英威腾 | Technical Guide

SV-DA200 Series AC Servo Drive

——CANopen

上海英威腾工业技术有限公司
INVT INDUSTRIAL TECHNOLOGY (SHANGHAI) CO., LTD.

July 26, 2017

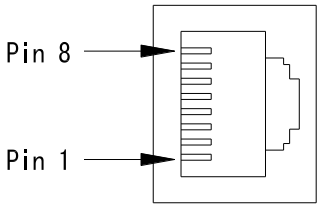
Contents

1 Hardware configuration

1.1 Terminal wiring

The CAN communication terminal is on the front panel of an SV-DA200 servo drive, named CN3. The CN3 terminal is a dual-port RJ45 socket, and the the pins of the two ports are numbered in the same way.

The following figure shows the pins and the table describes the functions of the pins.



CN3 port functions			
Pin No.	**Name**	**Function**	**Remarks**
1	5V	Power supply	485 and CAN share one interface. Each signal corresponds to two pins, facilitating the networking of multiple servo drives.
2	GND	Power ground	
3	/	CANL data cable	
4	RS485+	RS485 data cable +	
5	RS485-	RS485 data cable -	
6	/	/	
7	**CAN_L**	**CAN data cable -**	
8	**CAN_H**	**CAN data cable +**	

1.2 Baud rate setting

The following table describes multiple baud rates and their corresponding maximum transmission distance.

Communication baud rate	Communication distance
1Mbit/s	25m
500kbit/s (Default)	**100m**
250kbit/s	250m
125kbit/s	500m
50kbit/s	1000m
20kbit/s	2500m

1.3 Precautions

1. All slave stations must be wired in series connection mode instead of star connection mode.
2. A terminal resistor of 120 Ω must be connected between the master station and the last node of a slave station.
3. The sample point of the master station CAN communication must be set to 80%.
4. To avoid interference, you are advised to use shielded twisted pairs (STP) as CAN connection cables.
5. A longer connection cable requires a CAN chip with higher drive capability.

2 Software configuration

2.1 Basic settings for using CANopen

Before using CANopen on a common SV-DA200 servo drive, you need to set the following three parameters:

1. Set **P0.03** (Control mode selection) to **7** (CANopen mode) through the LED panel or ServoPlorer software setting.
2. Set **P4.02** (CAN communication baud rate) through the LED panel or ServoPlorer software setting (**0**: 1Mbps; **1**: 500kbps; **2**: 250kbps; **3**: 125kbps; **4**: 50kbps; **5**: 20kbps).
3. Set **P4.05** (CAN communication node) through the LED panel or ServoPlorer software setting (value range: 1–127).

Note:

1. These settings of the three parameters described above take effect after restart. Power on or perform soft reset on the drive after modifying these parameters.
2. The number of a slave station (servo drive) node cannot be the same as that of a master station node (CNC or PLC) or that of another slave station.
3. A synchronizing signal is generally generated by the master station, but you can configure a slave station to generate synchronizing signal. Set the unit of the synchronous communication period to 1 us. The minimum period unit supported by SV-DA200 is 1000 us, that is, 1 ms.
4. The **0x1017** parameter needs to be set if the master station needs a slave station to transmit heartbeat packets. The unit is 1 ms.
5. When a CANopen state machine exit from the OP state, the drive automatically turns off the "Enable" signal.
6. It is recommended that the PDO transmission type be set to synchronous transmission. For details, see the description in the PDO section.

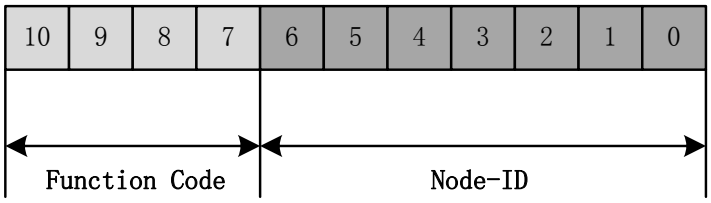
2.2 CANopen basics

CANopen is a high-level communication protocol based on the Control Area Network (CAN) communication protocol, including the communication sub-protocol and device sub-protocol. CANopen is usually used in embedded systems and is also a common fieldbus used in industrial control. The basic CANopen device and communication sub-protocols are defined in CAN in Automation (CiA) draft standard 301. Sub-protocols are extended based on CiA 301 for special devices, such as CiA 402 for motion control.

CANopen frame structure:

To reduce the configuration workload of simple networks, CANopen defines a mandatory default identifier (CANID) assignment table.

The default ID assignment table is an 11-bit CAN ID defined based on CANopen 2.0A, including a 4-bit function code and a 7-bit node ID, as shown in the following figure.



Node IDs are defined by the system integrator. The node IDs of SV-DA200 can be modified through the panel or PC software. The node IDs range from 1 to 127 (0 cannot be used).

Function Code: Data transmission function code, defines the transmission levels of various PDO, SDO, and management packets. A smaller function code indicates a higher priority.

2.3 Supported basic protocols

As a standard slave station of CANopen, an SV-DA200 servo drive supports the 301 standard protocol and some parameters of the 402 motion control protocol.

The supported basic CANopen protocols include NMT, SYNC, SDO, PDO, and EMCY.

The predefined connection set defines 4 Receive-PDOs, 4 Transmit-PDOs, 1 SDO (occupying 2 CAN-IDs), 1 emergency object, and 1 Node-Error-Control ID.

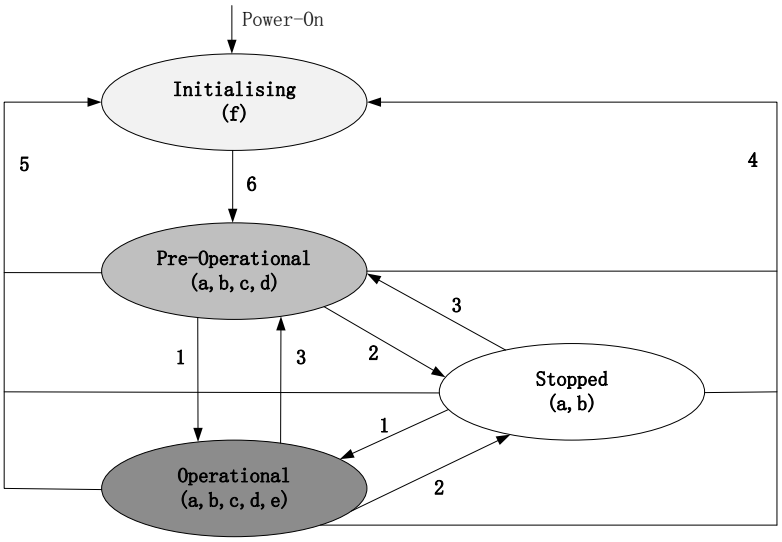
Broadcast objects in the CANopen predefined master/slave connection set			
Object	Function code (ID-bits 10-7)	COB-ID	Index of communication parameter in OD
NMT Module Control	0000	000_h	-
SYNC	0001	080_h	1006_h, 1007_h

Equivalent objects of the CANopen master/slave connection set			
Object	Function code (ID-bits 10-7)	COB-ID	Index of communication parameter in OD
EMCY	0001	081_h – $0FF_h$	1014_h, 1015_h
TPDO1	0011	181_h – $1FF_h$	1800_h
RPDO1	0100	201_h – $27F_h$	1400_h
TPDO2	0101	281_h – $2FF_h$	1801_h
RPDO2	0110	301_h – $37F_h$	1401_h
TPDO3	0111	381_h – $3FF_h$	1802_h
RPDO3	1000	401_h – $47F_h$	1402_h
TPDO4	1001	481_h – $4FF_h$	1803_h
RPDO4	1010	501_h – $57F_h$	1403_h
SDO (Tx/Server)	1011	581_h – $5FF_h$	1200_h
SDO (Rx/Client)	1100	601_h – $67F_h$	1200_h
NMT Error Control	1110	701_h – $77F_h$	1016_h, 1017_h

2.3.1 NMT

The NMT protocol is used to control the network behavior of CANopen NMT slave station devices. Both private network members and common network members switch slave state machines through the NMT protocol. All CANopen devices assess the received NMT commands. Only CANopen devices with the NMT master station functions can transmit NMT messages.

Slave station state switching diagram



After being started, the servo drive automatically switches from **Initialising** to **Pre-Operational**. To start a slave station, the master station needs to transmit an NMT command of starting a slave node. After receiving the command, the slave station switches from Pre-Operational to Operational.

PDO can be modified only in non-Operational state.

2.3.2 SYNC

The network behavior of synchronization can be implemented through the SYNC protocol. SYNC messages transmitted periodically are used to instruct receivers to start specific behavior that is related to the receiving of the SYNC messages. For synchronous PDOs, an SYNC message is a triggering event of PDO transmission and also can be an instruction of exchanging valid data received before the SYNC message is received.

SV-DA200 servo drives support only the default COB-ID (0x80) of SYNC frames. The COB-ID of SYNC frames cannot be modified.

2.3.3 SDO

Service data objects (SDO) are used to access items in the CANopen object dictionary. An SDO establishes a point-to-point communication channel between two devices. In addition, the SDO protocol can be used to transmit any amount of data in segments. Therefore, the SDO protocol is mainly used for transmitting configuration data. An SDO connection between two devices can be established by configuring the related SDO server and client channel.

The commands of the SDO protocol are transmitted between the master station and slave stations, and include 8-byte data. Information such as data length is also added, which ensures operation reliability but also occupies some data length. SDO commands are transmitted at a relatively low rate, and are used for parameter modification or monitoring, of which the rate requirement is lower.

Example of reading or writing a word

►Parameter modification

The master station transmits a packet.

Identifier	DLC	Daten							
		0	1	2	3	4	5	6	7
0x600+Node_ID	8	Transmits a command word	Object index		Object sub-index	**			

The maximum length of ** is 4 bytes, that is, 32 bits.

The slave station returns a packet.

Identifier	DLC	Daten							
		0	1	2	3	4	5	6	7
0x580+Node_ID	8	Transmits a command word	Object index		Object sub-index	**			

If the parameter is successfully modified, the command word is 0x60; if the modification fails, the command word is 0x80, and ** is a fault code.

►Parameter reading

The master station transmits a packet.

Identifier	DLC	Daten							
		0	1	2	3	4	5	6	7
0x600+Node_ID	8	Transmits a command word	Object index		Object sub-index	00			

The transmitted command word is 0x40.

The slave station returns a packet.

Identifier	DLC	Daten							
		0	1	2	3	4	5	6	7
0x580+Node_ID	8	Transmits a command word	Object index		Object sub-index	**			

The maximum length of ** is 4 bytes, that is, 32 bits.

When the data length is 1 byte, the command word is 0x4F.

When the data length is 2 bytes, the command word is 0x4B.

When the data length is 4 bytes, the command word is 0x43.

2.3.4 PDO

A process data object (PDO) includes a CAN frame with a data length of 8 bytes, which are all used to transmit data. The content of the data is predefined in the object dictionary, and generally PDOs are not responded. Therefore, the communication efficiency and rate are higher. PDOs are divided into Receive-PDOs (RPDOs) and Transmit-PDOs (TPDOs) received and transmitted by the master station, and are used for control and monitoring in scenarios where the requirements on time is higher.

PDOs can be transmitted and received in the following modes: asynchronous time triggering, asynchronous event triggering, cyclic synchronization, acyclic synchronization, and remote request.

Transmission type	**PDO Transmission**				
	cyclic	acyclic	synchronous	asynchronous	RTR only
0		X	X		
1-240	X		X		
241-251	reserved				
252			X		X
253				X	X
254				X	
255				X	

1. Synchronous PDOs (triggered by synchronous packets, of the 0–240 and 252 transmission types)

For synchronous PDO transmission, the master station must have the capability of transmitting

synchronous packets (transmitting the packets at a maximum frequency of 1 kHz). A servo transmits data after receiving a synchronous packet.

For TPDOs: Acyclic (0), the servo transmits data only once after receiving a synchronous packet; cyclic (1–240), the servo transmits data once after receiving n synchronous packets; (252), the servo is triggered, after receiving a remote frame request, by the next synchronous signal to transmit data.

For RPDOs: In all synchronization modes, the received RPDO packets are cached first and then written into the control program after the next synchronous signal is received.

2. Asynchronous PDOs (triggered by asynchronous events or periodically, of the 253–255 transmission types)

For TPDOs: 253 indicates transmitting data after receiving a remote frame request, 254 indicates transmitting data immediately after a parameter value changes, and 255 is not supported currently.

For RPDOs, a parameter value is immediately transmitted to the controller after receiving an RPDO.

Note:

1. You can set **Inhibit Time** (inhibition time) for TPDOs to specify the minimum time interval for transmitting TPDOs.

2. You need to take the relationship between baud rates and transmission rates into account when configuring PDOs. Otherwise, the bus load rate may be too high or other communication faults may be caused.

Some rules are set for the PDO receiving and transmission configuration supported by SV-DA200 servo drives. The maximum mapping of each PDO is 4 parameters, and except the node ID, the COB-ID of a PDO cannot be modified.

RPDO	COB-ID	TPDO	COB-ID
RPDO1	0x200 + servo node ID	TPDO1	0x180 + servo node ID
RPDO2	0x300 + servo node ID	TPDO2	0x280 + servo node ID
RPDO3	0x400 + servo node ID	TPDO3	0x380 + servo node ID
RPDO4	0x500 + servo node ID	TPDO4	0x480 + servo node ID

Default PDO configuration

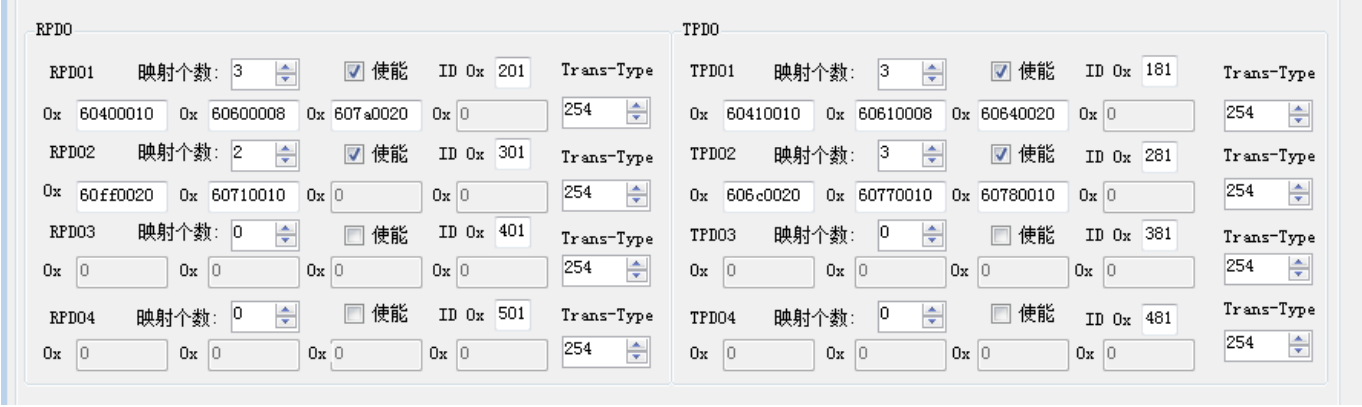
PDO	Object1	Object2	Object3	Transmission Type
RPDO1	**Controlword**	**Modes of operation**	**Target Position**	254
RPDO2	**Target_velocity**	**Target_torque**	---	254
RPDO3	---	---	---	254
RPDO4	---	---	---	254
TPDO1	**Controlword**	**Modes of operation**	**Position actual value**	254
TPDO2	**Velocity_actual_value**	**Torque_actual_value**	**Current_actual_value**	254
TPDO3	---	---	---	254
TPDO4	---	---	---	254

The default PDO mapping can implement the basic control over rates, position loops, and torque loops for common customers. Other related parameters can be modified through SDO parameters.

If the default PDO configuration cannot meet requirements, you can map the required parameters to the PDO list through the CANopen master station. The transmission type can also be modified. Each group of PDOs support the mapping of a maximum of 4 parameters or 64-byte data.

The default transmission type is 254, indicating asynchronous transmission. However, Transmission types of 1–240 (indicating synchronous transmission) are recommended for scenarios with more nodes. You can optimize the load rate of the CAN bus as required.

SV-DA200 servo drives support the modification of PDO mapping through CANopen master stations. If a master station does not provide the modification function, you can use ServoPlorer on the DA200 upper computer to modify the PDO mapping, as shown in the following figure.



2.3.5 EMCY

A device uses an emergency object to point out an internal error of the device. When receiving this signal, other network members assess the received information and start to take specific measures defined by corresponding manufacturers.

The following table describes the emergency error codes.

Byte	0	1	2	3	4	5	6	7
Content	Emergency Error Code		Error register (Object 1001_h)	Manufacturer specific Error Field				
				Error index	Error subindex	-	-	-

Self-defined area: Bytes 3 to 7 are the fault area defined by manufacturers. We define the third byte as the primary key of faults and the fourth byte as the subkey.

You can find fault codes in Chapter 5 according to the corresponding primary keys and subkeys.

An emergency error code indicates a fault type defined in the CANopen standard protocol. For more information, see the related description in this guide.

2.3.6 Node Guarding

NodeGuarding packets are used by a master station to send query requests, and the corresponding slave stations return their current states.

2.3.7 Heartbeat

A slave station transmits Heartbeat packets to automatically report its state to a master station periodically, which indicates that the communication is normal. If Heartbeat packets are required, you need to set the heartbeat time in the object dictionary to a required time.

2.4 Unsupported protocol

SV-DA200 servo drives do not support the Time Stamp protocol.

3 Operation modes

3.1 Profile Position Mode

3.1.1 Basic description

A servo drive (slave station) receives a position command transmitted by an upper computer. The position obtained by converting the position command based on the electronic gear ration is used as the target position in the internal position control.

Encoder-defined unit of a position command = User-defined unit of the position command × OD-6093_h-Sub1 / OD-6093_h-Sub2

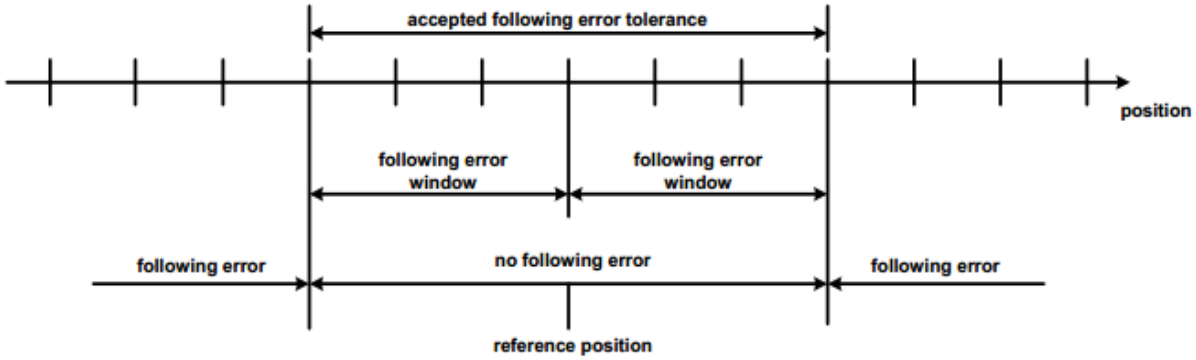
3.1.2 Operation procedure

1. Set **6060_h: Mode of operations** to **1** (**Profile position mode**).
2. Set **6081_h: Profile velocity** to the planned speed (unit: rpm). The corresponding parameter on the drive is **P5.21**.
3. Set **6083_h: Profile acceleration** to **Planned speed** (unit: ms, ranging from 0 to the rated rotating speed). **Note:** in this mode, **6083_h** and **6084_h** correspond to the same parameter **P5.37** on the drive.
4. Set **Sub-1** and **Sub-2** of **6093_h: Position factor** to adjust the electronic gear ratio (**Sub-1** indicates the numerator, and **Sub-2** indicates the denominator, corresponding to the parameters **P0.25** and **P0.26** on the drive).

   Note: Set the parameter **P0.22** to **0** and power on the drive again before setting these two parameters. The parameter **OD-6093h-Sub-2 (P0.26)** takes effect when the servo is disabled, and **OD-6093h-Sub-1 (P0.25)** takes effect immediately.
5. Set **$607A_h$: Target position** to the target position (unit: user-defined unit). The corresponding parameter on the drive is **P6.01**.
6. Set **6040_h: Control word** to enable the servo drive and trigger the target position to take effect. **0x0F** indicates **Enable**. For details about other position parameters, see the description of **6040_h** in section 4.5.
7. Query **6064_h: Position actual value** to obtain the feedback of the actual position of the motor.
8. Query **6041_h: Status word** to obtain the feedback of the state of the servo drive (following error, set-point acknowledge, target reached, and internal limit active).

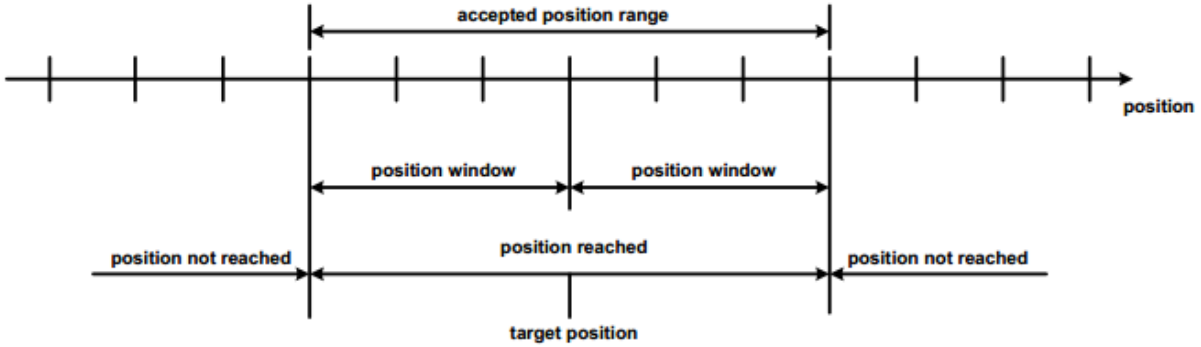
3.1.3 Other objects

1. Query **6062_h: Position actual value** to obtain the feedback of the actual position of the motor (unit: user-defined unit).
2. Query **6063_h: Position actual value*** to obtain the feedback of the actual position increment of the motor (unit: user-defined unit).
3. Set **6065_h: Following error window** to adjust the out-of-tolerance position range (unit: user-defined unit).
4. Query **$60F4_h$: Following error actual value** to obtain the actual position deviation of the motor (unit: user-defined unit).



Reference position

5. Set **6067_h: Following error window** to adjust the range of position reached (unit: user-defined unit).



Position reached

3.1.4 List of objects related to this operation mode

Index	Name	Type	Attr.
6040_h	Control word	UNSIGNED16	RW
6041_h	Status word	UNSIGNED16	RO
6060_h	Modes of operation	INTEGER8	RW
6061_h	Modes of operation display	INTEGER8	RO
6062_h	Position demand value	INTEGER32	RO
6063_h	Position actual value*	INTEGER32	RO
6064_h	Position actual value	INTEGER32	RO
6065_h	Following error window	UNSIGNED32	RW
6067_h	Position window	UNSIGNED32	RW
$607A_h$	Target position	INTEGER32	RW
6081_h	Profile velocity	UNSIGNED32	RW
6083_h	Profile acceleration	UNSIGNED32	RW
6093_h	Position factor	UNSIGNED32	RW
$60F4_h$	Following error actual value	INTEGER32	RO
$60FC_h$	Position demand value*	INTEGER32	RO

Note: For details about the objects, see the CiA DS402 standard.

3.1.5 Control word (6040) of Profile Position Mode

15 9	8	7	6	5	4	3 0
(see 10.3.1)	Halt	(see 10.3.1)	abs / rel	Change set immediately	New set-point	(see 10.3.1)
MSB						LSB

Name	Value	Description
New set-point	0	Does not assume *target position*
	1	Assume *target position*
Change set immediately	0	Finish the actual positioning and then start the next positioning
	1	Interrupt the actual positioning and start the next positioning
abs / rel	0	*Target position* is an absolute value
	1	*Target position* is a relative value
Halt	0	Execute positioning
	1	Stop axle with *profile deceleration* (if not supported with *profile acceleration*)

3.1.6 Status word (6041) of Profile Position Mode

15 14	13	12	11	10	9 0
(see 10.3.2)	Following error	Set-point acknowledge	(see 10.3.2)	Target reached	(see 10.3.2)
MSB					LSB

Name	Value	Description
Target reached	0	Halt = 0: *Target position* not reached Halt = 1: Axle decelerates
	1	Halt = 0: *Target position* reached Halt = 1: Velocity of axle is 0
Set-point acknowledge	0	Trajectory generator has not assumed the positioning values (yet)
	1	Trajectory generator has assumed the positioning values
Following error	0	No following error
	1	Following error

3.1.7 Application example

1. Set **6060_h** to **1** to select **Profile Position Mode**.
2. Set **6040_h** to enable the drive and trigger the position command to take effect.

a. Single set-point mode

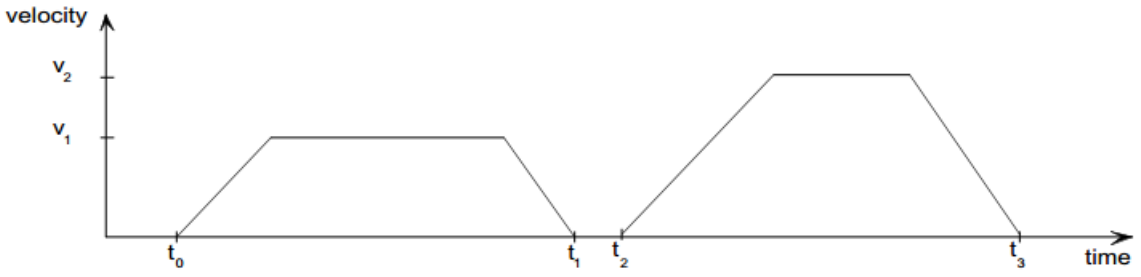


Diagram of the single set-point mode

If the target position transmitted is in incremental mode, you must perform the following steps:

(1) Set **6040_h** to **0x4F** (of which bit 6 is used to set the incremental mode, and bits 3 to 0 are used to

enable the drive).

(2) Set **607A$_h$** to the target position command.

(3) Set **6040$_h$** to **0x5F** to trigger the position command to take effect (the target position command is triggered to take effect by the edge generated when the value of bit 4 changes from 0 to 1).

(4) The drive returns **6041$_h$. bit 12** after receiving **6040$_h$. bit 4** (where bit 4= 1), and the master station deletes the value of **6040$_h$. bit 4** after receiving **6041$_h$** to get ready for transmitting the next target position command.

If the target position transmitted is in absolute mode, you must perform the following steps:

(1) Set **6040$_h$** to **0x0F**.

(2) Set **607A$_h$** to the target position command.

(3) Set **6040$_h$** to **0x1F** to trigger the position command to take effect.

(4) The drive returns **6041$_h$. bit 12** after receiving **6040$_h$. bit 4** (where bit 4= 1), and the master station deletes the value of **6040$_h$. bit 4** after receiving **6041$_h$** to get ready for transmitting the next target position command.

b. Multi-setpoint Change set immediately mode

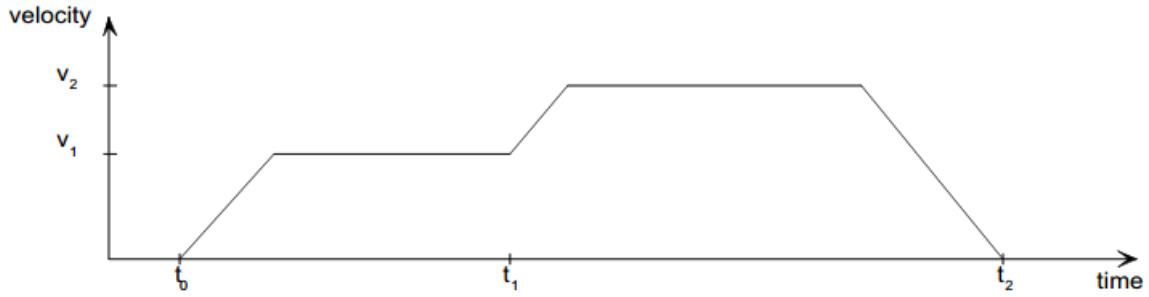


Diagram of the multi-setpoint **Change set immediately** mode

If the target position transmitted is in incremental mode, you must perform the following steps:

(1) Set **6040$_h$** to **0x6F** (of which bit 6 is used to set the incremental mode, bit 5 is used to set the immediately taking effect mode, and bits 3 to 0 are used to enable the drive).

(2) Set **607A$_h$** to the target position command.

(3) Set **6040$_h$** to **0x7F** to trigger the position command to take effect (the target position command is triggered to take effect by the edge generated when the value of bit 4 changes from 0 to 1).

(4) The drive returns **6041$_h$. bit 12** after receiving **6040$_h$. bit 4** (where bit 4= 1), and the master station deletes the value of **6040$_h$. bit 4** after receiving **6041$_h$** to get ready for transmitting the next target position command.

If the target position transmitted is in absolute mode, you must perform the following steps:

(1) Set **6040$_h$** to **0x2F** (of which bit 5 is used to set the immediately taking effect mode, and bits 3 to 0 are used to enable the drive).

(2) Set **607A$_h$** to the target position command.

(3) Set **6040$_h$** to **0x3F** to trigger the position command to take effect.

(4) The drive returns **6041$_h$. bit 12** after receiving **6040$_h$. bit 4** (where bit 4= 1), and the master station deletes the value of **6040$_h$. bit 4** after receiving **6041$_h$** to get ready for transmitting the next

target position command.

3. If multiple target positions are to be transmitted, repeat step 2.

Note: SV-DA200 servo drives support the internal caching of 8 steps of target positions.

3.2 Interpolation Position Mode

3.2.1 Basic description

1. The master station transmits SYNC broadcast frames (0x80) periodically.
2. The master station uses PDOs to transmit the next position reference **Xi** and **Control word**.
3. After receiving the control data PDO, the slave station transmits the position reference Xi to the position control application program when the next SYNC is received.
4. Caching of input data is not supported, and only the linear interpolation mode is supported (The parameter **$60C0_h$** can only be set to **0**).
5. If no SYNC broadcast frames are received in a period of more than twice of the communication period, the slave station automatically stops and generates an alarm.

3.2.2 Operation procedure

1. Set **6060_h: Mode of operations** to **7** (**Interpolation position mode**).
2. Set **1006_h: Communication cycle period** to the time interval for transmitting SYNC frames (unit: us. It is recommended that the unit be set to ms, that is, 1000 us).
3. Set **Sub-1** and **Sub-2** of **6093_h: Position factor** to adjust the electronic gear ratio (**Sub-1** indicates the numerator, and **Sub-2** indicates the denominator, corresponding to the parameters **P0.25** and **P0.26** on the drive).

   Note: Set the parameter **P0.22** to **0** and power on the drive again before setting these two parameters. The parameter **OD-6093h-Sub-2 (P0.26)** takes effect when the servo is disabled, and **OD-6093h-Sub-1 (P0.25)** takes effect immediately.
4. Set **6040_h: Control word** to enable the servo drive. **0x0F** indicates **Enable**. For details about other position parameters, see the description of **6040_h** in section 4.5.
5. Set **1600_h Sub-3** (PDO Communication & Mapping parameters) to **$60C1_h$ Sub-1** (interpolated position data **Xi**) by using SDOs as the target position (unit: user-defined unit).
6. Query **6064_h: Position actual value** to obtain the feedback of the actual position of the motor.
7. Query **6041_h: Status word** to obtain the feedback of the state of the servo drive (following error, target reached, ip mode active, and internal limit active).
8. Receive NMT frames transmitted by the master station to start or stop the slave station.

3.2.3 List of objects related to this operation mode

Index	Name	Type	Attr.
6040_h	Control word	UNSIGNED16	RW
6041_h	Status word	UNSIGNED16	RO
6060_h	Modes of operation	INTEGER8	RW
6061_h	Modes of operation display	INTEGER8	RO
6064_h	Position actual value	INTEGER32	RO
6065_h	Following error window	UNSIGNED32	RW
6067_h	Position window	UNSIGNED32	RW
6093_h	Position factor	UNSIGNED32	RW

$60C0_h$	Interpolation sub mode select	INTEGER16	RO
$60C1_h$	Interpolation data record	ARRAY	RW
$60F4_h$	Following error actual value	INTEGER32	RO

Note: For details about the objects, see the CiA DS402 standard.

3.2.4 Application example

1. Set **6060_h** to **7** to select **Interpolation Position Mode**.
2. Set **6040_h** to enable the drive. Transmit **0x1F** (of which bit 4 is **Enable ip mode**).
3. Set **$60C2_h$** (position interpolation period) based on the SYNC period of PDOs. The corresponding parameter on the drive is **P0.34**.

3.3 Homing Mode

3.3.1 Basic description

The drive automatically locates the position of origin in Homing mode. You can set the rotating speed in Homing mode.

Note: In this mode, the signals of the limit switch and origin switch must be transmitted to the switching value input terminal CN1 of the drive. If the signals of the limit switch is transmitted to the upper computer or PLC, the homing process led by the upper computer must be performed.

3.3.2 Operation procedure

1. Set **6060_h: Mode of operations** to **6** (**Homing mode**).
2. Set **6098_h: Homing method**. The value ranges from **1** to **35**. For details, see the DS402 standard.
3. Set **$607C_h$: Homing offset**. Set the origin offset. The corresponding parameter on the drive is **P5.14**.
4. Set **6099_h Sub-1: Homing speeds** to modify the speed for searching for the limit switch in the homing process (unit: rpm. The corresponding parameter on the drive is **P5.12**.
5. Set **6099_h Sub-2: Homing speeds** to modify the speed for searching for the zero position in the homing process (unit: rpm). The corresponding parameter on the drive is **P5.13**.
6. Set **6040_h: Control word** to enable the servo drive. The homing process is started when the value of **Homing operation start** (bit 4) is changed from **0** to **1**, and is stopped when the value is changed from **1** to **0**.
7. The motor searches for the limit switch and Home switch to complete the homing action.
8. Query **6041_h: Status word** to obtain the feedback of the state of the servo drive (Homing error, Homing attained, and Target reached).

3.3.3 List of objects related to this operation mode

Index	Name	Type	Attr.
6040_h	Control word	UNSIGNED16	RW
6041_h	Status word	UNSIGNED16	RO
6060_h	Modes of operation	INTEGER8	RW
6061_h	Modes of operation display	INTEGER8	RO
$607C_h$	Homing offset	INTEGER32	RW
6098_h	Homing method	UNSIGNED32	RW
6099_h	Homing speeds	ARRAY	RW

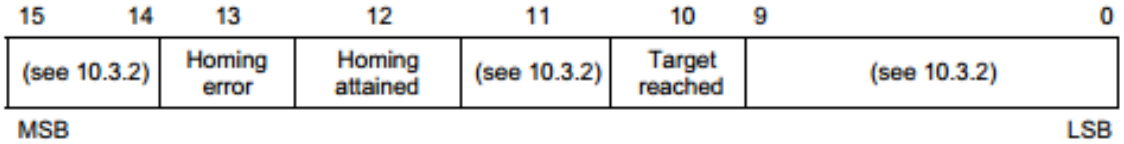
Note: For details about the objects, see the CiA DS402 standard.

3.3.4 Application example

When using the **Homing** mode, you need to perform the following steps:

1. Set **6060_h** to **6** to select **Homing Mode**.
2. Set **6098_h** to select the homing mode to be used.
3. Set **6040_h** to enable the drive and trigger the homing action: transmit **0x0F** first, and then transmit **0x1F** to trigger the homing action.
4. In the homing process, if **0x0F** is transmitted, the homing action is stopped; and if **0x0** is transmitted, the drive is disabled.
5. Determine whether the homing process is complete according to bit 12 of **6041_h**, and determine whether a fault occurs in the homing process according to bit 13.

3.3.5 Statusword of homing mode



Name	Value	Description
Target reached	0	Halt = 0: Home position not reached Halt = 1: Axle decelerates
	1	Halt = 0: Home position reached Halt = 1: Axle has velocity 0
Homing attained	0	Homing mode not yet completed
	1	Homing mode carried out successfully
Homing error	0	No homing error
	1	Homing error occurred; Homing mode carried out not successfully; The error cause is found by reading the error code

3.4 Velocity Mode

3.4.1 Basic description

In Velocity mode, the drive receives the rotating command transmitted by the master station and plans internal speeds according to the settings of the acceleration planning parameters and the RFG control parameters in **6040**.

3.4.2 Operation procedure

1. Set **6060_h: Mode of operations** to **2** (**velocity mode**).
2. Set **6046_h Sub-2: vl velocity max amount** to modify the maximum rotating speed limit (unit: rpm).
3. Set **6048_h Sub-1: vl velocity acceleration-delta speed** to modify the acceleration time (unit: rpm).
4. Set **6048_h Sub-2: vl velocity acceleration-delta time** to modify the acceleration time (unit: ms).
5. Set **6049_h Sub-1: vl velocity deceleration-delta speed** to modify the deceleration time (unit: rpm).
6. Set **6049_h Sub-2: vl velocity deceleration-delta time** to modify the deceleration time (unit: ms).
7. Set **6040_h: Control word** to enable the servo drive, start the motor, and select the RFG working mode.
8. Set **6042_h: vl target velocity** to modify the target speed (unit: rpm).
9. Query **6041_h: Status word** to obtain the feedback of the state of the servo drive (Target reached).

3.4.3 Other objects

1. Query **6043_h: vl velocity demand** to obtain the internal rotating speed command (unit: rpm).
2. Query **6044_h: vl control effort** to obtain the feedback of the actual speed (unit: rpm).
3. Set **6047_h Sub-2: vl velocity max pos** to modify the maximum forward-rotating speed limit (unit: rpm).
4. Set **6047_h Sub-4: vl velocity max neg** to modify the maximum reverse-rotating speed limit (unit: rpm).

3.4.4 List of objects related to this operation mode

Index	Name	Type	Attr.
6040_h	Control word	UNSIGNED16	RW
6041_h	Status word	UNSIGNED16	RO
6060_h	Modes of operation	INTEGER8	RW
6061_h	Modes of operation display	INTEGER8	RO
6042_h	vl target velocity	INTEGER16	RW
6043_h	vl velocity demand	INTEGER16	RO
6044_h	vl control effort	INTEGER16	RO
6046_h	vl velocity min max amount	ARRAY	RW
6047_h	vl velocity min max	ARRAY	RW
6048_h	vl velocity acceleration	RECORD	RW
6049_h	vl velocity deceleration	RECORD	RW

Note: For details about the objects, see the CiA DS402 standard.

3.5 Profile Velocity Mode

3.5.1 Basic description

In Profile velocity mode, the drive receives a rotating speed command transmitted by the master station and plans internal speeds according the settings of the acceleration planning parameters.

3.5.2 Operation procedure

1. Set **6060_h: Mode of operations** to **3** (**Profile velocity mode**).
2. Set **6083_h: Profile acceleration** to modify the acceleration curve (unit: ms). The value ranges from 0 to the rated rotating speed. The corresponding parameter on the drive is **P0.54**.
3. Set **6084_h: Profile deceleration** to modify the deceleration curve (unit: ms). The value ranges from 0 to the rated rotating speed. The corresponding parameter on the drive is **P0.55**.
4. Set **6040_h: Control word** to enable the servo drive and start the motor.
5. Set **$60FF_h$: Target velocity** to set the target rotating speed (unit: rpm). The corresponding parameter on the drive is **P4.13**.
6. Query **6041_h: Status word** to obtain the feedback of the state of the servo drive (Speed zero, Max slippage error, Target reached, and Internal limit active).

3.5.3 Other objects

1. Query **6069_h: Velocity sensor actual value** to obtain the feedback of the actual speed (unit: pulse/s).
2. Query **$606B_h$: Velocity demand value** to obtain the internal actual speed command (unit: rpm).
3. Query **$606C_h$: Velocity actual value** to obtain the feedback of the actual speed (unit: rpm).
4. Set **$606D_h$: Velocity window** to modify the speed range (unit: rpm).
5. Set **$606F_h$: Velocity threshold** to modify the zero-speed range (unit: rpm).
6. Set **$60F8_h$: Max slippage** to modify the speed out-of-tolerance setting (unit: rpm).

3.5.4 List of objects related to this operation mode

Index	Name	Type	Attr.
6040_h	Control word	UNSIGNED16	RW
6041_h	Status word	UNSIGNED16	RO
6060_h	Modes of operation	INTEGER8	RW
6061_h	Modes of operation display	INTEGER8	RO
6069_h	Velocity sensor actual value	INTEGER32	RO
$606B_h$	Velocity demand value	INTEGER32	RO
$606C_h$	Velocity actual value	INTEGER32	RO
$606D_h$	Velocity window	UNSIGNED16	RW
$606F_h$	Velocity threshold	UNSIGNED16	RW
6083_h	Profile acceleration	UNSIGNED32	RW
6084_h	Profile deceleration	UNSIGNED32	RW
$60F8_h$	Max slippage	INTEGER32	RW
$60FF_h$	Target velocity	INTEGER32	RW

Note: For details about the objects, see the CiA DS402 standard.

3.5.5 Application example

When using the **Profile Speed** mode, you need to perform the following steps:

1. Set **6060_h** to **3** to select **Profile Speed Mode**.
2. Set **6040_h** to enable the drive. Transmit **0x0F** to enable the drive, and transmit **0x0** to disable the drive.
3. Set **$60FF_h$** to modify the target speed command.
4. Set **6083_h** and **6084_h** to modify the acceleration time and deceleration time.

3.6 Profile Torque Mode

3.6.1 Basic description

In Profile torque mode, the drive receives a torque command transmitted by the master station and plans internal torques according to the settings of torque planning parameters.

3.6.2 Operation procedure

1. Set **6060_h: Mode of operations** to **4** (**Profile torque mode**).
2. Set **6087_h: Torque slope** to the torque planning time (unit: ms). It indicates the time it takes to step up the torque from 0 to 100% of the rated torque. The corresponding parameter on the drive is **P0.68**.
3. Set **6040_h: Control word** to enable the servo drive and start the motor.
4. Set 6071_h: Target torque to set the target torque (unit: 0.1% of the rated torque). The corresponding parameter on the drive is **P4.14**.
5. Query 6041_h: Status word to obtain the feedback of the state of the servo drive (Target reached).

3.6.3 Other objects

1. Set **6072_h: Max torque** to modify the maximum torque limit (unit: 0.1% of the rated torque).
2. Query **6074_h: Torque demand value** to obtain the internal actual torque command (unit: 0.1% of the rated torque).
3. Query **6076_h: Motor rated torque** to obtain the rated torque of the motor (unit: mNm).
4. Query **6077_h: Torque actual value** to obtain the feedback of the actual torque (unit: 0.1% of the rated torque).
5. Query **6078_h: Current actual value** to obtain the actual output current (unit: mA).

3.6.4 List of objects related to this operation mode

Index	Name	Type	Attr.
6040_h	Control word	UNSIGNED16	RW
6041_h	Status word	UNSIGNED16	RO
6060_h	Modes of operation	INTEGER8	RW
6061_h	Modes of operation display	INTEGER8	RO
6071_h	Target torque	INTEGER16	RO
6072_h	Max torque	UNSIGNED16	RW
6073_h	Max current	UNSIGNED16	RO
6074_h	Torque demand value	INTEGER16	RO
6075_h	Motor rated current	UNSIGNED32	RO
6076_h	Motor rated torque	UNSIGNED32	RO
6077_h	Torque actual value	INTEGER16	RO
6078_h	Current actual value	INTEGER16	RO
6079_h	DC link circuit voltage	UNSIGNED32	RO
6087_h	Torque slope	UNSIGNED32	RW

Note: For details about the objects, see the CiA DS402 standard.

3.6.5 Application example

When using the **Profile Torque** mode, you need to perform the following steps:

1. Set **6060_h** to **4** to select **Profile Torque Mode**.
2. Set **6040_h** to enable the drive. Transmit **0x0F** to enable the drive, and transmit **0x0** to disable the drive.
3. Set **6071_h** to modify the target torque command.
4. Set **6087_h** to modify the torque slope.

4 Object dictionary

4.1 Object specification description

4.1.1 Object type

Object name	Description
VAR	Value of a variable, such as UNSIGNED8, Boolean, float, and INTEGER16.
ARRAY	Array of multiple values, consisting of multiple basic variables of the same type. If **Sub-index 0** is of the **UNSIGNED8** type, it indicates the number of values in the array but is not part of the ARRAY data.
RECORD	Structure formed by multiple basic variables of the same type of different types. If **Sub-index 0** is of the **UNSIGNED8** type, it indicates the number of values in the structure but is not part of the RECORD data.

4.1.2 Data type

See CANopen Standard 301.

4.2 Overview of Object Group 1000_h

Index	Object Type	Name	Data Type	Access	Mappable
CANopen DS301					
1000_h	VAR	Device type	UNSIGNED32	RO	N
1001_h	VAR	Error register	UNSIGNED8	RO	Y
1005_h	VAR	COB-ID SYNC	UNSIGNED32	RW	N
1006_h	VAR	Communication cycle period	UNSIGNED32	RW	N
1017_h	VAR	Producer Heartbeat Time	UNSIGNED32	RW	N
1018_h	RECORD	Identity Object	UNSIGNED32	RO	N
1400_h~03_h	RECORD	Receive PDO parameter	UNSIGNED16/32	RW	N
1600_h~03_h	RECORD	Receive PDO mapping	UNSIGNED32	RW	N
1800_h~03_h	RECORD	Transmit PDO parameter	UNSIGNED16/32	RW	N
$1A00_h$~03_h	RECORD	Transmit PDO mapping	UNSIGNED32	RW	N

4.3 Overview of Object Group 6000_h

Index	Object Type	Name	Data Type	Access	Mappable
CANopen DS402					
6040_h	VAR	Control word	UNSIGNED16	RW	Y
6041_h	VAR	Status word	UNSIGNED16	RO	Y
6042_h	VAR	vl target velocity	INTEGER16	RW	Y
6043_h	VAR	vl velocity demand	INTEGER16	RO	Y
6044_h	VAR	vl control effort	INTEGER16	RO	Y
6046_h	ARRAY	vl velocity min max amount	UNSIGNED32	RW	Y
6047_h	ARRAY	vl velocity min max	UNSIGNED32	RW	Y
6048_h	RECORD	vl velocity acceleration	UNSIGNED32	RW	Y
6049_h	RECORD	vl velocity deceleration	UNSIGNED32	RW	Y
6060_h	VAR	Mode of operation	INTEGER8	RW	Y
6061_h	VAR	Mode of operation display	INTEGER8	RO	Y
6062_h	VAR	Position demand value	INTEGER32	RO	Y
6063_h	VAR	Position actual value*	INTEGER32	RO	Y
6064_h	VAR	Position actual value	INTEGER32	RO	Y

Index	Object Type	Name	Data Type	Access	Mappable
6065_h	VAR	Following error window	UNSIGNED32	RW	Y
6066_h	VAR	Following error time out	UNSIGNED16	RW	Y
6067_h	VAR	Position window	UNSIGNED32	RW	Y
6069_h	VAR	Velocity sensor actual value	INTEGER32	RO	Y
$606B_h$	VAR	Velocity demand value	INTEGER32	RO	Y
$606C_h$	VAR	Velocity actual value	INTEGER32	RO	Y
$606D_h$	VAR	Velocity window	UNSIGNED16	RW	Y
$606F_h$	VAR	Velocity threshold	UNSIGNED16	RW	Y
6071_h	VAR	Target torque	INTEGER16	RW	Y
6072_h	VAR	Max torque	UNSIGNED16	RW	Y
6073_h	VAR	Max current	UNSIGNED16	RO	Y
6074_h	VAR	Torque demand value	INTEGER16	RO	Y
6075_h	VAR	Motor rated current	UNSIGNED32	RO	Y
6076_h	VAR	Motor rated torque	UNSIGNED32	RO	Y
6077_h	VAR	Torque actual value	INTEGER16	RO	Y
6078_h	VAR	Current actual value	INTEGER16	RO	Y
6079_h	VAR	DC link circuit voltage	UNSIGNED32	RO	Y
$607A_h$	VAR	Target position	INTEGER32	RW	Y
$607C_h$	VAR	Home offset	INTEGER32	RW	Y
$607D_h$	ARRAY	Software position limit	INTEGER32	RW	Y
6080_h	VAR	Max motor speed	UNSIGNED32	RW	Y
6081_h	VAR	Profile velocity	UNSIGNED32	RW	Y
6083_h	VAR	Profile acceleration	UNSIGNED32	RW	Y
6084_h	VAR	Profile deceleration	UNSIGNED32	RW	Y
6085_h	VAR	Quick stop deceleration	UNSIGNED32	RW	Y
6086_h	VAR	Motion profile type	INTEGER16	RO	Y
6087_h	VAR	Torque slope	UNSIGNED32	RW	Y
6088_h	VAR	Torque profile type	INTEGER16	RO	Y
6093_h	ARRAY	Position factor	UNSIGNED32	RW	Y
6098_h	VAR	Homing method	INTEGER8	RW	Y
6099_h	ARRAY	Homing speeds	UNSIGNED32	RW	Y
$60C0_h$	VAR	Interpolation sub mode select	INTEGER16	RO	Y
$60C1_h$	ARRAY	Interpolation data record	INTEGER32	RW	Y
$60C2_h$	RECORD	Interlopation time period	INTEGER8	RW	Y
$60F4_h$	VAR	Following error actual value	INTEGER32	RO	Y
$60F8_h$	VAR	Max slippage	INTEGER32	RW	Y
$60FA_h$	VAR	Control effort	INTEGER32	RO	Y
$60FC_h$	VAR	Position demand value*	INTEGER32	RO	Y
$60FD_h$	VAR	Digital inputs	UNSIGNED32	RO	Y
$60FE_h$	ARRAY	Digital outputs	UNSIGNED32	RO	Y
$60FF_h$	VAR	Target velocity	INTEGER32	RW	Y

4.4 Overview of Object Group 2000_h

Index	Object Type	Name	Data Type	Access	Mappable
SV-DA200 parameter					
$2xxx_h$	VAR	Px-xx	INTEGER16/32	RW	Y
$3xxx_h$	VAR	Rx-xx	INTEGER16/32	RO	Y

Note: You can access the self-defined parameter area of the drive by simulating the LED panel through SDOs to perform operations, or map the self-defined parameters to PDOs to perform control operations.

The rules of mapping between parameter numbers and CANopen communication indexes are as follows:

Pa.bc ←→ $2aBC_h$

Ra.bc ←→ $3aBC_h$

where "BC" indicates the hexadecimal form of "bc", the number of a parameter in the parameter group.

For example, the CANopen communication index of the parameter **P0.03** is **2003_h**.

The CANopen communication index of the parameter **P0.21** is **3015_h**.

4.5 Detail of Object 6040_h

The control word **6040_h** includes the following content:

1. Bits for status control.
2. Bits related to control modes.
3. Control bits defined by manufacturers.

Details about each bit of 6040_h are described as follows.

15 11	10 9	8	7	6 4	3	2	1	0
manufacturer specific	reserved	halt	Fault reset	Operation mode specific	Enable operation	Quick stop	Enable voltage	Switch on
O	O	O	M	O	M	M	M	M

MSB LSB

where **MSB**: most significant bit; **LSB**: least significant bit.

O: Optional; **M**: Mandatory.

Bits 0–3 and 7 (bits for status control):

Command	Bit of the *controlword*					Transitions
	Fault reset	Enable operation	Quick stop	Enable voltage	Switch on	
Shutdown	0	X	1	1	0	2,6,8
Switch on	0	0	1	1	1	3*
Switch on	0	1	1	1	1	3**
Disable voltage	0	X	X	0	X	7,9,10,12
Quick stop	0	X	0	1	X	7,10,11
Disable operation	0	0	1	1	1	5
Enable operation	0	1	1	1	1	4,16
Fault reset	↑	X	X	X	X	15

where, X indicates that the operation is not involved; and ↑ indicates a rising edge.

Bits 4, 5, 6 and 8 (bits related to control modes):

Bit	Operation mode					
	Velocity mode	Profile position mode	Profile velocity mode	Profile torque mode	Homing mode	Interpolation position mode
4	rfg enable	New set-point	reserved	reserved	Homing operation start	Enable ip mode
5	rfg unlock	Change set immediately	reserved	reserved	reserved	reserved
6	rfg use ref	abs / rel	reserved	reserved	reserved	reserved
8	Halt	Halt	Halt	Halt	Halt	Halt

Bits 9 and 10: Reserved.

Bits 11–15: Defined by manufacturers.

4.6 Detail of Object 6041_h

The control word **6041_h** includes the following content:

1. Bits that indicate the current state of the drive.
2. Status bits related to control modes.
3. Status bits defined by manufacturers.

Details about each bit of 6041_h are described as follows.

Bit	Description	M / O
0	Ready to switch on	M
1	Switched on	M
2	Operation enabled	M
3	Fault	M
4	Voltage enabled	M
5	Quick stop	M
6	Switch on disabled	M
7	Warning	O
8	Manufacture specific	O
9	Remote	M
10	Target reached	M
11	Internal limit active	M
12 – 13	Operation mode specific	O
14 – 15	Manufacturer specific	O

Bits 0–3, 5, and 6:

Value (binary)	State
xxxx xxxx x0xx 0000	Not ready to switch on
xxxx xxxx x1xx 0000	Switch on disabled
xxxx xxxx x01x 0001	Ready to switch on
xxxx xxxx x01x 0011	Switched on
xxxx xxxx x01x 0111	Operation enabled
xxxx xxxx x00x 0111	Quick stop active
xxxx xxxx x0xx 1111	Fault reaction active
xxxx xxxx x0xx 1000	Fault

where, X indicates that the bit is not involved.

Bits 4–11:

See the DS402 standard file.

Bits 12 and 13:

Bit	Operation mode					
	vl	pp	pv	tq	hm	ip
12	reserved	Set-point Acknowledge	Speed	reserved	Homing attained	ip mode active
13	reserved	Following error	Max slippage error	reserved	Homing error	reserved

Bits 14–15: Defined by manufacturers.

4.7 Detail of Object 6060_h

The control word **6060_h** is used to select a control mode.

Value	Description
-1 … -128	manufacturer specific modes of operation
0	reserved
1	Profile Position Mode
2	Velocity Mode
3	Profile Velocity Mode
4	Torque Profile Mode
5	reserved
6	Homing Mode
7	Interpolated Position Mode
8 … 127	reserved

4.8 Other objects

Object 1000 h: Device Type

Index	0x1000
Name	device type
Object Code	VAR
Data Type	UNSIGNED32
Access	RO
PDO Mapping	NO
Value Range	UNSIGNED32
Default Value	0x04020200

Object 1001 h: Error Register

Index	0x1001
Name	Error Register
Object Code	VAR
Data Type	UNSIGNED8
Access	RO
PDO Mapping	NO
Value Range	UNSIGNED8
Default Value	0

Object 1005 h: COB-ID SYNC message

Index	0x1005
Name	COB-ID SYNC

	message
Object Code	VAR
Data Type	UNSIGNED32
Access	RO
PDO Mapping	NO
Value Range	UNSIGNED32
Default Value	0x80

Object 1006 h: Communication Cycle Period

Index	**0x1006**
Name	Communication Cycle Period
Object Code	VAR
Data Type	UNSIGNED32
Access	RO
PDO Mapping	NO
Value Range	UNSIGNED32
Default Value	0

Object 1017 h: Producer Heartbeat Time

Index	**0x1017**
Name	Producer Heartbeat Time
Object Code	VAR
Data Type	UNSIGNED16
Access	RW
PDO Mapping	NO
Value Range	UNSIGNED16
Default Value	0

Object 1018 h: Identity Object

Index	**0x1018**
Name	Identity Object
Object Code	RECORD
Data Type	UNSIGNED16
Access	Identity
PDO Mapping	NO

Subindex	0
Description	number of entries
Data Type	UNSIGNED8
Access	RO
PDO Mapping	No
Value Range	3
Default Value	3

Sub-Index	1
Description	Vendor ID
Data Type	UNSIGNED32

Access	RO
PDO Mapping	NO
Value Range	UNSIGNED32
Default Value	4

Subindex	2
Description	Product code
Data Type	UNSIGNED32
Access	RO
PDO Mapping	NO
Value Range	UNSIGNED32
Default Value	3

Subindex	3
Description	Revision number
Data Type	UNSIGNED32
Access	RO
PDO Mapping	NO
Value Range	UNSIGNED32
Default Value	0x10

Subindex	4
Description	Serial number
Data Type	UNSIGNED32
Access	RO
PDO Mapping	NO
Value Range	UNSIGNED32
Default Value	0x10

5 Faults and diagnosis

5.1 Information format of CANopen communication faults

Emergency Object:

Byte	0	1	2	3	4	5	6	7
Content	Emergency Error Code		Error register	Panel Error Code		N/A		

5.2 CANopen communication faults and solutions

Displayed code	Fault name	Cause	Solution
Er26-0	SDO timeout	After reading/writing an SDO, the master station does not receive a response sent by the drive in a specified period.	Check whether the communication is normal.
Er26-1	SDO index cannot be found	When an SDO is used to read or write a parameter, the corresponding index does not exist in the object dictionary or is not supported by the drive.	Check the indexes queried by the master station and those supported by the drive, and modify the EDS file.
Er26-2	SDO sub-index cannot be found	When an SDO is used to read or write a parameter, the corresponding index exists in the object dictionary but the sub-index does not exist in the object dictionary or is not supported by the drive.	Check the indexes and sub-indexes queried by the master station and those supported by the drive, and modify the EDS file.
Er26-3	SDO data length error	The length information in the SDO read/write command does not match the data length in the object dictionary on the drive.	Modify the length of SDO read/write commands according to the data length in the object dictionary on the drive.
Er26-4	Data written through SDO exceeds the range	The range of the data written through SDOs exceeds the data range in the object dictionary on the drive.	Modify the size of the data written through SDOs according to the data range in object dictionary.
Er26-5	Read only, cannot be modified	Attempt to modify a read-only parameter.	Check whether you have attempted to perform a write operation on a read-only parameter.
Er26-6	PDO mapping length error	The total length of the data mapped by a PDO exceeds 64 bits.	Check the total length of the corresponding PDO mapping.
Er26-7	Data mapped by PDO cannot be found	The corresponding parameter of the data mapped by a PDO cannot be found in the object dictionary.	Check whether the PDO mapping indexes and sub-indexes exist in the object dictionary.
Er26-8	PDO cannot be modified in the operation state	Attempt to modify a PDO mapping in the operation state (OP state).	Switch the CANopen state machine to the pre-operation state (Pre-OP) before modifying the PDO mapping.
Er26-9	PDO mapping not allowed	Attempt to map a parameter that is not allowed to be mapped to a PDO.	Check whether you attempt to map a read-only PDO parameter to an RPDO.
Er26-a	SYNC signal transmitted too fast	In synchronous working mode, the number of frames received by the	1. Modify the time interval of the master station for transmitting data frames or synchronous frames.

Displayed code	Fault name	Cause	Solution
		slave station exceeds the allowed baud rate range.	2. Modify the communication baud rate.
Er26-b	Receiving fault	CAN communication is disconnected or a receiving error occurs. The count value of the counter exceeds 128.	1. Check the communication wiring. 2. Restart the servo drive.
Er26-c	Transmission fault	CAN communication is disconnected or a transmission error occurs. The count value of the counter exceeds 128.	1. Check the communication wiring. 2. Restart the servo drive.
Er26-d	Duplicate SYNC signals	After a slave station is configured to generate SYNC signals, SYNC signals are input by an external source.	Modify the configuration to ensure that there in only one SYNC signal generation source in one communication network.
Er26-e	Bus load rate too high	In asynchronous working mode, the number of frames received by the slave station exceeds the allowed baud rate range.	1. Modify the time interval of the master station for transmitting data frames. 2. Modify the mode of the slave station for transmitting TPDOs. 3. Modify the communication baud rate.
Er26-f	Parameter modification state error	Attempt to modify a parameter through an SDO in a state where modification is not allowed.	Modify the CANopen state machine to the Pre-OP or OP state before attempting to modify the parameter.
Er22-3	SYNC signal timeout	In Interpolation position mode, the time interval for transmitting SYNC frame signals exceeds twice of the communication period.	1. Check the communication wiring and improve communication reliability. 2. Check whether the time interval of the SYNC signal generation source for transmitting SYNC frames is correctly configured.

5.3 SV-DA200 servo faults and fault codes

Displayed code	Fault name	32-bit fault code (16-bit error code + 16-bit additional information)
Er01-0	IGBT fault	FF01-0101$_h$
Er02-0	Encoder fault—encoder disconnected	7300-0200$_h$
Er02-1	Encoder fault—Encoder feedback deviation too large	7300-0201$_h$
Er02-2	Encoder fault—Parity error	7300-0202$_h$
Er02-3	Encoder fault—CRC error	7300-0203$_h$
Er02-4	Encoder fault—Frame error	7300-0204$_h$
Er02-5	Encoder fault—Short frame error	7300-0205$_h$
Er02-6	Encoder fault—Encoder timeout error	7300-0206$_h$
Er02-7	Encoder fault—FPGA timeout error	7300-0207$_h$
Er02-8	Encoder fault—Encoder battery low-voltage alarm	7300-0208$_h$
Er02-9	Encoder fault—Encoder battery undervoltage fault	7300-0209$_h$
Er02-a	Encoder fault—Encoder overheating	7300-020A$_h$
Er02-b	Encoder fault—Encoder EEPROM write error	7300-020B$_h$

Displayed code	Fault name	32-bit fault code (16-bit error code + 16-bit additional information)
Er03-0	Current sensor fault—U-phase current sensor fault	7300-0300$_h$
Er03-1	Current sensor fault—V-phase current sensor fault	7300-0301$_h$
Er03-2	Current sensor fault—W-phase current sensor fault	7300-0302$_h$
Er04-0	System initialization fault	FF01-0400$_h$
Er05-1	Setting fault—Motor model does not exit	FF01-0501$_h$
Er05-2	Setting fault—Motor model and drive model do not match	FF01-0502$_h$
Er05-3	Setting fault—Software limit setting fault	FF01-0503$_h$
Er05-4	Setting fault—Homing mode setting fault	FF01-0504$_h$
Er05-5	Setting fault—Positioning control stroke overflow fault	FF01-0505$_h$
Er07-0	Regenerative discharge fault	7100-0700$_h$
Er08-0	Analog input overvoltage fault—Analog speed command	5441-0800$_h$
Er08-1	Analog input overvoltage fault—Analog torque command	5442-0801$_h$
Er08-2	Analog input overvoltage fault—Analog input 3	5443-0802$_h$
Er09-0	EEPROM fault—Read/write fault	5530-0900$_h$
Er09-1	EEPROM fault—Data verification fault	5530-0901$_h$
Er10-0	Hardware fault—FPGA fault	5544-0A00$_h$
Er10-1	Hardware fault—Communication card fault	5544-0A01$_h$
Er10-2	Hardware fault—Grounding short-circuit fault	5544-0A02$_h$
Er10-3	Hardware fault—External input fault	5544-0A03$_h$
Er10-4	Hardware fault—Emergency stop fault	4458-0A04$_h$
Er11-1	Software fault—Reentrancy of a periodical task	6100-0B01$_h$
Er11-2	Software fault—Invalid operation	6100-0B02$_h$
Er12-0	IO fault—Switching value input allocation repeated	FF01-0C00$_h$
Er12-1	IO fault—Analog input allocation repeated	FF01-0C01$_h$
Er12-2	IO fault—Pulse input frequency too high	FF01-0C01$_h$
Er13-0	DC fault—Overvoltage fault	3110-0D00$_h$
Er13-1	DC fault—Undervoltage fault	3120-0D01$_h$
Er14-0	Control power undervoltage fault	5200-0E00$_h$
Er18-0	Motor overload fault	2310-1200$_h$
Er19-0	Speed fault—Overspeed fault	7180-1300$_h$
Er20-0	Speed out-of-tolerance fault	8400-1400$_h$
Er22-0	Out-of-tolerance fault—Position out of tolerance	8500-1600$_h$
Er22-1	Out-of-tolerance fault—Hybrid control deviation too large	FF01-1601$_h$
Er22-2	Position increment overflow fault	FF01-1602$_h$
Er22-3	CANopen fault—SYNC signal timeout	FF01-1603$_h$
Er23-0	Drive overtemperature fault	4210-1700$_h$
Er24-0	Profibus-DP fault—PWK parameter ID error	8100-1800$_h$
Er24-1	Profibus-DP fault—PWK parameter out of range	8100-1801$_h$
Er24-2	Profibus-DP fault—PWK parameter read only	8100-1802$_h$
Er24-3	Profibus-DP fault—PZD configuration parameter does not exist	8100-1803$_h$

Displayed code	Fault name	32-bit fault code (16-bit error code + 16-bit additional information)
Er24-4	Profibus-DP fault—PZD configuration parameter property does not match	8100-1804$_h$
Er25-6	Application fault—Homing offside	FF01-1903$_h$
Er25-7	Application fault—Inertia identification failure	FF01-1903$_h$
Er26-0	CANopen fault—SDO timeout	FF01-1A00$_h$
Er26-1	CANopen fault—SDO index cannot be found	FF01-1A01$_h$
Er26-2	CANopen fault—SDO sub-index cannot be found	FF01-1A02$_h$
Er26-3	CANopen fault—SDO data length error	FF01-1A03$_h$
Er26-4	CANopen fault—Data written through SDO exceeds the range	FF01-1A04$_h$
Er26-5	CANopen fault—Read only, cannot be modified	FF01-1A05$_h$
Er26-6	CANopen fault—PDO mapping length error	FF01-1A06$_h$
Er26-7	CANopen fault—Data mapped by PDO cannot be found	FF01-1A07$_h$
Er26-8	CANopen fault—PDO cannot be modified in the operation state	FF01-1A08$_h$
Er26-9	CANopen fault—PDO mapping not allowed	FF01-1A09$_h$
Er26-a	CANopen fault—SYNC signal transmitted too fast	FF01-1A0A$_h$
Er26-b	CANopen fault—Receiving fault	FF01-1A0B$_h$
Er26-c	CANopen fault—Transmission fault	FF01-1A0C$_h$
Er26-d	CANopen fault—Duplicate SYNC signals	FF01-1A0D$_h$
Er26-e	CANopen fault—Bus load rate too high	FF01-1A0E$_h$
Er26-f	CANopen fault—Parameter modification state error	FF01-1A0F$_h$

5.4 SDO Abort Codes

Abort Code	Description
05040001$_h$	Client/server command specifier not valid or unknown
06010002$_h$	Attempt to write a read only object
06020000$_h$	Object does not exist in the object dictionary
06040041$_h$	Object cannot be mapped to the PDO
06040042$_h$	The number and length of the objects to be mapped would exceed PDO length
06060000$_h$	Access failed due to an hardware error(store or restore error)
06070010$_h$	Data type does not match, length of service parameter does not match
06090011$_h$	Sub-index does not exist
06090030$_h$	Value range of parameter exceeded(only for write access)
08000000$_h$	General error
080000a1$_h$	Object error when reading from EEPROM
080000a2$_h$	Object error when writing to EEPROM
080000a3$_h$	Invalid Range when accessing EEPROM
080000a4$_h$	Checksum error when accessing EEPROM
080000a5$_h$	Password error when writing encryption zone
08000020$_h$	Data cannot be transferred or stored to the application (store or restore signature error)
08000021$_h$	Data cannot be transferred or stored to the application because of the local control(store or restore while wrong state)
05040022$_h$	Object is on the fly

6 References

1. *CANopen Application Layer and Communication Profile, CiA Draft Standard 301, Version 4.02* Date: 13 February 2002;
2. *CANopen Device Profile Drives and Motion Control, CiA Draft Standard Proposal 402, Version 2.0* Date: 17 March 2005.