第三章 CANopen

3.1 协议介绍

CAN 是控制器局域网络(Controller Area Network)的简称，由 2 根线（CAN-H 和 CAN-L）组成，结构简单，并且内部集成了错误探测和管理模块；采用了多主竞争式总线结构，可在各节点之间实现自由通信。

CAN 是由德国 BOSCH 公司开发，并最终成为应用广泛的国际标准（ISO 11898）。CANopen 是一种架构在 CAN 上的高层通讯协定，它由非营利组织 CiA（CAN in Automaion）进行标准的起草及审核工作。基本的 CANopen 设备及通讯子协定定义在 CiA draft standard 301 中。以 CiA 301 为基础再扩充了针对运动控制的子协定 CiA draft standard 402。

DS / RS 伺服驱动器在 CANopen 总线网络（支持“CIA 301”和“CIA 402”）中作为从站使用，参数功能通过使用对象字典“制造商指定数据”区实现，所有的参数、参数值和功能都是通过 Index 和 Sub-Index 组成的地址来访问和存取。

理论上可以有 CAN 节点可以达到 127 个，而实际中最大节点数取决于 CAN 收发器的性能。CAN 总线的长度取决于通讯速度具体如下：

通讯波特率	CAN 总线最大长度
1Mbps	25m
500Kbps	100m
250Kbps	250m
125Kbps	500m

用户可通过 CAN 总线控制 DS / RS 型号伺服驱动器，以下为常见测试接线：

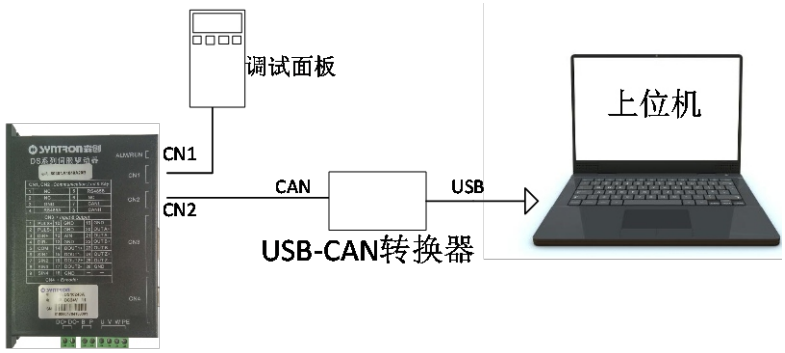


图 3-1 上位机 CAN 测试连接

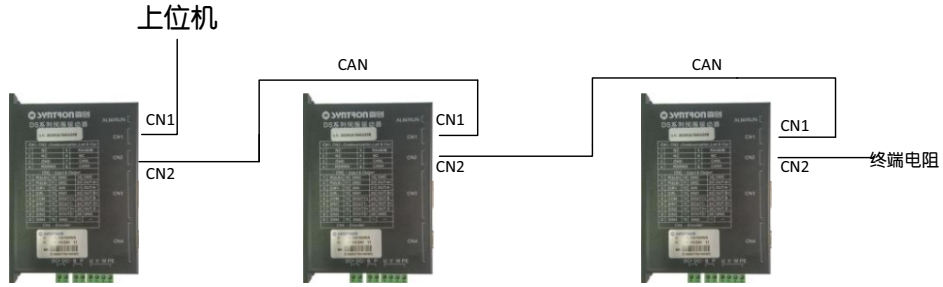


图 3-2 CAN 总线组网示例

用户可通过 USB 转 CAN 工具来进行测试和熟悉。

3.2 Key parameters (hệ tham số 1 - dùng cho các driver đơn trục)

单轴驱动器 CAN 总线 ID 默认为 1，波特率为 500Kbps，用户可按需求配置。

参数编号	CAN INDEX	参数名称	参数内容	设定范围	出厂设定
Fn00	0x2000	控制模式	0：外部速度；1：内部速度；2：位置模式； 3：JOG 模式；4：转矩模式；5-8：混合控制模式 20：CANopen 模式	0-20	2
Fn8D	0x208D	协同模式目标 ID	如目标 CAN ID 驱动报警，则协同模式生效，本驱动器报警方式由 FnF8 决定，报警码为 Err_BE。	0-127	0
Fn8E	0x208E	故障上报时间	0：故障后上报一次报警码； >1：故障后定时上报时间（单位：ms）	0-4000	50
Fn8F	0x208F	6063H 反馈处理	0：正转增计数；1：正转减计数；14：绝对值码盘计数转换	0-1	0
Fn99	0x2099	通讯参数模式	1：参数在 RAM 中运行； 0：参数写入 flash 保存 注：Flash 写入寿命有限，如该参数经常更改且无需保存，请置 Fn99 为 1，否则会写坏驱动器 FLASH	0~1	0
Fn9A	0x209A	CAN 通讯 ID	0 为广播地址，1~127 为站址编号	0~127	1
Fn9C	0x209C	CAN 通讯波特率	CAN 总线波特率，单位为 Kbps	100-1000	500
Fn9E	0x209E	TPDO 上传机制	0:上电自动上传; 1:需配置启动上传;	0~1	0
FnEA	0x20EA	6063 清零	置 1 则对象字典 6063 清 0	0~1	0
FnF7	0x20F7	CAN 总线断线检测时间	CAN 总线断线检测时间，单位为 ms，报警方式由 FnF8 决定，报警码为 Err_Lo。	0-30000	0
FnF8	0x20F8	自主停车、总线报警方式	0：报警；1：关使能；2：给零速；	0-2	0
FnFF	0x20FF	CAN 限位主动上传开关	0：限位等信息不主动上传； 1：限位信息、位置到达信息通过紧急帧上传。	0-1	0
Dn12	0x3012	报警信息码	见附录报警信息		
Dn1D	0x301D	接收帧总数	*开发参数，用于简易检测接收 can 指令的总数		
Dn1E	0x301E	发送帧总数	*开发参数，用于简易检测发送 can 报文的总数		

注意：如需修改以上参数，修改完毕后，需重启生效。

3.3 Key parameters (hệ tham số 2 - phù hợp với driver Ds20240)

Đối với driver 2 trục, mặc định CAN ID của trục A, B là 1 v2, baudrate 1000Kps

T.số	CAN INDEX	Tên tham số	Nội dung tham số	Phạm vi	Defaut
F0.1.002	0x2102	A axis: control mode	1: Tốc độ set sẵn 2: Mode vị trí 20: CANopen	0-20	1
F1.1.002	0x3102	B axis: control mode			
F0.2.010	0x2210	A axis: phủ định vị trí	Phủ định hướng đếm của object dictionary phản hồi vị trí 6063	0~1	0
F1.2.010	0x3210	B axis: phủ định vị trí			
Pn5.006	0x4506	CAN BUS ID	Hai số đầu là ID của trục B, 2 số cuối là ID của trục A.	0-9999	0201
Pn5.007	0x4507	CAN BUS baudrate	Baudrate, đơn vị Kps	0-1000	1000
Pn5.00A	0x450A	Set vị trí ghi tham số	0: ghi vào RAM; 1: ghi vào flash	0~1	0
Pn5.00B	0x450B	Thời gian kiểm tra kết nối CAN bus	Thời gian kiểm tra kết nối CAN (đơn vị ms) chuyển sang trạng thái bảo vệ nếu quá giờ	0-9999	0
Pn5.00C	0x450C	Bảo vệ khi mất kết nối CAN	0: cảnh báo; 1: disable servo; 2: set tốc 0	0-9999	0
Pn5.00D	0x450D	Đóng ngắt bản tin báo tín hiệu differential	0: không báo; 1: báo. Chỉ dùng khi điều khiển bằng differential	0-1	0
Pn6.002	0x4602	Xóa cảnh báo	Khi có sự cố, set là 0 sẽ xóa cảnh báo	0-1	0
Pn6.003	0x4603	Restart driver	Restart driver	0-1	0
Pn6.00A	0x460A	Đường kính bánh	Đơn vị 0.1mm, dùng khi điều khiển differential	0-9999	0
Pn6.00B	0x460B	Khoảng cách bánh	Đơn vị 0.1mm, dùng khi điều khiển differential	0-20000	0
Pn6.00C	0x460C	Định nghĩa bánh trái, phải	Dùng khi điều khiển differential, định nghĩa bánh trái là A, hay B, từ đó định nghĩa hướng tiến.	0-1	0
Dn0.012	0x5012	Mã cảnh báo	Xem mục 4.2: thông tin cảnh báo 2 trục	0-20	0

3.4 CANopen - Object dictionary

Object dictionary là đặc tính quan trọng nhất của CANopen, mỗi một object dictionary gồm 16 bit index, và 8 bit sub index. Thông qua đọc viết object dictionary để điều khiển thiết bị. Index của object dictionary của mỗi một node nằm trong phạm vi từ 0x1000 đến 0x9FFF, cấu trúc như dưới đây:

Index（hex）	Nội dung
0001-001F	Static data (như Boolean, Integer 16)
0020-003F	Loại data phức tạp (tổ hợp từ loại đơn giản, như PDOCommPar)
0040-005F	Nhà sản xuất định nghĩa loại data phức tạp
0060-007F	Static data do thiết bị định nghĩa
0080-009F	Loại data phức tạp do thiết bị định nghĩa
1000-1FFF	Vùng giao thức truyền thông con (như loại thiết bị, error registers, số lượng PDO)
2000-5FFF	Vùng dữ liệu do nhà sản xuất chỉ định
6000-9FFF	Vùng giao thức thiết bị tiêu chuẩn
A000-BFFF	Vùng giao thức kết nối tiêu chuẩn

Tham số Fn, Dn, Pn nằm trong vùng 0x6000~ 0x9FFF. Độ dài của Fn, Pn là 16bit, Dn dài 32 bit.

Loại hình tham số	Địa chỉ bắt đầu Fn	Địa chỉ bắt đầu Dn	Địa chỉ bắt đầu Pn
Hệ tham số 1	0x2000	0x3000	-
Hệ tham số 2	A 0x2000 B 0x3000	A 0x5000 B 0x5100	0x4000

Dưới đây là các object dictionary hay dùng:

3.4.1 Coltrol word 6040H

Object dictionary 6040H được định nghĩa trong Ds402:

Bit15-9	**Bit8**	**Bit7**	**Bit6-4**	**Bit3**	**Bit2**	**Bit1**	**Bit0**
Reserved	Halt	Fault Reset	Operation Mode specific	Enable Operation	Quick Stop	Enable Voltage	Switch on

Lần đầu enable driver phải ghi lần lượt 0x06, 0x07, 0x0F. Sau đó nếu ghi 0x07 thì disable driver, ghi tiếp 0x0F mới enable driver.

Chú ý: một số phiên bản không hỗ trợ 6040H dừng khẩn cấp, reset; tham khảo Fn, Pn.

3.4.2 Status word 6041H

Object dictionary 6041H được định nghĩa trong Ds402 như sau:

Bit15-14	**Bit13-12**	**Bit11**	**Bit10**	**Bit9**	**Bit8**	**Bit7**
Manufacturer Specific	Operation Mode Specific	Internal Limit Active	Target Reached	Remote	Manufacturer Specific	Warning
Bit6	**Bit5**	**Bit4**	**Bit3**	**Bit2**	**Bit1**	**Bit0**
Switch On Disabled	Quick Stop	Voltage Disabled	Fault	Operation Enabled	Switch On	Ready to Switch On

Ở trạng thái ban đầu, 6041H như sau (0x00 50):

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
M	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0

Sau khi 6040H ghi xong 0x06, thì 8bit thấp của 6041H như sau (0x31):

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
M	-	-	-	-	-	-	-	-	0	0	1	1	0	0	0	1

Sau khi 6040H ghi xong 0x07, thì 8bit thấp của 6041H như sau (0x33):

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
M	-	-	-	-	-	-	-	-	0	0	1	1	0	0	1	1

Sau khi 6040H ghi xong 0x0F, thì 8bit thấp của 6041H như sau (0x37):

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
M	-	-	-	-	-	-	-	-	0	0	1	1	0	1	1	1

Sau khi có cảnh bảo, 6041H như sau (0x00 88):

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
M	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0

3.4.3 Danh sách Object dictionary

Mode	Index	Name	Type	Attr.	unit
ALL	603Fh	Error Code	UNSIGNED16	RO	-
ALL	6040h	Controlword	UNSIGNED16	RW	-
ALL	6041h	Statusword	UNSIGNED16	RO	-
ALL	6060h	Modes of operation	INTEGER8	RW	-
ALL	6061h	Modes of operation display	INTEGER8	RO	-
PP	6062h	Position demand value	INTEGER32	RO	pulse
PP	6063h	Position actual value*	INTEGER32	RW	increments
PP	6064h	Position actual value	INTEGER32	RO	cmd position
PP	6065h	Following error window	UNSIGNED32	RW	pulse
PP	6066h	Following error time out	UNSIGNED16	RW	ms
PP	6067h	Position window	UNSIGNED32	RW	pulse
PP	6068h	Position window time	UNSIGNED16	RW	ms
PV	6069h	velocity_sensor_actual_value	INT32	RO	0.1rpm
PV	606Bh	velocity_demand_value	INT32	RO	0.1rpm
PV	606Ch	velocity_actual_value	INT32	RO	
PV	606Dh	velocity_window	UINT16	RW	0.1rpm
PV	606Eh	velocity_window_time	UINT16	RW	ms
PV	606Fh	velocity_threshold	UINT16	RW	0.1rpm
PV	6070h	velocity_threshold_time	UINT16	RW	ms
PT	6071h	Target torque	INTEGER16	RW	0.1%Torque
PP	607Ah	Target position	INTEGER32	RW	pulse
HM	607Ch	Home offset	INTEGER32	RW	-
PP	6081h	Profile velocity	UNSIGNED32	RW	0.1rpm

Mode	Index	Name	Type	Attr.	unit
PV/PP	6083h	profile_acceleration	UINT32	RW	millisecond from 0rpm to 3000rpm
PV/PP	6084h	profile_deceleration	UINT32	RW	millisecond from 0rpm to 3000rpm
PT	6087h	Torque slope	INTEGER32	RW	-
PP/HM	6093h	Position factor	UNSIGNED32	RW	-
HM	6098h	Homing method	INTEGER8	RW	-
HM	6099h	Homing speeds	ARRAY	RW	0.1rpm
HM	609Ah	Homing acceleration	UNSIGNED32	RW	millisecond from 0rpm to 3000rpm
PP	60F4h	Following error actual value	INTEGER32	RO	pulse
PP	60FCh	Position demand value	INTEGER32	RO	pulse
ALL	60FDh	Digital input	UNSIGNED32	RO	-
ALL	60FEh	Digital output	UNSIGNED32	RO	-
PV	60FFh	target_velocity	INTEGER32	RW	0.1rpm

注意：PP 为协议位置模式，PV 为协议速度模式，PT 为转矩模式，HM 为回原点模式，ALL 为全部模式。

3.5 CANopen 数据帧和标识符

CANOpen 协议的通讯对象主要利用了 CAN 协议中的数据帧和远程帧。为了区分不同的通讯对象，CANOpen 协议利用数据帧/远程帧中的 ID。其中第 7 位到第 10 位为功能代码。第 0 位到第 6 位为节点 ID，用以区分不同节点的相同功能。这样就允许最多 127 个从节点与主节点通讯。CANopen 协议数据帧结构如下：

帧头	仲裁域		控制域	数据域	校验域	应答域	帧尾
	COB-ID（通讯对象标识符）	RTR（远程请求）					
1 位	11 位	1 位	6 位	0～8 字节	16 位	2 位	7 位

RTR 为远程帧标识，置为 0 表示该报文为数据帧，置为 1 表示该报文为远程帧。而本文描述的 CANopen 指令语句，皆以 COB-ID + DATA 的格式呈现，例如指令 “601 2B 17 10 00 00 00 00 00”中，“601”为 COB-ID，后续的 8 个 byte 为数据域。

每个 CANopen 数据帧都以 COB-ID 开头，COB-ID 用作 CAN 帧的标识符。在配置阶段，接受帧的 COB-ID 的节点是帧的提供者或使用者。COB_ID 越小报文优先级别越高。CANopen 通信对象 ID(COB-ID)结构如下：

COB-ID										
功能码				节点 ID(NODE ID)						
10	9	8	7	6	5	4	3	2	1	0

下面是预定义 CAN 标识符分配表：

CANopen 预定义主/从连接集的广播对象			
对象	功能码（ID-bits 10-7）	COB-ID	通讯参数在 OD 中的索引
NMT Module Control	0	000H	-
SYNC	1	080H	1005H，1006H，1007H
TIME SSTAMP	10	100H	1012H，1013H
CANopen 主/从连接集的对等对象			
对象	功能码（ID-bits 10-7）	COB-ID	通讯参数在 OD 中的索引
EMCY	1	081H-0FFH	1014H，1015H
PDO1(发送)	11	181H-1FFH	1800H
PDO1(接收)	100	201H-27FH	1400H
PDO2(发送)	101	281H-2FFH	1801H
PDO2(接收)	110	301H-37FH	1401H
PDO3(发送)	111	381H-3FFH	1802H
PDO3(接收)	1000	401H-47FH	1402H
PDO4(发送)	1001	481H-4FFH	1803H
PDO4(接收)	1010	501H-57FH	1403H
SDO(发送/服务器)	1011	581H-5FFH	1200H
SDO(接收/客户)	1100	601H-67FH	1200H
Heartbeat(心跳报文)	1110	701H-77FH	1016H，1017H

3.6 CANopen 报文简介

CANopen 通讯模型定义了如下几种报文（通讯对象）：

SDO	Service Data Object	用于非时间关键数据，如驱动器参数的设置
PDO	Porcess Data Objec	快速过程数据交互（如：实际位置）
EMCY	Emergency Message	故障信息传输
SYNC	Synchronization Message	同步信息：多个 CAN 节点的同步
NMT	Network Management	网络服务：例如，可以同时激活所有的 CAN 节点

3.6.1 NMT 管理报文

NMT（Network Management）管理报文用来实现 CANopen 网络中主节点对从节点监控和管理，如初始化、启动和停止节点，侦测失效节点。这种服务主要采用主从通讯模式来实现，此消息不需要应答。只有主节点能够发送 NMT Modual Control 报文。

NMT 消息格式

COB-ID	DLC	Byte0	Byte1
0x000	2	命令字	Node-ID

COB-ID 为 0x000，Byte0 为命令字，占一个字节，Byte 1 为 CANopen 网络设备节点地址，0 表示广播。

例：发送：000h 01h 06h　// 将节点 6 置为可操作状态；
　　发送：000h 01h 00h　// 将所有节点置为可操作状态；

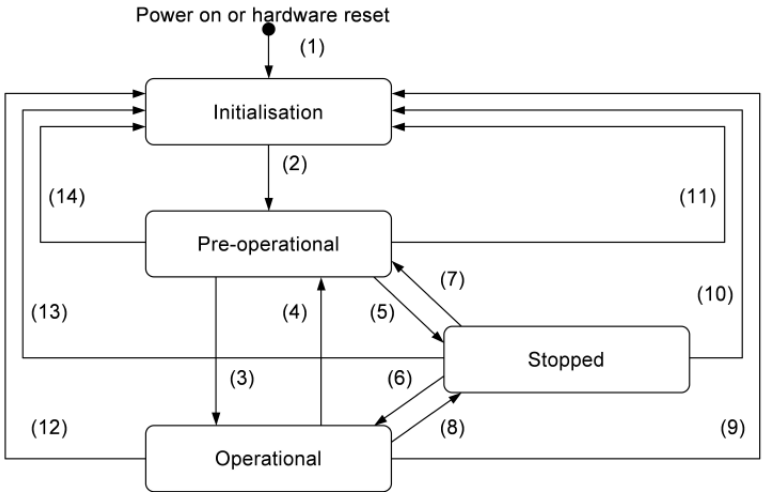
命令字具体格式如下：

命令字	NMT 服务	
1(0x01)	Start Remote Node	节点置为可操作状态
2(0x02)	Stop Remote Node	节点置为停止状态
128(0x80)	Enter Pre-operational State	节点置为预操作状态
129(0x81)	Reset Node	节点置为应用重置状态
130(0x82)	Reset Communication	节点置为通讯重置状态

NMT 节点保护（NMT Node Guarding）

通过节点保护服务，MNT 主节点可以检查每个节点的当前状态，当这些节点没有数据传送时这种服务尤其有意义。节点保护分为心跳报文（Heartbeat）和心跳监护（Life Guard）。

NMT 状态机如下：



Transition	Description
(1)	At Power on, the initialisation state is entered autonomously
(2)	Once initialisation is finished, the Pre-Operational state is automatically entered
(3), (6)	Start_Remote_Node
(4), (7)	Enter_Pre-Operational_State
(5), (8)	Stop_Remote_Node
(9), (10), (11)	Reset_Node
(12), (13), (14)	Reset_Communication

心跳报文（Heartbeat）

驱动器上电后，会主动上报一包上线报文。通过设置心跳，可以以一个固定的频率发送心跳报文，用于告诉主机自己的状态。报文的格式：COB-ID 为 0x700+ ID，数据为一字节的状态数据。例如 ID 为 2 的驱动器在接入 CAN 总线后会主动发送报文：0x70200。心跳报文格式如下：

COB-ID	DLC	Byte0	状态含义
0x700+Node_ID	1	0x00	BOOTUP 启动状态
		0x04	STOPPED 停止
		0x05	OPERATIONAL 可操作
		0x7F	PRE-OPERATIONAL 预操作

心跳间隔为字典 0x1017 设置值(单位 ms)，设置 0x1017 为 0，则禁止心跳报文，设置 0x1017 为非 0，则使能心跳报文。例如：

	COB-ID	DATA	注释
发送	602	2B 17 10 00 00 00 00 00	禁止心跳报文
发送	602	2B 17 10 00 64 00 00 00	使能心跳报文，心跳间隔 100ms

心跳监测（Life Guard）

主站可以通过 Life Guard 来监测从站状态：

	COB-ID	DLC	Byte0
主站问	0x700+Slave_ID	0	-
从站答	0x700+Slave_ID	1	State

State 状态字节含义见上一节，低 7 位为状态含义，最高位为翻转位，这个位的值必须在两个连续的响应中交替出现，例程如下：

例 1：上电监控从站状态：从站处于 PRE-OPERATIONAL 状态（以 2 号 ID 为例）

	COB-ID	DATA	注释
发送	702	-	Life guard 状态询问
回复	702	FF	状态回复,1 个字节长度
发送	702	-	Life guard 状态持续询问
回复	702	7F	状态回复,最高位翻转

例 2：主站发送激活后：从站处于 OPERATIONAL 状态（以 2 号站为例）

	COB-ID	DATA	注释
发送	000	01 02	激活节点 2
发送	702	-	Life guard 状态询问
回复	702	85	状态回复,1 个字节长度
发送	702	-	Life guard 状态持续询问
回复	702	05	状态回复,最高位翻转

3.6.2 EMCY 紧急报文

驱动器在正常情况下，错误码[注] 603Fh 值为 0。驱动器在故障报警时，会通过紧急报文（Emergency Message）主动上传一次错误码，同时 603Fh 显示相关错误内容。错误码上传模式单轴驱动器可配置为定时上传模式，上传时间由驱动器内部参数控制。紧急报文格式如下：

COB-ID	Byte0	Byte1
0x080+Node_ID	错误码低位	错误码高位

报警发生后，详细信息请参见**第四章 报警信息**。建议用户通过报警码排查报警原因，软报警可以通过控制字 6040h、Fn 参数或者 IO 清除。

注：二合一驱动部分型号不完全支持 603Fh，故障码存在 0x5012 中。

另外，如果驱动器使能标志位上报功能，当触发以下条件时，也会主动通过 EMCY 向主站报告：

信号名称	传输码	解析
CCW_Disable	0x8681	正限位
CW_Disable	0x8682	负限位
POS_Arrived	0x8683	位置到达
Home_Done	0x8684	回原点完成

3.6.3 SDO 服务数据对象

SDO（Service Data Object）主要用来访问节点的对象字典，主站问，从站答。主要用来在设备之间传输低优先级的对象，典型是对从设备进行配置、管理，比如模式配置、PDO 配置等，类似于 Modbus 的一问一答体系。

指令结构如下：

<table>
<tr><td colspan="2" rowspan="2">COB_ID</td><td>Byte0</td><td>Byte 1</td><td>Byte 2</td><td>Byte 3</td><td>Byte 4</td><td>Byte 5</td><td>Byte 6</td><td>Byte 7</td></tr>
<tr><td>CMD</td><td colspan="2">INDEX</td><td rowspan="11">SUB INDEX</td><td colspan="4">DATAs</td></tr>
<tr><td rowspan="5">问</td><td rowspan="5">0x600+ Node_ID</td><td>0x23 写 4byte</td><td rowspan="10" colspan="2">Fn 参数地址
Dn 参数地址
Pn 参数地址
对象字典 Index</td><td rowspan="2">DATA1</td><td rowspan="2">DATA2</td><td>DATA3</td><td>DATA4</td></tr>
<tr><td>0x2B 写 2byte</td><td rowspan="3">0x00</td><td rowspan="3">0x00</td></tr>
<tr><td>0x2F 写 1byte</td><td rowspan="2">0x00</td></tr>
<tr><td>0x40 读数据</td><td rowspan="5">DATA1</td></tr>
<tr><td rowspan="5">答</td><td rowspan="5">0x580+ Node_ID</td><td>0x43 回 4byte</td><td>DATA2</td><td>DATA3</td><td>DATA4</td></tr>
<tr><td>0x4B 回 2byte</td><td rowspan="2">0x00</td><td rowspan="2">0x00</td></tr>
<tr><td>0x4F 回 1byte</td><td>0x00</td></tr>
<tr><td>0x60 写回复</td><td>DATA2</td><td>DATA3</td><td>DATA4</td></tr>
<tr><td>0x80 错误码</td><td colspan="3"></td></tr>
</table>

例 1：读单轴驱动器速度 Dn00（若 CAN-ID 为 2）

方向	COB-ID	DATA（Hex）	解析
发送	602	40 00 30 00 00 00 00 00	读当前速度
回复	582	4B 00 30 00 64 00 00 00	转速为100rpm

例 2：写单轴驱动器内部速度 Fn33（若 CAN-ID 为 2）

方向	COB-ID	DATA（Hex）	解析
发送	602	2B 33 20 00 C8 00 00 00	Fn33写入200RPM
回复	582	60 33 20 00 C8 00 00 00	回复

注意：在 CANopen 指令中，INDEX 和 DATA 都为低字节在前。

3.6.4 PDO 过程数据对象

PDO（Process Data Object）是用来发送（TPDO）或者接收（RPDO）数据的，实现实时数据的传输，只传不回。1 个 PDO 单次最多传输 8 个字节的数据。以下为 PDO 的对象字典参数和例程：

映射参数 Index		通信参数 Index	
RPDO1~4	TPDO1~4	RPDO1~4	TPDO1~4
1600h~1603h	1A00h~1A03H	1400h~1403h	1800h~1803h

单轴驱动器支持设置 4 个 TPDO 或 RPDO，支持断电保存 PDO 的设置，信息保存在参数体系 1 的 Fn100~Fn17C 中，如下所示：

PDO 类型	RPDO1	RPDO2	RPDO3	RPDO4
PDO 对象数	Fn100	Fn110	Fn120	Fn130
传输 ID	Fn101	Fn111	Fn121	Fn131
传输类型	Fn102	Fn112	Fn122	Fn132
上报时间	Fn103-104	Fn113-114	Fn123-124	Fn133-134
SUB1 索引	Fn105	Fn115	Fn125	Fn135
SUB1 子索引	Fn106	Fn116	Fn126	Fn136
SUB2 索引	Fn107	Fn117	Fn127	Fn137
SUB2 子索引	Fn108	Fn118	Fn128	Fn138
SUB3 索引	Fn109	Fn119	Fn129	Fn139
SUB3 子索引	Fn10A	Fn11A	Fn12A	Fn13A
SUB4 索引	Fn10B	Fn11B	Fn12B	Fn13B
SUB4 子索引	Fn10C	Fn11C	Fn12C	Fn13C

PDO 类型	TPDO1	TPDO2	TPDO3	TPDO4
PDO 对象数	Fn140	Fn150	Fn160	Fn170
传输 ID	Fn141	Fn151	Fn161	Fn171
传输类型	Fn142	Fn152	Fn162	Fn172
上报时间	Fn143-144	Fn153-154	Fn163-164	Fn173-174
SUB1 索引	Fn145	Fn155	Fn165	Fn175
SUB1 子索引	Fn146	Fn156	Fn166	Fn176
SUB2 索引	Fn147	Fn157	Fn167	Fn177
SUB2 子索引	Fn148	Fn158	Fn168	Fn178
SUB3 索引	Fn149	Fn159	Fn169	Fn179
SUB3 子索引	Fn14A	Fn15A	Fn16A	Fn17A
SUB4 索引	Fn14B	Fn15B	Fn16B	Fn17B
SUB4 子索引	Fn14C	Fn15C	Fn16C	Fn17C

双轴驱动器支持每个轴设置4个TPDO或RPDO，支持断电保存PDO的设置。信息保存在Fx.7.000~Fx.7.07F中。每个PDO配置信息占用13个Fn寄存器。PDO每组寄存器结构如下（参数体系2）：

PDO 类型	A 轴 RPDO1	A 轴 RPDO2	A 轴 RPDO3	A 轴 RPDO4
PDO 对象数	F0.7.000	F0.7.010	F0.7.020	F0.7.030
传输 ID	F0.7.001	F0.7.011	F0.7.021	F0.7.031
传输类型	F0.7.002	F0.7.012	F0.7.022	F0.7.032
上报时间	F0.7.003 F0.7.004	F0.7.013 F0.7.014	F0.7.023 F0.7.024	F0.7.033 F0.7.034
SUB1 索引	F0.7.005	F0.7.015	F0.7.025	F0.7.035
SUB1 子索引	F0.7.006	F0.7.016	F0.7.026	F0.7.036
SUB2 索引	F0.7.007	F0.7.017	F0.7.027	F0.7.037
SUB2 子索引	F0.7.008	F0.7.018	F0.7.028	F0.7.038
SUB3 索引	F0.7.009	F0.7.019	F0.7.029	F0.7.039
SUB3 子索引	F0.7.00A	F0.7.01A	F0.7.02A	F0.7.03A
SUB4 索引	F0.7.00B	F0.7.01B	F0.7.02B	F0.7.03B
SUB4 子索引	F0.7.00C	F0.7.01C	F0.7.02C	F0.7.03C

PDO 类型	A 轴 TPDO1	A 轴 TPDO2	A 轴 TPDO3	A 轴 TPDO4
PDO 对象数	F0.7.040	F0.7.050	F0.7.060	F0.7.070
传输 ID	F0.7.041	F0.7.051	F0.7.061	F0.7.071
传输类型	F0.7.042	F0.7.052	F0.7.062	F0.7.072
上报时间	F0.7.043 F0.7.044	F0.7.053 F0.7.054	F0.7.063 F0.7.064	F0.7.073 F0.7.074
SUB1 索引	F0.7.045	F0.7.055	F0.7.065	F0.7.075
SUB1 子索引	F0.7.046	F0.7.056	F0.7.066	F0.7.076
SUB2 索引	F0.7.047	F0.7.057	F0.7.067	F0.7.077
SUB2 子索引	F0.7.048	F0.7.058	F0.7.068	F0.7.078
SUB3 索引	F0.7.049	F0.7.059	F0.7.069	F0.7.079
SUB3 子索引	F0.7.04A	F0.7.05A	F0.7.06A	F0.7.07A
SUB4 索引	F0.7.04B	F0.7.05B	F0.7.06B	F0.7.07B
SUB4 子索引	F0.7.04C	F0.7.05C	F0.7.06C	F0.7.07C

PDO 类型	B 轴 RPDO1	B 轴 RPDO2	B 轴 RPDO3	B 轴 RPDO4
PDO 对象数	F1.7.000	F1.7.010	F1.7.020	F1.7.030
传输 ID	F1.7.001	F1.7.011	F1.7.021	F1.7.031
传输类型	F1.7.002	F1.7.012	F1.7.022	F1.7.032
上报时间	F1.7.003 F1.7.004	F1.7.013 F1.7.014	F1.7.023 F1.7.024	F1.7.033 F1.7.034
SUB1 索引	F1.7.005	F1.7.015	F1.7.025	F1.7.035
SUB1 子索引	F1.7.006	F1.7.016	F1.7.026	F1.7.036
SUB2 索引	F1.7.007	F1.7.017	F1.7.027	F1.7.037
SUB2 子索引	F1.7.008	F1.7.018	F1.7.028	F1.7.038
SUB3 索引	F1.7.009	F1.7.019	F1.7.029	F1.7.039
SUB3 子索引	F1.7.00A	F1.7.01A	F1.7.02A	F1.7.03A
SUB4 索引	F1.7.00B	F1.7.01B	F1.7.02B	F1.7.03B
SUB4 子索引	F1.7.00C	F1.7.01C	F1.7.02C	F1.7.03C

PDO 类型	B 轴 TPDO1	B 轴 TPDO2	B 轴 TPDO3	B 轴 TPDO4
PDO 对象数	F1.7.040	F1.7.050	F1.7.060	F1.7.070
传输 ID	F1.7.041	F1.7.051	F1.7.061	F1.7.071
传输类型	F1.7.042	F1.7.052	F1.7.062	F1.7.072
上报时间	F1.7.043 F1.7.044	F1.7.053 F1.7.054	F1.7.063 F1.7.064	F1.7.073 F1.7.074
SUB1 索引	F1.7.045	F1.7.055	F1.7.065	F1.7.075
SUB1 子索引	F1.7.046	F1.7.056	F1.7.066	F1.7.076
SUB2 索引	F1.7.047	F1.7.057	F1.7.067	F1.7.077
SUB2 子索引	F1.7.048	F1.7.058	F1.7.068	F1.7.078
SUB3 索引	F1.7.049	F1.7.059	F1.7.069	F1.7.079
SUB3 子索引	F1.7.04A	F1.7.05A	F1.7.06A	F1.7.07A
SUB4 索引	F1.7.04B	F1.7.05B	F1.7.06B	F1.7.07B
SUB4 子索引	F1.7.04C	F1.7.05C	F1.7.06C	F1.7.07C

例如：如果要配置RPDO1为6040H和6060H，配置RPDO2为6083H和6084H，那么参数体系1和参数体系2需要设置的Fn参数如下：

参数体系	RPDO1		RPDO2	
	参数	设值	参数	设值
参数体系 1	Fn100	2	Fn110	2
	Fn101	513	Fn111	769
	Fn105	24640	Fn115	24707
	Fn107	24672	Fn117	24708
参数体系 2 A 轴	F0.7.000	2	F0.7.010	2
	F0.7.001	513	F0.7.011	769
	F0.7.005	24640	F0.7.015	24707
	F0.7.007	24672	F0.7.017	24708
参数体系 2 B 轴	F1.7.000	2	F1.7.010	2
	F1.7.001	513	F1.7.011	769
	F1.7.005	24640	F1.7.015	24707
	F1.7.007	24672	F1.7.017	24708

例如：如果要配置 TPDO1 为 6041H、6060H 和 603F，配置 TPDO2 为 6063H 和 6069H，上传模式为同步模式 1，那么参数体系 1 和参数体系 2 需要设置的 Fn 参数如下：

参数体系	TPDO1		TPDO2	
	参数	设值	参数	设值
参数体系 1	Fn140	3	Fn150	2
	Fn141	385	Fn151	641
	Fn142	1	Fn152	1
	Fn143	2	Fn153	2
	Fn144	2	Fn154	2
	Fn145	24641	Fn155	24675
	Fn147	24672	Fn157	24681
	Fn149	24639	Fn159	0

参数体系	TPDO1		TPDO2	
	参数	设值	参数	设值
参数体系 2 A 轴	F0.7.040	3	F0.7.050	2
	F0.7.041	385	F0.7.051	641
	F0.7.042	1	F0.7.052	1
	F0.7.043	2	F0.7.053	2
	F0.7.044	2	F0.7.054	2
	F0.7.045	24641	F0.7.055	24675
	F0.7.047	24672	F0.7.057	24681
	F0.7.049	24639	F0.7.059	0
参数体系 2 B 轴	F1.7.040	3	F1.7.050	2
	F1.7.041	385	F1.7.051	641
	F1.7.042	1	F1.7.052	1
	F1.7.043	2	F1.7.053	2
	F1.7.044	2	F1.7.054	2
	F1.7.045	24641	F1.7.055	24675
	F1.7.047	24672	F1.7.057	24681
	F1.7.049	24639	F1.7.059	0

RPDO配置

RPDO主要用于快速变化数据的传输，例如速度、控制字等对象字典。例如，以下RPDO1和RPDO2的配置，若CANopen节点号设为2：

	序号	报文内容	解析
RPDO1 0x202	1	602 2F 00 16 00 00 00 00 00	RPDO1 stop
	2	602 23 00 16 01 20 00 FF 60	60FFh映射至RPDO1 sub1
	3	602 2F 00 16 00 01 00 00 00	RPDO1 enable
RPDO2 0x302	1	602 2F 01 16 00 00 00 00 00	RPDO2 stop
	2	602 23 01 16 01 10 00 40 60	6040h映射至RPDO2 sub1
	3	602 23 01 16 02 20 00 FF 60	60FFh映射至RPDO2 sub2
	4	602 2F 01 16 00 02 00 00 00	RPDO2 enable

由上例配置完毕，发送“202 64 00 00 00”表示通过RPDO1发送给60FFh的速度值为100。发送“302 06 00 64 00 00 00”表示通过PRDO2发送给6040h的值为6，发送给60FFh的值为100，分别对应。数据域长度为所配置对象字典的总长度。

1600h~1603h的子索引（=0）表示RPDO的启用或停止，当DATA1为0时为停止，DATA1不为0时，表示当前RPDO所配的对象字典数量，并且启动当前RPDO。

1600h~1603h的子索引（>0）表示RPDO的对象字典序列，其数据域DATA1表示该对对象字典的长度，一般为8位、16位或32位，数据域DATA2表示对象字典子索引，数据域DATA3~DATA4为对象字典。

TPDO配置

TPDO用于快速传输用户需求的信息，例如位置、状态等对象字典。TPDO有异步触发、同步触发两类触发方式，可通过SDO配置相关对象字典来修改。

例如，以下TPDO1和TPDO2的一种配置，若CANopen对象节点号为2：

	序号	报文内容	解析	
TPDO1 0x182	1	602 2F 00 1A 00 00 00 00 00	TPDO1 stop	本例是将对象字典6041H设为TPDO1，让其以50ms的频率向总线上报；上报方式为254；
	2	602 23 00 1A 01 10 00 41 60	6041h	
	3	602 2F 00 1A 00 01 00 00 00	TPDO1 enable	
	4	602 2F 00 18 05 32 00 00 00	设置触发时间50ms	
	5	602 2F 00 18 03 32 00 00 00		
	6	602 2F 00 18 02 FE 00 00 00	254上报方式	
TPDO2 0x282	1	602 2F 01 1A 00 00 00 00 00	TPDO2 stop	本例是将对象字典6063H和6069H设为TPDO2，让其以50ms的频率同时向总线上报；上报方式为254；
	2	602 23 01 1A 01 20 00 63 60	6063h	
	3	602 23 01 1A 02 20 00 69 60	6069h	
	4	602 2F 01 1A 00 02 00 00 00	TPDO2 enable	
	5	602 2F 01 18 05 32 00 00 00	设置触发时间50ms	
	6	602 2F 01 18 03 32 00 00 00		
	7	602 2F 01 18 02 FE 00 00 00	254上报方式	

1A00h~1A03h的数据结构参考1600h~1603h。

1800h~1803h的子索引(=5，=3)表示TPDO的触发时间，触发时间在DATA1-DATA4里，时间单位为ms；子索引（=2）表示TPDO的触发方式，触发方式在DATA1中，其中1-240方式为同步触发（响应同步帧SYNC）；254表示以触发时间定时触发；255方式表示事件触发（状态变化上传）。

3.7 Ví dụ set tham số CANopen

Chú ý: Khi dùng master-slave, bắt buộc phát set node đó là controlable status. Driver phải set thành CANopen mode, sau khi sửa mode điều khiển phải reset driver mới có tác dụng.

3.7.1 Profile Velocity Mode

Ở mode vận tốc, bắt buộc phải set các object sau: 6060H, 60FFH, 6083H, 6084H (6083H/6084H mặc định là 1000ms).

Dưới đây là ví dụ các bước config điều khiển SDO, node ID là 2. Tùy theo nhu cầu, user tự config PDO.

Step1: Đảm bảo driver đang ở mode CANopen
Hệ tham số 1: Fn00=20; hệ tham số 2: F0/1.1.002=20;
Step2: Node được set thành controllable status:
send: 000 01 02 set node 2 thành controllabe staus
Step3: Config
1. Set mode làm việc (6060h) làm mode vận tốc:
send: 602 2F 60 60 00 03 00 00 00 // set 6060h thành 3 (mode vận tốc).

2. Set thời gian thay đổi tốc độ từ 0rpm đến 3000 rpm (gia tốc 6083H, giảm tốc 6084H) đơn vị ms, maximum 200K):

Send: 602 23 83 60 00 E8 03 00 00 //Set 6083h =1s
Send: 602 23 84 60 00 E8 03 00 00 //Set 6084h =1s

3. Set tốc độ mục tiêu: 60FFh, đơn vị 0,1 rpm

Send: 602 23 FF 60 00 64 00 00 00 // Set 60FFh =10rpm

4. Set controlword 6040h để enable driver:

Lần đầu enable driver phải ghi lần lượt 0x06, 0x07, 0x0F. Sau đó nếu ghi 0x07 thì disable driver, ghi tiếp 0x0F mới enable driver.

Send: 602 2B 40 60 00 06 00 00 00 // Set 6040h =6 driver chuẩn bị
Send: 602 2B 40 60 00 07 00 00 00 // Set 6040h =7 disable
Send: 602 2B 40 60 00 0F 00 00 00 // Set 6040h =F enable driver

Động cơ chạy với tốc độ 60FFh sau khi enable xong.

STEP4. Ánh xạ TPDO

Ví dụ dưới đây config 2 TPDO: config velocity_sensor_actual_value:6069h và Position actual value:6063h thành TPDO1, config Statusword 6041h thành TPDO2, phương thức upload TPDO là phương thức updoad đồng bộ.

TPDO1: COB-ID tương ứng là 0x182

Send: 602 2F 00 1A 00 00 00 00 00
Send: 602 23 00 1A 01 20 00 69 60 // tốc độ thực tế tương ứng với object 6069h
Send: 602 23 00 1A 02 20 00 63 60 // vị trí encoder tương ứng với object 6063h
Send: 602 2F 00 1A 00 02 00 00 00
Send: 602 2F 00 18 02 01 00 00 00 // mode upload đồng bộ
Send: 602 2F 00 18 03 02 00 00 00
Send: 602 2F 00 18 05 02 00 00 00

TPDO2: COB-ID tương ứng là 0x282

发送: 602 2F 01 1A 00 00 00 00 00
发送: 602 23 01 1A 01 10 00 41 60 // Statusword tương ứng với object 0x6041
发送: 602 2F 01 1A 00 01 00 00 00
发送: 602 2F 01 18 02 01 00 00 00 // mode upload đồng bộ
发送: 602 2F 01 18 03 02 00 00 00
发送: 602 2F 01 18 05 02 00 00 00

5. Send frame đồng bộ để nhận được bản tin TPDO

Send: 080 // Send frame đồng bộ
Feedback1: 182 byte0-7 //TPDO1 bản tin upload
Feedback2: 282 byte0-1 //TPDO2 bản tin upload

Chú ý:

1. Ở mode điều khiển tốc độ, master node sẽ nhận được các thông tin sau:
- Xác nhận Velocity demand value 606Bh nhận được giá trị vận tốc nội bộ (đơn vị 0.1 rpm)
- Xác nhận Velocity_sensor_actual_value 6069h nhận được giá trị tốc độ thực tế (đơn vị 0.1 rpm)

2. Master có thể set các giá trị ngưỡng theo dõi:
- Set Velocity window 606Dh, để chia vùng đạt đến vận tốc mục tiêu (đơn vị 0.1 rpm)
- Set Velocity window time 606Eh, thời gian đạt đến vận tốc mục tiêu (đơn vị ms)
- Set Velocity threshold 6060Fh, set ngưỡng tốc độ 0 (0.1 rpm)
- Set Velocity window time 6070h, thời gian đạt vận tốc 0 (đơn vị ms)

3. Tốc độ mục tiêu 60FFh là số 32 bit có dấu, âm dương tương ứng với chiều quay.

3.7.2 协议位置模式（Profile Position Mode）

驱动器接收主站位置指令可控制电机到达目标位置，该模式下，必须设置的对象字典有：6060H、607AH、6081H、6083H、6084H、6093H。

以目标驱动器的CAN站号为2，通过SDO控制为例，以下为配置步骤。用户可根据需求自行设置PDO映射。

STEP1. 确认驱动器处于CANopen控制模式

参数体系1：Fn00=20；参数体系2：F0/1.1.002=20；

STEP2. 节点置为可操作状态:

发送：000 01 02——将节点2置为可操作状态；

STEP3. 设置

1. 设置运行模式【Mode of operations:6060h】为位置模式
   发送：602 2F 60 60 00 01 00 00 00 //设置6060h为1（Profile Position Mode）
2. 设置目标位置【Target position:607Ah】. (单位: pulse)
   发送：602 23 7A 60 00 60 EA 00 00 //60000脉冲
3. 设置速度【Profile velocity:6081h】. (单位: 0.1rpm)
   发送：602 23 81 60 00 B8 0B 00 00 //300rpm
4. 设置加速度和减速度【Profile acceleration:6083h；Profile deceleration:6084h】(单位时间：ms，从0rpm到3000rpm的时间，最大可设8K)：
   发送：602 23 83 60 00 E8 03 00 00
   发送：602 23 84 60 00 E8 03 00 00
5. 设置电子齿轮比【Position factor：6093h；Sub1：Numerator；Sub2：Divisor】
   发送：602 2B 93 60 01 01 00 00 00 //设置电子齿轮比分子
   发送：602 2B 93 60 02 01 00 00 00 //设置电子齿轮比分母
6. 设置控制字【Controlword:6040h】来使能电机：.
   发送：602 2B 40 60 00 06 00 00 00 // 设置6040h为6
   发送：602 2B 40 60 00 07 00 00 00 // 设置6040h为7
   发送：602 2B 40 60 00 0F 00 00 00 // 设置6040h为F，使电机使能

7. 触发电机运行（电机使能后）：

增量位置触发：

发送：602 2B 40 60 00 5F 00 00 00 //增量位置触发

发送：602 2B 40 60 00 4F 00 00 00 //bit4清零，为下次启动做准备

绝对位置触发（**注意：部分版本暂不支持**）：

发送：602 2B 40 60 00 1F 00 00 00 //绝对位置触发

发送：602 2B 40 60 00 0F 00 00 00 //清零

8. 确认位置实际值【Position actual value:6063h】得到电机位置反馈。
9. 确认状态字【Statusword:6041h】得到驱动器状态，例如错误位、目标到达等。

注意事项：

1、驱动器是以6040h的bit4的上升沿接收新的位置命令，所以每次执行完一次运行后需要把此位清零。

2、上位机可根据驱动器的状态字6041h来判断是否要给伺服新数据。当驱动器的状态字6041的bit10为0，表示电机正在执行中；为1时，表示电机已到达位置，驱动器可以接收新的位置数据和命令了。为了消除抖动，建议主站设置位置到达阈值：

设置【Position window: 6067h】位置到达区（建议设为10）。

设置【Position window time: 6068h】位置到达时间（建议设为10）。

3、在增量位置模式下，电机在执行命令中接收到新的位置指令，则新命令暂不被采纳，而会保存在缓存区。驱动器最大可缓存32条位置触发信息（每条触发指令含加减速、目标脉冲和目标速度四个对象字典内容），超过缓存数的位置指令将丢失，所以请勿高速更新位置指令。

4、在增量位置模式下，驱动器支持位置触发后的速度修改。

5、在绝对位置模式下，电机在执行命令中接收到新的位置指令，则执行最新的位置指令，原位置不再执行。

3.7.3 转矩模式（Torque Profile Mode）

驱动器支持转矩模式运行，在该模式下，必须设置的对象字典有：6060H、6087H、6071H。以 CAN 节点 2 为例，以下为配置：

STEP1. 确认驱动器处于CANopen控制模式

参数体系1：Fn00=20；参数体系2：F0/1.1.002=20；

STEP2.节点置为可操作状态:

发送：000 01 02——将节点2置为可操作状态；

STEP3. 设置

1. 设置运行模式【Mode of operations:6060h】为转矩模式：

发送：602 2F 60 60 00 04 00 00 00 //设置6060h为4（转矩模式）

2. 设置加减速【Torque slope:6087h】：

发送：602 2F 87 60 00 E8 03 00 00 //设置6087H为1000；

3. 设置目标转矩【Target torque:6071h】:

发送：602 2F 71 60 00 64 00 00 00 // 100对应电机转矩的10%;

4. 设置状态字【Controlword:6040h】来使能电机:

发送：602 2B 40 60 00 06 00 00 00 //设置6040h为6

发送：602 2B 40 60 00 07 00 00 00 //设置6040h为7

发送：602 2B 40 60 00 0F 00 00 00 //设置6040h为F，使电机使能;

5. 确认状态字【Statusword:6041h】得到驱动器状态。

注意事项:

转矩模式下，参数体系 1 型驱动器最大限制速度为 Fn95，地址为 0x2095。参数体系 2 型驱动器最大限制速度为 F0/1.4.026，地址分别为 0x2426 和 0x3426。

3.7.4 回零点模式（Homing Mode）

用户可设置速度、加减速等配置回零模式。动器支持13种回零方式，对象字典0x6098的不同回零方式定义如下:

Index	Sub-Index	设定值	回零方式
0x6098	0x00	-1	反向回零,找Z,转矩到达信号为零位信号
		-2	正向回零,找Z,转矩到达信号为零位信号
		-17	反向回零,不找Z,转矩到达信号为零位信号
		-18	正向回零,不找Z,转矩到达信号为零位信号
		1	反向回零,找Z,负限位信号为零位信号
		2	正向回零,找Z,正限位信号为零位信号
		3	反向回零,找Z,机械零点信号为零位信号
		4	正向回零,找Z,机械零点信号为零位信号
		17	反向回零,不找Z,负限位信号为零位信号
		18	正向回零,不找Z,正限位信号为零位信号
		19	反向回零,不找Z,机械零点信号为零位信号
		20	正向回零,不找Z,机械零点信号为零位信号
		35	不回零

以CAN节点2为例，配置步骤如下:

STEP1. 确认驱动器处于CANopen控制模式

参数体系1：Fn00=20；参数体系2：F0/1.1.002=20;

STEP2. 节点置为可操作状态:

发送：000 01 02——将节点2置为可操作状态;

STEP3、设置

1. 设置运行模式【Mode of operations:6060h】为回原点模式:

发送：602 2F 60 60 00 06 00 00 00//设置6060h为6（回原点模式）

2. 设置回零方式【Homing method:6098h】:

发送：602 2F 98 60 00 14 00 00 00//设置6098H为20

3. 设置回零速度【Homing speeds:6099h】:
   发送：602 23 99 60 01 D0 07 00 00
   发送：602 23 99 60 02 C8 00 00 00
4. 设置回零加减速【Homing acceleration:609Ah】:
   发送：602 23 9A 60 00 E8 03 00 00
5. 设置电子齿轮比【Position factor:6093h】:
   发送：602 23 93 60 01 01 00 00 00 //设置6093H，齿轮比分子1;
   发送：602 23 93 60 02 01 00 00 00 //设置6093H，齿轮比分母1;
6. 设置控制字【Controlword:6040h】来使能电机：.
   发送：602 2B 40 60 00 06 00 00 00 //设置6040h为6
   发送：602 2B 40 60 00 07 00 00 00 //设置6040h为7
   发送：602 2B 40 60 00 0F 00 00 00 //设置6040h为F，使电机使能;
7. 触发电机运行:
   发送：602 2B 40 60 00 0F 00 00 00 //确保电机在使能状态
   发送：602 2B 40 60 00 5F 00 00 00
   发送：602 2B 40 60 00 4F 00 00 00 //触发，则电机动作

注意：零点限位开关在输入口定义中，应设为18。

3.8 Giao thức CAN tự định nghĩa

3.8.1 Mode tốc độ (hệ tham số 1)

用户可在内部速度模式（Fn00=1）下，直接通过 SDO 读写驱动器 Fn 功能参数控制电机。读取 Dn 状态观测参数，以及设置 TPDO 上报信息可辅助用户的控制。示例如下:

相关参数:

Index	Name	Type	Attr.	unit
2000h	Fn00 控制模式	UNSIGNED16	RW	-
2033h	Fn33 内部速度	INTEGER16	RW	1rpm
2038h	Fn38 使能	UNSIGNED16	RW	-
2099h	Fn99 通讯指令保存开关	UNSIGNED16	RW	-
20AAh	FnAA 速度模式减速倍率	UNSIGNED16	RW	-

配置步骤（以CAN节点2为例）:

用户可通过手持面板修改，也可通过通讯修改参数，具体步骤如下:

STEP1. 确认驱动器参数:
Fn00 = 1（速度模式）;

Fn33 = 0（初始上电速度为0）；
Fn38 = 0（初始状态不使能）；
Fn99 = 1（通讯修改参数不保存）.

STEP2. 设置：

1. 设置电机速度Fn33：
   发送：602 2B 33 20 00 64 00 00 00 //设置速度为100rpm
2. 使能电机：
   发送：602 2B 38 20 00 01 00 00 00 //设置 Fn38=1，使电机使能；

3.读 Dn00 电机转速：
   发送：602 40 00 30 00 00 00 00 00 //读速度，单位 rpm

注意事项：

1、部分 CANopen 对象字典也可直接读取，例如读编码器绝对位置，在 Modbus 下可以读 Dn2E 和 Dn2F，而在 CAN 模式下，可以读对象字典 6063H：
   发送：602 40 63 60 00 00 00 00 00
   回复：582 43 63 60 00 D1 D2 D3 D4
   以上，D1-D4 分别为低位-高位；
2、DS、RS 型伺服驱动器支持本手册中对象字典，如用户需求也可定制增加相应功能，具体请联系和利时电机业务员及技术员。

3.8.2 Mode tốc độ (hệ tham số 2)

Người dùng có thể ở mode tốc độ nội bộ (F0.1.002/ F1.1.002=1) trực tiếp thông qua SDO đọc Fn, đọc Dn, và config TPDO.

Tham số liên quan:

Axis No.	Index	Name	Type	Attr.	unit
A	2102h	F0.1.002 A- control mode	UNSIGNED16	RW	
	2318h	F0.3.018 A- Tốc độ nội bộ	INTEGER16	RW	1rpm
	2100h	F0.1.000 A- enable	UNSIGNED16	RW	
B	3102h	F1.1.002 B- control mode	UNSIGNED16	RW	
	3318h	F1.3.018 B- Tốc độ nội bộ	INTEGER16	RW	1rpm
	3100h	F1.1.000 B- enable	UNSIGNED16	RW	
ALL	450Ah	Pn5.00A Lưu command truyền thông	UNSIGNED16	RW	

Các bước config (ví dụ trục A, node 2):

Người dùng có thể thay đổi tham số bằng bộ điều khiển cầm tay, hoặc qua truyền thông:

Bước 1. xác nhận tham số driver:

F0.1.002 = 1 Mode tốc độ
F0.3.018 = 0 Tốc độ lúc bật điện bằng 0
F0.1.000 = 0 Trạng thái ban đầu không enable
Pn5.00A = 0 Tham số thay đổi do truyền thông không bảo lưu

STEP2. Config

1. Set tốc độ F0.3.018:
   Send: 602 2B 18 23 00 64 00 00 00 // Set tốc độ trục A = 100rpm
2. Enable động cơ
   Send: 602 2B 00 21 00 01 00 00 00 // Set=1, ennable trục A
3. Đọc D0.000, tốc độ quay:
   Send: 602 40 00 50 00 00 00 00 00 // Đọc tốc độ quay của A, đơn vị rpm

3.8.3 Mode Diffential (2 bánh vi sai)

双轴（二合一）驱动器支持自定义的差速指令，即通过一条 CAN 报文同时控制双轴动作，从而达到双轴完全同步的控制。

STEP1. 确认驱动器参数：

F0/1.1.002 = 1（内部速度模式，改完重启生效）
F0/1.3.018 = 0（内部速度置为 0）
Pn5.00D = 1（根据需求设置差速上报开关）
Pn6.00A = ？（轮径 0.1mm，按实际填写）
Pn6.00B = ？（轮距 0.1mm，按实际填写）
Pn6.00C = 0（定义左右轮 0/1）

STEP2. 设置（目标ID以A轴为准）：

1、设置完毕后，第一步，先使能驱动器：
   发送：602 2B 00 21 00 01 00 00 00 //置 1，使 A 轴电机使能；
   发送：602 2B 00 31 00 01 00 00 00 //置 1，使 B 轴电机使能；

2、写差速指令，格式如下：

ID	指令				指令线速度		指令角速度	
0x600+Node_ID	EA	00	00	00	ByteVL	ByteVH	ByteAL	ByteAH

上报信息格式如下：

ID	指令				实际线速度		实际角速度	
0x180+Node_ID	EA	00	00	00	ByteVL	ByteVH	ByteAL	ByteAH

其中，线速度单位为 mm/s，角速度单位为 mrad/s，两个字节组成的线速度、角速度均低字节在前。例如：
   发送：602 EA 00 00 00 01 06 00 00 //指令 1537mm/s 线速度，0mrad/s

发送：602 EA 00 00 00 B7 02 03 00　//实际返回 695mm/s,3mrad/s

注：如果参数 Pn5.00D 为 1，则驱动器收到主控指令后立即上报小车差速信息，主控不发送指令数据驱动器不会自动上传，如果 Pn5.00D 为 0，则不管主控发不发指令，驱动器都不会上报小车差速信息。

3.8.4 Đọc viết 2 trục đồng thời

Driver 2 trục hỗ trợ dùng 1 câu lệnh đọc đồng thời 1 tham số Fn, Dn của 2 trục, câu lệnh lấy trục A làm chuẩn, format như sau:

COB_ID		Byte0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
		CMD	INDEX			DATA			
Hỏi	0x600+ Node_ID	0xE8 Đọc	Địa chỉ tham số Fn Địa chỉ tham số Dn Địa chỉ tham số Pn		SUB INDEX	0x00	0x00	0x00	0x00
		0xE9 Ghi				DATA1	DATA2	DATA3	DATA4
Đáp	0x580+ Node_ID	0xE8 Đọc FB				DATA1	DATA2	DATA3	DATA4
		0x60 Ghi FB				DATA1	DATA2	DATA3	DATA4

Vd1：Đọc vị trí 1 vòng của encoder

Send：601 E8 04 50 00 00 00 00 00　// Đọc 0x5004 và 0x5104, lấy địa chỉ trục A làm chuẩn

FB：　581 E8 04 50 00 B7 13 AB 12　// Phản hồi D0.04=5047, D1.04=4779

Vd2：Ghi tốc độ 2 trục

Send：601 E9 18 23 00 36 00 CA FF　// Ghi 0x2318 và 0x3318 lần lượt bằng 54, và -54

FB：　581 60 18 23 00 64 00 CA FF　// Ghi phản hồi

Bởi vì chức năng đọc viết đồng thời 2 trục có thể sửa được tham số Fn, do đó cũng có thể dùng nó để điều khiển theo mode tốc độ.

3.9 PLC 配置例程

DS/RS 系列低压伺服驱动器支持多家主流 PLC，例如西门子 CM、倍福、施耐德、科尔摩根、台达等 CANopen 主站。与 PLC 有关的 EDS 文件请联系和利时业务员及技术员获取。以下为部分 PLC 的配置例程，以供参考：

3.9.1 台达 PLC 添加例程

1、 硬件准备：

PLC：台达 DVP32ES2；

线缆：usb-232；台达 232 编程线；

其他：DS10240A，创芯科技 CAN 分析仪