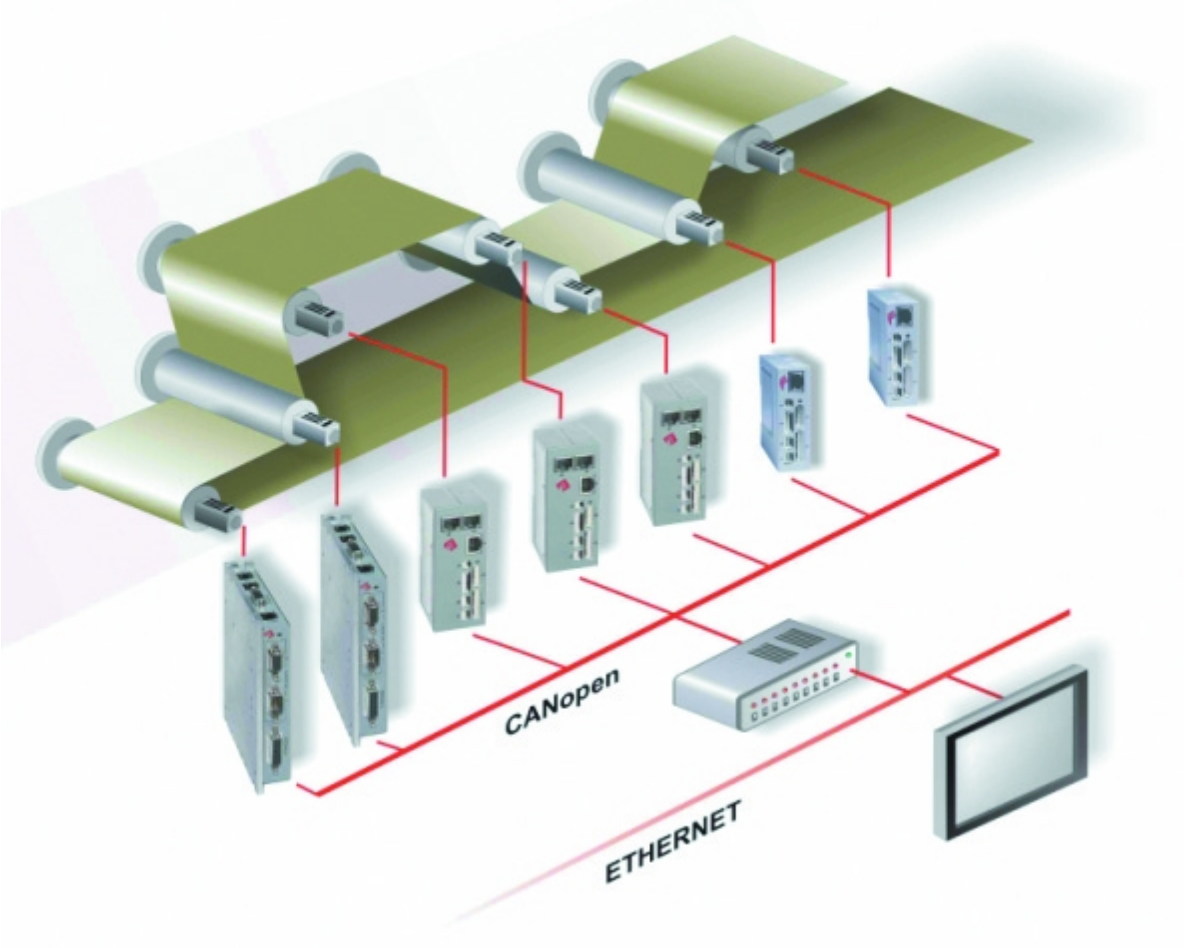
Inspiring Motion
Since 1988

CAN DS-402 Implementation Guide



March 2017 (Ver. 1.015)

www.elmomc.com

Elmo
Motion Control

Notice

This guide is delivered subject to the following conditions and restrictions:

- This guide contains proprietary information belonging to Elmo Motion Control Ltd. Such information is supplied solely for the purpose of assisting users of the CAN DS‑402 servo drive in its installation.
- The text and graphics included in this manual are for the purpose of illustration and reference only. The specifications on which they are based are subject to change without notice.
- Information in this document is subject to change without notice.

Elmo Motion Control and the Elmo Motion Control logo are registered trademarks of Elmo Motion Control Ltd.

EtherCAT Conformance Tested. EtherCAT® is a registered trademark and patented technology, licensed by Beckhoff Automation GmbH, Germany.

CANopen compliant. CANopen® is a registered trademark and patented technology, licensed by CAN in Automation (CiA) GmbH, Kontumazgarten 3, DE-90429 Nuremberg, Germany.

Document no. MAN-G-DS402 (Ver. 1.015)

Revision History

Version	Date	Changes
Ver. 1.000	May 2014	Initial Release updates: 0x6007 INT16 type 0x6069 RO type 0x606A INT16 type 0x6079 ECAT TxMap 0x60F2 description fixed 0x603F description fixed Fig 6.3 changed Updated Chapter 2;: The Object Dictionary Small change to 0x605A
Ver. 1.001	Apr 2015	Changes to Objects: 0x607E, 0x608F, 0x6090, 0x6091, 0x6092, 0x6096, 0x6097, 0x6099, 0x609A, 0x607F, 0x6086, 0x60C5, 0x60C6, 0x6072, 0x6073, 0x6075, 0x6076, 0x6098, 0x6099,0x605E, 0x6066 Object Dictionary Table change Changes to Touch Probe Objects in chapter 19
Ver. 1.002	May 2015	Change in object table Change in objects: 0x607B, 0x607D, 0x60C5
Ver. 1.003	May 2015	Definition of NONE added Change in object table Change in objects: 0x607B, 0x607D, 0x60C5, 0x60C6, 0x60B2, 0x606A, 0x6072, 0x6073, 0x6079, 0x60B8
Ver. 1.004	May 2015	Change to Fig. 6.3 Change tables in sub clauses 6.3, 6.4 Change in text of chapters 12.2, 12.16, 12.17 New figures 12-3 – 12-8 including One single set-point, Set-point handling for four set-points and Five set-points,the 5th ignored replace the present graphs.
Ver. 1.005	June 2015	Correction to Fig. 10.3

Version	Date	Changes
Ver. 1.006	Aug 2015	Correction to Object 0x60FD: Digital inputs
Ver. 1.007	Sep 2015	Correction to default values of objects: 0x6081, 0x6083, 0x6084, 0x6085, 0x60C5, 0x60C6, 0x607F, 0x6072, 0x6073, 0x6087, ox60E0, 0x60E1, 0x607D, 0x6099, 0x609A, 0x605A Change in objects: 0x605B, 0x605C Changes to section 10.3.1
Ver. 1.008	Sep 2015	Changes to table 1.1; The Object List
Ver. 1.009	Dec 2015	Change in objects: 0x60c1.0, 0x6060, 0x60B8 Objects 0x60E0, 0x60E1 removed (also in object dictionary) General changes to Object Dictionary.
Ver. 1.010	Apr 2016	Changes to heading 18.3; *Statusword* of the profiled torque mode Updates to chapters 13.2, 13.5, 13.7, 13.8, 13.13.2, 14.2 Replaced and corrected Fig 13-1
Ver. 1.011	Aug 2016	Correction to Objects 2205/2, 0x607A Changes to section 10.9.35 Change to headings of Homing sections Overall corrections to text
Ver 1.012	Oct 2016	1. Section 18.1: 0x6088 removed 2. Section 18.1 fixed typo: 0x6074 3. Change in section 13.5 4. Changes in Fig. 13-1 “Interpolated mode structure”.
Ver 1.013	Jan 2017	Correction to Figs 14.1, 15.1, 16.1
Ver. 1.014	Feb 2017	Correction to Homing Method 21, 10, 27 drawings
Ver. 1.015	Mar 2017	Correction to Fig. 19-1. Chapter 16 small change to heading title names

Chapter 1: Introduction

This document describes the objects and operational modes of the Elmo DSP-based motion controller implementation of the CiA DSP-402 protocol.

The Elmo drive includes a digital and power section. As it comes to motion, the digital section includes the profiler where a requested target (set-point) has been processed to result in a desired trajectory and the control loops where a control algorithm (PI/P) insures that the physical load will follow the desired trajectory within the specified limits.

Generally, the DS-402 protocol refers only to the profiler behavior such as the mode of operation, desired target also named set-point, required speed, acceleration, limits and handling of violations. It does not deal with control parameters such as PI/P, scheduling and feed forward. The motor can be tuned and the plant parameters set with the Elmo EAS. The protocol offers methods in which a profiled reference can be given to the final load.

The Elmo controller provides a number of different options for setting commands and parameters, such as via the proprietary binary interpreter, OS interpreter, RS-232 interpreter and user programs. When the user works with DS-402, all relevant motion commands must be given through this method only. Other command sources may prevent it from operating properly according to the protocol.

Subsequently modifying controller states, modes and reference parameters using other methods may lead to undefined states. For example, in an error state, a FAULT_RESET from the *Controlword* must be given before enabling the motor again. But sending **MO**=1 through the OS interpreter may activate the motor and leave the status word of the DS-402 with an undefined status.

Other command sources are still useful for purposes not covered by the DS-402 protocol. Examples include:

- Monitoring the states of inputs to the Gold digital servo drives.
- Using the Elmo EAS application to monitor Gold digital servo drive behavior through the USB port while the digital servo drive is under control of the CAN DS-402 protocol.
- Using the user program (or any of the interpreters) to program issues outside the range of DS-402 usage. For example, when the DS-402 digital output command is not used, the digital outputs can be operated freely by a user program.

1.1. Operating Principles

The CiA DS-402 CANopen *Device Profile for Drives and Motion Control* is used to provide drives as well as controller (PLS, CANopen masters) in a CAN network with an understandable and consistent behavior. The profile is built on top of a CAN communication profile, called CANopen, which is defined in DS-301 protocol and describes the basic communication mechanisms common to all devices in the CAN network.

Note: DS-402 was designed for CANopen communication and was well adopted by the EtherCAT communication protocols from Beckhoff. The CAN Over EtherCAT (CoE) is designed to meet the requirements of the DS-402 operating via an EtherCAT network. Unless mentioned specifically, the

described operation or object complements both communication channels. For more details about EtherCAT refer to the *EtherCAT Application Manual.*

The drive receives messages from the motion controller (host) such as Elmo G-MAS. These messages include configuration parameters, operational commands and status requests. The drive transmits the following back to the host; status messages, required feedbacks and fault indications over the network.

Typical operation via the CANopen network is that the host configures the drive via Service Data objects (SDO messages) and with the requested real time parameters. After the configuration is confirmed, the host transfers the drive status to operational state (OP state) where the preconfigured Process Data Objects (PDO) are used for the real time control. The PDOs increase the communication throughput allowing the motion controller to set the next set-point (motion target), modify the drive status and receive feedback information such as actual position, velocity and current in highly efficient way.

The motion controller controls the drive using the Control word (object 0x6040) and receives the status of the drive via Status Word (object 0x6041). It can select the motion mode of the drive via the Mode Of Operation object (0x6060). The optional modes are: Profile Position, Profile Velocity, Profile torque, Interpolated Position , Homing mode and in EtherCAT based drives: Cyclic Synchronous Position, velocity or torque modes.

The most important part of a device profile is the **object dictionary** description. The object dictionary is essentially a grouping of objects accessible via the network in an ordered pre-defined fashion. The DS-402 standard objects of single-axis drives, , are all in the index range of 0x6000 to 0x67ff and possible sub-index range of 1 to 254.

1.2. Abbreviations and Terms

The following terms are used in this document:

Abbreviation/Term	Definition
abs/rel	Absolute and relative, which are indications of how to treat the position reference command in relation to the actual location.
EAS	An Elmo Application Studio application used for controller setup, application downloading and monitoring.
Hexadecimal	Numbers marked with either *h* (such as 1000h) or *0x* (such as 0x1000) refer to a hexadecimal value. Objects and numbers may appear in either form in different CAN documents.
Load position	What the position sensor measures, expressed in position units (in contrast to position sensor increments).
Non-volatile	The object data may be saved to the flash memory of a device using the SV command, or by setting object 0x1010 (sub1).

Abbreviation/Term	Definition
Position sensor increments	Units measured by the load position sensor. The speed is derived from the position sensor.
pp	Profiled position mode
tq	Profiled torque mode
pv	Profiled velocity mode
ip	Interpolated position mode (available in CANopen only)
hm	Homing mode
csp csv cst	Cyclic Synchronous Position mode Cyclic Synchronous Velocity mode Cyclic Synchronous Torque mode
Reference	Motion parameters can be specified in terms of meters/second for speed, or encoder counts for position.
rfg	The reference generator, which generates the trajectory for velocity mode only.
NONE	**Wherever NONE is defined as the Default Value for an Object or parameter, no default value is relevant.**

The following table lists the shortened terms used in this manual:

Prefix/Suffix	Definition
UU	User defined Units
Cnt/sec	counts per second
Sub	Sub Index
TxMap	Mappable to TPDO
RxMap	Mappable to RPDO

Table 1-1: Shortened Terms

1.3. Elmo Documentation

This manual which is based on the Elmo CANopen Implementation Guide, is part of the Elmo Gold digital servo drive documentation set:

In addition to this document, the Gold documentation set includes:

- Installation Guides which provide full instructions for installing any of Elmo's drives
- The EAS User Manual, which includes explanations of all the software tools that are a part of Elmo's EAS software environment
- The Gold set of Drive Manuals, which describe the comprehensive software used with the Gold Line of digital servo drives

This is the main source of detailed explanations of all Gold commands mentioned in this manual.

- The CANopen DS-301 implementation guide, which details the basic terms and communication object of the CANopen communication with a Gold digital servo drive.

Gold drives are fully compliant with CiA's DS-305 protocol for Layer Setting Service (LSS).

Chapter 2: The Object Dictionary

The object dictionary is essentially a grouping of objects that are accessible via receive and transmit SDOs. Part of the object can be mapped to transmit and receive PDOs (TPDO and RPDO, respectively) in a predefined manner.

The following layout (Table 2-1) is used with the objects in the object dictionary:

Index (Hex)	Object
0	Not used
0001 - 001F	Static data type
0020 - 003F	Complex data type
0040 - 005F	Manufacturer-specific data type
0060 - 0FFF	Reserved
1000 - 1FFF	Communication profile area
2000 - 2FFF	Manufacturer-specific profile area
6000 - 6FFF	Standardized device profile area
A000 - FFF	Reserved

Table 2-1: Object Dictionary Layout

The following table (Table 2-2: Object Dictionary) lists the objects supported by Gold digital servo drives. Each object is addressed by a 16-bit index. Some of the objects may include 8-bit sub-indices, which are described in the object description. The object **Name** is that given by CiA or Elmo according to object type. An **Attribute** can be RO (read only), WO (write only) or RW (read and write). The objects 0x6000 – 0x6FFF are described in the remaining chapters of this manual. Objects 0x0001 – 0x2FFF are described in MAN-G-DS301 manual.

Refer to the Table 1-1: Shortened Terms for a definition of the shortened terms used in this manual.

Object(Hex) /Hi Sub(Dec)	Name	Data Type	Attribute	Mappable?	Comment
1000/0	Device type	UINT32	RO	No	CAN, ECAT. Return 0x192
1001/0	Error Register	UINT8	RO	No	CAN, ECAT.
1002/0	Manufacturer Status Register	UINT32	RO	TxMap	CAN, ECAT. Similar to **SR** command
1003/16	Pre-Defined Error Field	UINT32	RO	No	CAN, ECAT. Up to 16 last transmitted EMCY messages
1006/0	Communication Cycle	UINT32	RW	No	CAN only. Present for

Object(Hex) /Hi Sub(Dec)	Name	Data Type	Attribute	Mappable?	Comment
	Period				compatibility reasons.
1008/0	Manufacture Device Name	STRING	CONSTANT	No	CAN, ECAT. Drive given name
1009/0	Manufacture Hardware Version	STRING	CONSTANT	No	CAN, ECAT.HW identification number
100A/0	Manufacture Software Version	STRING	CONSTANT	No	CAN, ECAT. Similar to **VR** command
100B/0	CANopen Node ID	UINT8	RO	No	CAN only. Similar to **PP[13]** command
1010/1	Store Parameters	UINT32	Sub 0: RO, Sub 1: RW	No	CAN, ECAT. Similar to **SV** command
1011/1	Restore Default Parameters	UINT32	Sub 0: RO, Sub 1: RW	No	CAN, ECAT. Similar to **LD** command
1016/2	Consumer heartbeat time	UINT32	Sub 0 –RO, Sub 1,2: RW	No	CAN only
1017/0	Producer heartbeat time	UINT16	RW	No	CAN only
1018/4	Identity Object	UINT32	RO	No	CAN, ECAT. Used in LSS for drive identification
1023/3	**OS** command	RECORD	RW	No	CAN only. See OS Interpreter chapter in DS301 Manual
1024/0	**OS** Command mode	UINT8	WO	No	CAN only.
1029/1	Error Behavior object	UINT8	Sub 0: RO, Sub 1: RW	No	CAN only. Loss of heartbeat communication response.
10E0/2	Device ID Reload	INT16	Sub 0: RO, Sub 1,2: RW	No	ECAT only
10F1/2	SYNC error setting	UINt32	Sub 0,1: RO; Sub 2: RW	No	ECAT only
1400/2 - 1403/2	RPDO communication parameter	Data type 0x20	CAN: RW ECAT: RO	No	CAN only. Receive PDO mapping communication parameters of PDO1 to PDO4.
1600	RPDO mapping parameter	UINT32	CAN: RW ECAT: RO	No	CAN, ECAT. Receive PDO mapping parameters; CAN: Up to 8 entries. ECAT: Up to 3 entries.
1601, 1602	RPDO mapping parameters	UINT32	CAN: RW ECAT: RO	No	CAN, ECAT. Receive PDO mapping parameters; CAN: Up to 8 entries. ECAT: Up to 2 entries.

Object(Hex) /Hi Sub(Dec)	Name	Data Type	Attribute	Mappable?	Comment
1603	RPDO mapping parameters	UINT32	CAN: RW ECAT: RO	No	CAN, ECAT. Receive PDO mapping parameters; CAN: Up to 8 entries. ECAT: Up to 4 entries.
1604/4	RPDO mapping parameters	UINT32	RO	No	ECAT only. Receive PDO mapping parameters;
1605/7	RPDO mapping parameters	UINT32	RO	No	ECAT only. Receive PDO mapping parameters;
1606/6	RPDO mapping parameters	UINT32	RO	No	ECAT only. Receive PDO mapping parameters;
1607/8, 1608/8	RPDO mapping parameters	UINT32	RW	No	ECAT only. Receive PDO mapping parameters;
160A/1	RPDO mapping parameters	UINT32	RO	No	ECAT only. Receive PDO mapping parameters;
160B/2	RPDO mapping parameters	UINT32	RO	No	ECAT only. Receive PDO mapping parameters;
160C/1-160F/1; 1611/1-1619/1; 161C/1, 161D/1	RPDO mapping parameters	UINT32	RO	No	ECAT only. Receive PDO mapping parameters;
161A/1	RPDO_161A Mapping	UINT32	RO	No	CAN,ECAT. Receive PDO mapping parameters;
161E/2	RPDO mapping parameters	UINT32	RO	No	ECAT only. Receive PDO mapping parameters;
161F/1 - 1621/1	RPDO mapping parameters	UINT32	RO	No	ECAT only. Receive PDO mapping parameters;
1800/5 – 1803/5	TPDO communication parameter	UINT32	CAN: RW ECAT: RO	No	CAN only. Transmit PDO mapping communication parameters of PDO1 to PDO4. CAN: Sub-indexes 1-3, 5 only exist. ECAT: Up to 4 entries.
1A00	TPDO mapping parameter	UINT32	CAN: RW ECAT: RO	No	CAN, ECAT. Transmit PDO mapping

Object(Hex) /Hi Sub(Dec)	Name	Data Type	Attribute	Mappable?	Comment
					parameters; CAN: Up to 8 entries. ECAT: Up to 3 entries
1A01	TPDO mapping parameter	UINT32	CAN: RW ECAT: RO	No	CAN, ECAT. Transmit PDO mapping parameters; CAN: Up to 8 entries. ECAT: Up to 4 entries
1A02	TPDO mapping parameter	UINT32	CAN: RW ECAT: RO	No	CAN, ECAT. Transmit PDO mapping parameters; CAN: Up to 8 entries. ECAT: Up to 5 entries
1A03	TPDO mapping parameter	UINT32	CAN: RW ECAT: RO	No	CAN, ECAT. Transmit PDO mapping parameters; CAN: Up to 8 entries. ECAT: Up to 4 entries
1A04/6	TPDO mapping parameter	UINT32	RO	No	ECAT only. Transmit PDO mapping parameters;
1A07/8-1A08/8	TPDO mapping parameter	UINT32	RW	No	ECAT only. Transmit PDO mapping parameters;
1A0A/1	TPDO mapping parameter	UINT32	RO	No	ECAT only. Transmit PDO mapping parameters;
1A0B/2	TPDO mapping parameter	UINT32	RO	No	ECAT only. Transmit PDO mapping parameters.
1A0C/1 - 1A24/1	TPDO mapping parameter	UINT32	RO	No	ECAT only. Transmit PDO mapping parameters.
1C00/4	SM Communication type	UINT8	RO	No	ECAT only
1C10/0	SM0 PDO assignment	UINT16	RW	No	ECAT only, NOT TO BE USED (CTT only)
1C11/0	SM1 PDO assignment	UINT16	RW	No	ECAT only, NOT TO BE USED (CTT only)
1C12/30	SM2 (Outputs) PDO assignment	UINT16	RW	No	ECAT only
1C13/35	SM3 (Inputs) PDO assignment	UINT16	RW	No	ECAT only
1C32/32	Sync Manager 2 output parameters	UINT32, UINT16	Sub 1, 7, 8, 10: RW; Rest Sub: RO	No	ECAT only ,ECAT Outputs

Object(Hex) /Hi Sub(Dec)	Name	Data Type	Attribute	Mappable?	Comment
1C33/32	Sync Manager 3 input parameters	UINT32, UINT16	Sub 0, 2, 6, 9, 11, 14, 32: RO, Rest sub: RW	No	ECAT only ,ECAT Inputs
2005/0	Fast reference	INT32	RW	Yes	CAN, ECAT
2012/0	Set binary Interpreter object	UINT64	WO	RxMap	CAN only. Map to rPDO2
2013/0	Get binary Interpreter object	UINT64	RO	TxMap	CAN only. Map to tPDO2
2020/5	Home Block limit parameters	UINT32, UINT16	Sub 0: RO, Sub 1-5: RW	No	CAN, ECAT. Sub 4 OV[64], Sub 5 OV[65]
2030/16	Upload recording data	UINT64	RO	No	CAN only
2035/0	Upload data parameters	UINT32	RW	No	CAN only
2036/0	Upload data (UL)	UINT64	RO	No	CAN only
2041/0	Time stamp uSec resolution	UINT32	RO	TxMap	CAN, ECAT
2045/0	Block upload Inhibit time parameter	UINT16	RW	No	CAN only
2046/0	Distributed clock inhibit time	UINT16	RW	No	ECAT only. In mSec
2051	Download data (DL)	UINT64	WO	No	CAN only
2060/0	Parameters Checksum	UINT16	RO	No	CAN, ECAT
2061/0	FoE Download Parameters Error	UINT16	RO	No	ECAT only
2062/0	FoE Parameters Last String Send To Drive	STRING	RO	No	ECAT only
207B/2	Additional Position range limit	INT32	Sub 0: RO, Sub 1,2: RW	No	CAN, ECAT. Modulo range
2081/5	Extended error code	INT32	RO	No	CAN, ECAT. Reflects **EE[]**
2082/0	CAN controller status	UINT32	RO	TxMap	CAN only, OV[60]
2085/0	Extra Status register	INT16	RO	TxMap	CAN, ECAT, OV[61]
2086/0	STO Status Register	UINT32	RO	No	CAN, ECAT, OV[62]
2087/0	PAL Version	UINT16	RO	No	CAN, ECAT
2090/0	CAN DF implementation	UINT32	WO	No	CAN only
20A0/0	Additional Position in UU	INT32	RW	TxMap	CAN, ECAT
20B0/9	Socket additional function	UINT32	Sub 0: R Sub 1-9: RW	No	CAN, ECAT
20E0/0	ECAT alias object	UINT16	RW	No	ECAT only
20FC/2	Absolute Sensors	UINT16	WO	No	CAN, ECAT

Object(Hex) /Hi Sub(Dec)	Name	Data Type	Attribute	Mappable?	Comment
	Functions				
20FD/0	Digital input (0x60FD alias)	UINT32	RW	No	CAN, ECAT. Allows write function
2201/0	Low byte of DS402 Digital inputs	UINT8	RO	TxMap	CAN only
2202/3	Extended input	UINT32	**ECAT:** Sub 0,1: RO, Sub 2,3: RW **CAN:** Sub 0: RO, Sub 1-3: RW	**ECAT:** Sub-index1 TxMap; **CAN:** Sub-index 1 TxMap	CAN, ECAT
2203/0	Application Object	UINT32	RO	TxMap	CAN, ECAT
2205/2	Analog input	INT16	RO	**ECAT:** Sub-index1 TxMap; **CAN:** Sub-index 1 Sub-index 2 TxMap	CAN, ECAT. Sub-index 1: in mVolts Sub-index 2: 0 – 4095 (A2D ticks)
2206/0	5V DC supply	UINT16	RO	TxMap	CAN, ECAT. In mV
22A0/0	Digital output	UINT8	RW	RxMap	CAN only, GP output only
22A1/3	Extended outputs	UINT32	Sub 0: RO, Sub 1-3: RW	**CAN:** Sub 1 RxMap, **ECAT:** Sub 1 RxMap	CAN, ECAT.
22A2/0	Drive Temperature in °C	UINT16	RO	TxMap	CAN only, Legacy object
22A3/3	Temperature	UINT16	RO	CAN Sub 1 TxMap	CAN, ECAT. Similar to **TI[]**
2E00/0	Gain scheduling manual index	UINT16	RW	RxMap	CAN, ECAT
2E06/0	Torque window	UINT16	RW	No	CAN only, OF[50], TR[5]
2E07/0	Torque window time	UINT16	RW	No	CAN only, OF[51], TR[6]
2E10/0	Set HOME Position according to last Touch Probe capture.	UINT16	RW	No	CAN, ECAT
2E15/0	Gantry YAW offset	INT16	RW	No	CAN, ECAT. Reflected in **TW[14]**
2F00/24	General purpose User Integer array	INT32	RW	CAN: RxMap, TxMap	CAN,ECAT. Reflects **UI[]**

Object(Hex) /Hi Sub(Dec)	Name	Data Type	Attribute	Mappable?	Comment
				ECAT: No	
2F01/24	General purpose User Float array	FLOAT	RW	CAN: RxMap, TxMap ECAT: No	CAN, ECAT. Reflects **UF[]**
2F05/0	Get drive control board type	UINT16	RO	No	CAN, ECAT. Similar to **WS[8]**
2F20/4	TPDO Asynchronous events	UINT32	RW	No	CAN only
2F21/0	Emergency event mask	UINT16	RW	No	CAN only
2F41/0	Configuration object	UINT32	RW	No	CAN, ECAT
2F45/4	Threshold parameter object	INT32	Sub 0: RO, Sub 1-4: RW	No	CAN, ECAT
2F70/2	CAN encoder range	INT32	Sub 0: RO, Sub 1-2: RW	No	CAN only
2F75/0	Extrapolation Cycles Timeout	INT16	RW	No	CAN, ECAT, OV[63]
0x3000 to 0x3300	Elmo legacy commands	UINT32	RW	No	CAN, ECAT
6007/0	Abort connection option code	INT16	RW	No	CAN, ECAT
603F/0	Error Code	UINT16	RO	No	CAN, ECAT
6040/0	Control word	UINT16	RW	RxMap	CAN, ECAT
6041/0	Status word	UINT16	RO	TxMap	CAN, ECAT
605A/0	Quick stop option code	INT16	RW	No	CAN, ECAT
605B/0	Shut down option code	INT16	RW	No	CAN, ECAT
605C/0	Disable operation option code	INT16	RW	No	CAN, ECAT
605D/0	Halt option code	INT16	RW	No	CAN, ECAT
605E/0	Fault reaction option code	INT16	RW	No	CAN, ECAT
6060/0	Modes of Operation	INT8	RW	**CAN:** RxMap, TxMap **ECAT:** RxMap	CAN, ECAT
6061/0	Modes Of operation display	INT8	RO	TxMap	CAN, ECAT
6062/0	Position demand value	INT32	RO	TxMap	CAN, ECAT
6063/0	Position actual internal value	INT32	RO	TxMap	CAN, ECAT
6064/0	Position actual value	INT32	RO	TxMap	CAN, ECAT
6065/0	Following error window	UINT32	RW	No	CAN, ECAT
6066/0	Following error time out	UINT16	RW	No	CAN, ECAT
6067/0	Position Window	UINT32	RW	No	CAN, ECAT

Object(Hex) /Hi Sub(Dec)	Name	Data Type	Attribute	Mappable?	Comment
6068/0	Position Window time	UINT16	RW	No	CAN, ECAT
6069/0	Velocity sensor actual value	INT32	RO	TxMap	CAN, ECAT
606A/0	Sensor selection code	INT16	RW	No	CAN, ECAT
606B/0	Velocity demand value	INT32	RO	TxMap	CAN, ECAT
606C/0	Velocity actual value	INT32	RO	TxMap	CAN, ECAT. In accordance with 606A
606D/0	Velocity window	UINT16	RW	No	CAN, ECAT
606E/0	Velocity window time	UINT16	RW	No	CAN, ECAT
606F/0	Velocity threshold	UINT16	RW	No	CAN, ECAT
6070/0	Velocity threshold time	UINT16	RW	No	CAN, ECAT
6071/0	Target Torque	INT16	RW	**ECAT:** RxMap **CAN:** RxMap, TxMap	CAN, ECAT
6072/0	Maxl torque	UINT16	RW	**ECAT:** RxMap **CAN:** RxMap, TxMap	CAN, ECAT
6073/0	Max current	UINT16	RW	**ECAT:** RxMap **CAN:** RxMap, TxMap	CAN, ECAT
6074/0	Torque Demand	INT16	RO	TxMap	CAN, ECAT
6075/0	Motor rated current	UINT32	RW	No	CAN, ECAT
6076/0	Motor rated torque	UINT32	RW	No	CAN, ECAT
6077/0	Torque actual value	INT16	RO	TxMap	CAN, ECAT
6078/0	Current actual value	INT16	RO	TxMap	CAN, ECAT
6079/0	DC link circuit voltage	UINT32	RO	**ECAT**:TxMap	CAN, ECAT
607A/0	Target Position	INT32	RW	**ECAT:** RxMap **CAN:** RxMap, TxMap	CAN, ECAT
607B/2	Position range limit	INT32	Sub 0: RO, Sub 1,2: RW	No	CAN, ECAT
607C/0	Home offset	INT32	RW	No	CAN, ECAT
607D/2	Software position limit	INT32	Sub 0: RO, Sub 1,2: RW	No	CAN, ECAT
607E/0	Polarity (speed & position)	UINT8	RW	**ECAT**: RxMap **CAN**: RxMap, TxMap	CAN, ECAT
607F/0	Max profile velocity	UINT32	RW	No	CAN, ECAT
6080/0	Max motor speed	UINT32	RW	No	CAN, ECAT
6081/0	Profile velocity	UINT32	RW	**ECAT:** RxMap	CAN, ECAT

Object(Hex) /Hi Sub(Dec)	Name	Data Type	Attribute	Mappable?	Comment
				CAN: RxMap, TxMap	
6082/0	End velocity	UINT32	RW	**ECAT:** RxMap **CAN:** RxMap, TxMap	CAN, ECAT
6083/0	Profile acceleration	UINT32	RW	**ECAT:** RxMap **CAN:** RxMap, TxMap	CAN, ECAT
6084/0	Profile deceleration	UINT32	RW	**ECAT:** RxMap **CAN:** RxMap, TxMap	CAN, ECAT
6085/0	Quick stop deceleration	UINT32	RW	**ECAT:** RxMap **CAN:** RxMap, TxMap	CAN, ECAT
6086/0	Motion profile type	INT16	RW	No	
6087/0	Torque slope	UINT32	RW	**ECAT:** RxMap **CAN:** RxMap, TxMap	CAN, ECAT
608F/2	Position encoder resolution	UINT32	Sub 0: RO, Sub 1,2: RW	No	CAN, ECAT
6090/2	Velocity Encoder resolution	UINT32	Sub 0: RO, Sub 1,2: RW	No	CAN, ECAT
6091/2	Gear ratio	UINT32	Sub 0: RO, Sub 1,2: RW	No	CAN, ECAT
6092/2	Feed constant	UINT32	Sub 0: RO, Sub 1,2: RW	No	CAN, ECAT
6093/0	Position factor of DS402	UINT32	RO	No	CAN only
6094/0	Velocity encoder factor of DS402	UINT32	RO	No	CAN only
6095/0	Velocity_factor_1 of DS402	UINT32	RO	No	CAN only
6096/2	Velocity factor	UINT32	Sub 0: RO, Sub 1,2: RW	No	CAN, ECAT
6097/2	Acceleration factor	UINT32	Sub 0: RO, Sub 1,2: RW	No	CAN, ECAT
6098/0	Homing Method	INT8	RW	No	CAN, ECAT
6099/2	Homing speeds	UINT32	Sub 0: RO, Sub 1,2: RW	No	CAN, ECAT
609A/0	Homing acceleration	UINT32	RW	No	CAN, ECAT
60B0/0	Position offset	INT32	RW	**ECAT:** RxMap **CAN:** RxMap	CAN, ECAT
60B1/0	Velocity offset	INT32	RW	**ECAT:** RxMap	CAN, ECAT

Object(Hex) /Hi Sub(Dec)	Name	Data Type	Attribute	Mappable?	Comment
				CAN: RxMap	
60B2/0	Torque offset	INT16	RW	**ECAT:** RxMap **CAN:** RxMap	CAN, ECAT
60B8/0	Touch probe function	UINT16	RW	RxMap	CAN, ECAT
60B9/0	Touch probe status	UINT16	RO	TxMap	CAN, ECAT
60BA/0	Touch probe 1 positive edge	INT32	RO	TxMap	CAN, ECAT
60BB/0	Touch probe 1 negative edge	INT32	RO	TxMap	CAN, ECAT
60BC/0	Touch probe 2 positive edge	INT32	RO	TxMap	CAN, ECAT
60BD/0	Touch probe 2 negative edge	INT32	RO	TxMap	CAN, ECAT
60C0/0	Interpolation sub mode select	INT16	RW	No	CAN only
60C1/2	interpolation data record	INT32	Sub 0:RO, Sub 1,2: RW	RxMap	CAN only
60C2/2	interpolation time period	INT8	Sub 0:RO, Sub 1,2: RW	**CAN:** RxMap **ECAT:** Sub 1: RxMap	CAN, ECAT
60C4/6	interpolation data configuration	INT16	RW	No	CAN only
60C5/0	Max acceleration	UINT32	RW	No	CAN, ECAT
60C6/0	Max deceleration	UINT32	RW	No	CAN, ECAT
60E3/33	Supported Homing Methods	UINT8	RO	No	CAN, ECAT
60E4/0	Additional Position Actual Value	INT32	RO	No	CAN only
60E5/0	Additional Velocity Actual Value	INT32	RO	No	CAN only
60F2/0	Positioning option code	UINT16	RW	No	CAN, ECAT
60F4/0	Following error actual value	INT32	RO	TxMap	CAN, ECAT
60FA/0	Control effort	INT32	RO	TxMap	CAN, ECAT
60FC/0	Position demand internal value	INT32	RO	TxMap	CAN, ECAT
60FD/0	Digital inputs	UINT32	RO	TxMap	CAN, ECAT
60FE/2	Digital outputs	UINT32	Sub 0: RO, Sub 1,2: RW	**CAN:** Sub 1 RxMap, **ECAT:** Sub1 RxMap	CAN, ECAT
60FF/0	target velocity	INT32	RW	**CAN:** RxMap	CAN, ECAT

Object(Hex) /Hi Sub(Dec)	Name	Data Type	Attribute	Mappable?	Comment
				ECAT: RxMap	
6502/0	Supported Drive Modes	UINT32	RO	No	CAN, ECAT

Table 2-2: Object Dictionary

Chapter 3: Device Type and Device Name Objects

0x1000: Device type

0x1008: Manufacturer device name

The following objects are described in MAN-G_DS301:

- 0x1000: Device Type. This object contains information about the device type and functionality. Return 0x20192 on read access. This means that the device is servo drive supporting the DS-402 device profile.
- Object 0x1008: Manufacture Device Name. This object contains the manufacturer device name, such as *Guitar*, *Trombone*, *Whistle*.

Chapter 4: Error Control Objects

4.1. General

Object	Definition
0x1001	Error register
0x1003	Pre-defined error field
0x2F21	Emergency event mask
0x6007	Abort connection option code
0x603F	Error Code

The drive functionality in case of an error is determined using the following objects, that are described below: 0x6007, 0x603F.

Additionally, the following objects are described in MAN-G_DS301:

- 0x1001: Error Register. The object returns error register of last error occurred.
- 0x1003: Pre-defined Error Field. This object holds the errors that have occurred in the device and have been signaled via the Emergency object.
- 0x2F21: Emergency Event Mask. This object selects events as the cause for transmitting emergency objects.

4.2. Object 0x6007: Abort connection option code

This object defines the motor control behavior when the following events occurred.

For CANopen communication:

- bus-off
- heartbeat
- NMT stopped state entered via NMT command

Note: The NMT reset communication and NMT reset application functions disables the motor immediately regardless to this object settings.

For EtherCAT communication:

- This object defines the behavior when the drive is enabled and in OP state and is requested to switch to any other state (e.g. PreOP state).

  This object defines the host EtherCAT communication watchdog and watchdog time elapsed.

- Object description:

Attributes	0x6007
Name	Abort connection option code
Object code	VAR
Data type	INTEGER16
Category	Optional

- Entry description:

Access	Read/Write
PDO mapping	No
Value range	-32768…32767
Default value	1

- Data description. Command details are found in the Gold Command Reference Manual:

Option Code	Description	Details
0	No action	
1	Set Fault Signal	The drive behavior is defined by *Fault reaction option code*, object **0x605E**. Besides the defined behavior, the following procedures are executed: • Motor is disabled (**MO=0**) • Motor Fault is set to 0x800 (**MF** command) and

Option Code	Description	Details
		possibly execute the user program AUTOERR routine. • Emergency message with error code 0x8130 will be sent (if not masked, see object **0x2F21**)
2	Device control command *Disable voltage*	Not supported
3	Device control command *Quick stop*	• Object **0x605A**. An emergency message with error code 0x8130 will be sent (if not masked, see object **0x2F21**)

4.3. Object 0x603F: Error code

This object captures the code of the last fault that occurred in the drive and initiates EMCY message transmission. It corresponds to the value of the lower 16 bits of object **0x1003** sub index 1, *pre-defined error field*.

The optional Error codes are described in the *Emergency section* of the MAN-G-DS-301 manual.

- Object description:

Attributes	0x603F
Name	Error code
Object code	VAR
Data type	UNSIGNED16
Category	Optional

- Entry description:

Access	Read only
PDO mapping	No
Value range	0…65535
Default value	0

Chapter 5: Drive Data Objects

5.1. General

Object	Definition
0x20FD	Digital Input
0x2201	Low byte of DS402 Digital inputs
0x2202	Extended Input
0x2203	Application Object
0x2205	Analog Input
0x22A0	Digital Output
0x22A1	Extended Outputs
0x60FD	Digital Inputs
0x60FE	Digital Outputs

The following objects present drive data and are described below: 0x60FE, 0x60FD.

Additionally, the following objects are described in MAN-G_DS301:

- 0x20FD: Digital Input. The object presents digital inputs status on read access Data placement is the same as in 0x60FD. Sticky bits can be reset on write access. Refer to **IL[]** command for further details about sticky bits.
- 0x2201: Low byte of DS402 Digital inputs. This object defines least 8 bits of object **0x20FD**, simple digital inputs for drives.
- 0x2202: Extended Input. The object presents extended inputs similar to XI[n] command
- 0x2203: Application Object. The object is a pointer object which reports different parameters depending on the setting of object **0x2F41** bits 16 to 19.
- 0x2205.1: Analog Input 1. This object returns the value of the analog inputs 1 in mV.
- 0x2205.2: Analog Input 2. This object returns the value of analog input 2 in A2D counts (0-4095)
- 0x22A0: Digital Output. The object returns the value of 8 bit digital outputs
- 0x22A1: Extended Outputs. The object presents extended outputs similar to XO[n] command

5.2. Object 0x60FD: Digital inputs

This object presents the digital inputs status on read access. Bit placement is presented in the following table (1: means logical active state, 0: logical not active).

For more details about the digital input functionality refer to **IL[]** command, **IP** command and **IB[]** command in the Gold command reference manual (MAN_G_CR).

- Object description:

Attributes	0x60FD
Name	Digital inputs
Object code	VAR
Data type	UNSIGNED32
Category	Optional

- Entry description:

Access	Read only
PDO mapping	Yes
Value range	0…(2^{32})-1
Default value	No

- Data description:

31 16	15 4	3	2	1	0
Logical state of digital inputs 1…16 equals to **IP** command bits 16…31	Reserve, 0	Interlock	Main Home Switch	FLS	RLS

Interlock bit is set to'1' when one (or both) of the STO inputs is disabled and the drive is in safety state. (from firmware version 1.1.10.7 B00)

Note: **Object 0x20FD can be used for the same purpose as 0x60FD but with a Write Access. The Write Access is required when the digital input is defined as "sticky-bit", after the input is set to '1'. The host is required to confirm it by writing '1' to this input which is then reset back to 0. See IL[] command (MAN_G_CR).for more details about "sticky-bits".**

5.3. Object 0x60FE: Digital outputs

The object sets or resets general purpose digital outputs.

Note: Bit 0 is defined as "Brake". For the Elmo drive, the brake function is performed automatically when the servo is on or off. For more information about the digital output refer to OL[] command, OB[] command and OP command in (MAN_G_CR).

The bit placement is:

31…….22	21	20	19	18	17	16	15…1	0
Reserved	DO6	DO5	DO4	DO3	DO2	DO1	Reserve	Brake not used

- Object description:

Attributes	0x60FE
Name	Digital outputs
Object code	VAR
Data type	UNSIGNED32
Category	Optional

- Entry description:

Sub-index	0
Description	Highest sub index
Entry category	Mandatory
Access	RO
PDO mapping	No
Value range	2
Default value	2

Sub-index	1
Description	Physical Outputs
Entry category	Mandatory
Access	RW
PDO mapping	RxMap
Value range	0…(2^{32})-1
Default value	0

Sub-index	2
Description	Bit Mask
Entry category	Optional
Access	TRW
PDO mapping	RxMap
Value range	0…(2^{32})-1
Default value	0

Chapter 6: Device Control Objects

6.1. General

Object	Definition
0x6040	*Controlword*
0x6041	*Statusword*

The following objects are presented in this chapter: 0x6040 *Controlword* and 0x6041 *Statusword*.

The Device Control function block controls all functions of the device, categorized as:

- Device control of the state machine
- Operation mode functions

The state of the device is controlled by the *Controlword,* while the status of the device is indicated by the *Statusword.*

The state machine is controlled externally by the *Controlword* and external signals. Write access to the *Controlword* is always allowed. The Gold drive is **always in external mode, thus the *Remote* indication in the *Statusword* is always '1'.** In addition, the state machine is controlled by internal signals such as faults and modes of operation.

The following diagram illustrates the Device Control function.

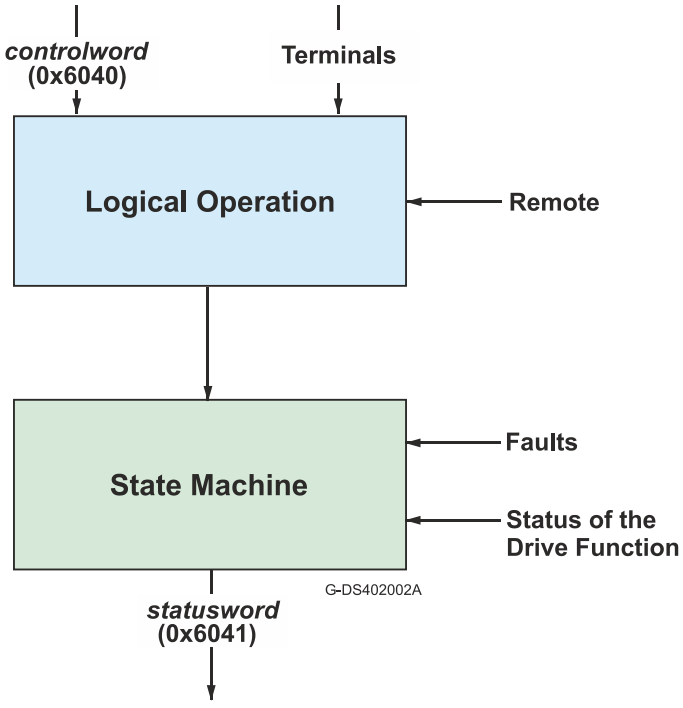


Figure 6-1: Remote Mode

Note: **The Elmo drive is always in remote mode; i.e. it can only be controlled externally using the SDO and PDO.**

6.2. State Machine

The state machine describes the device status and the possible control sequence of the drive. A single state represents a special internal or external behavior. The state of the drive also determines which commands are accepted; for example, a point-to-point motion can be started only when the drive is in OPERATION ENABLED state.

States may be changed using the *Controlword* and/or according to internal events. The current state can be read using the *Statusword*.

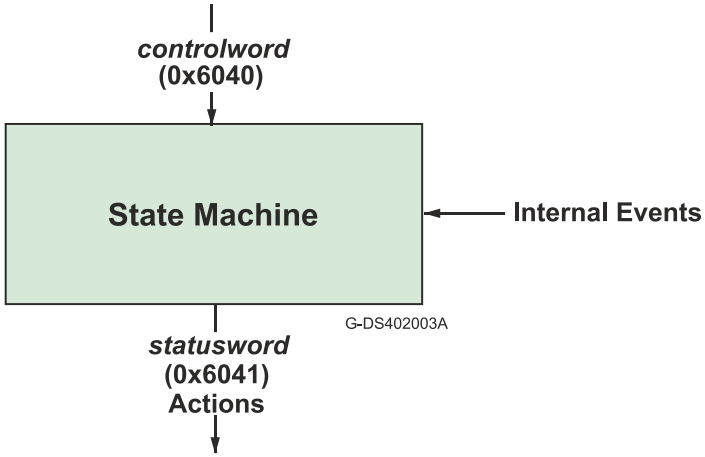


Figure 6-2: State Machine in System Context

The device states and possible control sequence of the drive are described by the state machine, as depicted in the following figure:

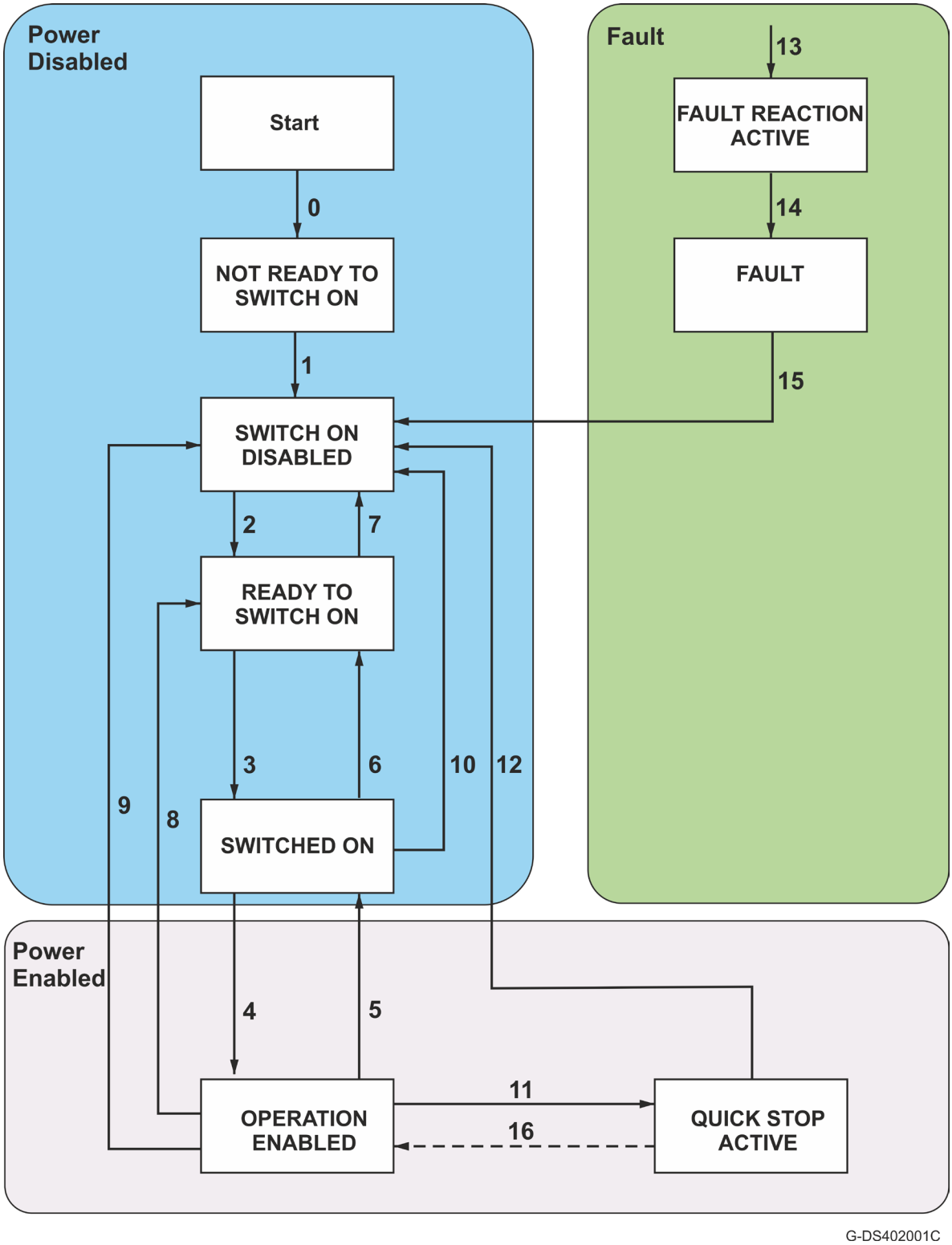


Figure 6-3: State Machine Block Diagram

6.3. Drive States

The drive states are as follows:

Drive State	Process	Description
NOT READY TO SWITCH ON	Low-level power (24V) has been applied to the drive. The drive is being initialized and is running the self-test. A brake output, if present, is applied in this state.	The drive function is disabled. This state is an internal state in which communication is enabled only at the end. The user can neither retrieve nor monitor this state.
SWITCH ON DISABLED	Drive initialization is complete. The drive parameters have been set up. Drive parameters may be changed.	SWITCH ON DISABLED is the minimum state to which a user may switch. The drive function is disabled. In this state, there is no error indication even if high power is applied. The application must be responsible for handling the state transition.
READY TO SWITCH ON	High voltage may be applied to the drive. The drive parameters may be changed. The drive function is disabled.	
SWITCHED ON	High voltage has been applied to the drive. The power amplifier is ready. The drive parameters may be changed.	The drive function is disabled. In this state, there is no error indication if the drive high voltage has not been applied.
OPERATION ENABLED	No faults have been detected. The drive function is enabled and power is applied to the motor. The drive parameters may be changed.	This state is the normal operation of the drive In this state, a brake is automatically released according to the brake parameter **(BP[N])** timing.
QUICK STOP ACTIVE	The drive parameters may be changed. The quick stop function is being executed. The drive function is enabled	In this state, according to the quick stop option code, the drive stops the motion with a quick stop deceleration or profiler deceleration and either stays in

Drive State	Process	Description
	and power is applied to the motor.	quick stop state or switches to SWITCH ON DISABLED state. In addition, there is the option to go to SWITCH ON DISABLED state immediately without stopping. The term *drive stops* means that the rfg completed the deceleration trajectory and not that the motor is stationary.
FAULT REACTION ACTIVE	The drive parameters may be changed. A fault has occurred and the fault reaction function is being executed. Following that, the drive automatically switches to FAULT state.	In this state, according to the fault reaction option code, the drive stops the motion with quick stop deceleration or profiler deceleration, and the drive function is disabled. In addition, there is the option to immediately disable a drive function without stopping it.
FAULT	The drive parameters may be changed. A fault has occurred in the drive.	In this state the high voltage switch-on/-off depends on the application. The drive function is disabled.

6.4. State Transitions of the Drive

State transitions are a result of internal events in the drive or commands from the host through the *Controlword*.

When a command that causes a change of state is received, it is processed completely and the new state is attained before the next command is processed.

The drive performs transitions 0 and 1 after initiation, either at power up or in case of CAN communication at NMT node reset. This state remains until there is a change due to a received host command.

The FAULT OCCURRED indication implies that a fault has occurred in the drive during OPERATION ENABLED state. This results in a transition to the FAULT REACTION ACTIVE state, during which the device executes a motor disable function. After executing this fault reaction, the device switches to the **fault** state. The transition from Fault state to switch-on-disable requires a Fault Reset command. The transition will happened even if the fault still exists.

If an error occurs in OPERATION ENABLE state, an emergency message (if not masked via object 0x2F21) is sent with the fault reason. The last 16 fault messages are latched and are retrieved by via object 0x1003, defined in DS301_MAN.

In a fault state, setting **MO=1** through methods other than the *Controlword* activates the motor and leads to an ambiguous state of the DSP 402 protocol. Note that if the ControlWord is cyclically transmitted than the **MO** setting will be override .

Transition State	Mode	Event	Action
State Transition 0	From **START** to **NOT READY TO SWITCH ON**	Reset	The drive self-tests and/or self-initializes.
State Transition 1	From **NOT READY TO SWITCH ON** to **SWITCH ON DISABLED**	The drive has completed self-test and/or initialized successfully.	Activate communication.
State Transition 2	From **SWITCH ON DISABLED** to **READY TO SWITCH ON**	Shutdown command received from host.	None
State Transition 3	From **READY TO SWITCH ON** to **SWITCHED ON**	Switch On command received from host.	The power section is switched on if it is not already on.
State Transition 4	From **SWITCHED ON** to **OPERATION ENABLED**	Enable Operation command received from host.	The drive function is enabled
State Transition 5	From **OPERATION ENABLED** to **SWITCHED ON**	Disable Operation command received from host.	The drive operation is disabled.

Transition State	Mode	Event	Action
State Transition 6	From **SWITCHED ON** to **READY TO SWITCH ON**	Shutdown command received from host.	The power section is switched off.
State Transition 7	From **READY TO SWITCH ON** to **SWITCH ON DISABLED**	Quick Stop or Disable Voltage commands received from host.	None.
State Transition 8	From **OPERATION ENABLED** to **READY TO SWITCH ON**	Shutdown command received from host.	The power section is switched off immediately, and the motor is free to rotate if no brake is applied.
State Transition 9	From **OPERATION ENABLED** to **SWITCH ON DISABLED**	Disable Voltage command received from host.	The power section is switched off immediately, and the motor is free to rotate if no brake is applied.
State Transition 10	From **SWITCHED ON** to **SWITCH ON DISABLED**	Disable Voltage or Quick Stop command received from host.	The power section is switched off immediately, and the motor is free to rotate if no brake is applied.
State Transition 11	From **OPERATION ENABLED** to **QUICK STOP ACTIVE**	Quick Stop command received from host.	The quick stop function is executed. If the drive in time depended modes (ip, csp) the interpolation stops and the relevant bits in the status word are set. In hm mode, the home error bit is set.
State Transition 12	From **QUICK STOP ACTIVE** to **SWITCH ON DISABLED**	This transition is performed automatically if the quick stop option code, **0x605A**, is >4.	The profile generator finished the deceleration and the motor is disabled.
State Transition 13	From **OPERATION ENABLED** to **FAULT REACTION ACTIVE**	A fault has occurred in the drive.	Execute appropriate fault reaction. Note that the fault reaction is override if the fault includes power stage (e.g. over voltage) or sensor faults.
State Transition 14	From **FAULT REACTION ACTIVE** to	The fault reaction is	The drive function is

Transition State	Mode	Event	Action
	FAULT	completed.	disabled.
State Transition 15	From **FAULT** to **SWITCH ON DISABLED**	Fault Reset command received from host.	The fault state is reset. After resetting the Fault state via the *Controlword* Fault reset bit (bit 7), the Fault Reset bit of the *Controlword* must be cleared by the host. The drive does not monitor this bit in other states. If this bit is not cleared from a previous fault state, when the next fault occurs, the drive may automatically transit to **switch on disabled** state without any indications or warning.
State Transition 16	From **QUICK STOP ACTIVE** to **OPERATION ENABLED**	Enable Operation command received from host. This transition is possible if the quick stop option code (**Object 0x605A**) is set to 5 or 6.	The drive function is enabled. This transition forces a *motion begin*. If the motor is turned off by an external source (such as the interpreter) during **operation enable**, the minimum state **switched on** will be entered with no further notification.

6.4.1. Illegal Transition

After initiation of a drive by either power up on or NMT node reset, the drive automatically performs transitions 0 and 1 to the **SWITCH ON DISABLED** state. The *Controlword* can then be used to initiate any of the transitions defined previously. If a transition is illegal (such as requesting a **QUICK STOP** in a **FAULT** state), it is ignored. No indication is provided by the drive in such occasions.

6.5. Object 0x6040: *Controlword*

The *Controlword* contains bits for:

- Controlling the state
- Controlling operating modes
- Manufacturer-specific options
- Object description:

Attributes	0x6040
Name	*Controlword*
Object code	VAR
Data type	UNSIGNED16
Category	Mandatory

- Entry description:

Access	Read/write
PDO mapping	Yes
Value range	0…65535
Default value	0

- Data description:

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:

O: Optional

M: Mandatory

6.5.1. Bits 0 – 3 and 7

Device control commands are triggered by the following bit patterns in the *Controlword*:

Command	Bits of the *Controlword*					Transitions
	7	**3**	**2**	**1**	**0**	
	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 and Enable Operation	0	1	1	1	1	3+4 Automatic transition to Enable operation state after executing SWITCHED ON state functionality
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

Table 6-1 Device Control Command Triggers

Bits marked with X are not relevant.

6.5.2. Bits 4, 5, 6 and 8

These bits are operation-mode specific. Their description is found in the chapter about the special mode. The following table summarizes the name and functionality of each Bit with respect to the Operation Mode.

Operation Mode	Bits of the *Controlword*			
	8	**6**	**5**	**4**
Profile Position Mode	Blending mode	relative	Change set immediately	New set-point
Profile Velocity Mode	Reserved	Reserved	Reserved	Reserved
Profile Torque Mode	Reserved	Reserved	Reserved	Reserved
Homing Mode	Reserved	Reserved	Reserved	Homing operation start
Interpolation Position Mode	Reserved	Reserved	Reserved	Enable Interpolation
CSP	Reserved	Reserved	Reserved	Reserved
CSV	Reserved	Reserved	Reserved	Reserved
CST	Reserved	Reserved	Reserved	Reserved

6.5.3. Bit 8

This bit enables the *Halt* function in all modes: pp, pv, tq, hm, ip, csp, csv and cst.

6.5.4. Bit 10

This bit is reserved for future use. It is de-activated by setting it to 0. If it has no special function, it is set to zero.

6.5.5. Bits 11, 12, 13, 14 and 15

These bits are manufacturer specific.

6.6. Object 0x6041: *Statusword*

The *Statusword* indicates the present state of the drive. No bits are latched. The *Statusword* contains bits for:

- The current drive state
- The operating state of the mode
- Manufacturer-specific options
- Object description:

Attributes	0x6041
Name	*Statusword*
Object code	VAR
Data type	UNSIGNED16
Category	Mandatory

- Entry description:

Access	Read only
PDO mapping	TxMap
Value range	0 - 65535
Default value	No

- Data description:

Bit	Description
0	Ready to switch on
1	Switched on
2	Operation enabled
3	Fault
4	Voltage enabled
5	Quick stop
6	Switch on disabled
7	Warning
8	Manufacturer specific, reserved, always set to 0
9	Remote, always set to 1.
10	Target reached
11	Internal limit active
12 - 13	Operation mode specific
14 - 15	Manufacturer specific, reserved, always set to 0

6.6.1. Bits 0 - 3, 5 and 6

The following bits indicate the status of the device:

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	Switch 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

6.6.2. Bit 4: Voltage Enabled

High voltage is applied to the drive when this bit is set to 1.

6.6.3. Bit 5: Quick Stop

When reset to 0, this bit indicates that the drive is reacting to a Quick Stop request or already in Quick Stop state. Bits 0, 1 and 2 of the *Statusword* must be set to 1 to indicate that the drive is capable of regenerating. The setting of the other bits indicates the status of the drive (for example, the drive is performing a quick stop in reaction to a non-fatal fault. The fault bit is set in addition to bits 0, 1 and 2).

6.6.4. Bit 7: Warning

A warning is indicated when a predefined parameter threshold is violated. The threshold can be set via **XT[]** (**Object 0x3269**).

A Warning bit is set in any of the following cases:

- Temperature threshold is exceeded
- Over voltage threshold is exceeded
- Under voltage threshold is exceeded
- Analog encoder amplitude low threshold is exceeded
- End at encoder warning is received

6.6.5. Bit 8

This bit is reserved for the manufacturer. It is not used and is set to 0.

6.6.6. Bit 9: Remote

If bit 9 is set, parameters may be modified via the CAN network, and the drive executes the contents of a command message. If the bit remote is reset, the drive is in local mode and does not

execute the command message. The drive may transmit messages containing actual valid values such as a *position actual value*, depending on the actual drive configuration. The drive accepts accesses via SDO in local mode.
The Remote bit is always set by the Elmo drive.

6.6.7. Bit 10: Target Reached

Bit 10 is set by the drive to indicate that a set-point has been reached. The set-point is dependent on the operating mode. The relevant description is found in the chapter about the special mode. The change of a target value by software alters this bit.

If the *quick stop option code* is 5 or 6, this bit is set when the quick stop operation is finished and the drive is halted.

If a Halt occurs and the drive has halted, this bit is also set.

If the motor axis was stopped by FLS, RLS or STOP switch the bit is reset.

6.6.8. Bit 11: Internal Limit Active

The drive sets this bit to indicate that an internal limitation is active. An Internal Limit includes the following; *software position limit, FLS, RLS* and *STOP switch*.

6.6.9. Bits 12 and 13

These bits are operation-mode specific. Their description is found in the chapter about the specific mode. For detailed information refer to the relevant mode chapter.

6.6.10. Bits 14 and 15

These bits are reserved. They are not used and are set to 0.

Chapter 7: Halt, Stop and Fault Objects

7.1. General

Object	Definition
0x605A	Quick stop option code
0x605B	Shutdown option code
0x605C	Disable operation option mode
0x605D	Halt option code
0x605E	Fault reaction option code

Slow down ramp **Object 0x6084**

Quick stop ramp SD value (**Object 0x31D7**)

Disable drive MO=0

7.2. Object 0x605A: Quick stop option code

This Object determines the action to take if the Quick Stop function is executed. Quick Stop is determined in two cases:

- Request from the master to switch from Operation enable to Quick Stop state
- A communication error which is defined by the Abort Option code causing an auto stated machine transition from Operation Enable to Quick Stop (**Object 0x6007**)
- Object description:

Attributes	0x605A
Name	Quick stop option code
Object code	VAR
Data type	INTEGER16
Category	Optional

- Entry description:

Access	Read/write
PDO mapping	No
Value range	1, 2, 5, 6
Default value	2

- Data description:

Value	Description
-32,768…-1	Manufacturer specific, Not supported
0	Disable drive function, go to SWITCH ON DISABLED state
1	Slow down on slow-down ramp and then disable the drive function, go to SWITCH ON DISABLED state
2	Slow down on quick-stop ramp and then disable the drive function, go to SWITCH ON DISABLED state
3	Not supported
4	Not supported
5	Slow down on slow-down ramp and stay in QUICK STOP state
6	Slow down on quick stop ramp and stay in QUICK STOP state
7…32,767	Not supported

An attempt to set an unsupported value causes the transmission of abort code 0609 0030: Value exceeded.

7.3. Object 0x605B: Shutdown option code

This Object determines which action to take in the following transition:
From OPERATION ENABLED to READY TO SWITCH ON.

- Object description:

Attributes	0x605B
Name	Shutdown option code
Object code	VAR
Data type	INTEGER16
Category	Optional

- Entry description:

Access	Read/write
PDO mapping	No
Value range	-32768…32767
Default value	0

- Data description:

Value	Description
-32,768…-1	Manufacturer specific, not supported
0	Disable drive function
1	Slow down on slow-down ramp; disable drive function
2…32,767	Not supported

Note: **An attempt to set an unsupported value causes the transmission of abort code 0609 0030, value exceeded.**

7.4. Object 0x605C: Disable operation option code

This Object determines which action to take in the following transition:
from OPERATION ENABLED to SWITCHED ON.

- Object description:

Attributes	0x605C
Name	Disable operation option code
Object code	VAR
Data type	INTEGER16
Category	Optional

- Entry description:

Access	Read/write
PDO mapping	No
Value range	-32768…32767
Default value	0

- Data description:

Value	Description
-32,768…-1	Manufacturer specific, not supported
0	Disable drive function
1	Slow down on slow-down ramp and then disable drive function
2…32,767	Reserved, not supported

Note: **An attempt to set an unsupported value causes the transmission of abort code 0609 0030: Value exceeded.**

7.5. Object 0x605D: Halt option code

This Object determines which action to take if bit 8 (halt) in the *Controlword* is active.

- Object description:

Attributes	0x605D
Name	Halt option code
Object code	VAR
Data type	INTEGER16
Category	Optional

- Entry description:

Access	Read/write
PDO mapping	No
Value range	-32768…32767
Default value	2

- Data description:

Value	Description
-32,768…-1	Manufacturer specific, not supported
0	Reserved, not supported
1	Slow down on slow down ramp and stay in Operation Enabled
2	Slow down on quick stop ramp and stay in Operation Enabled
3	Not supported

7.6. Object 0x605E: Fault reaction option code

This Object indicates what motion will be performed in case of a drive fault.

The drive enters a fault state when a critical violation occurs during Operation Enabled state (servo is enabled). In any situation of a fault state in the *Statusword*, the drive will transmit an EMCY message and will inform the Fault state in **Object 0x6041**.

The following violations are unique:

- Physical violation:
  The drive disables the servo immediately.
  Typical cases: Over Voltage, Temperature, Safe Torque Off switch was set
- Operational violation:
  The drive performs the reaction as requested via **Object 0x605E**. The optional reactions are described in the table below. Typical cases: Speed tracking error, Feedback Position is out of position limits etc.

For more details refer to EMCY messages in MAN-G-DS301 and **MF** command in the Command reference manual.

- Object description:

Attributes	0x605E
Name	Fault reaction option code
Object code	VAR
Data type	INTEGER16
Category	Optional

- Entry description:

Access	Read/write
PDO mapping	No
Value range	0..2
Default value	2

- Data description:

Value	Description
-32,768…-1	Reserved
0	Disable drive, motor free to rotate
1	Slow down on slow-down-ramp to velocity 0, and then disable-motor
2	Slow down on quick-stop ramp to velocity 0, and then disable motor
3…32,767	Reserved

The following should be noted:

- Due to compatibility reasons the default value is set to 0, where the motor is disabled immediately after the fault.
- When a Fault occurs, the drive will always be in fault state. In cases where a deceleration to speed 0 is performed before the Fault state, a status of Fault Reaction State is informed via **Object 0x6041**.
- An attempt to set an unsupported value causes the transmission of abort code 0609 0030: Value exceeded.
- Slow down ramp is defined by **Object 0x6084**: Profile Deceleration **OF[6]** is the alias command for this Object.

Chapter 8: Modes of Operation

8.1. General

Object	Definition
0x6060	Modes of operation
0x6061	Modes of operation display
0x6502	Supported drive modes

8.2. Functional Description

The drive behavior depends on the activated modes of operation. Different modes can be implemented, although not in parallel. Therefore, the user must activate the required function by selecting a mode of operation. The modes-of-operation variables are initialized at reset to *no mode* (value 0). Modes can be set in any state, including OPERATION ENABLED.

If set to OPERATION ENABLED, the motor stands still until an explicit motion command is received via a *Controlword*. Bit 10 in the *Statusword* (Target reached) is set in all modes, exclusive cyclic synchronous position to mode 0x6060=8.

To prevent the motor shaft jumping, the control device should change mode only after a complete stop, according to the definition of target reached. The actual mode is reflected via **Object 0x6061**.

The *Statusword* contains bits whose meaning depends on the mode of operation. When switching modes, the *mode dependent* bits in the *Controlword* and *Statusword* must be monitored.

8.3. Object 0x6060: Modes of operation

This Object indicates the next requested operation mode. The actual operation mode is denoted by the **Object 0x6061**.

An attempt to access an unsupported mode results-in abort code 0609 0030: Value exceeded.

- Object description:

Attributes	0x6060
Name	Modes of operation
Object code	VAR
Data type	INTEGER8
Category	Mandatory

- Entry description:

Access	Read/write
PDO mapping	Yes
Value range	As detailed below
Default value	0*

- Data description:

Value	Description
*All values indicated with a *are not supported modes and cannot be written by the user	
-128…-4	Reserved, not supported*
-3	CAN encoder mode. Read access of 0x6060 and 0x6061 returns 7
-2, -1	Reserved, not supported*
0	Not supported*
1	Profile position mode
2	Velocity (not supported)*
3	Profiled velocity mode
4	Torque profiled mode
5	Reserved, not supported*
6	Homing mode
7	Interpolated position mode (available in CANopen only)
8	Cyclic synchronous position

Value	Description
9	Cyclic synchronous velocity
10	Cyclic synchronous torque
11…127	Reserved, not supported*

8.4. Object 0x6061: Modes of operation display

This Object shows the current mode of operation. The meaning of the returned value corresponds to that of the *modes of operation* option code (**Object 0x6060**).

- Object description:

Attributes	0x6061
Name	Modes of operation display
Object code	VAR
Data type	INTEGER8
Category	Mandatory

- Entry description:

Access	Read only
PDO mapping	TxMap
Value range	refer to table in **Object 0x6060**
Default value	0

- Data description:
  Similar to **Object 0x6060**, *modes of operation.*

The actual mode is reflected in the modes of operation display (**Object 0x6061**), and not in modes of operation (**Object 0x6060**).

8.5. Object 0x6502: Supported drive modes

- Object description:

Attributes	0x6502
Name	Supported drive modes
Object code	VAR
Data type	UNSIGNED32
Category	Optional

- Entry description:

Access	Read only
PDO mapping	No
Value range	0x3ED
Default value	0x3ED

- Data description, if bit is set – the mode is supported:

31 10	9	8	7	6	5	4	3	2	1	0
Reserve	Cst	csv	csp	ip	hm	Reserve	tq	pv	Reserve	pp

Chapter 9: Factors

9.1. General

Object	Definition
0x 608F	Position encoder resolution
0x 6090	Velocity encoder resolution
0x 6091	Gear Ratio
0x 6092	Feed Constant
0x 6093	Position factor of DS402
0x 6094	Velocity encoder factor of DS402
0x 6096	Velocity factor
0x 6097	Acceleration factor

Physical dimensions and sizes need to be converted into the device internal units, requiring a number of different factors. This chapter describes how these factors influence the system, how they are calculated and which data is needed to build them.

9.2. Relationship between Physical and Internal Units

The factors defined in the factor group determine a relationship between the Elmo drive internal units and the application physical units. Objects, which values are not dependent on the factor group have fixed units specified with the Objects. Figure 8-1 explains the scaling concept.

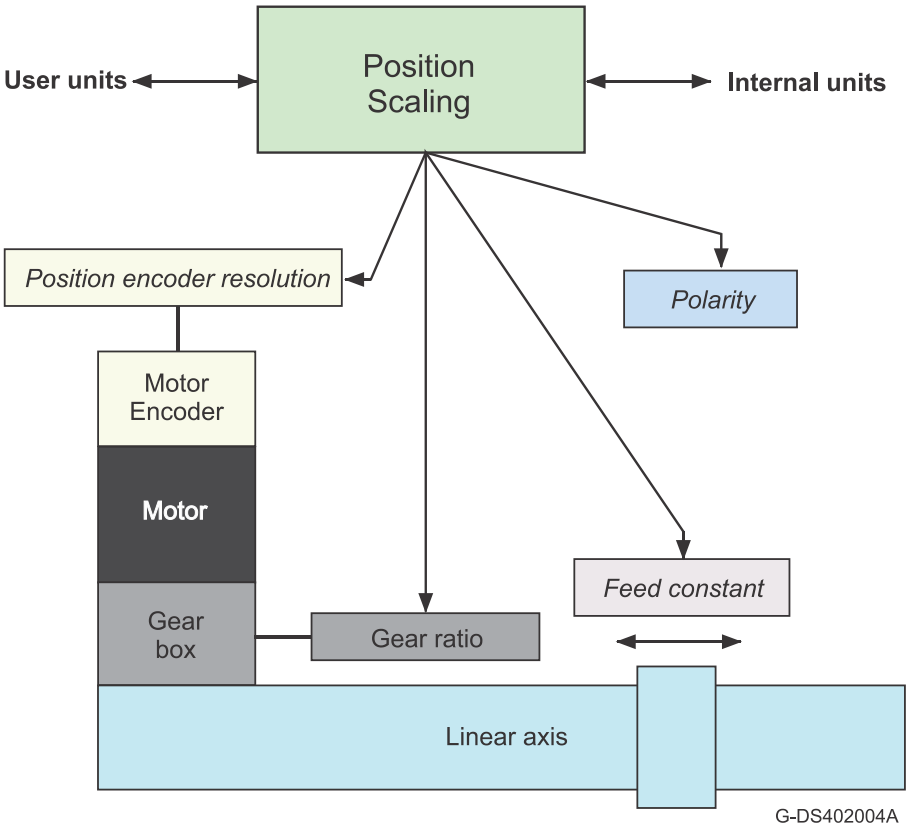


Figure 8-1: Position Scale concept

9.3. Position Units

Position value in user units can be calculated from position in internal units by the formula:

$$PosUU = \frac{PosIU * FC}{PosEncRes * G}$$

Where:

PosUU	Position value in user units, **Object 0x6064**
PosIU	Position in internal units (encoder counts), **Object 0x6063**
FC	Feed constant, **Object 0x6092**
PosEncRes	Position encoder resolution, **Object 0x608F**
G	Gear Ratio, **Object 0x6091**

Position in internal units can be calculated from position value in user units by the formula:

$$PosIU = \frac{PosUU * PosEncRes * G}{FC}$$

9.4. Velocity Units

Velocity value in user units can be calculated from velocity in internal units by the formula:

$$VelUU = \frac{VelIU * FC}{VelEncRes * G} * \text{VFac}$$

Where:

VelUU	Velocity value in user units, **Object 0x606C**
VelIU	Velocity in internal units (encoder counts/sec), **Object 0x6069**
VelEncRes	Velocity encoder resolution, **Object 0x6090**
VelFac	Velocity factor, **Object 0x6096**, =0x6096.1/0x6096.2

Velocity in internal units (encoder counts/sec) can be derived from the same position encoder, when sensor selection code , **Object 0x606A** is zero or from separate encoder, when sensor selection code , **Object 0x606A** is 1.

9.5. Acceleration Units

Acceleration value in user units can be calculated from velocity in user units by the formula:

$$AccUU = \frac{VelUU}{Sec} * \text{AccFac}$$

Where:

AccUU	Acceleration value in user units
AccFac	Acceleration factor, **Object 0x6097**, =0x6097.1/0x6097.2

9.6. Jerk Units

Jerk value in user units can be calculated from acceleration in user units by the formula:

$$JerkUU = \frac{AccUU}{Sec}$$

9.7. Functions and Limits

- Factors cannot be set while the drive is in OPERATION ENABLED state.
- Divisors cannot be set to 0. An abort message with abort code 0609 0030 is transmitted.
- Values are truncated to the nearest integer.

9.8. Object 0x607E: Polarity

The Polarity object allows the master to inverse the direction of the motion in a single command.

The Polarity can be set to either Position or Velocity reference. By setting the relevant bit (according to table below) the motion is reversed without the need to reverse the reference command.

Bit 6 of the Polarity Object is relevant in Profile Velocity mode and Cyclic Synchronous Velocity mode.

Bit 7 of the Polarity Object is relevant in Profile Position mode and in Cyclic Synchronous Position mode.

- Data Description:

7	6	5..... 0
Position polarity value	Velocity polarity value	Reserved, is set to 0

- Object description:

Attributes	0x607E
Name	Polarity
Object code	VAR
Data type	UNSIGNED8
Category	Optional

- Entry description:

Access	Read/write
PDO mapping	Yes
Value range	0x00, 0x40, 0x80, 0xc0
Default value	0

- Description values:

Value	Description
0	Multiply by 1
1	Multiply by -1

9.9. Object 0x608F: Position encoder resolution

This Object defines the ratio of encoder increments per motor revolution:

$$position\_encoder\_resolution = \frac{encoder\_increments}{motor\_revolutions}$$

- Object description:

Attributes	0x608F
Name	Position encoder resolution
Object code	ARRAY
Data type	UNSIGNED32
Category	Optional

- Entry description:

Sub-index	0
Description	Number of entries
Entry category	Mandatory
Access	Read only
PDO mapping	No
Value range	2
Default value	2

Sub-index	1
Description	Encoder increments
Entry category	Optional
Access	Read/Write
PDO mapping	No
Value range	1…0x7FFFFFFF
Default value	1

Sub-index	2
Description	Motor revolutions
Entry category	Optional
Access	Read / Write
PDO mapping	No
Value range	1…0x7FFFFFFF
Default value	1

9.10. Object 0x6090: Velocity encoder resolution

This Object defines the ratio of encoder increments/second per motor revolutions/ second.

$$velocity\_encoder\_resolution = \frac{encoder\_increments/\sec}{motor\_revolutions/\sec}$$

- Object description:

Attributes	0x6090
Name	Velocity encoder resolution
Object code	ARRAY
Data type	UNSIGNED32
Category	Optional

- Entry description:

Sub-index	0
Description	Number of entries
Entry category	Mandatory
Access	Read only
PDO mapping	No
Value range	2
Default value	2

Sub-index	1
Description	Encoder increments per second
Entry category	Mandatory
Access	Read/write
PDO mapping	No
Value range	1…0x7FFFFFFF
Default value	1

Sub-index	2
Description	Motor revolutions per second
Entry category	Mandatory
Access	Read/write
PDO mapping	No
Value range	1…0x7FFFFFFF
Default value	1

9.11. Object 0x6091: Gear Ratio

This Object defines the ratio of motor shaft revolutions per driving shaft revolutions:

$$Gear\ Ratio = \frac{Motor\ Shaft\ Revolutions}{Driving\ Shaft\ Revolutions}$$

- Object description:

Attributes	0x6091
Name	Gear Ratio
Object code	ARRAY
Data type	UNSIGNED32
Category	Optional

- Entry description:

Sub-index	0
Description	Number of entries
Entry category	Mandatory
Access	Read only
PDO mapping	No
Value range	2
Default value	2

Sub-index	1
Description	Motor shaft revolutions
Entry category	Mandatory
Access	Read/write
PDO mapping	No
Value range	1…0x7FFFFFFF
Default value	1

Sub-index	**2**
Description	Driving shaft revolutions
Entry category	Mandatory
Access	Read/write
PDO mapping	No
Value range	1…0x7FFFFFFF
Default value	1

9.12. Object 0x6092: Feed Constant

This Object defines the ratio of measured distance per driving shaft revolutions:

$$Feed\ Constant = \frac{Feed}{Driving\ Shaft\ Revolutions}$$

- Object description:

Attributes	0x6092
Name	Feed Constant
Object code	ARRAY
Data type	UNSIGNED32
Category	Optional

- Entry description:

Sub-index	0
Description	Number of entries
Entry category	Mandatory
Access	Read only
PDO mapping	No
Value range	2
Default value	2

Sub-index	1
Description	Feed
Entry category	Mandatory
Access	Read/write
PDO mapping	No
Value range	1…0x7FFFFFFF
Default value	1

Sub-index	**2**
Description	Driving shaft revolutions
Entry category	Mandatory
Access	Read/write
PDO mapping	No
Value range	1…0x7FFFFFFF
Default value	1

9.13. Object 0x6093: Position factor of DS-402

This Object converts the desired position (in position units) into the internal format (in increments).

The Elmo drive does not use this Object; it is available for user convenience.

The Object is not checked for value and consistency. In addition, this Object is non-volatile.

- Object description:

Attributes	0x6093
Name	Position factor of DS402
Object code	VAR
Data type	UNSIGNED32
Category	Optional

- Entry description:

Access	Read only
PDO mapping	No
Value range	0…(2^{32})-1
Default value	0

9.14. Object 0x6094: Velocity encoder factor of DS-402

This Object converts the desired velocity (in velocity units) into the internal format (in increments/second).

The Elmo drive does not use this Object; it is available for user convenience.

The Object is not checked for value and consistency. In addition, this Object is non-volatile.

- Object description:

Attributes	0x6094
Name	Velocity encoder factor of DS402
Object code	VAR
Data type	UNSIGNED32
Category	Optional

- Entry description:

Access	Read only
PDO mapping	No
Value range	0…(2^{32})-1
Default value	0

9.15. Object 0x6096: Velocity factor

This Object is used to define the relationship between the velocity encoder data and velocity in user units, because they are based on different dimensions.

- Object description:

Attributes	0x6096
Name	Velocity factor
Object code	ARRAY
Data type	UNSIGNED32
Category	Optional

- Entry description:

Sub-index	0
Description	Number of entries
Entry category	Mandatory
Access	Read only
PDO mapping	No
Value range	2
Default value	2

Sub-index	1
Description	Numerator
Entry category	Mandatory
Access	Read/write
PDO mapping	No
Value range	1…0x7FFFFFFF
Default value	1

Sub-index	**2**
Description	Divisor
Entry category	Mandatory
Access	Read/write
PDO mapping	No
Value range	1…0x7FFFFFFF
Default value	1

9.16. Object 0x6097: Acceleration factor

This Object converts the acceleration (in acceleration units/second^2) into the internal format (in increments/second^2).

- Object description:

Attributes	0x6097
Name	Acceleration factor
Object code	ARRAY
Data type	UNSIGNED32
Category	Optional

- Entry description:

Sub-index	0
Description	Number of entries
Entry category	Mandatory
Access	Read only
PDO mapping	No
Value range	2
Default value	2

Sub-index	1
Description	Numerator
Entry category	Mandatory
Access	Read/write
PDO mapping	No
Value range	1…0x7FFFFFFF
Default value	1

Sub-index	2
Description	Divisor
Entry category	Mandatory
Access	Read/write
PDO mapping	No
Value range	1…0x7FFFFFFF
Default value	1

Chapter 10: Homing

10.1. General

Object	Definition
0x607C	Home offset
0x6098	Homing method
0x6099	Homing speeds
0x609A	Homing acceleration
0x60E3	Supported homing methods

10.2. General Information

This chapter describes the method by which a drive seeks the home position (also called the datum, reference point or zero point). Homing can be performed using limit switches at the ends of travel or a home switch (zero point switch) in mid-travel; most of the methods also use the index (zero) pulse typically from an incremental encoder.

10.3. Inputs and Outputs of Homing mode

The user can specify the speeds, acceleration and method of homing. An additional Object, home offset, is used to displace zero in the user's coordinate system from the home position. There are two homing speeds: in a typical cycle the faster speed is used to find the home switch and the slower speed is used to find the index pulse. The Homing mode function is presented on Figure 10-1.

There is no output data except for those bits in the *Statusword* that return the status or result of the homing process and the demand to the position control loops.

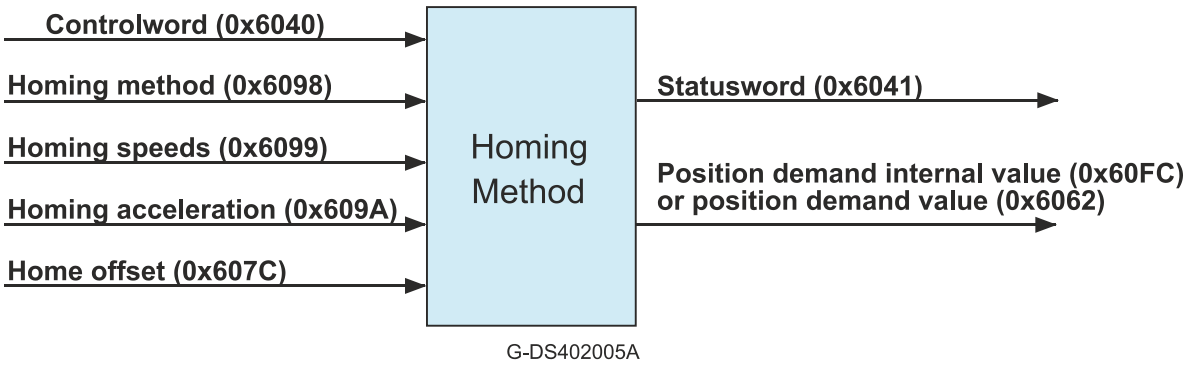


Figure 10-1: Homing mode function

The Homing method determines:

- Homing signal: positive limit switch, negative limit switch, home switch, index pulse
- Direction of actuation
- Using of index pulse
- Event that is considered as *home attained*

The homing mode is controlled by the bits of the *Controlword* and *Statusword*.

10.3.1. Homing Mode *Controlword*

Bit	Function
0	Switch On
1	Enable Voltage
2	Quick Stop
3	Enable Operation
4	Home operation start
5..6	Reserved
7	Fault Reset
8	Halt
9,10	Reserved
11..15	Reserved

Name	Value	Description
Homing operation start	0	Homing mode inactive
	0»1	Start homing mode
	1	Homing mode active
	1»0	Interrupt homing mode
Halt	0	Execute the instruction of bit 4
	1	Stop axle with homing deceleration

If homing is interrupted by setting bit 4 from *1* to *0*, the movement of the motor is stopped with home deceleration and a homing error indicated in the status word.

If a Halt occurs, the drive stops the homing method and halts the motor according to Object $609A_h$. When this bit is set to *0* and bit 4 remains *1*, the home method begins again.

10.3.2. Homing Mode *Statusword*

Bit	Function
0	Ready to switch on
1	Switched on
2	Operation enabled
3	Fault
4	Voltage enabled
5	Quick stop
6	Switch on disabled
7	Warning
8	Manufacturer specific
9	Remote
10	Target reached
11	Internal limit active
12	Homing attained
13	Homing error
14…15	Reserved

The following table describes the homing operation as reflected by the influence of bits 10, 12 and 13.

Bit13	Bit 12	Bit 10	Definition
0	0	0	Homing procedure is in progress
0	0	1	Homing procedure is interrupted or not started
0	1	0	Homing is attained, but target is not reached
0	1	1	Homing procedure is completed successfully
1	0	0	Homing error occurred, velocity is not 0
1	0	1	Homing error occurred, velocity is 0
1	1	X	Reserved

Table 10-1 Definition of bit 10, bit 12, and bit 13

There is no impact of software limits on the homing procedure. FLS, RLS have no impact if they are not a part of the homing procedure.

10.4. Object 0x607C: Home offset

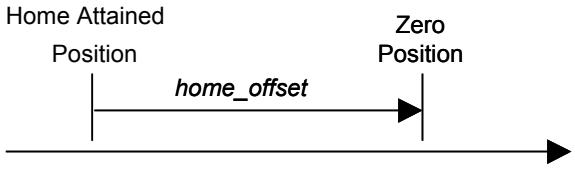
This Object is the difference between the zero position for the application and the machine home position (found during homing), measured in position units, or

Home Offset = Zero Position – Home Attained Position

So, zero position is calculated:

Zero position = Home Attained Position + Home Offset

During homing, the machine home position is found. Once homing is completed, the zero position is offset from the home position by adding the *home offset* to the home position. All subsequent absolute moves are taken relative to this new zero position, as illustrated in the following diagram.



For example:

Before homing, the zero position is 1000. Homing offset is set by the user equals 100. The Homing method defines the index pulse as *home attained* event. When moving and this event occurred, the position was 900. The Gold drive changed actual position in the index pulse from 900 to (-)home offset, setting it -100. As a result of this, the zero position is changed from 1000 to 0.

- Object description:

Attributes	0x607C
Name	Home offset
Object code	VAR
Data type	INTEGER32
Category	Optional

- Entry description:

Access	Read/write
PDO mapping	No
Value range	-2^{31}...$(2^{31})-1$
Default value	0

10.5. Object 0x6098: Homing Method

This Object determines the method used during homing.

- Object description:

Attributes	0x6098
Name	Homing method
Object code	VAR
Data type	INTEGER8
Category	Mandatory

- Entry description:

Access	Read/write
PDO mapping	No
Value range	INTEGER8
Default value	1

- Data description:

Value	Description
-2, -1	Manufacturer specific
0	Not Supported
1…14, 17…30, 33…35	Refer to the functional description
36…127	Reserved

10.6. Object 0x6099: Homing Speeds

This Object defines the speeds used during homing, in velocity units. The value is normalized to increments by *velocity code factor*. Typically, a high speed is used when searching for a home switch and the slow speed is used when searching for the index.

- Object description:

Attributes	0x6099
Name	Homing speeds
Object code	ARRAY
Data type	UNSIGNED32
Category	Mandatory

- Entry description:

Sub-index	0
Description	Number of entries
Entry category	Mandatory
Access	Read only
PDO mapping	No
Value range	2
Default value	2

Sub-index	1
Description	Speed during search for switch High homing speed
Entry category	Mandatory
Access	Read/write
PDO mapping	No
Value range	0…2147483647
Default value	1000

Sub-index	**2**
Description	Speed during search for zero Low Homing Speed
Entry category	Mandatory
Access	Read/write
PDO mapping	No
Value range	0…2147483647
Default value	1000

The speed is submitted to the maximum speed limit given by the user during setup. Otherwise, an abort message with abort code 0609 0030, *Value range of parameter exceeded* is activated.

10.7. Object 0x609A: Homing Acceleration

This Object establishes the acceleration to be used for all accelerations and decelerations with the standard homing modes, and is given in acceleration units.

- Object description:

Attributes	0x609A
Name	Homing acceleration
Object code	VAR
Data type	UNSIGNED32
Category	Optional

- Entry description:

Access	Read/write
PDO mapping	No
Value range	0… 2147483647
Default value	No

Home deceleration is performed according to the same value.

10.8. Object 0x60E3: Supported Homing Methods

This Object indicates supported homing methods.

- Object description:

Attributes	0x60E3
Name	Supported homing methods
Object code	ARRAY
Data type	INTEGER8
Category	Optional

- Entry description:

Sub-index	0
Description	Number of entries
Entry category	Mandatory
Access	RO
PDO mapping	No
Value range	33
Default value	33

Sub-index	1…33
Description	1…33 supported homing method
Entry category	Mandatory
Access	RO
PDO mapping	No
Value range	No
Default value	No

10.9. DS-402 Homing Description and Methods

The following sub-sections describe the details of how each homing mode functions. The Elmo drives support each of these methods.

Various homing positions are illustrated in the diagrams that follow (Figure 10-2 - Figure 10-35). A circled number indicates the code for selecting the homing position. The direction of movement is also indicated.

Additional homing methods are available with other modes of the Elmo drives, such as the binary interpreter or the user program. Limit switches must be previously defined during the drive setup (using the **IL[N]** command).

In the homing sequence diagrams, the encoder count increases as the axle position moves to the right. In other words, the left is the minimum position and the right is the maximum position.

Note: **In the drawings below, the thick section of the motion line denotes high speed representing 0x6069.1 and the thin section of the motion line denotes slow speed defined by 0x 6099.2**

10.9.1. Error Situations

Error situations are events in which the drive failed to perform the homing or was interrupted from performing the homing when, for example, a limit switch is detected and the limit is not part of the homing method. In cases where the limit is known in advance, e.g. when the home speed is higher than the speed limit, an abort message is executed. Where a fault occurred during the operation of the home procedure, the drive goes into a fault state and an emergency message is transmitted if not masked. The Homing error is also set when bit 4 of the Control word is set from 1 to 0 during the homing procedure.

10.9.2. Method 1: Homing on RLS and Index Pulse

When using this method, the motor moves in the negative direction with a homing speed defined by 0x6099.1 as long as Reverse Limit Switch (RLS) remains inactive (low in Figure 10-2). When the active state of RLS is detected, the motor changes direction and speed, and moves in the positive direction with a homing speed defined by 0x6099.2 until the first index pulse is detected after RLS attains the inactive state.

When receiving this event, the drive sets *home attained* bit in *Statusword* and decelerates with a homing acceleration value (0x609A). The target reached bit in *Statusword* is set when the motor completely stops.

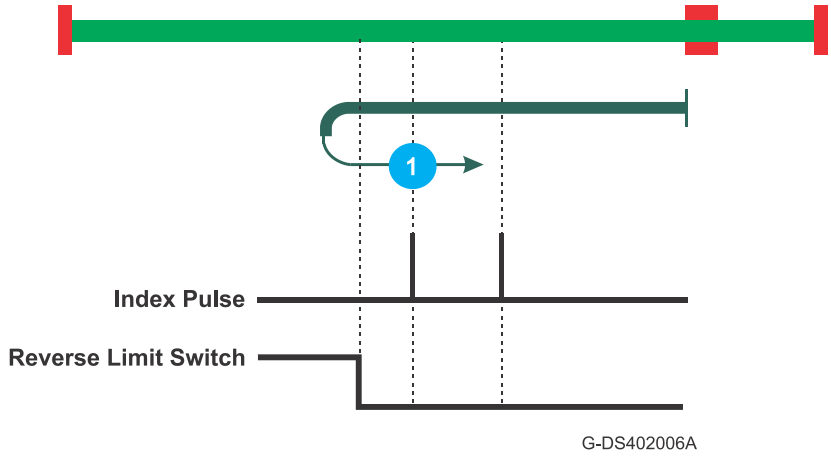


Figure 10-2: Homing on the negative limit switch and index pulse. Method 1

10.9.3. Method 2: Homing on FLS and Index Pulse

When using this method, the motor moves in the positive direction with a homing speed defined by 0x6099.1 as long as Forward Limit Switch (FLS) remains inactive (low in Figure 10-3). When the active state of FLS is detected, the motor changes direction and speed and moves in the negative direction with a homing speed defined by 0x6099.2 until the first index pulse is detected after FLS attains the inactive state.

When receiving this event, the drive sets *home attained* bit in *Statusword* and decelerates with a homing acceleration value (0x609A). The target reached bit in *Statusword* is set when the motor completely stops.

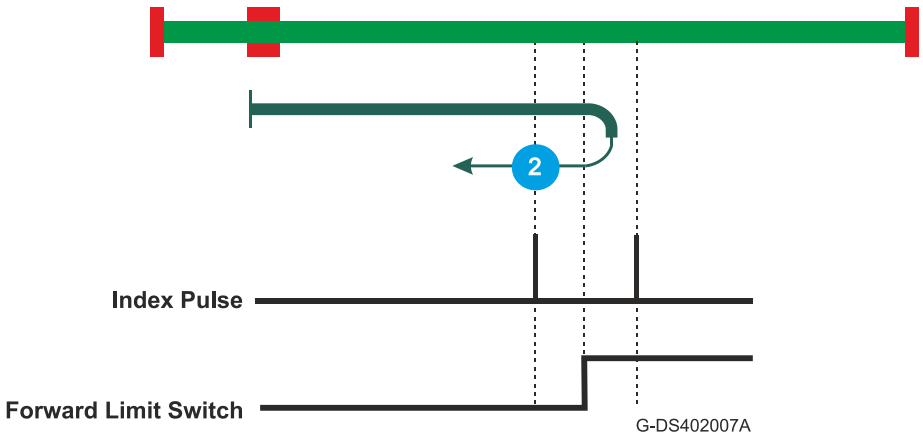


Figure 10-3: Homing on the positive limit switch and index pulse. Method 2

10.9.4. Method 3: Homing on Positive Home Switch and Index Pulse

When using method 3, the initial direction of movement depends on the state of the home switch when start homing.

If homing started and the home switch is inactive, the motor moves in the positive direction with a homing speed defined by 0x6099.1 as long as home switch remains inactive. When the active state of home switch is detected, the motor changes direction and speed and moves in the negative direction with a homing speed defined by 0x6099.2 until home switch attains the inactive state. Then the first index pulse will be detected. This is considered as home attained event.

If homing started and the home switch is active, the motor moves in the negative direction with a homing speed defined by 0x6099.2 until home switch attains inactive state. Then the first index pulse will be detected. This is considered as home attained event.

When receiving the home attained event, the drive sets the *home attained* bit in *Statusword* and decelerates with a homing acceleration value (0x609A). The target reached bit in *Statusword* is set when the motor completely stops.

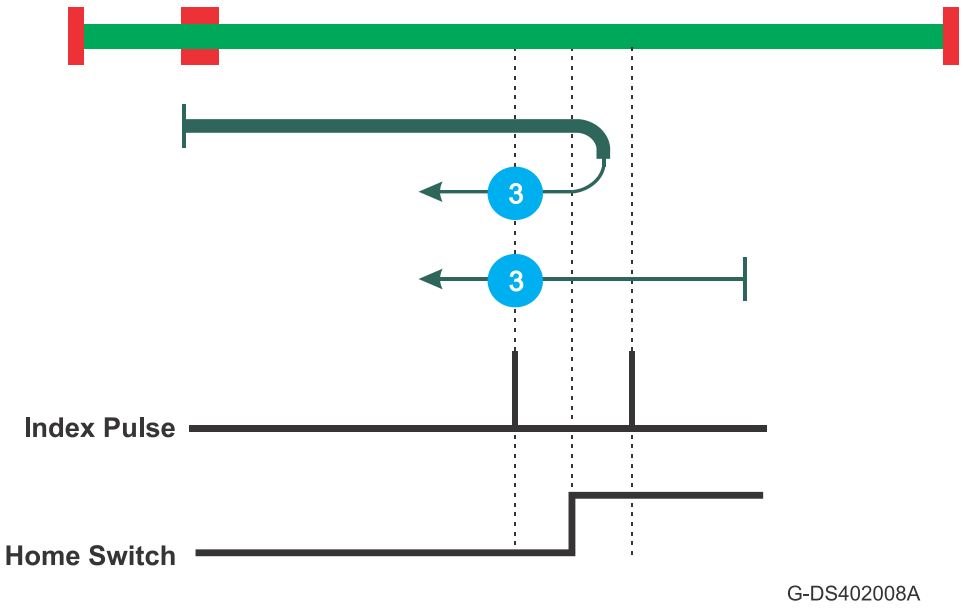


Figure 10-4: Homing on the positive home switch and index pulse. Method 3

10.9.5. Method 4: Forward Homing on Positive Home Switch and Index Pulse

When using method 4, the initial direction of movement depends on the state of the home switch when start homing.

If homing is started and home switch is active, the motor moves in the negative direction with a homing speed defined by 0x6099.1 as long as home switch remains active. When inactive state of home switch is detected, the motor changes direction and speed and moves in the positive direction with a homing speed defined by 0x6099.2 until home switch attains active state. Then the first index pulse will be detected. This is considered as home attained event.

If homing starts and the home switch is inactive, the motor moves in the positive direction with a homing speed defined by 0x6099.2 until the home switch attains active state. Then the first index pulse will be detected. This is considered as home attained event.

When receiving the home attained event, the drive sets *home attained* bit in *Statusword* and decelerates with homing acceleration value (0x609A). Target reached bit in *Statusword* is set when the motor completely stopped.

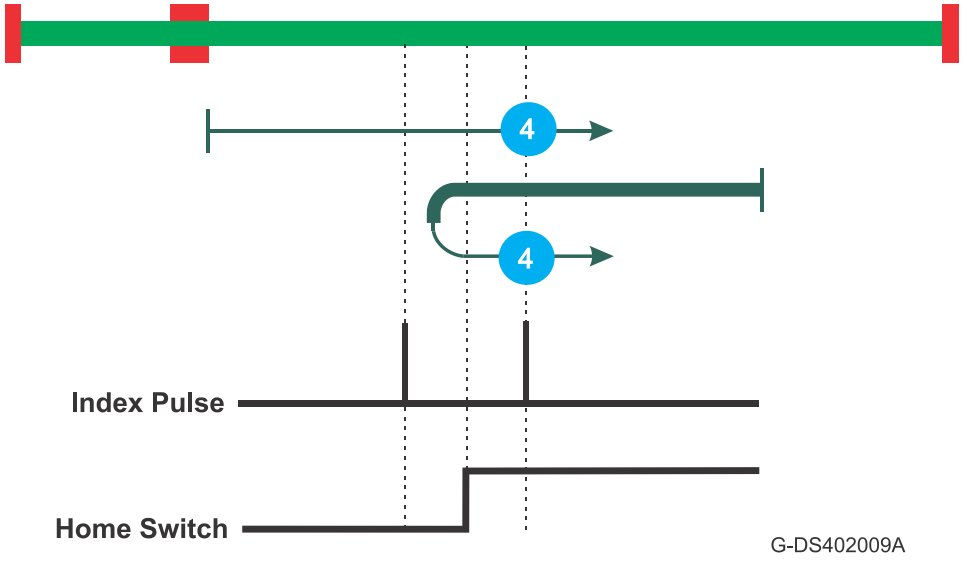


Figure 10-5: Homing on the positive home switch and index pulse. Method 4

10.9.6. Method 5: Forward Homing on Negative Home Switch and Index Pulse

When using method 5, the initial direction of movement depends on the state of the home switch when start homing. If homing starts and the home switch is inactive, the motor moves in the negative direction with a homing speed defined by 0x6099.1 as long as home switch remains inactive.

When active state of home switch is detected, the motor changes direction and speed, and moves in the positive direction with a homing speed defined by 0x6099.2 until home switch achieves inactive state. Then the first index pulse will be detected. This is considered as home attained event.

If homing starts and the home switch is active, the motor moves in the positive direction with a homing speed defined by 0x6099.2 until the home switch attains inactive state. Then the first index pulse will be detected. This is considered as home attained event.

When receiving home attained event, the drive sets *home attained bit* in *Statusword* and decelerates with homing acceleration value (0x609A). The target reached bit in *Statusword* is set when the motor completely stops.

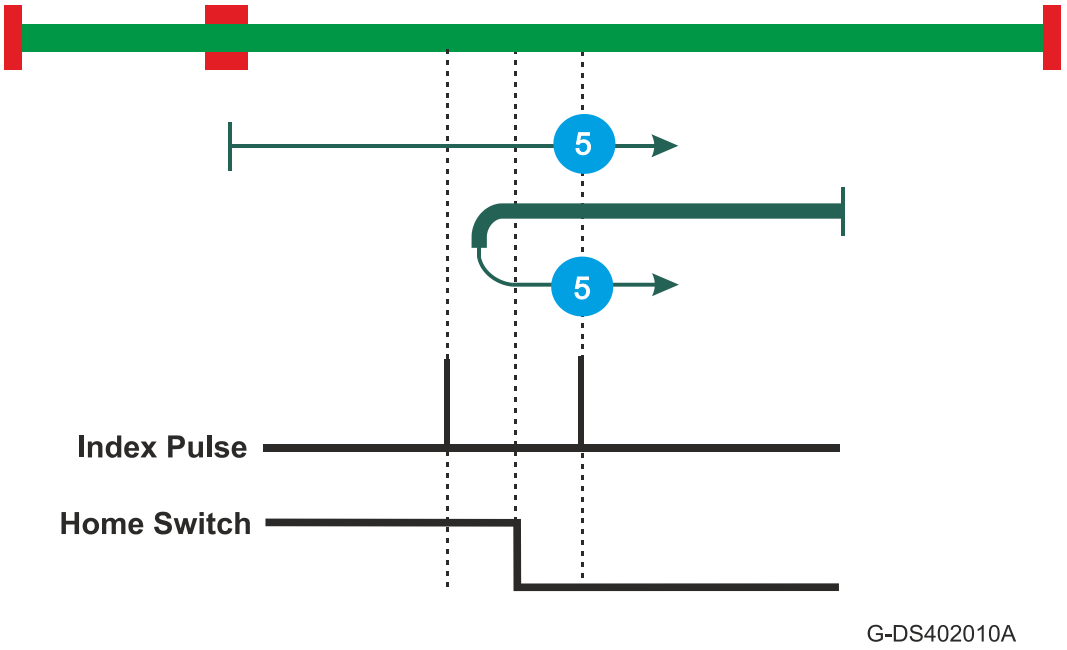


Figure 10-6: Homing on the negative home switch and index pulse. Method 5

10.9.7. Method 6: Reverse Homing on Negative Home Switch And Index Pulse

When using method 6, the initial direction of movement depends on the state of the home switch when start homing.

If homing starts and the home switch is active, the motor moves in the positive direction with a homing speed defined by 0x6099.1 as long as the home switch remains active. When the inactive state of home switch is detected, the motor changes direction and speed, and moves in the negative direction with a homing speed defined by 0x6099.2 until home switch attains active state. Then the first index pulse will be detected. This is considered as home attained event.

If homing starts and the home switch is inactive, the motor moves in the negative direction with a homing speed defined by 0x6099.2 until the home switch attains the active state. Then the first index pulse will be detected. This is considered as home attained event.

When receiving home attained event the drive sets *home attained bit* in *Statusword* and decelerates with a homing acceleration value (0x609A). The target reached bit in *Statusword* is set when the motor completely stops.

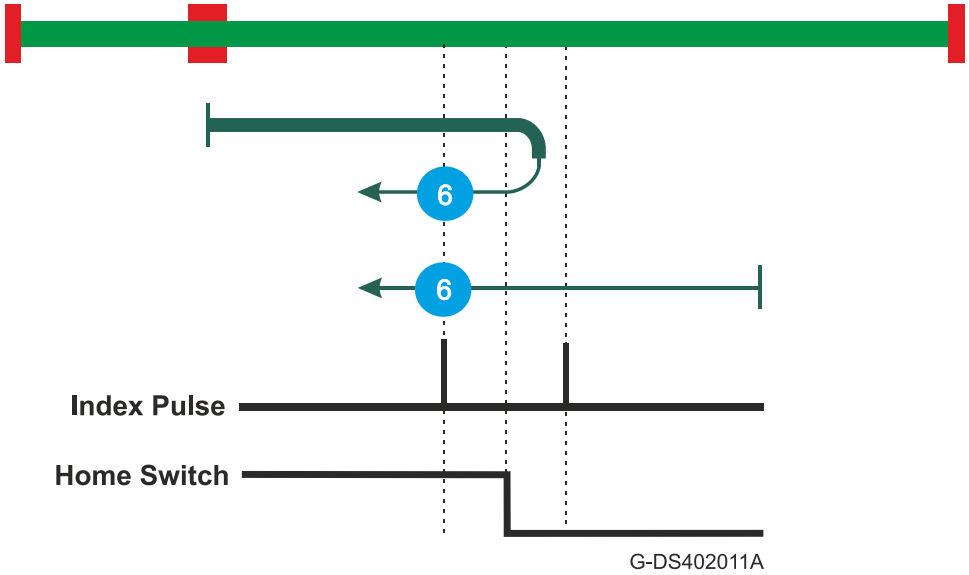


Figure 10-7: Homing on the negative home switch and index pulse. Method 6

10.9.8. Method 7: Reverse Homing on Home Switch/FLS and Index Pulse

When using method 7, the initial direction of movement depends on the state of the home switch when start homing. If homing starts and the home switch is inactive, the motor moves in the positive direction with a homing speed defined by 0x6099.1 until one of two possible events occur:

- Active state of home switch is detected. In this case the motor changes direction and speed and moves in the negative direction with a homing speed defined by 0x6099.2 until home switch inactive state is achieved. Then the first index pulse will be detected. This is considered as home attained event.
- Active state of FLS is detected. In this cases the motor changes direction and speed and moves in the negative direction with a speed defined by 0x6099.2 until the home switch changes state from inactive to active, and then from active to inactive state. Then the first index pulse will be detected. This is considered as home attained event.

If homing starts and the home switch is active, the motor moves in the negative direction with a homing speed defined by 0x6099.2 until home switch changes state from active to inactive. Then the first index pulse will be detected. This is considered as home attained event.

When receiving home attained event the drive sets *home attained bit* in *Statusword* and decelerates with a homing acceleration value (0x609A). The target reached bit in *Statusword* is set when the motor completely stops.

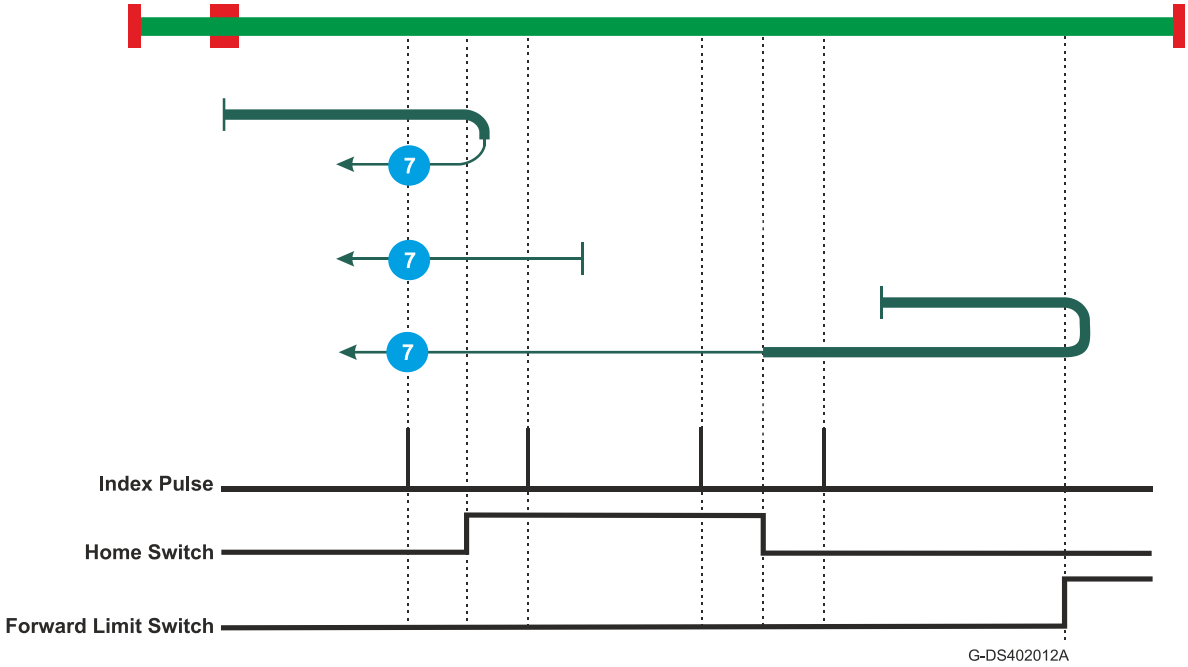


Figure 10-8: Homing on the home switch and index pulse — positive initial move. Method 7

10.9.9. Method 8: Forward Homing on Home Switch/FLS and Index Pulse

When using method 8, the initial direction of movement depends on the state of the home switch when start homing. If homing starts and the home switch is inactive, the motor moves in the positive direction with a homing speed defined by 0x6099.2 until one of two possible events occur:

- Active state of home switch is detected. In this case the motor continue moving in the positive direction with the same speed until the first index pulse is detected. This is considered as home attained event.
- Active state of FLS is detected. In this cases the motor changes direction and speed and moves in the negative direction with a speed defined by 0x6099.1 until home switch changes its state from inactive to active, and then from active to inactive state. When the change from active to inactive state is detected, the motor changes direction and speed defined by 0x6099.2 and moves in the positive direction until the home switch changes from inactive to active state. Then the first index pulse will be detected. This is considered as home attained event.

If homing starts and the home switch is active, the motor moves in the negative direction with a homing speed defined by 0x6099.1 until home switch changes state from active to inactive. When the change from active to inactive state is detected, the motor changes direction and speed defined by 0x6099.2 and moves in the positive direction until the home switch changes from inactive to active state. Then the first index pulse will be detected. This is considered as home attained event.

When receiving home attained event, the drive sets *home attained bit* in *Statusword* and decelerates with homing acceleration value (0x609A). The target reached bit in *Statusword* is set when the motor completely stops.

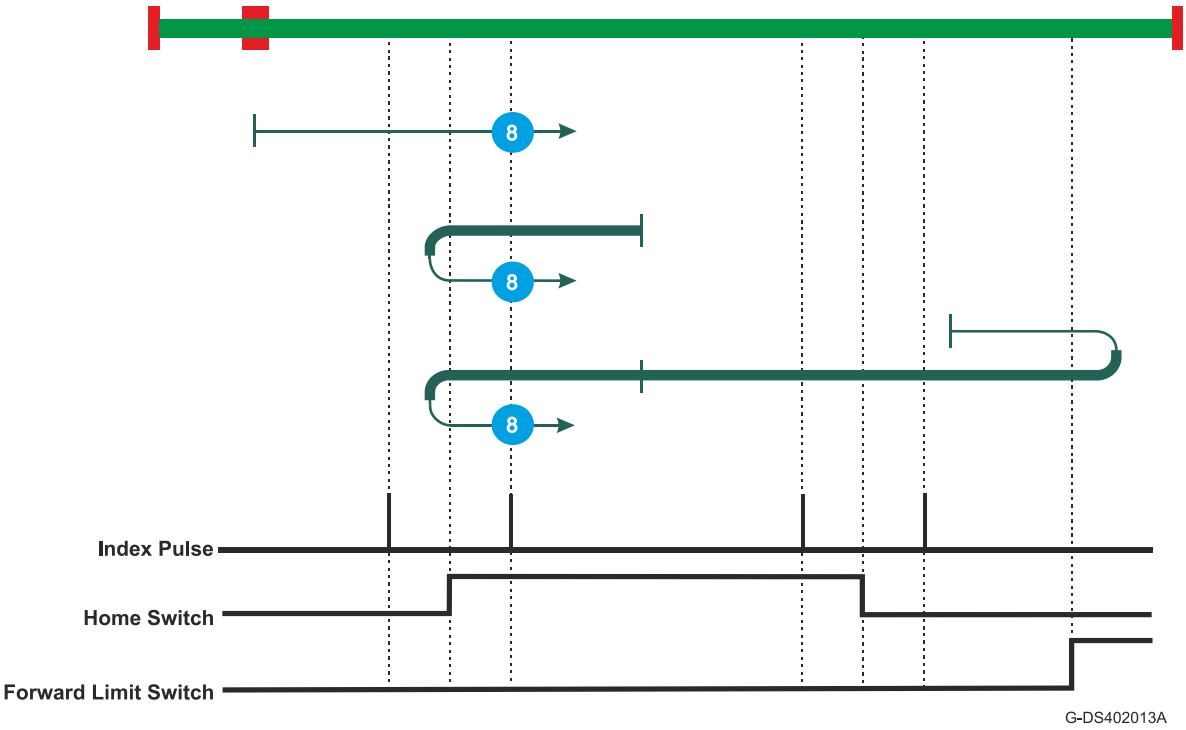


Figure 10-9: Homing on the home switch and index pulse — positive initial move. Method 8

10.9.10. Method 9: Reverse Homing on Positive Home Switch/FLS and Index Pulse

When using method 9, the initial direction of movement is always positive. If homing starts and the home switch is inactive, the motor moves in the positive direction with a homing speed defined by 0x6099.1. Until, one of two possible events occurs:

- Active state of home switch is detected. In this case the motor continues to move in the positive direction with the same speed as long as home switch remain active. When the home switch inactive state is detected, the motor changes direction and speed and moves in the negative direction with a speed defined by 0x6099.2 until the home switch attains active state. Then the first index pulse will be detected. This is considered as home attained event.
- Active state of FLS is detected. In this case the motor changes direction, and speed defined by 0x6099.2, and moves in the negative direction until the home switch changes state from inactive to active. Then the first index pulse will be detected. This is considered as home attained event.

If homing starts and the home switch is active, the motor moves in the positive direction with a homing speed defined by 0x6099.1 as long as home switch remains active. When the home switch inactive state is detected, the motor changes direction and speed and moves in the negative direction with a homing speed defined by 0x6099.2 until home switch attains the active state. Then the first index pulse will be detected. This is considered as home attained event.

When receiving event home attained, the drive sets *home attained bit* in *Statusword* and decelerates with a homing acceleration value (0x609A). The target reached bit in *Statusword* is set when the motor completely stops.

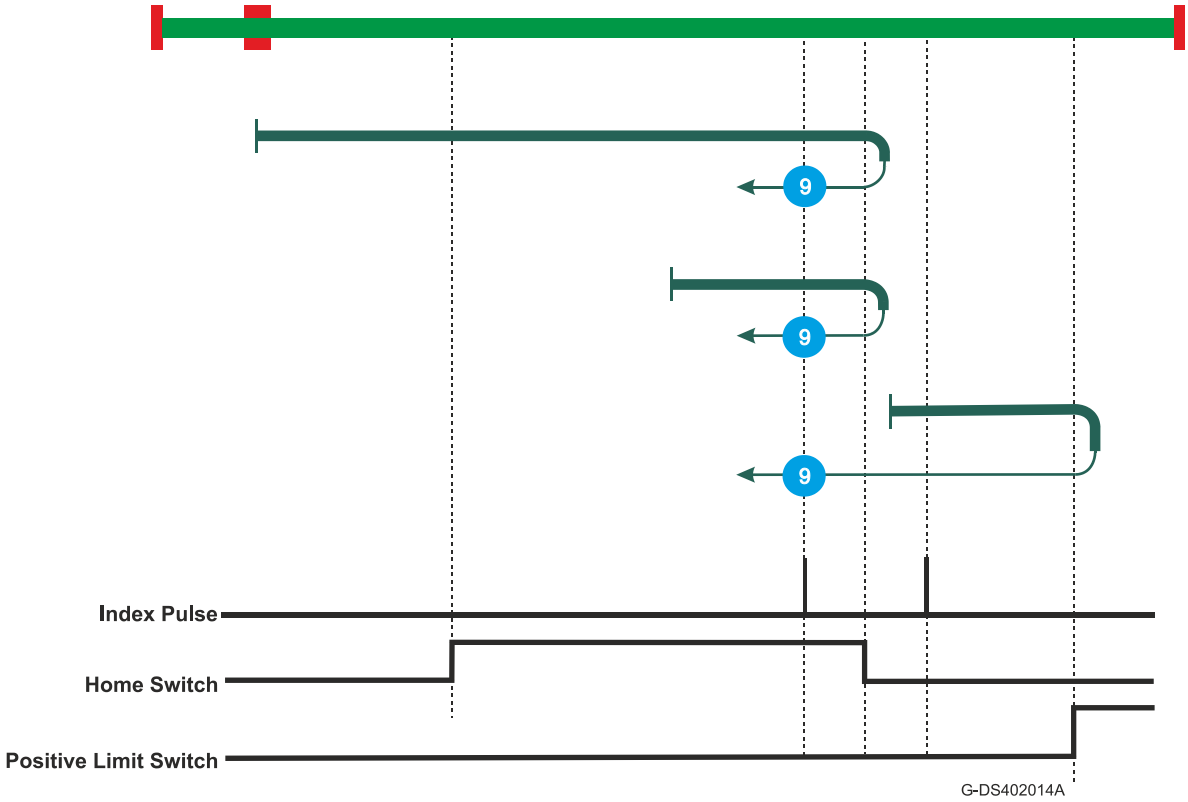


Figure 10-10: Homing on the home switch and index pulse — positive initial move. Method 9

10.9.11. Method 10: Forward Homing on Negative Home Switch/FLS and Index Pulse

When using method 10, the initial direction of movement is always positive. If homing starts and the home switch is inactive, the motor moves in the positive direction with a homing speed defined by 0x6099.1. until one of two possible events occurs:

- Active state of home switch is detected. In this case the motor changes speed to one defined by 0x6099.2 and continues to move in the positive direction until the home switch attains the inactive state. Then the first index pulse will be detected. This is considered as home attained event.
- Active state of FLS is detected. In this case the motor changes direction and moves in the negative direction. When the home switch changes state from inactive to active, the motor changes direction and speed to one defined by 0x6099.2 and moves in the positive direction until the home switch attains the inactive state. Then the first index pulse will be detected. This is considered as home attained event.

If homing starts and the home switch is active, the motor moves in the positive direction with homing speed defined by 0x6099.2 until home switch attains the inactive state. Then the first index pulse will be detected. This is considered as home attained event.

When receiving event home attained the drive sets *home attained bit* in *Statusword* and decelerates with homing acceleration value (0x609A). The target reached bit in *Statusword* is set when the motor completely stops.

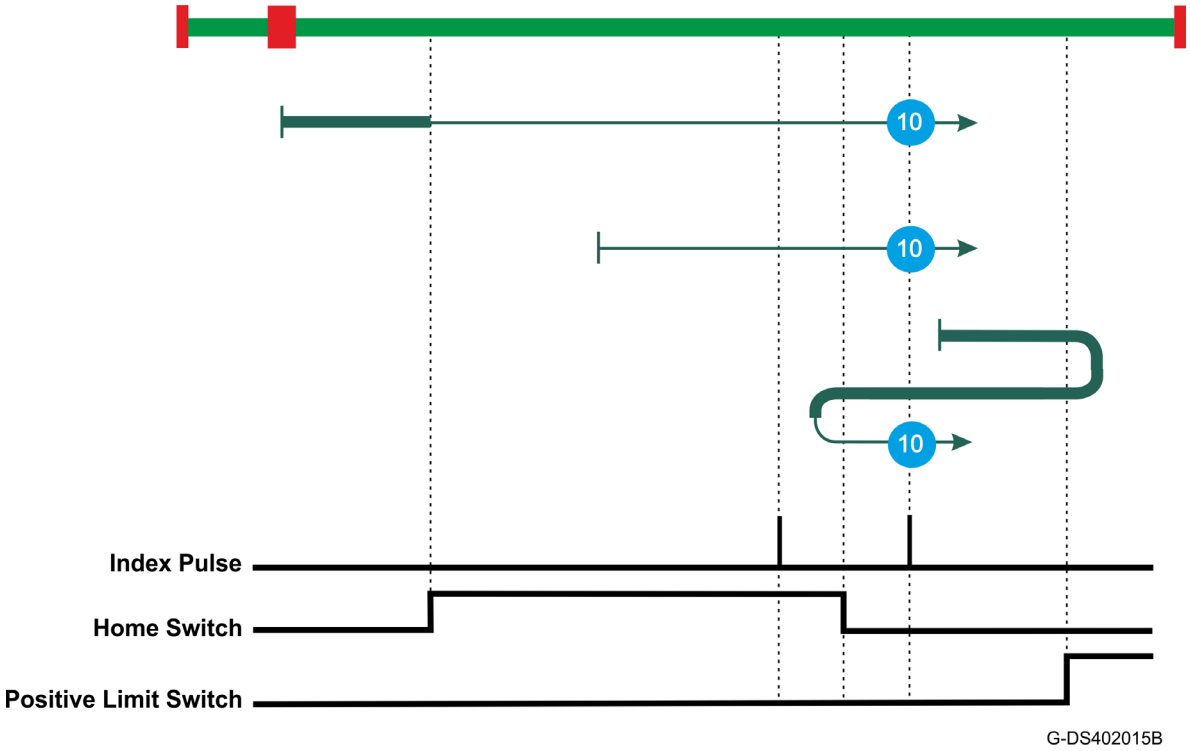


Figure 10-11: Homing on the home switch and index pulse — positive initial move. Method 10

10.9.12. Method 11: Forward Homing on Negative Home Switch/RLS and Index Pulse

When using method 11, the initial direction of movement depends on the state of the home switch when start homing. If homing starts and the home switch is inactive, the motor moves in the negative direction with a homing speed defined by 0x6099.1 until one of two possible events occurs:

- Active state of home switch is detected. In this case the motor stops and changes direction, and now accelerates to a speed defined by 0x6099.2 and continues to move in the positive direction until home switch attains inactive state and then the first index pulse is detected. This is considered as home attained event.
- Active state of RLS is detected. In this case the motor changes direction and moves in the positive direction. When the home switch changes its state from inactive to active, the motor changes speed to one defined by 0x6099.2 and moves in the positive direction until home switch attains inactive state. Then the first index pulse will be detected. This is considered as home attained event.

If homing starts and the home switch is active, the motor moves in the positive direction with a homing speed defined by 0x6099.2 until the home switch attains inactive state. Then the first index pulse will be detected. This is considered as home attained event.

When receiving event home attained the drive sets *home attained bit* in *Statusword* and decelerates with homing acceleration value (0x609A). The target reached bit in *Statusword* is set when the motor completely stops.

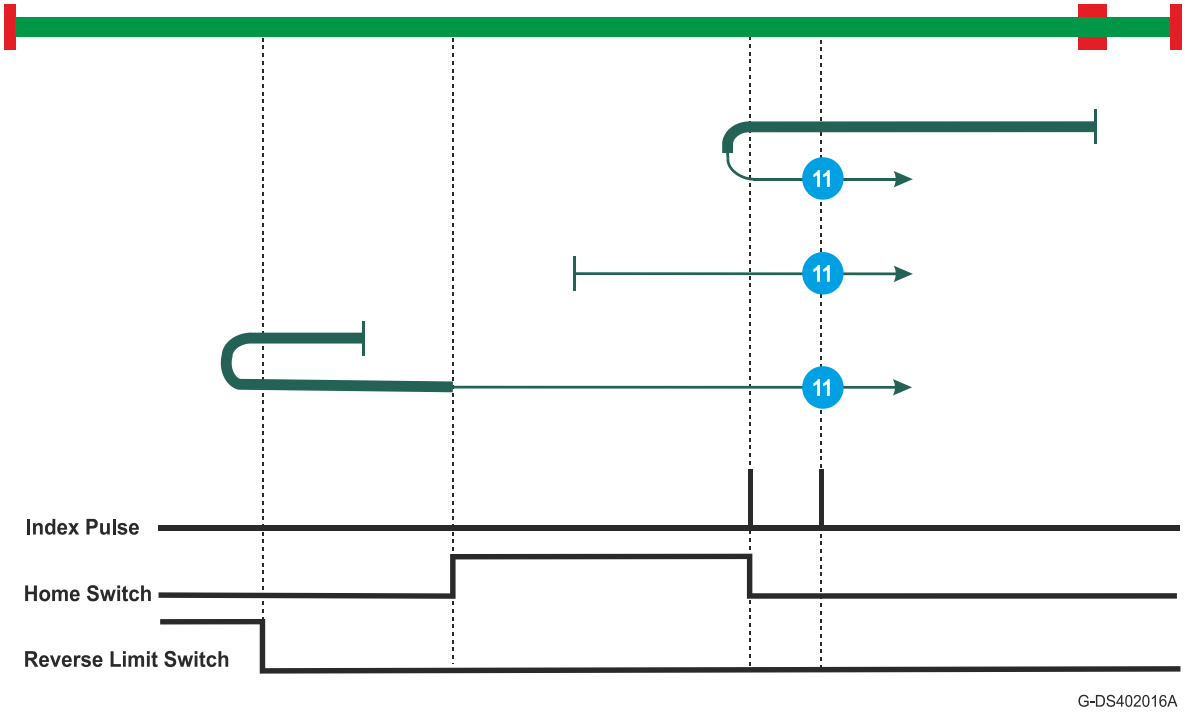


Figure 10-12: Homing on the home switch and index pulse — positive initial move. Method 11

10.9.13. Method 12: Reverse Homing on Positive Home Switch/RLS and Index Pulse

When using method 12, the initial direction of movement depends on the state of the home switch when start homing. If homing starts and the home switch is inactive, the motor moves in the negative direction with a homing speed defined by 0x6099.2 until one of two possible events occurs:

- Active state of home switch is detected. In this case the motor continues to move with the same speed and direction until the first index pulse is detected. This is considered as home attained event.
- Active state of RLS is detected. In this case the motor changes direction and speed to one defined by 0x6099.1 and moves in the positive direction until the home switch changes state from inactive to active and then from active to inactive. The motor then changes direction and speed to one defined by 0x6099.2 and moves in the negative direction until the home switch attains active state. Then the first index pulse is detected. This is considered as home attained event.

If homing starts and the home switch is active, the motor moves in the positive direction with a homing speed defined by 0x6099.1 until the home switch attains inactive state. Then, the motor changes direction and speed and moves in the negative direction with a speed defined by 0x6099.2 until the home switch attains active state. Then the first index pulse is detected. This is considered as home attained event.

When receiving event home attained the drive sets *home attained bit* in *Statusword* and decelerates with homing acceleration value (0x609A). The target reached bit in *Statusword* is set when the motor completely stops.

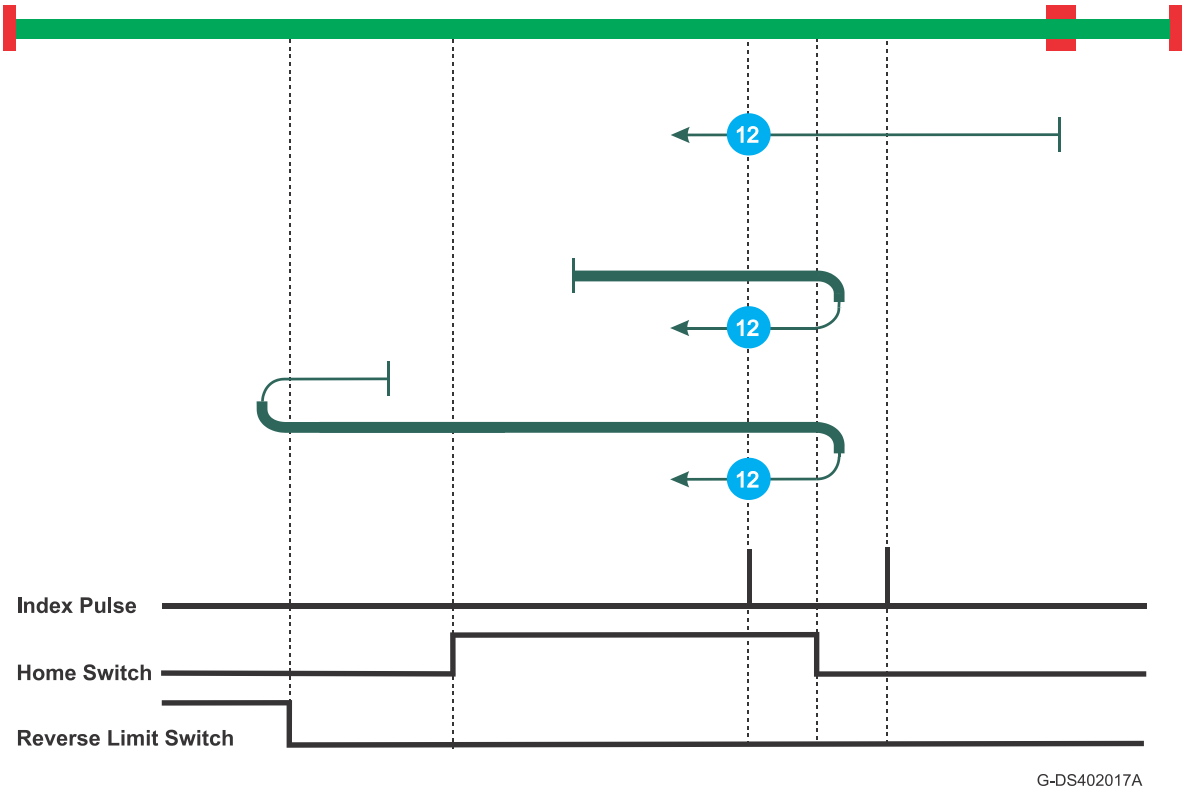


Figure 10-13: Homing on the home switch and index pulse —negative initial move. Method 12

10.9.14. Method 13: Forward Homing on Positive Home Switch/RLS and Index Pulse

When using method 13, the initial direction of movement is always negative. If homing starts and the home switch is inactive, the motor moves in the negative direction with a homing speed defined by 0x6099.1 until one of two possible events occurs:

- Active state of home switch is detected. In this case the motor continues to move with the same speed and direction until the home switch inactive state is detected. Then, the motor changes direction and speed to one defined by 0x6099.2 and moves in the positive direction until the home switch attains active state. Then the first index pulse is detected. This is considered as home attained event.
- Active state of RLS is detected. In this case the motor changes direction and speed to defined by 0x6099.2 and moves in the positive direction until the home switch attains active state. Then the first index pulse will be detected. This is considered as home attained event.

If homing starts and the home switch is active, the motor moves in the negative direction until the home switch attains inactive state. Then, the motor changes direction and speed and moves in the positive direction with a speed defined by 0x6099.2 until the home switch attains active state. Then the first index pulse is detected. This is considered as home attained event.

When receiving event home attained the drive sets *home attained bit* in *Statusword* and decelerates with homing acceleration value (0x609A). Target reached bit in *Statusword* is set when the motor completely stopped.

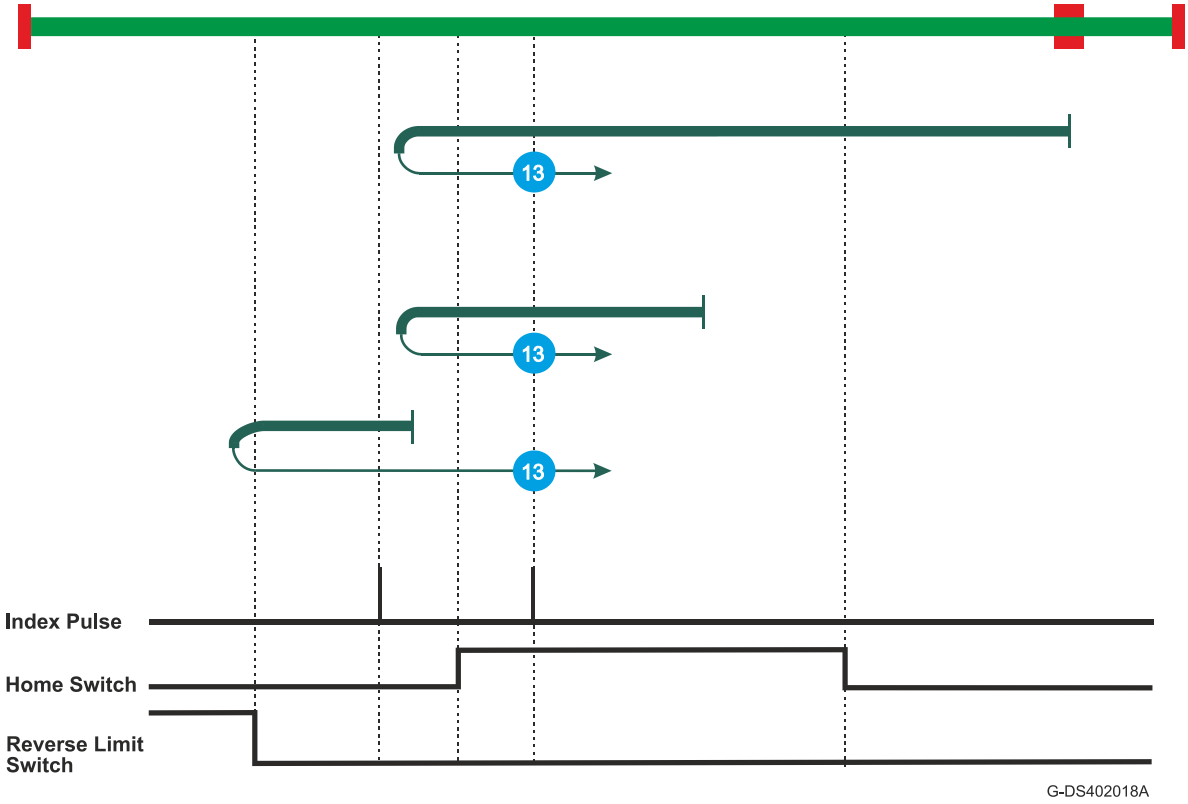


Figure 10-14: Homing on the home switch and index pulse —negative initial move. Method 13

10.9.15. Method 14: Reverse Homing on Negative Home Switch/RLS and Index Pulse

When using method 14, the initial direction of movement is always negative. If homing starts and the home switch is inactive, the motor moves in the negative direction with a homing speed defined by 0x6099.1. until one of two possible events occurs:

- Active state of home switch is detected. In this case the motor changes the speed to one defined by 0x6099.2 and continues to move in the negative direction until home switch attains inactive state, and then the first index pulse is detected. This is considered as home attained event.
- Active state of RLS is detected. In this case the motor changes direction and moves in the positive direction until the home switch changes state from inactive to active. Then, the motor changes direction and speed to one defined by 0x6099.2 and moves in the negative direction until home switch attains inactive state and then the first index pulse is detected. This is considered as home attained event.

If homing starts and the home switch is active, the motor moves in the negative direction with a homing speed defined by 0x6099.2 until the home switch attains inactive state, Then the first index pulse is detected. This is considered as home attained event.

When receiving event home attained the drive sets *home attained bit* in *Statusword* and decelerates with homing acceleration value (0x609A). The target reached bit in *Statusword* is set when the motor completely stops.

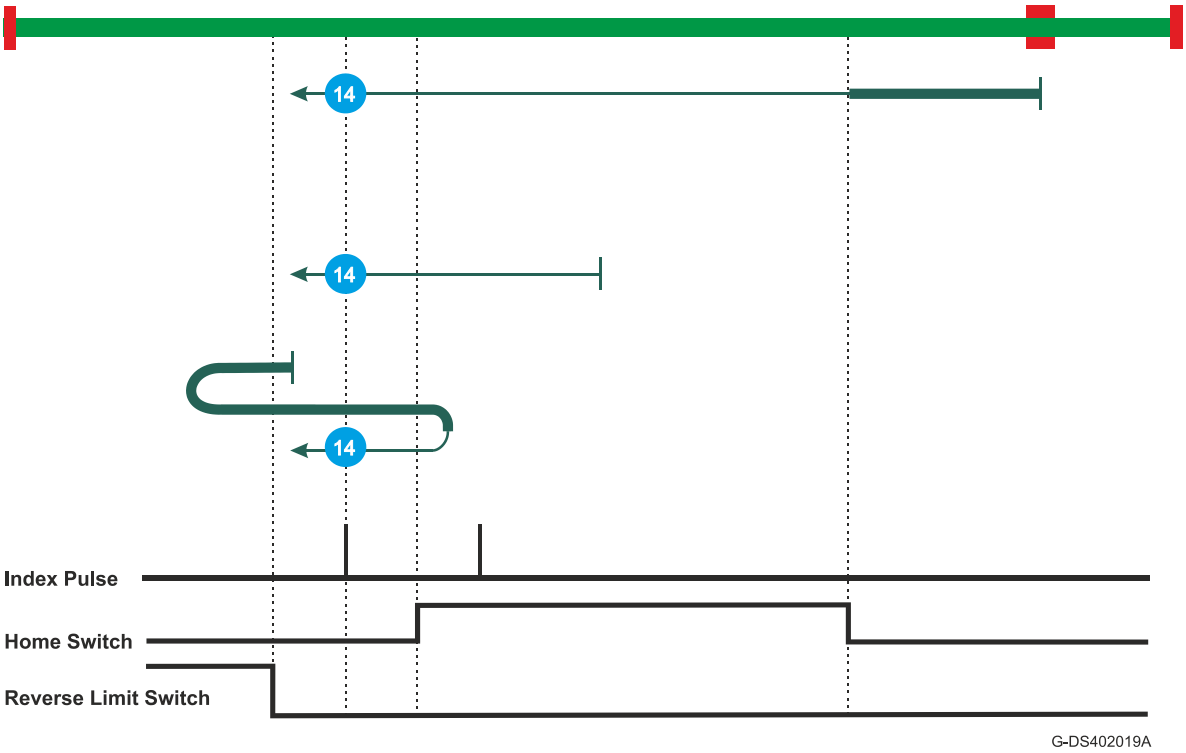


Figure 10-15: Homing on the home switch and index pulse —negative initial move. Method 14

10.9.16. Methods 15 and 16: Reserved

These methods are reserved for future expansion of the homing mode.

10.9.17. Method 17: Homing on RLS

This method is similar to methods 1, except that the home position is not dependent on the index pulse; t is dependent only on the RLS transitions.

When using this method the motor moves in the negative direction with a homing speed defined by 0x6099.1 as long as RLS remains inactive (low on fig.12.2). When active state of RLS is detected, the motor changes direction and speed and moves in the positive direction with a speed defined by 0x6099.2 until RLS changes from active to inactive state. It is considered as a home attained event. The drive sets *home attained bit* in *Statusword* and decelerates with a homing acceleration value (0x609A). The target reached bit in *Statusword* is set when the motor completely stops.

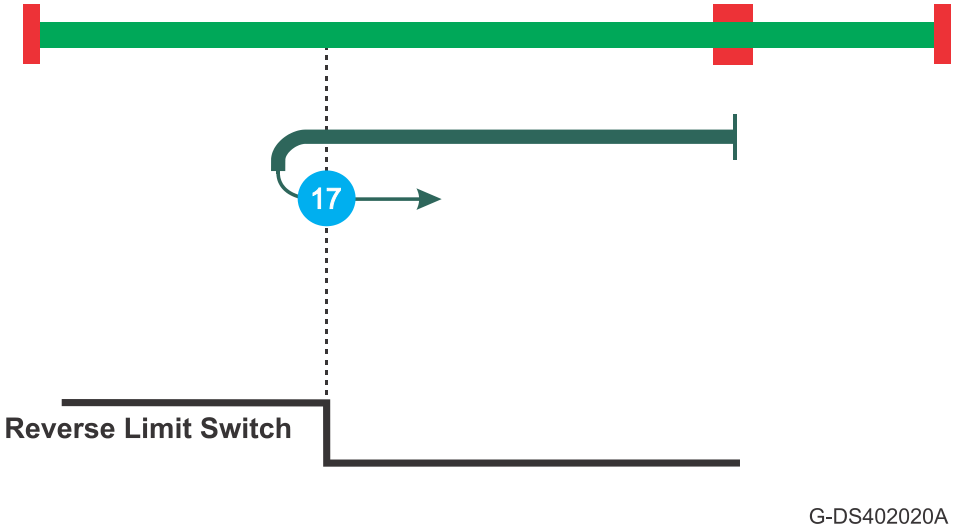


Figure 10-16: Homing on the RLS, Method 17

10.9.18. Method 18: Homing on FLS

This method is similar to method 2, except that the home position is not dependent on the index pulse; it is dependent only on the FLS transitions.

When using this method the motor moves in the positive direction with a homing speed defined by 0x6099.1 as long as FLS remains inactive (low on fig.12.2). When active state of RLS is detected the motor changes direction and speed to one defined by 0x6099.2, and moves in the negative direction until inactive state of FLS is detected. It is considered as a home attained event. The drive sets *home attained bit* in *Statusword* and decelerates with a homing acceleration value (0x609A). The target reached bit in *Statusword* is set when the motor completely stops.

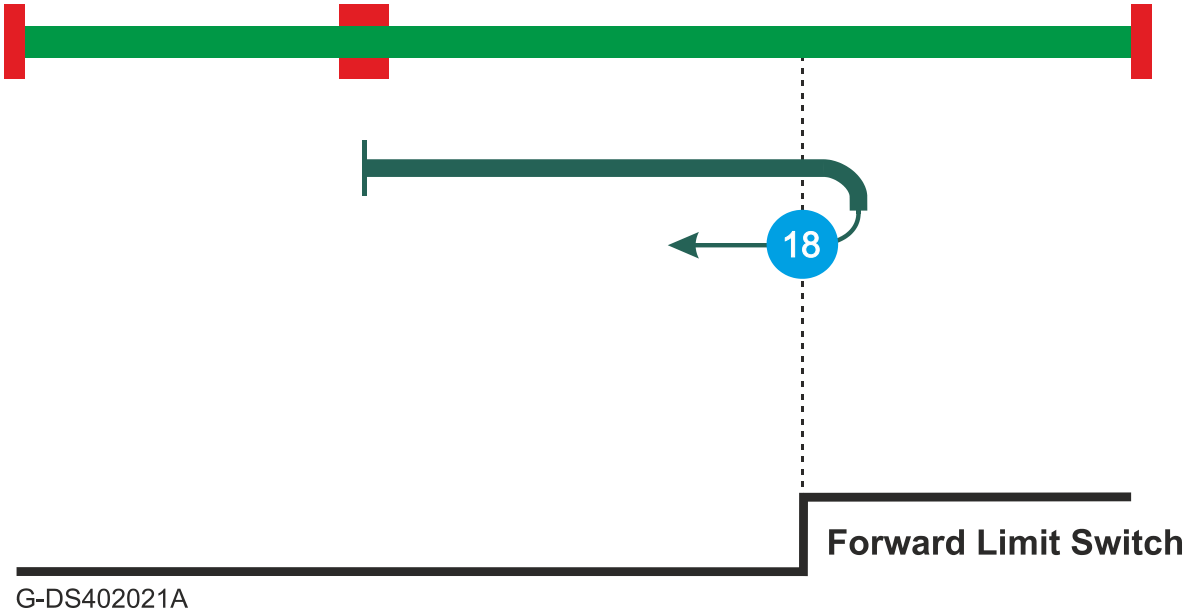


Figure 10-17: Homing on the FLS, Method 18

10.9.19. Method 19: Reverse Homing on Negative Home Switch

This method is similar to method 3, except that the home position is not dependent on the index pulse; it is dependent only on the home switch transitions.

When using method 19, the initial direction of movement depends on the state of the home switch when start homing.

If homing starts and the home switch is inactive, the motor moves in the positive direction with a homing speed defined by 0x6099.1 as long as home switch remains inactive. When active state of home switch is detected the motor changes direction and speed to one defined by 0x6099.2 and moves in the negative direction until the home switch state change from active to inactive is detected. It is considered as home attained event.

If homing starts and the home switch is active, the motor moves in the negative direction with a homing speed defined by 0x6099.1 until home switch inactive state is detected. This is considered as home attained event.

When receiving home attained event the drive sets *home attained bit* in *Statusword* and decelerates with homing acceleration value (0x609A). The target reached bit in *Statusword* is set when the motor completely stops.

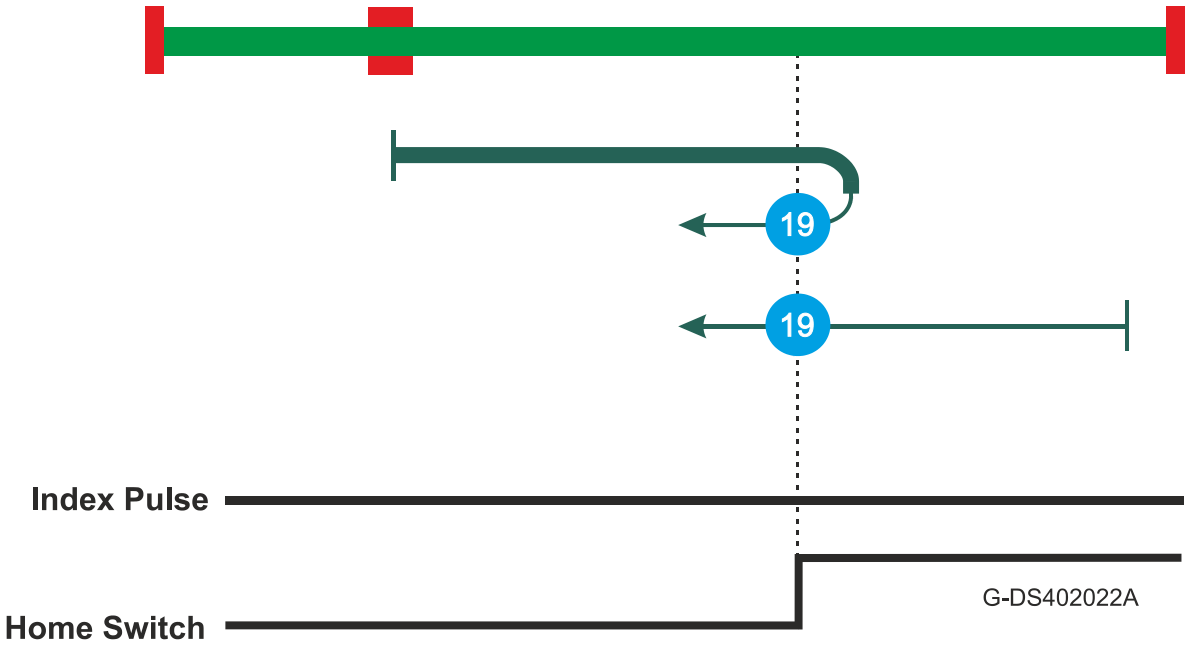


Figure 10-18: Homing without an index pulse, Method 19

10.9.20. Method 20: Forward Homing on Positive Home Switch

This method is similar to method 4, except that the home position is not dependent on the index pulse; it is dependent only on the home switch transitions.

When using method 20, the initial direction of movement depends on the state of the home switch when start homing. If homing starts and the home switch is active, the motor moves in the negative direction with a homing speed defined by 0x6099.1 as long as the home switch remains active. When the inactive state of the home switch is detected, the motor changes direction and speed to one defined by 0x6099.2 and moves in the positive direction until the active state of home switch is detected. It is considered as home attained event.

If homing starts and the home switch is inactive, the motor moves in the positive direction with a homing speed defined by 0x6099.1 until active state of home switch is detected. This is considered as home attained event.

When receiving home attained event the drive sets *home attained bit* in *Statusword* and decelerates with a homing acceleration value (0x609A). The target reached bit in *Statusword* is set when the motor completely stops.

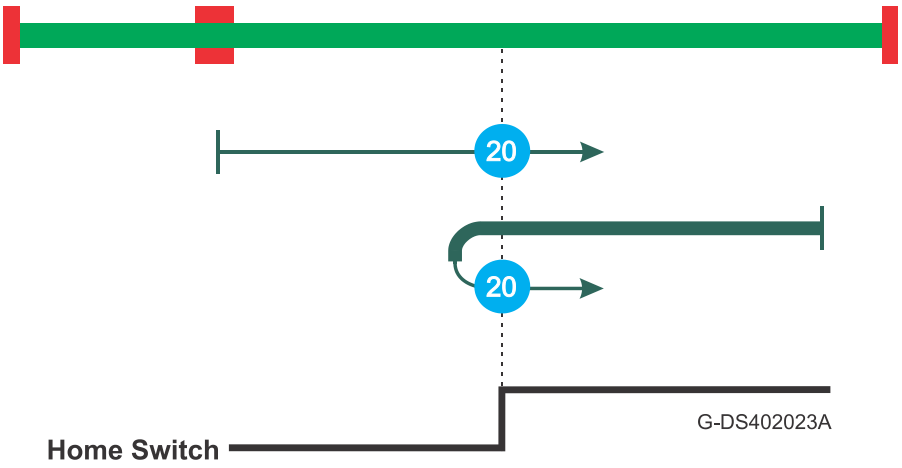


Figure 10-19: Homing without an index pulse, Method 20

10.9.21. Method 21: Forward Homing on Negative Home Switch

This method is similar to method 5, except that the home position is not dependent on the index pulse; it is dependent only on the home switch transitions.

Using method 21, the initial direction of movement depends on the state of the home switch when start homing.

If homing started and home switch is inactive, the motor moves in negative direction with defined by homing speed defined by 0x6099.1 as long as home switch remains inactive. When active state of home switch is detected the motor changes its direction and speed to defined by 0x6099.2 and moves in positive direction until home switch transition from active to inactive state will be detected. It is considered as home attained event.

If homing starts and the home switch is active, the motor moves in positive direction with a homing speed defined by 0x6099.1 until inactive state of home switch is detected. This is considered as home attained event.

When receiving home attained event the drive sets *home attained bit* in *Statusword* and decelerates with a homing acceleration value (0x609A). The target reached bit in *Statusword* is set when the motor completely stops.

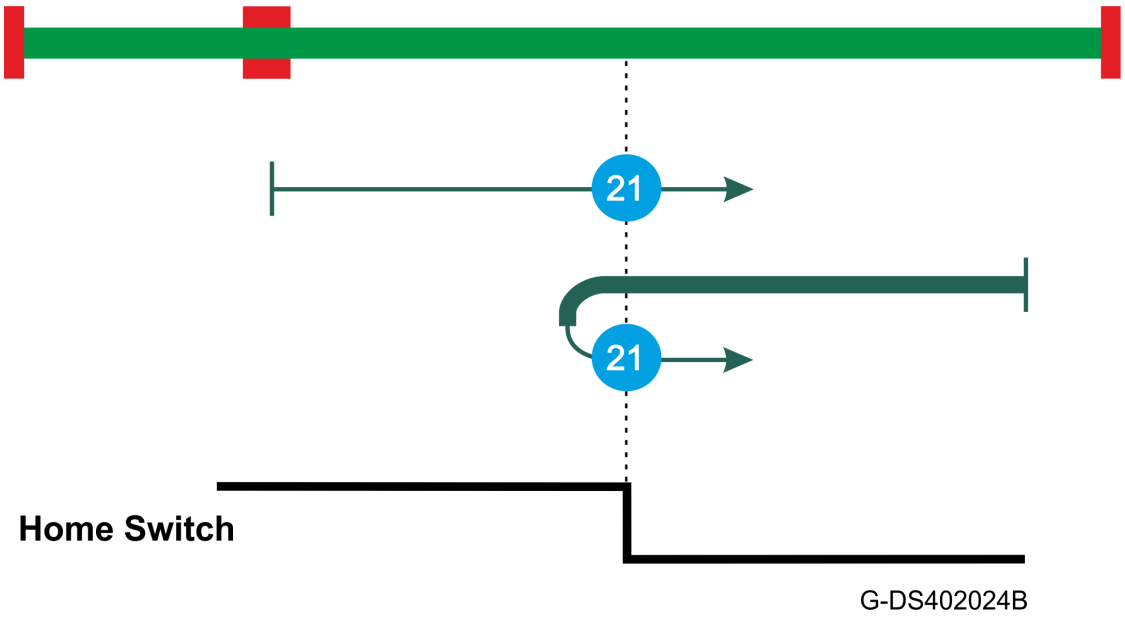


Figure 10-20: Homing without an index pulse, Method 21

10.9.22. Method 22: Reverse Homing on Positive Home Switch

This method is similar to method 6, except that the home position is not dependent on the index pulse; it is dependent only on the home switch transitions.

When using method 22, the initial direction of movement depends on the state of the home switch when start homing.

If homing starts and the home switch is active, the motor moves in the positive direction with a homing speed defined by 0x6099.1 as long as home switch remains active. When the home switch inactive state is detected the motor changes its direction and speed to one defined by 0x6099.2 and moves in the positive direction until home switch state transition from inactive to active is detected. It is considered as a home attained event.

If homing starts and the home switch is inactive, the motor moves in the negative direction with a homing speed defined by 0x6099.2 until the home switch active state is detected. This is considered as home attained event.

When receiving home attained event the drive sets *home attained bit* in *Statusword* and decelerates with a homing acceleration value (0x609A). The target reached bit in *Statusword* is set when the motor completely stops.

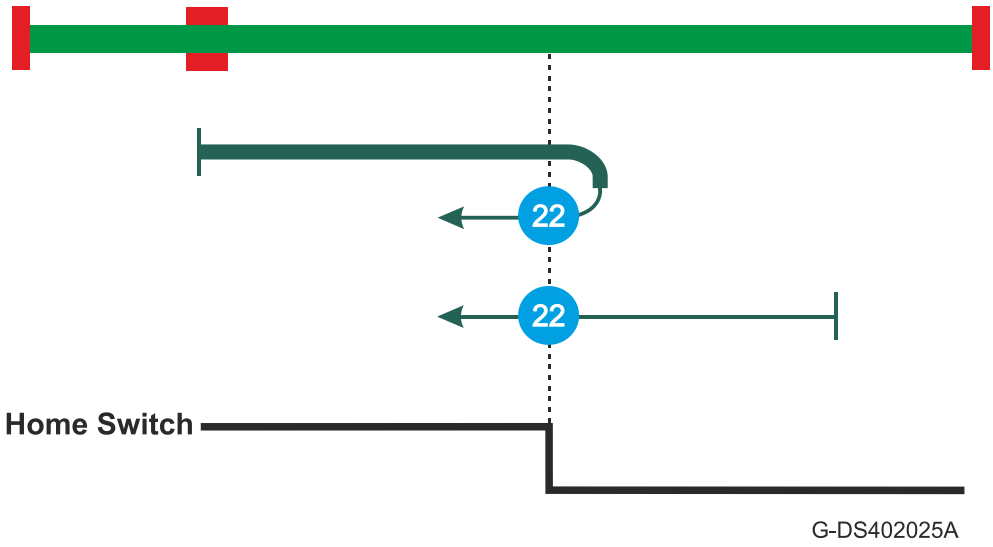


Figure 10-21: Homing without an index pulse, Method 22

10.9.23. Method 23: Reverse Homing on Negative Home Switch/FLS

This method is similar to method 7, except that the home position is not dependent on the index pulse; it is dependent only on the home switch transitions and FLS transition.

When using method 7, the initial direction of movement depends on the state of the home switch when homing is started.

If homing starts and the home switch is inactive, the motor moves in the positive direction with a homing speed defined by 0x6099.1 until one of two possible events will occur:

- If active state of home switch is detected the motor changes its direction and speed to one defined by 0x6099.2 and moves in the negative direction until the home switch state transition from active to inactive is detected. This is considered as a home attained event.
- If active state of FLS is detected, the motor changes direction and speed and moves in the negative direction with the speed defined by 0x6099.2 until the home switch state transition from active to inactive is detected. This is considered as a home attained event.

If homing starts and the home switch is active, the motor moves in the negative direction with a homing speed defined by 0x6099.2 until the home switch changes state from active to inactive. This is considered as home attained event.

When receiving home attained event the drive sets *home attained bit* in *Statusword* and decelerates with a homing acceleration value (0x609A). The target reached bit in *Statusword* is set when the motor completely stops.

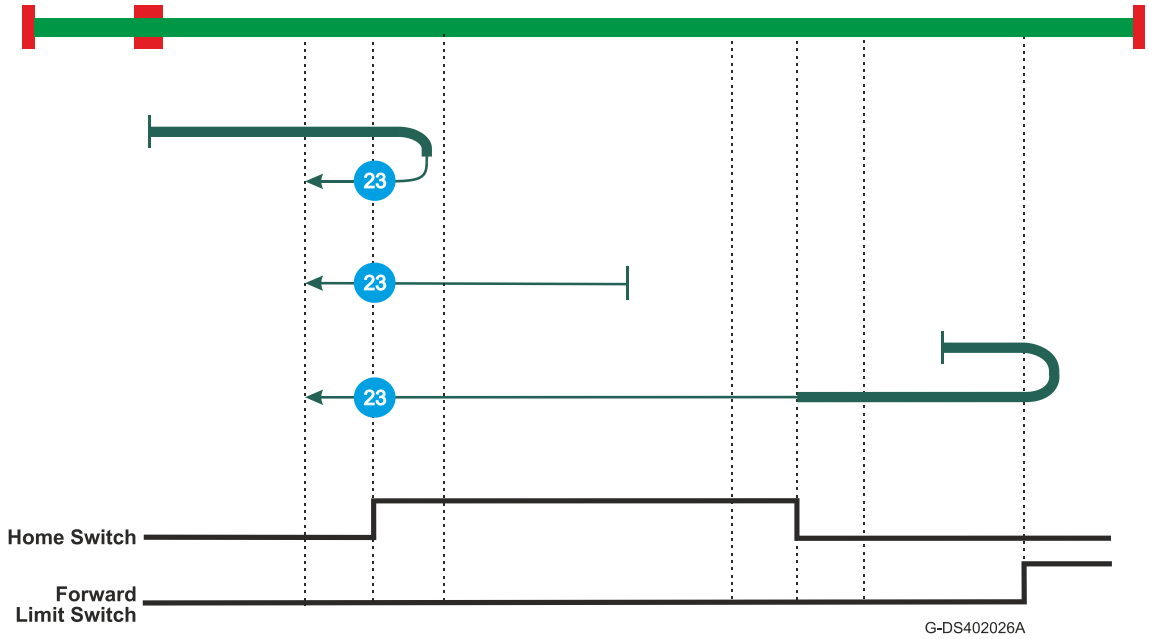


Figure 10-22: Homing without an index pulse, Method 23

10.9.24. Method 24: Forward Homing on Positive Home Switch/FLS

This method is similar to method 8, except that the home position is not dependent on the index pulse; it is dependent only on the home switch transitions and FLS transition.

When using method 24, the initial direction of movement depends on the state of the home switch when start homing.

If homing starts and the home switch is inactive, the motor moves in the positive direction with a homing speed defined by 0x6099.2 until one of two possible events will occur:

- Active state of home switch is detected. This is considered as home attained event.
- Active state of FLS is detected. In this cases the motor changes direction and speed and moves in the negative direction with the speed defined by 0x6099.1 until the home switch state inactive is detected. Then the motor changes its direction and speed to one defined by 0x6099.2 and moves in the positive direction until the home switch state transition from inactive to active is detected. This is considered as home attained event.

If homing starts and the home switch is active, the motor moves in the negative direction with a homing speed defined by 0x6099.1 until the home switch changes state from active to inactive. Then the motor changes its direction and speed to onedefined by 0x6099.2 and moves in the positive direction until the home switch state transition from inactive to active is detected. This is considered as home attained event.

When receiving home attained event the drive sets *home attained bit* in *Statusword* and decelerates with a homing acceleration value (0x609A). The target reached bit in *Statusword* is set when the motor completely stops.

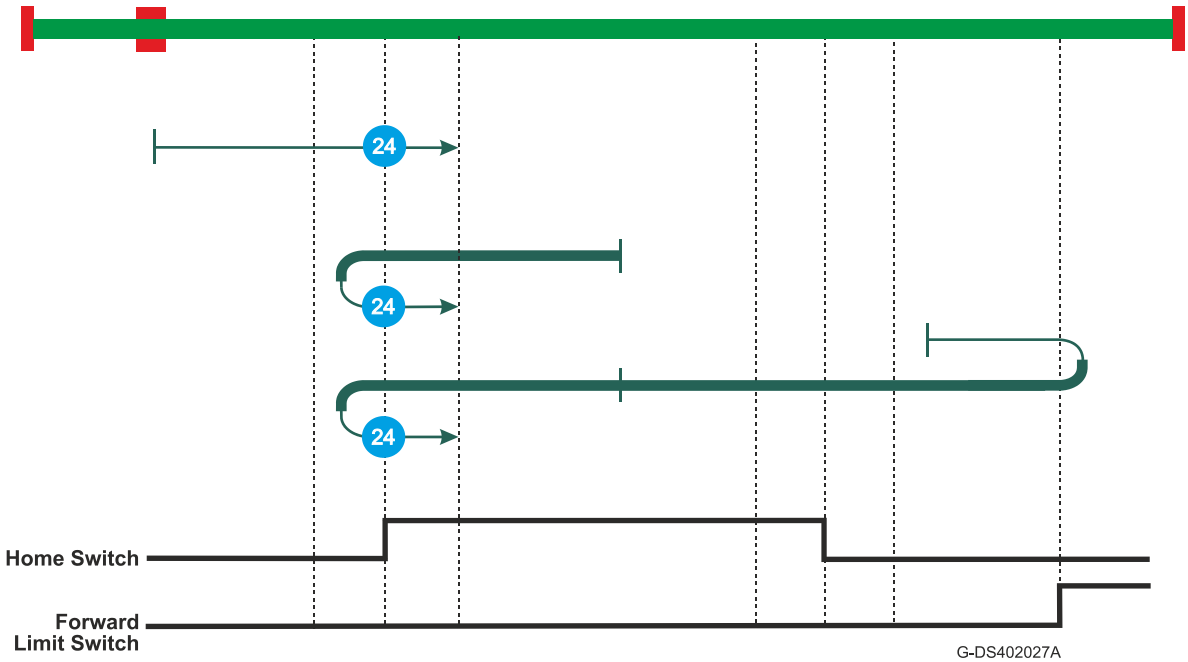


Figure 10-23: Homing without an index pulse, Method 24

10.9.25. Method 25: Reverse Homing on Positive Home Switch/FLS

This method is similar to method 9, except that the home position is not dependent on the index pulse; it is dependent only on the home switch transitions and FLS transition.

When using method 25, the initial direction of movement is always positive.

The motor moves in the positive direction with a homing speed defined by 0x6099.1 until one of two possible events will occur:

- Home switch state transition from active to inactive is detected.
- Active state of FLS is detected.

In both cases the motor changes its direction and speed to one defined by 0x6099.2 and moves in the negative direction until the home switch state transition from inactive to active is detected. This is considered as home attained event.

When receiving home attained event the drive sets *home attained bit* in *Statusword* and decelerates with homing acceleration value (0x609A). Target reached bit in *Statusword* is set when the motor completely stops.

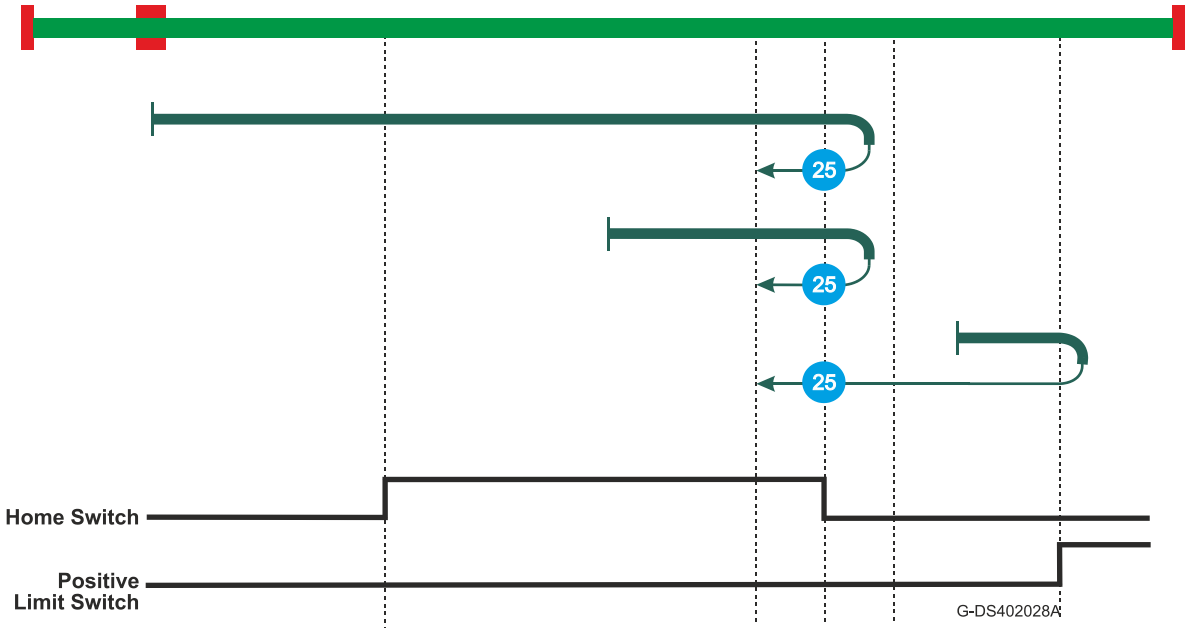


Figure 10-24: Homing without an index pulse, Method 25

10.9.26. Method 26: Forward Homing on Negative Home Switch/FLS

This method is similar to method 10, except that the home position is not dependent on the index pulse; it is dependent only on the home switch transitions and FLS transition.

When using method 26, the initial direction of movement is always positive.

If homing starts and the home switch is inactive, the motor moves in the positive direction with a homing speed defined by 0x6099.1 until one of two possible events will occur:

- Home switch state transition from inactive to active is detected. In this case the motor changes its speed to one defined by 0x6099.2 and continues to move in the positive direction until the home switch from active to inactive state transition is detected. This is considered as home attained event.
- Active state of FLS is detected. In this case the motor changes direction and moves in the negative direction with the same speed until the home switch state active is detected. Then the motor changes its direction and speed to one defined by 0x6099.2 and moves in the positive direction until the home switch state transition from active to inactive is detected. This is considered as home attained event.

If homing started and home switch is active, the motor moves in the positive direction with a homing speed defined by 0x6099.2 until the home switch changes state from active to inactive. This is considered as home attained event.

When receiving home attained event the drive sets *home attained bit* in *Statusword* and decelerates with a homing acceleration value (0x609A). The target reached bit in *Statusword* is set when the motor completely stops.

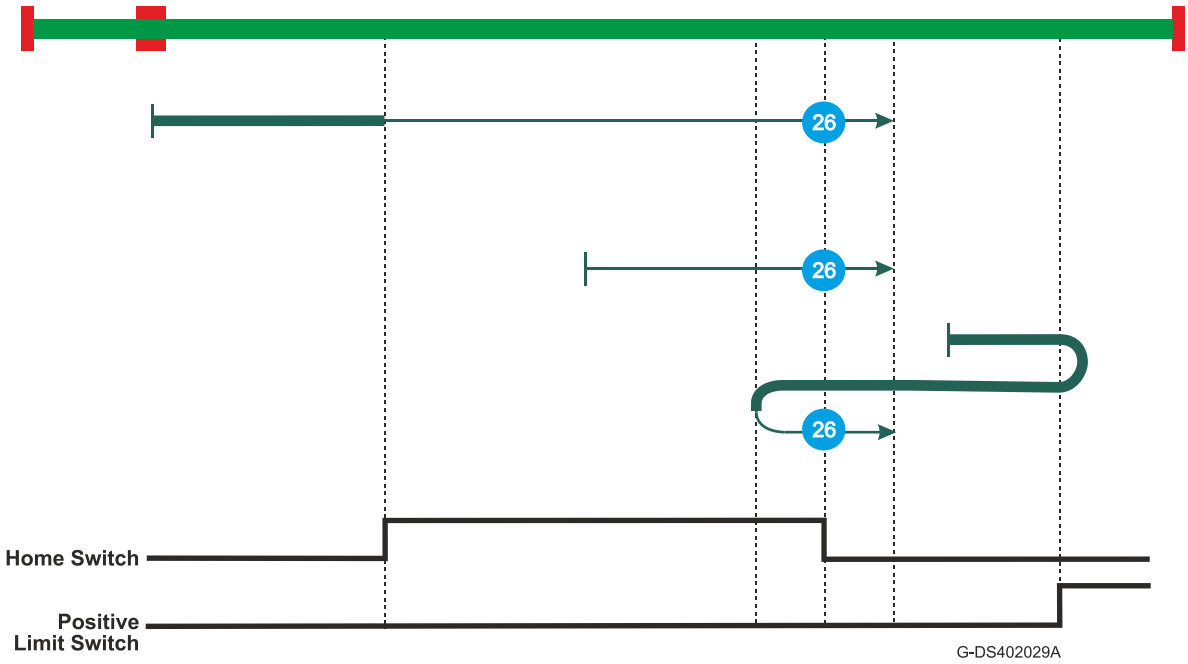


Figure 10-25: Homing without an index pulse, Method 26

10.9.27. Method 27: Forward Homing on Negative Home Switch/RLS

This method is similar to method 11, except that the home position is not dependent on the index pulse; it is dependent only on the home switch transitions and RLS transition.

When using method 27, the initial direction of movement depends on the state of the home switch when start homing.

If homing starts and the home switch is inactive, the motor moves in the negative direction with a homing speed defined by 0x6099.1 until one of two possible events will occur:

- Home switch state active is detected. In this case the motor changes its direction and speed to one defined by 0x6099.2 and moves in the positive direction until the home switch state transition from active to inactive is detected. This is considered as home attained event.
- Active state of RLS is detected. In this case the motor changes direction and moves in the positive direction with the same speed until the home switch active state is detected. Then motor changes speed to one defined by 0x6099.2 and moves in the positive direction until the home switch state transition from active to inactive is detected. This is considered as home attained event.

If homing started and home switch is active, the motor moves in the positive direction with a homing speed defined by 0x6099.2 until the home switch state transition from active to inactive is detected. This is considered as home attained event.

When receiving home attained event the drive sets *home attained bit* in *Statusword* and decelerates with homing acceleration value (0x609A). The target reached bit in *Statusword* is set when the motor completely stops.

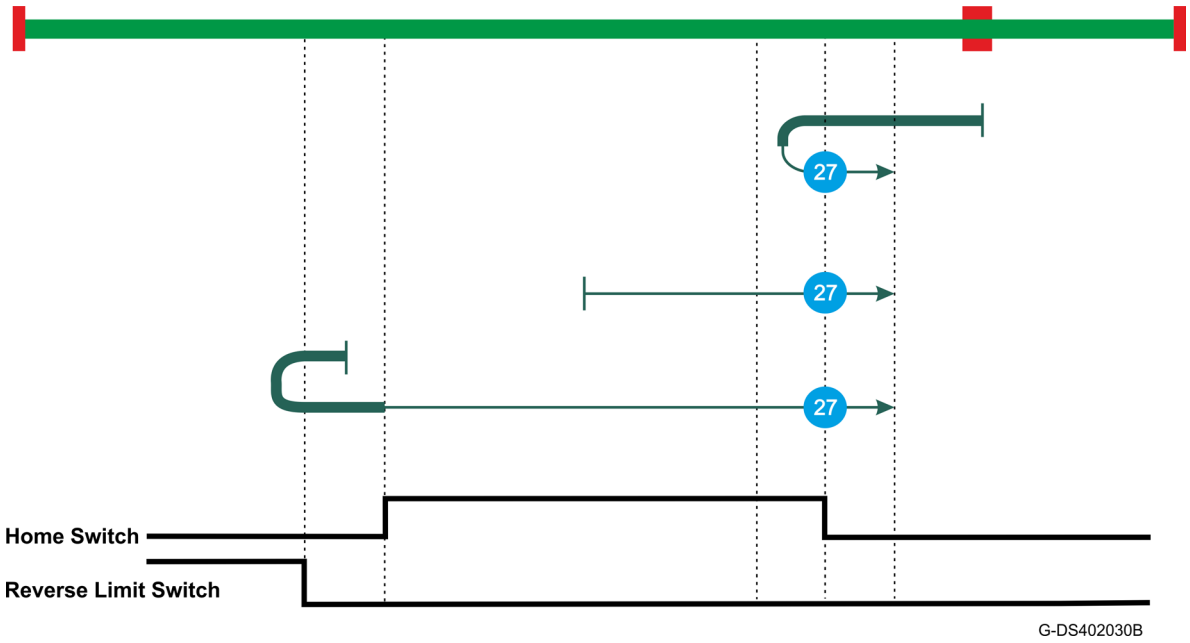


Figure 10-26: Homing without an index pulse, Method 27

10.9.28. Method 28: Reverse Homing on Positive Home Switch/RLS

This method is similar to method 12, except that the home position is not dependent on the index pulse; it is dependent only on the home switch transitions and RLS transition.

When using method 28, the initial direction of movement depends on the state of the home switch when start homing.

If homing starts and the home switch is inactive, the motor moves in the negative direction with a homing speed defined by 0x6099.2 until one of two possible events will occur:

- Home switch state transition from inactive to active is detected. This is considered as home attained event.
- Active state of RLS is detected. In this case the motor changes direction and speed to one defined by 0x6099.1 and moves in the positive direction until the home switch state transition from active to inactive is detected. Then the motor changes direction and speed to one defined by 0x6099.2 and moves in the negative direction until the home switch state transition from inactive to active is detected. This is considered as home attained event.

If homing started and home switch is active, the motor moves in the positive direction with a homing speed defined by 0x6099.1 until the home switch changes state from active to inactive. Then the motor changes its direction and speed to one defined by 0x6099.2 and moves in the negative direction until the home switch state transition from inactive to active is detected. This is considered as home attained event.

When receiving home attained event the drive sets *home attained bit* in *Statusword* and decelerates with a homing acceleration value (0x609A). The target reached bit in *Statusword* is set when the motor completely stops.

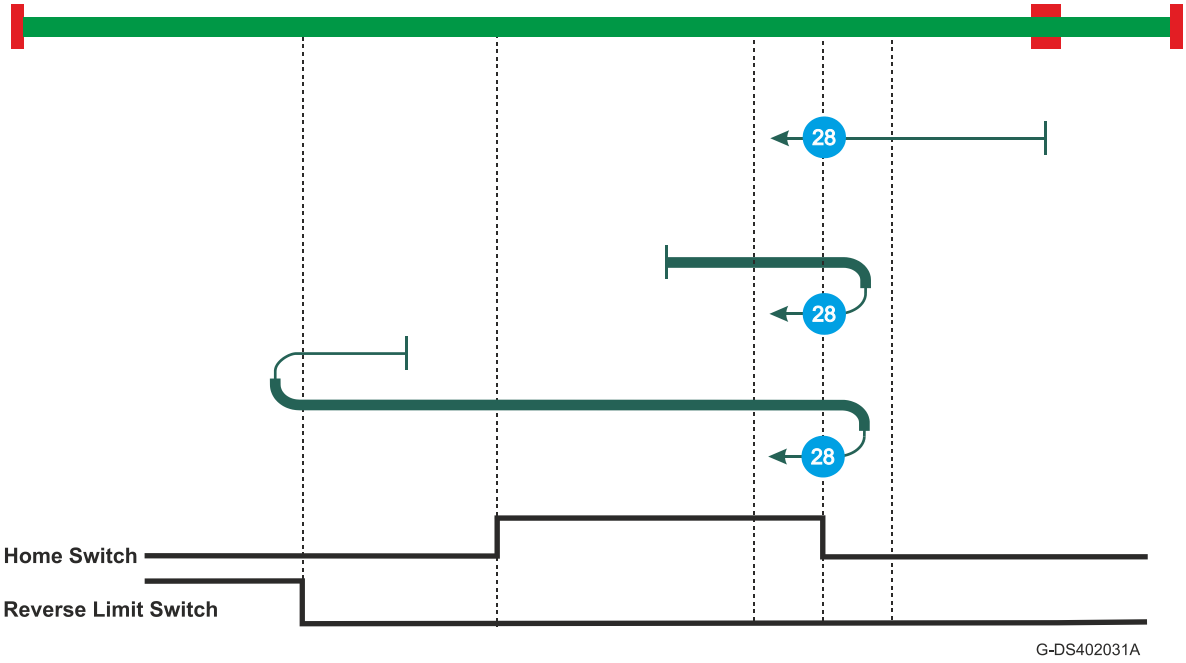


Figure 10-27: Homing without an index pulse, Method 28

10.9.29. Method 29: Forward Homing on Positive Home Switch/RLS

This method is similar to method 13, except that the home position is not dependent on the index pulse; it is dependent only on the home switch transitions and RLS transition.

When using method 29, the initial direction of movement is always negative.

The motor moves in the negative direction with a homing speed defined by 0x6099.1 until one of two possible events will occur:

- Home switch state transition from active to inactive is detected.
- Active state of RLS is detected.

In both cases the motor changes its direction and speed to one defined by 0x6099.2 and moves in the positive direction until the home switch state transition from inactive to active is detected. This is considered as home attained event.

When receiving home attained event the drive sets *home attained bit* in *Statusword* and decelerates with a homing acceleration value (0x609A). Target reached bit in *Statusword* is set when the motor completely stops.

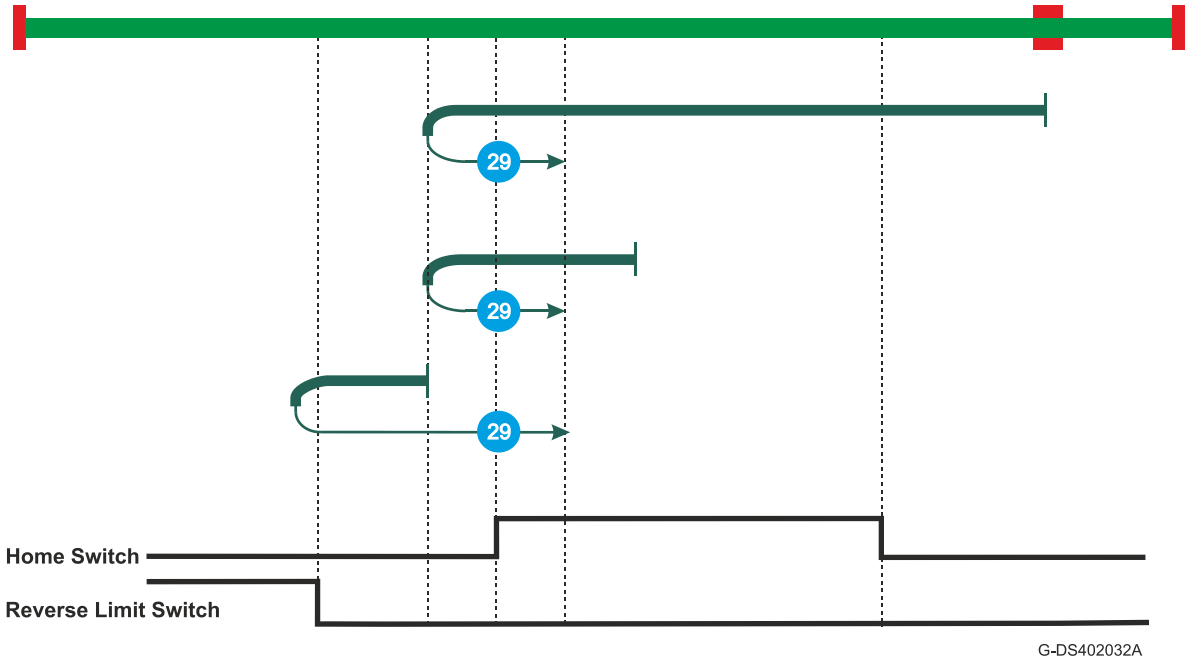


Figure 10-28: Homing without an index pulse, Method 29

10.9.30. Method 30: Reverse Homing on Negative Home Switch/RLS

This method is similar to method 14, except that the home position is not dependent on the index pulse; it is dependent only on the home switch transitions and RLS transition.

When using method 30, the initial direction of movement depends on the state of the home switch when start homing.

If homing starts and the home switch is inactive, the motor moves in the negative direction with a homing speed defined by 0x6099.1 until one of two possible events will occur:

- Home switch state transition from inactive to active is detected. In this case the motor changes its speed to one defined by 0x6099.2 and moves in the negative direction until the home switch state transition from active to inactive is detected. This is considered as home attained event.
- Active state of RLS is detected. In this case the motor changes direction and moves in positive direction until the home switch state transition from inactive to active is detected. Then the motor changes direction and speed to one defined by 0x6099.2 and moves in the negative direction until the home switch state transition from active to inactive is detected. This is considered as home attained event.

If homing starts and the home switch is active, the motor moves in the negative direction with a homing speed defined by 0x6099.2 until the home switch state transition from active to inactive is detected. This is considered as home attained event.

When receiving home attained event the drive sets *home attained bit* in *Statusword* and decelerates with homing acceleration value (0x609A). The target reached bit in *Statusword* is set when the motor completely stops.

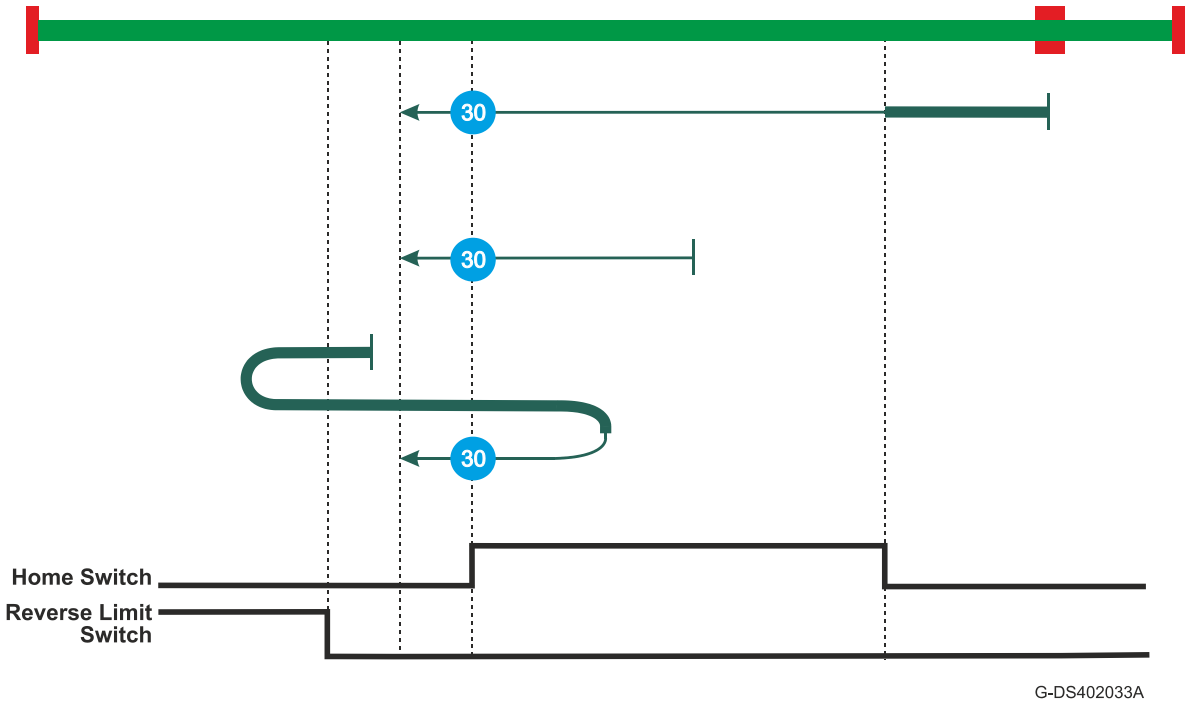


Figure 10-29: Homing without an index pulse, Method 30

10.9.31. Methods 31 and 32: Reserved

These methods are reserved for future expansion of the homing mode.

10.9.32. Methods 33 and 34: Homing on the Index Pulse

Using methods 33 or 34, the direction of homing is negative or positive, respectively. The home position is at the index pulse found in the selected direction.

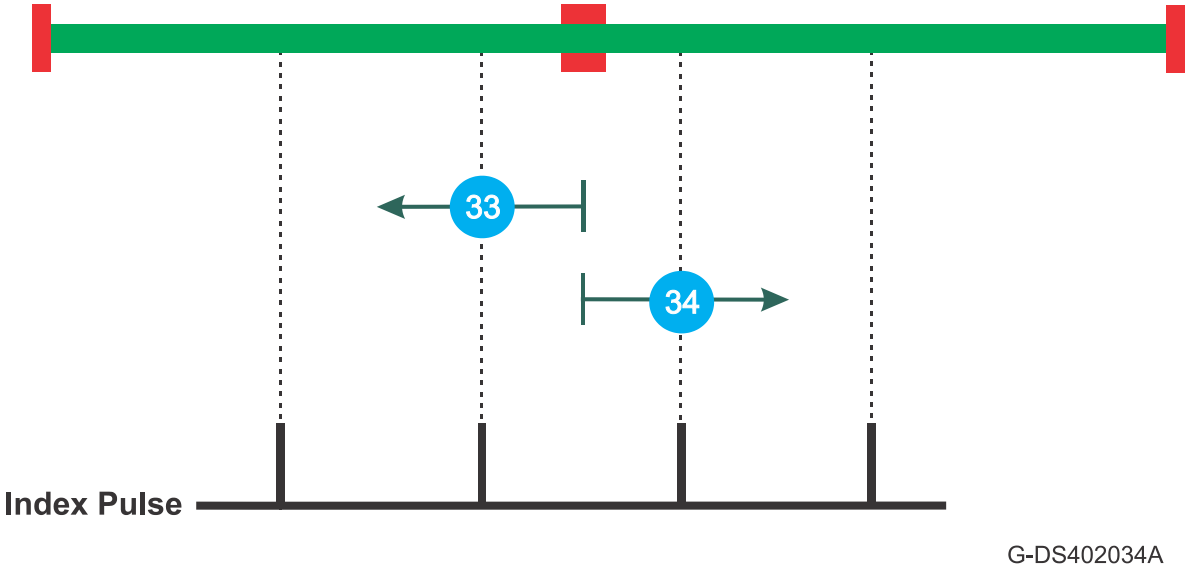


Figure 10-30: Homing on the index pulse

10.9.33. Method 35: Homing on the current position

In this method, the current position is taken to be the home position.

10.9.34. Methods -1 and -2: PLC open Home on Block

These two methods are based on the PLCopen standard and implemented according to the DS-402 Profile Homing motion mode as a manufacturer specific methods. When using these method the drive moves until it is blocked at a define torque and the actual speed is reduced to a defined value. The home is determined by torque, speed and time thresholds.

Method -1 is homing on block (against the wall) in a negative direction.

In this homing mode, the motor moves in negative direction with limited torque (object 0x2020.1) and a defined speed threshold (namely Detection Velocity defined by 0x2020.4) until it reaches a block (see Figure 10-31).

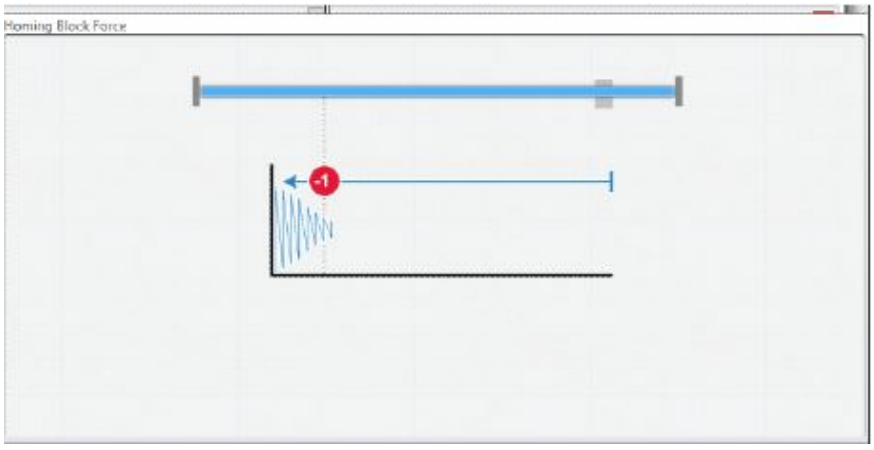


Figure 10-31: Method -1 - Homing on block (against the wall) in a negative direction

When the motor is blocked and the actual torque (object 0x6074) is equal to the limited torque, the drive checks the speed (object 0x606C). If the speed is lower than the detection speed for Detection Speed Time (0x2020.5), home is attained (see Figure 10-32).

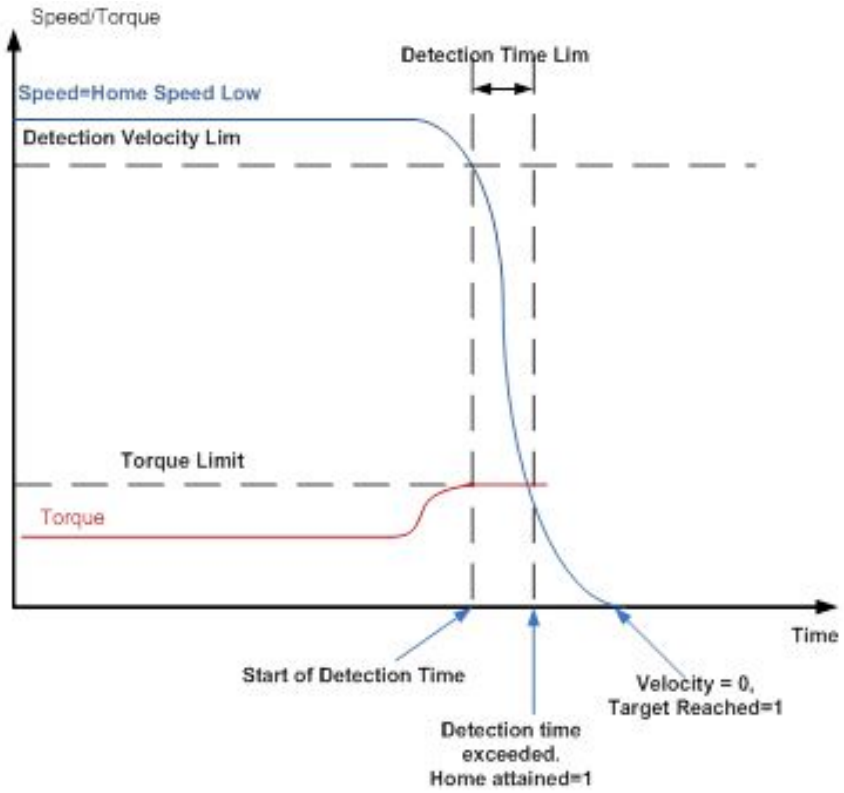


Figure 10-32: Torque is in limit and the actual Speed is lower than the detection speed for Detection Speed Time

Method -2 is homing on block (against the wall) in a positive direction.

In this homing mode, the motor moves in positive direction with limited torque (object 0x2020.1) and a defined speed threshold (namely Detection Velocity defined by 0x2020.4) until it reaches a block (see Figure 10-33). When the motor is blocked and the actual torque (object 0x6074) is equal to the limited torque, the drive checks the speed (object 0x606C). If the speed is lower than the detection speed for Detection Speed Time (0x2020.5), home is attained.

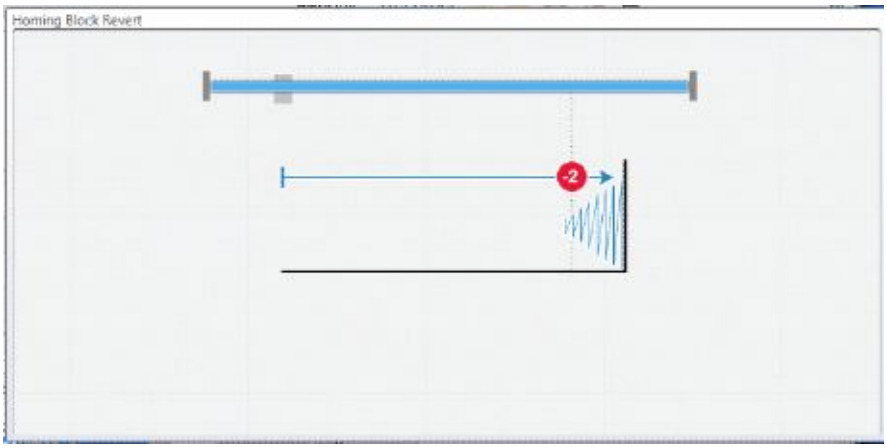


Figure 10-33: Method -2 - Homing on block (against the wall) in a positive direction

10.9.35. Methods -3 and -4: PLC open Home on Block

These two methods are based on the PLCopen standard and implemented according to the DS-402 Profile Homing motion mode as a manufacturer specific methods. When using these methods method the drive moves until it is blocked by a define torque and the speed is reduced to a defined speed. Than the drive moves to the other direction until an Index Pulse is detected.

Method -3 is homing on block (against the wall) in a negative direction and Index pulse.

In this homing mode, the motor moves in negative direction with limited torque (object 0x2020.1) and a defined speed threshold (namely Detection Velocity defined by 0x2020.4) until it reaches a block (see Figure 10-34). When the motor is blocked and the actual torque (object 0x6074) is equal to the limited torque, the drive checks the speed (object 0x606C). If the speed is lower than the detection speed for Detection Speed Time (0x2020.5), then the motor moves to the positive direction until the Index pulse is detected where the home is attained. Note that during the negative move, the drive limits the torque to 0x2020.1 and for the positive direction the torque limit which was used prior to the homing sequence will be restored.

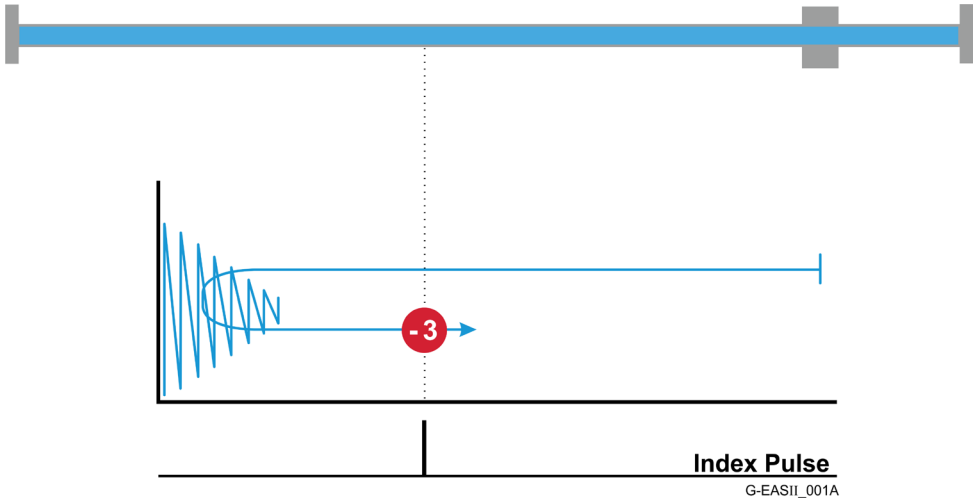


Figure 10-34: Method -3 - Homing on block (against the wall) in a negative direction

Method -4 is homing on block (against the wall) in a positive direction and Index Pulse.

In this homing mode, the motor moves in the positive direction with limited torque (object 0x2020.1) and a defined speed threshold (namely Detection Velocity defined by 0x2020.4) until it reaches a block (see Figure 10-35). When the motor is blocked and the actual torque (object 0x6074) is equal to the limited torque, the drive checks the speed (object 0x606C). If the speed is lower than the detection speed for Detection Speed Time (0x2020.5), the motor moves to the negative direction until the Index pulse is detected where Home is attained. Note that during the positive move, the drive limits the torque to 0x2020.1 and for the negative direction the torque limit which was used prior to the homing sequence will be restored.

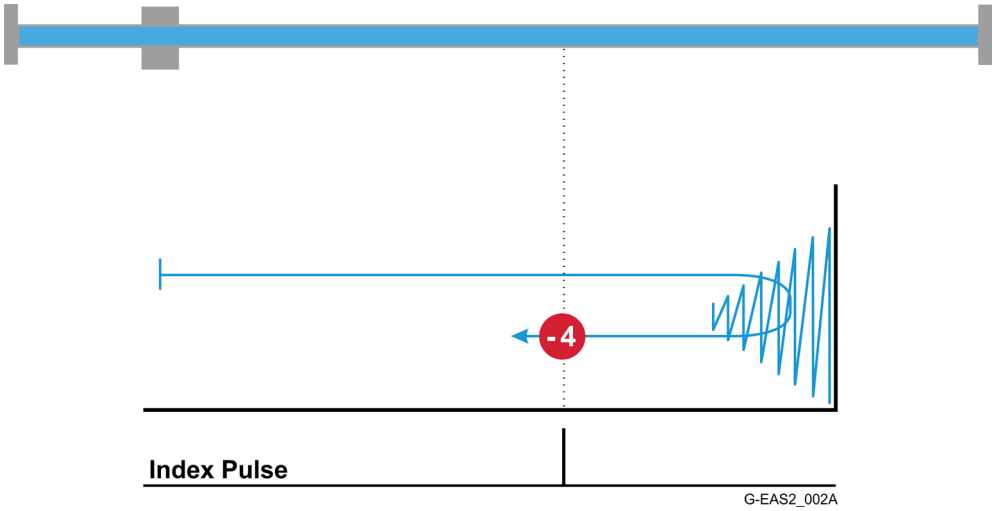


Figure 10-35: Method -4 - Homing on block (against the wall) in a positive direction

Chapter 11: Position Control Function

11.1. General

Object	Definition
0x6062	Position demand value in position units
0X6063	Position actual value in increments
0X 6064	Position actual value
0X 6065	Following error window
0X 6067	Position window
0X 6068	Position window time out
0X 60F4	Following error actual value
0X 60FA	Control effort
0X 60FC	Position demand internal value in increments

This chapter describes all parameters required for closed-loop position control. The control loop is fed with the position demand value 0x6062 as one of the outputs of the trajectory generator and with the output of the position detection unit (position actual value, *0x6064*) as input parameters. The behavior of the control is influenced by the control parameters. Position control parameters (PI/P) may be set using the EAS during setup. The inputs and output of closed loop position control function is presented on Figure 11-1.

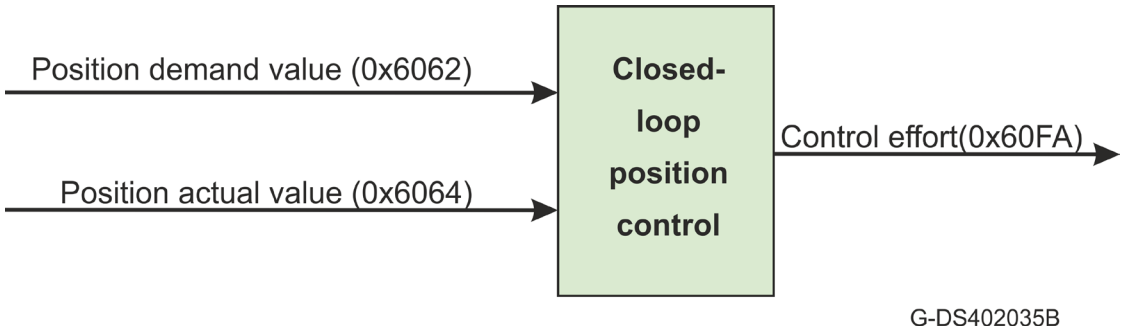


Figure 11-1: Position Control Function

To ensure that the physical limits of a drive are not exceeded, an absolute limit function is implemented for the position control effort. The Elmo drive implements a cascaded control loop in which the position control effort is a velocity demand value for the velocity control loop. For further information about tuning the position loop and using the EAS, refer to the EAS User Manual and the Drive Administrative Software Manual.

The following terms are used in this chapter:

Following error	A *position actual value* outside the allowed range of the *following error window* around a *position demand value* for longer than the *following*

	error timeout results in setting bit 13, *following error*, in the *Statusword* to 1 in operating modes csp, ip, pp. Depending on the supported modes of operation and on the capabilities of different categories of drives, only some of the mentioned input parameters may be necessary.
Position reached	This function, shown in Figure 11-2, provides the option of defining a position range around a *position demand value* to be regarded as valid. If a drive position is within this area for a specified time — the *position window time* — the related control bit 10 target reached in the *Statusword* is set to 1 in operating modes csp, ip, pp.

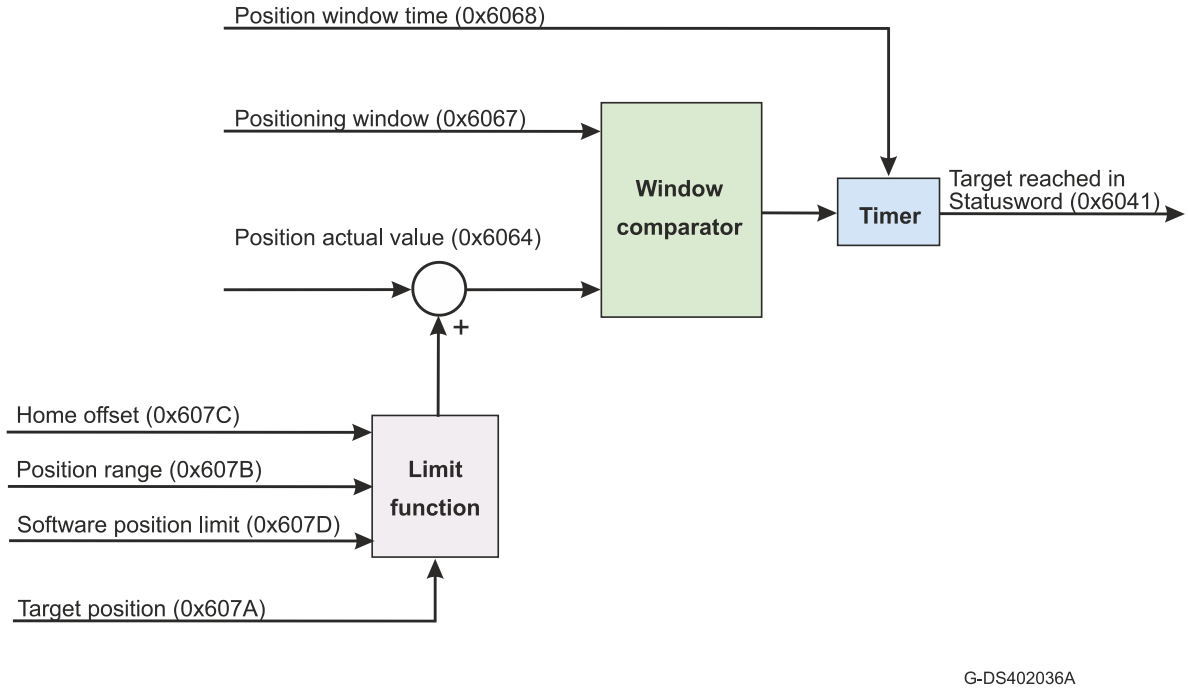


Figure 11-2: Position Reached Function

11.2. Object 0x6062: Position demand value

The value of this Object is taken from the internal position command and is given in position units after being converted by *position factor.*

- Object description:

Attributes	0x6062
Name	Position demand value
Object code	VAR
Data type	INTEGER32
Category	Optional

- Entry description:

Access	Read only
PDO mapping	Yes
Value range	-2^{32} - $(2^{32})-1$
Default value	0

11.3. Object 0x6063: Position actual internal value

The actual value of the position measurement device is one of the two input values of the closed loop position control. The data unit is defined as increments.

- Object description:

Attributes	0x6063
Name	Position actual value - increments
Object code	VAR
Data type	INTEGER32
Category	Mandatory

- Entry description:

Access	Read only
PDO mapping	Yes
Value range	-2^{32} - $(2^{32})-1$
Default value	0

11.4. Object 0x6064: Position actual value

This Object represents the actual value of the position sensor, in user-defined units. It returns the motor position feedback (**PU** command) value.

- Object description:

Attributes	0x6064
Name	Position actual value
Object code	VAR
Data type	INTEGER32
Category	Optional

- Entry description:

Access	Read only
PDO mapping	Yes
Value range	-2^{32} - $(2^{32})-1$
Default value	0

11.5. Object 0x6065: Following error window

This Object defines a range of tolerated position values symmetrical to the *position demand value.* If the *position actual value* is outside the *following error window* during a defined following-error-time-out, a following error bit is set in *Statusword* in operating modes csp, ip, pp. Setting of the following error bit does not cause to a Fault state and the servo will not be disabled.

The following error indications may occur:

- When a drive is blocked
- When the profile velocity is unreachable
- Due to wrong closed loop coefficients

The value of this object is incremental. By default, when the drive is powered up, this object is set internally to **ER[3]**.

Note: **This object does not disable the motor in situations that the tracking error violates the 0x6065 threshold.**

- Object description:

Attributes	0x6065
Name	Following error window
Object code	VAR
Data type	UNSIGNED32
Category	Optional

- Entry description:

Access	Read/write
PDO mapping	None
Value range	0 - 2^{32}-1
Default value	Is set internally to **ER[3]**

11.6. Object 0x6066: Following Error Time Out

When a following error occurs and continues longer than the defined value of the timeout, the corresponding bit 13 *following error* in the *Statusword* is set to 1 indicating a following error threshold violation.This indication is valid in csp, ip, pp modes. The drive power state is not effected. The drive remains in servo and does not report a Fault state.

- Object description:

Attributes	0x6066
Name	Following error time out
Object code	VAR
Data type	UNSIGNED16
Category	Optional

- Entry description:

Access	Read/write
PDO mapping	No
Value range	1 - 65535
Default value	20

11.7. Object 0x6067: Position window

This Object defines a symmetrical range of accepted positions relative to the *target position*. If the actual value of the position encoder is within the *position window during position window time*, this *target position* is regarded as reached. The Target reached bit(bit 10) in the *Statusword* will be set to 1 in profile position mode. The object has no role in IP & CSP motion modes.

- Object description:

Attributes	0x6067
Name	Position window
Object code	VAR
Data type	UNSIGNED32
Category	Optional

- Entry description:

Access	Read/write
PDO mapping	No
Value range	0 – (2^{32})-1
Default value	100

11.8. Object 0x6068: Position window time

When the actual position is within the *position window* during the defined *position window time* — given in milliseconds — the corresponding bit 10 target reached in the *Statusword* is set to 1 in pp mode. The object has no role in csp & ip modes. Refer to the description in the *position window* object.

- Object description:

Attributes	0x6068
Name	Position window time
Object code	VAR
Data type	UNSIGNED16
Category	Optional

- Entry description:

Access	Read/write
PDO mapping	No
Value range	0 - 65535
Default value	20

11.9. Object 0x60F4: Following error actual value

The object defines the following error actual value. The following error is the difference between the position demand and the position feedback values and is given in position units.

- Object description:

Attributes	0x60F4
Name	Following error actual value
Object code	VAR
Data type	INTEGER32
Category	Optional

- Entry description:

Access	Read only
PDO mapping	TxMap
Value range	-2^{31} - $(2^{31})-1$
Default value	0

11.10. Object 0x60FA: Control effort

This object indicates the control effort as the output of the position control loop. The value is measured in user-defined position units.

- Object description:

Attributes	0x60FA
Name	Control effort
Object code	VAR
Data type	INTEGER32
Category	Optional

- Entry description:

Access	Read only
PDO mapping	TxMap
Value range	-2^{31} - $(2^{31})-1$
Default value	0

11.11. Object 0x60FC: Position demand internal value - increments

This output of the trajectory generator in position mode is an internal value using increments.

- Object description:

Attributes	0x60FC
Name	Position demand value – in increments
Object code	VAR
Data type	INTEGER32
Category	Optional

- Entry description:

Access	Read only
PDO mapping	Yes
Value range	-231 - (231)-1
Default value	0

11.12. Object 0x60F2: Positioning Option Code

This Object defines the configured positioning behavior. Figure 11-3 presents the Object bit placement.

15	14 12	11 8	7 6	5 4	3 2	1 0
ms	Reserved	ip option	rado	rro	cio	Relative option

MSB **LSB**

Figure 11-3: Object 0x60F2 Bit Placement

The following abbreviations are part of Figure 11-3:

Abbreviation	Description
ms	Manufacturer-specific, always 0, not in use
rro	Request-response option
cio	Change immediately option
rado	Rotary axis direction option

The *relative option* bits control the drives behavior when *Absolute\Relative* bit - bit 6 - in *Controlword* is set to 1 in pp mode.

Bit 1	Bit 0	Definition
0	0	Positioning moves are performed relative to the preceding -**Target Position (0x607A)** -target position
0	1	Positioning moves are performed relative to the position demand value –**Object 60FCh** – output of the trajectory generator
1	0	Positioning moves are performed relative to the position actual value - **Object 0x6064**
1	1	Reserved

The *cio* bits control the behavior of the drive when *change set point immediately* (bit 5) in *Controlword* is set to 1 in pp mode

Bit 3	Bit 2	Definition
0	0	The drive performs the new set-point immediately
0	1	The actual performed set-point shall continue and blend to the new set point when the target position is reached – w/o attempting to stop.
1	0	Reserved
1	1	Reserved

The *rro* bits enable the drive to release the *new set point* - bit 4 -in Controlword in pp mode internally in order to avoid resetting this bit by control device –master.

Bit 5	Bit 4	Definition
0	0	The drive performs motion according to the *Controlword* received from master
0	1	The drive releases bit 4 in *Controlword* automatically as soon as the target is reached.
1	0	The drive releases bit 4 in *Controlword* automatically if there is place in set point buffer
1	1	Reserved

The *rado* bits control the movement when software position limits, see the description in **Object 0x607D** and the target position, see the description in **Object 0x607A** are wider then position range limits, see the description in **Object 0x607B**. As a result, different movements are possible depending on *rado* bits.

Bit 7	Bit 6	Definition
0	0	The normal positioning is similar to linear axis. When reaching or exceeding the position range limits, see **Object 0x607B**, the input value is automatically wrapped to the other end of the range. Positioning is relative or absolute. This is the only bit combination in which the target position can be greater than a modulo value.
0	1	Positioning is only in a negative direction. If target position is higher than the actual position, the axis moves over the minimum position limit, see **Object 0x607D** sub-index 0x01, to the target position
1	0	Positioning is only in a positive direction. If target position is lower than the actual position, the axis moves over the maximum position limit, see **Object 0x607D** sub-index 0x01, to the target position
1	1	Positioning by using the shortest way to the target position. If the difference between the actual value and target position in a 360^0 system is 180^0, the axis moves in a positive direction.

The following Object settings are an example of movement depending on setting in bits 6, 7. The settings are presented in Figure 11-4.

Note: **The only normal motion type (rado bits are 0,0) is available in the cases:**

- XM[1]=XM[2]
- XM[1]<=VL[3] && XM[2]>=VH[3]

Object	Setting
0x607B.1	0
0x607B.2	360

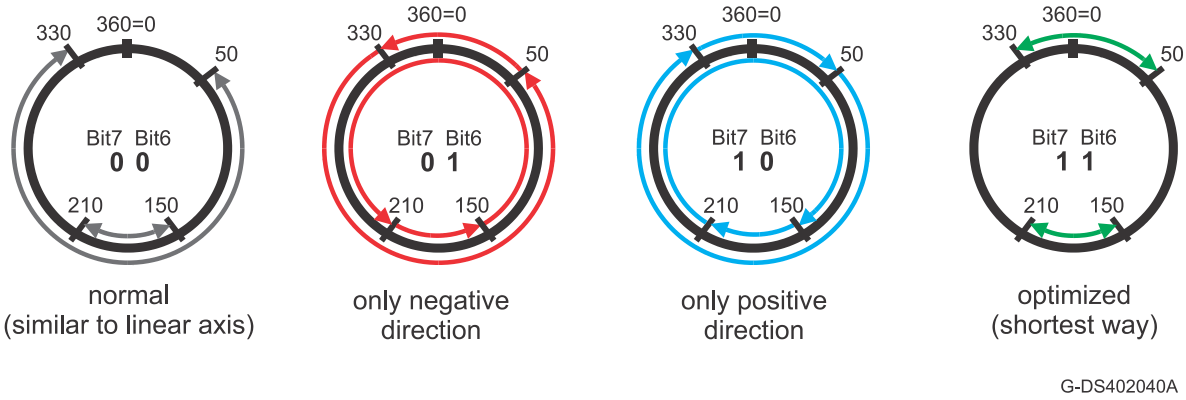


Figure 11-4: Influence of Rado Bits

The following Object settings are an example of two absolute motions with settings bits 6, 7. The settings are presented in Figure 11-5.

Object	Setting
0x607B.1	0
0x607B.2	1000
0X607A	Initially 2500
0607A	Changed to 200

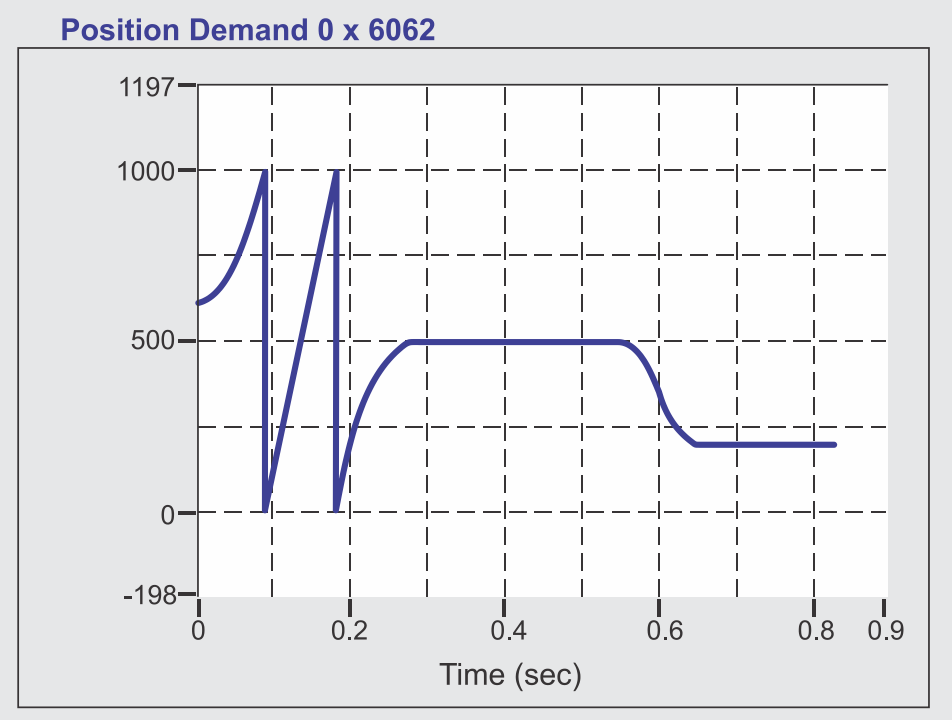


Figure 11-5: Absolute Motion with Rado = 0

The following Object settings are an example of two relative motions with settings bits 6, 7. The settings are presented in Figure 11-6

Object	Setting
0x607B.1	0
0x607B.2	1000
0X607A	Initially 2500
0607A	Changed to 200

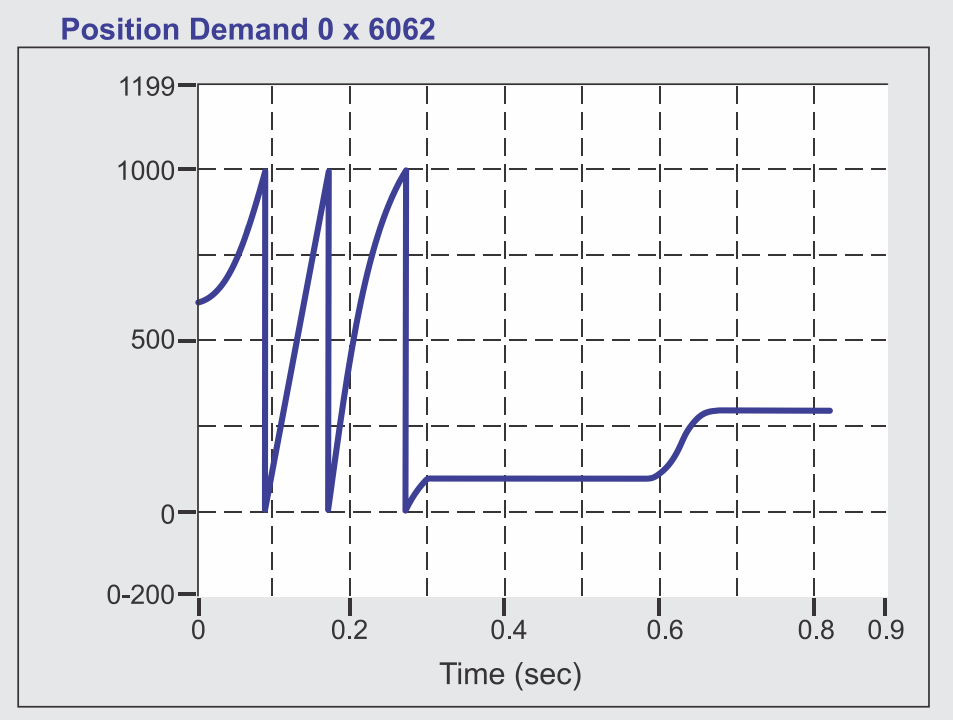


Figure 11-6: Relative Motion with Rado = 0

Chapter 12: Profiled Position

12.1. General

Object	Definition
0x607A	Target position
0x607B	Position range limit
0x607D	Software position limit
0x607F	Maximum profile velocity
0x6081	Profiled velocity
0x6082	End velocity
0x6083	Profiled acceleration
0x6084	Profiled deceleration
0x6086	Motion profile type
0x60C5	Maximum acceleration
0x60C6	Maximum deceleration
0x60F2	Positioning Option Code

This chapter describes how to set a point-to-point (PTP) movement under a profiled position where a *target position* is applied to the trajectory generator. It generates a *position demand value* to the control loop. The trajectory generator input includes profiled velocity, acceleration, deceleration, and selection of motion type, motion polarity and stopping option.

12.2. Profile Position Mode

The general structure of the Profile Position Mode is presented in Figure 12-1

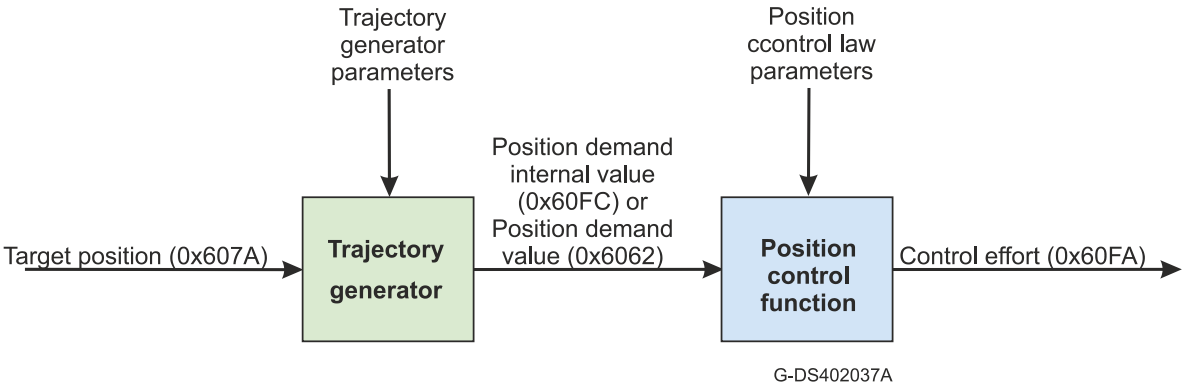


Figure 12-1: Profile Position Mode General Structure

Figure 12-2 presents a detailed structure of a trajectory generator (profiler).

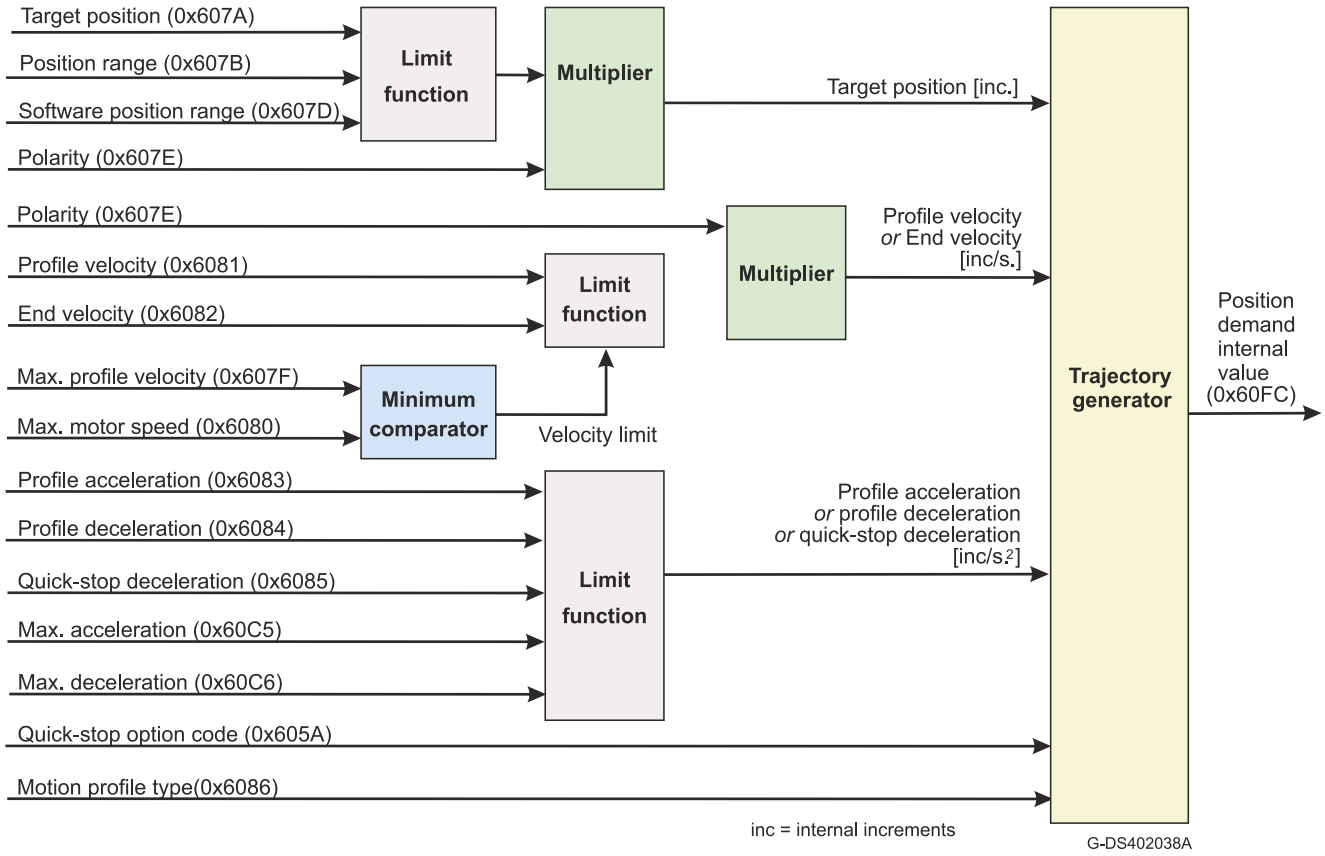


Figure 12-2: Profiler Position Mode, Profiler Structure

12.2.1. *Controlword* of the Profiled Position Mode

Bit	Function
0	Switch on, see the description in **Object 0x6040** *Controlword*
1	Enable voltage, see the description in **Object 0x6040** *Controlword*
2	Quick stop, see the description in 0x6040 *Controlword*
3	Enable operation (set motor on) , see the description in **Object 0x6040** *Controlword*
4	New set-point
5	Change set-point immediately
6	Absolute\Relative movement 0: Target position is absolute value 1: Target position is relative value depending on object 0x60F2
7	Fault reset, see the description in **Object 0x6040** *Controlword*
8	Halt: 0: Position shall be continued or executed 1: When set the drive halts according to Halt Option Code (**0x605D**)
9	Change on set-point (Blended)
10	Reserved
11	Reserved
12	Reserved
13	Reserved
14	Reserved
15	Reserved

The relations between the motion mode bits of the *Controlword* is described below. Further details refer to functional descriptions.

Motion Type	Bit Name and Values				
	Bit 9. Blended	Bit 8. Halt	Bit 6. Relative	Bit 5. Change set point immediately	Bit 4. New set point
New set point is buffered until previous set point is reached. After which the new set point is executed	0	0	X	0	0->1

Motion Type	Bit Name and Values				
The new set point is executed immediately. The previous set point is aborted.	X	0	X	1	0->1
Blended motion. The actual performed positioning task is continued without attempting to stop on target position and is blended to the newly commanded task (considering potentially changed profile velocity and accelerations etc.) when target position is touched	1	0	X	0	0->1
Set point is processed as absolute position	X	0	0	X	0->1
Set-point is processed as relative position in accordance with Positioning option code (0x60F2).	X	0	1	X	0->1
Motion stopped according to halt option code (**0x605D**)	X	1	X	X	X

12.2.2. *Statusword* of the profiled position mode

Bit	Function
0	Ready to switch on, see the description in **Object 0x6041** *Statusword*
1	Switched on, see the description in **Object 0x6041** *Statusword*
2	Operation enabled, see the description in **Object 0x6041** *Statusword*
3	Fault, see the description in **Object 0x6041** *Statusword*
4	Voltage enabled, see the description in **Object 0x6041** *Statusword*
5	Quick stop, see the description in **Object 0x6041** *Statusword*
6	Switch on disabled, see the description in **Object 0x6041** *Statusword*
7	Warning, see the description in **Object 0x6041** *Statusword*
8	Manufacturer specific, reserved, always 0

Bit	Function
9	Remote, see the description in **Object 0x6041** *Statusword*
10	Target reached
11	Internal limit active, see the description in **Object 0x6041** *Statusword*
12	Set new point acknowledge
13	Following error
14 - 15	Manufacturer specific, reserved, always 0

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.
New set-point acknowledge	0	Set-point buffer is available, previous Set-point already processed
	1	Previous Set-point still processed, Set-point buffer is not available.
Following error	0	No following error
	1	Following error.

12.3. Object 0x607A: Target position

The *target position* is the set-point position to which the drive should move in profile position mode, using the current settings of motion control parameters such as velocity, acceleration, deceleration and *motion profile type*. The *target position* is given in user-defined position units. It is converted to position increments using the *position factor*. The *target position* is interpreted as absolute or relative, depending on the Relative flag in the *Controlword*.

The object is also relevant for csp mode

- Object description:

Attributes	0x607A
Name	Profile target position
Object code	VAR
Data type	INTEGER32
Category	Mandatory

- Entry description:

Access	Read/write
PDO mapping	**ECAT:** RxMap **CAN:** RxMap, TxMap
Value range	-2^{31}…$(2^{31})-1$
Default value	0

12.4. Object 0x607B: Position range limit

This object contains two sub-indices that limit the numerical range of the motion: *min position range limit* and *max position range limit.* On reaching or exceeding these limits, the feedback value automatically wraps to the other end of the range. Wrap-around is prevented by setting software position limits.

The value of the position range limit is reflected in the **XM[1]** and **XM[2]** commands, to which the range and restrictions are ultimately submitted, see the Gold drive Command Reference Manual.

When the drive is powered up the object gets **XM[1]** and **XM[2]** to sub index 1 & 2 respectively.

If both values are set to 0 then the software position limits are and the modulo is called. The dive shall wrap around the maximum range of the position register (2^{31}).

The **Object 0x607B** cannot be set while in OPERATION ENABLED or QUICK STOP state.

- Object description:

Attributes	0x607B
Name	Position range limit
Object code	ARRAY
Data type	INTEGER32
Category	Mandatory

- Entry description:

Sub-index	0
Description	Number of entries
Entry category	Mandatory
Access	Read only
PDO mapping	No
Value range	2
Default value	2

Sub-index	1
Description	Min position range limit
Entry category	Mandatory
Access	Read/write
PDO mapping	No
Value range	-2^{31}…$(2^{31})-1$
Default value	0

Sub-index	2
Description	Max position range limit
Entry category	Mandatory
Access	Read/write
PDO mapping	No
Value range	-2^{31}…$(2^{31})-1$
Default value	0

12.5. Object 0x607D: Software position limit

This object contains the sub-parameters *min position limit* and *max position limit,* which define the absolute position limits for the *position demand value* and the *position actual value*. Every new *target position* is checked against these limits. The position limits are specified in position units (same as *target position*). If the feedback position detect the Software Position limit the drive shall stop and the internal Limit bit in the Status word is set to 1 (bit 11).

If the drive is enabled via the Control word while the actual position is outside of the position limit boundaries (e.g. Object 0x607D.1= -10,000 and 0x6064 = -12,000), then the motion is permitted toward the boundaries (e.g. 0x607A = -9000) but is restricted to the other side (e.g. 0x607A = -13,000).

When the drive is powered up **VL[3]** & **VH[3]** are set to sub index 1 & 2 respectively

The value of the software position limit is reflected in the **VH[3]** and **VL[3]** commands, to which the range and restrictions are submitted . See the Gold drive Command Reference Manual).Object description:

Attributes	0x607D
Name	Software position limit
Object code	ARRAY
Data type	INTEGER32
Category	Mandatory

- Entry description:

Sub-index	0
Description	Number of entries
Entry category	Mandatory
Access	Read only
PDO mapping	No
Value range	2
Default value	2

Sub-index	1
Description	Min position limit
Entry category	Mandatory
Access	Read/write
PDO mapping	No
Value range	-2^{31} - $(2^{31})-1$
Default value	0

Sub-index	2
Description	Max position limit
Entry category	Mandatory
Access	Read/write
PDO mapping	No
Value range	$-(2^{31})$... $(2^{31})-1$
Default value	0

12.6. Object 0x607F: Max Profile Velocity

The max profile velocity is the maximum speed allowed in either direction during a profiled move. It is given in the same units as profile velocity.

The value of this object is limited internally to the maximum allowed velocity as reflected in **VH[2]**.

- Object description:

Attributes	0x607F
Name	Max profile velocity
Object code	VAR
Data type	UNSIGNED32
Category	Optional

- Entry description:

Access	Read/write
PDO mapping	No
Value range	0….(2^{31})-1
Default value	2e9

12.7. Object 0x6080: Max motor speed

The max motor speed is the maximum speed allowed for the motor. If the value is <0 than the max speed is assumed (2,147,483,647 cnt/sec). Otherwise the RPM value from the user is converted to counts per second according to the motor resolution (CA[18]). The value is used to saturate the speed command to the speed controller. Note that the motor actual speed can overshoot this value

Note: **For compatibility reason the object can be set to a negative value which means that the object does not limit the speed.**

- Object description:

Attributes	0x6080
Name	Max profile velocity
Object code	VAR
Data type	SIGNED32
Category	Optional

- Entry description:

Access	Read/write
PDO mapping	No
Value range	0xFFFFFFFF 0x7FFFFFFF
Default value	0xFFFFFFFF

12.8. Object 0x6081: Profile velocity

This object is the velocity normally attained at the end of the acceleration ramp during a profiled position move and is valid for both directions of motion. The *profile velocity* is given in user-defined speed units. It is converted to position increments per second using the *velocity encoder factor*.

When **BG** command is set, the object gets the value of **SP** command.

The value and default value of the profile velocity is reflected in the SP command, to which the range and restrictions are submitted, see the Gold Drive Command Reference Manual.

Note that if the distance is too short, the actual velocity will be lower than the Profile velocity. In such cases a triangle profile is performed.

- Object description:

Attributes	0x6081
Name	Profile velocity
Object code	VAR
Data type	UNSIGNED32
Category	Optional

- Entry description:

Access	Read/write
PDO mapping	Yes
Value range	0 - (2^{32})-1
Default value	2147483647 (from firmware version 1.1.11.0)

12.9. Object 0x6082: End velocity

The *end velocity* defines the velocity that the drive will have upon reaching the target position in pp mode. Normally, the drive stops at the target position; that is, the *end velocity* = 0. The *end velocity* is given in the same units as *Profile Velocity* **0x6081**.

When **BG** command is set, the object obtains the value of the **FS** command.

- Object description:

Attributes	0x6082
Name	End velocity
Object code	VAR
Data type	UNSIGNED32
Category	Optional

- Entry description:

Access	Read/write
PDO mapping	Yes
Value range	0 - (2^{32})-1
Default value	0

12.10. Object 0x6083: Profile acceleration

The *profile acceleration* defines the acceleration limits for the pp and pv modes. The object is given in user-defined acceleration units. It is converted to position increments per second^2 using the normalizing factors. When the **BG** command is set the object gets the value of the **AC** command

The value of the profile acceleration is reflected in the **AC** command, to which the range and restrictions are submitted. Refer to the Gold Drive Command Reference Manual.

- Object description:

Attributes	0x6083
Name	Profile acceleration
Object code	VAR
Data type	UNSIGNED32
Category	Optional

- Entry description:

Access	Read/write
PDO mapping	Yes
Value range	1…(2^{32})-1
Default value	2147483647 (from firmware version 1.1.11.0)

12.11. Object 0x6084: Profile deceleration

The *profile deceleration* defines the deceleration limits for the profile position and profile velocity modes. The object is given in the same units as profile acceleration. If the *end velocity* (**Object 0x6082**) is different than 0, which means that the drive will not decelerate to full stop, this object is not valid and the profiled deceleration is considered to be similar to the profiled acceleration. When the **BG** command is set the object gets the value of the **DC** command.

The value of the profile deceleration is reflected in the **DC** command, to which the range and restrictions are submitted. Refer to the Gold Command Reference Manual.

- Object description:

Attributes	0x6084
Name	Profile deceleration
Object code	VAR
Data type	UNSIGNED32
Category	Optional

- Entry description:

Access	Read/write
PDO mapping	No
Value range	1…(2^{32})-1
Default value	2147483647 (from firmware version 1.1.11.0)

12.12. Object 0x6085: Quick stop deceleration

The *quick stop deceleration* is the deceleration used to stop the motor if the Quick Stop command is given and the *quick stop option code* (**0x605A**), is set to 2. When the **BG** command is set the object gets the value of the **SD** command.

The *quick stop deceleration* is given in the same units as the *profile acceleration*.

- Object description:

Attributes	0x6085
Name	Quick stop deceleration
Object code	VAR
Data type	UNSIGNED32
Category	Optional

- Entry description:

Access	Read/write
PDO mapping	Yes
Value range	1…(2^{32})-1
Default value	2147483647 (from firmware version 1.1.11.0)

12.13. Object 0x6086: Motion Profile Type

This object is used to select the type of motion profile used to perform a profile move.

Note: **In Profile position and Profile Velocity modes the SF command can be used to smooth the motion**

- Object description:

Attributes	0x6086
Name	Motion profile type
Object code	VAR
Data type	INTEGER16
Category	Mandatory

- Entry description:

Access	Read/write
PDO mapping	No
Value range	0 - 65535
Default value	0

- Data description:

Value	Description
-32,768…-1	Manufacturer specific, reserved
0	Linear ramp (trapezoidal profile)
1	Not supported
2	Not supported
3	Supported for compatibility
4…32,767	Reserved

12.14. Object 0x60C5: Max Acceleration

This object defines the configured maximal acceleration. It is used to limit the profile acceleration that was set as part of the profile position mode set point. The value is measured in user-defined acceleration physical units.

The actual acceleration limit is the minimum value between the max acceleration and the **SD** command

- Object description:

Attributes	0x60C5
Name	Max acceleration
Object code	VAR
Data type	INTEGER32
Category	Mandatory

- Entry description:

Access	Read/write
PDO mapping	No
Value range	1…(2^{31})-1
Default value	2147483647 (from firmware version 1.1.11.0)

12.15. Object 0x60C6: Max Deceleration

This object defines the configured maximal deceleration. It is used to limit the profile deceleration that was set as part of the profile position mode set point. The value is measured in user-defined acceleration physical units.

The actual deceleration limit is the minimum value between the max deceleration and the **SD** command.

- Object description:

Attributes	0x60C6
Name	Max deceleration
Object code	VAR
Data type	UNSIGNED32
Category	Mandatory

- Entry description:

Access	Read/write
PDO mapping	No
Value range	1…(2^{31})-1
Default value	2147483647 (from firmware version 1.1.11.0)

12.16. Functional Description

The set-points setting is controlled by the timing of the new set-point bit and the *change set immediately* bit in the *Controlword* as well as the set-point acknowledge bit in the *Statusword*.

Single set-point (*change set immediately* is set to 1)	The drive immediately processes the new received *target position* that results in a new motion . When reaching the *target position* of processing set-point, the velocity gets to zero and target reached bit is set to 1. The drive is ready to receive a new set point. For details refer to section 0
Set of set-points (*change set immediately* is set to 0)	Buffering of set-points is available. The drive has four set-point buffers. There are two options: • *Controlword* bit 9 (blended) is set to 1 - velocity of the drive is not normally reduced to zero after achieving a set-point position. • *Controlword* bit 9 (blended) is set to 0 - velocity of the drive is reduced to zero after achieving a set-point position. In both cases *target reached* bit is set when the last set point is reached. For details refer to section 12.16.3 onwards

The new set-point , change set immediately bits in the *Controlword*, and set-point acknowledge bit in *Statusword* define a handshake request-response mechanism.

The sequence of a setting and activating new set-point is:

1. The host sends the trajectory data.
2. The host validate the data and initiates the motion by setting the new set- point bit.
3. The drive acknowledges reception and buffering of the new data by setting set-point acknowledge bit.
4. The motion begins.
5. The host resets the new set-point bit. If the drive can accept more set points, the set-point acknowledge bit resets. (Note that this operation can be forced automatically via object 0x60F2)
6. Unless interrupted by a change set immediately bit, the next trajectory, if acknowledged, is executed. The target reached bit in the Status word will be set only after all set-points were processed.

Figure 12-3 below represents using the new set-point , change set immediately and set-point acknowledge bits for request-response mechanism.

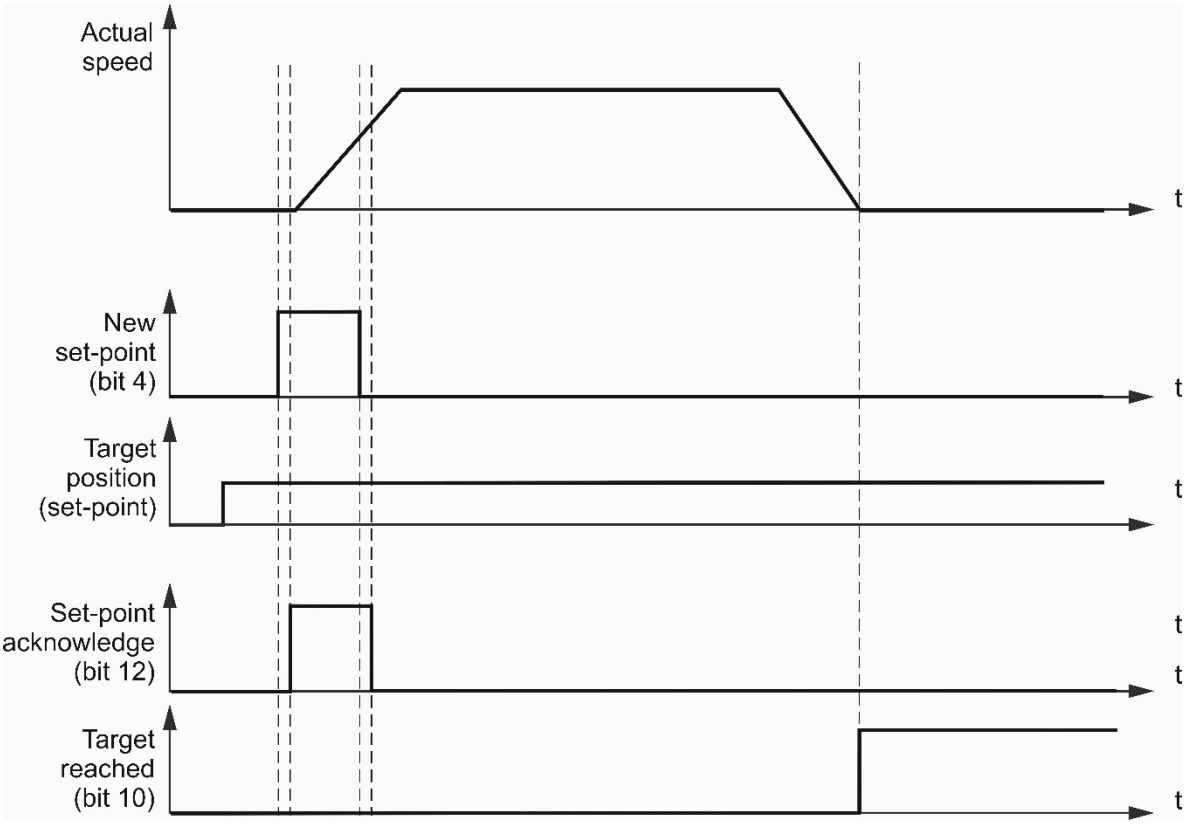


Figure 12-3: Request-response mechanism

The detailed descriptions of the drive`s behavior in the cases of:

- Single set-point
- Buffered set-point, non-blended motion
- Buffered set-point, blended motion

are described in the sections 12.16.1 below onwards.

12.16.1. Single Set-Point

The example of two single set-point point (change set immediately bit of *Controlword* is 1)is represented in Figure 12-4. Both set-points are acknowledged immediately when receiving set-point position (bit 12 of *Statusword* reset) and the new motion starts with its set point data. The first set-point is discarded when receiving the new one without setting target reached bit in *Statusword*. This bit is set when the target position of the second set-point is reached.

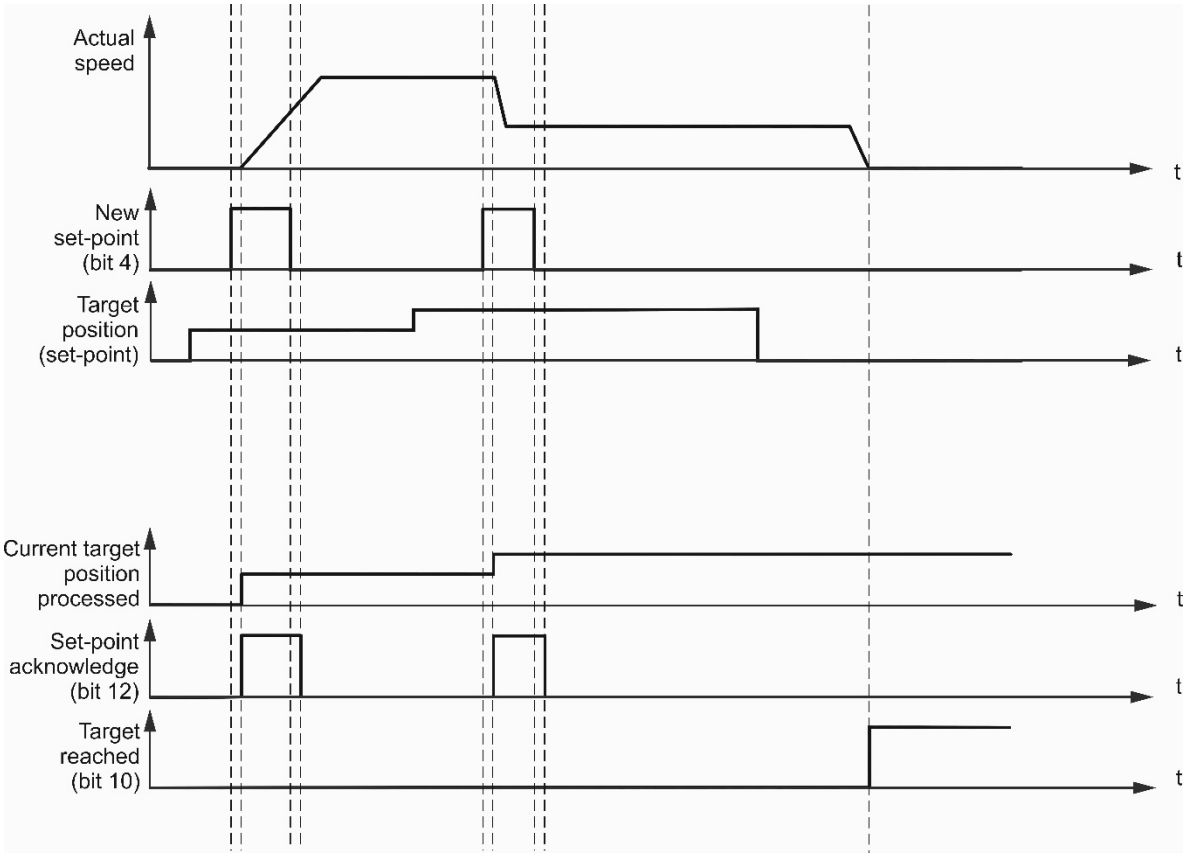


Figure 12-4: Profile position mode of two single set points

The following script simulates two single set points:

```
//Start
td 1 5 0x00 0x01 0x00
// Set pp mode
td 1 10 0x601 0x22 0x60 0x60 0x00 0x01 0x00 0x00 0x00
// Set Prof acceleration (1e6)
td 1 2 0x601 0x22 0x83 0x60 0x00 0x40 0x42 0x0f 0x00
// Set Prof deceleration (1e6)
td 1 2 0x601 0x22 0x84 0x60 0x00 0x40 0x42 0x0f 0x00
// ready 2 switch on
td 1 2 0x601 0x22 0x40 0x60 0x00 0x06 0x00 0x00 0x00
// switch on
td 1 2 0x601 0x22 0x40 0x60 0x00 0x07 0x00 0x00 0x00
// Set PX=0
td 1 2 0x301 0x50 0x78 0x00 0x00 0x00 0x00 0x00 0x00
// start mo=1
td 1 2 0x601 0x22 0x40 0x60 0x00 0x0f 0x00 0x00 0x00
//---------- 1st set-point ----------
```

// Set target position (30000)
```
td 1 2 0x601 0x22 0x7a 0x60 0x00 0x30 0x75 0x00 0x00
```
// Set Speed (50000)
```
td 1 2 0x601 0x22 0x81 0x60 0x00 0x50 0xc3 0x00 0x00
```
// Set 1st "SET POINT" absolute, immediate, non-blended motion.
```
td 1 10 0x601 0x22 0x40 0x60 0x00 0x3f 0x00 0x00 0x00
```
//clear SP
```
td 1 1 0x601 0x22 0x40 0x60 0x00 0x0f 0x00 0x00 0x00

delay 200
```

//---------- 2nd set-point ----------
// Set target position (100000)
```
td 1 2 0x601 0x22 0x7a 0x60 0x00 0xa0 0x86 0x01 0x00
```
// Set Speed 3000
```
td 1 2 0x601 0x22 0x81 0x60 0x00 0xb8 0x0b 0x00 0x00
```
// Set 2nd "SET POINT" absolute Immediate, non-blended motion.
```
td 1 10 0x601 0x22 0x40 0x60 0x00 0x3f 0x00 0x00 0x00
```
//clear SP
td 1 1 0x601 0x22 0x40 0x60 0x00 0x0f 0x00 0x00 0x00

12.16.2. Buffered Set-Point

If the change set immediately bit of the *Controlword* is set to 0, the buffering of set-points is available. The drive has four buffers the set-points. When the first set-point is processed, the drive can receive additionally three set-points that will be processed when the preceding was reached.

Note that the target reached bit will be set only after the last set-point was reached. The example of four set-points handling is presented in Figure 12-5. Set-point acknowledge bit is set to 1 when new set-point is buffered and it is reset when new set point bit is reset and if one buffer is still available. Target reached bit is set when the last set point is reached.

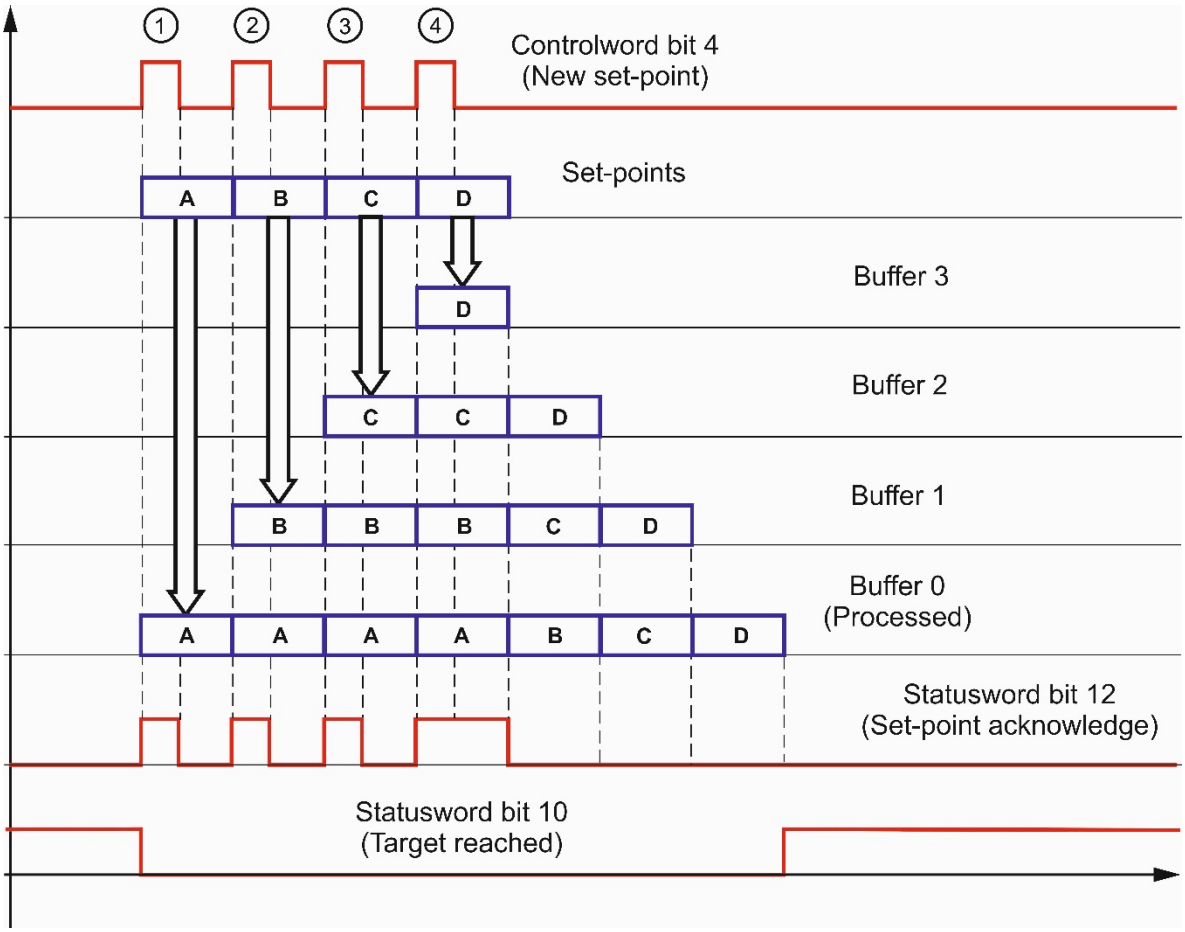


Figure 12-5: Set-point handling for four set-points

12.16.3. Buffered Set-Point, non-blended motion

If the change set immediately bit of the *Controlword* is set to 0, one set-point is still in progress and a new one is activated with the blended bit reset, then the next set point will be processed after the previous one was completed.

Note, that in this case the previous motion is not discarded. The drive reaches target position and comes to a complete stop without setting the target reached bit. After which a new motion is started with the next buffered set-point. The target reached bit is set at the end of the new profile. This example is represented in Figure 12-6.

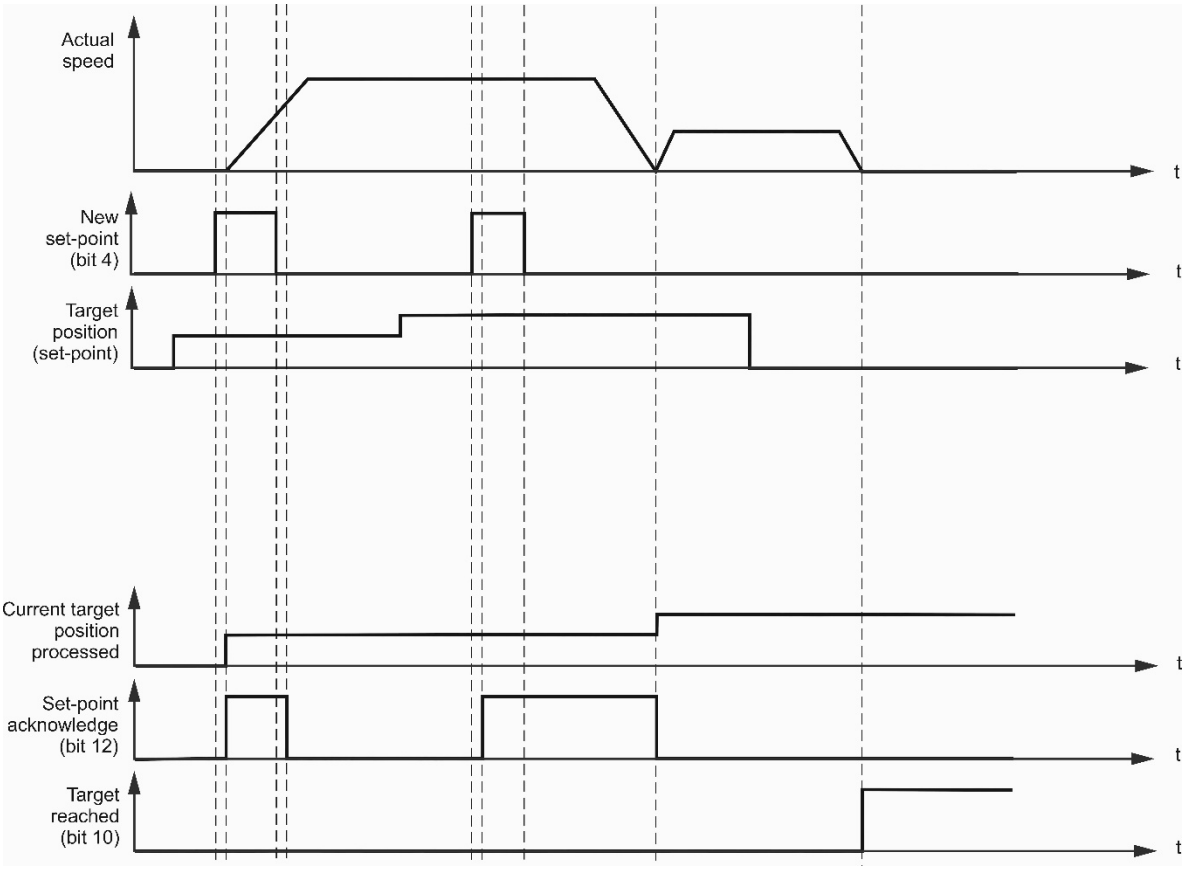


Figure 12-6: Profile Poistion mode buffered set points, non-blended motion

The following script emulates the use of non-blended set of set points.

```
//Start
td 1 5 0x00 0x01 0x00

// Set pp mode
td 1 10 0x601 0x22 0x60 0x60 0x00 0x01 0x00 0x00 0x00
// Set Prof acceleration (1e6)
td 1 2 0x601 0x22 0x83 0x60 0x00 0x40 0x42 0x0f 0x00
// Set Prof deceleration (1e6)
td 1 2 0x601 0x22 0x84 0x60 0x00 0x40 0x42 0x0f 0x00

// ready 2 switch on
td 1 2 0x601 0x22 0x40 0x60 0x00 0x06 0x00 0x00 0x00
// switch on
```

```
td 1 2 0x601 0x22 0x40 0x60 0x00 0x07 0x00 0x00 0x00
// Set PX=0
td 1 2 0x301 0x50 0x78 0x00 0x00 0x00 0x00 0x00 0x00


// start mo=1
td 1 2 0x601 0x22 0x40 0x60 0x00 0x0f 0x00 0x00 0x00

//------------------- 1st set-point ----------------------------------------------------
// Set target position (30000)
td 1 2 0x601 0x22 0x7a 0x60 0x00 0x30 0x75 0x00 0x00
// Set Speed (50000)
td 1 2 0x601 0x22 0x81 0x60 0x00 0x50 0xc3 0x00 0x00
// Set "SET POINT" absolute, No immediate, non-blended motion.
td 1 10 0x601 0x22 0x40 0x60 0x00 0x1f 0x00 0x00 0x00
//clear SP
td 1 1 0x601 0x22 0x40 0x60 0x00 0x0f 0x00 0x00 0x00

delay 200

//------------------- 2nd set-point ----------------------------------------------------
// Set target position (100000)
td 1 2 0x601 0x22 0x7a 0x60 0x00 0xa0 0x86 0x01 0x00
// Set Speed 10000
td 1 2 0x601 0x22 0x81 0x60 0x00 0x10 0x27 0x00 0x00
// Set "SET POINT" absolute NO Immediate, non-blended motion.
td 1 10 0x601 0x22 0x40 0x60 0x00 0x1f 0x00 0x00 0x00
//clear SP
td 1 1  0x601 0x22 0x40 0x60 0x00 0x0f 0x00 0x00 0x00
```

The following example shown in Figure 12-7 introduces a set of five set-points for non-blended buffered mode where the 5th set-point is ignored by the drive.

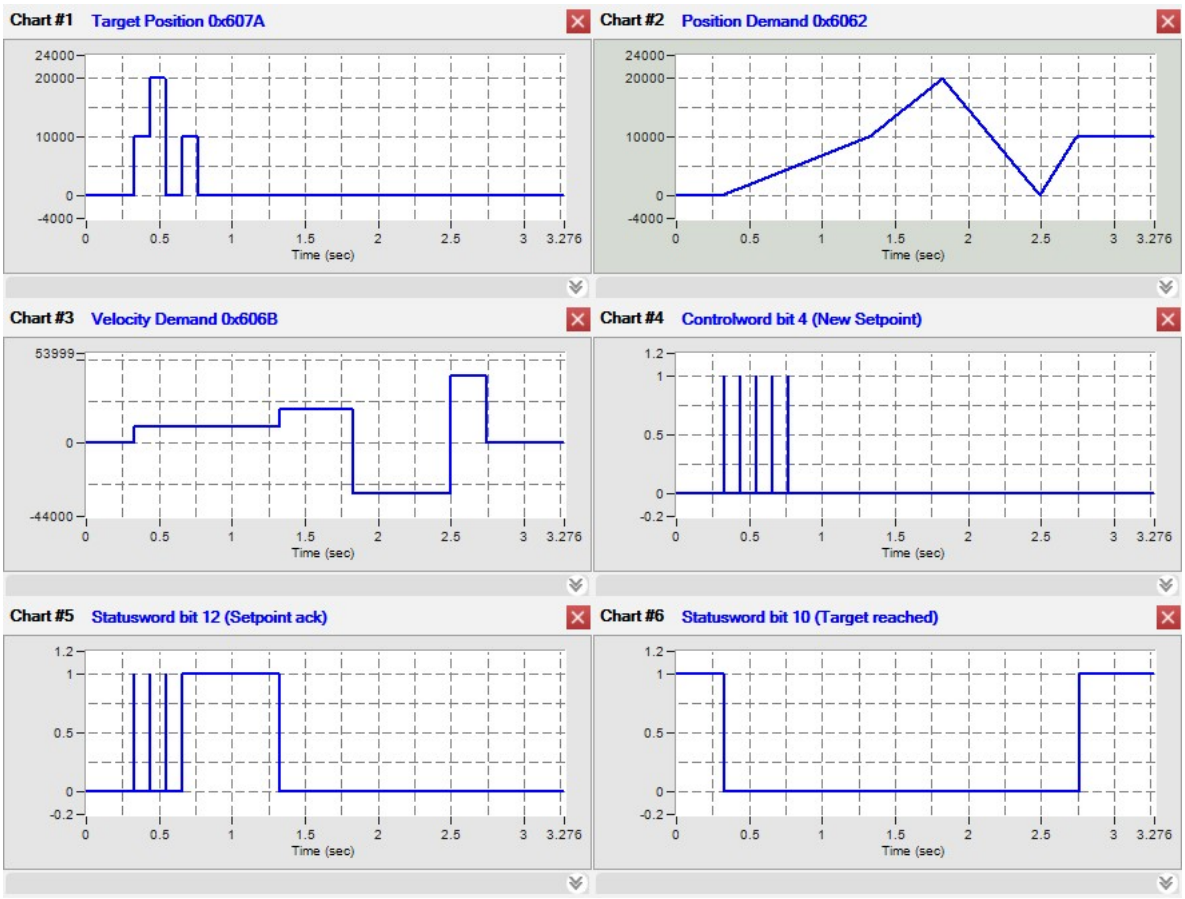


Figure 12-7: Five set-points (the 5th ignored)

Set points 2,3,4 are sent to drive while the 1-st one is processed. After the acknowledgement of the drive, set-points 2,3,4 are waiting in buffer. The 5th is ignored, because there is no space in the buffer. At the end of the profile, and after set-point 4 was processed and reached, the axis gets to position defined by 4th set-point and target reached bit is set.

In the example the 4th set-point position is 10,000 while the 5th is 0. The Position Demand shows that when the target reached bit was set, the position is 10,000. The position 0, defined by set-point 5 is ignored.

When receiving 1th, 2nd and 3rd set-points, the Set-point acknowledge bit is set. In these set-points, when the host resets new set-point bit, the set-point acknowledge bit is reset to 0. This is done since there is a space for next set-point in the buffer. But when receiving the 4th set-point, Set-point acknowledge bit is set and then remains in high state since there is no space in the buffer. The *set-point acknowledge* bit is cleared when the 1st set point process is finished and the place in the set-point buffer is released.

The following is the script used to execute the example above:

```
//Start
td 1 1 0x00 0x01 0x00
// reset any fault
td 1 2 0x601 0x22 0x40 0x60 0x00 0x80 0x00 0x00 0x00
```

```
// ready 2 switch on
td 1 10 0x601 0x22 0x40 0x60 0x00 0x06 0x00 0x00 0x00
delay 2000
// Set pp mode
td 1 10 0x601 0x22 0x60 0x60 0x00 0x01 0x00 0x00 0x00
// Set PX=0
td 1 2 0x301 0x50 0x78 0x00 0x00 0x00 0x00 0x00 0x00

delay 100
// ready 2 switch on
td 1 10 0x601 0x22 0x40 0x60 0x00 0x06 0x00 0x00 0x00
// switch on
td 1 10 0x601 0x22 0x40 0x60 0x00 0x07 0x00 0x00 0x00
// start mo=1
td 1 2 0x601 0x22 0x40 0x60 0x00 0x0f 0x00 0x00 0x00
delay 1000
//---------- 1 st set-point ----------------------
// Set target position (10000)
td 1 2 0x601 0x22 0x7a 0x60 0x00 0x10 0x27 0x00 0x00
// Set Speed (10000)
td 1 2 0x601 0x22 0x81 0x60 0x00 0x10 0x27 0x00 0x00
// Set "SET POINT" absolute not Immediate.
td 1 1 0x601 0x22 0x40 0x60 0x00 0x1f 0x00 0x00 0x00
// Set Clear "SET POINT
td 1 1 0x601 0x22 0x40 0x60 0x00 0x0f 0x00 0x00 0x00
delay 100

//---------- 2 nd set-point ----------------------
// Set target position (20000)
td 1 2 0x601 0x22 0x7a 0x60 0x00 0x20 0x4e 0x00 0x00
// Set Speed (20000)
td 1 2 0x601 0x22 0x81 0x60 0x00 0x20 0x4e 0x00 0x00
// Set "SET POINT" absolute not Immediate.
td 1 1 0x601 0x22 0x40 0x60 0x00 0x1f 0x00 0x00 0x00
// Set Clear "SET POINT"
td 1 1 0x601 0x22 0x40 0x60 0x00 0x0f 0x00 0x00 0x00
delay 100

//---------- 3 rd set-point ----------------------
// Set target position (0)
td 1 2 0x601 0x22 0x7a 0x60 0x00 0x00 0x00 0x00 0x00
// Set Speed (30000)
td 1 2 0x601 0x22 0x81 0x60 0x00 0x30 0x75 0x00 0x00
// Set "SET POINT" absolute not Immediate.
td 1 1 0x601 0x22 0x40 0x60 0x00 0x1f 0x00 0x00 0x00
// Set Clear "SET POINT"
td 1 1 0x601 0x22 0x40 0x60 0x00 0x0f 0x00 0x00 0x00
delay 100
//---------- 4 th set-point ----------------------
// Set target position (10000)
td 1 2 0x601 0x22 0x7a 0x60 0x00 0x10 0x27 0x00 0x00
// Set Speed (40000)
td 1 2 0x601 0x22 0x81 0x60 0x00 0x40 0x9c 0x00 0x00
// Set "SET POINT" absolute not Immediate.
td 1 1 0x601 0x22 0x40 0x60 0x00 0x1f 0x00 0x00 0x00
// Set Clear "SET POINT"
td 1 1 0x601 0x22 0x40 0x60 0x00 0x0f 0x00 0x00 0x00
delay 100
//---------- 5 th set-point ----------------------
// Set target position (0)
td 1 2 0x601 0x22 0x7a 0x60 0x00 0x00 0x00 0x00 0x00
// Set Speed (50000)
td 1 2 0x601 0x22 0x81 0x60 0x00 0x50 0xc3 0x00 0x00
// Set "SET POINT" absolute not Immediate.
td 1 1 0x601 0x22 0x40 0x60 0x00 0x1f 0x00 0x00 0x00
// Set Clear "SET POINT"
td 1 1 0x601 0x22 0x40 0x60 0x00 0x0f 0x00 0x00 0x00
delay 100
```

12.16.4. Buffered Set-Point, Blended

If the change set immediately bit of the *Controlword* is set to 0, one set-point is still in progress and a new one is activated with *blended* bit set, the previous motion is not discarded and must be completed.

The drive reaches target position with defined profile speed without decreasing it to zero and without setting the target reached bit. After that, a new motion is started with its target position and speed. The target reached bit is set on the end of the new profile.

The example is represented in Figure 12-8.

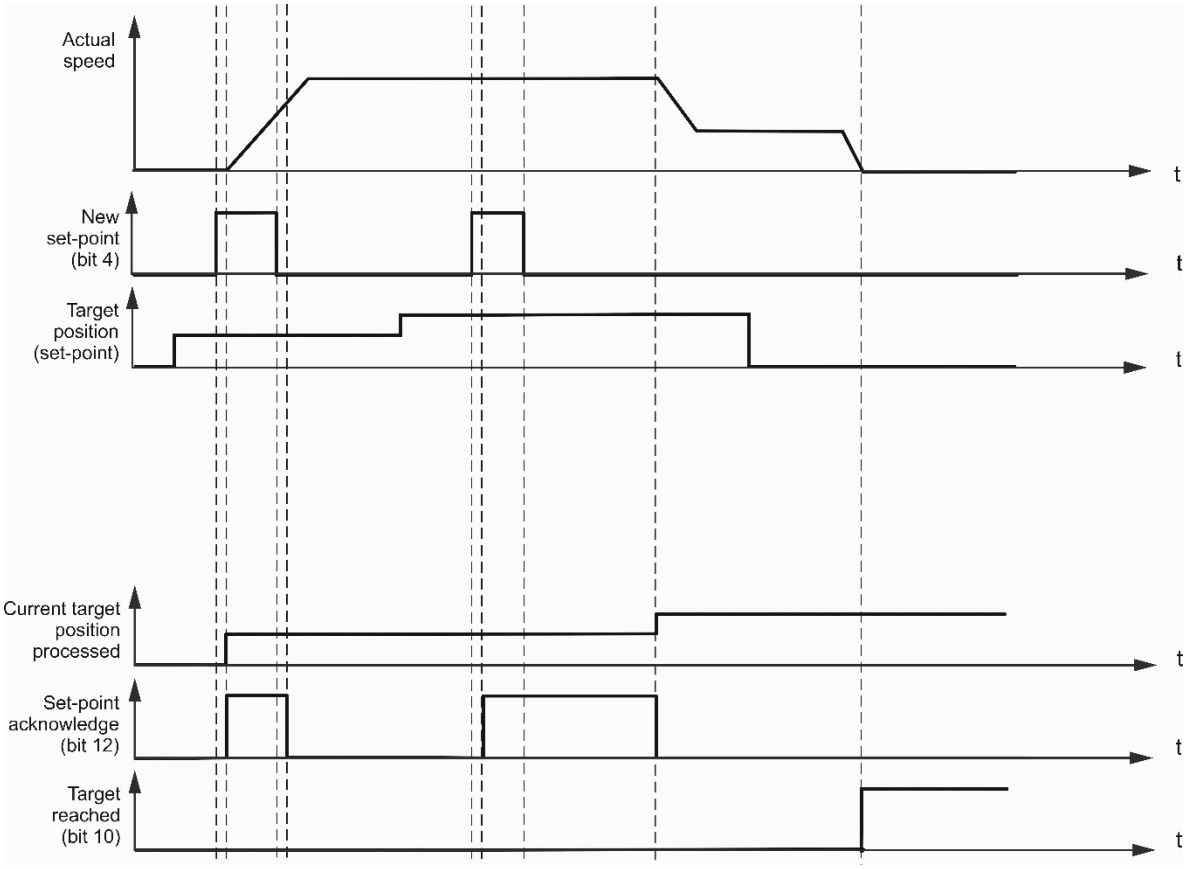


Figure 12-8: Profile Position mode, buffered set points, blended motion

Following is the script that is used in the CAN Analyzer to execute the example described motion:

```
//Start
td 1 5 0x00 0x01 0x00

// Set pp mode
td 1 10 0x601 0x22 0x60 0x60 0x00 0x01 0x00 0x00 0x00
// Set Prof acceleration (1e6)
td 1 2 0x601 0x22 0x83 0x60 0x00 0x40 0x42 0x0f 0x00
// Set Prof deceleration (1e6)
td 1 2 0x601 0x22 0x84 0x60 0x00 0x40 0x42 0x0f 0x00

// ready 2 switch on
td 1 2 0x601 0x22 0x40 0x60 0x00 0x06 0x00 0x00 0x00
// switch on
```

```
td 1 2 0x601 0x22 0x40 0x60 0x00 0x07 0x00 0x00 0x00
// Set PX=0
td 1 2 0x301 0x50 0x78 0x00 0x00 0x00 0x00 0x00 0x00

// start mo=1
td 1 2 0x601 0x22 0x40 0x60 0x00 0x0f 0x00 0x00 0x00

//---------- 1 st set-point ----------------------
// Set target position (30000)
td 1 2 0x601 0x22 0x7a 0x60 0x00 0x30 0x75 0x00 0x00
// Set Speed (50000)
td 1 2 0x601 0x22 0x81 0x60 0x00 0x50 0xc3 0x00 0x00

// Set "SET POINT" absolute, No immediate, blended motion.
td 1 10 0x601 0x22 0x40 0x60 0x00 0x1f 0x02 0x00 0x00
//clear SP
td 1 1 0x601 0x22 0x40 0x60 0x00 0x0f 0x00 0x00 0x00
delay 200

//---------- 2 nd set-point ----------------------
// Set target position (100000)
td 1 2 0x601 0x22 0x7a 0x60 0x00 0xa0 0x86 0x01 0x00
// Set Speed (20000)
td 1 2 0x601 0x22 0x81 0x60 0x00 0x20 0x4e 0x00 0x00
// Set "SET POINT" absolute NO Immediate, blended motion.
td 1 10 0x601 0x22 0x40 0x60 0x00 0x1f 0x02 0x00 0x00
//clear SP
td 1 1  0x601 0x22 0x40 0x60 0x00 0x0f 0x00 0x00 0x00
```

Chapter 13: Interpolated Position

13.1. General

Object	Definition
0x60C0	Interpolation sub mode select
0x60C1	Interpolation data record
0x60C2	Interpolation time period
0x60C3	Interpolation sync definition
0x60C4	Interpolation data configuration

. In ip mode, the control device (master) sends the target position (with optional target velocity too) to the Gold drive, every interpolation period using synchronous or asynchronous PDO. Every target is considered as a small step from the previous target and the drive performs interpolation between these two steps namely namely the interpolation period.

Interpolated Position mode allows a host controller to transmit a stream of interpolated data with an explicit time reference to the drive. The Elmo drive supports an input buffer that allows the interpolated data to be sent in bursts rather than continuously in real time. The actually available and the maximum size of the input buffer can be requested by the host using the *interpolation data configuration*. The buffer size is the number of *interpolation data records* that may be sent to a drive to fill the input buffer; it is not the size in bytes.

The interpolation algorithm is defined in the *interpolation sub mode select*. Linear interpolation is the default interpolation method. For each interpolation cycle, the drive calculates a *position demand value* by interpolating interpolation data over a period of time. Only linear interpolation is implemented.

The ip mode is based on Synchronization mechanism allowing the host to synchronized several axes to the same set-point timing. More details refer to 13.12 Motion Synchronization.

The interpolation data buffer may be implemented as a FIFO or a ring. The definition of a valid data record for each type of buffer is:

- For the FIFO implementation, every new set-point is placed at the end of the buffer.
- For the Ring implementation, the host can set-point at any place by changing the buffer pointer. The drive reads the set-points from the beginning as an internal read pointer.

Limit functions of speed acceleration deceleration and position are applied to the interpolation data.

This chapter describes objects and protocol features related to ip mode. The Elmo Gold Drive Administrative Guide contains the ip mode full description. The structure of ip mode is presented in Figure 13-1.

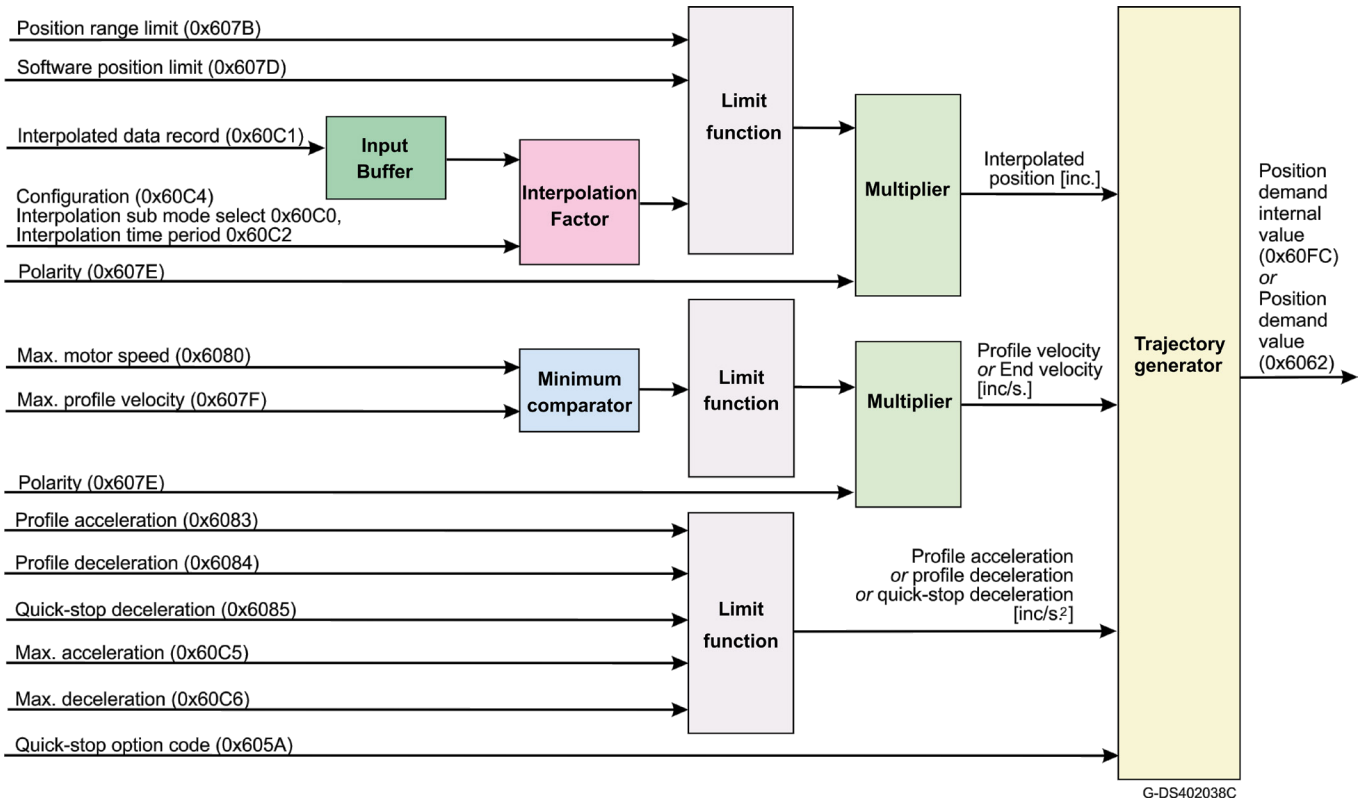


Figure 13-1: Interpolated mode structure

13.2. Internal states

Internal states of ip mode are: *interpolation inactive* and *interpolation active*. Figure 13-2 presents the states and transitions between them.

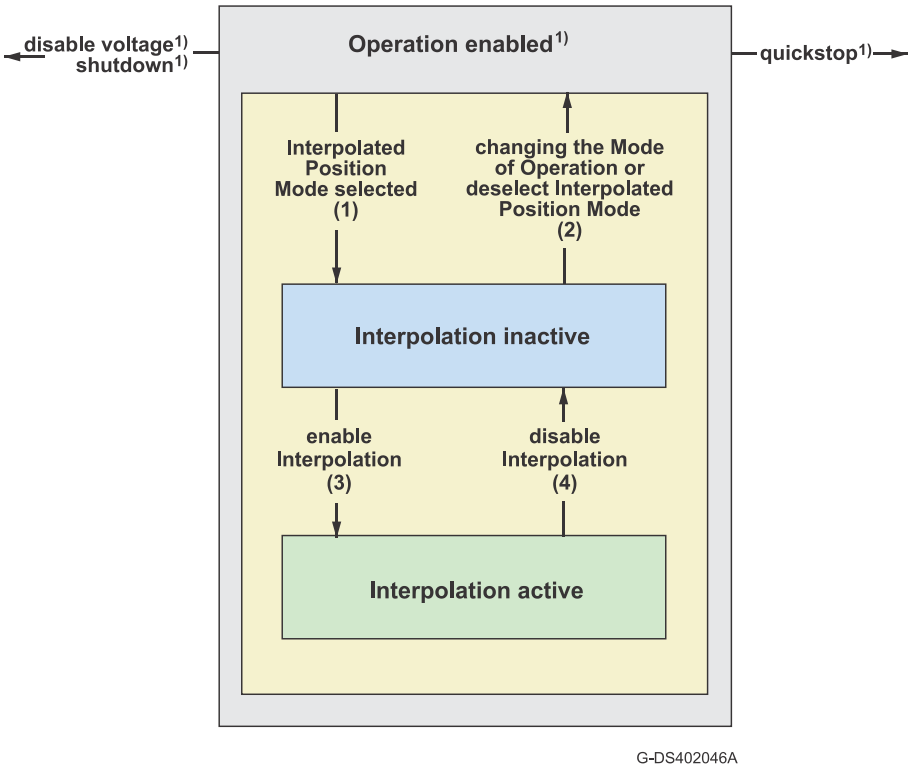


Figure 13-2: ip mode internal states

Note: 1)	*Operation enabled* is the state of a drive. Refer to Figure 6-3 *State machine block diagram. Disable voltage*, *Shutdown*, *Quickstop* are commands from the control device that can be sent via *Controlword*.
(1)	Transition performed by settings object *Operating Mode* 0x6060 to 7
(2)	Transition performed by settings object 0x6060 to value not equals 7
(3)	Transition performed by setting bit 4 of *Controlword* to 1
(4)	Transition performed by setting bit 4 of *Controlword* to 0

13.2.1. Interpolation inactive

The state entered when the device is in OPERATION ENABLED state and Interpolated Position mode (available in CANopen only) is selected and displayed (object 0x6061). The drive unit accepts input data and buffers it for interpolation calculations, but does not move the axles.

13.2.2. Interpolation active

The interpolation starts when the device is in Operation Enable state in ip mode and bit 4 in the Controlword is set to 1. Note that until the first set-point arrives, the drive interpolates the actual position. This means that there will be no underflow indication before the arrival of the first set-point. Nevertheless it is recommends that the host will send a number of SYNC messages and

set-points (via **Object 0x60C1**) allowing the drive to synchronize and arrange the set-point buffers, prior to setting the Enable Interpolation bit (bit 4 in the Controlword).

13.2.3. Set-point Buffer Reset

The interpolation data buffer is reset in the following cases:

- Shutdown the motor
- Change of operating mode
- Change interpolated data configuration object 0x60C4
- Change interpolation sub mode select object 0x60C0
- Entering *interpolation inactive* state due to underflow emergency

13.3. Controlword

Controlword of ip mode is presented in Table 13-1.

Bit	Function
0	Switch on, see the description in **Object 0x6040** *Controlword*
1	Enable voltage, see the description in **Object 0x6040***Controlword*
2	Quick stop, see the description in **Object 0x6040** *Controlword* **Note that in this state, the Interpolation no longer active.**
3	Enable operation (set motor on) , see the description in **Object 0x6040** *Controlword*
4	Enable Interpolation
5	Reserved, 0
6	Reserved, 0
7	Fault reset, see the description in **Object 0x6040** *Controlword*
8	Halt **Note that in this state, the Interpolation no longer active.**
9	Reserved, 0
10	Reserved, 0 Reserved, 0
11	Reserved, 0 Reserved, 0
12	Reserved, 0 Reserved, 0
13	Reserved, 0 Reserved, 0

Table 13-1 ip Mode *Controlword*

If the interpolation is interrupted by setting bit 4 from 1 to 0, the motion is stopped in accordance with the *halt options code* (object 0x605D).

In case of a Halt, the drive stops the interpolation and stops the motor according to *halt options code* (object 0x605D).

If the motor is stopped due to an internal fault or *Controlword* command, the interpolation is disabled, even if bit 4 is 1. Interpolation can be enabled again only after the device enters the OPERATION_ENABLE state and bit 4 is set to 1.

13.4. Statusword

The ip mode *Statusword* is presented in Table 13-2 and Table 13-3.

Bit	Function
0	Ready to switch on, see the description in Object 0x6041 *Statusword*
1	Switched on, see the description in Object 0x6041 *Statusword*
2	Operation enabled, see the description in Object 0x6041 *Statusword*
3	Fault, see the description in Object 0x6041 *Statusword*
4	Voltage enabled, see the description in Object 0x6041 *Statusword*
5	Quick stop, see the description in Object 0x6041 *Statusword*
6	Switch on disabled, see the description in Object 0x6041 *Statusword*
7	Warning, see the description in Object 0x6041 *Statusword*
8	Manufacturer specific, reserved, always 0
9	Remote, see the description in Object 0x6041 *Statusword*
10	Target reached
11	Internal limit active, see the description in Object 0x6041 *Statusword*
12	Interpolation active
13	Following error
14 - 15	Manufacturer specific, reserved, always 0

Table 13-2 ip Mode *Statusword*

Name	Value	Description
Target reached	0	Halt = 0: *Position* not reached. Halt = 1: Axle decelerates.
	1	Halt = 0: *Position* reached. Halt = 1: Velocity of axle is 0.
Interpolation active	0	Interpolation mode not active.
	1	Interpolation mode active.

Table 13-3 ip Mode *Statusword*, bits 10,12

13.5. Object 0x60C0: Interpolation Sub-Mode Select

This object reflects or changes the actual chosen interpolation mode, selected by the user.

The interpolation sub-modes can be changed only when the interpolated mode is inactive. An attempt to change the object in *interpolation active* state will produce the abort SDO message with ELMO error code 185.

When modifying the interpolation sub mode, a new mapping (if necessary) of object 0x60C1 must be made after the sub mode is modified. Failing to do so may cause unpredictable results.

- Object description:

Attributes	0x60C0
Name	Interpolation sub mode select
Object code	VAR
Data type	INTEGER16
Category	Optional

- Entry description:

Access	Read/write
PDO mapping	No
Value range	-1, 0
Default value	0 (linear interpolation)

- Data description:

Value	Description
-32768..-2	Reserved
-1	Linear interpolation, data record contains target position (0x60c1.1) and target velocity (0x60C1.2)
0	Linear interpolation, data record contains target position (0x60c1.1) only
1…32767	Reserved

13.6. Object 0x60C1: Interpolation Data Record

This object includes the set-points fo the ip mode. . The interpretation of the data words may vary with the different possible interpolation modes as set by 60C0h.

Note **The 0x60C1 object is subjected to synchronization with the drives and host timing. Underflow and overflow mechanism are used to inform the host if the set-point timing do not match the dive interpolation timing. Refer to 13.10 Timeout and Underflow Emergency Message for more details.**

- Object description:

Attributes	0x60C1
Name	Interpolation data record
Object code	ARRAY
Data type	60C0h = -1 : DS-402 PV data record (0x44) – Refer to MAN-G-DS301 60C0h = 0 : INTEGER32 60C0h > 0 : not defined
Category	Optional

- Entry description:

Sub-index	0
Description	Number of entries
Entry category	Mandatory
Access	Read only
PDO mapping	Yes
Value range	2
Default value	2

Sub-index	1
Description	Interpolation data record (target position)
Entry category	Mandatory
Access	Read/Write
PDO mapping	Yes
Value range	-2^{31}...(2^{31})-1
Default value	0

Sub-index	**2**
Description	Interpolation data record (target velocity)
Entry category	Mandatory
Access	Read/Write
PDO mapping	Yes
Value range	-2^{31}...$(2^{31})-1$
Default value	0

13.7. Object 0x60C2: Interpolation time period

This object is used to define the relative time taken between two set points for the interpolation position modes. The interpolation time unit is given in $10^{\text{interpolation time index}}$ seconds.

The interpolation time period can be changed only when the interpolated mode is inactive. An attempt to change the object in *interpolation active* state produces the abort SDO message with ELMO error code 185.

- Object description:

Attributes	0x60C2
Name	Interpolation time period
Object code	RECORD
Data type	Interpolation time period (object 0x80) – Refer to MAN-G-DS301
Category	Optional

- Entry description:

Sub-index	0
Description	Number of entries
Entry category	Mandatory
Access	Read only
PDO mapping	No
Value range	2
Default value	2

Sub-index	1
Description	Interpolation time unit
Entry category	Mandatory
Access	Read/Write
PDO mapping	Yes
Value range	1..255
Default value	1

Sub-index	**2**
Description	Interpolation time index
Entry category	Mandatory
Access	Read/Write
PDO mapping	No
Value range	-6… -2
Default value	-3

13.8. Object 0x60C4: Interpolation data configuration

The interpolation data configuration enables the user to obtain information about the buffer size and set the buffer configuration and strategy. An attempt to change the object in *interpolation active* state produces the abort SDO message with ELMO error code 185.

- Object description:

Attributes	0x60C4
Name	Interpolated data configuration
Object code	RECORD
Data type	Interpolated data configuration record (object 0x81) – Refer to MAN-G-DS301
Category	Optional

- Entry description:

Sub-index	**0**
Description	Number of entries
Entry category	Mandatory
Access	Read only
PDO mapping	No
Value range	6
Default value	6

Sub-index	**1**
Description	Maximum buffer size, it is a number of interpolated data records, not size in bytes
Entry category	Mandatory
Access	Read only
PDO mapping	No
Value range	1 for sub mode 0 16 for sub mode -1
Default value	1

Sub-index	2
Description	Actual buffer size, it is a number of interpolated data records, not size in bytes
Entry category	Mandatory
Access	Read/Write
PDO mapping	No
Value range	1…16 (may be from 1 up to Maximum buffer size)
Default value	1

Sub-index	3
Description	Buffer organization, FIFO or ring buffer
Entry category	Mandatory
Access	Read/Write
PDO mapping	No
Value range	0…1 (asdescribed below)
Default value	0

Sub-index	4
Description	Buffer position
Entry category	Mandatory
Access	Read/Write
PDO mapping	No
Value range	0…15
Default value	0

Sub-index	5
Description	Size of data record
Entry category	Mandatory
Access	Read only
PDO mapping	No
Value range	60C0h = -1 : 8 bytes 60C0h = 0: 4 bytes
Default value	4

Sub-index	**6**
Description	Buffer clear
Entry category	Mandatory
Access	Write only
PDO mapping	No
Value range	0,1
Default value	No

- Type of buffer organization:

Value	Description
0	FIFO buffer
1	Ring buffer
2…255	Reserved

- Description of buffer clear values:

Value	Description
0	• Clear input buffer • Access disabled • Clear all ip data records
1	Enable access to input buffer for drive functions
2…255	Reserved

13.9. Buffer strategies

The contents of the buffer items can only be accessed via the *interpolation data record.* The maximum buffer size is given in object 0x60C4 is used by the host to determine the actual buffer size.

There are two types of buffer that can be defined by object 0x60C4.3 : first-in-first-out (FIFO) structures or ring buffers.

13.9.1. FIFO

If the buffer is organized as FIFO, every new received *interpolation data record* is placed at the end of the queue, and the drive takes the next data record from the top of the queue. When the last item of a data record is stored, the buffer pointer is incremented in order to point to the next buffer position. With this buffer principle, the object *buffer position* has no affect. The FIFO buffer is organized as a cyclic buffer so that after the last buffer entry is updated (entry of *max buffer size)*, the first entry may be available again depending on *actual buffer size*.

13.9.2. Ring buffer

If the buffer is structured as a ring, the host can place an interpolation data record into any valid position in the ring by changing the pointer defined in buffer position. Without changing the buffer position, all data records are written at the same location. The drive reads the next entry out of the buffer by an internal ring pointer. It is set to the first data record with buffer clear, after the reorganization of the input buffer. The user cannot exceed the max buffer

13.10. Timeout and Underflow Emergency Message

The control device is responsible to provide *interpolated data records* with a period equal to the *interpolation time period* in sub mode 0 and in sub mode -1 with *actual buffer size* 1.

The control device is responsible to make sure that the drive buffer is not empty. When interpolated time period elapses and no new set-point is buffered, the drive performs linear extrapolation using the last received interpolated data during interpolation timeout defined by **Object 0x2F75** (see MAN-G-DS301) and returns to normal operating if the new data has been received or releases underflow emergency message.

Note **The underflow mechanism is active only after the first set-point was received by the drive. It is recommended that the host will set the Enable Interpolation bit only after sending several SYNC & set-points (via 0x60C1) messages.**

If the new data has not been received yet, then the motion stops with deceleration defined by *halt option code* **Object 0x605D** and the drive resets bit 12 *Interpolation active* in its *Statusword*. Operating mode remains 7 (*ip mode*). The emergency message can be masked by **Object 0x2F21** *Emergency event mask*. For additional details see the Gold Administrative Guide.

13.11. Overflow Emergency Message

The control device is responsible for not overwriting interpolation data in the drive`s buffer. The drive monitors this event in *Interpolation active* state only. If interpolated data in the drive`s buffer is overwritten by the new one and the previous data did not participate in the interpolation calculation, the overflow emergency message is sent by the drive if it is not masked by **Object 0x2F21**. In this case the motion continues, but it can be instable due to the loss of a set-point. The monitoring overwriting event is not performed when buffer organization is *ring* (**Object 0x60C4.3=1**). For additional details see the Gold drive Administrative Guide.

13.12. Motion Synchronization

The ip mode enables the synchronized motion of multiple axes. The motions of several slave axes are synchronized if they all run in ip mode, and are all invoking the Interpolated Profile motion simultaneously. Synchronization can proceed continuously using the SYNC-Time mechanism.

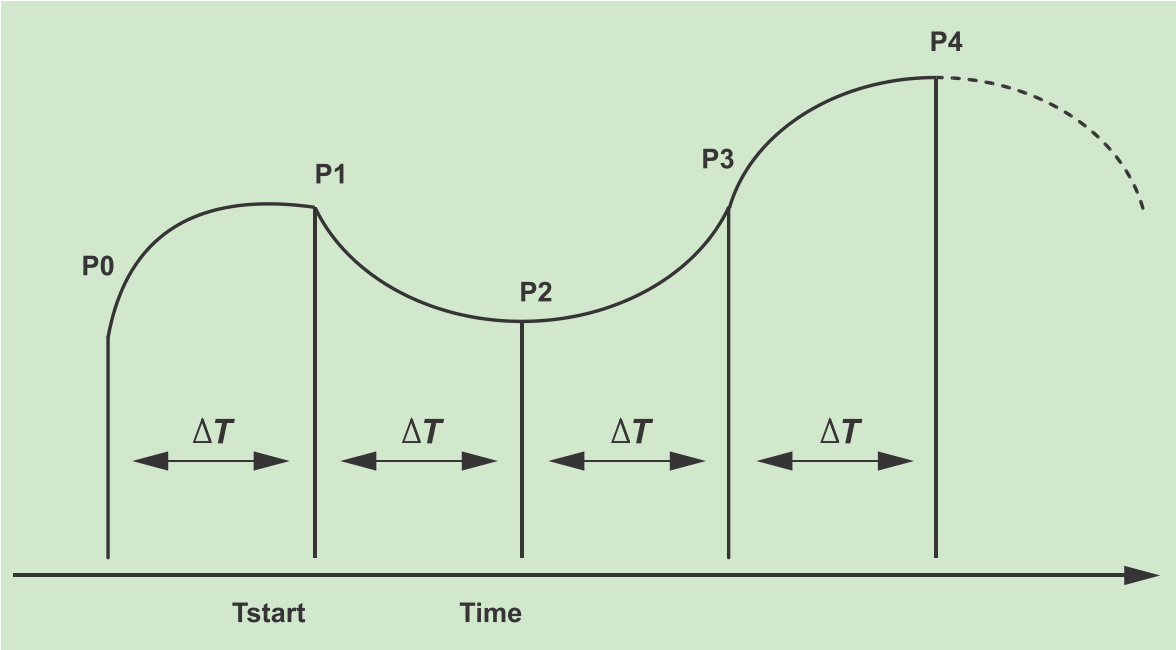
In order to start several axes synchronously, map the *Controlword* to a synchronous RPDO, and then use the mapped *Controlword* to enable interpolation for all axes. Nothing will happen until the next SYNC. Then, all drives will enable interpolated motion at once, setting the SYNC arrival time as the *zero* time of the path specification.

If the axes have been previously synchronized by SYNCs, the moving axes will be relatively synchronized to the precision of microseconds.

Note: **The drives are synchronizing the internal interpolation time according to the arrival of the SYNC and the required interpolation time via Object 0x60C2. It is recommended to send 64 SYNC messages allowing the internal time to be synchronized prior the start of the interpolation by bit 4 of 0x6064.**

13.13. Functional Description

In Interpolated Position mode (available in CANopen only), the drive executes a time-synchronized motion path. The user specifies the value of the reference signal at an initial time, and at fixed-time intervals from then on, as shown in Figure 13-3.



G-DS402047A

Figure 13-3: Interpolated Motion

In Figure 13-3, the time interval ΔT is set by the object 0x60c2, in milliseconds. The data records P0,P1,P2,… (object 0x60C1) to define the motion path data relating to the times

$$T_{start}, T_{start} + \Delta, T_{start} + 2\Delta, \ldots$$

The motion path is synchronized to the CAN microsecond timer, as set and corrected by the SYNC-Time stamp mechanism.

The user must specify the data records P0,P1,P2,…fast enough – at an average rate of at least one record per ΔT. The drive can store up to 16 records in sub mode -1 and one record in sub mode 0. Therefore, in sub mode -1, the path may be programmed in bursts in order to relax the feeding real-time requirements.

To enter ip mode, *Operation mode* object 0x6060 should be used. To monitor the operating mode *Operation mode display* object 0x6061 should be used.

The *Controlword* (0x6040) is used to enable the motor and to move it. The status of the drive can be monitored by the *Statusword* (0x6041).

The *interpolation sub-mode select*, object 0x60C0 determines the type of interpolation performed. Elmo Gold drive supports linear interpolation only.

13.13.1. Sub Mode 0. Linear Interpolation

Linear interpolation requires only the target position specified in data record object 0x60C1. For sub mode 0 the data type is INTEGER32 target position only. The profile velocity at each time point is calculated by finding the difference between the corresponding target position and the target position of the previous point and by dividing this difference with *interpolation time period*.

$$V = [P(n) - P(n-1)]/\Delta T$$

Where:

ΔT	interpolation time period, 0x60C2;
P(n),P(n-1)	interpolated data record, 0x60C1 related to interpolation periods n and (n-1)
V	Profile speed out for interpolation period n

The drive performs interpolation between P(n) and P(n-1) every 250 microseconds. For example, if *interpolated time period* is 2 ms, than there are 8 interpolation steps in one *interpolation time period*. The interpolation is performed in accordance with formula:

$$PPO(k) = P(n) + V * 250us * k$$

Where:

K	number of steps, k=0…7 for the example
PPO(k)	Profiler position out in step k

The *Max buffer size* and *actual buffer size* are set to 1 in sub mode 0. If the *interpolated data records* are sent by the control device via synchronous RPDO, the control device should provide SYNC messages with a period equal to the defined *interpolation time period*. Any other period produces unstable motion.

13.13.2. Sub-mode -1. Linear interpolation

For sub mode -1, the data type is defined by **Object 0x44** and includes position and velocity (refer to the Elmo MAN-G-DS301 CANopen Implementation Guide). The profile velocity at each interpolation period is set by the velocity received via **Object 0x60C1.2**.

When operating in sub mode -1 with an *actual buffer size* set to 1, and if the *interpolated data records* are sent by control device via synchronous RPDO, the control device should provide SYNC messages with a period equal to the defined *interpolation time period*. Any other period results in an unstable motion.

However, when operating in sub mode -1 and the *actual buffer size* is set to greater than 1, the control device should not send an interpolation data record every interpolation period, but it can send data in burst transmissions. In this case, the control device is responsible for providing interpolated data records so, that drive`s buffer to be not empty.

13.13.3. Using the control device to prepare the Gold drive to run in ip mode

1. Set the control device as described in the table below.

Object	Value	Meaning	Comment
0x60c4.6	0	Clear all data records	
0x60c4.6	1	Enable access to buffer	
0x60c0	0 or -1	Set sub mode	
0x60c4.2	0…Max buffer size	Actual buffer size	
0x60c4.5	4 (sub mode 0)or 8 (sub mode -1)	Size of data records	
0x60c4.3	0 (FIFO), 1 (ring)	Buffer organization	There is no meaning to use ring buffer in sub mode -1 with buffer size 1
0x60c2.1 and 0x60c2.2	Set to proper values 1…10 ms	Interpolation period	
0x2F75	Set to proper value	Missed frame interpolation time	
0x2F41	Set to proper value		Take into account: bit 2 is used in Gold for: *Do commutation every motor On*

2. The objects 0x607B, 0x607D, 0x607C, 0x607E, 0x6081, 0x6082, 0x6080, 0x607F, 0x6083, 0x6084, 0x6085, 0x60C5, 0x60C6, 0x605A must be loaded with the desired values.
3. Perform PDO mapping.
4. Set the operating mode object 0x6060 = 7.
5. Now the control device can send rPDO1, rPDO2 and SYNC every interpolated period.

Chapter 14: Cyclic Synchronous Position Mode

14.1. General

Object	Definition
Object 0x60B0	Position Offset
Object 0x60B1	Velocity Offset
Object 0x60B2	Torque Offset

The overall structure for this mode is shown in Figure 14-1. With this mode, the trajectory generator is located in the control device, not in the ELMO drive. In cyclic synchronous operating mode, it provides a target position to the drive device, which performs position control, velocity control and torque control. Optionally, additive velocity and torque values can be provided by the control system in order to allow for velocity and/or torque feedforward.

Measured by sensors, the ELMO drive may provide actual values for position, velocity and torque to the control device. The behavior of the control function is influenced by control parameters like limit functions, which are externally applicable.

14.2. Functional Description

Figure 14-1 shows the inputs and outputs of the drive control function. The input values (from the control function point of view) are the target position and optionally a position offset (to be added to the target position to allow two instances to set up the position) as well as an optional velocity offset and an optional torque offset used for feedforward control. Especially in cascaded control structures, where a position control is followed by a velocity or torque control, the output of the position control loop is used as an input for a further calculation in the drive device. Limit functions may be used to restrict the range of values to avoid unintended positions.

The drive device monitors the following error. Other features specified in this mode are limitation of motor speed and a quick stop function for emergency reasons. The torque may be limited as well.

The interpolation time period defines the time period between two updates of the target position and/or additive position and is used for intercycle interpolation.

The target position is interpreted as absolute value. The position actual value is used as mandatory output to the control device. Further outputs may be the velocity actual value, torque actual value and the velocity sensor actual value.

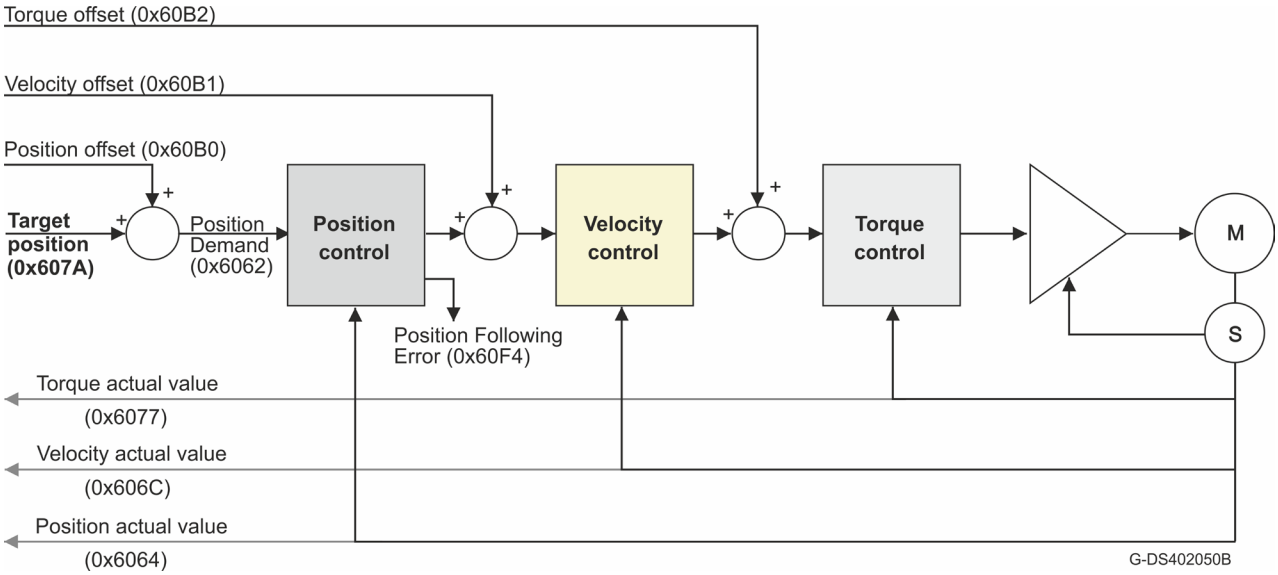


Figure 14-1:Cyclic synchronous position control function

If necessary, all values can be transformed from user-defined units to internal units such as increments with factors. A target position value or position offset outside the allowed range of the following error window around a position demand value for longer than the following error time out results in setting bit 13 (following error) in the *Statusword* to 1.

14.2.1. *Controlword* of Cyclic Synchronous Position Mode

Bit	Function
0	Switch on, see the description in object 0x6040 *Controlword*
1	Enable voltage, see the description in object 0x6040 *Controlword*
2	Quick stop, see the description in 0x6040 *Controlword*
3	Enable operation (set motor on) , see the description in object 0x6040 *Controlword*
4	Reserved, 0
5	Reserved, 0
6	Reserved, 0
7	Fault reset, see the description in object 0x6040 *Controlword*
8	Halt
9	Reserved, 0
10	Reserved, 0
11	Reserved, 0
12	Reserved, 0
13	Reserved, 0
14	Reserved, 0

Bit	Function
15	Reserved, 0

In case of a *Halt*, the drive stops the interpolation and stops the motor according to *halt options code*, see the description in object 0x605D.
If the motor is stopped due to an internal fault or *Controlword* command, the interpolation is disabled. Interpolation can be enabled again only after the device enters the OPERATION_ENABLE mode.

If no target 0x607A is received within the extrapolation timeout, the motor stops according to *quick stop options code*, see the description in object 0x605A.

14.2.2. *Statusword* of Cyclic Synchronous Position Mode

Bit	Function
0	Ready to switch on, see the description in object 0x6041 *Statusword*
1	Switched on, see the description in object 0x6041 *Statusword*
2	Operation enabled, see the description in object 0x6041 *Statusword*
3	Fault, see the description in object 0x6041 *Statusword*
4	Voltage enabled, see the description in object 0x6041 *Statusword*
5	Quick stop, see the description in object 0x6041 *Statusword*
6	Switch on disabled, see the description in object 0x6041 *Statusword*
7	Warning, see the description in object 0x6041 *Statusword*
8	Manufacturer specific, reserved, always 0
9	Remote, see the description in object 0x6041 *Statusword*
10	Reserved
11	Internal limit active, see the description in object 0x6041 *Statusword*
12	Drive follows the command value 0
13	Following error
14 - 15	Manufacturer specific, reserved, always 0

Usage of bits 12 and 13 is demonstrated in the Table 14-1:

Bit	Value	Definition
12	0	Drive does not follow the command value – Target position ignored
	1	Drive follows the command value – Target position used as input to position control loop
13	0	No following error
	1	Following error

Table 14-1: Cyclic Synchronous Position Mode *Statusword*, bits 12,13

14.3. Object 0x60B0: Position offset

This object defines the offset of the target position. The offset is provided in user defined position units. The value itself is absolute and independent of how often it is transmitted over the communication system, for example, transmitted twice does not mean double value. Since the additive position value represents an offset to the target position it can also be used to control the drive with relative values with regard to the target position.

- Object description:

Attributes	0x60B0
Name	Position offset
Object code	VAR
Data type	INTEGER32
Category	Optional

- Entry description:

Sub Index	**0**
Access	Read/write
PDO mapping	Yes
Value range	-2^{31} - $(2^{31})-1$
Default value	0

14.4. Object 0x60B1: Velocity offset

This object provides the offset for the velocity value. The offset is given in user defined velocity units. In cyclic synchronous position mode. This object contains the input value for velocity feed forward. In cyclic synchronous velocity mode it contains the commanded offset of the ELMO drive. The value itself is absolute and independent of how often it is transmitted over the communication system, for example transmitted twice does not mean double value. Since the additive velocity value represents an offset to the target velocity, it can be also used to control the drive with relative values with regard to the target velocity.

- Object description:

Attributes	0x60B1
Name	Velocity offset
Object code	VAR
Data type	INTEGER32
Category	Optional

- Entry description:

Sub Index	**0**
Access	Read/write
PDO mapping	Yes
Value range	-2^{31} - $(2^{31})-1$
Default value	0

14.5. Object 0x60B2: Torque offset

This object provides the offset for the torque value. The offset is given in per thousand rated torque. In cyclic synchronous position mode and cyclic synchronous velocity mode. This object contains the input value for torque feed forward. In cyclic synchronous torque mode it contains the commanded additive torque of the drive, which is added to the target torque value. The value itself is absolute and thus independent of how often it is transmitted over the communication system, for example transmitted twice does not mean double value.

- Object description:

Attributes	0x60B2
Name	Torque offset
Object code	VAR
Data type	INTEGER16
Category	Optional

- Entry description:

Sub Index	**0**
Access	Read/write
PDO mapping	Yes
Value range	-2^{15}...$(2^{15})-1$
Default value	0

Chapter 15: Cyclic Synchronous Velocity Mode

15.1. General

The overall structure for this mode is shown in Figure 15-1. With this mode, the trajectory generator is located in the control device, not in the drive device. In cyclic synchronous mode, it provides a target velocity to the ELMO drive device, which performs velocity control and torque control. If desired, the position control loop may be closed over the communication system. Optionally, additive velocity and torque values may be provided by the control system in order to allow a second source for velocity and/or a torque feed forward. Measured by sensors, the ELMO drive device may provide actual values for position, velocity and torque to the control device.

The cyclic synchronous velocity mode covers the following sub-functions:

- Demand value input
- Velocity capture using position sensor or velocity sensor
- Velocity control function with appropriate input and output signals
- Limitation of torque demand

Various sensors may be used for velocity capture. In particular, the aim is that costs are reduced and the drive power system is simplified by evaluating position and velocity using a common sensor. This option is achieved by using a resolver or an encoder. The behavior of the control function is influenced by control parameters such as limit functions that are externally applicable.

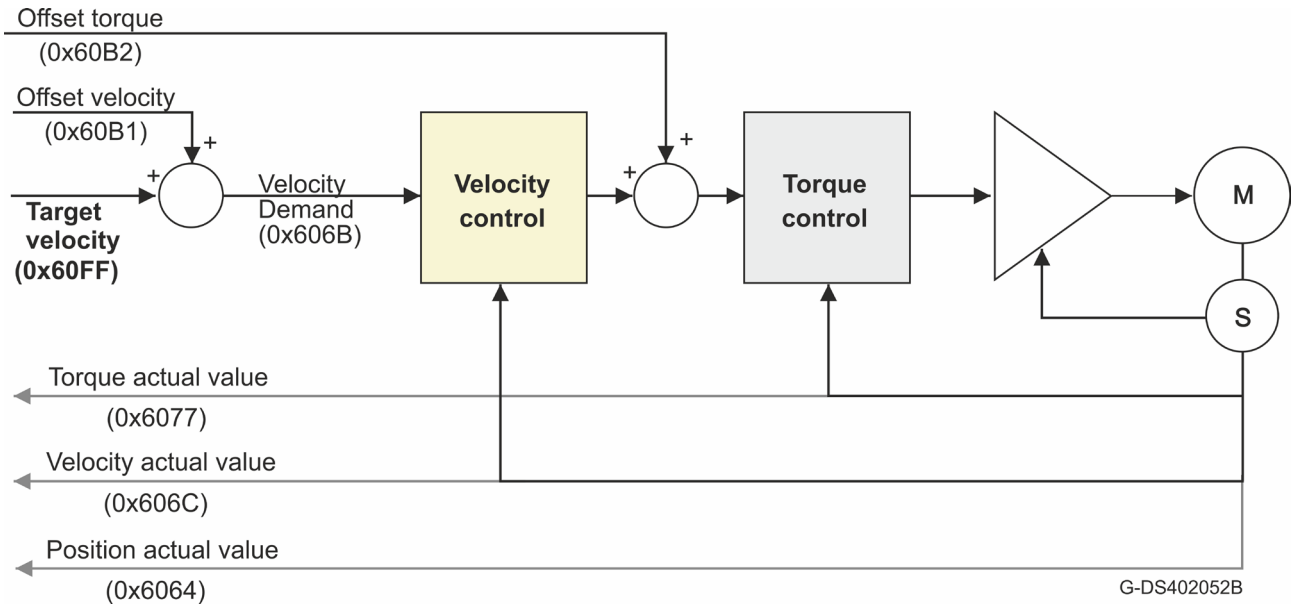


Figure 15-1: Cyclic synchronous velocity mode

15.2. Functional Description

Figure 15-2 demonstrates the inputs and outputs of the drive control function. The input (from the control device point of view) is the target velocity and optionally, a velocity offset (to be added to the target velocity to allow two instances to set up the velocity) as well as a torque offset. Especially in cascaded control structures, where a velocity control is followed by a torque control, the output of the velocity control loop is used as an input for a further calculation in the ELMO drive device.

The drive device may support limitation of motor speed and a quick stop function for emergency reasons. The torque may be limited as well.

The interpolation time period defines the time period between two updates of the target velocity and/or additive velocity and is used for intercycle interpolation.

The velocity actual value is used as mandatory output to the control device. Further outputs may be the torque actual value and the velocity sensor actual value.

All values are transformed, if necessary, from user-defined units to internal units such as increments with the functions described in the **Factors** Chapter.

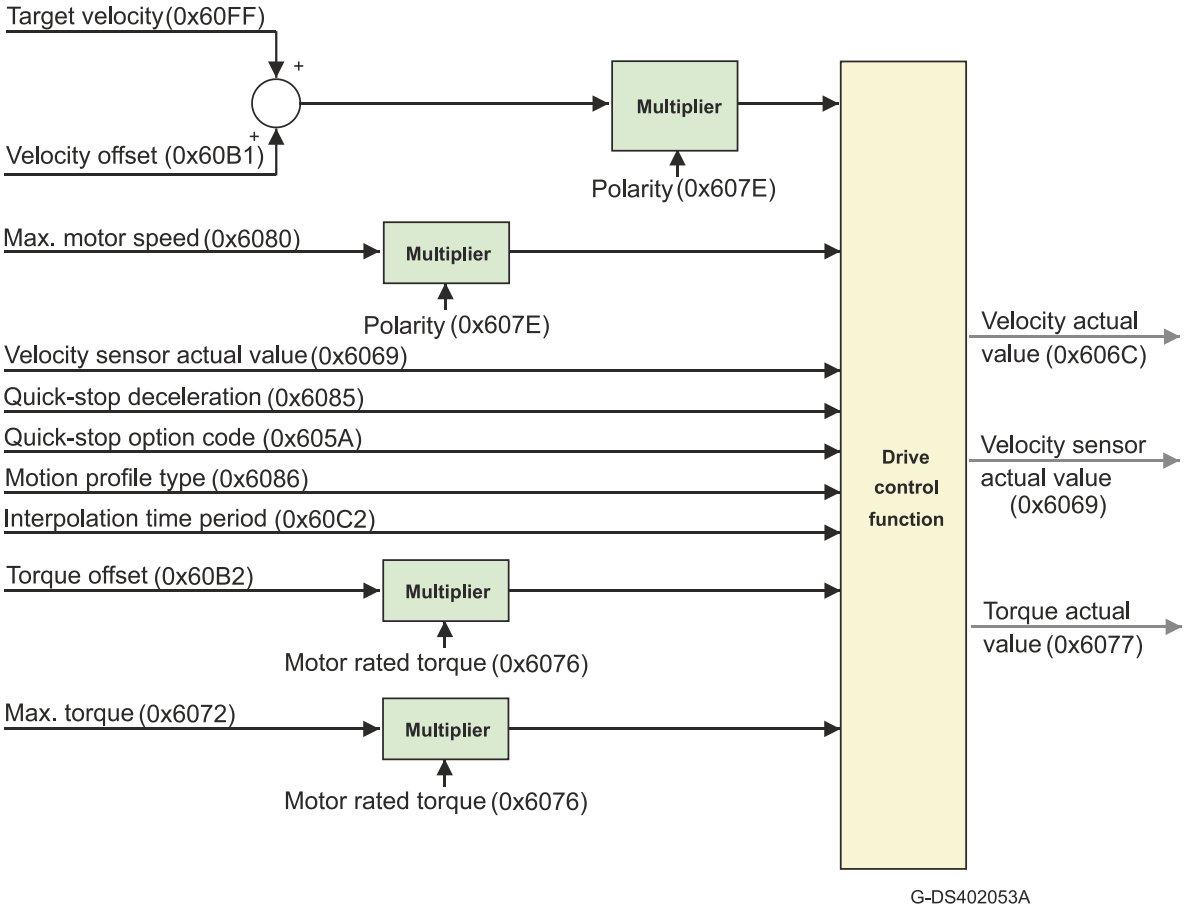


Figure 15-2: Cyclic synchronous velocity control function

15.3. Use of *Controlword* and *Statusword*

15.3.1. *Controlword* of Cyclic Synchronous Velocity Mode

Bit	Function
0	Switch on, see the description in object 0x6040 *Controlword*
1	Enable voltage, see the description in object 0x6040 *Controlword*
2	Quick stop, see the description in 0x6040 *Controlword*
3	Enable operation (set motor on) , see the description in object 0x6040 *Controlword*
4	Reserved, 0
5	Reserved, 0
6	Reserved, 0
7	Fault reset, see the description in object 0x6040 *Controlword*
8	Halt
9	Reserved, 0
10	Reserved, 0
11	Reserved, 0
12	Reserved, 0
13	Reserved, 0
14	Reserved, 0
15	Reserved, 0

In case of a Halt, the drive stops the interpolation and stops the motor according to *halt options code*, see the description in object 0x605D.

In case the motor is stopped due to an internal fault or *Controlword* command, the interpolation is disabled. Interpolation can be enabled again only after the device enters the OPERATION_ENABLE.

If no target 0x6071 received more than extrapolation timeout, the motor stops according to *halt options code*, see the description in object 0x605D

15.3.2. *Statusword* of Cyclic Synchronous Velocity Mode

Bit	Function
0	Ready to switch on, see the description in object 0x6041 *Statusword*
1	Switched on, see the description in object 0x6041 *Statusword*
2	Operation enabled, see the description in object 0x6041 *Statusword*
3	Fault, see the description in object 0x6041 *Statusword*

Bit	Function
4	Voltage enabled, see the description in object 0x6041 *Statusword*
5	Quick stop, see the description in object 0x6041 *Statusword*
6	Switch on disabled, see the description in object 0x6041 *Statusword*
7	Warning, see the description in object 0x6041 *Statusword*
8	Manufacturer specific, reserved, always 0
9	Remote, see the description in object 0x6041 *Statusword*
10	Reserved
11	Internal limit active, see the description in object 0x6041 *Statusword*
12	Drive follows the command value 0
13	Following error
14 - 15	Manufacturer specific, reserved, always 0

Table 15-1 demonstrates the usage of Bit 12:

Bit	Value	Definition
12	0	Drive does not follow the command value – Target velocity ignored
	1	Drive follows the command value – Target velocity used as input to velocity control loop

Table 15-1: Cyclic Synchronous Velocity Mode. *Statusword*, using bit 12

Chapter 16: Cyclic Synchronous Torque Mode

16.1. General

Figure 16-1 demonstrates the overall structure for this mode. With this mode, the trajectory generator is located in the control device, not in the drive device. In cyclic synchronous mode, it provides a target torque to the ELMO drive device, which performs torque control. Optionally, additive torque value may be provided by the control system in order to allow a second source to set up the torque. Measured by sensors, the ELMO drive device may provide actual values for position, velocity and torque to the control device.

The cyclic synchronous torque mode covers the following sub-functions:

- Demand value input
- Torque capture.
- Torque control function with appropriate input and output signals
- Limitation of torque demand

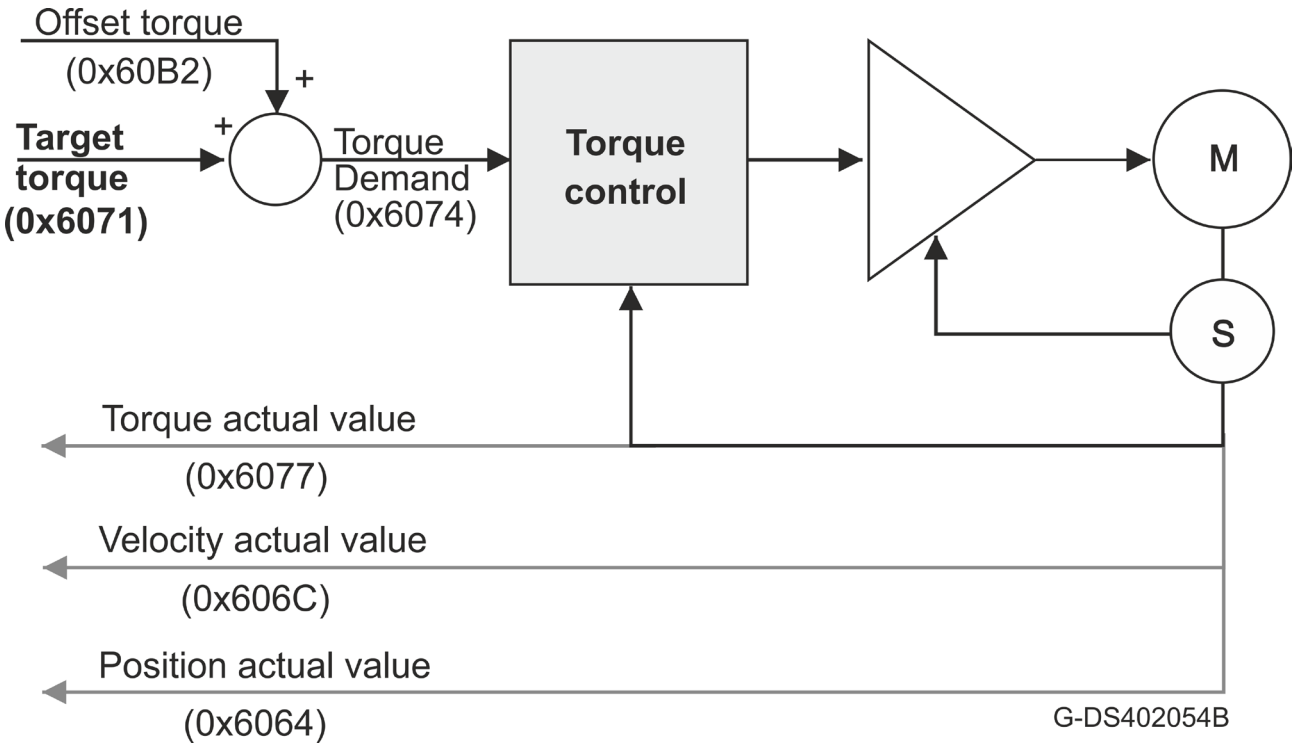


Figure 16-1: Cyclic synchronous torque mode

16.2. Functional Description

Figure 16-2 demonstrates the inputs and outputs of the torque control function. The input (from the control device point of view) is the target torque and optionally, a torque offset (to be added to the target torque to allow two instances to set up the torque) as well as a torque offset.

The drive device supports limitation of motor speed and torque. The interpolation time period defines the time period between two updates of the target

Torque. The outputs may be the torque actual value and the velocity sensor actual value. All values are transformed, if necessary, from user-defined units to internal units with the functions described in the **Factors** Chapter.

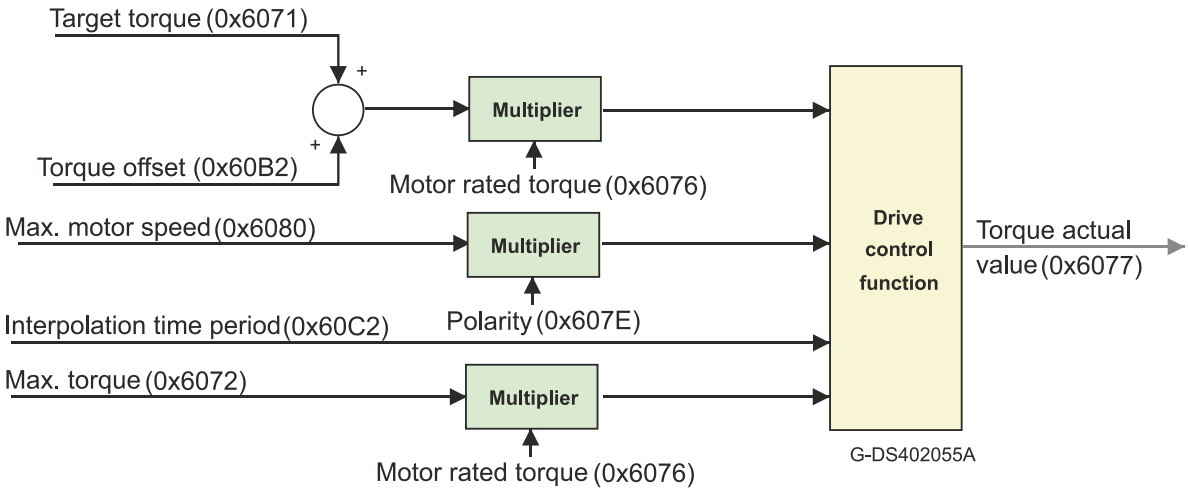


Figure 16-2: Cyclic synchronous torque control function

16.3. Use of *Controlword* and *Statusword*

16.3.1. *Controlword* of Cyclic Synchronous Torque Mode

Bit	Function
0	Switch on, see the description in object 0x6040 *Controlword*
1	Enable voltage, see the description in object 0x6040 *Controlword*
2	Quick stop, see the description in 0x6040 *Controlword*
3	Enable operation (set motor on) , see the description in object 0x6040 *Controlword*
4	Reserved, 0
5	Reserved, 0
6	Reserved, 0
7	Fault reset, see the description in object 0x6040 *Controlword*
8	Halt
9	Reserved, 0
10	Reserved, 0
11	Reserved, 0
12	Reserved, 0
13	Reserved, 0
14	Reserved, 0
15	Reserved, 0

In case of a Halt, the drive stops the interpolation and stops the motor according to *halt options code*, see the description in object 0x605D.

If the motor is stopped due to an internal fault or *Controlword* command, the interpolation is disabled. Interpolation can be enabled again only after the device enters the OPERATION_ENABLE mode.

If no target 0x6071 received more than extrapolation timeout, the motor stops according to *halt options code*, see the description in object 0x605D.

16.3.2. *Statusword* of Cyclic Synchronous Torque Mode

Bit	Function
0	Ready to switch on, see the description in object 0x6041 *Statusword*
1	Switched on, see the description in object 0x6041 *Statusword*
2	Operation enabled, see the description in object 0x6041 *Statusword*

Bit	Function
3	Fault, see the description in object 0x6041 *Statusword*
4	Voltage enabled, see the description in object 0x6041 *Statusword*
5	Quick stop, see the description in object 0x6041 *Statusword*
6	Switch on disabled, see the description in object 0x6041 *Statusword*
7	Warning, see the description in object 0x6041 *Statusword*
8	Manufacturer specific, reserved, always 0
9	Remote, see the description in object 0x6041 *Statusword*
10	Reserved
11	Internal limit active, see the description in object 0x6041 *Statusword*
12	Drive follows the command value 0
13	Following error
14 - 15	Manufacturer specific, reserved, always 0

The usage of bit 12 is presented in Table 16-1.

Bit	Value	Definition
12	0	Drive does not follow the command value – Target torque ignored
	1	Drive follows the command value – Target torque used as input to torque control loop

Table 16-1: Cyclic Synchronous Torque Mode. *Statusword*, using bit 12

Chapter 17: Profiled Velocity

17.1. General

Object	Definition
0x6069	Velocity sensor actual value
0x 6060	Velocity window
0x 606A	Sensor selection code
0x 606B	Velocity demand value
0x 606C	Velocity actual value
0x 606D	Velocity window
0x 606E	Velocity window time
0x 606F	Velocity threshold
0x 6070	Velocity threshold time
0x 60FF	Target velocity

Profile Velocity mode includes the following sub-functions:

- Demand value input via trajectory generator
- Velocity capture using the position sensor or velocity sensor
- Velocity control function with the appropriate input and output signals
- Monitoring of the *profile velocity* using a window function
- Monitoring of the *velocity actual value* using a threshold

The input parameters of the reference value generator are:

- Profile velocity
- Profile acceleration
- Profile deceleration
- Emergency stop
- Motion profile type

The velocity controller calculates a torque variable. When a different *target position* arrives, it is executed immediately.

A target velocity can be executed only in OPERATION ENABLED state. Otherwise, the process aborts with an emergency message.

The velocity, acceleration and deceleration are submitted to the limits according to the relevant limit range.

17.2. Functional Description

Figure 17-1 demonstrates the defined structure of the profile velocity mode. The actual velocity may be obtained through differentiation from the position encoder and is represented in position encoder increments.

The target reached bit -bit 10- is set to 1 in the *Statusword* when the difference between the target velocity and the velocity actual value is within the velocity window longer than the velocity window time.

As soon as the velocity actual value exceeds the velocity threshold longer than the velocity threshold time, then bit 12 in *Statusword* is set to 0. Below this threshold the bit is set to 1 and indicates that the axis is stationary.

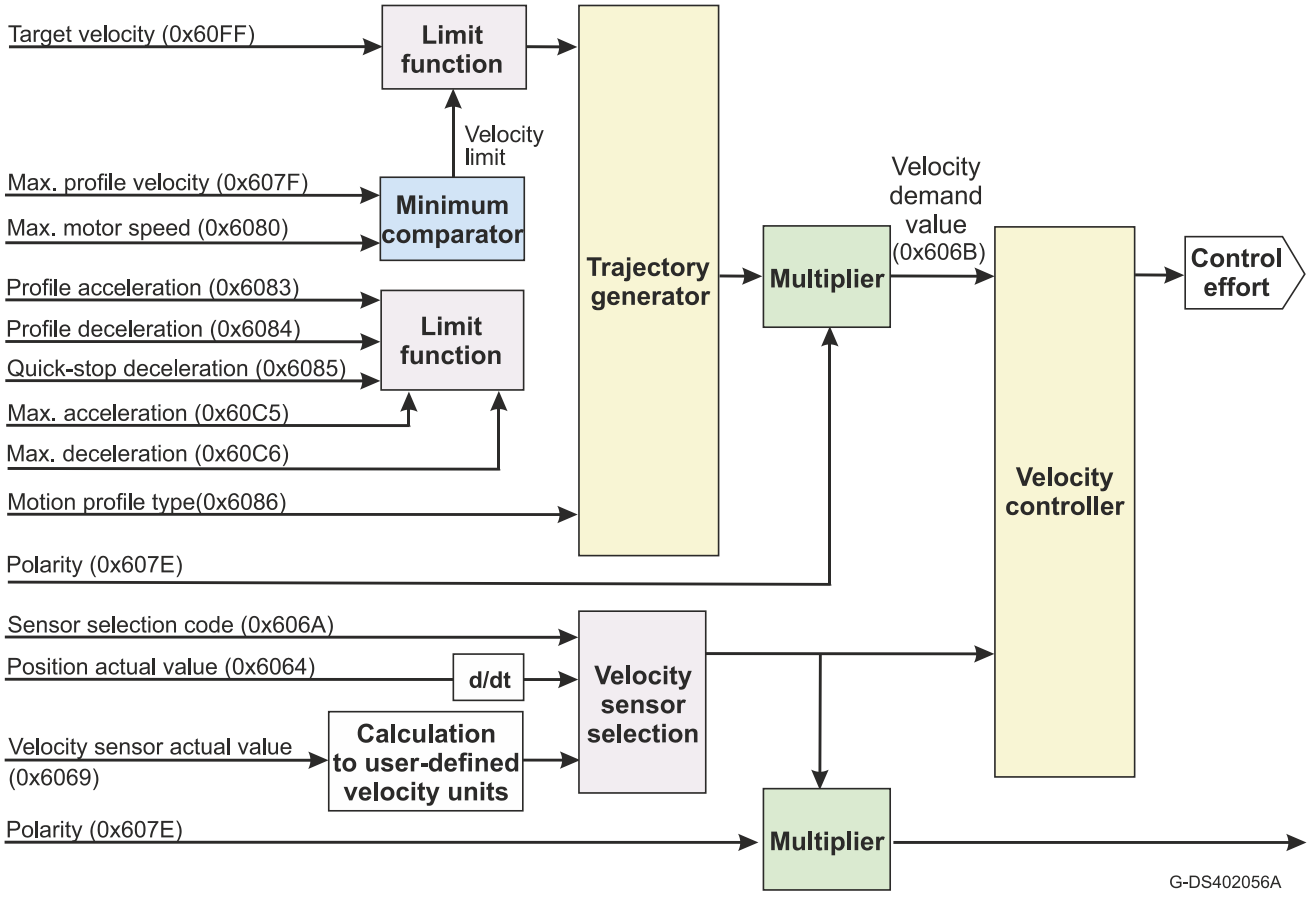


Figure 17-1: Profile velocity mode

17.2.1. *Controlword* of the profiled velocity mode

Bit	Function
0	Switch on, see the description in object 0x6040 *Controlword*
1	Enable voltage, see the description in object 0x6040 *Controlword*
2	Quick stop, see the description in object 0x6040

Bit	Function
	Controlword
3	Enable operation (set motor on) , see the description in object 0x6040 *Controlword*
4	Reserved, 0
5	Reserved, 0
6	Reserved, 0
7	Fault reset, see the description in object 0x6040 *Controlword*
8	Halt
9	Reserved, 0
10	Reserved, 0
11	Reserved, 0
12	Reserved, 0
13	Reserved, 0
14	Reserved, 0
15	Reserved, 0

Name	Value	Description
Halt	0	Execute the motion
	1	Stop axle

17.2.2. *Statusword* of the profiled velocity mode

Bit	Function
0	Ready to switch on, see the description in object 0x6041 *Statusword*
1	Switched on, see the description in object 0x6041 *Statusword*
2	Operation enabled, see the description in object 0x6041 *Statusword*
3	Fault, see the description in object 0x6041 *Statusword*
4	Voltage enabled, see the description in object 0x6041 *Statusword*
5	Quick stop, see the description in object 0x6041 *Statusword*
6	Switch on disabled, see the description in object 0x6041 *Statusword*
7	Warning, see the description in object 0x6041 *Statusword*
8	Manufacturer specific, reserved, always set to 0

Bit	Function
9	Remote, see the description in object 0x6041 *Statusword*
10	Target reached
11	Internal limit active, see the description in object 0x6041 *Statusword*
12	Speed is zero
13	Reserved, always set to 0
14…15	Manufacturer specific, reserved, always set to 0

Name	Value	Description
Target reached Bit 10	0	Halt = 0: *Target velocity* not (yet) reached. Halt = 1: Axle decelerates
	1	Halt = 0: *Target velocity* reached. Halt = 1: Velocity of axle is 0
Speed is zero, bit 12	0	Speed not equal to 0
	1	Speed not equal to 0
	1	Speed equals 0

17.3. Object 0x6069: Velocity Sensor Actual Value

This object defines the value read from a velocity encoder, in increments/second. If the *velocity sensor actual value* is reflected by object 0x606C, it is scaled by *velocity factor 0x6096.*

- Object description:

Attributes	0x6069
Name	Velocity sensor actual value
Object code	VAR
Data type	INTEGER32
Category	Mandatory

- Entry description:

Access	RO
PDO mapping	TxMap
Value range	-2^{31} - (2^{31})-1
Default value	0

17.4. Object 0x606A: Sensor Selection Code

This object is also used to determine the source of the *velocity actual value* (object 0x606C), which determines whether a differentiated position signal or the signal from a separate velocity sensor needs to be evaluated.

- Object description:

Attributes	0x606A
Name	Sensor selection code
Object code	VAR
Data type	INTEGER16
Category	Optional

- Entry description:

Access	Read/write
PDO mapping	No
Value range	0,1
Default value	1

- Data description:

Value	Description
0x0000	Actual velocity value from position encoder
0x0001	Actual velocity value from velocity encoder
0x0002 – 0x7FFF	Reserved
0x8000 - $FFFF_h$	Manufacturer specific, Reserved

17.5. Object 0x606B: Velocity Demand Value

This object defines the value of the velocity command as reflected by the trajectory generator. This value is in user defined units.

- Object description:

Attributes	0x606B
Name	Velocity demand value
Object code	VAR
Data type	INTEGER32
Category	Mandatory

- Entry description:

Access	Read only
PDO mapping	TxMap
Value range	-2^{31} - $(2^{31})-1$
Default value	0

17.6. Object 0x606C: Velocity Actual Value

This object is represented in velocity units and is coupled with the velocity used as input to the velocity controller. The object is taken from either the position sensor or the velocity sensor. In **UM=5** (single position loop), this object reflects the load and the motor value; in **UM=4** (dual loop), object 0x606A determines which sensor is reflected.

- Object description:

Attributes	0x606C
Name	Velocity actual value
Object code	VAR
Data type	INTEGER32
Category	Mandatory

- Entry description:

Access	Read only
PDO mapping	TxMap
Value range	-2^{31} - $(2^{31})-1$
Default value	No

17.7. Object 0x606D: Velocity Window

This object monitors whether the required process velocity has been achieved after an eventual acceleration or deceleration (braking) phase. It is measured in velocity units.

- Object description:

Attributes	0x606D
Name	Velocity window
Object code	VAR
Data type	UNSIGNED16
Category	Optional

- Entry description:

Access	Read/write
PDO mapping	No
Value range	0…65535
Default value	100

17.8. Object 0x606E: Velocity Window Time

The corresponding bit 10 target reached is set in the *Statusword* when the difference between the *target velocity* and the *velocity actual value* is within the *velocity window* longer than the *velocity window time*. The value of the *velocity window time* is given in multiples of milliseconds.

- Object description:

Attributes	0x606E
Name	Velocity window time
Object code	VAR
Data type	UNSIGNED16
Category	Optional

- Entry description:

Access	Read/write
PDO mapping	No
Value range	0 - 65535
Default value	20

17.9. Object 0x606F: Velocity Threshold

As soon as the *velocity actual value* exceeds the *velocity threshold* longer than the *velocity threshold time* bit 12, velocity is 0, is reset in the *Statusword*. Below this threshold, the bit is set and indicates that the axle is stationery. The value is given in velocity units.

- Object description:

Attributes	0x606F
Name	Velocity threshold
Object code	VAR
Data type	UNSIGNED16
Category	Optional

- Entry description:

Access	Read/write
PDO mapping	No
Value range	0 - 65535
Default value	100

17.10. Object 0x6070: Velocity Threshold Time

The *velocity threshold time* is given in multiples of milliseconds. See the description in object 0x606F.

- Object description:

Attributes	0x6070
Name	Velocity threshold time
Object code	VAR
Data type	UNSIGNED16
Category	Optional

- Entry description:

Access	Read/write
PDO mapping	No
Value range	0 - 65535
Default value	20

17.11. Object 0x60FF: Target velocity

The *target velocity* is the input for the trajectory generator. The value is given in velocity units.

- Object description:

Attributes	60FFh
Name	Target velocity
Object code	VAR
Data type	INTEGER32
Category	Mandatory

- Entry description:

Access	Read/write
PDO mapping	Yes
Value range	-2^{31} - $(2^{31})-1$
Default value	0

Chapter 18: Profiled Torque Mode

18.1. General

Object	Definition
0x6071	Target torque
0x6072	Max torque
0x6073	Max current
0x6074	Torque demand value
0x6075	Motor rated current
0x6076	Motor rated torque
0x6077	Torque actual value
0x6078	Current actual value
0x6087	Torque slope

This chapter describes the profile torque mode. The profile torque mode allows a host (external) control system (i.e. closed-loop speed controller, open-loop transmission force controller) to transmit the target torque value, which is processed via the trajectory generator. In profile torque mode, torque slope parameters are required.

If the host control system switches the *Controlword* bit 8 (halt) from 0 to 1, then the trajectory generator ramps its control output down to zero. However, if the host control system switches the *Controlword* bit 8 (halt) from 1 to 0, then the trajectory generator ramps its control output up to the target torque. In both cases the trajectory generator takes the torque slope into consideration.

All definitions within this document refer to rotating motors. Linear motors require that all *torque* objects refer to a *force* instead. For the sake of simplicity, the objects are not duplicated and their names should not be modified. As an example, the linear motor target force must be transmitted using the target torque object. Refer to the object descriptions for additional information.

The torque control parameters, power stage parameters and motor parameters are defined as objects so that they can be handled (i.e. downloaded) in a standard way.

The torque demand, torque actual value, current actual value may be available to the user as parameters, if they are monitored.

Elmo's Gold drives support Profile Torque mode when selected. When Profile Torque objects are set, some internal commands are affected. These internal commands remain even after another operating mode is chosen. The following list shows the objects that are affected, additional information about these commands is available in the *Gold* Command Reference manual:

Objects	Command Reference commands
Max Torque [0x6072]	Set **PL[1]**
Max Current [0x6073]	Set **PL[1]**
Torque Demand Value [0x6074]	Same as **DV[1]**
Motor Rated Current [0x6075]	Set **CL[1]**
Torque Actual Value [0x6077]	Same as **IQ**
Current Actual Value [0x6078]	Same as **IQ**

The following should also be noted:

Because DS-402 defines all relevant torque and current as relative to rate values, motor current and motor torque are, herein, considered to be the same.

The following objects imitate each other and the last value entered is valid:

- 0x6072, 0x6073 for reference
- 0x6077, 0x6078 for feedback

18.2. *Controlword* of the Profile Torque Mode

Bit	Function
0	Switch on, see the description in **Object 0x6040** *Controlword*
1	Enable voltage, see the description in **Object 0x6040** *Controlword*
2	Quick stop, see the description in **Object 0x6040** *Controlword*
3	Enable operation (set motor on) , see the description in **Object 0x6040** *Controlword*
4	Reserved, 0
5	Reserved, 0
6	Reserved, 0
7	Fault reset, see the description in **Object 0x6040** *Controlword*
8	Halt
9	Reserved, 0
10	Reserved, 0
11	Reserved, 0
12	Reserved, 0
13	Reserved, 0
14	Reserved, 0
15	Reserved, 0

Name	Value	Description
Halt	0	Execute the motion
	1	Stop axle

18.3. *Statusword* of the Profiled Torque Mode

Bit	Function
0	Ready to switch on, see the description in **Object 0x6041** *Statusword*
1	Switched on, see the description in **Object 0x6041** *Statusword*
2	Operation enabled, see the description in **Object 0x6041** *Statusword*
3	Fault, see the description in **Object 0x6041** *Statusword*
4	Voltage enabled, see the description in **Object 0x6041** *Statusword*
5	Quick stop, see the description in **Object 0x6041** *Statusword*
6	Switch on disabled, see the description in **Object 0x6041** *Statusword*
7	Warning, see the description in **Object 0x6041** *Statusword*
8	Manufacturer specific, reserved, always 0
9	Remote, see the description in **Object 0x6041** *Statusword*
10	Target Reached
11	Internal limit active, see the description in **Object 0x6041** *Statusword*
12	Reserved
13	Reserved, 0
14 - 15	Manufacturer specific, reserved, always 0

Name	Value	Description
Target reached	0	*Target torque* not (yet) reached.
	1	Target torque reached.

18.4. Object 0x6071: Target Torque

This object is the input value for the torque controller in profile torque mode and the value is given per thousand of rated torque.

- Object description:

Attributes	0x6071
Name	Target torque
Object code	VAR
Data type	INTEGER16
Category	Mandatory

- Entry description:

Access	Read/write
PDO mapping	Yes
Value range	-32768…32767
Default value	0

Example

If a torque that is relative to current of 2 amps is needed, and object 0x6075 (Motor Rate Current) is 3200 mA,

Then

[0x6071] = 2000 mA x 1000 / 3200 mA = 625

This number means 62.5 % of Motor Rate Current.

18.5. Object 0x6072: Max Torque

This value represents the maximum permissible torque in the motor and is given per thousand of rated torque.

- Object description:

Attributes	0x6072
Name	Max torque
Object code	VAR
Data type	UINT16
Category	Mandatory

- Entry description:

Access	Read/write
PDO mapping	No
Value range	1…50000
Default value	50000

The behavior of this object is the same as 0x6073, because the current and the torque are proportional, and both of those objects (0x6072, 0x6073) are relative to the torque (or current).

This object sets **PL[1]**.

18.6. Object 0x6073: Max Current

This value represents the maximum permissible torque creating current in the motor and is given per thousand of rated current.

- Object description:

Attributes	0x6073
Name	Max current
Object code	VAR
Data type	UINT16
Category	Mandatory

- Entry description:

Access	Read/write
PDO mapping	No
Value range	1…50000
Default value	50000

The value in 0x6073 (which is in mA) is entered in **PL[1]** after it is converted to Amperes.

For example:

If we want **PL[1**] to be 4 Amps,

and in [0x6075] is set to 3200 mA

then [0x6073] = 4000 * 1000 / 3200 = 1250

18.7. Object 0x6074: Torque Demand Value

This value represents the maximum permissible torque creating current in the motor and is given in units of per thousand of rated current.

- Object description:

Attributes	0x6074
Name	Torque demand value
Object code	VAR
Data type	INTEGER16
Category	Mandatory

- Entry description:

Access	Read only
PDO mapping	TxMap
Value range	-32768…32767
Default value	0

This Object is the same as **DV[1].**

18.8. Object 0x6075: Motor Rated Current

This value is taken from the motor nameplate and is entered in multiples of milliamp. Depending on the motor and drive technology, this current may be either DC, peak or rms (root-mean-square) current. All relative current data refers to this value.

- Object description:

Attributes	0x6075
Name	Motor Rate Current
Object code	VAR
Data type	UNSIGNED32
Category	Mandatory

- Entry description:

Access	Read/write
PDO mapping	No
Value range	1…**MC***1000
Default value	1

This value of 6075h is set to **CL[1]** after it is converted to Amperes.

18.9. Object 0x6076: Motor Rate Torque

This value is taken from the motor name plate and is entered as a multiple of mNm (millNewtonmeter). All relative torque data refer to this value.

For linear motors, the object name is not changed, but the motor rated force value must be entered as a multiple of mN (mill Newton).

Since the relation between torque and current is linear, consider the torque units as mA in the following descriptions.

- Object description:

Attributes	0x6076
Name	Motor Rate Torque
Object code	VAR
Data type	UNSIGNED32
Category	Mandatory

- Entry description:

Access	Read/write
PDO mapping	No
Value range	1…**MC***1000
Default value	1

18.10. Object 0x6077: Torque Actual Value

The *torque actual value* corresponds to the instantaneous torque in the drive motor. The value is given in units of per thousand of rated torque.

- Object description:

Attributes	0x6077
Name	Torque Actual value
Object code	VAR
Data type	INTEGER16
Category	Mandatory

- Entry description:

Access	Read only
PDO mapping	TxMap
Value range	-32768…32767
Default value	0

This Object is the same as the Actual Current (**IQ**)

18.11. Object 0x6078: Current Actual Value

This object is the current actual value. The value is given in units of per thousand of rated current.

- Object description:

Attributes	0x6078
Name	Current Actual value
Object code	VAR
Data type	INTEGER16
Category	Mandatory

- Entry description:

Access	Read
PDO mapping	TxMap
Value range	-32768…32767
Default value	0

This Object is the same as the Actual Current (**IQ**)

18.12. Object 0x6079: DC Link Circuit Voltage

This object provides the instantaneous DC link voltage at the drive. The value is given in mV.

- Object description:

Attributes	0x6079
Name	DC link circuit voltage
Object code	VAR
Data type	UNSIGNED32
Category	Optional

- Entry description:

Access	Read only
PDO mapping	No
Value range	0…(2^{32})-1
Default value	none

18.13. Object 0x6087: Torque Slope

This object describes the rate of change of torque in units of per thousand of rated torque per second.

- Object description:

Attributes	0x6087
Name	Torque Slope
Object code	VAR
Data type	UNSIGNED32
Category	Mandatory

- Entry description:

Access	Read/write
PDO mapping	Yes
Value range	0…(2^{32})-1
Default value	10000000

If a user submits a value that is greater than of the maximum admissible value of this object, the drive loads the maximum possible value, without issuing a receive error message.

Chapter 19: Touch Probe Functionality

19.1. General

Object	Definition
0x60B8	Touch probe configured function
0x60B9	Touch probe status
0x60BA	Touch probe 1 positive edge
0x60BB	Touch probe 1 negative edge
0x60BC	Touch probe 2 positive edge
0x60BD	Touch probe 2 negative edge

DS-402 defines the functionality of a touch probe. A touch probe is a function that can capture up to two inputs on the main position feedback sensor. The capturing can be performed on both the rising and falling edges.

The touch probe functionality can be switched on and off under any motion mode (except for the DS-402 homing motion mode). Touch probe captures the rising edge before the falling edge. This means that if the input is 1 and the touch is required on a falling edge (from 1 to 0) the first falling edge will be ignored, and the capture will be performed on the next falling edge.

Touch probe 1 input always connected to EQEP timer index of CPU

Touch probe 2 input always connected to EQEP timer strobe of CPU

19.2. 0x60B8: Touch Probe Function

Object **0x60B8** indicates the configured function of the touch probe.

- Object description:

Attributes	0x60B8
Name	Touch probe function
Object code	VAR
Data type	UNSIGNED16
Category	Optional

- Entry description:

Attribute	Value
Sub-index	0x00
Access	Read/write
PDO mapping	See /CiA402-3/
Value range	Bitfield, range is not applicable
Default value	Manufacturer-specific

Value definitions are shown in Table 19-1.

Bit	Binary Value	Definition
0	0	Switch off touch probe 1
	1	Enable touch probe 1
1	0	Trigger first event
	1	Continuous **(Reserved)**
3, 2 **(Reserved, See GI[] & 0x20B0)**	00	Trigger with touch probe 1 input
	01	Trigger with zero impulse signal or position encoder
	10	Touch probe source as defined in object 0x60D0, sub-index 0x01
	11	Reserved

Bit	Binary Value	Definition
4	0	Switch off sampling at the positive edge of touch probe 1
	1	Enable sampling at the positive edge of touch probe 1
5	0	Switch off sampling at the negative edge of touch probe 1
	1	Enable sampling at the negative edge of touch probe 1
6, 7 **(Reserved)**	-	User defined (for example, for testing)
8	0	Switch off touch probe 2
	1	Enable touch probe 2
9	0	Trigger first event
	1	Continuous (**Reserved)**
10, 11 (**Reserved, See GI[] & 0x20B0)**	00	Trigger with touch probe 2 input
	01	Trigger with zero impulse signal or position encoder
	10	Touch probe source as defined in object 0x60D0, sub-index 0x02
	11	Reserved
12	0	Switch off sampling at the positive edge of touch probe 2
	1	Enable sampling at the positive edge of touch probe 2
13	0	Switch off sampling at the negative edge of touch probe 2
	1	Enable sampling at the negative edge of touch probe 2
14, 15 **(Reserved)**	-	User defined (for example, for testing)

Table 19-1 Touch Probe value definitions

19.3. 0x60B9: Touch Probe Status

Object **0x60B9** indicates the status of the touch probe.

- Object description:

Attributes	0x60B9
Name	Touch probe status
Object code	Variable
Data type	Unsigned16
Category	Optional

- Entry description:

Attribute	Value
Sub-index	0x00
Access	Read-only
PDO mapping	See /CiA402-3/
Value range	Unsigned16
Default value	Not defined

Value definitions are shown in Table 19-2.

Bit	Binary Value	Definition
0	0	Touch probe 1 is switched off
	1	Touch probe 1 is enabled
1	0	Touch probe 1 does not have a positive edge value stored
	1	Touch probe 1 has a positive edge value stored
2	0	Touch probe 1 does not have a negative edge value stored
	1	Touch probe 1 has a negative edge value stored
3 to 5	0	Reserved
6, 7	-	User defined (for example, for testing) **(Reserved)**
8	0	Touch probe 2 is switched off
	1	Touch probe 2 is enabled
9	0	Touch probe 2 does not have a positive edge value stored
	1	Touch probe 2 has a positive edge value stored
10	0	Touch probe 2 does not have a negative edge value stored
	1	Touch probe 2 has a negative edge value stored

Bit	Binary Value	Definition
11 to 13	0	Reserved
14, 15	-	User defined (for example, for testing) **(Reserved)**

Table 19-2 Touch Probe Status definitions

19.4. 0x60BA: Touch Probe 1 Positive Edge

This object provides the position value of the touch probe 1 at positive edge. The value is given in user-defined position units.

- Object description:

Attributes	0x60BA
Name	Touch probe 1 positive edge
Object code	VAR
Data type	INTEGER32
Category	Optional

- Entry description:

Attribute	Value
Sub-index	0x00
Access	Read-only
PDO mapping	See /CiA402-3/
Value range	Integer32
Default value	Not defined

19.5. 0x60BB: Touch Probe 1 Negative Edge

This object provides the position value of the touch probe 1 at negative edge. The value is given in user-defined position units.

- Object description:

Attributes	0x60BB
Name	Touch probe 1 negative edge
Object code	VAR
Data type	INTEGER32
Category	Optional

- Entry description:

Attribute	Value
Sub-index	0x00
Access	Read-only
PDO mapping	See /CiA402-3/
Value range	Integer32
Default value	Not defined

19.6. 0x60BC: Touch Probe 2 Positive Edge

This object provides the position value of the touch probe 2 at positive edge. The value is given in user-defined position units.

- Object description:

Attributes	0x60BC
Name	Touch probe 2 positive edge
Object code	VAR
Data type	INTEGER32
Category	Optional

- Entry description:

Attribute	Value
Sub-index	0x00
Access	Read-only
PDO mapping	See /CiA402-3/
Value range	Integer32
Default value	Not defined

19.7. 0x60BD: Touch Probe 2 Negative Edge

This object provides the position value of the touch probe 2 at negative edge. The value is given in user-defined position units.

- Object description:

Attributes	0x60BD
Name	Touch probe 2 negative edge
Object code	VAR
Data type	INTEGER32
Category	Optional

- Entry description:

Attribute	Value
Sub-index	0x00
Access	Read-only
PDO mapping	See /CiA402-3/
Value range	Integer32
Default value	Not defined

Timing diagram for touch probe example

Figure 19-1 shows a timing diagram for an example touch probe configuration and the corresponding behavior. Table 19-3 explains the timing diagram.

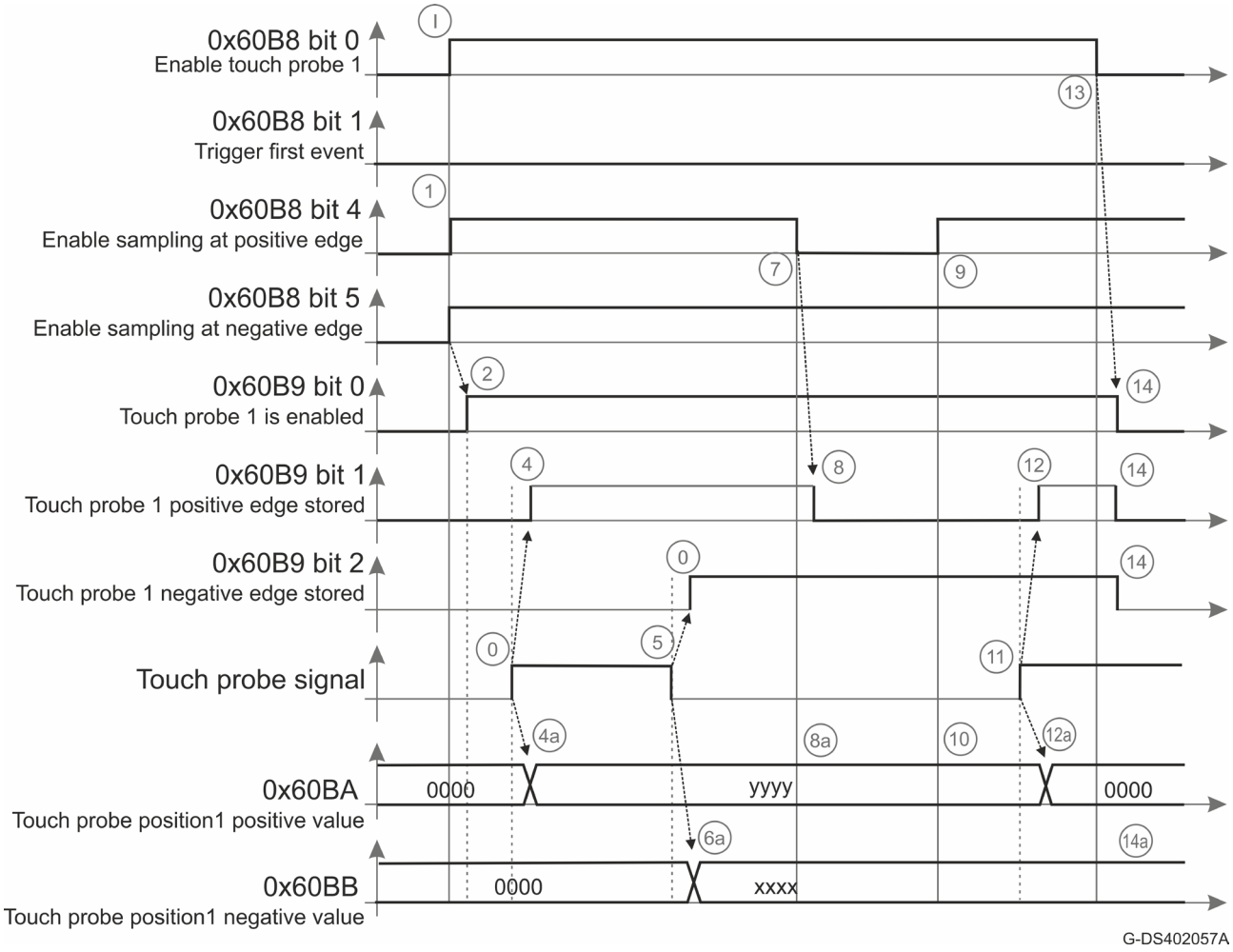


Figure 19-1: Timing diagram for touch probe example

Number	Object	Touch Probe Behavior
(1)	0x60B8, bit 0 = 1	Enable touch probe 1.
	0x60B8, bits 1, 4, 5	Configure and enable touch probe 1, positive and negative edges.
(2)	→ 0x60B9, bit = 1	The status "Touch probe 1 enabled" is set.
(3)		The external touch probe signal has a positive edge.
(4)	→ 0x60B9, bit 1 = 1	The status "Touch probe 1 positive edge stored" is set
(4a)	→ 0x60BA	The touch probe position 1 positive value is stored.
(5)		The external touch probe sign has a negative edge.
(6)	→ 0x60B9, bit 1 = 1	The status "Touch probe 1 negative edge stored" is set.

Number	Object	Touch Probe Behavior
(6a)	→ 0x60BB	The touch probe position 1 negative value is stored.
(7)	0x60B8, bit 4 = 0	The sample positive edge is disabled.
(8)	→ 0x60B9, bit 0 = 0	The status "Touch probe 1 positive edge stored" is reset.
(8a)	→ 0x60BA	The touch probe position 1 positive value is not changed.
(9)	0x60B8, bit 4 = 1	The sample positive edge is enabled.
(10)	→ 0x60BA	The touch probe position 1 positive value is not changed.
(11)		The external touch probe signal has a positive edge.
(12)	→ 0x60B9, bit 1 = 1	The status "Touch probe 1 positive edge stored" is set.
(12a)	→ 0x60BA	The touch probe position 1 positive value is stored.
(13)	0x60B8, bit 0 = 0	Touch probe 1 is disabled.
(14)	→ 0x60B9, bits 0, 1, 2 = 0	The status bits are reset.
(14a)	→ 0x60BA, 0x60BB	The touch probe position 1 positive/negative values are not changed.

Table 19-3 Explanation of the timing diagram

Chapter 20: Support of Additional Sensor Interfaces

20.1. General

ELMO drives support additional sensor interfaces.

Object	Definition
0x60E4	Additional position actual value
0x60E5	Additional velocity actual value

20.2. 0x60E4: Additional Position Actual Value

This object provides the position value of the touch probe 2 at negative edge. The value is given in user-defined position units.

- Object description:

Attributes	0x60E4
Name	Additional position actual value
Object code	VAR
Data type	INTEGER32
Category	Optional

- Entry description:

Access	Read only
PDO mapping	No
Value range	$-2^{32}...(2^{32})-1$
Default value	0

20.3. 0x60E5: Additional Position Actual Value

This object is represented in user defined velocity units. The object is not implemented, returns 0.

- Object description:

Attributes	0x 60E5
Name	Additional velocity actual value
Object code	VAR
Data type	INTEGER32
Category	Mandatory

- Entry description:

Access	Read only
PDO mapping	No
Value range	$-2^{31}...(2^{31})-1$
Default value	0

Chapter 21: Tables

21.1. Dimension Index Table

Physical Dimension	Unit	Dimension Index
None	-	0x00
Way/length	m	0x01
Area	m^2	0xA0
Volume	m^3	0xA1
Time	s	0xA2
	min	0x47
	h	0x48
	d	0x49
	y	0x4A
Power	W	0x24
Revolutions/time	rev / s	0xA3
	rev / min	0xA4
	rev / h	0xA5
Angle	rad	0x10
	s	0x43
	m	0x42
	º	0x41
Velocity	m / s	0xA6
	m / min	0xA7
	m / h	0xA8
Torque	N / m	0xA9
Temperature	K	0x05
	º C	0x2D
	F	0xAA
Voltage	V	0x26
Current	A	0x04

Physical Dimension	Unit	Dimension Index
Ratio	%	0xAB
Frequency	Hz	0x20
Steps	steps	0xAC
Steps / revolution	steps / rev	0xAD

21.2. Notation Index Table

Prefix	Factor	Symbol	Notation Index
unused	-	-	0x13…0x7F
exa	10^{18}	E	0x12
-	10^{17}	-	0x11
-	10^{16}	-	0x10
peta	10^{15}	P	0x0F
-	10^{14}	-	0x0E
-	10^{13}	-	0x0D
tera	10^{12}	T	0x0C
-	10^{11}	-	0x0B
-	10^{10}	-	0x0A
giga	10^{9}	G	0x09
-	10^{8}	-	0x08
-	10^{7}	-	0x07
mega	10^{6}	M	0x06
-	10^{5}	-	0x05
-	10^{4}	-	0x04
kilo	10^{3}	k	0x03
hecto	10^{2}	h	0x02
deca	10^{1}	da	0x01
-	10^{0}		0x00

Prefix	Factor	Symbol	Notation Index
deci	10^{-1}	d	0xFF
centi	10^{-2}	c	0xFE
milli	10^{-3}	m	0xFD
-	10^{-4}	-	0xFC
-	10^{-5}	-	0xFB
micro	10^{-6}	μ	0xFA
-	10^{-7}	-	0xF9
-	10^{-8}	-	0xF8
nano	10^{-9}	n	0xF7
-	10^{-10}	-	0xF6
-	10^{-11}	-	0xF5
pico	10^{-12}	p	0xF4
-	10^{-13}	-	0xF3
-	10^{-14}	-	0xF2
femto	10^{-15}	f	0xF1
-	10^{-16}	-	0xF0
-	10^{-17}	-	0xEF
atto	10^{-18}	a	0xEE
unused	-	-	0xED…0x80


Inspiring Motion
Since 1988
For a list of Elmo's branches, and your local area office, refer to the Elmo site www.elmomc.com
Elmo
Motion Control