Title **BONMET paramterization**

Subject Description of parameters/register accessed and fieldbus MODBUS and CANOPEN

Revisions

* تغییراطلاعات مندرج در راهنمای مربوط به این محصول، بدون اطلاع قبلی امکان پذیر می باشد.

1.0	20 december 2010	E.Cavina	Initial revision
1.1	16 march 2011	E.Cavina	Revision on canopen release

Summary

1. Parameter Settings

Comunication parameters are accessible via Modbus or CanOpen: the following table shows register addresses and format details.

Servo drive can be controlled in the following ways:

1. using analog and digital inputs and Bonmet control modes
2. using RS232/RS485 Modbus and Bonmet control modes
3. using CanOPEN/Dsp402 control modes

To enable CanOPEN user need to change parameter (184), (Fieldbus selection) to 2 (CanOPEN) and to setup parameters 188 (node-id) and 189 (canbus bitrate) accordingly to network configuration.

Comunication Parameters

The following describes the communication addresses for the communication parameters.

Modbus Address	Pn	Attrib	Parameter	Parameter description	Range	Default
0x12F7	184	R/W	Fieldbus selection	Choose fieldbus interface between the following: 0 ：Modbus RS232 interface 1 ：Modbus RS485 interface 2 ：CanOPEN using Can-Bus 3 ：CanOPEN over EtherCAT (*1)	0 ~ 3	0
0x12F8	185	R/W	Modbus Address	Set the servo fieldbus address in ModBus communication NOTE: address must be set to 1 to communicate with Servofly.	1 ~ 247	1
0x12F9	186	R/W	Modbus Communication Rate	Set Modbus communication rate; the following baudrates are supported: 0 - 》9600bps ; 1 - 》19200 bps ; 2 - 》38400 bps ; 3 - 》57600 bps ; 4 - 》115200 bps (The communication rate must be set to 115200bps when communicate with Servofly)	0 ~ 4	4

0x12FA	187	R/W	ModBus Communication Frame Type	Set the frame type in ModBus communication; three frame types are supported as following: 0 - 》8 bit, no parity bit, 2 stop bit ; 1 - 》8 bit, odd parity bit, 2 stop bit ; 2 - 》8 bit, even parity bit, 2 stop bit.	0 ~ 1	1
-	188	R/W	Canbus NodeID	Set can-bus Node-ID	1 ~ 127	1
-	189	R/W	Canbus Bitrate	Set the communication Rate for can-bus communication; the following baudrates are supported: 0 = 20 Kbps 1 = 50 Kbps 2 = 125 Kbps 3 = 250 Kbps 4 = 500 Kbps 5 = 1000Kbps	0 ~ 5	5

Note：R/W for Read And Write, RO for Read Only, RC for Read And Clear.

Synchronization Parameters

Write 0x1C27 to 2,select synchronization control mode, users can control the drives through synchronization command, then trigger "Synchronous command enable", servo system will operate as command given.**"_IQx" means** $\text{value}/2^x$**.**

Modbus Address	Canbus Object	Attributes	Parameter	Parameter description	Range	Default
0x1C27	-	R/W	ModBus synchronous command selection	①This parameter is used to select the input for position/speed/torque control mode; ②Function as follows: 0：Synchronous command invalid，drive would receive command through traditional I/O mode only. 1：Manufacturer parameter, doesn't open to customers. 2：Synchronous command valid, drive would receive synchronous command, traditional I/O command is invalid.	0 ~ 2	0
0x1C28	-	R/W	Synchronous torque command	①This parameter is used in synchronous torque control mode as torque command; ②Data format is _IQ7; ③The corresponding physical quantity of data is Q-axis current of the motor, the unit is Amps; ④This parameter is valid in synchronous torque control mode, when ModBus synchronous command is valid and set "Synchronous command enable" ON.	-65535 ~ 65535	0
0x1C29	-	R/W	Synchronous speed command	①This parameter is used in synchronous speed control mode as speed command; ②Data format is _IQ2; ③The corresponding physical quantity of data is speed, the unit is rpm; ④This parameter is valid in synchronous speed control mode, when ModBus synchronous command is valid and set "Synchronous command enable" ON.	-65535 ~ 65535	0
0x1C2A	-	R/W	High bit synchronous position command	①This parameter is used in synchronous position control mode as high bit of position command; ②Data format is _IQ0; ③The corresponding physical quantity of data is the motor position, the unit is pulse. The parameter is directly spliced by (H, L) to form a 32-bit position command. ④This parameter is valid in synchronous position control mode, when ModBus synchronous command is valid and set "Synchronous command enable" ON.	-65535 ~ 65535	0
0x1C2B	-	R/W	Low bit synchronous position command	①This parameter is used in synchronous position control mode as low bit of position command; ②Data format is _IQ0; ③The corresponding physical quantity of data is the motor position, the unit is pulse. The parameter is directly spliced by (H,	-65535 ~ 65535	0

				L) to form a 32-bit position command. ④This parameter is valid in synchronous position control mode, when ModBus synchronous command is valid and set "Synchronous command enable" ON.		
0x1C55	-	WC	Synchronous command enable	①This parameter is used to trigger the parameters of synchronous torque, speed, position register effective; ②Write 0x55AA to the register, "Synchronous command enable" is ON, other value is in invalid.	- 65535 ~ 65535	0

Monitor Function Parameters

Modbus Address	Canbus Object	Attributes	Parameter	Parameter description	Range	Default
0x1C62	-	RO	The implementation state of EEPROM operation	①This parameter is used for DSP and application interaction, the application queries the value of the parameter to determine the EEPROM operation is completed or not; ②Function as follows: 0 - 》Operation completed successfully; 1 - 》operation in progress; 2 - 》Operation failed;	-	-
0x1C64	-	RO	FPGA version	FPGA version number is stored in this parameter for application inquiry;	-	-
0x1C65	-	RO	Index Reset Status	①This parameter instructs encoder counter to complete index reset operation or not; ②Function as follows: [3] U phase signal of encoder; [2] V phase signal of encoder; [1] W phase signal of encoder; [0] Home position reset completed;	-	-
0x1C66	-	RO	Index reset encoder counts	①This parameter is used to latch encoder count value when index reset operation carried out. ②The hardware carries out latch operation during home position reset operation, the parameter is used to guarantee the coherence of pulse counting.	-	-
0x1C67	-	RO	Real-time encoder counts	①This parameter indicates the current value of the encoder counter, the unit is pulse; ②This parameter reflects the physical location of the motor shaft;	-	-
0x1C68	-	RO	CN2 input terminal status	①This parameter indicates the signal terminal state of CN2 input interface; ②Function as follows: [9] ServoEn status; [8] AlarmClr status; [7]CCWDis status; [6] CWDis status; [5] CCWLtd status; [4] CWLtd status; [3] CLE status; [2] INH status; [1] SignInv status; [0] PulseInv status;	-	-
0x1C69	-	RO	Encoder input terminal status	①This parameter indicates the status of encoder input terminals of; ②Function as follows: [5] U Hall signal status; [4] V Hall signal status; [3] W Hall signal status; [2] A signal status; [1] B signal status; [0] Z signal (Index) status;	-	-

0x1C6A	-	RO	CN2 output terminal status	①This parameter indicates the signal terminal state of CN2 output interface; ②Function as follows: [3] SRDY status; [2] Alarm status; [1] COIN status; [0] BRK status;	-	-
0x1C6B	-	RO	The analog voltage of analog input port A	①This parameter indicates the actual input voltage of analog input port A (torque); ②The unit is mV;	-	-
0x1C6C	-	RO	The analog voltage of analog input port B	①This parameter indicates the actual input voltage of analog input port B (speed); ②The unit is mV;	-	-
0x1C6D	-	RO	Bus voltage	①This parameter indicates the current power supply board DC bus voltage; ②The unit is V ；	-	-
0x1C6F	-	RO	The current torque value	①Data format is _IQ7; ②The corresponding physical quantity of data is the actual Q-axis current of the motor,the unit is Amps;	-	-
0x1C70	-	RO	The current speed value	①Data format is _IQ2 ； ②The corresponding physical quantity of data is the current actual speed, the unit is rpm;	-	-
0x1C71	-	RO	The current high bit position value	①Data format is _IQ0, directly spliced by (H, L) to form a 32-bit position command; ②The corresponding physical quantity of data is the motor position, the unit is pulse;	-	-
0x1C72	-	RO	The current low bit position value	①Data format is _IQ0, directly spliced by (H, L) to form a 32-bit position command; ②The corresponding physical quantity of data is the motor position, the unit is pulse;	-	-
0x1C73	-	RO	Torque error value	①Data format is _IQ7 ； ②The corresponding physical quantities of data is the difference between torque command and actual torque (Q-axis current), the units is Amps;	-	-
0x1C74	-	RO	Speed error value	①Data format is _IQ2 ； ②The corresponding physical quantities of data is the difference between speed command and actual speed, the units is rpm;	-	-
0x1C75	-	RO	High bit position error value	①Data format is _IQ0, directly spliced by (H, L) to form a 32-bit position command; ②The corresponding physical quantities of data is the difference between high bit position command and actual high bit position, the units is pulse.	-	-
0x1C76	-	RO	Low bit position error value	①Data format is _IQ0, directly spliced by (H, L) to form a 32-bit position command; ②The corresponding physical quantities of data is the difference between low bit position command and actual low bit position, the units is pulse.	-	-
0x1C77	-	RO	Torque command	①Data format is _IQ7 ； ②The corresponding physical quantities of data is the Q-axis current command, the units is Amps.	-	-
0x1C78	-	RO	Speed command	①Data format is _IQ2 ； ②The corresponding physical quantities of data is the speed command, the units is rpm.	-	-
0x1C79	-	RO	High bit position command	①Data format is _IQ0, directly spliced by (H, L) to form a 32-bit position command; ②The corresponding physical quantities of data is the high bit position command, the units is rpm.	-	-
0x1C7A	-	RO	Low bit position command	①Data format is _IQ0, directly spliced by (H, L) to form a 32-bit position command; ②The corresponding physical quantities of data is the low bit position command, the units is rpm.	-	-

0x1C7B	-	RO	Alarm code	①This parameter indicates the current alarm code; ②When the value is 0, indicating there is no warning.	-	-
0x1C7C	-	RO	BootRom version	Boot Rom version number is stored in this parameter for application inquiry;	-	-
0x1C7D	-	RO	Software version	Software version number is stored in this parameter for application inquiry.	-	-

Title **BonMET CanOPEN interface specification**

Subject Description of functions exposed by the CanOPEN interface

Revisions

1.0	20 december 2010	E.Cavina	Initial revision
1.1	16 march 2011	E.Cavina	Release of canopen

Summary

1 Overview

This guide describes the CanOPEN interface of the servodrive including the comunication layer and the application profiles layer.

The comunication layer includes the following services:

- Object dictionary
- Node management (NMT)
- Service data objects (SDO)
- Process data objects (PDO)
- Emergency objects (EMCY)
- Synchronization objects (SYNC)

The application layer of the servodrive include the following profiles:

- Profile position (point-to-point movement, ramps generated inside servodrive)
- Profile velocity (speed control)
- Profile torque (torque control)
- Interpolated mode (cyclic target update, ramps generated in the motion controller)
- Homing mode (setup of position reference, search of minimum switch etc.)

1.1 Referenced standards

- CiA DS301 (revision 4.02) - standard comunication profile
- CiA DSP402 (revision 2.0) - servodrive application profile
- IEC61800-7: ETG Implementation Guideline for CiA402 Profile

2 DS301 - CANopen Comunication profile

2.1 NMT - Network management

Through NMT services the master can init, start, reset network nodes.

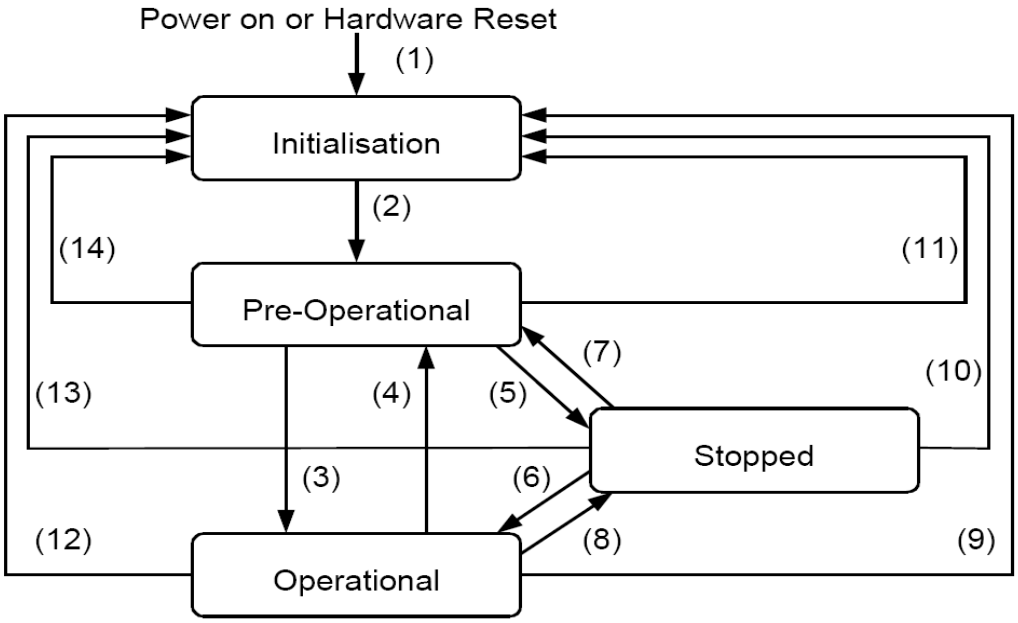
During network startup, each slave sends a broadcast boot-up message: in this way network master can check if network partners are properly connected and powered.

Furthermore, NMT permits runtime check of slave through one of the following services:

- Node-Guarding: periodically the master query for activity state each slave
- Hard-Beating: each slave publish its state periodically

This CanOPEN implementation supports NODE GUARDING.

NMT state machine



(1)	At Power on the initialisation state is entered autonomously
(2)	Initialisation finished - enter PRE-OPERATIONAL automatically
(3),(6)	Start_Remote_Node indication
(4),(7)	Enter_PRE-OPERATIONAL_State indication
(5),(8)	Stop_Remote_Node indication
(9),(10),(11)	Reset_Node indication
(12),(13),(14)	Reset_Communication indication

The following table summarize the services handled by CanOPEN layer according to NMT state:

	INITIALISING	PRE-OPERATIONAL	OPERATIONAL	STOPPED
PDO			X	
SDO		X	X	
Synchronisation Object		X	X	
Time Stamp Object		X	X	
Emergency Object		X	X	
Boot-Up Object	X			
Network Management Objects		X	X	X

Node guarding description

At startup (during pre-operational) NMT master has to setup the following object in each slave:

- guard time (100Ch): time between two subsequent queries
- life factor (100Dh): max number of guard time beyond which slave is declared down

NOTE: At startup, slave imlementation sets guard-time and life-factor to ZERO, meaning that node guarding will not be used.

When network goes operational, the slave expects a node guarding polling every guard-time; if the slave doesn't receive a node guarding polling within a time t=guard-time*life-factor, slave NMT status goes to STOPPED.

In the STOPPED state, application profile processing is inhibited; in this state master has to restart NMT network by sending a broadcast NMT reset command.

2.2 SDO - Service data object

SDO service permits access to object dictionary; in the following table are described SDO code returned by slave implementation in case of error:

SDO_ERR_TIMEOUT	05040000h	Master request timeout
SDO_ERR_CODE_UNKNOWN	05040001h	Generic error while processing SDO comand
SDO_ERR_UNSUPPORTED_ACCESS	06010000h	Access not supported (write over a read/only register etc.)
SDO_ERR_OD_NOT_EXISTS	06020000h	Object doesn't exists.
SDO_ERR_CANT_MAP_TO_PDO	06040041h	Unable to build PDO; slave emits this error if, during PDO configuration, master write PDO object number without writing all IDX/SUBIDX couples.
SDO_ERR_PDO_MAP_ERROR	06040042h	Unable ti build PDO; slave emits this error if, during PDO configuration, total size of mapped objects exceed PDO max length (es. canbus: 8byte)
SDO_ERR_WRONG_SIZE	06070010h	Emitted when the length of data to be read or written is different from object data length
SDO_ERR_WRONG_SUBINDEX	06090011h	Specified subindex doesn't exists
SDO_ERR_WRONG_VALUE	06090030h	When writing an object, specified value is out of range

In the current implementation every object has the maximum size of 32bit; for this reason slave implementation deals only with expedited SDO transfers.

2.3 PDO - Process data object

PDO permits data trasmission of several objects with low overhead; PDO emission can be:

- asynchronous: PDO is emitted when the value of one object changes
- synchronous: every N sync signals

The current implementation can use the following PDOs:

- CanOPEN over can-bus: n.4 Pdo RX + n.4 Pdo TX
- CanOPEN over EtherCAT: n.1 Pdo RX + n.1 Pdo RX

Default configuration of PDOs:

Pdo n.	Config. Object	Mapping Object	Default CobId	Mapped objects	Mapping meaning
0-RX	1400h	1600h	200h + NodeId	6040h	Control word
1-RX	1401h	1601h	300h + NodeId	6040h + 6060h	Control word + Mode of operation
2-RX	1402h	1602h	400h + NodeId	6040h + 607Ah	Control word + Target position
3-RX	1403h	1603h	500h + NodeId	6040h + 607Ah	Control word + Target position
0-TX	1800h	1A00h	180h + NodeId	6041h	Status word
1-TX	1801h	1A01h	280h + NodeId	6041h + 6061h	Status word + Mode of operation display
2-TX	1802h	1A02h	380h + NodeId	6041h + 6064h	Status word + Position actual value
3-TX	1803h	1A03h	480h + NodeId	6041h + 6064h	Status word + Position actual value

On slave startup, every PDO is de-activated: in the related configuration object CobID had MSB set (mask 8000_0000h).

Example:
NodeId=2
PdoRx1:
Configuration object: 1401h
SubIndex1: CobID
Default CobID value: 8000_0302h

PDO configuration must take the following steps:

1. Mapping definition
2. PDO activation

Mapping definition

On mapping configuration, master choose which data insert into PDOs; mapping definition must follow these steps:

- PDO deactivation: CobID MSB has to be set (see subindex 1 of configuration PDO object)
- number of mapped objects has to be zeroed (subindex 0 of PDO mapping object); this operation unlocks PDO configuration
- setup of data to be mapped in PDOs in the PDO mapping object (index/subindex and bit dimension)
- set of the number of mapped object in subindex 0 of PDO mapping configuration;

this operation locks PDO configuration

- eventual change of CobID keeping MSB set
- eventul change of trasmit-type anch inhibit-time in PDO TX config object
- activation of PDO emission/reception by clearing MSB in CobID (subindex 1 of PDO config object)

2.4 SYNC - Synchronization object

Master can synchronize data flow using the SYNC service:

- in CanOPEN over canbus, sync service means that master periodically emits a sync message on the bus
- in CanOPEN over EtherCAT sync service means that slave network interface generate a synchorization pulse using global time; during runtime master ensures that global time of the slaves in the network remains synchronized

The predefined COB-ID of SYNC message is 80h; cyclic comunication period is not implemented at comunication level; see interpolated mode to get more information about synchronous comunication period.

2.5 EMCY - Emergency object

With emergency objects, slave informs master that an application fault has occurred.

In servo-drive application profile DSP402, emergency are linked to application state machine (see Application profile description) and can be handled by the master at application level.

Emergency emission can be disabled by de-activating COB-ID in object 1014h.

Emergency messages are sent by slave once per application fault; slave implementation permits to specify inhibiton time between two subsequent emergency messages (object 1015h, value in hundreds of micro-seconds).

List of EMCY

Code [hex]	Meaning	Description
2220h	Overcurrent	During operation, current measures have exceeded allowrd range
2230h	Spm protection	During operation the inverter bridge has been deactivated by protection hardware
2380h	I2T motor protection	During operation, the motor overload threshold has exceeded allowed value
2280h	I2T spm protection	During operation, the power inverter overload threshold has exceeded allowed value
3110h	Overvoltage	During operation, the system has detected an overvoltage on the dc-link
3120h	Undervoltage	During operation, the system has detected an undervoltage on dclink
5210h	Autoset gain fail	During autoset operation, the system failed to autotune ADC current gains
5441h	Relay disconnection	During operation, internal precharge relais did not operate correctly
5530h	Storage alarm at startup	During startup the system has detected a problem on eeprom

5531h	Storage alarm on runtime	During operation, the system has detected a problem on eeprom on volatile memory
7110h	Brake chopper alarm	During operation, the brake overload threshold has exceeded the allowed value
7305h	Encoder index alarm	An error has been detected on encoder index pulse
7306h	Encoder hall alarm	An error has been detected on hall sensors in the position sensor
7307h	Encoder fault	An error has been detected on encoder incremental pulses
7320h	Position control alarm	Position control error has exceeded maximum parametrized position error
7810h	Limit switch fault	During operation the system has detected both limit switch pressed

EMCY message format

By default, EMCY message CobID is linked to Node ID:

COB-ID: 80h+NodeID; Ex. Nodeid=1 → COB-ID=81h

EMCY message data length is 8 byte: the meaning of data is the following:

D0	D1	D2	D3	D4	D5	D6	D7
EMCY Code		1001h Error register	0	0			

Byte ordering of data is little endian:

- D0=Lsb EMCY code
- D1=Msb EMCY code
- D2=Error register canopen
- D3=0
- D4=Lsb Info
- D5=Info
- D6=Info
- D7=Msb Info

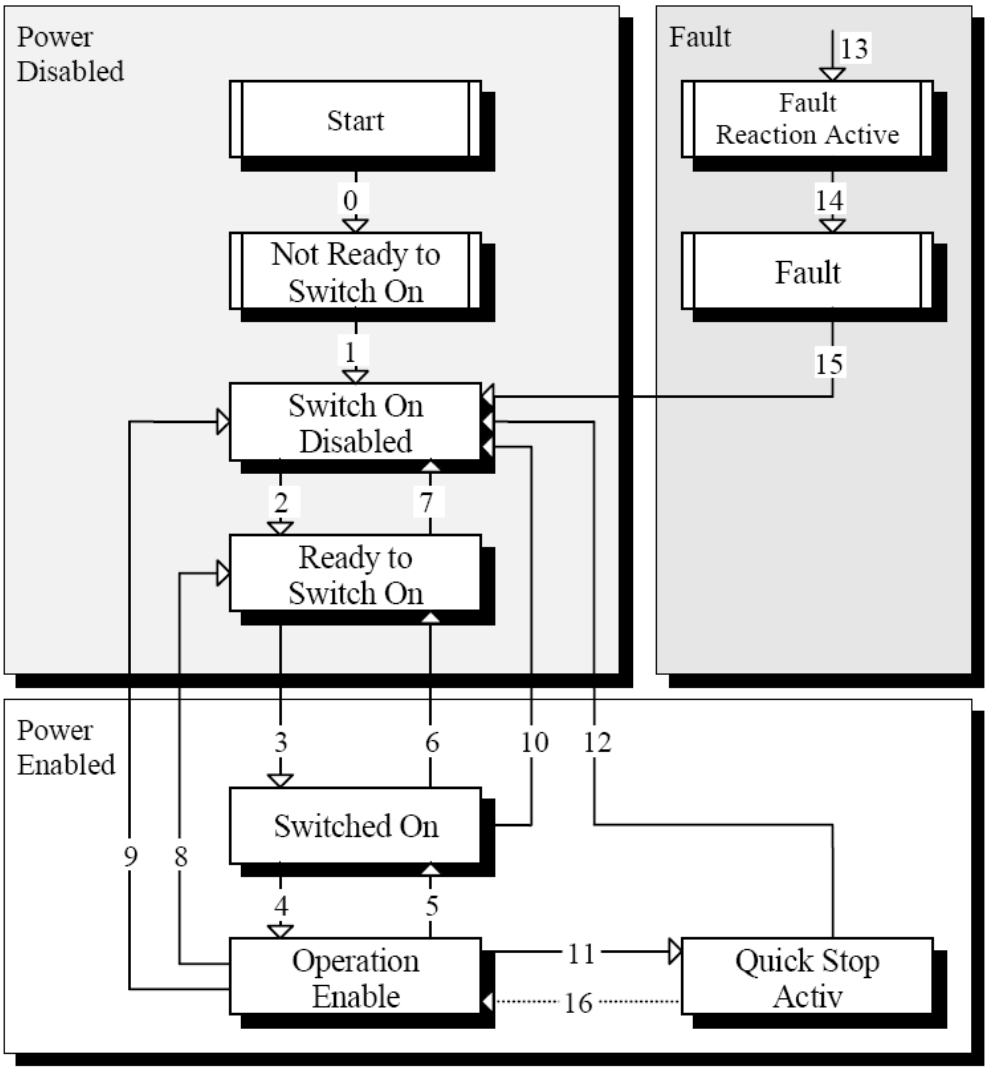
3 DSP402 - Servodrive application profile

3.1 Device control

This module defines functional operation of a CanOPEN servo drive.

Master controls the slave through a control word and slave gives feedback of actual state through a status word.

The following state machine shows how acts a Dsp402 compliant CanOPEN servodrive.



State description

StatusWord	Description
Not Ready to Switch On (StatusWord and 006Fh)=0h	• Power on self test • Brake applied if present • Servodrive disabled
Switch On Disabled (StatusWord and 006Fh)=40h	• Initialization completed • Parameter initialized and changeable • Servodrive disabled; mains shoudn't be applied
Ready to Switch On (StatusWord and 006Fh)=21h	• Parameter changeable • Servodrive disabled, but mains voltage can be applied

Switched On (StatusWord and 006Fh)=23h	• Parameter changeable • Mains activated • Servodrive disabled, but power stage ready
Operation Enable (StatusWord and 006Fh)=27h	• Servodrive is working • No fault present
Quick Stop Active (StatusWord and 006Fh)=07h	• Quick stop function activated • QuickStop-OptionCode select behaviour
Fault Reaction Active (StatusWord and 006Fh)=2Fh	• Parameter changeable • Servodrive has detected a manageble fault, i.e. motor can be safely stopped (FaultReaction-OptionCode select behaviour)
Fault (StatusWord and 006Fh)=2Fh	• Parameter changeable • Servodrive had detected a non-manageble fault (e.g. inverter short circuit) or fault reaction function has been completed

Transition description

Transition	Event	Action
0	Reset	PowerOn Self-Test started
1	PowerOn Self-Test ended	Runtime data process start
2	'Shutdown' received from master	-
3	'Switch On' received from master	-
4	'Enable Operation' received from master	Drive function gets activated
5	Disable Operation' received from master	Drive function gets disabled following DisableOperationOptionCode
6	Shutdown' received from master	Power stage gets deactivated
7	'Quick stop' received from master	-
8	'Shutdown' received from master	Drive function gets disabled following ShutdownOptionCode
9	'Disable Voltage' received from master	Drive function gets disabled
10	'Disable Voltage' o 'Quick Stop' received from master	Drive function gets disabled
11	'Quick Stop' received from master	Selected quick stop behaviour is executed
12	'Quick Stop' ended or 'Disable Voltage' received from master ; QuickStop-OptionCode is not “Stay in Quick-Stop”	Drive function gets disabled

13	Fault has been detected	If possible, selected fault reaction behaviour is been executed
14	Fault reaction handling completed	Drive function gets disabled
15	'Fault Reset' ricevuto da master	Fault gets cleared
16	'Enable Operation' ricevuto da master nelle opzioni “Stay in Quick-Stop”	Drive function gets disabled

Controlword

MSB								LSB							
Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
High-Byte								Low-Byte							

Bitfield description

Bit	Descrizione
0	Switch On
1	Disable Voltage
2	Quick Stop
3	Enable Operation
4	Operation Mode Specific
5	Operation Mode Specific
6	Operation Mode Specific
7	Reset Fault
8	Halt
9..15	Reserved (zero-filled)

Control-word command encoding:

Controlword Bit / Command	Bit 7 Fault Reset	Bit 3 Enable Operation	Bit 2 Quick Stop	Bit 1 Disable Voltage	Bit 0 Switch On	Transition
Shutdown	0	X	1	1	0	2,6,8
Switch On	0	X	1	1	1	3
Disable Voltage	0	X	X	0	X	7,9,10,12
Quick Stop	0	X	0	1	X	7,10,11
Disable Operation	0	0	1	1	1	5
Enable operation	0	1	1	1	1	4,16
Fault Reset	0→1	X	X	X	X	15

Control-word bit 4,5,6,8 meaning's depend of operation mode:

Bit	Profile position	Profile velocity	Profile torque	Homing	Ip-Mode
4	new_setpoint	reserved	reserved	Homing Operation Start	enable ip_mode
5	change_set immediatly	reserved	reserved	reserved	reserved
6	0: absolute 1: relative	reserved	reserved	reserved	reserved
8	Halt	Halt	Halt	Halt	Halt

Statusword

MSB								LSB							
Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
High-Byte								Low-Byte							

Bitfield description

Bit	Descrizione
0	Ready to Switch On
1	Switched On
2	Operation Enabled
3	Fault
4	Voltage Disabled
5	Quick Stop
6	Switch On Disabled
7	Warning
8	Manufacturer Specific
9	Remote
10	Target Reached
11	Internal Limit Active
12	Operation Mode Specific
13	Operation Mode Specific
14..15	Reserved (zero-filled)

Servodrive state encoding:

Statusword Bit / Stato	Bit 6 SwitchOn Disable	Bit 5 Quick Stop	Bit 3 Fault	Bit 2 Operation Enable	Bit 1 Switched On	Bit 0 Ready to Switch On
Not Ready to Switch On	0	X	0	0	0	0
Switch On Disabled	1	X	0	0	0	0
Ready to Switch On	0	1	0	0	0	1
Switched On	0	1	0	0	1	1
Operation Enabled	0	1	0	1	1	1
Fault	0	X	1	1	1	1
Fault Reaction Active	0	X	1	1	1	1
Quick Stop Active	0	0	0	1	1	1

X: not used

other configuration are prohibited

Status word bit 12,13 are operating mode dependent:

Bit	Profile position	Profile velocity	Profile torque	Homing	Ip-Mode
12	setpoint acknowledge	Speed = 0	reserved	Homing attained	Ip-Mode active
13	following error	max_slippage error	reserved	Homing error	reserved

Automatical drive stop:

DSP402 permits to define the servodrive behaviour in the transition from several states:

SHUTDOWN (Transition 2/6/8, to state SWITCH_ON_DISABLED): after a shutdown command (CONTROL_WORD=0) the following behaviour are possibile:

- ✓ 0: drive deactivation (motor shaft is free or braked, if brake present)
- ✓ 1: deceleration through value PROFILE_DECELERATION;on zero-speed, drive will be deactivated

Default: 0

DISABLE_OPERATION (Transition 5, OPERATION_ENABLED » SWITCHED_ON): on disable_operation command, the following behaviour are possible:

- ✓ 0: drive deactivation (motor shaft is free or braked, if brake present)
- ✓ +1: deceleration through value PROFILE_DECELERATION; on zero-speed, drive will be deactivated

Default: +1

QUICK_STOP (Transition 11 - OPERATION_ENABLED » QUICK_STOP): on quick_stop command, the following behaviour is possible:

- ✓ 0: drive deactivation (motor shaft is free or braked, if brake present)
- ✓ +1: deceleration through value PROFILE_DECELERATION; on zero-speed, transition to SWITCH_ON_DISABLED
- ✓ +2: deceleration through value QUICK_STOP_DECELERATION; on zero-speed, transition to SWITCH_ON_DISABLED
- ✓ +5: deceleration through value PROFILE_DECELERATION; on zero-speed, state remains QUICK_STOP
- ✓ +6: deceleration through value QUICK_STOP_DECELERATION; on zero-speed, state remains QUICK_STOP

Default: +2

FAULT_REACTION (Transition 13 - to FAULT_REACTION_ACTIVE): after detecting a fault that can be managed (controlled stop), it's possibile to define the following behaviour:

- ✓ 0: drive deactivation (motor shaft is free or braked, if brake present); stato changes immediatly to FAULT
- ✓ +1: deceleration through value PROFILE_DECELERATION; on zero-speed, transition to FAULT
- ✓ +2: deceleration through value QUICK_STOP_DECELERATION; on zero-speed transition to FAULT

Default: 0

If the servodrive detects a comunication problem (node-guarding timeout or busoff for can-bus, miss-link for EtherCAT), the drive will be stopped with the behaviour specified by DISABLE_OPERATION_OPTION_CODE (by default, slow_down with PROFILE_DECELERATION).

3.2 Factor group

Factor Group category contains object to scale servodrive position, speed and acceleration to physical units.

Servodrive implementation DO NOT implement factor group, since this work is left to motion master; only two object are implemented:

- encoder resolution
- encoder polarity

Position encoder resolution (608Fh)

This object expose servodrive encoder position resolution

- subindex 0: number of subindexes
- subindex 1: encoder increments
- subindex 2: motor revolution

Note: this object is READ-ONLY; encoder increments are fixed by type of motor/encoder,

motor revolution are fixed to 1.

Polarity (607Eh)

Using this object, master can set polarity of position and speed of servodrive.

- subindex 0 is a bitfield, see following table

Bit	Meaning
0...6	Not used
7	Position polarity 0 => positivo (default) 1 => negativo

3.3 Homing mode

Using homing mode, master can require to the drive to start with a position zero procedure; Dsp402 standard defined several type of homing: using index pulse, using switch etc.

To start homing master has to follow the procedure:

1. Set homing method (6098h)
2. Shutdown command (0006h) to control_word (6040h)
3. SwitchOn command (0007h) to control_word (6040h)
4. EnableOperation command (000Fh) to control_word (6040h)
5. StartHoming command (001Fh) to control_word (6040h)
6. ** Servodrive start homing procedure **
7. Wait bit “Homing Attained” (bit 12) in the status word (6041h)
8. If servodrive detect fault, it will set flag “Homing Error” (bit13) of status_word (6041h); master has to check EMCY messages or register 1003h to read fault cause

Supported homing method

Homing method can be selected through HOMING_METHOD register (6098h); Servodrive implementation support the following homing methods:

Homing method		Description
1	01h	Search for negative switch; then search for index in positive direction (*1)
2	02h	Search for positive switch; then search for index in negative direction (*1)
33	21h	Search for index in negative direction
34	22h	Search for index in positive direction
35	23h	Set quota in the current position

3.4 Profile position

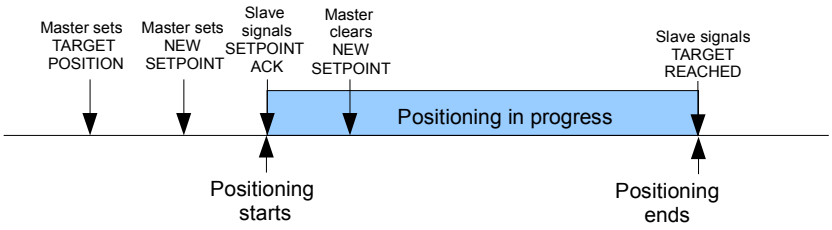
Profile position function allows position tracking using servodrive generated references: master define a new position target and servodrive create a smooth profile using defined cinematical costrains (velocity/acceleration).

Master/slave interaction during profile position follows the list below:

1. Master sets *target_position* (object 607Ah)

2. Master set *new_setpoint* flag (bit4 of the control_word, value changes from from 000Fh to 001Fh)
3. Master wait for *setpoint_ack flag* (bit12 of the status_word set to 1)
4. Master clears *new_setpoint flag* (bit4 of control_word, value changes from from 001Fh to 000Fh)
5. Slave computes position trajectory and position control
6. Upon completion, slave sets *target_reached* (bit10 of status_word)

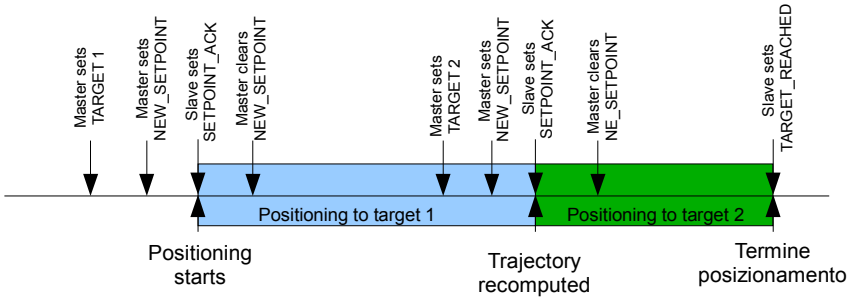
PROFILE POSITION SEQUENCE



On the fly setpoint change

Implementation permits to change target "on the fly": setpoint can be changed during position control, even if target is not yet reached; the following diagrams show the sequence for on the fly setpoint change.

ON THE FLY TARGET CHANGE



Halt

Using HALT flag (bit8 of control_word, mask 0100h) is possible to temporarily stop positioning movement, compatibly with cinematical costraints (max speed, max deceleration).

NOTE: when slave detect a rising edge over HALT flag, target_position register is not altered but clear the flag "setpoint accepted": to restart movement is necessary the following procedure:

1. Master clears of *halt* flag (bit8 of status_word)
2. Master sets *new_setpoint* flag (bit4 of control_word)
3. Master waits *setpoint_ack* flag (bit12 of status_word)
4. Master clears *new_setpoint* flag (bit 4 of control_word)

Software limits

Slave implementation can handle software position limit (object 607Dh); maximum limit and minimum limit are then used by slave upon reception of every new target position: if the target received is outside the given range, target will be rejected.

By default, slave define software limit to the numerical extreme of target_position register:

Min = 0x8000_0000 = -2147483648 pulses

Max = 0x7FFF_FFFF = +2147483647 pulses

Target reached

The specification defines a position window (object 6067h) and a window time (object 6068h) for target reached determination: when the difference between position actual value (object 6064h) and current position target becomes less that position window for a time greater than window time, status word flag "target reached" is set.

3.5 Profile velocity

Profile velocity permits speed control of the servodrive; to start speed control master has to follow the procedure:

- If servodrive has operation enabled, master disable servodrive by writing to control_word (6040h) value 0007h
- Master sets mode_of_operation (6060h) to profile velocity (value=3)
- Master enables operation setting control_word (6040h) to 000Fh
- Master modify target by writing *target_speed* (60FFh)

By default *target_speed* register (60FFh) is set to zero, so when slave enters OPERATION_ENABLED, speed target is zero.

Speed target is expressed in pulses per second; speed trajectory is generated observing cinematical costraints *profile_acceleration* (6083h) and *profile_deceleration* (6084h) registers.

Target reached

The specification defines a *velocity_window* (object 606Dh) and a window_time (object 606Eh) for target reached determination: when the difference between speed actual value (object 60FFh) and current speed target becomes less that speed window for a time greater than window time, status word flag "target reached" is set.

Zero speed

The specification defines a velocity_threshold (object 606Fh) and a velocity_threshold time (object 6070h) for zero speed determination: when the difference between speed actual value (object 60FFh) and 0 becomes less that threshold for a time greater than threshold time, status word flag "zero speed" is set.

3.6 Profile torque

Profile torque permits torque/current control of the servodrive; to start torque control master has to follow the procedure:

- If servodrive has operation enabled, master disable servodrive by writing to control_word (6040h) value 0007h
- Master sets mode_of_operation (6060h) to profile torque (value=4)
- Master enables operation setting control_word (6040h) to 000Fh
- Master modify target by writing *target_torque* (6071h)

By default *target_torque* register (6071h) is set to zero, so when slave enters OPERATION_ENABLED, torque target is zero.

Torque target is expressed in thousandths of nominal torque; torque trajectory is generated observing *torque_slope* (6087h) which is expressed in thousandths of nominal torque per second; torque slope permits to limit mechanical stress during torque control.

NOTE: during torque control, if motor shaft load is low respect target torque, shaft speed will accelerate until reaching nominal speed; master has to check if acceleration of shaft is dengerous for the application and eventually switch to a speed or position control.

3.7 Interpolated position mode

Interpolated position mode permits to control position of motor shaft following a trajectory generated by the master; in this mode, slave only interpolate between tho subsequent targets in a linear way.

Interpolated cycle time supported by servodrive are 1ms, 2ms, 4ms.

To enable interpolated position mode, master has to follow the procedure:

- If servodrive has operation enabled, master disable servodrive by writing to control_word (6040h) value 0007h
- Master sets mode_of_operation (6060h) to IP mode (value=7)
- Master enables operation setting control_word (6040h) to 000Fh
- Master enable interpolated mode setting control_word (6040h) to 001Fh
- Master writes periodically interpolation_data_record (60C1h) compling with SYNC signal

Note about interpolated position mode workings:

- during interpolated mode, master has to respect cinematical costraints of motor and load (maximum speed, acceleration and higher order derivatives)
- discontinuities of trajectory (both in value and due to delays) will generate bumps in motor mechanical load
- during interpolated mode, software position limits (object 607Dh) are ignored; master has to respect eventual mechanical bounds of axis. Hardware limit switch are always checked in interpolated mode

3.8 Digital I/O

Some digital input output are accessible via canopen registers, as specified in DSP402

Digital inputs: 60FDh

Contains CN2 input signals, in the following format:

Bit0	PulseInv
Bit1	SignInv
Bit2	INH
Bit3	CLE
Bit4	CWLtd
Bit5	CCWLtd
Bit6	CWDis
Bit7	CCWDis
Bit8	AlarmClr
Bit9	ServoEn

Digital outputs: 60FEh

This register allows to set some output that are treated as general purpose when servodrive is configured in CanOPEN; the format is the following:

Bit0	reserved
Bit1	COIN - GpOUT1
Bit2	Alarm - GpOUT2
Bit3	SRDY - GpOUT3

4 Object Dictionary

In the following table are described object dictionary indexes range

OBJECT RANGE		*DESCRIPTION*
1000h-1FFFh		**Communication Profile Area**
2000h-5EFFh		**Manufacturer Specific Profile Area**
	2240h - 2307h	canopen access to servodrive parameters
	2C35h - 2C53h	canopen access to special registers
6000h - 9FFFh		**DSP402 Profiled registers**
A000h - FFFFh		reserved

Following sections decribes register implemented in servodrive CanOPEN interface; until otherwise stated, object charateristics and format is specificed as in DS301 (comunication profile) and DSP402 (servodrive profile); please refer to the norms for further details.

4.1 DS301 - Comunication profile

Comunication

Index	Object		Name	Type	Acc.	U.M.	Default value
1000h	VAR		device type	UNSIGNED32	ro		00020192h
1001h	VAR		error register	UNSIGNED8	ro		0h
100Ch	VAR		guard time (*1)	UNSIGNED16	rw	ms	0
100Dh	VAR		life factor (*1)	UNSIGNED8	rw	unità	0
1014h	VAR		COB-ID EMCY (*1)	UNSIGNED32	rw		0080h | node-id
1015h	VAR		EMCY inhibit time (*1)	UNSIGNED16	rw	100us	500 (50ms)
1018h	RECORD		Identity Object	Identity (23h)	ro		-
		0	Number of sub-objects	UNSIGNED8	ro		4
		1	Vendor ID	UNSIGNED32	ro		424F4E4Dh
		2	Product Code	UNSIGNED32	ro		0 - TBD
		3	Revision	UNSIGNED32	ro		*Servodrive dependent*
		4	Serial number	UNSIGNED32	ro		*Servodrive dependent*

(*1) these registers are meaningful only when using CanOPEN over can-bus; in CanOPEN over Ethercat these registers are ignored.

Receive PDO Communication Parameter

Configuration objects for PDO RX (received from servodrive):

Index	Object	Name	Type
1400h	RECORD	1st receive PDO Parameter	PDO CommPar (20h)
1401h (*)	RECORD	2st receive PDO Parameter	PDO CommPar (20h)
1402h (*)	RECORD	3st receive PDO Parameter	PDO CommPar (20h)
1403h (*)	RECORD	4st receive PDO Parameter	PDO CommPar (20h)

(*) not present in CanOPEN-over-Ethercat

Configuration object PDO sub-indexes:

SubIdx	Object	Name	Type	Acc.
0	VAR	Number of sub-indexes	Unsigned8	ro
1	VAR	COB-ID	Unsigned32	rw
2	VAR	not used	Unsigned8	rw
3	VAR	not used	Unsigned16	rw

Receive PDO Mapping Parameter

Mapping objects for PDO RX (received from servodrive):

Index	Object	Name	Type
1600h	RECORD	1st receive PDO mapping	PdoMapping (21h)
1601h (*)	RECORD	2st receive PDO mapping	PdoMapping (21h)
1602h (*)	RECORD	3st receive PDO mapping	PdoMapping (21h)
1603h (*)	RECORD	4st receive PDO mapping	PdoMapping (21h)

(*) not present in CanOPEN-over-Ethercat

Transmit PDO Communication Parameter

Configuration objects for PDO TX (trasmitted from servodrive):

Index	Object	Name	Type
1800h	RECORD	1st transmit PDO Parameter	PDO CommPar (20h)
1801h (*)	RECORD	2st transmit PDO Parameter (*1)	PDO CommPar (20h)
1802h (*)	RECORD	3st transmit PDO Parameter (*1)	PDO CommPar (20h)
1803h (*)	RECORD	4st transmit PDO Parameter (*1)	PDO CommPar (20h)

(*) not present in CanOPEN-over-Ethercat

Configuration object for PDO TX sub-indexes:

SubIdx	Object	Name	Type	Acc.
0	VAR	Number of sub-indexes	Unsigned8	ro
1	VAR	COB-ID	Unsigned32	rw
2	VAR	Trasmit type (*1)	Unsigned8	rw
3	VAR	Inhibit time (espresso in multipli di 100microsec), valido se Trasmit type=255	Unsigned16	rw

(*1) Trasmit type has the following meaning:

0 = acyclic asynchronous (emitted only once after SYNC)

1 = cyclic every SYNC

2 = cyclic every 2 SYNC

...

240 = cyclic every 240 SYNC

255 = asynchronous (when object value changes, using inhibit time)

Using type 255, pdo is emitted when object changes: minimum time between PDO TX emission is the "inhibit time" (configuration object, subindex 3), in 100us units.

Inhibit time is useful when mapping, for example, position actual value in asynchronous

PDO TX: since the value may change continuosly, using inhibit user cant prevent band saturation.

NOTE: trasmit type selection is used only in canopen over can-bus; in CanOPEN over EtherCAT, PDO TX emission is always synchronous to master (trasmit type = 1).

Transmit PDO Mapping Parameter

Mapping objects for PDO TX (trasmitted from servodrive):

Index	Object	Name	Type
1A00h	RECORD	1st transmit PDO mapping	PdoMapping (21h)
1A01h (*)	RECORD	2st transmit PDO mapping (*1)	PdoMapping (21h)
1A02h (*)	RECORD	3st transmit PDO mapping (*1)	PdoMapping (21h)
1A03h (*)	RECORD	4st transmit PDO mapping (*1)	PdoMapping (21h)

(*) not present in CanOPEN-over-Ethercat

4.2 DSP402 - Servodrive profile

This section list all object implementated by servodrive implementation; for further information about object format, see DSP402 v2.0 - Servodrive application profile.

Device control

Index	Object		Name	Default	Type	Acc.	PDO
6040h	VAR		control_word	0	Unsigned16	rw	yes
6041h	VAR		status_word	250h	Unsigned16	ro	yes
605Ah	VAR		quick_stop_option_code 0=switch off on quickstop 1=profile_acceleration slow-down, go to SwitchOnDisabled 2=quick_stop_deceleration slow-down, go to SwitchOnDisabled 5=as 1, stay in Quick stop state 6=as 2, stay in Quick stop state	2	Integer16	rw	no
605Bh	VAR		shutdown_option_code 0=switch off on shutdown 1=deceleration ramp on shutdown	0	Integer16	rw	no
605Ch	VAR		disable_operation_option_code 0=switch off on disable_operation 1=deceleration ramp on disable_operation	1	Integer16	rw	no
605Eh	VAR		fault_reaction_option_code 0=switch off axis on fault 1=profile_deceleration slowdown on fault 2=quickstop_deceleration slowdown on fault	2	Integer16	rw	no
6060h	VAR		modes_of_operation 1=profile position 3=profile velocity 4=profile torque 6=homing 7=interpolated position	3	Integer8	wo	yes
6061h	VAR		modes_of_operation_display (same format as 6060h, servodrive image of actual operation mode)	3	Integer8	ro	yes
6085h	VAR		quick_stop_deceleration (inc/s^2)	(*1)	Unsigned32	rw	yes
60FD	VAR		digital inputs bitfield	(*2)	Unsigned32	ro	yes
60FE	ARRAY		digital outputs		Unsigned32	ro	yes
		0	number of sub-indexes	1	Unsigned8	ro	no
		1	digital output bitfield	(*3)	Unsigned32	rw	yes

(*1) Default quickstop deceleration: 30000rpm/s as inc/s^s (stop from 3000rpm in 100ms)

(*2), (*3) See section 3.8 for format of digital input/output bitfield

Factor group

Index	Object		Name	Default	Type	Acc.	PDO
608Fh	ARRAY		position_encoder_resolution		Unsigned32	rw	no
		0	number of sub-indexes	2	Unsigned 8	ro	no
		1	encoder increments	(*)	Unsigned32	ro	no
		2	shaft rounds	1	Unsigned32	ro	no
607Eh	VAR		polarity bit7=0: no polarity inversion (default) bit7=1: polarity inversion activated	0	Unsigned8	rw	no

(*) depends on motor selection code

Homing mode

Index	Object		Name	Default	Type	Acc.	PDO
607Ch	VAR		home_offset	0	Integer32	rw	no
6098h	VAR		homing_method (see Homing mode description)	1	Integer8	rw	yes
6099h	ARRAY		homing_speeds		Unsigned32	rw	no
		0	Number of subindexes	2	Unsigned8	ro	no
		1	Switch search speed	(*1)	Integer32	rw	no
		2	Index search speed	(*2)	Integer32	rw	no
609Ah	VAR		homing_acceleration	(*3)	Unsigned32	rw	yes

(*1) default homing switch search is 100rpm expressed as pulses/sec

(*2) default homing index search is 25rpm expressed as pulses/sec

(*3) default acceleration is 600rpm/s expressed as pulses/sec^2

Profile position mode

Index	Object		Name	Default	Type	Acc.	PDO
607Ah	VAR		target_position (increments)	0	Integer32	rw	yes
6081h	VAR		profile_velocity (inc/s)	(*1)	Unsigned32	rw	yes
6083h	VAR		profile_acceleration (inc/s^2)	(*2)	Unsigned32	rw	yes
6084h	VAR		profile_deceleration (inc/s^2)	(*3)	Unsigned32	rw	yes
607Dh	ARRAY		software position limit		Integer32	rw	yes
		0	number of subindexes	2	Unsigned8	ro	no
		1	Minimun position	80000000h (minimum negative integer)	Integer32	rw	no
		2	Maximal position	7FFFFFFFh (maximum positive integer)	Integer32	rw	no
6062h	VAR		position_demand_value (increments)	0	Integer32	ro	yes
6064h	VAR		position_actual_value (increments)	0	Integer32	ro	yes
6067h	VAR		position_window (increments)	3	Unsigned32	rw	yes

6068h	VAR	position_window_time (ms)	25	Unsigned16	rw	yes

(*1) default profile velocity is 95% of rated speed as inc/s

(*2), (*3) default profile acceleration/deceleration is 12000rpm/s, as inc/s^2

Interpolated position mode

Index	Object		Name	Default	Type	Acc.	PDO
60C1h	RECORD		interpolation_data_record			rw	yes
		0	sub-indexes number	1	Unsigned8	ro	no
		1	interpolation_data_record[1] ip mode setpoint (increments)	0	Integer32	rw	yes
60C2h	RECORD		interpolation_time_period			rw	yes
		0	sub-indexes number	2	Unsigned8	ro	no
		1	ip_time_units ticks of interpolation time	1	Unsigned8	rw	no
		2	ip_time_index tick value for interpolation time (*1)	-3	Signed8	rw	no
6062h	VAR		position_demand_value (increments)	0	Integer32	ro	yes
6064h	VAR		position_actual_value (increments)	0	Integer32	ro	yes
6067h	VAR		position_window (increments)	2	Unsigned32	rw	yes
6068h	VAR		position_window_time (ms)	25	Unsigned16	rw	yes

(*) ip_time_index is espressed as power of 10:

```
ip_time_index = -2 >> 0,01 (10ms)
ip_time_index = -3 >> 0,001 (1ms)
ip_time_index = -4 >> 0,0001 (100us)
```

Profile velocity

Index	Object	Name	Default	Type	Acc.	PDO
606Bh	VAR	velocity_demand_value (inc/s)	0	Integer32	ro	yes
606Ch	VAR	velocity_actual_value (inc/s)	0	Integer32	ro	yes
606Dh	VAR	velocity_window (inc/s) (target reached speed window)	80	Unsigned16	rw	yes
606Eh	VAR	velocity_window_time (ms)	25	Unsigned16	rw	yes
606Fh	VAR	velocity_threshold (inc/s) soglia per dichiarare velocita' nulla	80	Unsigned16	rw	yes
6070h	VAR	velocity_threshold_time (ms)	25	Unsigned16	rw	yes
6080h	VAR	max_motor_speed	(*1)	Unsigned32	ro	no
6083h	VAR	profile_acceleration (inc/s^2)	(*2)	Unsigned32	rw	yes
6084h	VAR	profile_deceleration (inc/s^2)	(*3)	Unsigned32	rw	yes
60FFh	VAR	target_velocity (inc/s)	0	Integer32	rw	yes

(*1) depends on motor type parameters, in inc/s

(*2), (*3) default profile acceleration/deceleration is 12000rpm/s, as inc/s^2

Profile torque

Index	Object	Name	Default	Type	Acc.	PDO
6071h	VAR	target_torque (thousandth of rated)	0	Integer16	rw	yes
6072h	VAR	max torque (thousands-th of rated)	(*1)	Integer16	ro	no
6074h	VAR	torque demand value (thousands-th of rated)	0	Integer16	ro	yes
6075h	VAR	motor rated current (milliamps)	(*2)	Unsigned32	ro	no
6077h	VAR	torque_actual_value (thousand-th of rated)	0	Integer16	rw	yes
6079h	VAR	DC_link_circuit_voltage (millivolts)	(*3)	Unsigned32	ro	no
6087h	VAR	torque_slope (thousandth of rated torque per second)	5000	Unsigned32	rw	no

(*1) depends on motor type parameters, in thousands of rated

(*2) depends on motor type parameters, in milliamperes

(*3) depends on dclink voltage