transtechnik
servomécanismes

TT230

ServoPac
TT230
User Guide

en

Digital drive for sinusoidal synchronous AC motors

TT230

WARNING

This is a general manual describing a series of servo drives having output capability suitable for driving AC brushless sinusoidal servo motors.

Please see **TT230 Installation Guide** for the hardware installation of the drive (mounting, wiring, …).

Instructions for storage, use after storage, commissioning as well as all technical details require the MANDATORY reading of the manual before getting the drives operational.

Maintenance procedures should be attempted only by highly skilled technicians having good knowledge of electronics and servo systems with variable speed (EN 60204-1 standard) and using proper test equipment.

The conformity with the standards and the "CE" approval is only valid if the items are installed according to the recommendations of the drive manuals. Connections are the user's responsibility if recommendations and drawings requirements are not met.

CAUTION
Any contact with electrical parts, even after power down, may involve physical damage.
Wait for at least 10 minutes after power down before handling the drives (a residual voltage of several hundreds of volts may remain during a few minutes).

ESD INFORMATION (ElectroStatic Discharge)
TRANSTECHNIK drives are conceived to be best protected against electrostatic discharges. However, some components are particularly sensitive and may be damaged if the drives are not properly stored and handled.

STORAGE

- The drives must be stored in their original package.
- When taken out of their package, they must be stored positioned on one of their flat metal surfaces and on a dissipating or electrostatically neutral support.
- Avoid any contact between the drive connectors and material with electrostatic potential (plastic film, polyester, carpet …).

HANDLING

- If no protection equipment is available (dissipating shoes or bracelets), the drives must be handled via their metal housing.
- Never get in contact with the connectors.

ELIMINATION
In order to comply with the 2002/96/EC directive of the European Parliament and of the Council of 27 January 2003 on waste electrical and electronic equipment (WEEE), all TRANSTECHNIK devices have got a sticker symbolizing a crossed-out wheeled dustbin as shown in Appendix IV of the 2002/96/EC Directive.
This symbol indicates that TRANSTECHNIK devices must be eliminated by selective disposal and not with standard waste.

TRANSTECHNIK does not assume any responsibility for any physical or material damage due to improper handling or wrong descriptions of the ordered items.
Any intervention on the items, which is not specified in the manual, will immediately cancel the warranty.

TRANSTECHNIK reserves the right to change any information contained in this manual without notice.

Issue: **1.2.1.80**

Contents

Chapter 1 - General Description

1.1 - INTRODUCTION

TT230 all-digital drives with sinusoidal PWM control are servo drives that provide the control of brushless AC motors with position sensor.

The standard control inferface can be:

- CANopen,
- EtherCAT®[1],
- analog,
- stepper motor emulation,
- logic I/Os.

But the **TT230** range also offers more sophisticated functions such as:

- DS402 including position capture,
- Master/slave and camming,
- Positioner with motion sequencing.

All versions are delivered as standard with the integrated protection function **Safe Torque Off** : **STO SIL 2**.

With its very small dimensions, the **TT230** is a single-axis stand-alone module that includes power supply and mains filtres. It is available in 230 Vac single-phase and particularly suited to low power applications from 0.5 to 3 kW.

Series **TT230** drives are fully configurable in order to fit various applications. Both drive versions of the **TT230** range are described below.

The **TT230** version with CANopen interface can be used in the following application types:

- Axes controlled by CANopen fieldbus according to the DS402 protocol,
- Stand-alone operation as a motion sequencer with control by means of logic I/Os,
- Traditional analog speed amplifier with +/- 10 V command and position output by A, B, Z encoder signal emulation,
- Stepper motor emulation with PULSE and DIR command signals.

The **TT230** version with EtherCAT® interface can be used in the following application types:

- Axes controlled by EtherCAT® fieldbus according to the DS402 protocol,
- Stand-alone operation as a motion sequencer with control by means of logic I/Os.

The configuration and parameterization software tool Gem Drive Studio allows a quick configuration of the **TT230** drives according to the application requirements.

In this manual, we will use the generic and standard vocabulary to describe these variables. The variables are specified as “parameters” from the communication side.
Each parameter is identified by:
- an Index number and a Sub-index number
- a Name.

Each parameter has the following properties:
- Access type: it is possible to read it, to write it….; "ro" ” means "read only" , “rw“ means "read & write".
- Length: byte, word (16 bit), long (32 bit).
- Possibility or not to access the parameter by using fast communication CANopen services (Process Data Object service PDO). If yes, the field “PDO mapping” of the object dictionary will be “yes”.

Convention: A numerical field can be filled in with numerical values described as “hexadecimal” or “decimal”. An hexadecimal value will be written “ 0xvalue”.

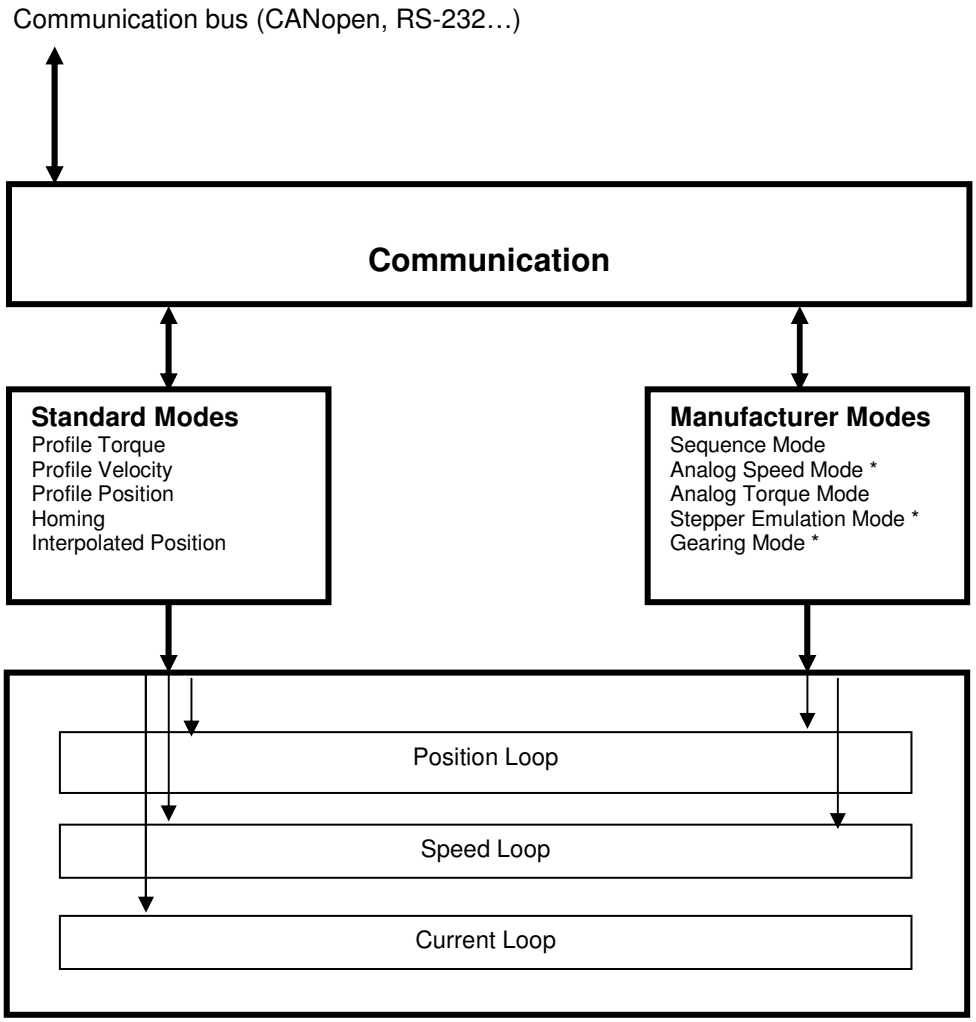
[1] *EtherCAT® is a registered trade mark and a patented technology of Company Beckhoff Automation GmbH, Germany.*

1.2 - ARCHITECTURE

TT230 is a free configurable drive.
The drive configuration includes servo-loop parameters, motor and sensor parameters, communication parameters and I/O configuration parameters. The configuration parameters can be stored into the drive non volatile memory.

The **TT230** drive can be controlled via the fieldbus (CANopen or EtherCAT®), via the analog input (analog speed drive), via the PULSE and DIR inputs (stepper emulation) or via the digital I/Os (stand-alone positioner) according to the selected operation mode.

The following figure describes the functional architecture of the **TT230** drive:



* Note: Analog Speed Mode, Stepper Emulation Mode and Gearing Mode are not available in the EtherCAT version.

1.3 – OTHER DOCUMENTS

- **TT230** Installation guide.
- **TT230** "Safe Torque Off " specification.
- **TT230** Templates.
- GemDriveStudio quick start guide
- EtherCAT® fieldbus interface.

Chapter 2 - Commissioning

This chapter describes the commissioning procedure of the drive by means of the "Gem Drive Studio" software.

CAUTION !
Do not perform the drive parameterization by means of both "Gem Drive Studio" software tool and CANopen bus at the same time.

2.1 - PC SOFTWARE INSTALLATION

The **Gem Drive Studio** software is PC compliant under Windows and allows an easy parameterization of the **ServoPac** drives.
Please see our website www.TRANSTECHNIK.com for downloading the "Gem Drive Studio" software.

Minimum Configuration

The use of the **Gem Drive Studio** software requires the minimum PC configuration described below:

- Processor: 800 MHz
- 256 MB RAM,
- screen with true colours and 1027x768 resolution
- keyboard + mouse
- Windows98© operating system or later
- At least 20 MB available on the hard disk.
- RS232 cable or USB/RS232 adapter cable or IXXAT Can card.

Installation

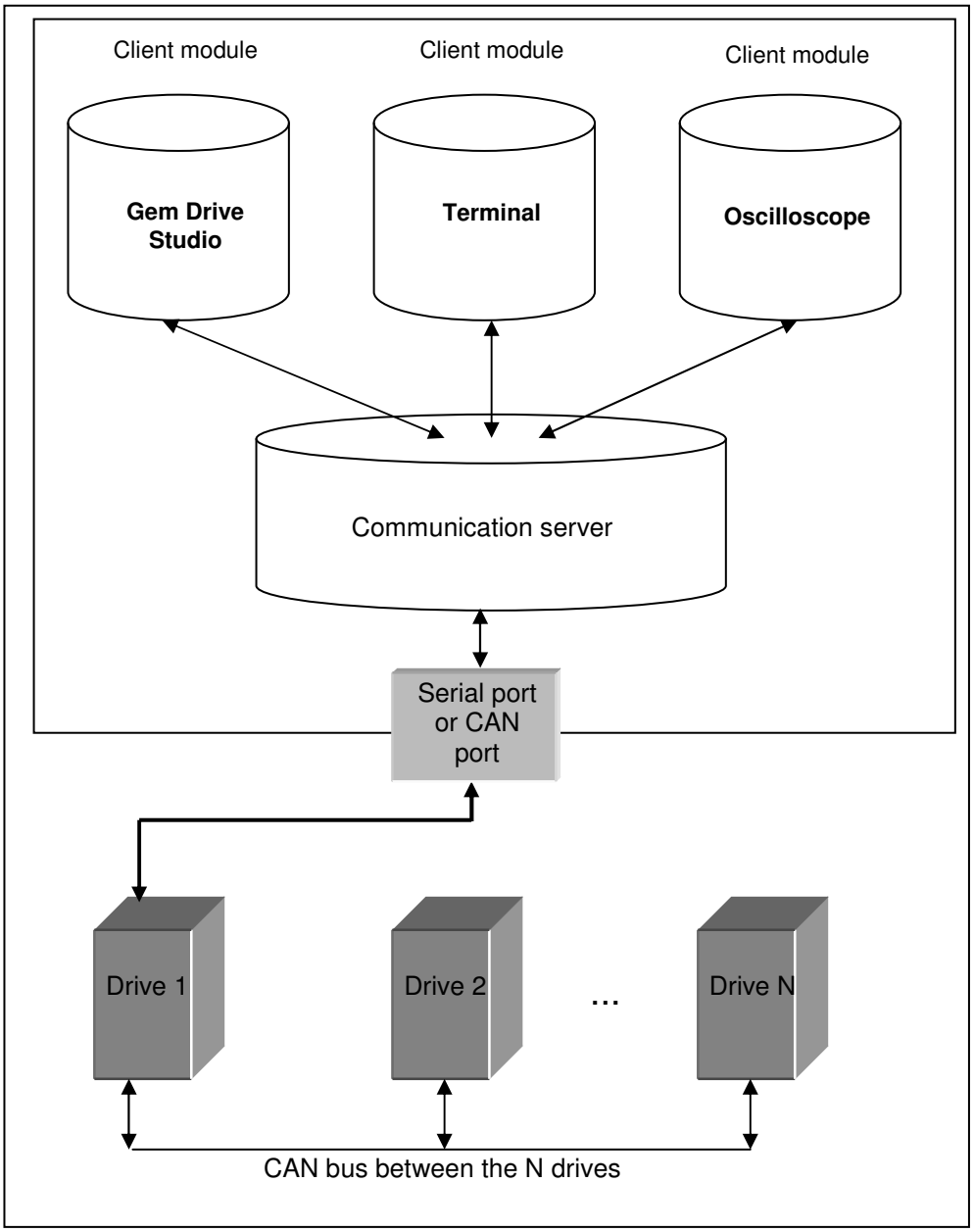
During the installation, one or several messages indicating that a currently copied file is older than a file already existing on the PC, may be displayed. In this case, keep the PC file.

When installing the software, 3 icons are created on the desktop :

- "GemDriveStudio", for launching the main interface.
- "GemDriveOscillo", for launching the digital oscilloscope.
- "GemDriveTerminal", for opening a dialog terminal.

Architecture of the software

The software is made of several independent software modules. Each of them can communicate with the drive(s) via a communication server.



- The server is automatically started when a client module is trying to establish a communication with a drive .
- The server is commissioning the drivers of the hardware peripherals.
- The server stops when the last connected client is stopped .
- The format of the exchanged data is the same whichever the communication type (RS232, CAN, ...).

2.2 - STARTING THE SOFTWARE

User levels
When starting the software, various user levels can be selected. The drive parameter modification levels are protected by passwords. **Administrator** is the highest level with full access.

Passwords
The Administrator can change all passwords by using the Tools/User identification menu. The default **password** for the administrator level is "admin".

Project management
The **Gem Drive Studio** software allows the parameterization of all **ServoPac** drives for a given application. All **ServoPac** drives of a given application, connected together via CANopen, are included in the same **project**. Each **ServoPac** drive of the project is identified by a **node ID** which is coded on the drive front panel by means of microswitches. The **ServoPac** drive node ID code values must all be different from each other in the same project.

The different software commands allow to:

- Create a project,
- Open an existing project,
- Add and/or remove axis in the project,
- Archive/Unarchive a project,

Object dictionaries
Each parameter (object) of the drive can be defined by an **Index**, a **Sub-index** and several properties (Save type, Data type, Unit, Min value, Max value, Default value). The object list with all properties can be downloaded from the drive to create the **object dictionary** file in XML format. This file, named **EEDS** (for Extended Electronic Data Sheet), is used by Gem Drive Studio to read and write parameters on the drive.

The different software commands allow to:

- Download an EEDS file from the drive and add it to the EEDS library,
- Import an EEDS file to the library.

For each new axis of the project, the software creates, in the project file directory, a new directory with the axis name. There will then be one directory per axis and each of these directories will contain the parameter files and the sequence files.

Starting Gem Drive Studio

- Start the software with the **Administrator** level.

- Create the **project**:
  - Define a project name
  - Select an output directory
  - Define all the axes of the application.

- Define the different project **axes**:
  - Select the device type
  - Define the axis name
  - Identify the Node ID for this axis

Once a project has been created, each axis can be independently selected by using the tree structure.

2.3 - DRIVE COMMUNICATION

Powering the drives

Please see manual "Installation Guide" before switching on the drives for the first time. For switching on the drives, proceed as follows:

- Switch on the +24V auxiliary supply:
The red front panel LED "**ERR**" must be blinking ("Undervolt" error displayed).
The AOK relay contact is closed. It is then possible to control the Power ON relay.

- Switch on the power supply:
The red **ERR** LED must be unlit. The drive is ready to be enabled.

Starting the communication

The **Gem Drive Studio** software can communicate with the drives by using either the RS232 serial link or the CANopen fieldbus. All drives of the application are connected together via CANopen:

- Set the node ID code value by using the microswitches on the front panel for all drives of the application (code values must be different from each other),

- Connect the serial link RS232 or the CANopen fieldbus between the PC and one drive of the application,

- Start the **Gem Drive Studio** software on the PC,

- Open the **Project,**

- Select the communication interface between the drives and the PC (Serial link or CANopen bus),

- Start the communication,

- If the **Project** is not defined, use the **Scan** function for starting the communication.

2.4 - PARAMETER SETTING

This chapter describes the parameterization procedure of the drive by means of the "Gem Drive Studio" software.

2.4.1 – CONFIGURATION OF THE DRIVE

For a standard drive application (analog speed drive, stand-alone positioner, or stepper emulation), select the required target application in the **Device Config** window. In this case, the drive input and output functionalities as well as the drive operation mode are automatically set according to the selected application template. The **GemDriveStudio** parameterization windows are also adapted to the target application in order to display only the required parameters and functions.

In order to access to the full parameter set and operation modes, select **Expert** mode in the **Device Config** window.

2.4.2 – CONFIGURATION OF THE MOTOR

If the motor is referenced in the **Gem Drive Studio** motor catalog, it can be simply selected in the proposed motor list.

If the motor is not referenced in the **Gem Drive Studio** motor catalog, the motor parameters can be adjusted manually or calculated by using the drive's built-in procedures: current loop calculation, auto-phasing, ... The motor can then be referenced in the **Gem Drive Studio** motor catalog by using the command **Add new motor** (see **GemDriveStudio** quick start manual). The motor and the position sensor parameter values are entered manually and then saved in the **Gem Drive Studio** motor catalog with a new motor reference.

2.4.2.1 - Selection in the motor list

Select, in the motor list, the motor used in the application. The motor selection will automatically set the following drive parameters : position sensor (resolver or encoder), thermal sensor, current limits, speed limit, current loop gains and motor control parameters.

Check that the thermal sensor calibration is complying with the motor application and modify the threshold values if necessary.

Check that the current limit and the I^2t protection adjustment are complying with the motor application, and modify them if necessary.

Check that the motor speed limit is complying with the application and reduce its value if necessary.

If external inductances are serially connected with the motor winding for filtering, renew the current loop gain calculation by using the total value of the phase-to-phase inductance.

If the position sensor adjustment (resolver or absolute encoder) has been modified, the auto-phasing procedure can be used to find the new adjustment (position offset).

2.4.2.2 - Manual motor configuration

If the motor configuration must be manually made (motor is not referenced in the **Gem Drive Studio** catalog), adjust first the motor position sensor parameters (resolver or encoder) before the motor parameter adjustment.

Configuration of the motor thermal sensor

Selection of the sensor type
The motor can be equipped either with a CTN sensor (ohmic resistance = decreasing temperature function) or with a CTP sensor (ohmic resistance = increasing temperature function).
Check that the selected thermal sensor type actually corresponds to the sensor type mounted on the application motor.

Triggering threshold adjustment
Enter the sensor ohmic value (kOhm) corresponding to the required temperature value for the release of the motor over-temperature protection, according to the manufacturer's specifications.

Warning threshold adjustment
Enter the sensor ohmic value (kOhm) corresponding to a warning temperature value.
When the warning temperature is reached, the warning bit in status word is set.

Note
When using a CTN sensor, the warning ohmic value will be higher than or equal to the triggering ohmic value.
When using a CTP sensor, the warning ohmic value will be lower than or equal to the triggering ohmic value.

Current limit adjustment

The parameter **Maximum current** defines the maximum output current value of the drive. It may vary between 20 % and 100 % of the drive current rating.

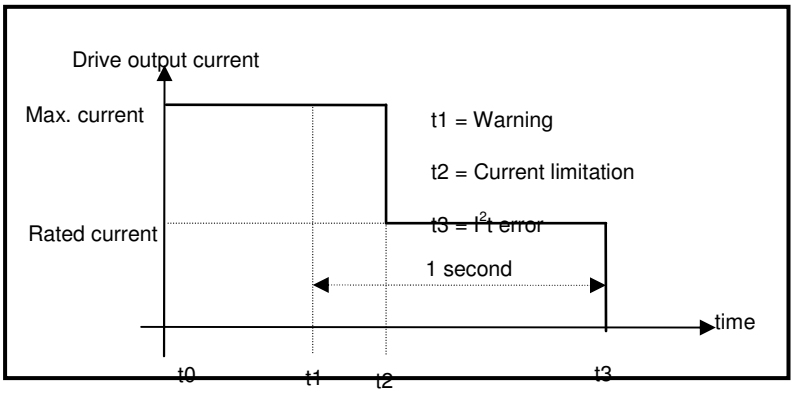
The parameter **Rated current** defines the limitation threshold of the drive output RMS current (I^2t). It can vary between 20 % and 50 % of the drive current rating.

I^2t protection adjustment

2 selection modes are available: Fusing or Limiting.
It is advisable to use the Fusing mode during the commissioning phases.
In **Fusing** mode, the drive is disabled when the current limitation threshold is reached.
In **Limiting** mode, the motor current is only limited at the value defined by the **Rated current** parameter when the limitation threshold is reached.

Operation of the Current Limitation in "Fusing" Mode
When the drive output RMS current (I^2t) reaches 85 % of the rated current, the I²t warning is displayed. If the RMS current (I^2t) has not dropped below 85 % of the rated current within 1 second, the I^2t error is released and the drive disabled (otherwise, the I²t warning is removed).
When the drive output RMS current (I^2t) reaches the rated current value, the I^2t limits the drive output current at this value.

Diagram of the drive output current limitation in an extreme case (motor overload or shaft locked):



The maximum current duration before release of the warning is depending on the value of the parameters **Rated current** and **Max. current**. This value is calculated as follows:

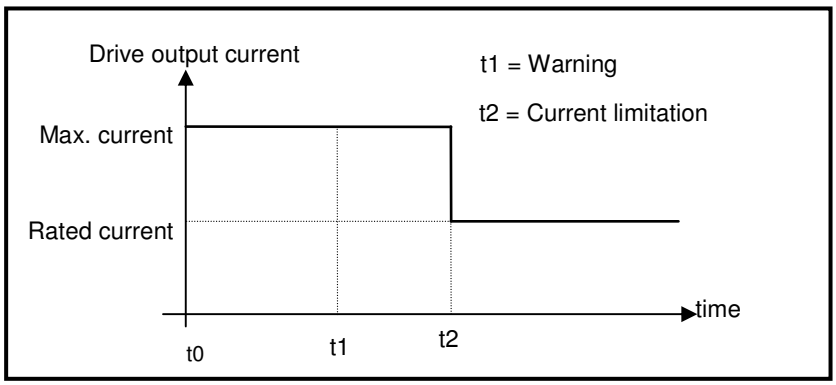
T_{dyn} (second) = t_1-t_0 = 3,3 x [rated current (A) / max. current (A)]^2 (shaft locked conditions)
T_{dyn} (second) = t_1-t_0 = 10 x [rated current (A) / max. current (A)]^2 (motor running with current frequency value higher than 2 Hz)

The maximum current duration before limitation at the rated current is also depending on the value of the **Rated current** and **Max. current** parameters. This value is calculated as follows:

T_{max} (second) = t_2-t_0 = 4 x [rated current (A) / max. current (A)]^2 (shaft locked conditions)
T_{max} (second) = t_2-t_0 = 12 x [rated current (A) / max. current (A)]^2 (motor running with current frequency value higher than 2 Hz)

NOTE
When the "Max. current / Rated current" ratio is close to 1, the Tdyn and Tmax values given by the formula above are quite below the real values. But this formula remains very precise as long as the "Max. current / Rated current" ratio is higher than 3/2.

Operation of the Current Limitation in "Limiting" Mode
When the drive output RMS current (I^2t) reaches 85 % of the rated current, the I²t warning is displayed. When the RMS current (I^2t) drops below 85 % of the rated current, the I²t warning is removed.
When the drive output RMS current (I^2t) reaches the rated current value, the I^2t protection limits the drive output current at this value.
Diagram of the drive output current limitation in an extreme case (motor overload or shaft locked):



The maximum current duration before warning (t1 - t0) and before limitation at the rated current (t2 - t0) is calculated the same way as in the "Fusing" mode.

Speed limit adjustment

The **Maximum speed** parameter defines the speed limit of the motor. This value is given in the motor catalog according to the rated supply voltage and the rated load conditions. If the drive output voltage is lower than the motor rated voltage value, the **Maximum speed** must be reduced accordingly.

The maximum value for the speed set point in the application must be adjusted in order to get a motor speed value lower than the **Maximum speed** parameter. A margin of 10 % to 20 % is recommended.

Current loop adjustment

Enter the value of the total phase-to-phase inductance connected to the drive (motor internal winding inductance + external filtering inductance if used).

The current loop gains are automatically calculated when the command **Calculate current loop gains** is selected.

NOTE
If the drive supply voltage value is changed, the current loop gains are automatically adjusted accordingly, inside the drive. A new calculation is not required.

Auto-phasing of the motor

The **Auto-phasing** procedure identifies the parameters **Pole pairs**, **Phase order** and **Position sensor offset** of a motor.

- The **Pole pairs** parameter defines the number of motor pole pairs.

- The **Phase order** parameter defines the sequence of the motor phases.

- The **Position sensor offset** parameter defines the mechanical shift between the motor and the position sensor (resolver or absolute encoder) reference.

Before executing the **Auto-phasing** procedure, proceed as follows:

- Check that the values of the **Maximum current** and **Rated current** parameters are compatible with the motor. Otherwise, modify them according to the motor specifications.

- Select the I^2t protection in fusing mode. The **Fusing** mode should be used for the commissioning phases.

- Uncouple the motor from the mechanical load and check that the motor shaft is free and for free rotation (1 revolution) that is not dangerous for the operator.

2.4.3 - POSITION SENSORS

The **TT230** drive has got 2 position sensor inputs: one for resolvers and a second for encoders.

Transmitter resolver type or SinCos tracks resolver type can both be connected to the drive resolver input.

Many different encoder types can also be connected to the **TT230** drive encoder input: TTL (square) signals, SinCos signals, incremental + Hall effect sensor channels, absolute encoders with HIPERFACE® communication protocol.

All internal position setpoints and displays are given by using the “user unit“ definition. All internal speed setpoints and displays are given by using the “user unit / second“ definition. So, it is necessary to define inside the drive the relationship between sensor data and “user unit” value.

Resolver input configuration

Select **Enable resolver input** if a resolver is connected to the drive. Otherwise, the **Enable resolver input** can be deselected.

Select the appropriate resolver type:

- A transmitter resolver is supplied by the drive modulation signal at 8 kHz. Transformation ratios from 0.3 to 0.5 are acceptable. The modulated Sine and Cosine signals of the resolver are connected to the drive resolver input.

- A SinCos tracks resolver is supplied by the drive +5V sensor supply. The Sin and Cos output signals have an amplitude of 1 Vpp (electrically compatible with SinCos encoders) and are connected to the drive resolver input.

Enter the **Resolver pole pairs** for a rotating resolver : number of resolver Sine or Cosine signal periods over one shaft revolution.

Adjust the resolver **Zero mark shift** and **Zero mark width** parameter values. The resolver provides one zero mark per pole pair.

Select **Reverse position** in order to reverse the resolver counting direction, if required.

Encoder input configuration

Select **Enable encoder input** if an encoder is connected to the drive. Otherwise, the **Enable encoder input** can be deselected.

Select the appropriate encoder type:

- TTL encoders refer to square quadrature signals electronically compatible with RS422 standard.

- SinCos encoders refer to analog Sine and Cosine signals with 90° phase shift and 1Vpp amplitude.

- Hall effect sensors refer to extra commutation channels for the motor current commutation. Hall effect sensors signal are adapted to the motor pole pairs.

- HIPERFACE® refers to standard communication protocols for absolute single-turn or absolute multi-turn encoders.

Incremental encoder setting:

Enter the **Zero Mark pitch** parameter value if the encoder has got a Zero mark channel. **Zero Mark pitch** is the number of encoder increments between 2 successive zero mark signals. If the encoder is not equipped with a Zero mark channel, set **Zero Mark pitch** value at 0.

Enter the **Resolution** parameter value according to the encoder mounting and the mechanical ratio for a given application.

- If the encoder is directly mounted on the motor: **Resolution** = 4 x number of encoder signal periods per shaft revolution for a rotating motor or number of encoder signal periods per pole pitch for a linear motor.

- If the encoder is coupled to the motor according to a mechanical ratio, the value of the mechanical ratio must be considered for the **Resolution** parameter calculation.

Select **Reverse direction** in order to reverse the counting direction of the encoder, if required.

Adjust the encoder **Zero mark shift** and **Zero mark width** parameter values if the encoder has got a zero mark channel.

Incremental encoder + HES setting:

Enter the **Zero Mark pitch** parameter value if the encoder has got a Zero mark channel. **Zero Mark pitch** is the number of encoder increments between 2 successive zero mark signals. If the encoder is not equipped with a Zero mark channel, set **Zero Mark pitch** value at 0.

Enter the **Resolution** parameter = 4 x number of encoder signal periods per shaft revolution for a rotating motor or number of encoder signal periods per pole pitch for a linear motor.

The parameters **HES type** and **Reverse HES tracks** are automatically calculated when the Auto-phasing procedure is performed.

Select **Reverse direction** in order to reverse the counting direction of the encoder, if required.

Adjust the encoder **Zero mark shift** and **Zero mark width** parameter values if the encoder has got a zero mark channel.

Hiperface encoder setting:

The command **Read encoder configuration** allows to read the encoder parameter values stored in the encoder memory via the Hiperface serial bus.

The parameter **Reverse incremental track** is manually identified according to the following procedure: move first the motor by hand. If the error “Encoder commutation channel / incremental channel” is released when moving the motor, then toggle the parameter **Reverse incremental track**.

Select **Reverse direction** in order to reverse the counting direction of the encoder, if required.

Position Feedback Selection

Select the position sensor currently mounted on the motor (resolver or encoder). The position sensor mounted on the motor is used by the drive for the motor torque or force control and for the speed regulation loop.

Select the position sensor to be used for the position regulation loop in the drive according to the application. Generally, the position regulation loop is using the motor position sensor (same sensor selection as in the previous case). However, for specific applications, the position regulation loop is using a second position sensor mounted directly on the mechanical load.

User Position Scaling

All internal position setpoints and displays are given by using the “user unit“ definition. All internal speed setpoints and displays are given by using the “user unit / s“ definition. So, it is necessary to define inside the drive the relationship between sensor data and “user unit” value.

Select the position unit according to the application.

Select the display factor according to the desired decimal number in the position set point and display.

Enter the load displacement value (in the previously defined position units) corresponding to one revolution for a rotating motor or one pole pitch for a linear motor. This parameter depends on the mechanical ratio between motor and load.

2.4.4 - SERVO LOOPS ADJUSTMENT

The **ServoPac** drive speed and position loop gain values can be automatically calculated by using the Autotuning procedure. This procedure identifies the motor and mechanical load specifications and calculates the appropriate gain values.

The Autotuning procedure can be executed with the drive disabled or enabled (for a vertical load). When the drive is enabled, the Auto-tuning can only be executed if the motor is at standstill.

Auto-tuning of the drive regulator

Select the **Controller type** according to the application:

- In Velocity mode, only the speed loop gains are calculated.

- In Position mode, all gains of both speed and position regulators are calculated.

Select the **Position loop requirements** if the position mode was selected before:

- The choice **Minimum following error** allows to get an accurate following of the position reference value during the whole motor displacement. In this case, all feedforward gain values are calculated.

- The choice **Minimum position overshoot** allows to get a motor positioning without any overshoot of the target position. In this case, all feedforward gain values are set at 0, and the motor position is lagging with regard to the position reference value during the whole motor displacement.

Select the **Speed measurement** filter time constant according to the motor position sensor resolution and the acceptable noise level in the speed measurement. The higher the time constant value, the lower the speed measurement noise, but also the lower the speed loop gains because of the increased speed measurement delay.
When **Auto-select** is selected, the most appropriate value is chosen during the Autotuning procedure execution.

Select the servo loop **Filter type** according to the application:

- The choice of the **Antiresonance** filter is necessary in case of loud noise in the motor, due to motor/load coupling elasticity.

- The choice of the **Maximum stiffness** filter allows to get the maximum stiffness on the motor shaft with regard to the torque disturbances. However, this choice is only possible without any resonance due to the motor/load coupling elasticity.

Select the desired closed loop **Bandwidth** (cut-off frequency value of the closed loop frequency response) according to the dynamic performances requirements of the application (Low = 50 Hz, Medium = 75 Hz, High = 100 Hz).

- **High** bandwidth means short response time of the servo loop and high gain values.

- **Low** bandwidth means larger response time of the servo loop and lower gain values.

Before executing the Autotuning procedure, check that the motor shaft is free and that its rotation over one revolution is not dangerous for operator and machine. Check also that the brake is released (the Autotuning command does not control the brake).

After the Autotuning, in case of loud noise in the motor at standstill or when running, check the rigidity of the mechanical transmission between motor and load (backlashes and elasticity in motor and couplings). If required, start a new Autotuning procedure by selecting a lower Bandwidth. If the instability remains, start a new Autotuning procedure by activating the Antiresonance filter. If necessary, adjust more accurately the loop response stability by adjusting the Gain scaling factor.

In case of loud noise in the motor, only when running, during the acceleration and deceleration phases, set **Feedforward acceleration gain** value at 0.

In the case of an axis with vertical load, proceed as follows:

- Select the **Limiting** current limitation mode (in order to avoid the drive being disabled in case of an I^2t protection release).

- Initialize the speed loop gains corresponding to the unloaded motor (execute therefore the Autotuning procedure with the motor uncoupled from its mechanical load).

- Couple the motor with its load. If possible, make a control in speed mode; otherwise, close the position loop with a stable gain.

- Move the axis until a stall position where one motor revolution is not dangerous for operator and machine (far enough from the mechanical stops).

- Then execute the Autotuning procedure with the motor at standstill. If the axis is moving, the Autotuning procedure has not been accepted by the drive.

Regulator gains

Speed loop gains are the most critical adjust because they greatly depend on the mechanical load characteristics (inertias, frictions, coupling stiffness, resonances,...).

- Proportional speed gain (KPv): defines the proportional gain of the controller which acts on the speed error. The higher this parameter value, the faster the speed loop response.

- Integral speed gain (KIv): defines the integral gain of the controller which acts on the speed error. The higher this parameter value, the better the axis stiffness.

- **Integrator low frequency limit (KIvf in Hz)**: defines the low frequency value from where the controller integrator term is saturated. This parameter is used for reducing the motor heating in applications with large dry frictions due to the mechanical load.

- Damping gain (KCv): defines the proportional gain of the controller which acts only on the speed feedback. This parameter allows to reduce the speed loop overshoot in response to a step like set point change .

- Derivative speed gain (KDv): defines the derivative gain of the controller which acts on the speed error.

- **Derivator high frequency limit (KDvf in Hz)**: defines the high frequency value from which the controller derivative term is saturated.

- **Gain scaling factor (KJv)**: defines a multiplying factor for all speed regulator gains. This parameter is scaling the speed regulator gains in order to avoid any saturation when large values are required. This parameter also allows to adjust the servo loop stability in case of load inertia changes.

The **Current command filter** is a 3rd order, low pass type, with 3 adjustable cut-off frequencies. Each cut-off frequency value can be freely adjusted according to the application for the filtering of high frequency noise or the filtering of mechanical resonances.

The **Speed measurement filter** is a 1st order, low pass type, with 3 selectable time constant values. The higher the time constant value, the lower the speed measurement noise, but also the lower the speed loop gains because of the increased speed measurement delay. The **Speed measurement filter** time constant is selected according to the motor position sensor resolution and the acceptable noise level in the speed measurement.

Position loop gains mainly influence the servo motor behaviour during the displacements (following error, position overshoot, audible noise, ...).

- Proportional position gain (KPp): defines the proportional gain of the controller which acts on the position error. The higher this parameter value, the better the axis stiffness and the lower the following error.

- Feedforward speed 1 gain(KFp): defines the feedforward speed amplitude corresponding to the speed input command. This term allows to reduce the following error during the motor displacement. Its value is set at maximum (65536) after the autotuning procedure, if a following error as small as possible is required.

- Feedforward speed 2 gain(KBv): defines the feedforward speed amplitude corresponding to the viscous frictions. This term allows to reduce the viscous friction effect during the motor displacement. The gain value is equal to the damping gain value + the viscous friction compensation term. After the autotuning procedure, the feedforward speed 2 gain is set equal to the damping gain value, if a following error as small as possible is required. The viscous friction compensation term can be calculated by measuring the current/speed ratio at various motor speed values.

- Feedforward acceleration gain(KAv): defines the feedforward acceleration amplitude corresponding to the acceleration input command. This term allows to reduce the following error during the motor acceleration and deceleration phases. Its value is calculated by the amplifier during the auto-tuning procedure if a following error as small as possible is required.

When the **autotuning** procedure is executed, the motor + mechanical load specifications are identified and the appropriate gain values are calculated according to the requirements selected by the user (controller type, filter type, bandwidth value, ...). All gain values can then be manually modified by the user if required.

Following error

Position error threshold defines the position following error triggering threshold. It is important to correctly adjust this value in order to get a good protection of the drive and the application.

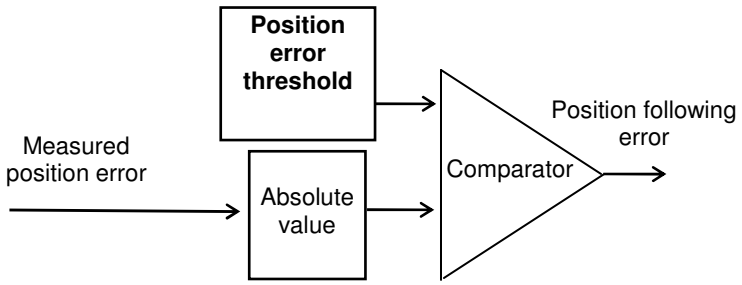
The **Position error threshold** parameter can be adjusted like follows:

- Make the motor running with the required operation cycles and measure the maximum value of the following error in the digital oscilloscope (max. following error value);

- Set then the **Position error threshold** parameter = 1.3 to 1.5 x Max. following error value.

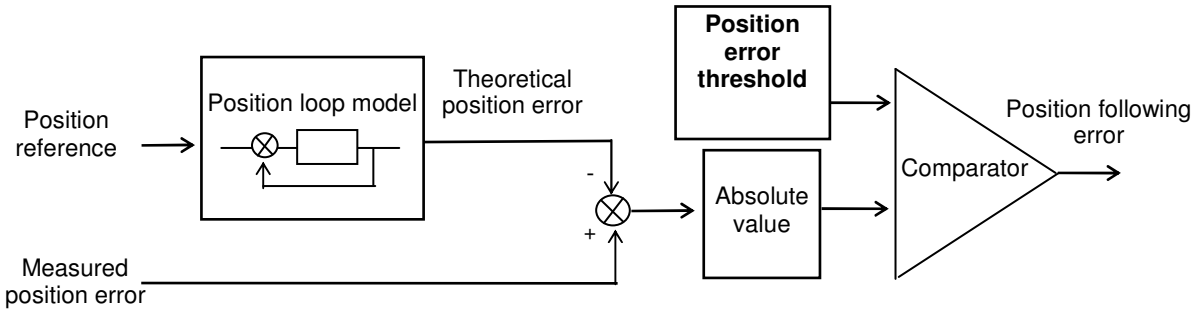
The **Position error detection mode** defines the operation mode of the axis following error protection.

- When **Absolute** is selected, the following error protection is operating as described below:



The measured position error value is continuously compared with the **Position error threshold** parameter value. When the measured position error is exceeding the **Position error threshold**, the position following error is released. This configuration is used for applications requiring the smallest possible following error.

- When **Relative to dynamic model** is selected, the following error protection is operating as described below:



The measured position error value is continuously compared with the theoretical position error given by the position loop model. When the difference is exceeding the **Position error threshold**, the position following error is released. In this configuration, when the position servo loop is adjusted to get the motor position continuously

lagging the reference position (applications for positioning without overshoot and with a large following error value), any small anomaly in the actuator behaviour can be detected.

2.4.5 - CONFIGURATION OF THE DRIVE ENABLE

When enable control by **SOFTWARE** is selected, the drive is enabled and disabled by using the control word (On/Off command in GemDriveStudio or fieldbus control).

When enable control by **HARDWARE** is selected, the drive is enabled and disabled by using the ENABLE logic input.

2.4.6 - QUICK TEST OF THE SERVO DRIVE

The servo loop stability can be tested on-line by moving the motor in speed profile mode or in position profile mode. The regulator gains can be manually optimised or by using the autotuning procedure.

Profile Velocity parameters

Enter the **Maximum velocity** parameter value according to the motor **Maximum speed** and the limitation due to the mechanical load in the application. For the first tests, a reduced velocity range is preferred in order to prevent hazardous movements with a wide amplitude. This parameter is active in both velocity profile mode and position profile mode.

Enter the **Acceleration** and **Deceleration** parameter values. Small values can be used as a starting point in order to prevent sharp movements on the mechanical load. This parameter is active in both velocity profile mode and position profile mode.

Profile Position parameters

Enter the **Maximum velocity** parameter value according to the motor **Maximum speed** and the limitation due to the mechanical load in the application. For the first tests, a reduced velocity range is preferred in order to prevent hazardous movements with a large amplitude. This parameter is active in both velocity profile mode and position profile mode.

Enter **Acceleration** and **Deceleration** parameter values. Small values can be used as a starting point in order to prevent sharp movements on the mechanical load. This parameter is active in both velocity profile mode and position profile mode.

Enter the **Profile velocity** parameter value according to the desired motor displacement speed. The **Profile velocity** parameter value must be lower than or equal to the **Maximum velocity** parameter value.

Checking the servo loop stability

In **velocity** mode:
Disable the motor brake, enable the drive, and check the servo loop stability at standstill: in case of loud noise in the motor, check the rigidity of the mechanical transmission between motor and load (backlashes and elasticity in motor and couplings). If required, start a new **Autotuning** procedure by selecting a lower **Bandwidth**. If the instability remains, start a new **Autotuning** procedure by activating the **Antiresonance** filter. If necessary, adjust more accurately the servo loop stability by adjusting the **Gain scaling factor**.

Move the axis in both directions (low velocity set point value), and check the servo loop stability in movement: in case of loud noise in the motor, during the displacement, the **Speed measurement filter** time constant can be increased. For high frequency noise or mechanical resonances, use the 3rd order low pass **Current command filter** and adjust the 3 cut-off frequencies with the most appropriate values.

Move the axis in both directions (higher velocity set point value), and check the servo loop time response. In case of undesired overshoot for a step-like velocity set point change, increase the **Damping speed gain** value and reduce the **Proportional speed gain** value accordingly.

In **position** mode:
Disable the motor brake, enable the drive, and check the servo loop stability at standstill: in case of loud noise in the motor, check the rigidity of the mechanical transmission between motor and load (backlashes and elasticity in motor and couplings). If required, start a new **Autotuning** procedure by selecting a lower **Bandwidth**. If the instability remains, start a new **Autotuning** procedure by activating the **Antiresonance** filter. If necessary, adjust more accurately the servo loop stability by adjusting the **Gain scaling factor**.

Move the axis in both directions with a low **Profile velocity** value, and check the servo loop stability in movement. In case of loud noise in the motor during the displacement, the **Speed measurement filter** time constant can be increased. For high frequency noise or mechanical resonances, use the 3rd order low pass **Current command filter** and adjust the 3 cut-off frequencies with the most appropriate values.

Move the axis in both directions with a higher **Profile velocity** value and check the motor positioning behaviour. In case of loud noise in the motor during the acceleration and deceleration phases, set **Feedforward acceleration gain** value at 0. In case of undesired position overshoot at the end of the deceleration phase, reduce the **Feedforward speed 1** value.

NOTE
In Profile velocity mode, only the speed regulator gains are active.
In Profile position mode, all gains of both speed and position regulators are active. However, if the Autotuning was executed in the Velocity mode, all the position loop gains are equal to 0 and the motor cannot move.
In Interpolated Position Mode, Feed forward Acceleration Gain must be manually cleared after Auto-tuning procedure.

2.4.7 - LOGIC INPUTS

ServoPac drives offer the use of built-in functions for the drive operation. These functions can be controlled by using “logical signal” or digital input. The default configuration is “logical signal”. If required, any digital input can be connected to a given function for the hardware control.

"ENABLE" INPUT
This function allows to enable and disable the drive when the enable control by HARDWARE is selected.

Remark: when a digital input is connected to this function for the hardware control, it is recommended to use a 24 Vdc signal on the input to enable the drive by choosing the appropriate value for the polarity parameter.

"INHIBIT" INPUT
The INHIBIT input must be deactivated in order to enable the drive by using the control word, when the enable control by SOFTWARE is selected. Activating the INHIBIT input during the operation will disable the drive.

Remark: when a digital input is connected to this function for the hardware control, it is recommended to use a 0 Vdc signal on the input to inhibit the drive by choosing the appropriate value for the polarity parameter.

"LIMIT SWITCH" INPUT
The "Limit switch" inputs are inputs for a detection sensor that allows to stop the motor with maximum deceleration. The purpose of both limit switches, when they are mounted at the right place on the axis stroke, is to protect the mechanics in case of uncontrolled movements.

The limit switches are only defined according to the motor hardware rotation. They are independent from the "rotation/counting direction" selection.

For checking the wiring of the limit switch inputs:
- move the motor in one direction,
- activate the limit switch placed in the rotation direction (artificially, if necessary),
- then check the motor stopping; if the motor goes on moving, reverse the wiring of the limit switch inputs.

Notes:
- When activating a limit switch input, the motor is stopped with maximum deceleration.
- The limit switch inputs must be setup to be activated if disconnected from the +24V potential.

"HOME SWITCH" INPUT
In Homing mode, according to the machine structure, it may be necessary to connect a digital sensor to identify the real position of an axis. In this case, a digital I/O has to be connected to this function. Home switch input is also a possible input for the capture function.

“CAPTURE" INPUT
The Capture function allows to record motor position and/or second sensor measurement when an external signal is changing.

“QUICK STOP” INPUT
Activating the QUICK STOP input during the operation makes the axis decelerate. At the end of the deceleration, the motor is maintained enabled at standstill.

“START PHASING” INPUT
The START PHASING input allows to start the motor phasing procedure at the drive power up when the motor is equipped with an incremental encoder without HES.

“ERROR RESET” INPUT
The ERROR RESET input allows to erase a released drive fault when the cause of the fault release is eliminated.

“SEQ START” INPUT
The SEQ START input allows to start the selected sequence when the drive Sequence mode is selected.

“SEQ STOP” INPUT
The SEQ STOP input allows to stop any sequence execution when the drive Sequence mode is selected.

“SEQ SEL 1” INPUT
The SEQ SEL 1 input is connected to the bit 0 of the sequence number selection when the drive Sequence mode is selected.

“SEQ SEL 2” INPUT
The SEQ SEL 2 input is connected to the bit 1 of the sequence number selection when the drive Sequence mode is selected.

“SEQ SEL 3” INPUT
The SEQ SEL 3 input is connected to the bit 2 of the sequence number selection when the drive Sequence mode is selected.

“SEQ SEL 4” INPUT
The SEQ SEL 4 input is connected to the bit 3 of the sequence number selection when the drive Sequence mode is selected.

“SEQ COND 1” INPUT
The SEQ COND 1 input can be used as a start condition or an end condition for a sequence when the drive Sequence mode is selected.

“SEQ COND 2” INPUT
The SEQ COND 2 input can be used as a start condition or an end condition for a sequence when the drive Sequence mode is selected.

“SEQ COND 3” INPUT
The SEQ COND 3 input can be used as a start condition or an end condition for a sequence when the drive Sequence mode is selected.

“SEQ COND 4” INPUT
The SEQ COND 4 input can be used as a start condition or an end condition for a sequence when the drive Sequence mode is selected.

2.4.8 - LOGIC OUTPUTS

Any drive state signal can be connected to a digital output.

"BRAKE" OUTPUT
This signal is useful for the motor brake control when the drive is enabled or disabled.

"FAULT" OUTPUT
This signal indicates that a fault is released inside the drive.

"WARNING" OUTPUT
This signal indicates that a warning is released inside the drive.

"VOTAGE ENABLED" OUTPUT
This signal indicates that the power supply is applied to the drive (Undervolt. is over).

"PHASING NOT OK" OUTPUT
This signal indicates that the motor is not ready to be enabled because a phasing or autophasing procedure is required.

"DRIVE ON" OUTPUT
This signal indicates that the motor is enabled and under servo control.

"IN POS" OUTPUT
This signal indicates that the motor has reached the target position when the drive Profile position or Sequence mode is selected.

"SEQ", "POS", "SPEED", "OUT1", "OUT2", "OUT3", "OUT4" OUTPUTS
These signals concern the sequence execution when the drive Sequence mode is selected.

"PULSE RX" OUTPUT
This signal indicates that a pulse train is received on the PULSE input when the drive Stepper emulation mode is selected.

2.5 - DRIVE PARAMETER SAVING

When all the adjustments and settings have been tested, they can be stored in the non volatile drive memory by selecting the command **Drive parameters file** >**Store parameters to flash memory**. In this case all the drive standard parameters are saved in the drive file DRIVEPAR.TXT.

The drive file DRIVEPAR.TXT can then be transferred in the project directory in the PC by selecting the command **Drive parameters file** > **Backup parameters to PC file**.

The command **Drive parameters file** > **Restore parameters** allows to transfer a file DRIVEPAR.TXT saved in the PC directory to the drive.

A user parameter list can also be edited and saved in the file USER_PAR.TXT by using the command **User parameters file** > **Edit Parameters**. The USER_PAR.TXT file can then be transferred to the drive by selecting the command **User parameters file** > **Restore parameters**. A drive file USER_PAR.TXT can be transferred from the drive to the PC directory by selecting the command **User parameters file** > **Backup parameters to PC file**. The user parameter file USER_PAR.TXT can be used for saving drive parameters that are not saved in the file DRIVEPAR.TXT (standard drive parameter list).

Remark: The commands **Tools** > **Drives files backup** and **Tools** > **Drives files restore** concern all project drive files: DRIVEPAR.TXT, USER_PAR.TXT, SEQUENCE.TXT, and so on.

2.6 - OSCILLOSCOPE

The oscilloscope can be started in the **Gem Drive Studio** software or in stand-alone mode.

This oscilloscope allows to display any drive signal by using the Index / Sub-index identification.

Four different channels are available to display signals. Multiaxis channel operation can be selected.

See **GemDriveStudio** quick start manual for more details.

2.7 - DIALOG TERMINAL

The dialog terminal can be started in the **Gem Drive Studio** software or in stand-alone mode.

This terminal allows to:
- Read a parameter value on a selected axis (continuous value monitoring can also be performed).
- Write a parameter value on a selected axis.

It is possible to read and/or write parameters on 4 different axes at the same time.

See **GemDriveStudio** quick start manual for more details.

Chapter 3 - Reference

REFERENCE

CiA DS-201..207	CAN Application Layer for Industrial Applications Version 1.1
CiA DS-301	Application Layer and Communication Profile Version 4.01
CiA DSP-402	Device Profile: Drive and Motion Control Version 1.1

DEFINITIONS & CONVENTIONS

CAN	Controller Area Network
CiA	CAN in Automation e. V. CAN-Bus international manufacturer and user organisation.
CAL	CAN Application Layer. The Application layer for CAN as specified by CiA.
COB	Communication Object is a CAN message. Data must be sent accross a CAN network inside a COB.
COB-ID	COB-Identifier. Each CAN message has a single identifier. There are 2032 different identifiers in a CAN network.
NMT	Network Management. One of the services of the application layer. It performs initialisation, configuration and error handling in a CAN network.
PDO	Process Data Object. A CANopen message used to exchange process data.
SDO	Service Data Object. A CANopen message for parameters setting.
pp	Profile Position Mode.
pv	Profile Velocity Mode.
hm	Homing Mode.
ip	Interpolated Position Mode.
tq	Profile Torque Mode.
pc	Position Control Function.
ServoPac	Generic name of a TRANSTECHNIK servo drive family with resolver and encoder feedback input.
Numerical value	hexa is preceded with 0x, decimal otherwise
Dynamic Variable	An element of an object indicated by index and sub-index which can be mapped in a PDO. An element of an object is addressed by its index and its sub-index.
Dataflow	An element of an object is qualified as dataflow (signal) if it is a variable (i.e. mappable). These variables can be of 8 bit, 16 bit or 32 bit. Depending on the using context, a dataflow must be of 16 bit or 32 bit or any size. The dataflow can come from: - An external source: Examples : Encoder position 0x3129-0 Analog Input 0x31F1-1 (16 bit) Analog Input 0x31F1-2 (32 bit)

- The CAN bus:
  Example: Interpolated data 0x30C1-0 (32 bit)

- An internal signal:
  Examples: Profile Speed Function Block output 0x3526-0 (32-bit)
  User variable : 0x3710-3 (32-bit)

3.1 - CANOPEN COMMUNICATION

3.1.1 - COMMUNICATION OBJECTS

3.1.1.1 - Can Telegram

CAN TELEGRAM

SOM	COB-ID	RTR	CTRL	Data segment	CRC	ACK	EOM

SOM	Start Of Message
COB-ID	COB-Identifier of 11 bits
RTR	Remote Transmission Request
CTRL	Control field
Data	up to 8 bytes
CRC	Cyclic Redundancy Check
ACK	Acknowledge
EOM	End Of Message

3.1.1.2 - Default COB-ID

The COB-ID is of 11 bits. Node-ID (bits 0 - 6) is the drive address from 1 to 127.

10	9	8	7	6	5	4	3	2	1	0
Function Code				NODE-ID						

Default COB-ID:

Broadcast objects of the pre-defined connection set:

Object	Function Code	Resulting COB-ID	Communication Parameter at Index
NMT	0000	0	-
SYNC	0001	128 (80h)	1005h, 1006h, 1007h

Peer-to-peer objects of the pre-defined connection set:

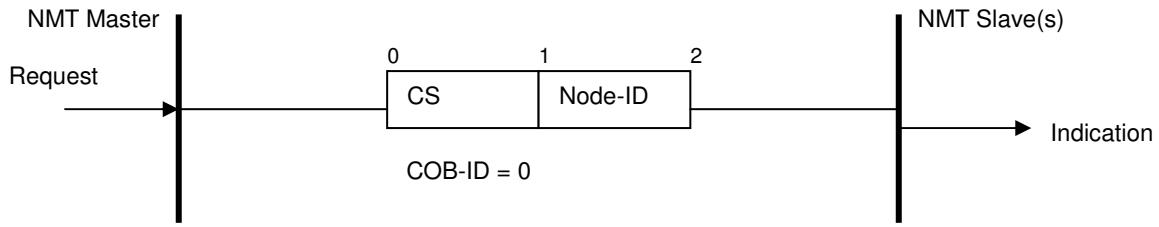
Object	Function Code	Resulting COB-ID	Communication Parameter at Index
EMERGENCY	0001	129 (81h) - 255 (FFh)	1014h
PDO1 (TX)	0011	385 (181h) - 511 (1FFh)	1800h
PDO1 (RX)	0100	513 (201h) - 639 (27Fh)	1400h
PDO2 (TX)	0101	641 (281h) - 767 (2FFh)	1801h
PDO2 (RX)	0110	769 (301h) - 895 (37Fh)	1401h
PDO3 (TX)	0111	897 (381h) - 1023 (3FFh)	1802h
PDO3 (RX)	1000	1025 (401h) - 1151 (47Fh)	1402h
PDO4 (TX)	1001	1153 (481h) - 1279 (4FFh)	1803h
PDO4 (RX)	1010	1281 (501h) - 1407 (57Fh)	1403h
SDO (TX)	1011	1409 (581h) - 1535 (5FFh)	1200h
SDO (RX)	1100	1537 (601h) - 1663 (67Fh)	1200h

TX = Transmit from drive to master

RX = Receive by drive from master

3.1.1.3 - Network Management Objects

NMT Protocols

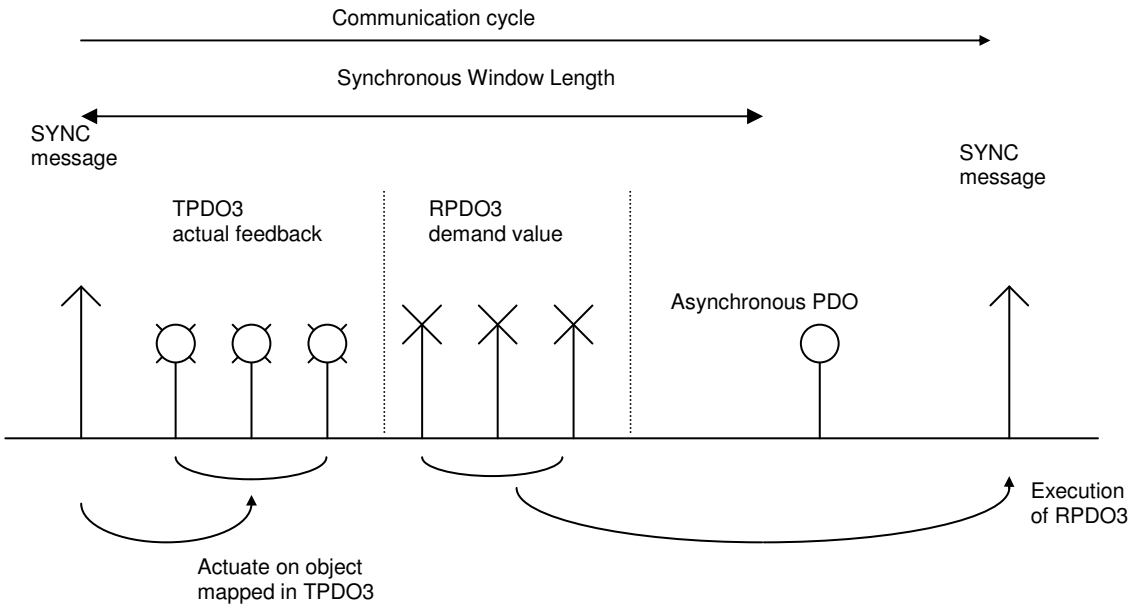


NMT Protocol	Command Specifier CS	Remarks
Start Remote Node	1	Change to NMT Operational state
Stop Remote Node	2	Change to NMT Stop state
Enter Pre-Operational	128	
Reset Node	129	
Reset Communication	130	

Node-ID: The Node-ID indicates the address of the drive. If Node_ID = 0, the protocol addresses all NMT slaves.

3.1.1.4 - Synchronisation Object

The SYNC object is a broadcast message sent by the master. This message provides a network clock. The period is specified by the communication cycle period (object 0x1006). The ServoPac servo-drives use this SYNC message to synchronize their local clock.
At least 180 ms are necessary for the servo-drive to start the synchronisation.



COB-ID Sync Message

Index	0x1005
Name	COB-ID Sync Message
Object Code	VAR
Data Type	Unsigned32
Object Class	all
Access	rw
PDO Mapping	No
Default Value	0x00000080

This object defines the COB-ID of the synchronisation object (SYNC).
The ServoPac drive does not support 29-bit ID.

Bit number	Value	Meaning
31 (MSB)		No Bootup message
30	0 1	Device does not generate SYNC message Device generates SYNC message
29	0	11-bit ID (CAN 2.0 A)
28-11	0	
10-0 (LSB)	x	bits 10-0 of SYNC COB-ID

Communication Cycle Period

Index	0x1006
Name	Communication Cycle Period
Object Code	VAR
Data Type	Unsigned32
Access	rw
PDO Mapping	No
Unit	µs
Value Range	0..20000 (only the values multiples of 500 are supported)
Default Value	10000

This object defines the communication cycle. This period is also used for the synchronisation in interpolated position mode. When the value of this object is reset at 0, the synchronisation is no more operative.

Sync Control

A PLL allows the internal cycle to be synchronized on SYNC message.

This object allows to adjust the PLL parameters.

Index	0x2006
Name	Sync control
Object Code	ARRAY
Number of Elements	4

Value Description

Sub Index	1
Description	Sync Phase defines the phase shift between local clock and SYNC
Data Type	Integer16
Object Class	all
Access	rw
PDO Mapping	No
Unit	µs
Default value	0

Sub Index	2
Description	Adjustment threshold. defines the limit to be applied to the adjustment.
Data Type	Unsigned16
Object Class	all
Access	rw
PDO Mapping	No
Unit	µs
Default value	20

Sub Index	3
Description	Adjustment value
Data Type	Unsigned16
Object Class	all
Access	rw
PDO Mapping	No
Unit	µs
Default value	2

Sub Index	4
Description	Sync Error Limit defines the triggering limit of the Sync error: | SyncPeriod - [0x1006-0] | < SyncErrorLimit
Data Type	Unsigned16
Object Class	all
Access	rw
PDO Mapping	No
Unit	µs
Default value	500

Sub Index	5
Description	Sync Filter applies a filter on Sync period measurement
Data Type	Unsigned16
Object Class	all
Access	rw
PDO Mapping	No
Value	0 disabled 1..4
Default value	0

3.1.1.5 - Process Data Objects (PDO)

PDOs are unconfirmed messages used for real-time data exchange.
PDOs sent by the master are RPDOs and PDOs sent by the drive are TPDOs.

Data in each PDO are defined by a list of objects (PDO mapping).

There are 4 pdos: TPDO1, RPDO1, TPDO2, RPDO2, TPDO3, RPDO3, TPDO4 and RPDO4.

Each PDO is defined by:

PDO communication parameters with
object 0x1400, 0x1401, 0x1402, 0x1403 for RPDOs
object 0x1800, 0x1801, 0x1802, 0x1803 for TPDOs

PDO mapping with
object 0x1600, 0x1601, 0x1602, 0x1603 for RPDOs
object 0x1A00, 0x1A01, 0x1A02, 0x1A03 for TPDOs

Communication parameters

Communication parameters are:
- PDO COB-ID
- Transmission type

The distribution of COB-ID is defined by default. The modification of COB-ID of PDO can be made in *NMT Pre-Operational State*; the new COB-ID will take effect when the NMT state machine changes to *Operation State*. The modification must not be taken in *NMT Operational State*, otherwise a Reset_Communication will be necessary before the new COB-ID takes effect.

Transmission type supported by ServoPac servo drives:

Transmission type	PDO transmission				
	cyclic	acyclic	synchronous	asynchronous	RTR only
1	TPDO1 TPDO2 TPDO3 TPDO4		TPDO1 TPDO2 TPDO3 TPDO4		
2-240					
253					TPDO1 TPDO2 TPDO3 TPDO4
254					
255				TPDO1 TPDO2 TPDO3 TPDO4	

- Transmission types 1 - 240 are synchronous transmissions with regard to the SYNC messages. A value between 1 and 240 means that the PDO is synchronously and cyclically transferred. The transmission type indicates the number of SYNC which are necessary to trigger PDO transmissions.
- Transmission type 253 means that the PDO is only transmitted on remote transmission request.
- Transmission type 255 is event trigger: The PDO will be transmitted when the first object (must be 16-bit) mapped in PDO has changed.

PDO transmission modes of:
- *Synchronous*: the message is transmitted in synchronisation with the SYNC message. A synchronous message must be transmitted within a pre-defined time-window immediately after the SYNC message.

- *Asynchronous*: the message is sent independently of the SYNC message.

Triggering modes:
- *Event _Driven*:
Message transmission by reception of SYNC.
Message transmission by specific event.

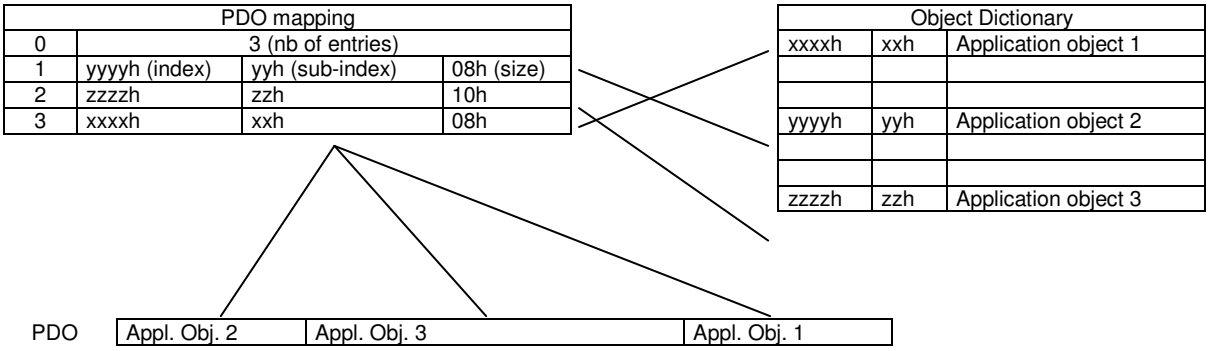
- *Remotely requested*: the transmission of an asynchronous PDO is initiated on reception of a remote request by any other device.

PDO Mapping
The sub-index 0 of mapping parameter contains the number of valid entries within the mapping record. This number of entries is also the number of application variables which shall be transmitted/received with the corresponding PDO. The sub-index 1 to number of entries contains the information about the mapped application variables. These entries describe the PDO contents by their index, sub-index and length (in bits).

Structure of PDO Mapping Entry:

Byte :	MSB		LSB
	index (16 bit)	sub-index (8 bit)	object length (8 bit)

Principle of PDO mapping:



Multiplexed data

The multiplexed data is used to multiplex more than one axis demand value into one message RPDOn.
It is possible to send 4 axis demand values (16 bit absolute) with one RPDOn. Therefore, the controller must modify the COB-ID of RPDOn of each axis to the same cob-ID. For example (see also the following diagram), for axis 1, object 60C1-1 is mapped into the first mapped object (object 1602-1), for axis 2, object 60C1-1 is mapped into the 2nd mapped object (object 1602-2) and so on... For each axis, the balance of the mapped objects must be mapped with a dummy object.

A dummy object mapped is realized with objects:

0x0002 (integer8)
0x0003 (integer16)
0x0004 (integer32)
0x0005 (unsigned8)
0x0006 (unsigned16)
0x0007 (unsigned32)

These objects can be used to map a PDO as a dummy object but cannot be accessed via SDO (see DS-301, 9.5.3 Data type entry specification).

Example of multiplexed data :

	MSB			LSB
TPDO Cob-ID 0x501	*Data_Ax4* (16bit)	*Data_Ax3* (16bit)	*Data _Ax2* (16bit)	*Data _Ax1* (16bit)

This PDO is transmitted with COB-ID 0x501 and contains 16bits x 4 of data

Object	Value
RPDO1 COB-ID (object 1400-1)	0x501
Number of mapped objects (object 1600-0)	0x1
1st Mapped Object (object 1600-1)	0x60C10110

In the drive 1, *"Data_Ax1"* will be written in the object 60C1-1

Object	Value
RPDO1 COB-ID (object 1400-1)	0x501
Number of mapped objects (object 1600-0)	0x2
1st Mapped Object (object 1600-1)	0x00060010 (dummy)
2st Mapped Object (object 1600-2)	0x60C10110

In the drive 2, *"Data _Ax2"* will be written in the object 60C1-1

Object	Value
RPDO1 COB-ID (object 1400-1)	0x501
Number of mapped objects (object 1600-0)	0x3
1st Mapped Object (object 1600-1)	0x00060010 (dummy)
2nd Mapped Object (object 1600-2)	0x00060010 (dummy)
3rd Mapped Object (object 1600-3)	0x60C10110

In the drive 3, *"Data _Ax3"* will be written in the object 60C1-1

Object	Value
RPDO1 COB-ID (object 1400-1)	0x501
Number of mapped objects (object 1600-0)	0x4
1st Mapped Object (object 1600-1)	0x00060010 (dummy)
2nd Mapped Object (object 1600-2)	0x00060010 (dummy)
3rd Mapped Object (object 1600-2)	0x37100110
4th Mapped Object (object 1600-4)	0x60C10110

In the drive 4, *"Data _Ax4"* will be written in the object 60C1-1 and *"Data _Ax3"* in the object 3710-1

Receive PDO Communication Parameter

Object 0x1400: 1st Receive PDO Communication Parameter

Index	**0x1400**
Name	1st Receive PDO Communication Parameter (RPDO1)
Object Code	RECORD
Number of Elements	2

Value Description

Sub Index	1
Description	COB-ID
Data Type	Unsigned32
Access	rw
PDO Mapping	No
Default Value	0x200 + Node-ID

Sub Index	2
Description	Transmission Type
Data Type	Unsigned8
Access	rw
PDO Mapping	No
Default Value	253

Object 0x1401: 2nd Receive PDO Communication Parameter

Index	**0x1401**
Name	2nd Receive PDO Communication Parameter (RPDO2)
Object Code	RECORD
Number of Elements	2

Value Description

Sub Index	1
Description	COB-ID
Data Type	Unsigned32
Access	rw
PDO Mapping	No
Default Value	0x300 + Node-ID

Sub Index	2
Description	Transmission Type
Data Type	Unsigned8
Access	rw
PDO Mapping	No
Default Value	253

Object 0x1402: 3rd Receive PDO Communication Parameter

Index	**0x1402**
Name	3rd Receive PDO Communication Parameter (RPDO3)
Object Code	RECORD
Number of Elements	2

Value Description

Sub Index	1
Description	COB-ID
Data Type	Unsigned32
Access	rw
PDO Mapping	No
Default Value	0x400 + Node-ID

Sub Index	2
Description	Transmission Type
Data Type	Unsigned8
Access	rw
PDO Mapping	No
Default Value	1

Object 0x1403: 4th Receive PDO Communication Parameter

Index	**0x1403**
Name	4th Receive PDO Communication Parameter (RPDO4)
Object Code	RECORD
Number of Elements	2

Value Description

Sub Index	1
Description	COB-ID
Data Type	Unsigned32
Access	rw
PDO Mapping	No
Default Value	0x500 + Node-ID

Sub Index	2
Description	Transmission Type
Data Type	Unsigned8
Access	rw
PDO Mapping	No
Default Value	0

Receive PDO Mapping

Object 0x1600: 1st Receive PDO Mapping

Index	**0x1600**
Name	1st Receive PDO Mapping
Object Code	RECORD
Number of Elements	0..4

Value Description

Sub Index	0
Description	number of mapped objects
Data Type	Unsigned8
Access	rw
PDO Mapping	No
Default Value	1

Sub Index	1
Description	1st mapped object
Data Type	Unsigned32
Access	rw
PDO Mapping	No
Default Value	0x60400010 (control word)

Object 0x1601: 2nd Receive PDO Mapping

Index	**0x1601**
Name	2nd Receive PDO Mapping
Object Code	RECORD
Number of Elements	0..4

Value Description

Sub Index	0
Description	number of mapped objects
Data Type	Unsigned8
Access	rw
PDO Mapping	No
Default Value	1

Sub Index	1
Description	1st mapped object
Data Type	Unsigned32
Access	rw
PDO Mapping	No
Default Value	0x60FF0020 (target velocity)

Object 0x1602: 3rd Receive PDO Mapping

Index	**0x1602**
Name	3rd Receive PDO Mapping
Object Code	RECORD
Number of Elements	0..4

Value Description

Sub Index	0
Description	number of mapped objects
Data Type	Unsigned8
Access	rw
PDO Mapping	No
Default Value	1

Sub Index	1
Description	1st mapped object
Data Type	Unsigned32
Access	rw
PDO Mapping	No
Default Value	0x60C10120 (Interpolated data record)

Object 0x1603: 4th Receive PDO Mapping

Index	**0x1602**
Name	4th Receive PDO Mapping
Object Code	RECORD
Number of Elements	0..4

Value Description

Sub Index	0
Description	number of mapped objects
Data Type	Unsigned8
Access	rw
PDO Mapping	No
Default Value	1

Sub Index	1
Description	1st mapped object
Data Type	Unsigned32
Access	rw
PDO Mapping	No
Default Value	

Transmit PDO Parameter

Object 0x1800: 1st Transmit PDO Parameter

Index	**0x1800**
Name	1st Transmit PDO Communication Parameter (TPDO1)
Object Code	RECORD
Number of Elements	2

Value Description

Sub Index	1
Description	COB-ID
Data Type	Unsigned32
Access	rw
PDO Mapping	No
Default Value	0x180 + Node-ID

Sub Index	2
Description	Transmission Type
Data Type	Unsigned8
Access	rw
PDO Mapping	No
Default Value	253

Object 0x1801: 2nd Transmit PDO Parameter

Index	**0x1801**
Name	2nd Transmit PDO Communication Parameter (TPDO2)
Object Code	RECORD
Number of Elements	2

Value Description

Sub Index	1
Description	COB-ID
Data Type	Unsigned32
Access	rw
PDO Mapping	No
Default Value	0x280 + Node-ID

Sub Index	2
Description	Transmission Type
Data Type	Unsigned8
Access	rw
PDO Mapping	No
Default Value	253

Object 0x1802: 3rd Transmit PDO Parameter

Index	**0x1802**
Name	3rd Transmit PDO Communication Parameter (TPDO3)
Object Code	RECORD
Number of Elements	2

Value Description

Sub Index	1
Description	COB-ID
Data Type	Unsigned32
Access	rw
PDO Mapping	No
Default Value	0x380 + Node-ID

Sub Index	2
Description	Transmission Type
Data Type	Unsigned8
Access	rw
PDO Mapping	No
Default Value	1

Object 0x1803: 4th Transmit PDO Parameter

Index	**0x1803**
Name	4th Transmit PDO Communication Parameter (TPDO4)
Object Code	RECORD
Number of Elements	2

Value Description

Sub Index	1
Description	COB-ID
Data Type	Unsigned32
Access	rw
PDO Mapping	No
Default Value	0x480 + Node-ID

Sub Index	2
Description	Transmission Type
Data Type	Unsigned8
Access	rw
PDO Mapping	No
Default Value	1

Transmit PDO Mapping

Object 0x1A00: 1st Transmit PDO Mapping

Index	**0x1A00**
Name	1st Transmit PDO Mapping
Object Code	RECORD
Number of Elements	0..4

Value Description

Sub Index	0
Description	number of mapped objects
Data Type	Unsigned8
Access	rw
PDO Mapping	No
Default Value	1

Sub Index	1
Description	1st mapped object
Data Type	Unsigned32
Access	rw
PDO Mapping	No
Default Value	0x60410010 (status word)

Object 0x1A01: 2nd Transmit PDO Mapping

Index	**0x1A01**
Name	2nd Transmit PDO Mapping
Object Code	RECORD
Number of Elements	0..4

Value Description

Sub Index	0
Description	number of mapped objects
Data Type	Unsigned8
Access	rw
PDO Mapping	No
Default Value	1

Sub Index	1
Description	1st mapped object
Data Type	Unsigned32
Access	rw
PDO Mapping	No
Default Value	0x606C0020 (velocity value)

Object 0x1A02: 3rd Transmit PDO Mapping

Index	**0x1A02**
Name	3rd Transmit PDO Mapping
Object Code	RECORD
Number of Elements	0..4

Value Description

Sub Index	0
Description	number of mapped objects
Data Type	Unsigned8
Access	rw
PDO Mapping	No
Default Value	1

Sub Index	1
Description	1st mapped object
Data Type	Unsigned32
Access	rw
PDO Mapping	No
Default Value	0x60640020 (Actual position value)

Object 0x1A03: 4th Transmit PDO Mapping

Index	**0x1A03**
Name	4th Transmit PDO Mapping
Object Code	RECORD
Number of Elements	0..4

Value Description

Sub Index	0
Description	number of mapped objects
Data Type	Unsigned8
Access	rw
PDO Mapping	No
Default Value	0

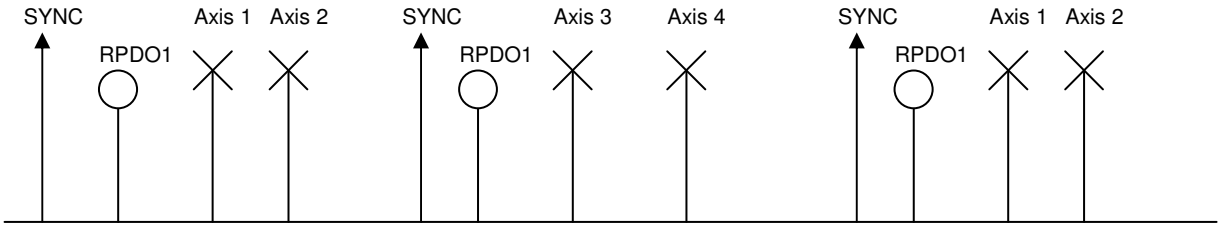
Sub Index	1
Description	1st mapped object
Data Type	Unsigned32
Access	rw
PDO Mapping	No
Default Value	0

Manufacturer PDO Transmission Mode

The **ServoPac** drive has a special transmission mode for the TPDOn defined by a TPDOn_Control (object 0x23A1-n) and a TPDO_Count (object 0x23A0). The purpose of this mode is to control the number of cyclic TPDOn for each axis.

TPDOn_Control is preset for each axis. TPDO_Count is counter value of the host. For each axis, when TPDO_Count is equal to TPDOn_Control, it will transmit the TPDOn in synchronisation with the SYNC message. The transmission type for the TPDOn must be 254.

Example: RPDO1 is used to transmit TPDO_Count value.
To be sure that all axes have got the same value of TPDO_Count at the same synchronisation, the RPDO1 COB-ID must be redefined to be the same for all axes and mapped with TPDO_Count object.



Index	**0x23A0**
Name	TPDO_Count
Object Code	VAR
Data Type	Unsigned8
Object Class	all
Access	rw
PDO Mapping	No
Value Range	0..255
Default Value	0

Index	**0x23A1**
Name	TPDO Control
Object Code	ARRAY
Number of Elements	4

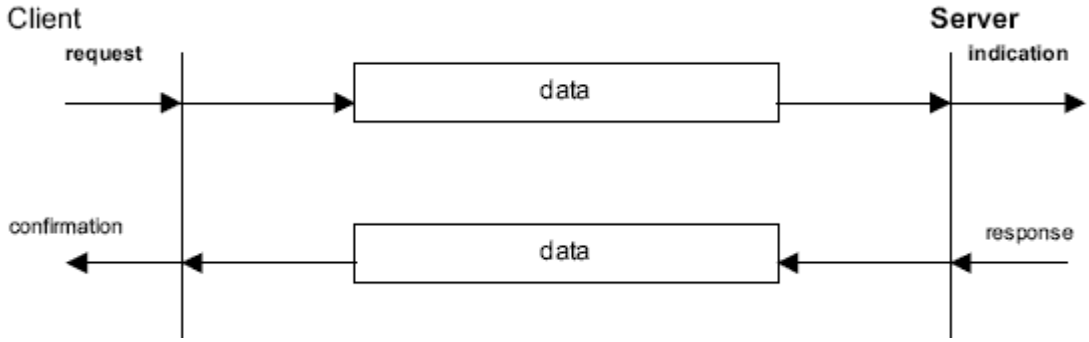
Value Description

Sub Index	1-4
Description	TPDO control for TPDO n.
Data Type	Unsigned8
Access	rw
PDO Mapping	No
Value Range	0..255
Default Value	0

3.1.1.6 - Service Data Objects (SDO)

The SDO is a communication channel with 2 basic characteristics:
- Client/Server relationship,
- Object Dictionary.

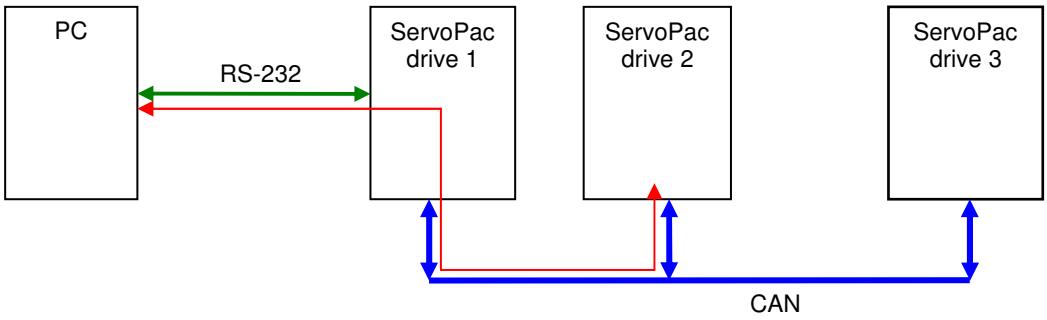
Client/Server:
This is a relationship between a single client and a single server (Servo Drive). A client issues a request (upload/download) thus triggering the server to perform a certain task. After finishing the task, the server answers the request.



Object Dictionary:
All objects (variables, constants, records...) of the server are defined as a list of objects where each element is appointed by an index and a sub-index. This list of objects is called object dictionary. This object dictionary allows the client the access to all objects of the server. The Servo Drive object dictionary consists of 2 parts: the communication profile (DS-301) for the objects related to the CAN communication and the device profile (DSP-402) for objects related to the drive functionality.

For more information about the SDO protocol, please report to the CiA DS-301 version 4.01 specification.

SDO Communication between drives



The ServoPac drive supports Node ID setting by switches from 1 to 63.

SDO message for node ID from 64 to 127 are used for communication between drives.
The ServoPac drive re-directs SDO message from RS-232 by PC to CANbus.

Example: 3 drives with Node ID 1, 2 and 3.
direct SDO messages: cobID = 0x601/0x581, 0x602/0x582 and 0x603/0x583
re-direct SDO messages: cobID = 0x641/0x5C1, 0x642/0x5C2 and 0x643/0x5C3

This allows the PC to communicate with any drive only via one RS-232 connection (example of the red line in the above diagram).

With an ServoPac drive with node ID = n, there must not be another device in the CANopen network with node ID = n+64, to avoid conflict with the re-direction SDO message of the ServoPac drive.

3.1.1.7 - Emergency Objects

Byte	0	1	2	3	4	5	6	7
Content	Emergency Error Code		Error register (object 1001h)	Manufacturer Specific Error Field				
				Error Code				

See object 0x3022 for the Error Code.

3.1.1.8 - Node Guarding

Network error behaviour

Index	0x205E
Name	Network Error Behaviour
Object Code	VAR
Data Type	Unsigned16
Object Class	all
Access	rw
PDO Mapping	No
Default Value	0

This object defines the drive behaviour when a Node guarding error occurs.

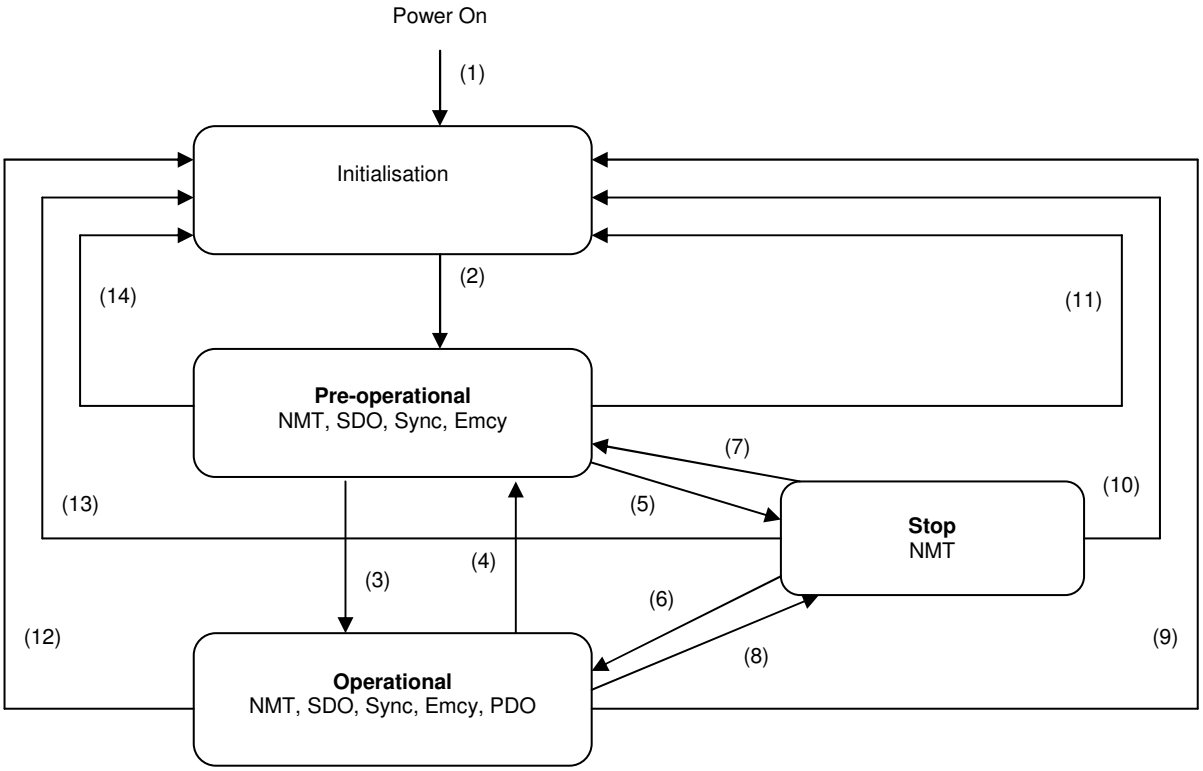
Value	Description
0	No Operation
1	Drive Error
2	Goes into Bus Stop state

See also NMT State machine

3.1.2 - NETWORK INITIALISATION

3.1.2.1 - NMT State Machine

The NMT state machine defines the communication status.



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

Minimum Boot-Up consists of one CAN telegram: a broadcast Start_Remote_Note message.

NMT reset

NMT_Reset_Comm:
The NMT_Reset_Comm restores communication parameters (default CobIDs, PDO mapping...) to the power-on values.

The NMT_Reset_Node:
Depending on object 0x205D, the NMT_Reset_Node can re-load the drive parameters file. An NMT_Reset_Comm is then executed.

NMT reset configuration

Index	0x205D
Name	NMT Reset configuration
Object Code	VAR
Data Type	Unsigned16
Object Class	all
Access	rw
PDO Mapping	No
Default Value	0

This object defines the bootup behaviour of the drive.

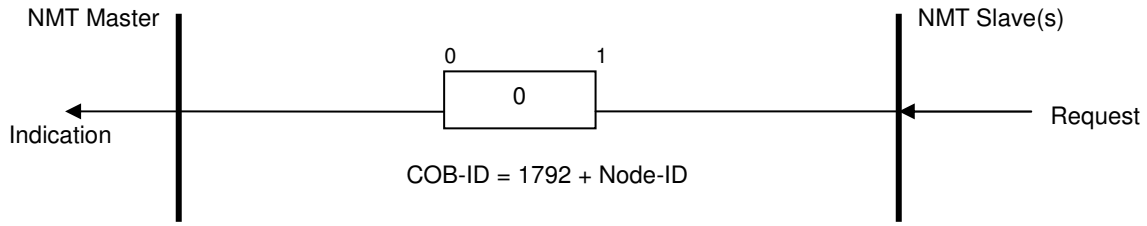
Bit	Description
0	0 -
	1 Load drive parameters file

See also NMT State machine.

3.1.2.2 - Bootup Protocol

This protocol is used to signal that a NMT slave has entered the node state PRE-OPERATIONAL after the state INITIALISING. The protocol uses the same identifier as the error control protocols.

Bootup Event



One data byte is transmitted with value 0.

Bootup configuration

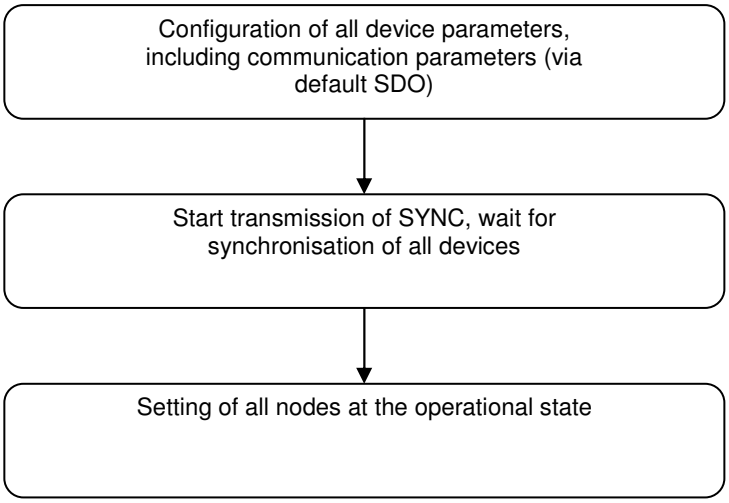
Index	0x2010
Name	Bootup configuration
Object Code	VAR
Data Type	Unsigned16
Object Class	all
Access	rw
PDO Mapping	No
Default Value	0

This object defines the bootup behaviour of the drive.

Value	Description
0	No Bootup message
1	Bootup message is sent when the drive goes into Pre-Op state

See also NMT State machine, Bootup protocol.

3.1.2.3 - Initialisation procedure

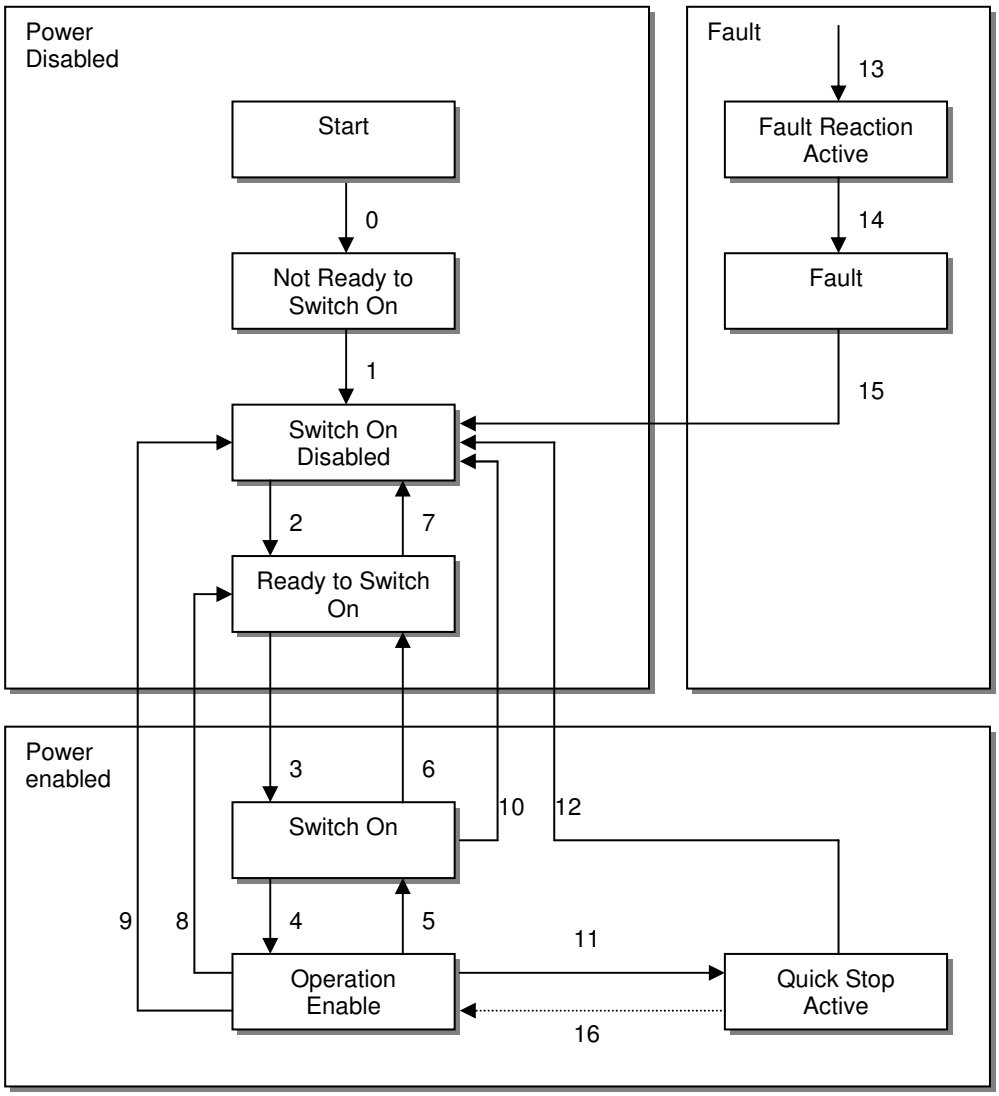


3.2 - DEVICE PROFILE

3.2.1 - DEVICE CONTROL

3.2.1.1 - Drive State Machine

The state machine describes the status and the control sequence of the drive.



Drive State

The following states of the device are possible:

• NOT READY TO SWITCH ON
Low level power has been applied to the drive.
The drive is being initialized or is running self test.
A brake, if present, has to be applied in this state.
The drive function is disabled.

• SWITCH ON DISABLED
Drive initialization is complete.
The drive parameters have been set up.
Drive parameters may be changed.
High voltage may not be applied to the drive, (e.g. for safety reasons).
The drive function is disabled.

• 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.

• OPERATION ENABLE
No faults have been detected.
The drive function is enabled and power is applied to the motor.
The drive parameters may be changed.
(This corresponds to normal operation of the drive.)

• QUICK STOP ACTIVE
The drive parameters may be changed.
The quick stop function is being executed.
The drive function is enabled and power is applied to the motor.

• FAULT REACTION ACTIVE
The drive parameters may be changed.
A fault has occurred in the drive.
The quick stop function is being executed.
The drive function is enabled and power is applied to the motor.

• FAULT
The drive parameters may be changed.
A fault has occurred in the drive.
High voltage switch-on/-off depends on the application.
The drive function is disabled.

State Transitions

State transitions are caused by internal events in the drive or by commands from the host via the *control word*.

• State Transition 0: START -> NOT READY TO SWITCH ON
Event: Reset.
Action: The drive self-tests and/or self-initializes.

• State Transition 1: NOT READY TO SWITCH ON -> SWITCH ON DISABLED
Event: The drive has self-tested and/or initialized successfully.
Action: Activate communication.

• State Transition 2: SWITCH ON DISABLED -> READY TO SWITCH ON
Event: 'Shutdown' command received from host.
Action: None

• State Transition 3: READY TO SWITCH ON -> SWITCHED ON
Event: 'Switch On' command received from host.
Action: The power section is switched on if it is not already switched on.

• State Transition 4: SWITCHED ON -> OPERATION ENABLE
Event: 'Enable Operation' command received from host.
Action: The drive function is enabled.

• State Transition 5: OPERATION ENABLE -> SWITCHED ON
Event: 'Disable Operation' command received from host.
Action: The drive operation will be disabled.

• State Transition 6: SWITCHED ON -> READY TO SWITCH ON
Event: 'Shutdown' command received from host.
Action: The power section is switched off.

• State Transition 7: READY TO SWITCH ON -> SWITCH ON DISABLED
Event: 'Quick Stop' and 'Disable Voltage' command received from host.
Action: None

• State Transition 8: OPERATION ENABLE -> READY TO SWITCH ON
Event: 'Shutdown' command received from host.
Action: The power section is switched off immediately, and the motor is free to rotate if unbraked.

• State Transition 9: OPERATION ENABLE -> SWITCH ON DISABLED
Event: 'Disable Voltage' command received from host.
Action: The power section is switched off immediately, and the motor is free to rotate if unbraked.

• State Transition 10: SWITCHED ON -> SWITCH ON DISABLED
Event: 'Disable Voltage' or 'Quick Stop' command received from host.
Action: The power section is switched off immediaetly, and the motor is free to rotate if unbraked.

• State Transition 11: OPERATION ENABLE -> QUICK STOP ACTIVE
Event: 'Quick Stop' command received from host.
Action: The quick stop function is executed.

• State Transition 12: QUICK STOP ACTIVE -> SWITCH ON DISABLED
Event: 'Quick Stop' is completed or 'Disable Voltage' command received from host.
This transition is possible, if the Quick-Stop-Option-Code is different from 5 (stay in the state 'Quick Stop Active').
Action: The power section is switched off.

• State Transition 13: All states -> FAULT REACTION ACTIVE
A fault has occurred in the drive.
Action: Execute appropriate fault reaction.

• State Transition 14: FAULT REACTION ACTIVE -> FAULT
Event: The fault reaction is completed.
Action: The drive function is disabled. The power section may be switched off.

• State Transition 15: FAULT -> SWITCH ON DISABLED
Event: 'Fault Reset' command received from host.
Action: A reset of the fault condition is carried out if no fault currently exists in the drive.
After leaving the state Fault the Bit 'Fault Reset' of the controlword *has to be cleared by the host.*

• State Transition 16: QUICK STOP ACTIVE -> OPERATION ENABLE
Event: 'Enable Operation' command received from host. This transition is possible if the Quick-Stop-Option-Code is 5, 6, 7 or 8.
Action: The drive function is enabled.

Objects definition

Index	Object	Name	Type	Attr.
0x6040	VAR	Control Word	Unsigned16	rw
0x6041	VAR	Status Word	Unsigned16	ro

Control Word

Index	0x6040
Name	Control Word
Object Code	VAR
Data Type	Unsigned16
Object Class	all
Access	rw
PDO Mapping	Possible
Default Value	0000

Bit Number	Function
0	Switch On
1	Disable Voltage
2	Quick Stop
3	Enable Operation
4	Operation Mode Specific
5	Operation Mode Specific
6	Operation Mode Specific
7	Reset Fault (rising edge)
8	Halt (mode PV, PT, AS, AT)

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

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

Bit 4, 5, 6 are operation mode specific:

Mode	Bit 4	Bit 5	Bit6
Profile Position Mode	new set point	change_set_immediately	0: absolute 1: relative
Homing Mode	Homing Operation Start	reserved	reserved
Interpolated Position Mode	enable ip_mode	reserved	reserved
Profile Velocity Mode	reserved	reserved	reserved

Correct sequence to enable the drive:

Seq	Control Word (0x6040)	Corresponding Status Word (0x6041)	Remarks
1	0x0000	0x0240	state "Switch On Disabled" drive is disabled
2	0x0006	0x0221	state "Ready To Switch On" drive is disabled
3	0x0007	0x0223	state "Switch On" drive is enabled
4	0x000F	0x0227	state "Operation Enable" drive is enabled

Notes:

- Some independent status bits may be set and are not represented in the table above. The mask for testing the status word is 0x026F
- Seq 1 (control word = 0x0000) and seq 3 (control word = 0x0007) may be omitted
- In some operation mode (interpolated position mode, servo mode...), the bit 4 of the control word must also be set after seq 4 to be fully operational. When switching between the modes, it is necessary to reset bit 4 of control word before changing the mode and then set it afterwards.

Status Word

Index	0x6041
Name	Status Word
Object Code	VAR
Data Type	Unsigned16
Object Class	all
Access	ro
PDO Mapping	Possible
Default Value	-

The status word indicates the current status of the drive. It is possible to define the TPDO to be transmitted at every change of the status word (Device Event transmission type).

Bit Number	Function
0	Ready to Switch On
1	Switch On
2	Operation Enabled
3	Fault
4	Voltage Enabled
5	Quick Stop
6	Switch On Disabled
7	Warning
8	
9	Remote
10	Target Reached
11	
12	Operation Mode Specific
13	Operation Mode Specific
14	
15	Manufacturer Specific: Drive Busy

Device Status Bit Meaning:

State	Bit 6 Switch On Disable	Bit 5 Quick Stop	Bit 3 Fault	Bit 2 Operation Enable	Bit 1 Switched On	Bit 0 Ready to Switch On
Not Ready to Switch On	0	X	0	0	0	0
Switch On Disabled	1	X	0	0	0	0
Ready to Switch On	0	1	0	0	0	1
Switched On	0	1	0	0	1	1
Operation Enable	0	1	0	1	1	1
Fault	0	X	1	0	0	0
Fault Reaction Active	0	X	1	1	1	1
Quick Stop Active	0	0	0	1	1	1

Bits 12, 13 are operation mode specific:

Mode	Bit 12	Bit 13	
Profile Position Mode	setpoint acknowledge	Following Error	
Homing Mode	Homing attained	Homing error	
Interpolated Position Mode	Ip-Mode active	reserved	
Profile Velocity Mode	Speed = 0	reserved	

3.2.1.2 - Error & Warning

3.2.1.2.1 - Error

Error:
Errors are displayed in object 0x3022,1 (32-bit) and 0x3022,2 (32-bit), each bit in this object correspond one error.
Error bit in status (bit 3) is set as well.
An emergency message is sent with the last error code (error code is error bit number+1).

The same bit in object 0x3025,1 and 0x3025,2 allows to inhibit the corresponding error in 0x3022,1 and 0x3022,2.

The same bit in object 0x3025,3 and 0x3025,4 allows to trigger a stop 1 when the corresponding error in 0x3022,1 and 0x3022,2 occurs.

The same bit in object 0x3025,5 and 0x3025,6 allows to trigger a stop 3 when the corresponding error in 0x3022,1 and 0x3022,2 occurs.

An error can be cleared by "Reset Fault" bit in control word (0x6040).

Error control:
Object 0x3025 allows to
- inhibit certain errors
- or trigger a stop 1 or stop 3 when the corresponding error occurs.

Index	Object	Name	Type	Attr.
0x3022	ARRAY	Error		ro
0x3025	ARRAY	Error Control		rw

Index	0x3022
Name	Error word
Object Code	ARRAY
Number of Elements	3

Value Description
This object contains two 32-bit words in which one bit is assigned to a difference error.
The Error code is the value which will be sent as an emergency message (EMCY).

Sub Index	1
Description	Error monitoring
Data Type	Unsigned32
Object Class	all
Access	ro
PDO Mapping	No
Value	See below
Default value	No

Bit	Value	Error Code	Protection	Troubleshooting
0	0x00000001	1	Hardware System 2 Error	-Check that the DNC/PLC-amplifier-motor ground connections and shield answer the Installation manual requirements. -Check the application EMC disturbances level.
1	0x00000002	2	24 Volt Error	-Check that the logic supply voltage value is within the specified range. -Check the logic supply voltage waveform (ripple value, over-voltage spikes, under-voltage spikes, …)
2	0x00000004	3	Undervolt (temporized)	-Check that the power supply is actually on.
3	0x00000008	4	Braking system error	-Check the presence of either the internal resistor jumper (ServoPac) or the external resistor (ServoPAC and ServoPac-B) -Check that the external resistor is not broken (open circuit) If the error cannot be reset, the braking system is out of order (transistor in short-circuit)
4	0x00000010	5	Safety channel 2 Error	-Check the correct STO2 input state with regard to the STO1 input state If the STO fault is released, the drive must be turned off in order to cancel the fault.
5	0x00000020	6	Overvoltage	If the failure occurs when starting the amplifier: -Check the AC supply voltage value. If the failure occurs during the operation: -Check the DC bus voltage during the deceleration phases. -Check the sizing of the braking resistor with regard to the motor deceleration phases.
6	0x00000040	7	Internal Communication 2 Error	-Check that the DNC/PLC-amplifier-motor ground connections and shield answer the Install manual requirements. -Check the application EMC disturbance level.
7	0x00000080	8	IGBT module	-Check for no short-circuit in the motor wiring and at the motor terminals. -Check for no short-circuit between one motor phase and the ground. -Check the amplifier **Rated current** adjustment with regard to the allowed value in the amplifier specifications. -Check that the amplifier max. temperature specifications are fulfilled. -Check that the amplifier fan is operating correctly.
8	0x00000100	9	Main Phase Error	
9	0x00000200	10	Mains phase loss	
10	0x00000400	11	Power Module over-temperature	-Check the amplifier **Rated current** adjustment with regard to the allowed value in the amplifier specifications. -Check that the amplifier max. temperature specifications are fulfilled. -Check that the amplifier fan is operating correctly.
11				
12	0x00001000	13	Fan	-Available only for some drive models -Check that the fan blades are not blocked by a foreign body -Check that the fan rotor is not locked
13				
14				
15				
16	0x00010000	17	Current measurement offset	-Check that the motor is not driven by the mechanical load. If the error cannot be reset, the amplifier current sensors are out of order (wrong current measurement)
17	0x00020000	18	Overcurrent	-Check the current loop adjustment with regard to the motor inductance.
18	0x00040000	19	Encoder counting error	-Check that the encoder max. pulse frequency at max. motor speed meets the encoder specification. -Check that the connections between the encoder and the amplifier are complying with the shield wiring recommendations. Remark: In the incremental encoder configuration without HES, the motor **Phasing** procedure must be executed again after a Counting fault release.
19	0x00080000	20	Resolver tracking error	If the failure occurs when starting the amplifier: -Check for the correct resolver type with regard to the amplifier

				specifications. If the failure occurs during the operation: -Check that the connections between the resolver and the amplifier are complying with the shield wiring recommendations.
20	0x00100000	21	Resolver (cable interrupted)	-Check the resolver connection on the amplifier X1 connector according to the connector descriptions. -Check for the correct resolver type with regard to the amplifier specifications. -Check the connections between resolver and amplifier (cable wiring).
21	0x00200000	22	Encoder (cable interrupted)	-Check the encoder supply connection on the amplifier X3 connector. -Check the encoder A channel and B channel connections on the amplifier X3 connector. Remark: In the Incremental encoder configuration without HES, the motor **Phasing** procedure must be executed again after an Encoder fault release.
22	0x00400000	23	Encoder (Z marker)	-Check the marker pulse connection on the amplifier X3 connector. If the motor encoder is not providing a marker pulse channel, the amplifier counting protection must be disabled by setting at 0 the **Zero mark pitch** parameter. -Check that the **Motor encoder resolution** and the **Zero mark pitch** parameter values are correct. Remark: In the incremental encoder configuration without HES, the motor **Phasing** procedure must be executed again after a Counting fault release.
23				
24				
25				
26				
27	0x08000000	28	Power Stage Controller Error	-Generic default for the amplifier power stage
28	0x10000000	29	Manufacturer parameters error	-Switch off and on again the 24V logic supply. If the error cannot be reset, the amplifier is out of order.
29	0x20000000	30	Internal Communication 1 error	-Check that the DNC/PLC-amplifier-motor ground connections and shield answer the Installation manual requirements. -Check the application EMC disturbance level.
30	0x40000000	31	Configuration error	
31	0x80000000	32	System error	-Switch off and on again the 24V logic supply. If the error cannot be reset, the amplifier is out of order.

Sub Index	2
Description	Error monitoring
Data Type	Unsigned32
Object Class	all
Access	ro
PDO Mapping	No
Value	See below
Default value	No

Bit	Value	Error Code	Protection	Troubleshooting
0				
1				
2	0x00000004	35	Position following error	-Check that the mechanical load is adjusted to motor and amplifier ratings. -Reduce the accelerations/decelerations. -Check that the axis is not on a mechanical limit. -Check the position loop adjustment. -Check that the value of the parameter **Following error threshold** is complying with the motion cycle. If necessary, increase the value of this parameter.
3				
4	0x00000010	37	Motor Temperature error	If the failure occurs when starting the amplifier: -Check the selected thermal sensor type (NTC or PTC). -Check the connection between the thermal sensor and the amplifier on the X1 or X3 connector. If the failure occurs during the operation: -Check the motor temperature and look for the reason of this overheating (mechanical shaft overload, duty cycle too high, motor type to small with regard to the machine cycle…).
5	0x00000020	38	I²t error	Check the amplifier current cycle with regard to the **Rated current** parameter value.
6				
7	0x00000080	40	Busy	
8	0x00000100	41	Calibration parameters file error	If the firmware has been downgraded, reload the correct firmware version. If the error cannot be reset after the amplifier has been turned off and on again, the sequence it is faulty.
9	0x00000200	42	Drive parameters file error	If the firmware has been downgraded, reload the correct firmware version. If the procedure “save parameter to Flash memory” is executed, some parameters will be definitely lost.
10	0x00000400	43	User parameters file error	Edit and check the “User parameter file”. Some objects are not compatible with the amplifier firmware version.
11	0x00000800	44	Sequence file error	Check the Sequence file. Some parameters are not compatible with the amplifier firmware version.
12	0x00001000	45	Cam file error	
13	0x00002000	46	Extension Error or Fieldbus watchdog error	
14	0x00004000	47	Extension Error or Fieldbus hardware error	
15	0x00008000	48	Extension Error or Fieldbus hardware error	
16	0x00010000	49	Fieldbus SYNC cycle error	- Check fieldbus cycle period (object 0x1006) - Check fieldbus SYNC signal timing: if great jitter (>=half-period) or period, accuracy is not within the tolerance (>=0.4%).
17	0x00020000	50	Fieldbus IP reference underflow/overflow	- Check if IP reference (0x60C1,1) is mapped in a RPDO - If yes, check if this RPDO is sent every bus cycle - To avoid mix-up, this RPDO must precede SYNC signal at least 100 µs
18	0x00040000	51	Fieldbus guarding error	For CANopen: Node guarding error or Heartbeat error.
19				
20	0x00100000	53	SD card error	See details in the SDcard chapter.
21	0x00200000	54	File Erase/Write Error	Renew the file transfer.
22	0x00400000	55	Computation overflow	
23	0x00800000	56	Safety channel 1 Error	-Check the correct STO1 input state with regard to the STO2 input state. If the STO fault is released, the drive must be turned off in order to cancel the fault.
24	0x01000000	57	User Program Error	

25				
26				
27				
28	0x10000000	61	Encoder Commutation channel / Incremental channel Error	For the Incremental encoder & HES configuration: -Check for the correct HES supply voltage value. -Check that the HES are correctly wired on the amplifier X3 connector. -Check the parameter **Reverse HES track** and toggle it if not correct. -Check for the correct **Motor encoder resolution** parameter value. -Check that the HES-amplifier-motor ground connections and shield answer the Install manual requirements. For the Absolute encoder (Hiperface) configuration: -Check the parameter **Reverse incremental track** and toggle it if not correct. -Check that the SinCos channels are correctly wired on the amplifier X3 connector. -Check that the Data communication channel is correctly wired on the amplifier X3 connector. -Check that the encoder-amplifier-motor ground connections and shield answer the Installation manual requirements. For the SinCos encoder with CD tracks configuration: -Check for the correct SinCos encoder supply voltage value. -Check that the encoder CD channels are correctly wired on the amplifier X3 connector. -Check the parameter **Reverse CD track** and toggle it if not correct. -Check that the **Motor encoder resolution** parameter value is correct. -Check for the correct encoder C and D channels signal waveforms. -Check that the encoder-amplifier-motor ground connections and shield answer the Installation manual requirements.
29	0x20000000	62	Encoder Absolute channel Error	-Check for the correct encoder supply voltage value. -Check that the Data communication channel is correctly wired on the amplifier X3 connector. -Check that the encoder-amplifier-motor ground connections and shield answer the Installation manual requirements.
30	0x40000000	63	User Program execution error	
31	0x80000000	64	Procedure error (Autotuning, autophasing...)	-If the Procedure fault is continuously displayed after the execution of the **AUTOPHASING** function, the procedure has failed because of an external cause and the calculated parameters are wrong. Check that the limit switch inputs are not active. Then check that the motor is unloaded and the shaft movement free during the procedure. -If the Procedure fault is continuously displayed after the execution of the **AUTOTUNING** function, the procedure has failed because of an external cause and the calculated parameters are wrong. Check that the limit switch inputs are not active. Then check that the motor shaft is free during the procedure.

Error Control

Index	**0x3025**
Name	Error control
Object Code	ARRAY
Number of Elements	6

Value Description

Sub Index	1
Description	Error mask 1
Data Type	Unsigned32
Object Class	all
Access	rw
PDO Mapping	No
Value	See 0x3022-1
Default value	No

Sub Index	2
Description	Error mask 2
Data Type	Unsigned32
Object Class	all
Access	rw
PDO Mapping	No
Value	See 0x3022-2
Default value	No

These 2 elements (0x3025,1 and 0x3025,2) allow to inhibit the corresponding error.

Sub Index	3
Description	Error Stop 1 mask 1
Data Type	Unsigned32
Object Class	all
Access	rw
PDO Mapping	No
Value	See 0x3022-1
Default value	No

Sub Index	4
Description	Error Stop 1 mask 2
Data Type	Unsigned32
Object Class	all
Access	rw
PDO Mapping	No
Value	See 0x3022-2
Default value	No

These 2 elements (0x3025,3 and 0x3025,4) allow to trigger a stop 1 when the corresponding error occurs.

Sub Index	5
Description	Error Stop 3 mask 1
Data Type	Unsigned32
Object Class	all
Access	rw
PDO Mapping	No
Value	See 0x3022-1
Default value	No

Sub Index	6
Description	Error Stop 3 mask 2
Data Type	Unsigned32
Object Class	all
Access	rw
PDO Mapping	No
Value	See 0x3022-2
Default value	No

These 2 elements (0x3025,5 and 0x3025,6) allow to trigger a stop 3 when the corresponding error occurs.

3.2.1.2.2 - Warning

Warning:
Warning is displayed in object 0x3024,0 (32-bit).
Warning bit in status (bit 7) is also set.
Warning can not be cleared by user, it will automatically cleared when the origin of the warning is discarded.

Warning Code

Index	**0x3024**
Name	Warning Code
Object Code	VAR
Data Type	Unsigned32
Object Class	all
Access	ro
PDO Mapping	Possible
Default Value	0

Bit	**Mask**	**Function**
0	0x00000001	STO active
9	0x00000200	Mains phase loss
10	0x00000400	IGBT module temperature
11		
12	0x00001000	Fan
13	0x00002000	Daughter boad/Plugin software incompatible
14	0x00004000	Daughter boad/Plugin hardware not ready
15	0x00008000	Daughter boad/Plugin software not ready
16	0x00010000	Limit Switch
17		
18	0x00040000	I^2t
19		
20		
21	0x00200000	Motor temperature
22		
23		
24	0x01000000	Position limit
25		
26		
27		
28	0x10000000	Multi-turn Absolute Encoder not Initialized
29	0x20000000	can not read/write to encoder
30	0x40000000	Motor phasing Init not ok
31		

3.2.1.2.3 - I²t Protection

I²t Function

Index	0x3404
Name	I²t Function
Object Code	RECORD
Number of Elements	

Value Description

Sub Index	1
Description	I²t Mode
Data Type	Unsigned16
Access	rw
PDO Mapping	No
Value Range	0 Limiting 1 Fusing
Default Value	

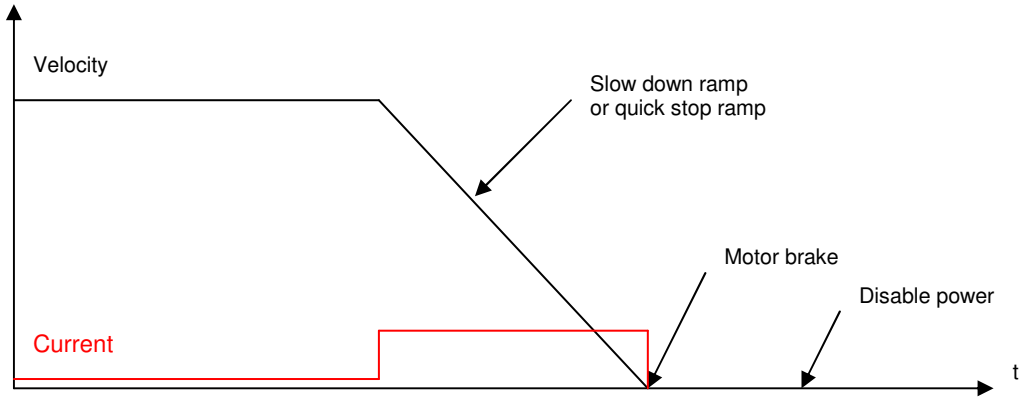
Sub Index	2
Description	I²t signal
Data Type	Unsigned16
Access	rw
PDO Mapping	No
Default Value	No

The motor RMS current value in Amps is calculated according to the following formula:
RMS motor current (A) = Amplifier current rating (A) x [value(0x3404-2) x 5000 / 16384]$^{1/2}$ / 100

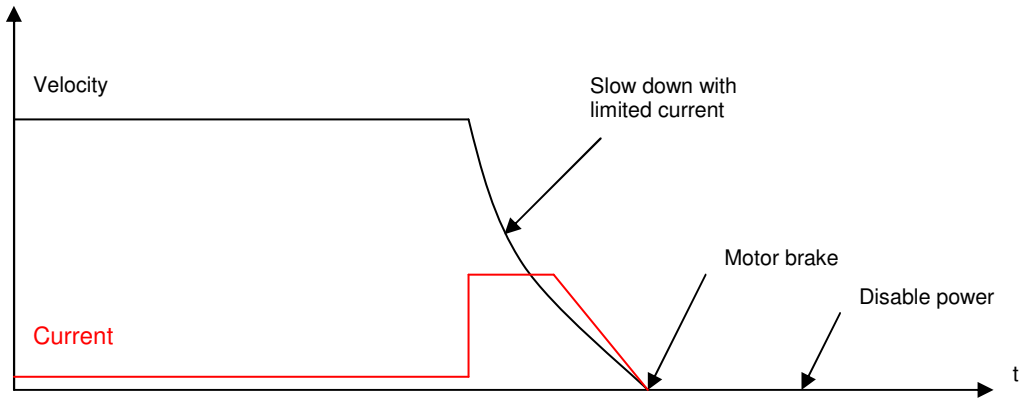
Sub Index	3
Description	Continuous measurement of the current
Data Type	Unsigned16
Access	rw
PDO Mapping	No
Unit	% of drive max current (0x6510) (0x3FFF = 100% Imax)
Default Value	No

3.2.1.3 - Stop Operation

With stop on speed ramp, the motor is slowed down in position loop with a slow down ramp.



With stop on current limit, the motor is slowed down in velocity loop with a current limitation.



Stop option code	Action
0	Disable drive
1	Stopped on Slow down speed ramp and disabled
2	Stopped on Quick Stop speed ramp and disabled
3	Stopped on current limit and disabled
5	Stopped on Slow down speed ramp and stay in Quick Stop state
6	Stopped on Quick Stop speed ramp and stay in Quick Stop state
7	Stopped on current limit and disabled and stay in Quick Stop state

When a transition of the state machine occurs, a stop can be performed. These transition are:
- Quick Stop (transition 11)
- Disable Operation (transition 5)
- Shut down (transition 8)

Each transition can has a difference way to stop defined respectively in object 0x605A, 0x605C and 0c605B.

The Inhibit input stops the drive with parameter defined in object 0x305A.

Hardware limit switches stops with slow down speed ramp (with parameter in 0x3300,1)

Stop on current limit uses the current limit value defined in object 0x3301,1
Stop on slow down speed ramp uses the speed ramp defined in object 0x3300,1
Stop on quick stop speed ramp uses the speed ramp defined in object 0x6085,0

Object definitions

Index	Object	Name	Type	Attr.
0x605A	VAR	Quick Stop Option Code	Integer16	rw
0x605B	VAR	Shut down Option Code	Integer16	rw
0x605C	VAR	Disable Operation Option Code	Integer16	rw
0x305A	VAR	Inhibit Option Code	Integer16	rw
0x3300	ARRAY	Slow down ramp	Unsigned32	rw
0x6085	VAR	Quick Stop ramp	Unsigned32	rw
0x3301	ARRAY	Stop Current Limit	Integer16	rw
0x3302	ARRAY	Stop Time Limit	Unsigned16	rw
0x3304	VAR	Amplifier Reaction Time	Unsigned16	rw
0x3305	VAR	Motor Brake Reaction Time	Unsigned16	rw

Quick Stop Option Code

Index	0x605A
Name	Quick Stop Option Code
Object Code	VAR
Data Type	integer16
Object Class	all
Access	rw
PDO Mapping	No
Default Value	1

This object defines the stop behaviour when a QUICK_STOP command is executed (see Drive State Machine transition 11).

Quick stop option code	Action
0	Disable drive
1	Stopped on Slow down speed ramp and disabled
2	Stopped on Quick Stop speed ramp and disabled
3	Stopped on current limit and disabled
5	Stopped on Slow down speed ramp and stay in Quick Stop state
6	Stopped on Quick Stop speed ramp and stay in Quick Stop state
7	Stopped on current limit and disabled and stay in Quick Stop state

Shut Down Option Code

Index	0x605B
Name	Shut Down Option Code
Object Code	VAR
Data Type	integer16
Object Class	all
Access	rw
PDO Mapping	No
Default Value	0

This object defines the stop behaviour when a SHUTDOWN command is executed (see Drive State Machine transition 8).

Quick stop option code	Action
0	Disable operation
1	Stopped on Slow down speed ramp
2	Stopped on Quick Stop speed ramp
3	Stopped on current limit

Disable Operation Option Code

Index	0x605C
Name	Disable Operation Option Code
Object Code	VAR
Data Type	integer16
Object Class	all
Access	rw
PDO Mapping	No
Default Value	1

This object defines the stop behaviour when a DISABLE_OPERATION command is executed (see Drive State Machine transition 5).

Quick stop option code	Action
0	Disable operation
1	Stopped on Slow down speed ramp
2	Stopped on Quick Stop speed ramp
3	Stopped on current limit

Inhibit Option Code

Index	0x305A
Name	Inhibit Option Code
Object Code	VAR
Data Type	integer16
Object Class	all
Access	rw
PDO Mapping	No
Default Value	1

This object defines the stop behaviour when a Inhibit logic input is activated (see Digital Inputs 0x60FD).

Quick stop option code	Action
0	Disable drive
1	Stopped on Slow down speed ramp and disabled
2	Stopped on Quick Stop speed ramp and disabled
3	Stopped on current limit and disabled

Slow Down Ramp

Index	0x3300
Name	Slow Down Ramp
Object Code	ARRAY
Number of Elements	2

These parameters define the time limit for a stop operation.

When a stop on current limit is executed, the end of the stop may not be detected correctly if the axis is oscillating. The time stop limit allows to limit the execution time of the stop operation.

Value Description

Sub Index	1
Description	Slow Down Ramp 1
Data Type	Unsigned32
Object Class	all
Access	rw
PDO Mapping	No
Unit	Acceleration unit
Default Value	

These parameters define the slow down deceleration with a stop executed with stop option code = 1 or 5 (Stopped on Slow down ramp).

Sub Index	2
Description	Slow Down Ramp 2 reserved for future use.
Data Type	Unsigned32
Object Class	all
Access	rw
PDO Mapping	No
Unit	Acceleration unit
Default Value	

Quick Stop Ramp

Index	**0x6085**
Name	Quick Stop Ramp
Object Code	VAR
Data Type	Unsigned32
Object Class	all
Access	rw
PDO Mapping	No
Unit	Acceleration unit
Default Value	0x00200000

This object defines the deceleration for a quick stop with Quick Stop Option Code = 2 or 6 (Stopped on Quick Stop ramp).

Stop Current Limit

Index	**0x3301**
Name	Stop Current Limit
Object Code	ARRAY
Number of Elements	2

Value Description

Sub Index	1
Description	Stop Current Limit 1 This parameter defines the current limit when a stop on current limit is performed.
Data Type	Unsigned16
Object Class	all
Access	rw
PDO Mapping	No
Unit	per thousand of rated current
Value Range	100..6000
Default Value	1000

This parameter is used with a Quick Stop with Quick Stop Option Code = 3 or 7 (Stopped on current).
This parameter is also applied with a stop at limit switches.

Sub Index	2
Description	Stop Current Limit 2
Data Type	Unsigned16
Object Class	all
Access	rw
PDO Mapping	No
Unit	per thousand of rated current
Value Range	100..6000
Default Value	1000

This parameter is reserved for future use.

Stop Time Limit

Index	0x3302
Name	Stop Time Limit
Object Code	ARRAY
Number of Elements	2

These parameters define the time limit for a stop operation.

When a stop on current limit is executed, the end of the stop may not be detected correctly if the axis is oscillating. The time stop limit allows to limit the execution time of the stop operation.

Value Description

Sub Index	1
Description	Stop Time Limit 1 Time limit for all stop operations with ramp.
Data Type	Unsigned16
Object Class	all
Access	rw
PDO Mapping	No
Unit	ms
Value Range	0..65000
Default Value	1000

Sub Index	2
Description	Stop Time Limit 2 Time limit for all stop operations with current limit.
Data Type	Unsigned16
Object Class	all
Access	rw
PDO Mapping	No
Unit	ms
Value Range	0..65000
Default Value	1000

3.2.2 - DRIVE PARAMETERS

3.2.2.1 - Motor parameters

The motor parameters are stored in object 0x6410
These values are the parameters given in the motor manufacturer's catalogue.

The motor control parameters
- number of pole pairs (0x6410-13),
- motor phase (0x6410-14),
- motor offset (0x6410-15)

will be respectively copied in objects 0x3410-1, 0x3410-2 and 0x3410-3.

Object 0x3410 can be possibly modified and will be used for the motor control (i.e. if the resolver wiring or adjustment is not correct).

The auto-phasing procedure will calculate these parameters of object 0x3410.

The motor inductance parameter of the catalogue (0x6410-14) will be copied in object 0x340F-0 and will be used for calculating the current loop gains (0x60F6).

Object 0x340F-0 can be possibly modified before calculating the gains if inductances are serially mounted with the motor.

The Maximum Motor Speed (0x6410-7) parameter of the catalogue will clip the motor speed peaks in 0x6080.

Index	0x6410
Name	Motor Data
Object Code	RECORD
Object Class	all
Number of Elements	19

This object defines the motor manufacturer's motor data.

Value Description

Sub Index	1
Description	Motor Manufacturer Name
Data Type	String
Access	rw
PDO Mapping	No
Value	Maximum 30 characters

Sub Index	2
Description	Motor Model Name
Data Type	String
Access	rw
PDO Mapping	No
Value	Maximum 30 characters

Sub Index	3
Description	Motor Code Special code or personalisation code.
Data Type	String
Access	rw
PDO Mapping	No
Value	Maximum 30 characters

Sub Index	4
Description	Catalog Date Code
Data Type	Unsigned16
Access	rw
Object Class	all
PDO Mapping	No

The structure of the entries is the following:

MSB		LSB
Year (7-bit)	Month (4-bit)	Date (5-bit)

Year is relative to 1984.

Sub Index	5
Description	Modification Date Code
Data Type	Unsigned16
Access	rw
PDO Mapping	No

Sub Index	6
Description	Motor Type
Data Type	Unsigned16
Access	rw
PDO Mapping	No
Value	0 Rotative 1 Linear

Sub Index	7
Description	Motor Max Speed
Data Type	Unsigned32
Access	rw
PDO Mapping	No
Unit	rpm

Sub Index	8
Description	Motor Rated Speed
Data Type	Unsigned32
Access	rw
PDO Mapping	No
Unit	rpm

Sub Index	9
Description	Motor Stall Current
Data Type	Unsigned32
Access	rw
PDO Mapping	No
Unit	mA

Sub Index	10
Description	Motor Peak Current
Data Type	Unsigned32
Access	rw
PDO Mapping	No
Unit	mA

Sub Index	11
Description	Torque Constant (Kt)
Data Type	Unsigned16
Access	rw
PDO Mapping	No
Unit	0.001Nm/A

Sub Index	12
Description	Inertia
Data Type	Unsigned16
Access	rw
PDO Mapping	No
Unit	0.001gm²

Sub Index	13
Description	Inductance
Data Type	Unsigned16
Access	rw
PDO Mapping	No
Unit	0.1mH

Sub Index	14
Description	Number of motor pole pairs
Data Type	Unsigned16
Access	rw
PDO Mapping	No
Value	1..24

Sub Index	15
Description	Motor Phase
Data Type	Unsigned16
Access	rw
PDO Mapping	No
Value	0x5555 or 0xAAAA (corresponding to 240° or 120°)

Sub Index	16
Description	Motor Phasing
Data Type	Unsigned16
Access	rw
PDO Mapping	No
Value	

Index	**0x3410**
Name	Motor Control Parameters
Object Code	ARRAY
Object Class	all
Number of Elements	3

This object defines the parameters which control the motor.

Value Description

Sub Index	1
Description	Number of motor pole pairs
Data Type	Unsigned16
Access	rw
PDO Mapping	No
Value	1..24

Sub Index	2
Description	Motor Phase
Data Type	Unsigned16
Access	rw
PDO Mapping	No
Value	0x5555 (240°) 0xAAAA (120°)

Sub Index	3
Description	Motor Offset
Data Type	Unsigned16
Access	rw
PDO Mapping	No
Value	

Auto-phasing procedure

Index	0x3413
Name	Start Auto-phasing procedure
Object Code	
Data Type	Unsigned32
Object Class	all
Access	rw
PDO Mapping	No

In order to avoid running the auto-phasing procedure by mistake, the auto-phasing is only executed when a specific signature is written to this sub-index. The signature is 'apha'.
Signature = 0x61687061

Writing 0 to this object when auto- phasing is running will abort the procedure.

When reading, this object returns the operation status:

Read Value	Meaning
0	Procedure never executed
1	Can not execute
2	Procedure running
3	Procedure aborted by user
4	Procedure stopped on error
>= 5	Procedure done

When running, the BUSY bit of status word (0x6041) is set.

The auto-phasing procedure calculates these parameters:
number of pole pairs 0x3410,1
motor phase 0x3410,2
motor offset 0x3410,3

Motor-phasing procedure

Index	0x3414
Name	Start Motor-phasing procedure
Object Code	
Data Type	Unsigned32
Object Class	all
Access	rw
PDO Mapping	No

In order to avoid running the motor-phasing procedure by mistake, the motor -phasing is only executed when a specific signature is written to this sub-index. The signature is 'mcal'.
Signature = 0x6C61636D

Writing 0 to this object when motor-phasing is running will abort the procedure.

When reading, this object returns the operation status:

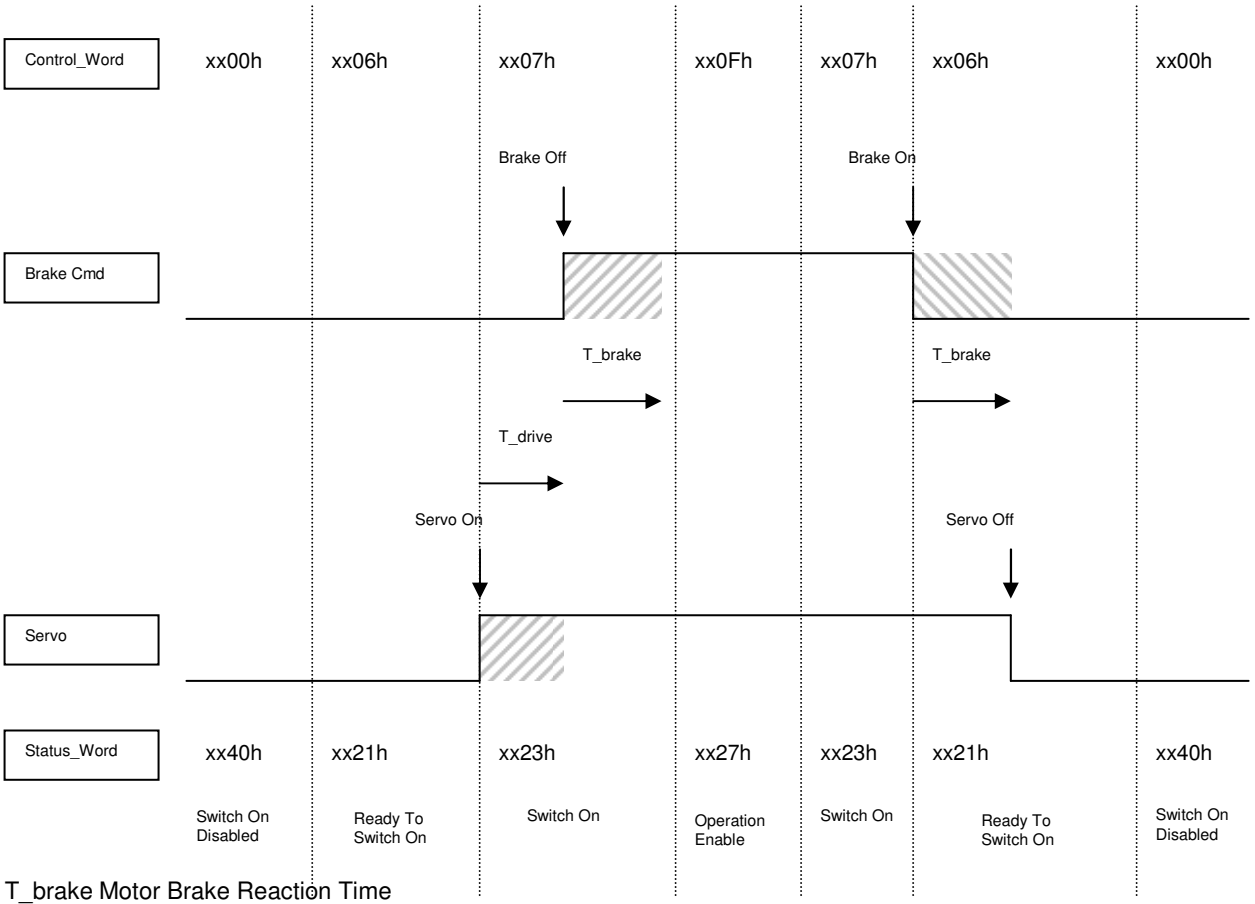
Read Value	Meaning
0	Procedure never executed
1	Can not execute
2	Procedure running
3	Procedure aborted by user
4	Procedure stopped on error
>= 5	Procedure done

When running, the BUSY bit of status word (0x6041) is set.

The motor-phasing procedure calculates these parameters: motor offset 0x3410,3

3.2.2.2 - Motor Brake

Servo On/Off Timing Diagram



T_brake Motor Brake Reaction Time
T_drive Drive Reaction Time

Index	Object	Name	Type	Attr.
0x3304	VAR	Amplifier Reaction Time	Unsigned16	rw
0x3305	VAR	Motor Brake Reaction Time	Unsigned16	rw

Note: The motor brake control is automatic with Switch On/Off by the control_word. To disable the motor brake control, it is necessary to set at 1 bit 0 of object 60FE sub-index 2 (digital output bitmask). The motor brake is then manually controlled by bit 0 of object 60FE sub-index 1.

Drive Reaction Time

Index	0x3304
Name	Drive Reaction Time
Object Code	VAR
Data Type	Unsigned16
Object Class	all
Access	rw
PDO Mapping	No
Unit	ms
Value Range	0..65535
Default Value	1

This parameter defines the reaction time of the drive when enabled / disabled.

Motor Brake Reaction Time

Index	0x3305
Name	Motor Brake Reaction Time
Object Code	VAR
Data Type	Unsigned16
Object Class	all
Access	rw
PDO Mapping	No
Unit	ms
Value Range	0..65535
Default Value	1

This parameter defines the reaction time of the motor brake.

3.2.2.3 - Motor current limits & Current Loop

The parameters defining the current limitation to be applied to the motor are the following:

- Motor Max Current 0x6073
- Motor Rated Current 0x6075

The motor parameters **Motor Max Current** (0x6410-8) and **Motor stall Current** (0x6410-9) will be used for calculating the internal limitations of the drive according to the drive maximum and rated currents (0x6510). The values of the drive internal limitations can be displayed by object 0x30F4.

The current loop gains are accessible in object 0x60F6.

Object 0x3411 allows:

- to calculate the current loop gains according to the motor parameters and the drive specifications:
  - Parameters:
    - Inductance (0x340F)
    - Drive Max current (0x6510-1)
  - Results:
    - Current Loop Gains (0x60F6)

Object 0x3412 allows:

- to calculate the drive current limitations according to the motor and drive currents (0x6510):
  - Parameters:
    - Motor Peak current (0x6410-10)
    - Motor Stall current (0x6410-9)
    - Drive Max current (0x6510-1)
    - Drive Rated current (0x6510-2)
  - Results:
    - Motor Max current (0x6073-0)
    - Motor Rated current (0x6075-0)

The input parameters must be previously defined.

Manufacturer Drive Data

Index	0x6510
Name	Manufacturer Drive Data
Object Code	ARRAY
Number of Elements	5

This object indicates the peak current and the rated current supported by the power module.

Value Description

Sub Index	1
Description	Drive Max Current gives the drive rating
Data Type	Unsigned32
Access	ro
PDO Mapping	No
Unit	mA

Sub Index	2
Description	Drive Rated Current gives the drive rated current
Data Type	Unsigned32
Access	ro
PDO Mapping	No
Unit	mA

Sub Index	3
Description	Drive Voltage gives the drive voltage
Data Type	Unsigned16
Access	ro
PDO Mapping	No
Unit	V

Sub Index	4
Description	Drive Service Voltage Defines the drive operating voltage
Data Type	Unsigned16
Access	rw
Backup	drive's parameter file
PDO Mapping	No
Unit	V
Value	Must be less than or equal to Drive Voltage (0x6510-3)

Sub Index	5
Description	Power Supply Voltage Threshold Defines the undervoltage error level.
Data Type	Unsigned16
Access	rw
Backup	drive's parameter file
PDO Mapping	No
Unit	V
min	20
max	100
Default value	100

Index	**0x3411**
Name	Current Loop Calculation
Object Code	VAR
Data Type	Unsigned32
Object Class	all
Access	rw
PDO Mapping	No
Default Value	0

When the motor inductance (0x6410) and drive current (0x6510) are correct, this object allows to calculate the current loop parameters.

In order to avoid this operation by mistake, the user must write a specific signature to this object to make the calculation.
The signature is 'calc'.

Signature = 0x636C6163

The parameters calculated are in object 0x60F6.

This procedure calculates also the current limit values (0x6073 and 0x6075)

Index	**0x3412**
Name	Current Limitation Calculation
Object Code	VAR
Data Type	Unsigned32
Object Class	all
Access	rw
PDO Mapping	No
Default Value	0

Signature = 0x636C6163

This procedure calculates the current limit values (0x6073 and 0x6075)

Index	**0x6073**
Name	Motor Max current
Object Code	VAR
Data Type	Integer16
Object Class	all
Access	rw
PDO Mapping	No
Unit	per thousand of rated current (0x6075)
Value Range	
Default Value	

This object defines the maximum current that the amplifier can deliver to the motor.

Index	**0x6075**
Name	Motor Rated Current
Object Code	VAR
Data Type	Integer32
Object Class	all
Access	rw
PDO Mapping	No
Unit	mA
Value Range	
Default Value	

This object defines the rated current that the amplifier can deliver to the motor.

Current Loop Parameters

This object defines the parameters of the current loops.

Index	**0x60F6**
Name	Current Loop Parameter Set
Object Code	RECORD
Number of Elements	5

Value Description

Sub Index	1
Description	Regulator Type
Data Type	Unsigned16
Object Class	all
Access	rw
PDO Mapping	No
Value Range	0.. 65535
Default Value	0

Sub Index	2
Description	q-Loop Proportional Gain
Data Type	Unsigned16
Object Class	all
Access	rw
PDO Mapping	No
Value Range	0.. 65535
Default Value	

Sub Index	3
Description	q-Loop Integral Gain
Data Type	Unsigned16
Object Class	all
Access	rw
PDO Mapping	No
Value Range	0.. 65535
Default Value	

Sub Index	4
Description	d-Loop Proportional Gain
Data Type	Unsigned16
Object Class	all
Access	rw
PDO Mapping	No
Value Range	0..65535
Default Value	

Sub Index	5
Description	d-Loop Integral Gain
Data Type	Unsigned16
Object Class	all
Access	rw
PDO Mapping	No
Value Range	0.. 65535
Default Value	

Torque Offset: This object allows to introduce an offset in the torque command.

Index	0x60B2
Name	Torque Offset
Object Code	VAR
Data Type	Integer16
Object Class	all
Access	rw
PDO Mapping	Yes
Unit	per thousand of rated current (0x6075)
Default Value	0

The "Current Actual Value" gives the value of the DC current in the drive. This signal is filtered by a low-pass filter (0x3078)

Index	0x6078
Name	Current Actual Value
Object Code	VAR
Data Type	Integer16
Object Class	all
Access	ro
PDO Mapping	Yes
Unit	per thousand of motor rated current (0x6075)
Value Range	-
Default Value	-

Low-pass filter on “Current Actual Value” (0x6078)

Index	0x3078
Name	Current measurement filter
Object Code	VAR
Data Type	Integer16
Object Class	all
Access	rw
PDO Mapping	No
Unit	Hz
Defaut Value	1000

3.2.2.4 - Dynamic current limits

The current applied to the motor is dynamically limited by the value of a defined object. By default, object 0x30D1 is used to limit the motor current (defined in 0x30DA).

The default value of object 0x30D1 is 0x3FFF and corresponds to the maximum current setting by the user (0x6073).

Dynamic Current Limit Input Source

Index	0x30DA
Name	Dynamic Current Limit Input Source
Description	Index/sub-index of input data
Data Type	Unsigned32
Object Class	all
Access	rw
PDO Mapping	No
Default Value	0x30D10000
Value	See below

This object allows to connect any dataflow as the source of the Dynamic Current Limit.

By default the object 0x30D1 is used as Dynamic Current Limit signal.

The structure of the entries is the following:

MSB		LSB
Index (16-bit)	Sub-index (8-bit)	0

Current Limit

Index	**0x30D1**
Name	Current Limit
Description	This object allows to limit the current dynamically applied to the motor. Changes on this object will be continuously effective.
Data Type	integer16
Object Class	all
Access	rw
PDO Mapping	Yes
Default Value	0x3FFF
Value	0-0x3FFF 0x3FFF corresponds to the maximum value setting (0x6073) for maximum current in the motor

Dynamic Current Limit Configuration

Index	**0x30D2**
Name	Dynamic Current Limit Configuration
Description	This object allows to define the effect of Dynamic Current Limit signal.
Data Type	Unsigned16
Object Class	all
Access	rw
PDO Mapping	No
Default Value	0
Value	bit description 0 0 normal effect of the Dynamic Current Limit signal: 0 current is limited to 0 0x3FFF corresponds to the maximum current (0x6073) 1 reverse effect of the Dynamic Current Limit signal 0x3FFF current is limited to 0 0 corresponds to the maximum current (0x6073) 1..15 reserved

Current Monitor

Index	0x30D4
Name	Current monitor
Object Code	VAR
Data Type	Integer16
Object Class	all
Access	ro
PDO Mapping	Yes
Unit	% of drive max current (0x6510) (0x3FFF = 100% Imax)
Value Range	-
Default Value	-

3.2.2.5 - Motor temperature probe

Index	0x3324
Name	Motor temperature probe configuration
Object Code	RECORD
Object Class	all
Number of Elements	3

This object defines Motor temperature probe configuration.

Value Description

Sub Index	1
Description	Motor temperature type
Data Type	Integer16
Access	rw
PDO Mapping	No
Value	-1 NTC probe 1 PTC probe 0 No probe

Sub Index	2
Description	Motor temperature warning threshold
Data Type	Unsigned32
Access	rw
PDO Mapping	No
Unit	Ω (ohm)
Default value	2400

This parameter defines the threshold of the equivalent resistor corresponding to the temperature at which a warning will be notified.

Sub Index	3
Description	Motor temperature error threshold
Data Type	Unsigned32
Access	rw
PDO Mapping	No
Unit	Ω (ohm)
Default value	2400

This parameter defines the threshold of the equivalent resistor corresponding to the temperature at which an error will be triggered.

Index	0x3323
Name	Motor temperature probe monitoring
Object Code	VAR
Data Type	Unsigned32
Object Class	all
Access	ro
Unit	Ω (ohm)
PDO Mapping	No

The returned value gives an image of the equivalent resistance (in Ω).

3.2.2.6 - Sensors

The **ServoPac** servo drive has 2 sensor inputs: Resolver and Encoder

Each sensor can be used as motor feedback or position feedback.

Index	Object	Name	Type	Attr.
0x306A	VAR	Position Feedback Sensor Select	Unsigned16	rw
0x3070	VAR	Motor Feedback Sensor Select	Unsigned16	rw

Position Feedback Sensor Select

Index	0x306A
Name	Position Feedback Sensor Select
Object Code	VAR
Data Type	Unsigned16
Object Class	all
Access	rw
PDO Mapping	No
Default Value	0

This object defines the feedback sensor which will be used to close the position loop.

Value	Function
0	Resolver Feedback
1	Encoder Feedback

When motor feedback and position feedback are different (resolver for motor feedback and encoder for position feedback, for example), both sensors must count in the same direction.

Motor Feedback Sensor Select

Index	0x3070
Name	Motor Feedback Sensor Select
Object Code	VAR
Data Type	Unsigned16
Object Class	all
Access	rw
PDO Mapping	No
Default Value	0

The motor feedback sensor is used to close the servo motor torque and speed control loops. The servo motor position loop can be closed by the motor feedback sensor or with the secondary sensor (see object 0x306A).

Value	Function
0	Resolver Feedback
1	Encoder Feedback

3.2.2.6.1 - Resolver

Resolver Parameters

Index	Sub	Name	Description	Type	Attribute
0x3100		Resolver	Resolver monitoring		
	1	Res_Sin		Integer16	ro
	2	Res_Cos		Integer16	ro
	3	Res_Amp2		Unsigned16	ro
	4	Res_Mod	Resolver value for one motor revolution. (absolute single-turn) one revolution -> 16-bit	Unsigned16	ro
	5	Res_Amp		Unsigned16	ro
0x3101		Res_Setp	Resolver Setup		
	1	Res_Type			rw
	2	Res_Cfg			rw
	3	Res_Zsh			rw
	4	Res_Zsz			rw
	5	Res_NP			rw
0x3102		Res_Err	Resolver Error control		
	1	Res_Thrs		Unsigned16	rw
	2	Res_Lim		Unsigned16	rw
	3	Res_AmpF		Unsigned16	rw
	4	Res_Rdc		Unsigned32	rw
	5	Res_Filt		Unsigned16	rw
0x3104		Res_Cal	Resolver Calibration procedure		
0x3105		Res_CalV	Resolver Calibration parameters		
0x3107	0	Res_TopZ	Resolver Virtual Top Z	Unsigned16	ro
0x3108	0	Res_ofs	Resolver Offset (user position unit)	Integer32	rw
0x3109	0	Res_pos	Resolver Position (user position unit)	Integer32	ro
0x310A	0	Res_vel	Resolver Velocity (user velocity unit)	Integer32	ro
0x310C	0	Res_raw	Resolver raw position	Integer32	ro

Resolver Setup

Index	0x3101
Name	Resolver Setup
Object Code	RECORD
Number of Elements	6

Value Description

Sub Index	1
Description	Resolver Type
Data Type	Unsigned16
Object Class	all
Access	rw
PDO Mapping	No
Default Value	0

Bit Number	Description
0	1 Enabled 0 Disabled
1, 2	reserved
3	1 SinCos Track
4, 5	reserved
6	1 Absolute Single-turn
7..15	reserved

For a resolver, the setting value is 0x41
For a SinCos track encoder, the setting is 0x49

Sub Index	2
Description	Resolver Configuration
Data Type	Unsigned16
Object Class	all
Access	rw
PDO Mapping	No
Default Value	0

Bit Number	**Description**
0	0 Normal direction 1 Reverse direction

Sub Index	3
Description	Resolver Virtual Top Z shift
Data Type	Unsigned16
Object Class	all
Access	rw
PDO Mapping	No
Default Value	0

This parameter defines the offset between marker Z of the encoder and the virtual marker Z.
The value is given in encoder increments (4096 increments / revolution).

Sub Index	4
Description	Resolver Virtual Top Z size
Data Type	Unsigned16
Object Class	all
Access	rw
PDO Mapping	No
Default Value	0

This parameter defines the width of the virtual marker Z.
The value is given in encoder increments (4096 increments / revolution).

The virtual marker Z is working with polling technique, the width of the virtual marker Z allows to increase the marker Z size in order to avoid a missing of the marker Z.

The status of the virtual marker Z can be read by object 0x3027

Sub Index	5
Description	Resolver Pole pairs reserved for future used
Data Type	Unsigned16
Object Class	all
Access	rw
PDO Mapping	No
Default Value	1

Resolver Position Offset

Index	0x3108
Name	Resolver Position Offset
Object Code	VAR
Data Type	Integer32
Object Class	all
Access	rw
PDO Mapping	Yes
Unit	User Position Unit
Value Range	$(-2^{31})..(2^{31}-1)$
Default Value	0

See Resolver Position (0x3109).

Resolver Position

Index	0x3109
Name	Resolver Position
Object Code	VAR
Data Type	Integer32
Object Class	all
Access	ro
PDO Mapping	Yes
Unit	User Position Unit
Value Range	$(-2^{31})..(2^{31}-1)$
Default Value	-

This object monitors the resolver position:

Resolver_Position = Resolver_Internal_Position + Resolver_Position_Offset

Resolver_Position (0x3109) in user position unit is the position given by the resolver. If the position loop feedback is resolver, and the modulo function (Position Limit) is not activated, then the resolver position is the same as 0x6064.

Resolver_Internal_Position in user position unit is the resolver position value related to the initial position at power on.

Resolver_Position_Offset (0x3108) defines an offset between user position (0x3109) and internal resolver position. If the position loop feedback is resolver, this offset will be calculated by the homing procedure. At power on Resolver_Position_Offset is 0.

3.2.2.6.2 - Encoder

Encoder support types:

- TTL Incremental Encoder
- TTL Incremental Encoder + Hall Effect Sensor
- Sin-Cos Incremental Encoder
- Sin-Cos Incremental Encoder + Hall Effect Sensor
- Hiperface Encoder

Encoder Parameters

Index	Sub	Name	Description	Type	Attribute
0x3120		Encoder1	Encoder 1		
	1	Enc1Sin		Integer16	ro
	2	Enc1Cos		Integer16	ro
	3	Enc1Amp2		Integer16	ro
	4	Enc1Mod	Encoder value for one motor revolution. one revolution -> 16-bit	Unsigned16	ro
	5	Enc1Amp		Integer16	ro
0x3121		Enc1Setp	Encoder 1 Setup		
0x3122		Enc1Err	Encoder 1 Error Control		
	1	Enc1Cnt		Unsigned32	rw
	2	Enc1Thrs		Unsigned16	rw
	3	Enc1Lim		Unsigned16	rw
	4	Enc1Zlim		Unsigned16	rw
	5	Enc1Clim		Unsigned16	rw
	6	Enc1Vlim		Unsigned32	rw
0x3124		Enc1CalP	Encoder 1 Calibration		
0x3127	0	Enc1TopZ	Encoder 1 Virtual Top Z	Unsigned16	ro
0x3128	0	Enc1ofs	Encoder 1 Offset (user position unit)	Integer32	rw
0x3129	0	Enc1pos	Encoder 1 Position (user position unit)	Integer32	ro
0x312A	0	Enc1vel	Encoder 1 Velocity (user velocity unit)	Integer32	ro
0x312C	0	Enc1raw	Encoder1 Raw Position	Integer32	ro

Encoder Setup

Index	0x3121
Name	Encoder Setup
Object Code	RECORD
Number of Elements	6

Value Description

Sub Index	1
Description	Encoder Type
Data Type	Unsigned16
Object Class	all
Access	rw
PDO Mapping	No
Default Value	

Bit Number	Description
0	1 Enabled 0
1	1 TTL Encoder 0
2	1 Sin/Cos Encoder 0
3	1 Encoder with CD track 0
4	1 HES 0
5	0 HAL 60° 1 HAL 120°
6	Absolute Single-turn
7	Absolute Multi-turn
8	Reverse Incremental track / Absolute track
12-15	Communication Protocol 1 Hiperface

Sub Index	2
Description	Encoder Configuration
Data Type	Unsigned16
Object Class	all
Access	rw
PDO Mapping	No
Default Value	

Bit Number	Description
0	0 Normal direction 1 Reverse direction

Sub Index	3
Description	Encoder Virtual Top Z shift
Data Type	Unsigned16
Object Class	all
Access	rw
PDO Mapping	No
Default Value	0

This parameter defines the offset between the marker Z of the encoder and the virtual marker Z.
The value is given in encoder increments (encoder resolution x 4)

Sub Index	4
Description	Encoder Virtual Top Z size
Data Type	Unsigned16
Object Class	all
Access	rw
PDO Mapping	No
Default Value	0

This parameter defines the width of the virtual marker Z.
The value is given in encoder increments (encoder resolution x 4).

The virtual marker Z is working with polling technique, the width of the virtual marker Z allows to increase the marker Z size in order to avoid the missing of the marker Z.

The status of the virtual marker Z can be read by object 0x3127

Sub Index	5
Description	Encoder Resolution x 4
Data Type	Unsigned32
Object Class	all
Access	rw
PDO Mapping	No
Default Value	

This parameter defines the resolution (period) of the encoder x 4.

Encoder Position Offset

Index	**0x3128**
Name	Encoder Position Offset
Object Code	VAR
Data Type	Integer32
Object Class	all
Access	rw
PDO Mapping	Yes
Unit	User Position Unit
Value Range	$(-2^{31})..(2^{31}-1)$
Default Value	0

See Encoder Position (0x3129).

Encoder Position

Index	**0x3129**
Name	Encoder Position
Object Code	VAR
Data Type	Integer32
Object Class	all
Access	ro
PDO Mapping	Yes
Unit	User Position Unit
Value Range	$(-2^{31})..(2^{31}-1)$
Default Value	-

This object monitors the encoder position:

Encoder_Position = Encoder_Internal_Position + Encoder_Position_Offset

Encoder _Position (0x3129) in user position unit is the position given by the encoder. If the position loop feedback is encoder and modulo function (Position Limit) is not activated, then the encoder position is the same as 0x6064.

Encoder_Internal_Position in user position unit, is the encoder position value related to the initial position at power on.

Encoder_Position_Offset (0x3128) defines an offset between user position (0x3129) and internal encoder position. If the position loop feedback is encoder, this offset will be calculated by the homing procedure. At power on Encoder_Position_Offset is 0.
If the encoder is absolute multi-turn, the Encoder_Position_Offset is saved in the drive parameter file and is restored at power-on.

3.2.2.6.3 - TTL Encoder

A TTL incremental encoder can be connected to ServoPac drives as motor feedback or only position feedback.

Motor Feedback:
TTL incremental encoder is not absolute for motor commutation, so:
- In a first time, an auto-phasing must be performed to define motor poles pairs number, motor phase order, and encoder offset.
- Each time the drive is restarted with 24V, a motor-phasing must be performed before motor can be controlled.

Note:
- motor-phasing applies torque and moves motor
- power supply must be on
- please check that motor is at standstill and its movement over one turn not dangerous for operator and machine.
- motor-phasing doest not work with vertical axis or axis with driving load.

Position Feedback:
If the encoder is used as position feedback only (motor feedback is resolver) then the encoder resolution defined in object 0x608F must be the encoder counts for one motor revolution.

3.2.2.6.4 - Sin-Cos Encoder

A Sin-Cos incremental encoder can be used with ServoPac drive as an TTL incremental encoder.

An internal Sin-Cos interpolation allows the drive working in higher resolution and so better result for speed loop.

3.2.2.6.5 - Hall Effect Sensor

The hall effect sensor can be used with a TTL incremental encoder or a sin-cos incremental encoder to avoid a motor phase search with motor-phasing operation each time the 24V is applied.

The hall effect sensor parameters are calculated with the auto-phasing procedure.

Parameters depending on hall effect sensor:
- Motor phase order: 0x3410,2
- Sensor offset: 0x3410,3
- Hall effect sensor parameter: 0x313E,0

Index	Object	Name	Type	Attr.
0x313E	VAR	HES configuration	Unsigned16	rw

Hall Effect Sensor configuration

Index	0x313E
Name	Encoder HES configuration
Description	Encoder Type
Data Type	Unsigned16
Object Class	all
Access	rw
PDO Mapping	No
Saved	Yes
Default Value	0

Value Description

Bit Number	Description
0-2	HES initial state
3	Direction
4	Type: 0 60° 1 120°

Manual Configuration for an incremental encoder + HES:
0x3121,1 = 0x0013 ; incremental TTL encoder + HES
0x313E,0 = HES config
0x3410,1 = pole pairs
0x3410,2 = phase order
0x3410,3 = sensor offset (mechanic)

3.2.2.6.6 - Hiperface

A Hiperface encoder type can be connected to an ServoPac drive.
Only Hiperface Encoder types different from 0xFF can be recognized.

Setup Hiperface encoder with Gem Drive Studio
The Hiperface Encoder commissioning can be made with Gem Drive Studio:
- Select Hiperface Encoder
- Check "Enable encoder input"
- "Read Encoder Configuration" to read encoder parameters
- "Apply"

Move the motor by hand, if there is an "Encoder Commutation channel / Incremental channel Error" then toggle "Reverse Incremental Track".

Setup Hiperface encoder manually
Enable and select Hiperface encoder are defined with object 0x3121,1

Writing a 1 to object 0x312B,1 allows to read Hiperface encoder parameters.

The Hiperface encoder has an absolute information track (serial) and an incremental information track (Sin-Cos). The two information tracks must evolve in the same direction. An inverting of Sin-Cos signals can change the count direction of Sin-Cos signal regarding the absolute value from serial channel.

If there is an "Encoder Commutation channel / Incremental channel Error" when moving the encoder (motor), then "Reverse Incremental track / Absolute track" bit in object 0x3121,1 must be toggled.

3.2.2.6.7 - Absolute Multi-turn Position

With an absolute encoder feedback, the motor absolute position value over one revolution is available and the servo motor can immediately be enabled after the amplifier power up. The servo drive behaviour at the amplifier power up is similar to a resolver sensor feedback. For a position application, an absolute multi-turn encoder allows to avoid the homing sequence after power up. In this case, the absolute position value over the axis travel range is available at power up and the positioning can be immediately started. However, the axis must never go out of encoder's absolute position range.

Setting Encoder Zero
The absolute encoder gives a position value between 0 and a max position value (depends on encoder model).

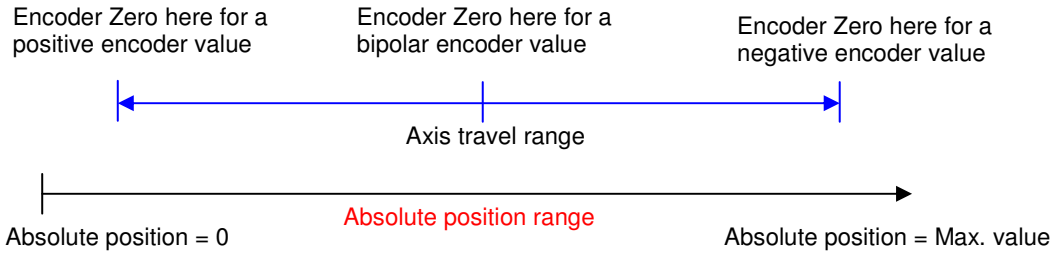
For a a Hiperface encoder, the max position value is given by:
(Number_of_revolutions x Number_of_periods x 4 x 8) - 1
Number_of_revolutions is the maximum revolution for that encoder.
Number_of_periods is the the number of Sin-Cos periods per revolution
4 is quadrature counter multiplier
8 is the interpolation factor.

Example 1:
Number_of_revolutions = 4096
Number_of_periods = 1024
Then the maximum position value given by the encoder is 4096 x 1024 x 4 x 8 - 1 = 134 217 727 = 0x7FF FFFF

Once the encoder parameters are set, this maximum position can be read with object 0x312D,1.
The current position of the encoder is given by object 0x312D,3
Note: these values are in encoder unit.

As the encoder gives only a position between 0 and encoder max value, an Encoder Zero Position can be defined for a convenient operation:

- always positive value,
- or always negative value,
- or bipolar value



The object 0x312B,1 allows to set the Encoder Zero at the current position:

- Place the motor at the desired Encoder Zero Position,
- drive must be disabled,
- write 0x312B,1=0x73626165
- save parameters into drive

the encoder offset is given in object 0x312D,5 (encoder unit) which must be is stored in the parameter file.

The Encoder Zero Position can be defined manually:

- drive is disabled,
- write offset value to 0x312D,5
- save parameters into drive
- restart 24V to apply this offset.

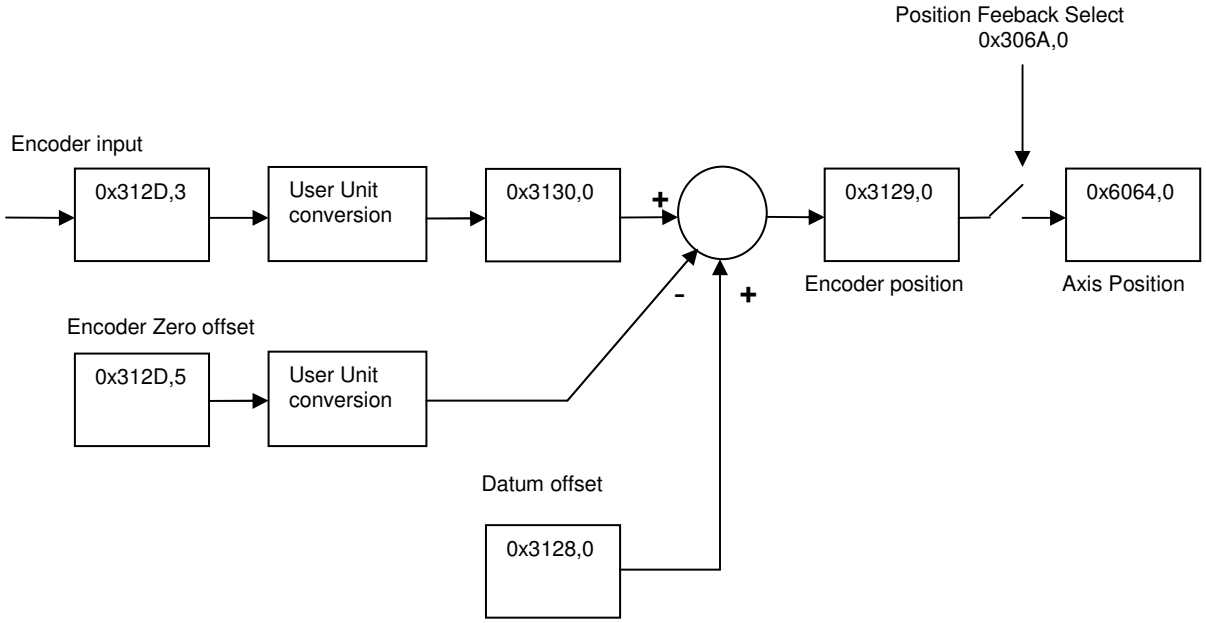
Example 2:
Set 0x312D,5 = 0x400 0000, so Encoder Zero Position is set in middle of absolute position range,
The absolute encoder value will be from -(Max_value+1)/2 to (Max_value+1)/2-1

User Datum
The user position reference related to themechanical machine can be defined with a homing operation as usual.
The encoder position offset from homing operation can be read with object 0x3128,0
After homing operation, parameters need to be saved (object 0x3128,0 is saved in the parameter file).
This operation needs to be made only once in the machine life-time.

The encoder position in user unit is defined by the relation below :
[0x3129,0] = [0x3130,0] + [0x3128,0] - user_unit([0x312D,5])

0x3129,0 and 0x3130,0 and 0x3128,0 are in user unit, 0x312D,5 is in encoder unit.

If the position feedback is encoder, then [0x6064,0] is the same as [0x3129,0].



3.2.2.7 - Factor and units

Factor and Units

The position unit is defined by object 0x6093

The velocity unit is defined by position unit per second.

The acceleration unit is defined by position unit per square second.

Index	Object	Name	Type	Attr.
0x608F	ARRAY	Encoder Position Resolution	Unsigned32	rw
0x6093	ARRAY	Position Factor	Unsigned32	rw
0x3089	VAR	Position Display Factor	Unsigned16	rw
0x308A	VAR	Position Unit Name	String	rw

Index	0x608F
Name	Encoder Position Resolution
Object Code	ARRAY
Number of Elements	2

Value Description

Sub Index	1
Description	Encoder Increments
Data Type	Unsigned32
Object Class	all
Access	rw
PDO Mapping	No
Unit	inc
Value Range	
Default Value	0x1000

This parameter defines the encoder position resolution for one motor revolution.

Sub Index	2
Description	Motor Revolutions
Data Type	Unsigned32
Access	ro
PDO Mapping	No
Default Value	1

Position Factor

Index	**0x6093**
Name	Position Factor
Object Code	ARRAY
Number of Elements	2

Value Description

Sub Index	1
Description	Position Factor Numerator
Data Type	Unsigned32
Object Class	all
Access	rw
PDO Mapping	No
Default Value	4096

Sub Index	2
Description	Position Factor Denominator
Data Type	Unsigned32
Object Class	all
Access	rw
PDO Mapping	No
Default Value	4096

The Denominator defines the increments in user unit for one motor revolution.

The Numerator defines the increments in motor unit for one motor revolution. This value must be set to 4096.

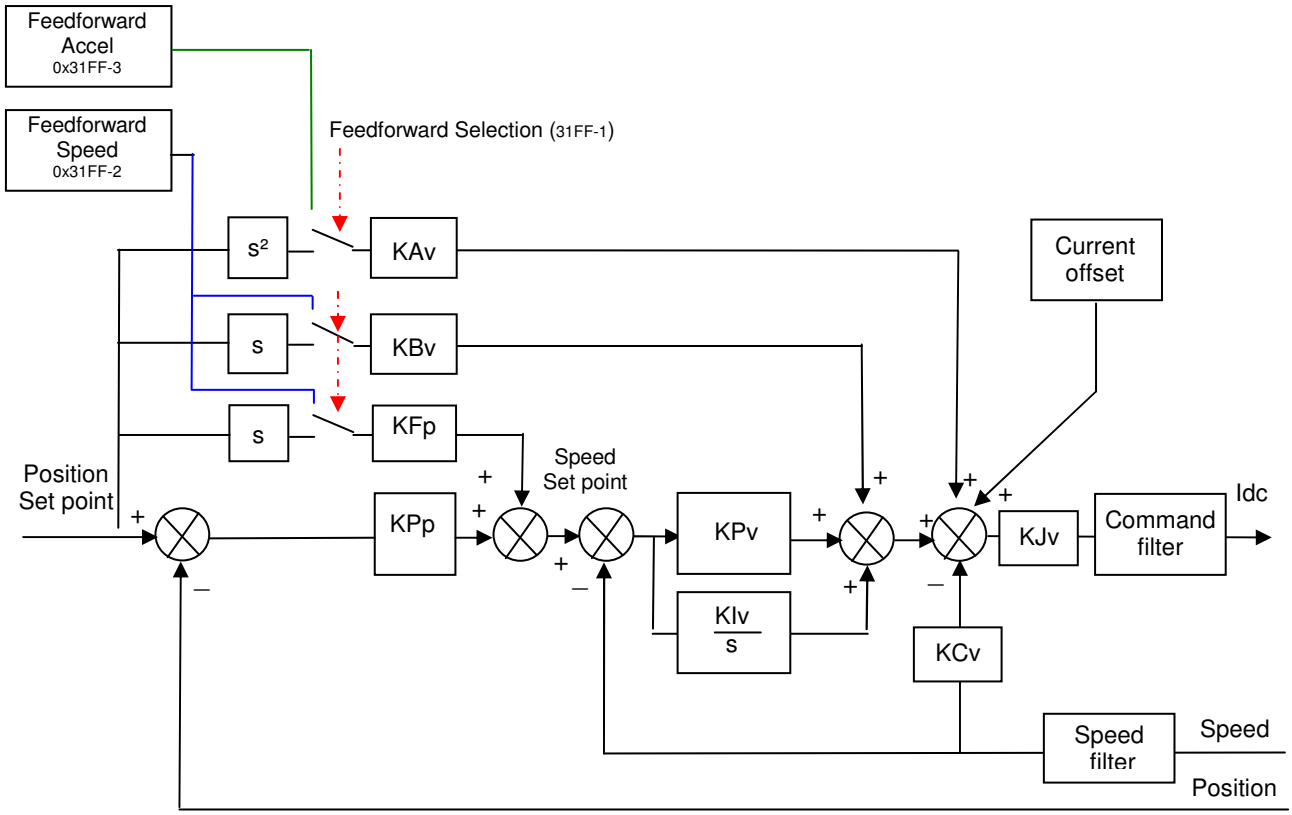
Motor_position = Numerator / Denominator * User_position

Example:

1 motor revolution corresponds to a displacement of 5 mm on the load.
The desired user resolution is µm.
Setting parameters:
Numerator = 4096
Denominator = 5000
User unit = µm

3.2.2.8 - Servo Loops

SERVO CONTROLLER STRUCTURE



Speed loop gains are the most critical adjust because they greatly depend on the mechanical load characteristics (inertias, frictions, coupling stiffness, resonances,...).

- Proportional speed gain (KPv): defines the proportional gain of the controller which acts on the speed error. The higher this parameter value, the faster the speed loop response.

- Integral speed gain (KIv): defines the integral gain of the controller which acts on the speed error. The higher this parameter value, the better the axis stiffness.

- **Integrator low frequency limit (KIvf in Hz)**: defines the low frequency value from where the controller integrator term is saturated. This parameter is used for reducing the motor heating in applications with large dry frictions due to the mechanical load.

- Damping gain (KCv): defines the proportional gain of the controller which acts only on the speed feedback. This parameter allows to reduce the speed loop overshoot in response to a step like set point change .

- Derivative speed gain (KDv): defines the derivative gain of the controller which acts on the speed error.

- **Derivator high frequency limit (KDvf in Hz)**: defines the high frequency value from which the controller derivative term is saturated.

- **Gain scaling factor (KJv)**: defines a multiplying factor for all speed regulator gains. This parameter is scaling the speed regulator gains in order to avoid any saturation when large values are required. This parameter allows also to adjust the servo loop stability in case of load inertia changes.

The **Current command filter** is a 3rd order, low pass type, with 3 adjustable cut-off frequencies. Each cut-off frequency value can be freely adjusted according to the application for the filtering of high frequency noise or the filtering of mechanical resonances.

The **Speed measurement filter** is a 1st order, low pass type, with 3 selectable time constant values. The higher the time constant value, the lower the speed measurement noise, but also the lower the speed loop gains because of the increased speed measurement delay. The **Speed measurement filter** time constant is selected according to the motor position sensor resolution and the acceptable noise level in the speed measurement.

Position loop gains influence mainly the servo motor behaviour during the displacements (following error, position overshoot, audible noise, ...).

- Proportional position gain(KPp): defines the proportional gain of the controller which acts on the position error. The higher this parameter value, the better the axis stiffness and the lower the following error.

- Feedforward speed 1 gain(KFp): defines the feedforward speed amplitude corresponding to the speed input command. This term allows to reduce the following error during the motor displacement. Its value is set to the max (65536) after the autotuning procedure if a following error as small as possible is required.

- Feedforward speed 2 gain(KBv): defines the feedforward speed amplitude corresponding to the viscous frictions. This term allows to reduce the viscous friction effect during the motor displacement. The gain value is equal to the damping gain value + the viscous friction compensation term. After the autotuning procedure, the feedforward speed 2 gain is set equal to the damping gain value if a following error as small as possible is required. The viscous friction compensation term can be calculated by measuring the current/speed ratio at various motor speed values.

- Feedforward acceleration gain(KAv): defines the feedforward acceleration amplitude corresponding to the acceleration input command. This term allows to reduce the following error during the motor acceleration and deceleration phases. Its value is calculated by the amplifier during the auto-tuning procedure if a following error as small as possible is required.

When the **autotuning** procedure is executed, the motor + mechanical load specifications are identified and the appropriate gain values are calculated according to the user selected requirements (controller type, filter type, bandwidth value, ...). All gain values can then be modified manually by the user if required.

The choice of the time interval for speed measurement (speed measurement filter) allows to select the speed measurement resolution value according to the position sensor resolution value:

speed resolution (rpm) = 60000 / position sensor resolution (ppr) / time interval (ms).

The higher the time interval value, the better the resolution, but also the lower the servo loop gains because of the increased speed measurement delay.

The choice of the anti-resonance filter is necessary in case of loud noise in the motor due to the motor/load coupling elasticity.

The choice of the maximum stiffness filter allows to get the maximum stiffness on the motor shaft with regard to the torque disturbances. However, this choice is only possible without any resonance due to the motor/load coupling elasticity.

The choice of the speed loop bandwidth defines the cut-off frequency value of the closed loop frequency response (Low = 50 Hz, Medium = 75 Hz, High = 100 Hz).

The choice "**minimum following error**" allows to get an accurate following of the position reference value during the entire motor displacement. In this case, all feedforward gain values are calculated.

The choice "**minimum position overshoot**" allows to get a motor positioning without any overshoot of the target position. In this case, all the feedforward gain values are set at 0, and the motor position is lagging with regard to the position reference value during the whole motor displacement.

Index	Object	Name	Type	Attr.
0x60FB	RECORD	Position Loop Gain		
0x6062	VAR	Position Demand Value	Integer32	ro
0x60F4	VAR	Following Error Actual Value	Integer32	ro
0x6063	VAR	Actual position*	Integer32	ro
0x6064	VAR	Actual position	Integer32	ro
0x6065	VAR	Following Error Window	Integer32	rw
0x3065	VAR	Following Error Control	Unsigned16	rw
0x31FF	RECORD	External Feedforward		rw
0x60F9	RECORD	Speed Loop Parameters		
0x30F9	ARRAY	Speed Error Low-pass Filter	Unsigned16	rw
0x30FA	VAR	Speed measurement filter	Unsigned16	rw
0x606C	VAR	Actual Velocity	Integer32	ro
0x306C	VAR	Actual Velocity Filter	Unsigned16	rw
0x60F6	RECORD	Current Loop Parameters		
0x60B2	VAR	Current Offset	Integer16	rw
0x6078	VAR	Actual Current	Integer16	ro
0x3078	VAR	Actual Current Filter	Unsigned16	rw

Velocity Control Parameter Set

This object defines the parameters of the current loops.

Index	0x60F9
Name	Velocity Control Parameter Set
Object Code	RECORD
Number of Elements	8

Value Description

Sub Index	1
Description	Regulator Type
Data Type	Unsigned16
Object Class	pp ip hm pv eg
Access	rw
PDO Mapping	No
Value Range	0..65535
Default Value	0

Sub Index	2
Description	Proportional Speed Gain Defines the proportional regulator gain (KPv) that acts upon the speed error.
Data Type	Unsigned16
Object Class	pp ip hm pv eg
Access	rw
PDO Mapping	No
Value Range	0..65535
Default Value	

Sub Index	3
Description	Integral Speed Gain Defines the integral regulator gain (KIv) that acts upon the speed error.
Data Type	Unsigned16
Object Class	pp ip hm pv eg
Access	rw
PDO Mapping	No
Value Range	0.. 65535
Default Value	

Sub Index	4
Description	Integral Gain Filter
Data Type	Unsigned16
Object Class	pp ip hm pv eg
Access	rw
PDO Mapping	No
Unit	0.1 Hz
Default Value	

Sub Index	5
Description	Damping Gain (Kc) This gain is used for getting the maximum servo loop stiffness.
Data Type	Unsigned16
Object Class	pp ip hm pv eg
Access	rw
PDO Mapping	No
Value Range	0.. 65535
Default Value	

Sub Index	6
Description	Derivative Gain (KD)
Data Type	Unsigned16
Object Class	pp ip hm pv eg
Access	rw
PDO Mapping	No
Value Range	0.. 65535
Default Value	

Sub Index	7
Description	Derivative Gain Filter
Data Type	Unsigned16
Object Class	pp ip hm pv eg
Access	rw
PDO Mapping	No
Unit	Hz
Default Value	

Sub Index	8
Description	KJv
Data Type	Unsigned16
Object Class	pp ip hm pv eg
Access	rw
PDO Mapping	No
Value Range	0.. 65535
Default Value	

Speed Error Low-pass Filter

Index	**0x30F9**
Name	Speed Loop Low-pass filter Defines the cut-off frequency at -3 dB (Fev) of the first order filter that acts upon the current control. The value of this parameter is depending on the selected bandwidth.
Object Code	ARRAY
Number of Elements	3

Value Description

Sub Index	1
Description	Speed Loop Low-pass filter 1
Data Type	Unsigned16
Object Class	pp ip hm pv eg
Access	rw
PDO Mapping	No
Unit	Hz
Value Range	20..1000Hz 0 not active
Default Value	

Sub Index	2
Description	Speed Loop Low-pass filter 2
Data Type	Unsigned16
Object Class	pp ip hm pv eg
Access	rw
PDO Mapping	No
Unit	Hz
Value Range	20..1000Hz 0 not active
Default Value	

Sub Index	3
Description	Speed Loop Low-pass filter 3
Data Type	Unsigned16
Object Class	pp ip hm pv eg
Access	rw
PDO Mapping	No
Unit	Hz
Value Range	20..1000Hz 0 not active
Default Value	

Index	**0x30FA**
Name	Velocity measurement filter
Object Code	VAR
Data Type	Unsigned16
Object Class	all
Access	rw
PDO Mapping	No
Value Range	0 0.5ms 1 1ms 2 2ms

Index	0x606C
Name	Velocity Actual Value
Object Code	VAR
Data Type	Integer32
Object Class	all
Access	ro
PDO Mapping	Yes
Unit	User Velocity Unit

The "Velocity Actual Value" gives the value of the actual motor velocity in user unit. This signal is filtered by a low-pass filter defined by 0x306C.
Object 0x3069 give the same Actual Velocity but in rpm unit.
Object 0x30F8-2 gives the actual velocity without the low-pass filter.

Index	0x306C
Name	Actual Velocity Filter
Object Code	VAR
Data Type	Unsigned16
Object Class	all
Access	rw
PDO Mapping	No
Unit	Hz
Value Range	
Default Value	800

The filter is applied on “Velocity Actual Value” (0x606C)
Actual Velocity without this filtering : “Velocity Feedback” (0x30F8)

Position Control Parameter Set

Index	0x60FB
Name	Position Control Parameter Set
Object Code	RECORD
Number of Elements	5

Value Description

Sub Index	1
Description	Regulator Type
Data Type	Unsigned16
Object Class	pp ip hm eg
Access	rw
PDO Mapping	No
Default Value	0

Sub Index	2
Description	Proportional Position Gain Defines the proportional gain that acts upon the position error (KP1).
Data Type	Unsigned16
Object Class	pp ip hm eg
Access	rw
PDO Mapping	No
Value Range	0..65535
Default Value	

Sub Index	3
Description	Feedforward Speed 1 Gain Defines the feedforward term amplitude (KF1) corresponding to the speed input command (derivation of the position input command). This feedforward term allows to reduce the following error during the motor acceleration and deceleration phases.
Data Type	Unsigned16
Object Class	pp ip hm eg
Access	rw
PDO Mapping	No
Value Range	0.. 65535
Default Value	

Sub Index	4
Description	Feedforward Acceleration Gain Defines the feedforward acceleration corresponding to the acceleration input command (second derivation of the position input command).This feedforward term allows to reduce the following error during the motor acceleration and deceleration phases.
Data Type	Unsigned16
Object Class	pp ip hm eg
Access	rw
PDO Mapping	No
Value Range	0.. 65535
Default Value	

Sub Index	5
Description	Feedforward Speed 2 Gain This gain value is equal to the damping speed gain value +Feedforward friction gain value. The feedforward friction gain allows to cancel the load viscous friction effect (load viscous friction torque is proportional to axis speed).This feedforward term allows to reduce the following error during the motor acceleration and deceleration phases.
Data Type	Unsigned16
Object Class	pp ip hm eg
Access	rw
PDO Mapping	No
Value Range	0.. 65535
Default Value	

Index	**0x6062**
Name	Position Demand Value
Object Code	VAR
Data Type	Integer32
Object Class	all
Access	ro
PDO Mapping	Yes
Unit	position unit
Default Value	-

This object gives the internal position value in entry of position loop.

Index	0x60F4
Name	Following Error Actual Value
Object Code	VAR
Data Type	Integer32
Object Class	pp ip hm sq sm se gb cm
Access	rw
PDO Mapping	Yes
Unit	position unit
Default Value	-

This object gives the difference between position demand value and position actual value:
FollowingErrorActualValue =PosDemand - PosActual

Index	0x6064
Name	Actual Position
Object Code	VAR
Data Type	Integer32
Object Class	all
Access	ro
PDO Mapping	Yes
Unit	position unit
Default Value	-

This object gives the axis actual position. If the position sensor is a resolver, then the value is the resolver position (0x3109,0). If the position sensor is an encoder, then the value is the encoder position (0x3129,0).
The sensor position is defined by object 0x306A,0.

Index	0x6065
Name	Following Error Window
Object Code	VAR
Data Type	Unsigned32
Object Class	pp ip hm sq sm se gb cm
Access	rw
PDO Mapping	Yes
Unit	position unit
Value Range	0..0xFFFFFFFF
Default Value	-

This object defines the tolerance for position value:
| PosDemand - PosActual | < FollowingErrorWindow

If the value of the following error window is FFFF FFFFh, the following control shall be switched off.

Index	0x3065
Name	Following Error Control This object defines the position error detection mode
Object Code	VAR
Data Type	Unsigned16
Object Class	pp ip hm sq sm se gb cm
Access	rw
PDO Mapping	No
Value Range	Position error detection mode: 0 Absolute value 1 Relative To dynamic model
Default Value	0

External Feedforward

Index	**0x31FF**
Name	External Feedforward
Object Code	RECORD
Object Class	pp, ip, hm sq, se, gb
Number of Elements	3

Value Description

Sub Index	1
Description	External Feedforward Selection
Data Type	Unsigned16
Access	rw
PDO Mapping	Yes
Default Value	0

Bit Number	**Function**
0	reserved
1	Select Feedforward Speed 0 Internal feedforward speed 1 External feedforward speed
2	Select Feedforward Acceleration 0 Internal feedforward acceleration 1 External feedforward acceleration
3..15	reserved

Sub Index	2
Description	External Feedforward Speed
Data Type	Integer32
Access	rw
PDO Mapping	Yes
Unit	Velocity Unit
Default Value	0

Sub Index	3
Description	External Feedforward Acceleration
Data Type	Integer32
Access	rw
PDO Mapping	Yes
Unit	Position Unit
PDO Mapping	No
Default Value	0

3.2.2.9 - Autotuning

Auto-tuning Parameters

Index	0x3425
Name	Auto-tuning parameters
Object Code	ARRAY
Number of Elements	4

Value Description

All these parameters must be set before starting the auto-tuning by 0x3426.

Sub Index	1
Description	Auto-tuning Bandwidth
Data Type	Unsigned16
Object Class	-
Access	rw
PDO Mapping	No
Value Range	0..2
Default Value	

This parameter defines the auto-tuning bandwidth:

Value	Bandwidth
0	Low Bandwidth
1	Medium Bandwidth
2	High Bandwidth

Sub Index	2
Description	Filter type
Data Type	Unsigned16
Object Class	-
Access	rw
PDO Mapping	No
Value Range	0..2
Default Value	

This parameter defines the auto-tuning filter:

Value	Filter
0	Standard filter
1	Anti-resonance filter
2	High stiffness filter

Sub Index	3
Description	Speed Filter
Data Type	Unsigned16
Object Class	-
Access	rw
PDO Mapping	No
Value Range	0..2
Default Value	

This parameter defines the speed filter:

Value	Filter
0	auto-select by auto-tuning
1	0.5ms
2	1ms
3	2ms

Sub Index	4
Description	Auto-tuning Application Requirements
Data Type	Unsigned16
Object Class	-
Access	rw
PDO Mapping	No
Value Range	0..1
Default Value	

Value	Application Requirements
0	Minimum tracking error
1	Minimum overshoot

Auto-tuning Procedure

Index	0x3426
Name	Start Auto-tuning procedure
Object Code	
Data Type	Unsigned32
Object Class	all
Access	rw
PDO Mapping	No

Parameters for Autotuning (0x3425) must be previously set.

In order to avoid running the auto-tuning procedure by mistake, the auto-tuning is only executed when a specific signature is written to this sub-index. The signature is 'atun'.
Signature = 0x6E757461

Writing 0 to this object when auto-tuning is running will abort the procedure.

When reading, this object returns the operation status:

Read Value	Meaning
0	Procedure never executed
1	Can not execute
2	Procedure running
3	Procedure aborted by user
4	Procedure stopped on error
>= 5	Procedure done

When running, the BUSY bit of status word (0x6041) is set.

Remark:
The parameters calculated by the auto-tuning depend on which mode it is executed (for example, if auto-tuning is executed in Profile Velocity Mode, the position loop gain will be equal to 0).

3.2.2.10 - Save / Load parameters

Internal Load/Save Command

The ServoPac drives can store parameters in its internal flash memory:

Writing to object 0x1010 initiates the saving procedure which stores the drive parameters in its internal flash memory (inside a file called DRIVEPAR.TXT).

Writing to object 0x1011 initiates the restoring procedure which re-loads the drive parameters from its internal flash memory (from the previously saved DRIVEPAR.TXT file).

Store parameters

Index	0x1010
Name	Store parameters
Object Code	RECORD
Number of Elements	

This command saves the drive parameters in a volatile memory (ram), in a file in an internal flash memory.

Value Description

Sub Index	1
Description	Save all parameter
Data Type	Unsigned32
Access	rw
PDO Mapping	No
Value	writing signature: 0x65766173 save drive parameters

Signature for difference operation:

Operation	Signature	Ascii
Saving of the manufacturer's parameters	0x6E616D73	"sman"
Saving of the calibration saves drive calibration parameters into flash memory.	0x6C616373	"scal"
Saving of the drive parameters saves drive parameters in memory into flash memory (DRIVEPAR.TXT file).	0x65766173	"save"
Saving of the sequence saves sequences from sequence memory into flash memory (SEQUENCE.TXT file).	0x71657373	"sseq"

While operation is running, busy bit in status word (0x6041) is set.

If the Hiperface encoder is selected, when saving drive parameters, the encoder reference (0x312D,5 and 0x3125,6) and homing offset (0x3128,0) are also stored in the Hiperface encoder non volatile memory.

Restore parameters

Index	0x1011
Name	Restore parameters
Object Code	RECORD
Number of Elements	

Value Description

Sub Index	1
Description	Load all parameters
Data Type	Unsigned32
Access	rw
PDO Mapping	No
Value	writing signature: 0x64616F6C load drive parameters

Signature for difference operation:

Operation	Signature	Ascii
Loading of the manufacturer's parameters	0x6E616D6C	"lman"
Loading of the calibration's parameters	0x6C61636C	"lcal"
Loading of the drive parameters (DRIVEPAR.TXT)	0x64616F6C	"load"
Loading of the USER_PAR.TXT file loads parameters from USER_PAR.TXT file into memory.	0x7273756C	"lusr"
Loading of the SEQUENCE.TXT file loads parameters from SEQUENCE.TXT file into sequence memory	0x7165736C	"lseq"
Merging of the SEQUENCE.TXT file merges parameters from SEQUENCE.TXT file into sequence memory	0x7165736D	"mseq"

While operation is running, busy bit in status word (0x6041) is set.

If the Hiperface encoder is selected, when loading drive parameters, the encoder reference (0x312D,5 and 0x3125,6) and homing offset (0x3128,0) are also loaded from the Hiperface Encoder non volatile memory. After a reset Hiperface error, these objects are also reloaded.

3.2.3 OPERATION MODES

3.2.3.1 - Supported Drive Modes

Supported Drive Modes

A drive can support more than one and several distinct modes of operation. This object gives an overview of the implemented operating modes in the device. This object is read only.

Index	0x6502
Name	Supported drive modes
Object Code	VAR
Data Type	Unsigned32
Object Class	all
Access	ro
PDO Mapping	No
Value	See below

Data Description

Bit Number	Function	Class	OpCode	Servo Loops	Supported
0	Profile Position Mode	pp	1	position, speed and current loops	x
1	Velocity Mode	vm	2		
2	Profile Velocity Mode	pv	3	speed and current loops	x
3	Profile Torque Mode	pt	4	current loop	x
4	reserved				
5	Homing Mode	hm	6	position, speed and current loops	x
6	Interpolated Position Mode	ip	7	position, speed and current loops	x
7	reserved	cst	8		
8	reserved	csv	9		
9	reserved	csp	10		
7..15	reserved				
16	Analog Speed Mode	as	-1	speed and current loops	x
17	Stepper Emulation Mode	se	-2	position, speed and current loops	x
18	Sequence Mode	sq	-3	position, speed and current loops	x
19	reserved	sm	-4		
20	Analog Torque Mode	at	-5	current loop	x
21	Master-Slave Gearbox Mode	gb	-6	position, speed and current loops	x
22	Master-Slave Cam Mode	cm	-7	position, speed and current loops	x

3.2.3.2 - Mode selection

Index	0x6060
Name	Mode of Operation
Object Code	VAR
Data Type	integer8
Object Class	all
Access	rw
Save	Yes
PDO Mapping	Yes

This parameter changes the operation mode of the drive.

Mode of Operation	Action
1	Profile Position Mode (PP)
3	Profile Velocity Mode (PV)
4	Profile Torque Mode (PT)
6	Homing Mode (HM)
7	Interpolated Position Mode (IP)
-1	Analog Speed Mode (AS)
-2	Stepper Emulation Mode (SE)
-3	Sequence Mode (SQ)
-4	reserved
-5	Analog Torque Mode (AT)
-6	Master-Slave Gearbox Mode (GB)
-7	Master-Slave Cam Mode (CM)

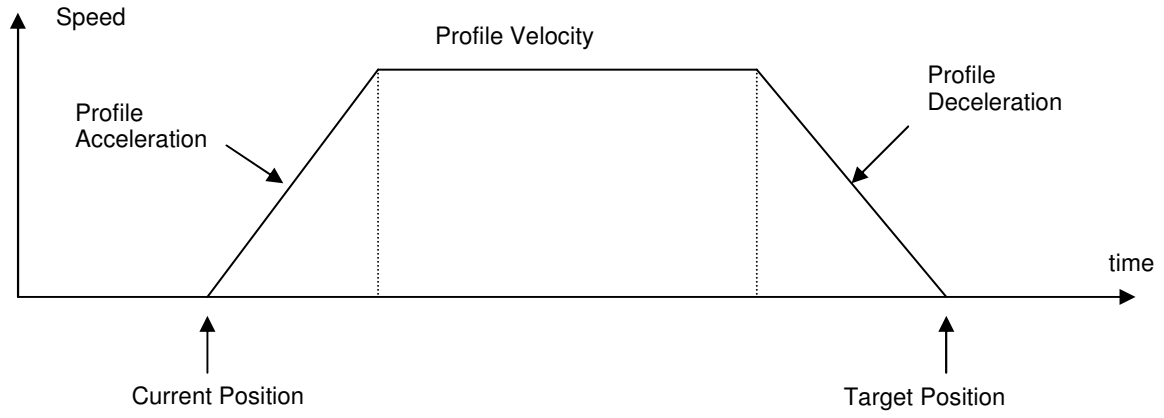
The actual mode is reflected in the operation mode display (object 0x6061).

Index	0x6061
Name	Mode of Operation Display
Object Code	VAR
Data Type	integer8
Object Class	all
Access	ro
PDO Mapping	Yes
Default Value	3

3.2.3.3 - Profile Position Mode

Profile Position Mode

In this mode, a trapezoidal trajectory generator gives the drive the possibility to execute a positioning with preset parameters as target position, profile speed and acceleration.



In profile position mode, these bits in the control word are relative to the control of the trajectory:

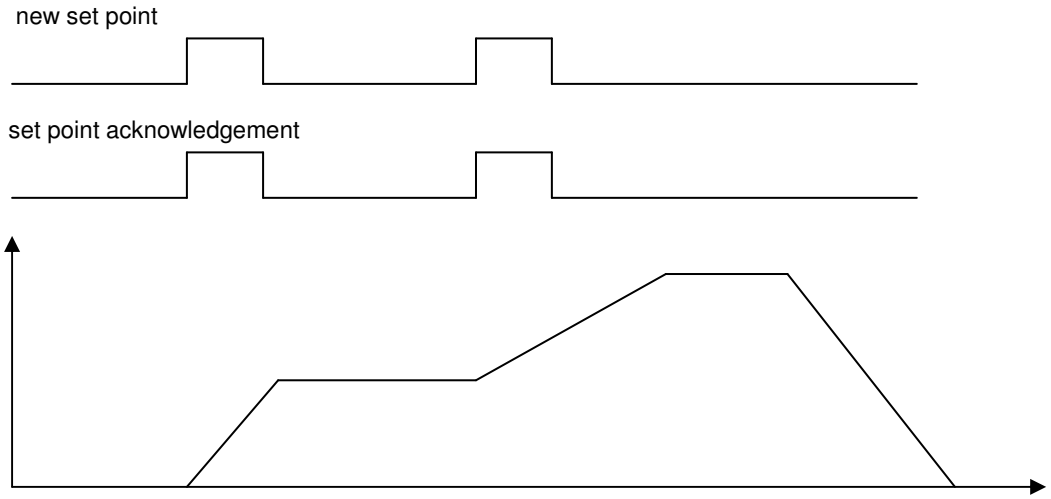
Bit Number	Profile Position Mode
4	new set point
5	change set immediately
6	0: absolute 1: relative

The movement will be triggered by a rising edge of bit 4 (new_set_point) of the control word. The acknowledgement of the new set point is confirmed by bit 12 (setpoint acknowledgement) of the status word. The target position will be taken as relative to the current position if bit 6 of control word = 1.

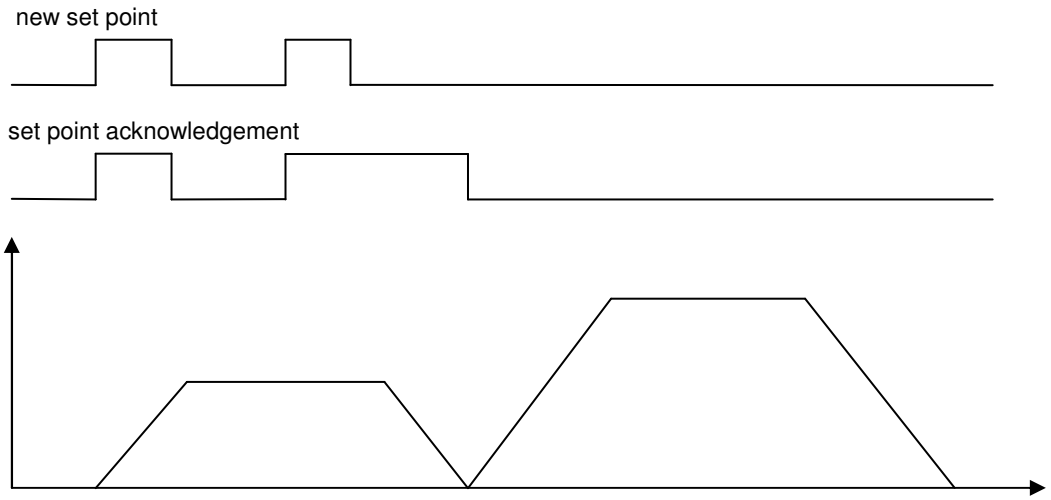
The speed profile is trapezoidal (motion profile type = 0) or S-curve (motion profile type = -1).

Change setpoint immediately

Bit change_set_immediately = 1 :



Bit change_set_immediately = 0 :



Object Dictionary Entries

Index	Object	Name	Type	Attr.
0x607A	VAR	Target Position	Integer32	rw
0x6080	VAR	Max Motor Speed	Unsigned16	rw
0x6081	VAR	Profile Velocity	Unsigned32	rw
0x6082	VAR	End Velocity	Unsigned32	rw
0x6083	VAR	Profile Acceleration	Unsigned32	rw
0x6084	VAR	Profile Deceleration	Unsigned32	rw
0x6086	VAR	Motion Profile Type	Integer16	rw
0x6067	VAR	Position Window	Unsigned32	rw
0x6068	VAR	Position Window Time	Unsigned16	rw
0x607F	VAR	Max Profile Velocity	Unsigned32	rw
0x3081	VAR	Speed Modulation Source	Unsigned32	rw

Index	0x607A
Name	Target Position
Object Code	VAR
Data Type	Integer32
Object Class	pp
Access	rw
PDO Mapping	Yes
Unit	User Position Unit
Value Range	$(-2^{31})..(2^{31}-1)$
Default Value	0

Target position is the final position where the motor will move to in profile position mode. The start position is the current position. The positioning begins with rising edge of bit 4 of the control word (new set point). Bit 6 of control word indicates if the target position is absolute (=0) or relative (=1) movement.

Index	0x6080
Name	Max Motor Speed
Object Code	VAR
Data Type	Integer32
Object Class	all
Access	rw
PDO Mapping	No
Unit	rpm
Value Range	100...60000
Default Value	3000

The *max motor speed* defines the maximum speed the drive can reach. To avoid a saturation of the servo loop, the running speed must be less than *max motor speed* (depends on the overshoot accepted for the servo loop response).

This parameter modifies the value of the Max Profile Velocity 0x607F.

Index	0x6081
Name	Profile Velocity
Object Code	VAR
Data Type	Unsigned32
Object Class	pp
Access	rw
PDO Mapping	Possible
Unit	User Velocity Unit
Value Range	-
Default Value	0x1000

The *Profile Velocity* is the running velocity for a positioning. If the positioning is too short, the profile velocity may not be reached.

Index	0x6082
Name	End Velocity
Object Code	VAR
Data Type	Unsigned32
Object Class	pp
Access	rw
PDO Mapping	Possible
Unit	User Velocity Unit
Value Range	-
Default Value	0

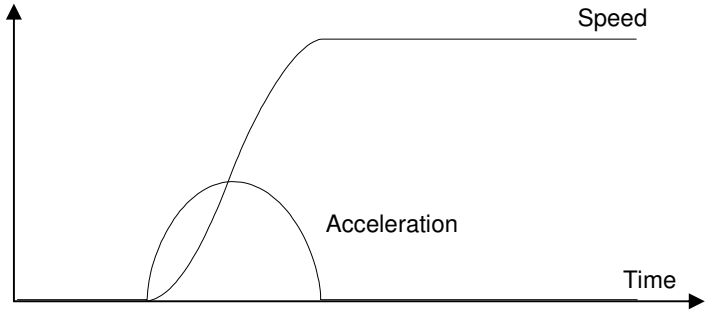
The *End Velocity* is the final velocity value when the target position is reached. When the motor must stop at the target position, *End Velocity=0*.

Index	0x6083
Name	Profile Acceleration
Object Code	VAR
Data Type	Unsigned32
Object Class	pp
Access	rw
PDO Mapping	No
Unit	User acceleration unit
Value Range	-
Default Value	0x10000

Index	0x6084
Name	Profile Deceleration
Object Code	VAR
Data Type	Unsigned32
Object Class	pp
Access	rw
PDO Mapping	No
Unit	User acceleration unit
Value Range	-
Default Value	0x10000

Index	0x6086
Name	Motion Profile Type
Object Code	VAR
Data Type	Integer16
Object Class	pp, sm
Access	rw
PDO Mapping	No
Value Range	0 -> Trapezoidal profile -1 -> S-Curve
Default Value	0

The S-curve is defined by a polynomial. The acceleration profile is therefore parabolic.



Index	0x6067
Name	Postion Window
Object Code	VAR
Data Type	Unsigned32
Object Class	pp
Access	rw
PDO Mapping	No
Unit	User Position Unit
Default Value	0

The *Position Window* defines a symmetrical range of accepted positions relatively to the target position. If the motor current position is within the position window, this target position is considered as reached (bit 10 or status word - Target Reached – is set). If the position window value is 0, the position window control is not active.

When the actual position is within the *Position Window* during the defined *Position Window Time*, the corresponding bit 10 *Target reached* in the *StatusWord* will be set at 1.

Index	0x6068
Name	Position Window Time
Object Code	VAR
Data Type	Unsigned16
Object Class	pp
Access	rw
PDO Mapping	Possible
Unit	Milliseconds
Value Range	0...32767
Default Value	0

Index	0x607F
Name	Max Profile Velocity
Object Code	VAR
Data Type	Unsigned32
Object Class	pv, pp, sm
Access	rw
PDO Mapping	Yes
Unit	User Velocity Unit
Value Range	0...(2^{32}–1)
Default Value	0

The **Max Profile Velocity** is the maximum allowed speed in any direction during a profiled move.
This parameter limits the input velocity reference in:

profile position mode (0x6081),
profile velocity mode (0x60FF),
profile position function block and profile velocity function block in servo mode.

Position Profile Speed Modulation Input Source

Index	0x3081
Name	Position Profile Speed Modulation Input Source
Description	Index/sub-index of input data
Data Type	Unsigned32
Object Class	sm, pp, sq
Access	rw
PDO Mapping	No
Default Value	0
Value	See below

This object allows to connect any dataflow as a speed modulation of the Profile generator in Profile Position Mode or Profile Generator Function Block in Servo Mode or Sequence Mode.

The structure of the entries is the following:

MSB		LSB
Index (16-bit)	Sub-index (8-bit)	0

The value of the modulation is between 0 and 0x7FFF. A value of 0x7FFF of the modulation means 100 % of programmed velocity.
If the value of the input source is negative, then the modulation value is the absolute value.

Position Profile Speed Modulation Configuration

Index	**0x3082**
Name	Position Profile Speed Modulation Configuration
Description	This object allows to define the effect of the Position Profile Speed Modulation signal.
Data Type	Unsigned16
Object Class	all
Access	rw
PDO Mapping	No
Default Value	0
Value	bit description 0 0 normal effect of the Position Profile Speed Modulation signal: 0 -> speed is limited to 0 0x7FFF -> 100% of programmed speed. 1 reverse effect of the Position Profile Speed Modulation signal 0x7FFF -> speed is limited to 0 0 -> 100% of programmed speed. 1..15 reserved

Axis Type

Index	**0x306A**
Name	Axis Type
Object Code	VAR
Data Type	Unsigned8
Object Class	all
Access	rw
PDO Mapping	No
Default Value	0

This parameter define the axis type: linear or rotative.
A linear axis has the software position limit active.

Value	**Function**
0	rotative
1	linear

Software Position Range Limit

The Software Position Range Limit defines a a Positive Position Limit and a Negative Position Limit which act as hardware limit switches.
The Software Position Range Limit is activated when Axis Type (0x3360) is linear.

Index	**0x607F**
Name	Software Position Range Limit
Object Code	ARRAY
Object Class	all
Number of Elements	2

Value Description

Sub Index	1
Description	Negative Position Limit
Data Type	Integer32
Access	rw
PDO Mapping	No
Unit	User position unit
Value	

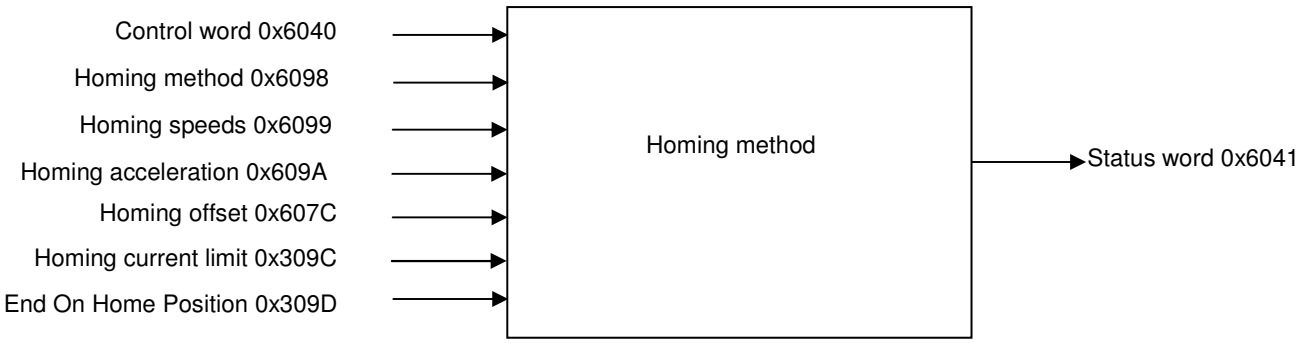
Sub Index	2
Description	Positive Position Limit
Data Type	Integer32
Access	rw
PDO Mapping	No
Unit	User position unit
Value	

3.2.3.4 - Homing Mode

When the feedback sensor does not give the absolute position, the homing mode is the right way to set up the motor to a known position. This position can be detected by using several signals such as positive or negative limit switch, home switch, index pulse or mechanical limit. The choice of the homing method depends on those signals and on the direction of the starting movement.

The drive generates the trajectory according to the homing method. This is the reason why the position loop of the drive is used.

Graphical representation of the trajectories as a function of the input signals:



Index	**Object**	**Name**	**Type**	**Attr.**
0x607C	VAR	Home Offset	Integer32	rw
0x6098	VAR	Homing Method	Integer8	rw
0x6099	ARRAY	Homing Speeds	Unsigned32	rw
0x609A	VAR	Homing Acceleration	Unsigned32	rw

Manufacturer Specific Objects:

Index	Object	Name	Type	Attr.
0x309C	VAR	Homing Current Limit	Unsigned16	rw
0x309D	VAR	End On Home Position	Unsigned16	rw

The homing procedure is launched on rising edge of bit 4 of the Control Word and can be interrupted when clear.

Meanings of operation mode specific bits of the Status Word:

Bit 13	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 successfully completed
1	0	0	Homing error occurred, velocity is not 0
1	0	1	Homing error occurred, velocity is 0
1	1	X	reserved

If Bit 10 is set, this indicates that the velocity is 0.

If bit 12 is set, this indicates that the home position is known but not available.
Bit 12 is reset at 0:
- at power-up,
- if a sensor fault occurs,
- on homing error,
- when homing is starting,
- when bit 4 of the Control Word is at 0.

Bit 13 indicates a homing error:
- homing launched whereas the drive is not in "operation enable" (except for homing method 35);
- homing launched with an unimplemented selected method.

Bit 13 is reset at zero:
- at drive power-up,
- on rising edge of bit 7 of the Control Word.

Homing Offset

The Home Offset defines the position feedback value when the motor reaches the homing position.

Index	0x607C
Name	Home Offset
Object Code	VAR
Data Type	Integer32
Object Class	hm
Access	rw
PDO Mapping	No
Unit	User position unit
Value Range	$(-2^{31})..(2^{31}-1)$
Default Value	0

Homing Method

The *Homing Method* defines various ways of the drive to search the homing position.

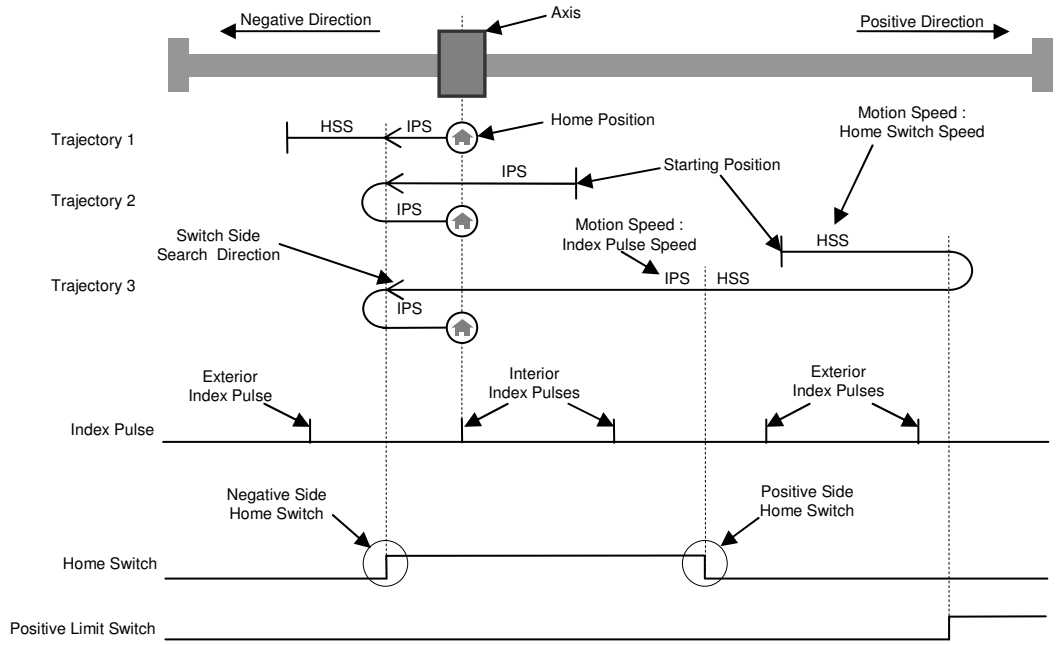
Index	**0x6098**
Name	Homing Method
Object Code	VAR
Data Type	Integer8
Object Class	hm
Access	rw
PDO Mapping	No
Default Value	23h

Value Description
Method supported: 1..14, 17..30, 33..35.
Methods specific: -1, -2, -3, -4.

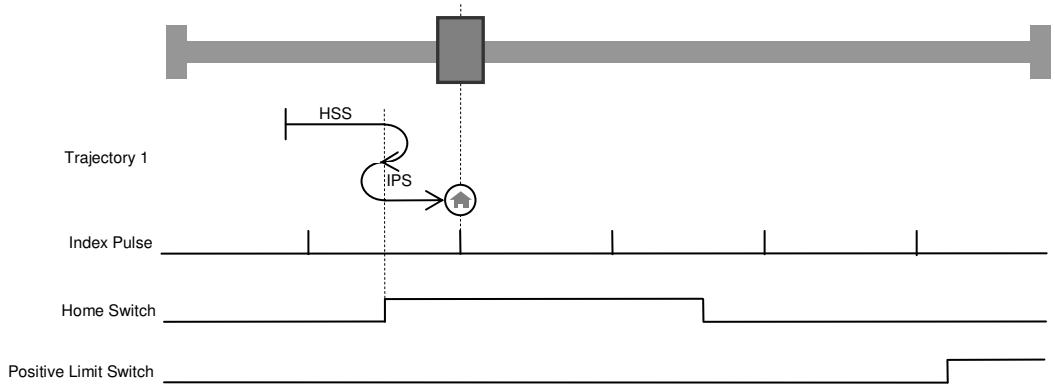
Method	Search for Switch	Search for Index Pulse	Remarks
1	Negative Limit Switch	Exterior	
2	Positive Limit Switch	Exterior	
3	Positive Home Switch	Exterior	
4	Positive Home Switch	Interior	
5	Negative Home Switch	Exterior	
6	Negative Home Switch	Interior	
7	Home Switch, Negative Side	Exterior	Positive Initial Move. Reverse direction on Positive Limit Switch.
8	Home Switch, Negative Side	Interior	Positive Initial Move. Reverse direction on Positive Limit Switch.
9	Home Switch, Positive Side	Interior	Positive Initial Move. Reverse direction on Positive Limit Switch.
10	Home Switch, Positive Side	Exterior	Positive Initial Move. Reverse direction on Positive Limit Switch.
11	Home Switch, Positive Side	Exterior	Negative Initial Move. Reverse direction on Negative Limit Switch.
12	Home Switch, Positive Side	Interior	Negative Initial Move. Reverse direction on Negative Limit Switch.
13	Home Switch, Negative Side	Interior	Negative Initial Move. Reverse direction on Negative Limit Switch.
14	Home Switch, Negative Side	Exterior	Negative Initial Move. Reverse direction on Negative Limit Switch.
17	Negative Limit Switch	-	
18	Positive Limit Switch	-	
19	Positive Home Switch	-	
20	Positive Home Switch	-	
21	Negative Home Switch	-	
22	Negative Home Switch	-	
23	Home Switch, Negative Side	-	
24	Home Switch, Negative Side	-	
25	Home Switch, Positive Side	-	
26	Home Switch, Positive Side	-	
27	Home Switch, Positive Side	-	
28	Home Switch, Positive Side	-	
29	Home Switch, Negative Side	-	
30	Home Switch, Negative Side	-	
33		First Index Pulse	Negative Initial Move.
34		First Index Pulse	Positive Initial Move.
35		-	Homing On Current Position
-1	Mechanical Limit, Negative Move	First Index Pulse	
-2	Mechanical Limit, Positive Move	First Index	

		Pulse	
-3	Mechanical Limit, Negative Move	-	
-4	Mechanical Limit, Positive Move	-	

According to the table above, each homing method can be detailed using a diagram representing all of the possible trajectories.
The homing Method 8 is taken as an example :



For simplifying diagrams, the trajectory of the switch side search is not explicitly drawn. However, an arrow indicates the direction used to search a switch side. Hence, trajectory 1 of homing method 8 is explained in the following diagram:



The following explanation describes only trajectory 1 of homing method 8 taken above as an example.
Using homing method 8, the initial direction of the movement is positive, except if the home switch is active at the motion start. So, the negative side of the home switch is first searched in the positive direction with the Home Switch Speed. When the activation of the home switch is detected, the drive reverses to look for the home switch deactivation. As the home switch has been found, the speed is the slowest home speed, namely the Index Pulse Speed. Once the deactivation of the home switch has been found, the drive reverses to position to look for the Index Pulse. At this stage, depending on the position sensor, the home position will directly be reached, for example a resolver. For sensors like incremental encoders, a search of Index Pulse is achieved in the positive direction and then the drive reverses to position on the captured Index Pulse position.

Homing Speeds

Homing Speeds defines the motor speed when searching the homing position.

Index	**0x6099**
Name	Homing Speeds
Object Code	ARRAY
Number of Elements	2
Data Type	Unsigned32

Value Description

Sub Index	1
Description	Speed during search of switch
Object Class	hm
Access	rw
PDO Mapping	No
Unit	User velocity unit
Default Value	00000019h

Sub Index	2
Description	Speed during search of zero
Object Class	hm
Access	rw
PDO Mapping	No
Unit	User velocity unit
Default Value	0000000Ah

Homing Acceleration

Index	**0x609A**
Name	Homing Acceleration
Object Code	VAR
Data Type	Unsigned32
Object Class	hm
Access	rw
PDO Mapping	No
Unit	User acceleration unit
Default Value	00010000h

Homing Current Limit

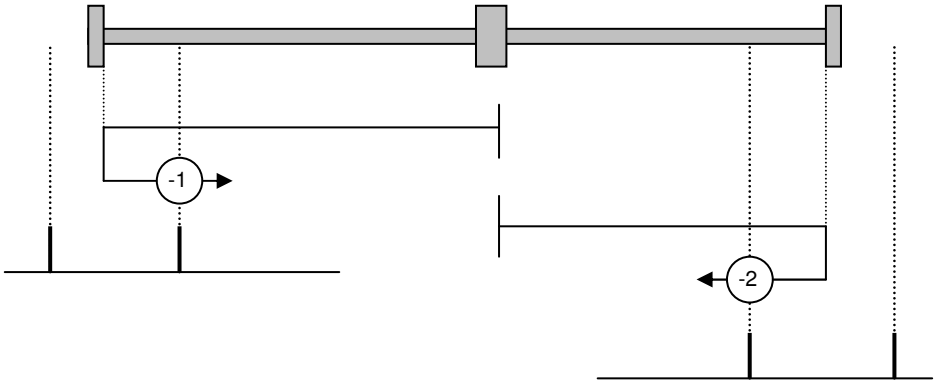
The "Homing current limit" defines the limit of current during the homing on the mechanical limit. The value is defined as a percent of the drive maximum current (defined by object 6510h sub-index 1).

Index	**0x309C**
Name	Homing Current Limit
Object Code	VAR
Data Type	Unsigned16
Object Class	hm
Access	rw
PDO Mapping	No
Unit	%
Conversion	0 to 0x3FFF -> 0% to 100 %
Default Value	0x0400

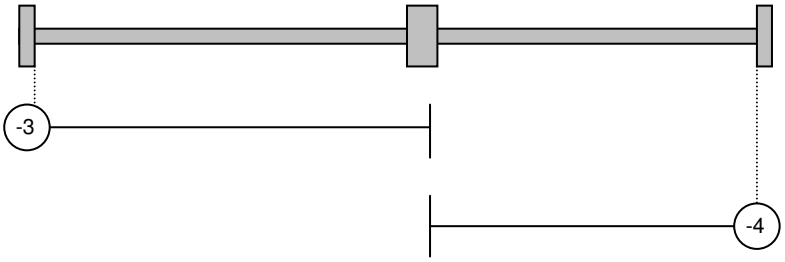
Functional Description

The "Homing Current Limit" parameter defines the limit of current in the motor during the homing procedure. When the mechanical limit is reached, the current in the motor increases up to this limit and the motor speed is 0. This position will be taken as the homing position. An offset value (object 607Ch) can be used to preset the homing position value.

Method -1 and -2 define the homing on the mechanical limit with index pulse.



Method -3 and -4 define the homing on the mechanical limit.



End on Home Position

This parameter allows the drive not to reverse at the end of the homing.
If set at 1, it makes a move to the home position when the homing is over. If cleared, the home position is found but not moved to.

Index	**0x309D**
Name	End on Home Position
Object Code	VAR
Data Type	Unsigned16
Object Class	hm
Access	rw
PDO Mapping	No
Default Value	1

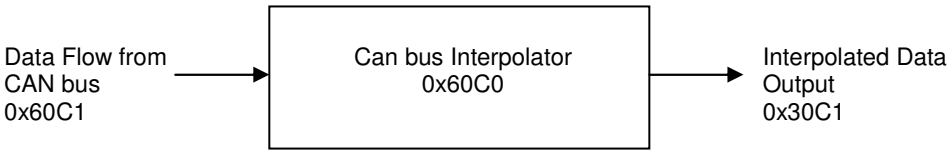
3.2.3.5 - Interpolated Position Mode

Interpolated Position Mode

The interpolated position mode is used to control several axes in coordination. The trajectory must be generated by the host controller and the elementary set point is sent at a fixed cycle time (same as communication cycle time) to all axes.

The cycle time synchronisation of all axes is ensured by the SYNC message. The set point data flow must be sent in real-time.

The elementary set point could be only position if linear interpolation is chosen. The PV interpolation mode requires position and velocity for each set point. The P3 cubic interpolation mode requires only position set point because the interpolator is using the three last position set points. However, the interpolation error is inherent when the acceleration is changing with the P3 cubic interpolation mode.
Both cubic interpolation modes require high position resolution when operating at low speed values. At very low speed, the linear interpolation mode is giving best results.



The CAN bus Interpolator is running in any mode but the result of the interpolator (0x30C1) is applied to the position loop only in Interpolated Position Mode.

When using the linear interpolation, the feedforward acceleration term (KAv) must be cleared (see interpolation and servo loop). Only a PV or P3 interpolation can fully support a feedforward acceleration term.

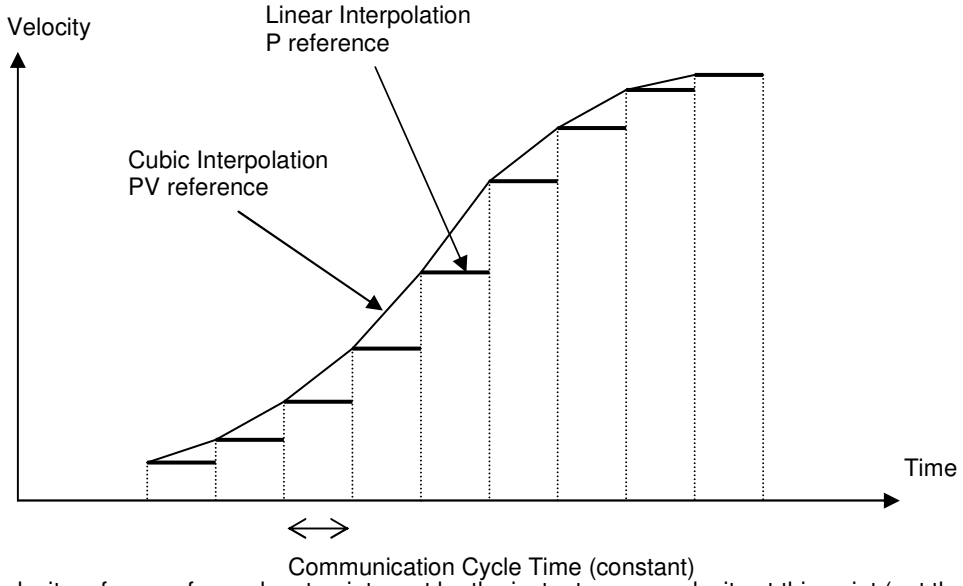
Index	Object	Name	Type	Attr.
0x60C0	VAR	Interpolation Submode Select	Integer16	rw
0x60C1	RECORD	Interpolation Data Record		rw
0x60C4	RECORD	Interpolation Data Configuration		rw
0x30C1	VAR	Interpolated Data Output	Integer32	rw

Interpolation Submode Select

Index	0x60C0
Name	Interpolation Submode Select
Object Code	VAR
Data Type	Integer16
Object Class	ip
Access	rw
PDO Mapping	No
Value Range	see below
Default Value	0

Interpolation Submode Select	Description
0	Linear interpolation
-1	PV interpolation
-2	P3 interpolation

When in linear interpolation mode, only the first parameter of the interpolation data record is used. The data must be the position reference.
When in PV interpolation mode, the first parameter of the interpolation data record must contain the position reference and the second parameter of the interpolation data record contains the velocity reference.



Note: The velocity reference for each set-point must be the instantaneous velocity at this point (not the average velocity).

Interpolation data record

Index	**0x60C1**
Name	Interpolation data record
Object Code	RECORD
Number of Elements	2

Value Description

Sub Index	1
Description	First parameter of ip function
Data Type	Integer32
Object Class	ip
Access	rw
PDO Mapping	Possible

This sub-index contains the position reference in IP mode.

Sub Index	2
Description	Second parameter of ip function
Data Type	Integer32
Object Class	ip
Access	rw
PDO Mapping	Possible

This sub-index contains the speed reference in IP mode if the interpolation submode select (0x60C0) is -1 (interpolation PV). Otherwise it is not used.

Absolute 16-bit Position Reference for IP mode

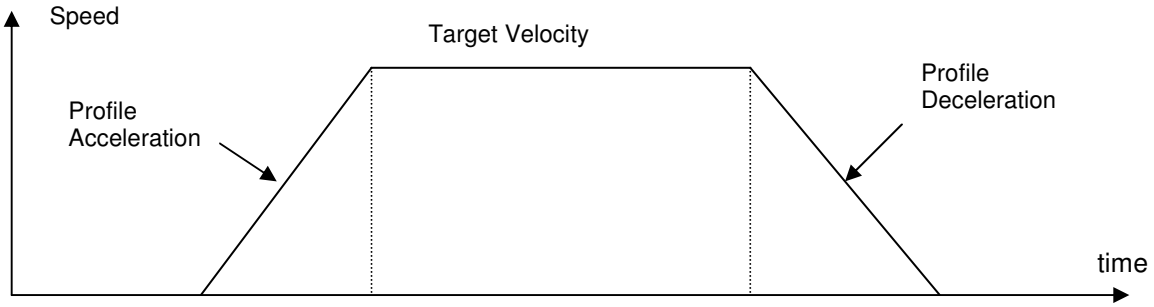
Index	0x3350
Name	Absolute 16-bit Position Reference
Object Code	VAR
Data Type	Unsigned8
Object Class	ip
Access	rw
PDO Mapping	No
Value Range	0..1
Default Value	0

The position reference in interpolated position mode can be defined as 16-bits only. This is to reduce the bus traffic.
When in 16-bit mode (object 3350h = 1), the position reference in object 60C1-1 via PDO is set at 16 bits and the drive calculates the upper word. At the beginning, it is necessary to set the upper word with object 60C1-1 via SDO (Integer32). The mapping of RPDO must be changed to object 60C1 sub-index 1 with 16-bit length.

3.2.3.6 - Profile Velocity Mode

Profile Velocity Mode

The profile velocity mode authorizes the drive to operate with a velocity reference. Only speed loop and current loop are closed in this mode.



Index	Object	Name	Type	Attr.
0x606B	VAR	Velocity Demand Value	Integer32	ro
0x606C	VAR	Velocity Actual Value	Integer32	ro
0x306C	VAR	Actual Velocity Filter	Unsigned16	rw
0x3069	VAR	Velocity Actual Value (rpm)	Integer32	ro
0x60FF	VAR	Target Velocity	Integer32	rw
0x6083	VAR	Profile Acceleration	Unsigned32	rw
0x6084	VAR	Profile Deceleration	Unsigned32	rw
0x606D	VAR	Velocity Window	Unsigned16	rw
0x606E	VAR	Velocity Window Time	Unsigned16	rw
0x606F	VAR	Velocity Threshold	Unsigned16	rw
0x6070	VAR	Velocity Threshold Time	Unsigned16	rw
0x30FF	VAR	Target Velocity Source	Unsigned16	rw

Index	0x6083
Name	Profile Acceleration
Object Code	VAR
Data Type	Unsigned32
Object Class	pp
Access	rw
PDO Mapping	No
Unit	User acceleration unit
Value Range	-
Default Value	0x10000

Index	0x6084
Name	Profile Deceleration
Object Code	VAR
Data Type	Unsigned32
Object Class	pp
Access	rw
PDO Mapping	No
Unit	User acceleration unit
Value Range	-
Default Value	0x10000

The **Velocity Window** defines a symmetrical range of accepted velocity relatively to the target velocity. If the motor current velocity is within the velocity window, this target velocity is considered as reached (bit 10 of status word - Target Reached – is set). If the velocity window value is 0, the velocity window control is not active.

Index	0x606D
Name	Velocity Window
Object Code	VAR
Data Type	Unsigned32
Object Class	pv
Access	rw
PDO Mapping	No
Unit	Velocity Unit
Default Value	0

When the actual velocity is within the **Velocity Window** during the defined **Velocity Window Time**, the corresponding bit 10 Target reached in the StatusWord will be set to 1.

Index	0x606E
Name	Velocity Window Time
Object Code	VAR
Data Type	Unsigned16
Object Class	pv
Access	rw
PDO Mapping	Possible
Unit	ms
Value Range	0...32767
Default Value	0

The **Velocity Threshold** defines a symmetrical range of accepted velocity relatively to the 0. If the motor current velocity is within the velocity threshold, this 0 velocity is considered as reached (bit 12 of status word - Velocity = 0 – is set). If the velocity threshold value is 0, the velocity threshold control is not active.

Index	**0x606F**
Name	Velocity Threshold
Object Code	VAR
Data Type	Unsigned32
Object Class	pv
Access	rw
PDO Mapping	No
Unit	Velocity Unit
Default Value	0

When the actual velocity is within the *Velocity Threshold* during the defined *Velocity Threshold Time*, the corresponding bit 12 *Velocity=0* in the *StatusWord* will be set at 1.

Index	**0x6070**
Name	Velocity Threshold Time
Object Code	VAR
Data Type	Unsigned16
Object Class	pv
Access	rw
PDO Mapping	Possible
Unit	ms
Value Range	0...32767
Default Value	0

Profile Velocity Mode Input Source

Index	**0x30FF**
Name	Profile Velocity Mode Input Source for Target Velocity
Description	Index/sub-index of input data
Data Type	Unsigned32
Class	pv
Access	rw
PDO Mapping	No
Value	See below
Default Value	0x60FF0000

This object allows to connect any 32-bit dataflow as target velocity for the Profile Velocity Mode.

The structure of the entries is the following:

MSB		LSB
Index (16-bit)	Sub-index (8-bit)	0

Example:
0x30FF,0 = 0x30F10200
connects the analog input as the target velocity for Profile Velocity Mode.

3.2.3.7 - Profile Torque Mode

Profile Torque Mode

In this mode, the drive operates only with current loops and there is no speed or position control.

Object Dictionary Entries

Index	Object	Name	Type	Attr.
0x6071	VAR	Target Torque	Integer16	rw
0x3071	VAR	Target Torque Input Source	Unsigned32	rw
0x6087	VAR	Torque Slope	Unsigned32	rw
0x6088	VAR	Torque Profile Type	Integer16	rw
0x60B2	VAR	Offset Torque	Integer16	rw
0x6074	VAR	Torque Demand Value	Integer16	ro
0x6077	VAR	Torque Actual Value	Integer16	ro
0x6078	VAR	Current Actual Value	Integer16	ro
0x6079	VAR	DC Voltage	Integer16	ro

The *Target Torque* is the input value for the current loop in profile torque mode. The value is given per thousand of rated current (0x6075).

Index	0x6071
Name	Target Torque
Object Code	VAR
Data Type	Integer16
Object Class	pt
Access	rw
PDO Mapping	Possible
Unit	per thousand of rated current (0x6075)
Value Range	-
Default Value	0

Profile Torque Mode Input Source

Index	0x3071
Name	Profile Torque Mode Input Source for Target Torque
Description	Index/sub-index of input data
Data Type	Unsigned32
Class	pt
Access	rw
PDO Mapping	No
Value	See below
Default Value	0x60710000

This object allows to connect any 16-bit dataflow as a target torque for the Profile Torque Mode.

The structure of the entries is the following:

MSB		LSB
Index (16-bit)	Sub-index (8-bit)	0

Example:
0x3071,0 = 0x30F10100
connects analog input 1 as the target torque for Profile Torque Mode.

This parameter defines the torque slope when target torque is changed.

Index	0x6087
Name	Torque Slope
Object Code	VAR
Data Type	Unsigned32
Object Class	pt
Access	rw
PDO Mapping	No
Unit	per thousand of rated current per second
Value Range	-
Default Value	0x10000

The "DC Voltage" gives the value of the DC voltage in the drive. This signal is filtered by a low-pass filter (0x3408-2)

Index	0x6079
Name	DC Voltage
Object Code	VAR
Data Type	Integer32
Object Class	all
Access	ro
PDO Mapping	Yes
Unit	mV
Value Range	-
Default Value	-

3.2.3.8 - Sequence Mode

The purpose of the sequencer mode is to allow basic moves.

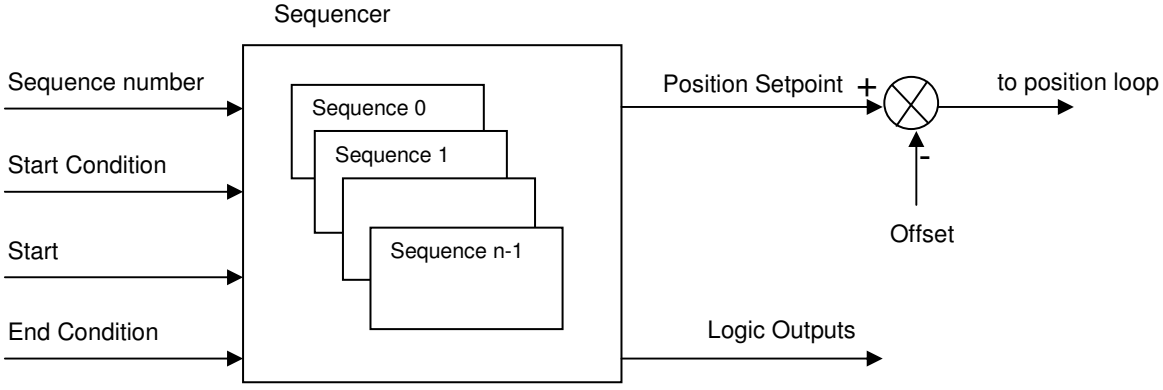
This basic move is called a sequence and a list of sequences can be pre-programmed and stored in the drive.

Each sequence is identified with a number (sequence number).
The maximum number of sequences for a given drive is showed in object 0x3612

The difference type of sequence are:
- Positioning sequence
- Homing sequence
- Speed sequence
- Torque sequence
- Gearing sequence

NOTE: depending on model and/or firmware version, not all above sequence types are supported. The sequence types supported are showed in object 0x360F

Various sequences can be linked sequentially together to build a complex move.



Sequence Number: allows the selection of the sequence to be executed. The "Sequence Number" can be connected to physical logic inputs or set via the fieldbus to select the sequence.

Start Condition: A Logic bits pattern can be defined as a condition for a sequence to be started. The "Start Condition" can be connected to physical logic inputs or to a variable via the fieldbus.

Start: A trigger signal (rising edge of start bit) allows to start the sequence which number is set by a sequence number and if the start condition is fulfilled.
If the start condition is not ok, the movement will not be executed until the start condition is valid.
A sequence is started with bit 4 of control word (0x6040) and stopped with bit 5 of control word.

End Condition: In some sequences, if an "End Condition" is defined, the sequence will finished when the "End Condition" is valid. The "End Condition" is defined by bits pattern (bits equal to 0, bits equal to 1...), and can be connected to physical logic inputs or to a variable via the fieldbus.

Control Word (0x6040):

Bit	Action
4	↑ start sequence
5	1 stop sequence
6	reserved

Status Word (0x6041):

Bit	Action
10	Target Reached
12	POS
13	SEQ

Sequence Chaining

The sequences chaining is controlled by the “SeqNext”, “SeqCount”, “SeqLink” and "StartCond" parameters.

Sequence Parameters

The parameters of all sequences are stored in a RAM memory (sequence memory).
These sequence parameters can be set:

- by parameter values defined in a sequence file named SEQUENCE.TXT (see Sequence File format).
- by direct access to the sequence parameters via appropriate objects.

Sequence Files

Loading a sequence file:

- all sequence parameters in the sequence memory will be erased by sequences defined in SEQUENCE.TXT
- if a sequence is not defined in SEQUENCE.TXT, then the sequence will be cleared.
- the SEQUENCE.TXT file will be loaded into the sequence memory when the 24V supply is applied
- the SEQUENCE.TXT file will be loaded into the sequence memory when writing into object 0x1011 with the signature = 0x7165736C (lseq)

Merging sequence file:

- only sequences defined in SEQUENCE.TXT will be loaded into the sequence memory; other sequences in the memory are not modified.
- the SEQUENCE.TXT file can be merged in sequence memory when writing into object 0x1011 with the signature = 0x7165736D (mseq).

Objects Definition

Sequence Control

These objects allow to control the execution of a sequence.

Index	Object	Name	Type	Attr.
0x3601	ARRAY	Sequence Inputs		rw
0x3602	ARRAY	Sequence Outputs		rw
0x3603	VAR	Minimum Sequence Pulse	Unsigned16	rw
0x3604	RECORD	Output Pulse Configuration		rw
0x3605	VAR	Sequence phase	Unsigned16	rw
0x360B	VAR	Sequence Capture Position	integer32	rw
0x360F	VAR	Supported Sequence Type	Unsigned16	ro
0x3612	VAR	Maximum Sequences Supported	Unsigned16	ro

Sequence Parameters

These objects allow to access directly any parameter of any sequence.

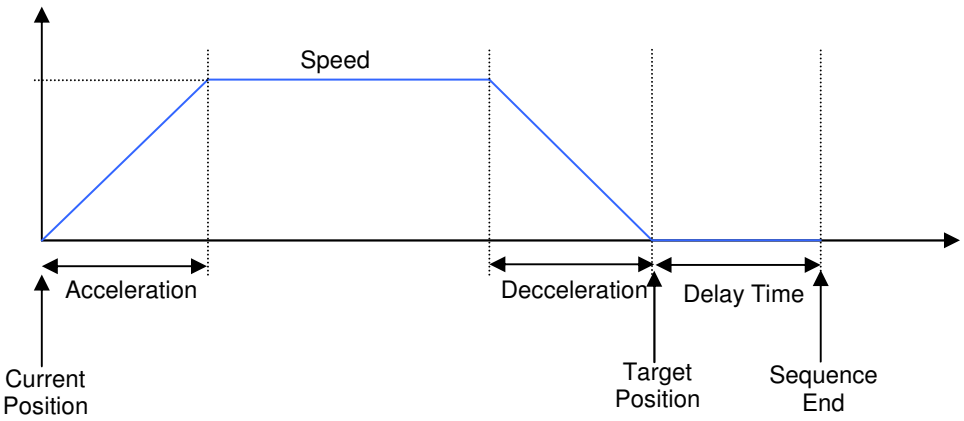
The selected sequence number is defined by object 0x3610, and all sequence parameters are accessed by object 0x3611.

Index	Object	Name	Type	Attr.
0x3610	VAR	Sequence Parameters Number	integer16	rw
0x3611	RECORD	Sequence Parameters		rw

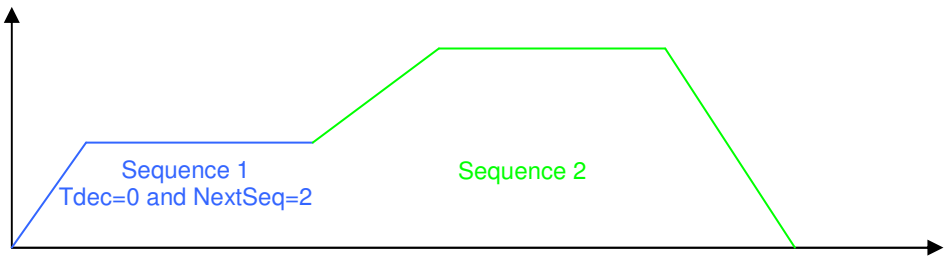
3.2.3.8.1. Positioning Sequence

The main parameters of a positioning sequence are:

- The position to be reached (absolute or relative)
- The motion speed
- The acceleration time
- The deceleration Time
- The delay time at the end of the motion



Example of 2 positioning sequences without stopping (the deceleration ramp of the first sequence is 0).



Sequence 1:
SeqType = POS
Speed = 150000
AccelTime = 400
DecelTime = 0
NextSeq = 2

Sequence 2:
SeqType = POS
Speed = 250000
AccelTime = 300
DecelTime = 400

Supported keyword and parameters for a positioning sequence

Keyword	Direct parameter entry	Description
SeqType	0x3611-1	value = POS for SEQUENCE.TXT file or value = 1 for direct parameter object
NextSeq	0x3611-2	see sequence parameters
SeqCount	0x3611-3	see sequence parameters
SeqLink	0x3611-4	see sequence parameters
Trigger	0x3611-5	see sequence parameters
Output	0x3611-6 0x3611-7 0x3611-8	see sequence parameters
StartCond	0x3611-9 0x3611-10	see sequence parameters
Tempo	0x3611-23	see sequence parameters
Speed	0x3611-15	defines the speed setpoint of the sequence in velocity unit
Speed2	0x3611-16	defines the speed setpoint at the end of the sequence in velocity unit
Accel	0x3611-17	defines the acceleration time in user unit per square second
Decel	0x3611-18	defines the deceleration time in user unit per square second
Position	0x3611-13	defines the position setpoint in user unit
EndCond	0x3611-11 0x3611-12	see sequence parameters

3.2.3.8.2. Homing Sequence

The Home sequence allows to perform a homing procedure.

The main parameters of a home sequence are:

- Home Offset
- Home method
- Speeds
- Acceleration
- Current limit (Torque Limit) for method -1, -2, -3 and -4.

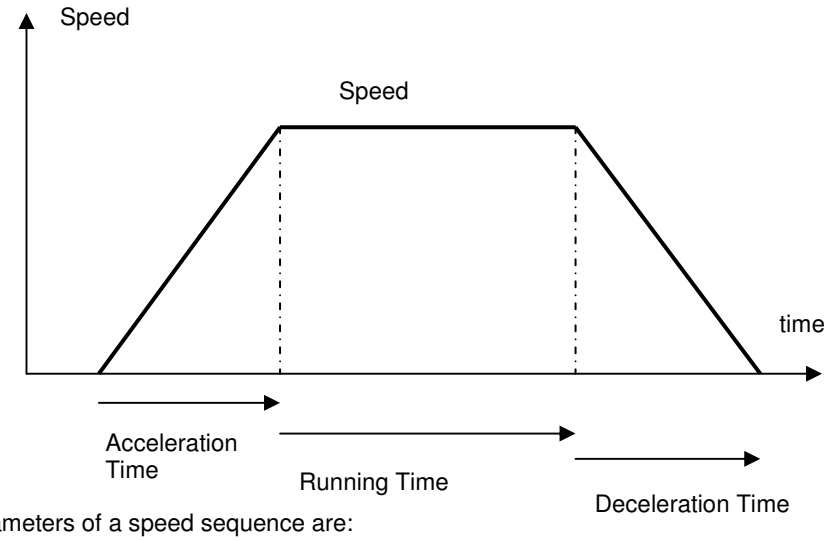
The Home sequence runs like in Homing Mode.

Supported keywords and parameters for a home sequence

Keyword	Direct parameter entry	Description
SeqType	0x3611-1	value = HOME for SEQUENCE.TXT file or value = 2 for direct parameter object
NextSeq	0x3611-2	see sequence parameters
SeqCount	0x3611-3	see sequence parameters
SeqLink	0x3611-4	see sequence parameters
Trigger	0x3611-5	see sequence parameters
Output	0x3611-6 0x3611-7 0x3611-8	see sequence parameters
StartCond	0x3611-9 0x3611-10	see sequence parameters
Method	0x3611-22	defines various ways of the drive to search the homing position
Home offset	0x3611-13	defines the position value when the motor reaches the homing position
Speed	0x3611-15	defines the speed during search of switch (velocity unit)
Speed2	0x3611-16	defines the speed during search of zero (velocity unit)
Accel	0x3611-19	defines the acceleration time in acceleration unit
Current Limit	0x3611-25	defines the current limit in per thousand of the rated current for a homing on mechanical limit
EndCond	0x3611-11 0x3611-12	see sequence parameters

3.2.3.8.3. Speed Sequence

The speed sequence allows to move the axis with a profile speed as follows:



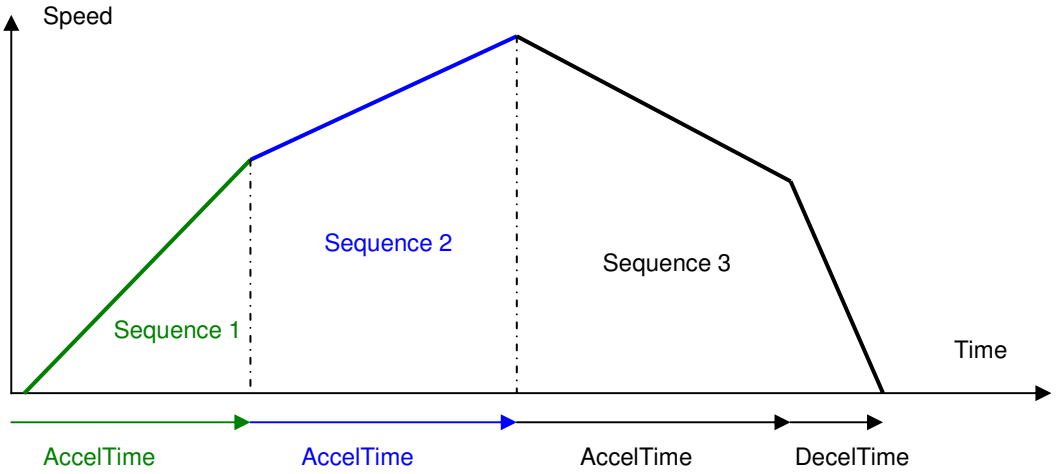
The main parameters of a speed sequence are:

- Speed setpoint
- Acceleration Time
- Deceleration Time
- Running Time

If the Running Time is 65535 (maximum of 16-bit) then the running phase will be executed forever. An "End Condition" can be used to exit this sequence.

If the deceleration Time is 0, then the sequence will end up after the running phase. This allows to combine several sequences for a special profile.

Example of combined sequences:



Sequence 1:
SeqType = SPEED
Speed = 150000
AccelTime = 400
RunTime = 0
DecelTime = 0
NextSeq = 2

Sequence 2:
SeqType = SPEED
Speed = 250000
AccelTime = 400
RunTime = 0
DecelTime = 0
NextSeq = 3

Sequence 3:
SeqType = SPEED
Speed = 140000
RunTime = 0
AccelTime = 400
DecelTime = 150

The speed setpoint of the Speed Sequence is also limited by the value of the Speed Modulation (0x3081). If the speed modulation is defined, then the sequence speed will be reduced by the speed modulation value.

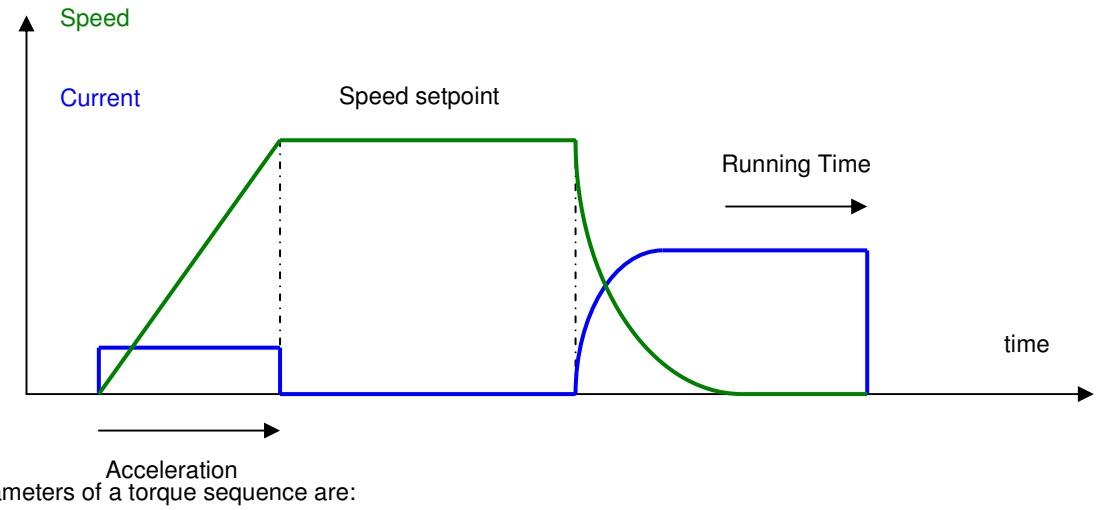
upported keyword and parameters for a speed sequence

Keyword	Direct parameter entry	Description
SeqType	0x3611-1	value = SPEED for SEQUENCE.TXT file or value = 3 for direct parameter object
NextSeq	0x3611-2	see sequence parameters
SeqCount	0x3611-3	see sequence parameters
SeqLink	0x3611-4	see sequence parameters
Trigger	0x3611-5	see sequence parameters
Output	0x3611-6	see sequence parameters
	0x3611-7	
	0x3611-8	
StartCond	0x3611-9	see sequence parameters
	0x3611-10	
Tempo	0x3611-23	see sequence parameters

Speed	0x3611-15	defines the speed setpoint the this sequence in velocity unit
AccelTime	0x3611-19	defines the acceleration time in ms.
DecelTime	0x3611-20	defines the deceleration time in ms.
RunTime	0x3611-24	defines the running time in ms. A value of 65535 corresponds to an infinite running time.
EndCond	0x3611-11 0x3611-12	see sequence parameters

3.2.3.8.4. Torque Sequence

The torque sequence allows to move the axis with a profile speed and a current limit.



The main parameters of a torque sequence are:

- Speed setpoint
- Acceleration
- Running Time
- Current limit (Torque Limit)

In the torque control sequence, the motor is running at the speed set point value until the current rises up to the limit value. The motor running direction depends on the sign of the speed set point. When the current limitation is reached, the amplifier is holding this current during the time interval defined by the Running Time parameter. If the Running Time = 65535, the torque holding time is infinite. In this case the sequence can be left by an end condition.
At the end of the Running Time, the current position will be captured in object 0x360B.

Notes:

When Torque Sequence is executed, the position following error is disabled.

The Torque Sequence speed is also limited by the value of the Speed Modulation (0x3081). If the speed modulation is defined, then the sequence speed will be reduced by the speed modulation value.

Supported keywords and parameters for a torque sequence

Keyword	**Direct parameter entry**	**Description**
SeqType	0x3611-1	Value = TORQUE for SEQUENCE.TXT file or Value = 4 for direct parameter object
NextSeq	0x3611-2	See sequence parameters
SeqCount	0x3611-3	See sequence parameters
SeqLink	0x3611-4	See sequence parameters
Trigger	0x3611-5	See sequence parameters
Output	0x3611-6 0x3611-7 0x3611-8	See sequence parameters
StartCond	0x3611-9 0x3611-10	See sequence parameters

Speed	0x3611-15	Defines the speed setpoint of this sequence in velocity unit
Accel	0x3611-19	Defines the acceleration time in acceleration unit
RunTime	0x3611-24	Defines the running time in ms. A value of 65535 corresponds a infinite running time.
Torque	0x3611-25	Defines the current limit in per thousand of the rated current
EndCond	0x3611-11 0x3611-12	See sequence parameters

3.2.3.8.5. Gearing Sequence

The Gearing sequence is a sequence with gearbox function (see Gearbox Function for more information).

Gearing sequence parameters:
The main parameters for a gearing sequence are:

config:	defines the gearing behavious: - Exit Mode - Trigger Mode - Start Mode - Ratio Set Select - Ratio Modulation Enable This parameter is defined as in gearbox configuration 0x3928,1
factor:	defines gearing ratio factor value
acceleration:	defines acceleration value for slave for acceleration phase, deceleration phase and slave phase shift adjustment.
differential speed:	defines the differential speed for slave phase shift adjustment.
master distance:	defines the distance for the master from the start to the synchronization point.
slave distance:	defines the distance for the slave from the start to the synchronization point.
synchronization distance:	defines the distance for the slave that the slave must be position synchronized. If the synchronization distance is 0, then the slave will synchronize with the master indefinitely. A stop condition can be used to exit the gearing sequence.

The master distance and the slave distance parameters must be adjusted so that the slave is synchronized before the synchronization point.

Gearing global parameters:

Beside parameters defined in a sequence, the gearing function has other global parameters which are not defined in the sequence and are applied to all gearing sequences.

Master input:	defines - master inputs source - master start position - hardware inputs for starting.
Gearing ratio sets:	2 sets of selectable gearring ratio (numeration and denominator) can be used
Gearing control:	allows to control a gearing sequence by an external source (i.e. via fieldbus or hardware input): - ratio set select - Slave Phase Shift start
Slave Phase Shift Distance:	defines the Slave Phase Shift value for slave position adjustment.

Supported keyword and parameters for a gearing sequence in sequence file:

Keyword	**Direct parameter entry**	**Description**
SeqType	0x3611-1	value = GEAR for SEQUENCE.TXT file or value = 5 for direct parameter object
NextSeq	0x3611-2	see sequence parameters
SeqCount	0x3611-3	see sequence parameters
SeqLink	0x3611-4	see sequence parameters
Trigger	0x3611-5	see sequence parameters
Output	0x3611-6 0x3611-7 0x3611-8	see sequence parameters
StartCond	0x3611-9 0x3611-10	see sequence parameters

Tempo	0x3611-23	see sequence parameters
Config	0x3611-21	defines the gearing behaviour
Factor	0x3611-26	defines the gearing ratio factor
Speed	0x3611-15	defines the differential speed for the slave phase shift adjustment
Accel	0x3611-19	defines the acceleration value for acceleration phase, deceleration phase and slave phase shift adjustment
MasterDtn	0x3611-13	defines the master distance
SlaveDtn	0x3611-14	defines the slave distance
SyncDtn	0x3611-16	defines the synchronization distance
EndCond	0x3611-11 0x3611-12	see sequence parameters

3.2.3.8.6. Sequence Chaining

The sequence chaining is controlled by 4 parameters:

- SeqCount,
- SeqNext,
- SeqLink,
- and StartCond.

"SeqCount" defines how many times this sequence will be executed. Then the sequencer will link to SeqNext if the counter is not 0 or link to SeqLink if the counter has expired.
There must be only one SeqCount at a time.

"SeqNext" defines the sequence to be executed after the current one.

When a sequence is started:

If "StartCond" is defined:
If "start condition" is valid then the sequence will be executed and then link "SeqNext"
If "Start condition" is not valid then the sequence is not executed but jump to "SeqLink"

If "StartCond" is not defined:
the sequence will be executed and then link "SeqNext".

COUNTER LOOP

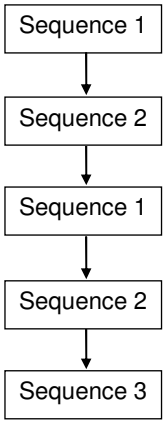
The sequences linkage is controlled by the “SeqNext”, “SeqCount” and “SeqLink” parameters.

Application example:

Sequence 1:
SeqCount = 0
SeqNext = 2
SeqLink = -1
Sequence 2:
SeqCount = 2
SeqNext = 3
SeqLink = 1
Sequence 3:
SeqCount = 0
SeqNext = -1
SeqLink = -1

Note: SeqNext = -1 or SeqLink = -1 corresponds to an empty field in the Gem Drive Studio software.

If the execution is starting at sequence 1, the program will be the following:



Start of sequence 1, then connection to sequence 2 (parameter "SeqNext ")

First execution of sequence 2, then connection to sequence 1 (parameter "SeqLink")

Execution of sequence 1, then connection to sequence 2 (parameter "SeqNext ")

Second execution of sequence 2, then connection to sequence 3 (parameter "SeqNext ")

Execution of sequence 3, then end of the program.

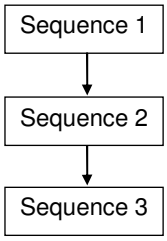
CONDITIONAL JUMP

The conditional jump is controlled by using the “StartCond” and the “SeqNext”, “SeqCount” and “SeqLink” parameters.

Application example:

Sequence 1:
SeqNext = 2
SeqCount = 0
SeqLink = -1
Sequence 2:
SeqNext = 3
SeqCount = 0
SeqLink = 4
Start Cond = "1......."
Sequence 3:
SeqNext = -1
SeqCount = 0
SeqLink = -1
Sequence 4:
SeqNext = -1
SeqCount = 0
SeqLink = -1

If the execution is starting at sequence 1 and logic input 8 is activated, the program will be the following:

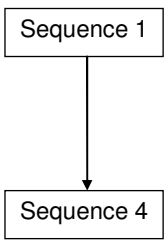


Start of sequence 1, then connection to sequence 2 (parameter "SeqNext ")

Execution of sequence 2, then connection to sequence 3
(start condition valid and parameter "SeqNext ")

Execution of sequence 3, then end of the program.

If the execution is starting at sequence 1 and logic input 8 is deactivated, the program will be the following:



Start of sequence 1, then connection to sequence 2 (parameter "SeqNext ")

No execution of sequence 2, connection to sequence 4
(start condition not valid and parameter "SeqLink")

Execution of sequence 4, then end of the program.

3.2.3.8.7. Sequence Parameters

Sequence Parameters

Supported keyword and parameters for all sequences

Keyword	Direct parameter entry	Description
SeqNb	0x3610	selects the sequence number (0..127)
SeqType	0x3611-1	this parameter defines the sequence type: POS (1) HOME (2) SPEED (3) TORQUE (4) GEAR (5) in parenthesis is the value for direct parameter in object 0x3611-1
NextSeq	0x3611-2	defines the next sequence to be executed after the current one if there is no condition or counter
SeqCount	0x3611-3	defines how many times the sequence must be executed. This counter is decremented each time a sequence is over.
SeqLink	0x3611-4	defines the number of the sequence to be executed when the SeqCount is not 0
Trigger	0x3611-5	defines the output triggering event
Output	0x3611-6	defines the output bit which will be reset
	0x3611-7	defines the output bit which will be set
	0x3611-8	defines the output bit which will be toggled
StartCond	0x3611-9	defines the condition bit which starts the sequence when equals 0
	0x3611-10	defines the condition bit which starts the sequence when equal to 1
Tempo	0x3611-23	defines the delay time in ms at the end of the positioning
EndCond	0x3611-11	defines the condition bit which stops the sequence when equal to 0
	0x3611-12	defines the condition bit which stops the sequence when equal to 1

Sequence Inputs

Index	**0x3601**
Name	Sequence Inputs
Object Code	RECORD
Number of Elements	3

Value Description

Sub Index	1
Description	Sequence Number Input
Data Type	Integer16
Object Class	sq
Access	ro
PDO Mapping	Yes
Default Value	0

This object defines the sequence that will be executed when START is rising up.

Sub Index	2
Description	Executed Sequence Number
Data Type	Integer16
Object Class	sq
Access	rw
PDO Mapping	Yes
Default Value	-

This object indicates the currently running sequence.
A value of -1 means no sequence is running.

Sub Index	3
Description	Conditional Input
Data Type	Integer16
Object Class	sq
Access	rw
PDO Mapping	Yes
Default Value	0

This object defines the bits pattern which is used for start condition or end condition.

Sequence Outputs

Index	**0x3602**
Name	Sequence Outputs
Object Code	RECORD
Number of Elements	4

Value Description

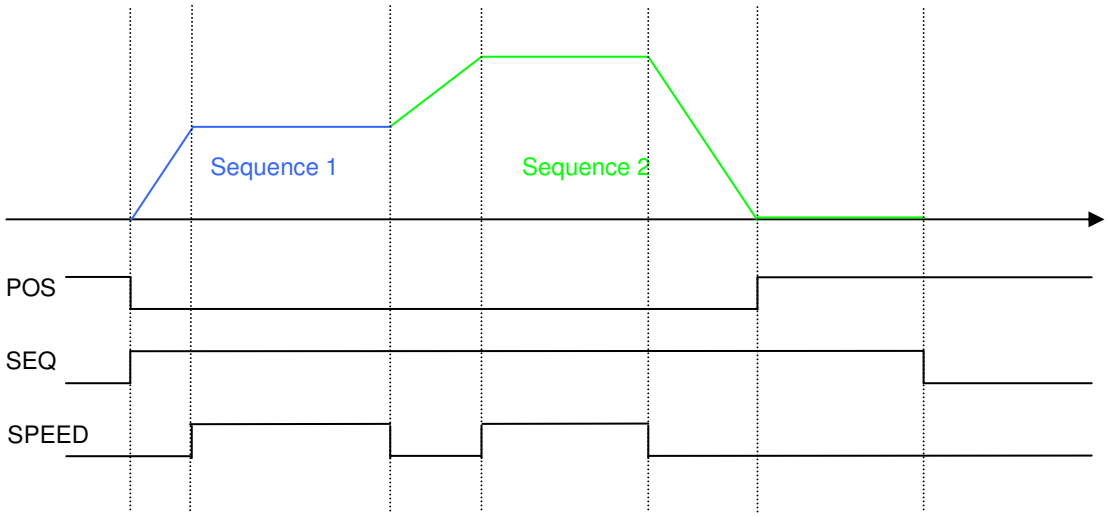
Sub Index	1
Description	Programmable Logic Outputs
Data Type	Unsigned16
Object Class	sq
Access	rw
PDO Mapping	Yes
Default Value	

Sub Index	2
Description	Programmable Logic Outputs Polarity
Data Type	Unsigned16
Object Class	sq
Access	rw
PDO Mapping	No
Default Value	0

Value	**Description**
0	For a positive polarity
1	For a negative polarity

Sub Index	3
Description	Dedicated Logic Outputs
Data Type	Unsigned16
Object Class	sq
Access	rw
PDO Mapping	Yes
Default Value	

Bit	Designation	Description
0	POS	This signal is activated when the motor reaches the position and remains enabled until the next motor movement
1	SEQ	This signal indicates that a sequence is currently executed
2	SPEED	This signal indicated that the speed set point is reached during a movement
3	READY	This signal is activated when the drive is OK



Sub Index	4
Description	Dedicated Logic Outputs Polarity
Data Type	Unsigned16
Object Class	sq
Access	rw
PDO Mapping	No
Default Value	0

Value	Description
0	For a positive polarity
1	For a negative polarity

Minimum Sequence Pulse

This function is useful for the detection of a sequence with a short duration.

Index	0x3603
Name	Minimum Sequence Pulse
Object Code	VAR
Data Type	Unsigned16
Object Class	Sq
Access	rw
PDO Mapping	No
Unit	ms
Value Range	0 this function is not activated 1…65535 this function defines the minimum duration of the SEQ output
Default Value	0

Sequence Outputs

Index	**0x3604**
Name	Output Pulse Configuration
Object Code	RECORD
Number of Elements	2

Value Description

Sub Index	1
Description	Output Pulse
Data Type	Unsigned16
Object Class	sq
Access	rw
PDO Mapping	No
Value Range	0 the bit number is configured as Output 1 the bit number is configured as Output Pulse
Default Value	0

Sub Index	2
Description	Output Pulse Duration
Data Type	Unsigned16
Object Class	sq
Access	rw
PDO Mapping	No
Unit	ms
Value Range	1…16000
Default Value	0

This parameter defines the duration of the output activation.

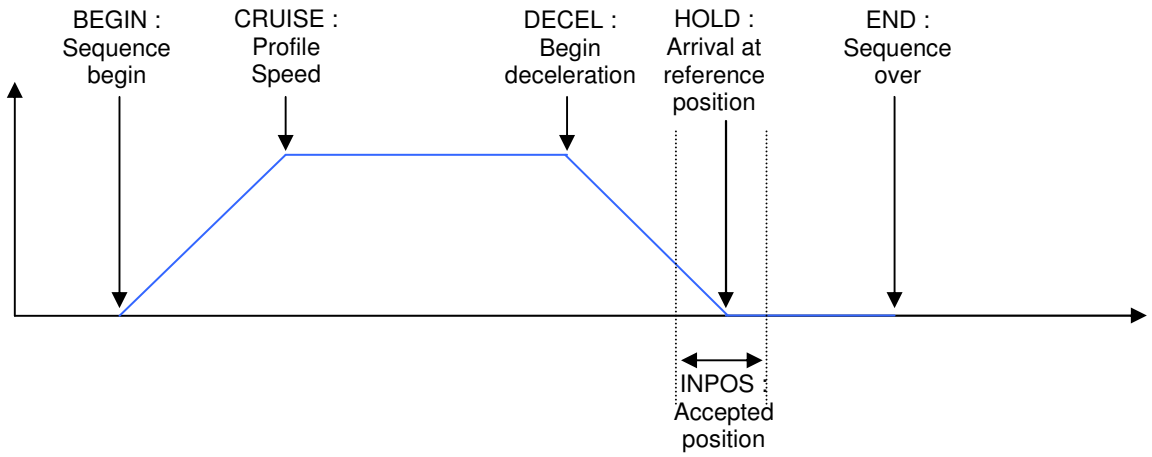
Sequence Phase

This object monitors the state inside a sequence.

Index	**0x3605**
Name	Sequence Phase
Object Code	VAR
Data Type	Unsigned16
Object Class	sq
Access	ro
PDO Mapping	Yes

Data Description

Bit Number	Function
0	begin
1	cruise
2	decel
3	hold
4	inpos
5	end



Sequence Captured Position

This object gives the value of the position captured by the torque sequence.

Index	0x360B
Name	Sequence Captured Position
Object Code	VAR
Data Type	Integer32
Object Class	sq
Access	ro
Unit	Position Unit
PDO Mapping	Yes

Supported Sequence Types

Various sequence types can be implemented in a given firmware and drive model. This object shows supported sequence types. This object is read only.

Index	0x360F
Name	Supported sequence types
Object Code	VAR
Data Type	Unsigned16
Object Class	sq
Access	ro
PDO Mapping	No
Value	See below

Data Description

Bit Number	Function
0	Positioning sequence supported
1	Homing sequence supported
2	Velocity sequence supported
3	Torque sequence supported
4	Gearbox sequence supported
5	Cam sequence supported

Maximum Sequences supported

This object gives the maximum sequences supported by a given device.
The sequence number is between 0 and maximum sequences supported - 1

Index	0x3612
Name	Maximum sequence supported
Object Code	VAR
Data Type	Unsigned16
Object Class	sq
Access	ro
PDO Mapping	No
Value	See below

Sequence Parameter Number

Index	0x3610
Name	Sequence Parameters Number
Object Code	VAR
Data Type	Integer16
Object Class	sq
Access	rw
PDO Mapping	No
Default Value	0

This parameter holds the sequence number for direct reading/writing into sequence parameters by object 0x3611.

Sequence Parameters

Index	0x3611
Name	Sequence Parameters
Object Code	RECORD
Number of Elements	26

This object allows to read/write all parameters of a sequence which number is given in object 0x3610.

Value Description

Sub Index	1
Description	Sequence Type
Data Type	Integer16
Object Class	sq
Access	rw
PDO Mapping	No
Default Value	

This parameter allows to read/write the type of a sequence. Check object 0x360F for supported sequence types.

The value is the sequence type:

Value	Function
0	Not defined
1	Positioning sequence
2	Homing sequence
3	Speed sequence
4	Torque sequence
5	Gearing sequence

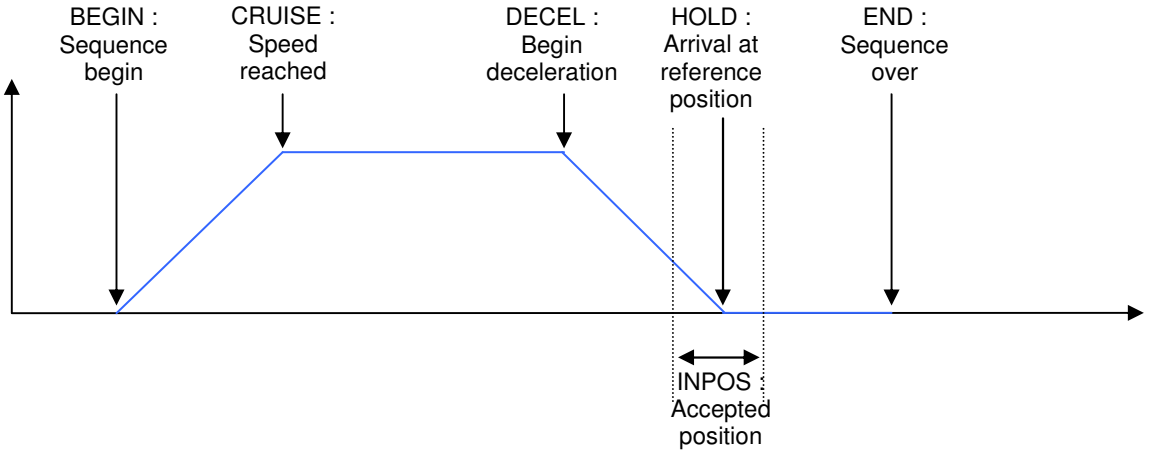
Sub Index	2
Description	Next sequence
Data Type	Integer16
Object Class	sq
Access	rw
PDO Mapping	No
Value Range	-1..127 -1 means there is no other sequence
Default Value	

Sub Index	3
Description	Sequence Counter
Data Type	Unsigned16
Object Class	sq
Access	rw
PDO Mapping	No
Value Range	
Default Value	

Sub Index	4
Description	Sequence Link
Data Type	Integer16
Object Class	sq
Access	rw
PDO Mapping	No
Value Range	-1..127
Default Value	

Sub Index	5
Description	Output Trigger
Data Type	Unsigned16
Object Class	sq
Access	rw
PDO Mapping	No
Value Range	
Default Value	

Bit Number	Function	Description
0	BEGIN	
1	CRUISE	
2	DECEL	
3	HOLD	
4	INPOS	The output is triggered according to the parameter Position Window (see 0x6067)
5	END	The output is triggered after Temporization at the end of the positioning



Sub Index	6
Description	Output Bits = 0
Data Type	Unsigned16
Object Class	sq
Access	rw
PDO Mapping	No
Value Range	
Default Value	0

Sub Index	7
Description	Output Bits = 1
Data Type	Unsigned16
Object Class	sq
Access	rw
PDO Mapping	No
Value Range	
Default Value	0

Sub Index	8
Description	Output Bits Toggle
Data Type	Unsigned16
Object Class	sq
Access	rw
PDO Mapping	No
Value Range	
Default Value	0

Sub Index	9
Description	Start Condition Bits = 0
Data Type	Unsigned16
Object Class	sq
Access	rw
PDO Mapping	No
Value Range	
Default Value	0

Sub Index	10
Description	Start Condition Bits = 1
Data Type	Unsigned16
Object Class	sq
Access	rw
PDO Mapping	No
Value Range	
Default Value	0

Sub Index	11
Description	End Condition Bits = 0
Data Type	Unsigned16
Object Class	sq
Access	rw
PDO Mapping	No
Value Range	
Default Value	0

Sub Index	12
Description	End Condition Bits = 1
Data Type	Unsigned16
Object Class	sq
Access	rw
PDO Mapping	No
Value Range	
Default Value	0

Sub Index	13
Description	Position
Data Type	Integer32
Object Class	sq
Access	rw
PDO Mapping	No
Unit	Position Unit
Value Range	

For a homing sequence, this parameter defines the home offset value.

Sub Index	14
Description	Position 2 (reserved for futur use)
Data Type	Integer32
Object Class	sq
Access	rw
PDO Mapping	No
Unit	Position Unit
Value Range	

Sub Index	15
Description	Speed
Data Type	Integer32
Object Class	sq
Access	rw
PDO Mapping	No
Unit	Speed Unit
Value Range	
Default Value	0

Sub Index	16
Description	Speed 2 / Position 3 (reserved for futur use)
Data Type	Integer32
Object Class	sq
Access	rw
PDO Mapping	No
Unit	Speed Unit / Position Unit
Value Range	
Default Value	0

Sub Index	17
Description	Acceleration
Data Type	Unsigned32
Object Class	sq
Access	rw
PDO Mapping	No
Unit	Acceleration Unit
Value Range	
Default Value	0

Sub Index	18
Description	Deceleration
Data Type	Unsigned32
Object Class	sq
Access	rw
PDO Mapping	No
Unit	Acceleration Unit
Value Range	
Default Value	0

Sub Index	19
Description	Acceleration Time
Data Type	Unsigned16
Object Class	sq
Access	rw
PDO Mapping	No
Unit	ms
Value Range	1..65535
Default Value	0

Sub Index	20
Description	Deceleration Time
Data Type	Unsigned16
Object Class	sq
Access	rw
PDO Mapping	No
Unit	ms
Value Range	0..65535
Default Value	0

Sub Index	21
Description	Configuration
Data Type	Unsigned16
Object Class	sq
Access	rw
PDO Mapping	No
Default Value	0

For a position sequence, this parameter defines the positioning type:

Value	Function
0	Absolute positioning
1	Relative positioning

For a homing sequence, this parameter defines the "return" configuration:

Value	Function
0	No return
1	Return to homing position

Sub Index	22
Description	Configuration 2
Data Type	Integer16
Object Class	sq
Access	rw
PDO Mapping	No
Default Value	0

For a homing sequence, this parameter defines the homing method.

Sub Index	23
Description	Temporization
Data Type	Unsigned16
Object Class	sq
Access	rw
PDO Mapping	No
Unit	ms
Value Range	0…16000
Default Value	0

Sub Index	24
Description	Running Time
Data Type	Unsigned16
Object Class	sq
Access	rw
PDO Mapping	No
Unit	ms
Value Range	
Default Value	0

For a speed sequence or a torque sequence, if the Running Time is 65535 (maximum of 16-bit) then the running phase will be executed forever. An "End Condition" can be used to exit this sequence.

Sub Index	25
Description	Analog In
Data Type	Integer16
Object Class	sq
Access	rw
PDO Mapping	No
Unit	16-bit scaled value
Value Range	
Default Value	0

For a torque sequence, this parameter defines the torque value.
For a homing sequence, this parameter defines the home current limit value.

Sub Index	26
Description	Analog In 2 (reserved for futur use)
Data Type	Integer16
Object Class	sq
Access	rw
PDO Mapping	No
Unit	16-bit scaled value
Value Range	
Default Value	0

3.2.3.8.8. Sequence File Format

Description

1. Sequence files are text files.
   Characters are not case sensitive.

2. Parameters syntax is:
   Key_word = value
   There must be only one key word per line

3. Parameter value can be:
   - number: decimal or hexa-decimal (preceded by 0x)
   - constant (text)

4. The character ; indicates the begin of a comment to the end of the line.

5. A sequence begins with keyword **SeqNb**

6. Parameters of a sequence are declared one after the other. Except for **SeqNb**, the parameter order has no importance.

7. There is no indication for the end of a sequence. A new sequence with SeqNb indicates the end of the current sequence.

8. Uncoherent parameters or values out of the limits will generate an error.

9. In a sequence, parameters which are not declared will have a default value.
   The default value can be changed by means of the **Default** keyword.

10. The sequencer can load sequence files in two ways:

    - LOAD: load declared sequences from the sequence file into memory. Sequences that are not declared will be cleared.
    - MERGE: load declared sequences from the sequence file into memory. Sequences that are not declared in the file will be kept.

Sequence file example:

```
; define some default values
Default
Accel=100000
Decel=100000

; sequence 1: positioning
SeqNb=1
SeqType=pos
Pos=0x001000
PosType=ABS      ; absolute positioning
Speed=100000
Output="..001000"
```

```
Trigger=begin ; activate outputs at the beginning of the sequence
Tempo=1000
SeqNext=3

; sequence 3: run at high speed during 10s
SeqNb=3
SeqType=speed
AccelTime=200000
DecelTime=200000
Speed=500000
RunTime=10000
```

Sequence Keyword
Supported sequence type:

- Positioning sequence
- Homing sequence
- Speed sequence
- Torque sequence

General Parameters
General parameters are for all sequence types.

Key word	Signification/Constance
SeqType	Sequence Type POS, SPEED, HOME, TORQUE, GEAR
SeqNext	Next sequence
SeqCount	Sequence Counter
SeqLink	Conditional Jump
Output	Output
Trigger	Output trigger BEGIN, CRUISE, DECEL, HOLD, END
StartCond	Start condition inputs
EndCond	End condition inputs

Positionning Sequence

Key word	Signification
PosType	Positioning type: ABS / REL
Pos	Positioning value
Speed	Move Speed
Speed2	End Speed
Accel	Acceleration
Decel	Deceleration
Tempo	Temporization at the end of positioning

Homing Sequence

Key word	Signification
HomeOfs	Position Offset
Speed	Speed during search for switch
Speed2	Speed during search for Zero
Accel	Acceleration
Decel	Deceleration
Method	Homing method
Torque	Torque limit for mechanical limit homing

Speed Sequence

Key word	Signification
Speed	Move Speed
AccelTime	Acceleration Time
DecelTime	Deceleration Time
RunTime	Move Time

Torque Sequence

Key word	Signification
Speed	Move Speed
Accel	Acceleration
Decel	Deceleration
RunTime	Torque limit Time
Torque	Torque limit

3.2.3.9 - Stepper Emulation Mode

Stepper Emulation Mode

The Stepper emulation mode emulates the behavious of a stepper motor and drive.

The position reference is given by PULSE input and DIR input: when pulse following control is enabled in the control word, the servo motor position setpoint is received via the PULSE and DIR input pins.

The stepper motor emulation application is only possible for motors equipped with a resolver as a position feedback sensor, the encoder input is used for pulse/dir command input (the encoder input must be selected with incremental TTL encoder).

When the amplifier is switched on with the stepper emulation mode selected, Pulse following control is disabled. In this case, the input pulses are not counted and the motor is enabled at standstill.
The motor starts following the input pulses when PULSE_ENA (in control word) is set or COUNT_ENA (in 0x3681-3) is set.

The specific bits of the control word (object 0x6040) used in stepper emulation mode are described below:

Bit	Name	Function
4	PULSE_ENA	Enable pulse following
5		reserved
6		reserved

The specific bits of the status word (object 0x6041) used in stepper emulation mode are described below:

Bit	Name	Function
12	PULSE_OK	Pulse following ok
13	PULSE_CNT	Pulse Count

The PULSE_OK is set when drive is enabled and PULSE_ENA or COUNT_ENA is set.
The PULSE_CNT is active only with PULSE_OK active. The PULSE_CNT signal is described in object 0x3681.

The motor Maximum speed value is calculated according to the host controller pulse frequency limit as follows : Maximum speed (rpm) = 60 x pulse frequency limit (Hz) / Stepper resolution. For simple count configuration (object 0x3681-3 bit 7 = 0), the Stepper resolution = User position scaling (object 0x6093-2). For double count configuration (object 0x3681-3 bit 7 = 1), the Stepper resolution = User position scaling (object 0x6093-2) / 2.
The Max Motor Speed parameter (object 0x6080) is set to the previously calculated maximum speed value + 10% to avoid amplifier speed saturation.

The motor speed depends on the pulse frequency and the parameter User position scaling (object 0x6093-2).

The motor displacement direction with regard to the DIR input logic state can be configured by using the reverse bit of resolver input.

The polarity of the PULSE and DIR inputs is configurable by 0x3681

Object Dictionary Entries

Index	Object	Name	Type	Attr.
0x3681	VAR	Stepper Emulation Configuration	ARRAY	rw
0x3685	VAR	Pulse following counter	Integer32	ro
0x3686	VAR	Position Set Point	Integer32	ro

Index	0x3681
Name	Stepper Emulation Configuration
Object Code	RECORD
Object Class	se
Number of Elements	4

This object allows to setup the stepper emulation mode parameters.

Value Description

Sub Index	1
Description	Stepper control reserved for futur used
Data Type	Unsigned16
Access	rw
PDO Mapping	Yes

Sub Index	2
Description	Stepper status
Data Type	Unsigned16
Access	ro
PDO Mapping	Yes

Bit	Name	Description
0	PULSE_OK	Pulse following ok
1	PULSE_CNT	Pulse Count

These 2 bits are exactly the same as bits 12 and 13 in the status word.

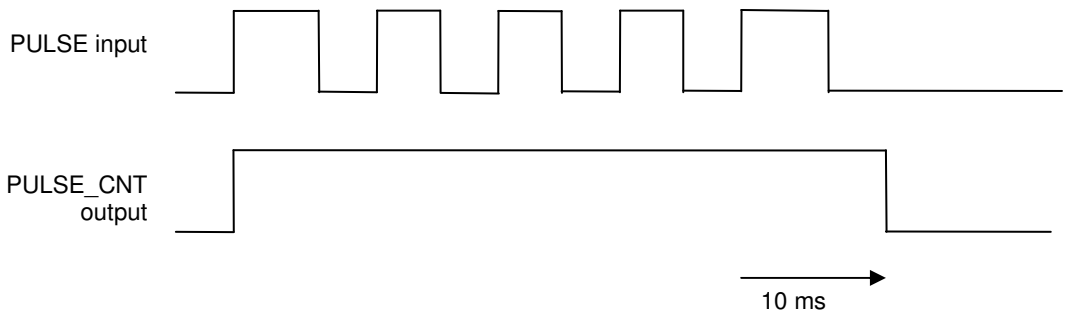
Sub Index	3
Description	Inputs Configuration
Data Type	Unsigned16
Access	rw
PDO Mapping	No
Value	This parameter can only be changed when drive is disabled.

Bit	Name	Description
0-1	SELECT_INP	Inputs Selection: 0 Inputs from Encoder connector (differential line driver inputs) A+/A- -> A or PULSE B+/B- -> B or DIR 1 Inputs from I/O connector (logic opto-coupler inputs) IN5 -> A or PULSE IN3 -> B or DIR 2 Inputs from Hall Effect Sensor (Encoder connector) Hall U -> A or PULSE Hall V -> B or DIR
2		Reserved. Must be 0.
3	CNT_MODE	Inputs Count Mode: 0 Quadrature (A/B) inputs 1 Pulse/Dir inputs
4	PULSE_POL	PULSE or A polarity
5	DIR_POL	DIR or B polarity
6		Reserved. Must be 0.
7	PULSE_DBL	Pulse Count Mode (only for PULSE/DIR input) 0 Simple count. Count on raising edge of PULSE Motor Speed (rpm)=60 * Pulse_Frequency / User position scaling (0x6093-2) 1 Double count. Count on raising edge and falling edge of PULSE Motor Speed (rpm)=120 * Pulse_Frequency / User position scaling (0x6093-2)
8	COUNT_ENA	Count Enable
9-15		Reserved. Must be 0.

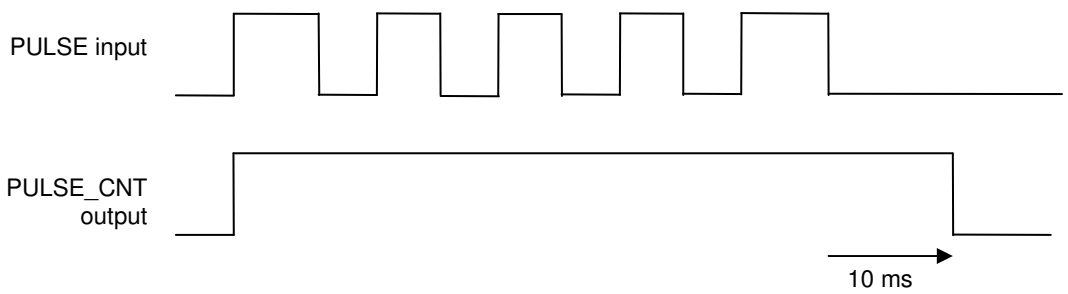
Sub Index	4
Description	PULSE_CNT timeout
Data Type	Unsigned16
Access	rw
PDO Mapping	No
Unit	ms
Default value	10

This parameter defines the timeout after the last pulse the signal PULSE_DIR will be reset.

Simple count:



Double count:



Example Stepper Emulation Configuration

1. Selection of Stepper Emulation Mode
- Mode of operation 0x6060,0 = -2 (this can be done in Gem Drive Studio)

2. Activate PULSE/DIR inputs:
- Enable TTL incremental encoder input
- Disable encoder error control: 0x3025,1 = 0x00240000
- Setup the PULSE/DIR with 0x3681,3 = 0x013A (Inputs from Hall Effect Sensor lines on the encoder connector)
- Set the User position scaling parameter (0x6093-2) according to the desired motor speed
  Motor Speed (rpm)=60 * Pulse_Frequency / User position scaling (0x6093-2) for simple count selection
  Motor Speed (rpm)=120 * Pulse_Frequency / User position scaling (0x6093-2) for double count selection

3. Setup the PULSE_CNT output:
- PULSE_CNT timing: 0x3681,4 = 10
- Connect PULSE_CNT signal to logic output OUT3
  0x3504,3 = 0x36810201 (this can be done in Gem Drive Studio)

4. Autotuning must be executed with "minimum position overshot".
After autotuning, the term Kav Feedforward acceleration Gain (0x60FB,4) must be reset to 0.

3.2.3.10 - Analog Speed Mode

Analog Speed Mode

In this mode, the ServoPac drive operates as a variable speed drive.

The speed reference is the analog input 1.

The maximum speed defined by 0x6080 is reached with 10V input.

The acceleration time from 0 to maximum speed and the deceleration time from maximum speed to 0 are defined in ms by object 0x604F.

The deceleration time is also defined in ms by object 0x304F. This allows to set a deceleration time different from the acceleration time.

Operation Mode number: -1 (0x6060)

If HALT bit in control word (0x6040) is set, the speed reference is reset to 0.

Index	**0x604F**
Name	Velocity Ramp
Object Code	VAR
Data Type	Unsigned32
Object Class	as
Access	rw
PDO Mapping	No
Unit	ms
Value Range	0 - 0x3FFF80
Default Value	0

This object define the acceleration time from 0 to maximum motor speed defined in 0x6080, and the deceleration time from maximum motor speed to 0.

Index	0x304F
Name	Velocity Ramp 2
Object Code	VAR
Data Type	Unsigned32
Object Class	as
Access	rw
PDO Mapping	No
Unit	ms
Value Range	0 - 0x3FFF80
Default Value	0

This object define the deceleration time from maximum motor speed to 0.

3.2.3.11 - Analog Torque Mode

Analog Torque Mode

In this mode, the ServoPac drive operates in current loop with current reference from analog input 1.

The Analog Input value is given by:
Analog_Input_1 = (ADC - AnalogIn1Offset) * AnalogIn1Gain / 256

ADC value = 0x7FF0 for 10V
AnalogIn1Offset is offset of analog input and is defined by object 0x30F1,3
AnalogIn1Gain is defined by object 0x30F1,4

The current reference is set with Analog_Input_1. A value of 0x7FFF correspond to amplifier size (0x6510,1)

If HALT bit in control word (0x6040) is set, then the current reference is reset to 0.

The object 0x3077,0 allows to define a window in which the status bit Target_Reached is set.

Analog torque operation mode selection code: -5 (0x6060)

Index	0x3077
Name	Torque Threshold
Object Code	VAR
Data Type	Integer16
Object Class	at
Access	rw
PDO Mapping	No
Unit	0x7FFF -> drive amplifier size current
Default Value	0

3.2.3.12 - Gearing Mode

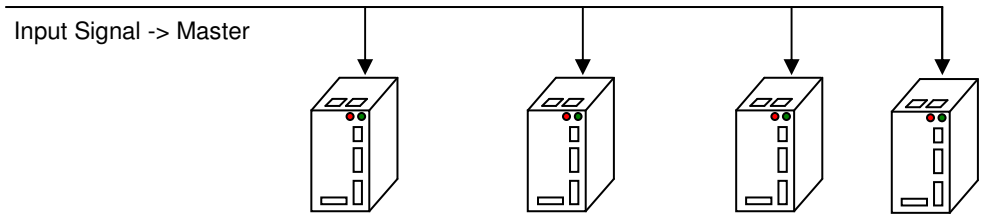
3.2.4. MASTER-SLAVE FUNCTIONS

3.2.4.1 - Master-Slave

Master-Slave Function

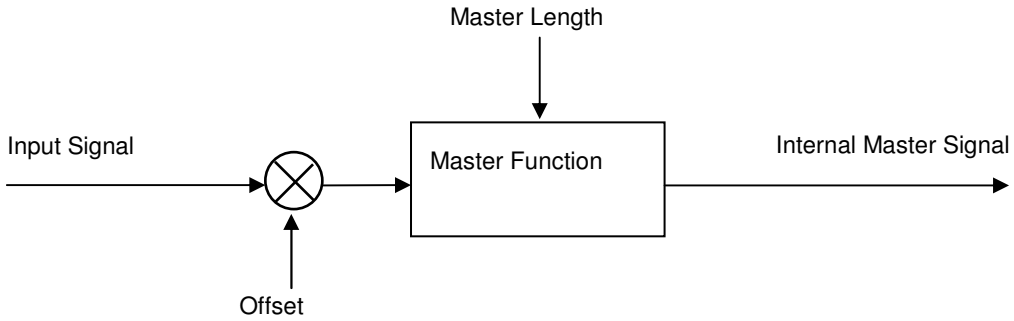
ServoPac drives support master-slave relationship in which several drives run as a slave following a master. One drive can be used as a master and the master's position reference can be distributed to all other drives via encoder output or CAN bus (EtherCAT can not be used).

A virtual master signal can be generated by a drive and distributed to all other drives via CAN bus, including the one with virtual master (which runs as a slave).



Master Input Function

In each drive running as slave, a master input function block allows to define the master input source, offset and possibly apply a modulo operation on master signal. The result signal "Internal Master Signal" is applied to the master-slave function block.



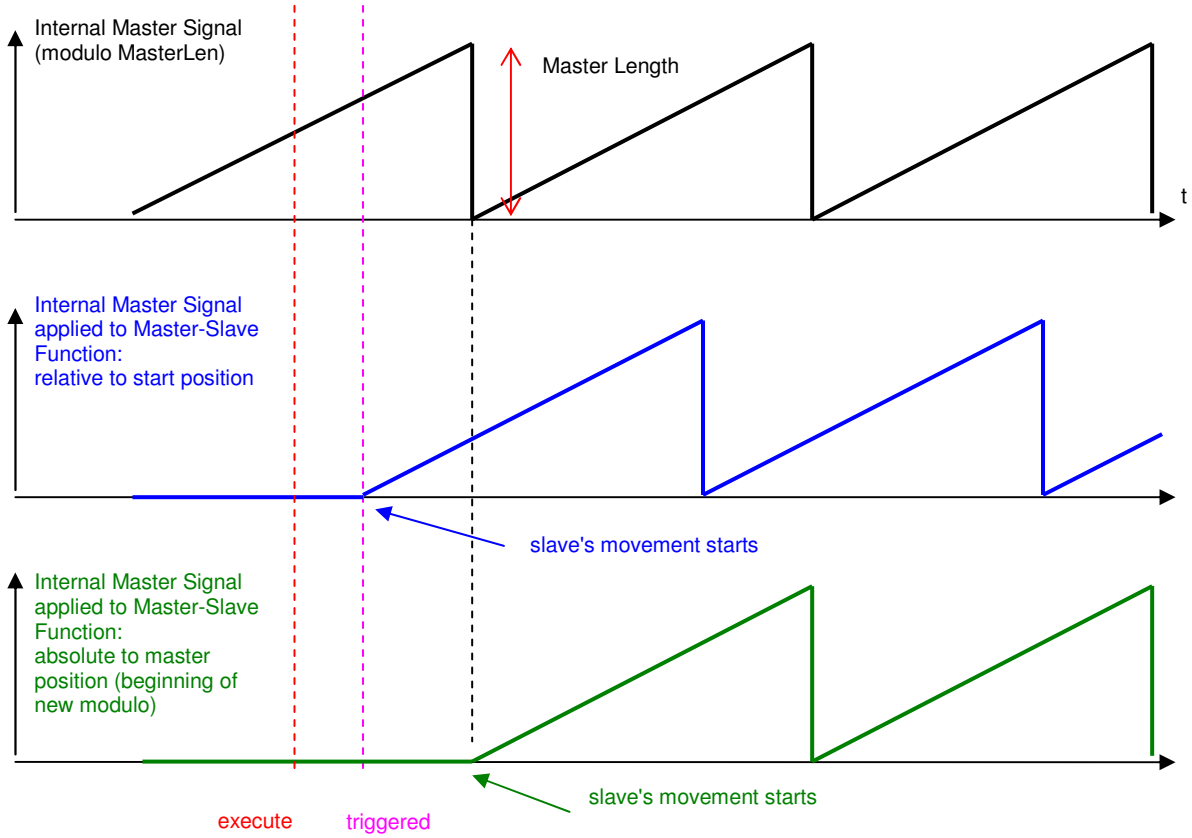
The input signal comes from:
- Encoder signal: from a true encoder or an encoder emulation output from one drive.
- Fieldbus signal: from fieldbus master or from one drive (with CAN bus).

The Internal Master Signal is used as master position for all master/slave functions.

Starting Master-Slave Operation
Master Length allows to apply a modulo operation on input signal. If Master Length = 0 then modulo operation is disabled.

Trigger Mode: defines how to start slave operation.
When the master-slave function is executed, the slave's movement can be triggered:
- immediately as soon as the master-slave function is executed,
- with raising edge on logic inputs.

Start Mode: defines where to start the slave's movement.
When slave is triggered, the slave's movement is started with the master position
- relative to the start position or
- absolute to the master position (only with modulo).



Note: The diagram above is shown with a master running at constant speed.

Master-Slave supported Functions
Depending on the device model, supported master-slave functions are:
- Master-Slave Gearbox
- Master-Slave Cam

Master Input Object Definition

Index	Object	Name	Type	Attr.
0x3910	VAR	Master Inputs Source	Unsigned32	rw
0x3911	VAR	Master Length	Integer32	rw
0x3915	VAR	Trigger Inputs	Unsigned16	rw
0x3916	VAR	Internal Master signal	Integer32	rw
0x3940	VAR	MS_Control	Unsigned16	rw
0x3941	VAR	MS_Status	Unsigned16	rw
0x3942	VAR	Jog Speed	Integer32	rw

Master Input Source

Index	0x3910
Name	Master Input Source
Description	Index/sub-index of input data
Data Type	Unsigned32
Class	sq, gb
Access	rw
PDO Mapping	No
Value	See below
Default Value	0

This object allows to connect any 32-bit dataflow as master input signal.

The structure of the entries is the following:

MSB		LSB
Index (16-bit)	Sub-index (8-bit)	0

Example:
0x3910,0 = 0x31290000
connects the encoder Input as master input signal.

0x3910,0 = 0x39060000
connects the virtual master as master input signal.

0x3910,0 = 0x30C10000
connects the interpolated signal from CAN bus as master input signal.

Index	0x3911
Name	Master Length
Object Code	VAR
Data Type	Unsigned32
Object Class	gb sq
Access	rw
PDO Mapping	No
Unit	master inc
Default Value	0

This object defines the modulo value for the master input.

Index	0x3915
Name	Master-Slave Trigger Inputs
Object Code	VAR
Data Type	Unsigned16
Object Class	gb sq
Access	rw
PDO Mapping	No
Default Value	0

This object defines the bit pattern that will trigger the master-slave functions.
Each bit (0..15) corresponds to a physical input in 0x60FD,0 (bit 16..31).

Index	0x3916
Name	Internal Master Signal
Object Code	VAR
Data Type	Integer32
Object Class	all
Access	ro
PDO Mapping	Yes
Unit	position unit
Default Value	-

This object gives the value of the internal master value (modulo) which will be applied to any supported master-slave functions.

Master-Slave Control

Index	0x3940
Name	Master-Slave Control
Object Code	VAR
Data Type	Unsigned16
Object Class	gb, cm, sq
Access	rw
PDO Mapping	Possible
Default Value	0000

This parameter allows to modify some behaviour of the slave on-the-fly.

Bit Number	Function
0,1	reserved
2	Slave Phase Shift Start
3	reserved
4	Jog+
5	Jog-
6..7	reserved
8	For gearing function: 0 gearing ratio set 1 1 gearing ratio set 2 This bit has effect only if gearbox config bit 9 is set. Gearbox config is defined by 0x3928,1 for gearing mode or defined directly in sequence parameters for gearing sequence. This bit allows to select and change-on-the fly the ratio set. For camming function: 0 cam profile 0 1 cam profile 1 This bit has effect only if cam config bit 9 is set. Cam config is defined by 0x3938,1 for caming mode or defined directly in sequence parameters for caming sequence. This bit allows to select and change-on-the fly the cam profile.
9	For gearing function: On raising edge, new value of gearing ratio (defined in 0x3921) are applied. This bit allows to change on the fly gearing ratio value. For camming function:
10..15	reserved

Master-Slave Status

Index	0x3941
Name	Master-Slave Status
Object Code	VAR
Data Type	Unsigned16
Object Class	gb, cm, sq
Access	ro
PDO Mapping	Possible
Default Value	0000

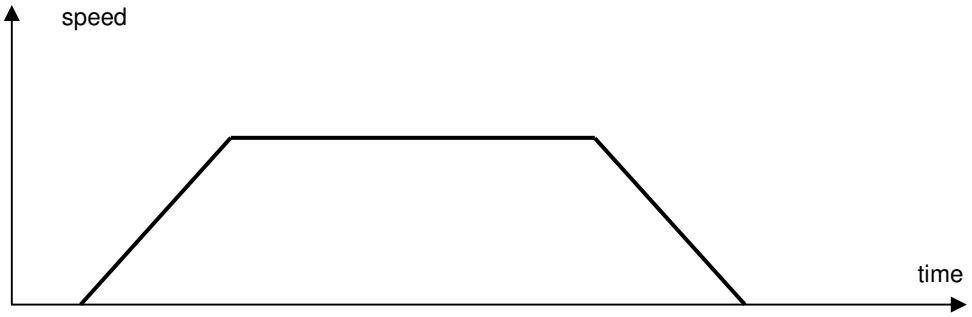
Bit Number	Function
0	Master-Slave function execute command acknowledged
1	Master-Slave function is triggered
2	Master-Slave function is running
3	Master-Slave function is stopping
4	Jog running
5	Error
6	Master Length Change
7	Slave Length Change
8	In Gear
9	Synchronization position
10	Slave Phase Shift running
11	reserved
12	Gearing function: ratio set 0 ratio set 1 selected 1 ratio set 2 selected Caming function: 0 profile 0 selected 1 profile 1 selected
13	Ratio/Profile changing
14	Gearing scaling
15	reserved

Index	0x3942
Name	Slave Jog Speed defines the slave's Jog speed parameter.
Object Code	VAR
Data Type	Integer32
Object Class	gb
Access	rw
PDO Mapping	No
Unit	User Velocity Unit
Value Range	-
Default Value	-

3.2.4.2 - Virtual Master

Virtual Master

Virtual master allows to generate a position signal with a profile as below:



The profile starts as soon as Virtual Master Speed is set with the defined acceleration, and the master signal slows down and stops when Virtual Master Speed is reset at 0. The virtual master output object can be used as master-slave input signal or distributed over CAN bus via PDO.

Virtual Master Objects Definition

Index	Object	Name	Type	Attr.
0x3901	VAR	Virtual Master Speed Source	Unsigned32	rw
0x3902	VAR	Virtual Master Speed	Integer32	rw
0x3903	VAR	Virtual Master Acceleration	Integer32	rw
0x3906	VAR	Virtual Master Output	Integer32	rw

Virtual Master Speed Source

Index	0x3901
Name	Virtual Master Speed Source
Description	Index/sub-index of input data
Data Type	Unsigned32
Class	sq, gb, cm
Access	rw
PDO Mapping	No
Value	See below
Default Value	0

This object allows to connect any 32-bit dataflow as virtual master speed signal. The content of virtual master speed (0x3902,1) will be replaced by the content of the source.

The structure of the entries is the following:

MSB		LSB
Index (16-bit)	Sub-index (8-bit)	0

Example:
0x3901,0 = 0x30F10200
connects the analog input 1 (32-bit format) to virtual master speed input (0x3902,0).

Index	0x3902
Name	Virtual Master Speed defines the target for the virtual master.
Object Code	VAR
Data Type	Integer32
Object Class	all
Access	rw
PDO Mapping	No
Unit	User Velocity Unit
Default Value	0

Index	0x3903
Name	Virtual Master Acceleration defines the acceleration for virtual master
Object Code	VAR
Data Type	Unsigned32
Object Class	-
Access	rw
PDO Mapping	No
Unit	User acceleration unit
Default Value	-

Index	0x3906
Name	Virtual Master Output
Object Code	VAR
Data Type	Integer32
Object Class	-
Access	ro
PDO Mapping	Yes
Unit	position unit
Default Value	-

This object gives the output value of the virtual master. This value
- can be applied directly to master-slave functions,
- or sent to other drives via CANbus. In this case, the virtual master output must also be sent to this drive itself via CAN bus to avoid delay between this drive and all other drives.

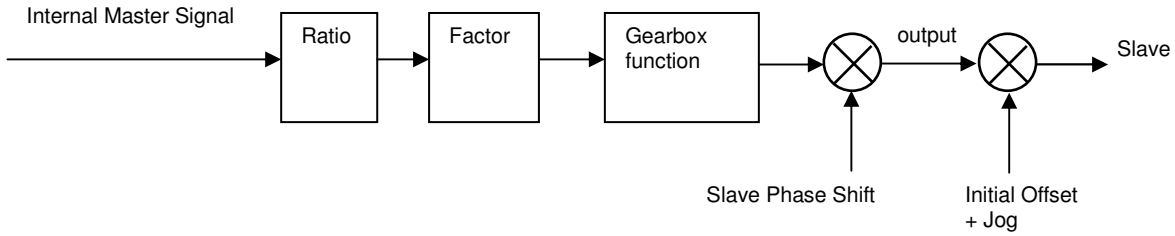
3.2.4.3 - Gearbox Function

The ServoPac drive's gearbox function allows to synchronize the slave movement with a master signal in a relationship defined by a ratio:

Slave_Pos = Numerator/Denominator * Master_Pos + Offset

The gearbox function can be used:
- in a specific mode of operation (manufacturer mode of operation)
- or as a sequence inside the sequencer mode.

Structure



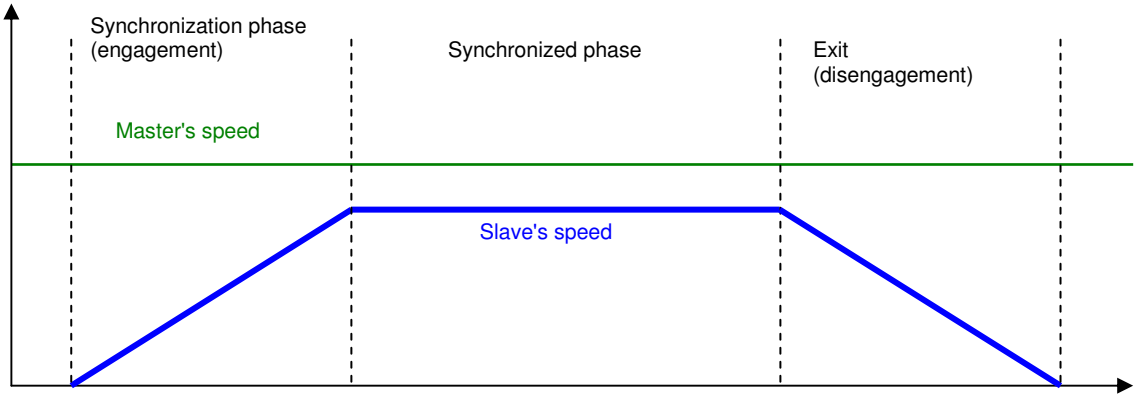
When the gearbox function is starting, the slave ramps up to the ratio of the master speed according to the gearing ratio value, and locks in when this is done. An adjustment on the slave's movement allows to adjust motor position. at the end of adjustment, the motor position is locked in frequency and phase with the master position according to the gearing ratio value.

The Gearing ratio is defined by a numerator / denumerator.
During a gearing sequence execution, this value can be multiplied by the sequence gearing ratio factor.

In gearing mode when gearbox function is not active, a jog input can be used to manually move the slave.

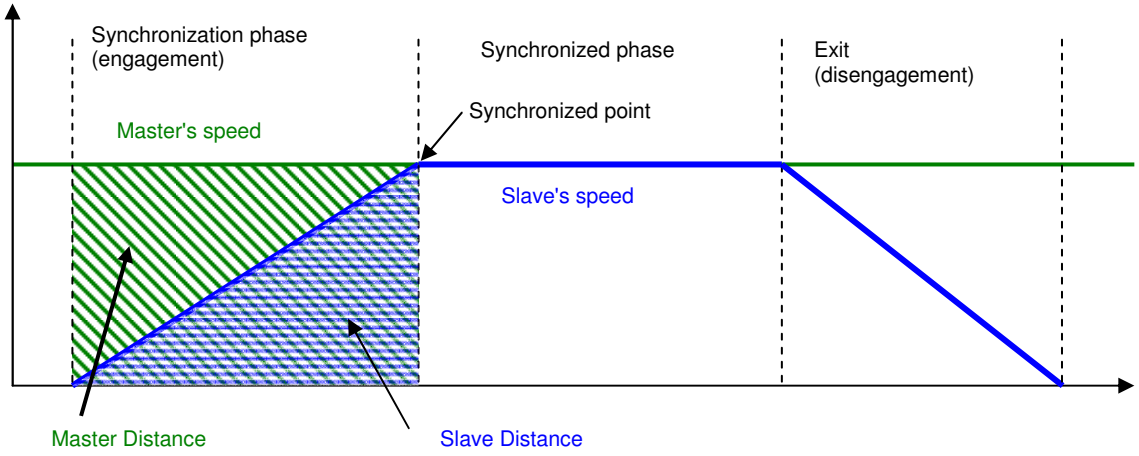
Coupling Mode

1. Constant acceleration



The slave ramps up to the ratio of the master velocity and locks in when the master's speed is reached. Any lost distance during synchronization is not caught up.

2. Constant Distance

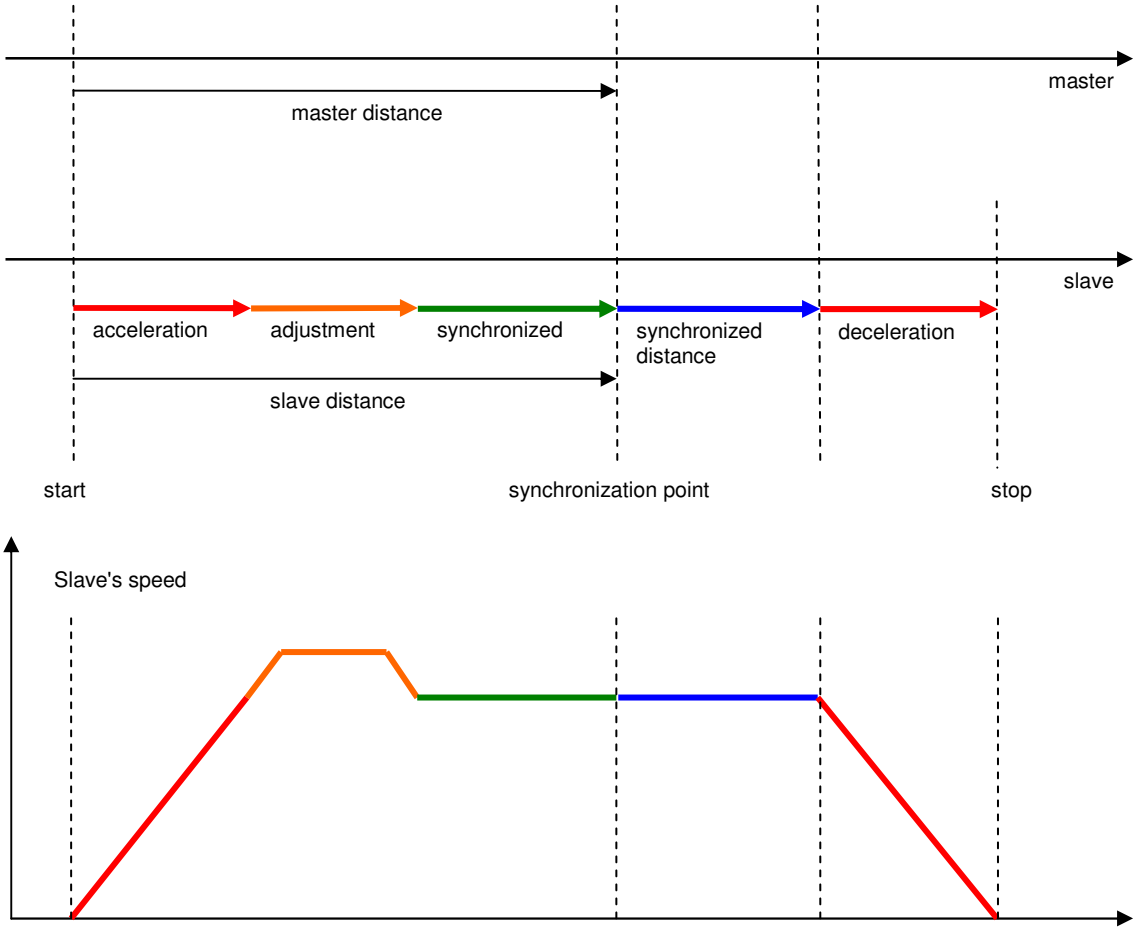


In this coupling mode, the slave accelerates to reach the master's speed so that at the synchronized point, the slave has moved the "slave distance" and the master has moved the "master distance".

So, if the master changes speed during the acceleration phase, the slave speed will change as well.
This coupling mode is for a master-slave ratio = 1, and the master distance = 2 x slave distance.

3. Constant acceleration with phase correction

In this coupling mode, the slave will accelerate at constant acceleration to reach the ratio of the master's speed and then a slave phase shift adjustment will be automatically performed to adjust the slave phase to the master phase with parameters defined in Master Distance and Slave Distance.



After adjustment, the slave is synchronized in the same phase with the master. The master distance and the slave distance must be defined in so that the slave is synchronized before the synchronization point.

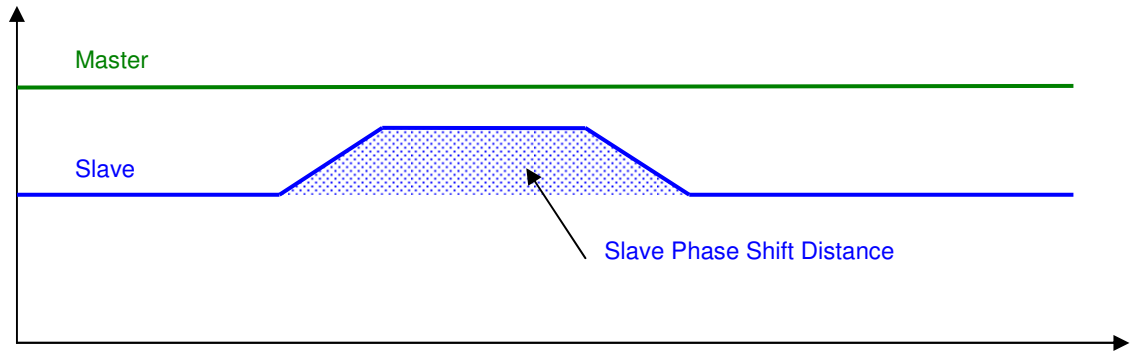
Gearbox function parameters:

Master input	defines master inputs source: encoder input, Fieldbus signal.
Ratio select	2 sets of gearing ratio (numerator and denominator) can be pre-defined. Ratio Select allows to select the desired ratio set. Ratio selection and ratio value can be modified on-the-fly.
Ratio factor	A ratio factor (scaling factor) can also be applied for a continuous modified ratio
Accel / decel	defines acceleration/deceleration values for slave ramp up/down and slave phase shift adjustment.
Differential speed	defines the relative speed for adjustment phase.
Master distance	defines the distance for the master from the start to the synchronization point.
Slave distance	defines the distance for the slave from the start to the synchronization point. If the slave distance is 0, then adjustment is not executed.

Synchronization distance	defines the distance for the slave that the slave must be synchronized in position. If the synchronization distance is 0, then the slave will synchronize with the master indefinitely. An exit command can be used to exit the gearing function.
Trigger Mode	- immediately executed command - raising edge of a logic input
Start Mode	- relative to initial position - raising edge of a logic input
Exit Mode	- exit gearbox immediately - slowdown before exit
Coupling Mode	- with constant acceleration - with constant distance - with constant acceleration and phase correction

Slave Phase Shift Adjustment

The slave phase shift allows to adjust the slave's phase to the master's phase: a small displacement can be superimposed on current slave's movement.
- the slave phase shift adjustment can be automatically executed with coupling mode = "constant acceleration with phase correction",
- or the slave phase shift Adjustment can be manually executed.



Gearbox Function Objects

Index	Object	Name	Type	Attr.
0x3920	VAR	Gearbox Ratio Modulation Source	Unsigned32	rw
0x3921	ARRAY	Gearbox Ratios	Integer32	rw
0x3925	ARRAY	Gearbox Distances	Integer32	rw
0x3926	VAR	Gearbox Output	Integer32	rw
0x3928	ARRAY	Gearbox Configuration/Factor	Unsigned16	rw
0x392A	VAR	Slave Acceleration	Integer32	rw
0x392C	VAR	Slave Differential Speed	Integer32	rw
0x392D	VAR	Slave Phase Shift Distance	Integer32	rw
0x392F	ARRAY	Gearbox Monitoring	Integer32	ro

Gearbox Ratio Modulation Source

Index	**0x3920**
Name	Gearbox Ratio Source
Description	Index/sub-index of input data
Data Type	Unsigned32
Class	sq, gb
Access	rw
PDO Mapping	No
Value	See below
Default Value	0x60FF0000

This object allows to connect any 32-bit dataflow as a modulation signal applied on the current ratio set.

The structure of the entries is the following:

MSB		LSB
Index (16-bit)	Sub-index (8-bit)	0

Example:
0x3920,0 = 0x30F10200
connects the analog input 1 as the ratio modulation signal.

Gearbox Ratio Parameters

Index	**0x3921**
Name	Gearbox Ratio Parameters
Object Code	ARRAY
Number of Elements	4

Value Description

Sub Index	1
Description	Gearbox ratio set 1 numerator
Data Type	Unsigned32
Access	rw
PDO Mapping	No
Range	1 to 65536

Sub Index	2
Description	Gearbox ratio set 1 denominator
Data Type	Unsigned32
Access	rw
PDO Mapping	No
Range	1 to 65536

Gearing ratio 1 value = Numerator1 / Denominator1, Gearing ratio range is from 1/256 to 256.

Sub Index	3
Description	Gearbox ratio set 2 numerator
Data Type	Unsigned32
Access	rw
PDO Mapping	No
Range	1 to 65536

Sub Index	4
Description	Gearbox ratio set 2 denominator
Data Type	Unsigned32
Access	rw
PDO Mapping	No
Range	1 to 65536

Gearing ratio 2 value = Numerator2 / Denominator2, Gearing ratio range is from 1/256 to 256.

Gearbox Distances Parameter

This object defines Master Distance, Slave Distance and Synchronization Distance for the gearbox mode.
For gearbox sequence in sequence mode, there are 3 equivalent parameters defined in each sequence.

Index	**0x3925**
Name	Gearbox Distances Parameter
Object Code	ARRAY
Object Class	sq, gb
Number of Elements	3

Value Description

Sub Index	1
Description	Master Distance
Data Type	Integer32
Access	rw
PDO Mapping	No
Unit	Position Unit

Sub Index	2
Description	Slave Distance
Data Type	Integer32
Access	rw
PDO Mapping	No
Unit	Position Unit

Sub Index	3
Description	Synchronization Distance
Data Type	Integer32
Access	rw
PDO Mapping	No
Unit	Position Unit

Index	**0x3926**
Name	Gearbox output
Object Code	VAR
Data Type	Integer32
Object Class	all
Access	ro
PDO Mapping	Yes
Unit	position unit
Default Value	-

Gearbox Configuration/Factor

Index	0x3928
Name	Gearbox Configuration/Factor
Object Code	ARRAY
Class	gb
Number of Elements	2

Value Description

Sub Index	1
Description	Gearbox Configuration
Data Type	Unsigned16
Access	rw
PDO Mapping	No

This parameter allows to define the behaviour of the gearbox function for gearbox mode.
In gearbox mode, object 0x3928,1 is taken into account when the gearbox is activated. It cannot be modify when the gearbox is running.
In a gearbox sequence, an equivalence parameter is defined in sequence parameter with difference value for each gearbox sequence.

Bit Number	Function
0..1	Exit Mode: 0 slowdown 1 immediate Exit Mode immediate allows to exit the gearbox sequence and link to another sequence without stop motor.
2..3	Coupling Mode: 0 with speed ramp (constance acceleration) 1 with constant distance (variable acceleration) 2 with speed ramp and position correction
4..5	Trigger Mode: 0 immediate 1 on logic input raising edge
6..7	Start Mode: 0 relative to initial position 1 absolute to master position (modulo)
8	Ratio set select 0 ratio set 1 1 ratio set 2
9	Allows Ratio Set Selection by master-slave control (0x3940,0)
10	Ratio modulation Enable
11	reserved
12	Forward only: prevent slave moves in negative direction. If master goes in negative direction, an error will be notified in master-slave status (0x3941,0)
13..15	reserved

Sub Index	2
Description	Gearbox Ratio Factor
Data Type	Unsigned16
Access	rw
PDO Mapping	Possible

This parameter defines a ratio factor applied to the master-slave ratio,
Factor = Gearbox_Ratio_Factor / 32768
If Gearbox_Ratio_Factor = 0, no ratio Factor is applied.

In a gearing sequence, an equivalence factor is defined in sequence parameter with difference value for each gearing sequence.

Index	0x392A
Name	Gearbox Slave Acceleration
Object Code	VAR
Data Type	Unsigned32
Object Class	gb sq
Access	rw
PDO Mapping	No
Unit	User acceleration unit
Value Range	-
Default Value	-

This object defines the acceleration used in gearbox:
- when the slave accelerates to catch-up the master's speed,
- when the slave phase shift is activated.

Index	0x392C
Name	Gearbox Slave Differential Speed
Object Code	VAR
Data Type	Unsigned32
Object Class	gb
Access	rw
PDO Mapping	No
Unit	User Velocity Unit
Value Range	-
Default Value	-

This object defines the slave's differential speed parameter for the slave phase shift function.

Index	0x392D
Name	Gearbox Slave Phase Shift Distance
Object Code	VAR
Data Type	Integer32
Object Class	gb
Access	rw
PDO Mapping	No
Unit	User Position Unit
Value Range	-
Default Value	-

This object defines the slave's distance parameter for the slave phase shift function.

3.2.5. APPLICATION FEATURE

3.2.5.1 - Digital Input/Output configuration

Digital Inputs / Outputs

The ServoPac drive allows:

- to connect any physical logic input to any bit in any variable,
- to connect any bit in any variable to any physical logic output.

The available logic input functions are:

- Negative Limit Switch
- Positive Limit Switch
- Homing Switch
- Inhibit

Index	Object	Name	Type	Attr.
0x60FD	VAR	ServoPac-B Digital Inputs	Unsigned32	ro
0x60FD	VAR	ServoPac Digital Inputs	Unsigned32	ro
0x3050	ARRAY	Digital Inputs Configuration	Unsigned32	rw
0x3051	VAR	Digital Inputs Polarity	Unsigned32	rw
0x60FE	ARRAY	ServoPac-B Digital Outputs	Unsigned32	rw
0x60FE	ARRAY	ServoPac Digital Outputs	Unsigned32	rw
0x3054	ARRAY	Digital Outputs Configuration	Unsigned32	rw
0x3055	VAR	Digital Outputs Polarity	Unsigned32	rw

Example: realize an ENABLE input with physical input IN1.
- Drive can move only when 24 V supply is applied,
- When 24V is lost, drive must stop.
So, input IN1 must be connected to the "Inhibit" function with 0x3050. When the "Inhibit" function is activated with logic level 1, the input polarity of IN1 must be reversed by object 0x3051.

Digital Inputs

Index	0x60FD
Name	Digital Inputs
Object Code	VAR
Data Type	Unsigned32
Object Class	all
Access	rw
PDO Mapping	Possible
Default Value	No

bit	Function
0	Logic Input Negative Limit Switch Function: 0 running 1 stopped in negative direction
1	Logic Input Positive Limit Switch Function: 0 running 1 stopped in positive direction
2	Logic Input HOME 0 - 1 Home switch activated
3	Logic Input INHIBIT 0 - 1 drive is disabled
12	Logic Input RESET ↑ fault reset
13	Logic Input ENABLE 0 drive is disabled ↑ drive is enabled
14	Logic Input Motor Phasing ↑ start motor phasing
16	Physical intput **IN1**
17	Physical intput **IN2**
18	Physical intput **IN3**
19	Physical intput **IN4**
20	Physical intput **IN5**
21	
22	
23	
24	
25	
26	
27	
28	
29	
30	Encoder Virtual Top Z (defined by 0x3127)
31	Resolver Virtual Top Z (defined by 0x3107)

Digital Inputs Configuration

Index	0x3050
Name	Digital Inputs Configuration
Object Code	ARRAY
Number of Elements	8

The digital Inputs configuration allows to affect any digital input to one bit in a variable indicated by index and sub-index.

Value Description

Sub Index	1-8
Description	Digital Inputs Destination defines the destination object for the corresponding digital input.
Data Type	Unsigned32
Access	rw
PDO Mapping	No
Value Range	
Default Value	

The structure of the entries is the following:

MSB		LSB
Index (16-bit)	Sub-index (8-bit)	Bit number n (0-15)

The state of the physical input will be copied into bit n of the object indicated by index and sub-index.

Digital Inputs Polarity

Index	**0x3051**
Name	Digital Inputs Polarity
Object Code	VAR
Data Type	Unsigned16
Object Class	all
Access	rw
PDO Mapping	Possible
Default Value	No

bit	**Function**
0	input **IN1**
1	input **IN2**
2	input **IN3**
3	input **IN4**
4	input **IN5**
5	
6	
7	
8	
9	
10	
11	
12	
13	
14	
15	

Digital Outputs

Index	**0x60FE**
Name	Digital Output
Object Code	ARRAY
Number of Elements	2

Value Description

Sub Index	1
Description	Digital Output
Data Type	Unsigned32
Access	rw
PDO Mapping	Possible
Default Value	0

bit	Function
0	Motor Brake
1	
2	
3	
14	
15	
16	**OUT1** Physical Output 1
17	**OUT2** Physical Output 2
18	**OUT3** Physical Output 3
19	
20	
21	
22	
23	
24	
25	
26	
27	
28	
29	
30	
31	

Sub Index	2
Description	Digital Output Bitmask
Data Type	Unsigned32
Access	rw
PDO Mapping	No
Default Value	0

If the Digital Output Bitmask corresponding to "Motor Brake" (bit 0) is set, the sub 1 allows to control the motor brake manually. Otherwise, the motor brake is automatically controlled when the drive is enabled/disabled with a delay.

Digital Outputs Configuration

Index	**0x3054**
Name	Digital Outputs Configuration
Object Code	ARRAY
Number of Elements	4

The digital outputs configuration allows to affect one bit of any variable indicated by the index and sub-index to a physical output.

Value Description

Sub Index	1-4
Description	Digital Output Source defines the source for digital output.
Data Type	Unsigned32
Access	rw
PDO Mapping	No
Value Range	
Default Value	

The structure of the entries is the following:

MSB		LSB
Index (16-bit)	Sub-index (8-bit)	Bit number n (0-31)

The state of bit n of the object index and sub-index will be copied to the physical output.

Digital Outputs Polarity

Index	0x3055
Name	Digital Outputs Polarity
Object Code	VAR
Data Type	Unsigned16
Object Class	all
Access	rw
PDO Mapping	Possible
Default Value	No

bit	Function
0	Physical Output 1
1	Physical Output 2
2	Physical Output 3
3	
4	
5	
6	
7	
8	
9	
10	
11	
12	
13	
14	
15	

3.2.5.2 - Analog Inputs/Output

The ServoPac servo drives have 2 analog inputs.

Index	Object	Name	Type	Attr.
0x30F1	RECORD	Analog Input 1		rw
0x30F2	RECORD	Analog Input 2		rw

Analog Inputs

Index	0x30F1, 0x30F2
Name	Analog Input
Object Code	RECORD
Number of Elements	7

Value Description

Sub Index	1
Description	Analog Input 16-bit Value Conversion data from ADC. The sampling rate is 16 kHz The result is left aligned.
Data Type	Integer16
Access	ro
PDO Mapping	Yes
Value Range	No
Default Value	No

Sub Index	2
Description	Analog Input 32-bit Value
Data Type	Integer32
Access	ro
PDO Mapping	Yes
Value Range	No
Default Value	No

Analog_Input_32bit_Value = (Analog_Input_16bit_Value - Offset) * Gain / 256
The Gain value is signed.

Example: using analog input as speed reference.
The speed reference is 32-bits, so the 32-bit value will be used.
Let's say that the maximum speed is 30000 rpm and the unit is inc/s with 4096 inc per motor revolution.

Maximum speed: 30000 rpm -> 500 rev/s -> 2048000 inc/s
The maximum 16-bit analog input is 32767
Gain = 2048000 / 32767 * 256 = 16000

Sub Index	3
Description	Offset
Data Type	Integer16
Access	rw
PDO Mapping	Yes
Value Range	-
Default Value	0

Sub Index	4
Description	Gain
Data Type	Integer16
Access	rw
PDO Mapping	Yes
Value Range	-
Default Value	256

Sub Index	5
Description	Filter
Data Type	Unsigned16
Access	rw
PDO Mapping	Yes
Unit	Hz
Value Range	5-20000
Default Value	100

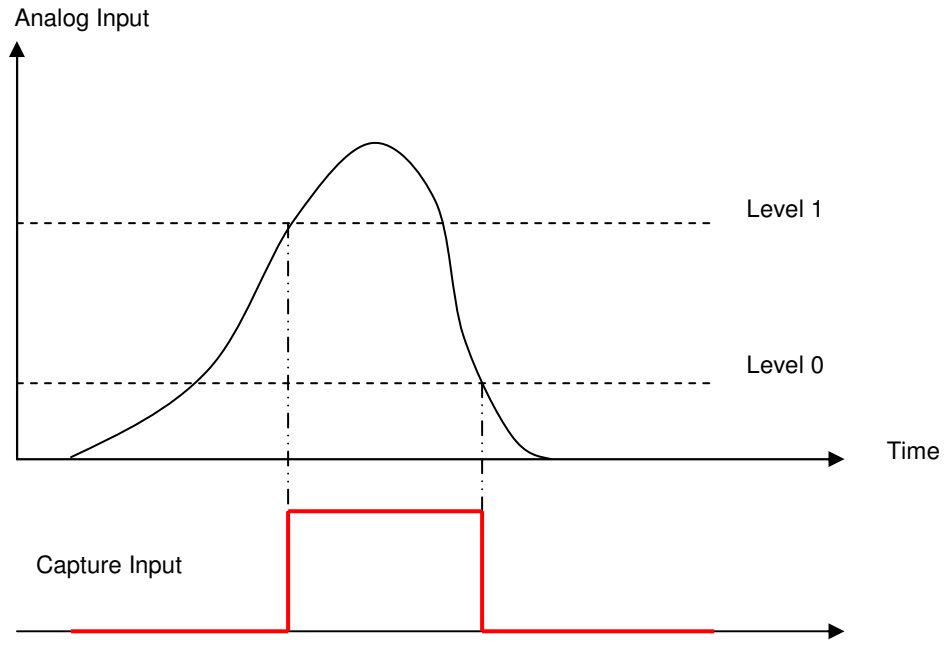
The filter is applied on Analog Input 16-bit Value.

Sub Index	6
Description	Analog In Level 0
Data Type	Integer16
Access	rw
PDO Mapping	No
Value Range	
Default Value	

This parameter defines level 0 for position capture with analog input (see diagram below).

Sub Index	7
Description	Analog In Level 1
Data Type	Integer16
Access	rw
PDO Mapping	No
Value Range	
Default Value	

This parameter defines level 1 for position capture with analog input (see diagram below).



The **TT230** drive has 1 analog output:

- pwm techniques at 48 kHz
- output sampling at 2 kHz
- output signal can be connected to any variable

Object definitions

Index	Object	Name	Type	Attr.
0x30A1	RECORD	Analog Output		rw

Analog Output

Index	0x30A1
Name	Analog Output
Object Code	RECORD
Number of Elements	4

Value Description

Sub Index	1
Description	Analog Output
Data Type	Integer16
Object Class	all
Access	ro
PDO Mapping	Yes

This object monitors the output value.
Output value is from -32768 to 32767 for 0V to 5V on physical analog output.

Sub Index	2
Description	Index/sub-index of Analog Output source
Data Type	Unsigned32
Object Class	all
Access	rw
PDO Mapping	No
Value	See below
Default value	0x30F80100 0x30F80200

This object allows to connect any dataflow as input source of the Analog Output module.

The structure of the entries is the following:

MSB		LSB
Index (16-bit)	Sub-index (8-bit)	0

The output value is defined by:

Analog_Output = (Source_signal + Analog_Output_Offset) * Analog_Output_Gain / 256

Sub Index	3
Description	Analog Output Offset
Data Type	Integer16
Object Class	all
Access	rw
PDO Mapping	No
Default Value	0

Sub Index	4
Description	Analog Output Gain
Data Type	Integer16
Object Class	all
Access	rw
PDO Mapping	No
Default Value	0x0100

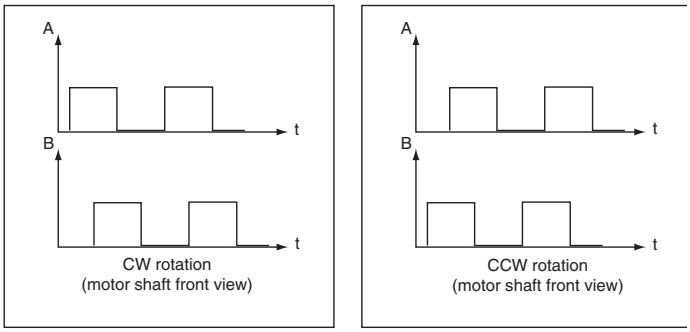
3.2.5.3 - Encoder Emulation Output

The **TT230** version "CAN" has en encoder emulation output.

"Incremental Encoder" module features:

- emulates an incremental encoder output with the resolver position or the encoder position.
- sends any value from a different of 2 variables to the incremental output
- output signal as quadrature signals or pulse/dir signals

Two A and B channels in quadrature with one Z marker pulse per revolution allow to close the position loop via the DNC.



The **Output encoder resolution** parameter is chosen according to following table:

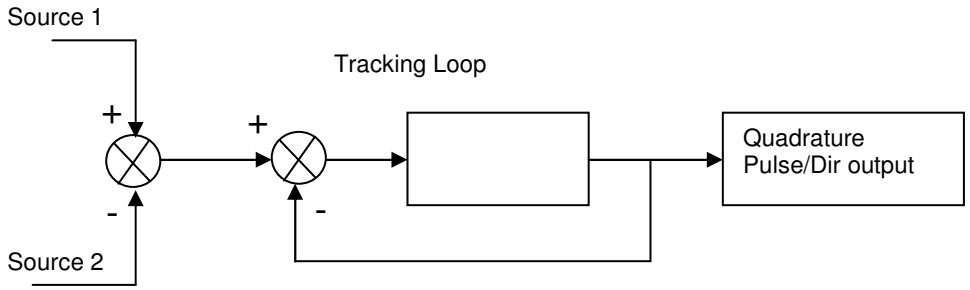
Maximum motor speed (rpm)	up to 1600	up to 3200	up to 6400	up to 12800	up to 25000
Encoder output resolution (ppr)	512 to 16384	512 to 8192	512 to 4096	512 to 2048	512 to 1024

Object definitions

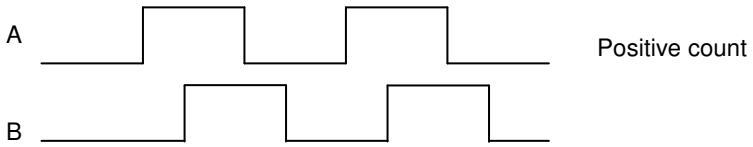
Index	Object	Name	Type	Attr.
0x3160	RECORD	Incremental Encoder Output		rw

Encoder Output

Structure of the Encoder output:



Quadrature output:



The top Z width is 1/4 of signal period.

Index	**0x3160**
Name	Encoder Output
Object Code	RECORD
Number of Elements	6

Value Description

Sub Index	1
Description	Index/sub-index of input source 1
Data Type	Unsigned32
Object Class	all
Access	rw
PDO Mapping	No
Value	See below
Default value	0x31000400

This object allows to connect any dataflow as input source of the Encoder Output module.

The structure of the entries is the following:

MSB		LSB
Index (16-bit)	Sub-index (8-bit)	0

Sub Index	2
Description	Index/sub-index of input source 2
Data Type	Unsigned32
Object Class	all
Access	rw
PDO Mapping	No
Value	See above
Default value	0

If value is 0, source 2 is not connected.

Sub Index	3
Description	Encoder Output Resolution
Data Type	Unsigned32
Object Class	all
Access	rw
PDO Mapping	No
Default Value	0x400

Sub Index	4
Description	Encoder Output Deadband
Data Type	Unsigned16
Object Class	all
Access	rw
PDO Mapping	No
Unit	Same as input source signal (Encoder Output Resolution x 4)
Default Value	0

Sub Index	5
Description	Encoder Output Top Z shift
Data Type	Unsigned16
Object Class	all
Access	rw
PDO Mapping	No
Unit	65536 correspond to an encoder revolution
Default Value	0

Sub Index	6
Description	Encoder Output Configuration
Data Type	Unsigned16
Object Class	all
Access	rw
PDO Mapping	No
Default Value	1

Bit Number	**Function**
0	0 Encoder Output disable 1 Encoder Output Enable
1	1 Encoder Emulation emulates an encoder output with "Encoder Output Resolution" 0 direct output the value of (Source1 - Source2) the value of "Encoder Output Resolution" has no effect
3	reserved
6	reserved
7	reserved, must be 0
9	Physical A-line: 0 A input 1 A output
10	Physical B-line: 0 B input 1 B output
11	Physical Z-line: 0 Z input 1 Z output
12	0 Quadrature output 1 Pulse/Dir output
15	reserved, must be 0

When the "Encoder Emulation" bit is set, a scaling of the input variable (reference by sub index 1) as follows:

input value from 0 to 0xFFFF is scaled to output value of 0 to (resolution x 4)

Only the lower 16-bit of the input value is processed.

If the "Encoder Emulation" bit is cleared, output value = input value.

Example: Encoder Output Emulation with resolver value.

```
0x3160,6 = 0           ; disables encoder output
0x3160,1 = 0x31000400 ; connects encoder output source to the resolver 16-bit value
0x3160,2 = 0
0x3160,3 = 1024        ; resolution : 1024
0x3160,6 = 0x0E03      ; enables encoder output
```

To emulate the encoder output with An Encoder Input, just set 0x3160,1 = 0x31200400.

3.2.5.4 - Digital Cam

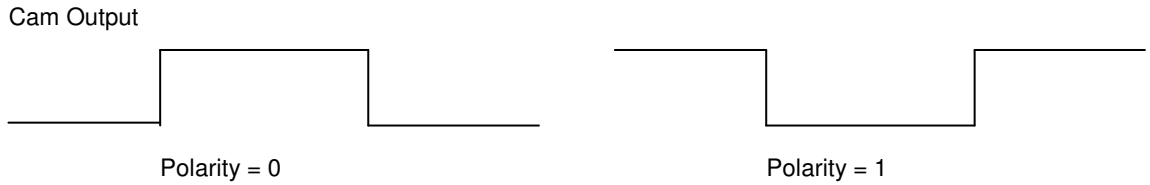
Digital Cam

Index	Object	Name	Type	Attr.
0x30E0	ARRAY	Digital Cam positions	Integer32	rw
0x30E1	ARRAY	Digital Cam configuration register	Unsigned16	rw

Cams are fully defined by objects 0x30E0 and 0x30E1. No parameter can be changed if Cam Enable Register is not 0.

Cam Polarity

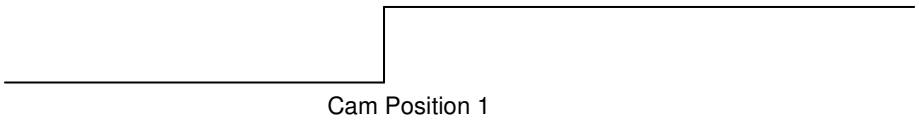
Each bit of the Cam Polarity Register allows to set the polarity of the cam output. Normal polarity (polarity bit = 0) sets the cam output with value 1 when the cam is active.



Cam Type

Each bit of the Cam Type Register defines the cam type.

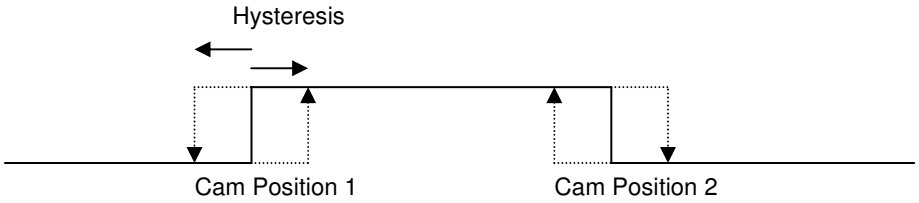
Cam Type = 0: Cam defined by 1 position.



Cam Type = 1: Cam defined by 2 positions.

Cam Hysteresis

Cam Hysteresis Register defines an hysteresis of the cam position.



Digital Cam Positions

Index	**0x30E0**
Name	Digital Cam Positions
Object Code	ARRAY
Number of Elements	32

Digital Cam Positions can only be changed when Cam Enable Register = 0 (0x30E1-5).

Value Description

Sub Index	1
Description	First Position of Cam number 1
Data Type	Integer32
Access	rw
PDO Mapping	Yes
Value Range	No
Default Value	No

Sub Index	2
Description	Second Position of Cam number 1
Data Type	Integer32
Access	rw
PDO Mapping	Yes
Value Range	No
Default Value	No

Digital Cam Configuration Registers

Index	**0x30E1**
Name	Digital Cam Configuration Registers
Object Code	ARRAY
Number of Elements	32

Registers with sub-indexes 2 to 4 can only be changed when Cam Enable Register = 0.

Value Description

Sub Index	1
Description	Cam Status
Data Type	Unsigned16
Access	ro
PDO Mapping	Yes
Value Range	No
Default Value	No

Each bit of Cam status register corresponds to a Digital Cam (max. 16 cams)

Sub Index	2
Description	Cam Type
Data Type	Unsigned16
Access	rw
PDO Mapping	Yes
Value Range	No
Default Value	0

Each bit of Cam Type register corresponds to a Digital Cam (max. 16 cams)

0 Cam with 1 position
1 Cam with 2 positions

Sub Index	3
Description	Cam Polarity
Data Type	Unsigned16
Access	rw
PDO Mapping	Yes
Value Range	No
Default Value	0

Each bit of Cam Polarity register corresponds to a Digital Cam (max. 16 cams)

0 Cam with normal polarity
1 Cam with reversed polarity

Sub Index	4
Description	Cam Hysteresis
Data Type	Unsigned16
Access	rw
PDO Mapping	Yes
Value Range	No
Unit	position unit
Default Value	0

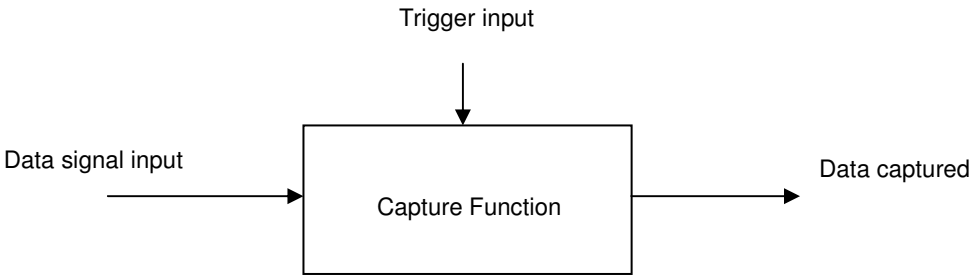
Sub Index	5
Description	Cam Enable
Data Type	Unsigned16
Access	rw
PDO Mapping	Yes
Value Range	No
Default Value	0

Each bit of Cam Polarity register corresponds to a Digital Cam (max. 16 cams)

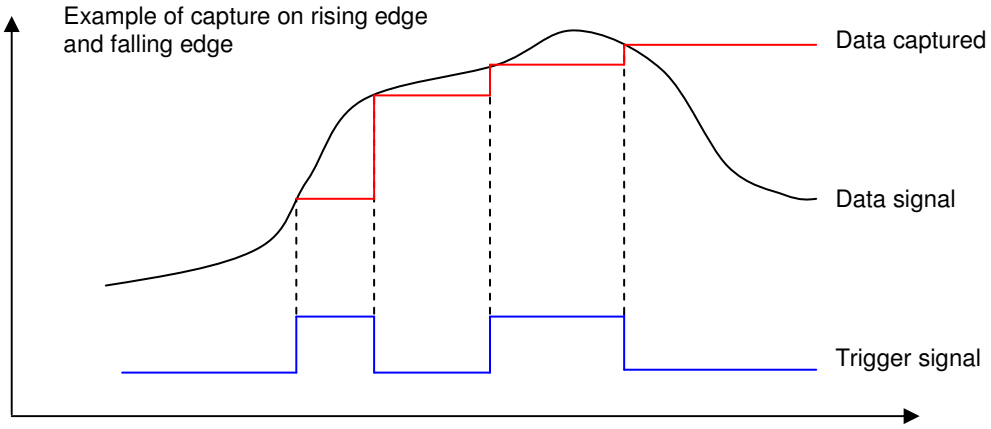
0 Disable Cam
1 Enable Cam

3.2.5.5 - Capture

Capture Function



The purpose of the capture function is to latch a data signal (generally position value from a sensor) on a trigger input signal (generally a logic input).

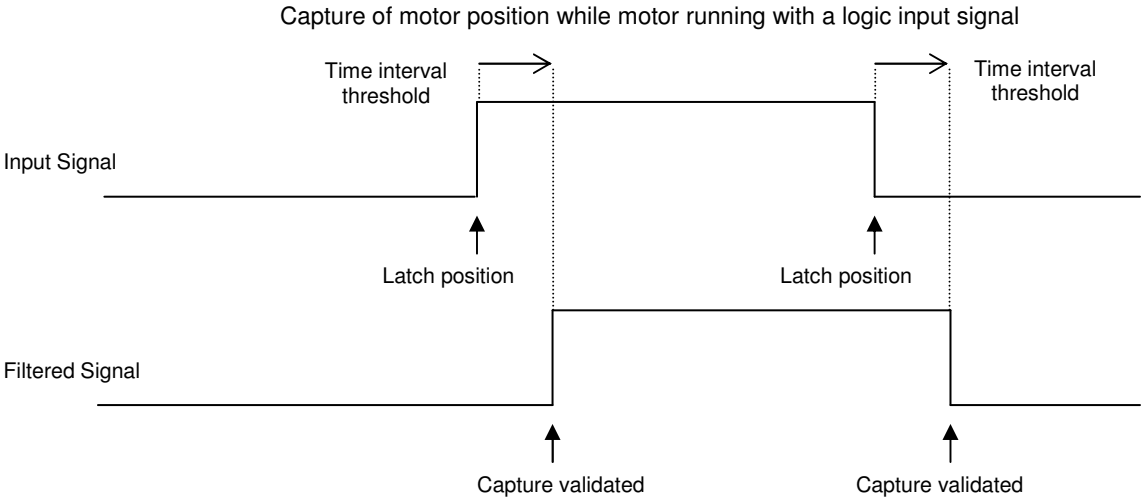


The ServoPac capture features:

- The data signal can be a resolver position value or an encoder position value,
- The trigger input signal can be any of the physical logic inputs, any of the analog inputs or the encoder marker Z,
- The capture can be triggered on rising edge, falling edge or both.
- The trigger input signal can be filtered by a time filter,
- The data signal can be filtered by a space filter.

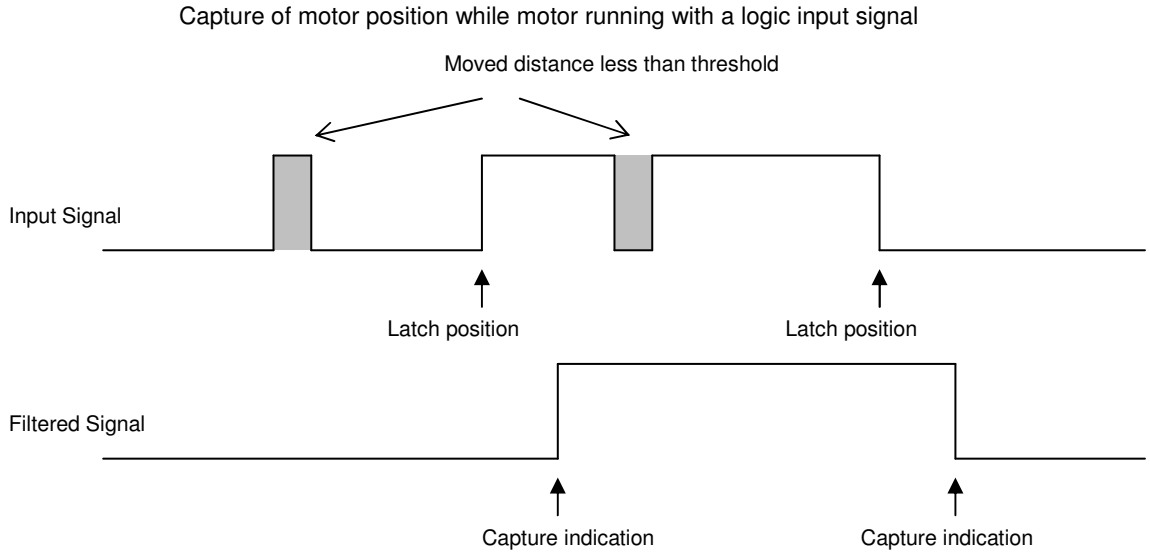
Capture Time Filter

This parameter defines the time interval threshold of the capture time filter. After the rising or the falling edge of the input signal, the input signal level must be stable for a time interval value greater than or equal to the time interval threshold defined by object 0x3371-4 (0x3372-4) in order to get the position capture validated as described below.



Capture Space Filter

This parameter defines the value in distance threshold of the capture position filter. If the position gap between rising and falling edges is less than the threshold, then the signal is the following:



Objects definition

Index	Object	Name	Type	Attr.
0x3370	VAR	Capture Status	Unsigned16	ro
0x337F	VAR	Capture Status for TPDO	Unsigned16	ro
0x3371	RECORD	Capture 1		rw
0x3372	RECORD	Capture 2		rw

Capture Status

Index	0x3370
Description	Capture Status
Data Type	Unsigned16
Access	ro
PDO Mapping	Possible
Value	-
Default value	-

Bit Number	Function
0	
1	Capture Input 1 image
2	change state: a capture on rising edge of input 1 occured
3	change state: a capture on falling edge of input 1 occured
4	
5	Capture Input 2 image
6	change state: a capture on rising edge of input 2 occured
7	change state: a capture on falling edge of input 2 occured
8	
9	Capture Input 3 image
10	change state: a capture on rising edge of input 3 occured
11	change state: a capture on falling edge of input 3 occured
12	
13	Capture Input 4 image
14	change state: a capture on rising edge of input 4 occured
15	change state: a capture on falling edge of input 4 occured

The Capture Status is clear when writing to Capture configuration (0x337n-1)

Capture Status for PDO

Index	0x337F
Description	Capture Status for PDO
Data Type	Unsigned16
Access	ro
PDO Mapping	Possible
Value	-
Default value	-

Bit Number	Function
0	
1	Capture Input 1 image
2	A capture on rising edge of input 1 occured
3	A capture on falling edge of input 1 occured
4	
5	Capture Input 2 image
6	A capture on rising edge of input 2 occured
7	A capture on falling edge of input 2 occured
8	
9	Capture Input 3 image
10	A capture on rising edge of input 2 occured
11	A capture on falling edge of input 2 occured
12	
13	Capture Input 4 image
14	A capture on rising edge of input 2 occured
15	A capture on falling edge of input 2 occured

Capture indicators (bit 2, 3, 6, 7, 10, 11, 14, 15) are cleared when this object is sent by a PDO.

Capture Parameters

Index	**0x3371** for capture 1 **0x3372** for capture 2 **0x3373** for capture 3 **0x3374** for capture 4
Name	Capture Parameters
Object Code	RECORD
Number of Elements	8

Value Description

Sub Index	1
Description	Capture 1/2/3/4 Config
Data Type	Unsigned16
Access	rw
PDO Mapping	No

Bit Number	**Function**
0	Capture on rising edge
1	Capture on falling edge
15	Enable Capture

Sub Index	2
Description	Capture 1/2/3/4 source
Data Type	Unsigned32
Access	rw
PDO Mapping	No

This parameter allows to connect a 32-bit dataflow as input of the capture data signal.

Only objects 0x3109 (resolver position) and 0x3129 (encoder position) are supported.

The structure of the entries is the following:

MSB		LSB
Index (16-bit)	Sub-index (8-bit)	0

Example:
Capture 1 data is connected to resolver position:
0x3371,2 = 0x31090000

Sub Index	3
Description	Capture 1/2/3/4 Input
Data Type	Unsigned16
Access	rw
PDO Mapping	No

This parameter allows to define a logic input as capture trigger signal.

Value	Function
0	IN1
1	IN2
2	IN3
3	IN4
4	IN5
5	
6	
7	
8	
9	
10	
11	
12	Analog In 1
13	Analog In 2
14	Encoder Top Z
15	

IN1 .. IN9 are physical inputs.
The capture triggered by the analog input is defined by analog levels (0x30F1).

Sub Index	4
Description	Capture Time Filter
Data Type	Unsigned16
Access	rw
PDO Mapping	No
Unit	

Sub Index	5
Description	Capture Position Filter
Data Type	Unsigned32
Access	rw
PDO Mapping	No
Unit	Position unit

Sub Index	6
Description	Capture Position
Data Type	Integer32
Access	ro
PDO Mapping	Yes
Unit	

Sub Index	7
Description	Rising Edge Capture Position
Data Type	Integer32
Access	ro
PDO Mapping	Yes
Unit	

Sub Index	8
Description	Falling Edge Capture Position
Data Type	Integer32
Access	ro
PDO Mapping	Yes
Unit	

3.2.5.6 - Modulo function

Index	0x307B
Name	Modulo configuration
Object Code	VAR
Data Type	Unsigned16
Object Class	all
Access	rw
PDO Mapping	No
Default Value	0

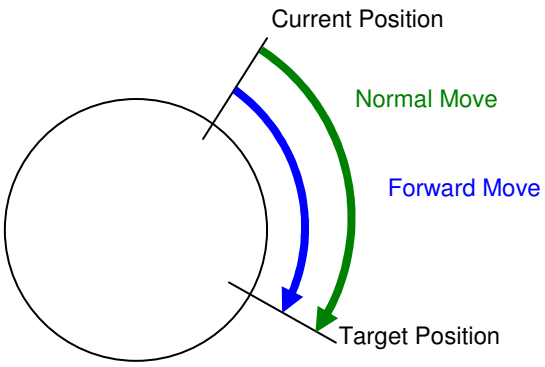
The motor position can be limited by the position limit function (modulo function).

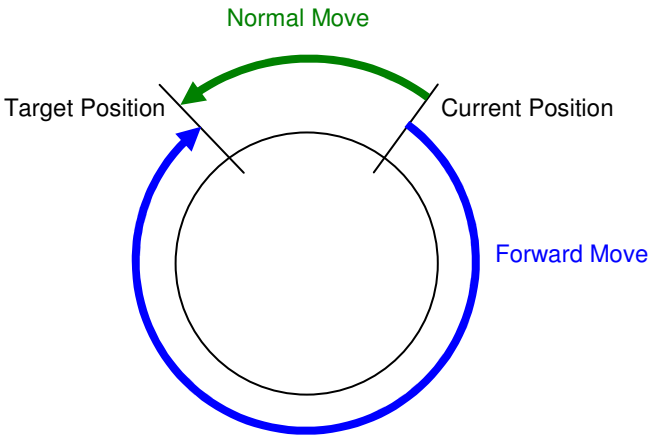
Minimum Position Limit <= Motor Position < Maximum Position Limit

Bit Number	Function
0	Modulo Function 0 disable 1 enable
2	Forward (always in positive direction)
3	Backward (always in negative direction)
4	CLEAR input function 0 disable 1 enable

- "Forward" and "Backward" cannot be set at same time.
- "Modulo Enable/Disable" (bit 0) and CLEAR input function (bit 4) can not be changed when drive is enabled.

Modulo Function with forward:
The forward bit forces the motor to move always in positive direction.





CLEAR input function:

The CLEAR input function allows to use the HOME input (0x60FD) to reset the position value.

CLEAR input function and modulo function must not be activated at the same time.

The motor position can be limited by the position limit function or modulo function.

The modulo function is enabled / disabled by object 0x307B.

Minimum Position Limit <= Motor Position < Maximum Position Limit

The Position Limit values are defined by object 0x607B. These position values can only be changed when the modulo function is disabled.

Index	**0x607B**
Name	Position Limit
Object Code	ARRAY
Object Class	all
Number of Elements	2

Value Description

Sub Index	1
Description	Minimum Position Limit
Data Type	Integer32
Access	rw
PDO Mapping	No
Unit	User position unit
Value	

Sub Index	2
Description	Maximum Position Limit
Data Type	Integer32
Access	rw
PDO Mapping	No
Unit	User position unit
Value	

3.2.6 - MAINTENANCE

3.2.6.1 - Files

TT230 Files

The **TT230** drive can store data files in its internal Flash memory:

Name	**Format**	**Description**
DRIVEPAR.TXT	text object file	Drive parameters are saved in these files. The user can save drive parameters by means of the ServoPac Gem Drive Studio software or via the communication bus by means of object 0x1010 (CAN bus, RS-232...)
USER_PAR.TXT	text object file	This file can keep extra parameters by the user. The parameters are set manually and the USER_PAR.TXT file must be sent to the **TT230** drive.
SEQUENCE.TXT	text sequence file	Sequence files for Positioner Mode

Object File

Object file format

The object file (i.e. CANopen object) is a plain text file allowing to define an object list in the drive, which values must be defined.

The syntaxis is:
```
index,sub=object_value
```

All digital values can be in hexa (preceded by 0x) or decimal.
Only one allocation per line is allowed.
A comment line begins with a ;
All lines that do not begin with a figure will be ignored.

Example:

```
0x3549,10=0x12
```
means the allocation of value 0x12 to object index 0x3549 sub-index 10

```
13641,0xA=18
```
gives the same result.

Notes

- The drive parameter file (DRIVEPAR.TXT) has also got this format.

- The USER_PAR.TXT file is not mandatory. It allows, for example, to define an initial configuration of the drive directly by the user.

3.2.6.2 - Firmware update

Update File

An Update File contains a file header and one or several data blocks.

File_Header
Binary_Block_1
Binary_Block_2
...
Binary_Block_n

File_Header (32 bytes):
```
00000000 File_code 'IDUF' (0x46554449)
00000004 File_crc32: from byte 4 to the end of file
00000008 Protect Data length (bytes): file_length - 8
0000000C Device Sectors
00000010 Update_Code
00000012 Number of Binary Blocks
00000014 Number of Block Type
00000016 Version
00000018 Device Address
0000001C reserved
00000020 First Binary Block
```

Binary_Block_k: Block_Header + Block_Data
Block header (16 bytes):
```
00000000 Block_crc32: from byte 4 to the end of block
00000004 Block_type: 1-algo, 2-security, 3-code
00000006 Block_Cmd
00000008 Block_addr: Device memory address
0000000C Block_length: length of block data (bytes)
00000010 Block data...
```

Update Interface

General Commands

Index	0x5F30
Name	Update
Object Code	RECORD
Number of Elements	5

Value Description

Sub Index	1
Description	Update_Code
Data Type	Unsigned16
Access	rw
Value	Write: Select firmware_Code (> 0) Read back: same code = Update_Code supported, 0 if not supported

Sub Index	2
Description	Update_Mode
Data Type	Unsigned16
Access	rw
Value	Write signature: 0x00000001 Change to update mode, Update_Code must be <> 0 depend on Update_Code, the execution time of this instruction may be very long: for example: update firmware -> switch from firmware mode to bootmanager mode

Update Init

Index	0x5F31
Name	Update Init
Object Code	RECORD
Number of Elements	5

Sub Index	1
Description	Number of Binary Blocks
Data Type	Unsigned16
Access	rw
Value	

Sub Index	2
Description	Number of Block Types
Data Type	Unsigned16
Access	rw
Value	

Sub Index	3
Description	Sectors
Data Type	Unsigned32
Access	rw
Value	Each bit = 1 sectors support up to 32 sectors

Sub Index	4
Description	Erase Command
Data Type	Unsigned32
Access	rw
Value	Signature = 0x00000001 the execution time of this instruction is very long

Block process

Index	**0x5F32**
Name	Block process
Object Code	RECORD
Number of Elements	5

Value Description

Sub Index	1
Description	Block_Type
Data Type	Unsigned16
Access	rw
Value	defines the Block_Type of data bock

Sub Index	2
Description	buffer_Size (read-only)
Data Type	Unsigned32
Access	ro
Value	gives the buffer size (bytes) for current block (depends on block_type)

Sub Index	*3*
Description	*Current_sector (read-only)*
Data Type	*Unsigned32*
Access	*ro*
Value	

Sub Index	*4*
Description	*Current_address (read-only)*
Data Type	*Unsigned32*
Access	*ro*
Value	

Sub Index	5
Description	Buffer (segmented)
Data Type	Unsigned32
Access	rw
Value	transfer data to/from buffer: binary_block (block_header + block_data)

Programming sequence

Initialization:		
Update Code:	write	0x5F30,1 = update_code
Change to program mode:	write	0x5F30,2 = 1
Verify program mode:	read	0x5F30,2 = 1
Erase:		
Number of Blocks:	write	0x5F31,1 = n_blocks
Number of Block_type:	write	0x5F31,2 = n_types
Sectors Mask:	write	0x5F31,3 = sectors_mask
Erase command:	write	0x5F31,4 = 1
Verify erase command:	read	0x5F31,4 = 0
Programming: repeat (n blocks)		
Write Block_type:	write	0x5F32,1 = block_type
Write Block (header & data):	write seg	0x5F32,5
Program Block command:	write	0x5F32,6 = 1
Verify program command:	read	0x5F32,6 = 0

3.3 - OBJECT LIST

parameters in bold are saved in the parameter file.

Index	Sub	Name	Description

Communication

0x1005		Sync_ID	Sync CobID
0x1006		**Period**	Communication Cycle Period
0x100C		Guard_T	NodeGuarding Guard Time
0x100D		LifeTime	NodeGuarding Life time factor
0x1014		Emcy_ID	Emcy CobID
0x1016		HeartBt	Consumer Heartbeat Time
0x1017		HBprod	Producer Heartbeat Time
0x1018		Identity	CANopen Identity object
0x1200		SrvSDO	Server SDO parameter
0x1201		SrvSDO2	Server SDO 2 parameter
0x1280		CliSDO1	Client SDO 1 parameter
0x1281		CliSDO2	Client SDO 2 parameter
0x1400		RPDO1par	RPDO1 parameter
0x1401		RPDO2par	RPDO2 parameter
0x1402		RPDO3par	RPDO3 parameter
0x1403		RPDO4par	RPDO4 parameter
0x1600		RPDO1map	RPDO1 mapping
0x1601		RPDO2map	RPDO2 mapping
0x1602		RPDO3map	RPDO3 mapping
0x1603		RPDO4map	RPDO4 mapping
0x1800		TPDO1par	TPDO1 parameter
0x1801		TPDO2par	TPDO2 parameter
0x1802		TPDO3par	TPDO3 parameter
0x1803		TPDO4par	TPDO4 parameter
0x1A00		TPDO1map	TPDO1 mapping
0x1A01		TPDO2map	TPDO2 mapping
0x1A02		TPDO3map	TPDO3 mapping
0x1A03		TPDO4map	TPDO4 mapping

0x2000		NMTmastr	NMT Start/Stop
0x2001		NMTstate	NMT state
0x2004		**AxisName**	Axis Name
0x2006		SyncCtrl	Can Synchronisation parameter
	1	**SCphase**	
	2	**SCthresh**	
	3	**SCadjust**	
0x200A		DevAddr	DeviceID
	1	**DeviceId**	
0x2010		**NMTboot**	NMT Boot-up
0x205D		**NMTcfg**	NMT config
0x205E		**NMTerror**	NMT error behaviour
0x205F		**EMCYmsg**	

0x2300		SerialP	RS-232 parameters
	1	**SP_baud**	
	2	SP_data	
	3	SP_par	
	4	SP_stop	
0x2301	**0**	**SP_pro**	RS-232 protocol select
0x2310		**Can_Baud**	Can Baud

General

0x1000		DevType	Device Type
0x1008		DevName	Manufacturer Device Name
0x1009		Hardware	Manufacturer Hardware Version
0x100A		Software	Manufacturer Software Version

0x1010		StorePar	Store parameters
0x1011		LoadPar	Restore parameters

0x6510	0	DrvData	Drive Data
	1	DrvMax	
	2	DrvRated	
	3	DrvVolt	
	4	**UserVolt**	User Voltage (230 or 400)
	5	**LowVolt**	Low Voltage Threshold
0x6502	0	DrvModes	Supported drive modes
0x6504	0	ManName	Manufacturer Name

Device Control

0x6040	0	ControlW	Control Word
0x6041	0	StatusW	Status Word
0x605A	**0**	**QStopOC**	Quick Stop option code
0x605B	**0**	**ShutDnOC**	Shutdown option code
0x605C	**0**	**DisOpOC**	Disable Operation option code
0x305A	**0**	**InhOpOC**	Inhibit option code
0x6060	**0**	**ModeOp**	Mode of Operation
0x6061	0	ModeOpDp	Mode of Operation Display
0x3041		DevState	Device state monitoring
0x3300		StopDec	Stop 1 Ramp
	1	**StopDec1**	
	2	**StopDec2**	
0x3301		StopI	Stop 3 current limit
	1	**StopI1**	
	2	**StopI2**	
0x3302		StopTime	Stop Time Limit
	1	**StopTm1**	
	2	**StopTm2**	
0x6085	**0**	**QS_dec**	Quick Stop Ramp
0x3304	**0**	**DrvTime**	Amplifier Reaction Time
0x3305	**0**	**BrkTime**	Motor Brake Reaction Time

Factor Group

0x608F		**PosResol**	Encoder Position Resolution
0x6093		**Pos1Fact**	Position Factor
0x6089	**0**	**Pos1Nota**	
0x608A	**0**	**Pos1Dim**	
0x3089	**0**	**Pos1Disp**	Position Display Factor
0x308A	**0**	**Pos1Unit**	Position Unit Name

Motor

0x6410			Motor Data
	1	**MotorMan**	
	2	**MotorNm**	
	3	**MotorCod**	
	4	**McatDate**	
	5	**MmodDate**	
	6	**Mtype**	
	7	**Mmaxspd**	
	8	**Mrtdspd**	
	9	**MstallI**	
	10	**MpeakI**	
	11	**M_Kt**	
	12	**M_J**	
	13	**Minduct**	
	14	**Mpolepr**	
	15	**MPhase**	
	16	**Moffset**	
	17	MTtype	
	18	MTthres1	
	19	MTthres2	
	20	**Mpolept**	
0x6072		MaxTq	Max Torque
0x6073		**MaxI**	Motor Max current
0x6075		**MotRtdI**	Motor Rated Current
0x6076		MotRtdTq	Motor Rated Torque

0x3410		MotorPar	Motor Parameters
	1	**PolePair**	Current Number of motor pole pairs
	2	**MotPhase**	Current Motor Phase
	3	**RotorOfs**	Current Motor Offset
0x340F	**0**	**Induct**	Current Motor Inductance
0x3323	0	MT_res	Motor temperature probe monitoring
0x3324		MT_cfg	Motor temperature probe config
	1	**MT_probe**	Motor temperature type (NTC/PTC)
	2	**MT_warn**	Motor temperature warning threshold
	3	**MT_error**	Motor temperature error threshold

Sensors

0x306A	**0**	**Pos_FB**	Position Feedback Sensor Select
0x3070	**0**	**Motor_FB**	Motor Feedback Sensor Select

Resolver Input

0x3100		Resolver	Resolver monitoring
	1	Res_Sin	
	2	Res_Cos	
	3	Res_Amp2	
	4	Res_Mod	
	5	Res_Amp	
0x3101	0	Res_Setp	Resolver Setup
	1	**Res_Type**	Enable/Setup Resolver Input
	2	**Res_Cfg**	
	3	**Res_Zsh**	
	4	**Res_Zsz**	
	5	**Res_NP**	
	6	**ResRatio**	

0x3102	0	Res_Err	Resolver Error control
	1	Res_Thrs	
	2	Res_Lim	
	3	**Res_AmpF**	
	4	**Res_Rdc**	
0x3104		Res_Cal	Resolver Calibration Procedure
0x3105		Res_CalV	Resolver Calibration parameters
0x3107	0	Res_TopZ	Resolver Virtual Top Z
0x3108	0	Res_ofs	Resolver Offset (user position unit)
0x3109	0	Res_pos	Resolver Position (user position unit)
0x310A	0	Res_vel	Resolver Velocity (user velocity unit)

Encoder Input

0x3120		Encoder1	Encoder
	1	Enc1Sin	
	2	Enc1Cos	
	3	Enc1Amp2	
	4	Enc1Mod	
	5	Enc1Amp	
0x3121		Enc1Setp	Encoder Setup
	1	**Enc1Type**	
	2	**Enc1Cfg**	
	3	**Enc1Zsh**	
	4	**Enc1Zsz**	
	5	**Enc1res**	
	6	**Enc1turn**	
	7	**Enc1Zlen**	
0x3122		Enc1Err	Encoder Error Control
	1	**Enc1Cnt**	
	2	**Enc1Thrs**	
	3	**Enc1Lim**	
0x3124		Enc1CalP	Encoder Calibration
0x3127	0	Enc1TopZ	Encoder Virtual Top Z
0x3128	**0**	**Enc1ofs**	Encoder Offset (user position unit)
0x3129	0	Enc1pos	Encoder Position (user position unit)
0x312A	0	Enc1vel	Encoder Velocity (user velocity unit)
0x312B	0	Enc1Ref	
	1	Enc1RefP	
0x312D	0	Enc1Abs	
	1	Enc1Max0	
	2	Enc1Max1	
	3	Enc1Abs0	
	4	Enc1Abs1	
	5	**Enc1Ref0**	
	6	**Enc1Ref1**	
0x313E	**0**	**Enc1HesC**	

Servo Loops

Current Loop

0x3400		Imon	Motor Current Monitoring
0x3402		Iofs	Motor Current offset measurement
0x3408		Vdcmon	Voltage monitoring
0x30DA		**IlimSrc**	Dynamic Current Limit Input Source
0x30D1		Ilimit	Current Limitation
0x30D2		**IlimCfg**	Dynamic Current Limit Configuration
0x30D4		Iq	Iq Current monitor
0x30D5		Id	Id Current monitor

0x3411	0	CalcIlp	Current Loop Calculation
0x3412	0	CalcIlim	Current Limitation Calculation
0x60F6		Tq_CTRL	Current Loop Parameters
	1	**IregType**	
	2	**KPq_I**	
	3	**KIq_I**	
	4	**KPd_I**	
	5	**KId_I**	
0x30F5		TqLpmon	Current Loop Monitoring
	1	IdRef	
	2	IqRef	
	3	Idmon	
	4	Iqmon	
	5	VdRef	
	6	VqRef	
	7	PosElec	
0x6079	0	DCvolt	DC Voltage
0x30F4		IdrvLim	Current limit parameters

0x3413		APstart	Autophasing

0x3414		MCstart	Motor phasing

Speed Loop

0x60F9		Vel_CTRL	Speed Loop Parameters
	1	**VregType**	
	2	**KPv**	
	3	**KIv**	
	4	**KIvf**	
	5	**KCv**	
	6	**KDv**	
	7	**KDvf**	
	8	**KJv**	
0x30F9		VFilter	Speed Error Low-pass Filter
	1	**SpErrLF1**	
	2	**SpErrLF2**	
	3	**SpErrLF3**	
0x30FA	**0**	**TVelMes**	Speed measurement filter
0x30F8		VelLpmon	Speed Loop Monitoring
	1	VelRef	Demand speed (0x7FFF -> Maximum motor speed)
	2	VelFb	Motor speed (0x7FFF -> Maximum motor speed)
	3	VelErr	
	4	Idc	
	5	IcomF	

Position Control

0x307B	**0**	**PosRgEna**	Modulo configuration
0x607B		PosRange	Position Limit
	1	**PosRgMin**	
	2	**PosRgMax**	
0x60FB		Pos_CTRL	Position Control Parameters Set
	1	**PregType**	
	2	**KPp**	
	3	**KFp**	
	4	**KAv**	
	5	**KBv**	
0x30FC		PosLpmon	Pos Loop monitoring
	1	PosRef	
	2	PosFB	
	3	Vref	
0x6062		PosDem	Pos Demand Value

0x60B0		PosOfs	Pos Offset
0x6063		IntPos	Position Actual Value
0x6064		ActPos	Actual position
0x6065		**PosErWin**	Following Error Window
0x3065		**FWctrl**	Following Error Error control
0x60F4		PosErr	Following Error Actual Value

0x3425	0	Autotune	Autotuning parameters
	1	**ATbwidth**	
	2	**ATtype**	
	3	**ATselect**	
	4	**ATappl**	
0x3426	0	ATstart	Autotuning
0x3427	**0**	**KsDig**	

Error Control

0x3022	0	Error	Error monitoring
	1	Error1	
	2	Error2	
	3	Error3	
0x3023	0	ErrCode	
	1	ErrState	
	2	LastErr	
	3	PrevErr	
0x3024	0	Warning	Warning
0x3025	0	Err_Ctrl	Error control (mask)
	1	**ErrMask1**	
	2	**ErrMask2**	
	3	**Stop1Mk1**	
	4	**Stop1Mk2**	
	5	**Stop3Mk1**	
	6	**Stop3Mk2**	

0x3404	0	Iprotect	I²t monitoring/parameter
	1	**I2tMode**	
	2	I2t	
	3	Imotor	

Profile Position Mode

0x607A	0	TargePos	Target Position
0x6080	**0**	**MaxSpeed**	Maximum motor speed
0x6081	**0**	**ProfiVel**	Profile Velocity
0x6082	0	PPendVel	End Velocity
0x6083	**0**	**ProfiAcc**	Profile Acceleration
0x6084	**0**	**ProfiDec**	Profile Deceleration
0x6086	**0**	**ProfType**	Motion Profile Type
0x6067	**0**	**PosWindo**	Position Window
0x6068	**0**	**PosWinTi**	Position Window Time
0x607D	0	PosLimit	Software Position Limit
	1	**MinPosLm**	Minimum position Limit
	2	**MaxPosLm**	Maximum position Limit
0x607F		**MaxPPvel**	Max Profile Velocity
0x3360	**0**	**AxeType**	Axis Type
0x3081	**0**	**SpModSrc**	Position Profile Speed Modulation Input Source
0x3082	**0**	**SpModCfg**	Position Profile Speed Modulation Configuration

Homing Mode

0x607C		**HomeOfs**	Home Offset
0x6098		**HomeMeth**	Homing Method
0x6099		HomeSpds	Homing Speeds
	1	**HomeSpd1**	Speed during search of switch
	2	**HomeSpd2**	Speed during search of zero
0x609A		**HomeAcc**	Homing Acceleration
0x309C		**HCurLim**	Home Current Limit
0x309D		**HEndHome**	End On Home Position

Interpolated Position Mode

0x60C0		**IPmode**	Interpolated SubMode Select
0x60C1		IPrecord	Interpolated Data Record
0x30C1		IPoutput	Interpolation output
0x60C4		IP_conf	Interpolation data configuration

Profile Velocity Mode

0x60B1	0	VelOfs	Offset Velocity
0x30B1	0	VelOfsSc	Offset Velocity input source
0x60FF	0	TargetV	Target Velocity
0x606B	0	VelDem	Velocity Demand Value
0x606C	0	VelAct	Velocity Actual Value
0x306C	**0**	**VelFilt**	Velocity measurement filter
0x3069	0	Velocity	Velocity Actual Value (rpm)
0x606D	**0**	**VelWin**	Velocity Window
0x606E	**0**	**VelWinTm**	Velocity Window Time
0x606F	**0**	**VelThr**	Velocity Threshold
0x6070	**0**	**VelThrTm**	Velocity Threshold Time
0x30FF	**0**	**VelInObj**	Target Velocity Input Object

Profile Torque Mode

0x6071	0	TqTarget	Target Torque
0x3071	0	TqSrc	Target Torque input source
0x6087	**0**	**TqSlope**	Torque Slope
0x6088	0	TqProfil	Torque profile type
0x60B2	0	TqOffset	Offset Torque
0x30B2	0	TqOfsSrc	Offset Torque input source
0x6074	0	TqDemand	Torque Demand Value
0x6077	0	TqValue	Torque Actual Value
0x6078	0	CurrAct	Current Actual Value
0x3078	0	CurrFilt	Current measurement filter

Sequence Mode

Sequence Control

0x3601		SQin	Sequence Inputs
	1	SQnb	Sequence Number Input
	2	SQrun	Executed Sequence Number
	3	SQcond	Conditional Input
0x3602		SQoutp	Sequence Outputs
	1	SQout	Programmable Logic Outputs
	2	SQoutpol	Programmable Logic Outputs Polarity
	3	SQsta	Dedicated Logic Outputs
	4	SQstapol	Dedicated Logic Outputs Polarity
0x3603	**0**	**SQSpulse**	Minimum Sequence Pulse
0x3604		SQoutcfg	Output Pulse Configuration
	1	**SQOpulse**	Output Pulse
	2	**SQOtime**	Output Pulse Duration
0x3605	0		Sequence phase
0x3606	0		Sequence Position Setpoint value
0x3609	0		Sequence Position Offset
0x360A	0		Sequence Position Output
0x360B	0		Sequence position capture
0x360C	0	SQconfig	Sequence Configuration
0x360F	0	SQavail	Supported Sequence Type

Sequence Parameters

0x3610	0	SQParNb	Sequence Parameters Number
0x3611	0	SQPar	Sequence Parameters
	1	SQPtype	Sequence Type
	2	SQPnext	Next sequence
	3	SQPcnt	Sequence Counter
	4	SQPlink	Sequence Link
	5	SQPtrig	Output Trigger
	6	SQPout0	Output Bits = 0
	7	SQPout1	Output Bits = 1
	8	SQPoutT	Output Bits Toggle
	9	SQPst0	Start Condition Bits = 0
	10	SQPst1	Start Condition Bits = 1
	11	SQPstop0	End Condition Bits = 0
	12	SQPstop1	End Condition Bits = 1
	13	SQPpos	Position
	14	SQPpos2	Position 2 (reserved for future use)
	15	SQPvel	Speed
	16	SQPext	Speed 2 / Position 3 (reserved for future use)
	17	SQPaccel	Acceleration
	18	SQPdecel	Deceleration
	19	SQPtacc	Acceleration Time
	20	SQPtdec	Deceleration Time
	21	SQPcfg	Configuration
	22	SQPcfg2	Configuration 2
	23	SQPtempo	Temporization
	24	SQPrtime	Running Time
	25	SQPana	Analog In
	26	SQPana2	Analog In 2 (reserved for future use)

Stepper Emulation Mode

0x3681	0	SE_mode	
	1	SEctrl	
	2	SEstatus	
	3	**SEconfig**	
	4	**SEtempo**	

Analog Speed Mode

0x604F	**0**	**Vramp**	
0x304F	**0**	**Vramp2**	

Application FE

Digital Inputs/Outputs

0x60FD	0	Dinput	Digital Inputs
0x3050		DInpCfg	Digital Inputs Configuration
	n	**Inp?Cfg**	
0x3051	**0**	**InpPol**	Digital Inputs Polarity
0x60FE		Doutput	Digital Outputs
	1	Dout	
	2	**DoutBMsk**	
0x3054		DOutpCfg	Digital Outputs Configuration
	n	**Outp?Cfg**	
0x3055	0	OutpPol	Digital Outputs Polarity

Analog Inputs

0x30F1		AnalogI1	Analog Input 1
	1	AnaIn1	
	2	AI1s32	
	3	**AI1_ofs**	
	4	**AI1_gain**	
	5	**AI1_filt**	
	6	**AI1_lv0**	
	7	**AI1_lv1**	
	8	AI1_proc	
	9	**AI1_db**	
0x30F2		AnalogI2	Analog Input 2
	1	AnaIn2	
	2	AI2s32	
	3	**AI2_ofs**	
	4	**AI2_gain**	
	5	**AI2_filt**	
	6	**AI2_lv0**	
	7	**AI2_lv1**	
	8	AI2_proc	
	9	**AI2_db**	

Analog Output

0x30A1	0	AnalogO1	Analog Output 1
	1	AO1s16	
	2	**AO1src**	
	3	**AO1ofs**	
	4	**AO1gain**	

Encoder Emulation Output

0x3160	0	eOut	Encoder Emulation Output
	1	**eOutSrc1**	
	2	**eOutSrc2**	
	3	**eOut_res**	
	4	**eOut_db**	
	5	**eOut_zsh**	
	6	**eOut_ctl**	
	7	eOut_sta	

Digital Cam

0x30E0		DCamPos	Digital Cam positions
	n	**DCam?P?**	
0x30E1	0	DCamCFg	
	1	**DCamStat**	
	2	**DCamType**	
	3	**DCamPol**	
	4	**DCamHyst**	
	5	**DCamEna**	

Oscilloscope

0x5800	0	Osc_Func	Oscillo function support
0x5804		**Osc_Buf**	Oscillo Buffer configuration
0x5805	0	OscBufDl	Oscillo Buffer delay
0x5810		OscChCfg	Oscillo Channel config
0x5811		**OscChan**	Oscillo Channel definitions
0x5812		OscUnit	Oscillo Channel Unit
0x5820		**OscTgSrc**	Oscillo Trigger configuration
0x5822		**OscTrig**	Oscillo Trigger 1
0x5828	0	OscTgCtl	Oscillo Trigger Control
0x5829	0	OscTgSta	Oscillo Trigger Status
0x5840		**OscTxCfg**	Oscillo Buffer transfer configuration
0x5841	0	OscTx	Oscillo Buffer transfer

Firmware Update

0x5F30		UpdtDrv	Update Firmware
0x5F31		UpdtInit	Update init
0x5F32		UpdtProc	Update process