



Manuale

Field Buses Information

INDICE

Capitolo		
1	Descrizione Manuale	pagina 2
	Quick Reference Registri e parametri	pagina 3
2	Modbus RTU protocol	pagina 9
	Configurazione della rete	pagina 9
	Descrizione generale protocollo	pagina 10
	Data link layer	pagina 11
	Composizione del frame Modbus RTU	pagina 13
	Codici Funzione	pagina 15
	Estensione indirizzi per PLC Siemens	pagina 17
	Salvataggio su Flash per Parametri e Registri	pagina 17
	Eccezioni	pagina 18
	Esempi	pagina 19
	I/O Digitali	pagina 26
3	Codici di Allarme	pagina 29
4	CANopen Protocol	pagina 31
	Descrizione Generale	pagina 31
	Struttura fisica di una rete CANopen	pagina 32
	Object Dictionary (OD)& Electronic Data Sheet(EDS)	pagina 33
	Data Transfer via SDO & PDO	pagina 34
	Device Configuration via SDO	pagina 35
	Process Data Exange via PDO	pagina 39
	Emergency Message	pagina 42
	CANopen Network Management (NMT)	pagina 44
	Node guarding o Heartbeat	pagina 44
	Implementazioni nei drive	pagina 47
	CANopen Device Profile	pagina 48
	Object & Object Dictionary	pagina 49
	Accesso al Drive	pagina 50
	Parametri del Drive	pagina 50
	Procedura per la lettura e la scrittura dei parametri	pagina 51
	Salvataggio su Flash	pagina 51
	Configurazione della rete	pagina 52
	Oggetti CANopen	pagina 53
	Descrizione Oggetti Device Profile & Manufacturer specific	pagina 60
	Note sui servizi di GUARD , SDO , e PDO	pagina 64
	Esempi implementazione CANopen	pagina 65
	I/O Digitali	pagina 74
	Avvertenze	pagina 76
5	EtherCAT Protocol	
	Concetti generali	pagina 82
	Topologia	pagina 82
	DLL (Data Link Layer)	pagina 83
	Error detection	pagina 83

	Parametri e dati di processo	pagina 84
	Collegamenti: open & direct mode	pagina 84
	Struttura generale del frame	pagina 86
	Composizione frame EtherCAT	pagina 87
	Protocollo Mailbox	pagina 87
	FMMU	pagina 88
	File descrittore .XML	pagina 89
	Stati macchina	pagina 89
	EtherCAT	pagina 91
	Connessioni e Leds Drive EtherCAT	pagina 91
	Parametrizzazione EtherCAT per Drive	pagina 91
	PDO	pagina 92
	Utilizzo degli Slots	pagina 93
	SDO: tipologia e dizionario oggetti	pagina 94
	Mappatura dinamica	pagina 94
	Lettura / Scrittura parametri del drive	pagina 95
	Salvataggio in flash dei parametri	pagina 96
	Settaggi SM/DC per drive ECO	pagina 97
	Importazione file .xml con Twincat®	pagina 98
	Cambio configurazione pdo con Twincat®	pagina 99
	Codici allarmi specifici	pagina 101
	Reset allarmi	pagina 101
6	Profinet IO	
	Struttura generale	pagina 102
	Parametrizzazione Profinet IO per drive	pagina 103
	Mappatura memoria condivisa	pagina 104
	Esempio interfaccia con Master Profinet Hilscher	pagina 104
	Accesso asincrono a registri e parametri	Pagina 106
	Importazione file GSDML	Pagina 106
	Indirizzamento drive nella rete Profinet	Pagina 107
	Esempio posizionamento punto a punto	pagina 108
	Codici allarmi specifici	pagina 109
	Reset allarmi	pagina 109

Capitolo 1

Descrizione Manuale

Il presente manuale tratta nel dettaglio tutte le Reti di Campo (o Fieldbuses) implementate.

Verranno trattate di seguito :

Modbus RTU

CANopen Device profile DS 401

CANopen Device Profile DS 402

EtherCAT (CoE)

Profinet IO

Il controllo del Drive dalla Rete di campo si effettua attraverso la gestione di **Registri** e **Parametri**. I registri sono elementi a 32 Bit in cui vi sono parole di controllo ed informazioni da e per il Drive. I parametri sono elementi a 16 bit che tipicamente hanno influenza sulle regolazioni degli anelli e sulle modalità di funzionamento del Drive. E' indispensabile conoscere dettagliatamente il funzionamento del Drive e nello specifico i **Registri**, i **Parametri**, le **Modalità di funzionamento** e lo **Space Controller**, prima di approcciare la lettura di questo manuale . Queste caratteristiche sono illustrate nel manuale "*Additional Information*" nei capitoli 2, 3, 4, 5, 6.

Per comodità di seguito viene riportato un riassunto dei registri e dei parametri (**quick reference**) per veloce consultazione, da utilizzare quando si è compreso approfonditamente il loro funzionamento .

MANUALE REV. 4.0

Cod. Man. -----

Cod. catalogo.-----

08 2017



Quick reference Registri & Parametri

Registri

Registro	Significato	Save	contenuto	Indirizzo
R00	Control Word	si	0-0xffff	0x8000



07	06	05	04	03	02	01	00
Specifico del modo selezionato	Specifico del modo selezionato	Specifico del modo selezionato	Ltc_rst	Alm_rst	Stp_cmd	Ien_cmd	Ten_cmd

bit 0 **Ten_cmd** - comando abilitazione coppia 1: asse in coppia

bit 1 **Ien_cmd** - comando abilitazione movimentazioni 1: abilitazione movimentazioni

bit 2 **Stp_cmd** - comando stop

bit 3 **Alm_rst** - reset stato allarmi

bit 4 **Ltc_rst** - reset bit 4 di StatusWord (latch posizione)

bit 5 se Posizionatore: **Pos_str** - start posizionamento 0 → 1: start nuovo posizionamento

se Interpolatore: **Ip_str** - start interpolatore 1: start interpolatore

se Save/load: **Save** - salva registri e parametri 0 → 1: salvataggio registri e parametri su EEPROM

bit 6 se Posizionatore: **Upd_mod** - modo aggiornamento set-point

aggiornamento al volo

se Zero asse (homing): **Start/Stop** 0 → 1: start operazioni homing

se Asse elettrico: **Start/stop** 1: start, asse slave in asse col master

se Save/load: **Load** - carica registri e parametri 0 → 1: load registri e parametri da EEPROM

bit 7 Posizionatore: **Inc_mod** - incrementale/assoluto 1: set-point

incrementale

Registro	Significato	Save	contenuto	Indirizzo
R01	Modes of operation	si	0x81-0x7f	0x8001

1: modo posizionatore **3**: modo controllo di velocità **6**: modo zero asse (homing)

7: modo interpolatore **-1**: modo posizionatore impulsi **-2**: modo asse elettrico

-50: modo save/load **-127**: modo riferimento analogico

Registro	Significato	Save	contenuto	Indirizzo
R02	Feed Constant	si	0-0x3ffff	0x8002
R03	Following_error_windows	si	0-0xffffffff	0x8003
R04	Following_error_timeout	si	0- 0x1fff	0x8004

Registro	Significato	Save	contenuto	Indirizzo
R05	Status Word	no	0-0xff	0x8005



07	06	05	04	03	02	01	00
Specifico del modo selezionato	Specifico del modo selezionato	Specifico del modo selezionato	Ltc_st	Alm_st	Stp_st	Ien_st	Ten_st

- bit 0* **Ten_st** – stato abilitazione coppia 1: asse in coppia
bit 1 **Ien_st** – stato abilitazione movimentazioni 1: movimentazioni abilitate
bit 2 **Stp_st** – stato di stop 1: rampa di stop in corso
bit 3 **Alm_st** – stato allarmi 1: macchina in allarme
bit 4 **Ltc_st** – stato latch posizione 1: latch di posizione eseguito, registro pronto per la lettura
bit 5 se Posizionatore: **sp_rdy** – 1: pronto per un nuovo set-point
 se Interpolatore: **Ip_st** – Interpolatore 1: interpolatore attivo
 se Save/load: **Save** – salva registri e parametri 1: salvataggio completato
bit 6 se Posizionatore: **EOJ –End of Job** 1: fine del posizionamento
 se Zero asse (homing): **Homing ok** 1: homing completato
 se Asse elettrico: **Start/stop** 1: start, asse slave in asse col master
 se Save/load: **Load** - carica registri e parametri: 1: Caricamento terminato
bit 7 Posizionatore: **Following error** : 1: errore
 Controllo Velocità: **velocità nulla**: 1: asse fermo
 Zero Assi : **Homing error** 1: Errore durante l’homing

Registro	Significato	Save	contenuto	Indirizzo
R06	position_actual_value	no	0x80000000 X7fffffff	0x8006
R07	latched_position	no	0x80000000 X7fffffff	0x8007
R08	velocity_actual_value	no	0x80000000 X7fffffff	0x8008
R09	target_position	no	0x80000000 X7fffffff	0x8009
R10	profile_acceleration	sì	0x0 -X7fffffff	0x800A
R11	profile_deceleration	sì	0x0 -X7fffffff	0x800B
R12	profile_velocity	sì	0x0 -X7fffffff	0x800C

Registro	Significato	Save	contenuto	Indirizzo
R13	target_velocity	no	0x80000000 X7fffffff	0x800D
R14	velocity_window	si	0 -xffff	0x800E
R15	velocity_window_time	si	0 -x7fff	0x800F
R16	velocity_threshold	si	0 -xffff	0x8010
R17	velocity_threshold_time	si	0 -x7fff	0x8011
R18	homing_method	si	0x0 - ff	0x8012
R19	home_offset	si	0x80000000 X7fffffff	0x8013
R20	homing_fast_speed	si	0x80000000 X7fffffff	0x8014
R21	homing_slow_speed	si	0x0 -X7fffffff	0x8015
R22	homing_acceleration	si	0x0 -X7fffffff	0x8016
R23	velocity_master_sensor	no	0x8000 0x7fff	0x8017
R24	master_encoder_increments	si	0x0001 0x7fff	0x8018
R25	gear_ratio_numerator	si	0x01 - 0x20	0x8019
R26	gear_ratio_divisor	si	0x01 - 0xFF	0x801A
R27	motion_profile_type	si	0x8000 0x7fff	0x801B
R28	acceleration_jerk	si	0x01 - 0x63	0x801C
R29	deceleration_jerk	si	0x01 - 0x63	0x801D
R30	Aux in 1	si	0x80000000 X7fffffff	0x801E
R31	Aux in 2	si	0x80000000 X7fffffff	0x801F
R32	Aux out 1	si	0x80000000 X7fffffff	0x8020
R33	Aux out 2	si	0x80000000 X7fffffff	0x8021
R34	Aux in 3	si	0x80000000 X7fffffff	0x8022
R35	Torque max ---Object CANopen (0x3402)	si	0x8000 0x7fff	0x8023
R36	aux16_out_3	si	0x8000 0x7fff	0x8024
R37	aux16_out_4	si	0x8000 0x7fff	0x8025
R38	aux_in_5	si	0x80000000 X7fffffff	0x8026
R39	aux_in_6	si	0x80000000 X7fffffff	0x8027
R40	aux_in_7	si	0x80000000 X7fffffff	0x8028
R41	aux_in_8	si	0x80000000 X7fffffff	0x8029
R42	Aux16_in_9	si	0x80000000 X7fffffff	0x802A

Registro	Significato	Save	contenuto	Indirizzo
R43	aux16_in_10	si	0x8000 0x7fff	0x802B
R44	aux16_in_11	si	0x8000 0x7fff	0x802C
R45	aux16_in_12	si	0x8000 0x7fff	0x802D
R46	alarm_code	no	0 -xff	0x802E
R47	ip_time_units	sì	0x0 - ff	0x802f
R48	digital_inputs	no	0x0 XFFFFFFFF	0x8030
R49	digital_out_mask	sì	0x0 XFFFFFFFF	0x8031
R50	digital_outputs	sì	0x0 XFFFFFFFF	0x8032
R51	aux_in_13	sì	0x80000000 X7FFFFFFFF	0x8033
R52	aux_in_14	sì	0x80000000 X7FFFFFFFF	0x8034
R53	aux_in_15	sì	0x80000000 X7FFFFFFFF	0x8035
R54	aux_in_16	sì	0x80000000 X7FFFFFFFF	0x8036
R55	aux_in_17	sì	0x80000000 X7FFFFFFFF	0x8037
R56	aux_in_18	sì	0x80000000 X7FFFFFFFF	0x8038
R57	aux_in_19	sì	0x80000000 X7FFFFFFFF	0x8039
R58	aux_in_20	sì	0x80000000 X7FFFFFFFF	0x803A
R59	aux_in_21	sì	0x80000000 X7FFFFFFFF	0x803B
R60	aux_in_22	si	0x80000000 X7FFFFFFFF	0x803C
R61	target_window	si	0x80000000 X7FFFFFFFF	0x803D
R62	Abs_enc_control	si	0x80000000 X7FFFFFFFF	0x803E
R63	Abs_enc_status	si	0x80000000 X7FFFFFFFF	0x803F
R64	NumExtEnc	si	0x80000000 X7FFFFFFFF	0x8040
R65	DenExtEnc	si	0x80000000 X7FFFFFFFF	0x8041
R66- R72	Auxiliary register for Mechatronic Functions	si	0x80000000 X7FFFFFFFF	0x8042 0x8048
R73	Lettura velocità rpm	si	0x80000000 X7FFFFFFFF	0x8049
R74- R133	Auxiliary register for Mechatronic Functions	si	0x80000000 X7FFFFFFFF	0x804A 0x8085

Parametri

Parametro	Significato	contenuto	Indirizzo
d1	Guadagno proporzionale dell'anello di velocità, KP	% 1-99	01 (0x0001)
d2	Guadagno integrale dell'anello di velocità, KI .	% 1-99	02 (0x0002)
d3	Velocità massima del motore, Vmax	% 1-99	03 (0x0003)
d4	Regolazione offset di velocità.	% 1-99	04 (0x0004)
d5	Corrente massima del motore.	% 1-99	05 (0x0005)
d6	Rapporto <i>corrente di picco/corrente nominale</i> .	% 1-99	06 (0x0006)
d7	Funzione I²T .	% 1-99	07 (0x0007)
d8	Selezione motore.	Numero	08 (0x0008)
E1	Velocità nominale massima.	% 1-99	09 (0x0009)
E2	Filtro sul riferimento di entrata.	% 1-99	10 (0x000A)
E3	Parametro Eccezioni.	Numero	58 (0x003A)
E4	Riservato	Numero	55 (0x0037)
F1	Rampa di massima accelerazione.	% 1-99	11 (0x000B)
F2	Rampa di massima decelerazione.	% 1-99	12 (0x000C)
F3	Rampa di emergenza.	% 1-99	13 (0x000D)
F4	Riservato	Numero	
c1	Selezione impulsi uscita RTE.	Numero	14 (0x000E)
c2	Verso rotazione motore	Numero	15 (0x000F)
c3	Selezione tipo di controllo (velocità/coppia).	Numero	16 (0x0010)
c4	Selezione coppie polari Resolver.	Numero	17 (0x0011)
c5	Selezione impulsi giro Encoder.	Numero	18 (0x0012)
c6	Selezione tipo di trasduttore	Numero	19 (0x0013)
c7	Numero nodo della rete.	Numero	20 (0x0014)
c8	NON UTILIZZATO.	--	21 (0x0015)
c9	Selezione modalità di funzionamento.	Numero	22 (0x0016)
I1	Guadagno proporzionale dell'anello di spazio.	% 1-99	35 (0x0023)
I2	Guadagno integrale dell'anello di spazio.	% 1-99	36 (0x0024)
I3	Guadagno derivativo dell'anello di spazio.	% 1-99	37 (0x0025)
I4	Valore di compensazione inerzia.	% 1-99	38 (0x0026)
I5	Timeout su errore di Posizione (in centesimi di secondo).	Numero	39 (0x0027)
I6	Selezione modalità encoder esterno	Numero	90 (0x005A)
I7	Riservato	Numero	99 (0x0063)
O1	Corrente istantanea.	Numero	81 (0x0051)
O2	Corrente efficace.	Numero	82 (0x0052)
O3	Sovraccarico (% di I ² T) .	Numero	83 (0x0053)
O4	Riservato	Numero	

Parametro	Significato	contenuto	Indirizzo
P1	Impulsi/giro albero motore -parte più significativa).	Numero	41 (0x0029)
P2	Impulsi/giro albero motore(parte meno significativa	Numero	42 (0x002A)
P3	Massimo errore di inseguimento (gradi) (parte più significativa)	Numero	43 (0x002B)
P4	Massimo errore di inseguimento (gradi) (parte meno significativa).	Numero	44 (0x002C)
P5	Rampa accelerazione (in ms/100 RPM).	Numero	45 (0x002D)
P6	Rampa decelerazione (in ms/100 RPM).	Numero	46 (0x002E)
P7	Velocità massima (in RPM/100).	Numero	47 (0x002F)

A1	Tacche Encoder Master hi (parte più significativa).	Numero	48 (0x0030)
A2	Tacche Encoder Master lo (parte meno significativa).	Numero	49 (0x0031)
A3	Numeratore Rapporto di Riduzione.	Numero	50 (0x0032)
A4	Denominatore Rapporto di Riduzione.	Numero	51 (0x0033)
A5	Errore inseguimento massimo (in gradi) hi (parte più significativa).	Gradi	52 (0x0034)
A6	Errore inseguimento massimo (in gradi) lo (parte meno significativa).	Gradi	53 (0x0035)

H1	Metodo per effettuare lo zero assi.	Numero	59 (0x003B)
H2	Velocità di ricerca switch in % della velocità nominale.	% 1-99	60 (0x003C)
H3	Velocità di ricerca Marker in millesimi della velocità nominale.	% 1-99	61 (0x003D)
H4	Tempo di rampa di accelerazione/decelerazione (in ms/100 RPM).	Numero	62 (0x003E)
H5	Segno offset di spazio.	Numero	63 (0x003F)
H6	Offset di spazio in impulsi encoder master (hi), decine di migliaia.	Numero	64 (0x0040)
H7	Offset di spazio in impulsi encoder master (mid), centinaia.	Numero	65 (0x0041)
H8	Offset di spazio in impulsi encoder master (lo), unità.	Numero	66 (0x0042)
H9	Utility per la taratura manuale dell'Offset di spazio.	Numero	67 (0x0043)

n1	CANbus Baud rate.	Numero	40 (0x0028)
-----------	-------------------	--------	-------------

Capitolo 2



Modbus RTU protocol

Configurazione della rete

Per scegliere la rete MODBUS RTU si può usare il tastierino del drive selezionando **C9 = 3**.

Il **numero del nodo** corrisponde al parametro **C7** e può assumere ogni valore tra 1 e 90 (valore di default 1). **Il numero nodo è letto solamente all'accensione del drive** e quindi un'eventuale modifica viene attuata solamente alla successiva accensione.

La **parità** della porta di comunicazione seriale è impostato con il parametro **e3**: se e3 = 6 non viene impostata nessuna parità, altrimenti viene usata la parità pari.

Lettura e scrittura registri e parametri

L'accesso ai registri e ai parametri del drive avviene attraverso i comandi di lettura e scrittura parametri illustrati nei capitoli dedicati alla descrizione del protocollo di comunicazione MODBUS. L'indirizzo specificato secondo la tabella illustrata nel paragrafo Quick Reference Registri & Parametri.

Descrizione Generale Protocollo

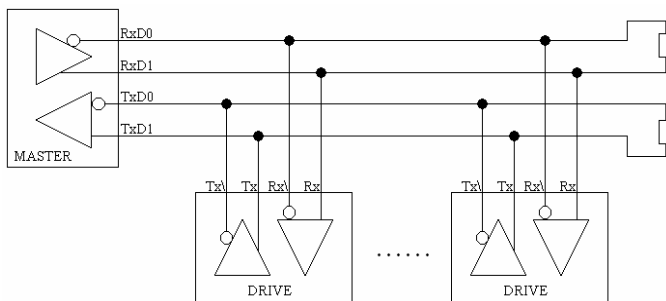
Il protocollo di comunicazione MODBUS RTU (di seguito MODBUS) implementato sui drive Eco è conforme allo standard, pur presentando alcune limitazioni, quali un ridotto numero di nodi disponibili e un ridotto, ma pur sempre completo, set di comandi. Ogni particolarità verrà comunque chiarita in questo capitolo.

Relativamente all'indirizzamento dei registri è necessario precisare che, al contrario dei registri MODBUS, quelli del drive hanno gli indirizzi allineati: ciò significa che l'indirizzo da inserire nel messaggio è effettivamente quello del registro.

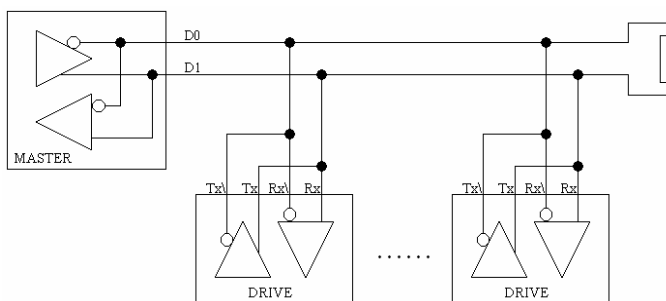
La comunicazione seriale che sfrutta il protocollo MODBUS comprende 3 livelli:

- MODBUS Application Layer: corrisponde al livello 7 (Application) del modello di riferimento ISO/OSI ed implementa una relazione di tipo client/server.
- MODBUS Serial line Protocol: corrisponde al livello 2 (Data Link) del modello di riferimento ISO/OSI ed implementa una relazione di tipo master/slave
- MODBUS Physical Layer

Secondo la convenienza, determinata dal tipo di drive utilizzato, si può decidere di utilizzare una tipologia RS485 o RS422.



Nella rete RS422 (Four Wire) i dati sul paio master (RxD1 e RxD0) sono in ricezione agli slaves, e i dati nel paio slave (TxD1 e TxD0) sono in ricezione al master. In ogni istante un solo drive slave ha il permesso di trasmettere.



Nella rete RS485 (Two Wire) in ogni istante un solo drive ha il permesso di trasmettere.

MODBUS Serial line Protocol (**data link layer**)

Il MODBUS Serial Line Protocol realizza un servizio di tipo master/slave. Al bus sono connessi, in ogni istante, un solo master e uno o più drive slave (1...90). Ogni comunicazione viene iniziata dal master, che può intraprendere una sola transazione alla volta. Gli slave possono trasmettere solo in risposta ad una richiesta del master e non prendono mai l'iniziativa per iniziare una transazione; i nodi slave non comunicano mai tra di loro.

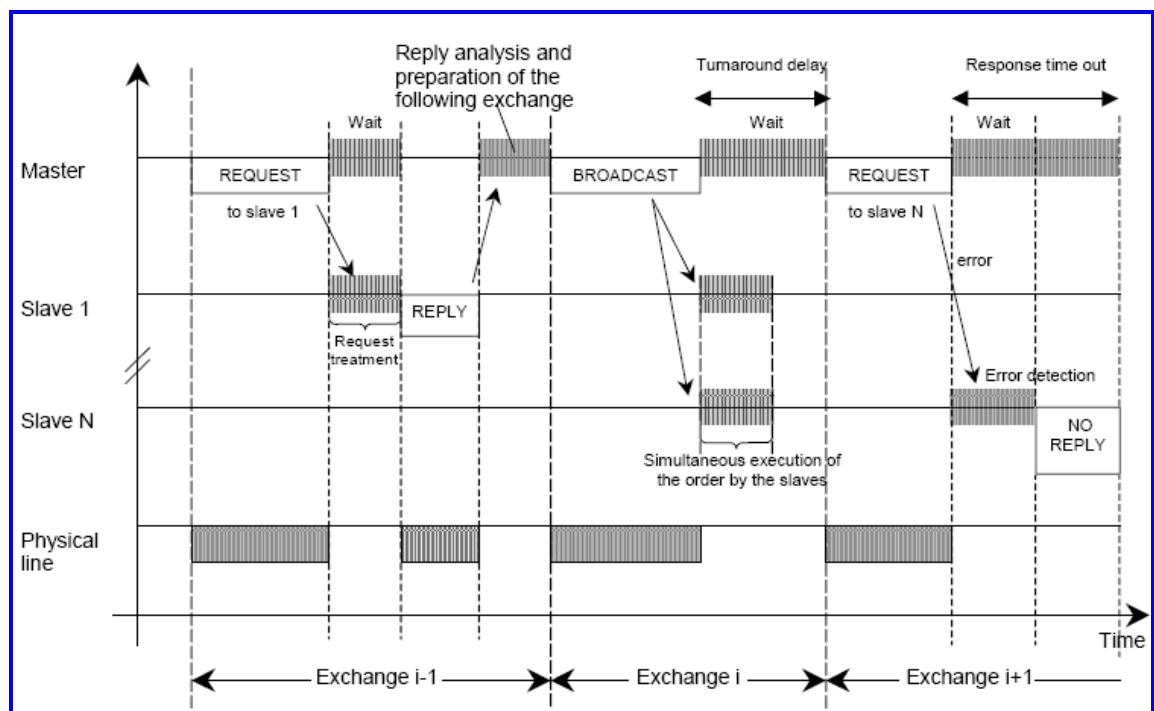
Il master può comunicare con i nodi slave in due modi:

- **Modo UNICAST:** il messaggio è inviato ad un solo nodo, indicandone all'interno l'indirizzo. Il drive slave, dopo la ricezione del messaggio, processa la richiesta e risponde al master. Ogni nodo slave deve avere un indirizzo che lo identifica in modo univoco in modo da poter essere indirizzato indipendentemente dagli altri nodi.
- **Modo BROADCAST:** il messaggio viene inviato a tutti i nodi della rete. L'indirizzo specificato nel messaggio deve essere 0. Ogni drive che riceve un messaggio broadcast è costretto processarlo. Non è richiesta una risposta degli slave al master.

Regole di trasmissione

Protocollo Master/Slave

Esempio di diagramma temporale di comunicazione master/slave



Master

All'accensione il master si trova nello stato **idle** (*libero*) da cui può inviare i messaggi.

Dopo l'invio di una richiesta, il master lascia questo stato, poiché è possibile iniziare una sola transazione alla volta. Se è stata inviata una richiesta unicast, il master si sposta nello stato **waiting for reply** e viene attivato il **response timeout**: se non la risposta non arriva entro il timeout viene generato un errore.

Quando viene ricevuta la risposta, il master la controlla e, se non ci sono errori, la processa. Se invece c'è un errore, può essere eseguita una ritrasmissione, il cui numero massimo dipende dal settaggio del master.

Terminate tutte le operazioni relative alla risposta il master torna nello stato **idle**, ed è pronto per iniziare una nuova transazione.

Quando viene generata una richiesta broadcast, che non richiede risposta, il master si sposta nello stato **waiting turnaround delay** in cui aspetta per un tempo determinato (**turnaround delay**) in modo da permettere a tutti i nodi di ricevere il messaggio. Passato tale tempo, il master ritorna nello stato **idle** ed è pronto per iniziare una nuova transazione.

Il **response timeout** per la richiesta unicast deve essere abbastanza lungo da permettere al nodo di ricevere e processare il messaggio e di inviare la risposta al master. Se la richiesta è broadcast il **turnaround delay** deve essere abbastanza lungo da permettere a tutti i nodi di ricevere e processare il messaggio. Quindi il **response timeout** è più lungo del **turnaround delay**, infatti, tipicamente, sono rispettivamente di 1s (fino a qualche secondo) e di 100-200 ms.

Slave

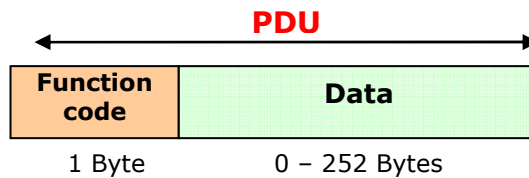
All'accensione ogni slave entra nello stato **idle** in cui attende le richieste.

Se lo slave riscontra un errore di comunicazione il messaggio viene scartato e non viene generata alcuna risposta. Se invece il messaggio arriva senza errori di trasmissione, viene prima controllato, in modo da individuare errori (indirizzi illegali, valori non permessi, ...). In caso di errore deve essere inviata una risposta al master, usando i **function code** relativi alle eccezioni per indicare il tipo di errore e permettere al master di determinare l'azione da intraprendere.

Se invece il messaggio è corretto, la richiesta viene processata e, se è **unicast**, il nodo slave provvede a confezionare ed a inviare la risposta per il master.

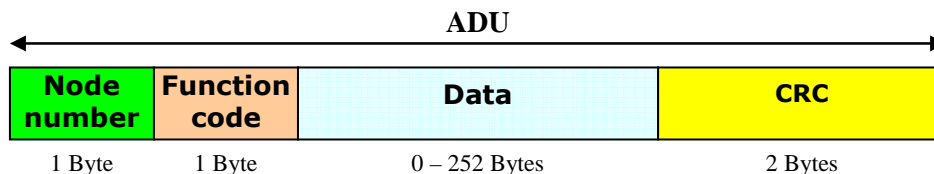
Composizione del frame MODBUS RTU

Il **MODBUS Application Layer** definisce una semplice unità chiamata **PDU (Protocol Data Unit)** indipendente dai livelli di comunicazione sottostanti. Ogni PDU è composto di due campi:



- **Function code:** indica l'operazione da eseguire. Questo campo è lungo 1 byte e quindi il suo valore è compreso tra 1 e 255. I codici fra 128 e 255 sono riservati per le risposte di eccezione; il valore 0 non è valido. La risposta dello slave riporta il **Function code** della relativa richiesta; nelle risposte di eccezione lo slave riporta il **Function code** originario in cui però il bit più significativo viene settato a 1.
- **Data:** contiene i parametri per l'esecuzione dell'istruzione specificata nel **Function code**. In alcuni tipi di messaggio il campo dati può non esistere (lunghezza 0); in questi casi il **Function code** è sufficiente per svolgere la funzione. Nel caso di risposta d'eccezione in questo campo viene indicato il codice dell'errore verificatosi.

Per spedire il PDU sul bus occorre aggiungere le informazioni necessarie affinché il destinatario riceva il messaggio e possa determinare gli eventuali errori di trasmissione. Il dispositivo trasmettente compone così un appropriato frame chiamato **Application Data Unit (ADU)**. La massima lunghezza che può avere un frame è di 256 byte.

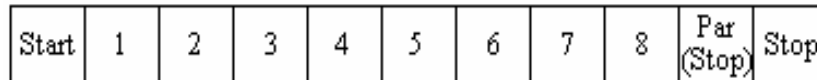


- **Node number:** contiene il numero di nodo del drive slave destinatario (in decimale 0..90). Il master invia una richiesta ad uno slave indicandone il numero di nodo in questo campo. Quando lo slave invia la risposta pone il proprio numero di nodo in questo campo, in modo che il master sappia di chi è la risposta. Gli indirizzi sono così distribuiti:
 - L'indirizzo 0 indica che il messaggio è broadcast
 - Gli indirizzi da 1 a 90 sono assegnati agli slave
 - Gli indirizzi da 91 a 255 sono riservati
 - Il master non ha indirizzo
- **CRC:** contiene il risultato del calcolo di ridondanza per il rilevamento di eventuali errori di trasmissione

- RTU transmission mode

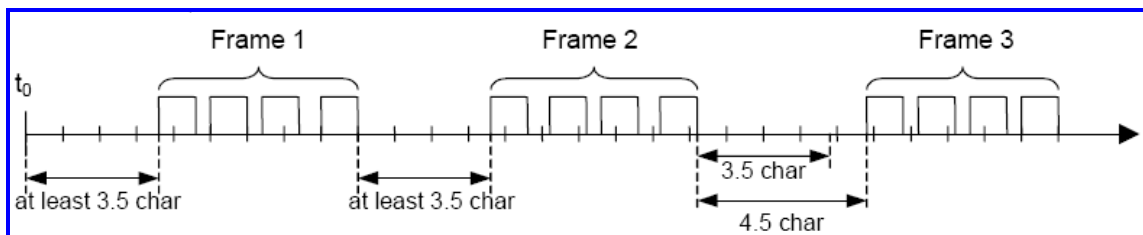
Quando un dispositivo comunica usando la trasmissione RTU (Remote Terminal Unit), ogni frame viene frammentato in bytes (caratteri). Si noti che ogni byte è scomponibile in 2 caratteri esadecimali da 4 bit ognuno: il vantaggio di questa modalità è, infatti, l'alta densità di caratteri che permettono di avere un alto throughput di dati.

Ogni byte (carattere) viene spedito separatamente partendo dal bit meno significativo, che viene preceduto da un segnale di start. Terminata la trasmissione del byte viene inserito un bit di parità, se previsto altrimenti viene inserito un segnale di stop seguito da un altro segnale di stop.



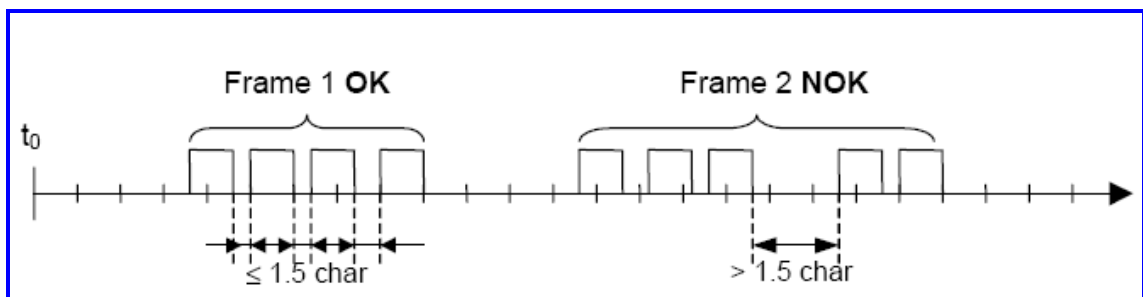
La trasmissione di ogni frame deve essere preceduta da un tempo di silenzio della durata di almeno 3,5 caratteri. Se questo tempo non viene rispettato, il dispositivo ricevente può considerare il messaggio come continuazione del precedente generando un errore.

Al termine della trasmissione deve essere mantenuto un tempo di silenzio di almeno 3,5 caratteri, trascorso il quale è possibile iniziare la trasmissione del frame successivo.



L'intero frame deve essere trasmesso con un continuo stream di caratteri. Se fra la trasmissione di due byte trascorre un intervallo temporale superiore a 1,5 caratteri, il messaggio viene dichiarato incompleto e viene scartato dal ricevitore: non viene generata dunque nessuna risposta per il master.

Metodi di rilevamento degli errori



Per ogni carattere spedito può essere effettuato il controllo della parità (parity check even or odd), mentre la verifica del CRC deve essere effettuata alla fine della ricezione del frame. Entrambi questi metodi di rilevamento dell'errore sono calcolati ed inseriti nel messaggio dal dispositivo che trasmette, mentre quello ricevente testa ogni carattere e il frame durante la ricezione. Il master deve aspettare un certo tempo in modo che lo slave possa controllare e attuare il messaggio e provvedere a

inviare la risposta. Se lo slave rileva un errore di trasmissione, il messaggio viene scartato e non viene inviata la risposta: il *response timeout* finisce e quindi il master sa che deve gestire un errore di trasmissione.

Codici funzione

I registri dell'azionamento sono tutti implementati a 32 bit, mentre i parametri sono a 16 bit: ogni registro viene quindi considerato come due registri a 16 bit indirizzati consecutivamente ed in cui quello con l'indirizzo più basso contiene i 16 bit meno significativi del registro a 32 bit.

I registri a 32 bit possono essere intesi come se fossero a 16 bit: in questo caso la lettura e la scrittura coinvolgono solo i 16 bit meno significativi di tale registro

L'indirizzo che il PLC deve inviare al drive per leggere o scrivere un determinato parametro o registro deve eventualmente essere aumentato di 1, a seconda dell'implementazione adottata per la realizzazione del protocollo MODBUS.

Le funzioni implementate sul drive sono:

- codice 03: read holding register
 - lettura parametro (16 bit)
 - lettura registro (32 bit)
 - lettura registro (16 bit meno significativi)
 - codice 06: write single register
 - scrittura parametro
 - scrittura registro (16 bit meno significativi)
 - codice 16: write multiple registers
 - scrittura registro (32 bit)
 - exception response
- protocollo MODBUS.

Codice 03 (0x03): Read Holding Registers

Con questo codice si richiede al drive slave la lettura del contenuto di un certo numero di registri che compongono un blocco continuo. Nel PDU di richiesta di lettura si specifica l'indirizzo del primo registro da leggere e il numero totale dei registri contigui da leggere.

Bisogna tenere presente che i parametri del drive sono effettivamente considerati come registri a 16 bit, mentre i registri a 32 bit sono considerati come due contigui da 16 bit.

Se un registro a 32 bit viene letto indicando **Quantity of registers = 1**, la risposta contiene il valore dei 16 bit meno significativi del registro. Se invece s'intende leggere l'intero registro a 32 bit, occorre impostare **Quantity of registers = 2**: nella risposta il suo valore viene scritto in una coppia di word, di cui la prima contiene la parte alta del registro, mentre la seconda contiene i bit meno significativi.

Con questa funzione non è ammesso il broadcast.

Error code associato è 0x83.

REQUEST:	Function code	0x03	1 byte
	Starting address	0x0000 to 0xFFFF	2 bytes
	Quantity of registers	1 to 125 (0x7D)	2 bytes
RESPONSE:	Function code	0x03	1 byte
	Byte count	2*N	1 byte
	Register value	0x0000 to 0xFFFF	2*N bytes

Codice 06 (0x06): Write Single Register

Con questo codice si esegue la scrittura del registro specificato in un dispositivo remoto. Nel PDU di richiesta di lettura si specifica l'indirizzo del registro da scrivere e il valore da assegnare. Questo codice si usa per scrivere i valori dei parametri, che sono contenuti in registri da 16 bit.

Se un registro a 32 bit viene scritto con questo comando viene trattato come se fosse un registro a 16 bit: nella parte bassa viene scritto il valore desiderato, mentre nei 16 bit più significativi viene riportata l'estensione del segno.

Quando il comando è usato in broadcast, si richiede la scrittura dello stesso parametro in tutti i drive della rete.

La risposta al comando (non broadcast) è l'indirizzo e il valore del parametro.

L'*Error code* associato è 0x86.

REQUEST:	Function code	0x06	1 byte
	Register address	0x0000 to 0xFFFF	2 bytes
	Register value	0x0000 to 0xFFFF	2 bytes
RESPONSE:	Function code	0x06	1 byte
	Register address	0x0000 to 0xFFFF	2 bytes
	Register value	0x0000 to 0xFFFF	2 bytes

Codice 16 (0x10): Write Multiple Registers

Questa funzione è usata per scrivere un blocco continuo di registri in un dispositivo remoto. Nella nostra implementazione il campo *Byte count* può contenere solo i valori 1 o 2: è pertanto possibile scrivere al massimo due registri contigui. In pratica questo comando è usato per scrivere un parametro se il numero di byte è 1 oppure per scrivere un registro da 32 bit (considerato come due a 16 bit contigui) se il numero di byte viene posto a 2.

Nella richiesta il valore da scrivere nei registri viene diviso in due parti: per ogni registro la prima word contiene i bit di ordine alto, mentre la seconda contiene i bit di ordine basso.

La risposta descrive quanti registri sono stati scritti e da quale indirizzo si è partiti.

Quando il comando è usato in broadcast, si richiede la scrittura dello stesso registro in tutti i drive della rete.

L'*Error code* associato è 0x90.

REQUEST:	Function code	0x10	1 byte
	Starting address	0x0000 to 0xFFFF	2 bytes
	Quantity of registers	1 or 2	2 bytes
	Byte count	2*N	1 byte
	Registers value	0x0000 to 0xFFFF	2*N bytes
RESPONSE:	Function code	0x10	1 byte
	Starting address	0x0000 to 0xFFFF	2 bytes
	Quantity of registers	1 or 2	2 bytes

Estensione della mappatura degli indirizzi per PLC Siemens

Normalmente la mappatura degli indirizzi dei parametri e dei registri è la stessa utilizzata per il protocollo di comunicazione S-Net. I parametri si trovano negli indirizzi da 0x0001, mentre i registri partono dall'indirizzo 0x8000.

Per particolari esigenze (PLC Siemens), è possibile impostare una diversa mappatura degli indirizzi, impostando il parametro del drive E3 = 9.

Con questa impostazione i parametri sono mappati a partire dall'indirizzo 40001, mentre i registri sono mappati a partire dall'indirizzo 44096.

L'indirizzo che il PLC deve inviare al drive per leggere o scrivere un determinato parametro o registro deve eventualmente essere aumentato di 1, a seconda dell'implementazione adottata per la realizzazione del

Salvataggio su flash di parametri e registri

Per il salvataggio permanente delle modifiche apportate a parametri e registri è necessario il salvataggio sulla memoria flash del drive.

Sono implementate due procedure per quest'operazione: la prima prevede di porre il **modes_of_operation al valore -50** e successivamente di portare la **ControlWord a 0x20**.

La seconda consente il salvataggio scrivendo il valore **0x0011** all'indirizzo **0x3802** (che eventualmente va aumentato di 1 a seconda dell'implementazione adottata per la realizzazione del protocollo MODBUS sul PLC) utilizzando dunque la funzione 0x06 (write single register). Per comodità dell'utilizzatore è consentito anche l'accesso a 32 bit, impiegando conseguentemente la funzione 0x10 (write multiple registers) di MODBUS-RTU. Se, infatti, l'utente ha sviluppato le proprie procedure MODBUS-RTU per accedere ai registri a 32 bit del drive, può utilizzarle anche per comandare il salvataggio su flash dei parametri e registri del drive.

Il drive risponde immediatamente al comando di salvataggio e di seguito esegue la memorizzazione vera e propria. Durante il salvataggio su flash (operazione della durata di qualche secondo) il drive non risponde ad eventuali comunicazioni su MODBUS-RTU.

Per entrambe le procedure viene fornito un esempio nel capitolo dedicato.

Importante: il salvataggio va effettuato ad asse non in coppia.

Eccezioni

Quando un client manda una richiesta al server si aspetta una risposta normale.

Sono quattro le situazioni possibili:

- Se il server riceve la richiesta senza errori di comunicazione e la può processare normalmente, risponde normalmente al client.
- Se il server non riceve la richiesta per un errore di comunicazione, non viene inviata alcuna risposta. Il client ha dei meccanismi di timeout per gestire questi eventi.
- Se il server riceve la richiesta, ma trova errori di comunicazione (parità, CRC, ...) non viene inviata nessuna risposta.
- Se il server riceve la richiesta senza errori di comunicazione, ma non è in grado di compiere l'azione (ad esempio leggere un registro inesistente) viene generata una risposta di eccezione in cui si riporta il tipo di errore occorso.

La risposta di eccezione ha due campi, per distinguerla dalla risposta normale che ne ha più di due.

- *Function Code Field*: nella risposta normale il server risponde con lo stesso codice della richiesta. Tutti i codici hanno il bit più significativo (MSB) a zero. In una risposta d'eccezione il server setta il MSB del codice a 1 in modo che venga sommato 0x80 al codice della risposta normale. Ad esempio se il messaggio che genera un errore ha 0x03 come codice funzione, la risposta di eccezione conterrà il codice funzione 0x83.
- *Data Field*: viene inviato il codice dell'eccezione che è stata generata.

Nel drive è implementato solo il codice d'errore *02 Illegal Data Address*. In tutti gli altri casi di errore il messaggio viene scartato e quindi non viene inviata alcuna risposta al master.



Esempi

L'indirizzo che il PLC deve inviare al drive per leggere o scrivere un determinato parametro o registro deve eventualmente essere aumentato di 1, a seconda dell'implementazione adottata per la realizzazione del protocollo MODBUS.

Codice 0x03: lettura dei parametri e dei registri

Field name		Value (Hex)
Node number		05
Function Code		03
Data	Starting Address Hi	00
	Starting Address Lo	01
	Number of Register Hi	00
	Number of Register Lo	03
CRC		hi
		lo

Si desidera leggere il valore dei parametri d1, d2, d3 del drive al nodo 5.

Si invia al drive il seguente messaggio:
05 03 00 01 00 03 crc high i crc Low

Function code 0x03
Starting address 0x0001
Quantity of registers 0x0003

Field name		Value (Hex)
Node number		05
Function Code		03
Data	Byte count	06
	Register Value Hi	00
	Register Value Lo	14
	Register Value Hi	00
	Register Value Lo	32
	Register Value Hi	00
	Register Value Lo	32
CRC		hi
		lo

La risposta che viene inviata è la seguente:

05 03 06 00 14 00 32 00 32 Chi Clo
Function code 0x03
Byte count 0x06
Register value
 0x0014
 0x0032
 0x0032

Da cui si legge che i valori dei parametri sono: d1 = 0x0014 = 20
d2 = 0x0032 = 50
d3 = 0x0032 = 50

Con lo stesso codice funzione è possibile leggere i registri, che sono variabili a 32 bit

Field name		Value (Hex)
Node number		05
Function Code		03
Data	Starting Address Hi	80
	Starting Address Lo	00
	Number of Register Hi	00
	Number of Register Lo	04
CRC		hi
		lo

Ad esempio si desidera leggere il valore dei registri R00 e R01 del drive al nodo 5.

Si invia al drive il seguente messaggio:
05 03 00 01 00 03 Chi Clo

Function code 0x03
Starting address 0x8000
Quantity of registers 0x0004
(2 registri a 32 bit = 4 registri a 16 bit)

Field name		Value (Hex)
Node number		05
Function Code		03
Data	Byte count	04
	Register Value Hi	11
	Register Value Lo	22
	Register Value Hi	33
	Register Value Lo	44
CRC		hi
		lo

La risposta che viene inviata è la seguente:

05 03 04 11 22 33 44 CHi CLo
 Function code 0x03
 Byte count 0x04
 Register value 0x1122
 0x3344

Nella lettura dei registri prima viene letta la parte alta del registro e poi la parte bassa del registro, perciò i valori letti sono: R00 = 0x1122 = 4386
 R01 = 0x3344 = 1312

Codice 0x06: scrittura di un parametro

Field name		Value (Hex)
Node number		05
Function Code		06
Data	Register Address Hi	00
	Register Address Lo	03
	Register Value Hi	00
	Register Value Lo	0A
CRC		hi
		lo

Si invia al drive il seguente messaggio:

05 06 00 03 00 0A CHi CLo
 Function code 0x06
 Register address 0x0003
 Register value 0x000A

Si desidera porre a 10 il valore Del parametro d3 del drive al nodo 05.

Field name		Value (Hex)
Node number		05
Function Code		06
Data	Register Address Hi	00
	Register Address Lo	03
	Register Value Hi	00
	Register Value Lo	0A
CRC		hi
		lo

La risposta che viene inviata è la seguente:

05 06 00 03 00 0A CHi CLo
 Function code 0x06
 Register address 0x0003
 Register value 0x000A

Da cui si deduce che il parametro di indirizzo 0x0003 è stato scritto col valore 0x000A.

Codice 0x10: scrittura di un registro

Si desidera porre a 1000 = 0x03E8 il registro 09 *target_position* del drive al nodo 05. Tale registro ha indirizzo 0x8009 ed è a 32 bit.

Si invia al drive il seguente messaggio:
05 10 80 09 00 02 04 00 00 03 E8 Chi Clo

Function code:	0x10
Starting address	0x8009
Quantity of registers	0x0002
Byte count	0x04
Registers value	0x03E8

Field name		Value (Hex)
Node number		05
Function Code		10
Data	Starting Address Hi	80
	Starting Address Lo	09
	Quantity of registers Hi	00
	Quantity of registers Lo	02
CRC		hi lo

La risposta che viene inviata è la seguente:

05 10 80 09 00 02 CHi CLo

Function code	0x10
Starting address	0x8009
Quantity of registers	0x0002

Da cui si deduce che è stato scritto il registro di indirizzo 0x8009.



Esempio di configurazione dell'asse in modalità posizionario

In quest'esempio si vuole far funzionare l'asse al nodo 01 come posizionario. Si configureranno quindi i seguenti registri:

Nome Registro	Indirizzo	Valore	Comando scrittura MODBUS RTU
<i>feed_constant</i>	0x8002	0x003C	01 10 80 02 00 02 04 00 00 00 3C Chi Clo
<i>profile_acceleration</i>	0x800A	0x0064	01 10 80 0A 00 02 04 00 00 00 64 Chi Clo
<i>profile_deceleration</i>	0x800B	0x0064	01 10 80 0B 00 02 04 00 00 00 64 Chi Clo
<i>profile_velocity</i>	0x800C	0x03E8	01 10 80 0C 00 02 04 00 00 03 E8 Chi Clo
<i>motion_profile_type</i>	0x801B	0x0000	01 10 80 1B 00 02 04 00 00 00 00 Chi Clo
<i>homing_method</i>	0x8012	0x0021	01 10 80 12 00 02 04 00 00 00 21 Chi Clo
<i>home_offset</i>	0x8013	0x0000	01 10 80 13 00 02 04 00 00 00 00 Chi Clo
<i>homing_slow_speed</i>	0x8015	0x000A	01 10 80 15 00 02 04 00 00 00 0A Chi Clo
<i>homing_acceleration</i>	0x8016	0x0064	01 10 80 16 00 02 04 00 00 00 64 Chi Clo

I parametri sono quelli di default, fatta eccezione per c9 che deve valere 3 e il parametro d8 che deve essere impostato conformemente al motore utilizzato.

Si fornisce quindi la 24 V_{dc} sugli ingressi "TEN" e "IEN".

Si resettano quindi eventuali allarmi presenti scrivendo nel registro **0x8000 (ControlWord)** il **valore 8**:
01 10 80 00 00 02 04 00 00 00 08 Chi Clo

Per fare **l'homing** si imposta il registro **modes_of_operation (0x8001)** a **6**
01 10 80 01 00 02 04 00 00 00 06 Chi Clo

si abilita e si mette in coppia l'asse scrivendo nel registro **0x8000 (ControlWord)** il valore **3**:
01 10 80 00 00 02 04 00 00 00 03 Chi Clo

comando di start: scrivere nel registro **0x8000 (ControlWord)** il valore **0x0043**:
01 10 80 00 00 02 04 00 00 00 43 Chi Clo

Per togliere coppia al motore 01 10 80 00 00 02 04 00 00 00 00 Chi Clo

Si seleziona, ora, **il modo posizionario** ponendo il registro **0x8001** a **1**
01 10 80 01 00 02 04 00 00 00 01 Chi Clo

si sceglie quindi

la posizione target e la si scrive nel registro **0x8009**

01 10 80 09 00 02 04 00 00 17 70 CHi CLo

e si **mette in coppia il motore**

01 10 80 00 00 02 04 00 00 00 03 CHi CLo

Si dà lo **start** portando la **ControlWord** a **0x0023**

01 10 80 00 00 02 04 00 00 00 23 CHi CLo

Da notare che quelle inviate sono tutte scritture , pertanto il drive risponde ad ognuna indicando l'indirizzo di inizio (che coincide con il numero del registro) e il numero di registri scritti (che è sempre 2).

Salvataggio su flash

Primo metodo di salvataggio

Per il salvataggio permanente delle modifiche apportate a parametri e registri è necessario il salvataggio sulla memoria flash del drive. Poiché il salvataggio in flash può richiedere alcuni secondi, la risposta, che viene inviata al termine dell'operazione, giungerà con un certo ritardo, tipicamente inferiore ai 5 secondi.

Occorre quindi porre il **modes_of_operation**, registro **01 (0x8001)**, a **-50:**

01 10 80 01 00 02 04 FF FF FF CE CHi CLo

E portare la **ControlWord** (registro **00 = 0x8000**) a **0x20:**

01 10 80 00 00 02 04 00 00 00 20 CHi CLo

Poiché quelle inviate sono tutte scritture il drive risponde ad ognuna indicando l'indirizzo di inizio (che coincide con il numero del registro) e il numero di registri scritti (che è sempre 2).

Secondo metodo di salvataggio con accesso a 16 bit

Field name		Value (Hex)
Node number		05
Function Code		06
Data	Register Address Hi	38
	Register Address Lo	02
	Register Value Hi	00
	Register Value Lo	11
CRC		Hi
		Lo

Si fa riferimento al nodo 5.

Si invia al drive il seguente messaggio:

05 06 38 02 00 11 CHi CLo

Function code 0x06
Register address 0x3802
Register value
 0x0011

Field name		Value (Hex)
Node number		05
Function Code		06
Data	Register Address Hi	38
	Register Address Lo	02
	Register Value Hi	00
	Register Value Lo	11
CRC		Hi
		Lo

La risposta che viene inviata dal drive è la seguente:

5 06 38 02 00 11 CHi CLo

Function code 0x06
Register address 0x3802
Register value
 0x0011

Da cui si deduce che il comando di salvataggio è giunto al drive e verrà processato immediatamente.

Secondo metodo di salvataggio con accesso a 32 bit

Field name		Value (Hex)
Node number		05
Function Code		10
Data	Starting Address Hi	38
	Starting Address Lo	02
	Quantity of registers Hi	00
	Quantity of registers Lo	02
	Byte count	04
	Registers value Hi	00
	Registers value Lo	00
	Registers value Hi	00
Registers value Lo	11	
CRC		Hi
		Lo

Si fa riferimento al nodo 5.

Si invia al drive il seguente messaggio:

05 10 38 02 00 02 04 00 00 00 11
CHi CLo

Function code 0x10
Starting address 0x3802
Quantity of registers 0x0002
Byte count 0x04
Registers value 0x00000011



Esempi di utilizzo dei codici funzione con diversa mappatura degli indirizzi per PLC Siemens

Codice 0x03: lettura dei parametri e dei registri

Si desidera leggere il valore dei parametri d1, d2, d3 del drive al nodo 5. L'indirizzo di partenza è quello del parametro d1, che nella nuova mappatura corrisponde a $40001 + 1 = 40002 = 0x9C42$

Più in dettaglio: Function code 0x03
 Starting address 0x9C42
 Quantità di registri 0x0003

Con lo stesso codice funzione è possibile leggere i registri, che sono variabili a 32 bit. Ad esempio si desidera leggere il valore dei registri R00 e R01 del drive al nodo 5. L'indirizzo di partenza è quello del parametro Registro 00, che nella nuova mappatura corrisponde a $44097 + 0 = 0xAC41$

Più in dettaglio: Function code 0x03
 Starting address 0xAC41
 Quantità di registri 0x0004 (2 registri a 32 bit = 4 registri a 16 bit)

Nella lettura dei registri prima viene letta la parte alta del registro e poi la parte bassa del registro a 32 bit.

Codice 0x06: scrittura di un parametro

Si desidera porre a 10 il valore del parametro d3 del drive al nodo 05. L'indirizzo di partenza è quello del parametro d1, che nella nuova mappatura corrisponde a $40001 + 3 = 40004 = 0x9C44$

Più in dettaglio: Function code 0x06
 Register address 0x9C44
 Register value 0x000A

Codice 0x10: scrittura di un registro

Si desidera porre a 1000 = 0x03E8 il registro 09 *target_position* del drive al nodo 05. Il registro è all'indirizzo $44097 + 9 = 0xAC4A$

Più in dettaglio: Function code: 0x10
 Starting address 0xAC4A
 Quantity of registers 0x0002
 Byte count 0x04
 Registers value 0x03E8



I/O digitali

In questo capitolo s'illustrano gli I/O (Input / Output digitali).

La morsettiera CN2 è composta di 16 connessioni (pin) delle quali 12 sono configurate come ingressi e 4 come uscite. Tutti i segnali presenti nella morsettiera lavorano con livelli di tensione 0 - 24 V_{dc}, con lo zero in comune con l'alimentazione 24 V_{dc} del drive. In altre parole, per portare alto un ingresso, occorre fornire +24 V_{dc} al pin 2 di CN1 (zero alimentazione). Analogamente, per le uscite (di tipo PNP), un livello alto significa che il segnale viene portato a +24 V_{dc} al pin 2 di CN1 (zero alimentazione).

Ad alcuni pin del connettore a 16 poli della morsettiera CN2 corrisponde una funzione con un significato preciso, come mostrato nella tabella seguente.

Morsettiera CN2 ECO2/ECO4

Pin number	modo	Funzione
1	In	Start
2	In	
3	In	
4	In	
5	In	Select
6	In	
7	In	
8	In	Negative limit switch
9	In	Positive limit switch
10	In	
11	In	Position latch
12	Out	End of Job
13	Out	
14	In	Home switch
15	Out	Near toTarget
16	Out	Brake

si riserva il diritto di utilizzare senza preavviso, gli ingressi e le uscite digitali attualmente non impiegate, per l'implementazione di nuove funzioni nelle release successive.

L'ingresso "Start" in modo posizionario viene utilizzato per dare il via al posizionamento. Il via viene dato in corrispondenza del fronte di salita del segnale. Il via al posizionamento può anche essere comandato via software ponendo a 1 il bit 5 di *ControlWord*; il bit ritorna automaticamente a zero.

L'ingresso "Start" in modo asse elettrico viene utilizzato per la messa in asse. Questa avviene se il segnale è a livello alto ed il bit 5 di *ControlWord* è a 1. Negli altri casi l'asse rimane bloccato (fermo in coppia).

L'ingresso "Select" in modo posizionario viene utilizzato per definire la quota da raggiungere. In particolare, se l'ingresso è zero o aperto, la quota da raggiungere col posizionamento è quella specificata nel registro 09 (*target_position*). Se invece l'ingresso è a livello alto, la quota utilizzata per il posizionamento è quella specificata nel registro 30 (*aux_in_1*).

L'ingresso "Select" in modo controllo di velocità viene utilizzato per definire la velocità da raggiungere. In particolare, se l'ingresso è a zero o aperto, la velocità impostata è quella specificata nel registro 13 (*target_velocity*). Se invece l'ingresso è a livello alto, la velocità da raggiungere è quella specificata nel registro 30 (*aux_in_1*).

"Negative limit switch", "Positive limit switch" e "Home switch" sono gli ingressi relativi agli switch utilizzati per la ricerca dello zero assi secondo il metodo di ricerca utilizzato (*homing_method*).

L'ingresso "Position latch" è utilizzato per catturare la posizione dell'asse (espressa in unità utente) in corrispondenza del fronte di salita del segnale. La posizione così memorizzata può essere acquisita attraverso un'operazione di lettura dell'apposito registro. Un sistema di filtraggio software evita che disturbi di breve durata diano luogo a latching imprevisti. Per un corretto funzionamento del sistema, è quindi necessario che l'ingresso digitale rimanga alto per almeno 2 ms. Ogni evento di cattura viene segnalato attraverso un bit del registro *StatusWord*.

L'uscita "End of Job" viene attivata se il target impostato (posizione, velocità, ecc.) è stato raggiunto.

Più in dettaglio:

- Posizionatore (*modes_of_operation* = 1): allo start del posizionamento quest'uscita viene abbassata. Quando il posizionamento è terminato e l'errore di inseguimento è inferiore a quello specificato nella relativa finestra (*following_error_window*), l'uscita "End of Job" viene alzata.
- Controllo di velocità (*modes_of_operation* = 3): al raggiungimento della velocità di target del generatore di traiettoria, l'uscita "End of Job" viene alzata.
- Homing (*modes_of_operation* = 6): allo start della ricerca zero l'uscita "End of Job" viene abbassata. Quando la ricerca di zero è terminata con successo l'uscita viene alzata.
- Posizionatore a impulsi (*modes_of_operation* = -1): il comportamento è del tutto simile a quello relativo al posizionatore. In questo caso, ogni impulso che giunge al drive va considerato come un nuovo start.
- Asse elettrico (*modes_of_operation* = -2): l'uscita "End of Job" è sempre bassa.
- Riferimento analogico (*modes_of_operation* = -127): l'uscita "End of Job" è sempre bassa.

L'uscita "Target" viene utilizzata con il posizionatore punto a punto (*modes_of_operation* = 1) e anche in modo posizionatore SAP (Stand Alone Positioning system). Allo start del posizionamento quest'uscita viene abbassata. Non appena la posizione dell'asse si viene a trovare ad una distanza inferiore a quella specificata nel registro *target_window* l'uscita viene alzata. Questa funzione è attiva a partire dalla release 2.40

L'uscita "Brake" pilota il freno in alcuni modelli di azionamento.

Attraverso il registro *digital_inputs* è possibile leggere lo stato degli ingressi. Ad ogni pin di CN2 è, infatti, associato un bit del detto registro, secondo quanto indicato nella tabella sottostante. In alcuni casi, per uno stesso ingresso sono indicate due funzioni. Ciò significa che qualora non si utilizzi in nessun caso quella primaria, si può liberamente utilizzare l'ingresso come generico digital input. Se ad esempio, per le operazioni di homing viene utilizzato solo l'*home switch* (pin 14), gli ingressi relativi ai limit switch positivo (pin 9) e negativo (pin 8) possono essere impiegati come ingressi digitali.

digital inputs

Pin number	digital_inputs bit number	Funzione
1	1	Start / Digital in
2	2	Digital in
3	3	Digital in
4	4	Digital in
5	5	Select / Digital In
6	6	Digital in
7	7	Digital in
8	8	Negative limit switch / Digital in
9	9	Positive limit switch / Digital in
10	11	Digital in
11	0	Position latch / Digital in
14	10	Home switch / Digital in

digital_inputs su MiniECO standard o miniECO **PLUS**

Nel miniECO sono disponibili soltanto alcuni ingressi ma anch'essi possono essere letti come input a cui è già stata assegnata la funzione specifica. Ad esempio se non si usano i limit switches del drive e quindi la procedura di zero assi non viene svolta dal Drive si possono utilizzare questi ingressi e leggerli tramite bus di campo. **E' necessario settare il C8 nella configurazione desiderata** vedi tabella sotto

Param. C8	CN1 Pin Name	Numero Bit digital_inputs	Funzione
0	11 -- DIR	8	Negative Limit Switch / Digital In
	10 -- PULSE	9	Positive Limit Switch / Digital In
	5 -- IEN	10	HomeSwitch / Digital In
1	11 -- DIR	8	Negative Limit Switch / Digital In
	10 -- PULSE	9	Positive Limit Switch / Digital In
	5 -- IEN	0	Position Latch / Digital In
2	11 -- DIR	8	Negative Limit Switch / Digital In
	10 -- PULSE	9	Positive Limit Switch / Digital In
	5 -- IEN	1	Start Move/ Digital In
3	11 -- DIR	1	Start Move/ Digital In
	10 -- PULSE	10	HomeSwitch / Digital In
	5 -- IEN	0	Position Latch / Digital In
4	11 -- DIR	1	Start Move/ Digital In
	10 -- PULSE	8	Negative Limit Switch / Digital In
	5 -- IEN	--	Input Enable
5	11 -- DIR	1	Start Move/ Digital In
	10 -- PULSE	9	Positive Limit Switch / Digital In
	5 -- IEN	--	Input Enable
6 default	11 -- DIR	1	Start Move/ Digital In
	10 -- PULSE	10	HomeSwitch / Digital In
	5 -- IEN	--	Input Enable

E' anche possibile disporre di alcune uscite digitali utilizzando i registri *digital_output* e *digital_out_mask*. Ad ogni pin di CN2 è associato un bit dei due registri, secondo quanto mostrato nella tabella che segue. Anche in questo caso, per una stessa uscita possono essere indicate due funzioni. Se non si utilizza quella primaria, si può impiegare il pin come generico digital output. Il registro *digital_out_mask* consente configurare le uscite. Ogni bit posto a 1 configura il pin corrispondente come uscita. Il registro *digital_output* controlla invece lo stato delle uscite. Ad ogni bit posto a 1 corrisponde un livello alto (24 V_{dc}) sulla corrispondente uscita.

digital_output e *digital_out_mask*

Pin number	<i>digital_output</i> <i>digital_out_mask</i> bit number	Funzione
12	0	Digital out
13	1	Digital out
15	3	Digital out
16	2	Brake / Digital out

digital_output in : miniECO e miniECO PLUS

Numero Pin di CN2	Numero Bit <i>digital_output</i>	Funzione
12	0	
13	1	Forzatura Ventola
15	3	
16	2	Brake / Digital Out

Capitolo 3



Codici di Allarme

I seguenti codici di allarme son da ritenersi validi in tutte le reti di Campo Implementate se NON diversamente indicato

Codice 1: Allarme modulo IGBT

Significato: Sovracorrente, sovratemperatura, cortocircuito IGBT.
Azione: **Asse rilasciato immediatamente.**

Codice 2: Allarme termica motore

Significato: Sovratemperatura motore.
Azione: **Fermata in rampa di emergenza e poi asse libero.**

Codice 3: Allarme corrente motore

Significato: Eccessiva corrente motore.
Azione: **Asse rilasciato immediatamente.**

Codice 4: Allarme sovratensione

Significato: Eccessiva tensione sul bus dc di potenza.
Azione: **Asse rilasciato immediatamente.**

Codice 5: Allarme sottotensione

Significato: Tensione sul bus dc di potenza inferiore al minimo.
Azione: **Fermata in rampa di emergenza e poi asse libero.**

Codice 6: Allarme termica drive

Significato: Sovratemperatura drive.
Azione: **Fermata in rampa di emergenza e poi asse libero.**

Codice 7: Allarme hardware

Significato: **Hardware guasto.**
Azione: -

Codice 8: Allarme sensore posizione

Significato: Malfunzionamento sensore posizione (resolver o encoder).
Azione: **Asse rilasciato immediatamente.**

Codice 9: Allarme memoria non volatile

Significato: Malfunzionamento alla memoria non volatile (Flash).
Azione: **Asse rilasciato immediatamente.**

Codice 10: Allarme resistore frenatura

Significato: Eccessivo intervento del resistore di frenatura.

Azione: **Fermata in rampa di emergenza e poi asse libero.**

Codice 13: Errore network

Significato: Errore comunicazione sulla rete.

Azione: **Fermata in rampa di emergenza e poi asse libero.**

Codice 14: Errore homing

Significato: Errore durante l'homing.

Azione: **Asse rilasciato immediatamente.**

Codice 15: Errore overflow

Significato: Overflow nel sistema di posizionamento.

Azione: **Fermata in rampa di emergenza e poi asse libero.**

Codice 16: Errore inseguimento posizione

Significato: Massimo errore d'inseguimento della posizione superato per un tempo superiore al timeout.

Azione: **Asse rilasciato immediatamente.**

Codice 22: Errore homing encoder assoluto

Significato: non è stato effettuato azzeramento dell'encoder assoluto, se utilizzato.

Azione: **Asse rilasciato immediatamente.**

Codice 23: Errore allineamento encoder assoluto

Significato: non è stato ancora effettuato l'allineamento del motore con encoder assoluto, se utilizzato.

Azione: **Asse rilasciato immediatamente.**

Codice 24: Errore comunicazione encoder assoluto

Significato: la comunicazione tra drive ed encoder assoluto è fallita.

Azione: **Asse rilasciato immediatamente.**

Codice 25: Errore asse in movimento all'accensione

Significato: è stato rilevato asse in movimento all'accensione utilizzando l'encoder assoluto.

Azione: **Asse rilasciato immediatamente.**

Codice 26: Configurazione non consentita

Significato: è stata impostata una modalità di funzionamento non compatibile con l'utilizzo di un encoder esterno.

Azione: **Asse rilasciato immediatamente.**



CANopen protocol

Introduzione

Questo documento è da considerare parte integrante dei documenti ufficiali relativi allo standard CANopen:

- CiA CANopen- Device Profile Drives and Motion Control – DSP 402 v1.1, che nel seguito sarà indicato con *DSP 402*
- CiA CANopen- Application Layer and Communication Profile – DS 301 v4.01, che nel seguito sarà indicato con *DS 301*

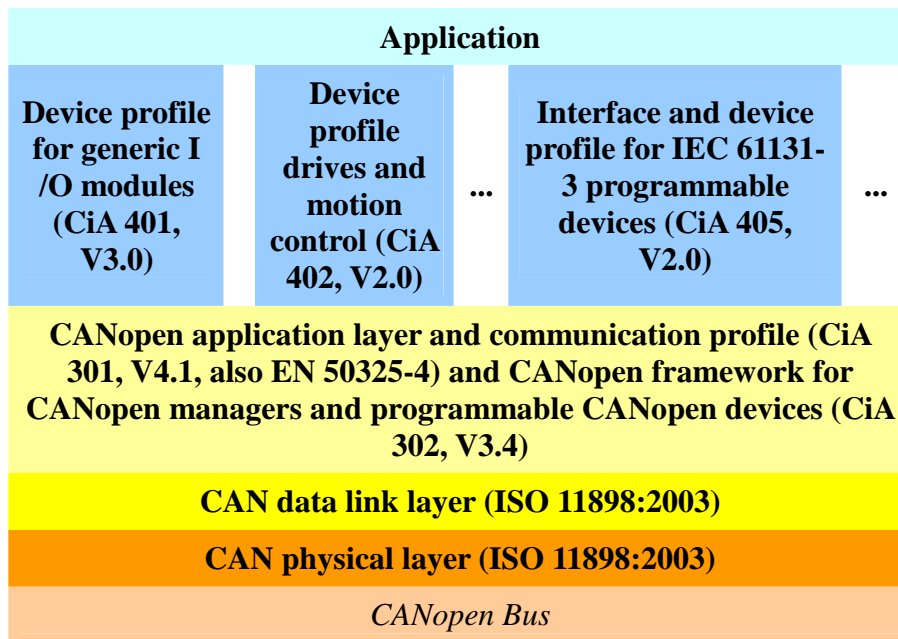
nella **descrizione generale** è data una descrizione dei principi del protocollo nei capitoli successivi **Implementazione sui Drive** sono forniti i dettagli delle implementazioni nel Drive



Descrizione Generale

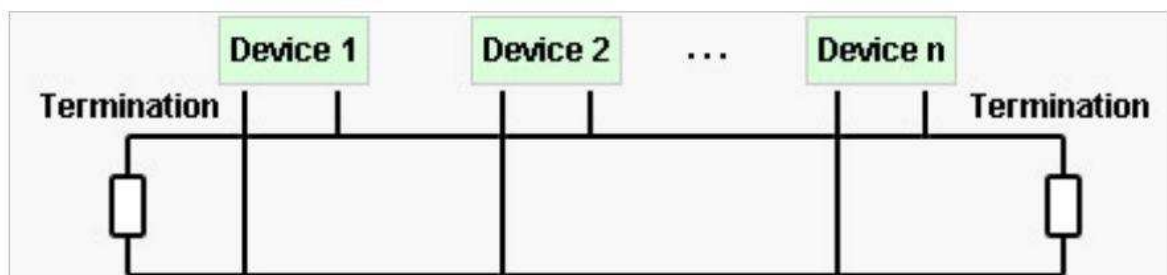
CANopen è un'applicazione standardizzata per i sistemi di automazione distribuita basate su CAN (Controller Area Network) che offre le seguenti caratteristiche prestazionali:

- ➡ Trasmissione di dati critici di processo secondo il principio produttore – consumatore
- ➡ Descrizione standardizzata del dispositivo (dati, parametri, funzioni, programmi) nella forma del cosiddetto "dizionario degli oggetti". L'accesso a tutti gli "oggetti" di un dispositivo con protocollo di trasmissione standardizzato secondo il principio client-server.
- ➡ Servizi standardizzati per il monitoraggio del dispositivo (node guarding / heartbeat), segnalazione di errore (messaggi di emergenza) e coordinamento della rete ("network management")
- ➡ Servizi di sistema standardizzati per operazioni sincrone (messaggio di sincronizzazione), unico orologio centralizzato con invio del dato di tempo tramite messaggio (Time stamp message)
- ➡ Funzioni di help standardizzate per configurare la velocità di trasmissione e il numero di identificazione del dispositivo via bus
- ➡ Assegnazione di modelli standardizzati per gli identificatori dei messaggi nelle configurazioni di semplici sistemi nella forma del cosiddetto "set di connessione predefinito" .



Struttura fisica di una rete CANopen

L'architettura CAN illustrata sotto definisce la struttura di base fisica della rete CANopen. Come si può osservare, viene utilizzata una tipologia ad una linea (bus). Per evitare riflessioni dei segnali, devono essere terminate entrambe le estremità della rete. Inoltre, sono da osservare, per il collegamento ai singoli nodi di rete, le lunghezze massime ammissibili.



I Bit rate ammessi per una rete CANopen sono riportati nella CiA 301: **10 kbps, 20 kbps, 50 kbps, 125 kbps, 250 kbps, 500 kbps, 800 kbps e 1000 kbps**. In CiA 301 sono indicate anche le disposizioni per la configurazione della temporizzazione dei bit.

Inoltre, per CANopen, due condizioni supplementari devono essere soddisfatte:

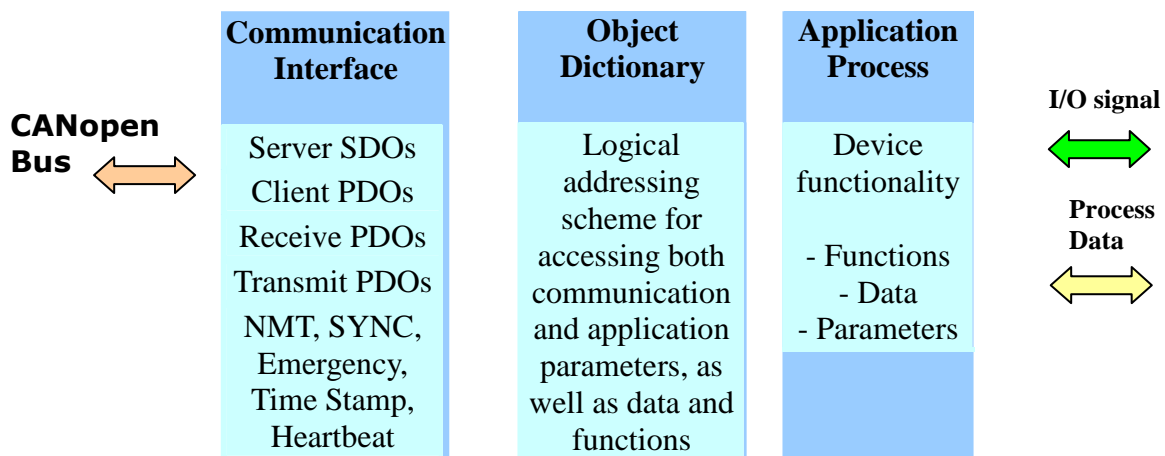
Tutti i nodi devono essere configurati per la stessa velocità bit

Non possono esistere due node-ID.

Purtroppo non ci sono meccanismi che garantiscano automaticamente queste condizioni. L'integratore di sistema è tenuto a verificare la velocità in bit e il nodo-ID di ogni singolo nodo di rete durante il cablaggio di una rete e se necessario modificarli. Normalmente il nodo-ID è configurato direttamente sul dispositivo tramite DIP-switch o selettori rotanti esadecimali. Soluzioni alternative richiedono l'impostazione di questi parametri tramite due identificatori software CAN riservati specificati nel "LSS-service" (impostazione di livello di servizio), come descritto in CiA 305 .

Object Dictionary (OD) & Electronic Data Sheet (EDS)

Una delle proprietà più importanti di CANopen è una descrizione del dispositivo standardizzato chiamato dizionario oggetti. È una tabella che ha la stessa struttura per tutti i tipi di dispositivi. In questo modo è possibile accedere dall "esterno" a tutti i dati importanti, i parametri e le funzioni di un dispositivo che utilizza un sistema di indirizzamento logico (indice, sottoindice) vale a dire attraverso il bus CAN.



Oltre alla descrizione standardizzata delle proprietà di comunicazione dei dispositivi secondo CiA 301, CANopen definisce i cosiddetti "profili dispositivo" per dispositivi dedicati a distinte aree applicative. Questi profili specificano i parametri più importanti, i dati e le funzioni per ogni tipo di dispositivo (ad esempio, moduli I / O, azionamenti, encoder, ecc.)

L' Electronic Data Sheet (EDS) contiene il tipo di dati e la funzione di ogni voce della directory. In genere, l'EDS è un file ASCII, contenente tutti i dati. Per consentire una gestione più flessibile ed estensibile dei dati, il formato è stato cambiato in XML

🔍 Data Transfer via SDO and PDO

Fondamentalmente, ci sono due diversi modi per trasferire i dati. Gli Oggetti del Servizio Dati (SDO Service Data Object) basati su una comunicazione client-server che permette l'indirizzamento diretto di un oggetto utilizzando il relativo indice e sottoindice. Viene utilizzato per la configurazione di un dispositivo, e per up-e download di grandi blocchi di dati, ma richiede un overhead aggiuntivo di protocollo.



Process Data Objects (PDO) forniscono una trasmissione di dati efficiente in base a un modello produttore-consumatore. La lunghezza dei dati è limitata a otto byte, ma non contiene alcun overhead di protocollo. Un PDO può contenere i valori di più voci del dizionario dell'oggetto, ma il contenuto di unPDO deve essere definito durante l'inizializzazione. Ogni dispositivo può specificare fino a 512 in ricezione e trasmissione entro i limiti del sistema (memoria, potenza di elaborazione) o la rete (numero di identificatori CAN disponibili)

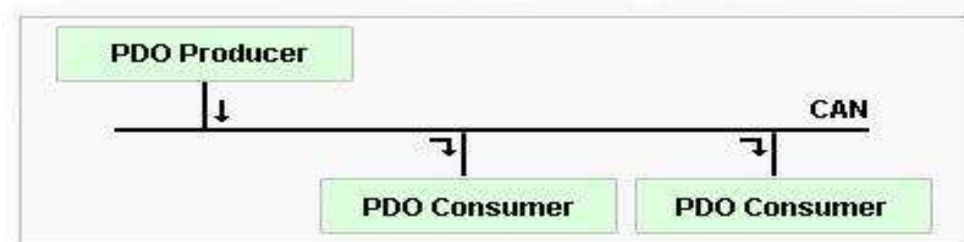
Byte 0

Byte 7

CAN-ID PDO1	Data						
--------------------	-------------	--	--	--	--	--	--

	Byte 0	Byte 1	Byte 3	Byte 6	Byte 7	
CAN-ID PDO2	Data 1 Speed		Data 2 Position		Data 3 Target	

	Byte 0	Byte 1	Byte 3	Byte 4	Byte 6	Byte 7
CAN-ID PDO3	Data 1 Temperature		Data 2 Voltage	Data 2 Current		Data 3 Target



Un PDO può essere generato remote request, da un evento interno come ad esempio una risposta ad un trigger , un timer, o quando risponde (ciclicamente) ad un messaggio in ricezione di trasmissione sincrona (SYNC) .Tutti i nodi della rete sono in grado di ricevere il messaggio (PDO-Consummers). Filtrando il CAN-ID solo gli oggetti selezionati possono essere selezionati per l'ulteriore elaborazione.

Emergency Messages

Poiché CANopen non è un gerarchica master-slave, il monitor del nodo trasmette soltanto lo stato di comunicazione e non lo stato del nodo attuale. Ogni nodo pertanto richiede un identificatore CAN ad alta priorità per indicare le situazioni di errore. Questo meccanismo è denominato "Emergency Messaging" ed il relativo oggetto di comunicazione associato "Emergency Message" . Tale messaggio di emergenza consiste di otto byte di dati nella forma seguente:

Error Code	Error Register	Vendor specific error field
-------------------	-----------------------	------------------------------------

I codici di errore sono specificati nella DS-301. Simultaneamente con la trasmissione del messaggio di emergenza, il dispositivo scrive il codice di errore nello storico degli error. Il registro di errore è contenuto della voce OD con codifica bit per bit della causa dell'errore .

Device Configuration via SDOs

Oggetti specifici di comunicazione, i cosiddetti "Service Data Objects" (SDO) vengono utilizzati per l'accesso diretto ai dispositivi CANopen. Con questi "Service Data Objects", possono essere letti e scritti gli oggetti presenti nelle voci del dizionario la cui comunicazione avviene sempre come un logico collegamento 1:1 (peer-to-peer) tra due nodi (ad esempio, la configurazione di un nodo e un nodo da configurare). Poiché il trasferimento dei dati avviene tramite una richiesta ed un messaggio di risposta , ciò significa che sono necessari due messaggi CAN : un messaggio per la richiesta al nodo di rete (richiesta SDO o "Client SDO") ed un secondo per l' risposta (risposta SDO o "Server SDO") del nodo. I due nodi di rete coinvolti sono indicati come SDO client e / o server SDO, qui l'ID del server è quello che offre o accetta i dati tramite il proprio dizionario oggetti e il client è quello che richiede (lettura) o trasferisce (scrive) i dati. Vi è una connessione logica tra i due partner, che è anche indicato come un "canale SDO"(SDO channel). L'iniziativa per un trasferimento SDO proviene sempre dal client. Dall' istante in cui è riconosciuto un trasferimento SDO, ogni richiesta deve essere risolta, anche se il dispositivo non è in grado di fornire dati significativi o la domanda in sé risulta errata. Tale risposta negativa si chiama "abort". In una tale risposta, oltre ai 4-byte di codice di errore (abort code), che specifica la causa della interruzione, deve avere anche l'indirizzo della voce del dizionario relativo all'oggetto che ha causato l'interruzione del canale SDO. Come già accennato, un trasferimento SDO è eseguito sempre come sequenza richiesta-risposta con un protocollo separato, che è specificato nel primo byte di dati dell'oggetto SDO. Pertanto l'identificatore del messaggio specifica l' SDO stesso e il primo byte di dati dell' SDO specifica il protocollo relativo. Per questo motivo il primo byte di dati è indicato anche come byte di protocollo o di

comando. Un messaggio SDO è sempre otto byte, i bit del campo dati che non sono necessari devono essere impostati a 0.

Possono essere trasmessi campi dati di qualsiasi lunghezza o sequenze di byte tramite l'accesso al dizionario oggetti. Per questo motivo, la lunghezza delle informazioni che possono essere trasmesse attraverso il protocollo SDO è teoricamente illimitato. Il protocollo SDO viene eseguito in due fasi: Nella fase di inizializzazione viene indirizzata una voce del dizionario oggetto ed è indicata inoltre la lunghezza dei dati da trasferire. Nella seconda fase sono poi trasmessi i dati di utili in segmenti (ciascuno di sette byte).

Il DS-301 distingue quattro diversi servizi SDO: Partenza SDO Upload, Upload segmento SDO, Partenza SDO Download e Download segmento SDO.

Come molto spesso accade solo pochi byte di dati sono richiesti, e quindi il trasferimento SDO può essere ridotto fino ad un massimo di quattro byte, trasmessi già nella fase di inizializzazione. Questo è indicato come un "expedited SDO".

Il messaggio per Partenza SDO download, con il relativo e contemporaneo accesso in scrittura ad una voce del dizionario degli oggetti di un nodo CANopen è strutturato come segue:



The SDO server responds with protocol byte 0x60:



comando espresso a bit il servizio è codificato con tre bit (command specifier). Un'altro bit specifica se è un "expedited SDO" oppure no. Un altro bit indica se la dimensione dei dati da trasmettere è specificato negli ultimi quattro byte del communication object, tuttavia, questo bit viene utilizzato solo per i trasferimenti NON "expedited".

Con un trasferimento accelerato, invece, i dati dell'utente sono trasmessi direttamente all'interno di questi ultimi quattro byte. Due ulteriori bit del byte di protocollo servono a specificare quanti di questi byte sono effettivamente assegnati (infatti è possibile la trasmissione anche di un solo byte dei dati utente). I dati utente devono quindi essere posizionati allineati a sinistra nel campo dati dell'oggetto SDO.

Generalmente è da notare che in CANopen i dati vengono trasmessi secondo la regola del "Little Endian" e quindi secondo la forma corrispondente ad INTEL. Ciò significa che il byte di valore più basso viene trasmesso per primo. Questo rende un po' più difficile per un essere umano eseguire un controllo di una sequenza del

protocollo SDO, ma alla fine è una questione di abitudine. Contando i bit del byte di protocollo descritto finora, vi sono sette bit. L'ottavo bit è riservato.

Un Download SDO alla voce OD [1017], con il quale si voglia impostare a 4 secondi l'intervallo di heartbeat (in ms come valore UNSIGNED16, cioè 0x0F A0), appare

Nel byte di dunque come segue:

2B	17 10	00	A0 0F 00 00
----	-------	----	-------------

Il nodo (SDO server) riconosce il completamento con successo dell'operazione con il messaggio

60	17 10	00	00 00 00 00
----	-------	----	-------------

Con **Initiate SDO Upload service (Start upload SDO)**, con cui viene letta una voce del dizionario degli oggetti di un nodo CANopen è valida la stessa divisione del campo dati solo che in questo caso il telegramma di richiesta e risposta sono rovesciate ad una determinata estensione. In questo esempio il command byte della richiesta del client è 0x40:

Command byte (0x40)	OD Main Index	OD Sub Index	Empty (4 bytes)
---------------------------	------------------	--------------------	-----------------

Il server SDO risponde con:

Command byte	OD Main Index	OD Sub Index	Data (4 bytes)
-----------------	------------------	--------------------	----------------

Il server SDO deve sempre rispondere ad una richiesta tipicamente quale " device manufacturer "

(Vendor ID , oggetto[1018] , subindex 1), Il dispositivo se poniamo l'ipotesi che il vendor ID sia 0006 il messaggio di risposta sarà come segue:

43	18 10	01	06 00 00 00
----	-------	----	-------------

Se non è utilizzato un trasferimento in modalità " **expedited**", i quattro byte di dati dell' SDO **Initiated SDO** (upload o download) **service** possono essere utilizzati per specificare la lunghezza (in byte) di user data da trasmettere. La trasmissione avviene poi con la trasmissione (**upload o download**) **segmento SDO**. Possono pertanto essere trasmessi 7 byte di dati utente per segmento . Il byte di comando di questi servizi contiene i tre bit del service-ID (identificatore di comando), un bit di commutazione e quattro bit inutilizzati, ad eccezione dell'ultimo segmento. Per trasferire gli user data in modo sicuro anche quelli che non sono multipli del segmento a dimensione 7, il numero di byte non utilizzati (un valore compreso tra 6 e 0) è codificato nei tre bit dell' ultimo segmento SDO. Infine l'LSB del del Byte di comando segnala la fine della trasmissione dei dati. L'ordine dei segmenti viene monitorato dal toggle bit, in cui SDO request ed SDO response hanno lo stesso bit di toggle al medesimo stato.

La sequenza commentata di un **non-expedited SDO upload** segmentato è illustrato qui di seguito:

```

40 08 10 00 00 00 00 00 // Initiate req: Read Device Name [1008]
41 08 10 00 1A 00 00 00 // Initiate resp: Fine. It's 26 bytes long
60 00 00 00 00 00 00 00 // Upload segment req, Toggle = 0
00 54 69 6E 79 20 4E 6F // Upload segment resp, Toggle = 0
70 00 00 00 00 00 00 00 // Upload segment req, Toggle = 1
10 64 65 20 2D 20 4D 65 // Upload segment resp, Toggle = 1
60 00 00 00 00 00 00 00 // Upload segment req, Toggle = 0
00 67 61 20 44 6F 6D 61 // Upload segment resp, Toggle = 0
70 00 00 00 00 00 00 00 // Upload segment req, Toggle = 1
15 69 6E 73 20 21 00 00 // Last segment, 2 bytes free, Toggle = 1

```

Con la versione 4 delle specifiche CANopen DS-301, è stato introdotto un nuovo modo **SDO** molto più efficace, ma anche più complicato : il trasferimento a blocchi chiamato anche **SDO Block Transfer**. In contrasto con il trasferimento a segmenti sopra descritto, qui i segmenti non sono più riconosciuti singolarmente, ma sono costruiti a blocchi, che vengono trasferiti ciascuno in una sola volta. Il partner riconosce solo il blocco. Da una dimensione dei dati utente di 29 byte in su, il trasferimento a blocchi è migliore in termini di overhead di protocollo. Con il trasferimento a blocchi, si possono trasmettere fino ad un massimo di 127 segmenti per blocco . La trasmissione ha una fase di inizializzazione in cui vengono comunicate le dimensioni dei blocchi di dati con una verifica da parte di entrambi i partner che le dimensioni siano coerenti tra loro, e da una fase di terminazione, in cui si verifica tramite un CRC l'intero trasferimento. Tuttavia il trasferimento a blocchi **SDO Block Transfer** è attualmente supportato solo da alcuni dispositivi.

Un importante servizio SDO è "Abort SDO Transfer"(command byte 0x80). Esso consta esattamente d un unico messaggio CAN che può essere trasmesso in qualsiasi momento da ciascuno dei due partner e ciò provoca l'immediata cessazione della trasmissione dell'SDO. La situazione più comune è l'SDO-Abort come risposta ad una richiesta " **Initiate SDO**" se non esiste l'indirizzo della voce nel dizionario degli oggetti. La struttura del messaggio di Abort SDO è la seguente:



Il campo dati di questo SDO service contiene la causa dell'errore nello formato dword. Le specifiche del CANopen listano tutti gli Abort Code . Essi sono approssimativamente 30 in totale . L'uso di un proprio Abort Code oppure un Abort code indefinito non è permesso. I principali abort codes sono:

```

0x05030000 Toggle bit not alternated
0x05040001 Invalid SDO Command specifier
0x0601000x Unsupported access to an object
0x06010002 Attempt to write a read only object
0x06020000 Object does not exist in the object dictionary
0x0607001x Data type does not match
0x06090011 Subindex does not exist
0x08000000 General error

```

Ciascun dispositivo deve supportare almeno un canale server SDO . Altri canali SDO possono essere settati tramite le entries del dizionario degli oggetti. Le entries dell' OD (Object Dictionary) da 1201 a 127F sono riservate alla definizione di altri canali server SDO aventi parametri dei record predefiniti.

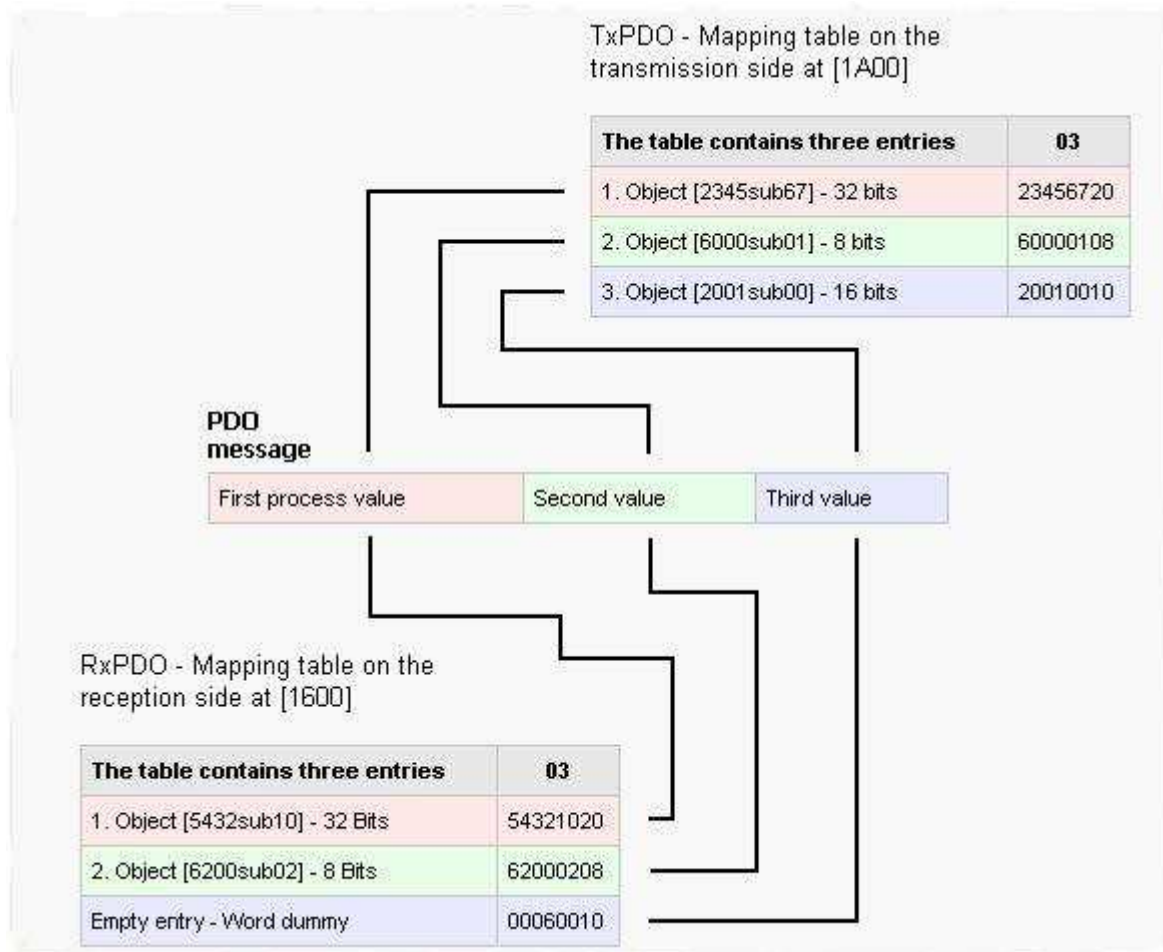


Process data exchange with PDOs ("Process Data Objects")

Il compito principale di un sistema CANopen è naturalmente lo scambio dei dati di processo. Per questo sono fornite le specifiche per la maggioranza degli identificatori CAN, ed anche la maggior parte delle voci del dizionario degli oggetti. Per la trasmissione dei dati di processo, questo avviene senza sovraccarico di un ulteriore protocollo aggiuntivo e la veicolazione avviene secondo il principio "produttore-consumatore". Ciò significa che un messaggio trasmesso da un nodo ("produttore") può essere ricevuto da tutti gli altri nodi (i "consumatori"). Questo principio è chiamato anche "broadcasting" e rappresenta un principio molto efficiente di trasmissione dei dati, in particolare se un messaggio (ad esempio un messaggio di sincronizzazione) è trasmesso contemporaneamente a tutti i partecipanti al processo.

Un messaggio CAN che contiene i dati di processo è denominato PDO ("Process Data Object"). Come già descritto, la trasmissione di PDO è possibile solo nello stato "Operational". I PDO non hanno formato fisso. Il campo dati di un PDO può essere compreso tra uno e otto byte di dati . Il contenuto di un PDO non può essere facilmente interpretato. L'idea di base è che sia il trasmettitore e il ricevitore a sapere come il contenuto di un PDO deve essere interpretato. Per questo motivo è sufficiente per identificare un PDO solo il suo COB-ID. La cosiddetta "**mappatura PDO (PDO Mapping)**" descrive nel campo dati di un PDO quali variabili di processo devono essere trasmesse, come devono essere disposte e quale tipo di dati e lunghezza esse abbiano. Quindi il contenuto e il significato del campo dati di ciascun PDO definito è descritto nella forma di un record nella mappatura - PDO all'interno del dizionario oggetti sia dal lato trasmissione che dal lato di ricezione. Il produttore del PDO compone il campo di dati di un PDO conformemente alla mappatura TxPDO. Per fare ciò prende i dati correnti delle variabili da trasmettere dal suo dizionario oggetti e copia questi dati nel campo del PDO prima che il messaggio CAN (PDO) venga inviato. Lo stesso accade sul lato dei consumatori: sulla base del record di mappatura RxPDO, i byte dei dati dei PDO ricevuti vengono copiati nelle voci locali del dizionario degli oggetti, e quindi generando le azioni specifiche del dispositivo (ad esempio, il set delle uscite digitali). Un nodo di rete che vuole accettare un certo PDO deve solo attivare il corrispondente messaggio CAN. Ciò viene fatto con la validazione del corrispondente COB-ID nel OD.

Ma ora torniamo alla Mappatura PDO. Il principio della disposizione delle variabili di processo (mapping) è mostrato successivamente (le variabili sono disponibili in forma di voci di dizionario di oggetto nell' "Application Profile"). La mappatura delle singole variabili di processo nel campo di dati di un PDO è descritta sotto forma di una tabella. Ciò viene anche dato come voce del dizionario oggetti (DO), vale a dire per ogni PDO in trasmissione e in ricezione-PDO [16xx] o [1Axx]. Queste tabelle, e quindi la relativa mappatura delle variabili di processo nel campo di dati di un PDO può essere configurata tramite SDO con accessi in scrittura.



In questo esempio ci sono esattamente due link agli oggetti: da oggetto (variabile di processo) [2345sub67] del produttore del PDO all'oggetto [5432sub10] del consumatore del PDO e da oggetto [6000sub01] del produttore all'oggetto [6200sub02] del consumatore. Il terzo oggetto di trasmissione, [2001sub00] non viene analizzato dal lato del ricevitore e viene quindi coperto con un cosiddetto oggetto fittizio.

Le tabelle per la mappatura PDO hanno un formato dati così composto Subindex 0 contiene il numero di oggetti mappati e i subindex 1-8 sono mappate le voci del dizionario degli oggetti stessi come DWORD. Questo contiene l'indirizzo lungo 24-bit indirizzo del OD (indice, sottoindice) e un campo di 8-bit della voce (entry) dell'Object Dictionary (OD). Un dispositivo che supporta otto voci di mapping ha una granularità di 8 e non può quindi realizzare mappature di "byte". Dispositivi con una granularità di 1, invece, supportano la mappatura di ogni singolo bit del PDO ("bit-mapping"). Di conseguenza, nella tabella di mappatura, ci devono essere 64 voci (entry).

Tuttavia, un PDO è descritto non solo dalla mappatura ma anche dai suoi parametri di comunicazione. Questi sono: COB-ID, identificativo cioè del messaggio CAN, il "tipo di trasmissione (Transmission type)", l'"Inhibit time", e possono essere configurati da una corrispondente voce nel dizionario oggetti. La posizione di questa voce aggiuntiva del Dizionario Oggetti ha un offset di 0x200 prima della sua associata voce di mappatura, e cioè [18xx] per i trasmit PDO e [14xx] per i receive PDO. Ogni PDO è quindi descritto da due diverse voci dell'OD, che

appartengono allo stesso insieme ed entrambi devono essere configurate dal system integrator. La tabella seguente mostra il record dei "parametri PDO"

Sub-Index	Contents	Data type	<input type="checkbox"/>
0	Largest sub-index supported	BYTE	<input type="checkbox"/>
1	COB-ID	DWORD	<input type="checkbox"/>
2	Type	BYTE	<input type="checkbox"/>
3	Inhibit time in ms	WORD	<input type="checkbox"/>
5	Event timer	WORD	<input type="checkbox"/>

Il subindex 1 contiene l'identificatore CAN del PDO che nella DWORD è allineato a destra. Il bit di valore più alto invece indica se il PDO è attivo (valido) o inattivo (non valido). Un MSB settato a 1 significa NON valido, per esempio, un valore di 0x80000199, descrive un TxPDO non valido del nodo numero 25.

Il tipo di trasmissione di un PDO può essere impostato tramite il secondo subindex. In primo luogo è necessario distinguere tra PDO sincroni e asincroni:

Value (dec)	Type
0	acyclic synchronous
1...240	cyclic synchronous
241...251	reserved
252	synchronous RTR only
253	asynchronous RTR only
254...255	asynchronous

I PDO asincroni sono controllati dagli eventi e rappresentano il tipo normale di trasmissione delle PDO. Ciò significa che, quando almeno una delle variabili di processo mappate nel PDO viene alterata, per esempio un valore di input, il PDO viene trasmesso immediatamente. Per questo, i valori 255 o 254 sono da inserire come tipo PDO.

I PDO sincroni vengono trasmessi solo dopo previo ricevimento di un messaggio di sincronizzazione (Sync Object). La trasmissione dei PDO viene quindi effettuata sincrona in tutta la rete, più o meno nello stesso momento. Ma ciò che è più importante è che tutti gli ingressi del dispositivo devono essere campionati all'arrivo dell'oggetto di sincronizzazione, in modo che ci sia una fotografia istantanea e uniforme dei risultati del processo. Con il successivo messaggio SYNC, i dati registrati vengono poi inviati nei PDO sincroni. Pertanto così come esiste un ritardo corrispondente al tempo di ciclo del messaggio Sync, altrettanto i consumatori ricevono le variabili di processo catturate al Sync precedente. In uscita i PDO sincroni ricevuti da un nodo sono validi soltanto all'arrivo del messaggio di sincronizzazione successiva.

Al fine di impedire che il bus non sia intasato/bloccata da un gran numero di PDO sincroni, che sono tutti inviate ad ogni messaggio Sync, i valori 1 .. 240 del tipo sincrono ciclico PDO sono utilizzati come divisori per l'intervallo di trasmissione. Di conseguenza, $[18xsub02] = 4$ significa che il PDO sincrono viene inviato solo con ogni quattro messaggi Sync. Non influenzato ciò, il dispositivo deve comunque effettuare la registrazione dei suoi ingressi ad ogni messaggio di sincronizzazione, in quanto può accadere che le variabili di processo sincrono registrati in un PDO vengono richiesti da RTR. In questo caso il dispositivo CANopen deve trasmettere il PDO corrispondente con i valori rilevati immediatamente sincrono. In [1006], il parametro "Periodo di ciclo di comunicazione", ai nodi può essere notificato l'intervallo di sincronizzazione in microsecondi

Infine, nel sottoindice 3 può essere impostato un "tempo di blocco" per PDO asincroni. Il valore deve essere indicato come un multiplo di 100 microsecondi. Il tempo di blocco è utile quando si verificano frequenti alterazioni incontrollate di valori di input: ad esempio con un ingresso analogico aperto. Se è configurato il tempo di inibizione, il nodo non può trasmettere il PDO rilevante nuovamente prima dello scadere del tempo di inibizione; questo assicura che non vi sia bus load troppo elevato a causa di un cosiddetto "babbling idiot". Il tempo di blocco viene utilizzato solo per TxPDO, naturalmente. Non ha significato per RxPDOs.

Nei TxPDO asincroni un PDO può essere trasmesso ciclicamente con il timer evento, sottoindice 5. Se viene scritto un valore maggiore di 0, il timer si attiva, ed il numero rappresenta il valore del espresso in mS. Allo scadere del timer viene trasmesso il PDO. La trasmissione avviene quindi sia da una variazione di un input esterno che dal timer evento scaduto. Questo sottoindice è significativo solo per trasmettere PDO

Infine, nel sottoindice 3 può essere impostato un "tempo di blocco" per PDO asincroni. Il valore deve essere indicato come un multiplo di 100 microsecondi. Il tempo di blocco è utile quando si verificano frequenti alterazioni incontrollate di valori di input: ad esempio con un ingresso analogico aperto. Se è configurato il tempo di inibizione, il nodo non può trasmettere il PDO prima dello scadere del tempo di inibizione; questo assicura che non vi sia caricamento del bus troppo elevato a causa di un cosiddetto "babbling idiot". Il tempo di blocco viene utilizzato naturalmente solo per TxPDO. Non ha significato per RxPDOs.



Emergency Message

Poiché CANopen non è un gerarchica master-slave, ed il monitoraggio nodo evidenzia solo lo stato della comunicazione e non bensì lo stato attuale del nodo , ogni nodo richiede quindi un identificatore di priorità CAN per indicare situazioni di errore. Questo meccanismo è denominato "Emergency Messaging" ed è associato all'oggetto "Emergency Message" . Tale messaggio di emergenza consiste di otto byte di dati nella forma seguente:



The error codes are specified in DS-301. The high byte distinguishes the error categories:

Error code (hex)	Error description
00xx	Error Reset / No Error
10xx	Generic Error
2xxx	Current
3xxx	Voltage
4xxx	Temperature
50xx	Device Hardware
6xxx	Device Software
70xx	Additional Modules
8xxx	Monitoring
90xx	External Error
F0xx	Additional Functions
FFxx	Device Specific

Simultaneamente alla trasmissione del messaggio di emergenza , il device scrive l'error code a[1003], dov'è memorizzata la storia dell'errore. Il registro di errore è il contenuto dell' OD entry [1001] con codifica a bit per la causa dell'errore:

Bit	Error cause
0	Generic Error
1	Current
2	Voltage
3	Temperature
4	Communication Error
5	Device Profile Specific
6	Reserved (always 0)
7	Manufacturer Specific

L'identificatore CAN del messaggio di emergenza viene memorizzato nel dizionario oggetti dispositivo sotto la voce [1014], però questa voce dell' OD è facoltativa.



CANopen network management (NMT)

Oltre a fornire servizi e protocolli per la trasmissione dei dati di processo e la configurazione dei dispositivi, il funzionamento di un sistema distribuito su una rete richiede funzioni per il comando di controllo dello stato di comunicazione dei singoli nodi di rete. La trasmissione dei dati dai dispositivi CANopen è in molti casi scatenata da eventi, pertanto è necessario un monitoraggio continuo della capacità di comunicazione dei nodi della rete . CANopen fornisce i cosiddetti servizi e protocolli per la "gestione della rete" vale a dire

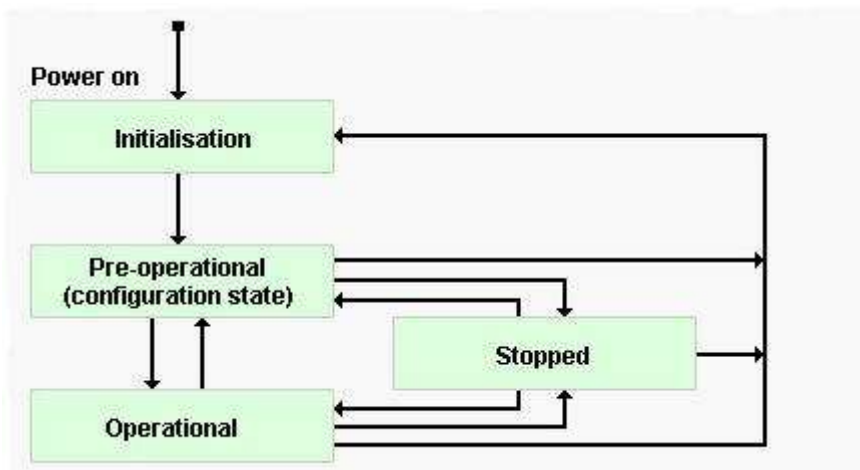
- Controllo dello stato di comunicazione dei nodi di rete e
- Monitoraggio nodo

Status dei nodi di una rete CANopen e controllo dello stato tramite messaggi NMT

CANopen descrive lo stato di comunicazione di un nodo di rete con diagramma di stato. Inviando messaggi specifici messaggi CAN (NMT message), il master di gestione di rete può controllare lo stato di comunicazione degli altri nodi di una rete CANopen, cioè può cambiare lo stato di tutti i nodi o di un singolo nodo da una singolo comando.

I messaggi NMT vengono trasmessi con l'identificatore a più alta priorità (CAN-ID 0). Il campo di dati è costituito da due soli byte: lo stato che dovrà assumere il nodo è codificato nel primo byte di dati, il secondo byte di dati specifica il numero del nodo il cui stato di comunicazione deve essere modificato. Tutti i nodi di una rete sono affrontati congiuntamente con il nodo virtuale-ID 0, in questo modo, per esempio, a tutti i nodi può essere impostato allo stato "Operational" nello stesso istante permettendo così un avvio simultaneo

Stati e Cambiamento di stato



Al fine di consentire anche un ripristino parziale di un certo nodo, questo stato è suddiviso in tre sotto-stati: "Reset-Application", "Reset-Communication" e "Initializing".

Dopo un HW-Reset o Power-On, un nodo si trova nello stato di "Initializing". Dopo il completamento del nodo di inizializzazione di base (host controller, controller CAN, software, ecc), il nodo trasmette il cosiddetto "boot-up message" e si pone allo stato "Pre-Operational".

Nel sub-stato "Reset-Application", i parametri specifici del costruttore ed del "device profile" del dispositivo sono ripristinati ai valori di Power-On (corrisponde agli ultimi valori salvati). Poi il nodo passa al sotto-stato "Reset Communication".

Nel sotto-stato "Reset-comunicazione", i parametri del profilo di comunicazione sono riportati ai valori di Power-On. Poi il nodo stato passa a "Initializing"

Pre-Operational

Questo stato viene utilizzato principalmente per la configurazione di dispositivi CANopen. Pertanto lo scambio di dati di processo (via PDO) non è possibile in questo stato. Le voci del dizionario oggetti (DO) del dispositivo sono accessibili tramite "Service Data Object" (SDO). Trasmettendo un messaggio SDO, può essere modificato il dizionario oggetti di un certo dispositivo, ad esempio con uno strumento di configurazione.

Oltre alla comunicazione tramite messaggi SDO possono essere trasmessi o ricevuti nello stato di pre-operationa, emergenza, sincronizzazione di data e ora, e, naturalmente, i messaggi di controllo NMT. Trasmettendo un "Start-Remote-node", un nodo passa allo stato "Operational".

Operational

In questo stato è possibile la trasmissione dei dati di processo tramite "process data objects" (PDOs).

Stopped

In questo stato un nodo non può trasmettere o ricevere qualsiasi altro messaggio che il node guarding o l'heartbeat

CANopen Device Monitoring usando I meccanismi di node guarding and heartbeat

Per assicurare funzionalità dei nodi di rete, CANopen fornisce due alternative: Interrogazione ciclica dello stato del nodo da un'istanza di ordine superiore, il cosiddetto "NMT-Master" ("node guarding") o Trasmissione automatica di un "messaggio di heartbeat" dai nodi della rete (principio del " heart bit").



Node Monitoring via Node-Guarding or Heartbeat Messages

Per assicurare funzionalità dei nodi di rete, CANopen fornisce due alternative: Interrogazione ciclica dello stato del nodo da un'istanza di ordine superiore, il cosiddetto "NMT-Master" ("node guarding") o Trasmissione automatica di un "messaggio di heartbeat" dai nodi della rete (principio del " heart bit").

Un CAN-ID per nodo è tenuto a monitorare lo stato del dispositivo di comunicazione. Sono riservati per questo scopo identificatori di bassa priorità dei messaggi con un valore di 1792 + node-ID.

Con il node guarding, l' NMT-master richiede agli altri nodi della rete (di cui al secondo che NMT-slave) con un remote frame CAN di trasmettere ad intervalli regolari (time guarding) uno dopo l'altro un telegramma con il proprio stato (stopped , pre operational e operational) insieme ad un toggle-bit. Se un nodo non risponde alla richiesta del NMT-master entro un determinato tempo (node life time), questo viene valutato come un guasto del nodo e indicato al controllore host del NMT-master come "node-guarding event ". D'altra parte anche le NMT-degli slaves se hanno ricevuto una richiesta del NMT-master nel loro "node life time". Se tale richiesta è assente per un tempo più lungo, NMT-slave presuppone che la stesso NMT-master non è riuscito e indica questo evento come "node-guarding event "al suo controller host.

Con il monitoraggio del nodo secondo il principio del' heart bit , un nodo trasmette automaticamente lo stato di comunicazione a intervalli regolari come prova della sua capacità di comunicazione. L'intervallo tra due messaggi heartbeat (" heartbeat interval") viene configurato tramite la voce di OD [1017]. Il valore 0 disattiva il meccanismo dell' heartbeat. Il cosiddetto " heartbeat consumer time " fino a 127 nodi di rete viene riportato alla voce OD [1016]. Questo intervallo di tempo descrive il periodo di tempo massimo per l'arrivo di un messaggio heartbeat.

Ad oggi, il node guarding non è più utilizzato. Questo potrebbe essere

principalmente motivato al sovraccarico del bus (a causa dei 2 messaggi CAN ogni intervallo di monitoraggio), ma anche alla indesiderabile centralizzazione della verifica del nodo funzionante nel NMT-master

In un messaggio **Node guarding** o **heartbeat** lo stato della comunicazione viene trasmesso sotto forma di un valore a un byte:

t/r	Node State	t Toggle-bit with node guarding
		r Reserved (= 0) with heartbeat

I valori di stato stabiliti sono i seguenti:

0x00 - Bootup; **0x04 - Arrestato**; **0x05 - Operational**; **0x7F - Pre-Operational**. Al bit di valore più alto viene assegnato un ruolo speciale – con **Node Guarding** deve diventare toggle, con **heratbeat** deve essere 0 costanti.

Il messaggio di stato del nodo ha una particolare applicazione chiamata "**bootup event**". Questo messaggio ("**bootup message**") viene inviato automaticamente da un nodo di rete non appena cambia dallo stato "Initializing" allo stato "Pre-Operational"; ciò notifica a tutti i nodi già presenti in una rete CANopen della presenza di un nuovo nodo . Inoltre, in questo modo un nodo master (NMT-master) viene informato del momento in cui quando può iniziare la configurazione di un nodo. Il byte di dati del messaggio di avvio ha il valore 0x00.



Implementazioni nei Drive

Caratteristiche principali del CANopen implementate nel Drive

NMT:	slave
Controllo degli errori:	node guarding, heartbeat
Node Id:	parametro salvato nella memoria EEPROM del drive
Bit-rate:	parametro salvato nella memoria EEPROM del drive
Numero di PDO:	4 in ricezione (RPDO), 4 in trasmissione (TPDO)
Modi PDO:	sincrono (ciclico e aciclico), asincrono
PDO linking:	si
PDO mapping:	variabile, fino a 8 entità per PDO
Numero di SDO:	1 server, 0 client
Messaggi di emergenza:	si
Versione CANopen:	DS 301 v4.01
Device Profile:	DSP 402 v1.1
Modalità standard:	posizionatore controllo di velocità zero assi (homing) interpolatore
Modalità specifiche (solo su alcuni modelli):	posizionatore ad impulsi asse elettrico riferimento analogico

Definizioni, acronimi e abbreviazioni

CAN	Controller Area Network
CiA	CAN in Automation
COB	Communication Object: è l'unità di trasporto. I dati devono essere trasmessi attraverso la rete CAN in un COB.
COB-ID	COB Identifier: identifica univocamente un COB sulla rete. L'identificatore determina la priorità del COB nel sottolivello MAC.
MAC	Medium Access Control: è uno dei sottolivelli del CAN Data Link Layer e la sua funzione è quella di arbitrare l'accesso al bus.
PDO	Process Data Object.
SDO	Service Data Object.
IDO	Internal Data Object.

I numeri in base esadecimale hanno il prefisso **0x**.

I numeri senza prefisso s'intendono in base dieci.



CANopen device profile: drives and motion control

Introduzione

Il protocollo CANopen specifica i servizi offerti dal **CAN Application Layer (CAL)**, corrispondente al settimo livello del modello di riferimento ISO/OSI.

Il CAL offre molti servizi che possono essere così raggruppati:

- **CAN-based Message Specification (CMS)** è un linguaggio con cui si descrivono i COB che vengono usati nel dispositivo. Include anche le regole per la codifica dei dati.
- **Network Management (NMT)** permette di gestire l'inizializzazione, la configurazione e la gestione degli errori dei nodi della rete.
- **Distributor (DBT)** si occupa dell'allocazione degli identificatori dei COBs che vengono usati dal servizio CMS.
- **Layer Management (LMT)** permette di configurare i parametri dei nodi della rete.

Profili

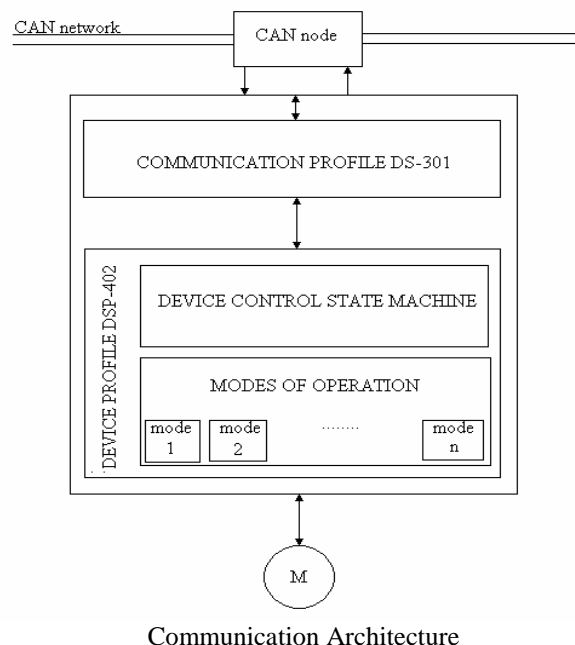
Il profilo definisce un sottoinsieme dei servizi definiti dal protocollo di comunicazione, restringendone il campo. In particolare le specifiche CANopen sono composte da profili basati sul modello di riferimento CAN.

I profili di CANopen sono due:

- **CANopen Communication Profile (DS 301)**: ogni dispositivo è dotato di questo profilo che realizza l'interfaccia fra CAN e CAL.
- **CANopen Device Profile**: definisce come le funzionalità del dispositivo sono raggiungibili dal CAN-bus e quale Communication Profile è necessario usare. È un profilo specifico per il drive che ne rende univoco il comportamento nei confronti della rete CAN, descrivendo i meccanismi di base per la comunicazione tra i dispositivi.

In questo documento si descrive il particolare Device Profile "Drives And Motion Control" del Drive in conformità allo standard DSP 402.

Il Device Profile si divide ulteriormente in due sezioni: il **device control** e il **modes of operation**. Il primo specifica la macchina a stati che regola il funzionamento del dispositivo; il secondo definisce il modo di funzionamento del drive. È il costruttore che decide quali modi di funzionamento vengono implementati.



Object e object dictionary

CANopen usa un approccio orientato agli oggetti: ogni dispositivo è rappresentato da un insieme di oggetti, ognuno dei quali ha una funzionalità specifica. In funzione della particolarità che rappresenta, l'oggetto può essere composto da una singola variabile, da un array, se le componenti sono tutte dello stesso tipo, o da un record, se le componenti sono eterogenee.

Ad ogni oggetto è associato un indice a 16 bit senza segno per selezionare l'oggetto stesso e un sub-indice a 8 bit senza segno che permette di selezionare ogni singolo elemento degli oggetti complessi. Il sub-indice è sempre 0 per gli oggetti semplici e parte da 1 per gli oggetti complessi.

La parte più importante del Device Profile è la descrizione dell'Object Dictionary. L'Object Dictionary è essenzialmente la lista degli oggetti del dispositivo accessibili dalla rete in maniera predefinita.

Gli oggetti del dizionario vengono divisi in categorie, a cui è associato uno specifico range di valori dell'indice:

- Index 0000 non usato
- **Index 0001 – 001F Static Data Types:** questi oggetti contengono le definizioni di tipo per i dati standard (booleani, interi, virgola mobile, stringhe, ...). Non si possono né leggere né scrivere.
- **Index 0020 – 003 Complex Data Types:** questi oggetti definiscono strutture composte da tipi standard che sono comuni a tutti i dispositivi.
- **Index 0040 – 005F Manufacturer Specific Complex Data Types:** anche questi oggetti definiscono strutture composte da tipi standard ma che sono specifiche del dispositivo.
- Index 0060 – 0FFF riservato per usi futuri.
- **Index 1000 – 1FFF Communication Profile Area:** contiene i parametri per la comunicazione. Questi oggetti sono comuni a tutti i dispositivi.
- **Index 2000 – 5FFF Manufacturer Specific Profile Area**
- **Index 6000 – 9FFF Standardised Device Profile Area:** contiene tutti gli oggetti comuni ad una certa classe di dispositivi che possono essere letti e scritti attraverso la rete e ne descrivono i parametri e le funzionalità.
- Index A000 – FFFF riservati per usi futuri.

Accesso al drive

L'accesso al drive dalla rete CAN viene fatto usando principalmente due tipi di oggetti, che si differenziano per le caratteristiche di comunicazione:

- I **Service Data Object (SDO)** trasportano grandi quantità di dati, hanno bassa priorità sulla rete e sono di tipo asincrono.
- I **Process Data Object (PDO)** sono usati per migliorare i trasferimenti real-time di piccole quantità di dati ad alta priorità. Sono di tipo sincrono e asincrono. Occorre usare SDO per mappare nell'Object Dictionary i campi e i parametri di comunicazione di ogni PDO.

Importante è anche il **Synchronization Object (Sync)**, che viene inviato in broadcast a tutti i nodi della rete da un dispositivo master e che è usato per implementare sulla rete eventi di tipo sincrono.

Parametri del drive

Gli indirizzi dei registri e dei parametri sono riportati in una short forms all'inizio di questo manuale .

Per una descrizione accurata dei parametri del drive si faccia riferimento al Manuale **"Additional Informations"**. L'indirizzo dei parametri del drive può essere modificato senza avviso.

La modifica del valore dei parametri può essere effettuata tramite il tastierino del drive oppure utilizzando la rete CANopen. In quest'ultimo caso **sfrutta l'oggetto di 0x2000 Drive Parameters Setting**. Ogni modifica è attiva dopo un intervallo di tempo di circa 20 ms.

I cambiamenti così apportati vengono persi se il drive viene spento: se si vogliono rendere permanenti le variazioni occorre effettuare un salvataggio nella memoria non volatile del drive usando la procedura di illustrata in seguito.



Procedura per la lettura e la scrittura dei parametri

Per scrivere un parametro è necessario specificarne il valore nell'oggetto 0x2000sub2. Successivamente si scrive l'indirizzo del parametro nell'entità 0x2000sub1. Una volta completata l'operazione di scrittura il campo dell'indirizzo viene forzato a zero automaticamente.

Per leggere il valore di un parametro è necessario scriverne l'indirizzo nell'entità 0x2000sub3. Deve essere effettuata una rilettura di quest'oggetto finché non viene forzato a zero: una volta raggiunta questa condizione si può leggere il valore del parametro nell'entità 0x2000sub4. L'indirizzo di lettura viene, infatti, portato a zero solo quando l'operazione di lettura è andata a buon fine. Se si vuole saltare questa seconda fase, occorre aspettare almeno 3 ms tra la scrittura dell'indirizzo e la lettura del valore.



Salvataggio su flash

Per salvare permanentemente le variazioni dei parametri del drive ci si serve dell'oggetto 0x1010, in accordo con lo standard *DS 301*.

I parametri e gli oggetti vengono salvati nella memoria non volatile del drive e alla successiva accensione vengono caricati nella memoria ram in modo che l'utente ritrovi il drive con la stessa configurazione che aveva salvato.

È importante notare che il salvataggio è possibile solo se il motore non è abilitato (segnali IEN e/o TEN disabilitati). Se la procedura di salvataggio viene effettuata col motore abilitato verrà portata a termine solo quando verrà disabilitato il motore.

La procedura per salvare i parametri si riassume così:

- disabilitazione del motore (togliere coppia)
- scrittura nell'oggetto 0x1010 al sub-index 1 con la parola chiave 0x65766173, che permette di salvare in codice ASCII tutti i parametri.

Con questa operazione vengono salvati in maniera permanente:

- tutti i parametri del drive che si possono settare anche attraverso il tastierino del drive
- i seguenti oggetti CANopen illustrate in tabella sotto:

0x6060	modes of operation	0x3002	gear ratio divisor
0x6092	feed constant	0x6086	motion profile type
0x6065	following error window	0x3100	acceleration jerk
0x6066	following error time out	0x3101	deceleration jerk
0x6083	profile acceleration	0x60C2	interpolation time period
0x6084	profile deceleration	0x60FE	digital output ai subindex 1 e 2
0x6081	profile velocity	0x3201	ingresso ausiliario 1 a 32 bit
0x606D	velocity window	0x3202	ingresso ausiliario 2 a 32 bit
0x606E	velocity window time	0x3601	ingresso ausiliario 5 a 32 bit
0x606F	velocity threshold	0x3602	ingresso ausiliario 6a 32 bit
0x6070	velocity threshold time	0x3603	ingresso ausiliario 7 a 32

		bit	
0x6098	homing method	0x3604	ingresso ausiliario 8 a 32
		bit	
0x607C	home offset	0x3401	ingresso ausiliario 3 a 16
		bit	
0x6099sub1	homing speed 1	0x3402	ingresso ausiliario 4 a 16
		bit	
0x6099sub2	homing speed 2	0x3701	ingresso ausiliario 9 a 16
		bit	
0x609A	homing acceleration	0x3701	ingresso ausiliario 10 a 16
		bit	
0x3000	master encoder increments	0x3701	ingresso ausiliario 11 a 16
		bit	
0x3001	gear ratio numerator	0x3704	ingresso ausiliario 12 a 16
		bit	



Configurazione della rete

Per selezionare la rete CANopen occorre usare il tastierino del drive, selezionando **c9 = 5**.

Il numero di nodo del drive corrisponde al parametro **c7** che può assumere un qualsiasi valore compreso tra 1 e 99.

Il numero del nodo viene letto solo all'accensione del drive. Qualsiasi modifica viene quindi attuata solamente alla successiva accensione.

Il **bit-rate** per il CAN bus viene impostato col parametro **n1** che può assumere i valori tra 1 e 8 a cui corrispondono i seguenti bit-rate:

n1 = 1	=>	10 kBaud
n1 = 2	=>	20 kBaud
n1 = 3	=>	50 kBaud
n1 = 4	=>	125 kBaud
n1 = 5	=>	250 kBaud
n1 = 6	=>	500 kBaud
n1 = 7	=>	800 kBaud
n1 = 8	=>	1 MBaud

Il bit-rate viene letto solamente all'accensione del drive. Qualsiasi modifica viene quindi attuata solamente alla successiva accensione.



Oggetti CANopen

Oggetti implementati

Nelle tabelle che seguono, per ogni oggetto implementato, vengono indicati:

- index e sub-index
- nome
- la modalità di accesso
 - const = costante
 - ro = sola lettura
 - wo = sola scrittura
 - wr = lettura/scrittura
- la mappabilità su PDO
- la lunghezza in Byte
- il capitolo del documento *DSP 402* o del *DS 301* o di questo documento in cui l'oggetto viene descritto.

Oggetti obbligatori

0x1000	Device Type	const	Non mappabile	4 B	cap 9 DS 301
0x1001	Error Register	ro	Non mappabile	1 B	cap 9 DS 301
0x1003	Pre Defined Error Field				
sub 0	Number Of Error	rw	Non mappabile	4 B	cap 9 DS 301
sub 1	Standard Of Error Type	ro	Non mappabile	4 B	
0x1018	Identity Object				
sub 0	Number Of Entries	ro	Non mappabile	1 B	cap 9 DS 301
sub 1	Vendor Id	ro	Non mappabile	4 B	

Oggetti opzionali

La mappabilità esposta in questa tabella è valida dalla versione 3.04 in poi.

0x1005	Cob-Id Sync	rw	NO map.	4 B	cap 9 DS 301
0x1008	Manufacturer Device Name	const	NO map.	6 B	cap 9 DS 301
0x1009	Manufacturer Hardware Version	const	NO map.	6 B	cap 9 DS 301
0x100A	Manufacturer Software Version	const	NO map.	6 B	cap 9 DS 301
0x100C	Guard Time	rw	NO map.	2 B	cap 9 DS 301
0x100D	Life Time Factor	rw	NO map.	1 B	cap 9 DS 301
0x1010	Store Parameter Field				cap 9 DS 301
sub 0	Number Of Entries	ro	NO map.	4 B	
sub 1	Save All Parameters	rw	NO map.	4 B	
0x1014	Cob Id Emcy	rw	NO map.	4 B	cap 9 DS 301
0x1015	Inhibit Emcy	rw	NO map.	2 B	cap 9 DS 301
0x1017	Producer Heartbeat Time	rw	NO map.	2 B	cap 9 DS 301
0x1400	Receive PDO 1 Communication Parameter				cap 7 DS 301
sub 0	Number Of Entries	ro	NO map.	1 B	
sub 1	Cob-Id	rw	NO map.	4 B	
sub 2	Transmission Type	rw	NO map.	1 B	
0x1401	Receive PDO 2 Communication Parameter				cap 7 DS 301
sub 0	Number Of Entries	ro	NO map.	1 B	
sub 1	Cob-Id	rw	NO map.	4 B	
sub 2	Transmission Type	rw	NO map.	1 B	
0x1402	Receive PDO 3 Communication Parameter				cap 7 DS 301
sub 0	Number Of Entries	ro	NO map.	1 B	
sub 1	Cob-Id	rw	NO map.	4 B	
sub 2	Transmission Type	rw	NO map.	1 B	
0x1403	Receive PDO 4 Communication Parameter				cap 7 DS 301
sub 0	Number Of Entries	ro	NO map.	1 B	
sub 1	Cob-Id	rw	NO map.	4 B	
sub 2	Transmission Type	rw	NO map.	1 B	

0x1600	Receive PDO 2 Mapping Parameter					
sub 0	Number Of Entries	rw	NO map.	1 B		
sub 1	PDO Mapping Entry	rw	NO map.	4 B		
sub 2	PDO Mapping Entry	rw	NO map.	4 B		
sub 3	PDO Mapping Entry	rw	NO map.	4 B	cap 7 DS	
sub 4	PDO Mapping Entry	rw	NO map.	4 B	301	
sub 5	PDO Mapping Entry	rw	NO map.	4 B		
sub 6	PDO Mapping Entry	rw	NO map.	4 B		
sub 7	PDO Mapping Entry	rw	NO map.	4 B		
sub 8	PDO Mapping Entry	rw	NO map.	4 B		
0x1601	Receive PDO 2 Mapping Parameter					
sub 0	Number Of Entries	rw	NO map.	1 B		
sub 1	PDO Mapping Entry	rw	NO map.	4 B		
sub 2	PDO Mapping Entry	rw	NO map.	4 B		
sub 3	PDO Mapping Entry	rw	NO map.	4 B	cap 7 DS	
sub 4	PDO Mapping Entry	rw	NO map.	4 B	301	
sub 5	PDO Mapping Entry	rw	NO map.	4 B		
sub 6	PDO Mapping Entry	rw	NO map.	4 B		
sub 7	PDO Mapping Entry	rw	NO map.	4 B		
sub 8	PDO Mapping Entry	rw	NO map.	4 B		
0x1602	Receive PDO 3 Mapping Parameter					
sub 0	Number Of Entries	rw	NO map.	1 B		
sub 1	PDO Mapping Entry	rw	NO map.	4 B		
sub 2	PDO Mapping Entry	rw	NO map.	4 B		
sub 3	PDO Mapping Entry	rw	NO map.	4 B	cap 7 DS	
sub 4	PDO Mapping Entry	rw	NO map.	4 B	301	
sub 5	PDO Mapping Entry	rw	NO map.	4 B		
sub 6	PDO Mapping Entry	rw	NO map.	4 B		
sub 7	PDO Mapping Entry	rw	NO map.	4 B		
sub 8	PDO Mapping Entry	rw	NO map.	4 B		
0x1603	Receive PDO 4 Mapping Parameter					
sub 0	Number Of Entries	rw	NO map.	1 B		
sub 1	PDO Mapping Entry	rw	NO map.	4 B		
sub 2	PDO Mapping Entry	rw	NO map.	4 B		
sub 3	PDO Mapping Entry	rw	NO map.	4 B	cap 7 DS	
sub 4	PDO Mapping Entry	rw	NO map.	4 B	301	
sub 5	PDO Mapping Entry	rw	NO map.	4 B		
sub 6	PDO Mapping Entry	rw	NO map.	4 B		
sub 7	PDO Mapping Entry	rw	NO map.	4 B		
sub 8	PDO Mapping Entry	rw	NO map.	4 B		

0x1800	Transmit PDO 1 Communication Parameter				cap 7 DS 301
sub 0	Number Of Entries	ro	NO map.	1 B	
sub 1	Cob-Id	rw	NO map.	4 B	
sub 2	Transmission Type	rw	NO map.	1 B	
sub 3	Inhibit Time	rw	NO map.	2 B	
sub 4	Compatibility Entry	rw	NO map.	1 B	
sub 5	Event Timer	rw	NO map.	2 B	
0x1801	Transmit PDO 2 Communication Parameter				cap 7 DS 301
sub 0	Number Of Entries	ro	NO map.	1 B	
sub 1	Cob-Id	rw	NO map.	4 B	
sub 2	Transmission Type	rw	NO map.	1 B	
sub 3	Inhibit Time	rw	NO map.	2 B	
sub 4	Compatibility Entry	rw	NO map.	1 B	
sub 5	Event Timer	rw	NO map.	2 B	
0x1802	Transmit PDO 3 Communication Parameter				cap 7 DS 301
sub 0	Number Of Entries	ro	NO map.	1 B	
sub 1	Cob-Id	rw	NO map.	4 B	
sub 2	Transmission Type	rw	NO map.	1 B	
sub 3	Inhibit Time	rw	NO map.	2 B	
sub 4	Compatibility Entry	rw	NO map.	1 B	
sub 5	Event Timer	rw	NO map.	2 B	
0x1803	Transmit PDO 4 Communication Parameter				cap 7 DS 301
sub 0	Number Of Entries	ro	NO map.	1 B	
sub 1	Cob-Id	rw	NO map.	4 B	
sub 2	Transmission Type	rw	NO map.	1 B	
sub 3	Inhibit Time	rw	NO map.	2 B	
sub 4	Compatibility Entry	rw	NO map.	1 B	
sub 5	Event Timer	rw	NO map.	2 B	

0x1A00	Transmit PDO 1 Mapping Parameter					
sub 0	Number Of Entries	rw	NO map.	1 B		
sub 1	PDO Mapping Entry	rw	NO map.	4 B		
sub 2	PDO Mapping Entry	rw	NO map.	4 B		
sub 3	PDO Mapping Entry	rw	NO map.	4 B	cap 7 DS	
sub 4	PDO Mapping Entry	rw	NO map.	4 B	301	
sub 5	PDO Mapping Entry	rw	NO map.	4 B		
sub 6	PDO Mapping Entry	rw	NO map.	4 B		
sub 7	PDO Mapping Entry	rw	NO map.	4 B		
sub 8	PDO Mapping Entry	rw	NO map.	4 B		
0x1A01	Transmit PDO 2 Mapping Parameter					
sub 0	Number Of Entries	rw	NO map.	1 B		
sub 1	PDO Mapping Entry	rw	NO map.	4 B		
sub 2	PDO Mapping Entry	rw	NO map.	4 B		
sub 3	PDO Mapping Entry	rw	NO map.	4 B	cap 7 DS	
sub 4	PDO Mapping Entry	rw	NO map.	4 B	301	
sub 5	PDO Mapping Entry	rw	NO map.	4 B		
sub 6	PDO Mapping Entry	rw	NO map.	4 B		
sub 7	PDO Mapping Entry	rw	NO map.	4 B		
sub 8	PDO Mapping Entry	rw	NO map.	4 B		
0x1A02	Transmit PDO 3 Mapping Parameter					
sub 0	Number Of Entries	rw	NO map.	1 B		
sub 1	PDO Mapping Entry	rw	NO map.	4 B		
sub 2	PDO Mapping Entry	rw	NO map.	4 B		
sub 3	PDO Mapping Entry	rw	NO map.	4 B	cap 7 DS	
sub 4	PDO Mapping Entry	rw	NO map.	4 B	301	
sub 5	PDO Mapping Entry	rw	NO map.	4 B		
sub 6	PDO Mapping Entry	rw	NO map.	4 B		
sub 7	PDO Mapping Entry	rw	NO map.	4 B		
sub 8	PDO Mapping Entry	rw	NO map.	4 B		
0x1A03	Transmit PDO 4 Mapping Parameter					
sub 0	Number Of Entries	rw	NO map.	1 B		
sub 1	PDO Mapping Entry	rw	NO map.	4 B		
sub 2	PDO Mapping Entry	rw	NO map.	4 B		
sub 3	PDO Mapping Entry	rw	NO map.	4 B	cap 7 DS	
sub 4	PDO Mapping Entry	rw	NO map.	4 B	301	
sub 5	PDO Mapping Entry	rw	NO map.	4 B		
sub 6	PDO Mapping Entry	rw	NO map.	4 B		
sub 7	PDO Mapping Entry	rw	NO map.	4 B		
sub 8	PDO Mapping Entry	rw	NO map.	4 B		

0x6040	Controlword	rw	Mappabile	2 B	cap10 DSP402
0x6041	Statusword	ro	Mappabile	2 B	cap10 DSP402
0x6060	Modes Of Operation	wo	Mappabile	1 B	cap10 DSP402
0x6061	Modes Of Operation Display	ro	Mappabile	1 B	cap10 DSP402
0x6064	Position Actual Unit Value	ro	Mappabile	4 B	cap14 DSP402
0x6065	Following Error Window	rw	Mappabile	4 B	cap14 DSP402
0x6066	Following Error Time Out	rw	Mappabile	2 B	cap14 DSP402
0x6069	Velocity Sensor Actual Value	ro	Mappabile	4 B	cap16 DSP402
0x606B	Velocity Demand Value	ro	Mappabile	4 B	cap16 DSP402
0x606C	Velocity Actual Value	ro	Mappabile	4 B	cap16 DSP402
0x606D	Velocity Window	rw	Mappabile	2 B	cap16 DSP402
0x606E	Velocity Window Time	rw	Mappabile	2 B	cap16 DSP402
0x606F	Velocity Threshold	rw	Mappabile	2 B	cap16 DSP402
0x6070	Velocity Threshold Time	rw	Mappabile	2 B	cap16 DSP402
0x607A	Target Position	rw	Mappabile	4 B	cap12 DSP402
0x607C	Home Offset	rw	Mappabile	4 B	cap13 DSP402
0x6081	Profile Velocity	rw	Mappabile	4 B	cap12 DSP402
0x6083	Profile Acceleration	rw	Mappabile	4 B	cap12 DSP402
0x6084	Profile Deceleration	rw	Mappabile	4 B	cap12 DSP402
0x6086	Motion Profile Type	rw	Mappabile	2 B	cap12 DSP402
0x6092	Feed Constant				
sub 0	Number Of Entries	ro	NO map.	4 B	cap11 DSP402
sub 1	Feed	rw	NO map.	4 B	
sub 2	Shaft Revolutions	ro	NO map.	4 B	
0x6098	Homing Method	rw	Mappabile	1 B	cap13 DSP402
0x6099	Homing Speeds				
sub 0	Number Of Entries	ro	NO map.	4 B	cap13 DSP402
sub 1	Fast Homing Speed	rw	Mappabile	4 B	
sub 2	Slow Homing Speed	rw	Mappabile	4 B	
0x609A	Homing Acceleration	rw	Mappabile	4 B	cap13 DSP402
0x60C1	Interpolation Data Record				
sub 0	Number Of Entries	ro	NO map.	1 B	cap15 DSP402
sub 1	Position Set-Point	wo	Mappabile	4 B	
0x60C2	Interpolation Time Period.Numofentries				
sub 0	Number Of Entries	ro	NO map.	1 B	cap15 DSP402
sub 1	Time Period	rw	NO map.	1 B	
sub 2	Time Index	ro	NO map.	1 B	
0x60F8	Max Slippage	rw	Mappabile	4 B	cap16 DSP402
0x60FD	Digital Inputs	ro	Mappabile	4 B	cap 9 DSP402
0x60FE	Digital Outputs.Numofentries				
sub 0	Number Of Entries	ro	NO map.	1 B	cap 9 DSP402
sub 1	Physical Outputs	rw	Mappabile	4 B	
sub 2	Bitmask	rw	Mappabile	4 B	
0x60FF	Target Velocity	rw	Mappabile	4 B	cap16 DSP402
0x6502	Supported Drive Modes	ro	NO map.	4 B	cap 9 DSP402
0x6504	Drive Manufacturer	ro	NO map.	14 B	cap 9 DSP402
0x6505	Http Drive Catalog Address	ro	NO map.	18 B	cap 9 DSP402
0x67FF	Single Device Type	rw	NO map.	4 B	cap 7 DSP402

Oggetti "manufacturer specific"

0x2000	Drive Parameters Settings				
sub 0	Number Of Entries	ro	NO map.	1 B	cap 3.4
sub 1	Parameter Write Address	rw	NO map.	1 B	e
sub 2	Parameter Write Value	wo	NO map.	1 B	cap 4.2
sub 3	Parameter Read Address	rw	NO map.	1 B	
sub 4	Parameter Read Value	ro	NO map.	1 B	
0x2001	Latched User Position	rw	Mappabile	4 B	cap 0
0x2002	IP Set Point 16	wo	Mappabile	2 B	cap 4.2
0x2003	Position Actual Value 16	rw	Mappabile	2 B	cap 4.2
0x2004	Latched User Position 16	rw	Mappabile	2 B	cap 4.2
0x2005	Drive Variable Read				
sub 0	Number Of Entries	ro	NO map.	1 B	cap 4.2
sub 1	Variable Read Address	rw	Mappabile	2 B	
sub 2	Variable Read Value	ro	Mappabile	4 B	
0x2006	Drive Variable Write				
sub 0	Number Of Entries	ro	NO map.	1 B	cap 4.2
sub 1	Variable Write Address	rw	Mappabile	2 B	
sub 2	Variable Write Value	rw	Mappabile	4 B	
0x3000	Master Encoder Increments	rw	NO map.	2 B	cap 4.2
0x3001	Gear Ratio Numerator	rw	Mappabile	1 B	cap 4.2
0x3002	Gear Ratio Divisor	rw	Mappabile	1 B	cap 4.2
0x3100	Acceleration Jerk	rw	Mappabile	1 B	cap 4.2
0x3101	Deceleration Jerk	rw	Mappabile	1 B	cap 4.2
0x3201	Aux In 1	rw	Mappabile	4 B	cap 4.2
0x3202	Aux In 2	rw	Mappabile	4 B	cap 4.2
0x3301	Aux Out 1	ro	Mappabile	4 B	cap 4.2
0x3302	Aux Out 2	ro	Mappabile	4 B	cap 4.2
0x3401	Aux16 In 3	rw	Mappabile	2 B	cap 4.2
0x3402	Aux16 In 4	rw	Mappabile	2 B	cap 4.2
0x3501	Aux16 Out 3	ro	Mappabile	2 B	cap 4.2
0x3502	Aux16 Out 4	ro	Mappabile	2 B	cap 4.2
0x3601	Aux In 5	rw	Mappabile	4 B	cap 4.2
0x3602	Aux In 6	rw	Mappabile	4 B	cap 4.2
0x3603	Aux In 7	rw	Mappabile	4 B	cap 4.2
0x3604	Aux In 8	rw	Mappabile	4 B	cap 4.2
0x3701	Aux16 In 9	rw	Mappabile	2 B	cap 4.2
0x3702	Aux16 In 10	rw	Mappabile	2 B	cap 4.2
0x3703	Aux16 In 11	rw	Mappabile	2 B	cap 4.2
0x3704	Aux16 In 12	rw	Mappabile	2 B	cap 4.2



Descrizione degli oggetti del *Device Profile* e *manufacturer specific*

Tutti gli oggetti vengono descritti nel documento ufficiale *DSP 402* nei capitoli indicati nella tabella precedente.

In questa sezione si riportano solo le specifiche a cura del costruttore e la descrizione degli oggetti di tipo "*manufacturer specific*" (index compreso tra 0x2000 e 0x5FFF).

Descrizione degli oggetti del *Device Profile*

- **Oggetto 0x6040 *ControlWord***: le specifiche dei bit dall'11 al 15 sono a cura del costruttore:
 - bit 11: nella modalità asse elettrico dà il comando di start.
 - bit 12: quando questo bit viene forzato a 1, l'oggetto 0x6069 (*Velocity Sensor Actual Value*) ritorna la velocità dell'asse master. La velocità è espressa in incrementi per ms.
 - bit 14: una transizione da 0 a 1 di questo bit resetta il bit 14 della *StatusWord* (latch update)
- **Oggetto 0x6041 *StatusWord***: le specifiche dei bit 8, 14 e 15 sono a cura del costruttore:
 - bit 8: *Latching Input Status*. Questo bit indica lo stato degli ingressi di latching: ogni transizione da 0 a 1 del segnale provoca l'acquisizione della posizione. La posizione acquisita viene scritta nell'oggetto 0x2001 e in 0x2004 nella forma a 16 bit.
 - bit 14: *Latched Position Update*: se questo bit risulta a 1, significa che è stata acquisita una nuova latched position. Viene resettato usando il bit 14 della *ControlWord*. La posizione acquisita viene scritta nell'oggetto 0x2001 e 0x2004 nella forma a 16 bit.
 - bit 15: *Target Velocity Forced To Zero*: la velocità target viene forzata a 0 se questo bit vale 1. Ciò accade quando l'asse viene disabilitato (segnale TEN basso) o quando viene tolta l'abilitazione al movimento (segnale IEN basso).
- **Oggetto 0x6060 *Modes of Operation***: i valori tra -128 e -1 specificano i modi di funzionamento dell'azionamento definiti dal costruttore. Questi sono disponibili solo se l'hardware del drive li prevede.
I modi implementati nel drive sono:
 - 1 modo posizionario
 - 3 modo controllo di velocità
 - 6 modo zero asse (homing)
 - 7 modo interpolatore
 - 1 modo posizionario impulsi
 - 2 modo asse elettrico
 - 127 modo riferimento analogico velocità
- **Oggetto 0x6065 *Following Error Window***: impostando questo oggetto al valore 0xFFFFFFFF non viene effettuato il controllo sull'errore di inseguimento.
- **Oggetto 0x6069 *Velocity Sensor Actual Value***: per rendere semplice l'uso di quest'oggetto, la velocità è sempre riferita ad un sensore con 8192 impulsi per giro.
- **Oggetto 0x606D *Velocity Window***: se quest'oggetto vale 0xFFFF non viene effettuato il controllo sull'errore d'inseguimento della velocità.

- **Oggetto 0x606F Velocity Threshold:** se quest'oggetto vale 0xFFFF non viene fatto il controllo sulla soglia di velocità.
- **Oggetto 0x6086 Motion Profile Type:** le rampe sono
 - 0: profilo trapezoidale
 - 1: rampa a S
 - 2: rampa a seno
 ampiamente descritte nel manuale "**Additional Information**" nel paragrafo "**Errore. L'origine riferimento non è stata trovata.**".
- **Oggetto 0x6092 Feed Constant:** definisce quante unità utente per giro dell'albero motore. Una volta fissata l'unità utente (u) rimangono fissate anche le unità di misura della velocità (u/s) e dell'accelerazione (u/s/ms). Si noti che il sub-index 2 (*Shaft Revolutions*) è fissato ad 1. Nel sub-index 1 (*Feed*) occorre dunque specificare l'incremento della posizione in unità utente per ogni giro dell'albero motore. Ad esempio in un sistema costituito da un motore accoppiato direttamente ad una vite senza fine con passo di 1 mm, se si è deciso di utilizzare come unità di misura il μm , occorrerà fissare la Feed a 1000. Se poi tra l'albero motore e l'asse viene interposto un riduttore con rapporto 1/4, il valore da utilizzare per la Feed Constant sarà pari a 250. In questo caso, infatti, ad un giro dell'albero motore corrisponde $\frac{1}{4}$ di giro della vite e dunque un avanzamento dell'asse pari a 250 μm .
- **Oggetto 0x6098 Homing Method:** vedere il paragrafo "**Errore. L'origine riferimento non è stata trovata.**" di "**Additional Information**".
- **Oggetto 0x60C1 Interpolation Data Record:** al sub-index 1 è contenuta la posizione di set-point impiegata dall'interpolatore lineare.
- **Oggetto 0x60C2 Interpolation Time Period:** al sub-index 2 viene definito il periodo d'interpolazione espresso in ms.

Descrizione degli oggetti *manufacturer specific*

- **Oggetto 0x2000 Drive Parameters Settings:** quest'oggetto permette la scrittura e la lettura dei parametri. Per scrivere un parametro è necessario specificarne il valore nell'oggetto 0x2000sub2. Successivamente si scrive l'indirizzo del parametro nell'entità 0x2000sub1. Una volta completata l'operazione di scrittura il campo dell'indirizzo viene forzato a zero automaticamente. Per leggere il valore di un parametro è necessario scriverne l'indirizzo nell'entità 0x2000sub3. Deve essere effettuata una riletture di quest'oggetto finché non viene forzato a zero: una volta raggiunta questa condizione si può leggere il valore del parametro nell'entità 0x2000sub4. L'indirizzo di lettura viene, infatti, portato a zero solo quando l'operazione di lettura è andata a buon fine. Se si vuole saltare questa seconda fase, occorre aspettare almeno 3 ms tra la scrittura dell'indirizzo e la lettura del valore.
- **Oggetto 0x2001 Hardware Latched User Position:** quest'oggetto contiene la posizione, in unità utente, che è stata "catturata" in corrispondenza del fronte di salita del segnale d'ingresso nella corrispondente linea di input. Ogni aggiornamento viene segnalato attraverso la *StatusWord*.
- **Oggetto 0x2002 IP Position Set-Point 16 bit:** nella modalità interpolatore è possibile scrivere il set-point di posizione con una parola di 16 o 32 bit. Il set-point standard a 32 bit va scritto nell'oggetto 0x60C1 sub-index 1, mentre quello a 16 bit usa l'oggetto 0x2002.
- **Oggetto 0x2003 Position Actual Value in User Unit:** l'oggetto standard 0x6064 contiene la posizione attuale a 32 bit, mentre l'oggetto 0x2003

contiene la posizione attuale a 16 bit, che sono i 16 bit meno significativi dell'oggetto 0x6064.

- **Oggetto 0x2004 Hardware Latched User Position:** questo oggetto contiene i 16 bit meno significativi della posizione, in unità utente, agganciata in corrispondenza della transizione del segnale d'ingresso nella corrispondente linea di input. Ogni aggiornamento viene segnalato attraverso la *StatusWord*.
- **Oggetto 0x2005 Drive Variable Read:** quest'oggetto permette la lettura delle variabili del drive. In particolare se l'indirizzo è minore di 99 si accede ad un parametro del drive (come con l'oggetto 0x2000) e se l'indirizzo è compreso fra 0x8000 e 0x8FFF si leggono i valori dei registri del drive. Per leggere il valore di una variabile è necessario scriverne l'indirizzo nell'entità 0x2005sub1. Quest'oggetto deve essere riletto finché non viene forzato a zero: una volta raggiunta questa condizione si può leggere il valore del parametro nell'entità 0x2005sub2. L'indirizzo di lettura viene, infatti, portato a zero solo quando l'operazione di lettura è andata a buon fine. Se si vuole saltare questa seconda fase, occorre aspettare almeno 3 ms tra la scrittura dell'indirizzo e la lettura del valore.
- **Oggetto 0x2006 Drive Variable Write:** quest'oggetto permette la scrittura delle variabili del drive. In particolare se l'indirizzo è minore di 99 si accede ad un parametro del drive (come con l'oggetto 0x2000) e se l'indirizzo è compreso fra 0x8000 e 0x8FFF si scrivono i valori dei registri del drive. Per scrivere una variabile è necessario specificarne il valore nell'oggetto 0x2006sub2. Successivamente si scrive l'indirizzo del parametro nell'entità 0x2006sub1. Una volta completata l'operazione di scrittura il campo dell'indirizzo viene forzato a zero automaticamente.
- **Oggetto 0x3000 Master Encoder Increments Per Shaft Revolution:** quest'oggetto contiene il numero di impulsi per giro dell'encoder master. Questo oggetto viene usato nella modalità asse elettrico.
- **Oggetto 0x3001 Electronic Gearbox Ratio Numerator:** contiene il numeratore del rapporto di riduzione. Viene usato nella modalità asse elettrico.
- **Oggetto 0x3002 Electronic Gearbox Ratio Divisor:** contiene il denominatore del rapporto di riduzione. Viene usato nella modalità asse elettrico.
- **Oggetto 0x3100 Jerk for Acceleration Phase:** quest'oggetto contiene il jerk usato nelle operazioni di posizionamento durante la fase di accelerazione. Non tutti i profili di velocità usano questo parametro.
- **Oggetto 0x3101 Jerk for Deceleration Phase:** quest'oggetto contiene il jerk usato nelle operazioni di posizionamento durante la fase di decelerazione. Non tutti i profili di velocità usano questo parametro.
- **Oggetto 0x3201 Auxiliary Input nr. 1:** input a 32 bit usato per scopi generici
- **Oggetto 0x3202 Auxiliary Input nr. 2:** input a 32 bit usato per scopi generici
- **Oggetto 0x3301 Auxiliary Output nr. 1:** input a 32 bit usato per scopi generici
- **Oggetto 0x3302 Auxiliary Output nr. 2:** input a 32 bit usato per scopi generici
- **Oggetto 0x3401 Auxiliary Input nr. 3:** input a 16 bit usato per scopi generici
- **Oggetto 0x3402 Auxiliary Input nr. 4:** input a 16 bit usato per scopi generici

-
- **Oggetto 0x3501 Auxiliary Output nr. 3:** output a 16 bit usato per scopi generici
 - **Oggetto 0x3502 Auxiliary Output nr. 4:** output a 16 bit usato per scopi generici
 - **Oggetto 0x3601 Auxiliary Input nr. 5:** input a 32 bit usato per scopi generici
 - **Oggetto 0x3602 Auxiliary Input nr. 6:** input a 32 bit usato per scopi generici
 - **Oggetto 0x3603 Auxiliary Input nr. 7:** input a 32 bit usato per scopi generici
 - **Oggetto 0x3604 Auxiliary Input nr. 8:** input a 32 bit usato per scopi generici
 - **Oggetto 0x3701 Auxiliary Input nr. 9:** input a 16 bit usato per scopi generici
 - **Oggetto 0x3702 Auxiliary Input nr. 10:** input a 16 bit usato per scopi generici
 - **Oggetto 0x3703 Auxiliary Input nr. 11:** input a 16 bit usato per scopi generici
 - **Oggetto 0x3704 Auxiliary Input nr. 12:** input a 16 bit usato per scopi generici



Note sui servizi di Guard, SDO e PDO

Servizi di Guard

Il meccanismo di HeartBeat è attivo se l'oggetto 0x1017 (*ProducerHeartBeatTime*) è diverso da zero. In questa configurazione se viene inviato al drive un Remote Request con il COB-ID del Node Guarding (701 + NodeID), lo slave risponde inviando lo stato ma senza variare il toggle bit.

Il servizio di Node Guarding è attivo se l'oggetto 0x100C (*GuardTime*) è diverso da zero. Il drive risponde ad ogni Remote Request con COB-ID del Node Guarding (701 + NodeID), variando ogni volta il toggle-bit. La risposta viene inviata anche se il campo DLC nella richiesta è posto a zero.

Se inoltre è stato impostato l'oggetto 0x100D (*LifeTimeFactor*) diverso da zero, il drive monitora i tempi con cui il master invia i Remote Request e, se il tempo massimo ($LifeTimeFactor * GuardTime$) scade, viene alzato un flag per abilitare l'invio di un EMCY. Quando il master ricomincia ad inviare Remote Request il drive abbassa il flag e ricomincia a rispondere. L'EMCY non viene resettato.

Servizio SDO

Lo slave accetta ogni tipo di SDO, anche se il master non indica la lunghezza dei dati spediti. E' a cura di chi progetta il master inviare dati consistenti.

Durante una scrittura (SDO Download), se viene riscontrato un errore, l'oggetto non viene scritto e viene inviato un SDO di risposta che contiene il codice dell'errore occorso (Abort Code). I codici di errore si trovano nel capitolo 9 del documento *DS 301*.

L'oggetto 0x1003 (*PreDefinedErrorField*) al sub0 può essere scritto solo col valore 0. In questo caso viene posto a 0 anche l'oggetto all'index 0x1003 sub1.

La mappatura di un PDO deve rispettare i parametri di lunghezza, mappabilità, esistenza dell'oggetto da mappare. Nello stesso PDO è possibile mappare oggetti per una lunghezza massima di 8 Byte; la granularità è di 1 Byte. Per eliminare una mappatura bisogna agire con il NumOfEntries del PDO.

Servizio PDO

I parametri di comunicazione che definiscono il funzionamento dei **TPDO** sono contenuti negli oggetti 0x1800, 0x1801, 0x1802 e 0x1803.

- Se *Type* vale 254 o 255 e *EventTimer* è posto diverso da zero il TPDO è di tipo **asincrono** e viene inviato o quando il timer interno supera il tempo indicato in *EventTimer* o se si presenta una specifica condizione (evento) specificata dall'utente. In entrambi i casi deve essere passato un tempo superiore a *InhibitTime* dall'ultimo invio. L'invio dei TPDO asincroni non è correlato all'evento *Sync*.
- Se *Type* ha un valore compreso fra 0 e 240 il TPDO è di tipo **sincrono** e il suo invio è strettamente correlato all'eventp *Sync*.
 - Se *Type* è nullo il TPDO è di tipo **sincrono aciclico** e viene inviato al *Sync* successivo il verificarsi di uno specifico evento.
 - Se *Type* vale **n**, compreso tra 1 e 240, il TPDO è di tipo **sincrono ciclico** e viene spedito ogni **n** *Sync* ricevuti.

I parametri di comunicazione che definiscono il funzionamento degli **RPDO** sono contenuti negli oggetti 0x1400, 0x1401, 0x1402 e 0x1403.

- Se *Type* è uguale a 254 o 255 gli RPDO sono **asincroni** e vengono attuati appena ricevuti dallo slave.
- Se *Type* ha un valore compreso tra 0 e 240 gli RPDO sono **sincroni** e vengono attuati al *Sync* immediatamente successivo alla loro ricezione.



Esempi CANopen

Configurazione del drive per il funzionamento in CANopen

I principali parametri del drive da configurare sono:

- c7 => numero nodo
- c9 => 5 modo di funzionamento del drive in CANopen
- d8 => selezione del motore da utilizzare
- n1 => bit rate CAN: secondo la seguente tabella:

valore	1	2	3	4	5	6	7	8
bit rate (kBaud)	10	20	50	125	250	500	800	1000

Letture parametri drive

Scrittura dell'indirizzo.

Invio dell'indirizzo 1 corrispondente al parametro "d1"

```
write object 2000, sub 3, value: u8 1
```

Attesa completamento operazioni lettura.

Il completamento delle operazioni è indicato dal valore 0 dell'oggetto 2003, sub 3.

```
addr = 1
while (addr != 0) {
    read object 2000, sub 3, value: u8
}
```

Questa operazione può essere evitata se tra scrittura dell'indirizzo e lettura del valore, trascorrono almeno 3 ms.

Letture del valore.

Il valore del parametro richiesto è disponibile all'oggetto 2003, sub 4.

```
valore = read object 2000, sub 4, value: u8
```

Scrittura parametri drive

Scrittura del valore.

Invio del valore 40

```
write object 2000, sub 2, value: u8 40
```

Invio dell'indirizzo.

Invio dell'indirizzo 2 corrispondente al parametro "d2"

```
write object 2000, sub 1, value: u8 2.
```

Abilitazione e disabilitazione del servizio node guarding

Abilitazione del node guarding.

Guard time di 500 ms

write object 100C, sub 0, value: u16 500

Life time factor: 2

write object 100D, sub 0, value: u8 2

Disabilitazione del node guarding.

write object 100C, sub 0, value: u16 0

write object 100D, sub 0, value: u8 0

Disabilitazione del drive

Disabilitazione dell'asse.

Consente di togliere coppia all'asse

write object 6040, sub 0, value: u16 0x0002



Modo Homing

Parametri generali.

feed constant: 6000 unità utente per giro albero motore.

write object 6092, sub 1, value: u32 6000

Parametri homing.

Metodo: 34 (ricerca marker con direzione positiva).

write object 6098, sub 0, value: u8 34

Home offset: -1000. Dopo l'homing la posizione attuale varrà 1000.

write object 607C, sub 0, value: i32 -1000

Velocità ricerca marker: 10 r.p.m. (la velocità ricerca switch in questo caso non è usata).

write object 6099, sub 2, value: u32 1000

Modo Homing

write object 6060, sub 0, value: i8 0x06

Note:

- La velocità ricerca switch (oggetto 6099 sub 1) in questo caso non è utilizzata.
- Viene usata la homing_acceleration (oggetto 609A) di default.

Abilitazione del drive (messa in coppia).

Mette in coppia l'asse ed effettua il reset degli eventuali allarmi.

write object 6040, sub 0, value: u16 0x0000

write object 6040, sub 0, value: u16 0x0080

write object 6040, sub 0, value: u16 0x0000

write object 6040, sub 0, value: u16 0x0006

write object 6040, sub 0, value: u16 0x0007

write object 6040, sub 0, value: u16 0x000F

Nota: la sequenza di comandi illustrata è necessaria solamente se l'asse non è in coppia.

Comando di start attraverso *ControlWord*.

Lo svolgimento delle operazioni viene indicato attraverso i bit 12 e 13 della *statusword* (oggetto 6041), secondo lo standard *DSP 402*. Sinteticamente:

- entrambi i bit a zero: operazione non completata
- bit 12 a 1 e bit 13 a zero: operazione completata con successo
- bit 12 a zero e bit 13 a 1: homing interrotto
- entrambi i bit a 1: condizione proibita.

write object 6040, sub 0, value: u16 0x000F

write object 6040, sub 0, value: u16 0x001F



Modo posizionario

Parametri generali.

feed constant: 6000 unità utente per giro albero motore.

write object 6092, sub 1, value: u32 6000

Parametri posizionario.

Velocità: 200 r.p.m.

write object 6081, sub 0, value: u32 20000

Accelerazione: 100 ms per passare da 0 a 200 r.p.m.

write object 6083, sub 0, value: u32 200

Decelerazione: 100 ms per passare 200 r.p.m. a zero.

write object 6084, sub 0, value: u32 200

Modo posizionario.

write object 6060, sub 0, value: i8 0x01

Nota: Nell'esempio viene usato il `motion_profile_type` (oggetto 6086) di default, cioè il profilo di velocità trapezoidale.

Abilitazione del drive (messa in coppia).

Mette in coppia l'asse ed effettua il reset degli eventuali allarmi.

write object 6040, sub 0, value: u16 0x0000

write object 6040, sub 0, value: u16 0x0080

write object 6040, sub 0, value: u16 0x0000

write object 6040, sub 0, value: u16 0x0006

write object 6040, sub 0, value: u16 0x0007

write object 6040, sub 0, value: u16 0x000F

Nota: la sequenza di comandi illustrata è necessaria solamente se l'asse non è in coppia.

Movimentazioni con spostamenti incrementali.

Target position: 10 giri albero motore.

write object 607A, sub 0, value: i32 60000

Comando di start su `ControlWord`, spostamento incrementale.

write object 6040, sub 0, value: u16 0x004F

write object 6040, sub 0, value: u16 0x005F

Movimentazioni con posizioni assolute.

Target position: 5 giri albero motore.

write object 607A, sub 0, value: i32 30000

Comando di start su `ControlWord`, spostamento a posizione assoluta.

write object 6040, sub 0, value: u16 0x000F

write object 6040, sub 0, value: u16 0x001F



Modo controllo di velocità

Parametri generali.

Feed constant: 60 unità utente per giro albero motore (le velocità risultano in r.p.m.).

write object 6092, sub 1, value: u32 60

Parametri controllo velocità.

Accelerazione: 200 ms per passare da 0 a 1000 r.p.m.

write object 6083, sub 0, value: u32 5

Decelerazione: 200 ms per passare 1000 r.p.m. a zero.

write object 6084, sub 0, value: u32 5

Modo controllo velocità.

write object 6060, sub 0, value: i8 0x03

Abilitazione del drive (messa in coppia).

Mette in coppia l'asse ed effettua il reset degli eventuali allarmi.

write object 6040, sub 0, value: u16 0x0000

write object 6040, sub 0, value: u16 0x0080

write object 6040, sub 0, value: u16 0x0000

write object 6040, sub 0, value: u16 0x0006

write object 6040, sub 0, value: u16 0x0007

write object 6040, sub 0, value: u16 0x000F

Comando di movimentazione.

Target velocity: 1000 r.p.m.

write object 60FF, sub 0, value: i32 1000



Salvataggio

Disabilitazione dell'asse.

L'operazione di salvataggio richiede che venga tolta coppia all'asse

write object 6040, sub 0, value: u16 0x0002

Salvataggio parametri e oggetti CANopen.

Salvataggio degli oggetti CANopen per i quali è previsto il salvataggio (vedi tabella) e dei parametri (esempio: d1, d2, ...). Il salvataggio consiste nel copiare gli oggetti CANopen e i parametri in uso (cioè memorizzati nella ram), nella memoria flash.

Alle successive accensioni, per quanto riguarda i valori salvati, il drive è nelle stesse condizioni del momento del salvataggio.

write object 1010, sub 1, value: u32 0x65766173

Note: L'operazione di salvataggio può richiedere qualche secondo. La modifica di alcuni parametri diviene attiva solo a seguito di una nuova accensione del drive.

Attesa termine delle operazioni di salvataggio.

Drive in READY TO SWITCH ON

write object 6040, sub 0, value: u16 0x0006

Il salvataggio è completo quando il bit 0 di *StatusWord* vale 1, cioè quando il drive si trova nello stato di READY TO SWITCH ON.

```
statusword = 0
while (statusword & 0x0001 == 0) {
    statusword = read object 6041, sub 0, value u16
}
```



Modo interpolatore

Parametri generali.

feed constant: 6000 unità utente per giro albero motore.

```
write object 6092, sub 1, value: u32 6000
```

Parametri interpolatore.

Tempo interpolazione: 1 ms.

```
write object 60C2, sub 1, value: u8 1
```

Modo interpolatore.

```
write object 6060, sub 0, value: i8 0x07
```

Abilitazione position set-point e *ControlWord* via PDO sincrono.

Set-point di posizione e *ControlWord* verranno ciclicamente trasmessi verso il drive. A seguito di ogni *Sync*, il drive provvederà alla loro attuazione.

Position set-point è espresso in unità utente.

```
write object 1400, sub 2, value: u8 1
write object 1600, sub 0, value: u8 0
write object 1600, sub 1, value: u32 0x60C10120
write object 1600, sub 2, value: u32 0x60400010
write object 1600, sub 0, value: u8 2
```

Nota: per inviare lo stesso set-point di posizione e la stessa *ControlWord* a più nodi utilizzando lo stesso PDO, è sufficiente anteporre alla precedente lista di comandi il seguente (per ogni nodo):

```
write object 1400, sub 1, value: u32 513
```

il numero 513 è il COB ID di default per il Receive PDO 1 del nodo 1.

Abilitazione position feedback e *StatusWord* via PDO sincrono.

Position feedback e *StatusWord* verranno trasmessi dal drive ad ogni *Sync*.

Position feedback è espresso in unità utente.

```
write object 1800, sub 2, value: u8 1
write object 1A00, sub 0, value: u8 0
write object 1A00, sub 1, value: u32 0x60640020
write object 1A00, sub 1, value: u32 0x60410010
write object 1A00, sub 0, value: u8 2
```

Abilitazione alla comunicazione via PDO.

Start remote node (NMT service).

Abilitazione del drive (messa in coppia).

Mette in coppia l'asse ed effettua il reset degli eventuali allarmi.

```
write object 6040, sub 0, value: u16 0x0000
```

```
write object 6040, sub 0, value: u16 0x0080
write object 6040, sub 0, value: u16 0x0000
write object 6040, sub 0, value: u16 0x0006
write object 6040, sub 0, value: u16 0x0007
write object 6040, sub 0, value: u16 0x000F
```

Nota: la sequenza di comandi illustrata è necessaria solamente se l'asse non è in coppia.

Attivazione interpolatore.

A questo punto, il drive si aspetta di ricevere il set-point di posizione (object 60C1, sub 1) e Sync ogni ms. Il set-point di posizione è espresso in unità utente.

```
write object 6040, sub 0, value: u16 0x001F
```

Loop

Ad ogni ciclo verrà inviato verso il drive un PDO (contenente il set-point di posizione e la *ControlWord*) ed un messaggio di sincronismo (COB ID di default: 0x80).

Il set-point di posizione è espresso in unità utente.

```
loop
  write Sync
  write PDO 1, set-point, ControlWord
end
```



Esempio completo di asse gestito con interpolatore

Abilitazioni hardware.

Consenso alla messa in coppia dell'asse

```
segnale TEN a 24 Vdc
```

Consenso al movimento dell'asse

```
segnale IEN a 24 Vdc
```

Nota: questi segnali di consenso rimarranno attivi per tutta le fasi in cui il controllo dell'asse è gestito attraverso CANopen. Potranno ad esempio venire disattivati in situazioni di emergenza o di guasto del sistema di comunicazione CAN.

Configurazione del drive.

Parametri generali.

feed constant: 6000 unità utente per giro albero motore.

```
write object 6092, sub 1, value: u32 6000 (*)
```

(*): possibilità di salvataggio in memoria permanente (flash).

Parametri interpolatore.

Tempo interpolazione: 1 ms.

```
write object 60C2, sub 1, value: u8 1 (*)
```

(*): possibilità di salvataggio in memoria permanente (flash).

Parametri homing.

Metodo: 34 (ricerca marker con direzione positiva).

```
write object 6098, sub 0, value: u8 34 (*)
```

Home offset: -1000. Dopo l'homing la posizione attuale varrà 1000.

```
write object 607C, sub 0, value: i32 -1000 (*)
```

Velocità ricerca marker: 10 r.p.m. (la velocità ricerca switch non è usata).

```
write object 6099, sub 2, value: u32 1000 (*)
```

(*): possibilità di salvataggio in memoria permanente (flash).

Parametri drive (accessibili anche da tastierino).

Parametro "d1" guadagno proporzionale anello velocità (indirizzo = 1) a 80

write object 2000, sub 2, value: u8 80

write object 2000, sub 1, value: u8 1.

Parametro "d2" guadagno integrale anello velocità (indirizzo = 2) a 20

write object 2000, sub 2, value: u8 20

write object 2000, sub 1, value: u8 2.

Questo set di parametri ha la possibilità di salvataggio nella memoria flash del drive.

Abilitazione position set-point e ControlWord via PDO sincrono.

Set-point di posizione e *ControlWord* verranno ciclicamente trasmessi verso il drive.

A seguito di ogni *Sync*, il drive provvederà alla loro attuazione.

Position set-point è espresso in unità utente.

write object 1400, sub 2, value: u8 1

write object 1600, sub 0, value: u8 0

write object 1600, sub 1, value: u32 0x60C10120

write object 1600, sub 2, value: u32 0x60400010

write object 1600, sub 0, value: u8 2

Abilitazione position feedback e StatusWord via PDO sincrono.

Position feedback e *StatusWord* verranno trasmessi dal drive ad ogni *Sync*.

Position feedback è espresso in unità utente.

write object 1800, sub 2, value: u8 1

write object 1A00, sub 0, value: u8 0

write object 1A00, sub 1, value: u32 0x60640020

write object 1A00, sub 1, value: u32 0x60410010

write object 1A00, sub 0, value: u8 2

Abilitazione alla comunicazione via PDO.

Start remote node (NMT service).

Attivazione dell'invio ciclico del Sync

Abilitazione del drive (messa in coppia).

Mette in coppia l'asse ed effettua il reset degli eventuali allarmi.

write object 6040, sub 0, value: u16 0x0000

write object 6040, sub 0, value: u16 0x0080

write object 6040, sub 0, value: u16 0x0000

write object 6040, sub 0, value: u16 0x0006

write object 6040, sub 0, value: u16 0x0007

write object 6040, sub 0, value: u16 0x000F

Esecuzione dell'homing.

Modo Homing

write object 6060, sub 0, value: i8 0x06

Comando di Start attraverso *ControlWord*

write object 6040, sub 0, value: u16 0x001F

Attesa del completamento dell'operazione

Lo svolgimento dell'operazione viene indicato attraverso i bit 12 e 13 della *statusword* (oggetto 6041):

entrambi i bit a zero: operazione non completata

-
- bit 12 a 1 e bit 13 a zero: operazione completata con successo
 - bit 12 a zero e bit 13 a 1: homing interrotto
 - entrambi i bit a 1: condizione proibita.

Una volta che l'operazione è terminata, si porta a zero il bit di richiesta homing:

```
write object 6040, sub 0, value: u16 0x000F
```

L'asse rimane fermo in coppia.

Attivazione interpolatore: modo interpolatore

```
write object 6060, sub 0, value: i8 0x07
```

L'asse rimane fermo in coppia fino all'attivazione dell'interpolatore.

Attivazione interpolatore.

```
write object 6040, sub 0, value: u16 0x001F
```

A questo punto, il drive si aspetta di ricevere il set-point di posizione (object 60C1, sub 1) e Sync ogni ciclo.

Il set-point di posizione è espresso in unità utente.

Attivazione interpolatore: loop

Ad ogni ciclo verrà inviato verso il drive un PDO (contenente il set-point di posizione

e la *ControlWord*) ed un messaggio di sincronismo (COB ID di default: 0x80).

Il set-point di posizione è espresso in unità utente e viene acquisito dal drive a seguito

del Sync successivo.

```
loop
```

```
write Sync
```

```
write PDO 1, set-point, ControlWord
```

```
end
```

```
write object 6040, sub 0, value: u16 0x001F
```

Fermata "normale" dell'asse.

Ad asse fermo si disabilita l'interpolatore.

```
write object 6040, sub 0, value: u16 0x000F
```

Da questo momento non è più necessario che al drive giungano il Sync ed il PDO contenente il set-point (object 60C1, sub 1). L'asse viene tenuto fermo in coppia fino al seguente comando che libera l'asse (toglie cioè coppia al motore).

```
write object 6040, sub 0, value: u16 0x0002
```



I/O digitali

In questo capitolo si illustra la configurazione degli I/O (Input / output) digitali. La morsettiera CN2 è composta da 16 connessioni (pin) delle quali 12 sono configurate come ingressi e 4 come uscite. Tutti i segnali presenti nella morsettiera lavorano con livelli di tensione 0 - 24 Vdc, con lo zero in comune con l'alimentazione 24 Vdc del drive. In altre parole, per portare alto un ingresso, occorre fornire +24 Vdc rispetto al pin 2 di CN1 (zero alimentazione). Analogamente, per le uscite (di tipo PNP), un livello alto significa che il segnale viene portato a +24 Vdc rispetto al pin 2 di CN1 (zero alimentazione).

Ad alcuni pin del connettore a 16 poli della morsettiera CN2 corrisponde una funzione con un significato preciso, come mostrato nella tabella seguente.

digital_inputs in : ECO2/ECO4 standard / Tech & Evolution / uECO

Numero Pin di CN2	Numero Bit <i>digital_inputs</i>	Funzione
1	1	Start / Digital In
2	2	Digital In
3	3	Digital In
4	4	Digital In
5	5	Select / Digital In
6	6	Digital In
7	7	Digital In
8	8	Negative Limit Switch / Digital In
9	9	Positive Limit Switch / Digital In
10	11	Digital In
11	0	Position Latch / Digital In
14	10	HomeSwitch / Digital In

digital_inputs su MiniECO standard o miniECO PLUS

Nel miniECO sono disponibili soltanto alcuni ingressi ma anch'essi possono essere letti come input a cui è già stata assegnata la funzione specifica. Ad esempio se non si usano i limit switches del drive e quindi la procedura di zero assi non viene svolta dal Drive si possono utilizzare questi ingressi e leggerli tramite bus di campo. **E' necessario settare il C8 nella configurazione desiderata** vedi tabella sotto

Param. C8	CN1 Pin	Name	Numero Bit digital_inputs	Funzione
0	11	-- DIR	8	Negative Limit Switch / Digital In
	10	-- PULSE	9	Positive Limit Switch / Digital In
	5	-- IEN	10	HomeSwitch / Digital In
1	11	-- DIR	8	Negative Limit Switch / Digital In
	10	-- PULSE	9	Positive Limit Switch / Digital In
	5	-- IEN	0	Position Latch / Digital In
2	11	-- DIR	8	Negative Limit Switch / Digital In
	10	-- PULSE	9	Positive Limit Switch / Digital In
	5	-- IEN	1	Start Move/ Digital In
3	11	-- DIR	1	Start Move/ Digital In
	10	-- PULSE	10	HomeSwitch / Digital In
	5	-- IEN	0	Position Latch / Digital In
4	11	-- DIR	1	Start Move/ Digital In
	10	-- PULSE	8	Negative Limit Switch / Digital In
	5	-- IEN	--	Input Enable
5	11	-- DIR	1	Start Move/ Digital In
	10	-- PULSE	9	Positive Limit Switch / Digital In
	5	-- IEN	--	Input Enable
6 default	11	-- DIR	1	Start Move/ Digital In
	10	-- PULSE	10	HomeSwitch / Digital In
	5	-- IEN	--	Input Enable

digital_output in : ECO2/ECO4 standard / Tech & Evolution / uECO

Numero Pin di CN2	Numero Bit digital_output	Funzione
12	0	Digital Out
13	1	Digital Out
15	3	Digital Out
16	2	Brake / Digital Out

digital_output in : miniECO e miniECO PLUS

Numero Pin di CN2	Numero Bit digital_output	Funzione
12	0	
13	1	Forzatura Ventola
15	3	
16	2	Brake / Digital Out

Alcuni pin del connettore sono dedicati ai segnali di input e gli altri sono per i segnali di output. L'oggetto 0x60FE *Digital Output* si incarica di definire quali bit sono assegnati agli ingressi e quali alle uscite: al sub-index 2 è definita la Bitmask. Ognuno dei bit di questa parola controlla il funzionamento della corrispondente uscita. Se il bit è a 1 significa che il pin corrispondente è configurato come uscita, se è a 0, significa che il corrispondente pin non viene utilizzato.

Il "*Bit Number*" è usato negli oggetti

0x60FD *Digital Input*

0x60FE *Digital Output*

"*Negative limit switch*", "*Positive limit switch*" e "*Home switch*" sono gli ingressi relativi agli switch utilizzati per la ricerca dello zero assi secondo il metodo scelto (*Homing Method*)

L'ingresso "*Position Latch*" è usato per catturare la posizione dell'asse in corrispondenza del fronte di salita di questo segnale di cattura. La posizione così acquisita è disponibile nell'oggetto 0x2001 e nell'oggetto 0x2004, in cui è espressa con 16 bit. Un sistema di filtraggio software evita che disturbi di breve durata diano luogo a latching imprevisti. Per un corretto funzionamento del sistema, è quindi necessario che l'ingresso digitale rimanga alto per almeno 2 ms. Ogni evento di cattura viene segnalato attraverso un bit del registro *StatusWord*.

L'uscita "*Brake*" pilota il freno in alcuni modelli di azionamento.

Attraverso l'oggetto *Digital Input* è possibile leggere lo stato degli ingressi. Ad ogni pin di CN2 è, infatti, associato un bit del detto oggetto, secondo quanto indicato nella tabella. In alcuni casi, per uno stesso ingresso sono indicate due funzioni. Ciò significa che qualora non si utilizzi in nessun caso quella primaria, si può liberamente utilizzare l'ingresso come generico digital input. Se ad esempio, per le operazioni di homing viene utilizzato solo l'*home switch* (pin 14), gli ingressi relativi ai *limit switch positivo* (pin 9) e *negativo* (pin 8) possono essere impiegati come ingressi digitali.

E' anche possibile disporre di: alcune uscite digitali utilizzando l'oggetto *Digital Output*. Ad ogni pin di CN2 è associato un bit dell'oggetto, secondo quanto mostrato nella tabella. Anche in questo caso, per una stessa uscita possono essere indicate due funzioni. Se non si utilizza quella primaria, si può impiegare il pin come generico digital output. Al sub-index 2 dell'oggetto *Digital Output* è possibile configurare le uscite: ogni bit posto a 1 configura il pin corrispondente come uscita. Al sub-index 1 si controlla invece lo stato delle uscite. Ad ogni bit posto a 1 corrisponde un livello alto (24 Vdc) sul pin corrispondente.



Avvertenze

Versioni precedenti alla v. 3.00

Position feedback (oggetto 0x6064): questo oggetto viene aggiornato internamente ogni ms. L'istante in cui la posizione viene campionata ha dunque un'incertezza pari ad 1 ms.

Servizio Node Guarding: nel frame di richiesta remota del master verso gli slaves è necessario indicare DLC = 1 (lunghezza del campo dati della risposta pari ad 1 Byte). Il motivo risiede nel fatto che la risposta è processata direttamente dall'hardware.

Comunicazione sincrona: ogni PDO sincrono DEVE arrivare con un certo anticipo rispetto al messaggio di *Sync*.

Oggetti implementati: sono implementati solo due TPDO (con parametri negli oggetti 0x1400, 0x1600 e 0x1401, 0x1601) e due RPDO (con parametri negli oggetti 0x1400, 0x1600 e 0x1401, 0x1601).

Versioni precedenti alla v. 3.05

Oggetti implementati: il terzo ed il quarto TPDO sono utilizzabili solo nella modalità sincrona (non sono implementati gli oggetti 0x1802 sub3, 0x1802 sub4, 0x1802 sub5 e 0x1803 sub3, 0x1803 sub4, 0x1803 sub5)



Codici d'allarme

I Codici di allarme sono spiegati al capitolo **Allarmi** di questo manuale.

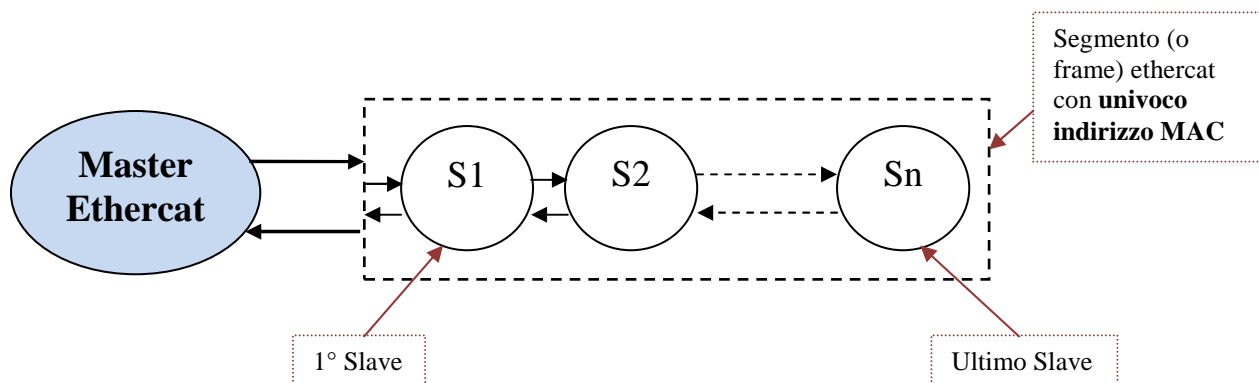


EtherCAT protocol

Il protocollo EtherCAT descrive una tecnologia Ethernet che massimizza la banda utilizzata nella comunicazione full-duplex. L'accesso al mezzo segue il principio master/slave in cui il master (di solito il sistema di controllo) invia pacchetti Ethernet ai nodi slave, i quali estraggono dati e inseriscono dati nello stesso frame. Siamo al livello 2 ISO / OSI (datalink).

Concetti Generali

Da un punto di vista Ethernet, un segmento ECAT è visto come un singolo dispositivo Ethernet che riceve ed invia frame Ethernet. Questo singolo dispositivo Ethernet può consistere in un certo numero di dispositivi slave EtherCAT. Ognuno di questi processa al volo il frame in arrivo, estraendo ed inserendo i dati rilevanti nel frame, ed inviando il frame al prossimo dispositivo slave. L'ultimo slave del segmento invia il frame completamente processato indietro, in modo che ritorni al primo slave e poi al master come frame di risposta. Vedi figura seguente:

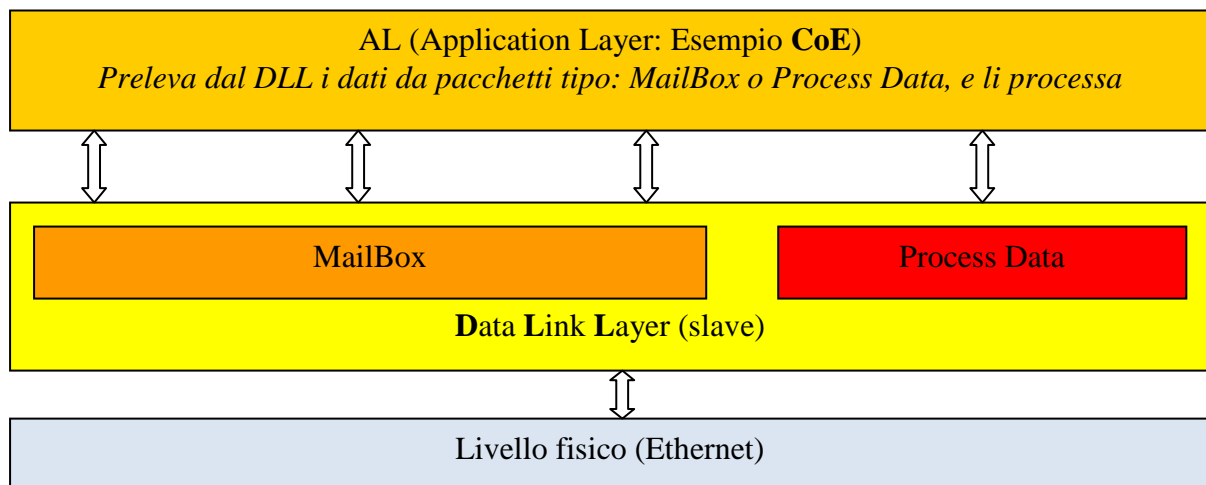


Viene utilizzato un canale full-duplex: entrambe le direzioni di comunicazione operano indipendentemente. Ogni periferica EtherCAT Slave è descritta attraverso un file .xml, che dovrà essere importato nel Master EtherCAT.

Topologia

La topologia *tipica* è ad anello aperto. Dal lato aperto si trova il master, che invia il frame al primo slave della catena. L'ultimo slave della catena rimanda indietro il pacchetto finché non torna al master. È possibile inserire ramificazioni in modo da realizzare strutture ad albero.

DLL (Data Link Layer)



Il DLL ha il compito di calcolare, comparare e generare la sequenza di frame-check, quindi provvedere all'estrapolazione o inserzione nel frame dei dati. I dati sono disponibili verso Application Layer da locazioni di memoria predefinite: attraverso il servizio di Mailbox o la sezione Process Data.

Più nodi (slave) possono essere indirizzati individualmente attraverso un singolo frame Ethernet che trasporta più telegrammi ECAT. Di conseguenza, un gruppo di nodi slave, può avere un solo indirizzo MAC.

Il DLL fornisce un supporto *time-critical* (in una determinata finestra di tempo devono essere terminate le operazioni richieste con un definito livello di certezza) per la comunicazione tra dispositivi. Il fallimento di un'operazione compromette la riuscita di tutto il processo.

Il DLL fornisce servizi al livello superiore: Application Layer (AL) .

Error Detection

Il master e gli slave testano il campo FCS. Poiché ogni nodo varia il contenuto del frame durante il trasferimento, è necessario che il FCS venga ricalcolato da ogni nodo. Se uno slave riscontra un errore nel checksum, il FCS non viene corretto ma viene incrementato un contatore di errori in modo che il master possa determinare con precisione la posizione dell'errore.



Parametri e dati di processo

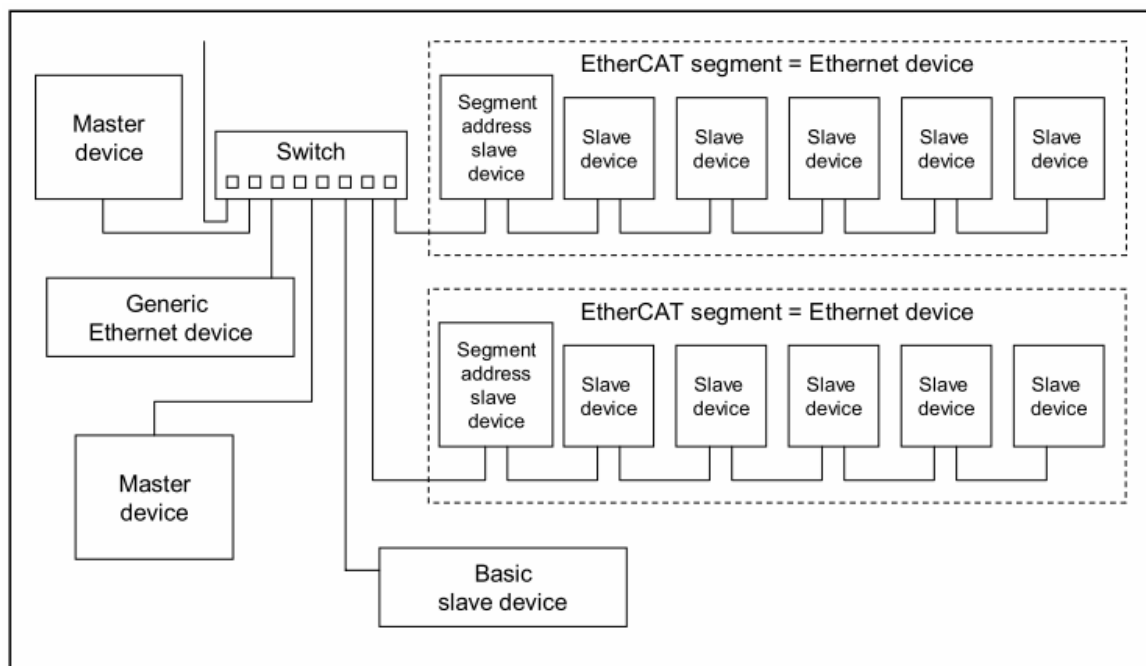
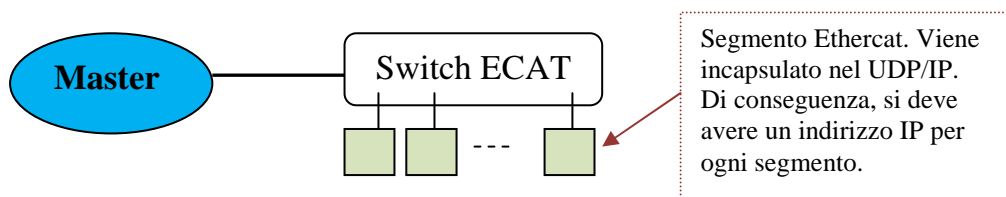
Il sistema di comunicazione deve saper conciliare le differenti richieste per quanto riguarda le caratteristiche della trasmissione dei dati.

- ✓ I *parametri* vengono inviati in modo aciclico ed in larga quantità senza particolari tempistiche.
- ✓ I *dati di diagnostica* sono anch'essi inviati in modo aciclico ma hanno necessità più stringenti in fatto di tempo.
- ✓ I *dati di processo* sono invece i più esigenti anche se sono di piccola entità: la tempistica è determinante, sono inviati ciclicamente.

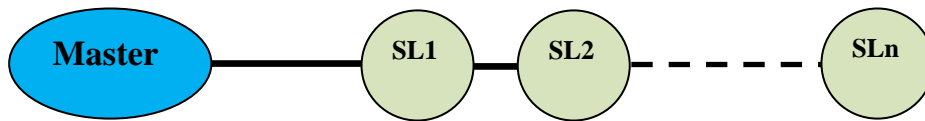


Collegamenti: open & direct mode

Nel *modo Open* uno o più *segmenti* ECAT possono essere connessi attraverso uno switch. Il primo slave di un segmento ECAT ha un MAC address che rappresenta l'intero segmento. Il frame che viene spedito è di tipo UDP/IP : il primo dispositivo del segmento adatta il frame Ethernet al tipo EtherCAT.



Nel *modo diretto* un solo segmento ECAT viene collegato direttamente alla porta Ethernet standard del master.



La topologia logica della rete è ad anello aperto. Dal lato aperto il master invia i frame direttamente e riceve il frame di risposta. *Ogni frame passa da uno slave al successivo; l'ultimo dispositivo spedisce indietro, attraverso tutti gli altri, il frame al master.*

Il frame viene processato Byte per Byte al volo dai dispositivi slave in accordo con la sequenza fisica all'interno della struttura ad anello. Il dispositivo legge dal frame i comandi che sono destinati a lui e li attua, mentre il frame, che viene ritardato di un tempo costante tipicamente minore di 1 ms, passa verso il prossimo dispositivo. L'estrazione e l'inserzione dei dati avviene nel DLL. È quindi indipendente dal tempo di risposta di ogni uP collegato.

Derivazioni nella topologia sono permesse se non rompono l'anello logico.

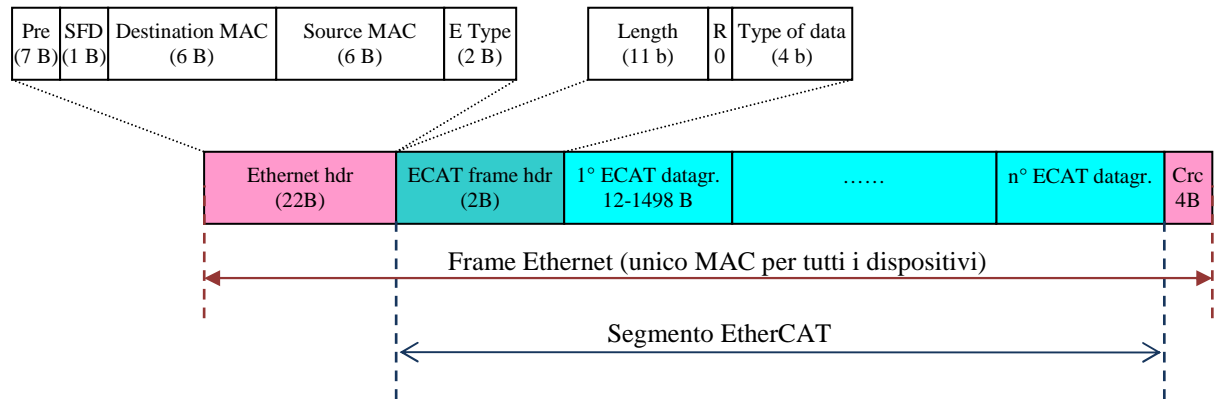
Nota: Il tipico collegamento per i drive è il modo diretto, in quanto l'azionamento permette un agevole collegamento ingresso-uscita.



Struttura generale del frame

Un frame è composto da uno o più datagrammi ECAT, ognuno dei quali è associato ad un singolo dispositivo ethercat.

Il frame sarà quindi composto dagli N datagrammi ECAT.



➤ Ethernet Header (22 bytes)

- *Pre* (preambolo): hanno valore 10101010
- *SFD* (Start Frame Delimiter), di 1 byte: questo byte ha valore 10101011
- *Destination MAC*: indirizzo di destinazione
- *Source MAC*: indirizzo sorgente
- *EtherType* (campo tipo): indica il tipo di protocollo del livello di rete in uso durante la trasmissione. 0x88A4 (ECAT)
- *CRC*: permette di rilevare se sono presenti errori di trasmissione

➤ ECAT frame Header (2 byte)

So che questo è un Header di ECAT (incapsulato in un frame ethernet) in seguito al valore di EtherType 0x88A4.

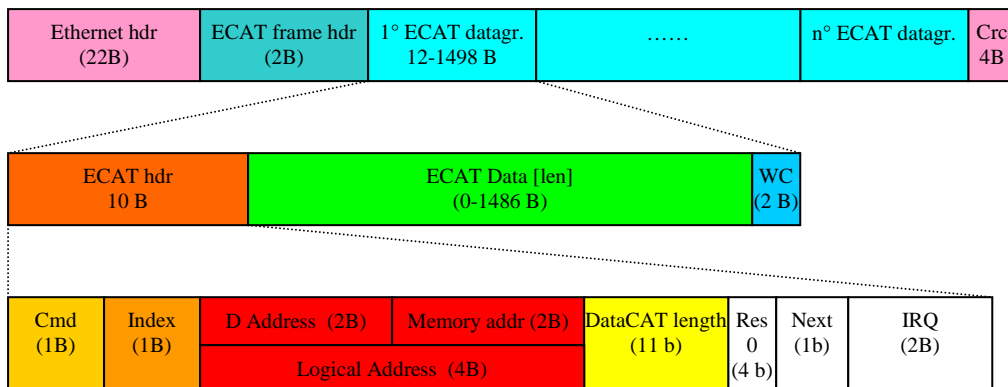
- *Length (11 bits)*: lunghezza di questo frame ECAT. Lunghezza massima dell'intero segmento ECAT è di 1498 bytes. Ci possono essere N frame ECAT, ma la loro dimensione complessiva deve essere minore o uguale di 1498 bytes.
- 1 bit riservato
- *Type of data (4bits)*, tipo di "dato" da aspettarsi nel pacchetto seguente:
 - 0x01: ECAT datagramms (**N.B: unica tipologia valutata**)
 - 0x04: network variables
 - 0x05: mailbox

Nota: per maggiori informazioni riferirsi al documento ETG.1000.4 consorzio ETG.



Composizione frame ECAT

Type of data = 0x01: ECAT datagrams



Ogni ECAT datagram termina con il campo *WC* (*working counter*): ogni slave coinvolto nella gestione del datagram (e quindi che viene indirizzato e ha libero accesso alla memoria indirizzata) *incrementa* questo campo. Il master ha il compito di comparare il WC atteso con quello della risposta, per stabilire se il messaggio è stato processato correttamente da *tutti* gli slave.

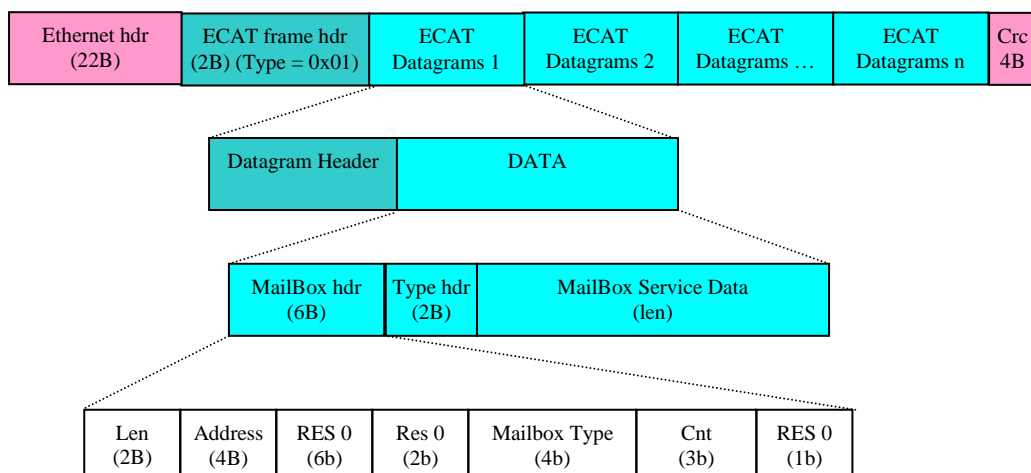
Ci possono essere *n* datagram ECAT, purchè la loro dimensione totale sia ≤ 1498 byte.

Campo *Next*: Mi dice se ci sono altri successivi datagram ECAT rispetto all'attuale, in pratica se questo è l'ultimo datagram o meno.

Protocollo Mailbox

Il protocollo tipico delle mailbox, può essere usato all'interno di un frame ethercat (type = 0x01) per la gestione veloce dei device ethercat. Così facendo ad ogni device corrisponda un datagram ethercat, anche a livello di dati aciclici (es. SDO).

In particolare:



La tipologia del ECAT Datagrams viene selezionata a seconda del comando contenuto nel campo CMD del ECAT Frame Header.

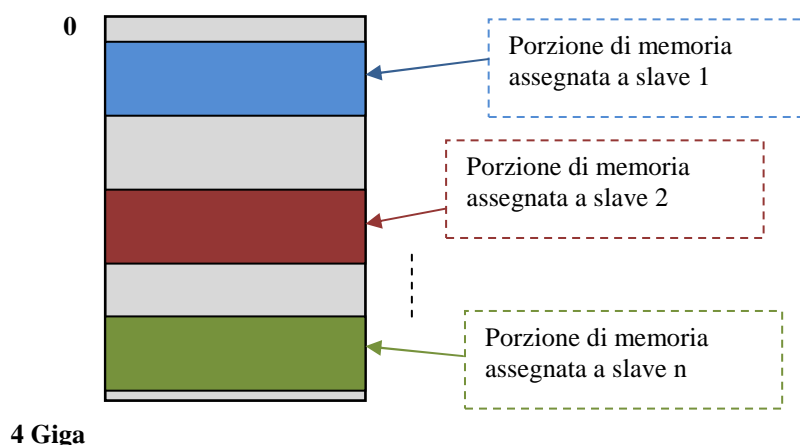
In particolare ogni lettura o scrittura o lettura/scrittura che implica un indirizzamento per nodo od indirizzo incrementale (auto increments), può avere nel rispettivo campo Data del datagram il protocollo Mailbox.
Vedi documento ETG 1000.4 da pagina 31 per ulteriori dettagli.

FMMU

FMMU sta per **F**ieldbus **M**emory **M**anagement **U**nits (Unità di gestione della memoria del bus di campo). Il master può vedere tutto il ramo di slave come se fosse una unica memoria logica (di max 4 Giga), le FMMU di ogni singolo slave convertono gli indirizzi logici in indirizzi fisici interni allo stesso slave.

Ogni dispositivo può contenere più FMMU, ognuna delle quali mappa un indirizzo logico (che arriva da datagram ECAT corrispondente) in un indirizzo fisico. Le FMMU supportano una mappatura bit per bit. Una FMMU consiste al massimo di 16 unità. Ogni unità descrive una traslazione di memoria tra quella logica e quella fisica del dispositivo.

Ad esempio, avendo 3 dispositivi slave, la situazione può essere schematizzata come segue:



La configurazione delle celle FMMU è compito del master e viene inviata allo slave nella fase di start-up. Per ogni FMMU occorre configurare lo start address logico, lo start address fisico, la lunghezza in bit e la direzione del segmento (in o out).

Quando un dispositivo riceve un messaggio indirizzato logicamente, lo slave cerca se una delle sue FMMU ha l'indirizzo fisico corrispondente. Se coincide, il dispositivo inserisce (scrive) i dati nella relativa posizione nel datagram (in) oppure estrae (legge) i dati dalla relativa posizione del datagram (out).

I dispositivi slave possono essere principalmente di due tipi: Full slave, che supportano tutti i tipi d'indirizzamento ed i Basic slave che non supportano il modo d'indirizzamento logico.

Nel caso , si ha un dispositivo full slave, in cui vengono utilizzati 3 FMMU. Due per i process data (coe: pdo) ed una per i mailbox (coe: sdo).



File descrittore .XML

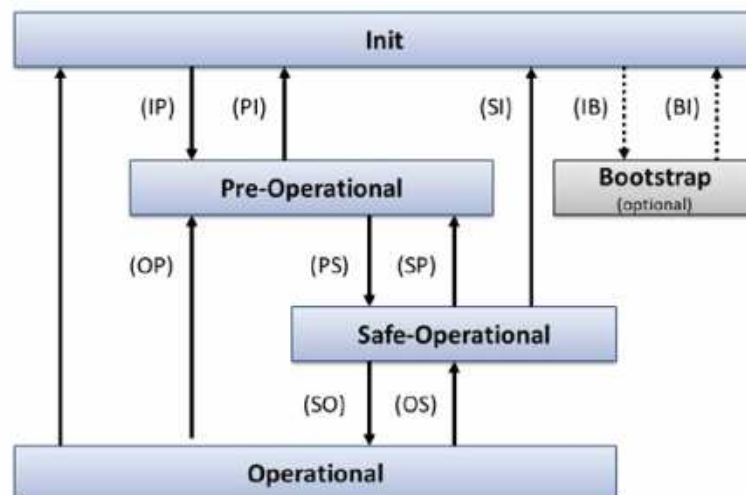
In ethercat, ogni dispositivo slave deve avere un file descrittore (analogo al file .eds del canopen). In modo da permettere al Master di poterlo configurare.

Nel caso , il file si chiama: *DriveECAT_vxxx.xml* dove xxx è la versione del file.

Al suo interno è definita una mappatura di default, comunque riconfigurabile dal Master.

Stati macchina

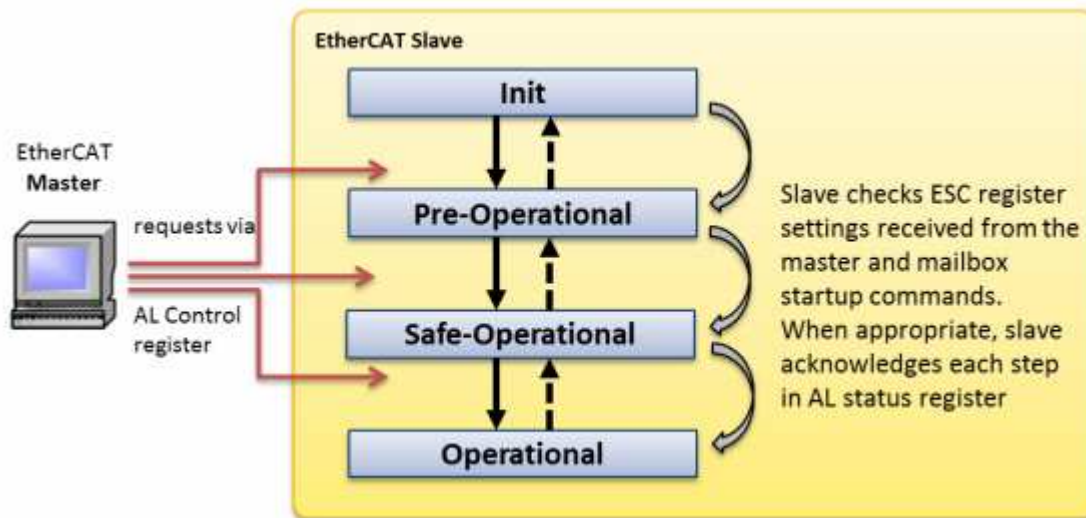
La richiesta di cambiamento di stato viene inviata dal Master, il cambiamento effettivo avviene solo nel caso in cui la configurazione richiesta sia valida. A seconda dello stato eseguito dallo slave, saranno disponibili o meno delle funzionalità di sistema.



La descrizione degli stati è mostrata in tabella seguente:

Stato	Descrizione
INIT	<i>Init state.</i> Nessuna comunicazione con AL, il Master ha accesso solo ai DL - register information.
PREOP	<i>Pre-operational state.</i> Disponibile la comunicazione via Mailbox (SDO), non è disponibile la comunicazione via process data (PDO).
SAFEOP	<i>Safe operational state.</i> Disponibile la comunicazione via Mailbox (SDO), disponibile <i>solo per gli input</i> la comunicazione via process data (PDO).
OP	<i>Operational mode state.</i> Disponibile sia comunicazione via Mailbox, sia via process data.
BOOT	<i>Boot state (opzionale).</i> Nota: non presente nel drive.

La transizione fra gli stati macchina avviene *solo* su richiesta del Master, in particolare:



Transizioni	Descrizione
I → P	<p>Il Master legge il VendorID, ProductCode e il RevisionNumber dalla EEPROM. Quindi configura:</p> <ul style="list-style-type: none"> - DL Control Registers del ESC (ET1100 bechhoff) - SyncManager per la comunicazione via mailbox - Inizializza il DC Clock se utilizzato <p>A questo punto il Master richiede il passaggio allo stato di Pre Operational ed aspetta la conferma dallo slave.</p>
P → S	<p>Il Master configura i parametri dello slave via mailbox (SDO per CoE), come ad esempio la mappatura dei process data (PDO per CoE), i SyncManager o le FMMU.</p> <p>Successivamente il Master richiede il passaggio allo stato di Safe Operational ed aspetta la conferma dallo slave.</p>
S → O	<p>Il Master richiede il passaggio al Operational State ed attende conferma dallo slave.</p>
Error Init Error PreOp Error SafeOp	<p>Errori nella configurazione (es sui DC, FMMU, SyncManager etc ..)</p>

Nota: per maggiori informazioni riferirsi al documento ETG.1000.6 consorzio ETG.



EtherCAT

Sui drive è stato implementato il protocollo COE (**C**anopen **O**ver **E**thercat). Tale protocollo permette di mantenere i concetti di SDO e PDO tipici del protocollo canopen, in modo da rendere agevole il passaggio al protocollo EtherCAT.

Il protocollo a livello applicativo è quindi il CANOpen ds402.

Per i PDO viene usato l'indirizzamento e la gestione delle FMMU, mentre per gli SDO viene utilizzato il protocollo Mailbox incapsulato in un ecat datagrams.

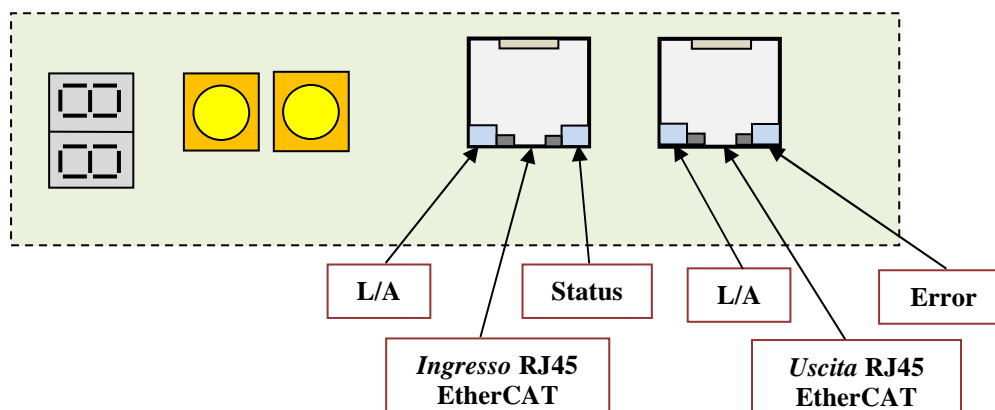
Il Vendor ID per il consorzio ETG è: 0x0000078F

Link ETG: http://www.ethercat.org/en/vendor_id_list.html

Connessioni e Leds Drive EtherCAT

L'azionamento permette un collegamento di tipo ingresso - uscita direct mode, in modo da poterlo collegare in ogni punto della rete.

In particolare, guardando l'azionamento frontalmente:



Significato dei leds:

Tipo	Colore Led	Descrizione
L/A	Verde	Link Active rete ethernet.
Status	Verde	Segnalazione dello stato ethercat. Se sempre acceso ethercat è in operational mode. Per maggiori informazioni riferirsi alla documentazione ufficiale ETG-1300.
Error	Rosso	Segnalazione warning/allarmi in ethercat. La frequenza di lampeggio è funzione della tipologia di allarme. Per maggiori informazioni riferirsi alla documentazione ufficiale ETG-1300.



Parametrizzazione EtherCAT per drive

Parametro drive **C9 = 11** (modalità ethercat)

Non è necessario impostare il nodo del drive (parametro C7), in quanto è lo stesso ethercat che definisce l'indirizzamento del drive a seconda del collegamento topologico all'interno della rete.

Nota: per una trasmissione il più possibile immune ai disturbi, è consigliato l'utilizzo di cavi *ethernet almeno di categoria 5e, ideale categoria 6*.

PDO

Attivi in totale 8 PDO di cui 4 in trasmissione e 4 in ricezione:

- *RxPDO*,
 - oggetto 0x1600
 - Control Word (2 bytes, oggetto 0x6040 sub0)
 - Set-Point Position (4 bytes, oggetto 0x60C1 sub1)
 - oggetto 0x1601
 - Control Word (2 bytes, oggetto 0x6040 sub0)
 - Modes of Operation (2 bytes, oggetto 0x6060 sub0)
 - oggetto 0x1602
 - Control Word (2 bytes, oggetto 0x6040 sub0)
 - Target Position (4 bytes, oggetto 0x607A sub0)
 - oggetto 0x1603
 - Control Word (2 bytes, oggetto 0x6040 sub0)
 - Target Velocity (4 bytes, oggetto 0x60FF sub0)

- *TxPDO*,
 - oggetto 0x1A00
 - Status Word (2 bytes, oggetto 0x6041 sub0)
 - oggetto 0x1A01
 - Status Word (2 bytes, oggetto 0x6041 sub0)
 - Modes of Operation Display (2 bytes, oggetto 0x6061 sub0)
 - oggetto 0x1A02
 - Status Word (2 bytes, oggetto 0x6041 sub0)
 - Actual Position (4 bytes, oggetto 0x6064 sub0)
 - oggetto 0x1A03
 - Status Word (2 bytes, oggetto 0x6041 sub0)
 - Actual Velocity (4 bytes, oggetto 0x606C sub0)

Il Master può riconfigurare i PDO, attraverso gli oggetti 0x1C12 (RxPDO Assign) e 0x1C13 (TxPDO Assign), e/o cambiare direttamente le variabili associate modificando gli oggetti 0x1A00 – 0x1A03 e/o 0x1600 – 0x1603.

Nota importante: come *default*, attraverso gli oggetti 0x1C12 e 0x1C13, sono selezionati gli oggetti 0x1600 e 0x1A02 per l'interpolazione in posizione (modes of operation = 7).

Riferirsi alle caratteristiche del livello applicativo CANOpen, presenti nel capitolo 4, per maggiori informazioni.



Utilizzo degli Slots

Nel caso il Master lo supporti, il file .xml permette la selezione di configurazioni standard di pdo, in relazione al modo operativo desiderato sul drive. Questi raggruppamenti di pdo sono detti slots.

In particolare, sono disponibili i seguenti slot mutualmente selezionabili:

Module	Description
<input checked="" type="radio"/> Position Mode	Position Mode
<input checked="" type="radio"/> Interpolated Mode	Profile Interpolated Mode
<input checked="" type="radio"/> Velocity Mode	Profile Velocity Mode

Dove:

- *Position Mode*: modo posizionario punto a punto
TxPDO:
 oggetto 0x1A02
 Status Word (2 bytes, oggetto 0x6041 sub0)
 Actual Position (4 bytes, oggetto 0x6064 sub0)
RxPDO
 oggetto 0x1602
 Control Word (2 bytes, oggetto 0x6040 sub0)
 Target Position (4 bytes, oggetto 0x607A sub0)

- *Interpolated Mode*: modo posizionario in interpolata
TxPDO:
 oggetto 0x1A02
 Status Word (2 bytes, oggetto 0x6041 sub0)
 Actual Position (4 bytes, oggetto 0x6064 sub0)
RxPDO:
 oggetto 0x1600
 Control Word (2 bytes, oggetto 0x6040 sub0)
 Set-Point Position (4 bytes, oggetto 0x60C1 sub1)

- *Velocity Mode*: modo velocità
TxPDO
 oggetto 0x1A03
 Status Word (2 bytes, oggetto 0x6041 sub0)
 Actual Velocity (4 bytes, oggetto 0x606C sub0)
RxPDO
 oggetto 0x1603
 Control Word (2 bytes, oggetto 0x6040 sub0)
 Target Velocity (4 bytes, oggetto 0x60FF sub0)

Ogni modalità operative **deve** essere selezionata attraverso l'oggetto *0x6060 mode of operation*. In particolare:

- 0x6060 per *Position Mode* = 1
- 0x6060 per *Interpolated Mode* = 7
- 0x6060 per *Velocity Mode* = 3
- 0x6060 per *Cyclic Synchronous Position Mode (csp)* = 8

La scrittura del modo operativo può essere eseguita via sdo download *una volta sola* allo startup del sistema.

In alternativa, si possono ulteriormente mappare gli oggetti 0x1601 e 0x1A01 per controllare le modalità operative via pdo, gestendo gli oggetti 0x6060 e 0x6061.

La prima strada è da preferire in quanto non viene occupata banda passante per una funzione che può benissimo essere eseguita solo all'avvio.

Nota: l'oggetto *mode of operation* è utilizzato solo per impostare il modo operativo, di conseguenza per verificare il modo operativo corrente, occorre leggere l'oggetto *0x6061 mode of operation display*.

SDO: tipologia e dizionario oggetti

Le operazioni SDO implementate sono i seguenti:

- SDO Upload (come nel canopen)
- SDO Download (come nel canopen)
- SDO Info support (tipico di ethercat)

Il dizionario oggetti, accessibile via SDO, è lo stesso del CANOpen, reperibile in questa documentazione.

L'unica differenza sta nell'aggiunta dell'oggetto *0x603F Error Codes*, contenente la codifica degli errori secondo specifica ethercat.

L'accesso agli oggetti può essere eseguito attraverso le operazioni di sdo upload e sdo download, secondo le specifiche proprietà di lettura / scrittura di ogni singolo oggetto.

Nota: utilizzando la funzione *SDO Info support* è possibile visualizzare l'elenco aggiornato degli oggetti implementati nel dizionario direttamente dal master.

Mappatura dinamica PDO

La mappatura dinamica è possibile attraverso gli oggetti 0x1C12 (RxPDO Assign) e 0x1C13 (TxPDO Assign). In questo modo posso selezionare una qualunque configurazione possibile fra i pdo di default (oggetti 0x1600 – 0x1603, 0x1A00 – 0x1A03).

Ad esempio, si può selezionare solo gli oggetti 0x1600 e 0x1A02 per l'interpolazione in posizione (slot Interpolated Mode, modes of operation = 7).

Nel caso si vogliano configurare delle combinazioni diverse da quelle di default, è comunque possibile, agendo prima direttamente sugli oggetti 0x1600 – 0x1603 o 0x1A00 – 0x1A03. Ed in seguito su gli oggetti 0x1C12 e 0x1C13 per selezionare i TxPDO e RxPDO desiderati.



Lettura / Scrittura parametri del drive

Utilizzando gli oggetti 0x2005 e 0x2006 è possibile accedere in lettura e/o scrittura ai parametri specifici del drive.

L'accesso a questi oggetti è possibile via SDO Upload / Download.

In particolare:

Lettura di un parametro (SDO Upload):

Si utilizza l'oggetto composto 0x2005. Esso possiede due subindex, di cui:

- Subindex 1: Variable Read Address
- Subindex 2: Variable Read Value

L'indirizzo viene impostato in Variable Read Address.

Il dato sarà presente in Variable Read Value almeno 3 ms dopo la scrittura dell'indirizzo.

Scrittura di un parametro (SDO Download):

Si utilizza l'oggetto composto 0x2006. Esso possiede due subindex, di cui:

- Subindex 1: Variable Write Address
- Subindex 2: Variable Write Value

L'indirizzo viene impostato in Variable Write Address.

Il dato da scrivere deve essere caricato in Variable Write Value.

Affinchè la scrittura sia effettiva occorre aspettare almeno 3ms.

Di seguito una tabella riassuntiva degli indirizzi per ogni parametro del drive.

Parametro	Indirizzo	Parametro	Indirizzo	Parametro	Indirizzo
d1	01 (0x0001)	e5	18 (0x0012)	o2	82 (0x0052)
d2	02 (0x0002)	e6	19 (0x0013)	o3	83 (0x0053)
d3	03 (0x0003)	e7	20 (0x0014)	a1	48 (0x0030)
d4	04 (0x0004)	e9	22 (0x0016)	a2	49 (0x0031)
d5	05 (0x0005)	i1	35 (0x0023)	a3	50 (0x0032)
d6	06 (0x0006)	i2	36 (0x0024)	a4	51 (0x0033)
d7	07 (0x0007)	i3	37 (0x0025)	a5	52 (0x0034)
d8	08 (0x0008)	i4	38 (0x0026)	a6	53 (0x0035)
e1	09 (0x0009)	i5	39 (0x0027)	h1	59 (0x003B)
e2	10 (0x000A)	i6	90 (0x005A)	h2	60 (0x003C)
e3	58 (0x003A)	i7	99 (0x0063)	h3	61 (0x003D)
e4	55 (0x0037)	p1	41 (0x0029)	h4	62 (0x003E)
f1	11 (0x000B)	p2	42 (0x002A)	h5	63 (0x003F)
f2	12 (0x000C)	p3	43 (0x002B)	h6	64 (0x0040)
f3	13 (0x000D)	p4	44 (0x002C)	h7	65 (0x0041)
c1	14 (0x000E)	p5	45 (0x002D)	h8	66 (0x0042)
c2	15 (0x000F)	p6	46 (0x002E)	h9	67 (0x0043)
c3	16 (0x0010)	p7	47 (0x002F)	n1	40 (0x0028)
c4	17 (0x0011)	o1	81 (0x0051)		

Nota: non tutti i parametri hanno valenza istantanea dopo la scrittura, per alcuni, come ad esempio il d8 (accoppiamento drive motore!) occorre salvare in flash e riavviare il drive.



Salvataggio in flash dei parametri

Attraverso l'oggetto "0x2007 Save Parameters" è possibile salvare nella flash del drive i parametri.

In particolare, seguire i passi successivi:

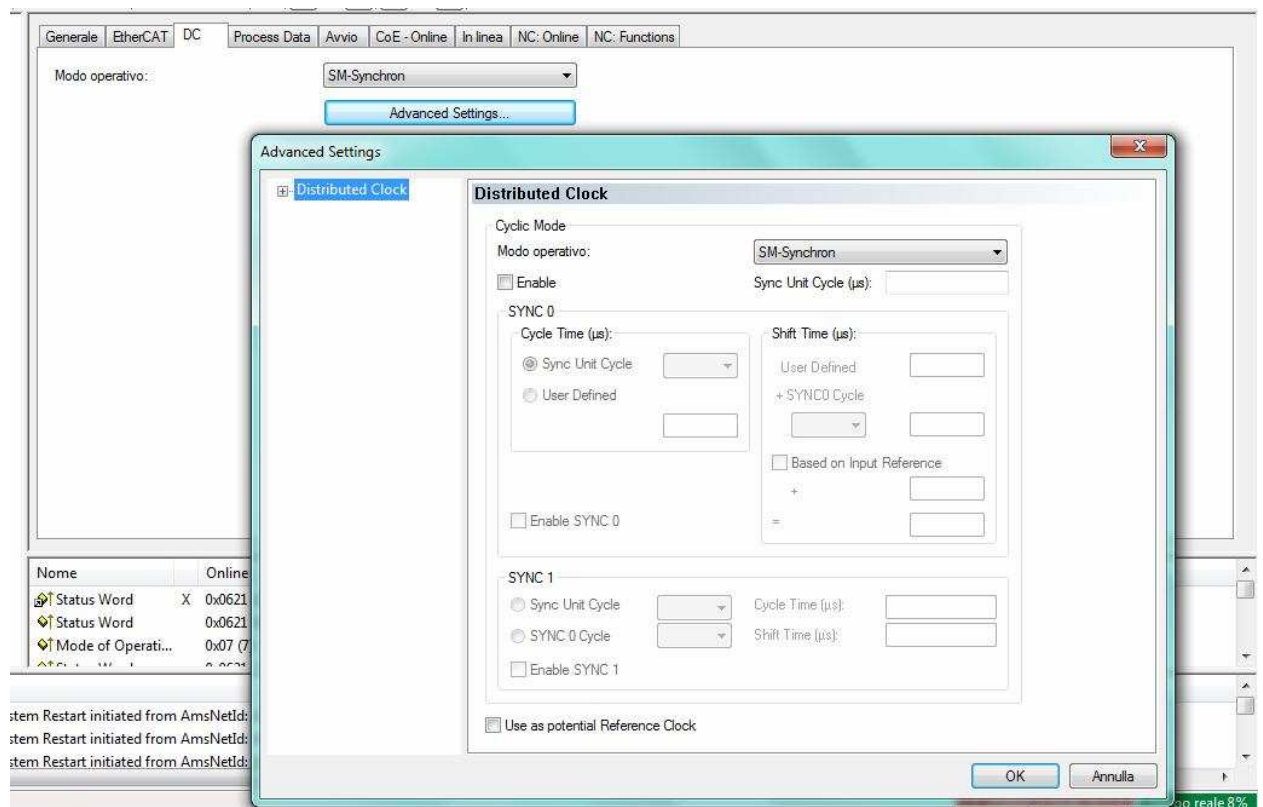
1. Disabilitare l'asse. Sul display del Drive deve apparire la scritta "Fr".
2. Via SDO Download scrivere il dato 0x11 nell'oggetto 0x2007.
3. Durante la fase di salvataggio in flash appariranno due lineette sul display del drive. Attendere che riappaia la scritta "Fr", essa segnalerà il corretto salvataggio.
4. Se fosse necessario (alcuni parametri necessitano del riavvio per essere attuati) spegnere e riaccendere il drive.



Settaggi SM / DC per drive ECO

Al fine di permettere una perfetta sincronizzazione con il master, indispensabile nel caso di funzionamento in interpolata, è consigliabile impostare sul controllo ethercat la sincronizzazione *SM - Synchron*.

Vedi figura seguente nel caso specifico di twincat.



I segnali SYNC0/SYNC1 **non** vanno abilitati utilizzando SM - Synchron.

Nota importante: nel caso in cui si voglia utilizzare la sincronizzazione con *DC - Synchron*, occorre poter garantire l'arrivo del dato di interpolazione allo slave con un basso jitter, all'interno dello stesso periodo di sync.

Una buona condizione è che il jitter sia minore del periodo di sync diviso 2.



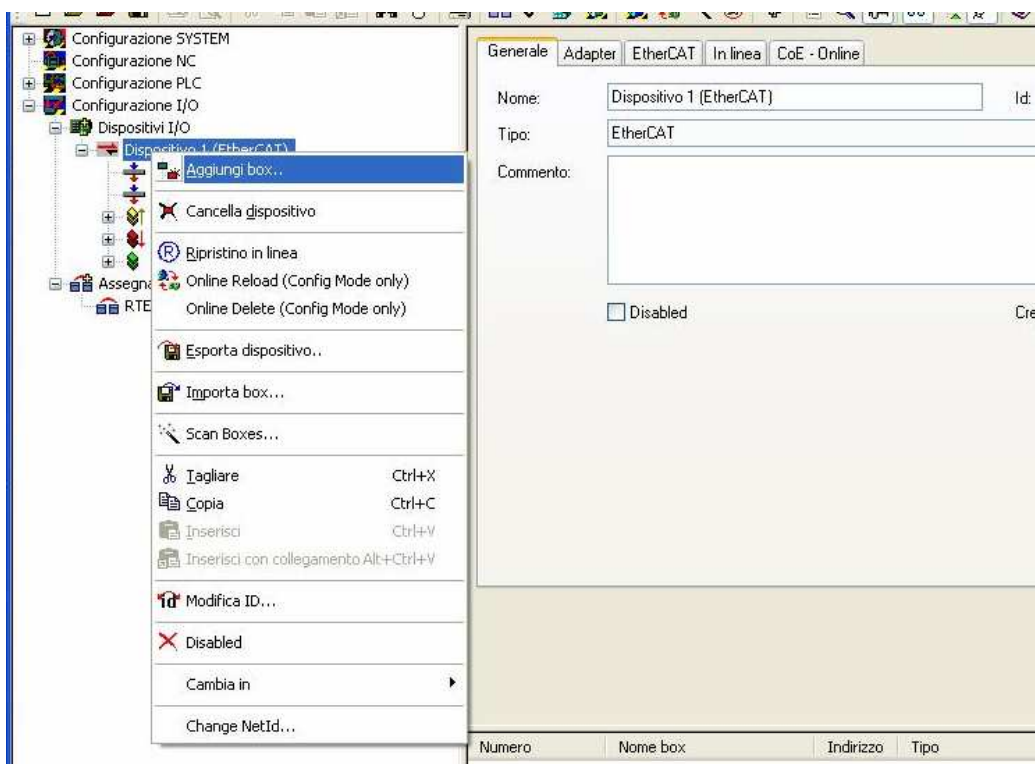
Importazione file .xml con Twincat®

Le modalità di importazione del file descrittivo del device slave dipendono sostanzialmente dal tipo di master.

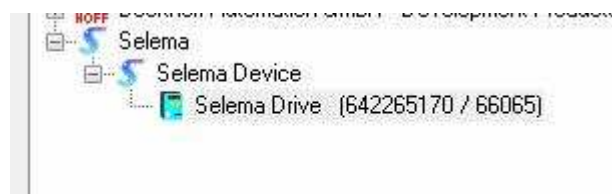
Importazione con Twincat®

Nell'esempio seguente viene mostrato il caso di un Master Ethercat Beckhoff, con sistema di sviluppo Twincat®.

Copiare il file .xml fornito col drive nella directory: C:\Twincat\Io\EtherCAT, dove C: è la lettera associata al disco rigido del pc in cui è installato il software. Una volta aperto il programma e configurata la scheda di rete per ethercat, cliccare col pulsante destro del mouse sul *Dispositivo* e selezionare *aggiungi box*:

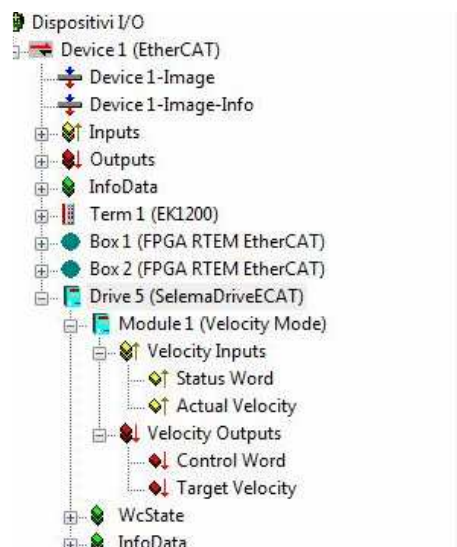


Verranno mostrati i dispositivi compatibili, selezionare:



E confermare premendo OK.

A questo punto il drive, verrà importato nel sistema Twincat®.



Cambio configurazione PDO con Twincat

Selezionando il tab Process Data, è possibile modificare la configurazione dei pdo, aggiungendo o togliendo variabili agli oggetti 0x1600 – 0x1603, 1A00 – 1A03, aggiungendo o togliendo pdo agli oggetti 0x1C12, 0x1C13.

Sync Manager:

SM	Size	Type	Flags
0	128	MbxOut	
1	128	MbxIn	
2	6	Outputs	
3	6	Inputs	

PDO List:

Index	Size	Name	Flags	SM	SU
0x1A02	6.0	TxPDO 3		3	0
0x1600	6.0	RxPDO 1		2	0

PDO Assignment (0x1C12):

0x1600

Download

PDO Assignment

PDO Configuration

PDO Content (0x1A02):

Index	Size	Offs	Name	Type	Default (hex)
0x6041:00	2.0	0.0	Status Word	UINT	
0x6064:00	4.0	2.0	Position Actual Value	DINT	
		6.0			

Predefined PDO Assignment: (nessuno)

Load PDO info from device

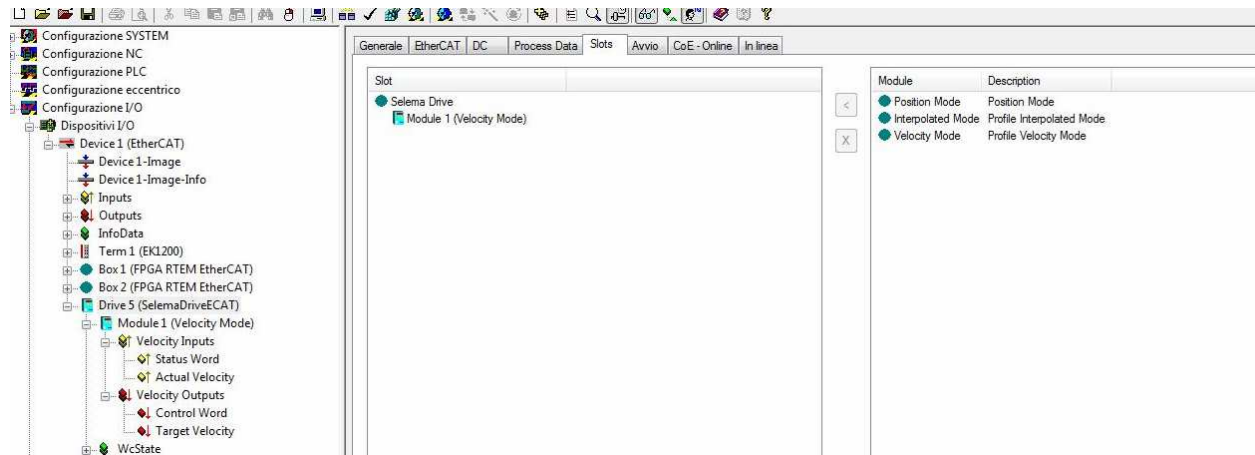
Sync Unit Assignment...

Per rendere effettivi i cambiamenti (vengono riscritti gli oggetti 0x1C12 e 0x1C13 ed eventualmente gli oggetti 0x1600 – 0x1603, 1A00 – 1A03), cliccare sull'icona




Usando gli slots con Twincat®

Ora rimane da *selezionare la configurazione pdo* desiderata. Per farlo selezionare l'icona del drive e quindi cliccare sul tab Slots.



Di default sarà impostata la modalità *Interpolated Mode*, che permette il comando del drive in posizione con set point interpolato.

Per cambiare configurazione, nel riquadro *Slot* selezionare la modalità corrente e prendere il pulsante X per cancellarla. Quindi selezionare la modalità desiderata nel riquadro *Module* e premere il pulsante < .

Per rendere effettivi i cambiamenti (vengono rimappati gli oggetti 0x1C12 e 0x1C13), cliccare sull'icona  .

Nota: tutte le operazioni eseguite graficamente da twincat, possono essere eseguite "a mano" scrivendo gli opportuni valori negli oggetti dedicati.



Codici allarmi specifici

Gli allarmi vengono visualizzati sia sul display del drive, sia sul pc collegandosi con il software drive watcher. Per ulteriori informazioni vedi capitolo **Codici di Allarme** di questo manuale.

Ad ogni allarme corrisponderà un *error code* segnalato nell'oggetto 0x603F sub0, il cui valore determinato da specifica ethercat.

Di seguito la tabella delle corrispondenze:

Codice Allarme ECO	Error Code 0x603F sub0	Descrizione Allarme
1	0x2340	Problema IGBT
2	0x7120	Termica motore
3	0x2310	Sovracorrente sul motore
4	0x3210	Sovratensione sul bus DC di potenza
5	0x3220	Sottotensione
6	0x4310	Termica drive
7	0x5220	Processore DSP in fault
8	0x7300	Allarme trasduttore motore
9	0x5500	Allarme flash memory
10	0x7110	Eccessivo intervento resistore di frenatura
11	0x7500	Errore di comunicazione in rete
12	0x7320	Errore durante operazione di Homing
13	0x7500	Errore comunicazione
14	0x7320	Procedura di Homing non terminata correttamente
15	0x5500	Allarme overflow
16	0x8611	Errore di inseguimento
20	0x8612	Allarme LS1
21	0x8612	Allarme LS2
23	0x7300	NO Allineamento encoder assoluto
24	0x7300	Encoder assoluto non comunica
25	0x7300	Encoder assoluto: asse in movimento all'accensione
26	0x5114	Allarme modalità non consentita
30	- -	Timeout comunicazione interna
31	0x7500	Errore 1 comunicazione interna al drive
32	0x7500	Errore 2 comunicazione interna al drive
33	0x6320	Errore mappatura pdo non eseguita correttamente
34	0x8700	Timeout sincronizzazione con il Master EtherCAT
35	0x5530	Errore caricamento eeprom ethercat all'avvio



Reset allarmi

L'allarme può essere resettato via coe utilizzando gli stessi comandi canopen (scrittura 0x80 su oggetto 0x6040) . Nel caso in cui l'operazione non abbia successo, può significare che la condizione di allarme persiste.

Capitolo 6

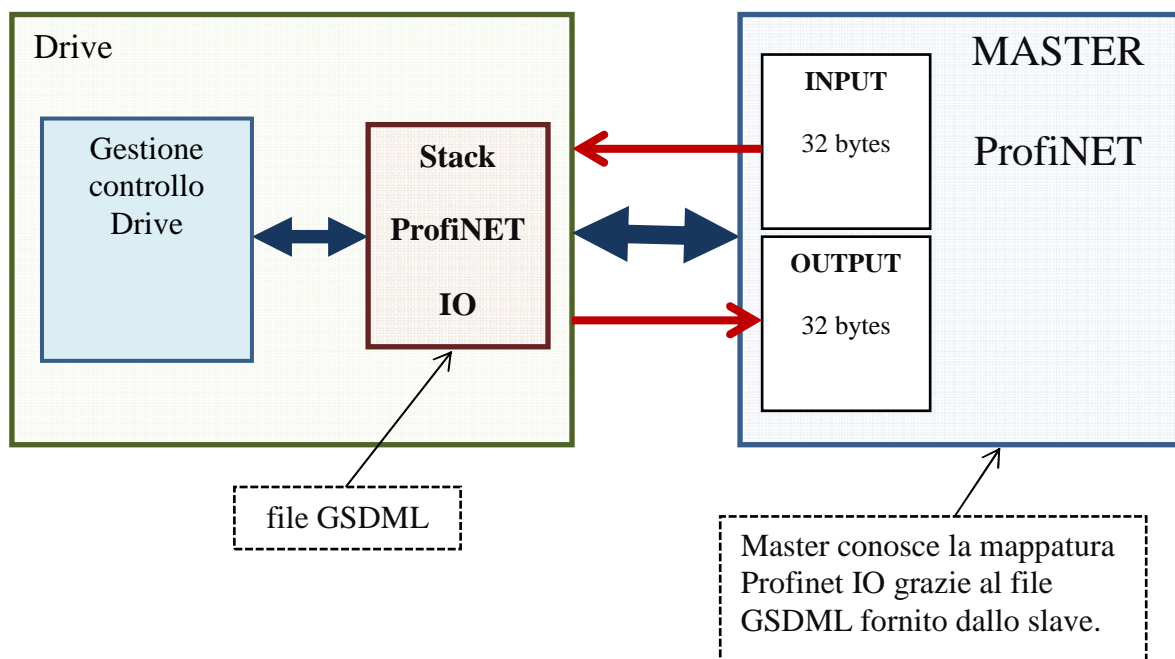


Profinet IO

La gestione del protocollo avviene attraverso un meccanismo di memoria condivisa. In particolare il Master Profinet vedrà l'azionamento come un blocco di dati, in cui una parte di 32 bytes sarà in ingresso ed una parte di 32 bytes in uscita. Viene fornito al Master un file .xml GSDML.

Struttura generale

Graficamente si può sintetizzare la struttura di comunicazione verso il protocollo Profinet come segue:



Nota: input ed output sono visti lato azionamento.

A seconda dei registri definiti nelle aree di memoria di input ed output, il master profinet sarà in grado di controllare l'azionamento, gestendolo come se si stesse interfacciando con un dispositivo IO.

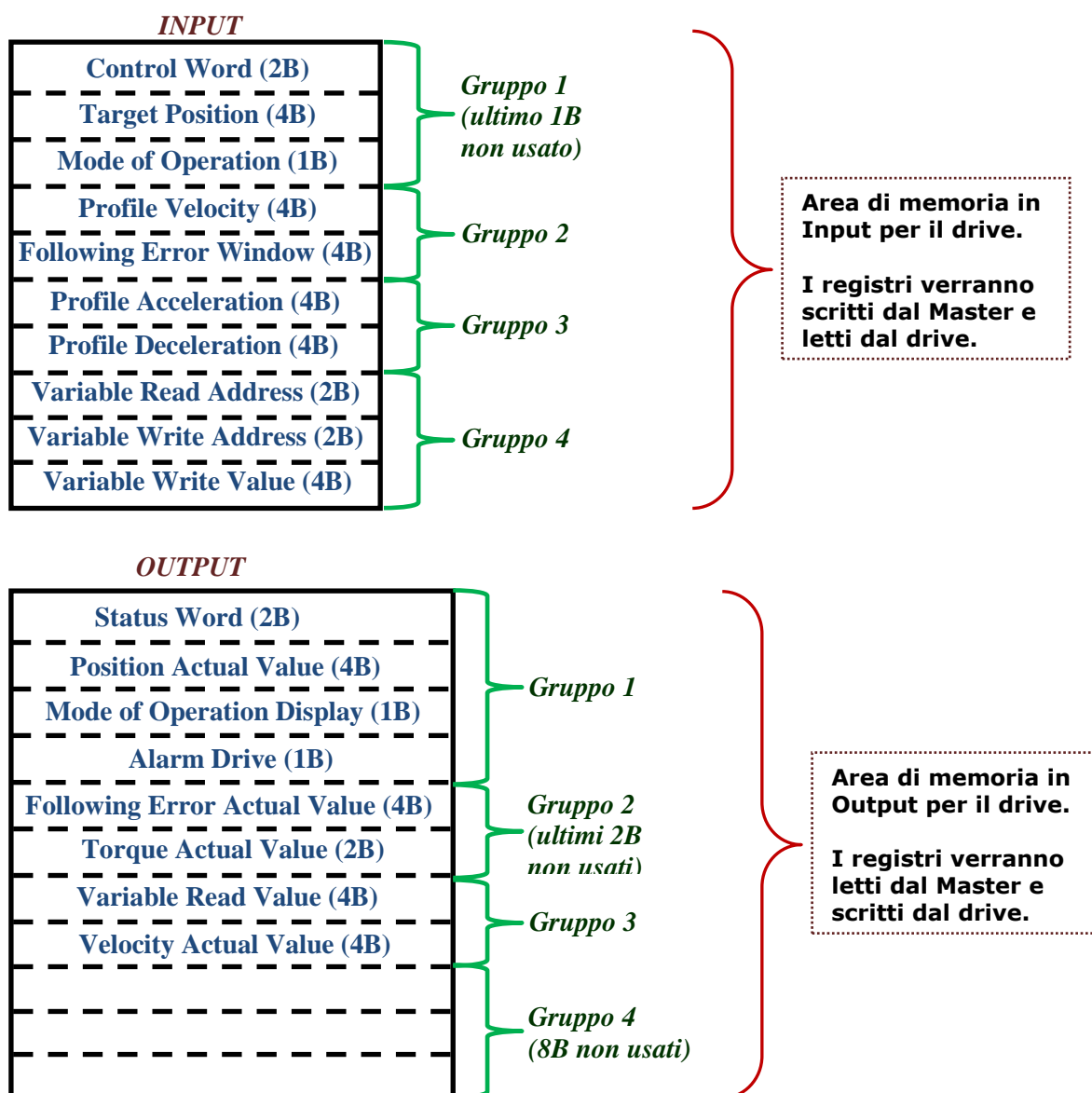
Il significato di ogni registro verrà descritto velocemente più avanti. Per maggiori informazioni riferirsi alla sezione Quick Reference Registri e Parametri. Per approfondire ulteriormente riferirsi al documento Additional Information, capitoli 2,3,4,5,6.

Parametrizzazione Profinet IO per drive

Per utilizzare il protocollo profinet con i drive occorre impostare il parametro **C9 = 13**, salvare in flash, quindi spegnere e riaccendere il drive.

Mappatura memoria condivisa

Viene definita in fase di avvio del sistema, ha dimensione fissa, è composta da 32 bytes per i dati di input e 32 bytes per i dati di output. E' suddivisa in sottogruppi di 8 byte ognuno. In particolare:



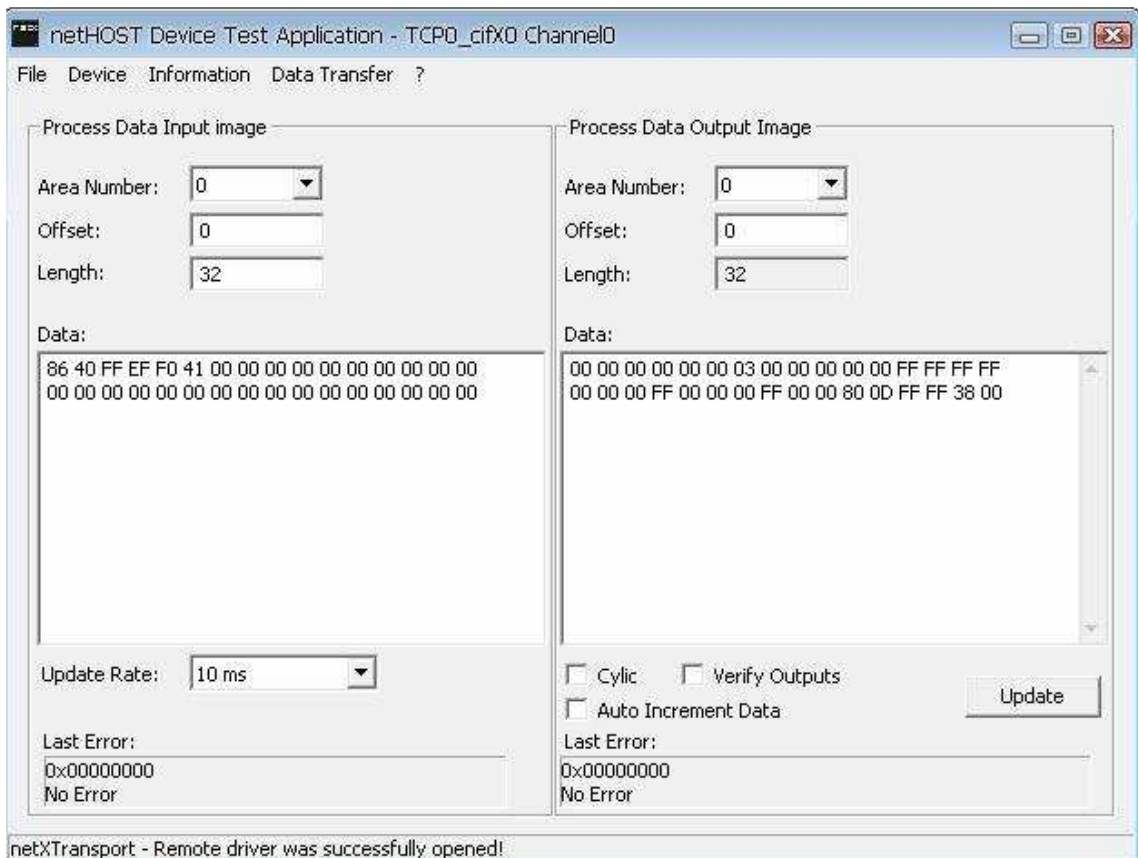
La codifica di ogni registro è di tipo big endian (MSB .. LSB)



Esempio interfaccia con Master Profinet Hilscher

Per maggiore chiarezza riguardo la memoria condivisa, viene mostrato l'interfacciamento con un master profinet di hilscher. Quest'ultimo permette di visualizzare l'area di memoria "grezza".

La schermata principale è la seguente, nella quale sarà possibile leggere i dati trasferiti dal drive al master e scrivere i dati dal master verso il drive.



Nel riquadro a sinistra sono presenti dati in output dal drive ed inviati al Master.

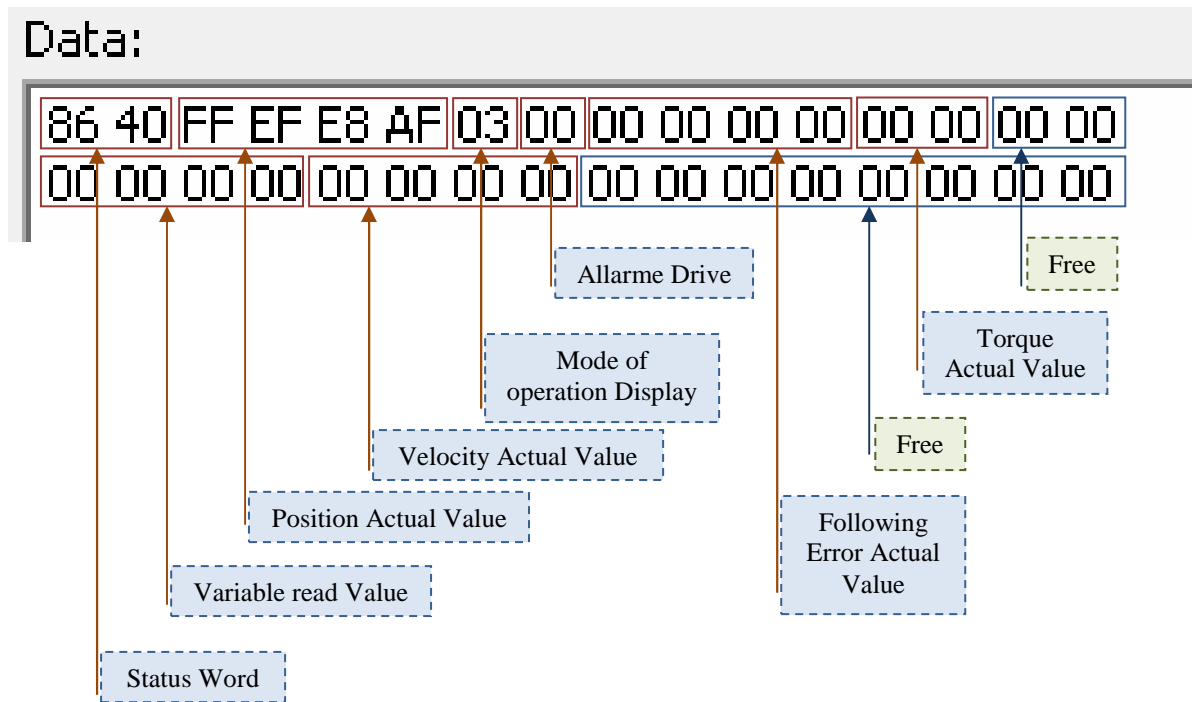
Nel riquadro di destra sono presenti i dati inviati dal Master in input per il drive.

I singoli registri sono in formato big endian.

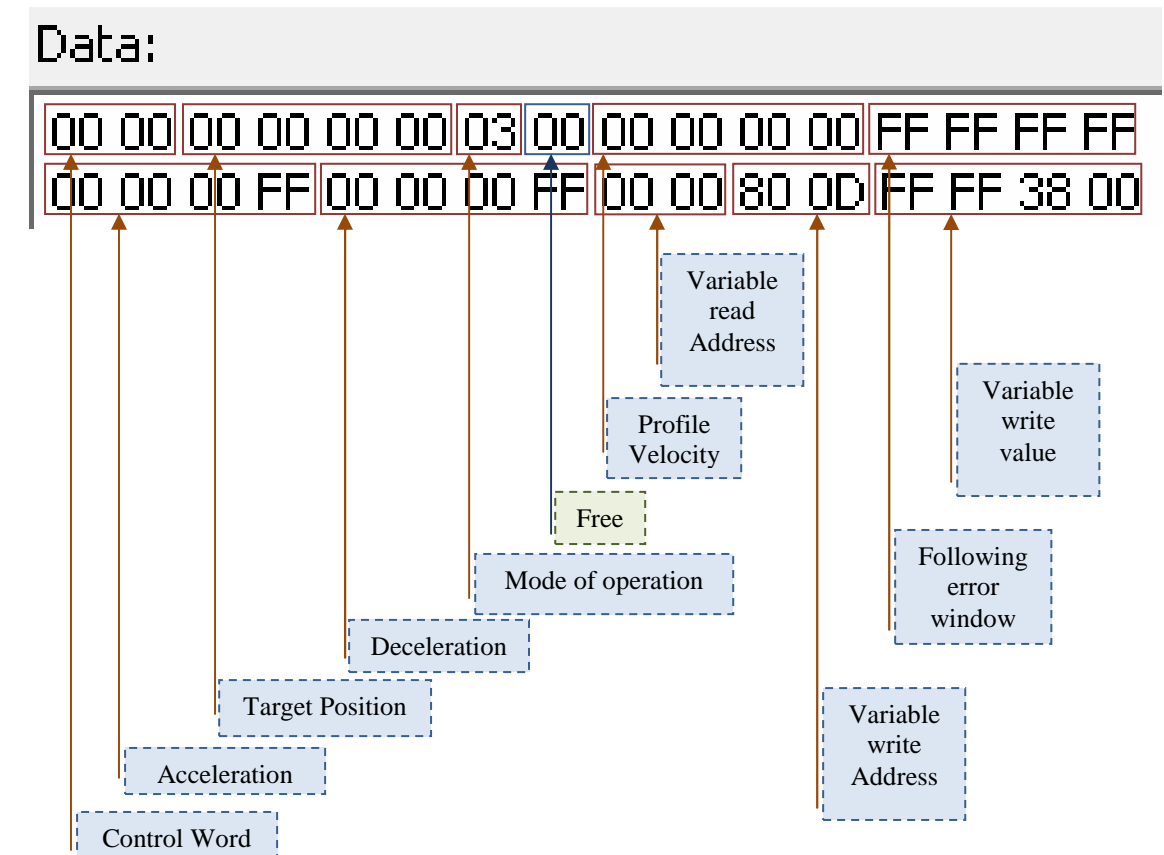
Ad esempio nella figura precedente il registro Status Word = 0x8640, è composto da MSB = 86_h, LSB = 40_h in sequenza.

Nota: i numeri sono visualizzati in esadecimale.

In particolare, la mappatura di default dei dati in **output** è la seguente:



La mappatura di default dei dati in **input** è la seguente:





Accesso asincrono a registri e parametri

Utilizzando i registri di gestione *Variable Read Value*, *Variable Write Value*, *Variable Read Address*, *Variable Write Address*, è possibile leggere e scrivere tutti i parametri ed i registri dell'azionamento.

In particolare:

- Letture di un registro / parametro: la coppia *Variable Read Value* e *Variable Read Address* permette la lettura delle variabili del drive. L'indirizzo viene impostato in *Variable Read Address*; se è compreso fra 0 e 99 si accede ad un parametro del drive, se l'indirizzo è compreso fra 0x8000 e 0x8FFF si leggono i valori dei registri.

Il dato sarà presente in *Variable Read Value* almeno 3 ms dopo la scrittura dell'indirizzo.

- Scrittura di un registro / parametro: la coppia *Variable Write Value* e *Variable Write Address* permette la scrittura delle variabili del drive. L'indirizzo viene impostato in *Variable Write Address*; se è compreso fra 0 e 99 si scrive un parametro del drive, se l'indirizzo è compreso fra 0x8000 e 0x8FFF si scrivono i valori dei registri.

Il dato da scrivere deve essere caricato in *Variable Write Value*.

Affinchè la scrittura sia effettiva occorre aspettare almeno 3ms.



Importazione file GSDML

Il file gsdml ha estensione .xml e permette al Master di conoscere le possibili configurazioni dello slave.

L'importazione e la gestione del file dipende dal master.

A livello generale, per lo slave profinet, occorre selezionare:

- 1 Slot in input di dimensione 32 bytes
- 1 Slot di output di dimensione 32 bytes



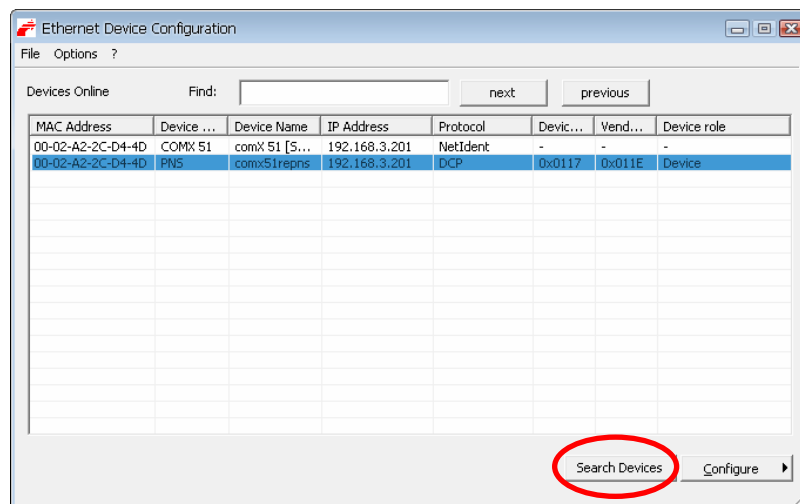
Indirizzamento drive nella rete Profinet

In profinet l'azionamento viene identificato all'interno della rete attraverso il suo indirizzo IP.

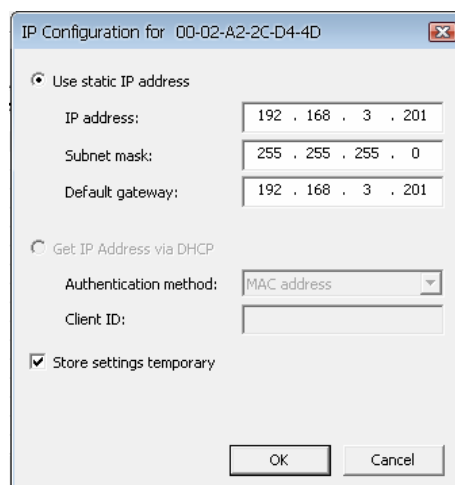
Esso può essere impostato sia dal Master in fase di configurazione, sia attraverso un tool PC per sistemi Windows® chiamato *Ethernet Device Configuration*, distribuito gratuitamente.

Di seguito viene mostrato come cambiare l'indirizzo utilizzando il tool per PC. Passi da seguire:

1. Installare il tool via installer *EnDevConfigTool.msi*.
2. Aprire il programma e cliccare su *Search Devices* per rilevare il nodo.



3. Selezionare il nodo Profinet di interesse *con campo protocol: DCP*
4. Cliccare quindi su *Configure* → *Set IP Address*. Apparirà la schermata seguente:



5. Modificare a piacimento i campi: *IP address*. *Subnet mask*, *Default gateway*. Quindi premere OK.

Nota: se l'azionamento non viene rilevato, come indicato al punto 2, verificare che la sottorete del proprio pc sia la stessa del drive.



Esempio posizionamento punto a punto

I registri non presenti direttamente nella zona di memoria condivisa andranno gestiti utilizzando il loro indirizzo globale attraverso un indirizzamento indiretto (vedi capitolo precedente). Per farlo occorre utilizzare i registri di comando: *Variable Read Value*, *Variable Write Value*, *Variable Read Address*, *Variable Write Address*. Di seguito in grassetto i registri in cui occorre utilizzare l'indirizzamento indiretto.

- Configurazione dell'asse come posizionatore.
In questo esempio si vuole far funzionare l'asse al nodo 1 come posizionatore.
 - **Registro 02 feed_constant** (indirizzo: 0x8002, valore: 0x003C)
 - Registro 10 profile_acceleration (indirizzo: 0x800A, valore: 0x0064)
 - Registro 11 profile_deceleration (indirizzo: 0x800B, valore: 0x0064)
 - Registro 12 profile_velocity (indirizzo: 0x800C, valore: 0x03E8)
 - **Registro 27 motion_profile_type** (indirizzo: 0x801B, valore: 0x0000)

- Configurazione per effettuare l'homing:
 - **Registro 18 homing_method** (indirizzo: 0x8012, valore: 0x0021)
 - **Registro 19 home_offset** (indirizzo: 0x8013, valore: 0x0000)
 - **Registro 21 homing_slow_speed** (indirizzo: 0x8015, valore: 0x000A)
 - **Registro 22 homing_acceleration** (indirizzo: 0x8016, valore: 0x0064)

- I parametri sono quelli di default, fatta eccezione per C9 che deve valere 13.
Il parametro d8 deve essere impostato conformemente al motore utilizzato.

- Si fornisce quindi la 24 Vdc sugli ingressi TEN e IEN hardware.

- Si resettano eventuali allarmi presenti
 - Registro 0 control_word (indirizzo: 0x8000, valore: 0x0008)

- Per fare l'homing:
 - Selezione modo operativo:
Registro 1 modes_of_operation (indirizzo: 0x8001, valore: 0x0006)
 - Messa in coppia del motore
Registro 0 control_word (indirizzo: 0x8000, valore: 0x0003)
 - Inizio operazione di zero assi, si porta alto il bit 6 della control word:
Registro 0 control_word (indirizzo: 0x8000, valore: 0x0043)

Alla terminazione della procedura di homing, il *bit 6 della status word* viene portato alto.

- Posizionamento punto a punto alla posizione 6000 (0x1770) :
 - Si seleziona, il modo posizionatore
Registro 1 modes_of_operation (indirizzo: 0x8001, valore: 0x0001)
 - Si imposta la posizione di target
Registro 9 target_position (indirizzo: 0x8009, valore: 0x1770)
 - Messa in coppia del motore
Registro 0 control_word (indirizzo: 0x8000, valore: 0x0003)
 - Start posizionamento, portando alto il bit 5 della control word
Registro 0 control_word (indirizzo: 0x8000, valore: 0x0023)
 - La posizione attuale si può leggere dal registro Position Actual Value
Registro 6 position_actual_value (indirizzo: 0x8006)



Codici allarmi specifici

Codici di allarme relativi al Profinet.

Codice allarme	Significato	Resettabile
30	Timeout comunicazione interna al Drive	NO
31	Errore fase 1 inizializzazione interfaccia Profinet	NO
32	Errore fase 2 inizializzazione interfaccia Profinet	NO
33	Fieldbus Profinet non partito correttamente	NO
34	Timeout sincronizzazione con il Master Profinet	SI

Per tutti gl'altri allarmi, non specifici della modalità Profinet, riferirsi al capitolo *Codici di Allarme* presente in questo documento.



Reset allarmi

L'allarme può essere resettato scrivendo il codice 0x08 nella Control Word . Nel caso in cui l'operazione non abbia successo, può significare che la condizione di allarme persiste o non sia resettabile.