midi ingénierie
NEXEYA Products Division

3509, route de Baziège
31670 LABÈGE
France
Tél. : 33 (0)5 61 39 96 18
Fax : 33 (0)5 61 39 17 58
www.midi-ingenierie.com

midi ingénierie

DMAC & BMAC
Manuel Utilisateur

CANopen

Date : 23.11.12

Référence : dmaccanopen_v2_um_fr.pdf
Réf. MI : NSN1480884.DOC

Révision : 2

Auteur : N.SERRES
http://www.midi-ingenierie.com

afaq ISO 9001 Qualité AFNOR CERTIFICATION

S.A.S au capital de 713 046 €
RCS Toulouse 333 373 108
Siret 333 373 108 00013
APE 7112B
TVA FR 60333 373 108

SOMMAIRE

1. CAN et CANopen 4
1.1. Bus CAN (control area network) 4
1.2. Norme CANopen 4
1.3. Communiquer avec un équipement CANopen 5
1.3.1. Objets 5
1.3.2. Dictionnaire d'objets 5
1.3.3. Objets de communication 6
1.4. SDO (Service Data Object) 6
1.4.1. Dialogue SDO 7
1.4.2. Description des messages SDO 8
1.5. PDO (Process Data Object) 10
1.5.1. RPDO (Receive PDO) 10
1.5.2. TPDO (Transmit PDO) 12
1.6. EMERGENCY 14
1.7. NMT (Network management) 15
1.8. Heartbeat 16
2. Configuration du module 16
2.1. Configuration de l'adresse du module (NodeID) 17
2.2. Configuration du Bitrate 17
2.3. Exemple de séquence de Configuration 18
3. Fonctionnalités Motion Control 18
3.1. Machine d'état 18
3.2. Modes de mouvements 19
3.3. Mode profile_position 20
3.4. Mode profile_velocity 21
3.5. Mode Homing 21
4. Dictionnaire des objets 22
4.1. Objets de communication (DS301) 22
4.1.1. 0x1000 device_type 22
4.1.2. 0x1001 error_register 22
4.1.3. 0x1002 manufacturer_status_register 23
4.1.4. 0x1008 manufacturer_device_name 23
4.1.5. 0x1009 manufacturer_hardware_version 23
4.1.6. 0x100A manufacturer_software_version 24
4.1.7. 0x1010 STORE Parameters 24
4.1.8. 0x1011 RESTORE Default Parameters 24
4.1.9. 0x1014 COB-ID_emergency_message 25
4.1.10. 0x1017 producer_heartbeat_time 25
4.1.11. 0x1018 identity_object 25
4.1.12. 0x1400…1403 RPD01…4_Communication_parameters 26
4.1.13. 0x1600…1603 RPDO1…4_mapping_parameters 26
4.1.14. 0x1800…1803 TPDO1…4_communication_parameters 26
4.1.15. 0x1A00…1A03 TPDO1…4_mapping_parameters 26
4.2. Objets spécifiques manufacturer 26
4.2.1. 0x2000 volatile_user_variables 26
4.2.2. 0x2001 non-volatile_user_variables 26
4.2.3. 0x2002 user_timers 27
4.2.4. 0x2003 position_capture 27
4.2.5. 0x2004 temperature_object 27
4.2.6. 0x2005 analog_inputs 28
4.2.7. 0x2006 analog_outputs (DMAC34) 28
4.2.8. 0x2008 current_control 28
0x2009 close_loop_config (BMAC) 29
4.2.9. 0x207D soft_ends_config 29
4.2.10. 0x20FD digital_inputs_config 29
4.2.11. 0x20FE digital_ouputs_config 30
4.2.12. 0x2083 profile_acceleration_time 30
4.2.13. 0x2084 profile_deceleration_time 30
4.3. Objets motion control (DSP402) 31

4.3.1. 0x603F error_code 31
4.3.2. 0x6040 controlword 31
4.3.3. 0x6041 statusword 31
4.3.4. 0x605A quickstop_option_code 32
4.3.5. 0x6060 modes_of_operation 32
4.3.6. 0x6061 modes_of_operation_display 32
4.3.7. 0x6063 position_internal_value (BMAC) 32
4.3.8. 0x6064 position_actual_value 32
4.3.9. 0x6069 velocity_sensor_actual_value 33
4.3.10. 0x606A sensor_selection_code 33
4.3.11. 0x606C velocity_actual_value 33
4.3.12. 0x6072 max_torque 33
4.3.13. 0x6073 max_current (BMAC) 33
4.3.14. 0x6075 motor_rated_current (BMAC) 33
4.3.15. 0x6076 motor_rated_torque 33
4.3.16. 0x6079 DC_link_circuit_voltage 33
4.3.17. 0x607A target_position 34
4.3.18. 0x607C home_offset 34
4.3.19. 0x607D software_position_limit 34
0x6081 profile_velocity 35
4.3.20. 0x6082 end_velocity 35
4.3.21. 0x6083 profile_acceleration 35
4.3.22. 0x6084 profile_deceleration 35
4.3.23. 0x6086 motion_profile_type 35
4.3.24. 0x6098 homing_method 35
4.3.25. 0x6099 homing_speeds 36
4.3.26. 0x60FD digital_input 36
4.3.27. 0x60FE digital_output 36
4.3.28. 0x60FF target_velocity 37
4.3.29. 0x6404 motor_manufacturer 37
4.3.30. 0x6502 supported_drive_modes 37
4.3.31. 0x6504 drive_manufacturer 37
4.3.32. 0x6510 drive_data 37
5. Résumé des objets du dictionnaire 38

1. CAN et CANopen

1.1. Bus CAN (control area network)

Le nom CAN correspond à la couche physique et à la couche de donnée de communication entre deux équipements. Il s'agit d'un bus de communication série conçu à l'origine par Bosh Gmbh pour coordonner les échanges d'informations dans l'automobile.

Dans un réseau CAN, chaque équipement (ou nœud) peut envoyer des messages sur le bus. Un nœud reçoit tous les messages émis et utilise un système de filtre pour ne retenir que ceux qui le concernent.

La couche physique est constituée d'une paire différentielle terminée par une impédance de 124ohms à chaque extrémité. Le débit maximal supporté par le CAN est de 1Mbit/s sur des distances allant jusqu'à 40 mètres. Les systèmes nécessitant des distances supérieures devront utiliser des bitrates plus faibles.

Les données sont transmises sur le bus sous forme de paquets appelés messages ou "communication objects" (COB). Chaque paquet est constitué d'un identifiant (CAN message ID ou COB-ID), de bits de contrôle et de zéro à huit octets de donnée.

Chaque message est envoyé avec un contrôle d'erreur de type CRC (cyclic redundancy check) pour permettre aux nœuds de vérifier la cohérence des messages émis et reçus. Les messages ayant un contrôle d'erreur négatif sont réémis.

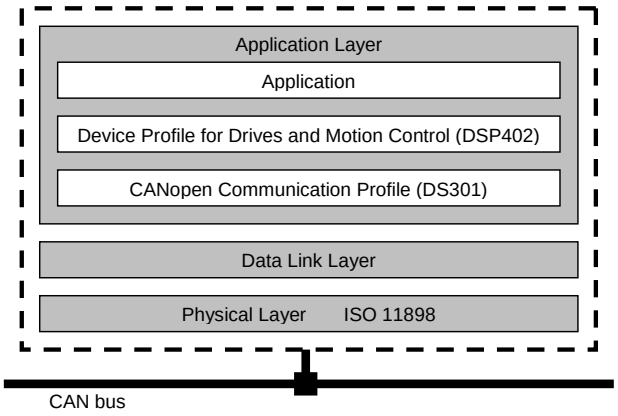
L'identifiant de message (COB-ID) joue deux rôles importants :

C'est grâce à lui que chaque nœud détermine quel message doit être accepté ou ignoré.

Il détermine également la priorité du message. En cas d'émission simultanée de deux messages, c'est celui qui a le COB-ID le plus faible qui est prioritaire. L'autre message est alors mémorisé par l'émetteur et réémis lorsque le bus est libre.

1.2. Norme CANopen

La norme CANopen a été développée par le groupe CANopen In Automation (CiA). Elle définit la manière dont les équipements utilisent le bus CAN pour communiquer. En fonction du type d'équipement dont il s'agit, le CANopen définit des "profils" qui permettent d'utiliser le bus de manière standard, robuste et efficace.



Le DMAC est compatible avec les profils DS301 (Application Layer and Communication Profile) et DSP402 (Device Profile for Drives and Motion Control).

Le **profil de communication DS301** forme l'interface entre les profils spécifiques aux dispositifs et le bus CAN. Sa spécification a été créée en 1995 et définit des standards uniformes pour l'échange de données entre des types de dispositifs différents sous CANopen.

Les objets du profil de communication se chargent des tâches d'échange de données et de paramètres avec les autres nœuds et initialisent, commandent et surveillent le dispositif dans le réseau.

Le **profil spécifique dispositif DSP402** (Device profile for drives and motion control) décrit les objets standardisés pour le positionnement, la surveillance et le réglage d'entraînements.

Les tâches des objets sont :

- Contrôle du dispositif et surveillance de son état (Device Control)
- Paramétrage standardisé
- Changement, contrôle et exécution de modes d'exploitation

1.3. Communiquer avec un équipement CANopen

1.3.1. Objets

Toutes les tâches sous CANopen sont exécutées à l'aide d'objets. Les objets exécutent le transport de données vers le bus de terrain, commandent l'établissement des liaisons ou surveillent les nœuds du réseau. Les objets en lien direct avec le dispositif (objets spécifiques au dispositif) permettent d'exploiter ou de modifier les fonctions de celui-ci.

1.3.2. Dictionnaire d'objets

Chaque équipement définit un certain nombre d'objets qui correspondent aux paramètres de configuration, de contrôle et d'état de l'équipement. L'ensemble de ces objets constitue le dictionnaire d'objet qui peut être vu comme une table d'indirection entre les variables internes de l'équipement et le bus CAN.

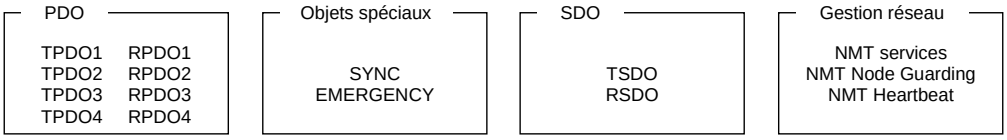
Dans le dictionnaire, chaque objet est repéré par un index (codé sur 2 octets, exprimé en hexadécimal). Lorsque plusieurs objets appartiennent à une même catégorie, ils ont le même index et sont repérés par un sous index (codé sur 1 octet).

Le dictionnaire d'objets est organisé de la façon suivante :

Index Range	Objects
0000	not used
0001-001F	Static Data Types
0020-003F	Complex Data Types
0040-005F	Manufacturer Specific Complex Data Types
0060-007F	Device Profile Specific Static Data Types (including those specific to motion control)
0080-009F	Device Profile Specific Complex Data Types (including those specific to motion control)
00A0-0FFF	Reserved for future use
1000-1FFF	Communication Profile Area (DS 301)
2000-5FFF	Manufacturer Specific Profile Area
6000-9FFF	Standardized Device Profile Area (including Profile for Motion Control) (DSP402)
A000-FFFF	Reserved for further use

1.3.3. Objets de communication

Les objets de communication sont standardisés avec le profil de communication CANopen DS301. En fonction de leurs tâches, les objets peuvent être divisés en 4 groupes :



• Objets de données de processus PDO (process data object) pour la transmission en temps réel des données de processus

• Objets de données de service SDO (service data object) pour l'accès en écriture et en lecture à la liste d'objets

• Objets pour la commande des messages CAN :
– Objet SYNC (synchronisation object) pour la synchronisation des noeuds du réseau
– Objet EMCY (emergency object) pour l'affichage d'erreur d'un appareil ou de sa périphérie.

• Services de gestion du réseau :
– NMT Services pour l'initialisation et la gestion du réseau (NMT : network management)
– NMT Node Guarding pour la surveillance des noeuds du réseau
– NMT Heartbeat pour la surveillance des noeuds du réseau

1.4. SDO (Service Data Object)

Les messages de type SDO (Service Data Object) permettent d'accéder en lecture et en écriture aux objets du dictionnaire.

La communication par SDO est de type Client/Server. Dans ce cas, le Maître CANopen (PC ou automate) est appelé le client, et le module DMAC est appelé serveur.

Conformément à la norme CANopen DS301, les messages SDO ont pour ID les valeurs suivantes:

(0x600 + Node_ID) pour les messages issus du Client vers le Serveur.

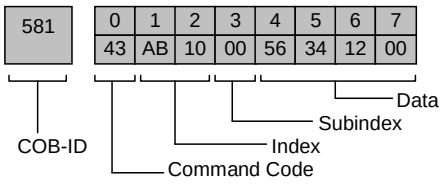
(0x580 + Node_ID) pour les messages issus du Serveur vers le Client.

Pour écrire des valeurs dans les objets du dictionnaire, on utilise des SDO de type "Download".

Pour relire les valeurs des objets du dictionnaire, on utilise des SDO de type "Upload".

Un message SDO est composé de manière simplifiée du COB-Id et du cadre-données SDO dans lequel jusqu'à 4 octets de données peuvent être transmis. Les suites de données plus longues sont divisées en plusieurs messages SDO via un protocole spécial appelé "segmented".

La figure suivante montre un exemple de message SDO:



Les index et les données sont transmis avec l'octet de poids faible (LSB) aligné à gauche (format Intel).
Si le SDO comprend des valeurs numériques d'une longueur de plus d'un octet, les données doivent être converties par octet avant et après une transmission.
Dans l'exemple précédent, l'index vaut donc 0x10AB et les datas 0x00123456

1.4.1. Dialogue SDO

Le client commence toujours par initialiser le SDO en envoyant un message "Initiate Request" puis envoie un certain nombre de messages "Segment Request".

Si 4 octets ou moins doivent être transférés, on peut utiliser un message "Expedited SDO" dans lequel les données sont entièrement contenues dans le message "Initiate SDO".

Chaque message "Request" émis par le Client est confirmé par le Serveur au moyen d'un message "Response".

Les échanges peuvent donc prendre une des quatre formes suivantes:

	Expedited	Segmented
Download (écriture)	>> Initiate Download Request >> << Initiate Download Response <<	>> Initiate Download Request >> << Initiate Download Response << >> Download Segment Request >> << Download Segment Response << … >> Download Segment Request >> << Download Segment Response <<
Upload (lecture)	>> Initiate Upload Request >> << Initiate Upload Response <<	>> Initiate Upload Request >> << Initiate Upload Response << >> Upload Segment Request >> << Upload Segment Response << … >> Upload Segment Request >> << Upload Segment Response <<

1.4.2. Description des messages SDO

Initiate Download Request:

Octet	0					1	2	3	4	5	6	7
Bits	7..5 ccs=001	4 x	3..2 n	1 e	0 s	ix		sx	d			

Initiate Download Response:

Octet	0				1	2	3	4	5	6	7
Bits	7..5 scs=011	4..0 x			ix		sx	reserved			

Download Segment Request:

Octet	0				1	2	3	4	5	6	7
Bits	7..5 ccs=000	4 t	3..1 n	0 c	seg-data						

Download Segment Response:

Octet	0			1	2	3	4	5	6	7
Bits	7..5 scs=001	4 t	3..0 x	reserved						

Initiate Upload Request:

Octet	0		1	2	3	4	5	6	7
Bits	7..5 ccs=010	4..0 x	ix		sx	reserved			

Initiate Upload Response:

Octet	0					1	2	3	4	5	6	7
Bits	7..5 scs=010	4 x	3..2 n	1 e	0 s	ix		sx	d			

Upload Segment Request:

Octet	0			1	2	3	4	5	6	7
Bits	7..5 ccs=011	4 t	3..0 x	reserved						

Upload Segment Response:

Octet	0			1	2	3	4	5	6	7
Bits	7..5 scs=000	4 t	3..0 x	seg-data						

ccs: client command specifier
- 001: initiate download request
- 000: download segment request
- 010: initiate upload request
- 011: upload segment request
- 011: upload segment request
- 100 : abort SDO

scs: server command specifier
- 011: initiate download response
- 001: download segment response
- 010: initiate upload response
- 000: upload segment response
- 100 : abort SDO

n: le champ n indique le nombre d'octets de d qui ne contiennent pas de données (uniquement valable si e = 1 et s = 1).

e: type de transfert.
e=0: normal.
e=1: expedited

s: indicateur de taille.
s=0: la taille n'est pas spécifiée.
s=1: la taille est spécifiée.

ix: index de l'objet adressé.

sx: sous-index de l'objet adressé.

c: segments restants
c=0: dernier segment à être transféré.
c=1: il reste un (ou des) segment(s) à transférer.

t: toggle bit. Ce bit est inversé à chaque segment téléchargé. Le premier segment a un toggle **bit à 0.** Le toggle bit doit être identique pour une requête et sa confirmation.

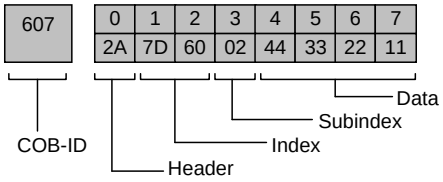
seg-data: au plus 7 octets des données transmises.

d: data
si e=0 et s=0: d est réservé pour un usage futur.
si e=0 et s=1: d contient le nombre d'octets qui vont être transférés
si e=1 et s=1: d contient les données transférées (4-n octets).
si e=1 et s=0: d contient un nombre non spécifié de données transférées.

x: non utilisé (toujours 0)

reserved: réservé pour un usage futur (toujours 0)

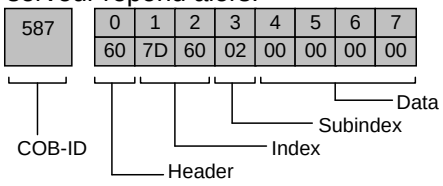
Exemple: Le client écrit la valeur 0x11223344 dans l'objet max_position_limit (index 0x607D, sous-index 2) en utilisant un "Expedited SDO" adressé au module d'adresse 7.



détail du header:

header:	ccs			x	n		e	s
0x2A	0	0	1	0	1	0	1	0

Le serveur répond alors:



détail du header

header:	scs			x	n		e	s
0x60	0	1	1	0	0	0	0	0

Les messages Abort SDO sont ainsi :

581	80 index s/index code Abort
CODID	580h + Nod-ID
Index	2 octets d'index du SDO en cours
s/index	1 octet s/index du SDO en cours
code Abort	4 octets donnant le type d'erreur lors du transfert SDO (voir DS301)

1.5. PDO (Process Data Object)

Les messages PDO permettent d'accéder à un certain nombre d'objets de manière prioritaire et rapide.

Chaque message PDO est pré-configuré avec les index et sub-index de un ou plusieurs objets. Il est ainsi possible de relire ou d'écrire plusieurs objets simultanément avec un seul message CAN.

Le DMAC dispose de quatre RPDO (Receive PDO) pour écrire les objets et de quatre TPDO (Transmit PDO) pour relire les objets.

1.5.1. RPDO (Receive PDO)

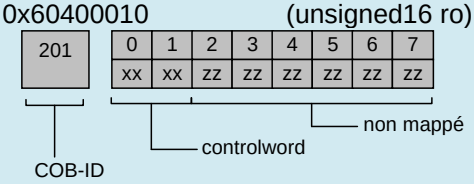
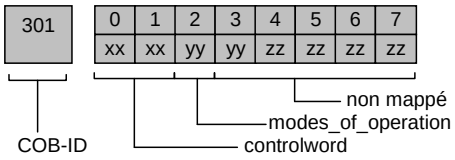
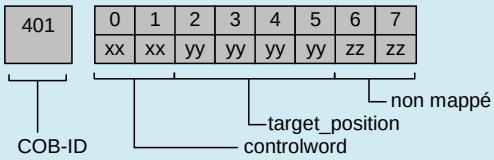
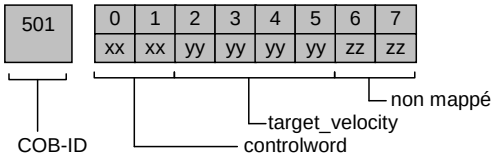
Les COB-ID des RPDOs sont fixes. Ils ne peuvent pas être modifiés.

RPDO Communication parameters "read only"

	index	sous-index	valeur	
RPDO1	0x1400	0x00: number_of_entries 0x01: COB-ID 0x02: Transmission type	2 0x200 + node_ID 255 (asynchrone)	(unsigned8 ro) (unsigned32 ro) (unsigned8 ro)
RPDO2	0x1401	0x00: number_of_entries 0x01: COB-ID 0x02: Transmission type	2 0x300 + node_ID 255 (asynchrone)	(unsigned8 ro) (unsigned32 ro) (unsigned8 ro)
RPDO3	0x1402	0x00: number_of_entries 0x01: COB-ID 0x02: Transmission type	2 0x400 + node_ID 255 (asynchrone)	(unsigned8 ro) (unsigned32 ro) (unsigned8 ro)
RPDO4	0x1403	0x00: number_of_entries 0x01: COB-ID 0x02: Transmission type	2 0x500 + node_ID 255 (asynchrone)	(unsigned8 ro) (unsigned32 ro) (unsigned8 ro)

Le mapping des RPDOs est fixe. Il ne peut pas être modifié:

RPDO Mapping parameters "read only"

	index	Sous-index	valeur
RPDO1	0x1600	0x00: number_of_entries 0x01: controlword	0x01 (unsigned8 ro) 0x60400010 (unsigned16 ro) 
RPDO2	0x1601	0x00: number_of_entries 0x01: controlword 0x02 : modes_of_operations	0x02 (unsigned8 ro) 0x60400010 (unsigned16 ro) 0x60600008 (unsigned8 ro) 
RPDO3	0x1602	0x00: number_of_entries 0x01: controlword 0x02 : target_position	0x02 (unsigned8 ro) 0x60400010 (unsigned16 ro) 0x607A0020 (unsigned32 ro) 
RPDO4	0x1603	0x00: number_of_entries 0x01: controlword 0x02 : target_velocity	0x02 (unsigned8 ro) 0x60400010 (unsigned16 ro) 0x60FF0020 (unsigned32 ro) 

1.5.2. TPDO (Transmit PDO)

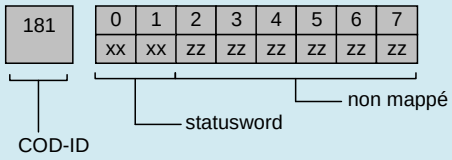
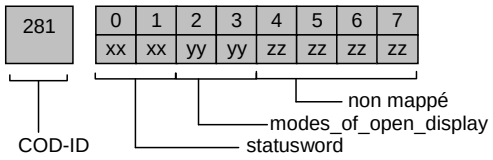
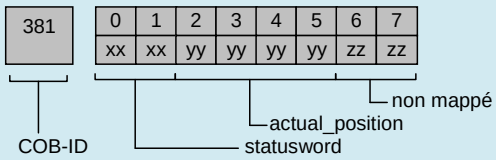
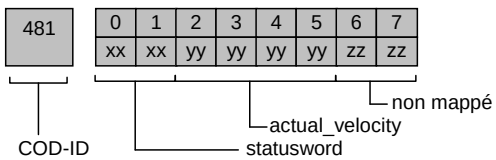
Les COB-ID des TPDOs sont fixes, ils ne peuvent pas être modifiés.

TPDO Communication parameters "read only"

	index	Sous-index	valeur	
TPDO1	0x1800	0x00: number_of_entries 0x01: COB-ID 0x02 : transmit type 0x03 : inhibit time 0x04 : compatibility Entry 0x05 : Event Timer	5 0x180 + mode_ID 254 (event timer ou RTR) 10 0	(unsigned8 ro) (unsigned32 ro) (unsigned8 ro) (unsigned16 rw) (unsigned8 rw) (unsigned16 rw)
TPDO2	0x1801	0x00: number_of_entries 0x01: COB-ID 0x02 : transmit type 0x03 : inhibit time 0x04 : compatibility Entry 0x05 : Event Timer	5 0x280 + mode_ID 254 (event timer ou RTR) 10 0	((unsigned8 ro) (unsigned32 ro) (unsigned8 ro) (unsigned16 rw) (unsigned8 rw) (unsigned16 rw)
TPDO3	0x1802	0x00: number_of_entries 0x01: COB-ID 0x02 : transmit type 0x03 : inhibit time 0x04 : compatibility Entry 0x05 : Event Timer	5 0x380 + mode_ID 254 (event timer ou RTR) 10 0	(unsigned8 ro) (unsigned32 ro) (unsigned8 ro) (unsigned16 rw) (unsigned8 rw) (unsigned16 rw)
TPDO4	0x1803	0x00: number_of_entries 0x01: COB-ID 0x02 : transmit type 0x03 : inhibit time 0x04 : compatibility Entry 0x05 : Event Timer	5 0x480 + mode_ID 254 (event timer ou RTR) 10 0	(unsigned8 ro) (unsigned32 ro) (unsigned8 ro) (unsigned16 rw) (unsigned8 rw) (unsigned16 rw)

Le Mapping des TPDOs est fixe, il ne peut pas être modifié.

TPDO Mapping parameters "read only"

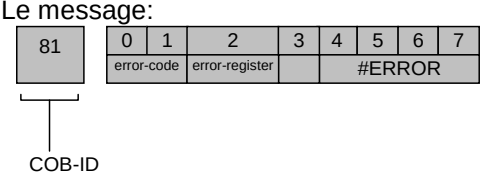
	index	Sous-index	valeur
TPDO1	0x1A00	0x00: number_of_entries 0x01: statusword	0x01 (unsigned8 ro) 0x60410010 (unsigned16 ro) 
TPDO2	0x1A01	0x00: number_of_entries 0x01: statusword 0x02 : modes_of_open_display	0x02 (unsigned8 ro) 0x60410010 (unsigned16 ro) 0x60610008 (unsigned8 ro) 
TPDO3	0x1A02	0x00: number_of_entries 0x01: statusword 0x02 : actual_position	0x02 (unsigned8 ro) 0x60410010 (unsigned16 ro) 0x60640020 (unsigned32 ro) 
TPDO4	0x1A03	0x00: number_of_entries 0x01: statusword 0x02 : actual_velocity	0x02 (unsigned8 ro) 0x60410010 (unsigned16 ro) 0x606C0020 (unsigned32 ro) 

1.6. EMERGENCY

Les messages de type emergency sont délivrés par le module de manière spontanée au gré des erreurs détectées.

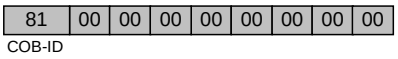
Conformément à la norme DS301, le COB-ID prend la valeur : 0x80 + Nod-ID

Le message:



Les deux 1ers octets	Contiennent l'error-code tel qu'on retrouve sa dernière valeur dans l'objet 0x603F
L'octet 2	Contient l'error-register tel qu'on le retrouve dans l'objet 0x1001
L'octet 3	est non significatif
Les quatre derniers octets	donnent le contenu de la variable # ERROR du DMAC.

Lorsque l'ensemble des erreurs survenues est résolu, un message contenant des octets à 0 est émis :



1.7. NMT (Network management)

Les commandes NMT sont utilisées pour contrôler l'état de communication de l'axe.
Seuls les services NMT obligatoires sont implémentés dans les DMACs.

L'axe peut être dans l'un des états suivants:

Etat		
Initialization	0	L'axe n'est pas prêt ou est en train de s'initialiser. Toute communication sur le bus CAN est ignorée.
Pre-Operational	127	L'axe a terminé sa procédure d'initialisation, mais n'a pas encore reçu de commande pour passer en mode Operational. Les messages de type SDO et NMT sont traités, mais pas les PDO.
Operational	5	L'axe est complètement opérationnel. Les messages SDO, NMT, ainsi que les PDO sont traités.
Stopped	4	La communication de l'axe est stoppée. Il ne peut répondre qu'aux messages NMT.

Au reset ou à la mise sous tension, le DMAC passe automatiquement en mode **Initialization** et effectue sa procédure d'initialisation. Lorsque celle-ci est terminée, il passe en mode **Pre-Operational**. Le passage dans les modes **Operational** et **Stopped** se fait au moyen des messages NMT.

L'identifiant COB-ID d'un message NMT est toujours 0 (zéro).

Un message NMT est constitué de deux octets. Le premier spécifie la commande envoyée et le second contient l'identifiant du module à qui s'adresse ce message. Dans le cas où ce deuxième octet est nul, tous les modules connectés effectuent la commande.

Les commandes suivantes sont supportées:

Identifiant de commande (hex)	Action
1	Enter operational
2	Enter stopped
128 (0x80)	Enter Pre-operational
129 (0x81)	Reset node (effectue un reset de l'axe)
130 (0x82)	Reset communication

Exemple:
Pour demander au module de Node-ID=7 d'effectuer un reset, il faut lui envoyer le message NMT:

000	0	1	2	3	4	5	6	7
	81	07	-	-	-	-	-	-

COB-ID — Reset node — node-ID

1.8. Heartbeat

Le protocole Heartbeat Producer est implémenté dans les modules DMAC.

Un message de type Heartbeat est donc émis périodiquement et au reset (Boot-up) par les modules pour permettre au Maître CAN (voire aux autres modules du bus) d'être informés du fonctionnement de l'axe et de réagir en conséquence.

L'objet producer_heartbeat_time (0x1017) permet de régler la période des messages Heartbeat. Si cet objet a une valeur nulle, la fonctionnalité est inhibée (valeur à défaut 0).

Les messages Heartbeat ont un identifiant COB-ID fixé qui vaut 0x700 + Node-ID. Par exemple, le module d'adresse 4 aura un COB-ID de 0x704 pour ses messages Heartbeat.

Les messages Heartbeat sont constitués d'un seul octet qui reflète l'état du module :

Heartbeat	Description
0	Boot-up
4	Stopped
5	Operational
127 (0x7F)	Pre-Operational

2. Configuration du module

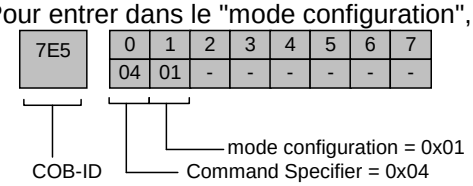
Pour configurer l'adresse et le bitrate d'un module, on utilise le protocole LMT décrit par la norme CANopen DS205. Un seul module sur la ligne.

Chaque message est composé d'un octet de commande (appelé Command Specifier) et d'un ou deux octets dont la signification varie en fonction de la commande.

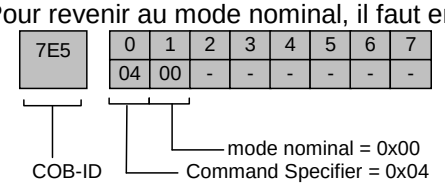
Les messages LMT émis du maître vers l'esclave ont pour COB-ID la valeur 0x7E5.

Les messages LMT émis de l'esclave vers le maître ont pour COB-ID la valeur 0x7E4.

Pour entrer dans le "mode configuration", il faut envoyer au module le message suivant:



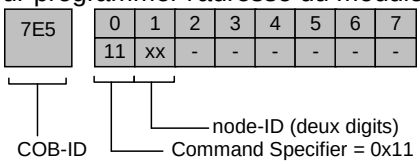
Pour revenir au mode nominal, il faut envoyer au module le message suivant:



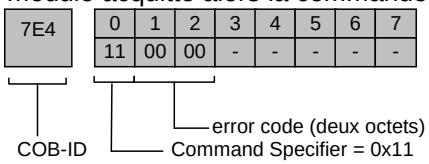
2.1. Configuration de l'adresse du module (NodeID)

Chaque module est repéré sur le bus par son adresse (NodeID).
Les adresses valides sont comprises entre 0x01 et 0x7F.

Pour programmer l'adresse du module, il faut envoyer le message suivant:



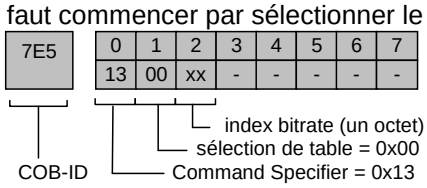
Le module acquitte alors la commande en renvoyant le message:



2.2. Configuration du Bitrate

La programmation du bitrate de la communication se fait en deux étapes.

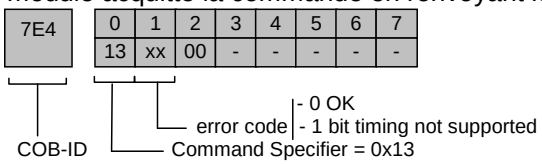
Il faut commencer par sélectionner le bitrate voulu en indiquant son index:



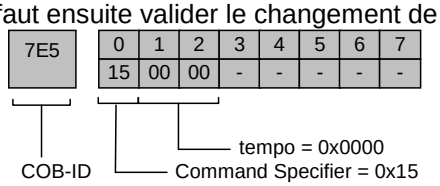
index	bitrate (kbps)	distance max*
0x00	1000	25m
0x01	800 (non implémenté)	50m
0x02	500	100m
0x03	250	250m
0x04	125	500m
0x05	50 (non implémenté)	1km
0x06	20 (non implémenté)	2,5km
0x07	10 (non implémenté)	5km

Les bitrates les plus importants ne permettent pas d'implémenter des longueurs de bus importantes. Suivant les applications, il peut être nécessaire de travailler avec des vitesses plus faibles.

Le module acquitte la commande en renvoyant le message



Il faut ensuite valider le changement de vitesse de communication en envoyant le message suivant:



**** Les distances indiquées sont inférieures à celles que l'on peut trouver dans la littérature, ces performances étant directement liées au délai de transfert des différents éléments de transmission de ligne. Les DMACs sont équipés d'éléments de protection sous forme d'opto-coupleurs qui introduisent des délais inhérents à cette technologie.***

2.3. Exemple de séquence de Configuration

Exemple de séquence de configuration pour passer le module à l'adresse 0x0A et au bitrate 125kbps.

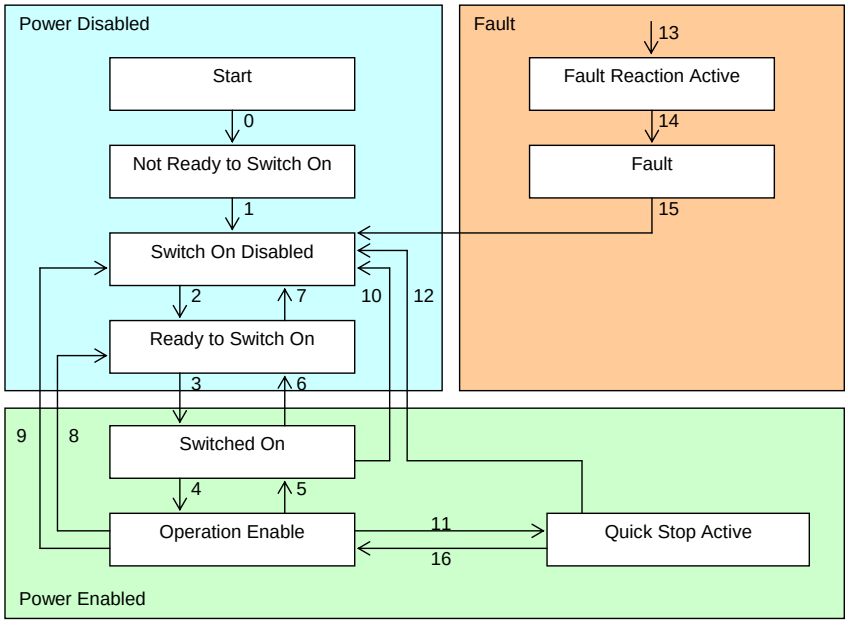
Si le bit rate actuel du module n'est pas connu au départ, il convient d'effectuer une phase de scrutation en faisant évoluer le bit rate de la ligne jusqu'à ce que le premier message de type LMT soit accepté (acquittement de la trame au sens Bus CAN).

```
7E5   04 01        Passage en mode configuration
7E5   11 0A        Affectation de la nouvelle adresse
7E4   11 00 00     Acquittement du module
7E5   13 00 04     Sélection du bitrate 125k
7E4   13 00 00     Acquittement du module
7E5   15 00 00     Validation du changement de bitrate
7E5   04 00        Sortie du mode configuration
```

3. Fonctionnalités Motion Control

3.1. Machine d'état

L'état du module est défini selon la spécification DSP402.



Les transitions entre les différents états sont soit gérées automatiquement par l'axe, soit contrôlées par l'utilisateur au moyen de l'objet Controlword (0x6040).

Transition	Evenement déclencheur	Action
0	Reset	initialisation de l'axe
1	Fin d'initialisation	-
2	Commande Shutdown Controlword = 0xxxx110	-
3	Commande SwitchOn Controlword = 0xxx0111	Mise sous puissance de l'axe
4	Commande Enable Operation Controlword = 0xxx1111	Mouvements autorisés
5	Commande Disable Operation Controlword = 0xxx0111	Mouvement arrêté (rampe) Axe reste sous puissance
6	Commande Shutdown Controlword = 0xxxx110	Axe hors puissance
7	Commande QuickStop ou disable voltage Controlword = 0xxxx01x ou 0x…xx0x	-
8	Commande Shutdown Controlword = 0xxxx110	Axe hors puissance
9	Commande Disable Voltage Controlword = 0xxxxx0x	Axe hors puissance
10	Commande QuickStop ou disable voltage Controlword = 0xxxx01x ou 0x…xx0x	Axe hors puissance
11	Commande QuickStop Controlword = 0xxxx01x	Mouvement arrêté (sans rampe) Axe reste sous puissance
12	Commande Disable Voltage ou automatic Selon quickstop option code Controlword = 0xxxxx0x	Axe hors puissance
13	Détection Fault	Disjonction
14	Disjonction effective	Mouvements non autorisés
15	Commande Fault Reset Controlword = 1xxxxxxx	-
16	Fin de QuickStop Controlword = 0xxx1111	Mouvements autorisés

3.2. Modes de mouvements

Les modules DMAC peuvent fonctionner dans un des modes suivants:

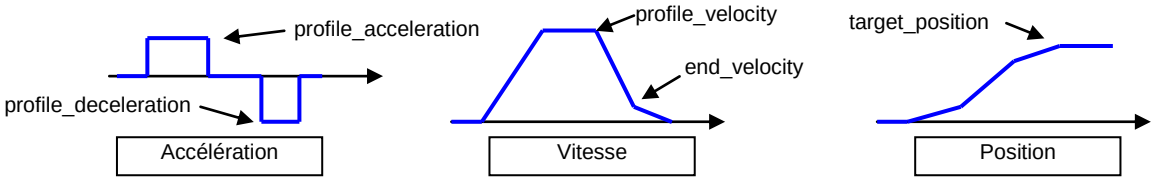
profile_position (positionnement)
profile_velocity (contrôle en vitesse)
homing (recherche d'origine)
interpolated_position (interpolation)

Le mode de mouvement se contrôle en modifiant l'objet modes_of_operation (0x6060).
Le mode de mouvement courant est indiqué dans l'objet modes_of_operation_display (0x6061).

3.3. Mode profile_position

Ce mode est utilisé pour effectuer des mouvements point-à-point avec le module.
Le mouvement est défini par les objets suivants:

0x607A	target_position	position de consigne (absolue ou relative)
0x6081	profile_velocity	vitesse du mouvement
0x6082	end_velocity	vitesse de fin de mouvement (asservissement de position)
0x6083	profile_acceleration	accélération du mouvement
0x6084	profile_deceleration	décélération du mouvement
0x6086	motion_profile_type	rampe trapézoïdale ou en sinus carré
0x6040	controlword	contrôle du mouvement



Pour lancer un mouvement, il faut que la machine d'état DS402 soit dans l'état Operation Enable et que l'objet modes_of_operation (0x6060) ait la valeur 1 (profile position).

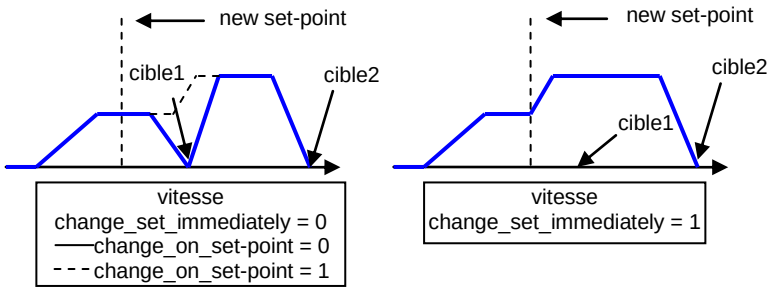
La mise à 1 du bit 4 (new_set-point) de l'objet controlword (0x6040) lance le mouvement (il doit être remis à 0 avant tout nouveau set-point)

Si le bit 6 (relative/absolute) du controlword est à 0, la consigne target_position représente la position absolue finale du mouvement. Si le bit 6 est à 1, la consigne target_position représente l'amplitude du déplacement par rapport à la position courante.

Dans le cas où une nouvelle commande de mouvement est envoyée (new_set-point) après avoir indiqué de nouvelles valeurs de consigne (vitesse et position).

- ✓ Le nouveau mouvement est différé à la fin du mouvement en cours lorsque le bit 5 (change_set_immediately) est à 0.
  - Lorsque le bit 9 (change_on_set-point) est positionné à 1, l'enchaînement des mouvements s'effectue selon la loi d'accélération pour obtenir la nouvelle vitesse de mouvement.
  - Lorsque le bit 9 (change_on_set-point) est positionné à 0, l'enchaînement des mouvements s'effectue selon les lois de décélération/accélération avec un passage à la vitesse nulle à la position attendue à la fin du premier mouvement en cours.
- ✓ Le nouveau mouvement est immédiatement effectué lorsque le bit 5 (change_set_immediately) est à 1.

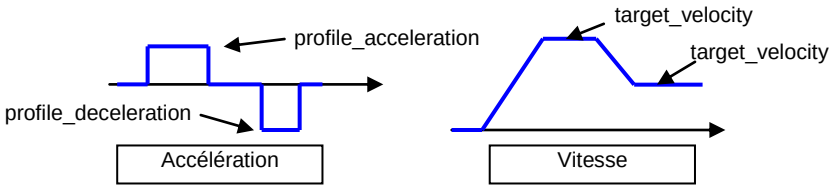
Nota : La capacité de bufferisation des mouvements successifs est de 1 étage.



3.4. Mode profile_velocity

Ce mode est utilisé pour effectuer un contrôle en vitesse du module.
Le mouvement est défini par les objets suivants:

0x60FF	target_velocity	vitesse de consigne (signée)
0x6083	profile_acceleration	accélération du mouvement
0x6084	profile_deceleration	décélération du mouvement
0x6086	motion_profile_type	rampe trapézoïdale ou en sinus carré



Pour lancer un mouvement, il faut que la machine d'état soit dans l'état Operation Enable et que l'objet modes_of_operation (0x6060) ait la valeur 3 (profile velocity).

La vitesse de consigne est donnée par la valeur absolue de l'objet target_velocity (0x60FF). Le sens de rotation est donné par le signe de ce même objet.

Lorsque la vitesse de consigne est modifiée, le module passe de la vitesse courante à la nouvelle vitesse de consigne en respectant les rampes d'accélération ou de décélération définies par les objets profile_acceleration (0x6083) et profile_deceleration (0x6084).

Une vitesse de consigne nulle provoque l'arrêt du module selon la loi de décélération.

3.5. Mode Homing

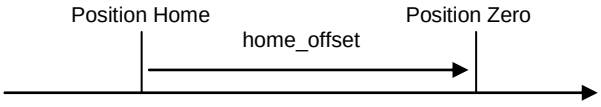
Ce mode est utilisé pour initialiser la position absolue du module.
Le mouvement est défini par les objets suivants:

0x607C	home_offset	différence entre la position Home et la position Zero
0x6098	homing_method	type de recherche d'origine
0x6099	homing_speeds	vitesse pendant la phase de recherche de capteur
0x609A	homing_acceleration	accélération pendant la phase de recherche de capteur

Pour lancer un homing, il faut que la machine d'état soit dans l'état Operation Enable et que l'objet modes_of_operation (0x6060) ait la valeur 6 (homing).
Le homing est déclenché par la mise à 1 du bit 4 (homing operation start) de l'objet controlword (0x6040). Ce bit est remis à 0 par le module lorsque la nouvelle origine a été prise en compte.

Seule la méthode 35 "Homing on the current position" est implémentée actuellement dans le module DMAC. Si l'objet homing_method (0x6098) contient une valeur différente de 35, aucune action n'est réalisée.

Au moment où le homing est déclenché, la position courante prend la valeur –(home_offset).



Les objets homing_speeds (0x6099) et homing_acceleration (0x609A) ne sont pas pris en compte.

4. Dictionnaire des objets

4.1. Objets de communication (DS301)

4.1.1. 0x1000 device_type

Data Type	VAR Unsigned32 ro
Description	type d'équipement. Valeur constante: 0x00060192 Les deux octets de poids fort (0x0006) indiquent que le DMAC est un Servo Drive et un Stepper Motor. Les deux octets de poids faible (0x0192 = 402) indiquent que le DMAC est compatible avec le profil DSP402

4.1.2. 0x1001 error_register

Data Type	VAR Unsigned8 ro
Description	L'objet Error Register montre l'état d'erreurs du DMAC par codage de bits. Le bit 0 est forcé dès qu'une erreur survient. La cause de la dernière erreur peut être déterminée en relisant l'objet error_code 0x603F Les erreurs sont signalées par l'envoi d'un message EMCY.

Bit	Message	Signification
0	generic error	Une erreur est survenue
1	Current	Surcourant moteur
2	Voltage	Surtension ou soustension
3	Temperature	Disjonction thermique
4	Communication Erreur	communication-réseau
5	Device profile specific	Erreur lors de l'exécution selon le profil spécifique
6	reserved	
7	Manufacturer specific	Message d'erreur spécifique fournisseur

4.1.3. 0x1002 manufacturer_status_register

Data Type	VAR Unsigned32 ro
Description	Registre d'Etat du module.

poids fort	bit 32	Mouvement interrompu anormalement
	bit 31	Erreur
	bit 30	
	bit 29	Busy
	bit 28	Mode boucle fermée BMAC
	bit 27	Asservissement de position en cours
	bit 26	Mouvement en cours
	bit 25	Puissance ON (1) ou OFF (0)
	bit 24	
	bit 23	Réservé
	bit 22	
	bit 21	
	bit 20	Etat Butée Soft négative
	bit 19	Etat Butée Soft positive
	bit 18	Etat Butée Hard négative
	bit 17	Etat Butée Hard positive
	bit 16	Réservé
	bit 15	Réservé
	bit 14	Polarité des sorties standard (0) ou inversée (1)
	bit 13	Polarité des entrées standard (0) ou inversée (1)
	bit 12	Rampes en « S » (1) ou Trapèze (0)
	bit 11	
	bit 10	
	bit 9	
	bit 8	Inversion de polarité des entrées butées hard
	bit 7	Autorisation des butées soft
	bit 6	Autorisation de la butée hard Inférieure
	bit 5	Autorisation de la butée hard Supérieure
	bit 4	Mode Courant Optimisé
	bit 3	
	bit 2	
poids faible	bit 1	

4.1.4. 0x1008 manufacturer_device_name

Data Type	VAR Visible String const
Description	Désignation du produit. Exemple: "DMAC17-C"

4.1.5. 0x1009 manufacturer_hardware_version

Data Type	VAR Visible String const
Description	date de fabrication – date de la dernière révision du produit. Exemple: "02/06/10 – 02/06/10"

4.1.6. 0x100A manufacturer_software_version

Data Type	VAR Visible String const
Description	Version du soft. Exemple: "EV v1.11 9148"

4.1.7. 0x1010 STORE Parameters

Data Type	Array unsigned32 rw
Description	Définit le type de sauvegarde des paramètres du module

Objet	save_all_parameters
Sous-Index	0x01
Description **Valeur en lecture**	Sauvegarde de l'ensemble des paramètres 0x00 pas de sauvegarde

Objet	save_communication_parameters
Sous-Index	0x02
Description **Valeur en lecture**	Sauvegarde des paramètres de communication (1000h à 1FFFh) 0x00 pas de sauvegarde communication

Objet	save_application_parameters
Sous-Index	0x03
Description **Valeur en lecture**	Sauvegarde des paramètres application (6000h à 9FFFh) 0x00 pas de sauvegarde application

Objet	save_manufacturer_defined_parameters
Sous-Index	0x04
Description **Valeur en lecture**	Sauvegarde des paramètres définis 0x02 Sauvegarde automatique à l'écriture de chaque paramètre

4.1.8. 0x1011 RESTORE Default Parameters

Data Type **Description**	Array unsigned32 Permet de revenir aux paramètres usine

Objet	restore_all_default_parameters
Sous-Index	0x01
Description **Valeur en lecture**	 0x00 pas de restore all default parameters

Objet	restore_communication_default_parameters
Sous-Index	0x02
Description **Valeur en lecture**	 0x00 pas de restore communication default parameters (0x1000 à 0x1FFF)

Objet	restore_application_default_parameters
Sous-Index	0x03
Description **Valeur en lecture**	 0x00 pas de restore application default parameters (0x6000 à 0x9FFF)

Objet	restore_manufacturer_defined_default_parameters
Sous-Index	0x04
Description **Valeur en lecture**	L'écriture de la clé "load" 0x64616F6Ch indique que le prochain Reset ou power on se fera avec les paramètres usine qui seront les valeurs sauvegardées 0x01 autorise restore manufacturer defined default parameters

4.1.9. 0x1014 COB-ID_emergency_message

Data Type	VAR Unsigned32 ro
Description	Identifiant des messages de type EMERGENCY Valeur: 0x00000080 + node_ID

4.1.10. 0x1017 producer_heartbeat_time

Data Type	VAR Unsigned16 rw
Description	Intervalle de temps entre deux messages Heartbeat. Exprimé en millisecondes. Si producer_heartbeat_time est nul, la fonctionnalité de Heartbeat est inhibée. Valeur : 0 à défaut

4.1.11. 0x1018 identity_object

Data Type	RECORD

Objet	number_of_entries
Sous-Index	0x00
Data Type	Unsigned8 ro
Description	nombre de sous-index

Objet	vendor_ID
Sous-Index	0x01
Data Type	Unsigned32 ro
Description **Valeur en lecture**	numéro de fabricant (délivré par CIA) 0x00 00 02 C9

Objet	product_code
Sous-Index	0x02
Data Type	Unsigned32 ro
Description **Valeur en lecture**	référence du produit

DMAC17	0x4B 31 34 38
DMAC23	0x4B 31 33 38
DMAC34	0x4B 31 34 32
BMAC	0x4B 31 38 39

Objet	revision_number
Sous-Index	0x03
Data Type	Unsigned32 ro
Description **Valeur en lecture**	numéro de version soft (16 bits) revision (16 bits) du produit Exemple : 0x00 01 00 0B

Objet	serial_number
Sous-Index	0x04
Data Type	Unsigned32 ro
Description **Valeur en lecture**	numéro de série du produit 0x .. ..

4.1.12. 0x1400…1403 RPD01…4_Communication_parameters

Cf chap. Process Data Object

4.1.13. 0x1600…1603 RPDO1…4_mapping_parameters

Cf chap. Process Data Object

4.1.14. 0x1800…1803 TPDO1…4_communication_parameters

Cf chap. Process Data Object

4.1.15. 0x1A00…1A03 TPDO1…4_mapping_parameters

Cf chap. Process Data Object

4.2. Objets spécifiques manufacturer

4.2.1. 0x2000 volatile_user_variables

Data Type	ARRAY Integer32 rw

Objet	number_of_entries
Sous-Index	0x00
Data Type	Unsigned8 ro
Description	nombre de sous index de l'objet.

Objet	volatile_user_variables
Sous-Index	0x01 à 0x20
Description	Chacun des 32 sous-index de l'objet volatile_user_variables contient une variable de 4 octets utilisable par l'utilisateur pour stocker des valeurs. Ces variables sont réinitialisées à zéro à chaque reset ou mise sous tension du module.

4.2.2. 0x2001 non-volatile_user_variables

Data Type	ARRAY Integer32 rw

Objet	number_of_entries
Sous-Index	0x00
Data Type	Unsigned8 ro
Description	nombre de sous index de l'objet.

Objet	non-volatile_user_variables
Sous-Index	0x01 à 0x08
Description	Chacun des 8 sous-index de l'objet non_volatile_user_variables contient une variable de 4 octets utilisable par l'utilisateur pour stocker et mémoriser des valeurs. Ces variables sont mémorisées dans le module à chaque reset et restaurées automatiquement à chaque mise sous tension du module.

4.2.3. 0x2002 user_timers

Data Type	ARRAY Unsigned32 rw

Objet	number_of_entries
Sous-Index	0x00
Data Type	Unsigned8 ro
Description	nombre de sous index de l'objet.

Objet	user_timers
Sous-Index	0x01 à 0x03
Description	Chacun des 3 sous-index de l'objet user_timers contient une variable de 4 octets utilisable par l'utilisateur pour servir de référence de temps. Ces timers doivent être initialisés à une valeur positive par l'utilisateur puis sont automatiquement décrémentés toutes les millisecondes, jusqu'à atteindre zéro.

4.2.4. 0x2003 position_capture

Data Type	VAR integer32 ro
Description	Cet objet contient la position du module qui a été mémorisée lors d'un front montant de l'entrée logique IN5

4.2.5. 0x2004 temperature_object

Data Type	RECORD

Objet	number_of_entries
Sous-Index	0x00
Data Type	Unsigned8 ro
Description	nombre de sous index de l'objet.

Objet	CPU_temperature
Sous-Index	0x01
Data Type	Unsigned32 ro
Description	Température du microprocesseur. Exprimé en dixièmes de degré Celsius.

Objet	Driver_temperature (DMAC34, BMAC)
Sous-Index	0x02
Data Type	Unsigned32 ro
Description	Température de l'étage de puissance. Exprimé en dixièmes de degré Celsius.

Objet	Motor_temperature (DMAC34)
Sous-Index	0x03
Data Type	Unsigned32 ro
Description	Température du moteur. Exprimé en dixièmes de degré Celsius.

4.2.6. 0x2005 analog_inputs

Data Type	RECORD

Objet	number_of_entries
Sous-Index	0x00
Data Type	Unsigned8 ro
Description	nombre de sous index de l'objet.

Objet	analog_inputs1
Sous-Index	0x01
Data Type	Unsigned32 ro
Description	Entrée analogique du module. Exprimé en millivolts

Objet	analog_inputs2 (DMAC34)
Sous-Index	0x02
Data Type	Unsigned32 ro
Description	Seconde entrée analogique du module. Exprimé en millivolts

4.2.7. 0x2006 analog_outputs (DMAC34)

Data Type	RECORD

Objet	number_of_entries
Sous-Index	0x00
Data Type	Unsigned8 ro
Description	nombre de sous index de l'objet.

Objet	analog_outputs1
Sous-Index	0x01
Data Type	Unsigned32 rw
Description	Première sortie analogique du module. Exprimé en millivolts

Objet	analog_outputs2
Sous-Index	0x02
Data Type	Unsigned32 rw
Description	Seconde sortie analogique du module. Exprimé en millivolts

4.2.8. 0x2008 current_control

Data Type	VAR Unsigned16 rw
Description	Le bit 0 de la variable permet d'activer (1) ou de désactiver (0) le mode "courant optimisé"

0x2009 close_loop_config (BMAC)

Data Type	VAR Unsigned8 rw
Description	Inhibe (valeur 0x00) ou autorise (autre valeur) le mode boucle fermée du BMAC.

4.2.9. 0x207D soft_ends_config

Data Type	VAR Unsigned8 rw
Description	Inhibe (valeur 0x00) ou autorise (autre valeur) l'utilisation des butées software.

4.2.10. 0x20FD digital_inputs_config

Data Type	RECORD

Objet	number_of_entries
Sous-Index	0x00
Data Type	Unsigned8 ro
Description	nombre de sous index de l'objet.

Objet	Digital_inputs_config_alternative
Sous-Index	0x01
Data Type	Unsigned32 rw
Description	Les entrées IN1 et IN2 peuvent être utilisées en entrées standard ou en entrée butée hard. Butée positive sur l'entrée IN1 Butée négative sur l'entrée IN2 Les bits 16 et 17 de l'objet permettent d'activer (1) ou de désactiver (0) la Fonction butée respectivement sur IN1 et IN2

Objet	Digital_inputs_config_polarity
Sous-Index	0x02
Data Type	Unsigned32 rw
Description	Gère de manière globale la polarité des entrées logiques (ainsi que des butées hard) 0 polarité normale Autre polarité inverse

4.2.11. 0x20FE digital_ouputs_config

Data Type	RECORD

Objet	number_of_entries
Sous-Index	0x00
Data Type	Unsigned8 ro
Description	nombre de sous index de l'objet.

Objet	Digital_ouputs_config_alternative
Sous-Index	0x01
Data Type	Unsigned32 rw
Description	Les sorties OUT1 et OUT2 peuvent être utilisées en sorties standard ou en Fonction alternative : - Information BUSY sur la sortie OUT1 - Information DEFAUT sur la sortie OUT2 Les bits 16 et 17 de l'objet permettent d'activer (1) ou de désactiver (0) la Fonction alternative respectivement sur OUT1 et OUT2

Objet	Digital_outputs_config_polarity
Sous-Index	0x02
Data Type	Unsigned32 rw
Description	Gère de manière globale la polarité des sorties logiques 0 polarité normale Autre polarité inverse

4.2.12. 0x2083 profile_acceleration_time

Data Type	VAR Unsigned32 ro
Description	Accélération des mouvements avec profil. Exprimé en millisecondes. Correspond au temps de rampe entre la vitesse nulle et profile_velocity La valeur de profile_acceleration_time est actualisée à chaque modification de profile_acceleration ou de target_velocity.

4.2.13. 0x2084 profile_deceleration_time

Data Type	VAR Unsigned32 ro
Description	Décélération des mouvements avec profil. Exprimé en millisecondes. Correspond au temps de rampe entre profile_velocity et la vitesse nulle. La valeur de profile_deceleration_time est actualisée à chaque modification de profile_deceleration ou de target_velocity.

4.3. Objets motion control (DSP402)

4.3.1. 0x603F error_code

Data Type	VAR Unsigned16 ro
Description	Fournit le détail de la dernière erreur relevée (détail dans la norme DSP402).

4.3.2. 0x6040 controlword

Data Type	VAR Unsigned16 rw
Description	Contrôle l'état du module.

Bits	Description		
	Profile Position	**Profile Velocity**	**Homing**
0	Switch On		
1	Enable Voltage		
2	/ Quick Stop		
3	Enable Operation		
4	new setpoint	reserved	Homing Operat. start
5	change set immediat.	reserved	reserved
6	relative / absolute	reserved	reserved
7	Reset Fault		
8	Halt		
9-15	Reserved for future use		

4.3.3. 0x6041 statusword

Data Type	**VAR Unsigned16 ro**
Description	**Décrit l'état du module.**

Bits	Description		
	Profile Position	**Profile Velocity**	**Homing**
0	Ready to Switch On		
1	Switched On		
2	Operation Enabled		
3	Fault		
4	Voltage Enabled		
5	/ Quick Stop		
6	Switch On Disabled		
7	Warning		
8	Manufacturer Specific		
9	Remote		
10	Target Reached		
11	Internal Limit Active		
12	setpoint acknowledge	Speed = 0	homing attained
13	following error	max slippage error	homing error
14-15	Manufacturer Specific		

4.3.4. 0x605A quickstop_option_code

Data Type	VAR Integer16 rw
Description	Type d'arrêt effectué par la fonction QuickStop. Valeurs possibles: 1: Slow-Down Ramp 3: Current Limit 5: Slow-Down Ramp, stay in QuickStop 7: Current Limit, stay in QuickStop Valeur à défaut : 1 Si quickstop_option_code vaut 1 ou 3, le module passe dans l'état Switch_On_Disabled (puissance moteur coupée) lorsqu'il s'est arrêté. Si quickstop_option_code vaut 5 ou 7, le module reste dans l'état QuickStop (le moteur reste sous puissance) lorsqu'il s'est arrêté.

4.3.5. 0x6060 modes_of_operation

Data Type	VAR Integer8 wo
Description	Sélectionne le mode d'opération du module. Valeurs possibles: 1: Profile Position Mode 3: Profile Velocity Mode 6: Homing Mode -1 : Phasage Réservé à un usage interne Valeur à défaut : 0 Le mode d'opération courant peut être relu dans l'objet 0x6061 Modes Of Operation Display

4.3.6. 0x6061 modes_of_operation_display

Data Type	VAR Integer8 ro
Description	Visualise le mode d'opération courant (voir objet 0x6060 modes_of_operation)

4.3.7. 0x6063 position_internal_value (BMAC)

Data Type	VAR Integer32 ro
Description	Position courante du codeur. Exprimé en incréments codeur.

4.3.8. 0x6064 position_actual_value

Data Type	VAR Integer32 ro
Description	Position courante de l'axe. Exprimé en 1/10000ème de tour moteur sur DMAC ou BMAC en boucle fermée. Dans le cas du BMAC en mode boucle ouverte, cette valeur exprime la position électrique en micropas (il y a 50 micropas par pas).

4.3.9. 0x6069 velocity_sensor_actual_value

Data Type	VAR Integer32 ro
Description	Vitesse de rotation du moteur Exprimé en incréments par secondes (1 inc = 1/10000ème de tour)

4.3.10. 0x606A sensor_selection_code

Data Type	VAR integer16 ro
Description	Type de calcul de l'objet Velocity_sensor_actual_value Valeur constante: Sensor_selection_code = 0 (la vitesse est obtenue à partir d'un codeur de position)

4.3.11. 0x606C velocity_actual_value

Data Type	VAR Integer32 ro
Description	Vitesse théorique (calculée) à laquelle tourne le module. Unités = 100ème de tour par minute Les vitesses négatives représentent une rotation dans le sens anti-horaire.

4.3.12. 0x6072 max_torque

Data Type	VAR Unsigned16 rw
Description	Couple maximum appliqué sur l'axe moteur DMAC (sans objet sur BMAC) Exprimé en pourmille du couple nominal (voir motor_rated_torque 0x6076)

4.3.13. 0x6073 max_current (BMAC)

Data Type	VAR Unsigned16 rw
Description	Courant efficace maximum injecté dans chaque bobine moteur (BMAC seulement). Exprimé en pourmille du courant moteur (voir motor_rated_current 0x6075)

4.3.14. 0x6075 motor_rated_current (BMAC)

Data Type	VAR Unsigned32 rw
Description	Courant nominal du moteur connecté sur BMAC. L'utilisateur doit programmer cette valeur avec la donnée spécifiée dans la documentation ou sur la plaque signalétique du moteur. Exprimé en mA.

4.3.15. 0x6076 motor_rated_torque

Data Type	VAR Unsigned32 ro
Description	Couple nominal du moteur DMAC (sans objet sur BMAC) Exprimé en milliNewton-mètre (mNm)

4.3.16. 0x6079 DC_link_circuit_voltage

Data Type	VAR Unsigned32 ro
Description	Tension d'alimentation du module. Exprimée en millivolts

4.3.17. 0x607A target_position

Data Type	VAR Integer32 rw
Description	Consigne du mouvement. Exprimé en incréments (1/10000ème de tour). En mode *profile position*, *target_position* est la position de consigne du mouvement. En fonction du bit absolute/relative du *controlword*, *target_position* est interprété comme absolu par rapport à l'origine ou relatif par rapport à la position courante.

4.3.18. 0x607C home_offset

Data Type	VAR Integer32 rw
Description	différence entre la Position Zero qui correspond à la référence de tous les mouvements et la Position Home trouvée lors d'une séquence de homing. home_offset = -(home_pos)

4.3.19. 0x607D software_position_limit

Data Type	ARRAY Integer32 rw

Objet	number_of_entries
Sous-Index	0x00
Data Type	Unsigned8 ro
Description	nombre de sous index de l'objet.

Objet	min_position_limit
Sub-Index	0x01
Description	Limite minimale de position de l'axe exprimée en 1/10000eme de tours. Lorsque l'utilisation des butées software est autorisée (objet 0x207D soft_ends_config), si cette limite est dépassée, l'axe est automatiquement arrêté. Seuls les mouvements dans le sens inverse sont acceptés.
Default Value	-100 000 (10 tours moteur)

Objet	max_position_limit
Sub-Index	0x02
Description	Limite maximale de position de l'axe exprimée en 1/10000eme de tours. Lorsque l'utilisation des butées software est autorisée (objet 0x207D soft_ends_config), si cette limite est dépassée, l'axe est automatiquement arrêté. Seuls les mouvements dans le sens inverse sont acceptés.
Default Value	100 000 (10 tours moteur)

0x6081 profile_velocity

Data Type	VAR Unsigned32 rw
Description	Vitesse de consigne du mode Profile Position en 1/100 t/min.

4.3.20. 0x6082 end_velocity

Data Type	VAR Unsigned32 rw
Description	Vitesse de l'axe à l'entrée dans le mode d'asservissement de position en 1/100 t/min. A la fin d'un mouvement de type profile position, le module décélère entre la vitesse profile velocity et la vitesse end velocity. L'asservissement de position est effectué sur 0,1 tour moteur avant d'atteindre la position de consigne.

4.3.21. 0x6083 profile_acceleration

Data Type	VAR Unsigned32 rw
Description	Accélération des mouvements avec profil. Exprimé en incréments par secondes au carré. (inc/s²) L'accélération minimale que le module peut accepter est limitée par un temps de rampe total de 12 secondes entre une vitesse nulle et la vitesse profile_velocity.

4.3.22. 0x6084 profile_deceleration

Data Type	VAR Unsigned32 rw
Description	Décélération des mouvements avec profil. Exprimé en incréments par secondes au carré. (inc/s²) La décélération minimale que le module peut accepter est limitée par un temps de rampe total de 12 secondes entre la vitesse profile_velocity et une vitesse nulle.

4.3.23. 0x6086 motion_profile_type

Data Type	VAR Integer16 rw
Description	Type de profil de vitesse utilisé pour les modes Profile Position et Profile Velocity. 0: profile de vitesse trapézoïdal 1: profile de vitesse en sinus carré

4.3.24. 0x6098 homing_method

Data Type	VAR Integer8 rw
Description	Type de recherche d'origine. Seule la valeur 35, correspondant à la méthode "Homing on current position", est implémentée dans les modules DMAC.

4.3.25. 0x6099 homing_speeds

Data Type	ARRAY Unsigned32 rw
Description	Vitesses utilisées pendant les phases de homing

Objet	number_of_entries
Sous-Index	0x00
Data Type	Unsigned8 ro
Description	nombre de sous index de l'objet.

Objet	speed_during search_for_switch
Sous-index	0x01
Description	Sans objet. Non pris en compte.

Objet	speed_during search_for_zero
Sous-index	0x02
Description	Sans objet. Non pris en compte.

4.3.26. 0x60FD digital_input

Data Type	VAR Unsigned32 ro
Description	Entrées logiques du module (en binaire actif à 1, inactif à 0).

Bit	31 - 22	21	20	19	18	17	16	15 - 0
Sortie	-	IN6	IN5	IN4	IN3	IN2	IN1	

4.3.27. 0x60FE digital_output

Data Type	ARRAY unsigned32 rw

Objet	number_of_entries
Sous-Index	0x00
Data Type	Unsigned8 ro
Description	nombre de sous index de l'objet.

Objet	physical_output
Sous-index	0x01
Description	Sorties logiques du module (en binaire actif à 1, inactif à 0).

Bit	31 à 20	19	18	17	16	15 à 1	0
Sortie	-	OUT4	OUT3	OUT2	OUT1	-	Brake

Objet	bitmask
Sous-index	0x02
Description	Masque sur les sorties logiques du module. 1 si physical_outputs doit être appliqué la sortie, 0 dans le cas contraire (la sortie reste dans son état actuel).

Bit	31 à 20	19	18	17	16	15 à 1	0
Entrée	-	OUT4	OUT3	OUT2	OUT1	-	Brake

4.3.28. 0x60FF target_velocity

Data Type	VAR Integer32 rw
Description	Vitesse de consigne du mode Profile Velocity en 1/100 t/min Le signe de la consigne définit le sens de la rotation de l'axe. Une consigne nulle correspond à l'arrêt de l'axe.

4.3.29. 0x6404 motor_manufacturer

Data Type	VAR String const
Description	"MIDI INGENIERIE"

4.3.30. 0x6502 supported_drive_modes

Data Type	VAR Unsigned32 ro
Description	Modes supportés par le module, à savoir: 0x65 (profile position + profile velocity + homing + interpolated position)

4.3.31. 0x6504 drive_manufacturer

Data Type	VAR String const
Description	"MIDI INGENIERIE"

4.3.32. 0x6510 drive_data

Data Type	RECORD

Objet	number_of_entries
Sous-index	0x00
Data Type	Unsigned8 ro
Description	nombre de sous-index

Objet	phasage moteur
Sous-index	0x01
Data Type	Unsigned16 ro
Description	Valeur de phasage du moteur.

5. Résumé des objets du dictionnaire

Index (hex)	Sous-index (hex)	Objet	Type	Format	Attribut	Mémorisé hors tension
1000		device_type	VAR	Unsigned32	ro	◊
1001		error_register	VAR	Unsigned8	ro	
1002		manufacturer_status_register	VAR	Unsigned32	ro	
1008		manufacturer_device_name	VAR	Visible String	const	◊
1009		manufacturer_hardware_version	VAR	Visible String	const	◊
1010		Store parameters	ARRAY	Unsigned32	rw	
	01	Save all parameters				
	02	Save communication parameters				
	03	Save application parameters				
	04	Save manufacturer defined parameters				
1011		Restore parameters	ARRAY	Unsigned32	rw	
	01	Restore all parameters				
	02	Restore communication default parameters				
	03	Save application default parameters				
	04	Save manufacturer defined default parameters				
100A		manufacturer_software_version	VAR	Visible String	const	◊
1014		COB-ID_EMERGENCY_message	VAR	Unsigned32	ro	
1017		producer_heartbeat_time	VAR	Unsigned16	rw	
1018		identity_object	RECORD			◊
	00	number_of_entries	VAR	Unsigned8	ro	◊
	01	vendor_ID	VAR	Unsigned32	ro	◊
	02	product_code	VAR	Unsigned32	ro	◊
	03	revision_number	VAR	Unsigned32	ro	◊
	04	serial_number	VAR	Unsigned32	ro	◊
1400		RPDO1_parameters	RECORD			
1401		RPDO2_parameters	RECORD			
1402		RPDO3_parameters	RECORD			
1403		RPDO4_parameters	RECORD			
1600		RPDO1_mapping	RECORD			
1601		RPDO2_mapping	RECORD			
1602		RPDO3_mapping	RECORD			
1603		RPDO4_mapping	RECORD			
1800		TPDO1_parameters	RECORD			
1801		TPDO2_parameters	RECORD			
1802		TPDO3_parameters	RECORD			
1803		TPDO4_parameters	RECORD			
1A00		TPDO1_mapping	RECORD			
1A01		TPDO2_mapping	RECORD			
1A02		TPDO3_mapping	RECORD			
1A03		TPDO4_mapping	RECORD			
2000	1-20	volatile_user_variables	ARRAY	Integer32	rw	
2001	1-8	non-volatile_user_variable	ARRAY	Integer32	rw	◊
2002	1-3	user_timers	ARRAY	Unsigned32	rw	
2003		position_capture	VAR	Integer32	ro	
2004		temperature_object	RECORD			
	00	number_of_entries		Unsigned8	ro	
	01	CPU_temperature		Unsigned32	ro	
	02	Driver_temperature (DMAC34, BMAC)		Unsigned32	ro	
	03	Motor_temperature (DMAC34)		Unsigned32	ro	
2005		analog_Inputs	RECORD			
	00	number_of_entries		Unsigned8	ro	
	01	analog_Inputs1		Unsigned32	ro	
	02	analog_Inputs2		Unsigned32	ro	

2006		analog_outputs	RECORD			
	00	number_of_entries		Unsigned8	ro	
	01	analog_outputs1		Unsigned32	rw	
	02	analog_outputs2		Unsigned32	rw	
2008		current_control	VAR	Unsigned16	rw	◊
2009		close_loop_control (BMAC)	VAR	Unsigned8	rw	◊
207D		soft_ends_config	VAR	Unsigned8	rw	◊
2083		profile_acceleration_time	VAR	Unsigned32	ro	◊
2084		profile_deceleration_time	VAR	Unsigned32	ro	◊
20FD		digital_inputs_config	RECORD			◊
	00	number_of_inputs		Unsigned8	ro	◊
	01	digital_inputs_config_alternative		Unsigned32	rw	◊
	02	digital_inputs_config_polarity		Unsigned32	rw	◊
20FE		digital_outputs_config	RECORD			◊
	00	number_of_outputs		Unsigned8	ro	◊
	01	digital_outputs_config_alternative		Unsigned32	rw	◊
	02	digital_outputs_config_polarity		Unsigned32	rw	◊
603F		error_code	VAR	Unsigned16	ro	
6040		controlword	VAR	Unsigned16	rw	
6041		statusword	VAR	Unsigned16	ro	
605A		quickstop_option_code	VAR	Integer16	rw	
6060		modes_of_operation	VAR	Integer8	wo	
6061		modes_of_operation_display	VAR	Integer8	ro	
6063		position_internal_value (BMAC)	VAR	Integer32	ro	
6064		position_actual_value	VAR	Integer32	ro	
6069		velocity_sensor_actual_value	VAR	Integer32	ro	
606A		sensor_selection_code	VAR	Integer16	ro	
606C		velocity_actual_value	VAR	Integer32	ro	
6072		max_torque	VAR	Unsigned16	rw	◊
6073		max_current (BMAC)	VAR	Unsigned16	rw	◊
6075		motor_rated_current (BMAC)	VAR	Unsigned32	rw	◊
6076		motor_rated_torque	VAR	Unsigned32	ro	◊
6079		DC_link_circuit_voltage	VAR	Unsigned32	ro	
607A		target_position	VAR	Integer32	rw	
607C		home_offset	VAR	Integer32	rw	
607D		software_position_limit	ARRAY	Integer32	rw	◊
	01	min_position_range_limit				◊
	02	max_position_range_limit				◊
6081		profile_velocity	VAR	Unsigned32	rw	◊
6082		end_velocity	VAR	Unsigned32	rw	◊
6083		profile_acceleration	VAR	Unsigned32	rw	◊
6084		profile_deceleration	VAR	Unsigned32	rw	◊
6086		motion_profile_type	VAR	Integer16	rw	
6098		homing_method	VAR	Integer8	rw	
6099		homing_speeds	ARRAY	Unsigned32	rw	
	01	speed_during_search_for_switch				
	02	speed_during_search_for_zero				
60FD		digital_input	VAR	Unsigned32	ro	
60FE		digital_output	ARRAY			
	01	physical_outputs		Unsigned32	rw	
	02	bitmask		Unsigned32	rw	
60FF		target_velocity	VAR	Integer32	rw	
6404		motor_manufacturer	VAR	String Visible	const	◊
6502		supported_drive_modes	VAR	Unsigned32	ro	◊
6504		drive_manufacturer	VAR	String Visible	const	◊
6510		drive_data	RECORD			◊
	00	number_of_entries		Unsigned8	ro	◊
	01	phasage moteur		Unsigned16	ro	◊