO\_#n)

Paramètres :    n =        ddddddddd valeur de déplacement  
                                   Numérique  
                                   Unité : incrément de position  
                                   10 digits maximum  
                                   Limite  $|n| \leq 2147483647$   
                                   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 d'amplitude n dans le sens explicitement fourni dans la commande.  
                           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 une amplitude identique au dernier mouvement relatif exécuté.

La valeur du déplacement peut être donnée par une variable : GO # n, le sens du déplacement est donné par le signe de la variable.

Exemple :        01GA 2500     ;Le moteur de l'axe 01 va à la position +2500  
                           ...                ;(le signe + est optionnel).  
                           01GO -3200    ;puis il va à la position absolue -700 = 2500 - 3200  
                           01GR            ;La puissance moteur est coupée  
                           01GM            ;La puissance moteur est rétablie.

Code d'erreur positionné :

- 0 :     - Le paramètre "n" n'est pas un paramètre numérique
- 1 :     - Le paramètre est hors limite ( $> 2^{31}-1$ )

Note :            *La commande équivalente en séquence est NP*

**GP** : Commande des sorties logiques déportées

**GP**

Voir commande GL, GP

**\*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.

Exemple :        00GA -2500 ;Le moteur de l'axe 00 va à la position -2500  
                      ... ;puis la puissance moteur est maintenue  
                      00GR ;La puissance moteur est coupée.

**Attention !**

- ✓ Dans le cas d'un module SIMPA indexeur 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.
- ✓ Les dernières versions des modules SIMPA : MICROSIMPA possèdent aussi une sortie Enable activée par la commande GM et désactivée par la commande GR.

Note :            La commande équivalente en séquence est NU  
                      La commande inverse est GM (puissance ON)

Code d'erreur Pas de code d'erreur spécifique positionné.

**\*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.

Exemple :        00GF -250 ;Le moteur de l'axe 00 tourne en sens – à 250 pas/s  
                      ...  
                      00GS ;Le moteur est stoppé instantanément  
                      ... ;et la puissance moteur est maintenue  
                      00GR ;La puissance moteur est coupée

Remarque : L'arrêt s'effectue sans décélération, il peut donc provoquer un glissement ou décrochement du moteur, pour l'éviter, il convient d'utiliser plutôt la commande GE qui arrête le moteur avec décélération.

Code d'erreur Pas de code d'erreur spécifique positionné.







**\*QC** : Lecture des paramètres de gestion des sorties logiques différées

**\*QC**

Syntaxe : @QC

Paramètres : aucun

Format de la réponse :

@EC\_ GL: Sorties : Masque \_ TypeGL\_GP : \_ Pos-activation [ : pos-désactivation ] ModeGP  
@EC\_ GL : hh : hh \_ +/- a \_ GP : ±ddddddddd : ±ddddddddd \_ aa

Description : demande de relecture de l'ensemble des paramètres des sorties logiques différées.

@	Adresse du module interrogé
Sorties :	Valeur des sorties logiques à appliquer
Masque :	Valeur du masque à appliquer
TypeGL :	Type de gestion à appliquer (signé) défini par la commande GL ± A absolue ± R relative
Pos-activation :	Position où les sorties sont programmées en concordance avec le sens défini par la commande GL
pos-désactivation :	Position où les sorties sont dépositionnées en concordance avec le sens (optionnel)
ModeGP :	Mode de répétition défini par la commande GP P permanent M répétitif

Exemple : 00MRZ ;Reset 'sortie usine'  
00QC ;réponse 00EC GL:00:FF R GP:+0

00GL00:01 +A ;La sortie Out1 sera activée au passage Absolue et dans le sens +  
00GP100:200 P ;entre la position +100 et +200 , la première fois après une mise sous tension ou une commande MR  
00QC ;réponse 00EC GL:00:01 +A GP:+100:+200 P

**\*QD** : Suivi des mouvements et séquences

**\*QD**

Syntaxe : @QD

Paramètres : aucun

Format de la réponse :

@ED\_Séquence\_Phase\_Sens\_Nature\_Position\_Entrées\_Sorties\_Contrôle\_Séq.Suivante\_cmp  
@ED\_ddd\_ddd\_±\_aa\_±dddddddddd\_hh\_hh\_aa\_ddd\_a

Soit un total de 45 caractères au maximum

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

Quel que soit l'état de fonctionnement du module, en mouvement ou non, en cours d'exécution de séquence il est possible de suivre le fonctionnement du module grâce aux paramètres suivants :

@ 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 en cours  
Position : Valeur du compteur de position absolue.  
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 : Deux caractères précisant :  
1) Etat de l'automate  
attente ou exécution d'une commande immédiate : L  
séquence en cours d'exécution : S  
sous-séquence en cours d'exécution : U  
entrée STOP (J11) active : A  
2) Etat du moteur  
hors puissance : F  
sous tension : O  
Séq.suivante : N° de la prochaine séquence (valable si séquence en cours), 0 pour aucune  
cmp : code d'erreur (alphanumérique)... voir commande QX  
**Attention ! Le code d'erreur n'est pas acquitté.**

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

Exemple : 00QD

si on obtient 00ED **20 7 + NW +99200 FF F7 UO 90 N** (exemple issu de l'annexe **Guide pratique**)  
alors on a Phase **7** de la Séquence **20** en cours d'exécution  
sens de rotation **+**, **NW** exécute la temporisation, position actuelle **+99200**,  
**FF** (aucune entrée active), **F7** Out4=Act. (seule la sortie 4 est active),  
soUs- Séquence et moteur **On** (sous tension),  
Séquence suivante **90** (hormis retour séq. d'appel), état **Nominal** (pas d'erreur)

00MRZ ;Remise à l'état d'origine du module  
00QD ;  
réponse 00ED **0 0 + XX +0 FF FF LF 0 M**  
ce qui correspond à **0** Pas de séquence, **0** ni de phase en cours,  
sens de rotation à défaut **+**, **XX**, n'exécute pas de commande, Position courante **+0**,  
**FF** (aucune entrée active), **FF** (aucune sortie active), **L** attente de commande et moteur **oFf** (hors puissance),  
**0** pas de séquence suivante et **reMise** en configuration sortie usine.



**\*QL** : Relecture des paramètres locaux

**\*QL**

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:dddd\_WH:dddd\_WT:dddd[:dddd]\_WN:ddd DR:±ddddddd\_GI:ddd\_DG:ddd\_MD :da\_aa\_a  
seuls les chiffres significatifs sont retournés

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

@	adresse du module interrogé	
Vmin	vitesse minimum en pas/s (paramètre de la commande WL)*	
Vmax	vitesse maximum en pas/s (paramètre de la commande WH)*	
ta	durée de l'accélération en ms (paramètre de la commande WT)	
td	durée de la décélération en ms (si différent de ta)	
résolution	résolution programmée en micropas/pas (paramètre de la commande WN)	
consigne	consigne du mouvement relatif en micropas mémorisée lors de la commande GO ±n	
courant	consigne du courant moteur (paramètre de la commande GI)	
glissement	valeur du glissement maximum toléré en micropas (commande DG)	
mode	0 = carte micropas et	S : gestion du standby (mode par défaut) B : gestion du standby et du surcourant N : forçage du courant nominal (voir commande MS)
Butée	MN	butées hard non gérées
	MB	butées hard gérées
Polarité	L	entrées actives à 0 (mode par défaut)
	H	entrées actives à 1

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

Exemple : 00MRZ ; configuration 'sortie usine' du module MICROSIMPA  
00QL ; Après cette initialisation, on obtient  
00EL WL:75 WH:1000 WT:200 WN:1 DR:+0 GI:0 DG:10 MD:0S MN L  
soit Vmin=75pas/s, Vmax=1000pas/s, Durée acc./déc=200ms,  
Résolution =1µpas/pas, consigne=+0, Courant nom.=0,  
glissement=10, MICROSIMPA(0) avec gestion Standby,  
Butée hard non gérées et entrées actives à 0(L)

\*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.

**\*QR** : Relecture d'une variable

**\*QR**

Syntaxe : @QR\_#n [\_#m\_#...] [\_H] [B]

Paramètres : n, [m...] = dd : numéro de variable utilisateur  $1 \leq n \leq 32$  ou  $M1 \leq n \leq M32$   
ou aaa : nom de la variable système (voir liste § IV.5)

H L'ajout du suffixe 'H' à la fin de la commande permet une relecture en hexadécimal de la variable.

B L'ajout du suffixe 'B' à la fin de la commande permet une relecture en binaire.

Format de la réponse :

en décimal @#n = v<sub>n</sub> [\_#m = v<sub>m</sub>...]  
@#dd = ±ddddddddd [\_#dd = ±ddddddddd] (ou #aaa=±ddddddddd)

en Hexadécimal @#n = Hv<sub>n</sub> [\_#m = Hv<sub>m</sub>...]  
@#dd= Hhhhhhhh [\_#dd = Hhhhhhhh] (ou #aaa=Hhhhhhhh)

en binaire @#n = Bv<sub>n</sub> [\_#m = Bv<sub>m</sub>...]  
@#dd= Bbbbbbb... [\_#dd = Bbbbb...] (ou #aaa=Bbbbb...)

Seuls les chiffres significatifs sont retournés.

Description : Commande de relecture d'une variable #n. Renvoie la valeur de la variable #n : v<sub>n</sub> en décimal ou Hv<sub>n</sub> en hexadécimal avec l'option 'H'  
Il est possible de demander simultanément la relecture d'au maximum 10 variables.  
Les noms des variables doivent simplement être séparés par des espaces :  
QR\_#n\_#m\_#p... en retour @ #n = v<sub>n</sub> #m=v<sub>m</sub> #p=v<sub>p</sub>...

Exemple :

```
00MR                                     ;définit les variables #18 et #M18
00#18 :=120, #M18 := 300                 ;Va en position 1220
00GA 1220                                 ;demande de relecture en decimal.
00QR #18 #M18 #CPA                       ;demande de relecture en Hexa.
00#18=120 #M18=300 #CPA=1220
00QR #18 #M18 H
00#18=H78 #M18=H12C
00#OUT.2 := 0                             ;active la sortie 3
00QR #OUT B                               ;demande de relecture binaire
00 #OUT = B11111011
```

Code d'erreur positionné :

0 : n° de variable erroné, absence de nom de variable, ...

Nota : *la taille du buffer utilisé par la MICROSIMPA est limitée à 127 caractères si celui-ci est dépassé, seules les premières valeurs sont retournées suivies par un point d'interrogation "?"*



**\*QV** : Demande d'identification

**\*QV**

Syntaxe : @QV [Xn]

Paramètres : n numéro de module d'extension (optionnel)

Description : demande d'identification du module MICROSIMPA (sans paramètre) ou du module d'extension connecté n°n.

Format de la réponse :

@EV Vv.r code texte  
@EV\_Vd.d\_aaaa\_"aaaaa...aa"  
ou EV Xn=Vv.z code texte  
ou EV Xd=Vd.d code texte

n : numéro du module d'extension  
v : numéro de version  
r : numéro d'indice  
code : information réservée pour Midi Ingénierie sur le type de carte  
Texte : référence exacte du produit, n°de série, identificateur,...

Exemples :

- Pour un module MICROSIMPA :  
00EV V2.1 A157 "MIDI-INGENIERIE\_US01\_A157-0074\_09/11/05\_00/00\_00\_" "A157-10000-A-0A0-021"
- Pour un module MICROSIMPA 4 fils:  
00EV V2.1 B157 "MIDI-INGENIERIE\_US41-3\_B157-0074\_09/11/05\_00/00\_00\_" "B157-10000-A-0A0-021"
- Pour un module MICROSIMPA 4 axes 6 fils :  
00EV V2.1 9160 "MIDI-INGENIERIE\_US64-2\_9160-0074\_09/11/05\_00/00\_00\_" "9160-11000-A-000-021"
- Pour un module d'extension entrées/sorties :  
00EV 3V.1.4 1x157-0012

**\*QX** : Demande d'état

**\*QX**

Syntaxe : @QX

Paramètres : aucun

Format de la réponse : @EE\_Code\_état

Description : Lorsque le module détecte une anomalie sur une commande, il positionne un code d'erreur qui est lu au moyen de la commande QX.

@ : 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 efface le code d'erreur et impose 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.

**Voir le § IV.9 "état du module et codes d'erreurs" pour trouver la signification des codes d'erreurs retournés.**

Exemple

00QX...	00EE_N	
00GI -10		; commande erronée car valeur signée
00QX...	00EE_0	; retourne le code d'erreur 0 et l'efface
00QX...	00EE_N	; code Nominal, pas d'erreur car précédemment corrigée

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

## SD : Séquence de démarrage

## SD

Syntaxe :        [@]SD\_ns

Paramètres :    ns =       ddd n° de la séquence à démarrer à la mise sous tension ou MR  
                                  Numérique,  
                                  Non signé,  
                                  Limite du paramètre :  $1 \leq N^\circ \text{ de séquence} \leq 99$

Ce paramètre ne peut pas être une variable

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

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

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.

Contrainte        La séquence doit être créée avant de la sélectionner.

Remarque        La commande SR permet d'arrêter le démarrage automatique d'une séquence sans toutefois l'effacer.  
                          Une séquence peut être lancée manuellement grâce à la commande SS.

Exemple            01SD11 ;programme la séquence 11 comme séquence automatique au  
                          .....démarrage pour le module MICROSIMPA à l'adresse 01  
                          01MR       ;reset le module 01.....la séquence 11 est lancée.  
                          .....  
                          01GS,SR ;arrête les séquences et mouvements en cours et supprime le démarrage  
                          .....automatique de la séquence 11 du module MICROSIMPA d'adresse 01.

### Code d'erreur positionné

- 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.

## SE : Effacement de séquence

## SE

Syntaxe :        [@]SE\_ns

Paramètres :    ns =     ddd : n° de la séquence à effacer  
                                  Numérique,  
                                  Non signé,  
                                  Limite du paramètre :  $1 \leq N^{\circ} \text{ de séquence} \leq 99$

Description :    effacement de la séquence ayant pour numéro le paramètre ns.  
                          Tout l'espace mémoire réservé précédemment pour décrire les phases de cette séquence redevient disponible pour la définition d'autres séquences.

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

Exemple :         SE 12                                 ;efface les séquences ayant pour numéro 12 sur tous les axes  
                          00SE0                                 ;effacement de toutes les séquences de l'axe 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 (< 0 ou > 99)

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

## SF : Sauvegarde en flash d'une séquence

## SF

Syntaxe :        [@]SF

Paramètres :    aucun.

Description :    La commande SF doit être utilisée conjointement avec la commande SN pour transférer le contenu de la séquence désirée écrite en RAM dans la flash aux fins de sauvegarde et d'exécution.

Ce transfert doit être effectué avant toute nouvelle ouverture de séquence ou tout nouveau calcul de loi (commandes WH, WL, WN, WT).

Exemple :         00SN01                                 ; création de la séquence n°1  
                          00SP01 1 NP + 1000 ;  
                          00SP01 2 NW 100 ; description des phases de la séquence n°1  
                          00SP01 3 NP - 1000 ;  
                          **00SF**                                 ; sauvegarde en flash OBLIGATOIRE avant exécution  
                          .....  
                          00SS1                                 ; exécution  
                          .....  
                          00SD1                                 ; démarrage automatique de la séquence  
                          00MR                                 ; redémarre le module (équivalent à une mise sous  
                          tension)  
                          .....  
                          ; =>relance la séquence  
                          00GS,SR                                 ;supprime le démarrage automatique de la séquence  
                          00GS,SE01                                 ;efface la séquence 01

Code d'erreur positionné : pas de code d'erreur positionné

## SN : Création d'une séquence

## SN

Syntaxe :        [@]SN\_ns

Paramètres :    ns =     ddd : numéro de la séquence à créer

Numérique,  
Non signé,  
2 digits maximum,  
Limites :  $1 \leq N^{\circ} \text{ de Séquence} \leq 99$

Description :    cette commande permet d'ouvrir une nouvelle séquence pour écriture et définition des phases.

- ✓ 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.

Exemple :        Création de la séquence n° 2 sur l'axe 01  
01GS,SE 2         ;arrete tous mvt. et efface Séq.2 (si la séquence 2 existe déjà)  
01SN 2             ;pour créer la nouvelle séquence

### Code d'erreur positionné

0 : Défaut sur les paramètres de la commande.  
Il n'y a pas le nombre exact de paramètres (1).  
Les paramètres sont non-numériques, ou signés.

1 : Le N° de Séquence fourni est hors limite

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

Contrainte        Il est impossible de modifier une séquence déjà existante. Il faut l'effacer et la recréer. On ne peut pas créer une séquence déjà existante (code erreur 4).

Il est possible de créer jusqu'à 10 séquences de 128 phases  
et 22 séquences de 32 phases.

Nota :            *Il convient de sauvegarder la séquence en flash grâce à la commande SF, préalablement à son exécution. L'ensemble de la séquence doit être réécrite à chaque création.*

## SP : Définition de la phase

## SP

### Syntaxe :

[@]SP\_np \_Na\_Cns[\_NL\_sc\_NQ\_sq][\_Ne\_X1\_... X8][\_NS\_ps] NS@1[@2 @3][\_NO\_out][:msq]]

### Description :

Ecriture d'une phase définie par des commandes et paramètres  
 Un ensemble de phases forme une séquence(SN...SF) qui peut être exécutée  
 manuellement (SS), automatiquement (SD) ou chaînée (NL, NQ)

### Paramètres :

Chaque paramètre est décrit ci-après (§ a .... i)

#### a) n° de phase

\* np = ddd : numéro de la phase dans la séquence

Les phases peuvent être décrites dans n'importe quel ordre

#### b) Nature de phase

\* Na = aa : nature de la phase

permet de spécifier le type de fonction ou commande à exécuter :

#### Gestion des mouvements

NP	: mouvement Relatif	commande directe équivalente
NH	: retour Origine	GO
NX	: mouvement Absolu	GH
NV	: à vitesse constante	GA
NA	: accélération	
NC	: modification de la consigne de vitesse palier	
ND	: décélération	

#### Gestion puissance moteur et délais d'attente

NT	: activation de la Puissance Moteur
NU	: suppression de la Puissance Moteur
NW	: attente

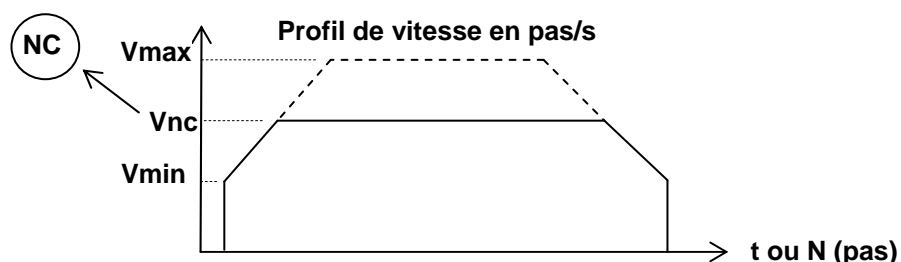
#### Gestion des variables

[PO]	: calcul et test sur les variables
------	------------------------------------



b1) Gestion des mouvements :

- NP, NX : Mouvements point à point relatifs (NP) et absolus (NX). La consigne fixe respectivement la variation de position à réaliser ou la nouvelle position absolue à atteindre. Comme pour les mouvements immédiats (voir commandes GO et GA), ceux-ci sont décomposés en 3 étapes : accélération, palier à vitesse constante, décélération.
- NH : Mouvement point à point 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 relatif à vitesse constante. La consigne fixe la variation de position à décrire durant la phase. La vitesse de la phase est la vitesse  $V_{min}$ , 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. Les variables système ne peuvent être utilisées comme consigne, seules les variables utilisateur le sont.
- NA, ND : Mouvements relatifs accélérés (NA) ou décélérés (ND). La consigne fixe le déplacement 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  $\omega_L$  et  $\omega_H$ .  
Pour qu'il y ait réellement changement de vitesse, il convient que le nombre de pas défini par la consigne soit supérieur au nombre de pas à réaliser sur le palier courant de la loi de vitesse.  
Lorsqu'une vitesse limite est atteinte, le mouvement continue à vitesse constante jusqu'à la réalisation complète de la variation définie par la consigne.  
Si une entrée logique détectée sur front est activée, la vitesse moteur est maintenue constante sur éventuellement quelques incréments 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 de la consigne, le mouvement moteur est maintenu à vitesse constante jusqu'à la fin du chargement.  
De plus, à vitesse élevée, la génération des incréments de position s'effectue par paquets pour des raisons liées au temps de traitement microprocesseur, aussi le déplacement obtenu 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 peuvent être générés.  
*Remarque* : la valeur minimum des consignes des phases NA et ND est 2, il n'est pas possible d'utiliser le contenu d'une variable système, seules les variables utilisateur peuvent être utilisées comme consigne.
- 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  $V_{nc}$  permet de tronquer provisoirement la loi d'accélération initiale. Lors des mouvements point à point (NP, NXW, NH)

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  $V_{nc}$  inférieure à  $V_{min}$  (différent de 0) impose un fonctionnement uniquement à vitesse min.

Une valeur  $V_{nc}$  supérieure à  $V_{max}$  restitue le fonctionnement avec la loi de vitesse initiale.

La valeur de consigne  $V_{nc}$  est conservée lors de l'appel d'une sous-séquence et au retour de celle-ci même si cette valeur a été programmée dans la sous-séquence.

La consigne de la commande NC peut pas être une variable utilisateur (#1...#32, #M1...#M32).

Dans ce cas, lorsque la variable est inférieure ou égale à  $V_{min}$ , la consigne est fixée à  $V_{min}$ .

Lorsque la variable est supérieure ou égale à  $V_{max}$ , la consigne est fixée à  $V_{max}$ .

Exemple : avec une vitesse min  $Wl=500p/s$ , une vitesse max  $Wh=1500p/s$ , une résolution  $Wn=16\mu pas/p$  un temps d'accélération  $Ta=500ms$  et de décélération  $Td=300ms$   $WT500:300$

⇒ nombre de micropas en accélération :  $(1500+500)/2 \times 0.5s \times 16 \mu pas/pas = 8000 \mu pas$

et nombre de micropas en décélération :  $(1500+500)/2 \times 0.3s \times 16 \mu pas/pas = 4800 \mu pas$

SP 1 NP 30000 déplace le moteur de 30000 incréments de position en suivant la loi d'accélération

SP 2 NX 12000 amène le moteur à la position 12000 en suivant la loi d'accélération

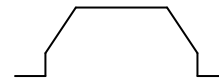
SP 3 NH ramène le moteur à sa position d'origine :0 en suivant la loi d'accélération

Avec une loi comportant 8000 incréments en accélération et 4800 en décélération. La suite de phases suivantes est équivalente à la phase 1 décrite plus haut de 30000 micropas.

SP 4 NA 8000 ; 8000 $\mu pas$  en 0,5s

SP 5 NV 17200 ;17200 $\mu pas$  en 0,717s

SP 6 ND 4800 ; 4800 $\mu pas$  en 0,3s



Les suites de phases suivantes provoquent les mêmes mouvements que la première phase décrite mais à vitesse palier de 1000 p/s au lieu de 1500 p/s à défaut.

SP 7 NC 1000

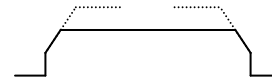
SP 8 NP 30000

⇔

SP 9 NA 3000 ;  $3000=(500+1000)/2 \times 0,25 \times 16$  en 0,25s

SP 10 NV 25200 ;  $25200= 30000 - 3000 - 1800$  en 1,575s à 1000pas/s

SP 11 ND 1800 ;  $1800=(500+1000)/2 \times 0,15 \times 16$  en 0,15s



## b2) Gestion puissance moteur et délais d'attente

- NT, NU : Respectivement, établissement 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.

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).

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

Exemple :  
 SP 20 NW 150      provoque simplement une attente de 150 ms  
           SP 21 NT 300      remet le moteur sous tension  
                                   et attend 300 ms avant de passer à la phase suivante.  
           SP 22 NU 13      coupe la puissance moteur et temporise 13 ms.

b3) Gestion des variables

[ PO ]: Modification de variable et calculs sur les variables

**La nature de phase PO est implicite. Son nom [ PO ] peut donc être omis comme l'indique l'écriture entre crochets [ PO ]. Elle est cependant toujours retournée dans la relecture du contenu de la phase par la commande QS : PO#1:=#3+#M4 ⇔ #1:=#3+#M4**

**Forme n°1 :** [ PO ]\_#n:=\_v ou [ PO ]\_#n:=\_#m  
Simple affectation, #n prend la valeur v = ± ddddddddd en décimal  
ou v = Hhhhhhhh en hexadécimal  
ou v = Bbbbbbbb en binaire  
ou respectivement #n prend la valeur de #m

**Forme n°2 :** [ PO ]\_#n : =\_#m\_(op)\_v ou [ PO ]\_#n:=\_#m\_(op)\_#p  
Réalise l'opération #m (op) v et place le résultat dans la variable #n  
ou respectivement #m (op) #p et place le résultat dans la variable #n

Avec (op)  
+ addition  
- soustraction  
\* multiplication  
/ division entière  
& et logique  
| ou logique  
^ ou exclusif  
> rotation vers la droite sur 32 bits  
< rotation vers la gauche sur 32 bits

Test sur variable

La fonction PO permet de conditionner le déroulement de la séquence à la valeur courante d'une variable ou de gérer un "compteur de boucle".

Il convient dans ce cas d'associer la directive NS dans sa forme étendue :

NS @1 [ : @2 : @3 ] qui permet de donner jusqu'à 3 adresses de phase suivante.

**Forme n°3 :** [ PO ]\_# n?\_v NS @1 : @2 : @3 avec v = ± ddddddddd en décimal  
ou v = Hhhhhhhh en hexadécimal  
ou v = Bbbbbbbb en binaire  
ou [ PO ]\_# n?\_#m NS @1 : @2 : @3

Compare la valeur courante de la variable n à la valeur v ou respectivement à #m et sélectionne en conséquence la phase suivante à exécuter.

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

Note : Les 3 adresses de sauts @1 @2 @3 sont obligatoires, même si certaines doivent être identiques.

Les variables #n et #m ne sont pas modifiées

**Forme n°4 :** [PO]\_# n\_:= #m\_ (op) \_v NS @1 : @2 : @3

ou [PO]\_# n\_:=\_#m\_ (op)\_#p NS @1 : @2 : @3

Le résultat de l'opération réalisée est comparé à 0 et la phase suivante devient :

@1 si résultat = 0  
@2 si résultat < 0  
@3 si résultat > 0

*Note :* Dans ce cas les 3 adresses de saut @1 @2 @3 sont obligatoires.

Exemples : au préalable les variables #1 à #5 et #CPA ont les valeurs suivantes :  
#1=100 #2=-23 #3=29 #4=10 #5=-64 et la position moteur  
#CPA=1000

Les phases suivantes de la séquence 02 de l'axe 00 réalisent :

00SN 02

...

00SP 30 #1 := #CPA ; #1= 1000 (CPA)

00SP 31 #4 := 12 ; #4= 12

00SP 32 #5 := #5 + #2 ; #5= -87 ( -64 + -23 )

00SP 33 #5 := #5 / #3 ; #5= -3 ( -87 / 29 )

00SP 34 #2 ? #5 NS 50:51:52; saut à la phase 51(-23<-3)

...

**La commande PO n'est pas compatible avec NE/NF.**

**La commande PO avec test (forme 3 et 4) n'est pas compatible avec NL et NQ.**

### c) Consigne

\* Cns = [±]ddddddddd : consigne

Cette consigne peut être :

#### - un déplacement relatif dans les commandes

NA = accélération sur un déplacement de valeur Cns

NV = vitesse constante sur un déplacement de valeur Cns

ND = décélération sur un déplacement de valeur Cns

NP = déplacement de valeur Cns en respectant la loi d'accélération

Le sens du mouvement est déterminé par le signe de la consigne.

La valeur est bornée par  $-2147483647 \leq \text{Cns} \leq 2147483647$

#### - une position absolue

Le paramètre représente la position que doit atteindre le moteur pour les commandes NX.

Format numérique signé ou non  $-2147483647 \leq \text{Cns} \leq 2147483647$

#### - un délai

Pour les fonctions NW, NU, NT, le délai est donné en ms (1 à 65535 ms).

#### - une vitesse

En pas/s pour les phases NC

#### - Consigne variable

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

Ex : NP\_#3 ou NX\_#M6...

#### - Une fonction

Pour la commande PO, la consigne décrit la fonction à réaliser.

d) Séquence chaînée Directive NL\_sc (directive optionnelle)

\*sc = ddd : numéro de séquence chaînée

Cette directive précise le numéro de la nouvelle séquence à exécuter dès que la séquence en cours se termine.

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. Par voie de conséquence, il n'y a pas de chaînage de séquence tant que le module ne rencontre pas cette directive dans la description d'une phase exécutée.

Pour modifier ou définir un numéro de séquence à chaîner, la phase contenant l'ordre NL doit obligatoirement être exécutée (pas seulement définie).

e) Sous-séquence Directive NQ\_sq (directive optionnelle)

\*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 puis l'exécute entièrement. L'exécution de la séquence initiale reprend à partir de la phase naturelle suivant la phase ayant provoqué le branchement à la sous-séquence ou lance la phase @1 si spécifiée par NS@1.

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 NQ),
- 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.**

f) Test des entrées logiques Directives NE ou NF (directive optionnelle)

\*Ne :  
aa\_ddd\_ddd\_ddd\_ddd\_ddd\_ddd\_ddd  
sélection des entrées conditionnelles

NE X<sub>1</sub> X<sub>2</sub> X<sub>3</sub> X<sub>4</sub> X<sub>5</sub> X<sub>6</sub> X<sub>7</sub> X<sub>8</sub> : gestion sur état

ou NF X<sub>1</sub> X<sub>2</sub> X<sub>3</sub> X<sub>4</sub> X<sub>5</sub> X<sub>6</sub> X<sub>7</sub> X<sub>8</sub> : 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<sub>1</sub> X<sub>2</sub> X<sub>3</sub> X<sub>4</sub> X<sub>5</sub> X<sub>6</sub> X<sub>7</sub> X<sub>8</sub> précisent les 8 adresses de phases rattachées aux 8 entrées logiques, respectivement IN1, IN2, ..., IN8.

Seules 6 entrées peuvent être testées simultanément. Au minimum, deux paramètres Xi doivent donc être 0.

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.

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.

**Nota : seules 6 entrées peuvent être testées simultanément.**

### Exemple

1 \* Sauter à la phase n° 7 dès l'apparition de l'entrée logique n° 3 lors de l'exécution de la phase 2 :

(IN1)(IN2)(IN3)(IN4)(IN5)(IN6)(IN7)(IN8)  
SP\_2 ...      NF\_00\_00\_07\_00\_00\_00\_00

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

2 \* Sauter à la phase n° 5 si l'entrée logique n° 5 est présente en fin de phase 4.

SP\_4 ...      NE\_00\_00\_00\_00\_05\_00\_00\_00...

g) Phase suivante      Directive NS @1      (directive optionnelle)

\* @1 = dd : n° de la phase suivante

Permet de forcer l'ordre de succession des phases en imposant un numéro de phase suivante, différent de l'ordre chronologique.

Cette information est optionnelle : par défaut la phase naturelle suivante a pour adresse l'adresse de la phase en cours + 1.

L'appel de la phase 254 (NS 254) ou le saut à une phase n'existant pas provoque l'arrêt de la séquence en cours.

Si la directive "NL\_sc" a été programmée ET exécutée lors du déroulement de la séquence, l'automate lance alors la séquence "sc" après un saut NS 254 ou un arrêt.

Dans le cas d'arrêt d'une sous séquence, c'est la séquence d'appel qui continue.

L'appel de la phase 255 (NS 255) provoque l'arrêt de la séquence en cours ET l'exécution de la séquence d'interruption "sq.99" (même lors du déroulement d'une sous-séquence).

Si cette séquence 99 n'existe pas, l'automate est arrêté (pour plus de précisions, voir chapitre IV.6.6).

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

SP\_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\_2 ...      NE\_03\_00\_00\_00\_00\_00\_00\_00\_NS\_02.

Cette phase reboucle sur elle-même (NS02) tant que la condition logique IN1 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\_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.

h) Sorties logiques Directive NO\_OUT:msq (directive optionnelle)

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

Cette directive permet de configurer l'état des sorties logiques du module dès le début de 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.

**Exemple** # n = h h h h  
msq ←      → out

NO\_A5 :      place les sorties logiques dans l'état      1010 0101 = (hA5)  
                 soit Out 7,5,4 et 2 = Actif                      S8.....S1  
                 et Out8,6,3et1= non-actif

NO\_00 : 04      active la sortie logique 3 sans modifier les 7 autres  
                 équivalent à #OUT.3 :=0

i) 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 ou le numéro de séquence choisi).

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.

Remarque :      Cette commande ne peut être exécutée lorsqu'une séquence est en cours.  
Il faut préalablement arrêter les séquences en cours grâce à la commande GS

Exemple :        00GS,SR           ;supprime le démarrage automatique de séquence même si une séquence est en cours d'exécution (celle-ci est préalablement arrêtée).

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 :  $1 \leq N^{\circ} \text{ de séquence} \leq 99$

Description :    exécution immédiate de la séquence demandée.

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

Exemple :        00SS15           ;l'Axe0 exécute la séquence 15

Remarque :      La commande GS arrête une séquence est en cours.  
Une séquence de démarrage peut être programmée grâce à la commande SD.

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,WL,WN,WT : loi de mouvement

## WH,WL,WN,WT

Syntaxe : [ @ ] WL\_wl, WH\_wh, WT\_tr, WN\_wn avec un temps de rampe identique

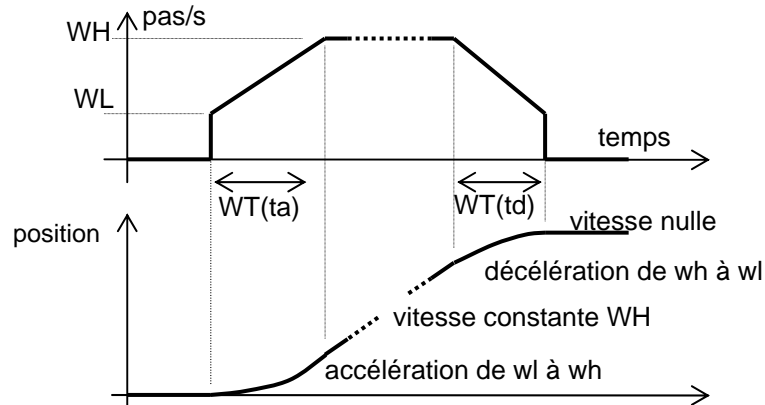
[ @ ] WL\_wl, WH\_wh, WT\_ta:td, WN\_wn pour des temps d'accélération et de décélération différents

Paramètres :

	wl	wh	tr ou ta:td	wn
définition	vitesse de consigne Vmin	vitesse de consigne Vmax	temps de rampe ou temps accélération:décélération	Résolution $\mu$
unité :	pas/s	pas/s	milli seconde	micropas/pas
donnée :	Numérique, Non signé, 5 digits maximums.	Numérique, Non signé, 5 digits maximums.	Numérique, Non signé, 5 digits maximums.	Numérique, Non signé, 3 digits maximums.
limite	$1 \leq wl < 20\ 000$	$1 < wh \leq 20\ 000$	$1 < t_{ms} < 65535\ ms$	MICROSIMPA $1 \leq \mu \leq 256$  MICROSIMPA 4 fils ou MICROSIMPA 4 axes 6 fils 1,2,4,8,16,32,64

Les paramètres de la loi de mouvement ne peuvent pas être définis par des variables mais seulement au moyen de constantes.

Description :



wl :Vmin définit la vitesse de démarrage du moteur et donc sa vitesse minimum.

wh :Vmax définit la vitesse maximum que pourra prendre le moteur, c'est la vitesse palier de la loi de mouvement et donc la vitesse de consigne du moteur.

ta, td définissent respectivement les temps d'accélération ou décélération pour passer de Vmin à Vmax et inversement tr permet de définir simultanément Ta et td lorsqu'il sont égaux  
 $ta = td = tr$

wn :  $\mu$  définit la résolution du driver en micropas par pas.

Vmin, Vmax et Trampe ne dépendent pas de la résolution en micropas par pas choisie.

On peut modifier la résolution sans que la vitesse finale du moteur ne soit modifiée, seule la vitesse de génération des micropas, quant à elle, est multipliée par wn.

Note : Dans le cas d'un couple indexeur MICROSIMPA et d'un amplificateur, MAC23 par exemple, il convient en général que la résolution programmée WN corresponde à la résolution de l'amplificateur.

Remarque : Les paramètres wh et wl sont mémorisés en période de résolution  $0,5 \mu\text{s}$  ( $2 * 10^6/\text{fréquence}$ ) et non pas sous la forme fréquence (pas/s). Une conversion Fréquence\_Période est donc assurée avant la mise à jour du paramètre.

La relecture de ces paramètres par la commande QL effectue les opérations inverses.

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.

**Remarque : Pour éviter plusieurs calculs successifs de la loi, il est recommandé de grouper sur la même ligne les commandes WH, WL, WT, WN.**

*Pendant, il est possible de ne modifier qu'un paramètre de la loi à la fois grâce aux syntaxes suivantes :*

`[@]WL_wl`  
`[@]WH_wh`  
`[@]WT_tr`                      ou `[@]WT_ta :td`  
`[@]WN_wn`

Exemple :

01WH 1350, WL 250, WT500, WN32 ;fixe pour l'axe 01 les paramètres de la loi suivants :

une résolution de  $32 \mu\text{pas/pas}$   
une vitesse de consigne à  $1350 \text{ pas/s}$ , soit  $43200 \mu\text{pas/s}$   
une vitesse de démarrage à  $250 \text{ pas/s}$ , soit  $8000 \mu\text{pas/s}$   
un temps de rampe de 500ms (en accélération ET en décélération)

WH 1000, WL 200, WT300:200, WN16 ;fixe pour l'ensemble des axes MICROSIMPA :

une résolution de  $16 \mu\text{pas/pas}$   
une vitesse de consigne à  $1000 \text{ pas/s}$ , soit  $16000 \mu\text{pas/s}$   
une vitesse de démarrage à  $200 \text{ pas/s}$ , soit  $3200 \mu\text{pas/s}$   
un temps de rampe de 300ms en accélération et de 200ms en décélération.

Code d'erreur positionné

Le module positionne les codes d'erreur suivants :

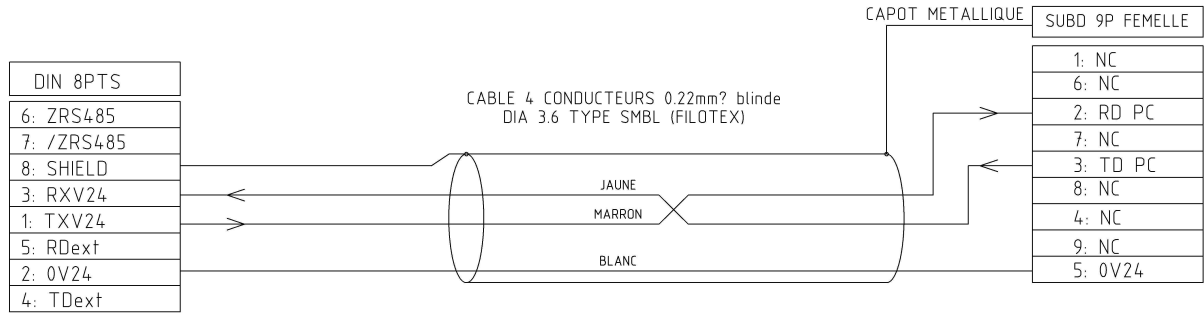
0 : La commande WH, WL, WT ou WN n'a pas de paramètre.  
Le paramètre fourni ne correspond pas à une valeur numérique.  
Le paramètre fourni est signé.

1 : La valeur donnée du paramètre est hors limite  
ou la cohérence entre les paramètres  $V_{\min}$ ,  $V_{\max}$ ,  $T_r$  et  $\mu$  n'est pas respectée. (cf. § IV.3.3.)

## VI – ANNEXES

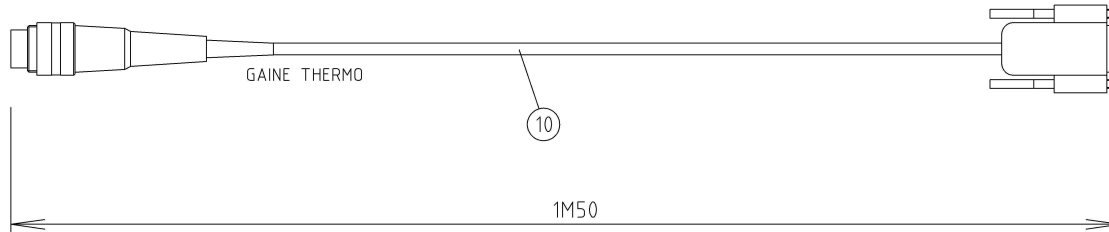
### VI.1 – Schémas câbles d'interconnexion

#### VI.1.1 – Câble liaison série PC RS232 en V24



1 CONNECTEUR DIN45326 8PTS DROIT  
ref: SV81 (LUMBERG)

1 CONNECTEUR SUBD 9P FEMELLE  
REF SOURIAU: 8657 09F064  
+ CAPOT METAL



#### VI.1.2 – Câble d'extension RS485

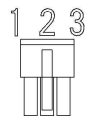
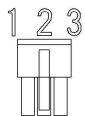
MODULE  
MICROSIMPA  
N1

MODULE  
MICROSIMPA  
N2

MOLEX 3PTS 43645-0300	
1:	0V24
2:	ZRS485
3:	/ZRS485

MOLEX 3PTS 43645-0300	
1:	0V24
2:	ZRS485
3:	/ZRS485

MODULES  
MICROSIMPA  
SUIVANTS

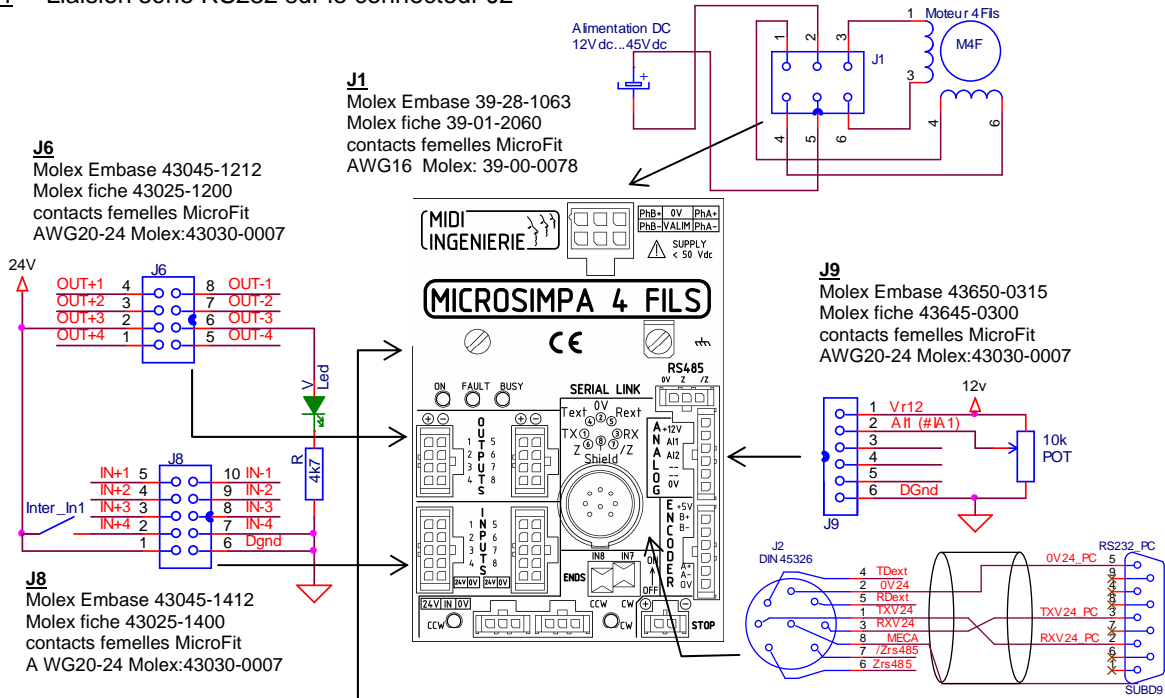


## VI.2 – Guide pratique d'utilisation de la carte MICROSIMPA 4 fils

Ce guide reprend les différentes étapes de mise en œuvre de la carte MICROSIMPA 4 Fils (MS41) associée à un moteur bipolaire.

1- **Réaliser le câblage** de la carte MICROSIMPA 4 Fils comme représenté ci après :

- 1 Alimentation DC 24V **hors tension** et Moteur 4 Fils au niveau du connecteur J1
- 2 une led de visualisation sur Out4 de J8 et un interrupteur sur In4 de J6
- 3 un potentiomètre 10K sur l'entrée analogique Vana (#IA1) de J9
- 4 Liaison série RS232 sur le connecteur J2



La carte étant hors tension, positionner les roues codeuses

sw1 sur 0 (adresse= 0)  
et sw2 sur 2 (Bd=38400, xon/Xoff)

2- **Mettre sous tension** la carte MICROSIMPA 4 Fils ainsi que le PC.

A la mise sous tension de la carte, les leds doivent être dans l'état suivant :

La led On <verte> doit être allumée.

Elle reste éteinte lorsque la carte n'est pas correctement alimentée.

Vérifier le câblage de l'alimentation et/ou le niveau de tension.

La led FAULT <rouge> s'allume 1s. à la mise sous tension puis s'éteint.

Elle reste allumée lorsqu'un défaut est détecté.

Ce défaut peut être de type	Alimentation	sous-tention(u) ou Sur-tension(U)
	Surcharge alimentation	24v/+12v/+5v (O:led clignote)
	Surcourant	Vérifier le câblage du moteur
	Courant de fuite moteur	Vérifier le câble du moteur
	Température.	

Elles clignotent en cas de sous tension(u) ou d'un défaut surcharge(O) sur 24v ou +5V.

Par ailleurs, FAULT s'allume 1 seconde à la mise sous tension et à la commande MR.

La led BUSY <jaune> doit être éteinte.

Elle s'allume lorsque le moteur tourne avec de la puissance.

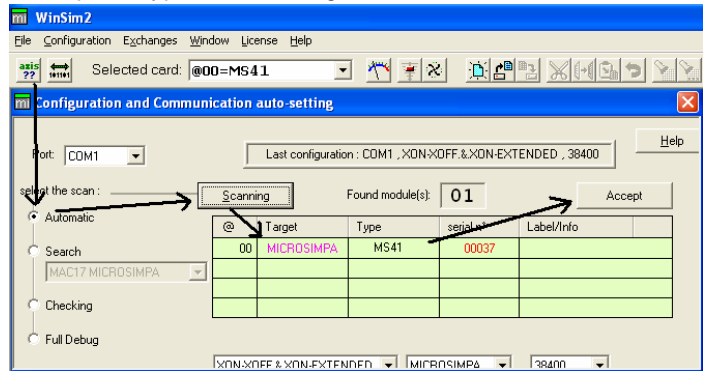
Elle clignote lorsque le moteur est à l'arrêt avec puissance

ou que le compteur de position s'incrémente sans puissance moteur

3- Lancer WinSim2 et Configurer la liaison série ainsi que le type de module grâce à l'écran ci-dessous

**Configuration de WS2**

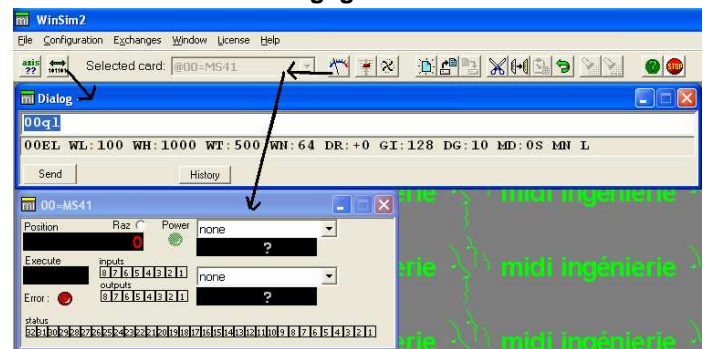
- 3-1 Ouvrir la configuration 'Axis'
  - 3-2 lancer en automatique le 'scanning'
  - 3-3 WS2 doit trouver l'axe 'US..'
  - 3-4 Accepter la configuration
- En cas de non reconnaissance, vérifier la connexion, le port série,...



4-Gestion de la carte MICROSIMPA 4 Fils et du moteur 4 Fils à l'aide du langage de base des modules.

**Mode commande de base**

- 4-1 lancer le 'suivi'
- 4-2 lancer le 'dialogue'
- 4-3 tester les commandes...



```

00MRZ ;reset module en mode 'sortie usine' (FAULT=On~1sec puis off)
00WL100,WH1000,WT500,WN64,G1128 ;programme les paramètres de l'axe0 de la carte MS41 :
;vitesse min. WL=100µpas/s. et max. WH=1000µpas/s.
;temps de rampe WT=500millis. résolution WN=64 µpas/pas,
;courant moteur GI=128 soit 1,5Aeff Attention: fonction moteur
;programme le mode 'standby' actif (leff/2 à l'arrêt du moteur)
;reset & init. origine +assure enregistrement des paramètres
;Vérifier les paramètres : WL:100 WH:1000 WT:500 WN:64
; DR:+0 GI:128 DG:10 MD:0S MN L
00GO +64000 ;moteur tourne de 1000 pas (BUSY=On puis Clig.)
00GH ;moteur revient à l'origine (BUSY=On puis Clig.)
00GF ;moteur tourne sens horaire (BUSY=On)
00GE ;moteur s'arrête en 0.5sec (BUSY 0.5s., puis Clig.)
00#CPA := 0 ;Initialise l'origine à cette position (Compteur=Home=0)
00GA +32000 ;Moteur va à la position abs. 32000upas (BUSY=On, Clig.)
00GF -500 ;moteur tourne sens inverse à 500 pas/sec (BUSY=On)
00GS ;moteur s'arrête brutalement (BUSY=Clig.)
00GR ;coupure puissance moteur (BUSY=off)
00GM ;force la puissance moteur (BUSY=Clig.)
00#OUT:=hBE ;Programme Out[8.5 4.1] à 1011110 1 :Iout=0 (non.actif)
00#OUT.3 := 0 ;force la sortie 3 à 0 (actif) 0 :Vout+=Vout- (actif)
00#01 := 129 ;force la variable '1' à 129
00QR #01 ;relit cette variable '1' et #OUT reponse: 00#1=#OUT
00QR #OUT B ;relit la variable '#OUT en binaire rep.: 00#OUT=B10111010
00 ... ;...tester les commandes
    
```

pour plus de précisions concernant les commandes directes ainsi que le mode séquence décrit ci-après, consulter le manuel utilisateur de la carte MICROSIMPA 4 Fils.

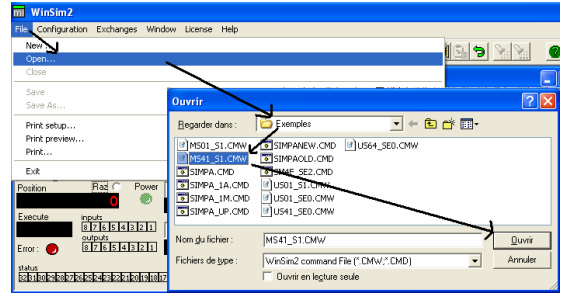
Adapter le courant au moteur utilisé si le courant nominal de ce dernier est inférieur à 1,5Aeff  
Nota : 1000 pas = 5 tours pour un moteur à 200 pas/tour

5- Mise en œuvre d'un **automatisme** grâce au mode séquence

Charger le fichier séquence  
'MS41\_S1.CMW'  
à l'aide de WinSim2 :

1-Ouvrir le fichier :

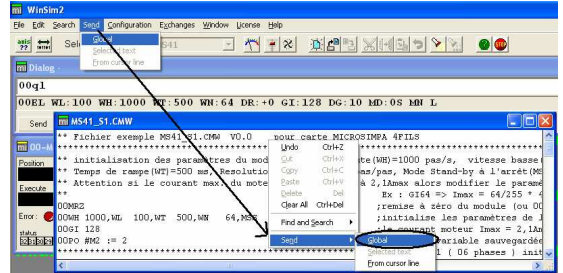
**File...Open...**  
selectionner dans  
.../Winsim2/Exemples  
'MS41\_S1.CMW'  
puis **Ouvrir**



2-puis le télécharger :

**send...global**

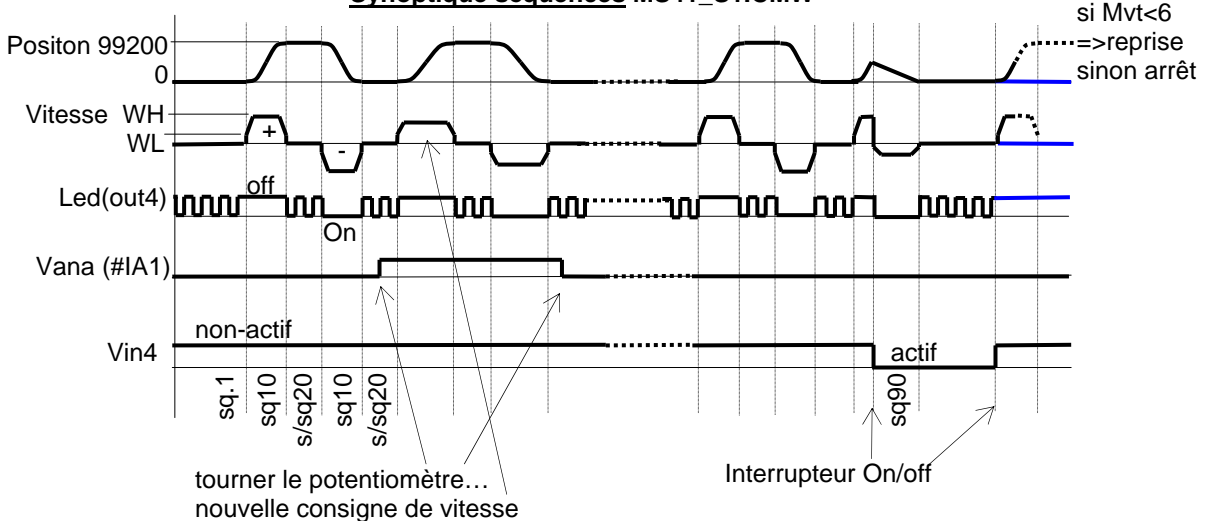
rem. :Pour que le fichier soit  
actif (bleu), cliquer dessus.



Avant d'exécuter le fichier séquence 'MS41\_S1.CMW', vérifier que le courant maximum admissible par le moteur est supérieur à 2,1Amax (1,5Aeff).  
Dans le cas contraire, modifier le paramètre G1128 avant de lancer les actions suivantes :

...Action	...Remarques
00SS01	;Lance la séquence 01 de début de cycle (voir synoptique ci-dessous)
00QR #1	;Affiche la variable #1 c.à.d. le compteur d'aller/retour en Sq.10
'Manœuvrer le potentiometre	;La vitesse moteur varie en fonction de Vana = 250, 500, 750,1000 pas/s
'Manœuvrer IN4 sur On'	;arrêt Mvt., Sq.90 <revient lentement en Home, puis led(out4) clignote...
'Manœuvrer IN4 sur off'	;... si MVT<6 alors retourne Sq.10 sinon En.=off&fin
00GS,SD01	;Arrête séquence et Mvt. en cours puis programme le mode séquence de
	;démarrage. La Sq.01 de démarrage est relancée automatiquement :
'Alimentation : off puis On'	; après une mise sous tension
00MR	;ou après une commande MR (RàZ, puis relance Sq.01)
00#M2 := .3	;Nombre de clignotement = 3
00GS,SR	;arrête séq./Mvt. en cours puis supprime le mode séquence au démarrage

**Synoptique séquences MS41\_S1.CMW**



**Remarque:** Toutes les lignes du listing commençant par \* sont des commentaires.  
Il est bien entendu possible de saisir toutes les commandes en manuel avec la fenêtre Dialogue

```

** Fichier exemple MS41_S1.CMW V0.0 pour carte MICROSIMPA 4FILS
*****
** initialisation des paramètres du module : Vitesse haute(WH)=1000 pas/s, vitesse basse(WL)= 100 pas/s,
** Temps de rampe(WT)=500 ms, Resolution(Wn) = 64 micropas/pas, Mode Stand-by à l'arrêt(MSS) et Ieff=1,5Aeff(Gil28)
** Attention si le courant max. du moteur est inférieur à 2,1Amax alors modifier le paramètre GI128
**
                                Ex : GI64 => Imax = 64/255 * 4,2Amax = 1,0Amax (=0,75Aeff)
00MRZ                                ;remise à zéro du module (ou 00SE0 efface toutes les séquences)
00WH 1000,WL 100,WT 500,WN 64,MSS    ;initialise les paramètres de la loi puis programme
00GI 128                                ;le courant moteur Imax = 2,1Amax = 128/255 * 4,2 (fonction Moteur)
00PO #M2 := 2                            ;programme variable sauvegardée #M2(Nbr. de clignotement) =2
***** Séquence 01 ( 07 phases ) init
00GS,SE01,SN01                            ;Début prog. de la séq.01 (GS,SE01:stop.Mvt & efface version precedente)
00SP 01 NO FF NU 10                            ;Out=non.actif, Puissance moteur off
00SP 02 #CPA := 0                            ;Remise à zéro compteur position (en micropas)
00SP 03 #1 := 4                                ;initialise la variable de calcul #1 à 4
00SP 04 NO 00:08 NW 1000                        ;force Out4(masque :08)=led actif. et attend 1s.
00SP 05 NO FF NW 500                            ;toutes les sorties sont non.actif. et attend 500ms
00SP 06 #1 := #1 -1 NS 07:07:04                ;décrémente #1 de (-)1, si #1 >0 =>retourne Phase 04
00SP 07 NW 1 NS254 NL 10                        ; sinon #1=<0 stop Sq.(NS254) puis Sq.10(NL10)
00SF                                ;enregistre la séquence courante 01
***** Séquence 10 ( 16 phases ) Mouvement moteur
00GS,SE10,SN10                            ;Début prog. de la séq.10 (GS,SE10:stop.Mvt & efface version precedente)
***** recherche consigne de vitesse =fonction de l'entrée analog.1
00SP 01 #AI1 ? 1000 NS 02:08:02                ;Si AI1 < 1000(1v) alors WH (ph08)
00SP 02 #AI1 ? 2500 NS 03:03:04                ;Si AI1 <ou= 2500(2v5)
00SP 03 NO FB NC 250 NS 10 ; => Out3=Act., NewConsigne=250p/s puis Ph.10
00SP 04 #AI1 ? 5000 NS 05:05:06                ;Si AI1 <ou= 5000(5v)
00SP 05 NO FD NC 500 NS 10 ; => Out2=Act., N.C.= 500pas/s puis ph.10
00SP 06 #AI1 ? 7500 NS 07:07:08                ;Si AI1 <ou= 7500(7v5)
00SP 07 NO FE NC 750 NS 10 ; => Out1=Act., N.C.= 750pas/s puis ph.10
00SP 08 NO FF NC 1000 NS 10 ; sinon Out.=na., N.C.=1000pas/s puis ph.10
*****Envoie d'un mouvement aller/retour de 1550pas avec test sur In4 pour arrêter le mvt. (NF 0 0 0 16)
** Rem.: Mouvement en 2sec car 0.5s{WT} à vit. croissante de 100{WL} à 1000{WH} soit 275pas = 17600 micropas
** 1sec à vit. constante de 1000{WH} pas/s. car 1550pas - 2x275pas =1000pas
** 1550pas x64(Wn)=99200µpas 0.5s{WT} à vit. dégressive de 1000{WH} à 100{WL} soit 275pas
00SP 10 #2 := #M2                            ;Initialise le Nbre. de clig. #2 = #M2
00SP 11 NO FF:F8 NP 99200 NF 0 0 0 16          ;Out8.5+4(led)=off, va en pos. +1550pas Dès que In4=Act. alors Ph16
00SP 12 NW 10 NQ 20                            ; puis appel s/seq.20 (#2 x clignot. )
00SP 13 NO 00:F8 NH NF 0 0 0 16                ;Out8.5+4(led)=On, retour pos.home(0) Dès que In4=Act. alors Ph16

```

```

00SP 14          NW    10          NQ 20          ;           puis appel s/seq.20 (#2 x clignot.)
00SP 15          #1 := #1 + 1      NS 01 ;Ajoute (+)1 au compteur d'aller/retour(#1) puis relance ph.01
00SP 16 NO FF    #32 = #CPA        NL 90 NS 254 ;Out.=off, #32=Cpt.position et =>Stop(NS254) + va en Sq.90(NL90)
00SF                                     ;enregistre la séquence courante 10
***** Séquence 20 ( 10 phases ) Clignotement de la led.(sous séquence)
** la led(Out4) clignote entre 1 et 4 fois en fonction de la variable d'appel #2 puis repositionne Out4 dans son état d'origine
** (variable #3=image de l'état d'origine de Out4) si ( #2 <1 ) ou ( #2 > 4 ) , le nombre de clignotement est par défaut #2 =2
00GS,SE20,SN20                                     ;Début prog. de la séq.20 (GS,SE20:stop.Mvt & efface version precedente)
00SP 01          #2 ? 0           NS 03:03:02       ;Test #2 :           si           #2 >           0
00SP 02          #2 ? 4           NS 05:05:03       ;           et si           #2 <ou= 4 => ok (ph05)
00SP 03          #2 := 2          NS 05             ;           sinon ph03 #2 =           2 (val. déf)
00SP 05          #3 := #OUT.4     NS 07:07:06       ;#3=état Out4, si #3=0(Out4=Act.)=>Ph07 sinon ph06(#3>0)
00SP 06 NO 00:08 NW 1000          ;Clignotement de la led : allume led(out4) pdt. 1000ms
00SP 07 NO FF:08 NW 1000          ;           éteint led(out4) pdt. 1s
00SP 08          #2 := #2 - 1     NS 09:09:06       ;décompte #2, si #2 =0 alors fin du clignotement(Ph09) sinon Ph06
00SP 09 NO FF:08 #3 ? 0          NS 10:10:254      ;Out4=off, si #3>0 => fin de la (sous)séq.20
00SP 10 NO 00:08 NW 1000          NS 254 ;           sinon(#3=0) Out4=On, attend 1s puis fin(ns254) sous/sq.20
00SF                                     ;enregistre la séquence courante 20
****rem. :le masque XX:08 permet de traiter Out4 sans modifier les autres sorties... out4: poids=2^(4-1)=08
***** Séquence 90 ( 07 phases ) Arrêt
00GS,SE90,SN90                                     ;Début prog. de la séq.90, GS,SE90:stop.Mvt & efface version precedente
00SP 01 NO F0    NC 200           ;Out1.4=On(led=out4) et Out8.5=off, force vitesse 200pas/s
00SP 02          NH              NL 10             ;retour pos. home
00SP 03          #2 := 1          ;force la variable #2 =1 (Nbre. de clignotement)
00SP 04 NO FF    NW 200          NE 0 0 0 -6     ;Out.=off, attend 0,2s puis si IN4 = n.a. => ph6
00SP 05          NW 10          NQ 20 NS 03 ;           sinon s/seq.20 avec 1 x clignot.
00SP 06 NO FF    #1 ? 7          NS 07:08:07       ;si #1(Nbr.de cycle) <7 alors stop(NS254) et retourne Sq.10(NL10)
00SP 07 NO FF    NU 100          NL 0  NS 254 ;sinon           puissance off et Stop (NL0)
00SP 08 NO FF    NU 100          NL 10 NS 254 ;           retour: stop(NS254) et retourne Sq.10(NL10)
00SF                                     ;enregistre la séquence courante 90
*****
00MR                                     ;00MR en fin de programmation assure la sauvegarde
;           et la réinitialisation de la carte

*Remarque: Toutes les lignes du listing commençant par * sont des commentaires.
***** Il est bien entendu possible de saisir toutes ces commandes en manuel avec la fenêtre Dialogue.

```



### **VI.3 – Eléments livrés avec la carte MICROSIMPA 4 fils**

- Un jeu de connecteurs spécifiques,
- Un cordon RS232C,
- Un manuel utilisateur,
- Un logiciel de démonstration pour PC Windows réf. WinSim2.

#### **Options possibles :**

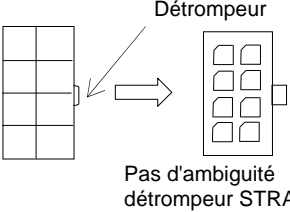
- Alimentation AC (80-220V) / DC 24V ou 42V / 320W réf. : SP320-24 ou SP320-42,
- La licence applications illimitées : WinSim2,
- La licence applications illimitées de la DLL Windows 98 à XP2 (compatible National Instrument),
- Les moteurs pas à pas 6 fils taille 17 et taille 23 (voir catalogue),
- Les capteurs de positions (effet Hall),
- Les logiciels applicatifs dédiés client selon cahier des charges.

## VI.4 – Récapitulatif des commandes MICROSIMPA

* [ @ ]MR	[Z]	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		
[ @ ]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 $65 \times 10^6 / V_{max}$ )		
[ @ ]WN	$\mu$	Résolution en micropas par pas (1, 2, 4, 8, 16, 32 ou 64)		
[ @ ]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	Courant moteur entre 0 et 2 Aeff (0-255) <i>(pas pour MICROSIMPA MS01)</i>		
* [ @ ]GL	Out:MSq type	Positionnement des sorties logiques différées (Out et MSq : en hexa)		
* [ @ ]GP	[p]:[d][M][P][A]	Gestion Sorties Logiques différées		
* [ @ ]GM		Puissance moteur (implicite avec GO, GA, GH et exécution séquence)		
[ @ ]GO	[±]/[n]	Exécution d'un mouvement relatif ( $\pm n$ micropas $< 2^{31}-1$ )		
* [ @ ]GR		Coupage puissance moteur		
* [ @ ]GS		Stop (arrêt immédiat d'un mouvement ou d'une séquence)		
* [ @ ]PO]	#n:=v	Affectation d'une valeur à une variable		
[ @ ]SD	ns	Sélection séquence de démarrage (exécutée sur RESET)		
[ @ ]SE	ns	Effacement séquence (ns = 0 : effacement de toutes les séquences)		
[ @ ]SF		Sauvegarde de la séquence numéro en cours d'édition		
[ @ ]SN	ns	Création séquence ns (1 à 200) de np phases (1 à 200)		
[ @ ]SP	np Na Cns [NL sc ou NQ ss] [Nex1x2...x8] [NS ps] [NO out : MSQ]	Définition de la phase np		
	Na : Nature			
	NA	accélération		
	NC	modification vitesse palier		
	ND	décélération		
	NH	retour origine (HOME)		
	NP	mouvement relatif		
	NT	puissance moteur ON	NW	attente
	NU	puissance moteur OFF	NX	mouvement absolu
	NV	vitesse constante	[PO]	Calcul et test sur variables
	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		
* [ @ ]QC		Etat sorties logiques déportées (GL/GP)		
* [ @ ]QD		Suivi des séquences et mouvements		
* [ @ ]QL		Lecture paramètres locaux		
* [ @ ]QR	#N[H][B]	Relecture variable		
* [ @ ]QS	ns np	Lecture de la phase np de la séquence ns		
* [ @ ]QV		Lecture version et indice du logiciel		
* [ @ ]QX		Lecture code état module		
[ @ ]	adresse du module			
*	commande utilisable quel que soit l'état du module			
[ @ ]	commande multimodule			
[Cmde]	commande optionnelle			
#1..#32	variable utilisateur - ( $2^{31} - 1$ ) $< \dots < (2^{31} - 1)$	#M1..#M32	variable utilisateur mémorisées - ( $2^{31} - 1$ ) $< \dots < (2^{31} - 1)$	
#IN	entrées logiques	#OUT	sorties logiques	
#CPA	position absolue - ( $2^{31} - 1$ ) $< \dots < (2^{31} - 1)$			
#VSUPPLY	valeur tension alimentation (mv)	#TEMP	température module (°C)	
#CODEUR	position CODEUR - ( $2^{31} - 1$ ) $< \dots < (2^{31} - 1)$		<i>pas pour Microsimpa 4 axes 6 fils (MS64)</i>	
#AO1, #AO2	sorties analogiques (mv)		<i>uniquement pour Microsimpa (MS01)</i>	
#AI1, #AI2	entrées analogiques (mv)		<i>#AI2 pas pour Microsimpa 4 axes 6 fils (MS64)</i>	
#VAUXP	valeur tension auxiliaire (mv)		<i>uniquement pour Microsimpa 4 axes 6 fils (MS64)</i>	

## FICHE DE MODIFICATION DOCS MI

Documentation concernée : Manuel d'utilisation du module MICROSIMPA 4 fils      réf. : BLN1570830.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 20/07/05	<b>Création</b>	Nom : B.LOPEZ Date : 20/07/05 Oui <input checked="" type="checkbox"/> Non <input type="checkbox"/> motif du refus :	Personne chargée de la réalisation : N.ROUMEGOUX Date réalisation : 20/07/05	0
M.NICOLAS 03/2006	<b>Corrective (éviter une mauvaise interprétation du schéma)</b> Page 8, 9 et 10 Sur <u>toutes les représentations</u> des connecteurs MOLEX MICROFIT  <div style="text-align: center;">  <p style="font-size: small;">Détrompeur</p> <p style="font-size: small;">Pas d'ambiguïté détrompeur STRAP</p> </div>	Nom : B.LOPEZ Date : 11/04/06 Oui <input checked="" type="checkbox"/> Non <input type="checkbox"/>	Personne chargée de la réalisation : N.ROUMEGOUX Date réalisation : 11/04/06	1
B.LOPEZ 03/2006	<b>Evolution</b> Passage en Soft MICROSIMPA			
B.LOPEZ 06/2007	Mise à jour des évolutions logiciel : gestion des variables, obsolescence des commandes, option extension.	Nom : B.LOPEZ Date : Oui <input checked="" type="checkbox"/> Non <input type="checkbox"/> motif du refus :	Personne chargée de la réalisation : N.ROUMEGOUX Date réalisation : 12/06/07	2
P.BERNADOU  04/2008	Corrections diverses	Nom : B.LOPEZ  Date : 03.04.08 Oui <input checked="" type="checkbox"/> Non <input type="checkbox"/> motif du refus :	Personne chargée de la réalisation : N.Roumégoux  Date réalisation : 04/2008	3